LEASY V6.2A

Programmschnittstelle und Konzepte

Ausgabe März 2007

Kritik… Anregungen… Korrekturen…

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Sie können uns Ihre Kommentare per E-Mail an manuals@fujitsu-siemens.com senden.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2000

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001:2000 erfüllt.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright und Handelsmarken

Inhalt

1 Einleitung

1.1 Kurzbeschreibung des Produkts

LEASY ist ein im BS2000 ablauffähiges, transaktionsorientiertes Datenverwaltungs- und Zugriffssystem.

Es bietet ein Sicherheitskonzept zur Erhaltung der Konsistenz von Dateien.

LEASY unterstützt folgende Anforderungen:

- Einfacher und einheitlicher Zugriff auf DVS-Dateien
- Sekundärschlüssel
- Transaktionen
- Datensicherheit.

Der Zugriff kann aus COBOL- oder Assembler-Programmen erfolgen. Die Schnittstelle entspricht KLDS, der Norm für kompatible Schnittstellen zu linearen Datenbanksystemen.

LEASY ist einsetzbar im Teilnehmerbetrieb (TIAM/Batch) und im Teilhaberbetrieb (openUTM, DCAM).

1.2 Zielsetzung und Zielgruppen des Handbuchs

Dieses Handbuch richtet sich an Organisatoren, Entwickler und Administratoren von LEASY-Anwendungen.

Welche Teile des Handbuchs für welche Personengruppen von Bedeutung sind, wird auf der folgenden Seite beschrieben. Allgemein gehen wir von den folgenden Voraussetzungen aus:

– Für die Verwendung von LEASY sind Kenntnisse des BS2000 inklusive des Datenverwaltungssystems erforderlich.

– Für die Bedienung von LEASY über die COBOL-Schnittstelle sind Kenntnisse der Programmiersprache COBOL, für die Bedienung über die Assembler-Schnittstelle Kenntnisse des Makroassemblers sowie der BS2000-Systemmakros Voraussetzung.

– Bei Einsatz von LEASY mit openUTM sind openUTM-Kenntnisse, bei Einsatz in DCAM-Umgebung sind DCAM-Kenntnisse und bei Einsatz in Mehrrechnersystemumgebung sind MRS- und RFA-Kenntnisse notwendig.

1.3 Konzept des Handbuchs

Das Softwareprodukt LEASY wird in drei Handbüchern beschrieben:

- LEASY Programmschnittstelle und Konzepte
- LEASY Dienstprogramme
- LEASY Taschenbuch

Das Handbuch **LEASY Dienstprogramme** beschreibt die Dienstprogramme von LEASY. Es richtet sich primär an Organisatoren und Administratoren von LEASY-Anwendungen.

Das **LEASY Taschenbuch** richtet sich an Anwendungsentwickler und Administratoren von LEASY. Es soll die praktische Arbeit mit LEASY durch eine handliche Zusammenstellung der LEASY-Kommandos und -Operanden sowie diverser Tabellen erleichtern.

Das vorliegende Handbuch **LEASY Programmschnittstelle und Konzepte** gibt einen Überblick über das Softwareprodukt LEASY und eine ausführliche Anleitung zum Programmieren von LEASY-Anwendungen.

- Als **Entwickler** von LEASY-Anwendungsprogrammen bietet Ihnen das Handbuch somit alle Informationen, die Sie benötigen.

  Das Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121 beschreibt die LEASY-Programmschnittstelle unabhängig von der Programmiersprache. Dieses Kapitel ist Voraussetzung zum Verständnis der Kapitel „COBOL-Schnittstelle“ auf Seite 195 und „Assembler-Schnittstelle“ auf Seite 221.

- Als **Administrator** finden Sie im Kapitel „LEASY - ein Überblick“ auf Seite 13 eine Übersicht über die wichtigsten Merkmale und Funktionen des Produktes. Besonders hingewiesen sei auch auf das Kapitel „Technische Daten“ auf Seite 425.

- Als **Organisator** wird für Sie neben dem Kapitel „LEASY - ein Überblick“ auf Seite 13. das Kapitel „Sicherheitskonzept“ auf Seite 51 von besonderer Bedeutung sein. Informieren Sie sich auch über die Kapitel „Dateiverwaltung“ auf Seite 23 und „Betriebsarten“ auf Seite 85.

Für „Einsteiger“

Das Kapitel „LEASY - ein Überblick“ auf Seite 13 bringt eine geraffte Einführung mit der Beschreibung der wichtigsten Leistungsmerkmale. Speziell Anwender, die das Softwareprodukt LEASY zum erstenmal verwenden, sollten dieses Kapitel unbedingt lesen.

Das Kapitel „Anwendungsbeispiele“ auf Seite 329 enthält zusammenhängende Beispiele zur COBOL- und Assembler-Schnittstelle sowie zum Einsatz der LEASY-Dienstprogramme.

Die restlichen Kapitel können auch unabhängig voneinander gelesen werden. Sind zum Verständnis einzelner Teile andere Kapitel nötig, so wird auf die betreffenden Stellen im Handbuch verwiesen.

Für erfahrene LEASY-Anwender

Informieren Sie sich im Änderungsprotokoll auf der folgenden Seite über die wesentlichen Neuerungen in der Version 6.2A. Von dort aus wird direkt auf jene Abschnitte verwiesen, in denen die Neuerungen ausführlich dargestellt sind.

1.4 Änderungen gegenüber dem Vorgängerhandbuch

Die Änderungen des vorliegenden Handbuchs gegenüber dem Benutzerhandbuch zu LEASY V6.1A sind auf folgende wesentliche Neuerungen der LEASY-Version 6.2A zurückzuführen:

- Kontrolliertes Freigeben von AIM-Dateigenerationen

  AIM-Dateigenerationen werden standardmäßig gegen Löschen geschützt. Bevor sie gelöscht werden können, müssen sie freigegeben werden. Diese Freigabe wird entweder automatisch von LEASY nach dem automatischen Nachziehen von Schattendateien durchgeführt, oder sie muss vom LEASY-Administrator mit der Funktion *AIMA* des Dienstprogramms LEASY-MASTER vorgenommen werden, wenn die Dateien nicht mehr benötigt werden, siehe Seite 60.

- Originaldateien im laufenden Betrieb durch Schattendateien ersetzen

  Die Funktion *REPO* des Dienstprogramms LEASY-MASTER ermöglicht es, Dateien durch ihre Schattendateien zu ersetzen, ohne den LEASY-Betrieb zu beenden, siehe Seite 81.

- Online-Sicherung

  Mit der Funktion *ROMS* im Dienstprogramm LEASY-MASTER können Dateien mit einer Schreibsperre versehen werden. Dies ermöglicht eine anschließende Sicherung dieser Dateien im laufenden Betrieb, siehe Seite 83.

- Zusätzliche Informationen in der Diagnosedatei, siehe Seite 421.

1.5 Verwendete Darstellungsmittel

Im Fließtext werden die Namen von Kommandos und Operanden sowie Dateinamen, Pfadnamen und Bildschirmelemente in *kursiver Schrift* wiedergegeben. Wichtige Begriffe und Gegensatzpaare sind im Text durch **fette Schrift** hervorgehoben.

Meldungstexte und Beispiele für Systemausgaben stehen in `dicktengleicher Schrift`.

Texte, die Sie eingeben müssen, stehen in **`fetter dicktengleicher Schrift`**.

Dieses Symbol steht vor besonders wichtigen Warnungen, die im Interesse der System- und Betriebssicherheit unbedingt beachtet werden sollten, da es ansonsten zu Sicherheitslücken, zu Datenverlust oder zur Blockierung von Rechnern oder Leitungen kommen kann.

Dieses Symbol steht vor wichtigen Hinweisen, die im Interesse der System- und Betriebssicherheit beachtet werden sollten.

Literaturhinweise werden im Text in Kurztiteln angegeben. Der vollständige Titel jeder Druckschrift, auf die verwiesen wird, ist im Literaturverzeichnis angeführt.

Für die formale Darstellung der Anweisungen und ihrer Operanden werden folgende Metazeichen verwendet:

Formale Darstellung	**Erläuterung**	**Beispiele**
GROSSBUCH-STABEN und Sonderzeichen	Großbuchstaben und Sonderzeichen bezeichnen Konstanten, die der Benutzer in dieser Form eingeben muss.	*CAT dateikatalog Einzugeben ist: *CAT TESTCAT
kleinbuchstaben	Kleinbuchstaben bezeichnen Variablen, die der Benutzer durch aktuelle Werte ersetzen muss.	
{ }	Geschweifte Klammern schließen Alternativen ein, von denen der Benutzer eine auswählen muss.	{ datei datei.zusatz datei. } Einzugeben ist: DATEI1 oder DATEI1.Z1 oder DATEI1.

Tabelle 1: Metazeichen (Teil 1 von 2)

Formale Darstellung	Erläuterung	Beispiele
[ ]	Eckige Klammern schließen Wahlangaben ein.	keyname[,iub] Einzugeben ist: KEY1 oder KEY1,X' 00'
...	Punkte bedeuten eine Wiederholung; die davor stehende Einheit kann mehrmals hintereinander wiederholt werden.	(pos,len),... Einzugeben ist: (12,4) oder (12,4),(14,10),(25,2)
_	Die Unterstreichung hebt den Standardwert hervor. Das ist der Wert, den das Dienstprogramm einsetzt, wenn der Benutzer keine Angabe macht.	INF= {Y / N} Einzugeben ist: INF=Y oder INF=N oder nichts (entspricht INF=N)

Tabelle 1: Metazeichen (Teil 2 von 2)

2 LEASY - ein Überblick

Im folgenden Kapitel werden die grundsätzliche Arbeitsweise sowie die wesentlichen Leistungsmerkmale von LEASY beschrieben.

2.1 Vorbereitung und Ablauf einer LEASY-Session

Zur Vorbereitung einer LEASY-Session muss als erster Schritt mit dem Dienstprogramm **LEASY-CATALOG** ein LEASY-Katalog eingerichtet werden.

Danach können mit LEASY-CATALOG

- neue Dateien im LEASY-Katalog eingetragen und physikalisch auf der Platte bzw. auf Band erzeugt werden.
- bereits bestehende DVS-Dateien in den LEASY-Katalog aufgenommen werden.
- Dateien aus dem LEASY-Katalog gelöscht werden.
- die Art der Sicherung für einzelne Dateien festgelegt werden. Prinzipiell wird zwischen BIM-(Before-Image) und AIM-Sicherung (After-Image-Sicherung) unterschieden.

Nach dieser Vorbereitung für die LEASY-Session wird mit dem Dienstprogramm **LEASY-MAINTASK** die LEASY-Maintask gestartet. Die Maintask richtet den Common Memory ein. Der Common Memory (CMMAIN) ist der gemeinsame Speicherplatz aller Tasks, die an einen bestimmten LEASY-Katalog angeschlossen sind. Die LEASY-Maintask bearbeitet während einer LEASY-Session die Anforderungen der Anwenderprogramme. Unter 'LEASY-Session' ist dabei der Zeitraum zwischen dem Einrichten und dem Auflösen des Common Memory zu verstehen.

Mit dem Dienstprogramm LEASY-MAINTASK werden außerdem folgende Einstellungen vorgenommen:

- Die Dateisicherung wird global festgelegt.
- Die Randparameter der Sicherung (z.B. speichern der AIM-Datei auf Privatplatte) werden eingestellt.
- Die Parameter für den Common Memory (z.B. Größe) werden festgelegt.

Wenn der Common Memory eingerichtet ist, kann der Anwender sein Programm starten. Das LEASY-Laufzeitsystem ist im **Anwenderprogramm** eingebunden. Der Anwender arbeitet über eine vordefinierte Schnittstelle (COBOL- bzw. Assembler-Schnittstelle) mit LEASY. Über die Schnittstelle kann der Anwender Operationen für die Dateibearbeitung und für die Steuerung der **Transaktionen** (vgl. Seite 15) auslösen.

Mit dem Dienstprogramm **LEASY-MASTER** wird der Ablauf der LEASY-Session überwacht und gesteuert.

Der Überwachung dienen die Anzeigefunktionen. Sie liefern Daten über

– den Zustand der Dateien
– den Zustand der Transaktionen
– den Zustand des Common Memory
– die Anzahl und Art der durchgeführten LEASY-Operationen.

Die LEASY-Session und die Transaktionen können durch folgende Steuerfunktionen beeinflusst werden:

– Anhalten, Fortsetzen und Beenden einer Session
– Anhalten, Fortsetzen und Rücksetzen einer Transaktion
– Beeinflussen des Ablaufs der AIM-Sicherung
– Sperren und Entsperren von Dateien.

Die Dateibearbeitung bzw. die Transaktionen werden in den Anwenderprogrammen mit den entsprechenden Operationen beendet.

Die LEASY-Session wird mit dem Dienstprogramm LEASY-MASTER beendet.

2.2 Leistungsmerkmale

In diesem Abschnitt sind die Leistungsmerkmale von LEASY zusammengefasst.

Transaktionsbetrieb

Der Begriff der Transaktion wird verwendet, um konsistente Einheiten zu definieren. Mehrere Aktionen bilden eine Transaktion und sollen entweder vollständig oder gar nicht durchgeführt werden.

Eine **LEASY-Transaktion** ist eine Folge von LEASY-Operationen zwischen der Operation *OPTR* (Beginn der Transaktion) und *CLTR* (Ende der Transaktion). Diese Operationen werden von einem Anwenderprogramm abgesetzt.

Eine Grundvoraussetzung zur Erhaltung der logischen und physikalischen Konsistenz bei gleichzeitig ablaufenden Transaktionen ist die Möglichkeit, unvollständige Transaktionen rückgängig machen zu können, ohne die Veränderungen der parallelen Transaktionen zu beeinflussen. Dieses Rücksetzen kann erfolgen:

- durch explizite Anforderung im aufrufenden Programm (siehe LEASY-Operation *CLTR* mit Zusatzfunktion *OPE1=R*).
- durch explizite Anforderung mit der LEASY-Operation *BACK*.
- durch explizite Anforderung mit der Funktion *RLBT* im Dienstprogramm LEASY-MASTER.
- durch Aktivierung einer STXIT-Routine, z.B. nach fehlerhafter Programmbeendigung.
- bei einem Warmstart des Systems durch das Dienstprogramm LEASY-MAINTASK.

Das Rücksetzen einer Transaktion wird stets mit transaktionsbezogenen Before-Image-Dateien durchgeführt. Es kann deshalb nur erfolgen, falls diese Dateien durch entsprechende Angaben in den Dienstprogrammen LEASY-CATALOG und LEASY-MAINTASK geführt werden.

Transaktionsorientiertes Sicherungsverfahren BIM

Tritt während der Ausführung einer Transaktion ein Fehler auf, d.h. die Transaktion kann nicht vollständig abgearbeitet werden, müssen alle Veränderungen in den betroffenen Dateien zurückgesetzt werden. Um dies zu ermöglichen, kann jeder Transaktion eine BIM-Datei (**B**efore-**Im**age-Datei) zugeordnet werden.

Die BIM-Datei enthält die Informationen über die durchgeführten Dateizugriffe seit dem letzten Konsistenzpunkt. Beim Rücksetzen einer Transaktion werden mit Hilfe der Informationen aus der BIM-Datei:

– eingefügte Sätze bzw. Blöcke wieder gelöscht.

– gelöschte Sätze bzw. Blöcke wieder eingefügt.

– Änderungen in Sätzen bzw. Blöcken wieder rückgängig gemacht.

Wurde die Transaktion vollständig durchgeführt bzw. zurückgesetzt, so wird die BIM-Datei als „leer“ definiert.

Automatischer Wiederanlauf bei Systemzusammenbruch

Wenn nach einem Systemabsturz ein Warmstart (automatischer Wiederanlauf) durchgeführt wird, werden offene Transaktionen zurückgesetzt. LEASY überprüft, ob BIM-Dateien existieren, die nicht logisch leer sind. Diese werden abgearbeitet und logisch gelöscht. Die Dateien bleiben erhalten. Anschließend werden für die neue Session BIM-Dateien initialisiert.

Dateiorientiertes Sicherungsverfahren AIM

Um im Falle einer Dateizerstörung den aktuellen Datenbestand rekonstruieren zu können, müssen die an der Originaldatei vorgenommenen Änderungen in eine Datei protokolliert werden. Dies geschieht durch das Führen von **A**fter-**Im**age-Dateien. Jedem LEASY-Katalog ist eine AIM-Dateigenerationsgruppe zugeordnet, in die alle an den Katalog angeschlossenen Programme die Daten der veränderten Sätze schreiben.

Zerstörte Dateien werden rekonstruiert, indem Sicherungsdateien eingespielt werden, in die der Inhalt der AIM-Datei mit dem Dienstprogramm LEASY-RECONST eingearbeitet wird.

Unterstützung von Schattendateien

Schattendateien sind Kopien der Originaldateien, die parallel zur Bearbeitung der Originale laufend aktualisiert werden. Das Mitführen von Schattendateien in LEASY ermöglicht den 24-Stunden-Betrieb und einen schnellen Wiederanlauf nach einer physikalischen Dateizerstörung.

Zur Aktualisierung der Schattendateien wird während der LEASY-Session von einer AIM-Dateigeneration auf die nächste umgeschaltet und mit der abgeschlossenen AIM-Datei die Schattendatei mit Hilfe des Dienstprogramms LEASY-RECONST aktualisiert. Wird der Inhalt der aktuellen AIM-Datei klein gehalten, so ist die Differenz zwischen der Schattendatei und dem Original gering. Bei einer Dateizerstörung ist dann eine schnelle Rekonstruktion möglich.

Das Umschalten der AIM-Datei wird über die Angabe einer Maximalgröße in LEASY-MAINTASK (**AIS*-Operand) oder durch Steuerung über das Dienstprogramm LEASY-MASTER vorgenommen.

Sekundärindizierung

Für jede ISAM-, DAM- oder PAM-Datei besteht die Möglichkeit, neben den **Primärschlüsseln** bis zu 255 **Sekundärschlüssel** mit Teilschlüsseln zu definieren und über diese zuzugreifen.

LEASY arbeitet dabei nach folgendem Verfahren: zu jeder ISAM-, DAM- oder PAM-Datei, für die ein oder mehrere Sekundärindizes vereinbart werden, wird eine weitere (ISAM-) Datei, die **Sekundärindexdatei** (SI-Datei) angelegt. Diese enthält Verweisinformationen vom Sekundärschlüssel-Wert auf den Primärschlüsselwert. Ein Sekundärindex kann aus mehreren Teilen (Teilschlüssel) zusammengesetzt sein und/oder mit einem Wiederholungsfaktor versehen werden (multipler Sekundärindex).

Die Gültigkeit von Sekundärindex-Definitionen kann vom Inhalt eines Satzartenfeldes abhängig gemacht werden. Das Satzartenfeld definiert der Anwender.

LEASY stellt zur Sekundärindex-Verwaltung zwei Dienstprogramme zur Verfügung:

LEASY-CATALOG	Damit können Name, Position und Länge der Sekundärindizes und der Satzartenfelder und die Zuordnung von Sekundärschlüsseln an bestimmten Satzarten definiert werden. Außerdem kann bestimmt werden, dass bei Änderungen in der Primärdatei automatisch die Verweise in der Sekundärindex-Datei verändert werden sollen.
LEASY-LOADSI	Damit wird das Neuerstellen, Hinzufügen und Löschen von Sekundärindex-Verweisen in Sekundärindex-Dateien ermöglicht.

In der LEASY-Programmschnittstelle kann der Name eines Sekundärschlüssels angegeben werden, über den direkt zugegriffen oder positioniert werden soll.

Interne Kopplung mit openUTM

Durch die Kopplung mit openUTM ist LEASY auch im Teilhaberbetrieb einsatzfähig. Die Datensicherheit in einer LEASY-openUTM-Anwendung ergibt sich durch das gemeinsame Transaktionskonzept von LEASY und openUTM.

LEASY in DCAM-Umgebung

Für spezielle Funktionen im Teilhaberbetrieb, die über das Funktionsspektrum von openUTM hinausgehen, stellt LEASY einen DCAM-Anschluss zur Verfügung.

DRIVE-LEASY

Die LEASY-Funktionen können auch über DRIVE, eine Programmiersprache der 4. Generation, genutzt werden. DRIVE-LEASY ist auch unter openUTM, nicht aber mit DCAM einsetzbar.

3 Sperrkonzept

LEASY bietet ein hierarchisch geordnetes, zweischichtiges Sperrkonzept zum Schutz von Dateien oder Datensätzen vor unberechtigten Zugriffen in parallelen Tasks bzw. Transaktionen.

3.1 Dateisperren

Der **OPEN-Modus** gibt die Art der Dateieröffnung an, wie sie vom DVS durchzuführen ist.

Der **USAGE-Modus** gibt die Art der Verwendbarkeit der Datei in einer Transaktion an. Die möglichen USAGE-Modi sind abhängig von der Angabe für den OPEN-Modus. Andererseits wird bei impliziter Dateieröffnung, d. h. keine Angabe von *OPFL* vor *OPTR*, der OPEN-Modus aus dem USAGE-Modus bestimmt. Beide Modi geben an, ob die Datei:

– neu erstellt oder eine bestehende Datei verarbeitet werden soll

– lesend, schreibend in einer oder mehreren Tasks bearbeitet werden darf.

Beide Modi sollten nur so einschränkend wie notwendig angegeben werden, da Exclusive-, Protected- oder Shared-Update-Zugriffsrechte parallele Tasks bzw. Transaktionen beeinflussen und behindern können. Dateien und Transaktionen sollten zum frühest möglichen Zeitpunkt wieder geschlossen werden.

3.2 Satzsperren

LEASY führt Satzsperren für Stamm- und Modelldateien der Zugriffsmethoden ISAM, DAM und PAM, wenn sie mit den USAGE-Modi *UPDT, PRUP, RETR, LDUP* oder *PLUP* eröffnet werden. Für LEASY-Fremddateien der Zugriffsmethoden ISAM, DAM und PAM lässt sich eine eingeschränkte Koordinierung über die ISAM- und PAM-Sperrmechanismen erreichen, die jedoch nicht automatisch transaktionsorientiert sind.

Hinsichtlich der sperrenden Operation unterscheidet man:

– **implizite Sperren**, die bei den Operationen *INSR* und *STOR* automatisch für den einzelnen Satz gesetzt werden.

– **explizite Sperren**, die bei den Operationen *LOCK, RHLD, RNHD* und *RPHD* für den angegebenen Schlüssel oder das angegebene Schlüsselintervall gesetzt werden. Explizit können auch so genannte Phantomsperren gesetzt werden, das sind Sperren von nicht oder noch nicht vorhandenen Sätzen.

Hinsichtlich des Sperrzwecks unterscheidet man zwischen READ-LOCK und WRITE-LOCK.

– **READ-LOCK**

  Wird für explizite Sperren in Dateien, die mit dem USAGE-Modus *RETR* eröffnet wurden und für explizite Sperren mit der Operationscodeergänzung *OPE1=S* angenommen.

  Der READ-LOCK kann von mehreren Transaktionen gesetzt werden und ermöglicht den Schutz vor Überschreiben der gesperrten Sätze in parallelen Transaktionen.

  Zum Schutz gegen Deadlocks besitzt LEASY ein Deadlock-Erkennungsverfahren. Bei einem Deadlock werden Transaktionen gegenseitig so blockiert, dass ihre Fortführung nicht mehr möglich ist.

  Beispiel

Transaktion A	sperrt Satz 1
Transaktion B	sperrt Satz 2
Transaktion A	will Satz 2 sperren, muss warten
Transaktion B	will Satz 1 sperren.

  Es ergibt sich ein Deadlock.

LEASY erkennt solche Deadlocksituationen und verhindert sie. Die zuletzt angeforderte Sperre von Satz 1 durch Transaktion B wird mit dem Returncode *RC-LC L007* abgewiesen, die Transaktion kann bei Bedarf zurückgesetzt werden.

– **WRITE-LOCK**

Wird für implizite und explizite Sperren ohne Angabe der Operationscode-Ergänzung *OPE1* angenommen.

Der WRITE-LOCK kann nur exklusiv in einer Transaktion gesetzt werden. Er ist Voraussetzung für die Änderung eines Datensatzes. Das folgende Diagramm zeigt die Kombinationsmöglichkeiten der Sperren in zwei Transaktionen (T1 und T2):

T1 / T2	READ-LOCK	WRITE-LOCK
READ-LOCK	JA	NEIN
WRITE-LOCK	NEIN	NEIN

Freigabe von Sperren

Ein READ-LOCK oder ein WRITE-LOCK, der sich auf einen noch nicht in der Transaktion eingefügten, geänderten oder gelöschten Satz bezieht, kann mit der LEASY-Operation *UNLK* wieder gelöscht werden.

Generell werden die Satzsperren bei Transaktionsende (Operation *CLTR*) freigegeben.

Zugriff ohne Sperrprotokoll

Die beiden Usage Modi *ULRT* (Unlocked Retrieval) und *ULUP* (Unlocked Update) erlauben es, parallel einen mehrfachen Lesezugriff und einen einfachen Schreibzugriff auszuführen, ohne dass ein Sperrprotokoll geführt werden muss.

Diese beiden Usage Modi sind bei SAM-Dateien nicht einsetzbar.

Sperrprotokoll

LEASY realisiert die Satzsperren durch Führung eines Sperrprotokolls im Common Memory. Jedes Element des Sperrprotokolls enthält u.a. den Sperrtyp und den Primärschlüssel des gesperrten Satzes.

Die Elemente des Sperrprotokolls werden über die Funktion *SHLE* (Show Lock Elements) des Dienstprogramms LEASY-MASTER angezeigt.

Die folgende Tabelle zeigt die Auswirkung verschiedener Aktionen auf das Sperrprotokoll; beachten Sie in diesem Zusammenhang die Hinweise auf der darauf folgenden Seite.

Aktion	Bedeutung	
LOCK, RHLD, RNHD, RPHD	Einzelne Sätze einer ISAM-, DAM- oder PAM-Datei können gesperrt werden (explizite Sperre).	
LOCK, RHLD	Einzelne Satzintervalle einer ISAM-, DAM- oder PAM-Datei können gesperrt werden.	
LOCK	Mit der Operation *LOCK* können auch Sätze/Satzintervalle gesperrt werden, die noch nicht vorhanden sind, sog. Phantome.	
INSR, STOR	Neu eingefügte Sätze werden automatisch gesperrt (implizite Sperre).	
INSR, STOR, REWR, DLET	Sperren veränderter oder hinzugefügter oder gelöschter Sätze bleiben automatisch bis zum Ende der Transaktion erhalten. Sie können nicht mit *UNLK* freigegeben werden. Liegt der Satz in einem Sperrintervall, so erzeugt LEASY ein zusätzliches Sperrelement für diesen Satz. Mit *UNLK* kann dann zwar das Intervall freigegeben werden, die zusätzlichen Satzsperren bleiben jedoch bis Transaktionsende erhalten.	
DLET, REWR	Sätze, die gelöscht oder verändert werden sollen, müssen vorher implizit oder explizit gesperrt worden sein.	
UNLK	Gesperrte, aber nicht veränderte Sätze/Satzintervalle werden freigegeben bzw. bei *OPE1='U'* auch verändert.	
CLTR	Bei Transaktionsende werden alle Sperren automatisch freigegeben (gilt nicht für Fremddateien).	
„unprotected read“	Gesperrte, auch veränderte Sätze können von anderen Transaktionen gelesen werden (*RDIR, RNXT, RPRI*). Dieser sog. „unprotected read“ wird erlaubt, um eine höhere Parallelität von Transaktionen zu ermöglichen.	
Initialisieren der LEASY-Maintask mit dem Operanden **TIME*	Die maximale Wartezeit auf das Freigeben von Sätzen, die von parallelen Transaktionen gesperrt wurden, kann mit diesem Operanden global festgelegt werden.	Das erfolglose Ablaufen dieser Wartezeit wird durch einen Returncode mitgeteilt. Der Sperrversuch wird in Abständen von 1 Sekunde wiederholt.
Feld OPE-WTIME im *RE*-Bereich	Für jede Operation kann eine individuelle Wartezeit auf das Freigeben gesperrter Sätze angegeben werden.	

Tabelle 2: Auswirkung verschiedener Aktionen auf das Sperrprotokoll

i Die Anzahl der Satzsperren sollte so gering wie möglich gehalten werden. Dies erreicht man durch:

- kurze Transaktionen
- möglichst spätes Setzen von Satzsperren innerhalb der Transaktion
- möglichst frühe Freigabe von Sperren mit *UNLK*
- schreibende Operationen innerhalb der Transaktion möglichst spät einsetzen.

4 Dateiverwaltung

In LEASY können Dateien neben den DVS-Eigenschaften noch weitere Dateieigenschaften zugeordnet werden. Zur Verwaltung dieser Dateiattribute und für den Zugriff von LEASY auf die Dateien werden sie in einen **LEASY-Katalog** eingetragen. Unter einer Benutzerkennung können ein oder mehrere, voneinander völlig unabhängige LEASY-Kataloge angelegt werden.

4.1 LEASY-Katalog einrichten

Der LEASY-Katalog wird mit dem Dienstprogramm **LEASY-CATALOG** eingerichtet. Er ist eine ISAM-Datei mit dem Namen

:catid:$userid.dateikatalog.LEASYCAT

Dies ist der DVS-Name des LEASY-Katalogs. Er wird auch als **physikalischer Dateiname** bezeichnet. Der Benutzer kann von diesem Namen nur den Teil *dateikatalog* wählen, der auch als **logischer Dateiname** des LEASY-Katalogs bezeichnet wird.

Der LEASY-Katalog wird unter der Benutzerkennung katalogisiert, unter der das Dienstprogramm LEASY-CATALOG gestartet wurde. Die vom LEASY-Katalog verwalteten Dateien werden unter derselben Kennung eingerichtet, wenn nicht explizit mit der Angabe eines DVS-Namens eine andere Benutzerkennung angegeben wird (siehe *NAM*-Operand der **FIL*-Anweisung im Dienstprogramm LEASY-CATALOG).

Der LEASY-Katalog kann auch über ein *CREATE-FILE*-Kommando vor dem Aufruf des Dienstprogramms LEASY-CATALOG erzeugt werden. Wurde der Katalog vor dem erstmaligen Aufruf durch LEASY-CATALOG bereits eröffnet und geschlossen, so wird die Operation

**CAT dateikatalog,TYP=N*

abgelehnt.

Der LEASY-Katalog ist durch ein internes DVS-Schreibkennwort geschützt und kann nur durch das Dienstprogramm LEASY-CATALOG geändert oder gelöscht werden.

4.2 Aufbau des LEASY-Katalogs

Der LEASY-Katalog ist eine ISAM-Datei mit den Eigenschaften

RECORD-FORMAT=V
KEY-POSITION=5
KEY-LENGTH=29
BUFFER-LENGTH=STD(SIZE=8)

In diese Datei werden beim Erstellen Einträge gemacht:

- Der 1. Satz der Datei enthält Verwaltungsinformationen wie Kennwort, letzte Sessionnummer oder Angaben zur Bildung der Namen von Schattendateien.
- Für jede BIM-Datei und für die AIM-Datei existiert ein Verwaltungssatz.
- Für jede Datei, die in den LEASY-Katalog aufgenommen wird, wird ein Satz angelegt, der den logischen und physikalischen Dateinamen, Informationen über den Dateiaufbau und die LEASY-Dateieigenschaften enthält.
- Sind für eine Datei Sekundärschlüssel definiert, so sind im Satz auch alle Sekundärindex-Definitionen abgelegt (bei Modelldateien nur beim Modell, nicht bei den einzelnen Exemplaren).
- Für Modelldateien folgen anschließend die Dateieinträge der Exemplare einer Modelldateigruppe. Der Aufbau dieser Sätze entspricht dem Dateieintrag des Modells; jedoch ist hier der Dateiname inklusive der Zusatzangabe eingetragen.

4.3 Dateitypen im Sinne von LEASY

LEASY unterscheidet vier logische Dateitypen, die jeweils nach den Zugriffsmethoden SAM, ISAM, PAM oder DAM organisiert sein können. Es handelt sich dabei um:

- Stammdateien
- Modelldateien
- Temporärdateien
- Fremddateien.

Diese Dateien unterscheiden sich vor allem bezüglich ihrer Lebensdauer. Sie haben neben den DVS-Eigenschaften zusätzlich spezielle LEASY-Dateieigenschaften.

4.3.1 Stammdateien

Stammdateien sind Dateien mit langfristigem Charakter, die von ihrem Verwendungszweck her nur einmal vorhanden sind. Mit ihnen können alle LEASY-Funktionen genutzt werden, z.B. Sekundärindizierung, Mehrfachzugriff usw.

Die Gesamtheit der Stammdateien eines LEASY-Katalogs entspricht in vielen Eigenschaften einer Datenbank mit voneinander unabhängigen Satztypen in verschiedenen Bereichen.

Stammdateien werden beim Eintragen in den LEASY-Katalog mit dem Dienstprogramm LEASY-CATALOG eingerichtet bzw. übernommen.

Wird mit dem Dienstprogramm LEASY-CATALOG ein expliziter DVS-Dateiname für die Stammdatei vergeben, so wird dieser Dateiname verwendet; ansonsten wird der DVS-Dateiname (physikalischer Dateiname) nach folgender Systematik gebildet:

:catid:$userid.dateikatalog.datei

Bedeutung

catid	Katalogkennung
userid	Benutzerkennung, unter der der LEASY-Katalog geführt wird
dateikatalog	Name des LEASY-Katalogs
datei	Name der Stammdatei (logischer Dateiname)

4.3.2 Modelldateien

Eine LEASY-Modelldateigruppe besteht aus einem Modell und einem oder mehreren Exemplaren. Mit dem Modell werden alle Dateieigenschaften festgelegt, die für die Exemplare gelten sollen. Das Modell enthält keine Daten.

Exemplare eines Modells sind Dateien mit den gleichen Eigenschaften und Namen wie das Modell. Die Exemplare erhalten unterschiedliche Zusatznamen.

Das LEASY-Modelldateisystem erlaubt es, eine LEASY-Anwendung wahlweise auf verschiedenen, aber gleich strukturierten Datenbeständen ablaufen zu lassen. Der Anwender steuert durch die Angabe des Zusatznamens bei der Operation *CATD*, welches Exemplar der Modelldateigruppe bearbeitet werden soll (z.B. Dateien in einem Lohnsystem für einzelne Firmen, Abrechnungsperioden etc.).

Erstellen einer Modelldateigruppe

Mit dem Dienstprogramm LEASY-CATALOG wird im LEASY-Katalog zuerst ein Modell mit dem Namen

:catid:$userid.dateikatalog.datei

angelegt und es werden die Dateieigenschaften definiert. Dann werden die einzelnen Exemplare mit gleichem logischen Dateinamen (*datei*) wie das Modell, aber verschiedenen Zusatznamen (*zusatz*) in den Katalog eingebracht:

:catid:$userid.dateikatalog.datei.zusatz

LEASY legt intern eine Kopie des Modells auf die neuen Exemplare an.

4.3.3 Temporärdateien

Bei den Temporärdateien wird, ähnlich wie bei den Modelldateien, ein Stellvertreter mit den Dateieigenschaften im LEASY-Katalog erstellt. Die Temporärdateien entstehen aber im Gegensatz zu den Stamm- und Modelldateien erst dynamisch bei der 1. Verwendung in der Task. Sie werden unter der Kennung des Benutzerprozesses stets auf gemeinschaftlichen Datenträgern erzeugt. Dieses Prinzip eignet sich z.B. für Übergabedateien zwischen Programmen.

Temporärdateien sind task-eigen und damit stets exklusiv, d.h. es ist kein Vielfachzugriff möglich. Sie werden für jede neue TSN (Task Sequence Number) neu katalogisiert und sollten vor Taskende wieder durch ein *DELETE-FILE*-Kommando gelöscht werden. Dies gilt aber nur bei ordnungsgemäßer Beendigung der Task. Im Fehlerfall sind die Dateien zwecks Restart aufzuheben.

Namensstruktur bei Temporärdateien

$userid.LEAT.tsnr.datei

Bedeutung

userid	taskeigene Benutzerkennung
tsnr	TSN der Task
datei	Name der Temporärdatei

Für die Speicherung der Dateieigenschaften im DVS-Katalog wird eine leere Datei mit folgendem Namen eingerichtet:

$userid.dateikatalog.datei

4.3.4 Fremddateien

Eine Datei wird als LEASY-Fremddatei definiert, wenn sie mit der **FIL*-Anweisung (Operand *LEA=F*) in den LEASY-Katalog eingetragen wird. Wird ohne Common Memory *CMMAIN* gearbeitet, so gelten **alle** Dateien als Fremddateien.

Bei Fremddateien ist zu beachten, dass für sie keine LEASY-Sekundärschlüssel definiert werden können.

Fremddateien sind z.B. Dateien fremder Benutzerkennungen.

Bei der Vereinbarung einer Fremddatei im LEASY-Katalog werden von den DVS-Eigenschaften die *ACCESS-METHOD* und *BLOCK-CONTROL-INFO* vermerkt. Alle anderen DVS-Operanden werden abgewiesen. Die beim Ablauf fehlenden DVS-Eigenschaften werden aus dem TFT-Eintrag eines *ADD-FILE-LINK*-Kommandos oder aus dem DVS-Katalog entnommen.

Zugriff auf Fremddateien

Der Zugriff erfolgt hier - im Gegensatz zu den anderen Dateitypen - über den Dateikettungsnamen:

ADD-FILE-LINK LINK-NAME=Dateikettungsname,...

Vor dem Zugriff auf eine bestehende Datei ist daher ein *ADD-FILE-LINK*-Kommando mit Angabe von Dateiname, Dateikettungsname und *ACCESS-METHOD* zu geben. Bei DAM-Dateien ist zusätzlich noch *RECORD-FORMAT=FIXED(RECORD-SIZE=...)* anzugeben.

Soll eine neue Datei eingerichtet werden, sind nach dem Anlegen des Katalogeintrags mit dem *CREATE-FILE*-Kommando im *ADD-FILE-LINK*-Kommando die *ACCESS-METHOD* und der passende *OPEN-MODE*-Operand (*OUTIN* oder *OUTPUT*) anzugeben. Alle weiteren Eigenschaften, die nicht den DVS-Standardwerten entsprechen, sind zusätzlich anzugeben.

Bei DAM-Dateien sind :

```
RECORD-FORMAT=FIXED(RECORD-SIZE=...), ACCESS-METHOD=UPAM,BUFFER-LENGTH=...
```

sowie alle anderen Eigenschaften der Dateien anzugeben. Der im OPEN-MODE-Operanden angegebene Wert hat Vorrang vor dem OPEN-Modus der *OPFL*-Anweisung.

Eröffnung von Fremddateien mit SHARED-UPDATE=NO

Für die zulässige Reihenfolge von LEASY-Operationen gelten dieselben Regeln wie bei Stammdateien.

Eröffnung von Fremddateien mit SHARED-UPDATE=YES

Eine Eröffnung mit *SHARED-UPDATE=YES* erfolgt im *ADD-FILE-LINK*-Kommando. Falls im *ADD-FILE-LINK*-Kommando der *SHARED-UPDATE*-Operand nicht angegeben wird, leitet LEASY aus dem OPEN-/USAGE-Modus ab, ob mit oder ohne *SHARED-UPDATE* eröffnet wird. Fremddateien, für die die LEASY-BIM-Sicherung gewünscht wird, dürfen nur zum Lesen mit *SHARED-UPDATE=YES* eröffnet werden.
Bei Eröffnung mit *SHARED-UPDATE=YES* gilt folgendes spezielle Verhalten:

- Die LOAD-USAGE-Modi (*LOAD, LDUP, EXLD, PLOD, ELOD, PLUP* und *ELUP*) werden nicht unterstützt, da es keine Synchronisation bezüglich der höchsten Schlüsselnummer (bei ISAM), der höchsten relativen Satznummer (bei DAM) bzw. der höchsten PAM-Blocknummer (bei PAM) gibt.
- Bei Änderung eines ISAM-Satzes muss der Satz direkt vor der Änderung mit Sperre gelesen werden, da die Sperre auf den ISAM-Lockmechanismus abgebildet wird, d.h. folgende Operationsreihenfolge ist erforderlich:

  ```
  RHLD/RNHD/RPHD
  REWR
  ```

  Die Operationen *STOR, INSR, REWR, DLET, SETL, RDIR, RHLD, RNXT, RNHD, RPRI, RPHD* und *UNLK* werden unterstützt, nicht aber die Operation *LOCK*.

  ISAM kann nur eine Sperre aktuell halten und gibt bei jedem nachfolgenden Makroaufruf die ISAM-Sperre frei. LEASY gibt bei Transaktionsende die ISAM-Sperre nicht automatisch frei.
- Beim Zugriff auf PAM-Dateien mit *SHARED-UPDATE* verwendet LEASY statt des LEASY-Sperrprotokolls im Common Memory *CMMAIN* die Möglichkeiten der PAM-Block-Sperre durch UPAM.

  Bei den LEASY-Operationen *RHLD, RNHD* und *RPHD* werden die PAM-Blöcke mit dem PAM-Operationscode *LRDWT* (Lesen mit Sperre) gelesen. Bei den Operationen *REWR, DLET* sowie *INSR* und *STOR* werden die Blöcke mit dem PAM-Operationscode *WRTWU* (Schreiben und Aufheben der Sperre) geschrieben.

  LEASY prüft, ob vor dem Update eines Blocks die erforderliche Sperre gesetzt wurde. Der Anwender kann dadurch bei richtiger Programmierung (realisiert durch die LEASY-Operationenfolge *RHLD/RNHD/RPHD* und nachfolgendem *REWR/DLET*) einen geschützten Update realisieren.

  LEASY unterstützt die Operationen *LOCK* und *UNLK* auf einen bestimmten Block. Diese Operationen werden auf die PAM-Operationscodes *LOCK* und *UNLOCK* abgebildet.

DVS-Einschränkungen bezüglich der Sperrung von PAM-Blöcken

– Für jeden Prozess können maximal 255 PAM-Blöcke gleichzeitig gesperrt sein.

– Weder LEASY noch DVS-UPAM haben einen Überblick über die aktuelle Sperrsituation. Deshalb kann LEASY auch bei Transaktionsende keine evtl. übriggebliebenen PAM-Sperren aufheben. Durch einen physikalischen Close werden alle PAM-Sperren der abzuschließenden Datei freigegeben.

– Wird von einem Prozess ein PAM-Block gesperrt, so kann bei einer weiteren Sperranforderung ein Deadlock entstehen, der weder von LEASY noch von UPAM erkannt wird. Nach Ablauf der Wartezeit liefert UPAM den DVS-Fehlercode *09B0*. Dieser wird von LEASY auf den Fehlerschlüssel *99ALL007* abgebildet. In diesem Fall muss der Anwender alle PAM-Sperren auf diese Datei freigeben. Nach Ablauf der Wartezeit auf den ersten zu sperrenden PAM-Block liefert UPAM den DVS-Code *09B1*, den LEASY auf den Fehlerschlüssel *99ALL006* abbildet.

– LEASY gibt die im Feld *OPE-WTIME* des RE-Bereichs hinterlegte Wartezeit an UPAM weiter.

4.4 Datei-Zugriffsmethoden

LEASY unterstützt folgende vier Zugriffsmethoden für Dateien:

- die sequenzielle Zugriffsmethode SAM
- die indexsequenzielle Zugriffsmethode ISAM
- die blockorientierte Zugriffsmethode PAM
- die direkte Zugriffsmethode DAM.

Für jede Zugriffsmethode gelten einige Erweiterungen bezogen auf die Dateizugriffe höherer Programmiersprachen und einige Einschränkungen bezogen auf die DVS-Makro-Schnittstelle des BS2000.

4.4.1 Sequenzielle Zugriffsmethode SAM

Die von LEASY unterstützte Zugriffsmethode für sequenziell organisierte Dateien entspricht im Wesentlichen der Zugriffsmethode SAM des DVS-BS2000 (siehe Handbuch „Einführung in das DVS“).

Erweiterung

- Mit der LEASY-Operation *SETL* ist das Positionieren auf einen bestimmten Satz der Datei möglich.
- Mit *RDIR* kann ein Satz direkt gelesen werden. Beim Aufruf ist die SAM-Wiedergewinnungsadresse anzugeben.
- Bestimmte USAGE-Modi (siehe Seite 188 ff.) erlauben ein Rückwärtslesen sequenzieller Dateien, d.h. die Datei wird sequenziell in Richtung Dateianfang gelesen.
- Die SAM-Wiedergewinnungsadresse wird bei Lese- und Schreiboperationen zurückgeliefert (siehe LEASY-Operationen *RNXT* und *RPRI*).
- 4- bzw. 8-Byte SAM-Wiedergewinnungsadressen:
  SAM-Wiedergewinnungsadressen in AIM- und BIM-Dateien werden ausschließlich im 8-Byte-Format hinterlegt.
  Das Rückrollen von Transaktionen und die Wiederherstellung von Positionen innerhalb von SAM-Dateien wird voll unterstützt.
  SAM-Wiedergewinnungsadressen können vom Anwender in 4-, wie auch in 8-Byte-Form angegeben werden. Die 4-Byte-Adressen können in der Form *'bbbbbbrr'* mit *bbbbbb* = Blocknummer und *rr* = Satznummer angegeben werden, bei den 8-Byte-Adressen sind für Block- und Satznummer jeweils 4 Byte vorzusehen.

Verwendet der Anwender 4-Byte-Adressen, sind maximal 255 Sätze pro Block adressierbar. Übersteigt die Anzahl der Sätze im Block die Größe 255 (z.B. durch Einfügeoperationen (*INSR*)), erfolgt LEASY-intern eine Umschaltung auf 8-Byte-Adressen. Dieses Vorgehen ermöglicht eine Satzanzahl >255 im Block. Der Anwender erhält in diesen Fällen immer eine 8-Byte-Adresse geliefert und als Information den Returncode 99ALL011.

Einschränkung

- Das Satzformat *U* (undefinierte Länge) wird nicht unterstützt.
- SAM-Dateien, die sich über mehrere Bänder erstrecken, werden nicht unterstützt.
- Bei Dateien mit Nichtstandard-Blöcken liefert das DVS die SAM-Wiedergewinnungsadresse nicht zurück. Deshalb können Funktionen von LEASY, die sich auf die Wiedergewinnungsadresse eines Satzes beziehen, für diese Dateien nicht durchgeführt werden. Dabei handelt es sich um die Operationen *OPTR* (*OPE1=W*) und *CLTR* mit Rollback und die Operationen *SETL* und *RDIR*.
- Dateigenerationen werden nicht unterstützt.

4.4.2 Indexsequenzielle Zugriffsmethode ISAM

Die von LEASY unterstützte Zugriffsmethode für indexsequenziell organisierte Dateien entspricht im Wesentlichen der Zugriffsmethode ISAM des BS2000-DVS.

Erweiterung

- Wird eine ISAM-Datei über LEASY bearbeitet, so ist Rückwärtslesen entgegen der Schlüsselreihenfolge (Primär- und Sekundärschlüssel) möglich.
- Für ISAM-Dateien können LEASY-Sekundärschlüssel definiert werden:
  - maximal 255 Sekundärschlüssel pro Datei
  - maximal 253 Sekundärschlüssel-Teile pro Sekundärschlüssel
  - maximal 512 Sekundärschlüssel-Teile pro Datei.

  Die Länge des längsten LEASY-Sekundärschlüssels + Länge des ISAM-Primärschlüssels jeder Datei muss kleiner oder gleich 254 sein.

Einschränkung

- Das Satzformat *U* (undefined) wird nicht unterstützt.
- Der ISAM-Schlüssel muss immer eindeutig sein. Duplikate der Primärschlüssel sind verboten!
- Die BS2000-ISAM-Sperrmechanismen stehen dem Benutzer außer bei LEASY-Fremddateien nicht zur Verfügung. LEASY bietet stattdessen das transaktionsorientierte Sperrprotokoll.
- Für LEASY gibt es keine Zugriffsoperation für logisch markierte ISAM-Dateien (logical flagged files); bei Schreiboperationen werden allerdings die Markierungen mit aufgebaut.
- Dateigenerationen werden nicht unterstützt.

ISAM-Pools

Für NK-ISAM-Dateien besteht die Möglichkeit, mit ISAM-Pools zu arbeiten. Es sind dies virtuelle Speicherbereiche, die als Zwischenspeicher für ISAM-Blöcke dienen und die Ein-/Ausgaberate reduzieren.

LEASY unterstützt folgende ISAM-Pools

– Globale Standard-ISAM-Pools

– Private ISAM-Pools, die von **LEASY** verwaltet werden

– Private ISAM-Pools, die vom **Benutzer** verwaltet werden.

Im Folgenden sind für diese ISAM-Pools die Maßnahmen zusammengestellt, die vom Benutzer vor dem Starten einer LEASY-Session und von LEASY während einer LEASY-Session getroffen werden müssen.

Globale Standard-ISAM-Pools

- Vor dem Starten einer LEASY-Session

  Dienstprogramm LEASY-CATALOG

  1. In der **FIL*-Anweisung entfallen die Operanden *PLK/SLK* oder sie sind mit dem Operandenwert *PLK=*STD/SLK=*STD* anzugeben.
  2. Die **POO*-Anweisung entfällt.

- Während einer LEASY-Session

  Laufzeitsystem

  Bei der Operation *OPFL* schließt sich LEASY an den globalen Standard-ISAM-Pool an. Dieser ist immer vorhanden.

Von LEASY verwaltete private ISAM-Pools

Will der Benutzer mit privaten ISAM-Pools arbeiten, die von LEASY verwaltet werden, gibt es folgende Möglichkeiten:

- LEASY legt den ISAM-Pool beim Starten der Maintask an (Operand *PCR=MAINTASK* in der **POO*-Anweisung)
- LEASY legt den ISAM-Pool beim ersten *OPEN* auf die Datei im Laufzeitsystem an (Operand *PCR=RUNTIME* in der **POO*-Anweisung).

Mit PCR=MAINTASK definierte ISAM-Pools

Bei ISAM-Pools, die von LEASY beim Starten der Maintask angelegt werden, ist Folgendes zu beachten:

- Vor dem Starten einer LEASY-Session

  Dienstprogramm LEASY-CATALOG

  1. In der **FIL*-Anweisung sind in den Operanden *PLK/SLK* die ISAM-Poolkettungsnamen anzugeben.
  2. Für jeden in der **FIL*-Anweisung definierten ISAM-Pool ist eine entsprechende **POO*-Anweisung mit dem Operanden *PCR=MAINTASK* anzugeben.

- Während einer LEASY-Session

  Maintask

  Die Maintask legt beim Starten einer LEASY-Session alle mit *PCR=MAINTASK* definierten ISAM-Pools an. LEASY legt mit *CREATION-MODE=NEW* an, d.h. die entsprechenden ISAM-Pools dürfen noch nicht vorhanden sein.

  Laufzeitsystem

  Bei der Operation *OPFL* schließt sich LEASY an den von der Maintask angelegten ISAM-Pool an.

  – Vorteil bei *PCR=MAINTASK*

    Die ISAM-Pools werden am Beginn einer LEASY-Session angelegt und bleiben bis zum Ende der LEASY-Session erhalten. Der ISAM-Pool bleibt nach der Operation *CLFL* weiter bestehen und muss für eine neue Operation *OPFL* nicht neu angelegt werden.

  – Nachteil bei *PCR=MAINTASK*

    LEASY legt beim Starten einer LEASY-Session *alle* mit *PCR=MAINTASK* definierten ISAM-Pools an, unabhängig davon, ob sie dann in der LEASY-Session wirklich benötigt werden.

Mit PCR=RUNTIME definierte ISAM-Pools

Bei ISAM-Pools, die von LEASY im Laufzeitsystem angelegt werden, ist Folgendes zu beachten:

- Vor dem Starten einer LEASY-Session

  Dienstprogramm LEASY-CATALOG

  1. In der **FIL*-Anweisung sind in den Operanden *PLK/SLK* die ISAM-Poolkettungsnamen anzugeben.
  2. Für jeden in der **FIL*-Anweisung definierten ISAM-Pool ist eine entsprechende **POO*-Anweisung mit dem Operanden *PCR=RUNTIME* anzugeben.

- Während einer LEASY-Session

  Laufzeitsystem

  Bei der Operation *OPFL* legt LEASY den ISAM-Pool an bzw. schließt sich an den ISAM-Pool an, wenn dieser bereits vorhanden ist. LEASY legt mit *CREATION-MODE=ANY* an, d.h. die entsprechenden ISAM-Pools dürfen, müssen aber nicht bereits vorhanden sein.

  – Vorteil bei *PCR=RUNTIME*

    LEASY legt nur diejenigen ISAM-Pools an, die wirklich benötigt werden.

  – Nachteil bei *PCR=RUNTIME*

    Nach jeder Operation *CLFL* wird der ISAM-Pool wieder freigegeben. Wenn in einer LEASY-Session viele Operationen *OPFL/CLFL* ausgeführt werden, kann es zu häufigem Anlegen und wieder Löschen der ISAM-Pools kommen (schlechtere Performance).

  Der Benutzer darf auch ISAM-Pools mit *CREATE-ISAM-POOL* bzw. *ADD-ISAM-POOL-LINK* selbst anlegen. Dadurch wird bei häufigen Operationen *OPFL/CLFL* ein Löschen des entsprechenden ISAM-Pools verhindert. Es ist darauf zu achten, dass der Poolname (*CREATE-ISAM-POOL*) und der Poolkettungsname (*ADD-ISAM-POOL-LINK*) mit dem Poolkettungsnamen (**FIL*-Anweisung in LEASY-CATALOG) übereinstimmen, d.h der Poolname muss identisch mit den Poolkettungsnamen sein.

  Der Benutzer ist bei jedem selbst angelegten ISAM-Pool dafür verantwortlich, dass der ISAM-Pool mit *REMOVE-ISAM-POOL-LINK* bzw. *DELETE-ISAM-POOL* wieder ordnungsgemäß gelöscht wird.

Vom Benutzer verwaltete private ISAM-Pools

Will der Benutzer mit privaten ISAM-Pools arbeiten, die von ihm selbst verwaltet werden, müssen folgende Maßnahmen getroffen werden:

- Vor dem Starten einer LEASY-Session

  Dienstprogramm LEASY-CATALOG

  – In der **FIL*-Anweisung sind in den Operanden *PLK/SLK* die Poolkettungsnamen anzugeben.

  – Die **POO*-Anweisung entfällt.

  Benutzer

  Vor der Eröffnung der Datei muss der Benutzer mit *CREATE-ISAM-POOL* bzw. *ADD-ISAM-POOL-LINK* den benötigten ISAM-Pool selbst anlegen.
  Es ist darauf zu achten, dass der Poolname (*CREATE-ISAM-POOL*) und der Poolkettungsname (*ADD-ISAM-POOL-LINK*) mit dem Poolkettungsnamen (**FIL*-Anweisung in LEASY-CATALOG) übereinstimmen, d.h der Poolname muss identisch mit den Poolkettungsnamen sein.
  Der Benutzer ist bei jedem selbst angelegten ISAM-Pool dafür verantwortlich, dass der ISAM-Pool mit *REMOVE-ISAM-POOL-LINK* bzw. *DELETE-ISAM-POOL* wieder ordnungsgemäß gelöscht wird.

- Während einer LEASY-Session

  Laufzeitsystem

  LEASY trägt den Poolkettungsnamen, der in der **FIL*-Anweisung für LEASY-CATALOG angegeben ist, in den FCB für den OPEN-Makroaufruf ein. LEASY setzt dabei voraus, dass zu diesem Zeitpunkt der dazugehörige ISAM-Pool bereits vom Benutzer angelegt wurde.

4.4.3 Blockorientierte Zugriffsmethode PAM

Die von LEASY unterstützte Zugriffsmethode für blockorientierte Dateien setzt auf die Zugriffsmethode UPAM des BS2000-DVS auf (siehe Handbuch „Einführung in das DVS“).

Allgemeine Eigenschaften

PAM-Dateien mit folgenden Eigenschaften sind möglich:

BLOCK-CONTROL-INFO=PAMKEY

BLOCK-CONTROL-INFO=WITHIN-DATA-BLOCK

BLOCK-CONTROL-INFO=NO

Für jede PAM-Datei, die über LEASY bearbeitet werden soll, ist die Länge des Benutzerdatenblocks festzulegen. Ist diese Länge größer als 2048 Bytes, so ist der entsprechende *BUFFER-LENGTH*-Operand des *ADD-FILE-LINK*-Kommandos erforderlich (Standard-Blockung: *STD(SIZE=n)*, $n \leq 16$; Standardwert: n=1).

Ansonsten wird die Länge des Benutzerdatenblocks mit dem *RECORD-SIZE*-Operanden des *ADD-FILE-LINK*-Kommandos festgelegt. Es gilt:

– *RECORD-SIZE* ≤ *BUFFER-LENGTH*

– Standardwert: *RECORD-SIZE=BUFFER-LENGTH*

Für NK-Dateien gilt:

– *RECORD-SIZE* ≤ *BUFFER-LENGTH - 12*

– Standardwert: *RECORD-SIZE=BUFFER-LENGTH - 12*

Die Datenübertragung vom und zum Datenträger wird immer in der Länge *RECORD-SIZE* durchgeführt - bei entsprechender Länge mit geketteter Ein-/Ausgabe.

Für Ein-/Ausgabeoperationen wird vom Anwenderprogramm die PAM-Blocknummer (Half Page Number) des ersten PAM-Blocks angegeben, der zum Benutzerdatenblock gehört. Durch diese Konstruktion sind implizit alle für Ein-/Ausgabeoperationen des LEASY-Benutzerprogramms gültigen PAM-Blocknummern festgelegt. Es gilt:

gültige PAM-Blocknummer = 1 + (n*(i-1))

mit:

n Blockungsfaktor aus *BLKSIZE=STD(SIZE=n)*

i 1, 2, 3, ... (fortlaufend aufsteigende natürliche Zahlen)

LEASY führt kein eigenes Puffermanagement, sondern überträgt direkt vom bzw. zum Puffer des aufrufenden Programms.

Erweiterung

- Bei PAM-Dateien mit *BLOCK-CONTROL-INFO=PAMKEY* wird das logische Löschen von Datenblöcken durch Markierungen im PAM-Schlüssel aller PAM-Blöcke, die zum Benutzerdatenblock gehören, realisiert. Dabei wird der Benutzerteil des PAM-Schlüssels auf *X'FF'* gesetzt.
- Bei PAM-Dateien mit *BLOCK-CONTROL-INFO=WITHIN-DATA-BLOCK* wird das logische Löschen von Datenblöcken durch Setzen des Löschkennzeichens *X'FF'* am Blockanfang hinter dem Control-Field (*CF*) realisiert.
- Bei PAM-Dateien mit *BLOCK-CONTROL-INFO=NO* wird das logische Löschen von Datenblöcken durch Setzen des Löschkennzeichens *X'FF'* im ersten Datenbyte des PAM-Blocks gekennzeichnet.
- Für PAM-Dateien können Sekundärschlüssel definiert werden:
  - maximal 255 Sekundärschlüssel pro Datei
  - maximal 253 Sekundärschlüssel-Teile pro Sekundärschlüssel
  - maximal 512 Sekundärschlüssel-Teile pro Datei.
- Für PAM-Dateien ist über LEASY sequenzielles Rückwärtslesen in absteigender Reihenfolge der Blocknummern möglich.
- Es werden jeweils soviele Dateiblöcke gelesen oder geschrieben, als für den angegebenen Wert von *RECSIZE* notwendig sind.

Einschränkung

- Es können nur **ganze** Dateiblöcke gelesen oder geschrieben werden.
- Alle Ein-/Ausgabeoperationen werden synchron durchgeführt (*RDWT*-, *WRTWT*-Makroaufrufe), asynchrone Ein-/Ausgabe wird nicht unterstützt (auch nicht über Eventing-Mechanismen).
- Die Verkettung mehrerer Ein-/Ausgabeoperationen zu einem Aufruf ist nicht möglich.
- Die BS2000-UPAM-Sperrmechanismen stehen dem Benutzer - außer bei LEASY-Fremddateien - nicht zur Verfügung. LEASY bietet statt dessen das transaktionsorientierte Sperrprotokoll.
- Der Benutzer kann keine PAM-Schlüssel bearbeiten.
- Wahlfreier schreibender Zugriff auf PAM-Banddateien ist verboten, wird aber derzeit nicht überprüft.
- Dateigenerationen werden nicht unterstützt.

4.4.4 Direkte Zugriffsmethode DAM

Die von LEASY unterstützte Zugriffsmethode DAM verarbeitet geblockte Sätze fester Länge. DAM ist aus der **relativen Dateiorganisation** von COBOL abgeleitet (siehe COBOL-Beschreibung) und nach der KLDS-Norm ausgelegt. LEASY bildet diese Zugriffsmethode intern auf DVS-UPAM-Dateien ab. DAM-Dateien sind mit *BLOCK-CONTROL-INFO=PAMKEY, BLOCK-CONTROL-INFO=WITHIN-DATA-BLOCK* sowie *BLOCK-CONTROL-INFO=NO* möglich.

Die Zugriffsmethode DAM erlaubt:

– Direktes und sequenzielles Vorwärts- und Rückwärtslesen von Datensätzen

– Positionieren

– Einfügen, Ändern und Löschen von Datensätzen.

Allgemeine Eigenschaften

- Jeder Datensatz einer DAM-Datei wird durch eine relative Satznummer (Primärschlüssel der DAM-Zugriffsmethode) eindeutig identifiziert. Dieser Primärschlüssel ist 4 Byte lang und nicht Teil des Datensatzes. Sein Wert ist ganzzahlig (größer oder gleich Null). Er muss im zulässigen Bereich liegen.
- Die Zugriffsmethode DAM ist mit *FCBTYPE=DAM* zu definieren (Dienstprogramm LEASY-CATALOG).
- LEASY führt Blockung und Entblockung der Datensätze durch. Block- und Satzlänge sind bei der Definition der Datei festzulegen. Dabei gilt:
  RECORD-SIZE ≤ *BUFFER-LENGTH*

  Für NK-DATA-Dateien gilt:
  RECORD-SIZE ≤ *BUFFER-LENGTH* - 12
- Jeder durch Schreibzugriffe angefangene Datenblock wird vorformatiert (Auffüllen durch „gelöschte“ Sätze).
- Ein Datensatz gehört in folgenden Fällen nicht zur Datei:
  – wenn er in einem Datenblock liegt, der noch keinen Datensatz enthält oder
  – wenn er im ersten Byte das Löschkennzeichen *X'FF'* enthält.

Erweiterung

Für DAM-Dateien können Sekundärschlüssel definiert werden:

– maximal 255 Sekundärschlüssel pro Datei
– maximal 253 Sekundärschlüssel-Teile pro Sekundärschlüssel
– maximal 512 Sekundärschlüssel-Teile pro Datei.

Einschränkung

– DAM darf nur für Plattendateien vereinbart werden.
– Dateigenerationen werden nicht unterstützt.

4.5 Sekundärindizierung

Die Sekundärindizierung ermöglicht den direkten Zugriff auf Datensätze zusätzlich zum Primärschlüssel über einen oder mehrere Sekundärschlüssel. Die Schlüsselfelder sind Teile des Datensatzes.

Auf Stamm- und Modelldateien der Zugriffsmethoden ISAM (K und NK), DAM und PAM kann über LEASY-Sekundärschlüssel, auf Stamm-, Modell- und Fremddateien der Zugriffsmethode NK-ISAM kann über ISAM-Sekundärschlüssel zugegriffen werden.

LEASY-Sekundärschlüssel und ISAM-Sekundärschlüssel können über die Operationen *RDIR, RHLD, RNXT, RNHD, RPRI*, *RPHD* und *SETL* angesprochen werden.

In einer LEASY-Anwendung kann auf eine Datei sowohl über LEASY- als auch über ISAM-Sekundärschlüssel zugegriffen werden.

Für jede ISAM-, DAM- oder PAM-Datei können in LEASY bis zu 255 Sekundärschlüssel definiert werden. Die Sekundärschlüssel können auch als Teilschlüssel angegeben werden, wobei sich die Teilschlüssel überlappen dürfen.

Die Sekundärschlüssel-Definitionen können mit der Anweisung **FIL* im Dienstprogramm LEASY-CATALOG angelegt, neue hinzugefügt oder bestehende gelöscht werden.

Beim Definieren der Sekundärschlüssel wird auch festgelegt, ob gleiche Sekundärschlüssel-Werte zulässig sind, d.h. ob zu einem Sekundärschlüssel-Wert mehrere Primärschlüssel existieren dürfen.

Ein Sekundärindex kann auch mit einem Wiederholungsfaktor versehen werden (multipler Sekundärindex). Ein multipler Sekundärindex wird wie folgt angegeben:

```
KEY=(rep,(pos1,len1,dist1),(pos2,len2,dist2))
```

Die folgende Abbildung veranschaulicht das Prinzip, wobei in diesem Beispiel der Sekundärindex aus zwei sich überlappenden Teilschlüsseln besteht.

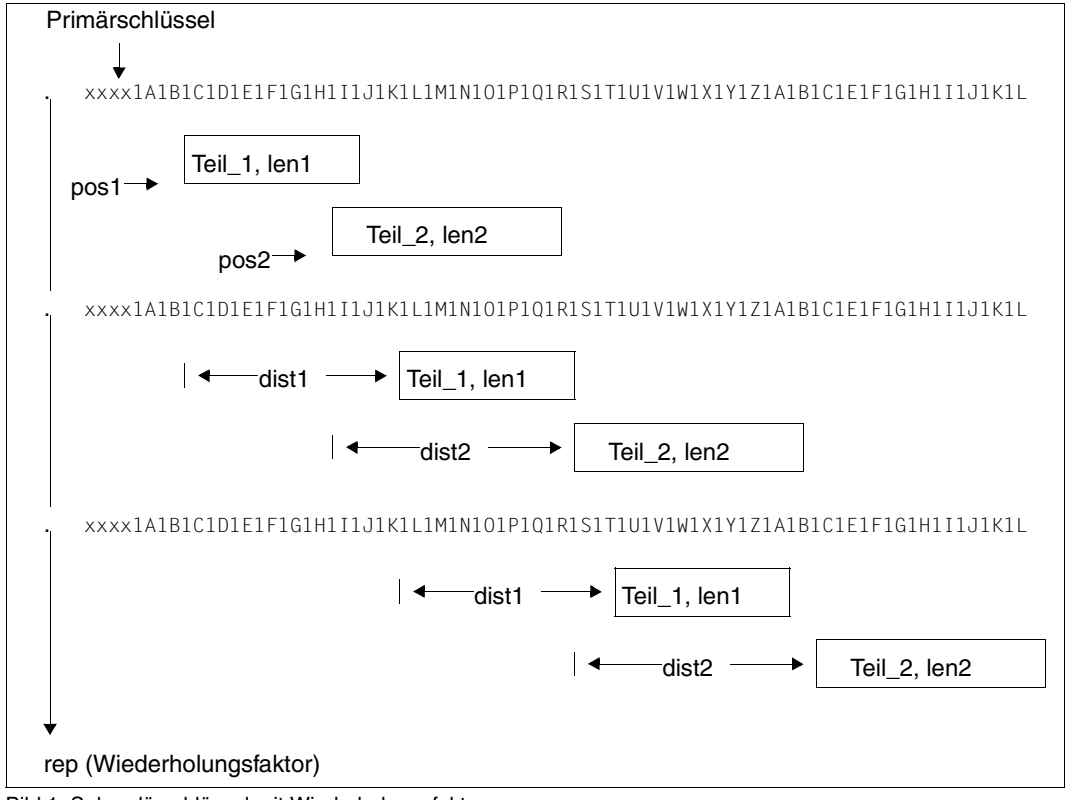


Bild 1: Sekundärschlüssel mit Wiederholungsfaktor

Realisierung der Sekundärindex-Verwaltung

Zu jeder ISAM-, DAM- oder PAM-Datei, für die Sekundärindizes vereinbart werden (Primärdatei), wird eine weitere Datei angelegt, die sog. Sekundärindex-Datei (SI-Datei). Diese ISAM-Datei enthält nur Verweisinformationen vom Sekundärschlüssel-Wert auf den Primärschlüssel-Wert.

Die Sekundärindex-Verweise können mit dem Dienstprogramm LEASY-LOADSI in der SI-Datei neu erstellt, hinzugefügt und gelöscht werden.

Bei Veränderungen der Sekundärschlüssel-Werte in der Primärdatei werden die Verweise in der SI-Datei vom LEASY-Laufzeitsystem aktualisiert.

Dateieigenschaften einer SI-Datei

```
FILENAME=:catid:$userid.dateikatalog.datei[.zusatz]-SI
bzw.
FILENAME=dvsname-SI

ACCESS-METHOD=ISAM
RECORD-FORMAT=V
KEY-POSITION=5
KEY-LENGTH=keylensi (siehe unten)
BUFFER-LENGTH=STD(SIZE=2)
```

Aufbau eines Satzes der SI-Datei

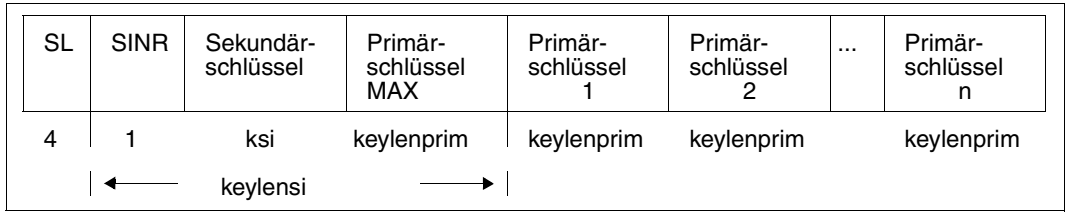


Bild 2: Aufbau eines Satzes der SI-Datei

Es bedeuten:

SL — Satzlängenfeld, Länge 4 Byte

keylensi — Länge des ISAM-Schlüssels der SI-Datei (keylensi=1+ksi+keylenprim).

SINR — interne Nummer des Sekundärschlüssels (1-255), Länge 1 Byte

Sekundärschlüssel / ksi
ksi ist die Länge des längsten definierten Sekundärschlüssels der zugehörigen Primärdatei. Da für alle definierten Sekundärindizes einer Datei nur eine SI-Datei geführt wird, wird das Feld Sekundärschlüssel mit der Länge des längsten vorhandenen Sekundärschlüssels angelegt. Kürzere Sekundärindex-Werte werden mit binären Nullen aufgefüllt.

Primärschlüssel MAX — Sind zu einem Sekundärschlüssel-Wert mehrere Primärschlüssel-Werte vorhanden, so werden die Primärschlüsselwerte hintereinander aufsteigend sortiert in den zugehörigen Satz der SI-Datei geschrieben. In einen Satz werden maximal soviele Primärschlüsselwerte geschrieben, bis die Satzlänge die Blocklänge (siehe Dateieigenschaften einer SI-Datei) erreicht.

Passen nicht alle Primärschlüssel-Werte in einen Satz, so werden mehrere Sätze zu einem Sekundärschlüssel-Wert angelegt, die durch die Art der ISAM-Schlüsselbildung innerhalb der SI-Datei wieder aufsteigend sortiert sind.

Der Primärschlüssel *MAX* ist größer oder gleich dem höchsten (d.h. gleich dem letzten) Primärschlüsselwert innerhalb des SI-Satzes. Der letzte oder einzige SI-Dateisatz zu einem Sekundärschlüssel-Wert enthält in diesem Feld den Wert *X'FF'*. Gibt es zu einem SI-Schlüsselwert keine Duplikate, so enthält *MAX* den Wert *X'00'*.

keylenprim — Die Primärschlüssel haben alle die Länge *keylenprim*; für PAM-Dateien gilt *keylenprim=3*; für DAM-Dateien ist *keylenprim=4*.

Wie aus der Struktur der SI-Datei ersichtlich ist, sollte der Primärschlüssel möglichst kurz gewählt werden. Ebenso ist zu überlegen, ob ein einziger, übermäßig langer Sekundärschlüssel die dadurch bewirkte Verlängerung des SI-Dateischlüssels rechtfertigt.

Größe einer SI-Datei

Eine vom Anwender über das *CREATE-FILE*-Kommando angelegte SI-Datei wird mit folgenden Werten für die Primär- und Sekundärgröße initialisiert:

```
MAX (Benutzerangabe, berechnete Größe)
```

Wenn der Benutzer keine Angabe macht, bzw. wenn der von LEASY berechnete Wert größer als die Benutzerangabe ist, wird der LEASY-Wert übernommen.

LEASY berechnet die Primär- bzw. Sekundärgröße nach folgender Formel:

```
[(#SPB) * (#BLK) * (#SKY) / (2048 / (2 * LPK + LSK + 5)) + 1]
```

Es bedeuten:

#SPB — Anzahl der Sätze der Primärdatei pro Block.

= Blocklänge/Satzlänge (bei fixem Satzformat)
= 5 (bei variablen ISAM-Dateien)
= 1 (bei PAM-Dateien)

#BLK — Primär-/Sekundärgröße der Primärdatei

#SKY — Anzahl der Sekundärschlüssel

LPK — Länge des Primärschlüssels

LSK — Länge des längsten Sekundärschlüssels

Der berechnete Wert wird auf eine ganze Zahl abgerundet. Dieses Ergebnis wird vom DVS auf das Vielfache einer Allokierungseinheit erhöht.
Bei der Sekundärgröße ist der berechnete Wert mit *1920* begrenzt.

ISAM-Sekundärschlüssel für NK-ISAM-Dateien

Ab BS2000 V10 unterstützt die DVS-Zugriffsmethode ISAM für NK-ISAM-Dateien (Nonkey-Isam) die Verwendung von Sekundärschlüsseln.

LEASY kann auf NK-ISAM-Datensätze sowohl über LEASY-Sekundärschlüssel, als auch über ISAM-Sekundärschlüssel zugreifen. Das Kriterium für die jeweilige Zugriffsart ist der Name des Sekundärschlüssels. Bei gleichen Namen des LEASY- bzw. ISAM-Sekundärschlüssels erfolgt der Zugriff über den LEASY-Sekundärschlüssel.

Die folgende Tabelle zeigt die wesentlichen Unterschiede zwischen LEASY- und ISAM-Sekundärschlüsseln:

Kriterium	**LEASY-SK**	**ISAM-SK**
Dateityp	Stammdateien	Stamm- und Fremddateien
Ablage der Indizes	eigene SI-Datei	Primärdatei
Zugriffsmethode	DAM, PAM, ISAM	NK-ISAM
Anzahl der SK	255	30
Länge der SK	PK + SK < 255	< 128
Multiple Keys	JA	NEIN
Unterdrückung	JA	NEIN
Reihenfolge bei gleichen SK	Primärkey	Zeit des Eintrages
Definition	Dienstprogramm LEASY-CATALOG	Kommando CREATE-ALTERNATE-INDEX, Makro CREAIX
Anzeige	Dienstprogramm LEASY-CATALOG	Dienstprogramm LEASY-CATALOG, Kommando SHOW-INDEX-ATTRIBUTES, Makro SHOWAIX
manueller Aufbau	Dienstprogramm LEASY-LOADSI	Kommando CREATE-ALTERNATE-INDEX, Makro CREAIX
automatischer Aufbau über LEASY-Runtimesystem	steuerbar über LEASY-CATALOG	für jeden definierten SK

Tabelle 3: Unterschiede zwischen LEASY- und ISAM-Sekundärschlüsseln

Ein einzelner Schlüsselwert kann an der ihm entsprechenden Stelle im *AR*-Bereich oder im *KB*-Bereich angegeben werden. Es wird der Satz gelesen, der den angegebenen Schlüsselwert enthält bzw. bei Duplikaten jener Satz, der als erster in die Datei geschrieben wurde.

Bei einem Intervall von Schlüsselwerten ist der Anfangswert im *KB*-Bereich anzugeben und der Endwert im *KE*-Bereich.

Im angegebenen Intervall wird der erste Satz gelesen. Wenn im angegebenen Intervall kein Satz gefunden wird, gibt LEASY der Returncode LC=L001 aus.

Bei Duplikaten ist es nicht möglich, zusätzlich zum Sekundärschlüssel einen Primärschlüssel zur weiteren Spezifikation anzugeben!

Einschränkungen der ISAM-SK gegenüber den LEASY-SK

1. Mehrere Dateipositionen können gleichzeitig nur über unterschiedliche Schlüsselnamen und Folgemerkmale unterschieden werden.
2. Es werden nur komplette Sätze in den AR-Bereich übertragen (der Operand *FA* wird nicht ausgewertet).
3. Bei Auftreten eines Fehlers ist der Inhalt des AR-Bereichs nicht definiert, das heißt, ein unveränderter Inhalt darf nicht vorausgesetzt werden.

Über den ISAM-Sekundärschlüssel kann mit den LEASY-Operationen *RDIR, RHLD, RNXT, RNHD, RPRI, RPHD* und *SETL* zugegriffen werden.

Um mit ISAM-Sekundärschlüssel in LEASY arbeiten zu können, sind folgende Hinweise zu beachten:

- Im Operanden *SI* ist der Name des ISAM-Sekundärschlüssels anzugeben.
- Ein einzelner ISAM-Sekundärschlüssel ist im *AR*-Bereich bzw. im *KB*-Bereich anzugeben.

  Sind mehrere Sätze mit dem gleichen Wert vorhanden, wird der Satz geliefert, der zuerst eingetragen wurde. Wird kein entsprechender Satz gefunden, erfolgt der Returncode *010LL001*.

  Ein Primärschlüssel kann nicht zusätzlich zum Sekundärschlüssel (zur weiteren Spezifikation von Duplikaten) angegeben werden.

- Bei der Angabe des Schlüsselintervalls wird der Anfangswert im Bereich *KB*, der Endwert im Bereich *KE* angegeben. Dabei gilt:

  KB ≤ KE Es wird der Satz mit dem kleinsten Sekundärschlüssel im angegebenen Intervall geliefert. Bei gleichen Sätzen wird der als Erster eingetragene Satz geliefert.

  KB > KE Es wird der Satz mit dem größten Sekundärschlüssel im angegebenen Intervall geliefert. Bei gleichen Sätzen wird der zuletzt eingetragene Satz geliefert.

  Wird kein entsprechender Satz gefunden, erfolgt der Returncode *010LL001*. Bei nachfolgenden sequenziellen Leseoperationen sind diese auf das angegebene Intervall begrenzt.

- Mit den Operationen *RNXT, RNHD, RPRI* und *RPHD* kann nach der Positionierung sequenzielltiell gelesen werden. Dabei gilt:

  RNXT, RNHD Es wird der Satz mit dem nächst größeren SK-Wert geliefert. Bei gleichen Sätzen wird der danach eingetragene Satz geliefert.

  RPRI, RPHD Es wird der Satz mit dem nächst kleineren SK-Wert geliefert. Bei gleichen Sätzen wird der zuvor eingetragene Satz geliefert.

  Bei Erreichen der Dateigrenze bzw. bei Überschreiten des angegebenen Intervalls wird der Returncode *010LL003* geliefert.

- Informationen über ISAM-Sekundärschlüssel können mit dem Dienstprogramm LEASY-CATALOG abgerufen werden.

4.6 Lage der Dateien

Die Ablage von Dateien auf unterschiedliche Datenträger ist in zweierlei Hinsicht von Bedeutung:

– Durch Einsparung von Positionierungen kann die Performance gesteigert werden.

– Das Konzept der Datensicherung erfordert örtliche Streuung der Dateien.

Private Datenträger können direkt gewählt werden, für öffentliche Datenträger ist eine Differenzierung nur über die Wahl des Pubsets möglich.

LEASY unterscheidet zwischen Anwenderdateien (Primär- und Sekundärdateien) und LEASY-Systemdateien (LEASY-Katalog, AIM-Datei, BIM-Dateien und LEASY-Statusdatei).

Bei der Streuung ist der Bezug der Dateien zueinander zu beachten:

– Die LEASY-Katalogdatei und die Anwenderdateien sollten auf dem gleichen Datenträger liegen.

– Die AIM-Datei, die BIM-Datei und die Anwenderdateien sollten auf unterschiedlichen Datenträgern liegen.

– Die BIM-Dateien sollten auf möglichst vielen Datenträgern verteilt werden (im Idealfall eine BIM-Datei je Datenträger);

– Bei großen K-ISAM-Dateien sollte von der Möglichkeit des Splittens von Index- und Datenteil Gebrauch gemacht werden (Operanden *VOL, DEV, DVOL*, und *DDEV* der **FIL*-Anweisung im Dienstprogramm LEASY-CATALOG). Die dazu gehörende Sekundärindex-Datei wird auf dem selben Datenträger angelegt, wie der Indexteil der Primärdatei.

  Für NK-ISAM-Dateien besteht diese Möglichkeit nicht, es kann jedoch durch Verwendung so genannter ISAM-Pools die Anzahl der Ein- und Ausgabeoperationen reduziert werden.

4.7 Reservierte Dateinamen

Im System LEASY gibt es systeminterne Dateien. Die Dateinamen für diese Dateien werden automatisch gebildet und lauten:

`dateikatalog.LEASYCAT`

`dateikatalog.LEASYAIM`

`dateikatalog.BIM#.`*nnn* (001 ≤ *nnn* ≤ 999)

`dateikatalog.LEATSTAT`

`dateikatalog.LEASAVE.TAPE`

`dateikatalog.LEASAVE.DISK`

`dateikatalog.LEADIAG`

Diese Dateinamen sind als reservierte Namen zu betrachten. Der Benutzer selbst darf sie mit Ausnahme der Datei *dateikatalog.LEADIAG* niemals ändern oder löschen.

Die Dateinamen der Sekundärindex-Dateien werden automatisch gebildet. Als Ausgangspunkt dazu dient der Name der jeweiligen Primärdatei. Zusätzlich wird zur Namensbildung die Zeichenkette „-SI“ verwendet, die an das Ende des Dateinamens gefügt wird.

Außerdem sind alle durch das Dienstprogramm LEASY-CATALOG gebildeten Dateinamen der Daten-Dateien und die während des Ablaufs einer Session evtl. gebildeten Temporärdateinamen als privilegiert zu betrachten. Die Kontrolle obliegt jedoch dem Benutzer.

5 Sicherheitskonzept

LEASY bietet dem Anwender zwei Sicherungskonzepte. Diese können sowohl session- als auch dateispezifisch über die Dienstprogramme LEASY-MAINTASK und LEASY-CATALOG gewählt werden:

– Die **Transaktionssicherung** ermöglicht das Rücksetzen von Transaktionen auf ihren Ausgangspunkt und damit die Erhaltung eines konsistenten Dateibestandes.

– Die **Datensicherung** ermöglicht das Wiederherstellen zerstörter Dateien aus Sicherungsbeständen. Einen Spezialfall der Datensicherung stellt das **Schattendateikonzept** dar, das den Zeitaufwand für Rekonstruktionen nach einem Fehlerfall für besonders zeitkritische Anwendungen möglichst gering hält.

Beide Konzepte sind im Folgenden beschrieben.

5.1 Transaktionssicherung

Die Transaktionssicherung ermöglicht das Rücksetzen von Transaktionen auf ihren Anfangszustand.

5.1.1 Transaktionsbegriff

Allgemein versteht man unter einer Transaktion eine Folge von Aktionen, die entweder vollständig oder gar nicht durchgeführt werden.

Eine LEASY-Transaktion ist eine Folge von LEASY-Operationen zwischen den LEASY-Operationen *OPTR* (Eröffnen der Transaktion) und *CLTR* (Schließen der Transaktion). Eine Grundvoraussetzung für die Erhaltung eines konsistenten Dateizustandes ist die Möglichkeit, unvollständige Transaktionen rückgängig machen zu können, ohne die Veränderungen in parallelen Transaktionen zu beeinflussen.

Dieses Rücksetzen kann erfolgen durch:

- explizite Aufforderung im Programm mit den Operationen *CLTR* (mit *OPE1=R*) oder *BACK*.
- explizite Aufforderung über die Funktion *RLBT* im Dienstprogramm LEASY-MASTER.
- Aktivierung einer STXIT-Routine (z.B. nach fehlerhafter Programmbeendigung).
- Warmstart des Systems durch LEASY-MAINTASK.

Die Transaktionssicherung ist über das BIM(Before-Image)-Sicherungsverfahren realisiert.

5.1.2 BIM-Sicherungsverfahren

Tritt während der Ausführung einer Transaktion ein Fehler auf, d. h. die Transaktion kann nicht vollständig durchgeführt werden, müssen alle Veränderungen in den betroffenen Dateien zurückgesetzt werden. Dabei werden:

– eingefügte Sätze bzw. Blöcke wieder gelöscht.

– gelöschte Sätze bzw. Blöcke wieder eingefügt.

– Änderungen in Sätzen bzw. Blöcken wieder rückgängig gemacht.

Das BIM-Sicherungsverfahren führt für jede der parallel ablaufenden Transaktionen eine BIM-Datei, in der zu jeder Operation die entsprechende Gegenoperation abgelegt wird.

Beispiel

Operation	**Inhalt der BIM-Datei**
Satz 8 in Datei A löschen	Satz 8 in Datei A einfügen Inhalt Satz 8
Satz 16 in Datei B ändern	Satz 16 in Datei B ändern alter Inhalt Satz 16
Satz 3 in Datei X einfügen	Satz 3 in Datei X löschen (Inhalt des Satzes wird zum Löschen nicht benötigt!)

Beim Rückrollen wird die BIM-Datei am Dateiende beginnend bis zum Dateianfang abgearbeitet.

Nach erfolgreichem Abarbeiten der BIM-Datei und bei ordnungsgemäßem Transaktionsende wird die BIM-Datei als „logisch leer“ definiert, d.h. der erste PAM-Block wird mit binären Nullen überschrieben.

5.1.3 BIM-Dateien

BIM-Dateien sind Dateien der Zugriffsmethode PAM, die mit dem Dienstprogramm LEASY-MAINTASK eingerichtet werden.

Folgende Anweisungen des Dienstprogramms LEASY-MAINTASK nehmen Bezug auf die anzulegende BIM-Datei:

*TRA	Anzahl der BIM-Dateien
*BCA	Pubset der BIM-Dateien
*BDE, *BVO	Gerätetyp und Name der Privatplatter für BIM-Dateien
*BIO	Performance-Attribut für BIM-Dateien

Namen der BIM-Dateien:

```
:catid:$userid.dateikatalog.BIM#.nnn
```

Bedeutung der Operanden

catid	*BCA-Angabe oder :catid: des LEASY-Katalogs
$userid	Benutzerkennung, unter der die LEASY-Maintask abläuft
dateikatalog	Name des LEASY-Katalogs, der der LEASY-Maintask zugewiesen wurde
BIM#	Kennzeichnung der Datei als BIM-Datei
nnn	interne Transaktionsnummer, beginnend bei 001

Die BIM-Dateien erhalten:

BUFFER-LENGTH	=STD(1)	auf NK2-Platten
	=STD(2)	auf NK4-Platten
BLOCK-CONTROL-INFO	=PAMKEY	auf KEY-Platten und *CLASS-2-OPTION BLKCTRL=PAMKEY*
	=NO	sonst

Die BIM-Dateien werden durch ein Lesekennwort gegen unerlaubte Zugriffe geschützt.

In Ausnahmefällen, wenn die von LEASY-MAINTASK vergebenen Dateiattribute nicht gewünscht werden, kann der Anwender die BIM-Dateien mit dem *CREATE-FILE*-Kommando auch selbst anlegen.

Beim Starten von LEASY-MAINTASK ohne BIM-Sicherung werden vorhandene BIM-Dateien gelöscht. Dies kann mit dem Operanden *KEEP-BIM-FILES* der **LOG*-Anweisung unterdrückt werden.

5.2 Datensicherung

Die Datensicherung ermöglicht das Wiederherstellen von Dateien, die auf Grund von Systemfehlern zerstört oder nicht mehr lesbar sind. Die Datensicherung ist über das AIM-(After-Image-) Sicherungsverfahren realisiert.

Das AIM-Sicherungsverfahren führt für jeden LEASY-Katalog eine AIM-Datei, in die alle Änderungen eingetragen werden, die während einer oder mehrerer LEASY-Sessions an einer Datei durchgeführt werden. Durch das Führen von Datum, Uhrzeit und Transaktionskennungen in der AIM-Datei ist diese auch transaktionsbezogen.

Zerstörte Originaldateien werden nach dem Einspielen von Sicherungskopien mit dem Dienstprogramm LEASY-RECONST rekonstruiert. LEASY-RECONST liest die AIM-Datei ein und bringt die LEASY-Dateien durch Aktualisierung auf den letzten Stand.

Der Anwender kann sich Generationen von Kopien der Original- und AIM-Dateien halten und die Rekonstruktion einer zerstörten Datei auch von einer älteren Kopie her durchführen.

5.2.1 AIM-Datei

Die AIM-Datei ist eine Datei der Zugriffsmethode PAM, die als Dateigenerationsgruppe angelegt werden muss. Der Benutzer hat folgende Möglichkeiten, AIM-Dateigenerationsgruppen einzurichten bzw. neue AIM-Generationen anzulegen:

- Dienstprogramm LEASY-MAINTASK

  Dazu stehen die Anweisungen **AGE, *ADE, *AVO, *ASP, *ACA, *AIO* und **AIS* zur Verfügung. Die **DES*-Anweisung legt zusätzlich fest, ob beim Löschen die AIM-Dateien mit binär Null überschrieben werden sollen. Die AIM-Dateien werden intern von LEASY-MAINTASK mit einem verschlüsselten Lesekennwort geschützt. Die Anweisungen **FAA* und **AGF* steuern die Freigabe von AIM-Dateigenerationen.

- *CREATE-FILE-GROUP*-Kommando im Format

  ```
  /CREATE-FILE-GROUP GROUP-NAME=:catid:$userid.dateikatalog.LEASYAIM
  ```

  kann vom Benutzer selbst eine Dateigenerationsgruppe erstmalig eingerichtet werden.

- *CREATE-FILE-GENERATION*-Kommandos im Format

  ```
  /CREATE-FILE-GENERATION
  GENERATION-NAME=dateikatalog.LEASYAIM(*n),SPACE=...
  ```

  kann vom Benutzer selbst die **erste** Generation angelegt werden. AIM-Dateien dürfen ebenso wie BIM-Dateien nur die beiden Formate *BLOCK-CONTROL-INFO=PAMKEY* sowie *BLOCK-CONTROL-INFO=NO* benutzen.

Das Löschen von AIM-Dateien ist mit dem Dienstprogramm LEASY-MASTER möglich.

Beispiel für das Anlegen einer neuen AIM-Dateigenerationsgruppe mit LEASY-MAINTASK

```
/START-LEASY-MAINTASK
CAT=LEACAT
LOG=Y
AGE=3
.
.
.
END
```

Mit diesen Anweisungen wird die Dateigenerationsgruppe *LEACAT.LEASYAIM* angelegt, von der maximal 3 Generationen gleichzeitig im Katalog enthalten sind. Sie kann unter beliebigen Benutzerkennungen bearbeitet werden.

Immer wenn eine neue (leere) AIM-Dateigeneration begonnen werden soll, muss im Dienstprogramm LEASY-MAINTASK die **ASP*-Anweisung angegeben werden. Dabei sind auch die entsprechenden Anweisungen **ADE* und **AVO* anzugeben, falls man die AIM-Datei auf einem privaten Datenträger anlegen will. Eine Dateigeneration darf jedoch nur vollständig auf gemeinschaftlichen oder vollständig auf privaten Datenträgern liegen. Soll in der nächsten (zuletzt beschriebenen) Generation weitergeschrieben werden, so ist keine **ASP*-Anweisung anzugeben.

Beispiel für das Anlegen einer neuen AIM-Dateigeneration mit LEASY-MAINTASK

```
*ASP=(160,100)
```

Mit dieser Anweisung wird die nächsthöhere AIM-Dateigeneration angelegt. Der *ASP*-Operand definiert die primäre (160 PAM-Seiten) und die sekundäre (100 PAM-Seiten) Speicherplatzzuweisung, d.h. dass für die AIM-Dateigeneration beim Erstellen 160 PAM-Seiten reserviert werden und - sobald mehr Platz benötigt wird - jeweils 100 PAM-Seiten zusätzlich reserviert werden.

Die Sekundärzuweisung muss mindestens 16 PAM-Seiten betragen, da mit PAM chained-I/O bis zu 32 K auf einmal geschrieben werden können. Dieser Wert wird von der LEASY-Maintask automatisch eingestellt, falls ein kleinerer Wert angegeben wurde oder die Angabe fehlt.

Bei Erreichen einer Maximalgröße oder durch Steuerung über das Dienstprogramm LEASY-MASTER kann auch während einer LEASY-Session von einer AIM-Dateigeneration auf die nächste umgeschaltet werden. Dabei werden ebenfalls die Angaben der **ASP-*, **ADE-* und der **AVO*-Anweisungen im LEASY-MAINTASK verwendet.

AIM-Dateien auf Privatplatte

Zum Einrichten einer AIM-Dateigenerationsgruppe auf Privatplatte sind im Dienstprogramm LEASY-MAINTASK mit der **ADE*-Anweisung der Gerätetyp und mit der **AVO*-Anweisung die Archivnummer der privaten Platte einzugeben. Alle AIM-Generationen müssen nun ebenfalls auf Privatplatte (nicht notwendigerweise auf derselben) liegen.

LEASY-MAINTASK richtet die AIM-Dateigenerationsgruppe mit *OVERFLOW-OPTION=REUSE-VOLUME* ein, um das Weiterschalten der AIM-Datei durch ein LEASY-Dienstprogramm zu ermöglichen. Dies ist nötig, da die von den LEASY-Dienstprogrammen gesetzten Makros keine Geräteangaben enthalten. Dabei gilt:

- Nur für die ersten Generationen bis zum Erreichen der mittels der **AGE*-Anweisung festgelegten maximalen Anzahl von Generationen müssen Geräteangaben explizit gesetzt werden.
- Beim Anlegen einer Generation mit einer höheren Generationsnummer wird die Datei automatisch auf dem Datenträger eingerichtet, auf dem sich die älteste Generation befand, die durch diesen Vorgang gelöscht wird.

