IBM

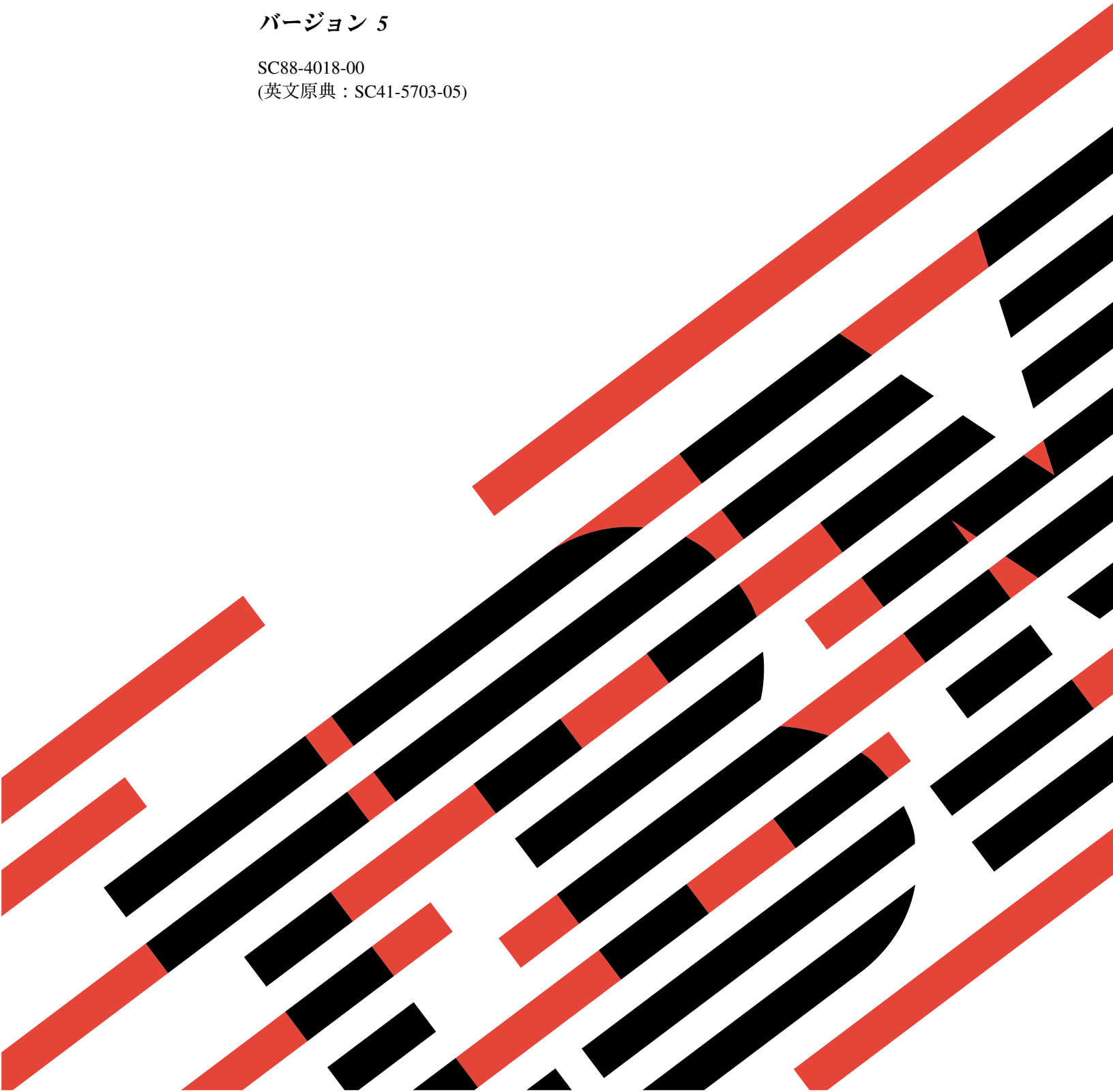
eserver

iSeries

DB2 Universal Database for iSeries Query 管理機能 プログラミング

バージョン 5

SC88-4018-00
(英文原典：SC41-5703-05)



IBM

@server

iSeries

DB2 Universal Database for iSeries Query 管理機能 プログラミング

バージョン 5

SC88-4018-00
(英文原典：SC41-5703-05)

お願い

本書および本書で紹介する製品をご使用になる前に、 255 ページの『付録 E. 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)

原　典： SC41–5703–05
iSeries
DB2 Universal Database for iSeries
Query Management Programming
Version 5

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 2002.8

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 UDB for iSeries Query 管理機能 プログラミング」について

DB2® UDB for iSeries Query 管理機能は、いろいろな DB2 プラットフォームにまたがるリレーショナル・データベースのデータにアクセスし、その結果を報告するための共通の方式を提供します。

Query 管理機能を使用すると、Query の処理によって得られる印刷報告書の設計およびフォーマット設定を行うことができます。これは、柔軟性のある優れた報告書作成ツールです。 (DB2 UDB for iSeries Query Manager は、Query 管理機能に対する使いやすいフロントエンドを提供します。) Query は、RPG、COBOL、FORTRAN、 C/400®、 ILE C/400、または PLI で書かれるプログラムに組み込むことができ、また、CL プログラムの中から実行することもできます。プログラマーは、これを使用して、環境を設定する際に柔軟性を持たせることができます。

Query 管理機能は、 OS/400® ライセンス・プログラムに組み込まれています。 OS/400 のコマンドを使用して、以下のことができます。

- ソース・ファイルまたは既存の Query for iSeries 定義からの Query オブジェクトのインポート。 これは、Query オブジェクトがソース・ファイル定義から作成されることを意味します。
- ソース・ファイルまたは既存の Query for iSeries の定義からの Query 書式オブジェクトのインポート。これは、関連するソース定義から書式オブジェクトが作成されることを意味します。
- Query オブジェクトのソース・ファイルへのエクスポート (EXPORT)。
- Query 書式オブジェクトのソース・ファイルへのエクスポート。
- コマンド行からの Query の実行。
- システムからの関係のあるオブジェクトの削除。
- 変換の前に、既存の Query for iSeries 定義を分析。

「Query 管理機能 プログラミング」の対象読者

Query 管理機能自体は、エンド・ユーザーを対象にしたツールではありません。この機能を最も効果的に使用するのは、データ処理の専門家です。このような専門家は、CL や高水準言語のプログラムの中で使用することができる Query を作成する技術を持っています。

エンド・ユーザーの中から何人かを選んで、DB2 UDB for iSeries Query Manager を使用して独自の Query を作成することができるように教育することもできます。 DB2 UDB for iSeries Query Manager を使用して作成し、保管した Query は、Query 管理機能コマンドにより、バッチ・モードで使用することができます。ただし、一般的に、エンド・ユーザーが独自の報告書を作成したい場合には、Query for iSeries または Query Manager を使用するほうがよいと言えます。

複数のマシン間での互換性が必要な場合は、Query 管理機能の使用を考慮する必要があります。

本書の対象読者は、次のとおりです。

- Query 管理機能の共通プログラミング・インターフェース (CPI) および QUERY インターフェースに精通しているアプリケーション・プログラマー。
- Query for iSeries のプロダクト、および Query for iSeries 定義の使用法に精通しているアプリケーション・プログラマー。
- iSeries システムの正規の研修を修了し、iSeries システムの操作、および Query の諸機能に精通しているシステム・オペレーター。

- 問題分析を行う担当者。
- システム・プログラムやこのプロダクトに関連する問題で、ユーザーに起因しない問題の解決を担当する IBM® のプログラミング・サービス担当員。

1 つの報告書で、大量のデータ行が生成される場合には、 Query 管理機能のパフォーマンスを検討しなければなりません。結果としての報告書がきわめて大きな報告書になる場合、対話式トランザクション・アプリケーションとしては、Query 管理機能はお勧めできません。

ただし、Query が、大きなファイルについて要約した計算に基づく報告書を生成する場合、パフォーマンスは良好です。たとえば、SUM、AVG、COUNT、MAX、または MIN の計算を使用し、それらの結果のみを印刷し、明細行は印刷しない場合です。

Query 管理機能が行わないこと

Query 管理機能は、データベースにアクセスした後、データをプログラムには戻しません。データをプログラムに戻す処理が必要な場合には、組み込み SQL を使用するようにしてください。

Query 管理機能は、Query の定義、Query の変更、または報告書のフォーマット設定を行うためのエンド・ユーザーに対するインターフェースを提供していません。 Query 管理機能は、被制御方式でこれを行います。

本書をお読みになる前に、次のことについて知っている必要があります。

- UDB Query Manager and SQL Development Kit ライセンス・プログラム
- iSeries システムの操作および諸機能
- Query for iSeries プロダクト

前提条件および関連情報

iSeries のテクニカルな情報を検索するには、まず iSeries Information Center を使用してください。

次の 2 つの方法で Information Center にアクセスできます。

- 次の Web サイトから

  `http://www.ibm.com/eserver/iseries/infocenter`
- オペレーティング・システム/400 オーダーと一緒に送られてきた CD-ROM から

  iSeries Information Center および PDF ライブラリー CD パッケージ, SK88-8055-01。このパッケージには、PDF 版の iSeries マニュアルである *iSeries Information Center: 補足マニュアル*, SK88-8056-01 も含まれています。これはソフトコピー・ライブラリー CD-ROM に置き換わるものです。

iSeries Information Center には、助言や重要なトピック (たとえば、 Java™、 TCP/IP、Web サービス、セキュア・ネットワーク、論理区画、クラスター化、CL コマンド、およびシステム・アプリケーション・プログラミング・インターフェース (API) など) があります。また、関連する IBM Redbooks™ へのリンク、および、Technical Studio や IBM ホーム・ページなどの IBM Web サイトへのインターネット・リンクもあります。

すべての新規のハードウェアのオーダーで、*iSeries セットアップおよびオペレーション CD*, SK88-8058-01 が届けられます。この CD-ROM には、IBM @server iSeries Access for Windows および EZ セットアップ・ウィザードが入っています。 iSeries Access は、PC を iSeries サーバーに接続するための強力なクライアント/サーバー機能を備えています。「EZ セットアップ」ウィザードにより、多くの iSeries セットアップ作業が自動化されています。

iSeries ナビゲーター

IBM iSeries ナビゲーターは、iSeries サーバーを管理するための強力なグラフィカル・インターフェースです。 iSeries ナビゲーターの機能には、システム・ナビゲーション、構成、計画の機能、作業の進行を助けるガイドなどがあります。 iSeries ナビゲーターは OS/400 オペレーティング・システムの新規拡張機能に対する唯一のユーザー・インターフェースであり、サーバーの運用および管理をより容易にし、生産性を向上させます。また、セントラル・サーバーから複数のサーバーを管理するためのマネージメント・セントラルも含まれています。

iSeries ナビゲーターについての詳細は、iSeries Information Center あるいは次の Web サイトを参照してください。

`http://www.ibm.com/eserver/iseries/navigator/`

第 1 章 Query 管理機能の紹介

この章では、オペレーティング・システム/400 (OS/400) の照会管理システム (DB2 UDB for iSeries Query Management) を紹介し、その特性、仕様、および要件を説明するとともに、Query for iSeries プログラムおよび機能との関係について説明します。

Query 管理機能の概要

本書では、Query 管理機能共通プログラミング・インターフェース (CPI) の OS/400 でのインプリメンテーションを取り上げています。 本書では、 DB2 UDB for iSeries Query 管理機能の CPI の機能とユーザー・インターフェースについて説明しています。

注: このプロダクトを取り扱うための前提知識として、Query 管理機能の共通プログラミング・インターフェース (CPI) に関する実務上の知識が必要です。また、Query for iSeries および構造化照会言語 (SQL) の知識も役立ちます。

ユーザーが CPI を使用すれば、リレーショナル・データベース中の情報にアクセスし、データを報告書用にフォーマット設定する時にそのデータの表示方法を制御することができます。 CPI により、照会と報告書作成という 2 つの大きなカテゴリーについてのサービスが受けられます。

DB2 UDB for iSeries Query 管理機能オブジェクトを使用するプログラム間呼び出し可能インターフェースを介して、アプリケーション・プログラムは Query 管理機能サービスを使用することができます。 QUERY 管理機能オブジェクトは、外部化 Query、プロシージャー、および書式定義が入っているファイルを使用して、CPI を介してのみ作成されます。外部化ファイルはエディターを使用して作成したり、アプリケーション・プログラムによって作成したり、別のシステム (ファイルはこのシステムからエクスポートされた) から転送したり、あるいは QUERY 管理機能を使用して作成された定義 (QRYDFN) オブジェクトを変換することによって、作成することができます。

Query 管理機能を使用するアプリケーションには、次のような利点があります。

- エラー処理および変換処理の必要性が削減されています。アプリケーションは SQL エラー・コードを検査する必要がありません。
- アプリケーション・コード外に定義され保管されている Query、プロシージャー、および書式の使用。Query 管理機能オブジェクトを変更することによって、アプリケーション・プログラムの変更や再コンパイルを行わずにアプリケーションを更新することができます。
- リレーショナル・データベース・マネージャー・プロトコルについての知識や操作が必要ありません。アプリケーションでは、数個の単純なコマンドを渡して、エクスポートされた形式内のデータへのアクセスを可能にすることができます。

プロシージャー内のステートメントを処理することによって、または呼び出し可能インターフェースを使用するプログラムを実行することによって、Query 管理機能コマンドを出すことができます。

OS/400 と Query 管理機能環境

Query 管理機能オブジェクトは、OS/400 システム・オブジェクトとして作成され維持されます。 2 ページの表 1 は、OS/400 システムの用語と Query 管理機能環境の用語の関係を示したものです。

表 1. iSeries および Query 管理機能の用語

OS/400 の用語	DB2 UDB for iSeries Query 管理機能での使用
ライブラリー — ライブラリーは、ユーザーが名前によってオブジェクトを探索できるように、関連オブジェクトをグループ化します。	**コレクション** — ライブラリー、ジャーナル、ジャーナル・レシーバー、データ・ディクショナリー、および SQL カタログで構成される集まり。 コレクションは、ユーザーが名前によってオブジェクトを探索できるように、関連オブジェクトをグループ化します。
物理ファイル — レコードのグループの集まり。	**表** — 列および行の集まり。
レコード — フィールドの集まり。	**行** — 表の横方向の部分で、連続した列の集まりが入る。
フィールド — 1 つのデータ・タイプの関連した情報の 1 つまたは複数の文字。	**列** — 1 つのデータ・タイプの表の縦方向の部分。
論理ファイル — 1 つまたは複数の物理ファイルのフィールドおよびレコードのサブセット。	**ビュー** — 1 つまたは複数の表の列および行のサブセット。
ユーザー・プロファイル — ユーザーを識別し、 iSeries システムにおける一連の特権を示す名前。	**許可 ID** — ユーザーを識別する 10 バイト以内の文字ストリング。

Query 管理機能で使用されるコレクション

Query 管理機能は、すべてのオブジェクトを 1 つのコレクションに属するものとして取り扱います。 次のリストは、Query 管理機能のコレクションの規則を説明しています。

- Query 管理機能オブジェクト (Query、書式、プロシージャー) は、コレクションの一部ではありません。 表 1 に示されているように、 iSeries システムではコレクションはライブラリーになります。 Query 管理機能オブジェクトがライブラリーの一部となることはありますが、そのライブラリーがコレクションである場合には、コレクション・カタログまたはジャーナルの中には Query 管理機能オブジェクトのための項目はありません。
- Query 管理機能コマンド (ERASE および SAVE) によって操作される表は、 SQL 表として取り扱われます。 Query 管理機能コマンドを使用して表を操作するときは、 SQL 表操作ステートメントに適用される DB2 UDB for iSeries SQL 規則が有効です。次のリストは、Query 管理機能コマンドと SQL ステートメントの間の対応を示しています。

  ERASE
  表の除去

  SAVE (新しい表へ)
  表の作成および挿入

  SAVE REPLACE
  削除および挿入

  SAVE APPEND
  挿入

Query 管理機能での命名規則

表またはビューを作成したり、あるいはデータベースに保管したいオブジェクトに名前を付ける場合には、以下の規則が適用されます。

Query 管理機能の Query オブジェクトの命名規則

Query オブジェクトは、SQL 名またはシステム名のいずれかを使用して、コマンドで指定することができます。使用する命名規則は、 START コマンドまたは Query 管理機能 Query の開始 (STRQMQRY) CL コマンドの Query コマンド・プロシージャーで指定します。Query 管理機能は、SQL による命名をデフォルトとして使用します。 1 つの Query インスタンスについて指定された命名規則は、インスタンス全体で使用され、かつそのインスタンスにのみ適用されます。 Query インスタンスに対して START コマンドを出した後で、命名規則を変更することはできません。

