

**Adabas**

**DBA タスク**

バージョン 8.1.3

June 2008

---

Adabas

This document applies to Adabas Version 8.1.3 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © Software AG 1971-2008. All rights reserved.

The name Software AG™, webMethods™, Adabas™, Natural™, ApplinX™, EntireX™ and/or all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. Other company and product names mentioned herein may be trademarks of their respective owners.

# 目次

1 DBA タスク .....	1
2 DBA の役割と責任 .....	3
3 集中的な管理と調整作業 .....	5
4 IS (情報システム) 部門の DBA .....	7
組織内での DBA の位置 .....	8
DBA に必要な能力 .....	8
マネージメントサポート .....	9
起こり得る誤り .....	9
5 データベースの制御と管理体系の確立 .....	11
データベース手順と標準の設定 .....	12
手順や標準の保守 .....	14
データベース設計の援助 .....	14
ユーザー教育 .....	14
データベースシステムに適したアプリケーションの選択 .....	196
DBA の役割のまとめ .....	15
6 データ定義と制御 .....	17
アプローチ方法：データの集中管理 .....	26
データの信頼性の決定 .....	18
アプリケーションの選択：システム開発へのアドバイス .....	18
データ収集と整合性確認に対するアドバイス .....	19
データベースの内容の定義 .....	19
7 データベースドキュメンテーション .....	21
標準 .....	22
データベースの記述 .....	23
データディクショナリ、機能と使用法 .....	24
Adabas データディクショナリ (Predict) .....	24
データベースを使用するアプリケーション .....	25
データ源の記述 .....	26
データのアクセスおよび操作手順 .....	27
パスワードおよびユーザー ID .....	27
バックアップ手順 .....	28
再スタートおよびリカバリ手順 .....	29
DBMS パフォーマンスおよび計測 .....	29
8 教育および訓練 .....	31
概要 .....	62
データベースの概念 .....	32
データベース設計 .....	33
プログラミング .....	33
オペレーティング手順およびテクニック .....	34
データ入力 .....	34
データベース問い合わせおよびレポート生成 .....	226
9 DBA とユーザー .....	37
ユーザーとの連絡 .....	38

アクセス要求 .....	39
アプリケーションインターフェイス .....	40
標準および制御の順守 .....	40
10 DBA とアプリケーションの選択/開発 .....	41
機器構成とアプリケーション開発のプランニング .....	42
データベース編成 .....	42
ユーザー要求の理解（現在および将来） .....	43
データベース活動の調整 .....	43
アクセス要求の分析 .....	43
データ使用のための準備 .....	44
パフォーマンス対柔軟性 .....	44
アプリケーション/プログラム/データベースの設計へのアドバイス .....	45
必要物理ストレージの決定 .....	45
テストデータベースとテスト方法 .....	46
11 DBA とコンピュータオペレーション .....	49
データベース管理者の影響 .....	247
コンピュータ時間のスケジューリング .....	50
オペレーティング手順 .....	50
再起動およびリカバリ手順 .....	51
データベースユーティリティ .....	51
Software AG との関係 .....	52
12 データベース設計 .....	55
13 システム設計におけるパフォーマンス制御 .....	57
システム設計におけるパフォーマンス制御方法 .....	58
14 ファイルおよびレコード設計 .....	61
マルチプルバリュースフィールドピリオディックグループ .....	62
1つの Adabas ファイルへの異なるレコードタイプの収容 .....	65
1つの論理ファイルへの物理ファイルのリンク .....	66
データの重複 .....	67
Adabas レコード設計 .....	67
15 データアクセス手法 .....	69
ディスクリプタの効率的な使用 .....	70
照合ディスクリプタ .....	70
スーパーディスクリプタ .....	71
サブディスクリプタ .....	71
フォネティックディスクリプタ .....	72
ハイパーディスクリプタ .....	72
ファイルカップリング .....	72
ユーザー割り当て ISN .....	74
ISN をディスクリプタとして使用 .....	74
ADAM の使用方法 .....	74
16 ディスクスペースの使用法 .....	77
データ圧縮 .....	78
フォワードインデックス圧縮 .....	80
パディングファクタ .....	81

17 Adabas Security .....	83
セキュリティ計画 .....	84
パスワードセキュリティ .....	84
セキュリティバイバリュー .....	86
暗号化 .....	86
Adabas SAF Security .....	86
Natural Security と Adabas Online System Security .....	87
18 リカバリ／再スタート設計 .....	89
Adabas リカバリ機能 .....	90
リカバリの計画とインストール .....	91
要求と機能のマッチング .....	91
トランザクションリカバリ .....	92
エンドトランザクション (ET) コマンド .....	92
クローズ (CL) コマンド .....	92
ET データの読み込み .....	130
システムおよびトランザクション障害 .....	93
Adabas トランザクションリカバリの制限 .....	93
Adabas チェックポイントコマンド .....	93
排他ファイル制御 .....	94
ユーザー再スタートデータ .....	95
19 Adabas Recovery Aid .....	97
リカバリログ (RLOG) .....	98
Adabas Recovery Aid の起動 .....	99
20 マルチクライアントサポート .....	101
オーナーの概念 .....	102
スーパーユーザー .....	103
プログラムの互換性 .....	104
ソフトカップリングのサポート .....	104
データとインデックスの構造 .....	104
パフォーマンスの考慮事項 .....	106
ユーザープロファイルテーブル .....	107
発生しうる Adabas レスポンスコード .....	107
マルチクライアントファイルのユーティリティサポート .....	107
21 拡張ファイル .....	111
概要 .....	112
拡張ファイルの定義 .....	113
コンポーネントファイルの挿入 .....	115
コンポーネントファイルの削除 .....	115
拡張ファイルの削除 .....	115
拡張ファイルの調査 .....	155
拡張ファイルと Adabas ニュークリアス .....	116
拡張ファイルと Adabas ユーティリティ .....	117
22 データベースのメンテナンス .....	121
23 Adabas データベースの定義 .....	123
ステップ 1：データベースに必要なサイズを見積ります。 .....	124


ステップ 2: データベースの割り当て .....	141
ステップ 3: データベースのフォーマット .....	143
ステップ 4: データベースパラメータの定義 .....	144
24 データベースのスペース管理 .....	145
Adabas 物理エクステント .....	146
Adabas 相対ブロック番号 (RABN) .....	147
Adabas 論理エクステント .....	149
Adabas スペース割り当ておよび割り当て解除 .....	150
データベースステータスレポートを使用したスペース使用状況の監視 .....	159
スペース使用に関する問題点とその対応策 .....	159
25 データベースのモニタリングとチューニング .....	163
リソース使用のモニタリング .....	164
データベース使用のレポート出力 .....	164
データベース制御のモニタリング .....	164
パフォーマンス管理、統計、チューニング .....	165
Adabas セッション統計 .....	166
コマンドロギング .....	171
26 エラー処理およびメッセージバッファリング .....	173
概要 .....	174
オペレーションの範囲 .....	174
リカバリまたはプラグイン (PIN) ルーチン .....	175
PIN ルーチンユーザー出口 .....	187
27 ユニバーサルエンコーディングサポート (UES) .....	191
概要 .....	192
ワイド文字のエンコード .....	192
ワイド文字データサポート .....	194
28 複数プラットフォームのサポート .....	197
概要 .....	198
エンコード .....	199
ADACOX 変換出口 .....	200
バリュースタック内における上限値の変換 .....	200
データ変換の制約 .....	201
プラットフォームの考慮事項 .....	201
29 ラージオブジェクト (LB) フィールドの基本 .....	203
30 Adabas Online System デモバージョン .....	207
31 概要 .....	209
AOS デモバージョンでできること .....	210
32 メインメニュー機能 .....	213
AOS デモバージョンでのデータベースの指定 .....	215
プログラム機能 (PF) キーの使用 .....	215
メニューオプションの選択 .....	215
ヘルプ .....	216
AOS デモバージョンメッセージ .....	216
33 セッションの監視 .....	217
ADARUN パラメータの表示 .....	219

ホールドキューの表示 .....	220
システムステータスおよびスレッド使用の表示 .....	221
メンテナンスレベルの表示 .....	223
34 チェックポイントのリスト .....	225
35 ファイルメンテナンス .....	229
36 データベースメンテナンス .....	233
37 システムオペレータコマンド機能 .....	235
拡張エラーリカバリ .....	237
ロックファイルの表示 .....	239
ユーザーの停止 .....	240
セッションの正常終了 (ADAEND) .....	241
38 データベースレポート .....	243
ファイルの表示 .....	244
全体的なデータベースレイアウトの表示 .....	248
39 AFPLOOK .....	251
AFPLOOK の有効化 .....	252
運用上のデフォルト .....	252
デフォルトの調整 .....	252
AFPLOOK パラメータ .....	253
AFPLOOK レポート .....	255
40 AVILOOK .....	259
AVILOOK の有効化 .....	260
SYSAVI – AVILOOK の選択 .....	260
SYSAVI – AVILOOK の使用 .....	261
目次 .....	265











# 1 DBA タスク

 **Note:** Adabas のマニュアルでは、接頭辞 DD を含まない VSE データセット名に対応するために、DD で始まるデータセット名は、DD とデータセット名の残りの部分をスラッシュで区切って示します。スラッシュはデータセット名の一部ではありません。

この章では、データベース環境を正常に運用するために必要な管理業務や制御業務の全範囲について記述しています。この情報は次の部分で構成されています。

 <b>DBA の役割と責任</b>	データベース管理者 (DBA) の役割と責任について説明します。
 <b>データベース設計</b>	データベースの設計について説明します。ここでは Adabas ファイルの構造、マルチプルバリューフィールドおよびピリオディックグループ、レコード設計、キーの使用法 (ディスクリプタ)、ディスクスペースの利用法 (圧縮、空値省略、パディングファクタ)、セキュリティプランニング、再スタートとリカバリの計画、マルチクライアントファイル、拡張ファイルに関する情報が含まれます。
 <b>データベースのメンテナンス</b>	チューニングとエラー処理を含む、データベースの定義と保守に関連する課題について説明します。
 <b>Adabas Online System デモバージョン</b>	Adabas に含まれる Adabas Online System (AOS) デモバージョンと、その他の選ばれた Adabas 製品のオンラインサービスの利用について説明します。
 <b>AFPLOOK</b>	Adabas Fastpath コマンド分析サンプラ (AFPLOOK) について説明します。これは、Adabas Fastpath の条件を満たすコマンド構造について報告することで、Adabas Fastpath で最高の結果が期待される箇所の特定に使用できます。
 <b>AVILOOK</b>	Adabas Vista コマンド分析サンプラ (AVILOOK) について説明します。これは、ファイルのアクセスに使用されるコマンド構造について報告することで、Adabas Vista のパーティションオプションの効果があるファイルの特定に使用できます。



## 2 DBA の役割と責任

---

データベース環境においては、データベースの設計、導入および使用を一括制御することが、その成功の鍵です。このような一括制御や調整は、データベース管理者（DBA）の役割です。

この章は、DBA の役割、DBA の権限や責任、必要な技量、DBA が策定し維持する必要がある処理手順、標準化、連絡窓口について述べています。

このマニュアルでは、DBA を1人の人物であるかのように扱っていますが、大規模な組織では、特定の技術や分野（データベース設計、チューニング、問題解決など）ごとに、DBA の役割を複数の担当者と分担してもかまいません。

この情報は次の項目で構成されています。

- 集中的な管理と調整作業
- IS（情報システム）部門の DBA
- データベースの制御と管理体系の確立
- データ定義と制御
- データベースドキュメンテーション
- 教育および訓練
- DBA とユーザー
- DBA とアプリケーションの選択／開発
- DBA とコンピュータオペレーション

---

## 3 集中的な管理と調整作業

---

Adabas のようなデータベース環境では、組織内の多数の部門から、多数のアプリケーション（ユーザー）により同じデータにアクセスされます。データの所有権と責任は、ニーズが多様で相反することもある複数の部門で分担します。ニーズが相反する場合、この問題を解決するのも DBA の仕事の 1 つです。

データのセキュリティと保全性は、個人または部門がそれぞれ独自に実装するのではなく、Adabas のようなシステムで実装します。このようなシステムが実現する DBA の管理機能やカスタマイズされたセキュリティプロファイルによって通常はセキュリティと保全性が向上します。

過去においてアプリケーション開発チームは、たいてい自身の都合からアプリケーションファイルの設計や維持について大きく責任を持っていました。そのデータを使用したい他のアプリケーションは、もとのファイルの設計を受け入れるか、使用目的に合わせてファイルを変換しなければなりません。このため、データ保全性、リカバリ手順およびプライバシーの保護は一貫していませんでした。加えて、システム全体の効率に対してはほとんど注意が払われていませんでした。あるシステムで行われた変更が不運にも他のシステムのパフォーマンスに影響を及ぼすことがありました。

統合化と共有化が可能なデータベースが存在しても、一括制御を行わないとすぐに混乱に陥ってしまいます。あるプロジェクトでファイル構造を変更したために、不運にも他のプロジェクトに影響を与えてしまいます。あるプロジェクトの効率を上げようとする他を犠牲にすることになります。異なるセキュリティやリカバリ手順を使用していると、良くて混乱や不安を招き、悪くすると不安定で確実性の乏しいデータベースになりかねません。

明らかに、適切なデータベース管理とは、共通の標準と全インストレーション的な見地からのハードウェアとソフトウェア要求が一括制御には必要であるということを意味します。この一括制御が DBA の責任です。以上のような理由から、この機能をデータベース開発サイクルの最初に設定することが重要です。

---

## 4 IS（情報システム）部門の DBA

---

■ 組織内での DBA の位置 .....	8
■ DBA に必要な能力 .....	8
■ マネージメントサポート .....	9
■ 起こり得る誤り .....	9

データベース管理者（DBA）が効果的に作業できるかどうかは、DBA がその作業に使うスキルと知識、また情報システム（IS）全体の運用における DBA の役割に依存します。このセクションでは、DBA の役割の定義方法、IS 部門と DBA の関係について説明し、その関係から最大の利益を得るための提案を行います。

このchapterでは、次のトピックについて説明します。

### 組織内での DBA の位置

---

データベースの使用を必要な程度にコントロールできるよう、また、ユーザー部門内の適切なレベルとコミュニケーションできるような組織内でも十分に高い地位に DBA を配置すべきです。しかし、データベースの使用を監視し、新規アプリケーションの選択やアドバイスをを行い、データベースの必要なレベルのデータ完全性を維持するという日々の処理からかけ離れてしまうほど高い地位にすべきではありません。

DBA の適切な位置付けと報告体系は、組織の性質と規模に依存します。

普通、DBA の地位はシステムマネージャ、プログラミングマネージャおよびオペレーションマネージャと同様、職務上のマネージャとして置きます。データベースの一連の操作について全面的に直接の責任を負う必要があります。また、DBA は DBMS 準拠の標準化について理解させ、遵守させる責任も負うため、プログラミングと IS オペレーションの標準化全般を管理する権限の一部を DBA にも与えておくと、その責任を果たしやすくなります。

### DBA に必要な能力

---

DBA には、部門の調整能力、高い技術力、折衝能力、監視能力が不可欠です。DBA は、公正な人物であり、データベースの問題を両サイド（IS 部門とユーザー部門）から見ることができ、どちら側にも偏らない人物であるべきです。組織全体の利益を考え DBA は思慮ある判断を下さねばなりません。

DBA には次の能力も必要です。

- データベース使用に関連した各種標準事項や手順を設定し実施させるための管理能力
- 使用しているオペレーティングシステムソフトウェアや DBMS について、深い知識があり、ハードウェアパフォーマンスに影響する因子がわかるだけの技術的能力
- 既存アプリケーションや将来のアプリケーションについての深い知識
- アプリケーションの要求を満たす効果的なデータベース設計を行えるだけの設計能力

中規模から大規模のインストレーションでは、DBA は 1 人ではなく、小人数のチームから成っていることが多く、この場合、チームのそれぞれの人物が、それぞれ専門の知識を持ち、データベースリソースの管理作業を分担します。



小規模のインストレーションでは、DBA チームを構成するのは困難ですが、必要なすべての能力を持った個人をみつけるのは不可能でしょう。この場合は、システムプログラマ、シニアオペレータ、シニアアナリスト等の他のスペシャリストから専従ではなく一時的に支援してもらいDBA の役割を果たしてもらおうようにしてください。

## マネージメントサポート

DBA の職務がまっとうしやすくなるように、IS 部門とユーザー管理部門には DBA の役割の重要性について認識してもらい、支援を受けられるようにする必要があります。DBA は、データベース操作やデータベースが組織に提供するサービスについて深い知識を持っています。データベースの使用や設計を含むすべての事項についての権限を有する担当者として、DBA を扱う必要があります。

原則として、データベース環境の整合性が維持されるように、データベースに影響するすべての決定にはDBA を参加させるようにしてください。また、経営陣が考えるよりもコスト効果の高いソリューションを DBA が提案できる場合もあります。

## 起こり得る誤り

DBA の役割を設定するときにより得る落とし穴を次に示します。

- DBA の組織内の位置が低過ぎる（権限が不十分）。DBA がその責任を十分に果たせるようにするには、DBA の責務に見合った権限が与えられていなければなりません。DBA は IS 部門の組織からかけはなれた位置に置いてはならないし、DBMS 環境で業務するときには必須の役割と考えるべきです。DBA は、各部門のマネージャ達の協力、支援および敬意が必要であり、DBA 業務を行うのに十分な権限が与えられていないと、マネージャ達の協力も得られません。
- DBA の組織内の位置が高過ぎる（権限が過剰）。DBMS 環境の運営がスムーズに行くようランク付けすべきであり、書類の山や、不要な制約処理や押しつけ管理のために現場を停滞させるべきではありません。過少権限と過剰権限との境界線は狭いのですが、組織ごとにこの境界線を認識し、はっきりと仕切らねばなりません。
- DBA の役割や責任範囲が未定義。DBA は、必要な役割を DBMS に適用できるように、その役割を果たすための権限を持っている必要があります。IS 部門とユーザー部門の両方からマネージャが参画して、組織に何が必要かを十分考慮した後、これらの役割を決める必要があります。役割が決まったら、DBA は、その職務を果たすために必要な手順を策定します。
- DBA のマネージメントとしての経験が不十分。DBA の役割は、経験の浅いマネージャが修練を積むことではありません。管理上の専門知識がかなり必要であり、特に人間関係が重要です。



## 5 データベースの制御と管理体系の確立

---

■ データベース手順と標準の設定 .....	12
■ 手順や標準の保守 .....	14
■ データベース設計の援助 .....	14
■ ユーザー教育 .....	14
■ データベースシステムに適したアプリケーションの選択 .....	196
■ DBA の役割のまとめ .....	15

データベース環境の制御や管理を実施できるようにシステムを設定するときに、DBA の一般的な役割を次に示します。

- データベース手順と標準の設定
- データベース設計の援助
- ユーザー教育
- データベースシステムに適したアプリケーション選択
- データベースドキュメントの維持
- データベースの一般的管理

このchapterでは、次のトピックについて説明します。

### データベース手順と標準の設定

---

通常 of データ処理 (DP) 環境と同様に、標準や手順は初期の計画の一部として設定した方が、問題が発生した後に設定するよりも効果的です。このセクションでは、手順や標準を定義するときに考慮すべき一般事項について述べます。

#### データベース手順

データベース環境を効果的に制御するために必要な手順は、DBMS を使用するとき、まず最初に設定する必要があります。

これらの手順について、この章で概略を述べます。多くのインストレーションでは、短時間で DBMS の使用を採用していますが、できるだけ早く運用を開始したいからといって、全プロセスのプランニング (特に、管理手順や制御手順の取り決めと導入) をおろそかにしてはなりません。

これらの手順の導入にあたっては、IS 部門内でも、またユーザーとも、多くの議論 (特に、何を採用するか、費用効果の良いものは何かなど) がかわされるでしょう。ある部署で DBMS テクノロジーを最初に使用するアプリケーションは、DBA プロシージャの開発と並行して開発されることになるでしょう。しかし、組織の IS やユーザーのマネジメントに、これらの手順が必要であることを気付かせ、(議論の後) これらを採用させることが基本です。

## データの機密保護手順

手順の設定データ項目をどのように機密保護するかを決めるのは誰でしょうか。ユーザーというのは、自分の業務にあまりにも密着しており、近視眼的です。また、アナリストがこのような決定にあたって、組織全体のかかわりあいについて把握できないこともあり得ます。最終的な決定はDBAが下します。つまり、DBAとはすべての処理を監視し違反するものをチェックする人です。

## リカバリ手順のプランニング

DBAはインストレーションで、使用する標準のリカバリ手順を設定しなければなりません。各アプリケーションが稼動する前にこれらのリカバリ手順が妥当であるかを確認しなければなりません。つまり、別のアプローチ（例えば、データベースセーブおよびリストアの代わりに、ファイルセーブ／リストアを使用するなど）が採用される可能性があるため、この段階でも決定にDBAが関与する必要があります。

## 標準の設定

このマニュアルの大部分は、データベース環境のガイドラインとして記述されています。ここで説明している内容は、必ずしもすべてのインストレーションに関係していません。ガイドラインが標準となるかどうかは、ユーザー／開発者の組織の規模と多様度に依りて異なります。小さな同一的なユーザーグループは通常、コミュニケーションが十分であるため、広範で厳密に定義された規則の集合を必要としません。

他方、さまざまな分野で構成され、地理的に分散されたユーザーグループは、正しい制御に必要なコミュニケーションが不足する傾向にあります。このようなグループには、データ構造やプログラム開発の不一致を防止するために規則が必要です。明らかな利点がない限り、規則を好む者はいません。しかし、標準は一般的な規則です。標準を定義するときの考慮事項をいくつか次に示します。

- 標準は簡潔、明白、かつ最小限に維持します。しかし、標準に曖昧さがある場合は、簡潔な注意書きを添える必要があります。例えば、Adabas環境での一時データセットの使用に関する標準には、次の注意書きが必要な場合があります。

	Adabas	Adabas
Recovery Aid		
	Adabas Recovery Aid	

- 標準は、少なくとも新たに追加するときには必ず見直し、古いものは削除または更新します。変更／削除／追加した標準の概要をユーザーに周知します。

特定の制御手順または管理手順が特定のサイトで必要であると思われる場合は、標準として定義する必要があります。

### 手順や標準の保守

---

データベースドキュメントの保守はアプリケーション開発プロセスに不可欠な要素として考える必要があります。このため、DBAはDBMSを使用する新しいシステムの開発に参加する必要があります。ドキュメント化プロセスでは、ツールとして主にデータディクショナリを使用します。

DBAは、現在のデータベースアプリケーション、今後の予定、データベースの保全性に対する責任に関する幅広い知識を活かして、データベースにデータを入力する各システムが備えるデータ整合性確認手順の設計に参加する必要があります。このようにすると、データベース内のデータの品質が、確実に許容できるレベルに保たれます。データディクショナリは、整合性確認の要件のドキュメント化にも使用できます。

### データベース設計の援助

---

プロジェクトチームがデータベース設計を行う際、DBAが行う援助は次のとおりです。

- 論理設計が、問題を適確に表わしているか。論理データベース設計は、ある特定のDBMSの制限や機能に制約されないようにします。
- 物理設計が、処理効率を悪くしないか。プロジェクトチームは、他に指示がなければ、特定の問題だけに対処しようとします。
- 将来を考えたとき、設計に柔軟性があるか。DBAと組織は、ある程度の期間をかけて、設計内容を練る必要があります。

独立した役割として、DBAは、結果としてできあがったデータベース設計を客観的に評価できる唯一の担当です。したがって、ときには設計段階にも介入する必要があり、この場合にはDBAは疑問点に対して的確に回答します。

### ユーザー教育

---

システムアナリストやDPリソースマネージャの尽力とは無関係に、ユーザーはDBMSテクノロジーの有益性を過剰評価することがあり得ます。柔軟性、プログラムとデータの独立性、およびデータ可用性のような議論により、ユーザーが過剰に期待し、創造の自由を錯覚してしまうことがあります。

データベース環境で業務を行うのに関連して、その利点と同様問題点をもユーザーに認識させるのは、DBAの責任です。DBAは、このような必要性に適したユーザーを対象に、教育内容をよく吟味し練ってから導入に関する教育を行う必要があります。

## データベースシステムに適したアプリケーションの選択

インストレーションに DBMS が必要な場合、アナリストやプログラマが DBMS を安易に使用してしまう傾向があります。新規アプリケーションに、突然 DBMS テクノロジーが必要になったように見られます。誰かが、この状態をしっかりと統括し、DBMS テクノロジーに本当に適したアプリケーションを選択するよう、制御しなければなりません。この誰かとは、DBA のことです。

DBA は、各アプリケーションに DBMS を使用した場合のプラス面とマイナス面の一覧表を作成する必要があります。DBMS の機能を検討する段階で（すなわち、多くのコストを費す前に）、この一覧表を作成しなければなりません。また、提案アプリケーションの分析に当たって、次の点を考慮する必要があります。

- DBMS が必要か。
- 既存の環境を妨げないか。
- 提案システムは柔軟性があるか。
- コストパフォーマンスは良いか。
- アプリケーション提案の契機となった問題をすべて分析済みか。

DBA は、データベース関係のすべての権限と責任を有する中心として、新規データベースプロジェクトの選択に寄与する担当です。

## DBA の役割のまとめ

次に、DBA が果さなければならない役割の一覧表を示します。

設計	標準のデータ定義 物理データベース セキュリティ、プライバシー、およびリカバリ手順 サポートソフトウェア（DBMS パッケージにないとき）
選択	データベース管理システム パフォーマンス測定ツール チューニングガイド
予測	トランザクション量の変化や新規アプリケーションの効果
決定	検索方法 アクセスメソッド データベース設計 レコード間の関係 データベース使用規則

教育	データベース技術についてアナリストやプログラマの教育 データベースの操作手順についてオペレータの教育
周知徹底	設計、ドキュメンテーションなどに対する標準 品質管理 アクセス
組織編成/管理	データディクショナリの作成と維持 ファイルの変換 完全性、セキュリティ、およびリカバリのベンチマークテスト 受け入れテスト ユーザーへの変更点についての伝達
計測	ハードウェアのパフォーマンス ソフトウェアのパフォーマンス データベース使用統計
チューニング	システムパフォーマンス



## 6 データ定義と制御

---

- アプローチ方法：データの集中管理 ..... 26
- データの信頼性の決定 ..... 18
- アプリケーションの選択：システム開発へのアドバイス ..... 18
- データ収集と整合性確認に対するアドバイス ..... 19
- データベースの内容の定義 ..... 19

このchapterでは、次のトピックについて説明します。

## アプローチ方法：データの集中管理

---

データベースに関係するすべての人は、データ定義について、一定の手法と標準手順を使用する必要があります（これは、データディクショナリを設定する業務とオーバーラップします）。DBAは、このような一貫した制御や標準を系統立て、設定し、維持します。標準を設定するには、関係するすべての部門として、ユーザー部門、DP部門、アプリケーションデザイナーおよびDBMSベンダが協力し、参画する必要があります。

## データの信頼性の決定

---

あるユーザー部門がデータベース内のデータのあるサブセットの維持について、すべての責任を負うようにしなければなりません。これにより、データベースの他のデータに関連して、その最新性、完全性が保証されます。どのユーザーをこれに当てるかは、DBAが決定します。誰が責任を持つかを決定するだけが必要なわけではなく、この決定を他の関係する全ユーザーに知らせ、同意してもらう必要があります。ここではDBAの折衝能力が問われます。現在はユーザーAが責任を持っているが、1年たってデータベースの使用方法が変わったときには、ユーザーBがデータの一部または全部について責任を持つのにふさわしいということもあり得ます。

## アプリケーションの選択：システム開発へのアドバイス

---

データベースについてどのようなアプリケーションを選択するかは、DBAが議長となり会議を行って決定します。ユーザーがこの過程で参加すべきかどうかについては、組織が決定を下します。データベース設計チームがデータ収集や業務の分析を十分に行えば、ユーザーは、一般的にシステムのコスト、機能、実現可能性および拡張性にしか興味を示しません。DBAは、議論した方がよいと思われるさまざまなトピックについて、公正な判断を下す必要があります。

意思決定能力を持つ中核として、DBAと関連スタッフは、システム開発についてアドバイスする地位にあるべきです（ただし、DBMSを使用すべきかどうかについてのみ）。DBAがデータベース環境内で十分に有益な役割を果たし、関係する全部門を十分にサポートできる場合のみ、このようなアドバイスができます。

システム開発に関わる問題に関する限り、一定のデータ定義手順に基づいて、新しいデータやデータ間の関係を定義し記述するという処理は、DBAが責任をもって行わなければなりません。DBAは、組織の論理データベースを記録しながら、どの部分を実装し、どの部分を実装しないかをコントロールしなければなりません。

## データ収集と整合性確認に対するアドバイス

データベース内のデータの属性を記述し定義する一定の手順を設定し実施させるのは、DBA の責任です。また、DBA はデータベースに対する入力を編集しチェックする標準設定を行う必要があります。

データの品質に対する最低限の条件を満足することを保証する他、データベースが実用的であり矛盾がないようにするため、入力の品質を一定にすることは重要です。

データディクショナリは、これらの編集および整合性確認のルールを記録しておくツールとして使用できます。

次の2つのタイプのデータを考慮する必要があります。

個人データ	個人が所有するものとして規定されているデータ。この場合、DBA は十分なデータ整合性確認手順と正当性チェックを行うことを要請するだけです。
共通データ	共通に使用するデータです。このデータを管理すべきユーザーを特定でき、そのユーザーがその責任を引き受ける準備ができている場合を除いては、DBA が、このデータの品質に対して適切なレベルの管理を引き受け、実施する必要があります。

## データベースの内容の定義

データベース環境のドキュメント化の要件の大部分は、データディクショナリでサポート可能です。データディクショナリは、DBA の最も重要なツールの1つです。前述したような一定のデータ定義手順に基づいていなければなりません。ディクショナリには、論理データフォーマットや関係が記録され、大きく分けると次の3つになります。

概念	データとそのデータが持つ現行の関係性
使用方法	現在の使用法
実施	現在、データベースにどのように格納されているか

特定のデータの解釈や使用に関する標準を設定する必要があります。

データディクショナリには、次のものを含まれます。

- 論理データ構造
- 物理ストレージ構成
- データ属性
- データ源の記述

- データの発生源
- 入手方法
- 編集方法およびチェック方法
- 正確さとセキュリティ要件
  - データに対する正確さの要求度
  - データに対するセキュリティ要件
  - アクセス権をもつ人
  - 更新権をもつ人
- レスpons要求：一部アプリケーション分野について、検索やレスポンス要求はどの位か

「[データベースドキュメンテーション](#)」では、これらの要件をより詳細に記述しています。Adabas のデータディクショナリである Predict のオンライン機能によって、ドキュメント化の要件を満たすための作業が大幅に軽減されます。

# 7 データベースドキュメンテーション

---

■ 標準 .....	22
■ データベースの記述 .....	23
■ データディクショナリ、機能と使用法 .....	24
■ Adabas データディクショナリ (Predict) .....	24
■ データベースを使用するアプリケーション .....	25
■ データ源の記述 .....	26
■ データのアクセスおよび操作手順 .....	27
■ パスワードおよびユーザー ID .....	27
■ バックアップ手順 .....	28
■ 再スタートおよびリカバリ手順 .....	29
■ DBMS パフォーマンスおよび計測 .....	29

データベースドキュメンテーションとは、データベースを適切に、効率良く、継続して使用するために必要な手順、標準、指針およびデータベース記述を記録しておくことです。

ドキュメンテーションは、次の人（部門）が使えるものを準備し、それぞれ配布しなければなりません。

- エンドユーザー
- DBA 部門自体
- コンピュータオペレーション部門
- アプリケーション開発部門

これらの人（部門）に対して適切なドキュメンテーションを準備し維持するのは、DBA の役割です。この章では、必要なドキュメンテーションの各種タイプについて説明します。この一覧は、必ずしもすべてを網羅していません。

このchapterでは、次のトピックについて説明します。

## 標準

---

特に次の分野で一貫したデータベース制御を設定し、維持します。

- データ定義：一貫した方法を採用します。
- データ使用法
- データベースの特定の部分の所有権と責任
- データのアクセスと操作：データベース内のデータのアクセス方法と更新方法の標準は、データ保全性を実現するために不可欠です。リクエストをコーディングするときの標準手順を設定することで、誤りの確率を減らすことができます。例えば、キー値の更新については、厳重に制御する必要があります。
- データの編集と整合性確認：DBA はデータベース中のデータの品質を一定のレベルに保ち、データが規則どおりであることを保証するための手順を設定しなければなりません。したがって、DBMSを使用する新規アプリケーションの受け入れテストやシステムにもDBAは参加すべきです。
- コンピュータのオペレーション：コンピュータオペレータがデータベースを扱う際の標準の手順を設定するのもDBAの役割です。この中には、標準バックアップ手順、再スタート／リカバリ手順、およびその他オペレーション方法が含まれます。

データベース環境の標準の設定に従いたくない人がいるかもしれません。標準の状態、また標準を設定する分野を慎重に検討します。一般に、新しい標準を設定するには、すべての関係者がそれを提案として受け入れるためだけでも交渉、調整、妥協が必要です。標準が実用的かどうかを確認する唯一の方法は、実装することです。

どんな標準を設定しても、変更されることがあり得ます。既存の標準の変更は、新しく標準を設定するときよりも厳重にコントロールしなければなりません。変更点は、正式に提案し、全関連ユーザーに変更点を伝達する必要があります。提案した変更に基づき業務を行った後、再吟味し、関連ユーザーと標準とするべきかどうかについて検討します。

DBA は定期的に、データベースの標準の効果を確認し、その標準に従っているかを確認するため、再吟味しなければなりません。この後、正しい方策を採る必要があります。また、データベース環境内で業務を行うすべての人に、データベースに対する標準事項の使用を知らせ、十分な教育を行うことを保証するのも、DBA の基本的な役割です。

データベースの導入先によって手順や要件はそれぞれ異なるので、このマニュアルでもデータベース環境に対する特定の標準については言及していません。

## データベースの記述

データベースの記述は、次の主要分野を満たしていなければなりません。

- 概念データベース：格納データに関する公式記述およびその固有の論理データ構造の定義です。DBA は、データ構造を明確に定義する必要があります。このとき、予想される開発に関する DBA の知識、また他の既存ユーザーや潜在ユーザーの要望を反映させます。
- データベースの使用：アプリケーションレベルと個々のデータ項目レベルの両方についての情報を格納すべきです。新規システムが開発されるとき、このドキュメンテーションは重要な役割を果たします。ここに格納すべき情報については、次項で説明します。
- 実施：データベース中のデータの格納方法です。データベースの物理格納構造に関するドキュメンテーションです。次の事項を含む必要があります。
  - 実装するデータ構造（論理データフォーマットと関係）。これらは通常、概念データベースのサブセットです。
  - ストレージ構造（物理データフォーマットと関係）。Adabas に関しては、フィールド定義テーブル、カップリング関係およびその他固有の関係です。
  - データ量各：ファイルのレコード件数と平均レコード長です。
  - 予想される増大量：ファイルおよびフィールドサイズ（つまりレコード長）別。
  - レコードの追加と削除件数：更新実行の平均件数。追加や削除方法（ユーティリティか、ユーザープログラム）と同様、追加や削除の制限（つまり、ファイル全体にランダムに更新が行われるか、それともある一部に限定しているか）に注意を払うことも大切です。

他に、なぜこの実施方法が選ばれたかも記録すべきです。この情報は、後日メンテナンス時や新規アプリケーションの設計時に有効であることがわかります。

DBA は、上記のような方法で、データベースを正式に記述し、データディクショナリ中に、この記述を維持するのは（この処理を自動化してもしなくても）、DBA の責任です。この業務を行うのに必要な全情報を DBA に供給するのにプロジェクトチームが必要です。

## データディクショナリ、機能と使用法

---

データディクショナリは、データの定義、構造、使用法についての情報をもつものです。実際のデータ自体ではなく、データに関する情報をもっています。簡単に云うと、データディクショナリは、各データタイプ（項目）の名前、その定義（大きさやタイプ）、使用場所と使用法および他のデータ要素との関連に関する情報をもつものです。

データディクショナリにより、DBAは組織のデータリソースの管理や制御がより良く行えます。また、データディクショナリの使用経験ユーザーは、プロジェクト管理やシステム設計においてこれが有効なツールであることを認めています。

また、データディクショナリにより、DBAは実際のデータ項目と、データ項目を使用するプログラムを別々に管理できます。このことにより、実質上データの有効性を高めることができます。データディクショナリは、データをより有効に使用するのに必要な情報を収集する役割を果たしています。

ディクショナリは、データの定義をすべて含む情報の格納場所であり、データ属性、特性、データ源、使用法および他のデータとの相互関係を含んでいます。データディクショナリは、次のような情報を提供します。

- このデータタイプに適用される整合性テストの種類
- このデータタイプを使用するモジュール、プログラム、システムおよびレポート
- 適用されているセキュリティのレベル、データタイプのアクセス権を有する人、このデータタイプの更新権を有する人
- 各種アプリケーション環境でのこのデータタイプの別名
- このデータタイプの入力源
- このデータタイプの文章記述

## Adabas データディクショナリ (Predict)

---

Predict (Adabas データディクショナリシステム) により、データディクショナリの設定と更新ができます。

オンラインモードでもバッチモードでも、ディクショナリに対してデータベースに関する情報を格納できます。Adabasディクショナリ内のデータの記述には、ファイル、各ファイルに定義されたフィールド、ファイル間の関係に関する情報が含まれます。使用法に関する記述には、そのデータを使用するシステム、プログラム、モジュールおよびレポートに関する情報の他、データの所有者と使用するユーザーに関する情報が含まれます。ディクショナリエントリは、次のような情報から成ります。

- ネットワーク構造



- Adabas データベース
- ファイル、フィールド、および相互の関係
- 所有者と使用者
- システム、プログラム、モジュールおよびレポート
- フィールド整合性チェック（処理ルール）

標準のデータディクショナリレポートは、次の情報を出力できます。

- データディクショナリの全内容
- フィールド情報、ファイル情報、相互の関連情報
- ファイルごとのフィールド情報

また、ディクショナリは標準の Adabas ファイルなので、Natural から直接ディクショナリデータにアクセスできます。

Adabas データディクショナリシステムの詳細については、Predict のマニュアルを参照してください。

## データベースを使用するアプリケーション

データベースを使用する各アプリケーションには、少なくとも次の情報が記録されていなければなりません。

- アプリケーションの特質
  - アプリケーションの機能に関する記述
  - 使用モード：バッチ／オンライン、シングル／マルチユーザーモード
  - 使用頻度：スケジュールされた標準回数
  - トランザクションのタイプと量
  - パフォーマンスに関する事項：許容最小限のレスポンスタイム
- ファイルの必要条件
  - アクセスするファイル
  - ファイルのアクセス方法（ディスクリプタの使用）
  - 使用するデータ項目（フィールド、サブフィールド、スーパーフィールドなど）
- セキュリティ要件
  - 暗号化、パスワードの使用（サイファキー、パスワードは特定ユーザーにのみ使用可能とする）
  - システム／プログラム／問い合わせの使用を許可されているユーザー

- バックアップ必要条件：ファイルバックアップの頻度と内容
- 再スタート必要条件

どのようなデータベースも概念データベースの一部を実装しているにすぎず（前のセクションを参照）、ユーザーの要求は時とともに変化するので、時が経つにつれて、データベースの新しいアプリケーションが見つかります。このような新しいアプリケーションは、完全に新規システムとして開発してもよいし、既存システムに追加してもかまいません。このようなアプリケーションは、最初はユーザーの単純な要求として現れます。

DBA は、このような未計画のデータベースの使用を記録しておく手順を設定する必要があります。また、あるアプリケーションが相対的によく使われたり重要になった場合、データベースやデータベース内のファイルの再設計や再編成により、処理効率を上げることができます。例えば、最初にファイルが顧客番号順にロードされたとします。その後、ファイルを営業マン番号順に処理するアプリケーションがより重要になった場合、ほとんどの既存アプリケーションの論理的な処理には影響を与えることなく、ファイルをアンロードし、営業マン番号順にリロードすることができるため、このファイルを使用するすべてのアプリケーションに必要な処理時間は全体的に減らすことができます。

このような計画になかったデータベースの使用や処理優先度の変更は、ユーザーが、このような作業を自分の部門で行うのか、または DBA に依頼して行うのかについての情報を要求してきたときに、記録します。このような記録は、定期的に DBA が再調査し、関連ユーザーと議論しなければなりません。

## データ源の記述

---

新規アプリケーションについては、最初にデータディクショナリを参照し、必要な情報源を判断します。新規プロジェクトのシステム分析および設計段階で、データ源の記述が行われます。

記録する必要がある情報は次のとおりです。

- 現在のデータの形式と位置：形式、ファイル、コンピュータのストレージ媒体
- データを入手するために使用するアクセス方法
- 必要な整合性確認や編集方法も含んだ、現在の正確性、完全性およびタイムリー性に関連するデータの用途
- データベースに格納する前にデータを修正する必要があるか
- データ使用权を有するエージェント
- データ取得コスト

## データのアクセスおよび操作手順

DBA は、データベース中のデータのアクセスや更新処理のすべてについて管理し制御しなければなりません。そうでないと、データベースに対し意味のある制御や保護が行えません。制御が十分でないと、データの安全性や完全性に関して重大な問題が起こることになります。

データベースに対する権限と責任は、組織という枠を超えているため、業務単位ごとのデータベース利用、および業務単位間のデータベース利用を対象とする全社的なポリシーを公表する必要があります。また、このようなポリシーを公表することで、DBA の管理上の制御を高めることになり、ユーザーやデータ処理オペレーション担当者は、データベースに関する手順を理解しやすくなります。

このポリシーには、次のものが含まれます。

- DBMS コマンドの使用 - DBMS が提供する各種機能を使用してよいのは誰か、誰が使用してはいけないか
- データベース使用 - ユーザープログラムが行うのか、標準インターフェイス（Natural や SQL など）を使用するのか、どのようなエラー処理手順を使用するべきか、問題点は誰にどのような形（例：トラブルレポート）で報告するか
- 保守および更新手順の責任の所在は誰にあるか

このポリシーは DBA が提案し、すべての関連部門に評価してもらい賛同を得る必要があります。

## パスワードおよびユーザー ID

DBA と関連ユーザーだけがアクセスできるようにするため、ユーザー ID とパスワードに関する情報は、DBA が厳重に保管する必要があります。このドキュメンテーションとしては、次のものがあります。

- サイファキー、パスワードおよびユーザー ID の割り当て手順
- サイファキー（DBA はこれを知っている必要がある）、パスワードおよびユーザー ID の実際の割り当て
- 端末およびデータアクセス手順
- 各データエンティティに対して、アクセス権を設定する必要があります。次のことを定義します。
  - データ所在、データ内容を誰が知る権利があり、また知る必要があるか
  - 誰がデータベースからデータを読むことができ、データに新オカレンスを追加でき、データの値を変更でき、またデータベースからデータを削除できるか

いったん、この権限が設定されたら、データベースセキュリティに違反しないように、適切な制御手順を設定しておくことが重要です。

- データベースセキュリティ手順 次のようなデータベース内のデータの物理的保護を記録しておきます。
  - データベースに対する絶対的な人的制御（コミュニケーションルーム、コンピュータおよび端末へのアクセス、データベースバックアップやログテープの保管）
  - データエンティティの物理的な分離（ファイルの分離、データベースの分離、部分マウント機能とファイルクラスタ機能の使用）
  - 端末の安全な領域の確保（ロック可能な端末、キーホルダー、専用回線またはダイヤルアップ回線、未使用時のオフライン化）
  - データベースに関して公表される情報を受け取る権限を有する人

パスワード保護を実装、制御するために、Adabas Security ユーティリティを使用します（詳細については、『Adabas Security Manual』を参照）。DBA は、このユーティリティを使用できる唯一の担当者です。

DBA は、セキュリティユーティリティ自体、セキュリティに関する全ドキュメンテーションを物理的に保護するための手順を策定する必要があります。

## バックアップ手順

---

バックアップファイル（AdabasのADASAVユーティリティで取得したデータベースまたはファイルのコピー）は、次の情報とともに記録する必要があります。

- どのような状態でいつ取得したバックアップデータか
- バックアップ可能なデータの識別
- 含まれるデータ量
- データベースをある状態に再設定するのに使用すべきバックアップ機能
- データベースバックアップオペレーションの頻度とスケジュール

DBA は、コンピュータオペレーション担当者と打ち合わせて、データベースのバックアップ作業の実行手順を定める必要があります。データベースバックアップは、データベースが破壊されたり、損傷された場合に、データベースを適切な状態にリストアするために必要なステップです。すべてのアプリケーションについて、データベース全体をバックアップするかファイルダンプ、リストアの方がよいかを決定する必要があります。

特定のアプリケーションに対してバックアップ手順を作成するときは、『Adabas オペレーションマニュアル』を参照してください。

## 再スタートおよびリカバリ手順

DBA は次の手順を公式化し監督する責任があります。

- 障害後の DBMS の再スタート
- 必要なら最新のチェックポイントまでデータベースをリカバリすることで、データベース保守作業を繰り返す必要がなくなります。
- データベース復元の優先度と順番の制御

再スタートとリカバリはデータベース保護の重要な問題であり、DBA は保護機能を行う標準、手順および規則を設定する必要があります。この標準や規則が守られるように監督します。DBMS 導入時に、再スタートやリカバリについて考え設計しておかなければなりません。後から考えるのでは遅過ぎます。

Adabas の再スタート／リカバリに関する詳細は、『Adabas オペレーションマニュアル』を参照してください。

## DBMS パフォーマンスおよび計測

DBA はデータベースシステムのパフォーマンスを維持し、向上する役割を継続します。

これを行うため、DBA はシステムのパフォーマンスを監視し、パフォーマンスを改善するために、代りの設計方法を試さねばなりません。企業内の業務パターンが変わるとトランザクションタイプの量も相対比率も、両方とも変化する可能性があります。したがってパフォーマンスにも影響し、これを防ぐのに設計変更が必要な場合もあります。

より長期間に間、業務の負荷の変化を予測し、再設計や設備拡張によりこれらの変化に対応するよう計画を練ることは可能です。

新規ハードウェアやソフトウェアの効果も評価すべきであり、おこり得る変化はコストに見合っていないければならず、長期対策に盛り込むべきです。

DBMS のパフォーマンスの追跡（および計測）は、したがって DBA 機能の重要な部分です。DBA は、次のような記録を取得し、維持すべきです。

- 使用するコンピュータリソース（アプリケーションごとの使用頻度も含む）
- ある特定のアプリケーションのサービスをどのユーザーが受けるか
- レスポンスタイムやコストに関する DBMS の効果

DBA は、また次のことを行うための手順を設定し、ドキュメントする必要があります。

- DBMS の使用頻度のモニタリング

### ■ DBMS パフォーマンスの管理

データベースの完全性を維持し、かつ効果的サービスレベルを提供することを保証するため、データベース環境を継続してモニタするのはDBAの役割です。このモニタリングの役割は、さまざまな活動や手順から構成され、パフォーマンスの管理もその1つです。

*Adabas Online System* は、データベースのモニタリングを行ううえでDBAの強力なツールとなります。詳細については、*Adabas Online System*のマニュアルを参照してください。

## 8 教育および訓練

---

■ 概要 .....	62
■ データベースの概念 .....	32
■ データベース設計 .....	33
■ プログラミング .....	33
■ オペレーティング手順およびテクニック .....	34
■ データ入力 .....	34
■ データベース問い合わせおよびレポート生成 .....	226

このchapterでは、データベース環境に適切な教育プログラムの主な特徴について簡単に説明します。

次のトピックについて説明します。

## 概要

---

DBA は、データベース概念や、データベース環境内のオペレーション手順やテクニックに関連するすべての人々の教育や訓練についても責任があります。したがってDBAは、教育カリキュラムを組み、使用する教材の内容を選択する必要があります。データベース環境内の実行、オペレーションおよびメンテナンスにたずさわる人々に対し、教育を行います。データ処理部門でない外部ユーザーについては、データベースの概念、データの可用性、データ入力、レポート生成および問い合わせ機能の使用法を教育します。

データベース環境と接触する各タイプの人々に対し、全般的な教育プログラムを作成するのが賢明です。この教育プログラムでは、教育を受ける人が出力の期待値を達成できるように、入力（知識）の期待値と出力（パフォーマンス）の期待値も教育内容に盛り込みます。教育が必要な人は入力基準ですぐに評価し、適切な教育コースに参加する前に補習、またはコース前に読む資料を指示できます。この方法によって、確実に教育の効果が得られます。

各個人の業務要求に応じた教育内容にします。DBA の教育は綿密に計画され、タイムリーでなければなりません（すなわち、要求してから数ヵ月も前後しません）し、教育の直後に補強期間（すなわち、教育を受けた内容の実地使用）が必要です。

DBMS が初めにインストールされたときは、教育が必要な人の数はかなり多くいます。新規プロジェクトが開始したり、新システムがインストールされるときも同様です。これら以外では、教育すべき人数は少なくなります（例えば、新人が入ったとき）。したがって、少人数にはパッケージ化した教育（例えばカセットテープやワークブック）、大人数にはフルコースの教育が推奨されます。

## データベースの概念

---

データベース環境に接するすべての人は、データベース管理システムの概念を理解しておくべきです。この教育での主題には次のものが含まれている必要があります。

- なぜ DBMS が発展したか
- 伝統的データ処理システムと DBMS との類似点と相違点
- データの重複削減の利点と欠点
- DBMS 特有の柔軟性
- 使い易さと理解し易さ



