

CICS Transaction Server for z/OS



Db2 ガイド

バージョン 5 リリース 5

CICS Transaction Server for z/OS



Db2 ガイド

バージョン 5 リリース 5

注記

本書および本書で紹介する製品をご使用になる前に、 205 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM CICS Transaction Server for z/OS バージョン 5 リリース 5 (製品番号 5655-Y04) および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： CICS Transaction Server for z/OS
Db2 Guide
Version 5 Release 5

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

© Copyright IBM Corporation 1997, 2018.

目次

この PDF について	vii
-----------------------	-----

第 1 章 CICS Db2 インターフェースの概要 1

概要: CICS を Db2 に接続する方法	1
Db2 アドレス・スペース	2
概要: スレッドの働き	3
オープン・トランザクション環境のスレッド TCB	5
概要: CICS アプリケーション・プログラムが Db2 にアクセスできるようにする	7
Db2 にアクセスする CICS アプリケーション・プログラムの準備	7
バインド・プロセス	9
計画、パッケージ、および動的計画出口	9

第 2 章 CICS Db2 接続の定義 13

概要: CICS Db2 接続の定義方法	13
Db2 グループ接続機能の使用	16
MAXOPENTCBS システム初期設定パラメーターおよび TCBLIMIT	18
SQL 処理時に何が起きるか	19
スレッドの作成	19
SQL 処理	20
コミット処理	20
スレッド解放	20
スレッド終了	21
スレッドの作成、使用、および終了	21
保護エントリ・スレッド	23
クリティカル・トランザクションの無保護エントリ・スレッド	24
バックグラウンド・トランザクションの無保護エントリ・スレッド	25
プール・スレッド	26
最適なパフォーマンスのためのスレッド・タイプの選択	27
最適なパフォーマンスのための BIND オプションの選択	29
DB2CONN、DB2ENTRY、および BIND オプションの調整	29

第 3 章 CICS Db2 での管理 33

CICS Db2 接続機能の開始	33
CICS Db2 接続機能の停止	33
CICS 終了時の自動切断	33
手動の切断	34
未確定作業単位 (UOW) の解決	34
グループ接続を使用した未確定 UOW の解決	35
Db2 restart-light を使用した未確定作業単位の解決	36
未確定 UOW の再同期情報のリカバリー	37

CICS Db2 接続機能の管理	37
Db2 コマンドの入力	38
Db2 アカウンティング、統計、および調整のための SMF の開始	39
Db2 アカウンティング、統計、および調整のための GTF の開始	40
CICS Db2 用の CICS 提供トランザクション	40
DSNC トランザクションの使用による Db2 へのコマンドの発行	42
DSNC DISCONNECT	44
DSNC DISPLAY	45
DSNC MODIFY	50
DSNC STOP	52
DSNC STRT	54

第 4 章 Db2 のセキュリティ 59

CICS での Db2 関連リソースへのアクセスの制御	60
DB2CONN、DB2TRAN、および DB2ENTRY リソース定義に対するユーザー・アクセスの制御	61
リソース・セキュリティを使用して、DB2ENTRY リソース定義および DB2TRAN リソース定義へのアクセスを制御する	62
コマンド・セキュリティを使用して、DB2CONN、DB2ENTRY、および DB2TRAN の各リソース定義に対する SPI コマンドの発行を制御する	64
代理セキュリティおよび AUTHTYPE セキュリティを使用して、CICS が DB2 に提供する許可 ID へのアクセスを制御する	66
Db2 関連の CICS トランザクションへのユーザー・アクセスの制御	69
CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する	70
CICS 領域用の許可 ID を Db2 に提供する	71
CICS 領域用の 1 次許可 ID を提供する	72
CICS 領域用の 2 次許可 ID を提供する	73
CICS トランザクション用の許可 ID を Db2 に提供する	74
CICS トランザクション用の 1 次許可 ID を提供する	75
CICS トランザクション用に 2 次許可 ID を提供する	78
ユーザーに対する Db2 内のリソースへのアクセス許可	80
Db2 コマンドへのユーザーのアクセスの制御	81
計画へのユーザー・アクセスの制御	82
Db2 マルチレベル・セキュリティおよび行レベル・セキュリティ	84

第 5 章 CICS Db2 に関するアプリケーション設計と開発の考慮事項 87

CICS アプリケーションと Db2 計画およびパッケージの間の関係の設計	88
サンプル・アプリケーション	89
Db2 パッケージの使用	90
すべてのトランザクションのために 1 つの大規模な計画を使用する	94
多数の小規模な計画を使用する	95
トランザクションのグループ化に基づく計画の使用	96
動的計画出口	97
既に開発されているアプリケーション用に計画を作成する必要がある場合	100
トランザクション内で計画を切り替える必要がある場合	101
動的計画切り替え	102
計画を切り替えるためのトランザクション ID の切り替え	103
CICS Db2 環境でのロック戦略の開発	107
CICS Db2 アプリケーションに関する SQL、スレッド・セーフ、その他のプログラミング考慮事項	108
スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにする	108
SQL 言語	112
修飾および非修飾 SQL の使用	113
ビュー	114
索引列の更新	114
固有索引の依存関係	115
コミット処理	115
トランザクションの直列化	115
ページの競合	116
CICS と CURSOR WITH HOLD オプション	118
EXEC CICS RETURN IMMEDIATE コマンド	119
AEY9 異常終了の回避	119

第 6 章 Java プログラムから Db2 データにアクセスするための JDBC および SQLJ の使用 123

IBM Data Server Driver for JDBC and SQLJ	123
Db2 をサポートするための JVM サーバーの構成	123
JVM サーバーでの Db2 スキーマの設定	124
JDBC および SQLJ API に対するプログラミング	125
JVM サーバーへのシリアライズされた SQLJ プロファイルのデプロイ	125
データベースへの接続の取得	126
データベースへの DriverManager 接続の取得	127
データベースへの DataSource 接続の取得	128
JDBC 接続と SQLJ 接続に関する考慮事項	129
データベースへの接続を閉じる	130
作業単位のコミット	130
自動コミット	131
明示的 URL およびデフォルト URL での DriverManager の同期点の問題	131

JDBC 要求または SQLJ 要求の間に CICS が異常終了する 132

第 7 章 CICS Db2 プログラムの実行および実動の準備 133

CICS Db2 テスト環境	133
CICS Db2 プログラムの準備	134
CICS SQLCA フォーマット設定ルーチン	137
プログラム変更後に何をバインドするか	138
プログラムのバインド・オプションと考慮事項	139
RETAIN	140
分離レベル	140
プラン検証時	140
ACQUIRE および RELEASE	141
CICS Db2 プログラムのテストおよびデバッグ	141
実動への移行: CICS Db2 アプリケーションのチェックリスト	141
Db2 にアクセスする CICS アプリケーションのチューニング	144

第 8 章 Db2 のモニター 147

CICS 提供のアカウントティングおよびモニター情報	147
Db2 提供のアカウントティングおよびモニター情報	148
CICS Db2 環境のモニター: 概要	149
CICS Db2 接続機能のモニター	151
CICS Db2 接続機能コマンドを使用した CICS Db2 接続機能のモニター	151
Db2 コマンドを使用した CICS Db2 接続機能のモニター	151
CICS Db2 統計を使用した CICS Db2 接続機能のモニター	152
Db2 リソースにアクセスする CICS トランザクションのモニター	155
CICS で使用されている場合の Db2 のモニター	156
Db2 統計機能を使用した Db2 のモニター	156
Db2 アカウントティング機能を使用した Db2 のモニター	159
Db2 パフォーマンス機能を使用した Db2 のモニター	159
CICS Db2 環境での CICS システムのモニター	160
CICS Db2 環境でのアカウントティング: 概要	160
Db2 アカウントティング機能によって提供されるアカウントティング情報	161
Db2 アカウントティング・レコードのデータ・タイプ	162
Db2 アカウントティング・レポート	164
Db2 アカウントティング・レコードと CICS パフォーマンス・クラス・レコードとの関連付け	165
Db2 アカウントティング・レコードと CICS パフォーマンス・レコードを突き合わせる際の問題	166
Db2 アカウントティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係の制御	167
Db2 アカウントティング・レコードのデータを使用した、対応する CICS パフォーマンス・クラス・レコードの識別	169

Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせてユーザーにリソースを課金する場合の方針 . . .	169	CSUB トレース	194
CICS Db2 環境でのプロセッサ使用量を考慮に入る	173	CICS Db2 のダンプ	196
アカウンティング CLASS 1 プロセッサ時間	178	Db2 スレッド ID	197
アカウンティング CLASS 2 プロセッサ時間	179	CICS Db2 のトランザクション異常終了コード	197
Db2 以前の CICS および DB2 バージョン 5 プロセッサ時間の計算	180	CICS Db2 の実行診断機能 (EDF)	198
Db2 の CICS および Db2 プロセッサ時間の計算	181	CICS Db2 環境でのデッドロックの処理	199
第 9 章 Db2 のトラブルシューティング 183		2 つのタイプのデッドロック	201
スレッド TCB (タスク制御ブロック)	183	デッドロックの検出	201
CICS Db2 の待機タイプ	183	関与するリソースの検出	202
CICS Db2 のメッセージ	187	関与する SQL ステートメントの検出	202
CICS Db2 のトレース	187	使用されたアクセス・パスの検出	203
		デッドロックが発生した理由の判別	203
		デッドロックの解決	203
		特記事項	205
		索引	211

この PDF について

この PDF には、CICS Db2[®] 接続機能の評価、インストール、および使用について、また CICS Db2 環境の定義および保守についての概要と指針を記載しています。

本書で使用される用語と表記について詳しくは、IBM Knowledge Center の CICS 資料で使用されている表記規則および用語 を参照してください。

この PDF の作成日

この PDF は、2018 年 12 月 14 日に作成されました。

第 1 章 CICS Db2 インターフェースの概要

このセクションでは、Db2 への CICS® インターフェースについて概説します。

概要: CICS を Db2 に接続する方法

CICS とともに CICS Db2 接続機能が提供されています。CICS Db2 接続機能により、CICS 環境で実行中の CICS アプリケーションが Db2 データにアクセスできます。

CICS は、トランザクションまたはシステムの障害発生時に Db2 と CICS のデータのリカバリーを調整します。

CICS Db2 接続機能は、CICS と Db2 との全体の接続を作成します。CICS アプリケーションは、この接続を使用して Db2 にコマンドおよび要求を発行します。CICS と Db2 の間の接続は、いつでも作成または終了できます。また、CICS と Db2 はそれぞれ個別に開始および停止できます。CICS の接続先となる個別の Db2 サブシステムを指定できます。あるいは、(DB2® バージョン 7 以降を使用している場合は) グループ接続機能を使用して、Db2 が接続対象として Db2 サブシステムのデータ共有グループの任意のアクティブ・メンバーを選択するように設定できます。また、CICS が Db2 に自動的に接続および再接続するように設定するオプションもあります。Db2 システムは、複数の CICS システムで共用できますが、各 CICS システムは一度に 1 つの Db2 サブシステムにしか接続できません。

CICS Db2 接続は、3 つの異なる CICS リソース定義を使用して定義できます。これらは、DB2CONN (Db2 接続定義)、DB2ENTRY (Db2 エントリー定義)、および DB2TRAN (Db2 トランザクション定義) です。CICS Db2 接続を開始する前に、DB2CONN リソース定義をインストールする必要があります。(このリソース定義を DB2CONN システム初期設定パラメーターと混同しないでください。このパラメーターは、CICS が初期化中に Db2 接続を自動的に開始するかどうかを指定するものです。) また、DB2ENTRY 定義や、必要に応じて DB2TRAN 定義を作成して、重要なトランザクションが優先されるように設定することもできます。概要: CICS Db2 接続の定義方法には、これらのリソース定義についての詳細情報が記載されています。

接続コマンドは、CICS Db2 接続機能の状況を表示および制御し、CICS 提供トランザクション DSNB を使用して発行されます。接続コマンドは以下のとおりです。

- STRT - Db2 への接続を開始します
- STOP - Db2 への接続を停止します
- DISP - スレッドの状況を表示したり、統計を表示したりします
- MODI - Db2 への接続の特性を変更します
- DISC - スレッドを切断します

CICS と Db2 の間の接続は、マルチスレッド接続です。CICS と Db2 との全体の接続内で、Db2 にアクセスするアクティブな CICS トランザクションごとに、スレ

ッド (Db2 への個別接続) が存在します。スレッドにより、各 CICS トランザクションは、コマンド・プロセッサやアプリケーション計画 (アプリケーション・プログラムの SQL 要求の内容や、それらの要求に応じたサービスを提供するために最も効率的な方法を Db2 に通知する情報) などの Db2 リソースにアクセスします。スレッドの仕組みについて詳しくは、概要: スレッドの動作方法を参照してください。

CICS 環境で稼働するアプリケーション・プログラムが、そのプログラムで最初の SQL 要求を発行すると、CICS および Db2 はその要求を以下のように処理します。

- 言語インターフェース (アプリケーション・プログラムとリンク・エディットされる DSNCLI スタブ) が、CICS リソース・マネージャー・インターフェース (RMI) を呼び出します。
- RMI は、要求を処理し、CICS Db2 接続機能のタスク関連ユーザー出口 (TRUE) に制御を渡します。このモジュールは、タスクごとに Db2 を呼び出します。
- CICS Db2 接続機能は、トランザクションのスレッドをスケジュールします。このステージで、Db2 は許可を検査し、適切なアプリケーション計画を見つけます。
- Db2 が制御を獲得し、CICS Db2 接続機能は、Db2 サービスが要求に対応している間待機します。
- SQL 要求が完了すると、Db2 は要求されたデータを CICS Db2 接続機能に戻します。
- CICS が再度制御を獲得し、CICS Db2 接続機能は、CICS アプリケーション・プログラムにデータを渡して、制御を返します。

Db2 アドレス・スペース

Db2 には、異なるいくつかのアドレス・スペースが必要です。

図 1 には、それらのアドレス・スペースが示されています。

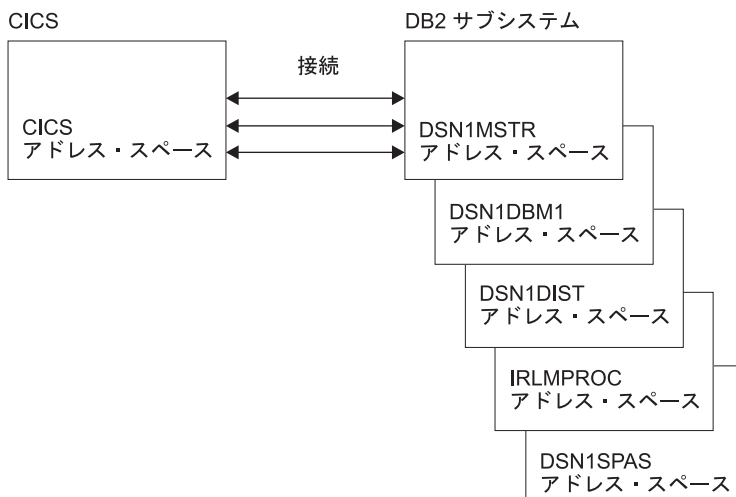


図 1. Db2 アドレス・スペース

以下のように、各種のタスクが異なるアドレス・スペースで実行されます。

DSN1MSTR

さまざまなシステム関連機能を実行するシステム・サービス用。

DSN1DBM1

ユーザーが作成したデータベース内の大部分の構造を操作するデータ・サービス用。

DSN1DIST

リモート要求にサポートを提供する分散データ機能用。

IRLMPROC

Db2 ロックを制御する内部リソース・ロック・マネージャー (IRLM) 用。

DSN1SPAS

ユーザー作成 SQL 用の分離実行環境を提供するストアード・プロシージャ用。

概要: スレッドの働き

個々の CICS トランザクションには、Db2 にアクセスするための独自のスレッドが必要です。

スレッドは、トランザクションがスレッドを必要としたとき、つまりアプリケーションが最初の SQL 要求またはコマンド要求を発行した時点で作成されます。トランザクションは、スレッドを使用して、Db2 が管理するリソースにアクセスします。トランザクションで、使用しなくてはならないすべてのリソースにアクセスしたためにスレッドが不要になると、スレッドは解放されます (通常は、同期点完了後)。スレッドの作成にはプロセッサ・リソースが使用されるため、スレッドが解放されると、CICS Db2 接続機能は、別のトランザクションがスレッドを必要としているかどうかをチェックします。別のトランザクションがスレッドを待機している場合、CICS Db2 接続機能は、そのトランザクションが Db2 にアクセスするために既存のスレッドを再使用します。スレッドを必要とするトランザクションがなくなったら、ユーザーがスレッドを一定期間保護 (保持) することを要求していない限り、スレッドは終了します。保護されたスレッドは、その一定期間内に別のトランザクションがスレッドを要求した場合に再使用されます。要求されなかった場合、保護する時間が経過したら、スレッドは終了します。

スレッドにはさまざまなタイプがあり、スレッドのタイプごとに一度にアクティブにできるスレッド数の制限を設定できます。これにより、CICS Db2 の全体の接続により処理に過剰な負荷がかからないようにします。DSNC トランザクションを使用して発行された Db2 コマンドには、特殊なタイプのスレッドが使用されます。また、速い応答時間を必要とするトランザクションなど、特殊な要件を持つ CICS トランザクションに対して特殊なスレッドを定義することもできます。必要とするタイプのスレッドを使用できない場合にトランザクションが取るべきアクションを定義できます。適切なタイプのスレッドが使用可能になるまで待機するか、プール・スレッドと呼ばれる汎用スレッドを使用するか、異常終了するかを定義できます。

CICS Db2 接続機能によって提供されているスレッドのタイプは、以下のとおりです。

コマンド・スレッド

コマンド・スレッドは、DSNC トランザクションを使用して Db2 にコマンドを発行するために CICS Db2 接続機能により予約されています。これらのコマンドは Db2 に渡されないため、CICS Db2 接続機能自体に作用するコマンドには使用されません。コマンド・スレッドを使用できない場合、コマンドは自動的にプールにオーバーフローして、プール・スレッドを使用します。コマンド・スレッドは、DB2CONN 定義のコマンド・スレッド・セクションで定義されます。

エントリー・スレッド

エントリー・スレッドは、速い応答時間を必要とするトランザクションや特殊なアカウント要求を持つトランザクションなど、特殊要件を持つトランザクション用に特別に定義されたスレッドです。特殊な CICS トランザクションにエントリー・スレッドを提供するよう CICS Db2 接続機能に指示することができます。さまざまなトランザクションで必要とされるさまざまなタイプのエントリー・スレッドを定義し、それらの各タイプのエントリー・スレッド数の制限を設定できます。トランザクションがエントリー・スレッドの使用を許可されているにもかかわらず、適切なエントリー・スレッドが使用可能でない場合、トランザクションは、エントリー・スレッド定義でのユーザー選択に従って、プールにオーバーフローしてプール・スレッドを使用するか、適切なエントリー・スレッドを待機するか、異常終了します。

一定数の各タイプのエントリー・スレッドを保護できます。解放されたエントリー・スレッドが保護スレッドの場合、このスレッドは即時に終了しません。一定期間保持され、この期間中に別の CICS トランザクションが同じタイプのエントリー・スレッドを要求した場合に、再利用されます。これにより、各トランザクションでのスレッドの作成および終了に関連するオーバーヘッドがなくなります。保護されていないエントリー・スレッドは、CICS トランザクションがこのスレッドの解放後にこのスレッドを使用するよう待機していないかぎり、即時に終了します。

エントリー・スレッドは、DB2ENTRY 定義を使用して定義できます。

プール・スレッド

プール・スレッドはエントリー・スレッドまたは Db2 コマンド・スレッドを使用しないすべてのトランザクションおよびコマンドで使用されます。プール・スレッドはトランザクションが少量の場合、およびエントリー・スレッドまたは Db2 コマンド・スレッドを取得できないオーバーフロー・トランザクションの場合に使用します。プール・スレッドは、このスレッドを使用するように待機している CICS トランザクションが存在しない場合は、即時に終了します。プール・スレッドは、DB2CONN 定義のプール・スレッド・セクションで定義されます。

さまざまなタイプのスレッドの作成方法、使用方法、および終了方法については、How threads are created, used, and terminatedを参照してください。

各スレッドは、CICS に属しているスレッド・タスク制御ブロック (スレッド TCB) の下で実行されます。CICS と Db2 は、接続制御ブロックをスレッド TCB にリンクします。CICS と DB2 は、これらの接続制御ブロックを使用して Db2 へのスレッドを管理し、スレッドに関する情報をお互いに伝達します。Db2 接続制御プロ

ックは、Db2 内のスレッドを制御します。CSUB という CICS 接続制御ブロックは、Db2 接続制御ブロックへのポインターとして機能します。CSUB には、スレッドが必要になったときに CICS が Db2 接続制御ブロックを呼び出すために必要な情報が含まれています。Db2 では、これらの接続制御ブロックを「エージェント構造」と呼びます。

CICS が Db2 サブシステムに接続している間、CICS Db2 タスク関連ユーザー出口 (タスクごとに Db2 を呼び出すモジュール) がオープン API として自動的に使用可能になり、オープン・トランザクション環境 (OTE) を使用できるようになります。

CICS Db2 接続機能がスレッドを実行するために使用する TCB には、オープン TCB とサブタスク TCB の 2 つのタイプがありますが、この資料では、これら両方のタイプの TCB を「スレッド TCB」と呼んでいます。多くの場合、これらの 2 つのタイプのスレッド TCB の特性の相違により、CICS Db2 接続の操作に違いが発生することはありません。スレッド TCB のタイプの相違により CICS Db2 接続の動作が変わるような状況においては、2 つのタイプが区別されます。スレッド TCB についての詳しい技術情報については、スレッド TCB (タスク制御ブロック) を参照してください。

オープン・トランザクション環境のスレッド TCB

CICS が DB2 バージョン 7 以降、または JDBC 2.0 以降に接続される場合、オープン・トランザクション環境が使用されます。この環境では、CICS Db2 接続機能はスレッド TCB として、特別に作成されたサブタスク TCB ではなく、オープン TCB (L8 モード) を使用します。

オープン TCB は、Db2 リソースへのアクセス以外の、その他のタスクを実行します。オープン・トランザクション環境では、CICS Db2 タスク関連ユーザー出口は、CICS のメイン TCB ではなく、オープン TCB で実行されます。Db2 要求を行ったアプリケーション・プログラムがスレッド・セーフである場合は、オープン TCB でも実行できます。(オープン・トランザクション環境でのアプリケーション・プログラムについて詳しくは、スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにするを参照してください)。

オープン TCB は、CSUB および Db2 接続制御ブロックに永続的に関連付けられるわけではありません。CICS Db2 接続機能は、使用可能な任意の CSUB および Db2 接続制御ブロック、つまり、使用可能な任意のスレッドにそれらを関連付けることができます。オープン TCB で Db2 への接続が不要になると、スレッド、CSUB、および Db2 接続制御ブロックから TCB は関連付け解除 (解放) されます。その後、Db2 でタスクを実行するために、別のオープン TCB がそのスレッド、CSUB、および Db2 接続制御ブロックを使用できます。

再利用される前にスレッドが終了されると、CSUB および Db2 接続制御ブロックはシステムで使用可能なままになります。既存のスレッドを使用できない場合に、オープン TCB が Db2 に接続する必要がある場合、CICS Db2 接続機能は未使用の CSUB および Db2 接続制御ブロックとオープン TCB を関連付け、それらを再利用して、Db2 に対して新しいスレッドを実行できます。

図 2 には、オープン・トランザクション環境でスレッド TCB が動作する仕組みがまとめられています。

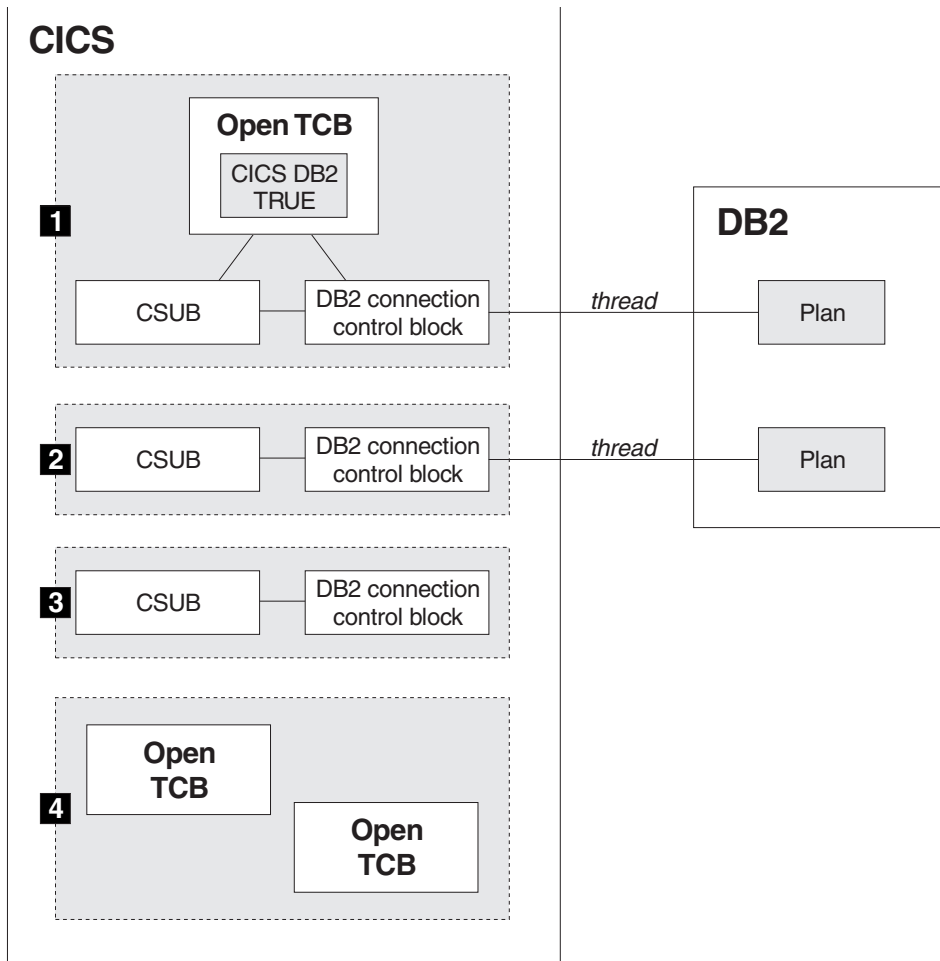


図 2. オープン・トランザクション環境のスレッド TCB

図 2 のシチュエーション 1 は、オープン・トランザクション環境で Db2 にアクセスするためにスレッドを使用する CICS を示しています。CICS Db2 タスク関連ユーザー出口がリソース・マネージャー・インターフェース (RMI) によって呼び出され、オープン TCB で動作しています。CICS Db2 接続機能によって CSUB および Db2 接続制御ブロックがオープン TCB に関連付けられています。Db2 接続制御ブロックには、Db2 へのスレッドがあります。スレッドに関連付けられたプランが Db2 に保持されています。

シチュエーション 2 は、現在使用されていないが、保護されているスレッドを示しています。CSUB および Db2 接続制御ブロックは相互にまだリンクされており、スレッドを 1 つ持ちますが、それらに接続されているオープン TCB はありません。このスレッドは再使用できます。

シチュエーション 3 は、スレッドが終了した後に残るアセンブリーを示しています。この CSUB および Db2 接続制御ブロックは再使用できます。それらには、新しいスレッドが必要です。

シチュエーション 4 は、再使用可能なオープン TCB を示しています。CICS Db2 接続機能はこれらのオープン TCB を使用して、CSUB および Db2 接続制御ブロック・アセンブリーをそれらに関連付け、Db2 に対してスレッドを実行できます。

概要: CICS アプリケーション・プログラムが Db2 にアクセスできるようにする

Db2 にアクセスする CICS アプリケーション・プログラムは、通常の CICS アプリケーション・プログラムと同じように作成しなければなりません。また、Db2 バインド・プロセスを経てアプリケーション・プランを生成する必要があります。

このタスクについて

Db2 バインド・プロセスはアプリケーション・プラン (しばしば単に「プラン」と呼ばれる) を生成します。各プランには、そのプランを使用するアプリケーション・プログラム内のすべての SQL ステートメントが、バインドされた形式で含まれています。それらのアプリケーション・プログラムは実行時にプランを使用することで、Db2 データにアクセスできます。

プランは Db2 で保持され、Db2 への各スレッドはプランに関係します。各タイプのスレッドが使用する計画は、DB2CONN 定義 (プール・スレッドの場合) またはスレッドの DB2ENTRY 定義 (エン트리・スレッドの場合) で指定されます。特定のタイプのスレッドのプランには、そのタイプのスレッドを使用して Db2 にアクセスするすべてのアプリケーション・プログラム内の SQL ステートメントが、バインドされた形式で含まれていなければなりません。プランを明示的に指定することも、動的プラン出口を指定することもできます。動的プラン出口とは、そのタイプのスレッドを要求したトランザクションに使用するプランを決定するルーチンのことです。

Db2 にアクセスする CICS アプリケーション・プログラムの準備

DB にアクセスする CICS アプリケーション・プログラムに対してさまざまな処理を行うことで、DBRM を構築し、アプリケーション・ロード・モジュールを作成した後で、バインド・プロセスに入ることができます。

このタスクについて

8 ページの図 3 は、Db2 にアクセスする CICS アプリケーション・プログラムを準備するためのステップを示しています。

最初のステップは、プログラムを Db2 プリコンパイラで処理することです。Db2 プリコンパイラは、プログラムの各 SQL ステートメントについての情報を含むデータベース要求モデル (DBRM) を作成します。

2 番目、3 番目、および 4 番目のステップは、(Db2 にアクセスするかどうかに関わらず) すべての CICS アプリケーション・プログラムを準備するための通常のプロセスです。2 番目のステップは、プログラムを CICS コマンド言語変換プログラムで処理することです。3 番目のステップは、プログラムをコンパイルまたはアセンブルすることです。4 番目のステップは、(CICS Db2 言語インターフェース・モジュール DSNCLI を含む) 必要なインターフェースを使ってプログラムをリン

ク・エディットすることです。ステップ 2、3、4 の結果として、プログラムの実行を可能にするアプリケーション・ロード・モジュールが生成されます。これらのステップの詳細については、CICS Db2 プログラムの準備を参照してください。

ステップ 1 で作成された DBRM 内の情報をプログラムで使えるようにするには、追加ステップが必要です。この 5 番目のステップは、バインド・プロセスです。バインド・プロセスでは Db2 が必要です。また、プログラムによる Db2 データ・アクセスを可能にするアプリケーション・プランを生成するために DBRM を使用します。バインド・プロセスの説明については、9 ページの『バインド・プロセス』を参照してください。

COBOL および PL/I 用のいずれかの Language Environment[®] 準拠コンパイラを使用している場合、上記のいくつかのステップを 1 つのタスクに結合することができます。これらのコンパイラには CICS コマンド言語変換プログラム、および(ご使用の Db2 のバージョンによっては) SQL ステートメント・コプロセッサが組み込まれているためです。詳しくは、CICS Db2 プログラムの準備を参照してください。

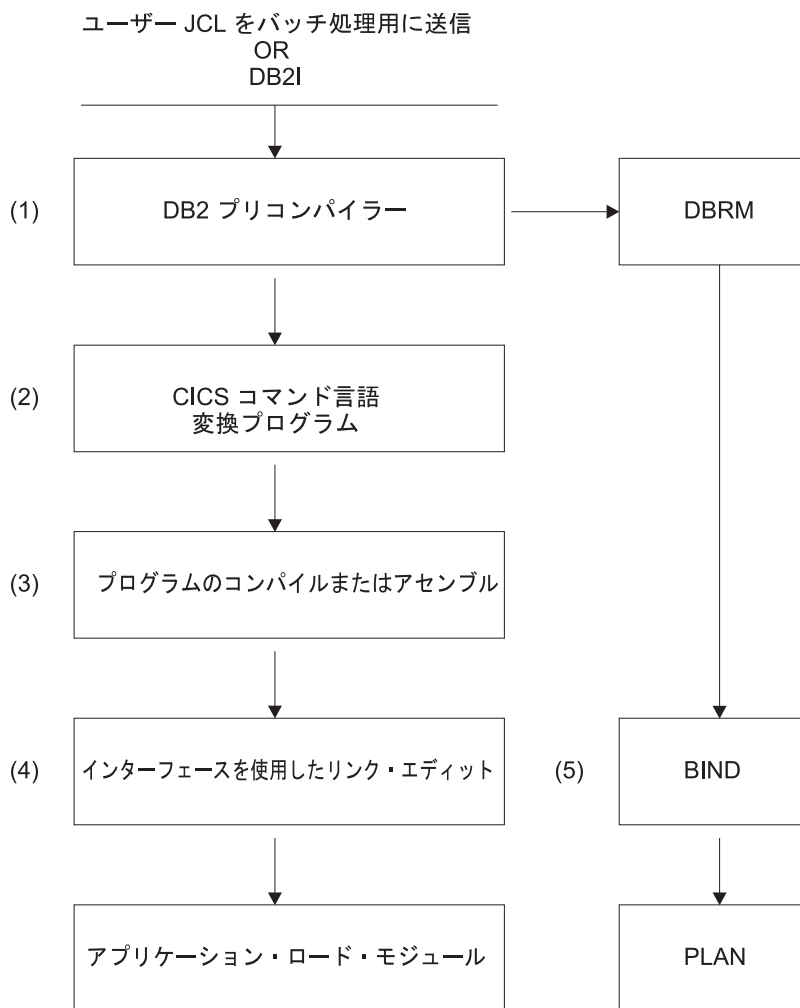


図 3. Db2 にアクセスする CICS アプリケーション・プログラムを準備するステップ

バインド・プロセス

Db2 にアクセスする CICS アプリケーションを実行可能にするには、バインド・プロセスを行う必要があります。

バインド・プロセスを行うには、以下のものがが必要です。

- Db2
- アプリケーション内の各プログラム用に Db2 プリコンパイラーで生成される DBRM (データベース要求モジュール)。

DBRM には、Db2 プリコンパイラーがアプリケーション・プログラムから抽出した SQL ステートメントが含まれます。バインド・プロセスで DBRM 内の SQL ステートメントは、SQL ステートメントの実行時に Db2 で使われる制御構造に変換され、こうして作動可能な (「バインドされた」) 形式になります。結果としてできる成果物を 1 つのパッケージにすることも、アプリケーション・プランの中に直接入れることもできます (『計画、パッケージ、および動的計画出口』を参照してください)。このようなプロセス全体を DBRM の「バインド」と呼びます。バインドについて詳しくは、Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』を参照してください。CICS Db2 環境でのバインド・プロセス中に選択すべきオプションの詳細については、プログラムのバインド・オプションと考慮事項を参照してください。

計画、パッケージ、および動的計画出口

アプリケーション計画により、実行時にアプリケーション・プログラムから Db2 データにアクセスできます。計画には、アプリケーション・プログラムから作成された DBRM に由来する、バインド済みで操作可能な形式の SQL ステートメントが含まれています。また、計画はアプリケーション・プロセス全体を、それが使用される Db2 のローカル・インスタンスに関連付けます。

DBRM からの操作可能な SQL ステートメントを計画に直接配置することができます。その場合、DBRM は計画にバインドされています。あるいは、単一の DBRM からの操作可能な SQL ステートメントを含むパッケージに (BIND PACKAGE コマンドを使用して) DBRM をバインドすることもできます。関連するパッケージはコレクションにグループ化できます。その後、パッケージ・リストにパッケージ名またはコレクション名を含め、パッケージ・リストを計画にバインドすることができます。単一の計画に、パッケージ・リストと、計画に直接バインドされた DBRM の両方を含めることができます。単一の CICS アプリケーションからの DBRM を使用して計画を作成したり、複数のアプリケーションからの DBRM を使用して単一の計画を作成することができます。計画とパッケージのさまざまな設計の利点および欠点については、CICS アプリケーションと Db2 計画およびパッケージの間の関係の設計で説明します。

アプリケーション計画は、作成するだけでなく、維持する必要もあります。計画を使用して、1 つ以上のアプリケーション・プログラムで SQL ステートメントを変更する場合は、変更されるアプリケーション・プログラムの DBRM を再作成する必要があります。古いバージョンの DBRM を直接計画にバインドした場合は、変更プログラムと未変更プログラムの両方について、その計画に直接バインドされた

すべての DBRM を識別し、それらすべてを計画に再度バインドする必要があります。DBRM を計画にバインドしている間は、アプリケーションは Db2 にアクセスするために計画を使用することはできません。ただし、変更されたアプリケーション・プログラム用の古いバージョンの DBRM をパッケージにバインドしてから、計画のパッケージ・リストにパッケージ (またはパッケージを含むコレクション) の名前を含めた場合は、他のパッケージや直接バインドされた DBRM を計画に再度バインドする必要はありません。変更されたアプリケーション・プログラムの新規バージョンの DBRM は、古いバージョンと同じ名前のパッケージにバインドしてください。計画を再度バインドする必要はありません。新規バージョンのパッケージは検出されます。パッケージを変更している間は、アプリケーション・プログラムは引き続き、計画に直接バインドされた DBRM と他のパッケージを使用できます。計画の保守について詳しくは、プログラム変更後に何をバインドするかを参照してください。

Db2 に送信されたスレッドはそれぞれ計画に関連します。スレッドについて詳しくは、3 ページの『概要: スレッドの働き』を参照してください。各タイプのスレッドが使用する計画は、DB2CONN 定義 (プール・スレッドの場合) またはスレッドの DB2ENTRY 定義 (エントリー・スレッドの場合) で指定されます。CICS は、アプリケーションが Db2 にアクセスするためのスレッドの使用を要求する際に、そのタイプのスレッドに関連付けられている計画の名前を Db2 に知らせ、Db2 はその計画を見つけます。各タイプのスレッドの定義では、特定の計画を指定することも、動的計画出口 (スレッドを要求したトランザクションで使用する計画を判別するルーチン) を指定することもできます。

プール・スレッドまたはエントリー・スレッドの定義で特定の計画を指定した場合は、そのタイプのスレッドを使用するすべてのトランザクションでその計画を使用する必要があります。エントリー・スレッド・タイプを使用できるトランザクションは、スレッドの DB2ENTRY および DB2TRAN 定義で指定されます。スレッドの DB2ENTRY 定義で特定の計画を指定する場合、それらすべてのトランザクション ID で実行される可能性があるすべてのアプリケーション・プログラムからの DBRM が同じ計画にバインドされるか、同じ計画にリストされるパッケージにバインドされる必要があります。それらのトランザクション ID で実行されるいずれかのアプリケーション・プログラムからの DBRM が計画に直接バインドされており、それらのアプリケーション・プログラムのいずれかで SQL ステートメントを変更すると、計画に直接バインドされたすべての DBRM を再度バインドしている間、計画全体がアクセス不能になります。これは、計画の保守時には、トランザクションがそのタイプのエントリー・スレッドを使用できないことを意味します。Db2 にアクセスする CICS アプリケーションでもプール・スレッドを使用する可能性があるため、DB2CONN 定義でプール・スレッドに特定の計画を指定する場合には、それらすべてのアプリケーションからの DBRM を使用して、計画を準備し、保守する必要があります。DBRM のいずれかが計画に直接バインドされている場合は、計画全体が保守時にアクセス不能となり、どのトランザクションもプール・スレッドを使用できなくなります。

計画を保守している間にさまざまなスレッドが使用不可にならないようにするための方法は 2 つあります。最善の解決策は、DBRM を計画に直接バインドせずに、パッケージを使用してバインドすることです。各 DBRM をパッケージとしてバインドし、計画のパッケージ・リストに組み込む場合、個々のパッケージを保守している間も計画には引き続きアクセス可能です。特定のプログラムで保守作業を行っ

ている間は、そのプログラムを含む計画が引き続きアクセス可能なので、それらの計画に関連するプール・スレッドまたはエントリー・スレッドは引き続き使用可能です。これは、スレッドごとに特定の計画を安全に指定できることを意味します。パッケージの使用を開始する場合は、パッケージの実装方法の詳細について、Db2 パッケージの使用および Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』を参照してください。

Db2 でパッケージが使用できるようになる前に開発された代替解決策は、動的計画出口を使用することです。動的計画出口を使用するということは、スレッドのタイプごとに特定の計画を指定する必要がないことを意味します。したがって、特定の計画が保守時にアクセス不能な場合でも、スレッドは引き続き使用可能です。この解決策を実行するには、(それぞれが少数の密接に関連したプログラムからの DBRM を含む) CICS アプリケーション用の多数の小さな計画を作成します。次に、スレッドのタイプごとに DB2CONN または DB2ENTRY 定義の PLAN 属性に計画名を指定する代わりに、PLANEXITNAME 属性に出口プログラムを指定します。アプリケーション・プログラムがその最初の SQL ステートメントを発行し、特定のタイプのスレッドが要求されると、スレッド定義で指定した出口プログラムがそのアプリケーション・プログラム用に使用する計画を選択します。特定の計画が保守時にアクセス不能な場合、その計画を必要とするアプリケーション・プログラムはスレッドを使用できませんが、他のアプリケーション・プログラムは独自の計画で同じタイプのスレッドを使用できます。ただし、あるタイプのスレッドの特定のインスタンスが、使用するアプリケーション・プログラム用にいったん作成されると、そのスレッド・インスタンスは動的計画出口で選択された計画に関連付けられます。スレッドを再使用する別のアプリケーション・プログラムでは、その同じ計画を使用する必要があります。動的計画出口でアプリケーション・プログラム用の別の計画を選択する場合は、正しい計画で別のスレッドを検索または作成する必要があります。これにより、スレッド再使用の機会が少なくなります。動的計画出口について詳しくは、動的計画出口を参照してください。

第 2 章 CICS Db2 接続の定義

CICS Db2 接続の構造とパフォーマンスは、DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトを定義するときに決定されます。

このタスクについて

DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトは、CICS Db2 接続のグローバル属性、CICS トランザクションと Db2 リソース (アプリケーション計画とコマンド・プロセッサを含む) の間の関係、各タイプのスレッドの属性、および各トランザクションが使用できるスレッドのタイプを記述します。

概要: CICS Db2 接続の定義方法

CICS Db2 接続の定義には、DB2CONN (Db2 接続定義)、DB2ENTRY (Db2 エントリー定義)、および DB2TRAN (Db2 トランザクション定義) という 3 つの異なる CICS リソース定義が関与します。

CICS Db2 接続は、CICS と Db2 との間の全体的な接続と、スレッドと呼ばれる個々の接続から成ります。これらのリソース定義を使用して、全体的な接続の属性と、さまざまなタイプのスレッドの属性を定義できます。キー・トランザクション用に特別に定義したエントリー・スレッドがある場合は、CICS Db2 接続機能に対して、どの CICS トランザクションがそれらのスレッドを使用できるのかを知らせることができます。

CICS に対して DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトを定義する方法はいくつかあります。以下のことを行うことができます。

- IBM® CICS Explorer® を使用して DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトを定義してインストールします。
- CICSplex SM を使用してオブジェクトを定義します。
- RDO を使用して DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトを定義してインストールします。
- CICS バッチ・ユーティリティー DFHCSDUP を使用してオブジェクトを定義します。

各オブジェクトの有効範囲は以下のとおりです。

DB2CONN

DB2CONN は CICS Db2 接続のメイン定義です。CICS Db2 接続を開始するためには、DB2CONN リソース定義をインストールしておく必要があります。

DB2CONN 定義を使用して以下の項目を定義します。

- 全体的な CICS Db2 接続の属性は以下のとおりです。
 - CICS が接続する Db2 サブシステム、または CICS が接続するアクティブ・メンバーを Db2 が取得する Db2 サブシステムのデータ共有グループ。

- Db2 への接続が失敗した場合に CICS が自動的に再接続をするかどうか、および、グループ接続機能を使用する場合に CICS が同じ Db2 サブシステムに再接続するかどうか。
- CICS Db2 接続機能がスレッドにサインオンするための汎用許可 ID (スレッドに関して他に許可が必要ない場合)。
- CICS が Db2 に対するスレッドの実行のために一度に使用できる TCB の総数の限度。
- 保護されたスレッドが終了するまでの存続時間。
- スレッドが終了するまでに再利用できる回数の限度。
- エラー・メッセージを CICS に伝達する方法、およびスレッドが失敗した場合にトランザクションが行う処理の内容。
- メッセージおよび統計の送信先。
- コマンド・スレッドの属性は以下のとおりです。コマンド・スレッドは、DSNC トランザクションが Db2 コマンド の実行に使用する特殊スレッドです。
 - CICS が一度に使用できるコマンド・スレッド数の限度。
 - コマンド・スレッドが要求されたときに Db2 が検査する 許可 ID のタイプ (例えば、ユーザー ID、トランザクション ID、および汎用 CICS Db2 接続機能 ID など)。
- プール・スレッドの属性は以下のとおりです。プール・スレッドは、トランザクションが特別なコマンドまたはエントリー・スレッドを実行する必要がない場合、またはトランザクションが使用する特殊スレッドが残っていない場合に使用する汎用スレッドです。
 - プール・スレッドが要求されたときに Db2 が検査する 許可 ID のタイプ (例えば、ユーザー ID、トランザクション ID、および汎用 CICS Db2 接続機能 ID など)。
 - CICS メイン TCB に対するスレッド TCB の相対的な優先度。
 - CICS が一度に使用できるプール・スレッド数の限度。
 - トランザクションがプール・スレッドを取得できない場合に、取得するまで待機するか、異常終了するか。
 - プール・スレッドに使用するアプリケーション計画出口または動的計画出口 (計画、パッケージ、および動的計画出口を参照してください)。
 - トランザクションがスレッドを使用している間のどの時点で Db2 アカウンティング・レコードを作成するか。
 - デッドロックがある場合に、トランザクションがスレッドを使用して行った変更をロールバックするかどうか。

CICS に一度にインストールできる DB2CONN 定義は 1 つだけです。また、システムにインストールするすべての DB2ENTRY 定義および DB2TRAN 定義は、DB2CONN 定義に関連付けられます。DB2CONN 定義を破棄する場合は、すべての DB2ENTRY 定義および DB2TRAN 定義も破棄されます。CICS が Db2 に接続されている間は、DB2CONN 定義を破棄して他の定義をインストールすることはできません。

DB2CONN リソース定義と DB2CONN システム初期設定パラメーターを混同しないでください。後者は CICS が初期設定時に自動的に Db2 接続を開始するかどうかを指定するものです。

CICS Db2 接続は、1 つの DB2CONN 定義がインストールされていれば開始できます。接続を確立するために DB2ENTRY 定義および DB2TRAN 定義は必要ありません。このようにすると、キー・トランザクション用の特殊スレッド (エントリー・スレッド) はありません。すべてのトランザクションがプールの汎用スレッドを使用し、最も重要なトランザクションが Db2 への各自の接続を行うのに、最も重要でないトランザクションと同じだけ待機しなければなりません。重要なトランザクションが優先されるようにするには、それらの DB2ENTRY を作成し、必要であれば DB2TRAN 定義も作成します。

DB2ENTRY

さまざまなタイプのエントリー・スレッドを定義するために、数多くの DB2ENTRY 定義をセットアップすることができます。指定したトランザクションでエントリー・スレッドを使用することで、Db2 リソースへの優先的なアクセス権限 (または特殊なアクセス権限) を得ることができます。実質的には、それらのトランザクションだけが使用できるいくつかのスレッドを予約することになります。また、各種エントリー・スレッドのいくつかを保護することもできます。これによって集中的に使用されるトランザクションのパフォーマンスが改善されます。

DB2ENTRY 定義では、DB2ENTRY に関連付けられている特定のトランザクション、またはトランザクションのグループを (後者の場合はワイルドカードを使用して) 指定したり、その中で定義されているエントリー・スレッドのタイプを使用したりすることができます。これらのトランザクションがスレッドを要求すると、そのタイプのエントリー・スレッドが (使用可能であれば) CICS Db2 接続機能によってそれらのトランザクションに与えられます。他のトランザクションで同じタイプのエントリー・スレッドを使用する場合は、それらのトランザクション用の DB2TRAN 定義を作成できます。これによって、それらのトランザクションが特定の DB2ENTRY に関連付けられていることを CICSに知らせます。

各 DB2ENTRY 定義を使用して以下の項目を定義できます。

- このタイプのエントリー・スレッドを使用できるトランザクション、またはトランザクションのグループ (後者の場合は名前にワイルドカードを使用して定義する)。
- このタイプのエントリー・スレッドの属性は以下のとおりです。
 - このタイプのエントリー・スレッドが要求されたときに Db2 が検査する許可 ID のタイプ (例えば、ユーザー ID、トランザクション ID、汎用 CICS Db2 接続機能 ID)。
 - CICS メイン TCB に対するスレッド TCB の相対的な優先度。
 - CICS が一度に使用できるこのタイプのエントリー・スレッド数の限度。
 - 使用されていないときに一定期間保護される、このタイプのエントリー・スレッドの数。

- DB2ENTRY に関連付けられているトランザクションがこのタイプのエントリー・スレッドを取得できない場合に、このタイプのエントリー・スレッドを待機するか、オーバーフローしてプール・スレッドを使用するか、異常終了するか。
- このタイプのエントリー・スレッドに使用するアプリケーション計画出口または動的計画出口（計画、パッケージ、および動的計画出口を参照してください）。
- トランザクションがスレッドを使用している間のどの時点で Db2 アカウンティング・レコードを作成するか。
- デッドロックがある場合に、トランザクションがスレッドを使用して行った変更をロールバックするかどうか。

DB2ENTRY は、トランザクションがそのタイプのエントリー・スレッドを使用している間は、破棄することができません。まず使用不可にして DB2ENTRY のアクティビティを停止してから、破棄します。DB2ENTRY を破棄する場合、それと関連付けられている DB2TRAN は「孤立」し、そこにリストされているトランザクションはプール・スレッドを使用することになります。

コマンド・レベル・インタープリター・トランザクションである CECI を使用してアプリケーション・プログラムの CICS コマンドの構文を検査する場合、その状況では CICS はトランザクション ID CECI を使用して一致する DB2ENTRY 定義がないかを検査することに留意してください。ご使用のアプリケーション・プログラム用の DB2ENTRY をセットアップしてあり、CECI の使用時にそれを複製する場合は、CECI トランザクションと同じプロパティを指定して DB2ENTRY をセットアップします。

DB2TRAN

DB2TRAN 定義を使用して、追加のトランザクションと特定の DB2ENTRY を関連付けることができます。そのようにすると、トランザクションはそのタイプのエントリー・スレッドを使用します。特定のトランザクションにインストールできる DB2TRAN 定義は 1 つだけです。各 DB2TRAN 定義で、以下の項目を指定します。

- DB2ENTRY の名前。
- 特定の DB2ENTRY に関連付けられていてこのタイプのエントリー・スレッドを使用できるトランザクション、またはトランザクションのグループ(後者の場合は名前にワイルドカードを使用して指定する)。

これらのトランザクションがスレッドを要求すると、CICS Db2 接続機能は、それらが特定の DB2ENTRY に関連付けられていることを認識し、DB2ENTRY 定義そのものに指定されているトランザクションのように扱います。

Db2 グループ接続機能の使用

CICS と Db2 の接続を定義する場合、CICS を特定の Db2 サブシステムに接続することを選択できます。

このタスクについて

DB2CONN 定義で DB2ID 属性を使用して、その Db2 サブシステムの名前を指定できます。ただし、Db2 以降を使用する DB2 バージョン 7 サブシステムが複数ある場合は、Db2 のグループ接続機能を使用して、指定したサブシステム 1 つだけでなく、任意のサブシステムに CICS を接続することができます。

グループ接続は Db2 機能の 1 つであり、特定の 1 つの Db2 サブシステムではなく、複数の Db2 サブシステムを含むデータ共有グループの任意のメンバーに CICS を接続することを可能にします。グループ接続機能は、CICS への接続に使用するローカル MVS™ イメージ上でアクティブなグループ・メンバーを 1 つ選択します (他の MVS イメージ上でアクティブなメンバーは選択対象になりません)。

グループ接続機能を使用するには、以下のようにします。

手順

1. DB2CONN 定義の DB2GROUPID 属性を使用して、グループ接続機能をアクティブにします。DB2ID 属性を使用して個々の Db2 サブシステムの ID を指定するのではなく、Db2 サブシステムのグループを表すグループ接続名を指定します。DB2 バージョン 10 では、サブグループの接続名を使用して、グループのサブセットを指定することもできます。グループ接続とは、DB2GROUPID を指定することで複数のクローン AOR で 1 つの共通の DB2CONN 定義を使用できることを意味します。そして、CICS はデータ共有グループまたはサブグループのいずれかのアクティブなメンバーに接続します。DB2CONN 定義を作成してインストールする方法については、DB2CONN リソースを参照してください。
2. CICS と Db2 の間の接続が切断された場合に未確定の作業単位を解決するための DB2CONN 定義の RESYNCMEMBER 属性を指定します。接続が切断された場合、一般には、CICS は同じ Db2 サブシステムに再接続するのではなく、Db2 サブシステムのデータ共有グループの別のメンバーを選択します。つまり、CICS が最初に接続した Db2 サブシステムに未確定の作業単位が保持された場合、その作業単位を解決できません。RESYNCMEMBER 属性とその設定方法については、未確定作業単位 (UOW) の解決を参照してください。
3. 接続が確立された後、**INQUIRE DB2CONN DB2ID()** コマンドを使用して、データ共有グループのどのメンバーが現行の接続用に選択されたかを調べます。
4. CICS を特定の Db2 サブシステムに接続させたい場合は、グループ接続をオーバーライドします。例えば、CICS を「xyz」という ID の Db2 サブシステムに接続させるには、次を使用して DB2ID を指定します。
 - **SET DB2CONN DB2ID(xyz)** コマンド
 - **DSNC STRT xyz** コマンド (DSNC STRT を参照)

上記の方法はどちらも、インストール済みの DB2CONN 定義に DB2ID を設定してグループ接続をオーバーライドします。

タスクの結果

上記のステップで DB2ID を指定すると、インストール済みの DB2CONN 定義の DB2GROUPID 属性が無効になります。グループ接続の使用に戻すには、**SET DB2CONN DB2GROUPID()** コマンドを使用して、DB2GROUPID 属性を再設定します。

ただし、**INITPARM**=(DFHD2INI=*db2id*) システム初期設定パラメーターでの DB2ID の指定は、グループ接続をオーバーライドしません。DB2GROUPID が DB2CONN 定義の中で設定されている場合、**INITPARM** 設定は無視されます。このパラメーターについて詳しくは、INITPARM システム初期設定パラメーターを参照してください。

MAXOPENTCBS システム初期設定パラメーターおよび TCBLIMIT

CICS Db2 接続は、ユーザーが定義する DB2CONN、DB2ENTRY、および DB2TRAN オブジェクトだけでなく、システム初期設定パラメーター MAXOPENTCBS の影響も受けます。MAXOPENTCBS は、CICS 領域で同時に使用できる L8 または L9 モードのオープン TCB の総数を制御します。

CICS が Db2 に接続する場合、CICS は L8 および L9 モードのオープン TCB を使用して Db2 内でスレッドを実行するので、MAXOPENTCBS は重要です。L8 および L9 モードのオープン TCB は、OPENAPI オプションで有効にされるタスク関連ユーザー出口で使用するために予約されています。それらのユーザー出口に、CICS Db2 のタスク関連ユーザー出口も含まれています。

オープン・トランザクション環境では、この CICS Db2 タスク関連ユーザー出口が Db2 内でスレッドを実行するために使用できる L8 および L9 モードのオープン TCB の数が、DB2CONN 定義の TCBLIMIT 属性によって制御されます。TCBLIMIT の値に達すると、CICS Db2 タスク関連ユーザー出口は MAXOPENTCBS で制御されるプールから TCB を取得することはできませんが、その TCB を使用して Db2 内でスレッドを実行できるようになるまで待機します。別のタスクが L8 または L9 モードの TCB を使用した Db2 内でのスレッド実行を停止し、スレッド実行のために使用されている TCB の数が TCBLIMIT を下回ると、待機していたタスクが独自の L8 または L9 モード TCB を使用して Db2 内でスレッドを実行できるようになります。しかし、MAXOPENTCBS の値に達した場合は、CICS 領域でそれ以上の L8 および L9 モードのオープン TCB は許可されないため、CICS Db2 タスク関連ユーザー出口は、使用する L8 または L9 モード TCB を取得することさえできなくなります。別のタスクが L8 モード TCB または L9 モード TCB (OPENAPI プログラムで使用するもの) を解放し、それが MAXOPENTCBS で制御されるプールに戻されるまで、待機しなければなりません。L8 TCB が解放された場合、タスクはその TCB を使用できます。解放されたものが L9 TCB であった場合、CICS はその TCB を「スチール」します。つまり、L9 TCB を切り離し、その代わりに L8 TCB を接続してタスクで使用できるようにします。

Db2 ワークロードに対応できる十分な数の L8 および L9 モード・オープン TCB を確保するために、MAXOPENTCBS システム初期設定パラメーターの制限を、DB2CONN 定義の TCBLIMIT 属性に設定する制限よりも大きい値にしてください。MAXOPENTCBS が TCBLIMIT より小さいと、TCBLIMIT に達する前にシステムの L8 および L9 モード・オープン TCB が不足する可能性があります。CICS が Db2 に接続したときに、SIT の MAXOPENTCBS 設定値が DB2CONN 定義の TCBLIMIT 設定より小さいことを CICS Db2 接続機能が検出すると、警告メッセージ DFHDB2211 が出されます。この警告メッセージを受け取った場合は、MAXOPENTCBS 制限を調整してください。

また、トランザクション分離がアクティブな状態で実行する場合は、MAXOPENTCBS をタスク最大数 (MXT) 以上の値に設定してください。その設定により、誤ったサブスペースに割り振られている TCB が原因で TCB スチーリングが発生する可能性を最小化できます。

SQL 処理時に何が起きるか

SQL 呼び出しを行う CICS トランザクションの処理で発生する Db2 の主なアクティビティは、スレッド作成、SQL 処理、コミット処理、スレッド解放、およびスレッド終了です。

