

IBM® DB2 Universal Database™



アプリケーション開発ガイド アプリケーションの構築および実行

バージョン 8.2

IBM® DB2 Universal Database™



アプリケーション開発ガイド アプリケーションの構築および実行

バージョン 8.2

ご注意！

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

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

<http://www.ibm.com/jp/manuals/main/mail.html>

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

<http://www.ibm.com/jp/manuals/> の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

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：	SC09-4825-01 IBM® DB2 Universal Database™ Application Development Guide: Building and Running Applications Version 8.2
発 行：	日本アイ・ビー・エム株式会社
担 当：	ナショナル・ランゲージ・サポート

第1刷 2004.8

この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財)日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

DB2 アプリケーション開発へようこそ . . .	ix
DB2 Developer's Edition 製品	ix
本書について	xi

第 1 部 アプリケーション開発環境 . . 1

第 1 章 DB2 環境サポート 3

DB2 Application Development Client	3
データベース・マネージャー・インスタンス	5
DB2 でサポートされるサーバー	8
DB2 でサポートされる開発ソフトウェア	9
AIX でサポートされる開発ソフトウェア	10
HP-UX でサポートされる開発ソフトウェア	12
Linux でサポートされる開発ソフトウェア	14
Solaris でサポートされる開発ソフトウェア	20
Windows でサポートされる開発ソフトウェア	23

第 2 章 セットアップ 27

一般的なセットアップ情報	27
アプリケーション開発環境のセットアップ	27
DB2 ルーチンの共用ライブラリーの再構築	29
データベース・マネージャー構成ファイルの更新	30
Java 環境のセットアップ	31
DB2 WebSphere MQ 関数のセットアップ	33
UNIX	36
UNIX アプリケーション開発環境のセットアップ	36
UNIX 環境変数の設定	37
UNIX Java 環境のセットアップ	38
AIX Java 環境のセットアップ	40
HP-UX Java 環境のセットアップ	41
Linux Java 環境のセットアップ	43
Solaris Java 環境のセットアップ	44
Windows	45
Windows アプリケーション開発環境のセットアップ	45
Windows Java 環境のセットアップ	49
サンプル・データベース	52
サンプル・データベースのセットアップ	52
サンプル・データベースの作成	52
ホスト・サーバーまたは AS/400 および iSeries	
サーバーでのサンプル・データベースの作成	54
サンプル・データベースのカタログ	55
サンプル・データベース・ユーティリティのバ	
インディング	55
アプリケーションの移行	57
アプリケーションの DB2 バージョン 8 への移行	57
Java アプリケーション、ルーチン、およびアプレ	
ットの移行	59
32 ビット環境から 64 ビット環境へのアプリケー	
ションの移行	60

アプリケーション移植性の確保	63
2 つのバージョンの DB2 でのアプリケーション	
の実行	64
次に行うこと	68

第 3 章 サンプル・プログラムおよび関連

ファイル 69

サンプル・ファイル	69
言語およびアプリケーション・インターフェース別	
のサンプル・プログラム	75
C サンプル	75
C++ サンプル	79
C# のサンプル	82
CLI のサンプル	83
コマンド行プロセッサ (CLP) のサンプル	85
COBOL のサンプル	86
動的再構成のサンプル	90
JDBC のサンプル	91
SQLJ のサンプル	94
Java WebSphere のサンプル	96
Java プラグインのサンプル	96
ログ管理ユーザー出口サンプル	97
オブジェクトのリンクと埋め込み (OLE) のサン	
プル	99
オブジェクトのリンクと埋め込みデータベース	
(OLE DB) 表関数のサンプル	100
Perl のサンプル	101
PHP のサンプル	102
REXX のサンプル	102
セキュリティ・プラグインのサンプル	105
SQL プロシージャのサンプル	105
Visual Basic のサンプル	108
Visual Basic .NET のサンプル	109
Visual C++ のサンプル	111
Windows Management Instrumentation のサンプル	112
ビルド・ファイル、makefile、およびエラー・チェ	
ック・ユーティリティ	112
ビルド・ファイル	112
makefile	116
エラー・チェック・ユーティリティ	119

第 2 部 プラットフォーム非依存ア

プリケーションの構築および実行 . . 123

第 4 章 Java 125

Java サンプル・プログラム	125
Java アプレットに関する考慮事項	127
JDBC	129
JDBC アプレットの構築	129
JDBC アプリケーションの構築	130

JDBC ルーチンの構築	131
SQLJ	134
SQLJ プログラムの構築	134
SQLJ アプレットの構築	135
SQLJ アプリケーションの構築	137
SQLJ アプリケーションおよびアプレットの UNIX ビルド・スクリプト	138
UNIX の SQLJ アプリケーションおよびアプレ ット・オプション	139
SQLJ アプリケーションおよびアプレットの Windows バッチ・ファイル	140
Windows の SQLJ アプリケーションおよびアプ レット・オプション	142
SQLJ ルーチンの構築	142
SQLJ ルーチンの UNIX ビルド・スクリプト	144
UNIX の SQLJ ルーチン・オプション	145
SQLJ ルーチンの Windows バッチ・ファイル	146
Windows の SQLJ ルーチン・オプション	148

| 第 5 章 コマンド行プロセッサ . . . 149

コマンド行プロセッサ (CLP) スクリプトの実行	149
コマンド行プロセッサ (CLP) からのプロシ ジャーの呼び出し	150

第 6 章 SQL プロシージャ . . . 153

SQL プロシージャの作成	153
クライアント・アプリケーションによる SQL プロ シージャの呼び出し	154
SQL プロシージャのプリコンパイル・オプショ ンと BIND オプションのカスタマイズ	155
DB2 8.2 より前に作成された SQL プロシージャ のバックアップとリストア	157
SQL プロシージャの再バインド	158

| 第 7 章 Perl . . . 161

Perl アプリケーションの構築	161
------------------	-----

| 第 8 章 PHP . . . 165

PHP アプリケーションの構築	165
-----------------	-----

第 3 部 プラットフォーム固有アプ 리케이션の構築および実行 . . . 169

| 第 9 章 UNIX . . . 171

UNIX C アプリケーションの構築	171
UNIX C 複数接続アプリケーションの構築	173
UNIX C ルーチンの構築	175
UNIX C++ アプリケーションの構築	179
UNIX C++ 複数接続アプリケーションの構築	181
UNIX C++ ルーチンの構築	183
UNIX Micro Focus COBOL アプリケーションの構 築	187
UNIX Micro Focus COBOL ルーチンの構築	189

第 10 章 AIX . . . 191

重要な考慮事項	191
ルーチン用の AIX エクスポート・ファイル	191
AIX ルーチンと CREATE ステートメント	192
AIX 共用ライブラリーの置換	193
AIX での COBOL のインストールに関する考慮 事項	193
IBM C	194
C アプリケーションのビルド・スクリプト	194
AIX C アプリケーションのコンパイルとリンク のオプション	195
C ルーチンのビルド・スクリプト	196
AIX C ルーチンのコンパイルとリンクのオプシ ョン	197
AIX での C マルチスレッド・アプリケーション の構築	198
VisualAge C++	199
C++ アプリケーションのビルド・スクリプト	199
AIX C++ アプリケーションのコンパイルとリン クのオプション	200
C++ ルーチンのビルド・スクリプト	201
AIX C++ ルーチンのコンパイルとリンクのオプ ション	202
AIX での C++ マルチスレッド・アプリケーシ ョンの構築	203
VisualAge C++ 構成ファイル	204
構成ファイルによる VisualAge C++ プログラム の構築	204
構成ファイルによる C++ DB2 API アプリケー ションの構築	205
構成ファイルによる C++ 組み込み SQL アプリ ケーションの構築	206
構成ファイルによる C++ ストアード・プロシー ジャーの構築	207
構成ファイルによる C++ ユーザー定義関数の構 築	208
IBM COBOL Set for AIX	209
AIX での IBM COBOL コンパイラーの構成	209
AIX での IBM COBOL アプリケーションの構 築	210
IBM COBOL アプリケーションのビルド・スク リプト	212
AIX IBM COBOL アプリケーションのコンパイ ルとリンクのオプション	212
AIX での IBM COBOL ルーチンの構築	213
IBM COBOL ルーチンのビルド・スクリプト	215
AIX IBM COBOL ルーチンのコンパイルとリン クのオプション	215
Micro Focus COBOL	217
AIX での Micro Focus COBOL コンパイラーの 構成	217
Micro Focus COBOL アプリケーションのビル ド・スクリプト	218
AIX Micro COBOL アプリケーションのコンパ イルとリンクのオプション	218
Micro Focus COBOL ルーチンのビルド・スクリ プト	219

AIX Micro Focus COBOL ルーチンのコンパイル とリンクのオプション	220	Linux での Micro Focus COBOL コンパイラー の構成.	252
REXX	221	Micro Focus COBOL アプリケーションのビル ド・スクリプト.	253
AIX での REXX アプリケーションの構築.	221	Linux Micro COBOL アプリケーションのコンパ イルとリンクのオプション.	254
第 11 章 HP-UX	223	Micro Focus COBOL ルーチンのビルド・スクリ プト	254
HP-UX C	223	Linux Micro Focus COBOL ルーチンのコンパイ ルとリンクのオプション.	255
C アプリケーションのビルド・スクリプト	223		
HP-UX C アプリケーションのコンパイルとリン クのオプション.	224		
C ルーチンのビルド・スクリプト	226		
HP-UX C ルーチンのコンパイルとリンクのオプ ション.	227		
HP-UX での C マルチスレッド・アプリケーシ ョンの構築	229		
HP-UX C++	230		
C++ アプリケーションのビルド・スクリプト	230		
HP-UX C++ アプリケーションのコンパイルとリン クのオプション.	231		
C++ ルーチンのビルド・スクリプト.	232		
HP-UX C++ ルーチンのコンパイルとリンクのオ プション.	233		
HP-UX での C++ マルチスレッド・アプリケー ションの構築	235		
Micro Focus COBOL	236		
HP-UX での Micro Focus COBOL コンパイラー の構成.	236		
Micro Focus COBOL アプリケーションのビル ド・スクリプト.	237		
HP-UX Micro COBOL アプリケーションのコン パイルとリンクのオプション.	238		
Micro Focus COBOL ルーチンのビルド・スクリ プト	238		
HP-UX Micro Focus COBOL ルーチンのコンパ イルとリンクのオプション.	239		
第 12 章 Linux.	241		
Linux C	241		
C アプリケーションのビルド・スクリプト	241		
Linux C アプリケーションのコンパイルとリン クのオプション.	242		
C ルーチンのビルド・スクリプト	244		
Linux C ルーチンのコンパイルとリンクのオプ ション.	244		
Linux での C マルチスレッド・アプリケーシ ョンの構築	246		
Linux C++	246		
C++ アプリケーションのビルド・スクリプト	246		
Linux C++ アプリケーションのコンパイルとリン クのオプション.	247		
C++ ルーチンのビルド・スクリプト.	249		
Linux C++ ルーチンのコンパイルとリンクのオ プション.	249		
Linux での C++ マルチスレッド・アプリケーシ ョンの構築	251		
Micro Focus COBOL	251		
		第 13 章 Solaris	257
		Solaris C	257
		C アプリケーションのビルド・スクリプト	257
		Solaris C アプリケーションのコンパイルとリン クのオプション.	258
		C ルーチンのビルド・スクリプト	259
		Solaris C ルーチンのコンパイルとリンクのオプ ション.	260
		Solaris での C マルチスレッド・アプリケーシ ョンの構築	261
		Solaris C++	262
		C++ アプリケーションのビルド・スクリプト	262
		Solaris C++ アプリケーションのコンパイルとリン クのオプション.	263
		C++ ルーチンのビルド・スクリプト.	265
		Solaris C++ ルーチンのコンパイルとリンクのオ プション.	265
		Solaris での C++ マルチスレッド・アプリケー ションの構築	267
		Micro Focus COBOL	268
		Solaris での Micro Focus COBOL コンパイラー の構成.	268
		Micro Focus COBOL アプリケーションのビル ド・スクリプト.	268
		Solaris Micro Focus COBOL アプリケーションの コンパイルとリンクのオプション.	269
		Micro Focus COBOL ルーチンのビルド・スクリ プト	270
		Solaris Micro Focus COBOL ルーチンのコンパイ ルとリンクのオプション.	270
		第 14 章 Windows	273
		WCHARTYPE CONVERT プリコンパイル・オプシ ョン	274
		オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数.	275
		Windows Management Instrumentation (WMI)	275
		Microsoft Visual Basic	276
		Visual Basic による ADO アプリケーションの構 築	276
		Visual Basic による疎結合トランザクションの構 築	279
		Visual Basic 疎結合トランザクション・プロジェ クトのトラブルシューティング	281

Visual Basic による RDO アプリケーションの構築	283
Visual Basic でのオブジェクトのリンクと埋め込み (OLE) オートメーション	284
.NET	285
C# .NET アプリケーションの構築	285
C# .NET アプリケーションのバッチ・ファイル	286
C# .NET アプリケーションのコンパイルとリンクのオプション	287
Visual Basic .NET アプリケーションの構築	289
Visual Basic .NET アプリケーションのバッチ・ファイル	290
Visual Basic .NET アプリケーションのコンパイルとリンクのオプション	291
Common Language Runtime (CLR) .NET ルーチンの構築	293
C# .NET ルーチンのバッチ・ファイル	296
Visual Basic .NET ルーチンのバッチ・ファイル	296
CLR .NET ルーチンのコンパイルとリンクのオプション	297
Microsoft Visual C++	298
Visual C++ による ADO アプリケーションの構築	298
Visual C++ でのオブジェクトのリンクと埋め込み (OLE) オートメーション	300
Windows での C/C++ アプリケーションの構築	301
C/C++ アプリケーションのバッチ・ファイル	303
Windows C/C++ アプリケーションのコンパイルとリンクのオプション	304
Windows での C/C++ ルーチンの構築	305
C/C++ ルーチンのバッチ・ファイル	308
Windows C/C++ ルーチンのコンパイルとリンクのオプション	309
Windows での C/C++ 複数接続アプリケーションの構築	310
IBM VisualAge COBOL	313
Windows での IBM COBOL コンパイラーの構成	313
Windows での IBM COBOL アプリケーションの構築	314
IBM COBOL アプリケーションのバッチ・ファイル	316
Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション	317
Windows での IBM COBOL ルーチンの構築	318
IBM COBOL ルーチンのバッチ・ファイル	319
Windows IBM COBOL ルーチンのコンパイルとリンクのオプション	320
Micro Focus COBOL	322
Windows での Micro Focus COBOL コンパイラーの構成	322
Windows での Micro Focus COBOL アプリケーションの構築	323
Micro Focus COBOL アプリケーションのバッチ・ファイル	324

Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション	325
Windows での Micro Focus COBOL ルーチンの構築	325
Micro Focus COBOL ルーチンのバッチ・ファイル	327
Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション	327
オブジェクト REXX	328
Windows でのオブジェクト REXX アプリケーションの構築	328

第 4 部 付録 329

付録 A. DB2 Universal Database 技術情報 331

DB2 ドキュメンテーションおよびヘルプ	331
DB2 ドキュメンテーションの更新	331
DB2 インフォメーション・センター	332
DB2 インフォメーション・センターのインストールのシナリオ	334
DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)	336
DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)	339
DB2 インフォメーション・センターの呼び出し	342
コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール	343
希望する言語での DB2 インフォメーション・センターのトピックの表示	344
DB2 PDF 資料および印刷された資料	345
DB2 の基本情報	345
管理情報	346
アプリケーション開発情報	347
ビジネス・インテリジェンス情報	348
DB2 Connect 情報	348
入門情報	348
チュートリアル情報	349
オプション・コンポーネント情報	349
リリース・ノート	350
PDF ファイルからの DB2 資料の印刷方法	351
DB2 の印刷資料の注文方法	352
DB2 ツールからコンテキスト・ヘルプを呼び出す	353
コマンド行プロセッサからメッセージ・ヘルプを呼び出す	354
コマンド行プロセッサからコマンド・ヘルプを呼び出す	354
コマンド行プロセッサから SQL 状態ヘルプを呼び出す	355
DB2 チュートリアル	355
DB2 トラブルシューティング情報	356
アクセス支援	357

キーボードによる入力およびナビゲーション	357
アクセスしやすい表示	358
支援テクノロジーとの互換性	358
アクセスしやすい資料	358
Ⅰ ドット 10 進シンタックス・ダイアグラム	359
Ⅰ DB2 Universal Database 製品の共通基準認証	361

付録 B. 特記事項	363
-----------------------------	------------

商標	365
--------------	-----

索引	367
---------------------	------------

IBM と連絡をとる	373
-----------------------------	------------

製品情報	373
----------------	-----

DB2 アプリケーション開発へようこそ

DB2 Developer's Edition 製品 ix | 本書について xi

この前書きでは、DB2 アプリケーション開発、とりわけ DB2 Developer's Edition 製品を使用して開発を始めるのに必要な情報が提供されています。また、3 冊の「アプリケーション開発ガイド」すべての概要も記述されています。

DB2 Developer's Edition で使用可能な製品のインストール方法に関する指示は、PDF CD から選択できる該当する「概説およびインストール」資料を参照するか、製品 CD のインストールに関する説明を確認してください。

ご自分のコンピュータで DB2 資料にアクセスしたいものの、まだ DB2 インフォメーション・センターをインストールしていない場合には、336 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』または 339 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』を参照してください。DB2 インフォメーション・センターには、DB2 Universal Database および DB2 関連製品の資料が含まれています。

DB2 Developer's Edition 製品

DB2® Universal Database には、DB2 Personal Developer's Edition と DB2 Universal Developer's Edition という、アプリケーション開発のための 2 つの製品パッケージがあります。Personal Developer's Edition では、Linux と Windows® オペレーティング・システムで稼働する、DB2 Universal Database™ および DB2 Connect™ Personal Edition 製品が提供されています。DB2 Universal Developer's Edition では、こうしたオペレーティング・システムに加えて、AIX®、HP-UX、さらには Solaris オペレーティング環境での DB2 製品が用意されています。サポートされるオペレーティング・システムの詳細なリストについては、IBM® 担当員にご連絡ください。

これらの製品に付属しているソフトウェアを使用すると、あるオペレーティング・システム上で稼働し、同じオペレーティング・システムまたは別のオペレーティング・システム上のデータベースにアクセスするアプリケーションを開発してテストできます。たとえば、Windows オペレーティング・システムで実行されるものの、AIX などの UNIX® オペレーティング・システム上のデータベースにアクセスできるアプリケーションを作成可能です。Developer's Edition 製品の使用条件については、使用許諾契約書を参照してください。

Personal Developer's Edition には、アプリケーションを開発してテストするのに必要なすべてのコードが含まれた数枚の CD-ROM が備えられています。各メディア・パックに、以下が含まれています。

- DB2 Universal Database 製品 CD-ROM (Linux および Windows オペレーティング・システム用)。それぞれの CD-ROM には、サポートされる各オペレーティング・システムの DB2 サーバーおよび Application Development Client が入っています。これらの CD-ROM は、アプリケーションのテストのみを目的に提供さ

れています。データベースをインストールして使用する必要がある場合には、Universal Database 製品を購入して有効なライセンスを取得しなければなりません。

- DB2 Connect Personal Edition。この製品をインストールして使用する必要がある場合には、DB2 Connect Personal Edition を購入して有効なライセンスを取得しなければなりません。
- PDF 形式の DB2 資料が含まれる DB2 文献 CD-ROM
- HTML 形式の DB2 資料が含まれる DB2 インフォメーション・センター CD-ROM
- DB2 Net Search Extender (Windows のみ)
- DB2 Spatial Extender (Windows のみ)
- VisualAge[®] for Java[™], Entry Edition

Universal Developer's Edition には、DB2 がサポートするすべてのオペレーティング・システム用の CD-ROM、および以下が含まれています。

- DB2 Universal Database Personal Edition、Workgroup Server Edition、および Enterprise Server Edition
- DB2 Connect Personal Edition および DB2 Connect Enterprise Edition
- すべてのプラットフォーム用の Administration Client。このクライアントには、コントロール・センターおよびイベント・アナライザーなどの、データベース管理用のツールが含まれています。また、このクライアントを使用すると、どのシステムでもアプリケーションを実行できます。
- すべてのプラットフォーム用の Application Development Client。このクライアントには、アプリケーション開発ツール、サンプル・プログラム、およびヘッダー・ファイルがあります。各 DB2 AD クライアントには、アプリケーションを開発するのに必要なものすべてが揃っています。
- すべてのプラットフォーム用のランタイム・クライアント。任意のシステムで、ランタイム・クライアントからアプリケーションを実行できます。ランタイム・クライアントには、DB2 コントロール・センターおよびイベント・アナライザーなどの管理クライアントの機能はないので、わずかなスペースしか必要としません。
- PDF 形式の DB2 資料が含まれる DB2 文献 CD-ROM
- HTML 形式の DB2 資料が含まれる DB2 インフォメーション・センター CD-ROM
- DB2 Net Search Extender
- DB2 Spatial Extender

加えて、どちらの Developer's Edition にも、アプリケーションの開発に有用な他のソフトウェアが含まれています。こうしたソフトウェアはその時々によって変わり、使用許諾契約書が同封されています。

関連概念:

- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 UDB Enterprise Server Edition』

- 「DB2 Universal Database クライアント機能 概説およびインストール」の『DB2 Run-Time Client』
- 「DB2 Universal Database クライアント機能 概説およびインストール」の『DB2 Administration Client』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 Workgroup Server Edition』

関連タスク:

- 「DB2 Universal Database Personal Edition 概説およびインストール」の『DB2 Personal Edition のインストール - 概要 (Windows)』
- 「DB2 Universal Database Personal Edition 概説およびインストール」の『DB2 Personal Edition のインストール - 概要 (Linux)』

関連資料:

- 3 ページの『DB2 Application Development Client』

本書について

「アプリケーション開発ガイド」は、DB2 アプリケーションのコーディング、デバッグ、ビルド、および実行に関して知っておくべきことを説明した 3 冊からなるガイドです。

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」では、DB2 クライアントで実行されるスタンドアロン DB2 クライアントをコーディングする際に知っておくべきことを説明しています。以下の情報を扱います。
 - DB2 でサポートされているプログラミング・インターフェース。DB2 Developer's Edition、サポートされているプログラミング・インターフェース、Web アプリケーションを作成するための機能、および DB2 が提供するルーチンやトリガーなどのプログラミング機能についてハイレベルな説明がなされています。
 - DB2 アプリケーションが従うべき一般的な構造。データベース中のデータ値やリレーションシップの推奨される保守方法や、許可に関する考慮事項が説明されており、アプリケーションのテストとデバッグ方法に関する情報もあります。
 - 動的組み込み SQL と 静的組み込み SQL。組み込み SQL に関する一般的な考慮事項、および DB2 アプリケーションで静的 SQL と動的 SQL を使用する際の特有な考慮事項。
 - C/C++、COBOL、Perl、および REXX などのサポートされているホストおよびインタープリター言語と、これらの言語で書かれたアプリケーションでの組み込み SQL の使用方法。
 - DB2 .NET Data Provider、および OLE DB .NET Data Provider と ODBC .NET Data Provider。
 - Java (JDBC と SQLJ) および WebSphere Application Server で使用する Java アプリケーションを構築する際の考慮事項。
 - IBM OLE DB Provider for DB2 Server。IBM OLE DB Provider による OLE DB サービス、コンポーネント、およびプロパティのサポートに関する一般

情報。 ActiveX Data Objects (ADO) 用の OLE DB インターフェースを使用する Visual Basic および Visual C++ アプリケーション特有の情報があります。

- 各国語サポートの問題。照合シーケンス、コード・ページとロケールから派生する問題、および文字変換などの一般トピックが説明されています。 DBCS コード・ページ、EUC 文字セット、および日本語と中国語 (繁体字) EUC および UCS-2 環境に適用される問題についても説明されます。
- トランザクション管理。マルチサイト更新を実行するアプリケーション、および並行トランザクションを実行するアプリケーションに適用される問題が説明されています。
- パーティション・データベース環境におけるアプリケーション。パーティション・データベース環境における指示 DSS、ローカル・バイパス、バッファ挿入、アプリケーションのトラブルシューティングについて説明します。
- 一般的に使用されるアプリケーション技法。生成列と ID 列、宣言済み一時表の使用方法、およびトランザクションを管理するためのセーブポイントの使用法について説明されます。
- 組み込み SQL アプリケーションの使用がサポートされている SQL ステートメント。
- ホストおよび iSeries 環境にアクセスするアプリケーション。ホストおよび iSeries 環境にアクセスする組み込み SQL アプリケーションに関する問題。
- EBCDIC バイナリー照合のシミュレーション。
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」では、ルーチン、ラージ・オブジェクト、ユーザー定義タイプ、およびトリガーを含むサーバー・サイド・オブジェクトを使用したプログラミングを行う上で知っておくべき事柄が説明されています。以下の情報を扱います。
 - ルーチン (ストアド・プロシージャ、ユーザー定義関数、およびメソッド)。以下のことも扱われています。
 - ルーチンのパフォーマンス、セキュリティ、ライブラリー管理上の考慮事項、および制限。
 - 外部ルーチンを含むルーチン、および CREATE ステートメントの作成。
 - プロシージャのパラメーター・モードおよびパラメーターの取り扱い。
 - プロシージャの結果セット。
 - デバッグおよび条件処理を含む SQL プロシージャ。
 - ユーザー定義のスカラーおよび表関数。
 - ユーザー定義のスカラーおよび表関数呼び出し (FIRST 呼び出し、FINAL 呼び出しなど) およびスクラッチパッド。
 - メソッド。
 - 許可、および外部ルーチンのバインディング。
 - C、Java、.NET Common Language Runtime、および OLE オートメーション・ルーチンの言語特有の考慮事項。
 - ルーチンの呼び出し。
 - 関数選択。
 - 特殊タイプと LOB の関数への引き渡し。

- コード・ページとルーチン。
 - LOB の使用法とロケーター、参照変数、および CLOB データを含むラージ・オブジェクト。
 - ユーザー定義特殊タイプ (UDT) (以下のことの説明も含まれます)。強い型定義、UDT の定義とドロップ、構造型による表の作成、特定のアプリケーション用の特殊タイプと型付き表の使用、複数の特殊タイプの取り扱いとそれらの間のキャスト、特殊タイプ間の比較と代入、特殊タイプ列における UNION 操作。
 - ユーザー定義構造型 (以下のことも説明されています)。インスタンスの保管インスタンス生成、構造型の階層、構造型の動作の定義、メソッドの動的ディスパッチング、比較関数、cast 関数、コンストラクター関数、および構造型用の mutator メソッドと observer メソッド。
 - 型付き表 (以下のことも説明されています)。オブジェクトの作成・ドロップ・置換・保管、システム生成オブジェクト ID の定義、およびオブジェクト ID 列における制約。
 - 参照タイプ (以下のことも説明されています)。型付き表のオブジェクト間のリレーションシップ、参照のあるセマンティック・リレーションシップ、および参照保全とスコープ限定参照。
 - 型付き表と型付きビュー (以下のことも説明されています)。列型としての構造型、トランスフォーム関数とトランスフォーム・グループ、ホスト言語プログラムのマッピング、および構造型ホスト変数。
 - トリガー (以下のことも説明されています)。INSERT / UPDATE / DELETE トリガー、参照制約との相互作用、作成に関するガイドライン、細分性、活動化時間、遷移変数と表、トリガー・アクション、多重トリガー、および複数のトリガーと制約とルーチン間の協同。
 - 「アプリケーション開発ガイド アプリケーションの構築および実行」では、DB2 でサポートされている以下のオペレーティング・システムにおいて DB2 アプリケーションを構築して実行する上で知っておくべきことが説明されています。
 - AIX
 - HP-UX
 - Linux
 - Solaris
 - Windows
- 以下の情報を扱います。
- DB2 がサポートするコンパイラーとインタープリターを含め、アプリケーションを構築するためのサポートされているサーバーとソフトウェア。
 - DB2 サンプル・プログラム・プログラム・ファイル、makefile、ビルド・ファイル、およびエラー・チェック・ユーティリティー・ファイル。
 - Java および WebSphere MQ 関数用の特定の命令を含む、アプリケーション開発環境のセットアップ方法。
 - サンプル・データベースのセットアップ方法。
 - 旧バージョンの DB2 からのアプリケーションの移行方法。
 - Java アプレット、アプリケーション、およびルーチンのビルドと実行の仕方。

- | - SQL プロシージャのビルドと実行の仕方。
- | - C/C++ アプリケーションとルーチンのビルドと実行の仕方。
- | - IBM COBOL および Micro Focus COBOL アプリケーションとルーチンのビルドと実行の仕方。
- | - AIX および Windows における REXX アプリケーションのビルドと実行の仕方。
- | - Windows での C# および Visual Basic .NET アプリケーション、さらに CLR .NET ルーチンのビルドと実行の仕方。
- | - Windows における Visual Basic および Visual C++ を使用した ActiveX Data Object (ADO) のあるアプリケーションのビルドと実行の仕方。
- | - Windows における Visual C++ を使用した リモート・データ・オブジェクトのあるアプリケーションのビルドと実行の仕方。

第 1 部 アプリケーション開発環境

第 1 章 DB2 環境サポート

DB2 Application Development Client	3	HP-UX でサポートされる開発ソフトウェア	12
データベース・マネージャー・インスタンス	5	Linux でサポートされる開発ソフトウェア	14
DB2 でサポートされるサーバー	8	Solaris でサポートされる開発ソフトウェア	20
DB2 でサポートされる開発ソフトウェア	9	Windows でサポートされる開発ソフトウェア	23
AIX でサポートされる開発ソフトウェア	10		

アプリケーション開発ガイドのこのボリュームでは、アプリケーション開発のための DB2 サポートについて説明しています。本書は、DB2 アプリケーションを開発するための環境をセットアップするのに必要な情報、また開発したアプリケーションをこの環境でコンパイル、リンク、および実行するための手順を段階的に説明しています。本書は、DB2 Universal Database バージョン 8.2 用の DB2 アプリケーション開発 (DB2 AD) クライアントを使用して、以下のプラットフォーム用のアプリケーションを構築する方法を説明しています。

- AIX
- HP-UX
- Linux
- Solaris オペレーティング環境
- Windows

DB2 Application Development Client

DB2 Application Development (DB2 AD) Client は、分散リレーショナル・データベース体系 (DRDA) を実現する DB2 サーバーおよびアプリケーション・サーバーにアクセスするアプリケーションを開発するためのツールおよび環境サポートを提供します。

DB2 AD Client をインストールすれば、DB2 アプリケーションを構築し、実行することができます。次の DB2 クライアントで DB2 アプリケーションを実行することもできます。

- DB2 Run-Time Client
- DB2 Administration Client

サポートされているプラットフォームの DB2 AD Client には、以下のものがあります。

- **C/C++、COBOL、および Fortran プリコンパイラー** (ただし、該当するプラットフォームでこの言語がサポートされている場合)。
- プログラミング・ライブラリー、組み込みファイル、およびコード・サンプルを含む、**組み込み SQL アプリケーション・サポート**。
- ODBC SDK への移植や ODBC SDK とのコンパイルが容易に行えるアプリケーションを開発するためのプログラミング・ライブラリー、組み込みファイル、およびコード例を含む、**DB2 コール・レベル・インターフェース (DB2 CLI) アプリケーション・サポート**。ODBC SDK は、Windows オペレーティング・システムの場合は Microsoft、および他のサポートされるプラットフォームの場合はさまざまなベンダーから入手可能です。Windows オペレーティング・システムの場合

合、DB2 クライアントには Microsoft ODBC Software Developer's Kit で開発されたアプリケーションをサポートする、ODBC ドライバーが含まれています。他のすべてのプラットフォームについては、そのプラットフォーム用の ODBC SDK があれば、それを使用して開発されたアプリケーションをサポートする、任意でインストールされた ODBC ドライバーが DB2 クライアントに含まれます。Windows オペレーティング・システム用の DB2 クライアントだけに ODBC Driver Manager が含まれています。

- Java アプリケーションおよびアプレットを開発する JDBC (Java Database Connectivity) サポート、および Java Embedded SQL アプリケーションとアプレットを開発する DB2 Java Embedded SQL (DB2 SQLJ) サポートを含む、**DB2 Java Enablement**。
- **Java Development Kit** または同等のソフトウェアは、サポートされるすべてのオペレーティング・システム用の DB2 に付属しています。特定のバージョンに関する詳細は、ご使用のオペレーティング・システムの「サポートされる開発ソフトウェア」のトピックを参照してください。
 - 10 ページの『AIX でサポートされる開発ソフトウェア』
 - 12 ページの『HP-UX でサポートされる開発ソフトウェア』
 - 14 ページの『Linux でサポートされる開発ソフトウェア』
 - 20 ページの『Solaris でサポートされる開発ソフトウェア』
 - 23 ページの『Windows でサポートされる開発ソフトウェア』
- Windows オペレーティング・システム上の、**ActiveX Data Objects (ADO)** およびオブジェクトのリンクと埋め込み (**OLE**) オートメーション・ルーチン (**UDF** とストアード・プロシージャ)。これには、Microsoft Visual Basic および Microsoft Visual C++ のコード・サンプルのインプリメンテーションも含まれます。また、Microsoft Visual Basic でインプリメントされた Remote Data Object (RDO) を持つコード・サンプル。
- Windows オペレーティング・システム上の、オブジェクトのリンクと埋め込みデータベース (**OLE DB**) 表関数。
- Windows オペレーティング・システム上の、**C#** および **Visual Basic .NET** アプリケーション、さらに **CLR .NET** ルーチン。
- **DB2 デベロップメント・センター**。これは、ルーチン (ストアード・プロシージャとユーザー定義関数) の迅速な開発と構造型をサポートするグラフィカル・アプリケーションです。デベロップメント・センターは、ワークステーションから z/OS までの DB2 ファミリー全体をサポートする単一の開発環境として機能します。デベロップメント・センターは、スタンドアロンのアプリケーションとして立ち上げることができますが、コントロール・センター、コマンド・エディター、またはタスク・センターなどの DB2 Universal Database のセンターから立ち上げることもできます。デベロップメント・センターは、Java を使用してインプリメントするので、データベース接続の管理には Java Database Connectivity (JDBC) API を使用します。また、デベロップメント・センターには、以下の各開発環境用の DB2 Development Add-In も備えられています。
 - Microsoft Visual C++ バージョン 6
 - Microsoft Visual Basic バージョン 6
 - Microsoft Visual InterDev バージョン 6

- SQL ステートメントのプロトタイプまたはデータベースの随時照会の実行のための、コマンド・エディターまたはコマンド行プロセッサ (CLP) を介する対話式 SQL。
- 他のアプリケーション開発ツールの製品が、それらの製品内で DB2 用のプリコンパイラー・サポートを直接実現するための文書化された API のセット。たとえば AIX 上の IBM COBOL は、このインターフェースを使用します。プリコンパイラー・サービス API のセットに関する情報は、以下の DB2 アプリケーション開発 Web サイトにある PDF ファイル `prepapi.pdf` に記載されています。
<http://www.ibm.com/software/data/db2/udb/ad>
- **SQL92 および MVS 適合 flagger**。これは、ISO/ANSI SQL92 Entry Level 基準に適合しないアプリケーションや、DB2 UDB (OS/390 および z/OS 版) がサポートしないアプリケーション内の組み込み SQL ステートメントを識別します。ワークステーション上で開発したアプリケーションを他のプラットフォームに移行する場合には、Flagger によって構文の非互換性が示されるため、時間が節約できます。

関連資料:

- 「コマンド・リファレンス」の『PRECOMPILE コマンド』
- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

データベース・マネージャー・インスタンス

DB2® は、同じマシン上の複数のデータベース・マネージャー・インスタンスをサポートします。1 つのデータベース・マネージャー・インスタンスには、それ独自の構成ファイル、ディレクトリー、およびデータベースがあります。

各データベース・マネージャー・インスタンスは、複数のデータベースを管理することができます。しかし、1 つのデータベースが属することができるのは 1 つのインスタンスだけです。以下の図に、その関係を示してあります。

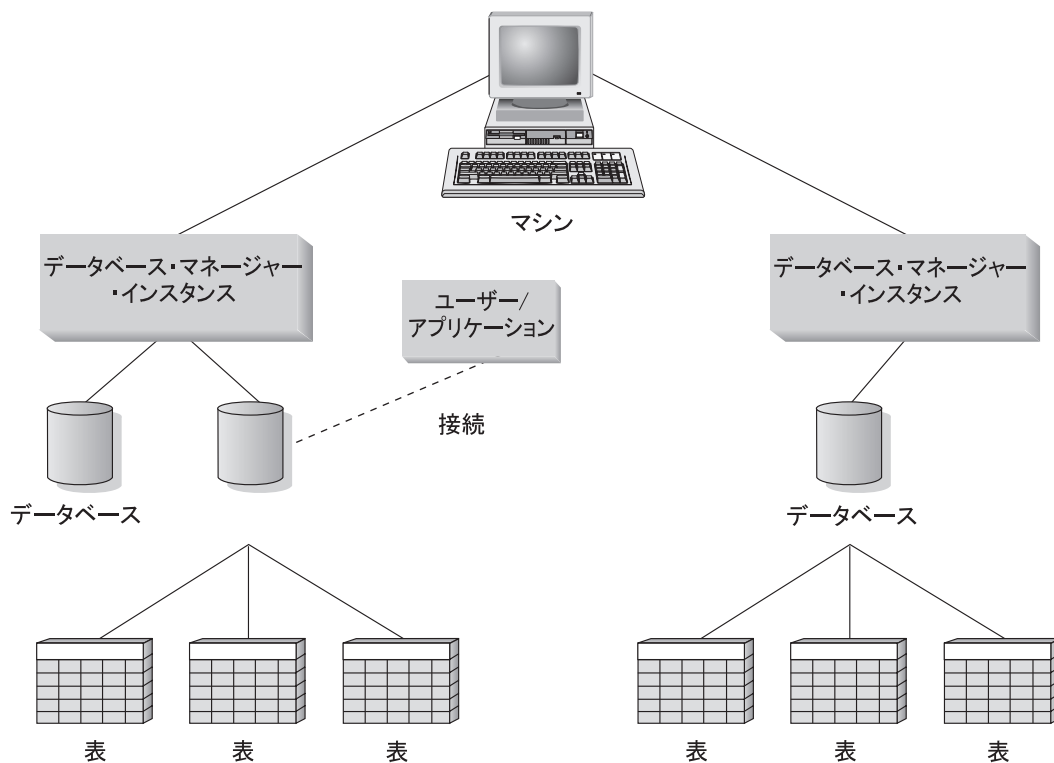


図 1. データベース・マネージャー・インスタンス

データベース・マネージャー・インスタンスでは、その柔軟性のおかげで同じマシン上で複数のデータベース環境を持つことができます。たとえば、1 つのデータベース・マネージャー・インスタンスを開発用に、別のインスタンスを実動用にすることができます。

UNIX[®] Enterprise Server Edition (ESE) サーバーを使用すると、異なるデータベース・マネージャー・インスタンス上で別々の DB2 のバージョンを使用することができます。たとえば、1 つのデータベース・マネージャー・インスタンスで DB2 Universal Database バージョン 7.1 を実行し、別のインスタンスで DB2 Universal Database バージョン 8.2 を実行することができます。DB2 バージョン 8 より前は、1 つのバージョン・レベル内では 1 つのリリースとフィックスパック・レベルしかサポートされていません。たとえば、DB2 バージョン 7.1 および DB2 バージョン 7.2 は、UNIX サーバー上に共存することはできません。DB2 バージョン 8 では、DB2 バージョン 8.1 と DB2 バージョン 8.2 などの同一の UNIX サーバー上に複数のフィックスパック・レベルを共存させることができます。

Windows[®] サーバーでは、各データベース・マネージャー・インスタンスの DB2 は、同じバージョン、リリース、およびフィックスパック・レベルでなければなりません。1 つのデータベース・マネージャー・インスタンスで DB2 Universal Database バージョン 7.1 を実行し、別のインスタンスで DB2 Universal Database バージョン 8.2 を実行することはできません。

使用するそれぞれのインスタンスについて、以下のことを知っておく必要があります。

インスタンス名

UNIX プラットフォームの場合は、データベース・マネージャー・インスタンスを作成するときに指定する、有効なユーザー名です。

Windows オペレーティング・システムの場合は、最大 8 文字の英数字ストリングです。DB2 インスタンスは、インストール時に作成されます。

インスタンス・ディレクトリー

インスタンスがあるホーム・ディレクトリー。

UNIX プラットフォームでは、インスタンス・ディレクトリーは `$HOME/sqlllib` です。 `$HOME` はインスタンス所有者のホーム・ディレクトリーです。

Windows オペレーティング・システムの場合、インスタンス・ディレクトリーは `%DB2PATH%\instance_name` です。変数 `%DB2PATH%` によって DB2 のインストール先が指定されます。 `%DB2PATH%` のデフォルトのインストール値は `%Program Files%\IBM\SQLLIB` であるため、このデフォルト値を変更しない限り、DB2 のインストール先のドライブに応じて、 `%DB2PATH%` は `drive :%Program Files%\IBM\SQLLIB` を指します。

Windows サーバーでのインスタンス・パスは、以下のいずれかに基づいて作成されます。

`%DB2PATH%\%DB2INSTANCE%`

(たとえば、`C:%Program Files%\IBM\SQLLIB\DB2`)

または、`DB2INSTPROF` が定義されている場合、

`%DB2INSTPROF%\%DB2INSTANCE%`

(たとえば、`C:%PROFILES\DB2`)

クライアント・マシンからは読み取りアクセスしかできないネットワーク・ドライブ上での DB2 の実行をサポートするために、Windows サーバーで `DB2INSTPROF` 環境変数を使用します。この場合、DB2 は `drive :%Program Files%\IBM\SQLLIB` を指すように設定され、また `DB2INSTPROF` はローカル・パス (`C:%PROFILES` など) を指すように設定されます。これには、カタログや構成などのインスタンス固有情報すべてが入っています。DB2 はこれらのファイルへの更新アクセスを必要とするためです。

関連概念:

- 「管理ガイド: インプリメンテーション」の『データベース・マネージャーの複数インスタンス』
- 「管理ガイド: インプリメンテーション」の『UNIX オペレーティング・システムでの複数インスタンス』
- 「管理ガイド: インプリメンテーション」の『Windows オペレーティング・システムでの複数インスタンス』

関連タスク:

- 30 ページの『データベース・マネージャー構成ファイルの更新』
- 「管理ガイド: インプリメンテーション」の『現行インスタンスの設定』

- 「管理ガイド: インプリメンテーション」の『追加のインスタンスの作成』

関連資料:

- 「コマンド・リファレンス」の『GET DATABASE MANAGER CONFIGURATION コマンド』
- 「コマンド・リファレンス」の『UPDATE DATABASE MANAGER CONFIGURATION コマンド』

DB2 でサポートされるサーバー

特定のプラットフォームで実行するアプリケーションを開発するには、DB2 AD クライアントを使用します。ただし、アプリケーションからは、次のオペレーティング・システム・サーバー上のリモート・データベースにアクセスすることができます。

- DB2 (AIX 版)
- DB2 (HP-UX 版)
- DB2 (Linux 版)
- DB2 (OS/2 版)
- DB2 (NUMA-Q 版)
- DB2 (Solaris 版)
- DB2 for Windows NT
- DB2 for Windows 2000
- DB2 for Windows XP
- DB2 for Windows Server 2003
- 分散リレーショナル・データベース体系 (DRDA) に準拠するアプリケーション・サーバー。たとえば、以下のものがあります。
 - DB2 for OS/390 and z/OS
 - DB2 for AS/400 and iSeries
 - DB2 Server for VSE & VM (以前は、SQL/DS (VM および VSE))
 - IBM 以外のデータベース・ベンダーからの、DRDA に準拠するアプリケーション・サーバー

注:

1. DB2 バージョン 8 の HP-UX 64 ビット・サーバーは、DB2 バージョン 7 の 64 ビット・ローカル・アプリケーションの実行をサポートしません。
2. DB2 (OS/2 版) は、DB2 バージョン 8 では利用できません。
3. DB2 (NUMA-Q 版) は PTX オペレーティング・システムで稼働し、DB2 バージョン 7 でのみ使うことができます。
4. DB2 バージョン 6 およびバージョン 7 の 32 ビット・クライアントから DB2 バージョン 8 の Windows 64 ビット・サーバーへの接続では、SQL 要求のみがサポートされます。バージョン 7 の 64 ビット・クライアントからの接続はサポートされません。

関連タスク:

- 「インストールおよび構成 補足」の『サーバー・データベースへのリモート・アクセスの構成』

関連資料:

DB2 でサポートされる開発ソフトウェア

DB2 バージョン 8 は、以下のオペレーティング・システム用のコンパイラー、インタープリター、およびソフトウェアをサポートします。

- AIX
- HP-UX
- Linux
- Solaris オペレーティング環境
- Windows

DB2 は、これらのオペレーティング・システムのそれぞれの 32 ビットおよび 64 ビット・バージョンをサポートします。多くの場合、これらのオペレーティング・システム上の 32 ビットおよび 64 ビット環境でのアプリケーションの構築はそれぞれ異なります。64 ビット・オペレーティング・システム環境では、DB2 は COBOL アプリケーションまたはルーチン (ストアード・プロシージャおよびユーザー定義関数) の実行をサポートしていません。他の言語では、Linux IA64 と Linux zSeries を除き、DB2 は、サポートされているすべての 64 ビット・オペレーティング環境での 32 ビット・アプリケーションおよびルーチンの実行をサポートします。

Windows 上の IBM COBOL を除き、これらのオペレーティング・システムに関して述べられているコンパイラー情報では、そのオペレーティング・システム用にユーザーが DB2 プリコンパイラーを使用しており、一覧中のコンパイラーのいずれかに組み込まれている可能性のあるプリコンパイラー・サポートを使用していないことを前提としています。Windows 上の IBM COBOL の場合、DB2 は IBM COBOL プリコンパイラーと DB2 プリコンパイラーをサポートします。

最新の DB2 コンパイラー情報とそれに関連したソフトウェアの更新については、次の DB2 アプリケーション開発の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

ソフトウェア・サポートに関しては、以下の点に気を付けてください。

- **Fortran および REXX。** DB2 では、DB2 Universal Database バージョン 5.2 において、該当する言語サポート・レベルを超えて Fortran および REXX の機能が拡張されることはありません。
- **Perl。** 本書の発刊時点では、Perl Database Interface (Perl DBI) バージョン 0.93 以降用の DB2 UDB ドライバー (DBD::DB2) のリリース 0.76 は、AIX、HP-UX、Linux、Solaris および Windows 用のものが用意されています。このドライバーは、次の Web サイトからダウンロードすることができます。

<http://www.ibm.com/software/data/db2/perl>

- **PHP。** PHP は Web ベースのアプリケーションまたはコマンド行から DB2 にアクセスするための手段として使用でき、AIX、HP-UX、Linux、Solaris、および Windows で使用できます。本書の印刷の時点で、最新バージョンは PHP 4.3.4 です。以下のサイトから、最新バージョンの PHP をダウンロードできます。

<http://www.php.net>

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

AIX でサポートされる開発ソフトウェア

DB2 (AIX 版) は、以下のオペレーティング・システムをサポートします。

AIX バージョン 4.3.3 (32 ビットのみ)

保守レベル 11

March 2003 C++ Runtime PTF を適用 (ダウンロード・リンクについては、下の C++ セクションを参照してください)、および

JFS ファイル・システムの場合:

APAR IY49385

Java の場合:

- OpenGL.OpenGL_X.rte.base
- OpenGL.OpenGL_X.rte.soft
- X11.adt.lib

AIX バージョン 5.1.0 (32 ビットおよび 64 ビット)

保守レベル 5

March 2003 C++ Runtime PTF を適用 (ダウンロード・リンクについては、下の C++ セクションを参照してください)、および

JFS ファイル・システムの場合:

APAR IY48735

JFS2 ファイル・システムの場合:

APAR IY49254

Java の場合:

Recommended Maintenance Package AIX 5100-04 および APAR IY46667

1000 を超える db2agent を実行する場合:

APAR IY49220。さらに、db2start の前、または AIX ブートアップで、「vmtune -T 0」を指定します。

AIX バージョン 5.2.0 (32 ビットおよび 64 ビット)

保守レベル 2、および

Concurrent I/O (CIO) および Direct I/O (DIO) のマウント・ボリュームの場合:

APAR IY49129 および IY49346

JFS ファイル・システムの場合:

APAR IY48339

JFS2 ファイル・システムの場合:

APAR IY49304

Java の場合:

Recommended Maintenance Package AIX 5200-01 および APAR IY46668

1000 を超える db2agent を実行し、32 ビット AIX カーネルを使用する場合: APAR IY49885。さらに、db2start の前、または AIX ブートアップで、「vmo -o pta_balance_threshold=0」を指定します。

注: 以下のコマンドを使用して、特定の APAR がシステムにインストールされているかどうかを照会できます。

```
instfix -v -i -k <APAR>
```

たとえば、`instfix -v -i -k IY31254`

DB2 (AIX 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C IBM C for AIX バージョン 5.0.2.3

IBM C for AIX バージョン 6.0

C++ March 2003 C++ Runtime PTF を適用した IBM VisualAge C++ バージョン 5.0.2.3

<http://www-1.ibm.com/support/docview.wss?rs=0&q=x1C.rte&uid=swg24004427>

March 2003 C++ Runtime PTF を適用した IBM VisualAge C++ バージョン 6.0

<http://www-1.ibm.com/support/docview.wss?rs=0&q=x1C.rte&uid=swg24004427>

COBOL

IBM COBOL Set for AIX バージョン 1.1

Micro Focus COBOL Server Express バージョン 2.2 (Service Pack 1 を適用)

注: COBOL サポートは、32 ビット用のみです。

Fortran

IBM XL Fortran for AIX バージョン 4.1 (32 ビットの場合のみ) および 5.1.0 (32 ビットおよび 64 ビットの場合)

Java Java Developer Kit バージョン 1.3.1 および IBM 提供の Java Runtime Environment (AIX 版) バージョン 1.3.1

IBM Developer Kit for AIX, Java Technology Edition, Version 1.4.1 Service Release 1 (AIX 5.1 および 5.2 のみ)

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

REXX IBM AIX REXX/6000 AISPO 製品番号: 5764-057

IBM Object REXX (AIX 版) バージョン 1.1

REXXSAA 4.00

注: REXX サポートは、32 ビット用のみです。

DB2 (AIX 版) のソフトウェア・サポートの更新の詳細は、以下の DB2 アプリケーション開発の Web サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

関連資料:

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 195 ページの『AIX C アプリケーションのコンパイルとリンクのオプション』
- 197 ページの『AIX C ルーチンのコンパイルとリンクのオプション』
- 200 ページの『AIX C++ アプリケーションのコンパイルとリンクのオプション』
- 202 ページの『AIX C++ ルーチンのコンパイルとリンクのオプション』
- 212 ページの『AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション』
- 215 ページの『AIX IBM COBOL ルーチンのコンパイルとリンクのオプション』
- 218 ページの『AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 220 ページの『AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 サーバーのインストール要件 (AIX)』

HP-UX でサポートされる開発ソフトウェア

DB2 (HP-UX 版) は、以下のオペレーティング・システムをサポートします。

HP-UX (PA-RISC 版)

HP-UX バージョン 11i(11.11) for PA-RISC 2 (以下が適用されたもの)

June 2003 GOLDBASE11i と June 2003 GOLDAPPS11i バンドル、および
パッチ PHSS_26560、PHKL_28489、PHCO_27434、PHCO_29960

HP-UX (IA64 版)

HP-UX バージョン 11i バージョン 2 (B.11.23) Intel Itanium 版 (パッチ
PHKL_30065)

注: Aries バイナリトランスレータを使用して PA-RISC 用に構築された、
DB2 アプリケーションまたはルーチンの HP-UX (IA64 版) での実行
は、サポートされていません。

DB2 for HP-UX (PA-RISC 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C HP C コンパイラーのバージョン B.11.11.02

C++ HP aC++ バージョン A.03.52

COBOL

Micro Focus COBOL Server Express バージョン 2.2 (Service Pack 1 を適用)

注: COBOL サポートは、32 ビット用のみです。

Fortran

HP-UX f90 B.11.01.06

Java HP-UX 32 ビット: Software Developer's Kit と、 Runtime Environment 1.3.1 および 1.4.2.01 for HP-UX 11.0、および Hewlett-Packard の 11i PA-RISC。

HP-UX 64 ビット: Software Developer's Kit と、 Runtime Environment 1.4.2.01 for HP-UX 11.0、および Hewlett-Packard の 11i PA-RISC。

注:

1. 32 ビットの Java ルーチンは、Software Developer's Kit 1.3.1 のみサポートされています。
2. DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、CD-ROM から手動で開発者キットをインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

DB2 for HP-UX (IA64 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C HP C コンパイラーのバージョン A.05.52

C++ HP (aCC) aC++ バージョン A.05.52

注: DB2 for HP-UX (IA64 版) は、使用すべきでないオプション -AP を使用して構築されていたり、または (libstd_v2 ではなく) libstd に従属関係があるような C++ アプリケーションまたはサード・パーティーの C++ ライブラリーをサポートしていません。

Fortran

HP-UX F90 B.11.23

注:

1. 64 ビット・コードを生成するには、+DD64 コンパイル・オプションが必要です。
2. 32 ビット・コードを生成するには、+DD32 コンパイル・オプションが必要です。

Java HP-UX 11 PA-RISC および Itanium ベースのシステム用の、 Software Developer's Kit および Runtime Environment 1.4.2.01 以降。

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされているバージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、CD-ROM から /opt/java1.4 に手動で開発者キットをインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

DB2 (HP-UX 版) のソフトウェア・サポートの更新の詳細は、以下の DB2 アプリケーション開発の Web サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

関連資料:

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 224 ページの『HP-UX C アプリケーションのコンパイルとリンクのオプション』
- 227 ページの『HP-UX C ルーチンのコンパイルとリンクのオプション』
- 231 ページの『HP-UX C++ アプリケーションのコンパイルとリンクのオプション』
- 233 ページの『HP-UX C++ ルーチンのコンパイルとリンクのオプション』
- 238 ページの『HP-UX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 239 ページの『HP-UX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 サーバーのインストール要件 (HP-UX)』

Linux でサポートされる開発ソフトウェア

DB2 for Linux は、以下のオペレーティング・システムをサポートします。

- Intel x86 (32 ビット) 版 Linux
- s/390 および zSeries 版 Linux
- AMD64 版 Linux
- PowerPC 版 Linux
- IA64 版 Linux

各アーキテクチャーでサポートされるディストリビューション、およびカーネルとライブラリーのレベルに関しては、以下にアクセスしてください。

<http://www.ibm.com/db2/linux/validate>

以下の表では、本書の印刷時点でサポートされている DB2 Linux アーキテクチャーについて説明しています。こうしたサポートの更新に関して、検証用の Web サイト (上記) を確認するようぜひお勧めします。

表 1. Intel x86 (32 ビット) 版 Linux

ディストリビューション	カーネル	ライブラリー	コメント
Conectiva Linux Enterprise Edition (CLEE)	2.4.19	glibc 2.2.5	Powered by United Linux 1.0
LINX Rocky Secure Server 2.1	2.4.21	glibc 2.2.5	
Red Flag Advanced Server 4.0	2.4.21-as.2	glibc 2.2.93-5	
Red Flag Function Server 4.0	2.4.20-8smp	glibc 2.2.93-5	
Red Hat Enterprise Linux 2.1 AS/ES/WS	2.4.9-e16	glibc 2.2.4	
Red Hat Enterprise Linux (RHEL) 3 AS/ES/WS	2.4.21-7.EL	glibc-2.3.2-95.3	
SCO Linux 4.0	2.4.19	glibc 2.2.5	Powered by United Linux 1.0
SuSE Pro 8.0	2.4.18	glibc 2.2.5	
SuSE Pro 8.1	2.4.19	glibc 2.2.5	
SuSE Linux Enterprise Server (SLES) 7	2.4.7	glibc 2.2.2	
SuSE Linux Enterprise Server (SLES) 8	2.4.19	glibc 2.2.5	SuSE Service Pack 2 レベルまで確認済み
Turbolinux 7 Server	2.4.9	glibc 2.2.4	
Turbolinux 8 Server	2.4.18-5	glibc 2.2.5	
Turbolinux Enterprise Server 8	2.4.19	glibc 2.2.5	
United Linux 1.0	2.4.19	glibc 2.2.5	

表 2. Intel x86 (32 ビット) 版 Linux、非エンタープライズ・ディストリビューション (ベンダーのサポートはもうありません)

ディストリビューション	カーネル	ライブラリー	コメント
Red Hat 7.2	2.4.9-34	glibc 2.2.4	
Red Hat 7.3	2.4.18	glibc 2.2.5	
Red Hat 8.0	2.4.18-14	glibc 2.2.93-5	
SuSE 7.3	2.4.10	glibc 2.2.4	

表 3. s/390 および zSeries 版 Linux (s/390 では 31 ビット・カーネル・バージョンが、また zSeries では 64 ビット・カーネル・バージョンがそれぞれサポートされます)

ディストリビューション	カーネル	ライブラリー	コメント
Red Hat 7.2	2.4.9-38	glibc 2.2.4	

表 3. s/390 および zSeries 版 Linux (s/390 では 31 ビット・カーネル・バージョンが、また zSeries では 64 ビット・カーネル・バージョンがそれぞれサポートされます) (続き)

ディストリビューション	カーネル	ライブラリー	コメント
SuSE Linux Enterprise Server (SLES) 7	2.4.7-58	glibc 2.2.4	compat.rpmc++ には libstdc++ 6.1 が含まれます。Java 用 JDK 1.3.1 SR 1 を使用します。
SuSE Linux Enterprise Server (SLES) 8	2.4.19	glibc 2.2.5	Powered by United Linux 1.0
Turbo Linux Enterprise Server (TLES) 8	2.4.19	glibc 2.2.5	Powered by United Linux 1.0
United Linux 1.0	2.4.19	glibc 2.2.5	

表 4. AMD64 版 Linux

ディストリビューション	カーネル	ライブラリー	コメント
Red Hat Enterprise Linux (RHEL) 3 AS/ES/WS	2.4.21-7.EL	glibc-2.3.2-95.3	
SuSE Linux Enterprise Server (SLES) 8.0	2.4.19-SMP	glibc 2.2.5-16	

表 5. PowerPC 版 Linux (iSeries および pSeries)

ディストリビューション	カーネル	ライブラリー	コメント
Red Hat Enterprise Linux (RHEL) 3 AS	2.4.21-7.EL	glibc-2.3.2-95.3	
SuSE Enterprise Server (SLES) 8	2.4.19-16	glibc 2.2.5	Powered by United Linux 1.0
Turbolinux Enterprise Server 8	2.4.19-16	glibc 2.2.5	Powered by United Linux 1.0
United Linux 1.0	2.4.19	glibc 2.2.5	

表 6. IA64 版 Linux

ディストリビューション	カーネル	ライブラリー	コメント
Red Hat Enterprise Linux 2.1 AS/ES/WS	2.4.18-e.12smp	glibc	
Red Hat Enterprise Linux (RHEL) 3 AS/ES/WS	2.4.21-7.EL	glibc-2.3.2-95.3	
SuSE Linux Enterprise Server (SLES) 8	2.4.19-SMP	glibc 2.2.5	Powered by United Linux 1.0
United Linux 1.0	2.4.19	glibc 2.2.5	

DB2 for Linux (Intel x86 版) は、以下のプログラム言語とコンパイラーをサポートします。

C GNU/Linux gcc バージョン 2.95.3 および 2.96

C++ GNU/Linux g++ バージョン 2.95.3 および 2.96

COBOL

Micro Focus COBOL Server Express バージョン 2.2 (Service Pack 1 を適用)

Java IBM Developer Kit and Runtime Environment for Linux、Java 2 Technology Edition、バージョン 1.3.1 および 1.4.1、Service Release 1、32 ビット・バージョン

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

REXX Object REXX Interpreter for Linux バージョン 2.1

DB2 for Linux (s/390 版) または DB2 for Linux (zSeries 版) 上の 32 ビット・インスタンスは、次のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 2.95.3

C++ GNU/Linux g++ バージョン 2.95.3

COBOL

Micro Focus COBOL Server Express バージョン 2.2 (Service Pack 1 を適用)

Java IBM zSeries Developer Kit for Linux、Java 2 Technology Edition (Sun 1.3.1 および 1.4.1 Service Release 1 SDK レベル)

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

REXX Object REXX 2.2.0 for Linux/390

DB2 for Linux (zSeries 版) 上の 64 ビット・インスタンスは、以下のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 3.2

C++ GNU/Linux g++ バージョン 3.2

Java IBM zSeries Developer Kit for Linux、Java 2 Technology Edition (Sun 1.4.1 Service Release 1 SDK レベル)

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。

Perl Perl 5.8

DB2 for Linux (AMD64 版) 上の 32 ビット・インスタンスは、以下のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 3.2 および 3.3

注: 「-m32」コンパイラー・オプションを使用して、32 ビットのアプリケーションまたはルーチン (ストアード・プロシージャおよびユーザー定義関数) を生成する必要があります。

C++ GNU/Linux g++ バージョン 3.2 および 3.3

注:

1. このバージョンの GNU/Linux g++ コンパイラーは、一部の fstream 関数の整数パラメーターを受け入れません。詳しくは、コンパイラーの資料を参照してください。
2. 「-m32」コンパイラー・オプションを使用して、32 ビットのアプリケーションまたはルーチン (ストアード・プロシージャおよびユーザー定義関数) を生成する必要があります。

Java IBM Developer Kit and Runtime Environment for Linux x86、Java 2 Technology Edition、バージョン 1.3.1 Service Release 4、32 ビット・バージョン、およびバージョン 1.4.1 Service Release 1、32 ビット・バージョン。

注:

1. DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。
2. SuSE SLES 8 によって提供される Developer kit 1.3.1 Service Release をインストール済みの場合、DB2 のインストール前にアンインストールしてください。アンインストールしない場合、DB2 は、推奨の開発者キットをインストールできません。SuSE SLES 8 提供の開発者キットをアンインストールせずに DB2 をインストールした場合、README を参照し、開発者キットを手動で更新してください。

Perl Perl 5.8

PHP PHP 4.3.4 以上

DB2 for Linux (AMD64 版) 上の 64 ビット・インスタンスは、以下のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 3.2 および 3.3

C++ GNU/Linux g++ バージョン 3.2 および 3.3

注: このバージョンの GNU/Linux g++ コンパイラーは、一部の fstream 関数の整数パラメーターを受け入れません。詳しくは、コンパイラーの資料を参照してください。

Java DB2 は現在、64 ビットの Java Developer Kit for Linux (AMD64 版) をサポートしません。

Perl Perl 5.8

PHP PHP 4.3.4 以上

DB2 for Linux (PowerPC 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 3.2

C++ GNU/Linux g++ バージョン 3.2

注: このバージョンの GNU/Linux g++ コンパイラーは、一部の fstream 関数の整数パラメーターを受け入れません。詳しくは、コンパイラーの資料を参照してください。

Java 32 ビット・インスタンスの場合: IBM Developer Kit and Runtime Environment for Linux、Java 2 Technology Edition、バージョン 1.3.1 および 1.4.1 Service Release 1、32 ビット・バージョンの PowerPC 版。

64 ビット・インスタンスの場合: IBM Developer Kit and Runtime Environment for Linux、Java 2 Technology Edition、バージョン 1.4.1 Service Release 1、64 ビット・バージョンの PowerPC 版。

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、サポートされている開発者キットを CD-ROM から手動でインストールする必要があります。

Perl Perl 5.8

PHP PHP 4.3.4 以上

DB2 for Linux (IA64 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C GNU/Linux gcc バージョン 3.2

C++ GNU/Linux g++ バージョン 3.2

注: このバージョンの GNU/Linux g++ コンパイラーは、fstream 関数の整数パラメーターを受け入れません。詳しくは、コンパイラーの資料を参照してください。

Java IBM Developer Kit および Runtime Environment (Linux 版)、Java 2 Technology Edition バージョン 1.3.1、64 ビット・バージョン。この Developer Kit を使用するには、gcc 3.2 と gcc3 libstdc++ ランタイム・ライブラリーもインストールする必要があります。

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はそれをインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、CD-ROM から手動で開発者キットをインストールする必要があります。

Perl Perl 5.8

注: DB2 の 32 ビット・アプリケーションまたはルーチン (ストアド・プロシージャおよびユーザー定義関数) の実行は、Linux IA64 ではサポートされていません。

DB2 for Linux のソフトウェア・サポートの更新の詳細は、以下の DB2 アプリケーション開発の Web サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

関連資料:

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 242 ページの『Linux C アプリケーションのコンパイルとリンクのオプション』
- 244 ページの『Linux C ルーチンのコンパイルとリンクのオプション』
- 247 ページの『Linux C++ アプリケーションのコンパイルとリンクのオプション』
- 249 ページの『Linux C++ ルーチンのコンパイルとリンクのオプション』
- 「DB2 Universal Database Personal Edition 概説およびインストール」の『DB2 Personal Edition のインストール要件 (Linux)』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 サーバーのインストール要件 (Linux)』
- 254 ページの『Linux Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 255 ページの『Linux Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Solaris でサポートされる開発ソフトウェア

DB2 (Solaris 版) は、以下のオペレーティング・システムをサポートします。

Solaris

DB2 Workgroup Server Edition は、次のバージョンの Solaris オペレーティング環境でサポートされます。

Solaris 7 (32 ビット) 「推奨 & セキュリティー・パッチ」 + 107226-17 + 107153-01 + 106327-10 適用済み

Solaris 8 (32 ビット)「推奨 & セキュリティー・パッチ」 + 108921-12 + 108940-24 + 108434-03 + 108528-12 適用済み

Solaris 9 (32 ビット)

DB2 Enterprise Server Edition は、次のバージョンの Solaris オペレーティング環境でサポートされます。

Solaris 7 (32 ビット)「推奨 & セキュリティー・パッチ」 + 107226-17 + 107153-01 + 106327-10 適用済み

Solaris 7 (64 ビット)「推奨 & セキュリティー・パッチ」 + 107226-17 + 107153-01 + 106300-11 + 106327-10 適用済み

Solaris 8 (32 ビット)「推奨 & セキュリティー・パッチ」 + 108921-12 + 108940-24 + 108434-03 + 108528-12 適用済み

Solaris 8 (64 ビット)「推奨 & セキュリティー・パッチ」 + 108921-12 + 108940-24 + 108435-03 + 108434-03 + 108528-12 適用済み

Solaris 9 (32 ビット)

Solaris 9 (64 ビット)

「推奨 & セキュリティー・パッチ」は、次のところから入手できます。

<http://sunsolve.sun.com>

SunSolve Online Web サイトで、左パネルで「パッチ」メニュー項目をクリックして、「ダウンロード」セクションから「推奨&セキュリティパッチ」を選択します。

さらに J2SE Solaris Patch Cluster も必要です。 <http://sunsolve.sun.com> Web サイトから入手できます。 SunSolve Online Web サイトで、左パネルで「パッチ」メニュー項目をクリックして、「各種パッチのダウンロード」セクションから「推奨パッチクラスタ」を選択します。

DB2 を Solaris にインストールするには、SUNWlibC ソフトウェアが必要です。

64 ビット Fujitsu PRIMEPOWER システムで DB2 を使用するには、以下も必要になります。

- Solaris 8 Kernel Update Patch 108528-16 以降 (パッチ 912040-01 のフィックスを取得するため)
- Solaris 9 Kernel Update Patch 112233-01 以降 (パッチ 912041-01 のフィックスを取得するため)

Solaris 版の Fujitsu PRIMEPOWER パッチは、以下のリンクの FTSI からダウンロードできます。

<https://download.ftsi.fujitsu.com/>

DB2 (Solaris 版) は、以下のプログラム言語およびコンパイラーをサポートします。

C Forte C バージョン 5.0、6、6.1、および 6.2

Sun ONE Studio 7 および 8、Compiler Collection

C++ Forte C++ バージョン 5.0、6、6.1、および 6.2
Sun ONE Studio 7 および 8、Compiler Collection

COBOL

Micro Focus COBOL Server Express バージョン 2.2 (Service Pack 1 を適用)

注: COBOL サポートは、32 ビット用のみです。

Fortran

SPARCompiler Fortran バージョン 4.2 および 5.0

Java Solaris 32 ビット: Sun Microsystems の Java Development Kit (JDK) (Solaris 版) バージョン 1.3.1 および 1.4.2

Solaris 64 ビット: Sun Microsystems の Java Development Kit (JDK) (Solaris 版) バージョン 1.4.2

注: DB2 インストールが以前の DB2 バージョン 8 インストールの更新である場合以外は、開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンの JDK をインストールします。以前の DB2 バージョン 8 インストールを更新している場合には、CD-ROM から手動で JDK をインストールする必要があります。

Perl Perl 5.004_04 以上、DBI 0.93 以上

PHP PHP 4.3.4 以上

DB2 (Solaris 版) のソフトウェア・サポートの更新の詳細は、以下の DB2 アプリケーション開発の Web サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

関連資料:

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 258 ページの『Solaris C アプリケーションのコンパイルとリンクのオプション』
- 260 ページの『Solaris C ルーチンのコンパイルとリンクのオプション』
- 263 ページの『Solaris C++ アプリケーションのコンパイルとリンクのオプション』
- 265 ページの『Solaris C++ ルーチンのコンパイルとリンクのオプション』
- 269 ページの『Solaris Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 270 ページの『Solaris Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 サーバーのインストール要件 (Solaris)』

Windows でサポートされる開発ソフトウェア

DB2 (Windows 32 ビット・オペレーティング・システム版) は、以下をサポートします。

Microsoft Windows XP

Microsoft Windows Server 2003

Microsoft Windows 2000

Windows ターミナル・サーバーには Service Pack 2 が必要です。

Microsoft Windows NT

Service Pack 6a 付きのバージョン 4.0 以降。

Microsoft Windows ME

Microsoft Windows 98

DB2 (Windows 32 ビット・オペレーティング・システム版) は、以下のプログラム言語をサポートします。

Basic Microsoft Visual Basic 6.0 Professional Edition

Microsoft .NET Framework バージョン 1.0 および 1.1 それぞれに対する
Microsoft Visual Basic .NET 7.0 および 7.1

注: DB2 .NET Data Provider をインストールする DB2 インストール・プログラムを使用する前に .NET Framework をインストールする必要があります。

C# Microsoft .NET Framework バージョン 1.0 および 1.1 それぞれに対する Microsoft Visual C# .NET Compiler バージョン 7.0 および 7.1

注: DB2 .NET Data Provider をインストールする DB2 インストール・プログラムを使用する前に .NET Framework をインストールする必要があります。

C/C++ Microsoft Visual C++ バージョン 6.0

Microsoft Visual C++ .NET 2002 および 2003

32 ビット・アプリケーション用の Intel C++ Compiler バージョン 6 以上

COBOL

Micro Focus COBOL バージョン 4.0.20

Micro Focus COBOL Net Express バージョン 3.1.0

IBM VisualAge COBOL バージョン 3.0.4 以上

Java IBM Developer Kit and Runtime Environment for Windows、Java 2 Technology Edition、バージョン 1.3.1 および 1.4.1 Service Release 1、32 ビット・バージョン

注: 開発者キットがまだインストールされていないと、DB2 はサポートされている最新バージョンをインストールします。

Java Development Kit (JDK) 1.3.1 (Win32 版) (Sun Microsystems 社提供)

Perl Perl 5.004_04、DBI 0.93

PHP PHP 4.3.4 以上

REXX IBM Object REXX for Windows NT/95 バージョン 1.1

IBM Object REXX for Windows の入手法の詳細は、以下を参照してください。

<http://www.ibm.com/software/ad/obj-rexx/>

Microsoft Windows スクリプティング ホスト

バージョン 5.1

注: Windows 2000 および Windows XP の場合: ODBC を使用する COM+ オブジェクトを持つ DB2 アプリケーション、または OLE DB リソース・プールが使用不可にされた OLE DB Provider for ODBC を使用するアプリケーションを実行するには、Windows 2000 Service Pack 3 または Windows XP Service Pack 1 を使用する必要があります。アプリケーション環境が適格であるかどうか分からない場合は、適切な Windows サービス・レベルをインストールすることが推奨されています。詳細については、以下の Microsoft Knowledge Base 記事を参照してください。

<http://support.microsoft.com/default.aspx?scid=KB;EN-US;306414>

これらの Service Pack は、DB2 サーバー自体または DB2 製品の一部として出荷されているアプリケーションでは必要ありません。

DB2 (Windows 64 ビット・オペレーティング・システム版) は、以下をサポートします。

Microsoft Windows XP 64-bit Edition

Microsoft Windows Server 2003

DB2 (Windows 64 ビット・オペレーティング・システム版) は、以下のプログラム言語をサポートします。

C/C++ Intel C++ Compiler for Itanium バージョン 7.1 (Microsoft Software Developer's Kit 3790 以上を使用)。

Intel Itanium アーキテクチャー用の Microsoft の C/C++ コンパイラー (Microsoft Software Developer's Kit 3790 で使用可能)。

Java IBM Developer Kit and Runtime Environment for Windows、Java 2 Technology Edition、バージョン 1.4.1 Service Release 1、64 ビット・バージョン

注: 開発者キットがまだインストールされていない場合には、DB2 がそれをインストールします。

Perl Perl 5.004_04、DBI 0.93

PHP PHP 4.3.4 以上

Microsoft Windows スクリプティング ホスト

バージョン 5.1

注: Windows Server 2003 サポートには、以下のものが含まれています。

- Windows Server 2003, Standard Edition
- Windows Server 2003, Enterprise Edition
- Windows Server 2003, Datacenter Edition

DB2 (Windows 版) のソフトウェア・サポートの更新の詳細は、以下の DB2 アプリケーション開発の Web サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

関連資料:

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 「*DB2 Universal Database* サーバー機能 概説およびインストール」の『DB2 サーバーのインストール要件 (Windows)』
- 304 ページの『Windows C/C++ アプリケーションのコンパイルとリンクのオプション』
- 309 ページの『Windows C/C++ ルーチンのコンパイルとリンクのオプション』
- 317 ページの『Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション』
- 320 ページの『Windows IBM COBOL ルーチンのコンパイルとリンクのオプション』
- 325 ページの『Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 327 ページの『Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 「*DB2 Universal Database Personal Edition* 概説およびインストール」の『DB2 Personal Edition のインストール要件 (Windows)』
- 291 ページの『Visual Basic .NET アプリケーションのコンパイルとリンクのオプション』
- 287 ページの『C# .NET アプリケーションのコンパイルとリンクのオプション』
- 297 ページの『CLR .NET ルーチンのコンパイルとリンクのオプション』

第 2 章 セットアップ

一般的なセットアップ情報	27	サンプル・データベース	52
アプリケーション開発環境のセットアップ	27	サンプル・データベースのセットアップ	52
DB2 ルーチンの共用ライブラリーの再構築	29	サンプル・データベースの作成	52
データベース・マネージャ構成ファイルの更新	30	ホスト・サーバーまたは AS/400 および iSeries	
Java 環境のセットアップ	31	サーバーでのサンプル・データベースの作成	54
DB2 WebSphere MQ 関数のセットアップ	33	サンプル・データベースのカatalog	55
UNIX	36	サンプル・データベース・ユーティリティのバ	
UNIX アプリケーション開発環境のセットアップ	36	インディグ	55
UNIX 環境変数の設定	37	アプリケーションの移行	57
UNIX Java 環境のセットアップ	38	アプリケーションの DB2 バージョン 8 への移行	57
AIX Java 環境のセットアップ	40	Java アプリケーション、ルーチン、およびアプレ	
HP-UX Java 環境のセットアップ	41	ットの移行	59
Linux Java 環境のセットアップ	43	32 ビット環境から 64 ビット環境へのアプリケー	
Solaris Java 環境のセットアップ	44	ションの移行	60
Windows	45	アプリケーション移植性の確保	63
Windows アプリケーション開発環境のセットアッ		2 つのバージョンの DB2 でのアプリケーション	
プ	45	の実行	64
Windows Java 環境のセットアップ	49	次に行うこと	68

一般的なセットアップ情報

DB2 CLI のセットアップ情報については、「コール・レベル・インターフェースガイドおよびリファレンス」を参照してください。

アプリケーション開発環境のセットアップ

DB2 アプリケーションを構築して実行するには、コマンド行プロセッサ (CLP) スクリプトまたは SQL プロシージャー (下記を参照) を使用している場合以外は、ご使用のオペレーティング・システムがサポートしているいずれかのプログラミング言語のコンパイラまたはインタープリターを使う必要があります。各自の開発上の要件に合わせて DB2 環境をセットアップして構成しなければなりません。旧バージョンの DB2 から DB2 アプリケーションを移行するには、従うべき特定の手順があります。また、テストのための DB2 サンプル・データベースを作成することもできます。	
---	--

前提条件:

まず最初に非 DB2 アプリケーションを構築することによって、使用する予定の DB2 のサポート対象のコンパイラまたはインタープリターの環境が必ず正しくセットアップされるようにします。その後、問題が発生した場合、ご使用のコンパイラまたはインタープリターに付属している資料を参照してください。

使用しているクライアントまたはサーバーのワークステーションに Application Development Client をインストールします。リモート・クライアントからアプリケーションを開発する場合、DB2 データベース・サーバーが置かれているマシンにクライアント・マシンからアクセスできることを確認します。また、クライアント

からデータベースに正常に接続できることも確認します。コマンド行プロセッサ (CLP) またはクライアント構成アシスタント (CCA) を使用して、接続をテストすることができます。

手順:

開発環境をセットアップするには、次のようにします。

1. デフォルトが受け入れ可能でない限り、 30 ページの『データベース・マネージャー構成ファイルの更新』にある指示に従います。
2. DB2 CLI、Java、または WebSphere プロシージャーを使用してプログラミングする予定の場合、ご使用の環境を構成する必要があります。プラットフォーム固有の何らかの変更を行う前に、以下の指示に従ってください。
 - CLI 環境のセットアップ
 - 31 ページの『Java 環境のセットアップ』
 - 33 ページの『DB2 WebSphere MQ 関数のセットアップ』
3. 以下の指示を参考にして、オペレーティング・システム環境を構成します。
 - 36 ページの『UNIX アプリケーション開発環境のセットアップ』
 - 45 ページの『Windows アプリケーション開発環境のセットアップ』
4. オプション: 52 ページの『サンプル・データベースのセットアップ』

SQL プロシージャー

DB2 バージョン 8.2 以降では、SQL プロシージャーの作成の際にサーバー上では C または C++ コンパイラーは必要ないため、そのセットアップは不要です。SQL プロシージャーが作成されると、プロシージャー・ステートメントは、他の SQL ステートメントで実行される場合と同様、ネイティブ表現に変換されて、データベース・カタログ内に保管されます。SQL プロシージャーが呼び出されるときに、この表現はカタログからロードされ、DB2 エンジンで実行されます。

関連概念:

- 5 ページの『データベース・マネージャー・インスタンス』
- 57 ページの『アプリケーションの DB2 バージョン 8 への移行』

関連タスク:

- 30 ページの『データベース・マネージャー構成ファイルの更新』
- 「コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻」の『CLI 環境のセットアップ』
- 31 ページの『Java 環境のセットアップ』
- 33 ページの『DB2 WebSphere MQ 関数のセットアップ』
- 36 ページの『UNIX アプリケーション開発環境のセットアップ』
- 45 ページの『Windows アプリケーション開発環境のセットアップ』
- 52 ページの『サンプル・データベースのセットアップ』

関連資料:

- 3 ページの『DB2 Application Development Client』
- 8 ページの『DB2 でサポートされるサーバー』

- 9 ページの『DB2 でサポートされる開発ソフトウェア』
- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

DB2 ルーチンの共用ライブラリーの再構築

DB2® は、ストアド・プロシージャとユーザー定義関数に使用される共用ライブラリーがロードされると、それをキャッシュに入れます。ルーチンを開発する場合、同じ共用ライブラリーのロードを何度もテストしたい場合もあります。そのキャッシングの際に、最新バージョンの共用ライブラリーを取得できないこともあります。キャッシングに関連する問題を回避するには、ルーチンのタイプによって方法が異なります。

1. **fenced** された、スレッド・セーフではないルーチン。 データベース・マネージャー構成キーワード `KEEPFENCED` のデフォルト値は、`YES` です。そのため、`fenced` モード・プロセスは存続し続けます。このデフォルト設定によって、ライブラリーの再ロードが妨げられることがあります。 `fenced` された、スレッド・セーフではないルーチンの開発中はこのキーワードの値を `NO` に変更しておいて、最終バージョンの共用ライブラリーをロードする準備ができれば、この値を `YES` に戻すのが最善の方法です。詳しくは、30 ページの『データベース・マネージャー構成ファイルの更新』を参照してください。
2. **信頼できる、またはスレッド・セーフのルーチン。** SQL ルーチン (SQL プロシージャを含む) 以外で、DB2 ルーチン・ライブラリーが信頼できるルーチンまたはスレッド・セーフ・ルーチンに使用される場合、更新されたバージョンが確実に選び出されるようにする唯一の方法は、コマンド行で `db2start` の後に `db2stop` と入力して、DB2 インスタンスをリサイクルすることです。SQL ルーチンではこれは必要ありません。SQL ルーチンが再作成される際には、コンパイラーは新しいユニークなライブラリー名を使用して、競合の可能性を回避するからです。

また SQL ルーチン以外のルーチンの場合、ライブラリーに別の名前 (たとえば、`foo.a` を `foo.1.a` にする) を付けて新規バージョンのルーチンを作成し、その新しいライブラリーで `ALTER PROCEDURE` または `ALTER FUNCTION SQL` ステートメントのいずれかを使用すると、キャッシングの問題を避けることができます。

関連タスク:

- 30 ページの『データベース・マネージャー構成ファイルの更新』

関連資料:

- 「SQL リファレンス 第 2 巻」の『ALTER FUNCTION ステートメント』
- 「SQL リファレンス 第 2 巻」の『ALTER PROCEDURE ステートメント』

データベース・マネージャー構成ファイルの更新

このファイルには、アプリケーション開発のための重要な設定が収められています。

キーワード `KEEPFENCED` のデフォルト値は `YES` です。 `fenced` された、スレッド・セーフではないルーチン (ストアード・プロシージャと UDF) の場合、ルーチン・プロセスは存続し続けます。こうしたルーチンの開発中はこのキーワードの値を `NO` に変更しておいて、最終バージョンの共用ライブラリーをロードする準備ができたなら、この値を `YES` に戻すのが最善の方法です。詳しくは、29 ページの『DB2 ルーチンの共用ライブラリーの再構築』を参照してください。

注: `KEEPFENCED` は、旧バージョンの DB2 では `KEEPDARI` という名称でした。

Java アプリケーション開発の場合、Java Development Kit のインストール先のパスを使用して `JDK_PATH` キーワードを更新する必要があります。

注: `JDK_PATH` は、旧バージョンの DB2 では `JDK11_PATH` という名称でした。

手順:

この設定を変更するには、次のように入力します。

```
db2 update dbm cfg using <keyword> <value>
```

たとえば、キーワード `KEEPFENCED` を `NO` に設定するには、次のようにします。

```
db2 update dbm cfg using KEEPFENCED NO
```

`JDK_PATH` キーワードをディレクトリー `/home/db2inst/jdk13` に設定するには、次のようにします。

```
db2 update dbm cfg using JDK_PATH /home/db2inst/jdk13
```

データベース・マネージャー構成ファイル内の現在の設定を表示するには、以下を入力します。

```
db2 get dbm cfg
```

注: Windows では、このコマンドを DB2 コマンド・ウィンドウに入力する必要があります。

関連概念:

- 29 ページの『DB2 ルーチンの共用ライブラリーの再構築』
- 5 ページの『データベース・マネージャー・インスタンス』

関連タスク:

- 31 ページの『Java 環境のセットアップ』

関連資料:

- 「SQL リファレンス 第 2 巻」の『CREATE FUNCTION ステートメント』
- 「SQL リファレンス 第 2 巻」の『CREATE PROCEDURE ステートメント』
- 「コマンド・リファレンス」の『GET DATABASE MANAGER CONFIGURATION コマンド』

- 「コマンド・リファレンス」の『RESET DATABASE MANAGER CONFIGURATION コマンド』

Java 環境のセットアップ

DB2 データベースにアクセスする Java プログラムは、ご使用のプラットフォームに対応した Java Developer Kit を使用して開発できます。Developer Kit には、Java 用の動的 SQL API である JDBC (Java Database Connectivity) が含まれています。

DB2 JDBC サポートは、DB2 クライアントおよびサーバー上の Java Enablement オプションの一部として提供されます。このサポートによって、JDBC アプリケーションとアプレットを構築し、実行できます。これらには動的 SQL だけが含まれ、Java 呼び出しインターフェースを使用して SQL ステートメントを DB2 に渡します。

DB2 embedded SQL for Java (SQLJ) サポートも、Java 環境の一部として提供されています。DB2 JDBC サポートとともに DB2 SQLJ サポートを利用することで、SQLJ アプレットおよびアプリケーションの構築と実行が可能になります。これらには、静的 SQL が含まれ、DB2 データベースにバインドされた組み込み SQL ステートメントを使用します。

DB2 AD クライアントによって提供される SQLJ サポートには、次のものが含まれます。

- DB2 SQLJ 変換プログラム **sqlj**。これは、SQLJ プログラム中の組み込み SQL ステートメントを Java ソース・ステートメントで置き換え、SQLJ プログラム中に検出される SQL 操作に関する情報を含む、順番に並べられたプロファイルを生成します。
- DB2 SQLJ プロファイル・カスタマイザー **db2sqljcustomize**。これはシリアルライズされたプロファイルに保管された SQL ステートメントをプリコンパイルし、それらを実行時関数呼び出しにカスタマイズし、そして DB2 データベース内にパッケージを生成します。

注: DB2 SQLJ プロファイル・カスタマイザーは、旧バージョンの DB2 では **db2profc** と呼ばれていました。

- DB2 SQLJ プロファイル・プリンター **db2sqljprint**。これはカスタマイズしたバージョンの DB2 プロファイルの内容をプレーン・テキスト形式で印刷します。

注: DB2 SQLJ プロファイル・プリンターは、旧バージョンの DB2 では **db2profp** と呼ばれていました。

- DB2 SQLJ プロファイル・バインド・プログラム **db2sqljbind**。既にカスタマイズされた SQLJ プログラムからパッケージを生成します。