- エンドユーザー機能 - 同一データに対する異なるビュー、DBMS の論理構造機能
- DBMS が提供する機能
- データベース環境内のセキュリティ、データ保全性およびリカバリ手順の重要性
- データベース標準設定および制御の必要性

## データベース設計

データベース設計者は、すぐに成果を上げられるように、そのサイトで指定されている設計方法に関する教育が必要です。

サイトの標準の使用、特にドキュメンテーションを教え、実践するための実践的な演習に時間をかける必要があります。Software AG が一般に提供しているコースではこれができない可能性があります。このコースを受けた場合、受講者は、仕事に戻ってからすぐにサイトの標準に関する教育を受ける必要があります。

教育主題として、次のものを含みます。

- Adabas 機能、制御およびオペレーションの高度な理解
- ファイルのロード、ファイル定義、Adabas ダイレクトアクセスメソッド (ADAM) ファイル、必要ディスクスペースの推定
- トランザクション処理、ET/BT ロジック
- 完全性、再スタート/リカバリ、自動バックアウト、自動再スタート
- セキュリティ、パスワード、暗号化
- Adabas ユーティリティの概要
- プログラムの設計と効率

## プログラミング

コンピュータプログラムの教育は、インストラクションプロシージャと標準に基づいていなければなりません。教育については、練習問題に費す時間の大部分は、できるだけ実地のものにすべきです。

教育コース中、受講者はコンピュータで実際に実行するアプリケーションプログラムを作成した方がよいでしょう。継続して教育の補強を計るため、教育コース終了後この練習問題を完成できるような用意をすべきです。

教育コースで教える主題には、次のものがあります。

- アプリケーションプログラマに適用できる Adabas 機能の概要

- プログラマが使用できる Adabas 直接モードコマンド、高水準プログラミングインターフェイス (SQL、Natural) 機能
- 効率の良い、保守しやすい Adabas プログラム設計

## オペレーティング手順およびテクニック

---

コンピュータオペレータに対する教育は、インストレーション手順および標準に基づいていなければなりません。この教育も、できるだけ実地教育（例えば、アプリケーションシステムの実行、リカバリ／再スタート手順の実行）にすべきです。

教育主題として、次のものを含みます。

- オペレーティング手順、Adabas セッションの起動とシャットダウン、通常のオペレーション、例外処理、リカバリおよび再スタート
- ユーティリティの実行：何を行い、何が得られるか
- コンピュータタイムのスケジューリング、DBA とのコミュニケーション
- パフォーマンス管理
- 制御および監査証跡
- エラーの記録と追跡

明らかに、これらの主題はインストレーションに大きく依存し、その教育については、インストレーションのスタッフが行う必要があります。

## データ入力

---

この教育は、新規にアプリケーションシステムがインストールされる時、ユーザー部門の人に行う基本的なものです。したがって、アプリケーションシステムに大きく依存しています。しかし、いくつかの一般的なガイドラインを与えることは可能です。

教育には、次のものを含みます。

- 入力手順、端末からあるいは入力文書を使用して入力、アプリケーションの規則
- 標準と制御、検査
- 入力に対しシステムが何を行うか
- 入力エラーとその修正

## データベース問い合わせおよびレポート生成

---

このタイプの教育内容は、データ処理部門の人かユーザー部門の人に大きく依存します。前者については、使用する Adabas 問い合わせ機能（例えば Natural）のコマンドと機能および要求の構成や実行の詳細の教育が必要です。

他方、ユーザーについてはより特別な教育が必要であり、使用するアプリケーションシステムに密着した事項を教育すべきです。

教育すべき事項としては、次のものがあります。

- Natural や SQL のような問い合わせ機能の働き（概要のみ）
- アプリケーションシステムが作成する標準レポート（その内容と適応性）
- 問い合わせ機能のコマンド、機能、使用方法（適用される標準を重点的に）



# 9 DBA とユーザー

---

■ ユーザーとの連絡 .....	38
■ アクセス要求 .....	39
■ アプリケーションインターフェイス .....	40
■ 標準および制御の順守 .....	40

通常のデータベースアプリケーション開発サイクルと其中でのDBAの役割を検討する前に、DBAはユーザーがデータベースに何を求めているのかを理解する必要があります。ユーザーという言葉は、最も広い意味で、ユーザーマネージメントおよび人、データ処理マネージメント、コンピュータオペレーション、プログラミング、システム、ソフトウェアサポートおよびDBAの部門を包含しています。

DBAとユーザー部門間の関係はデリケートであり、特に、あるユーザーがデータベース環境でない場合よりも明らかに低いレベルのサービスを受けたり、労力が余計にかかった場合がそうです。

DBAは、組織の全体としての発展を計り、（そのポリシーをサポートする）公明正大な偏りのない権限を持つとユーザーが感じられるようにしなければなりません。

DBAは長期計画と長期にわたるユーザー要求の両方に注意を払わねばなりません。DBAは、ユーザー間や、特定ユーザーと企業プラン間に起きた問題の調停を行わねばなりません。

新規アプリケーション開発の間は、DBAはプロジェクトグループとエンドユーザーの仲立ちをする必要があります。

このchapterでは、次のトピックについて説明します。

## ユーザーとの連絡

---

ユーザー（プログラマやエンドユーザー）との連絡は、DBAの仕事の中でも、最も重要であり、最もデリケートです。

エンドユーザーの要求（通常は検索要求に関する）に応じて、DBAは本番データベースについてのドキュメンテーションをチェックし（特に論理データ構造）、この要求を直ちに扱えるかを調べます。

その基本的な結果としては、次の4種類があり得ます。

1. 現時点で要求に応えることができない  
この場合、DBAは要求を記録し、一定の間隔で見直して状況（ニーズ）が変化したかどうかを確認する必要があります。
2. ワンタイムリクエストの準備  
Naturalを使用し、既存データベースからの要求を最小限の労力で満足させることができる場合があります。DBAの部門が実際に要求を満足させるアプリケーションを作成するか（ユーザー部門にその能力がある場合がある）、プログラマがアプリケーションを作成するかに関係なく、重要な点は、DBAがデータベース情報に対する要求の解決策を定義することです。

3. 新しいアプリケーションプログラムの作成  
アプリケーションを定期的に行います。DBAはプログラムを指定し、プログラムの作成とテストについてプログラママネージャと交渉します。
4. 既存のアプリケーションの変更  
もちろんアプリケーションシステムの責任者との交渉（要求を出した人と異なるとき）を含みます。変更はシステムのパフォーマンスや柔軟性に影響するからです。

エンドユーザーの要求は結果がどうであれ、十分にドキュメントにしておくべきです。このような要求事項の議論はDBAにフィードバックする最も有効な情報の1つであり、エンドユーザーに対する教育により、十分効果が上がったか、上らなかったかを示しています。

通常データ処理部門からの要求は次のものです。

**教育** DBAは、要求者がどの教育をいつ必要としているかを定める必要があります。このような教育はすぐには提供できないことをデータ処理マネジメントに気付かせる必要があります、あらかじめ適切な教育プランを立てておくべきです。

**情報** DBAは、要求者に知らせる必要があります、その情報を得る権利のある場合、それを満足させる必要があります。これらの質問のいずれかに対する解答が通常の状態から引き出せるとき、既存の標準や手段に一時的または長期的に調整を行う必要がある場合もあります。

**問題** DBAは解答を知っているか、またはSoftware AGに問題を問い合わせる必要があるかどうかです。後者の場合、DBAは、できるだけその問題についてのドキュメンテーションを作成すべきです。

**援助** これは、いろいろな場合があります。DBAは、要求されている援助を提供するのはDBAの責任であることを保証しなければなりません。

## アクセス要求

DBAは、データベースのアクセスや更新に対する管理、制御を行わなければなりません。DBAはユーザーとともに、データベースの各項目に対するアクセス権を設定しなければなりません。これらのアクセス権や更新権のほとんどは、データを使用するアプリケーションシステムの設計時にはっきりし、データ項目は正しく保護されます。突発的な要求が起こった場合、ユーザーはDBAと相談すべきです。このときデータベースドキュメンテーションを参照し、DBAはユーザー要求を満たす最善の方法をアドバイスできます。さらに、その要求が存在すること自体がデータベース使用に関するDBAのモニタリングの有効な入力情報となります。

データベースに対するアクセス要求は（アプリケーションシステムの一部でも突発的な要求でも）データベースのユーザービューまたはサブスキーマと考えられます。その内容、セキュリティ、アクセス/操作モードのすべてを相談し記録しておく必要があります。

場合によっては、アクセス要求はアプリケーションシステムの境界を超えることがあります。この場合、DBAはデータ項目のアクセス権についてデータ所有者と相談する必要があります。

## アプリケーションインターフェイス

---

ドキュメンテーションの標準には、アプリケーションプログラムと DBMS との通常のインターフェイス方法の記述を含む必要があります。次に示すように、2つのアプローチのいずれかが使用できます。

1. ホストプログラミング言語からの Adabas ダイレクトコール
2. アクセスモジュールへのサービスコール

いずれのアプローチが採用されても、標準インターフェイスポリシーに固執するのが不適當であり望ましくない場合もあります。このような場合、事前に DBA に相談し、その許可を得るべきです。

突発的要求を扱うとき、DBA はユーザーに対してどのインターフェイスアプローチを採用すべきかをアドバイスする必要があります。これは、アプリケーションプログラム（SQL アクセスモジュールの有無は関係ない）、Natural、またはその他です。

## 標準および制御の順守

---

DBA はユーザーに対し、データベース標準および制御に従うことで生じる利益について説明し、ユーザーがこれを無視した場合に起こり得る問題について十分に説明しておかなければなりません。

ユーザーと DBA の相互の信頼を深めなければなりません。DBA の権限が公正であり偏りがなく、DBA の決定が利益をもたらす組織全体のポリシーを支援していると、ユーザーが感じることができなければなりません。

Natural を使用してユーザーにデータベースアクセスを許可している場合、ユーザーに突発的な要求を記録させ、DBA に定期的にこれらの要求事項を知らせるべきです。DBA がデータベース環境の効果を継続的に保証するのに役立つフィードバック／モニタリング情報の一部となります。



# 10 DBA とアプリケーションの選択／開発

---

■ 機器構成とアプリケーション開発のプランニング .....	42
■ データベース編成 .....	42
■ ユーザー要求の理解（現在および将来） .....	43
■ データベース活動の調整 .....	43
■ アクセス要求の分析 .....	43
■ データ使用のための準備 .....	44
■ パフォーマンス対柔軟性 .....	44
■ アプリケーション／プログラム／データベースの設計へのアドバイス .....	45
■ 必要物理ストレージの決定 .....	45
■ テストデータベースとテスト方法 .....	46

このchapterでは、次のトピックについて説明します。

### 機器構成とアプリケーション開発のプランニング

---

DBA は、データベース使用の知識とそのパフォーマンスをモニタすることで、機器構成やアプリケーションプランニングについて、データ処理マネージメントの決定に重要な役割を果たすことができます。DBA は、日々の問題や困難さとともに、ユーザーの短期および長期の要求を知っています。DBA は Software AG と接触し、DBMS に対する開発予定も知ることができます。したがって、DBA はこのような議論に参加するべきです。

DBA は、アプリケーション開発プロジェクトに最初から参加すべきです。DBA は、組織のデータ処理開発計画の観点から、データベースアプローチが正当かどうかを決めるため、初期調査を助けることができます。

データベースプロジェクトの初期段階以後も、DBA は継続してプロジェクト開発に参加すべきです。新規プロジェクトの追加は特殊な問題を生じる場合があり、DBA は注意深く解決しなければなりません。

既存データの使用変更や新データの追加は、既存システムのパフォーマンス特性を変える可能性があります。全ユーザーに妥当なサービスを行うためには、物理構造やデータの位置の再設計が必要な場合もあります。

### データベース編成

---

すでに述べたように、DBA は論理データ構造を定義するためにデータ間の関係の公式化と定義を行う責任があります。これらのデータ構造には、予想できる開発に対する DBA の知識が反映され、他の関連ユーザーの要求が含まれる必要があります。

次の2つの主要な観点があります。

- 既存データの定義と編成
- 新データの追加

効率良い物理構造にするには、論理関係の解釈や導入についてかなりの専門知識が必要です。スペース、パフォーマンスおよびコストは、均り合いがとれていなければならず、次に考慮すべき点を示します。

- データ構造（論理データフォーマットおよび関係）
- ストレージ構造（物理データフォーマットおよび関係）
- アクセスメソッド（使用可能な、および使用すべき）
- アクセス頻度

- 必要な物理ストレージ媒体
- タイミングに関する配慮
- 検索方法

解答（およびその背後の理由）は十分にドキュメントにしておくべきです。

## ユーザー要求の理解（現在および将来）

DBA は、プロジェクトチームのメンバがユーザーの現在および将来の要求を理解し、認識できるように援助できる理想的な立場にあります。この場合、ユーザーは最も広い意味で使用しています。アプリケーションシステムを設計し開発するユーザー部門だけでなく、組織全体として忘れてはならない他の潜在ユーザーも含んでいます。また、すでにわかっている将来の開発予定も考慮すべきであり、場合によっては、DBA は新規アプリケーションの開発全体を統制する必要があります。これは、このような開発が、完了後にはデータベースオペレーションの一部になることもあるからです。

## データベース活動の調整

このドキュメンテーションの内容を実行に移せば、DBA はすべてのデータベース活動を調整できるはずですが、DBA は、データベースに対するすべての開発計画にアドバイスする必要があります。また、組織全体を対象とした総合情報システムに向けて、安定した、よく制御された経過をたどるよう保証する必要があります。

一般に、データベースの実際の使用に関するアドバイスをし、DBA の品質管理の職務を遂行するために、DBA は新規プロジェクトの成否の検討段階からはじまり、すべての段階に参画すべきです。

したがって、DBA は、プロジェクトチーム間、現在のユーザー、将来的なユーザーに対して一貫した窓口として機能します。全ユーザーの利益が最大となるようデータベース設計の心構えを奨励するのがその目的です。

## アクセス要求の分析

これは、データベース設計の重要な部分です。新規プロジェクトがデータ分析とファイル設計の段階に入ったら、DBA は新規アプリケーションシステムで使用するデータへのアクセス要求の観点が狭くならないように監督することが重要です。

アクセス要求の分析も行うべき作業です。組織の要求が変わると（時間とともに必然的に変わるため）、DBA は要求の変更についてのフィードバックを受け取りますが、新規要求に過度に

対応しないよう注意しなければなりません。つまり、そのときだけに必要なことかもしれないからです。組織のアクセスや処理要求に対する重要度が、少しずつ変化しているときや、目立って変化したときは、その都度、DBA は対応する必要がありますが、そのような対応は関連部門すべてと十分議論した後だけに限定します。

## データ使用のための準備

---

DBA は、プロジェクトチームを援助し（おそらく、データディクショナリを使用し）、次の点を考慮し、適切なデータ取得プログラムのプランニングを行うべきです。

- 現在のデータの形式と位置
- データの収集方法
- 現在のデータの正確さと完全度
- 最終的にデータベースに格納する前にデータに行う必要のある変更事項
- アプリケーションシステムの導入に関していつまでにデータを収集すべきか 中間の期間をどのように処理するか

このプロセスは、データ収集プログラムで行うこととなります。このプログラムはデータディクショナリに記録するために DBA に情報を提供するとともに、要求される特殊な編集や整合性確認について必要な処理を含みます。

## パフォーマンス対柔軟性

---

データベース（の一部）の設計は、本来、柔軟性（将来的に新しいニーズが出てきたときに容易に対応できること）と、パフォーマンス（ディスクスペースの有効利用やコンピュータ処理時間）の考察を含みます。

DBA は、プロジェクトチームが柔軟性とパフォーマンスのどちらに優先度を設定するのかを確認しなければなりません。DBA はアプリケーションの分野で必要な柔軟性（すなわち、計画中のシステムもこのデータを使用する）やパフォーマンスの向上のための設計に関してプロジェクトチームにアドバイスするのに都合の良い地位にあります。このようなパフォーマンス対柔軟性に対する考察の最終目的は、この一方だけを考慮し、他方を犠牲にするような決定を下すことを防ぐことです。

## アプリケーション／プログラム／データベースの設計へのアドバイス

DBA は、プロジェクトチーム、他の Adabas ユーザー、および Software AG と接触することで、アプリケーションプログラムとデータ構造の設計についてのかなりの知識を漸次得ることができます。DBA はその地位により、組織全体に適切な情報を流します。さらに、アプリケーションの設計についてアドバイスします。

DBA 自身は実際にデータベース設計を行わないかもしれませんが、ファイル／レコードの設計やディスクリプタの選択、その他については、プロジェクトチームのメンバにアドバイスできます。これによりデータベース設計において、他のユーザーの要求を反映させる機会ができます。

DBA は物理データベースの設計が、後のプロジェクトの成功を損うことなく、最初のアプリケーションの論理要求を効率良くサポートできるようにする必要があります。DBA は、アプリケーションシステムに必要なセキュリティ、完全性およびリカバリ、設計規則や手順を満足させるため Adabas をどのように使用するべきかをアドバイスします。ある場合、追加のソフトウェアが必要かもしれないし、その場合は DBA がその設計を援助します。DBA は、データベースのセーブ／リストア、パフォーマンスの測定、データベースの実際の内容の分析等に追加ユーティリティが必要かどうかを考慮します。

この段階では、DBA はデータ分析の最善のアプローチ、Adabas の使用法、論理データ構造の設計方法およびどの設計オプションが最も効果的であるかについても、アプリケーションプロジェクトチームにアドバイスできます。

## 必要物理ストレージの決定

DBA は、プロジェクトチームが特定のアプリケーションシステムに必要な物理ストレージを決定するときに援助する責任があります。

次のパラメータを考慮すべきです。

- 格納するデータの量
- データの予想増加量
- レコードの平均長
- ある期間内の追加レコード数と削除レコード数
- データの関連（データ構造）
- データの表現（内部フォーマット）
- 圧縮の効果
- データに対して使用すべきアクセスメソッド

ストレージ媒体と処理コストを最小にするのか、サービス（スピードとスループットについて）を最大にするのかを十分に比較検討する必要があります。また、アプリケーションシステムの実装に必要な柔軟性も配慮する必要があります。この要件は将来的に変化する可能性があります、また他のアプリケーションシステムがこのアプリケーションのデータにアクセスする必要がある場合があります。この場合、必要物理ストレージを最小にすることで柔軟性が失われる場合、将来起こり得る問題点を考慮せずに、この決定を行うべきではありません。DBA はもちろん、理想的にこの種の情報をプロジェクトチームに提供する位置にあります。

## テストデータベースとテスト方法

---

DBA は新規アプリケーションに使用するテストデータベースのタイプについて、プロジェクトチームにアドバイスする必要があります。テストを支援するために、テストデータベースを設定します。システムテスト中にDBA自身のモニタリング、監査、エラー修正および制御の手順でテストしてから、システムの本稼働を開始します。これらの手順はシステムが本稼働してから設計したのでは遅すぎます。アプリケーションシステムの開発と並行してDBAがこの手順を作成します。

テストデータベースについては、本番データベースと分離し、別のディスクパック、別のデータベースにロードするのが最善の方法です。これには、次の2つの問題があります。

- マルチユーザーモードで本番業務と並行してテストを行うことはできません（シングルユーザーモードではこの問題は起きません）。
- 本番データ（あるいは本番ファイルのいくつかのフィールド）が新システムのテストに必要なこともあります。

1 番目の問題については、ストレージの容量が許せば、Adabas ニュークリアスを2つ同時に実行して、1つの本番用、1つをテスト作業用に割り当てることで解決できます。

2 番目の問題については、Adabas の ADASAV ユーティリティを使用して本番データベースから必要データをテストパックにコピーすることで解決できます。この場合、テストデータに対するアクセス権は、テストが始まる前に承認されていなければなりません。

テストデータベースを別に持つことで、システムが本稼働するときのファイル番号でファイルをロードできるし、テストにより本番データベースを破壊することもないという利点が生じます。既存ファイルにフィールドを追加したり、新規アプリケーションシステム用にディスクリプタを新規に設定する場合、特に重要な考察事項です。

システムテストを行う前に、ファイル変換やデータベース初期化をどのように行うかを決め、必要な特定の変換やセットアッププログラムが全部揃えておきます。並列運用の方法は十分に配慮する必要があり、ここでも既存システムからの出力と新システムからの出力を比べたり、正当性確認を行ったりするために、特定のプログラムが必要な場合があります。特定の問い合わせ機能（例えば、Natural）もテストや並列運用を助けるのに必要かもしれません（これらは、テストだけでなく以降の本番運用でも有効なことが多い）。

新規データベースが最終的に実装する前に、システムのあらゆる観点（パフォーマンスと弾力性を含む）が十分かを調べる受け入れテストを実行します。これらは、並列運用に追加してもいいし、しなくてもかまいません。

新プロジェクトのデータアクセス方法を綿密に制御することは、データベースの既存ユーザーのデータ安全性が損なわれないようにするために必要です。システムテストについては、データベースへのテスト変更が、実際の運用に使用するデータベースに影響しないように、特別なテストモードが必要である場合があります。





# 11 DBA とコンピュータオペレーション

---

■ データベース管理者の影響 .....	247
■ コンピュータ時間のスケジューリング .....	50
■ オペレーティング手順 .....	50
■ 再スタートおよびリカバリ手順 .....	51
■ データベースユーティリティ .....	51
■ Software AG との関係 .....	52

このchapterでは、次のトピックについて説明します。

### データベース管理者の影響

---

DBA はコンピュータオペレーション機能が、データベース環境に関する義務を果たしていることを保証する責任があります。この責任とは、データベースに関連するオペレーティング手順、再スタート/リカバリ手順、特定のデータベースユーティリティおよびデータベース関連業務のコンピュータ時間のスケジューリングに関してです。

DBA はさらに、データベースの使用に関しての手順や保護について日々の管理を行う必要があります。

DBA は、ダンプやログが正しく採られているかを見守り、オペレーション手順が正しく行われているかを確認します。また、リカバリシステムを定期的にテストします。

緊急の場合、リカバリ制御にも DBA は参画し、ユーザーと問題点に関して議論し、データの破損を最少に食い止めるよう努力すべきです。

### コンピュータ時間のスケジューリング

---

DBA は、コンピュータのスケジューリングについてもある程度コントロールする必要があります。これにより、問題が発生した場合のスケジューリングが容易になり、緊急のときデータベースを優先使用させることができます。

コンピュータスケジュールの直接のコントロールはコンピュータオペレーション担当者が行うが、データベース処理に関係する事項については、そのスケジュール決定について DBA にある程度の選択権があった方がよいでしょう。そうすれば、例えば同時更新の問題を避けたり、相対的に頻度の少ない負荷ピーク時に対するレスポンスタイム要求も無駄な努力を払うことなく満足させることができます。

### オペレーティング手順

---

DBA は、データベース関連ジョブのオペレーションについて、コンピュータオペレーション担当者とともにオペレーティング手順を正式にドキュメント化しなければなりません。

次のことを考慮すべきです。

- 新データベースのロード
- データベースユーティリティの実行
- データディクショナリの保守

- データベースの保守
- バックアップ手順
- 再スタート／リカバリ手順
- 本番およびテスト要求

## 再スタートおよびリカバリ手順

DBA は、データベースが破壊された場合、適切な状態にデータベースをリストアできるように保証しなければなりません。再スタートおよびリカバリは重要な保護方法であり、DBA はこれらについての標準、手順および規則を設定する必要があります。

コンピュータオペレーション担当者は、これらの標準と手順を学習し守る必要があります、これにより、データ安全性を損うことなくデータベースのリカバリ、再スタートを実行できます。

標準方法のいろいろ（例えば、あるアプリケーションシステムの再スタート後プログラムを実行すべき順序など）も、そのアプリケーションのコンピュータオペレーション規則指示書に記載しておかなければなりません。

## データベースユーティリティ

DBA は、Adabas ユーティリティの使用や、データベース内のある機能を簡単にするための特定のユーティリティの開発や取得についても責任を負います。これらのユーティリティには、次のものが含まれます。

- 実際のデータベースの全機能を含む適切な大きさのテストデータベースの生成（ADALOD ユーティリティ）
- 各ファイルまたはデータベース全体のセーブ／リストア（ADASAV ユーティリティ）
- データベース内のデータの完全性を見るためのレポート自動作成（ADAREP ユーティリティ）
- セキュリティ違反の自動レポート機能（ADALOG 機能）

DBA は、ユーティリティ実行時、ユーティリティの使用権を持つ者が使用しているか等のコントロールを行うべきです。ユーティリティを使用する前に DBA の許可を受けるべきです（ドキュメントが整ったテスト済みのリカバリ／再スタート処理は除きます）。

## Software AG との関係

---

DBA は組織と Software AG との間のメインの連絡窓口になります。DBA は Software AG との間で次のことを行います。

- 組織スタッフに対する教育や訓練の入手
- Adabas ニュークリアスやユーティリティへのシステム変更および新規リリースの受け取りとインストール
- 電子ドキュメンテーション、マニュアルやその他の資料の受け取りと配布
- アドバイスの入手
- 問題の報告
- システムの改良提案

このセクションでは、上記の事項を詳細に説明します。

### 訓練および教育

Software AG は、次の 2 種類の教育と訓練のコースを提供しています。

*組織内* 特定のユーザーサイトの要件に合わせてカスタマイズします。

*オープン* 一般的な情報を提供し、すべてのユーザーが参加できます。

組織内訓練は通常は Adabas システムが初めて導入されたときに行いますが、その後、このようなコースを追加で実施する必要が生じる場合もあります。このコースは顧客の特定のニーズや訓練の目的に合わせてカスタマイズできます。

オープンコースはより一般的で、広範囲をカバーしていますが、DBA が定義する特定の要件をすべて満たさない可能性があります。この場合、DBA は、目標を達成するために補足的な訓練を用意する必要があります。

Software AG は次の分野の訓練を提供しています。

- Adabas によるアプリケーションプログラミング
- データベース設計
- 問い合わせ機能（例えば Natural）
- Adabas システムの内部仕様

推奨される順序、前提条件、スケジュール、申し込みに関する情報など、訓練の詳細については、Software AG 営業部門にお問い合わせください。

## 新規リリース

新規リリースのチェックが完全に終わると、自動的に全 Adabas ユーザーサイトに対して自動的に配布されます。このとき、新リリースへの移行方法についての指示書も一緒に配布されます。

新リリースについては、DBA は本番業務を新リリースに移行する前に完全にチェックする必要があります。このような場合には、以前に可能であった機能が新リリースで正しく動作することを調べる最良の方法としては、標準のテストプログラム群（ジョブストリームの形式）を使用することです。このテストジョブストリームは、Adabas に新機能が追加されるごとに増えていきます。

## ドキュメンテーションと更新の配布

DBA は、Software AG からの新しい資料の唯一の受け取り人として、配布されたコピーを記録し、資料が可能な限り最新の状態を保つようにする必要があります。許可されたドキュメント保持者の記録を維持するのは簡単で、これがおそらく DBA がこの役割を果たすことができる最も簡単な方法です。

## Software AG のアドバイスや相談

Adabas システムを初めて導入するとき、Software AG は、Adabas をユーザーのシステムライブラリにインストールし、テストデータベースを生成し、チェックアウトテストを実行する手助けをします。

この期間を過ぎてから、DBA が Software AG にアドバイスや相談を求める必要が生じる場合があります。そのような依頼は必ず DBA を通してください。

Software AG は、予定されている Adabas パッケージの拡張について DBA に情報を逐次提供します。原則として、このような拡張は、Software AG によって定義が確定したら、訓練コースに組み込まれます。ただし、場合によっては、新しい機能が公開されたらすぐに利用し、後で設計やプログラミングを手直ししなくてもいいように、DBA がこのような情報を既存のプロジェクトに伝える必要があります。

## 問題の報告

データベースに問題が見つかった場合、通常は Software AG に問い合わせなくても DBA が解決できます。ただし、Software AG も、可能な限り短時間で運用を再開できるように、包括的なサポートを提供しています。DBA は、問題を正確かつ簡潔に Software AG の技術サポートチームに報告することで、このサポートの効果を高めることができます。万一の場合に備えて、まずすべての出力を書き留めるか収集し、必要が生じるか、求められたら Software AG に送付します。

### DBMS の改善

モニタリング、監査、運用の作業の結果、システム内で改善できる可能性がある箇所が論理的に生じます。これらの改良見込み事項の評価や、改良作業の準備は DBA の責任です。Software AG は、システムのユーザーグループを奨励、サポートしています。ユーザーグループは、このような改良点を議論するために最適なフォーラムです。ユーザーは、変更／改良要求をユーザーグループの担当者に送信することでプロセスを開始できます。

# 12 データベース設計

---

この章では、データベース設計に関する情報や手引について説明します。説明する内容は、パフォーマンス、ファイル構成、レコード設計、ディスクリプタの効率的な使用、Adabasダイレクトアクセスメソッド（ADAM）の使用、ディスクストレージスペースの格納方法、データベースのリカバリと再スタート、およびセキュリティです。

この情報は次の項目で構成されています。

- システム設計におけるパフォーマンス制御
- ファイルおよびレコード設計
- データアクセス手法
- ディスクスペースの使用法
- *Adabas Security*
- リカバリ／再スタート設計
- *Adabas Recovery Aid*
- マルチクライアントサポート
- 拡張ファイル





# 13 システム設計におけるパフォーマンス制御

---

- システム設計におけるパフォーマンス制御方法 ..... 58

システムのパフォーマンスの判断基準は、実行に必要な時間とコンピュータリソースです。これらは次のような理由から重要です。

- 指定の時間枠内であるシステム機能を完了しなければならない場合
- 厳しい時間の制約のある他のシステムと、コンピュータリソースの競合が起こる場合

しかし、パフォーマンスが最も重要な目標であるとは限りません。パフォーマンスと次の項目のどちらを優先するかについてよく考える必要があります。

- 柔軟性
- データ独立性
- 情報のアクセスし易さ
- 検査およびセキュリティに関する配慮
- 情報が最新であること
- データベースを同時使用するユーザーのスケジュールのしやすさ、影響
- ディスクスペース

パフォーマンスが、最適化すべき目標ではなく制約になる場合もあります。システムは時間やボリュームの問題が生じたとき、パフォーマンスよりは他の項目に注意を向けるべき場合もあり得ます。

このchapterでは、次のトピックについて説明します。

## システム設計におけるパフォーマンス制御方法

---

満足のいくパフォーマンスを達成するためには次の事柄に影響を与えます。

- データベースの設計
- データベースを最初にロードするときに定義するオプション
- アプリケーションの機能のロジック（例えば、ダイレクトアクセスを使用するか、ソートと順次アクセスを組み合わせるか）
- オペレーション手順とスケジュール

どの程度のパフォーマンスが必要かはシステム設計段階の早い時期に考慮しなければなりません。パフォーマンスを制御するうえでの基本要件として、次の事項を確認します。

1. 主なシステム機能について、ユーザーに時間の制約を聞きます。これら要求事項は絶対的と思われる。すなわちこの制限が満たされないとシステムは使いものになりません。
2. ユーザーおよびオペレーション担当者からコンピュータリソースに関する制約を聞き、最も厳しいものを使用します。

3. 次の内容を指定して、論理的な設計モデルにおける各機能を説明します。
  - 各レコードタイプの処理方法
  - 各レコードのアクセスパス、また各レコードが必要になる順序
  - 実行の頻度と回数
  - 稼働時間
4. 最もパフォーマンスがクリティカルであるプログラムを判断します。他のシステムのスケジューリングやパフォーマンスに関する影響、量、頻度、期限等を考慮し、選択します。他のプログラムにも、クリティカルな機能を最適化できるようにその範囲を制約する可能性がある最小のパフォーマンス要件があることも考えられます。
5. クリティカルな各機能について、アクセスパスを短くし、そのロジックを改良し、オーバーヘッドを増加するようなデータベース機能を削除して、パフォーマンスを最適化します。第1のパスでは、柔軟性、情報の取り出し易さ、あるいはシステムの他機能要求を犠牲にすることなく、パフォーマンスを最適化する努力をしなければなりません。
6. クリティカルな各機能のパフォーマンスを推定します。この推定値では満足できる結果にはならない場合、時間の制約か機能要求を少しゆるめるよう調整するか、ハードウェアを向上させる必要があります。
7. その他のシステム機能のパフォーマンスを推定します。総費用を計算し、金銭上の制約と、費用やリソース要求のピーク期間とを比較します。この推定値が制約を満たさないと、ユーザー、オペレーション部門あるいはシニアマネージメントと調整を行い、解決しなければなりません。
8. できれば、テストデータベースをロードし、各機能の時間を計測して、推定値を検証します。テストデータベースのファイル内のレコード数やディスクリプタの値の数は、本番データベースと類似しているべきです。各レコードサイズは順次処理のテスト以外ではあまり重要ではありません。レコードを物理順に近い形で処理するときのみ重要です。



# 14      ファイルおよびレコード設計

---

■ マルチプルバリューフィールドピリオディックグループ .....	62
■ 1つの Adabas ファイルへの異なるレコードタイプの収容 .....	65
■ 1つの論理ファイルへの物理ファイルのリンク .....	66
■ データの重複 .....	67
■ Adabas レコード設計 .....	67

概念設計段階で分けたレコードタイプごとに1ファイルとした Adabas データベースを設計することができます。この構成はどんなアプリケーション機能をもサポートし、突発的な要求を扱うのに最も簡単ですが、パフォーマンスという観点からみると必ずしも最良とはいえません。

- Adabas ファイルの数が増えるにつれ、Adabas コールの数も増えます。各 Adabas コールは解釈、整合性チェックが必要であり、マルチユーザーモードではスーパーバイザーコール (SVC) とキューイングのオーバーヘッドが発生します。
- 1つ以上のインデックス、アドレスコンバータ、また各ファイルのデータストレージブロックにアクセスするために必要な I/O オペレーションに加えて、レコードタイプあたり1ファイルの構造にはバッファプールスペースが必要なので、後で要求に必要なブロックが上書きされる可能性があります。

したがって、クリティカルなプログラムが使用する Adabas ファイルの数を減らした方がよい場合があります。それには、次のような手法を使用します。

- マルチプルバリュースフィールドとピリオディックグループを使用します。
- Adabas ファイル内に複数のレコードタイプを入れます。
- 複数の物理ファイルをリンクして1つの論理（拡張）ファイルにします。
- データの重複を制御します（リソースの使用を少なくします）。

これらの各手法については、後述します。

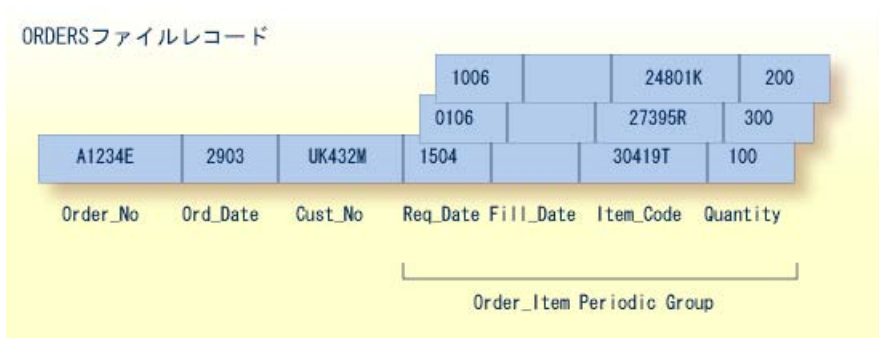
このchapterでは、次のトピックについて説明します。

## マルチプルバリュースフィールドピリオディックグループ

---


次の例は、ピリオディックグループの実用的な使い方を示しています。

注文番号	注文日	発送日	顧客番号	入手日	品目コード	数量
A1234E	29MAR	--	UK432M	10JUN	24801K	200
		--		15APR	30419T	100
		--		01JUN	273952	300



### ピリオディックグループの例

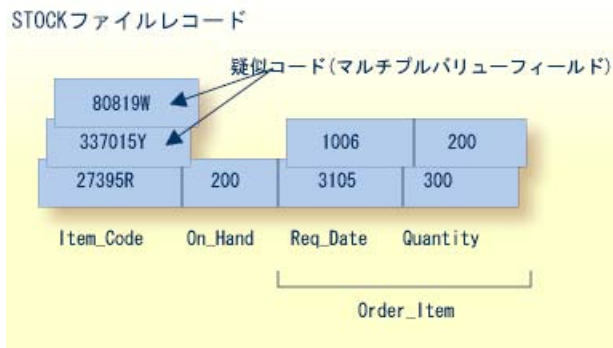
上の例では、表に示されている注文情報が ORDERS という名前の Adabas ファイルのレコードフォーマットに変換されています。すべての注文レコードにはピリオディックグループが含まれており、注文品の数が可変であることを可能にしています。この場合、ITEM\_CODE フィールド（品目コード）と関連フィールド QUANTITY（数量）、REQ\_DATE（入手日）、FILL\_DATE（発送日）を含むピリオディックグループ ORDER\_ITEM で、1つのレコードに最大 65,534 個の異なる品目を指定できます。ORDER\_ITEM ピリオディックグループの1回ごとの出現をオカレンスといいます。ピリオディックグループあたり最大 65,534 回のオカレンスが可能です。

 **Note:** ファイル中で191個を超えるMUフィールドまたはPEグループを使用することは、そのファイルが明示的に許可されている必要があります（デフォルトでは許可されません）。この場合には、ADADBS MUPEX 機能または ADACMP COMPRESS MUPEX および MUPCOUNT パラメータを使用します。

ピリオディックグループ構造を選択した理由は、ピリオディックグループのユニークな特徴、すなわち、オカレンスの順番を維持する能力のためです。最初に3つのオカレンスが含まれていたピリオディックグループは、第1または第2オカレンスが削除されると、それらのオカレンスが空値にセットされます。第3のオカレンスは、そのまま3番目の位置に残されます。これとは対照的に、マルチプルバリューフィールドは、先行する空値エントリを異なった方法で処理します（下記参照）。

また、ORDERS ファイルのレコードフォーマットは、論理的には必ずしも最適ではありませんが、データベースのスペースを節約するために、空値を含む可能性が高いフィールドはレコードの終わりにまとめておく必要があります。したがって、ピリオディックグループを構成するフィールドは、レコード内のその他のフィールドの後に結合させます。

ORDERS ファイルのレコード構造は、注文管理に適していますが、在庫管理には適しません。ORDERS ファイル中の品目の在庫管理アプリケーションは、まったく異なるレコード構造を必要とします。これらのレコードは、STOCKと呼ばれる異なるデータベースファイルに保持されています（下図参照）。



### マルチプルバリュースフィールドの例

STOCK のレコードフォーマットは、ORDERS ファイルのフォーマットよりも在庫管理アプリケーションに適しています。このレコード設計は、在庫がなくなった他品目との交換用として品目を指定しています。ITEM\_CODE（品目コード）フィールドにマルチプルバリュースを入力できるので、現在の在庫品目は、すでに新しい品目に置き換えられている在庫切れになった品目の番号でも参照できます。つまり、旧品目を参照すると自動的に新しい在庫品目を選択されます。このためには、ITEM\_CODE（品目コード）フィールドをマルチプルバリュースフィールドとして定義する必要があります。

例えば、品目 80819W と 337015Y が在庫切れになると、それらの品目コードは品目 27395R の別名となります。在庫切れになった品目を参照するアプリケーションプログラムは、最初にすべての ITEM\_CODE 値から旧コードを探し出した後、マルチプルバリュースフィールドの 1 番目の ITEM\_CODE 値を参照することで置き換えを識別できます。

ITEM\_CODE フィールドには、ファイルの設定によって 1~65,534 の値が含まれます。しかし、ピリオディックグループとは異なり、値の 1 つが削除された場合、マルチプルバリュースフィールド中の個々の値は初期の位置を保持し続けません。例えば、上記の STOCK レコード中の品目 337015Y が在庫切れになって疑似コードが空値にセットされると、80819W が自動的に ITEM\_CODE の第 2 オカレンスになります。

マルチプルバリュースフィールドやピリオディックグループの使用法に関しては、次のような制限があります。

- マルチプルバリュース（MU）フィールドの値の最大数は 65,534 です。許可される実際のオカレンス数は、ファイルごとに明示的に設定する必要があります（デフォルトでは許可されていません）。この場合には、ADADBS MUPEX 機能または ADACMP COMPRESS MUPEX および MUPECOUNT パラメータを使用します。
- ピリオディックグループ（PE）のオカレンスの最大数は 65,534 です。許可される実際のオカレンス数は、ファイルごとに明示的に設定する必要があります（デフォルトでは許可されてい



ません)。この場合には、ADADBS MUPEX 機能または ADACMP COMPRESS MUPEX および MUPECOUNT パラメータを使用します。

- ピリオディックグループの中に別のピリオディックグループを含めることはできません。
- 1 オカレンスの圧縮後のサイズによっては、Adabas がサポートする最大レコード長より大きなレコード長になることもあります。

ピリオディックグループ内のディスクリプタや、ピリオディックグループ内のフィールドから作られるサブディスクリプタやスーパーディスクリプタは、論理順読み込みの制御や、検索コマンド、ソートコマンドのソートキーとしては使用できません。さらに、マルチプルバリュウフィールドやピリオディックグループ内のフィールドのディスクリプタをいくつか含む検索要求については、特別な規則があります。これらの規則については、『Adabas ユーティリティマニュアル』の ADACMP ユーティリティのセクションで説明しています。

## 1つの Adabas ファイルへの異なるレコードタイプの収容

ファイル数を減らすもう1つの方法に、同一 Adabas ファイルに2つの論理レコードタイプのデータを格納するという方法があります。次の例は、顧客ファイルと注文ファイルとを組み合わせる方法を示しています。この手法は、Adabas の空値省略機能の利点を活用しています。

\*       \*       \*

注文項目レコードのキーは、注文番号とこの注文内のシーケンス番号から構成されます。

この手法では顧客レコードと注文レコードタイプが各種のコントロールブロックと上位レベルのインデックスブロックを共用するので、入出力回数を減らすことができます。したがって、ファイルの処理を開始するために読み込むブロック数も減り、空きスペースが増えるため、バッファプールには別タイプのブロックを読み込むことができます。

顧客レコードと注文レコードは、データストレージにいっしょにグルーピングされるので、ある顧客に対するすべての注文を検索するとき読むブロックを減らすことができます。顧客と同時に全注文を追加するときも、必要な入出力回数が少なくなります。後で注文を追加する場合、このように最初からグルーピングされない可能性もありますが、ADAORD ユーティリティを使用して、後からグルーピングできます。

顧客データと注文データのどちらも効率良くアクセスできるように、キーの設計は注意しなければなりません。同じ顧客に属する各注文を区別できるように、顧客レコードのキーの接尾語は通常、空値です。例を次に示します。

A00231 000	A00231	A00231 001	1	A00231 002	2	A00231 003	3
A00232 000	A00232	A00232 001	1				

プログラムではキーの接尾語の内容により顧客レコードか注文レコードか区別できるのでレコードタイプフィールドは不要です。プログラムは付加フィールドを読むためにレコードを読み直すか、あるいはどちらのレコードタイプに関するフィールドすべて返す必要があります。

## 1つの論理ファイルへの物理ファイルのリンク

3 バイト ISN の Adabas ファイルには最大 16,777,215 件までのレコードを収容でき、4 バイト ISN の Adabas ファイルには 4,294,967,294 件までのレコードを収容できます。同じタイプのレコードが大量に存在する場合は、複数の物理ファイルに分散させる必要があります。

アクセスするファイル数を減らすために、同一フォーマットのレコードをもった複数の物理ファイルをリンクして、1つの論理ファイルに結合することができます。このようなファイル構造のことを拡張ファイルと呼び、この拡張ファイルを構成する物理ファイルのことをコンポーネントファイルと呼びます。拡張ファイルは、それぞれに異なった論理 ISN の範囲を持った 128 までのコンポーネントファイルによって構成されます。1つの拡張ファイルのレコード数は、4,294,967,294 までです。



**Note:** Adabas バージョン 6 では、より大きいファイルサイズ、より多くの Adabas 物理ファイルやデータベースがサポートされるので、拡張ファイルの必要性はほとんどありません。

アプリケーションプログラムから論理ファイルがアクセスされても（ファイルのアドレスは拡張ファイルの基本コンポーネントの番号、あるいはアンカーファイル）、Adabas では基準フィールドとして定義したフィールドのデータに従って正しいファイルが選択されます。このフィールドのデータは1つのコンポーネントファイル内でのみ、レコードをユニークに識別する特性を持ちます。アプリケーションが拡張ファイルを更新するとき、Adabas は更新するコンポーネントファイルを決断するために書き込むレコードの基準フィールドのデータを検索します。拡張ファイルデータを読み込むとき、Adabas は論理 ISN をキーとして使用し、正しいコンポーネントファイルを見つけます。

Adabas ユーティリティは常に拡張ファイルを認識している訳ではありません。つまり、一部のユーティリティオペレーションはすべてのコンポーネントファイルに対してユーティリティ機能を自動的に実行するが、他のユーティリティは各物理ファイルのみを認識します。詳細については「[拡張ファイル](#)」を参照してください。

## データの重複

- 物理的重複
- 論理的重複

### 物理的重複

ディテールレコードをアクセスするたびに、ヘッダーレコード内のいくつかのフィールドが、必要になる場合があります。例えば、請求書を作成するには、注文項目データと製品レコードの一部である製品の記述が必要です。この情報がプログラムで迅速に使用できるようにする最も簡単な方法は、注文項目データ内に製品記述情報のコピーを持つことです。すでに、他（この場合は製品レコード）で物理的に存在するデータを重複して持つので、これを "物理的重複" といいます。各ディテールレコードのフィールドをいくつかヘッダーレコード内にピリオディックグループとして格納すれば物理的重複は有効です。

物理的重複は、ほとんど更新されないフィールドに限定して使用すれば、滅多に問題にはなりません。上記の例では、製品記述情報はほとんど変更されません。重複フィールドに対するアクティビティが少ないほど、よい結果になります。

### 論理的重複

クレジットコントロールルーチンが顧客に送るすべての請求書の合計を必要とすると仮定します。この情報は関連する請求書を読み合計して求めるので、大量のレコードをランダムにアクセスすることになります。したがって、この情報が顧客レコード内に常に格納され、正しく更新されていれば、もっと迅速に得ることができます。これを、論理的重複といい、この情報そのものはすでに他に存在するのではなく別のレコードの内容に暗に含まれています。

物理的または論理的に重複している情報を更新するプログラムでは、重複しているデータの両方を更新しなければならないので処理は遅くなります。論理的重複では、たいていの場合1レコードの変更が他の何レコードかのデータに影響するので、更新も重複して行わねばなりません。論理的重複は、TP 環境では同一レコードを多くのユーザーが更新しなければならないので、効率が著しく低下します。

## Adabas レコード設計

Adabas ファイル構造が決まったら、通常、次にはファイルに対するフィールド定義テーブルを定義します。フィールド定義は、『Adabas ユーティリティマニュアル』で説明しているように、入力ステートメントとして ADACMP ユーティリティの COMPRESS 機能に入力します。このセクションでは、フィールドに対して使用できるオプションとパフォーマンスとの関係について述べます。

ファイルのフィールドは、読み込みや更新の頻度の高い順に先頭から並べる必要があります。こうすると、スキャンするフィールド数が減るので、データ転送に必要な CPU 時間が少なくて済みます。読む頻度が低いのが主に検索条件として使用するというフィールドは、後の方に置くべきです。

例えば、ディスクリプタフィールドがレコード構造の最初に配列されず、論理的に物理レコードの終了を超えた場合、パフォーマンスの理由から、そのレコードに対するインバーテッドリストエントリが生成されないことに注意してください。このケースでインバーテッドリストエントリを生成するには、ファイルのアンロード (SHORT モード)、圧縮解除、そして再ロードが必要です。あるいは、アプリケーションプログラムを使用してフィールドをファイルの各レコードの最初に再配列します。

### フィールドの組み合わせ

複数フィールドを常に一緒に読み更新するのなら、これらを1つの Adabas フィールドとして定義すれば CPU 時間を減らすことができます。このようにフィールドを組み合わせた場合の短所としては、次のものがあります。

- フィールドを組み合わせると圧縮の可能性が低くなるので、追加のディスクスペースが必要です。
- SQL のような問い合わせ言語プログラムで、このようなフィールドを扱うのは難しくなります。

### フィールドグループの使用

グループを使用すると、読み込みや更新コマンドの内部処理の効率が良くなります。つまり、Adabas コントロールブロックのフォーマットバッファが短くなるので、処理時間が短くなり、内部フォーマットバッファプールが少なくて済みます。

### 数値フィールド

数値フィールドは、最もよく使用するフォーマットを指定する必要があります。こうすれば、必要になるフォーマット変換が最小限になります。

### 固定ストレージオプション

固定ストレージ (FI) オプションを使用すると、フィールドの処理時間は減りますが、必要なディスクスペースが増え、特にピリオディックグループ内のフィールドのときは著しくなります。FI フィールドは、NU フィールドと同様に、できるだけまとめる必要があります。

# 15 データアクセス手法

---

▪ ディスクリプタの効率的使用 .....	70
▪ 照合ディスクリプタ .....	70
▪ スーパーディスクリプタ .....	71
▪ サブディスクリプタ .....	71
▪ フォネティックディスクリプタ .....	72
▪ ハイパーディスクリプタ .....	72
▪ ファイルカップリング .....	72
▪ ユーザー割り当て ISN .....	74
▪ ISN をディスクリプタとして使用 .....	74
▪ ADAM の使用方法 .....	74