AIM-Dateien auf Band

Das Einrichten der AIM-Dateigenerationsgruppe auf Band erfolgt durch Anweisungen an das Dienstprogramm LEASY-MAINTASK.

Mit der **ADE*-Anweisung ist der Gerätetyp und mit der **AVO*-Anweisung die Archivnummer des Bandes festzulegen. In der **LOG*-Anweisung muss der Wert *Y,M* angegeben werden, damit das Schreiben in die AIM-Datei von der Maintask durchgeführt wird. Die **ASP*-Anweisung (Operand *TAPE*) steuert das Weiterschalten auf die nächste AIM-Dateigeneration.

Folgende Regeln sind bei AIM-Dateien auf Band zu beachten:

- Auf einem Band darf nur eine AIM-Dateigeneration sein.
- Eine AIM-Dateigeneration darf sich nicht über mehrere Bänder erstrecken. Der Anwender ist selbst dafür verantwortlich, dass rechtzeitig auf die nächste Generation umgeschaltet wird.

  Die Angaben für **ADE* und **AVO* werden nur für die erste in einer Session angelegte AIM-Dateigeneration ausgewertet. Sollen weitere AIM-Dateigenerationen auf Bändern angelegt werden, kann dies auf folgende Arten erfolgen:

  – Erneutes Starten von LEASY-MAINTASK (**ADE*/**AVO*-Anweisung)

  – *CREATE-FILE-GROUP*-Kommando

  – *CREATE-FILE-GENERATION*-Kommando.

- Wird LEASY-MAINTASK mit einer ENTER-Prozedur gestartet, ist zu beachten, dass die CPU-Zeit, die im *ENTER-JOB*-Kommando mit angegeben werden kann, bei AIM auf Band größer zu wählen ist als bei AIM auf Platte.

Fehler beim Schreiben in die AIM-Datei

Falls beim Schreiben in die AIM-Datei ein DVS-Fehler auftritt, kann keine AIM-Protokollierung mehr durchgeführt werden. Die aktuelle AIM-Dateigeneration ist dann in der Regel unbrauchbar.

In einem solchen Fall übermittelt LEASY den Returncode `99ALLS75` an das Anwenderprogramm und setzt die Transaktion zurück. Jede weitere LEASY-Anforderung erhält Returncode `99ALLS81` mit folgender Bedeutung: AIM-Datei kann wegen eines Fehlers nicht mehr beschrieben werden, keine weitere LEASY-Anforderung mehr erlaubt, Transaktion wurde von LEASY zurücksetzt.

Folgende Schritte sind erforderlich, damit LEASY wieder ordnungsgemäß ablaufen kann:

- Beendigung von LEASY-MAINTASK
- Sicherung des Originaldatenbestands
- Ermittlung der Fehlerursache und ggf. Vorkehrungen, um das erneute Auftreten des Fehlers zu verhindern (z.B. bei Plattenspeichermangel genügend freien Plattenspeicher zur Verfügung stellen)

Die weiteren Schritte sind abhängig davon, ob die LEASY-Session, in der das DVS-Problem beim Schreiben der AIM-Datei-Generation auftrat, mit automatischem Nachziehen von Schattendateien betrieben wurde oder nicht:

- ohne automatisches Nachziehen von Schattendateien
  - Sicherstellen, dass mindestens eine AIM-Dateigeneration frei ist, damit ein Umschalten durchgeführt werden kann.
  - Neustart der LEASY-MAINTASK mit zusätzlicher Angabe des Parameters ASP, damit auf eine neue AIM-Dateigeneration umgeschaltet wird.
- mit automatischem Nachziehen von Schattendateien
  - Erzeugung korrekter Schattendateien durch Kopieren der Sicherung des Originaldatenbestandes.
  - Neustart der LEASY-MAINTASK wieder mit AIM-Protokollierung und automatischem RECONST.

5.2.2 AIM-Dateigenerationen freigeben

LEASY nutzt beim Anlegen von AIM-Dateien Dateigenerationen auf folgende Weise: Mit der LEASY-MAINTASK-Anweisung **AGE* wird eine Maximalzahl von AIM-Generationen festgelegt. Solange dieser Grenzwert nicht erreicht ist, wird bei Bedarf eine neue Generation angelegt. Wenn der Grenzwert jedoch erreicht ist, wird die bisher älteste Generation freigegeben und ihr Katalogeintrag gelöscht. Damit bei diesem Vorgang keine Daten aus der ältesten AIM-Generation verloren gehen können, werden die AIM-Generationen ab LEASY V6.2 standardmäßig gegen Freigeben geschützt. Bevor ein Umschalten auf die älteste AIM-Generation durchgeführt werden kann, muss diese freigegeben werden. Die dazu notwendigen Maßnahmen hängen im Wesentlichen davon ab, ob mit automatischem Nachziehen von Schattendateien gearbeitet wird.

Bei Bedarf kann das bisherige Verhalten, bei dem die zuletzt bearbeitete AIM-Dateigeneration nach dem erfolgreichen Umschalten sofort freigegeben wird, mit der LEASY-MAINTASK-Anweisung **FAA* eingestellt werden.

Arbeiten ohne automatisches Nachziehen von Schattendateien

Wenn nicht mit automatischem Nachziehen von Schattendateien gearbeitet wird (Angabe **AUT=N* bzw. keine **AUT*-Angabe im Dienstprogramm LEASY-MAINTASK), muss der LEASY-Administrator die nicht mehr benötigten Dateigenerationen freigeben (Funktion *AIMA* des Dienstprogramms LEASY-MASTER oder **AGF*-Anweisung im LEASY-MAINTASK).

ACHTUNG!

Der LEASY-Administrator ist selbst dafür verantwortlich, dass er nur AIM-Dateigenerationen freigibt, die bereits gesichert oder nachgefahren sind.

Wenn der LEASY-Administrator die AIM-Dateigenerationen nicht rechtzeitig freigibt, kann der Fall eintreten, dass keine Dateigenerationen mehr angelegt werden können. In diesem Fall können keine Umschaltvorgänge mehr durchgeführt werden. Wird in dieser Situation dennoch versucht, die Dateigeneration umzuschalten, sind folgende Fälle zu unterscheiden:

- Explizites Umschalten mit dem Dienstprogramm LEASY-MASTER

  Explizite Umschaltvorgänge mit den Anweisungen *AIMC*, *AIMI* oder *AIMW* des Dienstprogramms LEASY-MASTER werden mit entsprechenden Meldungen abgewiesen. Die zuletzt verwendete AIM-Dateigeneration wird weiter benutzt.

  ► Der LEASY-Administrator muss mit der LEASY-MASTER-Anweisung *AIMA* nicht mehr benötigte AIM-Dateigenerationen freigeben. Anschließend muss er den geplanten Umschaltvorgang wiederholen.

- Implizites Umschalten wegen LEASY-MAINTASK-Anweisung **AIS*

  Umschaltvorgänge, die erforderlich sind, weil die mit dem Operanden *pamblocknummer* der LEASY-MAINTASK-Anweisung **AIS* angegebene Dateigröße erreicht ist, werden mit entsprechenden Meldungen abgewiesen. Die zuletzt verwendete AIM-Dateigeneration wird weiter benutzt.

  Das weitere Verhalten hängt von der Angabe *inkrement* in der Anweisung **AIS* ab:

  – ist *inkrement* nicht angegeben, wird die aktuelle AIM-Dateigeneration weiter benutzt, bis die maximale Dateigröße von 16775000 PAM-Blöcken erreicht ist.

  – ist *inkrement* angegeben, wird *pamblocknummer* um diesen Wert erhöht. Die aktuelle AIM-Dateigeneration wird weiter benutzt, bis dieser neue Wert erreicht ist. Dann wiederholt sich der oben beschriebene Vorgang.

  Wenn *inkrement* den Wert 0 hat oder wenn die maximale Dateigröße von 16775000 PAM-Blöcken erreicht ist, kann nicht mehr in die aktuelle AIM-Dateigeneration geschrieben werden. LEASY-MAINTASK gibt eine entsprechende Meldung (*LEA5012*) aus. Alle weiteren LEASY-Anweisungen werden mit dem Returncode 99ALLS75 abgewiesen.

  ► Der LEASY-Administrator muss mit der LEASY-MASTER-Anweisung *AIMA* nicht mehr benötigte AIM-Dateigenerationen freigeben. Der nächste Umschaltvorgang kann erfolgreich vorgenommen werden. Nach dem erfolgreichen Umschalten wird der Wert für *pamblocknummer* auf den ursprünglich festgelegten Wert zurückgesetzt.

- Implizites Umschalten der AIM-Dateigeneration wegen LEASY-MAINTASK-Anweisung *ASP* oder wegen Versionswechsel

  Ist ein Umschaltvorgang erforderlich, weil die LEASY-MAINTASK-Anweisung **ASP* angegeben ist oder weil ein Versionswechsel von LEASY aus einer Version < V6.1 stattgefunden hat, beendet sich das Dienstprogramm LEASY-MAINTASK abnormal.

  ► Der LEASY-Administrator muss mindestens eine AIM-Dateigeneration freigeben. Die eigentlich hierfür vorgesehene Anweisung *AIMA* des Dienstprogramms LEASY-MASTER steht jedoch erst nach erfolgreichem Start des Dienstprogramms LEASY-MAINTASK zur Verfügung.

  Abhängig vom aktuellen Wert der LEASY-MAINTASK-Anweisung **AGE* kann das Problem folgendermaßen gelöst werden:

  1. Wert < 255

     Der LEASY-Administrator erhöht diesen Wert mindestens um 1 und startet LEASY-MAINTASK erneut.
     Das Umschalten der AIM-Dateigeneration kann nun erfolgreich durchgeführt werden, da mindestens eine AIM-Dateigeneration angelegt werden kann.

2. Wert = 255

   Da der Wert nicht mehr erhöht werden kann, muss der LEASY-Administrator mit der Anweisung **AGF* im Dienstprogramm eine Anzahl von Dateigenerationen festlegen, die freigegeben werden sollen. Diese Anweisung ist nur bei **AGE=255* erlaubt. Maximal können auf diese Weise 254 Dateigenerationen freigegeben werden.

   Anschließend muss der LEASY-Administrator LEASY-MAINTASK erneut starten.

Arbeiten mit automatischem Nachziehen von Schattendateien

Wenn im Dienstprogramm LEASY-MAINTASK automatisches Nachziehen von Schattendateien festgelegt ist (**AUT=Y*), gibt LEASY-MAINTASK automatisch nach jeder Aktualisierung der Schattendateien die nicht mehr benötigte AIM-Generation frei. Eingriffe des LEASY-Administrators sind in diesem Fall in der Regel nicht erforderlich.

Dieses automatiche Freigeben ist jedoch nur möglich, wenn für **alle** Dateien im LEASY-Katalog automatisches Mitführen von Schattendateien vereinbart wurde (*AIM=(Y,A)* oder *AIM=(R,A)* in der **FIL*-Anweisung des Dienstprogramms LEASY-CATALOG).

Falls der LEASY-Katalog mindestens eine Datei enthält, für die zwar AIM-Logging, aber kein automatisches Mitführen von Schattendateien vereinbart wurde (*AIM=Y* oder *AIM=R*), würden deren Informationen nach dem Freigeben der AIM-Generation verloren gehen. Falls *AUT=Y* angegeben ist, prüft LEASY-MAINTASK deshalb, ob Dateien existieren, die mit *AIM=Y* oder *AIM=R* vereinbart sind.

Das weitere Verhalten ist abhängig von der Angabe in der Anweisung *FAA:*

- *FAA=N* (Standard)

  LEASY-MAINTASK wird abnormal beendet und weist mit einer Meldung auf den drohenden Datenverlust hin. Der LEASY-Administrator muss dann mit dem Dienstprogramm LEASY-CATALOG alle betroffenen Dateien mit *AIM=N*, *AIM=(Y,A)* oder *AIM=(R,A)* vereinbaren und gegebenenfalls noch Schattendateien anlegen. Danach kann er das Dienstprogramm LEASY-MAINTASK erneut starten.

  i Dateien, die mit *AIM=Y* oder *AIM=R* vereinbart wurden, können auch mit der Anweisung **INF ,M* des Dienstprogramms LEASY-CATALOG ermittelt werden.

- *FAA=Y* (Verhalten wie LEASY der Version ≤ V6.1)

  LEASY-MAINTASK weist zwar mit einer Meldung auf den drohenden Datenverlust hin, arbeitet aber normal weiter.

5.2.3 AIM-Elemente

Die einzelnen Einträge in der AIM-Datei (AIM-Elemente) sind variabel lang und werden fortlaufend aneinander geschrieben. Die Elementeinträge sind in Vorwärts- und Rückwärtsrichtung miteinander verkettet. Jedes AIM-Element besteht aus 2 Teilen:

- **variabler** Eintrag

  In diesen Teil schreibt das LEASY-Laufzeitsystem die Daten, die sich bei jeder LEASY-Operation unterscheiden, z.B. die Dateinamen beim physikalischen Eröffnen von Dateien oder Dateiname und die neuen Satz- bzw. Blockinhalte bei Veränderungen.

- **fester** Eintrag

  In diesen Teil werden die Informationen geschrieben, die je Aktion nötig sind, um die Datei rekonstruieren zu können. Dabei handelt es sich um den Operationscode, die TSN-Nummer, den Zeitzähler, die Session- und Transaktionsnummer, die interne LEASY-Transaktionsnummer und 2 Elementlängenfelder.

Die nachfolgend aufgeführten Namen der AIM-Elemente werden bei einem Rekonstruktionslauf mit dem Dienstprogramm LEASY-RECONST in die Systemdatei *SYSLST* protokolliert.

Name	Aktion	Bedeutung
MTSK	Maintask-Eintrag	Start von LEASY-MAINTASK
SESS	Session-Eintrag	Beginn einer neuen Session (LEASY-MAINTASK)
CATD	CATD-Eintrag	Anschluss an Common Memory
OPEN	OPEN-Eintrag	Physikalisches Eröffnen von Dateien
CLOS	CLOSE-Eintrag	Physikalisches Schließen von Dateien
OPTR	OPTR-Eintrag	Beginn einer neuen Transaktion
CLTR	CLTR-Eintrag	Ende einer Transaktion
RLBK	Rollback-Eintrag	Beginn eines Rollback
STOR[1]	STORE/PUT-Eintrag	Hinzufügen eines Satzes bei ISAM, DAM oder SAM oder Blocks bei PAM
DLET[2]	DLET-Eintrag	Löschen eines ISAM- oder DAM- Satzes oder Blocks einer PAM-Datei
PUTX	PUTX-Eintrag	Überschreiben eines ISAM- oder DAM-Satzes oder Blocks einer PAM- Datei
PUTS	PUTXSAM-Eintrag	Überschreiben eines SAM-Satzes
ELIF	ELIMFILE-Eintrag	Löschen einer ganzen ISAM-, DAM- oder PAM-Datei
ELIR	ELIMREC-Eintrag	Löschen ISAM-, DAM- oder PAM-Datei ab einem bestimmten Schlüssel

Tabelle 4: Bei einem Rekonstruktionslauf protokollierte AIM-Elemente (Teil 1 von 2)

Name	Aktion	Bedeutung
SETS	SETLSAM-Eintrag	Dateiverkürzung einer SAM-Datei durch *SETL*
ENDA	Ende der AIM-Datei-Eintragung	Maintask hat AIM-Datei während der Session umgeschaltet
CSES	Continue-Session-Eintrag	Maintask führt Session in neu angelegter AIM-Datei weiter
OLDB	Old-Buffer-Eintrag	Maintask konnte nach dem Umschalten auf die neue AIM-Datei den *ENDA*-Eintrag in die alte AIM-Datei wegen eines E/A-Fehlers nicht schreiben. Sie rettet daher den noch nicht geschriebenen Inhalt des AIM-Puffers im CMMAIN in die neue AIM-Datei hinüber und schließt ihn mit dem *OLDB*-Element ab
CTSK	Continue-Task-Eintrag	Eine LEASY-Task hat sich an die neu eingerichtete AIM-Datei angeschlossen
FILS	Files-Liste-Eintrag	Eine LEASY-Task hat zum Zeitpunkt des Anschlusses an die neue AIM-Datei die im *FILS*-Eintrag aufgeführten Dateien physikalisch eröffnet
PETR	PETR-Eintrag	Vorläufig beendete Transaktion
STOD	Store-DAM-Buffer-Eintrag	Hinzufügen oder Überschreiben eines Blockes einer DAM-Datei
CINF	CINF-Eintrag	Currency Information übergeben
LOCK	LOCK-Eintrag	Satzsperre setzen
RDIR	RDIR-Eintrag	Satz direkt lesen
RHLD	RHLD-Eintrag	Satz direkt lesen mit Satzsperre
RNXT	RNXT-Eintrag	Nächsten Satz lesen
RNHD	RNHD-Eintrag	Nächsten Satz lesen mit Satzsperre
RPRI	RPRI-Eintrag	Vorhergehenden Satz lesen
RPHD	RPHD-Eintrag	Vorhergehenden Satz lesen mit Satzsperre
UNLK	UNLK-Eintrag	Satzsperre aufheben

Tabelle 4: Bei einem Rekonstruktionslauf protokollierte AIM-Elemente (Teil 2 von 2)

[1] Die erste Zeile der AIM-Datei enthält die ersten 24 Byte des Primärschlüssels. Die zweite Zeile enthält den Inhalt des Satzes ab Byte 1 (einschließlich Primärschlüssel).

[2] Hier wird nur der Primärschlüssel eingetragen.

i Es wird nur dann ein Satz in die AIM-Datei eingetragen, wenn der entsprechende LEASY-Aufruf fehlerfrei abgelaufen ist.

Einzige Ausnahme bildet hier der Returncode LP11 (angegebener *CINF*-Bereich zu klein) bei der Operation *CINF*. Tritt dieser Fehler auf, wird trotzdem ein AIM-Satz geschrieben, da der *CINF*-Aufruf auch bei LP11 Teilinformationen liefert.

Tritt bei einem solchen Aufruf zusätzlich ein Fehler beim AIM-Schreiben auf, dann wird der *CINF*-Returncode (LP11) im Feld *RC-LC* (RE-Bereich) und der AIM-Schreib-Returncode im Feld *RC-LCE* (RE-Bereich) angezeigt.

Verkürzte AIM-Elemente

Der Anwender hat die Möglichkeit, nur die Teile eines Satzes in die AIM-Datei schreiben zu lassen, die tatsächlich geändert wurden. Mit Hilfe von Längenfeldern, die Auskunft über die unveränderten Satzteile geben, und der zugehörigen Sicherungsdatei, ist es dem Dienstprogramm LEASY-RECONST möglich, den geänderten Satz zusammenzufügen.

Dadurch wird eine Einsparung an Speicherplatz erreicht, die abhängig ist vom Verhältnis geänderter Satzteile zu unveränderten Satzteilen.

– Das Performance-Verhalten ist abhängig davon, wie die Transaktionen aussehen: Bei einer Transaktion *OPTR ... RHLD ... REWR ... CLTR* vergrößert sich die Pfadlänge um das Erzeugen des verkürzten Satzes, denn eine Ausgabe auf die AIM-Datei ist bei jedem *CLTR* nötig.

– Liegen jedoch Transaktionen mit mehreren *REWR*-Aufrufen mit evtl. langen Sätzen vor, so kann ein Performance-Gewinn eintreten, da evtl. ein oder mehrere Ausgaben auf die AIM-Datei eingespart werden.

Im Rekonstruktionsfall tritt auf jeden Fall ein Performance-Verlust ein. Erkennt LEASY-RECONST einen verkürzten Satz, so muss erst aus der Sicherungsdatei der unverkürzte Satz gelesen werden, damit dann mit der Information aus dem AIM-Element der veränderte Satz zusammenfügt werden kann.

Ob verkürzte oder vollständige AIM-Sätze geschrieben werden, legt der Anwender durch Versorgen des entsprechenden Operanden im Dienstprogramm LEASY-CATALOG fest (Operand *AIM* in der **FIL*-Anweisung).

Eine einmal getroffene Entscheidung, eine Datei durch verkürzte AIM-Sätze zu sichern, kann mit dem Operanden *AIM* in der **FIL*-Anweisung des Programms LEASY-CATALOG wieder rückgängig gemacht werden.

Sicherungsverfahren mit der AIM-Datei

Das Sicherungskonzept beruht auf folgender Vorgehensweise:

- Zu bestimmten Zeitpunkten müssen die Dateien des Benutzers auf andere Datenträger gesichert werden. Dies kann z.B. mit dem *COPY-FILE*-Kommando (siehe Handbuch „Kommandos Band 1 - 5“), mit dem Dienstprogramm ARCHIVE (siehe Handbuch „ARCHIVE (BS2000/OSD)“) oder dem Dienstprogramm LEASY-SAVE geschehen.

  Für SAM-Dateien sind Sicherungen mit Dateiumsetzern oder sonstigen Programmen, die satzweise lesen, verboten, weil dadurch die SAM-Wiedergewinnungsadressen der Sätze verändert werden können.

- Zerstörte Dateien werden mit einer Sicherungsdatei überschrieben. Danach arbeitet das Dienstprogramm LEASY-RECONST die AIM-Dateien vom Zeitpunkt der Sicherung bis zum aktuellen Stand ab.

  Durch entsprechende Operanden des Dienstprogramms LEASY-RECONST können dabei die Dateien und/oder der Beginn der Rekonstruktion (Datum oder Sessionnummer) ausgewählt werden.

- Der Zeitpunkt der Dateisicherung muss nicht zwingend mit dem Anlegen einer neuen Generation der AIM-Datei zusammenfallen, da bei der Rekonstruktion sowohl ein Ausschnitt aus einer AIM-Datei gewählt werden kann als auch mehrere aufeinander folgende AIM-Dateien nacheinander in die Datei des Benutzers eingearbeitet werden können.

  Die Dateisicherung sollte jedoch nicht während einer LEASY-Session erfolgen, sondern nur zwischen 2 Sessions, d.h. vor dem Start von LEASY-MAINTASK.

5.3 Schattendateikonzept

Das Schattendateikonzept sieht vor, dass parallel zu den Originaldateien ein Satz von Kopiedateien mitgeführt wird. Zusätzlich werden AIM-Dateien angelegt.

Von den zu bearbeitenden Originaldateien werden Kopien, sog. Schattendateien, erzeugt. Die anschließend vorgenommenen Änderungen an der Originaldatei werden in die AIM-Datei eingetragen. Dann wird die AIM-Sicherung umgeschaltet, d.h. die Änderungen in eine neue AIM-Dateigeneration eingetragen.
Anschließend wird die Schattendatei mit der abgeschlossenen AIM-Dateigeneration auf den aktuellen Stand gebracht. Die Differenz zwischen der Schattendatei und dem Original ist also maximal der Inhalt einer AIM-Dateigeneration. Wird der Inhalt der aktuellen AIM-Dateigeneration gering gehalten, so ist also auch die Differenz gering. Bei einer Dateizerstörung ist dann eine schnelle Rekonstruktion möglich.

5.3.1 Schattendateien anlegen

Mit dem Dienstprogramm LEASY-CATALOG wird die Namenskonvention der Schattendateien für den gesamten Katalog festgelegt (siehe Operanden *CPC* und *CPS* in der **CAT*-Anweisung).

Die Schattendateien muss der Anwender selbst anlegen.

Wird das Schattendateikonzept verwendet, so muss für jede Anwenderdatei, für die AIM-Sicherung verlangt wird, auch die Schattendatei erzeugt werden. Auch für SI-Dateien sind Schattendateien zu erzeugen.

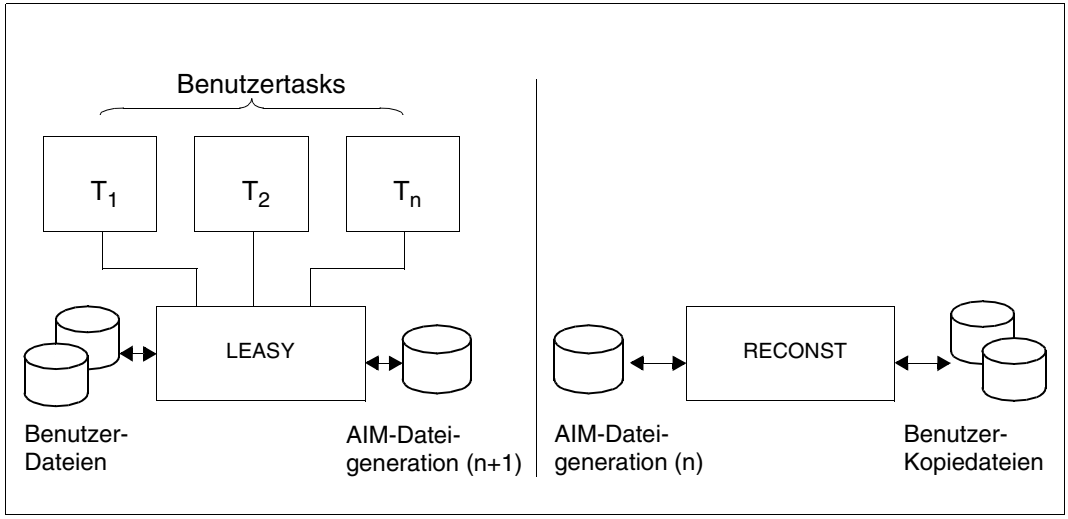


Bild 3: Schattendateien

Zur Unterstützung des Schattendateikonzepts muss es möglich sein, während einer laufenden LEASY-Session die aktuelle AIM-Datei abzuhängen und in einer neuen (leeren) Datei weiterzuschreiben. Hierzu wird von der aktuellen Generation auf die nächste AIM-Dateigeneration umgeschaltet.

Das Umschalten der AIM-Datei kann ausgelöst werden durch:

- Erreichen einer vorgegebenen Maximalgröße der AIM-Datei. Diese Größe wird für jede LEASY-Session über die Anweisung **AIS* im Dienstprogramm LEASY-MAINTASK eingestellt.
- Verwendung der Anweisungen *AIMI*, *AIMC* oder *AIMW* im Dienstprogramm LEASY-MASTER.

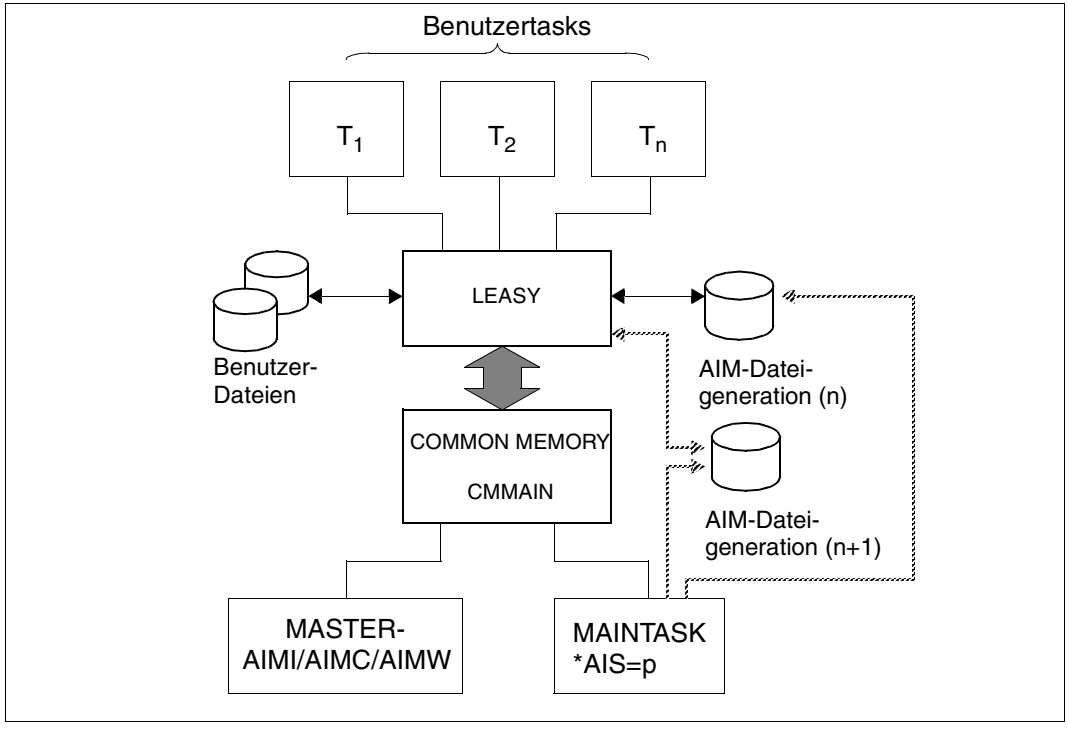


Bild 4: Umschalten auf eine neue AIM-Dateigeneration

Zum Umschalten auf die neue AIM-Dateigeneration wird automatisch die LEASY-Maintask. angestoßen.

Wird der AIM-Puffer von der Maintask in die AIM-Datei geschrieben, schreibt die Maintask nach dem Umschalten in die neue Generation. Wird der AIM-Puffer von den Anwendertasks in die AIM-Datei geschrieben, muss sich jedes an die alte AIM-Datei angeschlossene Anwenderprogramm an die neue Generation anhängen. Das geschieht für jede Task beim nächsten LEASY-Aufruf. Bis zu diesem nächsten LEASY-Aufruf ist die alte Generation von den Tasks noch eröffnet und kann daher noch nicht für eine Rekonstruktion verwendet werden.

Beim Umschalten werden die Elemente *ENDA, CSES, OLDB, CTSK* und *FILS* in die AIM-Datei geschrieben. Dabei wird folgende Reihenfolge eingehalten:
Zuerst schreibt die Maintask *CSES* in die neue AIM-Datei, dann *ENDA* in die alte AIM-Datei bzw. bei Fehler (PAM-Makro) *OLDB* in die neue AIM-Datei. Jede LEASY-Anwendertask schreibt *CTSK* (+*FILS*) in die neue AIM-Datei.

Es besteht somit die Möglichkeit, bei Auftreten von Ein-/Ausgabefehlern beim Schreiben in die AIM-Datei mit einer Anweisung an das Programm LEASY-MASTER bei laufender Session auf eine neue AIM-Datei umzuschalten, ohne dass evtl. noch nicht in die AIM-Datei geschriebene Elemente aus dem im CMMAIN gelegenen AIM-Puffer verloren gehen.

Der Übergang auf eine neue AIM-Datei kann auch während offener Transaktionen stattfinden.

5.3.2 Maximale Ausfallsicherheit

Maximale Ausfallsicherheit bei minimaler Wiederanlaufzeit erreichen Sie durch Schattendateien, die entsprechend dem folgenden Schema auf zwei Pubsets verteilt sind.

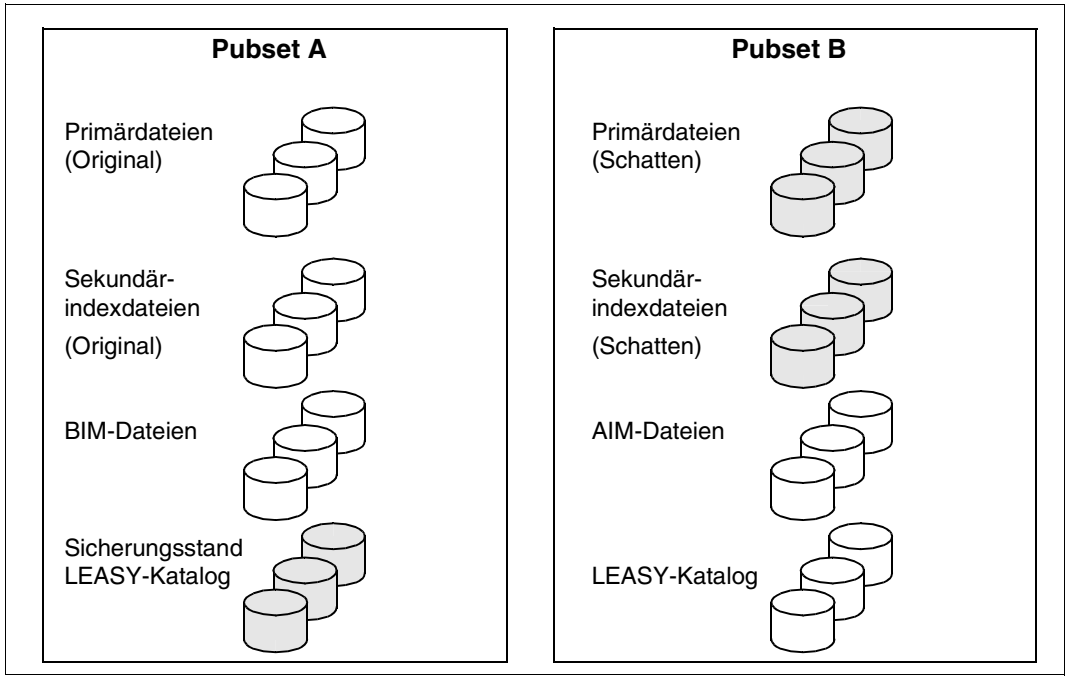


Bild 5: Auf zwei Pubsets verteilte Schattendateien

Diese Konfiguration können Sie folgendermaßen erreichen:

Pubset A ist das Default-Pubset, Pubset B ist ein weiteres Pubset, das für die Sicherung zur Verfügung steht.

```
/START-LEASY-CATALOG
```

(Starten von LEASY-CATALOG).

```
*CAT :B:dateikatalog,TYP=N,CPC=:B:copycat[,CID=YES]
```

Die CAT-Anweisung legt den Katalog auf Pubset B an. Auch die Schattendateien sollen auf Pubset B liegen. Statt der Angabe *CPC* kann auch der Parameter *CPS* verwendet werden, um die Namen der Schattendateien festzulegen. Stellen Sie sicher, dass der Parameter *CID* auf *YES* gesetzt wird, damit LEASY-MAINTASK später die Parameter *ACA* bzw. *BCA* auswertet. Eine explizite Angabe des Parameters ist im Normalfall nicht erforderlich.

Mit *FIL*-Anweisungen legen Sie anschließend die Benutzerdateien an (eine Anweisung pro Datei):

```
*FIL datei,NAM=:A:dateiname,AIM=(Y,A),...
```

Damit die Originaldateien auf Pubset A angelegt werden, ist der Parameter *NAM* notwendig. Für *dateiname* wird die Namenskonvention *dateikatalog.datei* empfohlen. Die AIM-Sicherung wird auf automatisches Rekonstruieren eingestellt.

```
*END
```

Mit der Anweisung END wird LEASY-CATALOG beendet.

```
/COPY-FILE :A:originaldatei,:B:schattendatei
```

```
/COPY-FILE :A:original-si-datei,:B:schatten-si-datei
```

Die Schattendateien werden durch Kopieren der Originaldateien auf das Pubset B angelegt.

```
/COPY-FILE :B:dateikatalog.LEASYCAT,:A:dateikatalog.LEASYCAT.SAVE
```

Der LEASY-Katalog wird durch diese Eingabe auf das Pubset A gesichert. Diese Sicherung muss nach jeder Änderung im Katalog wiederholt werden.

```
/START-LEASY-MAINTASK
```

Starten von LEASY-MAINTASK mit folgenden Parametern:

```
*CAT=:B:dateikatalog
*ACA=B
*BCA=A
*LOG=Y
*AUT=Y
*REN=enter-kommando
*AGE=3(oder > 3)
.
.
.
*END
```

5.3.3 Reparaturmaßnahmen

Falls auf einem Pubset ein Defekt auftritt, sind die beiden, im Folgenden beschriebenen, Fälle bei der Planung von Reparaturmaßnahmen zu unterscheiden:

– Das defekte Pubset ist in angemessener Zeit nicht reparierbar.

– Das defekte Pubset kann in angemessener Zeit repariert werden.

Defektes Pubset ist in angemessener Zeit nicht reparierbar

Die neue LEASY-Session wird auf dem intakten Pubset neu gestartet, ohne Schattendateien zu führen. Dadurch kann die Arbeit rasch wieder aufgenommen werden.

- Pubset A ist defekt

  Maßnahmen:

  1. Pubset B zum Default-Pubset machen.
  2. Im Katalog bei Pubset B *CID=NO* setzen.
  3. Schattendateien in Originaldateien umbenennen.
  4. Reconst-Lauf der angeschlossenen Transaktionen mit noch nicht nachgezogenen AIM-Generationen durchführen (*MOD TRA=V*).
  5. Neue Session mit Katalog, AIM-Dateien und Originaldateien auf B starten.

- Pubset B ist defekt

  Maßnahmen:

  1. Im Katalog bei Pubset A *CID=NO* setzen.
  2. Originaldateien durch Warmstart mit Katalog A - ohne AIM-Schreiben - in einen konsistenten Zustand bringen.
  3. Maintask beenden.
  4. Originaldateien sichern.
  5. Neue Session mit Katalog, mit neuen AIM-Dateien und Originaldateien auf A starten.

Defektes Pubset kann in angemessener Zeit repariert werden

Zuerst sind die Reparaturmaßnahmen abzuschließen.

Anschließend kann die Session mit Schattendatei-Sicherung in folgender Weise wieder gestartet werden:

- Pubset A war defekt
  1. BIM-Dateien löschen.
  2. Warmstart der neuen Session. Die Schattendateien werden dabei automatisch nachgezogen.
  3. Schattendateien auf die Originaldateien kopieren.
- Pubset B war defekt
  1. Katalog auf B mit dem Katalog auf A überschreiben.
  2. Warmstart der neuen Session ohne automatischen RECONST. Dadurch wird die Konsistenz der Originaldateien wieder hergestellt.
  3. AIM-Generationen löschen.
  4. LEASY-MAINTASK beenden.
  5. Schattendateien mit den Originaldateien überschreiben.
  6. Original- und Schattendateien sichern.
  7. Neue Session mit automatischem RECONST starten.

5.3.4 Nachziehen einer AIM-Dateigeneration

Das Nachziehen der alten AIM-Dateigeneration auf die Schattendateien, d.h. das Aktualisieren der Schattendateien, kann entweder vom Benutzer oder automatisch von LEASY durchgeführt werden. Das Nachziehen der alten AIM-Dateigeneration ist mit dem Dienstprogramm LEASY-RECONST möglich, sobald alle Tasks die alte AIM-Datei geschlossen haben. Dazu ist in der *CAT*-Anweisung der Operand *COPY=YES* anzugeben.

Werden AIM-Dateigenerationen automatisch nachgezogen, übernimmt LEASY folgende Aktionen:

– Abwarten, bis eine AIM-Dateigeneration nachgezogen werden kann.

– Starten des Dienstprogramms LEASY-RECONST mit geeigneten Anweisungen.

– Warmstart mit Hilfe der AIM-Datei.

Die AIM-Dateigenerationen werden nur dann automatisch nachgezogen, wenn der Benutzer Schattendateien angelegt hat und bei den Dienstprogrammen LEASY-CATALOG und LEASY-MAINTASK folgende Anweisungen angegeben werden:

Dienstprogramm LEASY-CATALOG	**Bedeutung**
CAT dateikatalog,...,CPC=...[,CPS=...]	Namen der Schattendateien festlegen
FIL datei,...,AIM={(Y,A) / (R,A)}	Automatisches Mitführen der Schattendateien vereinbaren

Dienstprogramm LEASY-MAINTASK	**Bedeutung**
LOG={A,<u>M</u> / Y,<u>M</u>}	Session mit AIM-Sicherung und Schreiben der Sätze durch die Maintask
AUT=Y	Automatische Rekonstruktion
REN=enter-kommando	ENTER-Kommando für die RECONST-Task
[PAS=kennwort]	Kennwörter für die RECONST-Task
AGE ≥ 3	Anzahl der AIM-Dateigenerationen sollte ≥ 3 sein, um einen sicheren Betrieb mit automatischem Nachziehen zu gewährleisten

Bei der Arbeit mit LEASY-Schattendateien sollten Sie die folgenden Punkte berücksichtigen:

– Bei der Betriebsart automatisches Mitführen von Schattendateien sollte eine Session soweit wie möglich mit CLOS beendet werden. Dadurch ist ein konsistenter Zustand von Schatten-, Original- und AIM-Dateien gewährleistet.

– LEASY-Anwendungen mit lang laufenden Transaktionen sind nicht für die Betriebsart automatisches Mitführen von Schattendateien geeignet. Transaktionen sollten sich über maximal 2 AIM-Dateigenerationen erstrecken.

– Es sind mindestens 3 AIM-Dateigenerationen für einen ordnungsgemäßen und sicheren Betrieb erforderlich.

– Das Löschen der AIM-Dateigenerationen sollte mit dem Dienstprogramm LEASY-MASTER (Anweisung *AIME*) durchgeführt werden. Es können nur Generationen gelöscht werden, die nicht von LEASY geöffnet sind. Sollen alle Generationen gelöscht werden (Antwort „WHOLE“), so muss LEASY-MAINTASK mit *LOG=A,R* oder *LOG=Y,R* gestartet werden. Es dürfen keine Funktionen zum Steuern der Maintask (TERM/CLOS/SHUT) und keine Anwendertasks, die AIM-Sicherung verwenden, aktiv sein.

Ein Beispiel für automatisches Nachziehen der AIM-Dateigenerationen finden Sie im Kapitel „Anwendungsbeispiele“ ab Seite 382.

AIM-Verwaltungssatz

Für die AIM-Datei wird im LEASY-Katalog ein eigener AIM-Verwaltungssatz geführt. Er enthält Einträge über die Anzahl und den Zustand der AIM-Dateigenerationen.

Folgende Zustände sind möglich (siehe Bild 6 auf Seite 77):

GENFREE Die AIM-Datei wurde neu angelegt oder das Nachziehen der AIM-Dateigenerationen wurde erfolgreich abgeschlossen.
Wird die AIM-Datei von der Maintask erstmalig angelegt, sind alle Generationen im Zustand GENFREE. Ist eine LEASY-Anwendung aktiv, muss die AIM-Datei in einem konsistenten Zustand sein, wenn auf automatisches Mitführen umgestellt wird. Konsistenter Zustand bedeutet, dass alle Schattendateien von der AIM-Dateigeneration aktualisiert werden und in die Originaldateien kopiert wurden. Wurde der Inhalt einer AIM-Datei in die Schattendateien eingearbeitet, ist ebenfalls der Zustand *GENFREE* erreicht.

GENINUSE In dieser AIM-Dateigeneration werden momentan AIM-Sätze geschrieben. Beim Start einer LEASY-Anwendung erhält die „letzte“ Generation den Zustand *GENINUSE*. In diese Generation schreiben alle angeschlossenen Anwendertasks.

GENSWIT Die AIM-Dateigeneration wird umgeschaltet. Das Umschalten auf eine neue Generation muss der Benutzer mit dem Dienstprogramm LEASY-MASTER direkt veranlassen (Funktionen *AIMI/AIMC/AIMW*) oder mit dem Dienstprogramm LEASY-MAINTASK durch Einstellen einer maximalen Dateigröße indirekt veranlassen (*AIS*-Anweisung).

GENWAIT Nicht alle Transaktionen, die in diese AIM-Dateigeneration geschrieben haben, sind beendet.
Sobald auf die neue Generation umgeschaltet wurde, geht die alte Generation in den Zustand *GENWAIT* über.

GENREADY Alle Transaktionen, die in dieser AIM-Dateigeneration begonnen wurden, sind abgeschlossen.
Alle Transaktionen, die in diese Generation geschrieben haben, sind ordnungsgemäß beendet. Dabei ist ohne Bedeutung, ob die Transaktionen normal mit *CLTR* oder durch Rollback mit *CLTR* und Zusatzfunktion *OPE1=R* abgeschlossen wurden. Die Maintask kann im Zustand *GENREADY* ein Nachziehen der AIM-Dateigenerationen veranlassen.

GENRECO Die AIM-Dateigenerationsgruppe wird nachgezogen. Solange die AIM-Dateigeneration von LEASY-RECONST bearbeitet wird, befindet sie sich im Zustand *GENRECO*.

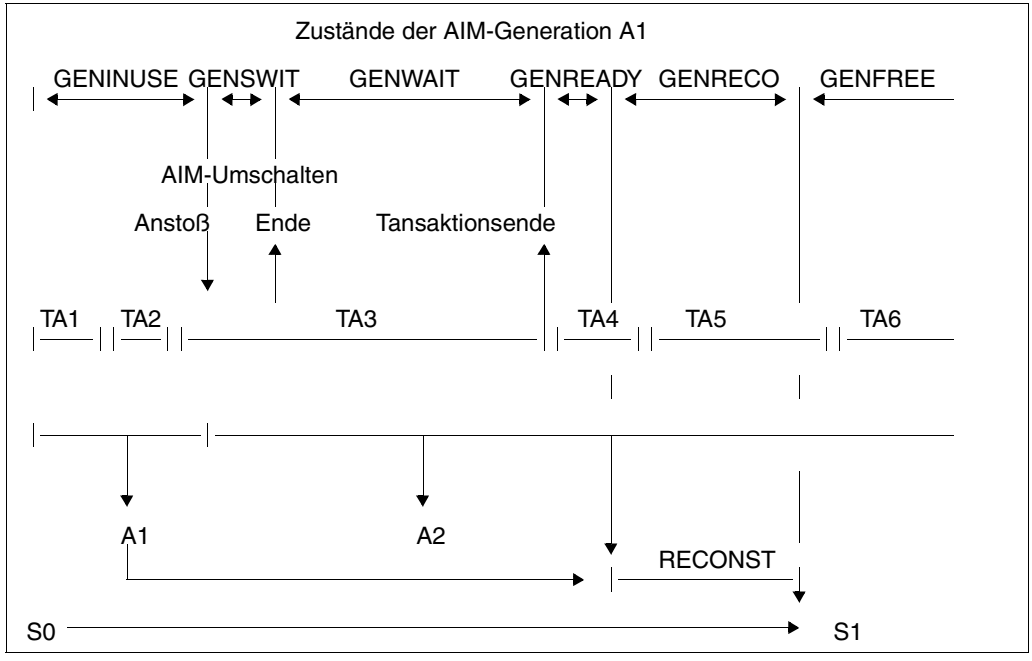


Bild 6: Zustände einer AIM-Dateigeneration

Erklärung

A1	AIM-Dateigeneration 1
A2	AIM-Dateigeneration 2
S0	Schattendatei Version 0
S1	Schattendatei Version 1
TA1-TA6	Transaktionen

LEASY-Session starten

Damit die AIM-Dateigenerationen automatisch nachgezogen werden, müssen beim Starten der Maintask im Dienstprogramm LEASY-MAINTASK die Anweisung *AUT=Y* und eine *REN*-Anweisung angegeben werden. Die *REN*-Anweisung für LEASY-MAINTASK bewirkt, dass LEASY-RECONST mittels eines *ENTER-JOB*-Kommandos als **eigene Task** (RECONST-Task) gestartet wird. Die im *ENTER-JOB*-Kommando angegebene Datei wird von LEASY-MAINTASK nur angelegt, wenn sie nicht schon existiert.

Ablauf von LEASY-MAINTASK

Nach der Parameteranalyse wird von LEASY-MAINTASK der CMMAIN aufgebaut. Dabei wird auch der AIM-Verwaltungssatz für die AIM-Datei aus dem LEASY-Katalog eingelesen. Anschließend wird eine Datei, wenn nicht vorhanden, mit dem vom Benutzer in der *REN*-Anweisung angegebenen Namen eingerichtet, mit den nötigen Kommandos und Anweisungen versorgt und mit dem CMD-Makro ein *ENTER-JOB*-Kommando abgesetzt.

Für die Kommunikation zwischen Maintask und RECONST-Task werden Börsen eingerichtet.

Der AIM-Verwaltungssatz wird mit dem Eintrag im DVS-Katalog verglichen und evtl. aktualisiert. Es wird eine neue AIM-Dateigeneration angelegt. Ist die AIM-Datei auf Platte, hat die neue Generation die vom Benutzer angegebene Größe (*ASP*=...) oder automatisch die gleiche Größe wie die vorhergehende Generation. Ist keine Generation frei, wird bereits jetzt ein Warmstart von LEASY-RECONST durchgeführt. Anschließend wird ein Kalt-/Warmstart, je nach Benutzerangabe, durchgeführt.

In einer Schleife wartet die Maintask an einer Börse auf das Eintreffen von folgenden Ereignissen:

– Ist das Umschalten auf eine neue AIM-Dateigeneration beendet (Anstoß von LEASY), wird der AIM-Verwaltungssatz auf den aktuellen Stand gebracht und LEASY-RECONST angestoßen.

– Ist das Nachziehen beendet (Mitteilung von LEASY-RECONST), wird die entsprechende AIM-Dateigeneration freigegeben.

Wird LEASY-MAINTASK mit der Anweisung *TERM* oder *SHUT* des Dienstprogramms LEASY-MASTER beendet, wird auch LEASY-RECONST sofort beendet. Wird die Maintask mit der Anweisung *CLOS* des Dienstprogramms LEASY-MASTER beendet, wartet die Maintask das Ende der letzten Transaktion ab, bevor sie die RECONST-Task mit dem Nachziehen der AIM-Dateigeneration beauftragt. Verlangt der Benutzer das Umschalten auf die nächste AIM-Dateigeneration, prüft die Maintask, ob diese frei ist.

Ablauf von LEASY-RECONST

In einer Schleife wartet die RECONST-Task auf Aufträge von der Maintask. Folgende Aufträge sind möglich:

Aufträge von LEASY-MASTER

TERM	Anweisung von LEASY-MASTER: RECONST-Task sofort beenden.
SHUT	Anweisung von LEASY-MASTER: RECONST-Task sofort beenden.
CLOS	Anweisung von LEASY-MASTER: Alle AIM-Dateigenerationen, die sich im Zustand GENREADY befinden, sind nachzuziehen.

LEASY-interne Aufträge

NACH	Nachziehen: Alle AIM-Dateigenerationen, die sich im Zustand GENREADY befinden, sind nachzuziehen.
KALT	Kaltstart: Untersuchen, ob sich Generationen im Zustand GENINUSE, GENWAIT oder GENSWIT befinden. Ist dies der Fall, Kaltstart ablehnen.
WARM	Warmstart: AIM-Verwaltungssatz untersuchen, in welchem Zustand sich die Generationen befinden und entsprechend verfahren (siehe unten unter Warmstart)

Kaltstart

Ein Kaltstart (Normalfall) ist nur möglich, wenn keine Generation im Zustand GENINUSE, GENWAIT oder GENSWIT ist und mindestens eine Generation frei ist. Noch nicht rekonstruierte AIM-Generationen werden parallel zum laufenden Betrieb nachgezogen.

Warmstart

Die BIM-Dateien sind nicht mehr das einzige Kriterium für einen Warmstart. Die Information, ob die letzte LEASY-Session ordnungsgemäß beendet wurde, enthält der AIM-Verwaltungssatz. Dort sind die Zustände der AIM-Dateigeneration abgespeichert. Ein Warmstart muss durchgeführt werden, wenn eine AIM-Dateigeneration nicht im Zustand *GENFREE* ist. Im Einzelnen führt LEASY-RECONST bei Rechnerausfall folgende Aktionen durch (siehe Bild 6 auf Seite 77). Die Zustände beziehen sich auf die AIM-Dateigeneration A1:

– Ausfall im Zustand *GENFREE*

  S1 ist der letzte Zustand der Schattendateien, die AIM-Dateigeneration 1 ist erfolgreich nachgezogen worden. Die abgeschlossenen Transaktionen aus Generation 2 müssen in die Schattendatei eingebracht, die offenen Transaktionen in A2 zurückgesetzt werden. D.h. LEASY-RECONST bearbeitet A2 mit *MOD TRA=V.*

– Ausfall im Zustand *GENRECO*

  Das Nachziehen der Generation A1 ist noch nicht abgeschlossen. In A1 sind nur Transaktionen offen, die in A2 beendet wurden. Beim Warmstart muss A1 vollständig eingebracht werden. A2 muss wie oben beschrieben rekonstruiert werden. D.h. LEASY-RECONST bearbeitet A1 mit *MOD TRA=A*, A2 mit *MOD TRA=V*.

– Ausfall im Zustand *GENREADY*

  Entspricht den Aktionen des Zustands *GENRECO*.

– Ausfall im Zustand *GENWAIT*

  In Generation A1 und A2 sind Einträge von Transaktionen, die zum Zeitpunkt des Ausfalls nicht abgeschlossen waren. Diese Einträge müssen zurückgesetzt werden. Danach werden beide Generationen in S0 eingebracht. D.h. LEASY-RECONST bearbeitet A1 und A2 mit *MOD TRA=V*.

– Ausfall im Zustand *GENSWIT*

  Entspricht den Aktionen für A2 im Zustand *GENFREE*.

– Ausfall im Zustand *GENINUSE*

  Entspricht den Aktionen für A2 im Zustand *GENFREE*.

Nach der erfolgreichen Rekonstruktion kann der Benutzer die Schattendateien in die Originaldateien kopieren.

LEASY-Session beenden

Zum Beenden der LEASY-Session stehen im Dienstprogramm LEASY-MASTER die Anweisungen *TERM*, *SHUT* und *CLOS* zur Verfügung. Diese Anweisungen wirken auf die Maintask. Diese reicht die Anweisungen weiter an die RECONST-Task. Bei *TERM* und *SHUT* beendet sich das Dienstprogramm LEASY-MAINTASK sofort, das Dienstprogramm LEASY-RECONST wird ebenfalls sofort beendet. Bei *CLOS* wartet LEASY-MAINTASK ab, bis LEASY-RECONST alle Generationen im Zustand *GENRECO* abgearbeitet hat, gibt diese frei (Zustand *GENFREE*) und beendet sich erst dann.

Maßnahmen bei Fehlern

Dieser Abschnitt beschreibt Maßnahmen, wenn Fehler beim automatischen Nachziehen von AIM-Dateigenerationen auftreten.

- Systemzusammenbruch

  Die neue LEASY-Session ist mit Warmstart zu starten.

- Fehlerhafte Beendigung einer Anwendertask

  Die neue LEASY-Session ist mit Warmstart zu starten.

- Fehlerhafte Beendigung der RECONST-Task

  Die LEASY-Anwendung kann weiterlaufen, solange freie Generationen verfügbar sind. Die neue LEASY-Session ist mit Warmstart zu starten.

- Zerstörung einer Datei

  Die LEASY-Session muss mit *CLOS* beendet werden. Dadurch werden die Schattendateien auf den aktuellen Stand gebracht. Nach Beendigung der LEASY-Session kann die zerstörte Datei durch Kopieren der Schattendatei wiederhergestellt werden. Die neue LEASY-Session ist mit Warmstart zu starten.

  Falls mit automatischem Mitführen von Schattendateien gestartet wurde und keine BIM-Sicherung vereinbart war, kopiert LEASY bei einem Warmstart (letzte Session wurde mit offenen Transaktionen fehlerhaft beendet) die rekonstruierten Schattendateien auf die Originaldateien. LEASY führt also den notwendigen Rollback mit den Informationen der AIM- und Schattendateien durch. Dieser Kopiervorgang kann bei großen Dateien einige Zeit beanspruchen. Kopiert werden alle Dateien, für die automatisches Mitführen von Schattendateien vereinbart wurde, und die an offenen Transaktionen beim Abbruch der Session beteiligt waren.

5.3.5 Originaldateien im laufenden Betrieb durch Schattendateien ersetzen

Das Schattendateikonzept ermöglicht die einfache Wiederherstellung von Dateien, die durch einen Fehler inkonsistent oder zerstört wurden. Sie können wieder in einen konsistenten Zustand versetzt werden, indem sie durch die entsprechenden Schattendateien ersetzt werden.

Die Funktion *REPO* des Dienstprogramms LEASY-MASTER ermöglicht es dem LEASY-Administrator, Schattendateien auf die zugehörigen Originaldateien zu kopieren, ohne dass alle LEASY-Anwendungen und die LEASY-Maintask beendet werden müssen.

Voraussetzungen

- Die Funktion wird im Haupt-MASTER aufgerufen
- Es wird mit automatischem Mitführen von Schattendateien gearbeitet
- Die betroffenen Dateien sind in der **FIL*-Anweisung des Dienstprogramms LEASY-CATALOG mit *AIM= (Y,A)* oder *AIM=(R,A)* vereinbart
- Die Dateien sind Stammdateien mit Zugriffsmethode ISAM oder PAM
- Original- und ggf. zugehörige SI-Dateien sind mit SHARED-UPDATE=YES eröffnet
- Die Funktion *REPO* macht einen impliziten Wechsel der AIM-Dateigeneration erforderlich. Kann dieser nicht durchgeführt werden, weil keine AIM-Dateigeneration frei ist, so wird die Funktion *REPO* abgebrochen.

Der LEASY-Administrator macht folgende Angaben:

- Auswahl der betroffenen Dateien
- Wartezeit, während der auf den Abschluss aller offenen Transaktionen gewartet werden soll, die die ausgewählten Dateien betreffen
- Reaktion, falls die Wartezeit verstreicht, ohne dass diese Transaktionen abgeschlossen werden konnten

LEASY wartet auf den Abschluss aller offenen Transaktionen, die die ausgewählten Dateien betreffen. Danach bringt der automatische RECONST alle zugehörigen Schattendateien auf den aktuellen Stand. Schließlich werden die betroffenen Originaldateien und ggf. die zugehörigen SI-Dateien durch die Schattendateien ersetzt.

Während der Ausführung der Funktion *REPO* werden alle neu eröffneten Transaktionen abgewiesen, dadurch kann es zu Behinderungen von LEASY-Anwendungen kommen. Implizite Wechsel der AIM-Dateigeneration wegen Erreichen der im MAINTASK-Parameter AIS angegebenen AIM-Dateigröße werden ebenfalls nicht ausgeführt.

Abhängig von der festgelegten Reaktion ist es möglich, dass nach Ablauf der Wartezeit noch offene Transaktionen vorhanden sind, die nicht zurückgesetzt werden. Damit kann die Funktion *REPO* nicht vollständig durchgeführt werden und es wurden nicht alle ausgewählten Originaldateien durch ihre Schattendateien ersetzt. Darauf wird mit der Meldung *LEA5510* hingewiesen.

Dieser Fall kann insbesondere bei Dateien eintreten, für die keine BIM-Sicherung vereinbart ist. Transaktionen, die solche Dateien bearbeiten, können grundsätzlich nicht zurückgesetzt werden.

5.3.6 Manuelle (explizite) Online-Sicherung

LEASY erlaubt es dem Anwender, einzelne Dateien explizit mit einer frei wählbaren Anwendung zu sichern. Damit keine inkonsistenten Sicherungen entstehen, dürfen die zu sichernden Dateien während dieses Sicherungsvorgangs nicht verändert werden. Um dies im laufenden LEASY-Betrieb zu gewährleisten, muss für die betroffenen Dateien eine Schreibsperre gesetzt werden.

Hierzu dient die Funktion *ROMS* im Dienstprogramm LEASY-MASTER. Sie ermöglicht es dem LEASY-Administrator, während des laufenden Betriebes einzelne oder alle Stammdateien (mit Zugriffsmethode ISAM oder PAM, einschließlich evtl. vorhandenener SI-Dateien) eines LEASY-Kataloges in den READ-ONLY-Modus zu versetzen.

Voraussetzungen

- Die Funktion wird im Haupt-MASTER aufgerufen
- Die betroffenen Dateien sind in der **CAT-* oder **FIL-*Anweisung des Dienstprogramms LEASY-CATALOG mit *ROM= Y* für das Setzen des READ-ONLY-Modus vorgemerkt
- Die Dateien sind Stammdateien mit Zugriffsmethode ISAM oder PAM
- Original- und ggf. zugehörige SI-Dateien sind mit SHARED-UPDATE=YES eröffnet

Der LEASY-Administrator macht folgende Angaben:

- Auswahl der betroffenen Dateien
- Wartezeit, während der auf den Abschluss aller offenen Transaktionen gewartet werden soll, die die ausgewählten Dateien betreffen
- Reaktion, falls die Wartezeit verstreicht, ohne dass diese Transaktionen abgeschlossen werden konnten

Das Dienstprogram LEASY-MASTER wartet auf den Abschluss aller offenen Transaktionen, die die ausgewählten Dateien betreffen. Danach versetzt es die ausgewählten Dateien in den READ-ONLY-Modus. Bei erfolgreichem Abschluss der Funktion *ROMS* wird eine entsprechende Meldung (*LEA5512*) ausgegeben. Zuvor werden ggf. weitere Meldungen ausgegeben, die darüber informieren, dass noch Transaktionen offen waren und dass diese zurückgesetzt wurden.

Nach erfolgreichem Abschluss der Funktion *ROMS* kann der LEASY-Administrator die gewünschte Sicherung durchführen. Danach muss er die Schreibsperre mit der Funktion *ROMR* wieder aufheben, damit wieder schreibende Operationen auf die betroffenen Dateien ausgeführt werden können. Er darf die Schreibsperre jedoch erst dann aufheben, wenn die Sicherung abgeschlossen ist, da andernfalls inkonsistente Sicherungsdateien entstehen können. Wird die Sicherung mit LEASY-SAVE durchgeführt, ist dies automatisch gewährleistet, da während dieser Sicherung die Funktion *ROMR* abgewiesen wird.

Abhängig von der festgelegten Reaktion ist es möglich, dass nach Ablauf der Wartezeit noch offene Transaktionen vorhanden sind, die nicht zurückgesetzt werden. Damit kann die Funktion *ROMS* nicht vollständig durchgeführt werden. Sie wird daher mit Meldung *LEA5511* abgebrochen. In diesem Fall ist eine Sicherung mit LEASY-SAVE **nicht** möglich und eine Sicherung mit anderen Mitteln kann nur für die Dateien durchgeführt werden, deren Transaktionen abgeschlossen oder zurückgesetzt wurden.

Dieser Fall kann insbesondere bei Dateien eintreten, für die keine BIM-Sicherung vereinbart ist. Transaktionen, die solche Dateien bearbeiten, können grundsätzlich nicht zurückgesetzt werden.

Während der READ-ONLY-Modus gesetzt ist, werden für die betroffenen Dateien nur Leseoperationen zugelassen. Bestandsverändernde Transaktionen und LEASY-Anweisungen für Transaktionen, die bereits nach Ablauf der Wartezeit zurückgerollt bzw. zurückgesetzt wurden, werden jedoch zurückgewiesen.

Der READ-ONLY-Modus kann nicht für Dateien gesetzt werden, die bereits durch ein fremdes Anwenderprogramm geöffnet sind, wenn der erstmalige Zugriff durch eine LEASY-Anwendung erfolgt.

Der aktuelle Zustand der Funktion *ROMS* wird in der Jobvariablen *LEACMST hinterlegt (siehe Seite 114).

LEASY-SAVE

Damit eine Sicherung der Dateien im READ-ONLY-Modus mit LEASY-SAVE durchgeführt werden kann, müssen folgende Voraussetzungen erfüllt sein:

- Der aktuelle LEASY-Katalog enthält ausschließlich Stammdateien mit Zugriffsmethode ISAM oder PAM.
- Alle Stammdateien (inklusive der SI-Dateien) des aktuellen LEASY-Katalogs müssen für den READ-ONLY-Modus vorgemerkt sein (Angabe *ROM=Y* der LEASY-CATALOG-Anweisung **CAT* oder **FIL*).
- Beim Einrichten des Katalogs wurden **keine** Angaben zu Schattendateien gemacht (*CPC/CPS*).
- Bei der Ausführung der Funktion *ROMS* im Dienstprogramm LEASY-MASTER ist in der Bildschirmmaske 47 (Datei für ROMS hinzufügen) *ALL angegeben.
- Die Funktion *ROMS* ist erfolgreich abgeschlossen; es gibt keine offenen Transaktionen.
- LEASY-Maintask ist gestartet

6 Betriebsarten

Prinzipiell ist das Softwareprodukt LEASY in folgenden Betriebsarten einsetzbar:

- Teilnehmerbetrieb

  Diese Betriebsart ist im folgenden Abschnitt beschrieben. Einen kurzen Überblick können Sie sich anhand von Bild 7 auf Seite 86 und den zugehörigen Erläuterungen verschaffen.

- Teilhaberbetrieb mit openUTM

  Diese Betriebsart ist im Abschnitt „Teilhaberbetrieb (openUTM)“ auf Seite 94ff. beschrieben. Einen kurzen Überblick können Sie sich anhand von Bild 9 auf Seite 96 und den zugehörigen Erläuterungen verschaffen.

- Teilhaberbetrieb mit DCAM

  Diese Betriebsart ist im Abschnitt „Teilhaberbetrieb (DCAM)“ auf Seite 100ff. beschrieben.

Einen Überblick über die Unterschiede können Sie sich anhand von Abschnitt „Unterschiede zwischen openUTM- und DCAM-Teilhaberbetrieb“ auf Seite 105 verschaffen.

6.1 Teilnehmerbetrieb (TIAM/Batch)

Zwischen Anwendertasks und Datenstationen / Stapelprogrammen besteht eine feste 1:1 Zuordnung. In jeder Anwendertask kann höchstens eine LEASY-Transaktion offen sein. An jedes Anwenderprogramm ist das Verbindungsmodul *LEASY* zu binden, das

- das Modul *LEACON* dynamisch bindet.
- die Operationen mit ihren Operanden an *LEACON* weiterreicht.
- STXIT-Routinen für den Fehlerfall enthält.

Alle LEASY-Operationen werden im von *LEACON* nachgeladenen Modul *LEACONX* durchgeführt.

Das folgende Bild gibt einen Systemüberblick über LEASY im Teilnehmerbetrieb.

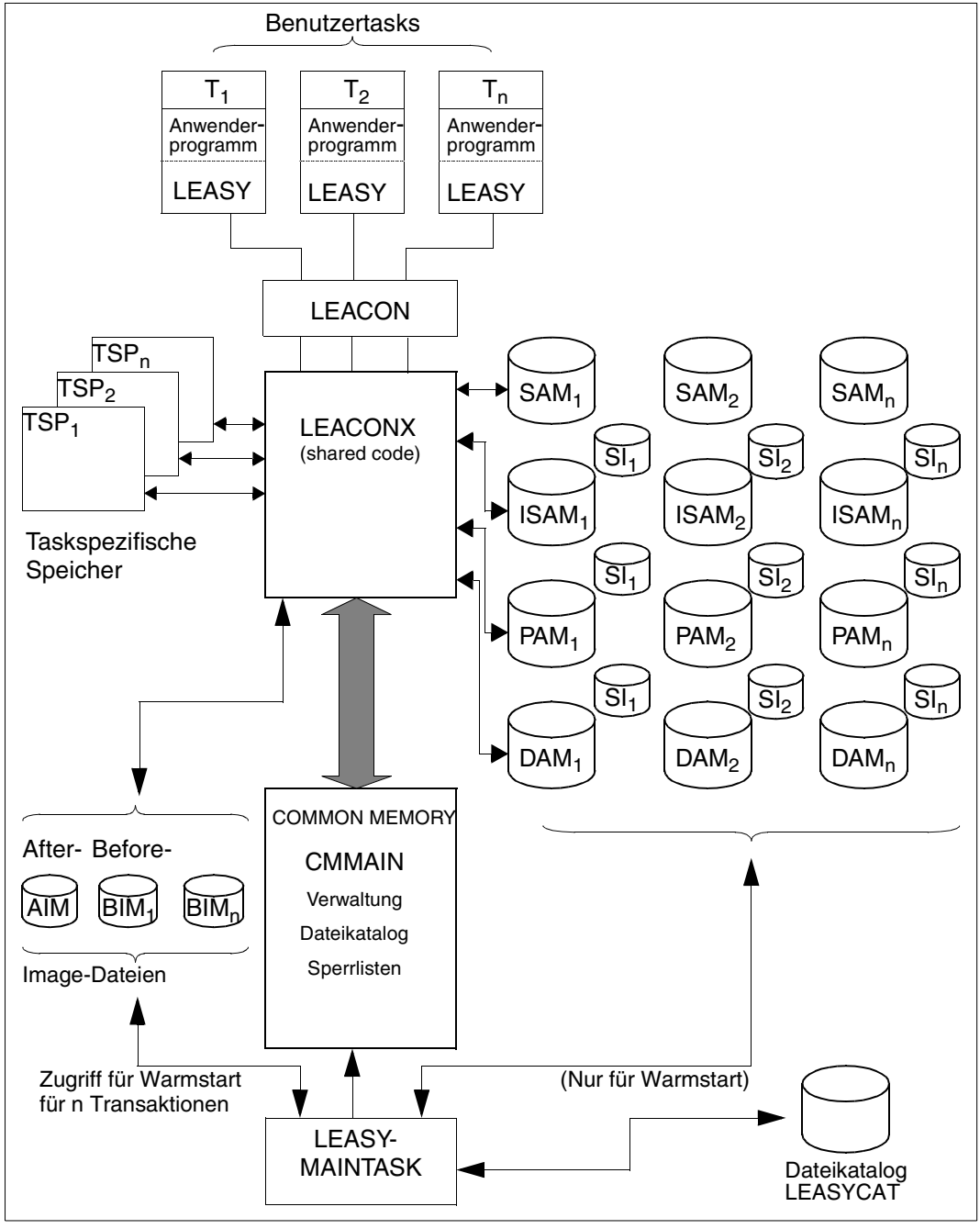


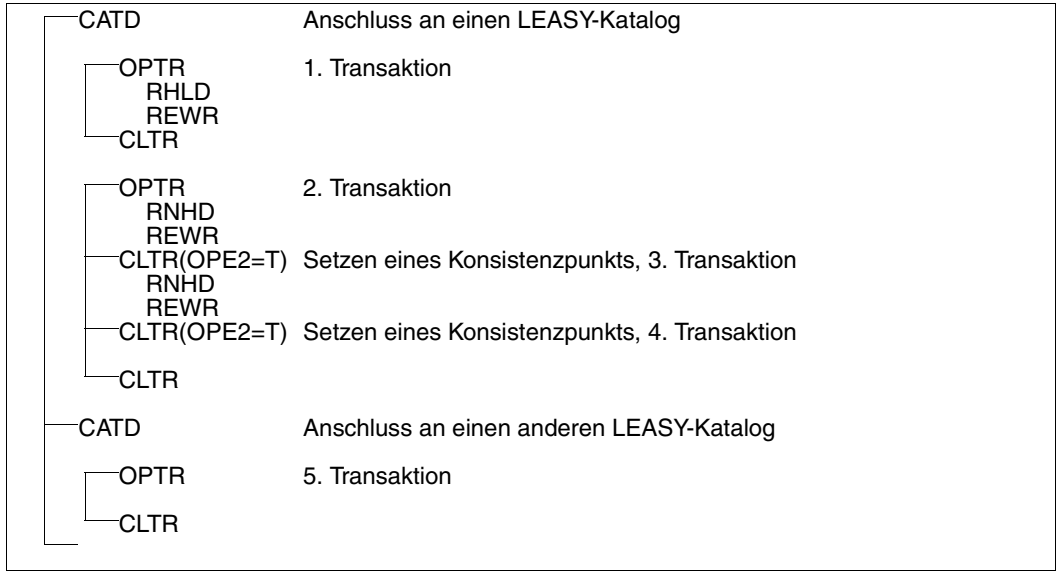
Bild 7: Systemübersicht für Teilnehmerbetrieb

6.1.1 Methoden zum Öffnen und Schließen von Dateien

Dateien können selektiv über die Operation *OPFL* geöffnet und über die Operation *CLFL* geschlossen werden. Es ist aber auch möglich, alle Dateien mit der Operation *OPTR* bei Programmbeginn zu öffnen und mit *CLTR* bei Programmende zu schließen.

Teilnehmerbetrieb ohne die Operationen OPFL und CLFL

Eine typische Operationsfolge in dieser Betriebsart hat z.B. folgenden Aufbau:



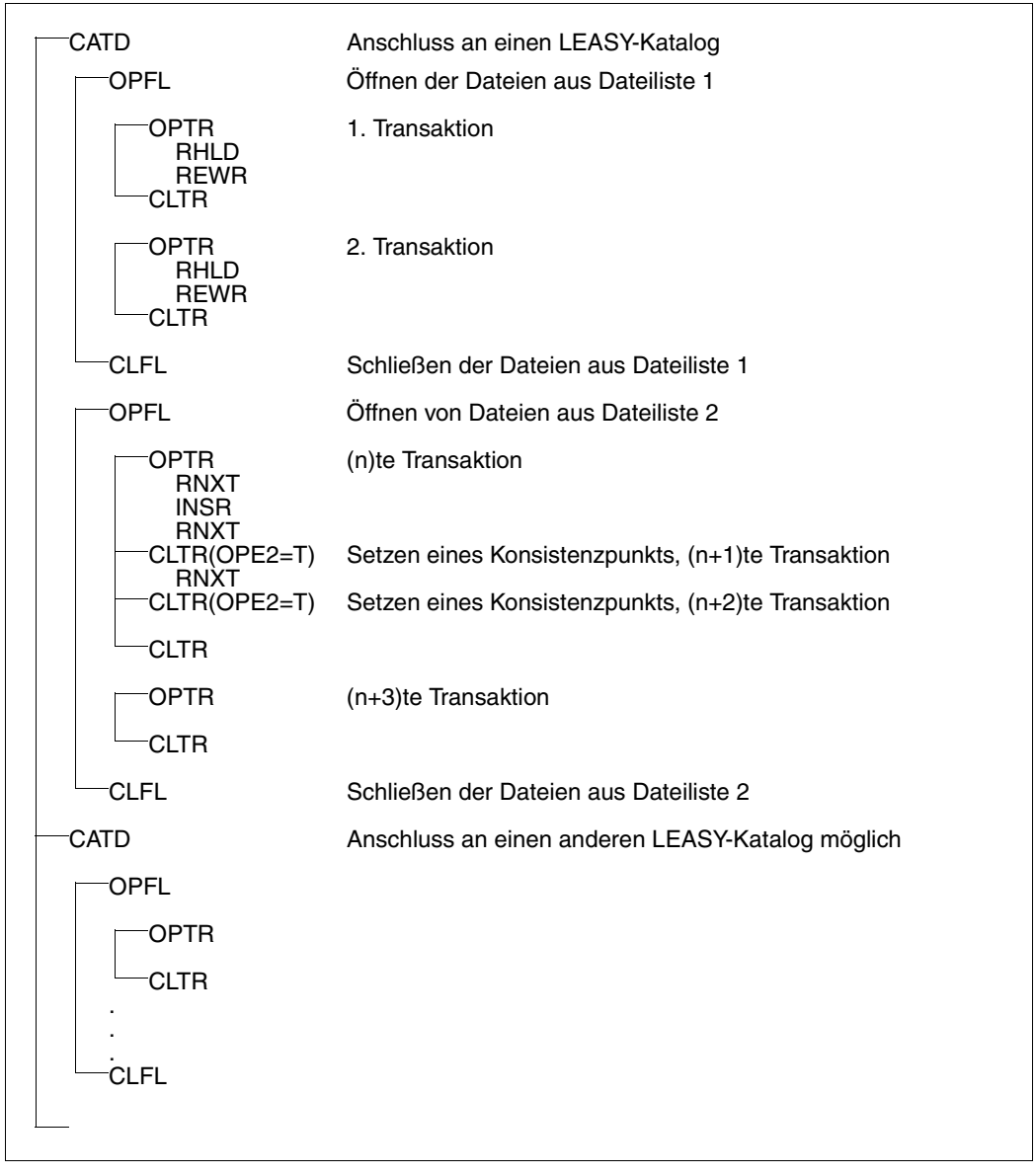
Alle an einer Transaktion beteiligten Dateien werden bei der Operation *OPTR* physikalisch eröffnet (DVS OPEN-Makro) und bei *CLTR* geschlossen. Eine Ausnahme bildet die Operation *CLTR* mit Zusatzfunktion *OPE2=T*, bei der nur ein Konsistenzpunkt gesetzt wird, d.h. dass die BIM-Datei als „leer“ definiert wird, aber der Dateizustand inklusive Dateipositionen erhalten bleibt.

Diese Betriebsart bietet sich für Stapelprogramme an, die zu Programmbeginn einmal durch *OPTR* alle Dateien eröffnen, ansonsten nur Konsistenzpunkte schreiben (*CLTR*,*OPE2=T*) und bei Programmende alle Dateien schließen (*CLTR*).

Es ist jedoch nicht günstig, Dialogtransaktionen jeweils durch die Operationen *OPTR* und *CLTR* einzuklammern, da das ständige physikalische Öffnen und Schließen der Dateien negative Auswirkungen auf die Laufzeit hat.

Teilnehmerbetrieb mit den Operationen OPFL und CLFL

Eine typische Operationsfolge im Teilnehmerbetrieb mit den Operationen *OPFL/CLFL* hat beispielsweise folgenden Aufbau:



Die Dateien werden mit der Operation *OPFL* physikalisch eröffnet (DVS-OPEN-Makro).

Die Transaktionen werden jeweils durch die Operationen *OPTR* und *CLTR* begrenzt, bei denen die Dateien nur logisch eröffnet und geschlossen werden. Erst durch die Operation *CLFL* werden die Dateien wieder physikalisch geschlossen.

Diese Betriebsart bietet sich für komplexe Dialoganwendungen an, die im Teilnehmerbetrieb in den einzelnen Modulen die verschiedensten Dateien mit unterschiedlichem USAGE-Modus verwenden.

Es ist möglich, die Betriebsarten mit und ohne *OPFL/CLFL*-Operationen wechselweise hintereinander anzuwenden.

6.1.2 Dateizugriffe über I/O-Task

Die Dateizugriffe werden im Modul *LEACONX* durchgeführt.

Der Teilnehmerbetrieb unterscheidet zwischen

- Dateizugriffen in der Anwendertask
- Dateizugriffen in der I/O-Task

Dateizugriffe in der Anwendertask

Das Modul *LEACON* bindet das Modul *LEACONX* an das Anwenderprogramm. LEASY kann höchstens 255 Transaktionen gleichzeitig bedienen. Damit ist die Anzahl der anschließbaren Datensichtstationen und der Stapelprogramme mit 255 begrenzt.

Die Belegung von Adressraum für Dateipuffer in jeder Anwendertask kann zu hohen Paging-Raten führen.

Dateizugriffe in der I/O-Task

Das Modul *LEACONX* liegt nicht mehr in jedem Anwenderprogramm, sondern (unter dem Namen LEAICNX) in einer oder mehreren gesonderten Tasks (I/O-Tasks). LEASY-Aufrufe des Anwenderprogramms werden nicht mehr vom Verbindungsmodul *LEACON* über Unterprogrammaufrufe, sondern über Intertask-Kommunikation an *LEAICNX* weitergereicht, dort abgearbeitet und wieder zurück an den Anwender geschickt.

Bis zu 1800 Anwendertasks können mit LEASY kommunizieren.

Die Kommunikation zwischen Anwenderprogramm und I/O-Task wird von Modulen abgewickelt, die in der Bibliothek *SYSLNK.LEASY.062.IOH* ausgeliefert werden. Aus Gründen der Kompatibilität haben diese Module ebenfalls die Namen *LEASY* und *LEACON* wie bei der Konfiguration, in der *LEACON* dynamisch in jedes Anwenderprogramm eingebunden wird („linked-in Version“).

Will der Anwender mit dem I/O-Handler arbeiten, muss er das Modul *LEASY* aus der Bibliothek *SYSLNK.LEASY.062.IOH* fest einbinden. Das Modul *LEASY* lädt dann das Modul *LEACON* aus der Bibliothek *SYSLNK.LEASY.062.IOH* in den Klasse-6-Speicher.

Einschränkungen an der LEASY-Schnittstelle

Die I/O-Task ruft LEASY intern über die DCAM-Schnittstelle auf. Alle Einschränkungen für die DCAM-Schnittstelle gelten damit auch für den I/O-Handler. Diese sind:

- Lademodus bei DAM:
  Jeder neue Satz liegt in einem neuen Block.
- SAM-Dateien können nur gelesen werden.
- Fremddateien können nur gelesen werden.
- Temporärdateien sind nicht zugelassen.

Wartezeit auf gesperrte Sätze

Eine I/O-Task, die bei der Bearbeitung eines Benutzerauftrags auf das Freiwerden einer Satzsperre wartet, steht nicht zur Bearbeitung anderer Aufträge zur Verfügung. Dies ist besonders bei nur einer I/O-Task zu beachten. Sie wartet auf das Freiwerden einer Sperre, die nur von ihr selbst freigegeben werden kann. Während der Wartezeit werden keine Aufträge von anderen Benutzern bearbeitet, d.h. sämtliche Anwender müssen warten. Der Anwender sollte bei nur einer aktiven I/O-Task darauf achten, dass die maximale Wartezeit im *RE*-Bereich auf 0 gesetzt ist.

Gleichzeitiger Betrieb mit und ohne I/O-Handler

Anwenderprogramme dürfen mit und ohne I/O-Handler gleichzeitig ablaufen. Die Unterscheidung wird durch das Nachladen von *LEACON* getroffen.

LEASY-Operationen

Falls Sie erstmals mit LEASY arbeiten, sollten Sie sich vor diesem Abschnitt mit der LEASY-Schnittstelle vertraut machen. Die LEASY-Schnittstelle mit allen Bereichen, Feldern und Operationen ist im Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121ff. beschrieben.

Operationen CATD und TERM

Die Operation *CATD* schließt das Anwenderprogramm an den von der Maintask eingerichteten Common Memory an. Die Operation *CATD* wird nicht an die I/O-Task weitergereicht. Wird bei der Operation *CATD* als Kataloginformation *CAT* ein Leerzeichen übergeben, trennt sich das Anwenderprogramm vom Common Memory.
Die Operation *TERM* wird bei offener Transaktion in die Operation *CLTR* mit Rollback (*OPE1=R*) umgewandelt und an die I/O-Task übergeben. Ansonsten wirkt diese Operation wie eine Operation *CATD* mit Leerzeichen als Kataloginformation *CAT*.
Wird ein Zusatzname zum Katalognamen angegeben, muss dieser mit dem Zusatznamen in der *CAT*-Anweisung des Programms LEASY-IOTASK übereinstimmen.

Operationen OPFL und CLFL

Die Operationen *OPFL* und *CLFL* werden nicht ausgeführt. Es findet auch keine syntaktische Überprüfung statt. Als Returncode wird immer *000LL000* geliefert. Dadurch wird es möglich, Anwendungen mit zentralem *OPFL/CLFL* ohne Änderungen auf den I/O-Handler umzustellen. Dies gilt nicht, wenn wechselnde OPEN-Modi für die Verarbeitung von Bedeutung sind (besonders bei SAM-Dateien).

Die *OPFL*-Operation wird beim Starten der I/O-Task abgesetzt. Sie hat damit für alle Anwender Gültigkeit, die über die I/O-Task mit LEASY arbeiten. Die USAGE-Modi der Transaktionen müssen mit den OPEN-Modi beim Starten der I/O-Task verträglich sein.

Die Dateien werden erst beim Beenden der I/O-Task geschlossen. Ein Wechseln des OPEN-Modus ist daher während einer Session mit I/O-Task nicht möglich.

Transaktionsoperationen

Alle übrigen Operationen werden vom Modul *LEACON* der Bibliothek *SYSLNK.LEASY.062.IOH* auf die LEASY-DCAM-Schnittstelle abgebildet und an die I/O-Task zur Bearbeitung geschickt.
Bei allen Operationen, insbesondere bei *OPTR*, gilt die *OPFL*-Operation, die die I/O-Task aus den *OPF*-Anweisungen des Programms LEASY-IOTASK zusammengesetzt und beim Start abgesetzt hat. SAM-Dateien können z.B. pro Session mit I/O-Task nur in einer Richtung gelesen werden.

Satzlängenbehandlung in der I/O-Task

Der AR-Bereich wird von der Anwendungstask zur I/O-Task und zurück je nach Operation in unterschiedlicher Weise übertragen.

Operation	**Satzlänge Anwendung → I/O**	**Satzlänge I/O → Anwendung**
SETL, DLET, LOCK, UNLK	In der I/O-Task angeg. Länge	---
INSR	aktuelle Satzlänge	aktuelle Satzlänge
RDIR, RHLD	In der I/O-Task angeg. Länge	aktuelle Satzlänge
RNXT, RNHD, RPRI, RPHD	---	aktuelle Satzlänge
REWR, STOR	aktuelle Satzlänge	---

Tabelle 5: Satzlängenbehandlung in der I/O-Task

Bei festem Satzformat ist die aktuelle Satzlänge gleich dem Wert von *RECSIZE*. Bei variablem Satzformat wird diese Länge aus den ersten beiden Bytes des AR-Bereiches gelesen.

6.2 Teilhaberbetrieb (openUTM)

Im folgenden Abschnitt sind openUTM-LEASY-Anwendungen beschrieben. Für das Verständnis dieses Abschnitts werden openUTM-Kenntnisse vorausgesetzt.

Generierung und Aufbau

Soll eine openUTM-Anwendung mit LEASY integriert werden, so ist dies bei der Generierung des openUTM-Anschlussprogramms (KDCROOT) durch Verwendung des folgenden Makros zu berücksichtigen:

```
KDCDBL.
```

Ein gebundenes openUTM-LEASY-Anwenderprogramm hat dann folgenden Aufbau:

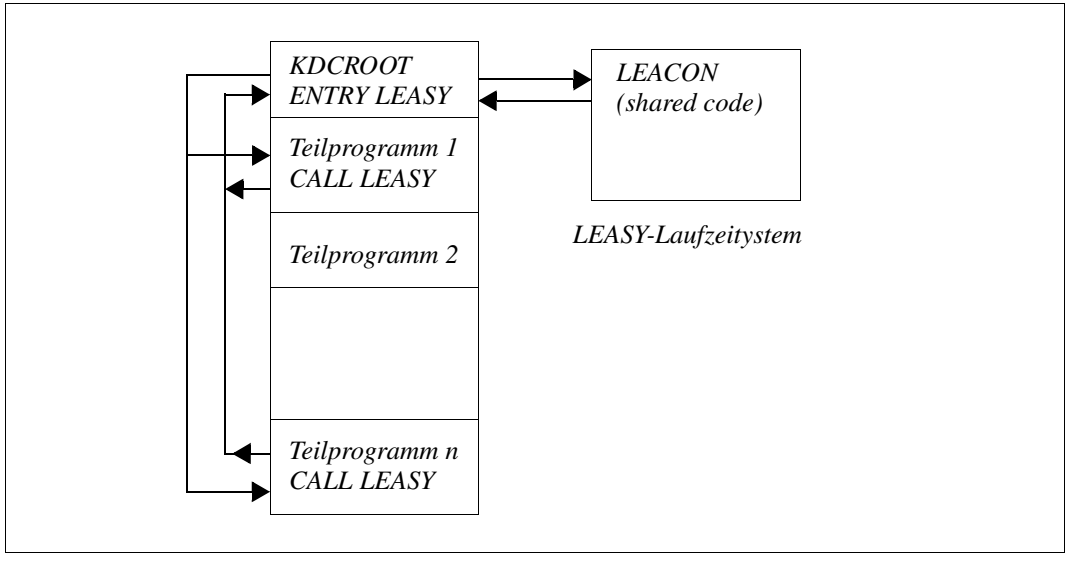


Bild 8: Struktur einer openUTM-LEASY-Anwendung

Jeder Aufruf *CALL "LEASY"* in den openUTM-Teilprogrammen wird über einen *ENTRY LEASY* in der KDCROOT geleitet. Die KDCROOT leitet den Aufruf an das LEASY-Laufzeitsystem weiter und übergibt zusätzlich bei jedem Aufruf an LEACON die Adressen eines transaktionsspezifischen und eines taskspezifischen Speichers.

Die KDCROOT enthält selbst alle Fehlerbehandlungen (z.B. STXIT-Routine) und führt die notwendigen Aktionen im Fehlerfall (*CLTR*, *OPE1=R* und *PEND ER*) selbst durch.

Zur Unterstützung von Mehrschritt-Transaktionen erfolgen von der KDCROOT auch Aufrufe an LEACON, die nicht durch *CALL "LEASY"* initiiert werden - z.B. bei *PEND KP*.

openUTM und LEASY besitzen ein gemeinsames Transaktionskonzept. Zur Realisierung eines gemeinsamen „Sicherungspunktes“ bei Transaktionsende wird von der KDCROOT folgendermaßen verfahren:

- Die LEASY-Operation *CLTR* wird an *LEACON* weitergeleitet, von *LEACON* jedoch nicht durchgeführt, sondern nur vermerkt (*CLTR*-Operationen sind wahlfrei).
- Erst innerhalb der folgenden *PEND*-Behandlung wird über einen speziellen internen Aufruf an *LEACON* die LEASY-Transaktion beendet.

Task- und Speicherstruktur

Im Vergleich mit dem Teilnehmerbetrieb ergeben sich bei openUTM-LEASY-Anwendungen folgende Unterschiede:

- Das LINK-Modul *LEASY* entfällt; statt dessen bindet die KDCROOT selbst das Laufzeitsystem *LEACON*.
- Um Mehrschritt-Transaktionen zu ermöglichen (Taskwechsel zwischen den einzelnen Dialogschritten), wird von *LEACONX* im Common Memory CMMAIN bei Taskwechsel (openUTM-Aufruf *PEND KP*) der aktuelle BIM-Puffer zwischengespeichert. Am Beginn des nächsten Dialogschritts werden diese Daten aus dem Common Memory in den Bereich der aktuellen Task übertragen. Der Taskwechsel kann auch in Einschritt-Transaktionen bei den Operationen *PEND/PA* und *PEND/PR* entstehen, falls der Anwender das TAC-Klassen-Konzept verwendet.

Das folgende Bild gibt einen schematischen Überblick über die Anwender- und LEASY-Module und deren Speicherbelegung.

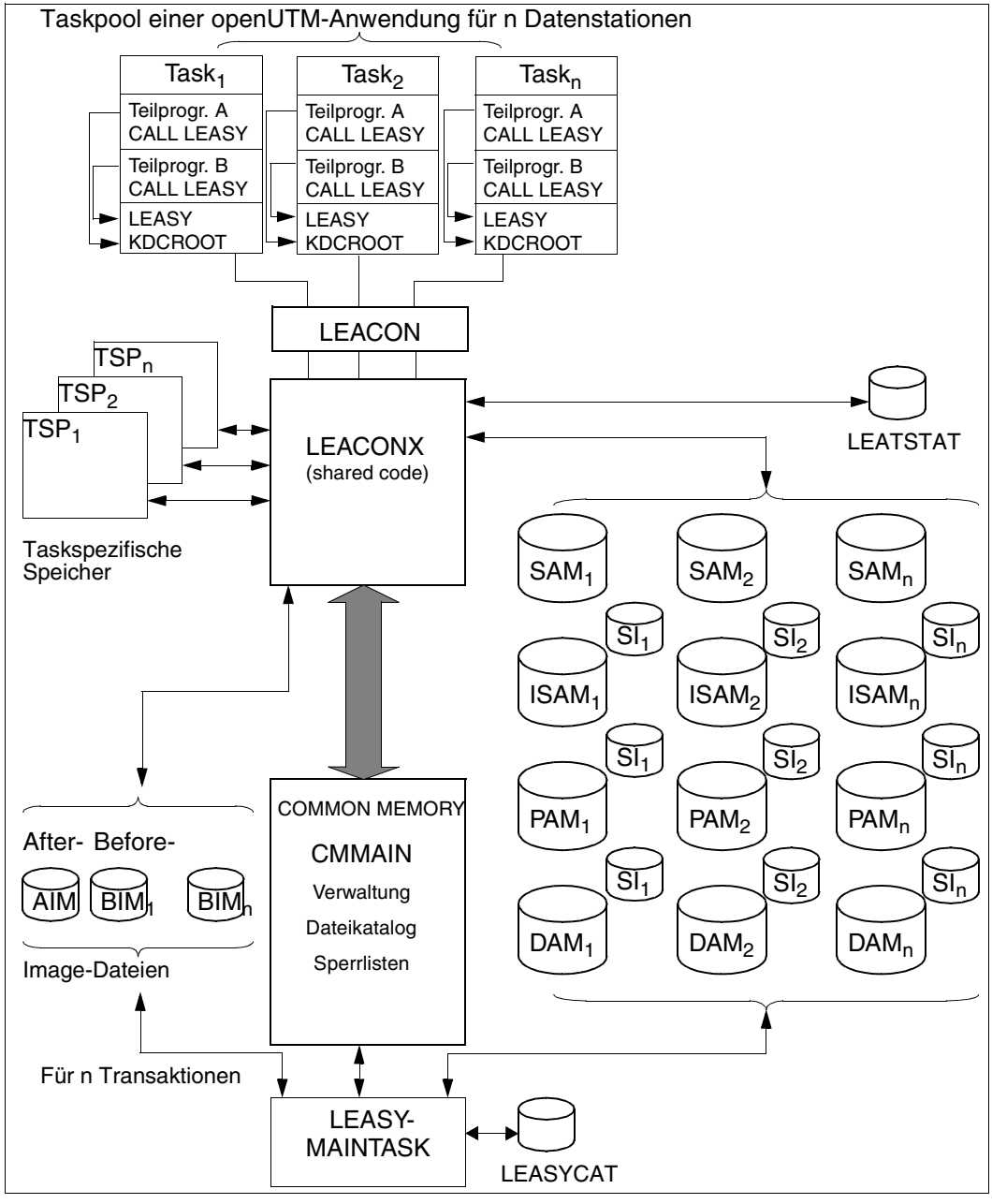


Bild 9: Systemübersicht einer LEASY-openUTM-Anwendung

openUTM-LEASY-Anwendung starten

Das openUTM-Anwenderprogramm wird mit dem Kommando *START-EXECUTABLE-PROGRAM* aufgerufen.