注: CLI ベースのタイプ 2 およびタイプ 3 JDBC ドライバーは推奨されていません。こうしたドライバーに新機能や機能拡張は計画されておらず、DB2 の今後のリリースでも使用できません。こうしたレガシー・ドライバーに代わって、完全に再設計され、より機能拡張された汎用 JDBC ドライバーが備えられています。できるだけ早急に、この新規ドライバーを使用するようにアプリケーションを移行するようお勧めします。

手順:

JDBC Universal Type 2 または JDBC Universal Type 4 接続でアプリケーションを構築、または JDBC Universal Type 4 接続でアプレットを構築するには、TCP/IP listener を実行している必要があります。これを確実に実行するには、次のようにします。

1. 環境変数 DB2COMM を「TCPIP」に設定します。

```
db2set DB2COMM=TCPIP
```

2. サービス・ファイルで指定されるように、データベース・マネージャーの構成ファイルを TCP/IP サービス名で更新します。

```
db2 update dbm cfg using SVCENAME <TCP/IP service name>
```

この設定を有効にするには、db2stop と db2start を実行する必要があります。

注: アプレットと SQLJ プログラムに使用されるポート番号は、データベース・マネージャーの構成ファイルで使用される TCP/IP SVCENAME 番号と同じである必要があります。

DB2 Java アプリケーションを実行するには、ネイティブ・スレッド・サポートを提供する Java 仮想マシン (JVM) をインストールして呼び出さなければなりません。ネイティブ・スレッドを使用して Java アプリケーションを実行するには、コマンド内の `-native` オプションを使用することができます。たとえば、Java サンプル・アプリケーション `DbInfo.class` を実行するには、次のコマンドを使うことができます。

```
java -native DbInfo
```

THREADS_FLAG 環境変数を `native` に設定すれば、Java 仮想マシンによっては、デフォルトのスレッド・サポートとしてネイティブ・スレッドを指定することができます。本書では、ネイティブ・スレッド・サポートがデフォルトであることを前提としています。システム上でネイティブ・スレッドをデフォルトにする方法については、JVM 資料を参照してください。

DB2 Java アプレットを実行するには、ネイティブ・スレッド・サポートまたはグリーン・スレッド・サポートのいずれかを提供する Java 仮想マシンを呼び出すことができます。

上記がすべてインストール済みで稼働状態になったら、以下に示すいずれかのステップを行って、各自のオペレーティング・システムの Java 環境をセットアップすることができます。

- 38 ページの『UNIX Java 環境のセットアップ』
- 49 ページの『Windows Java 環境のセットアップ』

DB2 Java アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

```
http://www.ibm.com/software/data/db2/udb/ad/v8/java
```

関連タスク:

- 38 ページの『UNIX Java 環境のセットアップ』
- 49 ページの『Windows Java 環境のセットアップ』

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『DB2 Universal JDBC ドライバーのインストール』

関連資料:

- ・「コマンド・リファレンス」の『db2sqljcustomize - DB2 SQLJ プロファイル・カスタマイザー・コマンド』
- ・「コマンド・リファレンス」の『db2sqljprint - DB2 SQLJ プロファイル・プリンター・コマンド』
- ・「コマンド・リファレンス」の『db2sqljbind - DB2 SQLJ プロファイル・バインド・プログラム・コマンド』

DB2 WebSphere MQ 関数のセットアップ

DB2 および WebSphere MQ は、メッセージングとデータベース・アクセスを結合するアプリケーションを構成するために使用できます。MQ 関数は、ユーザー定義関数 (UDF) と類似しており、DB2 内でオプションで使用可能です。これらの基本関数を使用して、フェデレーテッド・データ・ソースを更新するために、単一のイベント通知からデータウェアハウジングまで、広範囲にわたるアプリケーションをサポートすることができます。

手順:

DB2 WebSphere MQ 関数をセットアップするには、次のようにします。

1. それぞれの物理マシンに WebSphere MQ をインストールします。

最新のフィックスパックを適用済みの最小限の WebSphere MQ バージョン 5.1 が DB2 Universal Database サーバー上にインストールされていることを確認します。このバージョンの WebSphere MQ がすでにインストールされている場合、次のステップ「WebSphere MQ AMI をインストールする」にスキップします。DB2 バージョン 8 には、DB2 で使用するための WebSphere MQ サーバーのコピーが含まれています。WebSphere MQ のインストール、または既存の WebSphere MQ インストールの更新に関するプラットフォーム固有の指示は、<http://www.ibm.com/software/ts/mqseries/library/manuals> にあるプラットフォーム固有の概説およびインストールの資料にあります。インストール・プロセスを実行する際に、必ずデフォルトのキュー・マネージャーをセットアップしてください。

2. それぞれの物理マシン上に WebSphere MQ Application Messaging Interface AMI をインストールします。

これは、管理作業とプログラミング作業を明確に区別する WebSphere MQ プログラミング・インターフェースの拡張機能です。DB2 WebSphere MQ 関数では、このインターフェースをインストールする必要があります。WebSphere MQ AMI が DB2 サーバー上にすでにインストールされている場合には、次のステップ「DB2 WebSphere MQ ユーザー定義関数を使用可能にして構成する」にスキップします。WebSphere MQ AMI がインストールされていない場合には、DB2 で提供されているインストール・パッケージを使用するか、または WebSphere MQ SupportPacs の Web サイト (<http://www.ibm.com/software/ts/mqseries/txppacs>) から AMI のコピーをダウンロードすることによって、WebSphere MQ AMI をインストールすることができます。

す。AMI は、『Category 3 - Product Extensions』の下にあります。便宜上、WebSphere MQ AMI は DB2 と共に提供されています。このファイルは、sqllib/cfg/mq ディレクトリー内にあります。

ファイルの名前は、オペレーティング・システムごとに異なります。

表 7. WebSphere MQ AMI

オペレーティング・システム	ファイル名
AIX バージョン 4.3 以降	maOf_ax.tar.Z
HP-UX	maOf_hp.tar.Z
Solaris オペレーティング環境	maOf_sol7.tar.Z または mqOf_sol26.tar.Z
Windows	maOf_win.zip

注: DB2 WebSphere MQ 関数は、Linux ではサポートされていません。

圧縮されたインストール・イメージに含まれる readme ファイルで概説されている、通常の AMI インストール・プロセスに従ってください。

3. DB2 WebSphere MQ ユーザー定義関数を使用可能にして構成します。

enable_MQFunctions ユーティリティーは、以下のアクションを実行する柔軟なコマンドです。

- 正しい WebSphere MQ 環境がセットアップされていることをチェックする。
- DB2 WebSphere MQ 関数のデフォルトの構成をインストールおよび作成する。
- これらの関数を使用する指定されたデータベースを使用可能にする。
- 構成が機能していることを確認する。

UNIX 64 ビットで、enable/disable_MQFunctions を実行するには、ランタイム・ライブラリー・パスに \$HOME/sqllib/lib32 を組み込むよう変更する必要があります。以下の設定によって、これを実行します。

AIX

```
LIBPATH=$HOME/sqllib/lib32 enable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc [-q qMgr -force -noValidate]  
LIBPATH=$HOME/sqllib/lib32 disable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc
```

HP-UX

```
SHLIB_PATH=$HOME/sqllib/lib32 enable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc [-q qMgr -force -noValidate]  
SHLIB_PATH=$HOME/sqllib/lib32 disable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc
```

Solaris

```
LD_LIBRARY_PATH=$HOME/sqllib/lib32 enable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc [-q qMgr -force -noValidate]  
LD_LIBRARY_PATH=$HOME/sqllib/lib32 disable_MQFunctions -n dbname ¥  
-u userid -p passwd -v 0pc
```

使用可能化のステップ中に、以下のステップを使って、DB2 WebSphere MQ 関数のデータベースを構成し、使用可能にします。

- Windows の場合、ステップ c に進む。

- b. DB2 インスタンス所有者 (多くの場合 db2inst1) および fenced ユーザー定義関数と関連したユーザー ID (多くの場合 db2fenc1) を、 WebSphere MQ グループ mqm に追加することによって、 UNIX 上の WebSphere MQ 関数を使用可能にする。これは、DB2 関数で WebSphere MQ にアクセスするために必要です。
- c. AMT_DATA_PATH 環境変数を DB2 によって理解されるリストに追加する。ファイル \$HOME/sqllib/profile.env (UNIX) または %DB2PATH%\profile.env (Windows) を編集し、 AMT_DATA_PATH を DB2ENVLIST に追加することができます。また、 **db2set** コマンドを使用することもできます。
- d. 環境変数の変更を有効にするために、データベース・インスタンスを再始動する。
- e. ディレクトリーを \$HOME/sqllib/cfg (UNIX) または %DB2PATH%\cfg (Windows) に変更する。
- f. コマンド **enable_MQFunctions** を実行して、 DB2 WebSphere MQ 関数のデータベースを構成し、使用可能にする。 DB2 ESE 環境では、カタログ・ノードでこのステップのみを実行します。このコマンドの完全な説明については、『enable_MQFunctions』のトピックを参照してください。いくつかの一般的な例が下記に示されています。正常に完了すると、指定されたデータベースが使用可能になり、構成がテストされます。
- g. コマンド行プロセッサを使用してこれらの関数をテストするには、使用可能になったデータベースに接続した後で、以下のコマンドを発行する。

```
values DB2MQ.MQSEND('a test')
values DB2MQ.MQRECEIVE()
```

最初のステートメントは、メッセージ 『a test』 を DB2MQ_DEFAULT_Q キューに送信し、 2 番目のステートメントは、そのメッセージを受け取ります。

注: enable_MQFunctions を実行した後、ユーティリティーは、デフォルトの WebSphere MQ 環境を確立します。また、ユーティリティーは、WebSphere MQ キュー・マネージャー DB2MQ_DEFAULT_MQM およびデフォルトのキュー・マネージャー DB2MQ_DEFAULT_Q も作成します。

AMT_DATA_PATH で示されているディレクトリー内にファイル amt.xml、amthost.xml、および amt.dtd がまだ存在していない場合には、ユーティリティーはそれらのファイルをインストールします。 amthost.xml ファイルが存在し、接続 DB2MQ の定義が含まれていない場合には、適切な情報を持つファイルに行が追加されます。元のファイルのコピーは、DB2MQSAVE.amthost.xml として保管されます。

- 4. トランザクション MQ UDF を使用する場合には、データベースがフェデレーテッド操作のために構成されていることを確認します。これを行うには、以下のコマンドを入力します。

```
update dbm cfg using federated yes
```

- 5. DB2 WebSphere MQ 関数によって提供されるパブリッシュ/サブスクライブ機能を使用するには、それぞれの物理マシン上に MQSeries Integrator または

WebSphere MQ パブリッシュ/サブスクライブ製品拡張機能のいずれかをインストールする必要があります。MQSeries Integrator に関する情報は、以下のサイトにあります。

<http://www.ibm.com/software/ts/mqseries/integrator>

WebSphere MQ パブリッシュ/サブスクライブ機能に関する情報は、以下のサイトにあります。

<http://www.ibm.com/software/ts/mqseries/txppacs>

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『MQSeries 使用可能性』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ 機能の概要』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ メッセージング』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ 関数を使用したメッセージ送信』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ 関数を使用したメッセージ取り出し』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ アプリケーション間の接続』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ 関数を使用した要求 / 応答通信』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『WebSphere MQ 関数を使用したパブリッシュ / サブスクライブ』
- 「IBM DB2 Information Integrator アプリケーション開発者向けガイド」の『DB2 内で WebSphere MQ 機能を使用する方法』

関連資料:

- 「コマンド・リファレンス」の『db2mqlns - MQ Listener コマンド』
- 「コマンド・リファレンス」の『enable_MQFunctions』
- 「コマンド・リファレンス」の『disable_MQFunctions』

UNIX

UNIX DB2 CLI のセットアップ情報については、「コール・レベル・インターフェース ガイドおよびリファレンス」を参照してください。

UNIX アプリケーション開発環境のセットアップ

データベース・インスタンス用の環境変数を設定する必要があります。各データベース・マネージャー・インスタンスにはそれぞれ、db2profile および db2cshrc という 2 つのファイルがあります。これらは、そのインスタンス用の環境変数を設定するためのスクリプトです。

手順:

次のように、ご使用のシェルの正しいスクリプトを実行します。

bash または Korn シェルの場合

```
. $HOME/sqlllib/db2profile
```

C シェルの場合

```
source $HOME/sqlllib/db2cshrc
```

ここで \$HOME は、インスタンス所有者のホーム・ディレクトリーです。

このコマンドを .profile または .login ファイルに組み込めば、ログオン時にコマンドは自動的に実行されます。

ODBC、DB2 CLI、または Java を使用する予定の場合、以下のトピックに示されているステップを行います。

- UNIX ODBC 環境のセットアップ
- UNIX Java 環境のセットアップ

関連概念:

- 37 ページの『UNIX 環境変数の設定』

関連タスク:

- 「コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻」の『UNIX ODBC 環境のセットアップ』
- 38 ページの『UNIX Java 環境のセットアップ』

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』

UNIX 環境変数の設定

ご使用の UNIX® プラットフォームに応じて、以下に示されている環境変数の値は db2profile (bash または korn シェルの場合) あるいは db2cshrc (C シェルの場合) 内に設定され、これらのファイルの呼び出しはインスタンス所有者の .profile (bash または korn シェル) あるいは .login (C シェル) ファイルに入れます。

AIX®:

- PATH。sqlllib/bin を含むいくつかの DB2® ディレクトリーが組み込まれます。
- LIBPATH。ディレクトリー sqlllib/lib が組み込まれます (以下の注を参照)。

HP-UX:

- PATH。sqlllib/bin を含むいくつかの DB2 ディレクトリーが組み込まれます。
- SHLIB_PATH (32 ビットおよび 64 ビット) または LD_LIBRARY_PATH (64 ビット)。ディレクトリー sqlllib/lib が組み込まれます (以下の注を参照)。

Linux および Solaris:

- PATH。sqllib/bin を含むいくつかの DB2 ディレクトリーが組み込まれます。
- LD_LIBRARY_PATH。ディレクトリー sqllib/lib が組み込まれます (以下の注を参照)。

注: 64 ビット DB2 インスタンスにおいて 32 ビットのローカル・アプリケーションを実行する場合、60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』を参照してください。

インスタンスの作成時に、sqllib/userprofile と sqllib/usercshrc というブランク・ファイルが作成されるので、ユーザーはこれらを使用して独自のインスタンス環境を設定できます。これらのファイルは、DB2 フィックスパック・インスタンスの更新 (db2iupdt) または将来のバージョンのインストールの際に変更されることはありません。db2profile または db2cshrc スクリプト中では新規の環境設定値を必要としない場合、それに対応する「ユーザー」スクリプト (db2profile または db2cshrc スクリプトの末尾で呼び出されます) を使用してその設定値をオーバーライドすることができます。インスタンスの移行 (db2imigr) の際には、ユーザーが変更した環境が引き続き使用されるようにするために、ユーザー・スクリプトがコピーされます。

関連タスク:

- 36 ページの『UNIX アプリケーション開発環境のセットアップ』
- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』

UNIX Java 環境のセットアップ

DB2 JDBC サポートを利用して UNIX 上で JDBC プログラムや SQLJ プログラムを実行するために、Java 環境を更新するコマンドが、データベース・マネージャー・ファイル db2profile と db2cshrc に組み込まれています。DB2 インスタンスの作成時には、.bashrc、.profile、および .cshrc の内容を以下のように変更します。

1. THREADS_FLAG を "native" に設定します。 (Solaris のみ)
2. CLASSPATH に以下のものを含めます。
 - "." (現行ディレクトリー)
 - ファイル sqllib/java/db2java.zip
 - ファイル sqllib/java/db2jcc.jar
 - ファイル sqllib/java/db2jcc_license_cu.jar

注: db2jcc_license_cisuz.jar は、DB2 Connect Personal Edition、DB2 Connect Enterprise Edition、および DB2 ESE の CLASSPATH にも組み込みます。こうすると、DB2 for z/OS and OS/390、DB2 for AS/400 and iSeries、および DB2 for VSE & VM に対する追加の接続が設けられます。

SQLJ プログラムを構築するには、次のファイルを組み込むように CLASSPATH を更新します。

sqllib/java/sqlj.zip

Java Developer Kit 1.3 または 1.4 を使用してデータ・ソース・プログラムを構築するには、以下を取得してインストールする必要もあります。

JNDI 1.2.1 クラス・ライブラリー (jndi.jar および providerutil.jar)

<http://java.sun.com/products/jndi/#download>

File System Service Provider 1.2 (fscontext.jar)

<http://java.sun.com/products/jndi/#download>

Java Developer Kit 1.3 の場合、さらに以下を取得してインストールする必要があります。

JDBC 2.0 オプション・パッケージ

<http://java.sun.com/products/jdbc/download.html#spec>

注: JDBC 2.0 オプション・パッケージは、Java Developer Kit 1.4 でデータ・ソース・プログラムを構築する場合は必要ありません。

データ・ソース・プログラムには、CLASSPATH の更新 (以下のファイルの組み込み) が必要です。

- jndi.jar
- fscontext.jar
- providerutil.jar

Java Developer Kit 1.3 の場合、さらに CLASSPATH を更新し、次のいずれかを組み込む必要があります。

- jdbc2_0-stdext.jar
- j2ee.jar

注:

1. Java Developer Kit 1.3 を使用しており、j2ee.jar にて CLASSPATH を更新済みの場合、jdbc2_0-stdext.jar は必要ありません。
2. Java Developer Kit 1.4 の場合、jdbc2_0-stdext.jar および j2ee.jar は CLASSPATH に必要ありません。

データ・ソースのサンプル・プログラムは、sqllib/samples/java/sqlj ディレクトリにあります。詳しくは、sqllib/samples/java の README サンプル・ファイルを参照してください。

注:

1. 他のファイルが CLASSPATH に組み込まれている場合は、必ず上記のファイルをまず指定してください。
2. 将来の DB2 のバージョンでは、このディレクトリはなくなるため、sqllib/java12 ディレクトリへの直接の参照は、すべて除去する必要があります。代わりに、sqllib/java ディレクトリを参照してください。
3. DB2 Java Enablement には、JDBC パッケージ・バインド・プログラム・ユーティリティである db2jdbcbind が組み込まれています。JDBC パッケージは、DB2 バージョン 8 サーバーに自動的にバインドされます。このユーティリティは、JDBC パッケージを、DB2 バージョン 6 や 7 などの下位レベルのサーバー上で作成するためのものです。

手順:

DB2 Java ルーチン (ストアード・プロシージャと UDF) を実行するには、そのマシンの Java Developer Kit のインストール先のパスを組み込むように、サーバー上の DB2 データベース・マネージャー構成を更新する必要があります。そのためには、サーバーのコマンド行で次のように入力します。

```
db2 update dbm cfg using JDK_PATH /home/db2inst/jdk13
```

ただし `/home/db2inst/jdk13` は、Java Developer Kit のインストール先のパスです。

次のコマンドをサーバーで入力して、DB2 データベース・マネージャー構成をチェックし、JDK_PATH フィールドの値が正しいことを確認できます。

```
db2 get dbm cfg
```

出力をファイルにリダイレクトすれば、一層容易に表示できます。JDK_PATH フィールドは、出力の先頭近くに表示されます。

上記のものがインストール済みで稼働状態になったら、以下に示すいずれかのステップを行って、各自の UNIX オペレーティング・システム環境をセットアップすることができます。

- AIX Java 環境のセットアップ
- HP-UX Java 環境のセットアップ
- Linux Java 環境のセットアップ
- Solaris Java 環境のセットアップ

関連タスク:

- 40 ページの『AIX Java 環境のセットアップ』
- 41 ページの『HP-UX Java 環境のセットアップ』
- 43 ページの『Linux Java 環境のセットアップ』
- 44 ページの『Solaris Java 環境のセットアップ』
- 30 ページの『データベース・マネージャー構成ファイルの更新』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『DB2 Universal JDBC ドライバーのインストール』

関連資料:

- 「コマンド・リファレンス」の『GET DATABASE MANAGER CONFIGURATION コマンド』
- 「コマンド・リファレンス」の『RESET DATABASE MANAGER CONFIGURATION コマンド』
- 「コマンド・リファレンス」の『db2jdbcbind - DB2 JDBC パッケージ・バインド・プログラム・コマンド』

AIX Java 環境のセットアップ

以下の指示をインプリメントする前に、38 ページの『UNIX Java 環境のセットアップ』で説明されているセットアップを実行してください。

手順:

AIX 上で DB2 JDBC サポートを利用して Java アプリケーションを構築するには、以下のものがが必要です。

1. 10 ページの『AIX でサポートされる開発ソフトウェア』にリストされている、サポートが提供されるいずれかの開発者キット。
2. DB2 Java Enablement。これは AIX クライアントおよびサーバーに対応した DB2 Universal Database バージョン 8 に装備されています。

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 38 ページの『UNIX Java 環境のセットアップ』
- 52 ページの『サンプル・データベースのセットアップ』

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』

HP-UX Java 環境のセットアップ

以下の指示をインプリメントする前に、38 ページの『UNIX Java 環境のセットアップ』で説明されているセットアップを実行してください。

手順:

HP-UX 上で DB2 JDBC サポートを利用して Java アプリケーションを構築するには、開発マシンで以下をインストールして構成する必要があります。

1. 12 ページの『HP-UX でサポートされる開発ソフトウェア』にリストされている、サポートが提供されるいずれかの開発者キット。
2. DB2 Java Enablement。これは HP-UX クライアントおよびサーバーに対応した DB2 Universal Database バージョン 8 に装備されています。

HP-UX での HP-UX 32 ビット Java ルーチン (ストアード・プロシージャとユーザ定義関数) では、最小 JAVA_HEAP_SZ は 2048 です。

HP-UX 64 ビットの場合、DB2 は最小ヒープ設定が最大ヒープ設定と等しくなるようハードコーディングします。DB2 for HP-UX (IA64 版) の場合、データベース・マネージャー構成変数 JAVA_HEAP_SZ は少なくとも 4096 に設定してください。

HP-UX 32 ビットで Java ルーチンを実行するには、共用ライブラリー・パスを必ず以下のようにしてください。

```
export SHLIB_PATH=$JAVADIR/jre/lib/PA_RISC:¥
                    $JAVADIR/jre/lib/PA_RISC/classic:¥
                    $HOME/sql1lib/lib:¥
                    /usr/lib:$SHLIB_PATH
```

\$JAVADIR は、通常、/opt/java1.3 (Java SDK for HP-UX 32 ビットのデフォルト・ロケーション) に設定されます。

HP-UX 64 ビットで Java ルーチンを実行するには、以下のコマンドをコマンド行で実行して、db2hplv ツールを使用可能にします。

```
db2hplv -e
```

次のコマンドは、このサポートを使用不可にします。

```
db2hplv -d
```

変更を有効にするために、db2hplv -e または db2hplv -d の実行後に、db2stop および db2start を実行する必要があります。Java ルーチン・サポートは、デフォルトで使用不可になっています。

注: DB2 for HP-UX は、Java ルーチン・サポートが db2hplv -e で使用可能にされている場合は実行されませんが、Java はシステムでアンインストールされます。

HP-UX 64 ビット (PA-RISC 版) では、以下のライブラリーへのシンボリック・リンクは、/usr/lib/pa20_64 に作成する必要があります。そうでない場合は、SQL4301N エラーとなる場合があります。

```
/opt/java1.4/jre/lib/PA_RISC2.0W/libnet.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libzip.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/librmi.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libnio.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libverify.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libmlib_image.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libhprof.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjaas_unix.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libawt.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libcmm.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libdcp.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libdt_socket.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libfontmanager.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libioser12.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libmawt.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjsound.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjava.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjawn.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjcov.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjcp.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjdp.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/libjpeg.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/hotspot/libjsig.sl  
/opt/java1.4/jre/lib/PA_RISC2.0W/hotspot/libjvm.sl
```

リンクが存在しない場合は、以下のコマンドで作成できます。(実行には root 権限が必要です)

```
ln -s /opt/java1.4/jre/lib/PA_RISC2.0W/*.sl /usr/lib/pa20_64  
ln -s /opt/java1.4/jre/lib/PA_RISC2.0W/hotspot/*.sl /usr/lib/pa20_64
```

HP-UX (IA64 版) では、以下のライブラリーへのシンボリック・リンクは、/usr/lib/hpux64 に作成する必要があります。そうでない場合は、SQL4301N エラーとなる場合があります。

```
/opt/java1.4/jre/lib/IA64W/hotspot/libjwind.so  
/opt/java1.4/jre/lib/IA64W/hotspot/libjvm.so  
/opt/java1.4/jre/lib/IA64W/hotspot/libjsig.so  
/opt/java1.4/jre/lib/IA64W/libjdbc0dbc.so  
/opt/java1.4/jre/lib/IA64W/libverify.so  
/opt/java1.4/jre/lib/IA64W/librmi.so  
/opt/java1.4/jre/lib/IA64W/libzip.so
```

```
|
|      /opt/java1.4/jre/lib/IA64W/libawt.so
|      /opt/java1.4/jre/lib/IA64W/libcmm.so
|      /opt/java1.4/jre/lib/IA64W/libmawt.so
|      /opt/java1.4/jre/lib/IA64W/libjava.so
|      /opt/java1.4/jre/lib/IA64W/libjcov.so
|      /opt/java1.4/jre/lib/IA64W/libjcpm.so
|      /opt/java1.4/jre/lib/IA64W/libjdpw.so
|      /opt/java1.4/jre/lib/IA64W/libjpeg.so
|      /opt/java1.4/jre/lib/IA64W/libjsound.so
|      /opt/java1.4/jre/lib/IA64W/libmli_image.so
|      /opt/java1.4/jre/lib/IA64W/libnet.so
|      /opt/java1.4/jre/lib/IA64W/libnio.so
|      /opt/java1.4/jre/lib/IA64W/libjaas_unix.so
|      /opt/java1.4/jre/lib/IA64W/libioserl2.so
|      /opt/java1.4/jre/lib/IA64W/libhprof.so
|      /opt/java1.4/jre/lib/IA64W/libfontmanager.so
|      /opt/java1.4/jre/lib/IA64W/libdt_socket.so
|      /opt/java1.4/jre/lib/IA64W/libdcpr.so
|      /opt/java1.4/jre/lib/IA64W/libjawt.so
|
```

| リンクが存在しない場合は、以下のコマンドで作成できます。(実行には root 権限
| が必要です)

```
|      ln -s /opt/java1.4/jre/lib/IA64W/*.so      /usr/lib/hpux64
|      ln -s /opt/java1.4/jre/lib/IA64W/hotspot/*.so /usr/lib/hpux64
|
```

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 38 ページの『UNIX Java 環境のセットアップ』
- 52 ページの『サンプル・データベースのセットアップ』

関連資料:

- 12 ページの『HP-UX でサポートされる開発ソフトウェア』

Linux Java 環境のセットアップ

| 以下の指示をインプリメントする前に、38 ページの『UNIX Java 環境のセットア
| ヅップ』で説明されているセットアップを実行してください。

手順:

Linux 上で DB2 JDBC サポートを利用して Java アプリケーションを構築するに
は、開発マシンで以下をインストールして構成する必要があります。

1. 以下のいずれかになります。

- 14 ページの『Linux でサポートされる開発ソフトウェア』にリストされてい
る、サポートが提供されるいずれかの開発者キット。
- DB2 Java Enablement。これは Linux クライアントおよびサーバーに対応した
DB2 Universal Database バージョン 8 に装備されています。

Java ストアード・プロシージャまたはユーザー定義関数を実行するには、Linux
ランタイム・リンカーが特定の Java 共用ライブラリーにアクセスできる必要があ
り、しかも DB2 がそのライブラリーと Java 仮想計算機の両方をロードできる必要

があります。そのロードを行うプログラムは、setuid 特権のもとで実行されるので、 /usr/lib 内の従属ライブラリーだけが探索されることになります。

Java 共用ライブラリーを指し示すシンボリック・リンクを /usr/lib に作成します。以下は、リンクしなければならない必須の共用ライブラリーです。構築して実行しているアプリケーションに応じて、さらに別の共用ライブラリーにリンクすることが必要な場合もあります。

IBM Developer Kit 1.3 の場合、libjava.so、libjvm.so、および libhpi.so へのシンボリック・リンクが必要です。シンボリック・リンクを作成するには、root として次のようなコマンドを実行します。

```
cd /usr/lib
ln -fs JAVAHOME/jre/bin/libjava.so .
ln -fs JAVAHOME/jre/bin/classic/libjvm.so .
ln -fs JAVAHOME/jre/bin/libhpi.so .
```

ただし JAVAHOME は、IBM Developer Kit のベース・ディレクトリーです。DB2 によってこのライブラリーが見つけれられない場合に Java ルーチンを実行しようとすると、-4301 エラーが発生し、ライブラリーが見つからないことを知らせるメッセージが管理通知ログ内に置かれます。

注: 別の方法としては、/usr/lib にリンクを作成するのではなく、Java 共用ライブラリーを /etc/ld.so.conf に追加できます。この場合には、/etc/ld.so.conf の変更後に、ldconfig を root で実行する必要があります。そうしないと正常に動作せずに、ルーチンへの呼び出しがハングします (完了しません)。この代替方法は特定のインスタンスでは正常に動作しない可能性があります、ルーチンのハングも生じるかもしれません (完了しません)。この場合には、上述のように /usr/lib ディレクトリーにリンクを作成してください。

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 38 ページの『UNIX Java 環境のセットアップ』
- 52 ページの『サンプル・データベースのセットアップ』

関連資料:

- 14 ページの『Linux でサポートされる開発ソフトウェア』

Solaris Java 環境のセットアップ

以下の指示をインプリメントする前に、38 ページの『UNIX Java 環境のセットアップ』で説明されているセットアップを実行してください。

手順:

Solaris オペレーティング環境で DB2 JDBC サポートを利用して Java アプリケーションを構築するには、開発マシンで次のものをインストールし、構成する必要があります。

1. 20 ページの『Solaris でサポートされる開発ソフトウェア』にリストされている、サポートが提供されるいずれかの開発者キット。
2. DB2 Java Enablement。これは Solaris クライアントおよびサーバーに対応した DB2 Universal Database バージョン 8 に装備されています。

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 38 ページの『UNIX Java 環境のセットアップ』
- 52 ページの『サンプル・データベースのセットアップ』

関連資料:

- 20 ページの『Solaris でサポートされる開発ソフトウェア』

Windows

Windows DB2 CLI のセットアップ情報については、「コール・レベル・インターフェース ガイドおよびリファレンス」を参照してください。

Windows アプリケーション開発環境のセットアップ

Windows NT、Windows 2000、Windows XP、または Windows Server 2003 上に DB2 AD クライアントをインストールすると、環境変数 INCLUDE、LIB、および PATH を使用してインストール・プログラムによって構成レジストリーが更新されます。インストールによって、システム全体の環境変数 DB2INSTANCE が、DB2 という名前で作成されるデフォルト・インスタンスに設定されます。DB2PATH の設定は、DB2 コマンド・ウィンドウを開いて行います。Windows 98 または Windows ME 上に DB2 AD クライアントをインストールすると、autoexec.bat ファイルがインストール・プログラムによって更新されます。

これらの環境変数をオーバーライドして、マシンまたは現在ログオンしているユーザーの値を設定することができます。これらの環境変数の変更は、慎重に行ってください。DB2PATH 環境変数は変更しないでください。DB2INSTANCE はシステム・レベルの環境変数として定義されます。DB2INSTANCE が設定されていないときに使用されるデフォルトのインスタンス名を定義する DB2INSTDEF DB2 レジストリー変数を使用する必要はありません。

手順:

環境変数の設定値をオーバーライドするには、以下のいずれかを使用してください。

- Windows XP のコントロール・パネル
- Windows Server 2003 のコントロール・パネル
- Windows NT のコントロール・パネル
- Windows 2000 のコントロール・パネル
- Windows 98 または Windows ME のコマンド・ウィンドウ

- Windows 98 または Windows ME の autoexec.bat ファイル

コマンド内で変数 %DB2PATH% を使用するときは、 set
LIB="%DB2PATH%\lib";%LIB% のように、絶対パスを引用符で囲んでください。こ
の変数のデフォルト・インストール値は %Program Files%\IBM\SQLLIB ですが、その
中でスペースが使われているため、引用符を使用しないとエラーになることがあり
ます。

さらに、以下に示す特定のステップに従って、 DB2 アプリケーションを実行させ
なければなりません。

- C または C++ プログラムを構築するときは、必ず INCLUDE 環境変数に
%DB2PATH%\INCLUDE が最初のディレクトリとして含まれていなければならませ
ん。

これを行うには、コンパイラ用の以下の環境セットアップ・ファイルを更新し
ます。

Microsoft Visual C++ 6.0

```
"C:\Program Files\Microsoft Visual Studio\VC98\bin\vcvars32.bat"
```

Microsoft Visual C++ .NET

```
"C:\Program Files\Microsoft Visual Studio  
.NET\Tools\vsvars32.bat"
```

これらのファイルには、以下のコマンドが含まれています。

Microsoft Visual C++ 6.0

```
set INCLUDE=%MSVCDir%\ATL\INCLUDE;%MSVCDir%\INCLUDE;  
%MSVCDir%\MFC\INCLUDE;%INCLUDE%
```

Microsoft Visual C++ .NET

```
@set INCLUDE=%MSVCDir%\ATLMFC\INCLUDE;...;  
%FrameworkSDKDir%\include;%INCLUDE%
```

これらのファイルを DB2 で使用するには、まず、%DB2PATH%\INCLUDE パスを設
定する %INCLUDE% を以下のようにリストの末尾から先頭に移動します。

Microsoft Visual C++ 6.0

```
set INCLUDE=%INCLUDE%;%MSVCDir%\ATL\INCLUDE;  
%MSVCDir%\INCLUDE;%MSVCDir%\MFC\INCLUDE
```

Microsoft Visual C++ .NET

```
@set INCLUDE=%INCLUDE%;%MSVCDir%\ATLMFC\INCLUDE;...;  
%FrameworkSDKDir%\include
```

- Micro Focus COBOL プログラムを構築するときは、 COBCPY 環境変数を
%DB2PATH%\INCLUDE\cobl_mf を指すように設定してください。
- IBM COBOL プログラムを構築するときは、 SYSLIB 環境変数を
%DB2PATH%\INCLUDE\cobl_a を指すように設定してください。
- 以下を使用して、必ず LIB 環境変数が %DB2PATH%\lib を指すようにしてくださ
い。

```
set LIB="%DB2PATH%\lib";%LIB%
```


注: 32 ビット環境からの 64 ビット・アプリケーションの相互開発を可能にするには、60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』を参照してください。

- DB2COMM 環境変数を、必ずリモート・データベースのサーバーで設定してください。
- セキュリティー・サービスが、SERVER 認証用のサーバーで、また CLIENT 認証を使用する場合はクライアントで開始されることを確認してください。

注: CLIENT 認証はサーバー・サイドではなくクライアント・サイドで行われるため、クライアント・アプリケーションはユーザーのコンテキストの下で実行されます。Win32 認証 API には特定の特権が必要となりますが、この特権をユーザーが保持している場合もあれば保持していない場合もあります。CLIENT 認証が正常に行われるようにするために、認証要求はクライアント・アプリケーションからセキュリティー・サーバー (デフォルトで特権アカウント・ローカル・システム下で実行され、認証 API の呼び出し権限を持つ) に渡されます。

セキュリティー・サービスを手動で開始するには、NET START DB2NTSECSERVER コマンドを使用します。

通常、セキュリティー・サービスを自動的に開始したいのは、ワークステーションが、クライアント認証用に構成されたサーバーと接続する DB2 クライアントとして動作している場合だけです。セキュリティー・サービスを自動的に開始させるためには、以下のことを実行してください。

Windows NT

1. 「スタート」ボタンをクリックします。
2. 「設定」をクリックします。
3. 「コントロール パネル」をクリックします。
4. 「コントロール パネル」で、「サービス」をクリックします。
5. 「サービス」ウィンドウで、「DB2 セキュリティー・サーバー (DB2 Security Server)」を強調表示します。
6. 設定値として「開始」と「自動」が表示されない場合は、「スタートアップ」をクリックします。
7. 「自動」をクリックします。
8. 「OK」をクリックします。
9. 設定値を有効にするために、マシンをリブートします。

Windows 2000 および Windows Server 2003

1. 「スタート」ボタンをクリックします。
2. Windows 2000 の場合は、「設定」をクリックしてから「コントロール パネル」をクリックします。

Windows Server 2003 の場合は、「コントロール パネル」をクリックします。

3. 「管理ツール」をクリックします。
4. 「サービス」をクリックします。

5. 「サービス」ウィンドウで、「DB2 セキュリティー・サーバー (DB2 Security Server)」を強調表示します。
6. 設定値として「開始」と「自動」が表示されない場合は、トップ・メニューで「操作」をクリックします。
7. 「プロパティ」をクリックします。
8. 「一般 (General)」タブにいることを確認します。
9. 「スタートアップの種類」ドロップダウン・メニューから「自動」を選択します。
10. 「OK」をクリックします。
11. 設定値を有効にするために、マシンをリブートします。

Windows XP

1. 「スタート」ボタンをクリックします。
2. 「設定」をクリックします。
3. 「コントロール パネル」をクリックします。
4. 「パフォーマンスと保守 (Performance and Maintenance)」をクリックします。
5. 「管理ツール」をクリックします。
6. 「サービス」をクリックします。
7. 「サービス」ウィンドウで、「DB2 セキュリティー・サーバー (DB2 Security Server)」を強調表示します。
8. 設定値として「開始」と「自動」が表示されない場合は、トップ・メニューで「操作」をクリックします。
9. 「プロパティ」をクリックします。
10. 「一般 (General)」タブにいることを確認します。
11. 「スタートアップの種類」ドロップダウン・メニューから「自動」を選択します。
12. 「OK」をクリックします。
13. 設定値を有効にするために、マシンをリブートします。

I Windows XP、Windows Server 2003、Windows NT、または Windows 2000 環境におけるデータベース・マネージャーはサービスとしてインプリメントされているため、問題が生じていた場合でも、このサービスの開始時にエラーや警告は戻されません。つまり、db2start または NET START コマンドを実行した場合、いずれかの通信サブシステムが開始できなかったとしても警告が戻されないということです。したがって、ユーザーは必ずイベント・ログまたは DB2 管理通知ログを調べて、これらのコマンドの実行中に起きた可能性のあるエラーを確認しなければなりません。

DB2 CLI または Java を使用する予定の場合、以下の該当するタスクに進んでください。

- Windows CLI 環境のセットアップ
- 49 ページの『Windows Java 環境のセットアップ』

関連タスク:

- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』
- 「コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻」の『Windows CLI 環境のセットアップ』
- 49 ページの『Windows Java 環境のセットアップ』

関連資料:

- 3 ページの『DB2 Application Development Client』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

Windows Java 環境のセットアップ

ここでは、Windows 環境で DB2 Java プログラムを構築して実行するのに必要な情報を示しています。

サポートされる Windows® プラットフォーム上で JDBC プログラムや SQLJ プログラムを DB2® JDBC サポートを利用して実行するために、CLASSPATH は以下のものを組み込むように DB2 のインストール時に自動的に更新されます。

- "." (現行ディレクトリー)
- ファイル `sqllib¥java¥db2java.zip`
- ファイル `sqllib¥java¥db2jcc.jar`
- ファイル `sqllib¥java¥db2jcc_license_cu.jar`

注: `db2jcc_license_cisuz.jar` は、DB2 Connect Personal Edition、DB2 Connect Enterprise Edition、および DB2 ESE の CLASSPATH にも組み込みます。こうすると、DB2 for z/OS and OS/390、DB2 for AS/400 and iSeries、および DB2 for VSE & VM に対する追加の接続が設けられます。

SQLJ プログラムを構築するには、次のファイルを組み込むように CLASSPATH を更新します。

`sqllib¥java¥sqlj.zip`

Java Developer Kit 1.3 または 1.4 を使用してデータ・ソース・プログラムを構築するには、以下を取得してインストールする必要もあります。

JNDI 1.2.1 クラス・ライブラリー (`jndi.jar` および `providerutil.jar`)

<http://java.sun.com/products/jndi/#download>

File System Service Provider 1.2 (`fscontext.jar`)

<http://java.sun.com/products/jndi/#download>

Java Developer Kit 1.3 の場合、さらに以下を取得してインストールする必要があります。

JDBC 2.0 オプション・パッケージ (`jdbc2_0-stdext.jar`)

<http://java.sun.com/products/jdbc/download.html#spec>

注: JDBC 2.0 オプション・パッケージは、Java Developer Kit 1.4 でデータ・ソース・プログラムを構築する場合は必要ありません。

データ・ソース・プログラムには、CLASSPATH の更新 (以下のファイルの組み込み) が必要です。

- jndi.jar
- fscontext.jar
- providerutil.jar

Java Developer Kit 1.3 の場合、さらに CLASSPATH を更新し、次のいずれかを組み込む必要があります。

- jdbc2_0-stdext.jar
- j2ee.jar

注:

1. Java Developer Kit 1.3 を使用しており、j2ee.jar にて CLASSPATH を更新済みの場合、jdbc2_0-stdext.jar は必要ありません。
2. Java Developer Kit 1.4 を使用している場合、jdbc2_0-stdext.jar または j2ee.jar は、CLASSPATH に必要ありません。

データ・ソースのサンプル・プログラムは、sqllib¥samples¥java¥sqlj ディレクトリにあります。詳しくは、sqllib¥samples¥java の README サンプル・ファイルを参照してください。

注:

1. 他のファイルが CLASSPATH に組み込まれている場合は、必ず上記のファイルをまず指定してください。
2. 将来の DB2 のバージョンでは、このディレクトリはなくなるため、sqllib¥java12 ディレクトリへの直接の参照は、すべて除去する必要があります。代わりに、sqllib¥java ディレクトリを参照してください。
3. DB2 Java Enablement には、JDBC パッケージ・バインド・プログラム・ユーティリティである db2jdbcbind が組み込まれています。JDBC パッケージは、DB2 バージョン 8 サーバーに自動的にバインドされます。このユーティリティは、JDBC パッケージを、DB2 バージョン 6 や 7 などの下位レベルのサーバー上で作成するためのものです。
4. Microsoft Software Developer's Kit for Java は DB2 バージョン 8 ではサポートされません。これは、SQLJ のカスタマイズでもタイプ 2 の JDBC アプリケーションでも使用できません。

手順:

Windows オペレーティング・システム上で DB2 JDBC サポートを利用して Java アプリケーションを構築するには、開発マシンで次のものをインストールして構成する必要があります。

1. 23 ページの『Windows でサポートされる開発ソフトウェア』にリストされている、サポートが提供されるいずれかの開発者キット。
2. DB2 Java Enablement。これは Windows クライアントおよびサーバーに対応した DB2 Universal Database バージョン 8 に装備されています。

DB2 Java ルーチン (ストアド・プロシージャと UDF) を実行するには、そのマシンの Java Developer Kit のインストール先のパスを組み込むように、サーバー上の DB2 データベース・マネージャー構成を更新する必要があります。そのためには、サーバーのコマンド行で次のように入力します。

```
db2 update dbm cfg using JDK_PATH c:¥jdk13
```

ただし `c:\jdk13` は、Java Developer Kit のインストール先のパスです。

Java Developer Kit のインストール先のパスに、1 つ以上のスペースを使ったディレクトリー名が入っている場合は、そのパスを単一引用符で囲んでください。たとえば、次のようにします。

```
db2 update dbm cfg using JDK_PATH 'c:\Program Files\jdk13'
```

あるいは、次のような、スペースを使わない短縮形式のディレクトリー名を使用します。

```
db2 update dbm cfg using JDK_PATH c:\progra~1\jdk13
```

次のコマンドをサーバーで入力して、DB2 データベース・マネージャー構成をチェックし、JDK_PATH フィールドの値が正しいことを確認できます。

```
db2 get dbm cfg
```

出力をファイルにリダイレクトすれば、一層容易に表示できます。JDK_PATH フィールドは、出力の先頭近くに表示されます。

IBM Java Developer Kit 用の Java 環境を設定するために、次のコマンドをバッチ・ファイルに入れることができます。バッチ・ファイルは、DB2 コマンド・ウィンドウで実行しなければなりません。すべての必須パスを、使用している特定の環境に合わせて変更してください。サポートされている他の Java Developer Kit にも同様のコマンドを使用できます。

以下は、Sun JDK 1.3.1 環境を設定するバッチ・ファイルの例のコマンドを示しています。

```
set JDKPATH=D:\JAVA\SUNjdk131
set PATH=%JDKPATH%\bin;%PATH%
set CLASSPATH=%CLASSPATH%;%JDKPATH%\lib\jdbc2_0-stdext.jar
db2 update dbm cfg using JDK_PATH %JDKPATH%
db2 terminate
db2stop
db2start
```

バッチ・ファイルは、DB2 コマンド・ウィンドウで実行しなければなりません。

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 52 ページの『サンプル・データベースのセットアップ』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『DB2 Universal JDBC ドライバーのインストール』

関連資料:

- 23 ページの『Windows でサポートされる開発ソフトウェア』

サンプル・データベース

サンプル・データベースのセットアップ

DB2 に付属しているサンプル・プログラムを使用するには、サーバー・ワークステーション上に `sample` データベースを作成する必要があります。このためには、約 23.5 MB のハード・ディスク・スペースが必要になります。空の DB2 データベースには、約 21.5 MB のハード・ディスク・スペースが必要です。

さらに、別バージョンの DB2 を実行しているサーバーや、別のオペレーティング・システム上で稼働しているサーバーにある `sample` データベースに、リモート・クライアントを使用してアクセスする予定であれば、DB2 CLI ユーティリティー・ファイルを含むデータベース・ユーティリティーを `sample` データベースにバインドする必要があります。

手順:

以下に、`sample` データベースをセットアップするためのステップが示されています。

1. 52 ページの『サンプル・データベースの作成』
2. 55 ページの『サンプル・データベースのカatalog』
3. 55 ページの『サンプル・データベース・ユーティリティーのバインディング』

関連タスク:

- 52 ページの『サンプル・データベースの作成』
- 55 ページの『サンプル・データベースのカatalog』
- 55 ページの『サンプル・データベース・ユーティリティーのバインディング』

関連資料:

- 「SQL リファレンス 第 1 巻」の『SAMPLE データベース』
- 「コマンド・リファレンス」の『db2samp1 - サンプル・データベースの作成コマンド』

サンプル・データベースの作成

`sample` データベースをコマンド行で作成するには、`db2samp1` コマンドを使用します。

前提条件:

データベースを作成するには、システム管理者 (SYSADM) またはシステム・コントロール (SYSCTRL) 権限が必要です。SYSADM と SYSCTRL はそれぞれ、DB2 の最上位とその次のレベルの権限です。

手順:

データベースを作成するには、サーバーで以下のことを行ってください。

1. db2samp1 (sample データベースを作成するプログラム) が必ず、ご使用のパスにあるようにします。ファイル db2profile または db2cshrc は、ご使用のパスの db2samp1 に置かれます。変更しない限り、そのパスがファイルのある場所です。

- UNIX サーバーでは、db2samp1 は次の場所にあります。

`$HOME/sql1lib/bin`

ここで \$HOME は、DB2 インスタンス所有者のホーム・ディレクトリーです。

- Windows では、db2samp1 は次の場所にあります。

`%DB2PATH%\bin`

ここで %DB2PATH% は、DB2 がインストールされているパスです。

2. DB2INSTANCE 環境変数が、sample データベースを作成するインスタンスの名前に設定されていることを確認してください。この変数が設定されていない場合は、以下に示すコマンドで設定することが可能です。

- UNIX の場合:

bash または Korn シェルでは次のように入力します。

```
DB2INSTANCE=instance_name
export DB2INSTANCE
```

C シェルでは次のように入力します。

```
setenv DB2INSTANCE instance_name
```

- Windows の場合は、次のように入力します。

```
set DB2INSTANCE=instance_name
```

ここで instance_name は、データベース・インスタンスの名前です。

3. db2samp1 と、それに続いて、サンプル・データベースを作成したい場所を入力することによって、sample データベースを作成します。UNIX プラットフォームでは、それはたとえば \$HOME のようなパスであり、次のように入力します。

```
db2samp1 path
```

たとえば、次のようにします。

```
db2samp1 $HOME
```

Windows では、それはたとえば C: のようなドライブであり、次のように入力します。

```
db2samp1 drive
```

たとえば、次のようにします。

```
db2samp1 C:
```

パスまたはドライブを指定しないと、インストール・プログラムは、データベース・マネージャー構成ファイルの DFTDBPATH パラメーターによって指定されているデフォルトのパスまたはドライブに、サンプル表をインストールします。データベースの認証タイプは、データベースが作成されるインスタンスの認証タイプと同じです。

関連タスク:

- 54 ページの『ホスト・サーバーまたは AS/400 および iSeries サーバーでのサンプル・データベースの作成』
- 55 ページの『サンプル・データベースのカタログ』
- 55 ページの『サンプル・データベース・ユーティリティのバインディング』

ホスト・サーバーまたは AS/400 および iSeries サーバーでのサンプル・データベースの作成

DB2 UDB (OS/390 および z/OS 版) などのホスト・サーバー、あるいは AS/400 および iSeries サーバーに対してサンプル・プログラムを実行したい場合は、SQL の解説書に説明されているサンプル表をもったデータベースを作成する必要があります。

注: ホスト・サーバーに接続するには、DB2 Connect が必要です。

制約事項:

ワークステーション上とホスト・システム上のどちらに DB2 があるかによって、SQL 構文と DB2 コマンドに多少の違いがあります。DB2 UDB (OS/390 および z/OS 版) または DB2 for AS/400 and iSeries 上のデータベースにアクセスする場合、それらのデータベース・システムでサポートされている SQL ステートメントとプリコンパイル/ BIND オプションを必ずプログラムが使用するようになっています。

手順:

データベースを作成するには、以下の手順で行います。

1. db2samp1 を使用して、DB2 ワークステーション・サーバー・インスタンスで sample データベースを作成します。
2. sample データベースに接続します。
3. サンプル表データをファイルにエクスポートします。
4. ホスト・データベースに接続します。
5. サンプル表を作成します。
6. ワークステーション・サーバー上のデータのエクスポート先のファイルからサンプル表データをインポートします。

関連概念:

- 「データ移動ユーティリティ ガイドおよびリファレンス」の『エクスポートの概要』
- 「データ移動ユーティリティ ガイドおよびリファレンス」の『インポートの概要』

関連タスク:

- 55 ページの『サンプル・データベースのカタログ』
- 55 ページの『サンプル・データベース・ユーティリティのバインディング』

関連サンプル:

- 『expsamp.sqb -- Export and import tables with table data to a DRDA database (IBM COBOL)』
- 『tbmove.sqc -- How to move table data (C)』
- 『tbmove.sqC -- How to move table data (C++)』

サンプル・データベースのカatalog

サーバー上の `sample` データベースにリモート・クライアントからアクセスするには、クライアント・ワークステーションで `sample` データベースをカatalogする必要があります。

サーバー・ワークステーションでは、`sample` データベースをカatalogする必要がありません。データベースの作成時にサーバー上でカatalogされているからです。

カatalogを作成すると、クライアント・ワークステーションのデータベース・ディレクトリーが、クライアント・アプリケーションのアクセスしたいデータベース名に更新されます。クライアントの要求を処理するときに、データベース・マネージャーはカatalogされた名前を使用して、データベースを見つけて接続します。

手順:

リモート・クライアント・ワークステーションでサンプル・データベースをカatalogするには、次のように入力します。

```
db2 catalog database sample as sample at node nodename
```

ここで `nodename` は、サーバー・ノードの名前です。

リモート・ノードもカatalogしてからでないと、データベースには接続できません。

関連タスク:

- 「インストールおよび構成 補足」の『DB2 クライアントからの TCP/IP ノードのカatalog』
- 「インストールおよび構成 補足」の『CLP による DB2 クライアントからのデータベースのカatalog』
- 「インストールおよび構成 補足」の『DB2 クライアントからの NetBIOS ノードのカatalog』
- 「インストールおよび構成 補足」の『クライアントからの名前付きパイプ・ノードのカatalog』
- 55 ページの『サンプル・データベース・ユーティリティーのバインディング』

サンプル・データベース・ユーティリティーのバインディング

別のバージョンの DB2 を実行しているリモート・クライアントから、サーバー上の `sample` データベースにアクセスする予定であれば、DB2 CLI ユーティリティーなどのデータベース・ユーティリティーを、`sample` データベースにバインドする必要があります。

アプリケーションの実行時にデータベースにアクセスするのにデータベース・マネージャが必要とするパッケージが、このバインドによって作成されます。バインドは、プリコンパイル時に作成されるバインド・ファイルに対して BIND コマンドを実行することによって、明示的に行うことができます。

手順:

使用するクライアント・ワークステーションのプラットフォームに応じて、データベース・ユーティリティは異なる仕方ではバインドします。

UNIX クライアント・ワークステーションの場合:

1. 次のように入力して、sample データベースに接続します。

```
db2 connect to sample user userid using password
```

ここで、*userid* と *password* は、sample データベースが置かれているインスタンスのユーザー ID とパスワードを表します。

2. 次のように入力して、データベースにユーティリティをバインドします。

```
db2 bind BNDPATH/@db2ubind.lst blocking all sqlerror continue ¥  
messages bind.msg grant public
```

```
db2 bind BNDPATH/@db2cli.lst blocking all sqlerror continue ¥  
messages cli.msg grant public
```

ここで、*BNDPATH* はバインド・ファイルが置かれているパスです。たとえば、*\$HOME/sqlllib/bnd* (*\$HOME* は DB2 インスタンス所有者のホーム・ディレクトリ)。

3. バインドが成功したかどうかを、バインド・メッセージ・ファイル *bind.msg* および *cli.msg* を調べて確認します。

Windows オペレーティング・システムが実行されるクライアント・ワークステーションの場合:

1. 「スタート」メニューから「プログラム」を選択します。
2. 「プログラム」(XP では「すべてのプログラム」)メニューで IBM DB2 を選択します。
3. 「IBM DB2」メニューから、「コマンド行ツール (Command Line Tools)」を選択します。
4. 「コマンド行ツール (Command Line Tools)」メニューから、「DB2 コマンド・ウィンドウ (DB2 Command Window)」を選択します。

コマンド・ウィンドウが表示されます。

5. 次のように入力して、sample データベースに接続します。

```
db2 connect to sample user userid using password
```

ここで、*userid* と *password* は、sample データベースが置かれているインスタンスのユーザー ID とパスワードを表します。

6. 次のように入力して、データベースにユーティリティをバインドします。


```
db2 bind "%DB2PATH%\bnd\@db2ubind.lst" blocking all
sqlerror continue messages bind.msg grant public
```

```
db2 bind "%DB2PATH%\bnd\@db2cli.lst" blocking all
sqlerror continue messages cli.msg grant public
```

ここで %DB2PATH% は、DB2 のインストール先を示すパスです。

7. コマンド・ウィンドウを終了し、バインド・メッセージ・ファイル bind.msg および cli.msg を調べて、バインドが成功したかどうかを確認します。

ホスト・サーバーにアクセスするすべてのクライアントの場合、db2ubind.lst ではなく以下の .lst ファイルのうちの 1 つを指定します。

ddcsmvs.lst

DB2 for OS/390 and z/OS の場合

ddcsvm.lst

DB2 (VM 版) の場合

ddcsvse.lst

DB2 (VSE 版) の場合

ddcs400.lst

DB2 for AS/400 and iSeries の場合

たとえば:

- UNIX クライアントから DB2 for OS/390 and z/OS サーバーにアクセスするには、以下を入力します。

```
db2 bind BNDPATH/@ddcsmvs.lst blocking all sqlerror continue ¥
messages bind.msg grant public
```

- Windows クライアントから DB2 for OS/390 and z/OS サーバーにアクセスするには、以下を入力します。

```
db2 bind "%DB2PATH%\bnd\@ddcsmvs.lst" blocking all
sqlerror continue messages bind.msg grant public
```

関連タスク:

- 52 ページの『サンプル・データベースの作成』
- 54 ページの『ホスト・サーバーまたは AS/400 および iSeries サーバーでのサンプル・データベースの作成』
- 55 ページの『サンプル・データベースのカatalog』

関連資料:

- 「コマンド・リファレンス」の『BIND コマンド』

アプリケーションの移行

アプリケーションの DB2 バージョン 8 への移行

DB2® バージョン 8 では以下の DB2 バージョンの移行がサポートされます。

- DB2 バージョン 6
- DB2 バージョン 7.1

- DB2 バージョン 7.2
- DataJoiner[®] バージョン 2.1.1

上記より後のバージョンの DB2 に移行すると、データベースとノードのディレクトリは自動的に移行されます。上記以外の旧バージョンの DB2 から移行するには、上記のサポート対象バージョンのいずれかにまず移行してから、そのバージョンから DB2 バージョン 8 に移行する必要があります。

HP-UX

HP-UX バージョン 10 以前から HP-UX バージョン 11 へ DB2 を移行する場合、ご使用の DB2 プログラムを HP-UX バージョン 11 (組み込み SQL がある場合) 上の DB2 で再プリコンパイルして、再コンパイルしなければなりません。これには、すべての DB2 アプリケーション、ストアード・プロシージャ、ユーザー定義関数、およびユーザー出口プログラムが含まれます。さらに、HP-UX バージョン 11 上でコンパイルされた DB2 プログラムは、HP-UX バージョン 10 以前の上では実行できません。HP-UX バージョン 10 上でコンパイルされ実行される DB2 プログラムは、HP-UX バージョン 11 サーバーにリモートで接続することができます。

Micro Focus COBOL

DB2 バージョン 2.1.1 またはそれ以前を使用してプリコンパイルし、Micro Focus COBOL を使用してコンパイルした既存アプリケーションは、現行バージョンの DB2 を使用して再プリコンパイルされた後、Micro Focus COBOL を使用して再コンパイルされる必要があります。IBM[®] プリコンパイラーの旧バージョンを使用して構築したアプリケーションを再プリコンパイルしないと、異常終了時にデータベースが破壊される恐れがあります。

関連概念:

- 「DB2 Universal Database サーバー機能 概説およびインストール」の『移行に関する推奨事項』
- 59 ページの『Java アプリケーション、ルーチン、およびアプレットの移行』
- 64 ページの『2 つのバージョンの DB2 でのアプリケーションの実行』

関連タスク:

- 「DB2 Universal Database サーバー機能 概説およびインストール」の『データベースの移行』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『インスタンスの移行 (UNIX)』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 UDB の移行 (Windows)』
- 「DB2 Universal Database サーバー機能 概説およびインストール」の『DB2 UDB サーバーの移行 (UNIX)』
- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』
- 63 ページの『アプリケーション移植性の確保』

関連資料:

- 「管理 API リファレンス」の『管理 API および アプリケーションの移行』

- 「DB2 Universal Database サーバー機能 概説およびインストール」の『移行の注意点』
- 「管理ガイド: プランニング」の『バージョン 8 と以前のリリースとの非互換性』

Java アプリケーション、ルーチン、およびアプレットの移行

SQLJ アプリケーションおよびルーチン

DB2® バージョン 8 では、SQLJ はプラットフォームに依存しない新しいアーキテクチャーに基づいています。このアーキテクチャーにより、DB2 がサポートするどのオペレーティング・システムでも、カスタマイズされた SQLJ アプリケーションおよびルーチンのパフォーマンスと移植性が大幅に向上します。DB2 バージョン 7 の SQLJ サポートからいくつかの変更が行われました。この変更に伴って、SQLJ アプリケーションおよびルーチンの変更がいくつか必要になる場合があります。また、DB2 バージョン 8 に移行する SQLJ アプリケーションおよびルーチンの再変換と再カスタマイズが必要になります。

バージョン 7 の既存の SQLJ アプリケーションおよびルーチンをバージョン 8 に移行するには、以下のステップを実行する必要があります。

1. すべての VALUES ステートメントをダミー選択に変更します。たとえば次のようにします。

```
#sql [ctxt] hv = {VALUES (DUMMY(1))}
#sql [ctxt] {SELECT DUMMY(1) INTO :hv FROM SYSIBM.SYSDUMMY1}
```

注: DB2 バージョン 8 では、SQLJ での VALUES ステートメントおよびコンパウンド SQL はサポートされなくなりました。

2. すべての BLOCK ステートメントを除去します。個別の SQL ステートメントに変更します。ブロック化されたステートメントが実行可能 SQL ステートメントと COMMIT の 2 つのみだった場合は、接続の際の自動コミットをオンにすることにより単一ステートメントを使用できます。BLOCK ステートメントの使用を SQLJ バッチ API 呼び出しに置き換えることによってブロック化を行うようにすることもできます。
3. バージョン 8 の SQLJ ユーティリティーで **sqlj** コマンドを使用してアプリケーションおよびルーチンを再変換し、それらのアプリケーションおよびルーチンを **db2sqljcustomize** コマンドを使用して再カスタマイズします。バージョン 8 が備えている新規クロスプラットフォーム・サポートでバイナリを実行できるようにするには、ソース・コードに変更がない場合でも、このステップを必ず実行する必要があります。

Java™ アプレット

これまで "net" ドライバーと呼ばれていたタイプ 3 JDBC ドライバーは、今後は使用すべきではありません。DB2 Java アプレットは、DB2 Universal JDBC ドライバーに移行する必要があります。このドライバーには Type 4 接続が含まれています。新規の DB2 Universal JDBC ドライバーを使用するためにタイプ 3 JDBC アプレットを変換するには、次のような変更を行います。

1. DB2 Universal JDBC ドライバーのアーカイブは db2jcc.jar です。このアプレットに関連した .html ファイル内で、アーカイブを db2java.zip から db2jcc.jar に変更します。db2jcc.jar を Web サーバーにコピーします。
2. DB2 Universal JDBC ドライバーのクラス名は com.ibm.db2.jcc.DB2Driver です。アプレット .java ファイル中のタイプ 3 JDBC ドライバーのクラス名 COM.ibm.db2.jdbc.net.DB2Driver を DB2 Universal JDBC ドライバーのクラス名に変更します。アプレットが javax.sql.DataSource を使用して接続を確立する場合は、この JDBC ドライバー・クラスへの参照がない場合があります。
3. タイプ 3 と DB2 Universal JDBC のどちらのドライバーも、jdbc:db2://server:portnumber/dbname という同じ形式のデータ・ソース URL を使用します。ただし、server、portnumber、および dbname の 3 つの部分は、ドライバーによって意味が異なります。

タイプ 3 JDBC ドライバーは、クライアント (アプレットを実行するブラウザ)、JDBC アプレット・サーバー、および DB2 サーバーから成る 3 層モデルです。URL 内の server と portnumber は JDBC アプレット・サーバーを指します。dbname は、JDBC アプレット・サーバーを実行するシステムでカタログされたデータベース別名です。

DB2 Universal JDBC ドライバー・クライアントは DB2 サーバーに直接接続するので、server と portnumber は DB2 サーバーの TCP/IP Listener のものを指します。dbname は、DB2 サーバー・システムでカタログされたデータベース別名です。

アプレットが DriverManager.getConnection を使用して DB2 に接続する場合、.java ファイルと (必要な場合) .html ファイルを、DB2 Universal JDBC ドライバー用の別の URL に更新します。

4. アプレットが COM.ibm.db2.jdbc.DB2DataSource を使用する場合、クラス com.ibm.jcc.db2.DB2SimpleDataSource の新規の javax.sql.DataSource オブジェクトを作成する必要があります。その新規のクラスを使用するには、アプレットを更新する必要があります。

関連概念:

- 57 ページの『アプリケーションの DB2 バージョン 8 への移行』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 137 ページの『SQLJ アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』
- 134 ページの『SQLJ プログラムの構築』

32 ビット環境から 64 ビット環境へのアプリケーションの移行

Windows 32 ビット・アプリケーションは、64 ビット環境用の変更なしで現状のまま Windows 64 ビット上で実行できます。UNIX の場合、Linux for IA64 および Linux for zSeries を除くすべての 64 ビット DB2 インスタンスでは、アプリケーションを再バインドして適切なライブラリー・パスを設定して実行することで、既存の 32 ビット・ローカル・アプリケーションを移行することができます。HP-UX の場合、これが可能なのは、アプリケーションが +s オプションを指定して

リンクされていた場合のみです。アプリケーションのリンクに +s オプションが指定されていなかった場合は、+s オプションを指定するか、組み込みランタイム・パスを使用して 32 ビット DB2 ライブラリー (下記参照) を組み込んで、アプリケーションを再構築する必要があります。

手順:

UNIX の場合、 64 ビット環境での 32 ビット・アプリケーション用の正しいライブラリー・パスは、lib32 です。環境変数の設定を lib32 に変更すると、インスタンス環境 (32 ビットおよび 64 ビット) でのすべてのアプリケーションに影響するため、この変更は多くの場合、好ましいものではありません。これを回避するには、ラッパー・スクリプトを使用してアプリケーションの環境変数を設定し、アプリケーションの実行もそのスクリプトから行います。

以下は、その場合に使用するラッパー・スクリプトの例です。

```
#!/bin/sh
echo <ENV_VAR_SETTING>
export <ENV_VAR_SETTING>
rm -rf $HOME/sqlllib/db2dump/* > /dev/null 2>&1
echo
echo Running application...
$1
echo ...Done running application.
```

ここで <ENV_VAR_SETTING> はプラットフォームに応じた環境変数の設定で、以下のようになります。

AIX: LIBPATH=\$HOME/sqlllib/lib32:\$LIBPATH

HP-UX (以下のいずれか):

SHLIB_PATH=\$HOME/sqlllib/lib32:\$SHLIB_PATH

LD_LIBRARY_PATH=\$HOME/sqlllib/lib32:\$LD_LIBRARY_PATH

注: HP-UX の場合、ラッパーを使用できるのは、アプリケーションが +s オプションを指定してリンクされていた場合のみです。

Linux: LD_LIBRARY_PATH=\$HOME/sqlllib/lib32:\$LD_LIBRARY_PATH

Solaris:

LD_LIBRARY_PATH=\$HOME/sqlllib/lib32:\$LD_LIBRARY_PATH

アプリケーションの再バインド後、次のようにコマンド行にラッパー・スクリプト名、実行可能ファイル名の順に入力して、このラッパー・プログラムを実行します。

```
<wrapper_script> <executable>
```

ラッパー内で環境変数を変更しても、(C system() 呼び出しなどで) 他の実行可能ファイルを読み出すアプリケーションでは、ラッパー・スクリプトのライブラリー・パスと読み出される実行可能ファイルに互換性がなければ、機能しないことがあります。こうしたアプリケーションを移行するには、オブジェクト・ファイルを再リンクして、アプリケーションを再バインドしなければなりません。

オブジェクト・ファイルをリンクするには、プラットフォームの環境変数ではなく、lib32 を指定したランタイム・ライブラリー・パスを使用する必要があります。

す。サンプル・プログラムの C、C++、および CLI ビルド・スクリプトでは、新規アプリケーションを 64 ビット環境に容易に移植できるように、適切なランタイム・パスが使われています。

64 ビット環境で既存の 32 ビット・アプリケーションのオブジェクト・ファイルをリンクするときにも、同じリンク・オプションを使用する必要があります (下記のサンプル関連リンクのビルド・スクリプトを参照)。ランタイム・ライブラリー・パスに 32 ビット DB2 ライブラリーを組み込むには、以下のフラグを使用します。

AIX: -L\$DB2PATH/lib32

別の方法として、-blibpath リンカー・オプションを使用して、完全ランタイム・ライブラリー・パスを指定することもできます。AIX サンプル・ビルド・スクリプトでは、前者の方式が使われています。

HP-UX:

-Wl,+b\$DB2PATH/lib32

Linux: -Wl,-rpath,\$DB2PATH/lib32

Solaris:

-R\$DB2PATH/lib32

注:

1. これらのコマンドはそれぞれ、ld を直接指定するリンクではなく、コンパイラーを使用してリンクすることを前提としています。
2. Solaris では、LD_LIBRARY_PATH と LD_LIBRARY_PATH_32 を設定解除してから、ランタイム・パスを使用してアプリケーションをリンクする必要があります。これを行わないと、ランタイム・パスの設定ではなく、LD_LIBRARY_PATH または LD_LIBRARY_PATH_32 の設定が使用されます。
3. Linux では、--enable-new-dtags リンク・オプションを使用する場合は、LD_LIBRARY_PATH を設定解除してから 32 ビット実行可能ファイルを実行してください。これを行わないと、ランタイム・パスの設定ではなく、LD_LIBRARY_PATH の設定が使用されます。

Windows 上でのクロス開発

Windows 32 ビット環境で 64 ビット・アプリケーションを開発するには、LIB 環境変数が %DB2PATH%\lib\Win64 を指すようにします。LIB パスにデフォルトで %DB2PATH%\lib (32 ビット・パス) が追加されるため、32 ビット環境で 64 ビット・アプリケーションをクロス開発するときは、デフォルトの 32 ビット・パスより前に %DB2PATH%\lib\Win64 パスがあるようにしてください。%DB2PATH%\lib は、32ビット環境で 32 ビット・アプリケーションを開発する場合、または 64 ビット環境で 64 ビット・アプリケーションを開発する場合に使用します。

長いデータ・タイプの変更

32 ビット・サーバーでの実行を続けながら、64 ビット・オペレーティング環境でも使用できるように 32 ビット・アプリケーションを移行したい場合、LONGERROR プリコンパイル・オプションを使用してアプリケーションの移植の準備をします。32 ビット環境で LONGERROR を YES に設定して、長いタイプのホスト変数の検出のたびにプリコンパイラからエラーが戻されるようにします。その後、以下のステップを行います。

1. 長いタイプが必要でない限り、ホスト変数での長いタイプの使用を控えます。その代わりに、`sqlint32` や `sqluint32` などの新規の移植可能ホスト変数を使用します。たとえば次のようにします。

```
EXEC SQL BEGIN DECLARE SECTION;
long y;      /* this declaration generates an error on 64 bit */
sqlint32 x; /* this declaration is acceptable for 64 bit */
EXEC SQL END DECLARE SECTION;
```

2. 64 ビット・サーバー上のデータベースに対してアプリケーションをプリコンパイルします。これで、移植しようとしているアプリケーション用の新しいパッケージが作成されます。
3. 64 ビット・モードでアプリケーションをコンパイルします。
4. アプリケーションを新規の 64 ビット DB2 ライブラリーにリンクします。
5. 64 ビット・サーバー上のデータベースに対してアプリケーションをバインドします。

注: DB2 は 32 ビット・インスタンスでの 64 ビット・アプリケーションの実行をサポートしていません。

関連概念:

- 37 ページの『UNIX 環境変数の設定』
- 57 ページの『アプリケーションの DB2 バージョン 8 への移行』

関連サンプル:

- 『bldapp -- Builds AIX C application programs (C)』
- 『bldapp -- Builds HP-UX C applications (C)』
- 『bldapp -- Builds Linux C applications (C)』
- 『bldapp -- Builds Solaris C applications (C)』

アプリケーション移植性の確保

以下に、アプリケーションを開発するときに気をつける必要のある点を示します。これらの点を参考にすれば、アプリケーションの移植化に役に立ちます。

手順:

- UNIX では、デフォルトのライブラリー検索パス `/usr/lib:/lib` だけをアプリケーションで使用してください。Windows® オペレーティング・システムでは、以下を使用して `LIB` 環境変数が `%DB2PATH%\lib` を指していることを確認してください。

```
set LIB=%DB2PATH%\lib;%LIB%
```

また、使用している DB2 のデフォルトのパスとバージョンとの間にシンボリック・リンクを作成します。そのリンク先が、アプリケーションが必要とする DB2 の最低レベルであることを確認してください。リンクの設定については、ご使用のプラットフォーム用の「概説およびインストール」またはインストール・トピックを参照してください。

- アプリケーションが特定のバージョンの DB2 を必要とする場合、アプリケーション中で DB2 バージョンを指定するパスを使用してください。たとえば、AIX®

アプリケーションが DB2 バージョン 5 を必要とするなら、
/usr/lpp/db2_05_00/lib を使用します。普通は、これを行う必要はありません。

- 内部開発ではなく、実動のためのアプリケーションを構築しているとき、通常、アプリケーション中のパスは、UNIX 上の sqllib/lib ディレクトリー、Windows オペレーティング・システム上の sqllib¥lib ディレクトリーのインスタンス所有者のコピーを指さないようにしてください。このようにすると、アプリケーションは特定のユーザー名と環境にかなり依存するようになります。
- 一般的に、一部の環境では、検索パスを変更するのに次に示す環境変数を使用しないでください。LIBPATH (AIX)、SHLIB_PATH (HP-UX 32 ビット)、LD_LIBRARY_PATH (HP-UX 64 ビット、Linux、および Solaris)、および LIB (Windows)。これらの変数は、環境内で実行するアプリケーションに指定されている検索パスをオーバーライドするため、アプリケーションが、必要とするライブラリーまたはファイルを検出できなくなる可能性があります。
- DB2 Universal Database™ バージョン 6、7、および 8 では、ストリング・セマンティクスのあるすべての文字配列項目は、unsigned char などの他のバリエーションの代わりにタイプ char をもっています。DB2 Universal Database バージョン 6、バージョン 7、またはバージョン 8 でコーディングするアプリケーションはすべて、この方式に従ってください。

unsigned char を使う DB2 バージョン 1 アプリケーションの場合、バージョン 1 アプリケーションの unsigned char と、バージョン 6、バージョン 7、またはバージョン 8 の関数プロトタイプの char との間でタイプの衝突が起きるため、コンパイラーが警告またはエラーを生成する可能性があります。これが起きた場合、コンパイラー・オプション -DSQLOLDCHAR を使用して問題を排除してください。

関連概念:

- 37 ページの『UNIX 環境変数の設定』
- 57 ページの『アプリケーションの DB2 バージョン 8 への移行』

関連タスク:

- 45 ページの『Windows アプリケーション開発環境のセットアップ』
- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』

2 つのバージョンの DB2 でのアプリケーションの実行

UNIX® プラットフォームの場合に、旧バージョンの DB2® のアプリケーションが存在していて、しかもそれを旧バージョンのデータベース・インスタンスと、同じマシン上の DB2 バージョン 8 のインスタンスのどちらでも実行したければ、環境に若干の変更を加える必要がある場合があります。どのような変更を行うかを判別するには、以下の質問に答え、『条件』の節を検討して、現状に当てはまる条件があるかどうかを調べてください。

挙げられた点を説明するために、AIX® システムが使われています。同じ概念が他の UNIX プラットフォームにも当てはまりますが、詳細な点 (環境変数と特定のコマンドなど) は異なる可能性があります。

質問

質問 1: どのように以前のバージョンのアプリケーションを、 DB2 クライアントのランタイム・ライブラリー (AIX 上の libdb2.a など) にリンクしましたか？

実行可能ファイルの組み込み共用ライブラリーの探索パスを判別するには、実行可能ファイルが置かれているディレクトリー (/usr/bin またはインスタンス・ディレクトリーなど) 内の以下のようなシステム・コマンドの 1 つを使用します。

AIX dump -H *executable_filename*

HP-UX

 chatr *executable_filename*

Linux objdump -p *executable_filename*

Solaris

 dump -Lv *executable_filename*

executable_filename はアプリケーションの実行可能ファイルの名前です。

以下に、AIX の C サンプル・アプリケーション dbcat 用の DB2 バージョン 7.2 のサンプル・ダンプ・リストを示します。これは、DB2 インスタンスのサンプル・サブディレクトリー /home/dbinst/samples/c 内に取り込まれたものです。

C アプリケーション dbcat のサンプル・ダンプ・リスト			

dbcat:			
Loader Section			
Loader Header Information			
VERSION#	#SYMtableENT	#RELOCent	LENidSTR
0x00000001	0x0000000f	0x00000015	0x00000047
#IMPfilID	OFFidSTR	LENstrTBL	OFFstrTBL
0x00000003	0x00000284	0x0000007f	0x000002cb
Import File Strings			
INDEX	PATH	BASE	MEMBER
0	/home/db2inst/sql1lib/lib:/usr/lib:/lib		
1		libc.a	shr.o
2		libdb2.a	shr.o

ライン 0 (ゼロ) は、リンクされている共用ライブラリーを検出するために、実行可能を検索するディレクトリー・パスを示します。ライン 1 および 2 は、アプリケーションがリンクされている共用ライブラリーを示しています。

アプリケーションが構築された方法によって、以下のパスを検索できます。
/usr/lpp/db2_07_01_0000/lib、 INSTHOME/sql1lib/lib (INSTHOME はデータベース・インスタンス所有者のホーム・ディレクトリー)、または /usr/lib:/lib の組み合わせ。

質問 2: どのような方法でシステムの DB2 ランタイム・ライブラリーを構成しましたか？

DB2 バージョン 1、2、5、6、7、または 8 がインストールされている場合、DB2 クライアント・ランタイム・ライブラリーを含むシステム・デフォルト共用ライブラリー・パス /usr/lib から、DB2 クライアント・ランタイム・ライブラリーを含む DB2 インストール・パスへのシンボリック・リンクを作成するステップ (オプション) があります。

AIX での DB2 の各バージョンのインストール・パスは、以下のとおりです。

バージョン 1

/usr/lpp/db2_01_01_0000/lib

バージョン 2

/usr/lpp/db2_02_01/lib

バージョン 5

/usr/lpp/db2_05_00/lib

バージョン 6.1

/usr/lpp/db2_06_01/lib

バージョン 7

/usr/lpp/db2_07_01/lib

バージョン 8

/usr/opt/db2_08_01/lib

いずれの場合も、ランタイム共用ライブラリーの名前は、libdb2.a です。

これらのライブラリーのバージョンは、一度に 1 つしかデフォルトになりません。DB2 がこのデフォルトを提供するので、アプリケーションを構築するときに DB2 の特定のバージョンに依存することはありません。

質問 3: 環境の中でさまざまな検索パスを指定しますか？

LIBPATH 環境変数 (AIX 上)、SHLIB_PATH (HP-UX 32 ビット上)、SHLIB_PATH または LD_LIBRARY_PATH (HP-UX 64 ビット上)、または LD_LIBRARY_PATH (Linux および Solaris 上) を使用して、アプリケーションでコーディングされている共用ライブラリーの検索パスをオーバーライドすることができます。Solaris の場合、32 ビット・アプリケーションには LD_LIBRARY_PATH_32 を、64 ビット・アプリケーションには LD_LIBRARY_PATH_64 を使用することもできます。ライブラリー検索パスは、質問 1 の答えの中で与えられたプラットフォーム用の適切なシステム・コマンドを使用して、調べることができます。

条件

前述の質問に答えたならば、環境の変更が必要な場合があります。下記に挙げた条件をお読みください。いずれかの条件が現状に当てはまるならば、変更が必要です。

条件 1: バージョン 7 アプリケーションが、AIX のデフォルト共用ライブラリー・パス `/usr/lib/libdb2.a` 以外の共用ライブラリーをロードし、以下の条件が当てはまる場合。

- `/usr/lib/libdb2.a` から `/usr/lpp/db2_07_01/lib/libdb2.a` へのシンボリック・リンクがあり、データベース・サーバーが DB2 Universal Database (AIX 版) バージョン 8 の場合、以下のどれかを行います。

- 以下のものを示すシンボリック・リンクを変更します。

`/usr/opt/db2_08_01/lib/libdb2.a`

`root` で、次のように `"db2ln"` コマンドを使用してリンクを変更できます。

`/usr/opt/db2_08_01/cfg/db2ln`

- `LIBPATH` 環境変数が `/usr/opt/db2_08_01/lib` または `INSTHOME/sql/lib/lib` を指すように設定します。 `INSTHOME` はバージョン 8 DB2 インスタンス所有者のホーム・ディレクトリーです。
- アプリケーション (クライアント) からサーバー・インスタンスへの TCP/IP 接続を構成します。
- `/usr/lib/libdb2.a` から `/usr/opt/db2_08_01/lib/libdb2.a` へのシンボリック・リンクがあり、データベース・サーバーが DB2 バージョン 7 の場合、アプリケーション (クライアント) インスタンスからサーバー・インスタンスへの TCP/IP 接続を構成します。

条件 2: バージョン 7 アプリケーションが、DB2 バージョン 7 インスタンス所有者 (`$HOME/sql/lib/libdb2.a`) の `$HOME` パス以外の共用ライブラリーをロードし、データベース・サーバーが DB2 Universal Database™ (AIX 版) バージョン 8 の場合、以下のどれかを行います。

- データベース・サーバー・インスタンスとして、アプリケーション・インスタンスを同じバージョンへ移行します。
- `LIBPATH` 環境変数が `/usr/opt/db2_08_01/lib` または `INSTHOME/sql/lib/lib` を示すように設定します。 `INSTHOME` はバージョン 8 インスタンス所有者のホーム・ディレクトリーです。
- アプリケーション (クライアント) からサーバー・インスタンスへの TCP/IP 接続を構成します。

条件 3: バージョン 7 アプリケーションが、DB2 バージョン 7 インストール・パス (`/usr/lpp/db2_07_01/lib/libdb2.a`) 以外の共用ライブラリーをロードし、データベース・サーバーが DB2 Universal Database (AIX 版) バージョン 8 の場合、以下のどちらかを行います。

- `LIBPATH` 環境変数が `/usr/opt/db2_08_01/lib` または `INSTHOME/sql/lib/lib` を示すように設定します。 `INSTHOME` はデータベース・インスタンス所有者のホーム・ディレクトリーです。
- アプリケーション (クライアント) からサーバー・インスタンスへの TCP/IP 接続を構成します。

条件 4: バージョン 7 アプリケーションが、DB2 Universal Database (AIX 版) バージョン 8 のインストール・パス (`/usr/opt/db2_08_01/lib/libdb2.a`) 以外の共用

ライブラリーをロードし、データベース・サーバーが DB2 バージョン 7 の場合、アプリケーション (クライアント) インスタンスからサーバー・インスタンスへの TCP/IP 接続を構成します。

関連概念:

- 37 ページの『UNIX 環境変数の設定』
- 5 ページの『データベース・マネージャー・インスタンス』
- 57 ページの『アプリケーションの DB2 バージョン 8 への移行』

関連タスク:

- 45 ページの『Windows アプリケーション開発環境のセットアップ』

次に行うこと

環境をセットアップしたなら、DB2 アプリケーションを構築する準備ができました。以下の章では、ビルド・ファイルを含むサンプル・プログラムおよび関連ファイルについて説明します。この後に続く章では、ご使用のプログラミング環境でアプリケーションをコンパイル、リンク、および実行する方法を示すビルド・ファイルおよびサンプルを使用します。特定のアプリケーション開発における必要については、その該当の章を参照してください。

第 3 章 サンプル・プログラムおよび関連ファイル

サンプル・ファイル	69	オブジェクトのリンクと埋め込みデータベース	
言語およびアプリケーション・インターフェース別		(OLE DB) 表関数のサンプル	100
のサンプル・プログラム	75	Perl のサンプル	101
C サンプル	75	PHP のサンプル	102
C++ サンプル	79	REXX のサンプル	102
C# のサンプル	82	セキュリティ・プラグインのサンプル	105
CLI のサンプル	83	SQL プロシージャのサンプル	105
コマンド行プロセッサ (CLP) のサンプル	85	Visual Basic のサンプル	108
COBOL のサンプル	86	Visual Basic .NET のサンプル	109
動的再構成のサンプル	90	Visual C++ のサンプル	111
JDBC のサンプル	91	Windows Management Instrumentation のサンプル	112
SQLJ のサンプル	94	ビルド・ファイル、makefile、およびエラー・チェ	
Java WebSphere のサンプル	96	ック・ユーティリティ	112
Java プラグインのサンプル	96	ビルド・ファイル	112
ログ管理ユーザー出口サンプル	97	makefile	116
オブジェクトのリンクと埋め込み (OLE) のサン		エラー・チェック・ユーティリティ	119
プル	99		

この章では、DB2 がサポートするすべてのプラットフォーム用のプログラム言語のサンプル・プログラムおよび関連ファイルを記載しています。この章では、DB2 のコンポーネント構造に基づくサンプルの設計を提供し、それぞれのサンプルの説明がある DB2 サンプルのリストを提供します。この章では、DB2 に付属しているビルド・ファイル、makefile、およびエラー・チェック・ユーティリティの使用についても説明します。

サンプル・ファイル

サンプル・プログラムは、DB2® Application Development (DB2 AD) Client に付属しています。すべてのプラットフォームまたはサポート対象のプログラミング言語で、すべてのサンプル・プログラムを利用できるわけではありません。サンプル・プログラムをテンプレートとして使用して、独自のアプリケーションを作成したり、理解を深めるためのツールとして DB2 の機能を習得したりすることができます。

DB2 サンプル・プログラムは、どのような保証も付帯されずに「現状のまま」提供されています。品質、パフォーマンス、何らかの欠陥の訂正のすべてのリスクはすべて、IBM® ではなくユーザー側で負っていただきます。

DB2 では、サンプル・プログラム・ファイルの他に、sqlllib/samples (UNIX) の下または sqlllib\samples (Windows) の下のサンプル・ディレクトリーに他のサンプル・ファイルも用意されています。そのようなファイルには、サンプル・プログラムのコンパイルとリンクのためのビルド・ファイルと makefile、たいていのサンプル・プログラムにリンクしているエラー・チェック・ユーティリティ・ファイル、およびアプリケーション開発で役に立つ各種スクリプト・ファイルが組み込まれています。たとえば、いくつかの言語サブディレクトリー内にストアード・プロシージャと UDF のカタログとアンカタログを行うためのスクリプトが備えられ

ています。どのサンプル・ディレクトリーにも、ディレクトリーに置かれているファイルを説明した README ファイルがあります。

大半のサンプル・プログラム・ソース・ファイルには HTML バージョンが用意されていて、それにはオンライン文書でアクセスすることができます。このような「HTML のサンプル」は、文書内のトピックにリンクされていて、そこで説明されている機能が分かるようになっています。SQL ステートメントや DB2 API といったキーワードが HTML のサンプルにホット・リンクされているので、ユーザーはその説明文に直接移動することができます。HTML のサンプルの大半において、コンパイル済みのサンプル・プログラムの通常の実行結果を示したサンプル出力ファイルへのリンクが、ファイルの最上部のコメント・セクションに設けられています。多くの場合、実際の出力はマシンとプラットフォームに依存するため、同じプログラムを実行しても得られる出力は異なる可能性があることに注意してください。

以下は、サポートされるメインのプログラミング言語/API のサンプル・ディレクトリーと README ファイルをプラットフォーム別に示した表です。README ファイルはオンライン文書にホット・リンクされていて、その中のサンプル・リストは、サンプル・ファイルのソース・コードにホット・リンクされています。そのリスト中のサンプル・ディレクトリー内のサンプル・ファイルにアクセスすることもできます。ディレクトリー・パスに関しては、`samples/c` のように UNIX[®] 形式のスラッシュが使用されます。ただし、`samples¥VB¥ADO` といった Windows[®] 専用のディレクトリーの場合を除きます。