このchapterでは、次のトピックについて説明します。

### ディスクリプタの効率的使用

---

ディスクリプタは、ユーザー指定の検索条件に依存してファイルからレコードを選択したり、論理順次読み込みを制御するために使用します。したがって、ディスクリプタの使用はファイルに対するアクセス方法に密接に関係します。各ディスクリプタについて、ディスクスペースも、処理オーバーヘッドも余計にかかります。更新が頻繁なときは特にそうです。ファイルに対して、どのタイプのディスクリプタをいくつ定義するかについては、次の点を考慮します。

- あるフィールドの処理を継続する前にデータを再び並び替える必要がある場合、照合ディスクリプタを定義することができます。
- ファイル内のレコード総数に対して少ない比率のレコードを選択するのに使用できるようなディスクリプタフィールドの値の配分にします。
- 既存のディスクリプタを使用して非常に少ない数のレコードを選択できる場合、検索条件をすっきりするためにディスクリプタを追加定義しないでください。
- 2つか3つのディスクリプタが頻繁に組み合わせて使用される（例えば地域、部、課）ときは、別々のディスクリプタで定義せず、スーパーディスクリプタとして定義することができます。
- あるディスクリプタについての選択条件がいつも値の範囲を含むときは、サブディスクリプタを使用することができます。
- あるディスクリプタについての選択条件が空値指定を含まず、このディスクリプタについて空値が多いときは、空値省略（NU）オプションを定義する必要があります。
- 更新の頻繁なフィールドは、ディスクリプタとして定義しないでください。
- 更新量が多い（大量の追加や削除）ファイルは、ディスクリプタ数を多くしないでください。

### 照合ディスクリプタ

---

照合ディスクリプタは、ディスクリプタフィールド値を、ユーザーが提供したアルゴリズムに基づいた特別なシーケンスでソート（照合）するために使用されます。英数字フィールドまたはワイド文字フィールドを照合ディスクリプタの親フィールドとして定義することができます。

特別な照合ディスクリプタユーザー出口は、ADARUNパラメータCDXnn（CDX01～CDX08）で指定されます。ユーザー出口は、照合ディスクリプタ値をエンコードするために、または本来のフィールド値にデコードするために使用されます。各照合ディスクリプタはユーザー出口に割り当てられなければならないが、単一のユーザー出口は複数の照合ディスクリプタを処理しません。

## スーパーディスクリプタ

スーパーディスクリプタは 20 個までのフィールド（またはフィールドの一部）の組み合わせによるディスクリプタです。スーパーディスクリプタを構成するフィールドは、ディスクリプタであってもなくてもかまいません。検索条件が値の組み合わせを含むとき、通常のディスクリプタを組み合わせるよりもスーパーディスクリプタを使用した方が効率が良くなります。これは、Adabas がインバーテッドリストを複数ではなく 1 つだけアクセスするだけで済み、また複数の ISN リストを AND で結合して最終的な該当レコードのリストを生成する必要がないからです。スーパーディスクリプタは、ファイルの論理順次読み込みの順を制御するのに通常のディスクリプタと同じように使用できます。

スーパーディスクリプタを含む検索条件に対する値は、スーパーディスクリプタのフォーマットと同じでなければなりません（構成フィールドがすべて数値のときバイナリ、そうでないときは英数字）。スーパーディスクリプタのフォーマットがバイナリのとき、Natural のような問い合わせまたはレポート機能を使用して選択条件値を入力するのは難しくなります。

スーパーディスクリプタの定義については、ADACMP ドキュメントの「*SUPDE*：スーパーディスクリプタの定義」を参照してください。

## サブディスクリプタ

フィールドの一部から作られるディスクリプタをサブディスクリプタといいます。サブディスクリプタの元のフィールドはディスクリプタであってもなくてもかまいません。検索条件が、英数字フィールドの先頭  $n$  バイトまたは数値フィールドの最後の  $n$  バイトを対象とするとき、そのフィールドの関連バイトだけからサブディスクリプタを定義できます。こうすると、検索条件では値の範囲の代わりに 1 つの値だけを指定すればよく、Adabas は中間の ISN リストを作りマージする必要がないので効率を上げることができます。

例えば、AREA が 8 バイトの英数字フィールドであり、先頭 3 バイトで地域、後 5 バイトで部門を表わすとします。地域が 111 のレコードだけが必要であるとき、サブディスクリプタを使用しない場合は AREA = 11100000 THRU 11199999 という検索条件が必要になります。AREA の先頭 3 バイトをサブディスクリプタと定義すれば、REGION = 111 という検索条件を指定することができます。

## フォネティックディスクリプタ

---

フォネティックディスクリプタはフォネティック検索を行うために定義します。FIND コマンドでフォネティックディスクリプタを利用すると、同じフォネティック値を含む全レコードが返ります。ディスクリプタのフォネティック値はフィールド値の先頭20バイトに基づくもので、英字の値のみが考慮されます（数値、特殊文字および空白は無視されます）。

## ハイパーディスクリプタ

---

ハイパーディスクリプタオプションを使用するとユーザーが提供するアルゴリズムに基づいてディスクリプタ値を生成できます。単一の物理 Adabas データベースごとに 31 個までの異なるハイパーディスクリプタを定義できます。使用するジョブの ADARUN ステートメントパラメータで適切な HEXnn を指定して各ハイパーディスクリプタに命名しなければなりません。

ハイパーディスクリプタは n 個から構成されるスーパーディスクリプタ、派生キー、またはその他のキー構造を実装するために使用できます。ハイパーディスクリプタの詳細は、ユーザーおよびハイパー出口に関するドキュメント、および『Adabas ユーティリティマニュアル』の「ADACMP ユーティリティ」の記述を参照してください。

## ファイルカップリング

---

ファイルカップリングにより、1つの FIND コマンドで指定の値を含むある別ファイルのレコードに関係する（カップリングされた）あるファイルのレコードを選択できます。例えば、CUSTOMER ファイルと ORDERS ファイルがあり、それぞれ顧客情報と注文情報が含まれているとします。各ファイルは共通のディスクリプタ顧客番号（CUSTOMER\_NUMBER）を持ち、これによりカップリングされています。

### 物理カップリング

この2ファイルは ADAINV ユーティリティによりカップリングされており、CUSTOMER ファイルのレコードと ORDERS ファイルのレコードとを互いに関連付ける（同一顧客番号を持つ）カップリングインデックスが、1組のインバーテッドリスト内に存在します。ファイルがカップリングされると、どちらのファイルのディスクリプタを使用しても1つの FIND コマンドで検索できます。例えば次のように指定します。



```
FIND CUSTOMER WITH NAME = JOHNSON  
AND COUPLED TO ORDERS  
WITH ORDER-MONTH = JANUARY
```

物理カップリングは、ファイルの更新が少ないか、カップリングリストのオーバーヘッドがかかっても、問い合わせの簡単さと比較して問題にならないければ有効です。また問い合わせの元になるファイルで、小さいファイルについても有効です。

ファイルの更新頻度が高いと、物理カップリングに対するオーバーヘッドがかなりの量になります。いずれかのファイルのレコードが追加／削除されたり、カップリングの元になるディスクリプタの更新が起きると、カップリングリストも更新しなければなりません。

物理カップリングは、カップリングインデックス用のディスクスペースも必要です。必要な容量はカップリングされるレコード数に依存します。カップリングディスクリプタが一方のファイルのユニークキーになっていると最良です。つまり、別ファイルのある1レコードにカップリングされるあるファイルのレコードが少ないことを示します。最悪のケースはファイル間に多対多の関係が存在する場合です。このとき、両ファイルとも、レコードにカップリングされるレコード数が大きくなってしまいます。

カップリングの基本として使用するディスクリプタは、通常、空値省略オプションとして定義する必要があります。こうすればこのディスクリプタが空値のレコードはカップリングインデックスに含まれません。

その他カップリング使用については、『Adabas ユーティリティマニュアル』「ADAINV ユーティリティ」の説明を参照してください。

## 論理カップリング

複数ファイルの問い合わせは、ファイル間の結合に使用するフィールドを検索条件に指定することもできます。この特徴は論理カップリングと呼ばれ、ファイルは物理的にカップリングされている必要はありません。

この手法には読み込みコマンドが必要ですが、物理カップリングリストが不要なため、カップリングディスクリプタに変更の多い場合には効果があります。ユーザープログラムには、ソフトカップリング結合に使用するフィールドと、検索条件だけを指定すればよいというのも重要です。Adabas がすべての必要な検索、読み込み、内部リストのマッチング操作を行います。

### ユーザー割り当て ISN

---

ファイル内の各レコードの ISN は、通常、Adabas が自動的に割り当てますが、オプションとしてユーザーが割り当てすることもできます。このことにより、インバーテッドリストを使用せず直接 ISN を使用して検索することが可能となります。ただし、ユーザー自身が ISN 割り当てのアルゴリズムを開発しなければなりません。結果 ISN は、ファイルのロード時に ADALOD ユーティリティで指定された MINISN および MAXISN パラメータ値の範囲内にする必要があります。

### ISN をディスクリプタとして使用

---

インバーテッドリストを使用せずに、直接関係レコードを読むために、レコード内に関係レコードの ISN を格納してもかまいません。

例えば、注文レコードを 1 つ読み、この注文に対する全顧客レコードを選択して読み込むアプリケーションを考えます。顧客レコードの ISN (1 注文レコードに複数の顧客があるとき、マルチプルバリューフィールドを使用できます) を注文レコードに格納すると、注文レコード内の ISN を使用して関連顧客レコードを直接読むことができます。

こうすればその注文に関する顧客レコードを探すのに顧客ファイルに FIND コマンドを発行する必要がなくなります。この手法では顧客レコードの ISN を含むフィールドを注文レコード内に保持する必要があり、顧客ファイルの ISN はファイルのアンロード/リロードで変えられることはないと仮定しています。

### ADAM の使用方法

---

Adabas ダイレクトアクセスメソッド (ADAM) を使用すると、インバーテッドリストをアクセスせずにデータストレージから直接レコードを検索できます。レコードが格納されているデータストレージブロック番号は、レコードの ADAM キーに基づきランダムマイジングアルゴリズムを使用して計算されます。ADAM は、アプリケーションプログラム、クエリおよびレポートライタの機能に対して完全に透過的に使用されます。

ADAM キーの値はユニークでなければなりません。レコードの ISN を ADAM キーとして使用してもかまいません。

ADAM ファイルのアクセス処理はかなり速いのですが、ADAM ファイルのロードや ADAM ファイルへの新レコードの追加は、一般に一連の新レコードが同一ブロックに格納されることはないので、標準ファイルよりも時間がかかります。

ADAM ファイルはランダムにも論理順にも処理されるなら、ADALOD ユーティリティのビット切り捨て機能を使用して論理順次処理を最適化できます。この機能を使用すると各 ADAM キーの右側を指定ビット数だけ切り捨てて、ランダムマイジングアルゴリズムへの入力にすることができます。したがって、キーの値の左側が似ているレコードは、同じデータストレージブロックに格納されます。

切り捨てるビットが多過ぎるとオーバーフローレコード数が増し、ランダムアクセスのパフォーマンスが低下するので注意する必要があります。この原因は、ADAM キーを使用して配置されたブロックに格納できないオーバーフローレコードが、標準インバーテッドリスト処理で他のブロックに格納されるからです。オーバーフローレコードもインバーテッドリストを使用して配置しなければなりません。他にオーバーフローを最小にする方法はファイルとパディングファクタサイズを相対的に大きく指定することです。

インバーテッドリストを使用した検索では3~4回のI/Oが必要ですが、ADAM では一般的に平均1.2~1.5回のI/Oで済みます。この平均には、アソシエータの制御下でファイルの他のブロックに格納されたオーバーフローレコードの平均も含まれています。オーバーフローレコードについては、通常のインバーテッドリスト経由の検索を行います。

したがって、ADAM ファイルのパフォーマンスは使用可能ディスクスペース（スペースが多いほど、インバーテッドリスト付きで配置する必要のあるオーバーフローレコードは少なくなります）と ADAM キーから切り捨てるビット数、レコードの追加／更新処理の量により決まります。ディスクスペースと切り捨てビット数との各種組み合わせに対する平均入出力回数は、ADAMER ユーティリティを使用して調べることができます。詳細は、『Adabas ユーティリティマニュアル』を参照してください。



# 16 ディスクスペースの使用方法

---

■ データ圧縮 .....	78
■ フォワードインデックス圧縮 .....	80
■ パディングファクタ .....	81

データベース環境においては、次の理由からディスクスペースの効率的な使用が重要です。

- 複数のユーザーの間で同時に、また異なる組み合わせでデータを共有するには、組織のデータの大部分をオンラインで格納する必要があります。
- いくつかのアプリケーションでは、非常に大容量のデータを保持しています。

ディスクスペースの効率的な使用方法については、システムの他の目標（パフォーマンス、柔軟性、使い易さ）を考慮しながら決定を下さなければなりません。このセクションでは、これら要素とディスクスペースの効果的使用法との兼ね合いに関する考察と手法について説明します。

このchapterでは、次のトピックについて説明します。

## データ圧縮

各フィールドは次の3つの圧縮に関する指定のうち1つを定義できます。

- 固定ストレージ (FI)。フィールドの圧縮を行いません。必ず値を持つ1バイトのフィールド（例えば社員レコードの性別）、およびフィールド全体で値を持つ英数字/数字フィールド（例えば社員番号）は、常に固定 (FI) フィールドにする必要があります。
- 通常の圧縮 (デフォルト)。Adabas は英数字フィールドの末尾の空白と数値フィールドの先頭のゼロを除きます。
- 空値省略 (NU)。通常の圧縮に加えて、フィールドが空値のとき詰めます。連続している複数の空値フィールドは、単一の値に結合されます。

次の表に5バイトの英数字フィールドに各圧縮オプションを使用したときの格納法を示します。

フィールド値	固定ストレージ	通常の圧縮	空値省略
ABCbb	ABCbb (5 バイト)	4ABC (4 バイト)	4ABC (4 バイト)
ABCDb	ABCDb (5 バイト)	5ABCD (5 バイト)	5ABCD (5 バイト)
ABCDE	ABCDE (5 バイト)	6ABCDE (6 バイト)	6ABCDE (6 バイト)
bbbbb	bbbbb (5 バイト)	2b (2 バイト)	* (1 バイト)
X	X (1 バイト)	2X (2 バイト)	2X (2 バイト)

各格納値の前の数字は、長さバイトを含むフィールドのバイト長を示します (FI フィールドにはありません)。空値省略に対する \* は、省略されたフィールド数を示すもので、レコード内のこの位置で現在、空値の (省略された) フィールド数を示す 1 バイトのフィールドです。このフィールドは最大 63 個の省略フィールドを示します。

選択する圧縮オプションは、フィールドのインバーテッドリストの作成 (ディスクリプタの場合)、またフィールドの圧縮と圧縮解除に必要な時間にも影響します。

## 固定ストレージ

固定ストレージと定義するとフィールドに対する圧縮は行われません。長さバイトも付加されず、フィールドの標準長で格納されます。固定ストレージは、ほとんど空値にはならない1~2バイトの小さいフィールドや、圧縮がまったく、またはほとんど不可能であるフィールドに指定します。FIフィールドの使用に関する制限等については、『Adabas ユーティリティマニュアル』のADACMP ユーティリティを参照してください。

## 通常の圧縮

通常の圧縮では英数字フィールドの末尾の空白や数値フィールドの先頭のゼロを除きます。少くとも2バイトの末尾の空白や先頭のゼロが除かれれば、ディスクスペースを節約できます。しかし、2バイトから成るフィールドに対するスペースの節約は起こりません。また、1バイトのフィールドに関しては、長さバイトの追加で必要スペースが事実上2倍になります。これらのフィールドおよび先頭や末尾のゼロと空白がほとんど存在しないフィールドは、固定ストレージ (FI) オプションで定義して圧縮を避けるべきです。

## 空値省略

フィールドに空値省略 (NU) を指定し、フィールド値が空値であると、長さバイトと圧縮された空値の代わりに1バイトの空値フィールドインジケータが格納されます (「[データ圧縮](#)」を参照)。この空値フィールドインジケータは、レコード内のその位置での空値省略されたフィールド (連続フィールド) の数を示します。したがって、よく空になるフィールドをレコード内で隣り合わせにし、それぞれに空値省略オプションを指定すると有利です。

ディスクリプタとしても定義されているNUフィールドについて、空値省略指定するとインバーテッドリストに空値は作られません。このため、検索コマンドでそのディスクリプタを検索しようとしても空値が含まれる該当ディスクリプタレコードは認識されません。

空値省略で定義したフィールドから作られるサブディスクリプタやスーパーディスクリプタについても同様です。つまり、元のフィールドの該当バイトが空値で、フィールドに空値省略 (NU) オプションが定義されていると、サブディスクリプタに対するエントリは作られません。スーパーディスクリプタの派生元のフィールドが、空値を含むNUフィールドの場合、このスーパーディスクリプタのエントリは作成されません。

したがって、ディスクリプタで空値を検索する必要がある場合や、空値を含むレコードをディスクリプタ順で読み込む必要がある場合、例えば論理順の読み込みまたはソートを制御する場合、ディスクリプタフィールドにNUオプションを指定しないでください。

空値省略をマルチプルバリューフィールドやピリオディックグループ内のフィールドに指定すると有効であり、必要なディスクスペースや必要な内部処理を減らすことができます。これらフィールドの更新は、使用する圧縮オプションによりさまざまです。

空値省略オプションを定義したマルチプルバリューフィールドが空値に更新されると、右側の値はすべて左にシフトされ、値カウントも減らされます。ピリオディックグループ内のフィールドがすべて空値省略と定義され、グループ全体が空値に更新されると、更新されたオカレンスが

最終のオカレンスの場合だけオカレンスカウントが減らされます。マルチプルバリューフィールドやピリオディックグループに関する詳細は、『Adabas ユーティリティマニュアル』の ADACMP ユーティリティ、および『Adabas コマンドリファレンスマニュアル』の A1/A4、N1/N2 コマンドを参照してください。

## フォワードインデックス圧縮

フォワード（またはフロント、プリフィクス）インデックス圧縮機能は、インデックス値から余分な接頭辞情報を削除してインデックススペースを節約します。これにより、ディスクスペースの使用量が減り、より少ないインデックスレベルが使用される可能性があり、インデックス I/O 操作が減ることから、全体の情報処理量が向上します。同量のインデックス情報がより少ないスペースを占めるので、バッファプールがより効率的になります。L3、L9、または S2 のようなコマンド（インデックスをシーケンシャルに横断する）はもっと速くなり、小さくなったインデックスサイズは、インデックスを読み込みまたは修正する Adabas ユーティリティの経過時間を減少します。

1つのインデックスブロック内で、最初の値は完全な長さで保存されます。後続のすべての値については、前者と共通の接頭辞が圧縮されます。インデックス値は、次のように表現されます。

```
<l,p,value>
```

ここでは次の内容を表しています。

p 前の値の接頭辞と等しいバイト数です。

l p バイトを含む残りの値の排他的な長さです。

例

圧縮解除	圧縮
ABCDE	6 0 ABCDE
ABCDEF	2 5 F
ABCGGG	4 3 GGG
ABCGGH	2 5 H

インデックス圧縮は、ディスクリプタのフォーマットによって影響されません。これは、PE オプションやマルチクライアントディスクリプタにも機能します。



インデックス値の圧縮が可能な限り大きくなるのは、次のようにピリオディックグループが英数字の場合です。

```
253 bytes for the proper value if no bytes are compressed 1 byte for the PE index  
1 byte for the p-byte.
```

よって、排他的な長さの合計はシングルバイトに保存することができます。

Adabasは、フォワードインデックス圧縮をファイルレベルで実装します。ファイルをロードする際（ADALOD）、そのファイルのインデックス値を圧縮するかどうかのオプションが提供されます。オプションは、ファイルを再び並べ替えることで変更することができます（ADAORD）。

Adabasは、データベースの全インデックス値をまとめて圧縮するオプションも提供します。この場合、特定のファイルを別々に設定することができます。ファイルレベルの設定は、データベースの設定をオーバーライドします。

インデックス値を圧縮するかどうかの判断は、次に示すようにインデックス値の類似点とファイルサイズに基づきます。

- インデックス値が類似するほど、圧縮の結果が良くなります。
- 小さいファイルは節約されるスペースの絶対量が小さくなるので、圧縮には向いていませんが、大きいファイルはインデックス圧縮に向いています。

ファイルのインデックス値の圧縮率がよくない場合がありますが、それでもインデックスが圧縮されていれば、圧縮されていない場合よりも必要になるインデックスブロックが少なくなります。

## パディングファクタ

大量のレコード更新が発生する場合、大規模な量のレコードが移動、すなわち、現在のブロックから削除し、拡張レコード用のスペースがある別のブロックへ移動することになります。ファイルのロード時、このファイルに対するデータストレージのパディングファクタを大きくとれば、移動するレコードの量を大幅に減らすことができます。パディングファクタとは、レコード拡張用に確保しておく各物理ブロック内の比率を示します。

ファイルロード時やファイルに新レコードを追加したとき、パディングエリアは使われません（ADAMファイルには適用されません。ADAMファイルでは計算で求められたデータストレージブロックにレコードを格納するのにパディングファクタが必要な場合に使用します）。レコードの拡張される比率が小さいとき、レコード拡張が起こらないブロックのパディングエリアは無駄になってしまうので、パディングファクタを大きくとる必要はありません。

大量のレコード更新／追加があり、1つまたは複数のディスクリプタの現在の値の範囲内に大量の新しい値が挿入される場合、アソシエータ内でも大規模な量の移動が起こります。これはアソシエータのパディングファクタを大きくとれば減らすことができます。

パディングファクタを大きくとったときの欠点は、必要なディスクスペースが大きくなり（ブロック当りのレコードやエントリが小さくなります）、順次処理について読まれる物理ブロックが多くなるので効率が低下します。

ファイルのロード時、パディングファクタを指定します。ファイルまたはデータベースに対して ADADBS MODFCB 機能または ADAORD ユーティリティを実行するときに、パディングファクタを変えることができます。

# 17 Adabas Security

---

■ セキュリティ計画 .....	84
■ パスワードセキュリティ .....	84
■ セキュリティバイバリュー .....	86
■ 暗号化 .....	86
■ Adabas SAF Security .....	86
■ Natural Security と Adabas Online System Security .....	87

このセクションでは、データベースのセキュリティに関する一般的な概念と、Adabas と Adabas サブシステムのセキュリティ機能について説明します。このセクションで説明している機能の詳細については、このドキュメントの他の部分、また Adabas Security のドキュメントを参照してください。

このchapterでは、次のトピックについて説明します。

## セキュリティ計画

---

セキュリティを効果的に行うには、次の点を考慮しなければなりません。

- システムの安全性はシステムの最も弱い要素で決まります。この要素はシステムのデータ処理部門以外の場合も考えられます。例えば、印刷されたリストを正しく保護しなかったなどです。
- 簡単に扱えるシステムを設計するのはほとんどの場合、不可能です。この機能保護から得られる利益がコストを超えと思われる場合は、このシステム設計が妥当であると考えられません。
- セキュリティのコストが高くなることがあります。コストにはセキュリティ手法のプランを調整し、その効果を監視するものに必要なオーバーヘッド、マシンリソースおよび時間を含みます。

セキュリティのためのコストは、セキュリティ違反の危険やコストを計るよりも容易ですが、あるセキュリティの方法では、セキュリティの分野以外でも利益をもたらすことがあるという事実も汲んでコストを求める場合もあります。セキュリティ違反のコストはその違反の種類によります。コストとしては、次のものを含みます。

- 違反を修復するためのロス時間
- 個人的な取り決め、契約違反等に対する罰則
- 顧客、供給者、雇用者等との関係に対する損害

## パスワードセキュリティ

---

パスワードセキュリティにより、DBA は次のようにしてユーザーによるデータベースの使用を制御できます。

- ユーザーの使用ファイルを制限します。
- 各ファイルについて、ユーザーのアクセス、更新が可能かどうか、アクセスのみ可能なのかどうかを指定します。
- あるフィールドのアクセスや更新を禁じ、同一ファイルの他のフィールドのアクセスや更新を可とします。

- ファイルに対するユーザービューを、指定するフィールド値（例えば部門コード）を含むレコードに制限します。

DBA は各ファイルやファイル内の各フィールドにセキュリティレベルを割り当てることができます。次の表で、x/y はアクセス/更新のセキュリティレベルを示します。値 0/0 は保護なしを示します。

ファイル	フィールド	
1 (2/3)	AA (0/0)	BB (4/5)
2 (6/7)	LL (6/7)	MM (6/9)
3 (4/5)	XX (4/5)	YY (4/5)
4 (0/0)	FF (0/0)	GG (0/15)

ユーザーがセキュリティで保護されたファイルのアクセス/更新を行うには、適切なパスワードを指定する必要があります。次の表で、x/y はパスワードのアクセス/更新の権限レベルを示します。

	パスワード	
	ALPHA	BETA
ファイル 1	2/3	4/5
ファイル 2	0/0	6/7
ファイル 3	4/5	0/0

ファイル、フィールド、パスワードを上の方の表のとおりとすると、次のことがいえます。

#### ■ パスワード ALPHA

- ファイル 1 のフィールド AA はアクセス/更新できますが、フィールド BB はできません。
- ファイル 3 の全フィールドはアクセス/更新できます。
- ファイル 2 はアクセス/更新ともできません。

#### ■ パスワード BETA

- ファイル 1 の全フィールドはアクセス/更新できます。
- ファイル 2 の全フィールドはアクセスでき、フィールド LL は更新できますが、フィールド MM は更新できません。
- ファイル 3 はアクセス、更新ともできません。
- ファイル 4 のどのフィールドのアクセスも、フィールド FF の更新にも、パスワードは不要です。
- ファイル 4 のフィールド GG は読み取り専用です。GG の更新セキュリティレベルは 15 であり、可能な最高権限レベルは 14 です。

パスワード ALPHA がアクセスできないフィールドをパスワード BETA でアクセスできるなら、パスワード ALPHA がアクセスできる同じファイルのフィールドはすべて、パスワード BETA でもアクセスできます。ALPHA がフィールド AA のアクセス権を持つがフィールド BB のアクセス権はなく、パスワード BETA は BB はアクセスできるが AA ではできないということはありません。更新についても同様の制限があります云えます（必ずしも同じフィールドの組み合わせや同じパスワードの優劣とは限りません）。BETA が更新できる全フィールドと BETA が更新できない他のフィールドを、ALPHA では更新できます。

この制限はファイルレベルのセキュリティには適用されません。例えば、ALPHA はファイル 3 は使用できるがファイル 2 は使用できないし、BETA はファイル 2 は使用できるが、ファイル 3 はできません。ファイルにレコードを追加するとき、Adabas はユーザーが値を指定したフィールドについての更新セキュリティレベルだけをチェックします。例えば、パスワード ALPHA は、フィールド BB の値が指定されなければファイル 1 にレコードを追加するのに使用できます。例えば、顧客レコードをゼロ値で作成するような状態です。レコード削除については、指定パスワードが該当ファイルの最高の更新セキュリティレベル以上でなければなりません。例えば、ファイル 2 からレコードを削除するには更新権限レベルの 9 が必要ですが、ファイル 4 のレコードはそのレベルでは削除できません。

## セキュリティバイバリュー

---

ファイル内のフィールド値で、そのファイルのフィールドのアクセス／更新を制限することも可能です。詳しくは『Adabas Security Manual』を参照してください。

## 暗号化

---

ファイルの初期ロード時、またはファイルにレコードを追加するとき、Adabas では暗号化して格納できます。暗号化機能を使用すると、データベースが格納されているディスクを物理的にダンプしても、その内容を解読することは非常に難しくなります。暗号化は、データストレージに格納されるレコードだけに適用されます。アソシエータには暗号化は行えません。

## Adabas SAF Security

---

Adabas のアドオン製品である Adabas SAF Security は、Software AG の Complete および Software AG 以外の次のセキュリティ環境で使用できます。

- CA-ACF2 (Computer Associates)
- CA-Top Secret (Computer Associates)
- RACF (IBM Corporation)

Adabas SAF Security の詳細については、Software AG 営業部門にお問い合わせください。

## Natural Security と Adabas Online System Security

---

Natural Security システムを利用することで、Adabas/Natural ユーザーに対し広範なセキュリティ機能が提供できます。詳しくは『NATURAL SECURITY マニュアル』を参照してください。

DBA 機能の Adabas Online System (AOS) の使用も制限できます。AOS Security は前提条件として Natural Security を必要とします。





# 18 リカバリ／再スタート設計

---

■ Adabas リカバリ機能 .....	90
■ リカバリの計画とインストール .....	91
■ 要求と機能のマッチング .....	91
■ トランザクションリカバリ .....	92
■ エンドトランザクション (ET) コマンド .....	92
■ クローズ (CL) コマンド .....	92
■ ET データの読み込み .....	130
■ システムおよびトランザクション障害 .....	93
■ Adabas トランザクションリカバリの制限 .....	93
■ Adabas チェックポイントコマンド .....	93
■ 排他ファイル制御 .....	94
■ ユーザー再スタートデータ .....	95

このchapterでは、データベースのリカバリと再スタートの設計概念について説明します。リカバリと再スタートの設計を適切に行うことはシステム設計の重要な部分です。特にデータベース環境では重要です。Adabasはリカバリおよび再スタートの両機能を備えています。これらの機能は別々に考慮しなければなりません。

このchapterでは、次のトピックについて説明します。

### Adabas リカバリ機能

---

データベースのリカバリは、最優先されなければなりません。データベーストランザクションが失敗した場合またはトランザクションをキャンセルしなければならない場合は、トランザクションの実効性がなくなるように、データベースをトランザクション開始以前の状態に復元しなければなりません。

標準のAdabasシステムは、データベースの保全性を保証するために、トランザクションロジック（ET ロジックと呼ぶ）、拡張チェックポイント／ロギング機能、およびトランザクションを戻すバックアウト処理を備えています。

システム障害後のデータベースの再スタートは、障害が発生する以前にセーブされていたレベルから障害が発生したステップまでの再構築を意味します。また、可能であれば中断したオペレーションを正常に終了し、通常のデータベースオペレーションを継続させることです。Adabasには、データベースをリカバリするためのリカバリジョブストリームを再構築するリカバリエイドがあります。

リカバリ機能は、当然の機能として特に明示されないことがよくあります。つまり、何が起ころしてもシステムがリカバリを行い再スタートできると皆が仮定しています。しかし、システムのいろいろなユーザーが必要とするリカバリレベルについて決定すべき事柄が存在します。リカバリ機能は、DBAがイニシアティブをとり、必要な事柄を設定すべき分野です。まず、システムの各潜在ユーザーにリカバリ／再スタート要求について質問する必要があります。考慮する必要がある最も重要な点としては次のものがあります。

- システムなしに、どの位の期間ユーザーが管理できるか。
- どの位長く、各フェーズの遅延が許容されるか。
- どんな手作業で、ユーザーが入出力チェックを行い、どの位の時間が必要か。
- リカバリ／再スタートが発生した場合、データ保全性が維持されていることを確認するためにどのような手順を行う必要があるか。

## リカバリの計画とインストール

一旦、リカバリ／再スタートが必要だと認められたら、DBA はこれら要求を満たすのに必要な方法の計画に進むことができます。ここで述べる手法は上記を行うための基本的な手引として使えるはずで

1. システム内のさまざまなユーザーがデータを共有するレベルや程度について決定します。
2. システムに対するリカバリパラメータを設定します。このパラメータとしては、予測／実際のブレークダウン率、平均の遅れと影響を受ける項目、およびセキュリティや監査に使用する項目を含みます。
3. システム内に監査処理を含む場合は、これに関する決定を含みます。
4. リカバリ設計の要点をまとめたアウトラインを準備します。このアウトライン内には次の情報を含むようにします。
  - 整合性チェックに関する計画。データの整合性チェックは、できるだけシステムへのデータ入力時に近い時点で行う必要があります。レコード入力時と別にデータを中間で更新すると、リカバリはより難しくコスト高になります。
  - データベースや選択ファイルのダンプ（バックアップコピー）。
  - ユーザーチェックポイントと Adabas チェックポイント。
  - ET ロジック、排他ファイル制御、ET データの使用。
  - 監査手順。
5. リカバリ／再スタートに必要なすべてのリソースが必要なときに使用できるかどうか判断できるようオペレーション担当者とも相談する必要があります。
6. 最終的なリカバリ設計はドキュメント化し、ユーザー、オペレーション担当者、その他システムに関連する人々に知らせ、再確認しなければなりません。

## 要求と機能のマッチング

大体のリカバリ要件が決まったら、次にリカバリ／再スタートを行うための関連機能（Adabas 機能も Adabas 以外の機能も）を選択します。以下に、リカバリ／再スタートに関連する Adabas 機能について説明します。

## トランザクションリカバリ

---

大部分のオンライン更新システムや多くのバッチ更新プログラムでは、次のような特性の入力トランザクションストリームを処理します。

- トランザクションにより、プログラムが検索、追加、更新、削除を行うレコードの数は少数です。例えば、注文入力プログラムでは、各注文に対して顧客レコードと製品レコードを検索し、データベースにその注文と注文項目データを追加し、製品レコードの注文フィールドの量を更新します。
- プログラムは、トランザクション開始から終了までに使用するレコードの排他制御が必要ですが、トランザクションが完了したら、レコードを解放し、他ユーザーが更新したり削除したりできるようにします。
- トランザクションが未完了の状態になってはなりません。すなわち、2レコードを更新する場合は、2レコードを両方とも更新するか、いずれも更新されないかのどちらかでなければなりません。

## エンドトランザクション (ET) コマンド

---

Adabas ET コマンドを使用すると、次のことが可能です。

- 完了したトランザクションによる追加、更新、削除処理がすべてデータベースに適用されるようになります。
- 全体的または部分的なシステム障害により中断されたトランザクションについては、すべての更新処理をデータベースから取り除かれます。
- プログラムで Adabas システムファイルに 2000 バイト以下のユーザー定義再スタートデータ (ET データ) を格納できます。この ET データは、Adabas の OP または RE コマンドで再スタート時に取り出すことができます。このため、プログラムや TP 端末ユーザーはどこから再開するかを判断できます。
- トランザクション処理中にホールド状態にあった全レコードを解放します。

## クローズ (CL) コマンド

---

Adabas CL コマンドでユーザーの最新 ET データを更新できます (例えばユーザー定義のジョブ完了フラグの設定)。詳細については、「[ユーザー再スタートデータ](#)」を参照してください。

## ET データの読み込み

AdabasのOPコマンドを使用して、ユーザー再スタート後、または新規ユーザー／Adabasセッション開始時、ETデータを読むことができます。OPコマンドには、ユーザーIDが必要です。このユーザーIDを基に、AdabasはユーザーETデータを格納します。また、ETデータを読み込むためのコマンドオプションも必要です。

現在または指定のユーザーのETデータの読み込み、例えばオンライン更新操作の監視時にはREコマンドも使用できます。

## システムおよびトランザクション障害

自動バックアウトルーチンは、すべてのAdabasセッションの開始時に起動されます。セッションが異常終了した場合、オートバックルーチンによって、中断されたすべてのトランザクションによる処理が、最新のETまでデータベースから削除されます。個々のトランザクションが中断された場合は、トランザクションが行った更新をすべてデータベースから自動的に除きます。また、アプリケーションプログラムでもAdabasのBTコマンドを使用して現在のトランザクションの処理をバックアウトするように指定できます。

## Adabas トランザクションリカバリの制限

トランザクションリカバリ機能はデータベースの内容を回復するだけです。TPメッセージシーケンスの回復、非Adabasファイルの再ポジショニング、あるいはユーザープログラムステータスの退避は行いません。

特定ユーザーのトランザクションによる処理だけをバックアウトすることはできません。このユーザーが追加または更新したレコードに対して別ユーザーがトランザクションを実行した可能性があるからです。

## Adabas チェックポイントコマンド

次の理由から、ある種のプログラムではETコマンドの使用は効果的ではありません。

- 各トランザクションの実行中、プログラムが多数のレコードをホールドしなければならない場合。このことは、デッドロックの可能性を増加させ（Adabasでは2ユーザーの一方のトランザクションをバックアウトすることでデッドロックを自動的に解消しますが、大量のトランザクションを再処理しなければならない）、Adabasホールドキューをかなり大きくする必要がります。

- 複合検索により見つかるレコードが多く長いリストをプログラムで処理する場合、このリストの途中から再スタートするのは困難です。

上記のようなプログラムでは、必要に応じて Adabas チェックポイントコマンド (C1) を使用して、プログラムが更新中である 1 つ以上のファイルのリストア可能地点を設定できます。

## 排他ファイル制御

---

ユーザーは 1 つ以上の Adabas ファイルの排他更新制御を要求できます。OP コマンドで排他制御を要求し、他ユーザーがファイルを現在更新中でなければ、排他制御を行えます。あるユーザーに排他制御が与えられると、他ユーザーはそのファイルを読み込むことはできますが、更新することはできません。論理順処理か、あるいは検索の結果、一連の多数のレコードを読んだり更新したりするプログラムでは、他ユーザーからの更新を防ぐために排他制御を使用できます。こうすれば、各レコードをホールド状態にする必要がなくなります。

### 排他制御ファイルのチェックポイント

排他制御ユーザーは ET コマンドを使用しても使用しなくてもかまいません。ET コマンドを使用しないときは、C1 コマンドを発行し、チェックポイントを取得することができます。

### システムおよびプログラム障害

システムまたはプログラム障害の場合、排他制御下で更新中のファイル (群) は、ADARES ユーティリティの BACKOUT 機能を使用してリストアできます。このユーティリティは自動的に呼ばれるのではなく、入力として Adabas データプロテクションログが必要です。ユーザーが ET コマンドを使用していれば、この処理は不要です (「[トランザクションリカバリ](#)」参照)。

### 排他ファイル制御の制限

排他ファイル制御には、次のような制限があります。

- 最終チェックポイントへのリカバリは自動的には行われず、障害が起きたときのデータプロテクションログがリカバリ処理に必要です。ただし、ユーザーが ET コマンドを発行したときは、これは適用されません。
- システム障害後の再スタート状態で、Adabas は、システム中断時排他制御の下で更新されていたファイルを他ユーザーが更新するのをチェックせず、また回避もしません。

## ユーザー再スタートデータ

Adabas の ET および CL コマンドでは、オプションとして Adabas システムファイルに 2000 バイトまでのユーザーデータを格納できます。ユーザーごとにユーザーデータを 1 レコード保持します。ユーザーが新規にユーザーデータを書き読むたびに、このレコードは書き換えられます。OP コマンドでユーザー ID を指定したときだけ、セッション間でデータが保持されます。

ユーザーデータの第 1 の目的は、プログラムが自身で再スタートが可能となり、リカバリ手順が適切に行われたかをチェックできることです。ユーザーデータとして有効な情報タイプは、次のものです。

- プログラムの実行日時および最終更新時刻。これにより、プログラムは端末ユーザー、コンソールオペレータまたはプリンタに適切なメッセージを送ることができ、ユーザーやオペレータはリカバリ／再スタート処理が正しく行われたかをチェックできます。特に、知らずに夜中に重大な障害が起きた場合、端末ユーザーはどの業務から再実行すべきかがわかります。
- 入力データ収集日
- バッチ番号。端末から再入力しなければならない業務が何であるかを管理者は把握できます。
- 識別データ。このデータを使用して、再スタート位置をプログラムで判別できます。例えば、論理順スキャンを行うプログラムでは再開するキー値が必要です。
- トランザクション番号／入力レコード位置。これにより、端末ユーザーやバッチプログラムが開始位置を決める労力を最小にできます。各トランザクションに Adabas はトランザクションシーケンス番号を返すが、次の理由からユーザーもシーケンス番号を保持したい場合があります。
  - 再スタート後、Adabas シーケンス番号がリセットされる時。
  - 各トランザクションの処理内容が複雑で、Adabas トランザクションシーケンス番号と次の入力レコードやドキュメントの位置との関係が単純ではないとき。
  - 入力エラーによりプログラムがトランザクションをバックアウトするとき。トランザクションを直ちに再入力するか（単純なキーイングミスなど）、または後で修正するために拒否するか（入力ドキュメントやレコードの基本的なエラーが発生した場合）を Adabas は判断できません。
- その他の記述データや中間データ。例えば、繰越しトータル、レポートのページ番号や見出し、実行統計など。
- ジョブ／バッチ完了フラグ。すべての処理が完了後でオペレータやユーザーに知らされる前にシステムがダウンする場合があります。この場合、オペレータがプログラムを再スタートしても、プログラムでは入力の終わりまで実行せずに、このフラグをチェックできます。同様なことが端末から入力したドキュメントのバッチに対しても適用されます。
- 最終のジョブ／プログラム名。いくつかのプログラムが一定の順序でデータベースを更新しなければならないとき、同一のユーザー ID を使用し、順序が正しく保たれているかをチェックするためにユーザーデータを使用できます。

OP コマンドまたは RE コマンドを使用して自分のユーザーデータを読み込むことができます。別ユーザーのユーザーデータは、RE コマンドを使用し、別ユーザーの ID を指定して読み込むことができます。全ユーザーのユーザーデータは、コマンドオプション付きで RE コマンドを使用すれば論理順に読み取ることができます。この場合にはユーザー ID を指定してはなりません。



# 19 Adabas Recovery Aid

---

■ リカバリログ (RLOG) .....	98
■ Adabas Recovery Aid の起動 .....	99

システム障害によってデータベース処理が中断した場合、障害時点の状態にデータベースを再構築するジョブストリームを Adabas Recovery Aid で作成できます。

Adabas Recovery Aid では、プロテクションログ (PLOG) と前の ADASAV 処理から得られたデータベースステータスを、独自のリカバリログ (RLOG) の情報と組み合わせて、ジョブシーケンスが再構築されます。結果的にジョブステートメント文字列 (リカバリジョブストリーム) が再構築され、特別な名前での出力データセットに置かれます。

Adabas Recovery Aid にはリカバリログ (RLOG) とリカバリエイドユーティリティ ADARAI の2つの主要要素があります。RLOG は、ADAFRM を使用して Adabas の他のファイルと同じようにフォーマットし、ADARAI ユーティリティで定義します。

DBA は、リカバリエイドユーティリティ ADARAI を実行して次の処理を行います。

- RLOG の定義と Adabas Recovery Aid 環境の設定
- 最新の RLOG 情報の表示
- リカバリジョブストリームの作成

このchapterでは、次のトピックについて説明します。

## リカバリログ (RLOG)

リカバリログ (RLOG) には基本的な情報が記録されます。この情報は、PLOG と組み合わせて、ADARAI ユーティリティの RECOVER 機能によって、データベースステータスを障害の時点までリカバリ、リストアするためのジョブストリームを再構築するために使用されます。

RLOG の情報は世代別に分類され、各世代には、連続する ADASAV SAVE、RESTORE (データベース)、または RESTORE GCB 操作の間のデータベースアクティビティが含まれます。RLOG に格納される連続世代数は最小で4つ、最大で RLOG をアクティブにしたときに指定した最大値になります。最大値は32まで指定できます。指定した世代数を保持するために十分なスペースが RLOG になかった場合、ラップアラウンド方式で最も古い世代に最新の世代が上書きされます。

RLOG ファイルは、他のデータベースコンポーネントと同じように ADAFRM ユーティリティ (SIZE パラメータを使用) を実行してフォーマットします。その後、Adabas Recovery Aid の ADARAI ユーティリティ (RLOGSIZE パラメータを使用) の PREPARE 機能を使用して定義します。RLOG ファイルに必要なスペースは3380のシリンダ約10個、または同等のデバイススペースです。

ADARAI PREPARE 機能は、最初の世代の記録が開始される ADASAV SAVE の実行の直前に実行する必要があります。ADARAI PREPARE の実行後は、データベースを更新する後続のすべてのニュークリアスとユーティリティのジョブで RLOG ファイルを指定する必要があります。必要な場合は、RLOG ファイルをすべてのジョブストリームに含めることもできます。

RLOG ファイルのジョブステートメントは次のようになります。

```
//DDRLOGR1 DD DISP=SHR,DSN=... .RLOGR1
```

## Adabas Recovery Aid の起動

Adabas Recovery Aid のアクティビティと RLOG のログは、ADARAI PREPARE の実行後に、ADASAV SAVE/RESTORE データベースまたは RESTORE GCB 機能を初めて実行したときに開始します。

ADARAI PREPARE の実行後の最初と 2 番目の ADASAV SAVE/RESTORE データベースの間、または RESTORE GCB 操作の間のすべてのアクティビティが最初の世代になります。ADARAI ユーティリティの LIST 機能を使用して世代を表示すると、各世代は古い世代から順に番号が付けられ、昇順で表示されます。

Adabas Recovery Aid の設定の詳細については、『Adabas オペレーションマニュアル』の「再スタートおよびリカバリ手順」と『Adabas ユーティリティマニュアル』にある ADARAI ユーティリティの説明を参照してください。

---

## 20 マルチクライアントサポート

---

■ オーナーの概念 .....	102
■ スーパーユーザー .....	103
■ プログラムの互換性 .....	104
■ ソフトカップリングのサポート .....	104
■ データとインデックスの構造 .....	104
■ パフォーマンスの考慮事項 .....	106
■ ユーザープロファイルテーブル .....	107
■ 発生しうる Adabas レスポンスコード .....	107
■ マルチクライアントファイルのユーティリティサポート .....	107

Adabas マルチクライアント機能は、複数のユーザーまたはグループのレコードを1つの Adabas ファイルに格納します。この機能は、ファイルレベルで指定します。マルチクライアント機能では、各レコードにオーナー ID をつけることで、物理ファイルを複数の論理ファイルに分割します。各ユーザーは、ユーザーのオーナー ID に関連付けられたレコードのサブセットだけにアクセスできます。そのファイルは引き続き、1つの物理 Adabas ファイルとして維持されます。

Adabas ニュークリアスは、マルチクライアントファイルへの全データベース要求を処理します。

このchapterでは、次のトピックについて説明します。

## オーナーの概念

マルチクライアントファイル中の各レコードには、それぞれオーナーが存在します。オーナーは各レコードにつけられた内部オーナー ID によって識別されます（RACF または CA-TOPSECRET などの外部セキュリティパッケージについても、Natural ETID または LOGON ID のどちらかによって識別されます）。オーナー ID は、ユーザー ID に対して割り当てられます。1つのユーザー ID は、1つのオーナー ID しか持つことができませんが、同じオーナー ID が複数のユーザーに所属していてもかまいません。

次の表は、ETID/オーナー ID の関係の例を示しています。

ETID	オーナー ID	説明
USER1	1	複数のユーザーが同じオーナー ID を使用できます。 この例では、USER1、USER2、および USER3 が同じオーナー ID を共有しています。 つまり、同じレコードを共有しています。
USER2	1	
USER3	1	
...		
USER4	2	

ユーザー ID とオーナー ID との関係は、Adabas チェックポイントファイルのプロファイルテーブルに格納されています。マルチクライアント機能に欠くことのできない Adabas Online System / 基本サービス (AOS) を使って、DBA はプロファイルテーブルを維持します。

ユーザー ID とオーナー ID との関係は、1:1 または n:1 です。1つのオーナー ID に対して単一のユーザーかユーザーグループを割り当てることができます。レコードの分離は、常にオーナー ID のレベルで行われます。

オーナー ID は、8 バイト（英数字）以下の固定長です。オーナー ID の長さは、ファイル作成時にユーザーによって定義されます。長さの変更は、マルチクライアントファイルのアンロードやリロードによってのみ可能です。各オーナー ID の長さは、ファイルに指定された長さ以下でなければなりません。そうでない場合は、ゼロ以外のレスポンスコードが発生します。スペースの使用効率が悪くならないように、必要以上に長いオーナー ID は避けるべきです。

次の表に、8つのファイルレコードから成るグループの、オーナーの分離例を示します。

ISN	オーナー ID	レコード	説明
1	1	データ	さまざまなユーザーが所有するレコードの入った物理 Adabas ファイルの例
2	2	データ	
3	1	データ	
4	3	データ	
5	2	データ	
6	3	データ	
7		- データなし -	
8	1	データ	

ISN	レコード	ISN	レコード
1	データ	1	- データなし -
2	- データなし -	2	データ
3	データ	3	- データなし -
4	- データなし -	4	- データなし -
5	- データなし -	5	データ
6	- データなし -	6	- データなし -
7	- データなし -	7	- データなし -
8	データ	8	- データなし -
オーナー ID=1 のユーザーから見たファイル		オーナー ID=2 のユーザーから見たファイル	

## スーパーユーザー

スーパーユーザーオーナー ID は、マルチクライアントファイル中の全レコードへのアクセスを提供します。スーパーユーザーオーナー ID は、アスタリスク (\*) で始まります。このようなオーナー ID を持ったユーザーは、他のすべてのオーナー ID と一致するとみなされ、ファイル中の全レコードを読み込むことができます。1つのマルチクライアントファイルに対して、アスタリスクで始まり、同等の特権を持った複数のスーパーユーザーオーナー ID を定義できます。