トランザクションごとに、主に以下のアクティビティが実行されます。各アクティビティで行われる正確な処理は、DB2CONN および DB2ENTRY オプション、BIND オプション、およびパッケージの使用有無によって決まります。

スレッドの作成

スレッド作成時には、いくつかのアクティビティが実行されます。例えば、プラン許可検査があります。これは必ず実行されます。

スレッド作成時には、指定された BIND オプションに応じて以下のアクティビティが実行されます。

- サインオンする。
- スレッドの最大数を検査する。
- アプリケーション・プランの場合は、スケルトン・カーソル表 (SKCT) とヘッダーを環境記述マネージャー (EDM) プールにロードする (それらがプールにまだ存在しない場合)。
- カーソル表 (CT) に SKCT ヘッダーのコピーを作成する。
- プラン許可検査を実行する。

アプリケーション・プランの制御ブロックである SKCT は、いくつかのセクションに分かれています。SKCT のヘッダーおよびディレクトリーには制御情報が保管されています。SQL セクションには、アプリケーションから渡された SQL ステートメントが保管されています。プランを実行するスレッドごとに、SKCT のコピーである CT が作成されます。スレッド作成時には、ヘッダーとディレクトリーのみがロードされます (それらが EDM プールにまだ存在しない場合)。

スレッド作成時に、SQL セクションのプラン・セグメントが CT にコピーされることはありません。対応する SQL ステートメントが実行されるときに、SQL セクションがセクション単位でコピーされます。RELEASE(DEALLOCATE) を指定した保護スレッドの場合、すべてのセグメントが CT にコピーされるまで、CT のサイズは増加します。

トランザクションの SQL ステートメントをプランではなくパッケージにバインドした場合、Db2 は SKCT ではなくスケルトン・パッケージ表 (SKPT) を、CT ではなくパッケージ表 (PT) を使用します。SKPT は最初の SQL ステートメントが実行されるたびに割り振られます。スレッドの作成時には割り振られません。

SQL 処理

処理される SQL ステートメントごとに、いくつかのアクティビティーが実行されます。これらのアクティビティーは、スレッドの再利用オプションや BIND オプションに応じて異なります。

- 新しい許可 ID でスレッドを再利用するトランザクションの最初の SQL 呼び出しでは、以下のアクションが実行される場合があります。
 - サインオン
 - 許可検査
- SKCT の SQL セクションをロードする (このセクションがまだ EDM プールに存在しない場合)。
- SKCT の SQL セクションのコピーを CT に作成する (コピーがまだ存在しない場合)。
- 参照されている TS ロックを取得する (まだ取得していない場合)。
- EDM プールに DBD をロードする (まだ存在しない場合)。
- SQL ステートメントを処理する。

SQL ステートメントがパッケージに含まれている場合は、SKPT のディレクトリーとヘッダーがロードされます。バインディング・プランの SKCT および CT の場合と同様に、PT はスレッドの作成時ではなくステートメントの実行時に割り振られます。

コミット処理

コミット時には、ページ・ロックや TS ロックの解放など、いくつかのアクティビティーが実行されます。

コミット時には、BIND オプションに応じて以下のアクティビティーが実行されます。

- ページ・ロックを解放する
- RELEASE(COMMIT) が指定されている場合:
 - TS ロックを解放する
 - CT ページを解放する

スレッド解放

トランザクションは、使用しているスレッドをさまざまに異なるタイミングで解放します。そのタイミングは、端末向けトランザクションかどうかによって、また、指定されているオプションによって異なります。

トランザクションが端末向けである場合、または非端末向け¹ で DB2CONN に NONTERMREL=YES が指定されている場合、スレッドはタスク終了 (EOT) だけでなく SYNCPOINT でも解放されます。このため、多数の SYNCPOINTS を発行するトランザクションには保護スレッドを使用すると効率的です (BIND オプション ACQUIRE(USE) および RELEASE(DEALLOCATE) と組み合わせる場合)。その場合、以下のアクティビティーを実行するリソースが、同期点ごとに保存されます。

- スレッドの終了と開始。
- TS ロックの解放と獲得。

- プランのセグメントの解放と CT へのコピー。

以下の場合、SYNCPOINT 時にスレッドは解放されません。

- 保留カーソルがオープンである。
- 特定の Db2 変更可能特殊レジスターが初期値ではない。これらの特殊レジスターは、Db2 for z/OS 製品資料内の『SQL: Db2 の言語』にリストされています。
- Db2 変更可能特殊レジスター CURRENT DEGREE が、CICS タスクの存続期間中に絶えず変化する。

トランザクションが端末向けでなく、NONTERMREL=NO を指定した場合、スレッドはタスク終了時 (EOT) にのみ解放されます。EXEC CICS SYNCPOINT コマンドの後にスレッドを再利用するために保護スレッドを使用する必要はありません。ただし、使用頻度が高いトランザクションである場合は、保護スレッドを使用することもできます。BIND オプション ACQUIRE(USE) と RELEASE(DEALLOCATE) を指定すると、多数の同期点を持つ端末向けトランザクションと同じ利点を得ることができます。

注: **1** 非端末向けトランザクションとは、端末に関連付けられていないトランザクションです。

スレッド終了

スレッド終了時には、TS ロックの解放や作業ストレージの解放など、いくつかのアクティビティーが実行されます。

スレッド終了時には、BIND オプションに応じて以下のアクティビティーが実行されます。

- RELEASE(DEALLOCATE) が指定されている場合:
 - TS ロックを解放する
 - CT ページを解放する
- 作業ストレージを解放する

スレッドの作成、使用、および終了

CICS Db2 接続機能では、コマンド・スレッド、エントリー・スレッド、およびプール・スレッドという 3 種類の主なスレッドが使用されます。スレッドの作成、使用、および終了には特定の規則が適用されます。

CICS を Db2 に接続すると、CICS Db2 接続機能が既に実行されているオープン TCB が、スレッド TCB になります。スレッドを実行する必要があると、CICS Db2 接続機能は、使用したい Db2 接続制御ブロックおよびスレッドをこのオープン TCB に関連付けます。すると、そのオープン TCB がスレッドを実行します。不要になったスレッドはオープン TCB から切り離されます。そして、Db2 接続制御ブロックとスレッドは別のオープン TCB で再利用できるようになります。

スレッドの作成、使用および終了には、以下の一般的な規則が適用されます。

- CICS が Db2 に接続されていて、オープン TCB がスレッド TCB として使用されている場合、SQL 要求を Db2 に渡すためには、トランザクション用に使用

可能なスレッドが存在しなければなりません。オープン TCB は自らをスレッドに関連付けるとスレッド TCB になります。この状態はスレッドから自らを切り離すまで続きます。

- スレッドが作成され、新しい許可 ID の別のトランザクションがスレッドを再利用しようとする、Db2 はその新しい許可 ID の許可検査を行います。
- 端末向けトランザクションは、通常、同期点およびタスク終了時点でスレッドを解放します。保持カーソルが開いた状態である場合、または変更可能な特殊レジスターが初期状態でない場合、スレッドは同期点で解放されません。
- 非端末向けトランザクションはタスク終了時点でのみスレッドを解放します (DB2CONN で NONTERMREL=YES が指定されている場合を除く)。
- トランザクションがスレッドを解放すると、そのスレッドは、同じプランを指定していて同じ DB2ENTRY に定義されている別のトランザクションで再利用できるようになります。プール・スレッドは、同じプランを指定しており、プール・スレッドを使用している待機 (キューに入っている) トランザクションで再利用可能になります。
- 無保護スレッドは、そのスレッドを待機している (キューに入っている) 別のトランザクションがない限り、解放されるとすぐに終了します。
- 保護スレッドは、連続する 2 回の消去サイクルにわたって使用されなければ、終了します。デフォルトの設定では、この平均は約 45 秒です。この値は、DB2CONN の PURGECYCLE パラメーターで変更できます。
- スレッドを使用している CICS タスクがパージまたは強制パージされる時点で Db2 でアクティブなスレッドは、取り消されて終了します。
- DB2CONN の REUSELIMIT パラメーターは、スレッドを終了させる前に再使用できる最大回数を指定します。再利用の制限は、プールと DB2ENTRY の両方の無保護スレッド、および DB2ENTRY の保護スレッドに適用されます。
- THREADWAIT パラメーターは、エントリー・スレッドまたはコマンド・スレッドが不足している場合に、スレッド要求をキューに入れるのか、異常終了するのか、またはプール・スレッドに送信するのかを定義します。
THREADWAIT=POOL の代わりに THREADWAIT=YES が指定されている場合、トランザクションはプール・スレッドに送信されないでキューに入れられます。THREADWAIT=YES を使用すると、スレッドの初期化と終了時のオーバーヘッドがなくなります。エントリー・スレッドの不足のためにトランザクションが待機させられる場合、CICS Db2 接続機能はトランザクションをキューに入れます。このキューに入れる方法の利点は、エントリー・スレッドが作業の現行部分を終了すると、次のトランザクションが即時に継続されることです。
- TCBLIMIT は Db2 スレッドを実行するために使用できる TCB の最大数を指定し、これにより、アクティブな Db2 スレッドの最大数を制限します。
THREADLIMIT はアクティブな Db2 スレッドの最大数を指定します。
THREADLIMIT は動的に変更されます。CTHREAD は ZPARMS に指定され、Db2 全体の並行スレッドの数を定義します。TSO ユーザー、CICS および IMS™ システム、および Db2 にアクセスしているその他のシステムからのすべてのアクティブ・スレッドの合計が CTHREAD を超えないようにする必要があります。そうしないと、応答時間が予測不能になることがあります。応答時間が予測不能になると、CICS Db2 接続機能の「create thread」要求が Db2 によってキューに入れられ、スレッドが使用可能になるまで CICS トランザクションが待ち状態になります。

CICS は、CICS 領域が稼動中に一度に保持できる L8 または L9 モード TCB の総数を管理します。オープン TCB プール内の L8 および L9 モード TCB の最大数は、MXT または MAXOPENTCBS システム初期設定パラメーターのどちらかによって制御されます。MAXOPENTCBS システム初期設定パラメーターが指定された場合、それによってオープン TCB プールの値が設定されます。MAXOPENTCBS システム初期設定が指定されない場合、CICS は CICS 領域に対して指定されたタスクの最大数 (MXT 値) に基づいて、L8 および L9 モードのオープン TCB プールの制限を自動的に設定します。このとき、公式 $(2 * \text{MXT 値}) + 32$ を使用します。オープン・トランザクション環境では、Db2 内でスレッドを実行するために CICS Db2 タスク関連ユーザー出口がこれらのオープン TCB をいくつ使用できるかを、TCBLIMIT が制御します。CICS で設定したこの制限に達すると、CICS 領域内ではそれ以上のオープン TCB が許可されず、CICS Db2 タスク関連ユーザー出口はオープン TCB を取得して使用できなくなります。

CICS を Db2 にリンクする各スレッドは、CICS アドレス・スペース内の TCB に対応しています。アドレス・スペース単位の TCB 数が多すぎる場合は、アクティブ TCB を識別するために MVS ディスパッチャーが TCB をスキャンします。多数の TCB が存在する場合は、プロセッサ時間が著しく損なわれることがあります。ただし、Db2 ワークロードを満たすために使用できる TCB 数が少なすぎる場合は、トランザクションは TCB を取得するために待機する必要があります。

TCBLIMIT 値を大きくするか、または同じ Db2 システムにアクセスする別の CICS システムを設定すると、Db2 の CTHREAD パラメーターを大きくする必要があります。

エントリー・スレッドが保護環境の場合は、アプリケーション計画数を検討し、可能であれば、計画サイズおよびセキュリティの問題を調整する間に使用頻度の高い計画を結合して、計画数を削減します。最初は、計画ごとに 1 つのスレッドから開始する必要があります。トランザクションを大量に処理する環境では、トランザクションのスレッド占有時間に予測トランザクション・レートを掛けて、初期値を見積もることができます。例えば、占有時間が 0.2 秒で、トランザクション・レートが 20 トランザクション/秒の場合 (0.2×20)、初期スレッド数は 3 から 4 となります。

保護エントリー・スレッド

保護エントリー・スレッドは、DB2ENTRY パラメーター PROTECTNUM(n) および THREADLIMIT(n) を使用して定義します。

保護エントリー・スレッドが推奨されるものを次に示します。

- あらゆるタイプの大量のトランザクション
- 多数のコミットがある端末向けトランザクション
- 多数のコミットがある非端末向けトランザクション (DB2CONN に NONTERMREL=YES を指定した場合)

保護エントリー・スレッドは、次のように作成され、使用されて終了します。

TCB 接続

CICS が Db2 に接続されていて、オープン・トランザクション環境を使用している場合、CICS Db2 接続機能呼び出す前に、新規および既存のオープン TCB が CICS タスクに割り当てられます。CICS によって自動的に設定された L8 および L9 モードのオープン TCB の数の制限に達すると、それ以上はオープン TCB を作成できなくなり、オープン TCB が利用できるようになるまで、タスクは CICS ディスパッチャー待機状態に入ります。タスクが終了すると、オープン TCB は CICS ディスパッチャーによって管理されるオープン TCB のプールに返されます。

スレッドの作成

スレッドは、既存のスレッドが使用できない場合にのみ作成されます。タスクに使用できるスレッドがない場合、THREADLIMIT を超過していなければ、新規スレッドが作成されて、そのタスクのオープン TCB に関係付けられます。TCBLIMIT に達している場合は、それ以上の数のオープン TCB を DB2ENTRY のスレッド TCB として使用することはできません。

スレッド終了

保護スレッドの現在の数が PROTECTNUM 値より少ない場合、スレッドが解放されたときに、キューに入っている新しい処理がなければ、そのスレッドには保護のマークが付けられます。そうでない場合、スレッドは終了します。保護スレッドは、連続する 2 回の消去サイクルにわたって使用されなければ、終了します。

スレッドの再利用

同じ DB2ENTRY を使用する他のトランザクションはスレッドを再利用できます (スレッドがある場合)。一般に、スレッドは PURGECYCLE 値に基づき、解放された後約 45 秒間アクティブな状態を維持するからです。

プールへのオーバーフロー

THREADWAIT=POOL を指定した場合、THREADLIMIT の値を超過すると、スレッドに対する要求がプールに転送されます。プールにオーバーフローしたトランザクションは、DB2CONN 定義でプール・スレッドに対して指定された PRIORITY、THREADLIMIT、および THREADWAIT 属性によって制御されるため、DB2ENTRY 定義で指定されたそれらの属性は無視されます。エントリー・スレッドに対して DB2ENTRY 定義で指定されている残りの属性、つまり ACCOUNTREC、AUTHID または AUTHTYPE、DROLLBACK、および PLAN または PLANEXITNAME は、引き続きトランザクションに適用されます。DB2ENTRY 定義の PROTECTNUM 属性は、プールにオーバーフローしたトランザクションにはもはや関係がないので、この設定は無視されます。

クリティカル・トランザクションの無保護エントリー・スレッド

クリティカル・トランザクションの無保護エントリー・スレッドは、DB2ENTRY パラメーター PROTECTNUM(0) および THREADLIMIT(n) を使用して定義します。

これは、次のトランザクションに対して推奨するタイプの定義です。

- 高速な応答時間が必要であるが、非常に少量であるために保護スレッドを使用できないクリティカル・トランザクション
- 並行性が限定されたトランザクション

スレッドの再利用が可能なトランザクション・レートである場合には、並行性が限定されたトランザクションにも保護スレッドを使用できます。

クリティカル・トランザクションの無保護エントリー・スレッドは、次のように作成され、使用されて、終了します。

TCB 接続

CICS Db2 接続機能が開始していても、スレッドと TCB は接続されていません。

TCB はスレッドで必要な場合にのみ接続されます。

スレッドの作成

スレッドはトランザクションで必要な場合にのみ作成されます。

CICS が Db2 に接続されている場合 (そのため CICS がオープン・トランザクション環境を使用している場合) に、使用できるスレッドがないが、オープン TCB をこの DB2ENTRY のスレッド TCB として使用できる場合、THREADLIMIT を超過していなければ、新規スレッドが作成されて、そのタスクのオープン TCB に関係付けられます。TCBLIMIT に達している場合は、それ以上の数のオープン TCB を DB2ENTRY のスレッド TCB として使用することはできません。

スレッド終了

スレッドは、キューに入っているトランザクションがなければ、解放されるとすぐに終了します。

スレッドの再利用

同じ DB2ENTRY を使用するように指定されている他のトランザクションは、使用可能なスレッドがあれば、それを再利用できます。再利用が行われるのは、トランザクションがスレッドを待機しているときに、スレッドが利用可能になった場合のみです。

プールへのオーバーフロー

THREADWAIT=POOL が指定されている場合、THREADLIMIT の値を超過すると、スレッドに対する要求がプールに転送されます。プールにオーバーフローしたトランザクションは、DB2CONN 定義でプール・スレッドに対して指定された PRIORITY、THREADLIMIT、および THREADWAIT 属性によって制御されるため、DB2ENTRY 定義で指定されたそれらの属性は無視されます。エントリー・スレッドに対して DB2ENTRY 定義で指定されている残りの属性、つまり ACCOUNTREC、AUTHID または AUTHTYPE、DROLLBACK、および PLAN または PLANEXITNAME は、引き続きトランザクションに適用されます。DB2ENTRY 定義の PROTECTNUM 属性は、プールにオーバーフローしたトランザクションにはもはや関係がないので、この設定は無視されます。

並行性が限定されたトランザクションに対してプールへのオーバーフローを許可するべきではないことに注意してください。

バックグラウンド・トランザクションの無保護エントリー・スレッド

無保護エントリー・スレッドは、DB2ENTRY パラメーター PROTECTNUM(0)、THREADLIMIT(0)、および THREADWAIT(POOL) を使用して定義します。

無保護エントリー・スレッドは、高速な応答時間が必要でない少量のトランザクションに推奨されます。すべてのトランザクションはプール・スレッドを使用するように強制されます。

バックグラウンド・トランザクションの無保護エントリー・スレッドは、次のように作成され、使用されて、終了します。

TCB 接続

THREADLIMIT=0 であるため、このスレッド定義でサブタスクのスレッド TCB が接続されることはありません。プール・スレッド TCB (または、オープン・トランザクション環境ではオープン TCB) が使用されます。このエントリー定義に関連するすべてのアクティビティは、プール内のスレッドおよび TCB を使用することを強制されます。そのため、トランザクションはプールの PRIORITY、THREADLIMIT、および THREADWAIT パラメーターによって制御されます。エントリー・スレッドに指定した PLAN および AUTHID/AUTHTYPE 値はトランザクションに保持されます。

スレッドの作成

トランザクションにスレッドが必要な場合、プールの THREADLIMIT 値に達していなければ、プールにスレッドが作成されます。

スレッド終了

スレッドが解放されたときに、スレッドのキューに入っているトランザクションが存在しなければ、スレッドは終了します。

スレッドの再利用

同じプランを使用する他のトランザクションは、利用可能になったスレッドを再使用できます。

プール・スレッド

プール・スレッドは、DB2CONN パラメーター THREADLIMIT(n) で定義します。

プール・スレッドが使用される状況には、以下の 4 つがあります。

1. あるトランザクションが、DB2ENTRY または DB2TRAN に指定されていないのに、SQL 要求を出した場合。このトランザクションは、デフォルトのプールを使用し、そのプールに指定されているプランを使用します。
2. あるトランザクションが DB2ENTRY または DB2TRAN を参照する DB2TRAN に指定されているのに、その DB2ENTRY に THREADLIMIT(0) と THREADWAIT(PPOOL) が指定されているために、プールの使用を強制された場合。このトランザクションは、DB2ENTRY に指定されているプランを使用します。
3. あるトランザクションが DB2ENTRY または DB2TRAN を参照する DB2TRAN に指定されているのに、THREADLIMIT 値を超過したためにプールにオーバーフローした場合 (THREADWAIT=PPOOL)。このトランザクションは、DB2ENTRY に指定されているプランを使用します。
4. あるトランザクションが DB2ENTRY または DB2TRAN を参照する DB2TRAN に指定されているのに、その DB2ENTRY が無効にされている場合。DISABLEDACT キーワードが PPOOL に設定されているので、プール・スレッドが使用されます。このトランザクションは、プールに指定されているプランを使用します。

プール・スレッドは常に無保護スレッドです。

プール・スレッドは、次のように作成され、使用されて、終了します。

TCB 接続

CICS Db2 接続機能が開始していても、スレッドと TCB は接続されていません。

TCB はスレッドで必要な場合にのみ接続されます。

スレッドの作成

スレッドはトランザクションで必要な場合にのみ作成されます。

スレッド終了

スレッドは、キューに入っているトランザクションがなければ、解放されるとすぐに終了します。

スレッドの再利用

同じプランを使用する他のトランザクションは、利用可能になったスレッドを再使用できます。プールで再利用が行われるのは、スレッドのキューがあり、そのキューの最初のトランザクションが、解放されるスレッドで使用されていたのと同じプランを要求している場合に限られます。

最適なパフォーマンスのためのスレッド・タイプの選択

パフォーマンスを最適化するために、トランザクションのセットに対して使用するスレッドのタイプと、対応するプランの BIND パラメーターを選択してください。

スレッド・タイプと BIND パラメーターと一緒に選択することが重要です。いくつかのアクティビティーは、BIND パラメーターによって、特定のスレッドまたはトランザクションに関係するかどうかが決まるからです。詳しくは、29 ページの『最適なパフォーマンスのための BIND オプションの選択』を参照してください。

使用可能なスレッドのタイプは、以下のとおりです。

保護エンタリー・スレッド

保護エンタリー・スレッドが推奨されるものを次に示します。

- あらゆるタイプの大量のトランザクション
- 多数のコミットがある端末向けトランザクション
- 多数のコミットがある非端末向けトランザクション (DB2CONN に NONTERMREL=YES を指定した場合)

詳細については、21 ページの『スレッドの作成、使用、および終了』を参照してください。

クリティカル・トランザクションの無保護エンタリー・スレッド

クリティカル・トランザクションの無保護エンタリー・スレッドが推奨されるものを次に示します。

- 高速な応答時間が必要であるが、非常に少量であるために保護スレッドを使用できないクリティカル・トランザクション
- 並行性が限定されたトランザクション

スレッドの再利用が可能なトランザクション・レートである場合には、並行性が限定されたトランザクションにも保護スレッドを使用できます。

バックグラウンド・トランザクションの無保護エントリー・スレッド

バックグラウンド・トランザクションの無保護エントリー・スレッドは、高速な応答時間が必要でない少量のトランザクションに推奨されます。すべてのトランザクションはプール・スレッドを使用するように強制されます。

保護スレッドを使用すると (エントリー・スレッドの DB2ENTRY 定義に PROTECTNUM=n を指定する)、スレッドの作成と終了に要するリソースを減らせるため、パフォーマンスが向上します。保護スレッドは、トランザクションが完了しても終了されず、同じ DB2ENTRY に関連付けられている次のトランザクションで再利用することができます。保護スレッドを使用すると、トランザクションごとにスレッドを作成および終了するために要する作業がほとんどなくなります。パフォーマンスのために、できる限り保護エントリー・スレッドを使用することをお勧めします。特に、使用頻度の高いトランザクションや、多数の SYNCPOINTS を実行するトランザクションがある場合にはお勧めします。例外は、使用頻度が低いトランザクションの場合です。このようなトランザクションでは、保護されたスレッドであっても、トランザクションが行われていない間にスレッドが終了する可能性があります。

アカウンティングの観点から見ると、状況は異なります。アカウンティング・レコードは、スレッドが終了するたびに、また新たにユーザーがサインオンするたびに作成されます。つまり、常に同じスレッドが活動中で、ユーザー ID が変更されることもなければ、アカウンティング・レコードは 1 つしか作成されません。このレコードには、同じスレッドを使用するすべてのトランザクションについて要約された値が記録されますが、特定のトランザクションに値を割り当てることはできません。この問題に対処するには、ACCOUNTREC(UOW) を指定してアカウンティング・レコードを作業単位ごとに分けるか、または ACCOUNTREC(TASK) を指定して CICS タスクごとにアカウンティング・レコードを分けます。CICS Db2 環境でのアカウンティングについて詳しくは、『モニター』の『CICS Db2 環境のアカウンティングとモニター』を参照してください。

複数のトランザクションを、同じ DB2ENTRY を使用するように指定することができます。すべてのトランザクションで同じプランを使用するのが理想的です。少量トランザクションは、独自の DB2ENTRY を使用できます。その場合、スレッドが再利用される可能性はないので、PROTECTNUM=0 を指定する必要があります。代わりに、いくつかの少量トランザクションを同じ DB2ENTRY のグループにして、PROTECTNUM=n を指定する方法もあります。このようにすると、スレッドの使用率が向上し、オーバーヘッドが減少します。

スレッドが作成されるときと、スレッドが新規ユーザーによって再利用されるときには、許可検査が実行されます。セキュリティ検査のオーバーヘッドを回避できることは、保護スレッドを使用することのパフォーマンス上の利点の一つです。つまり、パフォーマンスの観点からすると、PROTECTNUM>0 が指定された同じ DB2ENTRY を使用するように定義されたすべてのトランザクションは、同じ許可 ID を使用する必要があります。少なくとも、AUTHTYPE パラメーターに TERM、OPID、および USERID を指定しないようにする必要があります。これらの値は同じトランザクションでもインスタンスごとに異なることが多いからです。

DB2CONN、DB2ENTRY、BIND の各オプションを調和させることは重要です。詳しくは、『DB2CONN、DB2ENTRY、および BIND オプションの調整』を参照してください。

最適なパフォーマンスのための BIND オプションの選択

プランに関する BIND オプションを選択するときには、そのプランに関連付けられるトランザクションで使われるスレッド・タイプと調和させて選ぶ必要があります。

バインド・プロセスの概要については、バインド・プロセスを参照してください。スレッド・タイプの選択の指針については、27 ページの『最適なパフォーマンスのためのスレッド・タイプの選択』を参照してください。

また、バインド・オプション VALIDATE はパフォーマンスに影響を与える可能性があります。CICS 環境では VALIDATE(BIND) を使用してください。詳しくは、プログラムのバインド・オプションと考慮事項を参照してください。

DB2CONN、DB2ENTRY、BIND の各オプションを調和させることは重要です。詳しくは、『DB2CONN、DB2ENTRY、および BIND オプションの調整』を参照してください。

DB2CONN、DB2ENTRY、および BIND オプションの調整

DB2CONN、DB2ENTRY、および BIND オプションは、さまざまに組み合わせることができます。

パフォーマンスを最適化するために必要な作業のうち最も重要な作業の 1 つは、トランザクション・セットで 1 つ以上の保護スレッドを使用するかどうかを決めることです。27 ページの『最適なパフォーマンスのためのスレッド・タイプの選択』に、この作業についてのアドバイスを詳細に記載しています。次に、SQL トランザクションの処理で発生する主なアクティビティーに関連する処理の総量を最小化するために、BIND パラメーターを定義することを検討します。詳しくは、『最適なパフォーマンスのための BIND オプションの選択』を参照してください。

一般に、DB2CONN、DB2ENTRY、および BIND オプションの初期値は、表 1 に示す値に設定することをお勧めします。トランザクションによっては他の組み合わせのほうがパフォーマンスが向上する場合があります。トランザクションのタイプごとに、推奨されるスレッド・タイプと BIND オプションを示します。トランザクションのプールへのオーバーフローについての推奨事項も記載します。

表 1. 推奨される DB2CONN、DB2ENTRY および BIND オプションの組み合わせ

トランザクションの説明	スレッド・タイプ	オーバーフロー	ACQUIRE	RELEASE
大量 (すべてのタイプ)	保護エントリー	注 1	USE	DEALLOCATE
端末向け、コミット多数 (および NONTERMREL=YES の場合の非端末)	保護エントリー	注 2	USE	DEALLOCATE
少量、短い応答時間が求められる	無保護エントリー	はい	USE	COMMIT

表 1. 推奨される DB2CONN、DB2ENTRY および BIND オプションの組み合わせ (続き)

トランザクションの説明	スレッド・タイプ	オーバーフロー	ACQUIRE	RELEASE
少量、限定的な並行性	無保護エントリー	Never	USE	COMMIT
少量、短い応答時間は求められない	プール	適用外	USE	COMMIT
非端末向け、多数のコミットあり (NONTERMREL=NO)	注 3	注 3	USE	DEALLOCATE
注: 1. Yes。ただし、頻繁に発生しないように十分な数のエントリー・スレッドを定義してください。 2. Yes。ただし、プールにオーバーフローした場合、保護スレッドは使用されません。 3. スレッドは EOT まで保持されます。短いトランザクションにはプール・スレッドを使用してください。長時間実行するトランザクションには、エントリー・スレッドを使用することを検討してください。				

29 ページの表 1 の「限定的な並行性」とは、同時に実行できるトランザクション数が限られていることを意味します。n=1 という特殊なケースがあります。このケースでは、トランザクションは直列化されます。その場合も、トランザクション・レートが十分に高く、保護スレッドを使用することが有効であれば、保護スレッドを使用できます。プールへのオーバーフローを許可した場合、トランザクションは制御できません。スレッドの数を制限してトランザクションを強制的にキューに入れるよりも、特定のクラスで実行するトランザクションの数を制限する CICS メカニズムを使用することをお勧めします。

29 ページの表 1 に示すように、一般には、いくつかの DB2CONN、DB2ENTRY、および BIND オプションの組み合わせをお勧めします。しかし、状況によっては他の組み合わせも使用できます。

表 2 に、DB2CONN、DB2ENTRY、および BIND の指定の組み合わせとして推奨される 3 セットについて、SQL 要求の処理で発生するアクティビティをまとめています。この表では、許可 ID を変更せずに保護スレッドを使用することで得られるパフォーマンス上の利点も説明しています。表中では、次のようなマークで必須アクティビティを表しています。

X 必須のアクティビティ

(1) 新規許可 ID の場合にのみ必須

(2) SQL セクションがまだ EDM プールにない場合にのみ必須

(3) SQL セクションがまだカーソル・テーブルにない場合にのみ必須

表 2. DB2CONN、DB2ENTRY、および BIND の指定に応じて SQL 要求の処理で発生するアクティビティ

アクティビティ型	保護スレッド		無保護スレッド	
	ACQUIRE(USE) RELEASE(DEALLOCATE)		ACQUIRE(USE) RELEASE(COMMIT)	ACQUIRE(USE) RELEASE(DEALLOCATE)
	スレッドごとの アクティビティ	トランザクションごとの アクティビティ	トランザクションごとの アクティビティ	トランザクションごとの アクティビティ
スレッドの作成:	X		X	X
SIGNON	X	(1)	X	X

表 2. DB2CONN、DB2ENTRY、および BIND の指定に応じて SQL 要求の処理で発生するアクティビティ (続き)

アクティビティ型	保護スレッド		無保護スレッド	
	ACQUIRE(USE) RELEASE(DEALLOCATE)		ACQUIRE(USE) RELEASE(COMMIT)	ACQUIRE(USE) RELEASE(DEALLOCATE)
	スレッドごとの アクティビティ	トランザクシ ョンごとのアクテ ィビティ	トランザクションご とのアクティビティ	トランザクションご とのアクティビティ
許可検査	X	(1)	X	X
SKCT ヘッダーのロード	X		X	X
CT ヘッダーのロード	X		X	X
すべての TS ロックの獲得		X	X	X
すべての DBD のロード		X	X	X
SQL ステートメントごと: SKCT SQL セクション のロード		(2)	(2)	(2)
CT コピーの作成		(3)	X	X
すべての TS ロックの獲得			X	X
すべての DBD のロード			X	X
コミット: ページ・ロックの解放		X	X	X
TS ロックの解放			X	
CT ページの解放			X	
スレッドの終了:	X		X	X
TS ロックの解放	X		X	X
CT ページの解放	X			X
作業ストレージの解放	X		X	X

第 3 章 CICS Db2 での管理

この情報では、CICS Db2 接続機能の操作について説明します。

CICS Db2 接続機能の開始

CICS Db2 接続機能は、初期設定時に自動的に開始するか、手動で開始することができます。

このタスクについて

CICS 初期設定時の自動接続

以下のいずれかの方法で CICS を構成し、初期設定時に CICS と Db2 間の自動接続が確立されるようにすることができます。

- SIT に、または SIT の指定変更として DB2CONN=YES を指定する
- PLTPI の DFHDELIM ステートメントの後にプログラム DFHD2CM0 を指定する
- PLTPI の DFHDELIM ステートメントの後に、**EXEC CICS SET DB2CONN CONNECTED** コマンドを発行するユーザー作成プログラムを指定する

手動接続

以下のいずれかの方法で、CICS と Db2 間の接続を手動で確立することができます。

- **DSNC STRT** コマンドを使用する
- **CEMT SET DB2CONN CONNECTED** コマンドを使用する
- **EXEC CICS SET DB2CONN CONNECTED** コマンドを発行するユーザー・アプリケーションを実行する

CICS Db2 接続機能の停止

CICS Db2 接続機能 (以降、CICS-Db2 接続) は、静止停止または強制停止することができます。

このタスクについて

静止停止は、Db2 を現在使用しているすべての CICS トランザクションが完了するのを待ちます。

強制停止では、Db2 を現在使用しているすべての CICS トランザクションは強制的にページされます。

CICS 終了時の自動切断

CICS Db2 接続機能の開始中、CICS Db2 タスク関連ユーザー出口 (TRUE) は、SHUTDOWN オプションを指定して使用可能化されます。つまり、CICS がシャットダウンされると、CICS は自動的に TRUE を呼び出し、CICS Db2 接続をシャットダウンします。

このタスクについて

CICS DB2 TRUE は、CICS の静止シャットダウン中に呼び出されると接続機能の静止停止を開始します。同様に、CICS を即時シャットダウンすると、接続機能の強制シャットダウンが TRUE によって開始されます。CICS がキャンセルされると、接続機能のシャットダウンは TRUE によって開始されません。

手動の切断

いくつかの方法で、CICS を Db2 から切断することができます。

このタスクについて

CICS と Db2 との間の接続は、以下のいずれかの方法を使用して停止または切断できます。

- DSNB STOP コマンドを使用する。

DSNB STOP コマンドについては、52 ページの『DSNB STOP』を参照してください。

- CEMT SET DB2CONN NOTCONNECTED コマンドを使用する。
- EXEC CICS SET DB2CONN NOTCONNECTED コマンドを発行する、CICS 提供の CDBQ トランザクションを実行する。

CDBQ トランザクションは、プログラム DFHD2CM2 を実行します。このトランザクションは、端末から直接実行するか、EXEC CICS START コマンドを使用して実行できます。端末にはメッセージは出力されません。ただし、CICS Db2 アダプターは、シャットダウン・プロシーチャーの一部として一時データにメッセージを出力します。

- EXEC CICS SET DB2CONN NOTCONNECTED FORCE コマンドを発行する、CICS 提供の CDBF トランザクションを実行する。

CDBF トランザクションは、プログラム DFHD2CM3 を実行します。このトランザクションは、端末から直接実行するか、EXEC CICS START コマンドを使用して実行できます。端末にはメッセージは出力されません。ただし、CICS Db2 アダプターは、シャットダウン・プロシーチャーの一部として一時データにメッセージを出力します。

- EXEC CICS SET DB2CONN NOTCONNECTED コマンドを発行するユーザー・アプリケーションを実行する。

未確定作業単位 (UOW) の解決

未確定作業単位 (UOW) は、トランザクションが同期点処理を実行している時、つまり **EXEC CICS SYNCPOINT** コマンドまたは **EXEC CICS RETURN** コマンドの処理中に、CICS、Db2、またはシステム全体で障害が起こった時に発生します。未確定 UOW は、通常、CICS と Db2 との間の接続が再確立された時に解決されます。

複数のリカバリー可能リソースが関連している場合、CICS は、UOW のコミットを試行する際に 2 フェーズ・コミット・プロトコルを使用します。フェーズ 1 の PREPARE 呼び出しに「はい」と応答してから、フェーズ 2 時点の COMMIT 呼び出しまたは BACKOUT 呼び出しを受信するまで、Db2は、UOW の結果に関し

て未確定になります。この時に障害が発生すると、Db2 は、UOW に関して未確定のままとなります。Db2 は未確定の UOW を持ち、CICS に再同期を依頼する必要があります。

Db2 と単一の CICS システムのみが作業単位に関連している状態では、CICS は常にコーディネーターとなります。その他のシステムが関連している場合、例えば、LU6.2 通信リンクを用いている場合は、ローカル CICS システムではなく、リモート CICS システムが、作業単位の全体コーディネーターである可能性があります。この状態では、Db2 とローカル CICS システムの両方が、UOW の結果について未確定である可能性があります。この状態で障害が起こると、トランザクションの定義に応じて、ローカル CICS システムは作業単位を中断する可能性があります。その後、その作業単位は、未確定で中断されていると見なされます。

未確定 UOW は、通常、CICS と Db2 との間の接続が再確立された時に自動的に解決されます。CICS と Db2 は、未確定 UOW に関する情報を交換します。つまり、CICS は Db2 に対し、UOW がバックアウトされたか、またはコミットされたかを通知します。UOW が未確定で中断されている場合、この情報の交換は、リモート・コーディネーターが CICS に結果を通知するまで据え置かれます。ただし、グループ接続を使用している場合、CICS は、別の Db2 サブシステムに再接続し、前に接続された Db2 サブシステムによって保留されている未確定 UOW に関する情報を交換できない可能性があります。この問題を解決するためには、DB2CONN 定義の RESYNCMEMBER 属性が使用されます。

グループ接続を使用した未確定 UOW の解決

グループ接続を使用していて、CICS と Db2 との間の接続が切断された場合、CICS は、同じ Db2 サブシステムに再接続せず、Db2 サブシステムのデータ共有グループ内の別のメンバーを選択する可能性があります。つまり、CICS が接続した最初の Db2 サブシステムによって未確定 UOW が保留されている場合、それらは解決されません。

この問題を解決するために、CICS は、接続した最後の Db2 データ共有グループ・メンバーの履歴を維持します。その履歴は、カタログされ、ウォーム・スタート、緊急時再始動、およびコールド・スタートが行われても維持されます (ただし、初期始動は除きます)。Db2 への接続または再接続中、CICS Db2 接続機能は、この履歴を確認し、接続した最後の Db2 データ共有グループ・メンバーについて未解決 UOW 情報が保留されているかどうかをチェックし、以下のアクションを実行します。

- 未解決の UOW 情報が保留されておらず、CICS が緊急時再始動を実行していない場合、グループ接続は正常に作動し、接続に対してデータ共有グループの任意のアクティブ・メンバーを選択します。CICS が緊急時再始動を実行している場合、CICS は、最後に接続したメンバーに接続しようとします。
- 未解決 UOW 情報が保留されている場合、次のアクションは、DB2CONN 定義の RESYNCMEMBER 属性に対して選択した設定によって決まります。
 - RESYNCMEMBER が YES に設定されており、最後に記録された Db2 データ共有グループ・メンバーとの再同期が必要であることを示している場合、CICS はグループ接続機能を見捨て、CICS Db2 接続機能は、その最後に接続した Db2 データ共有グループ・メンバーに再接続できるまで待って、未確定の作業単位を解決します。未確定で中断されている UOW は、CICS 自体

がこの時点ではそれらの UOW を解決できないため、このプロセスには含まれません。それらの UOW の再同期は、CICS がそのリモート・コーディネーターと再同期したときに生じます。

- (おそらくできるだけ早く再接続したいという理由から) RESYNCMEMBER が NO に設定されている場合、CICS は、最後に記録された Db2 データ共用グループ・メンバーへの再接続を 1 回試行します。この試行が成功すると、未確定 UOW (未確定で中断している UOW を除く) を解決できます。成功しないと、CICS はグループ接続を使用して Db2 データ共用グループの任意のアクティブ・メンバーに接続します。すると、最後に記録されたメンバーに未解決の未確定 UOW がある可能性があることを示す警告メッセージ DFHDB2064 が発行されます。

RESYNCMEMBER の設定については、DB2CONN リソースを参照してください。RESYNCMEMBER オプションは、**SET DB2CONN RESYNCMEMBER** コマンドを使用しても設定できます。

Db2 restart-light を使用した未確定作業単位の解決

CICS は、Db2 で提供されている、拡張された DB2 バージョン 8 restart-light 機能をサポートしています。

restart-light モードは、CICS システムがグループ接続 (Db2 グループ接続機能の使用を参照) を使用するよう構成されている、MVS システム障害が発生した際のシステム間再始動を対象としています。DB2 バージョン 7 では、Db2 restart-light 機能により、Db2 データ共用メンバーは、最小のストレージ占有スペースで再始動し、保持されているロックを解放してから、正常に終了することができました。しかし、これが適用されるのは、未完了の作業単位だけであり、未確定の作業単位には適用されませんでした。DB2 バージョン 8 では、この機能は未確定作業単位を含むように拡張されました。これにより、CICS システムは、未確定作業単位を再同期する目的で、Db2 restart-light サブシステムに接続することが可能になります。Db2 ではピア・リカバリーがサポートされない、つまり、ある Db2 サブシステムが別の Db2 サブシステムの代わりに未確定作業単位を再同期することができないため、この機能は特に有用です。

典型的なシナリオでは、MVS LPAR で障害が起こると、システム間再始動を開始することができます。システム間再始動では、別の LPAR 上で、障害が発生した CICS システムが再始動され、その LPAR 上で、障害が発生した Db2 サブシステムが restart-light モードで起動されます。DB2CONN 定義の RESYNCMEMBER (YES) および STANDBYMODE(RECONNECT) を使用して、CICS システムがグループ接続を使用するように構成されており、Db2 内に未解決の未確定作業単位があると想定します。Db2 restart-light サブシステムは初期化し、未完了の UOW に対する保持ロックを解放し、未確定 UOW について CICS との再同期を待ちます。各 CICS システムが初期化し、未完了の作業単位があるかどうか、および RESYNCMEMBER(YES) が指定されているかどうかを検出します。これらの条件が両方とも TRUE の場合、CICS システムはグループ接続を無視し、Db2 restart-light サブシステムである、最後の Db2 サブシステムに再接続します。これで、未確定 UOW が再同期されましたが、新しいトランザクションは Db2 へのアクセスを許可されません。Db2 restart-light サブシステムですべての未確定 UOW が解決されると、Db2 restart-light サブシステムは終了します。CICS システム

は、STANDBYMODE(RECONNECT) を使用して定義されているため、Db2 restart-light サブシステムが終了すると待機モードに入り、Db2 への再接続を試行します。これで、すべての未確定作業単位が解決されたため、RESYNCMEMBER は適用されず、グループ接続を使用できます。CICS システムは、通常のアクティブな Db2 サブシステムに再接続します。

未確定 UOW の再同期情報のリカバリー

CICS は、再同期に必要な、UOW に関する情報をシステム・ログに維持しています。再同期情報は、ウォーム・スタート、緊急時再始動、およびコールド・スタートが行われても CICS によって維持されます。

CICS の初期始動が行われると、システム・ログが初期化され、すべての情報が消失し、前の作業単位に関するすべての情報が削除されるため、再同期情報は失われます。CICS の初期始動が必要になることはめったにありません。CSD からリソースを再インストールする場合は、コールド・スタートを使用してください。コールド・スタートでは、再同期情報はすべてリカバリー可能です。特に、前の CICS のウォーム・シャットダウンで、再同期が未完了であることを示すメッセージ DFHRM0131 が発行された場合は、CICS の初期始動を行わないようにしてください。

Db2 の再同期が必要な時に CICS が初期始動されると、CICS Db2 接続が再確立された時に、再同期に失敗した各 UOW に対してメッセージ DFHDB2001 が出力されます。そして、その UOW は、**DB2 RECOVER INDOUBT** コマンドを使用して Db2 で再同期する必要があります。

CICS Db2 接続機能の管理

CICS と Db2 との間の接続の状況は、CEMT コマンド、および CICS Db2 接続機能によって提供されるコマンドを使用して管理できます。CEMT によって提供される同じ機能は、EXEC CICS の INQUIRE コマンドおよび SET コマンドを介しても使用可能です。

このタスクについて

使用可能な機能の要約を以下に示します。

CEMT INQUIRE および SET DB2CONN

これらのコマンドを使用して、接続とその属性、およびスプール・スレッドとコマンド・スレッドの属性の全体的な状況を照会および設定することができます。CEMT INQUIRE DB2CONNおよびCEMT SET DB2CONNを参照してください。

CEMT INQUIRE および SET DB2ENTRY

これらのコマンドを使用して、特定のトランザクションまたはトランザクション・セットによって使用されるスレッドを定義する特定の DB2ENTRY の属性を照会および設定できます。CEMT INQUIRE DB2ENTRYおよびCEMT SET DB2ENTRYを参照してください。

CEMT INQUIRE および SET DB2TRAN

これらのコマンドを使用して、どのトランザクション ID がどの DB2ENTRY

を使用しているかを調査および変更します。CEMT INQUIRE DB2TRANおよび CEMT SET DB2TRANを参照してください。

DSNC DISC (disconnect)

この CICS Db2 接続コマンドを使用して、現在接続されているスレッドを、トランザクションに使用されなくなった時点ですぐに終了することができます。アクティブ・スレッドの場合、これは、トランザクションがそのスレッドを解放した時 (通常、同期点時) を意味します。保護スレッドは、現在トランザクションによって使用されておらず、通常、2 回の保護スレッドのページ・サイクルの後に再使用されていない場合に解放されるスレッドです。DSNC DISCONNECT コマンドは、ページ・サイクルをあらかじめ阻止し、スレッドを即時終了させます。DSNC DISCONNECT コマンドについて詳しくは、44 ページの『DSNC DISCONNECT』を参照してください。

DSNC DISP (display)

この CICS Db2 接続コマンドは、特定のプラン、特定のトランザクション、またはすべてのプランとトランザクションのアクティブ・スレッドの状況を表示するために使用できます。詳しくは、45 ページの『DSNC DISPLAY』を参照してください。

また、DSNC DISP コマンドは、CICS 端末に CICS Db2 統計を表示します。これらの統計は、CICS COLLECT STATISTICS コマンドと CICS PERFORM STATISTICS コマンドを使用して取得できる CICS Db2 統計のサブセットにすぎません。これらの統計は、すべての CICS 統計と同じリセット特性の影響を受けます。

DSNC DISP STAT コマンドの詳細については、48 ページの『DISPLAY STATISTICS 出力』を参照してください。使用可能な、完全な CICS Db2 統計について詳しくは、『Reference』の『CICS Db2 統計』を参照してください。

DSNC MODI (modify)

この CICS Db2 接続コマンドは、非送信請求メッセージの送信先、および DB2ENTRY で許可されるスレッドの数、あるいはプール・スレッドまたはコマンド・スレッドで許可されるスレッドの数を変更するために使用できます。この機能は、これらの属性、および DB2CONN、DB2ENTRY、または DB2TRAN のすべての属性の変更を可能にする、記載されている CEMT コマンドによって置き換えられます。詳しくは、50 ページの『DSNC MODIFY』を参照してください。

Db2 コマンドの入力

CICS と Db2 との間の接続が確立されたら、CICS によって許可される端末ユーザーは、DSNC トランザクションを使用してコマンドを Db2 サブシステムに経路指定できます。

このタスクについて

DSNC トランザクションを使用してコマンドを Db2 サブシステムに経路指定するコマンドは、以下のように定義されます。

DSNC -DB2command

コマンドは、処理のために Db2 にルーティングされます。Db2 は、入力されたコマンドを発行する権限がユーザーにあるかどうかをチェックします。応答は、送信側の CICS ユーザーにルーティングされます。Db2 コマンドを CICS Db2 接続機能のコマンドから区別するために、コマンド認識文字 (CRC) としてハイフン (-) を使用する必要があります。このコマンド認識文字は、コマンドの送信先である Db2 サブシステムを識別するためには使用されません。このコマンドは、CICS が現在接続されている Db2 サブシステムに送信されます。図 4 は、CICS Db2 接続機能コマンドを示しています。これらは、DSNC トランザクションおよび Db2 コマンドを使用するために CICS の許可が必要です。DSNC -DB2COMMAND について詳しくは、42 ページの『DSNC トランザクションの使用による Db2 へのコマンドの発行』を参照してください。

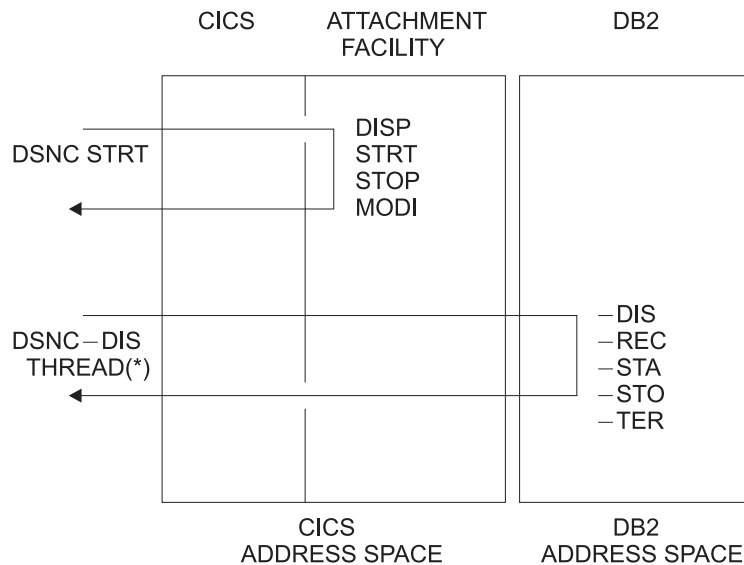


図 4. CICS Db2 接続機能コマンドといくつかの Db2 コマンドの例

Db2 アカウンティング、統計、および調整のための SMF の開始

Db2 は独自の計測機能を使用して、CICS Db2 スレッドが終了するたびに、および CICS Db2 サインオンのたびに、システム管理機能 (SMF) タイプ 101 アカウンティング・レコードを生成します。また、Db2 サブシステム・ベースで Db2 統計を保持するために Db2 は、SMF タイプ 100 レコードの生成も行います。パフォーマンスおよびグローバル・トレース・レコードは、SMF タイプ 102 レコードとして生成されます。

このタスクについて

Db2 によって生成される SMF レコードは、以下の方法で SMF データ・セットに送信することができます。

- Db2 のインストール時にアカウンティング・データまたは統計データ、あるいはその両方が必要であることを指定する。

これを行うために、初期設定マクロ DSN6SYSP の MON、SMFACCT または SMFSTAT が YES に設定されます。詳細については、Db2 11 for z/OS の資料を参照してください。

- レコードを作成するように SMF を活動化する。

タイプ 100、101、および 102 レコードの項目を SYS1.PARMLIB の既存の SMF メンバー (SMFPRMxx) 項目に追加します。

- Db2 トレースを開始する。

初期設定マクロ DSN6SYSP のパラメーターを設定するか、特定のトレースおよび開始するクラスを指定して Db2 -START TRACE コマンドを使用することで、Db2 トレースの開始を Db2 の起動時に行うことができます。後者の方法では、アカウントिंग・データ、統計データ、およびパフォーマンス・データの記録を周期的に開始できます。

アカウントING、モニター、またはパフォーマンスの目的で記録されたデータから Db2 によってレポートが生成されることはありません。独自のレポートを生成するには、このデータを処理するための独自のプログラムを作成するか、 Db2 Performance Monitor (DB2PM) プログラム製品を使用することができます。

Db2 アカウンティング、統計、および調整のための GTF の開始

アカウントING・データ、統計データ、およびパフォーマンス・データを SMF に記録する代わりに、または記録することに加えて、Db2 を使用してこのデータを汎用トレース機能 (GTF) に送ることができます。Db2 には、-START TRACE コマンドと -STOP TRACE コマンドがあります。これらは、DSNC トランザクションを使用して CICS 端末から発行することができます。

このタスクについて

以下のコマンドを DSNC トランザクションとともに使用して、トレースの有効範囲、トレースの宛先、およびトレースの制約を指定します。

トレースの有効範囲

Db2 サブシステム・アクティビティーの場合は GLOBAL、パフォーマンスの場合は PERF、アカウントINGの場合は ACCTG、統計の場合は STAT、またはモニター・アクティビティーの場合は MONITOR にこれを設定します。

トレースの宛先

GTF (主記憶域内)、または SMF。

トレースの制約

特定の計画、許可 ID、クラス、場所、およびリソース・マネージャー。

CICS Db2 用の CICS 提供トランザクション

CICS は、Db2 とのインターフェースを管理するために使用できるいくつかのトランザクションを提供しています。

CEMT

使用可能な CEMT オプションは次のとおりです。

- CEMT INQUIRE DB2CONN
- CEMT SET DB2CONN
- CEMT INQUIRE DB2ENTRY
- CEMT SET DB2ENTRY
- CEMT INQUIRE DB2TRAN
- CEMT SET DB2TRAN

これに相当する **EXEC CICS INQUIRE** および **EXEC CICS SET** コマンドについては、システム・コマンドで説明されています。

INQUIRE DB2TRAN コマンドでは PLAN オプションと PLANEXITNAME オプションを使用することができます。これにより、指定されたトランザクションまたはトランザクションのセットでどの計画が使用されるか、また指定された計画をどのトランザクションが使用するかが単一ステップで分かります。

DSNC

DSNC トランザクションを使用して、以下を実行できます。

- Db2 コマンドを CICS 端末から入力します。
- スレッドが解放されるときに終了するようにします (DSNC DISCONNECT)。
- CICS Db2 インターフェースを使用してトランザクションに関する情報を表示し、統計を表示します (DSNC DISPLAY)。
- 非送信請求メッセージの宛先を変更し、DB2ENTRY、プール、またはコマンドで使用されるアクティブ・スレッドの数を変更します (DSNC MODIFY)。
- CICS Db2 インターフェースをシャットダウンします (DSNC STOP)。
- CICS Db2 インターフェースを開始します (DSNC STRT)。

DSNC トランザクションはプログラム DFHD2CM1 を実行します。このプログラムは、CICS Db2 接続機能コマンドおよび Db2 コマンドの両方を処理します。Db2 コマンドを CICS Db2 接続機能コマンドと区別するために、Db2 コマンドではハイフン (-) 文字を入力します。この文字は Db2 サブシステム認識文字ではなく、コマンド認識文字です。CICS が接続できる Db2 サブシステムは一度に 1 つだけなので、これは Db2 サブシステムを定義する文字に関係なく常に - です。CICS から異なる Db2 サブシステム認識文字を使用する必要はなく、デフォルトの - 文字だけを使用します。

DFHD2CM1 を DSNC 以外のトランザクションによって活動化することもできます。このため、各 CICS Db2 接続機能コマンド、および各 Db2 コマンドに対してそれぞれ異なるトランザクション・コードを提供することができます。これにより、異なるレベルのセキュリティを各コマンドに適用することが可能になります。

サンプル・グループ **DFH\$DB2** で提供されている、**CICS Db2** の代替トランザクション定義

CICS Db2 接続機能コマンドの代替トランザクション定義は、以下の名前で提供されています。

- DISC
- DISP
- STRT
- STOP
- MODI

Db2 コマンドの代替トランザクション定義も、以下の名前で提供されています。

-ALT	-CAN	-ARC
-DIS	-MOD	-REC
-RES	-SET	-STA
-STO	-TER	

DSNC トランザクションの使用による **Db2** へのコマンドの発行

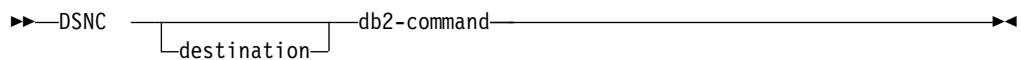
DSNC トランザクションを使用して、Db2 コマンドを CICS から入力することができます。

環境

このコマンドは、CICS 端末からのみ発行できます。

構文

DSNC 構文



省略語

関連トランザクション定義を CICS Db2 サンプル・グループ DFH\$DB2 からインストールする場合、Db2 コマンドで DSNC を省略することができます。例えば、-DIS トランザクション定義をインストールするとします。

DSNC -DIS THD(*)

これは、次のように省略できます。

-DIS THD(*)

サンプル CICS Db2 グループ DFH\$DB2 には、Db2 コマンドを発行するための以下のトランザクション定義が含まれています。

-ALT	-ARC	-CAN
-DIS	-MOD	-REC

-RES
-STO

-SET
-TER

-STA

Authorization

DSNC を使用して Db2 コマンドを発行する際、DSNC トランザクションの実行に必要なトランザクション接続セキュリティ以外に必要な CICS の許可はありません。ただし、Db2 特権が必要です。CICS セキュリティの詳細については、CICS DB2 環境でのセキュリティを参照してください。

パラメーターの説明

destination

表示情報を受け取る別の端末を識別します。この端末は CICS に対して定義され、基本マッピング・サポート (BMS) によってサポートされた有効な端末でなければなりません。

db2-command

CICS 端末から入力する Db2 コマンドを正確に指定します。先頭にハイフンを指定する必要があります。

使用上の注意

画面スクロール

BMS ページング・サポートを使用して、端末からの DSNC Db2 コマンドのスクロールをサポートできます。例えば、SIT キーワードを以下のように指定します。

PGCHAIN=X/,

PGCOPY=C/,

PGPURGE=T/,

PGRET=P/,

SKRPF7='P',

SKRPF8='N',

SIT キーワードとパラメーターについて詳しくは、システム初期設定パラメーターの説明と要約を参照してください。

例

Db2 コマンド -DISPLAY THREAD を CICS 端末から発行して、アプリケーション ID IYK4Z2G1 を持つ CICS のスレッドを表示します。

```
DSNC -DISPLAY THREAD(IYK4Z2G1)
```

```

DSNV401I : DISPLAY THREAD REPORT FOLLOWS -
DSNV402I : ACTIVE THREADS -
NAME      ST  A   REQ ID          AUTHID  PLAN      ASID  TOKEN
IYK4Z2G1  N           3             JTILLI1    00BA      0
IYK4Z2G1  T           3  ENTRXC060001  CICSUSER  TESTC06  00BA    16
IYK4Z2G1  T           3  POOLXP050002  CICSUSER  TESTP05  00BA    17
IYK4Z2G1  T  *        6  COMDDSN0003  JTILLI1    00BA    18
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I : DSNVDT '-DIS THREAD' NORMAL COMPLETION
DFHDB2301 07/09/98 13:36:36 IYK4Z2G1 DSN0 DB2 command complete.