表 8. プラットフォーム別のサンプル README ファイル

プラットフォーム → 言語	AIX®	HP-UX	Linux	Solaris	Windows
C samples/c	README	README	README	README	README
C++ samples/cpp	README	README	README	README	README
C# samples¥.NET¥cs	n/a	n/a	n/a	n/a	README
CLI samples/cli	README	README	README	README	README
CLP samples/clp	README	README	README	README	README
IBM COBOL samples/cobol	README	n/a	n/a	n/a	README
Micro Focus COBOL samples/cobol_mf	README	README	README	README	README
JDBC samples/java/jdbc	README	README	README	README	README
SQLJ samples/java/sqlj	README	README	README	README	README
Perl samples/perl	README	README	README	README	README
PHP samples/php	README	README	README	README	README
SQL プロシージャ samples/sqlproc	README	README	README	README	README
Visual Basic samples¥VB¥ADO	n/a	n/a	n/a	n/a	ReadMe.txt
Visual Basic .NET samples¥.NET¥vb	n/a	n/a	n/a	n/a	README

サンプル・プログラムのファイル拡張子は、サポートされる各言語ごとに異なり、各言語内でも、組み込み SQL プログラムと非組み込み SQL プログラムとでは異なります。また、ファイル拡張子は、言語内のプログラム・グループで異なる場合があります。これらのサンプル・ファイル拡張子を分類したのが、次の表です。

言語別のサンプル・ファイル拡張子

72 ページの表 9。

プログラム・グループ別のサンプル・ファイル拡張子

72 ページの表 10。

表 9. 言語別のサンプル・ファイル拡張子

言語	ディレクトリー	組み込み SQL プログラム	組み込み SQL を含まないプログラム
C	samples/c samples/cli (CLI プログラム)	.sqc	.c
C++	samples/cpp	.sqc (UNIX) .sqx (Windows)	.C (UNIX) .cxx (Windows)
C#	samples¥.NET¥cs		.cs
COBOL	samples/cobol samples/cobol_mf	.sqb	.cbl
Java™	samples/java/jdbc samples/java/sqlj samples/java/WebSphere samples/java/plugin	.sqlj	.java
REXX	samples/rexx	.cmd	.cmd
Visual Basic	samples¥VB¥ADO samples¥VB¥MTS samples¥VB¥RDO		.bas .frm .vbp
Visual Basic .NET	samples¥.NET¥vb		.vb
Visual C++	samples¥VC¥ADO		.cpp .dsp .dsw

表 10. プログラム・グループ別のサンプル・ファイル拡張子

サンプル・グループ	ディレクトリー	ファイル拡張子
CLP	samples/clp	.db2
OLE	samples¥ole¥msvb (Visual Basic) samples¥ole¥msvc (Visual C++)	.bas .vbp (Visual Basic) .cpp (Visual C++)
OLE DB	samples¥oledb	.db2
SQL プロシージャ	samples/sqlproc	.db2 (SQL プロシージャ・スクリプト) .c (CLI クライアント・アプリケーション) .sqc (組み込み C クライアント・アプリケーション) .java (JDBC クライアント・アプリケーション)
ユーザー出口	samples/c	.ctsm (UNIX および Windows) .cdisk (UNIX および Windows) .ctape (UNIX) .cxbsa (UNIX)

注:

ディレクトリー区切り文字

UNIX でのディレクトリーの区切り文字は / です。 Windows では ¥ です。ディレクトリーが Windows でのみ使用可能でない限り、表の中では UNIX の区切り文字が使用されます。

組み込み SQL プログラム

このプログラムは、プリコンパイルが必要です。REXX 組み込み SQL プログラムは、プログラムの実行時に組み込み SQL ステートメントが解釈されるので例外になります。

IBM COBOL サンプル

AIX および Windows 32 ビットのエレーティング・システムの場合にのみ、cobol サブディレクトリーに用意されています。

Micro Focus COBOL サンプル

AIX、HP-UX、Solaris オペレーティング環境、および Windows 32 ビットのエレーティング・システム用にのみ、cobol_mf サブディレクトリーに用意されています。

Java サンプル

JDBC (Java Database Connectivity) アプレット、アプリケーション、およびルーチン、Java Embedded SQL (SQLJ) アプレット、アプリケーション、およびルーチンです。さらに、WebSphere® サンプルと、DB2 コントロール・センターのプラグイン例のファイルが入っています。Java サンプルは、サポートされるすべての DB2 プラットフォーム上で使用可能です。

REXX のサンプル

AIX および Windows 32 ビット・オペレーティング・システム用のもののみが提供されています。

CLP サンプル

SQL ステートメントを実行するコマンド行プロセッサのスクリプトです。

OLE サンプル

Microsoft® Visual Basic および Microsoft Visual C++ のオブジェクトのリンクと埋め込み (OLE) のためのサンプルで、Windows オペレーティング・システム上でのみ提供されます。

Visual Basic のサンプル

ActiveX Data Object、Remote Data Objects、および Microsoft Transaction Server サンプル (Windows オペレーティング・システムでのみ提供されます)。

Visual C++ のサンプル

ActiveX Data Object サンプル。これは Windows オペレーティング・システムでのみ提供されます。

ユーザー出口サンプル

データベース・ログ・ファイルを保存し検索するのに使用する、ログ管理ユーザー出口プログラムです。ファイルは、.c 拡張子を付けて名前変更し、C 言語プログラムとしてコンパイルしなければなりません。

サンプル・プログラム・ディレクトリーは、たいていのプラットフォームでは一般に読み取り専用です。サンプル・プログラムは、変更または構築する前に、ユーザーの作業ディレクトリーにコピーしてください。

構造および設計

C、CLI、C++、C#、Java、Perl、PHP、Visual Basic ADO、および Visual Basic .NET の DB2 サンプルの大半は、データベース・コンポーネントのオブジェクト・ベースの設計モデルを反映するように編成されています。これらのサンプルは、さまざまなレベルの DB2 を表すカテゴリー別にグループに分けられています。サンプルが属するレベルは、サンプル名の先頭に付く 2 文字の接頭部で示されます。各アプリケーション・プログラミング・インターフェースごとにすべてのレベルがサンプル内に提示されるわけではありません。サンプルの場合はレベルはむしろ、次のように表されます。

接頭部 DB2 のレベル

il	インストール・イメージ・レベル
cl	クライアント・レベル
in	インスタンス・レベル
db	データベース・レベル
ts	表スペース・レベル
tb	表レベル
dt	データ・タイプ・レベル

レベルは階層構造を示します。インストール・イメージ・レベルは、DB2 の最上位レベルです。このレベルの下では、クライアント・レベルのアプリケーションはさまざまなインスタンスにアクセスすることができます。たとえばインスタンスは 1 つ以上のデータベースをもつことができ、データベースは、テーブルが置かれた表スペースをもち、さらにそれらのテーブルはさまざまなデータ・タイプのデータをもちます。

この設計には、すべての DB2 サンプルが組み込まれているわけではありません。一部のサンプルの目的は、データにアクセスするさまざまな方式を示すことにあります。そのような方式がサンプルの主な目的であるため、上記のような方法でそれらの方式をサンプルで紹介しています。

接頭部 プログラミング方式

fn	SQL 関数
sp	ストアド・プロシージャ
ud	ユーザー定義関数

これらのカテゴリーのほかに、データベース・プログラミングの基本概念を紹介するための一連のチュートリアル・サンプルもあります。そのようなサンプルは、サンプル設計に示されている簡単ないくつかの関数を使用し、`tut` という文字で始まっています。

この設計に含まれていないその他のサンプルもあります。たとえば、ログ管理ユーザー出口のサンプル、COBOL のサンプル、Visual C++、REXX、オブジェクト・リンクおよび埋め込み (OLE) サンプル、CLP スクリプト、および SQL プロシージャなどがあります。

注: Java、C#、および Visual Basic .NET プログラム名では、最初の文字 (場合によっては他の文字も) は大文字になります。チュートリアル・サンプル名中では、

Java プログラムには下線は使用されません。 Visual Basic ADO サンプルでは、一部の文字は大文字になります (ただし先頭文字ではありません)。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 119 ページの『エラー・チェック・ユーティリティー』

関連資料:

- 75 ページの『C サンプル』
- 83 ページの『CLI のサンプル』
- 91 ページの『JDBC のサンプル』
- 94 ページの『SQLJ のサンプル』
- 105 ページの『SQL プロシージャのサンプル』
- 108 ページの『Visual Basic のサンプル』
- 111 ページの『Visual C++ のサンプル』
- 99 ページの『オブジェクトのリンクと埋め込み (OLE) のサンプル』
- 100 ページの『オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数のサンプル』
- 85 ページの『コマンド行プロセッサ (CLP) のサンプル』
- 97 ページの『ログ管理ユーザー出口サンプル』
- 86 ページの『COBOL のサンプル』
- 96 ページの『Java WebSphere のサンプル』
- 96 ページの『Java プラグインのサンプル』
- 112 ページの『Windows Management Instrumentation のサンプル』
- 102 ページの『REXX のサンプル』
- 90 ページの『動的再構成のサンプル』
- 82 ページの『C# のサンプル』
- 109 ページの『Visual Basic .NET のサンプル』
- 101 ページの『Perl のサンプル』
- 102 ページの『PHP のサンプル』
- 105 ページの『セキュリティ・プラグインのサンプル』

言語およびアプリケーション・インターフェース別のサンプル・プログラム

C サンプル

UNIX ディレクトリ: `sqllib/samples/c`。 Windows ディレクトリ: `sqllib\samples\c`。

ファイル拡張子: .c (非組み込み SQL); .sqc (組み込み SQL)

表 11. C サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
チュートリアル・サンプル - データベースの基本操作の例を示すプログラム。	
tut_mod.sqc	表データの変更方法。
tut_read.sqc	表の読み方。
tut_use.sqc	データベースの使用方法。
クライアント・レベル - DB2 のクライアント・レベルを扱うサンプル。	
cli_info.c	クライアント・レベル情報の入手および設定方法。
clisnap.c	クライアント・レベルのスナップショットのキャプチャー方法。
インスタンス・レベル - DB2 のインスタンス・レベルを扱うサンプル。	
inattach.c	インスタンスのアタッチ/切り離しの方法。
inauth.sqc	インスタンス・レベルでの権限の表示方法。
ininfo.c	インスタンス・レベル情報の入手および設定方法。
insnap.c	インスタンス・レベルのスナップショットのキャプチャー方法。
instart.c	現行ローカル・インスタンスの停止と始動の方法。
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
dbauth.sqc	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
dbcfg.sqc	データベースとデータベース・マネージャー・パラメーターの構成方法。
dbconn.sqc	データベースからの接続および切断方法。
dbcreate.c	データベースの作成とドロップの方法。
dbhistfile.sqc	データベース・リカバリー履歴ファイル項目の読み取りおよび更新方法。
dbinfo.c	データベース・レベルの情報の入手および設定方法。
dbinline.sqc	インライン SQL プロシージャ言語の使用方法。
dbinspec.sqc	DB2 API db2Inspect による体系の保全性の検査方法。
dblogconn.sqc	データベース接続によるデータベース・ログ・ファイルの非同期読み取り方法。
dblognoconn.sqc	データベース接続なしでのデータベース・ログ・ファイルの非同期読み取り方法。
dbmcon.sqc	複数のデータベースの接続および切断方法。
dbmcon1.h	dbmcon1.sqc のヘッダー・ファイル。
dbmcon1.sqc	dbmcon.sqc のサポート・ファイル。
dbmcon2.h	dbmcon2.sqc のヘッダー・ファイル。
dbmcon2.sqc	dbmcon.sqc のサポート・ファイル。
dbmigrat.c	データベースの移行方法。
dbpkg.sqc	パッケージの処理方法。
dbrec.sqc	db2GetRecommendations API の使用方法。
dbrecov.sqc	データベースのリカバリー方法。
dbredirect.sqc	データベースのリダイレクト・リストアの実行方法。

表 11. C サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
dbrestore.sqc	データベースのバックアップからのリストア方法。
dbrollfwd.sqc	データベースのリストア後のロールフォワードの実行方法。
dbsample.sqc	ホストおよび AS/400 の表およびビューを含めたサンプル・データベースの作成方法。
dbsnap.c	データベース・レベルのスナップショットのキャプチャー方法。
dbthrds.sqc	UNIX での複数コンテキスト API の使用方法。
dbthrds.sqc	Windows での複数コンテキスト API の使用方法。
dbuse.sqc	データベース・オブジェクトの使用法。
表スペース・レベル - 表スペース・レベルの DB2 を扱うサンプル。	
tscreate.sqc	バッファ・プールと表スペースの作成とドロップの方法。
tsinfo.sqc	表スペース・レベルの情報の入手方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
tbast.sqc	ステージング表を使用した据え置き自動サマリー表の更新方法。
tbcompress.sqc	null およびデフォルト値圧縮オプションを使用した表の作成方法。
tbconstr.sqc	表の制約の処理方法。
tbcreate.sqc	表の作成、変更、およびドロップ方法。
tbident.sqc	ID 列の使用法。
tbinfo.sqc	表レベルの情報の入手および設定方法。
tbintrig.sqc	ビューでの 'INSTEAD OF' トリガーの使用法。
tbload.sqc	パーティション・データベースへのロード方法。
tbmerge.sqc	MERGE ステートメントの使用法。
tbmod.sqc	表内の情報の修正方法。
tbmove.sqc	表データの移動方法。
tbonlineinx.sqc	表の索引の作成および再編成方法。
tbpriv.sqc	表レベル特権の GRANT /表示/取り消しの方法。
tbread.sqc	表内の情報の読み取り方法。
tbreorg.sqc	表の再編成方法。
tbrunstats.sqc	表に対する RUNSTATS の実行方法。
tbsavept.sqc	外部セーブポイントの使用法。
tbset.sqc	挿入、更新、削除のそれぞれでの選択の方法。
tbsetcreate.db2	tbset プログラム用の表の作成方法。
tbsetdrop.db2	tbset プログラム用の表のドロップ方法。
tbtemp.sqc	宣言済み一時表の使用法。
tbtrig.sqc	表でのトリガーの使用法。
tbumqt.sqc	ユーザー・マテリアライズ照会表 (サマリー表) の使用法。
tbunion.sqc	UNION ALL ビューによる挿入方法。
tbxload.sqc	SELECT ステートメントから同時にデータを戻して表にロードする方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	

表 11. C サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
dtformat.sqc	ロードおよびインポートのデータ・フォーマットの拡張子の使用方法。
dtlob.sqc	LOB データの読み取りおよび書き込み方法。
dtudt.sqc	ユーザー定義特殊タイプの作成、使用、およびドロップ方法。
DB2 関数レベル	
fnuse.sqc	SQL 関数の使用方法。
ストアド・プロシージャ・レベル - ストアド・プロシージャを示すサンプル。	
spcat	spserver プログラムのストアド・プロシージャ・カタログ・スクリプト。このスクリプトは、spdrop.db2 と spcreate.db2 を呼び出します。
spcreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
spdrop.db2	カタログからストアド・プロシージャをドロップするための CLP スクリプト。
spclient.sqc	spserver.sqc 内で宣言されるサーバー・ルーチンを呼び出すために使用されるクライアント・プログラム。
spserver.sqc	サーバー上で構築および実行されるストアド・プロシージャ・ルーチン。
UDF レベル - ユーザー定義関数を示すサンプル。	
udfcli.sqc	udfsrv.c、udfsrv.C 内のユーザー定義関数を呼び出すクライアント・アプリケーション。
udfsrv.c	udfcli.sqc によって呼び出されるユーザー定義関数 ScalarUDF。
udfemcli.sqc	組み込み SQL ユーザー定義関数ライブラリー udfemsrv を呼び出すクライアント・アプリケーション。
udfemsrv.sqc	udfemcli によって呼び出される組み込み SQL ユーザー定義関数ライブラリー。
その他	
evm.sqc	ファイル、パイプ、および表のイベント・モニターの作成および解析方法。
utilrecov.c	バックアップ、リストア、およびログ・ファイルのサンプル用のユーティリティー。
utilsnap.c	スナップショット・モニター・サンプル用のユーティリティー。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 119 ページの『エラー・チェック・ユーティリティー』
- 69 ページの『サンプル・ファイル』

C++ サンプル

UNIX ディレクトリ: `sqllib/samples/cpp`。 Windows ディレクトリ: `sqllib¥samples¥cpp`。

UNIX ファイル拡張子: `.C` (非組み込み SQL); `.sqC` (組み込み SQL)

Windows ファイル拡張子: `.cxx` (非組み込み SQL); `.sqx` (組み込み SQL)

表 12. C++ サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
チュートリアル・サンプル - データベースの基本操作の例を示すプログラム。	
<code>tut_mod.sqC</code>	表データの変更方法。
<code>tut_read.sqC</code>	表の読み方。
<code>tut_use.sqC</code>	データベースの使用方法。
クライアント・レベル - DB2 のクライアント・レベルを扱うサンプル。	
<code>cli_info.C</code>	クライアント・レベル情報の入手および設定方法。
<code>clisnap.C</code>	クライアント・レベルのスナップショットのキャプチャー方法。
インスタンス・レベル - DB2 のインスタンス・レベルを扱うサンプル。	
<code>inattach.C</code>	インスタンスのアタッチ/切り離しの方法。
<code>inauth.sqC</code>	インスタンス・レベルでの権限の表示方法。
<code>ininfo.C</code>	インスタンス・レベル情報の入手および設定方法。
<code>insnap.C</code>	インスタンス・レベルのスナップショットのキャプチャー方法。
<code>instart.C</code>	現行ローカル・インスタンスの停止と始動の方法。
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
<code>dbauth.sqC</code>	データベース・レベルでの権限の <code>GRANT</code> /表示/取り消しの方法。
<code>dbcfg.sqC</code>	データベースとデータベース・マネージャー・パラメーターの構成方法。
<code>dbconn.sqC</code>	データベースからの接続および切断方法。
<code>dbcreate.C</code>	データベースの作成とドロップの方法。
<code>dbhistfile.sqC</code>	データベース・リカバリー・ファイル項目の読み取りおよび更新方法。
<code>dbinfo.C</code>	データベース・レベルの情報の入手および設定方法。
<code>dbinline.sqC</code>	インライン SQL プロシージャ言語の使用法。
<code>dbinspec.sqC</code>	DB2 API <code>db2Inspect</code> による体系の保全性の検査方法。
<code>dblogconn.sqC</code>	データベース接続によるデータベース・ログ・ファイルの非同期読み取り方法。
<code>dblognoconn.sqC</code>	データベース接続なしでのデータベース・ログ・ファイルの非同期読み取り方法。
<code>dbmcon.sqC</code>	複数のデータベースの接続および切断方法。
<code>dbmcon1.h</code>	<code>dbmcon1.sqC</code> のヘッダー・ファイル。
<code>dbmcon1.sqC</code>	<code>dbmcon.sqC</code> のサポート・ファイル。
<code>dbmcon2.h</code>	<code>dbmcon2.sqC</code> のヘッダー・ファイル。
<code>dbmcon2.sqC</code>	<code>dbmcon.sqC</code> のサポート・ファイル。
<code>dbmigrat.C</code>	データベースの移行方法。

表 12. C++ サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
dbpkg.sqC	パッケージの処理方法。
dbrec.sqC	db2GetRecommendations API の使用方法。
dbrecov.sqC	データベースのリカバリー方法。
dbredirect.sqC	リダイレクト・リストアを使用したデータベースのリカバリー方法。
dbrestore.sqC	データベースのリカバリー方法。
dbrollfwd.sqC	ロールフォワード・リカバリーを使用したデータベースのリカバリー方法。
dbsample.sqC	ホストおよび AS/400 の表およびビューを含めたサンプル・データベースの作成方法。
dbsnap.C	データベース・レベルのスナップショットのキャプチャー方法。
dbthrs.sqC	UNIX での複数コンテキスト API の使用方法。
dbthrs.sqC	Windows での複数コンテキスト API の使用方法。
dbuse.sqC	データベース・オブジェクトの使用方法。
表スペース・レベル - 表スペース・レベルの DB2 を扱うサンプル。	
tscreate.sqC	バッファ・プールと表スペースの作成とドロップの方法。
tsinfo.sqC	表スペース・レベルの情報の入手方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
tbconstr.sqC	表の制約の処理方法。
tbcreate.sqC	表の作成、変更、およびドロップ方法。
tbident.sqC	ID 列の使用方法。
tbinfo.sqC	表レベルの情報の入手および設定方法。
tbintrig.sqC	ビューでの 'INSTEAD OF' トリガーの使用方法。
tbmerge.sqC	MERGE ステートメントの使用方法。
tbmod.sqC	表内の情報の修正方法。
tbmove.sqC	表データの移動方法。
tbpriv.sqC	表レベル特権の GRANT /表示/取り消しの方法。
tbread.sqC	表内の情報の読み取り方法。
tbreorg.sqC	表の再編成方法。
tbsepts.sqC	外部セーブポイントの使用方法。列のデフォルト値の変更方法も示します。
tbsetl.sqC	挿入、更新、削除のそれぞれでの選択の方法。
tbsetlcreate.db2	tbsetl プログラム用の表の作成方法。
tbsetldrop.db2	tbsetl プログラム用の表のドロップ方法。
tbtemp.sqC	宣言済み一時表の使用方法。
tbtrig.sqC	表でのトリガーの使用方法。
tbunion.sqC	UNION ALL ビューによる挿入方法。
tbxload.sqC	SELECT ステートメントから同時にデータを戻して表にロードする方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
dtformat.sqC	ロードおよびインポートのデータ・フォーマットの拡張子の使用方法。

表 12. C++ サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
dtlob.sqC	LOB データの読み取りおよび書き込み方法。
dtstruct.sqC	構造型および型付き表の階層の作成、使用、およびドロップの方法。
dtudt.sqC	ユーザー定義特殊タイプの作成、使用、およびドロップ方法。
DB2 関数レベル	
fnuse.sqC	SQL 関数の使用方法。
ストアド・プロシージャ・レベル - ストアド・プロシージャを示すサンプル。	
spcat	spserver プログラムのストアド・プロシージャ・カタログ・スクリプト。このスクリプトは、spdrop.db2 と spcreate.db2 を呼び出します。
spcreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
spdrop.db2	カタログからストアド・プロシージャをドロップするための CLP スクリプト。
spclient.sqC	spserver.sqc、spserver.sqC 内で宣言されたサーバー・ルーチンの呼び出しで使用されるクライアント・プログラム。
spserver.sqC	サーバー上で構築および実行されるストアド・プロシージャ・ルーチン。
UDF レベル - ユーザー定義関数を示すサンプル。	
udfcli.sqC	udfsrv.c、udfsrv.C 内のユーザー定義関数を呼び出すクライアント・アプリケーション。
udfsrv.C	udfcli.sqc、udfcli.sqC によって呼び出されるユーザー定義関数 ScalarUDF。
udfemcli.sqC	組み込み SQL ユーザー定義関数ライブラリー udfemsrv を呼び出すクライアント・アプリケーション。
udfemsrv.sqC	udfemcli によって呼び出される組み込み SQL ユーザー定義関数ライブラリー。
その他	
evm.sqC	ファイル、パイプ、および表のイベント・モニターの作成および解析方法。
utilrecov.C	バックアップ、リストア、およびログ・ファイルのサンプル用のユーティリティ。
utilsnap.C	スナップショット・モニター・サンプル用のユーティリティ。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 119 ページの『エラー・チェック・ユーティリティ』
- 69 ページの『サンプル・ファイル』

C# のサンプル

ディレクトリ: sqllib\samples\%.NET\cs。

表 13. C# .NET サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
DbAuth.cs	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
DbDatAdp.cs	DB2DataAdapter の使用方法。
DbDatMap.cs	DataTable および DataColumn マッピングのセットアップ方法。
DbDsetCn.cs	既存の制約の DataSet への追加方法。
DbEvent.cs	DB2DataAdapter イベントの処理方法。
DbUse.cs	データベース・オブジェクトの使用法。
DbValue.cs	データベースからの単一値の取得方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
TbConstr.cs	表の制約の処理方法。
TbInfo.cs	表レベルの情報の入手および設定方法。
TbPriv.cs	表レベル特権の GRANT /表示/取り消しの方法。
TbSel.cs	挿入、更新、削除のそれぞれでの選択の方法。
TbTrig.cs	表でのトリガーの使用法。
TbUse.cs	表データの操作およびデータベースへの接続/データベースからの切断の方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
DtLob.cs	LOB データ・タイプの使用法。
ストアド・プロシージャ - ストアド・プロシージャの例を示すサンプル。	
SpCat.db2	SpServer.cs にインプリメントされるプロシージャのドロップおよび作成。
SpClient.cs	SpServer.cs 内のストアド・プロシージャの呼び出しに使用されるクライアント・プログラム。
SpCreate.db2	SpServer.cs にインプリメントされる外部プロシージャの作成。
SpDrop.db2	C# 用 SpCreate.db2 に作成される外部プロシージャのドロップ。
SpReturn.cs	ストアド・プロシージャ EMP_DETAILS を呼び出して戻り値を取得するクライアント・アプリケーション。
SpServer.cs	SpCat.db2 に作成されるプロシージャの C# 外部コード・インプリメンテーション。
EmpDetails.db2	EMP_DETAILS という名前のストアド・プロシージャを作成する CLP スクリプト。
ユーザー定義関数 - UDF の例を示すサンプル。	
UdfCat.db2	UdfSrv.cs にインプリメントされる外部 UDF のドロップおよび作成。
UdfCli.cs	UdfSrv.cs 内のユーザー定義関数を呼び出すクライアント・アプリケーション。
UdfCreate.db2	UdfSrv.cs にインプリメントされる外部 UDF の作成。
UdfDrop.db2	C# 用 udfcreate.db2 に作成される外部 UDF のドロップ。

表 13. C# .NET サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
UdfSrv.cs	UdfCli.cs によって呼び出されるユーザー定義スカラー関数。
疎結合トランザクション	
empcat.bat	C# クライアント・プログラム SpReturn 用ストアード・プロシージャ EMP_DETAILS のカタログ。
LCTrans.cs	疎結合トランザクションを示します。
regCOM.bat	C# LCTrans プログラム用 COM+ オブジェクトの登録。
RootCOM.cs	このファイルは、ライブラリー・アセンブリー RootCOM.dll を作成するのに使用します。 LCTrans.cs は、このファイル中に定義されているクラスと方式を参照します。
SubCOM.cs	このファイルは、ライブラリー・アセンブリー SubCOM.dll を作成するのに使用します。 LCTrans.cs は、このファイル中に定義されているクラスと方式を参照します。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 285 ページの『C# .NET アプリケーションの構築』
- 293 ページの『Common Language Runtime (CLR) .NET ルーチンの構築』

CLI のサンプル

UNIX ディレクトリー: sqllib/samples/cli。 Windows ディレクトリー: sqllib\samples\cli。

表 14. サンプル CLI プログラム・ファイル

サンプル・プログラム名	プログラムの説明
チュートリアル・サンプル - データベースの基本操作の例を示すプログラム。	
tut_mod.c	表データの変更方法。
tut_read.c	表の読み方。
tut_use.c	データベースの使用方法。
インストール・イメージ・レベル - DB2 と CLI のインストール・イメージ・レベルを扱うサンプル。	
ilinfo.c	インストール・レベル情報 (CLI ドライバーのバージョンなど) の入手および設定方法。
クライアント・レベル - DB2 のクライアント・レベルを扱うサンプル。	
cli_info.c	クライアント・レベル情報の入手および設定方法。
clihandl.c	ハンドルの割り当ておよび解放方法。

表 14. サンプル CLI プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
clisqlca.c	SQLCA データの処理方法。
インスタンス・レベル - DB2 のインスタンス・レベルを扱うサンプル。	
ininfo.c	インスタンス・レベル情報の入手および設定方法。
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
dbcongui.c	グラフィカル・ユーザー・インターフェース (GUI) を使用したデータベースへの接続方法。
dbconn.c	データベースからの接続および切断方法。
dbinfo.c	データベース・レベルの情報の入手および設定方法。
dbmcon.c	複数のデータベースからの接続および切断方法。
dbmconx.c	組み込み SQL による複数のデータベースからの接続および切断方法。
dbmconx1.h	dbmconx1.sqc のヘッダー・ファイル。
dbmconx1.sqc	dbmconx プログラムの組み込み SQL ファイル。
dbmconx2.h	dbmconx2.sqc のヘッダー・ファイル。
dbmconx2.sqc	dbmconx プログラムの組み込み SQL ファイル。
dbnative.c	ODBC エスケープ文節を含むステートメントを、データ・ソース特有の形式に変換する方法。
dbuse.c	データベース・オブジェクトの使用方法。
dbusemx.sqc	組み込み SQL によるデータベース・オブジェクトの使用法。
dbxamon.c	未確定トランザクションの表示およびロールバック方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
tbconstr.c	表の制約の処理方法。
tbcreate.c	表の作成、変更、およびドロップ方法。
tbinfo.c	表レベルの情報の入手および設定方法。
tbload.c	CLI LOAD ユーティリティを使用してデータを挿入する方法です。
tbmod.c	表内の情報の修正方法。
tbread.c	表内の情報の読み取り方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
dtinfo.c	データ・タイプに関する情報の入手方法。
dtlob.c	LOB データの読み取りおよび書き込み方法。
dtudt.c	ユーザー定義特殊タイプの作成、使用、およびドロップ方法。
ストアド・プロシージャ・レベル - ストアド・プロシージャを示すサンプル。	
spcat	spserver プログラムのストアド・プロシージャ・カタログ・スクリプト。このスクリプトは、spdrop.db2 と spcreate.db2 を呼び出します。
spcreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
spdrop.db2	カタログからストアド・プロシージャをドロップするための CLP スクリプト。
spclient.c	spserver.c 内で宣言されるサーバー関数を呼び出すために使用されるクライアント・プログラム。
spserver.c	サーバー上で構築および実行されるストアド・プロシージャ関数。

表 14. サンプル CLI プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
spclires.c	複数の結果セットの SQLMoreResults と SQLNextResults の相違を示すクライアント・アプリケーション。
spcall.c	任意のストアド・プロシージャを呼び出すためのクライアント・プログラム。
UDF レベル - ユーザー定義関数を示すサンプル。	
udfcli.c	udfsrv.c 内のユーザー定義関数を呼び出すクライアント・アプリケーション。
udfsrv.c	udfcli.c によって呼び出されるユーザー定義関数 ScalarUDF。
共通ユーティリティー・ファイル	
utilcli.c	CLI サンプルで使用されるユーティリティー関数。
utilcli.h	CLI サンプルで使用されるユーティリティー関数用のヘッダー・ファイル。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 119 ページの『エラー・チェック・ユーティリティー』
- 69 ページの『サンプル・ファイル』

コマンド行プロセッサ (CLP) のサンプル

UNIX ディレクトリ: sqllib/samples/clp。 Windows ディレクトリ: sqllib\samples\clp。

表 15. コマンド行プロセッサ (CLP) サンプル・スクリプト

サンプル・ファイル名	ファイルの説明
autocfg.db2	パフォーマンス構成ウィザードの推奨に基づいて、データベースおよびデータベース・マネージャーの構成パラメーターを自動的に構成する方法。
const.db2	CHECK CONSTRAINT 文節がある表を作成します。
cte.db2	共通表式を示します。
flt.db2	再帰的照会を示します。
healthmon.db2	ヘルス・モニター・スナップショットでの表関数の使用方法。
join.db2	表の外部結合を示します。
onlineload.db2	ALLOW READ ACCESS オプションを使用したオンライン・ロードの方法。
stock.db2	トリガーの使用例を示します。

表 15. コマンド行プロセッサ (CLP) サンプル・スクリプト (続き)

サンプル・ファイル名	ファイルの説明
testdata.db2	ランダムに生成されるテスト・データを表に入れるための RAND() および TRANSLATE() などの DB2 組み込み関数を使用します。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 150 ページの『コマンド行プロセッサ (CLP) からのプロシーチャーの呼び出し』
- 149 ページの『コマンド行プロセッサ (CLP) スクリプトの実行』

COBOL のサンプル

UNIX ディレクトリーは次のとおりです。 IBM COBOL: sqllib/samples/cobol。
Micro Focus COBOL: sqllib/samples/cobol_mf。

Windows ディレクトリーは次のとおりです。 IBM COBOL:
sqllib\samples\cobol。 Micro Focus COBOL: sqllib\samples\cobol_mf。

注: COBOL サンプルは、C、CLI、C++、C#、Java、Perl、PHP、Visual Basic ADO、および Visual Basic .NET サンプルで使用される DB2 レベル設計の構造にはなっていません。

表 16. 組み込み SQL なしの COBOL DB2 API サンプル・プログラム

サンプル・プログラム	組み込み API
checkerr.cbl	<ul style="list-style-type: none"> • sqlaintp - エラー・メッセージの入手 • sqllogstt - SQLSTATE メッセージの入手
client.cbl	<ul style="list-style-type: none"> • sqleqryc - クライアントの照会 • sqlesetc - クライアントの設定
d_dbconf.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqledtin - 切り離し • sqlfddb - データベース構成デフォルト値の入手
d_dbmcon.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqledtin - 切り離し • sqlfdsys - データベース・マネージャー構成デフォルト値の入手

表 16. 組み込み SQL なしの COBOL DB2 API サンプル・プログラム (続き)

サンプル・プログラム	組み込み API
db_udcs.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlecrea - データベースの作成 • sqledrpd - データベースのドロップ
dbcat.cbl	<ul style="list-style-type: none"> • sqlecadb - データベースのカタログ • db2DbDirCloseScan - データベース・ディレクトリー・スキャンのクローズ • db2DbDirGetNextEntry - 次のデータベース・ディレクトリー項目の入手 • db2DbDirOpenScan - データベース・ディレクトリー・スキャンのオープン • sqleuncd - データベースのアンカタログ
dbcmt.cbl	<ul style="list-style-type: none"> • sqledcgd - データベースのコメントの変更 • db2DbDirCloseScan - データベース・ディレクトリー・スキャンのクローズ • db2DbDirGetNextEntry - 次のデータベース・ディレクトリー項目の入手 • db2DbDirOpenScan - データベース・ディレクトリー・スキャンのオープン • sqleisig - シグナル・ハンドラーのインストール
dbconf.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlecrea - データベースの作成 • sqledrpd - データベースのドロップ • sqlfrdb - データベース構成のリセット • sqlfudb - データベース構成の更新 • sqlfxdb - データベース構成の入手
dbinst.cbl	<ul style="list-style-type: none"> • sqleatcp - アタッチおよびパスワードの変更 • sqleatin - アタッチ • sqledtin - 切り離し • sqlegins - インスタンス
dbmconf.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqledtin - 切り離し • sqlfrsys - データベース・マネージャー構成のリセット • sqlfusys - データベース・マネージャー構成の更新 • sqlfxsys - データベース・マネージャー構成の入手
dbsnap.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlmonss - スナップショットの入手
dbstart.cbl	<ul style="list-style-type: none"> • sqlepstart - データベース・マネージャーの開始

表 16. 組み込み SQL なしの COBOL DB2 API サンプル・プログラム (続き)

サンプル・プログラム	組み込み API
dbstop.cbl	<ul style="list-style-type: none"> • sqlfrce - アプリケーションの強制 • sqlpstp - データベース・マネージャーの停止
dcscat.cbl	<ul style="list-style-type: none"> • sqlgdad - DCS データベースのカatalog • sqlgdcl - DCS ディレクトリー・スキャンのクローズ • sqlgdel - DCS データベースのアンカatalog • sqlgdge - データベースの DCS ディレクトリー項目の入手 • sqlgdgt - DCS ディレクトリー項目の入手 • sqlgdsc - DCS ディレクトリー・スキャンのオープン
ebcdicdb.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlcrea - データベースの作成 • sqldrpd - データベースのドロップ
migrate.cbl	<ul style="list-style-type: none"> • sqlmgdb - データベースの移行
monreset.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlmrset - モニターのリセット
monsz.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlmonss - スナップショットの入手 • sqlmonsz - sqlmonss() 出力バッファに必要のサイズの見積もり
nodecat.cbl	<ul style="list-style-type: none"> • sqlctnd - ノードのカatalog • sqlencls - ノード・ディレクトリー・スキャンのクローズ • sqlengne - 次のノード・ディレクトリー項目の入手 • sqlenops - ノード・ディレクトリー・スキャンのオープン • sqluncn - ノードのアンカatalog
restart.cbl	<ul style="list-style-type: none"> • sqlerstd - データベースの再始動
setact.cbl	<ul style="list-style-type: none"> • sqlsact - アカウンティング・ストリングの設定
sws.cbl	<ul style="list-style-type: none"> • sqleatin - アタッチ • sqlmon - モニター・スイッチの入手/更新

表 17. COBOL DB2 API 組み込み SQL サンプル・プログラム

サンプル・プログラム	組み込み API
dbauth.sqb	<ul style="list-style-type: none"> • sqluadai - 許可の入手
dbstat.sqb	<ul style="list-style-type: none"> • db2Reorg - 表の再編成 • db2Runstats - 統計の実行
expsamp.sqb	<ul style="list-style-type: none"> • db2Export - エクスポート • sqluimpr - インポート
impexp.sqb	<ul style="list-style-type: none"> • db2Export - エクスポート • sqluimpr - インポート

表 17. COBOL DB2 API 組み込み SQL サンプル・プログラム (続き)

サンプル・プログラム	組み込み API
loadqry.sqb	<ul style="list-style-type: none"> • db2LoadQuery - 照会のロード
rebind.sqb	<ul style="list-style-type: none"> • sqlarbnd - 再バインド
tabscont.sqb	<ul style="list-style-type: none"> • sqlbctcq - 表スペース・コンテナ照会のクローズ • sqlbftcq - 表スペース・コンテナ照会の取り出し • sqlbotcq - 表スペース・コンテナ照会のオープン • sqlbtcq - 表スペース・コンテナ照会 • sqlfmem - 空きメモリー
tabspace.sqb	<ul style="list-style-type: none"> • sqlbctsq - 表スペース照会のクローズ • sqlbftpq - 表スペース照会の取り出し • sqlbgts - 表スペース統計の入手 • sqlbmtsq - 表スペース照会 • sqlbotsq - 表スペース照会のオープン • sqlbstpq - 単一表スペース照会 • sqlfmem - 空きメモリー
tload.sqb	<ul style="list-style-type: none"> • db2Export - エクスポート • sqluload - ロード • sqluvqdp - 表の表スペースの静止
tspace.sqb	<ul style="list-style-type: none"> • sqlbctcq - 表スペース・コンテナ照会のクローズ • sqlbctsq - 表スペース照会のクローズ • sqlbftcq - 表スペース・コンテナ照会の取り出し • sqlbftpq - 表スペース照会の取り出し • sqlbgts - 表スペース統計の入手 • sqlbmtsq - 表スペース照会 • sqlbotcq - 表スペース・コンテナ照会のオープン • sqlbotsq - 表スペース照会のオープン • sqlbstpq - 単一表スペース照会 • sqlbstsc - 表スペース・コンテナの設定 • sqlbtcq - 表スペース・コンテナ照会 • sqlfmem - 空きメモリー

表 18. DB2 API なしの COBOL 組み込み SQL サンプル・プログラム

サンプル・プログラム名	プログラムの説明
advsql.sqb	CASE、CAST、およびスカラー全選択などの拡張 SQL 式の使用例を示します。
cursor.sqb	静的 SQL を使用するカーソルの使用例を示します。
delet.sqb	データベースから項目を削除する静的 SQL の使用例を示します。
dynamic.sqb	動的 SQL を使用するカーソルの使用例を示します。

表 18. DB2 API なしの COBOL 組み込み SQL サンプル・プログラム (続き)

サンプル・プログラム名	プログラムの説明
joinsql.sqb	拡張 SQL 結合式の使用例を示します。
lobeval.sqb	LOB ロケーターの使用例を示し、実際の LOB データの評価を遅らせます。
lobfile.sqb	LOB ファイル・ハンドルの使用例を示します。
lobloc.sqb	LOB ロケーターの使用例を示します。
openftch.sqb	静的 SQL を使用した行の取り出し、更新、および削除について示します。
static.sqb	情報を検索する静的 SQL を示します。
tabsql.sqb	拡張 SQL 表式の使用例を示します。
trigsq1.sqb	拡張 SQL トリガーおよび制約の使用例を示します。
updat.sqb	データベースを更新する静的 SQL の使用例を示します。
varinp.sqb	パラメーター・マーカーを使用した組み込み動的 SQL ステートメント呼び出しへの変数入力を示します。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 119 ページの『エラー・チェック・ユーティリティー』
- 69 ページの『サンプル・ファイル』

動的再構成のサンプル

ディレクトリー (AIX および Solaris): sql1ib/samples/DLPAR。

表 19. 動的再構成のサンプル・スクリプト

サンプル・スクリプト名	ファイルの説明
ibm_db2_sl1n	この AIX 用の Korn シェル動的再構成スクリプト (DR スクリプト) は、p690 や p670 などの POWER4 ベースの pSeries システムで実行される AIX バージョン 5.2 に付属する Dynamic Logical Partitioning (DLPAR) 機能の使用を容易にします。これらの機能によって、中央演算処理装置やメモリーなどのリソースをアクティブな論理区画に対してリブートすることなく動的に追加および除去できるようになります。この DR スクリプトを動的再構成イベントの際に呼び出せば、DB2 の操作に影響を与えないでイベントが生じることを保証できます。DB2 構成は、イベントに合わせて動的に変更されます。詳細については、DR スクリプト自体を参照してください。

表 19. 動的再構成のサンプル・スクリプト (続き)

サンプル・スクリプト名	ファイルの説明
IBM,DB2	Solaris オペレーティング環境用のこの perl スクリプトは、DB2 と Reconfiguration and Coordination Manager (RCM) との間のインターフェースを提供する動的再構成および調整ツールによって使用されます。これらのツールは、3800 シリーズ以降のマシン上で実行される Solaris 9 以降に付属しています。このスクリプトを正しく使用すると、ハードウェア・リソースが除去された場合でも DB2 が動作を続けることが保証されます。詳細については、スクリプト自体を参照してください。

ibm_db2_slm サンプルは DB2 for AIX の sqllib/samples/DLPAR ディレクトリーに、IBM,DB2 サンプルは DB2 for Solaris の sqllib/samples/DLPAR ディレクトリーにあります。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 「管理ガイド: パフォーマンス」の『パラメーターの動的な構成』

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』

JDBC のサンプル

UNIX ディレクトリー: sqllib/samples/java/jdbc。

Windows ディレクトリー: sqllib¥samples¥java¥jdbc。

表 20. JDBC サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
チュートリアル・サンプル - データベースの基本操作の例を示すプログラム。	
TutMod.java	表データの変更方法。
TutRead.java	表の読み方。
インストール・イメージ・レベル - DB2 のインストール・イメージ・レベルを扱うサンプル。	
IlInfo.java	インストール・レベル情報の入手および設定方法。
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
DbAuth.java	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
DbConn.java	データベースからの接続および切断方法。
DbInfo.java	データベース・レベルの情報の入手および設定方法。

表 20. JDBC サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
DbMCon.java	複数のデータベースからの接続および切断方法。
DbNative.java	ODBC エスケープ文節を含むステートメントを、データ・ソース特有の形式に変換する方法。
DbRsHold.java	Legacy JDBC Type 2 および Universal JDBC ドライバーでの、結果セット・カーソル保留機能の使用法。このサンプルをコンパイルするには、Java Developer Kit 1.4 以上が必要です。このサンプルを実行するには、Java Runtime Environment 1.4 以上が必要です。
DbSeq.java	データベース内のシーケンスの作成、変更、およびドロップの方法。
DbUse.java	データベース・オブジェクトの使用法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
TbConstr.java	表の制約の処理方法。
TbCreate.java	表の作成、変更、およびドロップ方法。
TbGenCol.java	生成された列の使用法。
TbIdent.java	ID 列の使用法。
TbInfo.java	表レベルの情報の入手および設定方法。
TbInTrig.java	ビューでの 'INSTEAD OF' トリガーの使用法。
TbMerge.java	MERGE ステートメントの使用法。
TbMod.java	表内の情報の修正方法。
TbPriv.java	表レベル特権の GRANT /表示/取り消しの方法。
TbRead.java	表内の情報の読み取り方法。
TbSel.java	挿入、更新、削除のそれぞれでの選択の方法。
TbTemp.java	宣言済み一時表の使用法。
TbTrig.java	表でのトリガーの使用法。
TbUnion.java	UNION ALL ビューによる挿入方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
DtInfo.java	データ・タイプに関する情報の入手方法。
DtLob.java	LOB データの読み取りおよび書き込み方法。
DtUdt.java	ユーザー定義特殊タイプの作成、使用、およびドロップ方法。
アプレット - アプレットの例を示すサンプル。	
Appl.t.java	アプレットの使用法。
ストアド・プロシージャ - ストアド・プロシージャの例を示すサンプル。	
spcat	spserver プログラムのストアド・プロシージャ・カタログ・スクリプト。このスクリプトは、SpDrop.db2 と SpCreate.db2 を呼び出します。
SpCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
SpDrop.db2	カタログからストアド・プロシージャをドロップするための CLP スクリプト。
SpClient.java	SpServer.java 内で宣言されたサーバー関数の呼び出しで使用するクライアント・プログラム。

表 20. JDBC サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
SpServer.java	サーバー上で構築および実行されるストアード・プロシージャ関数。
UDF - ユーザー定義関数の例を示すサンプル。	
UDFcli.java	ユーザー定義関数ライブラリー UDFsrv を呼び出すクライアント・アプリケーション。
UDFsrv.java	UDFcli.java によって呼び出されるユーザー定義関数。
udfcat	UDFsrv プログラム用の UDF カタログ・スクリプト。このスクリプトは、UDFDrop.db2 と UDFCreate.db2 を呼び出します。
UDFDrop.db2	UDF をカタログからドロップするための CLP スクリプト。
UDFCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
UDFjcli.java	ユーザー定義関数ライブラリー UDFjsrv を呼び出すクライアント・アプリケーション。
UDFjsrv.java	UDFjcli.java によって呼び出されるユーザー定義関数。
udfjcat	UDFjsrv プログラム用の UDF カタログ・スクリプト。このスクリプトは、UDFjDrop.db2 と UDFjCreate.db2 を呼び出します。
UDFjDrop.db2	UDF をカタログからドロップするための CLP スクリプト。
UDFjCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
UDFsqlsv.java	UDFsqlsv.java に入っている UDF をカタログする方法。
UDFsqlsv.java	UDFsqlsv.java に入っている UDF をアンカタログする方法。
UDFsqlcl.java	UDFsqlsv.java 内の UDF を呼び出します。
UDFsqlsv.java	UDFsqlcl.java によって呼び出される SQL ステートメント呼び出しを備えたユーザー定義関数。
Java bean - Java bean クラスの例を示すサンプル。	
CreateEmployee.java	社員レコードの作成方法。
GeneratePayroll.java	部門別の給与計算レポートの生成方法。
その他	
Util.java	JDBC サンプル・プログラム用ユーティリティー。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 69 ページの『サンプル・ファイル』

関連資料:

- 94 ページの『SQLJ のサンプル』
- 96 ページの『Java WebSphere のサンプル』
- 96 ページの『Java プラグインのサンプル』

SQLJ のサンプル

UNIX ディレクトリ: sqllib/samples/java/sqlj。

Windows ディレクトリ: sqllib¥samples¥java¥sqlj。

表 21. SQLJ サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
チュートリアル・サンプル - データベースの基本操作の例を示すプログラム。	
TutMod.sqlj	表データの変更方法。
TutRead.sqlj	表の読み方。
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
DbAuth.sqlj	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
DbConn.sqlj	データベースからの接続および切断方法。
DbMCon.sqlj	複数のデータベースからの接続および切断方法。
DbUse.sqlj	データベース・オブジェクトの使用方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
TbConstr.sqlj	表の制約の処理方法。
TbCreate.sqlj	表の作成、変更、およびドロップ方法。
TbIdent.sqlj	ID 列の使用方法。
TbInfo.sqlj	表レベルの情報の入手および設定方法。
TbMod.sqlj	表内の情報の修正方法。
TbPriv.sqlj	表レベル特権の GRANT /表示/取り消しの方法。
TbRead.sqlj	表内の情報の読み取り方法。
TbSel.sqlj	挿入、更新、削除のそれぞれでの選択の方法。
TbTrig.sqlj	表でのトリガーの使用方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
DtUdt.sqlj	ユーザー定義特殊タイプの作成、使用、およびドロップ方法。
アプレット・レベル - アプレットの例を示すサンプル。	
Appl1t.sqlj	アプレットの使用方法。
ストアド・プロシージャ・レベル - ストアド・プロシージャを示すサンプル。	
spcat	SpServer プログラムのストアド・プロシージャ・カタログ・スクリプト。このスクリプトは、SpDrop.db2 と SpCreate.db2 を呼び出します。
SpCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
SpDrop.db2	カタログからストアド・プロシージャをドロップするための CLP スクリプト。
SpClient.sqlj	SpServer.sqlj 内で宣言されたサーバー関数の呼び出しで使用されるクライアント・プログラム。
SpServer.sqlj	サーバー上で構築および実行されるストアド・プロシージャ関数。
SpIterat.sqlj	SpServer.sqlj 用の ITERATOR クラス・ファイル。
UDF レベル - ユーザー定義関数を示すサンプル。	

表 21. SQLJ サンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
UDFcli.sqlj	ユーザー定義関数ライブラリー UDFsrv を呼び出すクライアント・アプリケーション。
UDFsrv.java	UDFcli によって呼び出されるユーザー定義関数。
udfcatsqlj	UDFsrv プログラム用の UDF カタログ・スクリプト。このスクリプトは、UDFDrop.db2 と UDFCreate.db2 を呼び出します。
UDFDrop.db2	UDF をカタログからドロップするための CLP スクリプト。
UDFCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
UDFjcli.sqlj	ユーザー定義関数ライブラリー UDFjsrv を呼び出すクライアント・アプリケーション。
UDFjsrv.java	UDFjcli によって呼び出されるユーザー定義関数。
udfjcatsqlj	UDFjsrv プログラム用の UDF カタログ・スクリプト。このスクリプトは、UDFjDrop.db2 と UDFjCreate.db2 を呼び出します。
UDFjDrop.db2	UDF をカタログからドロップするための CLP スクリプト。
UDFjCreate.db2	CREATE PROCEDURE ステートメントを発行するための CLP スクリプト。
Java bean - Java bean クラスの例を示すサンプル。	
CreateEmployee.sqlj	社員レコードの作成方法。
GeneratePayroll.sqlj	部門別の給与計算レポートの生成方法。
データ・ソース - データ・ソースの例を示すサンプル。	
Batch1Demo.sqlj	SQLJ バッチ -- SQLJ バッチの動作方法。
Batch2Demo.sqlj	SQLJ バッチ - ExecutionContext と BatchContext の関連。
Batch3Demo.sqlj	SQLJ バッチ - バッチを暗黙的に実行する場合。
BlobClobDemo.sqlj	DB2 表中の Blob または Clob フィールドにアクセスする方法。
createRegisterDS.java	DataSources プロパティ・ファイルの指定どおりのデータ・ソースの作成および登録。
CreateDemoSchema.sqlj	このプログラムは、データ・ソース・デモ・プログラムのスキーマを作成します。
DbConnDataSource.sqlj	DB2 Universal JDBC ドライバーでデータ・ソースを使用してデータベースに接続する方法。
DbConnDataSources.sqlj	DB2 Universal JDBC ドライバーで複数のデータ・ソースを使用してデータベースに接続する方法。
ScrollIterDemo.sqlj	SQLJ で名前付きおよび定位置スクロール可能イテレーターを使用する方法。
その他	
Util.sqlj	SQLJ サンプル・プログラム用ユーティリティ。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 69 ページの『サンプル・ファイル』

関連資料:

- 91 ページの『JDBC のサンプル』
- 96 ページの『Java WebSphere のサンプル』
- 96 ページの『Java プラグインのサンプル』

Java WebSphere のサンプル

UNIX ディレクトリ: `sql1lib/samples/java/WebSphere`。

Windows ディレクトリ: `sql1lib¥samples¥java¥WebSphere`。

表 22. Java WebSphere サンプル・ファイル

サンプル・プログラム名	プログラムの説明
AccessEmployee.ear	この Enterprise ARchive (.EAR) ファイルは、32 個の別々の .class、.JSP、および .HTML ファイルの入った 4 つのモジュールで構成されます。この EAR ファイルは、IBM WebSphere Application Server を使用して簡単に配置できます。Enterprise Java Beans (EJB) と対話して、DB2 に保管されているデータに Java クライアントからアクセスする方法がこのファイルによって示されます。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 69 ページの『サンプル・ファイル』

関連資料:

- 91 ページの『JDBC のサンプル』
- 94 ページの『SQLJ のサンプル』
- 96 ページの『Java プラグインのサンプル』

Java プラグインのサンプル

UNIX ディレクトリ: `sql1lib/samples/java/plugin`。

Windows ディレクトリ: `sql1lib¥samples¥java¥plugin`。

表 23. Java コントロール・センター・プラグインのサンプル・ファイル

サンプル・プログラム名	プログラムの説明
Example1.java	「コントロール・センター (Control Center)」ツールバーに新規のツールバー・ボタンを追加する方法。

表 23. Java コントロール・センター・プラグインのサンプル・ファイル (続き)

サンプル・プログラム名	プログラムの説明
Example2.java	コントロール・センター・データベース・オブジェクトに新規のメニュー・アクションを追加する方法。
Example3.java	コントロール・センター・ツリーのデータベース・オブジェクトの下に新規のオブジェクトを追加する方法。
Example3Child.java	コントロール・センター・ツリーのデータベース・オブジェクトの下にプラグイン・オブジェクトを追加する方法。
Example3Folder.java	コントロール・センター・ツリーのデータベース・オブジェクトの下に新規のオブジェクトを追加する方法。

関連概念:

- ・ 「管理ガイド: インプリメンテーション」の『コントロール・センター用のプラグイン・アーキテクチャーの紹介』
- ・ 「管理ガイド: インプリメンテーション」の『サンプル・プラグインのコンパイルおよび実行』
- ・ 125 ページの『Java サンプル・プログラム』
- ・ 69 ページの『サンプル・ファイル』
- ・ 「管理ガイド: インプリメンテーション」の『コントロール・センターの拡張機能としてのプラグインの作成』

関連タスク:

- ・ 「管理ガイド: インプリメンテーション」の『ツールバー・ボタンを追加するプラグインの作成』
- ・ 「管理ガイド: インプリメンテーション」の『プラグイン・ツリー・オブジェクトの属性の設定』

関連資料:

- ・ 91 ページの『JDBC のサンプル』
- ・ 94 ページの『SQLJ のサンプル』
- ・ 96 ページの『Java WebSphere のサンプル』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

ログ管理ユーザー出口サンプル

UNIX ディレクトリ: `sqllib/samples/c`。 Windows ディレクトリ: `sqllib\samples\c`。

注: ログ管理ユーザー出口プログラムのコンパイル方法の解説は、以下の表に一覧で示されている各ソース・ファイルの冒頭に述べられています。

表 24. ログ管理ユーザー出口のサンプル・プログラム・ファイル

サンプル・ファイル名	ファイルの説明
db2uext2.ctsm	<p>これは、Tivoli Storage Manager (TSM) API を使用してデータベース・ログ・ファイルの保存と検索を行うサンプル・ユーザー出口です。このサンプルは、タイム・スタンプと受け取ったパラメーターを含む呼び出しの監査証跡 (オプションごとに別々のファイルに保存) を提供します。また、エラーのタイム・スタンプとエラー分離ストリングを含め、エラーになった呼び出しの証跡も問題判別用として示します。これらのオプションは使用不能であることがあります。このファイルは、db2uext2.c に名前変更し、C プログラムとしてコンパイルしなければなりません。UNIX および Windows オペレーティング・システムで使用可能です。</p> <p>注: AIX で TSM API クライアントを使用するアプリケーションは、そのアプリケーションがシングル・スレッドであるとしても、xlc や xlc ではなく xlc_r もしくは xlc_r コンパイラ呼び出しで作成されている必要があります。これによりライブラリーがスレッド・セーフであることが保証されます。非スレッド・セーフ・ライブラリーでコンパイルされたアプリケーションをお持ちの場合は、フィックス・テスト IC21925E を適用するかアプリケーション・プロバイダーに連絡します。このフィックス・テストは、匿名 ftp サーバーの index.storsys.ibm.com で使用可能です。</p>
db2uext2.cdisk	<p>これは、出荷時の特定のプラットフォームに合わせた、システム・コピー・コマンドを使用するサンプル・ユーザー出口です。このプログラムは、データベース・ログ・ファイルを保存および検索し、タイム・スタンプと受け取ったパラメーターを含む呼び出しの監査証跡 (オプションごとに別々のファイルに保存) を提供します。また、問題判別のために、エラーのタイム・スタンプとエラー分離ストリングを含む呼び出しのエラー証跡も提供します。これらのオプションは使用不能であることがあります。このファイルは、db2uext2.c に名前変更し、C プログラムとしてコンパイルしなければなりません。UNIX および Windows オペレーティング・システムで使用可能です。</p>
db2uext2.ctape	<p>これは、出荷時の特定の UNIX プラットフォームに合わせた、システム・テープ・コマンドを使用するサンプル・ユーザー出口です。プログラムは、データベース・ログ・ファイルをアーカイブおよび検索します。システム・テープ・コマンドの制限すべてが、このユーザー出口の制限になります。このサンプルは、タイム・スタンプと受け取ったパラメーターを含む呼び出しの監査証跡 (オプションごとに別々のファイルに保存) を提供します。また、問題判別のために、エラーのタイム・スタンプとエラー分離ストリングを含む呼び出しのエラー証跡も提供します。これらのオプションは使用不能であることがあります。このファイルは、db2uext2.c に名前変更し、C プログラムとしてコンパイルしなければなりません。UNIX プラットフォームでのみ使用可能です。</p>
db2uext2.cxbsa	<p>これは、データベース・ログ・ファイルを保存して検索するために、XBSA API を使用するサンプル・ユーザー出口です。このサンプルは、タイム・スタンプと受け取ったパラメーターを含む呼び出しの監査証跡 (オプションごとに別々のファイルに保存) を提供します。また、問題判別のために、エラーのタイム・スタンプとエラー分離ストリングを含む呼び出しのエラー証跡も提供します。これらのオプションは使用不能であることがあります。このファイルは、db2uext2.c に名前変更し、C プログラムとしてコンパイルしなければなりません。UNIX プラットフォームでのみ使用可能です。</p>

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 「データ・リカバリーと高可用性 ガイドおよびリファレンス」の『ログ・アーカイブを使用したログ・ファイルの管理』
- 69 ページの『サンプル・ファイル』

関連資料:

- 「データ・リカバリーと高可用性 ガイドおよびリファレンス」の『Tivoli Storage Manager』

オブジェクトのリンクと埋め込み (OLE) のサンプル

ディレクトリは次のとおりです。 Visual Basic: sqllib\samples\ole\msvb。

Visual C++: sqllib\samples\ole\msvc。

表 25. オブジェクトのリンクと埋め込み (OLE) サンプル・プログラム

サンプル・プログラム名	プログラムの説明
sales	Microsoft Excel スプレッドシート上でのロールアップ照会を示します (Visual Basic で実現)。
names	Lotus Notes アドレス・ブックを照会します (Visual Basic で実現)。
inbox	OLE/メッセージング機能を使用する Microsoft Exchange インボックス E メール・メッセージを照会します (Visual Basic で実現)。
invoice	Microsoft Word インボイス文書を E メール接続として送信する OLE オートメーション・ユーザー定義関数 (Visual Basic で実現)。
bcounter	インスタンス変数を使用したスクラッチパッドのデモを示す OLE オートメーション・ユーザー定義関数 (Visual Basic で実現)。
ccount	カウンター OLE オートメーション・ユーザー定義関数 (Visual C++ で実現)。
salsrv	sample データベースの STAFF 表の給与の中央値を計算する OLE オートメーション・ストアード・プロシージャ (Visual Basic で実現)。
salcltvc	Visual Basic ストアード・プロシージャ、salsrv という Visual C++ DB2 CLI サンプル。
salcltvb	Visual Basic ストアード・プロシージャ salsrv を呼び出す Visual Basic DB2 CLI サンプル。
salsvado	32-bit Visual Basic および ADO でインプリメントされている OLE 自動ストアード・プロシージャは、新しく作成された表 STAFF2 内で給与の中央値を計算することによって出力パラメータを示し、さらにその表から給与を検索することによって結果セットを示します。
salclado	Visual Basic ストアード・プロシージャ salsvado を呼び出す Visual Basic クライアント。
testcli	ストアード・プロシージャ tstsrv を呼び出す OLE オートメーション組み込み SQL クライアント・アプリケーション (Visual C++ で実現)。

表 25. オブジェクトのリンクと埋め込み (OLE) サンプル・プログラム (続き)

サンプル・プログラム名	プログラムの説明
tstsrv	クライアントとストアード・プロシージャの間での各種の受け渡しのデモを示す OLE オートメーション・ストアード・プロシージャ (Visual C++ で実現)。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連資料:

- 100 ページの『オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数のサンプル』

オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数のサンプル

ディレクトリ: `sqllib\samples\oledb`。

表 26. オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数

サンプル・プログラム名	プログラムの説明
inora.db2	INTERSOLV Oracle8 OLE DB Provider
jet.db2	Microsoft.Jet.OLEDB.3.51 Provider
jetsrv.db2	Microsoft.Jet.OLEDB.4.0 Provider を使用するフェデレーテッド・データベース機能
mapi.db2	INTERSOLV Connect OLE DB for MAPI
msdaora.db2	Microsoft OLE DB Provider (Oracle 用)
msdsql.db2	Microsoft OLE DB Provider (ODBC ドライバー用)
msidxs.db2	Microsoft OLE DB Index Server Provider
notes.db2	INTERSOLV Connect OLE DB (ノーツ用)
sampprov.db2	Microsoft OLE DB Sample Provider
sqloledb.db2	Microsoft OLE DB Provider (SQL サーバー用)

関連概念:

- 69 ページの『サンプル・ファイル』

関連資料:

- 99 ページの『オブジェクトのリンクと埋め込み (OLE) のサンプル』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

Perl のサンプル

UNIX ディレクトリ: `sqllib/samples/perl`。

Windows ディレクトリ: `sqllib¥samples¥perl`。

表 27. Perl サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
<code>dbauth.pl</code>	データベース・レベルでの権限の <code>GRANT</code> /表示/取り消しの方法。
<code>dbuse.pl</code>	データベースの使用方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
<code>tbconstr.pl</code>	制約の作成、使用、およびドロップ方法。
<code>tbinfo.pl</code>	表に関する表レベルの情報の取得方法。
<code>tbpriv.pl</code>	表に関する特権の <code>GRANT</code> 、表示、および取り消しの方法。
<code>tbse1.pl</code>	挿入、更新、削除のそれぞれでの選択の方法。
<code>tbse1create.pl</code>	<code>tbse1</code> プログラム用の表の作成方法。
<code>tbse1drop.pl</code>	<code>tbse1</code> プログラム用の表のドロップ方法。
<code>tbtrig.pl</code>	表でのトリガーの使用方法。
<code>tbuse.pl</code>	データベースの基本操作実行、およびデータベースへの接続/データベースからの切断の方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
<code>dtlob.pl</code>	LOB データ・タイプの使用方法。
ストアド・プロシージャ - ストアド・プロシージャの例を示すサンプル。	
<code>spclient.pl</code>	ストアド・プロシージャを呼び出すための 10 個の関数が含まれるクライアント・プログラム。
他のサンプル・ファイル	
<code>DB2SampUtil.pm</code>	コマンド行の引き数の検査などの共通関数の定義。SQL ステートメントの準備と実行のための関数、およびエラーが発生した場合のロールバックのための関数も定義されています。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『Perl でのプログラミングに関する考慮事項』
- 69 ページの『サンプル・ファイル』

関連タスク:

- 161 ページの『Perl アプリケーションの構築』

PHP のサンプル

UNIX ディレクトリー: `sqllib/samples/php`。

Windows ディレクトリー: `sqllib¥samples¥php`。

表 28. PHP サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
dbauth.php	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
dbuse.php	データベースの使用方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
tbconstr.php	制約の作成、使用、およびドロップ方法。
tbinfo.php	表に関する表レベルの情報の取得方法。
tbpriv.php	表に関する特権の GRANT、表示、および取り消しの方法。
tbse1.php	挿入、更新、削除のそれぞれでの選択の方法。
tbse1create.db2	tbse1 プログラム用の表の作成方法。
tbse1drop.db2	tbse1 プログラム用の表のドロップ方法。
tbtrig.php	表でのトリガーの使用方法。
tbuse.php	データベースの基本操作実行、およびデータベースへの接続/データベースからの切断の方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
dtlob.php	LOB データ・タイプの使用方法。
ユーザー定義関数 - ユーザー定義関数の例を示すサンプル。	
udfcli.php	さまざまなタイプのユーザー定義関数を呼び出すクライアント・プログラム。
他のサンプル・ファイル	
util_funcs.php	コマンド行の引き数の検査などの共通関数の定義。SQL ステートメントの準備と実行のための関数、およびエラーが発生した場合のロールバックのための関数も定義されています。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 165 ページの『PHP アプリケーションの構築』

REXX のサンプル

AIX ディレクトリー: `sqllib/samples/rexx`。 Windows ディレクトリー: `sqllib¥samples¥rexx`。

表 29. REXX サンプル・プログラム・ファイル

サンプル・ ファイル名	ファイルの説明
blobfile.cmd	バイナリー・ラージ・オブジェクト (BLOB) の操作法の例を示します。
chgisl.cmd	CHANGE ISOLATION LEVEL API の例を示します。
client.cmd	SET CLIENT と QUERY CLIENT API の例を示します。
d_dbconf.cmd	GET DATABASE CONFIGURATION DEFAULTS の API の例を示します。
d_dbmcon.cmd	GET DATABASE MANAGER CONFIGURATION DEFAULTS の API の例を示します。
db_udcs.cmd	CREATE DATABASE と DROP DATABASE API の例を示して、DB2 for MVS/ESA CCSID 500 (EBCDIC International) の照合シーケンスの照合動作をシミュレートします。
dbauth.cmd	GET AUTHORIZATIONS API の例を示します。
dbcat.cmd	以下の API の例を示します。 CATALOG DATABASE CLOSE DATABASE DIRECTORY SCAN GET NEXT DATABASE DIRECTORY ENTRY OPEN DATABASE DIRECTORY SCAN UNCATALOG DATABASE
dbcmt.cmd	以下の API の例を示します。 CHANGE DATABASE COMMENT GET ERROR MESSAGE INSTALL SIGNAL HANDLER
dbconf.cmd	以下の API の例を示します。 CREATE DATABASE DROP DATABASE GET DATABASE CONFIGURATION RESET DATABASE CONFIGURATION UPDATE DATABASE CONFIGURATION
dbinst.cmd	以下の API の例を示します。 ATTACH TO INSTANCE DETACH FROM INSTANCE GET INSTANCE
dbmconf.cmd	以下の API の例を示します。 GET DATABASE MANAGER CONFIGURATION RESET DATABASE MANAGER CONFIGURATION UPDATE DATABASE MANAGER CONFIGURATION
dbstart.cmd	START DATABASE MANAGER API の例を示します。
dbstat.cmd	以下の API の例を示します。 REORGANIZE TABLE RUN STATISTICS
dbstop.cmd	以下の API の例を示します。 FORCE USERS STOP DATABASE MANAGER

表 29. REXX サンプル・プログラム・ファイル (続き)

サンプル・ファイル名	ファイルの説明
dcscat.cmd	以下の API の例を示します。 ADD DCS DIRECTORY ENTRY CLOSE DCS DIRECTORY SCAN GET DCS DIRECTORY ENTRY FOR DATABASE GET DCS DIRECTORY ENTRIES OPEN DCS DIRECTORY SCAN UNCATALOG DCS DIRECTORY ENTRY
dynamic.cmd	動的 SQL を使用する CURSOR の使用方法の例を示します。
ebcdicdb.cmd	CREATE DATABASE と DROP DATABASE API の例を示して、DB2 for MVS/ESA CCSID 037 (EBCDIC 米国英語) の照合シーケンスの照合動作をシミュレートします。
impexp.cmd	EXPORT および IMPORT API の例を示します。
lobeval.cmd	データベース内での LOB の評価の据え置き例を示します。
lobfile.cmd	LOB ファイル・ハンドルの使用例を示します
lobloc.cmd	LOB ロケーターの使用例を示します
lobval.cmd	LOB の使用例を示します
migrate.cmd	MIGRATE DATABASE API の例を示します。
nodecat.cmd	以下の API の例を示します。 CATALOG NODE CLOSE NODE DIRECTORY SCAN GET NEXT NODE DIRECTORY ENTRY OPEN NODE DIRECTORY SCAN UNCATALOG NODE
quitab.cmd	QUIESCE TABLESPACES FOR TABLE の API の例を示します。
rechist.cmd	以下の API の例を示します。 CLOSE RECOVERY HISTORY FILE SCAN GET NEXT RECOVERY HISTORY FILE ENTRY OPEN RECOVER HISTORY FILE SCAN PRUNE RECOVERY HISTORY FILE ENTRY UPDATE RECOVERY HISTORY FILE ENTRY
restart.cmd	RESTART DATABASE API の例を示します。
sqlcsrx.cmd	照合シーケンスの例。
updat.cmd	動的 SQL を使用してデータベースを更新します。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 221 ページの『AIX での REXX アプリケーションの構築』
- 328 ページの『Windows でのオブジェクト REXX アプリケーションの構築』

セキュリティ・プラグインのサンプル

UNIX ディレクトリー: `sqllib/samples/security/plugins`。

Windows ディレクトリー: `sqllib\samples\security\plugins`。

表 30. セキュリティ・プラグインのサンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
<code>combined.c</code>	複合ユーザー ID/パスワード認証およびグループ検索のサンプル。
<code>group_file.c</code>	単純ファイル・ベースのグループ管理プラグインのサンプル。
<code>gssapi_simple.c</code>	基本 GSS-API 認証プラグインのサンプル (クライアントとサーバーの両方)。
<code>IBMkrb5.c</code>	IBM 提供の Kerberos セキュリティ・プラグイン (UNIX 用) のソース・コード。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『セキュリティ・プラグイン』

SQL プロシージャのサンプル

UNIX ディレクトリー: `sqllib/samples/sqlproc`。

Windows ディレクトリー: `sqllib\samples\sqlproc`。

表 31. SQL プロシージャのサンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
<code>basecase.db2</code>	UPDATE_SALARY プロシージャは、"sample" データベースの "staff" 表内の "empno" の IN パラメーターによって識別される従業員の給料を上昇させます。このプロシージャは、"rating" の IN パラメーターを使用する CASE ステートメントによって上昇率を判別します。
<code>basecase.sqc</code>	UPDATE_SALARY プロシージャを呼び出します。
<code>baseif.db2</code>	UPDATE_SALARY_IF プロシージャは、"sample" データベースの "staff" 表内の "empno" の IN パラメーターによって識別される従業員の給料を上昇させます。このプロシージャは、"rating" の IN パラメーターを使用する IF ステートメントによって上昇率を判別します。
<code>baseif.sqc</code>	UPDATE_SALARY_IF プロシージャを呼び出します。
<code>dynamic.db2</code>	CREATE_DEPT_TABLE プロシージャは、動的 DDL を使用して新しい表を作成します。表の名前は、このプロシージャの IN パラメーターの値に基づいています。
<code>dynamic.sqc</code>	CREATE_DEPT_TABLE プロシージャを呼び出します。

表 31. SQL プロシーチャーのサンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
iterate.db2	ITERATOR プロシーチャーは、FETCH ループを使用して、 "department" 表からデータを検索します。 "deptno" 列の値が 'D11' で ない場合、修正されたデータが "department" 表の中に挿入されます。 "deptno" 列が 'D11' である場合、 ITERATE ステートメントは、LOOP ステートメントの冒頭に制御の流れを渡します。
iterate.sqc	ITERATOR プロシーチャーを呼び出します。
leave.db2	LEAVE_LOOP プロシーチャーは、 "not_found" 条件ハンドラーが LEAVE ステートメントを呼び出す前に LOOP ステートメント内で実行 される FETCH 操作の数をカウントします。 LEAVE ステートメント は、制御の流れがループから出て、ストアード・プロシーチャーを完了 するようにします。
leave.sqc	LEAVE_LOOP プロシーチャーを呼び出します。
loop.db2	LOOP_UNTIL_SPACE プロシーチャーは、カーソルが "midinit" 列のス ペース (' ') 値を持つ行を検索するまで、 LOOP ステートメント内で実 行される FETCH 操作の数をカウントします。 LOOP ステートメント は、制御の流れがループから出て、ストアード・プロシーチャーを完了 するようにします。
loop.sqc	LOOP_UNTIL_SPACE プロシーチャーを呼び出します。
nestcase.db2	BUMP_SALARY プロシーチャーは、ネストされた CASE ステートメン トを使用して、 "sample" データベースの "staff" 表から部署の IN パラ メーターによって識別される部署内の従業員の給料を上昇させます。
nestcase.sqc	BUMP_SALARY プロシーチャーを呼び出します。
nestif.db2	BUMP_SALARY_IF プロシーチャーは、ネストされた IF ステートメン トを使用して、 "sample" データベースの "staff" 表から部署の IN パラ メーターによって識別される部署内の従業員の給料を上昇させます。
nestif.sqc	BUMP_SALARY_IF プロシーチャーを呼び出します。
nestedsp.db2	OUT_AVERAGE、OUT_MEDIAN、および MAX_SALARY プロシージ ャーは、サンプル・データベースの "staff" 表中の平均値、中央値、およ び最大値を戻します。
nestedspdrop.db2	nestedsp.db2 スクリプトで作成される OUT_AVERAGE、 OUT_MEDIAN、および MAX_SALARY SQL プロシーチャーをドロッ プします。
NestedSP.java	OUT_AVERAGE プロシーチャーを呼び出します。
repeat.db2	REPEAT_STMT プロシーチャーは、カーソルが行を検索できなくなるま で、繰り返しステートメント内で実行される FETCH 操作の数をカウン トします。条件ハンドラーは、制御の流れが繰り返しループから出て、 ストアード・プロシーチャーを完了するようにします。
repeat.sqc	REPEAT_STMT プロシーチャーを呼び出します。
rsultset.c	MEDIAN_RESULT_SET プロシーチャーを呼び出し、給与の中央値を表 示し、SQL プロシーチャーによって生成された結果セットを表示しま す。このクライアントは、結果セットを扱える CLI API で書き出され ます。

表 31. SQL プロシーチャーのサンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
resultset.db2	MEDIAN_RESULT_SET プロシーチャーは、"sample" データベースの "staff" 表から "dept" の IN パラメーターによって識別される部署内の従業員の給与の中央値を入手します。給与の中央値は、給料の OUT パラメーターに割り当てられ、"resultset" クライアントに戻されます。次に、このプロシーチャーは、WITH RETURN カーソルをオープンし、中央値よりも高い給料の従業員の結果セットを戻します。このプロシーチャーは、クライアントに結果セットを戻します。
spserver.db2	この CLP スクリプト内の SQL プロシーチャーは、基本的なエラー処理、ネストされたストアド・プロシーチャーの呼び出し、クライアント・アプリケーションまたは呼び出し側アプリケーションへの結果セットの戻りを例示します。CLI サンプル・ディレクトリーで、"spcall" アプリケーションを使用してプロシーチャーを呼び出すことができます。C および CPP サンプル・ディレクトリーで、"spclient" アプリケーションを使用して、結果セットを戻さないプロシーチャーを呼び出すこともできます。
tbfn.db2	tbfnuse サンプルで使用される表および表関数を作成します。tbfnuse スクリプトの実行後は、すべての変更がロールバックされ、このファイルで作成された表および関数はドロップされます。
tbfnuse.db2	tbfnuse サンプルで作成される表関数の使用法を示します。このスクリプトの終わりにステートメントがロールバックされ、tbfn.db2 で作成された表および関数がドロップされます。
tbssel.sqc	挿入、更新、削除のそれぞれでの選択の方法。このサンプルは、tbsselcreate.db2 で作成される SQL プロシーチャー BUY_COMPANY を呼び出します。BUY_COMPANY には、データ変更ステートメントでの SELECT の使用例が含まれます。
tbsselcreate.db2	tbssel プログラムで使用される表およびプロシーチャーの作成方法。
tbsseldrop.db2	tbssel プログラムで使用される表およびプロシーチャーのドロップ方法。
whiles.db2	DEPT_MEDIAN プロシーチャーは、"sample" データベースの "staff" 表から "dept" の IN パラメーターによって識別される部署内の従業員の給与の中央値を入手します。給与の中央値は、給料の OUT パラメーターに割り当てられ、"whiles" クライアントに戻されます。次に、whiles クライアントは、給与の中央値を印刷します。
whiles.sqc	DEPT_MEDIAN プロシーチャーを呼び出します。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 153 ページの『SQL プロシーチャーの作成』

Visual Basic のサンプル

ディレクトリは次のとおりです。 ActiveX Data Object: sqllib\samples\VB\ADO。 Microsoft Transaction Server: sqllib\samples\VB\MTS。 Remote Data Object: sqllib\samples\VB\RD0。

表 32. Visual Basic ActiveX Data Object のサンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
クライアント・レベル	
cliExeSQL.bas	SQL ステートメントの実行方法。
cli_Info.bas	クライアント・レベル情報の取得/設定方法。
データベース・レベル	
dbConn.bas	データベースの接続および切断方法。
dbInfo.bas	データベース・レベルの情報の入手および設定方法。
dbCommit.bas	データベース・レベルでの自動コミットの動的コントロールの方法。
データ・タイプ・レベル	
dtHier.bas	階層データの検索方法。
dtLob.bas	LOB データの取得方法。
ストアド・プロシージャ	
spCall.bas	ストアド・プロシージャの呼び出し方法。
ユーザー定義関数	
udfUse.bas	UDT と UDF の作成とその処理の方法。

表 33. Visual Basic Microsoft Transaction Server のサンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
db2com.vbp	<p>この Visual Basic プロジェクトは、Microsoft Transaction Server を使用した、データベースの更新例を示します。また、クライアント・プログラム db2mts.vbp が使用するサーバー DLL を作成します。 Visual Basic プロジェクトには、以下の 4 つのクラス・モジュールがあります。</p> <ul style="list-style-type: none"> • UpdateNumberColumn.cls • UpdateRow.cls • UpdateStringColumn.cls • VerifyUpdate.cls <p>このプログラムの場合、一時表 DB2MTS が sample データベースに作成されます。</p>
db2mts.vbp	<p>これは、Microsoft Transaction Server を使用して、 db2com.vbp から作成されたサーバー DLL を呼び出す、クライアント・プログラム用の Visual Basic プロジェクトです。</p>
LCTransTest.vbp	<p>この Visual Basic プロジェクトは、DB2 データベースでの疎結合トランザクションの例を示します。また、クライアント・プログラム main.vbp が使用するサーバー DLL を作成します。このプロジェクトには、1 つのクラス・モジュール TestClass.cls があります。一時表 LCTEST がサンプル・データベースに作成されます。</p>

表 33. Visual Basic Microsoft Transaction Server のサンプル・プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
main.vbp	これは、クライアント・プログラム用の Visual Basic プロジェクトで、Microsoft Transaction Server を使用して LCTransTest.vbp から作成されたサーバー DLL を呼び出します。

表 34. Visual Basic Remote Data Object のサンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
Bank.vbp	カスタマー・アカウントでトランザクションを実行する機能を使用して、銀行の支店に関するデータを作成し、保守する RDO プログラム。このプログラムには、アプリケーションがデータを格納するために必要な表を作成する DDL が入っているので、ユーザーが指定する任意のデータベースに使用できます。

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 276 ページの『Visual Basic による ADO アプリケーションの構築』
- 283 ページの『Visual Basic による RDO アプリケーションの構築』
- 279 ページの『Visual Basic による疎結合トランザクションの構築』
- 281 ページの『Visual Basic 疎結合トランザクション・プロジェクトのトラブルシューティング』

関連資料:

- 112 ページの『Windows Management Instrumentation のサンプル』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

Visual Basic .NET のサンプル

ディレクトリ: sql1lib\samples\%.NET\%.vb。

表 35. サンプル Visual Basic .NET プログラム・ファイル

サンプル・プログラム名	プログラムの説明
データベース・レベル - DB2 内のデータベース・オブジェクトを扱うサンプル。	
DbAuth.vb	データベース・レベルでの権限の GRANT /表示/取り消しの方法。
DbDatAdp.vb	DB2DataAdapter の使用方法。

表 35. サンプル Visual Basic .NET プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
DbDatMap.vb	DataTable および DataColumn マッピングのセットアップ方法。
DbDsetCn.vb	既存の制約の DataSet への追加方法。
DbEvent.vb	DB2DataAdapter イベントの処理方法。
DbUse.vb	データベース・オブジェクトの使用方法。
DbValue.vb	データベースからの単一値の取得方法。
表レベル - DB2 内の表オブジェクトを扱うサンプル。	
TbConstr.vb	表の制約の処理方法。
TbInfo.vb	表レベルの情報の入手および設定方法。
TbPriv.vb	表レベル特権の GRANT /表示/取り消しの方法。
TbSel.vb	挿入、更新、削除のそれぞれでの選択の方法。
TbTrig.vb	表でのトリガーの使用方法。
TbUse.vb	表データの操作およびデータベースへの接続/データベースからの切断の方法。
データ・タイプ・レベル - データ・タイプを扱うサンプル。	
DtLob.vb	LOB データ・タイプの使用方法。
ストアド・プロシージャ - ストアド・プロシージャの例を示すサンプル。	
SpCat.db2	SpServer.vb にインプリメントされるプロシージャのドロップおよび作成。
SpClient.vb	SpServer.vb 内のストアド・プロシージャの呼び出しで使用するクライアント・プログラム。
SpCreate.db2	SpServer.vb にインプリメントされる外部プロシージャの作成。
SpDrop.db2	SpCreate.db2 に作成される外部プロシージャのドロップ。
SpReturn.vb	ストアド・プロシージャ EMP_DETAILS を呼び出して戻り値を取得するクライアント・アプリケーション。
SpServer.vb	SpCat.db2 で作成されるプロシージャの Visual Basic .NET 外部コード・インプリメンテーション。
EmpDetails.db2	EMP_DETAILS という名前のストアド・プロシージャを作成する CLP スクリプト。
ユーザー定義関数 - UDF の例を示すサンプル。	
UdfCat.db2	UdfSrv.vb にインプリメントされる外部 UDF のドロップおよび作成。
UdfCli.vb	UdfSrv.vb 内のユーザー定義関数を呼び出すクライアント・アプリケーション。
UdfCreate.db2	UdfSrv.vb にインプリメントされる外部 UDF の作成。
UdfDrop.db2	udfcreate.db2 に作成される外部 UDF のドロップ。
UdfSrv.vb	UdfCli によって呼び出されるユーザー定義スカラー関数。
疎結合トランザクション	
empcat.bat	クライアント・プログラム SpReturn 用ストアド・プロシージャ EMP_DETAILS のカタログ。
LCTrans.vb	疎結合トランザクションを示します。
regCOM.bat	LCTrans プログラム用 COM+ オブジェクトの登録。

表 35. サンプル Visual Basic .NET プログラム・ファイル (続き)

サンプル・プログラム名	プログラムの説明
RootCOM.vb	このファイルは、ライブラリー・アセンブリー RootCOM.dll を作成するのに使用します。 LCTrans.vb は、このファイル中に定義されているクラスと方式を参照します。
SubCOM.vb	このファイルは、ライブラリー・アセンブリー SubCOM.dll を作成するのに使用します。 LCTrans.vb は、このファイル中に定義されているクラスと方式を参照します。

関連概念:

- 69 ページの『サンプル・ファイル』

関連タスク:

- 289 ページの『Visual Basic .NET アプリケーションの構築』
- 293 ページの『Common Language Runtime (CLR) .NET ルーチンの構築』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

Visual C++ のサンプル

ディレクトリー: sql1lib\samples\VC\ADO。

表 36. Visual C++ サンプル・プログラム・ファイル

サンプル・プログラム名	プログラムの説明
BLOBAccess.dsw	このサンプルは、Microsoft Visual C++ を使用した ADO/Blob 強調表示アクセスを示します。これは Visual Basic サンプル、Blob.vbp と類似しています。 BLOB サンプルには次の 2 つの main 関数があります。 1. サンプル・データベースから BLOB を読み取り、画面に表示する 2. ファイルから BLOB を読み取り、データベースに挿入する (インポート)
VarCHAR.dsp	ADO を使用してテキスト・フィールドとして VarChar にアクセスする、 Visual C++ プログラム。グラフィカル・ユーザー・インターフェースを提供するので、ユーザーは sample データベースの ORG 表のデータを表示および更新できます。

関連概念:

- 300 ページの『Visual C++ でのオブジェクトのリンクと埋め込み (OLE) オートメーション』
- 69 ページの『サンプル・ファイル』

関連タスク:

- 298 ページの『Visual C++ による ADO アプリケーションの構築』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

Windows Management Instrumentation のサンプル

ディレクトリ: `sqllib\samples\wmi`。

表 37. *Windows Management Instrumentation* のサンプル・プログラム・ファイル

サンプル・ファイル名	ファイルの説明
<code>backupdb.vbs</code>	データベースのバックアップ方法。
<code>createdb.vbs</code>	データベースの作成とドロップの方法。
<code>listsvr.vbs</code>	サーバー・インスタンスの列挙の方法と DB2 インスタンスの開始と停止の方法。
<code>perfmon.mof</code>	<code>perfmon.vbs</code> 用の MOF ファイル。
<code>perfmon.vbs</code>	DB2 パフォーマンス・カウンターの入手方法。注: 先に <code>mofcomp perfmon.mof</code> を実行する必要があります。
<code>regvar.mof</code>	<code>regvar.vbs</code> 用の MOF ファイル。
<code>regvar.vbs</code>	DB2 レジストリー変数の入手方法。注: 先に <code>mofcomp regvar.mof</code> を実行する必要があります。
<code>restoredb.vbs</code>	データベースのリストア方法。
<code>rollfwd.db.vbs</code>	データベースのロールフォワード方法。
<code>updatedbcfg.vbs</code>	データベース構成の入手と更新の方法。
<code>updatedbmcfg.vbs</code>	データベース・マネージャー構成の入手と更新の方法。

関連概念:

- 69 ページの『サンプル・ファイル』
- 275 ページの『Windows Management Instrumentation (WMI)』

関連資料:

- 108 ページの『Visual Basic のサンプル』

サンプルの最新の更新事項については、次の DB2 アプリケーション開発サンプル Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/samples.html>