LEASY kennt für openUTM derzeit 3 Startparameter:

- Angabe des LEASY-Katalogs

  ```
  .LEASY CATD=[:catid:][$userid.]dateikatalog[.zusatz]
  ```

  Dieser Startparameter führt während der STARTUP-Phase der KDCROOT zum Anschluss an den Common Memory CMMAIN.

- Angabe der zu eröffnenden Dateien

  ```
  .LEASY OPFL=((datei1,mod1),...)
  ```

  Die Datei bzw. die Dateien werden im Format DB4 der Dateizuweisung (siehe Seite 138ff.) an das openUTM-Anwenderprogramm übergeben.

  Dieser Startparameter kann mehrmals erscheinen. Jede Datei darf aber in der gesamten Liste der Startparameter nur einmal vorkommen. Alle Dateiangaben werden in einer gemeinsamen Tabelle vermerkt und während der STARTUP-Phase einer impliziten *OPFL*-Anweisung zugeführt, so dass zum Zeitpunkt des ersten openUTM-Vorgangs in allen Tasks alle Dateien in gleicher Weise physikalisch eröffnet sind.

- Angabe, ob Parameterübergabe nach ILCS-Konventionen erfolgen soll

  ```
  .LEASY ILCS
  ```

  Weitere Informationen zur Parameterübergabe gemäß ILCS-Konventionen finden Sie auch in den Abschnitten „LEASY binden“ auf Seite 121 und „LEASY aufrufen“ auf Seite 123.

LEASY-Statusdatei in openUTM-Umgebung

In bestimmten Fällen benötigt openUTM vom angeschlossenen Dateizugriffssystem Informationen über den Zustand einzelner Transaktionen. Diese Transaktionszustände werden bei LEASY in einer eigenen Datei, die je LEASY-Katalog geführt wird, gespeichert. Es handelt sich dabei um eine ISAM-Datei mit dem Namen

```
dateikatalog.LEATSTAT
```

Sie wird beim Rückrollen von Transaktionen aus openUTM-Umgebung durch die Dienstprogramme LEASY-MAINTASK bzw. LEASY-RECONST beschrieben.

Standardmäßig wird die Statusdatei auf gemeinschaftlichem Datenträger errichtet. Soll sie auf einen privaten Datenträger geschrieben werden, so ist vor dem ersten Warmstart bzw. Rekonstruktionslauf, bei dem auf die Datei zugegriffen wird, folgendes Kommando abzusetzen:

```
/CREATE-FILE dateikatalog.LEATSTAT,
SUPPORT=*PRIVATE-DISK(VOL=vsn,DEV-TYPE=device)
```

bzw.

```
/CREATE-FILE dateikatalog.LEATSTAT,SUPPORT=*TAPE(VOL=vsn,DEV-TYPE=device)
```

Die Datei existiert nur, wenn sie vom Benutzer eingerichtet oder von einem LEASY-Dienstprogramm durch einen Rückrollvorgang automatisch erzeugt wurde. Sie hat aus Sicherheitsgründen ein intern vergebenes Schreibkennwort.

Einschränkungen gegenüber dem Teilnehmerbetrieb

- Der Wechsel des LEASY-Katalogs ist nicht möglich, da der *CATD*-Aufruf in der Taskinitialisierungsphase durch openUTM selbst durchgeführt wird. *CATD* in Anwenderprogrammen ist nicht erlaubt.
- Die Operation *OPFL* wird nur einmal pro Anwendung aufgerufen. Bei Verwendung von openUTM sind die entsprechenden Dateiangaben als Startparameter *OPFL* der Anwendung anzugeben. Diese führen in der Startup-Phase zu einem impliziten *OPFL*-Aufruf. Die Verwendung von *OPFL*- und *CLFL*-Aufrufen wird in Verbindung mit openUTM in den Benutzerprogrammen nicht erlaubt!
- Die Operation *CLTR* mit Zusatzfunktionen *OPE2=T* ist nicht möglich, da sie dem Transaktionskonzept von openUTM widersprechen würde.
- SAM-Dateien können nur gelesen werden, da das DVS keine Möglichkeit bietet, SAM-Dateien von mehreren Tasks (openUTM-Taskpool) gleichzeitig schreibend zu eröffnen (Ausweichmöglichkeit: ISAM-Datei mit USAGE-Modus = *LOAD*).

- Temporärdateien sind durch die TSN der Task und nicht der Transaktion bzw. der Datenstation zugeordnet. Daher sind sie für eine Verwendung in einer openUTM-Anwendung ungeeignet, da - bedingt durch den Taskpool einer Anwendung - unkontrollierbar verschiedene Tasks und damit verschiedene physikalische Dateien einer logischen Datei zugeordnet würden. Temporärdateien werden daher bei der Operation *OPFL* mit Fehlercode *UTMLLU13* abgelehnt.
- Fremddateien werden vom LEASY-Laufzeitsystem nicht im Common Memory CMMAIN verwaltet. Es werden auch keine Sperrlisten geführt. Eine gleichzeitige schreibende Verwendung von mehreren Benutzern ist daher nicht möglich; nur die OPEN-Modi ’1’ (SAM, ISAM, PAM) und ’5’ (SAM) sind erlaubt.
- Das Ausschalten der BIM-Sicherung ist nur für lesende Transaktionen erlaubt.
- I/O-Tasks können im Teilhaberbetrieb nicht eingesetzt werden.

Diagnoseinformation in der openUTM-DB-DIAGAREA

- openUTM dokumentiert eingetretene Ereignisse in taskspezifischen Trace-Bereichen, die zyklisch beschrieben werden. In der sogenannten DB-DIAGAREA werden Aufträge an das LEASY-System dokumentiert (siehe Handbuch „openUTM Meldungen, Test und Diagnose“, DB-DIAGAREA).
- Innerhalb eines Trace-Satzes der DB-DIAGAREA legt LEASY in einem 32 Byte langen Feld („Secondary DB Trace Information“) Daten über den einzelnen Auftrag ab. Diese Informationen dienen dem Kundenservice zur besseren Diagnose bei auftretenden Problemen.

6.3 Teilhaberbetrieb (DCAM)

Für spezielle Funktionen im **Teilhaberbetrieb**, die über das Funktionsspektrum von openUTM hinausgehen, stellt LEASY einen DCAM-Anschluss zur Verfügung.

Ablauf einer DCAM-Anwendung mit LEASY

Eine DCAM-Anwendung wird zweigeteilt:

- In einen Monitor, der die Steuerung und Weiterleitung von Nachrichten übernimmt.
- In Anwenderprogrammbausteine, die u.a. auf Daten zugreifen.

Der Datenzugriff kann mit Hilfe von LEASY auf Dateiebene transaktionsorientiert erfolgen.

Damit die Schnittstelle möglichst vielseitig verwendbar ist, ist sie so ausgelegt, dass innerhalb von Transaktionen jederzeit die DCAM-seitige Steuerung auf andere Transaktionen übergeben werden kann.

DCAM kann auf folgende Weisen verwendet werden:

- Ein DCAM-Programm bedient in mehreren verzahnt ablaufenden Prozessen mehrere Datenstationen mit verzahnt ablaufenden LEASY-Transaktionen (entsprechend openUTM-Eintaskbetrieb mit Mehrschrittransaktionen), wobei aber nach jeder beliebigen LEASY-Operation die gerade bearbeitete Transaktion unterbrochen wird und die Bearbeitung mit einer anderen - eventuell früher bereits unterbrochenen - Transaktion weitergeführt werden kann.

  In diesem Fall können Dateien bei ausschließlicher Verwendung in dieser DCAM-Anwendung auch ohne SHARED UPDATE eröffnet werden.

- Mehrere DCAM-Programme bedienen in mehreren verzahnt ablaufenden Prozessen mehrere Datenstationen mit verzahnt ablaufenden LEASY-Transaktionen (entsprechend openUTM-Mehrtaskbetrieb mit Mehrschrittransaktionen), wobei nicht nur innerhalb einer Task die Transaktionen unterbrochen und fortgeführt werden können, sondern die Weiterführung einer beliebigen Task übergeben werden kann.
  Dabei muss sichergestellt sein, dass jede Task, bevor sie Transaktionen bearbeiten kann, die LEASY-Operationen *CATD* und *OPFL* fehlerfrei durchführt. Diese Operationen müssen für alle Tasks einer DCAM-Anwendung **vollkommen identisch** sein, für *OPFL* auch die Dateireihenfolge. Dies kann z.B. durch Verwendung einer Parameterdatei in allen Tasks der Anwendung oder durch ein gemeinsam verwendetes Modul sichergestellt werden. Alle zu eröffnenden Dateien sind in einer *OPFL*-Operation anzugeben. Schreibend bearbeitete Dateien können nur mit den LEASY-Zugriffsmethoden ISAM, PAM oder DAM definiert sein und müssen mit SHARED UPDATE eröffnet werden.

Bei der Operation *CATD* muss LEASY der DCAM-Anwendungsname zur Verfügung gestellt werden.

Vor der ersten *OPTR*-Operation jeder Transaktion muss das Feld, das später die Transaktions-Identifikation aufnimmt, gelöscht sein. LEASY liefert in diesem Feld bei der Transaktionseröffnung die Transaktions-Identifikation zurück. Diese Identifikation muss bei jeder LEASY-Operation, die diese Transaktion betrifft, mitgeliefert werden.

Der DCAM-Monitorteil muss also intern eine Verwaltungstabelle führen, in der er die Datenstationsadresse und zugehörige Transaktions-Identifikation verwaltet und aktualisiert.

Eine *CLTR*-Operation löscht dieses Identifikationsfeld der LEASY-Schnittstelle. Der DCAM-Monitorteil muss in seiner Verwaltungstabelle ebenfalls die Transaktions-Identifikation für diesen Bildschirm löschen.

Für eine ordnungsgemäße Abmeldung einer DCAM-Task ist die folgende LEASY-Operationsfolge zu verwenden:

`CLTR [R]` (im Eintaskbetrieb für alle Transaktionen)

`CLFL`

`CATD` mit Leerzeichen im Feld für den LEASY-Katalognamen

Bei Bearbeitung von DAM-Dateien kann **jede** LEASY-Datenoperation und *CLTR* in einen Rückrollvorgang für die LEASY-Transaktion münden. Wird der Rückrollvorgang ordnungsgemäß beendet, wird ebenfalls die Transaktions-Identifikation aus dem Feld *IDE* des *RE*-Bereichs gelöscht.

Die Aktionsfunktionen des Dienstprogrammes LEASY-MASTER lösen z.T. Returncodes an der LEASY-Programmschnittstelle aus. Auch dadurch können bei beliebigen Datenoperationen Rückrollvorgänge für LEASY-Transaktionen ausgelöst werden.

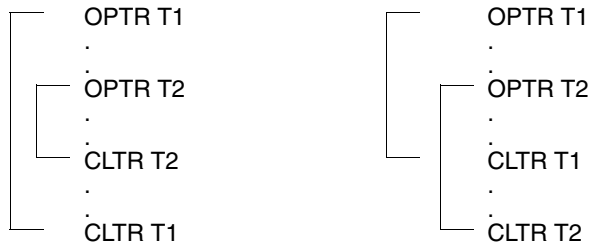
Verbindungsmodul LEADCAM

Um eine leichtere Handhabung zu erreichen, ist für die DCAM-Anwendungen statt des Moduls *LEASY* das Anschlussmodul *LEADCAM* vorgeschrieben. Dieses Modul wird in der Bibliothek *SYSLNK.LEASY.062.DCAM* zur Verfügung gestellt. Es enthält die Einsprungadresse LEASY, so dass der Anwender LEASY über einen Unterprogrammaufruf (*CALL*-Aufruf) aufrufen kann.

Das Anschlussmodul *LEADCAM* enthält STXIT-Routinen, die für Standardanwendungsfälle ausgelegt sind.

Der LEASY-DCAM-Anschluss bietet dem Anwender die Möglichkeit, Transaktionen zu schachteln, und zwar auch im Teilnehmerrbetrieb, wenn statt des Anschlussmoduls *LEASY* das Modul *LEADCAM* verwendet wird und die Schnittstelle wie oben beschrieben versorgt wird. Die Transaktionen dürfen ineinander oder gezahnt verschachtelt werden.

Beispiel



Die LEASY-Operationen nach dem zweiten *OPTR*-Aufruf gehören **nicht** beiden Transaktionen an, sondern nur derjenigen Transaktion, deren Identifikation sie jeweils enthalten. Es entsteht also eine zeitliche Verzahnung ansonsten unabhängiger Transaktionen mit entsprechenden Konsequenzen (z.B. auf die Transaktionszahl, Sperrlogik).

Diese Schachtelung ist nicht identisch mit der seit LEASY V3.0 bestehenden Möglichkeit, Dateipfade zu einer bestehenden Transaktion mittels weiterer *OPTR*-Operationen nach Transaktionsbeginn nachzureichen.

LEASY-Operationen

Falls Sie erstmals mit LEASY arbeiten, sollten Sie sich vor diesem Abschnitt mit der LEASY-Schnittstelle vertraut machen. Die LEASY-Schnittstelle mit allen Bereichen, Feldern und Operationen ist im Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121ff. beschrieben.

Operation CATD

Die Operation *CATD* ist obligatorisch. DCAM-Programme müssen zusätzlich zur sonstigen Parameterversorgung beim Anmelden an eine LEASY-Anwendung im Feld *IDE* des *RE*-Bereichs den DCAM-Anwendungsnamen übergeben. Der Anwendungsname darf nicht aus lauter Leerzeichen oder binären Nullen bestehen, alle anderen Bitkombinationen sind erlaubt. In den LEASY-Dienstprogrammen, die den Anwendungsnamen an der Datenstation oder auf Drucker ausgeben, können allerdings nur abdruckbare Zeichen dargestellt werden.
Nach erfolgreicher Durchführung wird der Feldinhalt von LEASY gelöscht.

Operationen OPFL und CLFL

Die Operation *OPFL* ist beim Start eines DCAM-Programms nach der Operation *CATD* obligatorisch, alle zu eröffnenden Dateien sind in **einer** Operation anzugeben.

Dadurch erreicht das Programm den ersten Synchronpunkt, der von allen Tasks der DCAM-Anwendung ebenfalls erreicht werden muss. Stehen eine oder mehrere Tasks auf diesem ersten Synchronpunkt (*CATD* und *OPFL*) und bearbeiten bereits Transaktionen, so können trotzdem weitere Tasks für diese Anwendung gestartet werden.

Im Eintaskbetrieb wird dieser erste Synchronpunkt durch beliebige *OPFL/CLFL*-Operationen nicht verlassen. Soll zu einer solchen DCAM-Task eine weitere Task zur selben Anwendung gestartet werden, so muss der aktuelle Zustand aller Dateien der ersten Task mit einer *OPFL*-Anweisung erreicht werden.

Zerstört eine Task im Mehrtaskbetrieb diese Tasksynchronisation durch Absetzen **einer** *OPFL*- oder *CLFL*-Operation, so können ab diesem Zeitpunkt keine weiteren Tasks mehr zu dieser Anwendung gestartet werden. Außerdem muss dieser weitere Synchronpunkt wieder von allen Tasks der Anwendung in gleicher Weise erreicht werden. Erst danach ist eine neuerliche Verletzung der Synchronität durch eine Task erlaubt.

Die Verletzung der Synchronität durch eine Task durch Absetzen mehrerer *OPFL*- bzw. *CLFL*-Operationen ist **nicht** gestattet.

Tritt bei der Operation *CLFL* ein Synchronisationsfehler auf (Returncode *DCALLU16*), so ist in dieser Task nur noch die Operation *TERM* erlaubt und diese nur in einer STXIT-Routine des Anwenders. Anschließend ist die Task zu beenden.

Ein Verlassen des ersten und aller folgenden Synchronpunkte durch eine Task ist nur erlaubt, falls in der **gesamten** Anwendung **eine** Transaktion eröffnet ist.

Operation OPTR

Bei der ersten *OPTR*-Operation jeder LEASY-Transaktion ist das Feld *IDE* im *RE*-Bereich gelöscht zu übergeben (binäre Nullen). LEASY liefert bei ordnungsgemäßer Durchführung im selben Feld die Transaktions-Identifikation zurück, die dann bei allen Operationen dieser Transaktion (weitere *OPTR*-Operationen und *CLTR*) zu übergeben ist.

Operation CLTR

Im Feld *IDE* ist die Transaktions-Identifikation zu übergeben. Bei ordnungsgemäßer Durchführung der Operation wird *IDE* gelöscht, falls nicht *OPE2=T* gesetzt wurde.

Operation CINF

Bei den Operationen auf Dateiebene und bei *CINF* ist die Transaktions-Identifikation im Feld *IDE* zu übergeben.

Operation TERM

Die Operation *TERM* ist auch bei DCAM-Programmen nur in einer STXIT-Routine des Anwenders gestattet. Nach einer *TERM*-Operation sind keine anderen LEASY-Operationen mehr erlaubt. Die Task ist zu beenden.

Kann das Anwenderprogramm für diese Operation eine gültige Transaktions-Identifikation im Feld *IDE* des *RE*-Bereichs zur Verfügung stellen, so wird diese Transaktion zurückgerollt. Kann das Anwenderprogramm keine gültige Transaktions-Identifkation angeben, so ist das Feld *IDE* zu löschen. In diesem Fall wird keine Transaktion zurückgesetzt.

War die Task die Einzige einer DCAM-Anwendung, so bleiben alle eventuell offenen Transaktionen (außer der einen, deren Transaktions-Identifikation angegeben worden war) eröffnet. Erst ein Warmstart des Dienstprogramms LEASY-MAINTASK setzt diese Transaktionen zurück.

Verhalten im Fehlerfall

Die LEASY-Operation *TERM* darf nur beim Abmelden der **letzten** noch offenen Transaktion verwendet werden, da die Verwendung für nicht-letzte Transaktionen weitere Rückrollvorgänge unterbindet (diese Operation meldet eine Task auch vom Common Memory CMMAIN ab).

Im Fehlerfall gilt:

- Wurden für die betreffenden DCAM-Anwendungen mehrere Tasks aktiviert, so wird nur eine eventuell in der Task aktive LEASY-Transaktion zurückgerollt.
- Wurde nur die Task mit dem fehlerhaften Programm für die Anwendung aktiviert, so werden alle an der DCAM-Anwendung beteiligten LEASY-Transaktionen zurückgerollt.
- Das Rückrollen von LEASY-Transaktionen erfolgt durch die STXIT-Routine von LEASY.

Weitere Informationen zu STXIT-Routinen finden Sie ab Seite 106.

6.4 Unterschiede zwischen openUTM- und DCAM-Teilhaberbetrieb

Es gibt kein gemeinsames Transaktionskonzept zwischen DCAM-Anwendungen und LEASY. Die DCAM-Programme (Monitorteil) sind für die Organisation im Programm allein verantwortlich. Von LEASY werden die Returncodes im *RE*-Bereich zur Verfügung gestellt, die zu Steuerungszwecken verwendet werden können. Es gibt keine LEASY-Statusdatei in DCAM-Umgebung. Schreibende Transaktionen ohne BIM-Sicherung sind zugelassen.

Die Operationen *CATD* und *OPFL* müssen verwendet werden, bevor die Task LEASY-Transaktionen bearbeiten kann. Bei Mehrtaskbetrieb muss sichergestellt sein, dass alle Tasks der Anwendung dieselben Dateien in gleicher Weise und gleicher Reihenfolge eröffnet haben (der Zwang zur gleichen Reihenfolge beim Eröffnen der Dateien für alle Tasks einer Anwendung gilt auch für den openUTM-Teilhaberbetrieb). Verstöße gegen die Forderung nach gleicher Dateireihenfolge werden mit dem Fehlercode *DCALLU16* quittiert.

Der Wechsel des LEASY-Katalogs ist bei Eintaskbetrieb ohne Einschränkung erlaubt. Bei Mehrtaskbetrieb muss beim Wechsel des LEASY-Katalogs innerhalb der DCAM-Anwendung eine Synchronisation aller Tasks erfolgen, so dass sichergestellt ist, dass in der gesamten Anwendung alle LEASY-Transaktionen und alle Dateien geschlossen sind.

Die Operationen *CLFL* und *OPFL* sind bei geschlossenen Transaktionen erlaubt. Das Schließen und Eröffnen von Dateien muss wie das Wechseln des LEASY-Katalogs schrittweise für jede LEASY-Operation synchronisiert werden.

Verstöße gegen die Synchronisation bei *OPFL*- und *CLFL*-Operationen werden mit dem Fehlercode *DCALLU16* quittiert. Nach einem Synchronisationsfehler bei *CLFL* wird die Task gegen jede weitere Verwendung gesperrt. Jede Operation außer *TERM* wird ebenfalls mit *DCALLU16* abgelehnt.

Die Operation *CLTR* mit *OPE2=T* ist erlaubt.

DCAM-Programme müssen das Anschlussmodul *LEADCAM* aus der Bibliothek *SYSLNK.LEASY.062.DCAM* eingebunden haben.

6.5 Fehlerbehandlung durch die STXIT-Routine von LEASY

Das Verbindungsmodul *LEASY* enthält zur Behandlung von Fehlern eine STXIT-Routine. Sie wird beim ersten Aufruf von LEASY aktiviert. Es werden folgende Ereignisklassen behandelt:

- ABEND abnormale Programmbeendigung
- ERROR nicht behebbarer Programmfehler
- PROCHK Programmüberprüfung
- RUNOUT Programmlaufzeitende
- TERM normale Programmbeendigung (*TERM/TERM DUMP=Y*)

Zusätzlich wird in den STXIT-Routinen nach Abzug eines Dumps und evtl. Rückrollen einer offenen LEASY-Transaktion durch die Operation *TERM* im Dialogbetrieb der Programmlauf unterbrochen (*BKPT*-Makro), im Prozedur- und Stapelbetrieb geschieht dies nicht. Diese Maßnahme führt zu einer besseren Testfähigkeit von Teilnehmerprogrammen.

In der STXIT-Routine werden folgende Aktionen durchgeführt:

- Meldung nach SYSOUT mit Angabe von Unterbrechungsgewicht und Befehlszählerstand (zum Fehlerzeitpunkt) ausgeben.
- PDUMP des gesamten Programms (außer bei Ereignisklasse TERM).
- *CLTR* mit Rücksetzen aufrufen. Dadurch wird vermieden, dass Sperreinträge bezüglich Dateien oder Sätzen im Common Memory bestehen bleiben.
- Den LEASY-Returncode der Operation *CLTR* (mit Zusatzfunktion *OPE1=R*) nach SYSOUT ausgeben.
- PDUMP, falls Rollback (Aufruf *CLTR OPE1=R*) nicht fehlerfrei bearbeitet wurde.
- Zur Unterstützung der Fehlerdiagnose werden die Registerzustände zum Zeitpunkt des Programmfehlers eingestellt und im Dialogbetrieb der Programmlauf unterbrochen (*BKPT*-Makro).
- Wurde vor dem ersten LEASY-Aufruf eine STXIT-Routine für das aufgetretene Ereignis angemeldet, wird diese aktiviert, andernfalls beendet sich das Programm mit

  ```
  TERM MODE=ABNORMAL,UNIT=STEP
  ```

Die oben beschriebenen Aktionen werden bei der Ereignisklasse TERM nur dann durchgeführt, wenn eine offene Transaktion vorgefunden wurde. Ansonsten beendet sich das Programm ohne zusätzliche Aktion.

7 LEASY in Mehrrechnersystemumgebung

Das System LEASY kann auch im MRS-Mehrrechnerverbund eingesetzt werden.

Befinden sich ein anzusprechender LEASY-Katalog und dessen LEASY-Systemdateien auf einem fremden Rechner, so ist mit den LEASY-Dienstprogrammen (außer bei LEASY-MASTER) und an der Benutzerschnittstelle die Katalogkennung (*catid*) dieses Rechners als Bestandteil des DVS-Dateinamens anzugeben. Dasselbe gilt auch bei der Vereinbarung von expliziten DVS-Dateinamen für Anwenderdateien oder den Zugriff auf sie.

Format dieser Dateinamen:

```
:catid:$userid.datei
```

Ein LEASY-Katalog wird durch

```
:catid:$userid.dateikatalog
```

eindeutig identifiziert, d.h. Kataloge in verschiedenen Rechnern bzw. in verschiedenen Benutzerkennungen desselben Rechners und auch innerhalb derselben Benutzerkennung sind voneinander völlig unabhängig.

Wird mit dem Namen einer Datei eine Katalogkennung vergeben und das MRS ist nicht verfügbar, so weist das DVS den Aufruf zurück.

7.1 LEASY-Systemdateien im Mehrrechnersystem

Derzeit gibt es folgende LEASY-Systemdateien, deren Existenz zum Teil von den gewählten LEASY-Funktionen abhängt:

- `dateikatalog.LEASYCAT`
- `dateikatalog.LEASYAIM`
- `dateikatalog.BIM#.nnn`
- `dateikatalog.LEATSTAT`

Diese Dateien werden teils vom Benutzer (z.B. *LEASYAIM*), teils von den LEASY-Dienstprogrammen eingerichtet. Alle LEASY-Systemdateien, die zu einem LEASY-Katalog gehören, müssen sich auf einem Rechner befinden.

7.2 Mehrrechnerbenutzbare private Platte

Anwenderdateien und LEASY-Systemdateien können prinzipiell auch auf mehrrechnerbenutzbaren privaten Platten (Shareable Private Disc) gespeichert werden.

Eine DVS-Datei darf aber nicht Mitglied in mehreren LEASY-Katalogen sein!

Es kann auch zu Komplikationen führen, wenn zu einer Katalogdatei abwechselnd auf verschiedenen Rechnern LEASY-Sessions ablaufen.

7.3 Fern-Datei-Zugriff und Lastverteilung in einem MRS-Netz

Der Fern-Datei-Zugriff (Remote File Access (RFA)) gibt dem Benutzer die Möglichkeit der verteilten Anwenderdateihaltung, d.h. in einem Mehrrechnersystem können Anwenderdateien, die logisch zu einem LEASY-Katalog gehören, auf verschiedenen Rechnern eingerichtet sein.

Beim Einrichten solcher Dateien ist im Dienstprogramm LEASY-CATALOG mit der Anweisung **FIL* im *NAM*-Operanden eine *catid* anzugeben (ist nur für LEASY-Stammdateien möglich) und der *CID*-Operand in der **CAT*-Anweisung muss auf *CID=Y* (Standardwert) gesetzt sein.

Sekundärindex-Dateien liegen immer auf dem Rechner, auf dem die zugehörige Primärdatei katalogisiert ist.

Die Dienstprogramme LEASY-LOADSI, LEASY-MAINTASK und LEASY-RECONST können auch von einem Rechner aus gestartet werden, auf dem die Katalogdatei nicht eingerichtet ist. Dann ist bei der Anweisung **CAT* (bzw. bei der Katalogspezifikation des Dienstprogramms LEASY-LOADSI) eine *catid* anzugeben.

Auf dem Rechner, auf dem das Dienstprogramm LEASY-MAINTASK gestartet wird, wird dann auch der Common Memory CMMAIN eingerichtet. Die Sperrverwaltung erfolgt also über den CMMAIN auf dem Rechner, auf dem die Maintask gestartet wird, und nicht unbedingt auf demjenigen, auf dem der LEASY-Katalog eingerichtet ist.

Da die Dienstprogramme LEASY-MASTER und LEASY-RECONST auf den CMMAIN zugreifen, müssen diese Dienstprogramme ebenfalls auf dem Rechner laufen, auf dem LEASY-MAINTASK gestartet wurde.

Wollen Anwenderprogramme auf den Common Memory CMMAIN zugreifen, so sind folgende Maßnahmen zu treffen:

- Die Anwenderprogramme sind auf dem Rechner zu starten, auf dem der Common Memory CMMAIN eingerichtet ist
- Falls der Common Memory CMMAIN und der LEASY-Katalog auf verschiedenen Rechnern sind, ist bei der Angabe des LEASY-Katalogs in folgenden Fällen eine Katalogkennung :*catid*: zu übergeben.
  - im 3. Operanden des LEASY-Aufrufs bzw.
  - beim *CATD*-Startparameter im openUTM-Betrieb

Diese Möglichkeiten können zur Lastverteilung in einem MRS-Netz herangezogen werden.

Das Dienstprogramm LEASY-RECONST erlaubt, dass es auch auf einem Rechner gestartet wird, auf dem die Katalogdatei nicht eingerichtet ist. Dann ist bei der Anweisung **CAT* eine *:catid:* anzugeben. Die Form der internen Ablage der Dateinamen garantiert, dass auch in einem solchen Fall die Dateien richtig rekonstruiert werden.

7.4 Dateikonsistenz in einem MRS-Netz

Der Datei-Transfer ist nicht erlaubt, da die Dateinamen, die in den AIM- und BIM-Dateien eingetragen werden, die Katalogkennung *catid* enthalten. Diese Katalogkennung würde sich beim Übertragen der Datei auf einen anderen Rechner verändern, wodurch der Zugriff über die AIM- und BIM-Datei nicht mehr möglich ist.

Es muss Folgendes sichergestellt sein:

- alle Dateien, die bei einem Rekonstruktionslauf benötigt werden, müssen auf Rechnern liegen, die auch verfügbar sind, d.h. dass eine MRS-Umgebung vorliegt.
- vor einem Warmstart von LEASY-MAINTASK mit offenen Transaktionen müssen alle Dateien auf dem Rechner liegen, auf dem sie in der letzten LEASY-Session vorgefunden wurden.
- vor einem Rekonstruktionslauf mit LEASY-RECONST müssen alle Dateien auf dem Rechner liegen, auf dem sie bei den schreibenden Veränderungen während der Lebensdauer der AIM-Datei durch LEASY vorgefunden wurden.

Bestehende LEASY-Anwendungen, die von den Möglichkeiten der Lastverteilung in einem MRS-Netz Gebrauch machen, können jederzeit auch auf Einrechner-Betrieb umgestellt werden. Dabei sind folgende Maßnahmen zu ergreifen:

- Alle LEASY-System- und Benutzerdateien müssen in den DVS-Katalog gebracht werden, dessen Katalogkennung als Standardwert für das DVS des betreffenden Rechners gilt.
- Das Dienstprogramm LEASY-CATALOG wird gestartet und der *CID*-Operand der Anweisung **CAT* wird auf *CID=N* gesetzt. Alternativ kann mittels der Operanden *OLDL* und *NEWL* die Katalogkennung aus den Pfadnamen entfernt werden.

8 Besonderheiten beim Einsatz von LEASY

Dieses Kapitel enthält folgende Themenbereiche:

Im unmittelbar folgenden Abschnitt ist beschrieben, wie Sie bei der Planung eines konkreten LEASY-Einsatzes vorgehen, um optimale Performance zu erreichen.

Im Abschnitt „Jobvariablen einsetzen“ auf Seite 113ff. ist beschrieben, wie Ihnen Jobvariable den Einsatz von LEASY erleichtern können.

Im Abschnitt „Adressierungsmodus“ auf Seite 119 sind Besonderheiten bei der Adressierung unter Batch/TIAM und DCAM behandelt.

8.1 Planung eines LEASY-Einsatzes

Bei der Planung eines LEASY-Einsatzes sollten folgende Punkte beachtet werden:

Parallele Dialog-/Stapelverarbeitung

Prinzipiell unterscheidet LEASY an der Programmschnittstelle nicht, ob es sich um ein Dialog- oder Stapel-Anwenderprogramm handelt. Die Programmierung der Schnittstelle ist daher für alle Betriebsarten gleich, Parallelablauf aller Betriebsarten ist möglich.

Bei der Diskussion, ob parallel zu einer oder mehreren Dialoganwendungen auch Stapelverarbeitung durchgeführt werden soll, sind jedoch mehrere Aspekte zu beachten:

– Dialoganwendungen sind im Allgemeinen zeitkritisch, daher sollte jede unnötige zusätzliche Belastung des Rechners vermieden werden. Meist ist dies durch organisatorische Maßnahmen des Rechenzentrumsbetriebes erreichbar. Nur wenn dies nicht möglich ist, sollte parallele Stapelverarbeitung in Betracht gezogen werden.

– Zu Dialoganwendungen parallel ablaufende Stapelprogramme dürfen nur kleine Transaktionen enthalten, um die Behinderung der Dialogprogramme durch Sperren möglichst niedrig zu halten. Daraus folgt, dass auch Stapelprogramme mehrere Transaktionen enthalten werden. Falls ein Stapelprogramm - z.B. durch einen Programmfehler - abgebrochen wird, wird von LEASY nur die jeweils offene Transaktion zurückgerollt. Der Datenzustand ist dann zwar wieder konsistent, jedoch irgendwo mitten in der Stapelverarbeitung. Stapelprogramme müssen daher wiederanlauffähig programmiert oder von den Daten her wiederholbar sein.

Performance-Steigerungen bei OPFL und OPTR

Bei einer großen Anzahl von Dateien wird empfohlen, die Namen dieser Dateien in lexikalisch aufsteigender Reihenfolge einzugeben. Dadurch wird der CPU-Aufwand bei der Anlage und Überprüfung von Dateielementen geringer gehalten.

LEASY-Laufzeitsystem

Eine Möglichkeit, über die CALL-Schnittstelle Verbesserungen der Programmlaufzeit zu erreichen, bietet die Operation *OPTR*. Durch Versorgung des Feldes *OPE-LOG* mit *N* im *RE*-Bereich wird für die Transaktion die BIM-Sicherung unterdrückt.

Durch Programmabbrüche entstehende Dateninkonsistenzen können dann wegen fehlender BIM-Dateien mit einem LEASY-Warmstart nicht mehr behoben werden.

Zu vermeiden ist:

– das ständige Öffnen und Schließen von Anwenderdateien durch Weglassen der Operation *OPFL*.

– unnötiges *SHARED-UPDATE*-Eröffnen von Dateien.

– unnötige Angabe von nicht verwendeten Dateien in den Operationen *OPFL* und *OPTR*.

– falsche Sperrebene (z.B. Dateisperre, obwohl eine Sperre auf Satzebene reichen würde).

Entscheidend für einen schnellen Programmablauf ist auch die Operationsfolge innerhalb von Transaktionen. Dabei sind folgende Operationskombinationen zu vermeiden:

– unnötiges Sperren von Datensätzen.

– wiederholtes Lesen des gleichen Datensatzes in einer Transaktion.

– unnötiges Neupositionieren (explizit durch *SETL*- und *RDIR/RHLD*-Operationen, implizit auch bei *RNXT/RNHD/RPRI/RPHD*-Operationen durch Wechseln der Dateikennung).

Zur Verbesserung der Performance beim Schreiben einer ISAM-Datei mit den USAGE-Modi *LOAD* und *LDUP* hat der Benutzer die Möglichkeit, einen Satz mit dem Schlüssel *X'FF...FF'* einzufügen. Dieser Satz muss mit einem USAGE-Modus geschrieben werden, bei dem LEASY nicht selbst die Schlüssel vergibt, z.B. *UPDT*. Er wird dann von LEASY bei der Schlüsselvergabe mit den USAGE-Modi *LOAD* und *LDUP* ignoriert. Dies bewirkt, dass beim Einfügen eines Satzes nicht sämtliche Indexstufen von ISAM berichtigt werden müssen. Siehe „Erläuterung der USAGE-Modi LOAD/PLOD/ELOD und LDUP/PLUP/ELUP“ auf Seite 190.

8.2 Jobvariablen einsetzen

LEASY bietet bietet folgende Jobvariablen zur Überwachung zentraler Resourcen:

*LEACMST Zustand des Common Memory

*LEAIOST Anzahl der aktiven I/O-Tasks

***LEACMST: Zustand des Common Memory in Jobvariable**

Für das Starten einer LEASY-Anwendung in Prozeduren kann der Benutzer eine Benutzer-Jobvariable (JV) verwenden. Informationen über den Zustand des Common Memory werden in der JV hinterlegt und können daraus vom Benutzer entnommen werden. In der JV werden Einträge während des Laufs der Dienstprogramme LEASY-MAINTASK, LEASY-MASTER und LEASY-RECONST hinterlegt.

Maßnahmen des Benutzers

Der Benutzer muss folgende Maßnahmen treffen:

1. Eine JV katalogisieren mit einem *CREATE-JV*-Kommando.

   ```
   /CREATE-JV jvname
   ```

   Der Name *jvname* der JV ist frei wählbar.

2. Den Kettungsnamen *LEACMST* der JV mit einem *SET-JV-LINK*-Kommando zuordnen.

   ```
   /SET-JV-LINK LINK=LEACMST,JV=jvname
   ```

3. Mit einem *MODIFY-JV*-Kommando die JV in der Länge 50 mit einem Wert vorbesetzen. Dazu ist ab Startposition 1 mit der Länge 50 ein beliebiger Wert einzutragen.

   ```
   /MODIFY-JV (jvname,1,50),SET-VAL=C'...'
   ```

Nach diesen Maßnahmen kann sich der Benutzer die von LEASY übergebene Information aus der JV mit dem *SHOW-JV*-Kommando bzw. *GETJV*-Makroaufruf ausgeben lassen.

Verhalten von LEASY

Beim Starten der LEASY-Maintask wird die JV in der Länge von 50 Byte zunächst mit Leerzeichen überschrieben. Danach werden in der JV nacheinander Einträge gemacht, die den aktuellen Zustand des Common Memory wiedergeben. Information wird von LEASY an fünf Stellen innerhalb der JV, nach Einträgen geordnet, hinterlegt. Die Einträge haben eine Länge von jeweils 10 Byte. Die Gesamtlänge aller fünf Einträge ist 50 Byte. Eine Übersicht über die fünf Einträge zeigt die folgende Tabelle.

Eintrag	**1**	**2**	**3**	**4**	**5**
Byte	1 - 10	11 - 20	21 - 30	31 - 40	41-50
Bedeutung	Zustand des CM	Umschalten der AIM-Datei-generation	Rekonstruktion	PETR-Behandlung	Zustand bei ROMS
Inhalt	INIT NORMAL NOT ACTIVE	IN ACTION IN ERROR END (BLANKS)	END ALL END VALID READY	ACTIVE FINISHED WAITING (BLANKS)	ACTIVE ERROR READY END

Erläuterung zur Tabelle

Eintrag 1 — Hier wird die Information über den allgemeinen Zustand des Common Memory (CM) hinterlegt.

- INIT — Die Initialisierung des CM ist noch im Gange.
- NORMAL — Die Maintask läuft normal.
- NOT ACTIVE — Die Maintask ist nicht aktiv. Sie wurde entweder durch LEASY-MASTER beendet, oder es trat ein Fehler auf.

Eintrag 2 — Hier wird die Information über das AIM-FILE Umschalten hinterlegt.

- IN ACTION — Das Umschalten der AIM-Dateigeneration ist im Gange.
- IN ERROR — Das Umschalten der AIM-Dateigeneration war fehlerhaft.
- END — Das Umschalten der AIM-Dateigeneration wurde fehlerfrei beendet. Die AIM-Dateigeneration ist evtl. zu diesem Zeitpunkt noch nicht freigegeben.
- (BLANKS) — Nach Beendigung wird das Feld gelöscht.

Eintrag 3	Hier wird die Information über einen Rekonstruktionslauf hinterlegt. Dieser Eintrag wird nur versorgt, wenn die Rekonstruktion durchgeführt wird (*MOD UPD=YES*) und nach der Rekonstruktion der CMMAIN nicht freigegeben wird (*MOD FRE=YES*).
END ALL	Ende der AIM-Rekonstruktion, bei der alle Transaktionen abgearbeitet wurden.
END VALID	Ende der AIM-Rekonstruktion, bei der alle gültigen Transaktionen abgearbeitet wurden.
READY	Bereit für AIM-Rekonstruktion.
Eintrag 4	Hier wird die Information über die vorläufig beendeten Transaktionen hinterlegt, d.h. der PETR-Behandlung (preliminary ended transactions).
ACTIVE	PETR-Behandlung ist aktiv.
FINISHED	Alle vorläufig beendeten Transaktionen wurden behandelt.
WAITING	Der Common Memory wartet auf die PETR-Behandlung.
(BLANKS)	Ist die PETR-Behandlung abgeschlossen, wird das Feld gelöscht.
Eintrag 5	Hier wird der aktuelle Zustand hinterlegt, in dem sich die Funktion *ROMS* gerade befindet.
ACTIVE	Funktion *ROMS* wurde gestartet, ist aber noch nicht beendet.
ERROR	Funktion *ROMS* wurde abgebrochen. Ein folgender Sicherungslauf ist nicht mehr möglich.
READY	Funktion *ROMS* wurde normal beendet, d.h. der READ-ONLY-Modus ist gesetzt. Die Online-Sicherung für die ausgewählten Dateien kann mit einem vom Benutzer bestimmbaren Sicherungsinstrument gestartet werden.
END	Der READ-ONLY-Modus wurde mit der Funktion *ROMR* erfolgreich zurückgesetzt.

i Ist keine Jobvariable katalogisiert oder zugeordnet, wird das Starten der LEASY-Anwendung **nicht** unterbrochen.

Zustand und Ergebnis eines Rekonstruktionslaufs

Als Voraussetzung für einen erfolgreichen Rekonstruktionslauf muss die Maintask in einem der beiden folgenden Modi gestartet werden:

*USE=RECONST	Anwendungen können nicht laufen; nur Rekonstruktion ist möglich. Dieser Modus ist für Original- und Schattendateien zulässig.
*USE=NORMAL	Anwendungen können ablaufen. Parallel dazu können Schattendateien manuell oder automatisch rekonstruiert werden.

Für Originaldateien ist nur manuelle Rekonstruktion möglich; Schattendateien können sowohl manuell wie auch automatisch rekonstruiert werden.

Bei manueller Rekonstruktion wird das autonom aufgerufene Dienstprogramm LEASY-RECONST nach Ausführung der angeforderten Rekonstruktion beendet.

Bei automatischer Rekonstruktion läuft das Dienstprogramm LEASY-RECONST parallel zu den Anwendungsprogrammen und wird immer dann aktiviert, wenn eine fertige AIM-Generation auf die Schattendateien nachgezogen werden muss. Das Dienstprogramm wird zusammen mit der Anwendung beendet.

Der RECONST-Bereich der JV **LEACMST* wird sowohl bei der Rekonstruktion von Originaldateien wie auch bei der von Schattendateien versorgt. Dadurch kann bei manuellen Rekonstruktionen während einer LEASY-Anwendung programmgesteuert auf das Ende einer Rekonstruktion reagiert werden.

Die Auswertung dieser Information ist nur bei manueller Rekonstruktion sinnvoll; bei automatischer Rekonstruktion wechselt die Information zyklisch bis zum Ende der Session.

Beispiel zu LEACMST

In einer Prozedurdatei soll das Dienstprogramm LEASY-MAINTASK als Stapelprozess mit einem *ENTER*-Kommando gestartet werden. Dazu soll gewartet werden, bis LEASY-MAINTASK normal läuft (siehe die Tabelle auf Seite 114; Eintrag 1 hat den Wert NORMAL). Daran anschließend wird ein Anwenderprogramm gestartet.

Starten der Prozedur für LEASY-MAINTASK und Anwenderprogramm:

```
/BEGIN-PROC LOG=*NO,PAR=*YES(PROC-PAR=(&PROG),ESC-CHAR=C'&')
/CREATE-JV JV.PROG  ------------------------------------------------------------ (01)
/SET-JV-LINK LINK=LEACMST,JV=JV.PROG -------------------------------------------- (02)
/MODIFY-JV (JV.PROG,1,50),SET-VAL=                                               -
/           C'.........0.........0.........0.........0.........0' ------------- (03)
/
/ENTER-JOB E.MTSTART ------------------------------------------------------------ (04)
/WAIT-EVENT UNTIL=JV(COND=((JV.PROG,1,6)=C'NORMAL')),TIME-LIM=600,      -
/                    TIMEOUT=ENDE) ------------------------------------------------ (05)
/START-EXE &PROG ---------------------------------------------------------------- (06)
/.ENDE END-PROCEDURE
```

Stapelprozess in der Datei E.MTSTART:

```
/SET-LOGON-PAR
/SET-JV-LINK LINK=LEACMST,JV=JV.PROG -------------------------------------------- (07)
/START-LEASY-MAINTASK ----------------------------------------------------------- (08)
*CAT=LCAT
*LOG=A
*END
/EXIT-JOB
```

Erklärung:

(01) Katalogisieren der Benutzer-JV mit Namen *JV.PROG*

(02) Benutzer-JV mit dem Linknamen *LEACMST* verknüpfen

(03) Vorbesetzen von *JV.PROG*

(04) Starten des Stapelprozesses aus der Datei *E.MTSTART*

(05) Maximal 600 sec auf Maintask warten, sonst beenden

(06) Starten des Anwenderprogramms

(07) Benutzer-JV mit dem Linknamen *LEACMST* verknüpfen

(08) Starten von LEASY-MAINTASK

*LEAIOST: Anzahl der aktiven I/O-Tasks in Jobvariable

Durch Einführung einer Jobvariablen, die über die Anzahl der aktiven I/O-Tasks informiert, wird dem Benutzer die Möglichkeit gegeben, Programme und Prozeduren abhängig von der Anzahl der aktiven I/O-Tasks zu steuern. Damit lassen sich auch mehrere I/O-Tasks kontrolliert aktivieren.

Maßnahmen des Benutzers

Der Benutzer muss folgende Maßnahmen treffen:

1. Eine Jobvariable katalogisieren mit einem *CREATE-JV*-Kommando.

   ```
   /CREATE-JV jvname
   ```

   Der Name *jvname* der JV ist frei wählbar.

2. Den Kettungsnamen *LEAIOST* mit einem *SET-JV-LINK*-Kommando der JV zuordnen.

   ```
   /SET-JV-LINK LINK=LEAIOST,JV=jvname
   ```

   Nach diesen Maßnahmen kann sich der Benutzer die von LEASY übergebene Information aus der JV mit dem *SHOW-JV*-Kommando bzw. *GETJV*-Makroaufruf ausgeben lassen.

   Der Benutzer muss beachten, dass in allen I/O-Tasks die gleiche JV zugeordnet bzw. abgefragt wird. LEASY prüft nicht, ob eine JV zugeordnet wurde.

Rückinformation von LEASY

LEASY übergibt in der JV folgende 10 Byte lange Information:

Byte	1-10
Inhalt	nnn-ACTIVE *nnn* ist die Anzahl der aktiven I/O-Tasks

8.3 Adressierungsmodus

LEASY schaltet, wenn es nicht unter openUTM läuft, bei jedem Aufruf in den 31-Bit-Adressmodus um. Vor der Rückkehr ins Anwendungsprogramm wird in den ursprünglichen Adressmodus zurückgeschaltet.

Bei Betrieb mit openUTM schaltet LEASY den Adressmodus nicht um.

8.4 LEASY als Subsystem

Das LEASY-Laufzeitsystem LEACONX kann in den Klasse-4-Speicher vorgeladen werden.

Die dazu notwendigen DSSM-Deklarationen befinden sich in der mitgelieferten *SYSSSC*-Datei *SYSSSC.LEASY.062*.

8.5 Koexistenz verschiedener LEASY-Versionen

Ab LEASY V6.0 besteht grundsätzlich volle Versionskoexistenz bezüglich Dienstprogrammen und Laufzeitsystem, d.h., es können mehrere Versionen von LEASY gleichzeitig mit IMON installiert sein und es können auch mehrere verschiedene LEASY-Versionen gleichzeitig eingesetzt werden.
Aus technischen Gründen ist mit LEASY V5.3 und allen LEASY-Versionen davor **keine** Versionskoexistenz möglich.

Allgemeine Voraussetzungen

Die Versionskoexistenz verschiedener LEASY-Versionen ist ab BS2000/OSD V3.0 möglich. Sollen mehrere verschiedene Versionen von LEASY gleichzeitig als Subsystem installiert werden, so ist dies ab DSSM V3.5 und ab SSCM V2.0 möglich.

Wenn Sie die Versionskoexistenz verschiedener LEASY-Versionen nutzen wollen, dann müssen Sie auf die Einhaltung der folgenden Punkte achten:

1. Es muss immer die zur höchsten installierten Version gehörende Meldungs- und SDF-Syntaxdatei eingehängt sein. Dies ist immer dann automatisch erfüllt, wenn die zuletzt installierte Version auch die höchste Version ist.
2. Die verschiedenen LEASY-Versionen müssen mit verschiedenen Katalogen arbeiten, d.h., die Kataloge müssen sich in mindestens einem Namensbestandteil (Benutzerkennung, Katalogkennung oder Katalognamen) unterscheiden. Dies gilt auch für mehrere gleichzeitig laufende LEASY-Systeme der gleichen Version.

3. Bezüglich der Verarbeitung **eines** LEASY-Katalogs ist kein Versionsmix möglich, d.h. ein Anwendungsprogramm muss mit derselben LEASY-Version arbeiten wie die Dienstprogramme.

Versionsauswahl beim Start eines LEASY-Dienstprogramms

Sowohl im Kommando *SELECT-PRODUCT-VERSION* als auch in den *START-LEASY-utility*-Kommandos kann eine 4- bis 7-stellige Version angegeben werden, d.h. es ist auch die Angabe von Korrekturversionen möglich.

Eine im *START-LEASY-utility*-Kommando angegebene Version hat stets Vorrang vor einer in einem eventuell vorangegangenen Kommando *SELECT-PRODUCT-VERSION* spezifizierten Version. Wird in einem *START-LEASY-utility*-Kommando keine Version angegeben, so wird die höchste installierte Version verwendet, falls vor dem *START-LEASY-utility*-Kommando kein *SELECT-PRODUCT-VERSION*-Kommando abgesetzt wurde. Andernfalls übernimmt das *START-LEASY-utility*-Kommando die im Kommando *SELECT-PRODUCT-VERSION* angegebene Version.

Versionsauswahl beim Start eines LEASY-Anwenderprogramms

Wenn Sie mehrere verschiedene LEASY-Versionen über IMON installiert haben, so können Sie die LEASY-Version, mit der Ihr Anwenderprogramm zusammenarbeiten soll, im Kommando *SELECT-PRODUCT-VERSION* auswählen. Das Kommando *SELECT-PRODUCT-VERSION* muss vor dem Start des Anwenderprogramms angegeben werden. Wenn Sie dieses Kommando vor dem Start seines Anwenderprogramms nicht angeben, dann wird die höchste installierte LEASY-Version verwendet.

9 Übersicht über die LEASY-Schnittstelle

Dieser Abschnitt beschreibt die LEASY-Programmschnittstelle unabhängig von der Programmiersprache.

Die LEASY-Schnittstelle für COBOL ist im Kapitel „COBOL-Schnittstelle“ auf Seite 195ff. behandelt.

Die Assembler-Makroaufrufe für LEASY sind im Kapitel „Assembler-Schnittstelle“ auf Seite 221ff. beschrieben.

9.1 LEASY binden

Um LEASY von einem Anwenderprogramm aus aufrufen zu können, muss ein LEASY-Verbindungsmodul in das Anwenderprogramm eingebunden werden, welches dann weitere Module des LEASY-Laufzeitsystems dynamisch nachlädt.

Welcher LEASY-Verbindungsmodul einzubinden ist, hängt zum einen davon ab, ob das Anwenderprogramm ein Batch/TIAM-, DCAM- oder IO-Task-Anwenderprogramm ist, zum anderen davon, ob die Parameter an LEASY wie bisher in früheren LEASY-Versionen übergeben werden (Adresse der Parameterleiste in Register 1, letzter Parameter wird durch Setzen des höchstwertigsten Bits in der zugehörigen Adresse gekennzeichnet) oder gemäß den ILCS-Konventionen (Adresse der Parameterleiste in Register 1, Anzahl der Parameter in Register 0).

Die folgende Tabelle gibt einen Überblick über alle zur Verfügung stehenden LEASY-Verbindungsmodule und die Bibliotheken, in denen sie zu finden sind:

Betriebs-modus	**Art der Parameterübergabe**	**Name des LEASY-Verbindungsmoduls** *leasy_vb_modul*	**Bibliothek, die den LEASY-Verbindungsmodul enthält** *bibliothek*
Batch/TIAM	wie bisher	LEASY	SYSLNK.LEASY.062
Batch/TIAM	ILCS-Konvention	LEASYI	SYSLNK.LEASY.062
DCAM	wie bisher	LEADCAM	SYSLNK.LEASY.062.DCAM
DCAM	ILCS-Konvention	LEADCAMI	SYSLNK.LEASY.062.DCAM
IO-Task	wie bisher	LEASY	SYSLNK.LEASY.062.IOH
IO-Task	ILCS-Konvention	LEASYI	SYSLNK.LEASY.062.IOH
openUTM	Will der Anwender Parameterübergabe nach ILCS-Konventionen, muss er den openUTM-Start-Parameter *.LEASY ILCS* angeben. Fehlt diese Angabe, wird die Parameterübergabe wie bisher erwartet.		

Tabelle 6: Überblick über LEASY-Verbindungsmodule

Informationen zur Parameterübergabe gemäß ILCS-Konventionen finden Sie auch im Abschnitt „LEASY aufrufen“ auf Seite 123.

Der LEASY-Verbindungsmodul kann entweder statisch oder dynamisch eingebunden werden. Das statische Einbinden geschieht im Programm BINDER mit der folgenden Anweisung:

```
//INCLUDE-MODULES LIBRARY=bibliothek,ELEM=leasy_vb_modul,TYPE=R
```

Das dynamische Einbinden des LEASY-Verbindungsmoduls geschieht mit dem BIND-Makro z.B. in der folgenden Weise:

```
BIND …,SYMBOL=leasy_vb_modul,SYMTYP=MODULE,LIBNAM=bibliothek,…
```

Besonders wichtig ist hierbei, dass der Parameter *SYMTYP=MODULE* beim Nachladen des LEASY-Verbindungsmoduls *LEASY* angegeben wird. Da es in den Bibliotheken noch andere Module mit einem **Entry** *LEASY* gibt, kann dies, wenn der Parameter *SYMTYP=MODULE* nicht angegeben wird (d.h. es wirkt die Voreinstellung *SYMTYP=ANY*), auf Grund der Suchstrategie des BIND-Makros dazu führen, dass statt des Verbindungsmoduls LEASY einer der anderen Module mit einem Entry LEASY nachgeladen wird, was in der Regel zu einem Fehler führt.

9.2 LEASY aufrufen

Die LEASY-Schnittstelle stimmt mit einigen Einschränkungen bzw. Erweiterungen mit der KLDS (Kompatible Schnittstelle zu linearen Datenbanken) überein.

In KLDS wird die LEASY-Schnittstelle über einen **Unterprogrammaufruf** (CALL-Aufruf) angesprochen, z.B. bei COBOL in der Form

```
CALL "LEASY" USING par1, par2, ...
```

Die Namen für die Operanden *par1*, *par2* ... sind frei wählbar. Die Reihenfolge ist jedoch, wie bei Unterprogrammaufrufen üblich, nicht beliebig, da die Operanden **Stellungsoperanden** sind.

Parameterübergabe gemäß ILCS-Konventionen:

Anwender, die die Parameter gemäss ILCS-Konventionen an LEASY übergeben wollen, müssen Folgendes beachten:

COBOL-Anwenderprogramme werden durch die Compiler COBOL85 und COBOL2000 automatisch so übersetzt, dass das Register 0 die Anzahl der Parameter enthält.

Bei ASSEMBLER-Anwenderprogrammen muss der Anwender selbst für die richtige Parameterübergabe im Programm sorgen. Die LEASY-Makros *LEA@*... können nicht verwendet werden.

Nach dem Übersetzen muss neu gebunden werden. Hierbei müssen Sie die Informationen aus Abschnitt „LEASY binden“ auf Seite 121 beachten.

Grundsätzlich können LEASY-Anwendungen, die die Parameter wie bisher an LEASY übergeben und solche, die die Parameterübergabe gemäss ILCS- Konvention vornehmen, parallel ablaufen.

9.3 LEASY-Schnittstelle versorgen

Die Operanden sind im LEASY-Aufruf durch ihre Position gekennzeichnet. Dadurch sind ihre Namen frei wählbar und ihr Inhalt ist zur Ablaufzeit veränderbar. Die richtige Reihenfolge im LEASY-Aufruf ist zwingend vorgeschrieben.

Die einzelnen LEASY-Operationen erfordern eine unterschiedliche Anzahl von Operanden (siehe Tabelle „LEASY-OPEN-Modi“ auf Seite 188). Maximal können 9 Operanden definiert werden. Liegen die nicht benötigten Operanden zwischen relevanten Operanden, müssen sie angegeben werden. Zudem kann ihr Inhalt mit Leerzeichen gelöscht werden. Einzige Ausnahme ist der Parameter *US*. Dieser wird immer als letzter Operand an die Operandenliste angehängt, egal wie viele Operanden vorher bereits angegeben wurden. Dass der letzte Operand als *US*-Operand zu interpretieren ist, ergibt sich aus dem Feld *U-PROT* (=*Y*) im *RE*-Bereich. Liegen die nicht benötigten Operanden am Ende der Operandenliste, dürfen sie entfallen.

Tabelle 7 zeigt alle möglichen Operanden und ihre Position im LEASY-Aufruf.

Der Übersichtlichkeit wegen sind die Operanden einheitlich bezeichnet. Diese Namen werden in der gesamten Beschreibung beibehalten.

Position	Name	Bedeutung	Art
1	OP	Operationscode	Ü
2	RE	Verständigungsbereich	Ü/R
3	DB	Dateizuweisung	Ü
	CI	Currency Information	Ü/R
	CAT	Kataloginformation	Ü
4	AR	Ein-Ausgabebereich	Ü/R
5	FA	Feldauswahl	Ü
6	SI	Sekundärindex	Ü
7	KB	Schlüsselanfang	Ü
8	KE	Schlüsselende	Ü
letzte	US	USER-Bereich	Ü

Tabelle 7: Zusammenstellung der auftretenden Operanden

Erläuterung

Ü Übergabe vom Anwenderprogramm an LEASY

R Rückgabe von LEASY an das Anwenderprogramm

Die auf Position 3 stehenden Operanden *DB/CI/CAT* treten alternativ auf.

Operationscode OP

Der Operationscode bezeichnet ein 4 Byte langes alphanumerisches Übergabefeld, das die von LEASY auszuführende Operation bestimmt.

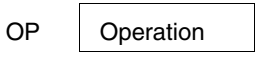


Tabelle 8 zeigt die möglichen LEASY-Operationen

LEASY-Operation	Bedeutung
CATD	LEASY-Katalog ansprechen
OPFL	Dateien eröffnen
CLFL	Dateien schließen
OPTR	Transaktion eröffnen oder erweitern
CLTR	Transaktion schließen
MARK	Sicherungspunkt erzeugen
BACK	Rollback durchführen
RDIR	Satz direkt lesen
RNXT	Nachfolger lesen
RPRI	Vorgänger lesen
RHLD	Satz direkt lesen mit Satzsperre
RNHD	Nachfolger lesen mit Satzsperre
RPHD	Vorgänger lesen mit Satzsperre
SETL	Dateizeiger positionieren
INSR	Noch nicht bestehenden Satz einfügen
STOR	Satz einfügen
REWR	Bestehenden Satz ändern
DLET	Bestehenden Satz löschen
LOCK	Satzsperre setzen
UNLK	Satzsperre aufheben
CINF	Currency Information übergeben

Tabelle 8: LEASY-Operationen

Die LEASY-Operationen sind ab Seite 150 in alphabetischer Reihenfolge beschrieben.

Verständigungsbereich RE

Im Verständigungsbereich werden Informationen vom Anwender an LEASY übergeben, aber auch von LEASY zurückgeliefert.

LEASY liefert zurück:

- den Returncode
- die SAM-Wiedergewinnungsadresse im 24- oder 31-Bit-Format (siehe Handbuch „Einführung in das DVS“)
- die PAM-Blocknummer bei sequenziellem Lesen oder Zugriff über Sekundärschlüssel
- den Operationscode und den Dateinamen des letzten Aufrufes
- die Transaktions-Identifikation für DCAM-Anwendungen bei der Operation *OPTR*.

Der *Anwender* kann angeben:

- den OPEN-Modus
- den USAGE-Modus
- die Steuerung der BIM-Sicherung bei *OPTR* (Eröffnen der Transaktion)
- die SAM-Wiedergewinnungsadresse im 24- oder 31-Bit-Format zum Positionieren mit *SETL* und zum direkten Lesen mit *RDIR* / *RHLD*.
- die PAM-Blocknummer
- das Versionskennzeichen (Versorgungspflicht!)
- Zusatzinformationen für die Operationen *OPTR*, *CLTR*, *RDIR*, *RHLD*, *RNHD*, *RPHD*, *CINF* und *LOCK*
- die Wartezeit auf gesperrte Sätze
- die Identifikationen für DCAM-Anwendungen
- die Transaktions-Identifikation für DCAM-Anwendungen bei jeder Operation innerhalb einer Transaktion.

Unbenutzte Felder sind mit Leerzeichen (*X'40'*) oder binären Nullen (*X'00'*) vorzubesetzen.

Aufbau des Verständigungsbereichs RE

Der Verständigungsbereich *RE* ist 80 Byte lang und besteht aus einem kompatiblen Teil (48 Byte) und der LEASY-Erweiterung (32 Byte). Tabelle 9 zeigt die Struktur des Verständigungsbereichs.

Feldnamen (abgestuft)	Position (Byte)	Länge	Typ	Art	Bedeutung	
RC-CC	1-3	3	A	R	kompatibler Returncode	KompatiblerTeil des RE nach KLDS
RC-KZ	4	1	A	R	Systemkennzeichen „L“	
RC-LC	5-8	4	A	R	LEASY-Returncode	
PASS	9-16	8	A	-	reserviert für Kennwort	
OPE OPE-STX OPE-OM OPE-LOG -----	17-24 17 18 19 20-24	8 1 1 1 5	A A A A -	Ü Ü Ü Ü -	Operationsergänzungen STXIT-Modus OPEN-/USAGE-Modus BIM-Loggingsteuerung reserviert	
INT SAMPTR PAMHPNR ----- SAMPTR	25-32 25-28 25-28 29-32 25-32	8 4 4 4 8	A A B B	Ü/R Ü/R Ü/R - Ü/R	Interner Ordnungsbegriff SAM-WGA (24-Bit-Format) oder PAM-Blocknummer reserviert SAM-WGA (31-Bit-Format)	
NUM	33-40	8	N	R	Anzahl der Primärsätze	
IDE	41-48	8	A	Ü/R	Identifikationsfeld für DCAM-Anwendung	
REOP	49-52	4	A	R	letzter Operationscode	LEASY-Erweiterung des RE
REDB	53-68	16	A	R	letzter Dateiname (+ SI-Name)	
L-OPT	69	1	A	Ü	Versionskennzeichen „1“	
OPE1 OPE2	70 71	1 1	A A	Ü Ü	Operationsergänzungen für OPTR/CLTR/RDIR/RHLD/ RNHD/RPHD/LOCK/CINF	
OPE-WTIME	72-74	3	N		Wartezeit für Sperren	
RC-LCE	75-79	5	A	R	LEASY-Returncode-Ergänzung	
U-PROT	80	1	A	Ü	USER-Information	

Tabelle 9: Struktur des Verständigungsbereichs RE

A alphanumerisches Feld

B numerisches Feld (binär)

N numerisches Feld (abdruckbar)

Ü Übergabe vom Anwenderprogramm an LEASY

R Rückgabe von LEASY an das Anwenderprogramm

Übergabe und Rückgabe in den einzelnen Feldern

Die folgende Tabelle zeigt die Übergabe und Rückgabe in den einzelnen Feldern des Verständigungsbereichs *RE*.

Feld	Art	Inhalt
RC-CC	R	Kompatibler Returncode von KLDS.
RC-KZ	R	„L“ als Kennzeichen für LEASY.
RC-LC	R	Von LEASY intern erzeugter Fehlercode. Dieser Fehlercode ist detaillierter als der kompatible Returncode. Die 4 Bytes von RC-LC können folgenden Aufbau haben: A ddd B ddd C ddd D ddd J ddd S ddd T ddd } DVS-Fehler bei Bearbeitung einer { AIM-Datei BIM-Datei Katalogdatei Primärdatei Jobvariablen (JV) Sekundärindex-Datei Statusdatei ddd: Bei 3-stelliger DVS-Meldungsnummer die rechten 3 Bytes des DVS-Fehlerschlüssels, der die Form 0ddd hat (siehe Handbuch „Systemmeldungen Band 1 bis Band 3“). Bei 4-stelliger DVS-Meldungsnummer (1. Ziffer nicht 0) die Zeichenfolge „DVS“. Das Feld *RC-LCE* enthält dann die 4-stellige DVS-Meldungsnummer (siehe Handbuch „Systemmeldungen Band 1 bis Band 3“). L eee: LEASY-interner Fehlercode. Im Feld *RC-LCE* kann ein weiterer Fehlercode als Zusatzinformation abgeliefert werden Die genaue Auflistung und Bedeutung der kompatiblen Returncodes und der von LEASY erzeugten Rückkehrinformationen finden Sie im Kapitel „Rückkehrinformationen“ auf Seite 401ff.
PASS		reserviert.
OPE-STX	Ü	Angaben im Feld OPE-STX werden ab LEASY V6.1A ignoriert, die STXIT-Routine in LEASY bleibt in jedem Fall aktiviert.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 1 von 8)

<table>
<tr><th>Feld</th><th>Art</th><th>Inhalt</th></tr>
<tr><td>OPE-OM</td><td>Ü</td><td>
Im Feld OPE-OM kann bei den Operationen OPFL und OPTR ein Kennzeichen für die Eröffnungsart der Dateien bzw. Dateikennungen angegeben werden.

OPE-OM= X'FF' bedeutet bei beiden Operationen, dass für die Dateizuweisung DB im 3. Operanden des LEASY-Aufrufs das Format DB4 gewählt wird, bei dem für jede Datei der zugehörige OPEN-Modus im Operanden DB angegeben wird.

Bei der Operation OPFL kann im Feld OPE-OM der 1 Byte lange OPEN-Modus angegeben werden, der dann für alle Dateien (die mit dem Format DB1/DB2 zugewiesen werden) in gleicher Weise gilt. Bei der Operation OPTR ist in diesem Feld, außer X' FF' , auch die Angabe einer 1 Byte langen Verarbeitungsart möglich, die dann für alle Dateikennungen (die mit DB1 oder DB2 zugewiesen werden) in gleicher Weise gilt. Diese Verarbeitungsart wird nach folgender Tabelle auf einen bestimmten LEASY-USAGE-Modus abgebildet:

<table>
<tr><th>Verarbeitungsart</th><th>USAGE-Modus</th></tr>
<tr><td>_ (Standardwert)</td><td>EXLD (SAM)/
UPDT (ISAM/PAM/DAM)</td></tr>
<tr><td>A</td><td>RETR</td></tr>
<tr><td>E</td><td>PRUP</td></tr>
<tr><td>G</td><td>EXRT</td></tr>
<tr><td>L</td><td>EXLD</td></tr>
<tr><td>I</td><td>PRRT</td></tr>
<tr><td>O</td><td>EXLD</td></tr>
<tr><td>Q</td><td>EXLD</td></tr>
<tr><td>X</td><td>EXRT</td></tr>
<tr><td>B</td><td>EXUP</td></tr>
<tr><td>R</td><td>ULRT</td></tr>
<tr><td>U</td><td>ULUP</td></tr>
</table>
Die Angabe einer Verarbeitungsart hat dieselbe Wirkung wie die Spezifizierung des zugeordneten USAGE-Modus für alle angeführten Dateikennungen mittels Format DB4.
</td></tr>
</table>

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 2 von 8)

Feld	Art	Inhalt
OPE-LOG	Ü	Bei der 1. Operation *OPTR* einer Transaktion kann mit der Angabe „N“ die BIM-Sicherung für diese Transaktion abgeschaltet werden. Standardmäßig ist das Feld mit Leerzeichen (X' 40') vorbelegt, d.h. die BIM-Sicherung ist für die laufende Transaktion eingeschaltet, wenn in den Dienstprogrammen LEASY-MAINTASK und LEASY-CATALOG die entsprechenden Operandenwerte zugewiesen sind. Bei Kopplung openUTM-LEASY ist das Ausschalten der BIM-Sicherung nur für lesende Transaktionen erlaubt.
SAMPTR	Ü/R	Bei SAM-Dateien wird nach jeder Operation die aktuelle Wiedergewinnungsadresse in das Feld *SAMPTR* zurückgeliefert, und zwar in dem Format (24- oder 31-Bit), das mit den Operationen *SETL* oder *RDIR* vorgegeben wird (*IDIRPTR*=’bbbbbbrr’ bzw. *IDIRPTR*=’bbbbbbbbrrrrrrrr’). Bei der Operation *SETL* bzw. *RDIR* ist durch den Anwender eine solche Wiedergewinnungsadresse im Feld *SAMPTR* entweder im 24-Bit-Format (’bbbbbbrr’) oder im 31-Bit-Format (’bbbbbbbbrrrrrrrr’) zu hinterlegen. Bei Verwendung des 24-Bit-Formats wird vorausgesetzt, dass das zweite Wort des Feldes *SAMPTR* dann mit binären Nullen oder Blanks vorbesetzt ist. So kann innerhalb der Datei für eine nachfolgende sequenzielle Leseoperation positioniert werden. Bei *RNXT/RNHD* wird auch im 24-Bit-Modus dann auf 31-Bit-Modus umgeschaltet, wenn die Nummer des zu lesenden Satzes im Block 255 übersteigt. Der 31-Bit-Modus bleibt so lange eingeschaltet, bis er evtl. von *SETL* oder *RDIR* wieder auf 24-Bit-Modus gesetzt wird.
PAMHPNR	Ü/R	Im Feld *PAMHPNR* ist bei PAM-Schreiboperationen und direktem Lesen die PAM-Blocknummer zu hinterlegen. Bei sequenziellen Leseoperationen oder Leseoperationen über Sekundärschlüssel besetzt LEASY dieses Feld mit der PAM-Blocknummer.
NUM	R	Im Feld *NUM* übergibt LEASY bei den Operationen *RDIR/RHLD* die Anzahl der Primärsätze, die zu einem Sekundärindex-Wert gehören. Voraussetzung ist, dass im Feld *OPE2* das Kennzeichen „*N*“ angegeben ist und beim Zugriff über Sekundärindex kein Intervall angegeben wurde.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 3 von 8)

Feld	Art	Inhalt		
IDE	Ü/R	Das Feld wird nur belegt, wenn LEASY von einer DCAM-Anwendung aufgerufen wird. Bei der Operation *CATD* ist in *IDE* der DCAM-Anwendungsname zu übergeben. Vor der 1. Operation *OPTR* jeder Transaktion muss *IDE* gelöscht sein. LEASY liefert in diesem Fall bei der Operation *OPTR* die Transaktions-Identifikation zurück. Diese Identifikation muss bei jeder LEASY-Operation, die diese Transaktion betrifft, mitgeliefert werden. Eine Operation *CLTR* löscht das Feld *IDE*.		
REOP/ REDB	R	In den Feldern *REOP* und *REDB* trägt LEASY stets den Operationscode und den Dateinamen (+ SI-Name) des letzten Aufrufs ein. Tritt bei den Operationen *OPFL* (Eröffnen der Dateien) oder *OPTR* (Eröffnen der Transaktion) ein Fehler auf, so wird in *REDB* die den Fehler verursachende Datei mit ihrem OPEN- bzw. USAGE-Modus hinterlegt. Bei der Operation *CATD* (Zuweisen des Katalogs) werden in *REDB* die ersten 16 Bytes des spezifizierten Katalognamens hinterlegt. Der Anwender kann dadurch für die Behandlung von Fehlern eine gemeinsame Fehlerroutine verwenden.		
L-OPT	Ü	LEASY-Schnittstellenkennzeichen. Dieses Feld muss stets mit '*1*' versorgt sein.		
OPE1/OPE2	Ü	In den Feldern *OPE1* und *OPE2* können bei folgenden Operationen Zusatzfunktionen spezifiziert werden:		
		OP=OPTR:	OPE1= '_'	Normaler Transaktionsbeginn (Angabe von *DB*).
			OPE1= 'W'	Transaktionsbeginn mit gleichzeitiger Dateipositionierung (Angabe von *CI* im 3. Operanden).
		OP=CLTR:	OPE1= '_'	Normales Transaktionsende.
			OPE1= 'R'	Rücksetzen der Transaktion.
			OPE2= '_'	Transaktionsende mit Aufgabe aller Dateizugriffsanforderungen.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 4 von 8)

Feld	Art	Inhalt			
OPE1/OPE2 (Forts.)	Ü		OPE2= 'T'		Transaktionsende mit gleichzeitigem Transaktionsbeginn (Konsistenzpunkt mit Freigabe der Satzsperren, aber Beibehaltung der Betriebsmittel und Dateipositionen).
		OP=RDIR/RHLD:			
			OPE2= 'N'		LEASY soll die Anzahl der Primärsätze zu einem Sekundärindex-Wert übergeben (im Feld *NUM*).
		OP=RHLD/RNHD/RPHD/LOCK:			
			OPE1=	'S'	Anlegen eines READ-LOCK beim Sperren
			OPE1=	'_'	Anlegen eines WRITE-LOCK beim Sperren
		OP=RNHD/RPHD:			
			OPE2=	L	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so setzt LEASY den Pointer so, als wäre der Satz gelesen worden.
			OPE2=	_	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so wird nach der Wartezeit der Returncode *99ALL006* übergeben. Der Satz wird nicht gelesen, der Pointer nicht verändert.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 5 von 8)

Feld	Art	Inhalt			
OPE1/OPE2 (Forts.)	Ü	OP=CINF:	OPE1=	'_'	Currency Information über alle in der Transaktion eröffneten Dateikennungen und ihrer aktuellen Dateizeiger. Diese Currency Information ermöglicht mit *OPTR* und *OPE1='W'* eine Transaktionseröffnung mit gleichzeitiger Dateipositionierung.
			OPE1=	'F'	Currency Information über die im LEASY- Katalog enthaltenen Dateien und deren Sekundärindizes.
			OPE2=	{'_' 'C'}	Currency Information (Typ1) über alle im LEASY-Katalog enthaltenen Dateien.
			OPE2=	'O'	Currency Information (Typ1) über alle mit *OPFL* eröffnete Dateien.
			OPE2=	'T'	Currency Information (Typ1) über alle an der Transaktion beteiligten Dateien.
			OPE2=	'S'	Currency Information (Typ2) über die in *CI* spezifierte Datei.
			OPE2=	'W'	Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden.
		– Eine Angabe von *OPE2* ist nur sinnvoll, wenn *OPE1='F'* angegeben wird. – Currency Information vom Typ1 umfasst nur allgemeine Informationen über die Datei. Currency Information vom Typ2 listet alle LEASY-internen Tabellen für die spezifizierte Datei auf.			
OPE1	Ü	OP=UNLK	OPE1=	'_'	Normale Satzfreigabe
			OPE1=	'U'	In Transaktionen ohne BIM-Sicherung werden auch modifizierte Sätze freigegeben.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 6 von 8)

Feld	Art	Inhalt
OPE-WTIME	Ü	Im Feld *OPE-WTIME* kann bei jeder Operation eine individuelle Wartezeit in Sekunden auf gesperrte Sätze oder logische Dateien angegeben werden. Wird das Feld nicht besetzt (*X'40'* oder *X'00'*), so gilt die globale Wartezeit der Session (**TIME*-Operand im Dienstprogramm LEASY-MAINTASK); ohne LEASY-Katalog ist die Voreinstellung *0*. Auch wenn eine *OPTR*-Operation bei einer Dateikennung der angegebenen Dateiliste auf eine USAGE-Modus-Unverträglichkeit zu einer parallelen Transaktion trifft, tritt die angegebene bzw. die globale Wartezeit in Kraft. Läuft diese Wartezeit ab, ohne dass die sperrende Transaktion beendet wurde, so erhält das Anwenderprogramm den Returncode *99ALL110*; ansonsten kann es innerhalb seiner Operation *OPTR* fortfahren.
RC-LCE	R	Die 5 Byte von *RC-LCE* können folgenden Aufbau haben: 1. 4-stelliger Meldungsschlüssel bei DVS-Fehler in der Form Axxxx DVS-Fehler bei AIM-Datei Bxxxx DVS-Fehler bei BIM-Datei Cxxxx DVS-Fehler bei Katalogdatei Dxxxx DVS-Fehler bei Primärdatei Jxxxx DVS-Fehler bei Jobvariablen Sxxxx DVS-Fehler bei Sekundärindex-Datei Txxxx DVS-Fehler bei Statusdatei xxxx 4-stellige DVS-Meldungsnummer (siehe Handbuch „Systemmeldungen Band 1 bis Band 3“) 2. Fehlercodeergänzung für den im Feld *RC-LC* abgespeicherten LEASY-internen Fehlercode in der Form L_eee eee LEASY-interner Fehlercode 3. NKISAM-Makro-Fehlercode für den im Feld *RC-LC* abgespeicherten NKISAM-Makrofehler in der Form Iiiii iiii Haupt-Returncode des NKISAM-Makros (siehe Handbuch „DVS-Makros“).

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 7 von 8)

Feld	Art	Inhalt
RC-LCE (Forts.)	R	4. Sonstiger Makro-Fehlercode für den im Feld *RC-LC* abgespeicherten Makrofehler in der Form Mbbaa bbaa entspricht dem Returncode des entsprechenden Makros in R15 (R15=X' bb0000aa')
U-PROT	Ü	Bei Angabe einer USER-Information muss in diesem Feld bei den Operationen *BACK, CATD, CLFL, CLTR, DLET, INSR, OPFL, OPTR, REWR* und *STOR* der Wert *' Y'* gesetzt werden. Bei den anderen Operationen wird dieses Feld nicht ausgewertet. U-PROT= ' ' Keine USER-Information angegeben U-PROT= 'Y' USER-Information angegeben. Der letzte Operand in der Operandenliste wird als USER-Information interpretiert.

Tabelle 10: Übergabe und Rückgabe in den Feldern des RE-Bereichs (Teil 8 von 8)

Dateizuweisung DB

Mit diesem Operanden werden die zu bearbeitenden Dateien zugewiesen. Diese Zuweisung wird auch als Dateiliste bezeichnet.

Die beiden folgenden Begriffe werden unterschieden:

– **Datei** (*datei*)

– **Dateikennung** (*dateikennung*)

Eine Dateikennung besteht aus dem max. 8-stelligen logischen Namen der Datei und wahlweise einem max. 3stelligen Kennzeichen für ein Folgemerkmal (*fm*). Dateiname und Folgemerkmal werden durch einen Schrägstrich (/) voneinander getrennt.

```
dateikennung : datei[/fm]
```

Zu einer logischen Datei lassen sich beliebig viele unterschiedliche Folgemerkmale definieren. Der Begriff der Dateikennung ermöglicht es, für eine logische Datei parallel mehrere unterschiedliche Dateipositionen aktuell zu halten. So kann man z.B. an verschiedenen Positionen einer Datei einen Lesebefehl (*RDIR*) aufsetzen und von diesen Positionen ein unabhängiges Folgelesen (z.B. *RNXT*) weiterführen. Die Bildung unabhängiger Lesefolgen (z.B. entlang verschiedener Sekundärschlüssel), zu deren Identifizierung das Folgemerkmal dient, ist bei vielen Anwendungen vorteilhaft.

i Sperrelemente verwaltet LEASY jedoch pro Datei und nicht pro Dateikennung.

Nur bei der Operation *OPFL* werden mit dem Operanden *DB* die Namen von logischen Dateien (*datei*) zugewiesen. Bei allen anderen LEASY-Operationen sind die Namen von Dateikennungen (*dateikennung*) anzugeben.

Je nach Anzahl und Verwendung gibt es verschiedene Formate, um die Dateien/Dateikennungen zu spezifizieren.

Format DB1

Dieses Format wird verwendet, wenn nur **eine** Datei bearbeitet werden soll. Der OPEN- bzw. USAGE-Modus (siehe Seite 188f) wird aus dem Feld *OPE-OM* des *RE*-Bereichs entnommen (ungleich *X'FF'*).

Format bei OPFL

```
datei
```

datei logischer Dateiname (max. 8 Zeichen)

Format bei OPTR und allen Lese- und Schreiboperationen

```
datei[/fm]
```

datei/fm	Dateikennung	
	datei	logischer Dateiname (max. 8 Zeichen)
	fm	Folgemerkmal (max. 3 Zeichen)

Beispiel für DB1-Formate

```
bei OPFL         DATEI
bei OPTR         DATEI/ABC
```

Format DB2

Dieses Format erlaubt, eine **variable** Anzahl von logischen Dateien bzw. Dateikennungen anzugeben. Der gemeinsame OPEN- bzw. USAGE-Modus (siehe Abschnitt „Dateien und Transaktionen eröffnen“ auf Seite 188f) wird aus dem Feld *OPE-OM* des *RE*-Bereichs entnommen (ungleich *X'FF'*).

Format bei OPFL

```
(datei1,datei2,...)
```

datei	logische Dateinamen (max. 8 Zeichen)

Format bei OPTR

```
(datei1[/fm1],datei2[/fm2],...)
```

datei/fm	Dateikennungen	
	datei	logische Dateinamen (max. 8 Zeichen)
	fm	Folgemerkmale (max. 3 Zeichen)

i Im Klammerausdruck dürfen keine Leerzeichen angegeben werden.

Beispiel für DB2-Formate

```
bei OPFL         (DATEI1,DATEI2,DATEI3)
bei OPTR         (DATEI1/ABC,DATEI2,DATEI3/XYZ)
```

Format DB3

Dieses Format kann nur bei den Operationen *CLFL* und *UNLK* verwendet werden. Mit *ALL* werden **alle** zugewiesenen Dateien angesprochen.

```
{(ALL)}
{ALL }
```

i Bei Angabe von *ALL* ohne Klammern muss das Feld 12 Byte lang sein.

Format DB4

Dieses Format erlaubt es, für jede angesprochene Datei bzw. Dateikennung einen **eigenen** OPEN- bzw. USAGE-Modus festzulegen (siehe Seite 188f). Das Feld *OPE-OM* des *RE*-Bereichs muss mit *X'FF'* besetzt sein. Dies ist das Kennzeichen für Angabe des Formats DB4.

Formate bei OPFL

für *eine* Datei	für *mehrere* Dateien
(datei,mod)	((datei1,mod1),(datei2,mod2)...)

datei logischer Dateiname (max. 8 Zeichen)

mod OPEN-Modus (1 Zeichen)

Formate bei OPTR

für *eine* Dateikennung	für *mehrere* Dateikennungen
(datei[/fm],mod)	((datei1[/fm1],mod1),(datei2[/fm2],mod2)...)

datei/fm Dateikennung

- datei logischer Dateiname (max. 8 Zeichen)
- fm Folgemerkmale (max. 3 Zeichen)

mod USAGE-Modus (4 Zeichen)

i Im Klammerausdruck dürfen keine Leerzeichen angegeben werden.

Beispiel für DB4-Formate

```
bei OPFL        (DATEI,4)
                ((DATEI1,4),(DATEI2,1),(DATEI3,2))
bei OPTR
                (DATEI/FM,RETR)
                ((DATEI/FM1,RETR),(DATEI/FM2,UPDT),(DATEI,EXUP))
```

Currency Information CI

Die Currency Information enthält folgende Informationen:

- Eine Liste der in der aktuellen Transaktion eröffneten Dateikennungen
- Eine Liste der aktuellen Dateizeiger
  - Sekundär- und Primärschlüssel bei ISAM, PAM und DAM
  - DVS-interner Dateipositionszeiger (Wiedergewinnungsadresse *ID1RPTR*) bei SAM
- Intervallgrenzen.

Mit den Operationscode-Ergänzungen *OPE1=F* und *OPE2=C,O,T,S* kann Currency Information angefordert werden über

– alle im LEASY-Katalog enthaltenen Dateien

– alle mit *OPFL* eröffneten Dateien

– alle an der aktuellen Transaktion beteiligten Dateien

– eine bestimmte anzugebende Datei.

CI wird bei folgenden LEASY-Operationen verwendet:

– Bei *CINF* wird die Currency Information angefordert

– Bei *OPTR* mit *OPE1=W* (Transaktionsbeginn mit gleichzeitiger Dateipositionierung) wird die Currency Information übergeben. Dabei ist die CI in der Form zur Verfügung zu stellen, wie sie bei der zugehörigen *CINF*-Operation erhalten wurde.

Aufbau der Currency Information CI

Die CI ist als Satz variabler Länge strukturiert mit einem 4 Byte langen Längenfeld am Anfang.

Feldname	Position (Byte)	Länge	Art	Bedeutung
ci-slf	1 - 2	2	Ü/R	Längenfeld. Enthält den Wert n+4
ci-slr	3 - 4	2	R	Längenfeld. Enthält die erforderliche Mindestlänge von CI
ci-inf	5 bis n+4	n	R	Informationsfeld der Länge n

Bei der Operation *CINF* mit *OPE1=_* hat der Anwender vor dem Aufruf das Längenfeld *ci-slf* mit der geschätzten Länge seines Informationsfeldes zu besetzen: *ci-slf=4+m*.
Er erhält als Rückmeldung in *ci-slf* die tatsächliche Länge und in *ci-inf* die Currency Information. Ist keine Transaktion offen, so wird *ci-slf=0* geliefert.

Bei der Operation *CINF* mit *OPE1=F* hat der Anwender vor dem Aufruf das Längenfeld *ci-slf* mit der geschätzten Länge seines Informationsfeldes (*ci-slf=4+m*) zu besetzen.

Bei der Operation *CINF* mit *OPE1=F* und *OPE2=S* muss er zusätzlich vor dem Aufruf in *ci-inf* den 8 Zeichen langen logischen Dateinamen der gewünschten Datei hinterlegen.

Der Anwender erhält als Rückmeldung in *ci-slf* die tatsächliche Länge der übergebenen Dateiinformationen und in *ci-inf* die Currency Information. Der Umfang der Liste ergibt sich aus der Operationscode-Ergänzung *OPE2=C, O, T* bzw. *S* im Verständigungsbereich *RE*.

Erfüllt keine Datei die Anforderungen, so wird *ci-slf=0* geliefert.

Wurde die Länge *ci-slf* vom Anwender zu kurz angegeben, so wird von LEASY der Fehlercode *04XLLP11* gemeldet. Reicht die Länge von *ci-inf* aus, um Currency Information von mindestens einer Datei aufzunehmen, so wird ein Teil der Dateiinformation in *ci-inf* hinterlegt und dessen Länge in *ci-slf* mitgeteilt. Das Feld *ci-slr* gibt dann die notwendige Mindestlänge für die gesamte Currency Information an. Anschließend kann der *CINF*-Aufruf mit vergrößertem Antwortbereich wiederholt werden, oder es kann mit *OPE2=W* die Fortsetzung des vorangegangenen *CINF*-Aufrufs angestoßen werden, wobei in *ci*-inf der nächste Teil der Dateiinformation zur Verfügung gestellt wird. Im zweiten Fall ist zu beachten, dass die Werte von *ci-slf* und der Inhalt von *ci-inf* in der ursprünglichen Länge nicht verändert werden dürfen.

Sollte der notwendige Bereich zur Übergabe der Dateiinformationen größer sein als 64K (d.h. mehr als 819 Dateien), so kann die Länge weder in *ci-slf* noch in *ci-slr* übergeben werden. In einem solchen Fall wird generell jeweils nur ein Teil der Dateiinformationen mitgeteilt und das Feld *ci-slr* mit *X'FF'* belegt.

Die folgende Übersicht zeigt die verschiedenen Rückgabewerte.

	ci-inf ausreichend groß?					
	ja		nein			
	Datei vorhanden?		len > 64K?			
	ja	nein	nein		ja	
			Platz für mindestens 1 Datei?		Platz für mindestens 1 Datei?	
			ja	nein	ja	nein
Fehlermeldung	-	-	+	+	+	+
ci-slf	len	0	teillen	0	teillen	0
ci-slr	-	-	len	len	X'FFFF'	X'FFFF'
ci-inf	inf	-	teilinf	-	teilinf	-

Es bedeuten

len	Länge der gesamten Dateiinformation
inf	gesamte Dateiinformation
teillen	Länge des übergebenen Informationsteils
teilinf	Teil der Dateiinformation

Berechnung der Länge ci-slf

Die Länge *ci-slf* errechnet sich folgendermaßen:

Bei OPE1=_

$$\text{ci-slf} = 4 + n*16 + n_1*5 + \sum_{i=1}^{n_2}(\text{KEYLEN}_i + 1) + n_3*8 + \sum_{i=1}^{n_4} 2*\text{KEYLENINT}_i$$

n	Zahl der Dateikennungen. $n = n_1 + n_2$
n_1	Zahl der Dateikennungen von SAM-Dateien
n_2	Zahl der Dateikennungen von ISAM-, PAM- und DAM-Dateien
$n_3 \leq n_1$ $n_4 \leq n_2$	Zahl der Dateikennungen mit aktuellen Intervallgrenzen (KB, KE)
KEYLEN_i	max (*KEYLEN-PRIMDATEI,KEYLEN-SIDATEI*) der (i)ten Dateikennung.
KEYLENINT_i	*KEYLEN-PRIMDATEI* oder *KEYLEN-SIDATEI* der (i)ten Dateikennung, für die Intervallgrenzen gelten.

Für PAM-Dateien gilt stets: *KEYLEN-PRIMDATEI=3*

Für DAM-Dateien gilt stets: *KEYLEN-PRIMDATEI=4*

Für SAM-Dateien gilt stets: *KEYLEN-PRIMDATEI=4* bzw. *8*

Bei OPE1=F und OPE2=C, O, T oder _

$$\text{ci-slf} = 4 + n*88 + v$$

n	Anzahl der Dateien
v	16 oder 0 Platz für LEASY-interne Verwaltungsinformation, falls nur ein Teil der Dateiinformation abgerufen werden soll, um mit *CINF* und *OPE2=W* weitere Teile anzufordern. In diesem Fall ist der Wert *v=16* zu verwenden.

Bei OPE1=F und OPE2=S

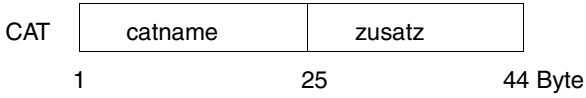
$$\text{ci-slf} = 4 + 111 + s*22 + \sum_{j=1}^{s}(st_j*5 + \sum_{k=1}^{r_j}(rid_{jk}+1))$$

aufgerundet auf ein Vielfaches von 4

s	Anzahl der Sekundärindex-Definitionen der Datei
st_j	Anzahl der Schlüsselteile von Sekundärindex-Definition *j*
r_j	Anzahl der Satzarten-Definitionen von Sekundärindex-Definition *j*
rid_{jk}	Länge der Satzarten-Definition *k* in Sekundärindex-Definition *j*.

Kataloginformation CAT

Dieser Operand muss bei der Operation *CATD* versorgt werden.



Feldname	Position (Byte)	Länge	Art	Bedeutung
catname	1 - 24	24	Ü	Name des LEASY-Katalogs
zusatz	25 - 44	20	Ü	Zusatzangabe für Modelldateien

In *catname* ist der Name des LEASY-Katalogs *[:catid:][$userid.]dateikatalog* anzugeben. Wird für *catname LINK=linkname* angegeben, können durch Zuweisung über das Kommando

```
/ADD-FILE-LINK LINK=linkname,F-NAME= [:catid:][$userid.]katalogname.LEASYCAT
```

ohne Änderung des Anwendungsprogrammes unterschiedliche Dateikataloge verarbeitet werden. In der Zeichenfolge *LINK=linkname* dürfen keine Leerzeichen vorkommen.

Die Angabe der Benutzerkennung kann entfallen, wenn der LEASY-Katalog unter derselben Benutzerkennung katalogisiert ist, unter der das aufrufende Programm abläuft.

Die Zusatzangabe verwendet LEASY für die Auswahl der gewünschten Exemplare aus Modelldateigruppen (*LEASYTYPE=M*) bei den Operationen *OPFL/OPTR*.

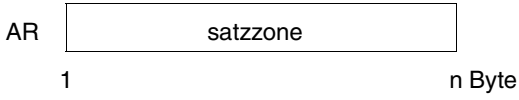
i Beachten Sie folgende Einschränkungen:

- *catname* und *zusatz* müssen rechts mit Leerzeichen aufgefüllt sein.
- Unterstützung von MPVS:
  Will sich eine LEASY-Anwendung an den CMMAIN eines LEASY-Katalogs anschließen, der nicht auf dem Pubset der Kennung eingerichtet ist, unter der das Anwenderprogramm gestartet wird, so ist in *catname* eine Katalogkennung (:*catid*:) für das Pubset mit dem LEASY-Katalog anzugeben.
- Einsatz im Mehrrechnersystem:
  Befindet sich der LEASY-Katalog auf einem fremden Rechner, so ist die Katalogkennung (:*catid*:) dieses Rechners als Bestandteil des Katalognamens mit anzugeben:

  ```
  [:catid:][$userid.]dateikatalog
  ```

Ein-Ausgabebereich AR

Der Operand *AR* verweist auf einen Übergabe- oder Rückgabebereich. Die Länge dieses Bereichs ist variabel.

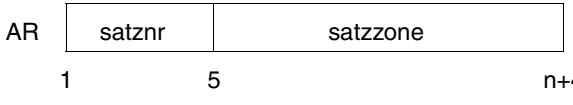


Mit dem Operanden *AR* ist bei Lese- und Schreiboperationen der Ein-Ausgabebereich in der Länge des Satzes zur Verfügung zu stellen. Das Übertragen des Satzes erfolgt bei Schreiboperationen immer in der vollen Satzlänge; bei Leseoperationen kann auf Schlüsselfelder beschränkt werden (vgl. Feldauswahl *FA*), wobei diese an ihren richtigen Positionen im *AR*-Bereich abgeliefert werden.