```

図 5. DSN0 -DISPLAY コマンドの出力例

DSNC DISCONNECT

CICS Db2 接続機能コマンドの **DSNC DISCONNECT** を使用してスレッドを切断します。このコマンドを使用すると、通常のトランザクションによって共用されているリソースの解放を手動で制御できるため、Db2 ユーティリティなどの特殊目的プロセスはリソースに排他的にアクセスすることができます。

環境

このコマンドは、CICS 端末からのみ発行できます。

構文

DISC 構文

▶▶—DSNC DISConnect —plan-name————▶▶

省略語

DSNC DISC または DISC (CICS Db2 サンプル・グループ DFH\$DB2 の DISC トランザクションを使用)。

Authorization

このコマンドへのアクセスは、トランザクション DSN0 に対する CICS トランザクション接続セキュリティ検査、およびリソース DB2CONN に対する CICS コマンド・セキュリティ検査を使用して制御できます。このコマンドには READ アクセス権限が必要です。

CICS セキュリティの詳細については、CICS DB2 環境でのセキュリティを参照してください。

パラメーターの説明

plan-name

有効なアプリケーション・プランを指定します。

使用上の注意

スレッドの作成の防止

コマンド **DSNC DISCONNECT** は、トランザクションのためにスレッドが作成されることを防止しません。このコマンドは、現在接続されているスレッドがトランザクションによって使用されなくなった時点で、それらのスレッドをすぐに終了させるだけです。トランザクションを中断し、スレッドをより迅速に取り消すには、**Db2 CANCEL THREAD** コマンドを使用できます。

TRANCLASS の **MAXACTIVE** 設定を使用して、**CICS** 内で特定のプラン ID に関連付けられているトランザクションを停止できます。これにより、トランザクションの新しいインスタンスがスレッドを再作成することが防止されます。

保護されたスレッドの代替

再バインドを行うためや、データベースに対してユーティリティを実行するために、プランを割り振り解除したい場合があります。保護されたスレッドを使用している場合は、**EXEC CICS SET DB2ENTRY(entryname) THREADLIMIT(0)** を使用します。または、**DSNC DISCONNECT** ではなく、**DSNC MODIFY** を使用して、すべてのスレッドをプールに送信します。保護されたスレッドは、2 回のページ・サイクル中に自然に終了します。**DB2CONN** の **PURGE CYCLE** 属性を参照してください。

例

以下のようにして、プラン **TESTP05** のアクティブおよび保護されたスレッドを切断します。

```
DSNC DISC TESTP05
```

```
DFHDB2021 07/09/98 13:46:29 IYK4Z2G1 The disconnect command is complete.
```

図 6. **DSNC -DISCONNECT** コマンドの出力例

DSNC DISPLAY

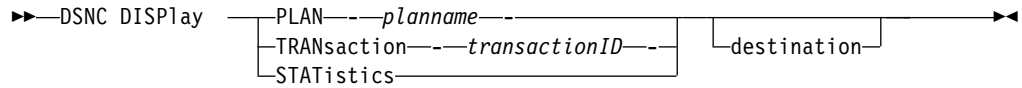
CICS Db2 接続機能コマンド DSNC DISPLAY を使用して、アクティブな **CICS Db2** スレッドとそれを使用している **CICS** トランザクションに関する情報を表示したり、**DB2ENTRY** リソースや **DB2CONN** リソースに関連した統計情報を表示したりすることができます。

環境

このコマンドは、**CICS** 端末からのみ発行できます。

構文

DISPLAY 構文



省略語

DSNC DISP または DISP (CICS Db2 サンプル・グループ DFH\$DB2 の DISP トランザクションを使用)。

Authorization

このコマンドへのアクセスは、トランザクション DSNC に対する CICS トランザクション接続セキュリティ検査、およびリソース DB2CONN に対する CICS コマンド・セキュリティ検査を使用して制御できます。このコマンドには READ アクセス権限が必要です。

CICS セキュリティの詳細については、CICS DB2 環境でのセキュリティを参照してください。

パラメーターの説明

PLAN *planname*

planname のスレッドに関する情報を表示します。*planname* は、情報の表示対象である有効なプラン名です。

planname を指定しない場合 (またはアスタリスク * を指定した場合) は、すべてのアクティブ・スレッドの情報が表示されます。

例えば、次のコマンドは、アクティブなすべての CICS Db2 プラン ID に関する情報を表示し、表示情報を MTO2 として指定された別の端末に送信します。

```
DSNC DISP PLAN * MTO2
```

TRAN *transactionID*

スレッド情報の表示対象であるトランザクション ID が含まれます。

transactionID は、スレッド情報が表示される有効なトランザクション ID です。

トランザクション ID を指定しない場合、すべてのアクティブ・スレッドの情報が表示されます。

例えば、次のコマンドは、アクティブなすべての CICS Db2 トランザクションに関する情報を表示します。

```
DSNC DISP TRAN
```

STAT

それぞれの CICS Db2 リソース定義に関連付けられた統計カウンターを表示します。これらのカウンターは、Db2 に対する、CICS Db2 接続機能の使用可能接続の使用状況に関するものです。

コマンド **EXEC CICS COLLECT STATISTICS** および **EXEC CICS PERFORM STATISTICS** などの標準の CICS 統計インターフェースを使用したり、DFH0STAT サンプル・プログラムを使用したりして、より詳細な CICS Db2 統計セットを取得できることに留意してください。

destination

要求された情報を受信する別の端末を指定します。*destination* は、要求された表示情報を受信する別の端末の ID です。この端末は CICS に対して定義され、基本マッピング・サポート (BMS) によってサポートされた有効な端末でなければなりません。

コマンド内でオプション宛先の前には、オプション・プラン名またはトランザクション ID が指定されていることがあるため、各パラメーターは、名前または端末 ID のいずれかとして、固有であり、別々に識別可能である必要があります。1 つのパラメーターのみが入力されている場合は、最初に、それがプラン名かトランザクション ID かを確認するための検査が行われ、その後に宛先として検査されます。プラン名またはトランザクション ID であり、また有効な端末 ID でもある文字ストリングを使用するには、名前パラメーターと宛先パラメーターの両方を使用して、必要な端末に必要な情報を表示する必要があります。

要求された表示情報を受信するための代替宛先が指定されると、以下のメッセージが要求端末に送信されます。

```
DFHDB2032 date time applid alternate destination display command complete
```

使用上の注意

CICS Db2 接続機能はアクティブだが、Db2 サブシステムはアクティブでない間に、CICS からこのコマンドを発行すると、サブシステムが作動可能でないことの明確な標識なしに統計表示が作成されます。

CICS メッセージ・ログにメッセージ DFHDB2037 が表示され、接続機能が Db2 の開始を待っていることを示します。

DISPLAY PLAN または TRAN の出力

DSNC DISPLAY PLAN コマンドまたは **DSNC DISPLAY TRANSACTION** コマンドの出力には、DB2ENTRY 名、*POOL、または *COMMAND (DSNC コマンド呼び出しの場合) が含まれます。

48 ページの図 7 は、**DSNC DISPLAY** (PLAN または TRANSACTION) コマンドの出力例を示しています。作成された各スレッドについて、出力は、DSNC コマンド呼び出しの DB2ENTRY の名前、またはプールの場合は *POOL、あるいは *COMMAND を表示します。

```

DFHDB2013 07/09/98 15:26:47 IYK4Z2G1 Display report follows for threads
accessing DB2 DB3A
DB2ENTRY S PLAN PRI-AUTH SEC-AUTH CORRELATION TRAN TASK UOW-ID
*POOL A TESTC05 JTILLI1 POOLXC050001 XC05 01208 AEEEC0321ACDCE00
XC06 * TESTC06 JTILLI1 ENTRXC060003 XC06 01215 AEEEC0432F8EFE01
XP05 A TESTP05 JTILLI1 ENTRXP050002 XP05 01209 AEEEC03835230C00
XP05 I TESTP05 JTILLI1 ENTRXP050004
DFHDB2020 07/09/98 15:26:47 IYK4Z2G1 The display command is complete.

```

図 7. *DSNC DISPLAY (PLAN または TRANSACTION)* コマンドの出力例

「S」という名前の列は、スレッドの状況を示しており、以下の値を取ることができます。

- * スレッドは、作業単位内でアクティブで、現在 Db2 で実行中です。
- A スレッドは、作業単位内でアクティブですが、現在 Db2 で実行されていません。
- I スレッドは非アクティブです。このスレッドは、新しい作業を待機している保護されたスレッドです。

スレッドに関連付けられた PLAN が表示されます (コマンド・スレッドにはプランはありません)。

「PRI-AUTH」フィールドには、スレッドにより使用された 1 次許可 ID が表示されます。「SEC-AUTH」フィールドには、そのスレッドの 2 次許可 ID (存在する場合) が表示されます。

「CORRELATION」フィールドには、12 バイトのスレッド相関 ID が表示されます。これは、*eeee¹tttt²nnnn* の形式になっています。*eeee* は COMD、POOL、ENTR のいずれかで、それぞれ、コマンド・スレッド、プール・スレッド、DB2ENTRY スレッドであることを示しています。*tttt* はスレッド ID、*nnnn* は固有の番号です。

注: Db2 に渡される相関 ID は、CICS 接続機能が Db2 にサインオンを発行することによってのみ変更されます。複数のトランザクションにわたって一定の間隔である 1 次許可 ID を使用して、スレッドによってサインオンの再使用が行われた場合 (例えば、AUTHID(name) を使用して)、1 つだけのサインオンが発生します。このインスタンスで、相関 ID 内の *tttt* は、実行中のトランザクション ID と一致しません。これは、最初のサインオンが発生したときのトランザクションの ID です。

スレッドが作業単位内でアクティブである場合、CICS トランザクション名とそのタスク番号が表示され、その後に CICS ローカル作業単位 ID が表示されます。

この表示で使用する相関 ID は、DISPLAY LOCK などの Db2 コマンドでも出力されます。例えば、この表示を、ロックを表示するコマンドと一緒に使用すると、どの CICS タスクが Db2 内でロックを保持しているかがわかります。

DISPLAY STATISTICS 出力

このトピックでは、**DSNC DISPLAY STATISTICS** コマンドの出力について説明します。

```

DFHDB2014 07/09/98 14:35:45 IYK4Z2G1 Statistics report follows for RCTJT
accessing DB2 DB3A

```

DB2ENTRY	PLAN	CALLS	AUTHS	W/P	HIGH	ABORTS	-----COMMITES-----	
							1-PHASE	2-PHASE
*COMMAND		1	1	1	1	0	0	0
*POOL	*****	4	1	0	1	0	2	0
XC05	TESTP05	22	1	11	2	0	7	5
XP05	*****	5	2	0	1	0	1	1

```

DFHDB2020 01/17/98 15:45:27 IYKA4Z2G1 The display command is complete.

```

図 8. *DSNC DISPLAY STATISTICS* コマンドの出力例

DB2ENTRY

DB2ENTRY の名前、または DSNC コマンド呼び出しの場合は「*COMMAND」、またはプール統計の場合は「*POOL」。

PLAN

この項目に関連付けられているプラン名。このフィールドに 8 個のアスタリスクが表示されている場合は、該当するトランザクションが動的プラン割り振りを使用していることを示します。コマンド・プロセッサ・トランザクション DSNC には関連付けられているプランはありません。

項目に関連付けられているプラン名が動的に変更される場合、最後のプラン名が、使用されるプラン名になります。

CALLS

この項目に関連付けられているトランザクションによって発行された SQL ステートメントの合計数。

AUTHS

この項目に関連付けられているトランザクションのサインオン呼び出しの合計数。サインオンは、新しいスレッドが作成されるか、または既存のスレッドが再使用されるかを示しません。スレッドが再利用される場合、サインオンが行われるのは、許可 ID またはトランザクション ID が変更された場合のみになります。

W/P

この項目のすべての使用可能スレッドが使用中だった回数です。この値は、この項目の THREADWAIT の値によって異なります。THREADWAIT が POOL に設定されている場合、W/P は、トランザクションがプールにオーバーフローした回数を示します。プールへのオーバーフローは、個別の DB2ENTRY の統計にのみ現れ、プール統計には反映されません。

THREADWAIT が YES に設定されている場合、これは、(アクティブ・スレッドの数が THREADLIMIT に達したために) トランザクションがスレッドを待機しなければならなかった回数、または (スレッドを実行している、使用中の TCB の数が TCBLIMIT に達したために) トランザクションが新しいスレッド TCB を使用できなかった回数のいずれかを反映します。

HIGH

接続が開始されて以降の任意の時点で、この項目に関連付けられているトランザクションによって獲得されたスレッドの最大数。つまり、スレッドの最高水準点です。

注: CICS Transaction Server for OS/390[®] バージョン 1 リリース 2 より前の CICS のリリースでは、HIGH 値には、スレッドを強制的に待機させられたトランザクション、またはプールに方向転換されたトランザクションも含まれていました。CICS Transaction Server for OS/390 バージョン 1 リリース 2 以降、HIGH 値は、項目で作成されたスレッドのみを表すようになりました。

ABORTS

ロールバックされたリカバリー単位の合計数。これには、異常終了と同期点ロールバック (-911 SQL コードによって生成された同期点ロールバックを含む) の両方が含まれます。

COMMITTS

この項目に関連付けられた Db2 トランザクションが実同期点または暗黙の (EOT などの) 同期点に達するたびに、次の 2 つのフィールドの一方の値が増加します。SQL 呼び出しを処理しないリカバリー単位は、この数には反映されません。

1-PHASE

この項目に関連付けられているトランザクションの単一フェーズ・コミットの合計数。この合計数には、2 フェーズ・コミットは含まれません (2-PHASE の説明を参照)。この合計数には、読み取り専用コミット、および更新を実行したリカバリー単位の単一フェーズ・コミットも含まれます。2 フェーズ・コミットは、アプリケーションが DB2 以外のリカバリー可能リソースを更新した場合にのみ必要です。

2-PHASE

この項目に関連付けられているトランザクションの 2 フェーズ・コミットの合計数。この数には、単一フェーズ・コミット・トランザクションは含まれません。

DSNC MODIFY

CICS Db2 接続機能コマンドの DSNC MODIFY を使用して、DB2CONN のメッセージ・キュー宛先、プールの THREADLIMIT 値、DSNC コマンド、または DB2ENTRY を変更します。これらの変更は、CEMT あるいは EXEC CICS の SET DB2CONN コマンドまたは SET DB2ENTRY コマンドを使用しても行うことができます。

環境

このコマンドは、CICS 端末からのみ発行できます。

構文

MODIFY 構文

➡—DSNC MODIFY—┬—DESTination—old—new—┬—
└—TRANsaction—transaction-id—integer—┘

省略語

DSNC MODI または MODI (CICS Db2 サンプル・グループ DFH\$DB2 の MODI トランザクションを使用)。

Authorization

このコマンドへのアクセスは、以下の CICS 許可検査を使用して制御できます。

- トランザクション DSNB に対するトランザクション接続セキュリティ。
- リソース DB2CONN に対するコマンド・セキュリティ。このコマンドには、UPDATE アクセス権限が必要です。
- DB2ENTRY の属性を変更する DSNB MODIFY TRANSACTION コマンドの場合。リソース DB2ENTRY と DB2TRAN に対するコマンド・セキュリティ検査、および DB2ENTRY に対するリソース・セキュリティもあります。このコマンドには、コマンド・セキュリティのために、リソース DB2TRAN への READ アクセス権限およびリソース DB2ENTRY への UPDATE アクセス権限が必要です。さらに、リソース・セキュリティ・コマンドには、関連する特定の DB2ENTRY への UPDATE アクセス権限も必要です。CICS セキュリティの詳細については、CICS DB2 環境でのセキュリティを参照してください。

パラメーターの説明

DESTination

DB2CONN テーブルの MSGQUEUE パラメーターが変更され、「old」の宛先 ID が「new」の宛先 ID に置換されることを指定します。

old DB2CONN の MSGQUEUE に現在設定されている宛先 ID。

new 新しい宛先 ID。

TRANsaction *transaction-ID integer*

指定されたトランザクションまたはグループに関連付けられた THREADLIMIT 値を変更することを指定します。このコマンドは、トランザクション ID を使用して、変更するプール、コマンド、または DB2ENTRY の THREADLIMIT 値を識別します。

- プールの THREADLIMIT 値を変更するには、CEPL のトランザクション ID を使用する必要があります。
- コマンド・スレッドの THREADLIMIT 値を変更するには、DSNB のトランザクション ID を使用する必要があります。
- DB2ENTRY の THREADLIMIT 値を変更するには、その DB2ENTRY を使用するよう定義されているトランザクションのトランザクション ID を使用します。

integer は、新しい最大値です。

使用上の注意

コマンド DSNB MODIFY TRANSACTION に指定される整数は、DB2CONN の TCBLIMIT パラメーターに指定される値より大きくすることはできません。指定可能な最低値はゼロです。

例

例 1 DB2CONN の MSGQUEUE パラメーターの仕様を MTO1 から MTO2 に変更するには、以下のようにします。

```
DSNB MODI DEST MT01 MT02
```

```
DFHDB2039 07/09/98 14:47:17 IYK4Z2G1 The error destinations are: MT02 ****
****.
```

図 9. *DSNC MODIFY DESTINATION* コマンドの出力例

例 2 プール・スレッドの制限を 12 に変更するには、以下のようにします。

```
DSNC MODI TRAN CEPL 12
```

```
DFHDB2019 07/09/98 14:49:28 IYK4Z2G1 The modify command is complete.
```

図 10. *DSNC MODIFY TRANSACTION* コマンド (プール・スレッド) の出力例

例 3 コマンド・スレッドの制限を 3 に変更するには、以下のようにします。

```
DSNC MODI TRAN DSNC 3
```

```
DFHDB2019 07/09/98 14:49:28 IYK4Z2G1 The modify command is complete.
```

図 11. *DSNC MODIFY TRANSACTION* コマンド (コマンド・スレッド) の出力例

例 4 トランザクション XP05 によって使用される DB2ENTRY のスレッドの制限を 8 に変更するには、以下のようにします。

```
DSNC MODI TRAN XP05 8
```

```
DFHDB2019 07/09/98 14:49:28 IYK4Z2G1 The modify command is complete.
```

図 12. *DSNC MODIFY TRANSACTION* コマンド (DB2ENTRY スレッド制限の変更) の出力例

DSNC STOP

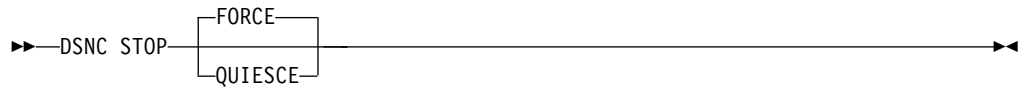
CICS Db2 接続機能コマンドの **DSNC STOP** を使用して接続機能を停止します。代わりに、**CEMT** または **EXEC CICS SET DB2CONN NOTCONNECTED** コマンドを発行して接続機能を停止することもできます。

環境

このコマンドは、CICS 端末からのみ発行できます。

構文

STOP 構文



省略語

DSNC STOP または STOP (CICS Db2 サンプル・グループ DFH\$DB2 の STOP トランザクションを使用)。

Authorization

このコマンドへのアクセスは、トランザクション DSNC に対する CICS トランザクション接続セキュリティ検査、およびリソース DB2CONN に対する CICS コマンド・セキュリティ検査を使用して制御できます。このコマンドには、UPDATE アクセス権限が必要です。

このコマンドには、UPDATE アクセス権限が必要です。CICS セキュリティの詳細については、CICS DB2 環境でのセキュリティを参照してください。

パラメーターの説明

QUIESCE

現在実行中の CICS トランザクションが完了した後、CICS Db2 接続機能を停止することを指定します。QUIESCE は、すべてのアクティブ・トランザクションが完了するのを待機するため、新しい UOW はスレッドを開始および獲得することができます。

FORCE

実行中のトランザクションに関係なく、Db2 との接続を強制的に切断して CICS Db2 接続機能を即時停止することを指定します。Db2 にアクセスしている、現在実行中のトランザクションは強制的にパージされます。これには、前の UOW で Db2 に更新をコミットしているが、現行の UOW ではまだ Db2 にアクセスしていない可能性があるトランザクションが含まれます。

使用上の注意

DSNC STOP QUIESCE の場合、メッセージ DFHDB2012 が端末に出力されます。その後、端末は、メッセージ DFHDB2025 が出力され、シャットダウンが完了するまでロックされたままになります。

DSNC STOP FORCE の場合、メッセージ DFHDB2022 が端末に出力されます。その後、端末は、メッセージ DFHDB2025 が出力され、シャットダウンが完了するまでロックされたままになります。

例

例 1 CICS Db2 接続機能を静止停止するには、以下のようにします。

DSNC STOP

```
DFHDB2012 07/09/98 14:54:28 IYK4Z2G1 Stop quiesce of the CICS-DB2 attachment
facility from DB2 subsystem DB3A is proceeding.
```

図 13. *DSNC STOP* コマンドの出力例

図 13 に示されている、*DSNC STOP* コマンドの結果として生じたメッセージは、シャットダウンが完了した時に、図 14 に示されているメッセージに置き換えられます。

```
DFHDB2025I 07/09/98 14:58:53 IYK4Z2G1 The CICS-DB2 attachment has disconnected
from DB2 subsystem DB3A
```

図 14. シャットダウンが完了した時の *DSNC STOP* の出力例

例 2 CICS Db2 接続機能を強制停止するには、以下のようになります。

DSNC STOP FORCE

```
DFHDB2022 07/09/98 15:01:51 IYK4Z2G1 Stop force of the CICS-DB2 attachment
facility from DF2D is proceeding.
```

図 15. *DSNC STOP FORCE* コマンドの出力例

図 15 に示されている、*DSNC STOP FORCE* コマンドの結果として生じたメッセージは、シャットダウンが完了した時に、図 16 に示されているメッセージに置き換えられます。

```
DFHDB2025I 07/09/98 15:10:55 IYK4Z2G1 The CICS-DB2 attachment has disconnected
from DB2 subsystem DF2D group DFP2
```

図 16. シャットダウンが完了した時の *DSNC STOP FORCE* の出力例

この例では、グループ接続が使用されたため、Db2 サブシステムの名前とそのグループの名前が表示されています。

DSNC STRT

DSNC STRT コマンドを使用して、CICS Db2 接続機能を開始します。CICS Db2 接続機能は、CICS アプリケーション・プログラムが Db2 データベースにアクセスすることを許可します。代わりに、**CEMT** または **EXEC CICS SET DB2CONN CONNECTED** コマンドを発行して、CICS アプリケーション・プログラムが Db2 データベースにアクセスすることを許可します。

環境

このコマンドは、CICS 端末からのみ発行できます。

構文

STRT 構文

→—DSNC STRT —→
 └──ssid┐

省略語

DSNC STRT または STRT (CICS Db2 サンプル・グループ DFH\$DB2 の STRT トランザクションを使用)。

Authorization

このコマンドへのアクセスは、トランザクション DSNC に対する CICS トランザクション接続セキュリティー検査、およびリソース DB2CONN に対する CICS コマンド・セキュリティー検査を使用して制御できます。このコマンドには、UPDATE アクセス権限が必要です。

CICS セキュリティーの詳細については、CICS DB2 環境でのセキュリティーを参照してください。

パラメーターの説明

ssid

DB2CONN に指定されている Db2 サブシステム ID (DB2ID) または Db2 データ共用グループ ID (DB2GROUPID) を指定変更するための Db2 サブシステム ID を指定します。DSNC STRT コマンドには Db2 データ共用グループ ID は指定できません。

使用上の注意

DSNC STRT コマンドが発行された時に DB2CONN がインストールされていない場合、DB2CONN がインストールされていないことを示すエラー・メッセージ DFHDB2031 が生成されます。CICS Db2 接続機能の開始を試行する前に、リソース定義を CSD からインストールしておく必要があります。

DSNC STRT コマンドを発行し、Db2 サブシステム ID を指定すると、インストールされている DB2CONN 定義内の DB2GROUPID がブランクになります。その後、グループ接続を使用するには、(CEDA INSTALL コマンドまたは SET DB2CONN コマンドを使用して) DB2GROUPID を再び設定する必要があります。

使用する Db2 サブシステムを判別するための階層は以下のようになります。

1. DSNC STRT コマンドに指定されている場合、サブシステム ID を使用します。
2. インストールされている DB2CONN 内の DB2ID がブランクでない場合、その DB2ID を使用します。
3. グループ接続には、インストールされている DB2CONN の DB2GROUPID を使用します。
4. 最後にインストールされた DB2CONN 内の DB2ID と DB2GROUPID がブランクの時 (または後でブランクに設定されている時) は、INITPARM にサブシ

システム ID が指定されている場合、そのサブシステム ID を使用します。最後にインストールされた DB2CONN にブランクの DB2ID およびブランクの DB2GROUPID が含まれていた場合は、その後に DB2ID または DB2GROUPID が SET コマンドを使用して変更された場合でも、開始時には、常に INITPARM が使用されます。

5. デフォルトのサブシステム ID である DSN を使用します。

例

- 例 1 Db2 サブシステム ID (DB2ID) または Db2 データ共有グループ ID (DB2GROUPID) を使用して、インストールされている DB2CONN から CICS Db2 接続機能を開始するには、以下のようになります。

DSNC STRT

この例では、グループ接続が使用されているため、Db2 サブシステムの名前およびそのグループの名前が表示されています。

```
DFHDB2023I 07/09/98 15:06:07 IYK4Z2G1 The CICS DB2 attachment has connected to
DB2 subsystem DF2D group DFP2
```

図 17. DSNC STRT コマンドの出力例

- 例 2 インストールされている DB2CONN を使用するが、DB2CONN 内の Db2 サブシステム ID (DB2ID) または Db2 データ共有グループ ID (DB2GROUPID) を Db2 サブシステム ID DB3A に置き換えて CICS Db2 接続機能を開始するには、以下のようになります。

DSNC STRT DB3A

```
DFHDB2023I 07/09/97 15:06:07 IYK4Z2G1 The CICS DB2 attachment has connected to
DB2 subsystem DB3A
```

図 18. DB3A を使用した DSNC STRT の出力例

グループ接続を使用しておらず、CICS Db2 接続機能の開始を試行した時に Db2 サブシステムがアクティブでない場合は、STANDBYMODE=NOCONNECT が DB2CONN に指定されている場合、以下の出力を受け取ります。

```
DFHDB2018 07/09/98 15:14:10 IYK4Z2G1 DB3A DB2 subsystem is not active.
```

図 19. Db2 がアクティブでなく、STANDBYMODE=NOCONNECT の時の DSNC STRT の出力例

STANDBYMODE=CONNECT または RECONNECT の場合、以下の出力を受け取ります。

```
DFHDB2037 07/09/98 15:15:42 IYK4Z2G1 DB2 subsystem DB3A is not active.  
The CICS DB2 attachment facility is waiting.
```

図 20. Db2 がアクティブでなく、STANDBYMODE=CONNECT または RECONNECT の時の DSNC STRT の出力例

グループ接続を使用しており、CICS Db2 接続機能の開始を試行した時にデータ共有グループ内にアクティブな Db2 サブシステムがない場合は、STANDBYMODE=NOCONNECT が DB2CONN に指定されている場合、以下の出力を受け取ります。

```
DFHDB2037 07/09/01 12:30:10 IYK2ZFV1 DB2 group DFP2 has no active members.
```

図 21. データ共有グループ内にアクティブな Db2 サブシステムがなく、STANDBYMODE=NOCONNECT の時の、グループ接続での DSNC STRT の出力例

STANDBYMODE=CONNECT または RECONNECT の場合、以下の出力を受け取ります。

```
DFHDB2037 07/09/01 12:55:00 IYK2ZFV1 DB2 group DFP2 has no active members.  
The CICS DB2 attachment facility is waiting.
```

図 22. データ共有グループ内にアクティブな Db2 サブシステムがなく、STANDBYMODE=CONNECT または RECONNECT の時の、グループ接続での DSNC STRT の出力例

第 4 章 Db2 のセキュリティー

CICS Db2 環境には、セキュリティー検査を実行できる 4 つの主なステージがあります。

それら 4 つのステージは以下のとおりです。

- CICS ユーザーが CICS 領域にサインオンするとき。CICS サインオンでは、ユーザーが有効なユーザー ID とパスワードを指定したことを確認することで、そのユーザーを認証します。
- CICS ユーザーが Db2 に関連している CICS リソースを使用または変更しようとしているとき。このリソースは、以下のものである可能性があります：
DB2CONN、DB2ENTRY、または DB2TRAN のリソース定義。あるいは、データを取得するために Db2 にアクセスする CICS トランザクション。あるいは、CICS Db2 接続機能または Db2 自体にコマンドを発行する CICS トランザクション。このステージでは、RACF[®] または同等の外部セキュリティー・マネージャーによって管理される CICS セキュリティー・メカニズムを使用して、リソースへの CICS ユーザーのアクセスを制御できます。
- CICS 領域を Db2 に接続するとき、およびトランザクションが Db2 に送信されたスレッドを取得するとき。CICS 領域とトランザクションの両方で Db2 に対する許可 ID を提供する必要があり、これらの許可 ID は RACF または同等の外部セキュリティー・マネージャーによって検証されます。
- CICS ユーザーが CICS トランザクションを使用して Db2 リソースを実行または変更しようとしているとき。これは、計画、Db2 コマンド、または動的 SQL を実行するために必要なリソースである場合があります。このステージでは、Db2 自体、あるいは RACF または同等の外部セキュリティー・マネージャーによって管理される Db2 のセキュリティー検査を使用して、リソースへの CICS ユーザーのアクセスを制御できます。

また、RACF、あるいは同等の外部セキュリティー・マネージャーを使用して、CICS と Db2 を構成するコンポーネントを無許可アクセスから保護することもできます。この保護は、Db2 のデータベース、ログ、ブートストラップ・データ・セット (BSDS)、Db2 の範囲外のライブラリー、および CICS のデータ・セットとライブラリーに適用できます。RACF で提供される保護を部分的に置き換えるものとして、VSAM パスワード保護を使用することもできます。詳しくは、CICS システム・リソースのセキュリティーを参照してください。

注: ここでは、RACF は CICS で使用される外部セキュリティー・マネージャーとして言及されています。明示的な RACF の例を除き、一般的な説明が、機能的に同等のすべての非 IBM 外部セキュリティー・マネージャーに同様に適用されます。

60 ページの図 23 は、CICS Db2 環境で使用されるセキュリティー・メカニズムを示しています。

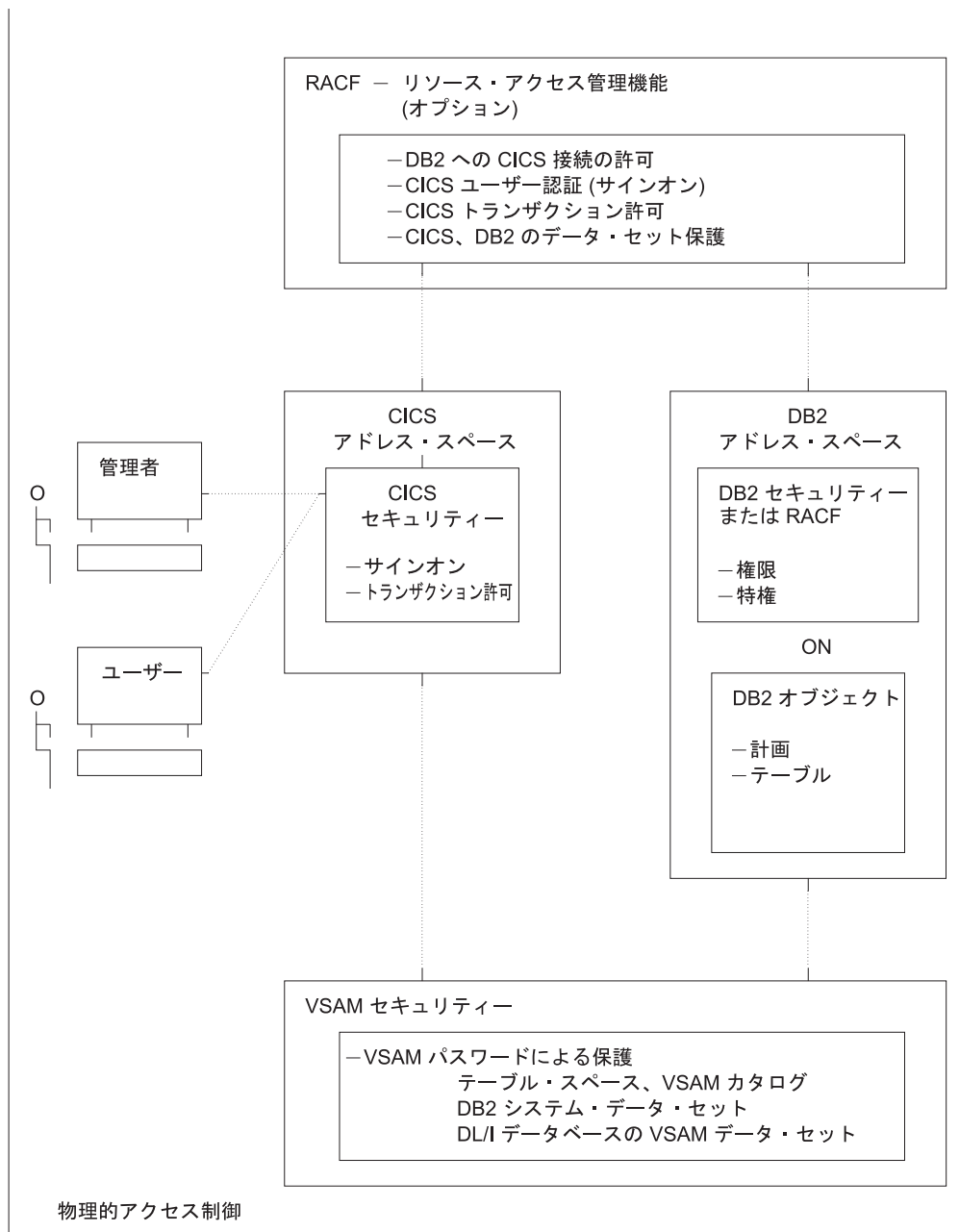


図 23. CICS Db2 セキュリティー・メカニズムの概要

CICS での Db2 関連リソースへのアクセスの制御

外部セキュリティ・マネージャーおよび適切な CICS セキュリティー・メカニズムを有効にすることによって、CICS 領域内の Db2 リソースへのアクセスを制御し、リソースのセキュリティ検査を開始することができます。

このタスクについて

CICS ユーザーは、Db2 に関係する以下のアクティビティーを実行することが必要になる場合があります。

- DB2CONN、DB2ENTRY、および DB2TRAN の各リソース定義の照会、変更、作成、または破棄。
- Db2 にアクセスするトランザクションを使用したデータの取得、または DSNB トランザクションの使用による CICS Db2 接続機能コマンドまたは Db2 コマンドの発行。

RACF または同等の外部セキュリティ・マネージャーを使用して、CICS 領域でセキュリティ検査を実行します。ユーザーが保護リソースにアクセスしようとする、CICS は外部セキュリティ・マネージャーを呼び出してセキュリティ検査を実行します。RACF は、ユーザーが CICS にサインオンするときに認証される、CICS ユーザーの ID を使用してセキュリティ検査を行います。ユーザー ID が事前設定セキュリティによって端末に永続的に関連付けられている場合を除き、ユーザーが CICS にサインオンしない場合は、デフォルト・ユーザー ID が付与されます。

CICS 領域に対する適切なセキュリティ・メカニズム (トランザクション接続セキュリティ、リソース・セキュリティ、コマンド・セキュリティ、または代理セキュリティ) を有効にします。

CICS セキュリティの詳細については、保護の概要を参照してください。

DB2CONN、DB2TRAN、および DB2ENTRY リソース定義に対するユーザー・アクセスの制御

さまざまな CICS セキュリティ・メカニズムを有効にすることで、DB2CONN、DB2TRAN、および DB2ENTRY の各リソース定義に対するユーザーのアクセスを制御できます。

このタスクについて

以下のセキュリティ・メカニズムを使用して、Db2 リソース定義に対するユーザー・アクセスを制御できます。

- CICS リソース・セキュリティ・メカニズムを使用して、特定のリソースにユーザーがアクセスできるかを制御します。リソース・セキュリティはトランザクション・レベルで実装されます。例えば、一部のユーザーに、特定の DB2ENTRY 定義を変更させないようにすることができます。このセキュリティ・メカニズムの使用法については、62 ページの『リソース・セキュリティを使用して、DB2ENTRY リソース定義および DB2TRAN リソース定義へのアクセスを制御する』の説明を参照してください。
- CICS コマンド・セキュリティ・メカニズムを使用して、Db2 関連リソースに対してユーザーが特定の SPI コマンドを発行できるかを制御します。コマンド・セキュリティもトランザクション・レベルで実装されます。例えば、特定のユーザーのみに、DB2ENTRY リソース定義に対して CREATE コマンドと DISCARD コマンドを発行することを許可できます。このセキュリティ・メカニズムの使用法については、64 ページの『コマンド・セキュリティを使用して、DB2CONN、DB2ENTRY、および DB2TRAN の各リソース定義に対する SPI コマンドの発行を制御する』の説明を参照してください。
- CICS 代理セキュリティおよび AUTHTYPE セキュリティのメカニズムを使用して、CICS が Db2 に提供する許可 ID をユーザーが変更できるかを制御し

ます。許可 ID は、Db2 のセキュリティ検査に使用され、Db2 関連リソース定義の AUTHID、COMAUTHID、AUTHTYPE および COMAUTHTYPE の各属性と、CICS 領域の DB2CONN 定義の SIGNID 属性によって設定されます。CICS は、許可 ID を変更を求めるユーザーが、リソース定義で指定された既存の許可 ID に代わる役割を果たすことが許可されるかどうかを検査します。これらのセキュリティ・メカニズムの使用法については、66 ページの『代理セキュリティおよび AUTHTYPE セキュリティを使用して、CICS が DB2 に提供する許可 ID へのアクセスを制御する』の説明を参照してください。

リソース・セキュリティを使用して、**DB2ENTRY** リソース定義および **DB2TRAN** リソース定義へのアクセスを制御する

CICS リソース・セキュリティ・メカニズムは、指定された CICS リソースへのユーザーのアクセスを制御します。例えば、このメカニズムを使用して、あるリソース (例えば、特定の DB2ENTRY 定義) が特定のユーザーによって変更されないようにすることができます。

このタスクについて

CICS コマンド・セキュリティは、ユーザーがリソースのタイプ (「すべての DB2ENTRY 定義」など) に対して特定のアクションを実行しないようにすることができますが、リソース・タイプ内の個々の項目を保護することはできません。

CICS 領域には 1 つの DB2CONN 定義しかないので、リソース・セキュリティを使用して保護する必要はありません。コマンド・セキュリティを使用して DB2CONN 定義へのアクセスを制御できます。また、リソース・セキュリティのために、DB2TRAN 定義はそれが参照する DB2ENTRY 定義の拡張と見なされ、それだけでリソース・セキュリティに定義されることはありません。ユーザーに DB2ENTRY 定義へのアクセス権限を与える場合、それを参照する DB2TRAN 定義へのアクセス権限も与えます。(トランザクションが、DB2TRAN 定義が関連付けられている DB2ENTRY の名前を変更する場合、二重セキュリティ検査が実行されます。この検査で、DB2TRAN 定義が参照した古い DB2ENTRY と、これから参照する新しい DB2ENTRY の両方を変更するユーザーの権限を確認します。) そのため、リソース・セキュリティでは、DB2ENTRY 定義を RACF に対して定義する必要のみがあります。

リソース・セキュリティがトランザクションに対して有効にされている場合、外部セキュリティ・マネージャーは、そのトランザクションに関連付けられているユーザー ID が、関係するリソースの変更を許可されているかどうかを検査します。リソース定義のセキュリティには、このプロセスに関する詳細情報があります。

リソース・セキュリティを使用して Db2 関連リソースを保護するには、以下の手順を実行します。

手順

1. RACF または同等の外部セキュリティ・マネージャーを有効にし、CICS 領域でリソース・セキュリティを有効にするには、CICS 領域のシステム初期設定パラメーターとして SEC=YES を指定します。

2. RACF で、Db2 関連のリソースを含める一般リソース・クラスを作成します。メンバー・クラスとグループ化クラスが必要です。CICS にデフォルトの RACF リソース・クラス名があるのとは異なり、DB2ENTRY には IBM 提供のデフォルトのクラス名はありません。RACF クラス記述子テーブル (CDT) のインストール定義部分 (モジュール ICHRRRCDE) に新規のクラス記述子を追加して、ユーザー独自のインストール定義クラス名を作成します。この方法の例については、CICSTS55.CICS.SDFHSAMP のメンバー DFH\$RACF で提供されている、IBM 提供のサンプル・ジョブ、RRCDTE を参照してください。ここには、XCICSDB2 というメンバー・クラスと、ZCICSDB2 というグループ化クラスの例があります。この例は、CICS のデフォルトのリソース・クラス名と同じ命名規則を使用します。Db2 関連のリソース定義に既存の CICS クラス名を使用しないでください。代わりに、同様の命名規則を使用して新しいクラス名を作成してください。
3. 作成したリソース・クラスで、DB2ENTRY 定義のプロファイルを定義します。例えば、DB2ENTRY 名の番号を XCICSDB2 リソース・クラスに追加するには、RDEFINE コマンドを以下のように使用します。

```
RDEFINE XCICSDB2 (db2ent1, db2ent2, db2ent3..., db2entn) UACC(NONE)
              NOTIFY(sys_admin_userid)
```

DB2ENTRY リソース定義を保護すると、関連する DB2TRAN 定義へのアクセスも保護されます。DB2TRAN はそれが参照する DB2ENTRY への拡張と見なされるからです。そのため、リソース・セキュリティを使用して、DB2CONN 定義を保護する必要はありません。

4. Db2 関連リソースのリソース・セキュリティをアクティブにするには、CICS 領域のシステム初期設定パラメーターとして XDB2=name を指定します (ここで、name は、Db2 関連リソースに定義された一般リソース・クラス名です)。
5. リソース・セキュリティを有効にしたい Db2 関連リソースが含まれるすべてのトランザクションに対して、リソース定義で RESSEC=YES を指定します。ユーザーがこれらのいずれかのトランザクションを使用して、保護されている Db2 関連リソースの 1 つにアクセスしようとする、RACF はユーザー ID がそのリソースへのアクセスを許可されていることを確認します。
6. CICS ユーザーまたはユーザーのグループに、保護されている各 Db2 関連リソースに対して適切なアクションを実行する権限を与えます。ユーザーが DB2ENTRY 定義に対してアクションを実行する権限を持っている場合、それに関連付けられている DB2TRAN 定義に対して同じアクションを実行することが自動的に許可されることを覚えておってください。ユーザーが特定のアクションを実行するために必要な権限は以下のとおりです。

INQUIRE コマンド

READ 権限が必要

SET コマンド

UPDATE 権限が必要

CREATE コマンド

ALTER 権限が必要

DISCARD コマンド

ALTER 権限が必要

例えば、次のように PERMIT コマンドを使用して、クラス XCICSDB2 の保護された DB2ENTRY である db2ent1 を UPDATE 権限で変更することをユーザーのグループに許可できます。

```
PERMIT db2ent1 CLASS(XCICSDB2) ID(group1) ACCESS(UPDATE)
```

コマンド・セキュリティを使用して、 **DB2CONN、DB2ENTRY、および DB2TRAN** の各リソース定義に 対する **SPI** コマンドの発行を制御する

CICS コマンド・セキュリティ・メカニズムを使用して、
DB2CONN、DB2ENTRY、および DB2TRAN の各リソース定義を保護します。

このタスクについて

CICS コマンド・セキュリティ・メカニズムは、Db2 関連リソースのタイプに対して特定の SPI コマンドをユーザーが発行する機能を制御します。例えば、このメカニズムを使用して、どのユーザーが DB2ENTRY リソース定義に対して CREATE コマンドおよび DISCARD コマンドを発行することが許可されるかを制御できます。リソース・セキュリティとは異なり、CICS コマンド・セキュリティは個々の指定されたリソースを保護できません。これはリソースのタイプを保護するように設計されています。コマンド・セキュリティを使用して、DB2CONN リソース定義、DB2ENTRY リソース定義、および DB2TRAN リソース定義を保護できます。

コマンド・セキュリティがトランザクションに対して有効にされている場合、外部セキュリティ・マネージャーは、そのトランザクションに関連付けられているユーザー ID が、そのコマンドを使用して関係するリソースのタイプを変更することを許可されているかどうかを検査します。CICS command securityには、このプロセスに関する詳細情報があります。

特定のトランザクションに対してリソース・セキュリティとコマンド・セキュリティの両方が有効にされている場合、RACF はユーザー ID に対して 2 つのセキュリティ検査を実行します。例えば、DB2ENTRY 定義 db2ent1 に対して DISCARD を発行するユーザーがトランザクションに含まれている場合、RACF は以下のことを検査します。

1. ユーザー ID が DB2ENTRY リソース・タイプに対して DISCARD コマンド (ALTER 権限) を発行することが許可されている。
2. ユーザー ID が ALTER 権限を使用して DB2ENTRY 定義 db2ent1 にアクセスすることが許可されている。

コマンド・セキュリティを使用して Db2 関連リソースを保護するには、以下の手順を実行します。

手順

1. RACF または同等の外部セキュリティ・マネージャーを CICS 領域に対して有効にするには、その領域のシステム初期設定パラメーターとして SEC=YES を指定します。
2. Db2 リソース名 DB2CONN、DB2ENTRY、および DB2TRAN を CICS コマンド用の IBM 提供の RACF リソース・クラスである CCICSCMD または

VCICSCMD のいずれかにリソース ID として追加します。あるいは、CICS コマンドにユーザー定義の一般リソース・クラスを使用することもできます。コマンド・セキュリティ検査の対象となる CICS リソースでは、これについて詳しく説明しています。例えば、次のように、REDEFINE コマンドを使用してデフォルト・クラス VCICSCMD に CMDSAMP というプロファイルを定義し、ADDMEM オペランドを使用して Db2 リソース・タイプがこのプロファイルによって保護されることを指定できます。

```
RDEFINE VCICSCMD CMDSAMP UACC(NONE)
        NOTIFY(sys_admin_userid)
        ADDMEM(DB2CONN, DB2ENTRY, DB2TRAN)
```

3. コマンド・セキュリティを CICS 領域に対して有効にするには、次のようにします。
 - a. CICS コマンド・プロファイルに IBM 提供の RACF リソース・クラス CCICSCMD または VCICSCMD を使用した場合、その領域のシステム初期設定パラメーターとして XCMD=YES を指定します。YES を指定することは、CCICSCMD および VCICSCMD が RACF のストレージ内のプロファイルの作成に使用されることを意味します。
 - b. CICS コマンドにユーザー定義の一般リソース・クラスを使用した場合、その領域の初期設定パラメーターとして XCMD=user_class を指定します (ここで、user_class はユーザー定義の一般リソース・クラスの名前です)。
4. コマンド・セキュリティを有効にしたい Db2 関連リソースが含まれるすべてのトランザクションに対して、リソース定義で CMDSEC=YES を指定します。ユーザーがこれらのいずれかのトランザクションを使用して、保護されている Db2 関連リソースの 1 つを変更しようとする、RACF はユーザー ID がそのリソースのタイプに対してそのコマンドを発行することを許可されていることを確認します。
5. CICS ユーザーまたはユーザーのグループに、Db2 関連リソースのそれぞれのタイプに対して適切なコマンドを発行する権限を与えます。コマンド・セキュリティの場合、DB2TRAN リソース・タイプおよび DB2ENTRY リソース・タイプに関連する別個の権限を与える必要があります。また、DB2CONN リソース・タイプ (すなわち、CICS 領域の DB2CONN 定義) を保護することもできます。

ユーザーが特定のコマンドを発行するために必要な権限は以下のとおりです。

INQUIRE コマンド

READ 権限が必要

SET コマンド

UPDATE 権限が必要

CREATE コマンド

ALTER 権限が必要

DISCARD コマンド

ALTER 権限が必要

例えば、ステップ 2 の例のように CMDSAMP プロファイルに Db2 リソース・タイプを定義した場合、次のように PERMIT コマンドを使用して、Db2 リソース・タイプに対して EXEC CICS INQUIRE コマンドを発行することをユーザーのグループに許可できます。

PERMIT CMDSAMP CLASS(VCICSCMD) ID(operator_group) ACCESS(READ)

トランザクション内で、**EXEC CICS QUERY SECURITY RESTYPE(SPCOMMAND)** コマンドを、DB2CONN、DB2ENTRY、または DB2TRAN を指定する RESID パラメーターとともに使用することで、ユーザー ID が Db2 リソース・タイプへのアクセス権限を持っているどうか照会できます。

代理セキュリティおよび **AUTHTYPE** セキュリティを使用して、**CICS** が **DB2** に提供する許可 ID へのアクセスを制御する

CICS 代理セキュリティ・メカニズムおよび AUTHTYPE セキュリティ・メカニズムは、CICS が DB2 に提供する許可 ID をユーザーが変更する機能を制御します。

このタスクについて

代理セキュリティおよび AUTHTYPE セキュリティを使用して、特定のユーザーのみに許可 ID の変更が許可されるようにします。この許可 ID は DB2 独自のセキュリティ検査に使用されます。代理セキュリティおよび AUTHTYPE セキュリティは CICS 領域全体に対して設定され、許可 ID の変更を含むトランザクションはそれらの影響を受けます。

70 ページの『CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する』では、これらの許可 ID を選択および変更する方法が説明されています。要約すると、CICS が DB2 に提供する許可 ID は、DB2 関連リソース定義では AUTHID 属性、COMAUTHID 属性、AUTHTYPE属性、および COMAUTHTYPE 属性によって設定され、CICS 領域の DB2CONN 定義では SIGNID 属性によって設定されます。許可 ID を変更するには、まず DB2CONN 定義および DB2ENTRY 定義を変更する権限が必要です。これらの定義はコマンド・セキュリティまたはリソース・セキュリティによって保護されている可能性があります。代理セキュリティは追加の保護レイヤーを提供します。代理セキュリティには DB2 に代わって機能する CICS が含まれており、許可 ID を変更するユーザーがリソース定義で指定された既存の許可 ID の代理として機能することを許可されているかどうか検査するからです。

真の代理セキュリティは、ユーザーが DB2CONN 定義または DB2ENTRY 定義で SIGNID 属性、AUTHID 属性、または COMAUTHID 属性を変更しようとするときにセキュリティ検査を行います。これらの属性はすべて、プロセスが DB2 にサインオンするときに使用される許可 ID を指定します。CICS は、RACF の代理ユーザー機能を使用して、この検査を実行します。代理ユーザーとは、他のユーザーのパスワードを知らなくてもそのユーザーに代わって作業を行う権限を持っているユーザーのことです。ユーザーが SIGNID 属性、AUTHID 属性、または COMAUTHID 属性のいずれかを変更しようすると、CICS は RACF を呼び出して、そのユーザーが SIGNID 属性、AUTHID 属性、または COMAUTHID 属性で現在指定されている許可 ID の代理として許可されていることを確認します。

AUTHTYPE 属性および COMAUTHTYPE 属性は、許可 ID そのものを指定するのではなく、使用される許可の ID のタイプを提供するため、CICS は真の代理セキュリティを使用できません。代わりに、AUTHTYPE セキュリティと呼ばれるメカニズムを使用します。ユーザーが AUTHTYPE 属性または

COMAUGHTYPE 属性のいずれかを変更しようとする、CICS は RACF を呼び出して、ユーザーが RACF FACILITY 一般リソース・クラスでリソース定義に対して定義されているプロファイルによって許可されていることを確認します。AUTHTYPE セキュリティーが真の代理セキュリティでない場合でも、同じシステム初期設定パラメーターによって有効にされており、それを代理セキュリティに加えて使用する可能性があります。そのため、このトピックの説明には両方のタイプのセキュリティのセットアップ方法が含まれています。

DB2CONN リソース定義および DB2ENTRY リソース定義が CICS のコールド・スタートまたは初期始動の一環としてインストールされているときに、代理セキュリティと AUTHTYPE セキュリティーが有効にされている場合、RACF は CICS 領域ユーザー ID に対して代理セキュリティ検査と AUTHTYPE セキュリティー検査を行います。このようにして DB2CONN リソース定義および DB2ENTRY リソース定義をインストールする場合、CICS 領域ユーザー ID がリソース定義で指定された許可 ID の代理ユーザーとして定義されていること、およびそれが RACF FACILITY 一般リソース・クラスの正しいプロファイルによって許可されていることを確認してください。

CICS が DB2 に提供する許可 ID を保護するために代理セキュリティおよび AUTHTYPE セキュリティーを実装するには、以下の手順を実行します。

手順

1. RACF または同等の外部セキュリティ・マネージャーを CICS 領域に対して有効にするには、その領域のシステム初期設定パラメーターとして SEC=YES を指定します。
2. CICS 領域の代理セキュリティおよび AUTHTYPE セキュリティーをアクティブにするには、その領域のシステム初期設定パラメーターとして XUSER=YES を指定します。このシステム初期設定パラメーターは、両方のセキュリティ・メカニズムを有効にします。セキュリティ・メカニズムが有効にされている場合、DB2CONN リソース定義および DB2ENTRY リソース定義の SIGNID、AUTHID、COMAUTHID、AUTHTYPE、および COMAUGHTYPE の各属性に対して機能する EXEC CICS SET コマンド、CREATE コマンド、および INSTALL コマンドがトランザクションに含まれているときはいつも、CICS は RACF を呼び出してセキュリティ検査を実行します。SIGNID 属性、AUTHID 属性、および COMAUTHID 属性の場合、RACF は代理セキュリティ検査を実行し、AUTHTYPE 属性または COMAUGHTYPE 属性の場合、RACF は AUTHTYPE セキュリティー検査を実行します。
3. 代理セキュリティのために、適切な CICS ユーザーまたはユーザーのグループを、DB2CONN 定義および DB2ENTRY 定義の SIGNID 属性、AUTHID 属性、または COMAUTHID 属性で指定された許可 ID の代理として定義する必要があります。ユーザー ID を許可 ID の代理として定義するには、以下のようになります。
 - a. RACF SURROGAT クラスに許可 ID のプロファイルを作成します。名前の形式は *authid.DFHINSTL* とし、許可 ID は所有者として定義します。例えば、DB2AUTH1 を SIGNID 属性、AUTHID 属性、または COMAUTHID 属性で許可 ID として指定した場合、次のコマンドを使用してプロファイルを作成します。

```
RDEFINE SURROGAT DB2AUTH1.DFHINSTL UACC(NONE) OWNER(DB2AUTH1)
```

- b. 作成したプロファイルの READ 権限を適切な CICS ユーザーに付与することで、そのユーザーが許可 ID の代理として機能することを許可します。例えば、ID が CICSUSR1 のユーザーに、許可 ID が DB2AUTH1 の代理として機能することを許可し、そのために既存の許可 ID として DB2AUTH1 を指定している SIGNID 属性、AUTHID 属性、または COMAUTHID 属性をインストールあるいは変更するには、次のコマンドを使用します。

```
PERMIT DB2AUTH1.DFHINSTL CLASS(SURROGAT) ID(CICSUSR1) ACCESS(READ)
```

SIGNID 属性、AUTHID 属性、または COMAUTHID 属性に指定されたすべての許可 ID に対して、このプロセスを繰り返します。

- c. SIGNID 属性、AUTHID 属性、または COMAUTHID 属性が CICS のコールド・スタートまたは初期始動の一環として含まれる DB2CONN リソース定義および DB2ENTRY リソース定義をインストールする必要がある場合、それらの属性によって指定された許可 ID の代理として機能することを CICS 領域ユーザー ID に許可します。DB2CONN リソース定義および DB2ENTRY リソース定義のデフォルトには、AUTHID 属性および COMAUTHID 属性は含まれていません。インストール済みの DB2CONN 定義のデフォルト SIGNID は、CICS 領域のアプリケーション ID です。
4. AUTHTYPE セキュリティーのために、RACF FACILITY 一般リソース・クラスの DB2CONN リソース定義または DB2ENTRY リソース定義ごとにプロファイルを作成し、そのプロファイルに対する READ アクセス権限を適切な CICS ユーザーまたはユーザーのグループに与える必要があります。(このプロセスは真の代理セキュリティ・メカニズムを模倣しますが、特定の許可 ID の使用は含まれません。代わりに、各リソース定義を保護します。) 以下はその方法です。

- a. RACF FACILITY 一般リソース・クラスに DB2CONN リソース定義または DB2ENTRY リソース定義のプロファイルを作成します。名前の形式は DFHDB2.AUTHTYPE.authname とします (ここで、authname は DB2CONN リソース定義または DB2ENTRY リソース定義の名前です)。例えば、DB2CONN1 という名前の DB2CONN リソース定義のプロファイルを定義するには、次のコマンドを使用します。

```
RDEFINE FACILITY DFHDB2.AUTHTYPE.DB2CONN1 UACC(NONE)
```

- b. 適切な CICS ユーザーに、作成したプロファイルに対する READ 権限を与えます。例えば、ID が CICSUSR2 のユーザーに、DB2CONN1 という名前の DB2CONN リソース定義の AUTHTYPE 属性または COMAUTHTYPE 属性をインストールあるいは変更することを許可するには、次のコマンドを使用します。

```
PERMIT DFHDB2.AUTHTYPE.DB2CONN1 CLASS(FACILITY) ID(CICSUSR2) ACCESS(READ)
```

このプロセスを DB2CONN リソース定義および DB2ENTRY リソース定義のそれぞれに対して繰り返します。また、AUTHTYPE 属性または COMAUTHTYPE 属性が CICS のコールド・スタートまたは初期始動の一環として含まれる DB2CONN リソース定義および DB2ENTRY リソース定義をインストールする必要がある場合は、そのリソース定義のプロファイルの CICS 領域ユーザー ID に READ 権限を与えます。