ビルド・ファイル、makefile、およびエラー・チェック・ユーティリティー

ビルド・ファイル

サンプル・プログラムの構築の例を示すのに使用するファイルは、UNIX® ではスクリプト・ファイル、Windows® ではバッチ・ファイルと呼ばれます。本書では、これらのファイルを総称してビルド・ファイルと呼ぶことにします。このファイルには、サポートされているプラットフォーム・コンパイラ用に推奨されるコンパイルとリンクのコマンドが入っています。

サポートされるプラットフォームでの各言語に対するビルド・ファイルが DB2® によって提供されています。そこには、各言語のサンプル・プログラムと同じディレクトリー内に、構築されるタイプのプログラムが入っています。以下の表に、さまざまなタイプのプログラムを構築するためのさまざまなタイプのビルド・ファイルを一覧で示してあります。他に明記されていない限り、これらのビルド・ファイルは、サポートされているすべてのプラットフォーム上のサポートされている言語用のファイルです。Windows ではこのビルド・ファイルには .bat (バッチ) の拡張子が付いていますが、この表では付けられていません。UNIX プラットフォームの場合には、拡張子はありません。

表 38. DB2 ビルド・ファイル

ビルド・ファイル	構築されるプログラムのタイプ
bldapp	アプリケーション・プログラム
bldrtn	ルーチン (ストアド・プロシージャと UDF)
bldsqlj	Java™ SQLJ アプリケーション
bldsqljs	Java SQLJ ルーチン (ストアド・プロシージャと UDF)
bldmc	C/C++ 複数接続アプリケーション
bldmt	C/C++ マルチスレッド・アプリケーション
bldcli	sqlproc サンプル・サブディレクトリー内の SQL プロシージャ用の CLI クライアント・アプリケーション

注: bldcli ファイルは、samples/cli ディレクトリー内の bldapp ファイルと同じです。これに別の名前が付いているのは、組み込み C の bldapp ファイルは samples/sqlproc ディレクトリーにも入っているからです。

以下の表は、プラットフォーム別およびプログラム言語別のビルド・ファイルと、それが置かれているディレクトリーを一覧で示しています。オンライン文書では、ビルド・ファイルの名前が HTML のソース・ファイルにホット・リンクされています。該当するサンプル・ディレクトリー内のテキスト・ファイルにアクセスすることもできます。

表 39. 言語別およびプラットフォーム別のビルド・ファイル

プラットフォーム → 言語	AIX®	HP-UX	Linux	Solaris	Windows
C samples/c	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp.bat bldrtn.bat bldmt.bat bldmc.bat
C++ samples/cpp	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp bldrtn bldmt bldmc	bldapp.bat bldrtn.bat bldmt.bat bldmc.bat
CLI samples/cli	bldapp bldrtn bldmc	bldapp bldrtn bldmc	bldapp bldrtn bldmc	bldapp bldrtn bldmc	bldapp.bat bldrtn.bat bldmc.bat
SQLJ samples/java/sqlj	bldsqlj bldsqljs	bldsqlj bldsqljs	bldsqlj bldsqljs	bldsqlj bldsqljs	bldsqlj.bat bldsqljs.bat
IBM® COBOL samples/cobol	bldapp bldrtn	n/a	n/a	n/a	bldapp.bat bldrtn.bat
Micro Focus COBOL samples/cobol_mf	bldapp bldrtn	bldapp bldrtn	bldapp bldrtn	bldapp bldrtn	bldapp.bat bldrtn.bat
C# samples¥.NET¥cs	n/a	n/a	n/a	n/a	bldapp.bat bldrtn.bat
Visual Basic .NET samples¥.NET¥vb	n/a	n/a	n/a	n/a	bldapp.bat bldrtn.bat

本書でアプリケーションとルーチンの構築にビルド・ファイルが使われているのは、DB2 がサポートされているコンパイラーに対しお勧めする、コンパイルおよびリンクのオプションが、それによって明らかに実体として示されるからです。通常はこれらの他にも使用できるコンパイルとリンクのオプションは多数あり、ユーザーはそれらを自由に試すことができます。用意されているコンパイルとリンクのすべてのオプションについて知りたい場合は、ご使用のコンパイラーのマニュアルを参照してください。開発者はそれらのビルド・ファイルを使用してサンプル・プログラムを構築するだけでなく、自分のプログラムを構築することも可能です。サンプル・プログラムをユーザーが変更できるテンプレートとして利用することにより、プログラム開発に役立てることが出来ます。

都合のよいことに、コンパイラーで許容される任意のファイル名でソース・ファイルを構築できるようにビルド・ファイルは設計されています。これは、プログラム名がファイル中にハードコーディングされる makefile とは異なります。makefile は、作成したプログラムのコンパイルとリンクのためにビルド・ファイルにアクセスします。ビルド・ファイルは、UNIX の場合には \$1 変数、Windows オペレーティング・システムの場合には %1 変数を使用して、プログラム名を内部的に置き換えます。このような変数名中の数字は、順に大きくなって、他の引き数が必要となるごとに入れ替わります。

たとえばスタンドアロン・アプリケーション、ルーチン (ストアド・プロシージャおよび UDF)、または複数接続あるいはマルチスレッド・プログラムなどのもっと特殊なプログラム・タイプといった、個々の種類のプログラム構築に対して個

々のビルド・ファイルがそれぞれ適応しているため、迅速かつ簡単にビルド・ファイルを試してみることができます。ビルド・ファイルの設計目的に沿った種類のプログラムがコンパイラーでサポートされてさえいれば、どこでもすべてのタイプのビルド・ファイルを利用することができます。

ビルド・ファイルが生成するオブジェクト・ファイルや実行可能ファイルは、ソース・ファイルが修正されない場合でさえ、プログラムが構築されるたびに自動的に上書きされます。それは、`makefile` を使用する場合には当てはまりません。つまり、開発者は以前のオブジェクト・ファイルや実行可能ファイルを削除したり、またはソースを修正したりすることなく、既存のプログラムを再構築することができます。

ビルド・ファイルには、サンプル・データベース用のデフォルト設定が組み込まれています。ユーザーが別のデータベースにアクセスする場合は、別のパラメーターを指定してデフォルトを上書きするだけで済みます。その別のデータベースを一貫して使用する予定であれば、ビルド・ファイルの中にある `sample` を置き換えて、このデータベースの名前をハードコーディングすることができます。

組み込み SQL プログラムでは、Windows で IBM COBOL プリコンパイラーを使用する場合を除き、ビルド・ファイルは、組み込み SQL プログラムのためのプリコンパイルとバインドのステップが入っている、別のファイル `embprep` を呼び出します。これらのステップでは、組み込み SQL プログラムをどこに構築するかによって、ユーザー ID とパスワード用のオプション・パラメーターが必要になる場合があります。

SQLJ の場合を除いて、データベースが置かれているサーバー・インスタンス上で開発者がプログラムを構築する場合は、ユーザー ID とパスワードはどちらにも共通であるため、指定する必要はありません。それに対して、開発者が別のインスタンスで作業する場合、たとえばサーバー・データベースにリモートからアクセスするクライアント・マシン上で作業する場合などは、これらのパラメーターを指定する必要があります。

SQLJ ビルド・ファイルの場合、ローカル・データベースへのアクセスであっても、`db2sqljcustomize` カスタマイザー用のユーザー ID とパスワードが必要です。これは、DB2 Universal JDBC ドライバーの規則に準拠しています。

最後の点として、ビルド・ファイルは開発者が自分の都合に合わせて修正することが可能です。開発者は (前述のように) ビルド・ファイル中のデータベース名を変更できるだけでなく、他のパラメーターをファイル内にハードコーディングしたり、コンパイルとリンクのオプションを変更したり、デフォルトの DB2 インスタンス・パスを変更したりすることが簡単に行えます。ビルド・ファイルはその性質上、簡単で分かりやすく、具体的であるため、自分の必要に応じてそれらのファイルに手を加えるのが容易です。

関連概念:

- 116 ページの『`makefile`』
- 119 ページの『エラー・チェック・ユーティリティー』
- 69 ページの『サンプル・ファイル』

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

makefile

サポートされているコンパイラ用の各サンプル・ディレクトリーには、付属のサンプル・プログラムの大半をディレクトリー内で構築するための **makefile** が入っています。 **makefile** はビルド・ファイルと呼び出して各プログラムをコンパイルおよびリンクします。 **makefile** の構文とそのコマンドからの出力は、ビルド・ファイルのものとはいくつかの重要な点で異なっています。ただし、以下のようにビルド・ファイルの「フロントエンド」として **makefile** を使用すれば、 **makefile** の簡略かつ有能なコマンドを活用することができます。

make <program_name>

指定したプログラムのコンパイルとリンクを実行します。

make all

makefile に記述されているすべてのプログラムのコンパイルとリンクを実行します。

make clean

makefile に記述されているすべてのプログラムの中間ファイル (オブジェクト・ファイルなど) を削除します。

make cleanall

makefile に記述されているすべてのプログラムの中間ファイルと実行可能ファイルをすべて削除します。

Java™ では通常は **makefile** を使用しないので、Java Developer Kit (JDK) には **make** 実行可能ファイルは付属されていません。ただし DB2® には、**make** コマンドの利便性を望むユーザーのために Java サンプル用のオプションとして **makefile** が装備されています。Java **makefile** を使用するには、通常は別の言語のコンパイラで使用する **make** 実行可能ファイルを使用可能にしておく必要があります。

以下に、DB2 に用意されているメインのプログラミング言語/API 用のプラットフォーム別の **makefile** と、それが置かれるサンプル・ディレクトリーを示してあります。オンライン文書ではこれらはホット・リンクされていて、これらが構築するサンプル・プログラムはそこにリンクされています。サンプル・ディレクトリー内のこれらのファイルにアクセスすることもできます。

表 40. プラットフォーム別のサンプル *makefile*

プラットフォーム → 言語	AIX®	HP-UX	Linux	Solaris	Windows®
C samples/c	makefile	makefile	makefile	makefile	makefile
C++ samples/cpp	makefile	makefile	makefile	makefile	makefile
CLI samples/cli	makefile	makefile	makefile	makefile	makefile
JDBC samples/java/jdbc	makefile	makefile	makefile	makefile	makefile
SQLJ samples/java/sqlj	makefile	makefile	makefile	makefile	makefile
IBM® COBOL samples/cobol	makefile	n/a	n/a	n/a	makefile
Micro Focus COBOL samples/cobol_mf	makefile	makefile	makefile	makefile	makefile
SQL プロシージャー samples/sqlproc	makefile	makefile	makefile	makefile	makefile
C# samples¥.NET¥cs	n/a	n/a	n/a	n/a	makefile
Visual Basic .NET samples¥.NET¥vb	n/a	n/a	n/a	n/a	makefile

ビルド・ファイルとは異なり、*makefile* はその中に記述されているプログラムの既存の中間ファイルと実行可能ファイルを上書きしません。*make all* コマンドを使用した場合、他のファイルに実行可能ファイルがすでにあると *make all* はそれらのファイルを単に無視するだけなので、いくつかのファイルの実行可能ファイルを作成するのであれば *makefile* による処理の方が高速です。ただし、既存のオブジェクト・ファイルと実行可能ファイルが不要な場合には、*make clean* および *make cleanall* コマンドを使用してそれらのファイルを除去しなければなりません。

makefile はプログラム開発に使用することができます。*makefile* では、ファイルそのもののなかでプログラム名をハードコーディングする必要があるため、ビルド・ファイルほど便利ではないと思われるかもしれませんが、*make* コマンドの有用性と利便性が必要な場合には検討対象とすることができます。

makefile は呼び出すプログラムを、変数によって表されるサーバー・クライアントとサーバー・プログラムのいくつかのカテゴリに分けて編成します (詳細は、*makefile* の項を参照してください)。プログラムを *makefile* に追加する場合、必ず正しい変数によってアクセスされるように追加してください。たとえば、任意のクライアント (サーバーに対してローカルまたはリモートのクライアント) で実行できるプログラムは、*client_run* 変数のもとに置かれます。

また、作成した実行可能ファイルを *make cleanall* コマンドで必ず削除できるように、*cleanall* 変数の下にプログラム名を指定する必要もあります。また、組み込み SQL プログラムの場合は、プリコンパイルの結果として作成された非組み込み

SQL ファイルを `clean` 変数の下に指定して、`make clean` コマンド (およびそれを呼び出す `make cleanall` コマンド) がその非組み込み SQL ファイルを削除するようにします。

さらに、正しい構文の新規のファイルを指定して、プログラムをコンパイルおよびリンクするのに適したビルド・ファイルを読み出す必要もあります。

いずれかのサンプルの `makefile` のどこに新規ファイルを追加する必要があるかを判断するために、以下に AIX C `makefile` 内での組み込み SQL プログラム `dbauth` が置かれているすべての場所を示してあります。

```
#####
#                               2f - make client_run
#####

client_run : ¥
    cli_info clisnap ¥
    dbauth dbconn dbcreate dbinfo dbmcon ¥
. . .
#####
#                               2g - make clean
#####

clean :
    $(ERASE) *.o
    $(ERASE) *.DEL *.TXT *.MSG
    $(ERASE) dbauth.c dbcfg.c dbconn.c dbmcon.c dbmcon1.c dbmcon2.c
. . .
#####
#                               2h - make cleanall
#####

cleanall : ¥
    clean
    $(ERASE) *.bnd
    $(ERASE) cli_info clisnap
    $(ERASE) dbauth dbcfg dbconn dbcreate dbinfo dbmcon dbmcon1 dbmcon2
. . .
#####
#                               3b - regular samples, embedded SQL
#####

dbauth :
    $(BLDAPP) dbauth $(ALIAS) $(UID) $(PWD)
```

上記の最終行中のプログラム名の後に続く `ALIAS`、`UID`、および `PWD` の 3 つの変数は、それぞれデータベースの別名、ユーザー ID、およびデータベースのパスワードを表します。これらの変数はビルド・ファイル (この場合は、`BLDAPP` 変数によって表される `bldapp` ビルド・ファイル) に渡されます。プログラムが組み込み SQL を使用する場合、`ALIAS`、`UID`、および `PWD` は、ビルド・ファイルが呼び出す `embprep` プリコンパイルおよびバインド・スクリプトに次々に渡されます。`makefile` を使用する前に、これらの変数の値を変更しなければならない場合があります。デフォルトでは、`ALIAS` は `sample` データベースに設定され、`UID` と `PWD` には値は設定されていません。

`UID` と `PWD` はオプション・パラメーターですが、ユーザーがすでにサーバー・データベースと同じインスタンスで作業している場合には設定する必要はありません。しかしそうではない場合に、たとえばユーザーがクライアント・マシンからサ

ーバーにリモート接続していれば、makefile を変更し、UID 変数と PWD 変数に正しい値を設定してデータベースにアクセスできるようにする必要があります。

複数接続のプログラムの場合、C、CLI、および C++ makefile には、ALIAS2 というもう 1 つ別のデータベース別名があります。これはデフォルトでは、sample2 データベースに設定されています。それに対応するユーザー ID とパスワードの変数 UID2 と PWD2 には値は設定されていません。UID と PWD 変数の場合と同様に、2 番目のデータベースにローカル・アクセスする場合はこれらの変数には値は必要ありません。

makefile はまた、make clean コマンドと make cleanall コマンドの呼び出し時にファイルを削除する ERASE 変数も定義します。これは UNIX® では rm -f に、Windows では del に設定されます。

関連概念:

- 112 ページの『ビルド・ファイル』
- 119 ページの『エラー・チェック・ユーティリティー』
- 69 ページの『サンプル・ファイル』

関連資料:

- 10 ページの『AIX でサポートされる開発ソフトウェア』
- 12 ページの『HP-UX でサポートされる開発ソフトウェア』
- 14 ページの『Linux でサポートされる開発ソフトウェア』
- 20 ページの『Solaris でサポートされる開発ソフトウェア』
- 23 ページの『Windows でサポートされる開発ソフトウェア』

エラー・チェック・ユーティリティー

DB2® AD クライアントは、いくつかのユーティリティー・ファイルを提供します。これらのファイルには、エラー・チェックとエラー情報の印刷出力を行う関数があります。ユーティリティー・ファイルは、サンプル・ディレクトリーの中に、言語ごとに別々のバージョンが用意されています。このユーティリティーはアプリケーション・プログラムで使用するときに有用なエラー情報を提供し、DB2 プログラムのデバッグの労力を大幅に軽減します。エラー・チェック・ユーティリティーのほとんどは、プログラム実行中に検出した問題に直接関連した SQLSTATE および SQLCA 情報を取得するのに、DB2 API GET SQLSTATE MESSAGE (sqllogstt) および GETERROR MESSAGE (sqlaintp) を使います。DB2 CLI ユーティリティー・ファイルである utilcli は、これらの DB2 API を使用する代わりに、それらと同じ働きをする DB2 CLI ステートメントを使用します。どのエラー・チェック・ユーティリティーを使用した場合でも詳細なエラー・メッセージが印刷出力されるため、開発者は短時間で問題を把握することができます。

ルーチンなどの一部の DB2 プログラム (ストアド・プロシージャやユーザー定義関数など) では、これらのユーティリティーを使用する必要はありません。例外が発生すると SQLException オブジェクトがスローされるので、このユーティリティーは Java™ にも不要です。

以下に示すのは、DB2 がサポートしているコンパイラーが使用する、プログラム言語別のエラー・チェック・ユーティリティー・ファイルです。

表 41. 言語別のエラー・チェック・ユーティリティー・ファイル

言語	非組み込み SQL ソース・フ ァイル	非組み込み SQL ヘッダー・ ファイル	組み込み SQL ソース・ ファイル	組み込み SQL ヘッダー・ ファイル
C samples/c	utilapi.c	utilapi.h	utilemb.sqc	utilemb.h
C++ samples/cpp	utilapi.C	utilapi.h	utilemb.sqC	utilemb.h
CLI samples/cli	utilcli.c	utilcli.h	n/a	n/a
IBM® COBOL samples/cobol	checkerr.cbl	n/a	n/a	n/a
Micro Focus COBOL samples/cobol_mf	checkerr.cbl	n/a	n/a	n/a

ユーティリティー関数を使用するには、まず最初にユーティリティー・ファイルをコンパイルした後、ターゲット・プログラムの実行可能ファイルの作成中にそのオブジェクト・ファイルをリンクしなければなりません。 samples ディレクトリー中の makefile とビルド・ファイルは両方とも、エラー・チェック・ユーティリティーを必要とするプログラムで使用するによってこの処理を行います。

以下の例は、エラー・チェック・ユーティリティーを DB2 プログラム中でどのように使用するかを示しています。 utilemb.h ヘッダー・ファイルは、関数 SqlInfoPrint() および TransRollback() 用の EMB_SQL_CHECK マクロを定義します。

```
/* macro for embedded SQL checking */
#define EMB_SQL_CHECK(MSG_STR)          ¥
SqlInfoPrint(MSG_STR, &sqlca, __LINE__, __FILE__);    ¥
if (sqlca.sqlcode < 0)                      ¥
{                                           ¥
    TransRollback();                      ¥
    return 1;                            ¥
}
```

SqlInfoPrint() は SQLCODE フラグをチェックします。この関数は、このフラグが示している特定のエラーに関連した、入手可能なすべての情報を印刷します。また、この関数は、ソース・コード内のどこでエラーが発生したかを示します。TransRollback() により、エラーが発生した場所にユーティリティー・ファイルがトランザクションを安全にロールバックできるようになります。データベースに接続し、ロールバックを実行するための組み込み SQL ステートメントが必要です。以下に、C プログラム dbuse がマクロを使用して、SqlInfoPrint() 関数の MSG_STR パラメーターに値 "Delete with host variables -- Execute" を提供することによって、ユーティリティー関数を呼び出す方法の例を示します。

```
EXEC SQL DELETE FROM org
      WHERE deptnumb = :hostVar1 AND
            division = :hostVar2;
EMB_SQL_CHECK("Delete with host variables -- Execute");
```

EMB_SQL_CHECK マクロは、DELETE ステートメントが失敗すると、トランザクションが安全にロールバックし、該当するエラー・メッセージが確実に印刷されるようにします。

開発者の方々には DB2 プログラムの作成時に、これらのエラー・チェック・ユーティリティを使用および拡張することをお勧めします。

関連概念:

- 112 ページの『ビルド・ファイル』
- 116 ページの『makefile』
- 69 ページの『サンプル・ファイル』

第 2 部 プラットフォーム非依存アプリケーションの構築および 実行

第 4 章 Java

Java サンプル・プログラム	125	UNIX の SQLJ アプリケーションおよびアプレット・オプション	139
Java アプレットに関する考慮事項	127	SQLJ アプリケーションおよびアプレットの Windows バッチ・ファイル	140
JDBC	129	Windows の SQLJ アプリケーションおよびアプレット・オプション	142
JDBC アプレットの構築	129	SQLJ ルーチンの構築	142
JDBC アプリケーションの構築	130	SQLJ ルーチンの UNIX ビルド・スクリプト	144
JDBC ルーチンの構築	131	UNIX の SQLJ ルーチン・オプション	145
SQLJ	134	SQLJ ルーチンの Windows バッチ・ファイル	146
SQLJ プログラムの構築	134	Windows の SQLJ ルーチン・オプション	148
SQLJ アプレットの構築	135		
SQLJ アプリケーションの構築	137		
SQLJ アプリケーションおよびアプレットの UNIX ビルド・スクリプト	138		

この章は、Java アプレットおよびアプリケーションを構築するための詳細な情報を提供します。DB2 Java アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad/v8/java>

Java サンプル・プログラム

動的 SQL と、静的 SQL を使用する SQLJ プログラムのみを使用する JDBC プログラムの構築と実行を例示するためのサンプル・プログラムが DB2[®] には用意されています。JDBC サンプルと SQLJ サンプルのディレクトリーは、java サンプル・ディレクトリーの下にそれぞれ別々に置かれています。以下に、UNIX[®] と Windows[®] オペレーティング・システムでの Java[™] サンプル・ディレクトリーの構造を示してあります。

- UNIX の場合:

- sqllib/samples/java**

- すべてのサブディレクトリー内の Java サンプル・プログラムの README ファイルが入っています。

- sqllib/samples/java/jdbc**

- JDBC サンプル・プログラム・ファイルが入っています。

- sqllib/samples/java/sqlj**

- SQLJ サンプル・プログラムが入っています。

- sqllib/samples/java/WebSphere**

- WebSphere サンプル・プログラムが入っています。

- sqllib/samples/java/plugin**

- DB2 コントロール・センターのプラグイン例のファイルが入っています。

- sqllib/samples/java/plugin/doc**

- プラグイン・インターフェース用の javadoc ファイルが入っています。

- Windows の場合;

sql1lib¥samples¥java

すべてのサブディレクトリー内の Java サンプル・プログラムの README ファイルが入っています。

sql1lib¥samples¥java¥jdbc

JDBC サンプル・プログラムが入っています。

sql1lib¥samples¥java¥sqlj

SQLJ サンプル・プログラムが入っています。

sql1lib¥samples¥java¥websphere

WebSphere サンプル・プログラムが入っています。

sql1lib¥samples¥java¥plugin

DB2 コントロール・センターのプラグイン例のファイルが入っています。

sql1lib¥samples¥java¥plugin¥doc

プラグイン・インターフェース用の javadoc ファイルが入っています。

SQLJ サンプル・ディレクトリーには、Java プログラム用の組み込み SQL を構築するためのビルド・ファイル (UNIX ではスクリプト、Windows ではバッチ・ファイル) が入っています。JDBC ディレクトリーにはビルド・ファイルは入っていません。javac を使用してコマンド行で JDBC プログラムを構築するのは非常に簡単なので、ビルド・ファイルは必要ないからです。

JDBC および SQLJ のどちらのサンプル・ディレクトリーにも、オプションの makefile が入っています。makefile は Java では広く使用されていません。したがって Java Development Kits (JDK) には make 実行可能ファイルは付属していません。ユーザーにとって便利のように、DB2 には Java サンプル makefile が用意されています。Java の各 makefile は、付属のすべてのサンプル・プログラムを JDBC または SQLJ サンプル・ディレクトリー内に構築します。別の言語コンパイラで使用するための gnumake などの make プログラムを使用することができます。

SQLJ アプレットとアプリケーションを構築するために、UNIX では bldsqlj、Windows では bldsqlj.bat、また、SQLJ ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するために、UNIX では bldsqljs、Windows では bldsqljs.bat という 2 つの SQLJ ビルド・ファイルが用意されています。

関連タスク:

- 31 ページの『Java 環境のセットアップ』
- 129 ページの『JDBC アプレットの構築』
- 130 ページの『JDBC アプリケーションの構築』
- 131 ページの『JDBC ルーチンの構築』
- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』
- 134 ページの『SQLJ プログラムの構築』

関連資料:

- 91 ページの『JDBC のサンプル』

- 94 ページの『SQLJ のサンプル』
- 96 ページの『Java WebSphere のサンプル』
- 96 ページの『Java プラグインのサンプル』

Java アプレットに関する考慮事項

DB2® には、Java™ アプレットを使用してアクセスすることができます。これを使用する際は、以下の点に注意してください。

1. すでに使用すべきでなくなったタイプ 3 ドライバー ('net' ドライバーとも呼びます) を使用している場合、Java アプレットで使用される db2java.zip ファイルが、JDBC アプレット・サーバーと同じフィックスパック・レベルにあることが不可欠です。通常の場合では、db2java.zip は、JDBC アプレット・サーバーが実行されている Web サーバーからロードされます。それによってレベルは必ず同一になります。ただし、構成が別のロケーションから db2java.zip をロードする Java アプレットを持っている場合は、不一致が起きることがあります。2 つのファイル間でのフィックスパック・レベルの一致は、接続時に厳しく強制されます。不一致が見つかり、接続はリジェクトされ、クライアントは以下の例外の 1 つを受け取ります。

- db2java.zip が DB2 バージョン 7 フィックスパック 2 以上の場合:

```
COM.ibm.db2.jdbc.DB2Exception: [IBM][JDBC Driver]
CLI0621E Unsupported JDBC server configuration.
```

- db2java.zip が DB2 バージョン 7 フィックスパック 2 より前の場合:

```
COM.ibm.db2.jdbc.DB2Exception: [IBM][JDBC Driver]
CLI0601E Invalid statement handle or statement is closed.
SQLSTATE=S1000
```

不一致が起きた場合、JDBC アプレット・サーバーは以下のメッセージの 1 つを jdbcerr.log ファイルに記録します。

- JDBC アプレット・サーバーが DB2 バージョン 7 フィックスパック 2 以降の場合:

```
jdbcFSQLConnect: JDBC Applet Server and client (db2java.zip)
versions do not match. Unable to proceed with connection., einfo= -111
```

- JDBC アプレット・サーバーが DB2 バージョン 7 フィックスパック 2 より前の場合:

```
jdbcServiceConnection(): Invalid Request Received., einfo= 0
```

注: sqllib¥samples¥java (Windows) または sqllib/samples/java (UNIX) の db2JDBCVersion.java サンプル・ファイルは、DB2 バージョン 8 で使用しないでください。このプログラムは DB2 バージョン 7 で使用可能で、使用中の DB2 JDBC ドライバーのバージョンと、JDBC 環境が正しくセットアップされているかを確認します。

アプレットを DB2 Universal JDBC ドライバーに移行することを強くお勧めします。

2. いくつかの Java クラスから成る大型の JDBC または SQLJ アプレットの場合は、そのクラスすべてを単一の JAR ファイルにパッケージ化するように選択できます。また、SQLJ アプレットでは、そのクラスに加えて、順番に並べられたプロファイルをパッケージしなければなりません。これを選択する場合、JAR ファイルを "applet" タグの archive パラメーターに追加します。詳細については、Java Developer Kit バージョン 1.3 の資料を参照してください。

SQLJ アプレットの場合、ブラウザによっては、アプレットに関連付けられたリソース・ファイルからのシリアル化ド・オブジェクトのロードをまだサポートしていません。たとえば、そのようなブラウザで付属のサンプル・アプレット Applet をロードしようとする、以下のようなエラー・メッセージが出されます。

```
java.lang.ClassNotFoundException: Applet_SJProfile0
```

これを回避する方法としては、シリアル化ド・プロファイルを Java クラス形式で保管するプロファイルに変換するユーティリティを使用することができます。このユーティリティは、sqlj.runtime.profile.util.SerProfileToClass という名前の Java クラスです。これはシリアル化ド・プロファイルのリソース・ファイルを入力として取り込み、そのプロファイルを含んだ Java クラスを出力として生成します。プロファイルは以下のいずれかのコマンドを使用して変換できます。

```
profconv Applet_SJProfile0.ser
```

または

```
java sqlj.runtime.profile.util.SerProfileToClass Applet_SJProfile0.ser
```

このコマンドの結果、クラス Applet_SJProfile0.class が作成されます。アプレットが使用している .ser 形式のすべてのプロファイルを .class 形式のプロファイルに置き換えれば、問題はなくなるはずです。

3. ファイル db2java.zip または db2jcc.jar あるいはその両方を、使用している Web サイトからロードされるいくつかのアプレットによって共用されるディレクトリーに置くことができます。db2java.zip は JDBC タイプ 3 ドライバーを使用するアプレット用で、db2jcc.jar は DB2 Universal JDBC ドライバーを使用するアプレットまたは SQLJ アプレット用です。これらのファイルは、Windows® オペレーティング・システムの場合には sql1lib\java ディレクトリーに、また UNIX® の場合には sql1lib/java ディレクトリーにあります。この場合、codebase パラメーターを、そのディレクトリーを識別する HTML ファイルの "applet" タグに追加する必要があることがあります。詳細については、Java Developer Kit バージョン 1.3 の資料を参照してください。
4. DB2 バージョン 5.2 以降、JDBC アプレット・サーバー (リスナー) である db2jd の機能を強化するために信号処理機能が追加されています。その結果、db2jd は Ctrl-C で強制終了することができません。したがって、「kill -9」(Unix の場合) または「タスク マネージャ」(Windows の場合) を使用してプロセスを強制終了するのが、リスナーを終了させる唯一の方法です。
5. Web サーバー、特に Domino™ Go Webserver での DB2 Java アプレットの実行についての詳細は、以下のサイトを参照してください。

<http://www.ibm.com/software/data/db2/db2lotus/gojava.htm>

関連タスク:

- 31 ページの『Java 環境のセットアップ』
- 129 ページの『JDBC アプレットの構築』
- 135 ページの『SQLJ アプレットの構築』

JDBC

JDBC アプレットの構築

Applet は、DB2 データベースにアクセスする動的 SQL Java アプレットの例を示します。

手順:

すでに使用すべきでなくなったタイプ 3 ドライバー (net ドライバーとも呼びます)、または Universal JDBC ドライバーのどちらを使用してもかまいません。どちらも DB2 Java Enablement とともにインストールされます。この 2 種類のドライバーでの接続については、この後の項に説明されています。アプレットを Universal JDBC ドライバーに移行することを強くお勧めします。

JDBC アプレット Applet を構築して、コマンド行にコマンドを入力して実行する場合、DB2 マシン (サーバーまたはクライアント) に Web サーバーがインストール済みで実行中であることを確認するか、または次のコマンドをクライアント・マシンの作業ディレクトリーで入力して、Java Development Kit に付属のアプレット・ビューアーを使用します。

```
appletviewer Applet.html
```

タイプ 3 (net) ドライバーとの接続

タイプ 3 ドライバーに接続するには、まず Applet.html ファイルを、ファイル内の指示に従って修正します。次に、Applet.html に指定されている TCP/IP ポートで JDBC アプレット・サーバーを開始します。たとえば Applet.html では、param name=port value='6789' と指定してから以下のように入力できます。

```
db2jstrtc 6789
```

接続ストリング内の JDBC ポート番号は、推奨デフォルトの "6789" にしてください。これは、番号が他のポート番号と競合しないと断定できる場合にだけ変更してください。データベースのポート番号 "50000" は使用しないでください。

Universal JDBC ドライバーとの接続

Universal JDBC ドライバーに接続するには、Applet.html ファイルを、ファイル内の指示に従って修正します。TCP/IP ポート番号には、データベース・ポート番号 "50000" を使用できます。

アプレットの構築

1. 以下に示すコマンドで Applet.java をコンパイルし、ファイル Applet.class を生成します。

```
javac Applet.java
```

2. Web ブラウザーが、作業ディレクトリーにアクセス可能であることを確認してください。アクセス可能ではない場合は、Applt.class と Applt.html をアクセス可能なディレクトリーにコピーします。
3. タイプ 3 ドライバーを使用する場合、Windows では sqllib¥java¥db2java.zip を、UNIX では sqllib/java/db2java.zip を、Applt.class および Applt.html と同じディレクトリーにコピーします。

Universal JDBC ドライバーを使用する場合、Windows では sqllib¥java¥db2jcc.jar を、UNIX では sqllib/java/db2jcc.jar を、Applt.class および Applt.html と同じディレクトリーにコピーします。

4. クライアント・マシンで、Web ブラウザー (Java 1.3 をサポートしていなければなりません) を開始し、Applt.html をロードします。

このプログラムは、Java の makefile を使用して構築することもできます。

関連概念:

- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 130 ページの『JDBC アプリケーションの構築』
- 131 ページの『JDBC ルーチンの構築』
- 135 ページの『SQLJ アプレットの構築』

関連資料:

- 91 ページの『JDBC のサンプル』

関連サンプル:

- 『Applt.java -- A Java applet that use JDBC applet driver to access a database (JDBC)』

JDBC アプリケーションの構築

DbInfo は、DB2 データベースにアクセスする動的 SQL Java アプリケーションの例を示します。

手順:

このアプリケーションを構築し実行するには、コマンド行で次のようにコマンドを入力します。

1. 以下に示すコマンドで DbInfo.java をコンパイルし、ファイル DbInfo.class を生成します。

```
javac DbInfo.java
```

2. 次のコマンドで、アプリケーションに対して Java インタープリターを実行します。

```
java DbInfo
```

このプログラムは、Java の makefile を使用して構築することもできます。

注: UNIX 上で 64 ビットの DB2 インスタンス内で Java アプリケーションを実行する場合に、Java Developer Kit が 32 ビットであると、アプリケーションを実行するにはまず DB2 ライブラリー・パスを変更する必要があります。AIX の場合はたとえば次のようにします。

- bash または Korn シェルを使用する場合:

```
export LIBPATH=$HOME/sql1lib/lib32
```

- C シェルを使用する場合:

```
setenv LIBPATH $HOME/sql1lib/lib32
```

関連タスク:

- 129 ページの『JDBC アプレットの構築』
- 131 ページの『JDBC ルーチンの構築』
- 137 ページの『SQLJ アプリケーションの構築』

関連資料:

- 91 ページの『JDBC のサンプル』

関連サンプル:

- 『DbInfo.java -- How to get/set info in a database (JDBC)』

JDBC ルーチンの構築

DB2 では、JDBC ルーチン (ストアド・プロシージャおよびユーザー定義関数) の例を示すサンプル・プログラムが、UNIX の場合は `samples/java/jdbc` ディレクトリーに、Windows の場合は `samples¥java¥jdbc` ディレクトリーに用意されています。ルーチンは、サーバー上でコンパイルされ保管します。そして、クライアント・アプリケーションによって呼び出されるとサーバー・データベースにアクセスし、そのクライアント・アプリケーションに情報を戻します。

手順:

以下の例は、ルーチン構成を構築する方法を示しています。

- ストアド・プロシージャ
- SQL ステートメントを使用しないユーザー定義関数
- SQL ステートメントを使用するユーザー定義関数

ストアド・プロシージャ

SpServer は、動的 SQL PARAMETER STYLE JAVA ストアド・プロシージャの例を示します。

コマンド行から、このプログラムを構築してサーバー上で実行するには、次のようにします。

1. 以下に示すコマンドで SpServer.java をコンパイルし、ファイル SpServer.class を生成します。

```
javac SpServer.java
```

2. SpServer.class を、Windows オペレーティング・システムの場合には sqllib/function ディレクトリーへ、また UNIX の場合には sqllib/function ディレクトリーへコピーします。
3. 次に、サーバーで spcat スクリプトを実行してルーチンをカタログします。次のように入力します。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば SpDrop.db2 を呼び出してルーチンをアンカタログし、次に SpCreate.db2 を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、SpDrop.db2 スクリプトと SpCreate.db2 スクリプトは、個別に実行することもできます。

4. 次に、データベースを一度停止してから再始動し、新しいクラス・ファイルが認識されるようにします。必要であれば、クラス・ファイルのファイル・モードを read に設定して、fenced ユーザーから読み取れるようにします。
5. SpClient クライアント・アプリケーションをコンパイルして実行し、ストアド・プロシージャ・クラスにアクセスします。

SQL ステートメントを使用しないユーザー定義関数

UDFsrv は、SQL ステートメントの入っていないユーザー定義関数ライブラリーです。DB2 には JDBC クライアント・アプリケーションである UDFcli と、UDFsrv ライブラリーにアクセスできる SQLJ クライアント・アプリケーションである UDFcli の両方が用意されています。

コマンド行から、UDF プログラムを構築してサーバー上で実行するには、次のようにします。

1. 以下に示すコマンドで UDFsrv.java をコンパイルし、ファイル UDFsrv.class を生成します。

```
javac UDFsrv.java
```

2. UDFsrv.class を、Windows オペレーティング・システムの場合には sqllib/function ディレクトリーへ、また UNIX の場合には sqllib/function ディレクトリーへコピーします。
3. UDFsrv ライブラリーへは、JDBC または SQLJ のいずれのクライアント・アプリケーションでもアクセスできます。どちらのバージョンのクライアント・プログラムにも、UDFsrv の中にある UDF をデータベースに登録するのに使用する CREATE FUNCTION SQL ステートメントと、登録後の UDF を利用するための SQL ステートメントが組み込まれています。

SQL ステートメントを使用するユーザー定義関数

UDFsqlsv は、SQL ステートメントの入っているユーザー定義関数ライブラリーです。DB2 には、UDFsqlsv ライブラリーにアクセスするための JDBC クライアント・アプリケーションである UDFsqlcl が用意されています。

コマンド行から、UDF プログラムを構築してサーバー上で実行するには、次のようにします。

1. 以下に示すコマンドで UDFsqlsv.java をコンパイルし、ファイル UDFsqlsv.class を生成します。

```
javac UDFsqlsv.java
```
2. UDFsqlsv.class を、Windows オペレーティング・システムの場合には sqllib¥function ディレクトリーへ、また UNIX の場合には sqllib/function ディレクトリーへコピーします。
3. UDFsqlsv ライブラリーにアクセスするには、UDFsqlsv の中にある UDF をデータベースに登録するのに使用する CREATE FUNCTION SQL ステートメントの入ったクライアント・プログラム UDFsqlcl を使用します。このクライアント・プログラムにはさらに、登録が済んだ UDF を利用するための SQL ステートメントも入っています。

上記のプログラムは、Java の makefile を使用して構築することもできます。

関連タスク:

- 129 ページの『JDBC アプレットの構築』
- 130 ページの『JDBC アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』

関連資料:

- 91 ページの『JDBC のサンプル』

関連サンプル:

- 『spcat -- To catalog SQLj stored procedures on UNIX』
- 『SpClient.java -- Call a variety of types of stored procedures from SpServer.java (JDBC)』
- 『SpCreate.db2 -- How to catalog the stored procedures contained in SpServer.java』
- 『SpDrop.db2 -- How to uncatalog the stored procedures contained in SpServer.java』
- 『SpServer.java -- Provide a variety of types of stored procedures to be called from (JDBC)』
- 『UDFccli.java -- Call the UDFs in UDFsrv.java (JDBC)』
- 『UDFCreate.db2 -- How to catalog the Java UDFs contained in UDFsrv.java』
- 『UDFDrop.db2 -- How to uncatalog the Java UDFs contained in UDFsrv.java』
- 『UDFsCreate.db2 -- How to catalog the UDFs contained in UDFsqlsv.java』
- 『UDFsDrop.db2 -- How to uncatalog the UDFs contained in UDFsqlsv.java』
- 『UDFsqlcl.java -- Call the UDFs in UDFsqlsv.java (JDBC)』
- 『UDFsqlsv.java -- Provide UDFs to be called by UDFsqlcl.java (JDBC)』
- 『UDFsrv.java -- Provide UDFs to be called by UDFccli.java (JDBC)』

SQLJ プログラムの構築

DB2 には、SQLJ サンプル・プログラムを構築するためのビルド・ファイルが備えられています。アプレットとアプリケーションの場合、UNIX 上では `bldsq1j` スクリプトを使用し、Windows 上では `bldsq1j.bat` バッチ・ファイルを使用することができます。ルーチン (ストアド・プロシージャとユーザー定義関数) の場合、UNIX 上では `bldsq1js` スクリプトを使用し、Windows 上では `bldsq1js.bat` バッチ・ファイルを使用することができます。

DB2 に付属している SQLJ 変換プログラムは Java コンパイラを呼び出して、変換後の `.java` ファイルを `.class` ファイルにコンパイルします。したがって、ビルド・ファイルは、いずれの場合も **sqlj** コマンドを使用します。

注:

1. 旧バージョンの DB2 では `db2profcc` コマンドは `-url=jdbc:db2:dbname` のフォームの URL を使用していました。ただし `dbname` は、ローカル・カタログされたデータベース別名です。新しい `db2sqljcustomize` コマンドは、DB2 Universal JDBC ドライバーの規則 `-url jdbc:db2://hostname:portnumber/dbname` に準じています。ここで `hostname` は DB2 サーバーの名前、`portnumber` は DB2 サーバーの TCP/IP Listener のポート番号、そして `dbname` は DB2 サーバーにカタログされたデータベースの別名です。つまり、DB2 サーバーは、TCP/IP 接続用に構成する必要があるということです。
2. 前のバージョンの `sqlj` コマンドを使用して変換された SQLJ プログラムは、DB2 バージョン 8 の `sqlj` コマンドを使用して再変換し、`db2sqljcustomize` コマンドを使用してカスタマイズする必要があります。
3. DB2 SQLJ プロファイル・プリンター `db2sqljprint` は、DB2 プロファイルの内容をプレーン・テキストで印刷します。

手順:

別のタイプの DB2 SQLJ プログラムを構築する場合は、以下を参照してください。

- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』

関連概念:

- 125 ページの『Java サンプル・プログラム』
- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』

関連資料:

- 「コマンド・リファレンス」の『db2sqljcustomize - DB2 SQLJ プロファイル・カスタマイザー・コマンド』
- 「コマンド・リファレンス」の『db2sqljprint - DB2 SQLJ プロファイル・プリンター・コマンド』
- 94 ページの『SQLJ のサンプル』
- 「コマンド・リファレンス」の『sqlj - DB2 SQLJ 変換プログラム・コマンド』

SQLJ アプレットの構築

以下のステップでは、DB2 データベースにアクセスする SQLJ アプレットの例を示す App1t サンプルの構築方法が示されています。このステップでは、ビルド・ファイル blsqlj (UNIX)、または blsqlj.bat (Windows) が使用されていますが、これには、SQLJ アプレットまたはアプリケーションを構築するためのコマンドが入っています。

ビルド・ファイルは、UNIX では \$1、\$2、\$3、\$4、\$5、および \$6、Windows では %1、%2、%3、%4、%5、および %6 の、最大 6 個のパラメーターをとります。最初のパラメーターは、プログラムの名前を指定します。2 番目のパラメーターはデータベース・インスタンスのユーザー ID を、3 番目のパラメーターはパスワードを指定します。4 番目のパラメーターは、サーバー名を指定します。5 番目のパラメーターはポート番号を指定します。そして 6 番目のパラメーターはデータベース名を指定します。最初のパラメーター (プログラム名) 以外のどのパラメーターでも、デフォルト値を使用することができます。デフォルトのパラメーター値の使用に関する詳細は、ビルド・ファイルを参照してください。

手順:

すでに使用すべきでなくなったタイプ 3 ドライバー (net ドライバーとも呼びます)、または Universal JDBC ドライバーのどちらを使用してもかまいません。どちらも DB2 Java Enablement とともにインストールされます。この 2 種類のドライバーでの接続については、この後の項に説明されています。アプレットを Universal JDBC ドライバーに移行することを強くお勧めします。

このアプレットを実行する場合、DB2 マシン (サーバーまたはクライアント) に Web サーバーがインストール済みで実行中であることを確認するか、または以下のコマンドをクライアント・マシンの作業ディレクトリーで入力して、Java Development Kit に付属のアプレット・ビューアーを使用することができます。

```
appletviewer Applt.html
```

タイプ 3 (net) ドライバーとの接続

タイプ 3 ドライバーに接続するには、まず Applt.html ファイルを、ファイル内の指示に従って修正します。次に、Applt.html に指定されている TCP/IP ポートで JDBC アプレット・サーバーを開始します。たとえば Applt.html では、`param name=port value='6789'` と指定してから以下のように入力できます。

db2jstrt 6789

接続ストリング内の JDBC ポート番号は、推奨デフォルトの "6789" にしてください。これは、番号が他のポート番号と競合しないと断定できる場合にだけ変更してください。データベースのポート番号 "50000" は使用しないでください。

Universal JDBC ドライバーとの接続

Universal JDBC ドライバーに接続するには、Applt.html ファイルを、ファイル内の指示に従って修正します。TCP/IP ポート番号には、データベース・ポート番号 "50000" を使用してください。

アプレットの構築

1. アプレットを以下のコマンドで構築します。

```
bldsqlj Applt <userid> <password> <server_name> <port_number> <db_name>
```

ただし、ビルド・ファイルに説明されているとおり、プログラム名以外のどのパラメーターにもデフォルト値を使用することができます。

2. Web ブラウザーから、またはアプレット・ビューアー (使用している場合) から、作業ディレクトリーにアクセスできることを確認してください。ディレクトリーにアクセスできない場合、アクセス可能なディレクトリーに以下のファイルをコピーします。

Applt.html	Applt.class
Applt_Cursor1.class	Applt_Cursor2.class
Applt_SJProfileKeys.class	Applt_SJProfile0.ser

3. タイプ 3 ドライバーを使用する場合、Windows では sqllib¥java¥db2jcc.jar および sqllib¥java¥db2java.zip を、UNIX では sqllib/java/db2jcc.jar および sqllib/java/db2java.zip を、Applt.class および Applt.html と同じディレクトリーにコピーします。

Universal JDBC ドライバーを使用する場合、Windows では sqllib¥java¥db2jcc.jar を、UNIX では sqllib/java/db2jcc.jar を、Applt.class および Applt.html と同じディレクトリーにコピーします。

4. クライアント・マシンで、Web ブラウザー (Java Developer Kit 1.3 をサポートしていなければなりません) または appletviewer を開始し、Applt.html をロードします。

このプログラムは、Java の makefile を使用して構築することもできます。

関連概念:

- 127 ページの『Java アプレットに関する考慮事項』

関連タスク:

- 129 ページの『JDBC アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』
- 142 ページの『SQLJ ルーチンの構築』

関連資料:

- 139 ページの『UNIX の SQLJ アプリケーションおよびアプレット・オプション』

- 142 ページの『Windows の SQLJ アプリケーションおよびアプレット・オプション』
- 94 ページの『SQLJ のサンプル』

関連サンプル:

- 『Applt.sqlj -- An SQLJ applet that uses a JDBC applet driver to access a database (SQLj)』
- 『blsqlj.bat -- Builds a Java embedded SQL (SQLJ) application or applet on Windows』
- 『blsqlj -- Builds Java embedded SQL (SQLJ) applications and applets on UNIX』

SQLJ アプリケーションの構築

以下のステップでは、DB2 データベースにアクセスする SQLJ アプリケーションの例を示す TbMod サンプルの構築方法が示されています。このステップでは、ビルド・ファイル blsqlj (UNIX)、または blsqlj.bat (Windows) が使用されていますが、これには、SQLJ アプレットまたはアプリケーションを構築するためのコマンドが入っています。

ビルド・ファイルは、UNIX では \$1、\$2、\$3、\$4、\$5、および \$6、Windows では %1、%2、%3、%4、%5、および %6 の、最大 6 個のパラメーターをとります。最初のパラメーターは、プログラムの名前を指定します。2 番目のパラメーターはデータベース・インスタンスのユーザー ID を、3 番目のパラメーターはパスワードを指定します。4 番目のパラメーターは、サーバー名を指定します。5 番目のパラメーターはポート番号を指定します。そして 6 番目のパラメーターはデータベース名を指定します。最初のパラメーター (プログラム名) 以外のどのパラメーターでも、デフォルト値を使用することができます。デフォルトのパラメーター値の使用に関する詳細は、ビルド・ファイルを参照してください。

手順:

ビルド・ファイル blsqlj (UNIX) または blsqlj.bat (Windows) を使用して TbMod を構築するには、以下のコマンドを入力します。

```
blsqlj TbMod <userid> <password> <server_name> <port_number> <db_name>
```

ただし、ビルド・ファイルに説明されているとおり、プログラム名以外のどのパラメーターにもデフォルト値を使用することができます。

次のコマンドで、アプリケーションに対して Java インタープリターを実行します。

```
java TbMod
```

このプログラムは、Java の makefile を使用して構築することもできます。

注: UNIX 上で 64 ビットの DB2 インスタンス内で Java アプリケーションを実行する場合に、Java Developer Kit が 32 ビットであると、アプリケーションを実行するにはまず DB2 ライブラリー・パスを変更する必要があります。AIX の場合はたとえば次のようにします。

- bash または Korn シェルを使用する場合:

```
export LIBPATH=$HOME/sql1lib/lib32
```

- C シェルを使用する場合:

```
setenv LIBPATH $HOME/sql1lib/lib32
```

関連タスク:

- 130 ページの『JDBC アプリケーションの構築』
- 135 ページの『SQLJ アプレットの構築』
- 142 ページの『SQLJ ルーチンの構築』

関連資料:

- 139 ページの『UNIX の SQLJ アプリケーションおよびアプレット・オプション』
- 142 ページの『Windows の SQLJ アプリケーションおよびアプレット・オプション』
- 94 ページの『SQLJ のサンプル』

関連サンプル:

- 『bldsqlj.bat -- Builds a Java embedded SQL (SQLJ) application or applet on Windows』
- 『bldsqlj -- Builds Java embedded SQL (SQLJ) applications and applets on UNIX』
- 『TbMod.sqlj -- How to modify table data (SQLj)』

SQLJ アプリケーションおよびアプレットの UNIX ビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldsqlj
# Builds Java embedded SQL (SQLJ) applications and applets on UNIX
# Usage: bldsqlj prog_name (requires hardcoding user ID and password)
#       bldsqlj prog_name userid password
#       bldsqlj prog_name userid password server_name
#       bldsqlj prog_name userid password server_name port_number
#       bldsqlj prog_name userid password server_name port_number db_name
#
# Defaults:
#   userid      = $USER variable requires updating if used
#   password    = $PSWD variable requires updating if used
#   server_name = $SERVER variable set to local hostname
#   port_number = $PORTNUM variable set to 50000
#   db_name     = $DB variable set to "sample"

# To hardcode user ID (USER) and password (PSWD)
# Replace "NULL" with the correct values in quotes
USER="NULL"
PSWD="NULL"
# You can replace the defaults for each of the following
# with a new value. Note that the PORTNUM number cannot
# be one already used by another process.
SERVER=`hostname`
PORTNUM=50000
DB="sample"

# Translate and compile the SQLJ source file
# and bind the package to the database.
if (( [ $# -eq 1 ] && [ $USER != "NULL" ] && [ $PSWD != "NULL" ] ) || ¥
    ( [ $# -ge 3 ] && [ $# -le 6 ] ) )
```

```

then
    # Remove .sqlj extension
    proname=${1%.sqlj}

    sqlj "${proname}.sqlj"

    if [ $# -eq 1 ]
    then
        db2sqljcustomize -url jdbc:db2://$SERVER:$PORTNUM/$DB ¥
        -user $USER -password $PSWD "${proname}_SJProfile0"
    elif [ $# -eq 3 ]
    then
        db2sqljcustomize -url jdbc:db2://$SERVER:$PORTNUM/$DB -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    elif [ $# -eq 4 ]
    then
        db2sqljcustomize -url jdbc:db2://$4:$PORTNUM/$DB -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    elif [ $# -eq 5 ]
    then
        db2sqljcustomize -url jdbc:db2://$4:$5/$DB -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    else
        db2sqljcustomize -url jdbc:db2://$4:$5/$6 -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    fi
else
    echo 'Usage: bldsqlj prog_name (requires hardcoding user ID and password)'
    echo '      bldsqlj prog_name userid password'
    echo '      bldsqlj prog_name userid password server_name'
    echo '      bldsqlj prog_name userid password server_name port_number'
    echo '      bldsqlj prog_name userid password server_name port_number db_name'
    echo ''
    echo '      Defaults:'
    echo '          userid      = '$USER
    echo '          password    = '$PSWD
    echo '          server_name = '$SERVER
    echo '          port_number = '$PORTNUM
    echo '          db_name     = '$DB
fi

```

UNIX の SQLJ アプリケーションおよびアプレット・オプション

以下の表は、UNIX 上の bldsqlj ビルド・スクリプトで使用する SQLJ 変換プログラム・オプションおよびカスタマイザー・オプションを示しています。これらは、UNIX プラットフォームでの SQLJ アプリケーションおよびアプレットの構築時に使用するようお勧めするオプションです。

bldsqlj の変換プログラム・オプションとカスタマイザー・オプション	
sqlj	SQLJ 変換プログラム (プログラムのコンパイルも行います)。 " \${progrname}.sqlj " SQLJ ソース・ファイル。 progrname=\${1%.sqlj} コマンドは、入力ファイル名に拡張子が含まれている場合にそれを除去するため、拡張子が付け直されるときに重複することはありません。
db2sqljcustomize	Java プロファイル・カスタマイザーが使用する DB2。
-url	データベース接続を確立するための JDBC URL を jdbc:db2://servername:50000/sample のように指定します。
-user	ユーザー ID を指定します。
-password	パスワードを指定します。
"\${progrname}_SJProfile0"	プログラムのシリアル化プロファイルを指定します。

関連タスク:

- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』

関連資料:

- 145 ページの『UNIX の SQLJ ルーチン・オプション』

関連サンプル:

- 『bldsqlj -- Builds Java embedded SQL (SQLJ) applications and applets on UNIX』

SQLJ アプリケーションおよびアプレットの Windows バッチ・ファイル

```

@echo off
rem BATCH FILE: bldsqlj.bat
rem Builds a Java embedded SQL (SQLJ) application or applet on Windows

rem To add defaults for user ID (USER) and password (PSWD)
rem Uncomment the following and add the appropriate values
rem set USR=
rem set PSWD=

rem You can replace the defaults for each of the following
rem with a new value. Note that the PORTNUM number cannot be
rem one already used by another process.
set SERVER=%COMPUTERNAME%
set PORTNUM=50000
set DB=sample

goto start
:usage
echo Usage: bldsqlj prog_name (requires hardcoding user ID and password)
echo          bldsqlj prog_name userid password
echo          bldsqlj prog_name userid password server_name
echo          bldsqlj prog_name userid password server_name port_number
echo          bldsqlj prog_name userid password server_name port_number db_name

```

```

echo.
echo      Defaults:
echo      userid      = %USR%
echo      password    = %PSWD%
echo      server_name = %SERVER%
echo      port_number = %PORTNUM%
echo      db_name     = %DB%
goto exit

:start
rem Translate and compile the SQLJ source file
rem and bind the package to the database.
if "%1" == "" goto usage
if "%2" == "" goto case1
if "%3" == "" goto usage
if "%4" == "" goto case3
if "%5" == "" goto case4
if "%6" == "" goto case5
if "%7" == "" goto case6
goto usage

:case1
if "%USR%" == "" goto usage
if "%PSWD%" == "" goto usage
if "%SERVER%" == "" goto nohostname
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%SERVER%:%PORTNUM%/%DB% -user %USR%
                  -password %PSWD% %1_SJProfile0
goto continue

:case3
if "%SERVER%" == "" goto nohostname
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%SERVER%:%PORTNUM%/%DB% -user %2
                  -password %3 %1_SJProfile0
goto continue

:case4
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%PORTNUM%/%DB% -user %2
                  -password %3 %1_SJProfile0
goto continue

:case5
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%5/%DB% -user %2
                  -password %3 %1_SJProfile0
goto continue

:case6
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%5/%6 -user %2
                  -password %3 %1_SJProfile0
goto continue

:continue
goto exit

:nohostname
echo Server name (hostname) could not be determined.
echo.
goto usage

:exit
@echo on

```

Windows の SQLJ アプリケーションおよびアプレット・オプション

以下の表は、Windows オペレーティング・システム上の bldsqlj.bat バッチ・ファイルで使用する SQLJ 変換プログラム・オプションおよびカスタマイザー・オプションを示しています。これらは、Windows での SQLJ アプリケーションおよびアプレットの構築時に使用するようお勧めするオプションです。

bldsqlj.bat の変換プログラム・オプションとカスタマイザー・オプション	
sqlj	SQLJ 変換プログラム (プログラムのコンパイルも行います)。
%1.sqlj	SQLJ ソース・ファイル。
db2sqljcustomize	Java プロファイル・カスタマイザーが使用する DB2。
-url	データベース接続を確立するための JDBC URL を jdbc:db2://servername:50000/sample のように指定します。
-user	ユーザー ID を指定します。
-password	パスワードを指定します。
%1_SJProfile0	プログラムのシリアル化プロファイルを指定します。

関連タスク:

- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』

関連資料:

- 148 ページの『Windows の SQLJ ルーチン・オプション』

関連サンプル:

- 『bldsqlj.bat -- Builds a Java embedded SQL (SQLJ) application or applet on Windows』

SQLJ ルーチンの構築

DB2 では、SQLJ ルーチン (ストアド・プロシージャおよびユーザー定義関数) の例を示すサンプル・プログラムが、UNIX の場合は samples/java/sqlj ディレクトリーに、Windows の場合は samples%java%sqlj ディレクトリーに用意されています。ルーチンは、サーバー上でコンパイルされ保管されます。そして、クライアント・アプリケーションによって呼び出されるとサーバー・データベースにアクセスし、そのクライアント・アプリケーションに情報を戻します。

また DB2 では、ルーチンを構築するためのコマンドの入ったビルド・ファイル bldsqlj (UNIX)、または bldsqlj.bat (Windows) も同じディレクトリー内に用意されています。

ビルド・ファイルは、UNIX では \$1、\$2、\$3、\$4、\$5、および \$6、Windows では %1、%2、%3、%4、%5、および %6 の、最大 6 個のパラメーターをとります。最初のパラメーターは、プログラムの名前を指定します。2 番目のパラメーターはデータベース・インスタンスのユーザー ID を、3 番目のパラメーターはパスワードを指定します。4 番目のパラメーターは、サーバー名を指定します。5 番目のパラメーターはポート番号を指定します。そして 6 番目のパラメーターはデータベース名を指定します。最初のパラメーター (プログラム名) 以外のどのパラメーターでも、デフォルト値を使用することができます。デフォルトのパラメーター値の使用に関する詳細は、ビルド・ファイルを参照してください。

手順:

以下の例は、ストアード・プロシージャを使用してクラス・ファイルを構築する方法を示しています。

SpServer は、DB2 データベースにアクセスするために、JDBC アプリケーション・ドライバを使用した PARAMETER STYLE JAVA ストアード・プロシージャの例を示します。

このストアード・プロシージャ・クラスをビルド・ファイル bldsqljs (UNIX) または bldsqljs.bat (Windows) で構築するには、以下に示すコマンドを入力します。

1. 次のコマンドを入力します。

```
bldsqljs SpServer <userid> <password> <server_name> ¥  
          <port_number> <db_name>
```

ただし、ビルド・ファイルに説明されているとおり、プログラム名以外のどのパラメーターにもデフォルト値を使用することができます。

2. 次に、サーバーで spcat スクリプトを実行してルーチンをカタログします。次のように入力します。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば SpDrop.db2 を呼び出してルーチンをアンカタログし、次に SpCreate.db2 を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、SpDrop.db2 スクリプトと SpCreate.db2 スクリプトは、個別に実行することもできます。

3. 次に、データベースを一度停止してから再始動し、新しいクラス・ファイルが認識されるようにします。必要であれば、クラス・ファイルのファイル・モードを read に設定して、fenced ユーザーから読み取れるようにします。
4. SpClient クライアント・アプリケーションを構築して実行し、ストアード・プロシージャを呼び出します。アプリケーションのビルド・ファイル bldsqlj (UNIX) または bldsqlj.bat (Windows) を使用して、SpClient を構築することができます。

上記のプログラムは、Java の makefile を使用して構築することもできます。

関連タスク:

- 131 ページの『JDBC ルーチンの構築』

- 135 ページの『SQLJ アプレットの構築』
- 137 ページの『SQLJ アプリケーションの構築』

関連資料:

- 145 ページの『UNIX の SQLJ ルーチン・オプション』
- 148 ページの『Windows の SQLJ ルーチン・オプション』
- 94 ページの『SQLJ のサンプル』

関連サンプル:

- 『bldsqljs.bat -- Builds a Java embedded SQL (SQLJ) stored procedure on Windows』
- 『bldsqljs -- Builds Java embedded SQL (SQLJ) stored procedures on UNIX』
- 『spcat -- To catalog SQLj stored procedures on UNIX』
- 『SpClient.sqlj -- Call a variety of types of stored procedures from SpServer.sqlj (SQLj)』
- 『SpCreate.db2 -- How to catalog the stored procedures contained in SpServer.sqlj』
- 『SpDrop.db2 -- How to uncatalog the stored procedures contained in SpServer.sqlj』
- 『SpIterat.sqlj -- Iterator class file for SpServer.sqlj (SQLj)』
- 『SpServer.sqlj -- Provide a variety of types of stored procedures to be called from (SQLj)』

SQLJ ルーチンの UNIX ビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldsqljs
# Builds Java embedded SQL (SQLJ) stored procedures on UNIX
# Usage: bldsqljs prog_name (requires hardcoding user ID and password)
#       bldsqljs prog_name userid password
#       bldsqljs prog_name userid password server_name
#       bldsqljs prog_name userid password server_name port_number
#       bldsqljs prog_name userid password server_name port_number db_name
#
# Defaults:
#       userid      = $USER variable requires updating if used
#       password    = $PSWD variable requires updating if used
#       server_name = $SERVER variable set to local hostname
#       port_number = $PORTNUM variable set to 50000
#       db_name     = $DB variable set to "sample"

# To hardcode user ID (USER) and password (PSWD)
# Replace "NULL" with the correct values in quotes
USER="NULL"
PSWD="NULL"

# You can replace the defaults for each of the following
# with a new value. Note that the PORTNUM number cannot
# be one already used by another process.
SERVER=`hostname`
PORTNUM=50000
DB="sample"

# Translate and compile the SQLJ source file
# and bind the package to the database.
if (( [ $# -eq 1 ] && [ $USER != "NULL" ] && [ $PSWD != "NULL" ] ) ) {
    || ( [ $# -ge 3 ] && [ $# -le 6 ] )
}
```

```

then
    # Remove .sqlj extension
    proname=${1%.sqlj}

    sqlj "${proname}.sqlj"

    if [ $# -eq 1 ]
    then
        db2sqljcustomize -url jdbc:db2://$SERVER:$PORTNUM/$DB ¥
        -user $USER -password $PSWD "${proname}_SJProfile0"
    elif [ $# -eq 3 ]
    then
        db2sqljcustomize -url jdbc:db2://$SERVER:$PORTNUM/$DB -user $2 ¥
        -password $3 "${proname}_SJProfile0"
    elif [ $# -eq 4 ]
    then
        db2sqljcustomize -url jdbc:db2://$4:$PORTNUM/$DB -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    elif [ $# -eq 5 ]
    then
        db2sqljcustomize -url jdbc:db2://$4:$5/$DB -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    else
        db2sqljcustomize -url jdbc:db2://$4:$5/$6 -user $2 -password $3 ¥
        "${proname}_SJProfile0"
    fi

    # Copy the *.class and *.ser files to the 'function' directory.
    rm -f "$DB2PATH/function/${proname}*.class"
    rm -f "$DB2PATH/function/${proname}*.ser"
    cp "${proname}*.class" "$DB2PATH/function"
    cp "${proname}*.ser" "$DB2PATH/function"

else
    echo 'Usage: bldsqljs prog_name (requires hardcoding user ID and password)'
    echo '      bldsqljs prog_name userid password'
    echo '      bldsqljs prog_name userid password server_name'
    echo '      bldsqljs prog_name userid password server_name port_number'
    echo '      bldsqljs prog_name userid password server_name port_number db_name'
    echo ''
    echo '      Defaults:'
    echo '      userid      = '$USER
    echo '      password    = '$PSWD
    echo '      server_name = '$SERVER
    echo '      port_number = '$PORTNUM
    echo '      db_name     = '$DB
fi

```

UNIX の SQLJ ルーチン・オプション

以下の表は、UNIX 上の bldsqljs ビルド・スクリプトで使用する SQLJ 変換プログラム・オプションおよびカスタマイザー・オプションを示しています。これらは、UNIX プラットフォームでの SQLJ ルーチン (ストアド・プロシージャとユーザー定義関数) の構築時に使用するようお勧めするオプションです。

bldsqljs の変換プログラム・オプションとカスタマイザー・オプション	
sqlj	SQLJ 変換プログラム (プログラムのコンパイルも行います)。 " \${progrname}.sqlj " SQLJ ソース・ファイル。 progrname=\${1%.sqlj} コマンドは、入力ファイル名に拡張子が含まれている場合にそれを除去するため、拡張子が付け直されるときに重複することはありません。
db2sqljcustomize	Java プロファイル・カスタマイザーが使用する DB2。
-url	データベース接続を確立するための JDBC URL を jdbc:db2://servername:50000/sample のように指定します。
-user	ユーザー ID を指定します。
-password	パスワードを指定します。
"\${progrname}_SJProfile0"	プログラムのシリアル化プロファイルを指定します。

関連タスク:

- 142 ページの『SQLJ ルーチンの構築』

関連資料:

- 139 ページの『UNIX の SQLJ アプリケーションおよびアプレット・オプション』

関連サンプル:

- 『bldsqljs -- Builds Java embedded SQL (SQLJ) stored procedures on UNIX』

SQLJ ルーチンの Windows バッチ・ファイル

```
@echo off
rem BATCH FILE: bldsqljs.bat
rem Builds a Java embedded SQL (SQLJ) stored procedure on Windows

rem To add defaults for user ID (USR) and password (PSWD)
rem Uncomment the following and add the appropriate values
rem set USR=
rem set PSWD=
rem You can replace the defaults for each of the following
rem with a new value. Note that the PORTNUM number cannot be
rem one already used by another process.
set SERVER=%COMPUTERNAME%
set PORTNUM=50000
set DB=sample

goto start
:usage
echo Usage: bldsqljs prog_name (requires hardcoding user ID and password)
echo          bldsqljs prog_name userid password
echo          bldsqljs prog_name userid password server_name
echo          bldsqljs prog_name userid password server_name port_number
echo          bldsqljs prog_name userid password server_name port_number db_name
echo.
echo          Defaults:
echo          userid      = %USR%
echo          password     = %PSWD%
```

```

echo          server_name = %SERVER%
echo          port_number = %PORTNUM%
echo          db_name     = %DB%
goto exit

:start
rem Translate and compile the SQLJ source file
rem and bind the package to the database.
if "%DB2PATH%" == "" goto nodb2cmd
if "%1" == "" goto usage
if "%2" == "" goto case1
if "%3" == "" goto usage
if "%4" == "" goto case3
if "%5" == "" goto case4
if "%6" == "" goto case5
if "%7" == "" goto case6
goto usage

:case1
if "%USR%" == "" goto usage
if "%PSWD%" == "" goto usage
if "%SERVER%" == "" goto nohostname
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%SERVER%:%PORTNUM%/%DB% -user %USR%
                -password %PSWD% %1_SJProfile0
goto continue

:case3
if "%SERVER%" == "" goto nohostname
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%SERVER%:%PORTNUM%/%DB% -user %2
                -password %3 %1_SJProfile0
goto continue

:case4
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%PORTNUM%/%DB% -user %2
                -password %3 %1_SJProfile0
goto continue

:case5
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%5/%DB% -user %2
                -password %3 %1_SJProfile0
goto continue

:case6
sqlj %1.sqlj
db2sqljcustomize -url jdbc:db2://%4:%5/%6 -user %2
                -password %3 %1_SJProfile0
goto continue

:continue
rem Copy the *.class and *.ser files to the 'function' directory.
copy %1*.class %DB2PATH%¥function¥
copy %1*.ser %DB2PATH%¥function¥
goto exit

:nodb2cmd
echo DB2 command line environment not initialized. Please run db2cmd and try again.
goto exit

:nohostname
echo Server name (hostname) could not be determined.
echo.

```

148 アプリケーションの構築および実行

第 5 章 コマンド行プロセッサ

コマンド行プロセッサ (CLP) スクリプトの実行 149

コマンド行プロセッサ (CLP) からのプロシージャの呼び出し 150

この章では、コマンド行プロセッサを使用して CLP スクリプトを実行し、コマンド行で CALL ステートメントを使って DB2 ストアド・プロシージャを呼び出して DB2 データベースにアクセスする方法を詳細に説明します。

SQL プロシージャのコードも、CLP スクリプトに含まれます。SQL プロシージャの作成については、153 ページの『第 6 章 SQL プロシージャ』に記載しています。

DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

コマンド行プロセッサ (CLP) スクリプトの実行

CLP は、DB2 インスタンスのコマンド行から直接アクセスすることができます。コマンド行に「db2」と入力することで、対話式に実行することができます。また、キーワード「db2」の後に DB2 コマンドを入力するか、DB2 コマンドを含む DB2 スクリプトを入力することで、非対話式モードで実行することができます。ここで示す例は、非対話式モードのものです。

CLP スクリプトのコードは、ホスト言語で SQL ステートメントを組み込むのではなく、SQL ステートメントそのままであるため、CLP スクリプトのプログラミングに必要なのは SQL を理解することだけです。さらに、CLP スクリプトの場合、スクリプトを実行するためにソース・ファイルをコンパイルしたりリンクしたりする必要がありません。

DB2 には、sample データベースに対する SQL ステートメントの作成および実行のための CLP スクリプトが用意されています。これらは、UNIX の場合は `sqllib/samples/clp` ディレクトリーに、Windows の場合は `sqllib\samples\clp` ディレクトリーにあります。このディレクトリーにある README ファイルに、プログラムとその実行方法の説明があります。

手順:

CLP スクリプト `cte.db2` で 2 つの共通表式 `PAYLEVEL` と `PAYBYED` が定義されており、これらには `SELECT` ステートメントでアクセスします。

このスクリプトを実行するには、以下のようにします。

1. まず、次のようにして sample データベースに接続します。

`db2 connect to sample`

2. 今度は次のコマンドを入力します。


```
db2 -vf cte.db2 -t
```

「v」は冗長フラグであり必須ではありませんが、詳細出力が得られるのでお勧めします。

この結果はデフォルトで画面に表示されます。これにより、SQL 共通表式とその出力が示されます。

```
WITH PAYLEVEL AS (SELECT EMPNO, YEAR(HIREDATE) AS HIREYEAR, EDLEVEL,
SALARY+BONUS+COMM AS TOTAL_PAY FROM EMPLOYEE WHERE EDLEVEL > 16 ),
PAYBYED (EDUC_LEVEL, YEAR_OF_HIRE, AVG_TOTAL_PAY) AS (SELECT EDLEVEL,
HIREYEAR, AVG(TOTAL_PAY) FROM PAYLEVEL GROUP BY EDLEVEL, HIREYEAR )
SELECT EMPNO, EDLEVEL, YEAR_OF_HIRE, TOTAL_PAY, AVG_TOTAL_PAY FROM
PAYLEVEL, PAYBYED WHERE EDLEVEL=EDUC_LEVEL AND HIREYEAR = YEAR_OF_HIRE
AND TOTAL_PAY < AVG_TOTAL_PAY

EMPNO  EDLEVEL YEAR_OF_HIRE TOTAL_PAY      AVG_TOTAL_PAY
-----
000210      17      1979      20132.00      25896.5000000000000000000000000000

1 record(s) selected.
```

関連概念:

- ・ 「コマンド・リファレンス」の『コマンド行プロセッサ (CLP)』

関連タスク:

- ・ 150 ページの『コマンド行プロセッサ (CLP) からのプロシーチャーの呼び出し』

関連資料:

- ・ 85 ページの『コマンド行プロセッサ (CLP) のサンプル』
- ・ 「コマンド・リファレンス」の『db2 - コマンド行プロセッサの呼び出しコマンド』

関連サンプル:

- ・ 『cte.db2 -- How to create a COMMON TABLE EXPRESSION 』

コマンド行プロセッサ (CLP) からのプロシーチャーの呼び出し

DB2 コマンド行プロセッサ・インターフェースから CALL ステートメントを呼び出せば、ストアド・プロシーチャーを呼び出すことができます。呼び出すことができるのは、DB2 システム・カタログ表で定義されているストアド・プロシーチャーだけです。

手順:

ストアド・プロシーチャーを呼び出すには、まず以下を使用してデータベースへ接続する必要があります。

```
db2 connect to sample user userid using password
```

ここで、*userid* と *password* は、sample データベースが置かれているインスタンスのユーザー ID とパスワードを表します。

CALL ステートメントを使用するには、 ストアド・プロシージャ名、IN または INOUT パラメーター値、 および各 OUT パラメーター値のプレースホルダーとしての '?' を入力します。

ストアド・プロシージャのパラメーターは、プログラム・ソース・ファイルにある、そのストアド・プロシージャの CREATE PROCEDURE ステートメントに示されています。

SQL プロシージャの例

SQL プロシージャの作成については、『SQL プロシージャの作成』を参照してください。

whiles.db2 ファイルにある、 DEPT_MEDIAN プロシージャ・シグニチャーの CREATE PROCEDURE ステートメントは、次のとおりです。

```
CREATE PROCEDURE DEPT_MEDIAN
(IN deptNumber SMALLINT, OUT medianSalary DOUBLE)
```

このプロシージャを呼び出すには、CALL ステートメントを使用し、そこでプロシージャ名と適切なパラメーター引き数を指定する必要があります。この場合は IN パラメーターの値と、OUT パラメーターの値の疑問符「?」を指定します。このプロシージャの SELECT ステートメントは、 STAFF 表の DEPT 列の deptNumber 値を使用するので、意味のある出力を取得するには、IN パラメーターが、たとえば次のように、「51」などの DEPT 列の有効な値である必要があります。

```
db2 call dept_median (51, ?)
```

注: UNIX プラットフォームでは、括弧はコマンド・シェルにとって特別な意味があるので、前に ¥ 文字を付けるか、または以下のように引用符で囲まなければなりません。

```
db2 "call dept_median (51, ?)"
```

コマンド行プロセッサを対話モードで使用する場合は、引用符を使用しません。

上記のコマンドを実行すると、以下の結果が生じるはずです。

```
Value of output parameters
-----
Parameter Name : MEDIANSALARY
Parameter Value : +1.76545000000000E+004

Return Status = 0
```

C ストアド・プロシージャの例

コマンド行プロセッサを使用して、サポートされるホスト言語で作成されたストアド・プロシージャを呼び出すこともできます。 DB2 にはストアド・プロシージャを作成するためのファイルが用意されており、 UNIX の場合は samples/c ディレクトリーに、 Windows の場合は samples%c ディレクトリーにあります。 spserver 共用ライブラリーには、ソース・ファイル spserver.sqc から作成できる多数のストアド・プロシージャが含まれています。 spcreate.db2 ファイルは、ストアド・プロシージャをカタログします。

spcreate.db2 ファイルにある、MAIN_EXAMPLE プロシージャの CREATE PROCEDURE ステートメントの初めの部分は、次のとおりです。

```
CREATE PROCEDURE MAIN_EXAMPLE (IN job CHAR(8),  
                                OUT salary DOUBLE,  
                                OUT errorcode INTEGER)
```

このストアド・プロシージャを呼び出すには、IN パラメーター job に CHAR 値を、また各 OUT パラメーターに疑問符「?」を入力する必要があります。このプロシージャの SELECT ステートメントは、EMPLOYEE 表の JOB 列の job 値を使用するので、意味のある出力を取得するには、IN パラメーターが JOB 列の有効な値である必要があります。ストアド・プロシージャを呼び出す C サンプル・プログラム spclient は、JOB 値として 'DESIGNER' を使用しています。ここでも同様に、以下のようにします。

```
db2 "call MAIN_EXAMPLE ('DESIGNER', ?, ?)"
```

上記のコマンドを実行すると、以下の結果が生じるはずです。

```
Value of output parameters  
-----  
Parameter Name : SALARY  
Parameter Value : +2.37312500000000E+004  
  
Parameter Name : ERRORCODE  
Parameter Value : 0  
  
Return Status = 0
```

ERRORCODE がゼロの場合は、結果が正常であることを示しています。

spclient プログラムと比較すると、spclient のほうが、結果が10 進数のフォーマットに設定されていて見やすいことが分かります。

```
CALL stored procedure named MAIN_EXAMPLE  
Stored procedure returned successfully  
Average salary for job DESIGNER = 23731.25
```

関連タスク:

- 153 ページの『SQL プロシージャの作成』
- 154 ページの『クライアント・アプリケーションによる SQL プロシージャの呼び出し』
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『トリガーまたは SQL ルーチンからのプロシージャの呼び出し』
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『アプリケーションまたは外部ルーチンからのプロシージャの呼び出し』

関連サンプル:

- 『spclient.sqc -- Call various stored procedures (C)』
- 『spcreate.db2 -- How to catalog the stored procedures contained in spserver.sqc (C)』
- 『spserver.sqc -- Definition of various types of stored procedures (C)』
- 『whiles.db2 -- To create the DEPT_MEDIAN SQL procedure 』
- 『whiles.sqc -- To call the DEPT_MEDIAN SQL procedure』

第 6 章 SQL プロシージャ

SQL プロシージャの作成	153	DB2 8.2 より前に作成された SQL プロシージャ	
クライアント・アプリケーションによる SQL プロ		のバックアップとリストア	157
シージャの呼び出し	154	SQL プロシージャの再バインド	158
SQL プロシージャのプリコンパイル・オプショ			
ンと BIND オプションのカスタマイズ	155		

SQL プロシージャの作成

UNIX では `sqllib/samples/sqlproc` ディレクトリーに、Windows では `sqllib\samples\sqlproc` ディレクトリーにある DB2 コマンド行プロセッサ・スクリプト (`.db2` 拡張子で終わるもの) は、`CREATE PROCEDURE` ステートメントを実行してサーバー上にストアード・プロシージャを作成します。どの CLP スクリプトにも、対応する同一名のクライアント・アプリケーション・ファイルがあります。なおその名前には、`.sqc` (C 組み込み SQL の場合)、`.c` (DB2 CLI の場合)、または `.java` (JDBC の場合) といった、言語とアプリケーション・インターフェースを表す拡張子が付いています。

注:

1. DB2 バージョン 8.2 以降では、SQL プロシージャの作成の際にサーバー上では C または C++ コンパイラーは必要ないため、そのセットアップは不要です。SQL プロシージャが作成されると、プロシージャ・ステートメントは、他の SQL ステートメントで実行される場合と同様、ネイティブ表現に変換されて、データベース・カタログ内に保管されます。SQL プロシージャが呼び出されるときに、この表現はカタログからロードされ、DB2 エンジンで実行されます。
2. `CALL` は DB2 バージョン 8 の SQL ステートメントです。これは、いかなる順序でも、プロシージャを作成できなくなったことを意味します。コンパイラーは、呼び出されるプロシージャがあるかどうかをコンパイル時にチェックし、プロシージャが見つからない場合は、`SQLCODE -440` を返します。

手順:

`CREATE PROCEDURE` CLP スクリプトを実行するには、次のコマンドでサンプル・データベースに接続してください。

```
db2 connect to sample user userid using password
```

ここで、*userid* と *password* は、sample データベースが置かれているインスタンスのユーザー ID とパスワードを表します。

`resultset.db2` スクリプト・ファイルの `CREATE PROCEDURE` を実行するには、以下のコマンドを入力します。

```
db2 -td@ -vf resultset.db2
```

これで、SQL プロシージャを呼び出す準備ができました。

関連タスク:

- 155 ページの『SQL プロシーチャーのプリコンパイル・オプションと BIND オプションのカスタマイズ』
- 158 ページの『SQL プロシーチャーの再バインド』

関連サンプル:

- 『resultset.db2 -- To register and create the MEDIAN_RESULT_SET SQL procedure』

クライアント・アプリケーションによる SQL プロシーチャーの呼び出し

SQL プロシーチャーの作成後は (153 ページの『SQL プロシーチャーの作成』 を参照)、クライアント・アプリケーションを構築および実行することによって、SQL プロシーチャーを呼び出すことができます。DB2 では、サンプル・クライアント・プログラムが `sqllib/samples/sqlproc` (UNIX)、および `sqllib\samples\sqlproc` (Windows) に用意されています。DB2 CLI、C 組み込み SQL、および JDBC 用のクライアント・ソース・ファイルがあります。DB2では、SQL プロシーチャーを呼び出すためのコマンド行プロセッサ・スクリプトおよびバッチ・ファイルが、それぞれ UNIX ディレクトリーと Windows ディレクトリーに用意されています。

手順:

使用するアプリケーション・インターフェースによっては、以下の例のようにして、SQL プロシーチャーを呼び出すサンプル・クライアント・プログラムを構築して実行することができます。

DB2 CLI

ソース・ファイル `resultset.c` から DB2 CLI クライアント・アプリケーション `resultset` を構築するには、次のように入力します。

```
bldcli resultset
```

このコマンドによって、実行可能ファイル `resultset` (UNIX 上) および `resultset.exe` (Windows 上) が作成されます。

SQL プロシーチャーを呼び出すには、実行可能ファイルの名前、接続しているデータベースの名前、そしてデータベース・インスタンスのユーザー ID とパスワードを入力して、サンプル・クライアント・アプリケーションを実行します。

```
resultset database userid password
```

C 組み込み SQL

ソース・ファイル `basecase.sqc` から組み込み SQL クライアント・アプリケーション `basecase` を構築するには、スクリプト・ファイル名、実行可能ファイル名、接続しているデータベース、およびデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
bldapp basecase database userid password
```

結果として、実行可能ファイル `basecase` (UNIX 上) および `basecase.exe` (Windows 上) が作成されます。

SQL プロシージャを呼び出すには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
basecase database userid password
```

JDBC ソース・ファイル `NestedSP.java` から JDBC クライアント・アプリケーション `NestedSP` を構築するには、次のようにソース・ファイルをコンパイルします。

```
javac NestedSP.java
```

これで、クラス・ファイル `NestedSP.class` が作成されます。

SQL プロシージャを呼び出すには、次のようにアプリケーションで Java インタープリターを実行します。

```
java NestedSP database userid password
```

関連タスク:

- 153 ページの『SQL プロシージャの作成』
- 150 ページの『コマンド行プロセッサ (CLP) からのプロシージャの呼び出し』
- 158 ページの『SQL プロシージャの再バインド』

関連サンプル:

- 『basecase.sqc -- To call the UPDATE_SALARY SQL procedure』
- 『NestedSP.java -- Client application for invoking nested stored procedures 』
- 『resultset.c -- To call the MEDIAN_RESULT_SET SQL procedure』

SQL プロシージャのプリコンパイル・オプションと BIND オプションのカスタマイズ

手順:

SQL プロシージャ用のプリコンパイルおよび BIND オプションは、インスタンス全体に適用される DB2 レジストリー変数 `DB2_SQLROUTINE_PREOPTS` を、次のコマンドで設定することによってカスタマイズできます。

```
db2set DB2_SQLROUTINE_PREOPTS=<options>
```

使用できるオプションは、次のものだけです。

```
BLOCKING {UNAMBIG | ALL | NO}
DATETIME {DEF | USA | EUR | ISO | JIS | LOC}
DEGREE {1 | degree-of-parallelism | ANY}
DYNAMICRULES {BIND | RUN}
EXPLAIN {NO | YES | ALL}
EXPLSNAP {NO | YES | ALL}
FEDERATED {NO | YES}
INSERT {DEF | BUF}
ISOLATION {CS | RR | UR | RS | NC}
QUERYOPT optimization-level
VALIDATE {RUN | BIND}
```

これらのオプションは、SET_ROUTINE_OPTS ストアード・プロシージャを使用して、プロシージャ・レベルで変更できます。現行セッションで SQL プロシージャを作成するために設定されているオプションの値は、GET_ROUTINE_OPTS 関数を使用して取得できます。

例。

この例に使用されている SQL プロシージャは、CLP スクリプトで定義されます(以下を参照)。これらのスクリプトは sqlproc サンプル・ディレクトリーにはありませんが、CREATE プロシージャ・ステートメントを独自のファイルにカット・アンド・ペーストすることによって、これらのファイルを簡単に作成できます。

これらの例では「expenses」という名前の表を使用します。これは、次のようにしてサンプル・データベース内に作成できます。

```
db2 connect to sample
db2 CREATE TABLE expenses(amount DOUBLE, date DATE)
db2 connect reset
```

初めに、日付用の ISO フォーマットの使用をインスタンス全体に適用される設定として指定します。

```
db2set DB2_SQLROUTINE_PREOPTS="DATETIME ISO"
db2stop
db2start
```

変更を有効にするには、DB2 をいったん停止してから再始動する必要があります。

次に、データベースに接続します。

```
db2 connect to sample
```

最初のプロシージャは、CLP スクリプト maxamount.db2 で次のように定義されます。

```
CREATE PROCEDURE maxamount(OUT maxamnt DOUBLE)
BEGIN
  SELECT max(amount) INTO maxamnt FROM expenses;
END @
```

オプション DATETIME ISO および ISOLATION UR を指定して作成されます。

```
db2 "CALL SET_ROUTINE_OPTS(GET_ROUTINE_OPTS() || ' ISOLATION UR')"
db2 -td@ -vf maxamount.db2
```

次のプロシージャは、CLP スクリプト fullamount.db2 で次のように定義されます。

```
CREATE PROCEDURE fullamount(OUT fullamnt DOUBLE)
BEGIN
  SELECT sum(amount) INTO fullamnt FROM expenses;
END @
```

オプション ISOLATION CS を指定して作成されます(ここでは、インスタンス全体に適用される DATETIME ISO 設定は使用しないことに注意してください)。

```
CALL SET_ROUTINE_OPTS('ISOLATION CS')
db2 -td@ -vf fullamount.db2
```


例の最後のプロシーチャーは、CLP スクリプト `perday.db2` で次のように定義されます。

```
CREATE PROCEDURE perday()  
BEGIN  
    DECLARE cur1 CURSOR WITH RETURN FOR  
        SELECT date, sum(amount)  
        FROM expenses  
        GROUP BY date;  
  
    OPEN cur1;  
END @
```

最後の `SET_ROUTINE_OPTS` 呼び出しでは、引き数として `NULL` 値を使用しています。これによって、`DB2_SQLROUTINE_PREPOPTS` レジストリーに指定されているグローバル設定がリストアされるため、最後のプロシーチャーはオプション `DATETIME ISO` を指定して作成されます。

```
CALL SET_ROUTINE_OPTS(NULL)  
db2 -td@ -vf perday.db2
```

関連タスク:

- 157 ページの『DB2 8.2 より前に作成された SQL プロシーチャーのバックアップとリストア』
- 153 ページの『SQL プロシーチャーの作成』
- 150 ページの『コマンド行プロセッサ (CLP) からのプロシーチャーの呼び出し』
- 154 ページの『クライアント・アプリケーションによる SQL プロシーチャーの呼び出し』
- 158 ページの『SQL プロシーチャーの再バインド』

関連資料:

- 「コマンド・リファレンス」の『PRECOMPILE コマンド』

DB2 8.2 より前に作成された SQL プロシーチャーのバックアップとリストア

DB2 バージョン 8.2 では、SQL プロシーチャーは、トリガーやビューと同様の方法で完全にデータベース内で管理されるという意味では、ファースト・クラスのデータベース・オブジェクトです。したがって、バックアップおよびリストア中に特別に考慮すべきことはありません。バージョン 8.2 より前に作成されたプロシーチャーの場合は、ファイル・システム内に DLL (ダイナミック・リンク・ライブラリー) を作成することが関係するため、特別に考慮すべきことがあります。V8.2 より前の DB2 で SQL プロシーチャーが作成されると、生成された共用ダイナミック・リンク・ライブラリー (DLL) は、ソース・テキスト、パッケージ、およびそれらに関連したファイルと一緒にデータベース・カタログに保管されます。したがって、データベース・バックアップを実行すれば、それらの情報はすべて保管されます。

手順:

データベースのリカバリー時には、リカバリー中のデータベースに属するファイル・システムにあるすべての SQL プロシージャ実行可能プログラムが除去されます。索引作成構成パラメーター `indexrec` が `RESTART` に設定されていると、すべての SQL プロシージャ実行可能プログラムは、カタログ表から抽出され、次の接続時にファイル・システムに書き戻されます。設定されていないと、SQL 実行可能プログラムは、SQL プロシージャの最初の実行で抽出されます。

実行可能ファイルは、以下のディレクトリーに置かれます。

UNIX `$HOME/sqllib/function/routine/sqlproc/<database_name>`

Windows

`sqllib¥function¥routine¥sqlproc¥<database_name>`

`<database_name>` は、SQL プロシージャが作成されたデータベースを表します。

リストア操作後にデータベースへの初めての接続を試みたときに以下のメッセージが返された場合:

SQL2048N オブジェクト "SQL PROCEDURE FILES" のアクセス中にエラーが発生しました。
理由コード: "7".