スーパーユーザーオーナー ID は、Lx 読み込みコマンドと非ディスクリプタ検索 (Sx) コマンドにのみ適用されます。スーパーユーザーによるディスクリプタ検索コマンドは、スーパーユーザーのオーナー ID を持ったレコードだけを返します。スーパーユーザーが格納したデータレコードやインデックス値は、スーパーユーザーのオーナー ID でラベル付けされます。



**Note:** スーパーユーザーが L3 または L9 コマンドを発行した場合、開始値オプションは無視されます。すなわち、Adabas は常に特定ディスクリプタの先頭値から読み込みを開始します。

## プログラムの互換性

---

マルチクライアント環境で使用するために既存アプリケーションプログラムを変更する必要はありません。しかし、マルチクライアントファイルにアクセスするユーザーは、各オープン（OP）コールの Adabas コントロールブロックのアディション1フィールドにユーザー ID を指定しなければなりません。これにより、Adabas はチェックポイントファイルからオーナー ID を取り出すことができます。ユーザー ID を指定しなかった場合、アプリケーションプログラムは、マルチクライアントファイルにアクセスするのか、あるいは標準の Adabas ファイルなのかを認識することもできません。

## ソフトカップリングのサポート

---

ソフトカップリングのマルチクライアントサポートがあります。

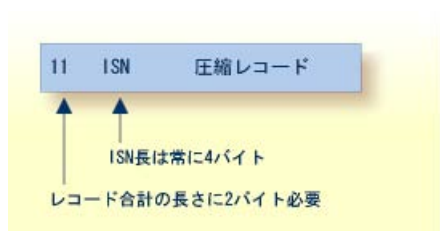
## データとインデックスの構造

---

マルチクライアントファイルのデータやインデックスの構造は、標準 Adabas ファイルの場合とは異なっています。

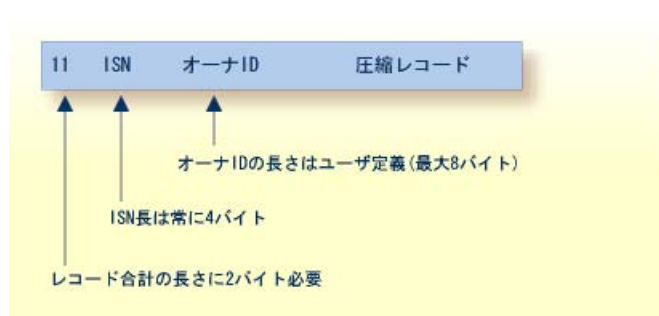
### データストレージ

標準ファイルのデータストレージ（DATA）レコードの構造は、次のとおりです。





マルチクライアントファイルのデータストレージレコードの構造は、次のとおりです。



### 通常とマルチクライアントレコードフォーマットの比較

#### アソシエータ

マルチクライアントファイルのすべてのノーマルインデックス値とアップパーインデックス値の先頭にオーナー ID が付いています。



#### ノーマルインデックスエレメントの構造

次の表は、マルチクライアントのインデックスの構造を示しています。1つのディスクリプタ値が複数のデータストレージレコードをポイントする場合、Adabasは後続にISNリストを付け、その拡張インデックス値を1回だけ格納します。同一ディスクリプタ値に複数のオーナー ID がある場合は、格納時にインデックスに複数のエントリが作成されます。

ISN	オーナー ID	NAME	
1	1	SMITH	フィールド NAME はディスクリプタ
2	2	SMITH	
3	1	SMITH	
4	3	JONES	
5	2	JONES	
6	3	HARRIS	
7		格納されない	
8	1	HARRIS	

オーナー ID	DE 値	ISN カウント	ISN リスト	
1	HARRIS	1	8	ディスクリプタ NAME のインデックス。値のソート順： オーナー ID ↓ (DE 値)
1	SMITH	2	1,3	
2	JONES	1	5	
2	SMITH	1	2	
3	HARRIS	1	6	
3	JONES	1	4	



**Notes:**

- すべてのディスクリプタタイプ（シンプルディスクリプタ、サブ/スーパーディスクリプタ、フォネティック、およびハイパーディスクリプタ）の先頭にオーナー ID が付きます。接頭辞のオーナー ID は、スーパーディスクリプタやハイパーディスクリプタの親フィールドとしてカウントされません。したがって、最大の親フィールド数には影響はありません。
- オーナー ID を含むディスクリプタ値の最大長は 253 バイトです。
- L3/L9 シーケンスでインデックス値を読み込むスーパーユーザーは、オーナー ID のソート順で値を取得します。すなわち、まず最小のオーナー ID の値を読み込んだ後、その次に高いオーナー ID の値を読み込み、以後同様に続きます。各オーナー ID の値は、昇順にソートされます。

## パフォーマンスの考慮事項

マルチクライアント機能は、検索 (S1、S2)、論理順読み込み (L3)、およびヒストグラム (L9) コマンドに対していっさいの処理オーバーヘッドを付加しません。インデックス構造によって特定レコードの選択が可能になり、選択後のデータストレージにはいっさいの処理もありません。

データストレージ上で選択を行う場合、Adabas はレコードを読み込んでオーナー ID をチェックしなければなりません。レコードのオーナー ID が、現在のユーザーのオーナー ID と一致し

なければ、そのレコードは読み飛ばされます。これにより、物理順読み込み (L2) や ISN による読み込み (I オプション付きの L1) コマンド、または非ディスクリプタ検索コマンドが遅くなることがあります。

## ユーザープロフィールテーブル

オーナー ID は、Adabas プロファイルテーブルに格納されたユーザーのプロファイルレコードの一部分です。プロフィールは、Adabas Online System を使って維持されます。詳しくは Adabas Online System のドキュメントを参照してください。

## 発生しうる Adabas レスポンスコード

マルチクライアントファイルをコールすると、エラーの発生を示す、次のようなゼロ以外の Adabas レスポンスコードが発生する可能性があります。

**読み込みおよび更新処理** レコードに該当しないオーナー ID を使って、マルチクライアントファイルのレコードの読み込みまたは変更を試みると、読み込みまたは更新処理の種類に応じてレスポンスコード 3 または 113 のどちらかが返されます。

**レコード追加処理** オーナー ID が空白であったり、マルチクライアントファイルに割り当てられたオーナー ID 長よりも長すぎたりすると、新レコードの追加を試みたときに、レスポンスコード 68 が返されます。

**空白またはオーナー ID の欠如** ユーザーのオーナー ID が空白の場合や指定されていない場合、マルチクライアントファイルにアクセスすると、レスポンスコード 3 または 113 が返されます。

## マルチクライアントファイルのユーティリティサポート

通常、マルチクライアントファイルは Adabas ユーティリティ処理に対して透過的です。ADALOD や ADAULD ユーティリティの特殊な機能は、標準環境からマルチクライアント環境へのアプリケーションの移行をサポートします。

## ADALOD ユーティリティ LOAD 機能

ADALOD LOAD 機能の2つのパラメータ、LOWNERID と ETID は、マルチクライアントファイルをサポートしています。これらのパラメータが組み合わされて、オーナー ID の定義と、およびファイルがマルチクライアントファイルかどうかの判断が行われます。

LOWNERID は、マルチクライアントファイルの各レコードに割り当てる内部オーナー ID 値の長さを指定します。

有効な長さは 0~8 です。ETID パラメータと組み合わせると、LOWNERID パラメータは、標準ファイルをマルチクライアントファイルとして再ロードしたり、ファイルのオーナー ID 長を変更したり、ファイルのレコードからオーナー ID を削除するために使用できます。

LOWNERID パラメータを指定しなければ、入力ファイル用のオーナー ID の長さは、引き続き同じ長さになります。

ETID は、マルチクライアントファイルの中にロードされる全レコードに新しいオーナー ID を割り当てます。入力ファイルにオーナー ID が含まれていない場合、つまり、マルチクライアントソースファイルからアンロードした入力ファイルでない場合、ETID パラメータを必ず指定しなければなりません。

次の表は、LOWNERID と ETID の設定値の効果を示しています。

LOWNERID パラメータ設定	入力ファイルのオーナー ID 長	
	0	2
0	非マルチクライアントファイルとして保持	非マルチクライアントファイルに変換します。
1	マルチクライアントファイルの設定 (ETID)	オーナー ID 長の縮小
2	マルチクライアントファイルの設定 (ETID)	オーナー ID 長の保持
3	マルチクライアントファイルの設定 (ETID)	オーナー ID 長の拡張
(LOWNERID が指定されていない)	非マルチクライアントファイルとして保持	マルチクライアントファイルとして保持

上の表で、"オーナー ID 長が 0" の欄に、" (ETID) " と示されている場合、ETID パラメータはロードするレコードのオーナーを識別するユーザー ID を指定しなければなりません。レコードに割り当てられるオーナー ID は指定したユーザー ID のユーザープロファイルから取得します。"オーナー ID 長が 2" の欄では、ETID パラメータはオプションです。ETID を省略すると、元のオーナー ID がそのまま保持されます。



**Note:** ETID パラメータを使用する場合、ADALOD ユーティリティはアクティブニュークリアスを必要とします。ニュークリアスは、ETID 値を内部オーナー ID 値に変換します。

## ADALOD ユーティリティの UPDATE 機能

UPDATE 機能を実行するとき、ADALOD は更新するファイルに以前から定義されていたオーナー ID 長を保持します。追加するレコードのオーナー ID は、ファイルに対して定義されたオーナー ID 長に合わせて調節されます。ロードレコードまたは新レコードのオーナー ID は、既存のオーナー ID スペースに収容されなければなりません。

例

```
ADALOD LOAD FILE=20,LOWNERID=2,NUMREC=0
```

ファイル 20 をマルチクライアントファイルとして作成します。内部オーナー ID 長は 2 バイトですが、実オーナー ID は 1 つも指定されていません。実際にファイルにはレコードが 1 つもロードされていません (NUMREC=0)。

```
ADALOD LOAD FILE=20,LOWNERID=2,ETID=USER1
```

ファイル 20 をマルチクライアントファイルとして作成し、すべてのレコードをユーザー USER1 用にロードします。内部オーナー ID の長さは 2 バイトです。

```
ADALOD UPDATE FILE=20,ETID=USER2
```

大量の更新を実行してマルチクライアントファイルのファイル 20 にレコードを追加します。新レコードはすべて USER2 用にロードします。

## ADAULD ユーティリティ

ADAULD ユーティリティは、Adabas ファイルからシーケンシャル出力ファイルにレコードをアンロードします。この出力ファイルは、後続の ADALOD 処理への入力として使用できます。

マルチクライアントファイルをアンロードした場合、出力ファイルにはオーナー ID を伴ったすべてのアンロードレコードが入っています。この情報は、後続の ADALOD 処理によって保持されるか、ADALOD ETID パラメータによって新情報に書き換えることができます。アンロードファイルや新規にロードされたファイルの LOWNERID パラメータ値が異なる場合、それらは自動的に ADALOD によって調節されます。

ADAULD の ETID パラメータを使い、UNLOAD 処理を指定ユーザーの所有するレコードだけに限定することができます。ETID パラメータを指定しなければ、全レコードをアンロードします。マルチクライアントファイルに対して SELCRIT/SELVAL パラメータが指定されている場合は、必ず ETID パラメータも指定しなければなりません。

例

```
ADAULD UNLOAD FILE=20,ETID=USER1
```

USER1 が所有する全レコードを物理順にアンロードします。

### ADACMP ユーティリティ

ADACMP ユーティリティは、シーケンシャル入力ファイルのデータを圧縮して Adabas 内部構造の中に入れるか、Adabas データをシーケンシャルユーザーファイルに圧縮解除します。COMPRESS 機能は、標準ファイルとマルチクライアントファイルをまったく区別せずに、まったく同じ方法で処理します。INFILE パラメータでマルチクライアントファイルを指定し正しい ETID 値を指定すれば、DECOMPRESS 機能は自動的に選択したレコードを圧縮解除します。

DECOMPRESS 機能は、オーナー ID (ある場合) をスキップします。マルチクライアントファイルの DECOMPRESS 処理の出力には、オーナー ID も ETID 情報も含まれていません。

DECOMPRESS 機能に対して INFILE パラメータがマルチクライアントファイルを指定した場合、ETID パラメータを使用した特定ユーザーのレコードだけに圧縮解除を制限することができます。このとき、ADACMP は特定ユーザーのレコードを読み込んで圧縮解除します。マルチクライアントファイルの圧縮解除時に ETID パラメータを指定しなければ、ファイル内のすべてのレコードが圧縮解除されます。

#### 例

```
ADACMP  DECOMPRESS  INFILE=20,ETID=USER1
```

ファイル 20 の USER1 が所有するレコードを圧縮解除して、シーケンシャル出力ファイルに出力します。

# 21 拡張ファイル

---

■ 概要 .....	112
■ 拡張ファイルの定義 .....	113
■ コンポーネントファイルの挿入 .....	115
■ コンポーネントファイルの削除 .....	115
■ 拡張ファイルの削除 .....	115
■ 拡張ファイルの調査 .....	155
■ 拡張ファイルと Adabas ニュークリアス .....	116
■ 拡張ファイルと Adabas ユーティリティ .....	117

このchapterでは、次のトピックについて説明します。

## 概要

---

拡張ファイルは、1つ以上の物理コンポーネントファイルで構成された論理ファイルです。各コンポーネントファイルには、物理 ISN 番号ではなく、論理的に番号付けされたレコードが入っています。これらの物理コンポーネントファイルには、次のものがが必要です。

- 同一のフィールド定義テーブル (FDT)
- ファイルの MINISN と MAXISN パラメータによって定義された、異なる論理 ISN 範囲。ISN 範囲は重複してはなりません。

コンポーネントファイル間は、昇順の ISN 順に連鎖しています。最小の ISN 範囲を持つファイルをアンカーファイルと呼び、そのファイル番号は、拡張ファイル全体の番号です。

拡張ファイルは、最高 128 個のコンポーネントファイルで構成できます。また、4,294,967,294 件のレコードを超過してはなりません。3 バイト ISN の Adabas コンポーネントファイルには、最高 16,777,215 件までのレコードが入り、4 バイト ISN のコンポーネントファイルには、4,294,967,294 件までのレコードが入ります。



**Note:** Adabas は大きいファイルサイズや、より多くの Adabas 物理ファイルやデータベースをサポートするので、拡張ファイルの必要性はほとんどなくなっています。

拡張ファイルは、Adabas の次の機能でサポートされます。

- ISN リストを処理するコマンド S8
- ソートコマンド S2 と S9。この機能を使う前に、データベースパフォーマンスやユーザーへの影響について調査してください。
- 拡張ファイル処理時の ET/BT サポートのために機能強化されたプリフェッチ/マルチフェッチ機能



## 拡張ファイルの定義

### ADALOD の使用

拡張ファイルはロード時に定義します。各物理コンポーネントファイルは、ADALOD LOAD 機能を使って別々にロードします。最初のコンポーネントファイルを除くすべてのファイルは、アンカーファイルを参照するために ANCHOR パラメータを指定しなければなりません。ADALOD LOAD 機能は、次の処理を実行します。

- 新しいファイルの FDT とアンカーファイルの FDT を比較して、それらが同じであることを確認します。
- 新しい ISN 範囲（新コンポーネントファイルの MINISN から MAXISN まで）をアンカーおよび他のすべてのコンポーネントファイルの ISN 範囲に対してチェックし、重複がないことを確認します。
- NOACEXTENSION パラメータの指定をチェックします（アドレスコンバータエクステンションはコンポーネントファイルには許可されません）。
- 新しいコンポーネントファイルの MAXRECL パラメータが、既存のアンカーファイルのと同じかチェックします。すべてのコンポーネントファイルの MAXRECL 値は同一でなければなりません。
- 新しいコンポーネントファイルをロードします。
- 新しいコンポーネントファイルを拡張ファイルチェーンの中にリンクします。

### 例

次の例は、ADALODLOAD ステートメントの拡張ファイルの定義について示しています（関連パラメータだけを示します）。

```
ADALOD LOAD FILE=11,NOACEXTENSION
ADALOD      MINISN=1,MAXISN=16000000
ADALOD      ...

ADALOD LOAD FILE=23,NOACEXTENSION
ADALOD      ANCHOR=11
ADALOD      MINISN=36000001,MAXISN=50000000
ADALOD      ...

ADALOD LOAD FILE=17,NOACEXTENSION
ADALOD      ANCHOR=11
ADALOD      MINISN=20000001,MAXISN=36000000
```

この例は、ファイル 11 を拡張ファイルとしてロードします。拡張ファイルの構成は、次のとおりです。

ファイル 11、ISN 範囲： 1～16,000,000

ファイル 17、ISN 範囲： 20,000,001～36,000,000

ファイル 23、ISN 範囲： 36,000,001～50,000,000

### オンラインシステムの使用

拡張ファイルは、Adabas Online System のファイル定義機能を使って定義することもできます。この機能は、拡張ファイルのアンカーまたはコンポーネントファイルとして指定でき、新しい空のファイルを作成します。拡張ファイルメンテナンス機能を使用して既存のファイルを連鎖させることができます。

### 拡張ファイルの定義規則

1. 指定ファイルのアドレスコンバータの拡張（つまり、MAXISN の増加）を防止するために、必ず NOACEXTENSION パラメータを設定しなければなりません。
2. 拡張ファイルのコンポーネントファイルをロードするとき、必ず MINISN パラメータを指定しなければなりません。
3. コンポーネントファイルのファイル番号は、自由に選択できます。
4. ANCHOR および FILE パラメータが同じファイル番号を指定すると、単一ファイルを拡張ファイルとしてロードします。
5. 2つめのコンポーネントファイルをロードするとき、拡張する既存の単一ファイルをアンカーファイルとして参照できます。この場合、ADALOD は最初のファイルに NOACEXTENSION を設定してアンカーファイルにします。



**Note:** この方法で作成したアンカーファイルは、コンポーネントファイルをすべて削除したときにそのアンカーステータスが失われます。必要な場合は、ファイルをそれぞれに挿入して、アンカーステータスを再び確立できます。

6. コンポーネントファイルの ISN 範囲は重複してはなりませんが、ファイル範囲間で未使用 ISN のギャップが存在してもかまいません。
7. コンポーネントファイルが任意の順番でロードできます。
8. 現在のアンカーファイルよりも低い ISN 範囲をもった新しいコンポーネントファイルをロードした場合、新しくロードしたファイルがアンカーファイルになります。それ以降にロードするすべてのコンポーネントファイルの ANCHOR パラメータは、新しいアンカーファイルを参照しなければなりません。

---

## コンポーネントファイルの挿入

---

拡張ファイルの中にコンポーネントファイルを挿入するには、「[拡張ファイルの定義](#)」で説明しているように ADALOD LOAD 機能を使うか、Adabas Online System を使います。

オンラインシステムを使う場合は、ファイル定義機能を使って新しいファイルの作成と拡張ファイルへの挿入ができます。また、コンポーネントファイル挿入機能を使って、既存のファイルを拡張ファイルの中に挿入することができます。

コンポーネントファイルの追加による影響については、[このページ](#)の「[拡張ファイルの定義規則](#)」を参照してください。

---

## コンポーネントファイルの削除

---

Adabas Online System のファイル削除機能を使って、コンポーネントファイルを拡張ファイルから取り除いて削除することができます。また、ファイルを削除せずに、拡張ファイルチェーンからコンポーネントファイルを取り除くには、Adabas Online System のコンポーネントファイルリムーブ機能を使用します。

コンポーネントファイルの追加による影響については、[このページ](#)の「[拡張ファイルの定義規則](#)」を参照してください。

---

## 拡張ファイルの削除

---

Adabas Online System のファイル削除機能では、拡張ファイル全体を削除することもできます。これにより、アンカーファイルとすべてのコンポーネントファイルが削除できます。ADADBS ユーティリティの `DELETE` 機能も、拡張ファイル全体の削除に使用できます。

---

## 拡張ファイルの調査

---

ADAREP ユーティリティの出力レポートは、個々のファイルに関する通常の情報に加えて、データベース内の各拡張ファイルのコンポーネントファイルリストを出力します。拡張ファイル情報は、Adabas Online System のファイル表示機能を使っても入手できます。

## 拡張ファイルと Adabas ニュークリアス

拡張ファイルを参照するユーザーコールは、Adabas ニュークリアスによって自動的に適切な物理コンポーネントファイルに送られます。ユーザーやアプリケーションは、選択したファイルが拡張ファイルであるという通知をいっさい受けません。

Adabas コントロールブロックのファイル番号が拡張ファイルのコンポーネントファイルを指定した場合、そのコールは拡張ファイル全体に向けられたものと解釈されます。したがって、拡張ファイルに統合されたファイルであれば、以前に既存のコンポーネントファイルにアクセスしたユーザーアプリケーションを変更せずにそのファイルにアクセスできます。なぜなら、コールは自動的に全拡張ファイルに適用されるからです。しかし、便宜上、コールはアンカーファイルを参照することをお勧めします。

拡張ファイルに対して実行した機能が、複数のコンポーネントファイルからの結果を必要とする場合、それらの結果は結合されて1つの結果を作成します。例えば、拡張ファイルに L2 コマンド（物理順に読み込み）を実行すると、アンカーファイルから順番にすべてのコンポーネントファイルの読み込みが行われます。すなわち、1つのコンポーネントファイルがエンドオブファイルに達すると、L2 は続けて次のコンポーネントファイルを自動的に読み込みます。実行結果は、読み込まれた全ファイルから順番に累積されます。

その一方で、L3 コマンド（ディスクリプタ順の論理順読み込み）は、各コンポーネントファイルを別々に並行してコールするので、結果はコール元に返される前に1つのシーケンスにマージされます。

### 拡張ファイルのためのニュークリアスの変更

1つのコマンドに対するコンポーネントファイルの並列処理を効率よく行うために、ニュークリアスセッションの ADARUN パラメータ値を大きくすることをお勧めします。

パラメータ	説明
LI	ISN テーブル (TBI) の長さ
LQ	シーケンシャルコマンドテーブルの長さ
LWP	Adabas ワークプールエリアの長さ
LS	検索/ソートエリアの長さ
NQCID	ユーザーあたりに許可された最大アクティブコマンド ID (CID) 数

## 拡張ファイル使用時の制限拡張

拡張ファイル上で実行するプログラムには、次の制限事項が適用されます。

- 現在、拡張ファイルでは、物理カップリングやソフトカップリングはサポートされません。
- 拡張ファイルのマルチクライアントサポートはありません。
- いったん設定した拡張ファイルのコンポーネントファイル番号を後からリナンバリングすることはできません。
- 拡張ファイルに Adabas Security を使用する場合、次の事項は全コンポーネントファイルで同一でなければなりません。
  - プロテクションプロファイル
  - パスワード
  - セキュリティバイバリュープロファイル
  - サイファコード

## 拡張ファイルと Adabas ユーティリティ

拡張ファイルは Adabas コールを発行したユーザーには透過的ですが、Adabas ユーティリティを実行する DBA は拡張ファイルの存在を意識しておく必要があります。Adabas ユーティリティは、次のどちらかの方法で拡張ファイルを処理します。

- 拡張ファイル全体を処理します。
- コンポーネントファイルを処理します。

### 拡張ファイル全体を処理する機能

拡張ファイル全体を処理するユーティリティ機能には、ADADBS、ADARES、および ADASAV の次の機能があります。

#### ADADBS DELETE 機能

拡張ファイル全体を削除します。

### ADARES REGENERATE および BACKOUT FILE 機能

ファイルリスト中にコンポーネントファイルの1つが指定されていると、拡張ファイル全体として処理します。このとき、他のすべてのコンポーネントファイルも指定しなければなりません。

### ADASAV

#### ■ RESTORE (ファイル) 機能

ファイルリスト中にコンポーネントファイルの1つが指定されていると、拡張ファイル全体として処理します。このとき、他のすべてのコンポーネントファイルも指定しなければなりません。

#### ■ SAVE (ファイル) 機能

ファイルリスト中にコンポーネントファイルの1つが指定されていると、拡張ファイル全体として処理します。Adabas ニュークリアスの稼働中に SAVE (ファイル) 機能を実行するときは、他のコンポーネントファイルもすべて指定しなければなりません。

### コンポーネントファイルを処理する機能

コンポーネントファイルを処理するユーティリティ機能には、ADADBS、ADAINV、ADALOD、ADAORD、ADAACK、ADADCK、ADAICK、ADAVAL、ADAULD、ADASCR の次の機能があります。

### ADAACK、ADADCK、ADAICK、ADAVAL、ADAULD

これらのユーティリティ機能はすべて、単一のコンポーネントファイルだけをチェックします。

### ADADBS

■ **CHANGE/NEWFIELD** 機能 これらの機能は、単一のコンポーネントファイルのフィールド定義テーブル (FDT) を変更します。DBA は、拡張ファイルの全コンポーネントファイルに対して CHANGE または NEWFIELD 機能を実行しなければなりません。ADADBS は、指定されたファイルが拡張ファイルの一部であることを示すメッセージを出力した後、コンディションコード 4 で終了します。

■ **RELEASE** 機能 単一コンポーネントファイルのディスクリプタのインデックスを解放します。DBA は、拡張ファイルの全コンポーネントファイルに対して RELEASE 機能を実行しなければなりません。ADADBS は、指定されたファイルが拡張ファイルの一部であることを示すメッセージを出力した後、コンディションコード 4 で終了します。

## ADAINV

- **INVERT 機能** 単一コンポーネントファイルの新しいディスクリプタにインデックスを作成します。DBAは、拡張ファイルの全コンポーネントファイルに対してINVERT機能を実行しなければなりません。ADAINVは、指定されたファイルが拡張ファイルの一部であることを示すメッセージを出力した後、コンディションコード4で終了します。
- **COUPLE 機能** ADAINV COUPLE 機能は、現在では拡張ファイルに対して使用できません。

## ADALOD UPDATE 機能

単一のコンポーネントファイルにレコードを追加／削除します。複数またはすべてのコンポーネントファイルの大量更新を実行するときは、すべてのコンポーネントファイルの完全な削除ISNリストを提供できます。ADALODは自動的に、現在処理中のコンポーネントファイルに該当するISN値だけを指定範囲から選択します。USERISN=YESで新規レコードを追加するときも同じです。

USERISN=NOで新レコードを追加するときに空きISNが見つからないと、ISN範囲の増加ができないので（NOACEXTENSIONは全コンポーネントファイルに対して有効）、ローダーは新アドレスコンバータエクステントを割り当てることができません。その代り、ADALODは、エンドオブファイルになったときと同じようにインデックスを作成します。ロードされなかった残りのレコードは後からSKIPRECパラメータを使用して別のコンポーネントファイルに追加することができます。

ADALODは、各コンポーネントファイル間に渡ってのユニークディスクリプタ値のチェックは行いません。

## ADAORD REORFILE/REORFASSO/REORDATA 機能

これらの機能は、単一コンポーネントファイルのそれぞれの領域をリオーダします。ファイルは論理的に変更されないため、これらの機能を拡張ファイルの全コンポーネントファイルに実行する必要はありません。

## ADASCR (Adabas Security) 機能

個々のコンポーネントファイルのセキュリティプロファイルを定義します。各コンポーネントファイルのプロテクション、パスワード、セキュリティバイバリュー、およびサイファコードは、拡張ファイルの全コンポーネントファイルに対して同じに定義しなければなりません。





# 22 データベースのメンテナンス

---

この章では、Adabas データベースの定義、保守、および実行に関わる作業について説明します。

この情報は次の項目で構成されています。

- *Adabas* データベースの定義
- データベースのスペース管理
- データベースのモニタリングとチューニング
- エラー処理およびメッセージバッファリング
- ユニバーサルエンコーディングサポート (**UES**)
- 複数プラットフォームのサポート
- ラージオブジェクト (**LB**) フィールドの基本

---

# 23 Adabas データベースの定義

---

- ステップ 1: データベースに必要なサイズを見積ります。 ..... 124
- ステップ 2: データベースの割り当て ..... 141
- ステップ 3: データベースのフォーマット ..... 143
- ステップ 4: データベースパラメータの定義 ..... 144

このchapterでは Adabas データベースを定義するステップについて説明します。DBA は新しいデータベースを定義する前に各ステップに示されたドキュメントを読み、理解しておく必要があります。

新しいデータベースを定義するには、次の手順に従います。

これらのステップが正しく完了すれば、ADACMP および ADALOD ユーティリティを使用して、ユーザーファイルをデータベースにロードできます。

## ステップ1：データベースに必要なサイズを見積ります。

---

### ニュークリアスに必要なコンポーネント

Adabas ニュークリアスには、データストレージ、アソシエータ、およびワークエリアの3つのデータベースコンポーネントが必要です。

#### データストレージ

データストレージコンポーネントには、データベース内の各ファイルの圧縮データレコードが含まれます。

#### アソシエータ

アソシエータには、データベース内の各ファイルに対応する要素、またデータベース全体に対応する要素が含まれます。

データベース内のファイルごとに、インバーテッドリスト、アドレスコンバータ、フィールド定義テーブル (FDT) がアソシエータに含まれます。

- インバーテッドリストは、Adabas の検索コマンドを処理し、論理順でレコードを読み込みます。インバーテッドリストは、ノーマルインデックス (NI) と最大 14 個のアップパーインデックス (UI) から構成されます。NI には、ファイル内の各ディスクリプタのすべての値が、各値を含むレコード数とこれらのレコードの ISN のリストとともに含まれます。検索の効率を向上するため、Adabas によって必要に応じて UI レベルが自動的に作成され、各レベルで下位レベルのインデックスが管理されます。管理対象の NI と同様に、最初のレベルの UI には、各インデックスブロックの 1 つのディスクリプタに対するエントリのみが含まれます。その他のすべての UI レベルには、各インデックスブロックのすべてのディスクリプタのエントリが含まれます。アップパーインデックスに必要なスペースはわずかです。最小サイズは 2 ブロックです。
- アドレスコンバータは、レコードの論理 ID (ISN) を、レコードが格納されているデータストレージブロックの相対的な Adabas ブロック番号 (RABN) にマッピングします。ISN 順の RABN のリストが含まれます。例えば、アドレスコンバータの 15 番目のエントリには、ISN 15 の RABN が含まれます。

- フィールド定義テーブル (FDT) は、Adabas ファイルの論理的な内容を定義します。FDT には、ファイル内の各フィールドの名前、レベル、長さ、形式、指定オプションが含まれます。

データベース全体については、アソシエータにストレージ管理テーブルとカップリングリストが含まれます。

- ストレージ管理テーブルには、アソシエータと、割り当て可能なデータストレージブロック、また各データストレージブロック内の未使用スペースのリストが含まれます。
- カップリングリストは物理的にカップリングされたファイルだけのためにあり、複数のファイルのディスクリプタが使用されている検索コマンドの処理に使用されます。

## ワーク

ワークワークエリアは、次の 4 つのパートに情報を格納します。

- パート 1 は、自動再起動と自動バックアウトのルーチンに必要なデータ保護情報を格納します。
- パート 2 は、検索コマンドの中間結果 (ISN リスト) を格納します。
- パート 3 は、検索コマンドの最終結果 (ISN リスト) を格納します。
- パート 4 は、分散トランザクション処理に関連するデータを格納します。



**Note:** Adabas Transaction Manager バージョン 7.5 以降では、DDWORKR1 の WORK パート 4 はサポートされていません。その代わりに別の WORK データセット DDWORKR4 が必要です。DDWORKR4 は、DDWORKR1 の WORK パート 4 と同じ目的で使用されるコンテナデータセットですが、クラスタのすべてのメンバが同時に使用できます。DDWORKR4 データセットは、DDWORKR1 以上のブロックサイズを使用して、通常の方法で割り当てとフォーマットを行う必要があります。少なくとも、データベースのクラスタの LP パラメータ、またはクラスタと同じ大きさのエクステンションが確保される必要があります。

## その他のコンポーネント

### ソートエリアと一時エリア

Adabas ユーティリティ ADAINV、ADALOD、および ADAVAL は、データのソートと中間データの格納のために、さらに 2 つのデータセット、SORT と TEMP を必要とします。

TEMP や SORT のサイズは、実行するユーティリティ機能に応じて異なります。これらのデータセットは、ジョブの実行中に割り当てと解放を行うか、永久的なデータセットを割り当てて再利用することができます。

## ログ

Adabas には、次のオプションログがあります。

- コマンドログ (CLOG) は、発行される各 Adabas コマンドのコントロールブロックの情報を記録します。CLOG が提供する監査証跡を使って、デバッグやリソース使用状況のモニタリングができます。シングル、デュアル、またはマルチ (2~8) データセットを使用できます (マルチデータセットの使用をお勧めします)。
- プロテクションログ (PLOG) は、データベースに変更が生じたときに、レコードやその他のエレメントの変更前イメージと変更後イメージを記録します。PLOG 情報は、再起動後のデータベースのリカバリに使われます (最新の ET まで)。シングル、デュアル、またはマルチ (2~8) データセットを使用できます (マルチデータセットの使用をお勧めします)。
- リカバリログ (RLOG) は、Adabas Recovery Aid でリカバリジョブストリームの構築に使用される追加情報を記録します。



**Note:** CLOG、PLOG、および RLOG の各データセットは、16,777,215 (x'FFFFFF') ブロック/RABN に制限されます。

## 一般的なスペース要件

アソシエータ (NI、UI、および AC) とデータストレージに必要なスペースは、それぞれ ADALOD ユーティリティと ADACMP ユーティリティによってファイルごとに自動的に計算されます。ファイルに対して特定のスペースを割り当てたいとき、またはこれらのユーティリティを実行せずにファイルの必要スペースを見積りたいときは、この章に示された公式を使って計算することができます。

データベースの設定時に、データベースに最終的にロードされるファイルの数やサイズが分からない場合は、アソシエータやデータストレージに余分なスペースを割り当てる必要はありません。なぜなら、ADADBS ユーティリティの ADD または INCREASE 機能を使って、後からスペースを増やすことができるからです。

しかし、アソシエータやデータストレージの初期のスペース割り当てでは、現在計画されているすべてのファイルをロードできるだけでなく、適量のデータベース拡張 (新ファイルの追加または既存のファイルの更新) が可能である必要があります。

アソシエータスペースの見積り時には、データベース内の各ファイル (ノーマルインデックス、アップパーインデックス、アドレスコンバータ) の見積もりに加えて、データベース全体としての次の要件を追加しなければなりません。

- アソシエータの先頭 30 ブロックは Adabas が内部制御情報の格納に使用します。アソシエータ、データストレージ、およびワークの物理ブロックサイズは、Adabas コンポーネントごとに異なり、各コンポーネントの配置されたデバイスタイプに応じて異なることに注意する必要があります。

- Adabas はファイル制御情報用に MAXFILES パラメータ値の 5 倍に相当するアソシエータブロックを確保します。MAXFILES パラメータは、ADADEF ユーティリティの実行時に設定します。

### 一般的なスペース見積り手順

1. ファイルごとの要件を見積った後、それらを加算してデータベース全体の見積りを出します。
  - アソシエータ（ノーマルインデックス、アッパーインデックス、アドレスコンバータ）
  - データストレージ
2. データベース全体について次の要件を見積ります。
  - アソシエータ（Adabas が確保するスペース）
    - 内部制御情報用に先頭 30 ブロックを確保します。
    - ファイル制御情報用に  $\text{MAXFILES} * 5$  ブロック（ADADEF の MAXFILES パラメータで、データベースにロード可能な最大ファイル数を指定）
  - ワークエリア、ソートエリア、一時エリア、ログ

### 見積りの公式

以降の各セクションでは、各コンポーネントに割り当てるスペースを見積る公式を記述します。

- アソシエータ（以下に換算して）
  - ノーマルインデックス (NI)
  - アッパーインデックス (UI)
  - アドレスコンバータ (AC)
- データストレージ
- ワーク（以下に換算して）
  - パート 1（データ保護情報）
  - パート 2（検索コマンドの中間結果）
  - パート 3（検索コマンドの ISN リスト）
  - パート 4（2 フェーズコミット処理に関連するデータ）
- ソートスペース

## 公式の計算規則

公式は一般的な規則に従って計算します。つまり、各式は次の順序で評価します。

1. カッコ内の要素
2. 乗算および除算
3. 加算および減算
4. 左から右へ（要素の優先レベルが同じであれば、左側にある要素を先に評価）。

## ノーマルインデックス (NI)

ファイルの各ディスクリプタに必要なノーマルインデックススペースの見積りには、次の公式を使用します。

```
NIRBYTES = ISNSIZE * AVUQVAL * RECORDS + DESCVALS * (AVLENG + 2)
```

ここでは次の内容を表しています。

**NIRBYTES** ノーマルインデックスの必要スペースをバイト数で示します。

**ISNSIZE** ファイル内の ISN 長（3 または 4 バイト）。ISN 長は ADALOD パラメータの ISNSIZE に指定します。

**AVUQVAL** 各レコードに存在するディスクリプタのユニーク値の平均個数を示します。

**RECORDS** ファイルに収容するレコード数。ADALOD の MAXISN パラメータによって指定します。

**DESCVALS** ファイル内に存在するディスクリプタのユニーク値の個数を示します。

**AVLENG** ディスクリプタの値の平均長を示します。

## AVUQVAL

ディスクリプタがマルチプルバリューフィールド (MU) またはピリオディックグループ (PE) の一部である場合を除いて、AVUQVAL は 1 以下の値です。

ディスクリプタが NU（空値省略）オプション付きで定義されている場合、AVUQVAL は各レコードに存在する値の平均個数から空値（空のフィールド）を含むレコードの割合を引いた値になります。例えば、各レコードにディスクリプタ値が 1 つずつ存在し、値の 20 パーセントが空値である場合は次のようになります。

```
AVUQVAL = 1 - 0.2 = 0.8
```

同様に、MU フィールドが各レコードに平均 10 個の値をもっていて、値の 20 パーセントが空値である場合は次のようになります。

```
AVUQVAL = 10 - 2 = 8
```



**AVLENG**

ディスクリプタフィールドが FI (固定長) オプション付きで定義されていない場合は、AVLENG は長さバイトを含むフィールドの平均圧縮長になります。ディスクリプタが FI オプション付きで定義されている場合、AVLENG はフィールドの標準長に等しくなります。

**ISNSIZE \* AVUQVAL \* RECORDS**

ISNSIZE \* AVUQVAL \* RECORDS は、ISN の格納に必要なスペースを表します。多数の重複値をもったディスクリプタには、重要な要因です。

**DESCVALS \* (AVLENG + 2)**

DESCVALS \* (AVLENG + 2) は、ディスクリプタ値の格納のために必要なスペースを表します。多数のユニーク値をもったディスクリプタにとっては、大事な要因となります。

**バイトからブロックへの変換**

バイトをブロックに変換するには、次の公式を使います。結果は次のブロックに切り上げます。

$$\text{NIRBLOCKS} = \text{NIRBYTES} / (\text{ASSOBLKSIZE} * (1 - \text{PADFACTOR} / 100))$$

ここでは次の内容を表しています。

**NIRBLOCKS** NI の必要スペースをブロック数で示します。

**NIRBYTES** NI の必要スペースをバイト数で示します (NIRBYTES の公式より)。

**ASSOBLKSIZE** ASSOR1 ブロック長。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

**PADFACTOR** ASSOR1 ブロックパディングファクタ。1~90 の値として表現するブロック長のパーセント。

**例**

次の各例では、ASSOR1 は 3380 デバイス上に格納されています。

**例 1**

ディスクリプタ AA は各レコードに 1 個の値を持ち、空値はありません。ファイルには、50 個の異なる AA 値が存在し、圧縮された値の長さの平均は 3 バイトです。

ISNSIZE=3

MAXISN=20000

PADFACTOR=10 (%)

```
NIRBYTES = 3 * 1 * 20,000 + 50 * (3 + 2) = 60,000 + 250 = 60,250 bytes
```

```
NIRBLOCKS = 60,250 / (2004 * (1 - 0.1)) = 33.41 = 34 blocks
```

### 例 2

ディスクリプタ BB は各レコードに 1 個の値を持ち、空値はありません。ファイルには 20,000 個の異なる BB 値が存在し、圧縮された値の長さの平均は 10 バイトです。

ISNSIZE=4

MAXISN=20000

PADFACTOR=10 (%)

```
NIRBYTES = 4 * 1 * 20,000 + 20,000 * (10 + 2) = 80,000 + 240,000 = 320,000 bytes
```

```
NIRBLOCKS = 320,000 / (2004 * (1 - 0.1)) = 177.42 = 178 blocks
```

### 例 3

ディスクリプタ CC は各レコードに平均 10 個のオカレンスと 3 個の空値をもった、空値省略のマルチプルバリュー (MU) フィールドです。ファイルには、およそ 300 個の異なる CC 値が存在し、圧縮された値の長さの平均は 4 バイトです。

ISNSIZE=3

MAXISN=10000

PADFACTOR=5 (%)

```
NIRBYTES = 3 * 7 * 10,000 + 300 * (4 + 2) = 210,000 + 1,800 = 211,800 bytes
```

```
NIRBLOCKS = 211,800 / (2004 * (1 - 0.05)) = 111.25 = 112
```

### 例 4

ディスクリプタ DD はピリオディックグループ内の空値省略フィールドです。DD は各レコードに平均 5 個の値を持ち、各レコードに平均 3 個の空値があります。ファイルには、10 個の異なる DD 値が存在し、圧縮された値の長さの平均は 5 バイトです。

ISNSIZE=4

MAXISN=10000

PADFACTOR=5 (%)

$$\text{NIRBYTES} = 4 * 2 * 10,000 + 10 * (5 + 2) = 80,000 + 70 = 80,070 \text{ bytes}$$

$$\text{NIRBLOCKS} = 80,070 / (2004 * (1 - 0.05)) = 42.06 = 43$$

## アッパーインデックス (UI)

ファイル内の各ディスクリプタに必要な UI スペースを見積るには、次の公式を使用します。

$$\text{UIRBYTES} = \text{NIRBLOCKS} * (\text{AVDESCLEN} + \text{ISNSIZE} + \text{RABNSIZE} + 1)$$

ここでは次の内容を表しています。

**UIRBYTES** UI の必要スペースをバイト数で示します。

**NIRBLOCKS** NI の必要スペースをブロック数で示します (NIRBLOCKS の公式より)。

**AVDESCLEN** ディスクリプタ値の平均圧縮長です。

**ISNSIZE** ファイル内の ISN 長 (3 または 4 バイト)。ISN 長は ADALOD パラメータの ISNSIZE に指定します。

**RABNSIZE** データベース内の RABN 長 (3 または 4 バイト)。RABNSIZE はデータベース定義時にデータベース内の全ファイルに指定されます。

**注意:** RABNSIZE は相対 Adabas ブロック番号の長さだけを参照します。ブロックサイズは参照しません。

## バイトからブロックへの変換

バイトをブロックに変換するには、次の公式を使います。結果は次のブロックに切り上げます。

$$\text{UIRBLOCKS} = \text{UIRBYTES} / (\text{ASSOBLKSIZE} * (1 - \text{PADFACTOR} / 100))$$

ここでは次の内容を表しています。

**UIRBLOCKS** UI に必要なスペースをブロック数で示します。

**UIRBYTES** UI に必要なスペースをバイト数で示します (UIRBYTES の公式より)。

**ASSOBLKSIZE** ASSOR1 ブロック長。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

**PADFACTOR** ASSOR1 ブロックパディングファクタ。1~90 の値として表現するブロック長のパーセント。

## 例

この例では、アソシエータは 3380 DASD 上に格納されています。したがって、ASSOR1 は各ブロックに 2004 バイトずつあります。

このファイルに必要な NI ブロック数は 45 バイトです。ディスクリプタ値の平均圧縮長は 3 バイトです。データベースには 3 バイト (24 ビット) RABN があり、ファイルには 3 バイト ISN があります。ASSOR1 ブロックパディングファクタは 5 (%) です。

```
UIRBYTES = 45 * (3 + 3 + 3 + 1) = 450
```

```
UIRBLOCKS = 450 / (2004 * (1 - 0.05)) = 450 / 1903.8 = 0.24 = 1 block( a minimum of 2 blocks can be allocated for the UI)
```

## アドレスコンバータ (AC)

ファイルに必要なアドレスコンバータスペースを見積るには、次の公式を使用します。計算結果は、次の完全なブロックに切り上げます。

```
ACBLOCKS = (MAXISN +!) * RABNSIZE / ASSOBLKSIZE
```

ここでは次の内容を表しています。

ACBLOCKS アドレスコンバータに必要なスペースをブロック数で示します。

MAXISN ファイルの MAXISN 設定値。

RABNSIZE データベース内の RABN 長 (3 または 4 バイト)。RABNSIZE はデータベース定義時にデータベース内の全ファイルに指定されます。

**注意:** RABNSIZE は相対 Adabas ブロック番号の長さだけを参照します。ブロックサイズは参照しません。

ASSOBLKSIZE アソシエータのブロックサイズです。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

## 例

アソシエータは 3380 DASD 上に格納されています。ASSOR1 は各ブロックに 2004 バイトずつあります。

### 例 1

MAXISN=2000000

RABNSIZE = 3

$$\text{ACBLOCKS} = (2,000,000 * 3) / 2004 = 6,000,000 / 2004 = 2994.01 = 2995 \text{ blocks}$$

## 例 2

MAXISN=2000000

RABNSIZE = 4

$$\text{ACBLOCKS} = (2,000,000 * 4) / 2004 = 8,000,000 / 2004 = 3992.02 = 3993 \text{ blocks}$$

## データストレージ

データストレージに必要なスペースを見積るには、次の公式を使用します。計算結果は、次の完全なブロックに切り上げます。

$$\text{DATASTORAGE} = \text{MAXISN} / ((\text{DSBLKSIZE} * (1 - (\text{PADFACTOR} / 100))) / \text{AVRECLEN})$$

ここでは次の内容を表しています。

**DATASTORAGE** データストレージに必要なスペースをブロック数で示します。

**MAXISN** ファイルの MAXISN 設定値。

**DSBLKSIZE** データストレージのブロックサイズを次の整数に切り下げた値です。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

**PADFACTOR** データストレージブロックパディングファクタ。1~90の値として表現するブロック長のパーセンテージ。

**AVRECLEN** 平均圧縮レコード長です。

## 例

MAXISN = 1000000

平均圧縮レコード長 = 50


モデル 3380 の DATA のブロックサイズ = 4820

データストレージのブロックパディングファクタ = 5 (%)

$$\text{DATASTORAGE} = 1,000,000 / ((4820 * (1 - 0.05)) / 50) = 1,000,000 / (4579 / 50) = 1,000,000 / 91 = 10,989.01 = 10,990 \text{ blocks}$$

## ワークスペース割り当ての見積り

WORK データセットを割り当てるときは、すべてのパーツに十分なスペースを割り当てる必要があります。許容可能な最小ワークスペースは300ブロックです。ADARUNの3つのパラメータを使用して、スペースを次のようにパート1~4に分割できます。


 **Note:** Adabas Transaction Manager バージョン7.5以降では、DDWORKR1のWORKパート4はサポートされていません。代わりに、DDWORKR4データセットを使用する必要があります。DDWORKR4は、WORKパート4と同じ目的で使用されるコンテナデータセットですが、クラスタのすべてのメンバと並行して使用できます。DDWORKR4データセットは、DDWORKR1以上のブロックサイズを使用して、通常の方法で割り当てとフォーマットを行う必要があります。少なくとも、データベースのクラスタのLPパラメータ、またはクラスタと同じ大きさのエクステントが確保される必要があります。

- ADARUN LPパラメータはWORKパート1のサイズを指定します。デフォルト設定は1000ブロック、最小は200です。ほとんどまたはまったく更新がないデータベースには500~1000ブロックが必要です。WORKパート1はRABN1から開始され、最後のRABNはLP値となります。
- WORKパート2のサイズ指定には、ADARUN LWKP2パラメータを使用することができます。LWKP2=0（デフォルト）の場合は、「**WORKパート2：検索コマンドの中間結果**」に示す公式を使ってAdabasが自動的にサイズを計算します。

WORKパート2は、WORKパート1の最終ブロックの次のブロックで開始されます。したがって、パート2の最初のRABNは次のようになります。

1 + LP

- ADARUN DTP=RMの場合、ADARUN LDTPパラメータはWORKパート4のサイズを指定します。LDTP=0（デフォルト）の場合、WORKパート4の長さはWORKパート1の長さ（ADARUN LP）と等しくなります。ゼロ以外の値が指定された場合、LPに指定された値よりも大きくなければなりません。小さい値が指定された場合、Adabasは値をLP値と等しくなるように変更します。

 **Note:** DDWORKR1のWORKパート4は、Adabas Transaction Manager バージョン7.5以降がインストールされている場合はサポートされていません。代わりに、DDWORKR4コンテナデータセットを使用する必要があります。

WORKパート4は、WORKパート2の最終ブロックの次のブロックで開始されます。したがって、パート4の最初のRABNは次のようになります。

1 + LP + LWKP2

- パート1と2、および4を割り当てた後、Adabasは残りのブロックをWORKパート3に割り当てます。パート3に少なくとも50ブロックが残るように、DDWORKR1データセットに十分なスペースを割り当てることが重要です。

## WORK パート 1：データ保護情報

並行して実行されるすべてのトランザクションのデータプロテクションエリアが、WORK パート 1 (LP) エリアの 1/4 に収まる必要があります。WORK パート 1 の適切なサイズを確認するための一般的なガイドラインを次に示します。