注: これらの命名規則は、コマンドに指定される Query オブジェクトにのみ適用されます。システムの命名規則は、常に、IMPORT および EXPORT コマンドに指定されるファイル名に適用されます。

Query 管理機能でのシステム命名

Query インスタンスがシステム名を使用する場合、 Query コマンドに指定される Query オブジェクト名に対して、次の規則が適用されます。

- Query オブジェクトは、区切り文字付き名を用いて指定することができます。区切り文字付き名とは、次の特性を持つ文字ストリングです。
  - 1〜8 文字の長さ
  - 引用符で始まり引用符で終わる (例: "MYFORM")
  - 以下の文字は含めてはなりません。
    - ブランク
    - アスタリスク (*)
    - 疑問符 (?)
    - アポストロフィ (')
    - 引用符 (")
    - 16 進数の 00〜3F または 16 進数の FF

  区切り文字付き名は修飾することができます。ただし、修飾子および名前は別個に引用符で囲んで区切らなければなりません (例: "MYLIB"/"MYFORM")。
- Query オブジェクトは、単純名を用いて指定することができます。単純名とは長さが 10 文字までの文字ストリングで、英字 (A 〜 Z、$、#、または @) で開始しなければなりません。単純名には、ピリオドおよびブランクは使用できません。
- Query コマンド・ファイル名は、10 文字までの長さのライブラリー名で修飾することができます。修飾ライブラリー名とファイル名は、スラッシュ (/) で区切らなければなりません。たとえば、MYLIB/FILE1 (ライブラリー MYLIB のファイル FILE1) は修飾名です。修飾子として使用されるライブラリー名には、引用符で囲まれた OS/400 の名前および単純名に適用される規則が適用されます。
- 同じライブラリーに保管される同じタイプのオブジェクトには、異なる名前を付けなければなりません (たとえば、 TEST という名前のファイルを 2 つ持つことはできません)。 Query と書式は、異なる OS/400 オブジェクト・タイプになります。したがって、Query と書式は同じ名前を持つことができます。プロシージャー、表、およびビューはすべて OS/400 ファイルであるため、名前が異なっていなければなりません。プロシージャー、表、またはビューは Query オブジェクトまたは書式オブジェクトと同じ名前を持つことができますが、別のプロシージャー、表、またはビューと同じ名前を持つことはできません。
- QUERY、書式、およびプロシージャーの名前には予約語 (FORM、 Query、 COUNT、 NULL など) を使用することができますが、SQL キーワードを用いる命名はお勧めできません。

命名規則

- Query コマンドで指定する Query 管理機能の Query、書式またはプロシージャーに修飾を付けない場合は、 OS/400 の探索規則に従います。 修飾を付けない Query 管理機能オブジェクト名を指定すると、Query 管理機能はライブラリー・リスト (*LIBL) からその Query 管理機能オブジェクトを探そうとします。 Query 管理機能オブジェクトを作成する時は、Query 管理機能はそのオブジェクトを現行ライブラリー (*CURLIB) に入れます。
- 表名またはビュー名を修飾しないで Query コマンドで指定する場合は、 SQL 解説書を参照してください。

システムによる命名および OS/400 探索規則の詳細については、 Information Center の制御言語 (CL) のトピックを参照してください。

Query 管理機能での SQL 命名

Query インスタンスで SQL 名を使用する時は、以下の規則が Query コマンドで指定する Query オブジェクト名に適用されます。これらの規則は、OS/400 のオブジェクト命名規則と類似しています。

- Query オブジェクトは、区切り文字付き名を用いて指定することができます。区切り文字付き名とは、次の特性を持つ文字ストリングです。
  - 1〜8 文字の長さ
  - 引用符で始まり引用符で終わる (例: "MYFORM")
  - 以下の文字は含めてはなりません。
    - ブランク
    - アスタリスク (*)
    - 疑問符 (?)
    - アポストロフィ (')
    - 引用符 (")
    - 16 進数の 00〜3F または 16 進数の FF

  区切り文字付き名は修飾することができます。ただし、修飾子および名前は別個に引用符で囲んで区切らなければなりません (例: "MYLIB"."MYFORM")。
- Query オブジェクトは、単純名を用いて指定することができます。単純名とは長さが 10 文字までの文字ストリングで、英字 (A 〜 Z、$、#、または @) で開始しなければなりません。単純名には、ピリオドおよびブランクは使用できません。
- 名前は、10 までの 1 バイト文字の別の名前 (通常は、ユーザー ID または許可 ID) で修飾し、修飾子と名前の間をピリオド (.) で区切ることができます。たとえば、Q.QUERY1 (コレクション Q の中の Query) は修飾名です。 Query 管理機能では、許可 ID をユーザー・プロファイルとして処理する DB2 UDB for iSeries SQL 規則に従います。 SQL の命名規則が適用される場合、Query 管理機能は許可 ID と同じ名前のオブジェクトをライブラリー内で探索しようとします。区切り文字付き名および単純名に適用される規則が、修飾子として使用されるライブラリー名に適用されます。
- 同じライブラリーに保管される同じタイプのオブジェクトには、異なる名前を付けなければなりません (たとえば、 TEST という名前のファイルを 2 つ持つことはできません)。 Query と書式は、異なる OS/400 オブジェクト・タイプになります。したがって、Query と書式は同じ名前を持つことができます。プロシージャー、表、およびビューはすべて OS/400 ファイルであるため、名前が異なっていなければなりません。プロシージャー、表、またはビューは Query オブジェクトまたは書式オブジェクトと同じ名前を持つことができますが、別のプロシージャー、表、またはビューと同じ名前を持つことはできません。
- Query 管理機能オブジェクト名を修飾しないで指定すると、Query 管理機能は、現行ユーザー・プロファイルと同じ名前をもつライブラリーからその Query オブジェクトを探索します。 Query オブジェク

トを作成する時は、Query 管理機能はそのオブジェクトを現行ユーザー・プロファイルと同じ名前のライブラリーに入れます。これは、DB2 UDB for iSeries Query 管理機能および SQL Development Kit ライセンス・プログラムが従っている規則と同じです。

Query 管理機能での OS/400 オブジェクトの命名規則: Query コマンドの IMPORT および EXPORT に指定するファイル名 は、 OS/400 のソース物理ファイルの命名規則に従います。

- Query オブジェクトは、区切り文字付き名を用いて指定することができます。区切り文字付き名とは、次の特性を持つ文字ストリングです。
  - 1〜8 文字の長さ
  - 引用符で始まり引用符で終わる (例: "MYLIB")
  - 以下の文字は含めてはなりません。
    - ブランク
    - アスタリスク (*)
    - 疑問符 (?)
    - アポストロフィ (')
    - 引用符 (")
    - 16 進数の 00〜3F または 16 進数の FF

  区切り文字付き名は修飾することができます。ただし、修飾子および名前は別個に引用符で囲んで区切らなければなりません (例: "MYLIB"/"MYFORM")。
- Query オブジェクトは、単純名を用いて指定することができます。単純名とは長さが 10 文字までの文字ストリングで、英字 (A 〜 Z、$、#、または @) で開始しなければなりません。単純名には、ピリオドおよびブランクは使用できません。
- 修飾名を用いて*ファイル名* を指定する場合も、OS/400 の規則が適用されます。 Query コマンドの中の*ファイル名* は、10 文字までのライブラリー名で修飾し、修飾子と名前の間をスラッシュ (/) で区切ることができます。たとえば、MYLIB/FILE1 (ライブラリー MYLIB の中のファイル) は修飾名です。
- Query コマンドで指定された*ファイル名* が修飾されていないと、Query 管理機能は、ライブラリー・リスト (*LIBL) を探索して、*ファイル名* という名前のソース・ファイルを探します。ファイルを作成している場合は、Query 管理機能はそのオブジェクトを現行ライブラリー (*CURLIB) に入れます。
- 物理ファイルが複数メンバーのソース・ファイルである場合、次の規則がメンバーの指定に適用されます。
  - メンバー名が指定されない場合、使用されるメンバー名のデフォルトとして、 IMPORT および EXPORT コマンドの *FIRST が使用されます。
  - メンバー名がファイル名の一部として与えられると、Query 管理機能は、物理ファイル内の指定されたメンバーを処理します。メンバー名はブランクを間に入れずに括弧でくくり、ファイル名の直後に続けなければなりません。たとえば、ファイル FILE1 内のメンバー MEMBER1 は、次のようにファイル名を入力して指定することができます。

    ```
    FILE1(MEMBER1)
    ```

Query 管理機能での変数名を使用した命名規則: 呼び出し可能インターフェースを介する SQL Query で変数を使用する場合に適用される規則については、 87 ページの『Query 管理機能 CI の拡張変数サポート』を参照してください。 OS/400 に固有の規則は次のとおりです。

- 変数名は、アンパーサンド (&) が前に付けられていなければなりません。アンパーサンドは変数名を区切るためのもので、変数名として使用できる 18 文字には含まれません。各アンパーサンドは個々の変数名の始まりを示す記号であり、 1 つの変数名に複数のアンパーサンドを用いることはできません。

命名規則

- ユーザー定義の変数を DSQ で始めることはできません。 DSQ で始まる変数をセットしようとした場合には、エラーが生成されます。
- Query 管理機能では、変数名の大文字と小文字は区別されます。したがって、i_owe_you という変数は、I_OWE_YOU という変数と同じではありません。

以下は有効な変数名です。

```
SQL Query では                GET/SET コマンドでは
--------------------          ------------------------
&I_owe_you                    I_owe_you
&MYVAR123                     MYVAR123
&THIS_IS_A_BIG_NAME           THIS_IS_A_BIG_NAME
```

Query 管理機能でのその他の Query 名の命名規則: Query インスタンスによって使用される命名規則は、そのインスタンス中で実行される SQL Query の SQL ステートメントに適用されます。その Query インスタンスでシステム名が使用されている場合には、システム名が SQL Query に適用されます。 DB2 UDB for iSeries SQL プロダクトが従う SQL 命名規則およびシステム命名規則については、「SQL 解説書」を参照してください。

Query 管理機能でのセキュリティーおよび権限

Query 管理機能では、 OS/400 のセキュリティーおよび権限モデルが使用されています。 iSeries システムのセキュリティーの概念については、*iSeries 機密保護解説書* を参照してください。

Query オブジェクト、OS/400 オブジェクト、および SQL について、次のような特別なセキュリティー権限の考慮事項があります。

- **Query オブジェクト:** Query コマンドによって Query オブジェクトを作成する場合には、他のユーザーに対して与えたい Query オブジェクトに対する共通権限のタイプを指定する各種の方法があります。権限のタイプは、Query コマンド・プロシージャーまたは START コマンドの DSQOAUTH キーワードを使用して指定することができます。 47 ページの『Query 管理機能での START』を参照してください。
- **OS/400 オブジェクト:** EXPORT におけるソース物理ファイルなどの非 Query オブジェクトを作成する場合にも、Query 管理機能は Query オブジェクトを作成する場合と同じ共通権限を使用します。
- **SQL:** SQL Query 内の SQL ステートメントに適用されるオブジェクト権限については、 SQL 解説書を参照してください。
- **借用権限:** ユーザーは、プログラムの実行中にプログラム所有者権限を得るので、そのプログラムに関連するファイルに対する権限を入手することになります。

Query 管理機能でのオブジェクト

Query 管理機能をサポートするためには、Query 管理機能プログラム・オブジェクトおよび Query 管理機能書式オブジェクトの 2 つのオブジェクト・タイプの定義が必要です。 Query 管理機能プログラム・オブジェクトの OS/400 オブジェクト・タイプは QMQRY です。 Query 管理機能書式オブジェクトの OS/400 オブジェクト・タイプは QMFORM です。 Query 管理機能プロシージャーは、単一メンバーのソース物理ファイルとして保管されます。

Query 管理機能でオブジェクトをサポートする CL コマンド

以下の Query 管理機能 CL コマンドは、QMQRY オブジェクトをサポートします。

CRTQMQRY

Query 管理機能プログラムの作成

DLTQMQRY
Query 管理機能プログラムの削除

RTVQMQRY
Query 管理機能プログラムの検索

STRQMQRY
Query 管理機能プログラムの開始

WRKQMQRY
Query 管理機能プログラムの処理

WRKQMQRY CL コマンドは、次の CL コマンドをサポートしています。

CHGOBJD
オブジェクト記述の変更

DLTQMQRY
Query 管理機能プログラムの削除

STRQMQRY
Query 管理機能プログラムの開始

以下の Query 管理機能 CL コマンドは、QMFORM オブジェクトをサポートします。

CRTQMFORM
Query 管理機能書式の作成

DLTQMFORM
Query 管理機能書式の削除

RTVQMFORM
Query 管理機能書式の検索

WRKQMFORM
Query 管理機能書式の処理

WRKQMFORM CL コマンドは、次の CL コマンドをサポートしています。

CHGOBJD
オブジェクト記述の変更

DLTQMFORM
Query 管理機能書式の削除

Query 管理機能での総称コマンド

QMQRY および QMFORM オブジェクトに対しては、次の総称コマンドを実行することができます。

CHKOBJ
オブジェクトの検査

CHGOBJD
オブジェクト記述の変更

CHGOBJOWN
オブジェクト権限の変更

CRTDUPOBJ
重複オブジェクトの作成

命名規則

DSPOBJAUT
オブジェクト権限の表示

DSPOBJD
オブジェクト記述の表示

EDTOBJAUT
オブジェクト権限の編集

GRTOBJAUT
オブジェクト権限の付与

MOVOBJ
オブジェクトの移動

RNMOBJ
オブジェクトの名前変更

RSTOBJ
オブジェクトの復元

RVKOBJAUT
オブジェクト権限の取り消し

SAVCHGOBJ
変更オブジェクトの保管

SAVOBJ
オブジェクトの保管

WRKOBJ
オブジェクトの処理

Query 管理機能でのメッセージ記述

Query 管理機能コマンドおよび機能の処理時には、次のようなエラー・メッセージが出されることがあります。

FAILURE
出された Query 管理機能コマンドは正常に実行されなかったので、処理が停止しています。コマンドが正常に実行されなかった理由を詳細に示し、エラーを訂正するために可能ないくつかの解決法を示すメッセージがディスプレイ装置に戻されます。

SEVERE
出されたコマンドを Query 管理機能が処理しようとした時に重大エラーが起こり、すべての処理が停止しています。エラーの理由を詳細に示し、そのエラーを訂正するために可能ないくつかの解決法を示すメッセージがディスプレイ装置に戻されます。

SUCCESS
出された Query 管理機能コマンドは正常に処理され、データは使用可能な状態にあります。

WARNING
Query 管理機能が、出されたコマンドまたはプロシージャーの中でエラーを検出しましたが、処理は続行されます。そのエラーは無視されるか、あるいはエラーを訂正するためにシステムのデフォルトが使用されます。エラー・メッセージの中で、警告の説明、およびエラーを訂正するために実行できる処置があればそれを調べてください。

第 2 章 Query 管理機能での照会機能

DB2 UDB for iSeries Query 管理機能は、SQL を使用したリレーショナル・データに対する照会をサポートします。 iSeries システムに対してリモートから Query を実行する場合には、ローカルのソート・シーケンス表が使用されます。 iSeries システム以外のシステムに対してリモートから Query を実行する場合には、ソート・シーケンス表は使用**されません**。

注: ソート・シーケンス・サポートを使用するには、リモートの iSeries システムは、バージョン 2 リリース 3 以降のレベルでなければなりません。そうでない場合には、ソート・シーケンス表は使用されません。

SQL 解説書に定義されている基本ステートメント、 SELECT 式、データ定義、および許可ステートメントが特にサポートされています。これらの SQL 機能を Query で使用することによって、ユーザー・アプリケーションは、表定義、データ・アクセス許可、データベースの照会とデータの挿入、更新、および削除を行うことができます。 Query の結果は、報告書として表示または印刷することができます。

Query 管理機能は、プロンプト Query (指示照会) および SQL Query をサポートします。プロンプト Query の詳細については、*Query Manager ご使用の手引き* を参照してください。

Query は作成、命名、記憶、および検索することができます。保管された Query は、複数のユーザーおよびアプリケーションの間で共有することができます。ユーザーのアプリケーションで使用される Query は、アプリケーション開発時に定義し保管することも、あるいは、ユーザーのアプリケーションで作成しアプリケーション実行時に Query 管理機能で使用することもできます。

Query を保管しておくことによって、次の 2 つの点で柔軟性を得ることができます。まず、Query を変更し、それをアプリケーション・プログラムとは別に保管することができます。 2 番目に、Query には変数を含めることができます。変数に割り当てられる値は、ユーザーのアプリケーションに先立って、あるいはアプリケーションとともにセットすることができます。これら 2 つの機能を使用すれば、ユーザーのアプリケーションを書き直したり再コンパイルせずに、データ Query のパラメーターを変更することができます。