`db2stop` を使用して DB2 をいったん停止してから `db2start` を使用して再始動します。

関連タスク:

- 155 ページの『SQL プロシージャのプリコンパイル・オプションと BIND オプションのカスタマイズ』
- 153 ページの『SQL プロシージャの作成』
- 150 ページの『コマンド行プロセッサ (CLP) からのプロシージャの呼び出し』
- 154 ページの『クライアント・アプリケーションによる SQL プロシージャの呼び出し』
- 158 ページの『SQL プロシージャの再バインド』

SQL プロシージャの再バインド

手順:

SQL プロシージャに対応するパッケージを再バインドするには、`SYSPROC.REBIND_ROUTINE_PACKAGE` 組み込みストアド・プロシージャを呼び出します。

たとえば、データベース内に `MYSHEMA.MYPROC` という SQL プロシージャが存在する場合に、そのパッケージをコマンド行プロセッサ (CLP) から再バインドするには、次のようなコマンドを発行します。

```
CALL SYSPROC.REBIND_ROUTINE_PACKAGE('P', 'MYSHEMA.MYPROC', 'CONSERVATIVE')
```

ただし 'P' は、'MYSHEMA.MYPROC' がプロシージャ名であることを示します。最初のパラメーターに 'SP' が置かれていると、'MYSHEMA.MYPROC' は具体的なプ

ロシージャー名であることを示し、'CONSERVATIVE' は、従来の再バインドのセマンティクスを適用する必要があることを示します。従来の再バインドの詳細は、以下の関連リンクにある REBIND コマンドを参照してください。

関連タスク:

- 155 ページの『SQL プロシージャのプリコンパイル・オプションと BIND オプションのカスタマイズ』
- 157 ページの『DB2 8.2 より前に作成された SQL プロシージャのバックアップとリストア』
- 153 ページの『SQL プロシージャの作成』
- 150 ページの『コマンド行プロセッサ (CLP) からのプロシージャの呼び出し』
- 154 ページの『クライアント・アプリケーションによる SQL プロシージャの呼び出し』

関連資料:

- 「コマンド・リファレンス」の『REBIND コマンド』

第 7 章 Perl

Perl アプリケーションの構築 161

この章は、DB2 データベースにアクセスする Perl プログラムを構築するための詳細な情報を提供します。

DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

Perl アプリケーションの構築

DB2 は、Perl で書かれたクライアント・アプリケーションのデータベース・アクセスをサポートしています。本書の発刊時点では、Perl Database Interface (Perl DBI) バージョン 0.93 以降用の DB2 UDB ドライバー (DBD::DB2) のリリース 0.76 は、AIX、HP-UX、Linux、Solaris および Windows 用のものが用意されています。最新のドライバーの入手方法については、以下を参照してください。

<http://www.ibm.com/software/data/db2/perl>

DB2 では、Perl サンプル・プログラムが、UNIX の場合は `sqllib/samples/perl` ディレクトリーに、Windows の場合は `sqllib\samples\perl` ディレクトリーに用意されています。

UNIX 上の 64 ビット環境用のセットアップ

UNIX 上では、64 ビット・バージョンの Perl を使用しない場合は、60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』で説明されているのとおり、64 ビット環境で 32 ビット・アプリケーションを実行する場合に推奨されているのと同じ手順を実行してください。ただし、アプリケーションを再バインドする必要はありません。ラッパー・プログラムを使用する場合は、`$1` の代わりに `perl $1` を使用します。

注: 以下の例で、UNIX 64 ビット環境でラッパー・プログラムを使用する場合は、`perl <program_name>` の代わりに `<wrapper_name> <program_name>` を使用します。

手順:

コマンド行で DB2 Perl プログラムに対して `perl` インタープリターを実行するには、インタープリター名とプログラム名 (拡張子を含む) を入力します。

- サーバーにローカル接続している場合:

```
perl dbauth.pl
```

- リモート・クライアントから接続している場合:

```
perl dbauth.pl sample <userid> <password>
```

一部のプログラムでは、サポート・ファイルを実行する必要があります。 `tbssel` サンプル・プログラムでは、 `tbsselcreate.db2` CLP スクリプトで複数の表を作成する必要があります。 `tbsselinit` スクリプト (UNIX)、または `tbsselinit.bat` バッチ・ファイル (Windows) は、 `tbssel.drop.db2` を呼び出して表をドロップしてから (存在する場合)、 `tbsselcreate.db2` を呼び出して表を作成します。したがって、プログラムを実行するには、次のコマンドを入力します。

- サーバーにローカル接続している場合:

```
tbsselinit
perl tbssel.pl
```

- リモート・クライアントから接続している場合:

```
tbsselinit
perl tbssel.pl sample <userid> <password>
```

注: リモート・クライアントの場合は、 `tbsselinit` または `tbsselinit.bat` ファイルの接続ステートメントを修正して、 `db2 connect to sample user <userid> using <password>` のように、自分のユーザー ID とパスワードをハードコーディングする必要があります。

ルーチンの呼び出し

DB2 クライアント・アプリケーションは、サポートされるホスト言語または SQL プロシージャで作成されたルーチン (ストアド・プロシージャおよびユーザー定義関数) にアクセスできます。例えば、サンプル・プログラム `spclient.pl` は、SQL プロシージャ `spserver` 共用ライブラリーにアクセスできます (データベースに存在する場合)。

注: ホスト言語ルーチンを構築するには、サーバー上に適切なコンパイラーをセットアップしておく必要があります。SQL プロシージャには、コンパイラーは不要です。共用ライブラリーは、サーバー上でのみ構築でき、リモート・クライアントからは構築できません。

SQL プロシージャの呼び出しをデモンストレーションするには、サーバー上の `samples/sqlproc` ディレクトリー (UNIX) または `samples%sqlproc` ディレクトリー (Windows) に移動し、次のコマンドを実行して SQL プロシージャを作成し、`spserver` ライブラリーにカタログします。

```
db2 connect to sample
db2 -td@ -vf spserver.db2
```

次に、`perl` サンプル・ディレクトリー (リモート・クライアント・マシン上でも構いません) に戻り、 `spserver` 共用ライブラリーにアクセスするクライアント・プログラムに対して、Perl インタープリターを実行します。

- サーバーにローカル接続している場合は、次のように入力します。

```
perl spclient
```

- リモート・クライアントから接続している場合は、次のように入力します。

```
perl spclient sample <userid> <password>
```

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『Perl でのプログラミングに関する考慮事項』

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『Perl DBI』

関連タスク:

- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』

関連資料:

- 101 ページの『Perl のサンプル』

第 8 章 PHP

PHP アプリケーションの構築 165

この章は、DB2 データベースにアクセスする PHP プログラムを構築するための詳細な情報を提供します。

DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

PHP アプリケーションの構築

DB2 は、PHP で書かれたクライアント・アプリケーションのデータベース・アクセスをサポートしています。PHP は、サーバー側の、HTML に組み込まれた、クロスプラットフォーム・スクリプト言語です。これは、Unified-ODBC アクセス方式を使用して DB2 アクセスをサポートしており、ユーザー・レベルの PHP は ODBC 呼び出しを使用して DB2 との通信を行います。標準の ODBC とは異なり、Unified-ODBC 方式では、通信は ODBC 層を介さずに DB2 CLI 層に対して直接行われます。DB2 での PHP の使用法についての詳細は、次の DB2 サポート・サイトを参照してください。

<http://www.ibm.com/software/data/db2/udb/winoss2unix/support>

本書の印刷の時点で、最新バージョンは PHP 4.3.4 です。以下のサイトから、最新バージョンの PHP をダウンロードできます。

<http://www.php.net>

tar ファイルをダウンロードしたら、オプション `--with-ibm-db2=<DIR>` を指定してファイルをコンパイルします。<DIR> は、プラットフォーム・パスによって次のようになります。

- UNIX の場合のオプション: `--with-ibm-db2=$HOME/sql1lib`
- Windows の場合のオプション: `--with-ibm-db2=%DB2PATH%`

php-4.3.4.tar ファイル (これ以降の PHP バージョンのファイルが入手可能であればそのファイル) を `untar` すると、php-4.3.4 ディレクトリーに `php.ini-dist` ファイルが入ります。このファイルには、PHP の新規インストール用のデフォルト設定が含まれています。このファイルは、ファイル名を `php.ini` に変更して、インストール・パスにコピーする必要があります。次のコマンドは、標準のインストール・パスを想定しています。

- UNIX の場合:

```
cd ../php-4.x.y
cp php.ini-dist /usr/local/lib/php.ini
```

- Windows の場合;

```
cd ..\php-4.x.y
copy php.ini-dist C:\Windows\php.ini
```


注: 次のオプションを指定して、インストール時に別のパスを指定することもできます。

`--with-config-file-path=<path>`

Linux の RPM を使用してインストールする場合は、`php.ini` のデフォルト・パスは `/etc` になります。Linux の RPM の場合は、新規ファイル `.odbc.ini` をホーム・ディレクトリーに作成し、DB2 で ODBC を構成するために、次の内容を組み込みます。

```
[ODBC Data Sources]
Sample              = <description>

[Sample]
Driver              = $HOME/sqllib/lib/libdb2.so
Description         = <description>
Host                = localhost
UserName            = <user>
Password            = <password>
Database            = sample
```

Windows の InstallShield を使用してインストールする場合は、IBM DB2 ODBC ドライバーを追加し、「管理ツール」の「データ ソース (ODBC)」を使用して、ドライバをサンプル・データベース用に構成します。

詳しくは、INSTALL ファイルを参照してください。

`php.ini` ファイルを編集して PHP オプションを設定できます。DB2 では、DB2 サンプル・プログラムを実行するために、次のオプションを設定することが推奨されています。

```
track_errors = On
register_globals = On
register_argc_argv = On
max_execution_time = 60
odbc.defaultlrl = 100000
```

注:

1. `track_errors = On`: エラーのトラッキングを許可します。
2. `register_globals = On`: `register_globals = Off` の場合は、`$argc` と `$argv` の代わりに、それぞれ `$_SERVER['argc']` と `$_SERVER['argv'][0]` を使用してください。
3. `register_argc_argv = On`: コマンド行引き数を許可します。
4. `max_execution_time = 60`: スクリプトの実行時間が長くなる場合があるため、デフォルト値を変更する必要があります。
5. `odbc.defaultlrl = 100000`: サンプルの BLOB データを保持するのに十分の大きさです (バイト単位)。

DB2 では、PHP サンプル・プログラムが、UNIX の場合は `sqllib/samples/php` フォルダに、Windows の場合は `sqllib\samples\php` フォルダに用意されています。

UNIX 上の 64 ビット環境用のセットアップ

UNIX 上では、64 ビット・バージョンの PHP を使用しない場合は、60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』で説明されてい

るとおり、64 ビット環境で 32 ビット・アプリケーションを実行する場合に推奨されているのと同じ手順を実行してください。ただし、アプリケーションを再バインドする必要はありません。ラッパー・プログラムを使用する場合は、\$1 の代わりに php \$1 を使用します。

注: 以下の例で、UNIX 64 ビット環境でラッパー・プログラムを使用する場合は、php <program_name> の代わりに <wrapper_name> <program_name> を使用します。

手順:

コマンド行で DB2 PHP ソース・ファイルに対して php インタープリターを実行するには、インタープリター名とソース・ファイル名 (拡張子を含む) を入力します。

- サーバーにローカル接続している場合:

```
php dbauth.php
```

- リモート・クライアントから接続している場合:

```
php dbauth.php sample <userid> <password>
```

一部のプログラムでは、サポート・ファイルを実行する必要があります。tbssel サンプル・プログラムでは、tbsselcreate.db2 CLP スクリプトで複数の表を作成する必要があります。tbsselinit スクリプト (UNIX)、または tbsselinit.bat バッチ・ファイル (Windows) は、tbsseldrop.db2 を呼び出して表をドロップしてから (存在する場合)、tbsselcreate.db2 を呼び出して表を作成します。したがって、プログラムを実行するには、次のコマンドを入力します。

- サーバーにローカル接続している場合:

```
tbsselinit  
php tbssel.php
```

- リモート・クライアントから接続している場合:

```
tbsselinit  
php tbssel.php sample <userid> <password>
```

注: リモート・クライアントの場合は、tbsselinit または tbsselinit.bat ファイルの接続ステートメントを修正して、db2 connect to sample user <userid> using <password> のように、自分のユーザー ID とパスワードをハードコーディングする必要があります。

ユーザー定義関数の呼び出し

DB2 クライアント・アプリケーションは、サポートされるホスト言語で作成されたユーザー定義関数にアクセスできます。例えば、サンプル・プログラム udfcli.php は、C ユーザー定義関数 udfsrv 共用ライブラリーにアクセスできます (データベースに存在する場合)。

注: ホスト言語ユーザー定義関数の共用ライブラリーを構築するには、サーバー上に適切なコンパイラーをセットアップしておく必要があります。共用ライブラリーは、サーバー上でのみ構築でき、リモート・クライアントからは構築できません。PHP は、クライアント・プログラムによるストアード・プロシージャの呼び出しをサポートしていません。

C コンパイラーがサーバー上にセットアップされているという前提で、ユーザー定義関数の呼び出しをデモンストレーションするには、サーバー上の `samples/c` ディレクトリー (UNIX) または `samples%c` ディレクトリーに移動し、次のコマンドを実行して `udfsrv` ライブラリーをデータベースに作成します。

```
bldrtn udfsrv
```

次に、`php` サンプル・ディレクトリー (リモート・クライアント・マシン上でも構いません) に戻り、`udfsrv` 共用ライブラリーにアクセスするクライアント・プログラムに対して、`php` インタープリターを実行します。

- サーバーにローカル接続している場合は、次のように入力します。

```
php udfcli.php
```

- リモート・クライアントから接続している場合は、次のように入力します。

```
php udfcli.php sample <userid> <password>
```

関連タスク:

- 60 ページの『32 ビット環境から 64 ビット環境へのアプリケーションの移行』

関連資料:

- 102 ページの『PHP のサンプル』

第 3 部 プラットフォーム固有アプリケーションの構築および実行

第 9 章 UNIX

UNIX C アプリケーションの構築	171	UNIX C++ ルーチンの構築	183
UNIX C 複数接続アプリケーションの構築	173	UNIX Micro Focus COBOL アプリケーションの構築	187
UNIX C ルーチンの構築	175	UNIX Micro Focus COBOL ルーチンの構築	189
UNIX C++ アプリケーションの構築	179		
UNIX C++ 複数接続アプリケーションの構築	181		

この章では、サポートされる UNIX オペレーティング・システム用のアプリケーションおよびルーチンを構築するための共通ステップについて説明します。コンパイラー・オプションなどのプラットフォーム固有の詳細は、次の章以降の各プラットフォームの章で取り上げます。

各オペレーティング・システムには、これらのプログラムのための特定の構築要件があるため、各プラットフォームの章に C/C++ マルチスレッド・アプリケーションを構築するための情報が含まれています。

UNIX C アプリケーションの構築

DB2 には、C 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/c` ディレクトリーに置かれています。

ビルド・ファイル `bldapp` には、DB2 アプリケーション・プログラムを構築するコマンドが入っています。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。このパラメーターは、唯一必要なパラメーターであり、組み込み SQL を含まない DB2 API プログラムに必要なパラメーターはこのパラメーターだけです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、オプションとして 3 つのパラメーターが用意されています。2 番目のパラメーターは `$2` で、接続するデータベースの名前を指定します。3 番目のパラメーターは `$3` で、データベースのユーザー ID を指定します。そしてもう 1 つが `$4` で、データベースのパスワードを指定します。

組み込み SQL プログラムの場合、`bldapp` は、プリコンパイルおよびバインドのスクリプト `embprep` にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

以下の例は、DB2 API と組み込み SQL のアプリケーションを構築して実行する方法を示しています。

ソース・ファイル `cli_info.c` から DB2 API 非組み込み SQL サンプル・プログラム `cli_info` を構築するには、次のように入力します。

```
bldapp cli_info
```

結果として、実行可能ファイル `cli_info` が作成されます。

この実行可能ファイルを実行するには、ファイル名を入力します。

```
cli_info
```

組み込み SQL アプリケーションの構築と実行

ソース・ファイル `tbmod.sqc` から組み込み SQL アプリケーション `tbmod` を構築する場合、次の 3 つの方法があります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

```
bldapp tbmod
```

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

```
bldapp tbmod database
```

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

```
bldapp tbmod database userid password
```

結果として、実行可能ファイル `tbmod` が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

```
tbmod
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
tbmod database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
tbmod database userid password
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』

関連資料:

- 195 ページの『AIX C アプリケーションのコンパイルとリンクのオプション』
- 224 ページの『HP-UX C アプリケーションのコンパイルとリンクのオプション』
- 242 ページの『Linux C アプリケーションのコンパイルとリンクのオプション』

- 258 ページの『Solaris C アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX C application programs (C)』
- 『bldapp -- Builds HP-UX C applications (C)』
- 『bldapp -- Builds Linux C applications (C)』
- 『bldapp -- Builds Solaris C applications (C)』
- 『cli_info.c -- Set and get information at the client level (C)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』
- 『tbmod.sqc -- How to modify table data (C)』

UNIX C 複数接続アプリケーションの構築

DB2 には、C 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/c` ディレクトリーに置かれています。

ビルド・ファイル `bldmc` には、DB2 複数接続プログラム (2 つのデータベースが必要) を構築するためのコマンドが入っています。コンパイル・オプションとリンク・オプションは、`bldapp` で使用されるオプションと同じです。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。第 2 パラメーター `$2` には、接続先の最初のデータベースの名前を指定します。第 3 パラメーター `$3` には、接続先の 2 番目のデータベースの名前を指定します。これらはすべて必要パラメーターです。

注: `makefile` には、データベース名として `"sample"` と `"sample2"` (それぞれ `$2` と `$3`) がハードコーディングされているため、この `makefile` を使用してこれらのデフォルトを受け入れる場合は、指定する必要があるのはプログラム名 (`$1` パラメーター) だけです。 `bldmc` スクリプトを使用する場合は、3 つのパラメーターすべてを指定する必要があります。

オプション・パラメーターは、ローカル接続の場合は必要ありませんが、リモート・クライアントからサーバーに接続する場合は必要です。オプション・パラメーターの `$4` と `$5` は、それぞれ最初のデータベースのユーザー ID とパスワードを指定し、`$6` と `$7` は、それぞれ 2 番目のデータベースのユーザー ID とパスワードを指定します。

手順:

複数接続サンプル・プログラム (`dbmcon`) には、2 つのデータベースが必要です。 `sample` データベースをまだ作成していない場合は、コマンド行に `db2samp1` と入力して作成できます。 2 番目のデータベース (ここでは `sample2` という名前) は、以下のいずれかのコマンドを使用して作成できます。

データベースをローカルで作成する場合:

```
db2 create db sample2
```


データベースをリモートから作成する場合:

```
db2 attach to <node_name>
db2 create db sample2
db2 detach
db2 catalog db sample2 as sample2 at node <node_name>
```

<node_name> は、データベースが常駐するノードです。

複数接続では、TCP/IP Listener も実行されている必要があります。これを確実に実行するには、次のようにします。

1. 環境変数 DB2COMM を TCP/IP に設定します。

```
db2set DB2COMM=TCPIP
```

2. サービス・ファイルで指定されるように、データベース・マネージャーの構成ファイルを TCP/IP サービス名で更新します。

```
db2 update dbm cfg using SVCENAME <TCP/IP service name>
```

各インスタンスは TCP/IP サービス名を持っており、この名前はサービス・ファイルにリストされています。このファイルが見つからない場合、またはサービス・ファイルを変更するファイル許可がない場合は、システム管理者に連絡してください。

3. データベース・マネージャーをいったん停止してから再始動して、これらの変更を有効にします。

```
db2stop
db2start
```

dbmcon プログラムは、以下の 5 つのファイルで構成されています。

dbmcon.sqc

両方のデータベースに接続するためのメイン・ソース・ファイル。

dbmcon1.sqc

最初のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon1.h

dbmcon.sqc に組み込まれている dbmcon1.sqc 用のヘッダー・ファイル。最初のデータベースにバインドする表を作成およびドロップする SQL ステートメントにアクセスするために必要です。

dbmcon2.sqc

2 番目のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon2.h

dbmcon.sqc に組み込まれている dbmcon2.sqc 用のヘッダー・ファイル。2 番目のデータベースにバインドする表を作成およびドロップする SQL ステートメントにアクセスするために必要です。

複数接続サンプル・プログラム (dbmcon) を構築するには、次のように入力します。

```
bldmc dbmcon sample sample2
```

結果として、実行可能ファイル dbmcon が作成されます。

この実行可能ファイルを実行するには、ファイル名を入力します。

dbmcon

2 つのデータベースへの 1 フェーズ・コミットが、プログラムによって例示されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連資料:

- 「管理ガイド: パフォーマンス」の『svcename - 「TCP/IP サービス名」構成パラメーター』
- 195 ページの『AIX C アプリケーションのコンパイルとリンクのオプション』
- 224 ページの『HP-UX C アプリケーションのコンパイルとリンクのオプション』
- 242 ページの『Linux C アプリケーションのコンパイルとリンクのオプション』
- 258 ページの『Solaris C アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldmc -- Builds AIX C multi-connection applications (C)』
- 『bldmc -- Builds HP-UX C multi-connection applications (C)』
- 『bldmc -- Builds Linux C multi-connection applications (C)』
- 『bldmc -- Builds Solaris C multi-connection applications (C)』
- 『dbmcon.sqc -- How to use multiple databases (C)』
- 『dbmcon1.h -- Function declarations for the source file, dbmcon1.sqc (C)』
- 『dbmcon1.sqc -- Functions used in the multiple databases program dbmcon.sqc (C)』
- 『dbmcon2.h -- Function declarations for the source file, dbmcon2.sqc (C)』
- 『dbmcon2.sqc -- Functions used in the multiple databases program dbmcon.sqc (C)』

UNIX C ルーチンの構築

DB2 には、C プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/c` ディレクトリーに置かれています。

スクリプト `bldrtn` には、ルーチン (ストアード・プロシージャとユーザー定義関数) を構築するためのコマンドが入っています。このスクリプトは、データベース・マネージャーがロードできてしかもクライアント・アプリケーションから呼び出せるルーチンを共用ライブラリー中でコンパイルします。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。第 2 パラメーター `$2` には、接続先のデータベースの名前を指定します。

データベース・パラメーターはオプションです。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。また、データベースが置かれているのと同じインスタンス上にストアード・プロシージャが構築される必要があるため、ユーザー ID とパスワード用のパラメーターはありません。

手順:

この後の例は、次のものを使用してルーチンの共用ライブラリーを構築する方法を示しています。

- ストアード・プロシージャ
- 非組み込み SQL ユーザー定義関数 (UDF)
- 組み込み SQL ユーザー定義関数 (UDF)

ストアード・プロシージャの共用ライブラリー

ソース・ファイル `spserver.sqc` からサンプル・プログラム `spserver` を構築するには、次のように入力します。

1. `sample` データベースに接続している場合は、次のようにビルド・スクリプト名とプログラム名を入力します。

```
bldrtn spserver
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn spserver database
```

スクリプトは、共用ライブラリーをサーバー上の `sqllib/function` というパスにコピーします。

2. 次に、サーバーで `spcat` スクリプトを実行してルーチンをカタログします。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば `spdrop.db2` を呼び出してルーチンをアンカタログし、次に `spcreate.db2` を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、`spdrop.db2` スクリプトと `spcreate.db2` スクリプトは、個別に実行することもできます。

3. 次に、ストアード・プロシージャを構築するのが今回が初めてではない場合は、データベースをいったん停止してから再始動し、新しいバージョンの共用ライブラリーが認識されるようにします。これを行うには、コマンド行で `db2stop` に続けて `db2start` を入力します。

共用ライブラリー `spserver` の構築が完了したら、共用ライブラリーにアクセスするクライアント・アプリケーション `spclient` を構築することができます。

`spclient` は、スクリプト `bldapp` を使用して構築することができます。

共用ライブラリーにアクセスするには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
spclient database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、sample かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `spserver` にアクセスし、さまざまなストアード・プロシージャ関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

非組み込み SQL UDF の共用ライブラリー

ユーザー定義関数プログラム `udfsrv` をソース・ファイル `udfsrv.c` から構築するには、次のようにビルド・スクリプト名とプログラム名を入力します。

```
bldrtn udfsrv
```

スクリプトは、UDF を `sqllib/function` ディレクトリーにコピーします。

`udfsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfcli` を構築できます。このプログラムの DB2 CLI および組み込み SQL バージョンが提供されています。スクリプト `bldapp` を使用して、`sqllib/samples/cli` 内のソース・ファイル `udfcli.c` から DB2 CLI `udfcli` クライアント・プログラムを構築することができます。

スクリプト `bldapp` を使用して、`sqllib/samples/c` 内のソース・ファイル `udfcli.sqc` から組み込み SQL `udfcli` クライアント・プログラムを構築することができます。

共用ライブラリー内の UDF を呼び出すには、以下を入力してクライアント・アプリケーションを実行します。

```
udfcli database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、sample かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `udfsrv` にアクセスし、ユーザー定義関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

組み込み SQL UDF の共用ライブラリー

sample データベースに接続しているときに、組み込み SQL ユーザー定義関数プログラム `udfemsrv` をソース・ファイル `udfemsrv.sqc` から構築するには、次のようにビルド・スクリプト名とプログラム名を入力します。

```
bldrtn udfemsrv
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn udfemsrv database
```

スクリプトは、UDF を `sqllib/function` ディレクトリーにコピーします。

`udfemsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfemcli` を構築できます。スクリプト `bldapp` を使用して、`sqllib/samples/c` 内のソース・ファイル `udfemcli.sqc` から `udfemcli` クライアント・プログラムを構築することができます。

共用ライブラリー内の UDF を呼び出すには、以下を入力してクライアント・アプリケーションを実行します。

```
udfemcli database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、`sample` かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `udfemsrv` にアクセスし、ユーザー定義関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連資料:

- 197 ページの『AIX C ルーチンのコンパイルとリンクのオプション』
- 227 ページの『HP-UX C ルーチンのコンパイルとリンクのオプション』
- 244 ページの『Linux C ルーチンのコンパイルとリンクのオプション』
- 260 ページの『Solaris C ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX C routines (stored procedures and UDFs) (C)』
- 『bldrtn -- Builds HP-UX C routines (stored procedures and UDFs) (C)』
- 『bldrtn -- Builds Linux C routines (stored procedures or UDFs) (C)』
- 『bldrtn -- Builds Solaris C routines (stored procedures or UDFs) (C)』

- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』
- 『spclient.sqc -- Call various stored procedures (C)』
- 『spserver.sqc -- Definition of various types of stored procedures (C)』
- 『udfcli.sqc -- Call a variety of types of user-defined functions (C)』
- 『udfemcli.sqc -- Call a variety of types of embedded SQL user-defined functions. (C)』
- 『udfemsrv.sqc -- Call a variety of types of embedded SQL user-defined functions. (C)』
- 『udfsrv.c -- Defines a variety of types of user-defined functions (C)』

UNIX C++ アプリケーションの構築

DB2 には、C++ 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/cpp` ディレクトリーに置かれています。

ビルド・ファイル `bldapp` には、DB2 API と組み込み SQL アプリケーションを構築するためのコマンドが入っています。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。このパラメーターは、唯一必要なパラメーターであり、組み込み SQL を含まない DB2 API プログラムに必要なパラメーターはこのパラメーターだけです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、オプションとして 3 つのパラメーターが用意されています。2 番目のパラメーターは `$2` で、接続するデータベースの名前を指定します。3 番目のパラメーターは `$3` で、データベースのユーザー ID を指定します。そしてもう 1 つが `$4` で、データベースのパスワードを指定します。

組み込み SQL プログラムの場合、`bldapp` は、プリコンパイルおよびバインドのスクリプト `embprep` にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

以下の例は、DB2 API と組み込み SQL のアプリケーションを構築して実行する方法を示しています。

ソース・ファイル `cli_info.c` から非組み込み SQL サンプル・プログラム `cli_info` を構築するには、次のように入力します。

```
bldapp cli_info
```

結果として、実行可能ファイル `cli_info` が作成されます。sample データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
cli_info
```

組み込み SQL アプリケーションの構築と実行

ソース・ファイル `tbmod.sqC` から組み込み SQL アプリケーション `tbmod` を構築する場合、次の 3 つの方法があります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

```
bldapp tbmod
```

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

```
bldapp tbmod database
```

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

```
bldapp tbmod database userid password
```

結果として、実行可能ファイル `tbmod` が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

```
tbmod
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
tbmod database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
tbmod database userid password
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 183 ページの『UNIX C++ ルーチンの構築』

関連資料:

- 200 ページの『AIX C++ アプリケーションのコンパイルとリンクのオプション』
- 231 ページの『HP-UX C++ アプリケーションのコンパイルとリンクのオプション』
- 247 ページの『Linux C++ アプリケーションのコンパイルとリンクのオプション』
- 263 ページの『Solaris C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX C++ applications (C++)』
- 『bldapp -- Builds HP-UX C++ applications (C++)』
- 『bldapp -- Builds Linux C++ applications (C++)』

- 『bldapp -- Builds Solaris C++ applications (C++)』
- 『cli_info.C -- Set and get information at the client level (C++)』
- 『tbmod.sqlC -- How to modify table data (C++)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

UNIX C++ 複数接続アプリケーションの構築

DB2 には、C++ 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/cpp` ディレクトリーに置かれています。

ビルド・ファイル `bldmc` には、DB2 複数接続プログラム (2 つのデータベースが必要) を構築するためのコマンドが入っています。コンパイル・オプションとリンク・オプションは、`bldapp` で使用されるオプションと同じです。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。第 2 パラメーター `$2` には、接続先の最初のデータベースの名前を指定します。第 3 パラメーター `$3` には、接続先の 2 番目のデータベースの名前を指定します。これらはすべて必要パラメーターです。

注: `makefile` には、データベース名として `"sample"` と `"sample2"` (それぞれ `$2` と `$3`) がハードコーディングされているため、`makefile` を使用してこれらのデフォルトを受け入れる場合は、指定する必要があるのはプログラム名 (`$1` パラメーター) だけです。`bldmc` スクリプトを使用する場合は、3 つのパラメーターすべてを指定する必要があります。

オプション・パラメーターは、ローカル接続の場合は必要ありませんが、リモート・クライアントからサーバーに接続する場合は必要です。オプション・パラメーターの `$4` と `$5` は、それぞれ最初のデータベースのユーザー ID とパスワードを指定し、`$6` と `$7` は、それぞれ 2 番目のデータベースのユーザー ID とパスワードを指定します。

手順:

複数接続サンプル・プログラム (`dbmcon`) には、2 つのデータベースが必要です。`sample` データベースをまだ作成していない場合は、コマンド行に `db2samp1` と入力して作成できます。2 番目のデータベース (ここでは `sample2` という名前) は、以下のいずれかのコマンドを使用して作成できます。

データベースをローカルで作成する場合:

```
db2 create db sample2
```

データベースをリモートから作成する場合:

```
db2 attach to <node_name>
db2 create db sample2
db2 detach
db2 catalog db sample2 as sample2 at node <node_name>
```


<node_name> は、データベースが常駐するノードです。

複数接続では、TCP/IP Listener も実行されている必要があります。これを確実に実行するには、次のようにします。

1. 環境変数 DB2COMM を TCP/IP に設定します。

```
db2set DB2COMM=TCP/IP
```

2. サービス・ファイルで指定されるように、データベース・マネージャーの構成ファイルを TCP/IP サービス名で更新します。

```
db2 update dbm cfg using SVCENAME <TCP/IP service name>
```

各インスタンスは TCP/IP サービス名を持っており、この名前はサービス・ファイルにリストされています。このファイルが見つからない場合、またはサービス・ファイルを変更するファイル許可がない場合は、システム管理者に連絡してください。

3. データベース・マネージャーをいったん停止してから再始動して、これらの変更を有効にします。

```
db2stop  
db2start
```

dbmcon プログラムは、以下の 5 つのファイルで構成されています。

dbmcon.sqC

両方のデータベースに接続するためのメイン・ソース・ファイル。

dbmcon1.sqC

最初のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon1.h

dbmcon.sqC に組み込まれている dbmcon1.sqC 用のヘッダー・ファイル。最初のデータベースにバインドする表を作成およびドロップする SQL ステートメントにアクセスするために必要です。

dbmcon2.sqC

2 番目のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon2.h

dbmcon.sqC に組み込まれている dbmcon2.sqC 用のヘッダー・ファイル。2 番目のデータベースにバインドする表を作成およびドロップする SQL ステートメントにアクセスするために必要です。

複数接続サンプル・プログラム (dbmcon) を構築するには、次のように入力します。

```
bldmc dbmcon sample sample2
```

結果として、実行可能ファイル dbmcon が作成されます。

この実行可能ファイルを実行するには、ファイル名を入力します。

```
dbmcon
```

2 つのデータベースへの 1 フェーズ・コミットが、プログラムによって例示されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連資料:

- 「管理ガイド: パフォーマンス」の『svcname - 「TCP/IP サービス名」構成パラメーター』
- 200 ページの『AIX C++ アプリケーションのコンパイルとリンクのオプション』
- 231 ページの『HP-UX C++ アプリケーションのコンパイルとリンクのオプション』
- 247 ページの『Linux C++ アプリケーションのコンパイルとリンクのオプション』
- 263 ページの『Solaris C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldmc -- Builds AIX C++ multi-connection applications (C++)』
- 『bldmc -- Builds HP-UX C++ multi-connection applications (C++)』
- 『bldmc -- Builds Linux C++ multi-connection applications (C++)』
- 『bldmc -- Builds Solaris C++ multi-connection applications (C++)』
- 『dbmcon.sqC -- How to use multiple databases (C++)』
- 『dbmcon1.h -- Class declaration for the source file, dbmcon1.sqC (C++)』
- 『dbmcon1.sqC -- Functions used in the multiple databases program dbmcon.sqC (C++)』
- 『dbmcon2.h -- Class declaration for the source file, dbmcon2.sqC (C++)』
- 『dbmcon2.sqC -- Functions used in the multiple databases program dbmcon.sqC (C++)』

UNIX C++ ルーチンの構築

DB2 には、C++ プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/cpp` ディレクトリーに置かれています。

スクリプト・ファイル `bldrtn` には、ルーチンを構築するためのコマンドが入っています。このスクリプト・ファイルは、データベース・マネージャーがロードでき、しかもクライアント・アプリケーションから呼び出せるルーチンを共用ライブラリー中でコンパイルします。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。第 2 パラメーター `$2` には、接続先のデータベースの名前を指定します。

データベース・パラメーターはオプションです。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。また、データ

ベースが置かれているのと同じインスタンス上にストアード・プロシージャが構築される必要があるため、ユーザー ID とパスワード用のパラメーターはありません。

手順:

この後の例は、次のものを使用してルーチンの共用ライブラリーを構築する方法を示しています。

- ストアード・プロシージャ
- 非組み込み SQL ユーザー定義関数 (UDF)
- 組み込み SQL ユーザー定義関数 (UDF)

ストアード・プロシージャの共用ライブラリー

ソース・ファイル `spserver.sql` からサンプル・プログラム `spserver` を構築するには、次のように入力します。

1. `sample` データベースに接続している場合は、次のようにビルド・スクリプト名とプログラム名を入力します。

```
bldrtn spserver
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn spserver database
```

スクリプト・ファイルは、共用ライブラリーをサーバー上の `sqllib/function` というパスにコピーします。

2. 次に、サーバーで `spcat` スクリプトを実行してルーチンをカタログします。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば `spdrop.db2` を呼び出してルーチンをアンカタログし、次に `spcreate.db2` を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、`spdrop.db2` スクリプトと `spcreate.db2` スクリプトは、個別に実行することもできます。

3. 次に、ストアード・プロシージャを構築するのが今回が初めてではない場合は、データベースをいったん停止してから再始動し、新しいバージョンの共用ライブラリーが認識されるようにします。これを行うには、コマンド行で `db2stop` に続けて `db2start` を入力します。

共用ライブラリー `spserver` の構築が完了したら、共用ライブラリーにアクセスするクライアント・アプリケーション `spclient` を構築することができます。

`spclient` は、スクリプト・ファイル `bldapp` を使用して構築することができます。

共用ライブラリーにアクセスするには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
spclient database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、sample かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `spserver` にアクセスし、さまざまなストアド・プロシージャ関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

非組み込み SQL UDF の共用ライブラリー

ユーザー定義関数プログラム `udfsrv` をソース・ファイル `udfsrv.C` から構築するには、次のようにビルド・スクリプト名とプログラム名を入力します。

```
bldrtn udfsrv
```

スクリプト・ファイルは、UDF を `sqllib/function` ディレクトリーにコピーします。

必要であれば、UDF にファイル・モードを設定してデータベース・マネージャーからアクセスできるようにします。

`udfsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfcli` を構築できます。スクリプト・ファイル `bldapp` を使用して、ソース・ファイル `udfcli.sqC` から `udfcli` を作成することができます。

共用ライブラリー内の UDF を呼び出すには、以下を入力してクライアント・アプリケーションを実行します。

```
udfcli database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、sample かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `udfsrv` にアクセスし、ユーザー定義関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

組み込み SQL UDF の共用ライブラリー

sample データベースに接続しているときに、組み込み SQL ユーザー定義関数プログラム `udfemsrv` をソース・ファイル `udfemsrv.sqC` から作成するには、次のようにビルド・スクリプト名とプログラム名を入力します。

`bldrtn udfemsrv`

他のデータベースに接続しているときは、さらにデータベース名も入力します。

`bldrtn udfemsrv database`

スクリプト・ファイルは、UDF を `sqllib/function` ディレクトリーにコピーします。

`udfemsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfemcli` を構築できます。スクリプト・ファイル `bldapp` を使用して、ソース・ファイル `udfemcli.sqC` から `udfemcli` を作成することができます。

共用ライブラリー内の UDF を呼び出すには、以下を入力してクライアント・アプリケーションを実行します。

`udfemcli database userid password`

ここで、

database

接続先のデータベースの名前です。 名前は、`sample` かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `udfemsrv` にアクセスし、ユーザー定義関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連資料:

- 202 ページの『AIX C++ ルーチンのコンパイルとリンクのオプション』
- 233 ページの『HP-UX C++ ルーチンのコンパイルとリンクのオプション』
- 249 ページの『Linux C++ ルーチンのコンパイルとリンクのオプション』
- 265 ページの『Solaris C++ ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『`bldrtn -- Builds AIX C++ routines (stored procedures and UDFs) (C++)`』
- 『`bldrtn -- Builds HP-UX C++ routines (stored procedures and UDFs) (C++)`』
- 『`bldrtn -- Builds Linux C++ routines (stored procedures and UDFs) (C++)`』
- 『`bldrtn -- Builds Solaris C++ routines (stored procedures or UDFs) (C++)`』
- 『`spclient.sqC -- Call various stored procedures (C++)`』
- 『`spserver.sqC -- Definition of various types of stored procedures (C++)`』

- 『udfcli.sqC -- Call a variety of types of user-defined functions (C++)』
- 『udfemcli.sqC -- Call a variety of types of embedded SQL user-defined functions. (C++)』
- 『udfemsrv.sqC -- Call a variety of types of embedded SQL user-defined functions. (C++)』
- 『udfsrv.C -- Defines a variety of types of user-defined functions (C++)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

UNIX Micro Focus COBOL アプリケーションの構築

DB2 には、Micro Focus COBOL 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/cobol_mf` ディレクトリーに置かれています。

ビルド・ファイル `bldapp` には、DB2 アプリケーション・プログラムを構築するためのコマンドが入っています。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。これは組み込み SQL を使用しないプログラムに必要な唯一のパラメーターです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、オプションとして 3 つのパラメーターが用意されています。2 番目のパラメーターは `$2` で、接続するデータベースの名前を指定します。3 番目のパラメーターは `$3` で、データベースのユーザー ID を指定します。そしてもう 1 つが `$4` で、データベースのパスワードを指定します。

組み込み SQL プログラムの場合、`bldapp` は、プリコンパイルおよびバインドのスクリプト `embprep` にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

ソース・ファイル `client.cbl` から組み込み SQL を含まないサンプル・プログラム `client` を構築するには、次のように入力します。

```
bldapp client
```

結果として、実行可能ファイル `client` ができます。`sample` データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
client
```

組み込み SQL アプリケーションの構築と実行

ソース・ファイル `updat.sqb` から組み込み SQL アプリケーション `updat` を構築する方法には、次の 3 つがあります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

`bldapp updat`

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

`bldapp updat database`

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

`bldapp updat database userid password`

結果として、実行可能ファイル `updat` が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

`updat`

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

`updat database`

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

`updat database userid password`

関連タスク:

- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 218 ページの『AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 238 ページの『HP-UX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 269 ページの『Solaris Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 254 ページの『Linux Micro COBOL アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX Micro Focus COBOL applications』
- 『bldapp -- Builds HP-UX Micro Focus COBOL applications』
- 『bldapp -- Builds Linux Micro Focus COBOL applications』
- 『bldapp -- Builds Solaris Micro Focus COBOL applications』
- 『client.cbl -- How to set and query a client (MF COBOL)』
- 『updat.sqb -- How to update, delete and insert table data (MF COBOL)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

UNIX Micro Focus COBOL ルーチンの構築

DB2 には、Micro Focus COBOL 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib/samples/cobol_mf` ディレクトリーに置かれています。

スクリプト `bldrtn` には、ルーチン (ストアード・プロシージャ) を構築するためのコマンドが入っています。このスクリプトは、クライアント・アプリケーションから呼び出せるルーチンのソース・ファイルを共用ライブラリーの中でコンパイルします。

第 1 パラメーター `$1` には、ソース・ファイルの名前を指定します。スクリプトは、そのソース・ファイル名を共用ライブラリー名として使用します。第 2 パラメーター `$2` には、接続先のデータベースの名前を指定します。共用ライブラリーは、データベースが置かれているのと同じインスタンス上で構築する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

最初のパラメーター (ソース・ファイル名) だけが、必須です。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。

Solaris 固有の設定

Solaris 上で Micro Focus ルーチンを構築する場合、事前に以下のコマンドを実行してください。

```
db2stop
db2set DB2LIBPATH=$LD_LIBRARY_PATH
db2set DB2ENVLIST="COBDIR LD_LIBRARY_PATH"
db2set
db2start
```

`db2stop` がデータベースを確実に停止するようにしてください。最後の `db2set` コマンドが設定値をチェックするために出されます。 `DB2LIBPATH` および `DB2ENVLIST` が正しく設定されるようにしてください。

手順:

サンプル・データベースに接続している場合、ソース・ファイル `outsrv.sqb` からサンプル・プログラム `outsrv` を構築するには、次のように入力します。

```
bldrtn outsrv
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn outsrv database
```

スクリプト・ファイルは、共用ライブラリーをサーバー上の `sqllib/function` というパスにコピーします。

ストアード・プロシージャ `outsrv` を構築してしまえば、そのストアード・プロシージャを呼び出すクライアント・アプリケーション `outcli` を構築できます。`outcli` は、スクリプト・ファイル `bldapp` を使用して構築することができます。

ストアド・プロシージャを呼び出すためには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
outcli database userid password
```

ここで、

database

接続先のデータベースの名前です。名前は、`sample` かその別名、またはその他の名前にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `outsrv` にアクセスし、ストアド・プロシージャ関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』

関連資料:

- 220 ページの『AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 239 ページの『HP-UX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 270 ページの『Solaris Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 255 ページの『Linux Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX Micro Focus COBOL routines (stored procedures)』
- 『bldrtn -- Builds HP-UX Micro Focus COBOL routines (stored procedures)』
- 『bldrtn -- Builds Linux Micro Focus COBOL routines (stored procedures)』
- 『bldrtn -- Builds Solaris Micro Focus COBOL routines (stored procedures)』
- 『outcli.sqb -- Call stored procedures using the SQLDA structure (MF COBOL)』
- 『outsrv.sqb -- Demonstrates stored procedures using the SQLDA structure (MF COBOL)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

第 10 章 AIX

重要な考慮事項	191	構成ファイルによる C++ ストアード・プロシージャの構築	207
ルーチン用の AIX エクスポート・ファイル	191	構成ファイルによる C++ ユーザー定義関数の構築	208
AIX ルーチンと CREATE ステートメント	192	IBM COBOL Set for AIX	209
AIX 共用ライブラリーの置換	193	AIX での IBM COBOL コンパイラーの構成	209
AIX での COBOL のインストールに関する考慮事項	193	AIX での IBM COBOL アプリケーションの構築	210
IBM C	194	IBM COBOL アプリケーションのビルド・スクリプト	212
C アプリケーションのビルド・スクリプト	194	AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション	212
AIX C アプリケーションのコンパイルとリンクのオプション	195	AIX での IBM COBOL ルーチンの構築	213
C ルーチンのビルド・スクリプト	196	IBM COBOL ルーチンのビルド・スクリプト	215
AIX C ルーチンのコンパイルとリンクのオプション	197	AIX IBM COBOL ルーチンのコンパイルとリンクのオプション	215
AIX での C マルチスレッド・アプリケーションの構築	198	Micro Focus COBOL	217
VisualAge C++	199	AIX での Micro Focus COBOL コンパイラーの構成	217
C++ アプリケーションのビルド・スクリプト	199	Micro Focus COBOL アプリケーションのビルド・スクリプト	218
AIX C++ アプリケーションのコンパイルとリンクのオプション	200	AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション	218
C++ ルーチンのビルド・スクリプト	201	Micro Focus COBOL ルーチンのビルド・スクリプト	219
AIX C++ ルーチンのコンパイルとリンクのオプション	202	AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション	220
AIX での C++ マルチスレッド・アプリケーションの構築	203	REXX	221
VisualAge C++ 構成ファイル	204	AIX での REXX アプリケーションの構築	221
構成ファイルによる VisualAge C++ プログラムの構築	204		
構成ファイルによる C++ DB2 API アプリケーションの構築	205		
構成ファイルによる C++ 組み込み SQL アプリケーションの構築	206		

この章は、AIX でアプリケーションを構築するための詳細な情報を提供します。

AIX 用の DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

重要な考慮事項

この節では、サポートされている各種コンパイラーで DB2 アプリケーションを構築するための情報として、AIX に固有の情報を提供します。

ルーチン用の AIX エクスポート・ファイル

外部ルーチンは、サーバー上でコンパイルされ、サーバー上の共用ライブラリーに保管されて実行されます。これらの共用ライブラリーは、ルーチンをコンパイルするときに作成されます。

AIX® では、ライブラリー内のどのグローバル関数を外部から呼び出せるかを指定するエクスポート・ファイルをユーザーから提供する必要があります。または、次の例のように、コンパイラの「すべてをエクスポート」動作を使用して AIX 共用オブジェクトおよびライブラリーを作成できます。

```
xlc -qmkshrobj -q32 -g dlsources.C -o libtestit.so
```

DB2® サンプルは、エクスポート・ファイルを使用します。そのファイルには、ライブラリー内のすべてのルーチンの名前が入っていなければなりません。他の UNIX® プラットフォームは単に、ライブラリー内のすべてのグローバル関数をエクスポートするだけです。次は、AIX エクスポート・ファイルの例です。

```
#!/spserver export file
outlanguage
```

エクスポート・ファイルの `spserver.exp` には、ストアード・プロシージャ `outlanguage` の一覧が示されます。リンカーは `spserver.exp` を使用して、`outlanguage` ストアード・プロシージャの入った共用ライブラリー `spserver` を作成します。

AIX リンカーの資料には、エクスポート・ファイルに関する追加情報が記載されています。

関連概念:

- 192 ページの『AIX ルーチンと CREATE ステートメント』

関連タスク:

- 193 ページの『AIX 共用ライブラリーの置換』

AIX ルーチンと CREATE ステートメント

ここでは、ルーチンのコンパイルおよびリンクと、CREATE ステートメントの EXTERNAL NAME 文節中に入力する情報との間の関係を説明します。

プログラムをコンパイルしてリンクするときには、`-bE:` オプションで指定するエクスポート・ファイルを使用して外部関数を識別することができます。

ライブラリー `myrtns` には、`modify`、`remove`、および `add` の 3 つのルーチンが入っていると仮定します。`modify` をデフォルトのエントリー・ポイントと指定します。そのためには、これを、リンクのステップでリンクしたエクスポート・ファイル内の最初のエントリーに置きます。`remove` と `add` 関数は、やはりエクスポート・ファイル内に配置することで追加のエクスポート可能関数と指定します。

リンク・ステップにおいて、次のように指定します。

```
-bE:myrtns.exp
```

これは、エクスポート・ファイル `myrtns.exp` を指定します。

エクスポート・ファイルは、次のようになります。

```
modify
remove
add
```

modify、remove、および add 関数を使用してインプリメントされたルーチンの EXTERNAL NAME 文節のコーディングは、最終的に次のようになります。

```
EXTERNAL NAME '/u/mydir/routines/myrtns!modify'
```

および

```
EXTERNAL NAME '/u/mydir/routines/myrtns!remove'
```

および

```
EXTERNAL NAME '/u/mydir/routines/myrtns!add'
```

注: 使用するデフォルト・パスは `sqllib/function` です。その意味は次のとおりです。たとえば EXTERNAL NAME 文節を次のように指定したとします。

```
EXTERNAL NAME 'myrtns!modify'
```

上記の場合、DB2[®] は `sqllib/function` からの `myrtns` のロードを試みます。

関連概念:

- 191 ページの『ルーチン用の AIX エクスポート・ファイル』

関連タスク:

- 193 ページの『AIX 共用ライブラリーの置換』

AIX 共用ライブラリーの置換

手順:

共用ライブラリーの構築が完了すると、通常は、DB2 からそのライブラリーへのアクセス先となるディレクトリーにコピーします。ルーチン共用ライブラリーを置換したい場合は、`/usr/sbin/slibclean` を実行して AIX 共用ライブラリーのキャッシュをフラッシュするか、またはライブラリーをターゲット・ディレクトリーから除去したうえで、ソース・ディレクトリーからターゲット・ディレクトリーにライブラリーをコピーする必要があります。そうしないと、参照されるライブラリーのキャッシュが AIX で保持されて、ライブラリーを上書きできないため、コピー操作が失敗する可能性があります。

関連概念:

- 191 ページの『ルーチン用の AIX エクスポート・ファイル』
- 192 ページの『AIX ルーチンと CREATE ステートメント』

AIX での COBOL のインストールに関する考慮事項

AIX[®] でのルーチンのロードや、その中のライブラリー参照の解決の仕方に起因して、COBOL をどのようにインストールすればよいかを規制する要件が生じます。これらの要件は、COBOL プログラムがランタイムに共用ライブラリー (ルーチン) をロードするときの要素となります。

ルーチンをロードするときには、それが参照する一連のライブラリーのチェーンもロードする必要があります。プログラムで間接的にのみ参照するライブラリーを AIX が探索するときには、言語プロバイダー (IBM COBOL または Micro Focus COBOL) の作成した参照ライブラリーにコンパイルされたパスを使用する必要があります。

ります。このパスは必ずしも、コンパイラーがインストールされたパスと同じとは限りません。チェーン内でライブラリーが見つからないと、ルーチンのロードは失敗し、SQLCODE -444 を受け取ります。

そのような事態にならないようにするには、必要なときには常にコンパイラーをインストールし、その後、すべての言語ライブラリーのシンボリック・リンクを、インストール・ディレクトリーから /usr/lib (ライブラリーのロードが必要なときには、ほぼ必ず探索されるディレクトリー) に作成します。ライブラリーを sqllib/function (ルーチンのディレクトリー) にリンクできますが、これは、1 つのデータベース・インスタンスに対してしか機能しません。 /usr/lib は、マシン上のすべてのデータベース・インスタンスに対して機能します。

関連タスク:

- 36 ページの『UNIX アプリケーション開発環境のセットアップ』
- 209 ページの『AIX での IBM COBOL コンパイラーの構成』
- 217 ページの『AIX での Micro Focus COBOL コンパイラーの構成』

IBM C

DB2 CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

サポートされる UNIX オペレーティング・システム上で C アプリケーションを構築する方法については、171 ページの『UNIX C アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C ルーチンを構築する方法については、175 ページの『UNIX C ルーチンの構築』を参照してください。

C アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds AIX C application programs
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set lib32 for 32-bit programs, lib for 64-bit,
# and set extra compile flag for 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB=lib32
    EXTRA_CFLAG=
else
    LIB=lib
    EXTRA_CFLAG=-q64
fi

# If an embedded SQL program, precompile and bind it.
# Note: some .sqc files contain no SQL but link in
# utilemb.sqc, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqc" ]
then
```

```

./embprep $1 $2 $3 $4
# Compile the utilemb.c error-checking utility.
xlc $EXTRA_CFLAG -I$DB2PATH/include -c utilemb.c
else
# Compile the utilapi.c error-checking utility.
xlc $EXTRA_CFLAG -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
xlc $EXTRA_CFLAG -I$DB2PATH/include -c $1.c

if [ -f $1".sqc" ]
then
# Link the program with utilemb.o
xlc $EXTRA_CFLAG -o $1 $1.o utilemb.o -ldb2 -L$DB2PATH/$LIB
else
# Link the program with utilapi.o
xlc $EXTRA_CFLAG -o $1 $1.o utilapi.o -ldb2 -L$DB2PATH/$LIB
fi

```

AIX C アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、AIX IBM C コンパイラを使用して、C 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
xlc	IBM C コンパイラ。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sql1lib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
xlc	コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	組み込み SQL プログラムでない場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを含みます。
-ldb2	DB2 ライブラリーとリンクします。
-L\$DB2PATH/\$LIB	DB2 ランタイム共有ライブラリーのロケーションを指定します。たとえば、\$HOME/sql1lib/\$LIB。 -L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連資料:

- 197 ページの『AIX C ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX C application programs (C)』

C ルーチンのビルド・スクリプト

```

#!/bin/sh
# SCRIPT: bldrtn
# Builds AIX C routines (stored procedures and UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Set lib32 for 32-bit programs, lib for 64-bit,
# and set extra compile flag for 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB=lib32
    EXTRA_CFLAG=
else
    LIB=lib
    EXTRA_CFLAG=-q64
fi

```

```

| # If an embedded SQL program, precompile and bind it.
| if [ -f $1".sqc" ]
| then
|     ./embprep $1 $2
| fi
|
| # Compile the program.
| xlc_r $EXTRA_CFLAG -I$DB2PATH/include -c $1.c
|
| # Link the program using the export file $1.exp,
| xlc_r $EXTRA_CFLAG -qmksbobj -o $1 $1.o -ldb2 -L$DB2PATH/$LIB -bE:$1.exp
|
| # Copy the shared library to the sqllib/function subdirectory.
| # Note: the user must have write permission to this directory.
| rm -f $DB2PATH/function/$1
| cp $1 $DB2PATH/function

```

AIX C ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、AIX IBM C コンパイラを使用して、C ルーチン (ストアード・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
xlc_r	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、マルチスレッド・バージョンの IBM C コンパイラを使用してください。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sqllib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
xlc_r	リンカーのフロントエンドとしてマルチスレッド・バージョンのコンパイラーを使用します。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-qmksbobj	共用ライブラリーを作成します。
-o \$1	出力ファイル名を指定します。
\$1.o	オブジェクト・ファイルを指定します。
-ldb2	DB2 ライブラリーとリンクします。
-L\$DB2PATH/\$LIB	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、\$HOME/sql1lib/\$LIB。 -L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
-bE:\$1.exp	エクスポート・ファイルを指定します。エクスポート・ファイルには、ルーチンの一覧が入っています。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』

関連資料:

- 195 ページの『AIX C アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX C routines (stored procedures and UDFs) (C)』

AIX での C マルチスレッド・アプリケーションの構築

AIX 上の C マルチスレッド・アプリケーションのコンパイルおよびリンクでは、xlc コンパイラーの代わりに xlc_r コンパイラーを、C++ の場合は、x1C コンパイラーの代わりに x1C_r コンパイラーを使用する必要があります。_r バージョンでは、マルチスレッド用のコンパイルを定義している適当なプリプロセッサが設定され、適当なスレッド・ライブラリー名がリンカーに付けられます。

マルチスレッド・コンパイラーのフロントエンドを使用したコンパイラーおよびリンク・フラグの設定についてのさらに詳しい情報は、コンパイラーの資料を参照してください。

sql1lib/samples/c のスクリプト・ファイル bldmt には、組み込み SQL マルチスレッド・プログラムを構築するためのコマンドが含まれています。xlc_r コンパイラーや、リンクされているユーティリティー・ファイルがないという点だけでな

く、コンパイルおよびリンク・オプションも、組み込み SQL スクリプト・ファイル bldapp で使用されているものと同じです。

手順:

ソース・ファイル dbthrrds.sqc からマルチスレッド・サンプル・プログラム dbthrrds を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル dbthrrds が作成されます。 sample データベースに対してこの実行可能ファイルを実行するには、次の実行可能ファイル名を入力します。

```
dbthrrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連資料:

- 195 ページの『AIX C アプリケーションのコンパイルとリンクのオプション』
- 75 ページの『C サンプル』

関連サンプル:

- 『bldmt -- Builds AIX C multi-threaded applications (C)』
- 『dbthrrds.sqc -- How to use multiple context APIs on UNIX (C)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

VisualAge C++

サポートされる UNIX オペレーティング・システム上で C++ アプリケーションを構築する方法については、179 ページの『UNIX C++ アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C++ ルーチンを構築する方法については、183 ページの『UNIX C++ ルーチンの構築』を参照してください。

C++ アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds AIX C++ applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set lib32 for 32-bit programs, lib for 64-bit,
# and set extra compile flag for 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB=lib32
```

```

        EXTRA_CFLAG=
    else
        LIB=lib
        EXTRA_CFLAG=-q64
    fi

    # If an embedded SQL program, precompile and bind it.
    # Note: some .sqC files contain no SQL but link in
    # utilemb.sqC, so if you get this warning, ignore it:
    # SQL0053W No SQL statements were found in the program.
    if [ -f $1".sqC" ]
    then
        ./embprep $1 $2 $3 $4
        # Compile the utilemb.C error-checking utility.
        xlc $EXTRA_CFLAG -qstaticinline -I$DB2PATH/include -c utilemb.C
    else
        # Compile the utilapi.C error-checking utility.
        xlc $EXTRA_CFLAG -qstaticinline -I$DB2PATH/include -c utilapi.C
    fi

    # Compile the program.
    xlc $EXTRA_CFLAG -qstaticinline -I$DB2PATH/include -c $1.C

    if [ -f $1".sqC" ]
    then
        # Link the program with utilemb.o
        xlc $EXTRA_CFLAG -o $1 $1.o utilemb.o -ldb2 -L$DB2PATH/$LIB
    else
        # Link the program with utilapi.o
        xlc $EXTRA_CFLAG -o $1 $1.o utilapi.o -ldb2 -L$DB2PATH/$LIB
    fi

```

AIX C++ アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、AIX IBM VisualAge C++ コンパイラーを使用して、C++ 組み込み SQL および DB2 API アプリケーションを構築するのに勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
xlc	VisualAge C++ コンパイラー。
EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-qstaticinline	非インラインのインライン関数に内部結合を与え、関数が複数のオブジェクト・ファイルに存在する場合でも、リンク時に警告が出ないようにします。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sql1lib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
x1C	コンパイラーをリンカーのフロントエンドとして使用します。
EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
utilapi.o	非組み込み SQL プログラムの場合に、API ユーティリティー・オブジェクト・ファイルを含みます。
utilemb.o	組み込み SQL プログラムの場合に、組み込み SQL ユーティリティー・オブジェクト・ファイルを含みます。
-ldb2	DB2 ライブラリーとリンクします。
-L\$DB2PATH/\$LIB	DB2 ランタイム共有ライブラリーのロケーションを指定します。たとえば、\$HOME/sql1lib/\$LIB。 -L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』
- 206 ページの『構成ファイルによる C++ 組み込み SQL アプリケーションの構築』
- 205 ページの『構成ファイルによる C++ DB2 API アプリケーションの構築』

関連資料:

- 202 ページの『AIX C++ ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX C++ applications (C++)』

C++ ルーチンのビルド・スクリプト

```

#!/bin/sh
# SCRIPT: bldrtn
# Builds AIX C++ routines (stored procedures and UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Set lib32 for 32-bit programs, lib for 64-bit,
# and set extra compile flag for 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB=lib32
    EXTRA_CFLAG=

```

```
else
LIB=lib
EXTRA_CFLAG=-q64
fi

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqc" ]
then
./embprep $1 $2
fi

# Compile the program.
x1C_r $EXTRA_CFLAG -qstaticinline -I$DB2PATH/include -c $1.C

# Link using export file $1.exp, creating shared library $1
x1C_r $EXTRA_CFLAG -qmksrobj -o $1 $1.o -L$DB2PATH/$LIB -ldb2 -bE:$1.exp

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

AIX C++ ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、AIX VisualAge C++ コンパイラーを使用して、 C++ ルーチン (ストアード・プロシーチャーとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
x1C_r	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じたマルチスレッド・バージョンの IBM VisualAge C++ コンパイラー。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-qstaticinline	非インラインのインライン関数に内部結合を与え、関数が複数のオブジェクト・ファイルに存在する場合でも、リンク時に警告が出ないようにします。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sqllib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
x1C_r	リンカーのフロントエンドとしてマルチスレッド・バージョンのコンパイラーを使用します。
\$EXTRA_CFLAG	64 ビット・サポートが使用可能なインスタンスの場合は「-q64」が入り、それ以外の場合は値は入りません。
-qmksbobj	共用ライブラリーを生成します。
-o \$1	出力を、共用ライブラリー・ファイルとして指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
-L\$DB2PATH/\$LIB	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、 \$HOME/sql1lib/\$LIB。 -L オプションを指定しないと、コンパイラーは次のパスを 想定します。 /usr/lib:/lib。
-ldb2	DB2 ライブラリーとリンクします。
-bE:\$1.exp	エクスポート・ファイルを指定します。エクスポート・ファイルには、ルーチンの 一覧が入っています。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』
- 207 ページの『構成ファイルによる C++ ストアード・プロシーチャーの構築』
- 208 ページの『構成ファイルによる C++ ユーザー定義関数の構築』

関連資料:

- 200 ページの『AIX C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX C++ routines (stored procedures and UDFs) (C++)』

AIX での C++ マルチスレッド・アプリケーションの構築

AIX 上の C++ マルチスレッド・アプリケーションのコンパイルおよびリンクでは、x1c コンパイラーの代わりに x1c_r コンパイラーを、C の場合は、x1C コンパイラーの代わりに x1C_r コンパイラーを使用する必要があります。_r バージョンでは、マルチスレッド用のコンパイルを定義している適当なプリプロセッサが設定され、適当なスレッド・ライブラリー名がリンカーに付けられます。

マルチスレッド・コンパイラーのフロントエンドを使用したコンパイラーおよびリンク・フラグの設定についてのさらに詳しい情報は、コンパイラーの資料を参照してください。

スクリプト bldmt には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記の x1C_r コンパイラーや、リンクされているユーティリ

ティー・ファイルがないという点だけでなく、コンパイルおよびリンク・オプションも、組み込み SQL スクリプト・ファイル bldapp で使用されているものと同じです。

手順:

ソース・ファイル dbthrrds.sqC からマルチスレッド・サンプル・プログラム dbthrrds を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル dbthrrds が作成されます。 sample データベースに対してこの実行可能ファイルを実行するには、次の実行可能ファイル名を入力します。

```
dbthrrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連資料:

- 200 ページの『AIX C++ アプリケーションのコンパイルとリンクのオプション』
- 75 ページの『C サンプル』

関連サンプル:

- 『bldmt -- Builds AIX C++ multi-threaded applications (C++)』
- 『dbthrrds.sqC -- How to use multiple context APIs on UNIX (C++)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

VisualAge C++ 構成ファイル

注: CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

構成ファイルによる VisualAge C++ プログラムの構築

VisualAge C++ バージョン 5.0 には、増分コンパイラーとバッチ・モード・コンパイラーの両方が装備されています。バッチ・モード・コンパイラーは makefile とビルド・ファイルを使用するのに対して、増分コンパイラーは構成ファイルを使用します。これに関して詳しく知りたい場合は、VisualAge C++ バージョン 5.0 に添付されている資料を参照してください。

DB2 は、VisualAge C++ コンパイラーで構築できるさまざまなタイプの DB2 プログラム用の構成ファイルを提供します。

手順:

DB2 構成ファイルを使用するためには、まず、コンパイルするプログラム名に合わせて環境変数を設定します。次に、VisualAge C++ が提供しているコマンドを使用して、プログラムをコンパイルします。以下のトピックに、DB2 に用意されている構成ファイルを使用して各種のプログラムをコンパイルする方法が説明されています。

- 構成ファイルによる C++ 組み込み SQL アプリケーションの構築
- 構成ファイルによる C++ DB2 API アプリケーションの構築
- 構成ファイルによる C++ ストアード・プロシーチャーの構築
- 構成ファイルによる C++ ユーザー定義関数の構築

関連タスク:

- 206 ページの『構成ファイルによる C++ 組み込み SQL アプリケーションの構築』
- 205 ページの『構成ファイルによる C++ DB2 API アプリケーションの構築』
- 207 ページの『構成ファイルによる C++ ストアード・プロシーチャーの構築』
- 208 ページの『構成ファイルによる C++ ユーザー定義関数の構築』
- 171 ページの『UNIX C アプリケーションの構築』
- 175 ページの『UNIX C ルーチンの構築』
- 179 ページの『UNIX C++ アプリケーションの構築』
- 183 ページの『UNIX C++ ルーチンの構築』

構成ファイルによる C++ DB2 API アプリケーションの構築

sqllib/samples/c と sqllib/samples/cpp の中の構成ファイル api.icc を使用すれば、AIX 上で C または C++ の DB2 API プログラムを構築することができます。

手順:

構成ファイルを使用して、ソース・ファイル cli_info.c から DB2 API サンプル・プログラム cli_info を構築するには、以下のようにします。

1. 次のように入力して、API 環境変数をプログラム名に設定します。

- bash または Korn シェルの場合

```
export API=cli_info
```

- C シェルの場合

```
setenv API cli_info
```

2. api.icc ファイルを使用して異なるプログラムを構築することによって生成された api.ics ファイルが作業ディレクトリーにある場合は、次のコマンドで api.ics ファイルを削除してください。

```
rm api.ics
```

既存の api.ics ファイルが、再構築するその同じプログラム用に生成されているのであれば、削除する必要はありません。

3. サンプル・プログラムを以下のように入力してコンパイルします。

```
vacbld api.icc
```


注: vacbld コマンドは、 VisualAge C++ で提供されます。

結果として、実行可能ファイル cli_info が作成されます。このプログラムを実行するには、次の実行可能ファイル名を入力します。

```
cli_info
```

関連タスク:

- 206 ページの『構成ファイルによる C++ 組み込み SQL アプリケーションの構築』
- 207 ページの『構成ファイルによる C++ ストアード・プロシーチャーの構築』
- 208 ページの『構成ファイルによる C++ ユーザー定義関数の構築』

構成ファイルによる C++ 組み込み SQL アプリケーションの構築

sqllib/samples/c と sqllib/samples/cpp の中の構成ファイル emb.icc を使用すれば、 AIX 上で C および C++ の DB2 組み込み SQL アプリケーションを構築することができます。

手順:

構成ファイルを使用してソース・ファイル tbmod.sqc から組み込み SQL アプリケーション tbmod を構築するには、次のようにします。

1. 次のように入力して、EMB 環境変数をプログラム名に設定します。

- bash または Korn シェルの場合

```
export EMB=tbmod
```

- C シェルの場合

```
setenv EMB tbmod
```

2. emb.icc ファイルを使用して異なるプログラムを構築することによって生成された emb.ics ファイルが作業ディレクトリーにある場合は、次のコマンドで emb.ics ファイルを削除してください。

```
rm emb.ics
```

既存の emb.ics ファイルが、再構築するその同じプログラム用に生成されているのであれば、削除する必要はありません。

3. サンプル・プログラムを以下のように入力してコンパイルします。

```
vacbld emb.icc
```

注: vacbld コマンドは、 VisualAge C++ で提供されます。

結果として、実行可能ファイル tbmod が作成されます。このプログラムを実行するには、次の実行可能ファイル名を入力します。

```
tbmod
```

関連タスク:

- 205 ページの『構成ファイルによる C++ DB2 API アプリケーションの構築』
- 207 ページの『構成ファイルによる C++ ストアード・プロシーチャーの構築』
- 208 ページの『構成ファイルによる C++ ユーザー定義関数の構築』

構成ファイルによる C++ ストアード・プロシーチャーの構築

sqlllib/samples/c と sqlllib/samples/cpp の中の構成ファイル stp.icc を使用すれば、AIX 上で C および C++ の DB2 組み込み SQL ストアード・プロシーチャーを構築することができます。

手順:

構成ファイルを使用して、ソース・ファイル spserver.sqc から組み込み SQL ストアード・プロシーチャーの共用ライブラリー spserver を構築するには、以下のようになります。

1. 次のように入力して、STP 環境変数をプログラム名に設定します。

- bash または Korn シェルの場合

```
export STP=spserver
```

- C シェルの場合

```
setenv STP spserver
```

2. stp.icc を使用して異なるプログラムを構築することによって生成された stp.ics ファイルが作業ディレクトリーにある場合は、次のコマンドで stp.ics ファイルを削除してください。

```
rm stp.ics
```

既存の stp.ics ファイルが、再構築するその同じプログラム用に生成されているのであれば、削除する必要はありません。

3. サンプル・プログラムを以下のように入力してコンパイルします。

```
vacbld stp.icc
```

注: vacbld コマンドは、VisualAge C++ で提供されます。

このストアード・プロシーチャーの共用ライブラリーは、パス sqlllib/function のサーバーにコピーされます。

次に、次のようにサーバーで spcat スクリプトを実行して、共用ライブラリーのストアード・プロシーチャーをカタログします。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ストアード・プロシーチャーがすでにカタログ済みであれば spdrop.db2 を呼び出してそれをアンカタログし、次に spcreate.db2 を呼び出してそのストアード・プロシーチャーをカタログし、そして最後にデータベースへの接続を切断します。また、spdrop.db2 スクリプトと spcreate.db2 スクリプトは、個別に実行することもできます。

カタログが終了したら、データベースを一度停止してから再始動し、新しい共用ライブラリーが認識されるようにします。必要であれば、共用ライブラリーにファイル・モードを設定して、DB2 インスタンスからアクセスできるようにします。

ストアード・プロシーチャーの共用ライブラリー spserver を作成し終わったら、その中のストアード・プロシーチャーを呼び出すクライアント・アプリケーション spclient を構築することができます。spclient は、構成ファイル emb.icc を使用して構築することができます。

ストアード・プロシージャを呼び出すためには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
spclient database userid password
```

ここで、

database

接続先のデータベースの名前です。 名前は、sample またはそのリモート別名、あるいはその他の名前にすることができます。

userid 有効なユーザー ID です。

password

有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー `spserver` にアクセスし、さまざまなストアード・プロシージャ関数をサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

関連タスク:

- 206 ページの『構成ファイルによる C++ 組み込み SQL アプリケーションの構築』
- 205 ページの『構成ファイルによる C++ DB2 API アプリケーションの構築』
- 208 ページの『構成ファイルによる C++ ユーザー定義関数の構築』

構成ファイルによる C++ ユーザー定義関数の構築

`sqllib/samples/c` と `sqllib/samples/cpp` の中の構成ファイル `udf.icc` を使用すれば、AIX 上で C および C++ のユーザー定義関数を構築することができます。

手順:

構成ファイルを使用して、ソース・ファイル `udfsrv.c` からユーザー定義関数プログラム `udfsrv` を構築するには、以下のようにします。

1. 次のように入力して、UDF 環境変数をプログラム名に設定します。

- bash または Korn シェルの場合

```
export UDF=udfsrv
```

- C シェルの場合

```
setenv UDF udfsrv
```

2. `udf.icc` ファイルを使用して異なるプログラムを構築することによって生成された `udf.ics` ファイルが作業ディレクトリーにある場合は、次のコマンドで `udf.ics` ファイルを削除してください。

```
rm udf.ics
```

既存の `udf.ics` ファイルが、再構築するその同じプログラム用に生成されているのであれば、削除する必要はありません。

3. サンプル・プログラムを以下のように入力してコンパイルします。

```
vacbld udf.icc
```

注: vacbld コマンドは、 VisualAge C++ で提供されます。

UDF ライブラリーは、サーバー上の sqllib/function というパスにコピーされます。

必要であれば、ユーザー定義関数にファイル・モードを設定して DB2 インスタンスがそれを実行できるようにしてください。

udfsrv の構築が完了したら、それを呼び出すクライアント・アプリケーション udfcli を構築できます。このプログラムの DB2 CLI および組み込み SQL パージョンが提供されています。

構成ファイル cli.icc を使用して、 sqllib/samples/cli 内のソース・ファイル udfcli.c から DB2 CLI udfcli プログラムを構築することができます。

構成ファイル emb.icc を使用して、 sqllib/samples/c 内のソース・ファイル udfcli.sqc から組み込み SQL udfcli プログラムを構築することができます。

UDF を呼び出すには、次の実行可能ファイル名を入力して、サンプルの呼び出しアプリケーションを実行します。

udfcli

この呼び出しアプリケーションは、 udfsrv ライブラリーから ScalarUDF 関数を呼び出します。

関連タスク:

- 206 ページの『構成ファイルによる C++ 組み込み SQL アプリケーションの構築』
- 205 ページの『構成ファイルによる C++ DB2 API アプリケーションの構築』
- 207 ページの『構成ファイルによる C++ ストアード・プロシージャの構築』

IBM COBOL Set for AIX

AIX での IBM COBOL コンパイラーの構成

組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合に、 IBM COBOL Set for AIX コンパイラーを使用していれば、以下のステップを行う必要があります。

手順:

- コマンド行プロセッサのコマンド db2 prep を使用してアプリケーションをプリコンパイルする場合は、 target ibmcob オプションを使用してください。
- ソース・ファイルの中でタブ文字を使用しないでください。
- コンパイル・オプションを設定するためには、ソース・ファイルの 1 行目で PROCESS および CBL キーワードを使うことができます。

- アプリケーションに組み込み SQL のみが含まれていて、DB2 API 呼び出しは含まれない場合には、pgmname(mixed) コンパイル・オプションを使う必要はありません。DB2 API 呼び出しを使用する場合には、pgmname(mixed) コンパイル・オプションを使う必要があります。
- IBM COBOL Set for AIX コンパイラーの「システム/390 ホスト・データ・タイプ・サポート」機能を使用している場合、アプリケーション用の DB2 組み込みファイルは、次のディレクトリー中にあります。

```
$HOME/sql1lib/include/cobol_i
```

提供されたスクリプト・ファイルを使用して DB2 サンプル・プログラムを構築している場合、スクリプト・ファイルで指定された組み込みファイルのパスは、cobol_a ディレクトリーではなく、cobol_i ディレクトリーを指すように変更しなければなりません。

IBM COBOL Set for AIX コンパイラーの「システム/390 ホスト・データ・タイプ・サポート」機能を使用していない場合、またはこのコンパイラーのそれよりも前のバージョンを使用している場合、アプリケーション用の DB2 組み込みファイルは、次のディレクトリー中にあります。

```
$HOME/sql1lib/include/cobol_a
```

次のように、COPY ファイル名を .cbl 拡張子を含めて指定します。

```
COPY "sql.cbl".
```

関連概念:

- 193 ページの『AIX での COBOL のインストールに関する考慮事項』

関連タスク:

- 36 ページの『UNIX アプリケーション開発環境のセットアップ』
- 210 ページの『AIX での IBM COBOL アプリケーションの構築』
- 213 ページの『AIX での IBM COBOL ルーチンの構築』

関連資料:

- 212 ページの『AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション』
- 215 ページの『AIX IBM COBOL ルーチンのコンパイルとリンクのオプション』

AIX での IBM COBOL アプリケーションの構築

DB2 には、IBM COBOL 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に sql1lib/samples/cobol ディレクトリーに置かれています。

ビルド・ファイル bldapp には、DB2 アプリケーション・プログラムを構築するためのコマンドが入っています。

第 1 パラメーター \$1 には、ソース・ファイルの名前を指定します。これは組み込み SQL を使用しないプログラムに必要な唯一のパラメーターです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、オプションとして 3 つのパラメーターが用意されています。2 番目のパラメーターは \$2 で、

接続するデータベースの名前を指定します。3 番目のパラメーターは \$3 で、データベースのユーザー ID を指定します。そしてもう 1 つが \$4 で、データベースのパスワードを指定します。

組み込み SQL プログラムの場合、bldapp は、プリコンパイルおよびバインドのスクリプト embprep にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの sample データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

ソース・ファイル client.cbl から組み込み SQL を含まないサンプル・プログラム client を構築するには、次のように入力します。

```
bldapp client
```

結果として、実行可能ファイル client ができます。sample データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
client
```

組み込み SQL アプリケーションの構築と実行

ソース・ファイル updat.sqb から組み込み SQL アプリケーション updat を構築する方法には、次の 3 つがあります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

```
bldapp updat
```

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

```
bldapp updat database
```

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

```
bldapp updat database userid password
```

結果として、実行可能ファイル updat が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある sample データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

```
updat
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
updat database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
updat database userid password
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 213 ページの『AIX での IBM COBOL ルーチンの構築』

関連資料:

- 212 ページの『AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『bldapp -- Builds AIX COBOL applications』
- 『client.cbl -- How to set and query a client (IBM COBOL)』
- 『embprep -- To prep and bind a COBOL embedded SQL sample on AIX』
- 『updat.sqb -- How to update, delete and insert table data (IBM COBOL)』

IBM COBOL アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds AIX COBOL applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    ./embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error checking utility.
cob2 -qpgmname$(mixed¥) -qlib -I$DB2PATH/include/cobol_a ¥
    -c checkerr.cbl

# Compile the program.
cob2 -qpgmname$(mixed¥) -qlib -I$DB2PATH/include/cobol_a ¥
    -c $1.cbl

# Link the program.
cob2 -o $1 $1.o checkerr.o -L$DB2PATH/lib -ldb2
```

AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、IBM AIX COBOL Set コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob2	IBM COBOL Set コンパイラー。
-qpgmname¥(mixed¥)	コンパイラーに、大文字小文字混合の名前を持つライブラリーのエンタリー・ポイントの CALL を許可するように指示します。
-qlib	コンパイラーに COPY ステートメントを処理するように指示します。
-I\$DB2PATH/include/cobol_a	DB2 組み込みファイルのロケーションを指定します。たとえば、 \$HOME/sql1lib/include/cobol_a。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。
リンク・オプション	
cob2	コンパイラーをリンカーのフロントエンドとして使用します。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
checkerr.o	エラー・チェック用のユーティリティー・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、 \$HOME/sql1lib/lib。 -L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
-ldb2	データベース・マネージャー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 210 ページの『AIX での IBM COBOL アプリケーションの構築』

関連資料:

- 215 ページの『AIX IBM COBOL ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX COBOL applications』

AIX での IBM COBOL ルーチンの構築

DB2 には、COBOL 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのビルド・スクリプトが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に sql1lib/samples/cobol ディレクトリーに置かれています。

sqllib/samples/cobol にあるスクリプト・ファイル bldrtn には、ルーチン (ストアド・プロシージャ) を構築するためのコマンドが入っています。このスクリプトは、クライアント・アプリケーションから呼び出せるルーチンを共用ライブラリーの中でコンパイルします。

第 1 パラメーター \$1 には、ソース・ファイルの名前を指定します。第 2 パラメーター \$2 には、接続先のデータベースの名前を指定します。共用ライブラリーは、データベースが置かれているのと同じインスタンス上に作成する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

最初のパラメーター (ソース・ファイル名) だけが、必須です。スクリプトは、ソース・ファイル名 \$1 を共用ライブラリー名として使用します。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの sample データベースを使用します。

手順:

サンプル・データベースに接続している場合、ソース・ファイル outsrv.sqb からサンプル・プログラム outsrv を構築するには、次のように入力します。

```
bldrtn outsrv
```

他のデータベースに接続しているときは、さらにデータベース名も含めます。

```
bldrtn outsrv database
```

スクリプト・ファイルは、共用ライブラリーをサーバー上の sqllib/function というパスにコピーします。

ルーチンの共用ライブラリー outsrv を構築し終わったら、クライアント・アプリケーション outcli を構築することができます。これは、ライブラリー内のルーチンを呼び出すアプリケーションです。outcli は、スクリプト・ファイル bldapp を使用して構築することができます。

ルーチンを呼び出すには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
outcli database userid password
```

ここで、

database

接続先のデータベースの名前です。名前は、sample またはそのリモート別名、あるいはその他の名前にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは共用ライブラリー outsrv にアクセスし、同一名のルーチンをサーバー・データベース上で実行します。この出力は、クライアント・アプリケーションに戻されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連タスク:

- 210 ページの『AIX での IBM COBOL アプリケーションの構築』

関連資料:

- 215 ページの『AIX IBM COBOL ルーチンのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『bldrtn -- Builds AIX COBOL routines (stored procedures)』
- 『embprep -- To prep and bind a COBOL embedded SQL sample on AIX』
- 『outcli.sqb -- Call stored procedures using the SQLDA structure (IBM COBOL)』
- 『outsrv.sqb -- Demonstrates stored procedures using the SQLDA structure (IBM COBOL)』

IBM COBOL ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds AIX COBOL routines (stored procedures)
# Usage: bldrtn <program_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
./embprep $1 $2

# Compile the checkerr.cbl error checking utility.
cob2 -qpdbname$(mixed¥) -qlib -I$DB2PATH/include/cobol_a ¥
    -c checkerr.cbl

# Compile the program.
cob2 -qpdbname$(mixed¥) -qlib -c -I$DB2PATH/include/cobol_a $1.cbl

# Link the program creating shared library $1 with export file $1.exp
cob2 -o $1 $1.o checkerr.o -bnoentry -bE:$1.exp ¥
    -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function directory of the DB2 instance.
# This assumes the user has write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

AIX IBM COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、AIX 上で IBM COBOL Set コンパイラーを使用して、COBOL ルーチン (ストアード・プロシージャ) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cob2	IBM COBOL Set コンパイラー。
-qpgmname¥(mixed¥)	コンパイラーに、大文字小文字混合の名前を持つライブラリーのエン트리・ポイントの CALL を許可するように指示します。
-qlib	コンパイラーに COPY ステートメントを処理するように指示します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。
-I\$DB2PATH/include/cobol_a	DB2 組み込みファイルのロケーションを指定します。たとえば、 \$HOME/sql1lib/include/cobol_a。
リンク・オプション	
cob2	リンク・エディットをするコンパイラーを使用します。
-o \$1	出力を、共用ライブラリー・ファイルとして指定します。
\$1.o	ストアード・プロシージャー・オブジェクト・ファイルを指定します。
checkerr.o	エラー・チェック用のユーティリティー・オブジェクト・ファイルを組み込みます。
-bnoentry	共用ライブラリーへのデフォルトのエン트리・ポイントを指定しません。
-bE:\$1.exp	エクスポート・ファイルを指定します。エクスポート・ファイルには、ストアード・プロシージャーのリストが含まれています。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、 \$HOME/sql1lib/lib。 -L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
-ldb2	データベース・マネージャー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 213 ページの『AIX での IBM COBOL ルーチンの構築』

関連資料:

- 212 ページの『AIX IBM COBOL アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX COBOL routines (stored procedures)』

Micro Focus COBOL

サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL アプリケーションを構築する方法については、187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL ルーチンを構築する方法については、189 ページの『UNIX Micro Focus COBOL ルーチンの構築』を参照してください。

AIX での Micro Focus COBOL コンパイラーの構成

Micro Focus COBOL コンパイラーを使用して、組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合は、次のようにします。

手順:

- コマンド行プロセッサのコマンド `db2 prep` を使用してアプリケーションをプリコンパイルする場合は、`target mfcob` オプションを使用してください。
- DB2 COBOL COPY ファイル・ディレクトリーを、Micro Focus COBOL 環境変数 `COBCPY` に含める必要があります。 `COBCPY` 環境変数は、`COPY` ファイルのロケーションを指定します。Micro Focus COBOL 用の DB2 COPY ファイルは、データベース・インスタンス・ディレクトリーの下にある `sqllib/include/cobol_mf` にあります。

このディレクトリーを含めるには、次のように入力します。

- bash または korn シェルの場合

```
export COBCPY=$COBCPY:$HOME/sqllib/include/cobol_mf
```
- C シェルの場合

```
setenv COBCPY $COBCPY:$HOME/sqllib/include/cobol_mf
```

注: `COBCPY` を `.profile` または `.login` ファイル中に設定することもできます。

関連概念:

- 193 ページの『AIX での COBOL のインストールに関する考慮事項』

関連タスク:

- 36 ページの『UNIX アプリケーション開発環境のセットアップ』
- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』
- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 218 ページの『AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 220 ページの『AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Micro Focus COBOL アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds AIX Micro Focus COBOL applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    ./embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error checking utility.
cob -c -x checkerr.cbl

# Compile the program.
cob -c -x $1.cbl

# Link the program.
cob -x -o $1 $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```

AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、AIX 上で Micro Focus COBOL コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	COBOL コンパイラー。
-c	コンパイルのみを実行し、リンクは実行しません。
-x	-c と一緒に使用すると、オブジェクト・ファイルが作成されます。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	実行可能プログラムを作成します。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
-L\$DB2PATH/lib	DB2 ランタイム共有ライブラリーのロケーションを指定します。たとえば、 \$HOME/sqllib/lib。 -L オプションを指定しないと、コンパイラーは次のパスを想 定します。 /usr/lib:/lib。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』

関連資料:

- 220 ページの『AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp -- Builds AIX Micro Focus COBOL applications』

Micro Focus COBOL ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds AIX Micro Focus COBOL routines (stored procedures)
# Usage: bldrtn <program_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# Precompile and bind the program.
./embprep $1 $2

# Compile the program.
cob -c -x $1.cbl

# Link the program.
cob -x -o $1 $1.o -Q -bnoentry ¥
-Q -bI:$DB2PATH/lib/db2g.imp -L$DB2PATH/lib -ldb2 -ldb2gmf

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

AIX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、AIX 上で Micro Focus COBOL コンパイラーを使用して、COBOL ルーチン (ストアード・プロシージャー) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	COBOL コンパイラー。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。
-x	-c オプションと一緒に使用すると、オブジェクト・モジュールへとコンパイルします。
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	共用ライブラリーを作成します。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
-Q -bnoentry	共用ライブラリーへのデフォルトのエントリー・ポイントを指定しません。
-Q -bI:\$DB2PATH/lib/db2g.imp	DB2 アプリケーション・ライブラリーへのエントリー・ポイントのリストを提供します。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、\$HOME/sql1lib/lib。-L オプションを指定しないと、コンパイラーは次のパスを想定します。 /usr/lib:/lib。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 218 ページの『AIX Micro COBOL アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldrtn -- Builds AIX Micro Focus COBOL routines (stored procedures)』

AIX での REXX アプリケーションの構築

以下に、AIX 上で REXX アプリケーションを構築する方法を示してあります。DB2 (AIX 版) では、オブジェクト REXX 以外に従来の REXX もサポートされます。オブジェクト REXX は、オブジェクト指向バージョンの REXX 言語です。オブジェクト指向の拡張子が従来の REXX に追加されていますが、既存の関数および命令には変更はありません。オブジェクト REXX インタープリターは、以下のサポートが加えられ、前のバージョンの拡張バージョンとなっています。

- クラス、オブジェクト、およびメソッド
- メッセージングおよびポリモアフィズム
- 単一および複数継承

オブジェクト REXX は、従来の REXX と完全な互換性があります。この節で REXX と述べる場合は、オブジェクト REXX を含むすべての REXX のバージョンのことを言います。

REXX プログラムはプリコンパイルまたはバインドしません。

手順:

AIX 上で DB2 REXX/SQL プログラムを実行するには、DB2 インストール・ディレクトリの下に lib を組み込むように、LIBPATH 環境変数を設定する必要があります。

bash または korn シェルの場合、以下を入力します。

```
export LIBPATH=$LIBPATH:/lib:/usr/lib:/usr/opt/db2_08_01/lib
```

C シェルの場合、以下を入力します。

```
setenv LIBPATH $LIBPATH:/lib:/usr/lib:/usr/opt/db2_08_01/lib
```

AIX 上で、アプリケーション・ファイルには、任意のファイル拡張子を付けることができます。アプリケーションは、次の 2 つの方法で実行することができます。

1. シェル・コマンド・プロンプトで、`rexx name` と入力します。ただし `name` は REXX プログラムの名前 (拡張子があればそれも付けます) です。
2. REXX プログラムの最初の行に「マジック・ナンバー」(#!) が含まれており、それが REXX/6000 インタープリターの常駐するディレクトリを識別する場合は、シェル・コマンド・プロンプトでその名前を入力すれば、REXX プログラムを実行することができます。たとえば、REXX/6000 インタープリター・ファイルが `/usr/bin` ディレクトリにある場合は、次の行を、REXX プログラムの最初の行として組み込みます。

```
#!/usr/bin/rexx
```

そうすれば、シェル・コマンド・プロンプトで次のコマンドを入力することによって、プログラムを実行可能にできます。

```
chmod +x name
```


シェル・コマンド・プロンプトでファイル名を入力することによって、REXX プログラムを実行します。

REXX サンプル・プログラムは、`sqllib/samples/rexx` ディレクトリーにあります。サンプル REXX プログラム `updat.cmd` を実行するには、次のように入力します。

```
updat.cmd
```

関連タスク:

- 36 ページの『UNIX アプリケーション開発環境のセットアップ』

関連資料:

- 102 ページの『REXX のサンプル』

第 11 章 HP-UX

HP-UX C	223	HP-UX での C++ マルチスレッド・アプリケーションの構築	235
C アプリケーションのビルド・スクリプト	223	Micro Focus COBOL	236
HP-UX C アプリケーションのコンパイルとリンクのオプション	224	HP-UX での Micro Focus COBOL コンパイラの構成	236
C ルーチンのビルド・スクリプト	226	Micro Focus COBOL アプリケーションのビルド・スクリプト	237
HP-UX C ルーチンのコンパイルとリンクのオプション	227	HP-UX Micro COBOL アプリケーションのコンパイルとリンクのオプション	238
HP-UX での C マルチスレッド・アプリケーションの構築	229	Micro Focus COBOL ルーチンのビルド・スクリプト	238
HP-UX C++	230	HP-UX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション	239
C++ アプリケーションのビルド・スクリプト	230		
HP-UX C++ アプリケーションのコンパイルとリンクのオプション	231		
C++ ルーチンのビルド・スクリプト	232		
HP-UX C++ ルーチンのコンパイルとリンクのオプション	233		

この章では、HP-UX で DB2 アプリケーションを構築するための詳細な情報を提供します。HP-UX 環境での DB2 アプリケーション開発の最新の更新事項については、次の DB2 アプリケーション開発 Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

HP-UX C

DB2 CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

サポートされる UNIX オペレーティング・システム上で C アプリケーションを構築する方法については、171 ページの『UNIX C アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C ルーチンを構築する方法については、175 ページの『UNIX C ルーチンの構築』を参照してください。

C アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds HP-UX C applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ] ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Determine the HP platform and set correct compile/link options
hpplat=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $hpplat = "ia64" ]; then
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DD64"
        LIB="-lib"
    else

```

```

        EXTRA_CFLAG="+DD32"
        LIB="lib32"
    fi
else
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DA2.0W"
        LIB="lib"
    else
        EXTRA_CFLAG=
        LIB="lib32"
    fi
fi

# The runtime path is recommended for all applications.
# If you need to use SHLIB_PATH or LD_LIBRARY_PATH, unset
# the RUNTIME variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="-Wl,+b$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
# Note: some .sqc files contain no SQL but link in
# utilemb.sqc, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.c error-checking utility.
    cc $EXTRA_CFLAG -Ae -I$DB2PATH/include -c utilemb.c
else
    # Compile the utilapi.c error-checking utility.
    cc $EXTRA_CFLAG -Ae -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc $EXTRA_CFLAG -Ae -I$DB2PATH/include -c $1.c

if [ -f $1".sqc" ]
then
    # Link the program with utilemb.o.
    cc $EXTRA_CFLAG -o $1 $1.o utilemb.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
else
    # Link the program with utilapi.o.
    cc $EXTRA_CFLAG -o $1 $1.o utilapi.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
fi

```

HP-UX C アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、HP-UX C コンパイラを使用して、C 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cc	C コンパイラー。
\$EXTRA_CFLAG	<p>HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 が入り、 32 ビット・サポートが使用可能な場合は、値 +DD32 が入ります。 HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。 PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。</p> <p>+DD64 IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。</p> <p>+DD32 IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。</p> <p>+DA2.0W PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。</p>
-Ae	HP ANSI 拡張モードを使用可能にします。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。 コンパイルとリンクは別個のステップです。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
cc	コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_CFLAG	HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 が入り、 32 ビット・サポートが使用可能な場合は、値 +DD32 が入ります。 HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。 PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。 +DD64 IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。 +DD32 IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。 +DA2.0W PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	非組み込み SQL プログラムの場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを組み込みます。
\$EXTRA_LFLAG	ランタイム・パスを指定します。設定する場合、32 ビットの場合は値 -Wl,+b\$HOME/sql1lib/lib32 、 64 ビットの場合は -Wl,+b\$HOME/sql1lib/lib が入ります。設定しない場合は、これには値が入りません。
-L\$DB2PATH/\$LIB	DB2 ランタイム共有ライブラリーのロケーションを指定します。 32 ビットの場合は \$HOME/sql1lib/lib32 、 64 ビットの場合は \$HOME/sql1lib/lib です。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds HP-UX C applications (C)』

C ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds HP-UX C routines (stored procedures and UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Determine the HP platform and set correct compile/link options
```

```

hpplat=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $hpplat = "ia64" ]; then
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DD64"
        LIB="lib"
    else
        EXTRA_CFLAG="+DD32"
        LIB="lib32"
    fi
else
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DA2.0W"
        LIB="lib"
    else
        EXTRA_CFLAG=
        LIB="lib32"
    fi
fi

# The runtime path is recommended for all applications.
# If you need to use SHLIB_PATH or LD_LIBRARY_PATH, unset
# the RUNTIME variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="+b$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2
fi

# Compile the program.
cc $EXTRA_CFLAG +u1 +z -Ae -I$DB2PATH/include ¥
-D_POSIX_C_SOURCE=199506L -c $1.c

# Link the program to create a shared library
ld -b -o $1 $1.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2 -lpthread

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function

```

HP-UX C ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、HP-UX C コンパイラを使用して、C ルーチン（ストアード・プロシージャとユーザー定義関数）を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
<p>コンパイル・オプション</p> <p>cc C コンパイラー。</p> <p>\$EXTRA_CFLAG</p> <p>HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 が入り、32 ビット・サポートが使用可能な場合は、値 +DD32 が入ります。HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。</p> <p>+DD64 IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。</p> <p>+DD32 IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。</p> <p>+DA2.0W</p> <p>PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。</p> <p>+u1 位置合わせしないデータ・アクセスを認めます。アプリケーションが位置合わせしないデータを使用する場合にのみ使用します。</p> <p>+z 位置に依存しないコードを生成します。</p> <p>-Ae HP ANSI 拡張モードを使用可能にします。</p> <p>-I\$DB2PATH/include</p> <p>DB2 組み込みファイルのロケーションを指定します。たとえば、 -I\$DB2PATH/include。</p> <p>-D_POSIX_C_SOURCE=199506L</p> <p>ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) ときに必要な、_REENTRANT が必ず定義されるようにするための POSIX スレッド・ライブラリー・オプション。</p> <p>-c コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。</p>	
<p>リンク・オプション</p> <p>ld リンクにリンカーを使用します。</p> <p>-b 通常の実行可能ファイルではなく、共用ライブラリーを作成します。</p> <p>-o \$1 出力を、共用ライブラリー・ファイルとして指定します。</p> <p>\$1.o プログラム・オブジェクト・ファイルを指定します。</p> <p>\$EXTRA_LFLAG</p> <p>ランタイム・パスを指定します。設定する場合、32 ビットの場合は値 +b\$HOME/sql1lib/lib32、64 ビットの場合は +b\$HOME/sql1lib/lib が入ります。設定しない場合は、これには値が入りません。</p> <p>-L\$DB2PATH/\$LIB</p> <p>DB2 ランタイム共用ライブラリーのロケーションを指定します。32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。</p> <p>-ldb2 DB2 ライブラリーとリンクします。</p> <p>-lpthread</p> <p>POSIX スレッド・ライブラリーとリンクします。</p> <p>他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。</p>	

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds HP-UX C routines (stored procedures and UDFs) (C)』

HP-UX での C マルチスレッド・アプリケーションの構築

HP-UX には、POSIX スレッド・ライブラリーと DCE スレッド・ライブラリーがあります。DB2 は、POSIX スレッド・ライブラリーを使用するマルチスレッド・アプリケーションだけをサポートします。

HP-UX のマルチスレッド・アプリケーションは、コンパイルするために `_REENTRANT` 定義が必要です。HP-UX の資料では、`-D_POSIX_C_SOURCE=199506L` でコンパイルすることをお勧めします。`_REENTRANT` が定義されていることも必ず確認してください。また、アプリケーションは、`-lpthread` とリンクされていることも必要です。

スクリプト・ファイル `bldmt` には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrrds.sqc` からサンプル・プログラム `dbthrrds` を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル `dbthrrds` が作成されます。sample データベースに対してこの実行可能ファイルを実行するには、次の実行可能ファイル名を入力します。

```
dbthrrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 224 ページの『HP-UX C アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldmt -- Builds HP-UX C multi-threaded applications (C)』
- 『dbthrrds.sqc -- How to use multiple context APIs on UNIX (C)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

HP-UX C++

サポートされる UNIX オペレーティング・システム上で C++ アプリケーションを構築する方法については、179 ページの『UNIX C++ アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C++ ルーチンを構築する方法については、183 ページの『UNIX C++ ルーチンの構築』を参照してください。

C++ アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds HP-UX C++ applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Determine the HP platform and set correct compile/link options
hpplat=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $hpplat = "ia64" ]; then
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DD64 -AA"
        LIB="lib"
    else
        EXTRA_CFLAG="+DD32 -AA"
        LIB="lib32"
    fi
else
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DA2.0W"
        LIB="lib"
    else
        EXTRA_CFLAG=
        LIB="lib32"
    fi
fi

# The runtime path is recommended for all applications.
# If you need to use SHLIB_PATH or LD_LIBRARY_PATH, unset
# the RUNTIME variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="-Wl,+b$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
# Note: some .sqc files contain no SQL but link in
# utilemb.sqc, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1.sqc ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.C error-checking utility.
    aCC $EXTRA_CFLAG -ext -I$DB2PATH/include -c utilemb.C
else
    # Compile the utilapi.C error-checking utility.
    aCC $EXTRA_CFLAG -ext -I$DB2PATH/include -c utilapi.C
```

```

fi

# Compile the program.
aCC $EXTRA_CFLAG -ext -I$DB2PATH/include -c $1.C

if [ -f $1".sqC" ]
then
    # Link the program with utilemb.o.
    aCC $EXTRA_CFLAG -o $1 $1.o utilemb.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
else
    # Link the program with utilapi.o.
    aCC $EXTRA_CFLAG -o $1 $1.o utilapi.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
fi

```

HP-UX C++ アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、HP-UX C++ コンパイラーを使用して、C++ 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
aCC	HP aC++ コンパイラー。
\$EXTRA_CFLAG	HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 -AA が入り、32 ビット・サポートが使用可能な場合は、値 +DD32 -AA が入ります。HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。
+DD64	IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
+DD32	IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。
-AA	IA64 上で、ネーム・スペース std および C++ 標準ライブラリーなどの ANSI C++ 標準機能を許可します。
+DA2.0W	PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
-ext	"long long" サポートを含むさまざまな C++ 拡張子を許可します。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sql1lib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
aCC	HP aC++ コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_CFLAG	HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 -AA が入り、 32 ビット・サポートが使用可能な場合は、値 +DD32 -AA が入ります。 HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。 PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。
+DD64	IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
+DD32	IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。
-AA	IA64 上で、ネーム・スペース std および C++ 標準ライブラリーなどの ANSI C++ 標準機能を許可します。
+DA2.0W	PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	非組み込み SQL プログラムの場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを組み込みます。
\$EXTRA_LFLAG	ランタイム・パスを指定します。設定する場合、32 ビットの場合は値 -Wl,+b\$HOME/sql1lib/lib32 、 64 ビットの場合は -Wl,+b\$HOME/sql1lib/lib が入ります。設定しない場合は、これには値が入りません。
-L\$DB2PATH/\$LIB	DB2 ランタイム共用ライブラリーのロケーションを指定します。 32 ビットの場合は \$HOME/sql1lib/lib32 、 64 ビットの場合は \$HOME/sql1lib/lib です。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds HP-UX C++ applications (C++)』

C++ ルーチンのビルド・スクリプト

```

#!/bin/sh
# SCRIPT: bldrtn
# Builds HP-UX C++ routines (stored procedures and UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
```

```

# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Determine the HP platform and set correct compile/link options
hpplat=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $hpplat = "ia64" ]; then
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DD64 -AA"
        LIB="lib"
    else
        EXTRA_CFLAG="+DD32 -AA"
        LIB="lib32"
    fi
else
    if [ $bitwidth = "¥64¥" ]; then
        EXTRA_CFLAG="+DA2.0W"
        LIB="lib"
    else
        EXTRA_CFLAG=
        LIB="lib32"
    fi
fi

# The runtime path is recommended for all applications.
# If you need to use SHLIB_PATH or LD_LIBRARY_PATH, unset
# the RUNTIME variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="-Wl,+b$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqC" ]
then
    ./embprep $1 $2
fi

# Compile the program. First ensure it is coded with extern "C".
aCC $EXTRA_CFLAG +u1 +z -ext -mt -I$DB2PATH/include -c $1.C

# Link the program to create a shared library.
aCC $EXTRA_CFLAG -mt -b -o $1 $1.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function

```

HP-UX C++ ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、HP-UX C++ コンパイラを使用して、C++ ルーチン（ストアード・プロシーチャーとユーザー定義関数）を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
aCC	HP aC++ コンパイラー。
\$EXTRA_CFLAG	HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 -AA が入り、32 ビット・サポートが使用可能な場合は、値 +DD32 -AA が入ります。HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。
+DD64	IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
+DD32	IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。
-AA	IA64 上で、ネーム・スペース std および C++ 標準ライブラリーなどの ANSI C++ 標準機能を許可します。
+DA2.0W	PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
+u1	位置合わせしないデータ・アクセスを認めます。
+z	位置に依存しないコードを生成します。
-ext	"long long" サポートを含むさまざまな C++ 拡張子を許可します。
-mt	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、HP aC++ コンパイラーのマルチスレッド・サポートを使用可能にします。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、 \$DB2PATH/include。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
aCC	HP aC++ コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_CFLAG	
HP-UX プラットフォームが IA64 で、64 ビット・サポートが使用可能な場合は、このフラグには値 +DD64 -AA が入り、32 ビット・サポートが使用可能な場合は、値 +DD32 -AA が入ります。HP-UX プラットフォームが PA-RISC で、64 ビット・サポートが使用可能な場合は、これには値 +DA2.0W が入ります。PA-RISC プラットフォームでの 32 ビット・サポートの場合は、このフラグには値が入りません。	
+DD64	IA64 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
+DD32	IA64 版の HP-UX 用の 32 ビット・コードを生成する場合に使用する必要があります。
-AA	IA64 上で、ネーム・スペース std および C++ 標準ライブラリーなどの ANSI C++ 標準機能を許可します。
+DA2.0W	PA-RISC 版の HP-UX 用の 64 ビット・コードを生成する場合に使用する必要があります。
-mt	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、HP aC++ コンパイラーのマルチスレッド・サポートを使用可能にします。
-b	通常の実行可能ファイルではなく、共用ライブラリーを作成します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
\$EXTRA_LFLAG	
ランタイム・パスを指定します。設定する場合、32 ビットの場合は値 -Wl,+b\$HOME/sql1lib/lib32 、64 ビットの場合は -Wl,+b\$HOME/sql1lib/lib が入ります。設定しない場合は、これには値が入りません。	
-L\$DB2PATH/\$LIB	
DB2 ランタイム共用ライブラリーのロケーションを指定します。32 ビットの場合は \$HOME/sql1lib/lib32 、64 ビットの場合は \$HOME/sql1lib/lib です。	
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 183 ページの『UNIX C++ ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds HP-UX C++ routines (stored procedures and UDFs) (C++)』

HP-UX での C++ マルチスレッド・アプリケーションの構築

HP-UX には、POSIX スレッド・ライブラリーと DCE スレッド・ライブラリーがあります。HP-UX 上の DB2 は、POSIX スレッド・ライブラリーを使用するマルチスレッド・アプリケーションだけをサポートします。

HP-UX C++ コンパイラーの場合、コンパイルとリンクのどちらのステップでも、マルチスレッド・アプリケーションには **-mt** を使用する必要があります。

スクリプト `bldmt` には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrrds.sqC` からサンプル・プログラム `dbthrrds` を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル `dbthrrds` が作成されます。 `sample` データベースに対してこの実行可能ファイルを実行するには、次の実行可能ファイル名を入力します。

```
dbthrrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 231 ページの『HP-UX C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『`bldmt -- Builds HP-UX C++ multi-threaded applications (C++)`』
- 『`dbthrrds.sqC -- How to use multiple context APIs on UNIX (C++)`』
- 『`embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)`』

Micro Focus COBOL

サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL アプリケーションを構築する方法については、187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL ルーチンを構築する方法については、189 ページの『UNIX Micro Focus COBOL ルーチンの構築』を参照してください。

HP-UX での Micro Focus COBOL コンパイラーの構成

組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合に、Micro Focus COBOL コンパイラーを使用するときは、気をつける点があります。

手順:

- コマンド行プロセッサのコマンド `db2 prep` を使用してアプリケーションをプリコンパイルする場合は、`target mfcob` オプションを使用してください。

- DB2 COBOL COPY ファイル・ディレクトリーを、Micro Focus COBOL 環境変数 COBCPY に含める必要があります。COBCPY 環境変数には、COPY ファイルのロケーションを指定します。Micro Focus COBOL 用の DB2 COPY ファイルは、データベース・インスタンス・ディレクトリーの下にある `sqllib/include/cobol_mf` にあります。

このディレクトリーを組み込むには

- bash または korn シェルでは以下を入力します。

```
export COBCPY=$COBCPY:$HOME/sqllib/include/cobol_mf
```

- C シェルでは以下を入力します。

```
setenv COBCPY ${COBCPY}:${HOME}/sqllib/include/cobol_mf
```

注: COBCPY を `.profile` または `.login` ファイル中に設定することもできます。

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』
- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 238 ページの『HP-UX Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 239 ページの『HP-UX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Micro Focus COBOL アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds HP-UX Micro Focus COBOL applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    ./embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error checking utility.
cob -cx checkerr.cbl

# Compile the program.
cob -cx $1.cbl

# Link the program.
cob -x $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```


HP-UX Micro COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、HP-UX 上で Micro Focus COBOL コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	Micro Focus COBOL コンパイラー。
-cx	オブジェクト・モジュールにコンパイルします。
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
checkerr.o	エラー・チェック用のユーティリティー・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds HP-UX Micro Focus COBOL applications』

Micro Focus COBOL ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds HP-UX Micro Focus COBOL routines (stored procedures)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    embprep $1 $2
fi

# Compile the program.
```

|
|
|
|
|
|
|
|
|
|

```
cob +z -cx $1.cbl

# Link the program.
ld -b -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2gmf ¥
    -L$COBDIR/coblib -lcobol -lcrtn

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

HP-UX Micro Focus COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、HP-UX 上で Micro Focus COBOL コンパイラーを使用して、COBOL ルーチン (ストアード・プロシージャ) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	COBOL コンパイラー。
+z	位置に依存しないコードを生成します。
-cx	オブジェクト・モジュールにコンパイルします。
リンク・オプション	
ld	リンクにリンカーを使用します。
-b	通常の実行可能ファイルではなく、共用ライブラリーを作成します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。
-ldb2	DB2 共用ライブラリーにリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
-L\$COBDIR/coblib	COBOL ランタイム・ライブラリーのロケーションを指定します。
-lcobol	COBOL ライブラリーにリンクします。
-lcrtn	crtn ライブラリーにリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds HP-UX Micro Focus COBOL routines (stored procedures)』

第 12 章 Linux

Linux C	241	Linux での C++ マルチスレッド・アプリケーションの構築	251
C アプリケーションのビルド・スクリプト	241	Micro Focus COBOL	251
Linux C アプリケーションのコンパイルとリンクのオプション.	242	Linux での Micro Focus COBOL コンパイラの構成.	252
C ルーチンのビルド・スクリプト	244	Micro Focus COBOL アプリケーションのビルド・スクリプト.	253
Linux C ルーチンのコンパイルとリンクのオプション.	244	Linux Micro COBOL アプリケーションのコンパイルとリンクのオプション.	254
Linux での C マルチスレッド・アプリケーションの構築	246	Micro Focus COBOL ルーチンのビルド・スクリプト	254
Linux C++	246	Linux Micro Focus COBOL ルーチンのコンパイルとリンクのオプション.	255
C++ アプリケーションのビルド・スクリプト	246		
Linux C++ アプリケーションのコンパイルとリンクのオプション.	247		
C++ ルーチンのビルド・スクリプト.	249		
Linux C++ ルーチンのコンパイルとリンクのオプション.	249		

この章は、Linux でアプリケーションを構築するための詳細な情報を提供します。Linux 環境での DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

Linux C

DB2 CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

サポートされる UNIX オペレーティング・システム上で C アプリケーションを構築する方法については、171 ページの『UNIX C アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C ルーチンを構築する方法については、175 ページの『UNIX C ルーチンの構築』を参照してください。

C アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Linux C applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ] ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Determine if we are running with 32-bit, and
# if we are running with 32-bit on Linux AMD64
LIB="lib"
EXTRA_C_FLAGS=""
HARDWAREPLAT=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB="lib32"
```

```

        if [ "$HARDWAREPLAT" = "x86_64" ]; then
            EXTRA_C_FLAGS="-m32"
        fi

# The runtime path is recommended for all applications.
# If you need to use LD_LIBRARY_PATH, unset the RUNTIME
# variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="-Wl,-rpath,$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
# Note: some .sqc files contain no SQL but link in
# utilemb.sqc, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.c error-checking utility.
    gcc $EXTRA_C_FLAGS -I$DB2PATH/include -c utilemb.c
else
    # Compile the utilapi.c error-checking utility.
    gcc $EXTRA_C_FLAGS -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
gcc $EXTRA_C_FLAGS -I$DB2PATH/include -c $1.c

if [ -f $1".sqc" ]
then
    # Link the program with utilemb.o.
    gcc $EXTRA_C_FLAGS -o $1 $1.o utilemb.o $EXTRA_LFLAG ¥
    -L$DB2PATH/$LIB -ldb2
else
    # Link the program with utilapi.o.
    gcc $EXTRA_C_FLAGS -o $1 $1.o utilapi.o $EXTRA_LFLAG ¥
    -L$DB2PATH/$LIB -ldb2
fi

```

Linux C アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Linux C コンパイラーを使用して、C 組み込み SQL および DB2 API アプリケーションを構築するのに
お勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
gcc	GNU/Linux C コンパイラー。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。 このスクリプト・ファイルでは、コンパイルとリンクは別個のステップです。
リンク・オプション	
gcc	コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-o \$1	実行可能ファイルを指定します。
\$1.o	オブジェクト・ファイルを指定します。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	非組み込み SQL プログラムの場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを組み込みます。
\$EXTRA_LFLAG	「RUNTIME=true」をコメント解除すると、32 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib32」、64 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib」が入ります。それ以外の場合は、これには値が入りません。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。 -L オプションを指定しないと、コンパイラーはパスとして /usr/lib:/lib を想定します。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds Linux C applications (C)』

C ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds Linux C routines (stored procedures or UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql/lib

# Determine if we are running with 32-bit, and
# if we are running with 32-bit on Linux AMD64
LIB="lib"
EXTRA_C_FLAGS=""
HARDWAREPLAT=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB="lib32"
    if [ "$HARDWAREPLAT" = "x86_64" ]; then
        EXTRA_C_FLAGS="-m32"
    fi
fi

# Set the runtime path.
EXTRA_LFLAG="-Wl,-rpath,$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sql" ]
then
    ./embprep $1 $2
fi

# Compile the program.
gcc $EXTRA_C_FLAGS -fpic -I$DB2PATH/include -c $1.c -D_REENTRANT

# Link the program and create a shared library
gcc $EXTRA_C_FLAGS -shared -o $1 $1.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2 -lpthread

# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

Linux C ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、Linux C コンパイラーを使用して、C ルーチン (ストアード・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
<p>コンパイル・オプション</p> <p>gcc GNU/Linux C コンパイラー。</p> <p>\$EXTRA_C_FLAGS AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。</p> <p>-fpic 位置に依存しないコードを生成します。</p> <p>-I\$DB2PATH/include DB2 組み込みファイルのロケーションを指定します。</p> <p>-c コンパイルのみを実行し、リンクは実行しません。 このスクリプト・ファイルでは、コンパイルとリンクは別個のステップです。</p> <p>-D_REENTRANT ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、_REENTRANT を定義します。</p>	
<p>リンク・オプション</p> <p>gcc コンパイラーをリンカーのフロントエンドとして使用します。</p> <p>\$EXTRA_CFLAG AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。</p> <p>-shared 共用ライブラリーを生成します。</p> <p>-o \$1 実行可能ファイルを指定します。</p> <p>\$1.o プログラム・オブジェクト・ファイルを組み込みます。</p> <p>\$EXTRA_LFLAG ランタイムの DB2 共用ライブラリーのロケーションを示します。 32 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib32」が入ります。 64 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib」が入ります。</p> <p>-L\$DB2PATH/\$LIB リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、 64 ビットの場合は \$HOME/sql1lib/lib です。 -L オプションを指定しないと、コンパイラーはパスとして /usr/lib:/lib を想定します。</p> <p>-ldb2 DB2 ライブラリーとリンクします。</p> <p>-lpthread POSIX スレッド・ライブラリーとリンクします。</p> <p>他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。</p>	

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds Linux C routines (stored procedures or UDFs) (C)』

Linux での C マルチスレッド・アプリケーションの構築

Linux C を使用するマルチスレッド・アプリケーションは、`-D_REENTRANT` でコンパイルし、`-lpthread` とリンクする必要があります。

スクリプト `bldmt` には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrrds.sqc` からサンプル・プログラム `dbthrrds` を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル `dbthrrds` が作成されます。sample データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
dbthrrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 242 ページの『Linux C アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『`bldmt -- Builds Linux C multi-threaded applications (C)`』
- 『`dbthrrds.sqc -- How to use multiple context APIs on UNIX (C)`』
- 『`embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)`』

Linux C++

サポートされる UNIX オペレーティング・システム上で C++ アプリケーションを構築する方法については、179 ページの『UNIX C++ アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C++ ルーチンを構築する方法については、183 ページの『UNIX C++ ルーチンの構築』を参照してください。

C++ アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Linux C++ applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ] ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib
```

```

# Determine if we are running with 32-bit, and
# if we are running with 32-bit on Linux AMD64
LIB="lib"
EXTRA_C_FLAGS=""
HARDWAREPLAT=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB="lib32"
    if [ "$HARDWAREPLAT" = "x86_64" ]; then
        EXTRA_C_FLAGS="-m32"
    fi
fi

# The runtime path is recommended for all applications.
# If you need to use LD_LIBRARY_PATH, unset the RUNTIME
# variable by commenting out the following line.
RUNTIME=true

if [ "$RUNTIME" != "" ]
then
    EXTRA_LFLAG="-Wl,-rpath,$DB2PATH/$LIB"
else
    EXTRA_LFLAG=""
fi

# If an embedded SQL program, precompile and bind it.
# Note: some .sqC files contain no SQL but link in
# utilemb.sqC, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqC" ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.C error-checking utility.
    g++ $EXTRA_C_FLAGS -I$DB2PATH/include -c utilemb.C
else
    # Compile the utilapi.C error-checking utility.
    g++ $EXTRA_C_FLAGS -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
g++ $EXTRA_C_FLAGS -I$DB2PATH/include -c $1.C

if [ -f $1".sqC" ]
then
    # Link the program with utilemb.o
    g++ $EXTRA_C_FLAGS -o $1 $1.o utilemb.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
else
    # Link the program with utilapi.o
    g++ $EXTRA_C_FLAGS -o $1 $1.o utilapi.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2
fi

```

Linux C++ アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Linux C++ コンパイラを使用して、C++ 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
g++	GNU/Linux C++ コンパイラー。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプト・ファイルでは、コンパイルとリンクは別個のステップです。
リンク・オプション	
g++	コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	非組み込み SQL プログラムの場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを組み込みます。
\$EXTRA_LFLAG	「RUNTIME=true」をコメント解除すると、32 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib32」、64 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib」が入ります。それ以外の場合は、これには値が入りません。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。-L オプションを指定しないと、コンパイラーはパスとして /usr/lib:/lib を想定します。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds Linux C++ applications (C++)』

C++ ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds Linux C++ routines (stored procedures and UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql/lib

# Determine if we are running with 32-bit, and
# if we are running with 32-bit on Linux AMD64
LIB="lib"
EXTRA_C_FLAGS=""
HARDWAREPLAT=`uname -m`
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥32¥" ]; then
    LIB="lib32"
    if [ "$HARDWAREPLAT" = "x86_64" ]; then
        EXTRA_C_FLAGS="-m32"
    fi
fi

# Set the runtime path.
EXTRA_LFLAG="-Wl,-rpath,$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2
fi

# Compile the program.
g++ $EXTRA_C_FLAGS -fpic -I$DB2PATH/include -c $1.C -D_REENTRANT

# Link the program and create a shared library.
g++ $EXTRA_C_FLAGS -shared -o $1 $1.o $EXTRA_LFLAG -L$DB2PATH/$LIB -ldb2 -lpthread

# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

Linux C++ ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、Linux C++ コンパイラーを使用して、C++ ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
g++	GNU/Linux C++ コンパイラー。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-fpic	位置に依存しないコードを生成します。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプト・ファイルでは、コンパイルとリンクは別個のステップです。
-D_REENTRANT	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、_REENTRANT を定義します。
リンク・オプション	
g++	コンパイラーをリンカーのフロントエンドとして使用します。
\$EXTRA_C_FLAGS	AMD64 での 32 ビット・サポートの場合は「-m32」が入ります。それ以外の場合は、これには値が入りません。
-shared	共用ライブラリーを生成します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
\$EXTRA_LFLAG	ランタイムの DB2 共用ライブラリーのロケーションを示します。 32 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib32」が入ります。 64 ビットの場合は値「-Wl,-rpath,\$DB2PATH/lib」が入ります。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、 64 ビットの場合は \$HOME/sql1lib/lib です。 -L オプションを指定しないと、コンパイラーはパスとして /usr/lib:/lib を想定します。
-ldb2	DB2 ライブラリーとリンクします。
-lpthread	POSIX スレッド・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 183 ページの『UNIX C++ ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds Linux C++ routines (stored procedures and UDFs) (C++)』

Linux での C++ マルチスレッド・アプリケーションの構築

Linux C++ を使用するマルチスレッド・アプリケーションは、`-D_REENTRANT` でコンパイルし、`-lpthread` とリンクする必要があります。

スクリプト・ファイル `bldmt` には、組み込み SQL マルチスレッド・プログラムを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrds.sqC` からサンプル・プログラム `dbthrds` を構築するには、次のように入力します。

```
bldmt dbthrds
```

結果として、実行可能ファイル `dbthrds` が作成されます。sample データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
dbthrds
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 247 ページの『Linux C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldmt -- Builds Linux C++ multi-threaded applications (C++)』
- 『dbthrds.sqC -- How to use multiple context APIs on UNIX (C++)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

Micro Focus COBOL

Micro Focus COBOL は、次のアーキテクチャーの Linux でのみサポートされます。

- **Intel x86 (32 ビット) 版 Linux**
- **s/390 版 Linux**

サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL アプリケーションを構築する方法については、187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL ルーチンを構築する方法については、189 ページの『UNIX Micro Focus COBOL ルーチンの構築』を参照してください。

Linux での Micro Focus COBOL コンパイラーの構成

手順:

Micro Focus COBOL ルーチンを実行するには、Linux ランタイム・リンカーが特定の COBOL 共用ライブラリーにアクセスできる必要があります、しかも DB2 がそのライブラリーをロードできる必要があります。そのロードを行うプログラムは、setuid 特権のもとで実行されるので、/usr/lib 内の従属ライブラリーだけが探索されることになります。

COBOL 共用ライブラリー用に /usr/lib へのシンボリック・リンクを作成します。これは root として実行する必要があります。これを行う最も簡単な方法は、次のようにして、すべての COBOL ライブラリー・ファイルを \$COBDIR/lib から /usr/lib にリンクするという方法です。

```
ln -s $COBDIR/lib/libcob* /usr/lib
```

\$COBDIR は Micro Focus COBOL のインストール先で、通常は /opt/lib/mfcobol です。

以下は、各個別ファイルをリンクするためのコマンドです (Micro Focus COBOL は /opt/lib/mfcobol にインストールされていると想定しています)。

```
ln -s /opt/lib/mfcobol/lib/libcobrts.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobrts_t.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobrts.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobrts_t.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobcrtn.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobcrtn.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobmisc.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobmisc_t.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobmisc.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobmisc_t.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobscreen.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobscreen.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobtrace.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobtrace_t.so /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobtrace.so.2 /usr/lib
ln -s /opt/lib/mfcobol/lib/libcobtrace_t.so.2 /usr/lib
```

各 DB2 インスタンスで、次のようにする必要があります。

- コマンド行プロセッサのコマンド db2 prep を使用してアプリケーションをプリコンパイルする場合は、target mfcob オプションを使用してください。
- DB2 COBOL COPY ファイル・ディレクトリーを、Micro Focus COBOL 環境変数 COBCPY に含める必要があります。COBCPY 環境変数は、COPY ファイルのロケーションを指定します。Micro Focus COBOL 用の DB2 COPY ファイルは、データベース・インスタンス・ディレクトリーの下にある sqllib/include/cobol_mf にあります。

このディレクトリーを含めるには、次のように入力します。

– bash または korn シェルの場合

```
export COBCPY=$HOME/sqllib/include/cobol_mf:$COBDIR/cpylib
```

– C シェルの場合

```
setenv COBCPY $HOME/sqllib/include/cobol_mf:$COBDIR/cpylib
```

- 環境変数を次のように更新します。
 - bash または korn シェルの場合


```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/sql1lib/lib:$COBDIR/lib
```
 - C シェルの場合


```
setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH:$HOME/sql1lib/lib:$COBDIR/lib
```
- DB2 環境リストを設定します。


```
db2set DB2ENVLIST="COBDIR LD_LIBRARY_PATH"
```

注: COBCPY、COBDIR、および LD_LIBRARY_PATH を、.bashrc、.kshrc (使用中のシェルによって異なる)、.bash_profile、.profile (使用中のシェルによって異なる)、または .login に設定することもできます。

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』
- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 254 ページの『Linux Micro COBOL アプリケーションのコンパイルとリンクのオプション』
- 255 ページの『Linux Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Micro Focus COBOL アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Linux Micro Focus COBOL applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
  ./embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error checking utility.
cob -cx checkerr.cbl

# Compile the program.
cob -cx $1.cbl

# Link the program.
cob -x -o $1 $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```


Linux Micro COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Linux 上で Micro Focus COBOL コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	Micro Focus COBOL コンパイラー。
-cx	オブジェクト・モジュールにコンパイルします。
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	実行可能プログラムを指定します。
-o \$1	実行可能ファイルを組み込みます。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
checkerr.o	エラー・チェック用のユーティリティー・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』
- 252 ページの『Linux での Micro Focus COBOL コンパイラーの構成』

関連サンプル:

- 『bldapp -- Builds Linux Micro Focus COBOL applications』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

Micro Focus COBOL ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds Linux Micro Focus COBOL routines (stored procedures)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# Precompile and bind the program.
```

```
./embprep $1 $2

# Compile the program.
cob -cx $1.cbl

# Link the program.
cob -x -o $1 $1.o -Q -G -L$DB2PATH/lib -ldb2 -ldb2gmf

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

Linux Micro Focus COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、Linux 上で Micro Focus COBOL コンパイラーを使用して、COBOL ルーチン (ストアド・プロシージャ) を構築するのに勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	COBOL コンパイラー。
-cx	オブジェクト・モジュールにコンパイルします。
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	実行可能プログラムを指定します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
-Q -G	共用ライブラリーを生成します。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』
- 252 ページの『Linux での Micro Focus COBOL コンパイラーの構成』

関連サンプル:

- 『bldrtn -- Builds Linux Micro Focus COBOL routines (stored procedures)』
- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

第 13 章 Solaris

Solaris C	257	Solaris での C++ マルチスレッド・アプリケーションの構築	267
C アプリケーションのビルド・スクリプト	257	Micro Focus COBOL	268
Solaris C アプリケーションのコンパイルとリンクのオプション	258	Solaris での Micro Focus COBOL コンパイラの構成	268
C ルーチンのビルド・スクリプト	259	Micro Focus COBOL アプリケーションのビルド・スクリプト	268
Solaris C ルーチンのコンパイルとリンクのオプション	260	Solaris Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション	269
Solaris での C マルチスレッド・アプリケーションの構築	261	Micro Focus COBOL ルーチンのビルド・スクリプト	270
Solaris C++	262	Solaris Micro Focus COBOL ルーチンのコンパイルとリンクのオプション	270
C++ アプリケーションのビルド・スクリプト	262		
Solaris C++ アプリケーションのコンパイルとリンクのオプション	263		
C++ ルーチンのビルド・スクリプト	265		
Solaris C++ ルーチンのコンパイルとリンクのオプション	265		

この章は、Solaris オペレーティング環境でアプリケーションを構築するための詳細な情報を提供します。Solaris 環境での DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

Solaris C

DB2 CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

サポートされる UNIX オペレーティング・システム上で C アプリケーションを構築する方法については、171 ページの『UNIX C アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C ルーチンを構築する方法については、175 ページの『UNIX C ルーチンの構築』を参照してください。

C アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Solaris C applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql/lib

# Set compile and link flags for 32-bit and 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥64¥" ];
then
    CFLAG_ARCH=v9
    LIB=lib
else
    CFLAG_ARCH=v8plusa
```

```
LIB=lib32
fi

# Set the runtime path.
# LD_LIBRARY_PATH will be followed instead of the runtime path unless
# you unset LD_LIBRARY_PATH first to allow the runtime path to be used.
EXTRA_LFLAG="-R$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
# Note: some .sqc files contain no SQL but link in
# utilemb.sqc, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.c error-checking utility.
    cc -xarch=$CFLAG_ARCH -I$DB2PATH/include -c utilemb.c
else
    # Compile the utilapi.c error-checking utility.
    cc -xarch=$CFLAG_ARCH -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc -xarch=$CFLAG_ARCH -I$DB2PATH/include -c $1.c

if [ -f $1".sqc" ]
then
    # Link the program with utilemb.o
    cc -xarch=$CFLAG_ARCH -mt -o $1 $1.o utilemb.o ¥
    -L$DB2PATH/$LIB $EXTRA_LFLAG -ldb2
else
    # Link the program with utilapi.o
    cc -xarch=$CFLAG_ARCH -mt -o $1 $1.o utilapi.o ¥
    -L$DB2PATH/$LIB $EXTRA_LFLAG -ldb2
fi
```

Solaris C アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Forte C コンパイラーを使用して、C 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cc	C コンパイラー。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sql1lib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプトでは、コンパイルとリンクは別個のステップです。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
cc	コンパイラーをリンカーのフロントエンドとして使用します。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	マルチスレッド・サポートにリンクします。libdb2 を使ったリンクに必要です。 注: POSIX スレッドを使用する場合、DB2 アプリケーションは、スレッド化されていなくても -lpthread にリンクする必要もあります。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	組み込み SQL プログラムでない場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを含みます。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。
\$EXTRA_LFLAG	ランタイムの DB2 共用ライブラリーのロケーションを示します。32 ビットの場合は値 "-R\$DB2PATH/lib32" が入り、64 ビットの場合は値 "-R\$DB2PATH/lib" が入ります。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 171 ページの『UNIX C アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds Solaris C applications (C)』

C ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds Solaris C routines (stored procedures or UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Set compile and link flags for 32-bit and 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥64¥" ];
```

```

then
    CFLAG_ARCH=v9
    LIB=lib
else
    CFLAG_ARCH=v8plusa
    LIB=lib32
fi

# Set the runtime path.
# LD_LIBRARY_PATH will be followed instead of the runtime path unless
# you unset LD_LIBRARY_PATH first to allow the runtime path to be used.
EXTRA_LFLAG="-R$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2
fi

# Compile the program.
cc -xarch=$CFLAG_ARCH -mt -DUSE_UI_THREADS -Kpic ¥
    -I$DB2PATH/include -c $1.c

# Link the program and create a shared library
cc -xarch=$CFLAG_ARCH -mt -G -o $1 $1.o -L$DB2PATH/$LIB ¥
    $EXTRA_LFLAG -l$db2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function

```

Solaris C ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、Forte C コンパイラーを使用して、C ルーチン (ストアード・プロシージャとユーザー定義関数) を構築するのに勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cc	C コンパイラー。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、マルチスレッド・サポートを使用可能にします。
-DUSE_UI_THREADS	Sun の「UNIX International」スレッド API を使用可能にします。
-Kpic	共用ライブラリー用の位置に依存しないコードを生成します。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプトでは、コンパイルとリンクは別個のステップです。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
cc	コンパイラーをリンカーのフロントエンドとして使用します。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	これが必要なのは、DB2 ライブラリーは -mt にリンクされているからです。
-G	共用ライブラリーを生成します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。
\$EXTRA_LFLAG	ランタイムの DB2 共用ライブラリーのロケーションを示します。32 ビットの場合は値 "-R\$DB2PATH/lib32" が入り、64 ビットの場合は値 "-R\$DB2PATH/lib" が入ります。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 175 ページの『UNIX C ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds Solaris C routines (stored procedures or UDFs) (C)』

Solaris での C マルチスレッド・アプリケーションの構築

SUN および POSIX のスレッド・ライブラリーを使用するマルチスレッド・アプリケーションが、DB2 でサポートされます。デフォルトは Sun スレッドです。Solaris 上で Forte C を使用するマルチスレッド・アプリケーションは、-mt を使用してコンパイルしてリンクする必要があります。これは、-D_REENTRANT をプリプロセッサに渡し、-lthread をリンカーに渡します。また、Sun の Unix International スレッド API を使用するには、コンパイル定義 -DUSE_UI_THREADS を指定する必要があります。

注: POSIX スレッドを使用したい場合は、getpwnam_r() などの関数の、POSIX の変形を許可するコンパイラー・オプション -D_POSIX_PTHREAD_SEMANTICS を追加し、さらにリンク・オプション -lpthread を追加する必要があります。既成の bldmt スクリプトを使用する場合、-DUSE_UI_THREADS 定義の削除も必要です。

スクリプト `bldmt` には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrrds.sqc` からサンプル・プログラム `dbthrrds` を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル `dbthrrds` が作成されます。 `sample` データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
dbthrrds
```

注: かなりの数の接続を持つマルチスレッド・プログラムの場合は、カーネル・パラメーター `semsys:seminfo_semume` と `shmsys:shminfo_shmseg` をデフォルトの値よりも大きく設定する必要がある場合があります。以下の `db2osconf` ユーティリティー関連のリンクを参照して、これらのパラメーターの推奨値を確認してください。

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 258 ページの『Solaris C アプリケーションのコンパイルとリンクのオプション』
- 「コマンド・リファレンス」の『`db2osconf` - カーネル・パラメーター値のためのユーティリティー・コマンド』

関連サンプル:

- 『`bldmt` -- Builds Solaris C multi-threaded applications (C)』
- 『`dbthrrds.sqc` -- How to use multiple context APIs on UNIX (C)』
- 『`embprep` -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

Solaris C++

サポートされる UNIX オペレーティング・システム上で C++ アプリケーションを構築する方法については、179 ページの『UNIX C++ アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で C++ ルーチンを構築する方法については、183 ページの『UNIX C++ ルーチンの構築』を参照してください。

C++ アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Solaris C++ applications
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]
```

```

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set compile and link flags for 32-bit and 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥64¥" ];
then
    CFLAG_ARCH=v9
    LIB=lib
else
    CFLAG_ARCH=v8plusa
    LIB=lib32
fi

# Set the runtime path.
# LD_LIBRARY_PATH will be followed instead of the runtime path unless
# you unset LD_LIBRARY_PATH first to allow the runtime path to be used.
EXTRA_LFLAG="-R$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
# Note: some .sqC files contain no SQL but link in
# utilemb.sqC, so if you get this warning, ignore it:
# SQL0053W No SQL statements were found in the program.
if [ -f $1".sqC" ]
then
    ./embprep $1 $2 $3 $4
    # Compile the utilemb.C error-checking utility.
    CC -xarch=$CFLAG_ARCH -I$DB2PATH/include -c utilemb.C
else
    # Compile the utilapi.C error-checking utility.
    CC -xarch=$CFLAG_ARCH -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
CC -xarch=$CFLAG_ARCH -I$DB2PATH/include -c $1.C

if [ -f $1".sqC" ]
then
    # Link the program with utilemb.o
    CC -xarch=$CFLAG_ARCH -mt -o $1 $1.o utilemb.o ¥
    -L$DB2PATH/$LIB $EXTRA_LFLAG -ldb2
else
    # Link the program with utilapi.o
    CC -xarch=$CFLAG_ARCH -mt -o $1 $1.o utilapi.o ¥
    -L$DB2PATH/$LIB $EXTRA_LFLAG -ldb2
fi

```

Solaris C++ アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Forte C++ コンパイラを使用して、C++ 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
CC	C++ コンパイラー。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。たとえば、\$HOME/sql1lib/include のように指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプトでは、コンパイルとリンクは別個のステップです。
リンク・オプション	
CC	コンパイラーをリンカーのフロントエンドとして使用します。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	マルチスレッド・サポートにリンクします。libdb2 を使ったリンクに必要です。 注: POSIX スレッドを使用する場合、DB2 アプリケーションは、スレッド化されていてもいなくても -lpthread にリンクする必要もあります。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
utilemb.o	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.o	非組み込み SQL プログラムの場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。
\$EXTRA_LFLAG	ランタイムの DB2 共用ライブラリーのロケーションを示します。32 ビットの場合は値 "-R\$DB2PATH/lib32" が入り、64 ビットの場合は値 "-R\$DB2PATH/lib" が入ります。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 179 ページの『UNIX C++ アプリケーションの構築』

関連サンプル:

- 『bldapp -- Builds Solaris C++ applications (C++)』

C++ ルーチンのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldrtn
# Builds Solaris C++ routines (stored procedures or UDFs)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set compile and link flags for 32-bit and 64-bit programs.
bitwidth=`LANG=C db2level | awk '/bits/{print $5}'`
if [ $bitwidth = "¥64¥" ];
then
    CFLAG_ARCH=v9
    LIB=lib
else
    CFLAG_ARCH=v8plusa
    LIB=lib32
fi

# Set the runtime path.
# LD_LIBRARY_PATH will be followed instead of the runtime path unless
# you unset LD_LIBRARY_PATH first to allow the runtime path to be used.
EXTRA_LFLAG="-R$DB2PATH/$LIB"

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqc" ]
then
    ./embprep $1 $2
fi

# Compile the program.
CC -xarch=$CFLAG_ARCH -mt -DUSE_UI_THREADS -Kpic ¥
    -I$DB2PATH/include -c $1.C

# Link the program and create a shared library
CC -xarch=$CFLAG_ARCH -mt -G -o $1 $1.o -L$DB2PATH/$LIB ¥
    $EXTRA_LFLAG -l$db2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

Solaris C++ ルーチンのコンパイルとリンクのオプション

以下は、bldrtn ビルド・スクリプトに示されているように、Forte C++ コンパイラを使用して、C++ ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
CC	C++ コンパイラー。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	ルーチンを他のルーチンと同じプロセスで実行する (THREADSAFE) か、またはエンジンそのもので実行する (NOT FENCED) かに応じて、マルチスレッド・サポートを使用可能にします。
-DUSE_UI_THREADS	Sun の「UNIX International」スレッド API を使用可能にします。
-Kpic	共用ライブラリー用の位置に依存しないコードを生成します。
-I\$DB2PATH/include	DB2 組み込みファイルのロケーションを指定します。
-c	コンパイルのみを実行し、リンクは実行しません。このスクリプトでは、コンパイルとリンクは別個のステップです。
リンク・オプション	
CC	コンパイラーをリンカーのフロントエンドとして使用します。
-xarch=\$CFLAG_ARCH	このオプションを使用すると、libdb2.so へのリンク時に必ず正しい実行可能ファイルがコンパイラーで生成されるようにすることができます。\$CFLAG_ARCH の値は、32 ビットの場合は v8plusa に、64 ビットの場合は v9 に設定されます。
-mt	これが必要なのは、DB2 ライブラリーは -mt にリンクされているからです。
-G	共用ライブラリーを生成します。
-o \$1	実行可能ファイルを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/\$LIB	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、32 ビットの場合は \$HOME/sql1lib/lib32、64 ビットの場合は \$HOME/sql1lib/lib です。
\$EXTRA_LFLAG	ランタイムの DB2 共用ライブラリーのロケーションを示します。32 ビットの場合は値 "-R\$DB2PATH/lib32" が入り、64 ビットの場合は値 "-R\$DB2PATH/lib" が入ります。
-ldb2	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 183 ページの『UNIX C++ ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds Solaris C++ routines (stored procedures or UDFs) (C++)』

Solaris での C++ マルチスレッド・アプリケーションの構築

SUN および POSIX のスレッド・ライブラリーを使用するマルチスレッド・アプリケーションが、DB2 でサポートされます。デフォルトは Sun スレッドです。

Solaris 上で Forte C++ を使用するマルチスレッド・アプリケーションは、`-mt` を使用してコンパイルしてリンクする必要があります。これは、`-D_REENTRANT` をプリプロセッサに渡し、`-lthread` をリンカーに渡します。また、Sun の Unix International スレッド API を使用するには、コンパイル定義 `-DUSE_UI_THREADS` を指定する必要があります。

注: POSIX スレッドを使用したい場合は、`getpwnam_r()` などの関数の、POSIX の変形を許可するコンパイラー・オプション `-D_POSIX_PTHREAD_SEMANTICS` を追加し、さらにリンク・オプション `-lpthread` を追加する必要があります。既成の `bldmt` スクリプトを使用する場合、`-DUSE_UI_THREADS` 定義の削除も必要です。

スクリプト `bldmt` には、マルチスレッド・アプリケーションを構築するためのコマンドが入っています。上記のオプションのほか、コンパイルおよびリンク・オプションは、組み込み SQL スクリプト・ファイル `bldapp` で使用されているものと同じです。

手順:

ソース・ファイル `dbthrrds.sqC` からサンプル・プログラム `dbthrrds` を構築するには、次のように入力します。

```
bldmt dbthrrds
```

結果として、実行可能ファイル `dbthrrds` が作成されます。sample データベースに対してこの実行可能ファイルを実行するには、次のように入力します。

```
dbthrrds
```

注: かなりの数の接続を持つマルチスレッド・プログラムの場合は、カーネル・パラメーター `semsys:seminfo_semum` と `shmsys:shminfo_shmseg` をデフォルトの値よりも大きく設定する必要がある場合があります。以下の `db2osconf` ユーティリティー関連のリンクを参照して、これらのパラメーターの推奨値を確認してください。

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 75 ページの『C サンプル』
- 263 ページの『Solaris C++ アプリケーションのコンパイルとリンクのオプション』
- 「コマンド・リファレンス」の『`db2osconf` - カーネル・パラメーター値のためのユーティリティー・コマンド』

関連サンプル:

- 『`bldmt` -- Builds Solaris C++ multi-threaded applications (C++)』
- 『`dbthrrds.sqC` -- How to use multiple context APIs on UNIX (C++)』

- 『embprep -- To prep and bind C/C++ and Micro Focus COBOL embedded SQL programs (C)』

Micro Focus COBOL

サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL アプリケーションを構築する方法については、187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』を参照してください。サポートされる UNIX オペレーティング・システム上で Micro Focus COBOL ルーチンを構築する方法については、189 ページの『UNIX Micro Focus COBOL ルーチンの構築』を参照してください。

Solaris での Micro Focus COBOL コンパイラーの構成

組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合に、Micro Focus COBOL コンパイラーを使用するには、気を付けなければならない点があります。

手順:

- コマンド行プロセッサのコマンド `db2 prep` を使用してアプリケーションをブリコンパイルする場合は、`target mfcob` オプションを使用してください。
- DB2 COBOL COPY ファイル・ディレクトリーを、Micro Focus COBOL 環境変数 `COBCPY` に含める必要があります。 `COBCPY` 環境変数には、`COPY` ファイルのロケーションを指定します。Micro Focus COBOL 用の DB2 COPY ファイルは、データベース・インスタンス・ディレクトリーの下にある `sqllib/include/cobol_mf` にあります。

このディレクトリーを含めるには、次のように入力します。

– bash または korn シェルの場合

```
export COBCPY=$COBCPY:$HOME/sqllib/include/cobol_mf
```

– C シェルの場合

```
setenv COBCPY $COBCPY:$HOME/sqllib/include/cobol_mf
```

注: `COBCPY` を `.profile` ファイル中に設定することもできます。

関連タスク:

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』
- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連資料:

- 269 ページの『Solaris Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 270 ページの『Solaris Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Micro Focus COBOL アプリケーションのビルド・スクリプト

```
#!/bin/sh
# SCRIPT: bldapp
# Builds Solaris Micro Focus COBOL applications
```

```
# Usage: bldapp [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sql1lib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    ./embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error-checking utility.
cob -cx checkerr.cbl

# Compile the program.
cob -cx $1.cbl

# Link the program.
cob -x $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```

Solaris Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp ビルド・スクリプトに示されているように、Solaris 上で Micro Focus COBOL コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	Micro Focus COBOL コンパイラー。
-cx	オブジェクト・モジュールにコンパイルします。
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-x	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを組み込みます。
checkerr.o	エラー・チェック用のユーティリティー・オブジェクト・ファイルを組み込みます。
-L\$DB2PATH/lib	リンク時の DB2 静的ライブラリーおよび共用ライブラリーのロケーションを示します。たとえば、\$HOME/sql1lib/lib。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

- 187 ページの『UNIX Micro Focus COBOL アプリケーションの構築』

- 『bldapp -- Builds Solaris Micro Focus COBOL applications』

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
84

```

#!/bin/sh
# SCRIPT: bldrtn
# Builds Solaris Micro Focus COBOL routines (stored procedures)
# Usage: bldrtn <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Set COBCPY to include the DB2 COPY files directory.
COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# If an embedded SQL program, precompile and bind it.
if [ -f $1".sqb" ]
then
    ./embprep $1 $2
fi

# Compile the program.
cob -cx $1.cbl

# Link the program.
cob -yo $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2gmf

# Copy the shared library to the sqllib/function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function

```

以下は、blldrtn ビルド・スクリプトに示されているように、Solaris 上で Micro Focus COBOL コンパイラを使用して、COBOL ルーチン (ストアード・プロシージャ) を構築するのに勧めするコンパイルとリンクのオプションです。

bldrtm のコンパイルとリンクのオプション	
コンパイル・オプション	
cob	COBOL コンパイラー。
-cx	オブジェクト・モジュールにコンパイルします。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
cob	コンパイラーをリンカーのフロントエンドとして使用します。
-y	必要なものを完備したスタンドアロン共用ライブラリーを作成します。
-o \$1	実行可能プログラムを指定します。
\$1.o	プログラム・オブジェクト・ファイルを指定します。
-L\$DB2PATH/lib	DB2 ランタイム共用ライブラリーのロケーションを指定します。たとえば、 \$HOME/sql1lib/lib。 -L オプションを指定しないと、コンパイラーは次のパスを想 定します。 /usr/lib:/lib。
-ldb2	DB2 ライブラリーとリンクします。
-ldb2gmf	Micro Focus COBOL 用 DB2 例外ハンドラー・ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 189 ページの『UNIX Micro Focus COBOL ルーチンの構築』

関連サンプル:

- 『bldrtn -- Builds Solaris Micro Focus COBOL routines (stored procedures)』

第 14 章 Windows

WCHARTYPE CONVERT プリコンパイル・オプション	274
オブジェクトのリンクと埋め込みデータベース	
(OLE DB) 表関数	275
Windows Management Instrumentation (WMI)	275
Microsoft Visual Basic	276
Visual Basic による ADO アプリケーションの構築	276
Visual Basic による疎結合トランザクションの構築	279
Visual Basic 疎結合トランザクション・プロジェクトのトラブルシューティング	281
Visual Basic による RDO アプリケーションの構築	283
Visual Basic でのオブジェクトのリンクと埋め込み (OLE) オートメーション	284
.NET	285
C# .NET アプリケーションの構築	285
C# .NET アプリケーションのバッチ・ファイル	286
C# .NET アプリケーションのコンパイルとリンクのオプション	287
Visual Basic .NET アプリケーションの構築	289
Visual Basic .NET アプリケーションのバッチ・ファイル	290
Visual Basic .NET アプリケーションのコンパイルとリンクのオプション	291
Common Language Runtime (CLR) .NET ルーチンの構築	293
C# .NET ルーチンのバッチ・ファイル	296
Visual Basic .NET ルーチンのバッチ・ファイル	296
CLR .NET ルーチンのコンパイルとリンクのオプション	297
Microsoft Visual C++	298
Visual C++ による ADO アプリケーションの構築	298
Visual C++ でのオブジェクトのリンクと埋め込み (OLE) オートメーション	300
Windows での C/C++ アプリケーションの構築	301
C/C++ アプリケーションのバッチ・ファイル	303

Windows C/C++ アプリケーションのコンパイルとリンクのオプション	304
Windows での C/C++ ルーチンの構築	305
C/C++ ルーチンのバッチ・ファイル	308
Windows C/C++ ルーチンのコンパイルとリンクのオプション	309
Windows での C/C++ 複数接続アプリケーションの構築	310
IBM VisualAge COBOL	313
Windows での IBM COBOL コンパイラーの構成	313
Windows での IBM COBOL アプリケーションの構築	314
IBM COBOL アプリケーションのバッチ・ファイル	316
Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション	317
Windows での IBM COBOL ルーチンの構築	318
IBM COBOL ルーチンのバッチ・ファイル	319
Windows IBM COBOL ルーチンのコンパイルとリンクのオプション	320
Micro Focus COBOL	322
Windows での Micro Focus COBOL コンパイラーの構成	322
Windows での Micro Focus COBOL アプリケーションの構築	323
Micro Focus COBOL アプリケーションのバッチ・ファイル	324
Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション	325
Windows での Micro Focus COBOL ルーチンの構築	325
Micro Focus COBOL ルーチンのバッチ・ファイル	327
Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション	327
オブジェクト REXX	328
Windows でのオブジェクト REXX アプリケーションの構築	328

この章は、Windows オペレーティング・システムでアプリケーションを構築するための詳細な情報を提供します。Windows 環境での DB2 アプリケーション開発の最新の更新事項については、次の Web ページを参照してください。

<http://www.ibm.com/software/data/db2/udb/ad>

WCHARTYPE CONVERT プリコンパイル・オプション

WCHARTYPE プリコンパイル・オプションは、`wchar_t` データ・タイプを使用して、マルチバイト形式またはワイド文字形式のいずれかで GRAPHIC データを処理するかどうかを決定します。

DB2® (Windows® 版) オペレーティング・システムの場合、Microsoft® Visual C++ コンパイラーでコンパイルされたアプリケーションでは WCHARTYPE CONVERT オプションがサポートされています。データベース・コード・ページとは異なるコード・ページのデータを DB2 データベースにアプリケーションが挿入する場合は、このコンパイラーで CONVERT オプションを使用しないでください。DB2 は通常はこのような状況でコード・ページ変換を実行します。しかし、Microsoft C ランタイム環境は、特定の 2 バイト文字の代入文字は処理しません。これは、ランタイム変換エラーとなる場合があります。

WCHARTYPE のデフォルト・オプションは NOCONVERT です。NOCONVERT オプションを使用すると、アプリケーションとデータベース・マネージャーとの間の暗黙の文字変換は起きなくなります。GRAPHIC ホスト変数のデータは、変換されない 2 バイト文字セット (DBCS) 文字として、データベース・マネージャーとの間で送受信されます。

GRAPHIC データをワイド文字形式からマルチバイト形式に変換したい場合は、`wcstombs()` 関数を使用してください。たとえば、

```
wchar_t widechar[200];  
wchar_t mb[200];  
wcstombs((char *)mb,widechar,200);  
  
EXEC SQL INSERT INTO TABLENAME VALUES(:mb);
```

同様に、`mbstowcs()` 関数を使用して、マルチバイト形式をワイド文字形式に変換することができます。

アプリケーションが静的に C ランタイム・ライブラリーにバインドされている場合は、アプリケーションから `setlocale()` 呼び出しを出さないでください。これは、C ランタイム変換エラーとなる可能性があります。アプリケーションが動的に C ランタイム・ライブラリーにバインドされている場合は、`setlocale()` の使用は問題になりません。これは、ルーチン (ストアド・プロシージャとユーザー定義関数) にも当てはまります。

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『C および C++ での `wchar_t` および `sqlbchar` データ・タイプ』
- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『C および C++ での WCHARTYPE プリコンパイル・オプション』
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『C/C++ ルーチンでの GRAPHIC ホスト変数』

関連資料:

- 「コマンド・リファレンス」の『PRECOMPILE コマンド』

オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数

DB2[®] は、OLE DB 表関数をサポートします。これらの関数については、CREATE FUNCTION DDL を作成する他にアプリケーションを構築する必要はありません。

DB2 の `sql1lib\samples\oledb` ディレクトリーに、OLE DB 表関数のサンプル・ファイルが提供されています。これらは、コマンド行プロセッサ (CLP) のファイルです。これらのファイルは、以下のステップで構築できます。

1. `database_name` への `db2` の接続
2. `db2 -t -v -f file_name.db2`
3. `db2` の終了

`database_name` は接続先のデータベース、`file_name` は CLP ファイルの名前 (拡張子は `.db2`) です。

これらのコマンドは、DB2 コマンド・ウィンドウで実行する必要があります。

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『OLE DB 表関数』
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『OLE DB ユーザー定義表関数』

関連資料:

- 100 ページの『オブジェクトのリンクと埋め込みデータベース (OLE DB) 表関数のサンプル』

Windows Management Instrumentation (WMI)

Windows[®] Management Instrumentation (WMI) は、Microsoft[®] の Windows 管理サービスのキー・コンポーネントです。WMI は、アプリケーションおよびシステムの構成、状況、および運用の各側面を詳しく記述した一貫性のあるモデルとして機能します。

DB2[®] WMI プロバイダーを使用すると、DB2 サーバー・サービスのモニター、データベースの列挙と作成、操作可能設定値の構成、データベースのバックアップ、リストア、およびロールフォワード操作の実行を WMI アプリケーションで行うことができます。

DB2 では、`sql1lib\samples\wmi` ディレクトリーに置かれている Visual Basic のスクリプト記述言語用の WMI サンプル・ファイルを利用することができます。このサンプル・プログラムを実行する場合は、以下のコマンドを実行して DB2 WMI プロバイダーが登録済みであることを事前に確認してください。

```
mofcomp %DB2PATH%\bin\db2wmi.mof
regsvr32 %DB2PATH%\bin\db2wmi.dll
```

ここで `%DB2PATH%` は、DB2 のインストール先を示すパスです。

Visual Basic スクリプト・サンプルを実行するには、`cscript` コマンドを使用します。たとえば、`listsvr` サンプル・スクリプトを実行するには、以下を入力します。

```
cscript listsvr.vbs
```

関連概念:

- 「管理ガイド: インプリメンテーション」の『Windows Management Instrumentation (WMI) の紹介』
- 「管理ガイド: インプリメンテーション」の『DB2 Universal Database と Windows Management Instrumentation の統合』

関連資料:

- 112 ページの『Windows Management Instrumentation のサンプル』

Microsoft Visual Basic

Visual Basic による ADO アプリケーションの構築

ActiveX Data Object (ADO) を使用すれば、OLE DB Provider を使用して、データベース・サーバー内のデータにアクセスしたり、操作したりするアプリケーションを書くことができます。ADO の主要な利点は、速度が速く、使用が容易で、メモリーのオーバーヘッドが少なく、ディスク・フットプリントが小さいことです。

Visual Basic ADO サンプル・プログラムは、`sql1lib¥samples¥VB¥ADO` ディレクトリに置かれています。

注: DB2 ADO サンプルを実行するには、以下のバージョン以降のコンポーネントをお勧めします。

1. Visual Basic 6.0 Professional Edition
2. Microsoft データ アクセス 2.7 SDK (選択によっては DB2 バージョン 8 と一緒にインストールします)
3. Visual Basic Service Pack 5
(<http://msdn.microsoft.com/vstudio/sp/vs6sp5/vbfixes.asp> に掲載されています)
4. 最新の Visual Studio Service Pack (<http://msdn.microsoft.com/vstudio/> に掲載されています)

手順:

以下の 2 つの ODBC 対応プロバイダーのどちらでも使用することができます。

- DB2 用の IBM OLE DB Provider
- ODBC 用の Microsoft OLE DB Provider

DB2 用の IBM OLE DB Provider の使用

Windows オペレーティング・システム上の DB2 バージョン 8.2 クライアントは、選択によっては、DB2 用の IBM OLE DB 2.0 準拠のプロバイダーである IBMDADB2 をインストールします。プロバイダーは、DB2 データベース内のデータにアクセスする消費者用のインターフェースを公開します。DB2 用の IBM OLE DB Provider は、以下の ADO アプリケーション・タイプをサポートします。

- Microsoft Active Server Pages (ASP)
- Microsoft Visual Studio C++ および Visual Basic アプリケーション
- Microsoft Visual Interdev

これらのタイプのアプリケーションの詳細については、ADO 資料を参照してください。

DB2 用の IBM OLE DB Provider を使用して DB2 サーバーにアクセスするには、Visual Basic アプリケーションは、以下のように ADO 接続ストリング内に PROVIDER キーワードを指定しなければなりません。

```
Dim c1 As ADODB.Connection
Dim c1str As String
c1str = "Provider=ibmdadb2; DSN=db2alias; UID=userid; PWD=password"
c1.Open c1str
...
```

db2alias は、DB2 データベース・ディレクトリー内でカタログされる DB2 データベースの別名です。

注: DB2 用の IBM OLE DB Provider を使用する場合、データ・ソース用の ODBC カatalog・ステップを実行する必要はありません。このステップが必要なのは、ODBC 用に OLE DB Provider を使用する場合です。

ODBC 用の Microsoft OLE DB Provider の使用

Microsoft OLE DB Provider および Visual Basic で ADO を使用する場合は、ADO タイプ・ライブラリーへの参照を設定する必要があります。以下のようにします。

1. プロジェクト・メニューから「参照 (References)」を選択する。
2. Check the box for "Microsoft ActiveX Data Objects <version_number> Library"
3. 「OK」をクリックします。

<version_number> は、ADO ライブラリーの現行バージョンです。

このようにすると、VBA オブジェクト・ブラウザーと IDE エディターを介して、ADO オブジェクト、メソッド、および特性にアクセスできるようになります。

接続を確立します。

```
Dim db As Connection
Set db = New Connection
```

ローカル・カーソル・ライブラリーによって提供されるクライアント側のカーソルを設定します。

```
db.CursorLocation = adUseClient
```

ADO が Microsoft ODBC Driver を使用するようにプロバイダーを設定します。

ADO でのサンプル・データベースへのアクセス

完全な Visual Basic プログラムには、フォーム、その他のグラフィカルなエレメントが含まれており、このプログラムは Visual Basic 環境の内部で表示する必要があります。上記で説明したように、IBM OLE DB Provider または Microsoft OLE

DB Provider のいずれかを使用してデータベースに接続した後で DB2 sample データベースにアクセスするプログラムの一部をなす Visual Basic コマンドを以下に示します。

ユーザー ID またはパスワードを指定しないで、sample データベースをオープンします。つまり、現行ユーザーを使用します。

```
db.Open "SAMPLE"
```

レコード・セットを作成します。

```
Set adoPrimaryRS = New Recordset
```

SELECT ステートメントを使用して、レコード・セットにデータを入れます。

```
adoPrimaryRS.Open "select EMPNO, LASTNAME, FIRSTNAME, MIDINIT, EDLEVEL, JOB  
from EMPLOYEE Order by EMPNO", db
```

この時点で、プログラマーは ADO メソッドを使用して、次のレコード・セットに移動する場合など、データにアクセスできるようになります。

```
adoPrimaryRS.MoveNext
```

レコード・セットの現行レコードを削除します。

```
adoPrimaryRS.Delete
```

またプログラマーは、以下のようにして、個々のフィールドにアクセスすることもできます。

```
Dim Text1 as String  
Text1 = adoPrimaryRS!LASTNAME
```

関連概念:

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider for DB2 の目的』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider for DB2 でサポートされているアプリケーション・タイプ』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『Visual Basic ADO アプリケーションによるデータ・ソースへの接続』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider で自動的に使用可能になる OLE DB サービス』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider によるラージ・オブジェクトの操作』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『MTS および COM+ 分散トランザクションのサポートと IBM OLE DB Provider』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider の制限』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『ADO (ActiveX Data Object) および RDO (Remote Data Object)』

関連資料:

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での OLE DB コンポーネントおよびインターフェースのサポート』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での OLE DB プロパティのサポート』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での ADO メソッドおよびプロパティのサポート』
- ・ 108 ページの『Visual Basic のサンプル』

Visual Basic による疎結合トランザクションの構築

XA では、制御のアプリケーション・スレッドが単一の XA グローバル・トランザクションに参加する方法として、密結合および疎結合の 2 つの方法があります。サンプル・プロジェクトの LCTransTest は、XA の疎結合トランザクションを示しています。サンプルの応答ファイルは、`sql1lib\samples\VB\MTS` ディレクトリーにあります。

手順:

疎結合トランザクション・サンプルを構築および実行するには、以下のステップに従ってください。

1. LCTransTest.vbp プロジェクトの構築

- "LCTransTest.vbp" プロジェクトをダブルクリックして開きます。
- エラー・メッセージ "Unable to set the version compatible component X:\...\LCTransTest.dll" を受け取った場合、「OK」をクリックして続行します。
- プロジェクトをコンパイルします。「ファイル」->「LCTransTest.dll の作成 (Make LCTransTest.dll)」と移動して、「OK」をクリックします。
- バージョンの非互換性の問題を修正するには、右上パネルにある「LCTransTest (LCTransTest.vbp)」プロジェクトを右クリックします。その後、「LCTransTest のプロパティ (LCTransTest Properties)」をクリックします。「LCTransTest - プロジェクトのプロパティ (LCTransTest - Project Properties)」ウィンドウで、「コンポーネント(Component)」タブをクリックします。「バージョン互換性 (Version Compatibility)」セクションの下で、「バイナリー互換性 (Binary Compatibility)」を選択します。
- プロジェクトを保管します。「ファイル」->「プロジェクトの保管 (Save Project)」と移動します。
- プロジェクトをクローズします。

2. Main.vbp プロジェクトの構築

- "Main.vbp" プロジェクトをダブルクリックして開きます。
- 場合によっては、警告メッセージ "Could not create reference: X:\...\LCTransTest.dll" を受け取ります。「OK」をクリックして続行します。
- (ツールバー上で)「プロジェクト (Project)」->「参照 (References)」と移動します。

- d. 「参照 - main.vbp (References - main.vbp)」ウィンドウで、「Microsoft ActiveX Data Objects 2.7 Library」ボックスがチェックされていることを確認します。「ブラウズ...」に移動します。ステップ 1 で生成した LCTransTest.dll を見付けて「開く (Open)」をクリックすることにより、この参照を追加します。「参照 - main.vbp (References - main.vbp)」ウィンドウで、「OK」をクリックします。
 - e. プロジェクトをコンパイルします。「ファイル」->「main.exe の作成 (Make main.exe)」と移動して、「OK」をクリックします。
 - f. プロジェクトを保管します。「ファイル」->「プロジェクトの保管 (Save Project)」と移動します。
 - g. プロジェクトをクローズします。
3. その他の設定
- a. Windows で、「スタート」->「設定」->「コントロール パネル」->「管理ツール」->「コンポーネント サービス」と移動します。
 - b. 「コンポーネント サービス」ウィンドウで、左パネルの「コンポーネント サービス」を展開して「COM+ アプリケーション」が表示されるようにします。
 - c. 「COM+ アプリケーション」を右クリックして、「新規」->「アプリケーション」と選択します。
 - d. 「COM アプリケーション インストール ウィザードへようこそ」ポップアップ・ウィンドウで、「次へ」をクリックします。
 - e. 「空のアプリケーションの作成」を選択します。
 - f. 新しいアプリケーションの名前を、「LCTransTest」と入力します。「アクティブ化の種類」は「サーバー アプリケーション」のままにします。「次へ」をクリックします。
 - g. 「次へ」をクリックしてから、「完了」をクリックします。
 - h. 「LCTransTest」を展開して、「コンポーネント」を右クリックします。「新規」->「コンポーネント」と移動し、「既に登録されているコンポーネントをインポートする」を選択してから、「LCTransTest.TestClass」->「次へ」->「完了」とクリックします。
 - i. 「コンポーネント」を展開します。「LCTransTest.TestClass」を右クリックします。「プロパティ」に移動します。「トランザクション」タブの下にある「トランザクション サポート」で、「必要」にチェックマークを付けます。
 - j. 「コンポーネント サービス」を右クリックしてから「コンピュータ」->「マイコンピュータ」と選択して、Microsoft Distributed Transaction Coordinator を再始動します。「MS DTC を停止する (Stop MS DTC)」を選択します。それが停止するまで待ってから、「マイコンピュータ」を右クリックして、「MS DTC を開始する (Start MS DTC)」を選択し、DTC を再始動します。
4. サンプルをデバッグ・モードで実行する方法
- a. LCTransTest.vbp を開きます。
 - b. LCTransTest で、「デバッグ (Debugging)」タブの下の「プロジェクトのプロパティ* (project properties*)」で「コンポーネントが作成されるまで待つ (Wait for components to be created)」がチェックされていることを確認します。(デフォルトでチェックされているはずです。)

- c. "con1.Open connString" の行にブレーク点を設定します (その行にカーソルを置いて F9 を押します)。
- d. F5 を押します (または「実行 (Run)」プルダウン・メニューの「開始」を選択します)。この dll がロードされてこのメソッドが実行されるとき、デバッガーはブレーク点で実行を停止します。
- e. main.vbp を開きます。
- f. main.vbp で、コマンド行引き数をセットアップします。プロジェクトのプロパティーで、「作成 (Make)」タブの下の「コマンド行引き数 (Command Line arguments)」テキスト・ボックスに、以下のとおり入力します。

```
provider=ibmdadb2;dsn=<dbname>;uid=<userid>;pwd=<password> <filename>
```

<dbname> は使用するデータベース、<filename> は出力情報を含むことになるファイルの名前 (パスを含む) です。たとえば、C:\lctoutput.txt となります。次に、「OK」をクリックします。
- g. Visual Basic で「デバッグ (Debug)」->「ステップイントウ (Step Into)」の <F8> または「デバッグ (Debug)」->「ステップオーバー (Step Over)」の <shift + F8> を使用して、実行可能プログラムのコードを 1 行ずつ実行することができます。
- h. main 実行可能プログラムで "transTest.RunTest" と呼ばれる行に到達して、それをステップオーバーしようとする場合、他の Visual Basic ウィンドウ (開いている LCTransTest プロジェクト) が前面に表示されて、そこに設定したブレークポイントで停止します。その後、「ステップイントウ (Step Into)」または「ステップオーバー (Step Over)」を使用して、RunTest メソッドをコードの 1 行ごとに実行して進むことができます。

関連タスク:

- 276 ページの『Visual Basic による ADO アプリケーションの構築』
- 281 ページの『Visual Basic 疎結合トランザクション・プロジェクトのトラブルシューティング』

関連資料:

- 108 ページの『Visual Basic のサンプル』

Visual Basic 疎結合トランザクション・プロジェクトのトラブルシューティング

Visual Basic 統合環境の特徴の 1 つとして、データ・リンク・ライブラリー (dll) をコンパイルするたびに、そのための新規のグローバル・ユニーク ID (GUID) が作成され、Windows レジストリーに登録されます。プロジェクト dll が構築されると、それは GUID を使用して、分散トランザクション・コーディネーター (DTC) に登録されます。プロジェクト dll が再構築されると、Visual Basic はそれに新規の GUID を割り当てると共に、Windows レジストリーに登録します。そのため、DTC がプロジェクト dll を参照するために使用している GUID は古くなり、Windows レジストリー内にそのプロジェクトに対する 2 つの異なる項目が存在することになります。

手順:

この問題を回避するため、最初にプロジェクトを作成した後に以下のようにプロジェクト・プロパティーを変更してください。

1. 「コンポーネント (Component)」タブの下で「バイナリー互換性 (Binary Compatibility)」を選択してから、「...」ボタンを使用して作成する dll の完全パスを検索します。
2. その後「OK」ボタンをクリックして、プロジェクトを即時に保管します。

これで、プロジェクト dll を再コンパイルするたびに、同じ GUID が保たれます。しかし、(メソッドの追加または削除、または既存のメソッドのパラメーターの変更などにより) dll へのインターフェースを変更した場合、新規の GUID を使用しなければならなくなります。

Windows レジストリー内にすでに複数の GUID 項目が存在する場合、以下のようになしてください。

1. レジストリー・エディターを使用してプロジェクト名を検索します。
2. Visual Basic の最新のプロジェクト・リストにあるもの以外の、プロジェクト名のすべてのオカレンスを除去します。

疎結合トランザクションが生じるためには、ユーザー ID およびパスワードを含む接続情報が DTC と Visual Basic アプリケーションとで同じでなければなりません。通常、dll のパブリック・メソッドを使用してそれに直接アクセスします。しかし DTC が関係している場合、それは dll をカプセル化して、メソッドへの入力呼び出しおよびそれらのメソッドから発信される結果をすべて代行受信します。このようにして、DTC はそれらのいずれかのメソッドでのデータベース・アクティビティーを同じオブジェクトに対する他のデータベース・アクティビティーまたは異なるオブジェクトに対するデータベース・アクティビティーといつ疎結合にする必要があるかを知ることができます。

これによって生じる可能性のある問題を回避するには、以下を行います。

1. DTC の「ID」タブの下にある「アプリケーション (Application)」プロパティーで、「ユーザー: (This user:)」を選択します。
2. 「ブラウズ (Browse)」ボタンを使用して、このプロジェクトを実行するユーザーの ID を検索します。
3. プロジェクト接続ストリング内のユーザー ID およびパスワードを使用します。

Visual Basic アプリケーション接続ストリングでコンピューターにログオンする際に使用するものと同じユーザー ID およびパスワードを使用する場合、この追加ステップは必要ありません。

疎結合トランザクション・プロジェクトが正常に機能していることを確かめるには、以下を行います。

1. 実行可能プログラムが作成する出力ファイルを見て、データベースの更新が生じていることを確認します。
2. cli トレースで ENLIST_IN_DTC があるかどうかを調べます。

これらのテストがどちらも失敗した場合、dll は DTC に正常に登録されないで、疎結合トランザクションは生じていません。

関連タスク:

- 276 ページの『Visual Basic による ADO アプリケーションの構築』

- 279 ページの『Visual Basic による疎結合トランザクションの構築』

関連資料:

- 108 ページの『Visual Basic のサンプル』

Visual Basic による RDO アプリケーションの構築

Remote Data Objects (RDO) は、ODBC を介してリモート・データ・ソースにアクセスするための情報モデルを提供します。RDO が提供するオブジェクトのセットを使用すれば、データベースへの接続、照会の実行、ストアード・プロシージャの実行、結果の操作、変更のサーバーへのコミットが容易になります。RDO は、リモート ODBC リレーショナル・データ・ソースにアクセスするために特別に設計されたものであり、複雑なアプリケーション・コードを使用せずに ODBC を使用することが容易になっています。そのため、ODBC ドライバーによって公開されるリレーショナル・データベースにアクセスするための主な手段となっています。RDO はオープン・データベース・コネクティビティー (ODBC) API を介したシン・コード層をインプリメントしています。また、接続の確立、結果セットとカーソルの作成を行い、ワークステーション・リソースを最小限に抑えて複雑なプロシージャを実行するドライバー・マネージャーもインプリメントしています。

DB2 の `sql1lib¥samples¥VB` に、Visual Basic RDO のサンプル・プログラムがあります。

手順:

RDO を Microsoft Visual Basic で使用する場合は、Visual Basic のプロジェクトへの参照を設定する必要があります。以下のようにします。

1. プロジェクト・メニューから「参照 (References)」を選択する。
2. 「Microsoft Remote Data Object <Version Number>」のボックスにチェックマークを付ける。
3. 「OK」をクリックします。

<version_number> は、RDO の現行バージョンです。

完全な Visual Basic プログラムには、フォーム、その他のグラフィカルなエレメントが含まれており、このプログラムは Visual Basic 環境の内部で表示する必要があります。以下に、DB2 プログラムの一部を成す Visual Basic コマンドを示します。DB2 プログラムは、sample データベースに接続して、EMPLOYEE 表のすべての列を選択するレコード・セットをオープンし、メッセージ・ウィンドウに従業員名を一人一人表示します。

```
Dim rdoEn As rdoEngine
Dim rdoEv As rdoEnvironment
Dim rdoCn As rdoConnection
Dim Cnct$
Dim rdoRS As rdoResultset
Dim SQLQueryDB As String
```

接続ストリングを割り当てます。

```
Cnct$ = "DSN=SAMPLE;UID=;PWD=;"
```

RDO 環境を設定します。


```
Set rdoEn = rdoEngine
Set rdoEv = rdoEn.rdoEnvironments(0)
```