1. WORK パート 1 の合計サイズは、1 つの平均トランザクションに必要なと予測されるサイズ (バイト数) の 4 倍に、並行して実行されるトランザクションの最大数をかけた値である必要があります。この値を WORK ブロックサイズ (バイト数) -200 で割ってバイト数をブロック数に変換します。
2. 一部のトランザクションが非常に長い場合は、これらのトランザクション単独と、並行して実行されるその他すべての短いトランザクションを使用して、1 つの平均トランザクションのサイズを求めます。
3. 1 つの平均トランザクションのサイズは、トランザクション当たりの更新 (変更、追加、削除) の平均回数に、更新ごとに必要であると推測されるバイト数をかけて求めます。これに ET データ用と ET レコード用のスペースをバイト数で加算します。
4. 更新ごとに必要なサイズを求めるには、平均圧縮レコード長 (バイト数) に 4 をかけ (ビフォーイメージ、アフターイメージ、それぞれの DVT スペース)、プロテクションレコードヘッダーごとに 100 バイト (100 x 4) を加算します。

このガイドラインを公式で表すと次のようになります。

$$WK1 = ( 4 * TASIZE * TAP ) / ( BLKSIZE - 200 )$$

ここでは次の内容を表しています。

WK1 WORK パート 1 に必要なスペースをブロック数で示します。

TASIZE 単一平均トランザクションサイズをバイト数で示します。

TAP 実際に並行して実行する最大トランザクション数です。

BLKSIZE WORK ブロックサイズをバイト数で示します。

$$TASIZE = ( ( ( 4 * AVCRL ) + 400 ) * UPDTA ) + ETDATA + 100$$

ここでは次の内容を表しています。

AVCRL 平均圧縮レコード長をバイト数で示します。

UPDTA トランザクション当たりの平均更新回数です。

ETDATA ET データの平均長をバイト数で示します。

例

AVCRL = 300 バイト、UPDATA = 4、ETDATA = 200 バイトのとき

$$TASIZE = ( ( ( 4 * 300 ) + 400 ) * 4 ) + 200 + 100 = 6700 \text{ bytes}$$

TAP = 100、BLKSIZE = 5492 のとき

$$WK1 = ( 4 * 6700 * 100 ) / ( 5492 - 200 ) = 506.46 \text{ blocks}$$

**WORK パート 2：検索コマンドの中間結果**

WORK パート 2 エリアに必要なスペースを見積るには、次の公式を使用します。計算結果は、次の完全なブロックに切り上げます。

$$WORK2 = 22 + 2 * ((4 * RECORDS) / (BLKSIZE - 16))$$

ここでは次の内容を表しています。

**WORK2** WORK パート 2 に必要なスペースをブロック数で示します。

**RECORDS** 大半のレコードをもったファイル内のレコード数。この数値は次のようになります。


$$TOPISN - MINISN + 1$$

上記の意味は次に示すとおりです。

TOPISN	現在ファイルで使用されている最高の ISN です。
MINISN	ファイルで使用されている最低の ISN です。

MINISN 値は ADACMP/ADALOD の MINISN パラメータで指定され、デフォルトは 1 です。ADAREP コーティリティを使えば、データベース内のファイルの TOPISN と MINISN 値を表示できます。

**BLKSIZE** WORK データセットが格納されているデバイスのブロックサイズです。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

 **Note:** WORK パート 2 の各ブロックにつき、Adabas 内部テーブルは 1 バイトのストレージを必要とします。



**例**

データベース内の最大ファイルのレコード数は 500,000 です。WORK データセットは 3380 デバイス上に格納されています。

```
WORK2 = 22 + 2 * ((4 * 500,000) / (5492 - 16)) = 752.46 = 753 blocks
```

**WORK パート 3：検索コマンドの結果の ISN リスト**

Adabas はパート 1（データ保護情報）とパート 2（中間結果）エリアを割り当てた後、残りのワークスペースを WORK パート 3（ISN 結果リスト）に割り当てます。

パート 3 の最小必要スペースは 50 ブロックです。

このエリアに供給したスペースが不十分であると、現在 ISN リストによって使用されたスペースが解放されるまで、Adabas はこれ以上の検索コマンドを実行できない場合があります。WORK パート 3 エリアに必要なスペースを見積るとき、次の要因について考慮する必要があります。

- 同時に処理される検索コマンドの数（各 ISN リストは別のブロックに格納されます）、および ISN 結果リストの予想サイズ（ファイルに指定された ISNSIZE に関係なく、各 ISN は 4 バイト必要です）。
- 同時に保持される SAVE ISN LIST オプション付きの直前の検索コマンドからセーブされた ISN 結果リストの数。
- 結果として Adabas によって要求されるメモリ量（このエリアに割り当てられた各ブロックは 4 バイトのメモリを要求します）。

**例**

セッション中にコマンド当たり平均 25 個の結果 ISN を伴った検索コマンドを最大 100 コマンドまで同時に処理します。

このとき、WORK パート 3 エリアに 100 ブロック必要となります。

**WORK パート 4：分散トランザクション処理に関連するデータ**

**Note:** DDWORKR1 の WORK パート 4 は、Adabas Transaction Manager バージョン 7.5 以降がインストールされている場合はサポートされていません。代わりに、DDWORKR4 データセットを使用する必要があります。DDWORKR4 は、WORK パート 4 と同じ目的で使用されるコンテナデータセットですが、クラスタのすべてのメンバと並行して使用できます。DDWORKR4 データセットは、DDWORKR1 以上のブロックサイズを使用して、通常の方法で割り当てとフォーマットを行う必要があります。少なくとも、データベースのクラスタの LP パラメータ、またはクラスタと同じ大きさのエクステン트가確保される必要があります。

WORK パート 4 は、分散処理に関連する、いくつかのグローバルトランザクションに関する情報を維持します。例えば、コミット処理のフェーズ 1 中に、グローバルトランザクションのプロテクションデータを WORK パート 1 に格納できなくなった場合、そのデータを WORK パート 1 から WORK パート 4 にコピーできます。

WORK パート 4 のオーバーフローが未解決の場合、ニュークリアスはトランザクションを強制的に終了することができます。これは、トランザクション ID (XID) およびローカルトランザクションステータス情報以外の WORK パート 4 をクリアします。

WORK パート 4 に必要なサイズは、データベースに対して実行するアプリケーションと、システムの負荷に依存します。Adabas Transaction Manager バージョン 7.4 以前では、最初はサイズ LP/4 が適切です。Adabas Transaction Manager 7.5 以降では、最小サイズ 8 ブロックを指定する必要があります。ブロックの最大数は、DDWORK4 データセットのサイズを 8 で割った数です。

WORK パート 4 に維持される情報は、現時点では異なるエリアに移動できないので、セッションとセッションの間にも WORK パート 4 のサイズを次のように変更することができます。

- 直前のセッションで全く使用されなかった場合、WORK パート 4 のサイズを減らすことができます。
- 直前のセッションで使用された場合、WORK パート 4 のサイズを増やすことができます。

## ソート

次の公式は、1つのディスクリプタのすべての値をソートするために使われるソートデータセットスペースの見積り式です。複数のディスクリプタの場合は、連続的にソートされます。すなわち、最初のディスクリプタのすべての値がソートされ、次に2つめのディスクリプタのすべての値がソートされ、以後同様に処理されます。したがって、最も大きいディスクリプタのソート用のスペースを見積れば、すべてのディスクリプタをソートするのに十分です。

ソートエリアに必要なスペースを見積るには、次の公式を使用します。

```
DESCSPACE = (AVDESCLEN + (1 + ISNSIZE)) * NUMRECS * AVPEOCCUR * AVMVOCCUR
```

ここでは次の内容を表しています。

- |           |   |
|-----------|---|
| DESCSPACE | 必要なディスクリプタスペースの合計をバイト数で示します。  |
| AVDESCLEN | 平均圧縮ディスクリプタ長をバイト数で示します。   |
| ISNSIZE   | 使用される ISN のサイズ (3 または 4) です。  |
| NUMRECS   | レコード数です。  |
| AVPEOCCUR | ディスクリプタがピリオディックグループ内に存在する場合、ピリオディックグループの平均オカレンス数です。それ以外の場合は、この値を 1 に設定します。      |
| AVMVOCCUR | ディスクリプタがマルチプルバリュースフィールドである場合、マルチプルバリュースフィールドの平均オカレンス数です。それ以外の場合は、この値を 1 に設定します。 |

## ワークプールサイズ

ワークプールに必要なスペースを見積るには、次の公式を使用します。

$$\text{LWPAVAIL} = \text{LWPSIZE} - 1216 - (32 * \text{SORTDEVTRKS}) - \text{SORTDEVBSIZ}$$

ここでは次の内容を表しています。

- LWPAVAIL**      ワークプールスペースの使用可能な部分をバイト数で示します。
- LWPSIZE**        ワークプールの合計サイズをバイト数で示します（ユーティリティの LWP パラメータ値）。
- SORTDEVTRKS**   シリンダ当たりのソートデバイストラック数。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム（z/OS、VSE、z/VM、または BS2000）の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。
- SORTDEVBSIZ**    ソートデバイスブロックサイズをバイト数で示します。

## ソートパーシャルシーケンス

ソートパーシャルシーケンスに必要なスペースを判断するには、次のいずれかの計算式を使用します。使用する計算式は、DESCSPACE 値（必要なディスクリプタスペースの合計）の計算に使った AVDESCLEN 値（平均ディスクリプタ長）に応じて異なります。

### ■ AVDESCLEN < 12 のとき

$$\text{LENGSEQ} = (\text{LWPAVAIL} * (\text{AVDESCLEN} + (1 + \text{ISNSIZE}))) / 2$$

ここでは次の内容を表しています。

- LENGSEQ**        ソートパーシャルシーケンスの長さ。
- LWPAVAIL**        使用可能なワークプールスペース。
- AVDESCLEN**      平均圧縮ディスクリプタ長をバイト数で示します。
- ISNSIZE**         使用される ISN のサイズ（3 または 4）です。

### ■ AVDESCLEN ≥ 12 のとき

$$\text{LENGSEQ} = (\text{LWPAVAIL} * 2) / 3$$

ここでは次の内容を表しています。

LENGSEQ ソートパーシャルシーケンスの長さ。

LWPAVAIL 使用可能なワークプールスペース。

### デバイスサーフェイス

次の公式を使ってデバイスサーフェイス数を計算します。計算結果は、次の整数値に切り上げます。

```
SURFACES = (DESCSPACE / LENGSEQ) / SORTDEVTRK
```

ここでは次の内容を表しています。

SURFACES ソートスペースに必要なサーフェイス数。結果は次の整数値に切り上げます。

DESCSPACE 必要なディスクリプタスペースの合計をバイト数で示します。

LENGSEQ ソートパーシャルシーケンスの長さ。

SORTDEVTRKS シリンダ当たりのソートデバイストラック数。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

### ソートサイズの見積り

LENGSEQ と SURFACES で計算された中間値を使用して、ソートサイズを次のように計算します。

```
SORTSIZE = (SURFACES * SORTDEVTRKS * LENGSEQ * 2) / (SORTDEVBSIZ - 4)
```

ここでは次の内容を表しています。

SORTSIZE ソートエリアの見積りをブロック数で示します。この値は、次の完全なシリンダに切り上げる必要があります。

SURFACES ソートスペースに必要なサーフェイス数で、以前に計算して切り上げた値。

SORTDEVTRKS シリンダ当たりのソートデバイストラック数。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

LENGSEQ ソートパーシャルシーケンスの長さ。

SORTDEVBSIZ ソートデバイスブロックサイズをバイト数で示します。

## ソートディスクリプタ数

上記の計算で得られた SORTSIZE スペースでソートできるディスクリプタ数の見積りを出すには、次の公式を使用します（DESCSPACE の計算時と同じディスクリプタ定義であると仮定します）。

$$\text{DESCOUNT} = \text{SURFACES} * \text{SORTDEVTRKS} * \text{LENGSEQ} / (\text{AVDESCLEN} + (1 + \text{ISNSIZE}))$$

ここでは次の内容を表しています。

DESCOUNT	上記の計算で得られた SORTSIZE スペースでソート可能な、以前の DESCSPACE 計算で定義されたディスクリプタ数。
SURFACES	ソートスペースに必要なサーフェイス数で、以前に計算して切り上げた値。
SORTDEVTRKS	シリンダ当たりのソートデバイストラック数。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム（z/OS、VSE、z/VM、または BS2000）の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。
LENGSEQ	ソートパーシャルシーケンスの長さ。
AVDESCLEN	平均圧縮ディスクリプタ長をバイト数で示します。
ISNSIZE	使用される ISN のサイズ（3 または 4）です。

## ステップ 2：データベースの割り当て

- オペレーティングシステムの標準の手順で、次の Adabas コンポーネントのデータセットの割り当てを行います。
  - Adabas ニュークリアスに必須
    - アソシエータ (ASSO)
    - データストレージ (DATA)
    - ワークエリア (WORK1)
    - ワークエリア (WORK4)、Adabas Transaction Manager 7.5 以降がインストールされている場合。
  - 数種の Adabas ユーティリティに必須
    - ソートエリア (SORT)
    - 一時エリア (TEMP)
  - オプションのログ（オプションであるが使用した方がよい）
    - デュアルまたはマルチコマンドログ (CLOG)
    - デュアルまたはマルチプロテクションログ (PLOG)
    - リカバリログ (RLOG)

通常、ASSO、DATA、WORK は1つのオペレーティングシステムのデータセットとして割り当てます。しかし、アソシエータおよびデータストレージはそれぞれ 99 個の別々のデータセットに割り当てることができます。これらのデータセットは同じデバイスタイプ上に割り当てても、異なるデバイスタイプ上に割り当ててもかまいません。物理エクステンツの実際の最大数は 99 より少ない場合があります。定義できる最大数は、最初のアソシエータデータセット (DDASSOR1) のブロックサイズから求められるからです。

- 競合を最小にし、データベース I/O 動作をハードウェアチャンネル間でより均等に分散するために、ASSO、DATA、WORK、PLOG、およびRLOGの各データセットは異なる物理ボリューム上に配置することをお勧めします。ボリュームを2つだけ使えるときは、ASSOデータセットを1つのボリュームに、DATAデータセットとWORKデータセットをもう1つのボリュームに割り当てます。

PLOG I/O 動作の後には、必ず WORK I/O 動作が伴うので、WORK データセットと PLOG データセットは、異なるボリュームにする必要があります。

RLOG データセットは、必ず同じタイプの別のデバイスに配置します。

TEMP と DATA、また SORT と ASSO を分離することでディスクのアクセス時間を大幅に短縮できる可能性があります。100,000 以上のレコードを含むファイルをロードするときは、SORT を2つのボリュームに分割すると、ディスクアームの移動が少なくなります。

- ディスクスペースの割り当ては、フォーマットユーティリティ (ADAFRM) のジョブ制御 (JCL/JCS または VM CONTROL ミニディスク) に指定します。詳しい情報と例は『Adabas ユーティリティマニュアル』を参照してください。

## 例

### (1) 2つのボリュームを使ったデータベースの割り当て例

ボリューム 1	ボリューム 2			
ASSO	DATA			
TEMP	WORK			
PLOG1	SORT			
PLOG2				

## (2) 3つのボリュームを使ったデータベースの割り当て例

ボリューム1	ボリューム2	ボリューム3		
ASSO	DATA	WORK		
PLOG1	SORT	TEMP		
	PLOG2			

## (3) 大きなファイルをロードする場合のデータベースの割り当て例

ボリューム1	ボリューム2	ボリューム3	ボリューム4	ボリューム5
ASSO	DATA	DATA	DATA	SORT (後半)
	PLOG1	PLOG2	SORT (前半)	WORK
		TEMP		


## パフォーマンスの考慮事項

プロテクションログファイルへのハードウェア圧縮 (IDRC) の使用は避けるべきです。なぜなら、ADARES ユーティリティの BACKOUT 機能は、圧縮データを処理するとき、z/OS 下で2倍以上の実行時間がかかるからです。加えて、VSE または z/VM の各システム上の圧縮データへの BACKOUT 機能のサポートはないからです。

## ステップ3：データベースのフォーマット

最初のファイルがデータベースにロードされる前に、ADAFRM ユーティリティを使用して ASSO、DATA、および WORK の各データセットをフォーマットします。ADAFRM ユーティリティについては、『Adabas ユーティリティマニュアル』を参照してください。

TEMP および SORT は、これらを必要とする Adabas ユーティリティを実行する前にフォーマットする必要があります。一時データセットを割り当ててフォーマットし、ユーティリティの実行後に削除するか、または繰り返し使用できるように、恒久データセットを割り当てることができます。

 **Note:** Adabas Recovery Aid (RLOG を含む) を使用する場合は、一時データセットをすべてカタログする必要があります。Adabas Recovery Aid を実行している場合、原則として、アソシエータデータセットが必要なジョブで一時データセットをカタログします。

ロギングを行う最初のセッションを開始する前に CLOG、PLOG、RLOG の各データセットをフォーマットします。

## ステップ4：データベースパラメータの定義

すべてのデータベースコンポーネントを物理的に割り当ててフォーマットした後、ADADEFユーティリティを使ってデータベース ID、最大ファイル数、システムファイル割り当てなどのデータベースパラメータを定義します。

ASSO、DATA、および WORK の各データセットのサイズは、ADADEF DEFINE パラメータで定義します。スペースの割り当ては、Adabas へのサイズ定義とは異なることに注意する必要があります。データセットは、Adabas に定義する前に割り当ててフォーマットする必要があります。その他のデータセットのサイズは、次のように Adabas に定義します。

- TEMP と SORT：これらのデータセットを使用するユーティリティを実行するときに定義します。
- CLOG と PLOG：ニュークリアセッションの開始時に ADARUN パラメータとともに定義します。
- RLOG：ロギングの開始時に、ADARAI ユーティリティの PREPARE 機能を使用して定義します。



**Note:** 各ログデータセット（CLOG、PLOG、または RLOG）は、16,777,215 (x'FFFFFF') ブロック/RABN に制限されます。



## 24 データベースのスペース管理

---

■ Adabas 物理エクステント .....	146
■ Adabas 相対ブロック番号 (RABN) .....	147
■ Adabas 論理エクステント .....	149
■ Adabas スペース割り当ておよび割り当て解除 .....	150
■ データベースステータスレポートを使用したスペース使用状況の監視 .....	159
■ スペース使用に関する問題点とその対応策 .....	159

このchapterではデータベースのスペース管理に関する情報を DBA に提供します。取り上げる項目は次のとおりです。

- Adabas 物理エクステントおよび論理エクステントの説明
- Adabas 相対ブロック番号 (RABN) の説明
- Adabas ニュークリアスおよび Adabas ユーティリティの行うスペース割り当て／割り当て解除
- データベースステータスレポートを使用してデータベーススペースの使用状況を監視する方法
- 潜在的なスペース利用問題と推奨する対応策の説明

この情報は次の項目で構成されています。

## Adabas 物理エクステント

---

Adabas 物理エクステント (データベースのコンテナ) は、データベースの定義時にデータベースコンポーネント (アソシエータ、データストレージ、ワーク) に割り当てた物理ブロックの集合です (ADADEF ユーティリティの ASSOSIZE、DATASIZE、WORKSIZE の各パラメータを参照)。

物理エクステントに対するスペースは、使用するオペレーティングシステムの標準の割り当て手続きを使用して割り当てます。

Adabas 物理エクステントは、1 次エクステントだけから成る単一のオペレーティングシステムエクステント内に割り当ててもよいし、1 次エクステントと 1 つまたは複数の 2 次エクステントとを合わせて割り当ててもかまいません。2 次エクステントは 1 次エクステントと連続している必要はなく、また、互いに連続している必要もありません。

Adabas 物理エクステントは、1 つの物理ボリューム内に含まれてもよいし複数ボリュームにまたがってもかまいません。アソシエータとデータストレージはそれぞれ約 99 個までの Adabas 物理エクステントを含むことができます。ただし、すべてのアソシエータ、データストレージ、およびデータストレージスペーステーブル (DSST) のエクステントのエクステント記述がジェネラルコントロールブロック (GCB) に収まる必要があるため、実際の最大数は、これよりも少なくなる可能性があります。例えば、標準の 3390 デバイスタイプでは、アソシエータ、データストレージ、および DSST の各エクステントの個数は 75 を超えることができます (他のタイプのエクステント数が少ない場合は、1 つのタイプのエクステントをより多く持てます)。

## Adabas 相対ブロック番号 (RABN)

Adabas 情報は割り当てられたスペースにブロック単位で格納されます。ブロックの大きさは次の 2 つの要因に応じて決定します。

- ブロックの物理デバイス
- ブロックに割り当てられた Adabas コンポーネント

例えば、Adabas で使用されるデフォルトのデバイスタイプは IBM 3380 ディスクです。別のデバイスタイプを指定しない限り、多くのユーティリティやオペレーティングパラメータでこのデバイスタイプが想定されます。

スペース割り当てを行うときは、アソシエータ (ASSO)、データストレージ (DATA)、ワークエリア (WORK)、ロギングエリア (PLOG、CLOG、RLOG)、あるいはソートエリア、一時エリアとして割り当てる必要があります。ASSO に割り当てた 3380 ブロックには 2004 バイトが含まれています。一方、DATA に割り当てた 3380 ブロックには 4820 バイトが含まれています。各デバイスタイプと Adabas コンポーネントのブロックサイズは事前に定義されています。デバイスタイプおよびコンポーネント別にブロックサイズのリストを確認するには、該当するシステムプラットフォーム (z/OS、VSE、z/VM、または BS2000) の『Adabas インストールマニュアル』に記載されている「デバイスとファイルについて」を参照してください。

Adabas のブロックサイズはハードウェアによって固定されていませんが、物理ブロック (FBA) デバイスの説明に合わせて物理ブロックと呼んでいます。Software AG では、リリース間でデバイスタイプごとにブロック定義の一貫性を保とうとしています。しかし、Adabas の機能拡張に合わせて、コンポーネントタイプのブロックサイズが変わる場合もあります。したがって、新しい Adabas リリースを実行する前に、特定 Adabas コンポーネント (PLOG、ASSO など) の再フォーマットが必要な場合があります。

Adabas はデータベースコンポーネント (アソシエータ、データストレージ、ワーク) 内の各物理ブロックをその *Adabas 相対ブロック番号 (RABN)* すなわちコンポーネントの先頭からの相対位置で示します。RABN は、各データベースコンポーネント内で 1 から順に昇順で割り当てられます。複数の物理エクステントが使用されるとき、RABN の割り当ては物理エクステント全体を通して行われます。

アソシエータ、データストレージ、およびワークの各コンポーネントの最初の物理エクステントの最初のトラックは使用されません。2 番目以降の物理エクステント、また TEMP、SORT、CLOG、および PLOG のすべてのエクステントの最初のトラックは使用されます。

ASSO と DATA に割り当て可能な RABN 数は、データベースの定義時に指定された RABNSIZE パラメータに応じて異なります。RABNSIZE はデータベース中の Adabas 相対ブロック番号の長さを意味します (ブロック自体の長さではありません)。

- RABNSIZE=3 のとき、ブロック番号は 24 ビット (3 バイト) であり、最大 RABN 数は 16,777,215 です。

- RABNSIZE=4 のとき、ブロック番号は 31 ビット（4 バイト）であり、最大 RABN 数は 2,147,483,646 です。

外部ストレージの特定の物理ユニット（トラック／シリンダ／ボリューム）に格納できる Adabas ブロックの数は、データベースコンポーネントとデバイスタイプによって異なります。

例えば、「サポートされるデバイスタイプ」に示す情報を使用して、特定のボリュームに格納できるブロック数をこの後の例に示すように計算できます。また、各 VOLSER に格納する RABN 範囲は Adabas Online System レポート機能を使用して簡単に決定できます。

**例1：**アソシエータデータベースコンポーネント、モデル **3380**（ボリュームで **880** 個のシリンダを使用可能と仮定）

```
number of ASSO blocks = blocks/track * tracks/cylinder * number of cylinders
= 19 * 15 * 880 = 250,800 Associator blocks
```

アソシエータの最初の物理エクステンットの最初のトラック用に 19 個のブロックを差し引きます。したがって、最初のアソシエータボリュームには最大 250,781 個のブロックを格納できます。

**例2：**データストレージデータベースコンポーネント、モデル **3370**（ボリュームで **748** 個のシリンダが使用可能と仮定）

```
number of DATA blocks = blocks/track * tracks/cylinder * number of cylinders
= 10 * 12 * 748 = 89,760 Data Storage blocks
```

データストレージの最初の物理エクステンットの最初のトラック用に 10 個のブロックを差し引きます。したがって、最初のデータストレージボリュームには最大 89,750 個のブロックを格納できます。

## Adabas 論理エクステント

Adabas の論理ファイルエクステントは、Adabas ニュークリアスまたは Adabas ユーティリティが割り当てる連続した RABN 群です。データベースにロードする各ファイルについて、次のタイプの Adabas 論理エクステントの各々の最小 1 つがファイルに割り当てられます。

論理エクステント	割り当てる物理エクステント
データストレージ	データストレージ
アドレスコンバータ	アソシエータ
ノーマルインデックス	アソシエータ
アッパーインデックス	アソシエータ

ファイルメンテナンスの結果、追加スペースが必要になった場合に、追加論理エクステントが Adabas ニュークリアスまたは Adabas ユーティリティによって割り当てられます。

任意のファイルに最大 2 つのアドレスコンバータを割り当てることができます。1 つはファイル用、もう 1 つはファイル内のスパンドレコード用です。スパンドレコードを使用していない場合、割り当てられるアドレスコンバータは 1 つだけです。

定義できる論理ファイルエクステントの合計数には、ファイルのすべてのアドレスコンバータ、データストレージ、ノーマルインデックス、アッパーインデックスのエクステント情報がファイルコントロールブロック (FCB) 内に収まる必要があるという制約があります (エクステントの情報は、FCB の可変のセクション内に格納されます)。例えば、標準の 3390 デバイスタイプの場合、ファイルはタイプごとに 40 を超えるエクステントを持つことができます (他のタイプのエクステント数が少ない場合、1 つのタイプのエクステント数はより多くすることができます)。

Adabas ニュークリアスの起動時に FCB が確認されます。FCB スペースが、さらに 4 つのエクステントに不十分である場合は、Adabas ニュークリアスによって、ファイルのリオーダを指示するメッセージが出力されます。また、各タイプの最後のファイルエクステントの空き ISN またはブロック数が 5 つ以下の場合、ニュークリアスによってファイルがユーティリティだけで使用されるようにロックされます。通常のユーザーがファイルにアクセスしようとすると、レスポンスコード 17 または 48 が返されます。

ADAREP ユーティリティレポートの [Contents of the Database] セクションでは、FCB スペースが、さらに 10 個のエクステントには不十分であるファイルにフラグが付けられます。レポートの [File Information] セクションには、各ファイルに追加で確実に構築できるファイルエクステントの数が出力されます。

## Adabas スペース割り当ておよび割り当て解除

---

このセクションでは、全 Adabas スペースの割り当ておよび割り当て解除の手順の概略を示します。これらの手順をよく理解すれば、正確で効率的なデータベーススペース管理が確実なものとなるはずです。

### フリースペーステーブル

割り当てに利用できる全スペースはフリースペーステーブル (FST) に確保されます。このテーブルには、現在いかなるファイルにも割り当て可能な全 RABN エクステントが含まれています。

### Adabas ニュークリアスによるスペース割り当て

レコードの追加、更新コマンドの処理中に Adabas ニュークリアスは次のファイルコンポーネントに追加エクステントを割り当てなければならないことがあります。

- アドレスコンバータ
- ノーマルインデックス
- アップーインデックス
- データストレージ

### アドレスコンバータ (AC)

アドレスコンバータのサイズは最初は ADALOD ユーティリティの MAXISN パラメータで定義します。アドレスコンバータはブロック単位で格納されるため、実際予測される最大の ISN はわずかに大きくなります。

#### 例

モデル 3380 でブロック当たり 668 個のエントリがある場合 (2004/3) に RABNSIZE=3、MAXISN = 5000 と指定すると、ブロック数は 8 個になります。したがって、予測される最大の ISN (さらに拡張する前) は 5343 ( $668 * 8 - 1$ ) です。

モデル 3380 でブロック当たり 501 個のエントリがある場合 (2004/4) に RABNSIZE=4、MAXISN = 5000 と指定すると、ブロック数は 10 個になります。したがって、予測される最大の ISN は 5009 ( $501 * 10 - 1$ ) です。

N1 コマンドの実行中に Adabas ニュークリアスがファイルに追加エクステントを必要とした場合、割り当てルーチンは現サイズの 25 % の新規エクステントを割り当てようとします。

- 25 % ~ 28 % の未使用エクステントが、フリースペーステーブル (FST) 内にあれば、ただちにそのスペースを使用します。
- FST に大きなエクステントしかなければ、25 % の新規エクステントが取られます。

- FST に小さなエクステントしかない場合には、利用できる最大のエクステントが使用されま  
す。
- 追加の AC エクステントが必要であり、最大エクステントがすでに割り当てられている場合、  
Adabas はコール元プログラムに適切なレスポンスコードを返します。
- ファイルに "1 AC エクステントのみ" という属性がある場合（つまり、ファイルが拡張ファイ  
ルの場合）、第 2 の AC エクステントを割り当てようとするとレスポンスコードが返されま  
す。

### ノーマルインデックス (NI) 、アッパーインデックス (UI) データストレージ (DS)

新しいエクステントを割り当てる場合は、次の公式を使用します。

$$Z1 = \text{MIN} \left( 2 * B, (E-U) * B/U \right)$$

$$Z = \text{MIN} \left( \text{MAX}(Z1, B/8 + 10), 1000000 \right)$$

ここでは次の内容を表しています。

B 現在割り当てられているブロック数

E 予測される最大 ISN

U 現在割り当てられている最大 ISN

FST で見つかったエクステントが前のエクステントの終わりと連続する場合、そのエクステント  
は最大 Z ブロックに割り当てられます。

FST でそのようなエクステントが見つからなかった場合、次のとおりになります。

- Z から  $9 * Z/8$  までの範囲にエクステントが見つかった場合はそれが割り当てられます。
- $9 * Z/8$  ブロックを超えるエクステントが見つかった場合は、Z ブロックの新しいエクステント  
が割り当てられます。
- FST 内の最長のエクステントが新しいエクステントとして割り当てられます。

MAXNI、MAXUI、または MAXDS のいずれかのパラメータが現在のファイルに指定されてい  
る場合、ニュークリアスによって、それぞれ NI、UI、または DS に指定された最大ブロック数  
以下が割り当てられます。

### ADADBS ユーティリティのスペース割り当て

#### アソシエータ、データストレージの ADD/INCREASE

アソシエータまたはデータストレージの物理エクステントがすべて使われた場合には、ADD 機能または INCREASE 機能（Adabas Online System または ADADBS ユーティリティ）を使用して、追加の物理スペースを定義できます。

ADD 機能では、アソシエータまたはデータストレージに追加データセットを割り当てる必要があります。新規データセットは現在使用中のデバイスタイプと同じデバイスタイプまたは異なるデバイスタイプに配置できます。アソシエータもデータストレージもそれぞれ 99 個以下のデータセットで構成されます。ただし、定義可能な物理エクステントの最大数は、最初のアソシエータデータセット（DDASSOR1）のブロックサイズに依存するため、実際の最大数はこれより小さくなります。例えば、標準の 3390 デバイスタイプでは、アソシエータ、データストレージ、および DSST の各エクステントの個数は 75 を超えることができません（他のタイプのエクステント数が少ない場合は、1つのタイプのエクステントをより多く持てます）。

INCREASE 機能は既存データセットを物理的に拡張します。新しいスペースは、最初に ADAFRM ユーティリティを使用してフォーマットする必要があります。INCREASE 機能を使用する回数に制限はありません。

ADD 機能の後、新しいデータセットを使用するにはその前に ADAFRM ユーティリティを使用してフォーマットする必要があります。また、適切な変更をすべての Adabas ジョブコントロールに対して行う必要があります。詳細については、『Adabas オペレーションマニュアル』を参照してください。

データストレージを 4 回増加したら、ADAORD ユーティリティの REORASSO 機能を実行して、エクステントを 1 つにするためにデータストレージスペーステーブル（DSST）をリオーダし、データストレージで 4 回以上増加できるようにする必要があります。

フォーマットまたはリオーダを可能にするため、ニュークリアスセッションは、ADADBS の ADD 操作または INCREASE 操作の後に自動的に停止します。

#### ALLOCATE 機能

ALLOCATE 機能（Adabas Online System または ADADBS ユーティリティ）を使用して次のファイルコンポーネントに特定サイズのエクステントを割り当てることができます。

- データストレージ
- アドレスコンバータ
- ノーマルインデックス
- アップパーインデックス

開始RABNを指定することで、エクステントを割り当てる場所を指定することも可能です。

この機能を使用すれば、Adabas が自動的に割り当てを行うのではなく、DBA がファイルのサイズを予測し、特定サイズのエクステントを割り当てることができます。これにより、Adabas が小さすぎたり大きすぎたりするエクステントを割り当てるのを防ぐことができま



す（ADALOD ユーティリティ参照）。アクセスするファイルの実際の MAXNI/MAXUI および MAXDS の値はチェックされません。

#### DEALLOCATE 機能

DEALLOCATE 機能（Adabas Online System または ADADBS ユーティリティ）を使用して、次のファイルコンポーネントに割り当てられているエクステンツの割り当てを解除できます。

- データストレージ
- アドレスコンバータ
- ノーマルインデックス
- アップパーインデックス

開始RABNを指定することで、スペース割り当て解除をどこから始めるかを指定することができます。割り当てを解除されたスペースはフリースペーステーブル（FST）に戻されます。

#### DELETE 機能

DELETE 機能（Adabas Online System または ADADBS ユーティリティ）は、既存ファイルをデータベースから削除します。ファイルに割り当てられていたスペースは、フリースペーステーブルに戻され再利用可能となります。DELETE では、完全な拡張ファイルだけを削除できます。

#### RECOVER 機能

RECOVER 機能（Adabas Online System または ADADBS ユーティリティ）を使用して、異常終了した ADAINV ユーティリティまたは ADALOD ユーティリティの実行中に割り当てられたスペースを回復できます。回復したスペースは、フリースペーステーブルに戻され、再利用可能となります。

#### REFRESH 機能

REFRESH 機能（Adabas Online System または ADADBS ユーティリティ）は、ファイルステータスをレコード 0 件状態にし、各ファイルコンポーネントへの割り当てを 1 エクステンツに設定します。第 1 エクステンツ以外の追加エクステンツはフリースペーステーブルに戻されます。

#### RELEASE、UNCOUPLE 機能

RELEASE および UNCOUPLE 機能（Adabas Online System または ADADBS ユーティリティ）は、インバーテッドリストまたは物理カップリングリストを削除します。リストに使用されていたスペースは、ADAORD ユーティリティでのみ回復できます。拡張ファイルのディスクリプタを解放する場合、各成分ファイルを個別に解放する必要があります。拡張ファイルのディスクリプタを解放しているとき、ADADBS ではメッセージを表示します。

## ADAINV ユーティリティのスペース割り当て

### COUPLE/INVERT 機能

COUPLE 機能と INVERT 機能（Adabas Online System または ADAINV ユーティリティ）を使用すると、NISIZE ファイルコンポーネントに追加ブロックが割り当てられる場合があります（DSSIZE または MAXISN はなし）。これは、入力データの処理中に使用可能なスペースが一杯になった場合に発生します。

この場合、ニュークリアスによる削除中に解放されたインデックスブロックがあった場合、これらのブロックが再利用されます。フリースペーステーブルにサイズが M1 と M2 の範囲内のブロックがあれば利用されます。

M1 および M2 は次のように計算します。

$$M2 = M1 + M1/8$$

$$M1 = \text{MAX} (A2, \text{NIB}/4 + \text{KZ})$$

ここでは次の内容を表しています。

KZ ZAP 可能な値（デフォルト = 10）

NIB 使用中の NI ブロック数

および

$$A2 = \text{MIN} (A1, \text{NIB} * 2)$$

$$A1 = \text{IUN} * \text{NIB}/\text{IUS}$$

ここでは次の内容を表しています。

IUN 未使用 ISN の数

IUS 使用済 ISN の数

拡張ファイルのディスクリプタをインバートする場合、各成分ファイルを個別にインバートする必要があります。拡張ファイルのディスクリプタをインバートしているとき、ADAINV はメッセージを表示します。

## ADALOD ユーティリティのスペース割り当て

### LOAD 機能

ADALOD ユーティリティの LOAD 機能はファイルをデータベースにロードするために使用します。

### DSSIZE パラメータ

ADALOD の実行の開始時に DSSIZE パラメータに指定されたブロック数またはシリンダ数分が割り当てられ、先頭の DS エクステントに割り当てられます。

DSRABN パラメータまたは DSDEV パラメータは特定の RABN またはデバイスに割り当てを行うために使用します。

入力データの処理中に最初に割り当てられたエクステントが一杯になった場合には、フリースペーステーブル内でサイズが M1 と M2 の範囲のフリーブロックの検索が行われます。

M1 および M2 は次のように計算します。

$$M2 = M1 + M1/8$$

$$M1 = \text{MAX} (A2, \text{DSB}/4 + KZ)$$

ここでは次の内容を表しています。

KZ ZAP 可能な値 (デフォルト = 10)

DSB 使用中の DS ブロック数

および

$$A2 = \text{MIN} (A1, \text{DSB} * 2)$$

$$A1 = \text{IUN} * \text{DSB}/\text{IUS}$$

ここでは次の内容を表しています。

IUN 未使用 ISN の数

IUS 使用済 ISN の数

フリースペーステーブル内に十分なスペースが見つかり、そのスペースがすでに割り当てられているエクステントのすぐ後ろに続いている場合には、そのスペースがエクステントの終わりに追加されます。この場合には、新しいエクステントは割り当てられません。

新しいエクステントが必要な場合には、フリースペーステーブルがチェックされ M1 から M2 のサイズを満たすだけの数のブロックが新規エクステントとして取られます。99 エクステントまで可能です。ただし、定義可能な物理エクステントの最大数は、最初のアソシエータデータセッ

ト (DDASSOR1) のブロックサイズに依存するため、実際の最大数はこれより小さくなります。スペースがなければ ADALOD はエラーメッセージを出して終了します。

定義可能になった論理ファイルエクステントの最大数は、アソシエータの第 1 データセット (DDASSOR1) のブロックサイズから決定されます。エクステントの情報は、FCB の可変セクション内に格納されます。使用された FCB サイズがアソシエータデータセットのブロックサイズに到達するまで、新しいエクステントを追加できるようになりました。

### MAXISN パラメータ

MAXISN 値はブロック数に変換され、ブロック境界の値に切り上げられます。この範囲のブロックは ADALOD 実行の開始時に割り当てられ、ファイルの先頭のアドレスコンバータエクステントに割り当てられます。

ACRABN パラメータを利用すれば、特定の位置から割り当てを開始できます。

入力データの処理中に最初に割り当てられたエクステントが一杯になった場合には、ADALOD は現在ある全 AC エクステントサイズの合計の 25 % の大きさの AC エクステントをもうひとつ割り当てようとします。

- フリースペーステーブル内に現在使用しているサイズの 25～28 % の範囲の利用可能な未使用ブロックがあれば、このブロックがファイルの新規 AC エクステントとして直ちに割り当てされます。
- 範囲が大きなものしかない場合には、一番範囲の小さいブロックから 25 % の新規 AC エクステントが取られます。
- 小さなエリアしかない場合には、最大のものが割り当てられます。

### NISIZE/UISIZE パラメータ

ADALOD 実行の開始時に、NISIZE および UISIZE パラメータに指定されたブロック数またはシリンダ数が割り当てられ、先頭の NI/UI エクステントに割り当てられます。

NIRABN/UIRABN パラメータを使用して、特定の RABN から割り当てを始めることができます。

これらのパラメータが省略されると、ADALOD は最初に NI および UI スペースを割り当てません。入力された全ディスクリプタ値が中間データセットに書かれた後、次のように NISIZE および UISIZE の見積りが行われます。

- 入力レコードが処理されなかった場合

```
NISIZE = Number of descriptors +1
```

```
UISIZE = 2 blocks
```

- 入力レコードが処理された場合

ファイルの各ディスクリプタに対して、16 個までの中間データセットブロックが選択され、読まれます。これらのブロックの内容はソートされ、このディスクリプタに使われた一時ブロックの総量が見積られます。

このアルゴリズムにより各ディスクリプタの NISIZE および UISIZE の値が求められます。この値の合計が K 倍されます。K は次の計算で求めます。

$$K = (\text{MAXISN} - \text{MINISN} + 1) / \text{number of records loaded}$$

処理中、ADALOD は求めた値が十分ではないと判断すると、実行中に後続のエクステントを割り当てます。このエクステントのサイズは、追加の DSSIZE エクステントと同様に、上記のように計算されます。

## UPDATE 機能

ADALOD ユーティリティの UPDATE 機能は、既存のファイルのレコードの一括追加／削除、再編成、さらに必要に応じてアソシエータまたはデータストレージのスペースを拡張します。

MAXISN パラメータで、より大きい最大 ISN 値を新しく指定した場合、UPDATE で実行した AC、NI、UI の再構成により、未使用のスペースが増えた場合でも、UPDATE 機能は追加の AC スペースを割り当てます。現在のデータストレージスペースに新規レコードを保持できなければ、ADALOD UPDATE はデータストレージスペースを追加します。

## MAXISN パラメータ

UPDATE 処理に現在のファイルの値よりも大きな MAXISN 値が指定されると、旧 MAXISN と新 MAXISN の設定の差が計算されます。その後この量を満たす数の AC ブロックが、フリースペーステーブルから追加エクステントとして割り当てられます。ACRABN パラメータを利用すれば、特定の位置から割り当てを開始できます。

入力データの処理中に、現在の AC エクステントやデータストレージエクステントが一杯になった場合には、ADALOD は現在ある全 AC エクステントサイズやデータストレージエクステントの合計の 25 % の大きさの AC エクステントやデータストレージエクステントをもうひとつ割り当てようとしています。

- フリースペーステーブル内に現在使用しているサイズの 25～28 % の範囲の利用可能な未使用ブロックがあれば、このブロックがファイルの新規 AC エクステントとして直ちに割り当てされます。
- 範囲が大きなものしかない場合には、一番範囲の小さいブロックから 25 % の新規 AC エクステントが取られます。
- 小さなエリアしかない場合には、最大のものが割り当てられます。

## ADAORD ユーティリティによるスペース割り当て

ADAORDは、使用できないスペースを再使用できるように、Adabas アソシエータの各コンポーネント（AC、NI/UI、DSST）およびデータストレージをリオーダします。ADAORD 機能は Adabas 拡張ファイルの指定成分ファイルにのみ有効であって、論理的な整合性を保つために拡張ファイルを変更することはありません。したがって、必要がない限り、ADAORDは拡張ファイルの各成分ファイルに実行してはなりません。

機能	アクセスするテーブルタイプ
REORFASSO	AC、NI、UI
REORASSO	AC、NI、UI、DSST
REORFDATA、REORDATA	DS
REORFILE、REORDB	AC、NI、UI、DS

アクセスするファイルとアクセスするテーブルタイプ（機能による）それぞれに対して次の処置が取られます。

- 使用済みの全スペースはフリースペーステーブルに戻されます。
- 特定の位置（ACRABN、DSRABN、NIRABN、UIRABN）を持つ全テーブルは先頭エクステントとして割り当てられます。使用サイズは指定された（MAXISN、DSSIZE、NISIZE、UISIZE）か、あるいは元のファイルから取られます。
- 特定の位置を持たない全テーブルが先頭エクステントとして割り当てられます。

エクステントの1つが一杯になると、ADALOD（UPDATE）ユーティリティで示したのと同じ処置が取られます（「UPDATE 機能」参照）。

## ADASAV によるスペース割り当て（RESTORE FILES 機能）

データベースにすでに存在するファイルをリストアする（OVERWRITE パラメータ必須指定）場合には、それらの全ファイルに使用されていたスペースはフリースペーステーブルに戻されます。拡張ファイルの成分ファイルを指定する場合、関連する成分ファイルもすべて指定しなければなりません。

### ■ RESTORE FILE=...

リストアするファイルごとに元のRABNが必要です。ADASAVは、必要なエクステントを、その元の場所と元のサイズで割り当てようとします。1つの割り当てに失敗した場合、ADASAVは ERROR-060 で終了します。

### ■ RESTORE FMOVE=...

リストアする各ファイルについて、少なくとも、もともと使用されていたスペース量が割り当てられます。各ファイルテーブルの先頭エクステントの割り当ては、オプションパラメータ ACRABN、DSRABN、NIRABN、UIRABN の1つを使用することで特定の位置にできます。これらのテーブルのサイズは MAXISN、DSSIZE、NISIZE、UISIZE を通して変更できます。

スペースがあれば、複数入力エクステントを新しい1つのエクステントに圧縮できます。連続したフリースペースが十分になれば ADASAV はテーブルをいくつかの新しいエクステント（各テーブルに 99 個まで）に分割します。そのようなスペースがない場合には ADASAV は ERROR-060 で終了します。

## データベースステータスレポートを使用したスペース使用状況の監視

データベーススペース使用に関する情報を得るには、ADAREP ユーティリティを実行するか Adabas Online System を使用してデータベースステータスレポートを出力する必要があります。

このレポートは、ファイル割り当てマップとブロック割り当てマップに加えて次のブロック数を示します。

- アソシエータ物理エクステントについて使用済み（および未使用）ブロック数
- データストレージ物理エクステントについて使用済み（および未使用）ブロック数
- ワーク物理エクステントに割り当てたブロック数
- 各ファイルについて、アドレスコンバータ、ノーマルインデックス、アップパーインデックスおよびデータストレージの各論理エクステントの使用済み（および未使用）ブロック

このレポートで得られる情報の詳細については、『Adabas ユーティリティマニュアル』の「ADAREP」の章を参照してください。

DBA はこのレポートを頻繁にチェックし、スペース使用状況に関する問題点を確認する必要があります。

次のセクションでは、ステータスレポートを使用して発見できる問題点と各問題点の推奨される回避／解決方法を示します。

## スペース使用に関する問題点とその対応策

ここでは、データベーススペースの使用に関してよく発生する問題点と、その予防策や対応策について述べます。

### 物理エクステントが一杯になった場合

1. アソシエータの物理エクステントがほぼ一杯または完全に一杯。
  - 物理エクステントを拡張します（ADADBS ユーティリティの INCREASE 機能を参照）。
  - 新物理エクステントを追加します（ADADBS ユーティリティの ADD 機能を参照）。
  - ADAORD ユーティリティを使用してアソシエータをリオーダします。アソシエータスペースに大量のフラグメントが発生したときのみ、この方法が有効です。
  - ADADBS の DEALLOCATE 機能で未使用のファイルエクステントを解放します。
  - 不要な Adabas ファイルを削除します（ADADBS ユーティリティの DELETE 機能を参照）。
  - 不要なファイルカップリングリストを削除します（ADADBS ユーティリティの UNCOUPLE 機能を参照）。
2. データストレージ物理エクステントがほぼ一杯または完全に一杯。
  - 物理エクステントを拡張します（ADADBS ユーティリティの INCREASE 機能を参照）。
  - 新物理エクステントを追加します（ADADBS ユーティリティの ADD 機能を参照）。これは新エクステントが新デバイスタイプ上に存在するときのみ望めます。
  - データストレージをリオーダします（ADAORD ユーティリティの REORDATA 機能を参照）。データストレージスペース内のフラグメントが大量にあるとき、またはデータストレージパディングファクタを減少するときのみ有効です。
  - あるファイルをリオーダします（ADAORD ユーティリティの REORFILE 機能を参照）。
  - 不要な Adabas ファイルを削除します（ADADBS ユーティリティの DELETE 機能を参照）。

### 最大物理エクステントに達した場合

1. アソシエータの物理エクステントの最大数（約 99）に達しました。
  - 最後のエクステントを ADADBS の INCREASE 機能で増加できます。
  - ADAORD REORASSO 機能を実行してアソシエータをリオーダします。
  - ADAORD RESTRUCTURE 機能を使ってすべてのファイルをアンロードした後、ADAORD STORE を使ってより大きいデータベースにリロードします。
2. データストレージの物理エクステントの最大数（約 99）に達しました。
  - 最後のエクステントを ADADBS の INCREASE 機能で増加できます。
  - データストレージをリオーダします（ADAORD ユーティリティの REORDATA 機能を参照）。データストレージスペース内のフラグメントが除かれます。
  - ADAULD ユーティリティを使用して全ファイルをアンロードした後、より大きなデータベースにリロードします。



## 最大論理エクステントに達した場合

1. ファイルのアドレスコンバータ、ノーマルインデックスまたはアップパーインデックスの最大論理エクステントに達しました。
  - ADAORD ユーティリティの REORFILE または REORFASSO 機能を実行し、ファイルの全アソシエータエントリをリオーダします。
  - ADADBS ユーティリティを使用して ISN の再利用を実行できます。
  - ファイルをアンロードし、削除し、リロードできます。
2. ファイルのデータストレージまたはデータストレージスペーステーブルの最大論理エクステント数に達しました。
  - ADAORD ユーティリティの REORFDATA 機能または REORFILE 機能を使用してこのファイル（およびその他すべてのファイル）をリオーダできます。リオーダすると、データストレージの複数のエクステントがより少ないエクステントに集約されます。
  - ファイルをアンロードし、削除し、リロードできます。



## 25 データベースのモニタリングとチューニング

---

■ リソース使用のモニタリング .....	164
■ データベース使用のレポート出力 .....	164
■ データベース制御のモニタリング .....	164
■ パフォーマンス管理、統計、チューニング .....	165
■ Adabas セッション統計 .....	166
■ コマンドロギング .....	171