タスクの結果

Db2 関連の CICS トランザクションへのユーザー・アクセスの制御

CICS トランザクション接続セキュリティ・メカニズムを使用して、データを取得するために Db2 にアクセスする CICS トランザクション、DSNC トランザクション、および CICS Db2 接続機能コマンドと Db2 コマンドを発行する他のすべてのトランザクションに対するユーザーのアクセスを制御します。

このタスクについて

トランザクション接続セキュリティが使用可能な場合、RACF または同等の外部セキュリティ・マネージャーは、CICS ユーザーが要求したトランザクションを実行する権限がその CICS ユーザーにあるかを検査します。

トランザクション接続セキュリティを使用する Db2 関連トランザクションを保護するには、トランザクション・セキュリティの説明に従ってください。プロセスはすべての CICS トランザクションについて同じです。トランザクション接続セキュリティ・メカニズムに関する限り、Db2 関連トランザクションに特別な考慮事項はありません。その指示は、以下の方法を示しています。

- トランザクション接続セキュリティをアクティブ化するために、CICS 領域用の適切なシステム初期設定パラメーターをセットアップする (トランザクション接続セキュリティを制御する CICS パラメーターを参照)。
- 保護したいトランザクションについて、トランザクション・プロファイルを RACF に定義する (RACF profilesを参照)。

CICS Db2 接続機能コマンドと Db2 コマンドを発行する、DSNC 以外のトランザクションを定義した場合 (例えば、各コマンドを実行するための別個のトランザクションを定義した場合) には、これらのトランザクションも同様に RACF に定義することを覚えておいてください。

どの CICS ユーザーが、Db2 にアクセスするトランザクションを使用できるかを制御できるようになりました。トランザクション・プロファイル用のアクセス・リストに、適切なユーザー、またはユーザーのグループを READ 権限と共に追加してください。RACF profilesには、これに関するいくつかの推奨事項があります。

CICS Db2 接続機能コマンドと Db2 コマンドを発行するトランザクションについて、以下の点に留意してください。

- CICS Db2 接続機能コマンドは、CICS と Db2 間の接続上で作動し、それらは完全に CICS 内で実行されます。Db2 コマンドは Db2 自体の中で作動し、それらは Db2 に差し向けられます。Db2 コマンドを CICS Db2 接続機能コマンドから区別するために、ハイフン (-) 文字を Db2 コマンドと一緒に入力します。
- DSNC トランザクションへのアクセス権限がある場合、CICS は、すべての CICS Db2 接続機能コマンドと Db2 コマンドを発行することを許可します。
- 個々の CICS Db2 接続機能コマンドと Db2 コマンドを実行するための別個のトランザクションを定義した場合は、別の CICS ユーザーに、これらのトランザクション・コードのサブセットに対する権限と、それに伴ってコマンドのサブセッ

トに対する権限を与えることができます。例えば、何人かのユーザーに CICS Db2 接続機能コマンドを発行する権限を与えるけれども Db2 コマンドについてはそうしない、ということも可能です。CICS Db2 用の CICS 提供トランザクションには、CICS Db2 接続機能コマンドと Db2 コマンドに対して CICS が供給する、別個のトランザクション定義の名前があります。

CICS Db2 接続機能コマンドが Db2 に流れることはないで、それらは、これ以上のセキュリティ検査の対象にはなりません。それらは CICS トランザクション接続セキュリティのみによって保護されます。ただし、Db2 コマンド、およびデータを取得するために Db2 にアクセスする CICS トランザクションは、以下のように Db2 のセキュリティ・メカニズムによるそれ以降の段階のセキュリティ検査の対象になります。

- トランザクションが Db2 にサインオンするときには、有効な許可 ID を Db2 に提供しなければなりません。許可 ID は RACF によって、または同等の外部セキュリティ・マネージャーによって検査されます。
- トランザクションは Db2 コマンドを発行したり、Db2 データにアクセスしたりするので、トランザクションが提供した許可 ID には、これらのアクションを Db2 内で実行する権限が必要です。Db2 では、GRANT ステートメントを使用して、アクションを実行する権限を許可 ID に与えることができます。

加えて、CICS 領域自体、Db2 サブシステムに接続する権限が必要です。

『CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する』には、Db2 サブシステムに接続するために CICS 領域に権限を与える方法と、トランザクションに有効な許可 ID を提供する方法が記されています。

80 ページの『ユーザーに対する Db2 内のリソースへのアクセス許可』には、トランザクションが Db2 に提供した許可 ID に権限付与する方法が記されています。

CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する

CICS には、Db2 に許可 ID を提供する必要のある 2 つのタイプのプロセスがあります。その 1 つは CICS 領域と Db2 の間の全体的な接続で、もう 1 つは Db2 へのスレッドを獲得する CICS トランザクションです。

このタスクについて

セキュリティの目的で、Db2 では、データへのあらゆる形式のアクセスを表すために「プロセス」という語を使用します。これは、ユーザーが直接 Db2 と対話するものと、CICS を含む他のプログラム経由でユーザーが Db2 と対話するもののどちらかになります。Db2 に接続するプロセス、またはサインオンするプロセスは、Db2 アドレス・スペースのセキュリティ検査に使用できる許可 ID という名前の短い Db2 ID を、1 つ以上提供する必要があります。すべてのプロセスは 1 次許可 ID を提供する必要があり、オプションとして 2 次許可 ID を 1 つ以上提供することもできます。Db2 特権および権限を 1 次または 2 次の許可 ID に付与することができます。例えばユーザーは、2 次許可 ID を使用して表を作成することができます。すると、その表はその 2 次許可 ID によって所有されます。同じ

2 次許可 ID を Db2 に提供する他のすべてのユーザーは、その表に対して関連した特権を持ちます。ユーザーから特権を取り去るために、管理者はそのユーザーをその許可 ID から切断することができます。

CICS には、Db2 に許可 ID を提供する必要がある、以下の 2 つのタイプのプロセスがあります。

- CICS 領域と Db2 との全体的な接続。これは CICS Db2 接続機能によって作成されます。このプロセスは、Db2 の接続処理を行って Db2 に許可 ID を提供する必要があります。
- Db2 へのスレッドを獲得する CICS トランザクション。これらは例えば、Db2 データベースからデータを取り出すトランザクション、あるいは Db2 コマンドを発行する DSNB トランザクションです。それぞれの CICS トランザクションごとに、Db2 から見える実際のプロセスはスレッド TCB です。これは、Db2 に入れるトランザクションのスレッドを制御するために CICS が使用するものです。これらのプロセスは、Db2 のサインオン処理を行って Db2 に許可 ID を提供する必要があります。

接続処理およびサインオン処理の間に、Db2 は、Db2 アドレス・スペースで使用するプロセスのために、1 次および 2 次の許可 ID を設定します。デフォルトで Db2 は、プロセスが提供した許可 ID を使用します。しかし、接続処理とサインオン処理の両方には出口ルーチンが関与しており、これらの出口ルーチンは、1 次および 2 次の許可 ID の設定にユーザーが影響を与えることを可能にします。Db2 には、デフォルトの接続出口ルーチンとデフォルトのサインオン出口ルーチンがあります。これらを独自の出口ルーチンに置き換えることができます。これを支援するために、サンプルの接続出口ルーチンとサインオン出口ルーチンが Db2 に付属しています。

CICS 領域用の許可 ID を Db2 に提供する

CICS は、1 次許可 ID および 1 つ以上の 2 次許可 ID を Db2 に提供します。

このタスクについて

CICS Db2 接続機能で CICS 領域と Db2 の間の全体的な接続を作成する場合、そのプロセスには Db2 の接続処理が含まれます。CICS 領域は、以下を提供することができます。

- 1 次許可 ID。1 次許可 ID は、Db2 での CICS 領域の 1 次 ID になります。CICS 領域と Db2 の間の接続の場合、Db2 の接続処理に最初に渡される 1 次許可 ID を選択することはできません。これは、CICS 領域のユーザー ID です。ただし、独自の接続出口ルーチンを作成することで、接続処理中に Db2 が設定する 1 次 ID を変更できます。RACF または同等の外部セキュリティ・マネージャがアクティブである場合、それに対して CICS 領域のユーザー ID が定義されている必要があります。72 ページの『CICS 領域用の 1 次許可 ID を提供する』は、CICS 領域に対して指定できる 1 次許可 ID について説明しています。
- 1 つ以上の 2 次許可 ID。RACF グループの名前、またはグループのリストを、CICS 領域の 2 次許可 ID として使用できます。これを実行する場合、デフォルトの Db2 接続出口ルーチン DSN3@ATH (1 次許可 ID を Db2 に渡すことのみを行う) を置き換える必要があります。サンプル Db2 接続出口ルーチン

DSN3SATH は、RACF グループの名前を 2 次許可 ID として Db2 に渡します。あるいは、CICS 領域の 2 次 ID を設定する、独自の接続出口ルーチンを作成できます。73 ページの『CICS 領域用の 2 次許可 ID を提供する』は、CICS 領域の 2 次許可 ID のセットアップ方法を説明しています。

CICS 領域用の 1 次許可 ID を提供する

Db2 に渡される 1 次許可 ID は、CICS が開始タスク、開始ジョブ、ジョブのいずれとして稼働しているかによって異なります。

このタスクについて

CICS が Db2 から要求する接続タイプは、単一アドレス・スペース・サブシステム (SASS) です。CICS 領域と Db2 の間の接続の場合、最初に Db2 の接続処理に渡される 1 次許可 ID を選択することはできません。CICS 領域の 1 次許可 ID として Db2 に渡される ID は次のいずれかです。

- CICS が開始タスクとして稼働している場合は、RACF 開始済みプロシージャ・テーブル、ICHRIN03 から取得されたユーザー ID。
- CICS が開始ジョブとして稼働している場合は、STARTED 一般リソース・クラス・プロファイルの STDATA セグメントのユーザー・パラメーター。
- CICS がジョブとして稼働している場合は、JOB カードの USER パラメーターで指定されたユーザー ID。

CICS 領域が使用する可能性があるユーザー ID は、RACF または同等の外部セキュリティ・マネージャー (外部セキュリティ・マネージャーがアクティブの場合) に対して定義する必要があります。ユーザー ID を RACF に対して USER プロファイルとして定義します。これを RESOURCE プロファイルとして定義するだけでは十分ではありません。

CICS 領域のユーザー ID を RACF に対して定義したら、以下のようにして Db2 へのアクセスを許可します。

1. 接続のタイプが単一アドレス・スペース・サブシステム (SASS) である Db2 サブシステムのプロファイルを RACF クラス DSNR に定義します。例えば、次の RACF コマンドでは、Db2 サブシステム DB2A への SASS 接続用のプロファイルをクラス DSNR に作成します。

```
RDEFINE DSNR (DB2A.SASS) OWNER(DB2OWNER)
```

2. CICS 領域のユーザー ID が Db2 サブシステムにアクセスすることを許可します。例えば、次の RACF コマンドでは、ユーザー ID が CICSHA11 の CICS 領域に Db2 サブシステム DB2A への接続を許可します。

```
PERMIT DB2A.SASS CLASS(DSNR) ID(CICSHA11) ACCESS(READ)
```

Db2 の接続出口ルーチンは、CICS 領域によって提供された 1 次許可 ID (ユーザー ID) を取得し、それを CICS 領域の 1 次 ID として Db2 に設定します。デフォルトの Db2 接続出口ルーチン DSN3@ATH とサンプルの Db2 接続出口ルーチン DSN3SATH はどちらもこのように動作します。独自の接続出口ルーチンを作成することによって、Db2 が設定する 1 次 ID を変更することができます。サンプル接続出口ルーチン、および出口ルーチンの作成について詳しくは、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。しかし、CICS 領域の 2 次許可

ID を提供し、1 次許可 ID ではなく、それらの 2 次許可 ID に基づいて CICS 領域に権限を付与する方がより簡単です。

CICS 領域用の 2 次許可 ID を提供する

CICS 領域が Db2 に接続するとき、1 次許可 ID に加えて、1 つ以上の 2 次許可 ID を Db2 に提供できます。CICS 領域が接続されている RACF グループまたはグループのリストの名前を 2 次許可 ID として使用できます。これにより、CICS 領域ごとに考えられるすべての 1 次許可 ID に Db2 特権を付与する代わりに、RACF グループに Db2 特権および権限を付与してから、複数の CICS 領域を同じグループに接続できます。

このタスクについて

CICS 領域が接続されている RACF グループまたはグループのリストの名前を Db2 に 2 次許可 ID として提供するには、以下の手順を実行します。

手順

1. CICS が RACF を使用するようにするために、CICS 領域のシステム初期設定パラメーターとして SEC=YES を指定します。
2. CICS 領域を適切な RACF グループまたはグループのリストに接続します。RACF グループ・プロファイルを参照してください。
3. デフォルトの Db2 接続出口ルーチン DSN3@ATH を置き換えます。この出口ルーチンは接続処理中に駆動されます。デフォルトの接続出口ルーチンは 2 次許可 ID をサポートしないため、それをサンプルのサインオン出口ルーチンである DSN3SATH に置き換える必要があります。これは、Db2 に付属するものか、またはユーザー独自のルーチンです。DSN3SATH は Db2 SDSNSAMP ライブラリー内のソースの形で出荷されるもので、それをベースにして独自のルーチンを作成できます。DSN3SATH は、CICS 領域が接続されている RACF グループの名前を 2 次許可 ID として Db2 に渡します。「RACF グループ・リスト」オプションがアクティブである場合、DSN3SATH は、CICS 領域が接続されているすべてのグループの名前を取得して、それらを 2 次許可 ID として使用します。「RACF グループ・リスト」オプションがアクティブでない場合、DSN3SATH は、CICS 領域の現在接続されているグループの名前を唯一の 2 次許可 ID として使用します。

タスクの結果

CICS 領域が Db2 に接続するとき、サンプルの接続出口ルーチンは CICS 領域の 1 次許可 ID (領域のユーザー ID) を 1 次 ID として設定し、CICS 領域が接続されている RACF グループの名前を 2 次 ID として設定します。

次のタスク

RACF グループの名前を 2 次許可 ID として Db2 に提供する代わりに方法として、CICS 領域の 2 次許可 ID を設定する独自の接続出口ルーチンを作成することもできます。接続出口ルーチンの作成については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

CICS トランザクション用の許可 ID を Db2 に提供する

CICS は Db2 に許可 ID を渡すことができるため、CICS は RACF を使用している必要があります。また、SIT で SEC=YES が指定されている必要があります。これは、CICS が RACF アクセス制御環境エレメント (ACEE) を Db2 に渡す必要があるためです。

このタスクについて

CICS トランザクションのスレッド TCB は、Db2 にサインオンして Db2 のサインオン処理を実行すると、以下を提供することができます。

- 1 次許可 ID。CICS トランザクションの場合、1 次許可 ID を選択できます。これは CICS ユーザーのユーザー ID またはオペレーター ID、端末 ID、トランザクション ID のいずれか、または指定した ID にすることができます。1 次許可 ID として使用される ID は、DB2ENTRY 定義 (エントリー・スレッドの場合) または DB2CONN 定義 (プール・スレッドおよびコマンド・スレッドの場合) の属性によって決定されます。75 ページの『CICS トランザクション用の 1 次許可 ID を提供する』は、CICS トランザクションの 1 次許可 ID を選択する方法を説明しています。
- 1 つ以上の 2 次許可 ID。RACF グループまたはグループのリストの名前を 2 次許可 ID として使用できます。これには、個々の CICS ユーザーにそれぞれ Db2 特権および権限を付与するのではなく、RACF グループにそれらを付与できるという利点があります。2 次許可 ID を使用するには、DB2ENTRY 定義の AUTHTYPE 属性 (エントリー・スレッドの場合)、あるいは DB2CONN 定義の AUTHTYPE 属性または COMAUTHTYPE 属性 (プール・スレッドまたはコマンド・スレッドの場合) を使用して、GROUP オプションを指定します。また、デフォルトの Db2 サインオン出口ルーチン DSN3@SGN を置き換える必要があります。デフォルトのルーチンが 2 次許可 ID を Db2 に渡すことはないからです。GROUP オプションを指定する際に、1 次許可 ID がトランザクションに関連付けられた CICS ユーザーのユーザー ID として自動的に定義されます。78 ページの『CICS トランザクション用に 2 次許可 ID を提供する』では、2 次許可 ID のセットアップ方法および使用方法を説明しています。

CICS トランザクションが Db2 に提供する許可 ID を選択する際の主要な考慮事項は、Db2 アドレス・スペースでのセキュリティ検査用に選択したセキュリティ・メカニズムです。このセキュリティ検査は、Db2 コマンド、計画、および動的 SQL へのアクセスを対象とします。このセキュリティ検査が以下によって実行されることを選択できます。

- Db2 内部セキュリティ。
- RACF または同等の外部セキュリティ・マネージャー。
- 一部は Db2、一部は RACF。

Db2 アドレス・スペース内の一部またはすべてのセキュリティ検査に RACF を使用している場合、Db2 にサインオンする CICS トランザクションは、以下のいずれかの方法で許可 ID を提供する必要があります。

- スレッド (DB2ENTRY または DB2CONN) の適切な定義で AUTHTYPE(USERID) または COMAUTHTYPE(USERID) を指定して、トランザクションに関連付けられた CICS ユーザーのユーザー ID を 1 次許可 ID として Db2 に提供します。
- スレッド (DB2ENTRY または DB2CONN) の適切な定義で AUTHTYPE(GROUP) または COMAUTHTYPE(GROUP) を指定して、トランザクションに関連付けられた CICS ユーザーのユーザー ID を 1 次許可 ID として、RACF グループまたはグループのリストの名前を 2 次許可 ID として Db2 に提供します。
- スレッド (DB2ENTRY または DB2CONN) の適切な定義で AUTHTYPE(SIGN) を指定し、DB2CONN の SIGNID 属性で CICS 領域ユーザー ID を指定して、CICS 領域 ID を 1 次許可 ID として Db2 に提供します。

CICS 領域の RACF アクセス制御環境エレメント (ACEE) が、CICS Db2 接続機能に影響を与えるような方法で変更される場合、サインオンが行われるまで Db2 は変更を認識しません。CEMT コマンドまたは EXEC CICS SET DB2CONN SECURITY(REBUILD) コマンドを使用して、スレッドが次回再利用されるとき、またはスレッドが既にサインオン済みの TCB で作成されるときに、CICS Db2 接続機能が Db2 サインオンを発行するように指定できます。これにより、Db2 にセキュリティ変更を認識させることができます。

CICS トランザクション用の 1 次許可 ID を提供する

CICS トランザクションのスレッド TCB が Db2 にサインオンするとき、Db2 に 1 次許可 ID を提供する必要があります。トランザクションがその 1 次許可 ID として使用する ID は、トランザクションが Db2 へのアクセスに使用するスレッドのリソース定義内の属性によって決定されます。

このタスクについて

これは、同じタイプのスレッド (同じタイプのエントリー・スレッド、プール・スレッド、またはコマンド・スレッドのいずれか) を使用するすべてのトランザクションが、同じタイプの 1 次許可 ID を使用する必要があることを意味します。各 CICS 領域で、以下に対して 1 次許可 ID を設定する必要があります。

- DB2ENTRY 定義を使用する、各タイプのエントリー・スレッド。
- DB2CONN 定義を使用するプール・スレッド。
- DB2CONN 定義を使用するコマンド・スレッド (DSNC トランザクションに使用される)。

1 次許可 ID の設定を開始する前に、それを実行する権限があることを確認してください。DB2CONN 定義または DB2ENTRY 定義を変更するための権限に加え、CICS 領域に対して代理ユーザー検査が強制される場合 (つまり、システム初期設定パラメーター XUSER が YES に設定されている場合)、Db2 許可 ID が関係する操作を実行するための特殊権限を取得する必要があります。これらの操作は、DB2ENTRY 定義または DB2CONN 定義上の AUTHID、COMAUTHID、AUTHTYPE、または COMAUTHTYPE の各属性の変更、および DB2CONN 定義上の SIGNID 属性の変更です。66 ページの『代理セキュリティおよび AUTHTYPE セキュリティを使用して、CICS が DB2 に

提供する許可 ID へのアクセスを制御する』は、これらの操作を実行する権限をユーザーに付与する方法を説明しています。

特定のタイプのスレッドに 1 次許可 ID を設定するには、次の 2 とおりの方法があります。

1. DB2ENTRY 定義の AUTHID 属性 (エントリー・スレッドの場合)、または DB2CONN 定義の AUTHID 属性あるいは COMAUTHID 属性 (プール・スレッドまたはコマンド・スレッドの場合) を使用して、1 次許可 ID を指定します。例えば、AUTHID=test2 を定義できます。この場合、CICS Db2 接続機能は、文字 TEST2 を 1 次許可 ID として Db2 に渡します。

AUTHID および COMAUTHID の使用は、2 次許可 ID の使用を許可するものではありません。さらにその使用は、Db2 アドレス・スペースのセキュリティ検査で、RACF または同等の外部セキュリティ・マネージャーの使用と両立しません。

2. DB2ENTRY 定義の AUTHTYPE 属性 (エントリー・スレッドの場合)、または DB2CONN 定義の AUTHTYPE 属性あるいは COMAUTHTYPE 属性 (プール・スレッドまたはコマンド・スレッドの場合) を使用して、トランザクションに関連する既存の ID を 1 次許可 ID として使用するように CICS に指示します。この ID は、CICS ユーザー ID、オペレーター ID、端末 ID、またはトランザクション ID でも構いません。あるいは、CICS 領域の DB2CONN 定義で指定した ID でも構いません。

AUTHTYPE または COMAUTHTYPE の使用は、USERID または GROUP オプションを使用する場合、Db2 アドレス・スペースのセキュリティ検査での RACF (または同等の外部セキュリティ・マネージャー) の使用と両立します。また、GROUP オプションを使用する場合、2 次許可 ID の使用と両立します。

1 次許可 ID を決定する 2 とおりの方法は相互排他的です。AUTHID と AUTHTYPE の両方、および COMAUTHID と COMAUTHTYPE の両方を同じリソース定義で指定することはできません。

セキュリティ・マネージャーが Db2 サブシステムに対してアクティブである場合、1 次許可 ID として選択したすべての ID は、RACF または同等の外部セキュリティ・マネージャーに対して定義する必要があることに注意してください。RACF に対して、1 次許可 ID は、単に RESOURCE プロファイルとして (例えば、端末またはトランザクションとして) ではなく、RACF USER プロファイルとして定義する必要があります。

DB2CONN リソースおよび DB2ENTRY リソースの説明に従って、DB2CONN 定義および DB2ENTRY 定義をセットアップまたは変更します。AUTHTYPE 属性または COMAUTHTYPE 属性を使用してスレッド・タイプのための 1 次許可 ID を決定する場合は、77 ページの表 3 を使用して、必要な許可 ID の提供と必要な機能のサポートを行うオプションを特定します。考慮すべきキーポイントは、次のとおりです。

- 1 次許可 ID だけでなく 2 次許可 ID も Db2 に提供する場合は、GROUP オプションを選択する必要があります。GROUP オプションを指定すると、1 次

許可 ID が CICS ユーザー ID として自動的に定義されますが、セキュリティ検査は 2 次許可 ID に基づいたものにできます。

- Db2 アドレス・スペースでのセキュリティ検査に RACF を使用する場合、GROUP オプションまたは USERID オプションのいずれかを選択する必要があります。RACF アクセス制御環境エレメント (ACEE) を Db2 に渡すことができるのはこれらのオプションのみであり、セキュリティ検査に RACF を使用する場合は必須になります。
- 許可 ID の選択による、パフォーマンスと保守への影響を考慮してください。これについては、パフォーマンスおよびメンテナンスのための許可 ID の選択に概説されています。USERID、OPID、TERM、TX、または GROUP オプションを使用すると、より多くの許可 ID にアクセス権を付与するため、サインオン処理がさらに頻繁に実行され、保守にもより多くの時間がかかります。SIGN オプションを使用する場合、または AUTHTYPE 属性の代わりに AUTHID 属性を使用する場合、サインオン処理は減り、保守の複雑さは低減されます。ただし、標準許可 ID を使用すると、Db2 のセキュリティ検査の精度は低下します。
- 許可 ID の選択による、アカウントティングへの影響を考慮してください。許可 ID は、各 Db2 アカウントティング・レコードで使用されます。アカウントティングの観点から、最も詳細な情報は USERID、OPID、GROUP、または TERM を使用する場合に取得されます。ただし、ACCOUNTREC の指定に応じて、どのようなケースでも個別のユーザー・レベルでアカウントティングを行うということは不可能である場合があります。CICS Db2 環境でのアカウントティングの詳細については、『モニター』の『CICS Db2 環境のアカウントティングとモニター』を参照してください。

表 3 は、AUTHTYPE 属性または COMAUTHTYPE 属性の各オプションを選択するときに CICS Db2 接続機能が Db2 に渡す、1 次許可 ID を示しています。

表 3. AUTHTYPE 属性および COMAUTHTYPE 属性で使用可能なオプション

オプション	Db2 に渡される 1 次許可 ID	Db2 に対する RACF 検査のサポート	2 次許可 ID のサポート
USERID	RACF に定義されて CICS サインオンで使用される、CICS トランザクションに関連付けられたユーザー ID	はい	いいえ
OPID	RACF ユーザー・プロファイルの CICS セグメントで定義された、ユーザーの CICS オペレーター ID	いいえ	いいえ
SIGN	CICS 領域に対する DB2CONN 定義の SIGNID 属性で指定された ID。デフォルトは CICS 領域のアプリケーション ID	はい (DB2CONN リソースの SIGNID 属性が CICS 領域ユーザー ID と一致する場合)。	いいえ
TERM	トランザクションと関連付けられた端末の端末 ID	いいえ	いいえ
TX	トランザクション ID	いいえ	いいえ

表 3. AUTHTYPE 属性および COMAUTHTYPE 属性で使用可能なオプション (続き)

オプション	Db2 に渡される 1 次許可 ID	Db2 に対する RACF 検査のサポート	2 次許可 ID のサポート
GROUP	CICS サインオンで使用される、ユーザーの CICS RACF ユーザー ID	はい	はい

CICS トランザクションに 2 次許可 ID を提供する計画がない場合は、デフォルトの Db2 サインオン出口ルーチン DSN3@SGN を置き換える必要はありません。デフォルトのサインオン出口ルーチンは、1 次許可 ID を処理します。ただし、接続先の Db2 サブシステムは、他の何らかの理由で別のサインオン出口ルーチンを使用する可能性があります。Db2 サブシステムがサンプル・サインオン出口ルーチン DSN3SSGN を使用する場合、以下のすべての条件が当てはまれば、DSN3SSGN への変更が必要になる可能性があります。

- GROUP ではなく AUTHID オプションまたは AUTHTYPE オプションを選択している。
- グループ処理の RACF リストがアクティブである。
- 1 次許可 ID が RACF に定義されていないトランザクションがある。

このような場合、サンプル・サインオン出口ルーチンに加える必要のある変更については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

CICS トランザクション用に 2 次許可 ID を提供する

CICS トランザクションに属するスレッド TCB が Db2 にサインオンするときに、1 次許可 ID に加えて、1 つ以上の 2 次許可 ID を Db2 に提供することができます。

このタスクについて

CICS ユーザーの RACF グループの名前、またはグループのリストを、2 次許可 ID として Db2 に提供できます。これには、個々の CICS ユーザーにそれぞれ Db2 特権および権限を付与するよりも、RACF グループに権限付与できるという利点があります。その後 CICS ユーザーを必要に応じて RACF グループに接続したりグループから除去したりすることができます。RACF グループ・プロファイルでは、ユーザーを RACF グループに配置する方法を説明しています。

CICS トランザクション用の 2 次許可 ID を Db2 に提供できるのは、DB2ENTRY 定義内の AUTHTYPE 属性 (エントリー・スレッドの場合)、あるいは DB2CONN 定義内の AUTHTYPE または COMAUTHTYPE 属性 (プール・スレッドまたはコマンド・スレッドの場合) に GROUP オプションを指定した場合だけです。

AUTHTYPE または COMAUTHTYPE に他の何らかのオプションを指定した場合は、2 次許可 ID はブランクに設定されます。GROUP オプションを指定するときは、スレッド・タイプに 1 次許可 ID を選択することはできません。それは自動的に、トランザクションに関連付けられた CICS ユーザーのユーザー ID として定義されます。代わりに、2 次許可 ID をセキュリティ検査のベースにしてください。

ユーザーの RACF グループの名前を Db2 に 2 次許可 ID として提供するには、以下のステップ全体を実行します。

1. CICS が RACF を使用するようにするため、CICS 領域のシステム初期設定パラメーターとして SEC=YES を指定します。CICS トランザクション・プロファイル名が接頭部付きで定義されている場合は、システム初期設定パラメーター SECPRFX=YES または SECPRFX=prefix も指定してください。
2. CICS 領域が MRO を使用している場合:
 - a. 接続されたそれぞれの CICS 領域も RACF セキュリティーを使用していること (SEC=YES) を確認します。
 - b. サインオン情報が TOR から AOR に伝搬されるようにするため、TOR の CONNECTION 定義に ATTACHSEC=IDENTIFY を指定します。
3. デフォルトの Db2 サインオン出口ルーチンである DSN3@SGN を置き換えます。このサインオン出口ルーチンはサインオン処理中に駆動されます。デフォルトのサインオン出口ルーチンは 2 次許可 ID をサポートしないので、それをサンプルのサインオン出口ルーチンである DSN3SSGN に置き換える必要があります。DSN3SSGN は Db2 に付属するものか、またはユーザー独自のルーチンです。DSN3SSGN は Db2 SDSNSAMP ライブラリー内のソースの形で出荷されるもので、それをベースにして独自のルーチンを作成できます。サインオン出口、および出口ルーチンの作成については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。「RACF グループ・リスト」オプションがアクティブでない場合、DSN3SSGN は、現在接続されているグループの名前を 2 次許可 ID として渡します。「RACF グループ・リスト」オプションがアクティブである場合は、DSN3SSGN は、ユーザーが接続されているすべてのグループの名前を取得して、それらを 2 次許可 ID として Db2 に渡します。
4. DB2ENTRY 定義内の AUTHTYPE 属性 (エントリー・スレッドの場合)、または DB2CONN 定義の AUTHTYPE または COMAUTHTYPE 属性 (プール・スレッドまたはコマンド・スレッドの場合) を使用して、2 次許可 ID を提供したいスレッドの各タイプごとに、GROUP オプションを許可タイプとして指定します。CICS 領域に対して代理ユーザー検査が強制される場合 (つまり、システム初期設定パラメーター XUSER が YES に設定されている場合) には、Db2 許可 ID が関係する操作を実行するための特殊権限を取得する必要があります。これを行う方法については 66 ページの『代理セキュリティおよび AUTHTYPE セキュリティーを使用して、CICS が DB2 に提供する許可 ID へのアクセスを制御する』を参照してください。

上記にリストされているすべてのステップを正常に完了した場合は、GROUP オプションを付けて定義したスレッドのタイプを使って CICS トランザクションのスレッド TCB が Db2 にサインオンするときに、CICS ユーザーのユーザー ID が 1 次許可 ID として Db2 に渡され、ユーザーの RACF グループまたはグループ・リストが 2 次許可 ID として Db2 に渡されます。すべてのステップを正常に完了していない場合は、CICS Db2 接続機能は、許可の失敗メッセージを出します。

RACF グループの名前を 2 次許可 ID として Db2 に提供する代わりに方法として、CICS トランザクションの 2 次許可 ID を設定する独自のサインオン出口ルーチンを書くこともできます。Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

ユーザーに対する Db2 内のリソースへのアクセス許可

ユーザーが CICS トランザクションから Db2 アドレス・スペースにアクセスすると、Db2 コマンドを発行したり計画を実行したりするための許可が必要になる場合があります。

このタスクについて

CICS ユーザーが Db2 コマンド発行や計画実行の際に必要な Db2 リソースへのアクセスは、Db2 のセキュリティ・メカニズムによるセキュリティ検査の対象になります。このセキュリティ検査が以下によって実行されることを選択できます。

- Db2 内部セキュリティ。
- RACF、または同等の外部セキュリティ・マネージャー。
- 一部は Db2、一部は RACF。

Db2 アドレス・スペースでセキュリティ検査を実行するための RACF のセットアップについて詳しくは、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

Db2 アドレス・スペース内の一部またはすべてのセキュリティ検査に RACF を使用している場合は、Db2 にサインオンする CICS トランザクションが許可 ID を提供する必要がありますことに注意してください。詳細については、74 ページの『CICS トランザクション用の許可 ID を Db2 に提供する』を参照してください。さらに、CICS は RACF を使用している必要もあります (SIT で SEC=YES を指定する必要があります)。これは、RACF を Db2 アドレス・スペース内のセキュリティ検査に使用する場合、CICS は RACF アクセス制御環境要素 (ACEE) を Db2 に渡す必要があるためです。CICS は RACF がアクティブである場合にのみ ACEE を生成でき、GROUP、SIGN、または USERID の各オプションで定義されたスレッドのみが ACEE を Db2 に渡すことができます。

ACEE は、Db2 に渡されると、Db2 出口の DSNX@XAC によって使用され、そこで RACF、IBM 以外の同等の外部セキュリティ・マネージャー、または Db2 内部セキュリティをセキュリティ検査に使用するかどうかが決まります。DSNX@XAC は、スレッドが Db2 にサインオンしているトランザクションが API 要求を発行すると駆動します。DSNX@XAC を変更できます。詳しくは、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

Db2 または外部セキュリティ・マネージャーは、CICS トランザクションが、使用スレッドが Db2 にサインオンしたときに Db2 に提供した許可 ID を使用して、セキュリティ検査を実行します。許可 ID は、個々の CICS ユーザー (例えば、CICS ユーザーのユーザー ID や、ユーザーが接続されている RACF グループ) に関連しているか、トランザクション (例えば、端末 ID やトランザクション ID) に関連しているか、または CICS 領域全体に関連している場合があります。詳しくは、70 ページの『CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する』を参照してください。

Db2 または外部セキュリティ・マネージャーは、Db2 内の関連アクションを実行する権限を、ユーザーが許可 ID に付与済みであることを検査します。Db2 で GRANT ステートメントを使用することによって、この権限を許可 ID に付与でき

ます。許可 ID に対する Db2 許可の付与、および取り消し方法については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

Db2 コマンドへのユーザーのアクセスの制御

CICS ユーザーの場合、ユーザーが Db2 コマンドを発行する CICS トランザクションにアクセスしようとする、Db2 コマンドに関連する最初のセキュリティ検査が CICS アドレス・スペースで実行されます。これは、DSNC トランザクション、または DFHD2CM1 を呼び出して個々の Db2 コマンドを実行するユーザー定義のトランザクションである可能性があります。

このタスクについて

69 ページの『Db2 関連の CICS トランザクションへのユーザー・アクセスの制御』では、CICS アドレス・スペースで Db2 コマンドを発行するトランザクションに対するユーザーのアクセスを制御する方法を説明しています。

ユーザーが CICS トランザクションで Db2 コマンドを発行すると、そのユーザーは Db2 のセキュリティ検査の対象にもなります。その検査で、Db2 でコマンドを発行する許可があるかどうかを検証されます。このセキュリティ検査では、トランザクションによって CICS から渡された許可 ID (1 次または 2 次) が使用されます。70 ページの『CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する』では、これらの許可 ID を選択して、Db2 に提供する方法を説明しています。DFHD2CM1 を使用して Db2 コマンドを発行するトランザクションの場合、許可 ID は CICS 領域の DB2CONN 定義の COMAUTHID 属性または COMAUTHTYPE 属性によって設定されます。Db2 コマンドを発行するその他のアプリケーションの場合、許可 ID は、トランザクションによって使用されるスレッドのタイプ (プール・スレッドまたはエントリー・スレッド) に対する CICS 領域のリソース定義の AUTHID 属性または AUTHTYPE 属性によって設定されます。これらの属性は、そのタイプのスレッドを使用するトランザクションによって Db2 に渡される許可 ID (許可 ID のタイプ) を制御します。

したがって、Db2 コマンドは 2 つのセキュリティ検査の対象になります。1 つは CICS アドレス・スペース、もう 1 つは Db2 アドレス・スペースで行われます。82 ページの図 24 は、このプロセスを示しています。

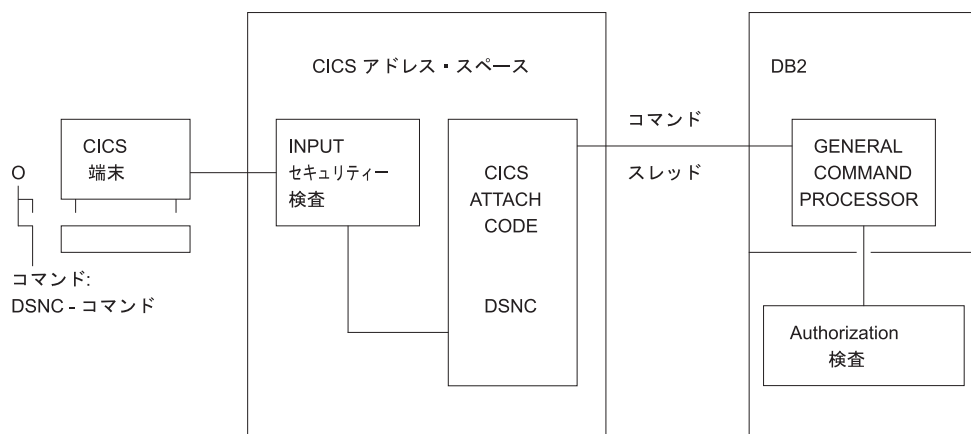


図 24. Db2 コマンドのセキュリティー・メカニズム

ほとんどの場合、Db2 コマンドの実行が許可されているのは、限られた数のユーザーだけです。有用な解決策は、DB2CONN 定義で COMAUGHTYPE(USERID) を指定することです。そうすると、Db2 での許可 ID として 8 バイトの CICS ユーザー ID に解決されます。この方法を使用すると、異なる Db2 特権を CICS ユーザー ID に明示的に与えることができます。例えば、GRANT DISPLAY を使用して、-DIS コマンドのみを使用する権限を特定の CICS ユーザー ID に与えることができます。

ユーザーに Db2 コマンドを発行する権限を与えるには、GRANT コマンドを使用して、トランザクションによって CICS から渡された許可 ID に Db2 コマンド特権を付与します。許可 ID に対する Db2 許可の付与、および取り消し方法については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

計画へのユーザー・アクセスの制御

ユーザーが Db2 の計画にアクセスする前に、いくつかの検査が行われます。この検査は CICS アドレス・スペースで始まり、その後 Db2 に要求が渡されてさらに検査が行われます。

このタスクについて

Db2 コマンドでは、CICS が、計画を実行するトランザクションへのアクセスがユーザーに許可されているかどうか確認するときに、計画へのユーザー・アクセスに対する最初のセキュリティー検査が CICS アドレス・スペースで行われます。Db2 が、トランザクションによって提供される許可 ID に計画の実行許可があるかどうか確認するときに、2 番目のセキュリティー検査が Db2 アドレス・スペースで行われます。83 ページの図 25はこの処理を示しています。

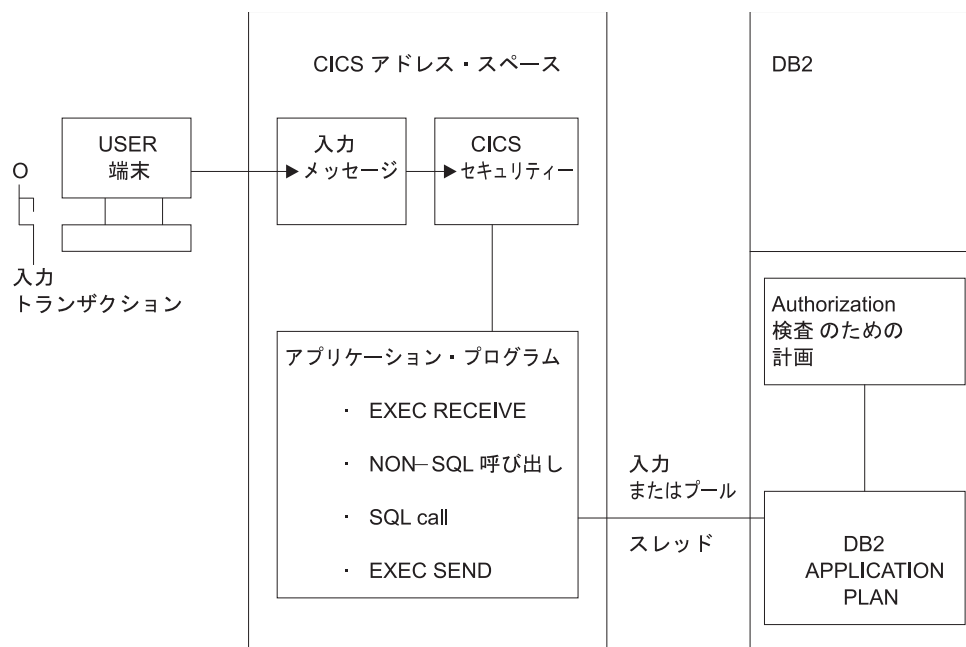


図 25. 計画の実行におけるセキュリティー・メカニズム

ユーザーに計画の実行許可を与えるには、GRANT コマンドを使用して、トランザクションによって CICS から渡された許可 ID に Db2 コマンド特権を付与します。許可 ID に対する Db2 許可の付与、および取り消し方法については、Db2 for z/OS 製品資料内の『Db2 の保護』を参照してください。

計画に動的 SQL が含まれる場合

静的 SQL を使用する場合、計画のバインダーには、データへのアクセスに必要な特権がなければなりません。CICS から Db2 に渡される許可 ID には、計画を実行するための特権があれば十分です。

このタスクについて

ただし、計画に動的 SQL の使用が含まれている場合、CICS から Db2 へ渡される許可 ID には、関係するすべての Db2 リソース (計画とデータの両方) にアクセスするために必要な特権がなければなりません。例えば、AUTHTYPE(USERID) を指定する場合、CICS ユーザー ID には、動的 SQL に関係する Db2 リソースに対する Db2 特権を付与する必要があります。このユーザー ID が TSO ユーザー ID でもある場合は、SPUFI、QMF™、およびその他のユーティリティーから Db2 リソースに直接アクセスできます。

動的 SQL の使用に関係する計画をトランザクションが実行する場合に、Db2 特権の付与にあまり時間を費やしたくない場合は、以下の、許可 ID を Db2 に提供する方式のいずれかを使用することを検討してください。

- トランザクションによって使用されるスレッドの DB2ENTRY 定義の AUTHTYPE 属性で、SIGN オプションを使用します。これにより、トランザクションは、CICS 領域の DB2CONN 定義の SIGNID 属性に指定された 1 次許可 ID を持つことになります。(この方式は、RACF が Db2 アドレス・スペースでのセキュリティー検査に使用される場合は適していません。)

- トランザクションによって使用されるスレッドの DB2ENTRY 定義の AUTHID 属性を使用して、標準許可 ID を指定します。動的 SQL にアクセスする必要があるすべてのトランザクションに対して、同じ許可 ID を使用します。(この方式は、RACF が Db2 アドレス・スペースでのセキュリティー検査に使用される場合は適していません。)
- RACF グループを作成し、CICS ユーザーをこの RACF グループに接続します。トランザクションで使用されるスレッドの DB2ENTRY 定義の GROUP 属性を使用して、RACF グループが Db2 に渡されるセカンダリー ID の 1 つになるようにします。

それぞれのケースで、すべての動的 SQL に関係する Db2 リソースに対する Db2 特権を、単一の ID (DB2CONN 定義または AUTHID 属性からの標準許可 ID であるか、あるいは RACF グループの名前のいずれか) に付与できます。70 ページの『CICS 領域用および CICS トランザクション用に、許可 ID を Db2 に提供する』は、これらのメソッドによって許可 ID を提供する方法を説明しています。

Db2 マルチレベル・セキュリティーおよび行レベル・セキュリティー

DB2 バージョン 8 では、マルチレベル・セキュリティーのサポートが導入されました。CICS はマルチレベル・セキュリティーのための固有のサポートを提供しますが、構成が気になる場合には、マルチレベル・セキュリティー環境で CICS を使用できます。

マルチレベル・セキュリティーについて詳しくは、以下の資料を参照してください。

- Db2 for z/OS 製品資料内の『Db2 の保護』
- z/OS マルチレベル・セキュリティーの計画および Common Criteria
- IBM Redbooks: z/OS での DB2 の保護と MLS の実装

DB2 バージョン 8 以降で、マルチレベル・セキュリティーがデータに対して行レベル (行レベルのセキュリティー) で実装されている場合、RACF SECLABEL クラスがアクティブ化され、一連のセキュリティー・レベルがユーザーおよび Db2 表の行に対して定義されます。RACF オプションである SETR MLS と MLACTIVE は、アクティブである必要はありません。Db2 行レベルのセキュリティーは、MVS システムの残りの部分に影響を与えることなく使用できます。

CICS は、このような方法で保護された Db2 行にアクセスできます。CICS の場合、Db2 行へのアクセスを必要とする CICS ユーザーの RACF ユーザー・プロファイルが、デフォルトの SECLABEL を含むように RACF に定義されていることを確認する必要があります。詳しくは、z/OS Security Server RACF セキュリティー管理者のガイドを参照してください。

CICS ユーザーが SIT に SEC=YES を指定して CICS 領域にサインオンすると、RACF はデフォルトの SECLABEL をユーザーの RACF アクセス制御環境エレメント (ACEE) に関連付けます。DB2ENTRY 定義 (プールが使用されている場合は DB2CONN 定義) は、AUTHTYPE=USERID または AUTHTYPE=GROUP を指定する必要があります。これにより、ACEE が今後のセキュリティー検査のために Db2 に渡されます。そのため、個々の CICS ユーザーは、関連付けられた SECLABEL を 1 つしか持つことができません。

PLT プログラムなどの非端末タスクまたはプログラムの場合、 PLTPUSR システム初期設定パラメーターが指定されておらず、PLTPSEC=NONE システム初期設定パラメーターが指定されている場合、PLT プログラムは CICS 領域ユーザー ID で実行されます。この場合、デフォルトの SECLABEL を使用して CICS 領域ユーザー ID を定義する必要があります。トランザクションにさまざまな SECLABEL を定義する必要がある場合、異なる CICS 領域ユーザー ID および関連 SECLABEL を持つ別個の CICS 領域で、それぞれのトランザクションを実行する必要があります。

第 5 章 CICS Db2 に関するアプリケーション設計と開発の考慮事項

設計プロセス中

アプリケーション設計プロセス中に、行った決定が、現在開発中のアプリケーションや計画されている今後のアプリケーションに影響を与える場合があります。主な設計上の考慮事項には以下のようなものがあります。

- CICS アプリケーションと、Db2 計画およびパッケージの間の関係。システムのパフォーマンスと管理について最大の利点を得るには、Db2 計画、トランザクション、およびアプリケーション・プログラムの間の関係を、アプリケーションの設計時に定義する必要があります。
- ロック戦略。ロックは、Db2 でのテーブル作成時に選択するオプション、アプリケーション・プログラムの設計、および計画のバインド時に選択するオプションから影響を受けるので、開発プロセス全体を通して考慮する必要があります。
- CICS Db2 テスト・システムと CICS Db2 実動システムの両方のセキュリティ一面を考慮します。
- コマンドとプログラミング手法の使用。これらはアプリケーションのパフォーマンスを向上させ、潜在的な問題を回避することができます。修飾および非修飾の SQL の使用が、CICS Db2 環境の他の面に影響する場合があります。JVM サーバー環境で Java™ アプリケーションを実行し、オープン・トランザクション環境 (OTE) のパフォーマンス上の利点を得るには、アプリケーション・プログラムがスレッド・セーフである必要があります。詳しくは、108 ページの『CICS Db2 アプリケーションに関する SQL、スレッド・セーフ、その他のプログラミング考慮事項』を参照してください。
- CICS Db2 環境内の Java プログラム。サポートおよびプログラミングの考慮事項については、123 ページの『第 6 章 Java プログラムから Db2 データにアクセスするための JDBC および SQLJ の使用』を参照してください。
- アプリケーション開発の完了時およびアプリケーションの実稼働環境への移行時に実行する追加ステップ。
- パフォーマンスの向上のために定義される接続。アプリケーションのパフォーマンスが向上するような CICS Db2 接続を定義します。例えば、Db2 にアクセスするアプリケーション・プログラムの保護エントリー・スレッドを使用すると、頻繁に使用されるアプリケーションのパフォーマンスが向上します。

設計から影響を受ける要因

実装する設計によって、以下のものが影響を受ける可能性があります。

- パフォーマンス
- 並行性
- 操作
- セキュリティー
- アカウンティング

- 開発環境

アプリケーション・パフォーマンスに影響を与える要因

優れた設計の CICS Db2 アプリケーションは、実稼働環境への最初のデプロイ時に適切に機能するはずですが、以下のいくつかの要因が、優れた設計のアプリケーションのパフォーマンスに影響を与える場合があります。

- トランザクション率の増大
- 既存アプリケーションの継続的な開発
- CICS Db2 アプリケーションの開発に参与する人の増加
- 新規アプリケーションで使用される既存のテーブル
- アプリケーションの統合

これらの要因により、CICS 環境での Db2 の使用に関する一貫性のある一連の標準を作成することが重要です。

VSAM および DL/I に保管されるデータを持つアプリケーション

VSAM または DL/I に保管されているデータがあるアプリケーションの場合、CICS Db2 アプリケーションに比べ、一部のプロセスの処理方法が異なります。考慮すべき相違点は以下のとおりです。

- ロック・メカニズム
- セキュリティー
- リカバリーと再始動
- バインド処理
- 操作手順
- パフォーマンス
- プログラミング手法

バッチ・プログラムとオンライン・プログラム設計の主な相違点の 1 つは、オンライン・システムを高度な並行性に対応するように設計する必要があるという点です。同時に、データ保全性にいかなる妥協があってもなりません。また、ほとんどのオンライン・システムにはパフォーマンスに関する設計基準があります。

追加情報

アプリケーション設計に関する以下の追加情報ソースを参照してください。

- Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』
- アプリケーションの設計

CICS アプリケーションと Db2 計画およびパッケージの間の関係の設計

CICS アプリケーションが Db2 データを使用するときには、アプリケーション・アーキテクチャーでは CICS Db2 接続機能に関する設計面を考慮する必要があります。考慮すべき最も重要な面の 1 つは、トランザクション ID、Db2 計画とパッケージ、およびプログラム・モジュールの間の関係です。

プログラム・モジュールのいずれかで SQL 呼び出しを使用する場合、対応する Db2 計画またはパッケージが使用可能でなければなりません。トランザクション ID に使用される計画は、トランザクションが Db2 へのアクセスに使用するスレッドの DB2CONN または DB2ENTRY 定義に指定されます。この計画に明示的に名前を付けることも、計画名を選択する計画出口ルーチンに名前を付けることもできます。計画には、このトランザクション ID の下で実行する可能性のあるすべてのモジュール内の DBRM を含める必要があります。DBRM を計画に直接バインドすることも、パッケージとしてバインドし、計画内のパッケージ・リストに指定することもできます。

計画と CICS Db2 接続機能スレッドの特性を制御するには、トランザクション ID、Db2 計画、およびプログラム・モジュールの間の関係を設計ステップで定義する必要があります。設計に依存するスレッド、環境記述マネージャー (EDM) プール、および計画の特性のいくつかを以下にリストします。

- 計画のサイズ
- 同時に使用されているさまざまな計画の数
- 同時に使用されているスレッドの数
- スレッドの再使用の可能性
- EDM プールのサイズ
- EDM プールでの I/O アクティビティ
- システムの全体のページ率
- 許可の細分度
- Db2 パッケージの使用

CICS トランザクションと Db2 計画を組み合わせるための設計手法はさまざまですが、推奨される手法はパッケージに基づいて計画を使用することです。

サンプル・アプリケーション

単純な例を使用して、さまざまなアプリケーション設計技法の結果について説明します。

90 ページの図 26 は、CICS MAP とトランザクション ID がどのように相互に関連付けられるか、トランザクションがどのように機能するかを Db2 については考慮しないで示しています。

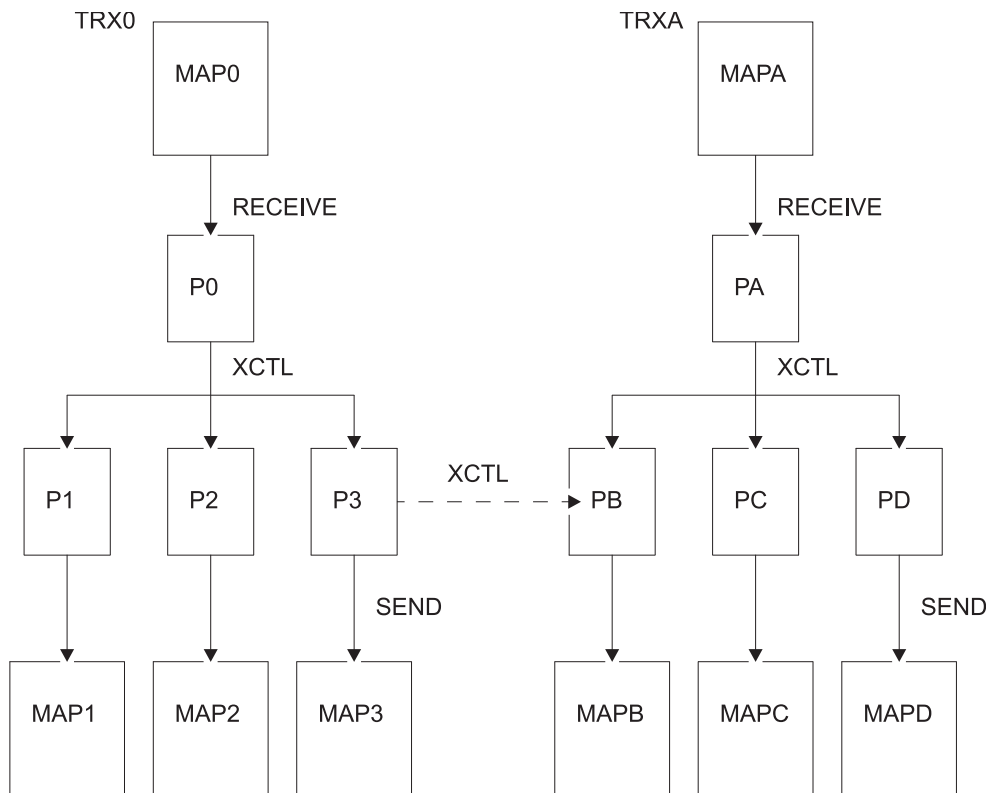


図 26. 標準的なアプリケーション設計の例

この例では、次のようになっています。

- トランザクション ID の TRX0 は、プログラム (ここには示されていない) が MAP0 を表示した後に制御を返すときに、EXEC CICS RETURN TRANSID(TRX0) コマンドで指定されます。
- 次のトランザクションでは、端末ユーザーにより実行される内容とは関係なく、トランザクション ID の TRX0 が使用されます。
- プログラム P0 はトランザクション TRX0 の初期プログラムです。
- 示されているすべてのプログラムが SQL 呼び出しを発行しているものと想定します。
- 端末ユーザーにより選択されるオプションに応じて、プログラム P0 は、プログラム P1、P2、または P3 のいずれかに対して CICS 制御権移動 (XCTL) を実行します。
- いくつかの SQL 呼び出しを発行した後、これらのプログラムは、マップの MAP1、MAP2、または MAP3 のいずれかを表示します。
- 図の右側に示している例も同様に機能します。
- 場合によっては、プログラム P3 がプログラム PB に制御を移動します。

Db2 パッケージの使用

CICS トランザクションと Db2 計画の間の関係を容易に管理する最良の方法は、パッケージの使用です。

パッケージが使用可能になる前の Db2 の初期リリースでは、特定の CICS トランザクションの下で実行されるすべてのプログラムからの DBRM を単一の計画に直接バインドする必要がありました。計画内の 1 つの DBRM を変更すると (そのプログラムの SQL 呼び出しを変更したため)、計画内のすべての DBRM を再バインドする必要がありました。大規模な計画をバインドすると非常に時間がかかる場合があります、その操作中は、トランザクション全体の処理が使用不可になります。使用頻度の高いアプリケーションの場合、アプリケーションを静止するだけでも容易ではありませんが、再バインドのために長時間にわたって計画を使用不能のままにすることも通常、受け入れられません。

動的計画出口は、この問題に対処するために設計された暫定の解決策でした。動的計画出口は、計画名を指定する代わりに、DB2CONN または DB2ENTRY 定義で指定する出口プログラムです。いくつかのプログラムからの DBRM をそれぞれに含む、CICS アプリケーションの多数の小規模な計画を作成して、各作業単位の開始時に、出口プログラムで適切な計画を選択します。トランザクション内の計画を切り替えるために動的計画出口を使用することもできます。ただし、いずれかのプログラムで使用する SQL ステートメントが変更されるたびに、各小規模計画は再バインドする必要があり、使用不能となります。また、動的計画切り替えの使用では、計画の切り替えを許可するために、暗黙的または明示的に CICS **SYNCPPOINT** が必要になります。

現在では、Db2 でパッケージが使用可能になったため、それらを使用することが、計画を管理する最良の方法です。パッケージを使用すると、プログラムをより小規模な単位に分割でき、計画全体に影響を及ぼさずに、また再バインド中の特定のパッケージの現行ユーザーにも影響を及ぼさずに、それらのプログラムを個別に再バインドできます。

計画の更新はパッケージを使用した方が容易なため、より大規模なアプリケーションを作成でき、Db2 のパフォーマンスや可用性に適応させるためにトランザクション、プログラム、または計画を切り替える必要もありません。このことはまた、多数の RDO 定義を保守する必要がないことを意味します。プログラムの柔軟性を得るために動的 SQL を使用するような状況も避けることができます。計画内で、まだ存在していないパッケージを指定することや、パッケージを追加できるパッケージ・コレクションを指定することもできます。これは、既存の計画を全く妨げずに、新規プログラムからの DBRM を追加できることを意味します。

異なるテーブル・セットを参照するための、パッケージや計画の修飾子オプションによって、より多くの柔軟性を得ることができ、計画の切り替えを回避できます。

要約すると、パッケージは以下のとおりです。

- プログラムをバインドしている間の、計画の停止時間、プロセッサ時間、およびカタログ・テーブル・ロックを最小限にします。ここで示すプログラムとは、**START**、**LINK**、または **RETURN TRANSID** で論理的にリンクされていて、DB2ENTRY 定義を削減するためにバインド済みの DBRM を含んでいるものです。
- CICS **START** または出口を削減します。
- CICS 定義および DB2ENTRY 定義のクローン作成を防止します。
- 後で計画に組み込むために、ヌル・パッケージで計画をバインドする機能を提供します。

- SET CURRENT PACKAGESET=*variable* を使用して、実行時にコレクションを指定することを可能にします。これは、**QUALIFIER** と共に使用すると強力な機能となります。
- **QUALIFIER** パラメーターを指定します。これにより、以下に柔軟性が加わります。
 - 非修飾 SQL を使用して別のテーブル・セットを参照できるようにします。
 - 同義語と別名の必要性を減らします。
 - 動的 SQL の必要性を減らします。

CICS 環境内でパッケージを使用すると得られるその他の利点は以下のとおりです。

- 使用する計画の数を減らすことができます。
- 使用頻度の低いトランザクションを単一の計画にバインドできます。
- スレッド再使用の可能性が増えます。

エントリーを順番に配列するパッケージ・リストを使用して検索時間を削減するか、パッケージ・リストのエントリーを最小に保持することによって、アプリケーションを最適化することもできます。

Db2 ではパッケージ・レベルでのアカウントリングも提供されます。パッケージについて詳しくは、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」のバインドの計画および実行するアプリケーション・プログラムの準備についての説明と、IBM Redbooks® 資料「Db2 9 for z/OS®: Packages Revisited」を参照してください。

パッケージへの変換

動的計画出口または切り替え手法を使用しているトランザクションを、代わりにパッケージを使用するように変換することができます。

このタスクについて

現在、動的計画出口または動的計画切り替え手法を使用しているトランザクションを、以下のようにパッケージを使用するように変換することができます。

- トランザクションに関連付けられた計画に含まれるすべての DBRM を、単一コレクション内のパッケージにバインドする。
- このコレクションの単一ワイルドカード・エントリーを含む PKLIST を使用するように、新規計画をバインドする。
- 新規計画を使用するために、このトランザクションの DB2ENTRY 項目を変更する。保護スレッドは現在、スレッドの再使用を最適化するためにこのトランザクションで使用できます。

アプリケーション全体に対して単一の計画、またはトランザクションごとに 1 つの計画を使用するように選択することができます。以下のセクションでは、この選択に基づいて、トランザクションを変換する方法を詳しく説明します。

CICS アプリケーションで動的計画出口を使用しているかどうかに関係なく、すべての CICS アプリケーションの変換の際に同様の方法を使用できます。

使用率の高いパッケージは、RELEASE(COMMIT) でバインドされた使用率の低いパッケージに、RELEASE(DEALLOCATE) でバインドできることに注意してください

い。これにより、計画が割り振り解除されるまで使用率の高いパッケージが計画のパッケージ・ディレクトリで保持され、使用率の低いパッケージがディレクトリから削除され、EDM プール内のスペースが解放されます。この方法の欠点は、RELEASE(DEALLOCATE) を使用して、頻繁に使用されるパッケージを再バインドする必要がある場合に、パッケージが長期実行スレッドに割り振られている場合には、そのパッケージを強制的に割り振り解除するためにユーザー介入が必要になることです。パッケージの再バインドは、それにかかる時間という意味で、プランの再バインドより低コストである点を考慮してください。

アプリケーションに対して 1 つの計画の使用:

アプリケーションに対して 1 つの計画を使用することで、アプリケーションの DB2ENTRY および DB2TRAN を定義する際に最高の柔軟性が得られます。これは、より効率的に保護スレッドを使用し、スレッドの再使用を最適化するために関係するトランザクションをグループ化できるためです。この環境に変換するには、以下の手順を実行します。

手順

1. COLLAPP1 などの単一のコレクションを使用して、アプリケーション内のトランザクションのすべての DBRM をパッケージにバインドします。
2. 単一項目 COLLAPP1.* で構成されるパッケージ・リストを使って、新規計画 PLANAPP1 をバインドします。

```
BIND PLAN (PLANAPP1) ..... PKLIST (COLLAPP1.*) ..
```

3. DB2ENTRY で、動的計画出口プログラム名をこの新規計画の名前に置き換えます。例えば、次のように置き換えます。

```
PLANEXITNAME=DSNCUEXT
```

with

```
PLAN=PLANAPP1
```

トランザクションごとに 1 つの計画の使用:

個々のパッケージ・レベルでアカウンティングできなかった (そのため、強制的に計画名でアカウンティングが行われていた) ため、DB2 バージョン 3 より前ではこの方法が推奨されました。しかし、現行リリースの Db2 ではパッケージ・レベルでのアカウンティングが提供されるため、必要なアカウンティングの細分性を引き続き提供しながら、計画を多数のパッケージで構成できます。

手順

1. トランザクション・グループまたはすべてのトランザクションの DBRM をパッケージにバインドします。使用されるコレクションは、TRAN1 などの変換されるトランザクション、またはアプリケーションに基づくものにすることができます。トランザクションに基づいてコレクションを作成および保守するためには、特に共通ルーチンが多数ある場合、より一層の管理努力が必要になるため、後者の方法をお勧めします。
2. 使用する方法によって異なる単一のワイルドカード・パッケージ・リスト項目を参照するパッケージ・リストを使って、各トランザクションの新規計画をバインドします。

- コレクションがトランザクションに基づく場合は、TRAN1.* を使用します。
BIND PLAN (TRAN1) PKLIST (TRAN1.*) ...
- すべてのトランザクションに単一のコレクションがセットアップされている場合は、COLLAPP1.* を使用します。
BIND PLAN (TRAN1) PKLIST (COLLAPP1.*) ...