Query 管理機能での Query の作成

OS/400 では、Query は QMQRY オブジェクトです。 Query は、以下を使用して作成することができます。

- CRTQMQRY CL コマンド
- Query 管理機能の呼び出し可能プログラミング・インターフェースを介した IMPORT QUERY コマンド
- DB2 UDB for iSeries Query Manager の Query 作成機能

Query 管理機能プログラム・オブジェクトは、 OS/400 の *QMQRY オブジェクトとして保管されます。外部化 Query のソース・メンバーには、 SQL ステートメントを含むテキスト・ストリングが入っていなければならず、また必要に応じて変数を入れることもできます。

変数は、Query のどのような部分にも入れることができ、 Query に書くことができるすべてのもの (以下に示すような) を表すことができます。

- 列名
- 検索条件

- 副選択
- 特定の値
- 複数の文節
- 部分文節

1 つまたは複数の変数を含む Query を作成することもできます。

以下は、Query プログラムの例です。

```
-- この Query は、ある部門の従業員について、名前、勤続年数、
-- および給料をリストします。部門 (DEPTNUM) は変数で、
-- Query を実行する前に必ずセットしなければなりません。
H QM4 05 Q 01 E V W E R 01 03 92/10/24 11:07
V 1001 050 Department Query
V 5001 011 QSYS/QASCII
V 5002 003 ENV
  SELECT NAME, YEARS, SALARY -- 使用する列の名前
  FROM Q.STAFF               -- 使用する表の名前
  WHERE DEPT=&DEPTNUM        -- 変数の選択条件
```

注: コメント、H レコード、V レコードの指定は任意です。 SQL ステートメントのみが必要です。

Query 管理機能は、この Query が処理のためにデータベース・マネージャーに渡される前に、コメントを取り除き、変数の置き換えを行います。

例: Query 管理機能での Query の作成

外部化 Query ソース・ファイル・メンバーをインポートして、 Query 管理機能プログラム・オブジェクトを作成することによって、 Query 管理機能により Query を作成することができます。

1. メンバーを入れるソース・ファイルを作成するには、次のコマンドを使用します。

   ```
   CRTSRCPF FILE(MYLIB1/QQRYSRC) MBR(QUERY1)
   ```

2. ソース・ファイル・メンバーの QUERY1 を編集して、次のような情報を追加します。

   ```
   H QM4 05 Q 01 E V W E R 01 03 92/10/24 11:07
   V 1001 050 Department Query
   V 5001 011 QSYS/QASCII
   V 5002 003 ENV
     SELECT NAME, YEARS, SALARY -- 使用する列の名前
     FROM Q.STAFF               -- 使用する表の名前
     WHERE DEPT=&DEPTNUM        -- 変数の選択条件
   ```

3. ソース・ファイル・メンバー QUERY1 を保管します。
4. MYLIB1 に Query オブジェクトを作成するには、 Query 管理機能プログラムの作成 (CRTQMQRY) コマンドを使用して、上記で作成したソース・ファイル・メンバーをインポートします。 OS/400 コマンド行に、次のように入力して、

   ```
   CRTQMQRY QMQRY(MYLIB1/QUERY1)
   SRCFILE(MYLIB1/QQRYSRC) SRCMBR(QUERY1)
   ```

   実行キーを押します。

Query 管理機能は、この Query が処理のためにデータベース・マネージャーに渡される前に、コメントを取り除き、変数の置き換えを行います。

Query 管理機能での Query の制約事項

Query 管理機能によって処理される Query には、次の制約事項が適用されます。

- Query はソース・ファイルのサイズによる制約を受けます。

- Query の単一行の長さは最大で 79 バイトまでです。
- 置換変数値の長さは 55 文字以下でなければなりません。
- 置換変数の名前は、30 文字を超えることはできません。
- 外部化 Query ソース・ファイルには、SQL ステートメントだけしか入れられません。外部化プロンプト Query (指示照会) を、照会管理機能にインポートすることはできますが、実行することはできません。
- コメントの前には、必ず 2 つのハイフン (--) がなければなりません。この 2 つのハイフンと行末の間にある文字は、すべてコメントの一部と見なされます。
- コメントとブランクを取り除き、変数の置き換えを行った後の Query の合計の長さは、 32767 バイトを超えることはできません。
- 外部化 Query には、H レコードとその直後にコメントの V レコードを続けて入れることができます。このコメントは、オブジェクトがインポートされる時点のテキスト記述として使用することができます。
- 外部化 Query には、ソート・シーケンス値、言語 ID、またはその両方を入れることができます。これらを入れる場合には、H レコードまたはコメント・レコードの直後に入れなければなりません。

Query 管理機能での変数置換

Query 管理機能プログラムでの変数置換には、次の規則が適用されます。

- 変数がコメントの中にある場合は、変数置換は行われません。
- 変数が固定情報または区切り文字付き名前で囲まれた名前の中にある場合には、変数置換は行われません。
- SQL Query 内の変数は、アンパーサンド (&) で始まり、有効な変数名文字でない任意の文字で終わる、1 つの文字ストリングとして定義されます。
- Query 管理機能では、変数の間の余分なブランクは置き換えられません。したがって、変数置換を連結手段として使用することができます。次の例では、Query 管理機能の SET コマンドは、呼び出し可能インターフェースを介して、あるいはプロシージャーの中で処理されます。

```
SET GLOBAL (library='MYLIB'
SET GLOBAL (table='MYTABLE'
SET GLOBAL (dol=10
SET GLOBAL (cnts=50
```

それから、次の SQL Query を実行すると、最後の SQL ステートメントを処理したことになります。

```
SELECT * FROM &library.&table
         WHERE PRICE  = &dol.&cnts
```

```
SELECT * FROM MYLIB.MYTABLE
         WHERE PRICE  = 10.50
```

Query 管理機能での変数プロンプト

Query 管理機能は、ユーザーの表示装置にメッセージを送り、使用すべき値についてのプロンプトを出します。 このプロンプトは、ジョブが対話式に実行され、しかも Query で指定された変数がグローバル変数プールにセットされていない場合に行われます。

変数に対して有効な値が入力されると、Query が処理されます。値を入力せずに実行キーを押した場合、あるいは F3 (終了) または F12 (取り消し) を押した場合、 Query はエラーになり、失敗します。特殊な値である *BLANK を入力することによって、変数名を 1 つのブランクに置き換えることもできます。

ジョブがバッチ・モードで実行され、しかも Query で指定した変数がグローバル変数プールにセットされていない場合には、 Query は失敗します。変数は置換されません。

正しくない変数名を使用して Query を実行しようとすると、いかなる場合も Query はエラーになり、失敗します。たとえば、変数名が長過ぎる場合や、& の後の最初の文字が英字でない場合には、 Query は失敗します。

Query 管理機能でのコメント

Query 管理機能プログラム内のコメントは、次のように取り扱われます。

- コメントは、変数置換前に SQL Query から取り除かれます。ただし、置き換えられた変数の中のコメント区切り文字は取り除かれないので、 SQL のエラーまたは予測できない結果となることがあります。
- 引用符で囲まれたストリング中のコメント区切り文字は、コメントとしては取り扱われません。このようなストリングは、引用符 (“”) によって区切られた名前か、またはアポストロフィ (’’) によって区切られた固定情報のいずれかです。
- コメントの区切り文字である 2 つのハイフン (--) の間に、ブランクを挟むことはできません。

Query 管理機能での行の継続

Query 管理機能プログラムでの行の継続の使用は、次の規則によって管理されます。

- SQL 文節には SQL Query の複数行にまたがるものがあります。 SQL は行の継続文字をサポートしません。したがって、読みやすくするためには、 SQL ステートメント中のブランクを挿入することができる位置で新しい行を開始してください。文節が複数行にまたがる場合には、前の行の最後の文字が文節の一部であり、次の行の最初の文字が余分なブランクを含まない文節の一部となるところで、文節を分割することができます。次の行に続く文節の最後の文字は、79 桁以内でなければなりません。
- 固定情報および区切り文字付き名を複数の行にまたがらせることができます。

  注: 1 バイト文字セット (SBCS) および 2 バイト文字セット (DBCS) の混用文字ストリングは、複数の行に正常に分割することはできません。この場合は、SQL の連結機能を使用してください。

Query 管理機能でのソート・シーケンスの使用

ソート・シーケンス表で定義された *QMQRY オブジェクトを実行するのに Query 管理機能共通プログラミング・インターフェース (CPI) インターフェースを使用した場合には、そのオブジェクトに関連するソート・シーケンス表が Query の実行に使用されます。ソート・シーケンス表は、結果のデータから生成される報告書すべての表示や印刷の書式設定にも使用されます。

ソート・シーケンスをサポートしているリモートの iSeries システムに接続している場合には、オブジェクトに関連するソート・シーケンス表が SQL 処理に使用されます。

RUN QUERY を用いて Query for iSeries の *QRYDFN オブジェクトを変換し、実行する場合、そのオブジェクトが Query for iSeries ソート・シーケンス選択画面においてオプションの 1、2、または 3 を使用すると定義されていると、Query 管理機能は *HEX ソート・シーケンスを使用します。 Query for iSeries プロダクトについては、*Query for iSeries Use* を参照してください。 *QRYDFN オブジェクトがオプション 4 または 5 を使用するものとして定義されている場合には、 Query 管理機能は、Query を実行し報告書の書式設定を行うのに使用するものと同じソート・シーケンスを使用します。

第 3 章 Query 管理機能でのインスタンス処理

Query 管理機能のインスタンスとは、データベース・ファイルまたは Query for iSeries 定義で見つかったデータから、表示報告書または印刷報告書を作成するステップの進行過程です。 Query 管理機能は、指定されたデータを Query 管理機能プログラム (QMQRY) オブジェクトと呼ばれる DATA セット (Query を実行した結果得られる実際の情報) に入れます。このオブジェクトは、Query 管理機能書式 (QMFORM) オブジェクトから編成されます。この書式を変更することにより、同じ QMQRY を使用して、特定の状況で必要に応じた複数の報告書を作成することができます。この章では、ユーザーの必要に合わせて報告書を作成する Query 管理機能インスタンスの作成、変更、および変換の方法について説明します。

Query 管理機能でのインスタンスの作成

制御言語 (CL) コマンドまたはユーザー・アプリケーションを開始することにより、 Query 管理機能プログラム機能へのアクセスを行います。 Query 機能へのアクセスができるようになれば、Query 管理機能コマンドを使用して、 Query 管理機能のインスタンスを作成することができます。 1 つのインスタンスは 1 つの DATA セットです。これにはデータベース・ファイル、および Query を定義するために使用される DSQ 変数が入っているグローバル変数プールから収集されたデータが入っています。

コマンドを出すことによって作成した Query 管理機能インスタンスを使用して、必要な情報が得られるように、書式を作成または変更して印刷報告書または表示報告書を作成します。 図 1 は、Query 管理機能インスタンスの作成方法を示しています。

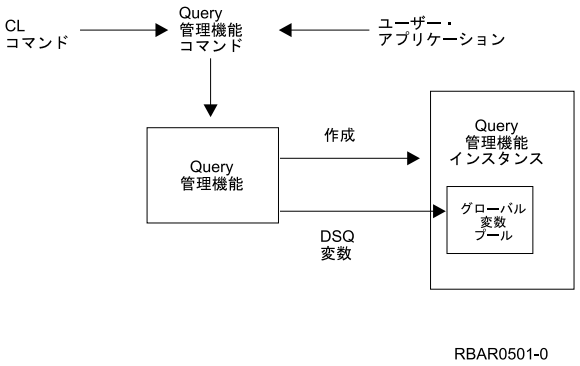


図 1. Query 管理機能インスタンスの作成

Query 管理機能インスタンスは、これと同じ手順および EXIT コマンドを使用して終了してください。その結果、インスタンスが破棄されます。

Query 管理機能での照会の実行

Query 管理機能プログラムを実行するためには、次の方法の 1 つを使用します。

- プロシージャーに、(STRQMPRC を介して) RUN QUERY コマンドを指定する。
- Query 管理機能プログラムの開始 (STRQMQRY) CL コマンドを出す。
- Query をユーザー・アプリケーションから実行する。

構造化照会言語 (SQL) の SELECT ステートメントを使用して、 Query 管理機能は OS/400 データベース・ファイルにアクセスし、 QMQRY で要求された情報をそのインスタンスに含まれる DATA セットに入れます。

Query の実行によって作成された DATA セットは、別の Query が処理されるか、あるいは関連したインスタンスが EXIT コマンドによって終了されるまで、存在し続けます。各 Query 管理機能インスタンスごとに、別々の DATA セットが作成されます。

注: DATA セットは、Query が SELECT ステートメントである場合にのみ作成されます。

図 2 は、Query 管理機能を使用した Query の実行方法を示しています。

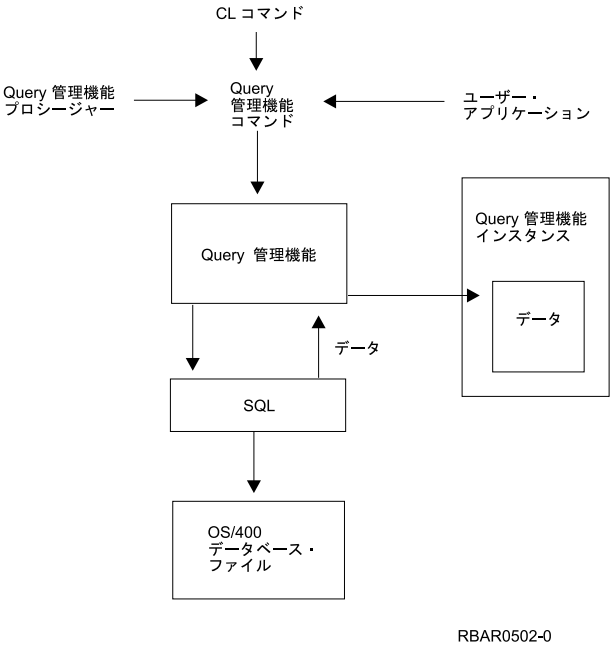


図 2. Query 管理機能プログラムの実行

Query 管理機能でのグローバル変数の置換

グローバル変数の置換を指定した Query を実行する場合、 Query 管理機能は前述した方法と同じ方法で要求を処理しますが、グローバル変数を分析解決するためには、インスタンスが作成された時に定義されたグローバル変数プールが検索されます。 Query で指定された変数がグローバル変数プールにセットされていない場合には、 Query 管理機能はディスプレイ装置にメッセージを送り、指定されたフィールドで使用される値を求めるプロンプトを表示します。プロンプトが出されたフィールドに正しい値を入力してください。そうすれば、Query はプロンプトに対して入力された変数値を使用して実行されます。

Query 管理機能でのレポートの作成

DATA セットを作成すると、報告書を印刷するかまたはディスプレイ装置で表示することを要求できます。 DATA セットの中のデータを、ユーザーの必要に合わせた固有の書式に入れる QMFORM オブジェクトを作成または変更するためには、 Query 管理機能書式の作成 (CRTQMFORM) コマンドまたは Query 管理機能書式の処理 (WRKQMFORM) コマンドを使用します。

QMFORM および DATA セットが作成された後に、報告書表示画面を使用して報告書をディスプレイ装置で表示するか、あるいは PRINT コマンドを使用して報告書データの印刷バージョンを作成してください。

注: 報告書を作成するには、前もって Query を実行し、 DATA セットを作成しておくことが必要です。

図 3 は、 Query 管理機能の指針を使用した報告書の表示または印刷方法を示しています。

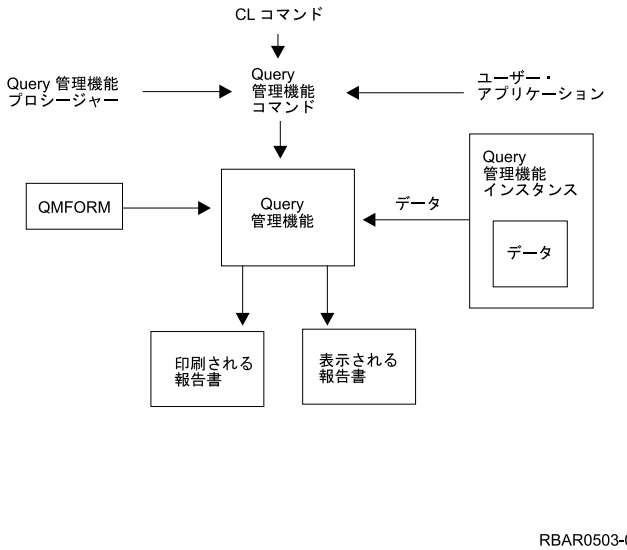


図 3. Query 管理機能報告書の作成

Query 管理機能での Query または書式オブジェクトのインポート

Query 管理機能の報告書を作成できる Query オブジェクトまたは書式オブジェクトを作成するには、以下のいずれかの方法を使用します。

- IMPORT コマンドをプロシージャーに指定する。
- Query 管理機能プログラムの作成 (CRTQMQRY) または Query 管理機能書式の作成 (CRTQMFORM) CL コマンドを出す。
- IMPORT コマンドを使用して、 Query オブジェクトまたは書式オブジェクトをユーザー・アプリケーションからインポートする。