モニタリングとチューニングでのデータベース管理者の作業は次のとおりです。

### リソース使用のモニタリング

---

データベースの完全性が維持され、かつ効率的なレベルのサービスが提供されているかを確認するため、継続してデータベース環境のモニタを行うのは DBA の役目です。

DBA は、サービス低下が起こる前にこれを予測したり、データベースのオペレーションや設計を定期的にコントロールされた方法で調整する一連の手順を設定しなければなりません。この手順とは、次のような作業を行います。

- サービス低下を引き起す可能性のある原因の特定
- データベースパフォーマンスをモニタするツールの設定
- 調整作業のコントロール

### データベース使用のレポート出力

---

DBA は、データ処理部門とユーザー部門のマネージメントに対して、データベース使用とパフォーマンスに関して定期的にレポートすべきです。このレポートは、できるだけ事実を反映させ、また必要があればデータベース環境に対するチューニングの勧告も含む必要があります。チューニングは組織全体には利益をもたらしますが、一部のユーザーにはサービスの低下になることもあります。したがって、チューニングに関する決定はすべての関連ユーザーのことを考慮しなければなりません。

### データベース制御のモニタリング

---

DBA は適切な管理方法を設定し、データベースの完全性を保証するためにこれをモニタする必要があります。

このコンピュータが作成する管理情報（コントロールトータル）がコンピュータ処理実行時間や生成レポートについてチェックするときに役立ちます。バッチレスポンス（あるいは問い合わせ）では、実行時間、サーチパラメータ、最終データ更新時刻およびプライマリパラメータコントロールのような情報を含みます。これは、信頼レベルを高め、データベースの完全性を保証するうえで役に立ちます。

コントロールトータルに問題がある場合、それぞれのインストールに応じて問題が異なることがあります。この分野で、しっかりした規則をすぐに作成することは不可能ですが、一般的な指標は示すことは可能です。

データベースを使用する各アプリケーションシステム的设计時に、DBA は次のような考慮がされているかを確認する必要があります。

- 各バッチ更新処理の実行時、どのようなチェックを行えるか。例えば、レコード件数、追加、削除、更新を行うときなど。
- これらをチェックするのにどんな制御がファイル全体にわたって必要か。例えば、値フィールドのハッシュトータルなど。
- ある期間の終わりに、コントロールトータルを調べ誤りであることがわかったとき、データを回復するためにどんな入力ランザクション、Adabas ログ等を保持しておかなければならないか。
- 地域別のコントロールトータル（支店別、製品グループ別）から、ファイルコントロールトータルエラーの影響を受ける領域を特定できるか。

## パフォーマンス管理、統計、チューニング

次の表に、使用する統計のモニタリングのいくつかとデータベース環境に対する調整（あるいはチューニング）を示します。

変化するもの	必要なチューニング				
	データベースの構造	使用アクセスメソッド	ハードウェアまたはソフトウェア構成	処理プライオリティ	ディスクストレージ割り当て
端末およびライントラフィック		○	○	○	○
レスポンスタイム（アプリケーションのパフォーマンス）	○	○	○	○	○
ユーザーの総アクセスおよびディスクリプタ	○	○			○
データベースサイズ	○	○	○		○
データベースの成長度	○	○	○		○

本番データベースに何らかの変化があったときは、信頼性や完全性を高度に保つよう十分な注意が必要です。何が変化しても、DBA は正しい判断を下しそれが正確に行われるよう保証しなければなりません。チューニング処理に対してDBA は絶対的なコントロールを持ち、正式な受入れ手順に沿っているかを確認しなければなりません。

上の表中の項目の変化について過度に対応することがないよう DBA は注意しなければなりません。ライントラフィックやレスポンスタイム等が突然変化した場合、一時的な現象かもしれないので、通常のオペレーティング状態でしばらく様子を見て、恒久的な傾向なのか一時的な現象なのかを判断する必要があります。

もう1つの表の見方として、新規プロジェクトの導入により、ライントラフィックやレスポンスタイム等にかかなりの変化がみられるとき、要求のチューニングが必要かを判断するのに使用します。DBAは、新規アプリケーションシステムを導入する前に、これらの影響を最小限にとどめるように対応できます。

## Adabas セッション統計

---

各 Adabas セッションの終わりに出力される統計を使用して Adabas のパフォーマンスをモニタできます。具体的には、セッション統計として次の統計が用意されています。

- 入出力 (I/O) 統計
- コマンド統計
- プール/キュー使用統計

### I/O 統計

次の I/O 統計が出力されます。

#### I/O 件数 (初期化を含む)

	読み込み	書き込み
ASSO	50	21
DATA	2388	2184
WORK	9	1385
PLOG	9	1603
CLOG	0	0
合計	2456	5193
LOG. READS	33899	
BUFFER EFF.	13.9	

I/O 件数はセッション中に実行されたアソシエータ (ASSO)、データストレージ (DATA)、ワーク (WORK)、データプロテクションログ (PLOG)、およびコマンドログ (CLOG) に対する物理 I/O 回数を示します。

さらにバッファプールに対して発行された論理読み込みの回数 (LOG. READS)、および論理読み込み回数をアソシエータとデータストレージの読み込み回数で割った、バッファ効率 (BUFFER EFF.) が得られます。バッファ効率の値が高いほどバッファプールの利用効率が良くなります。その値が 10 未満の場合には DBA は Adabas バッファプールのサイズを大きくする必要があります (『Adabas オペレーションマニュアル』の「LBP パラメータ」参照)。

## VOL-SER 番号ごとの ASSO/DATA I/O の配分 (初期化を除く)

VOL-SER	HIGH RABN	COUNT
ADA003	(ASSO : 894)	38
ADA003	(ASSO : 2544)	6
ADA003	(DATA : 894)	0
ADA003	(DATA : 1344)	4572
合計		4616

物理ボリュームごとのアソシエータ、およびデータストレージについてのI/O配分に関する情報も提供されます。アクセス/更新された最大 RABN (HIGH RABN)、および I/O 回数 (COUNT) が示されます。DBA はこのデータを使用して、バッファプールパラメータやデータベースの物理割り当てに調整が必要かどうか判断する必要があります。

## コマンド統計

次の例では、Adabas は5つのスレッドで12,687 コールを実行したセッションのコマンド統計が出力されます。

## ソース別のコマンド数の配分

次の表は、セッションのコマンドソースを表示します。同一の環境 (ローカル) またはネットワークを経由するリモート環境のどちらかです。

ソース	件数
REMOTE LOGICAL	0
REMOTE PHYSICAL	0
LOCAL LOGICAL	0
LOCAL PHYSICAL	12,686

## スレッド別のコマンド数の配分

次の表はセッションにおけるスレッドの活動を示します。

スレッド	件数
1	7,328
2	2,728
3	1,240
4	814
5	541

スレッド	件数
合計	12,651

最大番号のスレッドがゼロより大きい活動回数を持つ場合には、スレッド数を増やせば Adabas ニュークリアスはより多くのコマンドを処理できることが推測できます。スレッド数の増加によりコマンドがコマンドキューで選択を待つのを防止できます。

### ファイル別のコマンド数の配分

次の表はファイル別のコマンド数の配分を示します。

ファイル	件数
0	4,247
1	8,404
合計	12,651

ファイルに関係しないコマンド（BT、ET など）はファイル 0 に数えられます。

### コマンドタイプ別のコマンド数の配分

次の表はコマンドタイプ別のコマンド数の配分を示します。

コマンドタイプ	件数
A1/4	4,198
ET	4,191
L1/4	4,242
OP	56
合計	12,687

コマンドタイプ UC は Adabas ユーティリティから発行された特権コールを示します。



**Note:** コマンドタイプ REST は C1、C5、RI、HI のようなコマンドを示します。



## 追加セッション統計

```
THERE WERE 56 USERS PARTICIPATING MOST CALLS ( 57) INITIATED BY USER user ID MOST
I/O-S ( 14) INITIATED BY USER user ID MOST THR.-TIME (04:16:32) WAS USED BY USER
user ID
```

28 フォーマットが変換された  
 0 フォーマットが上書きされた  
 0 自動再スタートが行われた  
 20 ISN 問題が原因で戻された  
 16 スペース問題が原因で戻された  
 186 バッファフラッシュが起きた

### フォーマットの変換／上書き

Adabasの読み込みコマンドと更新コマンドでは読み込む、あるいは更新するフィールドをフォーマットバッファに指定する必要があります。このフォーマットバッファはAdabasによって解釈され、内部フォーマットバッファに変換されます。結果としてできた各内部フォーマットバッファは内部フォーマットバッファプールに格納され、ユーザーIDとコマンドIDの組み合わせにより識別されます。

Adabasは新しい読み込み／更新コマンドについてそれぞれユーザーID／コマンドIDで識別されるエントリがすでにプール内に存在しているかをチェックし、存在していない場合、Adabasによってコマンドの新しいフォーマットバッファが変換され、プールに入れられます。フォーマットバッファプールが一杯になると、既存のエントリは新しいエントリを格納するために上書きされます。

フォーマット変換処理では、CPUに大きな負荷がかかります。したがって、DBAはフォーマットの上書きが過剰にならないよう注意し、次の処置を取る必要があります。

1. ユーザープログラムがコマンドIDを正しく使用しているかどうか、つまり適時空白以外のコマンドIDを使用したり、不要なコマンドIDを解放しているかどうかを確認します。コマンドID使用の詳細については、『Adabas コマンドリファレンスマニュアル』を参照してください。
2. 内部フォーマットバッファプールのサイズの拡大を考慮します（『Adabas オペレーションマニュアル』「ADARUN LFP パラメータ」参照）。

Adabas ニュークリアスによって、各セッションの終了時にフォーマット変換とフォーマット上書きの統計表が作成されます。Adabas オペレータコマンド DSTAT を使用してこの情報を確認することもできます。

### 自動再スタート

セッション中に行われた自動再スタートの回数です。

### コマンドのスローバック

Adabas ニュークリアスが、次のリソースのどちらかを待っていたためにコマンドが実行できなかった回数です。

- 使用可能な ISN
- Adabas ワークプールスペース

このような場合にはコマンドは後で処理するようコマンドキューに戻されます（スローバック）。

この数が 0 より大きければ、次の処置を取る必要があります。

1. ADARUN LWP（ワークプールサイズ）パラメータと LS（ソートワークエリア）パラメータの比率を調整します。
2. Adabas ワークプールのサイズを大きくします（ADARUN LWP パラメータ）。
3. ADARUN TT（トランザクションタイムリミット）パラメータを評価します。
4. アプリケーションプログラムのホールドロジックをチェックします。
5. Adabas ホールドキューサイズを大きくします（ADARUN NH パラメータ）。
6. スーパーディスクリプタを使って検索コマンドの複雑さを減らします。

ADARUN パラメータについては、『Adabas オペレーションマニュアル』を参照してください。

### バッファフラッシュ

セッション中に起きたバッファフラッシュの回数。


Adabas バッファプールはアクティブな全ユーザーに共有される仮想データベースを示します。バッファプールには、最もよく使われるアソシエータおよびデータストレージブロックが含まれ、その目的は物理 I/O を最小にすることです。

バッファプールの大きさは ADARUN LBP パラメータで決まります。LBP は、できる限り大きく設定する必要があります。その設定が大きすぎると、オペレーティングシステムのページング回数が極端に増えてしまうため、そうならないように注意します。

## バッファおよびキューの統計

セッション統計には、セッション中に使用したバッファおよびキューの最大利用率も含まれます。このような統計は、バッファプールを除き、最大値を計算できるすべてのバッファおよびキューについて示されます。次の表にサンプルセッションの最大値を示します。

プールエリア	ADARUN パラメータ	最大値	%
AB	NAB = 10	12032	29
CQ	NC = 20	3648	95
DUQ	LDEUQP = 5000	500	10
FI	LFP = 12000	1760	14
HQ	NH = 100	552	23
SC	LCP = 10000	0	0
TBI	LI = 10000	0	0
TBS	LQ = 10000	0	0
UQ	NU = 20	4880	86
UQF	NU = 20		
WORK	LWP = 14000	70464	50
XID	XID = 0	0	0

 **Note:** UQF は、ファイルリストを保持するユーザーキュー拡張です。そのプールのサイズは、UQ プールサイズを使用して計算されます。

セッション中に有効であった ADARUN パラメータ設定値に関する最大値が示されます。

DBA は、それぞれの最大値を監視し、必要ならば適切な ADARUN パラメータを調整しなければなりません。

## コマンドロギング

Adabas コマンドロギングは Adabas に対してユーザーが発行した全コマンドに関する情報を生成します。次の情報が得られます。

- ユーザー ID
- 日時
- 使用されたコマンド
- アクセスされたファイル
- アクセスされたレコード
- 受け取った Adabas レスポンスコード

### ■ コマンドの実行にかかった時間

コマンドロギングは ADARUN パラメータ LOGGING により制御されます。

## 26 エラー処理およびメッセージバッファリング

---

■ 概要 .....	174
■ オペレーションの範囲 .....	174
■ リカバリまたはプラグイン (PIN) ルーチン .....	175
■ PIN ルーチンユーザー出口 .....	187

このchapterでは、次のトピックについて説明します。

### 概要

---

エラー処理およびメッセージバッファリング機能は、DBA の介入をほとんど必要とすることなく、特定タイプのエラーを自動的に分析および回復するため、24X7 オペレーションの実装に有効です。また、追加情報も生成されるため、ユーザー独自によるエラー診断と、Software AG によるエラー診断が可能です。

機能を有効にするには、ADARUNSMGTパラメータを設定します。メッセージバッファリングを使用する場合、ADARUN MSGBUF パラメータを使用してバッファのサイズを決定します。ラップアラウンド方式のメッセージバッファは、コンソールや DDPRINT メッセージへのオンラインアクセスが不可能になった場合に、後で Adabas Online System で確認するために Adabas メッセージを収集します。バッファは、問題分析とパフォーマンスチューニングに有効です。

エラー処理機能は、オペレータコマンド SMGT のオペランドとして実装され、オペレータコンソールまたは Adabas Online System から起動することができます。詳細は、『Adabas オペレーションマニュアル』を参照してください。

現在実装されている機能は、ニュークリアスが自分自身を保護し、次のエラーが発生した場合に、追加のエラーリカバリ情報を提供することができます。

- パラメータエラー 31：自動再スタートエラー。
- パラメータエラー 73：初期化中にチェックポイントファイルが一杯になった。
- 診断を助けるためのストレージ関連エリアの取得による非レスポンスコード。
- 追加のダンプエリアの提供によるプログラム中断。

### オペレーションの範囲


---

#### ユーザー出口エラー

SMGT オペレータコマンドのオペランドの1つを使用して、Adabas ニュークリアスの操作に必要なユーザー出口およびハイパー出口を、クリティカル（デフォルト）、または非クリティカルとして扱うことができます。

- ユーザー出口がクリティカルとして定義された場合、エラー処理およびメッセージバッファリング機能による影響はありません。出口内で異常終了すると、Adabas ニュークリアスも異常終了します。
- ユーザー出口が非クリティカルとして定義され、その中で異常終了が発生した場合、機能はアクティブな Adabas ニュークリアスを維持し、オプションによってはその出口の起動を止め

て、出口エラー発生時のニュークリアスのダンプを取得し、システムログにメッセージを発行して DBA に問題を通知します。その後、DBA は診断情報を調査し、問題を修正し、SMGT オペレータコマンドのオペランドを使用して、修正された出口をロードおよび再度アクティブにすることができます。

 **Note:** Adabas 出口がサブタスクをアタッチする場合、サブタスクはエラー処理およびメッセージバッファリング機能によって保護されることはありません。

## リカバリまたはプラグイン (PIN) ルーチン

エクステンション (プラグインルーチンまたは PIN) は、ニュークリアスが処理を継続できるようにしながら、アベンドを分析し、原因を判断するようにデザインされています。PIN は、ニュークリアスが処理を継続しても安全かどうかを判断し、適切なメッセージを表示して DBA に知らせます。

PIN ルーチンのユーザー出口 ADASMXIT を使用して、レスポンスコードおよびアベンドについての追加情報を得ることができます。ユーザー出口では、特定のレスポンスコード、またはレスポンスコード/サブコードの組み合わせを監視するように指定することができます。ユーザー出口を修正すると、それをリロードし、データベースをダウンさせることなく変更を有効にすることができます。

各プラグイン (PIN) サービスルーチンは、事前に定義された条件に合致すると、これを処理し、Adabas ニュークリアスに次のいずれかを許可します。

- 異なる方法で異常終了するときは、アクティブなままにする。
- エラーリカバリの補助として、拡張エラー診断を出力する。

実行状況に応じて、PIN モジュールは、通常の処理を再開するために制御を Adabas ニュークリアスに移す (通常レスポンスコードを伴う) か、エラー処理およびメッセージバッファリング機能に制御を返して、Adabas ニュークリアスを異常終了させることができます。

PIN の実行中、異常なイベントの発生時に、PIN は大部分の Adabas 機能をレジスタとして使用できます。PIN は、ニュークリアスがアクティブであり続ける必要があるかどうかを判断します。

PIN を使用すると、特定のレスポンスコードまたは ABEND コードをデバッグしやすくするために、さまざまな状況に合わせてダンプをわかりやすい形式で出力させることもできます。

PIN が、ニュークリアスがアクティブであり続けることを決定した場合、PIN はレスポンスコードを設定します。

## PIN 処理

異常終了またはゼロ以外のレスポンスコードが発生した場合、エラー処理およびメッセージバッファリング機能は、検出された特定条件と位置について PIN ルーチンを最初に検索し、次に特定条件、最後に位置（すべての条件）を検索します。該当する PIN が見つかった場合、それが起動されます。それ以外の場合、ニュークリアスは終了されます。

PIN ルーチンが起動された場合、次の状態になります。

- 条件を処理すると、エラー処理機能からの介入なしで処理を継続します。
- 条件を処理できない場合、PIN はエラー処理機能に制御を返し、ニュークリアスを終了します。


レスポンスコードエラーについては、エラー処理機能は、監視しているレスポンスコードかどうかを最初に判断します。

- 監視しているレスポンスコードではない場合、PIN はニュークリアスに制御を返し、レスポンスコードはユーザーに通常どおりに返されます。
- 監視しているレスポンスコードの場合、適切な PIN が起動され、問題の解決に有効なレスポンスコードについて追加情報を出力します。その後、PIN はニュークリアスに制御を返し、レスポンスコードはユーザーに通常どおりに返されます。

PIN がレスポンスコードを処理し終わると、エラー処理機能に制御が返され、通常のレスポンスコード処理を継続することができます。

## デフォルト PIN モジュール ADAMXY

ADAMXY モジュールは、Adabas とともに配布され、初期化中に自動的にインストールされる、PIN ルーチンで構成されるデフォルトの PIN モジュールです。

 **Note:** SMGT,DELPIN または SMGT,DEACTPIN オペレータコマンドを使用して、デフォルトの PIN ADAMXY を使用不可にすることができます。

次の表は、ADAMXY の PIN ルーチンで処理される割り込みについて説明します。それぞれの割り込みについて、エラー分析のために拡張ダンプフォーマットが用意されています。

コード	例外タイプ	プロセッサ
01	オペレーション	無効なオペレーションコードを持つ命令を実行するところです。
02	特権を持つオペレーション	問題がある状態で、管理的な命令を実行しようとしています。
03	実行	問題の診断を支援するために、プログラムを故意に中断します。
04	保護（セグメントおよびページも）	システムまたはハードウェアストレージの変更、フェッチ保護システムまたはハードウェアストレージのアクセス、または割り当てられていないストレージのアクセス／修正を試みます。ジョブログのユーザー通



コード	例外タイプ	プロセッサ
		知と一緒に、レコード／ファイルレベルのロックを必要とします。コード 16（セグメント例外）およびコード 17（ページ例外）も、コード 04 としてエラー処理機能に提示されることに注意してください。
05	アドレス	無効な読み込みアドレスへの参照を検出しました。
06	指定	古いアドレス、または整列された引数を持たないフィールドに整列を要求する命令を、設定または分岐しようと試みます。
07	データ	破壊されたデータレコードを検出しました。パック 10 進であるべきフィールドに誤りがある可能性があります。
08	固定小数点	上位の桁繰り上げが発生しています。または、固定小数点の加算、減算、シフト、または符号制御演算で上位の重要なビットが失われています。
09, 11, 15	除算	除算命令に、ゼロの除数がありました。レコードが壊れている可能性があります。コード 9 はバイナリ、コード 11 はパック 10 進、コード 15 は浮動小数の計算です。

## メッセージ

```
*****DEFAULT PIN OUTPUT*****
```

デフォルトの PIN ADAMXY が起動されると生成されます。この後に、ADAMXY のプログラム割り込み処理に関連するすべての出力が続きます。メッセージ

```
*****END OF DEFAULT PIN OUTPUT*****
```

ADAMXY が完了すると生成されます。

## 付属の追加 PIN モジュール

このセクションで説明するいくつかの PIN モジュールは、Adabas のアドオン製品とともに提供されます。それらはニュークリアス起動時に関連するサーバーコンポーネントが初期化されるとき、自動的に設定されます。

PINAFP  
PINATM  
PINAVI  
PINCOR  
PINSAF

このセクションで説明する残りの PIN モジュールは Adabas に含まれていますが、ADAMXY の一部ではないので、Adabas 初期化時に自動的にインストールされません。

PINAUTOR  
PINOPRSP  
PINRSP

### PINUES

PINRSP および PINUES は、ニュークリアスがアクティブなときに、SMGT,ADDPIN=モジュール名コマンドを使用してインストールされます。PINAUTOR および PINOPRSP は、オペレータコマンドを使用できないシステムの初期化中に起動されるので、Adabas ロードライブラリの特定モジュールをリネームしてアクティブ化されます。

- NOOPRSP を PINOPRSP にリネームすると、その PIN をアクティブ化します。
- NOAUTOR を PINAUTOR にリネームすると、その PIN をアクティブ化します。

PIN モジュールのインストールについての詳細は、『Adabas インストールマニュアル』を参照してください。

### PINAFP

PINAFP はアドオン製品 Adabas Fastpath とともに提供されます。これは、ニュークリアスの起動 (ADARUN FASTPATH=YES) 時に Adabas Fastpath サーバーコンポーネントが初期化する際に、自動的に設定されます。

Adabas Fastpath サーバーコンポーネントでプログラムの割り込み (「デフォルト PIN モジュール ADAMXY」のセクションの表を参照) が発生すると、PINAFP に制御が渡され、コンポーネントで使用されているメインメモリエリアがフォーマットおよび出力されます。これらの診断は、次のタイトル付きで DDPRINT データセットに書き込まれます。

```
ADABAS FASTPATH - memory-area-name : SNAP BY PINAFP
```

その後、Adabas が異常終了できるように、PINAFP はエラー処理およびメッセージバッファリング機能に制御を返します。

必要であれば、PINAFP を稼動および停止することができます。ただし、PINAFP を再稼動すると、次のニュークリアスセッションまで再設定することができません。

### PINATM

PINATM はアドオン製品 Adabas Transaction Manager (ATM) とともに提供されます。ATM ジョブの初期化時 (ADARUN DTP=TM) に自動設定されます。

ATM ロジックでプログラムの割り込みが発生すると (「デフォルト PIN モジュール ADAMXY」のセクションの表を参照)、制御が PINATM に渡され、ATM で使用されているメインメモリエリアがフォーマットおよび出力されます。これらの診断は、次のタイトル付きで DDPRINT データセットに書き込まれます。

ADABAS TRANSACTION MANAGER - memory-area-name : SNAP BY PINATM

その後、Adabasが異常終了できるように、PINATMはエラー処理およびメッセージバッファリング機能に制御を返します。

必要であれば、PINATMを稼動および停止することができます。しかし、PINATMを再稼動すると、次のATMセッションまで再設定することができません。

## PINAUTOR

Adabas ロードライブラリのNOAUTORがPINAUTORにリネームされ、自動再スタート中にパラメータエラー31が発生すると、PINAUTORが制御を取得します。PINAUTORは、エラーがあるファイルを識別し、可能であれば自動再スタートからそれを取り除こうとします。

自動再スタート処理からファイルを排除する前に、PINAUTORは次をチェックします。

- ファイルが、セキュリティまたはチェックポイントファイルではないか。
- レスポンスコードが、9、65、72、88、97、99、148、または151ではなく、排除処理に有効ではないか。

さらに、PINAUTORは、特定データベースの一定ファイルまたは特定レスポンスコードが、排除に不適格と指定されていないかをチェックします。例えば、データベースが必要とする特定ファイルがないと、データベースを開始する意味がありません。ADASMXITをカスタマイズして、排除できないファイルおよびレスポンスコードを組み込むことができます。また、排除付き自動再スタートを試行できる最大数を指定することができます。

AREXCLUDEプロシージャは、ファイルを排除するために自動起動されます。詳細は、『Adabas オペレーションマニュアル』で「ADARUN パラメータ」の章の「AREXCLUDE」を参照してください。排除されたファイルは、整合性がなくなり、ADASAV RESTORE を使用してバックアップからリストアする必要があるかもしれないことに注意してください。

ファイルが自動再スタートから排除された場合、SM-PINAUTOR2 メッセージが生成され、ADAN50 メッセージが続きます。両方とも、排除されたファイルのファイル番号を示します。

PINAUTOR が起動されると、メッセージ [PINAUTOR OUTPUT] が生成され、特定の PINAUTOR 状態（『Adabas メッセージおよびコードマニュアル』を参照）に関連するメッセージが続きます。PINAUTOR 処理の完了を示すために、メッセージ [ENDPINAUTOROUTPUT] が生成されます。

### PINAVI

PINAVI はアドオン製品 Adabas Vista とともに提供されます。これは、ニュークリアスの起動（ADARUN VISTA=YES）時に Adabas Vista サーバーコンポーネントが初期化する際に、自動的に設定されます。

Adabas Vista サーバーコンポーネントでプログラムの割り込み（「**デフォルト PIN モジュール ADAMXY**」のセクションの表を参照）が発生すると、PINAVI に制御が渡され、コンポーネントで使用されているメインメモリエリアがフォーマットおよび出力されます。これらの診断は、次のタイトル付きで DDPRINT データセットに書き込まれます。

```
ADABAS VISTA - memory-area-name : SNAP BY PINAVI
```

その後、割り込みが発生したプログラムが PINAVI によって使用不可にされ、Adabas が継続できるように、制御が Adabas に返されます。プログラムを使用不可にしても、Adabas サービスは中断されませんが、Adabas Vista ファイルへのアクセスは制限される可能性があります。この場合、ユーザーにはゼロ以外のレスポンスコードが返されるので、制限がかかっていることが分かります。

必要であれば、PINAVI を稼動および停止することができます。ただし、PINAVI を再稼動すると、次のニュークリアスセッションまで再設定することができません。

### PINCOR

PINCOR は Adabas オプションの System Coordinator で提供されます。ニュークリアス起動時に System Coordinator サーバーコンポーネント（ADAPOP）が初期化するとき、自動的に設定されます。


System Coordinator サーバーコンポーネントでプログラム割り込みが発生すると、コンポーネントに使用されるメインメモリエリアをフォーマットし、出力する PINCOR に制御が渡されます。

これらの診断は、次のタイトル付きで DDPRINT データセットに書き込まれます。

```
COMMON RUNTIME - memory-area-name : SNAP BY SMGT
```

その後、Adabas が異常終了できるように、PINCOR はエラー処理およびメッセージバッファリング機能に制御を返します。

## PINOPRSP

 **Caution:** PINOPRSP は、データベースを初期化し、チェックポイントを書き込まないで操作できるようにします。データベースリカバリ手順が必要になった場合、チェックポイント情報がないと致命的なエラーが発生する場合があります。これを防止するには、PINOPRSP の起動直後にチェックポイントファイルをリオーダーし、初期化とチェックポイントファイルのリオーダーとの間に発生した、データベースのステータス変更をすべて再現できる必要があります。

NOOPRSP が Adabas ロードライブラリで PINOPRSP にリネームされ、システムの初期化中にチェックポイントのオーバーフロー状態を示すパラメータエラー 73 が発生すると、PINOPRSP が起動されます。


メッセージ [PINOPRSP OUTPUT] が生成され、PINOPRSP が起動されたことを示します。その後、PINOPRSP は DBA に警告メッセージを生成し、チェックポイントファイルが一杯になったとしても Adabas が稼動することを知らせます。

```
response code INTERCEPTED BY PINOPRSP BECAUSE THE CHECKPOINT FILE IS
FULL. THE ADABAS NUCLEUS WILL ACTIVATE BUT THE CHECKPOINT FILE NEEDS
TO BE REORDERED AS SOON AS POSSIBLE.
```

この場合、レスポンスコードは 75 または 77 です。チェックポイントは書き込まれませんが、ニュークリアスは稼動します。すぐに正しい対処を行う必要があります。その後、メッセージ [END PINOPRSP OUTPUT] が生成され、PINOPRSP 処理の完了を示します。

## PINRSP

SMGT,ADDPIN=PINRSP オペレータコマンドは、PINRSP をアクティブにし、レスポンスコードの診断を補助する拡張情報を提供します。

 **Note:** Adabas によって設定されたレスポンスコードだけが記録されます。Adabas に到達する前に Adabas SVC によって設定される 22 (無効なコマンドコード) などのレスポンスコードは記録されません。

Adabas PIN ルーチンのユーザー出口 ADASMXIT を修正しないで、PINRSP がインストールされた場合、すべてのレスポンスコードが記録されます。次のために、ADASMXIT をカスタマイズすることができます。

- 特定のレスポンスコードまたはレスポンスコード/サブコードの組み合わせを組み込む。
- 特定のレスポンスコードを監視できる回数を示す。

ゼロ以外のレスポンスコードが発生した場合、PINRSP が制御を取得します。メッセージ [PINRSP OUTPUT] が生成され、PINRSP が制御を持っていることを示します。発生するレスポンスコードによって、異なるエリアが記録されます。

記録されるエリアに関して、5つのカテゴリのレスポンスコードを示します。

1. 基本レスポンスコードの記録
  - アクティブスレッド
  - FCB (可能な場合)
  - GETMAIN された現在使用中のエリア
2. インデックス関連のレスポンスコード (177 など)
  - 基本レスポンスコードの記録
  - スレッドからのインデックス構造
  - アクティブ CQE
  - バッファプールヘッダー
3. 追加 IUB エリア関連のレスポンスコード (40 など)
  - 基本レスポンスコードの記録
  - IUB
  - アクティブ CQE
4. 追加のアタッチドバッファ情報が必要であるというレスポンスコード (255 など)
  - 基本レスポンスコードの記録
  - アクティブ CQE
  - アタッチドバッファ情報
5. ユーザーキューが有効なレスポンスコード (72 など)
  - 基本レスポンスコードの記録
  - アクティブ CQE
  - ユーザーキュー

### 例

特定レスポンスコードの診断に追加情報が必要なとき、CLOG を取得するよりも、ADASMXIT を修正してレスポンスコードを取得し、再アセンブルし、ニュークリアスの稼動中にロードすることができます。そうすると、レスポンスコードが次に発生したときに、情報が記録されます。

情報を取得すると、ADASMXIT を修正してレスポンスコードを削除し、情報がこれ以上取得されないようにリロードすることができます。ADASMXIT を最初に設定して、情報を n 回だけ記録することもできます。

また、ADASMXIT とともに PINRSP を使用し、レスポンスコード 201、202、または 203 に生成される ADAN77 メッセージを省略することもできます。これは、新しいアプリケーションが多くのセキュリティエラーを受け取り、SYSLOG を一杯にするような状況に有効です。推奨して

いませんが、一時的に ADASMXIT を修正して N77 メッセージを省略し、ADASMXIT に示されるレスポンスコード 201、202、203 で PINRSP をアクティブにすることができます。

メッセージの省略がアクティブな場合、ADAN77 メッセージ [Message suppression in effect] が生成され、レスポンスコードに関連するフォーマット情報を提供する PINRSP 出力は省略されます。

PINRSP が処理を完了すると、メッセージ [END PINRSP OUTPUT] が生成されます。

## PINSAF

PINSAF はアドオン製品 Adabas SAF Security (ADASAF) とともに提供されます。これは、ニュークリアスのスタートアップで ADASAF が初期化する際に自動的に起動されます。

ADASAF でプログラムの割り込みが発生すると（「[デフォルト PIN モジュール ADAMXY](#)」のセクションの表を参照）、制御が PINSAF に渡され、ADASAF で使用されているメインメモリエリアがフォーマットおよび出力されます。これらの診断は次のタイトルでデータセットに書き込まれます。

```
ADABAS SAF INTERFACE - control-block-name : SNAP BY SMGT
```

その後、Adabas が異常終了できるように、PINSAF はエラー処理機能に戻ります。



**Note:** セキュリティの理由から、PINSAF は、ADASAF のアベンド後に Adabas が継続することを許可しません。

その他の PIN ルーチンのように、PINSAF は稼動および停止することができます。しかし、PINSAF が再稼動された後は、PINSAF が正しく機能する前に、ADASAF 自体を再スタートしなければなりません。詳細については、Adabas SAF Security のドキュメントを参照してください。

## PINUES

PINUES は、ユニバーサルエンコーディングサポート (UES) システムに関する Adabas レスポンスコードを処理します。PINUES は、次を試行した場合に入力/出力エラーを取得します。

- 存在しないエンコードオブジェクトをロードする。
- 無効なデータを変換する。



**Note:**

他の PIN ルーチンが同じエラー状態を処理する場合、最後にロードされた PIN がエラー処理のためにコールされます。例としては、次のようなものがあります。

## エラー処理およびメッセージバッファリング

```
F NUC227,SMGT,ADDPIN=PINRSP
```

```
F NUC227,SMGT,ADDPIN=PINUES
```

この例では、PINUESがPINRSPの後にロードされるので、処理できるエラー状態を処理します（レスポンスコード 17、48、および 55）。その他すべてのレスポンスコードは、PINRSPによって処理されます。

メッセージ [PINUES OUTPUT] が生成され、PINUESが制御を得たことを示します。PINUES処理が完了すると、メッセージ [END PINUES OUTPUT] が生成されます。

### エラー状態の処理

- ユーザーおよび Adabas ファイル間のデータ変換を判断するために必要な ECS オブジェクトがない場合、OP コマンドにレスポンスコード 17 および 48 が発生する可能性があります。この場合、PINUES はオプション IUB および UQE で ADAMXF をコールし、診断の出力を取得します。
- ECS が、テキストの変換または移動に失敗したことを示すレスポンスを返した場合、レスポンスコード 55 が発生する可能性があります。この場合、変換パラメータおよびバッファは、診断の出力を取得するためにスナップされます。

### 出力の作成

PINUES が診断情報を書き込むと、次の行がコンソール上に表示されます。

```
***** P I N U E S   OUTPUT *****'

ADANX1 dbid COMMAND cmd COMMAND ID hex-cid FNR file-number

      RESPONSE adabas-response-code SUBCODE adabas-subcode FLD

field-name'

      TID hex-internal-user-id UID open-userid JOB job-name'

***** END P I N U E S   OUTPUT *****'
```

### セッションのスナップショット



```
18:17:37 ADAN19 00227 BUFFERFLUSH IS A S Y N C H R O N O U S
18:17:37 ADAN01 00227 A D A B A S V7.1.0 IS ACTIVE
18:17:37 ADAN01 00227 MODE = MULTI
18:17:37 ADAN01 00227 RUNNING WITHOUT RECOVERY-LOG
18:18:04 ADAI29 OPER CMD: SMGT,ADDPIN=PINUES
18:18:04 ADANTG 00227 PIN MODULE PINUES LOADED
18:18:04 ADANO2 00227 SMGT COMMAND PROCESSED
18:18:04 ADAN41 00227 1999-01-00 18:18:03 FUNCTION COMPLETED

18:36:33 ADAN7A 00227 ECS ERROR -2 IN FUNCTION GETHANDL

18:37:21
18:37:21 ***** P I N U E S OUTPUT *****
18:37:21 ADANX1 00227 COMMAND OP COMMAND ID 00000000 FNR 00014
18:37:21 RESPONSE 017 SUBCODE 023
18:37:21 TID 00000013 UID BLAUTOPF JOB TXG.....
18:37:21 ADAH51 00227 DUMP FORMAT CALLED
```

次の出力は、ADAMXF モジュールによって作成されます。

```
18:37:22 ADAH52 00227 DUMP FORMAT COMPLETED
18:37:22 ***** END P I N U E S OUTPUT *****
18:37:22
18:49:45 ADAN7A 00227 ECS ERROR 54 IN FUNCTION CVFTXTX
18:49:45
18:49:45 ***** P I N U E S OUTPUT *****
18:49:45 ADANX1 00227 COMMAND A1 COMMAND ID 00000000 FNR 00014
```

18:49:45                   RESPONSE 055 SUBCODE 004  
18:49:45                   TID 00000017 UID ANDECHS. JOB TXG.....

ECS CONVERSION PARAMETERS

0C190BE0+0000  00106570 00000080 00000200 00000000 \*.....\*  
0C190BF0+0010  00000004 0C10ACB4 00000004 0010A6E0 \*.....w.\*  
0C190C00+0020  00004000 0C190BEC 0000020C 00000000 \*.....\*  
0C190C10+0030  001065AD 0C190BE4 00000000 00000000 \*.....U.....\*

ECSE FROM ENCODING 3026 TO ENCODING 3035

00106570+0000  02000002 00000BD2 00000BDB 00000004 \*.....K.....\*  
00106580+0010  00000000 001062C8 00106350 001064E8 \*.....H.....Y\*  
00106590+0020  0000BD20 0000BDB0 0000000C 00000000 \*.....\*  
001065A0+0030  001065AD 0004362C 40000000 00FEFE00 \*.....\*  
001065B0+0040  00000340 40000000 04404000 00000000 \*.....\*  
001065C0+0050  00000000 00000000 00000000 00000000 \*.....\*  
001065D0+0060  00000000 00000000 00000000 00000000 \*.....\*  
001065E0+0070  00000000 00000000 00000000 00000000 \*.....\*  
001065F0+0080  00000000 00000000 02000001 00000BDB \*.....\*

ECONV INPUT AREA

0C10ACB4+0000  4141F1F2 40404040 40404040 40404040 \*..12.....\*

```
ECONV OUTPUT AREA

0010A6E0+0000  414150C2 50C350C4 50C550C6 50C750C8 *...B.C.D.E.F.G.H*
0010A6F0+0010  50C150C2 50C350C4 50C550C6 50C750C8 *.A.B.C.D.E.F.G.H*

      LINES 0010A700 TO 0010A7C0 SAME AS ABOVE

0010A7D0+00F0  50C150C2 50C350C4 50C550C6 00000000 *.A.B.C.D.E.F....*
0010A7E0+0100  00000000 00000000 00000000 00000000 *.....*

      LINES 0010A7F0 TO 0010E6D0 SAME AS ABOVE

18:49:45  ***** END P I N U E S  OUTPUT *****
```

## PIN ルーチンユーザー出口

PIN ルーチンユーザー出口（エントリ名 ADASMXIT）は、次のために使用できます。

- 各種 PIN にパラメータを供給します。出口がインストールされていない場合、パラメータはデフォルト値に設定されます。
- Adabas が提供するもの以外のリカバリアクションを実装できるように、検出された条件を PIN ルーチンが起動される前に検査します。

ADASMXIT ロードモジュールは、ニュークリアスがロードできるように、ロードライブラリ連結、または z/OS システムリンクリストなどのシステムコールライブラリに配置しなければなりません。z/OS 上で ADASMP または Adaplex+ を実行する場合、ADASMXIT モジュールは権限を与えられたロードライブラリに配置しなければなりません。

PIN ルーチンユーザー出口は、アセンブラで書かれています。

### ユーザー 出口入力

出口は、次のレジスタセットで入力されます。

R13 Adabas PIN ルーチンセーブエリア

R14 リターンアドレス/AMODE

R15 エントリポイントアドレス

R0 機能コードは次のとおりです。

1-	ニュークリアス初期化
2-	アベンド
4-	レスポンスコード
5-	ニュークリアス終了

R1 パラメータリスト

+0	2つのユーザーワードのアドレス
+4	機能 2、4 用のコンディションディスクリプションブロック (CDB) のアドレス

### ユーザー 出口出力

出力はありません。リターンコードは無視され、15 以外の全レジスタが、変更されずに返さなければなりません。

### コンディションディスクリプションブロック

各プログラムチェック、異常終了、またはレスポンスコードエラーのために、コンディションディスクリプションブロック (CDB) と呼ばれるコントロールブロックが生成され、発生したイベント、発生場所、および発生時のレジスタとマシンの状態を示します。CDB は、エラー処理およびメッセージバッファリング機能に渡され、PIN ルーチンをコールするか、Adabas ユーザー出口を終了するかを判断するために使用されます。PIN ルーチンは、CDB を使用して、状況の発生について情報を取得します。

## 出口の修正およびリロード

PINルーチンユーザー出口は、ニュークリアスがアクティブなときに、修正、再アセンブル、および再ロードすることができます。新しく再アセンブルされた出口をロードするには、次のコンソールオペレータコマンドを発行してください。

`SMGT, XD= SXnn` PIN ルーチンユーザー出口を停止

`SMGT, XLOAD= SXnn` 修正バージョンの出口をロード

`SMGT, XA= SXnn` 出口をアクティブ化

『Adabas Online System マニュアル』で、このタスクを達成する別の方法を参照してください。

## PINAUTOR と出口の使用

PINAUTOR が有効な場合、次のことが可能です。

- PIN ルーチンユーザー出口がない場合、自動再起動から除外できる最大ファイル数は 10（デフォルト）です。そして、チェックポイントおよびセキュリティ（システム）ファイルを除く全ファイルを除外できます。レスポンスコードは、Adabas が一般的な規則として禁止しているもの以外は、すべて除外が可能です。
- PIN ルーチンユーザー出口がある場合、AUTOPARM を修正して、自動再起動から除外できる最大ファイル数を変更し、特定のファイルやレスポンスコードが排除されることを防ぐことができます。

## AUTOPARM の例

```
AUTOPARM DS 0D
MAXARPIN DC F'6' Maximum of 6 files can be excluded from autorestart
BADRSPS DC XL1'48' Response code 72 cannot be excluded from autorestart
          DC XL29'00' 28 more entries are possible
NOTFILE DC XL2'0041' File number 65 cannot be excluded from autorestart
```


```
DC XL48'00' 23 more entries are possible
```

### PINRSP と出口の使用

PINRSP が有効な場合、次が可能です。

- PIN ルーチンユーザー出口がない場合、特定のサブコードをチェックしないで全レスポンスコードが監視されます。各レスポンスコードは、最大10回監視されます。ADAN77メッセージは省略されません。
- PIN ルーチンユーザー出口がある場合、レスポンスコードを修正して、監視する特定レスポンスコードおよびサブコードと、各レスポンスコードの最大監視回数を指定することができます。レスポンスコード 201/202/203 の監視とともに N77MSG を YES に設定して、メッセージ ADAN77 を省略することができます。

### レスポンステーブルエントリの例

 **Note:** 255 までのレスポンスコードごとに、1つのエントリがあります。

```
XL5'000A000000'  
.  
.... subcodes (up to 3; specified in hexadecimal)  
.  
.... maximum number of times to invoke PINRSP for response code (default=10)  
.  
.... 00 don't log, 01 log
```

### 例


テーブルの9番目のエントリは、レスポンスコード9と一致します。

エントリ X'0105020311' は、レスポンスコード9、サブコード2、3、および16が記録されることを示します。レスポンスコード9は、最大5回記録されます。

# 27 ユニバーサルエンコーディングサポート (UES)

---

- 概要 ..... 192
- ワイド文字のエンコード ..... 192
- ワイド文字データサポート ..... 194

 **Note:** UES サポートを使用するには、バージョン7以降の Adabas SVC またはルーターを使用する必要があります。

このchapterでは、次のトピックについて説明します。

### 概要

---

ユニバーサルエンコーディングサポート (UES) は Adabas が次の処理を行なうことを可能にするデータベースオプションです。

- データ変換の実行
- ワイド文字エンコードの処理
- 照合順序などの国際化対応タスクの基礎の設定

データ変換は、異なるシステムとの通信時に必要です。例えば、英数字データを異なるコードページに変換したり、異なるマシンアーキテクチャに合わせて数値データを変換したりする必要があります（「[複数プラットフォームのサポート](#)」も参照）。

アジア言語環境ではワイド文字のエンコードが使用されます。多数の各種文字の要求のために、非1バイト文字セットが定義されました。また、Unicode（汎用文字セット）がより頻繁に使用されます（「[ワイド文字のエンコード](#)」も参照）。

頻繁にリストされる国際化対応タスクは、エンコードによって定義されるように、バイナリ順ではなく、言語固有の順番でデータを検索し、ソートすることです（「[ユーザー出口](#)」内の「[照合ディスクリプタ出口](#)」も参照）。

### ワイド文字のエンコード

---

たいていの場合、アジアのテキスト文字は、単一バイトを使用してエンコードすることはできません。例えば、10,000 文字以上の日本語セットは、1 文字あたりに2バイト以上を使用してエンコードされます。必要なエンコードにより、これらは2バイト文字セット (DBCS) またはマルチバイト文字セット (MBCS) と呼ばれ、大部分の西洋言語の特徴である1バイト文字セット (SBCS) と相反します。

以前のバージョンの Adabas は、DBCS にエンコードしたデータを英数字フィールドに保存していました。この解決方法には、次の問題点がありました。

- 英数字フィールドのデフォルトの空白が、2バイトまたはマルチバイト文字フィールドに必要な空白と異なる可能性があります。
- 長さで上書きするフィールドの切り詰めが、無効または変更された文字列になる場合があります（文字列が、文字境界ではなくバイト境界で切り捨てられます）。



- クライアントおよびサーバーが、異なるエンコードを2バイトまたはマルチバイト文字セットに使用する場合、クライアント/サーバーアプリケーションの実装が困難です。

Adabas バージョン7は、英数字フィールドのDBCS エンコードデータの保存をサポートしていますが、適切に定義されたエンコードおよび文字セットでデータを保存するために、ワイド文字 (W) フィールドフォーマットを導入しています。

ワイド文字フォーマットのデフォルトエンコードは、ストレージおよびユーザーに対してUnicode です。このデフォルトはユーザーおよびストレージレベルで、意図された使用方法に合わせたエンコードに変更することができます。

次の図では、日本語かな (最初の2つ) および漢字 (次の2つ) 文字は、メインフレームモデル (ミックス) および非モーダル (ピュア) にエンコードされます。

- EBCDIC ベースのマシンに使用する DBCS
- ASCII ベースのマシンに使用する JIS

Unicode では、固定2バイトのエンコードは他のエンコードよりも一般的であり、Adabas のデフォルトエンコードとして使用されます。

かな		漢字		
<SO>	, f , o   X 2	<SI>		
0E	4486 4496 4F58 48F2	0F		IBM-DBCS混在
	4486 4496 4F58 48F2			IBM-DBCSのみ
	82A9 82C8 8ABF 8E9A			シフトJIS (MS CP932)
<ESC>	\$ B \$ + \$ J 4 A ; z	<ESC>	( J	
1B	24 42 242B 244A 3441 3B7A	1B	28 4A	JIS
	304B 306A 6F22 5B57			Unicode

## ワイド文字エンコード例

モーダルエンコードは、1バイトおよび2バイト文字エンコードの間を前後にシフトします。混在する DBCS 文字列は、常に1バイトモードで開始および終了します。

2バイト文字のみのフィールド長は、偶数バイトでなければなりません。

EBCDIC エンコードでは、パディングまたは空白文字は、X'40' または X'4040' です。日立マシンでは、ワイドスペースは X'A1A1'、1バイトスペースは X'40' です。Adabas では、モード切換えなしで、1バイトスペースを2バイトモードで使用することができます。

## ワイド文字データサポート

---

Adabas は、次のワイド文字データをサポートします。

- 拡張英数字フォーマットフィールド
- ワイド文字フォーマットフィールド

既存のデータベースまたはファイルのために、エンコードは ADADBS ユーティリティを使用して英数字フィールドまたはワイド文字フィールドに割り当てられます（アンロード／リロードなし）。フィールドレベルのオプション NV（変換されていないフィールドをコール元に、またはコール元から渡す）が使用可能です。

### 拡張英数字フィールド

Adabas は、データベースおよびファイルレベルの両方にエンコードキーを定義することで、ワイド文字データをサポートするために英数字フィールドを拡張します。ファイルレベルのエンコードは、データベースエンコードよりも優先します。エンコードは、データを保存するフォーマットを指定します。これは、ローカルユーザーと交換するデータのデフォルトフォーマットとしても使用されます。

エンコードは、EBCDIC と互換性がなければなりません。つまり、スペース文字は X'40' でなければなりません。内部的な処理の理由により、次のエンコードファミリの中から1つだけがファイルにサポートされます。

- EBCDIC (1 バイト文字セット)
- 混在ホスト DBCS
- DBCS オンリーオプションを持つホスト DBCS

### 長所と短所

拡張英数字フィールドを使用する利点は次のとおりです。

- DBCS データを含む既存データベースを迅速にサポートします。
- Natural のようなアプリケーションが、変更なしで実行を継続します。
- 同一のエンコード／アーキテクチャからのコールに、Adabas ニュークリアスのロジック変更はありません（英数字フィールドが内部的なコーディングを定義しないため）。