3. DB2ENTRY 定義を変更して、動的計画出口をトランザクションに関連付けられている計画名に置き換えます。

パッケージを使用してさらに動的計画切り替え技法を使用する場合

動的計画切り替えユーザーの場合、変更可能特殊レジスター (例えば、CURRENT PACKAGESET レジスター) の値が、処理中に変更されてから、SYNCPOINT の前に初期状態にリセットされないと、以下ようになります。

- スレッドが CICS Db2 接続機能によって解放されません。
- 同じ計画が割り振られて、タスクがこのスレッドを継続して使用するため、スレッドの再使用が発生しません。
- 同じスレッドと計画が使用されるため、動的計画切り替えが発生しません。つまり、SYNCPOINT に続いて発行される最初の SQL ステートメントで動的計画切り替え出口が得られません。

したがって、動的計画切り替えを使用する場合は、SYNCPOINT の前に変更可能特殊レジスターが確実に初期値にリセットされるように注意する必要があります。変更可能特殊レジスターは、Db2 for z/OS 製品資料内の『SQL: Db2 の言語』にリストされています。

すべてのトランザクションのために 1 つの大規模な計画を使用する

トランザクション中の計画切り替えを回避する簡単な方法は、SQL 呼び出しを発行するすべての CICS トランザクションのために 1 つの大規模な計画を使用することです。

90 ページの図 26 の例の場合、以下ようになります。

- 1 つの計画 PLAN0 があり、P0、P1、P2、P3、PA、PB、PC、および PD からの DBRM を使用します。
- CICS では、PLAN0 を指定する 1 つの DB2ENTRY と、必要なトランザクション ID ごとに DB2TRAN を定義します (ワイルドカード・トランザクション ID を指定できる場合は除きます)。保護スレッドが使用されている場合、1 つの DB2ENTRY によってスレッド全体の最良の使用効率が得られます。

利点

- 任意のトランザクション ID で実行できるプログラム・モジュールに関する制約事項はありません。例えば、プログラム P3 からプログラム PB への制御移動が可能です。この場合、CICS で計画または定義を変更する必要はありません。
- DBRM はそれぞれ 1 つの計画 (PLAN0) にのみ存在します。
- スレッドの再使用が簡単に行えます。すべてのトランザクションで同じ DB2ENTRY を使用できます。

欠点

すべてのトランザクションで単一の大きな計画を使用する場合にいくつかの欠点があります。

- どんな Db2 プログラムの変更に対しても、計画全体を再バインドする必要があります。
- 大きな計画のバインドには時間がかかることがあります。
- 計画を使用している間はバインド処理を行うことはできません。実動システムではほとんどの時間に、通常のアクティビティーにより計画が使用されていると考えられます。テスト環境では、通常、トランザクション率は低いですが、プログラマーが使用できるデバッグ・ツールは、会話型プログラムでは応答時間が長くなります。これにより、実質的にスレッドと計画がビジー状態のままになることがあります。
- Db2 によって呼び出される REBIND (計画の無効化による) では、Db2 トランザクションでこの計画を実行できません。
- 計画が 1 つしかないため、計画名を示すメッセージや統計には実際の情報価値はありません。
- EDMPOOL は DBD、SKCT、および CT に対応するために十分な大きさでなければならない、幾らかフラグメント化できる必要があります。計画セグメンテーション機能では、Db2 は、実行されているアプリケーション計画部分を CT にしかロードできないことに注意してください。ただし、アプリケーション計画のヘッダーとディレクトリー部分は、SKCT にそのままロードされ (まだロードされていない場合)、その後、SKCT から CT にコピーされます。これは、スレッドの作成時に行われます。

アプリケーション計画のディレクトリー・サイズは計画内のセグメントの数に直接依存しているため、大きな計画を使用すると、EDMPOOL サイズと、その制御に必要な入出力操作の数に影響します。

多数の小規模な計画を使用する

多数の小規模な計画を使用すると、多数のトランザクション ID (それぞれに計画が対応する) が必要になります。この技法によって、計画のサイズおよび計画のオーバーラップが削減されます。

多数の小規模な計画を使用するよりも、パッケージを使用することをお勧めします。

多数の小規模な計画を使用することは、プログラム・フローが、分岐の可能性が限定的な狭いパスに従うことを意味するか、または計画切り替えが頻繁に発生することを意味します。

90 ページの図 26 の例では、プログラム P0 とすぐ下のレベルのプログラムとの間、またはプログラム PA とすぐ下のレベルのプログラムとの間で切り替えが発生する可能性があります。

- PLAN1 (TRX0) はプログラム P0、P1、P2、および P3 からの DBRM を使用します。
- PLAN2 (TRXA) はプログラム PA、PB、PC、および PD からの DBRM を使用します。

ただし、プログラム P3 はプログラム PB に制御を移動できます (XCTL コマンドを使用して)。このとき、計画切り替え技法を使用する必要があります。

小規模な計画を使用する特殊な場合があります。アプリケーションによっては、次のトランザクションのために端末ユーザーがトランザクション ID を指定する方が便利な場合があります。これは、通常、読み取り専用アプリケーションの場合で、ユーザーが、システムで作成された 1 から 4 文字のトランザクション ID を入力することによって、多数の異なる通知パネルの中から選択できます。ユーザーの利点は、任意のパネルから他のパネルに、サブメニューの階層を通ることなくジャンプできる点です。

Db2 計画が各トランザクション ID に関連付けられている場合、アプリケーションは多数の小規模な計画になります。

利点

- 計画間のオーバーラップはわずかなので、計画保守は比較的簡単です。
- 計画名を示すメッセージや統計などの情報価値が高くなります。

注: パッケージにもこれらの利点があります。

欠点

- アプリケーション・フローが細いパスをたどる場合を除き、計画は頻繁に切り替えられます。
- トランザクションがトランザクション ID、計画、およびスレッドの多数のセットに分散しているため、保護スレッドの使用が困難です。
- 計画が切り替えられ、スレッドの再使用頻度が低いため、リソースの消費が高くなる可能性があります。

注: パッケージではこれらの欠点は避けられます。

トランザクションのグループ化に基づく計画の使用

トランザクションのグループ化によって、多数の中規模の独立した計画が作成され、必要に応じて計画切り替えが発生する場合があります。

グループ内のプログラムが密接に関連するようなプログラム・グループを定義できる場合がしばしばあります。つまり、それらは、同じトランザクション内で、または連続して実行される異なるトランザクション内でしばしば実行されることを意味します。グループごとに、1 つの別個の計画を使用する必要があります。

90 ページの図 26 の例では、2 つの計画が作成される可能性があります。

- PLAN1 (TRX0) はプログラム P0、P1、P2、および P3 からの DBRM を使用します。
- PLANA (TRXA) はプログラム PA、PB、PC、および PD からの DBRM を使用します。

ただし、プログラム P3 はプログラム PB に制御を移動できます。このとき、計画切り替え技法が使用される可能性があります。計画切り替えは例外であり、通常は使用しないことをお勧めします。これは、主に、プロセッサのオーバーヘッドが追加されるためです。

この場合、トランザクションのグループ化技法の結果は、多数の小規模な計画を使用する技法の結果と一致します。これは、使用している例が簡潔であるためです。通常は、トランザクションのグループ化技法では、大規模な計画が作成されます。

利点

- 計画サイズと、異なる計画の数をユーザーが制御できます。
- グループ内のトランザクション率によっては、スレッドの再使用の可能性があります。

欠点

- 計画がオーバーラップする可能性があります。
- 最適なグループ化は、時間の経過に伴って変化する可能性があります。
- 計画の切り替えが必要になることがあります。
- 計画の切り替えがアプリケーション・コードで処理されます。

動的計画出口

パッケージの利点を考慮し、動的計画出口ではなく、パッケージを使用するようお勧めします。動的計画出口の使用は、パッケージが Db2 で使用できるようになる前に、CICS Db2 環境で問題に対応するために設計された暫定ソリューションです。

多数の小さな計画用に CICS アプリケーションを設計し、実行時に計画を動的に選択することができます。小さな計画は、データベース要求モジュール (DBRM) に厳密に 1 対 1 で対応するパッケージと同じではありません。

通常、動的計画出口は、トランザクションの最初の作業単位 (UOW) の開始時に使用する計画を決定するために駆動されます。これは、動的計画の選択 と呼ばれます。

動的計画出口は、次の UOW で使用する計画を決定するために (スレッドが同期点で解放されたと想定)、後続の UOW の開始時に駆動することもできます。計画出口は異なる計画を使用するように決定できます。これは、動的計画切り替え と呼ばれます。

動的計画出口が使用されている場合、Db2 計画割り振りは、プログラム内の最初の SQL ステートメントの実行時にのみ、またはプログラムが同期点を発行し、別個の DBRM で別のプログラムに制御をリンクまたは転送した後にのみ、行われます。

これは、以下で指定された出口プログラムを使用して行われます。

- DB2ENTRY。キーワード PLANEXITNAME で指定された特定のトランザクション・コードの出口。
- DB2CONN。キーワード PLANEXITNAME で指定されたプールを使用するトランザクションの出口。

IBM は、ソースとオブジェクト・コード形式の両方で、サンプルとして 2 つのアセンブラー言語出口プログラム DSNCUEXT および DFHD2PXT を提供します。他の出口プログラムを作成することもできます。

動的出口プログラム内での以下の項目の使用はサポートされていないことに注意してください。

- SQL コマンド。
- IFI 呼び出し。
- EXEC CICS SYNCPOINT コマンド。

サンプル出口プログラム DSNCUEXT および DFHD2PXT

サンプルとして 2 つのアセンブラー言語出口プログラム DSNCUEXT および DFHD2PXT が、ソースとオブジェクト・コードの両方の形式で提供されています。これらのプログラムは機能的には同一ですが、DSNCUEXT に対する CICS 提供のプログラム定義は CONCURRENCY(QUASIRENT) を指定し、DFHD2PXT に対する CICS 提供のプログラム定義は CONCURRENCY(THREADSAFE) を指定します。

サンプル・プログラムが 2 つあるのは、CICS がオープン・トランザクション環境 (OTE) を活用する際に、CICS Db2 タスク関連ユーザー出口はスレッド・セーフ・プログラムとして作動し、オープン TCB (L8 モード) で制御を受け取ることができるためです。詳細については、108 ページの『スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにする』を参照してください。Db2 要求を行ったアプリケーション・プログラムがスレッド・セーフである場合は、オープン TCB でも実行できます。この場合、TCB 切り替えは必要ありません。

ただし、DSNCUEXT のように、CONCURRENCY(QUASIRENT) で定義されている動的計画出口を使用する場合、QR TCB にスイッチバックされ、追加のコストが発生します。DFHD2PXT のように、CONCURRENCY(THREADSAFE) で定義され、スレッド・セーフ標準に合わせてコーディングされている動的計画出口はオープン TCB で実行でき、追加のコストは発生しません。

したがって、CICS が Db2 に接続されている場合は、動的計画出口として、DSNCUEXT ではなく、DFHD2PXT を使用します。提供されるサンプル・プログラムにスレッド・セーフでないロジックを追加する場合、または出口で非スレッド・セーフの CICS コマンドを発行する場合、QR TCB にスイッチバックされ、追加のコストが発生することを覚えておってください。TCB 切り替えを回避してパフォーマンス上の利点を得るには、スレッド・セーフとして定義されている動的計画出口を使用し、スレッド・セーフである論理コードを使用し、プログラムにスレッド・セーフ・コマンドのみが含まれていることを確認します。

サンプル計画出口用のオブジェクト・コード (ロード・モジュール) は SDFHLOAD ライブラリーに含まれています。DSNCUEXT はデフォルトの動的計画出口であり、CICS ユーザー置換可能プログラムとして呼び出されます。両方のサンプル・プログラムのソース・コードはアセンブラー言語で作成され、SDFHSAMP ライブラリーで提供されます。サンプル・プログラムによってパラメーター・リストの使用方法が示されますが、計画名は変更されません。

パラメーター・リストは COMMAREA を介してサンプル・プログラムに渡されます。アセンブラー・バージョンでの形式は以下のとおりです。

表 4. COMMAREA を介して DSNCUEXT または DFHD2PXT サンプル・プログラムに渡されるパラメーター・リストの例：

アセンブラー・バージョン形式			
CPRMPLAN	DS	CL8	サンプル・プログラムへの最初の進入における SQL ステートメントの DBRM/計画名。新規計画を設定するためにフィールドを変更することができます。
CPRMAUTH	DS	CL8	サインオン時に Db2 に渡される現行の許可 ID。これは単なる通知メッセージです。変更が行われてもすべて無視されます。
CPRMUSER	DS	CL4	サンプル・プログラムで使用するために予約されているユーザー域。CICS Db2 接続では、サンプル・プログラムの呼び出し全体でこのフィールドが保持されます。
CPRMAPPL	DS	CL8	SQL 呼び出しを実行したアプリケーション・プログラムの名前。

- ・ パラメーター・リストのアセンブラー・バージョンは、SDFHMAC ライブラリー内のメンバー DSNCPRMA として付属しています。
- ・ COBOL バージョンは SDFHCOB ライブラリー内の DSNCPRMC です。
- ・ PL/I バージョンは SDFHPLI ライブラリー内の DSNCPRMP です。

動的計画出口を呼び出す前に、CICS Db2 接続機能により、作業単位で実行される最初の EXEC SQL ステートメントのパラメーター・リストに設定されている DBRM の名前に CPRMPLAN が設定されます。CICS によって提供される動的計画出口 DSNCUEXT および DFHD2PXT は CICS Db2 接続機能による CPRMPLAN の入力として計画名を変更せずに、即時に返すため、計画名は CICS Db2 接続機能によって選択された名前のままになります。

CICS 用の Java アプリケーションの JDBC および SQLJ サポートを追加した結果、CICS 提供の動的計画出口が変更されました。SQLJ および JDBC では、分離レベルの動的変更をサポートするためにアプリケーション・プログラムごとに 4 つの DBRM を生成する Db2 が必要です。JDBC および SQLJ アプリケーションの場合、DBRM 名は、8 番目の文字を 1、2、3、または 4 の接尾部として使用できるように、7 文字に制限されます。したがって、JDBC および SQLJ アプリケーションの場合、DBRM 名の接尾部が 1、2、3、または 4 になるため、*program name* = *dbrm name* = *plan name* というデフォルトの命名規則を使用することはできません。

JDBC アプリケーション、SQLJ アプリケーション、および JDBC と SQLJ の混合アプリケーションの場合、Db2 JDBC ドライバーは JDBC プロファイルからの情報を使用して、実行される最初の EXEC SQL ステートメントのパラメーター・リスト内の DBRM の名前を設定します。実行される最初の SQL では常に、DBRM 名はデフォルトの分離レベルが付加された JDBC 基本プログラムに設定されたものになります (つまり、デフォルトで DSNJDBC2 に設定されます)。例えば、*pgmname=OTHER* を使用して JDBC プロファイルが生成される場合、DBRM 名は

OTHER2 になります。 動的計画出口が使用されていない場合、計画名は DB2CONN または DB2ENTRY 定義から取得され、プロパティ・ファイル内の計画名は無視されます。

IBM Data Server Driver for JDBC and SQLJ のデフォルトの命名規則をサポートするために、CICS 提供の動的計画出口 DSNCEXT および DFHD2PXT は、最初の文字が SYSSTAT、SYSLH、または SYSLN である、入力された CPRMPLAN 名を検出できます。このような計画名が検出された場合、計画名は「DSNJCC」(7 番目と 8 番目の文字は空白に設定される) に変更されます。CICS 用の Java アプリケーションでデフォルトの動的計画出口を使用するには、DSNJCC という計画に複数の DBRM をバインドします。

ユーザー独自の出口プログラムの作成

独自の出口プログラムをアセンブラ、COBOL、または PL/I で作成することができます。

このタスクについて

出口プログラムは CICS プログラムです。つまり、以下を行う必要があります。

- 通常の CICS 規則に従う。
- CICS に定義する (プログラムの自動インストールが使用されていない場合)。
- CICS コマンド・レベル・ステートメントを使用する。
- EXEC CICS RETURN コマンドを使用して CICS に戻る。

CICS Db2 接続機能プログラムは、COMMAREA を使用して出口プログラムにパラメーター・リストを渡します。出口プログラムは、最初の SQL ステートメントが CICS Db2 接続機能により処理されたときに、パラメーター・リストに示されているデフォルトの計画名 (DBRM 名) を変更できます。この名前は、トランザクションのこの実行用の計画名を指定します。

CICS が DB2 バージョン 7 以降に接続されている場合は、108 ページの『スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにする』で説明されているように、動的計画出口のロジックがスレッド・セーフ標準に合わせてコーディングされており、プログラムが (サンプル出口プログラム DFHD2PXT のように) スレッド・セーフとして定義されていることを確認する必要があります。このプログラムがスレッド・セーフとして定義されていない場合は、プログラムが使用されるたびに QR TCB にスイッチバックされるので、追加のコストが発生します。このプログラムがスレッド・セーフとして定義されているものの、非スレッド・セーフ CICS コマンドを使用している場合は (これは許可されている)、個々の非スレッド・セーフ・コマンドが実行されるたびに QR TCB にスイッチバックされるので、追加のコストが発生します。

既に関係されているアプリケーション用に計画を作成する必要がある場合

Db2 計画の側面にほとんど注意を払わずにアプリケーションが開発された場合に、この技法を使用します。アプリケーションの開発が完了した後に、トランザクションと一致するように計画を定義します。

このタスクについて

通常、アプリケーションが開発された後に計画を定義することは推奨されませんが、この技法はアプリケーション設計を変更せずにアプリケーションで Db2 を使用することになった場合の移行プロジェクト で役立ちます。

この技法の利点と欠点は、実際のアプリケーション設計とステップの結果に完全に依存します。

手順

Db2 計画および DB2ENTRY 仕様を定義する際に、以下のステップを実行できます。

1. SQL 呼び出しを使用するプログラム・モジュールごとに、それらがどの CICS トランザクション・コードの下で実行されるかを分析します。特定のプログラム・モジュールが複数の CICS トランザクション・コードの下で使用されていると考えられます。

このステップの出力は、トランザクションごとの DBRM のリストです。

2. CICS トランザクション・コードごとに、使用する計画を決定します。1 つの CICS トランザクション・コードに対して DB2ENTRY で指定できる計画は 1 つのみです。複数のトランザクションで同じ計画を使用できます。

このステップでは、多くの選択肢があります。使用する計画の数は、1 からトランザクション ID の数までの間にすることができます。計画を管理する最良の方法は、パッケージを使用することであり、すべての DBRM をパッケージにバインドし、それらパッケージを計画にリストします。詳しくは、90 ページの『Db2 パッケージの使用』を参照してください。

パッケージを使用しない場合の代わりの技法については、94 ページの『すべてのトランザクションのために 1 つの大規模な計画を使用する』、95 ページの『多数の小規模な計画を使用する』、および 96 ページの『トランザクションのグループ化に基づく計画の使用』で説明しています。

例

90 ページの図 26 の例に適用した場合、考えられるソリューションは以下のとおりです。

- 1 つの計画 PLAN0 は、トランザクション ID TRX0 によって使用され、P0、P1、P2、P3、および PB からの DBRM を使用します。
- 1 つの計画 PLAN1 は、トランザクション ID TRXA によって使用され、PA、PB、PC、および PD からの DBRM を使用します。
- 2 つの DB2ENTRY 定義 (つまり計画ごとに 1 つ) を指定します。

トランザクション内で計画を切り替える必要がある場合

トランザクションが Db2 にアクセスするために使用する計画は、そのトランザクション ID の下で実行されるすべてのアプリケーション・プログラムからの DBRM を含むか、または参照する必要があります。計画にすべての関連 DBMR が含まれていない場合は、トランザクション内で計画を切り替えることを考慮できます。

アプリケーションの設計が変更された場合、いくつかの古い計画にはすべての関連 DBRM が含まれていない可能性があります。このケースでは、現行の計画に含まれていないプログラムの実行がトランザクションで必要な場合に、アプリケーション設計により、トランザクションの過程で別の計画への切り替えが必要とされることがあります。この状態は、アプリケーションの設計中に、Db2 計画の構造についての考慮が不完全であった場合にも発生する可能性があります。

この問題に対する最良の解決方法は、パッケージを使用することです。欠落しているプログラムからの DBRM をパッケージにバインドして、そのパッケージを既存の計画のパッケージ・リストに追加できます。これで、トランザクションは欠落しているプログラムにアクセスでき、計画を切り替える必要もありません。この他にも考えられる解決方法が 2 つあります。これについては、以下で説明します。

- 『動的計画切り替え』
- 103 ページの『計画を切り替えるためのトランザクション ID の切り替え』

動的計画切り替え

トランザクションで複数の計画を使用する場合は、動的計画切り替え (DPS) を使用します。

動的計画切り替え (DPS) により、1 つのトランザクションで複数の計画を使用できます。ただし、前述のように、CICS トランザクション・インスタンスではめったに計画切り替えは行われません。動的計画出口は、トランザクションの開始時に動的に計画を選択 (動的計画選択) するために設計されたものであり、トランザクション内で計画を頻繁に変更するためのものではありません。

動的計画切り替えを行うには、スレッドを同期点で解放し、再取得して動的計画出口を駆動する必要があります。その後、動的計画出口を使用して、実行する必要があるプログラムの計画を選択できます。これにより、トランザクション内のさまざまな UOW で異なる計画を使用できるようになります。

独自の動的計画出口をコーディングした場合は、ロジックによる同じタスクの後続の呼び出しの処理を確認してください。出口の追加呼び出しの結果を許容するには、ユーザー・アプリケーションまたは動的計画出口を作成する必要があります。動的計画出口が計画を不要なときに変更しようとしても、ユーザー・アプリケーションは、スレッドが同期点で解放されないようにすることで、これを回避できます。スレッドが解放されている場合、可能であれば、動的計画出口の呼び出し時の新しいケースのための適切な計画 (つまり、THREADLIMIT > 0 の DB2ENTRY) を備える必要があります。

UOW の終了後に計画切り替えを行うために動的計画出口を呼び出すには、トランザクションが同期点でスレッドを解放する必要があります。トランザクションが同期点でスレッドを解放するのは、以下の場合のみです。

- 端末主導タスク、または非端末主導タスクであり、NONTERMREL=YES が DB2CONN に設定されている。
- オープンしている保留カーソルがない。
- 変更されたすべての Db2 特殊レジスターが初期状態に戻されている。
- Db2 特殊レジスター CURRENT DEGREE がこのトランザクションによって変更されたことがある。

計画を切り替えるためのトランザクション ID の切り替え

トランザクションがいったん開始されると、トランザクション ID で使用される計画を制御することはできませんが、トランザクション ID 自体を制御することはできます。したがって、トランザクション ID を切り替えることで、計画を切り替えられます。ただし、これを行うことは推奨されていません。

このタスクについて

計画を切り替える場合、トランザクション ID を切り替えるのではなく、欠落しているプログラムからの DBRM をパッケージにバインドし、そのパッケージを既存の計画のパッケージ・リストに追加することをお勧めします。

トランザクション ID を切り替える必要があるときは、多くの場合、最初のプログラムが次のプログラムにデータを転送することに注意してください。これを行うための推奨される方法は、EXEC CICS RETURN IMMEDIATE コマンドを使用することです。あるいは、EXEC CICS START コマンドを使用するか、トリガー・レベルが 1 の一時データ・キューを使用して、同じ端末に対して新しい CICS タスクを開始できます。古いプログラムは、CICS に対して RETURN を発行し、新しいタスクを開始する必要があります。これらの切り替え手法の両方で、1 つのユーザー・トランザクションで行われる作業が複数の UOW に分割されます。新しいタスクがバックアウトされると、最初のタスクで行われた作業はコミットされたままになります。

計画を切り替えるためにトランザクション ID を切り替える場合、アプリケーション・プログラムにはトランザクション ID をいつ切り替えるかを決定するためのロジックが含まれます。これは、計画構造を (パフォーマンス上の理由や操作上の理由などで) 変更する場合は、アプリケーション・プログラムも変更する必要があることを意味します。

別の手法として、いくつかのプログラムでこれを行うためのコードを用意する代わりに、プログラム・フローをテーブルで制御する手法があります。主旨は、ロジックを実行する各アプリケーション・プログラムでコードを保守するより、テーブルで計画、プログラム、およびトランザクション ID の間の関係を維持するほうが簡単であるということです。標準インターフェースが提供されている場合は、アプリケーション・プログラムの開発もより簡単になります。

プログラム・フローを制御するための制御テーブルの使用

プログラム・フローを制御する場合は、この手法の代わりにパッケージを使用することをお勧めします。

示されている設計原理は、さまざまなタイプのアプリケーション設計を実装するために使用できる標準的な方法の例です。これにより、プログラムを変更せずに 1 つの大きな計画や多数の小さな計画などを使用できるようになります。

104 ページの表 5 に、制御テーブルの内容例を示します。この例は、90 ページの図 26 で説明されている設計シチュエーションに基づいています。

表 5. サンプル・アプリケーションの制御テーブル

関数名	プログラム	トランザクシ ョン	計画名	新規トランザク ション ID
	P0	TRX0	PLAN0	*
Sales	P1	TRX0	PLAN0	
Order	P2	TRX0	PLAN0	
Pay	P3	TRX0	PLAN0	
	PA	TRXA	PLANA	*
Price	PB	TRXA	PLANA	
Order	PC	TRXA	PLANA	
Parts	PD	TRXA	PLANA	
Price	PB	TEMP	PLANX	*

関数名

表の関数名フィールドはプログラム・フィールドを補足するものです。動作はまったく同じです。端末ユーザーがコマンド・フィールドに関数名を入力する場合に使用されます (最終的にはキーで指定)。PFCP プログラムは関数名またはプログラム名を受け入れることができます。

その後、PFCP は関数列を使用して、有効なトランザクションを検索します。

この方法で、ユーザーの選択項目を解釈するためのロジックもプログラムから除去され、保守が簡単なテーブルに配置されます。

プログラム

この行で説明されているプログラムの名前。

トランザクション

プログラム列のプログラムを実行できるトランザクション ID の名前。

計画名

計画名フィールドは使用されません。説明のみの目的で示されています。対応するトランザクションで使用される計画名を表示します。

新規トランザクション ID

テーブルのこの列の * は、所定のプログラムを開始するために新規トランザクション ID を検索するときに、対応する行を使用できることを意味しています。

制御テーブルには以下の主な関係が反映されています。

- Db2 カタログ・テーブル SYSIBM.SYSDBRM で説明されている、計画とプログラム/DBRM の間の関係
- DB2ENTRY および DB2TRAN CICS 定義を使用して説明されている、トランザクション・コードと計画の関係

制御テーブルはさまざまな方法で実装できます。最も有効な解決策はおそらく Db2 表または主記憶テーブルを使用することです。

Db2 表 は開発と保守が最も簡単です。 PFCP の呼び出しごとに 1 つまたは 2 つの SELECT 呼び出しが必要です。これらの SELECT では 1 行のみを返す必要があり、索引を使用することができます。データと索引ページは頻繁に参照され、おそらくバッファ・プールに保持されます。したがって、応答時間への影響は最小限に抑えられます。

主記憶テーブル の場合、少なくともテーブル内の一定数の行に達するまでは、アクセスがより速くなります。保守はより複雑になります。

プログラム・フローの原理は図 27 に示されています。

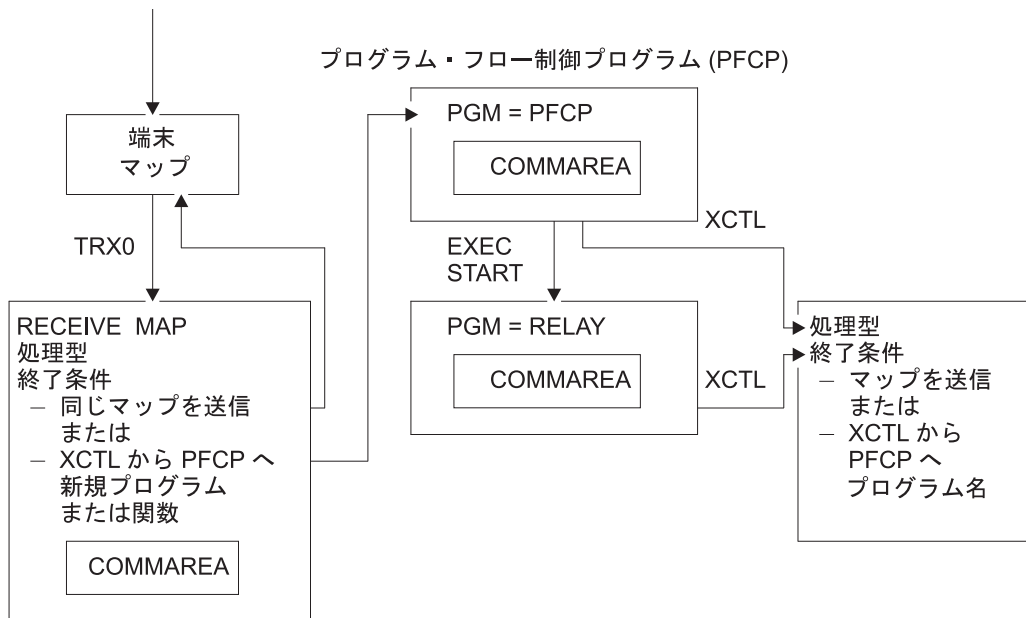


図 27. テーブル制御プログラム・フロー

90 ページの図 26 で使用されるアプリケーション設計のフローは、以下のとおりです。

1. 端末ユーザーはトランザクションを CICS に送信します。トランザクション ID は TRX0 です。
2. トランザクション定義はプログラム P0 を指します。
3. プログラム P0 はマップを受け取り、何らかの処理を行って、プログラム P3 が必要であると判断します。
4. P0 は、制御をプログラム P3 (P3 の DBRM は別の計画の一部である可能性があります) に転送する代わりに、プログラム・フロー制御プログラム (この例では PFCP) に転送します。P0 は COMMAREA も渡します。
5. PFCP は、制御テーブルでテーブル検索を行い、現在使用中の計画 (PLAN0) にプログラム P3 が含まれているかどうかを確認します。これは、現行のトランザクション ID (TRX0) と同じ行に P3 が指定されているかどうかを確認することで行われます。この例ではこれが該当します (テーブルの 4 行目)。
6. 次に、PFCP は制御をプログラム P3 に転送します。P0 から受け取った COMMAREA も P3 に渡します。

7. P3 は必要な SQL 呼び出しを処理し、マップを端末に送信するか、別のプログラムを実行するために制御を PFCP に転送して (図示なし)、作業を終了します。
8. この別のプログラムが PB であると仮定した場合、PFCP は、PB が現行のトランザクション ID (まだ TRX0 である) で実行できるかどうかを再度確認します。

テーブルには、PB を TRX0 で実行してはならないと示されています。したがって、PFCP はテーブルを調べ、PB を実行できるトランザクション ID を探します。例では、TRXA と TEMP の両方が有効なトランザクション ID です。ただし、TRXA はトランザクション定義内のプログラム PA を指しています。テーブルの「新規トランザクション ID」列には、* が示されている行のみが、所定のプログラムを開始するための新規トランザクション ID を検索するときに使用できることが示されています。この場合、TEMP トランザクションが該当します。

TEMP トランザクション ID の RDO トランザクション定義項目では、プログラム名として以下の 2 つが考えられます。

- RDO トランザクション定義は新規プログラム (PB) を直接指すことができます。この場合、この方法で開始できる各プログラムのトランザクション ID がなければなりません。また、COMMAREA を使用するには、開始されるプログラムに、そのプログラムが START で開始されるのか、XCTL から制御を取得して開始されるのかを確認するためのロジックが含まれていなければなりません。
- RDO トランザクション定義は共通プログラム (ここでは RELAY プログラムと呼ぶ) を指すことができます。この場合、1 つ以上のトランザクション ID を使用できます。それらはすべて、RDO トランザクション定義内の RELAY プログラムを指します。RELAY の目的は、適切なプログラムに制御を転送することです。その後、これらすべてのプログラムが START で開始されることはなく、この状態を処理する必要はありません。

RELAY プログラムによる解決策は 105 ページの図 27 に示されています。

9. PFCP はトランザクション TEMP を開始し、COMMAREA を渡します。
10. RELAY プログラムが開始されます。COMMAREA を取得するには、EXEC CICS RETRIEVE コマンドを使用する必要があります。
11. RELAY は、COMMAREA からプログラム名 PB を取得します。
12. RELAY は制御を PB に転送し、COMMAREA を渡します。
13. これで計画の切り替えは完了です。

利点

- この方法では、さまざまなタイプのアプリケーション設計 (1 つの大きな計画または多数の小さな計画を使用するなど) を実装できます。
- 計画をいつ切り替えるかの決定は開発プロセスとは別に行われ、コーディングの対象にはなりません。

- 新しいプログラムを実動に移す際に更新する必要があるのは制御テーブルのみです。既存のプログラムは、新しい関数を呼び出す場合でも変更する必要はありません。
- トランザクション ID、Db2 計画、およびプログラム間の関係を、プログラムを変更せずに変更することができます。ただし、制御テーブルは変更する必要があります。
- Db2 カタログ (SYSPLAN および SYSDBRM) からの情報を使用して、制御テーブルを作成することができます。
- あるいは、制御テーブルを使用して、計画の DBRM 構造に関する情報を生成できます。
- 制御テーブルには、CICS で DB2ENTRY および DB2TRAN を定義する際に役立つ情報が含まれています (計画名の列が使用可能な場合)。
- 他の関数 (EXEC CICS RETURN コマンドの TRANSID オプションで使用されるトランザクション ID に関する情報など) を制御テーブルの構造に組み込むことができます。

欠点

計画を切り替えるためのトランザクション ID の切り替えにはいくつかの欠点があります。

この手法の 2 つの主な欠点は、ソリューションの設計および開発にかかるコストと、実行時のオーバーヘッドです。

プログラム間の移行コストは約 2 倍になります。ただし、トランザクションのプロセッサ時間の増加率は、通常は数パーセントに過ぎません。このようなソリューションを使用するかどうかを決定する場合は、これらの欠点と利点とのバランスを考慮する必要があります。

CICS Db2 環境でのロック戦略の開発

Db2 はロック・メカニズムを使用して、データ保全性を維持しながら、並行性を可能にします。

CICS 環境では、並行性が高くなる可能性があります。最大並行性を指定するには、テーブル・スペース・ロックの代わりに、行レベル・ロックまたはページ・ロックを使用する必要があります。これは、テーブル・スペースの作成時に LOCKSIZE(PAGE)、LOCKSIZE(ROW)、または LOCKSIZE(ANY) を定義し、バインド時にカーソル固定として分離レベルを定義することで行うことができます。詳しくは、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

LOCKSIZE(ANY) を指定することで、Db2 は、テーブル・スペースに対してロック・エスカレーションを行えるかどうかを決定できます。Db2 パラメーター NUMLKTS は、テーブル・スペースに対する並行ロックの数です。ロックの数が NUMLKTS を超えると、ロック・エスカレーションが行われます。そのため、ロック・エスカレーションが通常の CICS 操作に対して行われないように、NUMLKTS を十分に高い値に設定する必要があります。

テーブル・スペース・ロックが行われ、計画が `RELEASE(DEALLOCATE)` でバインドされた場合、ページ・ロックのみが解放されるため、テーブル・スペースはコミット時に解放されません。これは、スレッドと計画がテーブル・スペースを独占して使用していることを意味する場合があります。

`PAGE` の代わりに `ANY` を使用すると、Db2 には、コミットする前に多くのページ・ロックを必要とするプログラムに対してロック・エスカレーションを使用するというオプションが生じます。通常、これはバッチ・プログラムの場合です。Db2 では、ページやテーブル・スペース・レベルではなく、行レベルでロックすることもできます。したがって、細分性が向上し、ロック競合が削減されます。

初期ロック属性を選択するための Db2 規則をオーバーライドできます。これは、アプリケーション・プログラムで `SQL ステートメント LOCK TABLE` を使用して行います。ただし、厳密には必要でない場合は、`LOCK TABLE` ステートメントの使用は避けてください。`LOCK TABLE` ステートメントをオンライン・プログラムで使用すれば、`RELEASE(DEALLOCATE)` と保護スレッドの使用を避けることができます。`LOCK TABLE` ステートメントを使用する場合は、計画でバインド・オプション `RELEASE(COMMIT)` を使用してください。

一般的には、以下のようにするために、CICS プログラムを設計することをお勧めします。

- ロックの保持期間をできるだけ短くする。
- 並行ロックの数を最小化する。
- テーブルのアクセス順序をすべてのトランザクションに対して同じにする。
- テーブル内の行のアクセス順序をすべてのトランザクションに対して同じにする。

CICS Db2 アプリケーションに関する SQL、スレッド・セーフ、その他のプログラミング考慮事項

アプリケーション・プログラムを開発するときには、アプリケーションのパフォーマンスを向上させ、潜在的な問題を回避するための特定のプログラミング考慮事項に注意する必要があります。修飾 SQL と非修飾 SQL のどちらを使用するかで、CICS Db2 環境の他の多くの面が影響を受けます。オープン・トランザクション環境 (OTE) のパフォーマンス上の利点を得るには、アプリケーション・プログラムがスレッド・セーフである必要があります。

スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにする

CICS Db2 接続機能には、CICS Db2 タスク関連ユーザー出口 (`TRUE`)、`DFHD2EX1` が含まれています。この出口は、アプリケーション・プログラムが SQL 要求を行うときに呼び出されます。これは Db2 へのスレッド接続の獲得プロセス、および Db2 処理の完了時にアプリケーション・プログラムに制御を戻すプロセスを管理します。

このタスクについて

CICS Db2 接続機能では、OTE を使用することによって、TCB を切り替えることなく CICS Db2 TRUE を呼び出し、Db2 から戻ることができます。OTE では、CICS Db2 TRUE はスレッド・セーフのオープン API TRUE プログラム (接続処理中に ENABLE PROGRAM コマンドで OPENAPI オプションを使用して自動的に使用可能になる) として作動します。これにより、この出口はオープン L8 モード TCB で制御を受け取ることができます。Db2 に対する要求も L8 TCB で発行されます。したがって、これはスレッド TCB として動作するため、サブタスク TCB に切り替える必要がありません。

OTE では、TRUE を呼び出したユーザー・アプリケーション・プログラムがスレッド・セーフ・コーディング規則に準拠し、CICS にスレッド・セーフとして定義されている場合、このプログラムは L8 TCB でも実行できます。最初の SQL 要求の前は、アプリケーション・プログラムは CICS のメイン TCB である QR TCB で実行されます。SQL 要求を行って TRUE を呼び出すと、制御が L8 TCB に渡され、Db2 の処理が実行されます。Db2 から戻る際、アプリケーション・プログラムがスレッド・セーフである場合は、引き続き L8 TCB で実行されます。

CONCURRENCY(REQUIRED) で定義されたプログラムは、プログラムの開始時からオープン TCB で実行されます。CICSAPI プログラムの場合、CICS はプログラムの実行キーに関係なく、L8 オープン TCB を使用します。OPENAPI プログラムの場合、CICS は EXECKEY(USER) が設定されるときは L9 TCB を使用し、EXECKEY(CICS) が設定されるときは L8 TCB を使用します。

適切な条件が満たされる場所では、CICS Db2 アプリケーションに対してオープン TCB を使用することで、QR TCB の使用が減り、TCB の切り替えが回避されます。OTE における理想的な CICS Db2 アプリケーション・プログラムは、スレッド・セーフ EXEC CICS コマンドのみを含み、スレッド・セーフ・ユーザー出口プログラムのみを使用するスレッド・セーフ・プログラムです。このようなアプリケーションは、最初に SQL 要求を出した際に L8 TCB に移動し、その後は TCB の切り替えを必要とせずに、すべての Db2 要求およびアプリケーション・コードを L8 TCB 上で実行し続けます。このような状態のためにアプリケーション・プログラムは複数の SQL 呼び出しを発行することができ、大きなパフォーマンスの向上につながります。アプリケーション・プログラムがあまり多くの SQL 呼び出しを発行しないという場合は、パフォーマンス上の利益があまり多くならないこともあります。

プログラムの実行がスレッド・セーフでないアクションに関係している場合、CICS はその時点で QR TCB にスイッチバックします。このようなアクションに該当するのは、プログラムがスレッド・セーフではない CICS 要求を発行すること、スレッド・セーフではない動的計画出口を使用すること、スレッド・セーフではない TRUE を使用すること、およびスレッド・セーフではないグローバル・ユーザー出口 (GLUE) を関与させることです。オープン TCB と QR TCB の間での切り替えは、アプリケーションのパフォーマンスに悪影響を及ぼします。

CICS Db2 アプリケーションに対する OTE のパフォーマンス上の利点を得るには、以下の条件を満たす必要があります。

- CICS が、サポートされているバージョンの Db2 に接続すること。サポートされている Db2 リリースおよび適用する必要がある APAR について詳しくは、<http://www.ibm.com/support/docview.wss?uid=swg27020857>を参照してください。
- システム初期設定パラメーター **FORCEQR** が YES に設定されていないこと。
FORCEQR を使用すると、スレッド・セーフとして定義されたプログラムが強制的に QR TCB 上で実行されます。これは、スレッド・セーフとして定義されたプログラムに関連する問題を調査して解決している間、一時的な手段として YES に設定される場合があります。
- CICS Db2 アプリケーションはスレッド・セーフ・アプリケーション論理を持ち(つまり、EXEC CICS コマンドの間のネイティブ言語コードがスレッド・セーフでなければならない)、スレッド・セーフ EXEC CICS コマンドのみを使用し、CICS に対してスレッド・セーフとして定義される必要があります。スレッド・セーフであると識別されたコードのみが、オープン TCB での実行を許可されます。ご使用の CICS Db2 アプリケーションがスレッド・セーフとして定義されていない場合や、スレッド・セーフでない EXEC CICS コマンドを使用する場合には、TCB の切り替えが生じ、OTE の活用によって得られるパフォーマンス上の利点の一部またはすべてが失われます。
- CICS Db2 接続機能で使用される動的計画出口が、スレッド・セーフ標準に従ってコーディングされていて、CICS にスレッド・セーフとして定義されていること。デフォルトの動的計画出口 DSNCEXT (CICS ユーザー置換可能プログラムとして開始される) は、CICS にスレッド・セーフとして定義されていませんが、CICS 提供の代替サンプル動的計画出口 DFHD2PXT はスレッド・セーフとして定義されています。詳しくは、97 ページの『動的計画出口』を参照してください。
- アプリケーションが使用する実行パス上の GLUE が、スレッド・セーフ標準に従ってコーディングされていて、CICS にスレッド・セーフとして定義されていること (CICS Db2 アプリケーションの場合も。具体的には、XRMIIN と XRMIOU の 2 つの GLUE)。
- アプリケーションで使用されるその他の TRUE は、CICS に対してスレッド・セーフ、または OPENAPI として定義されなければなりません。

アプリケーション・プログラムとユーザー出口プログラムをスレッド・セーフにする方法については、「スレッド・セーフ・プログラム」を参照してください。CICS に対してプログラムをスレッド・セーフとして定義することは、アプリケーション・ロジックがスレッド・セーフであることを指定しているに過ぎず、プログラムに含まれているすべての EXEC CICS コマンドがスレッド・セーフであることを指定しているわけではありません。CICS は、まだ変換されておらず、引き続き準再入可能性に依存している EXEC CICS コマンドについて、QR TCB に切り替えることでそれらのコマンドを安全に処理できるようにします。プログラムをオープン TCB 上で実行できるようにするために、アプリケーション・ロジックがスレッド・セーフであることを CICS に保証することが必要です。

スレッド・セーフであり、TCB 切り替えを伴わない EXEC CICS コマンドは、API および SPI コマンドの説明に記載されているコマンド構文図に示されています。

OTE のユーザー・アプリケーション・プログラムがスレッド・セーフとして定義されていない場合、CICS Db2 TRUE は引き続き L8 TCB で実行されますが、アプ

リケーション・プログラムはタスク全体を通して QR TCB で実行されます。プログラムが SQL 要求を行うたびに、CICS は QR TCB から L8 TCB への切り替えと、そのスイッチバックを行うため、OTE のパフォーマンス上の利益はなくなります。CICS Db2 アプリケーションで TCB 切り替えの発生が最大になるのは、Db2 要求が行われるたびに、プログラムが非スレッド・セーフ・ユーザー出口プログラムと非スレッド・セーフ EXEC CICS コマンドを使用した場合です。特に、CICS-Db2 メインライン・パスで非スレッド・セーフ出口プログラム (例えば、XRMIIN または XRMIOU で使用可能になるプログラム) を使用すると、CICS が旧バージョンの Db2 に接続されている場合よりも、多くの TCB 切り替えが発生します。

CICS が Db2 の異なるバージョンに接続されている場合、アプリケーション・プログラムによって呼び出される CICS Db2 TRUE の動作が、アプリケーションに設定された並行性属性の違いによって、どのように影響されるのかを表で説明します。

表 6. アプリケーション・プログラムと CICS Db2 TRUE の組み合わせ

プログラムの並行性属性	CICS Db2 TRUE の操作	効果
QUASIRENT	スレッド・セーフおよびオープン API	アプリケーション・プログラムは CICS QR TCB の下で実行されます。TRUE は L8 TCB で実行され、Db2 要求は L8 TCB で実行されます。CICS は、Db2 要求ごとに CICS QR TCB と L8 TCB 間の切り替えを行います。
THREADSAFE	スレッド・セーフおよびオープン API	OTE の活用。TRUE は L8 TCB で実行され、Db2 要求は L8 TCB で実行されます。制御がアプリケーション・プログラムに戻ると、アプリケーション・プログラムも L8 TCB で稼働します。TCB 切り替えは、タスクが終了するまで必要になりません。あるいは、タスクが非スレッド・セーフな CICS 要求を発行した場合には、QR TCB へのスイッチバックが強制されます。
REQUIRED および API(CICSAPI)	スレッド・セーフおよびオープン API	OTE の活用。TRUE は L8 TCB で実行され、Db2 要求は L8 TCB で実行されます。アプリケーション・プログラムは始めから L8 TCB で実行されます。プログラムの実行キーに関わらず、プログラムは常に L8 TCB を使用します。TCB 切り替えは、タスクが終了するまで必要になりません。あるいは、タスクが非スレッド・セーフな CICS 要求を発行した場合には、QR TCB へのスイッチバックが強制された後、再び L8 TCB にスイッチバックします。

表 6. アプリケーション・プログラムと CICS Db2 TRUE の組み合わせ (続き)

プログラムの並行性属性	CICS Db2 TRUE の操作	効果
REQUIRED および API(OPENAPI)	スレッド・セーフおよびオープン API	OTE の活用。ユーザー・キー CICS-Db2 アプリケーション (およびストレージ保護がアクティブの場合) には推奨されません。この設定では、Db2 要求が行われるたびに、L9 TCB から L8 TCB に切り替えられ、その後でスイッチバックが行われるためです。

要約すると、OTE のパフォーマンス上の利点を得るには、以下の条件が満たされている必要があります。

- CICS が Db2 に接続されていること。
- FORCEQR が YES に設定されていないこと。
- CICS Db2 アプリケーションにスレッド・セーフ・アプリケーション論理がなければなりません (つまり、EXEC CICS コマンドの間のネイティブ言語コードがスレッド・セーフでなければならない)。アプリケーション・ロジックがスレッド・セーフでない場合は、プログラムを CONCURRENCY(QUASIRENT) として定義し、CICS QR TCB で実行されるようにする必要があります。この場合、TRUE がオープン TCB で実行されているとしても、Db2 要求が行われるたびに TCB 切り替えが発生します。
- スレッド・セーフ・アプリケーションを CICS に対して CONCURRENCY (THREADS SAFE) API(CICS API) または CONCURRENCY(REQUIRED) API(CICS API) として定義できます。使用する設定は、プログラムが使用する非スレッド・セーフ EXEC コマンドの数によって異なります。多数の非スレッド・セーフ CICS コマンドがある場合には、プログラムを CONCURRENCY (THREADS SAFE) として定義することが最善です。プログラムで使用する非スレッド・セーフ CICS コマンドが少数であるか全くない場合は、CONCURRENCY(REQUIRED) を使用できます。CONCURRENCY (REQUIRED) で定義されたプログラムは、L8 オープン TCB で開始されるという利点がありますが、非スレッド・セーフな CICS コマンドが発行されるたびに、2 回の TCB 切り替えが発生します。
- CICS Db2 アプリケーションが、スレッド・セーフまたはオープン API 動的計画出口 (TRUE および GLUE) のみを使用すること。非スレッド・セーフ出口を使用すると、QR TCB へのスイッチバックが強制的に行われます。

以上のすべての条件が満たされていれば、OTE のパフォーマンス上の利点を得られます。

SQL 言語

CICS プログラマーは、少しの制限のみで完全な SQL 言語を使用できます。

CICS プログラム内での SQL 言語の使用について詳しくは、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」を参照してください。

CICS プログラムでは、以下を使用できます。

- データ操作言語 (DML)
- データ記述言語 (DDL)
- GRANT ステートメントおよび REVOKE ステートメント

また、CICS では動的 SQL ステートメントと静的 SQL ステートメントの両方がサポートされます。

ただし、パフォーマンスおよび並行性の理由から、通常は、CICS で DDL と GRANT ステートメントおよび REVOKE ステートメントを発行しないことをお勧めします。動的 SQL の使用も制限する必要があります。

これらのことを推奨する理由は、Db2 カタログ・ページがロックされて、その結果、並行性レベルが低下する可能性があるためです。また、これらのタイプの SQL ステートメントのリソース消費量は通常、静的 DML SQL ステートメントのリソース消費量よりも多くなります。

修飾および非修飾 SQL の使用

CICS Db2 プログラムを作成するプログラマーは修飾または非修飾 SQL を使用できます。修飾 SQL では、作成者がテーブルまたはビュー名の前に指定されます。非修飾 SQL では、作成者は指定されません。

プログラマーが CICS Db2 標準を開発する場合、修飾と非修飾のどちらの SQL を使用するかを決定することは重要です。この決定は、Db2 環境の他の多くの面に影響します。他の Db2 領域との主な関係および 2 つのタイプの SQL ステートメントのいくつかの結果が表 7 に示されています。

表 7. 修飾および非修飾 SQL

他の Db2 領域との関係	修飾 SQL	非修飾 SQL
シノニムの使用	不可	可
バインダー ID	任意	作成者と同じ
テーブルとテーブル・スペースに対する作成者の数	任意	1
VALIDATE(RUN) の使用	修飾。	バインダーを使用して修飾
動的 SQL の使用	修飾。	実行プログラムを使用して修飾
個別のテスト Db2 サブシステムが必要	はい	いいえ
テスト Db2 および実動 Db2 で同じ作成者が必要	はい	いいえ
同じテスト Db2 サブシステムで複数のバージョンのテスト・テーブルを使用する可能性	いいえ	はい

Db2 プリコンパイラを呼び出す前にソース・コードを変更するために独自のプリプロセッサを開発する場合は、表 7 に示されている制限の一部を迂回できます。これにより、SQL ステートメントの作成者の変更などが可能になります。

動的 SQL ステートメントでは、管理が簡単になるため、修飾 SQL を使用することをお勧めします。

非修飾 SQL を使用する場合は、テーブルやビューを完全に識別するための CREATOR の指定方法を決定する必要があります。その場合、以下の 2 つの方法が考えられます。

- シノニムを使用できます。シノニムは、DB2ENTRY および DB2CONN で指定された許可 ID で作成する必要があります。シノニムは許可 ID 自体でのみ作成可能です。これは、シノニムを作成するメソッドを開発する必要があることを意味します。DB2ENTRY または DB2CONN で指定された許可 ID と同じ ID を持つ TSO ID を使用できます。別の可能性は、それ自体が CREATE SYNONYM ステートメントを実行できる CICS トランザクション ID (同じ許可 ID を使用) を設計することです。ただし、これらの方法はいずれも推奨されません。
- シノニムを使用しない場合、バインド処理で使用する CREATOR がバインダーの許可 ID となります。そのため、動的 SQL で参照されるすべてのテーブルとビューをこの ID で作成する必要があります。Db2 リソースの共通セットへのアクセスに動的 SQL を使用するすべてのトランザクションには、DB2ENTRY または DB2CONN で指定された同じ許可 ID が必要です。ほとんどの場合、これは SIGNID、または文字ストリングでなければなりません。この制約は、通常は受け入れられません。