Query 管理機能プログラムまたは Query 管理機能書式は、 Query、または書式のソース仕様が入っているソース・ファイルのメンバーから作成されます。

Query の作成方法についての詳細は、9 ページの 『Query 管理機能での Query の作成』を参照してください。

書式の作成方法についての詳細は、61 ページの 『Query 管理機能での書式の作成』を参照してください。

プロシージャーの作成方法についての詳細は、 57 ページの 『Query 管理機能でのプロシージャーの作成』を参照してください。

Query 管理機能での Query または書式オブジェクトのエクスポート

Query オブジェクトまたは書式オブジェクトを変更できるようにオブジェクトをエクスポートするためには、以下のいずれかの方法を使用します。

- EXPORT コマンドをプロシージャーに指定する。
- Query 管理機能プログラムの検索 (RTVQMQRY) または Query 管理機能書式の検索 (RTVQMFORM) CL コマンドを出す。

- EXPORT コマンドを使用して、Query または書式をユーザー・アプリケーションからエクスポートする。

エクスポート処理を行うと、指定されたソース・ファイル・メンバーに、 Query または書式のソース仕様が作成されます。これにより、Query または書式のソース仕様を変更してインポートできるようになり、再度その Query を実行した場合には、行われた変更が反映されている報告書を作成することができます。

注: 属性が PROMPT の Query 管理機能プログラムは、属性が SQL の Query と同じようにエクスポートされます。

Query 管理機能でのプロシージャーのインポートおよびエクスポート

Query 管理機能プロシージャーをインポートおよびエクスポートするプロセスは、 1 つの例外を除いて、Query と書式をインポートおよびエクスポートするプロセスと同じです。 Query 管理機能プロシージャーは、1 つのソース物理ファイル・メンバーです。プロシージャーのインポートまたはエクスポートとは、情報を 1 つのメンバーから別のメンバーにコピーすることです。したがって、プロシージャーをインポートまたはエクスポートする必要はありません。それはすでに外部化された形式になっているからです。 図 4 は、Query、書式、およびプロシージャーのインポートおよびエクスポート処理を示しています。

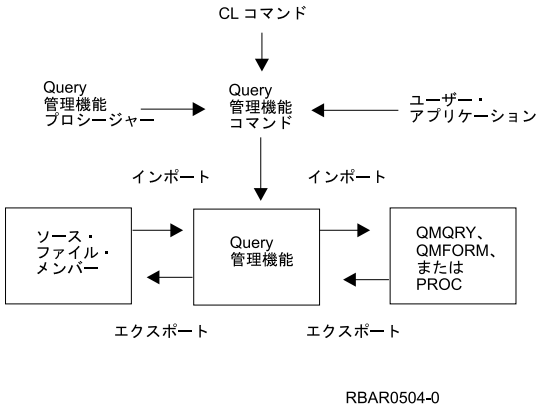


図 4. Query 管理機能メンバーのインポートおよびエクスポート

Query 管理機能でのプロシージャーの実行

Query 管理機能環境でプロシージャーの処理を開始するためには、次の方式の 1 つを使用します。

- RUN コマンドを使用して、別のプロシージャーの中から該当のプロシージャーを指定する。
- Query 管理機能プロシージャーの開始 (STRQMPRC) CL コマンドを出す。
- RUN コマンドを使用して、ユーザー・アプリケーションから該当のプロシージャーを開始する。

Query 管理機能が Query 管理機能インスタンスの作成に使用するために、ソース・ファイル・メンバーまたは別のプロシージャーからのデータを要求するためには、 Query 管理機能コマンドを使用します。プロシージャー内のコマンドは、 RUN PROC コマンドと関連したインスタンスと同じインスタンスを使用して処理されます。

また、Query 管理機能によって、この処理を使用してインスタンスを作成する時に複数のプロシージャーを呼び出すこともできます。 図 5 は、Query 管理機能プロシージャーの実行方法を示しています。

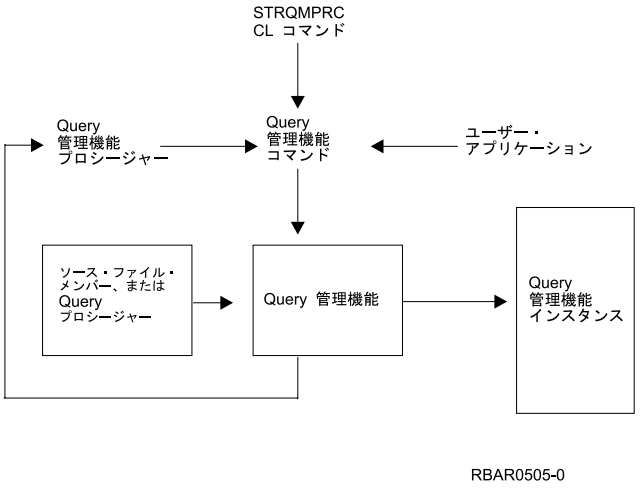


図 5. Query 管理機能プロシージャーの実行

Query 管理機能での SAVE DATA AS コマンドの使用

Query 管理機能では、ユーザー・インスタンスの DATA セットの中に作成されたデータを、 Query 管理機能の表に保管することができます。 SAVE DATA AS を使用して処理する場合、 Query 管理機能の処理を次のような方法で開始します。

- SAVE DATA AS コマンドを使用して、プロシージャーから保管操作を指定する。
- Query 管理機能プログラムの開始 (STRQMQRY) CL コマンドを出す。
- SAVE DATA AS コマンドを使用して、ユーザー・アプリケーションからコマンドを開始する。

前に作成されたインスタンス中の DATA セットからデータを使用して、 OS/400 データベース・ファイルのデータを Query 管理機能表に保管するように要求するためには、 Query 管理機能コマンドを使用してください。 Query 管理機能 DATA セットを作成するためには、 RUN QUERY コマンドが同じインスタンスのもとで処理されていることが必要です。

18 ページの図 6 は、Query 管理機能インスタンスを使用して、データベース・ファイルのデータを表に保管する方法を示しています。

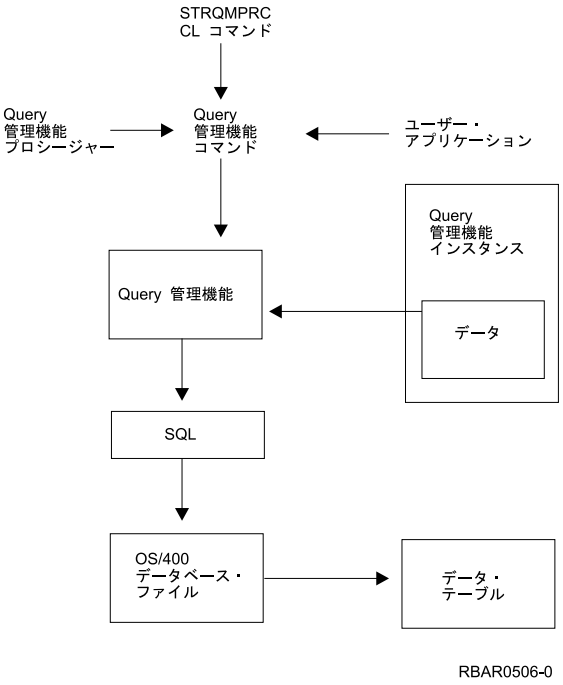


図 6. Query 管理機能表へのデータの保管

Query 管理機能での SET GLOBAL および GET GLOBAL コマンドの使用

Query 管理機能によって、グローバル変数プールの変数を入手し変更することができます。前に作成されたインスタンスの中の Query 管理機能変数の値を入手し、それをユーザー・プログラムまたはプロシージャーに提供するためには、 GET GLOBAL コマンドを使用します。

ユーザー・プログラムまたはプロシージャーから、前に作成されたインスタンスの中の Query 管理機能変数の値をセットしたり変更するためには、 SET GLOBAL コマンドを使用します。 GET GLOBAL および SET GLOBAL コマンドを使用して処理する場合、 Query 管理機能の処理を次のような方法で開始します。

- プロシージャーからコマンドを指定する。
- Query 管理機能プログラムの開始 (STRQMQRY) CL コマンドを出す。
- ユーザー・アプリケーションからコマンドを開始する。

前に作成されたインスタンスの中のグローバル変数プールからの値が、ユーザー・プログラムまたはプロシージャーからセットされるように要求するためには、 Query 管理機能コマンドを使用します。 GET GLOBAL 処理では、Query 管理機能がユーザー・プログラムのために 変数値を入手するという点を除いて、GET GLOBAL の処理は SET 処理の場合と同じです。

19 ページの図 7 は、 Query 管理機能が GET GLOBAL および SET GLOBAL コマンドを使用して、Query 管理機能変数を変更または検索する方法を示しています。

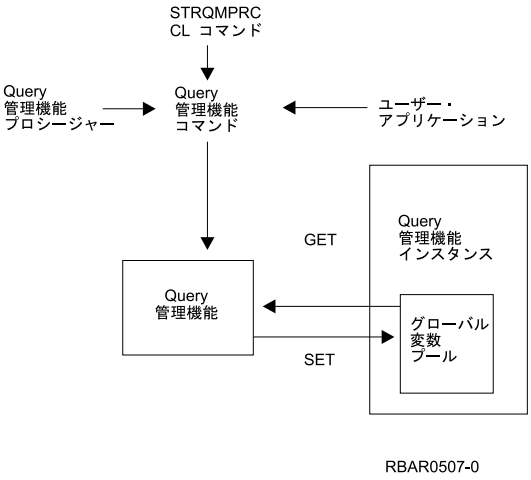


図 7. GET GLOBAL および SET GLOBAL コマンドの使用

Query 管理機能での活動化グループの紹介

活動化グループは、ジョブの中にある小さなジョブのようなものです。各活動化グループは、実行時ジョブの副構造です。それぞれ、1 つまたは複数のプログラムに割り振られたシステム・リソース (プログラムまたはプロシージャーの変数用のストレージ、コミットメント定義、およびオープン・ファイル) で構成されます。

アプリケーション・プログラムは、デフォルトの活動化グループに関連付けられている限り、処理を続行します。

ILE C/400 の使用を予定している場合には、iSeries システムではアプリケーション・プログラムを活動化グループに関連付けることができるため、活動化グループがプログラムにどのように影響するかについてさらに詳しく理解しておく必要があります。

デフォルト以外の活動化グループに関連付けることができるのは、ILE C/400 を使用して作成したアプリケーション・プログラムだけです。 DB2 UDB for iSeries Query Manager の機能は、デフォルトの活動化グループに関連付けられます。 Query 管理機能の CL コマンドもすべてデフォルトの活動化グループで実行されます。

Query 管理機能インスタンスは、そのインスタンスを開始したプログラムの活動化グループに関連付けられます。 ILE C/400 以外のプログラムにより作成された Query 管理機能インスタンスは、デフォルトの活動化グループに関連付けられます。 Query 管理機能の CL コマンドを使用して作成された Query 管理機能インスタンスは、常にデフォルトの活動化グループに関連付けられます。

アプリケーション・プログラムは、そのプログラムに関連付けられている活動化グループが Query 管理機能インスタンスにも関連付けられている場合にのみ、その Query 管理機能インスタンスを使用できます。複数の活動化グループが、1 つの Query 管理機能インスタンスを共用することはできません。ある活動化グループにおいて Query 管理機能インスタンスを介して行われる処理は、他の活動化グループでの Query 管理機能インスタンスによって行われる処理に影響を与えることはありません。

第 4 章 Query 管理機能でのコマンド

この章では、Query 管理機能で使用可能なコマンドについて説明します。データベース・ファイルから一般的な報告書を編成するアプリケーションを作成する場合には、次の Query 管理機能コマンドを使用してください。

- COMMIT
- CONNECT
- DISCONNECT
- ERASE
- EXIT
- EXPORT
- GET
- IMPORT
- PRINT
- RELEASE
- RUN
- SAVE DATA AS
- SET CONNECTION
- SET GLOBAL
- START

それぞれのコマンドについて詳細に説明し、またコマンドの構文を一目で理解できるように構文図を示してあります。構文図の説明については、22 ページの『Query 管理機能での構文図の読み方』を参照してください。

Query 管理機能でのコマンドおよびキーワードの指定

システム相互間でアプリケーションを移動する時の一貫性を確保するために、コマンドはプロシージャー内で使用されるか呼び出し可能インターフェースを介して渡されるかにかかわらず、すべて大文字で指定する必要があります。同じように、Query オブジェクトの処理時には、文字キーワードはすべて大文字で指定しなければなりません。テキスト行 (たとえば、ページ見出し) は、小文字でも大文字でも指定することができます。

Query 管理機能でのコマンド構文解析

Query 管理機能では、すべてのコマンドについて、コマンドに関連するキーワードおよび変数を、コマンド・ストリングの一部として、また呼び出し可能インターフェースの拡張パラメーター・リストの一部として示すことができます。重複が生じた場合、コマンド・ストリングに指定されているキーワードが、拡張パラメーター・リストに指定されているキーワードに優先します。

コマンド・ストリングおよび拡張パラメーター・リストのキーワードと値の解析は、次に示すように異なっています。

コマンド・ストリングのキーワードおよび変数

Query 管理機能で使用するコマンド・ストリングおよび変数は、以下の規則に従います。

コマンドおよびキーワードの指定

- Query 管理機能は、コマンド値はすべて文字ストリングであると想定します。
- 値は引用符 (") またはアポストロフィ (') で区切ることができます。
- 区切り文字は、値の一部とは見なされません。
- 引用符で区切られた値の中にある引用符は、2 つ重ねて使用しなければなりません (たとえば、'Joe"s' または "Joe""s")。 (この規則は、アポストロフィにも引用符にも適用されます。)
- アポストロフィで区切られた値の中にある引用符については、これを 2 つ重ねて使用する必要はありません。引用符で区切られた値の中のアポストロフィについても同様です。
- 引用符で区切られた値の始めまたは終わりにあるブランクは、除去されません。
- アポストロフィまたは引用符で区切られていない値は、ブランクまたはコマンド・ストリングの終わりによって区切られます。
- 整数として使用されるキーワードの値 (PRINT コマンドの WIDTH および LENGTH など) は、Query 管理機能に対しては、整数値または文字ストリング値として渡されます。文字ストリングとして渡された場合は、使用前に整数に変換されます。

拡張パラメーター・リストのキーワードおよび変数

Query 管理機能で使用するパラメーター・リストのキーワードおよび変数は、以下の規則に従います。

- 先行ブランクおよび後書きブランクは、使用前にキーワード名から取り除かれます。
- 後書きブランクはキーワード値から取り除かれますが、先行ブランクは取り除かれません。
- 先行ブランクおよび後書きブランクは、変数名および変数値が Query 管理機能変数プールに追加される前に、変数値から取り除かれます。
- アポストロフィおよび引用符は、Query 管理機能によって取り除かれることはありません。
- Query 管理機能は、引用符またはアポストロフィを除去することはありません。
- 整数として使用されるキーワードの値 (PRINT コマンドの WIDTH および LENGTH など) は、Query 管理機能に対しては、整数値または文字ストリング値として渡されます。文字ストリングとして渡された場合は、使用前に整数に変換されます。

Query 管理機能での構文図の読み方

この章では、以下に定義する構造を使用して構文を説明しています。コマンド構文では、コマンド・ストリングのワードを区切るために、 1 つまたは複数のブランクをどの位置でも使用することができます。

- 構文図は、線に沿って左から右へ、上から下へ読んでください。

  ▸▸── の記号は、ステートメントの始めを示します。

  ──▸ の記号は、ステートメントの構文が次の行に続くことを示します。

  ▸── の記号は、ステートメントが前の行からの続きであることを示します。

  ──▸◂ の記号は、ステートメントの終わりを示します。

  完全なステートメント以外の構文単位の図は、 ▸── の記号で始まり、──▸ の記号で終わります。

- 必要項目は水平方向の線 (主線) 上に示されます。

```
▸▸─STATEMENT─*必要項目*──────────────────────────────────────────▸◂
```

- オプション項目は、メインパスの下に示されます。

```
▸▸─STATEMENT─┬──────────────────────────────────────────────────▸◂
             └*オプション項目*┘
```