データベースに接続します。

```
Set rdoCn = rdoEv.OpenConnection("", , , Cnct$)
```

レコード・セットの SELECT ステートメントを割り当てます。

```
SQLQueryDB = "SELECT * FROM EMPLOYEE"
```

レコード・セットをオープンして、照会を実行します。

```
Set rdoRS = rdoCn.OpenResultset(SQLQueryDB)
```

レコード・セットの終わりに While not を置き、メッセージ・ボックスに、表にあるラストネームとファーストネーム (1 回につき 1 人の従業員) を表示します。

```
While Not rdoRS.EOF
MsgBox rdoRS!LASTNAME & ", " & rdoRS!FIRSTNAME
```

レコード・セット内の次の行に移動します。

```
rdoRS.MoveNext
Wend
```

プログラムをクローズします。

```
rdoRS.Close
rdoCn.Close
rdoEv.Close
```

関連概念:

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『ADO (ActiveX Data Object) および RDO (Remote Data Object)』

関連資料:

- ・ 108 ページの『Visual Basic のサンプル』

Visual Basic でのオブジェクトのリンクと埋め込み (OLE) オートメーション

OLE では言語は限定されないもので、任意の言語で OLE オートメーション UDF およびストアード・プロシージャをインプリメントすることができます。そのためには、OLE オートメーション・サーバーのメソッドを公開し、そのメソッドを UDF として DB2[®] に登録します。OLE オートメーション・サーバーの開発をサポートするアプリケーション開発環境には、以下の特定のバージョンが含まれます。Microsoft[®] Visual Basic、Microsoft Visual C++、Microsoft Visual J++、Microsoft FoxPro、Borland Delphi、Powersoft PowerBuilder、および Micro Focus COBOL。さらに、たとえば Microsoft Visual J++ のもののような、OLE 用に正しくラップされた Java[™] Bean オブジェクトにも、OLE オートメーションを介してアクセスすることができます。

OLE オートメーション・サーバーの開発の詳細については、該当するアプリケーション開発環境の資料を参照する必要があります。

OLE オートメーション UDF およびストアード・プロシージャ

Microsoft Visual Basic は、OLE オートメーション・サーバーの作成をサポートします。新しい種類のオブジェクトは、Visual Basic ではクラス・モジュールを Visual Basic プロジェクトに追加することによって作成します。メソッドは、パブリック・サブプロシージャをクラス・モジュールに追加することによって作成します。これらのパブリック・プロシージャは、DB2 に OLE オートメーション UDF およびストアード・プロシージャとして登録できます。OLE サーバーの作成および構築の詳細については、Microsoft Visual Basic のマニュアル、「*Creating OLE Servers, Microsoft Corporation, 1995*」、および Microsoft Visual Basic に付属している OLE サンプルを参照してください。

DB2 は、Microsoft Visual Basic 内の OLE オートメーション UDF およびストアード・プロシージャのサンプルを提供しており、それはディレクトリー `sqllib\samples\ole\msvb` にあります。OLE オートメーション UDF およびストアード・プロシージャのサンプルを構築して実行する方法については、`sqllib\samples\ole` の README ファイルを参照してください。

関連概念:

- ・「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『OLE オートメーション・ルーチンの設計』
- ・「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『BASIC および C++ での OLE オートメーション・ルーチン』

関連資料:

- ・ 99 ページの『オブジェクトのリンクと埋め込み (OLE) のサンプル』

.NET

C# .NET アプリケーションの構築

DB2 には、DB2 C# .NET アプリケーションのコンパイルとリンクのためのバッチ・ファイル `bldapp.bat` が用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib\samples\%.NET\cs` ディレクトリーに置かれています。このバッチ・ファイルは 1 つのパラメーター `%1` をとります。このパラメーターは、コンパイルするソース・ファイルの名前を (`.cs` 拡張を付けずに) 指定します。

手順:

ソース・ファイル `DbAuth.cs` からプログラム `DbAuth` を構築するには、次のように入力します。

```
bldapp DbAuth
```

実行可能ファイルを実行する際に必要なパラメーターを確実に指定するようにするには、以下のように入力する情報の数に応じてさまざまなパラメーターの組み合わせを指定できます。

1. パラメーターなし。次のように、プログラム名のみ入力します。

```
DbAuth
```


2. 1 つのパラメーター。次のように、プログラム名に加えてデータベース別名を入力します。

```
DbAuth <db_alias>
```

3. 2 つのパラメーター。次のように、プログラム名に加えてユーザー ID とパスワードを入力します。

```
DbAuth <userid> <passwd>
```

4. 3 つのパラメーター。次のように、プログラム名に加えてデータベース別名、ユーザー ID、およびパスワードを入力します。

```
DbAuth <db_alias> <userid> <passwd>
```

5. 4 つのパラメーター。次のように、プログラム名に加えてサーバー名、ポート番号、ユーザー ID、およびパスワードを入力します。

```
DbAuth <server> <portnum> <userid> <passwd>
```

6. 5 つのパラメーター。次のように、プログラム名に加えてデータベース別名、サーバー名、ポート番号、ユーザー ID、およびパスワードを入力します。

```
DbAuth <db_alias> <server> <portnum> <userid> <passwd>
```

LCTrans サンプル・プログラムを構築して実行するには、ソース・ファイル LCTrans.cs に示されている詳細な指示に従う必要があります。

関連タスク:

- 293 ページの『Common Language Runtime (CLR) .NET ルーチンの構築』

関連資料:

- 82 ページの『C# のサンプル』
- 287 ページの『C# .NET アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp.bat -- Builds C# applications on Windows』
- 『DbAuth.cs -- How to Grant, display and revoke privileges on database』
- 『LCTrans.cs -- Demonstrates loosely coupled transactions (CSNET)』

C# .NET アプリケーションのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldapp.bat
rem Builds C# applications on Windows
rem Usage: bldapp prog_name

rem Default compiler is set to Microsoft C# Compiler
rem To use a different compiler, comment out 'set BLDCOMP=csc'
rem and write 'set BLDCOMP=x' where x is the the required compiler
set BLDCOMP=csc

rem When using the .NET Framework Version 1.0 point to netf10
rem set VERSION=netf10¥
rem When using the .NET Framework Version 1.1 point to netf11
set VERSION=netf11¥

if exist %1.cs goto build

:build
if "%1"=="LCTrans" goto LCTransbuild
if "%1"=="SubCOM" goto SubCOMbuild
```

```

if "%1"=="RootCOM" goto RootCOMbuild
%BLDCOMP% %1.cs /r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll
goto exit

:RootCOMbuild
%BLDCOMP% /out:RootCOM.dll /target:library %1.cs /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /r:System.Data.dll /r:System.dll
/r:SubCOM.dll
goto exit

:SubCOMbuild
%BLDCOMP% /out:SubCOM.dll /target:library %1.cs /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /r:System.Data.dll /r:System.Xml.dll
/r:System.dll
goto exit

:LCTransbuild
%BLDCOMP% %1.cs /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /r:System.Data.dll /r:System.dll
/r:SubCOM.dll /r:RootCOM.dll
goto exit

:exit
@echo on

```

C# .NET アプリケーションのコンパイルとリンクのオプション

以下は、bldapp.bat バッチ・ファイルに示されているように、Windows 上で Microsoft C# コンパイラを使用して、C# アプリケーションを構築するのに勧めるコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション
スタンドアロン C# アプリケーションのコンパイルおよびリンクのオプション
%BLDCOMP% コンパイラ用の変数。デフォルトは csc (Microsoft C# コンパイラ) です。
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll 使用している .NET Framework のバージョン用の DB2 データ・リンク・ライブラリーを参照します。
%VERSION% アプリケーション用にサポートされている .NET Framework のバージョンは 2 つあります。それぞれのデータ・リンク・ライブラリーは、別々のサブディレクトリーにあります。 .NET Framework 1.0 の場合、%VERSION% は netf10¥ サブディレクトリーを示し、.NET Framework 1.1 の場合、%VERSION% は netf11¥ サブディレクトリーを示します。

bldapp のコンパイルとリンクのオプション	
疎結合サンプル・プログラム LCTrans のコンパイルおよびリンク・オプション:	
%BLDCOMP%	コンパイラー用の変数。デフォルトは csc (Microsoft C# コンパイラー) です。
/out:RootCOM.d11	LCTrans アプリケーションが使用する RootCOM データ・リンク・ライブラリーを、RootCOM.cs ソース・ファイルから出力します。
/out:SubCOM.d11	LCTrans アプリケーションが使用する SubCOM データ・リンク・ライブラリーを、SubCOM.cs ソース・ファイルから出力します。
/target:library %1.cs	入力ソース・ファイル (RootCOM.cs または SubCOM.cs) からデータ・リンク・ライブラリーを作成します。
/r:System.EnterpriseServices.d11	Microsoft Windows の System EnterpriseServices データ・リンク・ライブラリーを参照します。
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.d11	使用している .NET Framework のバージョン用の DB2 データ・リンク・ライブラリーを参照します。
%VERSION%	アプリケーション用にサポートされている .NET Framework のバージョンは 2 つあります。それぞれのデータ・リンク・ライブラリーは、別々のサブディレクトリーにあります。 .NET Framework 1.0 の場合、%VERSION% は netf10 サブディレクトリーを示し、 .NET Framework 1.1 の場合、%VERSION% は netf11 サブディレクトリーを示します。
/r:System.Data.d11	Microsoft Windows の System Data データ・リンク・ライブラリーを参照します。
/r:System.d11	Microsoft Windows の System データ・リンク・ライブラリーを参照します。
/r:System.Xml.d11	Microsoft Windows の System XML データ・リンク・ライブラリーを参照します (SubCOM.cs 用)。
/r:SubCOM.d11	SubCOM データ・リンク・ライブラリーを参照します (RootCOM.cs および LCTrans.cs 用)。
/r:RootCOM.d11	RootCOM データ・リンク・ライブラリーを参照します (LCTrans.cs 用)。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 285 ページの『C# .NET アプリケーションの構築』

関連サンプル:

- 『bldapp.bat -- Builds C# applications on Windows』

Visual Basic .NET アプリケーションの構築

DB2 には、DB2 Visual Basic .NET アプリケーションのコンパイルとリンクのためのバッチ・ファイル bldapp.bat が用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib¥samples¥.NET¥vb` ディレクトリーに置かれています。このバッチ・ファイルは 1 つのパラメーター %1 をとります。このパラメーターは、コンパイルするソース・ファイルの名前を (.vb 拡張を付けずに) 指定します。

手順:

ソース・ファイル DbAuth.vb からプログラム DbAuth を構築するには、次のように入力します。

```
bldapp DbAuth
```

実行可能ファイルを実行する際に必要なパラメーターを確実に指定するようにするには、以下のように入力する情報の数に応じてさまざまなパラメーターの組み合わせを指定できます。

1. パラメーターなし。次のように、プログラム名のみ入力します。

```
DbAuth
```

2. 1 つのパラメーター。次のように、プログラム名に加えてデータベース別名を入力します。

```
DbAuth <db_alias>
```

3. 2 つのパラメーター。次のように、プログラム名に加えてユーザー ID とパスワードを入力します。

```
DbAuth <userid> <passwd>
```

4. 3 つのパラメーター。次のように、プログラム名に加えてデータベース別名、ユーザー ID、およびパスワードを入力します。

```
DbAuth <db_alias> <userid> <passwd>
```

5. 4 つのパラメーター。次のように、プログラム名に加えてサーバー名、ポート番号、ユーザー ID、およびパスワードを入力します。

```
DbAuth <server> <portnum> <userid> <passwd>
```

6. 5 つのパラメーター。次のように、プログラム名に加えてデータベース別名、サーバー名、ポート番号、ユーザー ID、およびパスワードを入力します。

```
DbAuth <db_alias> <server> <portnum> <userid> <passwd>
```

LCTrans サンプル・プログラムを構築して実行するには、ソース・ファイル LCTrans.vb に示されている詳細な指示に従う必要があります。

関連タスク:

- 293 ページの『Common Language Runtime (CLR) .NET ルーチンの構築』

関連資料:

- 109 ページの『Visual Basic .NET のサンプル』

- 291 ページの『Visual Basic .NET アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp.bat -- Builds Visual Basic .Net applications on Windows』
- 『DbAuth.vb -- How to Grant, display and revoke privileges on database』
- 『LCTrans.vb -- Demonstrates loosely coupled transactions』

Visual Basic .NET アプリケーションのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldapp.bat
rem Builds Visual Basic .Net applications on Windows
rem Usage: bldapp prog_name [ db_name [ userid password ]]

rem Default compiler is set to Microsoft Visual Basic .NET Compiler
rem To use a different compiler, comment out 'set BLDCOMP=vbc'
rem and write 'set BLDCOMP=x' where x is the the required compiler
set BLDCOMP=vbc

rem When using the .NET Framework Version 1.0 point to netf10
rem set VERSION=netf10¥
rem When using the .NET Framework Version 1.1 point to netf11
set VERSION=netf11¥

if exist %1.vb goto build

:build
if "%1"=="LCTrans" goto LCTransbuild
if "%1"=="SubCOM" goto SubCOMbuild
if "%1"=="RootCOM" goto RootCOMbuild
%BLDCOMP% %1.vb /r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /r:System.dll
/r:System.Data.dll /r:System.Xml.dll
goto exit

:RootCOMbuild
%BLDCOMP% %1.vb /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /target:library /r:System.Data.dll
/r:System.dll /r:SubCOM.dll /out:RootCOM.dll
goto exit

:SubCOMbuild
%BLDCOMP% %1.vb /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /target:library /r:System.Data.dll
/r:System.Xml.dll /r:System.dll /out:SubCOM.dll
goto exit

:LCTransbuild
%BLDCOMP% %1.vb /r:System.EnterpriseServices.dll
/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll /r:System.Data.dll /r:System.dll
/r:SubCOM.dll /r:RootCOM.dll
goto exit

:exit
@echo on
```

Visual Basic .NET アプリケーションのコンパイルとリンクのオプション

以下は、bldapp.bat バッチ・ファイルに示されているように、Windows 上で Microsoft Visual Basic .NET コンパイラーを使用して、Visual Basic .NET アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
スタンドアロン VB .NET アプリケーションのコンパイルおよびリンクのオプション	
%BLDCOMP%	コンパイラー用の変数。デフォルトは vbc (Microsoft Visual Basic .NET コンパイラー) です。
/r:"%DB2PATH%\bin¥%VERSION%IBM.Data.DB2.d11	使用している .NET Framework のバージョン用の DB2 データ・リンク・ライブラリーを参照します。
%VERSION%	アプリケーション用にサポートされている .NET Framework のバージョンは 2 つあります。それぞれのデータ・リンク・ライブラリーは、別々のサブディレクトリーにあります。 .NET Framework 1.0 の場合、 %VERSION% は netf10¥ サブディレクトリーを示し、 .NET Framework 1.1 の場合、 %VERSION% は netf11¥ サブディレクトリーを示します。

bldapp のコンパイルとリンクのオプション

疎結合サンプル・プログラム LCTrans のコンパイルおよびリンク・オプション:

%BLDCOMP%

コンパイラー用の変数。デフォルトは vbc (Microsoft Visual Basic .NET コンパイラー) です。

/out:RootCOM.dll

LCTrans アプリケーションが使用する RootCOM データ・リンク・ライブラリーを、RootCOM.vb ソース・ファイルから出力します。

/out:SubCOM.dll

LCTrans アプリケーションが使用する SubCOM データ・リンク・ライブラリーを、SubCOM.vb ソース・ファイルから出力します。

/target:library %1.cs

入力ソース・ファイル (RootCOM.vb または SubCOM.vb) からデータ・リンク・ライブラリーを作成します。

/r:System.EnterpriseServices.dll

Microsoft Windows の System EnterpriseServices データ・リンク・ライブラリーを参照します。

/r:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll

使用している .NET Framework のバージョン用の DB2 データ・リンク・ライブラリーを参照します。

%VERSION%

アプリケーション用にサポートされている .NET Framework のバージョンは 2 つあります。それぞれのデータ・リンク・ライブラリーは、別々のサブディレクトリーにあります。 .NET Framework 1.0 の場合、%VERSION% は netf10 サブディレクトリーを示し、 .NET Framework 1.1 の場合、%VERSION% は netf11 サブディレクトリーを示します。

/r:System.Data.dll

Microsoft Windows の System Data データ・リンク・ライブラリーを参照します。

/r:System.dll

Microsoft Windows の System データ・リンク・ライブラリーを参照します。

/r:System.Xml.dll

Microsoft Windows の System XML データ・リンク・ライブラリーを参照します (SubCOM.vb 用)。

/r:SubCOM.dll

SubCOM データ・リンク・ライブラリーを参照します (RootCOM.vb および LCTrans.vb 用)。

/r:RootCOM.dll

RootCOM データ・リンク・ライブラリーを参照します (LCTrans.vb 用)。

他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。

関連タスク:

- 289 ページの『Visual Basic .NET アプリケーションの構築』

関連サンプル:

- 『bldapp.bat -- Builds Visual Basic .Net applications on Windows』

Common Language Runtime (CLR) .NET ルーチンの構築

DB2 には、DB2 .NET プログラムのコンパイルとリンクのためのバッチ・ファイルが用意されていて、バッチ・ファイルは、このファイルを使用して作成できるサンプル・プログラムと一緒に `sqllib\samples\%.NET%cs` および `sqllib\samples\%.NET%vb` ディレクトリーに置かれています。

バッチ・ファイル `bldrtn.bat` には、CLR ルーチン (ストアド・プロシージャとユーザー定義関数) を作成するためのコマンドが入っています。バッチ・ファイルは、サーバー上に .NET アセンブリー DLL を作成します。これは 2 個のパラメーターをとります。それらは、バッチ・ファイル内では変数 `%1` と `%2` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。このバッチ・ファイルは、アセンブリー DLL 名としてソース・ファイル名を使用します。第 2 パラメーター `%2` には、接続先のデータベースの名前を指定します。アセンブリー DLL は、データベースが置かれているのと同じインスタンス上で作成する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

第 1 パラメーター (ソース・ファイル名) だけが必須です。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。

前提条件:

データベース・サーバーでは、Windows オペレーティング・システムが実行され、Microsoft .NET Framework 1.1 がインストールされている必要があります (クライアント・アプリケーションには .NET Framework 1.0 と .NET Framework 1.1 の両方がサポートされています)。 .NET Framework は単独で、または Microsoft .NET Framework 1.1 Software Development Kit の一部として使用できます。

以下のバージョンの DB2 がインストールされていることが必要です。

サーバー:

DB2 8.2 以降

クライアント:

DB2 7.2 以降

ルーチンに対して CREATE ステートメントを実行する権限が付与されている必要があります。 CREATE ステートメントを実行するために必要な特権については、CREATE ステートメントのルーチン・タイプ CREATE PROCEDURE または CREATE FUNCTION を参照してください。

手順:

以下の例は、ストアド・プロシージャとユーザー定義関数を使用してルーチンのアセンブリー DLL を作成する方法を示しています。

ストアード・プロシージャのアセンブリー DLL

VB .NET ソース・ファイル SpServer.vb または C# ソース・ファイル SpServer.cs から SpServer アセンブリー DLL を作成するには、次のようにします。

1. 次のように、バッチ・ファイル名とプログラム名を入力します (拡張子なし)。

```
bldrtn SpServer
```

デフォルトの sample データベース以外のデータベースに接続しているときは、データベース名も入力します。

```
bldrtn SpServer database
```

このバッチ・ファイルは、アセンブリー DLL の SpServer.dll を sqllib¥function ディレクトリーにコピーします。

2. 次に、サーバーで spcat スクリプトを実行してルーチンをカタログします。

```
SpCat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば SpDrop.db2 を呼び出してルーチンをアンカタログし、次に SpCreate.db2 を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、SpDrop.db2 スクリプトと SpCreate.db2 スクリプトは、個別に実行することもできます。

3. アセンブリー DLL の作成が今回初めてでなければ、データベースを一度停止してから再始動し、新しい DLL が認識されるようにします。必要であれば、アセンブリー DLL にファイル・モードを設定して、DB2 インスタンスからアクセスできるようにします。

アセンブリー DLL SpServer を作成し終わったら、それを呼び出すクライアント・アプリケーション SpClient を作成することができます。

SpClient は、バッチ・ファイル bldapp.bat を使用して構築することができます。

実行可能ファイルを実行する際に必要なパラメーターが確実に指定されるようにするために、デフォルトの設定のままにしておくのではなく、以下のように入力されるパラメーターの数に応じたパラメーターのさまざまな組み合わせを指定することができます。

1. パラメーターなし。プログラム名をそのまま入力します (サーバー・インスタンスでローカルに呼び出す場合)。

```
SpClient
```

2. 1 つのパラメーター。次のように、プログラム名に加えてデータベース別名を入力します (サーバー・インスタンスで sample データベースではない他のデータベースをローカルに呼び出す場合)。

```
SpClient <db_alias>
```

3. 3 つのパラメーター。次のように、プログラム名に加えてデータベース別名、ユーザー ID、およびパスワードを入力します (リモート・クライアントから呼び出す場合)。

```
SpClient <db_alias> <userid> <passwd>
```

4. 5 つのパラメーター。次のように、プログラム名に加えてデータベース別名、サーバー名、ポート番号、ユーザー ID、およびパスワードを入力します (リモート・クライアントから呼び出す場合)。

```
SpClient <db_alias> <server> <portnum> <userid> <passwd>
```

クライアント・アプリケーションは、アセンブリー DLL SpServer にアクセスして、サーバー・データベース上のいくつかのルーチンを実行します。出力は、クライアント・アプリケーションに戻されます。

ユーザー定義関数のアセンブリー DLL

VB .NET ソース・ファイル UDFsrv.vb または C# ソース・ファイル UDFsrv.cs からユーザー定義関数のアセンブリー DLL UDFsrv を作成するには、次のようにします。

```
bldrtn UDFsrv
```

デフォルトの `sample` データベース以外のデータベースに接続しているときは、データベース名も入力します。

```
bldrtn UDFsrv database
```

このバッチ・ファイルは、ユーザー定義関数のアセンブリー DLL の UDFsrv.dll を `sqlllib\function` ディレクトリーにコピーします。

udfsrv の構築が完了したら、それを呼び出すクライアント・アプリケーション udfcli を構築できます。

UDFcli は、バッチ・ファイル bldapp.bat を使用して構築することができます。

実行可能ファイルを実行する際に必要なパラメーターが確実に指定されるようにするために、デフォルトの設定のままにしておくのではなく、以下のように入力されるパラメーターの数に応じたパラメーターのさまざまな組み合わせを指定することができます。

1. パラメーターなし。プログラム名をそのまま入力します (サーバー・インスタンスでローカルに呼び出す場合)。

```
UDFcli
```

2. 1 つのパラメーター。次のように、プログラム名に加えてデータベース別名を入力します (サーバー・インスタンスで `sample` データベースではない他のデータベースをローカルに呼び出す場合)。

```
UDFcli <db_alias>
```

3. 3 つのパラメーター。次のように、プログラム名に加えてデータベース別名、ユーザー ID、およびパスワードを入力します (リモート・クライアントから呼び出す場合)。

```
UDFcli <db_alias> <userid> <passwd>
```

4. 5 つのパラメーター。次のように、プログラム名に加えてデータベース別名、サーバー名、ポート番号、ユーザー ID、およびパスワードを入力します (リモート・クライアントから呼び出す場合)。

```
UDFcli <db_alias> <server> <portnum> <userid> <passwd>
```

この呼び出しアプリケーションは、 udfsrv アセンブリー DLL から ScalarUDF 関数を呼び出します。

関連概念:

- ・ 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『共通言語ランタイム (CLR) ルーチン』

関連サンプル:

- ・ 『bldrtn.bat -- Builds C# routines (stored procedures and UDFs)』
- ・ 『SpServer.cs -- C# external code implementation of procedures created in spcat.db2』
- ・ 『SpClient.cs -- Call different types of stored procedures from SpServer.java』
- ・ 『UDFcli.cs -- Client application that calls the user-defined functions 』
- ・ 『UDFsrv.cs -- User-defined scalar functions called by udfcli.cs』
- ・ 『bldrtn.bat -- Builds Visual Basic .NET routines (stored procedures and UDFs)』
- ・ 『SpServer.vb -- VB.NET implementation of procedures created in SpCat.db2』
- ・ 『SpClient.vb -- Call different types of stored procedures from SpServer.java』
- ・ 『UDFcli.vb -- Client application that calls the user-defined functions 』
- ・ 『UDFsrv.vb -- User-defined scalar functions called by udfcli.vb 』

C# .NET ルーチンのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldrtn.bat
rem Builds C# routines (stored procedures and UDFs)
rem Usage: bldrtn prog_name

rem When using the .NET Framework Version 1.1 point to netf11
set VERSION=netf11¥

rem Compile the program.
csc /out:%1.dll /target:library /debug /lib:"%DB2PATH%"¥bin¥netf11¥
/reference:"%DB2PATH%"¥bin¥%VERSION%IBM.Data.DB2.dll %1.cs

if exist "%DB2PATH%"¥function¥%1.dll" goto delete else goto copydll

:delete
del "%DB2PATH%"¥function¥%1.dll"
goto copydll

:copydll
rem Copy the routine assembly data link library to the 'function' directory
copy "%1.dll" "%DB2PATH%"¥function"

@echo on
```

Visual Basic .NET ルーチンのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldrtn.bat
rem Builds Visual Basic .NET routines (stored procedures and UDFs)
rem Usage: bldrtn prog_name

rem Compile the program.
vbc %1.vb /out:%1.dll /target:library /debug /libpath:"%DB2PATH%"¥bin¥netf11¥
/reference:IBM.Data.DB2.dll /reference:System.dll /reference:System.Data.dll
```

```
if exist "%DB2PATH%\function%\1.dll" goto delete else goto copydll

:delete
del "%DB2PATH%\function%\1.dll"
goto copydll

:copydll
rem Copy the routine assembly data link library to the 'function' directory
copy "%1.dll" "%DB2PATH%\function"

@echo on
```

CLR .NET ルーチンのコンパイルとリンクのオプション

以下は、>samples¥.NET¥cs¥bldrtn.bat および samples¥.NET¥vb¥bldrtn.bat バッチ・ファイルに示されているように、Windows 上で Microsoft Visual Basic .NET コンパイラーまたは Microsoft C# コンパイラーのどちらかを使用して、Common Language Runtime (CLR) .NET ルーチンを構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
Microsoft C# コンパイラーを使用したコンパイルとリンクのオプション	
csc	Microsoft C# コンパイラー
/out:%1.dll /target:library データ・リンク・ライブラリーをストアード・プロシーチャーのアセンブリー DLL として出力します。	
/debug デバッガーを使用します。	
/lib: "%DB2PATH%\bin¥netf11¥ .NET Framework 1.1 のライブラリー・パスを使用します。	
/reference:IBM.Data.DB2.dll .NET Framework 1.1 の DB2 データ・リンク・ライブラリーを使用します。	
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	
Microsoft Visual Basic コンパイラーを使用したコンパイルとリンクのオプション	
vbc	Microsoft Visual Basic コンパイラー。
/out:%1.dll /target:library データ・リンク・ライブラリーをストアード・プロシーチャーのアセンブリー DLL として出力します。	
/debug デバッガーを使用します。	
/libpath:"%DB2PATH%\bin¥netf11¥ .NET Framework 1.1 のライブラリー・パスを使用します。	
/reference:IBM.Data.DB2.dll .NET Framework 1.1 の DB2 データ・リンク・ライブラリーを使用します。	
/reference:System.dll Microsoft Windows の System データ・リンク・ライブラリーを参照します。	
/reference:System.Data.dll Microsoft Windows の System Data データ・リンク・ライブラリーを参照します。	
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

|

関連タスク:

- |
- 293 ページの『Common Language Runtime (CLR) .NET ルーチンの構築』

|

関連サンプル:

- |
- 『bldrtn.bat -- Builds C# routines (stored procedures and UDFs)』
 - |
 - 『bldrtn.bat -- Builds Visual Basic .NET routines (stored procedures and UDFs)』

Microsoft Visual C++

このセクションでは、組み込み SQL および DB2 API に加え、ActiveX Data Objects (ADO) およびオブジェクトのリンクと埋め込み (OLE) を使用したアプリケーションの構築について説明します。

DB2 CLI アプリケーションおよびルーチンの構築に関する情報は、「コール・レベル・インターフェース ガイドおよびリファレンス」内にあります。

Visual C++ による ADO アプリケーションの構築

ActiveX Data Object (ADO) を使用すれば、OLE DB Provider を使用して、データベース・サーバー内のデータにアクセスしたり、操作したりするアプリケーションを書くことができます。ADO の主要な利点は、速度が速く、使用が容易で、メモリーのオーバーヘッドが少なく、ディスク・フットプリントが小さいことです。

DB2 の sqllib¥samples¥VC に、Visual C++ ADO のサンプル・プログラムがあります。

手順:

以下の 2 つの ODBC 対応プロバイダーのどちらでも使用することができます。

- DB2 用の IBM OLE DB Provider
- ODBC 用の Microsoft OLE DB Provider

DB2 用の IBM OLE DB Provider の使用

|

|

|

Windows オペレーティング・システム上の DB2 バージョン 8.2 クライアントは、選択によっては、DB2 用の IBM OLE DB 2.0 準拠のプロバイダーである IBM DADB2 をインストールします。プロバイダーは、DB2 データベース内のデータにアクセスする消費者用のインターフェースを公開します。DB2 用の IBM OLE DB Provider は、以下の ADO アプリケーション・タイプをサポートします。

- Microsoft Active Server Pages (ASP)
- Microsoft Visual Studio C++ および Visual Basic アプリケーション
- Microsoft Visual Interdev

これらのタイプのアプリケーションの詳細については、ADO 資料を参照してください。

ODBC 用の Microsoft OLE DB Provider の使用

以下に示す変更を行うと、Microsoft OLE DB Provider および Visual C++ を使用する DB2 ADO プログラムは、正規の C++ プログラムと同じようにコンパイルできるようになります。

C++ ソース・プログラムを ADO プログラムとして実行するには、以下の IMPORT ステートメントをソース・プログラム・ファイルの先頭に置くことができます。

```
#import "C:\program files\common files\system\ado\msado<VERSION NUMBER>.dll" ¥
    no_namespace ¥
    rename( "EOF", "adoEOF")
```

<VERSION NUMBER> は、ADO ライブラリーのバージョン番号です。

プログラムがコンパイルされたら、ユーザーは msado<VERSION NUMBER>.dll が指定されたパスにあるかどうか検証する必要があります。C:\program files\common files\system\ado を環境変数 LIBPATH に追加し、短くした IMPORT ステートメントを以下のようにソース・ファイルで使用することもできます。

```
#import <msado<VERSION NUMBER>.dll> ¥
    no_namespace ¥
    rename( "EOF", "adoEOF")
```

DB2 サンプル・プログラム BLOBAccess.dsp では、この方法が使用されています。

この IMPORT ステートメントにより、DB2 プログラムには ADO ライブラリーへのアクセス権が与えられます。これで、Visual C++ プログラムも他のプログラムと同様にコンパイルできるようになります。また、DB2 API または DB2 CLI など、別のプログラミング・インターフェースを使用する場合、プログラム構築の詳細については、該当するトピックを参照してください。

関連概念:

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider for DB2 の目的』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider for DB2 でサポートされているアプリケーション・タイプ』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『C/C++ アプリケーションのコンパイルおよびリンクと IBM OLE DB Provider』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider による、C/C++ アプリケーションでのデータ・ソースへの接続』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider で自動的に使用可能になる OLE DB サービス』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider によるラージ・オブジェクトの操作』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider の制限』

関連資料:

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『DB2 と OLE DB の間のデータ・タイプ・マッピング』

- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『OLE DB タイプから DB2 タイプにデータを設定するためのデータ変換』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『DB2 タイプから OLE DB タイプにデータを設定するためのデータ変換』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での OLE DB コンポーネントおよびインターフェースのサポート』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での OLE DB プロパティのサポート』
- ・「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『IBM OLE DB Provider での ADO メソッドおよびプロパティのサポート』
- ・ 111 ページの『Visual C++ のサンプル』

Visual C++ でのオブジェクトのリンクと埋め込み (OLE) オートメーション

OLE では言語は限定されないので、任意の言語で OLE オートメーション UDF およびストアード・プロシージャをインプリメントすることができます。そのためには、OLE オートメーション・サーバーのメソッドを公開し、そのメソッドを UDF として DB2[®] に登録します。OLE オートメーション・サーバーの開発をサポートするアプリケーション開発環境には、以下の特定のバージョンが含まれます。Microsoft[®] Visual Basic、Microsoft Visual C++、Microsoft Visual J++、Microsoft FoxPro、Borland Delphi、Powersoft PowerBuilder、および Micro Focus COBOL。さらに、たとえば Microsoft Visual J++ のもののような、OLE 用に正しくラップされた Java[™] Bean オブジェクトにも、OLE オートメーションを介してアクセスすることができます。

OLE オートメーション・サーバーの開発の詳細については、該当するアプリケーション開発環境の資料を参照する必要があります。

OLE オートメーション UDF およびストアード・プロシージャ

Microsoft Visual C++ は、OLE オートメーション・サーバーの作成をサポートします。サーバーは、Microsoft Foundation Classes および Microsoft Foundation Class アプリケーション・ウィザードを使用して、または Win32 アプリケーションとして実現することができます。サーバーは、DLL または EXE にすることができます。詳細については、Microsoft Visual C++ の資料および Microsoft Visual C++ によって提供される OLE サンプルを参照してください。

DB2 は、Microsoft Visual C++ に含まれている OLE オートメーション UDF およびストアード・プロシージャのサンプルを提供しており、それはディレクトリー `sqllib\samples\ole\msvc` にあります。OLE オートメーション UDF およびストアード・プロシージャのサンプルを構築して実行する方法については、`sqllib\samples\ole` の README ファイルを参照してください。

関連概念:

- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『OLE オートメーション・ルーチンの設計』
- 「アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」の『BASIC および C++ での OLE オートメーション・ルーチン』

関連資料:

- 99 ページの『オブジェクトのリンクと埋め込み (OLE) のサンプル』

Windows での C/C++ アプリケーションの構築

DB2 には、DB2 API と組み込み SQL C/C++ プログラムのコンパイルとリンクのためのバッチ・ファイルが用意されています。このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib\samples\c` および `sqllib\samples\cpp` ディレクトリーに置かれています。

バッチ・ファイル `bldapp.bat` には、DB2 API と組み込み SQL プログラムを構築するためのコマンドが入っています。これは最大 4 個のパラメーターを取り、それらはバッチ・ファイル内で変数 `%1`、`%2`、`%3`、および `%4` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。これは組み込み SQL を使用しないプログラムに必要な唯一のパラメーターです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、さらに別の 3 つのパラメーターが用意されています。2 番目のパラメーター `%2` は、接続するデータベースの名前を指定します。3 番目のパラメーター `%3` は、データベースのユーザー ID を指定します。そして `%4` は、パスワードを指定します。

組み込み SQL プログラムの場合、`bldapp` は、プリコンパイルおよびバインドのファイル `embprep.bat` にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

以下の例は、DB2 API と組み込み SQL のアプリケーションを構築して実行する方法を示しています。

`sqllib\samples\c` のソース・ファイル `cli_info.c`、または `sqllib\samples\cpp` のソース・ファイル `cli_info.cxx` のどちらかから、DB2 API の組み込み SQL を含まないサンプル・プログラム `cli_info` を構築するには、次のように入力します。

```
bldapp cli_info
```

結果として、実行可能ファイル `cli_info.exe` が作成されます。この実行可能ファイルを実行するには、次の実行可能ファイル名を (拡張子なしで) コマンド行に入力します。

```
cli_info
```

組み込み SQL アプリケーションの構築と実行

sqllib%samples%c の C ソース・ファイル tbmod.sqc 、または
sqllib%samples%cpp の C++ ソース・ファイル tbmod.sqx から組み込み SQL アプリケーション tbmod を構築する方法は 3 つあります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

```
bldapp tbmod
```

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

```
bldapp tbmod database
```

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

```
bldapp tbmod database userid password
```

結果として、実行可能ファイル tbmod.exe が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある sample データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

```
tbmod
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
tbmod database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
tbmod database userid password
```

マルチスレッド・アプリケーションの構築と実行

Windows 用の C/C++ マルチスレッド・アプリケーションは、-MT または -MD オプションを指定してコンパイルする必要があります。-MT オプションを指定すると静的ライブラリー LIBCMT.LIB を使用してリンクされ、-MD を指定すると動的ライブラリー MSVCRT.LIB を使用してリンクされます。-MD を指定してリンクされたバイナリーはサイズは小さいですが MSVCRT.DLL がないと動作せず、-MT を指定してリンクされたバイナリーはサイズは大きくても実行時に必要なものを内蔵しています。

バッチ・ファイル bldmt.bat は、-MT オプションを使用してマルチスレッド・プログラムを構築します。その他のコンパイルとリンクのオプションは、バッチ・ファイル bldapp.bat が通常のスタンドアロン・アプリケーションを構築するとき使用するものと同じです。

ソース・ファイル samples%c%dbthdrs.sqc または samples%cpp%dbthdrs.sqx からマルチスレッド・サンプル・プログラム dbthdrs を構築するには、次のように入力します。

```
bldmt dbthdrs
```

結果として、実行可能ファイル dbthdrs.exe が作成されます。

このマルチスレッド・アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前 (拡張子なし) を入力します。

```
dbthrrds
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
dbthrrds database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
dbthrrds database userid password
```

関連資料:

- 304 ページの『Windows C/C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldapp.bat -- Builds C applications on Windows』
- 『bldmt.bat -- Builds C multi-threaded applications on Windows』
- 『cli_info.c -- Set and get information at the client level (C)』
- 『dbthrrds.sqc -- How to use multiple context APIs on Windows (C)』
- 『embprep.bat -- Prep and binds a C/C++ or Micro Focus COBOL embedded SQL program on Windows』
- 『tbmod.sqc -- How to modify table data (C)』
- 『bldapp.bat -- Builds C++ applications on Windows』
- 『bldmt.bat -- Builds C++ multi-threaded applications on Windows』
- 『cli_info.C -- Set and get information at the client level (C++)』
- 『dbthrrds.sqC -- How to use multiple context APIs on Windows (C++)』
- 『tbmod.sqC -- How to modify table data (C++)』

C/C++ アプリケーションのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldapp.bat
rem Builds C/C++ applications on Windows
rem Usage: bldapp prog_name [ db_name [ userid password ]]

rem Default compiler is set to Microsoft Visual C++
rem To use a different compiler, comment out 'set BLDCOMP=c1'
rem and uncomment 'set BLDCOMP=icl' or 'set BLDCOMP=ec1'
rem Microsoft C/C++ Compiler
set BLDCOMP=c1

rem Intel C++ Compiler for 32-bit applications
rem set BLDCOMP=icl

rem Intel C++ Compiler for Itanium 64-bit applications
rem set BLDCOMP=ec1

if exist "%1.sqx" goto embedded
if exist "%1.sqc" goto embedded
goto non_embedded
```

```

:embedded
rem Precompile and bind the program.
rem Note: some .sqc/.sqx files contain no SQL but link in
rem utilemb.sqc/.sqx, so if you get this warning, ignore it:
rem SQL0053W No SQL statements were found in the program.
call embprep %1 %2 %3 %4
rem Compile the program.
if exist "%1.cxx" goto cpp_emb
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 %1.c utilemb.c
goto link_embedded
:cpp_emb
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 %1.cxx utilemb.cxx
rem Link the program.
:link_embedded
link -debug -out:%1.exe %1.obj utilemb.obj db2api.lib
goto exit

:non_embedded
rem Compile the program.
if exist "%1.cxx" goto cpp_non
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 %1.c utilapi.c
goto link_non_embedded
:cpp_non
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 %1.cxx utilapi.cxx
rem Link the program.
:link_non_embedded
link -debug -out:%1.exe %1.obj utilapi.obj db2api.lib
:exit
@echo on

```

Windows C/C++ アプリケーションのコンパイルとリンクのオプション

以下は、bldapp.bat バッチ・ファイルに示されているように、Windows 上で Microsoft Visual C++ コンパイラーを使用して、C/C++ 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
%BLDCOMP%	コンパイラー用の変数です。デフォルトは、cl (Microsoft Visual C++ コンパイラー) です。またこれは、icl、32 ビット・アプリケーション用の Intel C++ コンパイラー、または icl (Itanium 64 ビット・アプリケーション用の Intel C++ コンパイラー) に設定することもできます。
-Zi	デバッグ情報を使用可能にします。
-Od	最適化なし。最適化をオフにしてデバッガーを使用する方が簡単です。
-c	コンパイルのみを実行し、リンクは実行しません。 バッチ・ファイルでは、コンパイルとリンクは別個のステップです。
-W2	警告、エラー、重大、およびリカバリー不能エラー・メッセージを出力します。
-DWIN32	オペレーティング・システムに必要なコンパイラー・オプション。

bldapp のコンパイルとリンクのオプション	
リンク・オプション	
link	リンクにリンカーを使用します。
-debug	デバッグ情報を組み込みます。
-out:%1.exe	ファイル名を指定します。
%1.obj	オブジェクト・ファイルを組み込みます。
utilemb.obj	組み込み SQL プログラムの場合に、エラー・チェックを行う組み込み SQL ユーティリティ・オブジェクト・ファイルを含みます。
utilapi.obj	組み込み SQL プログラムでない場合に、エラー・チェックを行う DB2 API ユーティリティ・オブジェクト・ファイルを含みます。
db2api.lib	DB2 ライブラリーとリンクします。
他のコンパイラ・オプションについては、コンパイラの資料をご覧ください。	

関連タスク:

- 301 ページの『Windows での C/C++ アプリケーションの構築』

関連サンプル:

- 『bldapp.bat -- Builds C applications on Windows』
- 『bldapp.bat -- Builds C++ applications on Windows』

Windows での C/C++ ルーチンの構築

DB2 には、C および C++ の DB2 API と組み込み SQL C/C++ プログラムのコンパイルとリンクのためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sql1lib%samples%c` および `sql1lib%samples%c` ディレクトリーに置かれています。

バッチ・ファイル `bldrtn.bat` には、組み込み SQL ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するためのコマンドが入っています。バッチ・ファイルは、サーバー上に DLL を構築します。これは 2 個のパラメーターをとります。それらは、バッチ・ファイル内では変数 `%1` と `%2` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。このバッチ・ファイルは、DLL 名としてソース・ファイル名を使用します。第 2 パラメーター `%2` には、接続先のデータベースの名前を指定します。DLL は、データベースが置かれているのと同じインスタンス上で構築する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

第 1 パラメーター (ソース・ファイル名) だけが必須です。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。

手順:

この後の例は、次のものを使用してルーチンの DLL を構築する方法を示しています。

- ストアード・プロシージャ
- 非組み込み SQL ユーザー定義関数 (UDF)
- 組み込み SQL ユーザー定義関数 (UDF)

ストアード・プロシージャの DLL

C ソース・ファイル `spserver.sqc` または C++ ソース・ファイル `spserver.sqx` から `spserver` DLL を構築するには、次のようにします。

1. 次のように、バッチ・ファイル名とプログラム名を入力します。

```
bldrtn spserver
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn spserver database
```

このバッチ・ファイルは、サンプル・プログラムと同じディレクトリーに入っている、モジュール定義ファイル `spserver.def` を使用して DLL を構築します。このバッチ・ファイルは、DLL の `spserver.dll` をパス `sqllib%function` のサーバーにコピーします。

2. 次に、サーバーで `spcat` スクリプトを実行してルーチンをカタログします。

```
spcat
```

このスクリプトは、サンプル・データベースに接続し、ルーチンがすでにカタログ済みであれば `spdrop.db2` を呼び出してルーチンをアンカタログし、次に `spcreate.db2` を呼び出してそのルーチンをカタログし、そして最後にデータベースへの接続を切断します。また、`spdrop.db2` スクリプトと `spcreate.db2` スクリプトは、個別に実行することもできます。

3. 次に、データベースを一度停止してから再始動し、新しい DLL が認識されるようにします。必要であれば、DLL にファイル・モードを設定して、DB2 インスタンスからアクセスできるようにします。

DLL `spserver` を構築し終わったら、それを呼び出すクライアント・アプリケーション `spclient` を構築することができます。

`spclient` は、バッチ・ファイル `bldapp.bat` を使用して構築することができます。

DLL を呼び出すには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
spclient database userid password
```

ここで、

database

接続先のデータベースの名前です。名前は、`sample` かその別名、またはその他のデータベース名にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは、DLL `spserver` にアクセスしてサーバー・データベース上のいくつかのルーチンを実行します。出力は、クライアント・アプリケーションに戻されます。

非組み込み SQL UDF の DLL

ユーザー定義関数 `udfsrv` をソース・ファイル `udfsrv.c` から構築するには、次のように入力します。

```
bldrtn udfsrv
```

このバッチ・ファイルは、サンプル・プログラム・ファイルと同じディレクトリーに入っているモジュール定義ファイル `udfsrv.def` を使用して、ユーザー定義関数 DLL を構築します。このバッチ・ファイルは、ユーザー定義関数 DLL の `udfsrv.dll` を `sqllib¥function` というパスのサーバーにコピーします。

`udfsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfcli` を構築できます。DB2 CLI が、このプログラムの組み込み SQL C および C++ バージョンとともに提供されます。

DB2 CLI `udfcli` プログラムは、`sqllib¥samples¥cli` のバッチ・ファイル `bldapp` を使用して、`udfcli.c` ソース・ファイルから構築できます。

組み込み SQL C `udfcli` プログラムは、`sqllib¥samples¥c` のバッチ・ファイル `bldapp` を使用して、`udfcli.sqc` ソース・ファイルから構築できます。

組み込み SQL C++ `udfcli` プログラムは、`sqllib¥samples¥cpp` のバッチ・ファイル `bldapp` を使用して、`udfcli.sqx` ソース・ファイルから構築できます。

UDF を実行するには、次のように入力します。

```
udfcli
```

この呼び出しアプリケーションは、`udfsrv` DLL から `ScalarUDF` 関数を呼び出します。

組み込み SQL UDF の DLL

`sqllib¥samples¥c` 内の C ソース・ファイル `udfemsrv.sqc` からか、または `sqllib¥samples¥cpp` 内の C++ ソース・ファイル `udfemsrv.sqx` から、組み込み SQL ユーザー定義関数ライブラリー `udfemsrv` を構築するには、次のように入力します。

```
bldrtn udfemsrv
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn udfemsrv database
```

このバッチ・ファイルは、サンプル・プログラムと同じディレクトリーに入っているモジュール定義ファイル `udfemsrv.def` を使用してユーザー定義関数 DLL を構

築します。このバッチ・ファイルは、ユーザー定義関数 DLL の `udfemsrv.dll` をパス `sqllib¥function` のサーバーにコピーします。

`udfemsrv` の構築が完了したら、それを呼び出すクライアント・アプリケーション `udfemcli` を構築できます。バッチ・ファイル `bldapp` を使用して、`sqllib¥samples¥c` 内の C ソース・ファイル `udfemcli.sqc` からか、または `sqllib¥samples¥cpp` 内の C++ ソース・ファイル `udfemcli.sqx` から、`udfemcli` を構築することができます。

UDF を実行するには、次のように入力します。

```
udfemcli
```

呼び出し側アプリケーションは、`udfemsrv` DLL 内の UDF を呼び出します。

関連資料:

- 309 ページの『Windows C/C++ ルーチンのコンパイルとリンクのオプション』

関連サンプル:

- 『`bldrtn.bat` -- Builds C routines (stored procedures and UDFs) on Windows』
- 『`embprep.bat` -- Prep and binds a C/C++ or Micro Focus COBOL embedded SQL program on Windows』
- 『`spclient.sqc` -- Call various stored procedures (C)』
- 『`spserver.sqc` -- Definition of various types of stored procedures (C)』
- 『`udfcli.sqc` -- Call a variety of types of user-defined functions (C)』
- 『`udfemcli.sqc` -- Call a variety of types of embedded SQL user-defined functions. (C)』
- 『`udfemsrv.sqc` -- Call a variety of types of embedded SQL user-defined functions. (C)』
- 『`udfsrv.c` -- Defines a variety of types of user-defined functions (C)』
- 『`bldrtn.bat` -- Builds C++ routines (stored procedures and UDFs) on Windows』
- 『`spclient.sqC` -- Call various stored procedures (C++)』
- 『`spserver.sqC` -- Definition of various types of stored procedures (C++)』
- 『`udfcli.sqC` -- Call a variety of types of user-defined functions (C++)』
- 『`udfemcli.sqC` -- Call a variety of types of embedded SQL user-defined functions. (C++)』
- 『`udfemsrv.sqC` -- Call a variety of types of embedded SQL user-defined functions. (C++)』
- 『`udfsrv.C` -- Defines a variety of types of user-defined functions (C++)』

C/C++ ルーチンのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldrtn.bat
rem Builds C/C++ routines (stored procedures and UDFs) on Windows
rem Usage: bldrtn prog_name [ db_name ]

rem Default compiler is set to Microsoft Visual C++
rem To use a different compiler, comment out 'set BLDCOMP=c1'
rem and uncomment 'set BLDCOMP=icl' or 'set BLDCOMP=ec1'
```

```

rem Microsoft C/C++ Compiler
set BLDCOMP=c1

rem Intel C++ Compiler for 32-bit applications
rem set BLDCOMP=icl

rem Intel C++ Compiler for Itanium 64-bit applications
rem set BLDCOMP=ec1

if exist "%1.sqc" goto embedded
if exist "%1.sqx" goto embedded
goto compile

:embedded
rem Precompile and bind the program.
call embprep %1 %2

:compile
rem Compile the program.
if exist "%1.cxx" goto cpp
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 -MD %1.c
goto link_step
:cpp
%BLDCOMP% -Zi -Od -c -W2 -DWIN32 -MD %1.cxx

:link_step
rem Link the program.
link -debug -out:%1.dll -dll %1.obj db2api.lib -def:%1.def

rem Copy the routine DLL to the 'function' directory
copy %1.dll "%DB2PATH%\function"
@echo on

```

Windows C/C++ ルーチンのコンパイルとリンクのオプション

以下は、bldrtn.bat バッチ・ファイルに示されているように、Windows 上で Microsoft Visual C++ コンパイラーを使用して、C/C++ ルーチン (ストアード・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
%BLDCOMP%	コンパイラー用の変数です。デフォルトは、c1 (Microsoft Visual C++ コンパイラー) です。またこれは、icl、32 ビット・アプリケーション用の Intel C++ コンパイラー、または ec1 (Itanium 64 ビット・アプリケーション用の Intel C++ コンパイラー) に設定することもできます。
-Zi	デバッグ情報を使用可能にします。
-Od	最適化なし。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。
-W2	警告、エラー、重大、およびリカバリー不能エラー・メッセージを出力します。
-DWIN32	オペレーティング・システムに必要なコンパイラー・オプション。
-MD	MSVCRT.LIB を使用してマルチスレッド DLL を作成します。

bldrtn のコンパイルとリンクのオプション	
リンク・オプション	
link	リンクにリンカーを使用します。
-debug	デバッグ情報を組み込みます。
-out:%1.dll	.DLL ファイルを構築します。
%1.obj	オブジェクト・ファイルを組み込みます。
db2api.lib	DB2 ライブラリーとリンクします。
-def:%1.def	モジュール定義ファイル。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 305 ページの『Windows での C/C++ ルーチンの構築』

関連サンプル:

- 『bldrtn.bat -- Builds C routines (stored procedures and UDFs) on Windows』
- 『bldrtn.bat -- Builds C++ routines (stored procedures and UDFs) on Windows』

Windows での C/C++ 複数接続アプリケーションの構築

DB2 には、C および C++ 組み込み SQL と DB2 API プログラムをコンパイルしてリンクするためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib%samples%c` および `sqllib%samples%c.cpp` ディレクトリーに置かれています。

バッチ・ファイル `bldmc.bat` には、2 つのデータベースを必要とする DB2 複数接続アプリケーション・プログラムを作成するコマンドが入っています。コンパイルとリンクのオプションは、`bldapp.bat` ファイルが使用するものと同じです。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。第 2 パラメーター `%2` には、接続先の 1 つ目のデータベースの名前を指定します。第 3 パラメーター `%3` には、接続先の 2 つ目のデータベースの名前を指定します。これらはすべて必要パラメーターです。

注: `makefile` にはデータベース名に `"sample"` と `"sample2"` の値が (それぞれ `%2` と `%3`) ハードコーディングされているので、この `makefile` を使用する場合は、プログラム名 (`%1` パラメーター) の指定だけ行います。 `bldmc.bat` スクリプトを使用する場合は、3 つのパラメーターすべてを指定します。

オプション・パラメーターは、ローカル接続の場合は必要ありませんが、リモート・クライアントからサーバーに接続する場合は必要です。オプション・パラメーターは次のとおりです。 `%4` と `%5` はそれぞれ第 1 データベースのユーザー ID とパスワードを指定し、 `%6` と `%7` はそれぞれ第 2 データベースのユーザー ID とパスワードを指定します。

手順:

複数接続サンプル・プログラム dbmcon.exe には、2 つのデータベースが必要です。sample データベースをまだ作成していなければ、DB2 コマンド・ウィンドウのコマンド行で db2samp1 と入力して作成できます。2 番目のデータベース (ここでは sample2 という名前) は、以下のいずれかのコマンドを使用して作成できます。

データベースをローカルで作成する場合:

```
db2 create db sample2
```

データベースをリモートから作成する場合:

```
db2 attach to <node_name>
db2 create db sample2
db2 detach
db2 catalog db sample2 as sample2 at node <node_name>
```

<node_name> は、データベースが常駐するノードです。

複数接続では、TCP/IP Listener も実行されている必要があります。これを確実に実行するには、次のようにします。

1. 環境変数 DB2COMM を TCP/IP に設定します。

```
db2set DB2COMM=TCPIP
```

2. サービス・ファイルで指定されるように、データベース・マネージャーの構成ファイルを TCP/IP サービス名で更新します。

```
db2 update dbm cfg using SVCENAME <TCP/IP service name>
```

各インスタンスは TCP/IP サービス名を持っており、この名前はサービス・ファイルにリストされています。このファイルが見つからない場合、またはサービス・ファイルを変更するファイル許可がない場合は、システム管理者に連絡してください。

3. データベース・マネージャーをいったん停止してから再始動して、これらの変更を有効にします。

```
db2stop
db2start
```

dbmcon.exe プログラムは、samples%c または samples%c%% ディレクトリーにある次の 5 つのファイルから作成されます。

dbmcon.sqc または dbmcon.sqx

両方のデータベースに接続するためのメイン・ソース・ファイル。

dbmcon1.sqc または dbmcon1.sqx

最初のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon1.h

主ソース・ファイル dbmcon.sqc または dbmcon.sqx に組み込まれた dbmcon1.sqc または dbmcon1.sqx のヘッダー・ファイルで、最初のデータベースにバインドされた表を作成およびドロップするための SQL ステートメントにアクセスするためのものです。

dbmcon2.sqc または dbmcon2.sqx

2 番目のデータベースにバインドされるパッケージを作成するためのソース・ファイル。

dbmcon2.h

主ソース・ファイル dbmcon2.sqc または dbmcon2.sqx に組み込まれた dbmcon.sqc または dbmcon.sqx のヘッダー・ファイルで、2 番目のデータベースにバインドされた表を作成およびドロップするための SQL ステートメントにアクセスするためのものです。

複数接続サンプル・プログラム dbmcon.exe を構築するには、次のようにします。

```
bldmc dbmcon sample sample2
```

結果として、実行可能ファイル dbmcon.exe が作成されます。

この実行可能ファイルを実行するには、実行可能ファイル名を拡張子なしで実行します。

```
dbmcon
```

2 つのデータベースへの 1 フェーズ・コミットが、プログラムによって例示されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 「管理ガイド: パフォーマンス」の『svcename - 「TCP/IP サービス名」構成パラメーター』
- 304 ページの『Windows C/C++ アプリケーションのコンパイルとリンクのオプション』

関連サンプル:

- 『bldmc.bat -- Builds C multi-connection application on Windows』
- 『dbmcon.sqc -- How to use multiple databases (C)』
- 『dbmcon1.h -- Function declarations for the source file, dbmcon1.sqc (C)』
- 『dbmcon1.sqc -- Functions used in the multiple databases program dbmcon.sqc (C)』
- 『dbmcon2.h -- Function declarations for the source file, dbmcon2.sqc (C)』
- 『dbmcon2.sqc -- Functions used in the multiple databases program dbmcon.sqc (C)』
- 『bldmc.bat -- Builds C++ multi-connection application on Windows』
- 『dbmcon.sqC -- How to use multiple databases (C++)』
- 『dbmcon1.h -- Class declaration for the source file, dbmcon1.sqC (C++)』
- 『dbmcon1.sqC -- Functions used in the multiple databases program dbmcon.sqC (C++)』
- 『dbmcon2.h -- Class declaration for the source file, dbmcon2.sqC (C++)』
- 『dbmcon2.sqC -- Functions used in the multiple databases program dbmcon.sqC (C++)』

Windows での IBM COBOL コンパイラーの構成

組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合に、IBM VisualAge COBOL コンパイラーを使用するときは、気をつける点がいくつかあります。

手順:

- DB2 プリコンパイラーでアプリケーションをプリコンパイルし、コマンド行プロセッサのコマンド `db2 prep` を使用する場合は、`target ibmcob` オプションを使用してください。
- ソース・ファイルの中でタブ文字を使用しないでください。
- ソース・ファイル内で `PROCESS` および `CBL` キーワードを使用して、コンパイル・オプションを設定します。キーワードは 8~72 桁目だけに置いてください。
- アプリケーションに組み込み SQL のみが含まれていて、DB2 API 呼び出しは含まれない場合には、`pgmname(mixed)` コンパイル・オプションを使う必要はありません。DB2 API 呼び出しを使用する場合には、`pgmname(mixed)` コンパイル・オプションを使う必要があります。
- IBM VisualAge COBOL コンパイラーの「システム/390 ホスト・データ・タイプ・サポート」機能を使用している場合、アプリケーション用の DB2 組み込みファイルは、次のディレクトリーの中にあります。

```
%DB2PATH%\include%cobol_i
```

提供されたバッチ・ファイルを使用して DB2 サンプル・プログラムを構築している場合、バッチ・ファイルで指定された組み込みファイルのパスは、`cobol_a` ディレクトリーではなく、`cobol_i` ディレクトリーを指すように変更しなければなりません。

IBM VisualAge COBOL コンパイラーの「システム/390 ホスト・データ・タイプ・サポート」機能を使用していない場合、またはこのコンパイラーのそれよりも前のバージョンを使用している場合、アプリケーション用の DB2 組み込みファイルは、次のディレクトリー中にあります。

```
%DB2PATH%\include%cobol_a
```

`cobol_a` ディレクトリーはデフォルトです。

- 次のように、`COPY` ファイル名を `.cbl` 拡張子を含めて指定します。

```
COPY "sql.cbl"
```

関連タスク:

- 314 ページの『Windows での IBM COBOL アプリケーションの構築』
- 318 ページの『Windows での IBM COBOL ルーチンの構築』

関連資料:

- 317 ページの『Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション』

- 320 ページの『Windows IBM COBOL ルーチンのコンパイルとリンクのオプション』

Windows での IBM COBOL アプリケーションの構築

DB2 には、DB2 API と組み込み SQL プログラムのコンパイルとリンクのためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib\samples\cobol` ディレクトリーに置かれています。

DB2 は、Windows での IBM COBOL アプリケーションの構築用に、DB2 プリコンパイラーと IBM COBOL プリコンパイラーの 2 種類のプリコンパイラーをサポートします。デフォルトは DB2 プリコンパイラーです。使用するバッチ・ファイルの該当する行をコメント化することにより、IBM COBOL プリコンパイラーを選択できます。IBM COBOL でのプリコンパイルは、特定のプリコンパイル・オプションを使用して、コンパイラー単体で実行できます。

バッチ・ファイル `bldapp.bat` には、DB2 アプリケーション・プログラムを構築するコマンドが入っています。これは最大 4 個のパラメーターを取り、それらはバッチ・ファイル内で変数 `%1`、`%2`、`%3`、および `%4` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。これは組み込み SQL を使用しないプログラムに必要な唯一のパラメーターです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、3 つのパラメーターがオプションとして用意されています。2 番目のパラメーターは `%2` で、接続するデータベースの名前を指定します。3 番目のパラメーターは `%3` で、データベースのユーザー ID を指定します。そしてもう 1 つが `%4` で、データベースのパスワードを指定します。

デフォルトの DB2 プリコンパイラーを使用する組み込み SQL プログラムの場合、`bldapp.bat` は、プリコンパイルおよびバインドのファイル `embprep.bat` にパラメーターを渡します。

IBM COBOL プリコンパイラーを使用する組み込み SQL プログラムの場合、`bldapp.bat` は `.sqb` ソース・ファイルを `.cbl` ソース・ファイルにコピーします。コンパイラーは `.cbl` ソース・ファイルに対して特定のプリコンパイル・オプションを使用して、プリコンパイルを実行します。

いずれのプリコンパイラーの場合も、データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

以下の例は、DB2 API と組み込み SQL のアプリケーションを構築して実行する方法を示しています。

ソース・ファイル `client.cbl` から組み込み SQL を含まないサンプル・プログラム `client` を構築するには、次のように入力します。

```
bldapp client
```

結果として、実行可能ファイル `client.exe` が作成されます。この実行可能ファイルを `sample` データベースに対して実行するには、次の実行可能名を（拡張子なしで）入力します。

`client`

組み込み SQL アプリケーションの構築と実行

ソース・ファイル `updat.sqb` から組み込み SQL アプリケーション `updat` を構築する方法には、次の 3 つがあります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

`bldapp updat`

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

`bldapp updat database`

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

`bldapp updat database userid password`

結果として、実行可能ファイル `updat` が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前を入力します。

`updat`

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

`updat database`

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

`updat database userid password`

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 317 ページの『Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『`bldapp.bat` -- Builds Windows VisualAge COBOL applications』
- 『`client.cbl` -- How to set and query a client (IBM COBOL)』
- 『`embprep.bat` -- To prep and bind a COBOL embedded SQL program on Windows』
- 『`updat.sqb` -- How to update, delete and insert table data (IBM COBOL)』

IBM COBOL アプリケーションのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldapp.bat
rem Builds Windows VisualAge COBOL applications
rem Usage: bldapp prog_name [ db_name [ userid password ]]

set IBMCOB_PRECOMP=
set EXTRA_COMPFLAG=

rem To use the IBM COBOL precompiler, uncomment the following line.
rem set IBMCOB_PRECOMP=true

rem If using the IBM COBOL precompiler
if "%IBMCOB_PRECOMP%" == "true" goto IBMCOB_precompile_step

rem Using the default DB2 precompiler,
rem If an embedded SQL program, precompile and bind it.
if not exist "%1.sqb" goto compile_step
call embprep %1 %2 %3 %4
goto compile_step

:IBMCOB_precompile_step
rem Using the IBM COBOL precompiler,
rem Copy the <prog_name>.sqb file to <prog_name>.cbl.
if exist "%1.sqb" cp -f %1.sqb %1.cbl
rem Assign input parameters to the EXTRA_COMPFLAG variable
if "%1" == "" goto error
if "%2" == "" goto case1
if "%3" == "" goto case2
if "%4" == "" goto error
goto case3

:case1
set EXTRA_COMPFLAG=-q"SQL('database sample CALL_RESOLUTION DEFERRED')"
goto compile_step
:case2
set EXTRA_COMPFLAG=-q"SQL('database %2 CALL_RESOLUTION DEFERRED')"
goto compile_step
:case3
set EXTRA_COMPFLAG=-q"SQL('database %2 user %3 using %4
CALL_RESOLUTION DEFERRED')"
goto compile_step

:compile_step
rem Compile the error-checking utility.
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include%cobol_a" checkerr.cbl

rem Compile the program.
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include%cobol_a" %1.cbl
%EXTRA_COMPFLAG%

rem Link the program.
cob2 %1.obj checkerr.obj db2api.lib
goto exit

:error
echo Usage: bldapp prog_name [ db_name [ userid password ]]

:exit
@echo on
```


Windows IBM COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp.bat バッチ・ファイルに示されているように、 Windows 上で IBM VisualAge COBOL コンパイラーを使用して、 COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cob2	IBM VisualAge COBOL コンパイラー。
-qpgmname(mixed)	コンパイラーに、大文字小文字混合の名前を持つライブラリーのエンタリー・ポイントの CALL を許可するように指示します。
-c	コンパイルのみを実行し、リンクは実行しません。コンパイルとリンクは別個のステップです。
-qlib	コンパイラーに COPY ステートメントを処理するように指示します。
-Ipath	DB2 組み込みファイルのロケーションを指定します。たとえば、 -I"%DB2PATH%\%include%cobol_a"。
%EXTRA_COMPFLAG%	"set IBMCOB_PRECOMP=true" のコメント化が解除されている場合、入力パラメーターに応じて、以下のいずれかの組み合わせの IBM COBOL プリコンパイル・オプションが使用されます。
-q"SQL('database sample CALL_RESOLUTION DEFERRED')"	デフォルトのサンプル・データベースを使用してプリコンパイルし、呼び出し解決を据え置きます。
-q"SQL('database %2 CALL_RESOLUTION DEFERRED')"	ユーザー指定のデータベースを使用してプリコンパイルし、呼び出し解決を据え置きます。
-q"SQL('database %2 user %3 using %4 CALL_RESOLUTION DEFERRED')"	ユーザー指定のデータベース、ユーザー ID、およびパスワードを使用してプリコンパイルし、呼び出し解決を据え置きます。これは、リモート・クライアント・アクセス用の書式です。
リンク・オプション	
cob2	コンパイラーをリンカーのフロントエンドとして使用します。
%1.obj	プログラム・オブジェクト・ファイルを組み込みます。
checkerr.obj	エラー・チェック・ユーティリティー・オブジェクト・ファイルを組み込みます。
db2api.lib	DB2 ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 314 ページの『Windows での IBM COBOL アプリケーションの構築』

関連サンプル:

- 『bldapp.bat -- Builds Windows VisualAge COBOL applications』

Windows での IBM COBOL ルーチンの構築

DB2 には、DB2 API と組み込み SQL プログラムを IBM COBOL でコンパイルおよびリンクするためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib\samples\coba1` ディレクトリーに置かれています。

DB2 は、Windows での IBM COBOL アプリケーションの構築用に、DB2 プリコンパイラーと IBM COBOL プリコンパイラーの 2 種類のプリコンパイラーをサポートします。デフォルトは DB2 プリコンパイラーです。使用するバッチ・ファイルの該当する行をコメント化することにより、IBM COBOL プリコンパイラーを選択できます。IBM COBOL でのプリコンパイルは、特定のプリコンパイル・オプションを使用して、コンパイラー単体で実行できます。

バッチ・ファイル `bldrtn.bat` には、組み込み SQL ルーチン (ストアド・プロシージャ) を構築するためのコマンドが入っています。このバッチ・ファイルは、サーバー上の DLL 内でルーチンをコンパイルします。これは 2 個のパラメーターをとります。それらは、バッチ・ファイル内では変数 `%1` と `%2` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。バッチ・ファイルでは、ソース・ファイル名 `%1` を DLL 名に使用します。第 2 パラメーター `%2` には、接続先のデータベースの名前を指定します。ストアド・プロシージャは、データベースが置かれているのと同じインスタンス上で構築する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

最初のパラメーター (ソース・ファイル名) だけが、必須です。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。

デフォルトの DB2 プリコンパイラーを使用している場合、`bldrtn.bat` は、プリコンパイルおよびバインドのファイル `embprep.bat` にパラメーターを渡します。

IBM COBOL プリコンパイラーを使用している場合、`bldrtn.bat` は `.sqb` ソース・ファイルを `.cbl` ソース・ファイルにコピーします。コンパイラーは `.cbl` ソース・ファイルに対して特定のプリコンパイル・オプションを使用して、プリコンパイルを実行します。

手順:

サンプル・データベースに接続している場合、ソース・ファイル `outsrv.sqb` からサンプル・プログラム `outsrv` を構築するには、次のように入力します。

```
bldrtn outsrv
```

他のデータベースに接続しているときは、さらにデータベース名も含めます。

```
bldrtn outsrv database
```

バッチ・ファイルは、DLL をサーバー上の `sqllib\function` というパスにコピーします。

DLL outsrv の構築が完了したら、DLL 内のルーチン (DLL と同名) を呼び出すクライアント・アプリケーション outcli を構築することができます。outcli は、bldapp.bat バッチ・ファイルを使用して構築することができます。

outsrv ルーチンを呼び出すには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
outcli database userid password
```

ここで、

database

接続先のデータベースの名前です。名前は、sample またはそのリモート別名、あるいはその他の名前にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは DLL outsrv にアクセスし、同一名のルーチンをサーバー・データベース上で実行します。この出力は、クライアント・アプリケーションに戻されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 320 ページの『Windows IBM COBOL ルーチンのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『bldrtn.bat -- Builds Windows VisualAge COBOL routines (stored procedures)』
- 『embprep.bat -- To prep and bind a COBOL embedded SQL program on Windows』
- 『outcli.sqb -- Call stored procedures using the SQLDA structure (IBM COBOL)』
- 『outsrv.sqb -- Demonstrates stored procedures using the SQLDA structure (IBM COBOL)』

IBM COBOL ルーチンのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldrtn.bat
rem Builds Windows VisualAge COBOL routines (stored procedures)
rem Usage: bldrtn prog_name [ db_name ]

set IBMCOB_PRECOMP=
set EXTRA_COMPFLAG=

rem To use the IBM COBOL precompiler, uncomment the following line.
rem set IBMCOB_PRECOMP=true

rem If using the IBM COBOL precompiler
if "%IBMCOB_PRECOMP%" == "true" goto IBMCOB_precompile_step
```

```

rem Using the default DB2 precompiler,
rem Precompile and bind the program.
call embprep %1 %2
    goto compile_step

:IBMCOB_precompile_step
rem Using the IBM COBOL precompiler,
rem Copy the <prog_name>.sqb file to <prog_name>.cbl.
if exist "%1.sqb" cp -f %1.sqb %1.cbl
rem Assign input parameters to the EXTRA_COMPFLAG variable
if "%1" == "" goto error
if "%2" == "" goto case1

    set EXTRA_COMPFLAG=-q"SQL('database %2 CALL_RESOLUTION DEFERRED')"
    goto compile_step

:case1
    set EXTRA_COMPFLAG=-q"SQL('database sample CALL_RESOLUTION DEFERRED')"
    goto compile_step

:compile_step
rem Compile the stored procedure.
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include\cobol_a" %1.cbl %EXTRA_COMPFLAG%

rem Link the stored procedure and create a shared library.
ilib /no1 /gi:%1 %1.obj
ilink /free /no1 /dll db2api.lib %1.exp %1.obj iwzrwin3.obj

rem Copy stored procedure to the %DB2PATH%\function directory.
copy %1.dll "%DB2PATH%\function"
goto exit

:error
echo Usage: bldrtn prog_name [ db_name ]

:exit
@echo on

```

Windows IBM COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn.bat バッチ・ファイルに示されているように、Windows 上で IBM VisualAge COBOL コンパイラーを使用して、COBOL ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
<p>コンパイル・オプション</p> <p>cob2 IBM VisualAge COBOL コンパイラー。</p> <p>-qpgmname(mixed) コンパイラーに、大文字小文字混合の名前を持つライブラリーのエントリー・ポイントの CALL を許可するように指示します。</p> <p>-c コンパイルのみを実行し、リンクは実行しません。このバッチ・ファイルでは、コンパイルとリンクは別個のステップです。</p> <p>-qlib コンパイラーに COPY ステートメントを処理するように指示します。</p> <p>-Ipath DB2 組み込みファイルのロケーションを指定します。たとえば、 -I"%DB2PATH%\include\cobol_a"。</p> <p>%EXTRA_COMPFLAG% "set IBMCOB_PRECOMP=true" のコメント化が解除されている場合、入力パラメータに応じて、以下のいずれかの組み合わせの IBM COBOL プリコンパイル・オプションが使用されます。</p> <p>-q"SQL('database sample CALL_RESOLUTION DEFERRED')" デフォルトのサンプル・データベースを使用してプリコンパイルし、呼び出し解決を据え置きます。</p> <p>-q"SQL('database %2 CALL_RESOLUTION DEFERRED')" ユーザー指定のデータベースを使用してプリコンパイルし、呼び出し解決を据え置きます。</p>	
<p>リンク・オプション</p> <p>ilink IBM VisualAge COBOL リンカーを使用します。</p> <p>/free 自由書式。</p> <p>/no1 ログなし。</p> <p>/dl1 DLL をソース・プログラム名を使用して作成します。</p> <p>db2api.lib DB2 ライブラリーとリンクします。</p> <p>%1.exp エクスポート・ファイルを組み込みます。</p> <p>%1.obj プログラム・オブジェクト・ファイルを組み込みます。</p> <p>iwzrwin3.obj IBM VisualAge COBOL が提供するオブジェクト・ファイルを組み込みます。</p> <p>他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。</p>	

関連タスク:

- 318 ページの『Windows での IBM COBOL ルーチンの構築』

関連サンプル:

- 『bldrtn.bat -- Builds Windows VisualAge COBOL routines (stored procedures)』

Windows での Micro Focus COBOL コンパイラーの構成

組み込み SQL および DB2 API 呼び出しの入ったアプリケーションを開発する場合に、Micro Focus コンパイラーを使用するときは、気をつける点がいくつかあります。

手順:

- コマンド行プロセッサのコマンド `db2 prep` を使用してアプリケーションをプリコンパイルする場合は、`target mfcob` オプションを使用してください。
- 以下のように入力して、必ず `LIB` 環境変数が `%DB2PATH%\lib` を指すようにしてください。

```
set LIB="%DB2PATH%\lib;%LIB%"
```

- Micro Focus COBOL 用の `DB2 COPY` ファイルは、`%DB2PATH%\include\cobol_mf` にあります。 `COBCPY` 環境変数を、以下のようにディレクトリーを含めて設定してください。

```
set COBCPY="%DB2PATH%\include\cobol_mf;%COBCPY%"
```

DB2 アプリケーション・プログラミング・インターフェースの呼び出しはすべて、呼び出し規則 74 を使用して行わなければなりません。DB2 COBOL プリコンパイラーは、自動的に `CALL-CONVENTION` 文節を `SPECIAL-NAMES` 段落に挿入します。 `SPECIAL-NAMES` 段落が存在しない場合、DB2 COBOL プリコンパイラーはそれを以下のように作成します。

```
Identification Division
Program-ID. "static".
special-names.
    call-convention 74 is DB2API.
```

さらにプリコンパイラーは、呼び出し規則を識別するために使用するシンボル `DB2API` を、DB2 API が呼び出されるたびに必ず `call` キーワードの後に自動的に置きます。これはたとえば、プリコンパイラーが DB2 API ランタイム呼び出しを組み込み SQL ステートメントから生成する場合にも必ず実行されます。

DB2 API への呼び出しをプリコンパイルされていないアプリケーションで実行する場合、前述したものと同様の `SPECIAL-NAMES` 段落を、手動で入力してアプリケーションに作成する必要があります。DB2 API を直接呼び出す場合は、`call` キーワードの後に `DB2API` シンボルを手動で追加する必要があります。

関連タスク:

- 323 ページの『Windows での Micro Focus COBOL アプリケーションの構築』
- 325 ページの『Windows での Micro Focus COBOL ルーチンの構築』

関連資料:

- 325 ページの『Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 327 ページの『Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』

Windows での Micro Focus COBOL アプリケーションの構築

DB2 には、DB2 API と組み込み SQL プログラムのコンパイルとリンクのためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqllib\samples\cobol_mf` ディレクトリーに置かれています。

バッチ・ファイル `bldapp.bat` には、DB2 アプリケーション・プログラムを構築するコマンドが入っています。これは最大 4 個のパラメーターを取り、それらはバッチ・ファイル内で変数 `%1`、`%2`、`%3`、および `%4` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。これは組み込み SQL を使用しないプログラムに必要な唯一のパラメーターです。組み込み SQL プログラムを構築するためにはデータベースへの接続が必要なため、3 つのパラメーターがオプションとして用意されています。2 番目のパラメーターは `%2` で、接続するデータベースの名前を指定します。3 番目のパラメーターは `%3` で、データベースのユーザー ID を指定します。そしてもう 1 つが `%4` で、データベースのパスワードを指定します。

組み込み SQL プログラムの場合、`bldapp` は、プリコンパイルおよびバインド・バッチ・ファイル `embprep.bat` にパラメーターを渡します。データベース名が指定されていない場合は、デフォルトの `sample` データベースが使用されます。なお、ユーザー ID とパスワードのパラメーターは、プログラムを構築するインスタンスとデータベースの置かれているインスタンスが異なる場合にのみ必要になります。

手順:

以下の例は、DB2 API と組み込み SQL のアプリケーションを構築して実行する方法を示しています。

ソース・ファイル `client.cbl` から組み込み SQL を含まないサンプル・プログラム `client` を構築するには、次のように入力します。

```
bldapp client
```

結果として、実行可能ファイル `client.exe` が作成されます。この実行可能ファイルを `sample` データベースに対して実行するには、次の実行可能名を (拡張子なしで) 入力します。

```
client
```

組み込み SQL アプリケーションの構築と実行

ソース・ファイル `updat.sqb` から組み込み SQL アプリケーション `updat` を構築する方法には、次の 3 つがあります。

1. 同じインスタンス上のサンプル・データベースに接続している場合には、次のように入力します。

```
bldapp updat
```

2. 同じインスタンスにある他のデータベースに接続している場合は、さらにデータベース名も入力します。

```
bldapp updat database
```

3. 他のインスタンスにあるデータベースに接続している場合は、さらにそのデータベース・インスタンスのユーザー ID とパスワードも入力します。

```
bldapp updat database userid password
```

結果として、実行可能ファイル `updat.exe` が作成されます。

この組み込み SQL アプリケーションを実行する方法には次の 3 つがあります。

1. 同じインスタンスにある `sample` データベースにアクセスする場合は、ただ実行可能ファイルの名前 (拡張子なし) を入力します。

```
updat
```

2. 同じインスタンスにある他のデータベースにアクセスする場合は、実行可能ファイル名とデータベース名を入力します。

```
updat database
```

3. 他のインスタンスにあるデータベースにアクセスする場合は、実行可能ファイル名、データベース名、およびそのデータベース・インスタンスのユーザー ID とパスワードを入力します。

```
updat database userid password
```

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 325 ページの『Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『bldapp.bat -- Builds Windows Micro Focus Cobol applications』
- 『client.cbl -- How to set and query a client (MF COBOL)』
- 『updat.sqb -- How to update, delete and insert table data (MF COBOL)』
- 『embprep.bat -- Prep and binds a C/C++ or Micro Focus COBOL embedded SQL program on Windows』

Micro Focus COBOL アプリケーションのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldapp.bat
rem Builds Windows Micro Focus Cobol applications
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

rem If an embedded SQL program, precompile and bind it.
if not exist "%1.sqb" goto compile_step
call embprep %1 %2 %3 %4

:compile_step
rem Compile the error-checking utility.
cobol checkerr.cbl;

rem Compile the program.
cobol %1.cbl;

rem Link the program.
cbllink -l %1.obj checkerr.obj db2api.lib
@echo on
```


Windows Micro Focus COBOL アプリケーションのコンパイルとリンクのオプション

以下は、bldapp.bat バッチ・ファイルに示されているように、Windows 上で Micro Focus COBOL コンパイラーを使用して、COBOL 組み込み SQL および DB2 API アプリケーションを構築するのにお勧めするコンパイルとリンクのオプションです。

bldapp のコンパイルとリンクのオプション	
コンパイル・オプション	
cobol	Micro Focus COBOL コンパイラー。
リンク・オプション	
cbllink	リンク・エディットにリンカーを使用します。
-l	lcobol ライブラリーとリンクします。
checkerr.obj	エラー・チェック・ユーティリティ・オブジェクト・ファイルとリンクします。
db2api.lib	DB2 API ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 323 ページの『Windows での Micro Focus COBOL アプリケーションの構築』

関連サンプル:

- 『bldapp.bat -- Builds Windows Micro Focus Cobol applications』

Windows での Micro Focus COBOL ルーチンの構築

DB2 には、DB2 API と組み込み SQL プログラムを Micro Focus COBOL でコンパイルおよびリンクするためのバッチ・ファイルが用意されていて、このファイルを使用して構築できるサンプル・プログラムと一緒に `sqlllib\samples\cobol_mf` ディレクトリーに置かれています。

バッチ・ファイル `bldrtn.bat` には、組み込み SQL ルーチン (ストアド・プロシージャ) を構築するためのコマンドが入っています。このバッチ・ファイルは、サーバー上の DLL 内でルーチンをコンパイルします。このバッチ・ファイルは 2 個のパラメーターをとります。それらは、バッチ・ファイル内では変数 `%1` と `%2` で表されます。

最初のパラメーター `%1` には、ソース・ファイルの名前を指定します。バッチ・ファイルでは、ソース・ファイル名 `%1` を DLL 名に使用します。第 2 パラメーター `%2` には、接続先のデータベースの名前を指定します。ストアド・プロシージャは、データベースが置かれているのと同じインスタンス上で構築する必要があるため、ユーザー ID やパスワードを指定するパラメーターはありません。

最初のパラメーター (ソース・ファイル名) だけが、必須です。データベース名は任意で指定します。データベース名を指定しない場合は、プログラムはデフォルトの `sample` データベースを使用します。

手順:

サンプル・データベースに接続している場合、ソース・ファイル `outsrv.sqb` からサンプル・プログラム `outsrv` を構築するには、次のように入力します。

```
bldrtn outsrv
```

他のデータベースに接続しているときは、さらにデータベース名も入力します。

```
bldrtn outsrv database
```

スクリプト・ファイルは、DLL をサーバー上の `sqllib/function` というパスにコピーします。

DLL `outsrv` の構築が完了したら、DLL 内のルーチン (DLL と同名) を呼び出すクライアント・アプリケーション `outcli` を構築できます。`outcli` は、`bldapp.bat` バッチ・ファイルを使用して構築することができます。

`outsrv` ルーチンを呼び出すには、次のように入力してサンプル・クライアント・アプリケーションを実行します。

```
outcli database userid password
```

ここで、

database

接続先のデータベースの名前です。名前は、`sample` かその別名、またはその他の名前にすることができます。

userid 有効なユーザー ID です。

password

ユーザー ID の有効なパスワードです。

クライアント・アプリケーションは DLL `outsrv` にアクセスします。これは、同一名のルーチンをサーバー・データベース上で実行します。出力は、クライアント・アプリケーションに戻されます。

関連概念:

- 112 ページの『ビルド・ファイル』

関連資料:

- 327 ページの『Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション』
- 86 ページの『COBOL のサンプル』

関連サンプル:

- 『`bldrtn.bat` -- Builds Windows Micro Focus Cobol routines (stored procedures)』
- 『`outcli.sqb` -- Call stored procedures using the SQLDA structure (MF COBOL)』
- 『`outsrv.sqb` -- Demonstrates stored procedures using the SQLDA structure (MF COBOL)』

- 『embprep.bat -- Prep and binds a C/C++ or Micro Focus COBOL embedded SQL program on Windows』

Micro Focus COBOL ルーチンのバッチ・ファイル

```
@echo off
rem BATCH FILE: bldrtn.bat
rem Builds Windows Micro Focus Cobol routines (stored procedures)
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2

rem Compile the stored procedure.
cobol %1.cbl /case;

rem Link the stored procedure and create a shared library.
cbllink /d %1.obj db2api.lib

rem Copy the stored procedure to the %DB2PATH%¥function directory.
copy %1.dll "%DB2PATH%¥function"
@echo on
```

Windows Micro Focus COBOL ルーチンのコンパイルとリンクのオプション

以下は、bldrtn.bat バッチ・ファイルに示されているように、 Windows 上で Micro Focus COBOL コンパイラーを使用して、 COBOL ルーチン (ストアド・プロシージャとユーザー定義関数) を構築するのにお勧めするコンパイルとリンクのオプションです。

bldrtn のコンパイルとリンクのオプション	
コンパイル・オプション	
cobol	Micro Focus COBOL コンパイラー。
/case	外部シンボルが大文字に変換されないようにします。
リンク・オプション	
cbllink	リンク・エディットのために Micro Focus COBOL リンカーを使用します。
/d	.DLL ファイルを作成します。
db2api.lib	DB2 API ライブラリーとリンクします。
他のコンパイラー・オプションについては、コンパイラーの資料をご覧ください。	

関連タスク:

- 325 ページの『Windows での Micro Focus COBOL ルーチンの構築』

関連サンプル:

- 『bldrtn.bat -- Builds Windows Micro Focus Cobol routines (stored procedures)』

オブジェクト REXX

Windows でのオブジェクト REXX アプリケーションの構築

オブジェクト REXX は、オブジェクト指向バージョンの REXX 言語です。オブジェクト指向の拡張子が従来の REXX に追加されていますが、既存の関数および命令には変更はありません。オブジェクト REXX インタープリターは、以下のサポートが加えられ、前のバージョンの拡張バージョンとなっています。

- クラス、オブジェクト、およびメソッド
- メッセージングおよびポリモアフィズム
- 単一および複数継承

オブジェクト REXX は、従来の REXX と完全な互換性があります。この節で REXX と述べる場合は、オブジェクト REXX を含むすべての REXX のバージョンのことを言います。

REXX プログラムはプリコンパイルまたはバインドしません。

Windows 上では、REXX プログラムはコメントで開始する必要はありません。しかし、移植性の理由から、第 1 行の第 1 列から始まるコメントで各 REXX プログラムを開始することをお勧めします。これによってプログラムを、他のプラットフォームのバッチ・コマンドと区別することができます。

```
/* Any comment will do. */
```

REXX サンプル・プログラムは、ディレクトリー `sqllib\samples\rex` にあります。

手順:

サンプル REXX プログラム `updat` を実行するには、次のように入力します。

```
rex x updat.cmd
```

関連概念:

- 「アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」の『REXX でのプログラミングに関する考慮事項』

関連資料:

- 102 ページの『REXX のサンプル』

第 4 部 付録

付録 A. DB2 Universal Database 技術情報

DB2 ドキュメンテーションおよびヘルプ

DB2 技術情報は、以下のツールや方法を使って利用できます。

- DB2 インフォメーション・センター
 - トピック
 - DB2 ツールのヘルプ
 - サンプル・プログラム
 - チュートリアル
- ダウンロード可能 PDF ファイル、CD に収録された PDF ファイル、および印刷文書
 - 各種「ガイド」
 - 各種「リファレンス・マニュアル」
- コマンド行ヘルプ
 - コマンド・ヘルプ
 - メッセージ・ヘルプ
 - SQL 状態ヘルプ
- インストール済みソース・コード
 - サンプル・プログラム