Der Ein-Ausgabebereich einer Datei wird auch als **Satzzone** *AR* bezeichnet.

Bei variablem Satzformat wird in der Satzzone das Satzlängenfeld bei Leseoperationen mitgeliefert; bei Schreiboperationen muss es der Anwender versorgen.

Bei DAM-Dateien hat der *AR*-Bereich folgenden Aufbau:

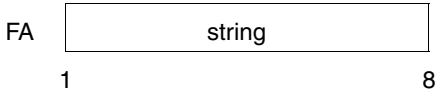


Feldname	Position (Byte)	Länge	Art	Bedeutung
satznr	1 - 4	4	Ü/R	relative Satznummer (binär)
satzzone	5 bis n+4	n	Ü/R	Satzzone der Länge n

Die relative Satznummer ist nicht Bestandteil des Satzes, wird aber im *AR*-Bereich (Byte 1-4) vom Anwender bzw. von LEASY versorgt.

Feldauswahl FA

Der Operand *FA* bezeichnet ein alphanumerisches Übergabefeld. Sein Inhalt legt fest, ob der ganze Datensatz oder nur die Schlüsselwerte in die Satzzone *AR* zurückgeliefert werden sollen.
Dieser Operand ist aus Kompatibilitätsgründen zu KLDS anzugeben, wenn im LEASY-Aufruf der Operand *SI* (6. Operand) folgt.



Für *string* können folgende Zeichenfolgen angegeben werden:

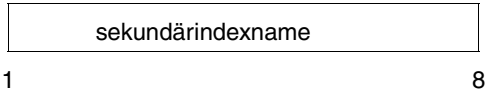
(ALL) Legt fest, dass der ganze Datensatz in die Satzzone *AR* zurückgeliefert wird (5 Byte lange Zeichenfolge).

ALL Legt fest, dass der ganze Datensatz in die Satzzone *AR* zurückgeliefert wird (8 Byte lange Zeichenfolge, mit Leerzeichen aufgefüllt).

MAINITEM Legt fest, dass bei allen Leseoperationen bei ISAM-, PAM- und DAM-Dateien nur die Schlüsselinhalte (Primärschlüssel und der jeweils aktuelle LEASY-Sekundärschlüssel) zurückgeliefert werden. Das Lesen des Primärsatzes unterbleibt. Dadurch wird z.B. beim sequenziellen Lesen entlang eines Sekundärschlüssels der direkte Zugriff auf den Primärsatz eingespart.

Sekundärindex SI

Der Operand *SI* bezeichnet ein 8 Byte langes alphanumerisches Übergabefeld.

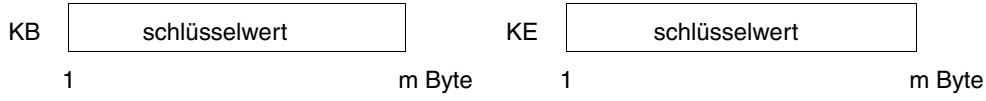


Bei den Operationen *RDIR*, *RHLD* und *SETL* kann in diesem Feld der Name eines Sekundärindex angegeben werden, über den zugegriffen werden soll. Dieser Name muss vorher für die betreffende Datei mit dem Dienstprogrammm LEASY-CATALOG vereinbart worden sein (siehe **FIL*-Anweisung, Operand *KEY*) oder als ISAM-Sekundärschlüsel definiert sein (Kommando *CREATE-ALTERNATE-INDEX* bzw. Makro *CREAIX*).

Besteht der Name des Sekundärindex aus Leerzeichen oder der Zeichenfolge *MAINITEM*, wird über den **Primärschlüssel** zugegriffen.

Schlüsselanfang KB und Schlüsselende KE

Die Operanden *KB/KE* bezeichnen Übergabefelder, die Schlüsselwerte enthalten. Die Länge der Felder richtet sich nach der Art der Schlüsselwerte, die sie aufnehmen sollen.



m Schlüssellänge

Bei den Operationen *RDIR/RHLD* und *SETL* kann mit den Operanden *KB* und *KE* ein Schlüssel**intervall** für den über den Operanden *SI* spezifizierten Index definiert werden, innerhalb dessen sequenziell gelesen werden soll. (Primär- und/oder Sekundärschlüssel)

Das angegebene Format für die SAM-Wiedergewinnungsadresse (4- bzw. 8-Byte-Format) im Feld *SAMPTR* im *RE*-Bereich bestimmt die Angabe in *KB* und *KE*. Das heißt, ist im *RE*-Bereich eine 8-Byte-Adresse angegeben, werden in *KB* und *KE* ebenfalls 8-Byte-Wiedergewinnungsadressen erwartet. Ebenso verhält es sich mit 4-Byte-Adressen.

Bei den Operationen *LOCK* und *UNLK* kann mit den Operanden *KB* und *KE* ein Schlüsselintervall definiert werden, das gesperrt bzw. entsperrt werden soll.

Der Inhalt von *KB* kann größer, kleiner oder gleich dem Inhalt von *KE* sein.
Die Erklärung der unterschiedlichen Wirkung ist bei den Einzeloperationen zu finden.

Innerhalb des hier definierten Intervalls kann über die Operationen *RNXT/RNHD* bzw. *RPRI/RPHD* in auf- bzw. absteigender Schlüsselfolge gelesen werden.

Bei Erreichen einer Intervallgrenze wird der Returncode *010LL003* geliefert.

Bei Zugriff über den Primärindex (*SI=MAINITEM* oder *SI=Leerzeichen*) sind in *KB*, *KE* Primärschlüsselwerte in der Länge des Primärschlüssels anzugeben. Bei PAM-Dateien sind die PAM-Blocknummern (4 Byte), bei SAM-Dateien die Wiedergewinnungsadressen (*ID1RPTR*) im 24-Bit-Format (4 Byte) oder im 31-Bit-Format (8 Byte), bei DAM-Dateien die relativen Satznummern (4-Byte) zu übergeben (analog zum Feld *PAMHPNR/SAMPTR* im *RE*-Bereich).

Beim Zugriff über einen Sekundärindex sind die Sekundärschlüssel-Werte in der Länge des aktuellen Sekundärindex (Operand *SI*) anzugeben.

Ist dieser logische Sekundärindex aus mehreren Sekundärschlüssel-Teilen zusammengesetzt definiert (siehe **FIL*-Anweisung für LEASY-CATALOG), so muss der Anwender die Zusammensetzung des gesamten Indexwertes aus den einzelnen Teilen selbst vornehmen.

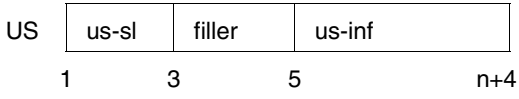
Bei der Operation *SETL* sind bei der Übergabe der Schlüsselwerte zuerst der Sekundärindex-Wert und anschließend der Primärindex-Wert anzugeben.

Sonderfunktion bei Angabe von KB ohne KE

In diesem Fall (genau 7 Operanden im LEASY-Aufruf angegeben) wird kein Intervall definiert, sondern nur ein einziger Schlüsselwert übergeben. Man erzielt die identische Wirkung, als ob man den Primär- bzw. Sekundärschlüssel über die Satzzone *AR* übergeben hätte (bei Verwendung von 4, 5 oder 6 Operanden).

USER-Bereich US

Dieser Operand definiert einen USER-Bereich für die USER-Information. Diese Information wird in die AIM-Datei aufgenommen und kann vom Dienstprogramm LEASY-RECONST protokolliert werden.



Feldname	Position (Byte)	Länge	Art	Bedeutung
us-sl	1 - 2	2	Ü	Länge der User=Information (=n+4)
filler	3 - 4	2	Ü	Füllzeichen
us-inf	5 bis n+4	n	Ü	USER-Information mit variabler Länge

Bei Angabe eines USER-Bereichs muss im Verständigungsbereich *RE* das Kennzeichen *U-PROT=Y* gesetzt werden (siehe Tabelle „Übergabe und Rückgabe in den Feldern des RE-Bereichs“, Zeile „U-PROT“ auf Seite 135).

9.4 LEASY-Operationen

Die LEASY-Operationen lassen sich in 4 Gruppen einteilen:

Organisationsoperationen		CATD, OPFL, CLFL
Transaktionsoperationen		OPTR, CLTR, MARK, BACK
Operationen auf Dateiebene	Leseoperationen	RDIR, RNXT, RPRI
	Leseoperationen mit Sperrnebenwirkung	RHLD, RNHD, RPHD
	Positionierungsoperation	SETL
	Schreiboperationen	INSR, STOR, REWR, DLET
	Sperroperationen	LOCK, UNLK
Operation zur Gewinnung einer Currency Information		CINF

Tabelle 11: Die LEASY-Operationen

Diese Operationen sind im Folgenden in alphabetischer Reihenfolge beschrieben.

BACK Rollback durchführen

Operanden im LEASY-Aufruf:

OP,RE[,US]

Funktion

Die Operation *BACK* setzt die aktuelle Transaktion zurück, wobei alle Dateiänderungen mit der BIM-Datei rückgängig gemacht werden.

Alle von LEASY verwalteten Satzsperren werden freigegeben, und es wird ein Konsistenzpunkt gesetzt. Gleichzeitig wird eine Folgetransaktion begonnen, in der alle zum Zeitpunkt der Operation *BACK* eröffneten Dateikennungen wieder eröffnet sind. Die Dateikennungspositionen zeigen jedoch alle auf Dateianfang.

Diese Operation ist nur im Teilnehmerbetrieb (Stapel und Dialog), nicht aber im Teilhaberbetrieb mit openUTM erlaubt.

BACK hat die gleiche Wirkung wie *CLTR* mit *OPE1=R* und *OPE2=T*, nur dass die Dateipositionen nicht erhalten bleiben.

CATD LEASY-Katalog ansprechen

Operanden im LEASY-Aufruf:

OP,RE,CAT[,US]

Funktion

Die Operation *CATD* weist den LEASY-Katalog zu, der mit dem Dienstprogramm LEASY-CATALOG erstellt wurde. In den folgenden Operationen *OPFL* (Eröffnen der Dateien) und *OPTR* (Eröffnen der Transaktion) wird auf die Dateien dieses LEASY-Katalogs zugegriffen.

Für etwaige Modelldateien, d.h. Dateien mit *LEASYTYPE=M* kann ein Zusatzname angegeben werden. Dieser Zusatz gilt für alle angesprochenen Modelldateien.

Die Operation *CATD* kann nur durchgeführt werden, wenn noch keine Dateien bzw. Transaktionen für den Prozess eröffnet sind.
Bei Kopplung mit openUTM wird diese Operation nicht vom Anwender, sondern von openUTM in dessen Startphase aufgerufen.

Soll auf einen LEASY-Katalog zugegriffen werden, der unter einer anderen Benutzerkennung katalogisiert ist als der, unter der das Programm abläuft, so ist der Name *$userid.dateikatalog* anzugeben.

Ein Leerzeichen als erstes Zeichen des Katalognamens bedeutet, dass ohne Dateikatalog gearbeitet werden soll. Alle Dateien werden dann implizit als Fremddateien betrachtet.

Wird die *CATD*-Operation überhaupt nicht durchgeführt, so hat dies die gleiche Wirkung wie eine *CATD*-Operation mit *Katalogname=Leerzeichen*.

Wurde mittels des Operanden *CATD* bereits ein LEASY-Katalog zugewiesen und wird eine weitere *CATD*-Operation mit Leerzeichen als erstes Zeichen des Katalognamens durchgeführt, so bewirkt dies, dass sich LEASY vom Katalog abhängt und der ursprüngliche Zustand (Arbeiten ohne LEASY-Katalog) wiederhergestellt wird. Insbesondere wird bei dieser Operation die LEASY-STXIT-Routine wieder deaktiviert.

i Meldet sich ein Anwenderprogramm wiederholt mit *CATD katalog* an und mit *CATD* wieder ab, so muss beachtet werden, daß für ein Programmsystem nur eine begrenzte Anzahl von STXIT-Verwaltungsblöcken angelegt werden kann (bei BS2000/OSD-BC V4.0 maximal 100).

Systemtechnisch erfolgt der Anschluss des Prozesses an den zugehörigen Common Memory CMMAIN. Der Common Memory CMMAIN ist der gemeinsame Speicherplatz aller Prozesse, die an einen bestimmten LEASY-Katalog angeschlossen sind.

Ist der angesprochene Common Memory CMMAIN im System vorhanden, aber noch in der Initialisierungsphase durch das Dienstprogramm LEASY-MAINTASK, so tritt eine Wartezeit von maximal 5 Minuten in Kraft, während deren Ablauf jede Sekunde geprüft wird, ob die Initialisierungsphase bereits beendet wurde.

Geht CMMAIN während dieser Zeit in den Zustand „Bereit für Anwenderprogramme“ über, so kann das Programm ablaufen. Läuft die gesamte Wartezeit ab, ohne dass dieser Zustand erreicht wurde, so erhält das Anwenderprogramm je nach Zustand des CMMAIN den Returncode *99ALLS01* oder *99ALLS04*.

i Wird LEASY von einer DCAM-Anwendung aufgerufen, muss zusätzlich der DCAM-Anwendungsname im Feld *IDE* des *RE*-Bereichs übergeben werden.
Der Anwendungsname darf nicht aus Leerzeichen (*X'40'*) oder *X'00'* bestehen. Alle anderen EBCDI-Codes sind erlaubt. Der Feldinhalt von *IDE* wird gelöscht (*X'00'*), wenn die Operation erfolgreich durchgeführt wurde.

CINF Currency Information übergeben

Operanden im LEASY-Aufruf:

OP,RE,CI

Funktion

Die Operation *CINF* fordert die Currency Information an. Dies ist die Liste aller in der Transaktion eröffneten Dateikennungen und ihrer aktuellen Dateizeiger, d.h. die Schlüsselbegriffe der Sätze oder Blöcke, auf die bei der letzten Leseoperation zugegriffen wurde bzw. auf die positioniert wurde. Bei ISAM-, PAM- und DAM-Dateien wird diese Position mit den Primär- und Sekundärschlüsseln dargestellt, bei SAM-Dateien mit der DVS-Wiedergewinnungsadresse *ID1RPTR*.

Mit den Operations-Code-Ergänzungen *OPE1=F* und *OPE2=C,O,T,S* kann Currency Information angefordert werden über:

– alle im LEASY-Katalog enthaltenen Dateien

– alle mit *OPFL* eröffneten Dateien

– alle an der aktuellen Transkation beteiligten Dateien

– eine bestimmte angegebene Datei.

Die Informationen umfassen LEASY-interne Tabellen, die im Wesentlichen die Angaben aus dem LEASY-Katalog zu den Dateien und Sekundärindizes enthalten. Die Currency Information wird im *CI*-Bereich, den der Benutzer in geeigneter Länge zur Verfügung stellen muss, als Satz mit variabler Satzlänge geliefert. Ist keine Transaktion eröffnet (*OPE1=_*) bzw. erfüllt keine Datei die Anforderungen (*OPE1=F*), so wird *ci-slf=0* zurückgeliefert.
Die Currency Information (*OPE1=_*) kann dazu verwendet werden, eine Transaktionseröffnung mit gleichzeitiger Dateipositionierung durchzuführen (siehe Operation *OPTR*, mit Zusatzfunktion *OPE1=W* und Angabe von *CI* als 3. Operand).

Zusatzfunktion (Angaben im RE-Bereich)

OPE1=_	Currency Information über alle in der Transaktion eröffneten Dateikennungen und ihre aktuellen Dateizeiger. Diese Currency Information ermöglicht mit *OPTR* und der Zusatzfunktion *OPE1=W* eine Transaktionseröffnung mit gleichzeitiger Dateipositionierung.
OPE1=F	Currency Information über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes.
OPE2 = { – / C }	Currency Information (Typ1) über alle im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes.
OPE2=O	Currency Information (Typ1) über alle mit OPFL eröffneten Dateien.
OPE2=T	Currency Information (Typ1) über alle an der Transaktion beteiligten Dateien.
OPE2=S	Currency Information (Typ2) über die in CI spezifizierte Datei.
OPE2=W	Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden.

Tabelle 12: Currency Information, Angaben im RE-Bereich

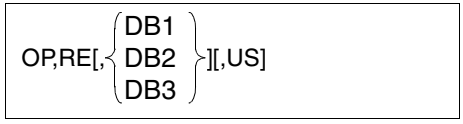
Eine Angabe von *OPE2* ist nur sinnvoll, wenn *OPE1=F* angegeben wird.

Currency Information vom Typ1 umfasst nur allgemeine Informationen über die Datei.

Currency Information vom Typ2 listet alle LEASY-internen Tabellen für die spezifizierte Datei auf.

CLFL Dateien schließen

Operanden im LEASY-Aufruf:



Funktion

Die Operation *CLFL* schließt die in der Dateiliste angegebenen Dateien. Es können

– alle Dateien (Format DB3 oder fehlender Operand DB) oder

– ausgewählte Dateien (Formate DB1/DB2)

geschlossen werden, die in vorangehenden Operationen *OPFL* eröffnet wurden. Die Operation ist nur zulässig, wenn für den Prozess keine **Transaktion** eröffnet ist.

Der Operand *DB3* kann aus Kompatibilitätsgründen zur Schnittstelle KLDS angegeben werden. Die Operation hat dann die gleiche Wirkung wie bei Angabe von nur 2 Operanden (Schließen aller Dateien).

Beispiel

```
OPFL      (D1,D2,D3,D4)
    OPTR  (D1,D2,D3,D4)
    .
    .
    .
    CLTR
CLFL      D2
    OPTR  (D1,D3,D4)
    .
    .
    .
    CLTR
CLFL      (ALL)
```

i Im openUTM-Betrieb ist diese Operation verboten.

CLTR Transaktion schließen

Operanden im LEASY-Aufruf:

OP,RE[,US]

Funktion

Diese Operation beendet die aktuelle Transaktion und setzt einen Konsistenzpunkt, d.h. die zugehörige BIM-Datei wird als „leer“ gekennzeichnet.

Zusatzfunktionen (Angaben im RE-Bereich)

OPE1=R — Die Transaktion wird zurückgesetzt, wobei alle Dateiveränderungen mit Hilfe der BIM-Datei rückgängig gemacht werden.

Wird diese Zusatzfunktion verwendet, obwohl die BIM-Sicherung für diese Transaktion nicht eingeschaltet ist (Parameter *LOG* der LEASY-Maintask), so wird der Returncode *99ALL014* ausgegeben, die Transaktion aber normal beendet.

OPE2=T — Die Transaktion wird beendet und gleichzeitig eine Folgetransaktion begonnen.
Für die abgeschlossene Transaktion wird ein Konsistenzpunkt gesetzt und die Satz- und Blocksperren aufgehoben.
Alle Dateikennungen bleiben jedoch für die Folgetransaktion mit den gleichen USAGE-Modi eröffnet; die Dateipositionen bleiben erhalten.
War für die ursprüngliche Transaktion die BIM-Sicherung abgeschaltet (*OPE-LOG=N*), gilt dies auch für Folgetransaktionen.

Wird LEASY von einer DCAM-Anwendung aufgerufen, ist im Feld *IDE* die Transaktions-Identifikation zu übergeben. Wird die Operation erfolgreich durchgeführt, wird *IDE* gelöscht (*X'00'*), falls nicht *OPE2=T* gesetzt wurde. Alle Satz- und Blocksperren werden aufgehoben.

Wurde beim Eröffnen ein implizites *OPFL* durchgeführt, d.h. dass ohne OPFL gearbeitet wurde, so werden die zugewiesenen Dateien und die BIM-Datei auch physikalisch geschlossen (außer bei *OPE2=T*). Dieser Vorgang wird als implizites *CLFL* bezeichnet.

Im Teilnehmerbetrieb sind alle Varianten von *CLTR* erlaubt.

Bei openUTM-Kopplung ist die Zusatzangabe *OPE2=T* verboten, da sie dem Transaktionskonzept von openUTM widerspricht.
Blocksperren des DVS kann LEASY bei *CLTR* nicht aufheben. (Nur relevant bei ISAM-, DAM- und PAM-Dateien, die als Fremddateien mit SHARED-UPDATE eröffnet werden).

DLET Bestehenden Satz löschen

Operanden im LEASY-Aufruf:

```
OP,RE,DB1[,AR[,FA,SI,KB]][,US]
```

Funktion

Die Operation *DLET* löscht einen Satz aus einer ISAM-Datei oder DAM-Datei bzw. einen Block aus einer PAM-Datei. Ist Aktualisierung der Sekundärindex-Verweise angegeben, so werden auch die Einträge in der Sekundärindex-Datei (SI-Datei) mitgelöscht. Der von LEASY intern geführte Positionszeiger für die Dateikennung wird nicht verändert.

Schlüsselwert übergeben

Die folgende Tabelle zeigt die Möglichkeiten für die Übergabe der Schlüsselwerte in Abhängigkeit von der Dateiart und der Anzahl der Operanden im LEASY-Aufruf.

Dateiart	**Anzahl der Operanden**	**versorgt werden**		**gelöscht wird**
ISAM, DAM	7	DB1 KB	mit Dateinamen mit Primärschlüssel	Satz mit dem Primärschlüssel aus *KB*
	4	DB1 AR	mit Dateinamen mit Primärschlüssel an der definierten Position bei ISAM; in den ersten vier Byte bei DAM	Satz mit dem Primärschlüssel aus *AR*
	3	DB1 AR	mit Dateinamen wird nicht versorgt	Satz, der zuletzt über dieselbe Dateikennung erfolgreich gelesen wurde
PAM	7	DB1 KB	mit Dateinamen mit PAM-Blocknummern	Block mit der PAM-Blocknummer aus *KB*
	3	DB1 PAMHPNR (*RE*-Bereich)	mit Dateinamen mit PAM-Blocknummer	Block mit der PAM-Blocknummer aus *PAMHPNR*. Für *PAMHPNR=0*: Block, der zuletzt über dieselbe Dateikennung erfolgreich gelesen wurde

Tabelle 13: Übergabe der Schlüsselwerte bei der Operation DLET

Wird für die Datei ein Sperrprotokoll geführt, so muss der Satz bzw. Block innerhalb der Transaktion gesperrt worden sein (nicht zwingend über dieselbe Dateikennung). Gelöschte Sätze bzw. Blöcke bleiben automatisch bis Transaktionsende gesperrt.

Wenn die Datei als Fremddatei mit SHARED-UPDATE eröffnet wird, müssen bei einer **PAM-Datei** die Blöcke vorher mit *RHLD/RNHD/RPHD/LOCK* gesperrt worden sein.

INSR Noch nicht bestehenden Satz einfügen

Operanden im LEASY-Aufruf:

```
OP,RE,DB1,AR[,US]
```

Funktion

Ein neuer Satz bzw. Block wird in die angegebene Datei eingetragen und alle erforderlichen Sekundärschlüssel-Werte in der SI-Datei aufgebaut, falls die Änderungen in der SI-Datei sofort erfolgen sollen (siehe Dienstprogramm LEASY-CATALOG, Operand *KEY* der **FIL*-Anweisung).
Der Satz bzw. Block muss in der Satzzone *AR* zur Verfügung gestellt werden.

Bei einer SAM-Datei wird der Satz am Dateiende angefügt und die Wiedergewinnungsadresse im Feld *SAMPTR* des *RE*-Bereichs zurückgeliefert, und zwar bei Sätzen mit Satznummer im Block ≤ 255 im 24-Bit-Format und > 255 im 31-Bit-Format.

Bei einer ISAM-, DAM- bzw. PAM-Datei wird der Satz bzw. Block entsprechend seines Primärschlüssels eingefügt. Es darf noch kein Satz bzw. Block mit dem angegebenen Primärschlüssel existieren. Bei einer PAM-Datei ist der Primärschlüssel im Feld *PAMHPNR* des *RE*-Bereichs zu hinterlegen.

Durch *INSR* eingebrachte Sätze bzw. Blöcke sind automatisch bis Transaktionsende exklusiv gesperrt.

Der von LEASY intern geführte Positionszeiger für die Dateikennung wird nicht verändert.

LOCK Satzsperre setzen

Operanden im LEASY-Aufruf:

```
OP,RE,DB1[,AR[,FA,SI,KB[,KE]]]
```

Funktion

Die Operation *LOCK* legt bei ISAM-, PAM- und DAM-Dateien Sperrelemente an für

– einzelne Sätze bzw. Blöcke, die durch einen Primärschlüssel

– Dateiabschnitte, die durch ein Primärschlüsselintervall

identifiziert sind.

Da mit der Operation *LOCK* kein Dateizugriff erfolgt, wird auch nicht geprüft, ob der entsprechende Satz bzw. Block oder Dateiabschnitt in der Datei vorhanden ist. Das bedeutet, dass auch Sätze bzw. Blöcke oder Dateiabschnitte gesperrt werden können, die noch nicht existieren (Phantomsperren).

Sperrelemente können nur für Dateien angelegt werden, für die auch ein Sperrprotokoll geführt wird. Siehe dazu den Abschnitt „Dateisperren“ auf Seite 19f.

Die Operation *LOCK* bietet die Möglichkeit, mit Hilfe der Sperrelemente einer (evtl. leeren) Datei transaktionsorientierte Koordinierungsaufgaben allein über Speicherverwaltung - ohne Dateizugriffe - zu realisieren.

Übergabe des Schlüsselwertes

Die folgende Tabelle 14 zeigt die Möglichkeiten für die Übergabe der Primärschlüsselwerte in Abhängigkeit von der Dateiart und der Anzahl der Operanden im LEASY-Aufruf.

Dateiart	**Anzahl der Operanden**	**versorgt werden**		**gesperrt wird**
ISAM, DAM	8	DB1	mit Dateinamen	Dateiabschnitt, der von den Primärschlüsseln *KB/KE* begrenzt wird
		SI	mit Leerzeichen oder „MAINITEM“	
		KB, KE	Primärschlüssel der Intervallgrenzen. Der Inhalt von *KB* kann größer, kleiner oder gleich dem Inhalt von *KE* sein.	
	7	DB1	mit Dateinamen	Satz mit dem Primärschlüssel aus *KB*
		SI	mit Leerzeichen oder „MAINITEM“	
		KB	mit Primärschlüssel	
	4	DB1	mit Dateinamen	Satz mit dem Primärschlüssel aus *AR*-Bereich
		AR	mit Primärschlüssel an der definierten Position bei ISAM; in den ersten 4 Byte bei DAM	
PAM	8	DB1	mit Dateinamen	Dateiabschnitt, der von den PAM-Blocknummern *KB/KE* begrenzt wird
		SI	mit Leerzeichen oder „MAINITEM“	
		KB, KE	Primärschlüssel der Intervallgrenzen. Der Inhalt von *KB* kann größer, kleiner oder gleich dem Inhalt von *KE* sein.	
	7	DB1	mit Dateinamen	Block mit der PAM-Blocknummer aus *KB*
		SI	mit Leerzeichen oder „MAINITEM“	
		KB	PAM-Blocknummer	
	3	DB1	mit Dateinamen	Block mit der PAM-Blocknummer aus *PAMHPNR*
		PAMHPNR (*RE*-Bereich)	mit PAM-Blocknummer	

Tabelle 14: Übergabe der Schlüsselwerte bei der Operation LOCK

Zusatzfunktionen (Angaben im RE-Bereich)

OPE1=S	Für die Datei wird beim Sperren ein READ-LOCK abgesetzt.
OPE1=_	Für die Datei wird beim Sperren ein WRITE-LOCK abgesetzt.

Das Sperren eines Primärschlüsselintervalls ist nur möglich, wenn sich für das ganze Intervall keine Unverträglichkeit mit Intervallen oder Schlüsselwerten anderer Transaktionen ergibt.

Wird die Datei als Fremddatei mit *SHARED-UPDATE=YES* eröffnet, so ist die Operation *LOCK* nur für PAM und DAM, nicht für ISAM-Dateien wirksam. Der Sperrauftrag wird dann auf den UPAM-Sperrmechanismus abgebildet.

MARK Sicherungspunkt erzeugen

Operanden im LEASY-Aufruf:

OP,RE[,US]

Funktion

Die Operation *MARK* beendet die aktuelle Transaktion und setzt einen Konsistenzpunkt, d.h. die zugehörige BIM-Datei wird als leer definiert.

Alle Satzsperren und von LEASY verwalteten Blocksperren werden aufgehoben.

Gleichzeitig wird eine Folgetransaktion begonnen, in der alle zum Zeitpunkt der Operation *MARK* eröffneten Dateikennungen wieder eröffnet sind.

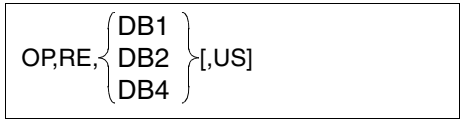
Die Dateikennungspositionen zeigen jedoch alle auf Dateianfang.

Die Operation ist im Teilnehmerbetrieb (Stapel und Dialog) und im DCAM-Teilhaberbetrieb, nicht aber im Teilhaberbetrieb mit openUTM erlaubt.

MARK hat die gleiche Wirkung wie *CLTR* mit *OPE2=T*, nur dass die Dateipositionen nicht erhalten bleiben.

OPFL Dateien eröffnen

Operanden im LEASY-Aufruf:



Funktion

Die Operation *OPFL* eröffnet die in der Dateiliste angegebenen Dateien physikalisch entsprechend dem zugehörigen OPEN-Modus (DVS-OPEN-Makro). Für den jeweiligen OPEN-Modus gilt:

– Er wird aus dem Feld *OPE-OM* des *RE*-Bereichs entnommen. Er ist dann für alle Dateien der Dateizuweisung *DB* identisch (Formate DB1/DB2).
– Er wird explizit für jede Datei angegeben (Format DB4).

Die zugehörigen OPEN-Modi sind ab Seite 188 beschrieben.

Die Operation *OPFL* ist nur zulässig, wenn für den Prozess keine **Transaktion** eröffnet ist.

Die zu eröffnenden Dateien können in **einer** *OPFL*-Operation angegeben werden oder auf **mehrere** hintereinanderfolgende *OPFL*-Operationen aufgeteilt werden. Maximal können 512 Dateien eröffnet werden (theoretische Obergrenze).

Beispiel

```
OPFL     (D1,D2) ———————————————————— Eröffnen der Dateien D1 und D2
    OPTR (D1,D2)
      .
      .
      .
    CLTR
OPFL     D3 ————————————————————————— Eröffnen der Datei D3
    OPTR (D1,D2,D3)
      .
      .
      .
    CLTR
CLFL     (ALL) ———————————————————— Schließen aller Dateien
```

Die Operation *OPFL* ist nur für den openUTM- und den DCAM-Betrieb zwingend vorgeschrieben; ansonsten ist sie wahlweise. Sie wird von openUTM in der Startphase aufgerufen (*OPFL*-Startparameter).
Im DCAM-Betrieb sind die zu eröffnenden Dateien bei Mehrtaskbetrieb in **einer** *OPFL*-Operation anzugeben. Bei openUTM-Betrieb dürfen die Dateien auf mehrere *OPFL*-Startparameter verteilt werden.

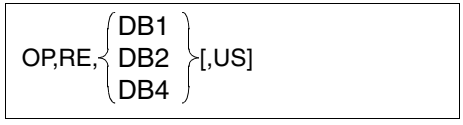
Dateien, deren Größe 32 Gigabyte übersteigt, werden von LEASY nicht bearbeitet. Ein OPFL auf eine derartige Datei wird abgewiesen.

OPTR Transaktion eröffnen oder erweitern

Bei dieser Operation sind zwei Funktionen mit unterschiedlichen Operanden im LEASY-Aufruf zu unterscheiden:

Funktion 1: Beginn einer Transaktion festlegen bzw. transaktionsorientierte Dateiliste erweitern

Operanden im LEASY-Aufruf:



Funktion

Ist zum Zeitpunkt des Aufrufs keine Transaktion des Benutzers aktiv, so wird mit *OPTR* der Beginn einer LEASY-Transaktion definiert. Dabei werden alle im Operanden *DB* des LEASY-Aufrufs angegebenen Dateikennungen mit ihren zugehörigen USAGE-Modi für die Transaktion logisch eröffnet.

Wurde die Operation *OPFL* vorher nicht durchgeführt, so werden die zu bearbeitenden Dateien und die BIM-Datei auch physikalisch eröffnet.
Der intern gewählte DVS-OPEN-Modus ist der Tabelle „Definierte USAGE-Modi für die Operation OPTR“ auf Seite 189 zu entnehmen. Dieser Vorgang wird als implizites *OPFL* bezeichnet. Wurde die Operation *OPFL* (Dateien sind bereits physikalisch eröffnet) durchgeführt, so werden die Dateikennungen nur in Tabellen transaktionsorientiert eingetragen.

Alle Dateien werden auf Dateianfang bzw. Dateiende (SAM-Dateien beim Rückwärtslesen) und Primärschlüssel positioniert.

Wird mit Before-Image-Sicherung gearbeitet, so werden gleichzeitig die an der Transaktion beteiligten Dateien in das erste Element der BIM-Datei protokolliert (siehe Abschnitt „BIM-Sicherungsverfahren“ auf Seite 53f).

Ist zum Zeitpunkt der Operation *OPTR* bereits eine Transaktion dieses Benutzers eröffnet, so wird diese Transaktion um die im Operanden *DB* angegebenen Dateikennungen erweitert. Die neu eröffneten Dateikennungen werden auf Dateianfang bzw. Dateiende (SAM-Dateien beim Rückwärtslesen) und Primärschlüssel positioniert. Diese Verwendung ist nur möglich, wenn die Dateien durch die Operation *OPFL* physikalisch eröffnet wurden.

Beispiel

```
   OPFL ((D1,4),(D2,4),(D3,3))
┌──OPTR ((D1,RETR),(D2,UPDT))
│  RNXT D1
│  OPTR (D3,EXUP) ------------ Verlängerung der transaktions-
│                              orientierten Dateiliste
│  RDIR D3
└──CLTR
```

In beiden Fällen gilt:

Eine Dateikennung kann innerhalb einer Transaktion nur einmal eröffnet werden.

Eine Datei kann innerhalb einer Transaktion mit verschiedenen Folgemerkmalen, d.h. verschiedenen Dateikennungen, mehrmals logisch eröffnet werden. In diesem Fall wird der bei Dateikennung 2 angegebene USAGE-Modus zuerst nach Tabelle „Verknüpfungsregeln für die USAGE-Modi einer logischen Datei“ auf Seite 193 auf Verträglichkeit mit den bisher zur selben Datei in derselben Transaktion angegebenen USAGE-Modi geprüft. Der sich ergebende neue Result-USAGE-Modus wird zur Prüfung auf Verträglichkeit gegen fremde, parallel laufende Transaktionen verwendet (nach Tabelle „Verträglichkeit der LEASY-USAGE-Modi“ auf Seite 192).

Beispiel

```
OPTR   ((D1/FM1,RETR),(D1/FM2,UPDT))
RNXT   D1/FM1
RDIR   D1/FM2
```

Trifft eine *OPTR*-Operation bei einer Dateikennung der angegebenen Dateiliste auf eine USAGE-Mode-Unverträglichkeit zu einer parallel ablaufenden Transaktion, so tritt die durch die **TIME*-Operation des Dienstprogramms LEASY-MAINTASK oder den Operanden *OPE-WTIME* im *RE*-Bereich eingestellte Wartezeit in Kraft.

Läuft diese Wartezeit ab, ohne dass die sperrende Transaktion beendet wurde, so erhält das Anwenderprogramm den Returncode *99ALL110*; ansonsten kann es innerhalb seiner Operation *OPTR* fortfahren.

Funktion 2: Transaktion eröffnen und Dateikennungen gemäß CI eröffnen und positionieren

Operanden im LEASY-Aufruf:

OP,RE(mit OPE1=W),CI[,US]

Funktion

Eine LEASY-Transaktion wird eröffnet und die in der Currency Information abgelegten Dateikennungen eröffnet und positioniert.

Unterschiede zur Funktion 1:

– Die Namen und die USAGE-Modi der zu eröffnenden Dateikennungen sind in der Currency Information enthalten und müssen nicht mit dem Operanden *DB* im LEASY-Aufruf angegeben werden.

– Die Dateikennungen werden nach der Eröffnung (physikalisch und/oder logisch) nicht auf Dateianfang positioniert, sondern auf die Positionen, die in der Currency Information hinterlegt sind.

Anwendung eines Wiederanlaufpunkts durch die Verwendung der Operation *OPTR* mit der Zusatzfunktion OPE1=W:

Mit der Anweisungsfolge

`CINF`	Sichern der Currency Information
`CLTR (OPE2=T)`	Beenden der Transaktion mit Transaktionsneubeginn

kann ein Wiederanlaufpunkt bezüglich des Dateizustands mit Erhaltung der Dateipositionen definiert werden. Bei diesem Wiederanlaufpunkt kann auch nach einem Systemabsturz wieder aufgesetzt werden.

Ein Wiederanlaufpunkt wird folgendermaßen realisiert:

1. Rücksetzen der Dateiinhalte bei einem Warmstart durch die LEASY-Maintask.
2. Lesen der gesicherten Currency Information im folgenden Programmlauf.
3. Durchführen der Operation *OPTR* (*OPE1=W*) mit der Currency Information.

Das Anwenderprogramm befindet sich nun bezüglich des Dateizustands (eröffnete Dateien und Dateipositionen) im gleichen Zustand wie zum Zeitpunkt des *CLTR* (*OPE2=T*). Der Zustand der Speicherbereiche im Anwenderprogramm kann durch LEASY **nicht** wiederhergestellt werden.

Für ein normales Wiederaufsetzen mit *OPTR* und Dateiliste (Dateien sind auf Dateianfang positioniert) ist dieses Vorgehen **nicht** erforderlich.

Trifft eine *OPTR*-Operation bei einer Dateikennung der angegebenen Dateiliste eine USAGE-Mode-Unverträglichkeit zu einer parallel ablaufenden Transaktion, so tritt die durch die **TIME*-Anweisung des Dienstprogramms LEASY-MAINTASK oder den Operanden *OPE-WTIME* im *RE*-Bereich eingestellte Wartezeit in Kraft.

Läuft diese Wartezeit ab, ohne dass die sperrende Transaktion beendet wurde, so erhält das Anwenderprogramm den Returncode *99ALL110*; ansonsten kann es innerhalb seiner Operation *OPTR* fortfahren.

Wird LEASY von einer DCAM-Anwendung aufgerufen, so ist bei der ersten Operation *OPTR* jeder LEASY-Transaktion das Feld *IDE* im *RE*-Bereich gelöscht zu übergeben (*X'00'*). LEASY liefert bei erfolgreicher Durchführung der Operation im Feld *IDE* die Transaktions-Identifikation zurück. Diese ist dann bei allen Operationen dieser Transaktion zu übergeben.

RDIR/RHLD Satz direkt lesen / mit Satzsperre

Operanden im LEASY-Aufruf:

```
OP,RE,DB1,AR[,FA[,SI[,KB[,KE]]]]
```

Funktion der Operation RDIR

Die Funktion *RDIR* liest einen Satz oder Block direkt in die Satzzone *AR*:

- Einen Satz einer ISAM- oder DAM-Datei über einen angegebenen Schlüssel
- Einen Satz einer SAM-Datei über die angegebene Wiedergewinnungsadresse (24- oder 31-Bit-Format)
- Einen Block einer PAM-Datei über die angegebene Blocknummer.

Über den Primärindex wird zugegriffen, wenn der Operand *SI* fehlt, wenn er keinen Inhalt (Leerzeichen) oder den Inhalt *MAINITEM* hat.

Über den Sekundärindex wird zugegriffen, wenn dieser im Operanden *SI* angegeben wurde (nur für ISAM-, PAM- und DAM-Dateien möglich). Beim Zugriff über ISAM-Sekundärschlüssel für NK-ISAM-Dateien ist im Operanden *SI* der Name des ISAM-Sekundärschlüssels anzugeben.

Werden 8 Operanden angegeben - Definition eines Intervalls (*KB*, *KE*) - so wird bei $KB < KE$ der Satz mit dem kleinsten Schlüsselwert des angegebenen Intervalls gebracht, bei $KB > KE$ der Satz mit dem größten Schlüsselwert.

Soll ein Satz mit einem ganz bestimmten Schlüssel gelesen werden, so erreicht man dies durch:

- $KB = KE$
- Angabe von 7 Operanden und Besetzung von *KB*
- Angabe von 4-6 Operanden und Versorgung des Schlüssels im *AR*-Bereich (ISAM und DAM) bzw. im *RE*-Bereich (PAM und SAM).

Zusatzfunktionen

Zusätzlich zum Lesen des Satzes erfolgt eine Positionierung innerhalb der Dateikennung auf den gefundenen Schlüssel und den verwendeten Index (Primär- oder Sekundärindex).

Werden 8 Operanden angegeben, so wird dadurch ein Schlüsselintervall definiert, innerhalb dessen mit den Operationen *RNXT/RPRI* sequenziell gelesen werden kann. *RDIR* liefert dann den ersten im Intervall vorhandenen Satz. Dies muss nicht der bei *KB* angegebene Schlüsselwert sein. Wird bei Intervallangabe mit der Operation *RDIR* kein Satz gefunden, so ist kein Intervall aktuell.

Werden weniger als 8 Operanden angegeben, so bilden Dateianfang und -ende bzw. Sekundärindexanfang und -ende die natürlichen Intervallgrenzen. Wird in diesem Fall mit *RDIR* kein Satz gefunden, so erfolgt die Positionierung wie bei einem entsprechenden *SETL*-Aufruf (siehe Operation *SETL*).

Wird der Operand *FA* mit dem Wert *MAINITEM* besetzt, so werden beim Lesen nur die Schlüsselfelder zurückgeliefert. Das bedeutet bei Zugriff über den Primärindex, dass nur die Existenz des Datensatzes überprüft und das Primärschlüsselfeld besetzt wird. Bei Zugriff über einen Sekundärindex werden nur Primär- und Sekundärschlüssel-Wert besetzt; der direkte Zugriff auf den Primärdatensatz entfällt.

Zusatzfunktion (Angaben im RE-Bereich)

OPE2=N:

Bei Zugriff über den Sekundärindex liefert LEASY im Feld *NUM* des *RE*-Bereichs die Anzahl der Primärschlüssel zum Sekundärindex-Wert. Voraussetzung ist, dass kein Schlüsselintervall angegeben wird.

Definition eines Leseintervalls

In den Operanden *KB* und *KE* sind als Primärschlüssel die ISAM-Schlüsselwerte, DAM-Satznummern, PAM-Blocknummern oder SAM-Wiedergewinnungsadressen anzugeben, oder für ISAM, DAM und PAM die Sekundärschlüssel-Werte.

Für SAM-, PAM-, DAM- und ISAM-Dateien gilt:

Die Intervallgrenzen Dateianfang bzw. Dateiende können durch die Schlüsselwerte *X'00'* bzw. *X'FF'* angegeben werden.

Bei ISAM-, DAM- und PAM-Dateien kann *KB* größer, gleich oder kleiner als *KE* gewählt werden. Bei SAM-Dateien muss bei Eröffnung zum Vorwärtslesen $KB \leq KE$, bei Eröffnung zum Rückwärtslesen $KB \geq KE$ gewählt werden.

Übergabe der Schlüsselwerte

Tabelle 15 zeigt die Möglichkeiten für die Übergabe der Schlüsselwerte in Abhängigkeit von der Dateiart und der Anzahl der Operanden.

Dateiart	**Anzahl der Operanden**	**versorgt werden**	**von LEASY zurückgeliefert werden**
ISAM PAM, DAM, SAM	8	DB1 mit Dateinamen FA mit { ALL / MAINITEM } SI mit { Sekundärindex / Leerzeichen / MAINITEM } KB } Schlüssel der Intervallgrenzen KE } (Primär- oder Sekundärschlüs- SAMPTR (RE-Bereich) bei SAM: Wiedergewinnungsadresse 4-Byte-Adresse: 2. Wort binär Null oder Blanks 8-Byte-Adresse: 2. Wort ≠ binär Null oder Blanks	FA: ALL Bei $KB<KE$ Satz mit dem kleinsten Schlüsselwert des Intervalls in AR-Bereich Bei $KB> KE$ Satz mit dem größten Schlüsselwert des Intervalls in AR-Bereich Bei $KB=KE$ Satz mit dem Schlüsselwert KB in AR-Bereich FA: MAINITEM Bei $KB<KE$ Kleinster Schlüssel des Intervalls in AR-Bereich bzw. RE-Bereich (PAM, SAM) Bei $KB> KE$ Größter Schlüssel des Intervalls in AR-Bereich bzw. RE-Bereich (PAM, SAM) Bei $KB=KE$ Schlüssel aus KB in AR-Bereich bzw. RE-Bereich (PAM, SAM)

Tabelle 15: Übergabe der Schlüsselwerte bei der Operation RDIR/RHLD (Teil 1 von 3)

Dateiart	Anzahl der Operanden	versorgt werden	von LEASY zurückgeliefert werden
ISAM PAM, DAM, SAM	7	DB1 mit Dateinamen FA mit { ALL / MAINITEM } SI mit { Sekundärindex / Leerzeichen / MAINITEM } KB mit Schlüssel (Primär- oder Sekundärschlüssel) SAMPTR (RE-Bereich) bei SAM: Wiedergewinnungsadresse 4-Byte-Adresse: 2. Wort binär Null oder Blanks 8-Byte-Adresse: 2. Wort ≠ binär Null oder Blanks	FA: ALL Satz mit dem Schlüssel aus *KB* in *AR*-Bereich FA: MAINITEM Schlüssel aus *KB* in *AR*-Bereich bzw. *RE*-Bereich (PAM, SAM)
	6	DB1 mit Dateinamen AR mit Primärschlüssel: Bei ISAM an der definierten Position. Bei DAM in den ersten 4 Byte (binär) PAMHPNR (RE-Bereich) bei PAM: FA mit { ALL / MAINITEM } SI mit { Sekundärindex / MAINITEM } bei SAM: nur Angabe MAINITEM oder Leerzeichen erlaubt SAMPTR (RE-Bereich) bei SAM: Wiedergewinnungsadresse	FA: ALL Bei ISAM und DAM: Satz mit dem im *AR*-Bereich angegebenen Primär schlüssel in *AR*-Bereich Bei PAM: Block mit dem in *PAMHPNR* angegebenen Primärschlüssel in *AR*-Bereich FA: MAINITEM Bei ISAM und DAM: Primärschlüssel in *AR*-Bereich Bei PAM: Primärschlüssel aus *PAMHPNR* in *RE*-Bereich Bei SAM: Satz mit der in *SAMPTR* angegebenen Wiedergewinnungsadresse

Tabelle 15: Übergabe der Schlüsselwerte bei der Operation RDIR/RHLD (Teil 2 von 3)

Dateiart	Anzahl der Operanden	versorgt werden	von LEASY zurückgeliefert werden
ISAM PAM, DAM, SAM	5	DB1 mit Dateinamen AR mit Primärschlüssel: Bei ISAM an der definierten Position Bei DAM in den ersten 4 Byte (binär) PAMHPNR (RE-Bereich) bei PAM: Primärschlüssel FA mit { ALL / MAINITEM } SAMPTR (RE-Bereich) bei SAM: Wiedergewinnungsadresse	FA: ALL Bei ISAM und DAM: Satz mit dem im *AR*-Bereich angegebenen Primärschlüssel in *AR*-Bereich Bei PAM: Satz mit dem in *PAMHPNR* angegebenen Primärschlüssel in *AR*-Bereich FA: MAINITEM Bei ISAM und DAM: Primärschlüssel in *AR*-Bereich Bei PAM: Primärschlüssel aus *PAMHPNR* in *RE*-Bereich Bei SAM: Satz mit der in *SAMPTR* angegebenen Wiedergewinnungsadresse
	4	DB1 mit Dateinamen AR mit Primärschlüssel: Bei ISAM an der definierten Position: Bei DAM in den ersten 4 Byte (binär) PAMHPNR (RE-Bereich) bei PAM: Primärschlüssel SAMPTR (RE-Bereich) bei SAM: Wiedergewinnungsadresse	Bei ISAM und DAM: Satz mit dem im Bereich angegebenen Primärschlüssel in *AR*-Bereich Bei PAM: Block mit dem in PAMHPNR angegebenen Primärschlüssel in *AR*-Bereich Bei SAM: Satz mit der in *SAMPTR* angegebenen Wiedergewinnungsadresse

Tabelle 15: Übergabe der Schlüsselwerte bei der Operation RDIR/RHLD (Teil 3 von 3)

Ablauf des Lesens beim Zugriff über LEASY-Sekundärschlüssel

Erfolgt der Zugriff über einen Sekundärindex, so wird als erster Schritt der zugehörige Primärschlüsselwert ermittelt und bei ISAM und DAM in den *AR*-Bereich, bei PAM in das Feld *PAMHPNR* des *RE*-Bereichs übertragen.

Gibt es zu einem Sekundärindex-Wert mehrere Duplikatsätze, so wird bei $KB \leq KE$ der kleinste, bei $KB > KE$ der größte Primärschlüsselwert genommen. Dann wird auf den Primärsatz direkt zugegriffen.

Kann der Primärsatz nicht erfolgreich gelesen werden (z.B. Sperre nicht möglich), so kann der Aufrufer dennoch den Primärschlüsselwert auswerten.

Bei multiplem Sekundärindex wird immer die erste Ausprägung (geringste Distanz zum Satzanfang) für die Positionierung ausgewertet.

Ablauf des Lesens beim Zugriff über ISAM-Sekundärschlüssel

Der Zugriff über ISAM-Sekundärschlüssel ist im Abschnitt „Sekundärindizierung“ auf Seite 46ff. beschrieben.

Funktion der Operation RHLD

Bei der Operation *RHLD* wird, neben den bei der Operation *RDIR* beschriebenen Funktionen, der gelesene Satz zusätzlich gesperrt. Dies geschieht jedoch nur bei erfolgreichem Lesen, d.h. Fehlercode=*000LL000*. Wird z.B. kein Satz gefunden, so wird auch kein Sperrelement angelegt.

Zusatzfunktionen (Angaben im RE-Bereich)

Für die Operation *RHLD* sind folgende Zusatzfunktionen möglich, die im *RE*-Bereich angefordert werden:

OPE1=S	Für die Datei wird beim Sperren ein READ-LOCK abgesetzt.
OPE1=_	Für die Datei wird beim Sperren ein WRITE-LOCK abgesetzt.

REWR Bestehenden Satz ändern

Operanden im LEASY-Aufruf:

```
OP,RE,DB1,AR[,US]
```

Funktion

Ein bereits vorhandener Satz bzw. Block wird geändert. Der von LEASY intern geführte Positionszeiger für die Dateikennung wird nicht verändert.

Wird für eine Datei ein Sperrprotokoll geführt, so muss der Satz bzw. Block innerhalb der Transaktion vorher gesperrt werden.

Geänderte Sätze bzw. Blöcke bleiben bis Transaktionsende gesperrt.

Ist Aktualisierung der Sekundärindex-Verweise angegeben, so werden die Sekundärindex-Verweise automatisch mitverwaltet.

Übergabe des Schlüsselwertes

SAM-Datei	Der zu verändernde Satz muss vorher gelesen worden sein. Unmittelbar nach dem Lesen kann er verändert werden.
ISAM/DAM-Datei	Der Satz mit dem in der Satzzone *AR* hinterlegten Primärschlüssel wird geändert.
PAM-Datei	Der Block mit dem im Feld *PAMHPNR* des *RE*-Bereichs hinterlegten Primärschlüssel wird geändert.

i Wird die Datei als Fremddatei mit SHARED-UPDATE eröffnet, gilt:

- Bei einer ISAM- und DAM-Datei muss der Satz direkt vor der Operation *REWR* mit *RHLD/RNHD/RPHD* gelesen und gesperrt worden sein.
- Bei einer PAM-Datei müssen die Blöcke vorher mit *RHLD/RNHD/RPHD/LOCK* gesperrt werden.

RNXT/RNHD Nachfolger lesen/mit Satzsperre

Operanden im LEASY-Aufruf:

OP,RE,DB1,AR[,FA]

Funktion der Operation RNXT

Der nächste Satz bzw. Block der angegebenen Dateikennungen wird - ausgehend von der für die Dateikennung aktuellen Position - sequenziell gelesen, bei SAM in Richtung Dateiende, bei ISAM, PAM oder DAM in Richtung aufsteigender Primär- oder Sekundärschlüssel-Werte.

Es wird auf den Index zugegriffen, auf den bei der letzten für diese Dateikennung durchgeführten Operation *RDIR* bzw. *RHLD* oder *SETL* positioniert wurde (Voreinstellung = Primärschlüssel).

Zusätzlich gilt das bei dieser Operation (*RDIR/RHLD/SETL*) evtl. angegebene Intervall (*KB,KE*).

Wurde bei *RDIR/RHLD/SETL* kein Intervall angegeben, so wird der Returncode *010LL003* (EOF) bei Erreichen von Dateiende bzw. Sekundärindex-Dateiende geliefert.

Wurde ein Lese**intervall** spezifiziert, so wird das Überschreiten des Intervallendes von LEASY mit dem Returncode *LC=L003* gemeldet.

Die Operation *RNXT* nach der Operation *SETL* bringt den Satz bzw. Block mit einem Schlüsselwert, der gleich oder größer dem bei der Operation *SETL* angegebenen Primär- oder Sekundärschlüssel-Wert ist.

Zusatzfunktionen

Bei SAM-Dateien wird der Wert der aktuellen Wiedergewinnungsadresse *ID1RPTR* in das Feld *SAMPTR* des *RE*-Bereichs zurückgeliefert, wobei der im Datenblock links stehende erste Datensatz die Nummer 01 innerhalb der Blocknummer erhält. Die Wiedergewinnungsadresse wird im 24- oder 31-Bit-Format zurückgeliefert (siehe hierzu Feld *SAMPTR* auf Seite 130).

Bei PAM-Dateien wird die Blocknummer des gelesenen Blocks in das Feld *PAMHPNR* zurückgeliefert.

Zusätzlich zum Lesen des Satzes bzw. Blocks wird die Position innerhalb der Dateikennung auf den neuen Satz eingestellt, jedoch nur bei erfolgreicher Durchführung (Fehlercode=*000LL000*).

Ein Überschreiten der Intervallgrenze wird in der Positionsverwaltung für die Dateikennung vermerkt.

Durch die Operandenbesetzung *FA=MAINITEM* kann der Benutzer verlangen, dass nur die Schlüsselfelder geliefert werden.
Dies bedeutet bei sequenziellem Zugriff entlang des Primärindex, dass nur der Wert des nächsten Primärschlüssels geliefert wird, die Satzzone ansonsten aber unverändert bleibt. Bei sequenziellem Zugriff entlang eines Sekundärindex werden in der Satzzone das Primärschlüsselfeld und das Sekundärschlüssel-Feld (bzw. bei geteiltem Sekundärschlüssel alle Teilfelder) besetzt.

Ablauf des Lesens

Erfolgt der Zugriff entlang eines Sekundärindex, so wird zuerst der zugehörige Primärschlüsselwert ermittelt und bei ISAM in die Satzzone *AR*, bei PAM in das Feld *PAMHPNR* des *RE*-Bereichs übertragen. Kann in der darauffolgenden Bearbeitung der Primärsatz nicht erfolgreich gelesen werden (z.B. Sperre nicht möglich), so kann der Benutzer dennoch den Primärschlüsselwert auswerten.

Gibt es zu einem Sekundärindex-Wert mehrere Duplikatsätze, dann wird zuerst der Satz bzw. Block mit dem kleinsten, dann jeweils mit dem nächstgrößeren Primärschlüsselwert geliefert.

Funktion der Operation RNHD

Bei der Operation *RNHD* wird zusätzlich der Satz bzw. Block gesperrt. Gesperrt wird nur, wenn die Operation erfolgreich durchgeführt wurde (Fehlercode=*000LL000*).

Erfolgt der Zugriff entlang des Primärindex, so wird zuerst der zugehörige Satz direkt in die Satzzone *AR* gelesen. Erst dann ist der Primärschlüsselwert bekannt, und es kann ein etwaiger Sperrversuch unternommen werden. Tritt dabei ein Fehler auf, so ist die Satzzone bereits verändert.

Zusatzfunktionen (Angaben im RE-Bereich)

Für die Operation *RNHD* sind folgende Zusatzfunktionen möglich, die im *RE*-Bereich angefordert werden:

OPE1=S	Für die Datei wird beim Sperren ein READ-LOCK abgesetzt.
OPE1=_	Für die Datei wird beim Sperren ein WRITE-LOCK abgesetzt.
OPE2=L	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so setzt LEASY den Pointer so, als wäre der Satz gelesen worden.
OPE2=_	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so wird nach der Wartezeit der Returncode (*99ALL006*) übergeben. Der Satz wird nicht gelesen, der Pointer wird nicht verändert.

RPRI/RPHD Vorgänger lesen/mit Satzsperre

Operanden im LEASY-Aufruf:

```
OP,RE,DB1,AR[,FA]
```

Funktion der Operation RPRI

Der nächste Satz der angegebenen Dateikennung wird - ausgehend von der für die Dateikennung aktuellen Position - sequenziell gelesen, bei SAM in Richtung Dateianfang, bei ISAM, PAM oder DAM in Richtung absteigender Primär- oder Sekundärschlüssel-Werte.

Es wird über den Index zugegriffen, auf den bei der letzten für diese Dateikennung durchgeführten Operation *RDIR* bzw. *RHLD* oder *SETL* positioniert wurde.

Zusätzlich gilt das bei dieser Operation (*RDIR/RHLD/SETL*) evtl. angegebene Intervall (*KB,KE*).

Wurde bei *RDIR/RHLD/SETL* kein Intervall angegeben, so wird der Returncode *010LL003* (EOF) bei Erreichen von Dateianfang bzw. Sekundärindex-Dateianfang geliefert.

Ist ein Intervall aktuell, so wird bei Überschreiten der Intervallgrenze von LEASY der Returncode *010LL003* geliefert.

Die Operation *RPRI* nach der Operation *SETL* bringt den Satz bzw. Block mit einem Schlüsselwert, der gleich oder kleiner dem bei der Operation *SETL* angegebenen Sekundär- oder Primärschlüssel-Wert (*KB*) ist.

Zusatzfunktion

Zusätzlich zum Lesen des Satzes bzw. Blocks wird für die aktuelle Dateikennung auf den neuen Satz/Block positioniert. Die Neupositionierung erfolgt jedoch nur bei erfolgreichem Lesen, d.h. Fehlercode=*000LL000*; ansonsten bleibt die alte Dateiposition unverändert.

Die restlichen Zusatzfunktionen und der Ablauf der Operation *RPRI* entspricht jeweils der Operation *RNXT*.

Funktion der Operation RPHD

Bei der Operation *RPHD* wird, zusätzlich zu den oben genannten Funktionen, der Satz bzw. Block gesperrt.
Gesperrt wird nur, wenn die Operation erfolgreich durchgeführt wurde (Fehlercode=*000LL000*).

Erfolgt der Zugriff entlang des Primärindex, so wird zuerst der zugehörige Satz direkt in die Satzzone *AR* gelesen. Erst dann ist der Primärschlüsselwert bekannt, und es kann ein etwaiger Sperrversuch unternommen werden. Tritt dabei ein Fehler auf, so ist die Satzzone bereits verändert.

Zusatzfunktionen (Angaben im RE-Bereich)

Für die Operation *RPHD* sind folgende Zusatzfunktionen möglich, die im *RE*-Bereich angefordert werden:

OPE1=S	Für die Datei wird beim Sperren ein READ-LOCK abgesetzt
OPE1=_	Für die Datei wird beim Sperren ein WRITE-LOCK abgesetzt
OPE2=L	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so setzt LEASY den Pointer so, als wäre der Satz gelesen worden.
OPE2=_	Ist der gewünschte Satz frei, so wird er in den *AR*-Bereich übernommen und gesperrt. Der Pointer steht je nach Leserichtung hinter bzw. vor dem gelesenen Satz. Ist der Satz gesperrt, so wird nach der Wartezeit der Returncode (*99ALL006*) übergeben. Der Satz wird nicht gelesen, der Pointer wird nicht verändert.

SETL Dateizeiger positionieren

Operanden im LEASY-Aufruf:

OP,RE,DB1[,AR[,FA,SI[,KB[,KE]]]]

Funktion

Mit der Operation *SETL* kann ein intern geführter Dateizeiger auf definierte Schlüssel für die angegebene Dateikennung positioniert werden. Zusätzlich wird der beim Operanden SI angegebene Index (Primär- oder Sekundärindex) und bei Angabe von 8 Operanden ein Schlüsselintervall für nachfolgende *RNXT/RNHD/RPRI/RPHD* Operationen eingestellt.

Bei multiplem Sekundärindex wird immer die 1. Ausprägung (geringste Distanz zum Satzanfang) für die Positionierung ausgewertet.

Bezüglich der Übergabe der LEASY-Schlüssel gilt:

- Bei Angabe von 8 Operanden wird durch (*KB,KE*) ein Intervall definiert; ansonsten bilden die Dateigrenzen die natürlichen Intervallgrenzen.

  Für ISAM-, PAM- und DAM-Dateien gilt:
  Die Intervallgrenzen Dateianfang bzw. -ende können durch die Schlüsselwerte *X'00...'* bzw. *X'FF...'* in der jeweils gültigen Schlüssellänge angegeben werden.

  Bei SAM-Dateien werden die Intervallgrenzen Dateianfang bzw. -ende bei 4-Byte-Adressen, wie auch bei 8-Byte-Adressen, durch die Schlüsselwerte *X'00000000'* bzw. *X'FFFFFFFF'* bestimmt.

  Bei ISAM-, PAM- und DAM-Dateien kann KB </>/= KE angegeben werden. Bei SAM-Dateien muss bei Eröffnung zum Vorwärtslesen $KB \leq KE$, bei Eröffnung zum Rückwärtslesen $KB \geq KE$ gewählt werden.

- Bei Zugriff über den **Primärschlüssel** ist die Versorgung der Operanden identisch mit der bei der Operation *RDIR/RHLD* (siehe Tabelle „Übergabe der Schlüsselwerte bei der Operation RDIR/RHLD“ auf Seite 171f).

- Bei Zugriff über einen **LEASY-Sekundärschlüssel** ist im Unterschied zur Operation *RDIR/RHLD* bei *SETL* stets der Sekundärschlüssel-Wert **und** der Primärschlüsselwert anzugeben. Damit ist eine Positionierung auf einen bestimmten Duplikatssatz möglich. Gibt es zu dem bei *SETL* angegebenen Sekundärschlüssel-Wert mehrere Duplikate, so wird bei *RNXT* der Satz bzw. Block mit einem Primärschlüsselwert ≥, bei *RPRI* der Satz bzw. Block mit einem Primärschlüsselwert ≤ dem bei *SETL* spezifizierten Primärschlüsselwert geliefert.

  Die Übergabe eines aus LEASY-Primär- und Sekundärschlüssel-Wert kombinierten Schlüsselpaares erfolgt bei 4-6 Operanden über die Satzzone *AR* und für eine PAM-Datei zusätzlich über das Feld *PAMHPNR*.

Bei Verwendung von 7 oder 8 Operanden erfolgt die Übergabe des Schlüsselpaares nur über *KB* (und *KE*). Die über *KB* und *KE* adressierten Schlüsselfelder bestehen je aus 2 Teilen:
Im linken Teil wird der Sekundärschlüssel-Wert in der Länge des aktuellen Sekundärindex versorgt. Daran anschließend muss der Primärschlüsselwert (ISAM-Primärschlüssel oder PAM-Blocknummer bzw. DAM-Satznummer in der Länge 4) versorgt werden. Damit ist bei Definition eines Leseintervalls (*KB,KE*) auch die Einschränkung auf eine Untermenge aller zu einem Sekundärschlüssel-Wert vorhandenen Duplikatssätze möglich.

- Der Zugriff über **ISAM-Sekundärschlüssel** ist auf Seite 46ff. beschrieben.

Durch die Operation *SETL* erfolgt noch keine Ein-/ Ausgabeoperation. Erst durch eine nachfolgende Operation *RNXT/RNHD* oder *RPRI/RPHD* wird ein Satz bzw. Block mit einem Schlüssel ≥ oder ≤ dem mit der Operation *SETL* angegebenen Schlüssel gelesen.

STOR Satz einfügen

Operanden im LEASY-Aufruf:

```
OP,RE,DB1,AR[,US]
```

Funktion

Der Datensatz bzw. -block wird in die Datei geschrieben, unabhängig davon, ob er bereits existiert oder nicht (im Gegensatz zu den Operationen *REWR* und *INSR*, die vorher prüfen, ob der Satz/Block schon existiert).

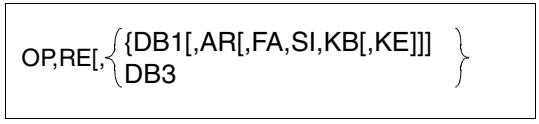
Der von LEASY intern geführte Positionszeiger für die Dateikennung wird nicht verändert.

Durch *STOR* eingebrachte Datensätze bzw. -blöcke bleiben bis Transaktionsende gesperrt.

Ist Aktualisierung der Sekundärindex-Verweise angegeben, werden Sekundärindex-Verweise mitverwaltet.

UNLK Satzsperre aufheben

Operanden im LEASY-Aufruf:



Funktion

Die Operation *UNLK* gibt bei ISAM-, PAM- und DAM-Dateien Sperrelemente frei:

– für einzelne Sätze bzw. Blöcke, die durch einen Primärschlüssel identifiziert sind.

– für Dateiabschnitte, die durch ein Primärschlüsselintervall identifiziert sind.

Bei Sperren, die über das LEASY-Sperrprotokoll im Common Memory CMMAIN verwaltet werden (die Anwendung arbeitet mit dem LEASY-Katalog) gilt Folgendes:

Im Allgemeinen können nur Sperren für Sätze bzw. Blöcke oder Dateiabschnitte aufgehoben werden, die zwar durch die Operation *LOCK* oder Leseoperationen mit Sperrwirkung (*RHLD*, *RNHD*, *RPHD*) innerhalb der Transaktion gesperrt, aber nicht verändert wurden.

Die Freigabe der Sperre von Sätzen, die in der selben Transaktion eingefügt, verändert oder gelöscht wurden, ist auch unter folgenden Voraussetzungen möglich:

– In der Operation *UNLK* wurde der Operand *OPE1='U'* im *RE*-Bereich gesetzt.

– Es wird für die betroffene Datei in der aktuellen Transaktion keine BIM-Sicherung geführt.

  Die Führung einer BIM-Sicherung wird durch folgende Angaben beeinflusst:

  – Sessionspezifisch durch den *LOG*-Operand der Maintask.

  – Dateispezifisch durch den *BIM*-Operanden des Katalogs.

  – Transaktionsspezifisch durch das Feld *OPE-LOG* des *RE*-Bereiches bei der Operation *OPTR*.

Es ist sinnvoll, angelegte Sperren, die nicht mehr benötigt werden, explizit mit der Operation *UNLK* freizugeben, um evtl. Zugriffen aus anderen Transaktionen auf den gesperrten Satz/Block die Wartezeit zu verkürzen.

Es ist jedoch nicht möglich, aus einem mit *LOCK* angegebenen Sperrintervall mittels *UNLK* nur ein Teilintervall oder einen einzelnen Schlüsselwert aus diesem Intervall freizugeben. Die bei *UNLK* und *LOCK* angegebenen Intervallgrenzen müssen übereinstimmen.

Bei Transaktionsende werden alle Sperren automatisch freigegeben.

Wird die *UNLK*-Operation auf einen modifizierten Satz angewendet, der den oben angeführten Bedingungen nicht entspricht, wird der Returncode *99AL008* geliefert.

Schlüsselwert übergeben

Tabelle 16 zeigt die Möglichkeiten für die Übergabe der Schlüsselwerte in Abhängigkeit von der Dateiart und der Anzahl der Operanden.

Dateiart	**Anzahl der Operanden**	**versorgt werden**		**freigegeben werden**
ISAM, DAM	8	DB1 SI KB KE	mit Dateinamen mit Leerzeichen oder „MAINITEM“ Primärschlüssel der Intervallgrenzen Der Inhalt von *KB* kann größer, kleiner oder gleich dem Inhalt von *KE* sein	Dateiabschnitt, der von den Primärschlüsseln *KB/KE* begrenzt wird
	7	DB1 SI KB	mit Dateinamen mit Leerzeichen oder „MAINITEM“ mit Primärschlüssel	Satz mit dem Primärschlüssel aus *KB*
	4	DB1 AR	mit Dateinamen mit Primärschlüssel an der definierten Position bei ISAM; in den ersten 4 Byte bei DAM	Satz mit dem Primärschlüssel aus *AR*-Bereich
	3	DB1 AR	mit Dateinamen wird nicht versorgt	alle gesperrten Sätze dieser Datei, die nicht geändert wurden

Tabelle 16: Übergabe der Schlüsselwerte bei der Operation UNLK (Teil 1 von 2)

Dateiart	Anzahl der Operanden	versorgt werden		freigegeben werden
PAM	8	DB1 SI KB KE	mit Dateinamen mit Leerzeichen oder „MAINITEM“ Primärschlüssel der Intervallgrenzen	Dateiabschnitt, der von den PAM-Blocknummern *KB/KE* begrenzt wird
	7	DB1 SI KB	mit Dateinamen mit Leerzeichen oder „MAINITEM“ PAM-Blocknummer	Block mit der PAM-Blocknummer aus *KB*
	3	DB1 PAMHPNR	mit Dateinamen (*RE*-Bereich) mit PAM-Blocknummer	Block mit der PAM-Blocknummer aus *PAMHPNR*
	3	DB1 PAMHPNR	mit Dateinamen (*RE*-Bereich) mit Wert 0	alle gesperrten Blöcke dieser Datei, die nicht verändert wurden
ISAM, PAM, DAM	3	DB3	mit „(ALL)“	alle gesperrten Sätze bzw. Blöcke aller an der Transaktion beteiligten Dateien, die nicht verändert wurden
	2	DB1	wird nicht versorgt	

Tabelle 16: Übergabe der Schlüsselwerte bei der Operation UNLK (Teil 2 von 2)

Wird eine Fremddatei mit SHARED-UPDATE eröffnet, so bildet LEASY Satz-/Blocksperren auf die ISAM/UPAM Sperrmechanismen ab. In diesem Fall kann durch einen *UNLK*-Aufruf maximal ein Satz/Block entsperrt werden.

Bei ISAM wird der gesperrte Satz der angegebenen Datei entsperrt. Bei PAM muss die Blocknummer explizit angegeben werden. Bei DAM muss eine Satznummer angegeben werden. Bei Transaktionsende können übriggebliebene Sperren durch LEASY nicht automatisch freigegeben werden.

Wird für eine Datei kein Sperrprotokoll geführt bzw. die Datei ohne SHARED-UPDATE eröffnet, so ist die Operation *UNLK* wirkungslos.

9.5 Tabelle aller LEASY-Operationen und ihrer Operanden

Operation	RE	OPE1	OPE2	INT	DB1/2/3/4/CI/CAT	AR	FA	SI	KB	KE	US
OPFL	x				DB1/DB2/DB4						[x]
OPTR	x	x			DB1/DB2/DB4/CI						[x]
CLFL	x				[DB1/DB2/DB3]						[x]
CLTR	x	x	x								[x]
MARK	x										[x]
BACK	x										[x]
RDIR	x		x	P+S	DB1	x	[x	[x	[x	[x]]]]	
RHLD	x	x	x	P+S	DB1	x	[x	[x	[x	[x]]]]	
SETL	x			P+S	DB1	[I+D	[x	x	[x	[x]]]]	
LOCK	x	x		P	DB1	[I+D	[I+P+D	I+P+D	I+P+D	[I+P+D]]]	
RNXT	x				DB1	x	[x]				
RNHD	x	x			DB1	x	[x]				
RPRI	x				DB1	x	[x]				
RPHD	x	x			DB1	x	[x]				
INSR	x			P	DB1	x					[x]
STOR	x			P	DB1	x					[x]
REWR	x			P	DB1	x					[x]
DLET	x			P	DB1	[I+D	[I+P+D	I+P+D	I+P+D]]		[x]
UNLK	x			P	[DB3]/[DB1	[I+D	[I+P+D	I+P+D	I+P+D	[I+P+D]]]]	
CINF	x	x	x		CI						
CATD	x				CAT						[x]

Tabelle 17: LEASY-Operationen und ihre Operanden

- x Operanden müssen angegeben werden.
- I Operanden sind bei ISAM anzugeben.
- P Operanden sind bei PAM anzugeben.
- D Operanden sind bei DAM anzugeben.
- S Operanden sind bei SAM anzugeben.
- [ ] Operanden sind wahlweise anzugeben.
- / Einer der aufgeführten Operanden ist anzugeben.

LEASY prüft, ob die erforderlichen Operanden angegeben sind. Es ist erlaubt, Operanden anzugeben, die nicht notwendig sind. Diese werden nicht weiter untersucht.

9.6 Dateien und Transaktionen eröffnen

Beim Eröffnen der Dateien mit der Operation *OPFL* können die in Tabelle 18 definierten OPEN-Modi angegeben werden.

Mit dem gewählten OPEN-Modus wird die Menge der möglichen USAGE-Modi für dieselbe Datei bei einer darauffolgenden Operation *OPTR* bestimmt.

LEASY-OPEN-Modus für die Operation OPFL

Mit dem OPEN-Modus wird der DVS-OPEN-Modus für das physikalische Eröffnen der angegebenen Datei festgelegt.

Das 1 Byte lange Kennzeichen für den LEASY-OPEN-Modus ist im Feld *OPE-OM* des *RE*-Bereichs zu hinterlegen (Formate DB1 und DB2) oder im Format DB4 der Dateizuweisung.

LEASY-OPEN-Modus	S=SAM I=ISAM P=PAM D=DAM	DVS-OPEN-Modus	bei OPTR erlaubte USAGE-Modi
1	I+P+D+S	INPUT	*PRRT, EXRT*
2	I+P+D	INPUT, SHARUPD	*RETR, PRRT, EXRT, ULRT*
3	I+P+D	INOUT	alle ISAM-, DAM- und PAM-USAGE-Modi außer *ULRT* und *ULUP*
4	I+P+D	INOUT, SHARUPD	alle ISAM-, DAM- und PAM-USAGE-Modi
5	S	REVERSE	*PRRR, EXRR*
6		(reserviert)	---
7	S	UPDATE	*EXUP*
8	S	OUTPUT	*EXLD*
9	S	EXTEND	*EXLD*
A	I+P+D	OUTIN	alle ISAM-, DAM- und PAM-USAGE-Modi außer *ULRT* und *ULUP*
B	I+P+D	OUTIN, SHARUPD	alle ISAM-, DAM- und PAM-USAGE-Modi außer *ULRT*

Tabelle 18: LEASY-OPEN-Modi

Als OPEN-Modus ist bei den Formaten DB1 und DB2 ebenfalls ein Leerzeichen erlaubt. Dieses steht für *OPEN-Modus=4* bei ISAM-, DAM- und PAM-Dateien und für *OPEN-Modus=9* bei SAM-Dateien.

Man beachte, dass in openUTM-LEASY-Programmen der *OPEN-Modus=3* nur für openUTM-Anwendungen mit einer Task möglich ist.

LEASY-USAGE-Modus für die Operation OPTR

Der USAGE-Modus legt die Zugriffsart der eigenen und die erlaubten Zugriffsarten paralleler Transaktionen fest.

Der 4 Byte lange alphabetische USAGE-Modus wird bei der Operation *OPTR* angegeben, wo er im Format DB4 zu hinterlegen ist, oder es gilt der Wert im Feld *OPE-OM* des *RE*-Bereichs.

Tabelle 19 gilt für Stamm- und Modelldateien. Fremd- und Temporärdateien werden zwar mit dem angeführten DVS-OPEN-Modus, aber stets mit *SHARED-UPDATE=NO* eröffnet, falls die Operation *OPFL* nicht angegeben wird.

USAGE-Modus	**SAM**	**ISAM PAM DAM**	**aktuelle Transaktion**	**erlaubte Zugriffe paralleler Transaktionen**	**DVS-OPEN-Modus**
RETR retrieval	-	+	lesen	lesen schreiben	INPUT SHARUPD
PRRT protected retrieval	+	+	lesen	lesen	INPUT
PRRR protected retrieval reverse	+	-	lesen rückwärts	lesen	REVERSE
EXRT exclusive retrieval	+	+	lesen	kein Zugriff	INPUT
EXRR exclusive retrieval reverse	+	-	lesen rückwärts	kein Zugriff	REVERSE
UPDT update	-	+	lesen schreiben	lesen schreiben	INOUT SHARUPD
PRUP protected update	-	+	lesen schreiben	lesen	INOUT SHARUPD
EXUP exclusive update	+	+	lesen schreiben	kein Zugriff	INOUT bzw. UPDATE
EXLD exclusive load	+	+	schreiben im Lademodus	kein Zugriff	INOUT bzw. EXTEND

Tabelle 19: Definierte USAGE-Modi für die Operation OPTR (Teil 1 von 2)

USAGE-Modus	SAM	ISAM PAM DAM	aktuelle Transaktion	erlaubte Zugriffe paralleler Transaktionen	DVS-OPEN-Modus
LOAD share load	-	+	lesen schreiben (der auf steigende Satzschlüssel wird von LEASY vergeben)	lesen schreiben	INOUT SHARUPD
PLOD protected load	-	+		lesen	INOUT SHARUPD
ELOD exclusive load	-	+		kein Zugriff	INOUT
LDUP load + update	-	+		lesen schreiben	INOUT SHARUPD
PLUP protected load +update	-	+		lesen	INOUT SHARUPD
ELUP exclusive load + update	-	+		kein Zugriff	INOUT
ULRT unlocked retrival	-	+	lesen	lesen schreiben	INPUT SHARUPD
ULUP unlocked update	-	+	lesen schreiben	lesen	INOUT SHARUPD

Tabelle 19: Definierte USAGE-Modi für die Operation OPTR (Teil 2 von 2)

Als USAGE-Modus ist bei den Formaten DB1 und DB2 ebenfalls ein Leerzeichen erlaubt. Es steht für *USAGE-Modus=UPDT* bei ISAM-, DAM- und PAM-Dateien, und für *USAGE-Modus=EXLD* bei SAM-Dateien.

Diese Werte passen zu den bei OPEN-Modus=Leerzeichen definierten Werten.

Der in Tabelle 19 angeführte DVS-OPEN-Modus kommt nicht zur Anwendung, wenn die Datei schon vorher durch die Operation *OPFL* eröffnet wurde.

Erläuterung der USAGE-Modi LOAD/PLOD/ELOD und LDUP/PLUP/ELUP

Bei SAM-Dateien ist kein schreibender Vielfachzugriff möglich (DVS-Einschränkung). Um dennoch die Möglichkeit zu bieten, dass mehrere Prozesse in eine gemeinsame fortlaufende Erfassungsdatei sequenziell schreiben können, wurden die USAGE-Modi *LOAD/PLOD/ELOD* und *LDUP/PLUP/ELUP* definiert.

Hier vergibt nicht der Anwender den Satzschlüssel, sondern LEASY. Der Schlüssel wird beim Einfügen von Sätzen (Operation *INSR*) von 1 beginnend fortlaufend hochgezählt und dem Anwender binär wie folgt zurückgeliefert:

– bei ISAM in dem in der Dateidefinition festgelegten Schlüsselfeld

– bei DAM in den ersten 4 Byte des AR-Bereichs

– bei PAM im Feld *PAMHPNR* des RE-Bereichs.

Die Schlüssellänge einer ISAM-Datei (*KEYLEN*) darf nur kleiner oder gleich 4 sein.

Der Unterschied zwischen *LOAD/PLOD/ELOD* und *LDUP/PLUP/ELUP* besteht darin, dass bei *LOAD/PLOD/ELOD* an schreibenden Operationen nur *INSR* erlaubt ist und keine Sperrelemente angelegt werden. Bei *LDUP/PLUP/ELUP* hingegen sind alle Operationen außer *STOR* erlaubt; deshalb wird bei *LDUP* und *PLUP* auch ein Sperrprotokoll geführt (siehe Tabelle „LEASY-Operationen in Abhängigkeit vom USAGE-Modus“ auf Seite 194).

Um diesen Ladevorgang mit der Schlüsselvergabe durch LEASY auch geschützt bzw. exklusiv vornehmen zu können, wurden die USAGE-Modi *PLOD/PLUP* und *ELOD/ELUP* festgelegt.

Zur Verbesserung der Performance beim Schreiben einer ISAM-Datei mit den USAGE-Modi *LOAD* und *LDUP* hat der Benutzer die Möglichkeit, einen Satz mit dem Schlüssel *X'FF...FF'* einzufügen. Dieser Satz muss mit einem USAGE-Modus geschrieben werden, bei dem LEASY nicht selbst die Schlüssel vergibt, z.B. *UPDT*. Er wird dann von LEASY bei der Schlüsselvergabe mit den USAGE-Modi *LOAD* und *LDUP* ignoriert. Dies bewirkt, dass beim Einfügen eines Satzes nicht sämtliche Indexstufen von ISAM berichtigt werden müssen.

Erläuterung der Usage Modi ULUP und ULRT

Die beiden Usage Modi *ULUP* und *ULRT* erlauben den einfachen Schreib- und mehrfachen Lesezugriff auf eine Datei ohne Führen eines Sperrprotokolls. Sie können nur miteinander, nicht aber mit anderen Usage-Modi kombiniert werden (siehe dazu auch die Tabelle „Verträglichkeit der LEASY-USAGE-Modi“ auf Seite 192).

Mögliche Kombinationen der USAGE-Modi

Die folgende Tabelle zeigt die erlaubten USAGE-Modi für Benutzer *B2*, nachdem Benutzer *B1* die Datei mit dem angegebenen USAGE-Modus eröffnet hat:

B2 → / B1 ↓	RETR	UPDT	PRRT	PRRR	PRUP	EXRT	EXRR	EXUP	LOAD	LDUP	EXLD	PLOD	ELDO	PLUP	ELUP	ULRT	ULUP
RETR	x	x	x	-	x	-	-	-	x	x	-	x	-	x	-	-	-
UPDT	x	x	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PRRT	x	-	x	x	-	-	-	-	-	-	-	-	-	-	-	-	-
PRRR	-	-	x	x	-	-	-	-	-	-	-	-	-	-	-	-	-
PRUP	x	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EXRT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EXRR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EXUP	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
LOAD	x	-	-	-	-	-	-	-	x	-	-	-	-	-	-	-	-
LDUP	x	-	-	-	-	-	-	-	-	x	-	-	-	-	-	-	-
EXLD	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PLOD	x	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ELOD	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PLUP	x	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ELUP	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ULRT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	x
ULUP	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	-

Tabelle 20: Verträglichkeit der LEASY-USAGE-Modi

Tabelle 20 gilt für Stamm- und Modelldateien. Bei Fremd- und Temporärdateien wird nicht geprüft, ob die USAGE-Modi kombiniert werden dürfen. In diesem Fall gelten die DVS-Verträglichkeiten.

Wenn Benutzer *B1* mit *OPFL* (vor *OPTR*) eröffnet und *B2* ohne *OPFL* (vor *OPTR*), dann gilt Tabelle 20 nur insoweit, als der für *B2* nach Tabelle 19 (Seite 189) gewählte DVS-OPEN-Modus mit dem für *B1* nach Tabelle 18 (Seite 188) gewählten DVS-OPEN-Modus verträglich ist.

Die folgende Tabelle zeigt die Verträglichkeiten der innerhalb einer Transaktion von einem Benutzer für dieselbe logische Datei, aber für verschiedene Folgemerkmale angegebenen USAGE-Modi.

U2 → U1 ↓	RETR	UPDT	PRRT	PRUP	EXRT	EXUP	LOAD	PLOD	ELOD	LDUP	PLUP	ELUP	EXLD	PRRR	EXRR	ULRT	ULUP
RETR	RETR	UPDT	PRRT	PRUP	EXRT	EXUP	LOAD	PLOD	ELOD	LDUP	PLUP	ELUP	-	-	-	-	-
UPDT	UPDT	UPDT	PRUP	PRUP	EXUP	EXUP	-	-	-	-	-	-	-	-	-	-	-
PRRT	PRRT	PRUP	PRRT	PRUP	EXRT	EXUP	PLOD	PLOD	ELOD	PLUP	PLUP	ELUP	-	-	-	-	-
PRUP	PRUP	PRUP	PRUP	PRUP	EXUP	EXUP	-	-	-	-	-	-	-	-	-	-	-
EXRT	EXRT	EXUP	EXRT	EXUP	EXRT	EXUP	ELOD	ELOD	ELOD	ELUP	ELUP	ELUP	-	-	-	-	-
EXUP	EXUP	EXUP	EXUP	EXUP	EXUP	EXUP	-	-	-	-	-	-	-	-	-	-	-
LOAD	LOAD	-	PLOD	-	ELOD	-	LOAD	PLOD	ELOD	PLUP	PLUP	ELUP	-	-	-	-	-
PLOD	PLOD	-	PLOD	-	ELOD	-	PLOD	PLOD	ELOD	PLUP	PLUP	ELUP	-	-	-	-	-
ELOD	ELOD	-	ELOD	-	ELOD	-	ELOD	ELOD	ELOD	ELUP	ELUP	ELUP	-	-	-	-	-
LDUP	LDUP	-	PLUP	-	ELUP	-	PLUP	PLUP	ELUP	LDUP	PLUP	ELUP	-	-	-	-	-
PLUP	PLUP	-	PLUP	-	ELUP	-	PLUP	PLUP	ELUP	PLUP	PLUP	ELUP	-	-	-	-	-
ELUP	ELUP	-	ELUP	-	ELUP	-	ELUP	ELUP	ELUP	ELUP	ELUP	ELUP	-	-	-	-	-
EXLD	-	-	-	-	-	-	-	-	-	-	-	-	EXLD	-	-	-	-
PRRR	-	-	-	-	-	-	-	-	-	-	-	-	-	PRRR	EXRR	-	-
EXRR	-	-	-	-	-	-	-	-	-	-	-	-	-	EXRR	EXRR	-	-
ULRT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ULRT	-
ULUP	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ULUP

Tabelle 21: Verknüpfungsregeln für die USAGE-Modi einer logischen Datei

Wenn Benutzer *B1* mit *OPFL* (vor *OPTR*) eröffnet und *B2* ohne *OPFL* (vor *OPTR*), dann gilt Tabelle 20 nur insoweit, als der für *B2* nach Tabelle 19 gewählte DVS-OPEN-Modus mit dem für *B1* nach Tabelle 18 gewählten DVS-OPEN-Modus verträglich ist.

Wird eine Dateikennung innerhalb einer Transaktion zuerst mit USAGE-Modus *U1* eröffnet und darauf eine Dateikennung derselben logischen Datei mit USAGE-Modus *U2*, so ergibt sich für die logische Datei der neue gemeinsame USAGE-Modus *U12* nach Tabelle 21.

U12 wird gegen die USAGE-Modus-Angaben paralleler Transaktionen anderer Benutzer nach Tabelle 21 geprüft und gilt dann als neuer Result-USAGE-Modus *U1* für die Datei.

Man beachte, dass die Tabelle 21 symmetrisch zur Diagonalen ist; dementsprechend ist die Reihenfolge der Verknüpfung ohne Bedeutung. Bezüglich der Verknüpfungen über Tabelle 21 ist es auch gleichwertig, ob die betrachtete Dateikennung inklusive ihrem USAGE-Modus innerhalb derselben *OPTR*-Operation oder in einer zusätzlichen *OPTR*-Operation innerhalb derselben Transaktion angegeben wird.

9.7 Tabelle der LEASY-Operationen in Abhängigkeit vom USAGE-Modus

Die folgende Tabelle zeigt die erlaubten Operationen in Abhängigkeit vom USAGE-Modus der Dateikennung und DVS-Dateityp (*ACCESS-METHOD*).

Operation → **USAGE-Modus ↓**	RDIR	RHLD	SETL	RNXT	RNHD	RPRI	RPHD	INSR	STOR	REWR	DLET	LOCK	UNLK
RETR + PRRT + EXRT (ISAM + PAM + DAM)	x	x	x	x	x	x	x	-	-	-	-	x	x
UPDT + PRUP + EXUP+ ULUP (ISAM + PAM + DAM)	x	x	x	x	x	x	x	x	x	x	x	x	x
LDUP + PLUP + ELUP (ISAM + PAM + DAM)	x	x	x	x	x	x	x	x	-	x	x	x	x
LOAD + PLOD + ELOD (ISAM + PAM + DAM)	x	x	x	x	x	x	x	x	-	-	-	x	x
ULRT (ISAM + PAM + DAM)	x	-	x	x	-	x	-	-	-	-	-	-	-
EXLD (ISAM + PAM + DAM)	-	-	-	-	-	-	-	x	-	-	-	x	x
PRRT + EXRT (SAM)	x	x	x	x	x	-	-	-	-	-	-	-	x
PRRR + EXRR (SAM)	x	x	x	-	-	x	x	-	-	-	-	-	x
EXUP (SAM)	x	x	x	x	x	-	-	-	-	x	-	-	x
EXLD (SAM)	-	-	-	-	-	-	-	x	-	-	-	-	x

Tabelle 22: LEASY-Operationen in Abhängigkeit vom USAGE-Modus

Obwohl die Entsperrfunktionen überall und die Sperrfunktionen bei allen ISAM-, DAM- und PAM-USAGE-Modi erlaubt sind, wird ein Sperrprotokoll nur bei den USAGE-Modi *UPDT*, *PRUP*, *RETR*, *LDUP* und *PLUP* geführt.

10 COBOL-Schnittstelle

Unentbehrliche Voraussetzung zum Verständnis dieses Abschnitts ist die Kenntnis von Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121ff. Die Gliederung beider Kapitel stimmt überein; zudem wird das Nachschlagen durch Verweise aus diesem Kapitel zu den entsprechenden Abschnitten im Übersichtskapitel erleichtert.

10.1 LEASY aufrufen

Das Anwenderprogramm ruft LEASY mittels *CALL*-Aufrufen über Unterprogrammverknüpfung auf, wie sie in höheren Programmiersprachen üblich sind.

Die Standardregister dafür sind:

R1	Adresse der Operandenliste
R14	Rücksprungadresse
R15	Sprungziel

Die Operandendefinition ist - mit wenigen Ausnahmen - mit der von KLDS identisch.

Aufruf aus dem COBOL-Hauptprogramm

```
CALL "LEASY" USING OP,RE,{CAT|DB|CI},AR,FA,SI,KB,KE,US.
```

10.2 COBOL-Schnittstelle versorgen

In der COBOL-Anweisung sind die je nach LEASY-Operation erforderlichen Operanden anzugeben.

Operationscode OP

Mit *OP* wird die von LEASY auszuführende Operation bestimmt.

```
01   OP   PIC    X(4).              Ü   Operationscode
```

Die zulässigen Operationscodes können mit dem COPY-Element *LEASYKON* aus der Bibliothek *SYSLIB.LEASY.062* in das Anwenderprogramm kopiert werden. Sie sind im COPY-Element *LEASYKON* mit folgenden Feldnamen vereinbart (im Programm sind diese Feldnamen und nicht die Konstanten zu verwenden).

```
01        OP-CODES.

          05 OP-CATD PIC X(4) VALUE     „CATD“.
          05 OP-OPFL PIC X(4) VALUE     „OPFL“.
          05 OP-CLFL PIC X(4) VALUE     „CLFL“.
          05 OP-OPTR PIC X(4) VALUE     „OPTR“
          05 OP-CLTR PIC X(4) VALUE     „CLTR“.
          05 OP-MARK PIC X(4) VALUE     „MARK“.
          05 OP-BACK PIC X(4) VALUE     „BACK“.
          05 OP-RDIR PIC X(4) VALUE     „RDIR“.
          05 OP-RNXT PIC X(4) VALUE     „RNXT“.
          05 OP-RPRI PIC X(4) VALUE     „RPRI“.
          05 OP-RHLD PIC X(4) VALUE     „RHLD“.
          05 OP-RNHD PIC X(4) VALUE     „RNHD“.
          05 OP-RPHD PIC X(4) VALUE     „RPHD“.
          05 OP-INSR PIC X(4) VALUE     „INSR“.
          05 OP-STOR PIC X(4) VALUE     „STOR“.
          05 OP-REWR PIC X(4) VALUE     „REWR“.
          05 OP-DLET PIC X(4) VALUE     „DLET“.
          05 OP-SETL PIC X(4) VALUE     „SETL“.
          05 OP-LOCK PIC X(4) VALUE     „LOCK“.
          05 OP-UNLK PIC X(4) VALUE     „UNLK“.
          05 OP-CINF PIC X(4) VALUE     „CINF“.
```

Die LEASY-Operationen sind im Abschnitt „LEASY-Operationen“ auf Seite 150ff in alphabetischer Reihenfolge beschrieben.

Verständigungsbereich RE

Im Verständigungsbereich werden Informationen vom Anwender an LEASY übergeben (mit *Ü* markiert) oder von LEASY zurückgeliefert (mit *R* markiert).

Der Verständigungsbereich *RE* kann mit den COPY-Elementen *LEASYPAR*, *LEASYRE* oder *LEASYREL* aus der Bibliothek *SYSLIB.LEASY.062* in das Anwenderprogramm kopiert werden.