- 複数の項目から選択することができる場合、項目をスタック状に積み重ねて示します。

項目の中から 1 つを選択しなければならない 場合は、スタックを形成する項目の 1 つが主線上に示されます。

項目の中から 1 つを選択してもしなくてもよいオプションの場合は、スタック全体が主線の下側に示されます。

オプション項目の 1 つがデフォルトになっている場合、デフォルトの選択項目が主線の上側に示され、残りのオプション項目が主線の下側に示されます。

- 主線から上方に別れて左に戻る矢印は、反復できる項目を示します。

  反復矢印がスタックの上方にある場合は、スタックの中の項目を反復できることを示します。
- キーワードは大文字で示されます (たとえば、PARM1)。これは、表示されているとおりにつづらなければなりません。変数はすべて小文字で示されます (たとえば、*parmx*)。これらは、ユーザーが提供する名前または値を示します。
- 句読点、括弧、算術演算子、またはこの種の記号が示されている場合は、構文の一部として入力する必要があります。

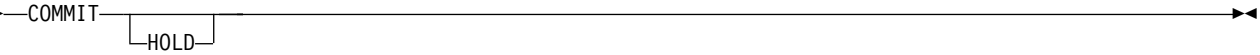
Query 管理機能での COMMIT

COMMIT コマンドは、リカバリーの 1 単位を終了し、そのリカバリー単位が行ったデータベースの変更をコミットします。

COMMIT コマンドが正常に終了すると、すべての作業がコミットされます。

COMMIT コマンドが正常に終了しなかった場合は、活動化グループの接続状態とその接続の状態は変わりません。

現行接続で RELEASE コマンドが実行されなかった場合は、 COMMIT コマンドによる現行接続の変更はありません。現行接続で RELEASE コマンドが実行されている場合は、その接続は除去され、活動化グループは非接続状態になります。このような場合は、次の Query 管理機能コマンドは、 CONNECT または SET CONNECTION コマンドでなければなりません。 Query 管理機能の方では、現行の接続が除かれたことは分かりません。 CONNECT または SET CONNECTION コマンドの前に出される SQL ステートメントは、すべて失敗します。



HOLD

リソースの保持 (hold) を指示します。これを指定すると、現在オープンしているカーソルはクローズされず、準備された SQL ステートメントは残され、また、この作業単位の間に獲得されたリソースは

COMMIT

すべて保持されます。ただし、トランザクション中に暗黙的に獲得された特定の行およびオブジェクトのロックは、解放されます。 HOLD を省略すると、オープン・カーソルは、WITH HOLD を指定して宣言されたものを除いて、クローズされ、準備済みの SQL ステートメントが廃棄され、また、保持されていたリソースは解放されます。

Query 管理機能での COMMIT の例

次の作業単位で RDBONE に接続する必要がないとします。この場合、次のコマンドを使用すると、コミット操作が正常に行われれば、接続が除去されます。

```
RELEASE RDBONE
COMMIT
```

RDBTWO になされた変更はコミットしたいが、現行作業単位中に獲得されたリソースは保持したい場合は、次のようにします。

```
COMMIT HOLD
```

Query 管理機能での CONNECT

CONNECT コマンドを使用して、 Query インスタンスに関連づけられたデータベースを変更することができます。このコマンドを使用して、アプリケーション・プロセスをデータベースに接続するか、あるいはその接続をローカル・データベース・マネージャーに戻すことができます。 CONNECT コマンドは、データベースに接続し、それを現行状態に置きます。現行状態とは、その接続が、出された SQL ステートメントで使用されることを意味します。活動化グループ 1 つに対し、接続は 1 つだけ現行状態になれます。

CONNECT コマンドが正常に行われると、次のようになります。

- 現行接続がある場合には、休止状態になります。
- 識別されたアプリケーション・サーバーは、現在の状態に入れられます。
- グローバル変数の DSQSDBNM と DSQCMTLV は更新されます。 DSQSDBNM には、現行データベースの名前が入ります。 DSQCMTLV には、使用されるコミットメント制御レベルが入ります。 InstanceInfo の ConnectionInfo は、SQL からの必要な情報で更新されて、現行状態を示します。

CONNECT コマンドが正常に終わらなかった場合は、活動化グループの接続状態とその接続の状態は変わりません。インスタンス情報は更新されません。

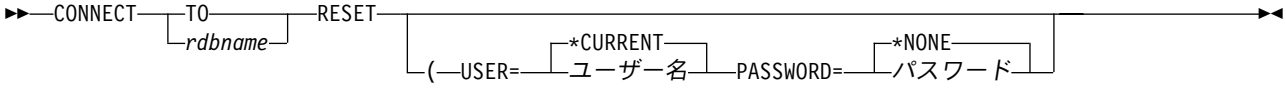
CONNECT の機能は、使用している接続管理方式によって異なります。この方式とは、分散作業単位 (*DUW) とリモート作業単位 (*RUW) です。デフォルト設定は、変更しない限り、*DUW です。接続管理方式を変更することについての詳細は、 47 ページの『Query 管理機能での START』を参照してください。 表 2 では、接続管理方式による機能の相違をリストしています。

*表 2. *DUW と *RUW の CONNECT コマンドの相違点*

*DUW	*RUW
複数の接続が可能	接続は 1 つだけ可能
別のデータベースに CONNECT すると、前の接続を休止状態に置く。前の接続は切断されない。	別のデータベースに CONNECT すると、前の接続は切断される。以前の 1 つまたは複数の接続は、この接続を実行する前に切断される。
同じデータベースに連続して CONNECT すると、失敗となる。	同じデータベースに連続して CONNECT しても、現行接続の変化は起こらない。

DUW で稼働しているシステムが、RUW で稼働しているシステムに接続すると、読み取り専用接続となります。

注: 同種接続は、コミットメント制御に関する限りは、読み取り専用です。コミットメント制御のもとで更新を行うためには、読み取り専用接続は使用できません。ただし、同種の読み取り専用接続は、引き続きコミット不可能な更新を行うことができます。



Query 管理機能での CONNECT のパラメーター・リスト

TO
接続先のリモート・データベースを指定する前に使用されるキーワード。

rdbname
アプリケーション・サーバーとして実行するデータベース・インスタンスの名前。この名前を使用して、DRDA® を介してアクセスされるリレーショナル・データベースを指定します。この名前には、次の規則が適用されます。

- 長さは最大 18 文字です。
- 以下の文字だけで構成します。
  - 大文字 (A〜Z)
  - 数字 (0〜9)
  - 下線 (_)

  先頭の文字は、大文字でなければなりません。

リモート・データベース名を指定する場合、USER および PASSWORD キーワードを使用して、そのリモート・データベースで使用されるユーザー ID およびパスワードを指定できます。これらのキーワードのいずれかを指定したい場合でも、両方を指定しなければなりません。これらのキーワードを指定しない場合には、デフォルト・ユーザー ID は *CURRENT に、そしてデフォルト・パスワードは *NONE になります。

RESET
リモート・データベースとの現行の会話を解除し (リモート・データベースが接続されている場合)、ローカル・データベースに接続し直す必要があることを示します。

注: ローカル・データベースに接続し直すためには、 SAVE DATA AS コマンドを使用する前に、CONNECT コマンドを使用してください。また、いろいろな接続情報を持つ複数の Query インスタンスを管理するためにも、このコマンドを使用する必要があります。

CONNECT コマンドを出す前には、アプリケーション・プログラムが接続可能状態になっていなければなりません。接続に対して、RUN QUERY コマンドおよび ERASE TABLE コマンドを出す場合、 Query 呼び出し可能プログラミング・インターフェースを呼び出すアプリケーション・プログラムが、呼び出し可能スタックの中に留まっていなければなりません。そうでない場合、アプリケーション・プログラムが終了した時点で暗黙のうちに切断されます。

RUW 接続管理下での CONNECT の例

次の CONNECT コマンドは、RDB1 接続を作成し、それを現行状態に置きます。

```
CONNECT TO RDB1
```

CONNECT

次のコマンドは、前の接続を切断し、ローカル・データベースへの接続を作成します。

```
CONNECT RESET
```

次のコマンドは、接続を行う前に、前の接続を切断します。

```
CONNECT TO RDB2 (USER=BIZET PASSWORD=CARMEN)
```

DUW 接続管理下での CONNECT の例

次の CONNECT コマンドは、RDB1 接続を作成し、それを現行状態に置きます。

```
CONNECT TO RDB1
```

次の CONNECT コマンドは、現行接続を休止状態に置き、ローカル・データベースへの接続を作成します。ローカル・データベースが現行接続になります。

```
CONNECT RESET
```

次のコマンドは、現行接続を休止状態に置き、RDB2 への接続を作成します。これで、RDB1 とローカル接続が休止状態になることになります。

```
CONNECT TO RDB2 (USER=BIZET PASSWORD=CARMEN)
```

Query 管理機能での DISCONNECT

DISCONNECT コマンドは、1 つまたは複数の接続を壊します。

DISCONNECT コマンドで指定する接続は、現行作業単位の間に SQL ステートメントを実行するために使用した接続であってはなりません。同様に、保護会話の接続にすることはできません。保護会話の接続を除去するには、RELEASE コマンドを使用します。

DISCONNECT コマンドが正常に行われると、指定した接続が除去されます。

DISCONNECT コマンドが正常に行われないと、活動化グループの接続状態とその接続の状態は変わりません。

接続の作成と維持には、リソースが使用されます。再利用する予定のない接続は切断してください。後続の作業単位で必要な接続は切断してはなりません。

DISCONNECT コマンドは、コミット操作の直後に実行しなければなりません。現行接続を除去すると、活動化グループは接続されていない状態になります。このような場合は、次の Query 管理機能コマンドは、CONNECT または SET CONNECTION コマンドでなければなりません。 Query 管理機能の方では、現行の接続が除かれたことは分かりません。 CONNECT または SET CONNECTION コマンドの前に出される SQL ステートメントは、すべて失敗します。

rdbname

アプリケーション・サーバーとして実行するデータベースのグローバル固有名。これは、アクセスするリレーショナル・データベースを指定します。リレーショナル・データベースは、 DRDB サポートの使用によってアクセスできるデータベースを識別する手段です。 *rdbname* は、次のようにすることができます。

- 長さは 18 文字までです。

- 大文字 (A～Z)、数字 (0～9)、および下線 (_) だけで構成します。先頭の文字は、大文字でなければなりません。

CURRENT
活動化グループにある現行接続を識別します。

ALL
活動化グループにあるすべての接続を識別します。どの接続も保護会話を使用できません。

Query 管理機能での DISCONNECT の例

次の作業単位では、RDBONE への接続が必要ないとします。コミット操作の後に次のコマンドが実行されます。

```
DISCONNECT RDBONE
```

次の作業単位では、現行の接続が必要ないとします。コミット操作の後に次のコマンドが実行されます。

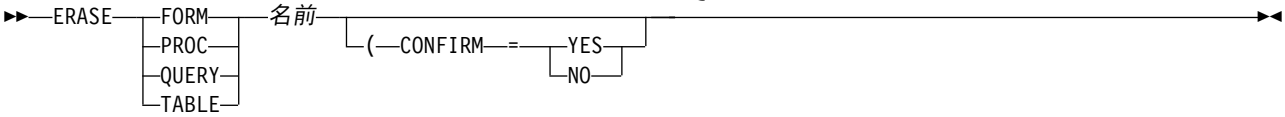
```
DISCONNECT CURRENT
```

次の作業単位では、既存の接続が必要ないとします。コミット操作の後に次のコマンドが実行されます。

```
DISCONNECT ALL
```

Query 管理機能での ERASE

ERASE コマンドは、データベースから FORM、PROC、QUERY、または TABLE を除去します。



Query 管理機能での ERASE のパラメーター・リスト

名前
除去したい FORM、PROC、QUERY、または TABLE の名前を指定します。オブジェクトを除去するためには、適切な所有権およびデータベース権限または Query 権限を持っていなければなりません。

この名前は、ライブラリー / オブジェクトまたはデータベース・オブジェクトの修飾名とすることができます。使用する予定の命名規則を最初の Query コマンド・プロシージャーに指定してください。

ユーザーは、*ALL 権限が認可されているオブジェクトだけを除去することができますが、そのオブジェクトが入っているライブラリーに対する *CHANGE または *ALL 権限を持っていることも必要です。

CONFIRM=YES | NO
このオプションにより、ERASE 要求を実行する前に確認することができます。確認要求が行われるのは、データベース内の既存のオブジェクトを除去しようとしている時だけです。保留中の ERASE コマンドを実行したいかどうかが尋ねられます。

CONFIRM=YES では、確認の表示が強制されます。 CONFIRM=NO は、確認の表示が抑止され、オブジェクトは除去されます。

ジョブを対話式で実行している場合には、照会メッセージがユーザーのディスプレイ装置に送られ、そのメッセージに応答するまでジョブが保留されます。メッセージは、オブジェクトを除去したいかどうかを尋ねてきます。ジョブがバッチ・ジョブとして実行されている場合、あるいは START 処理時に DSQSMODE がバッチ・モードにセットされた場合、 CONFIRM=YES はエラーを引き起こします。

ERASE

デフォルトは CONFIRM=YES です。 DSQCONFIRM 変数を、START コマンドのパラメーターとして設定するかまたは開始プロシージャーで設定することによって、このデフォルトを変更することができます。

ERASE コマンドを出した場合には、システムは次のようなメッセージを戻します。

```
Object exists. Do you want to replace it?
```

このメッセージは、CONFIRM=YES オプションを指定するか、あるいは CONFIRM オプションを省略した場合にだけ表示されます。

Query 管理機能での ERASE の例

```
ERASE TABLE EMP

ERASE TABLE SMITH.EMP (CONFIRM=YES
ERASE TABLE SMITH/EMP (CONFIRM=YES

ERASE PROC SMITH/MONTHEND (CONFIRM=NO
```

ERASE TABLE コマンドは、データベース物理ファイルについてのみ出すことができます。

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

Query 管理機能での EXIT

EXIT コマンドは、Query 管理機能とのユーザー・アプリケーション・セッションを停止し、ユーザーのプロセスにおける Query 管理機能の関連インスタンスを終了します。このコマンドでは、パラメーターを使用することはできません。メッセージは生成されません。

EXIT コマンドが有効であるのは、呼び出し可能インターフェースを介して出された場合だけです。

ジョブが終了した時には、すべての Query インスタンスについて暗黙の EXIT コマンドが処理されます。

EXIT コマンドは、Query プロシージャーの中では有効ではありません。

EXIT コマンドに関しては、権限について考慮すべき事項はありません。

Query 管理機能での EXIT の例

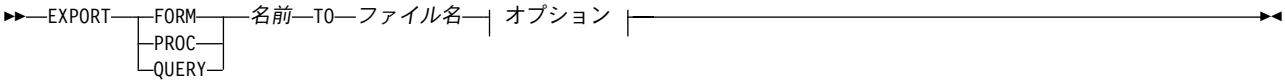
EXIT コマンドを使用するプログラムの例については、 『第 7 章 Query 管理機能での呼び出し可能インターフェース』を参照してください。

Query 管理機能での EXPORT

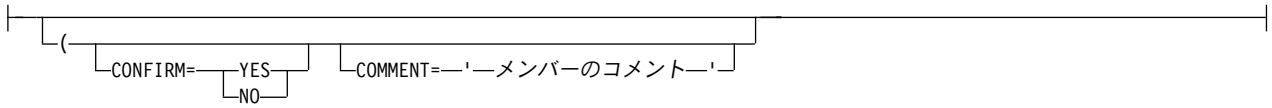
EXPORT コマンドは、特定の Query 管理機能オブジェクトの内容を含んでいるファイルを作成するのに使用されます。エクスポート・オブジェクトの詳細については、『第 8 章 Query 管理機能でのオブジェクトのエクスポートとインポート』で説明しています。エクスポートできるオブジェクトは、FORM、 PROC、または QUERY です。 *QMQRY オブジェクトに関連付けられているソート・シーケンス表の内容を、エクスポートすることはできません。

ソート・シーケンスについての詳細は、245 ページの 『付録 D. Query 管理機能でのソート・シーケンスの例』を参照してください。

iSeries システムのバージョン 2 リリース 3 より前に定義された *QMQRY オブジェクトは、常に 16 進数のソート・シーケンスを使用して実行されます。これらのオブジェクトがエクスポートされる場合、SRTSEQ は *HEX に設定され、 LANGID パラメーターはありません。



オプション



Query 管理機能での EXPORT のパラメーター・リスト

名前

エクスポートしたい FORM、PROC、または QUERY の名前を指定します。

この名前は、library/object または database.object の形式の修飾名にすることができます。使用する予定の命名規則を最初の Query コマンド・プロシージャーに指定してください。

指定した書式または Query が見つからず、START コマンドに DSQSCNVT=YES が指定されている場合は、Query 管理機能は、その名前をもつ Query for iSeries 定義を探索します。 Query 定義が見つかれば、その情報を使用して、Query 管理機能が使用可能な一時 Query または一時書式が作成されます。

ファイル名

エクスポート・オブジェクトを受け取るシステム・ファイルの名前を指定します。

複数のシステムで一貫性のあるものにするためには、修飾名または拡張子のない複数のファイル名を指定しないでください。こうすることによって、システムのデフォルトを有効にすることができます。