このような理由から、動的 SQL ステートメントでの非修飾 SQL の使用は推奨されません。

ビュー

一般的に、適切な場面でビューを使用することをお勧めします。一部のビューは更新できないことに注意してください。

リアルタイムのオンライン・システムでは、しばしば、ビューを使用して取得した行を更新する必要があります。ビュー更新の制約事項により、ベース・テーブルを直接 (または別のビューを使用して) 更新することが強制される場合、更新可能なビューについてのみ考慮してください。ほとんどの場合、これによって、プログラムの読み取りと変更が容易になります。

索引列の更新

索引列を更新するときは、2 つのことを考慮する必要があります。

- テーブル内のフィールドを更新するときは、Db2 はこのフィールドを含む索引を使用しないで行を受け取ります。これには、DECLARE CURSOR ステートメントの FOR UPDATE OF リストにリストされているフィールドが含まれます。フィールドが更新されるかどうかは、無関係です。
- 現在区画化されていないテーブル・スペースは、複数の区画を使用して再作成できます。SQL 更新は、区画化キー・フィールドに対して許可されていません。つまり、これらのフィールドを更新するプログラムは、代わりに DELETE ステートメントと INSERT ステートメントを使用するように変更する必要があります。

固有索引の依存関係

SELECT ステートメントで完全なキーが提供される場合、プログラマーは、Db2 が固有索引を使用してテーブルから 1 行のみを返すという事実を利用できます。この場合、カーソルは不要です。

索引が変更されて固有性がなくなった場合に、2 行以上が返されると、プログラムは正常に実行されません。プログラムは SQL エラー・コードを受け取ります。

コミット処理

CICS では、アプリケーション・プログラム内の EXEC SQL COMMIT ステートメントは無視されます。Db2 コミットは CICS と同期が取られる必要があります。つまり、プログラムで EXEC CICS SYNCPOINT コマンドを発行する必要があります。そうすると、CICS が Db2 を使用してコミット処理を実行します。タスクの終了時に EXEC CICS RETURN によって暗黙的な SYNCPOINT が常に呼び出されます。

SYNCPOINT で発生する以下のアクションに注意してください。

- UOW が完了します。つまり、すべての更新が CICS と Db2 の両方でコミットされることを意味します。
- スレッドは端末向けトランザクション用に解放されます (保留カーソルが開いている場合を除く)。スレッドが解放されて使用されない場合、それが保護スレッドでない限り終了します。
- スレッドは非端末向けトランザクション用には解放されません。ただし、DB2CONN で NONTERMREL=YES が指定されている場合を除きます。これは、トランザクションが終了したときに最初に発生します。
- すべてのオープン・カーソルが閉じられます。
- すべてのページ・ロックが解放されます。
- BIND 処理で RELEASE(COMMIT) が指定されている場合、以下のようになります。
 - テーブル・スペース・ロックが解放されます。
 - EDM プール内の計画のカーソル・テーブル・セグメントが解放されます。
- 動的 SQL により取得されたテーブル・スペース・ロックが BIND パラメーターとは関係なく解放されます。

トランザクションの直列化

1 つ以上のトランザクションの実行を直列化する必要があるかもしれません。通常、これが必要になるのは、並行性を扱うようにアプリケーション・ロジックが設計されておらず、デッドロックのリスクが高い場合です。

このタスクについて

直列化は、キューイング時間の可能性から、小規模のトランザクションでのみ許可する必要があります。

以下の方式では、直列化の開始および終了の時点がそれぞれ異なります。

CICS トランザクション・クラス

1 つの CLASS で一度に 1 つのトランザクションのみを実行するという CICS 機能は、完全なトランザクションを直列化するために役立ちます。

Db2 スレッドの直列化

直列化が最初の SQL 呼び出しから同期点までの間隔に限られている場合 (端末向けトランザクション、および NONTERMREL=YES が定義されている場合の非端末向けトランザクション)、DB2ENTRY 仕様を使用して、一度に特定タイプのスレッド 1 つのみが作成されるようにできます。この技法では、トランザクションの最初の部分の並行性が許可されます。この技法は、最初の SQL 呼び出しがトランザクションの開始部分にない場合に役立ちます。トランザクションが最初の SQL ステートメントを発行する前に、他のリソースを更新している場合は、この技法を使用しないでください。

CICS エンキューおよびデキュー

必要な直列化期間がプログラムのわずかな部分である場合、CICS のエンキューおよびデキューの技法は役立ちます。利点は、トランザクションの重要な部分のみを直列化する点です。この部分は、SQL ステートメント 1 つにまで小さくすることができます。これを使用すると、直列化を最小限に抑えられるため、他の方式に比べて高いトランザクション率を得ることができます。

他の技法と比べた場合の欠点は、直列化がアプリケーション・コード内で行われ、プログラムを変更する必要がある点です。

LOCK TABLE ステートメント

LOCK TABLE ステートメントは使用しない ことをお勧めします。

EXCLUSIVE モードが指定されている場合、LOCK TABLE ステートメントは、CICS トランザクションと他のプログラムを直列化するために使用されることがあります。ロックされるのは、ステートメントで参照されるテーブルではなく、テーブル・スペース全体であることに注意してください。

LOCK ステートメントが実行されると、直列化が開始されます。直列化の終了時刻は、テーブル・スペース・ロックが解放されるときです。これは同期点またはスレッドの割り振り解除時になる場合があります。

スレッドの割り振り解除時までテーブル・スペースがロックされるリスクがあるため、この技法の使用には注意してください。ただし、この技法は、完全な Db2 システム全体にわたって機能する唯一の技法です。他の技法は、CICS トランザクションのみの直列化制御に限られます。

ページの競合

アプリケーションおよびデータベースを設計するときには、多数のトランザクションがテーブル・スペースの同じ部分にアクセスする際の影響について考慮してください。用語「ホット・スポット」は、アクセス密度がテーブル・スペースの他の部分のアクセス密度に比べて非常に高くなっている、テーブル・スペースの小さな部分を表すためにしばしば使用されます。

ページが SELECT 処理にのみ使用される場合、並行性の問題は発生しません。ページはバッファ・プール内に留まる傾向があるため、入出力アクティビティはわずかしかな行われません。ただし、ページが頻繁に更新される場合、最初の更新から

同期点に至るまでページがロックされるため、並行性の問題が発生する可能性があります。同じページを使用する他のトランザクションは待機する必要があります。ホット・スポットに関連して、デッドロックおよびタイムアウトがしばしば発生します。

ホット・スポットの 2 つの例は、順序番号の割り振りと挿入の順序に関するものです。

連番の割り振り

1 つ以上のカウンターを使用して、アプリケーションに新しい連番を付ける場合は、以下の点を考慮してください。

- 各カウンターの更新頻度を計算する必要があります。また、カウンターの更新からコミットまで測定される、更新トランザクションの経過時間も計算する必要があります。 $(\text{更新頻度}) \times (\text{計算された経過時間})$ がピーク時に約 0.5 を超えた場合、キュー時間が受け入れられなくなります。
- 同じ表スペース内での複数のカウンターの使用を検討している場合は、カウンターの総ビジー時間を計算する必要があります。
- 複数のカウンターが同じ行に配置されている場合は、それらは常にまとめてロックされます。
- 複数のカウンターが同じ表スペース内の別の行に配置されている場合、それらは同じページにある可能性があります。ロックはページ・レベルで取得されるため、この場合、行も一緒にロックされます。
- 行が同じ表スペースの別のページに強制的に配置された場合 (フリー・スペースを 99% にするなどして)、トランザクションがキューに入れられる可能性はまだあります。

例えば、2 ページ目の 2 行目にアクセスすると、表スペース・スキャンが行われる場合があります。このページが別のトランザクションでロックされている場合、スキャンは停止し、1 ページ目で待機します。したがって、表スペース・スキャンを避ける必要があります。

- 索引が表スペース・スキャンを避けるように定義されている場合は、使用できるかどうかは不明です。表スペース内のページ数が少ない場合は、索引は使用されません。
- この場合の解決策は、各表スペース内のカウンターを 1 つだけにすることです。複数の CICS システムがカウンターにアクセスする場合は、この解決策をお勧めします。
- カウンターにアクセスする CICS システムが 1 つだけの場合は、BDAM ファイルを代替解決策として使用できます。ただし、後で CICS システムを 2 つ以上の CICS システムに分割する可能性がある場合は、この解決策の有効性が低くなる可能性があります。

順次挿入

多数のトランザクションが同じ表スペース内に行を挿入する場合、挿入される行の順番を考慮してください。

タイム・スタンプまたは連番の付いたフィールドに基づいてクラスタリング索引を行う場合、Db2 は相互に隣接したすべての行を挿入しようとします。そのため、行が挿入されるページをホット・スポットと見なすことができます。

クラスタリング索引では、一定期間内に、同じページですべての挿入も行われることに注意してください。

複数の索引がある状態で、非クラスタリング索引をデータ検索で使用する場合、索引とデータ間のデッドロックのリスクが増加します。一般的には、INSERT は以下の順序で X ロック (排他ロック) を取得します。

1. クラスタリング索引のリーフ・ページ
2. データ・ページ
3. 非クラスタリング索引のリーフ・ページ

SELECT ステートメントが非クラスタリング索引を使用すると、S ロック (共用ロック) が以下の順序で取得されます。

1. 非クラスタリング索引のリーフ・ページ
2. データ・ページ

これは、INSERT ロックの順序と逆の順序です。多くの場合、新しい行の SELECT 率が高くなります。これは、データ・ページが INSERT および SELECT ステートメントで共通であることを意味します。索引ページも同じである場合、デッドロックが発生する可能性があります。

デッドロック・リスクに対する解決策は、別の索引をクラスタリングとして選択し、行を分散することです。

デッドロック状態の一般的な処理方法については、CICS Db2 環境でのデッドロックの処理を参照してください。

CICS と CURSOR WITH HOLD オプション

CICS プログラムで CURSOR WITH HOLD を使用すると、SYNCPOINT 処理中にカーソルを一定の位置で開いたままの状態にします。

CICS プログラムの CURSOR 宣言で WITH HOLD オプションを使用すると、SYNCPOINT 中に以下の作用が生じます。

- カーソルが開いたままになる。
- 取得された最後の行と結果テーブルの次の行の間にカーソルが置かれたままになる。
- 動的 SQL ステートメントが引き続き準備された状態になる。

カーソルの位置を維持するために必要なロック以外のすべてのロックが解除されます。排他的ページ・ロックは、共用ロックにダウングレードされます。

会話型 CICS アプリケーションでは、DECLARE CURSOR...WITH HOLD を使用して、カーソルが同期点時に閉じられないように要求できます。ただし、すべてのカーソルは、タスクの終了 (EOT) 時および SYNCPOINT ROLLBACK では常に閉じられます。すべての EOT 時に、WITH HOLD で宣言されたカーソルを

WITH HOLD オプションが指定されなかった場合と同様に、再オープンして位置を変更する必要があります。保留カーソルの有効範囲は単一のタスクです。

要約すると、以下のようになります。

- 同期点後の次の FETCH は、同じタスクから行われなければなりません。
- タスクの終了をまたいでカーソルを保留することはできません。
- したがって、カーソルは疑似会話型トランザクションの EOT 部分をまたいで保留されません。

EOT をまたいでカーソルを保留しようとする、カーソルは閉じられ、次の FETCH の実行時に SQLCODE -501 を受け取ります。これはプリコンパイラーでは検出できないので、この状態を通知する警告メッセージは表示されません。

一般的には、スレッドが各同期点での再使用候補になることができます。CICS アプリケーションで DECLARE CURSOR...WITH HOLD を使用する場合は、以下の推奨事項を考慮してください。

- 保留カーソルは、不要になったらすぐに閉じる。すべての保留カーソルを閉じると、同期点で再使用するためにスレッドを解放することができます。
- 保留カーソルは、EOT の前に必ず閉じる。保留カーソルを閉じないと、CICS Db2 接続機能により、サインオン時にスレッドが強制的に初期状態に復元されるため、プロセッサ時間が長くなります。

EXEC CICS RETURN IMMEDIATE コマンド

TRANSID オプションが IMMEDIATE オプションと共に指定されている場合、CICS は、RETURN コマンドを発行したトランザクションの終了時にブラケット終了 (EB) が端末に送信されないようにして、TRANSID オプションで指定されたトランザクションを即時に開始します。EB が端末に送信されないため、このトランザクション中はキーボードがロックされたままになります。

新規トランザクションは、端末からの入力によって開始された場合と同様に動作します。COMMAREA を使用して、TRANSID オプションで指定されたトランザクションにデータを渡すことができます。選択によっては、RETURN コマンドを発行するトランザクションで、INPUTMSG および INPUTMSGLEN オプションを使用して端末入力メッセージを渡すこともできます。この機能により、端末入力の結果として開始される必要があるトランザクションを即時に開始できます。

この機能では、TERMID(...) が EIBTRMID 値に設定された EXEC CICS START TRANSID(...) を発行することで行われる機能と同じ一般機能が提供されますが、オーバーヘッドはかなり少なく、一時的にキーボードがアンロックされることもあります。EXEC CICS RETURN TRANSID() IMMEDIATE コマンドにより、疑似会話型トランザクションでのトランザクション・コードの切り替えが許可されます。これは、Db2 計画のサイズをより小さく維持する場合や、チャージバック目的のアカウンティング統計を改善する場合などにお勧めします。

AEY9 異常終了の回避

AEY9 異常終了は、CICS Db2 接続機能が有効になっていない状態で、アプリケーションが EXEC SQL コマンドを発行した場合に発生します。

このタスクについて

以下の CICS コマンドを使用して、CICS Db2 接続機能が有効かどうかを検出することができます。

```
EXEC CICS EXTRACT EXIT PROGRAM('DFHD2EX1')
ENTRY('DSNCSQL')
GASET(name1)
GALENGTH(name2)
```

プログラム名として DSNCEXT1 または DSN2EXT1 を指定すると、CICS はそれを必要な名前である DFHD2EX1 に動的に変更します。INVEXITREQ 状態が発生した場合、CICS Db2 接続機能が有効ではありません。

CICS Db2 接続機能が有効な場合でも、Db2 に必ずしも接続されているわけではありません。Db2 の初期化を待機している可能性があります。このような状態で、CONNECTERROR=ABEND が DB2CONN に指定されている場合にアプリケーションが EXEC SQL コマンドを発行すると、AEY9 異常終了になることがあります。CONNECTERROR=SQLCODE が指定されている場合は、-923 SQL コードがアプリケーションに返されます。

CICS が Db2 に接続されているかどうかを判別するためには、EXTRACT EXIT コマンドの代わりに、CONNECTST キーワード付きの INQUIRE EXITPROGRAM コマンドを使用できます。

INQUIRE EXITPROGRAM コマンドの CONNECTST キーワードにより、以下の値が返されます。

- CONNECTED。CICS Db2 接続機能が SQL 要求を受け入れる準備ができている場合。
- NOTCONNECTED。CICS Db2 接続機能が SQL 要求を受け入れる準備ができていない場合。

コマンドが PGMIDERR で失敗する場合は、NOTCONNECTED と同じ状態です。

図 28 に、INQUIRE EXITPROGRAM コマンドを使用するアセンブラー・コードの例を示します。

```
CSTAT DS F
  ENTNAME DS CL8
  EXITPROG DS CL8
  ...
  MVC ENTNAME,=CL8'DSNCSQL'
  MVC EXITPROG,=CL8'DFHD2EX1'
  EXEC CICS INQUIRE EXITPROGRAM(EXITPROG) X
  ENTRYNAME(ENTNAME) CONNECTST(CSTAT) NOHANDLE
  CLC EIBRESP,DFHRESP(NORMAL)
  BNE NOTREADY
  CLC CSTAT,DFHVALUE(CONNECTED)
  BNE NOTREADY
```

図 28. INQUIRE EXITPROGRAM コマンドの例

プログラム名として DSN2EXT1 を指定すると、CICS はそれを必要な名前である DFHD2EX1 に動的に変更します。

z/OS ワークロード・マネージャー (WLM) を使用する動的ワークロード・バランシングが行われる環境での実行時には、アプリケーションによる EXTRACT EXIT または INQUIRE EXITPROGRAM コマンドの使用についてさらに考慮する必要があります。

Db2 の可用性をテストすると「ストーム・ドレーン作用」になる可能性があります。ワークロード・マネージャーへの接続が使用不能であるときにアプリケーションが正常に戻ると、ワークロード・マネージャーはだまされたような状態になって良好な応答時間が達成されていると見なすので、より多くの作業を CICS 領域にルーティングすることがあります。ストーム・ドレーン作用の説明については、ストーム・ドレーン作用の回避を参照してください。したがって、以下のことを行うようお勧めします。

- DB2CONN に STANDBYMODE=RECONNECT を指定します。これにより、最初の接続試行時に Db2 がダウンしている場合に、CICS Db2 接続機能は (待機モードで)、DB2 が自動的に初期化および接続するまで確実に待機するようになります。また、その後、Db2 で障害が発生した場合、CICS Db2 接続機能が再び待機モードに戻り、Db2 を待機します。その後、Db2 が正常な状態に戻ったときに自動的に接続されます。
- アプリケーションが正しく -923 コードを処理する場合は、CONNECTERROR=SQLCODE を使用します。
- CONNECTERROR=SQLCODE を使用できる場合は、EXTRACT EXIT や INQUIRE EXITPROGRAM コマンドの使用を避けます。
- AEY9 異常終了が必要な場合は、CONNECTERROR=ABEND を使用します。EXTRACT EXIT コマンドの代わりに、INQUIRE EXITPROGRAM コマンドを使用してください。
- 次のような場合は、STANDBYMODE=RECONNECT と CONNECTERROR=SQLCODE が指定されていても、AEY9 異常終了が依然として発生する可能性があることに注意してください。
 - CICS Db2 接続機能が全く開始されない場合。アプリケーションが EXEC SQL コマンドを発行すると AEY9 が発生します。常に、SIT に DB2CONN=YES を指定するか、PLTPI にプログラム DFHD2CM0 を指定する必要があります。したがって、CICS Db2 接続は少なくとも待機モードになります。
 - CICS Db2 接続は、DSNC STOP または CEMT/EXEC CICS SET DB2CONN NOTCONNECTED コマンドを使用してシャットダウンされます。

接続のシャットダウンを回避することをお勧めします。CICS Db2 SPI コマンドを使用すれば、接続をシャットダウンせずに環境の動的変更を行うことができます。

第 6 章 Java プログラムから Db2 データにアクセスするための JDBC および SQLJ の使用

CICS で実行される Java プログラムでは、Db2 データベースに保持されているデータにアクセスするためにいくつかの方法を使用できます。

Java プログラムでは、以下のことが可能です。

- JCICS LINK コマンドを使用することにより、構造化照会言語 (SQL) コマンドを使用してデータにアクセスする CICS プログラムにリンクする。
- Java Data Base Connectivity (JDBC) または Structured Query Language for Java (SQLJ) アプリケーション・プログラミング・インターフェースを使用して、データに直接アクセスする。

IBM Data Server Driver for JDBC and SQLJ

CICS 用の Java アプリケーションで、IBM Db2 および Db2 for z/OS データベース用のその他のプログラムに対してタイプ 2 接続またはタイプ 4 接続のいずれかの JDBC 要求と SQLJ 要求が行われると、要求は IBM Data Server Driver for JDBC and SQLJ によって処理されます。

タイプ 2 接続を使用する CICS 環境では、IBM Data Server Driver for JDBC and SQLJ によって、JDBC 要求または SQLJ 要求がそれらと同等の EXEC SQL に変換されます。変換された要求は、非 Java プログラムからの EXEC SQL 要求と同じ方法で、CICS Db2 接続機能に到着します。CICS Db2 接続機能のカスタマイズ・オプションおよびチューニング・オプションは、Java プログラムと非 Java プログラムに同様に適用されます。

タイプ 2 接続を使用する場合、ClientUser 値や ClientHostName 値などの Db2 クライアント情報はサポートされず、無視されます。

Liberty JVM サーバーでは、Db2 サブシステムへの接続に CICS Db2 接続機能ではなく TCP/IP を使用するタイプ 4 接続もサポートされます。CICS で IBM Data Server Driver for JDBC and SQLJ を使用するには、適切なレベルのドライバー・バージョンをサポートする Db2 サブシステムに接続している必要があります。

ご使用のバージョンの Db2 に適用される JDBC および SQLJ のアプリケーション・プログラミング・インターフェースを使用する Java アプリケーションのコーディングとビルドの方法について詳しくは、IBM データ・サーバー用の Java アプリケーション開発を参照してください。

Db2 をサポートするための JVM サーバーの構成

JVM サーバーは、Java アプリケーション用のランタイム環境です。JDBC および SQLJ ベースのアプリケーションをサポートするように JVM サーバーを構成できます。

始める前に

Db2 で JVM サーバーを使用するには、IBM Data Server Driver for JDBC and SQLJ の最新バージョンをインストールするのがベスト・プラクティスです。Db2 SDSNLOD2 ライブラリーを CICS STEPLIB 連結に追加する必要があります。必要な APAR について詳しくは、詳細なシステム要件を参照してください。

このタスクについて

以下の手順に従って、Db2 に付属の IBM Data Server Driver for JDBC and SQLJ をアプリケーションで使用できるようにします。

手順

1. Liberty JVM サーバー用のドライバーをセットアップするには、Liberty JVM サーバーの構成を参照してください。OSGi JVM サーバーについては、OSGi アプリケーションのための JVM サーバーの構成を参照してください。
2. Db2 環境変数 **DB2SQLJPROPERTIES** は、JVM サーバーではサポートされません。したがって、Db2 ドライバーに関連するプロパティを JVM プロファイルに直接設定する必要があります。使用可能なプロパティについては、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」を参照してください。
3. オプション: JVMSERVER の Db2 トレースを生成する場合は、以下の JVM プロファイル・プロパティが役立ちます。

```
-Ddb2.jcc.traceDirectory=/u/<userID>/db2trace/  
-Ddb2.jcc.traceFile=jccTrace.txt  
-Ddb2.jcc.appendFile=true  
-Ddb2.jcc.traceLevel=-1
```

重要: JVM プロファイルに Db2 プロパティが設定されている場合、このプロパティが設定されていないすべての `dataSource` のデフォルトになります。

次のタスク

Java 2 セキュリティー・ポリシー・メカニズムがアクティブな OSGi JVM サーバーで Java アプリケーションから JDBC または SQLJ を使用する場合は、Java セキュリティー・マネージャーの有効化を参照してください。

JVM サーバーでの Db2 スキーマの設定

さまざまな方法で Db2 にアクセスする JVM サーバー用の現行の Db2 スキーマを設定できます。

デフォルトの Db2 スキーマが必要なスキーマではない場合、以下の 2 つのオプションがあります。

- `db2.jcc.currentSchema` を、JVM プロファイルに必要なスキーマの名前に設定できます。例えば、JVM プロファイルの
`-Ddb2.jcc.currentSchema=CICSDB2Setting` スキーマでは、JCC ドライバーを介したすべての Db2 アクセスで、必ず選択したスキーマが尊重されます。Liberty JVM サーバーの場合、`dataSource` 定義で指定変更されない限り、すべての `dataSource` 定義でこの値が尊重されます。

- Liberty JVM サーバーを使用している場合は、個々の dataSource にスキーマを設定できます。以下に例を示します。

```
<dataSource id="ds1" jndiName="jdbc/ds1">
  <jdbcDriver libraryRef="db2Lib"/>
  <properties.db2.jcc currentSchema="TESTER"/>
</dataSource>
```

ヒント: CICS の (自動構成を使用して作成された) デフォルト dataSource を使用する場合は、サーバーが始動されるたびに server.xml に設定されたすべての値が上書きされます。CICS のデフォルト dataSource に更新されて値が上書きされる可能性がないように、独自の dataSource を作成します。

JDBC および SQLJ API に対するプログラミング

CICS 用の Java プログラムでは、JDBC アプリケーション・プログラミング・インターフェース (API) のプログラミング規則を順守する必要があります。この規則は、一般的な CICS プログラミング・モデルよりも厳格な場合があります。

IBM Data Server Driver for JDBC and SQLJ では、JDBC 4.1 以前のバージョンがサポートされます。

OSGi Java プログラムを DB2 に接続する場合は、バンドル・マニフェスト内の Import-Package に com.ibm.db2.jcc;resolution:=optional を追加します。

注: Liberty JVM サーバーでは、タイプ 2 接続を使用する SQLJ がサポートされます。タイプ 4 接続を使用する SQLJ は、サポートされません。

JDBC API の詳細については、Oracle JDBC Web サイトを参照してください。JDBC および SQLJ の API を使用する Java アプリケーションの開発方法については、を参照してください。

JVM サーバーへのシリアルライズされた SQLJ プロファイルのデプロイ

SQLJ は、Java アプリケーションの組み込み静的 SQL を使用できるようにするものです。JDBC アプリケーション・プログラミング・インターフェースを使用する代替の方法として、シリアルライズされた SQLJ プロファイルを OSGi JVM サーバーまたは Liberty JVM サーバーにデプロイできます。

始める前に

SQLJ プログラム用のプログラム準備の手順に従って、SQLJ プログラムの準備を行う必要があります。

DBRM を作成したら、それらを Db2 プランまたはパッケージと、CICS に定義された Db2 プランを指定した Db2 Entry にバインドする必要があります。これに失敗すると、SQL -805 エラーになるか、JDBC を使用するように SQLJ がデフォルト設定されます。バンドルを作成するワークステーションに、シリアルライズされたプロファイルのファイルをダウンロードします。コード・ページ変換の必要がないように、転送はバイナリー形式で行ってください。これにより、後で問題が発生することを防止できます。

次に、シリアル化されたプロファイルを配置する場所を選択する必要があります。次の 2 つのシナリオがあります。

手順

1. シナリオ 1 - シリアル化されたプロファイルを、デプロイしたバンドル内に保持します。
 - a. OSGI バンドルの場合、シリアル化されたプロファイルをバンドルのルート・ディレクトリーまたはバンドル・クラス・ディレクトリーに保持することができます。Liberty バンドルの場合、他のクラスと共に bin ディレクトリーに配置する必要があります。
2. シナリオ 2 - プロファイルを外部化します。
 - a. シリアル化されたプロファイルを、jar の中から USS ファイル・システム内のディレクトリー (/usr/lpp/cicsts/dev/sqlj.profile.dir など) に移動します。ファイルは、/usr/lpp/cicsts/dev/sqlj.profile.dir/com/ibm/cics/test/sqlj/CurrentTimeStamp_SJProfile0.ser のように、同じパッケージ構造のディレクトリーに配置する必要があります。
 - b. バンドル・マニフェストに項目を追加します。

```
Bundle-ClassPath: .,external:$sqlj.profile.dir$
```

「external」という文字列はバンドルの外部の場所を表し、\$ で囲まれた文字列は、Java システム・プロパティー sqlj.profile.dir の値に置き換えられます。

- c. JVM プロファイルに次の例のような Java システム・プロパティーを追加します。

```
-Dsqlj.profile.dir=/usr/lpp/cicsts/dev/sqlj.profile.dir
```

タスクの結果

これで、シリアル化された SQLJ プロファイルが正常にデプロイされました。

データベースへの接続の取得

SQL ステートメントを実行する前に、JDBC または SQLJ アプリケーションで、データベースへの接続を獲得する必要があります。アプリケーションは、2 つの Java インターフェースのうちのいずれかを使用してターゲット・データ・ソースに接続します。

このタスクについて

2 つの Java インターフェースとは、以下の Java クラスです。

- **DriverManager**: このクラスは、データベース URL で指定されたデータベースにアプリケーションを接続します。
- **DataSource**: このインターフェースは、基礎データベースに関する詳細がアプリケーションから認識されないようにします。これにより、アプリケーションの移植性が高まるため、DriverManager よりも好まれます。DataSource の実装は、Java プログラムの外部で作成され、Java Naming and Directory Interface (JNDI) の DataSource 名ルックアップを使用して取得されます。DataSource インターフェースは Liberty JVM サーバーでのみ使用可能です。

OSGi JVM サーバーでは、Db2 で使用できるのは、DriverManager を使用するタイプ 2 接続のみです。

Liberty JVM サーバーでは、DriverManager インターフェースと DataSource インターフェースのどちらを使用しても、Db2 にアクセスできます。

タイプ 2 接続の JDBC を使用する場合は、代わりに既存の CICS Db2 セキュリティー手順が使用されるため、ユーザー ID とパスワードを指定する必要はありません。また、デフォルト URL を使用することをお勧めします。作業単位のコミットを参照してください。

IBM CICS SDK for Java には、OSGi と Liberty JVM サーバーの両方の JDBC サンプルが用意されています。詳しくは、Liberty JVM サーバーで実行する Java アプリケーションの開発を参照してください。

JDBC の DriverManager および DataSource インターフェースを使用して接続を獲得する方法、およびアプリケーションで利用できるサンプル・コードについて詳しくは、使用する Db2 バージョンに適した「Db2 for z/OS」の『Java 用プログラミング』を参照してください。

データベースへの **DriverManager** 接続の取得

DriverManager クラスを使用する場合、データベースへの接続は、JDBC ドライバーに渡されるデータベース URL (Uniform Resource Locator) によって識別されます。

このタスクについて

OSGi JVM サーバーで DriverManager 接続を構成するには、JVM プロファイルの OSGI_BUNDLES および LIBPATH_SUFFIX に Db2 JDBC jar およびネイティブ DLL を指定します。

Liberty JVM サーバーでは、DriverManager 構成は CICS JDBC Liberty 機能によって提供されます。

JDBC ドライバーは次の 2 つのタイプの URL を認識します。

デフォルト URL

デフォルト URL には、Db2 サブシステムのロケーション名は含まれません。Db2 for z/OS のデフォルト URL は次の 2 つの形式のうちのいずれかで指定できます。

```
jdbc:db2os390sqlj:
```

```
jdbc:default:connection
```

デフォルト URL を指定すると、アプリケーションに、CICS の接続先のローカル Db2 への接続が指定されます。ご使用のシステムで Db2 データ共有を使用している場合は、ローカル Db2 からシスプレックス内のすべてのデータにアクセスできます。

明示的 URL

明示的 URL には Db2 サブシステムのロケーション名が含まれます。
Db2 for z/OS の明示的 URL の基本構造は以下のとおりです。

```
jdbc:db2os390:<location-name>
```

```
jdbc:db2os390sqlj:<location-name>
```

通常、ロケーション名は、CICS の接続先のローカル Db2 の名前です。ただし、アクセスするリモート Db2 の名前を指定することができます。この場合、CICS はローカル Db2 をパススルーとして使用し、Db2 分散データ機能を使用してリモート Db2 にアクセスします。

タイプ 2 接続の Db2 にアクセスする場合は、デフォルト URL を使用することをお勧めします。明示的 URL を使用すると、同じアプリケーション・スイートで複数のプログラムが使用されている場合に不都合となる可能性のある特定のアクションが、接続のクローズ時に実行される場合があります。また、デフォルト URL を使用すれば、接続の動作が JDBC ドライバーのバージョンによって影響を受けることはありません。

接続を獲得するには、Java アプリケーションから、URL を指定して `getConnection()` メソッドを呼び出す必要があります。例:

```
Connection connection =  
DriverManager.getConnection("jdbc:default:connection");
```

OSGi Java プログラムを Db2 に接続する場合は、バンドル・マニフェスト内の `Import-Package` に `com.ibm.db2.jcc;resolution:=optional` を追加します。

データベースへの **DataSource** 接続の取得

`DataSource` インターフェースを使用する場合、データベースへの接続は、Java Naming and Directory Interface (JNDI) の `DataSource` 名ルックアップを使用して取得されます。JDBC `DataSource` 実装は、`<cicsts:jdbc-1.0>` または `Liberty <jdbc-4.1>` のいずれかのフィーチャーによって提供されます。

このタスクについて

データ・ソースという用語は、以下のように 2 つの異なるコンテキストで使用することができますが、大文字化によって区別されるため、注意が必要です。

- データ・ソースは、データベースなどのデータのソースです。
- `DataSource` は、これによって表される実世界のデータ・ソースを識別および記述するプロパティのセットをカプセル化する Java EE の方法です。
`DataSource` オブジェクトは、それが表わす特定のデータ・ソースへの接続のファクトリーと考えることができます。
- `DataSource` オブジェクトが JNDI ネーミング・サービスに登録されると、アプリケーションで JNDI API を使用してその `DataSource` オブジェクトにアクセスできます。その後、このオブジェクトを使用して、これが表わすデータ・ソースに接続できます。

- DataSource オブジェクトでは、接続プール (物理接続の一時論理表現のセット) の実装を選択できます。アプリケーションでそのような接続が閉じられると、一時的な接続 (JDBC 接続とも呼ばれる) は閉じられませんが、再利用のためにプールに返されます。

タイプ 4 接続の JDBC を使用しているときに、CICS 作業単位で行われた更新によるデータベース更新を調整する場合は、CICS JTA 統合サポートを使用して、UserTransaction の有効範囲内で更新を行います。詳しくは、Java Transaction API (JTA) を参照してください。さらに、DataSource が XADataSource (要素 type="javax.sql.XADataSource" を定義に追加することで指定されます) であることを確認する必要があります。Liberty のデフォルト DataSource である <dataSource id="DefaultDataSource"> を使用する場合は、これはデフォルトで XADataSource になります。

DataSource が構成された後、アプリケーションは、Liberty サーバー構成の cicsts_dataSource エレメントまたは dataSource エレメントのいずれかにそれぞれ指定された jndiName を使用して、JNDI ネーミング・サービスから該当する DataSource クラスのインスタンスを取得できます。そして、その DataSource オブジェクトの getConnection() メソッドを呼び出して接続を取得できます。例:

```
Context context = new InitialContext();
DataSource dataSource = (DataSource)
context.lookup("jdbc/defaultCICSDataSource");

Connection connection = dataSource.getConnection();
```

JDBC 接続と SQLJ 接続に関する考慮事項

CICS Java アプリケーションで IBM Data Server Driver for JDBC and SQLJ を使用する場合は、以下の情報を考慮する必要があります。

タイプ 2 接続の IBM Data Server Driver for JDBC and SQLJ の使用

タイプ 2 接続を使用する場合、CICS 用の Java アプリケーションでは、一度に開くことができる JDBC 接続または SQLJ 接続は 1 つのみです。

JDBC では、アプリケーションは同時に複数の接続を行うことができますが、タイプ 2 接続を使用する場合は、これが 1 つに制限されます。ただし、アプリケーションは、必要に応じて既存の接続を閉じ、新規の Db2 ロケーションへの接続を開くことができます。

ガイドラインとして、接続が開いているアプリケーションは、JDBC または SQLJ を使用する可能性がある別のアプリケーションにリンクする前に、その接続を閉じる必要があります。アプリケーション・スイートの一部である Java プログラムでは、接続を閉じた場合の影響を考慮する必要があります。明示的 URL を使用している場合、接続を閉じると同期点が取られる可能性があるからです。デフォルト URL を使用する場合は、接続を閉じるときに同期点を取る必要はありません。

タイプ 2 接続について詳しくは、130 ページの『作業単位のコミット』を参照してください。

タイプ 4 接続の IBM Data Server Driver for JDBC and SQLJ の使用 (Liberty JVM サーバーのみ)

タイプ 4 接続を使用する場合、CICS 用の Java アプリケーションでは、同時に複数の JDBC 接続または複数の SQLJ 接続を行うことができます。データ保全性を維持するためには注意が必要です。Java Transaction API を使用して、トランザクション境界を明確に区切ることをお勧めします。

注:

JDBC 4.0 実装と JDBC 4.1 実装は、同じドライバー内にあります。

jdbc-4.1 機能を使用すると、**RowSetFactory** インターフェースと **RowSetProvider** クラスは新規 JDBC 接続の作成を試みます。

データベースへの接続を閉じる

データベースへの接続を閉じると、JDBC と SQLJ のリソースが自動的に解放されます。通常、データベースへの接続はタスクの終了時に閉じられます。

パフォーマンス上の理由から、アプリケーションは、同じ JVM のユーザーが後でできるように JDBC 接続を開いたままにする場合があります。JDBC 接続が開いたままになっている場合、アプリケーションは、JDBC リソースが時間の経過に伴いリークしないようにする必要があります。

アプリケーションで JDBC または SQLJ 接続を開いたままにする場合は、そのアプリケーションで以下を行う必要があります。

- JDBC と SQLJ のリリースが解放されていることの確認。
- 基礎となる Db2 接続の Db2 SIGNON 後のリカバリー。
- キャッシュされた接続が無効になった場合 (例えば、StaleConnection, SQLCODE=4499) の接続のリサイクル。

接続がキャッシュされている場合は、所定の数 of トランザクション後に接続をリサイクルするロジックをアプリケーションに組み込むことをお勧めします。キャッシュされた接続のリサイクルはリソースの漏えいを防ぎます。

作業単位のコミット

タイプ 2 接続を使用する場合は、JDBC アプリケーションと SQLJ アプリケーションで、JDBC と SQLJ のコミットおよびロールバックのメソッド呼び出しを発行できます。IBM Data Server Driver for JDBC and SQLJ は、これらの呼び出しを JCICS コミットまたは JCICS ロールバック呼び出しに変換し、その結果、CICS 同期点が取られます。

このタスクについて

JDBC または SQLJ コミットでは、Db2 に対して行われた更新だけでなく、CICS 作業単位全体がコミットされます。CICS では、残りの CICS 作業単位とは別個に JDBC 接続を使用して行われた作業のコミットはサポートされません。

JDBC または SQLJ アプリケーションでは JCICS コミットまたはロールバックを直接実行することもでき、その結果は JDBC または SQLJ のコミットまたはロールバック・メソッド呼び出しを実行した場合と同じになります。作業単位全体は、Db2 更新と CICS 制御リソースに対する更新の両方について、まとめてコミットまたはロールバックされます。

JDBC 接続で作業する場合、Db2 データベースへの接続のクローズ時に、同期点の取得と作業単位のコミットを避けられないことがあります。これは、以下のいずれかの場合に当てはまります。

- JDBC 接続の自動コミット・プロパティを使用した。
- 明示的な URL を使用して DriverManager 接続を取得した。

スタンドアロン・アプリケーションの場合、これらの規則が原因で問題が発生することはありません。CICS では、接続のクローズ時だけでなく、タスクの終了時にも同期点が確実に取得されるためです。ただし、JDBC および SQLJ アプリケーション・プログラミング・インターフェースでは、作業単位ごとの複数のアプリケーション・プログラムの概念はサポートされません。アプリケーションを構成する複数のプログラムがある場合、1 つのプログラムが Db2 にアクセスしてから、単一の作業単位の過程で、同様に Db2 へのアクセスを行う別のプログラムを呼び出す可能性があります。これらのプログラムとして、JDBC または SQLJ を使用する Java プログラムを指定する場合は、Db2 への接続のクローズ時に作業単位がコミットされないようにする必要があります。そうしないと、アプリケーションは計画通りに作動しません。既存のアプリケーションのプログラムを、JDBC または SQLJ を使用する Java プログラムに置き換えて、プログラム間で同じ CICS-Db2 スレッドを共用する場合は、特にこの要件に注意する必要があります。この問題に対処するには、デフォルトの URL で DriverManager を使用するか、DataSource 接続を使用してください。

自動コミット

JDBC アプリケーションでは、JDBC 接続の自動コミット・プロパティを使用できます。自動コミット・プロパティによって、Db2 がそれぞれ更新された後に、コミットが行われます。タイプ 2 接続を使用する場合、このコミットは CICS[®] コミットであり、結果として作業単位全体がコミットされます。

自動コミット・プロパティを使用すると、接続が閉じられたときにもコミットされます。タイプ 2 接続に自動コミットを使用することはお勧めしません。このため、IBM Data Server Driver for JDBC and SQLJ では、タイプ 2 接続の CICS 環境で実行される場合に、autocommit(false) のデフォルトが設定されます。タイプ 4 接続の場合、デフォルトは autocommit(true) です。

明示的 URL およびデフォルト URL での DriverManager の同期点の問題

JDBC または SQLJ を使用する CICS 用の Java アプリケーションで、明示的 URL を使用して接続が獲得される場合、アプリケーションは、SYNCONRETURN 属性でリンクされている DPL サーバー・プログラムの環境に類似した環境で動作します。

JDBC または SQLJ を使用するアプリケーション・プログラムが明示的 URL 接続を閉じるときに、CICS が IBM Data Server Driver for JDBC and SQLJ を使用している場合、暗黙的な同期点は取られません。

ただし、明示的 URL 接続のクローズは、作業単位の境界にあるときにのみ成功します。したがって、アプリケーションでは、接続を閉じる前に、JDBC または SQLJ のコミット・メソッド呼び出しで JCICS コミットを発行して、同期点を取る必要があります。（アプリケーションでは、確実に同期点を取るために `autocommit(true)` を使用できますが、このプロパティの使用は CICS 環境では推奨されません。）アプリケーション・プログラムが明示的 URL 接続を閉じたときに、作業単位の終了となります。

明示的 URL の代わりにデフォルト URL を使用して接続を獲得するか、デフォルト URL 接続を提供するデータ・ソースを使用することによって、この制限に対処できます（126 ページの『データベースへの接続の取得』を参照）。デフォルト URL が使用される場合、Java アプリケーションでは、作業単位の境界で接続を閉じる必要はなく、接続が閉じられるときに同期点は取られません（`autocommit(true)` が指定されていないことを前提とします）。

DriverManager タイプ 2 接続を使用する場合は、常にデフォルト URL 接続を使用することをお勧めします。

JDBC 要求または SQLJ 要求の間に CICS が異常終了する

IBM Data Server Driver for JDBC and SQLJ によって作成され、EXEC SQL 要求の処理中に発行される CICS の異常終了は、Java 例外に変換されないため、CICS 用の Java アプリケーションではキャッチされません。CICS トランザクションは異常終了して、最終同期点までロールバックされます。

第 7 章 CICS Db2 プログラムの実行および実動の準備

このセクションでは、CICS Db2 環境でのプログラムの準備について説明します。

CICS Db2 環境での Java プログラムのサポートについては、Java プログラムから Db2 データにアクセスするための JDBC および SQLJ の使用を参照してください。

以下のトピックには、診断、変更または調整に関する情報が含まれています。

CICS Db2 テスト環境

CICS Db2 テスト環境をセットアップするときには、いくつかの CICS システムを Db2 システムに接続させる必要があるかを考慮してください。

複数の CICS システムを同じ Db2 システムに接続することができます。ただし CICS Db2 接続機能では、1 つの CICS システムを複数の Db2 システムに同時に接続することはできません。

次のように、実動環境およびテスト環境をセットアップすることが可能です。

- 1 つの CICS システムを 1 つの Db2 システムに接続する
- 実動用とテスト用に複数の CICS システムを同じ Db2 システムに接続する
- 複数の CICS システムを、複数の異なる Db2 システムに接続する

最初の選択肢 (実動とテストに 1 つの CICS システムを使用すること) は推奨されません。テストのアプリケーションが実動システムのパフォーマンスに影響を与える可能性があるためです。

2 番目の選択肢 (ただ 1 つの Db2 システム) をテスト用および実動用に使用することも可能です。これが適しているかどうかは、実際の開発環境と実動環境に応じて異なります。テスト CICS システムと実動 CICS システムを別個に実行することで、テストにおける障害が実動に影響するのを防ぐことができます。

3 番目の選択肢 (例えばテスト用と実動用にそれぞれ 1 つの Db2 システムを使用すること) は最も柔軟です。2 つの CICS サブシステムを 1 つ以上の Db2 システムと共に実行することができます。複数の CICS システムが異なる Db2 システムに接続される場合、

- ユーザー・データと Db2 カタログは共用されません。テスト・データを実動データから分離する必要がある場合には、これが利点となります。
- テスト対象アプリケーションにおける間違った設計やプログラム・エラーは、実動システムのパフォーマンスに影響を与えません。
- 実動データが使用不可であるため、テスト・システムでの権限付与をそれほど厳しくする必要がないかもしれません。2 つの CICS システムが同じ Db2 システムに接続される場合、プログラマーにとって使用可能な機能とデータに関連する権限を厳しく制御する必要があります。

CICS Db2 プログラムの準備

Db2 Interactive (DB2I) インターフェースを使用して CICS Db2 プログラムを準備することも、独自の JCL をバッチ処理のために実行依頼することもできます。

このタスクについて

図 29に示すステップは、アプリケーション・プログラム設計およびコーディングの完了後に、プログラムを実行するための準備方法を要約しています。

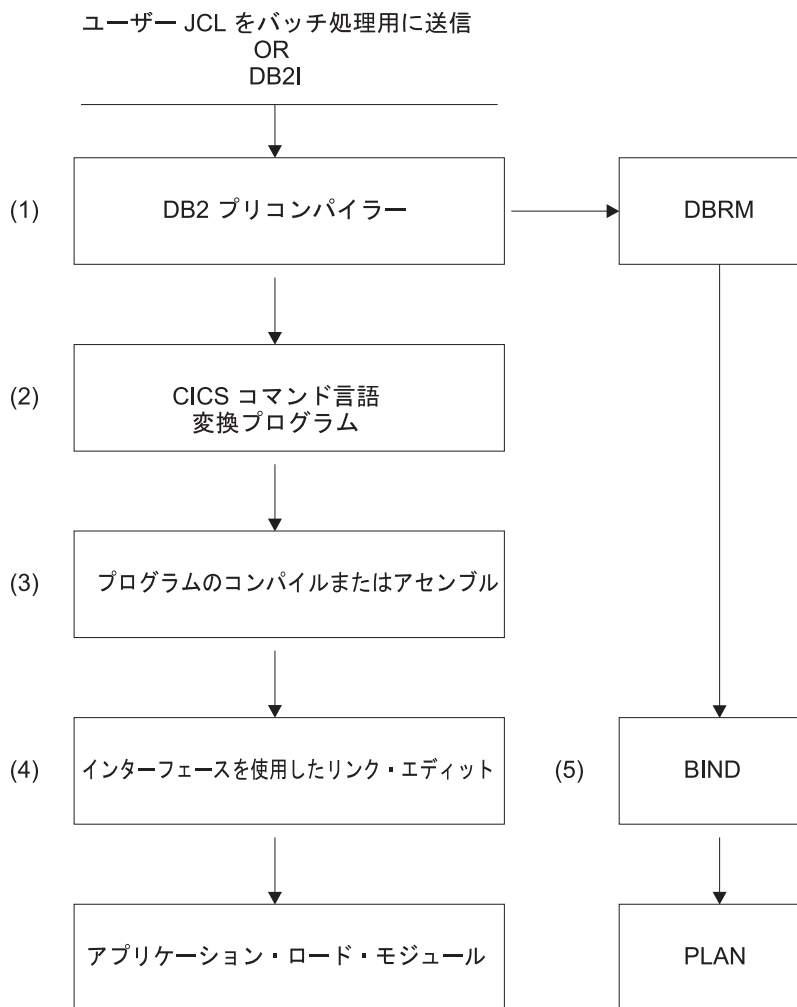


図 29. Db2 にアクセスする CICS アプリケーション・プログラムを準備するステップ

このプロセスの各段階の概要については、Db2 にアクセスする CICS アプリケーション・プログラムの準備を参照してください。

Db2 にアクセスする CICS アプリケーション・プログラムを準備するには、次のようにします。

- Db2 プリコンパイラー (ステップ 1) では、プログラムの各 SQL ステートメントに関する情報を含む DBRM を作成します。また、プログラム内の SQL ステ

ートメントを妥当性検査します。Db2 プリコンパイラーの使用について詳しくは、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」を参照してください。

- ソース・プログラムが PL/I で作成される場合、ステップ 1 の Db2 プリコンパイラーへの入力、PL/I マクロ局面 (使用された場合) からの出力となります。
- ステップ 1 の Db2 プリコンパイラー、およびステップ 2 の CICS コマンド言語変換プログラムを、どちらの順序でも実行できます。示した順序は、推奨される方式であり、これは、DB2I プログラム準備パネルでサポートされている方式です。CICS コマンド言語変換プログラムを最初に実行した場合、EXEC SQL ステートメントが検出されるたびに、警告メッセージが生成されますが、これらのメッセージは結果に影響しません。
- CICS 変換プログラムを統合した、言語環境プログラム (Language Environment) に準拠するコンパイラー (COBOL および PL/I) の 1 つを使用する場合、EXEC CICS コマンド (ステップ 2) の変換は、プログラムのコンパイル (ステップ 3) 中に行われます。統合 CICS 変換プログラム、およびそれをサポートするコンパイラーの詳細については、変換およびコンパイルを参照してください。
- DB2 バージョン 7 以上を実行しており、言語環境プログラム準拠の COBOL コンパイラーまたは PL/I コンパイラーのいずれかを使用して、COBOL プログラムまたは PL/I プログラムを準備する場合、コンパイラーが SQL ステートメント・コプロセッサ (DBRM を生成する) も提供するので、独立した Db2 プリコンパイラー (ステップ 1) を使用する必要はありません。SQL ステートメント・コプロセッサの使用方法について詳しくは、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」を参照してください。
- DB2 バージョン 6 以前を実行しており、COBOL プログラムまたは PL/I プログラムを準備している場合は、独立した Db2 プリコンパイラーを使用します。COBOL プログラムの場合、Db2 プリコンパイラーおよび統合 CICS 変換プログラムの場合と同様に、ストリング区切り文字を指定するようにしてください。デフォルトの区切り文字は、互換性がありません。
- プログラムのリンク・エディット (ステップ 4) では、コーディングしている言語に適した CICS EXEC インターフェース・モジュールまたはスタブと、CICS Db2 言語インターフェース・モジュール DSNCLI の両方を組み込みます。CICS EXEC インターフェース・モジュールを最初にロード・モジュールに組み込む必要があります。24 ビット・アドレッシング・モードまたは 31 ビット・アドレッシング・モード (AMODE=31) のいずれかで、DSNCLI をプログラムにリンクできます。アプリケーション・プログラムが 31 ビット・アドレッシング・モードで稼働する場合は、アプリケーションが 16MB より上で稼働できるように、属性 AMODE=31 および RMODE=ANY を指定して DSNCLI スタブをアプリケーションにリンク・エディットする必要があります。
- バインド・プロセス (ステップ 5) では、Db2 が必要です。バインド・プロセスでは、DBRM を使用して、プログラムによる Db2 データへのアクセスを可能にするアプリケーション・プラン (多くの場合、単にプランと呼ばれる) を作成します。バインド・プロセスの詳細については、バインド・プロセスを参照してください。同一のエントリー・スレッド (つまり、同一の DB2ENTRY で指定されている) を使用するトランザクションのグループは、同一のアプリケーション・プランを使用しなければなりません。それらの DBRM は、同一のアプリケ

ーション・プランにバインドするか、または後で同一のアプリケーション・プラン内にリストされるパッケージにバインドする必要があります。

表 8 では、プログラムの言語および Db2 のバージョンに応じて、CICS Db2 プログラムを準備するために必要となるタスクを示しています。

表 8. Db2 にアクセスする CICS プログラムを準備するタスク

Db2 バージョン およびプログラ ム言語	ステップ 1 (SQL ステ ートメント 処理)	ステップ 2 (CICS コマン ド変換)	ステップ 3 (プログラム・ コンパイル)	ステップ 4 (リ ンク・エディッ ト)	ステッ プ 5 (バ インド)
DB2 バージョン 6 およびアセン ブラー	Db2 プリコ ンパイラー	CICS 提供の 独立変換プロ グラム	言語コンパイ ラー	EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 6 および PL/I	Db2 プリコ ンパイラー	統合 CICS 変換プログラムを サポートする言語コンパイラ ー		EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 6 および COBOL	Db2 プリコ ンパイラー	統合 CICS 変換プログラムを サポートする言語コンパイラ ー		EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 6 およびその他 の言語	Db2 プリコ ンパイラー	CICS 提供の 独立変換プロ グラム	言語コンパイ ラー	EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 7 (またはそれ以 降) およびアセン ブラー	Db2 プリコ ンパイラー	CICS 提供の 独立変換プロ グラム	言語コンパイ ラー	EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 7 (またはそれ以 降) および PL/I	統合 CICS 変換プログラムおよび SQL ス テートメント・コプロセッサをサポート する言語コンパイラー			EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス
DB2 バージョン 7 (またはそれ以 降) および COBOL	統合 CICS 変換プログラムおよび SQL ス テートメント・コプロセッサをサポート する言語コンパイラー			EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス

表 8. Db2 にアクセスする CICS プログラムを準備するタスク (続き)

Db2 バージョン およびプログラ ム言語	ステップ 1 (SQL ステ ートメント 処理)	ステップ 2 (CICS コマン ド変換)	ステップ 3 (プログラム・ コンパイル)	ステップ 4 (リ ンク・エディッ ト)	ステッ プ 5 (バ インド)
DB2 バージョン 7 (またはそれ以 降) および他の言 語	Db2 プリコ ンパイラー	CICS 提供の 独立変換プロ グラム	言語コンパイ ラー	EXEC インター フェースおよび DSNCLI を使用 したリンク・エ ディット	バイン ド・プ ロセス

CICS Db2 プログラムの準備は、Db2 Interactive (DB2I) インターフェースを使っ
て行うか、または独自の JCL をバッチ実行のために処理依頼して行います。

- Db2 Interactive (DB2I) インターフェース: DB2I は、プリコンパイル、コンパイル、またはアセンブルするためのパネル、アプリケーション・プログラムをリンク・エディットするためのパネル、およびプランをバインドするためのパネルを提供します。アプリケーション・プログラムの準備の詳細については、「Db2 for z/OS 製品資料内の『Db2 for z/OS のプログラミング』」を参照してください。
- バッチ実行のために処理依頼されるユーザー JCL: Db2 ライブラリー SDSNSAMP 内のメンバー DSNTEJ5C および DSNTEJ5P には、CICS 用の COBOL プログラムおよび PL/I プログラムの準備に必要な JCL のサンプルが含まれています。

CICS の稼働中にプログラムを実行のために準備する場合、プログラムの新しいバージョンを CICS に認識させるために、CEMT NEWCOPY コマンドの実行が必要になることがあります。

CICS SQLCA フォーマット設定ルーチン

IBM 提供の SQLCODE メッセージ・フォーマット設定プロシージャである DSNTIAR を使用すると、アプリケーションに「SQL メッセージをオンラインで」送信できます。

DB2 バージョン 3.1 では、DSNTIAR は 2 つのフロントエンド・モジュール (DSNTIAC と DSNTIAR) および 1 つのランタイム・モジュール (DSNTIA1) に分割されました。DSNTIAC は CICS アプリケーションに使用され、DSNTIAR はその他の Db2 インターフェースに使用されます。DB2 バージョン 3.1 より前では、(リリース変更またはメンテナンス適用によって) DSNTIAR が変更されるたびにアプリケーション・モジュールを再リンク・エディットする必要がありましたが、上記のような変更点により、その必要がなくなりました。以前に DSNTIAR でリンク・エディットされたアプリケーションが存在する場合、代わりに DSNTIAC を使ってそれらを再びリンク・エディットすることを考慮してください。こうするとパフォーマンスが改善され、DSNTIAR の変更からそれらを分離することができます。

CICS フロントエンド部分である DSNTIAC は、Db2 ライブラリー SDSNSAMP でソース・メンバーとして提供されます。

DSNTIAC および DSNTIA1 に必要なプログラム定義は、CSD における IBM 提供グループ DFHDB2 で提供されます。DSNTIA1 をロード可能にするには、SDSNLOAD ライブラリーを CICS DFHRPL 連結に追加する必要があります (CICS ライブラリーの後)。

プログラム変更後に何をバインドするか

プログラムを変更した場合、そのプログラムを使用する前に、それを準備して再バインドする必要があります。

このタスクについて

バインド・プロセスの概要については、バインド・プロセスを参照してください。プランとパッケージの概要については、計画、パッケージ、および動的計画出口を参照してください。

4 つのプログラム・モジュールから成る CICS トランザクションがあるとして、モジュール 1 がメイン・モジュールです。モジュール 1 がモジュール 2 を呼び出します。モジュール 1 はモジュール 3 も呼び出し、モジュール 3 はモジュール 4 を呼び出します。実際のトランザクションでモジュールの数が多いのは珍しくありません。モジュールの 1 つで少なくとも 1 つの SQL ステートメントが変更されたとすると、次の手順を実行してプログラムを準備し、トランザクションを再度実行可能にする必要があります。

手順

1. Db2 でプログラムをプリコンパイルします。
2. CICS 変換プログラムを使ってプログラムを変換します。
3. ホスト言語ソース・ステートメントをコンパイルします。
4. リンク・エディットします。
5. プログラム C 用の **DBRM** がパッケージの中にバインドされていた場合、新しい **DBRM** を使ってそのパッケージをバインドすると、プログラム C を使用するすべてのアプリケーション・プランは新しいパッケージを自動的に見つけます。
6. プログラム C 用の **DBRM** がいずれかのアプリケーション・プランの中に直接バインドされていた場合、プログラム C 用 **DBRM** を含むすべてのアプリケーション・プランを見つけて、直接バインドされた全プログラム用の **DBRM** を使ってすべてのアプリケーション・プランを再びバインドして、新しいアプリケーション・プランを獲得します。変更されなかったプログラムに関しては、古い **DBRM** を使用します。REBIND サブコマンドを使用できないことに注意してください。REBIND への入力 **DBRM** ではなく、プランであるためです。

注: 以前にパッケージを使用しなかった場合は、パッケージを使用すると再バインド・プロセスが簡単になります。それぞれ別個の **DBRM** を 1 つのパッケージとしてバインドし、パッケージ・リストにそれらを含めることができます。パッケージ・リストを PLAN の中に含めることができます。その後、(BIND PLAN コマンドを使ってアプリケーション・プラン全体をバインドする代わりに) BIND PACKAGE コマンドを使用して、変更済みプログラム用の **DBRM**