Der Teil der COPY-Elemente, der den Verständigungsbereich betrifft, ist wie folgt strukturiert:

```
01   RE.                                          Verständigungsbereich
                                                  (80 Bytes)
  05 RE-K.                                        kompatibler Teil des RE
                                                  (48 Bytes)
     10 RC-CODE.
        15 RC-CC    PIC X(3).   R                 kompatibler Returncode
        15 RC-KZ    PIC X.      R                 Systemkennzeichen "L"
        15 RC-LC    PIC X(4).   R                 LEASY-Returncode
     10 PASS        PIC X(8).                     reserviert für Kennwort
     10 OPE.                                      Operationsergänzungen
        15 OPE-STX PIC X.      Ü                  STXIT-Routinen-Steuerung
        15 OPE-OM  PIC X.      Ü                  OPEN-Modus/USAGE-Modus
        15 OPE-LOG PIC X.      Ü                  BIM-Loggingsteuerung, nur
                                                  bei OPTR
        15 FILLER  PIC X(5).                      unbenutzt

     10 INT.                                      interner Ordnungsbegriff
        15 SAMPTR  PIC X(4).   Ü/R                 SAM-WGA (24-Bit-Format)
        15 PAMHPNR REDEFINES SAMPTR PIC 9(8) COMP.
                               Ü/R                PAM-Blocknummer
        15 SAMNUM  PIC X(4).   Ü/R                 SAM-WGA (31-Bit-Format)
        15 FILLER  REDEFINES SAMNUM PIC X(4).

     10 NUM         PIC 9(8).   R                 Anzahl der Primärsätze
     10 IDE         PIC X(8).   Ü/R               Identifikationsfeld für
                                                  DCAM
  05 RE-LEASY-EXT.                                LEASY-Erweiterung des
                                                  RE (32 Bytes)
     10 REOP        PIC X(4).   R                 letzter Operationscode
     10 REDB        PIC X(16). R                  letzter Dateiname
                                                  (+SI-Name)
     10 L-OPT       PIC X.      Ü                 Versionskennzeichen, muss
                                                  "1" enthalten
     10 OPE1        PIC X.      Ü                 ⎧ OPTR/CLTR/LOCK/RDIR/
                                                  ⎨ RHLD/RNHD/RPHD/CINF
     10 OPE2        PIC X.      Ü                 ⎩ Operationsergänzungen
     10 OPE-WTIME   PIC 9(3).   Ü                 Wartezeit für Sperren
     10 RC-LCE      PIC X(5).   R                 LEASY-Returncode-Ergänzung
     10 U-PROT      PIC X.      Ü                 AIM-Protokollierungs-
                                                  kennzeichen
```

Ü vom Anwender zu übergeben

R vom System zurückgeliefert

Dateizuweisung DB

Mit diesem Operanden werden die zu bearbeitenden Dateien bzw. Dateikennungen zugewiesen.

Je nach Anzahl und Verwendung gibt es verschiedene Formate, um die Dateien/Dateikennungen zu spezifizieren.

Format DB1

Dieses Format wird verwendet, wenn nur **eine** Datei bzw. Dateikennung bearbeitet werden soll.

Format bei OPFL

```
01        DB       PIC X(8).      Ü            logischer Dateiname
```

Format bei OPTR und allen Lese- und Schreiboperationen

```
01        DB       PIC X(12).     Ü            Name einer Dateikennung
```

Der OPEN-Modus bzw. USAGE-Modus wird aus *OPE-OM* (*RE*-Bereich) entnommen (ungleich *X'FF'*).

Beispiel für DB1-Formate

Bei *OPFL*:

```
01       DB1-OPFL.
  05     DB-MITABDAT  PIC X(8) VALUE "MITABDAT".
  05     DB-PROTDAT   PIC X(8) VALUE "PROTDAT".
```

Bei *OPTR*:

```
01       DB1-OPTR.
  05     DB-MITABDAT  PIC X(12) VALUE "MITABDAT/ABC".
  05     DB-PROTDAT   PIC X(12) VALUE "PROTDAT".
```

Format DB2

Dieses Format erlaubt, eine variable Anzahl logischer Dateien bzw. Dateikennungen anzugeben.

Format bei OPFL

```
01       DBLISTE    PIC X(m) VALUE "(datei,...)".
```

Format bei OPTR

```
01       DBLISTE    PIC X(m) VALUE "(dateikennung,...)".
```

i Im Klammerausdruck dürfen keine Leerzeichen angegeben werden.

Der gemeinsame OPEN- bzw. USAGE-Modus wird aus *OPE-OM* (*RE*-Bereich) entnommen (ungleich *X'FF'*).

Beispiel für DB2-Formate

Bei *OPFL*:

```
01       DB2-OPFL.
  05     DBLISTE    PIC X(18) VALUE "(MITABDAT,PROTDAT)".
```

Bei *OPTR*:

```
01       DB2-OPTR.
  05     DBLISTE    PIC X(22) VALUE "(MITABDAT/ABC,PROTDAT)".
```

Format DB3

Dieses Format kann nur bei den Operationen *CLFL* und *UNLK* verwendet werden. Mit *ALL* werden alle zugewiesenen Dateien angesprochen.

```
01        DB        PIC X(5)  VALUE "(ALL)".
oder
01        DB        PIC X(12) VALUE "ALL".
```

Format DB4

Dieses Format erlaubt, für jede angesprochene Datei bzw. Dateikennung einen *eigenen* OPEN- bzw. USAGE-Modus festzulegen.

Format bei OPFL

für **eine** Datei *datei*

```
01        DB        PIC X(m) VALUE "(datei,mod)".
```

für **mehrere** Dateien *datei1* bis *datein*.

```
01        DBLISTE   PIC X(m) VALUE "((datei1,mod),...)".
```

datei	logischer Dateiname ≤ 8 Zeichen
mod	OPEN-Modus (1 Byte)
m	Länge des Format DB4

Format bei OPTR

für **eine** Dateikennung

```
01        DB        PIC X(m) VALUE "(dateikennung,mod)"
```

für **mehrere** Dateikennungen

```
01        DB        PIC X(m) VALUE "((dateikennung,mod),...)"
```

dateikennung	datei [/fm]	≤	12 Byte
fm	Folgemerkmal	≤	3 Byte
mod	USAGE-Modus	=	4 Byte
m	Länge des Formats DB4		

Innerhalb des Klammerausdrucks dürfen keine Leerzeichen enthalten sein.

Das Feld *OPE-OM* des *RE*-Bereichs muss mit *X'FF'* besetzt sein. Dies ist das Kennzeichen für DB4-Angabe von OPEN-Modus bzw. USAGE-Modus im Operanden *DB*.

Die definierten OPEN- und USAGE-Modi sind im Abschnitt „Dateien und Transaktionen eröffnen“ auf Seite 188f. beschrieben.

Beispiel für DB4-Formate

Bei *OPFL*:

```
01        DB4-OPFL.
  05      DATEI      PIC X(9)   VALUE "(DATEI,4)".
  05      DATEILISTE PIC X(34)  VALUE "((DATEI1,4),(DATEI2,1),(DATEI3,2))".
```

Bei *OPTR*:

```
01        DB4-OPTR.
  05      DATEI      PIC X(16)  VALUE "(DATEI/FM1,RETR)".
  05      DATEILISTE PIC X(48)  VALUE "((DATEI/FM1,RETR),(DATEI/FM2,UPDT),
                                       (DATEI,EXUP))".
```

Currency Information CI

Dieser Operand definiert einen Bereich variabler Länge mit Currency Information. Er beginnt mit einem 4 Byte langen Satzlängenfeld.

```
01        CI.                                    Currency Informationen
  05      CI-SLF PIC S9(4) COMP.  Ü/R            Längenfeld
  05      CI-SLR PIC S9(4) COMP.  R              Längenfeld
  05      CI-INF PIC X(m).        R              Informationsfeld der Länge m.
```

– Bei der Operation *CINF* mit *OPE1=_* hat der Anwender vor dem Aufruf das Längenfeld *CI-SLF* mit der geschätzten Länge seines Informationsfeldes zu besetzen: *CI-SLF=4+m*.
Er erhält als Rückmeldung in *CI-SLF* die tatsächliche Länge und in *CI-INF* die Currency Information. Ist keine Transaktion offen, so wird *CI-SLF=0* geliefert.

– Bei der Operation *CINF* mit *OPE1=F* hat der Anwender vor dem Aufruf das Längenfeld *CI-SLF* mit der geschätzten Länge seines Informationsfeldes (*CI-SLF=4+m*) zu besetzen. Bei der Operation *CINF* mit *OPE1=F* und *OPE2=S* muss er zusätzlich vor dem Aufruf in *CI-INF* den 8 Zeichen langen logischen Dateinamen der gewünschten Datei hinterlegen. Erfüllt keine Datei die Anforderungen, so wird *CI-SLF=0* geliefert.

Kataloginformation CAT

Dieser Operand definiert einen Bereich für die Übergabe der Kataloginformation.

```
01        CAT.                                  Katalogangaben
  05      CATNAME PIC X(24).     Ü              Name des LEASY-Katalogs
  05      ZUSATZ  PIC X(20).     Ü              Zusatzangabe für Modell-
                                                dateien
```

– In *CATNAME* ist der Name des LEASY-Katalogs *[:catid:][$userid.]dateikatalog* anzugeben.

– *CATNAME* und *ZUSATZ* müssen rechts mit Leerzeichen aufgefüllt sein.

– Unterstützung von MPVS

  Will sich eine LEASY-Anwendung an den CMMAIN eines LEASY-Katalogs anschließen, der nicht auf dem Pubset der Kennung, unter der das Anwenderprogramm gestartet wurde, eingerichtet ist, so ist in *CATNAME* eine Katalogkennung (*:catid:*) für das Pubset mit dem LEASY-Katalog anzugeben.

– Einsatz im Mehrrechnersystem

  Befindet sich der LEASY-Katalog auf einem fremden Rechner, so ist die Katalogkennung dieses Rechners als Bestandteil des Katalognamens mit anzugeben:

  [:catid:][$userid.]dateikatalog

Ein-Ausgabebereich AR

Dieser Operand definiert einen Ein-Ausgabebereich für Lese- und Schreiboperationen (Satzzone). Die Satzzone muss bei Schreiboperationen die Länge des ganzen Satzes umfassen. Bei Leseoperationen kann auf Schlüsselfelder begrenzt werden.

```
01        AR        PIC X(n).     Ü/R          Satzzone der Länge n
```

Bei variablem Satzformat wird in der Satzzone das Satzlängenfeld bei Leseoperationen mitgeliefert. Bei Schreiboperationen muss es der Anwender versorgen.

Beispiele

SAM-Datei, feste Satzlänge

```
01        AR        PIC X(20).
```

Ausgabe in Druckdatei

```
01        AR.
  05      VORSCHUB PIC X.             Das Vorschubsteuerzeichen muss
  05      SATZ     PIC X(20).         selbst besetzt werden
```

Datei mit variabler Satzlänge

```
01        AR.
  05      LAENGE   PIC S9(4) COMP VALUE 104.
  05      FILLER   PIC XX VALUE SPACE.
  05      SATZ     PIC X(100).
```

ISAM-Datei mit variabler Satzlänge und Sekundärschlüsseln

```
01        AR.
  05      SL       PIC S9(4) COMP VALUE 150.
  05      FILLER   PIC XX.
  05      DATEN1   PIC X(12).
  05      PRIMKEY  PIC X(4).
  05      SIKEY1   PIC X(20).
  05      SIKEY2   PIC X(10).
  05      DATEN2   PIC X(100).
```

Bei ISAM-Dateien ist der Schlüssel (bzw. die Teilschlüssel) stets an der Stelle und in der Länge anzugeben, wie im DVS-Katalog (bzw. im LEASY-Katalog) vermerkt. Eine Überprüfung kann nicht durchgeführt werden.

PAM-Datei mit 2 Sekundärschlüsseln

```
01        AR.
  05      DATEN1   PIC X(10).
  05      SIKEY1   PIC X(10).
  05      DATEN2   PIC X(5).
  05      SIKEY2   PIC X(15).
  05      DATEN3   PIC X(2008).
```

Bei PAM-Dateien ist der Primärschlüssel (=PAM-Blocknummer) im Feld *PAMHPNR* des *RE*-Bereichs zu versorgen; evtl. SI-Schlüssel sind in der Satzzone vorzusehen und zu besetzen.

DAM-Datei mit 2 Sekundärschlüsseln

```
01        AR.
  05      PRIMKEY  PIC S9(8) COMP.
  05      DATEN.
    10    DATEN1   PIC X(10).
    10    SIKEY1   PIC X(5).
    10    DATEN2   PIC X(15).
    10    SIKEY2   PIC X(8).
    10    DATEN3   PIC X(32).
```

Bei DAM-Dateien ist der Primärschlüssel (relative Satznummer des Satzes in der Datei) in den ersten 4 Byte des *AR*-Bereichs **vor** dem Datensatz zu versorgen.

Feldauswahl FA

Dieser Operand legt fest, ob der ganze Datensatz in die Satzzone *AR* zurückgeliefert wird oder nur die Schlüsselwerte.

Der Operand *FA* ist aus Kompatibilitätsgründen zu KLDS anzugeben, wenn der 6. Operand der *CALL*-Anweisung *SI* angegeben wird.

Format1

Dieses Format legt fest, dass der ganze Datensatz in die Satzzone *AR* zurückgeliefert wird.

```
01        FA       PIC  X(5) VALUE "(ALL)".
oder
01        FA       PIC  X(12) VALUE "ALL".
```

Format2

Dieses Format legt fest, dass nur Schlüsselwerte in die Satzzone *AR* zurückgeliefert werden.

```
01        FA       PIC  X(8) VALUE "MAINITEM".
```

Sekundärindex SI

Dieser Operand definiert den Namen eines Sekundärindex, über den zugegriffen werden soll.

```
01        SI       PIC X(8).      Ü            Name des Sekundärindex
```

Der Name ist 8 Zeichen lang. Besteht er aus der Zeichenfolge *MAINITEM* oder aus Leerzeichen, so wird über den Primärschlüssel zugegriffen.

Schlüsselanfang KB und Schlüsselende KE

Mit den Operanden *KB* und *KE* kann ein Schlüsselintervall für den über den Operanden *SI* spezifizierten Index definiert werden. Der Inhalt von *KB* kann größer, kleiner oder gleich dem Inhalt von *KE* sein.

```
01        KB        PIC X(m).          Ü          Schlüsselwert der
                                                  Länge m
01        KE        PIC X(m).          Ü          Schlüsselwert der
                                                  Länge m
```

Beispiele

Zugriff über den ISAM-Primärschlüssel der Länge 5

```
01        KB        PIC X(5) VALUE "A".
01        KE        PIC X(5) VALUE "BZZZZ".
```

Kleinster Schlüssel: *Abbbb* mit *b*=Leerzeichen auf Grund der COBOL-Konvention für die PIC-Klausel.

Zugriff auf eine PAM-Datei

```
01        KB        PIC 9(8) COMP VALUE 1.
01        KE        PIC 9(8) COMP VALUE 5.
```

Das Intervall besteht aus den PAM-Blocknummern von 1-5.

Zugriff auf eine SAM-Datei

```
01        KB        PIC 9(8) COMP VALUE 257.
01        KE        PIC 9(8) COMP VALUE 511.
```

Das Intervall besteht aus den Wiedergewinnungsadressen *X'00000101'* bis *X'000001FF'*.
(Erster bis letzter Satz des 1. Blocks der Datei, bei 4-Byte-Wiedergewinnungsadressen)

Zugriff auf eine DAM-Datei

```
01        KB        PIC 9(8) COMP VALUE 1.
01        KE        PIC 9(8) COMP VALUE 5.
```

Das Intervall besteht aus den relativen DAM-Satznummern von 1-5.

Zugriff über einen Sekundärschlüssel der Länge 20

```
01        KB        PIC X(20) VALUE "ADAM".
01        KE        PIC X(20) VALUE "BERTA".
```

Zugriff über einen aus 3 Teilschlüsseln zusammengesetzten Sekundärschlüssel

```
01        KB.
          05 KB-TEIL1 PIC X(2)  VALUE "AA".
          05 KB-TEIL2 PIC X(5)  VALUE LOW-VALUE.
          05 KB-TEIL3 PIC X(10) VALUE LOW-VALUE.

01        KE.
          05 KE-TEIL1 PIC X(2)  VALUE "EZ".
          05 KE-TEIL2 PIC X(5)  VALUE HIGH-VALUE.
          05 KE-TEIL3 PIC X(10) VALUE HIGH-VALUE.
```

USER-Bereich US

Dieser Operand definiert einen USER-Bereich für die USER-Information.

```
01        US.                                        Satzzone der Laenge n
          05 LAENGE   PIC S9(4)     COMP VALUE n.
          05 FILLER   PIC XX        VALUE SPACE.
          05 SATZ     PIC X(n-4).
```

10.3 LEASY-Operationen

Die LEASY-Operationen lassen sich in 4 Gruppen einteilen:

Organisationsoperationen		CATD, OPFL, CLFL
Transaktionsoperationen		OPTR, CLTR, MARK, BACK
Operationen auf Dateiebene	Leseoperationen	RDIR, RNXT, RPRI
	Leseoperationen mit Sperrnebenwirkung	RHLD, RNHD, RPHD
	Positionierungsoperation	SETL
	Schreiboperationen	INSR, STOR, REWR, DLET
	Sperroperationen	LOCK, UNLK
Operation zur Gewinnung einer Currency Information		CINF

Tabelle 23: Die LEASY-Operationen

Diese Operationen sind im Folgenden in alphabetischer Reihenfolge beschrieben.

BACK Rollback durchführen

Die Operation *BACK* führt einen Rollback durch. Die aktuelle Transaktion wird mit der BIM-Datei rückgängig gemacht. Satzsperren werden freigegeben und ein Konsistenzpunkt gesetzt. Siehe auch „BACK Rollback durchführen“ auf Seite 151.

```
CALL "LEASY" USING  OP,RE[,US].
```

CATD LEASY-Katalog ansprechen

Die Operation *CATD* weist den LEASY-Katalog zu, der mit dem Dienstprogramm LEASY-CATALOG erstellt wurde. Siehe auch „CATD LEASY-Katalog ansprechen“ auf Seite 152.

```
CALL "LEASY" USING OP,RE,CAT[,US].
```

CINF Currency Information übergeben

Die Operation *CINF* liefert im *CI*-Bereich die gesamte Currency Information. Sie besteht aus der Liste aller in der Transaktion eröffneten Dateikennungen und ihrer aktuellen Dateizeiger oder aus Informationen über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes. Siehe auch „CINF Currency Information übergeben“ auf Seite 154.

```
CALL "LEASY" USING OP,RE,CI.
```

Im *RE*-Bereich können folgende Zusatzfunktionen angefordert werden:

OPE1=_	Currency Information über alle in der Transaktion eröffneten Dateikennungen.
OPE1=F	Currency Information über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes.
OPE2={ – / C }	Currency Information (Typ1) über alle im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes.
OPE2=O	Currency Information (Typ1) über alle mit OPFL eröffneten Dateien.
OPE2=T	Currency Information (Typ1) über alle an der Transaktion beteiligten Dateien.
OPE2=S	Currency Information (Typ2) über die in CI spezifizierte Datei.
OPE2=W	Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden.

CLFL Dateien schließen

Die Operation *CLFL* schließt die angegebenen Dateien. Diese müssen in vorangehenden Operationen *OPFL* eröffnet worden sein. Siehe auch „CLFL Dateien schließen“ auf Seite 156.

```
CALL "LEASY" USING OP,RE[,{DB1 / DB2 / DB3}][,US].
```

Der Operand *DB3* kann aus Gründen der Kompatibilität zur Schnittstelle KLDS angegeben werden; er ist aber ohne jede zusätzliche Wirkung.

CLTR Transaktion schließen

Die Operation *CLTR* beendet die Transaktion und setzt einen Konsistenzpunkt. Siehe auch „CLTR Transaktion schließen“ auf Seite 157.

```
CALL "LEASY" USING OP,RE[,US].
```

Im *RE*-Bereich können folgende Zusatzfunktionen angefordert werden:

OPE1=R	Rücksetzen der Transaktion.
OPE2=T	Beenden der Transaktion und Eröffnen einer Folgetransaktion.

DLET Bestehenden Satz löschen

Die Operation *DLET* löscht einen Satz aus einer ISAM- oder DAM-Datei bzw. einen Block aus einer PAM-Datei. Siehe auch „DLET Bestehenden Satz löschen“ auf Seite 158.

```
CALL "LEASY" USING OP,RE,DB1[,AR[,FA,SI,KB]][,US].
```

Abhängig von der Anzahl der Operanden und der Dateiart stehen folgende Möglichkeiten zur Verfügung:

1. ISAM/DAM/PAM, 7 Operanden

   Der Schlüsselwert des zu löschenden Satzes ist in *KB* zu hinterlegen.

2. ISAM/DAM 4 Operanden

   Der Schlüsselwert des zu löschenden Satzes wird an den definierten Stellen im *AR*-Bereich übergeben

3. ISAM/DAM/PAM, 3 Operanden

   – *DLET* löscht bei ISAM- und PAM-Dateien ohne explizite Übergabe des Schlüsselwertes den Satz, der zuletzt über dieselbe Dateikennung erfolgreich gelesen wurde

   – Bei PAM-Dateien ist das Feld *PAMHPNR* im *RE*-Bereich zu versorgen. *DLET* löscht den Block mit der dort angegebenen Blocknummer. Wurde das Feld *PAMHPNR* auf Null gesetzt, löscht *DLET* den Block, der zuletzt über dieselbe Dateikennung erfolgreich gelesen wurde.

INSR Noch nicht bestehenden Satz einfügen

Die Operation *INSR* fügt einen neuen Satz bzw. Block in die angegebene Datei ein. Siehe auch „INSR Noch nicht bestehenden Satz einfügen“ auf Seite 159.

```
CALL "LEASY" USING OP,RE,DB1,AR[,US].
```

Im *AR*-Bereich ist der gesamte Satz bzw. Block zu übergeben.

Bei PAM-Dateien ist das Feld *PAMHPNR* im *RE*-Bereich zu versorgen.

LOCK Satzsperre setzen

Die Operation *LOCK* legt Sperrelemente an für einzelne Sätze bzw. Blöcke oder Dateiabschnitte von ISAM-, DAM- oder PAM-Dateien. Siehe auch „LOCK Satzsperre setzen“ auf Seite 160.

```
CALL "LEASY" USING OP,RE,DB1,[,AR[,FA,SI,KB[,KE]]].
```

Abhängig von der Anzahl der Operanden und der Dateiart stehen folgende Möglichkeiten für die Schlüsselübergabe zur Verfügung:

ISAM/DAM/PAM, 8 Operanden

Die Schlüsselwerte des zu sperrenden Intervalls sind in *KB* und *KE* zu hinterlegen.

ISAM/DAM/PAM, 7 Operanden

Der Schlüsselwert des zu sperrenden Satzes bzw. Blockes ist in *KB* zu hinterlegen.

ISAM/DAM, 4 Operanden

Der Schlüsselwert des zu sperrenden Satzes ist an der definierten Stelle im *AR*-Bereich zu übergeben.

PAM, 3 Operanden

Der Schlüsselwert des zu sperrenden Blocks ist im Feld *PAMHPNR* zu hinterlegen.

Im *RE*-Bereich können folgende Zusatzfunktionen angefordert werden:

OPE1=S	Anlegen eines READ-LOCK
OPE1=_	Anlegen eines WRITE-LOCK

MARK Sicherungspunkt erzeugen

Die Operation *MARK* beendet die aktuelle Transaktion und setzt einen Konsistenzpunkt. Siehe auch „MARK Sicherungspunkt erzeugen“ auf Seite 163.

```
CALL "LEASY" USING OP,RE[,US].
```

OPFL Dateien eröffnen

Die Operation *OPFL* eröffnet die in der Dateiliste angegebenen Dateien entsprechend dem zugehörigen OPEN-Modus. Siehe auch „OPFL Dateien eröffnen“ auf Seite 164.

```
                                 ⎧DB1⎫
CALL "LEASY" USING OP,RE,        ⎨DB2⎬[,US].
                                 ⎩DB4⎭
```

OPTR Transaktion eröffnen oder erweitern

Die Operationen *OPTR* eröffnet bzw. erweitert eine Transaktion (siehe auch „OPTR Transaktion eröffnen oder erweitern“ auf Seite 165). Bei dieser Operation sind zwei Funktionen mit unterschiedlicher Operandenübergabe zu unterscheiden.

Funktion 1: Beginn einer Transaktion festlegen bzw. transaktionsorientierte Dateiliste erweitern (siehe Seite 165).

```
                                 ⎧DB1⎫
CALL "LEASY" USING OP,RE,        ⎨DB2⎬[,US].
                                 ⎩DB4⎭
```

Funktion 2: Transaktion eröffnen und Dateikennungen gemäß CI eröffnen und positionieren (siehe Seite 167).

```
CALL "LEASY" USING OP,RE (mit OPE1=W),CI[,US].
```

Das Feld *OPE1* im *RE*-Bereich muss *"W"* enthalten.

Der *CI*-Bereich muss mit der zuvor gesicherten Currency Information versorgt sein.

RDIR/RHLD Satz direkt lesen / mit Satzsperre

Die Operation *RDIR* liest einen Satz oder Block. Die Operation *RHLD* liest einen Satz oder Block und sperrt ihn. Siehe auch Seite 169.

```
CALL "LEASY" USING OP,RE,DB1,AR[,FA[,SI[,KB[,KE]]]].
```

Über den **Primärindex** wird zugegriffen, wenn der Operand *SI* fehlt, wenn er keinen Inhalt (Leerzeichen) oder den Inhalt *MAINITEM* hat.

Über den **Sekundärindex** wird zugegriffen, wenn dieser im Operanden *SI* angegeben wurde (nur für ISAM-, DAM- und PAM-Dateien möglich).

Werden 8 Operanden angegeben - Definition eines Intervalls (*KB,KE*) - so wird bei $KB < KE$ der Satz mit dem kleinsten Schlüsselwert des angegebenen Intervalls glesen, bei $KB > KE$ der Satz mit dem größten Schlüsselwert.

Soll ein Satz mit einem ganz bestimmten Schlüssel gelesen werden, so gibt es dafür drei Möglichkeiten:

- *KB=KE*
- Angabe von 7 Operanden und Besetzung von *KB*
- Angabe von 4-6 Operanden und Versorgung des Schlüssels im *AR*-Bereich (ISAM und DAM) bzw. im *RE*-Bereich (PAM und SAM).

Übergabe eines einzelnen Schlüsselwertes

- Bei Verwendung von 7 Operanden erfolgt die Übergabe des Schlüsselwerts (ISAM-Primärschlüssel, PAM-Blocknummer, SAM-Wiedergewinnungsadresse, DAM-Satznummer bzw. Sekundärschlüssel) über den Operanden *KB*.
- Bei Verwendung von 4-6 Operanden ist ein Sekundärschlüssel-Wert oder ISAM-Primärschlüsselwert vor dem Zugriff im Ein-Ausgabebereich *AR* an den definierten Positionen, eine DAM-Satznummer in den ersten 4 Byte zu hinterlegen. Eine PAM-Blocknummer ist im Feld *PAMHPNR* des *RE*-Bereichs anzugeben, eine SAM-Wiedergewinnungsadresse im 24-Bit-Format im Feld *SAMPTR* und eine SAM-Wiedergewinnungsadresse im 31-Bit-Format in den Feldern *SAMPTR* und *SAMNUM*.

Wird über den Sekundärindex zugegriffen und ist nur ein einzelner Schlüsselwert angegeben, kann im *RE*-Bereich folgende Zusatzinformation angefordert werden:

OPE2=N Anzahl der Primärschlüssel zu einem Sekundärschlüssel-Wert wird im Feld *NUM* des *RE*-Bereichs übergeben

Für die Operation *RHLD* können im *RE*-Bereich folgende Zusatzfunktionen angefordert werden:

OPE1=S Anlegen eines READ-LOCK

OPE1=_ Anlegen eines WRITE-LOCK

REWR Bestehenden Satz ändern

Die Operation *REWR* ändert einen bestehenden Satz oder Block. Siehe auch „REWR Bestehenden Satz ändern“ auf Seite 175.

```
CALL "LEASY" USING OP,RE,DB1,AR[,US].
```

Übergabe des Schlüsselwertes

SAM-Datei	Der zu verändernde Satz muss vorher gelesen worden sein. Unmittelbar nach dem Lesen kann er verändert werden.
ISAM/DAM-Datei	Der Satz mit dem in der Satzzone *AR* hinterlegten Primärschlüssel wird geändert.
PAM-Datei	Der Block mit dem im Feld *PAMHPNR* des *RE*-Bereichs hinterlegten Primärschlüssel wird geändert.

Wird die Datei als Fremddatei mit SHARED-UPDATE eröffnet, gilt:

- Bei einer ISAM- und DAM-Datei muss der Satz direkt vor der Operation *REWR* mit *RHLD/RNHD/RPHD* gelesen und gesperrt worden sein.
- Bei einer PAM-Datei müssen die Blöcke vorher mit *RHLD/RNHD/RPHD/LOCK* gesperrt werden.

RNXT/RNHD Nachfolger lesen / mit Satzsperre

Die Operation *RNXT* liest sequenziell den nächsten Satz oder Block der angegebenen Datei. Die Operation *RNHD* liest sequenziell den nächsten Satz oder Block der angegebenen Datei und sperrt ihn. Siehe auch „RNXT/RNHD Nachfolger lesen/mit Satzsperre“ auf Seite 176.

Bei SAM-Dateien wird in Richtung Dateiende, bei ISAM-, DAM- oder PAM-Dateien in aufsteigender Richtung des Primär- oder Sekundärschlüssels gelesen.

```
CALL "LEASY" USING OP,RE,DB1,AR[,FA].
```

Bei SAM-Dateien wird der Wert der aktuellen Wiedergewinnungsadresse im 24-Bit-Format im Feld *SAMPTR* und im 31-Bit-Format in den Feldern *SAMPTR* und *SAMNUM* des *RE*-Bereichs zurückgeliefert, bei PAM-Dateien die Blocknummer im Feld *PAMHPNR* des *RE*-Bereichs, bei DAM-Dateien die relative Satznummer in den ersten 4 Byte des *AR*-Bereichs.

Soll der gesamte Satz bzw. Block gelesen werden, ist der Operand *FA* mit *ALL* oder *(ALL)* zu besetzen oder es dürfen nur 4 Operanden angegeben werden.

Wird *FA* mit *MAINITEM* besetzt, werden nur die Schlüsselfelder geliefert

Für die Operation *RNHD* können im *RE*-Bereich folgende Zusatzfunktionen angefordert werden:

OPE1=S	Anlegen eines READ-LOCK
OPE1=_	Anlegen eines WRITE-LOCK
OPE2=L	Gesperrte Sätze werden überlesen
OPE2=_	Gesperrte Sätze werden nicht überlesen

RPRI/RPHD Vorgänger lesen / mit Satzsperre

Die Operation *RPRI* liest sequenziell den nächsten Satz oder Block der angegebenen Datei. Die Operation *RPHD* liest sequenziell den nächsten Satz oder Block der angegebenen Datei und sperrt ihn. Bei SAM-Dateien wird in Richtung Dateianfang, bei ISAM-, DAM- und PAM-Dateien in absteigender Richtung des Primär- oder Sekundärschlüssels gelesen.

Siehe auch „RPRI/RPHD Vorgänger lesen/mit Satzsperre“ auf Seite 179.

```
CALL "LEASY" USING OP,RE,DB1,AR[,FA].
```

Bei SAM-Dateien wird der Wert der aktuellen Wiedergewinnungsadresse im 24-Bit-Format im Feld *SAMPTR* und im 31-Bit-Format in den Feldern *SAMPTR* und *SAMNUM* des *RE*-Bereichs zurückgeliefert, bei PAM-Dateien die Blocknummer im Feld *PAMHPNR* des *RE*-Bereichs, bei DAM-Dateien die relative Satznummer in den ersten 4 Byte des *AR*-Bereichs.

Soll der gesamte Satz bzw. Block gelesen werden, ist der Operand *FA* mit *ALL* oder *(ALL)* zu besetzen oder es dürfen nur 4 Operanden angegeben werden.

Wird *FA* mit *MAINITEM* besetzt, werden nur die Schlüsselfelder geliefert.

Für die Operation *RPHD* können im *RE*-Bereich folgende Zusatzfunktionen angefordert werden:

OPE1=S	Anlegen eines READ-LOCK
OPE1=_	Anlegen eines WRITE-LOCK
OPE2=L	Gesperrte Sätze werden überlesen
OPE2=_	Gesperrte Sätze werden nicht überlesen

SETL Dateizeiger positionieren

Die Operation *SETL* positioniert einen intern geführten Dateizeiger auf einen angegebenen Satz oder Block. Siehe auch „SETL Dateizeiger positionieren“ auf Seite 181.

```
CALL "LEASY" USING OP,RE,DB1[,AR[,FA,SI[,KB[,KE]]]].
```

SETL mit 7 oder 8 Operanden:

– Beim Zugriff über den Primärschlüssel oder über den ISAM-Sekundärschlüssel ist die Versorgung von *KB/KE* mit den Schlüsselwerten analog zu der Operation *RDIR/RHLD*.

– Ist die Übergabe eines aus Primär- und LEASY-Sekundärschlüsseln kombinierten Schlüsselpaares notwendig, ist auf den speziellen Aufbau von *KB/KE* zu achten (siehe Seite 181f).

SETL mit 4 bis 6 Operanden:

– Die Schlüsselwerte sind in der Satzzone *AR* zu übergeben.

– Bei PAM-Dateien ist zusätzlich das Feld *PAMHPNR* im *RE*-Bereich zu versorgen; bei SAM-Dateien das Feld *SAMPTR* des *RE*-Bereichs (24-Bit-Format) bzw. die Felder *SAMPTR* und *SAMNUM* des *RE*-Bereichs (31-Bit-Format).

STOR Satz einfügen

Die Operation *STOR* fügt einen Satz oder Block in die angegebene Datei ein. Siehe auch „STOR Satz einfügen“ auf Seite 183.

```
CALL "LEASY" USING OP,RE,DB1,AR[,US].
```

Der gesamte Satz oder Block ist in der Satzzone *AR* zu übergeben.

Bei PAM-Dateien ist das Feld *PAMHPNR* zu versorgen.

Bei DAM-Dateien ist der Primärschlüssel in den ersten 4 Byte des *AR*-Bereichs zu übergeben.

UNLK Satzsperre aufheben

Die Operation *UNLK* hebt Satzsperren auf. Siehe auch „UNLK Satzsperre aufheben“ auf Seite 184.

```
                                 ⎧DB1[,AR[,FA,SI,KB[,KE]]]⎫
CALL "LEASY" USING OP,RE,[,⎨                              ⎬].
                                 ⎩DB3                      ⎭
```

Die Übergabe der Schlüsselwerte erfolgt analog zu der Operation *LOCK*.

Der 3. Operand *DB* ist mit dem Dateinamen bzw. der Dateikennung im Format DB1 oder mit *ALL* im Format DB3 zu versorgen.

Das Entsperren erfolgt abhängig von der Operationcode-Ergänzung *OPE1*:

OPE1 = ’_’ Es können nur Sperren für Sätze oder Blöcke aufgehoben werden, die innerhalb der Transaktion gesperrt wurden, aber nicht verändert wurden.

OPE1 = ’U’ Auch modifizierte Sätze oder Blöcke können freigegeben werden. Die Voraussetzungen hierfür sind auf Seite 184ff genannt.

11 Assembler-Schnittstelle

Zur Unterstützung der Assembler-Programmierung stellt LEASY Makros zur Verfügung.

Diese unterteilen sich in Definitionsmakros und Aktionsmakros:

– **Definitionsmakros** haben die Funktion, Datenbereiche zu definieren. Sie erzeugen eine Reihe von DS- und DC-Anweisungen und keinen Operationsteil.

– **Aktionsmakros** haben die Funktion, Handlungen durchzuführen (z.B. lesen, schreiben). Sie erzeugen einen Operations- und einen Datenteil.

Die von den Makros erzeugte Schnittstelle ist mit der beschriebenen COBOL-Schnittstelle identisch.

Unentbehrliche Voraussetzung zum Verständnis dieses Abschnitts ist die Kenntnis von Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121ff. Die Gliederung beider Kapitel stimmt überein; zudem wird das Nachschlagen durch Verweise aus diesem Kapitel zu den entsprechenden Abschnitten im Übersichtskapitel erleichtert.

11.1 Operanden der LEASY-Makros

Als Operanden der LEASY-Makros werden Stellungsoperanden und Schlüsselwortoperanden angegeben. Für alle Makros können symbolische Namen verwendet werden.

Stellungsoperanden

Stellungsoperanden in den Aktions- und Definitionsmakros sind symbolische Adressen von Datenbereichen, Mehrzweckregister und Direktoperanden. Im Einzelnen gilt:

symb Symbolische Adresse eines Datenbereichs.
In den Aktionsmakros liegt die Adresse entweder in einer adressierbaren DSECT oder innerhalb der aktuellen Basisadressierung. In den Definitionsmakros muss sie als A-Konstante darstellbar sein

(r) Nummer oder symbolischer Name eines Mehrzweckregisters (vom Benutzer mit EQU zugewiesen), in dem die Adresse des Datenbereichs steht.

string Direktoperand in Form einer Zeichenkette.
Bei Aktionsmakros ist *string* in Hochkommata ('*string*') anzugeben. Bei Definitionsmakros ist *string* ohne Hochkommata anzugeben. Die Längenbegrenzungen des BS2000-Assemblers sind zu beachten.
Wird der Definitionsmakro innerhalb einer DSECT aufgerufen oder selbst als DSECT generiert, so können Direktoperanden nicht in die Generierung eingetragen werden. Der Makro wird in diesem Fall ordnungsgemäß generiert, jedoch wird - falls der Operand *MF=D* angegeben ist - mit einer *MNOTE* des Gewichtes *0* auf diesen Zustand hingewiesen.

Bei den Aktionsmakros und einem Definitionsmakro (*LEA@CALL* mit *MF*-Operand) können - je nach Makro - zwischen 1 und 9 Stellungsoperanden angegeben werden. Sie bezeichnen die Datenbereiche, die über die entsprechenden Definitionsmakros definiert werden und für die Übergabe der zugehörigen Daten von bzw. an LEASY verwendet werden. Sie entsprechen den gleichnamigen LEASY-Schnittstellenoperanden mit den Bezeichnungen:

op	Operationscode
re	Verständigungsbereich
db	Dateizuweisung
ar	Ein-Ausgabebereich
fa	Feldauswahl
si	Sekundärindex
kb	Schlüsselanfang
ke	Schlüsselende
us	USER-Information

Für alle Operandenformen sind die Angaben *symb* und (*r*) zulässig. Für die Operanden *op*, *db*, *fa* und *si* ist auch die Form '*string*' erlaubt. Wird Registerschreibweise verwendet, so können die Mehrzweckregister 2 bis 13 angegeben werden. Hierbei ist darauf zu achten, dass das höchstwertige Bit des Registers nicht gesetzt ist, da sonst die nachfolgenden Stellungsoperanden nicht ausgewertet werden.

Schlüsselwortoperanden

Schlüsselwortoperanden bestehen aus einem Schlüsselwort und einem Operandenwert. Dem Schlüsselwort wird mit einem Gleichheitszeichen der Wert zugewiesen. Es darf angegeben werden:

PAR=symb	Eine symbolische Adresse wird zugewiesen.
PAR=(r)	Ein Mehrzweckregister wird zugewiesen.
PAR=string	Eine Zeichenkette ohne Hochkommata wird angegeben.
PAR=wert	Eine Dezimalzahl wird angegeben. Die Grenzen sind den einzelnen Makrobeschreibungen zu entnehmen.
PAR=X'hexwert'	Sedezimalziffern werden angegeben.

Operand MF

Bei allen Definitionsmakros (Ausnahme: *LEA@OPS*) und allen Aktionsmakros (Ausnahmen: *LEA@PARC*, *LEA@TOLR*) können die Schlüsselwortoperanden *MF* und *PRE* angegeben werden.

Zulässiges Format bei Definitionsmakros

[,MF= $\left\{\begin{matrix}\underline{\text{L}}\\ \text{D}\end{matrix}\right\}$][,PRE=präfix]

MF=<u>L</u>	Der Bereich wird als normaler Datenbereich generiert. (Standardwert) In dieser Form kann der Definitionsmakro selbst innerhalb einer DSECT oder CSECT aufgerufen werden.
MF=D	Der zu erzeugende Datenbereich wird als DSECT generiert. Hinter diesem Makroaufruf folgende *DS*- und *DC*-Anweisungen werden der generierten DSECT zugeordnet.
PRE=präfix	Anzugeben ist ein einzelner Buchstabe. Alle symbolischen Namen des erzeugten Bereichs erhalten das angegebene Präfix. Standardwert *L*. Ist dem Definitionsmakro ein symbolischer Name vorangestellt und soll eine DSECT generiert werden (*MF=D*), so wird der Name nur zur Benennung der DSECT verwendet. Ist kein symbolischer Name angegeben, so erhält die DSECT den Namen `präfix@@bereichDS`

Beispiel

Adresse	Makro	PRE=	DSECT-Name
-	LEA@OP	B	B@@OPDS
-	LEA@RE	-	L@@REDS
ABCD	LEA@RE	-	ABCD

Zulässiges Format bei Aktionsmakros

$$[,MF = \left\{ \begin{array}{l} \underline{L} \\ D[,PRE=präfix] \\ (E,adresse1) \end{array} \right\}]$$

MF=L Eine Operandenliste der angegebenen Stellungsoperanden (z.B. *re, db*) wird im Makro erzeugt.

Die Operandenangabe im Format (*r*) als Register ist hier nicht zulässig.

MF=D[,PRE=präfix] Für die Operandenliste wird eine DSECT erzeugt. Sie ist für alle LEASY-Aufrufe verwendbar.

Mit *PRE*= wird das Präfix der symbolischen Namen festgelegt.

MF=(E,adresse1) *adresse1* ist die Adresse einer Operandenliste. Im Makro selbst wird nur der Befehlsteil generiert. Sie wird angegeben als symbolische Adresse (*symb*) oder als Register ((*r*))

Die Operandenliste wurde mit einem Makro (z.B. *LEA@CALL*) mit dem Operanden *MF=L* erzeugt.

Wird *MF=(E,adresse1)* und mindestens 1 Stellungsoperand angegeben, so wird die Operandenliste über den Stellungsoperanden korrigiert. Dies gilt auch für Operanden, die für den jeweiligen Makro nicht relevant sind (z.B. Angabe von *KB*, *KE* bei *LEA@RNXT*).

Wird *MF=(E,adresse1)* nicht angegeben, so ist der vom Aktionsmakro generierte Code nicht wiederverwendbar.

Operand SAVE

Zusätzlich kann bei den Aktionsmakros folgender Operand angegeben werden:

```
[,SAVE=adresse2]
```

Die Adresse kann als symbolische Adresse (*symb*) oder über ein Register ((*r*)) angegeben werden. Der definierte Bereich dient dazu, den LEASY-Verständigungsbereich (*re*) zu sichern, falls in diesem Korrekturen durchgeführt werden. Er muss daher 80 Bytes umfassen. Dieser Bereich wird als Zwischenbereich bezeichnet.

Operanden P und T

Im Verständigungsbereich (*re*) können einzelne Felder als Permanentkorrekturen oder als Temporärkorrekturen über Schlüsseloperanden verändert werden. Die Schlüsselworte sind die Namen der Bereiche mit dem Präfix *P* für Permanentkorrekturen oder *T* für Temporärkorrekturen.

Mit **permanenten** Korrekturen werden die angesprochenen Bereiche verändert. Dadurch gelten die Veränderungen auch für nachfolgende Aktionsmakroaufrufe, sofern derselbe Verständigungsbereich (*re*) verwendet wird.

Temporäre Korrekturen führen die Änderungen nur im Zwischenbereich durch. Sie gelten also nur für den aktuellen Makro. Der Verständigungsbereich wird hierzu in den Zwischenbereich kopiert. Ist kein *SAVE*-Operand angegeben, so wird der Zwischenbereich innerhalb des Aktionsmakroaufrufs angelegt.

Ist eine Korrektur sowohl als Permanent- als auch als Temporärkorrektur angegeben, so wird zuerst die permanente Korrektur im eigentlichen Operandenbereich durchgeführt. Dann wird dieser in den Zwischenbereich kopiert und anschließend die temporäre Korrektur durchgeführt.

Wird eine Operandenliste zur Verfügung gestellt (*MF=(E,adresse1)*), so wirken Permanentkorrekturen auf den über *adresse1* spezifizierten Verständigungsbereich.

Sind zusätzlich Temporärkorrekturen angegeben, so wird der Inhalt des Verständigungsbereichs (*re*) in einen Zwischenbereich übertragen, die Felder dieses Bereichs korrigiert und die Adresse des Zwischenbereichs permanent in die Operandenliste eingetragen. Dadurch kann der Benutzer nach dem LEASY-Aufrufmakro auf die Felder des gerade verwendeten *RE*-Bereichs zugreifen.

Sind Temporärkorrekturen angegeben und wird kein Zwischenbereich (*SAVE*-Operand) zur Verfügung gestellt, so ist der durch Aktionsmakros generierte Code nicht reentrant.

Operanden ERRCODE und ERRADDR

Bei den Aktionsmakros kann man zu tolerierende und nicht zu tolerierende Fehlercodes unterscheiden. Dazu dienen die beiden Schlüsselwortoperanden:

[,ERRCODE = { (fehlercode,fehlercode,...) / adresse3 },ERRADDR = adresse4]
[,ERRADDR=adresse4]

ERRCODE = (fehlercode,...)

Eine beliebige Anzahl von LEASY-Fehlercodes wird angegeben in der Form 1 bis 8 Zeichen langer Strings. Sie stellen eine Liste der zu tolerierenden Fehlercodes dar, die für den jeweiligen Makroaufruf zulässig sind. Der Code für gültige Durchführung muss nicht extra angegeben werden.

Ist ein 1, 2 oder 4 Zeichen langer Fehlercode bei *ERRCODE* angegeben, so wird im *RE*-Bereich ab der Adresse *L@@RCL*, in den anderen Fällen (max. 8 Zeichen) ab der Adresse *L@@RCCC* verglichen.

Es sind die Längenbegrenzungen des BS2000-Assemblers zu beachten.

ERRCODE=adresse3 Angegeben wird eine symbolische Adresse (*symb*). Der Aktionsmakro verzweigt nach dem LEASY-Aufruf an diese Adresse. Dort kann eine Fehlercodeunterscheidung erfolgen, wie sie z.B. durch den Makro *LEA@TOLR* angeboten wird. Der Mechanismus ist der gleiche wie bei der Angabe von Fehlercodes im Aktionsmakro; nur kann auf diese Weise für mehrere Aktionsmakros dieselbe Fehlerbehandlung erfolgen.

ERRADDR=adresse4 An diese Adresse (*symb* oder (*r*)) wird verzweigt, wenn ein anderer Fehlercode auftritt als bei *ERRCODE* angegeben.

Wird *ERRADDR*, nicht aber *ERRCODE* angegeben, so wird bei jedem Fehlercode (außer bei gültiger Durchführung) zur *adresse4* verzweigt.
ERRCODE darf nicht alleine angegeben werden.

Zum Zeitpunkt des Sprungs in die Returncode-Analyse-Routine (*adresse3*) enthält Register 15 die Adresse der Fehlerroutine (*adresse4*). Register 14 zeigt hinter das fehlerauslösende Aktionsmakro und Register 1 ist Basisregister des aktuellen *RE*-Bereichs

mit dem Return-Code. So kann auch bei temporären Korrekturen ohne Angabe des *SAVE*-Operanden der Returncode analysiert werden.

Es ist dafür gesorgt, dass nach Durchlaufen der Fehlerroutine über das Mehrzweckregister 14 hinter den fehlerauslösenden Aktionsmakro zurückgesprungen werden kann. Dies gilt auch, falls die Fehlerunterscheidung in einem eigenen Codeteil liegt und über *ERRCODE=adresse3* angesprungen wird. Für Registersicherung und -wiederherstellung in der Fehlerroutine hat der Benutzer zu sorgen.

Symbolische Namen

Vor allen Definitions- und Aktionsmakros können symbolische Namen angegeben werden. Sie dürfen maximal 7 Zeichen lang sein.

Bei den Definitionsmakros wird dieser Name als DSECT-Name verwendet, falls *MF=D* angegeben wurde. Ist keine Operandenangabe *MF=D* vorhanden, so wird ein Adresspegel - hinter einer evtl. Ausrichtung - auf das 1. adressierbare Datenelement gesetzt.

Wird keine DSECT erzeugt, so wird der angegebene symbolische Name nur beim Makro *LEA@AR* zur Bildung der Namen innerhalb des erzeugten Bereichs verwendet.

Bei Aktionsmakros wird der Name mit dem Befehl

```
Name DS 0H
```

vor den generierten Code-Teil gesetzt. Er kann als Sprungziel verwendet werden.

Die Namen der Datenelemente der Definitionsmakros werden von den Makros vergeben.

Operandenfehler, die die Generierung eines Makros (oder eines Untermakros, wie z.B. für Fehlerbehandlung verwendet) verhindern, werden mit *MNOTE* des Gewichtes *1* abgewiesen. Wurden Feldinhalte beim Aufbau einer DSECT angegeben, so weist eine *MNOTE* mit Gewicht *0* auf diesen Zustand hin.

11.2 Registerkonventionen

Die Aktionsmakros verwenden die Mehrzweckregister 0, 1, 14 und 15. Registersicherung wird für diese Register nicht betrieben, d.h. der Benutzer darf nicht erwarten, dass der Inhalt dieser Register über Makroaufrufe hinweg unverändert erhalten bleibt.

Die Mehrzweckregister 2 bis 13 werden vom LEASY-Laufzeitsystem gesichert und stehen dem Anwender nach jedem Aktionsmakroaufruf wieder mit dem ursprünglichen Inhalt zur Verfügung.

Gleitpunktregister werden nicht verändert.

Bei Aktionsmakros kann der Sprung zu einer Fehlerroutine programmiert werden (*ERRCODE=adresse3*). Aus dieser Fehlerroutine kann über Mehrzweckregister 14 hinter den fehlerauslösenden Aktionsmakroaufruf zurückgesprungen werden. Für die Registersicherung und -wiederherstellung innerhalb der Fehlerroutine ist der Benutzer verantwortlich.

11.3 Definitionsmakros

Übersicht über die Definitionsmakros:

Makro	**Bedeutung**
LEA@AR	Ein-Ausgabebereich
LEA@CALL	Operandenliste
LEA@CAT	Kataloginformation
LEA@CI	Currency Information
LEA@DB	Dateizuweisung (DB2/DB4)
LEA@DB1	Dateizuweisung (DB1)
LEA@FA	Feldauswahl
LEA@OP	Operationscode
LEA@OPS	Operationscode/Konstantenliste
LEA@RE	Verständigungsbereich
LEA@SI	Sekundärindex
LEA@US	USER-Information

Nachfolgend sind die Definitionsmakros in alphabetischer Reihenfolge beschrieben.

In den Überschriftszeilen der Einzelbeschreibung der Makros ist jeweils der Bereich mitangegeben, der durch den Makro erzeugt wird.

LEA@AR Ein-Ausgabebereich

Dieser Makro generiert einen Ein-Ausgabebereich für Lese- und Schreiboperationen. Dieser Bereich muss die Länge eines Satzes umfassen und wird auf Halbwortgrenze ausgerichtet. Er wird auch als Satzzone *AR* bezeichnet.

Die Operanden *MF* und *PRE* sind erlaubt (siehe Seite 224ff).

Name	Operation	Operanden
name	LEA@AR	[LEN=länge] [,FOR={V / F / D}]

name — wird bei *MF=L* auf die Adresse des Datenbereichs gelegt. Bei *MF=D* ist *name* der Name der DSECT.
name wird auch zur Bildung des Feldnamens verwendet (siehe unten).

i — Zur Bildung der Namen der Bereichsfelder wird der beim Makro angebbare Name herangezogen. Ist beim Makro kein Name angegeben, so werden die Feldnamen statt mit *name* mit *präfix@@AR* gebildet, wobei *PRE=präfix* ausgewertet wird (z.B. *Q@@ARD* für den Datenbereich, wenn *PRE=Q* angegeben ist).

LEN=länge — Der Datenbereich wird in der Gesamtlänge *länge* erzeugt.
Ist dieser Operand nicht angegeben, so wird nur ein Adresspegel (mit *DS 0H*) erzeugt (bei Angabe von *FOR=V* hinter einem Standardsatzlängenfeld).

FOR=V — Der Datenbereich erhält ein 4 Byte langes Standardsatzlängenfeld (Standardwert).
Der Name des Satzlängenfeldes lautet *nameS*, der Name des Datenbereichs *nameD*.

name wird bei *MF=L* auf die Adresse des Satzlängenfeldes gelegt; bei *MF=D* ist *name* der Name der DSECT.

=F — Der Datenbereich wird ohne Satzlängenfeld erzeugt.

Der Name des Datenbereichs lautet *nameD*.

name wird bei *MF=L* auf die Adresse des Datenbereichs gelegt; bei *MF=D* ist *name* der Name der DSECT.

=D Dem Datenbereich wird ein 4 Byte langes Satznummernfeld vorangestellt (DAM-Format). Die Länge des Satznummernfeldes ist nicht in der Gesamtlänge enthalten.
Der Name des Satznummernfeldes lautet *nameS*, der Name des Datenbereichs *nameD*.

name wird bei *MF=L* auf die Adresse des Satznummernfeldes gelegt; bei *MF=D* ist *name* der Name der DSECT.

Beispiel

```
  STAMM    LEA@AR LEN=256
1          LEA@@BP TYP=L,PRE=L,NAM=STAMM,GRU=GEN,ID=AR
1 STAMM    DS    0H
1 STAMMS   DC    Y(256)
1          DC    CL2' '
1 STAMMD   DS    CL(256-4)
  *
           LEA@AR LEN=256,PRE=Q
1          LEA@@BP TYP=L,PRE=Q,NAM=,GRU=GEN,ID=AR
1 Q@@AR    DS    0H
1 Q@@ARS   DC    Y(256)
1          DC    CL2' '
1 Q@@ARD   DS    CL(256-4)
```

LEA@CALL Operandenliste

Dieser Makro generiert eine Operandenliste für die Verwendung in Aktionsmakros. Die Operandenliste wird für alle 9 Operanden erzeugt. Für diese Form des *LEA@CALL*-Makros muss der *MF*-Operand angegeben werden (*MF=L* bzw. *D*), der Operand *PRE* ist erlaubt (siehe Seite 224ff).

Name	Operation	Operanden
`name`	`LEA@CALL`	`[[op],[re],[db],[ar],[fa],[si],[kb],[ke],[us]]`

op *symb* Adresse des Operationscodebereichs.

'*string*' Name der Operation.

re *symb* Adresse des Verständigungsbereichs.

db *symb* Adresse der Dateiliste bzw. Dateikennung.

'*string*' Logischer Dateiname bzw. Dateikennung in den Formaten DB1/DB2/DB4.

ar *symb* Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindex.

kb *symb* Adresse des *KB*-Bereichs (Schlüsselanfang).

ke *symb* Adresse des *KE*-Bereichs (Schlüsselende).

us *symb* Adresse des *US*-Bereichs (Benutzerbereich).

Sind Stellungsoperanden angegeben, so werden ihre Adressen in die entsprechenden Adressfelder eingetragen und die letzte verwendete Adresse mit dem Kennzeichen letzter Operand (*X'80'* im höchsten Byte) markiert. Die Operandenform *symb* (für alle Operanden) und *string* (für *op, db, fa, si*) sind zulässig (die Adressen müssen als A-Konstante bzw. Sternadressen darstellbar sein).

Für nicht angegebene Stellungsoperanden werden Adresskonstanten mit der Adresse 0 generiert.

Beispiel

```
      ADDRLIST LEA@CALL L@@OP,
   -
     L@@RE,                                                  -
     L@@DB1,                                                 -
     L@@AR,                                                  -
     L@@FA,                                                  -
     L@@SI,                                                  -
     LKB01,                                                  -
     LKE01,                                                  -
     MF=L
    1        LEA@@BP GRU=GEN,NAM=ADDRLIST,TYP=E,PRE=L
    1      LEA@@AL L@@OP,L@@RE,L@@DB1,L@@AR,L@@FA,L@@SI,LKB01,LKE01,PRE=L
    2 ADDRLIST DS    OF
    2         DC    A(L@@OP+X'00000000')
    2         DC    A(L@@RE+X'00000000')
    2         DC    A(L@@DB1+X'00000000')
    2         DC    A(L@@AR+X'00000000')
    2         DC    A(L@@FA+X'00000000')
    2         DC    A(L@@SI+X'00000000')
    2         DC    A(LKB01+X'00000000')
    2         DC    A(LKE01+X'80000000')
```

LEA@CAT Kataloginformation

Dieser Makro erzeugt einen Datenbereich für die Übergabe der Kataloginformation. Der Bereich ist 44 Zeichen lang. Davon sind 24 für den Katalognamen und 20 für einen Zusatz (bei Exemplaren von Modelldateien) belegt.

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
`name`	`LEA@CAT`	`[katalog]` `[,katalog-zusatz]`

Der von *LEA@CAT* definierte Bereich wird nur im LEASY-Aufruf *CATD* (Makro *LEA@CATD*) benützt.

katalog — Name des LEASY-Katalogs in der Form

```
[:catid:][$userid.]dateikatalog
```

24 Zeichen lang.

Ist der LEASY-Katalog nicht auf dem Pubset der Kennung eingerichtet, unter der das Anwenderprogramm gestartet wird, so ist eine Katalogkennung :*catid*: für das Pubset mit dem LEASY-Katalog anzugeben.

katalog-zusatz — Zusatzangabe für Modelldateien 20 Zeichen lang.

Beispiel

```
            LEA@CAT $USER.KATALOG,FEBRUAR
   1        LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=CAT
   1 L@@CAT   DS    OCL44
   1 L@@CATC  DC    CL24'$USER.KATALOG'
   1 L@@CATZ  DC    CL20'FEBRUAR'
```

LEA@CI Currency Information

Dieser Makro erzeugt einen Bereich variabler Länge für die Currency Information. Er beginnt mit einem 4 Byte langen Satzlängenfeld.

Die Currency Information ist eine Liste der in der aktuellen Transaktion eröffneten Dateikennungen und der aktuellen Dateizeiger.

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@CI	[LEN=länge]

Der vom Makro *LEA@CI* definierte Bereich wird nur bei den LEASY-Aufrufen CINF (Makro *LEA@CINF*) und *OPTR* mit *OPE1=W* (Makro *LEA@OPTR*) benützt.

LEN=länge Länge des *CI*-Bereichs (einschließlich Satzlängenfeld) für die Übergabe an LEASY.

Beispiel

```
            LEA@CI LEN=80
1           LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=CI
1 L@@CI     DS    0H
1 L@@CIS    DC    Y(80)
1           DC    CL2' '
1 L@@CID    DS    CL(80-4)
```

LEA@DB Dateizuweisung (DB2/DB4)

Dieser Makro erzeugt einen Datenbereich für die Zuweisung der zu bearbeitenden Dateien oder Dateikennungen im Format DB2 und DB4 (siehe „Dateizuweisung DB“ auf Seite 136ff).

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
`name`	`LEA@DB`	`[string]` `[,LEN=länge]`

string — sind die zu bearbeitenden Dateien.

Angegeben werden darf

(*datei*,...) oder (*dateikennung*,...)

oder

(*datei,mod*) oder (*dateikennung,mod*)

oder

((*datei,mod*),(*datei,mod*),...) oder ((*dateikennung,mod*),

(*dateikennung,mod*),...).

datei	Dateiname als String
dateikennung	datei[/fm]
fm	Folgemerkmal
mod	OPEN- bzw. USAGE-Modus

(siehe Format DB2 und DB4, Seite 137ff).

LEN=länge — Länge des zu erzeugenden Datenbereichs.

– Wird weder *string* noch *länge* angegeben, so wird nur ein Adresspegel (mit EQU *) generiert.
– Es sind die Längenbegrenzungen des BS2000-Assemblers zu beachten.

Beispiel

```
           LEA@DB ((DATEI1,1),(DATEI2,2))
1          LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=DB
1 L@@DB    DC     '((DATEI1,1),(DATEI2,2))'
```

LEA@DB1 Dateizuweisung (DB1)

Dieser Makro erzeugt einen Datenbereich in der Länge von 12 Bytes für die Zuweisung der zu bearbeitenden Datei oder Dateikennung im Format DB1 (siehe „Dateizuweisung DB“ auf Seite 136ff).

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
`name`	`LEA@DB1`	`[dateikennung]`

dateikennung

Angabe der zu bearbeitenden Datei oder Dateikennung als String (max. 12 Zeichen) in der Form:

dateiname[/fm]

dateiname max. 8stelliger logischer Name der Datei.

fm max. 3stelliges Folgemerkmal der Datei.

Ist der Operand *dateikennung* angegeben, so wird der angegebene Dateiname in den Bereich eingesetzt (außer bei Verwendung innerhalb einer DSECT oder wenn *MF=D* angegeben wird).

Beispiel

```
  STAMDAT  LEA@DB1 STAMDAT
1          LEA@@BP TYP=L,PRE=L,NAM=STAMMDAT,GRU=GEN,ID=DB1
1 STAMDAT  DC    CL12'STAMDAT'
```

LEA@FA Feldauswahl

Dieser Makro generiert den Feldauswahl-Operanden in der Länge von 8 Bytes. Er legt fest, ob der ganze Datensatz in die Satzzone (*ar*) zurückgeliefert wird oder nur die Schlüsselinhalte (siehe „Feldauswahl FA“ auf Seite 146ff).

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@FA	[string]

string kann sein:

{(ALL) / ALL}	Der Datensatz wird zurückgeliefert
MAINITEM	Die Schlüsselinhalte werden zurückgeliefert.

Beispiel

```
           LEA@FA ALL
1          LEA@@BP PRE=L,NAM=,GRU=GEN,ID=FA,TYP=L
1 L@@FA    DC    CL8'ALL'
```

LEA@OP Operationscode

Dieser Makro generiert einen Operationscodebereich in der Länge 4 Bytes (siehe „Operationscode OP“ auf Seite 125ff).

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@OP	[operation]

operation
: wird als String angegeben und muss mit einer gültigen LEASY-Operation übereinstimmen.

  Wird *operation* angegeben, so wird der Bereich mit dem Wert dieses Operanden vorbesetzt, außer bei Verwendung innerhalb einer DSECT oder bei Angabe von *MF=D*.

Beispiel

```
  READNXT  LEA@OP RNXT
1          LEA@@BP TYP=L,PRE=L,NAM=READNXT,GRU=GEN,ID=OP
1 READNXT  DC    'RNXT'
```

LEA@OPS Operationscode/Konstantenliste

Dieser Makro erzeugt eine Konstantenliste aller LEASY-Operationscodes.

Der Operand *PRE* ist erlaubt.

Name	Operation	Operanden
name	LEA@OPS	

Die Operationscodes werden definiert mit:

```
L@@xxxx  DC  C'xxxx'
         .
         .
         .
```

xxxx ist je eine LEASY-Operation (z.B. *CATD*, etc.).

Die symbolischen Namen der Liste werden über *PRE*= mit einem Präfix versorgt.

Beispiel

```
          LEA@OPS
1 ********       LEASY OPERATION CODES
1 L@@CATD  DC    'CATD'                CONNECT TO / DISCONNECT FROM
1 *                                    COMMON MEMORY
1 L@@OPFL  DC    'OPFL'                OPEN FILE
1 L@@CLFL  DC    'CLFL'                CLOSE FILE
1 L@@OPTR  DC    'OPTR'                OPEN TRANSACTION
1 L@@CLTR  DC    'CLTR'                CLOSE TRANSACTION
1 L@@MARK  DC    'MARK'                MARK A CONSISTENCY POINT
1 L@@BACK  DC    'BACK'                BACK TO LAST CONSISTENCY POINT
1 L@@RDIR  DC    'RDIR'                READ DIRECT
1 L@@RHLD  DC    'RHLD'                READ DIRECT AND LOCK
1 L@@RNXT  DC    'RNXT'                READ NEXT
1 L@@RNHD  DC    'RNHD'                READ NEXT AND LOCK
1 L@@RPRI  DC    'RPRI'                READ PREVIOUSLY
1 L@@RPHD  DC    'RPHD'                READ PREVIOUSLY AND LOCK
1 L@@SETL  DC    'SETL'                SET FILE POINTER
1 L@@INSR  DC    'INSR'                INSERT
1 L@@STOR  DC    'STOR'                STORE
1 L@@REWR  DC    'REWR'                REWRITE
1 L@@DLET  DC    'DLET'                DELETE
1 L@@LOCK  DC    'LOCK'                LOCK
1 L@@UNLK  DC    'UNLK'                UNLOCK
1 L@@CINF  DC    'CINF'                READ CURRENT INFORMATION
```

LEA@RE Verständigungsbereich

Dieser Makro generiert einen LEASY-Verständigungsbereich. In diesem Bereich werden Informationen vom Anwender an LEASY übergeben (z.B. OPEN-Modus) oder von LEASY zurückgeliefert (z.B. Returncode).

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@RE	

Der erzeugte Verständigungsbereich ist 80 Bytes lang. Er wird auf Wortgrenze ausgerichtet.

Beispiel

```
          LEA@RE PRE=T
1 ********       LEASY COMMUNICATION AREA, VERSION 62A
1          DS    OF
1 T@@RE         DS    OCL80           LEASY COMMUNICATION AREA
1 T@@REK        DS    OCL48           COMPATIBLE PART
1 T@@RCCC       DC    CL3'000'        COMPATIBLE RETURNCODE
1 T@@RCOK       EQU   C'000'          RETURN-CODE: NO ERROR
1 T@@RCKZ       DC    CL1'L'          LEASY SYSTEM CHARACTERISTIC
1 T@@RCLC       DC    CL4'L000'       LEASY RETURN CODE
1 T@@PASS       DC    CL8' '          PASSWORD (FOR FUTURE VERSIONS)
1 T@@OPE        DS    OCL8            SUPPLEMENT FOR LEASY OPERATIONS
1 T@@OPSTX      DC    CL1' '          RESERVED
1 T@@STXA1      EQU   C' '             LEASY-STXIT-ROUTINE
1 T@@STXA2      EQU   X'00'            LEASY-STXIT-ROUTINE
1 T@@STXN       EQU   C' '             * VALUES 'N' AND 'P' NO LONGER
1 T@@STXP       EQU   C' '             * SUPPORTED SINCE LEASY V6.1
1 T@@OPOM       DC    CL1' '          OPEN MODE
1 T@@BLAN       EQU   C' '             STD FOR FORMAT DB1 AND DB2
1 T@@INPUT      EQU   C'1'             DVS OPEN MODE INPUT
1 T@@INPUS      EQU   C'2'             DVS OPEN MODE INPUT, SHARUPD
1 T@@INOUT      EQU   C'3'             DVS OPEN MODE INOUT
1 T@@INOUS      EQU   C'4'             DVS OPEN MODE INOUT, SHARUPD
1 T@@REVER      EQU   C'5'             DVS OPEN MODE REVERSE
1 T@@UPDAT      EQU   C'7'             DVS OPEN MODE UPDATE
1 T@@OUTPU      EQU   C'8'             DVS OPEN MODE OUTPUT
1 T@@EXTEN      EQU   C'9'             DVS OPEN MODE EXTEND
1 T@@OUTIN      EQU   C'A'             DVS OPEN MODE OUTIN
1 T@@OUTIS      EQU   C'B'             DVS OPEN MODE OUTIN, SHARUPD
1 *                              USAGE-MODES
1 T@@EXLD       EQU   C' '             USAGE-MODE EXLD (SAM)
1 T@@UPDT       EQU   C' '             USAGE-MODE UPDT (ISAM/PAM)
```

```
1 T@@RETRA      EQU   C'A'              USAGE-MODE RETR
1 T@@PRUPE      EQU   C'E'              USAGE-MODE PRUP
1 T@@EXRTG      EQU   C'G'              USAGE-MODE EXRT
1 T@@EXLDL      EQU   C'L'              USAGE-MODE EXLD
1 T@@PRRTI      EQU   C'I'              USAGE-MODE PRRT
1 T@@EXLDO      EQU   C'O'              USAGE-MODE EXLD
1 T@@ULRTR      EQU   C'R'              USAGE-MODE ULRT
1 T@@EXLDQ      EQU   C'Q'              USAGE-MODE EXLD
1 T@@ULUPU      EQU   C'U'              USAGE-MODE ULUP
1 T@@EXRTX      EQU   C'X'              USAGE-MODE EXRT
1 T@@EXUPB      EQU   C'B'              USAGE-MODE EXUP
1 *
1 T@@HVAL       EQU   X'FF'             FORMAT DB4
1 T@@OPLG       DC    CL1' '           BIM LOGGING FIELD
1 T@@YBIM       EQU   C' '              WITH BIM LOGGING
1 T@@NBIM       EQU   C'N'              WITHOUT BIM LOGGING
1          DC    CL5' '          UNUSED
1 T@@INT        DS    0XL8              FIELD FOR INTERNAL KEYS
1 T@@SPTR       DC    F'0'              SAM : ID1RPTR(24 BIT) OR
1 *                                           ID1BLK# (31-BIT)
1          ORG   T@@SPTR
1 T@@PAMHP      DC    F'0'              PAM : PAM BLOCK NUMBER
1 T@@SASNR      DC    F'0'              SAM:  ID1REC# (31-BIT)
1          ORG   T@@SASNR
1 T@@UNUSE      DC    F'0'              PAM :UNUSED, SAM UNUSED (24-BIT)
1 T@@NUM        DC    CL8' '            NUMBER OF PRIMARY RECORDS
1 T@@IDE        DC    CL8' '            DCAM IDENTIFIKATION
1 T@@RELE       DS    0CL32             LEASY EXTENSION OF RE
1 T@@REOP       DC    CL4' '            LAST OPERATION CODE
1 T@@REDB       DC    CL16' '           LAST FILE (+SI-NAME)
1 T@@LOPT       DC    CL1'1'            LEASY VERSION BYTE
1 *                                    HAS TO CONTAIN '1'
1 T@@OPE1       DC    CL1' '            FOR OPTR,CLTR,RHLD,RNHD,RPHD,
1 *                                   LOCK,CINF,UNLK
1 *                                   OPTR,CLTR:
1 T@@OCST       EQU   C' '                STANDARD OPEN/CLOSE OF TRANSACTION
1 *                                   OPTR:
1 T@@OPW        EQU   C'W'                OPEN TRANSAKTION USING
1 *                                    FILE POINTERS IN CI-AREA
1 *                                   CLTR:
1 T@@CLR        EQU   C'R'                ROLL BACK TRANSACTION
1 *                                   RHLD,RNHD,RPHD,LOCK:
1 T@@SLOC       EQU   C'S'                READ LOCK (SHARE LOCK)
1 T@@WLOC       EQU   C' '                WRITE LOCK
1 *                                   CINF:
1 T@@CINFW      EQU   C' '                CINF-AUFRUF FUER OPTR/W
1 T@@CIFI       EQU   C'F'                CINF FUER FILE-INFO
1 *
```

```
1 *                                UNLK:
1 T@@ULST      EQU   C' '              STANDARD FOR UNLK
1 T@@ULUP      EQU   C'U'              UNLK OF UPDATED RECORDS
1 *
1 T@@OPE2      DC    CL1' '      CLOSE TRANSACTION WITH
1 *                                   OR WITHOUT NEW START
1 T@@CLST      EQU   C' '         NO NEW START
1 T@@CLT       EQU   C'T'         NEW START
1 *
1 *
1 T@@NUMY      EQU   C'N'         RDIR/RHLD WITH NUM
1 T@@NUMN      EQU   C' '         RDIR/RHLD WITHOUT NUM
1 *                              NAEHERE ANGABEN FUER CINF
1 T@@CICA      EQU   C' '             INFO UEBER GANZEN KATALOG
1 T@@CICT      EQU   C'C'             INFO UEBER GANZEN KATALOG
1 T@@CIOP      EQU   C'O'             INFO UEBER OFFENE DATEIEN
1 T@@CITA      EQU   C'T'             INFO UEBER DATEIEN DER TA
1 T@@CISP      EQU   C'S'             INFO UEBER SPEZIELLE DATEI
1 T@@CIW       EQU   C'W'             FORTSETZUNG DER AUSKUNFTSFKT.
1 *
1 T@@ROL       EQU   C'L'             READ OVER LOCKED RECORD
1 T@@RNOL      EQU   C' '             DO NOT READ OVER LOCKED REC.
1 *
1 T@@OPWTM     DS    OZL3             WAIT TIME FOR LOCKING
1          DC    XL3'0'          DEFAULT VALUE
1 *
1 T@@EXRC      DC    CL5' '      ZUSAETZLICHER RETURNCODE
1 T@@UPROT     DC    C' '         AIM PROTOKOLLIERUNGS-KENNZEICHEN
1 T@@NPROT     EQU   C' '         NO USER-INFO AVAILABLE
1 T@@YPROT     EQU   C'Y'         USER-INFORMATION AVAILABLE
```

LEA@SI Sekundärindex

Dieser Makro erzeugt einen Datenbereich für die Übergabe eines Sekundärindex.

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@SI	[index]

index wird als String angegeben und ist maximal 8 Zeichen lang. Für *index* steht entweder der Name eines Sekundärindex oder *MAINITEM*. Bei *MAINITEM* wird auf den Primärschlüssel zugegriffen.

Beispiel

```
  FAMNAME  LEA@SI FAMNAME
1          LEA@@BP TYP=L,PRE=L,NAM=FAMNAME,GRU=GEN,ID=SI
1 FAMNAME  DC    CL8'FAMNAME'
```

LEA@US USER-Bereich

Dieser Makro erzeugt einen Bereich variabler Länge für die USER-Information. Er beginnt mit einem 4 Byte langen Satzlängenfeld.

Die Operanden *MF* und *PRE* sind erlaubt.

Name	Operation	Operanden
name	LEA@US	[LEN=laenge]

LEN=laenge — Länge des *USER*-Bereichs (einschließlich Satzlängenfeld) für die Übergabe an LEASY;
5<=*laenge*<=1024

Beispiel

```
           LEA@US LEN=1024
1          LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=US
1 L@@US    DS    0H
1 L@@USL   DC    Y(1024)
1          DC    CL2' '
1 L@@USD   DS    CL(1024-4)
```

11.4 Aktionsmakros

Die Aktionsmakros führen die Operationen durch, die im Einzelnen ebenso wie die verwendeten Stellungsoperanden im Kapitel „Übersicht über die LEASY-Schnittstelle“ auf Seite 121ff. beschrieben sind.

Übersicht über die Aktionsmakros:

Makro	Bedeutung
LEA@BACK	Rollback durchführen
LEA@CALL	LEASY-Operation ausführen
LEA@CATD	LEASY-Katalog ansprechen
LEA@CINF	Currency Information übergeben
LEA@CLFL	Dateien schließen
LEA@CLTR	Transaktion schließen
LEA@DLET	Bestehenden Satz löschen
LEA@INSR	Noch nicht bestehenden Satz einfügen
LEA@LOCK	Satzsperre setzen
LEA@MARK	Sicherungspunkt erzeugen
LEA@OPFL	Dateien eröffnen
LEA@OPTR	Transaktion eröffnen oder erweitern
LEA@PARC	Operandenliste korrigieren
LEA@RDIR	Satz direkt lesen
LEA@REWR	Bestehenden Satz ändern
LEA@RHLD	Satz lesen mit Satzsperre
LEA@RNHD	Nachfolger lesen mit Satzsperre
LEA@RNXT	Nachfolger lesen
LEA@RPHD	Vorgänger lesen mit Satzsperre
LEA@RPRI	Vorgänger lesen
LEA@SETL	Dateizeiger positionieren
LEA@STOR	Satz einfügen
LEA@TOLR	Fehlercodes behandeln
LEA@UNLK	Satzsperre aufheben

Nachfolgend sind die Aktionsmakros in alphabetischer Reihenfolge beschrieben.

LEA@BACK Rollback durchführen

Dieser Makro führt einen Rollback durch. Die aktuelle Transaktion wird mit der BIM-Datei rückgängig gemacht. Satzsperren werden freigegeben und ein Konsistenzpunkt gesetzt. Siehe „BACK Rollback durchführen“ auf Seite 151.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@BACK	[[re],[us]] [,SAVE=adresse2] [,PIDE=ide][,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes 1 bis 8 Zeichen.

=adresse3 — Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 — Adresse (*symb* oder (*r*))

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
LBACK       LEA@CALL ,L@@RE,MF=L
1           LEA@@BP GRU=GEN,NAM=LBACK,TYP=E,PRE=L
1           LEA@@AL ,L@@RE,,,,,,,PRE=L
2 LBACK     DS    OF
2           DC    A(X'00000000')
2           DC    A(L@@RE+X'80000000')
2           DC    A(X'00000000')
2           DC    A(X'00000000')
2           DC    A(X'00000000')
2           DC    A(X'00000000')
2           DC    A(X'00000000')
2           DC    A(X'00000000')

            LEA@BACK MF=(E,LBACK),                                          -
                 ERRADDR=ERROR
1           LEA@CALL 'BACK',,                                               C
1                  MF=(E,LBACK),SAVE=,PRE=L,                                C
1                 PIDE=,TIDE=,                                              C
1                  ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,LBACK
2           CNOP  2,4
2           LA    15,*+34
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'BACK'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           PRINT OFF
3           XR    15,15
3           BCTR  15,0                     R15=4X‘FF‘
3           LA    14,*+12+4+((L‘$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L‘$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                       RC OK
3           B     ERROR                    RC: ERROR
3           DC CL(L‘$@@RCCC)‘000‘
```

LEA@CALL LEASY-Operation ausführen

Mit diesem Makro kann jede LEASY-Operation durchgeführt werden. Er wird von allen anderen Aktionsmakros als Untermakro verwendet.

MF=(E,adresse1) kann angegeben werden.

Operation	Operanden
LEA@CALL	[[op],[re],[db],[ar],[fa],[si],[kb],[ke],[us]] [,SAVE=adresse2] [,POPE1=erg1] [,TOPE1=erg1] [,POPE2=erg2] [,TOPE2=erg2] [,POPEOM=openmode] [,TOPEOM=openmode] [,POPELOG=log] [,TOPELOG=log] [,POPEWTM=wartezeit] [,TOPEWTM=wartezeit] [,PSAMPTR=X'samzeiger'] [,TSAMPTR=X'samzeiger'] [,PPAMHP=X'pamblocknummer'] [,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

op *symb* oder (*r*) Adresse des Operationscodebereichs.
string Name der Operation.

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateiliste bzw. Dateikennung.
string Logischer Dateiname bzw. Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).
string Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

string Name eines Sekundärindex.

kb *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).

us *symb* oder (*r*) Adresse des *USER*-Bereichs

Mit *op* wird der Operationscode angegeben. Von diesem ist die benötigte bzw. berechtigte Anzahl von Stellungsoperanden und die Gültigkeit der Schlüsselwortoperanden abhängig (siehe Einzelbeschreibung der Aktionsmakros). Ist *op* in der Form *string* angegeben, so werden alle weiteren Operanden auf logische Verträglichkeit dazu geprüft. Andernfalls werden keine logischen, sondern nur syntaktische Prüfungen vorgenommen.

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1=erg1
TOPE1=erg1
} Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=W Transaktionsbeginn mit gleichzeitiger Dateipositionierung; nur bei *OPTR* zulässig.

=R Rücksetzen der Transaktion; nur bei *CLTR* zulässig.

=S Lesesperre bei den Operationen *LOCK, RHLD, RNHD, RPHD*.

=F Currency Information über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes; nur bei *CINF* zulässig.

=U Freigeben von Sperren modifizierter Sätze; nur bei *UNLK* zulässig.

=BLANK normales Transaktionsende bei *CLTR*

Schreibsperre bei den Operationen *LOCK,RHLD,RNHD,RPHD*

Currency Information über alle in der Transaktion eröffneten Dateikennungen bei *CINF*.

POPE2=erg2
TOPE2=erg2
} Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=T	Transaktionsende mit gleichzeitigem Transaktionsbeginn bei *CLTR*. Currency Information über alle an der Transaktion beteiligten Dateien bei *CINF*.
=N	Es wird die Anzahl der Primärschlüssel zu einem Sekundärschlüssel-Wert ermittelt nur bei *RDIR* und *RHLD* zulässig.
=C	Currency Information über alle im LEASY-Katalog enthaltenen Dateien; nur bei *CINF* zulässig.
=O	Currency Information über alle mit *OPFL* eröffneten Dateien; nur bei *CINF* zulässig.
=S	Currency Information über die in *CI* speizifizierte Datei; nur bei *CINF* zulässig.
=W	Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden; nur bei *CINF* zulässig.
=L	gesperrte Sätze sollen überlesen werden; nur bei *RPHD* und *RNHD* zulässig.
=BLANK	Transaktionsende mit Aufgabe aller Dateizugriffsanforderungen bei *CLTR*. Currency Information über alle im LEASY-Katalog enthaltenen Dateien bei *CINF*. Die Anzahl der Primärschlüssel wird nicht ermittelt; nur bei *RDIR* und *RHLD* zulässig.

POPEOM=openmode
TOPEOM=openmode

Eröffnungsart der Dateien bzw. Transaktionen (*P*=permanent, *T*=temporär).

Folgende Werte sind zulässig:

1, 2, 3, 4, 5, 7, 8, 9, A, B, E, G, I, L, O, Q, R, U, X, BLANK und X'FF'; nur bei *OPFL* und *OPTR* zulässig.

POPELOG=log
TOPELOG=log

Ein-/Ausschalten der BIM-Sicherung
(*P*=permanent, *T*=temporär).Nur bei *OPTR* zulässig.

=N BIM-Sicherung für laufende Transaktion wird abgeschaltet.

=BLANK BIM-Sicherung ist eingeschaltet.

POPEWTM=wartezeit
TOPEWTM=wartezeit

Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär) zulässig bei *OPTR, RHLD, RNHD, RPHD, INSR, STOR* und *LOCK*.

=wartezeit $0 \leq wartezeit \leq 999$.

=BLANK Es gilt die globale Wartezeit der Session (*TIME*-Anweisung in LEASY-MAINTASK).

PSAMPTR=samzeiger
TSAMPTR=samzeiger

Wiedergewinnungsadresse bei SAM-Dateien im 24- oder 31-Bit-Format (*P*=permanent, *T*=temporär).
Nur bei *SETL* und *RDIR* zulässig.

=X'bbbbbbrr' 24-Bit-Format

=X'bbbbbbbbrrrrrrrr'
31-Bit-Format

(*b* = Blocknummer, *r* = Satznummer innerhalb des Blocks)

PPAMHP
TPAMHP

PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Blocknummer in Sedezimalzeichen zulässig bei *SETL, RDIR, RHLD, STOR, INSR, REWR, DLET, LOCK* und *UNLK*, wenn eine PAM-Datei bearbeitet wird.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

ide={C'*String*' / X'*string*'} Die bis zu 8 Byte lange Zeichenkette stellt den Namen einer DCAM-Anwendung dar. Bei Angabe von *C*'...' wird der Name rechts mit Leerzeichen, bei Angabe von *X*'....' rechts mit binären Nullen aufgefüllt, falls der angegebene Name kürzer als 8 Zeichen ist.

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das den Namen einer DCAM-Anwendung oder eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das den Namen einer DCAM-Anwendung oder eine Transaktions-Identifikation enthält.

i Die Angaben *C'string'* oder *X'string'* sind nur beim Aktionsmakro *LEA@CATD* sinnvoll, da nur mit *LEA@CATD* der Name einer DCAM-Anwendung übergeben wird. Für alle übrigen Aktionsmakros sind in *LEA@CALL ide=symb* oder *ide=(r)* für Rückgabe der Transaktions-Identifikation anzugeben.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;
1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*))
Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@CALL MF=(E,ADDRLIST)
1           LEA@@BP GRU=AKT,NAM=
1           LA    1,ADDRLIST              R1=A(PARAMS)
1           CNOP  2,4
1           LM    14,15,*+6
1           BR    15                      CALL LEASY
1           DC    A(*+8)
1           DC    V(LEASY)
1           L     1,4(1)                  R1=A(RE)
  *
  ADDRLIST LEA@CALL L@@OP,                                              -
                L@@RE,                                                  -
                L@@DB1,                                                 -
                L@@AR,                                                  -
                L@@FA,                                                  -
                L@@SI,                                                  -
                LKB01,                                                  -
                LKE01,                                                  -
                SAVE=T@@RE,                                             -
                POPEWTM=0,                                              -
                TOPEWTM=25,                                             -
                ERRADDR=ERROR,                                          -
                ERRCODE=(O1O,LO12)
1           LEA@@BP GRU=AKT,NAM=ADDRLIST
1 ADDRLIST  LA    14,L@@RE                R14=A(RE)
1           LEA@@KO KORR=PERM,                                          C
1                 OPE1=,OPE2=,                                          C
1                 OPEOM=,OPELOG=,OPEWTM=0,                              C
1                 SAMPTR=,PAMHP=,IDE=
2           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10     MODIFY OPEWTIM
2           B     *+4+((L'$@@OPWTM+1)/2)*2
2           DC    ZL(L'$@@OPWTM)'0'
1           MVC   T@@RE(L'$@@RE),0(14)       SAVE=TEMP. RE
1           LA    14,T@@RE
1           LEA@@KO KORR=TEMP,                                          C
1                 OPE1=,OPE2=,                                          C
1                 OPEOM=,OPELOG=,OPEWTM=25,                             C
1                 SAMPTR=,PAMHP=,IDE=
2           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10     MODIFY OPEWTIM
2           B     *+4+((L'$@@OPWTM+1)/2)*2
2           DC    ZL(L'$@@OPWTM)'25'
1           CNOP  2,4
1           LA    15,L@@OP
1           ST    15,*+78                 A(OP)
1           ST    14,*+78                 A(RE)
1           LA    15,L@@DB1
1           ST    15,*+74                 A(DB)
1           LA    15,L@@AR
1           ST    15,*+70                 A(AR)
1           LA    15,L@@FA
1           ST    15,*+66                 A(FA)
1           LA    15,L@@SI
1           ST    15,*+62                 A(SI)
1           LA    15,LKB01
1           ST    15,*+58                 A(KB)
1           LA    15,LKE01
```

```
1          ST    15,*+54                 A(KE)
1          OI    *+50,X'80'              LAST PARAM-ADDRESS
1          LM    14,15,*+10
1          LA    1,*+14
1          BR    15                      CALL LEASY
1          DC    A(*+40)
1          DC    V(LEASY)
1 *
1          DC    A(0)             PTR(OP)
1          DC    A(0)             PTR(RE)
1          DC    A(0)             PTR(DB)
1          DC    A(0)             PTR(AR)
1          DC    A(0)             PTR(FA)
1          DC    A(0)             PTR(SI)
1          DC    A(0)             PTR(KB)
1          DC    A(0)             PTR(KE)
1          L     1,4(1)                  R1=A(RE)
1          LEA@@FB ERRADDR=ERROR,ERRCODE=(010,L012),R14=ON
2          XR    15,15
2          BCTR  15,0                    R15=4X'FF'
2          LA    14,*+32+4+16+((L'$@@RCCC+1)/2)*2      R14=RETURN ADDRESS
2          CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+12      TEST RC
2          BRE   14                      RC OK
2          B     *+4+((L'$@@RCCC+1)/2)*2 RC: ERROR
2          DC    CL(L'$@@RCCC)'000'
2          CLC   $@@RCCC-$@@RE(3,1),*+12      FEASIBLE RETURN CODE?
2          BRE   14                      YES, FEASIBLE
2          B     *+4+4
2          DC    C'010'
2          CLC   $@@RCLC-$@@RE(4,1),*+12      FEASIBLE RETURN CODE?
2          BRE   14                      YES, FEASIBLE
2          B     *+4+4
2          DC    C'L012'
2          B     ERROR                   RC: ERROR
```

LEA@CATD LEASY-Katalog ansprechen

Dieser Makro weist den LEASY-Katalog zu, der mit dem Dienstprogramm LEASY-CATALOG erstellt wurde. Siehe „CATD LEASY-Katalog ansprechen“ auf Seite 152.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@CATD	[[re],[cat],[us]] [,SAVE=adresse2] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

cat — *symb* oder (*r*) Adresse des *CAT*-Bereichs für die Katalogzuweisung.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide } Identifikation für DCAM (*P*=permanent, *T*=temporär).

ide={C’*String*’ / X’*string*’} — Die bis zu 8 Byte lange Zeichenkette stellt den Namen einer DCAM-Anwendung dar. Bei Angabe von *C*’...’ wird der Name rechts mit Leerzeichen, bei Angabe von *X*’....’ rechts mit binären Nullen aufgefüllt, falls der angegebene Name kürzer als 8 Zeichen ist.

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das den Namen einer DCAM-Anwendung enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das den Namen einer DCAM-Anwendung enthält.

ERRCODE=(fehlercode,...)

Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*))

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
           LA    R5,LCATD
           USING DCATD,R5
  *
  ECATD    LEA@CATD MF=(E,(R5)),
1 ECATD    LEA@CALL 'CATD',,,                                              C
1               MF=(E,(R5)),SAVE=,PRE=L,                                   C
1               PIDE=,TIDE=,                                               C
1               ERRCODE=,ERRADDR=
2          LEA@@BP GRU=AKT,NAM=ECATD
2 ECATD    LR    1,R5
2          CNOP  2,4
2          LA    15,*+34
2          TM    0(1),X'80'
2          BZ    *+8
2          O     15,=A(X'80000000')
2          ST    15,0(1)
2          LM    14,15,*+6
2          BR    15                      CALL LEASY
2          DC    A(*+12)
2          DC    V(LEASY)
2 *
2          DC    C'CATD'
2          L     1,4(1)                  R1=A(RE)
```

LEA@CINF Currency Information übergeben

Dieser Makro liefert dem Benutzer im *CI*-Bereich die gesamte Currency Information. Sie besteht aus einer Liste aller in der Transaktion eröffneten Dateikennungen und ihrer aktuellen Dateizeiger oder aus Informationen über die im LEASY-Katalog enthaltenen Dateien mit ihren Sekundärindizes. Siehe „CINF Currency Information übergeben“ auf Seite 154.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@CINF	[[re],[ci]] [,SAVE=adresse2] [,POPE1={F / BLANK}][,TOPE1={F / BLANK}] [,POPE2={C / O / T / S / W / BLANK}][,TOPE2={C / O / T / S / W / BLANK}] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

ci — *symb* oder (*r*) Adresse der Currency Information.

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

POPE1
TOPE1
} Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=F Currency Information über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes.

=BLANK Currency Information über alle in der Transaktion eröffneten Dateikennungen. Sie ermöglicht mit *OPTR* und *OPE1=W* eine Transaktionseröffnung mit gleichzeitiger Dateipositionierung.

POPE2
TOPE2
} Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=C
=BLANK
} Currency Information über alle im LEASY-Katalog enthaltenen Dateien.

=O Currency Information über alle mit *OPFL* eröffneten Dateien.

=T Currency Information über alle an der Transaktion beteiligten Dateien.

=S Currency Information über die in *CI* spezifizierte Datei.

=W Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes:

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
                LEA@CINF L@@RE,L@@CI,ERRADDR=ERROR
1               LEA@CALL 'CINF',L@@RE,L@@CI,                                  C
1                     MF=,SAVE=,PRE=L,                                        C
1                     POPE1=,POPE2=,PIDE=,                                    C
1                     TOPE1=,TOPE2=,TIDE=,                                    C
1                     ERRCODE=,ERRADDR=ERROR
2               LEA@@BP GRU=AKT,NAM=
2               CNOP  2,4
2               LA    15,L@@RE
2               ST    15,*+38                 A(RE)
2               LA    15,L@@CI
2               ST    15,*+34                 A(DB)
2               OI    *+30,X'80'              LAST PARAM-ADDRESS
2               LM    14,15,*+10
2               LA    1,*+14
2               BR    15                      CALL LEASY
2               DC    A(*+24)
2               DC    V(LEASY)
2 *
2               DC    A(*+12)           PTR(OP)
2               DC    A(0)              PTR(RE)
2               DC    A(0)              PTR(DB)
2               DC    C'CINF'
2               L     1,4(1)                  R1=A(RE)
2               LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3               XR    15,15
3               BCTR  15,0                    R15=4X'FF'
3               LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3               CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3               BRE   14                      RC OK
3               B     ERROR                   RC: ERROR
3               DC    CL(L'$@@RCCC)'000'
```

LEA@CLFL Dateien schließen

Dieser Makro schließt die angegebenen Dateien. Diese müssen in vorangehenden Operationen mit *OPFL* eröffnet worden sein. Siehe „CLFL Dateien schließen“ auf Seite 156.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@CLFL	[[re],[db],[us]] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateiliste bzw. Dateikennung.