オブジェクトを iSeries システム以外のシステムにエクスポートする場合には、ファイルの名前を 1〜8 文字の長さにすることをお勧めします。

Query 管理機能は、単一名を修飾して、操作環境のデータ命名要件に適合させます。

これらの命名上の制約に注意して、複数の操作環境の間で Query 管理機能オブジェクトを転送する時に命名制約条件を処理できるようにしておかなければなりません。

CONFIRM=YES | NO

このオプションによって、EXPORT 要求を実行する前に確認することができます。確認要求は、既存のファイルが置き換えられようとする時にだけ行われます。保留中の変更を行いたいかどうかが尋ねられます。

CONFIRM=YES では、確認の表示が強制されます。 CONFIRM=NO では、確認の表示が抑止されます。

デフォルトは CONFIRM=YES です。 DSQCONFIRM 変数を、START コマンドのパラメーターとして設定するかまたは開始プロシージャーで設定することによって、このデフォルトを変更することができます。

COMMENT=メンバーのコメント

コメント・オプションは、書式、プロシージャー、または Query オブジェクトのエクスポート時にメ

EXPORT

ンバー・テキストを指定するために使用します。コメントは、オブジェクトに関する情報を保管しておきたい場合に便利です。通常、コメントには間にブランクが含まれているため、コメント全体をアポストロフィで囲む必要があります。コメント内にアポストロフィがある場合は、2 つの連続アポストロフィを使用しなければなりません。

例:

```
COMMENT='SALES QUERY'
COMMENT='THIS QUERY DOESN'T INCLUDE SALES'
```

Query 管理機能でのコメントの最大の長さは、50 文字です。

Query 管理機能での EXPORT の CCSID に関する考慮事項

Query 管理機能プログラム、書式、またはプロシージャーがエクスポートされ、ソース・ファイルが存在しない場合には、ジョブの CCSID を用いてソース・ファイルが作成されます。書式 CCSID のためのデータは、ソース・ファイルの CCSID に変換されます。ソース・ファイルが存在し、ソース・ファイルの CCSID が Query 管理機能プログラム、書式、またはプロシージャーの CCSID と異なっている場合には、Query、書式、またはプロシージャーの CCSID からソース・ファイルの CCSID にデータが変換されます。

Query 管理機能のプロシージャーがエクスポートされ、エクスポート元のファイルの CCSID がエクスポート先のファイルの CCSID と異なっている場合には、データは最初にジョブの CCSID に変換され、次にエクスポート先のファイルの CCSID に変換されます。このような余分な変換を避けるためには、プロシージャーをエクスポートする代わりにコピーしてください。

Query 管理機能での EXPORT の例

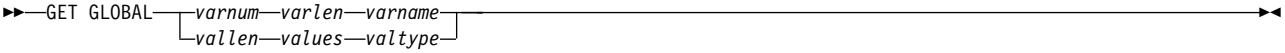
```
EXPORT QUERY SAMP1 TO SAMP1EX

EXPORT FORM EXLIB/EX1 TO EXLIB/FILE(EX1F) (CONFIRM=YES
```

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

Query 管理機能での GET

GET コマンドは、Query 管理機能変数の値を入手して、それをユーザー・プログラムに渡したい場合に使用します。プログラムから GET コマンドを使用する場合には、以下のコマンド構文を使用しなければなりません。



GLOBAL

Query 管理機能では、グローバル変数プールに入っている変数の変数名が要求元に戻されます。変数がグローバル・プールで見つからなかった場合には、エラー・メッセージが戻されます。

拡張パラメーター・リスト:

varnum

この呼び出しで要求された変数名の数。

varlen

指定されるそれぞれの変数名の長さ。

varname

Query 管理機能の変数プールの中にある変数の名前。

vallen

変数名値を含むプログラム・ストレージの長さ。

values

変数名値を含むプログラム・ストレージ域。

valtype

変数名値を含むストレージ域のデータ・タイプ。

Query 管理機能での GET の例

プログラムの中で GET コマンドを使用する場合の例については、 『第 7 章 Query 管理機能での呼び出し可能インターフェース』を参照してください。

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

Query 管理機能での IMPORT

IMPORT を使用して、エクスポートされたオブジェクトが入っているファイルをQuery 管理機能の FORM、 PROC、または QUERY のいずれかのオブジェクトにコピーすることができます。 IMPORT コマンドは外部ファイルに影響を与えません。

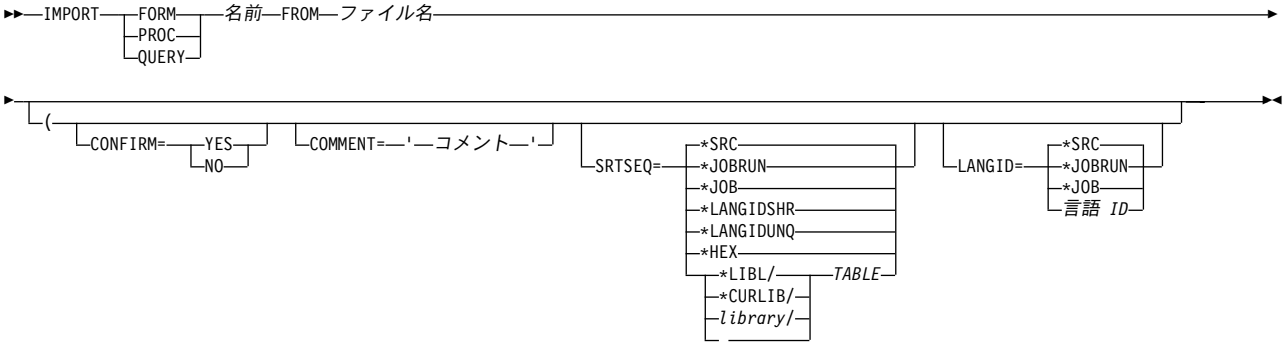
名前

インポートしたい FORM、PROC、または QUERY の名前。修飾名を指定することもできます。

ファイル名

Query 管理機能で読み取るべきシステム・ファイル (すなわち、インポートされるオブジェクトのためのソース・ファイル) の名前。

複数のシステムで一貫性のあるものにするためには、修飾名または拡張子のない複数のファイル名を指定しないでください。システム・デフォルトを使用してください。



IMPORT

CONFIRM=YES | NO

このオプションによって、IMPORT 要求を実行する前に確認することができます。確認要求は、データベース内の既存のオブジェクトが置き換えられようとする時にだけ行われます。保留中のデータベース変更を行いたいかどうかが尋ねられます。

ジョブが対話式で実行されている場合、照会メッセージがユーザーのディスプレイ装置に送られ、ユーザーがそのメッセージに応答するまでジョブは延期されます。メッセージは、オブジェクトをインポートしたいかどうかを尋ねてきます。ジョブをバッチ・モードで実行している場合、あるいは START 処理時に DSQSMODE がバッチにセットされていた場合には、 CONFIRM=YES の結果はエラーとなります。

CONFIRM=YES では、確認の表示が強制されます。 CONFIRM=NO では、確認の表示が抑止されます。

デフォルトは CONFIRM=YES です。 DSQSCNRM 変数を、START コマンド・パラメーターとして設定するか、または開始プロシージャーで設定することによって、このデフォルトを変更することができます。

COMMENT=コメント

コメント・オプションは、書式、プロシージャー、または Query オブジェクトのエクスポート時にメンバー・テキストを指定するために使用します。コメントは、オブジェクトに関する情報を保管しておきたい場合に便利です。通常、コメントには組み込みブランクが含まれているため、コメント全体をアポストロフィで囲む必要があります。コメント内にアポストロフィがある場合は、2 つの連続アポストロフィを使用しなければなりません。

例:

```
COMMENT='My form'
COMMENT='This form doesn''t include breaks'
```

Query 管理機能でのコメントの最大の長さは、50 文字です。

SRTSEQ

この Query に使用するソート・シーケンスを指定します。ソート・シーケンスは、Query を実行する時にどのようなソート・シーケンス表を使用するかを決定します。 SRTSEQ パラメーターは、Query を作成する場合にのみ使用することができます。書式を作成する場合には、SRTSEQ パラメーターは使用できません。可能な値は、次のとおりです。

*SRC

Query の作成時に用いられたソート・シーケンスの仕様が使用されます。ソース・ファイル・メンバーに SRTSEQ が指定されていない場合、 SRTSEQ のデフォルトは *JOBRUN です。

*JOBRUN

Query を実行する時点のジョブに関連する SRTSEQ を使用します。

*JOB

Query を作成する時点のジョブに関連するソート・シーケンスを使用します。

*HEX

ソート・シーケンスを判別するために、文字の 2 進値が使用されます。

*LANGIDSHR

ソート・シーケンスとして、各文字に*共用の* 重みを付けたシステム提供の表を使用します。使用される表は、LANGID パラメーターに指定された値によって決定されます。

*LANGIDUNQ

ソート・シーケンスとして、ある文字に*固有の* 重みを付けたシステム提供の表を使用します。使用す

る表は、LANGID パラメーターに指定した値によって決定されます。 SRTSEQ=*LANGIDUNQ で、しかも LANGID パラメーターが指定されていない場合、 LANGID のデフォルトは *JOBRUN です。

表名

Query の実行時点で使用するソート・シーケンスが入っている外部ソート・シーケンス表オブジェクトを使用します。

LANGID=*SRC | *JOBRUN | *JOB | 言語 ID

その Query に関連づける言語 ID。この時点で、このパラメーターの唯一の用途は、Query の実行時点で、 IBM 提供のどのソート・シーケンス表を使用するかを決定することです。これは、SRTSEQ=*LANGIDUNQ または SRTSEQ=*LANGIDSHR の場合にのみ使用することができます。SRTSEQ の値に関係なく、LANGID の値は検証され、作成される Query には指定された LANGID の値が入ります。

***SRC**

Query の作成時に使用する言語 ID の指定が入ります。ソース仕様に LANGID が指定されていない場合には、デフォルトの *JOBRUN が使用されます。

***JOBRUN**

Query の実行時点で使用する言語 ID は、Query を実行する時点で決定されます。

***JOB**

Query の実行時点で使用する言語 ID は、Query が作成される時点で決定されます。

言語 ID

3 文字の言語 ID。

IMPORT コマンドが使用される代表的な状況は、次のような場合です。

- Query 管理機能がインストールされているあるシステムから別のシステムに、FORM、 PROC、または QUERY のオブジェクトをコピーまたは伝搬する場合 (オブジェクトは送信側のシステムではエクスポートされ、受信側のシステムではインポートされます)。
- Query 管理機能の外で全機能エディターを使用する場合。通常、次のように行われます。
  1. まず最初に、Query 管理機能オブジェクトをエクスポートします。これによって、外部ファイルの作成が行われます。
  2. 次に、テキスト・エディターを呼び出します。エディター内に入れば、コピーおよび移動のような通常の編集機能を実行することができます。
  3. 編集を終了したら、Query 管理機能に戻り、ファイルをインポートします。 編集されたオブジェクトがデータベースに保管されます。
- アプリケーション・プログラマーが変更および対話式処理 (テスト) の目的で、通常は Query 管理機能外に常駐するプログラム・ライブラリーから、Query 管理機能内のライブラリーに SQL Query をマイグレーションしたい場合。

IMPORT コマンドは、指定されたファイルの内容をデータベースにコピーします。 SQL Query およびプロシージャーの場合には、ファイル内の各レコードがオブジェクトの中で別々の行になります。 Query 管理機能の EXPORT コマンドを使用してエクスポートされたファイルは、すべて、インポートすることができます。

SQL Query およびプロシージャーが入っているファイルをインポートする場合、 DB2 UDB for iSeries Query 管理機能は、論理レコード長が 79 を超えるレコードを受け入れますが、結果としてデータは切り捨てられることになります。 79 より大きい論理レコード長を見つけると、Query 管理機能は警告メッセージを表示します。

IMPORT

インポート Query に固定レコード形式 (および 79 より大きい論理レコード) が含まれている場合は、Query 管理機能は、1〜79 桁目のデータだけを受け入れ、残りのデータは無視します。

インポート書式に固定レコード形式 (および 150 より大きい論理レコード) が含まれている場合は、Query 管理機能は、1〜150 桁目のデータだけを受け入れ、残りのデータは無視します。

論理レコード長が 79 より小さい Query オブジェクトをインポートする場合は、 DB2 UDB for iSeries Query 管理機能はそのレコードの 79 桁目までをブランクで埋め込みます。 行に DBCS 混用の区切り文字付きストリングが含まれていると、埋め込みが区切り文字付きストリング内に組み込まれるため、予想外の結果となることがあります。

論理レコード長が 150 より小さい書式オブジェクトをインポートする場合は、Query 管理機能はそのレコードの 150 桁目までをブランクで埋め込みます。 行に DBCS 混用の区切り文字付きストリングが含まれていると、埋め込みが区切り文字付きストリング内に組み込まれるため、予想外の結果となることがあります。

SQL QUERY および PROC オブジェクトをインポートする場合、Query 管理機能は、ファイルの内容について、妥当性検査もセマンティック検査も行いません。 したがって、表示不能文字が入っている Query および PROC オブジェクトが設定される可能性があります (プログラムのオブジェクト・ファイルが QUERY としてインポートされた場合に起こることがあります)。 また、SQL ステートメントが PROC オブジェクトにインポートされたり、その逆の場合が生じる可能性もあります。 Query 管理機能には内容の “再カテゴリー化” の規定がないので、このような誤りはユーザー自身の責任で回避しなければなりません。

Query 管理機能は FORM オブジェクトの妥当性検査を行います。ファイルの一部に妥当性テストに合格しない部分がある場合は、このオブジェクトはデータベースに入れられますが、警告メッセージが送られてきます。 ファイルは妥当性テストに合格しても、フォーマット設定に使用した時点で予想外の結果が生じる可能性があります。

Query 管理機能での IMPORT の CCSID に関する考慮事項

Query 管理機能プログラムまたは書式がインポートされる場合には、Query または書式のインポート元のファイルの CCSID が付加されます。 Query 管理機能プロシージャーがインポートされ、ファイルが存在していない場合には、ジョブの CCSID を用いてファイルが作成されます。 Query 管理機能プロシージャーがインポートされ、該当するファイルが存在する場合には、プロシージャーはジョブの CCSID に変換されてから、インポートされるファイルの CCSID に変換されます。この余分な変換を避けるためには、プロシージャーをインポートする代わりにコピーしてください。

Query 管理機能での IMPORT の例

```
IMPORT FORM REPORT1 FROM REPT1EX

IMPORT QUERY SALARYWK FROM JENSON
```

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

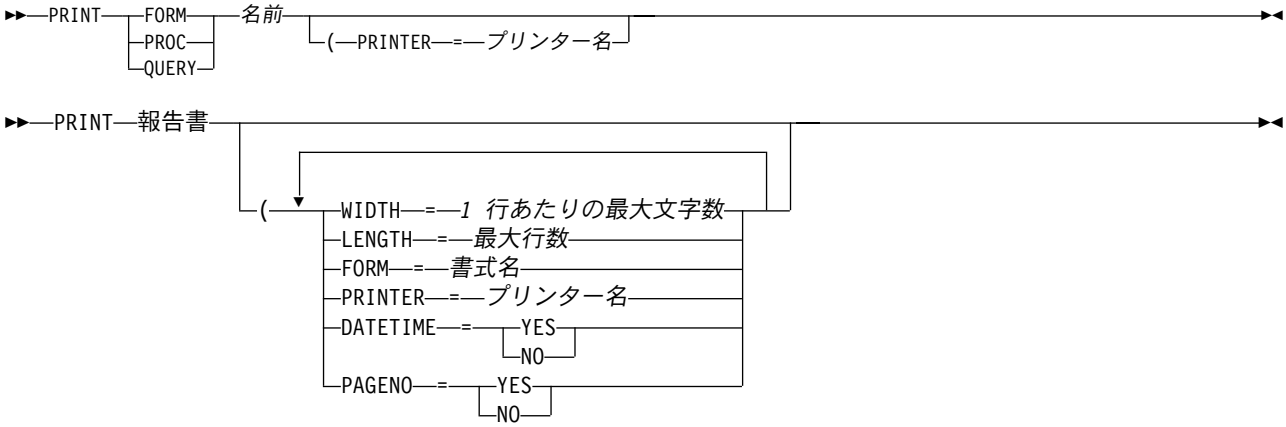
Query 管理機能での PRINT

PRINT コマンドは、Query 管理機能オブジェクトのハード・コピー・リストを印刷するために使用されます。 *QMQRY オブジェクトに関連するソート・シーケンス表の内容を印刷することはできません。ただし、Query に関連したソート・シーケンス表の名前またはソート・シーケンスの値は印刷されます。

PRINT コマンドは、印刷の標準システム機能を使用します。 Query 管理機能は、プリンター属性をご使用のアプリケーションに外部化しません。また、Query 管理機能では、それらの属性値を変更することもありません。その代わりに、現在有効なプリンター定義が使用されます。