技術情報、白書、および Redbook など DB2 Universal Database の補足的な技術情報へは ibm.com でオンライン・アクセスできます。

www.ibm.com/software/data/pubs/ の「DB2 Information Management software library」にアクセスしてください。

DB2 ドキュメンテーションの更新

IBM では、DB2 インフォメーション・センターに対してドキュメンテーション・フィックスパックおよびその他のドキュメンテーションの更新を定期的に行い、利用可能にしています。DB2 インフォメーション・センター

(<http://publib.boulder.ibm.com/infocenter/db2help/>) にアクセスすると、常に最新の情報が表示されるようになっています。DB2 インフォメーション・センターをローカルにインストールしている場合には、手動で更新情報をインストールしてから表示する必要があります。ドキュメンテーションの更新により、新規情報が利用可能になると、「DB2 Information Center CD」からインストールした情報を更新できます。

インフォメーション・センターは、PDF やハードコピー資料より高い頻度で更新されます。最新の DB2 技術情報を得るために、利用可能になり次第ドキュメンテーションの更新をインストールするか、または www.ibm.com サイトから DB2 インフォメーション・センターにアクセスしてください。

関連概念:

- 「コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻」の『CLI サンプル・プログラム』
- 125 ページの『Java サンプル・プログラム』
- 332 ページの『DB2 インフォメーション・センター』

関連タスク:

- 353 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』
- 343 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 354 ページの『コマンド行プロセッサからメッセージ・ヘルプを呼び出す』
- 354 ページの『コマンド行プロセッサからコマンド・ヘルプを呼び出す』
- 355 ページの『コマンド行プロセッサから SQL 状態ヘルプを呼び出す』

関連資料:

- 345 ページの『DB2 PDF 資料および印刷された資料』

DB2 インフォメーション・センター

DB2[®] インフォメーション・センターを使用すると、DB2 Universal Database[™]、DB2 Connect[™]、DB2 Information Integrator および DB2 Query Patroller[™]などのDB2 ファミリー製品を最大限に活用するのに必要なすべての情報にアクセスできます。また、DB2 インフォメーション・センターは、DB2 の主な機能とコンポーネントに関する情報を提供します (レプリケーション、データウェアハウジング、および DB2 の種々の Extender など)。

Mozilla 1.0 以上または Microsoft[®] Internet Explorer 5.5 以上で表示する場合、DB2 インフォメーション・センターには以下の機能があります。以下のいくつかの機能では、JavaScript[™] のサポートを使用可能にする必要があります:

柔軟なインストール・オプション

以下の中から、ご使用の環境に最も適したオプションを使って DB2 資料を表示できます。

- 最新の資料を常に自動的に利用できるようにするには、IBM[®] の Web サイト (<http://publib.boulder.ibm.com/infocenter/db2help/>) にある DB2 インフォメーション・センターからすべての資料に直接アクセスします。
- 更新処理を最小化し、イントラネット内のネットワーク・トラフィックだけに制限するには、イントラネット上の 1 つのサーバーに DB2 資料をインストールします。
- 柔軟性を改善し、ネットワーク接続への依存を軽減するには、個々のコンピューターに DB2 資料をインストールします。

検索 「検索」テキスト・フィールドに検索語を入力することにより、DB2 インフォメーション・センターのすべてのトピックを検索できます。複数の語句を引用符で囲めば、完全一致を検索できます。また、ワイルドカード演算子 (*、?) とブール演算子 (AND、NOT、OR) を使用して検索を絞り込むことができます。

タスク指向の目次

単一の目次の中から、DB2 資料のトピックを見付けることができます。目

次は、主に実行するタスクの種類に従って編成されていますが、そのほかに製品概要、特定のゴール (目的) の情報、参照情報、索引、および用語集も含まれます。

- 製品概要では、DB2 ファミリーで使用可能な製品間の関係、そうした各製品で提供される機能、および各製品の最新リリース情報について説明されています。
- インストール、管理および開発などのゴール・カテゴリーには、タスクを迅速に完了し、そのための背景情報をよく理解できるようにするトピックが含まれています。
- 「参照」トピックでは、その対象に関する詳細な情報 (ステートメントとコマンドの構文、メッセージ・ヘルプ、構成パラメーターなど) が説明されています。

現在のトピックを目次に表示する

現在のトピックが目次のどの部分に該当するかを表示するには、目次フレーム内の「リフレッシュ/現在のトピックの表示 (Refresh/Show Current Topic)」ボタンをクリックするか、コンテンツ・フレーム内の「目次に表示 (Show in Table of Contents)」ボタンをクリックします。幾つかのファイルで関連トピックへの複数のリンクをたどった場合、または検索結果からトピックにアクセスした場合には、この機能が役立ちます。

索引 索引から、すべての資料にアクセスすることができます。索引では、用語が 50 音順に編成されています。

用語集 用語集を見れば、DB2 資料で使われているさまざまな用語の定義を調べることができます。用語集では、用語が 50 音順に編成されています。

組み込まれているローカライズ情報

DB2 インフォメーション・センターは、ブラウザで設定された言語でトピックを表示します。設定された言語のトピックが利用できない場合、DB2 インフォメーション・センターにはそのトピックの英語版が表示されます。

iSeries™ 技術情報については、IBM eServer™ iSeries Information Center (www.ibm.com/eserver/iseriess/infocenter/) を参照してください。

関連概念:

- 334 ページの『DB2 インフォメーション・センターのインストールのシナリオ』

関連タスク:

- 343 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 344 ページの『希望する言語での DB2 インフォメーション・センターのトピックの表示』
- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 336 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 339 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

DB2 インフォメーション・センターのインストールのシナリオ

作業環境が異なれば、DB2 情報にアクセスする方法の要件も異なる場合があります。DB2 インフォメーション・センターは、IBM Web サイトから、所属のネットワークのサーバーから、または自分のコンピューターにインストールしたものからアクセスできます。これらいずれの場合も、ドキュメンテーションは、ブラウザーで表示するトピック・ベースの情報が統合された Web 形式の DB2 インフォメーション・センターに組み込まれています。デフォルトでは、DB2 製品は IBM Web サイトから DB2 インフォメーション・センターにアクセスします。ただし、イントラネット・サーバーやご使用のコンピューターから DB2 インフォメーション・センターにアクセスする場合は、製品メディア・パックに同梱されている「DB2 インフォメーション・センター CD」を使用して DB2 インフォメーション・センターをインストールする必要があります。下記の DB2 ドキュメンテーションへのアクセスのオプションのサマリーおよび 3 つのインストールのシナリオを参照して、DB2 インフォメーション・センターへのアクセス方法として自分の環境に最適な方法はどれか、また考慮すべきインストールの問題について判断してください。

DB2 ドキュメンテーションへのアクセスのオプションのサマリー:

次の表は、DB2 インフォメーション・センターの DB2 製品ドキュメンテーションへのアクセスに関して、作業環境に最適な推奨オプションを示します。

インターネット・アクセス	イントラネット・アクセス	推奨
可	可	IBM Web サイトの DB2 インフォメーション・センターへのアクセス、またはイントラネット・サーバー上にインストールされている DB2 インフォメーション・センターへのアクセス。
可	不可	IBM Web サイトの DB2 インフォメーション・センターへのアクセス。
不可	可	イントラネット・サーバー上にインストールされている DB2 インフォメーション・センターへのアクセス。
不可	不可	ローカル・コンピューター上の DB2 インフォメーション・センターへのアクセス。

シナリオ: コンピューター上の DB2 インフォメーション・センターへのアクセス:

土屋さんは小さな町で工場を営んでいますが、その町にはインターネット・アクセスを行えるインターネット・サービス・プロバイダー (ISP) がありません。土屋さんは DB2 Universal Database を購入して、在庫、製品注文、銀行口座情報、および事業費を管理することにしました。これまで DB2 製品を一度も使ったことがない土屋さんは、DB2 製品の資料を見て使い方を学習する必要があります。

「標準」インストール・オプションで DB2 Universal Database をインストールし終えて、土屋さんは DB2 ドキュメンテーションにアクセスしようとしませんが、開こうとしたページが見つからないというエラー・メッセージがブラウザーから戻されます。インストールした DB2 製品のインストール・マニュアルを調べてみると、コンピューター上の DB2 ドキュメンテーションにアクセスするには、DB2 インフ

オメーション・センターをインストールしなければならないと書いてあります。そこで、土屋さんはメディア・バックから「DB2 Information Center CD」を見つけてインストールします。

インストールが完了すると、オペレーティング・システムのアプリケーション・ランチャーから DB2 インフォメーション・センターにアクセスできるようになりました。これで土屋さんは DB2 製品の使い方を学習することができ、事業の成功を後押しできそうです。

シナリオ: IBM Web サイトの DB2 インフォメーション・センターへのアクセス:

小折さんは教育促進企業の IT コンサルタントです。彼の専門はデータベース・テクノロジーと SQL で、DB2 Universal Database を使用する北米の企業でこのテーマのセミナーを開いています。小折さんのセミナーの課程では、DB2 ドキュメンテーションを教材として使用しています。たとえば、SQL の学習コースでは、データベース照会の基本構文と上級構文を教えるときに SQL に関する DB2 ドキュメンテーションを使ったりもしています。

小折さんがセミナーを開く企業には、大抵インターネット・アクセスの環境があります。そこで、最新版の DB2 Universal Database をインストールするのを機に、小折さんは自分のモバイル・コンピューターの構成を変更し、IBM Web サイトの DB2 インフォメーション・センターにアクセスするよう設定することにしました。このように構成したことで、小折さんはセミナーの際に、最新の DB2 ドキュメンテーションにオンライン・アクセスできるようになりました。

しかし、移動中にはインターネット・アクセスが使えないことがあります。特に、セミナーの準備をするために DB2 ドキュメンテーションにアクセスしたいときには困ったことになります。そうならないようにするため、小折さんはモバイル・コンピューターに DB2 インフォメーション・センターをインストールしました。

こうして、小折さんは好きなときにいつでも DB2 ドキュメンテーションを使える快適さを満喫しています。**db2set** コマンドを使うと、レジストリー変数をモバイル・コンピューター上で簡単に構成することができるので、小折さんは IBM Web サイトの DB2 インフォメーション・センターにアクセスするか、それともモバイル・コンピューター上を参照するかを、状況に応じて切り替えています。

シナリオ: イン트라ネット・サーバー上の DB2 インフォメーション・センターへのアクセス:

江原さんは生命保険会社で上級データベース管理者として働いています。江原さんの仕事には、会社の UNIX データベース・サーバーに最新版の DB2 Universal Database をインストールして構成することがあります。この会社では最近、セキュリティを考慮して、オフィスからはインターネット・アクセスをできなくすることが社員に通達されました。会社の環境はネットワークで結ばれているので、江原さんは DB2 インフォメーション・センターをイントラネット・サーバーにインストールし、会社のデータウェアハウスを定期的に使用する社員全員（営業担当者、営業部の管理職、および業務アナリスト）が DB2 ドキュメンテーションにアクセスできるようにすることにしました。

江原さんは、全社員のコンピューターに最新版の DB2 Universal Database を応答ファイルを使ってインストールし、イントラネット・サーバーのホスト名とポート番

号を使用して DB2 インフォメーション・センターにアクセスするように各コンピューターを構成するよう、自分のデータベース・チームに指示します。

しかし、江原さんの指示を誤解した同チーム下級データベース管理者の三田村さんは、社員のコンピューター数台に直接 DB2 インフォメーション・センターをインストールしてしまい、イントラネット・サーバーの DB2 インフォメーション・センターにアクセスするように DB2 Universal Database 構成を変更する作業は行いませんでした。この状態を直すために、江原さんは、構成を間違えたコンピューターで **db2set** コマンドを使って DB2 インフォメーション・センターのレジストリー変数を変更するよう、三田村さんに伝えます (ホスト名を DB2_DOCHOST、ポート番号を DB2_DOCPORT にする)。こうして、ネットワーク上のコンピューターはすべて DB2 インフォメーション・センターにアクセスできるようになり、社員は DB2 に関する質問の答えを DB2 ドキュメンテーションで調べられるようになりました。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』

関連タスク:

- 343 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 336 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 339 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』
- 『DB2 インフォメーション・センターへのアクセスのロケーションの設定: Common GUI help』

関連資料:

- 「コマンド・リファレンス」の『db2set - DB2 プロファイル・レジストリー・コマンド』

DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)

DB2 製品資料は、IBM Web サイトから、イントラネット・サーバーから、または自分のコンピューターにインストールしたものからアクセスできます。デフォルトでは、DB2 製品は IBM Web サイトから DB2 ドキュメンテーションにアクセスします。イントラネット・サーバーやご使用のコンピューターから DB2 ドキュメンテーションにアクセスする場合は、製品メディア・パックに同梱されている DB2 インフォメーション・センター CD からドキュメンテーションをインストールする必要があります。DB2 セットアップ・ウィザードを使用すると、インストール設定を定義して、UNIX オペレーティング・システムを使用するコンピューターに DB2 インフォメーション・センターをインストールできます。

前提条件:

この節では、UNIX コンピューターに DB2 インフォメーション・センターをインストールする際の、ハードウェア、オペレーティング・システム、ソフトウェア、および通信に関する要件をリストします。

• **ハードウェア要件**

次のいずれかのプロセッサが必要です。

- PowerPC (AIX)
- HP 9000 (HP-UX)
- Intel 32 ビット (Linux)
- Solaris UltraSPARC コンピューター (Solaris オペレーティング環境)

• **オペレーティング・システム要件**

次のいずれかのオペレーティング・システムが必要です。

- IBM AIX 5.1 (PowerPC で実行)
- HP-UX 11i (HP 9000 で実行)
- Red Hat Linux 8.0 (Intel 32 ビットで実行)
- SuSE Linux 8.1 (Intel 32 ビットで実行)
- Sun Solaris 8 (Solaris オペレーティング環境 UltraSPARC コンピューターで実行)

注: DB2 クライアントがサポートされているすべての種類の UNIX オペレーティング・システムで、DB2 インフォメーション・センターが公式にサポートされているわけではありません。そのため、DB2 インフォメーション・センターへは、IBM Web サイトからアクセスするか、またはインターネット・サーバー上に DB2 インフォメーション・センターをインストールしてアクセスすることをお勧めします。

• **ソフトウェア要件**

- 次のブラウザがサポートされています。

- Mozilla バージョン 1.0 以上

- DB2 セットアップ・ウィザードは、グラフィック・インストーラーです。ご使用のコンピューターで「DB2 セットアップ」ウィザードを実行するには、グラフィカル・ユーザー・インターフェースを表示できる X window ソフトウェアが必要です。DB2 セットアップ・ウィザードを実行する前に、ディスプレイを正しくエクスポートしたことを確認してください。たとえば、次のコマンドをコマンド・プロンプトで入力します。

```
export DISPLAY=9.26.163.144:0.
```

• **通信要件**

- TCP/IP

手順:

DB2 セットアップ・ウィザードを使用して DB2 インフォメーション・センターをインストールするには、次のようにします。

1. システムにログオンします。
2. DB2 インフォメーション・センター製品 CD をシステムに挿入してマウントします。

3. 次のコマンドを入力して、CD がマウントされているディレクトリーに移動します。

```
cd /cd
```

ここで、/cd は CD のマウント・ポイントを表します。

4. **/db2setup** コマンドを入力して、DB2 セットアップ・ウィザードを開始します。
5. 「IBM DB2 セットアップ・ランチパッド」がオープンします。DB2 インフォメーション・センターのインストールに直接進むには、「製品のインストール」をクリックします。残りのステップについて説明しているオンライン・ヘルプを利用できます。オンライン・ヘルプを呼び出すには、「ヘルプ」をクリックします。「キャンセル」をクリックすれば、いつでもインストールを終了できます。
6. 「インストールしたい製品を選択します」ページで、「次へ」をクリックします。
7. 「DB2 セットアップ・ウィザードへようこそ」ページで「次へ」をクリックします。DB2 セットアップ・ウィザードがプログラムのセットアップ操作を案内します。
8. インストールを進めるには、ご使用条件を受け入れる必要があります。「ご使用条件」ページで、「使用条件の条項に同意します。」を選択して「次へ」をクリックします。
9. 「インストール・アクションの選択」ページで、「DB2 インフォメーション・センターをこのコンピューターにインストールする」を選択します。後で同じまたは他のコンピューターに DB2 インフォメーション・センターをインストールするときに応答ファイルを使用する場合は、「設定を応答ファイルに保管する」を選択します。「次へ (Next)」をクリックします。
10. 「インストールする言語の選択」ページで、DB2 インフォメーション・センターをインストールする言語を選択します。「次へ (Next)」をクリックします。
11. 「DB2 インフォメーション・センター・ポートの指定」ページで、着信通信用に DB2 インフォメーション・センターを構成します。「次へ」をクリックして、インストールを続行します。
12. 「ファイルのコピーの開始」ページで、指定したインストール選択項目を確認します。設定を変更するには、「戻る」をクリックします。DB2 インフォメーション・センター・ファイルをコンピューターにコピーするには、「インストール」をクリックします。

応答ファイルを使用して、DB2 インフォメーション・センターをインストールすることもできます。

インストール・ログ db2setup.his、db2setup.log、および db2setup.err は、デフォルトでは /tmp ディレクトリーにあります。ログ・ファイルの位置は指定できます。

db2setup.log ファイルには、エラーを含むすべての DB2 製品インストール情報がキャプチャーされます。db2setup.his ファイルには、コンピューター上での DB2 製品インストールがすべて記録されます。DB2 は db2setup.log ファイルを

db2setup.his ファイルに付加します。 db2setup.err ファイルには、Java によって戻されたエラー出力 (たとえば、例外やトラップ情報) がキャプチャーされます。

インストールが完了すると、ご使用の UNIX オペレーティング・システムに応じて、DB2 インフォメーション・センターが以下のいずれかのディレクトリーに挿入されます。

- AIX: /usr/opt/db2_08_01
- HP-UX: /opt/IBM/db2/V8.1
- Linux: /opt/IBM/db2/V8.1
- Solaris オペレーティング環境: /opt/IBM/db2/V8.1

関連概念:

- 332 ページの『DB2 インフォメーション・センター』
- 334 ページの『DB2 インフォメーション・センターのインストールのシナリオ』

関連タスク:

- 「インストールおよび構成 補足」の『応答ファイルによる DB2 のインストール (UNIX)』
- 343 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 344 ページの『希望する言語での DB2 インフォメーション・センターのトピックの表示』
- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 339 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)

DB2 製品資料は、IBM Web サイトから、イントラネット・サーバーから、または自分のコンピューターにインストールしたものからアクセスできます。デフォルトでは、DB2 製品は IBM Web サイトから DB2 ドキュメンテーションにアクセスします。イントラネット・サーバーやご使用のコンピューターから DB2 ドキュメンテーションにアクセスする場合は、製品メディア・パックに同梱されている DB2 インフォメーション・センター CD から DB2 ドキュメンテーションをインストールする必要があります。DB2 セットアップ・ウィザードを使用すると、インストール設定を定義して、Windows オペレーティング・システムを使用するコンピューターに DB2 インフォメーション・センターをインストールできます。

前提条件:

この節では、Windows に DB2 インフォメーション・センターをインストールする際の、ハードウェア、オペレーティング・システム、ソフトウェア、および通信に関する要件をリストします。

• ハードウェア要件

次のいずれかのプロセッサが必要です。

- 32 ビット・コンピューター: Pentium または Pentium 互換 CPU

• オペレーティング・システム要件

次のいずれかのオペレーティング・システムが必要です。

- Windows 2000
- Windows XP

注: DB2 クライアントがサポートされているすべての種類の Windows オペレーティング・システムで、DB2 インフォメーション・センターが公式にサポートされているわけではありません。そのため、DB2 インフォメーション・センターへは、IBM Web サイトからアクセスするか、またはインターネット・サーバー上に DB2 インフォメーション・センターをインストールしてアクセスすることをお勧めします。

• ソフトウェア要件

- 以下のブラウザがサポートされています。
 - Mozilla 1.0 以上
 - Internet Explorer バージョン 5.5 または 6.0 (Windows XP の場合はバージョン 6.0)

• 通信要件

- TCP/IP

手順:

DB2 セットアップ・ウィザードを使用して DB2 インフォメーション・センターをインストールするには、次のようにします。

1. DB2 インフォメーション・センター・インストールのために定義したアカウントで、システムにログオンします。
2. CD をドライブに挿入します。自動実行機能が使用可能になっていれば、「IBM DB2 セットアップ」ランチパッドが自動的に起動します。
3. 「DB2 セットアップ」ウィザードはシステム言語を判別してから、その言語用のセットアップ・プログラムを立ち上げます。セットアップ・プログラムを英語以外の言語で実行したい場合や、セットアップ・プログラムが自動始動に失敗する場合には、DB2 セットアップ・ウィザードを使用して手動で開始することができます。

次のようにして、DB2 セットアップ・ウィザードを手動で開始します。

- a. 「スタート」をクリックし、「ファイル名を指定して実行」を選択します。
- b. 「開く」フィールドで、次のコマンドを入力します。

```
x:%setup language
```

x: は CD ドライブを表し、*language* はセットアップ・プログラムを実行する言語を表します。

- c. 「OK」をクリックします。
4. 「IBM DB2 セットアップ・ランチパッド」がオープンします。DB2 インフォメーション・センターのインストールに直接進むには、「製品のインストール」をクリックします。残りのステップについて説明しているオンライン・ヘル

ルプを利用できます。オンライン・ヘルプを呼び出すには、「ヘルプ」をクリックします。「キャンセル」をクリックすれば、いつでもインストールを終了できます。

5. 「インストールしたい製品を選択します」ページで、「次へ」をクリックします。
6. 「DB2 セットアップ・ウィザードへようこそ」ページで「次へ」をクリックします。DB2 セットアップ・ウィザードがプログラムのセットアップ操作を案内します。
7. インストールを進めるには、ご使用条件を受け入れる必要があります。「ご使用条件」ページで、「使用条件の条項に同意します。」を選択して「次へ」をクリックします。
8. 「インストール・アクションの選択」ページで、「DB2 インフォメーション・センターをこのコンピューターにインストールする」を選択します。後で同じまたは他のコンピューターに DB2 インフォメーション・センターをインストールするときに応答ファイルを使用する場合は、「設定を応答ファイルに保管する」を選択します。「次へ (Next)」をクリックします。
9. 「インストールする言語の選択」ページで、DB2 インフォメーション・センターをインストールする言語を選択します。「次へ (Next)」をクリックします。
10. 「DB2 インフォメーション・センター・ポートの指定」ページで、着信通信用に DB2 インフォメーション・センターを構成します。「次へ」をクリックして、インストールを続行します。
11. 「ファイルのコピーの開始」ページで、指定したインストール選択項目を確認します。設定を変更するには、「戻る」をクリックします。DB2 インフォメーション・センター・ファイルをコンピューターにコピーするには、「インストール」をクリックします。

応答ファイルを使用して、DB2 インフォメーション・センターをインストールすることができます。db2rspgn コマンドを使用して、既存のインストールに基づいた応答ファイルを生成することもできます。

インストール時に検出されるエラーの詳細については、'My Documents'¥DB2LOG¥ディレクトリーにある db2.log と db2wi.log ファイルを参照してください。My Documents ディレクトリーのロケーションは、ご使用のコンピューターの設定によって異なります。

db2wi.log ファイルには、最新の DB2 インストール情報がキャプチャーされます。db2.log には、DB2 製品インストールの履歴がキャプチャーされます。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』
- 334 ページの『DB2 インフォメーション・センターのインストールのシナリオ』

関連タスク:

- 「インストールおよび構成 補足」の『応答ファイルによる DB2 製品のインストール (Windows)』

- 343 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 344 ページの『希望する言語での DB2 インフォメーション・センターのトピックの表示』
- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 336 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』

関連資料:

- 「コマンド・リファレンス」の『db2rspgn - 応答ファイル生成プログラム・コマンド』

DB2 インフォメーション・センターの呼び出し

DB2 インフォメーション・センターは、Linux、UNIX、および Windows オペレーティング・システム用の DB2 製品 (DB2 Universal Database、DB2 Connect、DB2 Information Integrator、DB2 Query Patroller など) を使用するために必要なすべての情報を提供します。

DB2 インフォメーション・センターは、以下の場所から呼び出すことができます。

- DB2 UDB クライアントまたはサーバーがインストールされているコンピューター
- DB2 インフォメーション・センターがインストールされているイントラネット・サーバーまたはローカル・コンピューター
- IBM の Web サイト

前提条件:

DB2 インフォメーション・センターを呼び出すための要件は、以下のとおりです。

- オプション: 希望する言語でトピックを表示するようブラウザを構成する
- オプション: コンピューターまたはイントラネット・サーバーにインストール済みの DB2 インフォメーション・センターを使用するよう DB2 クライアントを構成する

手順:

DB2 UDB クライアントまたはサーバーがインストールされているコンピューターから DB2 インフォメーション・センターを呼び出すには、以下のようになります。

- (Windows オペレーティング・システムの)「スタート」メニューから:「スタート」→「プログラム」→「IBM DB2」→「情報」→「インフォメーション・センター」をクリックします。
- コマンド行プロンプトから:
 - Linux および UNIX オペレーティング・システムの場合、**db2icdocs** コマンドを発行します。
 - Windows オペレーティング・システムの場合、**db2icdocs.exe** コマンドを発行します。

イントラネット・サーバーまたはローカル・コンピュータにインストール済みの DB2 インフォメーション・センターを Web ブラウザーで開くには、以下のようにします。

- Web ページ `http://<host-name>:<port-number>/` を開きます (<host-name> はホスト名、<port-number> は DB2 インフォメーション・センターを利用可能なポート番号)。

IBM Web サイトにある DB2 インフォメーション・センターを Web ブラウザーで開くには、以下のようにします。

- Web ページ `publib.boulder.ibm.com/infocenter/db2help/` を開きます。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』
- 334 ページの『DB2 インフォメーション・センターのインストールのシナリオ』

関連タスク:

- 344 ページの『希望する言語での DB2 インフォメーション・センターのトピックの表示』
- 353 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』
- 343 ページの『コンピュータまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 354 ページの『コマンド行プロセッサからコマンド・ヘルプを呼び出す』
- 『DB2 インフォメーション・センターへのアクセスのロケーションの設定: Common GUI help』

関連資料:

- 「コマンド・リファレンス」の『HELP コマンド』

コンピュータまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール

`http://publib.boulder.ibm.com/infocenter/db2help/` から利用できる DB2 インフォメーション・センターは、資料の新規追加または変更によって定期的に更新されます。さらに、更新された DB2 インフォメーション・センターをコンピュータまたはイントラネット・サーバーにダウンロードしてインストールできる場合もあります。DB2 インフォメーション・センターを更新しても、DB2 クライアント製品またはサーバー製品は更新されません。

前提条件:

インターネットに接続されたコンピュータへのアクセスが必要です。

手順:

DB2 インフォメーション・センターの更新をコンピュータまたはイントラネット・サーバーにインストールするには、以下のようにします。

1. IBM の Web サイト (`http://publib.boulder.ibm.com/infocenter/db2help/`) にある DB2 インフォメーション・センターを開きます。

2. 「DB2 インフォメーション・センターによるこそ」ページの見出し「サービスおよびサポート」の「ダウンロード」セクションで、「**DB2 資料**」リンクをクリックします。
3. 最新のドキュメンテーション・イメージのレベルと、インストール済みのドキュメンテーション・レベルを比較して、DB2 インフォメーション・センターを更新する必要があるかどうかを確認します。「DB2 インフォメーション・センターによるこそ」ページに、インストール済みのドキュメンテーションのレベルがリストされます。
4. より新しいバージョンの DB2 インフォメーション・センターが存在する場合、ご使用のオペレーティング・システムに対応する最新の DB2 インフォメーション・センター・イメージをダウンロードします。
5. 最新の DB2 インフォメーション・センター・イメージをインストールするには、Web ページの指示に従ってください。

関連概念:

- 334 ページの『DB2 インフォメーション・センターのインストールのシナリオ』

関連タスク:

- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 336 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 339 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

希望する言語での DB2 インフォメーション・センターのトピックの表示

DB2 インフォメーション・センターは、ブラウザーの設定に指定されている言語でトピックを表示しようとします。トピックが希望する言語に翻訳されていない場合は、DB2 インフォメーション・センターはそのトピックを英語で表示します。

手順:

Internet Explorer ブラウザーを使ってトピックを希望する言語で表示するには、以下のようになります。

1. Internet Explorer で、「ツール」 → 「インターネット オプション」 → 「言語...」 ボタンをクリックします。「言語の優先順位」ウィンドウがオープンします。
2. 希望する言語が言語リストの最初の項目に指定されていることを確認してください。
 - リストに言語を新たに追加するには、「追加...」 ボタンをクリックします。

注: 言語を追加したとしても、希望する言語でトピックを表示するために必要なフォントがコンピューターに入っていることを保証するものではありません。

- リストの上部に言語を移動するには、言語を選択し、その言語がリストの先頭に来るまで「上へ」 ボタンをクリックします。

3. ページを最新表示して、DB2 インフォメーション・センターを希望する言語で表示します。

Mozilla ブラウザーを使ってトピックを希望する言語で表示するには、以下のようになります。

1. Mozilla で、「編集」 → 「設定...」 → 「言語」 ボタンを選択します。「設定」ウィンドウに「言語」パネルが表示されます。
2. 希望する言語が言語リストの最初の項目に指定されていることを確認してください。
 - リストに言語を新たに追加するには、「追加...」ボタンをクリックして表示される「言語を追加」ウィンドウから言語を選択します。
 - リストの上部に言語を移動するには、言語を選択し、その言語がリストの先頭に来るまで「上へ」ボタンをクリックします。
3. ページを最新表示して、DB2 インフォメーション・センターを希望する言語で表示します。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』

DB2 PDF 資料および印刷された資料

以下の表は、正式な資料名、資料番号、および PDF ファイル名を示しています。ハードコピー版の資料を注文するには、正式な資料名を知っておく必要があります。PDF ファイルを印刷するには、PDF ファイル名を知っておく必要があります。

DB2 資料は、以下のカテゴリーに分類されています。

- DB2 中核情報
- 管理情報
- アプリケーション開発情報
- ビジネス・インテリジェンス情報
- DB2 Connect 情報
- 入門情報
- チュートリアル情報
- オプション・コンポーネント情報
- リリース・ノート

以下の表は、DB2 ライブラリー内の各資料について、その資料のハードコピー版を注文したり、PDF 版を印刷または表示したりするのに必要な情報を示しています。DB2 ライブラリー内の各資料に関する詳細な説明については、www.ibm.com/shop/publications/order にある IBM Publications Center にアクセスしてください。

DB2 の基本情報

こうした資料の情報は、すべての DB2 ユーザーに基本的なもので、プログラマーおよびデータベース管理者にとって役立つ情報であるとともに、DB2 Connect、

DB2 Warehouse Manager、または他の DB2 製品を使用するユーザーにとっても役立つ内容です。

表 42. DB2 の基本情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Universal Database コマンド・リファレンス」	SC88-9140	db2n0j81
「IBM DB2 Universal Database 用語集」	資料番号なし	db2t0j81
「IBM DB2 Universal Database メッセージ・リファレンス 第 1 巻」	GC88-9152 (ハードコピーな し)	db2m1j81
「IBM DB2 Universal Database メッセージ・リファレンス 第 2 巻」	GC88-9153 (ハードコピーな し)	db2m2j81
「IBM DB2 Universal Database 新機能」	SC88-9158	db2q0j81

管理情報

これらの資料の情報は、DB2 データベース、データウェアハウス、およびフェデレーテッド・システムを効果的に設計し、インプリメントし、保守するために必要なトピックを扱っています。

表 43. 管理情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Universal Database 管理ガイド: プランニング」	SC88-9135	db2d1j81
「IBM DB2 Universal Database 管理ガイド: インプリメンテー ション」	SC88-9133	db2d2j81
「IBM DB2 Universal Database 管理ガイド: パフォーマンス」	SC88-9134	db2d3j81
「IBM DB2 Universal Database 管理 API リファレンス」	SC88-9136	db2b0j81
「IBM DB2 Universal Database データ移動ユーティリティー ガイドおよびリファレンス」	SC88-9142	db2dmj81
「IBM DB2 Universal Database データ・リカバリーと高可用性 ガイドおよびリファレンス」	SC88-9143	db2haj81
「IBM DB2 Universal Database データウェアハウス・センター 管理ガイド」	SC88-9165	db2ddj81
「IBM DB2 Universal Database SQL リファレンス 第 1 巻」	SC88-9155	db2s1j81
「IBM DB2 Universal Database SQL リファレンス 第 2 巻」	SC88-9156	db2s2j81

表 43. 管理情報 (続き)

資料名	資料番号	PDF ファイル名
「IBM DB2 Universal Database システム・モニター ガイドおよびリファレンス」	SC88-9157	db2f0j81

アプリケーション開発情報

これらの資料の情報は、DB2 Universal Database (DB2 UDB) のアプリケーション開発者またはプログラマーが特に興味を持つ内容です。サポートされるさまざまなプログラミング・インターフェース (組み込み SQL、ODBC、JDBC、SQLJ、CLI など) を使用して DB2 UDB にアクセスするのに必要な資料とともに、サポートされる言語およびコンパイラーについても紹介されています。また、DB2 インフォメーション・センターをご使用の場合には、サンプル・プログラムのソース・コードの HTML バージョンにアクセスすることもできます。

表 44. アプリケーション開発情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Universal Database アプリケーション開発ガイド アプリケーションの構築および実行」	SC88-9137	db2axj81
「IBM DB2 Universal Database アプリケーション開発ガイド クライアント・アプリケーションのプログラミング」	SC88-9138	db2a1j81
「IBM DB2 Universal Database アプリケーション開発ガイド サーバー・アプリケーションのプログラミング」	SC88-9139	db2a2j81
「IBM DB2 Universal Database コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻」	SC88-9159	db2l1j81
「IBM DB2 Universal Database コール・レベル・インターフェース ガイドおよびリファレンス 第 2 巻」	SC88-9160	db2l2j81
「IBM DB2 Universal Database データウェアハウス・センター アプリケーション統合ガイド」	SC88-9166	db2adj81
「IBM DB2 Universal Database XML Extender 管理およびプログラミングのガイド」	SC88-9172	db2sxj81

ビジネス・インテリジェンス情報

これらの資料の情報は、さまざまなコンポーネントを使用して、DB2 Universal Database のデータウェアハウジング機能および分析機能を拡張する方法を説明しています。

表 45. ビジネス・インテリジェンス情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Warehouse Manager Standard Edition インフォメーション・カタログ・センター 管理ガイド」	SC88-9167	db2dij81
「IBM DB2 Warehouse Manager Standard Edition インストール・ガイド」	GC88-9164	db2idj81
「IBM DB2 Warehouse Manager Standard Edition DB2 Warehouse Manager を使用時の ETI ソリューション・コンバージョン・プログラムの管理」	SC88-9894	iwhe1mstx80

DB2 Connect 情報

このカテゴリの情報は、DB2 Connect Enterprise Edition または DB2 Connect Personal Edition を使用して、メインフレーム・サーバーおよびミッドレンジ・サーバー上のデータにアクセスする方法を説明しています。

表 46. DB2 Connect 情報

資料名	資料番号	PDF ファイル名
「IBM コネクティビティ 補足」	資料番号なし	db2h1j81
「IBM DB2 Connect Enterprise Edition 概説およびインストール」	GC88-9145	db2c6j81
「IBM DB2 Connect Personal Edition 概説およびインストール」	GC88-9146	db2c1j81
「IBM DB2 Connect ユーザーズ・ガイド」	SC88-9147	db2c0j81

入門情報

このカテゴリの情報は、サーバー、クライアント、および他の DB2 製品をインストールして構成する場合に役立ちます。

表 47. 入門情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Universal Database DB2 クライアント機能 概説およびインストール」	GC88-9144 (ハードコピーなし)	db2itj81
「IBM DB2 Universal Database DB2 サーバー機能 概説およびインストール」	GC88-9148	db2isj81
「IBM DB2 Universal Database DB2 Personal Edition 概説およびインストール」	GC88-9150	db2ilj81
「IBM DB2 Universal Database インストールおよび構成 補足」	GC88-9149 (ハードコピーなし)	db2iyj81
「IBM DB2 Universal Database DB2 Data Links Manager 概説およびインストール」	GC88-9141	db2z6j81

チュートリアル情報

チュートリアル情報は、DB2 機能を紹介し、さまざまなタスクを実行する方法を示します。

表 48. チュートリアル情報

資料名	資料番号	PDF ファイル名
「ビジネス・インテリジェンス・チュートリアル: データウェアハウス・センターの紹介」	資料番号なし	db2tuj81
「ビジネス・インテリジェンス・チュートリアル: データウェアハウジングの上級者向けガイド」	資料番号なし	db2taj81
「インフォメーション・カタログ・センター チュートリアル」	資料番号なし	db2aij81
「Video Central for e-business チュートリアル」	資料番号なし	db2twj81
「Visual Explain チュートリアル」	資料番号なし	db2tvj81

オプション・コンポーネント情報

このカテゴリーの情報は、DB2 のオプション・コンポーネントを使用する方法について説明しています。

表 49. オptional・コンポーネント情報

資料名	資料番号	PDF ファイル名
「IBM DB2 Cube Views Guide and Reference」	SC18-7298	db2aax81
「IBM DB2 Query Patroller インストール、管理、使用法のガイド」	GC88-9154	db2dwj81
「IBM DB2 Spatial Extender and Geodetic Extender ユーザーズ・ガイドおよびリファレンス」	SC88-9171	db2sbj81
「IBM DB2 Universal Database Data Links Manager 管理ガイドおよびリファレンス」	SC88-9169	db2z0x82
「DB2 Net Search Extender 管理およびユーザーズ・ガイド」	SH88-8546	N/A

注: この資料の HTML 版は、HTML ドキュメンテーション CD からインストールされません。

リリース・ノート

リリース・ノートは、ご使用の製品のリリースおよびフィックスパック・レベルに特有の追加情報を紹介します。また、リリース・ノートには、各リリース、アップデート、およびフィックスパックで組み込まれた資料上の更新の要約も含まれています。

表 50. リリース・ノート

資料名	資料番号	PDF ファイル名
「DB2 リリース・ノート」	「注」を参照。	「注」を参照。
「DB2 インストール情報」	製品 CD-ROM でのみ参照可能。	使用できません。

注: リリース・ノートは以下の形式で入手できます。

- XHTML およびテキスト形式 (製品 CD 内)
- PDF 形式 (PDF ドキュメンテーション CD 内)

さらに、リリース・ノートの中で、『既知の問題と予備手段』および『リリース間の非互換性』に関する部分は DB2 インフォメーション・センターにも表示されます。

UNIX ベースのプラットフォームでテキスト形式でリリース・ノートを確認するには、Release.Notes ファイルを参照してください。このファイルは、DB2DIR/Readme/%L ディレクトリーに収録されています。%L はロケール名を表しています。DB2DIR は以下になります。

- AIX オペレーティング・システムの場合: /usr/opt/db2_08_01
- その他のすべての UNIX ベースのオペレーティング・システムの場合:
/opt/IBM/db2/V8.1

関連概念:

- 331 ページの『DB2 ドキュメンテーションおよびヘルプ』

関連タスク:

- 351 ページの『PDF ファイルからの DB2 資料の印刷方法』
- 352 ページの『DB2 の印刷資料の注文方法』
- 353 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』

PDF ファイルからの DB2 資料の印刷方法

DB2 PDF ドキュメンテーション CD に収録されている DB2 資料を印刷することができます。Adobe Acrobat Reader を使用すれば、資料全体または特定のページを印刷できます。

前提条件:

Adobe Acrobat Reader がインストールされていることを確認してください。Adobe Acrobat Reader をインストールする必要がある場合、Adobe Web サイト (www.adobe.com) から入手できます。

手順:

PDF ファイルから DB2 資料を印刷するには以下のようにします。

1. *DB2 PDF* ドキュメンテーション CD をドライブに挿入します。UNIX オペレーティング・システムの場合、*DB2 PDF* ドキュメンテーション CD をマウントします。UNIX オペレーティング・システムで CD をマウントする方法については、「概説およびインストール」を参照してください。
2. index.htm を開きます。ブラウザ・ウィンドウにファイルが開きます。
3. 参照したい PDF のタイトルをクリックします。Acrobat Reader で PDF が開きます。
4. 「ファイル」→「印刷」を選択して、所要の資料の任意の部分を印刷します。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』

関連タスク:

- 「*DB2 Universal Database* サーバー機能 概説およびインストール」の『CD-ROM のマウント (AIX)』
- 「*DB2 Universal Database* サーバー機能 概説およびインストール」の『HP-UX 上での CD-ROM のマウント』
- 「*DB2 Universal Database* サーバー機能 概説およびインストール」の『CD-ROM のマウント (Linux)』
- 352 ページの『DB2 の印刷資料の注文方法』

- 「DB2 Universal Database サーバー機能 概説およびインストール」の『CD-ROM のマウント (Solaris)』

関連資料:

- 345 ページの『DB2 PDF 資料および印刷された資料』

DB2 の印刷資料の注文方法

ハードコピー版の資料を望む場合には、以下のいずれかの方法で注文できます。

印刷資料の注文方法:

一部の国または地域では、印刷された資料を注文することもできます。お客様がお住まいの国または地域でこのサービスが利用可能かどうかを確認するには、お住まいの国または地域の IBM Publications Web サイトをご覧ください。資料のご注文が可能な場合、以下のようにすることができます。

- 正規の IBM 製品販売業者または営業担当員に連絡してください。お客様がお住まいの地域の IBM 担当員の情報については、お手数ですが IBM の Web サイト (www.ibm.com/planetwide) の IBM Worldwide Directory of Contacts で確認してください。
- IBM Publications Center (<http://www.ibm.com/shop/publications/order>) にアクセスしてください。なお、IBM Publications Center から資料を注文できない国もあります。

DB2 製品がご利用可能になった時点で、印刷された資料は DB2 PDF ドキュメンテーション CD にある PDF 形式の資料と同じものです。さらに、DB2 インフォメーション・センター CD に収録されている印刷された資料の内容もまた、これらと同じです。ただし、DB2 インフォメーション・センター CD には、PDF 資料にならない追加情報も含まれます (たとえば、SQL 管理作業や HTML サンプル)。DB2 PDF ドキュメンテーション CD に収録されている資料の中には、ハードコピーとしてご注文できない資料もあります。

注: DB2 インフォメーション・センターは、PDF またはハードコピーの資料よりも頻繁に更新されます。ドキュメンテーションの更新が入手可能になった時点でインストールするか、DB2 インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/db2help/>) を参照して最新の情報を入手してください。

関連タスク:

- 351 ページの『PDF ファイルからの DB2 資料の印刷方法』

関連資料:

- 345 ページの『DB2 PDF 資料および印刷された資料』

DB2 ツールからコンテキスト・ヘルプを呼び出す

コンテキスト・ヘルプは、特定のウィンドウ、ノートブック、ウィザード、またはアドバイザに関連したタスクまたはコントロールの情報を提供します。コンテキスト・ヘルプは、グラフィカル・ユーザー・インターフェースのある DB2 管理ツールおよび開発ツールから利用できます。コンテキスト・ヘルプには、以下の 2 種類があります。

- それぞれのウィンドウまたはノートブックにある「ヘルプ」ボタンからアクセス可能なヘルプ
- infopop (ポップアップ情報ウィンドウ)。これは、マウス・カーソルを特定のフィールドまたはコントロール上に置いたとき、またはウィンドウ、ノートブック、ウィザード、アドバイザ内でフィールドまたはコントロールを選択して F1 を押すと表示されます。

「ヘルプ」ボタンを押すと、概説、前提条件、およびタスク情報が表示されます。infopop は、それぞれのフィールドおよびコントロールについて説明します。

手順:

コンテキスト・ヘルプを呼び出すには、以下のようになります。

- ウィンドウおよびノートブックのヘルプを表示するには、いずれかの DB2 ツールを開始して、任意のウィンドウまたはノートブックを開きます。ウィンドウまたはノートブックの右下隅にある「ヘルプ」ボタンをクリックして、コンテキスト・ヘルプを呼び出します。

また、それぞれの DB2 ツール・センターの上部にある「ヘルプ」メニュー項目からコンテキスト・ヘルプにアクセスすることもできます。

ウィザードおよびアドバイザでは、最初のページの「タスクの概要」リンクをクリックすると、コンテキスト・ヘルプを表示できます。

- ウィンドウまたはノートブック上の各コントロールの infopop ヘルプを表示するには、コントロールをクリックしてから、**F1** を押します。コントロールの詳細情報を示すポップアップ情報が、黄色いウィンドウに表示されます。

注: フィールドまたはコントロールにマウス・カーソルを置いておくだけで infopops が表示されるようにするには、「ツール設定」ノートブックの「**文書 (Documentation)**」ページの「**infopops の自動表示**」チェック・ボックスを選択します。

infopop に似た別のコンテキスト・ヘルプに、診断ポップアップ情報があります。これにはデータ入力規則が示されます。診断ポップアップ情報は、無効または不十分なデータが入力されたとき、紫色のウィンドウに表示されます。診断ポップアップ情報は、以下に関して表示されます。

- 必須フィールド。
- 日付フィールドのように、正確なフォーマットを必要とするデータのフィールド。

関連タスク:

- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 354 ページの『コマンド行プロセッサからメッセージ・ヘルプを呼び出す』

- 354 ページの『コマンド行プロセッサからコマンド・ヘルプを呼び出す』
- 355 ページの『コマンド行プロセッサから SQL 状態ヘルプを呼び出す』
- 『DB2 インフォメーション・センターへのアクセス: Concepts help』
- 『DB2 UDB ヘルプの使用法: Common GUI help』
- 『DB2 インフォメーション・センターへのアクセスのロケーションの設定: Common GUI help』
- 『DB2 コンテキスト・ヘルプと資料へのアクセスを設定する: Common GUI help』

コマンド行プロセッサからメッセージ・ヘルプを呼び出す

メッセージ・ヘルプは、メッセージが出された原因と、エラーへの応答として実行すべきアクションを説明します。

手順:

メッセージ・ヘルプを呼び出すには、コマンド行プロセッサを開いて以下のように入力します。

`? XXXnnnnnn`

ここで、`XXXnnnnnn` は有効なメッセージ ID を表します。

たとえば、`? SQL30081` と入力すると、メッセージ `SQL30081` に関するヘルプを表示します。

関連概念:

- 「メッセージ・リファレンス 第 1 巻」の『メッセージの概要』

関連資料:

- 「コマンド・リファレンス」の『db2 - コマンド行プロセッサの呼び出しコマンド』

コマンド行プロセッサからコマンド・ヘルプを呼び出す

コマンド・ヘルプは、コマンド行プロセッサでのコマンドの構文を説明します。

手順:

コマンド・ヘルプを呼び出すには、コマンド行プロセッサを開いて以下のように入力します。

`? command`

ここで `command` はキーワードまたはコマンド全体を表します。

たとえば、`? catalog` と入力すると、すべての `CATALOG` コマンドに関するヘルプが表示され、`? catalog database` と入力すると、`CATALOG DATABASE` コマンドのヘルプだけが表示されます。

関連タスク:

- 353 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』
- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 354 ページの『コマンド行プロセッサからメッセージ・ヘルプを呼び出す』
- 355 ページの『コマンド行プロセッサから SQL 状態ヘルプを呼び出す』

関連資料:

- 「コマンド・リファレンス」の『db2 - コマンド行プロセッサの呼び出しコマンド』

コマンド行プロセッサから SQL 状態ヘルプを呼び出す

DB2 Universal Database は、SQL ステートメントの結果の原因となったと考えられる条件の SQLSTATE 値を戻します。SQLSTATE ヘルプは、SQL 状態および SQL 状態クラス・コードの意味を説明します。

手順:

SQL 状態ヘルプを呼び出すには、コマンド行プロセッサを開いて以下のように入力します。

```
? sqlstate または ? class code
```

ここで、*sqlstate* は有効な 5 桁の SQL 状態を、*class code* は SQL 状態の最初の 2 桁を表します。

たとえば、? 08003 を指定すると SQL 状態 08003 のヘルプが表示され、? 08 を指定するとクラス・コード 08 のヘルプが表示されます。

関連タスク:

- 342 ページの『DB2 インフォメーション・センターの呼び出し』
- 354 ページの『コマンド行プロセッサからメッセージ・ヘルプを呼び出す』
- 354 ページの『コマンド行プロセッサからコマンド・ヘルプを呼び出す』

DB2 チュートリアル

DB2® チュートリアルは、DB2 Universal Database のさまざまな機能について学習するのを支援します。このチュートリアルでは、アプリケーションの開発、SQL 照会のパフォーマンス調整、データウェアハウスの処理、メタデータの管理、および DB2 を使用した Web サービスの開発の各分野で、段階的なレッスンが用意されています。

はじめに:

インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/db2help/>) から、このチュートリアルの XHTML 版を表示できます。

チュートリアルの中で、サンプル・データまたはサンプル・コードを使用する場合があります。個々のタスクの前提条件については、それぞれのチュートリアルを参照してください。

DB2 Universal Database チュートリアル:

以下に示すチュートリアルのタイトルをクリックすると、そのチュートリアルを表示できます。

ビジネス・インテリジェンス・チュートリアル: データウェアハウス・センターの紹介
データウェアハウス・センターを使用して簡単なデータウェアハウジング・タスクを実行します。

ビジネス・インテリジェンス・チュートリアル: データウェアハウジングの上級者向けガイド
データウェアハウス・センターを使用して高度なデータウェアハウジング・タスクを実行します。

インフォメーション・カタログ・センター・チュートリアル
インフォメーション・カタログを作成および管理して、インフォメーション・カタログ・センターを使用してメタデータを配置し使用します。

Visual Explain チュートリアル
Visual Explain を使用して、パフォーマンスを向上させるために SQL ステートメントを分析し、最適化し、調整します。

DB2 トラブルシューティング情報

DB2[®] 製品を使用する際に役立つ、トラブルシューティングおよび問題判別に関する広範囲な情報を利用できます。

DB2 ドキュメンテーション

トラブルシューティング情報は、DB2 インフォメーション・センター、および DB2 ライブラリーに含まれる PDF 資料の中でご利用いただけます。DB2 インフォメーション・センターで、(ブラウザー・ウィンドウの左側の) ナビゲーション・ツリーの「サポートおよびトラブルシューティング (Support and troubleshooting)」ブランチを参照すると、DB2 トラブルシューティング・ドキュメンテーションの詳細なリストが見つかります。

DB2 Technical Support の Web サイト

現在問題が発生していて、考えられる原因とソリューションを検索したい場合は、DB2 Technical Support の Web サイトを参照してください。

Technical Support サイトには、最新の DB2 出版物、TechNotes、プログラム診断依頼書 (APAR)、フィックスパック、DB2 内部エラー・コードの最新リスト、その他のリソースが用意されています。この知識ベースを活用して、問題に対する有効なソリューションを探し出すことができます。

DB2 Technical Support の Web サイト

(<http://www.ibm.com/software/data/db2/udb/win02unix/support>) にアクセスしてください。

DB2 Problem Determination Tutorial Series

DB2 製品で作業中に直面するかもしれない問題を素早く識別し、解決する方法に関する情報を見つけるには、DB2 Problem Determination Tutorial Series の Web サイトを参照してください。あるチュートリアルでは、使用可能な DB2 問題判別機能およびツールを紹介し、それらをいつ使用すべきかを判断する助けを与えます。別のチュートリアルは、『データベース・エ

ンジン問題判別 (Database Engine Problem Determination)』、『パフォーマンス問題判別 (Performance Problem Determination)』、『アプリケーション問題判別 (Application Problem Determination)』などの関連トピックを扱っています。

DB2 Technical Support

(<http://www.ibm.com/software/data/support/pdm/db2tutorials.html>) には、DB2 問題判別チュートリアルがすべて揃っています。

関連概念:

- 332 ページの『DB2 インフォメーション・センター』
- 「問題判別の手引き」の『Introduction to Problem Determination - DB2 テクニカル・サポートのチュートリアル』

アクセス支援

アクセス支援機能は、身体に障害のある（身体動作が制限されている、視力が弱いなど）ユーザーがソフトウェア製品を十分活用できるように支援します。DB2[®] バージョン 8 製品に備わっている主なアクセス支援機能は、以下のとおりです。

- すべての DB2 機能は、マウスの代わりにキーボードを使ってナビゲーションできます。詳細については、『キーボードによる入力およびナビゲーション』を参照してください。
- DB2 ユーザー・インターフェースのフォント・サイズおよび色をカスタマイズすることができます。詳細については、358 ページの『アクセスしやすい表示』を参照してください。
- DB2 製品は、Java[™] Accessibility API を使用するアクセス支援アプリケーションをサポートします。詳細については、358 ページの『支援テクノロジーとの互換性』を参照してください。
- DB2 資料は、アクセスしやすい形式で提供されています。詳細については、358 ページの『アクセスしやすい資料』を参照してください。

キーボードによる入力およびナビゲーション

キーボード入力

キーボードだけを使用して DB2 ツールを操作できます。マウスを使って実行できる操作は、キーまたはキーの組み合わせによっても実行できます。標準のオペレーティング・システム・キー・ストロークを使用して、標準のオペレーティング・システム操作を実行できます。

キーまたはキーの組み合わせによって操作を実行する方法について、詳しくは キーボード・ショートカットおよびアクセラレーター: Common GUI help を参照してください。

キーボード・ナビゲーション

キーまたはキーの組み合わせを使用して、DB2 ツールのユーザー・インターフェースをナビゲートできます。

キーまたはキーの組み合わせによって DB2 ツールをナビゲートする方法の詳細については、キーボード・ショートカットおよびアクセラレーター: Common GUI help を参照してください。

キーボード・フォーカス

UNIX® オペレーティング・システムでは、アクティブ・ウィンドウの中で、キー・ストロークによって操作できる領域が強調表示されます。

アクセスしやすい表示

DB2 ツールには、視力の弱いユーザー、その他の視力障害をもつユーザーのためにアクセシビリティを向上させる機能が備わっています。これらのアクセシビリティ拡張機能には、フォント・プロパティのカスタマイズを可能にする機能も含まれています。

フォントの設定

「ツール設定」ノートブックを使用して、メニューおよびダイアログ・ウィンドウに使用されるテキストの色、サイズ、およびフォントを選択できます。

フォント設定に関する詳細情報は、メニューおよびテキストのフォントを変更する: Common GUI help を参照してください。

色に依存しない

本製品のすべての機能を使用するために、ユーザーは必ずしも色を識別する必要はありません。

支援テクノロジーとの互換性

DB2 ツールのインターフェースは、Java Accessibility API をサポートします。これによって、スクリーン・リーダーその他の支援テクノロジーを DB2 製品で利用できるようになります。

アクセスしやすい資料

DB2 形式は、ほとんどの Web ブラウザーで表示可能な XHTML 1.0 形式で提供されています。XHTML により、ご使用のブラウザーに設定されている表示設定に従って資料を表示できます。さらに、スクリーン・リーダーや他の支援テクノロジーを使用することもできます。

シンタックス・ダイアグラムはドット 10 進形式で提供されます。この形式は、スクリーン・リーダーを使用してオンライン・ドキュメンテーションにアクセスする場合にのみ使用できます。

関連概念:

- 359 ページの『ドット 10 進シンタックス・ダイアグラム』

関連タスク:

- 『キーボード・ショートカットおよびアクセラレーター: Common GUI help』
- 『メニューおよびテキストのフォントを変更する: Common GUI help』

ドット 10 進シンタックス・ダイアグラム

スクリーン・リーダーを使用してインフォメーション・センターを利用するユーザーのために、シンタックス・ダイアグラムがドット 10 進形式で提供されます。

ドット 10 進形式では、各シンタックス・エレメントは別々の行に書き込まれます。複数のシンタックス・エレメントが常に同時に存在する (または常に同時に不在の) 場合、単一のコンパウンド・シンタックス・エレメントとみなせるので同一行に表示できます。

各行は、ドット 10 進数で開始します。たとえば、3 または 3.1 ないしは 3.1.1 です。こうした数を適切に聞き取るには、スクリーン・リーダーが句読点を読み取るように設定されていることを確認してください。同じドット 10 進数を持つすべてのシンタックス・エレメント (たとえば、3.1 という数値を持つすべてのシンタックス・エレメント) は、相互に排他的な代替エレメントです。3.1 USERID および 3.1 SYSTEMID という行を聞き取る場合、シンタックスには両方ではなく USERID または SYSTEMID のどちらかが含まれることが分かります。

ドット 10 進レベルは、ネストのレベルを表示します。たとえば、ドット 10 進数 3 のシンタックス・エレメントの後に、一連のドット 10 進数 3.1 のシンタックス・エレメントが続きます。3.1 の番号が付されたシンタックス・エレメントすべては、番号 3 の付されたシンタックス・エレメントに従属します。

シンタックス・エレメントに関する情報を追加するため、ドット 10 進数の次に特定のワードおよびシンボルが使用されます。時折、こうしたワードおよびシンボルはエレメントの最初に表示される場合もあります。簡単に識別するため、ワードやシンボルがシンタックス・エレメントの一部である場合には、円記号 (¥) 文字が先頭に付きます。* シンボルはドット 10 進数の次に使用でき、シンタックス・エレメントが反復することを示します。たとえば、ドット 10 進数 3 のシンタックス・エレメント *FILE は、3 ¥* FILE という形式になります。3* FILE という形式は、シンタックス・エレメント FILE が反復されることを示します。3* ¥* FILE という形式は、シンタックス・エレメント * FILE が反復されることを示します。

シンタックス・エレメントのストリングを分離するのに使用されるコンマなどの文字は、シンタックス内の分離する項目の直前に表示されます。こうした文字は、それぞれの項目と同一行に表示するか、同じドット 10 進数を持つ関連する項目のある別の行に表示できます。またその行には、シンタックス・エレメントに関する情報を提供する別のシンボルを表示することも可能です。たとえば、複数の LASTRUN および DELETE シンタックス・エレメントを使用している場合には、5.1*、5.1 LASTRUN、および 5.1 DELETE という行は、エレメントをコンマで区切る必要があります。区切り文字が指定されないと、各シンタックス・エレメントを区切るのにブランクが使用されると想定されます。

シンタックス・エレメントの前に % シンボルが付く場合、他の箇所で定義されている参照であることを示します。% シンボルの後のストリングは、リテラルではなくシンタックス・フラグメントの名前です。たとえば、2.1 %OP1 という行は別のシンタックス・フラグメント OP1 を参照すべきことを意味します。

以下のワードおよびシンボルが、ドット 10 進数の次に使用されます。

- ? は、オプションのシンタックス・エレメントであることを表します。? シンボルが後に続くドット 10 進数は、対応するドット 10 進数のシンタックス・エレメント、および任意の従属のシンタックス・エレメントがオプションであることを示します。ドット 10 進数の付いたシンタックス・エレメントが 1 つしかない場合、? シンボルはそのシンタックス・エレメントと同じ行に表示されます (たとえば、5? NOTIFY)。ドット 10 進数の付いたシンタックス・エレメントが複数ある場合、? シンボルだけで行に表示され、その後にオプションのシンタックス・エレメントが続きます。たとえば、「5 ?, 5 NOTIFY、および 5 UPDATE」という行を聞き取る場合、シンタックス・エレメント NOTIFY および UPDATE がオプションである、つまりそのいずれかを選択でき、どちらも選択しないこともできることが分かります。? シンボルは、線路型ダイアグラムのバイパス線に相当します。
- ! は、デフォルトのシンタックス・エレメントであることを表します。! シンボルおよびシンタックス・エレメントが後に続くドット 10 進数は、そのシンタックス・エレメントが、同じドット 10 進数を共用するシンタックス・エレメントすべてのデフォルト・オプションであることを示します。同じドット 10 進数を共用するシンタックス・エレメントのうち 1 つだけに、! シンボルを指定できます。たとえば、「2? FILE、2.1! (KEEP)、および 2.1 (DELETE)」という行を聞き取る場合、FILE キーワードのデフォルト・オプションは (KEEP) になります。この例では、FILE キーワードを含めてもオプションを指定しない場合には、デフォルト・オプション KEEP が適用されます。デフォルト・オプションは、次に高位のドット 10 進数にも適用されます。この例の場合、FILE キーワードが省略されると、デフォルトの FILE(KEEP) が使用されます。しかし、「2? FILE、2.1、2.1.1! (KEEP)、および 2.1.1 (DELETE)」という行を聞き取る場合、デフォルト・オプション KEEP は次に高位のドット 10 進数 2.1 (関連キーワードを持っていない) にのみ適用され、2? FILE には適用されません。キーワード FILE が省略されると、どれも使用されません。
- * は、0 回以上反復できるシンタックス・エレメントを示します。* シンボルが後に続くドット 10 進数は、このシンタックス・エレメントが 0 回以上使用できること、つまりオプションであり、なおかつ反復できることを表します。たとえば、5.1* データ域という行を聞き取る場合、1 つまたは複数のデータ域を含めるか、またはデータ域を全く含めないことが可能です。「3*, 3 HOST、および 3 STATE」という行を聞き取る場合、HOST、STATE をどちらか一方または両方同時に含めるか、どちらも含めないことができます。

注:

1. ドット 10 進数の後にアスタリスク (*) が付き、ドット 10 進数の付いた項目が 1 つしかない場合には、同じ項目を複数回反復できます。
 2. ドット 10 進数の後にアスタリスクが付き、ドット 10 進数の付いた項目が複数ある場合、リストから複数の項目を使用できますが、各項目を複数回使用することはできません。前述の例では、HOST STATE と書くことはできますが、HOST HOST とは書けません。
 3. * シンボルは、線路型シンタックス・ダイアグラムのループバック線に相当します。
- + は、1 回以上含める必要のあるシンタックス・エレメントであることを示します。+ シンボルが後に続くドット 10 進数は、このシンタックス・エレメントを 1 回以上含める必要があること、つまり少なくとも 1 回は含める必要があり、反

復できることを表します。たとえば、「6.1+ データ域」という行を聞き取る場合、データ域を少なくとも 1 回は含めなければなりません。「2+, 2 HOST、および 2 STATE」という行を聞き取る場合には、HOST、STATE、またはその両方を含める必要があります。* シンボルと同様に、+ シンボルは、ドット 10 進数の付いた項目が 1 つしかない場合に限り、その特定の項目のみを反復できます。* シンボルと同様、+ シンボルは線路型シンタックス・ダイアグラムのループバック線に相当します。

関連概念:

- 357 ページの『アクセス支援』

関連タスク:

- 『キーボード・ショートカットおよびアクセラレーター: Common GUI help』

関連資料:

- 「*SQL* リファレンス 第 2 巻」の『構文図の見方』

DB2 Universal Database 製品の共通基準認証

DB2 Universal Database は、Common Criteria の評価検定レベル 4 (EAL4) で認証の評価を受けています。Common Criteria の詳細については、以下の Common Criteria の Web サイトを参照してください。 <http://niap.nist.gov/cc-scheme/>

付録 B. 特記事項

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。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービス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。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。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を提供し、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 Canada Limited

Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

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せんが、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。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

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証できません。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をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

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。これらの名称はすべて架空のものであり、名称や住所が類似する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。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _年を入れる_. All rights reserved.

商標

以下は、IBM Corporation の商標です。

ACF/VTAM	iSeries
AISPO	LAN Distance
AIX	MVS
AIXwindows	MVS/ESA
AnyNet	MVS/XA
APPN	Net.Data
AS/400	NetView
BookManager	OS/390
C Set++	OS/400
C/370	PowerPC
CICS	pSeries
Database 2	QBIC
DataHub	QMF
DataJoiner	RACF
DataPropagator	RISC System/6000
DataRefresher	RS/6000
DB2	S/370
DB2 Connect	SP
DB2 Extenders	SQL/400
DB2 OLAP Server	SQL/DS
DB2 Information Integrator	System/370
DB2 Query Patroller	System/390
DB2 Universal Database	SystemView
Distributed Relational Database Architecture	Tivoli
DRDA	VisualAge
eServer	VM/ESA
Extended Services	VSE/ESA
FFST	VTAM
First Failure Support Technology	WebExplorer
IBM	WebSphere
IMS	WIN-OS/2
IMS/ESA	z/OS
	zSeries

以下は、それぞれ各社の商標または登録商標です。

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。

Pentium は、Intel Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. 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。

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。
他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

【ア行】

アクセシビリティ
機能 357
ドット 10 進構文図 359
アプリケーション開発
移行
2 つのバージョンの DB2 での実行 64
環境のセットアップ 27
サンプル・データベースのセットアップ 52
Java 環境のセットアップ 31
Perl
構築, アプリケーションの 161
PHP
構築, アプリケーションの 165
UNIX 環境のセットアップ 36
Windows 環境のセットアップ 45
アプレット
使用するためのポイント 127
JDBC の構築 129
JDBC のサンプル 91
SQLJ の構築 135
SQLJ のサンプル 94
移行
アプリケーション 57
2 つのバージョンの DB2 での実行 64
32 ビットから 64 ビット環境へ 60
アプリケーションの移植性 63
Java アプリケーション 59
Java アプレット 59
Java ルーチン 59
移植性
アプリケーションの移行における 63
印刷
PDF ファイル 351
印刷資料, 注文 352
インスタンス
データベース・マネージャー 5
インストール
インフォメーション・センター 334, 336, 339

インフォメーション・センター
インストール 334, 336, 339
エラー・チェック
ユーティリティ・ファイル 119
オブジェクトのリンクと埋め込み
オートメーション
Visual Basic での 284
Visual C++ での 300
サンプル 99
データベース関数
サンプル・ファイル 100
説明 275
DB2 AD Client サポート 3
オペレーティング・システム
サポートされる AIX バージョン 10
サポートされる HP-UX バージョン 12
サポートされる Linux バージョン 14
サポートされる Solaris バージョン 20
サポートされる Windows バージョン 23
DB2 インストール・パス 57
DB2 でサポートされる 9
オンライン
ヘルプへのアクセス 353

【カ行】

カタログ
サンプル・データベース 55
環境
アプリケーション開発
セットアップ 27
環境変数
UNIX 37
キーボード・ショートカット
サポート対象 357
起動
コマンド・ヘルプ 354
メッセージ・ヘルプ 354
SQL ステートメント・ヘルプ 355
共用ライブラリー
ルーチンの再構築 29
AIX の置換 193
組み込み SQL
AIX C++ 構成ファイル 206
DB2 AD Client サポート 3
コール・レベル・インターフェース (CLI)
DB2 AD Client サポート 3

更新
HTML 資料 343
構成ファイル
AIX 上の VisualAge C++ 204
コマンド行プロセッサ (CLP)
サンプル・ファイル 85
スクリプトの実行 149
DB2 AD Client サポート 3
コマンド・ヘルプ
起動 354
コンパイラー
サポートされる AIX バージョン 10
サポートされる HP-UX バージョン 12
サポートされる Linux バージョン 14
サポートされる Solaris バージョン 20
サポートされる Windows バージョン 23
ビルド・ファイル 112
AIX IBM COBOL の使用 209
AIX Micro Focus COBOL の使用 217
HP-UX Micro Focus COBOL の使用 236
makefile 116
Solaris Micro Focus COBOL の使用 268
Windows IBM COBOL の使用 313
Windows Micro Focus COBOL の使用 322

【サ行】

サーバー
DB2 でサポートされる 8
再バインド
SQL プロシージャ 158
サンプル
オブジェクトのリンクと埋め込み 99
データベース関数 100
コマンド行プロセッサ (CLP) 85
サポートされる言語 69
動的再構成 90
プログラム
Java サンプル・ディレクトリー 125
プログラム・ファイル 69
ログ管理ユーザ出口 97
C 75
COBOL 86
C++ 79

サンプル (続き)

Java WebSphere 96

Java プラグイン 96

JDBC 91

Perl 101

PHP 102

SQLJ 94

サンプル・データベース

カタログ 55

作成 52

セットアップ 52

バインディング 55

ホスト・システム上での作成 54

資料

表示 342

身体障害 357

スクリプト

コマンド行プロセッサ (CLP) の実行
149

ストアド・プロシージャ

AIX C++ 構成ファイル 207

CALL ステートメント 150

Visual Basic での OLE オートメーシ
ョン 284

Visual C++ での OLE オートメーショ
ン 300

セキュリティ

サンプル 105

疎結合トランザクション

Visual Basic

トラブルシューティング 281

Windows 上での構築 279

Visual Basic のサンプル 108

[タ行]

チュートリアル 355

トラブルシューティングおよび問題判
別 356

注文、DB2 資料 352

データベース・マネージャ
ーインスタンス 5

デベロップメント・センター

DB2 AD Client サポート 3

動的構成

サンプル 90

ドット 10 進構文図 359

トラブルシューティング

オンライン情報 356

チュートリアル 356

[ハ行]

バインディング

SQL プロシージャ 155

バインディング・ユーティリティ

サンプル・データベース 55

バックアップ

SQL プロシージャ 157

バッチ・ファイル 112

表関数

オブジェクトのリンクと埋め込み

サンプル 100

OLE DB 275

ビルド・ファイル 112

ファイル拡張子

サンプル 69

複数接続アプリケーション

ビルド・ファイル 112

UNIX C の構築 173

UNIX C++ の構築 181

Windows C/C++ の構築 310

プラグイン

セキュリティ・サンプル 105

Java サンプル 96

プリコンパイラ

DB2 AD Client サポート 3

プリコンパイル

SQL プロシージャ 155

プログラム

サンプル 69

ヘルプ

コマンド用

起動 354

表示 342, 344

メッセージ用

起動 354

SQL ステートメント用

起動 355

ホスト・システム

サポートされるサーバー 8

サンプル・データベースの作成 54

[マ行]

マルチスレッド・アプリケーション

ビルド・ファイル 112

AIX C を使用した構築 198

AIX C++ を使用した構築 203

HP-UX C を使用した構築 229

HP-UX C++ を使用した構築 235

Linux C を使用した構築 246

Linux C++ を使用した構築 251

Solaris C を使用した構築 261

Solaris C++ を使用した構築 267

Windows C/C++ を使用した構築 301

メッセージ・ヘルプ

起動 354

問題判別

オンライン情報 356

チュートリアル 356

[ヤ行]

ユーザー定義関数 (UDF)

AIX C++ 構成ファイル 208

Visual Basic での OLE オートメーシ
ョン 284

Visual C++ での OLE オートメーショ
ン 300

ユーザー出口プログラム

サンプル・ファイル 97

[ラ行]

ライブラリー、共用

ルーチンの再構築 29

AIX の置換 193

リストア

SQL プロシージャ 157

リモート・データ・オブジェクト

Visual Basic のサンプル 108

Visual Basic を使用した構築 283

ルーチン

共用ライブラリーの再構築 29

サンプル・プログラム・ファイル

SQL プロシージャ 105

ビルド・ファイル 112

AIX 入り口点 191

AIX 上での COBOL 共用ライブラリ
ーのロード 193

AIX 上の CREATE ステートメント
192

ルーチンの入り口点、AIX 191

ログ管理

ユーザー出口のサンプル・ファイル
97

[数字]

32 ビット・アプリケーション

64 ビット環境への移行 60

A

ActiveX データ・オブジェクト

DB2 AD Client サポート 3

Visual Basic のサンプル 108

Visual Basic を使用した構築 276

Visual C++ のサンプル 111

Visual C++ を使用した構築 298

AIX

C アプリケーション

コンパイルおよびリンク・オブショ
ン 195

C マルチスレッド・アプリケーション
構築 198

AIX (続き)

- C ルーチン
 - コンパイルおよびリンク・オプション 197
- C++ API アプリケーション
 - 構成ファイルを使用した構築 205
- C++ アプリケーション
 - コンパイルおよびリンク・オプション 200
- C++ 組み込み SQL
 - 構成ファイルを使用した構築 206
- C++ ストアード・プロシージャ
 - 構成ファイルを使用した構築 207
- C++ ユーザー定義関数
 - 構成ファイルを使用した構築 208
- C++ ルーチン
 - コンパイルおよびリンク・オプション 202
- IBM COBOL アプリケーション
 - 構築 210
 - コンパイルおよびリンク・オプション 212
- IBM COBOL ルーチン
 - 構築 213
 - コンパイルおよびリンク・オプション 215
- Java
 - 環境のセットアップ 40
- Micro Focus COBOL アプリケーション
 - コンパイルおよびリンク・オプション 218
- Micro Focus COBOL ルーチン
 - コンパイルおよびリンク・オプション 220
- REXX アプリケーション
 - 構築 221

API

- AIX C++ 構成ファイル 205

C

C

- アプリケーション
 - AIX 上でのコンパイル・オプション 195
 - HP-UX 上でのコンパイル・オプション 224
 - Linux 上でのコンパイル・オプション 242
 - Solaris 上でのコンパイル・オプション 258
 - UNIX 上での構築 171
 - Windows 上での構築 301
 - Windows でのコンパイル・オプション 304

C (続き)

- エラー・チェック・ユーティリティ
ー・ファイル 119
- サポートされる AIX バージョン 10
- サポートされる HP-UX バージョン 12
- サポートされる Linux バージョン 14
- サポートされる Solaris バージョン 20
- サポートされる Windows バージョン 23
- サンプル 75
- ビルド・ファイル 112
- 複数接続アプリケーション
 - UNIX 上での構築 173
 - Windows 上での構築 310
- マルチスレッド・アプリケーション
 - AIX 198
 - HP-UX 229
 - Linux 246
 - Solaris 261
 - Windows 301
- ルーチン
 - AIX 上でのコンパイル・オプション 197
 - HP-UX 上でのコンパイル・オプション 227
 - Linux 上でのコンパイル・オプション 244
 - Solaris 上でのコンパイル・オプション 260
 - UNIX 上での構築 175
 - Windows 上での構築 305
 - Windows でのコンパイル・オプション 309
- makefile 116
- CALL ステートメント
 - コマンド行プロセッサ 150
- CLI (コール・レベル・インターフェース)
 - サンプル・プログラム・ファイル 83
- CLR (common language runtime)
 - ルーチン
 - コンパイルおよびリンク・オプション 297
 - Windows 上での構築 293
- COBOL 言語
 - エラー・チェック・ユーティリティ
ー・ファイル 119
 - サポートされる AIX バージョン 10
 - サポートされる HP-UX バージョン 12
 - サポートされる Linux バージョン 14
 - サポートされる Solaris バージョン 20
 - サポートされる Windows バージョン 23

COBOL 言語 (続き)

- サンプル 86
- ビルド・ファイル 112
- AIX
 - AIX 上でのインストールおよび実行 193
 - IBM コンパイラー 209
 - Micro Focus コンパイラー 217
- HP-UX
 - Micro Focus コンパイラーの使用 236
- IBM COBOL アプリケーション
 - AIX 上での構築 210
 - AIX 上でのコンパイル・オプション 212
 - Windows 上での構築 314
 - Windows でのコンパイル・オプション 317
- IBM COBOL ルーチン
 - AIX 上での構築 213
 - AIX 上でのコンパイル・オプション 215
 - Windows 上での構築 318
 - Windows でのコンパイル・オプション 320
- Linux
 - Micro Focus コンパイラー 252
- makefile 116
- Micro Focus アプリケーション
 - AIX 上でのコンパイル・オプション 218
 - HP-UX 上でのコンパイル・オプション 238
 - Linux 上でのコンパイル・オプション 254
 - Solaris 上でのコンパイル・オプション 269
 - UNIX 上での構築 187
 - Windows 上での構築 323
 - Windows でのコンパイル・オプション 325
- Micro Focus ルーチン
 - AIX 上でのコンパイル・オプション 220
 - HP-UX 上でのコンパイル・オプション 239
 - Linux 上でのコンパイル・オプション 255
 - Solaris 上でのコンパイル・オプション 270
 - UNIX 上での構築 189
 - Windows 上での構築 325
 - Windows でのコンパイル・オプション 327
- Solaris オペレーティング環境
 - Micro Focus コンパイラー 268

COBOL 言語 (続き)

Windows

IBM コンパイラー 313

Micro Focus コンパイラー 322

CREATE PROCEDURE ステートメント

SQL プロシージャでの 153

CREATE ステートメント

AIX ルーチン 192

C# .NET

アプリケーション

コンパイルおよびリンク・オプション 287

Windows 上での構築 285

サポートされる Windows バージョン 23

サンプル 82

バッチ・ファイル 112

C++

アプリケーション

AIX 上でのコンパイル・オプション 200

HP-UX 上でのコンパイル・オプション 231

Linux 上でのコンパイル・オプション 247

Solaris 上でのコンパイル・オプション 263

UNIX 上での構築 179

Windows 上での構築 301

Windows でのコンパイル・オプション 304

エラー・チェック・ユーティリティ
ー・ファイル 119

サポートされる AIX バージョン 10

サポートされる HP-UX バージョン 12

サポートされる Linux バージョン 14

サポートされる Solaris バージョン 20

サポートされる Windows バージョン 23

サンプル 79

ビルド・ファイル 112

複数接続アプリケーション

UNIX 上での構築 181

Windows 上での構築 310

マルチスレッド・アプリケーション

AIX 203

HP-UX 235

Linux 251

Solaris オペレーティング環境 267

Windows 301

ルーチン

AIX 上でのコンパイル・オプション 202

C++ (続き)

ルーチン (続き)

HP-UX 上でのコンパイル・オプション 233

Linux 上でのコンパイル・オプション 249

Solaris 上でのコンパイル・オプション 265

UNIX 上での構築 183

Windows 上での構築 305

Windows でのコンパイル・オプション 309

AIX 上の VisualAge 構成ファイル 204

makefile 116

Visual C++ での OLE オートメーション 300

D

DB2 CLI

サンプル・プログラム・ファイル 83

DB2 Personal Developer's Edition ix

DB2 Universal Developer's Edition ix

DB2 インフォメーション・センター 332

起動 342

DB2 資料

PDF ファイルの印刷 351

DB2 チュートリアル 355

DB2INSTANCE 環境変数 52

DB2INSTPROF

およびデータベース・マネージャ 5

DB2PATH

およびデータベース・マネージャ 5

E

EXTERNAL NAME 文節

CREATE ステートメント 192

F

FORTTRAN 言語

DB2 サポート 9

H

HP-UX

C アプリケーション

コンパイルおよびリンク・オプション 224

C マルチスレッド・アプリケーション構築 229

HP-UX (続き)

C ルーチン

コンパイルおよびリンク・オプション 227

C++ アプリケーション

コンパイルおよびリンク・オプション 231

C++ ルーチン

コンパイルおよびリンク・オプション 233

Java

環境のセットアップ 41

Micro Focus COBOL アプリケーション

コンパイルおよびリンク・オプション 238

Micro Focus COBOL ルーチン

コンパイルおよびリンク・オプション 239

HTML 資料

更新 343

J

Java

アプリケーションの移行 59

アプレット、使用するためのポイント 127

環境のセットアップ 31

JDBC アプリケーション 130

JDBC アプレット 129

SQLJ アプリケーション 134, 137

SQLJ アプレット 135

サポートされる AIX JDK 10

サポートされる HP-UX JDK 12

サポートされる Linux JDK 14

サポートされる Solaris JDK 20

サンプル

ディレクトリー 125

プラグイン・サンプル・ファイル 96

AIX 環境のセットアップ 40

DB2 AD Client サポート 3

HP-UX 環境のセットアップ 41

JDBC のサンプル 91

JDBC ルーチンの構築 131

Linux

環境のセットアップ 43

makefile 116

Solaris オペレーティング環境
セットアップ 44

SQLJ のサンプル 94

SQLJ ルーチンの構築 142

UNIX 環境のセットアップ 38

WebSphere サンプル・ファイル 96

Java (続き)

Windows

サポートされる JDK バージョン
23

セットアップ 49

JDBC (Java database connectivity)

アプレット、使用するためのポイント
127

アプレットの構築 129

構築、アプリケーションの 130

サンプル 91

ルーチンの構築 131

DB2 AD Client サポート 3

JDK_PATH、データベース・マネージャー

構成キーワード 30

K

KEEPFENCED データベース・マネージャ
ー構成キーワード 30

Kerberos

セキュリティ・プロトコル

サンプル 105

L

Linux

C アプリケーション

コンパイルおよびリンク・オプション 242

C マルチスレッド・アプリケーション
構築 246

C ルーチン

コンパイルおよびリンク・オプション 244

C++ アプリケーション

コンパイルおよびリンク・オプション 247

C++ ルーチン

コンパイルおよびリンク・オプション 249

Java

環境のセットアップ 43

Micro Focus COBOL

コンパイラの構成 252

Micro Focus COBOL アプリケーション
ン

コンパイルおよびリンク・オプション 254

Micro Focus COBOL ルーチン

コンパイルおよびリンク・オプション 255

M

makefile 116

Management

Instrumentation、Windows 275

サンプル 112

Microsoft Transaction Server

Visual Basic のサンプル 108

MQ ユーザー定義関数

セットアップ 33

N

NOCONVERT オプション 274

O

OLE DB プロバイダー

Visual Basic での 276

Visual C++ での 298

P

Perl

構築、アプリケーションの 161

サンプル 101

DB2 サポート 9

PHP

構築、アプリケーションの 165

サンプル 102

DB2 サポート 9

R

REXX 言語

サポートされる AIX バージョン 10

サポートされる Windows バージョン
23

サンプル 102

AIX アプリケーションの構築 221

DB2 サポート 9

Windows アプリケーションの構築
328

S

Solaris オペレーティング環境

アプリケーション

C コンパイルおよびリンク・オプ
ション 258

C++ コンパイルおよびリンク・オ
プション 263

ルーチン

C コンパイルおよびリンク・オプ
ション 260

Solaris オペレーティング環境 (続き)

ルーチン (続き)

C++ コンパイルおよびリンク・オ
プション 265

C マルチスレッド・アプリケーション
構築 261

Java セットアップ 44

Micro Focus COBOL アプリケーショ
ン

コンパイルおよびリンク・オプション 269

Micro Focus COBOL ルーチン

コンパイルおよびリンク・オプション 270

SQL 92 および MVS 適合 flagger

DB2 AD Client サポート 3

SQL ステートメント・ヘルプ

起動 355

SQL プロシージャー

クライアント・アプリケーション 154

再バインド 158

作成 153

サンプル・プログラム・ファイル 105

バックアップおよびリストア 157

プリコンパイルおよび BIND オプシ
ョン 155

CALL ステートメント 150

SQL プロシージャーの呼び出し

クライアント・アプリケーション 154

SQLJ

アプリケーション

UNIX でのコンパイル・オプション
139

Windows でのコンパイル・オプション 142

ルーチン

UNIX でのコンパイル・オプション
145

Windows でのコンパイル・オプション 148

SQLJ (Java 用組み込み SQL)

アプリケーション

構築 137

アプレット

構築 135

アプレット。使用するためのポイント
127

サンプル 94

ビルド・ファイル 112

プログラム

構築 134

ルーチンの構築 142

DB2 AD Client サポート 3

U

UNIX

- アプリケーション開発
- 環境変数設定 37
- セットアップ 36

C

- 複数接続アプリケーションの構築 173

C アプリケーション

- 構築 171

C ルーチン

- 構築 175

C++

- 複数接続アプリケーションの構築 181

C++ アプリケーション

- 構築 179

C++ ルーチン

- 構築 183

Java セットアップ 38

Micro Focus COBOL アプリケーション

- 構築 187

Micro Focus COBOL ルーチン

- 構築 189

SQLJ アプリケーション

- コンパイル・オプション 139

SQLJ ルーチン

- コンパイル・オプション 145

V

Visual Basic

- サポートされる Windows バージョン 23

サンプル 108

疎結合トランザクション

- トラブルシューティング 281

Windows 上での構築 279

ADO アプリケーションの構築 276

OLE オートメーション 284

RDO アプリケーションの構築 283

Visual Basic .NET

アプリケーション

- コンパイルおよびリンク・オプション 291

構築、アプリケーションの 289

サンプル 109

バッチ・ファイル 112

Visual C++

サンプル 111

ADO アプリケーションの構築 298

OLE オートメーション 300

W

WCHARTYPE CONVERT

- プリコンパイラー・オプション 274

wchar_t データ・タイプ

- CONVERT プリコンパイル・オプション 274

WebSphere MQ ユーザー定義関数

- セットアップ 33

Windows

アプリケーション開発

- 環境のセットアップ 45

C/C++ アプリケーション

- 構築 301

- コンパイルおよびリンク・オプション 304

C/C++ ルーチン

- 構築 305

- コンパイルおよびリンク・オプション 309

IBM COBOL アプリケーション

- 構築 314

- コンパイルおよびリンク・オプション 317

IBM COBOL ルーチン

- 構築 318

- コンパイルおよびリンク・オプション 320

Java

- セットアップ 49

Management Instrumentation 275

- サンプル 112

Micro Focus COBOL アプリケーション

- 構築 323

- コンパイルおよびリンク・オプション 325

Micro Focus COBOL ルーチン

- 構築 325

- コンパイルおよびリンク・オプション 327

SQLJ アプリケーション

- コンパイル・オプション 142

SQLJ ルーチン

- コンパイル・オプション 148

Windows 用の Object REXX 328

- サンプル 102

[特殊文字]

.NET

バッチ・ファイル 112

ルーチン

- コンパイルおよびリンク・オプション 297

Windows 上での構築 293

.NET (続き)

C# アプリケーション

- コンパイルおよびリンク・オプション 287

Windows 上での構築 285

C# のサンプル 82

Visual Basic アプリケーション

- コンパイルおよびリンク・オプション 291

Windows 上での構築 289

Visual Basic のサンプル 109

IBM と連絡をとる

技術上の問題がある場合は、お客様サポートにご連絡ください。

製品情報

DB2 Universal Database 製品に関する情報は、
<http://www.ibm.com/software/data/db2/udb> から入手できます。

このサイトには、技術ライブラリー、資料の注文方法、製品のダウンロード、ニュースグループ、フィックスパック、ニュース、および Web リソースへのリンクに関する最新情報が掲載されています。

米国以外の国で IBM に連絡する方法については、IBM Worldwide ページ (www.ibm.com/planetwide) にアクセスしてください。



部品番号: CT2TUJA

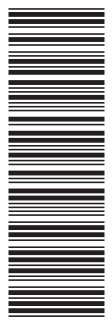
Printed in Japan

SC88-9137-01



日本アイ・ビー・エム株式会社
〒106-8711 東京都港区六本木3-2-12

(1P) P/N: CT2TUJA



Spine information:



IBM® DB2™ Universal
Database

アプリケーションの構築および実行

バージョン 8.2