'*string*' Logischer Dateiname bzw. Dateikennung bzw. Dateiliste

us *symb* oder (*r*) Adresse des *USER*-Bereichs

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@CLFL (R4),(R5)
1           LEA@CALL 'CLFL',(R4),(R5),                                      C
1                 MF=,PRE=L,                                                C
1                 ERRCODE=,ERRADDR=
2           LEA@@BP GRU=AKT,NAM=
2           CNOP  2,4
2           ST    R4,*+34                 A(RE)
2           ST    R5,*+34                 A(DB)
2           OI    *+30,X'80'              LAST PARAM-ADDRESS
2           LM    14,15,*+10
2           LA    1,*+14
2           BR    15                      CALL LEASY
2           DC    A(*+24)
2           DC    V(LEASY)
2 *
2           DC    A(*+12)           PTR(OP)
2           DC    A(0)              PTR(RE)
2           DC    A(0)              PTR(DB)
2           DC    C'CLFL'
2           L     1,4(1)                  R1=A(RE)
```

LEA@CLTR Transaktion schließen

Der Makro *LEA@CLTR* beendet die Transaktion und setzt einen Konsistenzpunkt. Siehe „CLTR Transaktion schließen“ auf Seite 157.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@CLTR	[[re],[us]] [,SAVE=adresse2] [,POPE1={R / BLANK}][,TOPE1={R / BLANK}] [,POPE2={T / BLANK}][,TOPE2={T / BLANK}] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1, TOPE1 — Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=R — Rücksetzen der Transaktion.

=BLANK Normales Transaktionsende.

POPE2
TOPE2
} Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=T Transaktionsende mit gleichzeitigem Transaktionsbeginn.

=BLANK Transaktionsende mit Aufgabe aller Dateizugriffsanforderungen.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
                LEA@CLTR L@@RE,POPE1=BLANK,POPE2=T
1               LEA@CALL 'CLTR',L@@RE,                                  C
1                     MF=,SAVE=,PRE=L,                                  C
1                     POPE1=BLANK,POPE2=T,PIDE=,                        C
1                     TOPE1=,TOPE2=,TIDE=,                              C
1                     ERRCODE=,ERRADDR=
2               LEA@@BP GRU=AKT,NAM=
2               LA    14,L@@RE                R14=A(RE)
2               LEA@@KO OPCD=CLTR,KORR=PERM,                            C
2                     OPE1=BLANK,OPE2=T,                                C
2                     OPEOM=,OPELOG=,OPEWTM=,                           C
2                     SAMPTR=,PAMHP=,IDE=
3               MVI   $@@OPE1-$@@RE(14),C' '     MODIFY OPE1
3               MVI   $@@OPE2-$@@RE(14),C'T'         MODIFY OPE2
2               CNOP  2,4
2               LA    15,L@@RE
2               ST    15,*+30                 A(RE)
2               OI    *+26,X'80'              LAST PARAM-ADDRESS
2               LM    14,15,*+10
2               LA    1,*+14
2               BR    15                      CALL LEASY
2               DC    A(*+20)
2               DC    V(LEASY)
2 *
2               DC    A(*+8)            PTR(OP)
2               DC    A(0)              PTR(RE)
2               DC    C'CLTR'
2               L     1,4(1)                  R1=A(RE)
```

LEA@DLET Bestehenden Satz löschen

Dieser Makro löscht einen Satz aus einer ISAM- oder DAM-Datei bzw. einen Block aus einer PAM-Datei. Siehe „DLET Bestehenden Satz löschen“ auf Seite 158.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@DLET	[[re],[db],[ar],[fa],[si],[kb],[us]] [,SAVE=adresse2] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateikennung

'*string*' Dateikennung.

ar — *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa — *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si — *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindexes.

kb — *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PPAMHP, TPAMHP — PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Nummer des zu löschenden PAM-Blocks.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@DLET MF=(E,ADDRLIST),                                       -
                  ERRADDR=ERROR
1           LEA@CALL 'DLET',,,,,,,,                                         C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                              C
1                 PPAMHP=,PIDE=,                                            C
1                 TPAMHP=,TIDE=,                                            C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           CNOP  2,4
2           LA    15,*+34
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                        CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'DLET'
2           L     1,4(1)                    R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                      R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                        RC OK
3           B     ERROR                     RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@INSR Noch nicht bestehenden Satz einfügen

Dieser Makro fügt einen neuen Satz bzw. Block in die angegebene Datei ein. Siehe „INSR Noch nicht bestehenden Satz einfügen“ auf Seite 159.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@INSR	[[re],[db],[ar],[us]] [,SAVE=adresse2] [,POPEWTM={wartezeit | BLANK}] [,TOPEWTM={wartezeit | BLANK}] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar — *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des *RE*-Bereichs.

POPEWTM, TOPEWTM — Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).

=wartezeit — $0 \leq wartezeit \leq 999$.

=BLANK Es gilt die globale Wartezeit der Session; (*TIME*-Anweisung in LEASY-MAINTASK).

PPAMHP
TPAMHP } PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Blocknummer des Blocks, in den eingefügt wird.

PIDE=ide
TIDE=ide } Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@INSR L@@RE,'PAMFILE     ',L@@AR,                          -
                  TOPEWTM=BLANK,                                           -
                  TPAMHP=X'1',                                             -
                  ERRADDR=ERROR
1           LEA@CALL 'INSR',L@@RE,'PAMFILE     ',L@@AR,                   C
1                 MF=,SAVE=,PRE=L,                                         C
1                 PPAMHP=,POPEWTM=,PIDE=,                                  C
1                 TPAMHP=X'1',TOPEWTM=BLANK,TIDE=,                         C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    14,L@@RE                  R14=A(RE)
2           MVC   *+14(L'$@@RE),0(14)        TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)                 TEMP. RE
2           LEA@@KO OPCD=INSR,KORR=TEMP,                                   C
2                 OPE1=,OPE2=,                                             C
2                 OPEOM=,OPELOG=,OPEWTM=BLANK,                             C
2                 SAMPTR=,PAMHP=X'1'
3           MVI   $@@OPWTM-$@@RE(14),C' ' MODIFY OPEWTIM
3           MVC   $@@OPWTM+1-$@@RE(L'$@@OPWTM-1,14),$@@OPWTM-$@@RE(14)
3           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10 MODIFY PAMHP
3           B     *+4+L'$@@PAMHP
3           DC    XL(L'$@@PAMHP)'1'
2           CNOP  2,4
2           ST    14,*+38               A(RE)
2           LA    15,L@@AR
2           ST    15,*+38               A(AR)
2           OI    *+34,X'80'            LAST PARAM-ADDRESS
2           LM    14,15,*+10
2           LA    1,*+14
2           BR    15                    CALL LEASY
2           DC    A(*+40)
2           DC    V(LEASY)
2 *
2           DC    A(*+16)               PTR(OP)
2           DC    A(0)                  PTR(RE)
2           DC    A(*+12)               PTR(DB)
2           DC    A(0)                  PTR(AR)
2           DC    C'INSR'
2           DC    C'PAMFILE     '
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4 TEST RC
3           BRE   14                      RC OK
3           B     ERROR                 RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@LOCK Satzsperre setzen

Dieser Makro legt Sperrelemente an für einzelne Sätze bzw. Blöcke von ISAM-, DAM- oder PAM-Dateien. Da kein Dateizugriff erfolgt, wird nicht geprüft, ob der entsprechende Satz oder Block vorhanden ist. Daher können auch Sätze oder Blöcke gesperrt werden, die noch nicht existieren. Siehe „LOCK Satzsperre setzen“ auf Seite 160.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@LOCK	[[re],[db],[ar],[fa],[si],[kb],[ke]] [,SAVE=adresse2] [,POPE1={S | BLANK}] [,TOPE1={S | BLANK}] [,POPEWTM={wartezeit | BLANK}] [,TOPEWTM={wartezeit | BLANK}] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindexes.

kb *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke — *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des *RE*-Bereichs.

POPE1
TOPE1
} Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=S — Anlegen eines READ-LOCK.

=BLANK — Anlegen eines WRITE-LOCK.

POPEWTM
TOPEWTM
} Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).

=wartezeit — 0 ≤ *wartezeit* ≤ 999.

=BLANK — Es gilt die globale Wartezeit der Session;
(*TIME*-Anweisung in LEASY-MAINTASK).

PPAMHP
TPAMHP
} PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' — Nummer des zu sperrenden Blocks.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;
1 bis 8 Zeichen.

=adresse3 — Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@LOCK MF=(E,ADDRLIST),                                       -
                  TOPEWTM=30,                                               -
                  ERRADDR=ERROR
1           LEA@CALL 'LOCK',,,,,,,,
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                              C
1                 PPAMHP=,POPEWTM=,PIDE=,POPE1=                             C
1                 TPAMHP=,TOPEWTM=30,TIDE=,OPE1=                            C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           L     14,4(1)                 R14=A(RE)
2           MVC   *+14(L'$@@RE),0(14)        TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)                TEMP. RE
2           LEA@@KO OPCD=LOCK,KORR=TEMP,                                    C
2                 OPE1=,OPE2=,                                              C
2                 OPEOM=,OPELOG=,OPEWTM=30,                                 C
2                 SAMPTR=,PAMHP=,IDE
3           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10 MODIFY OPEWTIM
3           B     *+4+((L'$@@OPWTM+1)/2)*2
3           DC    ZL(L'$@@OPWTM)'30'
2           CNOP  2,4
2           LA    15,*+50
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                 MODIFY RE
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'LOCK'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4 TEST RC
3           BRE   14                      RC OK
3           B     ERROR                RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@MARK Sicherungspunkt erzeugen

Dieser Makro beendet die aktuelle Transaktion und setzt einen Konsistenzpunkt. Siehe „MARK Sicherungspunkt erzeugen“ auf Seite 163.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@MARK	[[re],[us]] [,SAVE=adresse2] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes:

1 bis 8 Zeichen.

=adresse3 — Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 — Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@CALL MF=(E,LMARK)
1           LEA@@BP GRU=AKT,NAM=
1           LA    1,LMARK                 R1=A(PARAMS)
1           CNOP  2,4
1           LM    14,15,*+6
1           BR    15                      CALL LEASY
1           DC    A(*+8)
1           DC    V(LEASY)
1           L     1,4(1)                  R1=A(RE)

  LMARK     LEA@MARK L@@RE,MF=L
1 LMARK     LEA@CALL 'MARK',L@@RE,                                          C
1                 MF=L,SAVE=,PRE=L,                                         C
                  PIDE=,TIDE=,                                              C
1                 ERRCODE=,ERRADDR=
2           LEA@@BP GRU=GEN,NAM=LMARK,TYP=E,PRE=L
2           LEA@@AL 'MARK',L@@RE,,,,,,,PRE=L
3 LMARK     DS    OF
3           DC    A(*+32+X'00000000')
3           DC    A(L@@RE+X'80000000')
3           DC    A(X'00000000')
3           DC    A(X'00000000')
3           DC    A(X'00000000')
3           DC    A(X'00000000')
3           DC    A(X'00000000')
3           DC    A(X'00000000')
3           DC    CL4'MARK'
```

LEA@OPFL Dateien eröffnen

Dieser Makro eröffnet die in der Dateiliste angegebenen Dateien entsprechend dem zugehörigen OPEN-Modus. Siehe „OPFL Dateien eröffnen“ auf Seite 164.

Die Operation ist nur zulässig, wenn für den Prozess keine Transaktionen eröffnet sind.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@OPFL	[[re],[db],[us]] [,SAVE=adresse2] [,POPEOM=openmode][,TOPEOM=openmode] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateiliste.

'*string*' Logischer Dateiname.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des *RE*-Bereichs.

POPEOM, TOPEOM — Eröffnungsart der Dateien bzw. Transaktionen (*P*=permanent, *T*=temporär).

=openmode — Zulässiger OPEN-Modus;

(siehe Seite 188).

Folgende Werte sind zulässig:

1, 2, 3, 4, 5, 7, 8, 9, A, B, X'FF'.

ERRCODE=(fehlercode,...)

Liste der zu tolerierenden Fehlercodes;
1 bis 8 Zeichen.

=adresse3 — Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LA    R5,L@@RE
            LA    R6,ERROR
  *
            LEA@OPFL (R5),'(DATEI1,DATEI2)',                              -
                  TOPEOM=1,                                               -
                  ERRCODE=ERROUT,                                         -
                  ERRADDR=(R6)
1           LEA@CALL 'OPFL',(R5),'(DATEI1,DATEI2)',                       C
1                 MF=,SAVE=,PRE=L,                                        C
1                 POPEOM=,TOPEOM=1,                                       C
1                 ERRCODE=ERROUT,ERRADDR=(R6)
2           LEA@@BP GRU=AKT,NAM=
2           LR    14,R5               R14=A(RE)
2           MVC   *+14(L'$@@RE),0(14)       TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)               TEMP. RE
2           LEA@@KO OPCD=OPFL,KORR=TEMP,                                  C
2                 OPE1=,OPE2=,                                            C
2                 OPEOM=1,OPELOG=,OPEWTM=,                                C
2                 SAMPTR=,PAMHP=,IDE=
3           MVI   $@@OPOM-$@@RE(14),C'1'    MODIFY OPEOM
2           CNOP  2,4
2           ST    14,*+30                   A(RE)
2           OI    *+30,X'80'                LAST PARAM-ADDRESS
2           LM    14,15,*+10
2           LA    1,*+14
2           BR    15                        CALL LEASY
2           DC    A(*+40)
2           DC    V(LEASY)
2 *
2           DC    A(*+12)                   PTR(OP)
2           DC    A(0)                      PTR(RE)
2           DC    A(*+8)                    PTR(DB)
2           DC    C'OPFL'
2           DC    C'(DATEI1,DATEI2)'
2           L     1,4(1)                    R1=A(RE)
2           LEA@@FB ERRADDR=(R6),ERRCODE=ERROUT,R14=ON
3           LA    14,*+16+2+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+12+2 TEST RC
3           BRE   14                        RC OK
3           LR    15,R6          R15=A(ERROR-ROUTINE)
3           B     ERROUT
3           DC    CL(L'$@@RCCC)'000'
```

LEA@OPTR Transaktion eröffnen oder erweitern

LEA@OPTR eröffnet bzw. erweitern eine Transaktion. Siehe „OPTR Transaktion eröffnen oder erweitern“ auf Seite 165.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@OPTR	[[re],[{db / ci}],[us]] [,SAVE=adresse2] [,POPE1={W / BLANK}] [,TOPE1={W / BLANK}] [,POPEOM=openmode] [,TOPEOM=openmode] [,POPELOG={N / BLANK}] [,TOPELOG={N / BLANK}] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennungsliste.

'*string*' Dateikennung bzw. Dateikennungsliste.

ci *symb* oder (*r*) Adresse der Currency Information.

Funktion a (Operand db)

Eine Transaktion wird eröffnet, falls noch keine aktiv ist. Alle in der Dateiliste (db) angegebenen Dateikennungen werden mit den zugehörigen USAGE-Modi für die Transaktion logisch eröffnet.

Die Dateizeiger werden auf Dateianfang und Primärschlüssel positioniert.

Ist zum Zeitpunkt des Makroaufrufs bereits eine Transaktion dieses Benutzers eröffnet, so wird diese Transaktion um die in der Dateiliste angegebenen Dateikennungen erweitert. Es wird auf Dateianfang und Primärschlüssel positioniert.

Funktion b (Operand ci und OPE1=W)

Eine Transaktion wird eröffnet. Die in der Currency Information abgelegten Dateien werden eröffnet. Es wird auf die Positionen positioniert, die in der Currency Information hinterlegt sind.

us — *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1=erg1
TOPE1=erg1
} Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=W — Transaktionsbeginn mit gleichzeitiger Dateipositionierung.

=BLANK — Normaler Transaktionsbeginn.

POPEOM
TOPEOM
} Eröffnungsart der Dateien bzw. Transaktionen (*P*=permanent, *T*=temporär).

Folgende Werte sind möglich:

A, B, E, G, I, L, O, Q, R, U, X, BLANK und *X'FF'*.

POPELOG
TOPELOG
} Ein-/Ausschalten der BIM-Sicherung (*P*=permanent, *T*=temporär).

=N — BIM-Sicherung für laufende Transaktion wird abgeschaltet.

=BLANK	BIM-Sicherung ist eingeschaltet.
PIDE=ide TIDE=ide	Identifikation für DCAM (*P*=permanent, *T*=temporär).
=symb	Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das binäre Nullen oder Leerzeichen enthält.
=(r)	Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das binäre Nullen oder Leerzeichen enthält.
ERRCODE=(fehlercode,...)	Liste der zu tolerierenden Fehlercodes; 1 bis 8 Zeichen.
=adresse3	Symbolische Adresse für Fehlerbehandlung.
ERRADDR=adresse4	Adresse (*symb* oder (*r*)); Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LA    R4,ADDRLIST
            LA    R5,T@@RE
  *
            LEA@OPTR MF=(E,(R4)),                                          -
                  SAVE=(R5),                                               -
                  TOPE1=W,                                                 -
                  ERRADDR=ERROR
1           LEA@CALL 'OPTR',,,                                             C
1                 MF=(E,(R4)),SAVE=(R5),PRE=L,                             C
1                 POPE1=,POPELOG=,POPEOM=,PIDE=,                           C
1                 TOPE1=W,TOPELOG=,TOPEOM=,TIDE=                           C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LR    1,R4
2           L     14,4(1)                 R14=A(RE)
2           MVC   0(L'$@@RE,R5),0(14)     SAVE=TEMP. RE
2           LR    14,R5
2           LEA@@KO OPCD=OPTR,KORR=TEMP,                                   C
2                 OPE1=W,OPE2=,                                            C
2                 OPEOM=,OPELOG=,OPEWTM=,                                  C
2                 SAMPTR=,PAMHP=,IDE=
3           MVI   $@@OPE1-$@@RE(14),C'W'  MODIFY OPE1
2           CNOP  2,4
2           LA    15,*+50
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                 MODIFY RE
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'OPTR'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4 TEST RC
3           BRE   14                      RC OK
3           B     ERROR               RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@PARC Operandenliste korrigieren

Dieser Makro bereitet einen Verständigungsbereich (*re* oder Adresse aus *ADDRLST*) bzw. eine Operandenliste (*ADDRLST*) für einen LEASY-Aufruf vor.

Operation	Operanden
LEA@PARC	[[op],[re],[db],[ar],[fa],[si],[kb],[ke]] [,ADDRLST=adresse1] [,LASTPAR=wert] [,POPE1=erg1] [,POPE2=erg2] [,POPSTX=stxitmode] [,POPEOM=openmode] [,POPELOG=log] [,POPEWTM={wartezeit | BLANK}] [,PSAMPTR=X'samzeiger'] [,PPAMHP=X'pamblocknummer'] [,PIDE=ide]

Die angegebenen Stellungsoperanden (*op, re* etc.) korrigieren die Operandenliste (*ADDRLST*). Die Operanden *POPE1*, *POPE2* etc. korrigieren den *RE*-Bereich, dessen Adresse im Makroaufruf (*re*) angegeben wurde bzw. aus der Operandenliste (2. Wort in *ADDRLST*) entnommen wird.

Sind *ADDRLST* und *LASTPAR* vorhanden, so wird in *ADDRLST* bei dem durch *wert* bezeichneten Operandenwort das Kennzeichen letzter Operand gesetzt. Ist z.B. *LASTPAR=4*, so wird das linke Byte der Adresse von AR (4. Operandenwort) mit *X'80'* belegt. Eine größere Ziffer als *8* bewirkt das Löschen aller linken Adressbytes.

op *symb* oder (*r*) Adresse des Operationscodebereichs.

'*string*' Name der Operation.

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db	*symb* oder (*r*) Adresse der Dateiliste.
	'*string*' Logischer Dateiname bzw. Dateikennung bzw. Dateiliste.
ar	*symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).
fa	*symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).
	'*string*' Kennzeichen für die Feldauswahl.
si	*symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).
	'*string*' Name eines Sekundärindexes.
kb	*symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).
ke	*symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).
ADDRLST=adresse1	Adresse (*symb* oder (*r*)) der Operandenliste, wie sie bei Aktionsmakro im Operanden *MF=(E, adresse1*) erwartet wird.
LASTPAR=wert	$1 \leq wert \leq 8$
	bezeichnet das Wort an der Position *wert* in der Operandenliste. Das linke Byte dieser Adresse wird mit dem Kennzeichen *X'80'* (letzter Operand) belegt.
POPE1=erg1	Operationscodeergänzung *OPE1*.
=W	Transaktionsbeginn mit gleichzeitiger Dateipositionierung; nur bei *OPTR* zulässig.
=R	Rücksetzen der Transaktion; nur bei *CLTR* zulässig.
=S	Lesesperre bei den Operationen *LOCK, RHLD, RNHD, RPHD*.
=F	Currency Information über die im LEASY-Katalog enthaltenen Dateien und deren Sekundärindizes; nur bei *CINF* zulässig.
=U	Freigeben von Sperren modifizierter Sätze; nur bei UNLK zulässig.
=BLANK	Normaler Transaktionsbeginn oder normales Transaktionsende oder Schreibsperre
	Currency Information über alle in der Transaktion eröffneten Dateikennungen bei *CINF*.
POPE2=erg2	Operationscodeergänzung *OPE2*.
=T	Transaktionsende mit gleichzeitigem Transaktionsbeginn bei *CLTR*.
	Currency Information über alle an der Transaktion beteiligten Dateien bei *CINF*.

=N	Es wird die Anzahl der Primärschlüssel zu einem Sekundärschlüssel-Wert ermittelt; nur bei *RDIR* und *RHLD* zulässig.
=C	Currency Information über alle im LEASY-Katalog enthaltenen Dateien; nur bei *CINF* zulässig.
=O	Currency Information über alle mit *OPFL* eröffneten Dateien; nur bei *CINF* zulässig.
=S	Currency Information über die in *CI* spezifizierte Datei; nur bei *CINF* zulässig.
=W	Die unmittelbar vorausgegangene Auskunftsfunktion soll fortgesetzt werden; nur bei *CINF* zulässig.
=L	gesperrte Sätze sollen überlesen werden; nur bei *RPHD* und *RNHD* zulässig.
=BLANK	Transaktionsende mit Aufgabe aller Dateizugriffsanforderungen bei *CLTR*. Currency Information über alle im LEASY-Katalog enthaltenen Dateien bei *CINF*. Die Anzahl der Primärschlüssel wird nicht ermittelt bei *RDIR* und *RHLD*.
POPSTX=stxitmode	Dieser Operand wird nur noch aus Kompatibilitätsgründen angeboten, aber nicht mehr ausgewertet.
=P	wird ignoriert
=N	wird ignoriert
=BLANK	LEASY-STXIT-Routine.
POPEOM	Eröffnungsart.
=openmode	Zulässiger OPEN-Modus oder USAGE-Modus; (siehe Abschnitt „Dateien und Transaktionen eröffnen“ auf Seite 188f). Folgende Werte sind zulässig: *1, 2, 3, 4, 5, 7, 8, 9, A, B, E, G, I, L, O, Q, R, U, X, BLANK* und *X’FF’*.
POPELOG=log	Ein-Ausschalten der BIM-Sicherung; nur bei *OPTR* zulässig.
=N	BIM-Sicherung für laufende Transaktion wird ausgeschaltet.
=BLANK	BIM-Sicherung ist eingeschaltet.

POPEWTM	Wartezeit auf gesperrte Sätze.
=wartezeit	$0 \leq wartezeit \leq 999$.
=BLANK	Es gilt die globale Wartezeit der Session (*TIME*-Anweisung in LEASY-MAINTASK).
PSAMPTR	SAM-Wiedergewinnungsadresse im 24- oder 31-Bit-Format.
=X'bbbbbbrr'	24-Bit-Format
=X'bbbbbbbbrrrrrrrr'	31-Bit-Format (b = Blocknummer, r = Satznummer innerhalb des Blocks)
PPAMHP	PAM-Blocknummer.
=X'bbbbbb'	Nummer des PAM-Blocks.
PIDE=ide	Identifikation für DCAM.
=symb	Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das den Namen einer DCAM-Anwendung oder eine Transaktions-Identifikation enthält.
=(r)	Das Mehrzweckregister r enthält die Adresse eines 8 Byte langen Feldes, das den Namen einer DCAM-Anwendung oder eine Transaktions-Identifikation enthält.

Beispiel

```
            LEA@PARC ,L@@RE,POPEWTM=5,PPAMHP=X'2'
1           LEA@@BP GRU=AKT,NAM=
1           LA    14,L@@RE                  R14=A(RE)
1           LEA@@KO                                                      C
1                 OPE1=,OPE2=,                                           C
1                 OPEOM=,OPELOG=,OPEWTM=5,                               C
1                 SAMPTR=,PAMHP=X'2',IDE=,                               C
1                 OPSTX=
2           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10 MODIFY OPEWTIM
2           B     *+4+((L'$@@OPWTM+1)/2)*2
2           DC    ZL(L'$@@OPWTM)'5'
2           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10 MODIFY PAMHP
2           B     *+4+L'$@@PAMHP
2           DC    XL(L'$@@PAMHP)'2'
  PARC      LEA@PARC ,L@@RE,POPEWTM=0,ADDRLST=ADDRLIST,LASTPAR=2
1           LEA@@BP GRU=AKT,NAM=PARC
1 PARC      LA    1,ADDRLIST
1           LA    14,L@@RE                  R14=A(RE)
1           LEA@@KO                                                      C
1                 OPE1=,OPE2=,                                           C
1                 OPEOM=,OPELOG=,OPEWTM=0,                               C
1                 SAMPTR=,PAMHP=,IDE=,                                   C
1                 OPSTX=
2           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10 MODIFY OPEWTIM
2           B     *+4+((L'$@@OPWTM+1)/2)*2
2           DC    ZL(L'$@@OPWTM)'0'
1           LA    15,L@@RE
1           TM    4(1),X'80'
1           BZ    *+8
1           O     15,=A(X'80000000')
1           ST    15,4(1)                MODIFY RE
1 *
1           LEA@@LP LASTPAR=2
2           PRINT OFF
2           NI    ($@@POP-$@@POP)(1),X'FF'-$@@LAST
2           OI    ($@@PRE-$@@POP)(1),$@@LAST
```

LEA@RDIR Satz direkt lesen

Dieser Makro liest einen Satz oder Block:

- Ein Satz einer ISAM- oder DAM-Datei wird über einen angegebenen Schlüssel
- Ein Satz einer SAM-Datei wird über die angegebene Wiedergewinnungsadresse
- Ein logischer Block einer PAM-Datei wird über eine angegebene Blocknummer

direkt in den Eingabebereich *AR* (siehe Seite 169ff).

Zusätzlich zum Lesen erfolgt eine Positionierung auf den gefundenen Schlüssel und den verwendeten Index (Primär- oder Sekundärindex).

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RDIR	[[re],[db],[ar],[fa],[si],[kb],[ke]] [,SAVE=adresse2] [,POPE2=erg2] [,TOPE2=erg2] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PSAMPTR=X'samzeiger'] [,TSAMPTR=X'samzeiger'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindexes.

kb — *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke — *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE2=erg2
TOPE2=erg2
— Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=N — Es wird die Anzahl der Primärschlüssel zu einem Sekundärschlüssel-Wert ermittelt.

=BLANK — Die Anzahl der Primärschlüssel wird nicht ermittelt.

PPAMHP
TPAMHP
— PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' — Nummer des Blocks, der gelesen werden soll.

PSAMPTR
TSAMPTR
— Wiedergewinnungsadresse eines Satzes in einer SAM-Datei im 24- oder 31-Bit-Format (*P*=permanent, *T*=temporär).

=X'bbbbbbrr' — 24-Bit-Format

=X'bbbbbbbbrrrrrrrr' — 31-Bit-Format

(*b* = Blocknummer, *r* = Satznummer innerhalb des Blocks)

Bei ISAM- und DAM-Dateien ist der Schlüsselwert im *AR*-Bereich zu übergeben. Zur Verwendung der Operanden *KB* und *KE* siehe Seite 148 und 169ff.

PIDE=ide
TIDE=ide
— Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
: Liste der zu tolerierenden Fehlercodes;

  1 bis 8 Zeichen.

=adresse3
: Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4
: Adresse (*symb* oder (*r*));

  Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LA    R4,ADDRLIST
            LA    R5,ERROR
            LA    R6,PAMDB
  *
            LEA@RDIR ,(R6),MF=(E,(R4)),                                    -
                  SAVE=T@@RE,                                              -
                  TPAMHP=X'ABC',                                           -
                  ERRADDR=ERROR
1           LEA@CALL 'RDIR',,(R6),,,,,,                                    C
1                 MF=(E,(R4)),SAVE=T@@RE,PRE=L,                            C
1                 PPAMHP=,PSAMPTR=,PIDE=,                                  C
1                 POPE2=                                                   C
1                 TPAMHP=X'ABC',TSAMPTR=,TIDE=,                            C
1                 TOPE2=,                                                  C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LR    1,R4
2           L     14,4(1)                  R14=A(RE)
2           MVC   T@@RE(L'$@@RE),0(14)        SAVE=TEMP. RE
2           LA    14,T@@RE
2           LEA@@KO OPCD=RDIR,KORR=TEMP,                                   C
2                 OPE1=,OPE2=,                                             C
2                 OPEOM=,OPELOG=,OPEWTM=,                                  C
2                 SAMPTR=,PAMHP=X'ABC',IDE=
3           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10 MODIFY PAMHP
3           B     *+4+L'$@@PAMHP
3           DC    XL(L'$@@PAMHP)'ABC'
2           CNOP  2,4
2           LA    15,*+30
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                  MODIFY RE
2           LM    14,15,*+6
2           BR    15                       CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'RDIR'
2           L     1,4(1)                   R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                     R15=4X'FF'
3           LA    14,*+12+2+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4 TEST RC
3           BRE   14                       RC OK
3           B     ERROR                    RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@REWR Bestehenden Satz ändern

Dieser Makro ändert einen bestehenden Satz oder Block. Wird für eine Datei ein Sperrprotokoll geführt, so muss der Satz bzw. Block vorher gesperrt worden sein. Siehe „REWR Bestehenden Satz ändern“ auf Seite 175.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@REWR	[[re],[db],[ar],[us]] [,SAVE=adresse2] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse2} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

us *symb* oder (*r*) Adresse des *USER*-Bereichs

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PPAMHP
TPAMHP
} PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Nummer des Blocks, der gelesen werden soll.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LA    R4,ERROR           R4=A(FEHLERROUTINE)
  *
            LEA@REWR MF=(E,ADDRLIST),                                    -
                  PPAMHP=X'2',                                           -
                  ERRADDR=(R4)
1           LEA@CALL 'REWR',,,,                                          C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                           C
1                 PPAMHP=X'2',PIDE=                                      C
1                 TRAMHP=,TIDE=,                                         C
1                 ERRCODE=,ERRADDR=(R4)
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           L     14,4(1)                  R14=A(RE)
2           LEA@@KO OPCD=REWR,KORR=PERM,                                 C
2                 OPE1=,OPE2=,                                           C
2                 OPEOM=,OPELOG=,OPEWTM=,                                C
2                 SAMPTR=,PAMHP=X'2'
3           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10 MODIFY PAMHP
3           B     *+4+L'$@@PAMHP
3           DC    XL(L'$@@PAMHP)'2'
2           CNOP  2,4
2           LA    15,*+22
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000'9
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                       CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'REWR'
2           L     1,4(1)                   R1=A(RE)
2           LEA@@FB ERRADDR=(R4),ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                     R15=4X'FF'
3           LA    14,*+12+2+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+2 TEST RC
3           BRE   14                       RC OK
3           BR    R4                       RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@RHLD Satz lesen mit Satzsperre

Dieser Makro liest einen Satz oder Block und sperrt ihn.

Bei ISAM- und DAM-Dateien wird über den angegebenen Schlüssel, bei PAM-Dateien über die Blocknummer zugegriffen.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RHLD	[[re],[db],[ar],[fa],[si],[kb],[ke]] [,SAVE=adresse2] [,POPE1=erg1] [,TOPE1=erg1] [,POPE2=erg2] [,TOPE2=erg2] [,POPEWTM={wartezeit / BLANK}] [,TOPEWTM={wartezeit / BLANK}] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindexes.

kb *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1=erg1
TOPE1=erg1 } Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=S Anlegen eines READ-LOCK.

=BLANK Anlegen eines WRITE-LOCK.

POPE2=erg2
TOPE2=erg2 } Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=N Die Anzahl der Primärschlüssel zu einem Sekundärschlüssel-Wert wird ermittelt.

=BLANK Die Anzahl der Primärschlüssel wird nicht ermittelt.

POPEWTM
TOPEWTM } Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).

=wartezeit $0 \leq wartezeit \leq 999$.

=BLANK Es gilt die globale Wartezeit der Session (*TIME*-Anweisung in LEASY-MAINTASK).

PPAMHP
TPAMHP } PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Nummer des Blocks, der gelesen werden soll.

PIDE=ide
TIDE=ide } Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@RHLD PAMRE,PAMDB,MF=(E,ADDRLST),                          -
                  POPEWTM=22,                                             -
                  TPAMHP=X'1',                                            -
                  ERRADDR=ERROR
1           LEA@CALL 'RHLD',PAMRE,PAMDB,,,,,,                             C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                            C
1                 PPAMHP=,POPEWTM=22,PIDE=,                               C
1                 POPE1=,POPE2=,                                          C
1                 TPAMHP=X'1',TOPEWTM=,TIDE=,                             C
1                 TOPE1=,TOPE2=,                                          C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           LA    14,PAMRE                  R14=A(RE)
2           LEA@@KO OPCD=RHLD,KORR=PERM,                                  C
2                 OPE1=,OPE2=,                                            C
2                 OPEOM=,OPELOG=,OPEWTM=22,                               C
2                 SAMPTR=,PAMHP=,IDE=
3           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10 MODIFY OPEWTIM
3           B     *+4+((L'$@@OPWTM+1)/2)*2
3           DC    ZL(L'$@@OPWTM)'22'
2           MVC   *+14(L'$@@RE),0(14)        TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)                 TEMP. RE
2           LEA@@KO OPCD=RHLD,KORR=TEMP,                                  C
2                 OPE1=,OPE2=,                                            C
2                 OPEOM=,OPELOG=,OPEWTM=,                                 C
2                 SAMPTR=,PAMHP=X'1',IDE=
3           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10 MODIFY PAMHP
3           B     *+4+L'$@@PAMHP
3           DC    XL(L'$@@PAMHP)'1'
2           CNOP  2,4
2           LA    15,*+70
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
```

```
2          BZ    *+8
2          O     14,=A(X'80000000')
2          ST    14,4(1)                              MODIFY RE
2          LA    15,PAMDB
2          TM    8(1),X'80'
2          BZ    *+8
2          O     15,=A(X'80000000')
2          ST    15,8(1)                              MODIFY DB
2          LM    14,15,*+6
2          BR    15                        CALL LEASY
2          DC    A(*+12)
2          DC    V(LEASY)
2 *
2          DC    C'RHLD'
2          L     1,4(1)                    R1=A(RE)
2          LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3          XR    15,15
3          BCTR  15,0                      R15=4X'FF'
3          LA    14,*+12+4+((L'$@@RCCC+1)/2)*2 R14=RETURN-ADDR.
3          CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4 TEST RC
3          BRE   14                        RC OK
3          B     ERROR                  RC: ERROR
3          DC    CL(L'$@@RCCC)'000'
```

LEA@RNHD Nachfolger lesen mit Satzsperre

Dieser Makro liest sequenziell den nächsten Satz oder Block der angegebenen Datei. Bei SAM-Dateien wird in Richtung Dateiende, bei ISAM-, DAM- oder PAM-Dateien in aufsteigender Richtung des Primär- oder Sekundärschlüssels gelesen. Wird die Operation erfolgreich durchgeführt, so wird der Satz bzw. Block gesperrt. Siehe Seite 176ff.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RNHD	[[re],[db],[ar],[fa]] [,SAVE=adresse2] [,POPE1=erg1] [,TOPE1=erg1] [,POPE2=erg2] [,TOPE2=erg2] [,POPEWTM={wartezeit | BLANK}] [,TOPEWTM={wartezeit | BLANK}] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar — *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa — *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1=erg1
TOPE1=erg1 — Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=S Anlegen eines READ-LOCK.

=BLANK Anlegen eines WRITE-LOCK.

POPE2=erg2
TOPE2=erg2
} Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).

=L Gesperrte Sätze werden überlesen

=BLANK Gesperrte Sätze werden nicht überlesen

POPEWTM
TOPEWTM
} Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).

=wartezeit $0 \leq wartezeit \leq 999$.

=BLANK Es gilt die globale Wartezeit der Session

(*TIME*-Anweisung in LEASY-MAINTASK).

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@RNHD MF=(E,ADDRLIST),                                       -
                  TOPEWTM=12,                                               -
                  ERRADDR=ERROR,                                            -
                  ERRCODE=(LOO3)
1           LEA@CALL 'RNHD',,,,,,,,                                         C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                              C
1                 POPEWTM=,PIDE=,POPE1=,POPE2=,                             C
1                 TOPEWTM=12,TIDE=,TOPE1=,TOPE2=,                           C
1                 ERRCODE=(LOO3),ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           L     14,4(1)                 R14=A(RE)
2           MVC   *+14(L'$@@RE),0(14)        TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)                TEMP. RE
2           LEA@@KO OPCD=RNHD,KORR=TEMP,                                    C
2                 OPE1=,OPE2=,                                              C
2                 OPEOM=,OPELOG=,OPEWTM=12,                                 C
2                 SAMPTR=,PAMHP=,IDE=
3           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10     MODIFY OPEWTIM
3           B     *+4+((L'$@@OPWTM+1)/2)*2
3           DC    ZL(L'$@@OPWTM)'12'
2           CNOP  2,4
2           LA    15,*+50
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                 MODIFY RE
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'RNHD'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=(LOO3),R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+16+4+16+((L'$@@RCCC+1)/2)*2      R14=RETURN ADDRESS
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+12      TEST RC
3           BRE   14                      RC OK
3           B     *+4+((L'$@@RCCC+1)/2)*2 RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
3           CLC   $@@RCLC-$@@RE(4,1),*+12      FEASIBLE RETURN CODE?
3           BRE   14                      YES, FEASIBLE
3           B     *+4+4
3           DC    C'LOO3'
3           B     ERROR                   RC: ERROR
```

LEA@RNXT Nachfolger lesen

Dieser Makro liest sequenziell den nächsten Satz oder Block der angegebenen Datei. Bei SAM-Dateien wird in Richtung Dateiende, bei ISAM-, DAM- oder PAM-Dateien in aufsteigender Richtung des Primär- oder Sekundärschlüssels gelesen. Siehe Seite 176ff.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RNXT	[[re],[db],[ar],[fa]] [,SAVE=adresse2] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide } Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@RNXT MF=(E,ADDRLIST),                                       -
                  ERRADDR=ERROR
1           LEA@CALL 'RNXT',,,,,,,,                                         C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                              C
1                 PIDE=,TIDE=,                                              C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           CNOP  2,4
2           LA    15,*+34
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                        CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'RNXT'
2           L     1,4(1)                    R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                      R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                        RC OK
3           B     ERROR                     RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@RPHD Vorgänger lesen mit Satzsperre

Dieser Makro liest den nächsten Satz oder Block der angegebenen Datei. Bei SAM-Dateien wird in Richtung Dateianfang, bei ISAM-, DAM- oder PAM-Dateien in absteigender Richtung des Primär- oder Sekundärschlüssels gelesen. Siehe Seite 179ff.

Wird die Operation erfolgreich durchgeführt, so wird der Satz bzw. Block gesperrt.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RPHD	[[re],[db],[ar],[fa]] [,SAVE=adresse2] [,POPE1=erg1] [,TOPE1=erg1] [,POPE2=erg2] [,TOPE2=erg2] [,POPEWTM={wartezeit | BLANK}][,TOPEWTM={wartezeit | BLANK}] [,PIDE] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar — *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa — *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des *RE*-Bereichs.

POPE1=erg1
TOPE1=erg1 — Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=S	Anlegen eines READ-LOCK.
=BLANK	Anlegen eines WRITE-LOCK.
POPE2=erg2 TOPE2=erg2	Operationscodeergänzung *OPE2* (*P*=permanent, *T*=temporär).
=L	Gesperrte Sätze werden überlesen
=BLANK	Gesperrte Sätze werden nicht überlesen
POPEWTM TOPEWTM	Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).
=wartezeit	0 ≤ *wartezeit* ≤ 999.
=BLANK	Es gilt die globale Wartezeit der Session (*TIME*-Anweisung in LEASY-MAINTASK).
PIDE=ide TIDE=ide	Identifikation für DCAM (*P*=permanent, *T*=temporär).
=symb	Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.
=(r)	Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.
ERRCODE=(fehlercode,...)	Liste der zu tolerierenden Fehlercodes; 1 bis 8 Zeichen.
=adresse3	Symbolische Adresse für Fehlerbehandlung.
ERRADDR=adresse4	Adresse (*symb* oder (*r*)); Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@RPHD MF=(E,ADDRLIST),                                       -
                  POPEWTM=3,                                                -
                  ERRADDR=ERROR
1           LEA@CALL 'RPHD',,,,,,,,                                         C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                              C
1                 POPEWTM=3,PIDE=,POPE1=,POPE2=,                            C
1                 TOPEWTM=,TIDE=,TOPE1=,TOPE2=,                             C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           L     14,4(1)                 R14=A(RE)
2           LEA@@KO OPCD=RPHD,KORR=PERM,                                    C
2                 OPE1=,OPE2=,                                              C
2                 OPEOM=,OPELOG=,OPEWTM=3,                                  C
2                 SAMPTR=,PAMHP=,IDE=
3           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10     MODIFY OPEWTIM
3           B     *+4+((L'$@@OPWTM+1)/2)*2
3           DC    ZL(L'$@@OPWTM)'3'
2           CNOP  2,4
2           LA    15,*+34
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'RPHD'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                      RC OK
3           B     ERROR                   RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@RPRI Vorgänger lesen

Dieser Makro liest sequenziell den nächsten Satz oder Block der angegebenen Datei. Bei SAM-Dateien wird in Richtung Dateianfang, bei ISAM-, DAM- oder PAM-Dateien in absteigender Richtung des Primär- oder Sekundärschlüssels gelesen. Siehe Seite 179.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@RPRI	[[re],[db],[ar],[fa]] [,SAVE=adresse2] [,PIDE=ide][,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re — *symb* oder (*r*) Adresse des Verständigungsbereichs.

db — *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar — *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa — *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

SAVE=adresse2 — Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb — Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) — Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes; 1 bis 8 Zeichen.

=adresse3 — Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@RPRI MF=(E,ADDRLIST)
1           LEA@CALL 'RPRI',,,,,,,,                                        C
1                 MF=(E,ADDRLIST),SAVE=,PRE=L,                             C
1                 PIDE=,TIDE=,                                             C
1                 ERRCODE=,ERRADDR=
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           CNOP  2,4
2           LA    15,*+34
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'RPRI'
2           L     1,4(1)                  R1=A(RE)
```

LEA@SETL Dateizeiger positionieren

Dieser Makro positioniert einen intern geführten Dateizeiger auf einen angegebenen Satz oder Block. Zusätzlich wird der beim Operanden *si* angegebene Index und bei Angabe von 8 Operanden ein Schlüsselintervall für nachfolgende *LEA@RNXT/LEA@RPRI* Operationen eingestellt. Siehe „SETL Dateizeiger positionieren“ auf Seite 181.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@SETL	[[re],[db],[ar],[fa],[si],[kb],[ke]] [,SAVE=adresse2] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PSAMPTR=X'samzeiger'] [,TSAMPTR=X'samzeiger'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Sekundärindex).

'*string*' Name eines Sekundärindexes.

kb *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselanfang).

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

PPAMHP
TPAMHP
} PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Nummer des Blocks, der gelesen werden soll.

PSAMPTR
TSAMPTR
} Wiedergewinnungsadresse eines Satzes in einer SAM-Datei im 24- oder 31-Bit-Format (*P*=permanent, *T*=temporär).

=X'bbbbbbrr' 24-Bit-Format

=X'bbbbbbbbrrrrrrrr'
31-Bit-Format

(*b* = Blocknummer, *r* = Satznummer innerhalb des Blocks)

PIDE=ide
TIDE=ide
} Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes; 1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*)); Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LA    R4,ADDRLIST
  *
            LEA@SETL ,,,,'MAINITEM',MF=(E,(R4)),                          -
                  TSAMPTR=X'1001',                                        -
                  ERRADDR=ERROR
1           LEA@CALL 'SETL',,,,,'MAINITEM',,,                             C
1                 MF=(E,(R4)),SAVE=,PRE=L,                                C
1                 PPAMHP=,PSAMPTR=,PIDE=,                                 C
1                 TPAMHP=,TSAMPTR=X'1001',TIDE=,                          C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LR    1,R4
2           L     14,4(1)                 R14=A(RE)
2           MVC   *+14(L'$@@RE),0(14)        TEMP. RE
2           LA    14,*+8
2           B     *+4+L'$@@RE
2           DS    CL(L'$@@RE)                TEMP. RE
2           LEA@@KO OPCD=SETL,KORR=TEMP,                                  C
2                 OPE1=,OPE2=,                                            C
2                 OPEOM=,OPELOG=,OPEWTM=,                                 C
2                 SAMPTR=X'1001',PAMHP=,IDE=
3           MVC   $@@SPTR-$@@RE(L'$@@SPTR,14),*+10     MODIFY SAMPTR
3           B     *+4+L'$@@SPTR
3           DC    XL(L'$@@SPTR)'1001'
2           CNOP  2,4
2           LA    15,*+70
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                 MODIFY RE
2           LA    15,*+38
2           TM    20(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,20(1)                MODIFY SI
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+20)
2           DC    V(LEASY)
2 *
2           DC    C'SETL'
2           DC    C'MAINITEM'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                      RC OK
3           B     ERROR                   RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

LEA@STOR Satz einfügen

Dieser Makro fügt einen Satz oder Block in die angegebene Datei ein - unabhängig davon, ob der Satz bzw. Block schon existiert oder nicht. Die eingefügten Sätze oder Blöcke bleiben bis Transaktionsende gesperrt. Siehe „STOR Satz einfügen“ auf Seite 183.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@STOR	[[re],[db],[ar],[us]] [,SAVE=adresse2] [,POPEWTM={wartezeit / BLANK}] [,TOPEWTM={wartezeit / BLANK}] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) / adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

us *symb* oder (*r*) Adresse des *USER*-Bereichs.

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPEWTM, TOPEWTM } Wartezeit auf gesperrte Sätze (*P*=permanent, *T*=temporär).

=wartezeit	$0 \leq wartezeit \leq 999$.
=BLANK	Es gilt die globale Wartezeit der Session (*TIME*-Anweisung in LEASY-MAINTASK).
PPAMHP TPAMHP	PAM-Blocknummer (*P*=permanent, *T*=temporär).
=X'bbbbbb'	Nummer des Blocks, der gelesen werden soll.
PIDE=ide TIDE=ide	Identifikation für DCAM (*P*=permanent, *T*=temporär).
=symb	Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.
=(r)	Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.
ERRCODE=(fehlercode,...)	Liste der zu tolerierenden Fehlercodes; 1 bis 8 Zeichen.
=adresse3	Symbolische Adresse für Fehlerbehandlung.
ERRADDR=adresse4	Adresse (*symb* oder (*r*)); Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@STOR L@@RE,L@@DB1,L@@AR,                                  -
                  POPEWTM=3
1           LEA@CALL 'STOR',L@@RE,L@@DB1,L@@AR,                           C
1                 MF=,SAVE=,PRE=L,                                        C
1                 PPAMHP=,POPEWTM=3,PIDE=,                                C
1                 TPAMHP=,TOPEWTM=,TIDE=,                                 C
1                 ERRCODE=,ERRADDR=
2           LEA@@BP GRU=AKT,NAM=
2           LA    14,L@@RE                R14=A(RE)
2           LEA@@KO OPCD=STOR,KORR=PERM,                                  C
2                 OPE1=,OPE2=,                                            C
2                 OPEOM=,OPELOG=,OPEWTM=3,                                C
2                 SAMPTR=,PAMHP=,IDE=
3           MVC   $@@OPWTM-$@@RE(L'$@@OPWTM,14),*+10     MODIFY OPEWTIM
3           B     *+4+((L'$@@OPWTM+1)/2)*2
3           DC    ZL(L'$@@OPWTM)'3'
2           CNOP  2,4
2           LA    15,L@@RE
2           ST    15,*+46                 A(RE)
2           LA    15,L@@DB1
2           ST    15,*+42                 A(DB)
2           LA    15,L@@AR
2           ST    15,*+38                 A(AR)
2           OI    *+34,X'80'              LAST PARAM-ADDRESS
2           LM    14,15,*+10
2           LA    1,*+14
2           BR    15                      CALL LEASY
2           DC    A(*+28)
2           DC    V(LEASY)
2 *
2           DC    A(*+16)           PTR(OP)
2           DC    A(0)              PTR(RE)
2           DC    A(0)              PTR(DB)
2           DC    A(0)              PTR(AR)
2           DC    C'STOR'
2           L     1,4(1)                  R1=A(RE)
```

LEA@TOLR Fehlercodes behandeln

Dieser Makro generiert einen Codeteil, der die bei *ERRCODE* angegebene Liste der zu tolerierenden Fehlercodes aussondert. Wird ein solcher Fehlercode entdeckt, so wird hinter den Aktionsmakro zurück verzweigt. Jeder andere Fehlercode löst einen Sprung zur Adresse adresse4 aus, die beim Aktionsmakro unter *ERRADDR* anzugeben ist.

In der Fehlercodeliste können beliebig viele Fehlercodes angegeben werden. Über das Mehrzweckregister 14 kann in der Fehlerroutine hinter das fehlerauslösende Aktionsmakro zurückgesprungen werden.

Operation	Operanden
`LEA@TOLR`	`ERRCODE=(fehlercode,...)`

ERRCODE=(fehlercode,...)

Liste der zu tolerierenden Fehlercodes:

1 bis 8 Zeichen.

Beispiel

```
            LEA@CLFL L@@RE,L@@DB1,                                          -
                  ERRCODE=TOLR,                                             -
                  ERRADDR=ERROR
1           LEA@CALL 'CLFL',L@@RE,L@@DB1,                                   C
1                 MF=,PRE=L,                                                C
1                 ERRCODE=TOLR,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           CNOP  2,4
2           LA    15,L@@RE
2           ST    15,*+38                 A(RE)
2           LA    15,L@@DB1
2           ST    15,*+34                 A(DB)
2           OI    *+30,X'80'              LAST PARAM-ADDRESS
2           LM    14,15,*+10
2           LA    1,*+14
2           BR    15                      CALL LEASY
2           DC    A(*+24)
2           DC    V(LEASY)
2 *
2           DC    A(*+12)           PTR(OP)
2           DC    A(0)              PTR(RE)
2           DC    A(0)              PTR(DB)
2           DC    C'CLFL'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=TOLR,R14=ON
3           LA    14,*+16+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+12+4         TEST RC
3           BRE   14                      RC OK
3           LA    15,ERROR                R15=A(ERROR-ROUTINE)
3           B     TOLR
3           DC    CL(L'$@@RCCC)'000'
  *
  TOLR      LEA@TOLR ERRCODE=(091LL118,043)
1           LEA@@BP GRU=AKT,NAM=TOLR
1           LEA@@FB ERRCODE=(091LL118,043),ERRADDR=(15),R14=OFF
2 TOLR      CLC   $@@RCCC-$@@RE(8,1),*+12      FEASIBLE RETURN CODE?
2           BRE   14                      YES, FEASIBLE
2           B     *+4+8
2           DC    C'091LL118'
2           CLC   $@@RCCC-$@@RE(3,1),*+12      FEASIBLE RETURN CODE?
2           BRE   14                      YES, FEASIBLE
2           B     *+4+4
2           DC    C'043'
2           BR    15                      RC: ERROR
```

LEA@UNLK Satzsperre aufheben

Dieser Makro hebt Satzsperren auf. Es können nur Sperren für Sätze oder Blöcke aufgehoben werden, die innerhalb der Transaktion gesperrt, aber nicht verändert wurden.

Nicht mehr benötigte Sperren sollte man aufheben, um Wartezeiten auf gesperrte Sätze bzw. Blöcke zu verkürzen. Bei Transaktionsende werden alle Sperren automatisch freigegeben. Siehe „UNLK Satzsperre aufheben“ auf Seite 184.

Die Operanden *MF* und *PRE* sind erlaubt.

Operation	Operanden
LEA@UNLK	[[re],[db],[ar],[fa],[si],[kb],[ke]] [,SAVE=adresse2] [,POPE1=erg1] [,TOPE1=erg1] [,PPAMHP=X'pamblocknummer'][,TPAMHP=X'pamblocknummer'] [,PIDE=ide] [,TIDE=ide] [,ERRCODE={(fehlercode,...) | adresse3} ,ERRADDR=adresse4] [,ERRADDR=adresse4]

re *symb* oder (*r*) Adresse des Verständigungsbereichs.

db *symb* oder (*r*) Adresse der Dateikennung.

'*string*' Dateikennung.

ar *symb* oder (*r*) Adresse des Ein-Ausgabebereichs (Satzzone).

fa *symb* oder (*r*) Adresse des *FA*-Bereichs (Feldauswahl).

'*string*' Kennzeichen für die Feldauswahl.

si *symb* oder (*r*) Adresse des *SI*-Bereichs (Schlüsselanfang).

'*string*' Name eines Sekundärindex.

kb *symb* oder (*r*) Adresse des *KB*-Bereichs (Schlüsselanfang).

ke *symb* oder (*r*) Adresse des *KE*-Bereichs (Schlüsselende).

SAVE=adresse2 Adresse (*symb* oder (*r*)) des Zwischenbereichs für die Sicherung des Verständigungsbereichs.

POPE1=erg1
TOPE1=erg1
Operationscodeergänzung *OPE1* (*P*=permanent, *T*=temporär).

=U Auch modifizierte Sätze werden freigegeben.

=BLANK Nur Sätze, die nicht verändert wurden, können freigegeben werden.

PPAMHP
TPAMHP
PAM-Blocknummer (*P*=permanent, *T*=temporär).

=X'bbbbbb' Nummer des Blocks, dessen Sperre aufgehoben werden soll.

PIDE=ide
TIDE=ide
Identifikation für DCAM (*P*=permanent, *T*=temporär).

=symb Die symbolische Adresse zeigt auf ein 8 Byte langes Feld, das eine Transaktions-Identifikation enthält.

=(r) Das Mehrzweckregister *r* enthält die Adresse eines 8 Byte langen Feldes, das eine Transaktions-Identifikation enthält.

ERRCODE=(fehlercode,...)
Liste der zu tolerierenden Fehlercodes;

1 bis 8 Zeichen.

=adresse3 Symbolische Adresse für Fehlerbehandlung.

ERRADDR=adresse4 Adresse (*symb* oder (*r*));

Sprungziel, falls der auftretende Fehlercode nicht in der Liste der zu tolerierenden Fehlercodes (*ERRCODE*) enthalten ist.

Beispiel

```
            LEA@UNLK MF=(E,ADDRLIST),                                       -
                  SAVE=T@@RE,                                               -
                  TPAMHP=X'3',                                              -
                  ERRADDR=ERROR
1           LEA@CALL 'UNLK',,,,,,,,                                         C
1                 MF=(E,ADDRLIST),SAVE=T@@RE,PRE=L,                         C
1                 PPAMHP=,PIDE=,                                            C
1                 TPAMHP=X'3',TIDE=,                                        C
1                 ERRCODE=,ERRADDR=ERROR
2           LEA@@BP GRU=AKT,NAM=
2           LA    1,ADDRLIST
2           L     14,4(1)                 R14=A(RE)
2           MVC   T@@RE(L'$@@RE),0(14)        SAVE=TEMP. RE
2           LA    14,T@@RE
2           LEA@@KO OPCD=UNLK,KORR=TEMP,                                    C
2                 OPE1=,OPE2=,                                              C
2                 OPEOM=,OPELOG=,OPEWTM=,                                   C
2                 SAMPTR=,PAMHP=X'3',IDE=
3           MVC   $@@PAMHP-$@@RE(L'$@@PAMHP,14),*+10     MODIFY PAMHP
3           B     *+4+L'$@@PAMHP
3           DC    XL(L'$@@PAMHP)'3'
2           CNOP  2,4
2           LA    15,*+50
2           TM    0(1),X'80'
2           BZ    *+8
2           O     15,=A(X'80000000')
2           ST    15,0(1)
2           TM    4(1),X'80'
2           BZ    *+8
2           O     14,=A(X'80000000')
2           ST    14,4(1)                 MODIFY RE
2           LM    14,15,*+6
2           BR    15                      CALL LEASY
2           DC    A(*+12)
2           DC    V(LEASY)
2 *
2           DC    C'UNLK'
2           L     1,4(1)                  R1=A(RE)
2           LEA@@FB ERRADDR=ERROR,ERRCODE=,R14=ON
3           XR    15,15
3           BCTR  15,0                    R15=4X'FF'
3           LA    14,*+12+4+((L'$@@RCCC+1)/2)*2    R14=RETURN-ADDR.
3           CLC   $@@RCCC-$@@RE(L'$@@RCCC,1),*+8+4         TEST RC
3           BRE   14                      RC OK
3           B     ERROR                   RC: ERROR
3           DC    CL(L'$@@RCCC)'000'
```

11.5 Makros für die Auswertung der Currency Information CI

Zur leichteren Auswertung der Currency Information CI, die bei Angabe von *OPE1=F* zurückgeliefert wird, stellt LEASY Makroaufrufe zu Verfügung, die Strukturen als DSECT's generieren. Folgende Makroaufrufe sind möglich:

– LEA@@DDL

– LEA@@DSI

– LEA@@DPL

– LEA@@DRI.

Struktur des Bereichs ci-inf

Die Currency Information *CI*, die bei Angabe von *OPE1=F* zurückgeliefert wird, ist im Bereich *ci-inf* wie folgt strukturiert:

Bereich ci-inf bei OPE2=C, BLANK, O, T, W

Für alle ausgewählten Dateien ist jeweils ein Teilbereich vorhanden, der die Struktur *LEA@@DDL* in der Länge *L@@DL1* enthält. *LEA@@DDL* enthält allgemeine Informationen zur Datei, z.B. Dateinamen, Dateityp.

LEA@@DDL	Teilbereich Datei 1
LEA@@DDL	Teilbereich Datei 2
. . .	. . .
LEA@@DDL	Teilbereich Datei n

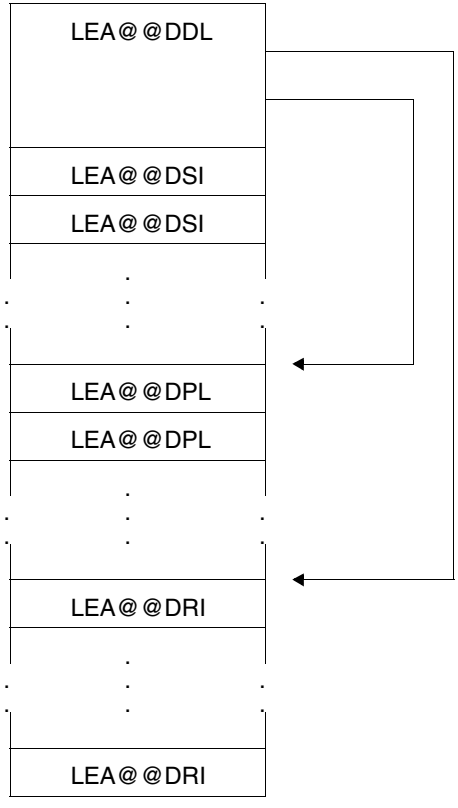
Bereich ci-inf bei OPE2=S

Am Anfang des Bereichs *ci-inf* befindet sich die Struktur *LEA@@DDL*. Für Dateien ohne *SI*-Definition ist diese Struktur die Einzige. Sie enthält allgemeine Informationen zur Datei, z.B. Dateiname, Dateityp.

Bei einer Datei mit *SI*-Definitionen folgt für jede *SI*-Definition dieser Datei jeweils eine Struktur *LEA@@DSI*, die allgemeine Angaben zum Sekundärindex enthält, z.B. Name, Anzahl Teilschlüssel.

Daran schließen sich Strukturen *LEA@@DPL* an, wobei für jeden einzelnen für diese Datei definierten Teilschlüssel eine Struktur vorhanden ist. Darin enthalten sind Angaben über Länge und Position der Teilschlüssel.

Zum Schluss folgt noch für jede für diese Datei definierte Satzart eine Struktur *LEA@@DRI*, die die Satzartbezeichnung und deren Länge enthält.



Makroaufrufe

LEA@@DDL

Dieser Makroaufruf generiert eine DSECT *LEA@@DDL*. Alle darin enthaltenen Namen haben das Präfix *L@@D*. Das Präfix kann mit dem Operanden *PRE* geändert werden.

Operation	Operanden
LEA@@DDL	[PRE=präfix]

Beispiel

```
            LEA@@DDL
1 *
1 *         LAYOUT EINES ELEMENTS ZUR DATEIBESCHREIBUNG
1 *
1 L@@DDL    DSECT
1 *
1            DS    A
1 *
1 L@@DLOGN DS    CL8                LOGISCHER DATEINAME
1 *
1           DS    A
1 *
1 L@@DPAD   DS    X                  PAD-FAKTOR
1 *
1 L@@DIND   DS    X                  INDIKATOR
1 L@@DINO   EQU   X'00'                WEDER AIM- NOCH BIM-SICHERUNG
1 L@@DIA    EQU   X'80'                AIM-SICHERUNG
1 L@@DIB    EQU   X'08'                BIM-SICHERUNG
1 L@@DISA   EQU   X'02'                VERKUERZTE AIM-SAETZE
1 L@@DRDP   EQU   X'20'                READPASS ANGEGEBEN
1 L@@DWRP   EQU   X'10'                WRITEPASS ANGEGEBEN
1 *
1           DS    2X
1 *
1 L@@DFN    DS    CL54               DATEINAME
1 *
1 L@@DFT    DS    C                  LEASY DATEI-TYP
1 L@@DFTS   EQU   'S'                  STAMMDATEI
1 L@@DFTM   EQU   'M'                  MODELLDATEI
1 L@@DFTF   EQU   'F'                  FREMDDATEI
1 L@@DFTT   EQU   'T'                  TEMPORAERDATEI
1 *
1 L@@DFCB   DS    X                  FCBTYPE
1 L@@DFCBI EQU   X'40'                 ISAM
1 L@@DFCBS EQU   X'00'                 SAM
1 L@@DFCBP EQU   X'C0'                 PAM
1 L@@DFCBD EQU   X'C1'                 DAM
1 *
1 L@@DRSM   DS    H                  MAXIMALE SATZLAENGE
1 *
1 L@@DKEYL DS    X                  SCHLUESSELLAENGE
1 *
```

```
1 L@@D#SI  DS    X                ANZAHL DER SI-DEFINITIONEN
1 *                                = ANZAHL DER ELEMENTE LEA@@DSI
1 L@@DNOSI EQU   X'00'              KEINE SI-DEFINITIONEN
1 *
1 L@@DL1   EQU   *-L@@DDL         LAENGE VON LEA@@DDL OHNE SI-DEF.
1 *
1          DS    X
1 *
1 L@@DSILM DS    X                MAXIMALE SI-SCHLUESSELLAENGE
1 *
1 L@@DPPLD DS    Y                DISTANZ DES 1. LEA@@DPL-ELEMENTS VOM
1 *                                BEGINN VON LEA@@DDL
1 *
1 L@@DRPOS DS    H                SATZARTENFELD-POSITION - 1
1 *
1 L@@DRLEN DS    H                SATZARTENFELD-LAENGE
1 *
1 L@@D#RID DS    H                ANZAHL DEFINIERTER SATZARTEN
1 *
1 L@@DPRID DS    Y                DISTANZ DES 1. LEA@@DRI-ELEMENTS VOM
1 *                                BEGINN VON LEA@@DDL
1 *
1 L@@DL2   EQU   *-L@@DDL         LAENGE VON LEA@@DDL MIT SI-DEF.
1 *
1 L@@DSI   EQU   *                BEGINN DES 1. ELEMENTS LEA@@DSI
1 *
1          CSECT
1 *
```

LEA@@DSI

Dieser Makroaufruf generiert eine DSECT *LEA@@DSI*. Alle darin enthaltenen Namen haben das Präfix *L@@S*. Das Präfix kann mit dem Operanden *PRE* geändert werden.

Operation	Operanden
LEA@@DSI	[PRE=präfix]

Beispiel

```
           LEA@@DSI
1 *
1 *        LAYOUT EINER SEKUNDAERINDEX-DEFINITION
1 *
1 L@@SSI   DSECT
1 *
1 L@@SKEY  DS    CL8                SI-NAME
1 *
1          DS    X
1 *
1 L@@SIUB  DS    X                  SI-UNTERDRUECKUNGSBYTE
1 *
1 L@@SIND  DS    X                  INDIKATORBYTE
1 L@@SDUP  EQU   X'01'                DUPKEY = JA
1 L@@SUPD  EQU   X'02'                KEYUPD = JA
1 L@@SIIUB EQU   X'04'                SI-UNTERDRUECKUNGSBYTE GUELTIG
1 *
1 L@@S#SIP DS    HL1                ANZAHL SCHLUESSELTEILE
1 *                                   = ANZAHL ELEMENTE LEA@@DPL FUER SI
1 *
1 L@@SKLSI DS    X                  LAENGE SEKUNDAERSCHLUESSEL
1 *
1          DS    X
1 *
1 L@@SPPLD DS    Y                  ADRESSPTR. AUF 1.EINTRAG LEA@@DPL
1 *                                   BZGL. ANF.ADR. L@@DPPLD IN LEA@@DDL
1 *
1 L@@SREP  DS    H                  WIEDERHOLUNGSFAKTOR
1 *
1 L@@S#RID DS    H                  ANZAHL SATZARTENDEFINITIONEN
1 *
1 L@@SPRID DS    Y                  ADRESSPTR. AUF 1.EINTRAG LEA@@DRI
1 *                                   BZGL. ANF.ADR. L@@DPRID IN LEA@@DDL
1 *
1 L@@SL    EQU   *-L@@SSI           LAENGE EINES EINTRAGES LEA@@DSI
1 *
1          CSECT
1 *
```

LEA@@DPL

Dieser Makroaufruf generiert eine DSECT *LEA@@DPL*. Alle darin enthaltenen Namen haben das Präfix *L@@P*. Das Präfix kann mit dem Operanden *PRE* geändert werden.

Operation	Operanden
LEA@@DPL	[PRE=präfix]

Beispiel

```
            LEA@@DPL
1 *
1 *         LAYOUT EINES ELEMENTS ZUR TEILSCHLUESSELDEFINITION
1 *
1 L@@PPL    DSECT
1 *
1 L@@PPOS   DS    AL2                TEILSCHLUESSEL-ANFANGSPOSITION - 1
1 *
1 L@@PLEN   DS    HL1                TEILSCHLUESSEL-LAENGE - 1
1 *
1 L@@PDIST  DS    HL2                TEILSCHLUESSEL-DISTANZ
1 *
1 L@@PL     EQU   *-L@@PPL
1 *
1           CSECT
1 *
```

LEA@@DRI

Dieser Makroaufruf generiert eine DSECT *LEA@@DRI*. Alle darin enthaltenen Namen haben das Präfix *L@@R*. Das Präfix kann mit dem Operanden *PRE* geändert werden.

Operation	Operanden
LEA@@DRI	[PRE=präfix]

Beispiel

```
            LEA@@DRI
1 *
1 *         LAYOUT EINES ELEMENTS ZUR SATZARTENDEFINITION
1 *
1 L@@RRI    DSECT
1 *
1 L@@RLRID DS    X                  LAENGE DER SATZARTENDEFINITION
1 *
1 L@@RRID  EQU   *                  SATZARTENDEFINITION
1 *
1           CSECT
1 *
```

12 Anwendungsbeispiele

12.1 COBOL-Programm

Dieses Beispiel zeigt, wie LEASY in einem COBOL-Programm über die CALL-Schnittstelle aufgerufen wird.
Ablaufprotokolle, die den Einsatz der LEASY-Dienstprogramme in Verbindung mit dem Ablauf dieses Programms veranschaulichen, finden Sie in den Abschnitten „Ablaufprotokoll 1“ (Seite 364) und „Ablaufprotokoll 2“ (Seite 382).

Ausdruck des Quellprogramms

```
 ID DIVISION.
 PROGRAM-ID. PERSDAT.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES.
     DATE-ISO4 IS DATE4
     TERMINAL IS DSS.
 DATA DIVISION.
* IN DIESEM PROGRAMM WERDEN ZWEI DATEIEN UEBER LEASY BEARBEITET:
* LEASY-DATEIKATALOG: LCAT
* DATEIEN:            MITABDAT:  ISAM
*                                KEYPOS=1
*                                KEYLEN=4
*                                RECFORM=F
*                                RECSIZE=74
*                                SI: ABT,POS=45,LEN=5
*                                SI: NAME,POS=5,LEN=20
*                     PROTDAT:   SAM
*                                RECSIZE=45
 WORKING-STORAGE SECTION.
 01  LEASY-PARAMS.
*
```

```
    COPY LEASYKON. ------------------------------------------------  (1)
*>
*>PM L61004
*>PM L61006
*>PM L61029
*>
*>COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006
*>                              ALL RIGHTS RESERVED
*>          LEASY-OPERATIONSCODES.
 02        OP-CODES.
   05      OP-CATD          PIC  X(4)  VALUE  "CATD".
   05      OP-OPFL          PIC  X(4)  VALUE  "OPFL".
   05      OP-CLFL          PIC  X(4)  VALUE  "CLFL".
   05      OP-OPTR          PIC  X(4)  VALUE  "OPTR".
   05      OP-OPDB          PIC  X(4)  VALUE  "OPDB".
   05      OP-CLTR          PIC  X(4)  VALUE  "CLTR".
   05      OP-CLSE          PIC  X(4)  VALUE  "CLSE".
   05      OP-MARK          PIC  X(4)  VALUE  "MARK".
   05      OP-BACK          PIC  X(4)  VALUE  "BACK".
   05      OP-RDIR          PIC  X(4)  VALUE  "RDIR".
   05      OP-RNXT          PIC  X(4)  VALUE  "RNXT".
   05      OP-RPRI          PIC  X(4)  VALUE  "RPRI".
   05      OP-RHLD          PIC  X(4)  VALUE  "RHLD".
   05      OP-RNHD          PIC  X(4)  VALUE  "RNHD".
   05      OP-RPHD          PIC  X(4)  VALUE  "RPHD".
   05      OP-INSR          PIC  X(4)  VALUE  "INSR".
   05      OP-STOR          PIC  X(4)  VALUE  "STOR".
   05      OP-REWR          PIC  X(4)  VALUE  "REWR".
   05      OP-DLET          PIC  X(4)  VALUE  "DLET".
   05      OP-SETL          PIC  X(4)  VALUE  "SETL".
   05      OP-LOCK          PIC  X(4)  VALUE  "LOCK".
   05      OP-UNLK          PIC  X(4)  VALUE  "UNLK".
   05      OP-CINF          PIC  X(4)  VALUE  "CINF".
   05      OP-TERM          PIC  X(4)  VALUE  "TERM".
*>
*>          LEASY-OPENMODES
 02        OP-MODES.
   05      OP-INPUT         PIC  X     VALUE "1".
   05      OP-INPUT-SHARUPD PIC  X     VALUE "2".
   05      OP-INOUT         PIC  X     VALUE "3".
   05      OP-INOUT-SHARUPD PIC  X     VALUE "4".
   05      OP-REVERSE       PIC  X     VALUE "5".
   05      OP-UPDATE        PIC  X     VALUE "7".
   05      OP-OUTPUT        PIC  X     VALUE "8".
   05      OP-EXTEND        PIC  X     VALUE "9".
   05      OP-OUTIN         PIC  X     VALUE "A".
   05      OP-OUTIN-SHARUPD PIC  X     VALUE "B".
*>
```

```
*>          LEASY-USAGEMODES
*>          LEASY-USAGE ABKUERZUNGEN FUER OPE-OM
 02        USAGE-MODES-OPE-OM.
*>                                                        RETR
   05      OP-A              PIC  X     VALUE "A".
*>                                                        EXUP
   05      OP-B              PIC  X     VALUE "B".
*>                                                        PRUP
   05      OP-E              PIC  X     VALUE "E".
*>                                                        EXRT
   05      OP-G              PIC  X     VALUE "G".
*>                                                        PRRT
   05      OP-I              PIC  X     VALUE "I".
*>                                                        EXLD
   05      OP-L              PIC  X     VALUE "L".
*>                                                        EXLD
   05      OP-O              PIC  X     VALUE "O".
*>                                                        EXLD
   05      OP-Q              PIC  X     VALUE "Q".
*>                                                        ULRT
   05      OP-R              PIC  X     VALUE "R".
*>                                                        ULUP
   05      OP-U              PIC  X     VALUE "U".
*>                                                        EXRT
   05      OP-X              PIC  X     VALUE "X".
*>
*>          LEASY-USAGES FUER DB1, DB2, DB4
 02        USAGE-MODES.
*>          RETRIEVAL
   05      RETR              PIC  X(4)  VALUE  "RETR".
*>          UNLOCKED  RETRIEVAL
   05      ULRT              PIC  X(4)  VALUE  "ULRT".
*>          PROTECTED RETRIEVAL
   05      PRRT              PIC  X(4)  VALUE  "PRRT".
*>          PROTECTED RETRIEVAL REVERSE
   05      PRRR              PIC  X(4)  VALUE  "PRRR".
*>          EXCLUSIVE RETRIEVAL
   05      EXRT              PIC  X(4)  VALUE  "EXRT".
*>          EXCLUSIVE RETRIEVAL REVERSE
   05      EXRR              PIC  X(4)  VALUE  "EXRR".
*>          UPDATE
   05      UPDT              PIC  X(4)  VALUE  "UPDT".
*>          UNLOCKED UPDATE
   05      ULUP              PIC  X(4)  VALUE  "ULUP".
*>          PROTECTED UPDATE
   05      PRUP              PIC  X(4)  VALUE  "PRUP".
*>          EXCLUSIVE UPDATE
   05      EXUP              PIC  X(4)  VALUE  "EXUP".
```

```
      *>          SHARE LOAD
           05     LOAD             PIC  X(4)  VALUE  "LOAD".
      *>          PROTECTED LOAD
           05     PLOD             PIC  X(4)  VALUE  "PLOD".
      *>          EXCLUSIVE LOAD
           05     ELOD             PIC  X(4)  VALUE  "ELOD".
      *>          SHARE LOAD + UPDATE
           05     LDUP             PIC  X(4)  VALUE  "LDUP".
      *>          PROTECTED LOAD + UPDATE
           05     PLUP             PIC  X(4)  VALUE  "PLUP".
      *>          EXCLUSIVE LOAD + UPDATE
           05     ELUP             PIC  X(4)  VALUE  "ELUP".
      *>          EXCLUSIVE LOAD
           05     EXLD             PIC  X(4)  VALUE  "EXLD".
      *>          UNLOCKED RETRIEVAL
           05     ULRT             PIC  X(4)  VALUE  "ULRT".
      *>
      *>          LEASY-KONSTANTE FUER RE-BEREICH
           02    OPE-CONST.
      *>          WERTE FUER FELD OPE1
           05     OPE1-F           PIC  X     VALUE  "F".
           05     OPE1-R           PIC  X     VALUE  "R".
           05     OPE1-S           PIC  X     VALUE  "S".
           05     OPE1-U           PIC  X     VALUE  "U".
           05     OPE1-W           PIC  X     VALUE  "W".
      *>          WERTE FUER FELD OPE2
           05     OPE2-C           PIC  X     VALUE  "C".
           05     OPE2-L           PIC  X     VALUE  "L".
           05     OPE2-N           PIC  X     VALUE  "N".
           05     OPE2-O           PIC  X     VALUE  "O".
           05     OPE2-S           PIC  X     VALUE  "S".
           05     OPE2-T           PIC  X     VALUE  "T".
           05     OPE2-W           PIC  X     VALUE  "W".
      *>
       COPY LEASYPAR. ------------------------------------------------ (2)
000010*
      *>
      *>COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006
      *>                            ALL RIGHTS RESERVED
      *>          LEASY-OPERATIONSCODE
           02    OP               PIC  X(4)  VALUE SPACE.
      *>
      *>          LEASY-VERSTAENDIGUNGSBEREICH.
           02    RE.
             05  RE-K.
               10  RC-CODE.
                 15  RC-CC         PIC  X(3)  VALUE SPACE.
                 15  RC-KZ         PIC  X     VALUE SPACE.
```

```
          15  RC-LC           PIC  X(4)  VALUE SPACE.
           88 RC-OK                     VALUE "L000".
           88 RC-NOFIND                 VALUE "L001".
           88 RC-DUPEKY                 VALUE "L002".
           88 RC-EOF                    VALUE "L003".
           88 RC-SEQCHK                 VALUE "L004".
           88 RC-NOLOCK                 VALUE "L005".
           88 RC-LOCKED                 VALUE "L006".
           88 RC-DEADLOCK               VALUE "L007".
           88 RC-CHANGED                VALUE "L008".
           88 RC-RECNOLOCK              VALUE "L009".
           88 RC-RECLERROR              VALUE "L010".
           88 RC-MORE255                VALUE "L011".
           88 RC-NOACTREC               VALUE "L012".
           88 RC-KEYOUTRNG              VALUE "L013".
           88 RC-NOBIM                  VALUE "L014".
           88 RC-TSKDEADLOCK            VALUE "L015".
           88 RC-L016                   VALUE "L016".
        10    PASS            PIC  X(8)  VALUE SPACE.
        10    OPE.
          15  OPE-STX         PIC  X     VALUE SPACE.
          15  OPE-OM          PIC  X     VALUE SPACE.
          15  OPE-LOG         PIC  X     VALUE SPACE.
          15  FILLER          PIC  X(5)  VALUE SPACE.
        10    INT.
          15  SAMPTR          PIC  X(4)  VALUE LOW-VALUE.
          15  PAMHPNR         REDEFINES SAMPTR
                               PIC  9(8) COMP.
          15  SAMNUM          PIC  X(4)  VALUE LOW-VALUE.
          15  FILLER          REDEFINES SAMNUM
                               PIC  X(4)            .
        10    NUM             PIC  9(8)  VALUE ZERO.
        10    IDE             PIC  X(8)  VALUE SPACE.
     05      RE-LEASY-EXT.
        10    REOP            PIC  X(4)  VALUE SPACE.
        10    REDB            PIC  X(16) VALUE SPACE.
        10    L-OPT           PIC  X     VALUE "1".
        10    OPE1            PIC  X     VALUE SPACE.
        10    OPE2            PIC  X     VALUE SPACE.
        10    OPE-WTIME-GLOBAL           VALUE LOW-VALUE.
          15  OPE-WTIME       PIC  9(3).
        10    RC-LCE          PIC  X(5)  VALUE SPACE.
        10    U-PROT          PIC  X     VALUE SPACE.
*>
*>          LEASY-DATEILISTEN
 02         DB1.
    05      DB1-NAME         PIC  X(12) VALUE  SPACE.
*>
```

```
  02        DB3             PIC  X(12) VALUE  "ALL".
 *>
 *>          LEASY-KATALOGINFORMATIONEN
  02        CAT.
    05      CATNAME         PIC  X(24) VALUE SPACE.
    05      ZUSATZ          PIC  X(20) VALUE SPACE.
 *>
 *>          KLDS-FELDAUSWAHL
  02        FA              PIC  X(8)  VALUE  "ALL".
 *>          LEASY-SEKUNDAERINDEX
  02        SI              PIC  X(8)  VALUE SPACE.
 *>
*
 01  DB1-USED.                                          }  ----------------  (3)
     05  DB-MITABDAT PIC X(12) VALUE "MITABDAT".       }
     05  DB-PROTDAT  PIC X(12) VALUE "PROTDAT".        }
*
 01  DB4-OPFL.
     05  FILLER      PIC X(27) VALUE "((MITABDAT,4),(PROTDAT,9))".  (4)
*
 01  DB4-OPTR-UPDT.
     05  FILLER      PIC X(33) VALUE "((MITABDAT,UPDT),(PROTDAT,EX  (5)
-                                   "LD))".
*
 01  DB4-OPTR-LIST.
     05  FILLER      PIC X(17) VALUE "(MITABDAT,RETR)". ----------  (6)
 01  SI-ABT          PIC X(8)  VALUE "ABT". ----------------------  (7)
 01  SI-NAME         PIC X(8)  VALUE "NAME". ---------------------  (8)
*
* EIN-AUSGABEBEREICH AR
*
 01  MITABSATZ.                                        }
     05  PNUM        PIC 9(6) BINARY.                  }
     05  NAME        PIC X(20).                        }
     05  VORNAME     PIC X(10).                        }
     05  WOHNORT     PIC X(10).                        }
     05  ABTEILUNG   PIC X(5).                         }
     05  STRASSE     PIC X(22).                        }
*                                                      }  ----------------------  (9)
 01  PROTSATZ.                                         }
     05  PAKTION     PIC X.                            }
     05  PNUM        PIC 9(6).                         }
     05  NAME        PIC X(20).                        }
     05  DATUM       PIC X(10).                        }
     05  ZEIT        PIC 9(8).                         }
*                                                      }
```

```
* TERMINALEIN-AUSGABE
*
 01  TERMINALSATZ.
     05  PNUM        PIC 9(6).
     05  PNUMX       REDEFINES PNUM PIC X(6).
         88  END-KZ             VALUE "*END".
     05  FILLER      PIC X      VALUE SPACE.
     05  ABTEILUNG   PIC X(5).
     05  FILLER      PIC X      VALUE SPACE.
     05  NAME        PIC X(20).
     05  FILLER      PIC X      VALUE SPACE.
     05  VORNAME     PIC X(10).
     05  FILLER      PIC X      VALUE SPACE.
     05  WOHNORT     PIC X(10).
     05  FILLER      PIC X      VALUE SPACE.
     05  STRASSE     PIC X(22).
*
 01  ERRORLINE.
     05  FILLER      PIC X(13) VALUE "LEASY-FEHLER ".
     05  ERR-KODE    PIC X(8).
*
 01  OP-LINE.
     05  FILLER      PIC X(23) VALUE "GEWAEHLTE OPERATION: ".
     05  LAST-OP     PIC X(4).
     05  FILLER      PIC X(20) VALUE ", GEWAEHLTE DATEI: ".
     05  LAST-DB     PIC X(16).
*
 01  AUSGABE.
     05  AUSGABE1   PIC X(44) VALUE "BITTE GEBEN SIE DIE GEWUENSCH
-                                "TE AKTION EIN:".
     05  AUSGABE2   PIC X(30) VALUE "   I..MITARBEITER EINFUEGEN".
     05  AUSGABE3   PIC X(29) VALUE "   D..MITARBEITER LOESCHEN".
     05  AUSGABE4   PIC X(30) VALUE "   L..MITARBEITER AUFLISTEN".
     05  AUSGABE5   PIC X(24) VALUE "   E..PROGRAMM BEENDEN".
*
 01  EINGABE.
     05  MITARB-EING PIC X(51) VALUE "BITTE GEBEN SIE DIE DATEN IM
-                                 " ANGEZEIGTEN FORMAT EIN".
     05  END-EING    PIC X(28) VALUE "(*END: ENDE DER EINGABE)".
     05  T-AUS.
         10  FILLER  PIC X      VALUE SPACE.
         10  FILLER  PIC X(34) VALUE "PERSNR ABTLG NAME".
         10  FILLER  PIC X(47) VALUE "VORNAME    WOHNORT    STRASS
-                              "E".
```

(10)

(11)

```
     05  ALLGEMEIN   PIC X(27) VALUE "BITTE ERROR-CODE BEACHTEN".
     05  AKTION      PIC X     VALUE SPACE.
         88  EINFUEGEN         VALUE "I".
         88  LOESCHEN          VALUE "D".
         88  AUSGEBEN          VALUE "L".
         88  ENDE              VALUE "E".
     05  TEXT-NUMERIC PIC X(30) VALUE
         "PERSONALNUMMER NICHT NUMERISCH".
     05  KODE         PIC X.
*
*
*
 01  TABELLE.
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(6)  VALUE ALL "*".
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(5)  VALUE ALL "*".
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(20) VALUE ALL "*".
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(10) VALUE ALL "*".
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(10) VALUE ALL "*".
     05  FILLER      PIC X     VALUE SPACE.
     05  FILLER      PIC X(22) VALUE ALL "*".
*
 PROCEDURE DIVISION.
 MENUE SECTION.
 PROG-ANF.
*        VERBINDUNG ZU LEASY KATALOG
     DISPLAY "NAME LEASY-DATEIKATALOG ?" UPON DSS. ------------------ (12)
     ACCEPT CATNAME FROM DSS. --------------------------------------- (13)
     CALL "LEASY" USING OP-CATD RE CAT. ----------------------------- (14)
     IF NOT RC-OK PERFORM LEASY-ERROR )
             GO TO PROG-END.          } ------------------------------ (15)
*                                     )
     MOVE HIGH-VALUE TO OPE-OM. ------------------------------------- (16)
     CALL "LEASY" USING OP-OPFL RE DB4-OPFL. ------------------------ (17)
     IF NOT RC-OK PERFORM LEASY-ERROR
             GO TO PROG-END.
*
     PERFORM AKTION ------------------------------------------------- (18)
     UNTIL ENDE. ---------------------------------------------------- (19)
*
     CALL "LEASY" USING OP-CLFL RE. --------------------------------- (20)
     IF NOT RC-OK PERFORM LEASY-ERROR.
 PROG-END.
     STOP RUN.
```

```
/
 AKTION.
     DISPLAY AUSGABE1 UPON DSS.
     DISPLAY AUSGABE2 UPON DSS.
     DISPLAY AUSGABE3 UPON DSS.
     DISPLAY AUSGABE4 UPON DSS.
     DISPLAY AUSGABE5 UPON DSS.
     ACCEPT AKTION FROM DSS.
     MOVE AKTION TO PAKTION.                                    (21)
     IF EINFUEGEN
     THEN PERFORM EINFUEGEN
     ELSE IF LOESCHEN
          THEN PERFORM LOESCHEN
          ELSE IF AUSGEBEN
               THEN PERFORM AUSGEBEN.
*
 EINFUEGEN SECTION.
 OPTR-INSERT.
*
*         TRANSAKTIONSEROEFFNUNG FUER INSERT
*
     CALL "LEASY" USING OP-OPTR RE DB4-OPTR-UPDT.               (22)
     IF NOT RC-OK PERFORM LEASY-ERROR
         GO TO INSERT-END.
*
 TERMINAL-INPUT.
     DISPLAY MITARB-EING UPON DSS.
     DISPLAY END-EING UPON DSS.
     DISPLAY T-AUS UPON DSS.                                    (23)
     DISPLAY TABELLE UPON DSS.
 *
     ACCEPT TERMINALSATZ FROM DSS.
     IF END-KZ GO TO CLTR-INSRT.
     IF PNUM OF TERMINALSATZ NOT NUMERIC
         DISPLAY TEXT-NUMERIC UPON DSS                          (24)
         GO TO TERMINAL-INPUT.
     MOVE CORR TERMINALSATZ TO MITABSATZ.
*
*         EINFUEGEN DES MITARBEITERS
 INSRT-MITABSATZ.
*
```

```
     CALL "LEASY" USING OP-INSR RE DB-MITABDAT MITABSATZ.  ------- (25)
 *                                                  )
     IF RC-DUPEKY                                   |
         DISPLAY "SATZ BEREITS VORHANDEN"           |
         UPON DSS                                   |
         GO TO TERMINAL-INPUT                       } ------------------- (26)
     ELSE IF RC-LOCKED                              |
             DISPLAY "SATZ GESPERRT"                |
             UPON DSS                               |
             GO TO TERMINAL-INPUT.                  )
     IF RC-OK PERFORM PROTOKOLL-WRITE.  ------------------------- (27)
     IF RC-OK GO TO TERMINAL-INPUT.
     PERFORM LEASY-ERROR.
*
*       BEENDEN DER TRANSAKTION FUER INSRT
*
 CLTR-INSRT.
     CALL "LEASY" USING OP-CLTR RE.  ---------------------------- (28)
     IF NOT RC-OK PERFORM LEASY-ERROR.
*
 INSERT-END.
         EXIT.
*
 LOESCHEN SECTION.
 OPTR-DELETE.
*
*       TRANSAKTIONSEROEFFNUNG FUER DELETE
*
     CALL "LEASY" USING OP-OPTR RE DB4-OPTR-UPDT.  -------------- (29)
     IF NOT RC-OK PERFORM LEASY-ERROR
         GO TO DELETE-END.
 DLET-EING.
     DISPLAY "BITTE GEBEN SIE DIE PERSONALNR. (6-STELLIG) EIN"  )
         UPON DSS.                                              } (30)
     DISPLAY END-EING UPON DSS.                                 )
     ACCEPT PNUMX OF TERMINALSATZ FROM DSS.          )
     IF END-KZ GO TO CLTR-DELETE.                    |
     IF PNUM OF TERMINALSATZ NOT NUMERIC             |
         DISPLAY TEXT-NUMERIC UPON DSS               } ---------- (31)
         GO TO DLET-EING.                            |
     MOVE PNUM OF TERMINALSATZ TO PNUM OF MITABSATZ. |
*                                                    )
     CALL "LEASY" USING OP-RHLD RE DB-MITABDAT MITABSATZ.  ------- (32)
     IF RC-NOFIND
         DISPLAY "SATZ NICHT VORHANDEN" UPON DSS
         GO TO DLET-EING
     ELSE IF RC-LOCKED
         DISPLAY "SATZ GESPERRT" UPON DSS
```

```
            GO TO DLET-EING.
         MOVE CORR MITABSATZ TO TERMINALSATZ.
         DISPLAY TERMINALSATZ UPON DSS.                    ---------- (33)
*
 DISP.
         DISPLAY "LOESCHEN ? (J/N)" UPON DSS.
         ACCEPT KODE FROM DSS
         IF KODE = "Y" OR "J" OR "N" NEXT SENTENCE         ---------- (34)
         ELSE GO TO DISP.
         IF KODE = "N" GO TO DLET-EING.
         CALL "LEASY" USING OP-DLET RE DB-MITABDAT MITABSATZ. ------- (35)
         IF RC-OK PERFORM PROTOKOLL-WRITE
             GO TO DLET-EING.
         PERFORM LEASY-ERROR.
*
 CLTR-DELETE.
         CALL "LEASY" USING OP-CLTR RE. ------------------------------ (36)
         IF NOT RC-OK PERFORM LEASY-ERROR.
*
 DELETE-END.
         EXIT.
*
 AUSGEBEN SECTION.
*
*          TRANSAKTIONSEROEFFNUNG FUER LIST
*
 OPTR-LIST.
         MOVE SPACES TO TERMINALSATZ. -------------------------------- (37)
         CALL "LEASY" USING OP-OPTR RE DB4-OPTR-LIST. ---------------- (38)
         IF NOT RC-OK PERFORM LEASY-ERROR
             GO TO LISTOUT-END.
*
 LDISP.
         DISPLAY "SORTIERT NACH PERSNR (P), NAMEN (N) ODER ABTEILUNG (
-        "A) ?"
         UPON DSS. --------------------------------------------------- (39)
         ACCEPT KODE FROM DSS.
         IF KODE = "A" MOVE SI-ABT TO SI
         ELSE IF KODE = "N" MOVE SI-NAME TO SI             ---------- (40)
         ELSE MOVE SPACE TO SI.
*
*            POSITIONIEREN AUF DATEIANFANG
         MOVE LOW-VALUE TO MITABSATZ. -------------------------------- (41)
         CALL "LEASY" USING OP-SETL RE DB-MITABDAT
             MITABSATZ FA SI. -------------------------------------- (42)
             IF NOT RC-OK PERFORM LEASY-ERROR
             GO TO CLTR-LIST.
```

```
 RNXT.
     CALL "LEASY" USING OP-RNXT RE DB-MITABDAT MITABSATZ ──────── (43)
     IF RC-OK
        MOVE CORR MITABSATZ TO TERMINALSATZ ┐
        DISPLAY TERMINALSATZ UPON DSS       ├──────────────────── (44)
        GO TO RNXT.                         ┘
        IF NOT RC-EOF PERFORM LEASY-ERROR.
*
 CLTR-LIST.
     CALL "LEASY" USING OP-CLTR RE. ──────────────────────────── (45)
     IF NOT RC-OK PERFORM LEASY-ERROR.
*
 LISTOUT-END.
*
 PROTOKOLL-WRITE SECTION.                                      ┐
 PROT.                                                         │
     MOVE CORR MITABSATZ TO PROTSATZ.                          │
     ACCEPT DATUM FROM DATE4.                                  │
     ACCEPT ZEIT FROM TIME.                                    ├ ──────── (46)
     CALL "LEASY" USING OP-INSR RE DB-PROTDAT PROTSATZ.        │
 PROT-END.                                                     │
     EXIT.                                                     │
*                                                              ┘
 LEASY-ERROR SECTION.
 ERROR-DISPLAY.
*                                  ┐
     MOVE RC-CODE TO ERR-KODE.     │
     MOVE REOP TO LAST-OP.         │
     MOVE REDB TO LAST-DB.         ├ ──────────────────────────── (47)
     DISPLAY ERRORLINE UPON DSS.   │
     DISPLAY OP-LINE UPON DSS.     │
     DISPLAY ALLGEMEIN UPON DSS.   ┘
*
 ERROR-END.
     EXIT.
```

Erklärung

(1) Das COPY-Element LEASYKON, in dem die Definitionen der LEASY-Operationen, der OPEN-Modi und der USAGE-Modi enthalten sind, wird an diese Stelle kopiert.

(2) Das COPY-Element LEASYPAR, in dem der Verständigungsbereich RE und die restlichen Definitionen der Operanden der CALL-LEASY-Anweisung vereinbart sind, wird an diese Stelle kopiert.

(3) Die Dateien MITABDAT und PROTDAT werden im Dateiformat DB1 definiert, um einzeln auf sie zugreifen zu können.

(4) Zuweisen aller zu eröffnenden Dateien im Dateiformat DB4 mit ihrem OPEN-Modus:

MITABDAT im OPEN-Modus 4: INOUT,SHARUPD
PROTDAT im OPEN-Modus 9: EXTEND

(5) Zuweisen der Dateien, auf die in der UPDT-Transaktion zugegriffen wird:

MITABDAT im USAGE-Modus UPDT
PROTDAT im USAGE-Modus EXLD

(6) Zuweisen der Datei, auf die in der LIST-Transaktion zugegriffen wird.

(7) Definieren des Sekundärschlüssels ABT.

(8) Definieren des Sekundärschlüssels NAME.