をバインドできます。これによりトランザクションの可用性が高まり、パフォーマンスが改善されます。パッケージの使用方法について詳しくは、Db2 パッケージの使用を参照してください。

プログラムのバインド・オプションと考慮事項

複数のプログラムを 1 つのアプリケーション・プランにバインドするとき、Db2 でタイム・スタンプが使用される方法に注意してください。

各プログラムに関して、Db2 プリコンパイラーは以下のものを作成します。

- Db2 プリコンパイラーは、Tdx タイム・スタンプを持つ DBRM を作成します。例えば、最初のプログラムには Td1、2 番目のプログラムには Td2 となります。
- Db2 プリコンパイラーは、SQL パラメーター・リスト内で Tsx タイム・スタンプを持つ変更されたソース・プログラムを作成します。例えば 2 つのプログラムが処理に含まれる場合は Ts1 および Ts2 です。

バインド時に、各プログラム用の DBRM は、指定したパッケージまたはプランの中にバインドされます。さらに、Db2 のカタログ・テーブル SYSIBM.SYSDBRM が更新され、DBRM ごとに 1 行およびそれぞれのタイム・スタンプが含まれるようになります。実行時に Db2 は各 SQL ステートメントのタイム・スタンプを検査して、DBRM のタイム・スタンプとソース・プログラムで設定されたタイム・スタンプが異なる場合には -818 SQL コードを返します (この例では Td1 と Ts1 が異なるか、Td2 と Ts2 が異なる場合)。-818 SQL コードを回避するには、以下のいずれかの方法を使用してください。

- すべてのプログラムをパッケージにバインドして、アプリケーション・プランでこれらのパッケージをリストします。1 つのプログラムが変更された場合、そのプログラムをプリコンパイル、コンパイル、およびリンク・エディットして、それを 1 つのパッケージに再びバインドします。
- いずれかのプログラムをアプリケーション・プランに直接バインドする場合は、それぞれの新しいプログラムまたは変更済みプログラムを必ずプリコンパイル、コンパイル、およびリンク・エディットした後、そのプログラムを含んでいるすべてのアプリケーション・プランをバインドします。その際、これらのプランに直接バインドされるすべてのプログラムからの DBRM を使用します。これを行うには REBIND コマンドではなく、BIND コマンドを使用してください。

プランをバインドするときには多数のオプションを使用できます。ほとんどすべてのバインド・オプションはアプリケーションに依存しているため、アプリケーション設計時にそれを考慮してください。異なるプランに対して異なる BIND オプションを扱うようプロシーチャーを開発する必要があります。さらに、プロシーチャーは、時間の経過に伴う同じプランの BIND オプションの変化に対処できる必要があります。

以下のセクションは、CICS での BIND オプションに関する具体的な推奨事項をいくつか示しています。

RETAIN

RETAIN は、古いプランからの BIND および EXECUTE 権限が変更されないことを意味します。

RETAIN オプションが使用されない場合、それより前の GRANT による権限はすべて REVOKED (取り消し) になります。BIND コマンドを実行しているユーザーがプランの作成者になります。新しい GRANT コマンドによってすべての権限を再び確立する必要があります。

このような理由で、CICS 環境でプランをバインドするときには RETAIN オプションを使用することをお勧めします。

分離レベル

完成したプランに関して、分離レベルが指定されます。反復可能読み取り (RR) を使用する具体的な必要が生じない限り、カーソル固定 (CS) を使用することをお勧めします。CS を使用することで、ハイレベルな並行性が可能になり、デッドロックのリスクが軽減されます。

分離レベルは、完成したプランに関して指定されることに注意してください。つまり、CICS での特定のモジュール用に RR が必要な場合、プランに含まれるすべての DBRM も RR を使用する必要があります。

さらに、パフォーマンス上の理由で、同じ DB2ENTRY を使う目的で使用頻度の低い多数のトランザクションを一緒にグループ化し、共通のプランを使用させるようにした場合、ただ 1 つのトランザクションだけが RR を必要とする場合でも、この新しいプランもまた RR を使用する必要があります。

プラン検証時

プランは VALIDATE(RUN) または VALIDATE(BIND) でバインドされます。バインドできない SQL ステートメントの処理方法を決定するには VALIDATE(RUN) を使用します。

あるステートメントを実行時にバインドする必要がある場合、それぞれの実行ごとにそれが再バインドされます。つまり、新しい作業単位 (UOW) ごとにステートメントが再バインドされます。

実行時にステートメントをバインドすると、パフォーマンスが影響を受ける可能性があります。実行時にバインドされるステートメントは、実行ごとに再バインドされます。つまり、それぞれの同期点の後でステートメントを再バインドする必要が生じます。CICS でこのオプションを使用することは推奨されません。

動的 SQL を使用するときには VALIDATE(RUN) が必要でないことに注意してください。それでも動的 SQL は、ステートメントが実行時にバインドされることを暗黙に意味します。

CICS Db2 環境では VALIDATE(BIND) を使用すべきです。

ACQUIRE および RELEASE

ACQUIRE および RELEASE パラメーターはトランザクション比率に関連しているため、時間の経過と共にプランによって変化します。

これらのパラメーターに関する一般的な推奨事項については、最適なパフォーマンスのための BIND オプションの選択で説明されています。

CICS Db2 プログラムのテストおよびデバッグ

Db2 にアクセスする CICS アプリケーション・プログラムをテストおよびデバッグするには、CICS 環境で通常使われるツールを使用することができます。これには実行診断機能 (EDF)、CICS 補助トレース、トランザクション・ダンプが含まれます。

これらの問題判別プロセス、および他の問題判別プロセスについては、Db2 のトラブルシューティングを参照してください。

実動への移行: CICS Db2 アプリケーションのチェックリスト

このチェックリストは、設計、開発、テストが終わった後のアプリケーションを実動に移すために実行する必要があるタスクを示しています。

このタスクについて

これらのタスクは、テスト・システムで使われた標準にかなり依存します。例えば以下のような場合には、実行すべきタスクが異なります。

- テストと実動で別個の Db2 システムが存在する
- テストと実動で 1 つの Db2 だけが使用される。

以下の説明では、テストと実動で別個の Db2 および CICS サブシステムを使用することを想定します。

実動への移行は、以下のアクティビティーを実行することを意味します。

DDL を使って実動データベースを準備する

テスト・システムからの DDL ステートメントを基礎として使用し、実動 Db2 システムに対してすべての DDL 操作が実行される必要があります。いくつかの変更が必要になる可能性があります。例えば、1 次および 2 次割り振りを増やすこと、他のボリューム通し番号の定義、CREATE STOGROUP ステートメントでの新しい VCAT の定義などです。

DCLGEN の準備

COBOL および PL/I プログラムの場合、テスト Db2 システムからの DCLGEN 入力を使用して、実動 Db2 システムに対する DCLGEN 操作を実行する必要が生じることがあります。

コンパイルに関するオプションによっては (実動システムでコンパイルが実行されない場合)、代わりの方法として、テスト・ライブラリーから実動ライブラリーに DCLGEN 出力構造をコピーすることができます。これにより、テスト・システムと実動システムの間ですべての情報が分離されたままになります。

実動システムのためのプリコンパイル

テスト・システムで既にプログラムをパッケージにバインドした場合は、このステップを実行する必要はありません。パッケージを実動システムに直接移すことができます。その方法について、詳しくは『実動システムのためのアプリケーション・プランの生成』を参照してください。ただし、プログラムをアプリケーション・プランに直接バインドするか、実動システムでプログラムをパッケージにバインドするためには、実動システムにプログラム用の DBRM を配置する必要があります。以下のいずれかを行うことができます。

- 実動システムで、EXEC SQL ステートメントを含む CICS モジュールをプリコンパイルする。または
- DBRM をテスト・システムから実動システム・ライブラリーにコピーする。

実動システムのためのコンパイルおよびリンク・エディット

ロード・モジュールを生成するには、次のようにします。

- 実動システムでのプリコンパイルによって DBRM が生成された場合は、実動システムで CICS モジュールをコンパイルしてリンク・エディットします。または、
- DBRM がコピーされた場合、またはテスト・システムから実動システムにパッケージを移動しようとしている場合には、ロード・モジュールをテスト・システムから実動システム・ライブラリーにコピーします。

表 9 に示されている手順に従って、変更後のロード・モジュールをテスト・システムから実動システム・ライブラリーにコピーし、古いバージョンのロード・モジュールを置換することができます。

表 9. 変更されたプログラムをテスト環境から実稼働環境に移動する

テスト・システム	実動システム	注
	USER.PROD.LOADLIB(PGM3)	元のロード・モジュール
USER.TEST.LOADLIB(PGM3)		テスト・ロード・モジュール
	USER.OLD.PROD.LOADLIB(PGM3)	古いバージョンのプログラムは別の 実動ライブラリーに配置されます
	USER.PROD.LOADLIB(PGM3)	新しいバージョンのプログラムが 実動ライブラリーに配置されます

適切な JCL を使って実稼働ライブラリーを選択することにより、古いバージョンまたは新しいバージョンのプログラムを実行できます。これにより、プログラム・ロード・モジュールに組み込まれた整合性トークンによって判別される正しいバージョンのパッケージが実行されます。

実動システムのためのアプリケーション・プランの生成

テスト・システムで既にプログラムをパッケージにバインドした場合は、パッケージを実動システムにコピーして、アプリケーション・プランにリストされる集合にそれらを含めることができます。パッケージを実動システムにコピーするとき、アプリケーション・プランに既にリストされている集合にそのパッケージが含まれている限り、実動システムでアプリケーション・プランを再びバインドする必要はありません。

表 10 に示されている手順に従って、変更後のパッケージをテスト・システムから実動システム・ライブラリーにコピーし、古いバージョンのパッケージを置換することができます。この例では、異なるバージョンのパッケージを識別するためにプリコンパイル時に **VERSION** キーワードを使用しています。**VERSION** キーワードに関する説明と使用法について詳しくは、**Db2 for z/OS** 製品資料内の『**Db2 for z/OS** のプログラミング』を参照してください。

表 10. 変更されたパッケージをテスト環境から実稼働環境に移動する

テスト・システム	実動システム	注
	location_name. PROD_COLL.PR3.VER1	古いバージョンのパッケージ
location_name. TEST_COLL.PR3.VER2		新しいバージョンのパッケージがテスト・システムでバインドされた後、実動システムにコピーされます
	location_name. PROD_COLL.PR3.VER1	古いバージョンが引き続き実動コレクションの中に残ります
	location_name. PROD_COLL.PR3.VER2	新しいバージョンが実動コレクションに配置されます

プログラムをアプリケーション・プランに直接バインドする場合、または実動システムでプログラムをパッケージにバインドする場合には、実動システムに配置された DBRM に対してバインド・プロセスを実行する必要があります。プログラムをアプリケーション・プランに直接バインドする場合には、それらのプログラムを含んでいる (実動システム上の) すべてのアプリケーション・プランをバインドする必要があります。バインド・プロセスについて詳しくは、バインド・プロセスを参照してください。なお、テーブルや索引のサイズなどさまざまな要因のために、テスト・システムと実動システムの間で **EXPLAIN** 出力を比較しても無意味である可能性があります。それでも、**Db2** 最適化プログラムの決定を検査するために、実動システムで最初にプランをバインドする時点で **EXPLAIN** を実行することをお勧めします。

GRANT EXECUTE

実動システムで **Db2** アプリケーション・プランに対する **EXECUTE** 権限をユーザーに付与する必要があります。

テスト

この時点で、これ以上のテストは必要ありませんが、リソース競合、デッドロック、およびタイムアウトの発生を最小限に抑え、期待されるトランザクション応答時間が得られることを確認するうえで、ストレス・テストが役立つので、それを行うことが推奨されます。

CICS 定義

新しいアプリケーション・プログラムの実行準備ができた状態にするには、**CICS** 実動システムで以下の **RDO** 定義を更新します。

- 新しいトランザクション・コード用の **RDO** トランザクション定義
- 新しいアプリケーション・プログラムおよびマップに関する **RDO** プログラム定義

- 特定の Db2 要件に関する SIT (それが実動に移される最初の Db2 向けアプリケーションである場合)
- アプリケーションに関する RDO DB2ENTRY および DB2TRAN 定義。RDO DB2CONN 定義 (それが実動に移される最初の Db2 向けアプリケーションである場合)。DB2ENTRY で新しいトランザクションおよびアプリケーション・プランを定義するとき、無保護スレッドを使用して、最初に詳細なアカウンティング情報とパフォーマンス情報を得ることができます。その後、必要に応じて、保護されたスレッドを使用できます。

加えて、RACF がインストールされている場合、新しいユーザーと Db2 オブジェクトを定義する必要があります。

Db2 にアクセスする CICS アプリケーションのチューニング

Db2 にアクセスする CICS アプリケーションを実動に移す前に調整する必要があります、実動に移した後も定期的に調整する必要があります。

このタスクについて

Db2 にアクセスする CICS アプリケーションを実動に移すときには、CICS に対して既に行われている検査のほかに、以下の検査を追加してください。

- Db2 要求を出すすべてのアプリケーション・プログラムがスレッド・セーフであることを確認します。そうである場合、オープン・トランザクション環境 (OTE) を活用して、アプリケーションのパフォーマンスが改善されます。オープン・トランザクション環境でアプリケーション・プログラムが作動する方法については、スレッド・セーフ・プログラミングにより CICS Db2 アプリケーションが OTE を使用できるようにするの説明を参照してください。
- 使用する SQL ステートメントの数とタイプが、プログラム仕様に合致していることを確認します (Db2 アカウンティング機能を使用)。
- バッファ・プール内で取得および更新されたページの数、予期した数よりも多いかどうかを確認します (Db2 アカウンティング機能を使用)。
- 計画された索引が使用されていることを確認し (EXPLAIN を使用)、非効率的な SQL ステートメントが使われていないことを確認します。
- DDL が使用されているかどうかを確認し、使用されている場合はその理由を確認します (Db2 アカウンティング機能を使用)。
- 会話型トランザクションが使用されているかどうかを確認します。

疑似会話型トランザクションを代わりに使用できるかどうかを判別します。会話型の設計が必要な場合は、複数の会話にわたってロックされる Db2 オブジェクトを調べます。また、この会話型設計のために必要とされる新しいスレッドの数が許容範囲内であることも確認します。

- 使用されるロックとそれらの継続期間を確認します。

例えば以下の項目の指定が間違っている (または最適なものではない) ことが原因で、表スペース・ロックが使用されていないかどうか確認します。

- LOCK TABLE ステートメント
- LOCKSIZE=TS の指定

- ISOLATION LEVEL(RR) の指定
- ロック・エスカレーション

この情報はカタログ表で入手可能です。ただしロック・エスカレーションは例外で、これはインストール・パラメーター (DSNZPARM) です。

- 使用されるプランとそれらのサイズを確認します。アプリケーション・プランがセグメント化されるとしても、プラン内でより多くの DBRM が使用されるほど、プランの BIND および REBIND (変更の場合) に要する時間が長くなります。可能なときには常に、パッケージの使用を試みてください。パッケージは以下のような問題を解決するために設計されました。
 - SQL アプリケーションを変更した後に、プラン全体を再びバインドする。
(これは動的プラン選択によって対処可能でしたが、パフォーマンスへの影響がありました。)
 - 変更された SQL アプリケーションが多数のアプリケーションによって使用される場合、各アプリケーション・プランをバインドする。

この調整が完了したら、想定されるトランザクション負荷を使用して、必要な DB2ENTRY 定義と、必要なスレッド数を決めてください。また、これらのトランザクションが Db2 および CICS サブシステムに与える影響も確認してください。

実動時に Db2 にアクセスする CICS アプリケーションを調整する際には、

- バッファ・プールでの GET PAGES の数をモニターすることにより、計画された索引が CICS アプリケーションで使われることを確認します (Db2 アカウンティング機能を使用)。使用されていない索引がある場合、その理由はおそらく、索引が既にドロップされたか、プランのバインド後に索引が作成されたためです。
- アカウンティング機能からのロック・マネージャー・データを使用して、サスペンション、デッドロック、およびタイムアウトを検査します。

第 8 章 Db2 のモニター

CICS Db2 環境のアカウントिंगとモニター。

CICS 提供のアカウントिंगおよびモニター情報

CICS には、アカウントINGおよびモニターを提供するいくつかの機能が含まれています。それらの機能を使用すると、CICS システムでのさまざまなリソースの使用を計測しやすくなります。

最もよく使用されるツールは、以下のとおりです。

- 統計データ。

CICS 統計は、CICS システムをモニターするための最も単純なツールです。これには、そのパフォーマンスやリソースの使用など、CICS システム全体の情報が含まれます。CICS 統計は、パフォーマンスの調整やキャパシティー・プランニングに適しています。CICS は、オンライン処理の際の統計を収集し、それを後でオフラインで処理します。統計ドメインはこのデータを収集し、MVS によって提供されるシステム管理機能 (SMF) データ・セットにレコードを書き込みます。レコードは SMF のタイプ 110 から構成されます。DFHSTUP プログラムを使用してこれらのレコードをオフラインで処理することができます。

- モニター・データ。

CICS モニターは、後でオフライン分析を行うために、オンライン処理中にすべてのユーザーおよび CICS 提供トランザクションに関するデータを収集します。CICS モニターによって生成されるレコードも SMF のタイプ 110 であり、SMF データ・セットに書き込まれます。CICS モニターによって提供されるデータは、パフォーマンスを調整する場合、およびユーザーが使用するリソース料金をユーザーに課金する場合に役立ちます。モニターは、以下のデータのクラスを提供します。

- 詳しいトランザクション・レベルの情報に関するパフォーマンス・クラス
- トランザクションがアクセスした個々のリソースに関する、トランザクション・レベルの追加情報についてのトランザクション・リソース・データ
- 例外条件の例外クラス

CICS はマルチタスキングのアドレス・スペースであり、CICS モニター機能は一般に、CICS によって実行される個々のトランザクションまたは機能によって消費されるプロセッサ時間およびその他のリソースを判別するために使用されます。

CICS モニター機能に関する詳しい説明や、この情報を活動化、収集、および処理する方法について詳しくは、CICS モニター機能: パフォーマンスおよび調整を参照してください。

統計データとモニター・データのどちらについても、オフライン処理機能を使用できます。CICS Performance Analyzer および Tivoli® Decision Support for z/OS

はどちらも、CICS およびその他の IBM システムおよび製品からのデータを収集し、分析するツールです。これらを使用することで、以下の点で役立つレポートを作成することができます。

- システム概要
- サービス・レベル
- 可用性
- パフォーマンスおよび調整
- キャパシティー・プランニング

詳しくは、CICS Performance Analyzer for z/OS (CICS PA)および「パフォーマンスの改善」の『z/OS の Tivoli 決断サポート』を参照してください。

Db2 提供のアカウンティングおよびモニター情報

Db2 の計測機能コンポーネントでは、6 つのタイプのトレースを使用することができます。統計、アカウンティング、監査、パフォーマンス、モニター、グローバルの 6 つです。各トレースのタイプでは、いくつかのトレース・クラスを活動化することができます。

トレース出力の宛先として SMF を使用することができます。別の代替手段として、トレース出力を GTF の制御下に外部化する方法があります。

統計 Db2 で実行される作業の合計を記述します。この情報は、いかなる特定ユーザーとの関連性もありません。Db2 統計トレースの主な目的は、以下のとおりです。

- Db2 キャパシティー・プランニングのためのデータを提供します。
- Db2 サブシステム・レベルでのモニターと調整を支援します。
- Db2 アクティビティーのアカウンティングを支援します。

統計レコードはユーザー定義の間隔で書き込まれます。MODIFY TRACE コマンドを使用することにより、トレースを停止して開始せずに統計収集間隔および開始時刻をリセットすることができます。統計収集間隔での Db2 アクティビティーはすべてレコードに報告されます。これにより、アクティビティーを特定ユーザーに直接関連付けることが難しくなります。

複数のクラスの Db2 統計トレースを活動化することができます。統計レコードが SMF に書き込まれる場合、SMF タイプは 100 および 102 です。

アカウンティング

特定ユーザー (DB2CONN または DB2ENTRY からの許可 ID) のために実行される作業について記述します。アカウンティング・レコードの主な目的は、Db2 コストを許可 ID に課金し、プログラム・レベルでモニターおよび調整を実行することです。Db2 はスレッドの終了時、またはトランザクションが新規許可 ID でスレッドを再利用するときにアカウンティング・レコードを生成します。つまり、スレッドが保護として定義され (PROTECTNUM>0)、この DB2ENTRY の同じトランザクション・コードを持つすべてのトランザクションが同じ許可 ID を使用する場合、アカウンティング・レコードは 1 つだけ生成され、スレッドで行われたすべてのアクティビティーがそこに記述されます。さらに、DB2ENTRY 定義または DB2CONN 定義の ACCOUNTREC を UOW、TASK、または TXID に設

定する場合、アカウントティング・レコードが書き込まれます。
ACCOUNTREC をこれらのオプションに設定する場合、同じ許可 ID を使用するとしても、サインオンと見なされます。

複数のクラスの Db2 アカウンティング・トレースを活動化することができます。アカウントティング・レコードが SMF に書き込まれる場合、SMF タイプは 101 および 102 です。

監査 Db2 セキュリティー管理に関する情報を収集します。これを使用することで、許可された目的でのみデータ・アクセスが行われるようにします。監査レコードが SMF に書き込まれる場合、SMF タイプは 102 です。

パフォーマンス

いくつかの異なるイベント・クラスに関する情報を記録します。この情報は、以下の目的で提供されます。

- プログラム関連のモニターおよび調整
- リソース関連のモニターおよび調整
- ユーザー関連のモニターおよび調整
- システム関連のモニターおよび調整
- アカウンティング関連のプロファイルの作成

複数のクラスの Db2 パフォーマンス・トレースを活動化することができます。パフォーマンス・レコードが SMF に書き込まれる場合、SMF タイプは 102 です。

モニター

ユーザー作成のプログラムを使ってオンライン・モニターのデータを記録します。

グローバル

保守を行いやすくします。グローバル・トレース・レコードが SMF に書き込まれる場合、SMF タイプは 102 です。

CICS Db2 環境のモニター: 概要

CICS Db2 接続機能のモニターの目的は、アカウントティングおよび調整の基礎を提供することにあります。

このタスクについて

モニターを使用して、以下のデータを入手できます。

- Db2 リソースにアクセスするトランザクションの数。
- トランザクションによって発行される SQL ステートメントの平均数。
- トランザクションの平均プロセッサ使用量。
- トランザクションの平均応答時間。
- 特定のトランザクションと関連付けられているコスト。
- トランザクションと関連付けられているバッファー・プール・アクティビティ。

- トランザクションと関連付けられているロック・アクティビティ。これには、ページ・ロックの代わりにテーブル・スペース・ロックが使用されるかどうか、また (例えば反復可能読み取りが原因で) ロック・エスカレーションが発生するかどうかが含まれます。
- DB2ENTRY およびプールのスレッド使用量のレベル。
- DB2ENTRY の保護スレッドのスレッド再利用のレベル。

以下の目的で、テスト環境をモニターすることをお勧めします。

- 新規プログラムがテスト・データベースに対して正しく機能している (つまり、正しい呼び出し順序を使用している) ことを確認する。
- 過剰な入出力操作または非効率的な SQL ステートメントが原因でパフォーマンス上の問題が生じている場合にそれを検出する。
- 設計プラクティスの問題 (画面对話をまたいで Db2 リソースを保持するなど) を検出する。
- アプリケーション分離要件と既存のアプリケーション要件のバランスを取るために、最適なロック・プロトコルをセットアップする。

新規アプリケーションで受け入れられる手順の中に、モニターを含めてください。テスト期間中に検出されなかった問題があれば、それを素早く特定して訂正できるようにするためです。

CICS Db2 接続機能および Db2 リソースにアクセスする CICS トランザクションをモニターするため、以下のツールのうちのいくつか、またはすべてを使用できます。

- CICS Db2 接続機能をモニターする際に使用するツールは、以下のとおりです。
 - CICS Db2 接続機能コマンド
 - Db2 コマンド
 - CICS Db2 統計

151 ページの『CICS Db2 接続機能のモニター』を参照してください。

- CICS トランザクションをモニターする際に使用するツールは、以下のとおりです。
 - CICS モニター機能 (CMF)
 - CICS 補助トレース

155 ページの『Db2 リソースにアクセスする CICS トランザクションのモニター』を参照してください。

- Db2 をモニターする際に使用するツールは、以下のとおりです。
 - Db2 統計レコード
 - Db2 アカウンティング・レコード
 - Db2 パフォーマンス・レコード

156 ページの『CICS で使用されている場合の Db2 のモニター』を参照してください。

- CICS システム (例えば、ディスパッチャー) のモニターは、CICS 統計を使用して行います。160 ページの『CICS Db2 環境での CICS システムのモニター』を参照してください。

CICS Db2 接続機能のモニター

Db2 と CICS Db2 接続機能自体の両方にアドレス指定したコマンドを使用して、CICS Db2 接続機能をモニターします。

このタスクについて

CICS Db2 接続機能コマンドを使用した CICS Db2 接続機能のモニター

CICS Db2 接続機能が提供する DSNB DISPLAY PLAN または TRAN コマンドを使用して、CICS-Db2 スレッドの状況、およびこれらのスレッドを使用する CICS トランザクションをモニターできます。

このタスクについて

詳細については、CICS Db2 用の CICS 提供トランザクションを参照してください。DB2CONN、または個々の DB2ENTRY では、CICS が提供するコマンド (CEMT または EXEC CICS INQUIRE コマンドなど) を使用することもできます。これらのコマンドを DB2CONN で使用する場合、CICS と Db2 との間の全体的な接続状況およびプールの使用状況をモニターできます。これらのコマンドを個々の DB2ENTRY で使用する場合には、DB2ENTRY の使用をモニターできます。

Db2 コマンドを使用した CICS Db2 接続機能のモニター

CICS と Db2 との間の接続が確立した後は、CICS セキュリティーで許可された端末ユーザーが DSNB トランザクションを使用して、コマンドを Db2 システムにルーティングできます。

このタスクについて

これらのコマンドの形式は、次のとおりです。

DSNB-DB2command

例えば、DSNB -DIS THREAD コマンドでは CICS Db2 スレッドを表示できます。

コマンドは、処理のために Db2 にルーティングされます。Db2 は、CICS から渡された許可 ID が、入力されたコマンドの実行を許可されているかどうかをチェックします。

応答は、送信側の CICS ユーザーにルーティングされます。Db2 コマンドを CICS Db2 接続機能のコマンドから区別するために、コマンド認識文字 (CRC) としてハイフン (-) を使用する必要があります。CICS から発行される Db2 コマンドの場合、サブシステム認識文字とは関係なく、CRC は常に - となります。

CICS 端末から Db2 コマンドを発行するには、CICS 許可と Db2 許可の両方が必要です。

- CICS 許可は、DSNB トランザクションを使用するために必要です。
- Db2 許可は、Db2 コマンドを発行するために必要です。

詳しくは、CICS Db2 用の CICS 提供トランザクションを参照してください。

CICS Db2 統計を使用した CICS Db2 接続機能のモニター

DSNC DISP STAT コマンドによる限られた統計出力、および接続機能終了時の DB2CONN の STATSQUEUE 宛先への出力に加えて、標準の CICS 統計インターフェースを使用して、以下のようなより包括的な CICS Db2 統計セットを収集できます。

このタスクについて

CICS Db2 グローバル統計およびリソース統計については、『Reference』の『CICS Db2 統計』で詳細を説明しています。

CICS Db2 統計は、すべてのタイプの CICS 統計に対してサポートされています。具体的には以下のタイプの統計です。

- 要求統計 - CICS Db2 要求は、Db2 をキーワードとして指定した **EXEC CICS PERFORM STATISTICS RECORD** コマンドの結果として書き込まれます。
- 要求されたりセット統計 - 要求統計の特殊ケースです。この場合、統計の収集後に統計カウンターがリセットされます。
- 間隔統計 - 要求された間隔の満了時に、統計が書き込まれます。
- 終業時統計 - 間隔統計の特殊ケースです。
- 非送信請求統計 - 接続機能のシャットダウン時に、CICS が Db2 グローバル統計およびリソース統計を SMF に書き込みます。また、DB2ENTRY が破棄されると、Db2 リソース統計を SMF に書き込みます。

手順

1. Db2 統計を収集するには、以下のいずれかの方法を使用します。
 - CICS Db2 グローバル統計の収集を可能にする DB2CONN キーワードを指定した **EXEC CICS COLLECT** 統計コマンドを使用します。CICS Db2 グローバル統計は、DFHD2GDS DSECT によってマップされます。
 - 特定の DB2ENTRY に関する CICS Db2 リソース統計の収集を可能にする DB2ENTRY() キーワードを指定した **EXEC CICS COLLECT** 統計コマンドを使用します。CICS Db2 リソース統計は、DFHD2RDS DSECT によってマップされます。
 - ユーザーによる SMF への CICS Db2 グローバル統計およびリソース統計の書き出し要求を可能にする Db2 キーワードを指定した **EXEC CICS PERFORM STATISTICS** コマンドを使用します。

あるいは、CICS サンプル統計プログラム DFH0STAT を使用して Db2 統計を収集することもできます。このプログラムは、統計を収集するための DB2CONN および DB2ENTRY キーワードを指定した **EXEC CICS COLLECT STATISTICS** コマンドを使用します。また、データを収集する DB2CONN および DB2ENTRY に対して **EXEC CICS INQUIRE** コマンドも使用します。 154 ページの図 30 に、DFH0STAT による出力例を記載します。

2. 統計を収集した後、パフォーマンスおよびチューニングに関して調べなければならない重要なフィールドには以下があります。
 - a. DB2ENTRY またはプールからのスレッドを使用して行われた呼び出しの数 (つまり、既存のスレッドが再利用された回数)。スレッドの再利用は、DB2ENTRY ごと、および (Db2 接続統計で) プールごとに報告されます。

スレッドを再利用すると、CICS トランザクションごと、または作業単位ごとにスレッドを作成することによって発生するオーバーヘッドが回避されるため、パフォーマンスに有利です。保護されたスレッドの使用によって、スレッドの再利用を増やすことができます。

- b. THREADWAIT(YES) を指定した場合、readyq 上のスレッドを待機中のタスクのピーク数。トランザクションがキューに入ってスレッドを待機できるようにするよりも、トランザクション・クラスを使用してトランザクション数を制限したほうが有利です。
- c. Db2 接続統計で、「プール Readyq 上のタスクのピーク数」フィールドと、「TCB Readyq 上のタスクのピーク数」フィールドを調べます。後者がゼロでない場合、タスクはスレッドを待機する代わりに、オープン TCB で使用する Db2 接続を待機するためにキューに入れられたことになります。タスクがキューに入れられた理由は、TCBLIMIT (Db2 に入るスレッドを制御するために使用可能な TCB の最大数) に達したためです。これは、使用可能なスレッドの数 (プール、コマンド・スレッド、およびすべての DB2ENTRY の THREADLIMIT 値の合計) が、使用可能な TCB 数を超過していることを示します。この場合、TCBLIMIT または THREADLIMIT の値を調整する必要があります。

パフォーマンスをモニターし、アカウントリング情報を収集するために、Db2 アカウンティング機能を持つ CICS Db2 環境で CICS モニターを使用します。

CICS パフォーマンス・クラス・レコードと Db2 アカウンティング・レコードの突き合わせについて詳しくは、165 ページの『Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの関連付け』を参照してください。プロセッサ使用量の計算について詳しくは、173 ページの『CICS Db2 環境でのプロセッサ使用量を考慮に入れる』を参照してください。

CICS アプリケーション・プログラムが発行する SQL 呼び出しを追跡するには、CICS 補助トレース機能を使用できます。

CICS Db2 接続機能によるトレース出力について詳しくは、Db2 のトラブルシューティングを参照してください。

CICS で使用されている場合の Db2 のモニター

CICS での Db2 使用量をモニターするには、Db2 によって生成される SMF または GTF レコードを使用します。

このタスクについて

Db2 Performance Monitor (DB2PM) プログラム製品は、以下に基づくレポートを提供するのに役立ちます。

- 統計レコード
- アカウンティング・レコード
- Performance records (パフォーマンス・レコード数)

このトピックのレポートは、例として記載されています。レポートに関連するフィールドの形式および意味については、ご使用の DB2PM リリースの資料を参照してください。

Db2 統計機能を使用した Db2 のモニター

Db2 は、インストール時に指定された時間間隔が終わるたびに、サブシステムごとの統計データを生成します。このデータは、この統計機能がアクティブになっている場合にのみ、収集されて SMF および GTF データ・セットに書き込まれます。

このタスクについて

これらの機能をアクティブにして、出力を SMF および GTF に送信する方法について詳しくは、DSNC トランザクションの使用による Db2 へのコマンドの発行および Db2 アカウンティング、統計、および調整のための GTF の開始を参照してください。

システム・サービスのアドレス・スペースに関するデータは、SMF 計測機能構成コンポーネント ID (IFCID) 0001 レコードとして書き込まれます。データベース・サービスのアドレス・スペースに関するデータは、SMF IFCID 0002 レコードとして書き込まれます。これらのレコードの説明については、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

これらの統計は、Db2 に接続されたすべてのサブシステムのアクティビティーを反映することから、Db2 サブシステムをチューニングする際に役立ちます。

複数のサブシステムが Db2 に接続されている場合 (つまり、CICS と TSO の両方が接続されている場合)、このデータを解釈するのは難しくなります。ただし、制御された環境 (つまり、CICS が唯一のサブシステムとして接続されている環境、または TSO アクティビティーが制限されている環境) で CICS Db2 接続機能が実行されている間に取得されたカウントが、大いに役立ちます。

Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』では、データベース・サービスおよびシステム・サービスのアドレス・スペースに関して報告される統計データについて、Db2 の観点から記載および分析しています。ここでは、統計レポートの短縮バージョンを記載します。このレポートを使用して、平均 CICS トランザクションをモニターできます。158 ページの図 32 に、DB2PM によって提供されるレポートの一部を示します。これらのレポートについて詳しくは、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

- サブシステム・サービス。このセクションは、スレッドの使用およびサインオン・アクティビティーに関する情報を提供します。パフォーマンス上の理由から、CICS トランザクションごとにスレッドを作成することによって発生するオーバーヘッドを回避するために、CICS 環境ではスレッドの再利用が推奨されています。

158 ページの図 32 には記載されていませんが、統計レポートには、以下の有用な情報も提供されます。

- ロック を使用して、タイムアウトおよびデッドロックの回数をモニターできます。デッドロックについて詳しくは、CICS Db2 環境でのデッドロックの処理を参照してください。
- 「バッファ・プール」は、以下の情報を提供します。
 - オープンされたデータ・セット数（「オープンされたデータ・セット数」）
 - 取得されたページ数（「GETPAGE 要求数」）
 - 入出力数（「読み取り操作数」および「書き込み入出力操作数」）

この統計データは、チェックポイント対象ではないため、Db2 を再始動すると失われます。

Db2 アカウンティング機能を使用した Db2 のモニター

単一のトランザクションに対する Db2 アカウンティング機能の出力は、モニターおよび調整のために使用できます。

このタスクについて

Db2 アカウンティング機能の使用については、164 ページの『Db2 アカウンティング・レポート』を参照してください。

Db2 パフォーマンス機能を使用した Db2 のモニター

Db2 パフォーマンス機能のトレースは、Db2 内の制御フローに関する詳細な情報を提供します。

このタスクについて

このトレースの主な目的は、デバッグ情報を提供することですが、各エントリーにはタイミング・データが併せて提供されることから、このトレースをモニター・ツールとして使用することもできます。

リソース消費量が高いことから、Db2 パフォーマンス・トレースは、他のツールを使用して Db2 関連のトランザクションをモニターすることが困難な場合にのみ使用してください。

使用する場合でも、時間を制限して使用し、慎重にモニターしなければならないトランザクションに対してのみ、パフォーマンス・トレースの必要なクラスだけを開始してください。

CICS Db2 環境での CICS システムのモニター

CICS ディスパッチャー統計を使用して、CICS Db2 TCB のアクティビティをモニターできます。

このタスクについて

CICS ディスパッチャーによって提供される統計の使用について詳しくは、CICS ディスパッチャー: パフォーマンスおよびチューニングを参照してください。

例えば、ディスパッチャー統計のセクションにあるデータは、CICS TCB ごとの累積時間を示します。「Accum/TCB」フィールドは、対応する TCB が使用したプロセッサ時間の合計です。CICS ディスパッチャー統計について詳しくは、ディスパッチャー・ドメイン統計を参照してください。

CICS アドレス・スペースで使用された合計プロセッサ時間は、RMF™ モニター II レポートから取得できます。この時間は、通常、すべての CICS タスク TCB の合計より大きくなります。

RMF によってレポートされるプロセッサ時間と、CICS ディスパッチャー統計レポートでの CICS TCB の合計との差は、他のすべてのサブタスク TCB によって消費されたプロセッサ時間です。これらのサブタスクの使用は、以下に該当します。

- DBCTL スレッドで消費されたプロセッサ時間
- IBM MQ スレッドで消費されたプロセッサ時間

これらのグローバル・パフォーマンス・レポートは、CICS によって使用されているプロセッサ時間の量を判別するのに役立ちます。

CICS Db2 環境でのアカウンティング: 概要

CICS Db2 環境でのアカウンティングを使用して、システムで実行されているトランザクションを分析できます。また、特定のトランザクション一式のために消費されたリソースの合計量を、明確に定義されたユーザーの集合に課金することもできます。ユーザーは実ユーザー、ユーザーのグループ、トランザクション、またはリソースを割り振る必要がある単位を示す他の任意の表現にすることができます。

通常、消費単位にはプロセッサ、入出力、および主記憶域が含まれ、これらの消費単位の比率には何らかの重みが付けられます。Db2 にアクセスする標準的な CICS トランザクションは、オペレーティング・システム、CICS システム、アプリケーション・コード、および Db2 アドレス・スペースのリソースを消費します。これらのコンポーネントのそれぞれがデータを生成でき、そのデータをアカウンティング処理への入力として使用できます。異なるソースからの出力を結合することで、トランザクションにおけるリソース使用量の全体像を把握することができます。

アカウンティング・プロシージャの一般的要件は、計算された結果が反復可能であることです。これは、一連のデータにアクセスするトランザクションのコスト

が、そのトランザクションが実行されるたびに同じであることを意味します。ほとんどの場合、これはアカウント処理への入力データが反復可能でなければならないことも意味します。

CICS Db2 環境のアカウント処理方針を計画する際には、以下の必要事項があります。

- アカウント処理で使用する Db2 アカウント処理・データのタイプ (プロセッサ使用量、入出力、呼び出しの数など) を決定します。『Db2 アカウント処理機能によって提供されるアカウント処理情報』で、Db2 アカウント処理機能から取得できるアカウント処理・データについて説明しています。
- トランザクションのリソース使用量の全体像を把握するために、各トランザクションに関する Db2 アカウント処理・レコードのデータを、そのトランザクションの CICS パフォーマンス・クラス・データに関連付ける方法を決定します。165 ページの『Db2 アカウント処理・レコードと CICS パフォーマンス・クラス・レコードとの関連付け』で、この 2 つのタイプのデータを突き合わせる方法を説明しています。
- トランザクションごとの CICS パフォーマンス・レコードと Db2 アカウント処理・レコードを特定のユーザーに関連付けるか、またはいくつかのモデル・トランザクションを定義して調整し、これらのトランザクションを制御された環境で測定して、各ユーザーが実行したモデル・トランザクションの数だけをカウントするかどうかを決定します。169 ページの『Db2 アカウント処理・レコードと CICS パフォーマンス・クラス・レコードを突き合わせてユーザーにリソースを課金する場合の方針』では、どのような場合にそれぞれの方式が最も適しているか、提案が載せられています。

プロセッサ使用量をアカウント処理の基準として使用することにした場合、Db2 アカウント処理・レコードにレポートされるプロセッサ時間のさまざまなクラス、およびトランザクションで利用したプロセッサ時間の合計を計算する方法について詳しくは、173 ページの『CICS Db2 環境でのプロセッサ使用量を考慮に入れる』を参照してください。

Db2 アカウント処理機能によって提供されるアカウント処理情報

Db2 アカウント処理機能は、CICS トランザクションによる Db2 リソースの使用に関する詳しい統計を提供します。Db2 アカウント処理・レコードを CICS トランザクションによって利用される Db2 リソースのアカウント処理および調整の基礎として利用することができます。

Db2 は、許可 ID のアカウント処理・データをスレッド単位で収集します。要求されるとアカウント処理機能はこのデータを収集し、スレッドが終了した時か許可 ID が変更された時に SMF、GTF、またはその両方に送ります。Db2 アカウント処理機能の活動化および SMF および GTF への出力の送信について詳しくは、Db2 アカウント処理、統計、および調整のための SMF の開始および Db2 アカウント処理、統計、および調整のための GTF の開始を参照してください。Db2 SMF および GTF レコードの一般的な構造について詳しくは、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

SMF および GTF に書き込まれる各 Db2 アカウンティング・レコードの ID セクションには、データをソートおよび要約する際に使用できるいくつかのキーがあります。これには、許可 ID、トランザクション ID、計画名、およびパッケージ名が含まれます。

Db2 Performance Monitor (DB2PM) プログラム製品は、Db2 アカウンティング・レコードから取られたアカウンティング・レポートを提供します。

Db2 アカウンティング・レコードのデータ・タイプ

このセクションでは、Db2 アカウンティング・レコードのさまざまなデータ・タイプの概要を示します。

Db2 アカウンティング・レコードに基づくコストの公式を定義する場合、いくつかの可能性があります。

- トランザクションの再現性と、トランザクションの複雑さに関するある程度の情報を得ることを優先する場合は、プロセッサ使用量、GETPAGE カウント、および Setwrite Intent カウントが有力な候補です。
- アカウンティング処理の目的が CICS トランザクションの動作を分析することであれば、Db2 アカウンティング・レコードのいずれかの情報を使用することができます。

Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』では、Db2 アカウンティング・レコードの個々のフィールドについて詳しく説明されています。

プロセッサ使用量

通常、Db2 アカウンティング・レコードに示されるプロセッサ使用量の情報には、SQL 呼び出しに使用された合計プロセッサ時間の大半が含まれます。

Db2 統計レコードは、個々のスレッドに直接関連付けることができなかった Db2 アドレス・スペース内で使用されたプロセッサ時間を報告します。

Db2 統計レコードに報告されたプロセッサ時間を、Db2 サブシステムのすべてのユーザー（トランザクション、バッチ・プログラム、TSO ユーザー）に均等に分配することを検討してください。

Db2 アカウンティング・レコードに報告されるプロセッサ時間は、一定の期間で（同じ処理に対して）比較的反復可能です。

CICS Db2 環境でのプロセッサ使用量のレポートについて詳しくは、173 ページの『CICS Db2 環境でのプロセッサ使用量を考慮に入れる』を参照してください。

入出力

Db2 システムでは、入出力を以下の 5 つのタイプに分類できます。

- 同期読み取り入出力
- 順次プリフェッチ（非同期読み取り）
- 非同期書き込み
- EDM プール読み取り（DBD および計画セグメント）

- ログ入出力 (主に書き込み)

この 5 つの入出力タイプのうち、同期読み取り入出力だけは Db2 アカウンティング・レコードに記録されます。

順次プリフェッチ読み取り要求の数も報告されますが、読み取り要求の数と入出力の数は同じではありません。

いずれの入出力タイプも、一定の期間で反復可能であるとは見なさないでください。これらはすべて、バッファ・サイズおよびワークロード・アクティビティーに依存します。

Db2 は、使用されているキャッシュを認識しません。つまり、キャッシュ・バッファが要求を満たすとしても、Db2 は入出力の発生数を報告します。

GETPAGE

GETPAGE は、Db2 がバッファ・マネージャーにページを要求した回数を示します。

GETPAGE が Db2 アカウンティング・レコード内で表す数は、同一のトランザクションに対しては、一定の期間にわたり、かなり一定しています。これは、Db2 がバッファ・マネージャーにページを要求した回数を示します。Db2 がページのデータを読み取るか、またはページにデータを書き込む必要があるときには、常にそのページを使用可能にする必要があります。この場合、そのページに対して、少なくとも 1 つの GETPAGE がカウントされます。このことは、索引およびデータ・ページの両方に当てはまります。複数回使用された特定のページに対して GETPAGE カウンターがどれだけ増分されるかは、選択されたアクセス・パスに依存します。ただし、同じデータにアクセスする同じトランザクションに対する GETPAGE の数は一定の期間にわたってかなり一定していますが、Db2 のリリースの間で、GETPAGE アルゴリズムが変わる可能性があります。

要求されたページがバッファ・プール内にある場合、入出力は発生しません。ページがバッファ内でない場合、バッファ・マネージャーはメディア・マネージャーからのページを要求します。この場合には、入出力が発生します。

したがって、GETPAGE の数は、SQL 要求を実行するために必要な Db2 内のアクティビティーの指標となります。

書き込みインテント

設定されている書き込みインテントの数は、Db2 アカウンティング・レコードの QBACSW ฟิลด์に保持されます。

設定されている書き込みインテントの数は、バッファ・プールの書き込み入出力の実際の数とは関係しません。これは、ページに更新のマークが付けられた回数を表します。読み取り専用トランザクションでも、この数値が存在する場合があります。Db2 のソートで使用される一時作業ファイルを対象とした書き込み操作もカウントされるためです。

典型的な事例は、設定されている書き込みインテントの数が、書き込み入出力の数を遥かに上回っている場合です。この 2 つの数値の比率は、バッファ・プールの

サイズとワークロードに依存します。書き込み入出力アクティビティーに有効な測定ではありませんが、トランザクションの複雑さは、この値に示されます。

SQL 呼び出しアクティビティー

Db2 アカウンティング・レコードには、トランザクションで実行された SQL 呼び出しの数およびタイプが報告されます。

複雑なプログラムによる多種多様なパスが考えられる場合、あるいはアクセス・パスが変更された場合を除き、SQL 呼び出しアクティビティーの値は反復可能です。選択されるアクセス・パスは、時間が経つにつれ、変わる可能性があります (例: 索引の追加)。

ある特定の SQL 呼び出しは、選択されたアクセス・パスや、要求に関連するテーブルおよび行の数などの要因によって、単純にも複雑にもなります。

GETPAGE の数は、ほとんどの場合、異なる SQL 呼び出しの数よりも正確な Db2 アクティビティーの指標となります。

トランザクションの発生

単純なアカウンティング方法は、実行されたトランザクションの数とタイプを追跡することです。

ストレージ

Db2 アカウンティング・レコードには、トランザクションの実行に関連した実ストレージや仮想ストレージに関する情報は含まれません。Db2 サブシステムの目的の 1 つは、ストレージの使用を最適化することです。この最適化は、トランザクション・レベルではなく、Db2 レベルで行われます。

トランザクションは Db2 サービスを要求する際に、複数の場所にあるストレージを使用します。最も重要な場所は、スレッド、EDM プール、およびバッファーク・プールです。

Db2 アカウンティング・レコードには、ストレージ使用量に関する情報が提供されません。しかも、ストレージの使用はサブシステム・レベルで最適化されることから、Db2 環境のストレージを把握するのは困難です。

Db2 アカウンティング・レポート

Db2 Performance Monitor (DB2PM) プログラム製品は、Db2 アカウンティング・レコードから取られたアカウンティング・レポートを提供します。

165 ページの図 33 および 165 ページの図 34 に、DB2Db2 リソースにアクセスする CICS トランザクションの長形式および短形式のアカウンティング・レポートの例を記載します。

アカウンティングではなく、パフォーマンス分析を行うためにリソース使用量を調べる場合には、必ず Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせる必要があります。ただし、アカウンティングの目的でリソース使用量を調べている場合には、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせる必要はありません。

アカウンティング基準としてプロセッサ時間の消費量を使用している場合、Db2 シスプレックス照会並列処理 (並列照会) を使用していなければ、レコードを突き合わせる必要はありません。この場合、並列照会が使用されていない場合は、Db2 で消費されたプロセッサ時間が、CICS パフォーマンス・クラス・レコードならびに Db2 アカウンティング・レコードで報告されます。したがって、CICS パフォーマンス・クラス・レコードでトランザクションについて調べることで、そのトランザクションのリソースに対してユーザーに課金するために必要なプロセッサ時間に関する情報のすべてが分かります。この状態に当てはまるとしたら、このセクションの残りの部分をスキップして、代わりに (プロセッサ時間消費量がどのように報告されるのかを理解するには) 173 ページの『CICS Db2 環境でのプロセッサ使用量を考慮に入れる』のセクション、および (入手可能なプロセッサ時間に関する情報の利用方法を調べるには) 181 ページの『Db2 の CICS および Db2 プロセッサ時間の計算』のセクションを読んでください。

アカウンティングの基準として、プロセッサ時間消費量のデータも併せて使用することにした場合、あるいはプロセッサ時間消費量以外のデータを使用することにした場合には、このセクションに関連する概念について読み、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせる方法を調べてください。アカウンティングにプロセッサ時間消費量を使用している場合には、対応する Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを使用してプロセッサ時間消費量を計算する方法について、173 ページの『CICS Db2 環境でのプロセッサ使用量を考慮に入れる』のセクションを読んでください。

Db2 アカウンティング・レコードと CICS パフォーマンス・レコードを突き合わせる際の問題

CICS と Db2 のアカウンティングのニーズは異なるため、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードは、必ずしも簡単に突き合わせられるわけではありません。

これらのレコードを突き合わせる際には、以下の 2 つの主要な問題があります。

1. Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードは、必ずしも 1 対 1 の関係ではありません。Db2 アカウンティング・レコードには、1 つの CICS トランザクション、複数の CICS トランザクション、あるいは CICS トランザクションの一部に関する情報が含まれる場合があります。
2. Db2 アカウンティング・レコードのフィールドは、対応する CICS パフォーマンス・レコードと完全に一致するわけではありません。

リソースに対してユーザーに課金する目的で、DB2ENTRY または DB2CONN の AUTHTYPE パラメーターに OPID、USERID、GROUP、または TERM を指定して、ユーザーに Db2 の視点からは異なる許可 ID を割り当てることができます。

この場合、生成される Db2 アカウンティング・レコードには、許可 ID に対応するデータだけが含まれます。したがって、許可 ID 別に Db2 アカウンティング・レコードのすべてを収集し、消費されたリソースを直接そのユーザーに課金することができます。この方式は、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせる必要がないことを意味します。ただし、使いやすさとパフォーマンスの観点からは、OPID、USERID、GROUP、および TERM を使用するのとは魅力的なソリューションではありません。その理由については、計画へのユーザー・アクセスの制御で説明されています。大規模ネットワークの場合、これらの許可 ID を指定すると、保守が複雑になり、パフォーマンス・オーバーヘッドが発生する原因になります。許可 ID の使用についてはパフォーマンスを考慮して計画し、Db2 アカウンティング・レコードをエンド・ユーザーに割り当てるには、CICS パフォーマンス・クラス・レコードと突き合わせるという方法を採用するほうが推奨されます。

このセクションで説明する内容は以下のとおりです。

- Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係を単純化するための方法
- 対応する CICS パフォーマンス・クラス・レコードを識別するために使用できる Db2 アカウンティング・レコード内の情報
- 4 つの代表的なシナリオで、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせるための方針

Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係の制御

デフォルトでは、Db2 はそのアカウンティング・レコードを、スレッド終了時、またはスレッドを再利用する新しい許可 ID のサインオン時に書き込みます。

このタスクについて

スレッドが、そのスレッドを使用した前のトランザクションと同じ CICS トランザクション ID および Db2 許可 ID を持つトランザクションによって再利用される場合、Db2 は再利用時にアカウンティング・レコードを書き込みません。(CICS トランザクション ID と Db2 許可 ID との関係については、CICS 領域用および CICS トランザクション用に、許可 ID を DB2 に提供するを参照してください)。これは、スレッドに関する各 Db2 アカウンティング・レコードに、複数の CICS トランザクションに関する情報が含まれる可能性があることを意味します。さらに、異なるタイプの CICS トランザクションが同じトランザクション ID を使用して Db2 にアクセスした場合には、Db2 アカウンティング・レコードに、異なるタイプの CICS トランザクションに関する情報が含まれることになります。

これらの問題に対処するために、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係を制御するには、以下の 3 つの方法があります。

- CICS トランザクション ID と Db2 許可 ID のそれぞれが、常に、同じリソースを消費する同じ処理を表すように、CICS アプリケーションを設計するという方法があります。これにより、各 Db2 アカウンティング・レコードに、単一の処理、または同じ処理の複数のオカレンスが含まれ、異なる項目は含まれないことが確実になります。そのような Db2 アカウンティング・レコードに複数の項

目が含まれるとしても、これらの項目はまったく同じであるため、リソースの使用量を項目間で均等に分けることができます。ただし、アプリケーションをこのように設計するのは実際的でない場合もあります。例えば、端末ユーザーの端末にメニューが表示され、次のトランザクションに (異なる処理に関連する) 異なるオプションを選択できる場合を考えてください。EXEC CICS RETURN TRANSID(zzzz) コマンドで CICS トランザクション ID をセットアップすることによって前のトランザクションが終了した場合、端末ユーザーがどのような処理を選択するかに関わらず、次のトランザクションはトランザクション ID zzzz で実行されます。このようなアプリケーションをアカウントティングを目的に再設計しようとは思わないはずです。

- スレッドが再利用されないようにするという方法があります。これによって確実に、各タスクの終了後に、スレッドが終了し、Db2 がアカウントティング・レコードを書き込みます。したがって、各 Db2 アカウントティング・レコードは確実に単一のタスクを表すことになります。ただし、この方法を採用することにより、スレッドの再利用がもたらす大きなパフォーマンス上の利点を失います。さらに、異なるタイプのトランザクションが同じトランザクション ID を使用して Db2 にアクセスした場合、各 Db2 アカウントティング・レコードは依然として、考えられる複数のタスクのいずれかを参照している可能性があります。
- DB2CONN または DB2ENTRY 定義に ACCOUNTREC(TASK) を指定して、CICS タスクが Db2 リソースの使用を終了するたびに、Db2 にアカウントティング・レコードを生成させることができます。ACCOUNTREC(UOW) ではなく、ACCOUNTREC(TASK) を指定することをお勧めします。これにより、各タスクに少なくとも 1 つの識別可能な Db2 アカウントティング・レコードが存在し、その Db2 アカウントティング・レコードには複数のタスクが含まれていないことが確実にになります。また、同じトランザクション ID を使用した異なるタイプのトランザクションが Db2 にアクセスするという問題を解決するには、ACCOUNTREC(TASK) を指定します。この場合、CICS は、アカウントティング・レコードに含める LU6.2 トークンを Db2 に渡します。このトークンを使用して、各 Db2 アカウントティング・レコードを該当する CICS トランザクションと突き合わせることができます。通常、Db2 アカウントティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係を制御するには、ACCOUNTREC(TASK) が最も実際的かつ完全なソリューションです。これにより、各トランザクションにオーバーヘッドがもたらされますが、一般に、アカウントティングのための有用性はこのオーバーヘッドを上回ります。

ACCOUNTREC(TASK) を指定したとしても、Db2 が単一の CICS タスクを認識できるのは、そのタスクが同じスレッドを使用し続けている場合に限りです。トランザクションに複数の UOW が含まれる場合、UOW の終了時にトランザクションがスレッドを解放するとすれば、トランザクションは UOW ごとに異なるスレッドを使用する可能性があります。これは、複数の同期点 (コミットまたはロールバック) を発行する端末向けトランザクションで起こり得ることです。また、非端末向けトランザクションでも、NONTERMREL(YES) が DB2CONN に設定されている場合には、起こる可能性があります。このような場合、Db2 は複数の UOW を単一のタスクとして認識しないため、UOW ごとにアカウントティング・レコードを生成します。この類のトランザクションには、各 Db2 アカウントティング・レコードにトランザクションの一部に関する情報しか含まれない可能性があるため、トランザクションに関連するすべての Db2 アカウントティング・レコードを識別する必要があります。

Db2 アカウンティング・レコードのデータを使用した、対応する CICS パフォーマンス・クラス・レコードの識別

Db2 アカウンティング・レコードのフィールドを使用して、対応する CICS パフォーマンス・レコードを識別します。

このタスクについて

必要となる可能性がある Db2 アカウンティング・レコードのフィールドは、以下のとおりです。

- CICS LU6.2 トークン。DB2ENTRY または DB2CONN に ACCOUNTREC(TASK) または ACCOUNTREC(UOW) のいずれかを指定すると、CICS はその LU6.2 トークンを Db2 に渡します。Db2 はそのトークンを Db2 トレース・レコードに含めます。トークンは、関連ヘッダーの QWHCTOKN に書き込まれます。このトークンの存在により、2 つのレコード・セットの突き合わせが大幅に単純化されます。
- スレッド関連 ID。これには、4 文字の CICS トランザクション ID が含まれます。ACCOUNTREC(TASK) または ACCOUNTREC(UOW) を指定していない場合、Db2 アカウンティング・レコードには、この ID を使用した複数のトランザクションに関する情報が含まれる可能性があることに注意してください。ACCOUNTREC(TASK) または ACCOUNTREC(UOW) を指定すると、Db2 アカウンティング・レコードにトランザクションの一部 (単一の UOW) しか含まれない可能性があるため、トランザクションに関連するその他すべてのレコードを見つける必要があります。異なるタイプの CICS トランザクションが同じトランザクション ID を使用する場合、この項目だけで正しく識別することはできません。
- 許可 ID フィールド。スレッド関連 ID の場合と同じように、Db2 アカウンティング・レコードには、この許可 ID を使用した複数のトランザクションに関する情報が含まれる可能性と、トランザクションの一部しか含まれない可能性の両方があります。CICS トランザクションが使用する許可 ID は、DB2CONN または DB2ENTRY の AUTHID または AUTHTYPE パラメーターによって決定されます。異なるタイプの CICS トランザクションが同じ許可 ID を使用する場合、この項目だけで正しく識別することはできません。
- タイム・スタンプ・フィールド。スレッドの開始時刻と終了時刻により、Db2 アカウンティング・レコードで扱われている CICS トランザクションの範囲を識別できます。

Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせてユーザーにリソースを課金する場合の方針

Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせるのに最適な方法は 1 つだけではありません。場合によっては、トランザクションが同時に実行されているために、正しい突き合わせを行えない可能性もあります。ただし、ほとんどの場合は、2 つのタイプのレコードをかなり正確に突き合わせるための方針があります。

個々のトランザクションで使用されるリソースがアカウントिंगの基準である場合には、CICS パフォーマンス・クラス・レコードと Db2 アカウンティング・レコードを突き合わせてから、これらのレコードを特定のユーザーに関連付けることができます。あるいは、いくつかのモデル・トランザクションを定義して調整し、これらのトランザクションを制御された環境で測定して、各ユーザーが実行したモデル・トランザクションの数だけをカウントすることもできます。

使用すべき方針を決定する主な要因には、以下の 2 つがあります。

- 各 CICS トランザクション ID が 1 つの可能なトランザクション・パスだけを表す (したがって、常に同じリソース消費量を表す) のか、あるいは多数のトランザクション・パスが同じ CICS トランザクション ID を共用する (したがって、異なるリソース消費量を表す可能性がある) のかどうか。
- 各 Db2 アカウンティング・レコードが、(167 ページの『Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係の制御』で説明したいずれかの措置を取ったために) 1 つのトランザクションまたは 1 つのトランザクションの一部にだけ関連するのか、あるいはそれぞれのそのアカウントING・レコードに複数のトランザクションに関する情報が含まれるのかどうか。

図 35 に、これらの要因の組み合わせにより、Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードを突き合わせる際に直面する可能性のある 4 つの標準的シナリオが作成される仕組みを示します。以降のセクションでは、それぞれのケースでのレコードの突き合わせの方針と、使用したリソースに対してユーザーに課金するための方針を提案します。

	各 DB2 アカウンティング レコードに 1 つのトランザクション	各 DB2 アカウンティング レコードに 多数のトランザクション
各トランザクション パスが含む独自の トランザクション ID を持つ	A	B
多数の異なる トランザクション・パスが 同じトランザクション ID を持つ	C	D

図 35. さまざまなアカウントING・シナリオ

シナリオ A: Db2 アカウンティング・レコードごとに、固有のトランザクション ID を持つ 1 つのトランザクションが含まれる場合

このシナリオでは、各 Db2 アカウンティング・レコードに、単一の識別可能な CICS トランザクションに関する情報だけが含まれることが分かっています。トラン

ザクションが Db2 リソースに複数回アクセスする場合、Db2 はそのトランザクションに対して複数のアカウントティング・レコードを作成する可能性があります。その場合、トランザクションに関連する複数の Db2 アカウントティング・レコードを、そのトランザクションの CICS パフォーマンス・レコードに突き合わせる必要があります。

トランザクションのユーザーは、CICS パフォーマンス・レコードから識別できます。このレコードには、当該トランザクションに関連する CICS アクティビティーが記録されます。このトランザクションに適用する Db2 アカウントティング・レコードを識別するには、169 ページの『Db2 アカウントティング・レコードのデータを使用した、対応する CICS パフォーマンス・クラス・レコードの識別』にリストされているいずれかのデータ項目を使用します。

これらのトランザクションはすべて同一のものであるため、消費するリソース量は同等であると見込むことができます。アカウントティングの目的で、トランザクション・タイプごとにモデル・トランザクションを作成することも可能です。どの Db2 アカウントティング・レコードがどの CICS トランザクションに該当するかを識別できるため、1 つのトランザクションに対する Db2 アカウントティング・レコードと CICS パフォーマンス・レコードを突き合わせてから、これらのアカウントティング・レコードでの Db2 リソース使用量を、以降の同じタイプのトランザクションに割り当てることができます。リソースの使用量が変化した場合に備え、これらのモデルの正当性を定期的に検証する必要があります。

シナリオ B: Db2 アカウントティング・レコードごとに複数のトランザクションが含まれる一方、各トランザクション・タイプに固有のトランザクション ID がある場合

このシナリオでは、各 Db2 アカウントティング・レコードに、複数のトランザクションに関する情報が含まれる可能性があることから、各 Db2 アカウントティング・レコードを、それに対応する CICS パフォーマンス・レコードに直接突き合わせることはできません。ただし、各トランザクション ID が参照するトランザクションのタイプは 1 つしかないため、Db2 アカウントティング・レコードに記録されているトランザクションのタイプを識別することは可能です。

特定の Db2 アカウントティング・レコードに記録されている CICS トランザクションのタイプが 1 つしかない場合は、アカウントティングを目的に、Db2 でのリソース消費量を各トランザクションの間で均等に分けることができます。トランザクションは (定義の上では) ほとんど同一であるため、これは妥当なことです。Db2 アカウントティング・レコードに記録されているコミット数とバックアウト数が、このレコードで扱っている作業単位の数を示します。ただし、167 ページの『Db2 アカウントティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係の制御』で注意しているように、同じトランザクション内の作業単位が異なるスレッドを使用した場合には、同じ Db2 アカウントティング・レコードにそのトランザクションは記録されません。該当するすべての Db2 リソースを各トランザクションに割り当てたことを確認してください。それには、複数の Db2 アカウントティング・レコードの調査が含まれる場合があります。

特定の Db2 アカウントティング・レコードに複数の異なるタイプの CICS トランザクションが記録されている場合 (同じ DB2ENTRY を使用し、したがって同じスレッドを使用するため)、リソースを均等に配分するという方法を使用することはでき

ません。異なるタイプのトランザクションは、異なるリソースを使用する可能性があるためです。この場合、各タイプの CICS トランザクションによる Db2 リソースの使用量を定期的に測定することで、モデル・トランザクションを作成できます。これらの測定値を取るには、スレッドの再利用を一時的に無効にして、生成された Db2 アカウンティング・レコードを調べます。このレコードには、1 つのトランザクションに関する情報だけが含まれます。これらのモデル・トランザクションを使用して、リソースに対してユーザーに課金します。モデル・トランザクションの正当性は、定期的に検証する必要があります。

シナリオ C: Db2 アカウンティング・レコードごとに 1 つのトランザクションが含まれる一方、複数のタイプのトランザクションが同じトランザクション ID を使用する場合

このシナリオでは、各 Db2 アカウンティング・レコードに、単一の CICS トランザクションに関する情報だけが含まれることは分かっていますが、複数のタイプのトランザクションが同じトランザクション ID を使用しているため、特定の Db2 アカウンティング・レコードがどのタイプのトランザクションを示しているのかを特定できません。

シナリオ A のように、トランザクションの 1 つのインスタンスに関するレコードのセットを突き合わせて、これらの数値を再利用することはできません。個々の CICS パフォーマンス・レコードのすべてを、それぞれに対応する Db2 アカウンティング・レコードに突き合わせる必要があります。この作業を行わない限り、各 Db2 レコードがどのタイプのトランザクションを示しているのかを判断することはできません。

それぞれの Db2 アカウンティング・レコードと、それに対応する CICS パフォーマンス・レコードを突き合わせるには、169 ページの『Db2 アカウンティング・レコードのデータを使用した、対応する CICS パフォーマンス・クラス・レコードの識別』にリストされているデータ項目を使用できます。DB2ENTRY または DB2CONN に ACCOUNTREC(TASK) または ACCOUNTREC(UOW) のいずれかを指定した場合、CICS はその LU6.2 トークンを Db2 に渡すため、レコードを簡単に突き合わせるすることができます。そうでない場合は、それぞれのタイム・スタンプを基準にレコードを突き合わせる必要があります。この場合、トランザクションが同時に実行されていると、正しい突き合わせにならない可能性があります。

突き合わせたレコードのセットを使用して、各トランザクションで使用されたリソースに対して、CICS パフォーマンス・レコードで識別されるユーザーに課金できます。

シナリオ D: Db2 アカウンティング・レコードに複数のトランザクションが含まれ、複数のトランザクション・タイプが同じトランザクション ID を使用する場合

このシナリオでは、各 Db2 アカウンティング・レコードに、複数のトランザクションに関する情報があり得るだけでなく、トランザクション ID を使用して、アカウンティング・レコードに記録されているトランザクションのタイプを識別することもできません。

この場合、おそらくレコードを正確に突き合わせることは不可能であるため、このような状況にならないようにすることが最善です。この状況にあることが分かった

場合の最善のソリューションとしては、まず、シナリオ B で説明したようにモデル・トランザクションを作成します。次に、CICS パフォーマンス・レコードに、各トランザクションに固有の ID でマークを付ける方法を見つけます。例えば、ユーザーが、実行中のトランザクションを識別する情報をパフォーマンス・レコードのユーザー・フィールドに提供するという方法が考えられます。このフィールドを使用すれば、この場合のアカウントिंगのために使用するモデル・トランザクションを識別できます。

CICS Db2 環境でのプロセッサ使用量を考慮に入れる

Db2 アカウンティングおよび統計トレース・レコードに、プロセッサ時間の使用量に関する情報が提供されます。これらのレコードを使用して、CICS および Db2 で使用される合計プロセッサ時間を計算することができます。

Db2 アカウンティング・トレース は、CLASS 1、CLASS 2、または CLASS 3 で開始できます。ただし、CLASS 2、CLASS 3、またはそれら両方のクラスの活動化によって収集される情報を外部化するために、CLASS 1 を常にアクティブにしておく必要があります。

Db2 アカウンティング・レコードに報告されるプロセッサ時間は、仮想記憶間サービスを使用する、CICS または Db2 アドレス・スペース内のスレッド TCB 実行コードのための TCB 時間と、CICS 内でスケジュールに入れられる作業のための SRB 時間です。

CLASS 1 (デフォルト) の結果は、通常の実行中にいくつかの Db2 コンポーネントによって累積されるアカウンティング・データになります。次いでこのデータは収集されて Db2 アカウンティング・レコードに書き込まれます。データ収集には、個々のイベント・トレースのオーバーヘッドは関わりがありません。

CLASS 2 および CLASS 3 は、多くの追加のトレース・ポイントを活動化します。これらのイベントの出現はそれぞれ内部的にトレースされますが、外部の宛先には書き込まれません。むしろ、アカウンティング機構がこれらのトレースを使用して、CLASS 2 または CLASS 3 が活動化されるときにアカウンティング・レコードに現れる追加の合計統計を計算します。その情報を外部化するためには、アカウンティング CLASS 1 がアクティブでなければなりません。

CLASS 2 は、'IN DB2' で費やされたデルタ経過時間およびプロセッサ時間を収集し、これをアカウンティング・レコードに記録します。

CLASS 3 は、'IN DB2' で費やされた、I/O 経過時間とロックおよびラッチ中断時間を収集し、これをアカウンティング・レコードに記録します。

Db2 内の CLASS 7 および CLASS 8 は、Db2 でのパッケージ・レベル・アカウンティングと、Db2 でのパッケージ・レベル・アカウンティング待機を収集します。パッケージ・レベル・アカウンティングについては、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

統計トレース は、プロセッサ時間を統計レコードに報告します。報告されるプロセッサ時間は以下のとおりです。

- CICS アドレス・スペースとは非同期で実行している Db2 アドレス・スペース TCB の下での時間。この例として、Db2 ログと、バッファー・プールからの書き込みがあります。
- Db2 アドレス・スペースのもとでスケジュールに入れられた SRB の下での時間。この例として、順次プリフェッチのための非同期読み取りエンジンがあります。

統計レコードで報告される Db2 アドレス・スペースは以下のとおりです。

- データベース・マネージャーのアドレス・スペース
- システム・サービスのアドレス・スペース
- IRLM

CICS Db2 環境では、Db2 アカウンティング・レコードからのプロセッサ時間は、大抵、Db2 統計レコードで報告されるプロセッサ時間よりもずっと大きくなります。そのようになるのは、使用されるプロセッサ時間の大半が、スレッド TCB 自体の中で、また仮想記憶間サービスを使用して Db2 アドレス・スペースの中で費やされるからです。

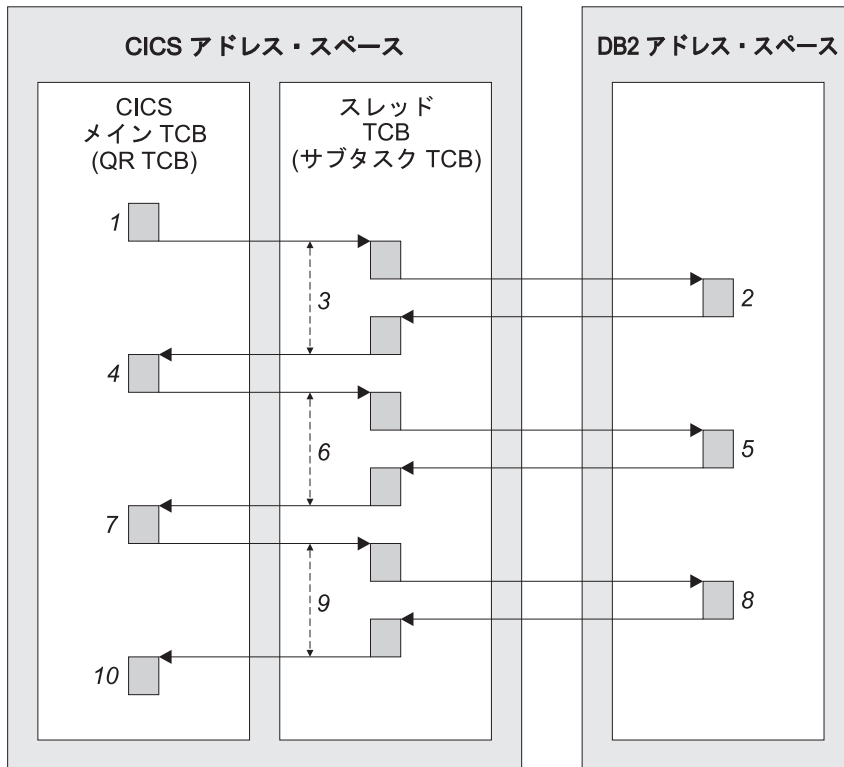
Db2 アカウンティング・レコードが生成されるのは、スレッドが終了したときか、サインオンが発生したときです。つまり、Db2 アカウンティング・レコード内で報告される期間は、スレッド開始またはユーザー・サインオン (以前に別のユーザーによって使用されたスレッドを再利用する場合) と、スレッド終了または別のサインオンとの間の時間であるということです。同じ DB2ENTRY にいくつかの異なるトランザクションが指定され、それらが同じ CICS トランザクション ID および同じ Db2 許可 ID を使用してスレッドにサインオンする場合には、このスレッドのための Db2 アカウンティング・レコードは、複数のトランザクション用のデータを含むことができます。また、トランザクションがその Db2 スレッドを同期点で解放し、スレッドが終了するか、別のトランザクションによって再使用される場合は、元のトランザクションは別のスレッドを使用する必要があります。したがって、単一トランザクションをその実行期間中に、複数の Db2 スレッドに関連付けることができます。つまり、Db2 がそれぞれのスレッドごとにアカウンティング・レコードを生成していく際に、複数のアカウンティング・レコードがトランザクションに対して生成されることがあります。各 Db2 アカウンティング・レコードに組み込まれるトランザクションを判別する方法の詳細については、167 ページの『Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの間の関係の制御』を参照してください。

プロセッサ時間は、トランザクション内の許可 ID ごとに、正確に取得することができます。そのためには、この許可 ID およびトランザクションについて、すべてのアカウンティング・レコードを集約します。ただし、アカウンティング・レコードに関連付けられた経過時間は、このケースではわずかな値にしかならないと考えられます。なぜなら、経過時間の合計には、1 つのコミット点から次の UOW 内の最初の SQL 呼び出しまでの時間は含まれないからです。スレッドの経過時間はスレッドに関連付けられますが、トランザクションには関連付けられません。

175 ページの図 36 および 177 ページの図 37 は、CICS および Db2 によって報告されるプロセッサ時間の各期間と、それがどこで発生するかを示しています。処理の場所は、Db2 にアクセスしているアプリケーションがスレッド・セーフかどうかによって異なります。その理由は、CICS が Db2 に接続されており、オー

プン・トランザクション環境を活用している場合、CICS Db2 接続機能は、CICS Db2 サブタスク TCB ではなく、CICS 管理のオープン TCB を使用するからです。

図 36 は、CICS が Db2 に接続されており、Db2 にアクセスしているアプリケーションが非スレッド・セーフの場合の状態を示しています。



この図は、CICS が Db2 に接続されており、Db2 にアクセスしているアプリケーションが非スレッド・セーフの場合の環境を示しています。

Time 3 + 6 + 9 が、CLASS 1 プロセッサ時間として報告される
(スレッド TCB の下で費やされた時間)
Time 2 + 5 + 8 が、CLASS 2 プロセッサ時間として報告される
(DB2 で費やされた時間)
Time 1 + 4 + 7 + 10 が、CICS プロセッサ時間として報告される
(CICS メイン TCB で費やされた時間)

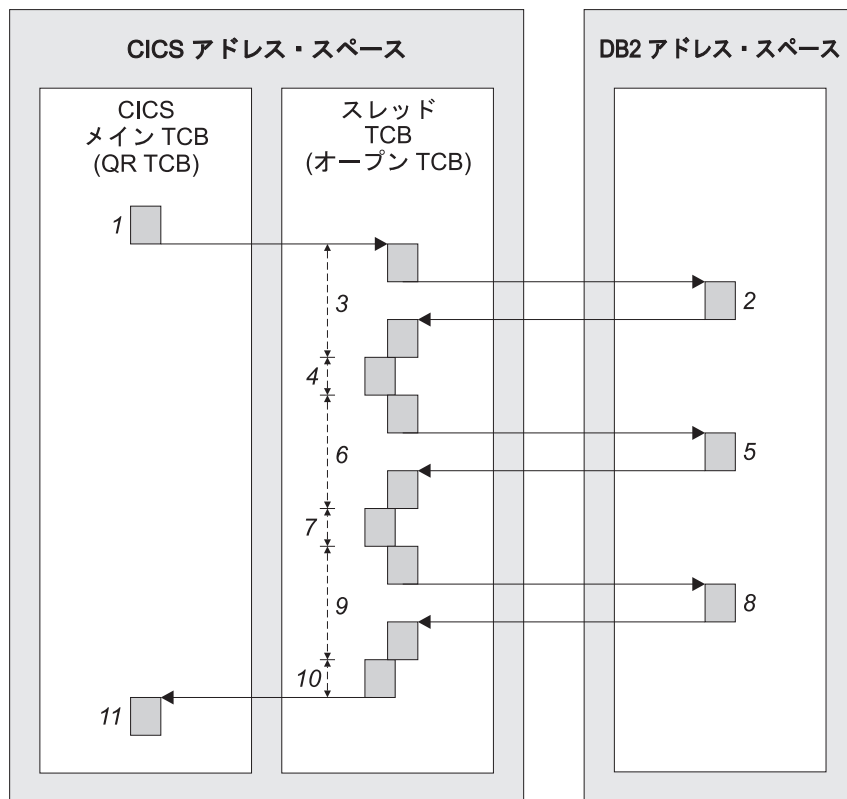
図 36. Db2 と非スレッド・セーフのアプリケーションを伴う CICS: 記録されるプロセッサ時間

CICS が Db2 に接続されているものの、Db2 にアクセスするアプリケーションが非スレッド・セーフの場合は、CICS はオープン TCB を使用してスレッドを実行して Db2 に入れますが、それはサブタスク TCB を使用するのと同じ方法でオープン TCB を使用します。スレッドを実行して Db2 に入れるために CICS が使用する TCB を、スレッド TCB といいます。

図に示されているプロセッサ時間の期間は次のとおりです。

- Time 1: アプリケーションが CICS メイン TCB で開始します。Time 1 の終わりに、アプリケーションは EXEC SQL 要求を出します。
- Time 2: Db2 がアプリケーションの要求を果たします。このプロセッサ時間は、Db2 アドレス・スペースで費やされます。Time 2 の終わりに、Db2 はその応答を CICS Db2 接続機能に渡します。
- Time 3: CICS Db2 接続機能がスレッド TCB 上で処理を遂行します。これによって、アプリケーションが Db2 にアクセスするために必要な処理と、Db2 応答をアプリケーションに戻すために必要な処理の両方が網羅されます。またそれには、Db2 が要求と応答を果たすために要したプロセッサ時間も含まれます。したがって Time 2 は Time 3 の中にネストされます。Time 3 の終わりには、Db2 からの応答がアプリケーションに渡され、Db2 へのアクセスが完了します。
- Time 4: アプリケーションは、当面の Db2 との作業を完了しました。アプリケーションが再び Db2 へのアクセスを必要とするようになるまでは、アプリケーション・コードは CICS メイン TCB 上で実行します。Time 4 の終わりに、アプリケーションは 2 番目の EXEC SQL 要求を出します。
- Time 5: Time 2 の場合と同様に、Db2 がアプリケーションからの 2 番目の要求を果たします。
- Time 6: CICS Db2 接続機能が、この 2 番目の要求に関連したスレッド TCB 上で処理を遂行するか、または Db2 からの応答を待ちます。
- Time 7: アプリケーション・コードが再び CICS メイン TCB 上で実行します。それは 3 番目の EXEC SQL 要求を出します。
- Time 8: Time 2 の場合と同様に、Db2 がアプリケーションからの 3 番目の要求を果たします。
- Time 9: CICS Db2 接続機能が、この 3 番目の要求に関連したスレッド TCB 上で処理を遂行するか、または Db2 からの応答を待ちます。
- Time 10: これで、アプリケーションはすべての EXEC SQL 要求を行いました。アプリケーション・コードは引き続き、終了まで CICS メイン TCB 上で実行します。

177 ページの図 37 は、CICS が Db2 に接続されていて、Db2 にアクセスしているアプリケーションがスレッド・セーフである場合の状態を示しています。



この図は、CICS が Db2 に接続されており、Db2 にアクセスしているアプリケーションがスレッド・セーフである場合の環境を示しています。

Time 3 + 4 + 6 + 7 + 9 + 10 が、CLASS 1 プロセッサ時間として報告される
(オープン TCB の下で Db2 へのアクセスの処理とアプリケーション・コードの
実行に費やされた時間)

Time 2 + 5 + 8 が、CLASS 2 プロセッサ時間として報告される
(Db2 で費やされた時間)

Time 1 + 11 が、CICS プロセッサ時間として報告される
(CICS メイン TCB で費やされた時間)

図 37. Db2 とスレッド・セーフのアプリケーションを伴う CICS : 記録されるプロセッサ時間

ここでは、CICS はオープン TCB を使用してスレッドを実行し、Db2 に入れます。アプリケーションがスレッド・セーフなので、アプリケーション・コードもオープン TCB 上で実行できます。図に示されているプロセッサ時間の期間は次のとおりです。

- Time 1: アプリケーションが CICS メイン TCB で開始します。Time 1 の終わりに、アプリケーションは EXEC SQL 要求を出します。
- Time 2: Db2 がアプリケーションの要求を果たします。このプロセッサ時間は、Db2 アドレス・スペースで費やされます。Time 2 の終わりに、Db2 はその応答を CICS Db2 接続機能に渡します。
- Time 3: CICS Db2 接続機能がスレッド TCB 上で処理を遂行します。これによって、アプリケーションが Db2 にアクセスするために必要な処理と、Db2 からの応答をアプリケーションに返すために必要な処理の両方が網羅されます。ま

たそれには、Db2 が要求と応答を果たすために取られるプロセッサ時間が含まれます。したがって Time 2 は Time 3 の中にネストされます。Time 3 の終わりには、Db2 応答がアプリケーションに渡され、Db2 へのアクセスが完了します。

- Time 4: アプリケーションは、当面の Db2 との作業を完了しました。アプリケーションがスレッド・セーフなので、アプリケーション・コードは今度はスレッド TCB (オープン TCB) 上で実行します。Time 4 の終わりに、アプリケーションは 2 番目の EXEC SQL 要求を出します。
- Time 5: Time 2 の場合と同様に、Db2 がアプリケーションからの 2 番目の要求を果たします。
- Time 6: CICS Db2 接続機能が、この 2 番目の要求に関連したスレッド TCB 上で処理を遂行するか、または Db2 からの応答を待ちます。
- Time 7: アプリケーション・コードが再びスレッド TCB 上で実行します。それは 3 番目の EXEC SQL 要求を出します。
- Time 8: Time 2 の場合と同様に、Db2 がアプリケーションからの 3 番目の要求を果たします。
- Time 9: CICS Db2 接続機能が、この 3 番目の要求に関連したスレッド TCB 上で処理を遂行するか、または Db2 からの応答を待ちます。
- Time 10: これで、アプリケーションはすべての EXEC SQL 要求を行いました。アプリケーション・コードが再びスレッド TCB 上で実行します。
- Time 11: アプリケーションがその処理を完了しました。アプリケーションは終了し、処理は CICS メイン TCB に戻ります。

スレッド・セーフとして定義されているアプリケーション (つまり、アプリケーション・コードがスレッド・セーフであることを CICS に伝えたもの) は、依然としてスレッド・セーフではない CICS コマンドを発行することに注意してください。それらのコマンドは強制的に CICS メイン TCB に戻し、アプリケーションが次の EXEC SQL 要求を行うまで、アプリケーション・コードは引き続き、CICS メイン TCB 上で実行します。非スレッド・セーフの CICS コマンドに関連付けられたアクティビティー、および CICS メイン TCB で実行するアプリケーション・コードに関連付けられたアクティビティーは、CLASS 1 プロセッサ時間としてではなく、CICS プロセッサ時間として報告されます。いったん EXEC SQL 要求が出されると、アプリケーション・コードは再びオープン TCB 上で実行でき、そのときから CLASS 1 プロセッサ時間として報告されます。

アカウンティング **CLASS 1** プロセッサ時間

CLASS 1 の場合、TCB が接続された時点で、タスクのプロセッサ・タイマーが作成されます。Db2 へのスレッドが開始されると、タイマー値が保管されます。スレッドが終了された場合や、許可 ID が変更された場合は、タイマーが再検査されて、タイマーの開始値と終了値の両方が SMF タイプ 101 レコードに記録されます。

175 ページの図 36 では、CLASS 1 は時間 3、6、および 9 の合計です。177 ページの図 37 では、CLASS 1 は時間 3、4、6、7、9、および 10 の合計です。

CLASS 1 プロセッサ時間のアカウンティングに使用される SMF タイプ 101 レコード (Db2 アカウンティング・レコード) の各フィールドは次のとおりです。

- ・ 開始スレッド TCB 時間の QWACBJST
- ・ 終了スレッド TCB 時間の QWACEJST
- ・ 開始 ASCB SRB 時間の QWACBSRB
- ・ 終了 ASCB SRB 時間の QWACESRB

Db2 アカウンティング・レコードの内容についての説明は、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』に記載されています。SDSNSAMP に付属のメンバー DSNWMSGs のアカウンティング・レコードのフィールドについての説明も記載されています。

CLASS 1 記録がスレッドに対してアクティブになると、L8 オープン TCB で費やされた時間が記録されます。L8 TCB は CICS アクティビティーと Db2 アクティビティーの両方に使用されるため、これには、トレース呼び出しを含めた CICS-Db2 接続機能で費やされたプロセッサ時間が含まれます。さらに、オープン TCB でアプリケーション・コード (アプリケーションがスレッド・セーフの場合) とスレッド・セーフ CICS コマンドの実行に費やされたプロセッサ時間も含まれます。スレッドが再使用された場合、スレッド・ハウスキーピング・プロセッサ時間も CLASS 1 プロセッサ時間に含められます。前のリリースと同様、CLASS 1 の時間で収集されない、ある比率のスレッド作成処理とスレッド終了処理があります。以下の場合、CLASS 1 プロセッサ時間には、QR TCB でアプリケーション・コードの実行に費やした時間は含まれません。

- ・ アプリケーションがその最初の EXEC SQL 要求を発行して、オープン TCB に移動する前の初期期間 (図中の時間 1)。この初期アプリケーション・コードは、CICS メイン TCB で実行され、Db2 アカウンティングでは確認されません。
- ・ アプリケーションが、スレッド・セーフではない CICS コマンドを発行した場合。これにより、処理は強制的に CICS メイン TCB に戻ります。その後、アプリケーション・コードの実行は、CICS メイン TCB で続行され、Db2 アカウンティングでは確認されません。この状態は、アプリケーションが次の EXEC SQL 要求を発行するまで続きます。この時点で、処理はオープン TCB に戻ることができます。

Db2 シスプレックス照会並列処理 (並列照会) を使用している場合を除き、アカウンティング目的で Db2 CLASS 1 プロセッサ時間を使用する必要はありません。アプリケーションで消費されたプロセッサ時間の完全なアカウンティングに必要なのは、CICS SMF タイプ 110 レコードに記録されたプロセッサ時間だけです。このレコードには、アプリケーション・コードで費やされた時間、スレッドの作成および終了のコスト、そして Db2 CLASS 1 プロセッサ時間で対象とされている時間が記載されます。詳しくは、181 ページの『Db2 の CICS および Db2 プロセッサ時間の計算』を参照してください。

アカウンティング CLASS 2 プロセッサ時間

アカウンティング CLASS 2 の場合、Db2 への出入りのたびにタイマーが検査され、SMF タイプ 101 レコードに「DB2 内」での時間が記録されます。この場合、この時間はレコードに保管されている時間とは異なります。CLASS 2 プロセッサ時間のアカウンティングに使用される SMF タイプ 101 レコードのフィールドは、QWACAJST です。

175 ページの図 36 および 177 ページの図 37 では、CLASS 2 は時間 2、5、および 8 の合計です。

Db2 統計レコードの内容の説明については、Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』を参照してください。

経過時間 (定義された時点を範囲とする開始時刻と終了時刻) もまた、SMF タイプ 101 レコードで報告されます (QWACASC)。

CICS が Db2 に接続されている場合、IMS と TSO の間で大きな違いが生じるように、CLASS 1 と CLASS 2 のプロセッサ時間にも顕著な違いが生じる可能性があります。これは、CICS が Db2 に接続されている場合には、CLASS 1 プロセッサ時間には、トレース呼び出しを含めた CICS-Db2 接続機能で費やされたプロセッサ時間が含まれるだけでなく、さらにオープン TCB でスレッド・セーフなアプリケーション・コードとスレッド・セーフな CICS コマンドの実行に費やされたプロセッサ時間も含まれるためです。このプロセッサ時間はすべて CICS アドレス・スペース内で発生するため、アカウンティング CLASS 2 にはレポートされません。CLASS 2 プロセッサ時間自体は、オープン・トランザクション環境に影響されません。

アカウンティング CLASS 1 とは異なり、CLASS 2 経過時間およびプロセッサ時間は、「DB2 内」で費やされた経過時間とプロセッサ時間を正確に反映します。したがって、アカウンティング CLASS 2 情報は、SQL 実行のパフォーマンスをモニターする上で非常に役立ちます。ただし、クラス・アカウンティングには、かなりのプロセッサ・オーバーヘッドが伴います。

オーバーヘッドを削減するために、通常は CLASS 2 トレースを、特定の許可 ID の計画およびロケーションに関するデータのトレースに制限するように推奨されていました。プロセッサ・オーバーヘッドは 0 から 5% の範囲で、通常は 2% です。

Db2 以前の CICS および DB2 バージョン 5 プロセッサ時間の計算

CICS システムが DB2 バージョン 5 以前に接続されている場合、CICS パフォーマンス・クラス・レコードには、Db2 で消費されるプロセッサ時間は含まれません。単一トランザクションの合計プロセッサ時間を見積もるには、対応する CICS パフォーマンス・レコードと Db2 アカウンティング・レコード (SMF タイプ 101 レコード) からの情報を加算します。

このタスクについて

CICS パフォーマンス・レコードと Db2 アカウンティング・レコードのマッチングについて詳しくは、165 ページの『Db2 アカウンティング・レコードと CICS パフォーマンス・クラス・レコードとの関連付け』を参照してください。

CICS パフォーマンス・クラス・レコードの CPU フィールドを使用します。Db2 アカウンティング・レコードを使用して、Db2 スレッド・プロセッサ時間 (T1) を次のように計算することができます。

$$T1 = QWACEJST - QWACBJST$$

これにより、CLASS 1 TCB プロセッサ時間が計算されます。CICS CPU フィールドからのプロセッサ時間と T1 の計算値の合計が、CICS と Db2 で費やされたプロセッサ時間の式です。

注:

- CLASS 2 プロセッサ時間は CLASS 1 プロセッサ時間の一部であるため、合計に加算すべきではありません。
- CLASS 1 プロセッサ時間から CLASS 2 プロセッサ時間を減算すると、CICS Db2 接続機能の CPU 利用の近似値になります。これは、調整を行う際の参考になります。
- Db2 アドレス・スペースで使用され、Db2 統計レコードに記録されるプロセッサ時間は、いかなる特定スレッドとの関連性也没有ありません。これは、Db2 アカウンティング・レコードに記録される CPU 時間に比例して配分することができます。
- CICS アドレス・スペースのサブタスク TCB の下で使用されるプロセッサ時間を CICS パフォーマンス・レコードに簡単に配分することはできません。これには Db2 サブタスクのプロセッサ時間が含まれているからです。これは既に計算値 T1 に含まれています。これは、スレッドのサブタスク以外のサブタスクで使用されるプロセッサ時間は加算に含まれないことを意味します。
- スレッドを作成するためにスレッド TCB で使用されるプロセッサ時間のほとんどは、このスレッドに関連付けられている Db2 アカウンティング・レコードに含まれていません。このプロセッサ時間は、スレッドが作成される前に費やされるからです。
- CICS および Db2 のキャプチャー率を考慮に入れる必要があります。キャプチャー率とは、CPU 使用時間合計に対する、報告された CPU 時間の比率のことです (詳しくは、*z/OS Resource Measurement Facility™ (RMF) パフォーマンス管理ガイド (SC88-6666)* を参照してください)。

Db2 の CICS および Db2 プロセッサ時間の計算

CICS が Db2 に接続されていて、オープン・トランザクション環境を利用している場合、CICS Db2 接続機能は CICS Db2 サブタスク TCB ではなく、CICS 管理オープン TCB を使用します。

このタスクについて

CICS 管理オープン TCB を使用するということは、CICS モニター機能が、以前は Db2 アカウンティング・レコード (SMF タイプ 101 レコード) でしか報告されていなかったアクティビティを測定できることを意味します。例えば現在の CICS では、Db2 スレッドで消費されたプロセッサ時間と、Db2 で消費されたプロセッサ時間 (CLASS 1 および CLASS 2 の CPU 時間) の測定を行えるようになりました。CICS が L8 オープン TCB を使用している場合、CICS モニター機能によってこれらの TCB に対して報告される CPU 時間には、Db2 CLASS 1 のプロセッサ時間が含まれます。

オープン・トランザクション環境では、CICS L8 タスク・プロセッサ時間に Db2 スレッドの作成にかかるコストも含めることができます。

Db2 並列照会を使用しているのではない限り、単一トランザクションの合計プロセッサ時間を計算する際に、CICS レコード (SMF タイプ 110 レコード) からのプロセッサ時間と Db2 アカウンティング・レコード (SMF タイプ 101 レコード) からのプロセッサ時間を加算しないでください。Db2 プロセッサ時間を 2 回足してしまうことになるためです。単一トランザクションの合計プロセッサ時間は、CICS レコードの USRCPUT フィールド (DFHTASK グループの 008 パフォーマンス・クラス・データ・フィールド) に記録されます。このフィールドには、CICS ディスパッチャーが管理する TCB 上で実行されていたトランザクションの使用したすべてのプロセッサ時間が含まれます。CICS 管理の TCB には、QR、RO、CO、および L8 モードの TCB が含まれています。

Db2 並列照会を使用している場合、Db2 アカウンティング・レコードの一部のプロセッサ時間を追加する必要があります。Db2 シスプレックス照会並列処理では、Db2 が Db2 アドレス・スペース内の複数の TCB を使用して照会に対応する可能性があります。この目的で使用される TCB は、非 CICS TCB であるため、これらの TCB が消費するプロセッサ時間は、CICS SMF タイプ 110 レコードでは考慮されません。したがって、並列照会に使用された合計プロセッサ時間を取得するには、トランザクションに適用される Db2 SMF タイプ 101 レコードに記録された PAR.TASKS プロセッサ時間を、CICS SMF タイプ 110 レコードの USRCPUT フィールドに記録されたプロセッサ時間に加算する必要があります。

第 9 章 Db2 のトラブルシューティング

このセクションでは、CICS Db2 の問題判別について説明します。

このトピックには、診断、変更、または調整に関する情報が含まれています。

スレッド TCB (タスク制御ブロック)

CICS Db2 環境では、Db2 への各スレッドは、スレッド・タスク制御ブロック (TCB) 下で実行されます。このセクションでは、問題判別を支援するために、スレッド TCB に関する詳しいテクニカル情報を提供します。

スレッド TCB の概要については、概要: スレッドの動作方法を参照してください。

スレッド TCB は、オープン L8 モード TCB です。オープン TCB は、メイン CICS TCB (QR TCB) の「ドーター」です。CICS Db2 タスク関連ユーザー出口自体がオープン TCB 上で実行され、さらにそれを使用して、スレッドが実行されます。タスク関連ユーザー出口は、スレッドを獲得する必要がある場合、CICS Db2 接続モジュール DFHD2D2 を使用して Db2 を呼び出します。異なる TCB 上で実行されている別のモジュール DFHD2CO が、Db2 への識別や Db2 からの CICS の切断など、全体的な CICS Db2 接続のさまざまな側面に対応します。

常時 Db2 へのスレッドを実行できるオープン TCB の最大数は、DB2CONN の TCBLIMIT パラメーターを使用して制御されます。スレッドを実行しているオープン TCB は、スレッドが終了する時点では終了しません。オープン TCB が終了する可能性があるのは、以下の場合です。

- CICS トランザクションが CICS から強制ページされ、スレッドが Db2 内でまだアクティブである。この場合、要求を Db2 からフラッシュする手段として、TCB が終了されます。Db2 内の現在の UOW はバックアウトされます。
- コーディネーターとの接続が失われたため、CICS が UOW の結果に関して未確定である。TCB を終了すると、Db2 はスレッドを解放しますが、UOW は未確定として維持し、そのロックを維持します。後に CICS がコーディネーターとの接続を再確立したときに、UOW は再同期によって完了します。
- CICS ディスパッチャー (CICS Db2 接続機能で使用されていない場合に、オープン TCB が返される場所) が、30 分後に、未使用のオープン TCB をクリーンアップする。

CICS Db2 の待機タイプ

CICS Db2 タスク関連ユーザー出口 DFHD2EX1 は、さまざまな目的で **WAIT_MVS** 呼び出しを発行します。このような呼び出しは、リソース名とリソース・タイプの値によって区別されます。リソース名とリソース・タイプの値は、CICS システム・ダンプのディスパッチャー・セクションに表示されます。また、**CEMT INQUIRE TASK** パネルにもそれぞれ Hty と Hva の値で表示されます。

表 15. CSUB トレース・テーブル・エントリーのレイアウト (続き)

バイト	コンテンツ	情報
バイト 4 から 7	トレース要求	発行された 4 文字表記の Db2 要求。可能な値は以下のとおりです。 ABRT - 異常終了要求 API - SQL または IFI 要求 ASSO - 関連付け要求 COMM - コミット要求 CTHD - スレッド作成要求 DISS - 切り離し要求 ERRH - エラー・ハンドラー要求 IDEN - ID 要求 PREP - 準備要求 PSGN - 部分的サインオン要求 SIGN - 完全サインオン要求 SYNC - 単一フェーズ要求 TERM - スレッド終了要求 TIDN - 端末 ID 要求 TSGN - サインオン終了要求 *REC - リカバリー・ルーチン開始
バイト 8 から 9	予約済み	
バイト 10 から 11	frb または ifc 戻りコード	2 バイトの戻りコード
バイト 12 から 15	frb または ifc 理由コード	4 バイトの理由コード

CSUB 制御ブロックは、フォーマット設定されて CICS システム・ダンプに出力されます。また、RI (RMI) レベル 2 トレースがアクティブな場合、CICS タスク関連ユーザー出口 DFHD2EX1 からのトレース出力、および DFHD2EX1 からのすべての例外トレースにも出力されます。

196 ページの図 38 は、CICS が DB2 バージョン 6 以降に接続されている場合の同じ状況を示しています。この例では、ID、サインオン、およびスレッド作成が Db2 に対して発行されていることが分かります。API 要求があり、その後に同期点および切り離し (Db2 接続制御ブロックを L8 TCB から切り離す) が出されています。次に、トランザクションは、別の API 要求を行ない、別の作業単位を開始して、Db2 接続制御ブロックが L8 TCB と再関連付け (ASSO) されます。前の作業単位のアカウンティング・レコードを作成するために、部分的サインオンが発生します。ここで、Db2 に対して API 要求が出されています。

CICS Db2 の実行診断機能 (EDF)

CICS Db2 タスク関連ユーザー出口 DFHD2EX1 は、FORMATEDF キーワードで使用可能にされ、トランザクションが EDF 下で実行されているときに、SQL API 要求の画面をフォーマット設定するために CICS によって呼び出されます。

EDF モードでは、CICS Db2 接続機能は、以下のことを行います。

- EXEC SQL コマンドごとに停止し、SQL ステートメントを暗号解読して、パネル上のファイルに表示します。
- SQL 呼び出しの処理の前と後に結果を表示します。
- 以下の情報を表示します。
 - SQL ステートメントのタイプ。
 - 入出力変数。
 - SQLCA の内容。
 - 1 次および 2 次の許可 ID。(これは、SQLCODE -922 の診断に役立ちます。)

1 つの EDF パネルに最大 55 個の変数が表示されます。これは、およそ 10 画面です。各 EDF SQL セッションに 12KB の CICS 一時ストレージが必要です。このストレージは、EDF から出ると解放されます。

SQL ステートメントの EDF 画面を図 40 と 199 ページの図 41 に示します。

```
TRANSACTION: XC05 PROGRAM: TESTC05 TASK:0000097 APPLID: CICS41 DISPLAY:00
STATUS: ABOUT TO EXECUTE COMMAND
```

```
EXEC SQL OPEN
DBRM=TESTC05, STMT=00221, SECT=00001
```

```
OFFSET: X'001692' LINE: UNKNOWN EIBFN=X'0E0E'
```

```
ENTER:
PF1 : UNDEFINED      PF2 : UNDEFINED      PF3 : UNDEFINED
PF4 :                 PF5 :                 PF6 :
PF7 :                 PF8 :                 PF9 :
PF10:                 PF11: UNDEFINED      PF12:
```

図 40. 「処理前」の SQL EXEC パネルの EDF の例

を使用している場合) またはトランザクションで使用される DB2ENTRY によって決まります。犠牲となるパートナーがロールバックされた後、他方のパートナーは処理を続行します。

デッドロック状態を解決するには、いくつかのアクティビティを実行する必要があります。デッドロックの解決とは、システム内のどこかで変更を適用し、デッドロックの可能性を減らすことを意味します。

多くの場合、以下のステップがデッドロック状態の解決に必要になります。

1. デッドロックを検出します。
2. 関係するリソースを見つけます。
3. 関係する SQL ステートメントを見つけます。
4. 使用されたアクセス・パスを見つけます。
5. デッドロックの発生理由を判別します。
6. 回避するために変更を加えます。

関連概念

183 ページの『第 9 章 Db2 のトラブルシューティング』
このセクションでは、CICS Db2 の問題判別について説明します。

183 ページの『スレッド TCB (タスク制御ブロック)』
CICS Db2 環境では、Db2 への各スレッドは、スレッド・タスク制御ブロック (TCB) 下で実行されます。このセクションでは、問題判別を支援するために、スレッド TCB に関する詳しいテクニカル情報を提供します。

183 ページの『CICS Db2 の待機タイプ』
CICS Db2 タスク関連ユーザー出口 DFHD2EX1 は、さまざまな目的で **WAIT_MVS** 呼び出しを発行します。このような呼び出しは、リソース名とリソース・タイプの値によって区別されます。リソース名とリソース・タイプの値は、CICS システム・ダンプのディスパッチャー・セクションに表示されます。また、**CEMT INQUIRE TASK** パネルにもそれぞれ Hty と Hva の値で表示されます。

187 ページの『CICS Db2 のメッセージ』
CICS Db2 接続機能によって発行されるメッセージは、DFHDB 接頭部を使用します (この接頭部は、CICS DBCTL メッセージにも使用されます)。メッセージ番号は 2000 から 2999 までの範囲で、CICS Db2 用に予約済みです。したがって、CICS Db2 接続機能メッセージの形式は、DFHDB2xxx となります。

187 ページの『CICS Db2 のトレース』
CICS Db2 接続機能は、3100 から 33FF までの範囲の AP ドメイン・トレース・ポイントを使用します。

196 ページの『CICS Db2 のダンプ』
CICS Db2 接続機能からの制御ブロックは、CICS IPCS verbexit の Db2 キーワードの制御下で、CICS システム・ダンプに定様式で出力されます。

197 ページの『Db2 スレッド ID』
CICS トランザクションの代わりに Db2 内で実行されているスレッドは、CICS Db2 接続機能によって設定される相関 ID によって識別されます。

197 ページの『CICS Db2 のトランザクション異常終了コード』
CICS Db2 接続機能では、複数の異常終了コードが使用され、それぞれ特定のエラーに固有です。

198 ページの『CICS Db2 の実行診断機能 (EDF)』

CICS Db2 タスク関連ユーザー出口 DFHD2EX1 は、FORMATEDF キーワードで使用可能にされ、トランザクションが EDF 下で実行されているときに、SQL API 要求の画面をフォーマット設定するために CICS によって呼び出されます。

2 つのタイプのデッドロック

Db2 内のデッドロックは、2 つのトランザクションの双方が相手のトランザクションが必要とするロックを保持している場合に発生する可能性があります。

Db2 環境では、以下の場合に、2 つのタイプのデッドロックが発生する可能性があります。

- 2 つのリソースが関与している。各トランザクションが 1 つのリソースをロックしており、非互換モードで他方のリソースを必要としています。リソースは、通常、索引ページおよびデータ・ページです。これは、従来のデッドロック状態です。
- 1 つだけのリソースが関与している。更新 (U) ロックの主な目的は、ロック・プロモーション に起因するデッドロックが発生する状況の数を減らすことです。U ロックは、こうした状況の大部分を解決しましたが、依然として特定の状況では、1 つだけのリソースが関与するデッドロックが発生する可能性があります。これは、リソースが複数の方法でロックされる可能性があるためです。

この代表的な例は、トランザクションが ORDER BY オプションを指定してカーソルを開き、ソートを避けるために索引を使用する場合です。ページ内の行を取り出すときに、Db2 はそのページに共用 (S) ロックを取得します。その後、最後に取り出された行に対して、トランザクションがカーソルを使用せずに更新を発行した場合、S ロックは排他的 (X) ロックにプロモートされます。

このようなトランザクションの 2 つが同時に実行され、両方が X ロックを取得する前に同じページで S ロックを取得した場合、デッドロックが発生します。

デッドロックの検出

デッドロックに関する情報は、MVS ログ内の Db2 パフォーマンス・トレースで入手できます。

このタスクについて

Db2 パフォーマンス・トレースをアクティブにせずに実行されている通常の実稼働環境では、デッドロックに関する情報を入手する最も簡単な方法は、MVS ログをスキャンして、202 ページの図 42 に示されているようなメッセージを見つけることです。

```
DSNT375I PLAN p1 WITH CORRELATION ID id1
AND CONNECTION ID id2 IS DEADLOCKED with
PLAN p2 WITH CORRELATION ID id3
AND CONNECTION ID id4.

DSNT501I DSNILMCL RESOURCE UNAVAILABLE
CORRELATION-ID=id1,CONNECTION-ID=id2
REASON=r-code
TYPE name
NAME name
```

図 42. デッドロック・メッセージ

これらのメッセージから、デッドロックの双方のパートナーを特定できます。パートナーは、計画名と相関 ID の両方で示されています。

また、2 番目のメッセージは、犠牲になったパートナーが取得できなかったリソースを識別しています。他方のリソースは (同じものであるかどうかに関係なく)、メッセージには表示されません。

関与するリソースの検出

デッドロックに関与している他方のリソースを見つけるには、Db2 パフォーマンス・トレースをアクティブにして、デッドロックを再作成することが必要になる場合があります。

デッドロックを解決するための理由が、デッドロックの数が多すぎることでであると仮定します。通常は、トレースが開始された後にデッドロックを再作成するのは、マイナーな問題です。

Db2 パフォーマンス・トレースの対象を、MVS ログ・メッセージに示された 2 つの計画に制限します。トレースの対象を、関与する 2 つの CICS トランザクション ID (許可 ID) に制限するのも合理的な場合があります。パフォーマンス・トレースには、汎用ロック・イベントの class(06) と、SQL イベントの class(03) を含めます。Db2 パフォーマンス・モニター (DB2PM) は、トレース出力をフォーマット設定するのに便利なツールです。DB2PM ロック競合レポートとロック中断レポートは、デッドロックに関与するリソースを判別するのに役立つことがあります。

DB2PM レポートの出力が大き過ぎる場合は、トレースの出力を分析するためのユーザー・プログラムを作成できます。目標は、デッドロックに関与するリソースと、関与するすべての SQL ステートメントを見つけることです。

関与する SQL ステートメントの検出

デッドロックには多数の SQL ステートメントに関与している可能性があります。多くの場合、デッドロックを解決するには、すべての SQL ステートメントを見つける必要があります。

このタスクについて

ロック・トレースから関与するリソースが特定された場合、SQL トレース・レポート内の関与する SQL ステートメントを、両方のトレースのタイム・スタンプを比較することで検出できます。

使用されたアクセス・パスの検出

デッドロックに関与する SQL ステートメントが使用したアクセス・パスを検出するには、対応する計画を対象にして Db2 の EXPLAIN オプションを使用します。

デッドロックが発生した理由の判別

デッドロックに関与する SQL ステートメントとリソースを識別し、アクセス・パスを見つければ、通常はデッドロックが発生した理由が示されます。デッドロックが発生した理由が分かると、解決策を開発するのに役立ちます。ただし、このプロセスには時間がかかることがあります。

デッドロックの解決

デッドロックを解決するための適切なアクションを選択できるように、デッドロックが発生した理由を理解することが必要です。

このタスクについて

一般に、デッドロックが発生する原因は、2 つ以上のトランザクションの双方が、競合するモードで、同時に反対の順序で同じリソースを必要することにあります。デッドロックを防止するために取るアクションは、こうした特性に対処する必要があります。

表 16 は、予防的処置とそれに対応する主な影響を示しています。

表 16. デッドロックの回避

アクション	リソースの拡張	ロック順序の変更	競合の削減	ロック・モードの変更
索引のフリー・スペースの増加	X			
索引のサブページ・サイズの増加	X			
TS フリー・スペースの増加	X			
クラスター化索引の変更	X	X		
表スペースの再編成	X	X	X	
索引の追加		X	X (1)	
索引の除去		X		
トランザクションの直列化			X	
追加 COMMIT の使用			X	
応答時間の最小化			X	
分離レベルの変更 (2)			X	X
アプリケーションの再設計	X	X	X	X
データベースの再設計	X	X	X	X
注:				
1. アクセス・パスの変更に起因する。				
2. 通常は、カーソル固定が、反復可能読み取りよりも適切です。				

適切なアクションを選択するには、まずデッドロックが発生した理由を理解する必要があります。それにより、選択するアクションを評価できます。こうしたアクションは、いくつかの影響を与える可能性があります。以下のものが考えられます。

- デッドロック問題を解決する
- 他のトランザクションのアクセス・パスを強制的に変更して、新たなデッドロックを引き起こす
- システム内に新たなデッドロックを引き起こす

そのため、影響を受けるトランザクション (例えば、Db2 内の EXPLAIN 機能) が使用するアクセス・パスを注意深くモニターすることが重要です。多くの場合、デッドロックの解決は反復プロセスです。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。この資料の他の言語版を IBM から入手できる場合があります。ただし、これを入手するには、本製品または当該言語版製品を所有している必要がある場合があります。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。IBM 製品、プログラムまたはサービスに代えて、IBM の知的所有権を侵害することのない機能的に同等のプログラムまたは製品を使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510

東京都中央区日本橋箱崎町19番21号

日本アイ・ビー・エム株式会社

法務・知的財産

知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様自身の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119 Armonk,
NY 10504-1785
United States of America*

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名前はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態を提供されるものであり、いかなる保証も提供されません。IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

プログラミング・インターフェース情報

CICS には、プログラミング・インターフェースと見なすことのできる資料と、プログラミング・インターフェースと見なすことのできない資料があります。

オンライン製品資料の以下のセクションには、CICS Transaction Server for z/OS, バージョン 5 リリース 5 のサービスを取得するプログラムをお客様が作成するためのプログラミング・インターフェースが含まれています。

- アプリケーションの開発
- Developing system programs
- 保護の概要
- 外部インターフェースに向けた開発
- リファレンス: アプリケーション開発h
- リファレンス: システム・プログラミング
- リファレンス: 接続

オンライン製品資料の以下のセクションには、CICS Transaction Server for z/OS, バージョン 5 リリース 5 のプログラミング・インターフェースとして意図されていない (プログラミング・インターフェースと誤解される可能性のある) 情報が含まれています。

- Troubleshooting and support
- リファレンス: 診断

PDF 形式のマニュアルで CICS 資料にアクセスする場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 5 のサービスを取得するプログラムをお客様が作成するためのプログラミング・インターフェースが以下のマニュアルに含まれています。

- アプリケーション・プログラミング・ガイドおよびアプリケーション・プログラミング・リファレンス
- Business Transaction Services
- Customization Guide
- C++ OO Class Libraries
- Debugging Tools Interfaces Reference
- Distributed Transaction Programming Guide
- External Interfaces Guide
- Front End Programming Interface Guide
- IMS Database Control Guide
- インストール・ガイド
- セキュリティー・ガイド
- Supplied Transactions
- CICSplex[®] SM Managing Workloads
- CICSplex SM Managing Resource Usage
- CICSplex SM アプリケーション・プログラミング・ガイドおよび CICSplex SM アプリケーション・プログラミング・リファレンス
- Java Applications in CICS

PDF 形式のマニュアルで CICS 資料にアクセスする場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 5 のプログラミング・インターフェース

として意図されていない (プログラミング・インターフェースと誤解される可能性のある) 情報が以下のマニュアルに含まれています。

- Data Areas
- Diagnosis Reference
- Problem Determination Guide
- CICSplex SM Problem Determination Guide

商標

IBM、IBM ロゴおよび [ibm.com](http://www.ibm.com/legal/copytrade.shtml)[®] は、世界の多くの国で登録された International Business Machines Corporation の商標または登録商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、<http://www.ibm.com/legal/copytrade.shtml> をご覧ください。

Adobe、Adobe ロゴ、PostScript、PostScript ロゴは、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Intel Centrino、Intel Centrino ロゴ、Celeron、Intel Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

製品資料に関するご使用条件

これらの資料は、以下のご使用条件に同意していただける場合に限りご使用いただけます。

適用範囲

IBM Web サイトの「ご利用条件」に加えて、以下のご使用条件が適用されます。

個人使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商用使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することがで

きます。ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

権利 ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

IBM オンラインでのプライバシー・ステートメント

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品（「ソフトウェア・オファリング」）では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項をご確認ください。

CICSplex SM Web ユーザー・インターフェース（メイン・インターフェース）の場合： このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、認証、お客様の利便性の向上、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名、およびその他の個人情報を、セッションごとの Cookie および持続的な Cookie を使用して収集する場合があります。これらの Cookie を無効にすることはできません。

CICSplex SM Web ユーザー・インターフェース（データ・インターフェース）の場合： このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、認証、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名またはその他の個人情報を、セッションごとの Cookie を使用して収集する場合があります。これらの Cookie を無効にすることはできません。

CICSplex SM Web ユーザー・インターフェース（「Hello World」ページ）の場合： このソフトウェア・オファリングは、展開される構成に応じて、個人情報を収集しないセッションごとの Cookie を使用する場合があります。これらの Cookie を無効にすることはできません。

CICS Explorer の場合:

このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、お客様の利便性の向上、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名、およびその他の個人情報を、セッションごとの設定および持続的な設定を使用して収集する場合があります。これらの設定を無効にすることはできませんが、ユーザー・パスワードの暗号化形式でのディスクへの保管は、サインオン中にチェック・ボックスにチェック・マークを付けることによるユーザーの明示的な操作によってのみ有効化することができます。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、『IBM オンラインでのプライバシー・ステートメント』(<http://www.ibm.com/privacy/details/jp/ja/>) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』および『IBM Software Products and Software-as-a-Service Privacy Statement』(<http://www.ibm.com/software/info/product-privacy>) を参照してください。

RETURN IMMEDIATE 119
RMI (リソース・マネージャー・インター
フェース) 2
RRCDTE サンプル・ジョブ 63

S

SASS (単一アドレス・スペース) 72
SMF 101 レコード
フィールド 178
SMF (システム管理機能) 39, 156
SQL
修飾または非修飾 114
静的 83
動的 83
SQL 言語 112
SQL 処理、主なアクティビティ 19
SQL ステートメントの EDF パネル。
198
SQL 戻りコード
-501 119
-818 139
-911 199
-913 199
SQL 要求 2
SQLCA フォーマット設定ルーチン 137
SQLJ 123
同期点 132
CICS 異常終了 133
STANDBYMODE コマンド 121
STOP 接続機能コマンド 52
STRT 接続機能コマンド 54
synconreturn 132

T

TCB 接続 25, 27
TCB 接続、スレッド 23
TCBLIMIT 18
THRDA 102
THREADLIMIT 26, 102
THREADWAIT 25
Tivoli Decision Support for OS/390 147
TYPE=ENTRY グルーピング・トランザク
ション 28

U

UOW (作業単位) 34

V

VALIDATE 140
VCICSCMD 一般リソース・クラス 65
VERSION キーワード 143

X

XCICSDB2 一般リソース・クラス 63
XCICSDB2 メンバー・クラス 63
XCTL、CICS 制御権移動 90

Z

ZCICSDB2 グループ化クラス 63