オブジェクトの印刷形態は、その表示形態と非常によく似ています。ただし、画面形式と印刷形式との間には次のような相違点があります。

- 表示された REPORT オブジェクトの “パネル・タイトル” 行は、印刷報告書では各ページの最上部で “ページ見出し” に置き換わります (ページ見出しは定義済みと仮定します)。
- 印刷報告書では “ページ・フッター” は各ページの最下部にありますが、表示オブジェクトでは最下部に 1 回だけ示されます。



名前

印刷したいオブジェクトの名前。 指定できる名前はデータベースの中の FORM、 PROC、または QUERY です。

指定した書式または Query が見つからず、START コマンドに DSQSCNVT=YES が指定されている場合は、Query 管理機能は、その名前をもつ Query for iSeries 定義を探索します。 Query 定義が見つかれば、その情報を使用して、Query 管理機能が使用可能な一時 Query または一時書式が作成されます。

WIDTH=*1 行あたりの最大文字数*

値は 22～378 の整数でなければなりません。

報告書が印刷 WIDTH よりも広い場合、ページ間分割されて次のページにまたがります。報告書以外の Query オブジェクトが、ページ間分割されて次のページにまたがることはありません。オブジェクトが印刷幅よりも広い場合、印刷出力の行の右側が切り捨てられます。

WIDTH と使用しているプリンターとの互換性を確保することが重要です。たとえば、現行のプリンターの設定値が、8.5 インチ幅の用紙を装着した 10 ピッチ装置 (10 文字 / インチ) であることを示す場合、 WIDTH 値が 132 であると出力が変更されることになります。正確な結果は、プリンターのハードウェアおよびソフトウェアによって変わることがあります。 Query 管理機能では物理的なプリンターの幅が分からないので、このような状態が起きても、特別なメッセージは表示されません。

PRINT

このオプションを指定しないと、Query 管理機能は対応するシステム・デフォルトまたはユーザー・デフォルトを使用します。このデフォルトが使用できない場合には、デフォルトは 80 にセットされます。

LENGTH=*1* ページあたりの最大行数

値は 1〜999 の整数でなければなりません。

報告書を印刷しようとして LENGTH の値が不十分な場合 (LENGTH の値が、列見出し、ページ見出しおよびフッターに必要な行数とページ番号または日付および時刻、あるいはその両方を印刷するために必要な行数を加えた合計行数より小さい場合)、エラー・メッセージが生成されて報告書は印刷されません。

LENGTH の値が許容範囲内にあれば、ページ上に印刷された報告書データの行数が LENGTH に等しくなると、Query 管理機能は必ず改ページを行います。

このオプションを指定しないと、Query 管理機能は対応するシステム・デフォルトまたはユーザー・デフォルトを使用します。このデフォルトが使用できない場合には、デフォルトは 66 にセットされます。

FORM=*書式名*

データのフォーマット設定に使用したい FORM の名前。書式を指定しなかった場合は、前の RUN QUERY で用いられた書式が使用されます。

PRINTER=*プリンター名*

出力を生成するプリンターの名前。

Query 管理機能は、OS/400 の OVRPTRF (プリンター・ファイルの一時変更) CL コマンドを使用して、別のプリンター・ファイルを宛先とすることができます。このコマンドで Query 管理機能プリンター・ファイルを永久的に変更することはできません。ただし、PRINT コマンドを再び実行してデフォルトのページ長およびページ幅値を使用したい場合には、 CHGPRTF CL コマンドを使用してプリンター・ファイルを永続変更してください。

DATETIME=YES | NO

このオプションは、各ページの最下部にシステム日付および時刻の生成および表示を行うかどうかを制御します。 DATETIME=YES を指定した場合、日付および時刻が各ページの最後の行に入れられます。 DATETIME=NO を指定した場合、システム日付および時刻は印刷されません。このオプションのデフォルトは YES です。

PAGENO=YES | NO

このオプションは、各ページの最後の行にページ番号の印刷を行うかどうかを制御します。このオプションのデフォルトは YES です。

OS/2 では、このオプションのデフォルトはユーザーのプロファイルから得られます。

注: PRINT REPORT コマンドに関連するオプションのいずれか、またはすべてを使用することができますが、各オプションはそれぞれ 1 回しか使用できません。同じオプションを 2 回以上使用した場合、最後に指定したオプションが使用されます。

Query 管理機能での PRINT の CCSID に関する考慮事項

Query 管理機能プログラムが印刷される場合には、Query に付加された CCSID からジョブの CCSID に変換されます。 Query 管理機能書式が印刷される場合には、CCSID 変換は行われません。 Query 管理機能報告書が印刷される場合には、データはジョブの CCSID となります。 PRINT REPORT コマンドで書式が使用される場合には、CCSID 変換は行われず、書式の部分は認識されないことがあります。 Query 管理機能プロシージャーが印刷される場合には、ジョブの CCSID に変換されます。

Query 管理機能での PRINT の例

```
PRINT QUERY querynname

PRINT FORM formname

PRINT PROC procname

PRINT REPORT

PRINT REPORT (WIDTH=80 LENGTH=60 DATETIME=YES PAGENO=YES

PRINT QUERY Q1 (PRINTER=PRT1

PRINT PROC LIBA/PROCA(MBRA)
```

Query 管理機能でのプリンター・ファイルの使用

Query 管理機能の一部として、QPQXOBJPF および QPQXPRTF と呼ばれるデフォルトのプリンター・ファイルが QSYS ライブラリーに含まれています。これらのプリンター・ファイルは、PRINT QUERY、PRINT PROC、または PRINT REPORT コマンドが出された時に使用されます。プリンター・ファイル QPQXOBJPF は、ページの長さと幅がそれぞれ 66 行と 132 文字のデフォルトを持っています。プリンター・ファイル QPQXPRTF は、ページの長さと幅がそれぞれ 66 行と 80 文字のデフォルトを持っています。プリンター・ファイルによって指定された印刷装置名は *JOB であり、これは、すべてのプリンター出力をジョブに合わせてセットアップされたプリンターへ出力することができるようにします。プリンター・ファイルの QPQXOBJPF および QPQXPRTF は、一時変更されていなければ、印刷されるオブジェクトおよび報告書のフォーマット設定をするために、Query 管理機能によって使用されます。

PRINT コマンドに値を指定するか、あるいは START コマンドで DSQAPRNM のデフォルトを *SAME から印刷装置名または *JOB に変更することによって、Query 管理機能に対して別のプリンターを使用するように指示することができます。

また、プリンター・ファイル一時変更 (OVRPRTF) CL コマンドを使用すれば、Query 管理機能に別のプリンター・ファイルを使用するように指示することができます。

| プリンター・ファイル変更 (CHGPRTF) CL コマンドを使用して、Query 管理機能のプリンター・ファイル
| である QPQXOBJPF および QPQXPRTF を永続的に変更することができます。デフォルトを再び使用する
| ためには、別の CHGPRTF CL コマンドを出して属性を変更し直してください。

| プリンター・ファイルは、インストールのたびに QSYS ライブラリーの中で作成し直されます。したがっ
| て、プリンター・ファイルに対する変更はすべて適用し直さなければなりません。プリンター・ファイルに
| 対する変更を保管するためには、所要の属性を持つユーザー独自のプリンター・ファイルをユーザー・ライ
| ブラリーに作成し、OVRPRTF CL コマンドを使用して、このプリンター・ファイルを使用するように
| Query 管理機能に指示することができます。プリンター・ファイルをユーザー独自のライブラリーにコピー
| して、そのライブラリーの中のコピーを変更することもできます。 QSYS ライブラリーにあるファイルと
| 同じ名前のプリンター・ファイルを使用するには、そのユーザー・ライブラリーが、ライブラリー・リスト
| の中で QSYS ライブラリーより前にあることが必要です。

Query 管理機能での PRINT オブジェクトのフォーマット設定

PRINT QUERY および PRINT PROC コマンドの処理中に、 Query 管理機能は印刷出力を 132 文字の行にフォーマット設定します。この行は 123 バイトのテキストと、 PRINT コマンドの実行時に生成される 7 バイトの行番号に分けられます。

PRINT

132 の幅は大部分のファイルの印刷を容易に処理できる広さであり、大部分の iSeries プリンターとも互換性があります。

プリンター・ファイルのレコードの折り返しパラメーターに *YES が指定されていない場合には、行の幅が 132 文字より小さいプリンターへプリンター出力を出した時に、データが失われる可能性があります。プリンター・ファイル QPQXOBJPF のレコードの折り返しパラメーターのデフォルトは *NO です。

Query 管理機能での PRINT 報告書のフォーマット設定

PRINT REPORT コマンドの処理中に、Query 管理機能は PRINT コマンドに指定された幅またはプリンター・ファイルからのデフォルトを使用して、印刷報告書をフォーマット設定します。報告書の幅が印刷の WIDTH より広い場合には、それが複数のページに分割されます。この場合には、複数のプリンター・ファイルがオープンされ、各報告書行のそれぞれのセグメントはオープンされた開かれている適切なプリンター・ファイルへ出力されます。

幅が PRINT コマンドまたはプリンター・ファイルに指定された WIDTH より狭いプリンターへ報告書が出力された場合には、各印刷レコードは切り捨てられます。プリンター・ファイルのレコードの折り返しオプションがデフォルトから *YES に変更された場合には、各印刷レコードが折り返されます。たとえば、印刷される時に PRINT REPORT (WIDTH=200 PRINTER=xyz) のコマンドによって 200 桁にフォーマット設定される報告書であれば、プリンターの幅が 200 より狭い場合には行の折り返しが行われます。プリンター・ファイルのレコードの折り返しオプションは、*YES に一時変更されています。レコードの折り返しオプションが一時変更されない場合には、報告書の右端の数桁が切り捨てられます。

Query 管理機能は、Query が、以下に対する報告書のフォーマット設定を行うために実行された場合、有効なソート・シーケンス表を使用します。

- 計算
  - MIN
  - MAX
- 次のデータ・タイプに対する BREAKn レベルの処理
  - CHAR
  - VARCHAR
  - DBCS 混用
  - DBCS 択一

注: Query 管理機能は、 GRAPHIC および VARGRAPHIC データ・タイプに対する BREAKn レベルの処理のための MIN および MAX の計算に対し、ソート・シーケンスの使用をサポートしません。ソート・シーケンスは、DBCS データには適用されません。その代わりに、比較には、DBCS データの各バイトの 2 進数値が使用されます。

ソート・シーケンスの使用の詳細については、245 ページの 『付録 D. Query 管理機能でのソート・シーケンスの例』を参照してください。

Query 管理機能での RELEASE

RELEASE コマンドは、1 つまたは複数の接続を解放状態に置きます。解放状態は、次の正常なコミット操作で、その接続が切断されることを意味します。ロールバックは接続に影響しません。解放状態は、切断の保留中と見なすことができます。

RELEASE コマンドが正常に実行されると、指定された各接続は解放状態に置かれ、次の正常なコミット操作で切断されます。

RELEASE コマンドが正常に実行されなかった場合は、活動化グループの接続状態およびその接続の状態は変わりません。

接続の作成と維持には、リソースが使用されます。再利用する予定のない接続は、解放状態に入れてください。後続の作業単位で必要な接続は解放してはなりません。

コミット操作が首尾よく実行された時に現行接続が解放状態であれば、その接続は除去され、その活動化グループは非接続状態のままになります。このような場合は、次の Query 管理機能コマンドは、 CONNECT または SET CONNECTION コマンドでなければなりません。 Query 管理機能の方では、現行の接続が除かれたことは分かりません。 CONNECT または SET CONNECTION の前に出された SQL ステートメントは、すべて失敗します。

rdbname
アプリケーション・サーバーとして実行するデータベースのグローバル固有名。これは、アクセスするリレーショナル・データベースを指定します。リレーショナル・データベースは、データベースを識別する手段であり DRDA サポートによりアクセスできます。 *rdbname* は、次のようにすることができます。

- 長さは 18 文字までです。
- 大文字 (A〜Z)、数字 (0〜9)、および下線 (_) だけで構成します。先頭の文字は、大文字でなければなりません。

CURRENT
活動化グループにある現行接続を識別します。

ALL
活動化グループにあるすべての接続を識別します。

RELEASE の例

次の作業単位では、RDBONE への接続が必要ないとします。次のコマンドを使用すると、次に実行される正常なコミット操作でその接続が除去されます。

```
RELEASE RDBONE
```

次の作業単位では、現行の接続が必要ないとします。次のコマンドを使用すると、次に実行される正常なコミット操作でその接続が除去されます。

```
RELEASE CURRENT
```

次の作業単位では、既存の接続が必要ないとします。次のコマンドを使用すると、次に実行される正常なコミット操作でその接続が除去されます。

```
RELEASE ALL
```

Query 管理機能での RUN

RUN コマンドは PROC または QUERY を処理します。 RUN コマンドを出す場合、処理したい PROC または QUERY を指定する必要があります。したがって、RUN コマンドを出す前に、 PROC または QUERY がデータベース内に存在していなければなりません。

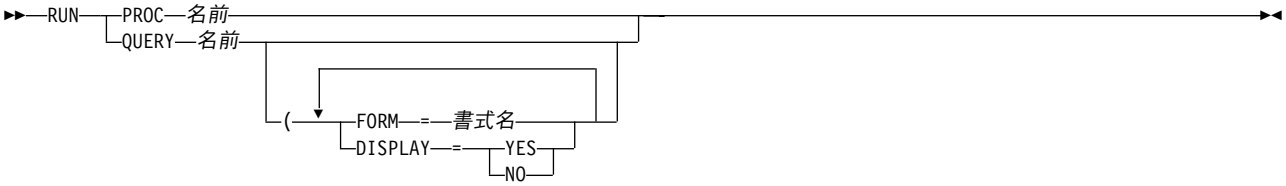
RUN コマンドは、QUERY (SELECT のみ) を処理するために使用すると、新しいデータを作成し、前の RUN QUERY によって作成されていた既存のデータを置き換えます。

対話方式で実行する時は、Query (SELECT のみ) の結果が表示されます。バッチ・モードで実行すると、結果は表示されません。

参照制約付きで定義されるフィールドを変更する Query を使用する時は、制約検査が行われ、違反エラーが起きる可能性があります。また、Query 管理機能が依存ファイルのフィールドを変更または参照する時、または親ファイルのフィールドを変更する時に、検査保留エラーが起こることがあります。

通常、QUERY では実行時に FORM を使用します。これには次の 2 つの方法があります。

1. 既存の FORM の名前が、FORM オプションを介して RUN コマンドで明示的に指定される。
2. デフォルト FORM が作成される。このデフォルトの書式は、DATA の列属性を考慮に入れた規則を用いて作成されます。



名前
実行しようとする QUERY の名前。

FORM=*書式名*
このオプションは、SELECT ステートメントを含む Query の場合にだけ意味があります。

FORM オプションは、対話方式で実行時に RUN が自動的に表示する REPORT のフォーマット設定で使用する FORM を指定します。

FORM オプションで指定した書式が見つからないと (たとえば、FORM が存在していない場合)、RUN コマンドは拒否され、エラー・メッセージが出されます。書式は存在するが、その DATA を*処理しない* 場合は (その列のデータ・タイプに異なるタイプが指定されていると考えられます)、Query 管理機能はエラー・メッセージを出します。

FORM オプションを省略した場合は、デフォルトの FORM が使用されます。

指定した書式または Query が見つからず、START コマンドに DSQSCNVT=YES が指定されている場合は、Query 管理機能は、その名前をもつ Query for iSeries 定義を探索します。 Query 定義が見つかれば、その情報を使用して、Query 管理機能が使用可能な一時 Query または一時書式が作成されます。

DISPLAY=YES | NO
DISPLAY キーワードは、報告書を表示するかどうかを示すのに使用します。対話式で処理している場合、このキーワードのデフォルトとして YES が使用されます。 START コマンドにバッチ方式が指定されている場合には、このキーワードは無視されます。

このコマンドの拡張パラメーター・リスト様式を使用することができます。この様式の詳細については、30 ページの『Query 管理機能での GET』の項を参照してください。

Query 管理機能での RUN の考慮事項

Query 管理機能での RUN の日付形式の使用

Query 管理機能プログラムにジョブの日付形式と異なる日付形式が含まれていて、Query にグローバル変数が含まれている場合、SQL ステートメントを解釈するために使用される日付形式は SQL 日付形式です。これは、SQL ステートメント中の日付リテラルの解釈時に、ジョブ日付形式か Query 日付形式のいずれかを選択しなければならないことを避けるためです。