(9) Definieren des Ein-/Ausgabebereichs AR mit dem Namen MITABSATZ für die Datei MITABDAT und mit dem Namen PROTSATZ für die Datei PROTDAT.

(10) Definieren des Bereichs für die Ein- und Ausgabe von der/an die Datensichtstation.

(11) Definitionen für die Ausgabe im Fehlerfall (siehe LEASY-ERROR SECTION).

(12) Der Name des zu bearbeitenden LEASY-Katalogs soll eingegeben werden.

(13) Die Eingabe wird in das Feld CATNAME geliefert, das bei der Kataloginformaton CAT definiert ist.

(14) CALL-LEASY-Aufruf für die Operation CATD. Der Verständigungsbereich RE und die Kataloginformation CAT werden mit übergeben.

(15) Steuerung des Fehlerverhaltens.

(16) Da in der anschließenden CALL-LEASY-Anweisung die Dateien im Format DB4 übergeben werden, muss das Feld OPE-OM auf den Wert HIGH-VALUE gesetzt werden.

(17) CALL-LEASY-Aufruf für die Operation OPFL. Die mit DB4-OPFL definierten Dateien werden physikalisch eröffnet.

(18) Der Anwender wird mit den an der Datensichtstation ausgegebenen Meldungen aufgefordert, die auszuführende Aktion auszuwählen. Anschließend wird im Programm entsprechend der Eingabe verzweigt.

(19) Dies geschieht solange, bis der Anwender das Endekriterium eingibt.

(20) Der CALL-LEASY-Aufruf mit der Operation CLFL schließt die Dateien nach der Angabe des Endekriteriums.

(21) siehe Punkt (18)

(22) Mit dieser CALL-LEASY-Anweisung wird die INSERT-Transaktion eröffnet. Die mit DB4-OPTR-UPDT definierten Dateien werden logisch eröffnet.

(23) Meldungen an die Datensichtstation. Sie teilen dem Anwender das Format der Eingabe mit.

(24) Die Eingabe wird in den Bereich TERMINALSATZ geliefert. Solange kein Endekriterium eingegeben wird und die Eingabe numerisch ist, wird der Inhalt von TERMINALSATZ in den Ein-/Ausgabebereich MITABSATZ der Datei MITABDAT gebracht.

(25) Mit diesem CALL-LEASY-Aufruf werden die Daten im AR-Bereich MITABSATZ in die mit DB-MITABDAT zugewiesene Datei MITABDAT geschrieben.

(26) Falls der eingegebene Satz bereits vorhanden oder gesperrt ist, werden entsprechende Meldungen ausgegeben.

(27) Wenn die INSERT-Transaktion ohne Fehler abgelaufen ist, wird zur PROTOKOLL-WRITE SECTION verzweigt.

(28) Wenn an der Datensichtstation das Endekriterium eingegeben wird oder beim Einfügen eines Satzes ein Fehler auftritt, wird die INSERT-Transaktion mit dem CALL-LEASY-Aufruf, Operation CLTR, beendet.

(29) Eröffnen der DELETE-Transaktion. Die mit DB4-OPTR-UPDT definierten Dateien werden logisch eröffnet.

(30) Die Personalnummer des zu löschenden Mitarbeiters ist einzugeben. In das Binärformat (4 Byte) konvertiert bildet sie den Primärschlüssel des zu löschenden Satzes.

(31) Die Eingabe wird in das Feld PNUMX des Bereichs TERMINALSATZ geliefert. Wenn kein Endekriterium eingegeben wird und der eingegebene Wert numerisch ist, wird der Inhalt des Feldes PNUM des Bereichs TERMINALSATZ in das Feld PNUM des Bereichs MITABSATZ übertragen.

(32) Mit diesem CALL-LEASY-Aufruf wird der Satz, dessen Primärschlüssel in der Satzzone MITABSATZ hinterlegt ist, aus der Datei MITABDAT gelesen und gesperrt.

(33) Der beim vorher durchgeführten Lesen in die Satzzone gelieferte Inhalt des Satzes wird in den Bereich TERMINALSATZ übertragen und an die Datensichtstation ausgegeben.

(34) Die Abfrage, ob der ausgegebene Satz gelöscht werden soll oder nicht, wird an die Datensichtstation ausgegeben. Je nach Antwort wird im Programm verzweigt.

(35) Der vorher gelesene und gesperrte Satz wird mit der CALL-LEASY-Anweisung, Operation DLET, aus der Datei MITABDAT gelöscht.

(36) Wenn an der Datensichtstation das Endekriterium eingegeben wird oder beim Löschen eines Satzes ein Fehler auftritt, wird die DELETE-Transaktion mit einem CALL-LEASY-Aufruf, Operation CLTR, beendet.

(37) Der Bereich TERMINALSATZ wird gelöscht.

(38) Die LIST-Transaktion wird eröffnet. Die mit DB4-OPTR-LIST definierte Datei wird logisch eröffnet.

(39) Mit dieser Meldung wird der Anwender dazu aufgefordert, auszuwählen, nach welchem Kriterium die auszugebenden Sätze sortiert werden sollen.

(40) Je nach Angabe des Anwenders wird entweder der Name des entsprechenden Sekundärschlüssels in das Feld des Sekundärindex SI geschrieben, auf den in der anschließenden Operation SETL positioniert werden soll, oder das Feld SI wird mit Leerzeichen überschrieben, wodurch auf den Primärschlüssel positioniert wird.

(41) Durch das Löschen des Ein-/Ausgabebereichs MITABSATZ wird mit der anschließenden Operation SETL auf Dateianfang positioniert.

(42) In dieser CALL-LEASY-Anweisung wird auf Dateianfang positioniert.

(43) Mit dieser CALL-LEASY-Anweisung wird der nächste Satz in Richtung aufsteigender Primär- oder Sekundärschlüssel-Reihenfolge (je nach Inhalt des Feldes SI) gelesen und in den Bereich MITABSATZ geschrieben. Gleichzeitig wird auf den neuen Satz positioniert.

(44) Der Inhalt des Bereichs MITABSATZ wird in den Bereich TERMINALSATZ geschrieben und an die Datensichtstation ausgegeben. Anschließend wird der neue Satz gelesen.

(45) Wenn beim Positionieren oder Lesen ein Fehler auftritt oder das Dateiende erreicht ist, wird die LIST-Transaktion mit dem CALL-LEASY-Aufruf, Operation CLTR, beendet.

(46) Wenn ein Satz eingefügt oder gelöscht wurde, wird zur PROTOKOLL-WRITE SECTION verzweigt. Hier wird der Inhalt der Felder PNUM und NAME von MITABSATZ in die entsprechenden Felder des Ein-Ausgabebereichs PROTSATZ der Datei PROTDAT übertragen. Das Tagesdatum und die Zeit werden ebenfalls in diesen Bereich eingetragen. Der Inhalt dieses Bereichs PROTSATZ wird mit der CALL-LEASY-Anweisung, Operation INSR, in die Datei PROTDAT eingetragen.

(47) Wenn bei einem CALL-LEASY-Aufruf ein Fehler auftritt, wird an diese Stelle des Programms verzweigt. Die von LEASY im Verständigungsbereich RE hinterlegten Fehlercodes bzw. der letzte Operationscode und Dateiname werden ausgegeben.

12.2 Assemblerprogramm

Dieses Beispiel zeigt, wie LEASY in einem Assemblerprogramm über die Makro-Schnittstelle aufgerufen wird.
Ablaufprotokolle, die den Einsatz der LEASY-Dienstprogramme in Verbindung mit dem Ablauf dieses Programms veranschaulichen, finden Sie in den Abschnitten „Ablaufprotokoll 1“ (Seite 364) und „Ablaufprotokoll 2" (Seite 382).

Ausdruck des Quellprogramms

```
  PERSDAT  START    0
           TITLE    'P E R S D A T'
           PRINT    NOGEN
  PERSDAT  AMODE    ANY
  PERSDAT  RMODE    ANY
           GPARMOD  31
2               *,VERSION 010
  *
  *
  ********************************************************************
  * IN DIESEM PROGRAMM WERDEN ZWEI DATEIEN UEBER LEASY BEARBEITET: *
  * LEASY-DATEIKATALOG: LCAT                                       *
  * DATEIEN:            MITABDAT:  ISAM                            *
  *                                KEYPOS=1                        *
  *                                KEYLEN=4                        *
  *                                RECFORM=F                       *
  *                                RECSIZE=74                      *
  *                                SI: ABT,POS=45,LEN=5            *
  *                                SI: NAME,POS=5,LEN=20           *
  *                       PROTDAT:   SAM                           *
  *                                RECSIZE=45                      *
  ********************************************************************
  *
  *
  ANF      BALR     R3,0
           USING    *,R3,R4
           USING    GTIMED,R5
           LA       R4,2048(0,R3)
           LA       R4,2048(0,R4)
           BIND     SYMBOL=I@GTIME,SYMBLAD=AENTRY
  *
  *
  MENUE    DS       0H
  *
  *                 VERBINDUNG ZU LEASY-KATALOG
  *
```

```
            MVC     AUSLEN,=Y(30)
            MVC     AUSTEXT(25),='NAME LEASY-DATEIKATALOG ?' ————————— (1)
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     EINTEXT,EINTEXT-1
            RDATA   EINB,TIAMERR
2                 *,@DCEI     999   921011   53531002
1                 *,RDATA     009   920320   53121057
            MVC     L@@CAT,EINTEXT ———————————————————————————————— (2)
            LA      R10,PROGEND
            LEA@CATD L@@RE,L@@CAT,ERRADDR=LEASYERR ———————————————— (3)
  *
            LA      R10,PROGEND
            LEA@OPFL L@@RE,DB4OPFL,POPEOM=X'FF',ERRADDR=LEASYERR ———— (4)
  *
  *                 AUSWAHL DER FUNKTIONEN
  AKTION    DS      0H ———————————————————————————————————————————— (5)
            MVC     AUSLEN,=Y(49)
            MVC     AUSTEXT(44),AUSGABE1
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     AUSLEN,=Y(35)
            MVC     AUSTEXT(30),AUSGABE2
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     AUSLEN,=Y(34)
            MVC     AUSTEXT(29),AUSGABE3
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     AUSLEN,=Y(35)
            MVC     AUSTEXT(30),AUSGABE4
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     AUSLEN,=Y(29)
            MVC     AUSTEXT(24),AUSGABE5
            WROUT   AUSB,TIAMERR
2                 *,@DCEO     999   921011   53531004
1                 *,WROUT     006   920316   53121058
            MVC     EINTEXT,EINTEXT-1
            RDATA   EINB,TIAMERR
2                 *,@DCEI     999   921011   53531002
1                 *,RDATA     009   920320   53121057
            MVC     PAKTION,EINTEXT
```

```
            CLI     PAKTION,'I'
            BE      EINFUEGN
            CLI     PAKTION,'D'
            BE      LOESCHEN
            CLI     PAKTION,'L'
            BE      AUSGEBEN
            CLI     PAKTION,'E' ------------------------------------------  (6)
            BNE     AKTION
  *
  *                 ABSCHLUSS-ROUTINEN
  *
  ENDE      LA      R10,PROGEND
            LEA@CLFL L@@RE,ERRADDR=LEASYERR ----------------------------  (7)
  *
  PROGEND   TERM
2                *,VERSION 100
  *
  *
  EINFUEGN DS       OH
  *
  *                 TRANSAKTIONSEROEFFNUNG FUER INSERT
  *
  OPTRINSR LA       R10,INSREND
            LEA@OPTR L@@RE,DB4OPTRU,ERRADDR=LEASYERR -------------------  (8)
  *
  *                 TERMINAL-EINGABE                                    )
  TERMINP   MVC     AUSLEN,=Y(56)                                       |
            MVC     AUSTEXT(51),MITARBEG                                |
            WROUT   AUSB,TIAMERR                                        |
2                *,@DCEO      999    921011   53531004                  |
1                *,WROUT      006    920316   53121058                  |
            MVC     AUSLEN,=Y(33)                                       |
            MVC     AUSTEXT(28),ENDEING                                 |
            WROUT   AUSB,TIAMERR                                        |
2                *,@DCEO      999    921011   53531004                  |
1                *,WROUT      006    920316   53121058                  } ----  (9)
            MVC     AUSLEN,=Y(84)                                       |
            MVC     AUSTEXT(79),TAUS                                    |
            WROUT   AUSB,TIAMERR                                        |
2                *,@DCEO      999    921011   53531004                  |
1                *,WROUT      006    920316   53121058                  |
            MVC     AUSLEN,=Y(84)                                       |
            MVC     AUSTEXT(79),TSTARS                                  |
            WROUT   AUSB,TIAMERR                                        |
2                *,@DCEO      999    921011   53531004                  |
1                *,WROUT      006    920316   53121058                  )
  *
```

```
            MVC      EINTEXT,EINTEXT-1                                  )
            RDATA    EINB,TIAMERR                                       |
2                 *,@DCEI      999    921011   53531002                  |
1                 *,RDATA      009    920320   53121057                  |
            MVC      TSATZ,EINTEXT                                      |
            CLC      TSATZ(4),ENDKZ                                     |
            BE       CLTRINSR                                           |
            LA       R10,TERMINP                                        |
            BAL      R11,TESTNUM                                        } ——— (10)
            PACK     DOWO,TPNUM                                         |
            CVB      R12,DOWO                                           |
            ST       R12,MPNUM                                          |
            MVC      MNAME,TNAME                                        |
            MVC      MVORNAME,TVORNAME                                  |
            MVC      MWOHNORT,TWOHNORT                                  |
            MVC      MABTLNG,TABTLNG                                    |
            MVC      MSTRASSE,TSTRASSE                                  )
  *
  *                  EINFUEGEN DES MITARBEITERS
  INSMSATZ  LA       R10,CLTRINSR
            LEA@INSR L@@RE,DBMDAT,MSATZ,                                          *
                  ERRCODE=(L002,L006),ERRADDR=LEASYERR ——————————————————— (11)
  TST02A    CLC      L@@RCLC,='L002'                                    )
            BNE      TST06A                                             |
            MVC      AUSLEN,=Y(27)                                      |
            MVC      AUSTEXT(22),='SATZ BEREITS VORHANDEN'              |
            WROUT    AUSB,TIAMERR                                       |
2                 *,@DCEO      999    921011   53531004                  |
1                 *,WROUT      006    920316   53121058                  |
            B        TERMINP                                            } ——— (12)
  TST06A    CLC      L@@RCLC,='L006'                                    |
            BNE      RCD00A                                             |
            MVC      AUSLEN,=Y(18)                                      |
            MVC      AUSTEXT(13),='SATZ GESPERRT'                       |
            WROUT    AUSB,TIAMERR                                       |
2                 *,@DCEO      999    921011   53531004                  |
1                 *,WROUT      006    920316   53121058                  )
            B        TERMINP
  RCD00A    LA       R10,TERMINP
            B        PROTWRT ———————————————————————————————————————— (13)
  *
  *                  BEENDEN DER TRANSAKTION FUER INSERT
  *
  CLTRINSR  LA       R10,INSREND
            LEA@CLTR L@@RE,ERRADDR=LEASYERR ——————————————————————————— (14)
  *
  INSREND   B        AKTION
  *
```

```
*
LOESCHEN DS       0H
*
*                 TRANSAKTIONSEROEFFNUNG FUER DELETE
*
OPTRDLET LA       R10,DLETEND
         LEA@OPTR L@@RE,DB4OPTRU,ERRADDR=LEASYERR ──────────────────── (15)
*
*                 TERMINAL-EINGABE
DLETEING MVC      AUSLEN,=Y(55)
         MVC      AUSTEXT(50),='BITTE GEBEN SIE DIE PERSONALNUMMER (6-S*
               TELLIG) EIN'                                          ┐
         WROUT    AUSB,TIAMERR                                        │
2                *,@DCEO      999    921011   53531004               │
1                *,WROUT      006    920316   53121058               │
         MVC      AUSLEN,=Y(33)                                       } ─── (16)
         MVC      AUSTEXT(28),ENDEING                                 │
         WROUT    AUSB,TIAMERR                                        │
2                *,@DCEO      999    921011   53531004               │
1                *,WROUT      006    920316   53121058               ┘
         MVC      EINTEXT,EINTEXT-1
         RDATA    EINB,TIAMERR                                        ┐
2                *,@DCEI      999    921011   53531002               │
1                *,RDATA      009    920320   53121057               │
         MVC      TPNUM,EINTEXT                                       │
         CLC      TPNUM(4),ENDKZ                                      │
         BE       CLTRDLET                                            } ─── (17)
         LA       R10,DLETEING                                        │
         BAL      R11,TESTNUM                                         │
         PACK     DOWO,TPNUM                                          │
         CVB      R12,DOWO                                            │
         ST       R12,MPNUM                                           │
         LA       R10,CLTRDLET                                        ┘
         LEA@RHLD L@@RE,DBMDAT,MSATZ,                                  *
               ERRCODE=(L001,L006),ERRADDR=LEASYERR ──────────────── (18)
TST01B   CLC      L@@RCLC,='L001'
         BNE      TST06B
         MVC      AUSLEN,=Y(25)
         MVC      AUSTEXT(20),='SATZ NICHT VORHANDEN'
         WROUT    AUSB,TIAMERR
2                *,@DCEO      999    921011   53531004
1                *,WROUT      006    920316   53121058
         B        DLETEING
TST06B   CLC      L@@RCLC,='L006'
         BNE      RCD00B
         MVC      AUSLEN,=Y(18)
         MVC      AUSTEXT,='SATZ GESPERRT'
         WROUT    AUSB,TIAMERR
```

```
2                *,@DCEO      999    921011   53531004
1                *,WROUT      006    920316   53121058
           B        DLETEING
  RCDOOB   L        R12,MPNUM                                    ┐
           CVD      R12,DOWO                                     │
           UNPK     TPNUM,DOWO                                   │
           OI       TPNUM+5,X'F0'                                │
           MVC      TNAME,MNAME                                  │
           MVC      TVORNAME,MVORNAME                            │
           MVC      TWOHNORT,MWOHNORT                            ├ ──── (19)
           MVC      TABTLNG,MABTLNG                              │
           MVC      TSTRASSE,MSTRASSE                            │
           MVC      AUSLEN,=Y(83)                                │
           MVC      AUSTEXT(78),TSATZ                            │
           WROUT    AUSB,TIAMERR                                 │
2                *,@DCEO      999    921011   53531004           ┘
1                *,WROUT      006    920316   53121058
  DISP     MVC      AUSLEN,=Y(21)                                ┐
           MVC      AUSTEXT(16),='LOESCHEN ? (J/N)'              │
           WROUT    AUSB,TIAMERR                                 │
2                *,@DCEO      999    921011   53531004           │
1                *,WROUT      006    920316   53121058           │
           MVC      EINTEXT,EINTEXT-1                            │
           RDATA    EINB,TIAMERR                                 │
2                *,@DCEI      999    921011   53531002           ├ ──── (20)
1                *,RDATA      009    920320   53121057           │
           CLI      EINTEXT,'N'                                  │
           BE       DLETEING                                     │
           CLI      EINTEXT,'J'                                  │
           BE       DELMSATZ                                     │
           CLI      EINTEXT,'Y'                                  │
           BNE      DISP                                         ┘
  *
  *                 LOESCHEN DES MITARBEITERS
  *
  DELMSATZ LA       R10,CLTRDLET
           LEA@DLET L@@RE,DBMDAT,MSATZ,ERRADDR=LEASYERR ──────────── (21)
           LA       R10,DLETEING
           B        PROTWRT
  *
  *                 BEENDEN DER TRANSAKTION FUER DELETE
  *
  CLTRDLET LA       R10,DLETEND
           LEA@CLTR L@@RE,ERRADDR=LEASYERR ──────────────────────── (22)
  *
  DLETEND  B        AKTION
  *
  *
```

```
AUSGEBEN DS       0H
*
*                 TRANSAKTIONSEROEFFNUNG FUER LIST
*                                                                    ┐
OPTRLIST MVI      TSATZ,' '                                          ├ ——— (23)
         MVC      TSATZ+1(77),TSATZ                                  ┘
         LA       R10,LISTEND
         LEA@OPTR L@@RE,DB4OPTRL,ERRADDR=LEASYERR ——————————————————— (24)
*
*                 TERMINAL-EINGABE
LDISP    MVC      AUSLEN,=Y(61)
         MVC      AUSTEXT,='SORTIERT NACH PERSNR (P), NAMEN (N) ODER AB*
               TEILUNG (A) ?'                                       ┐
         WROUT    AUSB,TIAMERR                                       ├ ——— (25)
2                 *,@DCEO      999    921011   53531004             ┘
1                 *,WROUT      006    920316   53121058
         MVC      EINTEXT,EINTEXT-1                                  ┐
         RDATA    EINB,TIAMERR                                       │
2                 *,@DCEI      999    921011   53531002             │
1                 *,RDATA      009    920320   53121057             │
LDA      CLI      EINTEXT,'A'                                        │
         BNE      LDN                                                │
         MVC      L@@SI,SIABT                                        │
         B        POSANF                                             │
LDN      CLI      EINTEXT,'N'                                        ├ ——— (26)
         BNE      LDP                                                │
         MVC      L@@SI,SINAME                                       │
         B        POSANF                                             │
LDP      CLI      EINTEXT,'P'                                        │
         BNE      LDISP                                              │
         MVC      L@@SI,=CL8' '                                      │
         B        POSANF                                             │
*                                                                    ┘
*                 POSITIONIEREN AUF DATEIANFANG
POSANF   XC       MSATZ,MSATZ ——————————————————————————————————————— (27)
         LA       R10,CLTRLIST
         LEA@SETL L@@RE,DBMDAT,MSATZ,L@@FA,L@@SI,ERRADDR=LEASYERR ——— (28)
*
*                 NAECHSTEN SATZ LESEN UND AUSGEBEN
*
RNXT     LA       R10,CLTRLIST
         LEA@RNXT L@@RE,DBMDAT,MSATZ,                                *
               ERRCODE=(L003),ERRADDR=LEASYERR ———————————————————— (29)
         CLC      L@@RCLC,='L003'
         BE       CLTRLIST
```

```
            L        R12,MPNUM                                          )
            CVD      R12,DOWO                                           |
            UNPK     TPNUM,DOWO                                         |
            OI       TPNUM+5,X'F0'                                      |
            MVC      TNAME,MNAME                                        |
            MVC      TVORNAME,MVORNAME                                  |
            MVC      TWOHNORT,MWOHNORT                                  } ---- (30)
            MVC      TABTLNG,MABTLNG                                    |
            MVC      TSTRASSE,MSTRASSE                                  |
            MVC      AUSLEN,=Y(83)                                      |
            MVC      AUSTEXT(78),TSATZ                                  |
            WROUT    AUSB,TIAMERR                                       |
2                 *,@DCEO      999    921011   53531004                 )
1                 *,WROUT      006    920316   53121058
            B        RNXT
  *
  *                  BEENDEN DER TRANSAKTION FUER LIST
  *
  CLTRLIST  LA       R10,LISTEND
            LEA@CLTR L@@RE,ERRADDR=LEASYERR ------------------------------ (31)
  *
  LISTEND   B        AKTION
  *
  *
  PROTWRT   DS       0H
  *
  *                  PROTOKOLL-SATZ AUSGABE
  *
  PROT      L        R12,MPNUM                                          )
            CVD      R12,DOWO                                           |
            UNPK     PPNUM,DOWO                                         |
            OI       PPNUM+5,X'F0'                                      |
            MVC      PNAME,MNAME                                        |
            LA       R13,SAVE                                           } ---- (32)
            LA       R5,GTIMEL                                          |
            GTIME    MF=E,PARAM=GTIMEL,LINKADR=AENTRY                   |
2                 *,VERSION 101                                         |
            MVC      PDATUM,NTIGDTIC                                    |
            MVC      PZEIT,NTIGTDI                                      |
            LEA@INSR L@@RE,DBPDAT,PSATZ,ERRADDR=LEASYERR                )
  *
  PROTEND   BR       R10
  *
  *
  TESTNUM   DS       0H
  *
  *                  PRUEFUNG AUF NUMERISCHEN WERT
  *
```

```
            LA       R12,6
            LA       R13,TPNUM
  TSTNUM1   CLI      0(R13),'9'
            BH       NOTNUM
            CLI      0(R13),'0'
            BL       NOTNUM
            LA       R13,1(0,R13)
            BCT      R12,TSTNUM1
            BR       R11
  NOTNUM    MVC      AUSLEN,=Y(35)
            MVC      AUSTEXT,TEXTNUME
            WROUT    AUSB,TIAMERR
2                 *,@DCEO      999    921011   53531004
1                 *,WROUT      006    920316   53121058
            BR       R10
  *
  *
  LEASYERR DS       0H
  *
  *                 ERROR-DISPLAY
  *                                                          ┐
            MVC      ERRKODE,L@@RCCC                         │
            MVC      LASTOP,L@@REOP                          │
            MVC      LASTDB,L@@REDB                          │
            MVC      AUSLEN,=Y(26)                           │
            MVC      AUSTEXT(21),ERRLINE                     │
            WROUT    AUSB,TIAMERR                            │
2                 *,@DCEO      999    921011   53531004      │
1                 *,WROUT      006    920316   53121058      │
            MVC      AUSLEN,=Y(68)                           } ——— (33)
            MVC      AUSTEXT(63),OPLINE                      │
            WROUT    AUSB,TIAMERR                            │
2                 *,@DCEO      999    921011   53531004      │
1                 *,WROUT      006    920316   53121058      │
            MVC      AUSLEN,=Y(32)                           │
            MVC      AUSTEXT(27),ALGEMEIN                    │
            WROUT    AUSB,TIAMERR                            │
2                 *,@DCEO      999    921011   53531004      │
1                 *,WROUT      006    920316   53121058      ┘
  *
  ERROREND BR       R10
  *
  *
  TIAMERR   WROUT    TIAMERRL,TERMD
2                 *,@DCEO      999    921011   53531004
1                 *,WROUT      006    920316   53121058
  *
  *
```

```
  TERMD    TERM     UNIT=STEP,MODE=ABNORMAL,DUMP=Y
2                *,VERSION 100
           PRINT    GEN
  *
  *
  *                 LEASY-PARAMS
  *
           LEA@RE ------------------------------------------------------- (34)
1 ********       LEASY COMMUNICATION AREA, VERSION 62A
1          DS    OF
1 L@@RE        DS    OCL80           LEASY COMMUNICATION AREA
1 L@@REK       DS    OCL48           COMPATIBLE PART
1 L@@RCCC      DC    CL3'000'        COMPATIBLE RETURNCODE
1 L@@RCOK      EQU   C'000'          RETURN-CODE: NO ERROR
1 L@@RCKZ      DC    CL1'L'          LEASY SYSTEM CHARACTERISTIC
1 L@@RCLC      DC    CL4'L000'       LEASY RETURN CODE
1 L@@PASS      DC    CL8' '          PASSWORD (FOR FUTURE VERSIONS)
1 L@@OPE       DS    OCL8            SUPPLEMENT FOR LEASY OPERATIONS
1 L@@OPSTX     DC    CL1' '         RESERVED
1 L@@STXA1     EQU   C' '             LEASY-STXIT-ROUTINE
1 L@@STXA2     EQU   X'00'            LEASY-STXIT-ROUTINE
1 L@@STXN      EQU   C' '             *VALUES 'N' AND 'P' NO LONGER
1 L@@STXP      EQU   C' '             *SUPPORTED SINCE LEASY V6.1
1 L@@OPOM      DC    CL1' '          OPEN MODE
1 L@@BLAN      EQU   C' '             STD FOR FORMAT DB1 AND DB2
1 L@@INPUT     EQU   C'1'             DVS OPEN MODE INPUT
1 L@@INPUS     EQU   C'2'             DVS OPEN MODE INPUT, SHARUPD
1 L@@INOUT     EQU   C'3'             DVS OPEN MODE INOUT
1 L@@INOUS     EQU   C'4'             DVS OPEN MODE INOUT, SHARUPD
1 L@@REVER     EQU   C'5'             DVS OPEN MODE REVERSE
1 L@@UPDAT     EQU   C'7'             DVS OPEN MODE UPDATE
1 L@@OUTPU     EQU   C'8'             DVS OPEN MODE OUTPUT
1 L@@EXTEN     EQU   C'9'             DVS OPEN MODE EXTEND
1 L@@OUTIN     EQU   C'A'             DVS OPEN MODE OUTIN
1 L@@OUTIS     EQU   C'B'             DVS OPEN MODE OUTIN, SHARUPD
1 *                              USAGE-MODES
1 L@@EXLD      EQU   C' '                  USAGE-MODE EXLD (SAM)
1 L@@UPDT      EQU   C' '                  USAGE-MODE UPDT (ISAM/PAM)
1 L@@RETRA     EQU   C'A'                  USAGE-MODE RETR
1 L@@PRUPE     EQU   C'E'                  USAGE-MODE PRUP
1 L@@EXRTG     EQU   C'G'                  USAGE-MODE EXRT
1 L@@EXLDL     EQU   C'L'                  USAGE-MODE EXLD
1 L@@PRRTI     EQU   C'I'                  USAGE-MODE PRRT
1 L@@EXLDO     EQU   C'O'                  USAGE-MODE EXLD
1 L@@ULRTR     EQU   C'R'                  USAGE-MODE ULRT
1 L@@EXLDQ     EQU   C'Q'                  USAGE-MODE EXLD
1 L@@ULUPU     EQU   C'U'                  USAGE-MODE ULUP
1 L@@EXRTX     EQU   C'X'                  USAGE-MODE EXRT
```

```
1 L@@EXUPB      EQU   C'B'             USAGE-MODE EXUP
1 *
1 L@@HVAL       EQU   X'FF'            FORMAT DB4
1 L@@OPLG       DC    CL1' '          BIM LOGGING FIELD
1 L@@YBIM       EQU   C' '             WITH BIM LOGGING
1 L@@NBIM       EQU   C'N'             WITHOUT BIM LOGGING
1          DC    CL5' '          UNUSED
1 L@@INT        DS    0XL8             FIELD FOR INTERNAL KEYS
1 L@@SPTR       DC    F'0'             SAM : ID1RPTR (24-BIT) OR
1 *                                          ID1BLK# (31-BIT)
1          ORG   L@@SPTR
1 L@@PAMHP      DC    F'0'             PAM : PAM BLOCK NUMBER
1 L@@SASNR      DC    F'0'             SAM : ID1REC# (31-BIT)
1          ORG   L@@SASNR
1 L@@UNUSE      DC    F'0'             PAM : UNUSED, SAM UNUSED (24-BIT)
1 L@@NUM        DC    CL8' '           NUMBER OF PRIMARY RECORDS
1 L@@IDE        DC    CL8' '           DCAM IDENTIFIKATION
1 L@@RELE       DS    0CL32            LEASY EXTENSION OF RE
1 L@@REOP       DC    CL4' '           LAST OPERATION CODE
1 L@@REDB       DC    CL16' '          LAST FILE (+SI-NAME)
1 L@@LOPT       DC    CL1'1'           LEASY VERSION BYTE
1 *                                   HAS TO CONTAIN '1'
1 L@@OPE1       DC    CL1' '           FOR OPTR,CLTR,RHLD,RNHD,RPHD,
1 *                                  LOCK,CINF,UNLK
1 *                                  OPTR,CLTR:
1 L@@OCST       EQU   C' '                STANDARD OPEN/CLOSE OF TRANSACTION
1 *                                  OPTR:
1 L@@OPW        EQU   C'W'                OPEN TRANSAKTION USING
1 *                                   FILE POINTERS IN CI-AREA
1 *                                  CLTR:
1 L@@CLR        EQU   C'R'                ROLL BACK TRANSACTION
1 *                                  RHLD,RNHD,RPHD,LOCK:
1 L@@SLOC       EQU   C'S'                READ LOCK (SHARE LOCK)
1 L@@WLOC       EQU   C' '                WRITE LOCK
1 *                                  CINF:
1 L@@CINFW      EQU   C' '                CINF-AUFRUF FUER OPTR/W
1 L@@CIFI       EQU   C'F'                CINF FUER FILE-INFO
1 *
1 *                                  UNLK:
1 L@@ULST       EQU   C' '                STANDARD FOR UNLK
1 L@@ULUP       EQU   C'U'                UNLK OF UPDATED RECORDS
1 *
1 L@@OPE2       DC    CL1' '      CLOSE TRANSACTION WITH
1 *                                   OR WITHOUT NEW START
1 L@@CLST       EQU   C' '         NO NEW START
1 L@@CLT        EQU   C'T'         NEW START
1 *
1 *
```

```
1 L@@NUMY       EQU   C'N'          RDIR/RHLD WITH NUM
1 L@@NUMN       EQU   C' '          RDIR/RHLD WITHOUT NUM
1 *                               NAEHERE ANGABEN FUER CINF
1 L@@CICA       EQU   C' '          INFO UEBER GANZEN KATALOG
1 L@@CICT       EQU   C'C'          INFO UEBER GANZEN KATALOG
1 L@@CIOP       EQU   C'O'          INFO UEBER OFFENE DATEIEN
1 L@@CITA       EQU   C'T'          INFO UEBER DATEIEN DER TA
1 L@@CISP       EQU   C'S'          INFO UEBER SPEZIELLE DATEI
1 L@@CIW        EQU   C'W'          FORTSETZUNG DER AUSKUNFTSFKT.
1 *
1 L@@ROL        EQU   C'L'          READ OVER LOCKED RECORD
1 L@@RNOL       EQU   C' '          DO NOT READ OVER LOCKED REC.
1 *
1 L@@OPWTM      DS    OZL3          WAIT TIME FOR LOCKING
1           DC    XL3'0'        DEFAULT VALUE
1 *
1 L@@EXRC       DC    CL5' '        ZUSAETZLICHER RETURNCODE
1 L@@UPROT      DC    C' '           AIM PROTOKOLLIERUNGS-KENNZEICHEN
1 L@@NPROT      EQU   C' '           NO USER-INFO AVAILABLE
1 L@@YPROT      EQU   C'Y'           USER-INFORMATION AVAILABLE
  *
            LEA@CAT ------------------------------------------------------------ (35)
1           LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=CAT
1 L@@CAT    DS    0CL44
1 L@@CATC    DC    CL24' '
1 L@@CATZ    DC    CL20' '
  *
            LEA@FA   ALL ------------------------------------------------------- (36)
1           LEA@@BP PRE=L,NAM=,GRU=GEN,ID=FA,TYP=L
1 L@@FA     DC    CL8'ALL'
  *
            LEA@SI ------------------------------------------------------------- (37)
1           LEA@@BP TYP=L,PRE=L,NAM=,GRU=GEN,ID=SI
1 L@@SI     DC    CL8' '
  *
  DBMDAT    LEA@DB1  MITABDAT -------------------------------------------------- (38)
1           LEA@@BP TYP=L,PRE=L,NAM=DBMDAT,GRU=GEN,ID=DB1
1 DBMDAT    DC    CL12'MITABDAT'
  *
  DBPDAT    LEA@DB1  PROTDAT
1           LEA@@BP TYP=L,PRE=L,NAM=DBPDAT,GRU=GEN,ID=DB1
1 DBPDAT    DC    CL12'PROTDAT'
  *
  DB4OPFL   LEA@DB   ((MITABDAT,4),(PROTDAT,9)) ------------------------------ (39)
1           LEA@@BP TYP=L,PRE=L,NAM=DB4OPFL,GRU=GEN,ID=DB
1 DB4OPFL   DC    '((MITABDAT,4),(PROTDAT,9))'
  *
  DB4OPTRU LEA@DB   ((MITABDAT,UPDT),(PROTDAT,EXLD)) ---------------------- (40)
```

```
1          LEA@@BP TYP=L,PRE=L,NAM=DB4OPTRU,GRU=GEN,ID=DB
1 DB4OPTRU DC    '((MITABDAT,UPDT),(PROTDAT,EXLD))'
  *
  DB4OPTRL LEA@DB   (MITABDAT,RETR) ------------------------------------ (41)
1          LEA@@BP TYP=L,PRE=L,NAM=DB4OPTRL,GRU=GEN,ID=DB
1 DB4OPTRL DC    '(MITABDAT,RETR)'
  *
  *
  SIABT    DC       CL8'ABT' ------------------------------------------- (42)
  *
  SINAME   DC       CL8'NAME' ------------------------------------------ (43)
           PRINT    NOGEN
  *
  *
  *                 EIN-AUSGABEBEREICHE AR
  *
           DS       0F
  MSATZ    DS       0CL74                                                 )
  MPNUM    DS       F                                                     |
  MNAME    DS       CL20                                                  |
  MVORNAME DS       CL10                                                  |
  MWOHNORT DS       CL10                                                  |
  MABTLNG  DS       CL5                                                   |
  MSTRASSE DS       CL22                                                  |
           DS       CL3                                                   } ---- (44)
  *                                                                       |
  PSATZ    DS       0CL45                                                 |
  PAKTION  DS       CL1                                                   |
  PPNUM    DS       CL6                                                   |
  PNAME    DS       CL20                                                  |
  PDATUM   DS       CL10                                                  |
  PZEIT    DS       CL8                                                   )
  *
  *                 TERMINAL-EIN-AUSGABEBEREICHE
  *                                                                       )
  TSATZ    DS       0CL78                                                 |
  TPNUM    DS       CL6                                                   |
           DC       CL1' '                                                |
  TABTLNG  DS       CL5                                                   |
           DC       CL1' '                                                |
  TNAME    DS       CL20                                                  } ---- (45)
           DC       CL1' '                                                |
  TVORNAME DS       CL10                                                  |
           DC       CL1' '                                                |
  TWOHNORT DS       CL10                                                  |
           DC       CL1' '                                                |
  TSTRASSE DS       CL22                                                  )
  *
```

```
ERRLINE  DS       OCL21                                          )
         DC       CL13'LEASY-FEHLER '                            |
ERRKODE  DS       CL8                                            |
*                                                                |
OPLINE   DS       OCL63                                          } ---- (46)
         DC       CL23'GEWAEHLTE OPERATION: '                    |
LASTOP   DS       CL4                                            |
         DC       CL20', GEWAEHLTE DATEI: '                      |
LASTDB   DS       CL16                                           )
*
AUSGABE1 DC       CL44'BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:'
AUSGABE2 DC       CL30'   I..MITARBEITER EINFUEGEN'
AUSGABE3 DC       CL29'   D..MITARBEITER LOESCHEN'
AUSGABE4 DC       CL30'   L..MITARBEITER AUFLISTEN'
AUSGABE5 DC       CL24'   E..PROGRAMM BEENDEN'
*
MITARBEG DC       CL51'BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT *
               EIN'
ENDEING  DC       CL28'(*END: ENDE DER EINGABE)'
TAUS     DC       CL79' PERSNR ABTLG NAME                 VORNAME    WO*
               HNORT    STRASSE'
TSTARS   DC       CL79' ****** ***** ******************** ********** ***
               ******** **********************'
*
ALGEMEIN DC       CL27'BITTE ERROR-CODE BEACHTEN'
*
TEXTNUME DC       CL30'PERSONALNUMMER NICHT NUMERISCH'
*
         DS       0H
EINB     DS       0CL260
EINLEN   DS       CL2
         DC       CL2' '
EINTEXT  DS       CL256
*
AUSB     DS       0CL260
AUSLEN   DS       CL2
         DC       CL3' '
AUSTEXT  DS       CL255
*
TIAMERRL DS       0CL22
         DC       AL2(22)
         DC       CL3' '
         DC       CL17'TIAM-MACRO-ERROR'
*
*
*                 DATENFELD FUER GTIME
*
GTIMEL   GTIME    MF=L,DATE=YES,TOD=YES
```

```
GTIMED   GTIME    MF=D
PERSDAT  CSECT
*
*
*                 SONSTIGE FELDER
*
DOWO     DS       D
*
AENTRY   DS       F
*
SAVE     DS       18F
*
ENDKZ    DC       CL4'*END'
*
*
R0       EQU      0
R1       EQU      1
R2       EQU      2
R3       EQU      3
R4       EQU      4
R5       EQU      5
R6       EQU      6
R7       EQU      7
R8       EQU      8
R9       EQU      9
R10      EQU      10
R11      EQU      11
R12      EQU      12
R13      EQU      13
R14      EQU      14
R15      EQU      15
*
*
         END      ANF
               ='LOESCHEN ? (J/N)'
               ='SORTIERT NACH PERSNR (P), NAMEN (N) ODER ABTEILUNG (A)
               =CL8' '
               ='L002'
               ='L006'
               ='L001'
               ='SATZ NICHT VORHANDEN'
               ='L003'
               =Y(30)
               =Y(49)
               =Y(35)
               =Y(34)
               =Y(29)
               =Y(56)
```

```
=Y(33)
=Y(84)
=Y(27)
='SATZ BEREITS VORHANDEN'
=Y(18)
=Y(55)
='BITTE GEBEN SIE DIE PERSONALNUMMER (6-STELLIG) EIN'
=Y(25)
=Y(83)
=Y(21)
=Y(61)
=Y(26)
=Y(68)
=Y(32)
='NAME LEASY-DATEIKATALOG ?'
='SATZ GESPERRT'
=X'9906021124049467' CONSISTENCY CONSTANT FOR AID
```

Erklärung

(1) Der Name des zu bearbeitenden LEASY-Katalogs soll eingegeben werden.

(2) Die Eingabe wird in das Feld LEA@CAT geliefert, das bei der Kataloginformation CAT definiert ist (siehe (35)).

(3) Makroaufruf für die Operation CATD. Der Verständigungsbereich RE (siehe (34)) und die Kataloginformation CAT (siehe (35)) werden übergeben.

(4) Makroaufruf mit der Operation OPFL. Die mit DB4OPFL definierten Dateien (siehe (39)) werden physikalisch eröffnet. Für das Format DB4 muss das Feld OPE-OM im RE-Bereich auf den Wert X'FF' gesetzt werden.

(5) Der Anwender wird mit den an der Datensichtstation ausgegebenen Meldungen aufgefordert, die auszuführende Aktion auszuwählen. Anschließend wird im Programm entsprechend der Eingabe verzweigt.

(6) Dies geschieht solange, bis der Anwender das Endekriterium eingibt.

(7) Der Makroaufruf mit der Operation CLFL schließt die Dateien nach der Eingabe des Endekriteriums.

(8) Mit diesem Makroaufruf wird die INSERT-Transaktion eröffnet. Die mit DB4OPTRU definierten Dateien (siehe (40)) werden logisch eröffnet.

(9) Meldungen an die Datensichtstation. Sie teilen dem Anwender das Format der Eingabe mit.

(10) Die Eingabe wird in den Bereich TSATZ geliefert. Solange kein Endekriterium eingegeben wird und die Eingabe numerisch ist, wird der Inhalt von TSATZ in den AR-Bereich MSATZ der Datei MITABDAT gebracht.

(11) Mit diesem Makroaufruf werden die Daten im AR-Bereich MSATZ in die mit DBMDAT zugewiesene Datei MITABDAT (siehe (38)) geschrieben.

(12) Falls der eingegebene Satz bereits vorhanden oder gesperrt ist, werden entsprechende Meldungen ausgegeben.

(13) Wenn die INSERT-Transaktion ohne Fehler abgelaufen ist, wird zur Protokollsatz-Ausgabe PROTWRT verzweigt.

(14) Wenn an der Datensichtstation das Endekriterium eingegeben wird oder beim Einfügen eines Satzes ein Fehler auftritt, wird die INSERT-Transaktion mit dem Makroaufruf für die Operation CLTR beendet.

(15) Eröffnen der DELETE-Transaktion. Die mit DB4OPTRU definierten Dateien (siehe (40)) werden logisch eröffnet.

(16) Die Personalnummer des zu löschenden Mitarbeiters ist einzugeben. In das Binärformat (4 Byte) konvertiert bildet sie den Primärschlüssel des zu löschenden Satzes.

(17) Die Eingabe wird in das Feld TPNUM des Bereichs TSATZ geliefert. Wenn kein Endekriterium eingegeben wird und der eingegebene Wert numerisch ist, wird der Inhalt des Feldes TPNUM des Bereichs TSATZ in das Feld MPNUM des AR-Bereichs MSATZ übertragen.

(18) Mit diesem Makroaufruf wird der Satz, dessen Primärschlüssel in der Satzzone MSATZ hinterlegt ist, aus der Datei MITABDAT gelesen und gesperrt.

(19) Der beim vorher durchgeführten Lesen in die Satzzone gelieferte Inhalt des Satzes wird in den Bereich TSATZ übertragen und an die Datensichtstation ausgegeben.

(20) Die Abfrage, ob der ausgegebene Satz gelöscht werden soll oder nicht, wird an die Datensichtstation ausgegeben. Je nach Antwort wird im Programm verzweigt.

(21) Der Makroaufruf mit der Operation DLET löscht den vorher gelesenen und gesperrten Satz aus der Datei MITABDAT.

(22) Wenn an der Datensichtstation das Endekriterium eingegeben wird oder beim Löschen eines Satzes ein Fehler auftritt, wird die DELETE-Transaktion mit einem Makroaufruf, Operation CLTR, beendet.

(23) Der Bereich TSATZ wird gelöscht.

(24) Die LIST-Transaktion wird eröffnet. Die mit DB4OPTRL definierte Datei (siehe (41)) wird logisch eröffnet.

(25) Mit dieser Meldung wird der Anwender dazu aufgefordert, auszuwählen, nach welchem Kriterium die auszugebenden Sätze sortiert werden sollen.

(26) Je nach Angabe des Anwenders wird entweder der Name des entsprechenden Sekundärschlüssels in das Feld L@@SI des SI-Bereichs geschrieben, auf den in der anschließenden Operation SETL positioniert werden soll, oder das Feld SI wird mit Leerzeichen überschrieben, wodurch auf den Primärschlüssel positioniert wird.

(27) Durch das Löschen des AR-Bereichs MSATZ wird mit der anschließenden Operation SETL auf Dateianfang positioniert.

(28) Der Makroaufruf mit der Operation SETL positioniert auf Dateianfang.

(29) Mit diesem Makroaufruf wird der nächste Satz in Richtung aufsteigender Primär- oder Sekundärschlüsselreihenfolge (je nach Inhalt des Feldes SI) gelesen und in den AR-Bereich MSATZ geschrieben. Gleichzeitig wird auf den neuen Satz positioniert.

(30) Der Inhalt des AR-Bereichs MSATZ wird in den Bereich TSATZ geschrieben und an die Datensichtstation ausgegeben. Anschließend wird der neue Satz gelesen.

(31) Wenn beim Positionieren oder Lesen ein Fehler auftritt oder das Dateiende erreicht ist, so wird die LIST-Transaktion mit dem Makroaufruf, Operation CLTR, beendet.

(32) Wenn ein Satz eingefügt oder gelöscht wurde, wird zur Protokollsatz-Ausgabe PROTWRT verzweigt. Hier wird der Inhalt der Felder MPNUM und MNAME von MITABSATZ in die entsprechenden Felder des AR-Bereichs PSATZ der Datei PROTDAT übertragen. Das Tagesdatum und die Zeit werden ebenfalls in diesen Bereich eingetragen. Der Inhalt des AR-Bereichs PSATZ wird mit dem Makroaufruf, Operation INSR, in die Datei PROTDAT eingetragen.

(33) Wenn bei einem Makroaufruf ein Fehler auftritt, wird an diese Stelle des Programms verzweigt. Die von LEASY im Verständigungsbereich RE hinterlegten Fehlercodes bzw. der letzte Operationscode und Dateiname werden ausgegeben.

(34) Dieser Makroaufruf generiert den Verständigungsbereich RE.

(35) Dieser Makroaufruf generiert den Bereich für die Kataloginformation CAT.

(36) Dieser Makroaufruf generiert den Feldauswahloperanden FA.

(37) Dieser Makroaufruf generiert den Bereich für den Sekundärindex SI.

(38) Die Dateien MITABDAT und PROTDAT werden im Dateiformat DB1 definiert, um einzeln auf sie zugreifen zu können.

(39) Zuweisen aller zu eröffnenden Dateien im Dateiformat DB4 mit ihrem OPEN-Modus:

MITABDAT im OPEN-Modus 4: INOUT,SHARUPD
PROTDAT im OPEN-Modus 9: EXTEND

(40) Zuweisen der Dateien, auf die in der UPDT-Transaktion zugegriffen wird:

MITABDAT im USAGE-Modus UPDT
PROTDAT im USAGE-Modus EXLD

(41) Zuweisen der Datei MITABDAT, auf die in der LIST-Transaktion zugegriffen wird, im USAGE-Modus RETR.

(42) Definieren des Sekundärschlüssels ABT.

(43) Definieren des Sekundärschlüssels NAME.

(44) Definieren des Ein-Ausgabebereichs AR mit dem Namen MSATZ für die Datei MITABDAT, mit dem Namen PSATZ für die Datei PROTDAT.

(45) Definieren des Bereichs TSATZ für die Ein- und Ausgabe von der/an die Datensichtstation.

(46) Definitionen für die Ausgabe im Fehlerfall.

12.3 Ablaufprotokolle

12.3.1 Ablaufprotokoll 1

Dieses Ablaufprotokoll zeigt den Ablauf des Anwenderprogramms PERSDAT (siehe COBOL-Programm ab Seite 329 und Assemblerprogramm ab Seite 345) in Verbindung mit den Dienstprogrammen LEASY-CATALOG, LEASY-MAINTASK, LEASY-MASTER und LEASY-RECONST. Bei der Rekonstruktion wird das Nachziehen der AIM-Dateigenerationen vom Benutzer durchgeführt.

```
/start-leasy-catalog ------------------------------------------------------------   (1)
%  BLS0523 ELEMENT 'CATALOG', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-CATALOG', VERSION '06.2A00' OF '2006-03-08 01:27:31' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0101 LEASY CATALOG PROGRAM VERSION V6.2A STARTED
*CAT LCAT,TYP=N,PAS=C'LCAT' -----------------------------------------------------   (2)
*FIL MITABDAT,AIM=Y,KEY=(ABT,45,5),KEY=(NAME,5,20),SHA=Y,RECFORM=F,  -
RECSIZE=74,KEYPOS=1,KEYLEN=4,SPACE=(12,3) ---------------------------------------   (3)
*FIL PROTDAT,LEA=T,FCBTYPE=SAM,RECFORM=F,RECSIZE=45,SPACE=(12,12) ----------------   (4)
*INF ,A -------------------------------------------------------------------------   (5)
%   DNAME=MITABDAT
%   FNAME=LCAT.MITABDAT
0000000012 :SPVS:$USER.LCAT.MITABDAT
  ---------------------------------- HISTORY      ----------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   07:35:01  ACC-TIME   =   07:35:01  CHANG-TIME =   07:35:01
  ACC-COUNT  = 1           S-ALLO-NUM = 0
  ---------------------------------- SECURITY     ----------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = ALL-USERS   ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  ---------------------------------- BACKUP       ----------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
  ---------------------------------- ORGANIZATION ----------------------------------
  FILE-STRUC = ISAM        BUF-LEN    = STD(1)      BLK-CONTR  = DATA (2K)
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (F,N)       REC-SIZE   = 74
  KEY-LEN    = 4           KEY-POS    = 1
  AVAIL      = *STD
```

```
  ------------------------------- ALLOCATION   -------------------------------
  SUPPORT    = PUB         S-ALLOC    = 3           HIGH-US-PA = 2
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.0       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        10 REL=         9 PAGES
0000000012 :SPVS:$USER.LCAT.MITABDAT-SI
  ------------------------------- HISTORY      -------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   07:35:01  ACC-TIME   =   07:35:01  CHANG-TIME =   07:35:01
  ACC-COUNT  = 1           S-ALLO-NUM = 0
  ------------------------------- SECURITY     -------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = ALL-USERS   ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  ------------------------------- BACKUP       -------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
  ------------------------------- ORGANIZATION -------------------------------
  FILE-STRUC = ISAM        BUF-LEN    = STD(2)      BLK-CONTR  = DATA (2K)
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (V,N)       REC-SIZE   = 0
  KEY-LEN    = 25          KEY-POS    = 5
  AVAIL      = *STD
  ------------------------------- ALLOCATION   -------------------------------
  SUPPORT    = PUB         S-ALLOC    = 4           HIGH-US-PA = 2
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.0       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        10 REL=         6 PAGES
%   LEASYTYPE=...S.......LOCK=.......NO
%   FCBTYPE=..ISAM.......PAD=........15
%   RECSIMAX=00074.......KEYLEN=....004
%   AIM=.......YES.......BIM=.......YES
%   WRPASS=.....NO.......RDPASS=.....NO
%   ROM=........NO
%   KEY=.(ABT     ,      (00045,005),YES,YES)        (001)
%   .....................HAS NO POINTERS IN SI FILE
%   KEY=.(NAME    ,      (00005,020),YES,YES)        (002)
%   .....................HAS NO POINTERS IN SI FILE
%   DNAME=PROTDAT
%   FNAME=LCAT.PROTDAT
0000000012 :SPVS:$USER.LCAT.PROTDAT
  ------------------------------- HISTORY      -------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   07:35:01  ACC-TIME   =   07:35:01  CHANG-TIME =   07:35:01
  ACC-COUNT  = 1           S-ALLO-NUM = 0
```

```
  ------------------------------- SECURITY     -------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = OWNER-ONLY  ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  ------------------------------- BACKUP       -------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
  ------------------------------- ORGANIZATION -------------------------------
  FILE-STRUC = SAM         BUF-LEN    = STD(1)      BLK-CONTR  = DATA
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (F,N)       REC-SIZE   = 45
  AVAIL      = *STD
  ------------------------------- ALLOCATION   -------------------------------
  SUPPORT    = PUB         S-ALLOC    = 12          HIGH-US-PA = 0
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.0       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        12 REL=         9 PAGES
%   LEASYTYPE=...T.......LOCK=.......NO
%   FCBTYPE=...SAM.......RECSIMAX=00045
%   AIM=.........NO.......BIM=.......YES
%   WRPASS=.....NO.......RDPASS=.....NO
%   ROM=.........NO
*END ──────────────────────────────────────────────────────────────────────────  (6)
%  LEA0110 NORMAL TERMINATION OF LEASY CATALOG PROGRAM
/show-file e.mtsk.1 ──────────────────────────────────────────────────────────  (7)

/SET-LOGON-PAR JOB-NAME=MAINTASK
/ASS-SYSLST  O.MTSK.1
/START-LEASY-MAINTASK─────────────────────────────────────────────────────────  (8)
CAT=LCAT ─────────────────────────────────────────────────────────────────────  (9)
FILES=2 ──────────────────────────────────────────────────────────────────────  (10)
LOG=Y ────────────────────────────────────────────────────────────────────────  (11)
TRANS=3 ──────────────────────────────────────────────────────────────────────  (12)
TIM=40 ───────────────────────────────────────────────────────────────────────  (13)
END
/EXIT-JOB SYSTEM-OUTPUT=*NONE

end                                                  S*SOF+     1(     1)

%  SHO0500 ':SPVS:$USER.E.MTSK.1' CLOSED
/enter-job e.mtsk.1,resources=*par(cpu-lim=10) ────────────────────────────────  (14)
%  JMS0066 JOB 'MAINTASK' ACCEPTED ON 06-04-21 AT 07:42, TSN = 91DS
/show-user-stat
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
 MAINTASK 91DS 2 BATCH    9 210      0.4962      10 FSC ──────────────────────  (15)
 DIATASK  91C7 3 DIALOG   0 210      0.4857    9000 FSC
%  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
```

```
/show-file-attr lcat.  ------------------------------------------------------------ (16)
      18 :SPVS:$USER.LCAT.BIM#.001
      18 :SPVS:$USER.LCAT.BIM#.002
      18 :SPVS:$USER.LCAT.BIM#.003
       3 :SPVS:$USER.LCAT.LEADIAG
       0 :SPVS:$USER.LCAT.LEASYAIM (FGG)
      18 :SPVS:$USER.LCAT.LEASYCAT
      12 :SPVS:$USER.LCAT.MITABDAT
      12 :SPVS:$USER.LCAT.MITABDAT-SI
      12 :SPVS:$USER.LCAT.PROTDAT
:SPVS: PUBLIC:      9 FILES RES=       111 FRE=        92 REL=        69 PAGES
/start-exe persdat  ------------------------------------------------------------ (17)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*I
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ***********************
*101721 AB212 SCHMITTINGER         FRANZ      MOOSBURG   ANDERLSTRASSE 3
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ***********************
*062224 AB212 SCHNIEIDINGER        GERHARD    BRUECKENAU SORTSTRASSE 7
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ***********************
*122510 AB21  EISBOSS              ROLF       MUENCHEN   WALTERWEG 25
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ***********************
*071531 AP360 EXBOSS               MONIKA     MUTTERHAUS BACHUFERSTR. 17
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ***********************
**END
```

```
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*N
122510 AB21  EISBOSS              ROLF       MUENCHEN   WALTERWEG 25
071531 AP360 EXBOSS               MONIKA     MUTTERHAUS BACHUFERSTR. 17
101721 AB212 SCHMITTINGER         FRANZ      MOOSBURG   ANDERLSTRASSE 3
062224 AB212 SCHNIEIDINGER        GERHARD    BRUECKENAU SORTSTRASSE 7
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*E ------------------------------------------------------------------------------ (18)
/show-user-stat ------------------------------------------------------------------ (19)
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
DIATASK  91C7 3 DIALOG   0 210      0.4857    9000 FSC
DIATASK  91C8 3 DIALOG   0 210      1.0013    9000 FSC
%  SPS0171 NO LOCAL SPOOLOUT JOB PRESENT
/copy-file lcat.mitabdat,save.mitabdat ------------------------------------------- (20)
/start-leasy-catalog ------------------------------------------------------------- (21)
%  BLS0523 ELEMENT 'CATALOG', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-CATALOG', VERSION '06.2A00' OF '2006-03-08 01:27:31' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0101 LEASY CATALOG PROGRAM VERSION V6.2A STARTED
*CAT LCAT,PAS=C'LCAT',INF=Y,CPC=SAVE --------------------------------------------- (22)
%  LEA5101 LAST SESSION: NUMBER = 1, DATE = 2006-04-21, TIME = 07:42:32-S. ------ (23)
%  LEA5002 SHADOW DIRECTORY NAME: $USER.SAVE SHADOW SUFFIX NAME:
%  LEA5105 DMS FILENAMES WITH CATID
%  LEA5108 ROM = NO IS SPECIFIED
*END
%  LEA0110 NORMAL TERMINATION OF LEASY CATALOG PROGRAM
/show-file e.mtsk.2 -------------------------------------------------------------- (24)

/SET-LOGON-PAR JOB-NAME=MAINTASK
/ASS-SYSLST  O.MTSK.2
/START-LEASY-MAINTASK
CAT=LCAT
FILES=2
LOG=Y
TRANS=3
TIM=40
ASP=(36,36)
END
/EXIT-JOB SYSTEM-OUTPUT=*NONE

END
```

```
/enter-job e.mtsk.2,resources=*par(cpu-lim=10)  ------------------------------------ (25)
%  JMS0066 JOB 'MAINTASK' ACCEPTED ON 06-04-21 AT 07:58, TSN = 91EL
/show-user-stat
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
MAINTASK 91EL 2 BATCH    9 210      0.4792      10 FSC      ------------------------ (26)
DIATASK  91C7 3 DIALOG   0 210      0.4857    9000 FSC
DIATASK  91C8 3 DIALOG   0 210      1.1819    9000 FSC
%  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
/show-file-attr lcat.leasyaim,inf=*all-attr
0000000000 :SPVS:$USER.LCAT.LEASYAIM (FGG)
  -------------------------------- HISTORY      --------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = NONE        CHANG-DATE = NONE
  CRE-TIME   =   07:42:32  ACC-TIME   = NONE        CHANG-TIME = NONE
  ACC-COUNT  = 0           S-ALLO-NUM = 0
  -------------------------------- SECURITY     --------------------------------
  READ-PASS  = YES         WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = ALL-USERS   ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  -------------------------------- BACKUP       --------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 0
  MIGRATE    = ALLOWED
  -------------------------------- GENERATION-INFO ------------------------------
  MAXIMUM    = 3           BASE-NUM   = 2           OVERFL-OPT = REUSE-VOL
  FIRST-GEN  = 1           LAST-GEN   = 2  ------------------------------------------ (27)
:SPVS: PUBLIC:      1 FILE  RES=         0 FRE=         0 REL=         0 PAGES
/start-exe persdat  ------------------------------------------------------------------ (28)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*P
062224 AB212 SCHNIEIDINGER        GERHARD    BRUECKENAU SORTSTRASSE 7
071531 AP360 EXBOSS               MONIKA     MUTTERHAUS BACHUFERSTR. 17
101721 AB212 SCHMITTINGER         FRANZ      MOOSBURG   ANDERLSTRASSE 3
122510 AB21  EISBOSS              ROLF       MUENCHEN   WALTERWEG 25
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*D
BITTE GEBEN SIE DIE PERSONALNR. (6-STELLIG) EIN
(*END: ENDE DER EINGABE)
*071531
```

```
071531 AP360 EXBOSS               MONIKA     MUTTERHAUS BACHUFERSTR. 17
LOESCHEN ? (J/N)
*J
BITTE GEBEN SIE DIE PERSONALNR. (6-STELLIG) EIN
(*END: ENDE DER EINGABE)
**END
 BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*I
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ************************
*094711 AB212 NEUBOSS              HARDY      MUENCHEN   AM KNACKEPUNKT 1
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ************************
*141719 AB212 HINHOLD              ANTJE      MUENCHEN   RAUCHERSTEG 3
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ************************
*291018 DP212 WALDI                BERND      MUENCHEN   SCHWABENSTR.30
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** ************************
**END
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*A
122510 AB21  EISBOSS              ROLF       MUENCHEN   WALTERWEG 25
062224 AB212 SCHNIEIDINGER        GERHARD    BRUECKENAU SORTSTRASSE 7
094711 AB212 NEUBOSS              HARDY      MUENCHEN   AM KNACKEPUNKT 1
101721 AB212 SCHMITTINGER         FRANZ      MOOSBURG   ANDERLSTRASSE 3
141719 AB212 HINHOLD              ANTJE      MUENCHEN   RAUCHERSTEG 3
291018 DP212 WALDI                BERND      MUENCHEN   SCHWABENSTR.30
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*E ──────────────────────────────────────────────────────────────────────────── (29)
```

```
*delete-file lcat.mitabdat  ------------------------------------------------------------ (30)
/start-exe c.persdat  ------------------------------------------------------------------ (31)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT  --------------------------------------------------------------------------------- (32)
LEASY-FEHLER 99ALDD33
GEWAEHLTE OPERATION:   OPFL, GEWAEHLTE DATEI:  MITABDAT    4
BITTE ERROR-CODE BEACHTEN
/help-msg dms0d33  --------------------------------------------------------------------- (33)
%  DMS0D33 SPECIFIED FILE NOT CATALOGED.
%  ? The requested file has not been cataloged in the system.
%    For the file no catalog entry could be found.
%  ! Correct the error and try again.
/start-leasy-reconst  ------------------------------------------------------------------ (34)
%  BLS0523 ELEMENT 'RECONST', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-RECONST', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0401 LEASY AFTER IMAGE PROGRAM (RECONST) VERSION V6.2A STARTED
*CAT LCAT,FRO=1,COP=Y  ----------------------------------------------------------------- (35)
*SES FRO=2  ---------------------------------------------------------------------------- (36)
*MOD SIU=N  ---------------------------------------------------------------------------- (37)
*END  ---------------------------------------------------------------------------------- (38)
SESS#=00002 TSN#=91EL D=2006-04-21 T=07:58:43-S  --------------------------------------- (39)
%  LEA0410 NORMAL TERMINATION OF LEASY AFTER IMAGE PROGRAM  (RECONST)
/copy-file save.mitabdat,lcat.mitabdat  ------------------------------------------------ (40)
/start-exe persdat  -------------------------------------------------------------------- (41)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
  I..MITARBEITER EINFUEGEN
  D..MITARBEITER LOESCHEN
  L..MITARBEITER AUFLISTEN
  E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*N  ------------------------------------------------------------------------------------ (42)
122510 AB21  EISBOSS              ROLF       MUENCHEN   WALTERWEG 25
141719 AB212 HINHOLD              ANTJE      MUENCHEN   RAUCHERSTEG 3
094711 AB212 NEUBOSS              HARDY      MUENCHEN   AM KNACKEPUNKT 1
101721 AB212 SCHMITTINGER         FRANZ      MOOSBURG   ANDERLSTRASSE 3
062224 AB212 SCHNIEIDINGER        GERHARD    BRUECKENAU SORTSTRASSE 7
291018 DP212 WALDI                BERND      MUENCHEN   SCHWABENSTR.30
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
```

Ausdruck der gewählten Funktion des Dienstprogramms LEASY-RECONST ———— (43)

Dieser Ausdruck wird bei jedem Rekonstruktionslauf automatisch erzeugt.

```
LEASY AFTER-IMAGE-RECONSTRUCTION       DATE OF RECONSTRUCTION: 2006-04-21  TIME: 08:08:40-S
SELECTED PARAMETERS:
--------------------

LEASY CATALOG  (*CAT):     SELECTED CATALOG NAME:..... :SPVS:$USER.LCAT
                           AFTER IMAGE FILE GENERATION NUMBER: 0001 UNTIL 0002
                           UPDATING SHADOW FILES:..... YES

OPERATION MODE (*MOD):     PRINT:..... NORMAL
                           UPDATING SHADOW FILES:..... YES
                           UPDATING SHADOW SI-FILES:..... NO
                           TRANSACTIONS SELECTED:..... ALL
                           UNLOAD CLASS-5-MEMORY:..... YES
                           SET FREE CMMAIN FOR RUNTIME-SYSTEM: USE OF CMMAIN NOT CHANGED

LIST REPORT    (*REP):     REQUESTED LENGTH OF RECORD OUTPUT:. SHORT
                           RECORD EXTRACTION:..... NOT SELECTED
                           LIST INVALID RECORDS:..... YES
                           RECORD SELECTION:..... ALL RECORDS
                           LIST USER-INFORMATION:..... NO
                           LIST PROTOCOL RECORDS:..... NO

DATE FILTER    (*DAT):     FROM DATE (YYYY-MM-DD):..... START OF FILE
                           TO   DATE (YYYY-MM-DD):..... END   OF FILE

SESSION FILTER (*SES):     FROM SESSION-NUMBER:.....     2
                           TO   SESSION-NUMBER:..... END   OF FILE
                           LAST TRANSACTION:..... END   OF TO-SESSION

FILE FILTER    (*FIL):     NOT SPECIFIED

RANGE FILTER   (*RAN):     NOT SPECIFIED
```

Ausdruck des Rekonstruktionsprotokolls (44)

```
LEASY AFTER IMAGE RECONSTRUCTION, AIMFILE=:SPVS:$USER.LCAT.LEASYAIM(*0002) NEW PAMBLOCK-LINK: BLOCK# = 0000001

OP   XYS POS SESSION TRANS ITR TSN  FILE                                    TIME        RECORD
-------------------------------------------------------------------------------------------------------------
SESS     4210 00002     0   0 91EL                                        07:58:43-S  D=2006-04-21
CATD     4270 00002     0   0 91C8                                        08:01:56-S  T=91C8 U=USER P=C.PERSDA I=TSN-91C8
OPEN     4372 00002     0   0 91C8                                        08:01:56-S  MITABDAT:SPVS:$USER.LCAT.MITABDAT
OPTR     4429 00002     2   1 91C8                                        08:01:58-S  B=         H=         A=$DIALOG  #=
DLET     4468 00002     2   1 91C8 :SPVS:$USER.SAVE.MITABDAT              08:01:58-S  KEY=X'0001176B'
CLTR     4500 00002     2   1 91C8                                        08:01:58-S
OPTR     4557 00002     3   1 91C8                                        08:01:58-S  B=         H=         A=$DIALOG  #=
STOR     4666 00002     3   1 91C8 :SPVS:$USER.SAVE.MITABDAT              08:02:01-S  KEY=X'000171F7'
STOR     4775 00002     3   1 91C8 :SPVS:$USER.SAVE.MITABDAT              08:02:01-S  KEY=X'00022997'
STOR     4884 00002     3   1 91C8 :SPVS:$USER.SAVE.MITABDAT              08:02:01-S  KEY=X'000470CA'
CLTR     4916 00002     3   1 91C8                                        08:02:01-S
CLOS     4951 00002     0   0 91C8                                        08:02:02-S
CATD     5011 00002     0   0 91C8                                        08:04:20-S  T=91C8 U=USER P=C.PERSDA I=TSN-91C8
ENDA     5083 00002     0   0 91EL                                        08:07:08-S  D=2006-04-21
```

Protokoll der ENTER-Prozedur E.MTSK.1 (45)

```
/START-LEASY-MAINTASK
%  BLS0523 ELEMENT 'MAINTASK', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MAINTASK', VERSION '06.2A00' OF '2006-03-08 01:28:12'
LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS
RESERVED
%  LEA0301 LEASY MAINTASK VERSION V6.2A STARTED
CAT=LCAT
FILES=2
LOG=Y
TRANS=3
TIM=40
END
%  LEA5303 WARM/COLD START SUCCESSFUL
%  LEA5307 NEW LEASY SESSION CREATED: SESSION NUMBER = 00003, DATE = 2006-04-
21, TIME = 08:21:04-S
%  LEA5304 *LEASY MAINTASK :SPVS:$USER.LCAT INITIALIZATION COMPLETED
%  LEA0310 NORMAL TERMINATION OF LEASY MAINTASK BECAUSE OF CLOSE DOWN
FUNCTION
```

Protokoll der ENTER-Prozedur E.MTSK.2 ——————————————————— (46)

```
/START-LEASY-MAINTASK
%  BLS0523 ELEMENT 'MAINTASK', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MAINTASK', VERSION '06.2A00' OF '2006-03-08 01:28:12'
LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS
RESERVED
%  LEA0301 LEASY MAINTASK VERSION V6.2A STARTED
CAT=LCAT
FILES=2
LOG=Y
TRANS=3
TIM=40
ASP=(36,36)
END
%  LEA5303 WARM/COLD START SUCCESSFUL
%  LEA5307 NEW LEASY SESSION CREATED: SESSION NUMBER = 00002, DATE = 2006-04-
21, TIME = 08:36:20-S
%  LEA5304 *LEASY MAINTASK :SPVS:$USER.LCAT INITIALIZATION COMPLETED
%  LEA5003 START OF AIM FILE GENERATION SWITCHING ON 2006-04-21 AT 08:38:06-S
%  LEA5004 AIM FILE GENERATION SWITCHING SUCCESSFUL
%  LEA0310 NORMAL TERMINATION OF LEASY MAINTASK BECAUSE OF CLOSE DOWN
FUNCTION
```

Protokoll des Dienstprogramms LEASY-MASTER

```
/start-leasy-master
%  BLS0523 ELEMENT 'MASTER', VERSION '06.2A' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MASTER', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0501 LEASY MASTER PROGRAM VERSION V6.2A STARTED
```

```
LEASY MASTER PROGRAM VERSION V6.2A..............SCREEN 001: MAINTASK SELECTION
................................................................................
PLEASE TYPE IN NAME OF LEASY DIRECTORY.....................................
(*END=END OF PROGRAM)......................................................
*LCAT                                                                          -(47)
................................................................................
PLEASE ENTER PASSWORD
*C'LCAT'                                                                       -(48)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:..........00001
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................NO CONTROL FUNCTION IS ACTIV
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0001                            -(49)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
BUCKET POOL MEMORY SIZE:.........00029696 BYTES
SIZE OF ONE BUCKET IN POOL:......00001024 BYTES
NUMBER OF BUCKETS IN BUCKET POOL:00000029
USED BUCKETS FOR LOCK ELEMENTS:..00000000
USED BUCKETS FOR TRANSACTIONS:...00000000 UNUSED
BUCKETS:.................00000029
NUMBER OF LOCKED DATA RECORDS:...00000000
NUMBER OF FREE LOCK ELEMENTS:....00000004
SYSLST PRINTOUT SWITCH IS SET:...ON
UPD. COMMANDS ON CMMAIN ALLOWED:.YES
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*CLOS                                                                          -(50)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:..........00001
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................TERMINATE MAINTASK
CMMAIN CONTROL:..................ACCEPT NO FURTHER TRANSACTIONS
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0001
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
FUNCTION SELECTION (OR +=CONTINUE; OR BLANK; OR *END)
**END                                                                          -(51)
```

```
%  LEA0510 NORMAL TERMINATION OF LEASY MASTER PROGRAM
/start-leasy-master
%  BLS0523 ELEMENT 'MASTER', VERSION '06.2A' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MASTER', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0501 LEASY MASTER PROGRAM VERSION V6.2A STARTED
```

```
LEASY MASTER PROGRAM VERSION V6.2A..............SCREEN 001: MAINTASK SELECTION
................................................................................
PLEASE TYPE IN NAME OF LEASY DIRECTORY.....................................
(*END=END OF PROGRAM)......................................................
*LCAT                                                                          -(52)
................................................................................
PLEASE ENTER PASSWORD
*C'LCAT'                                                                       -(53)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:..........00002
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................NO CONTROL FUNCTION IS ACTIV
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0002                    -(54)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
BUCKET POOL MEMORY SIZE:.........00028672 BYTES
SIZE OF ONE BUCKET IN POOL:......00001024 BYTES
NUMBER OF BUCKETS IN BUCKET POOL:00000028
USED BUCKETS FOR LOCK ELEMENTS:..00000000
USED BUCKETS FOR TRANSACTIONS:...00000000
UNUSED BUCKETS:..................00000028
NUMBER OF LOCKED DATA RECORDS:...00000000
NUMBER OF FREE LOCK ELEMENTS:....00000003
SYSLST PRINTOUT SWITCH IS SET:...ON
UPD. COMMANDS ON CMMAIN ALLOWED:.YES
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*AIMI                                                                  -(55)
```

```
................................................................................
................................................................................
................................................................................
%  LEA5003 START OF AIM FILE GENERATION SWITCHING ON 2006-04-21 AT 08:38:06-S
%  LEA5004 AIM FILE GENERATION SWITCHING SUCCESSFUL
................................................................................
................................................................................
................................................................................
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*CLOS                                                                  -(56)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:.........00002
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................TERMINATE MAINTASK
CMMAIN CONTROL:..................ACCEPT NO FURTHER TRANSACTIONS
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0003                          -(57)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
FUNCTION SELECTION (OR +=CONTINUE; OR BLANK; OR *END)                       -(58)
**END
```

```
%  LEA0510 NORMAL TERMINATION OF LEASY MASTER PROGRAM
```

Erklärung

(1) Aufruf des Dienstprogramms LEASY-CATALOG.

(2) Neuerstellen des LEASY-Katalogs LCAT mit dem Kennwort C'LCAT'.

(3) Erstellen der Stammdatei MITABDAT mit den angegebenen Eigenschaften.

(4) Erstellen der Temporärdatei PROTDAT mit den angegebenen Eigenschaften.

(5) Die Dateieigenschaften aller im Katalog enthaltenen Dateien sollen in vollem Umfang an die Datensichtstation ausgegeben werden.

(6) Beenden des Dienstprogramms LEASY-CATALOG.

(7) Ausgabe der ENTER-Datei E.MTSK.1, die die Maintask startet.

(8) In der ENTER-Datei wird LEASY-MAINTASK gestartet.

(9) Der LEASY-Katalog LCAT wird zugewiesen.

(10) Es dürfen gleichzeitig 2 Dateien eröffnet werden.

(11) Die Session wird mit AIM- und BIM-Sicherung gefahren.

(12) Es können gleichzeitig 3 Transaktionen ablaufen.

(13) Es soll maximal 40 Sekunden auf das Freiwerden der Session gewartet werden.

(14) Starten der Batchjobs E.MTSK.1. Die Maintask wird gestartet, die erste LEASY-Session initialisiert.

(15) Die Maintask ist gestartet.

(16) Auflisten der im LEASY-Katalog LCAT eingerichteten Dateien.

(17) Das Anwenderprogramm PERSDAT wird aufgerufen.

(18) Beenden des Anwenderprogramms.

(19) Die LEASY-Session wurde mit dem Dienstprogramm LEASY-MASTER mit der Funktion CLOS beendet (siehe Punkt (50)). Protokoll der Maintask siehe Punkt (45).

(20) Es wird eine Schattendatei der Personaldatei MITABDAT mit dem Namen SAVE.MITABDAT angelegt.

(21) Aufruf von LEASY-CATALOG.
Für eine spätere Rekonstruktion der Schattendateien muss vor dem Hochfahren der Maintask der Operand CPC gesetzt werden.

(22) Die Namenssystematik für die Rekonstruktion von Schattendateien wird festgelegt. SAVE ist der Katalogname im Namen der Schattendatei.

(23) Vom Dienstprogramm LEASY-CATALOG werden durch die Angabe von INF=YES in der *CAT-Anweisung die Sessionnummer, das Datum, die Uhrzeit der letzten LEASY-Session und der Wert von CPC ausgegeben.

(24) Ausgabe der ENTER-Datei E.MTSK.2, die die Maintask startet. Zusätzlich zu den Anweisungen der ENTER-Datei E.MTSK.1 (siehe Punkt (7)) wird mit der *ASP-Anweisung auf die nächste AIM-Dateigeneration umgeschaltet.

(25) Starten des Batchjobs E.MTSK.2. Die Maintask wird gestartet, die zweite LEASY-Session initialisiert.

(26) Die Maintask ist gestartet.

(27) Es wurde von der ersten auf die zweite AIM-Dateigeneration umgeschaltet.

(28) Erneutes Starten des Anwenderprogramms PERSDAT.

(29) Beenden des Anwenderprogramms.

(30) Die Primärdatei LCAT.MITABDAT wird versehentlich gelöscht.

(31) Das Anwenderprogramm PERSDAT wird aufgerufen.

(32) Bei dem Versuch, auf die Dateien mit dem Anwenderprogramm zuzugreifen, bricht das Programm ab.

(33) Das HELP-MSG-Kommando zeigt die Fehlerursache.

(34) Nach Umschalten auf die nächste AIM-Dateigeneration (siehe Punkte (54) und (55)) wird das Dienstprogramms LEASY-RECONST aufgerufen.
Bei einer Rekonstruktion der Schattendateien kann die Maintask im Zustand *USE=N oder *USE=R sein. Es muss keine neue Maintask gestartet werden.

(35) Der LEASY-Katalog LCAT wird zugewiesen. Von der ersten (FRO=1) bis zur höchsten existierenden AIM-Dateigeneration sollen alle für die Rekonstruktion herangezogen werden.
Die Schattendateien sollen rekonstruiert werden (COP=Y).

(36) Die Rekonstruktion soll ab der Session mit der Sessionnummer 2 beginnen.

(37) Die SI-Dateien sollen nicht rekonstruiert werden.

(38) Beenden der Anweisungseingabe.

(39) Informationen über die rekonstruierten Sessions werden ausgegeben. Die Rekonstruktionsprotokolle werden automatisch erstellt (siehe Punkte (43) und (44)).

(40) Von der rekonstruierten Schattendatei SAVE.MITABDAT wird die Originaldatei LCAT.MITABDAT kopiert.

(41) Das Anwenderprogramm PERSDAT wird aufgerufen.

(42) Die Dateien werden weiter verarbeitet.

(43) Ausdruck der gewählten Funktionen des Dienstprogramms LEASY-RECONST.

(44) Ausdruck des Rekonstruktionsprotokolls.

(45) Protokoll der Maintask aus der ersten Session (siehe Punkt (14)).

(46) Protokoll der Maintask aus der zweiten Session (siehe Punkt (25)).

(47) Nach Aufruf des Dienstprogramms LEASY-MASTER wird der LEASY-Katalog zugewiesen, für den die MASTER-Funktionen ausgeführt werden sollen.

(48) Das Kennwort C'LCAT' wird angefordert.

(49) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die erste AIM-Dateigeneration beschrieben wird.

(50) Die Maintask wird mit der Funktion CLOS beendet.

(51) Das Dienstprogramm LEASY-MASTER wird mit *END beendet.

(52) Nach erneutem Aufruf des Dienstprogramms LEASY-MASTER wird der LEASY-Katalog zugewiesen, für den die MASTER-Funktionen ausgeführt werden sollen.

(53) Das Kennwort C'LCAT' wird angefordert.

(54) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die zweite AIM-Dateigeneration beschrieben wird.

(55) Es wir unmittelbar auf die nächste AIM-Dateigeneration umgeschaltet.

(56) Die Maintask wird mit der Funktion CLOS beendet.

(57) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die dritte AIM-Dateigeneration beschrieben wird.

(58) Das Dienstprogramm LEASY-MASTER wird mit *END beendet.