短所は、DBCS がユニバーサルエンコードではなく、Unicode と異なり世界中の言語が使用するすべての文字をサポートしないことです。

## 制限

アプリケーションに対して、すべての英数字フィールドが同一のエンコードを持ちます。同一セッションの異なるフィールドに異なるエンコードを使用することはできません。

## 変換の考慮点

純粋な 1 バイト文字エンコードから変換するとき、可変フィールドのフィールド長は、変換レコードのシフトを必要とすることがあります。

## ワイド文字フィールド

Adabas は、フィールドにワイド文字 (W) フォーマットを定義します。W フォーマットフィールドは、英数字 (A) フォーマットフィールドに類似しており、エンコードキーはデータベースおよびファイルレベルの両方に定義されます。ファイルエンコードは、データベースエンコードよりも優先します。A フィールドエンコードと異なる点は次のとおりです。

- エンコードが指定されない場合、デフォルトの Unicode エンコードが使用されます。
- 内部的なエンコードにより、保存データのフォーマットが決まります。
- ユーザーエンコードにより、ユーザーに提示されるデータのデフォルトフォーマットが決まります。

A ディスクリプタは、内部的なエンコードで保存（およびソート）されます。

## 長所と短所

ワイド文字 (W) フィールドを使用する利点は次のとおりです。

- ローカルエンコードの文字セットは、ユーザーおよび特殊エンコードの全文字セットのスーパーセットになることができるので、問題の往復を避けます。
- 内部エンコードは、ECS によってサポートされる場合、UTF-8 の使用を認めるので、スペースが節約されます。
- ネイティブ Unicode (ユーザーエンコード)、標準 Java テキストエンコードをダイレクトに保存および取り出すことができます。

不利な点は次のとおりです。

- Natural およびその他の商品は、新しいフォーマットをすぐにはサポートしません。
- W フォーマットフィールドのサポートには、現在制限があります（次のセクションにリスト）。いくつかは、将来の Adabas リリースで解決される予定です。

## 制限

- アプリケーションに対して、すべてのワイド文字 (W) フィールドは同一のエンコードを持ちます。同一セッションの異なるフィールドに異なるエンコードを使用することはできません。
- W フィールドは、フォネティックディスクリプタまたはハイパーディスクリプタの親フィールドにすることはできません。
- 数字 (U、P、B、F) から W フォーマットへのフォーマット変換は不可能です。
- W フィールドは、カップリングフィールド (物理的またはソフト) の一部にすることはできません。
- W フィールドは、フォーマット選択基準 (条件フォーマット) の一部になることはできません。この制限は、主に基準入力 (フォーマットバッファ、サーチバッファ、およびユーティリティ) の 1 バイト文字エンコードに原因があります。
- W フィールドは、セキュリティバイバリュー基準の一部にすることはできません。
- W フィールドを編集マスク付きで使用することはできません。
- フォーマットバッファのリテラルは、変換不可能な 1 バイト文字列として処理されます。

## 特殊 DBCS フォーマットの変換規則

混在 DBCS および DBCS オンリーデータを使用する既存のアプリケーションから、円滑に変換できるように、特殊フォーマット変換規則が定義されました。

1. 1 バイトおよび 2 バイト文字のスーパーセットを構成するモーダル DBCS エンコードは、英数字フィールドでは混在 DBCS エンコードとして、ワイド文字フィールドでは DBCS オンリーエンコードとして扱われます。
2. ワイド文字 DBCS オンリーから、ユーザーの英数字混在 DBCS エンコードに変換するとき、エンコードの違いは無視されます。

例えば、英数字フォーマットおよびワイドフォーマットの両方のユーザーエンコードが DBCS として定義され、FDT では、フィールド AA は英数字として定義され、フィールド WW はワイドとして定義されている場合があります。

フォーマットバッファ	ユーザーバッファの値
AA[,A]	混合 DBCS
AA,W	DBCS オンリー
WW,A	DBCS オンリー
WW[,W]	DBCS オンリー

## 28 複数プラットフォームのサポート

---

■ 概要 .....	198
■ エンコード .....	199
■ ADACOX 変換出口 .....	200
■ バリュースタック内における上限値の変換 .....	200
■ データ変換の制約 .....	201
■ プラットフォームの考慮事項 .....	201

このchapterでは、次のトピックについて説明します。

### 概要

---

Adabas バージョン7より前では、Adabas バッファのデータを異なるマシンアーキテクチャ（ASCII、EBCDIC）間で変換する処理は Entire Net-Work によって行われていました。クライアントとサーバー（データベース）でのエンコードが異なるアプリケーションの使用が増え、Adabas 自体のデータの転送と変換の機能を拡張する必要が生まれました。Entire Net-Work によって、ターゲットデータベースに変換機能があるかどうかを判別され、変換機能がある場合はそこで変換されるように、未変換のデータがデータベースに渡されます。

Adabas 内でデータを変換する別の利点として、他の送信メカニズムをサポートできるようになりました。UES 対応データベースについては、Adabas バージョン7で、Web ベースのアプリケーション、または Software AG の Jdabas などの PC ベースのアプリケーションから、TCP/IP プロトコルを使用した z/OS メインフレームのデータベースへの Entire Net-Work アクセスが可能です。詳細については、『ADABAS オペレーションマニュアル』の ADARUN パラメータ TCP と TCPURL の説明を参照してください。

Adabas では次のようにデータが変換されます。

- クライアントアプリケーションは特殊エンコードを指定して、セッション開始時（OP コマンド）にそれを Adabas ニュークリアスに伝えることができます。
- LNKUES/ADALNKにより、コール元のアーキテクチャに応じて Adabas バッファデータが変換されます。
- 多くのユーティリティが特殊なエンコードおよびアーキテクチャ設定に対応しています。

EBCDIC から ASCII、また ASCII から EBCDIC への変換表が『Adabas インストールマニュアル』にあります。Adabas バージョン7に含まれるエンコードキーの表が『Adabas コマンドリファレンスマニュアル』と『Adabas ユーティリティマニュアル』にあります。

## エンコード

Adabas では 4 種類のエンコードが認識されます。これらは同時に指定できます。

エンコード	文字列エンコード
ファイル	内部で格納、処理されます。
デフォルトユーザー	Adabas ローカルコールインターフェイス要求と ADACMP DDEBAND のデフォルトとして使用されます。
ユーザー	ユーザーセッションまたは ADACMP 実行でデフォルトのユーザーエンコードに優先します。クライアントプログラムの特別なニーズに対応するために使用されます。
照合	許可されるソート順になります。照合は、一般にローカル定義で特定される言語や国の標準で定義できます。

ユーザーデータは変換が不要なので、Adabas では、処理速度を上げるため、ローカルのデフォルトのユーザーエンコードとファイルエンコードが同等とみなされます。ASCII アーキテクチャのリモート要求はデータベースのデフォルトの ASCII ユーザーエンコードを使用して変換されません。

2 バイト文字セットはネイティブのメインフレーム EBCDIC アーキテクチャエンコード (IBM、富士通、または日立のホスト DBCS) を使用して変換されます。

特殊なアプリケーションやリモートクライアントでは、セッションの開始時にそれぞれの処理環境に適合する特定のユーザーエンコードが選択されます。

アーキテクチャおよびエンコード間の上位と下位方向の互換性を確保するために、Adabas では、デフォルトユーザーエンコードおよびあらゆる特定のユーザーエンコードで定義されたすべての文字のスーパーセットを保持するファイルエンコードを使用します。ファイルエンコードなどのワイド文字フィールドのデフォルトのエンコードは、汎用文字セットの Unicode です。

ディスクリプタフィールドには照合エンコードが定義されます。このエンコードの値は、文化的に正しいソートキー、つまり文字のディクショナリを生成するようにプログラミングされた照合出口を呼び出すことでアルゴリズムに従って取得されます。照合エンコードは、英数字およびワイド文字フィールドの両方に定義できます。照合エンコード/出口は、英数字とワイド文字ディスクリプタフィールドのいずれか、またはその両方にファイルレベルで定義されます。

## ADACOX 変換出口

---

変換出口 ADACOX を使用できます。ADACOX では、UES 対応のデータベースに対して、Windows-1256 と EBCDIC アラビア語または EBCDIC ペルシア語のコードページ間でのコンテンツ依存の変換がサポートされます。

Windows 変換 1256 において、アラビア語の文字は *unshaped* ですが、サポートされる EBCDIC エンコードでは前後の文字に応じて *shaped* 形式が使用されます。さらに、特定の連続文字には結合形式が使用されます（例えば LAM-ALEF 合字）。

現在、論理順序と視覚順序の間における変換および対称文字の変換はサポートされていません。

UES 対応のデータベースでは、変換出口が常にロードされます。

2つのエンコード間で新しい変換が初めて使用される時、その変換が出口によってサポートされているかどうかは照会されます。変換がサポートされている場合は、その変換のときに出口が呼び出されます。変換がサポートされていない場合は、Adabas または Entire Conversion Services によって変換が実行されます。

ただし、ADACOX 内で定義される変換には、対応する ECS オブジェクトが必要です。例えば、420 から 1256 に変換する場合、文字セットのプロパティは ECS オブジェクトによって決定されます。

## バリュースタック内における上限値の変換

---

S 演算子を使用して検索を行うとき、通常、上限値は連続した X'FF' バイトです。

UES=YES の場合、ソースとターゲットのコードページによってデータの変換が制御されます。文字 X'FF' の変換は、ターゲットのコードページへのマッピングに依存します。したがって、X'FF' は変換後の値では X'FF' ではなくなる可能性があります。

例えば、819 (ISO 8859-1 Latin1) から 37 (EBCDIC) に変換する場合、分音符号付きの Latin の小文字 'y' は X'FF' から X'DF' にマッピングされます。その結果、検索機能で見つかるレコードが減るか、まったくなくなります。

この問題は次のように解決します。UES の値が YES で英数字またはワイドを変換する場合、すべての FROM-TO 検索／論理読み込み条件は、TO 条件内において値の最後の上限値が、内部検索値に変換されるときに保持され、値の変換から除外されるように処理されます。



**Note:** この解決策は値演算子 (EQ、GT、GE、LE、LT) には実装されていません。FROM-TO 検索条件 (S 演算子) の TO 値に限定されています。これは英数字フォーマットとワイドフォーマットのフィールド、またスーパーディスクリプタとサブディスクリプタの英数字／ワイドフォーマット部分に当てはまります。



## データ変換の制約

Adabas の変換には、Entire Net-Work の次の制約があります。

- 圧縮レコード (FB=C) は変換されません。
- テキストリテラルは変換されず、そのまま渡されます。レコードを読み込むとき、リテラルは変更されずに返されます (例えば FB AA, '-do not convert-',BB)。
- 変換でプリフェッチオプション P はサポートされていません。
- ET データは変換されません。ET データの読み込み時に EBCDIC の空白が埋め込まれます。

Adabas によるその他の制約は次のとおりです。

- CSCI で使用されるすべての C'Xn' コマンドコードで、コントロールブロックだけが変換され、バッファは変換されません。これは Adabas バージョン 7 のサーバーだけが対象です。
- Entire System Server (NPR) /XCOM アプリケーションは Adabas 変換の範囲に含まれません。これらのアプリケーションではアプリケーション内で変換を行う必要があります。
- OS/2 のアンパック数値記号 X'Dn' は使用されず、サポートされていません。
- Adabas には、フィールドレベル変換のユーザー変換出口はありません。このような出口は Entire Net-Work にあります。

## プラットフォームの考慮事項

異なるプラットフォームで動作する Adabas バージョン間の違いは徐々になくなってきましたが、アプリケーションを移植するときには次の点を考慮する必要があります。

	メインフレーム	オープンシステム	OpenVMS
固定小数点フィールド長	2 と 4 のみ	1,2,4,8	1,2,4,8
バイナリスーパーディスクリプタフォーマットのデフォルトは U (アンパック)	X	○	○
符号付きバイナリスーパーディスクリプタ	○	X	X
バイナリスーパーディスクリプタフォーマットの変換	○	X	X
MU フィールドと PE フィールドがあるスーパーディスクリプタ	○	X	X
スーパーフィールドとサブフィールド	○	X	X
親が浮動小数点フォーマットのスーパーディスクリプタ	X	○	○
アンパックフィールドの最大長	29	29	27

## 複数プラットフォームのサポート

---

	メインフレーム	オープンシステム	OpenVMS
パックフィールドの最大長	15	15	14
読み込みコマンドと ET/BT コマンドのプリフェッチオプション	○	X	X
ロング英数字 (LA) フィールドのオプション	○	X	X
フォーマットバッファのフィールド計算更新オプション	X	○	○
MC コマンド	X	○	○
バリュースタックからのハイパーフィールド値生成	X	○	○

また、メインフレームの ADALNK ユーザー出口で提供されるユーザーデータは ASCII マシンに送信されません。

## 29 ラージオブジェクト (LB) フィールドの基本

Adabas 8 にはラージオブジェクトフィールド (LB フィールド) があります。これらは、通常の英数字フィールドの 253 バイトや、LA フィールドの 16,381 バイトよりも多くのデータを格納できるフィールドです。ラージオブジェクトを含めるフィールドは新しい LB オプションで定義します。LB フィールドの値の理論上の最大サイズは 2 GB をわずかに下回ります。実際に使用できるサイズはもう少し小さくなります。

▶ **手順 29.1. Adabas 8 で LB フィールドを使用するには、次の手順に従います。**

- 1 つまたは複数のフィールドに LB オプションを割り当てて FDT を定義します。ADACMP COMPRESS 機能を使用して LB フィールドを作成できます。例としては、次のようなものがあります。

```
ADACMP COMPRESS
ADACMP FNDEF='1,AA,8,A,DE'   Key field
ADACMP FNDEF=...
ADACMP FNDEF='1,AZ,250,A,NU' Data field
ADACMP FNDEF='1,L1,0,A,LB,NV,NU,NB' Binary LOB field
```

ここではフィールド L1 をラージオブジェクトフィールドとして定義しています。NV オプションと NB オプションは、このフィールドで文字の変換と末尾の空白の圧縮を行わないことを指定します。NU オプションは、このフィールドの空値を省略することを指定します (NB オプションを指定するフィールドには、NU オプションまたは NC オプションのいずれかを定義する必要があります)。フィールド値は、どのような場合でも変更されません。LB フィールドオプションとその他すべてのフィールドオプション (NV と NB を含む) の詳細については、「フィールドオプション」を参照してください。

ADACMP で圧縮する入力データを指定する場合、入力には短い LB フィールド値 (253 バイトまで) または大きい LB フィールド値 (253 バイトを超える) を含めることができます。非圧縮の入力レコード内の LB フィールドに対応する箇所 (この例では L1)、LB データが後に続かない (LB 値は空なので) 長さ X'00000004' を指定することで空の LB 値を指定できます。COMPRESS の入力データの構造の詳細については、「入力データの必要条件」を参照してください。

Or:

LB フィールドを定義する FNDEF を指定して ADADBS NEWFIELD 機能を使用することで既存の FDT を更新することもできます。Adabas Online System にも同等の機能があります。この操作を行う場合、新しい基本ファイルを定義してロードする必要はありません。また次の手順を省略して [手順 3](#) に進むことができます。

- 2 ADALODLOAD 機能を使用して、LB フィールドがある基本ファイル (空の可能性があるので) をデータベースにロードします。Adabas は、LB フィールド値を LOB ファイルと呼ばれる個別のファイルに格納します。このファイルは、基本ファイルと呼ばれる LB フィールドを含むファイルと密接に関連付けられます。

次の手順で空の LOB ファイルを別個にロードしますが、LOB ファイルと基本ファイルはどちらを先にロードしても、また同時にロードしてもかまいません。例としては、次のようなものがあります。

```
ADALOD LOAD FILE=11,NAME='BASE-FILE'  
  
ADALOD      LOBFILE=12  
  
ADALOD      MAXISN=100000,DSSIZE=5000B  
  
ADALOD      ...  
  
ADALOD      SORTSIZE=100,TEMPSIZE=100
```

LOBFILE パラメータで、この基本ファイルに対応する LOB ファイルのファイル番号を指定しています。LOB に関連付けられた ADALOD パラメータの詳細については、「LOAD: ファイルのロード」を参照してください。



**Note:** 基本ファイルと LOB ファイルの組み合わせも、LB フィールド (基本ファイル) のあるファイルがすでに表示されている場合は、LOB ファイルをロードするだけで設定できます。

ADACMP COMPRESS 手順の //DDAUSBA 出力データセットを、ADALOD LOAD 機能の //DDEBAND 入力データセットとして指定します。

- 3 ADALOD LOAD 機能を使用して空の LOB ファイルをロードし、このファイルを前の手順でロードするか、[手順1](#)で更新した基本ファイルに関連付けます。LOB ファイルと基本ファイルは任意の順序で、また同時にロードすることもできます。

```
ADALOD LOAD FILE=12,LOB,NAME='LOB-FILE'

ADALOD      BASEFILE=11

ADALOD      MAXISN=500000,DSSIZE=500000B,NISIZE=4000B,UISIZE=40B

ADALOD      ISNREUSE=YES,DSREUSE=YES,INDEXCOMPRESSION=YES

ADALOD      ...

ADALOD      SORTSIZE=100,TEMPSIZE=100
```

LOB ファイルタイプは、ADALOD で、FDT が事前に定義された LOB ファイルをロードする必要があることを指定します。空の LOB ファイルをロードする場合、//DDEBAND 入力データセットを指定する必要はありません。

BASEFILE パラメータで、この LOB ファイルに対応する基本ファイルのファイル番号を指定します。LOB に関連付けられた ADALOD パラメータの詳細については、「[LOAD：ファイルのロード](#)」を参照してください。



**Note:** 基本ファイルと LOB ファイルの組み合わせも、LB フィールド (基本ファイル) のあるファイルがすでに表示されている場合は、LOB ファイルをロードするだけで設定できます。

- 4 ADAREP を使用してデータベースレポートを実行し、基本ファイルと LOB ファイルがどのように表示されるかを確認します。
- 5 Adabas ニュークリアス (ADARUN) のパラメータ NISNHQ、NH、LP、LDEUQ、LWP、NAB、および LU を再評価して調整し、LB フィールドを処理するために十分なリソースをニュークリアスで使用できるようにします。これらのパラメータの詳細については、「[Adabas の初期化 \(ADARUN ステートメント\)](#)」を参照してください。これらのパラメータの調整については、「[旧バージョンの Adabas からの移行](#)」を参照してください。
- 6 LB フィールド値を含むデータを基本ファイルにロードするアプリケーションを作成または調整します。バックグラウンドでは Adabas が LB フィールド値 (非常に短いものを除く) を LOB ファイルに格納しますが、この処理はアプリケーションに対して透過的に行われず、アプリケーションプログラムのコマンドは常に基本ファイルに対して発行され、LOB ファイルの存在を意識する必要はありません。
- LB フィールドを操作するには Natural 4.2 を使用する方法があります。LOB ファイルの LB フィールド (この例では L1) は Natural で動的変数にマッピングされます。アプリケーションプログラムを通常の方法で作成することもできます。Natural では、大きな LB フィールド値の処理に必要なフォーマットで Adabas コールが自動的に発行されます。

- アプリケーションプログラムで Adabas に対してダイレクトコールを発行することもできます。大きな LB フィールド値 (32KB を超えるデータ) を使用したコールには、Adabas 8 の ACBX ダイレクトコールインターフェイスを使用する必要があります。ACB ダイレクトコールインターフェイスを使用するコールは、指定するすべての LB フィールド値が単一の 32KB バッファに収まる場合に使用できます。詳細については、「ACBX インターフェイスダイレクトコールの指定」を参照してください。例えば、ACBX ダイレクトコールで次の情報を指定できます。

ダイレクトコールコンポーネント	指定
ACBX	<pre>ACBXCMD = N1 (insert record)  ACBXFNR = 11 (base file number)  ...</pre>
最初のフォーマット/レコードバッファセグメントペア	<pre>FB1 = 'AA,8,A,L1L,4,B,...,AZ,250,A.'  RB1 = 'KEY-1      ' X'000186A0' ... 'Some arbitrary data...'</pre>
2 番目のフォーマット/レコードバッファセグメントペア	<pre>FB2 = 'L1,*.'  RB2 = X'100,000 bytes of binary data'</pre>

2 つのバッファセグメントペアの場所と長さは、4 つの Adabas バッファ記述 (ABD) で指定する必要があります。これらのバッファ記述はここには示しません。詳細については、「Adabas バッファ記述 (ABD)」を参照してください。

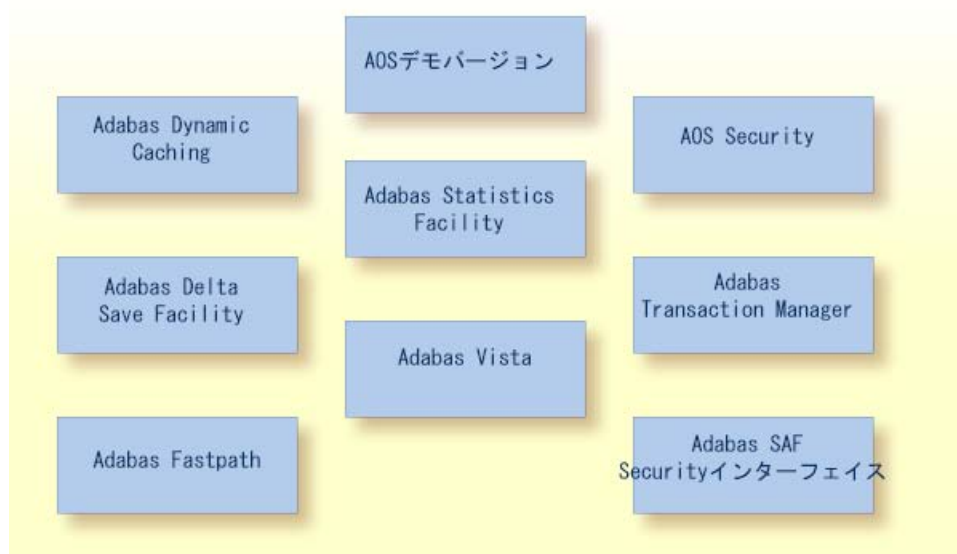
最初のフォーマットバッファセグメント (L1L,4,B) 内の LB フィールドの長さインジケータ (L) は、L1 フィールドの固有値を、最初のレコードバッファセグメント内の対応する場所に格納することを指定します。これは LB フィールド値長さ 100,000 バイト (16 進数で 186A0) を示します。長さインジケータの詳細については、「長さインジケータ (L)」を参照してください。

2 番目のフォーマットバッファセグメント (L1,\*) 内のアスタリスク (\*) を使った長さ表記は、LB フィールドの固有値 (追加長さ情報なし) を、2 番目のレコードバッファセグメント内の対応する場所に格納することを指定します。アスタリスクを使った長さ表記の詳細については、「アスタリスク (\*) を使った長さ表記」を参照してください。

# 30

## Adabas Online System デモバージョン

Adabasには、次の図に示すように、デモバージョンのAdabas Online System (AOS) と、その他の選ばれた Adabas 製品および機能のオンラインサービス利用が含まれています。



### AOS デモバージョン

この章は、AOS デモバージョンの操作と使用方法を説明します。その他の Adabas 製品および機能のオンラインサービス利用についての説明は、該当する製品および機能のマニュアルに記載されています。Adabas Fastpath (AFPLOOK) と Adabas Vista (AVILOOK) で使用可能なデモサービスについては、「[AFPLOOK](#)」と「[AVILOOK](#)」で説明しています。AOSセキュリティについては、『Adabas Security Manual』を参照してください。

この情報は次の項目で構成されています。

- 概要
- メインメニュー機能
- セッションの監視
- チェックポイントのリスト
- ファイルメンテナンス
- データベースメンテナンス
- システムオペレータコマンド機能
- データベースレポート

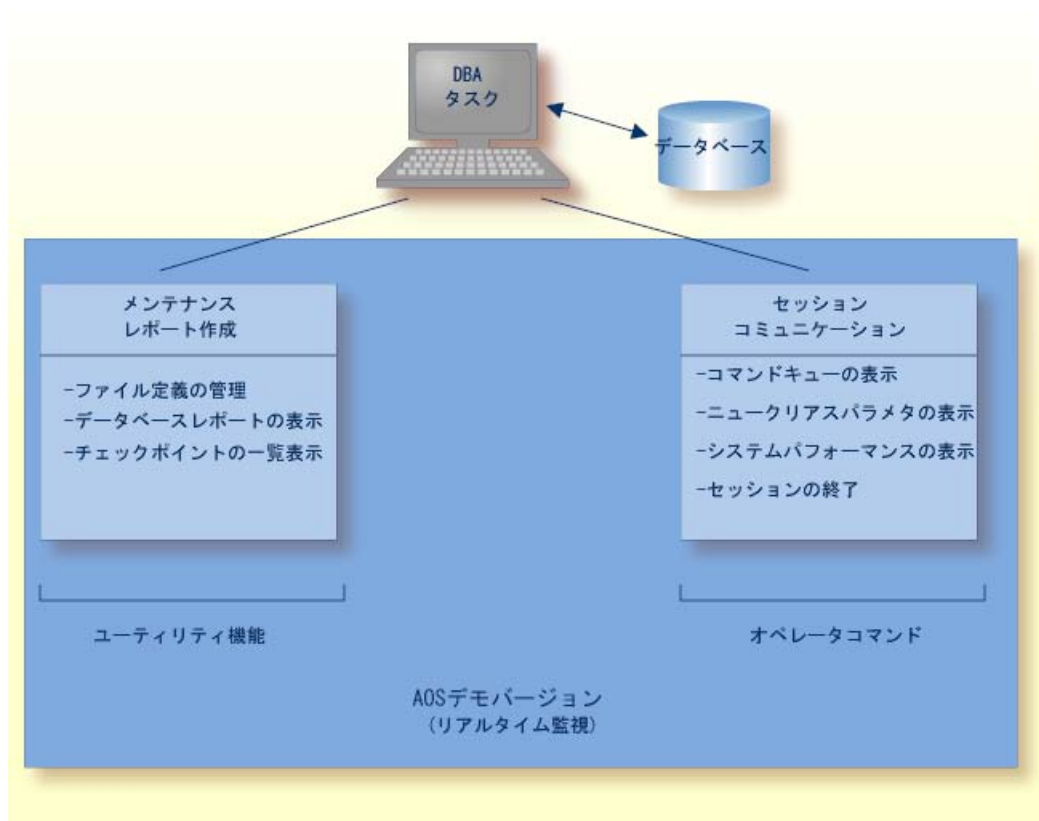


# 31 概要

---

■ AOS デモバージョンでできること .....	210
---------------------------	-----

AOS デモバージョンは、Adabas オペレータコマンドおよびユーティリティに相当する機能を持っており、Adabas データベースの分析と制御に使用します。



### AOS デモバージョンの概要

AOS デモバージョンのインストールについては、『Adabas インストールマニュアル』を参照してください。

AOS アドオン製品は、追加のユーティリティ機能およびオペレータコマンドに対応するサービスを含みます。詳細については、Adabas Online System のマニュアルを参照してください。

このchapterでは、次のトピックについて説明します。

## AOS デモバージョンでできること

DBA は、AOS デモバージョンを使用して、Adabas セッションの活動中に、Adabas データベースの状態を監視することができます。メニューオプションを使用して、DBA はリソースやホールドキューの状況、スペースの割り当て、ファイルやデータベースのパラメータを表示したり、新しい FDT を作成したり、現在の Adabas セッションを終了したりできます。

パフォーマンスの分析とデータベース操作のモニタリングについては、AOS デモバージョンではシステムをユーザーとある特定のシステムリソースの両面から見ることができます。例えば、次を実行できます。

- ホールドキューの状況のチェック
- ニュークリアスのパラメータの確認
- コマンド、ファイルの使用、システムのパフォーマンス情報の監視
- ファイルレイアウトとエクステンタステータスのリスト
- VOLSER によるデータベースのファイル分布のリスト

Adabas セッション全体の制御のために、AOS デモバージョンを使用して次のことができます。

- 新規 FDT の作成
- Adabas ニュークリアスセッションの終了 (ADAEND)




## 32      メインメニュー機能

---

■ AOS デモバージョンでのデータベースの指定 .....	215
■ プログラム機能 (PF) キーの使用 .....	215
■ メニューオプションの選択 .....	215
■ ヘルプ .....	216
■ AOS デモバージョンメッセージ .....	216

## メインメニュー機能

AOSデモバージョンを開始するには、NaturalアプリケーションSYSAOSにログオンし、NEXTプロンプトが表示された場合は「MENU」と入力します。

 **Note:** 完全なバージョンの AOS がシステムにインストールされている場合、代わりに「MENU」と入力してください。詳しくは Adabas Online System のドキュメントを参照してください。

```

14:40:37          ***** A D A B A S  BASIC  SERVICES *****          2002-05-29
                   - Main Menu -                                     PMAIN02

Code  Basic Services          Code  Other Services
-----
A     Session monitoring      1     Adabas Cache Facility
C     Checkpoint maintenance  2     Delta Save Facility
F     File maintenance        *     Trigger Maintenance
M     Database maintenance    4     AOS Security
O     Session opercoms        5     Transaction Manager
R     Database report         6     Adabas Statistics
*     Space calculation       7     Vista
?     Help                    8     Fastpath
.     Exit                    9     SAF Security
-----

Code ..... _
Database ... 1955      (WIS1955)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit
  
```

メインメニューには、Adabas Online System で使用可能な機能が表示されます。デモバージョンで使用できない AOS の機能は、画面上でアスタリスク (\*) でマークされます。

メインメニューから、使用可能な [Basic Services] の機能か、システムにインストールされた [Other Services] の中の 1 つにアクセスすることができます。ご使用のサイトにインストールされたその他のサービスは、強調表示されます。

メインメニューは、デモバージョンでの DBA の主なタスクを示します。

コード	機能
<b>A</b>	セッションの監視-ニュークリアスパラメータ、セッションおよびスレッドステータス情報、ホールドキューの ISN、およびニュークリアスモジュールのメンテナンスレベルを表示します。
<b>C</b>	チェックポイントメンテナンス-チェックポイント情報をリストします。
<b>F</b>	ファイルメンテナンス-新規ファイルに新規 FDT を定義できます。
<b>M</b>	データベースメンテナンス-デモバージョンでは使用できません。

コード	機能
O	Session opercoms - 拡張されたエラーリカバリ機能で使用する PIN モジュールの追加または削除、現在ロードされている PIN ルーチンの表示とそのアクティブ化と非アクティブ化、ロックされたファイルの表示、セッションの正常終了 (ADAEND) を行います。
R	データベースレポート - 全体的なデータベースレイアウト情報、データベースファイルのテーブル、およびファイルの詳細情報を表示します。

この章のこの後のセクションでは、AOS デモバージョンの主な機能とメニュー／画面構造を、メインメニューに表示される順序で説明します。

このchapterでは、次のトピックについて説明します。

## AOS デモバージョンでのデータベースの指定

AOS デモバージョンがインストールされているデータベースがデモバージョン機能のデフォルトデータベースになります。ただし、任意のアクティブな Adabas ニュークリアスセッションのデータベースを指定することができます。別のデータベースを指定するか AOS デモバージョンを終了するまで、同じデータベースを参照し続けます。

## プログラム機能 (PF) キーの使用

有効な PF キーおよびその機能は、各画面の下段にリストされます。次のプログラム機能 (PF) キーは、AOS デモバージョンのすべての画面に表示されます。一部の画面にはその他のナビゲーションキーが表示されます。

PF1	ヘルプ
PF3	前の画面に戻る
PF12	メインメニューに戻る

## メニューオプションの選択

機能またはオプションを選択するには、[Code] フィールドにオプションコードを入力します。

メインメニュー機能を選択すると、その機能に対する選択メニューが表示されます。

### ヘルプ

---

すべての AOS デモバージョンメニューで、PF1 を使用して、現在のメニューについて簡単に説明する画面を表示することができます。「[プログラム機能 \(PF\) キーの使用](#)」を参照してください。

### AOS デモバージョンメッセージ

---

AOS デモバージョンは各機能の完了を知らせるために確認メッセージを出力します。エラーが起きた場合は、参照番号とエラーを記述したメッセージが出力されます。

エラーを分析する前にヘルプ情報 (PF1) を見て、最後に行ったステップで必要なものを見落としていないか調べ操作をもう一度行ってみます。

レスポンスコード 22 が返った場合には、AOS デモバージョンの実行中に Adabas セッションが終了し、再スタートしたことを示すので、アプリケーションを終了し、再スタートしなければなりません。



# 33 セッションの監視

---

■ ADARUN パラメータの表示 .....	219
■ ホールドキューの表示 .....	220
■ システムステータスおよびスレッド使用の表示 .....	221
■ メンテナンスレベルの表示 .....	223

## セッションの監視

Adabasセッションの監視機能では、Adabasの主なリソースが表示されます。システムパフォーマンスやパフォーマンスの問題の原因を究明するとき、Adabasセッションの処理を監視する機能が最も有効です。

```
14:38:19          ***** A D A B A S BASIC SERVICES *****          2002-05-29
                    - Session Monitoring -                               PAC0002

Code  Service                                Code  Service
-----
*    Display cluster members                *    Refresh nucleus statistics
*    Maintain user profiles                 *    Current session statistics
D    Display parameters                     *    Maintain TCP/IP URL
*    Modify parameters                     U    Display session utilization
Q    Display queues                         Z    Display maintenance levels
?    Help
.    Exit
-----

Code ..... _
Database ID .. 1955      (WIS1955)          NucID .. 1022

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                Menu
```

セッションの監視環境を使用して、マルチプロセッシング（パラレルサービスまたはクラスタサービス）環境で Adabas ニュークリアスを監視できます。パラレルサービスまたはクラスタサービスデータベースのDBIDをセッション監視メニューに入力すると、後続の画面には、監視するクラスタのニュークリアス ID を指定するためのフィールドが表示されます。

セッション監視メニューの各機能について説明します。

コード	機能
D	Adabas ニュークリアス (ADARUN) パラメータの表示
Q	ホールドキューの表示
U	セッション状況およびスレッド使用の表示
Z	Adabas ニュークリアスモジュール (メンテナンスレベルと適用 ZAP) の表示

このchapterでは、次のトピックについて説明します。

## ADARUN パラメータの表示

Adabas ニュークリアス (ADARUN) パラメータを参照することができます。

パラメータを参照するには、[Session Monitoring] メニューのオプション D を選択します。

パラメータの表示のための画面は 3 画面あります。

```

16:33:03          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - Display Parameters -                               PACP002

Modify parameters below, as required:
----- Pools -----
Sort Area          (LS) .. 19968
Int. User Buffer    (LU) .. 65535
Buffer Pool        (LBP) .. 127936
Format Pool        (LFP) .. 12000
ISN List Table     (LI) .. 10000
Seq. Cmd. Table    (LQ) .. 2500
Work Pool          (LWP) .. 500000
Attached Buffer     (NAB) .. 35
Security Pool      (LCP) .. 10000
UQ-DE Pool         (LDEUQP) .. 5000
Flush I/O Pool     (LFIOP) .. 125000
Err. Recovery(SMGTBUFF) .. 0

----- Queues -----
Command Queue      (NC) .. 100
Hold Queue         (NH) .. 9000
User Queue         (NU) .. 700

----- Time Windows -----
Transaction Time   (TT) .. 300
Max Transaction Time (MXTT) .. 3600
Nonactivity ACC-User (TNAA) .. 300
Nonactivity ET-User (TNAE) .. 300
Nonactivity EXU-User (TNAX) .. 300
Max Nonactivity Time(MXTNA) .. 3600
Time Limit Sx-Cmds (TLSCMD) .. 286
Max Time for Sx-Cmds(MXTSX) .. 3600
Command Time       (CT) .. 9000
SYNS60 Interval   (INTNAS) .. 3600

Page 1 of 3
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help          Exit          +          Menu

```

```

16:33:03          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - Display Parameters -                               PACP002

Modify parameters below, as required:
----- Miscellaneous -----
ReadOnly session (READONLY) .. NO
UTI only session (UTIONLY) .. NO
OPEN required    (OPENRQ) .. NO
Ignore DIB Entry (IGNDIB) .. NO
Local nucleus    (LOCAL) .. NO
Number of Threads (NT) .. 4
Non DE Search    (NONDES) .. YES
Log AOS/DBS Update (AOSLOG) .. NO
Batch Support     (BATCH) .. NO
Data Protection Area (LP) .. 500
Ignore Work Part 4 (IGNTPC) .. NO

----- User Specific Limits -----
Hold Queue Limit (NISNHQ) .. 2200
CIDs per User    (NQCID) .. 75
ISNs / TBI Element (NSISN) .. 51

----- Buffer Pool -----
Bufferflush Dur. (TFLUSH) .. 1
Parallel LFIOP I/O (FMXIO) .. 60
Async. by Vol-Ser (ASYTVS) .. YES

```

## セッションの監視

```
WORK-Part-4 Area      (LTPC) .. 0
WORK-Part-2 Area      (LWKP2) .. 0
```

Page 2 of 3

Modify parameters below, as required:

```
---- Command Logging ----      ----- Protection Logging -----
Command Logging .. NO          PLOG required          (PLOGRQ) .. NO
LOGCB ..... NO                DUAL PLOG Size        (DUALPLS) .. 0
LOGFB ..... NO                DUAL PLOG Device      (DUALPLD) .. 0
LOGRB ..... NO                ----- Other Services -----
LOGSB ..... NO                Triggers and Procedures (SPT) .. NO
LOGVB ..... NO                Delta Save Facility    (DSF) .. NO
LOGIB ..... NO                Cache Facility         (CACHE) .. NO
LOGIO ..... NO                Transaction Manager    (ATM) .. NO
LOGUX ..... NO                TCP/IP Support         (TCPIP) .. NO
LOGSIZE ..... 8904            Ext. Error Recovery    (SMGT) .. NO
DUAL CLOG Size ... 0          2 Phase Commit Support (TPC) .. NO
DUAL CLOG Dev. ... 0          Review Support         (REVIEW) .. NO
```

Page 3 of 3

```
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit
```

## ホールドキューの表示

[Session Monitoring] メニューの [Queue displays] (オプション Q) を選択すると、次のメニューが表示されます。

```
09:00:20          ***** A D A B A S BASIC SERVICES *****          1997-01-29
                   - Queue Displays -                               PACQ002
```

```
Code Service
-----
* Display User Queue Elements
* Display Command Queue
H Display Hold Queue
? Help
. Exit
-----
```

```
Code ..... _
Max No. Elements ... 100
Last Activity ..... 0          (elapsed time in seconds)
```

Selection Criteria

```
ET-ID (User-ID) .. _____ User Type ... ____
```

```
Job Name ..... _____
```

```
Terminal ID ..... _____
```

```
Database ID ..... 105          (RD-105)
```

Command ==>

```
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit      Clear UID                Menu
```

オプション H は、現在ホールドステータスにある ISN リストを表示します。

キューが現在のところ空の場合、その旨を示すメッセージが表示されます。

## システムステータスおよびスレッド使用の表示

[Session Monitoring] メニューの [Resource utilization] (オプション U) を選択すると、[Resource Utilization] メニューが表示されます。

```
11:44:10          ***** A D A B A S  BASIC SERVICES *****          1997-01-30
                   - Resource Utilization -                          PACU002

Code      Service
-----
*         Command usage
*         File usage
*         High water marks (pools/queues)
*         Workpool (LWP) usage
*         PLOG status
S         System status
T         Thread usage
*         WORK status
?         Help
.         Exit
-----

Code ..... _
File Number .. 0
Database ID .. 105      (RD-MPM105)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                Menu
```

各オプションでは、表示された値をリフレッシュ (PF4) することができるので、Adabas システム機能を長時間監視するときには使用すると便利です。

システムステータス

[System status] (オプション S) を選択すると、ASSO、DATA、WORK、PLOG の各データセットに対する I/O カウント、リモートおよびローカルのコール分布、現セッションのその他のステータス情報が表示されます。

```

16:44:13          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - System Status -          PACUS02

                Physical
                Reads          Writes          Call Distribution
                -----          -----          -----
ASSO              132              29 Remote Logical .....          0
DATA               3              9 Remote Physical .....          0
WORK               4              29 Local Logical .....          145
PLOG               0              0 Local Physical .....          0
Logical Reads .....          194 Logical Reads (binary) ..... 000000C2
Buffer Efficiency ....          1.4 No. of HQEs active .....          0
                                     No. of UQEs in User Queue ..          2
Format Translations ..          0 No. of CQEs waiting in CQ ..          0
Format Overwrites ....          0
                                     Total intern. Autorestarts .          0
Throw Backs for ISN ..          0 No. of PLOG switches .....          0
Throw Backs for Space.          0 No. of Bufferflushes .....          8

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help              Exit          Refresh          Menu
    
```

スレッド使用

[Thread usage] (オプション T) を選択すると、定義された全 Adabas スレッドのテーブル、それぞれのステータス、アクティブな各スレッドで現在処理中のコマンドタイプ、および現セッションの各スレッドで処理されたコマンド数が表示されます。

```

11:47:18          ***** A D A B A S  BASIC SERVICES *****          1997-01-30
DBID 105          - Thread Status -          PACUT02

Nr. I Thread Status          I Command Type          I Nr. CMDs I
-----
1 I Active          I Simple Cmd.          I 18992 I
2 I Not active          I          I 109 I
3 I Not active          I          I 0 I
4 I Not active          I          I 0 I
5 I Not active          I          I 0 I
I          I          I I
    
```

## メンテナンスレベルの表示

[Session Monitoring] メニューの [Display maintenance levels] (オプション Z) を選択すると、Adabas ニュークリアスモジュールについての情報が表示されます。

```

13:43:09      ***** A D A B A S  BASIC SERVICES *****      1999-06-09
DBID 105      - Display Maintenance Levels -                      DPACZ02

Select Module Name: _____
-----
ADARUN  RUNMVS  Date 1998-10-27, Version 7.1. 0, Zap Base AO10000
          RUNIND  Date 1998-10-27, Version 7.1. 0, Zap Base AI10000
ADATSP   Date 1998-10-30, SM Level 00, Zap Level 0000
          Zaps 0034 0040 0043 0083 0084 0099
ADATCP   Date 1998-10-30, SM Level 00, Zap Level 0000
          Zaps 0136
ADAMSG   Date 1998-10-30, SM Level 00, Zap Level 0000
ADAIOR   Date 1998-10-29, SM Level 00, Zap Level 0000
ADAIOS   Date 1998-10-29, SM Level 00, Zap Level 0000
          Zaps 0001 0003 0004 0005 0007
ADANC0   Date 1998-11-01, SM Level 00, Zap Level 0000
          Zaps 0036

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                --                -                +                Menu

```

各モジュールのメンテナンスレベルが表示されます。モジュールに適用されたすべてのZAPもリストされます。

画面の上の方にある [Select Module Name] フィールドに特定のモジュール名を入力して、モジュールのリストを制限することができます。開始値も使用することができます。例えば、ADANC3を指定すると、ADANC3モジュールだけの情報が表示されます。ADANC\*を指定すると、ADANCで始まる名前のモジュールをすべてリストします。





# 34 チェックポイントのリスト

メインメニューの [Checkpoint maintenance] (オプション C) を選択すると、[Checkpoint Maintenance] メニューが表示されます。

```
14:41:59          ***** A D A B A S  BASIC  SERVICES *****          2002-05-29
                  - Checkpoint Maintenance -                          PCP0002

                                Code      Service
                                -----
                                C         List checkpoints
                                *         Delete checkpoints
                                ?         Help
                                .         Exit
                                -----

Code ..... _
Date(YYYY-MM-DD) . 0000-00-00
Ext. CP-list ..... N
Checkpoint Name .. ALL
Database ID ..... 1955      (WIS1955)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help              Exit                               Menu
```

オプション C でチェックポイントファイルの現在の状態をリストします。

[External CP-list] フィールド (CPEXLIST オペレーティング制御パラメータを上書きできる) の設定によって、結果は基本形式または拡張形式でリストされます。

[Date] フィールドに、画面のとおり形式で日付を入力して、特定の日付でチェックポイントリストを開始することができます。

## チェックポイントのリスト

チェックポイントリストを書き込むデータベースを指定できます。

[Checkpoint Type] フィールドの [ALL] 指示を次のいずれかに変更して特定のチェックポイントタイプにリストを限定することができます。

SYNC ニュークリアス初期化  
SYNF ユーザーオープン EXF  
SYNP NUC なしユーティリティ  
SYNS ADARES  
SYNV ボリューム ID 変更  
SYNX ユーティリティ  
SYN1 ADASAV DB 開始  
SYN2 ADASAV DB 開始  
SYN4 ADASAV ファイル開始  
SYN5 ADASAV ファイル開始

チェックポイントタイプについては『Adabas ユーティリティマニュアル』の ADAREP に関する説明を参照してください。

次の画面は通常のチェックポイントリストを表示しています。

```
15:12:22          ***** A D A B A S Basic Services *****          1997-01-30
DBID 105          - List Checkpoints -          PCPC002

  CP   CP   Date      Time      PLOG      Block      Vol/Ser      User Job Name
  Name Type                                     Number      Number      Number      Type
  ---- -
SYNC  01  1996-04-01  00:29:41                                     MPM105
SYNS  60  1996-04-01  02:00:37                                     ADABAS
SYNP  06  1996-04-01  02:43:55                                     PMS105SS
SYNV  0A  1996-04-01  02:59:51                                     PMS105SS
SYNV  0A  1996-04-01  02:59:51                                     PMS105SS
SYNV  0A  1996-04-01  02:59:51                                     PMS105SS
SYNV  0A  1996-04-01  02:59:51                                     PMS105SS
SYNV  0A  1996-04-01  02:59:51                                     PMS105SS
```

SYNC	01	1996-04-01	03:12:30				MPM105
SYNS	60	1996-04-01	04:34:43				ADABAS

次の画面は各チェックポイントについての追加情報を提供する拡張チェックポイントリストです。

```

12:58:49          ***** A D A B A S Basic Services *****          1997-01-31
DBID 105          - List Checkpoints -          PCPC002

  CP   CP   Date      Time      PLOG      Block      Vol/Ser      User Job Name
  Name Type                                Number     Number     Number     Type
  ---- -
SYNC  01  1996-04-01  00:29:41                                MPM235
      SESSION OPEN          IGNDIB = N , FORCE = N
SYNS  60  1996-04-01  02:00:37                                ADABAS
      STATISTIC RECORD
SYNP  06  1996-04-01  02:43:55                                PMS235SS
      SAVE DB
SYNV  0A  1996-04-01  02:59:51                                PMS235SS
      SAVE DB          VOL-SER = 502461  SESSION = 933
SYNV  0A  1996-04-01  02:59:51                                PMS235SS
      SAVE DB          VOL-SER =          SESSION = 933
SYNV  0A  1996-04-01  02:59:51                                PMS235SS
      SAVE DB          VOL-SER = 502215  SESSION = 933
    
```



# 35 ファイルメンテナンス

メインメニューのオプションFを選択すると、[File Maintenance] メニューが表示されます。

```
14:42:25          ***** A D A B A S  BASIC  SERVICES *****          2002-05-29
          - File Maintenance -          PFL0002
          Code      Service
          ----      -
          C      Define/modify FDT
          *      Release descriptor
          *      Delete existing file
          *      Define new file
          *      Modify file parameters
          *      Reorder file online
          *      Refresh file to empty status
          *      Allocate/deallocate file space
          *      Maintain expanded files
          ?      Help
          .      Exit
          ----      -
Code ..... _
File No ..... 0      Descriptor Name .. ___
Database ID .. 1955      (WIS1955)

Command ==>
```

## ファイルメンテナンス

```
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help           Exit           Menu
```

[File Maintenance] メニューのオプション C を選択すると、[FDT/SDT Definition / Modification] が表示されます。

```
15:34:30          ***** A D A B A S  BASIC  SERVICES  *****          1998-07-30
                   - FDT/SDT Definition / Modification -          PFLC002

Code      Service
-----
*      Add new field(s)
*      Change field length
D      Define new FDT
*      Online invert
*      Define/add SDT
?      Help
.      Exit
-----

Code ..... _
File No. .... 50
Field Name ... _
Database ID .. 105      (RD-MPM105)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help      Def. File Exit           Menu
```

[FDT/SDT Definition / Modification] メニューのオプション D を選択すると [Define FDT] が表示されます。この画面では、新規ファイルの新規 FDT を定義できます。

```
5:13:34          ***** A D A B A S  BASIC  SERVICES  *****          1997-02-12
DBID 105          - Define FDT -          PFLCD02

File Number ... 200          New FDT ... Y
Enter Field Description(s) :

I Level I Name I Length I Format I Options          I
I-----
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
I  _   I  _   I  _   I  _   I  _   _   _   _   _   I
```

```

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                                Menu
    
```

既存ファイルの FDT をこのオプションで再定義することはできません。

この機能は、Adabas ユーティリティの ADACMP COMPRESS 機能に対応します。





## 36 データベースメンテナンス

メインメニューのオプション M を選択すると、[Database Maintenance] メニューが表示されます。

```
14:42:42          ***** A D A B A S  BASIC  SERVICES *****          2002-05-29
                  - Database Maintenance -                               PDM0002

          Code      Service
          ----      -
          *          Add new dataset to ASSO/DATA
          *          Increase/decrease ASSO/DATA
          *          List/reset DIB block entries
          *          Recover unused space
          *          Uncouple two ADABAS files
          ?          Help
          .          Exit
          ----      -

Code ..... _
File No. .... 0
Coupled File .. 0
Database ID ... 1955 (WIS1955)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help          Exit          Menu
```