12.3.2 Ablaufprotokoll 2

Dieses Ablaufprotokoll zeigt den Ablauf des Anwenderprogramms PERSDAT (siehe COBOL-Programm ab Seite 329 und Assemblerprogramm ab Seite 345) in Verbindung mit den Dienstprogrammen LEASY-CATALOG, LEASY-MAINTASK, LEASY-MASTER und LEASY-RECONST. Bei der Rekonstruktion werden die AIM-Dateigenerationen automatisch nachgezogen.

```
/start-leasy-catalog ------------------------------------------------------------   (1)
%  BLS0523 ELEMENT 'CATALOG', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-CATALOG', VERSION '06.2A00' OF '2006-03-08 01:27:31' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0101 LEASY CATALOG PROGRAM VERSION V6.2A STARTED
*CAT LCAT,TYP=N,PAS=C'LCAT',CPC=SAVE --------------------------------------------   (2)
*FIL MITABDAT,AIM=(Y,A),KEY=(ABT,45,5),KEY=(NAME,5,20),SHA=Y,RECFORM=F,-
RECSIZE=74,KEYPOS=1,KEYLEN=4,SPACE=(12,3) ---------------------------------------   (3)
*FIL PROTDAT,LEA=T,FCBTYPE=SAM,RECFORM=F,RECSIZE=45,SPACE=(12,12) ----------------   (4)
*INF ,A -------------------------------------------------------------------------   (5)
%   DNAME=MITABDAT
%   FNAME=LCAT.MITABDAT
0000000012 :SPVS:$USER.LCAT.MITABDAT
  -------------------------------- HISTORY      --------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   08:45:13  ACC-TIME   =   08:45:13  CHANG-TIME =   08:45:13
  ACC-COUNT  = 1           S-ALLO-NUM = 0
  -------------------------------- SECURITY     --------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = ALL-USERS   ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  -------------------------------- BACKUP       --------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
  -------------------------------- ORGANIZATION --------------------------------
  FILE-STRUC = ISAM        BUF-LEN    = STD(1)      BLK-CONTR  = DATA (2K)
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (F,N)       REC-SIZE   = 74
  KEY-LEN    = 4           KEY-POS    = 1
  AVAIL      = *STD
  -------------------------------- ALLOCATION   --------------------------------
  SUPPORT    = PUB         S-ALLOC    = 3           HIGH-US-PA = 2
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.0       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        10 REL=         9 PAGES
```

```
0000000012 :SPVS:$USER.LCAT.MITABDAT-SI
  ------------------------------- HISTORY      --------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   08:45:13  ACC-TIME   =   08:45:13  CHANG-TIME =   08:45:13
  ACC-COUNT  = 1           S-ALLO-NUM = 0
  ------------------------------- SECURITY     --------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = ALL-USERS   ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  ------------------------------- BACKUP       --------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
  ------------------------------- ORGANIZATION -------------------------------
  FILE-STRUC = ISAM        BUF-LEN    = STD(2)      BLK-CONTR  = DATA (2K)
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (V,N)       REC-SIZE   = 0
  KEY-LEN    = 25          KEY-POS    = 5
  AVAIL      = *STD
  ------------------------------- ALLOCATION   -------------------------------
  SUPPORT    = PUB         S-ALLOC    = 4           HIGH-US-PA = 2
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.0       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        10 REL=         6 PAGES
%   LEASYTYPE=...S.......LOCK=.......NO
%   FCBTYPE=..ISAM.......PAD=........15
%   RECSIMAX=00074.......KEYLEN=....004
%   AIM=YES, AUTOM.......BIM=.......YES
%   WRPASS=.....NO.......RDPASS=.....NO
%   ROM=........NO
%   KEY=.(ABT     ,      (00045,005),YES,YES)        (001)
%   .....................HAS NO POINTERS IN SI FILE
%   KEY=.(NAME    ,      (00005,020),YES,YES)        (002)
%   .....................HAS NO POINTERS IN SI FILE
%   DNAME=PROTDAT
%   FNAME=LCAT.PROTDAT
0000000012 :SPVS:$USER.LCAT.PROTDAT
  ------------------------------- HISTORY      --------------------------------
  CRE-DATE   = 2006-04-21  ACC-DATE   = 2006-04-21  CHANG-DATE = 2006-04-21
  CRE-TIME   =   08:45:13  ACC-TIME   =   08:45:13  CHANG-TIME =   08:45:13
  ACC-COUNT  = 1           S-ALLO-NUM = 0
  ------------------------------- SECURITY     --------------------------------
  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
  USER-ACC   = OWNER-ONLY  ACCESS     = WRITE       ACL        = NO
  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2006-04-21
  DESTROY    = NO          FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00
  SP-REL-LOCK= NO
  ------------------------------- BACKUP       --------------------------------
  BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
  MIGRATE    = ALLOWED
```

```
  ------------------------------- ORGANIZATION --------------------------------
  FILE-STRUC = SAM         BUF-LEN    = STD(1)      BLK-CONTR  = DATA
  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
  REC-FORM   = (F,N)       REC-SIZE   = 45
  AVAIL      = *STD
  ------------------------------- ALLOCATION  ---------------------------------
  SUPPORT    = PUB         S-ALLOC    = 12          HIGH-US-PA = 0
  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
      1       SPVS.1       D3435
  NUM-OF-EXT = 1
:SPVS: PUBLIC:      1 FILE  RES=        12 FRE=        12 REL=         9 PAGES
%   LEASYTYPE=...T.......LOCK=.......NO
%   FCBTYPE=...SAM.......RECSIMAX=00045
%   AIM=........NO.......BIM=.......YES
%   WRPASS=.....NO.......RDPASS=.....NO
%   ROM=........NO
*END ------------------------------------------------------------------------------------------  (6)
%  LEA0110 NORMAL TERMINATION OF LEASY CATALOG PROGRAM
/copy-file lcat.mitabdat,save.mitabdat ---------------------------------------------------------  (7)
/copy-file lcat.mitabdat-si,save.mitabdat-si ---------------------------------------------------  (8)
/show-file e.mtsk.3 ----------------------------------------------------------------------------  (9)

/SET-LOGON-PAR JOB-NAME=MAINTASK
/ASS-SYSLST O.MSTK.3
/START-LEASY-MAINTASK --------------------------------------------------------------------------  (10)
CAT=LCAT ---------------------------------------------------------------------------------------  (11)
FILES=2 ----------------------------------------------------------------------------------------  (12)
LOG=Y,M ----------------------------------------------------------------------------------------  (13)
AUT=Y ------------------------------------------------------------------------------------------  (14)
REN=ENTER-JOB E.RECONST.AUT,JOB-NAME=RECOAUT ---------------------------------------------------  (15)
TRANS=3 ----------------------------------------------------------------------------------------  (16)
TIM=40 -----------------------------------------------------------------------------------------  (17)
END
/EXIT-JOB SYSTEM-OUTPUT=*NONE
  end                                                          S*SOF+     1(     1)

/enter-job e.mtsk.3,resources=*par(cpu-lim=10) -------------------------------------------------  (18)
%  JMS0066 JOB 'MAINTASK' ACCEPTED ON 06-04-21 AT 08:50, TSN = 91KN
/show-user-stat
 NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
  DIATASK  91C7 3 DIALOG   0 210      3.6877    9000 FSC
  DIATASK  91C8 3 DIALOG   0 210      2.9012    9000 FSC
  DIATASK  91C9 3 DIALOG   0 210      3.3986    9000 FSC
  MAINTASK 91KN 2 BATCH    9 210      0.5239      10 FSC-------------------------  (19)
  RECOAUT  91KP 2 BATCH    9 210      0.4036     200 FSC -------------------------  (20)
  %  SPS0171 NO LOCAL SPOOLOUT JOB PRESENT
  %  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
  %  SCP1095 DPRINTSV WARNING : SOME DPRINT PRINT-JOBS CANNOT BE DISPLAYED
```

```
 /show-file-attr lcat. ──────────────────────────────────────────────────────── (21)
     18 :SPVS:$USER.LCAT.BIM#.001
     18 :SPVS:$USER.LCAT.BIM#.002
     18 :SPVS:$USER.LCAT.BIM#.003
      3 :SPVS:$USER.LCAT.LEADIAG
      0 :SPVS:$USER.LCAT.LEASYAIM (FGG)
     18 :SPVS:$USER.LCAT.LEASYCAT
     12 :SPVS:$USER.LCAT.MITABDAT
     12 :SPVS:$USER.LCAT.MITABDAT-SI
     12 :SPVS:$USER.LCAT.PROTDAT
:SPVS: PUBLIC:      9 FILES RES=       111 FRE=        92 REL=        69 PAGES
/start-exe persdat ──────────────────────────────────────────────────────────── (22)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*I
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
*094711 AB212 NEUBOSS               HARDY      MUENCHEN   AM KNACKEPUNKT 1
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
*151921 DP212 BUSCHBADER            REINHARD   MUENCHEN   AM WEICH 7
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
*291018 AB212 WALDI                 BERND      MUENCHEN   SCHWABENSTR. 30
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
**END
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*N
151921 DP212 BUSCHBADER            REINHARD   MUENCHEN   AM WEICH 7
094711 AB212 NEUBOSS               HARDY      MUENCHEN   AM KNACKEPUNKT 1
291018 AB212 WALDI                 BERND      MUENCHEN   SCHWABENSTR. 30
```

```
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*E ──────────────────────────────────────────────────────────────────────────────── (23)
/show-user-stat
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
DIATASK  91C7 3 DIALOG   0 210      4.2662    9000 FSC ──────────────────────────── (24)
DIATASK  91C8 3 DIALOG   0 210      2.9012    9000 FSC
DIATASK  91C9 3 DIALOG   0 210      3.5208    9000 FSC
NAME      TSN TYPE  PRI   SIZE  COPIES PRSIZE   RTSN OPT
RECOAUT  91K2 4 FT  240      2       0      0   91KP
%  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
%  SCP1095 DPRINTSV WARNING : SOME DPRINT PRINT-JOBS CANNOT BE DISPLAYED
/enter-job e.mtsk.3,resources=*par(cpu-lim=10) ──────────────────────────────────── (25)
%  JMS0066 JOB 'MAINTASK' ACCEPTED ON 06-04-21 AT 08:59, TSN = 91K5
/show-user-stat
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
MAINTASK 91K5 2 BATCH    9 210      0.4825      10 FSC ──────────────────────────── (26)
DIATASK  91C7 3 DIALOG   0 210      4.3008    9000 FSC
DIATASK  91C8 3 DIALOG   0 210      2.9012    9000 FSC
RECOAUT  91K6 2 BATCH    9 210      0.4015     200 FSC ──────────────────────────── (27)
DIATASK  91C9 3 DIALOG   0 210      3.5208    9000 FSC
%  SPS0171 NO LOCAL SPOOLOUT JOB PRESENT
%  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
%  SCP1095 DPRINTSV WARNING : SOME DPRINT PRINT-JOBS CANNOT BE DISPLAYED
/start-exe persdat ──────────────────────────────────────────────────────────────── (28)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*I
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
*281731 AB212 BLONDIE               OTTILIE    MUENSTER   SUDSCHWEDENW. 22
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
*302015 DP212 WUPPI                 HARTMUT    MUENCHEN   WALDTALSTR. 19
BITTE GEBEN SIE DIE DATEN IM ANGEZEIGTEN FORMAT EIN
(*END: ENDE DER EINGABE)
 PERSNR ABTLG NAME                 VORNAME    WOHNORT    STRASSE
 ****** ***** ******************** ********** ********** **********************
**END
```

```
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*A
094711 AB212 NEUBOSS              HARDY      MUENCHEN   AM KNACKEPUNKT 1
281731 AB212 BLONDIE              OTTILIE    MUENSTER   SUDSCHWEDENW. 22
291018 AB212 WALDI                BERND      MUENCHEN   SCHWABENSTR. 30
151921 DP212 BUSCHBADER           REINHARD   MUENCHEN   AM WEICH 7
302015 DP212 WUPPI                HARTMUT    MUENCHEN   WALDTALSTR. 19
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*E  ------------------------------------------------------------------------------ (29)
/delete-file lcat.mitabdat  ------------------------------------------------------ (30)
/start-exe c.persdat  ------------------------------------------------------------ (31)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT  --------------------------------------------------------------------------- (32)
LEASY-FEHLER 99ALDD33
GEWAEHLTE OPERATION:   OPFL, GEWAEHLTE DATEI:  MITABDAT    4
BITTE ERROR-CODE BEACHTEN
/help-msg dms0d33  --------------------------------------------------------------- (33)
%  DMSOD33 SPECIFIED FILE NOT CATALOGED.
%  ? The requested file has not been cataloged in the system.
%    For the file no catalog entry could be found.
%  ! Correct the error and try again.
/show-user-stat
NAME      TSN TYPE       PRI      CPU-USED CPU-MAX ACCOUNT#
MAINTASK 91K5 2 BATCH    9 210      0.5174      10 FSC
DIATASK  91C7 3 DIALOG   0 210      5.1639    9000 FSC
DIATASK  91C8 3 DIALOG   0 210      2.9012    9000 FSC
RECOAUT  91K6 2 BATCH    9 210      0.4565     200 FSC
DIATASK  91C9 3 DIALOG   0 210      3.6355    9000 FSC
%  SPS0171 NO LOCAL SPOOLOUT JOB PRESENT
%  SRO0376 NO RSO JOB OF TYPE 'T7' PRESENT
%  SCP1095 DPRINTSV WARNING : SOME DPRINT PRINT-JOBS CANNOT BE DISPLAYED
/show-fil-att lcat.mitabdat
          12 :SPVS:$USER.LCAT.MITABDAT  ---------------------------------------- (34)
  :SPVS: PUBLIC:      1 FILE  RES=        12 FRE=         9 REL=         9 PAGES
/start-exe c.persdat  ------------------------------------------------------------ (35)
%  BLS0500 PROGRAM 'C.PERSDAT', VERSION ' ' OF '2006-03-27' LOADED
NAME LEASY-DATEIKATALOG ?
*LCAT
```

```
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*L
SORTIERT N.PSNR(P),NAME(N) ODER ABT(A)?
*P
094711 AB212 NEUBOSS              HARDY      MUENCHEN   AM KNACKEPUNKT 1
151921 DP212 BUSCHBADER           REINHARD   MUENCHEN   AM WEICH 7
281731 AB212 BLONDIE              OTTILIE    MUENSTER   SUDSCHWEDENW. 22      —  (36)
291018 AB212 WALDI                BERND      MUENCHEN   SCHWABENSTR. 30
302015 DP212 WUPPI                HARTMUT    MUENCHEN   WALDTALSTR. 19
BITTE GEBEN SIE DIE GEWUENSCHTE AKTION EIN:
   I..MITARBEITER EINFUEGEN
   D..MITARBEITER LOESCHEN
   L..MITARBEITER AUFLISTEN
   E..PROGRAMM BEENDEN
*E
```

SYSLST-Protokolle des Dienstprogramms LEASY-RECONST (37)

```
LEASY AFTER-IMAGE-RECONSTRUCTION                        DATE OF RECONSTRUCTION: 2006-04-21   TIME: 09:14:18-S

SELECTED PARAMETERS:
--------------------
LEASY CATALOG  (*CAT):     SELECTED CATALOG NAME:..... :SPVS:$USER.LCAT
                           AFTER IMAGE FILE GENERATION NUMBER: 0002 UNTIL 0002
                           UPDATING SHADOW FILES:..... YES, AUTOMATIC

OPERATION MODE (*MOD):     PRINT:..................... NORMAL
                           UPDATING SHADOW FILES:..... YES
                           UPDATING SHADOW SI-FILES:.. YES
                           TRANSACTIONS SELECTED:..... ALL
                           UNLOAD CLASS-5-MEMORY:..... YES
                           SET FREE CMMAIN FOR RUNTIME-SYSTEM: USE OF CMMAIN NOT CHANGED

LIST REPORT    (*REP):     REQUESTED LENGTH OF RECORD OUTPUT:. SHORT
                           RECORD EXTRACTION:......... NOT SELECTED
                           LIST INVALID RECORDS:...... YES
                           RECORD SELECTION:.......... ALL RECORDS
                           LIST USER-INFORMATION:..... NO
                           LIST PROTOCOL RECORDS:..... NO

DATE FILTER    (*DAT):     FROM DATE (YYYY-MM-DD):..... START OF FILE
                           TO   DATE (YYYY-MM-DD):..... END   OF FILE
SESSION FILTER (*SES):     FROM SESSION-NUMBER:........ START OF FILE
                           TO   SESSION-NUMBER:........ END   OF FILE
                           LAST TRANSACTION:.......... END   OF TO-SESSION
FILE FILTER    (*FIL):     SELECTED FILES:............ :SPVS:$USER.LCAT.MITABDAT

RANGE FILTER   (*RAN):     NOT SPECIFIED



LEASY AFTER IMAGE RECONSTRUCTION, AIMFILE=:SPVS:$USER.LCAT.LEASYAIM(*0002) NEW PAMBLOCK-LINK: BLOCK# = 0000001

OP   XYS POS SESSION TRANS ITR TSN  FILE                                    TIME        RECORD
-------------------------------------------------------------------------------------------------------------
MTSK     4138 00001     0   0 91K5                                        08:59:13-S  D=2006-04-21
SESS     4210 00002     0   0 91K5                                        08:59:14-S  D=2006-04-21
CATD     4270 00002     0   0 91C7                                        09:02:22-S  T=91C7 U=USER P=C.PERSDA I=TSN-91C7
OPEN     4372 00002     0   0 91C7                                        09:02:22-S  MITABDAT:SPVS:$USER.LCAT.MITABDAT
OPTR     4429 00002     1   1 91C7                                        09:02:23-S  B=         H=         A=$DIALOG  #=
STOR     4538 00002     1   1 91C7 :SPVS:$USER.SAVE.MITABDAT              09:02:23-S  KEY=X'00044C83'
STOR     4647 00002     1   1 91C7 :SPVS:$USER.SAVE.MITABDAT              09:02:24-S  KEY=X'00049BBF'
CLTR     4679 00002     1   1 91C7                                        09:02:24-S
CLOS     4714 00002     0   0 91C7                                        09:02:25-S
CATD     4774 00002     0   0 91C7                                        09:11:03-S  T=91C7 U=USER P=C.PERSDA I=TSN-91C7
ENDA     4846 00002     0   0 91K5                                        09:14:18-S  D=2006-04-21
```

```
LEASY AFTER-IMAGE-RECONSTRUCTION                          DATE OF RECONSTRUCTION: 2006-04-21   TIME: 09:20:54-S

SELECTED PARAMETERS:
--------------------
LEASY CATALOG  (*CAT):     SELECTED CATALOG NAME:.............. :SPVS:$USER.LCAT
                           AFTER IMAGE FILE GENERATION NUMBER: 0002 UNTIL 0003
                           UPDATING SHADOW FILES:.............. YES, AUTOMATIC

OPERATION MODE (*MOD):     PRINT:.............................. NORMAL
                           UPDATING SHADOW FILES:.............. YES
                           UPDATING SHADOW SI-FILES:........... YES
                           TRANSACTIONS SELECTED:.............. ALL
                           UNLOAD CLASS-5-MEMORY:.............. YES
                           SET FREE CMMAIN FOR RUNTIME-SYSTEM: USE OF CMMAIN NOT CHANGED

LIST REPORT    (*REP):     REQUESTED LENGTH OF RECORD OUTPUT:. SHORT
                           RECORD EXTRACTION:.................. NOT SELECTED
                           LIST INVALID RECORDS:............... YES
                           RECORD SELECTION:................... ALL RECORDS
                           LIST USER-INFORMATION:.............. NO
                           LIST PROTOCOL RECORDS:.............. NO

DATE FILTER    (*DAT):     FROM DATE (YYYY-MM-DD):............. START OF FILE
                       TO  DATE (YYYY-MM-DD):.............. END  OF FILE

SESSION FILTER (*SES):    FROM SESSION-NUMBER:................ START OF FILE
                       TO  SESSION-NUMBER:................. END  OF FILE
                           LAST TRANSACTION:................... END   OF TO-SESSION

FILE FILTER    (*FIL):     SELECTED FILES:..................... :SPVS:$USER.LCAT.MITABDAT

RANGE FILTER   (*RAN):     NOT SPECIFIED


LEASY AFTER IMAGE RECONSTRUCTION, AIMFILE=:SPVS:$USER.LCAT.LEASYAIM(*0002) NEW PAMBLOCK-LINK: BLOCK# = 0000001

OP   XYS POS SESSION TRANS ITR TSN  FILE                                    TIME        RECORD
-------------------------------------------------------------------------------------------------------------------------
MTSK     4138 00001     0   0 91K5                                         08:59:13-S  D=2006-04-21
SESS     4210 00002     0   0 91K5                                         08:59:14-S  D=2006-04-21
CATD     4270 00002     0   0 91C7                                         09:02:22-S  T=91C7 U=USER P=C.PERSDA I=TSN-91C7
OPEN     4372 00002     0   0 91C7                                         09:02:22-S  MITABDAT:SPVS:$USER.LCAT.MITABDAT
OPTR     4429 00002     1   1 91C7                                         09:02:23-S  B=         H=         A=$DIALOG  #=
STOR     4538 00002     1   1 91C7 :SPVS:$USER.SAVE.MITABDAT               09:02:23-S  KEY=X'00044C83'
STOR     4647 00002     1   1 91C7 :SPVS:$USER.SAVE.MITABDAT               09:02:24-S  KEY=X'00049BBF'
CLTR     4679 00002     1   1 91C7                                         09:02:24-S
CLOS     4714 00002     0   0 91C7                                         09:02:25-S
CATD     4774 00002     0   0 91C7                                         09:11:03-S  T=91C7 U=USER P=C.PERSDA I=TSN-91C7
ENDA     4846 00002     0   0 91K5                                         09:14:18-S  D=2006-04-21


LEASY AFTER IMAGE RECONSTRUCTION, AIMFILE=:SPVS:$USER.LCAT.LEASYAIM(*0003) NEW PAMBLOCK-LINK: BLOCK# = 0000001
OP   XYS POS SESSION TRANS ITR TSN  FILE                                    TIME        RECORD
-------------------------------------------------------------------------------------------------------------------------
CSES     4138 00002     0   0 91K5                                         09:14:18-S  D=2006-04-21
CATD     4198 00002     0   0 91C7                                         09:19:15-S  T=91C7 U=USER P=C.PERSDA I=TSN-91C7
OPEN     4300 00002     0   0 91C7                                         09:19:15-S  MITABDAT:SPVS:$USER.LCAT.MITABDAT
```

Protokoll der ENTER-Prozedur E.MTSK.3 (38)

```
/START-LEASY-MAINTASK
%  BLS0523 ELEMENT 'MAINTASK', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MAINTASK', VERSION '06.2A00' OF '2006-03-08 01:28:12' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0301 LEASY MAINTASK VERSION V6.2A STARTED
CAT=LCAT
FILES=2
LOG=Y,M
AUT=Y
REN=ENTER-JOB E.RECONST.AUT,JOB-NAME=RECOAUT
TRANS=3
TIM=40
END
%  JMS0066 JOB 'RECOAUT' ACCEPTED ON 06-04-21 AT 08:59, TSN = 91K6
%  LEA5303 WARM/COLD START SUCCESSFUL
%  LEA5307 NEW LEASY SESSION CREATED: SESSION NUMBER = 00002, DATE = 2006-04-21, TIME =
08:59:14-S
%  LEA5304 *LEASY MAINTASK :SPVS:$USER.LCAT INITIALIZATION COMPLETED
%  LEA5003 START OF AIM FILE GENERATION SWITCHING ON 2006-04-21 AT 09:14:18-S
%  LEA5004 AIM FILE GENERATION SWITCHING SUCCESSFUL
%  LEA0310 NORMAL TERMINATION OF LEASY MAINTASK BECAUSE OF CLOSE DOWN FUNCTION
END
```

Protokoll der ENTER-Prozedur E.RECONST.AUT (39)

```
/START-LEASY-RECONST
%  BLS0523 ELEMENT 'RECONST', VERSION '06.2A00', TYPE 'L' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-RECONST', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0401 LEASY AFTER IMAGE PROGRAM (RECONST) VERSION V6.2A STARTED
CAT :SPVS:$USER.LCAT    ,COP=(Y,A)
END
SESS#=00002 TSN#=91K5 D=2006-04-21 T=08:59:14-S
%  LEA0410 NORMAL TERMINATION OF LEASY AFTER IMAGE PROGRAM  (RECONST)
```

Protokoll des Dienstprogramms LEASY-MASTER

```
/start-leasy-master
%  BLS0523 ELEMENT 'MASTER', VERSION '06.2A' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MASTER', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0501 LEASY MASTER PROGRAM VERSION V6.2A STARTED
```

```
.....................................................................
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 001: MAINTASK SELECTION
.....................................................................
PLEASE TYPE IN NAME OF LEASY DIRECTORY.....................................
(*END=END OF PROGRAM)......................................................
*LCAT                                                                    -(40)
.....................................................................
PLEASE ENTER PASSWORD
*C'LCAT'                                                                 -(41)
```

```
LEASY MASTER PROGRAM VERSION V6.2A............SCREEN 003: GENERAL INFORMATION
.....................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:..........00001
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................NO CONTROL FUNCTION IS ACTIV
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0001                      -(42)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
BUCKET POOL MEMORY SIZE:.........00029696 BYTES
SIZE OF ONE BUCKET IN POOL:......00001024 BYTES
NUMBER OF BUCKETS IN BUCKET POOL:00000029
USED BUCKETS FOR LOCK ELEMENTS:..00000000
USED BUCKETS FOR TRANSACTIONS:...00000000
UNUSED BUCKETS:..................00000029
NUMBER OF LOCKED DATA RECORDS:...00000000
NUMBER OF FREE LOCK ELEMENTS:....00000003
SYSLST PRINTOUT SWITCH IS SET:...ON
UPD. COMMANDS ON CMMAIN ALLOWED:.YES
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*CLOS                                                                    -(43)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:..........00001
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................TERMINATE MAINTASK
CMMAIN CONTROL:..................ACCEPT NO FURTHER TRANSACTIONS
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0001
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
FUNCTION SELECTION (OR +=CONTINUE; OR BLANK; OR *END)
**END                                                                          -(44)
```

```
%  LEA0510 NORMAL TERMINATION OF LEASY MASTER PROGRAM
/start-leasy-master
%  BLS0523 ELEMENT 'MASTER', VERSION '06.2A' FROM LIBRARY
':5OSH:$TSOS.SYSPRG.LEASY.062' IN PROCESS
%  BLS0524 LLM 'LEASY-MASTER', VERSION '06.2A00' OF '2006-03-08 01:28:19' LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006. ALL RIGHTS RESERVED
%  LEA0501 LEASY MASTER PROGRAM VERSION V6.2A STARTED
```

```
................................................................................
LEASY MASTER PROGRAM VERSION V6.2A..............SCREEN 001: MAINTASK SELECTION
................................................................................
PLEASE TYPE IN NAME OF LEASY DIRECTORY..........................................
(*END=END OF PROGRAM)...........................................................
*LCAT                                                                          -(45)
................................................................................
PLEASE ENTER PASSWORD
*C'LCAT'                                                                       -(46)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
.................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:.........00002
CMMAIN STATUS:..................NORMAL WORKING
CMMAIN CONTROL:.................NO CONTROL FUNCTION IS ACTIV
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0002                          -(47)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
BUCKET POOL MEMORY SIZE:.........00028672 BYTES
SIZE OF ONE BUCKET IN POOL:......00001024 BYTES
NUMBER OF BUCKETS IN BUCKET POOL:00000028
USED BUCKETS FOR LOCK ELEMENTS:..00000000
USED BUCKETS FOR TRANSACTIONS:...00000000
UNUSED BUCKETS:..................00000028
NUMBER OF LOCKED DATA RECORDS:...00000000
NUMBER OF FREE LOCK ELEMENTS:....00000002
SYSLST PRINTOUT SWITCH IS SET:...ON
UPD. COMMANDS ON CMMAIN ALLOWED:.YES
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*REPO                                                                       -(48)
```

```
LEASY MASTER PROGRAM VERSION V6.2A..................SCREEN 039: COPY SHADOWFILE
.................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.................................................................................
TIME TO WAIT FOR END OF TRANSACTIONS:...005
REACTION FOR UNFINISHED TRANSACTIONS:.....R
.................................................................................
.................................................................................
FUNCTION SELECTION (OR R=REACTION, IN CASE OF OPEN TRANSACTIONS AFTER
WAITING TIME; OR W=ENTER WAITING TIME; OR F=FILE SELECTION;
OR S=START FUNCTION PROCESSING; OR BLANK=MAINTASK SELECTION;
OR *END=END OF PROGRAM)
*W                                                                          -(49)
```

```
LEASY MASTER PROGRAM VERSION V6.2A................SCREEN 040: ENTER WAITING TIME
.................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.................................................................................
PLEASE ENTER THE TIME TO WAIT FOR THE COMPLETION OF NOT YET CLOSED TRANSACTIONS
(0<=WAITING TIME<=120; BLANK=5 MINUTES (STANDARD VALUE))
*1                                                                          -(50)
```

```
LEASY MASTER PROGRAM VERSION V6.2A...................SCREEN 039: COPY SHADOWFILE
.....................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.....................................................................................
TIME TO WAIT FOR END OF TRANSACTIONS:...001
REACTION FOR UNFINISHED TRANSACTIONS:.....R
.....................................................................................
.....................................................................................
FUNCTION SELECTION (OR R=REACTION, IN CASE OF OPEN TRANSACTIONS AFTER
WAITING TIME; OR W=ENTER WAITING TIME; OR F=FILE SELECTION;
OR S=START FUNCTION PROCESSING; OR BLANK=MAINTASK SELECTION;
OR *END=END OF PROGRAM)
*F                                                                          -(51)
```

```
LEASY MASTER PROGRAM VERSION V6.2A...................SCREEN 042: FILE SELECTION
.....................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.....................................................................................
NO FILES SELECTED FOR FUNCTION REPO
.....................................................................................
FILE SELECTION (A=ADD FILENAME; OR R=REMOVE FILENAME;
OR E=END OF FILE SELECTION)
*A                                                                          -(52)
```

```
LEASY MASTER PROGRAM VERSION V6.2A....................SCREEN 043: ADD FILENAME
.....................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.....................................................................................
PLEASE ENTER LOGICAL FILENAME TO BE ADDED
(BLANK=STOP ADDING LOGICAL FILENAMES):
*MITABDAT                                                                   -(53)
```

```
LEASY MASTER PROGRAM VERSION V6.2A...................SCREEN 042: FILE SELECTION
.....................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.....................................................................................
SELECTED FILES:
.....................................................................................
MITABDAT
.....................................................................................
FILE SELECTION (A=ADD FILENAME; OR R=REMOVE FILENAME;
OR E=END OF FILE SELECTION)
*E                                                                                  -(54)
```

```
LEASY MASTER PROGRAM VERSION V6.2A..................SCREEN 039: COPY SHADOWFILE
.....................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
.....................................................................................
TIME TO WAIT FOR END OF TRANSACTIONS:...001
REACTION FOR UNFINISHED TRANSACTIONS:.....R
.....................................................................................
.....................................................................................
FUNCTION SELECTION (OR R=REACTION, IN CASE OF OPEN TRANSACTIONS AFTER
WAITING TIME; OR W=ENTER WAITING TIME; OR F=FILE SELECTION;
OR S=START FUNCTION PROCESSING; OR BLANK=MAINTASK SELECTION;
OR *END=END OF PROGRAM)
*S                                                                                  -(55)
```

```
%  LEA5003 START OF AIM FILE GENERATION SWITCHING ON 2006-04-21 AT 09:14:18-S
%  LEA5004 AIM FILE GENERATION SWITCHING SUCCESSFUL
%  LEA5536 FILE MITABDAT COPIED
%  LEA5536 FILE MITABDAT-SI COPIED
%  LEA5509 FUNCTION REPO NORMALLY TERMINATED
.....................................................................................
FUNCTION SELECTION (OR BLANK=MAINTASK SELECTION; OR *END=END OF PROGRAM)
*CLOS                                                                               -(56)
```

```
LEASY MASTER PROGRAM VERSION V6.2A.............SCREEN 003: GENERAL INFORMATION
................................................................................
CURRENT LEASY DIRECTORY:.........:SPVS:$USER.LCAT
CURRENT SESSION NUMBER:.........00002
CMMAIN STATUS:...................NORMAL WORKING
CMMAIN CONTROL:..................TERMINATE MAINTASK
CMMAIN CONTROL:..................ACCEPT NO FURTHER TRANSACTIONS
USE BEFORE IMAGE LOGGING:........YES
USE AFTER IMAGE LOGGING:.........YES, AIM GEN#=0003                    -(57)
NUMBER OF ACTIVE TASKS:..........000 OF MAX. 003
NUMBER OF ACTIVE TRANSACTIONS:...000 OF MAX. 003
NUMBER OF OPEN FILES:............000 OF MAX. 002
NUMBER OF ACT. TA APPLICATIONS:..000 OF MAX. 003
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
................................................................................
FUNCTION SELECTION (OR +=CONTINUE; OR BLANK; OR *END)
**END                                                                  -(58)
```

```
%  LEA0510 NORMAL TERMINATION OF LEASY MASTER PROGRAM
```

Erklärung

(1) Aufruf des Dienstprogramms LEASY-CATALOG.

(2) Neuerstellen des LEASY-Katalogs LCAT mit dem Kennwort C'LCAT'.

(3) Erstellen der Stammdatei MITABDAT mit den angegebenen Eigenschaften. Die zugehörige Schattendatei soll automatisch mitgeführt werden (AIM=(Y,A)). Die Namenssystematik für die Rekonstruktion von Schattendateien wird festgelegt (CPC=SAVE).

(4) Erstellen der Temporärdatei PROTDAT mit den angegebenen Eigenschaften. Sie soll nicht in der AIM-Datei geführt werden (AIM=N als Standardwert).

(5) Die Dateieigenschaften aller im Katalog enthaltenen Dateien sollen in vollem Umfang an der Datensichtstation ausgegeben werden.

(6) Beenden des Dienstprogramms LEASY-CATALOG.

(7) Es wird eine Schattendatei der Personaldatendatei MITABDAT mit dem Namen SAVE.MITABDAT angelegt.

(8) Es wird eine Schattendatei der zugehörigen SI-Datei mit dem Namen SAVE.MITABDAT-SI angelegt.

(9) Ausgabe der ENTER-Datei E.MTSK.3, die die Maintask startet.

(10) In der ENTER-Datei wird LEASY-MAINTASK gestartet.

(11) Der LEASY-Katalog LCAT wird zugewiesen.

(12) Es dürfen gleichzeitig 2 Dateien eröffnet werden.

(13) Die Session wird mit AIM- und BIM-Sicherung gefahren. Das Schreiben des AIM-Puffers wird von der Maintask durchgeführt

(14) Die AIM-Dateigenerationen werden automatisch auf die Schattendateien nachgezogen.

(15) Ein ENTER-JOB-Kommando zum Starten der RECONST-Task mit der ENTER-Prozedur E.RECONST.AUT wird festgelegt.

(16) Es können gleichzeitig 3 Transaktionen ablaufen.

(17) Es soll maximal 40 Sekunden auf das Freiwerden der Session gewartet werden.

(18) Starten des Batchjobs E.MTSK.3. Die Maintask und die RECONST-Task werden gestartet. Die erste LEASY-Session wird initialisiert.

(19) Die Maintask ist gestartet.

(20) Die RECONST-Task wurde von der Maintask gestartet.

(21) Auflisten der im LEASY-Katalog LCAT eingerichteten Dateien.

(22) Das Anwenderprogramm PERSDAT wird aufgerufen.

(23) Beenden des Anwenderprogramms.

(24) Die LEASY-Session wurde mit dem Dienstprogramm LEASY-MASTER mit der Funktion CLOS beendet (siehe Punkt (43)). Dies bewirkt, dass die erste AIM-Dateigeneration automatisch auf die Schattendateien nachgezogen wird. Anschließend beendet die Maintask die RECONST-Task.

(25) Neues Starten des Batchjobs E.MTSK.3 (Ausgabe des Inhalts der ENTER-Datei siehe Punkt (9)). Die Maintask und die RECONST-Task werden gestartet. Die zweite LEASY-Session wird initialisiert.

(26) Die Maintask ist gestartet.

(27) Die RECONST-Task ist gestartet.

(28) Das Anwenderprogramm PERSDAT wird aufgerufen.

(29) Beenden des Anwenderprogramms PERSDAT.

(30) Die Primärdatei LCAT.MITABDAT wird versehentlich gelöscht.

(31) Das Anwenderprogramm PERSDAT wird aufgerufen.

(32) Bei dem Versuch, auf die Dateien mit dem Anwenderprogramm zuzugreifen, bricht das Programm ab.

(33) Das HELP-MSG-Kommando zeigt die Fehlerursache.

(34) Die Datei MITABDAT wird mit der Funktion REPO im laufenden Betrieb durch ihre Schattendatei ersetzt (siehe Punkte (48) bis (55)).

(35) Das Anwenderprogramm PERSDAT wird aufgerufen.

(36) Die Dateien werden weiter verarbeitet.

(37) Ausdruck der LEASY-RECONST-Protokolle (gewählte Funktionen und Rekonstruktionsprotokoll).

(38) Protokoll der Maintask aus der zweiten Session.

(39) Protokoll der RECONST-Task aus der zweiten Session.

(40) Nach Aufruf des Dienstprogramms LEASY-MASTER wird der LEASY-Katalog zugewiesen, für den die MASTER-Funktionen ausgeführt werden sollen.

(41) Das Kennwort C'LCAT' wird angefordert.

(42) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die erste AIM-Dateigeneration beschrieben wird.

(43) Die LEASY-Session wird mit der Funktion CLOS beendet.

(44) Das Dienstprogramm LEASY-MASTER wird mit *END beendet.

(45) Nach erneutem Aufruf des Dienstprogramms LEASY-MASTER wird der LEASY-Katalog zugewiesen, für den die MASTER-Funktionen ausgeführt werden sollen.

(46) Das Kennwort C'LCAT' wird angefordert.

(47) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die zweite AIM-Dateigeneration beschrieben wird.

(48) Mit der Funktion REPO soll die Datei MITABDAT im laufenden Betrieb durch ihre Schattendatei ersetzt werden.

(49) Die Maske zur Eingabe einer Wartezeit wird aufgerufen.

(50) Es wird 1 Minute als Wartezeit festgelegt.

(51) Die Maske für die Dateiauswahl wird aufgerufen.

(52) Es soll eine Datei zur Auswahlliste hinzugefügt werden.

(53) Die Datei MITABDAT wird ausgewählt.

(54) Die Dateiauswahl wird abgeschlossen.

(55) Die Funktion REPO wird mit den angezeigten Parametern gestartet. Die Datei wird kopiert und REPO beendet sich normal.

(56) Die LEASY-Session wird mit der Funktion CLOS beendet.

(57) Es werden generelle Information ausgegeben. U.a. ist zu sehen, dass die dritte AIM-Dateigeneration beschrieben wird.

(58) Das Dienstprogramm LEASY-MASTER wird mit *END beendet.

13 Rückkehrinformationen

Die Rückkehrinformationen der LEASY-Schnittstelle werden in diesem Kapitel in mehreren Tabellen dargestellt. In untenstehender Tabelle 24 ist aufsteigend nach dem Fehlercode RC-LC sortiert. In den weiteren Tabellen (25 auf Seite 409 und 26 auf Seite 410ff.) ist aufsteigend nach der kompatiblen Rückkehrinformation von KLDS **und** nach RC-LC sortiert.

In einzelnen Fällen werden auch Meldungen des Laufzeitsystems ausgegeben, diese Meldungen sind im Handbuch „LEASY (BS2000/OSD) Dienstprogramme“ beschrieben.

LEASY-interner Fehlercode RC-LC aufsteigend sortiert

RC-LC	Bedeutung
L000	Funktion wurde ordnungsgemäß durchgeführt (bei allen Operationen)
L001	Satz mit Schlüssel nicht gefunden (RDIR, RHLD, REWR, DLET)
L002	Duplikat (RNXT, INSR bei Primär- oder Sekundärschlüssel, REWR, STOR bei Sekundärschlüssel mit DUPKEY = NO)
L003	EOF beim sequenziellen Lesen (für RNXT und RNHD bei Dateiende, für RPRI und RPHD bei Dateibeginn) oder Positionierungsfehler: sequenzieller Lesebefehl ohne aktuelles Intervall (RNXT, RNHD, RPRI, RPHD) oder EOF bei INSR bei ISAM (USAGE-Modi LOAD/PLOD/ELOD und LDUP/PLUP/ELUP)
L004	Folgefehler bei Lademodus (INSR)
L005	Satz wurde nicht gesperrt (DLET, REWR)
L006	Wartezeit bei Sperrversuch abgelaufen (LOCK, RHLD, RNHD, RPHD, INSR, STOR)
L007	Deadlock bei Sperrversuch (LOCK, RHLD, RNHD, RPHD, INSR, STOR)
L008	Freizugebender Satz kann nicht entsperrt werden, da er in der Transaktion geändert wurde (UNLK)
L009	Warnung - freizugebender Satz war nicht gesperrt (UNLK)
L010	Satzlängenfehler bei variabler Satzlänge (INSR, REWR, STOR)
L011	Warnung - mehr als 255 Sätze je Block (RNXT, RPRI; SAM) bei Verwendung der SAM-Wiedergewinnungsadresse im 24-Bit-Format
L012	Kein aktueller Satz vorhanden (REWR; SAM) oder kein gültiger Lesebefehl für Dateikennung (vor DLET ohne Schlüsselangabe)

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 1 von 8)

RC-LC	Bedeutung
L013	Schlüssel außerhalb des zulässigen Bereichs; die höchste PAM-Blocknummer des zu schreibenden Blocks muss ≤ (FILESIZE + SECONDARY ALLOCATION) sein (INSR, STOR; PAM und DAM)
L014	Rollback nicht möglich, da Transaktion ohne BIM-Sicherung
L015	openUTM: Task-Deadlock
L016	Schreiben eines Satzes einer DAM-Datei oder einer PAM-Datei mit BLOCK-CONTROL-INFO=WITHIN-DATA-BLOCK bzw. BLOCK-CONTROL-INFO=NO nicht möglich, da X'FF' im ersten Byte des Datensatzes (Löschkennzeichen für DAM). (INSR, STOR, REWR)
L017	Für den angegebenen Linknamen wurde kein /ADD-FILE-LINK Kommando abgegeben.
L018	Der Name der über /ADD-FILE-LINK Kommando zugeordneten Datei ist syntaktisch kein LEASY Katalog
L019	Beim sequenziellen Lesen über einen ISAM-Sekundärschlüssel kann der unmittelbar zuvor gelesene Satz nicht mehr gefunden werden.
L101	Datei bei OPTR dieser Transaktion nicht angegeben (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
L102	Unzulässige Operation - Widerspruch zu FCBTYPE und/oder USAGE-Modus (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
L103	Keine Transaktion eröffnet (CLTR, bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
L104	Transaktion eröffnet bei CATD bzw. DISCONNECT/openUTM
L105	Dateiname oder Zusatzname in LEASY-Katalog nicht definiert (OPFL, OPTR)
L106	USAGE-Modus mit OPEN-Modus unverträglich (OPTR nach OPFL)
L107	Für Modelldatei fehlt Zusatzangabe (OPFL, OPTR)
L108	Überlauf FILE-Tabelle (OPFL, OPTR) – Anweisung *FILE bei LEASY- MAINTASK erhöhen!
L109	Sekundärindexname im LEASY-Katalog nicht definiert (RDIR, RHLD, SETL) oder ISAM-Sekundärindex wurde bei SETL angegeben.
L110	Datei/Dateikennung kann mit dem geforderten USAGE-Modus bzw. Result-USAGE-Modus nicht eröffnet werden, da diese von anderer Transaktion schon mit stärkerem USAGE-Modus eröffnet ist (OPTR)
L111	USAGE-Modus unverträglich mit bereits eröffneter Datei/Dateikennung in derselben Transaktion
L112	KEYLEN (ISAM-Datei) > *KEY-Anweisung bei LEASY-MAINTASK (OPTR)
L113	KEYLEN > 4 bei USAGE-Modi LOAD, ELOD, PLOD, LDUP, PLUP, ELUP (OPTR; ISAM)
L114	Satzlänge mit Blocklänge nicht verträglich oder ungültige BLKSIZE (OPFL, OPTR)

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 2 von 8)

RC-LC	Bedeutung
L115	In bisherigen OPTR-Operationen dieser Transaktion wurde das geforderte Folgemerkmal für diese Datei nicht angegeben (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
L116	Kein CLFL durchgeführt (CATD nach OPFL) oder die Datei ist bereits eröffnet (OPFL)
L117	Kein CLTR durchgeführt (OPFL nach OPTR)
L118	CLFL: mindestens eine der angegebenen Dateien wurde nicht durch OPFL eröffnet
L119	Kein CLTR durchgeführt (CLFL nach OPTR)
L120	Datei (OPTR) nicht in vorangegangener Dateiliste (OPFL) angegeben (OPTR nach OPFL)
L122	Dateikennung bereits eröffnet
L123	AIM-Puffer zu klein gewählt (*AIB in LEASY-MAINTASK) im Verhältnis zu maximaler RECSIZE (OPFL, OPTR) bzw. Warmstart mit LEASY-MAINTASK ohne AIM-Sicherung, obwohl für die rückzurollende Transaktion AIM-Sicherung eingeschaltet war.
L124	2. OPTR-Aufruf ohne Verwendung von OPFL
L125	Inkonsistenz zwischen den Eintragungen im LEASY-Katalog und im DVS-Katalog
L126	Falsches Dateiformat (BLKCTRL=NO)
L130	Dateigröße übersteigt 32 GByte
LI01	CATD-Aufruf fehlt (Fremddateien nicht erlaubt)
LI02	Keine Transaktion aktiv (DCAM LU80)
LI03	Überlauf im Übergabebereich. Maximalzahl der Anwenderprogramme überschritten
LI04	IOH-interner Fehler: Wartezeit auf I/O-Task abgelaufen (*WAI-Anweisung)
LI05	IOH-interner Fehler: I/O-Task bei Bearbeitung eines LEASY-Aufrufs fehlerhaft beendet. Transaktion zurückgesetzt
LI06	IOH-interner Fehler: I/O-Task bei Bearbeitung eines LEASY-Aufrufs fehlerhaft beendet. Transaktion nicht zurückgesetzt
LI07	IOH-interner Fehler: Initialisierungsfehler. Der Common Memory ist nicht freigegeben
LI08	Versionsfehler. Interne Version mit I/O-Task unverträglich
LI09	IOH-interner Fehler: Auf Semaphor (geschützte Variable) kann nicht zugegriffen werden. Fehler bei interner Synchonisation
LI10	IOH-interner Fehler: Die Satzlänge im CINF-Bereich ist größer als in der DBL-Anweisung angegeben
LI11	Datei nicht in OPF-Anweisung angegeben
LI12	Satzlänge gleich 0 oder größer als Wert der ARL-Anweisung angegeben
LI20	Versionen von Laufzeitsystem und I/O-Task passen nicht zusammen
LI26	Version des Verbindungsmoduls < V5.1

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 3 von 8)

RC-LC	Bedeutung
LP01	Operationscode falsch (bei allen Operationen)
LP02	Zu wenig Operanden (bei allen Operationen)
LP04	OPE1/OPE2 falsch (CLTR)
LP06	USAGE-Modus falsch oder unzulässig (OPTR)
LP07	OPEN-Modus falsch oder unzulässig: OPFL: Fremddatei, SHAREUP=YES, BIM=YES, Open-Modus für Schreiben. OPTR: Usage-Modus passt nicht zum Open-Modus.
LP08	Feldauswahl falsch, '(ALL)' (SETL, RDIR, RHLD)
LP09	Syntaxfehler in der Dateiliste (OPFL, OPTR, CLFL)
LP10	Syntaxfehler im Katalognamen (CATD)
LP11	CI-Bereich zu klein für Currency Information (CINF) oder keine Information im CI-Bereich (ci-slf=0)
LP12	L-OPT falsch, ≠'1' (bei allen Operationen)
LP14	PAMHPNR/SAMPTR ungültig (bei allen Operationen, bei denen diese Felder ausgewertet werden)
LP15	OPE-WTIME nicht numerisch (bei allen Operationen)
LP16	OPE-OM im RE-Bereich falsch besetzt
LP17	Ungültige Kombination von (KB, KE) (SETL, RDIR bei SAM-Datei)
LP18	Syntaxfehler in der Dateikennung (bei allen Operationen mit Angabe von DB1)
LP19	OPE-STX falsch (CATD)
LP20	Die Länge des USER-Bereiches liegt nicht im Bereich 5 < len < 1024
LS01	Common Memory CMMAIN von Maintask für angegebenen LEASY-Katalog nicht eingerichtet (CATD, OPTR)
LS02	Operation wegen CLOS- oder SHUT-Funktion abgelehnt (CATD, OPFL, OPTR)
LS03	Zu viele Transaktionen - Überlauf Transaktionstabelle (OPTR) Anweisung *TRANS bei LEASY-MAINTASK erhöhen!
LS04	Common Memory CMMAIN ist für das Laufzeitsystem gesperrt (*USE=R in LEASY-MAINTASK)
LS05	Wegen HOLD-Funktion momentan überhaupt keine Operation möglich
LS06	Wegen QUIE-Funktion derzeit keine neue Transaktion möglich
LS07	Wegen LOCT- oder QUIE-Funktion derzeit keine Operation für diese Transaktion möglich
LS08	Rollback wegen zweitem LS12
LS09	Wegen SHUT-, CLOS-, RLBT- oder REPO-Funktion wird bei CLTR-Operationen OPE2=T ignoriert

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 4 von 8)

RC-LC	Bedeutung
LS10	Operation wegen RLBT- oder SHUT-Funktion in CLTR mit OPE1=R umgewandelt
LS11	Virtueller Speicher erschöpft (bei REQM, ENAMP Makro)
LS12	Überlauf des Bereichs für Transaktionselemente (bei OPTR) oder des Bereichs für Sperrlistenelemente beim Versuch, neues Sperrelement anzulegen Anweisung *MEM bei LEASY-MAINTASK erhöhen!
LS13	Die Datei ist vom Dienstprogramm LEASY-MASTER geperrt (OPFL und OPTR)
LS14	Die Datei ist vom Dienstprogramm LEASY-MASTER gegen schreibende Eröffnung gesperrt (OPFL und OPTR)
LS15	Überlauf Task-Tabelle, *TSK-Operand im Dienstprogramm LEASY-MAINTASK erhöhen
LS17	Fehler bei JV-Funktion
LS18	DVS-Fehler bei Katalog-Datei
LS19	DVS-Fehler bei SI-Datei
LS20	Allgemeiner DVS-Fehler
LS21	DVS-Fehler bei BIM-Datei
LS22	DVS-Fehler bei AIM-Datei
LS23	Fehler bei Rollback (CLTR,OPE1=R)
LS26	Version des Verbindungsmoduls < V5.1
LS30	Fehler beim STXIT-Makro im Modul LEASY
LS31	Fehler beim dynamischen Nachladen eines Moduls
LS32	Fehler beim Makro ENASI
LS33	Fehler beim Makro RELM
LS34	Fehler beim Makro DISSI
LS35	Fehler beim Makro ENAMP
LS36	Version des Moduls LEACON ist unverträglich mit der Version des Moduls LEASY
LS37	Fehler beim Makro ENQAR
LS38	Fehler beim Makro DEQAR
LS40	LEASY-Systemfehler: angelegtes Sperrelement nicht mehr gefunden
LS41	LEASY-Systemfehler: Interne Sperre für Satzsplitting in Sekundärindexdatei hängengeblieben
LS42	LEASY-Systemfehler: Duplikat in Sekundärindexdatei bei Satzsplitting

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 5 von 8)

RC-LC	Bedeutung
LS43	Dateiinkonsistenz zwischen Primär- und Sekundärindexdatei: zu SI-Eintrag kein Primärsatz vorhanden oder er enthält einen falschen Sekundärschlüsselwert. Satz mit Primärschlüssel nicht vorhanden. Satz vorhanden, aber ungültiges Satzartenfeld Satz vorhanden, aber enthält nicht SI-Schlüssel.
LS44	Strukturfehler in BIM-Datei (beim Rollback)
LS45	LEASY-Systemfehler: Inkonsistenz in Common Memory (interne Sekundärindexnummer nicht gefunden)
LS47	LEASY-Systemfehler: Logikfehler in LEAWRAIM
LS48	LEASY-Systemfehler: MVC-Sperre hängt in LEAWRAIM
LS49	LEASY-Systemfehler: WRT-Sperre hängt in LEAWRAIM
LS51	LEASY-Systemfehler: AIM-Puffer ist voll und kann wegen eines Fehlers bei PAM-WRITE nicht entleert werden
LS52	Strukturfehler in PAM-Datei
LS53	LEASY-Systemfehler: AIMSWITCH-Sperre hängt in LEALAISW
LS54	LEASY-Systemfehler: Open-File-Tabellen-Sperre hängt in LEASPERR
LS55	LEASY-Systemfehler: Transaktions-Tabellen-Sperre hängt in LEASPERR
LS56	LEASY-Systemfehler: Freiketten-Sperre hängt in LEASPERR
LS57	LEASY-Systemfehler: Freigabe-Sperre hängt in LEASPERR
LS58	LEASY-Systemfehler: Datei-Tabellensperre hängt in LEAFTIN
LS59	Fehler beim Schreiben eines DAM-Datenblocks: Fehler bei der SI- oder AIM-Behandlung erzwang einen automatischen Rollback der Transaktion (CLTR, alle Operationen, die im 3. Operanden eine Dateikennung angeben).
LS60	LEASY-Systemfehler: Sperre der Deadlock-Bitmatrix hängt
LS61	Fehler beim Makro ENAEI
LS62	Fehler beim Makro ENACO
LS63	Fehler beim Makro SOLSIG
LS64	Fehler beim Makro POSSIG
LS65	Maintask fehlerhaft beendet (z.B. beim Schreiben des AIM-Puffers auf Band)
LS66	LEASY-Systemfehler: Fehler beim Verkürzen von AIM-Sätzen
LS67	LEASY-Systemfehler: falsche Aufrufversorgung für Modul LEAKMP
LS68	Version des Laufzeitsystems ungleich Version des Common Memory CMMAIN
LS69	Fehler beim Makro DISMP
LS70	Fehler beim Makro DISEI

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 6 von 8)

RC-LC	Bedeutung
LS71	Fehler beim Makro CREPOOL (für NK-ISAM)
LS72	Fehler beim Makro DELPOOL (für NK-ISAM)
LS73	Fehler beim Makro ADDPLNK (für NK-ISAM)
LS74	Fehler beim Makro REMPLNK (für NK-ISAM)
LS75	Die LEASY-Anweisung kann nicht bearbeitet werden. Die AIM-Dateigeneration hat die maximal mögliche Größe erreicht oder sie kann nicht umgeschaltet werden (aus Systemgründen, z.B. Pubspace-Limit erreicht, oder weil keine AIM-Dateigeneration frei ist und in der AIS-Anweisung von LEASY-MAINTASK der Wert 0 als Inkrement angegeben war).
LS76	Transaktionssemaphor konnte nicht erworben werden.
LS77	Wegen ROMS-Funktion derzeit keine bestandsverändernden LEASY-Anweisungen (DLET, INSR, REWR, STOR) möglich.
LS78	Wegen REPO keine neuen Transaktionen erlaubt.
LS79	Transaktion wegen READONLY-Modus (LEASY-MASTER, ROMS) oder Kopieren von Schattendateien (LEASY-MASTER,REPO) bereits zurückgesetzt.
LS80	Wegen Funktion REPO keine Anweisungen außer CLTR erlaubt.
LS81	AIM-Datei kann wegen eines Fehlers nicht mehr beschrieben werden, keine weitere LEASY-Anforderung mehr erlaubt, Transaktion wurde von LEASY zurücksetzt.
LU01	openUTM: falscher Startoperand
LU02	openUTM: Syntaxfehler bei Startoperand
LU10	openUTM: keine oder zu wenige Startoperanden DCAM: Fehler in der Operationsfolge beim Start (CATD und/oder OPFL fehlen)
LU11	openUTM/DCAM: weniger als 2 LEASY-Operanden
LU12	openUTM/DCAM: OPEN-Modus unzulässig für Fremd- oder SAM-Datei (Datei darf nur gelesen werden) (OPFL)
LU13	openUTM/DCAM: LEASY-Temporärdatei verboten (OPFL)
LU14	openUTM: nach verzögertem CLTR kein CALL-LEASY im selben Dialogschritt erlaubt (alle Operationen)
LU15	openUTM: In Transaktionen ohne BIM-Sicherung darf Datei nicht zum Schreiben eröffnet werden (OPTR)
LU16	openUTM/DCAM: Fehler in der Intertask-Synchronisation für OPFL bzw. CLFL oder unterschiedliche Reihenfolge bei OPFL
LU17	DCAM: Fehler. Offene Transaktion innerhalb der DCAM-Anwendung bei OPFL oder CLFL
LU18	DCAM: Fehler. Transaktion kann nicht gleichzeitig in mehreren Tasks aktiv sein
LU50	openUTM/DCAM: Überlauf Anwendungstabelle

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 7 von 8)

RC-LC	Bedeutung
LU51	openUTM/DCAM: Inkonsistenz in der Anwendungstabelle
LU52	openUTM/DCAM: Interne Sperre der Anwendungstabelle ist hängengeblieben
LU53	DVS-Fehler bei Status-Datei
LU54	openUTM: Status-Abfrage zur aktuellen LEASY-Session mit noch offenen Transaktionen der openUTM-Anwendung
LU80	openUTM: Fehlerhafte Aufruffolge an der openUTM-Datenbank-Schnittstelle DCAM: DCAM-Anwendungsname fehlt (CATD), Transaktions-Identifikation fehlt oder ist fehlerhaft (bei allen Operationen innerhalb einer LEASY-Transaktion)
LU81	openUTM: OPFL-Aufruf fehlt (OPTR)
LU82	openUTM: Startoperand beginnt nicht mit ".LEASY_"
LU83	openUTM: Operationscode falsch
LU84	openUTM: Status-Aufruf: Operationscode weder "Abfrage" noch "Löschen"
LU85	openUTM: Fehler bei der Behandlung vorläufig beendeter Transaktionen
Addd	DVS-Fehler bei Bearbeitung einer AIM-Datei
Bddd	DVS-Fehler bei Bearbeitung einer BIM-Datei
Cddd	DVS-Fehler bei Bearbeitung einer Katalogdatei
Dddd	DVS-Fehler bei Bearbeitung einer Primärdatei
Jddd	Fehler bei Bearbeitung einer Jobvariablen (JV)
Sddd	DVS-Fehler bei Bearbeitung einer Sekundärindexdatei
Tddd	DVS-Fehler bei Bearbeitung einer LEASY-Statusdatei

Tabelle 24: LEASY-interner Fehlercode RC-LC aufsteigend sortiert (Teil 8 von 8)

Meldungen der LEASY-Schnittstelle

Die folgenden Meldungen der LEASY-Schnittstelle sind nach dem kompatiblen Returncode sortiert.

Returncodes zur Programmsteuerung

RC-CC	RC-LC	
000	L000 Funktion wurde ordnungsgemäß durchgeführt (bei allen Operationen)	
010	Es gibt keinen Satz, der der Suchbedingung genügt	
	L001	Satz mit Schlüssel nicht gefunden (RDIR, RHLD, REWR, DLET)
	L003	EOF beim sequenziellen Lesen (für RNXT und RNHD bei Dateiende, für RPRI und RPHD bei Dateibeginn) oder Positionierungsfehler: sequenzieller Lesebefehl ohne aktuelles Intervall (RNXT, RNHD, RPRI, RPHD) oder EOF bei INSR bei ISAM (USAGE-Modi LOAD/PLOD/ELOD und LDUP/PLUP/ELUP)
051	Inhalt des Ordnungsfeldes ist bereits in der Datei enthalten oder außerhalb des zulässigen Bereichs	
	L002	Duplikat (RNXT, INSR bei Primär- oder Sekundärschlüssel, REWR, STOR bei Sekundärschlüssel mit DUPEKY= NO)
	L013	Schlüssel außerhalb des zulässigen Bereichs; die höchste PAM-Blocknummer des zu schreibenden Blocks muss ≤ (FILESIZE + SECONDARY ALLOCATION) sein (INSR, STOR; PAM und DAM)

Tabelle 25: Returncodes zur Programmsteuerung

Returncodes zur Fehleranzeige

RC-CC	RC-LC	
01A	Satz wurde vor Änderung nicht festgehalten	
	L005	Satz wurde nicht gesperrt (DLET, REWR)
	L012	Kein aktueller Satz vorhanden (REWR; SAM) oder kein gültiger Lesebefehl für Dateikennung (vor DLET ohne Schlüsselangabe)
01B	Änderung des Inhalts des Ordnungsfeldes	
	L012	Kein aktueller Satz vorhanden (REWR; SAM) oder kein gültiger Lesebefehl für Dateikennung (vor DLET ohne Schlüsselangabe)
02A	Ein-Ausgabefehler des Systems	
	Dddd	E/A Fehler bei der Bearbeitung einer Primärdatei
	LS20	E/A-Fehler
031	L003	EOF beim sequenziellen Lesen (für RNXT und RNHD bei Dateiende, für RPRI und RPHD bei Dateibeginn) oder Positionierungsfehler: sequenzieller Lesebefehl ohne aktuelles Intervall (RNXT, RNHD, RPRI, RPHD) oder EOF bei INSR bei ISAM (USAGE-Modi LOAD/PLOD/ELOD und LDUP/PLUP/ELUP)
043	Ungültiger Dateiname	
	L105	Dateiname oder Zusatzname in LEASY-Katalog nicht definiert (OPFL, OPTR)
	L107	Für Modelldatei fehlt Zusatzangabe (OPFL, OPTR)
	LP18	Syntaxfehler in der Dateikennung (bei allen Operationen mit Angabe DB1)
	LU13	openUTM/DCAM: LEASY-Temporärdatei verboten (OPFL)
04A	Eintrag in Feldauswahl ungültig	
	LP08	Feldauswahl falsch, '(ALL)' (SETL, RDIR, RHLD)
04B	Ungültiger Operationsschlüssel	
	LP01	Operationscode falsch (bei allen Operationen)

Tabelle 26: Returncodes zur Fehleranzeige (Teil 1 von 10)

RC-CC	RC-LC	
04D	Ungültige Operationsergänzung	
	LP04	OPE1/OPE2 falsch (CLTR)
	LP06	USAGE-Modus falsch oder unzulässig (OPTR)
	LP07	OPEN-Modus falsch oder unzulässig: OPFL: Fremddatei, SHAREUP=YES, BIM=YES, Open-Modus für Schreiben. OPTR: Usage-Modus passt nicht zum Open-Modus
	LP12	L-OPT falsch, ≠'1' (bei allen Operationen)
	LP14	PAMHPNR/SAMPTR ungültig (bei allen Operationen, bei denen diese Felder ausgewertet werden)
	LP15	OPE-WTIME nicht numerisch (bei allen Operationen)
	LP16	OPE-OM im RE-Bereich falsch besetzt
	LP19	OPE-STX falsch (CATD)
04X	Formalfehler im LEASY-Aufruf	
	LP02	Zu wenig Operanden (bei allen Operationen)
	LP10	Syntaxfehler im Katalognamen (CATD)
	LP11	CI-Bereich zu klein für Currency Information (CINF) oder keine Information im CI-Bereich (ci-slf=0)
	LP20	Die Länge des USER-Bereiches liegt nicht im Bereich 5 < len < 1024
04Y	Syntaxfehler bei Mehrfachoperanden	
	LP09	Syntaxfehler in der Dateiliste (OPFL, OPTR, CLFL)
05A	Ungültige Suchbedingung	
	L109	Sekundärindexname im LEASY-Katalog nicht definiert (RDIR, RHLD, SETL) oder ISAM-Sekundärindex wurde bei SETL angegeben.
	LP17	Ungültige Kombination von (KB, KE) (SETL, RDIR bei SAM-Datei)
07A	Systemfehler bei DLET	
	Addd Bddd Dddd Sddd	DVS-Fehler bei Bearbeitung einer AIM-Datei DVS-Fehler bei Bearbeitung einer BIM-Datei DVS-Fehler bei Bearbeitung einer Primärdatei DVS-Fehler bei Bearbeitung einer Sekundärindexdatei
	LSxx	Systemfehler, wie sie bei RC-CC='99A' mit RC-LC='LSxx' beschrieben sind, können auch hier auftreten
07B	Systemfehler bei INSR oder STOR RC-LC Wie bei RC-CC = É	
07C	Systemfehler bei REWR RC-LC Wie bei RC-CC = '07A'	

Tabelle 26: Returncodes zur Fehleranzeige (Teil 2 von 10)

RC-CC	RC-LC	
091	Datei nicht verfügbar	
	L101	Datei bei OPTR dieser Transaktion nicht angegeben (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
	L103	Keine Transaktion eröffnet (CLTR, bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
	L115	In bisherigen OPTR-Operationen dieser Transaktion wurde das geforderte Folgemerkmal für diese Datei nicht angegeben (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
	L118	CLFL: mindestens eine der angegebenen Dateien wurde nicht durch OPFL eröffnet
	L120	Datei (OPTR) nicht in vorangegangener Dateiliste (OPFL) angegeben (OPTR nach OPFL)
	L126	Falsches Dateiformat (BLKCTRL=NO)
	L130	Dateigröße übersteigt 32 GByte
	LU15	openUTM: In Transaktionen ohne BIM-Sicherung darf Datei nicht zum Schreiben eröffnet werden (OPTR)
092	Ungültiger Modus in der Dateibehandlung (z.B. falsche Reihenfolge der Operationen)	
	L102	Unzulässige Operation - Widerspruch zu FCBTYPE und/oder USAGE-Modus (bei allen Operationen, die im 3. Operanden eine Dateikennung angeben)
	L106	USAGE-Modus mit OPEN-Modus unverträglich (OPTR nach OPFL)
	L119	Kein CLTR durchgeführt (CLFL nach OPTR)
	L124	2. OPTR-Aufruf ohne Verwendung von OPFL
	LU12	openUTM/DCAM: OPEN-Modus unzulässig für Fremd- oder SAM-Datei (Datei darf nur gelesen werden) (OPFL)
	LU14	openUTM: nach verzögertem CLTR kein CALL-LEASY im selben Dialogschritt erlaubt (alle Operationen)
093	Datei bereits angemeldet	
	L104	Transaktion eröffnet bei CATD bzw. DISCONNECT/openUTM
	L116	Kein CLFL durchgeführt (CATD nach OPFL) oder die Datei ist bereits eröffnet (OPFL)
	L117	Kein CLTR durchgeführt (OPFL nach OPTR)
	L122	Dateikennung bereits eröffnet
094	Fehler bei MARK	
	Lxxx	Fehler, wie sie bei RC-CC='99A' mit RC-LC='xxx' beschrieben sind, können auch hier auftreten

Tabelle 26: Returncodes zur Fehleranzeige (Teil 3 von 10)

RC-CC	RC-LC	
095	Fehler bei BACK	
	L014	Rollback nicht möglich, da Transaktion ohne BIM-Sicherung
	Lxxx	Fehler, wie sie bei RC-CC='99A' mit RC-LC='xxx' beschrieben sind, können auch hier auftreten
802	Freier Speicherplatz in Datei erschöpft	
	Dddd	Speicherplatz für Primärdatei erschöpft
	LS20	Speichermangel bei Datei
99A	Sonstige Fehler	
	Addd	DVS-Fehler bei Bearbeitung einer AIM-Datei
	Bddd	DVS-Fehler bei Bearbeitung einer BIM-Datei
	Cddd	DVS-Fehler bei Bearbeitung einer Katalogdatei
	Dddd	DVS-Fehler bei Bearbeitung einer Primärdatei
	Jddd	Fehler bei Bearbeitung einer Jobvariablen (JV)
	Sddd	DVS-Fehler bei Bearbeitung einer Sekundärindexdatei
	Tddd	DVS-Fehler bei Bearbeitung einer LEASY-Statusdatei
	L004	Folgefehler bei Lademodus (INSR)
	L006	Wartezeit bei Sperrversuch abgelaufen (LOCK, RHLD, RNHD RPHD, INSR, STOR)
	L007	Deadlock bei Sperrversuch (LOCK, RHLD, RNHD, RPHD, INSR, STOR)
	L008	Freizugebender Satz kann nicht entsperrt werden, da er in der Transaktion geändert wurde (UNLK)
	L009	Warnung - freizugebender Satz war nicht gesperrt (UNLK)
	L010	Satzlängenfehler bei variabler Satzlänge (INSR, REWR, STOR)
	L011	Warnung - mehr als 255 Sätze je Block (RNXT, RPRI; SAM) bei Verwendung der SAM-Wiedergewinnungsadresse im 24-Bit-Format
	L014	Rollback nicht möglich, da Transaktion ohne BIM-Sicherung
	L015	openUTM: Task-Deadlock
	L016	Schreiben eines Satzes einer DAM-Datei oder einer PAM-Datei mit BLOCK-CONTROL-INFO=WITHIN-DATA-BLOCK bzw. BLOCK-CONTROL-INFO=NO nicht möglich, da X'FF' im ersten Byte des Datensatzes (Löschkennzeichen für DAM). (INSR, STOR, REWR)
	L017	Für den angegebenen Linknamen wurde kein /ADD-FILE-LINK Kommando abgegeben
	L018	Der Name der über /ADD-FILE-LINK Kommando zugeordneten Datei ist syntaktisch kein LEASY Katalog

Tabelle 26: Returncodes zur Fehleranzeige (Teil 4 von 10)

RC-CC	RC-LC	
99A	L019	Beim sequenziellen Lesen über einen ISAM- Sekundärschlüssel kann der unmittelbar zuvor gelesene Satz nicht mehr gefunden werden.
	L108	Überlauf FILE-Tabelle (OPTR) – Anweisung *FILE in LEASY-MAINTASK erhöhen!
	L110	Datei/Dateikennung kann mit dem geforderten USAGE-Modus bzw. Result-USAGE-Modus nicht eröffnet werden, da diese von anderer Transaktion schon mit stärkerem USAGE-Modus eröffnet ist (OPTR)
	L111	USAGE-Modus unverträglich mit bereits eröffneter Datei/Dateikennung in derselben Transaktion
	L112	KEYLEN (ISAM-Datei) > *KEY-Anweisung bei LEASY-MAINTASK (OPTR)
	L113	KEYLEN > 4 bei USAGE-Modi LOAD, ELOD, PLOD, LDUP, PLUP, ELUP (OPTR; ISAM)
	L114	Satzlänge mit Blocklänge nicht verträglich oder ungültige BLKSIZE (OPFL, OPTR)
	L116	Kein CLFL durchgeführt (CATD nach OPFL) oder die Datei ist bereits eröffnet (OPFL)
	L123	AIM-Puffer zu klein gewählt (*AIB in LEASY-MAINTASK) im Verhältnis zu maximaler RECSIZE (OPFL, OPTR) bzw. Warmstart mit LEASY-MAINTASK ohne AIM-Sicherung, obwohl für die rückzurollende Transaktion AIM-Sicherung eingeschaltet war
	L125	Inkonsistenz zwischen den Eintragungen im LEASY-Katalog und im DVS-Katalog
	LS01	Common Memory CMMAIN von Maintask für angegebenen LEASY-Katalog nicht eingerichtet (CATD, OPTR)
	LS02	Operation wegen CLOS- oder SHUT-Funktion abgelehnt (CATD, OPFL, OPTR)
	LS03	zu viele Transaktionen - Überlauf Transaktionstabelle (OPTR) Anweisung *TRANS bei LEASY-MAINTASK erhöhen!
	LS04	Common Memory CMMAIN ist für das Laufzeitsystem gesperrt (*USE=R in LEASY-MAINTASK)
	LS05	Wegen HOLD-Funktion momentan überhaupt keine Operation möglich
	LS06	Wegen QUIE-Funktion derzeit keine neue Transaktion möglich
	LS07	Wegen LOCT- oder QUIE-Funktion derzeit keine Operation für diese Transaktion möglich
	LS08	Rollback wegen zweitem LS12
	LS09	Wegen SHUT-, CLOS-, RLBT- oder REPO-Funktion wird bei CLTR-Operationen OPE2=T ignoriert

Tabelle 26: Returncodes zur Fehleranzeige (Teil 5 von 10)

RC-CC	RC-LC	
99A	LS10	Operation wegen RLBT- oder SHUT-Funktion in CLTR mit OPE1=R umgewandelt
	LS11	Virtueller Speicher erschöpft (bei REQM, ENAMP Makro)
	LS12	Überlauf des Bereichs für Transaktionselemente (bei OPTR) oder des Bereichs für Sperrlistenelemente beim Versuch, neues Sperrelement anzulegen. Anweisung *MEM bei LEASY-MAINTASK erhöhen!
	LS13	Die Datei ist vom Dienstprogramm LEASY-MASTER gesperrt (OPFL und OPTR)
	LS14	Die Datei ist vom Dienstprogramm LEASY-MASTER gegen schreibende Eröffnung gesperrt (OPFL und OPTR)
	LS15	Überlauf Task-Tabelle, *TSK-Operand im Dienstprogramm LEASY-MAINTASK erhöhen
	LS17	Fehler bei JV-Funktion
	LS18	DVS-Fehler bei Katalog-Datei
	LS19	DVS-Fehler bei SI-Datei
	LS20	Allgemeiner DVS-Fehler
	LS21	DVS-Fehler bei BIM-Datei
	LS22	DVS-Fehler bei AIM-Datei
	LS23	Fehler bei Rollback (CLTR,OPE1=R)
	LS26	Version des Verbindungsmoduls < V 5.1
	LS30	Fehler beim STXIT-Makro im Modul LEASY
	LS31	Fehler beim dynamischen Nachladen eines Moduls
	LS32	Fehler beim Makro ENASI
	LS33	Fehler beim Makro RELM
	LS34	Fehler beim Makro DISSI
	LS35	Fehler beim Makro ENAMP
	LS36	Version des Moduls LEACON ist unverträglich mit der Version des Moduls LEASY
	LS37	Fehler beim Makro ENQAR
	LS38	Fehler beim Makro DEQAR
	LS40	LEASY-Systemfehler: angelegtes Sperrelement nicht mehr gefunden
	LS41	LEASY-Systemfehler: Interne Sperre für Satzsplitting in Sekundärdatei hängengeblieben
	LS42	LEASY-Systemfehler: Duplikat in Sekundärindexdatei bei Satzsplitting

Tabelle 26: Returncodes zur Fehleranzeige (Teil 6 von 10)

RC-CC	RC-LC	
99A	LS43	Dateiinkonsistenz zwischen Primär- und Sekundärindexdatei: zu SI-Eintrag kein Primärsatz vorhanden oder er enthält einen falschen Sekundärschlüsselwert. Satz mit Primärschlüssel nicht vorhanden. Satz vorhanden, aber ungültiges Satzartenfeld Satz vorhanden, aber enthält nicht SI-Schlüssel.
	LS44	Strukturfehler in BIM-Datei (beim Rollback)
	LS45	LEASY-Systemfehler: Inkonsistenz in Common Memory (interne Sekundärindexnummer nicht gefunden)
	LS47	LEASY-Systemfehler: Logikfehler in LEAWRAIM
	LS48	LEASY-Systemfehler: MVC-Sperre hängt in LEAWRAIM
	LS49	LEASY-Systemfehler: WRT-Sperre hängt in LEAWRAIM
	LS51	LEASY Systemfehler: AIM-Puffer ist voll und kann wegen eines Fehlers bei PAM-WRITE nicht entleert werden
	LS52	Strukturfehler in PAM-Datei
	LS53	LEASY-Systemfehler: AIMSWITCH-Sperre hängt in LEALAISW
	LS54	LEASY-Systemfehler: Open-File-Tabellen-Sperre hängt in LEASPERR
	LS55	LEASY-Systemfehler: Transaktions-Tabellen-Sperre hängt in LEASPERR
	LS56	LEASY-Systemfehler: Freiketten-Sperre hängt in LEASPERR
	LS57	LEASY-Systemfehler: Freigabe-Sperre hängt in LEASPERR
	LS58	LEASY-Systemfehler: Datei-Tabellensperre hängt in LEAFTIN
	LS59	Fehler beim Schreiben eines DAM-Datenblocks: Fehler bei der SI- oder AIM-Behandlung erzwang einen automatischen Rollback der Transaktion (CLTR, alle Operationen, die im 3. Operanden eine Dateikennung angeben).
	LS60	LEASY-Systemfehler: Sperre der Deadlock-Bitmatrix hängt
	LS61	Fehler beim Makro ENAEI
	LS62	Fehler beim Makro ENACO
	LS63	Fehler beim Makro SOLSIG
	LS64	Fehler beim Makro POSSIG
	LS65	Maintask fehlerhaft beendet (z.B. beim Schreiben des AIM-Puffers auf Band)
	LS66	LEASY-Systemfehler: Fehler beim Verkürzen von AIM-Sätzen
	LS67	LEASY-Systemfehler: falsche Aufrufversorgung für Modul LEAKMP
	LS68	Version des Laufzeitsystems ungleich Version des Common Memory CMMAIN

Tabelle 26: Returncodes zur Fehleranzeige (Teil 7 von 10)

RC-CC	RC-LC	
99A	LS71	Fehler beim Makro CREPOOL (für NK-ISAM) Im Feld RC-LCE wird der Haupt-Returncode des Makros ausgegeben (siehe Handbuch „DVS-Makros“)
	LS72	Fehler beim Makro DELPOOL (für NK-ISAM) Im Feld RC-LCE wird der Haupt-Returncode des Makros ausgegeben (siehe Handbuch „DVS-Makros“)
	LS73	Fehler beim Makro ADDPLNK (für NK-ISAM) Im Feld RC-LCE wird der Haupt-Returncode des Makros ausgegeben (siehe Handbuch „DVS-Makros“)
	LS74	Fehler beim Makro REMPLNK (für NK-ISAM) Im Feld RC-LCE wird der Haupt-Returncode des Makros ausgegeben (siehe Handbuch „DVS-Makros“)
	LS75	Die LEASY-Anweisung kann nicht bearbeitet werden. Die AIM-Dateigeneration hat die maximal mögliche Größe erreicht oder sie kann nicht umgeschaltet werden (aus Systemgründen, z.B. Pubspace-Limit erreicht, oder weil keine AIM-Dateigeneration frei ist und in der AIS-Anweisung von LEASY-MAINTASK der Wert 0 als Inkrement angegeben war).
	LS76	Transaktionssemaphor konnte nicht erworben werden.
	LS77	Wegen ROMS-Funktion derzeit keine bestandsverändernden LEASY-Anweisungen (DLET, INSR, REWR, STOR) möglich.
	LS78	Wegen REPO keine neuen Transaktionen erlaubt.
	LS79	Transaktion wegen READONLY-Modus (LEASY-MASTER, ROMS) oder Kopieren von Schattendateien (LEASY-MASTER,REPO) bereits zurückgesetzt.
	LS80	Wegen Funktion REPO keine Anweisungen außer CLTR erlaubt.
	LS81	AIM-Datei kann wegen eines Fehlers nicht mehr beschrieben werden, keine weitere LEASY-Anforderung mehr erlaubt, Transaktion wurde von LEASY zurücksetzt.
UTM	Fehler bei openUTM-Betrieb	
	LU01	Falscher Startparameter
	LU02	Syntaxfehler bei Startparameter
	LU10	Keine oder zu wenige Startparameter
	LU11	Weniger als 2 LEASY-Operanden
	LU12	openUTM/DCAM: OPEN-Modus unzulässig für Fremd- oder SAM-Datei (Datei darf nur gelesen werden) (OPFL)
	LU13	openUTM/DCAM: LEASY-Temporärdatei verboten (OPFL)
	LU14	openUTM: nach verzögertem CLTR kein CALL-LEASY im selben Dialogschritt erlaubt (alle Operationen)
	LU15	openUTM: In Transaktionen ohne BIM-Sicherung darf Datei nicht zum Schreiben eröffnet werden (OPTR)

Tabelle 26: Returncodes zur Fehleranzeige (Teil 8 von 10)

RC-CC	RC-LC	
UTM	LU16	Fehler in der Intertask-Synchronisation für OPFL bzw. CLFL oder unterschiedliche Reihenfolge der Dateinamen bei OPFL
	LU50	Überlauf Anwendungstabelle
	LU51	Inkonsistenz in der Anwendungstabelle
	LU52	Interne Sperre der Anwendungstabelle ist hängengeblieben
	LU53	DVS-Fehler bei Status-Datei
	LU54	Status-Abfrage zur aktuellen LEASY-Session mit noch offenen Transaktionen der openUTM-Anwendung
	LU80	Fehlerhafte Aufruffolge an der openUTM-Datenbank-Schnittstelle
	LU81	OPFL Aufruf fehlt (OPTR)
	LU82	Startparameter beginnt nicht mit ".LEASY_"
	LU83	Operationscode falsch
	LU84	Status-Aufruf: Operationscode weder "Abfrage" noch "Löschen"
	LU85	Fehler bei der Behandlung vorläufig beendeter Transaktionen
DCA	Fehler bei DCAM-Betrieb	
	LU10	Fehler in der Operationsfolge beim Start (CATD und/oder OPFL fehlen)
	LU11	Weniger als 2 LEASY-Operanden
	LU12	openUTM/DCAM: OPEN-Modus unzulässig für Fremd- oder SAM-Datei (Datei darf nur gelesen werden) (OPFL)
	LU13	openUTM/DCAM: LEASY-Temporärdatei verboten (OPFL)
	LU16	Fehler in der Intertask-Synchronisation für OPFL bzw. CLFL oder unterschiedliche Reihenfolge der Dateinamen bei OPFL
	LU17	Fehler: Offene Transaktion innerhalb der DCAM-Anwendung bei OPFL oder CLFL
	LU18	Fehler: Transaktion kann nicht gleichzeitig in mehreren Tasks aktiv sein
	LU50	Überlauf Anwendungstabelle
	LU51	Inkonsistenz in der Anwendungstabelle
	LU52	Interne Sperre der Anwendungstabelle ist hängengeblieben
	LU80	DCAM-Anwendungsname fehlt (CATD), Transaktions-Identifikation fehlt oder ist fehlerhaft (bei allen Operationen innerhalb einer LEASY-Transaktion)
DRV	Fehler bei DRIVE-Aufruf	
	LD01	Kein freier Entry in DRIVE-User-Tabelle
	LD02	Entry bereits in DRIVE-User-Tabelle
	LD03	Entry nicht in DRIVE-User-Tabelle

Tabelle 26: Returncodes zur Fehleranzeige (Teil 9 von 10)

RC-CC	RC-LC	
IOH	Fehler im I/O-Handler	
	L104	CATD-Fehler: CATD bei offener Transaktion
	LI01	CATD Aufruf fehlt (Fremddateien nicht erlaubt)
	LI02	Keine Transaktion aktiv (DCAM LU80)
	LI03	Überlauf im Übergabebereich. Maximalzahl der Anwenderprogramme überschritten
	LI04	IOH-interner Fehler: Wartezeit auf I/O-Task abgelaufen (*WAI-Anweisung)
	LI05	IOH-interner Fehler: I/O-Task bei Bearbeitung eines LEASY-Aufrufs fehlerhaft beendet. Transaktion zurückgesetzt
	LI06	IOH-interner Fehler: I/O-Task bei Bearbeitung eines LEASY-Aufrufs fehlerhaft beendet. Transaktion nicht zurückgesetzt.
	LI07	IOH-interner Fehler: Initialisierungsfehler. Der Common Memory ist nicht freigegeben
	LI08	Versionsfehler. Interne Version mit I/O-Task unverträglich
	LI09	IOH-interner Fehler: Auf Semaphor (geschützte Variable) kann nicht zugegriffen werden. Fehler bei interner Synchronisation
	LI10	IOH-interner Fehler: Die Satzlänge im CINF-Bereich ist größer als in der *DBL-Anweisung angegeben
	LI12	Satzlänge gleich 0 oder größer als Wert der ARL-Anweisung angegeben
	LI20	Versionen von Laufzeitsystem und I/O-Task passen nicht zusammen
	LI26	Version des Verbindungsmoduls < V5.1
	LP02	CATD-Fehler: CATD mit zuwenig Operanden
	LP10	CATD-Fehler: Syntaxfehler im Katalognamen
	LS01	Common Memory CMMAIN von Maintask für angegebenen LEASY-Katalog nicht eingerichtet (CATD, OPTR)
	LS35	Makrofehler: Fehler beim Makro ENAMP
	LS61	Makrofehler: Fehler beim Makro ENAEI
	LS63	Makrofehler: Fehler beim Makro SOLSIG
	LS64	Makrofehler: Fehler beim Makro POSSIG
	LS69	Makrofehler: Fehler beim Makro DISMP
	LS70	Makrofehler: Fehler beim Makro DISEI

Tabelle 26: Returncodes zur Fehleranzeige (Teil 10 von 10)

14 Diagnosedatei

Es existiert eine zentrale Diagnosedatei (im Folgenden „*leadiag*“ genannt), die für einen LEASY-Katalog Gültigkeit hat. Alle Anwendungen, die mit diesem Katalog arbeiten, schreiben bestimmte Meldungen in diese Datei. Die Diagnosedatei dient dem Kundendienst zur Untersuchung von Fehlerursachen.

Charakteristika der Diagnosedatei

- Um die Anforderungen bezüglich parallelem Schreib- und Lesezugriff zu erfüllen, handelt es sich um eine ISAM-Shared-Update-Datei mit folgenden Eigenschaften:
  `FCBDAT FCB FCBTYPE=ISAM, EXIT=EXLST, KEYARG=SATZKEY, KEYPOS=5, KEYLEN=8, SHARUPD=YES, OPEN-MODUS=INOUT`
- Name der Diagnosedatei: *katalog-name.LEADIAG*.
  Die Datei wird von der LEASY-Maintask eingerichtet, falls sie noch nicht vorhanden ist. Das ist möglich, weil LEASY-Maintask immer in derjenigen Kennung läuft, in der der Katalog liegt. Dabei wird die beim Katalog eventuell angegebene *CATID* berücksichtigt, d.h. die Diagnosedatei *leadiag* liegt auf demselben Pubset wie der LEASY-Katalog.

Schreiben in die Diagnosedatei

Wenn eine Anforderung zum Schreiben in die Diagnosedatei *leadiag* vorliegt (siehe Seite 422), wird folgendermaßen vorgegangen:

- Beim Öffnen des Katalogs (*CATD*) findet eine allgemeine Prüfung statt:
  Es wird mit *FSTAT* festgestellt, ob in der Kennung, in der der Katalog liegt, die Diagnosedatei mit dem festen Namen *katalog-name.LEADIAG* schon bzw. noch existiert. Ist dies nicht der Fall, wird die Datei eingerichtet.
- Die Meldung wird in einem standardisierten Format mit dem nächsthöheren Schlüssel in die Diagnosedatei geschrieben. Dieses Format enthält u.a. folgende Informationen: Datum, Uhrzeit, Benutzername, DB-Name, Modulname/-ID, wo der Fehler aufgetreten ist. Für Fehler bei Betriebssystem- und DMS-Aufrufen gibt es spezielle Formate zur Ausgabe der entsprechenden Fehlercodes. Weitere spezielle Formate gibt es für Dienstprogramme sowie Start und Ende von Anwenderprogrammen.

Die zentrale Diagnosedatei bleibt immer bestehen; der Administrator kann sie exklusiv (d.h. wenn das LEASY-Laufzeitsystem oder LEASY-Maintask diesen Katalog nicht geöffnet hat) löschen oder durch Löschung älterer Sätze verkleinern.

Wann wird ein Satz in die Diagnosedatei geschrieben?

- Allgemeine wichtige Hinweise, z.B. das Anlegen des Common Memory Pools durch LEASY-Maintask und die Beendigung von LEASY-Maintask werden in die Diagnosedatei ausgegeben.
- Bei Systemfehlern, das ist in der Regel die Rückgabe eines Returncodes *LSxxx*, wird ein Fehlersatz in die Diagnosedatei geschrieben und ein Dump erzeugt.
- Wenn eine Dump-Vereinbarung für andere als *LSxxx*-Fehlercodes besteht, wird ein Satz in die Diagnosedatei ausgegeben. In diesem Fall wird als Modul-ID „*DPRC 00*“ (*DPRC* ist die Anweisung zum Einschalten eines Dump) ausgegeben, weil der verursachende Modul nicht bekannt ist.
- Beim Auftreten eines für das Logbuch relevanten Ereignisses werden die zugehörigen Informationen in die Diagnosedatei geschrieben. Diese Ereignisse sind:
  - Folgende Eingaben zur Administration mit dem Dienstprogramm LEASY-MASTER: QUIE, HOLD, CONT, RLBT, LOCT, UNLT, LOCF, UNLF, AIMI, AIMC, AIME, AIMW, IOTE, AIMA, REPO, ROMS, ROMR
  - Start des Dienstprogramms LEASY-MAINTASK
    Alle Startparameter mit Ausnahme von **COM*, **PAS* und **REN* werden einschließlich ihrer Operandenwerte protokolliert.
  - Start des Dienstprogramms LEASY-RECONST
    Die Startparameter **CAT*, **DAT*, **FIL*, **MOD* und **SES* werden einschließlich ihrer Operandenwerte protokolliert.
  - Start oder Beendigung von Anwenderprogrammen

Fehler bei leadiag

Treten beim Einrichten bzw. Öffnen der Diagnosedatei oder beim Schreiben in die Diagnosedatei DVS-Fehler auf (z.B. kein ausreichender Platz verfügbar), dann reagiert LEASY auf folgende Weise:

- **LEASY-Maintask:** Die Diagnosedatei muss geöffnet bzw. angelegt werden können und der Hinweis auf die Einrichtung des Common Memory Pools muss ausgegeben werden können, andernfalls erfolgt ein harter Abbruch, weil das System grundsätzlich nicht arbeitsfähig ist. Dabei wird eine Meldung mit DVS-Code nach SYSOUT ausgegeben, damit der Administrator Maßnahmen ergreifen kann.
- **LEASY-Laufzeitsystem:** Die eigentlich für *leadiag* vorgesehene Meldung wird nach SYSOUT geschrieben, zusätzlich wird der DVS-Code des misslungenen Aufrufs (*OPEN* oder *PUT*) ausgegeben, damit der Administrator Maßnahmen ergreifen kann. Zur Hervorhebung dieser Meldung im SYSOUT-Protokoll werden davor und danach zusätzlich die Meldungen `LEA5014` bzw. `LEA5015` ausgegeben.

Mengenbegrenzung der Diagnosedatei

Die Diagnosedatei *leadiag* wird nach Erreichen von 100.000 protokollierten Sätzen (entspricht einer Datenmenge von ca. 8 MB bei einer Satzlänge von 80 Byte) automatisch gesichert.

Dazu wird sie unter neuem Namen katalogisiert, der als Suffix das Tagesdatum und aktuelle Uhrzeit in der Form *yyyy-mm-dd.hhmmss* erhält, z.B. `LEATEST.LEADIAG.2006-07-24.110523`.

Die aktuell zu beschreibende Diagnosedatei wird mit dem Standardnamen (*katalog-name.LEADIAG*) neu eingerichtet. Mit dieser Vorgehensweise erhält man eine zusammenhängende Dateifolge. Der LEASY-Administrator muss selbst dafür sorgen die Dateianzahl zu begrenzen, d.h. alte Dateien (ggf. nach Sicherung auf Band) löschen.

15 Technische Daten

Lieferumfang

- Bibliothek *SYSLNK.LEASY.062*:
  - LEACON: Verbindungsmodul, das das LEASY-Laufzeitmodul *LEACONX* nachlädt.
  - LEASY: (nicht reentrant, Größe ca. 3 KB) Verbindungsmodul enthält den Entry *LEASY*. Beim 1. Ansprung wird das Verbindungsmodul *LEACON* nachgeladen.
  - LEASYI: (nicht reentrant, Größe ca. 3KB) Verbindungsmodul (Parameterübergabe gemäß ILCS-Konventionen) enthält den Entry *LEASY*. Beim 1. Ansprung wird das Verbindungsmodul *LEACON* nachgeladen.
  - Nachlademodule:

LEACNV	LEASY-CONVERT
LEACONX	LEASY-Laufzeitsystem (Größe ca.114 KB)
LEACTX	LEASY-CATALOG
LEAICNX	I/O-TASK-Modul
LEAILCS	ILCS-Verbindungsmodul
LEAITX	LEASY-IOTASK
LEALDX	LEASY-LOADSI
LEAMSX	LEASY-MASTER
LEAMTX	LEASY-MAINTASK
LEARCX	LEASY-RECONST
LEASVX	LEASY-SAVE

- Bibliothek *SYSLNK.LEASY.062.DCAM*:
  - LEADCAM: (nicht reentrant, Größe ca. 5 KB) mit Einsprungadresse LEASY als Ersatz für das Modul *LEASY* bei DCAM-Anwendungen.
  - LEADCAMI: (nicht reentrant, Größe ca. 5KB; Parameterübergabe gemäß ILCS-Konventionen) mit Einsprungadresse LEASY als Ersatz für das Modul *LEASY* bei DCAM-Anwendungen.

- Bibliothek *SYSLNK.LEASY.062.IOH*:
  - LEASY: (nicht reentrant, Größe ca. 3 KB) Das Modul *LEASY* muss verwendet werden, wenn mit der I/O-TASK gearbeitet werden soll.
  - LEASYI: (nicht reentrant, Größe ca. 3 KB) Das Modul *LEASYI* (Parameterübergabe gemäß ILCS-Konventionen) muss verwendet werden, wenn mit der I/O-TASK gearbeitet werden soll.
  - LEACON: Verbindungsmodul, das das I/O-TASK-Modul *LEAICNX* nachlädt.
- Bibliothek *SYSLIB.LEASY.062*:
  - COPY-Elemente für die COBOL-Schnittstelle

LEASYKON	Konstanten für die Schnittstelle
LEASYPAR	Parameter für die Schnittstelle
LEASYRE	LEASY-RE-Bereich für die DATA DIVISION (WORKING-STORAGE-SECTION)
LEASYREL	LEASY-RE-Bereich für die DATA DIVISION (LINKAGE-SECTION)

  - Makros für openUTM-Anwendungen

KDCDB	Makro für die Generierung der KDCROOT
KDCDBL	Makro für die Generierung der KDCROOT bei Multi-DB-Betrieb

  - Makros für die Assembler-Schnittstelle
- Bibliothek *SYSPRG.LEASY.062*:
  - Programmdateien *LEA.xxx* für die alte Aufrufschnittstelle *START-PROGRAM phase* (nähere Einzelheiten siehe Freigabemitteilung)
  - Startmodule (LLMs) der LEASY-Dienstprogramme für die SDF-Aufrufschnittstelle *START-LEASY-xxx*
- Meldungsdatei *SYSMES.LEASY.062*
- Subsystemdeklarationen *SYSSSC.LEASY.062*
- System-Syntaxdatei *SYSSDF.LEASY.062*
- Informationen für IMON *SYSSII.LEASY.062*
- Extraktionsprozedur *SINPRC.LEASY.062* (nähere Einzelheiten siehe Freigabemitteilung)
- Freigabemittteilung *SYSFGM.LEASY.062.D* (deutsch) und *SYSFGM.LEASY.062.E* (englisch)

Größe der dynamischen Speicher

Während des Laufes fordert *LEACON* noch zusätzlich taskspezifischen Speicher mit REQM (Speicherklasse 6) an, dessen Größe im Wesentlichen von der Zahl der verwendeten Dateien abhängt. Seine Größe lässt sich durch Addieren des Speicherbedarfs für folgende Dateien ermitteln (die aktuellen Größenangaben entnehmen Sie bitte der Freigabemitteilung):

1 Stack-Bereich für interne Prozedur-Daten

1 BIM-Puffer

1 FCB-Bereich je eröffnete BIM-Datei und interne Verwaltung

1 FCB-Bereich für die AIM-Datei und interne Verwaltung

1 FCB-Bereich für die Katalogdatei

1 Bereich je LEASY-Datei (SAM, ISAM, DAM oder PAM) für
FCB und interne Verwaltung

Bereich für TRACE-Information

Bereich für globale Verwaltungsdaten

1 Arbeitsbereich für DAM-Dateibearbeitung

Zusätzlich fordert das DVS (in Speicherklasse 5) für jede eröffnete SAM-, DAM- oder ISAM-Datei 2 IOAREA's in der Größe *BLKSIZE* der Datei an. Für PAM-Dateien werden keine IOAREA's angelegt.

Werden für eine ISAM-, DAM- oder PAM-Datei Sekundärschlüssel definiert, so wird zu jeder Primärdatei eine zusätzliche ISAM-Sekundärschlüsseldatei angelegt, für die folgender Speicherbedarf entsteht (in Byte):

1 Bereich für FCB und LEASY-internen SI-Puffer (Speicherklasse 6)

2 Bereiche für die IOAREA's (Speicherklasse 5)

Größe und Aufbau des Common Memory

Die Größe des erforderlichen Common Memory CMMAIN lässt sich nach folgender Formel berechnen:

$$\text{SIZE [Byte]} = 292 + 64 + 16 + 88*d + 8*tk*dam + (t+7)/8 + 33*m + 22*si +$$

$$5*pl + \sum_{i=1}^{k} (1 + l(SADEFk)) + 80*tk + 22*app + 144*t + 32*f +$$

$$(5 + 2*\,maxkey)*10*t + buc*t\ [+\ 4096*ai]\ [+\ 4096*t]$$

d	Anzahl der Dateien im LEASY-Katalog (außer Modelldateiexemplare)		
m	Anzahl der Modelldateiexemplare im LEASY-Katalog		
dam	Anzahl der DAM-Dateien im LEASY-Katalog		
si	Anzahl der Sekundärindizes aller Dateien		
pl	Anzahl der (pos, len, dist)-Definitionen aller Sekundärschlüssel aller Dateien		
k	Anzahl aller Satzartenabhängigkeiten bei Sekundärschlüssel-Definitionen		
l(SADEF)	Länge einer Satzartendefinition für satzartenabhängige Sekundärschlüssel (l(SADEF) = 0 bei RTP=NONE in der FIL-Anweisung, LEASY-CATALOG)		
tk	Anzahl der parallel zugelassenen Tasks (Teilnehmer-, Stapel- und Teilhabertasks)	*TSK	MAINTASK Anweisungen für das Festlegen der entsprechenden Werte
app	Anzahl der parallel zugelassenen Teilhaber-Anwendungen	*APP	
t	Anzahl der parallel zugelassenen Transaktionen	*TRA	
f	Maximalzahl der Dateien, die gleichzeitig eröffnet sein können	*FIL	
maxkey	größte Keylen aus ISAM bzw. PAM	*KEY	
buc	Größe einer Speichereinheit im Bucket Pool	*MUS	
ai	Anzahl der Seiten (4 Kbyte) des AIM Puffers	*AIB	

Im Allgemeinen wird die Voreinstellung für die Größe des Common Memory (die LEASY-MAINTASK-Anweisung **MEM=1* entspricht einem Segment, d.h. 64 Kbyte) ausreichen. Bei stärkerer Belastung ist der Wert der **MEM*-Anweisung entsprechend größer zu wählen.

Man beachte, dass der AIM-Puffer im Common Memory CMMAIN enthalten ist und bei entsprechender Größe auch den Wert der **MEM*-Anweisung stark beeinflussen kann.

Der Common Memory CMMAIN besteht aus festen Teilen am Anfang und Ende des Speicherbereichs und einem dazwischenliegenden Teil, der dynamisch verwaltet wird.

Die festen Teile bestehen aus:

- Verwaltungsblock.
- Koordinierungsblock.
- Abbild des LEASY-Katalogs.
- Tasktabelle.
- Teilhaber-Anwendungstabelle.
- Transaktionstabelle.
- After-Image-Puffer, falls *LOG=A/Y.
- Before-Image-Bereich, falls *LOG=B/Y.

Der dynamisch verwaltete Speicherteil enthält

- einen zusammenhängenden Bereich für die Sperrtabelle, in der für jede Transaktion für 100 Sperrelemente Platz vorgesehen ist.
- einen in gleich große Speichereinheiten (Buckets) geteilten Speicherbereich (Bucket Pool).

Die Größe der Speichereinheiten (Buckets) kann der Benutzer durch die Anweisung **MUS* (Memory Unit Size) einstellen. Für jede parallel ablaufende Transaktion muss mindestens 1 Bucket zur Verfügung stehen, in dem die transaktionsorientierte Dateikennungsverwaltung abgelegt wird.
Reicht ein Bucket dafür nicht aus, so werden der Transaktion weitere Buckets aus dem Bucket-Pool zugeteilt. Bei Transaktionsende werden alle diese Buckets wieder freigegeben und einer Freiplatzverwaltung für Buckets zugeführt.

Läuft der zusammenhängende Speicher für Sperrelemente über, so werden ebenfalls dynamisch Speichereinheiten aus dem Bucket-Pool der Sperrtabelle zugeordnet. Solche für Sperrelemente angeforderten Speichereinheiten bleiben bis ans Lebensende des CMMAIN der Sperrtabelle zugeordnet. Bei Freiwerden der Sperrelemente werden diese der Freiplatzverwaltung für Sperrelemente eingegliedert.

Fachwörter

AIM

After-Image-Sicherungskonzept:

- Speicherung des logischen Dateninhalts nach der Änderung.
- Schutz bei Hardware-Fehlern (Zerstörung von Datenfeldern).
- Rekonstruktion nach Systemzusammenbruch.
- Realisierung über AIM-Datei und Sicherungskopien der Original-Dateien.

BIM

Before-Image-Sicherungskonzept:

- Speicherung des Dateninhalts vor der Änderung.
- Schutz bei Software-Fehlern (Programmabbruch).
- Keine Unterbrechung oder Beeinflussung anderer Transaktionen.
- Warmstartfähigkeit (Wiederanlauf nach Softwarefehlern).
- Realisierung über BIM-Dateien.

Common Memory

Gemeinsamer Speicherplatz aller an einen Katalog angeschlossenen Prozesse.

DAM

Direkte Zugriffsmethode:

Abgeleitet aus der relativen Dateiorganisation von COBOL und nach der KLDS-Norm ausgelegt mit zusätzlicher Unterstützung von Sekundärindizes (siehe Seite 40ff).

DCAM

Data Communication Access Method:

Teilhaberbetrieb im BS2000.

Deadlock

Allgemeiner Wartezustand auf Betriebsmittel in einem System, der auf Grund der Konstellation nie von selbst beendet werden würde.

ISAM

Indexsequentielle Zugriffsmethode im Sinne des Datenverwaltungssystems (siehe Handbuch „Einführung in das DVS“), mit zusätzlicher Unterstützung von Sekundärindizes (siehe Seite 33ff).

openUTM

Universeller Transaktions-Monitor. Er realisiert den Teilhaberbetrieb im BS2000.

PAM

Primäre, blockorientierte Zugriffsmethode des BS2000 (siehe Handbuch „Einführung in das DVS“) mit zusätzlicher Unterstützung von Sekundärindizes (siehe Seite 38ff).

Primärindex

Der Primärindex ist ein eindeutiger Ordnungsbegriff für Dateien, die direkten Zugriff erlauben.

- Für ISAM-Dateien gilt als primärer Ordnungsbegriff der ISAM-Schlüssel der Datei.
- Für PAM-Dateien wird als primärer Ordnungsbegriff die Blocknummer (Half Page Number) des PAM-Blocks verwendet, mit der der logische Datenblock des Benutzers beginnt (der Datenblock kann sich über mehrere PAM-Blökke zu je 2048 Bytes erstrecken).
- Für DAM-Dateien ist dies ein 4 Byte langer Wert größer oder gleich Null, der nicht Bestandteil des Datensatzes der Datei ist.

SAM

Sequentielle Dateien im Sinne des Datenverwaltungssystems (siehe Handbuch „Einführung in das DVS“).

Sekundärindex

Ordnungsbegriff für Sätze einer ISAM-, DAM- oder PAM-Datei, der ebenso wie der primäre Ordnungsbegriff Direktzugriff zu den Sätzen der Datei ermöglicht.

SI

Sekundärindex.

SI-Datei

Sekundärindexdatei. Sie enthält die Verweise der Sekundärschlüssel auf die primären Schlüsselbegriffe der Benutzerdatei.

Sperre

Kennzeichen des primären Schlüsselbegriffs eines Datensatzes zur logischen Trennung von Schreibzugriffen verschiedener Transaktionen auf dieselben Datensätze.

Transaktion

Folge von Dateizugriffsoperationen, die als eine verarbeitungstechnische Einheit aufgefaßt wird.

UPAM

Siehe PAM.

UTM

Siehe openUTM.

Literatur

Die Handbücher sind online unter *http://manuals.fujitsu-siemens.com* zu finden oder in gedruckter Form gegen gesondertes Entgelt unter *http://FSC-manualshop.com* zu bestellen.

LEASY (BS2000/OSD)
Dienstprogramme
Benutzerhandbuch

LEASY (BS2000)
Taschenbuch

COBOL85 (BS2000/OSD)
COBOL-Compiler
Benutzerhandbuch

COBOL2000 (BS2000/OSD))
COBOL-Compiler
Benutzerhandbuch

Assembler (BS2000)
Beschreibung

SDF (BS2000/OSD)
Einführung in die Dialogschnittstelle SDF
Benutzerhandbuch

BS2000/OSD-BC
Kommandos Band 1 - 5
Benutzerhandbuch

BS2000/OSD
Makroaufrufe an den Ablaufteil
Benutzerhandbuch

BS2000/OSD-BC
Dienstprogramme
Benutzerhandbuch

SORT (BS2000/OSD)
Benutzerhandbuch

ARCHIVE (BS2000/OSD)
Benutzerhandbuch

BS2000/OSD-BC
Systemmeldungen Band 1 bis Band 3
Benutzerhandbuch

openUTM (BS2000/OSD)
Anwendungen generieren und betreiben
Benutzerhandbuch

openUTM (BS2000/OSD, UNIX,WIndows NT)
Anwendungen administrieren
Benutzerhandbuch

openUTM (BS2000/OSD, UNIX, Windows NT)
Anwendungen programmieren mit KDCS für COBOL, C und C++
Basishandbuch

openUTM Meldungen, Test und Diagnose

DCAM (BS2000/OSD, TRANSDATA)
Programmschnittstellen
Beschreibung

DCAM (BS2000/OSD, TRANSDATA)
COBOL-Aufrufe
Benutzerhandbuch

JV (BS2000/OSD)
Jobvariablen
Benutzerhandbuch

BS2000/OSD-BC
Sicherheitshandbuch für die Systembetreuung

DRIVE/WINDOWS (BS2000)
Programmiersystem
Benutzerhandbuch

BS2000/OSD-BC
Einführung in das DVS
Benutzerhandbuch

BS2000/OSD-BC
DVS-Makros
Benutzerhandbuch

Stichwörter

M

N

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.