データベースメンテナンス機能は、デモバージョンでは使用できません。



# 37 システムオペレータコマンド機能

---

▪ 拡張エラーリカバリ .....	237
▪ ロックファイルの表示 .....	239
▪ ユーザーの停止 .....	240
▪ セッションの正常終了 (ADAEND) .....	241

## システムオペレータコマンド機能

メインメニューの [Session opercoms] (オプションO) を選択すると、次のメニューが表示されます。

```

14:43:03          ***** A D A B A S  BASIC  SERVICES *****          2002-05-29
                  - Session Opercoms -                               PACI002

Code  Service                                Code  Service
-----
*    Allocate/Deallocate CLOG/PLOG          S    Stop user(s)
E    Extended Error Recovery                T    Termination Commands
*    Force Dual CLOG or PLOG switch         *    Manage Online Utilities
L    Lock or unlock files                   *    User Table Maintenance
*    Reset ONLINE-DUMP-Status
?    Help
.    Exit
-----

Code ..... _
Userid(ETID) ... _____
CLOG/PLOG Ind .. _      Global.. _
Database ID .... 1955   (WIS1955)          NucID .. 1022

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                Menu

```

次の機能が AOS デモバージョンにあります。

コード	機能
E	拡張エラーリカバリ機能 (現在のPINルーチンの表示、アクティブ化、または非アクティブ化) が使用する PIN モジュールを、追加または削除します。
L	ロックされているファイルを表示します。
S	特定ユーザー、特定ファイルやジョブの全ユーザー、またはアクティブでない全ユーザーを停止します。
T	セッションを正常終了します (ADAEND)。

このchapterでは、次のトピックについて説明します。

## 拡張エラーリカバリ

[SessionOpercoms] メニューのオプションE (拡張エラーリカバリ) を選択すると、[Extended Error Recovery] メニューが表示されます。

```

14:44:35          ***** A D A B A S  BASIC SERVICES *****          1999-05-12
                  - Extended Error Recovery -                          DPACIE2

          Code      Service
          ----      -
          *          Display message buffer
          *          Display/modify environment
          *          Display/modify Exit routines
          M          Add/Delete PIN modules
          P          Display/modify PIN routines
          *          Refresh threshold and alert exits
          *          SNAP a nucleus dump
          ?          Help
          .          Exit
          -----

Code ..... -
Start Address ..          End Address ...
Database ID .... 823      (RD-CK-823)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                                Menu

```

このメニューでは、次の操作を行うことができます。

- PIN モジュールの追加または削除
- 特定 PIN ルーチンの表示、アクティブ化、または非アクティブ化

## PIN モジュールの追加／削除

[Extended Error Recovery] メニューのオプション M (PIN モジュールの追加／削除) を選択すると、現在使用可能な PIN モジュールのリストが表示されます。

```
13:42:45          ***** A D A B A S  BASIC SERVICES *****          1999-06-18
DBID 823          - Add/Delete PIN Modules -          PACIEM2

Mark entries with 'A' to Add or 'D' to Delete:

      M  Module      Description      Message
      -  - - - - -  - - - - -  - - - - -
      _  ADAMXY      Standard Nucleus PIN Routines
      _  PINAAF        SAF Security
      _  PINAFP        Adabas Fastpath
      _  PINATM        Adabas Transaction Manager
      _  PINAVI        Adabas Vista
      _  PINRSP        Adabas Response Code Handler
      _  PINUES        Universal Encoding Support
```

PIN モジュールをメモリにロードするには、モジュール名の隣の M 列に「A」を入力します。

このコマンドは、Adabas ニュークリアスにアクセスできるライブラリに出口モジュールが存在する場合にのみ成功します。

PIN モジュールをメモリから削除するには、モジュール名の隣の M 列に「D」を入力します。

PIN モジュールをメモリから削除すると、すべての関連 PIN ルーチンも削除されます。

これらの機能は、次の拡張エラーリカバリのオペレータコマンドと同じです。

```
SMGT, {ADDPIN | DELPIN}=module-name
```

## PIN ルーチンの表示／修正

[Extended Error Recovery] メニューのオプション P (PIN ルーチンの表示／修正) を選択すると、メモリに現在ロードされている PIN のリストが表示されます。

```
13:08:49          ***** A D A B A S  BASIC SERVICES *****          1999-06-16
DBID 105          - List/Modify PIN Routines -          PACIEP2

Mark entries with 'A' Activate, or 'D' Deactivate:          Total Pins: 012

      M  Condition      Error Location      Status  Uses  Module      Message
      -  - - - - -  - - - - -  - - - - -  - - - - -  - - - - -
      _  000C1000  All Locations      Active   0    ADAMXY
      _  000C2000  All Locations      Active   0    ADAMXY
      _  000C3000  All Locations      Not Act  0    ADAMXY
      _  000C4000  All Locations      Active   0    ADAMXY
```

_	000C5000	All Locations		Active	0	ADAMXY	
_	000C6000	All Locations		Active	0	ADAMXY	
_	000C7000	All Locations		Not Act	0	ADAMXY	
_	000C8000	All Locations		Active	0	ADAMXY	
_	000C9000	All Locations		Active	0	ADAMXY	
_	000CB000	All Locations		Active	0	ADAMXY	
_	000CF000	All Locations		Active	0	ADAMXY	
_	00047000	All Locations		Active	0	ADAMXY	
PF1-----	PF2-----	PF3-----	PF4-----	PF6-----	PF7-----	PF8-----	PF12-----
Help		Exit	Refr	--	-	+	Menu

リストに表示される PIN ルーチンについて、実行の原因となった状態、現在のステータス、使用回数、PIN ルーチンが配置されているモジュールが表示されます。

PIN のステータスをこの画面から変更するには、PIN 番号の隣にある M 列に次の値を入力します。

A PIN のアクティブ化

D PIN の非アクティブ化

変更したら、PF4 を使用して画面を更新します。

これらの機能は、次の拡張エラーリカバリのオペレータコマンドと同じです。

```
SMGT,DISPLAY=PINS
SMGT,{ACTPIN | DEACTPIN}=pin-number
```

## ロックファイルの表示

[Session Opercoms] メニューのオプション L を選択すると、次の画面が表示されます。

```
16:02:10          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
                  - Lock / Unlock Files -                          PACIL02

Code      Service
-----
D         Display locked files
*         Lock file for all users
*         Advance lock file
*         Lock file except for UTI/EXF users
*         Unlock file from general lock
*         Release an advance lock
*         Unlock file from UTI/EXF lock
?         Help
.         Exit
-----
```

```
Code ..... _
File Number .. 30
UTI/EXF Ind .. U
Database ID .. 105      (RD-105)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                                Menu
```

このメニューのオプション D を選択すると、 [Display Locked Files] 画面が表示されます。

```
10:57:45          ***** A D A B A S  BASIC SERVICES *****          1998-07-31
DBID 105          -  Display Locked Files  -          PACID02

M  Fnr.   Lock Status                M  Fnr.   Lock Status
-  ----  -
1   1      Locked for ALL users
35  35     Locked except for UTI
50  50     Locked except for EXU/EXF
55  55     Locked for ALL users
60  60     Locked for ALL users

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                --      -      +      Menu
```

## ユーザーの停止

[SessionOpercoms] メニューのオプション S (ユーザーの停止) を選択すると、 [Stop Users] メニューが表示されます。

```
18:26:02          ***** A D A B A S  BASIC SERVICES *****          1999-07-29
                        -  Stop Users  -          PACIS02

Code  Service
----  -
*     Stop users using file
*     Stop inactive users
*     Stop users by jobname
```



```

          *   Stop a selected user
          ?   Help
          .   Exit
-----
Code ..... _
File Number ..... _____
Last Activity .... _____ (elapsed time in seconds)
Job Name ..... _____
Purge UQE(s) ..... N
Selected Userid ..
Database ID ..... 105   (RD-105)

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help      Disp UQ   Exit      Clear UID                               Menu

```

ユーザー停止機能はデモバージョンでは使用できません。

## セッションの正常終了 (ADAEND)

[SessionOpercoms] メニューのオプションTを選択すると、[Session Termination] メニューが表示され、セッションを正常に終了 (ADAEND) することができます。

```

11:43:00          ***** A D A B A S  BASIC  SERVICES  *****          1997-01-30
                   - Session Termination -                               PACT002

          Code   Service
          ----   -
          A      Normal session termination (ADAEND)
          *      Cancel session immediately (CANCEL)
          *      Stop session                    (HALT)
          ?      Help
          .      Exit
          -----

Code ..... _
Database ID .. 105   (RD-MPM105)
Current nr. of users in User Queue ... 9
Nr. of users with open transactions .. 0

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help      Exit                               Menu

```

セッションを終了する前に、終了してもよいかどうかを確認するメッセージが表示されます。

# 38 データベースレポート

---

- ファイルの表示 ..... 244
- 全体的なデータベースレイアウトの表示 ..... 248

データベースレポート機能は、Adabas ADAREP ユーティリティの一部の機能に該当し、全般的な情報と詳細な情報を表形式またはレポート形式で表示します。

```
14:43:26          ***** A D A B A S BASIC SERVICES *****          2002-05-29
                  - Database Report -                               PDR0002

Code      Service
-----
*         List files with crit. no. of extents
*         Display field description table (FDT)
F         Display file(s)
G         General database layout
*         List VOLSER distribution of database
*         Display ASSO/DATA block (RABN)
*         Display unused storage
?         Help
.         Exit
-----

Code ..... _
File No ..... 0_____ Password ..
Database ID .. 1955 (WIS1955)
VOLSER ..... _____

Command ==>
PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help           Exit           Menu
```

AOS デモバージョンで使用可能なオプションを選択して、データベースレベルの全般的な情報とデータベースファイルのテーブル、およびファイルの詳細情報を参照することができます。

コード	機能
<b>F</b>	指定データベースの全ファイルのリストまたは特定ファイルの詳細情報を表示します。
<b>G</b>	指定データベースの全体的なレイアウトを表示します。

このchapterでは、次のトピックについて説明します。

## ファイルの表示

特定のファイルを指定しない場合、オプションFは指定データベースの全ファイルをリストします。ファイルを指定した場合、オプションFは、そのファイルの詳細なレイアウト情報を表示します。物理デバイスおよびファイルレイアウト情報は特定ファイルに対してだけ有効です。

## 指定データベースのファイルリストの表示

[Database Report] メニューの [File No] フィールドにファイル番号を指定しなかったり 0 を指定した場合、指定データベースのファイルリストが表示されます。

```

09:24:38          ***** A D A B A S  BASIC  SERVICES  *****          1997-02-20
          DBID 105          - Display Files -          PDRF002

Fnr  File Name          Loaded          Top-ISN          Max-ISN          Ext. Pad %  Ind.  %Used
          NUAD  A  D  ACISEXU  A  D
-----
  1  EMPLOYEEES          1993-06-15          1110          5511 1111  3  3  NNISNNN 68 88
  2  MISCELLANEOUS          1993-06-15          1779          5511 1111  3  3  NNISNNN 32 88
  4  AUTOMOBILES          1993-06-15          1000          5511 1111  3  3  NNISNNN 34 36
  5  PERSONNEL          1993-06-15          1000          5511 1111  3  3  NNISNNN 38 52
  6  FINANCE          1993-06-15          1000          5511 1111  3  3  NNISNNN 52 52
  7  GDMUSIC          1992-05-01          3292          16535 1111  3  3  NNNNSNNN 81 95
  8  SAMPC-REV311DATA  1992-05-01          44679          100593 1111  3  3  NNNNSNNN 79 99
  9  RD-NAT217-FUSER  1993-09-23          163272          175005 1111  3  3  NNISNNN 76 99
 10  RD-PRD314-FDIC  1992-05-01          60016          63387 1111  3  3  NNNNSNNN 73 90
 11  REV320-DBFILE  1992-05-01          4442          11023 1111 10 10  NNNNSNNN 42 82
 12  REV340-DBFILE  1993-02-15          52008          63387 1111 10 10  NNNNSNNN  6 13
 13  SASRM-ZAP-TEST  1992-05-01           11          1377 1111  3  3  NNNNSNNN 93  4
 14  SASRM-ZAPSYS  1992-05-01           5          1377 1111  3  3  NNNNSNNN 28  4
 16  SAGDT-PRD-FDIC  1992-05-01          25649          30315 1111  3  3  NNISNNN 57 85

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help      Repos      Exit      --      -      +      Menu

```

PF2 (再位置づけ) キーを押すと、ファイルリストの開始値を新規に入力できるウィンドウが表示されます。ファイル番号を入力すると、[Display Files] リストの先頭にそのファイルの情報が表示されます。

[Display Files] 画面には、各ファイルについて次の詳細情報が表示されます。

- ファイル番号およびファイル名
- ファイルがデータベースにロードされた日付
- 現在ファイルで使用されている最高の ISN とファイルに許可されている最高の ISN
- 現在割り当てられている論理エクステントの数：アソシエータの N はノーマルインデックス、U はアッパーインデックス、A はアドレスコンバータ、データストレージは D ファイルには最大 5 つの論理エクステントの割り当てが可能です。
- アソシエータとデータストレージに定義されたブロックパディングファクタのパーセント。

■ インジケータの内容は次のとおりです。

A	ADAM オプション。A は ADAM ISN またはディスクリプタ選択ファイル。N は非 ADAM ファイル。
C	カップリングファイル (C) 、または非カップリングファイル (N) 。
I	ISN の再利用が可能 (I) 、または不可能 (N) 。
S	データストレージブロックの再利用が可能 (S) 、または不可能 (N) 。
E	データファイルが暗号化されている (E) か、されていない (N) か。
X	拡張ファイル (X) または通常のファイル (N) 。
U	ファイルの USERISN オプションが有効である (U) 、または有効でない (N)

■ アソシエータおよびデータストレージで現在ファイルに使用されている割り当てスペースのパーセント

### 特定ファイルの情報の表示

[Database Report] メニューにシステムファイル番号を指定すると、 [Display File Layout] 画面に移り、そのファイルのファイルレイアウト情報が表示されます。

```

18:27:37          ***** A D A B A S  BASIC  SERVICES  *****          1999-01-28
DBID 105          - Display File Layout -          PDRF012
*****
* File 75      *   UES-FILE
*****
Records loaded ..... 1107          Date loaded ..... 1999-01-26 12:18:17
Top ISN ..... 1107
Max ISN expected ... 1502          Max Compr Rec Lngth .. 4816
Minimum ISN ..... 1          Asso/Data Padding .... 3%/3%
Size of ISN ..... 3 Bytes          Highest Index Level .. 3
Number of Updates .. 0
ISN Reusage ..... NO          USERISN ..... NO
Space Reusage ..... YES          MIXDSDEV ..... NO
ADAM File ..... NO          PGMREFRESH ..... NO
Ciphered File ..... NO          NOACEXTENSION ..... NO
Coupled Files ..... NONE          Universal Encoding ... YES
Blk per DS Extent .. 0
Blk per UI Extent .. 0
Blk per NI Extent .. 0          Length of Owner ID ... 0

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help          Exit          Refresh          Menu
    
```

ファイルの情報は PF4 でリフレッシュできます。

Enter キーを押して、スペース割り当てについての追加情報を表示することができます。

[Display File Layout] 画面には、ファイルについて次の情報が表示されます。

- ファイル番号およびファイル名
- ファイルの現在のレコード件数
- ISN 情報：現在ファイルで使用されている最高の ISN、ファイルに予想される最高の ISN (ADALOD ユーティリティの MAXISN パラメータを参照)、ファイル内のレコードに割り当てることができる最小 ISN (ADALOD ユーティリティの MINISN パラメータを参照)、ファイルに 3 バイトと 4 バイトのどちらの ISN を使用しているか、ISN を再利用可能かどうか
- ファイルが最後にロードされた後の合計更新回数
- その他のファイルオプション設定：データストレージスペースを再利用可能かどうか、ファイルが ADAM オプション、サイファオプション、USERISN オプションを指定してロードされたかどうか、ファイルが別のファイルと物理的にカップリングされているかどうか、データストレージエクステンツが別のデバイスタイプにあってもいいかどうか、E1 コマンドを使用してファイルをリフレッシュできるかどうか、ファイルで MAXISN 設定を増やせるかどうか
- データストレージ、アッパーインデックス、およびノーマルインデックスエクステンツに許されたブロック数
- ファイルが最後にロードされた日付および時刻
- ファイルに許可される最大圧縮レコード長 (ADALOD ユーティリティの MAXRECL パラメータを参照)
- アソシエータおよびデータストレージに対するパディングファクタ
- ファイルに対して現在有効な最高インデックスレベル
- ファイルが最後にロードされた後に更新によって変更されたファイルのブロック数の合計
- マルチクライアントファイルのオーナー ID 長
- ユニバーサルエンコーディングサポート (UES) が使用されているかどうか

最初の [Display File Layout] 画面で ENTER を押すと、次のスペース割り当ておよび使用情報が表示されます。

```

18:33:41          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - Display File Layout -          PDRF022

File 75

      IDeviceIListI  Space allocated  I      From      To      I  Unused      I
      I Type ITypeI  Blocks / Cyls. I      RABN    RABN    I  Blocks / Cyls.I
-----I-----I-----I-----I-----I-----I-----I-----I-----I
      I      I      I      I      I      I      I      I      I      I
ASSO I 3380 I AC I      3      0 I      724 -      726 I      0      0 I
      I 3380 I UI I      15     0 I      747 -      761 I      0      0 I
      I 3380 I NI I      20     0 I      727 -      746 I      0      0 I
      I 3380 I NI I      56     0 I      762 -      817 I      2      0 I

```

I	I	I		I		I		I
DATA I	3380 I	DS I	116	0 I	216 -	331 I	29	0 I

PF1-----	PF2-----	PF3-----	PF4-----	PF6-----	PF7-----	PF8-----	PF12-----
Help		Exit	Refresh				Menu

## 全体的なデータベースレイアウトの表示

オプション G を選択すると、[Display General DB-Layout] 画面に全般的なデータベース情報が表示されます。

```

18:43:07          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
  DBID 105          - Display General DB-Layout -          PDRG002

Isolated
Database Name ..... RD-105
Database Number ..... 105
Database Version ..... 7.1
Database Load Date ..... 1998-10-21 14:40:47
System Files ..... 19 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0
Maximum Number of Files .. 100
Number of Files Loaded ... 5
Highest File Loaded ..... 75
Trigger File Number ..... 14
Size of RABN ..... 4 Bytes
Current Log Tape Number .. 5
Delta Save Facility ..... Inactive
Recovery Aid Facility ... Inactive
Universal Encoding Sup. .. Yes

PF1----- PF2----- PF3----- PF4----- PF6----- PF7----- PF8----- PF12-----
Help                Exit                Refresh                Menu
  
```

Enter キーを押して、UES コード、カップリング、およびスペース割り当てについての追加情報を表示することができます。

[Display General DB Layout] 画面には、ファイルについて次の情報が表示されます。

- データベースの名前と番号
- Adabas データベースソフトウェアのバージョンレベル
- データベースがロードされた日付と時刻



- データベースに割り当てられた Adabas システムファイル数
- データベースに許可された最大ファイル数、現在ロードされているファイル数、現在使用されている最大ファイル番号
- ファイルに使用されている RABN (3 バイトか 4 バイトか)
- データベースの最新データプロテクションログテープの番号
- Adabas Delta Save Facility または Adabas Recovery Aid (ADARAI) がデータベースに対してアクティブであるかどうか
- ユニバーサルエンコーディングサポート (UES) が使用されているかどうか

ユニバーサルエンコーディングサポート (UES) が使用されている場合、最初のファイルレイアウトの表示画面で ENTER を押すと、現在のコード値がリストされます。

```

18:51:22          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - Display General DB-Layout -          PDRG002

Universal Encoding Support Enabled

UES Encoding Keys:

Alpha File Encoding ..... 37
Wide File Encoding ..... 4095
Alpha ASCII Encoding ..... 437
Wide User Encoding ..... 4095

```

どのような場合でも、最初の [Display File Layout] 画面で ENTER を押すと、次のスペース割り当ておよび使用情報が表示されます。

```

18:52:01          ***** A D A B A S  BASIC SERVICES *****          1999-01-28
DBID 105          - Display General DB-Layout -          PDRG002

      IDeviceI      Total Number of I      Extents in Block I DD-Names I
      I Type I      Blocks / Cyls. I      From To I      I
-----I-----I-----I-----I-----I-----I-----I-----I
      I      I      I      I      I      I      I      I
ASSO I 3380 I      14231      50 I      1      14231 I DDASSOR1 I
      I      I      I      I      I      I      I      I
DATA I 3380 I      6741      50 I      1      6741 I DDDATAR1 I
      I      I      I      I      I      I      I      I
WORK I 3380 I      3592      30 I      1      3592 I DDWORKR1 I

```

## データベースレポート

---

PF1-----	PF2-----	PF3-----	PF4-----	PF6-----	PF7-----	PF8-----	PF12-----
Help		Exit					Menu

# 39 AFPLOOK

---

▪ AFPLOOK の有効化 .....	252
▪ 運用上のデフォルト .....	252
▪ デフォルトの調整 .....	252
▪ AFPLOOK パラメータ .....	253
▪ AFPLOOK レポート .....	255

コマンドサンプラー（AFPLOOK）は、FASTPATHに適したコマンド構造についてレポートを作成し、FASTPATHから最高の結果を得るためのパラメータを決定するために使用できます。

サンプラーの制御および表示はSYSAFPを使用してオンラインで行うことができますが、デフォルトのモジュール AFPLUKD を使用すればSYSAFPなしでも制御できます。いずれの場合も、その時点でサンプルが有効になっていれば、サンプラーはADAENDに指定した時間にDDPRINT上にレポートを出力します。

## AFPLOOKの有効化

---

AFPLOOKは次のADARUNコマンドを使用して有効にします。

```
ADARUN FASTPATH=YES
```

サンプルを有効にするには、SYSAFP管理センタを使用します。

## 運用上のデフォルト

---

AFPLOOKには、次を制限することでコマンドの分析中に使用されるメモリの大きさを制御できるように運用上のデフォルトが設定されます。

- 最大サンプリングファイル数。
- 同時ユーザー数。

いずれかのパラメータを超えた場合、超過した分は無視されて最大数の情報として出力されます。この方法で、AFPLOOKはデータベースコマンドの処理負荷のサンプリングを全般的に監査し、FASTPATHの最適化パラメータを決定します。「[AFPLOOKパラメータ](#)」で説明しているように、サイトの要件に合わせて運用上のデフォルトを修正できます。

## デフォルトの調整

---

AFPLOOKは、必要に応じて使用します。AFPLOOKは、デフォルトでは非アクティブになっています。AFPLOOKをアクティブにしたり設定したりするには、SYSAFP管理センタを使用します。ただし、管理センタを使用しなくてもAFPLOOKを制御できます。

デフォルトモジュールAFPLUKDを作成することで、SYSAFPを使用しなくてもAFPLOOKを設定できます。Adabasリリーステープの一部として提供されるサンプルジョブSAGLUKDを参照してください。

「**AFPLOOKパラメータ**」で説明している各パラメータは、AFPLUKDを使用して事前に設定できます。

## AFPLOOK パラメータ

ここでは、サンプルの範囲の定義および必要なメモリ量の制限に使用されるAFPLOOKパラメータについて説明します。

- 最大ファイル数
- ファイルごとのコマンド／ディスクリプタ数
- 最大同時ユーザー数
- ユーザー当たりの最大コマンド ID 数
- 最大処理コマンド数
- ジョブ名
- 選択ファイル数
- デモサンプリング (AFPLUKD のみ)
- 実際のサンプリング (AFPLUKD のみ)
- レポートタイトル (AFPLUKD のみ)

### 最大ファイル数

サンプリングされる最大ファイル数。

分析テーブルに最大ファイル数を登録すると、それ以上のファイルはサンプリングされませんが、コマンド数には含まれるため、次に実行するときはこのパラメータの値を大きくする必要があるかどうかの判断材料になります。

デフォルト：64

### ファイルごとのコマンド／ディスクリプタ数

ファイルごとのコマンド／ディスクリプタエントリの最大数。

このパラメータは、パラメータとともに使用され、必要なメモリ量を制限します。ファイルの最大エントリ数に達すると、最後のエントリは累積値になります。Adabas コマンドタイプ L1、L2、S8、および S9 に必要なエントリは 1 つだけです。

デフォルト：32

## 最大同時ユーザー数

同時ユーザーのテーブルの最大サイズ。

すべてのユーザーエリアが一度に使用される場合は、最も前に使用されたユーザーエリアを解放することで新しい要求が実行されます。このようなエリアの再使用が行われた回数は、サマリに注釈として出力されます。再使用率が高い場合は、このパラメータを調整する必要があります。

デフォルト：100

## ユーザー当たりの最大コマンドID数

処理ユーザーごとに同時にサンプリングする Adabas コマンド ID の最大数。

このパラメータは、最大同時ユーザー数パラメータとともに使用され、必要なメモリ量を制限します。この最大数を超えたコマンドIDのコマンドは無視され、拒否されたものとしてレポートされます。拒否率が高い場合は、このパラメータを調整する必要があります。

デフォルト：10

## 最大処理コマンド数

サンプリングされる最大コマンド数。

デフォルト：制限なし

## ジョブ名

サンプリングを特定のジョブ名に制限するために使用します。ジョブ名には、ワイルドカード文字として1つ以上のアスタリスク (\*) を使用できるため、サンプルでアスタリスク (\*) が使用されている文字位置を無視して、名前と一致するすべてのジョブを選択できます。

デフォルト：なし

## 選択ファイル数

指定したファイルにサンプリングを制限するために使用します。

このオプションは、最大ファイル数を超えている場合やファイルのアクティビティが分かっており、詳細な分析が必要な場合に便利です。

デフォルト：すべてのファイル

### デモサンプリング (AFPLUKD のみ)

デフォルトのモジュール AFPLUKD で AFPLOOK を事前設定するときに、ライセンスされた FASTPATH がない場合に使用します。

DEMO= は、デモの AFPLOOK をデフォルトでサンプリングするかどうかを設定します。

設定は ON/OFF です。

デフォルト：ON

### 実際のサンプリング (AFPLUKD のみ)

デフォルトのモジュール AFPLUKD で AFPLOOK を事前設定するときに、ライセンスされた FASTPATH がある場合に使用します。

REAL= は、実際の AFPLOOK をデフォルトでサンプリングするかどうかを設定します。

設定は ON/OFF です。

デフォルト：OFF

### レポートタイトル (AFPLUKD のみ)

デフォルトのモジュール AFPLUKD で AFPLOOK を事前設定するときに使用します。

このパラメータは、出力レポートのタイトルを設定します。タイトルは最大 30 文字までです。

デフォルト：AFPLOOK レポートサマリ

## AFPLOOK レポート

---

ここでは、AFPLOOK レポートで使用できる情報の種類について説明します。

- ファイルのサマリ
- 潜在的な最適化のサマリ
- サンプルコマンド分析

■ レポートのパラメータ

ファイルのサマリ

レポートのこのセクションでは、ファイルコマンドのサマリが表示されます。

```

-----
FNR  CC  DESC  DIRECT ACC          RC  SEQUENTIAL  SEQUENCES
-----
 20  L1  --          1
     L2  --          4          4
     L3  CC          1
     L9  AA          1
     L9  BB          2
     L9  CC          2          1          1
     S1  AA          3          1
TOTALS          7          8          21(18%)

                EXCLUDED COMMANDS:          2
                ALREADY PREFETCHED:          3

(UPDATES 2,INSERTS 1,DELETES 1) (MAX.RBL DA 0,SEQ 32)
-----
    
```

列	説明
FNR	Adabas ファイル番号。
CC	Adabas コマンドコード。
DESC	プライマリディスクリプタ。
DIRECT ACC	最適化できるダイレクトアクセスの最大コマンド数。
RC	最適化できる RC コマンドの最大数。
SEQUENTIAL	最適化できるシーケンスコマンドの最大数。
SEQUENCES	シーケンシャルコマンドの番号の元となったシーケンス数。最適化のシーケンス係数は、これらの2つの数から計算できます。

右端の番号は、特定ファイルに対してサンプリングされたコマンドの数の合計と、サンプリングされたすべてのファイルコマンドに対するパーセンテージを示しています。サンプリング対象が多い場合には、この番号を使って、最適化を検討する必要があるファイルを容易に判断できます。

また、ファイルに対して除外されたコマンドも、その除外の理由とともにリストされます。

最後の行には、更新コマンドと、最適化可能なダイレクトアクセスまたはシーケンシャルコマンドの最大レコードバッファ長が出力されます。



## 潜在的な最適化のサマリ

レポートのこのセクションには、すべてのファイルに対してサンプリングされたコマンドの合計数のサマリと、表示されたすべてのコマンドに対するパーセンテージが示されます。除外されたコマンドも同様にレポートされます。

----- POTENTIAL OPTIMIZATION SUMMARY -----			
SAMPLED COMMANDS		MAXIMUM OPTIMIZATION	
SAMPLED FILE COMMANDS	116 ( 77%) <-----	SEQUENTIAL:	55 (47%)
		DIRECT ACCESS:	32 (27%)
		RCS:	4 ( 3%)
EXCLUDED COMMANDS	33 ( 22%)		
TOTALS	149 (100%)		91 (61%)

最適化の最大回数は、潜在的な最適化の推定値です。シーケンシャルコマンド、ダイレクトアクセス、およびRCSの合計は、サンプリングされたファイルコマンド合計数に対するパーセンテージとして示されます。合計数は、すべてのコマンドに対するパーセンテージで示されます。

これらの数は、FASTPATH を使用して、推定される潜在的な最適化の数を示します。実際の最適化数は、各ユーザー環境に固有のさまざまな要素によって異なります。サンプルを解釈するときは、Software AG にお問い合わせください。

## サンプルコマンド分析

レポートのこのセクションには、コマンド分析情報が出力されます。

----- COMMAND ANALYSIS -----			
REJECTED COMMANDS			
MAX. USERS EXCEEDED:	0		
MAX. CIDS EXCEEDED:	0		
MAX. FILES EXCEEDED:	0	0 ( 0%)	
EXCLUDED COMMANDS			
BAD COMMANDS:	4		
NON-FILE COMMANDS:	7		
NON-FILE RCS:	2		
EXCLUDED FILE COMMANDS:	8		
UPDATE COMMANDS:	4		
ALREADY PREFETCHED:	8	33 ( 22%)	
SAMPLED FILE COMMANDS		116 ( 77%)	
ALL COMMANDS SEEN		149 (100%)	

次の数が出力されます。

- 処理されたコマンドの種類を示します。
- 前のセクションで説明した内容も出力されます。

拒否されたコマンドは、ユーザー、CID、およびファイルごとに分類されます。合計のパーセンテージが高い場合、レポートされた見積もりの中に誤った評価が存在している可能性があります。

除外されたコマンドは、次のカテゴリに分割されます。

カテゴリ	説明
無効コマンド	予期しない Adabas レスポンスコード。
非ファイルコマンド	OP CL ET C1 RE などの、ファイルに属さないコマンド。ファイルコマンド HI LF RI も含まれます。
非ファイル RC	すべての RC コマンドおよび AFPLOOK によって記録されていない CID の RC。
除外ファイルコマンド	L4 L5 L6 S4 S5
更新コマンド	A1 A4 E1 E4 N1。
プリフェッチ済みコマンド	プリフェッチまたはマルチフェッチがすでに設定されているシーケンシャル最適化の対象となるすべてのコマンド。

## レポートのパラメータ

レポートのこのセクションには、

- レポートの作成に使用される重要なパラメータが出力されます。また、
- 必要なパラメータも示されます。

```

-----
PARAMETERS USED
MAX. FILES:      64  FILES NEEDED:      5
  ..MAX. DE:     32  OVERFLOWS:      0
MAX. USERS:     100  HIGH USERS:      15
  ..MAX.CID:     10  HIGH CIDS:      4
* REUSED USER AREA OCCURRENCES:      0
MAX.RECORDS: NO LIMIT
-----

```

# 40 AVILOOK

---

■ AVILOOK の有効化 .....	260
■ SYSAVI – AVILOOK の選択 .....	260
■ SYSAVI – AVILOOK の使用 .....	261

コマンド分析サンプラー（AVILOOK）は、ファイルへのアクセスに使用されるコマンド構造についてのレポートを作成することで、Adabas Vista のパーティションオプションの効果が認められるファイルを判断するのに使用できます。

サンプラーは、SYSAVIを使用してオンラインで制御および出力できます。また、通常のデータベースの終了時に結果を出力することもできます。



**Note:** Adabas Vista のライセンスを持たない Adabas のお客様には、AVILOOK サンプラーと SYSAVI のデモバージョンが Adabas リリーステープに収録されています。

## AVILOOK の有効化

AVILOOK は次の ADARUN コマンドを使用して有効にします。

```
ADARUN VISTA=YES
```

## SYSAVI – AVILOOK の選択

▶ **手順 40.1. AVILOOK を選択するには、次の手順に従います。**

- SYSAVI メインメニューからサービス 4 を選択すると、次の AVILOOK メニューオプションが表示されます。

```

+-----+
|          Code   Service          |
|          ---   -|
|             1   File Maintenance |
|             .   Exit              |
|          ---   -|
| Code..: _                |
|                               |
| Database ID..: _____ |
|                               |
| Nucleus ID..: _____  |
|                               |
| System Coordinator Node.: _____ |
+-----+

```

## SYSAVI – AVILOOK の使用

ここでは、AVILOOK の使用方法について説明します。

- AVILOOK ファイルメンテナンス
- AVILOOK ファイル統計
- AVILOOK ファイルの追加

### AVILOOK ファイルメンテナンス

ファイルメンテナンス画面には、指定されたデータベース番号の AVILOOK にすでに定義されたファイルが一覧表示されます。データベース名も表示されます。

▶手順40.2.AVILOOKメニューからファイルメンテナンス画面を表示するには、次の手順に従います。

- 1 サービスオプション1 を選択します。
- 2 AVILOOK を実行するデータベースの Adabas データベース番号を指定します。

データベースは、ADARUN VISTA=YES パラメータを指定して実行する必要があります。

- 3 クラスタデータベースの場合は、ニュークリアス ID を指定します（オプション）。

指定したデータベース番号がクラスタデータベースの場合は、監視するクラスタニュークリアスのニュークリアス ID を指定するオプションがあります。

```

+-----+
| 13:31:07      Select Cluster Members  2006-04-20 |
|                                                    |
| DBID:   231   |
|                                                    |
|  C   Nuc   C   Nuc   C   Nuc   C   Nuc   |
|  _   1   _   2   |
|                                                    |
| Mark to select |
| Command ==>   |
|                PF3 Exit      PF4 Refr      |
+-----+

```

このリストから、適切なニュークリアスを選択できます。

ニュークリアス ID を指定しない場合、または値に 0 を指定する場合は、ローカル Adabas System Coordinator のノード ID を指定する必要があります。クラスタ内で現在アクティブになっているニュークリアスが一覧表示されたウィンドウが表示されます。

SYSAVI を使用するジョブが System Coordinator グループに定義されている場合は、ローカル System Coordinator のノード ID は自動的に設定されます。

```
DBID: 231      (TEST-V7-DB)

C   File      Command Limit      Commands      Started      Status
-   -         -                -             -            -
-   12         0                0             -            Paused
-   2         0                5768         2006-04-20 09:09:20  Active

Mark with (A)ctivate,(P)ause,(R)eset,(S)tatistics,(X)Delete

Command ==>
```

コマンド数を更新するには、PF4 キーを押します。

▶ **手順 40.3. ファイルエントリを選択するには、次の手順に従います。**

- 選択するファイルの隣の [C] 列に、次のいずれかのオプションを入力します。

a	アクティブ化。統計の収集を開始します。
p	停止。統計の収集を停止します。
r	統計をゼロにリセットします。
s	ファイルの現在の統計を表示します。
x	ファイルリストからファイルを削除します。

## AVILOOK ファイル統計

▶手順 40.4. AVILOOK メニューからファイル統計画面を表示するには、次の手順に従います。

- ファイルエントリの隣の [C] 列に、オプション「s」を入力します。

```

DBID: 231      (TEST-V7-DB)
File: 2                               Started: 2006-04-20 09:09:20
                                       Paused :

      CC      Desc      Command Count      CC      Desc      Command Count
      L3      AA          2836           L3      AA          2836
      S1      AB          1324           S1      AB          1324
      L3      BC           24            L3      BC           24
      L9      S1           26            L9      S1           26

Other Commands not listed above: 1558

Command ==>

```

この画面には、ファイルへのアクセスに使用されるコマンド構造に関する統計が表示されま  
す（例えば S1 L3 および L9 コマンド）。統計は、コマンドコード（ [CC] ）および Adabas  
の 2 文字のフィールド名（ [Desc] ）ごとに降順に表示されます。

この例では、Adabas フィールド AA をプライマリシーケンスフィールドとして指定した  
ファイル 2 に対して、L3 コマンドを使用したアクセスが 2836 回あります。このように、主  
なアクセスが単一キーを使用したものであるようなファイルは、Adabas のフィールド AA  
を Adabas Vista パーティションオプションとして使用してパーティションを分割するこ  
とが有効なことがあります。

## AVILOOK ファイルの追加

▶手順 40.5. 新しい AVILOOK ファイルを追加するには、次の手順に従います。

- 1 AVILOOK のファイルメンテナンス画面で PF10 キーを押します。

```
+-----+
| 08:11:07          AVILOOK Add File          2006-04-20 |
|                                     V14100M2 |
|                                     |
|      File _____ Status (A/P) P      (Active/Paused) |
|                                     Command Limit 0      (0=No Limit) |
|                                     |
|                                     PF3 Exit   PF5 Update |
+-----+
```

- 2 ファイル番号を入力します。
- 3 AVILOOKですぐに統計の収集を開始するか（アクティブステータス）、今ファイルを定義して後で有効にするか（停止ステータス）を示します。
- 4 アクティブなファイルが停止ステータスになる前に処理できるコマンドの最大数を事前定義します（オプション）。
- 5 ファイルを追加するには、PF5 キーを押します。



**Note:** 定義されたファイル（およびアクティブ化されたファイル）のみが、現在のデータベースセッション中にサンプリングされます。



# 索引

## A

### Adabas Cluster Services

オンラインでのセッションのモニタ, 218

### Adabas Online System

デモバージョン, 207, 210

エラーメッセージ, 216

オプションの範囲, 210

概要, 214

データベースの指定, 215

プログラム機能 (PF) キーの使用, 215

ヘルプ情報の表示, 216

メインメニューの機能, 213

メニューオプションの選択, 215

呼び出し, 214

ログオン, 213

### Adabas Parallel Services

オンラインでのセッションのモニタ, 218

### ADADBS ユーティリティ

スペースの割り当て, 152

### ADAINV ユーティリティ

スペースの割り当て, 154

### ADALOD ユーティリティ

スペースの割り当て, 155

### ADAM

インバーテッドリストを使用しないレコードへのダイレクトアクセス, 74

### ADAMXY モジュール, 176

### ADAORD ユーティリティ

スペースの割り当て, 158

### ADARAI ユーティリティ

開始, 99

世代の概念, 99

全般的な説明, 97

### ADARUN

パラメータの表示, 219

### ADASAV ユーティリティ

スペースの割り当て, 158

### ADASMxit, 187

## D

### DBA

IS 組織内での位置, 8

Software AG との関係, 52

アプリケーションの選択, 18

オペレーションとの関係, 49

教育の責任, 32

再スタートおよびリカバリ手順のドキュメント化, 29

システム開発へのアドバイス, 18

適切なアプリケーションの選択, 15

手順や標準の保守, 14

データ源の記述, 26

データ収集と整合性確認に対するアドバイス, 19

データディクショナリの使用, 24

データのアクセスおよび操作手順のドキュメント化, 27

データの信頼性の決定, 18

データベース設計の援助, 14

データベース手順と標準の設定, 12

データベースのドキュメント化, 22

データベースの内容の定義, 19

データベースパフォーマンスの計測, 29

データベース標準のドキュメント化, 22

データベースプランニング, 42

データベースを使用するアプリケーションのドキュメント化, 25

バックアップ手順のドキュメント化, 28

必要な能力, 8

マネージメントサポート, 9

役割のまとめ, 15

ユーザー ID とパスワードの情報の維持, 27

ユーザー教育, 14

ユーザーとの連絡, 38

## E

### ETID

ADALOD によるオーナー ID への変換, 108

## I

### ISN

ディスクリプタとして使用, 74

ユーザー割り当て, 74

## M

### MAXISN

ADALOD パラメータ, 150

## P

### PIN モジュール

ADAMXY, 176

インストール, 177

オンラインでの追加/削除, 238

リスト, 177

### PIN ルーチン

オンラインでのアクティブ化/非アクティブ化, 238  
 処理, 176  
 定義済み, 175  
 表示, 238  
 ユーザー出口, 187  
 PINAUTOR, 179  
   アクティブ化, 178  
   インストール, 178  
   ユーザー出口, 189  
 PINAVI, 180  
 PINCOR, 180  
 PINOPRSP, 181  
   アクティブ化, 178  
   インストール, 178  
 PINRSP, 181  
   ユーザー出口, 190  
 PINSAP, 183  
 PINUES, 183  
 Predict  
   Adabas データディクショナリ, 24

## R

RABN  
 説明, 147  
 必要なデバイススペース, 147

## あ

アソシエータ  
 Adabas によって確保されるブロック, 126  
 公式を使用したアップパーインデックス (UI) 用スペース計算, 131  
 公式を使用したアドレスコンバータ用スペース計算, 132  
 公式を使用したノーマルインデックス (NI) 用スペース計算, 128  
 自動スペース計算, 126  
 デバイスの割り当て, 141  
 アップパーインデックス  
   公式を使用したスペース計算, 131  
 アドレスコンバータ  
   公式を使用したスペース計算, 132  
   ニュークリアスによるスペース割り当て, 150  
 暗号化  
   データ, 86

## い

一時  
   デバイスの割り当て, 141  
 インデックス  
   ニュークリアスによるスペース割り当て, 151

## え

エクステント  
   物理エクステントの説明, 146  
   論理エクステントの説明, 149  
 エラー  
   拡張エラーリカバリメニュー, 237  
   システム/トランザクション障害, 93  
   トランザクションリカバリ, 92  
   リカバリの計画, 91  
 エラー処理機能, 173

## お

大きなファイル  
   データベース割り当て, 143

## か

拡張ファイル  
   ADALOD を使用した定義, 113  
   コンポーネントファイルの削除, 115  
   コンポーネントファイルの挿入, 115  
   削除, 115  
   使用時の制限, 117  
   説明, 111  
   一般的な説明, 66  
   調査, 115  
   定義規則, 114  
   ニュークリアス, 116  
   ニュークリアスの変更, 116  
   ユーティリティ, 117

## こ

コマンド  
   CL (クローズ), 92  
   ET (エンドトランザクション), 92  
   チェックポイント, 93  
 コマンドログ  
   デバイスの割り当て, 141  
 コンディションディスクリプションブロック (CDB), 188

## さ

再スタート  
   プランニングと設計, 89  
 サブディスクリプタ  
   説明, 71

## し

システム  
   障害, 93  
   ステータスの表示, 222  
 自動バックアウト  
   ルーチンの機能, 93

## す

スペース  
   Adabas コンポーネントへの割り当て, 141  
   ADADBS を使用した割り当て, 152  
   ADAFRM ユーティリティを使用したフォーマット, 143  
   ADAINV を使用した割り当て, 154  
   ADALOD を使用した割り当て, 155  
   ADAORD を使用した割り当て, 158  
   ADASAV を使用した割り当て, 158  
   一般的な見積り手順, 127  
   効率的な使用, 77  
   最大物理エクステントに達した場合, 160  
   最大論理エクステントに達した場合, 161  
   データベース管理, 145  
   データベース全体の見積り, 127  
   ニュークリアスによる割り当て, 150

- ファイルごとの見積り, 127
- 物理エクステントが一杯になった場合, 160
- 見積りの公式, 127
- 問題点とその対応策, 159
- 割り当て/割り当て解除手順, 150
- スレッド
  - 使用統計の表示, 222
- スーパーディスクリプタ
  - 説明, 71

## せ

- セキュリティ
  - ADASAF の使用, 86
  - AOS Security の使用, 87
  - Natural Security の使用, 87
  - 計画と機能, 83
  - データの暗号化, 86
  - バイバリュウ, 86
  - パスワードと値レベル, 84
- セッション
  - オンラインでの終了, 241
  - オンラインでのモニタ, 217, 218
  - ステータスの表示, 222

## そ

- ソート
  - デバイスの割り当て, 141
- ソートデータセット
  - 公式を使用したスペース計算, 138

## ち

- チェックポイント
  - 現在のチェックポイントの表示, 225
  - タイプ, 226

## て

- ディスクリプタ
  - ISN を使用, 74
  - 効率的な使用, 70
  - マルチクライアントオーナー ID の接頭辞, 106
- デバイスタイプ
  - Adabas コンポーネントの割り当て先, 141
- データアクセス
  - 手法, 69
- データ圧縮
  - 空値オプション, 79
  - 空値省略のオプション, 78
  - 固定ストレージ (FI) オプション, 78, 79
  - デフォルトのオプション, 78, 79
  - パディングファクタ, 81
- データ記録機能の向上
  - 使用に関する推奨事項, 143
- データストレージ
  - 公式を使用したスペース計算, 133
  - 自動スペース計算, 126
  - デバイスの割り当て, 141
  - ニュークリアスによるスペース割り当て, 151
- データの重複
  - 物理, 67
  - 論理的, 67

- データプロテクションエリア
  - 公式を使用したスペース計算, 135
- データベース
  - オンラインメンテナンスメニュー, 233
  - オンラインレポート, 244
  - サイズの見積り, 124
  - ステータスレポートを使用したスペース使用状況の監視, 159
  - スペース管理, 145
  - 設計, 55
  - 全体的なレイアウトの表示, 248
  - 定義, 123
  - パラメータの定義, 144
  - 標準, 22
  - ファイルとレコードの設計, 61
  - リカバリと再スタート設計, 89

## と

- トランザクション
  - 障害, 93
  - リカバリ, 92
  - リカバリの制限, 93

## に

- ニュークリアス
  - スペース割り当て, 150
  - 必要なコンポーネント, 124

## の

- ノーマルインデックス
  - 公式を使用したスペース計算, 128

## は

- 排他ファイル制御, 94
- ハイパーディスクリプタ
  - 説明, 72
- パスワード
  - セキュリティ, 84
- パフォーマンス
  - 満足のいくパフォーマンスを達成する方法, 58

## ひ

- ピリオディックグループ
  - 説明, 62

## ふ

- ファイル
  - 1つの論理ファイル内への複数の物理ファイルのリンク, 66
  - 大きなファイルのデータベース割り当て, 143
  - オンラインメンテナンスメニュー, 229
  - 排他制御, 94
  - ファイルレイアウトの表示, 246
  - リストの表示, 244
  - レコードの設計, 61
  - ロックファイルの表示, 239, 240
- ファイルカップリング
  - 物理, 72

- 論理的, 73
- 論理ファイルのマルチクライアントサポート, 104
- フィールド
  - 空値省略 (NU) オプション, 79
  - 組み合わせの長所と短所, 68
  - グループの使用, 68
  - 固定ストレージ (FI) オプション, 68, 79
  - 数値フィールドの使用, 68
- フィールド定義テーブル (FDT)
  - 新しいファイルに定義
    - AOS デモバージョンの使用, 230
- フォネティックディスクリプタ
  - 説明, 72
- フリースペーステーブル (FST)
  - 説明, 150
- プロテクションログ
  - デバイスの割り当て, 141

## ほ

- ホールドキュー
  - 表示, 220

## ま

- マルチクライアントファイル
  - ADACMP ユーティリティのサポート, 110
  - ADALOD ユーティリティのサポート, 108
  - ADAULD ユーティリティのサポート, 109
  - アプリケーションプログラムに透過的, 104
  - オーナー ID とユーザー ID (ETID) の関係, 102
  - オーナーの概念, 102
  - シングルクライアントから変換, 102
  - スーパーユーザーの概念, 103
  - 説明, 102
  - ソフトカップリングのサポート, 104
  - データとインデックスの構造, 104
  - パフォーマンスの考慮事項, 106
  - ユーザープロファイルに保存されたオーナー ID, 107
  - レスポンスコード, 107
- マルチプルバリューフィールド
  - 説明, 62
- マルチプロセッシング
  - セッションの監視, 218

## め

- メンテナンス
  - レベルの表示, 223

## ゆ

- ユニバーサルエンコーディングサポート (UES) , 191
- ユーザー
  - オンラインでの停止, 240
  - 処理の停止
    - 基本サービスの使用, 240
- ユーザー出口
  - ADASMxit, 187
  - PIN ルーチン, 187
  - エラー処理, 174
- ユーザーデータ
  - 再スタート, 95
  - ダイレクトコールコマンドによる読み込み, 93

## り

- リカバリ
  - プランニングと設計, 89
- リカバリログ
  - 全般的な説明, 98
  - 定義, 98
  - デバイスの割り当て, 141
- リソース
  - 使用統計の表示, 221

## れ

- レコード
  - 設計, 67
  - ダイレクトアクセス, 74
  - ファイルの設計, 61
  - 複数のタイプを 1 ファイルに統合, 65

## わ

- ワーク
  - 公式を使用したパート 1 のスペース計算, 135
  - 公式を使用したパート 2 のスペース計算, 136
  - データセットの割り当て, 134
  - パート 3 のスペース計算, 137
- ワークエリア
  - デバイスの割り当て, 141
  - パート 4 のスペース計算, 137