Query 管理機能での RUN の CCSID の使用

Query 管理機能は、その Query に特化されている CCSID で解釈されます。ジョブの CCSID とは異なる CCSID 内にあるファイルからデータが取り出される時には、データを表示または印刷する場合、データはジョブの CCSID に変換されます。データがデータベース・ファイルに保管される時には、ファイルがすでに存在する場合にはデータがファイルの CCSID に変換され、新しいファイルが作成される場合にはデータが取り出されたファイルの CCSID にデータが変換されます。データが保管されていても表示されない場合には、新しいファイルが作成されても CCSID 変換は行われません。元のファイルの CCSID からデータの保管先のファイルに対するジョブの CCSID へデータを変換するのを避けるためには、 SAVE DATA AS コマンドを実行する前に、 RUN QUERY コマンドで DISPLAY=NO パラメーターを指定してください。

ジョブとは異なる CCSID のファイルを使用して作成された書式を使うと、テキスト・フィールドは変換されません。この結果として、報告書には認識できないテキストが表示されることがあります。報告書に正しいテキストを表示するためには、ジョブと同じ CCSID を持つファイルに書式をエクスポートしてから、その書式をインポートし直してください。

Query 管理機能での CALL SQL RUN の制約

Query 管理機能を介して CALL SQL ステートメントを実行し、しかも呼ばれたプログラムがローカルで実行される場合、呼び出されたプログラムは次のものを使用することはできません。

- Query 管理機能 CPI インターフェース
- Query 管理機能 CL コマンド
- SQL Query Manager CL コマンド

Query 管理機能は再帰的に実行することはできません。ただし、呼び出されたプログラムをリモート側で実行する場合は、呼び出されたプログラムはリモート・システムで Query 管理機能を呼び出すことができます。出力パラメーターを指定して定義されているプロシージャーを呼ぼうとすると、CALL SQL ステートメントは SQL0469 エラーを伴って失敗します。これは実行時エラーであり、Query 定義エラーではありません。

Query 管理機能での RUN の例

```
RUN QUERY QN1

RUN PROC WEEKREPT

RUN QUERY SMITH.Q6 (FORM=SMITH.SAL_REPT
```

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

Query 管理機能での SAVE

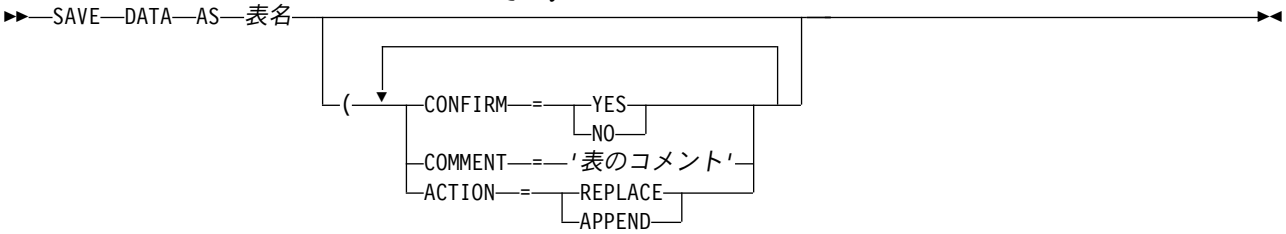
データベースの中の表にデータを保管するためには、SAVE コマンドを使用してください。保管された TABLE は、コマンドで指定した名前にしたがって名前が付けられます。

DATA が保管され、TABLE/VIEW が実際に置き換えられる場合、データは既存の定義と互換性がなければなりません。互換性のあるデータとは、データ・タイプ、長さ、およびヌルの属性が一致しているデータのことです。特に、DATA の列数がターゲットに一致し、列は互換性のあるデータ・タイプとヌル属性を備えていなければなりません。 2 つのオブジェクトに互換性がない場合、Query 管理機能は SAVE コマンドを拒否し、データベースは変換されません。

SAVE コマンド上の名前がデータベースのビューとしてすでに存在している場合、そのビューが定義されている表は、ビューを用いて表を更新する規則にしたがって変更されます。

まだ存在しない保管表の列名は、 FORM オブジェクトのデフォルト見出しを生成するときに使用するのと同じアルゴリズムを使って、Query 管理機能が作成します。列名はユーザーが変更することはできません。

2 バイト文字セット (DBCS) グラフィック・データ (DBCS グラフィック・データ・タイプ) での SAVE コマンドの使用については、 『付録 A. Query 管理機能での DBCS データ』を参照してください。



DATA
前に実行している QUERY からの実際の結果を参照します。

表名
データベースでデータが保管される表またはビューの名前。通常、この名前は非修飾名です。

この表名は、iSeries システムの命名規則に合致していなければなりません。 SQL のサポートするロング・ネームおよび特殊文字を使用した名前は、SAVE DATA AS コマンドではサポートされません。

指定された表名がデータベースに存在していない場合には、新しい表が作成されます。この名前は、library/object または database.object の形式の修飾名にすることができます。使用する予定の命名規則を最初の Query コマンド・プロシージャーに指定してください。

データを表に保管するためには、表を変更または作成するための適切な SQL 権限を持っていることが必要です。表の許可規則については、SQL 解説書を参照してください。 ACTION=REPLACE が指定されている場合、物理ファイル・メンバー消去 (CLRPFM) CL コマンドに対する権限も持っていなければなりません。

SQL の規則では、ビューが 1 つの表とだけ関連している場合にのみ、それを更新することができます。複数の表に基づくビューの更新は許されていません。また、SQL カタログである表と関連したビューを更新することはできません。

CONFIRM=YES | NO
このオプションによって、SAVE 要求を実行する前に確認することができます。確認要求は、データベース内の既存のオブジェクトが置き換えられようとする時にだけ行われます。保留中のデータベース変更を行いたいかどうかが尋ねられます。

CONFIRM=YES では、確認の表示が強制されます。 CONFIRM=NO では、確認の表示が抑止されます。どちらの場合も、SAVE 操作が完了したことを通知する確認メッセージが生成されます。

デフォルトは CONFIRM=YES です。 DSQCONFIRM 変数を、START コマンドのパラメーターとして設定するかまたは開始プロシージャーで設定することによって、このデフォルトを変更することができます。

COMMENT='*表のコメント***'**

データを表として保管する場合、このオプションを使用してコメントを指定します。コメントは、オブジェクトに関する記述的な情報を保存する場合に有用です。

通常、コメントは複数の語とその間のブランクで構成されているため、アポストロフィで囲まなければなりません。コメントの一部として含まれるアポストロフィは、2 個の連続アポストロフィとして入力する必要があります。

例:

```
COMMENT='The master EMPLOYEE table-see John (X3971)'

COMMENT='Don't ERASE this data without telling Phil!'
```

Query 管理機能では、オブジェクトコメントの長さは、前後のアポストロフィを除いて最大 50 文字に制限されています。

ACTION=REPLACE | APPEND

ACTION=REPLACE オプションでは、既存の表またはビューが置き換えられることになります。 ACTION=APPEND オプションは、追加したいデータを既存の表またはビューの終わりに追加します。 デフォルトは ACTION=REPLACE です。表またはビューが存在していない場合には、ACTION キーワードは無視されます。

注: SAVE コマンドに関連したオプションはどれでも、またはすべてを使用することができますが、各オプションはそれぞれ 1 回しか使用できません。

Query 管理機能での SAVE の例

```
SAVE DATA AS EMP12

SAVE DATA AS EMP12 (COMMENT='CLASSIC TWO TABLE JOIN'
```

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、47 ページの『Query 管理機能での START』を参照してください。

Query 管理機能での SAVE のヌル値に関する考慮事項

フィールドおよび列にはヌル値を指定できるので、 SAVE DATA AS コマンドを使用してデータを保管する場合には、以下の事項に留意する必要があります。

- SQL 機能の結果として生じた列には、ヌル値を使用することができます。
- 報告書にグループ化を使用した SQL 機能からの列があり、レコードが選択されなかった場合には、その列の値はヌル値です。
- SQL 数値表現の結果として生じる列には、ヌル値を使用することができます。
- ヌル値可能フィールドに基づいて結果フィールドが作成された場合には、その結果フィールドにはヌル値を使用することができます。

- 結果フィールドの計算時にヌル値が見つかった場合、その結果フィールドの値はヌル値です。たとえば、連結したフィールドにヌル値が入っている場合には、その結果フィールドにはヌル値が含まれます。サブストリング操作で使用されるフィールドにヌル値が入っている場合には、その結果フィールドにはヌル値が入ります。

Query 管理機能での SAVE の参照制約に関する考慮事項

OS/400 の参照制約機能を使用する時は、SAVE DATA AS コマンドの使用において次の点に留意する必要があります。

- 選択した表が参照制約を持っていても、SAVE DATA AS コマンドで作成される出力表には、その制約は伝搬しません。
- すでに存在する表に Query の結果を出力する時に、その表を置換しようとし、しかもそれが制約関係を持つ基本表であると、SAVE DATA AS コマンドは失敗します。
- すでに存在する表に Query の結果を出力する時に、その出力表に参照制約があり、それに違反していると、SAVE DATA AS コマンドは失敗します。このエラーは、その表が変更された後にならないと、検出できません。たとえば、既存の表のデータを置換する時、その表は、新しいデータを挿入する前に消去されます。
- 設定済み / 使用可能化制約が検査保留になっている時に Query 管理機能が従属ファイルまたは親ファイルにデータを保管しようとすると、検査保留エラーとなる可能性があります。検査保留エラーは、既存の表を変更する前に見つけなければなりません。 SAVE DATA AS コマンド処理では、エラーが起きた時にファイルが変更されているかどうかを示します。
- 表のデータを置換するために SAVE DATA AS コマンドを使用する時は、注意が必要です。たとえば、レコードを表から削除する場合に、別の表の 1 つまたは複数のレコードも削除する参照制約を設定することができます。 SAVE DATA AS コマンドを実行する時に、そのデータを置換する場合は、ファイルが物理ファイルであれば物理ファイル・メンバー消去 (CLRPFM) コマンドが実行され、論理ファイルに対しては SQL DELETE が実行されます。いずれの場合も、実際に出力表にデータを挿入するには、SQL INSERT が行われます。

Query 管理機能での SAVE の長い列名に関する考慮事項

SAVE DATA AS コマンドにより出力表が作成される場合、選択された長い列名および特殊文字を含む列名は出力表に保持されます。選択された列の列名が別の列名またはシステム列名と同じである場合は、それは修飾されて、固有の名前になります。出力ファイルの列名がもとの列名と異なることを示す警告は出ません。たとえば、次のファイル A と B の Query を

表名	システム列名	列名
A	NAME	EMPLOYEENAME
B	EMPNAME	EMPLOYEENAME
B	CUSNAME	NAME

表 C に出力すると、以下の名前になります。

表名	システム列名	列名
C	NAME	EMPLOYEENAME
C	EMPNAME	EMPLOYEENAME1
C	CUSNAME	NAME2

Query 管理機能での SET CONNECTION

SET CONNECTION は、接続の状態を休止状態から現行状態に変更します。休止状態とは、接続が保留になっていることを意味します。接続が休止状態にあるときは、コミットとロールバック以外の SQL ステートメントはいずれもその接続を使用できません。現行状態とは、その接続が、出された SQL ステートメントで使用されることを意味します。活動化グループ 1 つに対し、接続は 1 つだけ現行状態になれます。SET CONNECTION が役に立つのは、DUW 接続管理方式のもとで実行している場合だけです。

SET CONNECTION コマンドが正常に行われると、次のようになります。

- 現行接続がある場合には、休止状態になります。
- 識別されたアプリケーション・サーバーは、現在の状態に入れられます。
- グローバル変数の DSQSDBNM と DSQCMTLV は更新されます。 DSQSDBNM には、現行データベースの名前が入ります。 DSQCMTLV には、使用されるコミットメント制御レベルが入ります。 Query 管理機能インスタンスの接続情報は、SQL CA からの必要な情報によって更新されて、現行状態を示します。

SET CONNECTION コマンドが正常に行われないと、活動化グループの接続状態およびその接続の状態は変わりません。インスタンス情報は更新されません。

rdbname
アプリケーション・サーバーとして実行するデータベースのグローバル固有名。これは、アクセスするリレーショナル・データベースを指定します。リレーショナル・データベースは、データベースを識別する手段であり DRDA サポートによりアクセスできます。 *rdbname* は、次のようにすることができます。

- 長さは 18 文字までです。
- 大文字 (A〜Z)、数字 (0〜9)、および下線 (_) だけで構成します。先頭の文字は、大文字でなければなりません。

Query 管理機能での SET CONNECTION の例

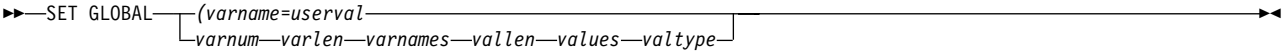
次の例は、最初のコマンドの後で SQL ステートメントを RDBONE に送り、 2 番目のコマンドの後で SQL ステートメントを RDBTWO に送り、さらに 3 番目のコマンドの後で SQL ステートメントを再び RDBONE に送ります。この例は、DUW 接続管理方式のもとで実行していることを前提としています。

```
CONNECT TO RDBONE
⋮
SQL ステートメント
⋮
CONNECT TO RDBTWO
⋮
SQL ステートメント
⋮
SET CONNECTION RDBONE
⋮
SQL ステートメント
```

最初の CONNECT コマンドで RDBONE 接続が作成され、それが現行状態になります。 2 番目の CONNECT コマンドで RDBTWO 接続が作成され、今度はそれが現行状態になり、 RDBONE 接続は休止状態に置かれます。次に、SET CONNECTION コマンドは、RDBONE を現行状態に戻し、RDBTWO を休止状態に置きます。

Query 管理機能での SET GLOBAL

SET GLOBAL コマンドは、ユーザーのプログラムまたはプロシージャーから Query 管理機能変数の値をセットするために使用します。プロシージャーから SET GLOBAL コマンドを使用するときは、コマンド構文は短バージョンを使用しなければなりません。プログラムから SET GLOBAL コマンドを使用するときは、コマンド構文は拡張バージョンを使用しなければなりません。



GLOBAL

Query 管理機能では、グローバル変数プールの中にある変数の変数名は要求元がセットします。変数が存在していない場合は、新しい変数が作成されます。変数が存在していた場合には、その内容が置き換えられます。

varname

Query 管理機能の変数プールの中にある変数の名前。呼び出し可能インターフェースを介して使用される変数名に適用される規則については、 88 ページの『Query 管理機能 CI での変数名』を参照してください。

userval

varname により指定される変数名に関連する値。この値がアポストロフィで囲まれている場合、アポストロフィは取り除かれます。

拡張パラメーター・リスト:

varnum

この呼び出しで要求された変数名の数。

varlen

指定されるそれぞれの変数名の長さ。

varnames

Query 管理機能の変数プールの中にある変数の名前。

vallen

変数名値を含むプログラム・ストレージの長さ。

values

変数名値を含むプログラム・ストレージ域。

valtype

変数名値を含むストレージ域のデータ・タイプ。

Query 管理機能での SET GLOBAL の例

次に示すのは、プロシージャーの中で使用される SET GLOBAL コマンドの例です。プログラムの中で使用される SET GLOBAL コマンドの例は、 『第 7 章 Query 管理機能での呼び出し可能インターフェース』に載せてあります。

```
SET GLOBAL (CHARVAR = 'abc'

SET GLOBAL (NUMBVAR = 199

SET GLOBAL (NAMEVAR = MYTABLE

SET GLOBAL (CHARVAR='abc'
```

Query 管理機能で SET GLOBAL を使用する際の *varname* 値での引用符

変数を短縮コマンドの構文でセットする場合、文字ストリング *varname* の値の中で引用符を表すためには、 2 つの連続した単一引用符を使用します。変数を拡張パラメーター・リスト形式によってセットする場合には、引用符を表すために文字ストリング *varname* の値の中で 1 つの単一引用符を使用してください。詳細については、『付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用』を参照してください。

コマンド・キーワードは、呼び出し可能インターフェースの拡張パラメーター・リストおよびコマンド・ストリングで使用できます。拡張パラメーター・リストの使用の詳細については、『Query 管理機能での START』を参照してください。

Query 管理機能での SET GLOBAL のプログラミング上の考慮事項

SET GLOBAL コマンドは、実行時レコードを選択する場合に役に立ちます。異なる SELECT フィールドまたは WHERE 条件を持つ多数の QMQRY オブジェクトを保管する必要はありません。

たとえば、アルファベットの前半部分の英字で始まる名前の社員に関するレコードが入っているファイルについて、Query を実行するとします。また、アルファベットの後半部分の英字で始まる名前の社員に関するレコードについても、同じ Query を実行するとします。この場合には、EMPREPORT という名前の Query オブジェクトに、次のような SQL SELECT ステートメントを含めることができます。

```
SELECT  NAME,DEPT,EMPNO  FROM  MASTER
        WHERE  NAME  =  &STARTAL  AND  NAME  <  &ENDAL
```

それから、次のようなステートメントによるプロシージャーをセットすることができます。

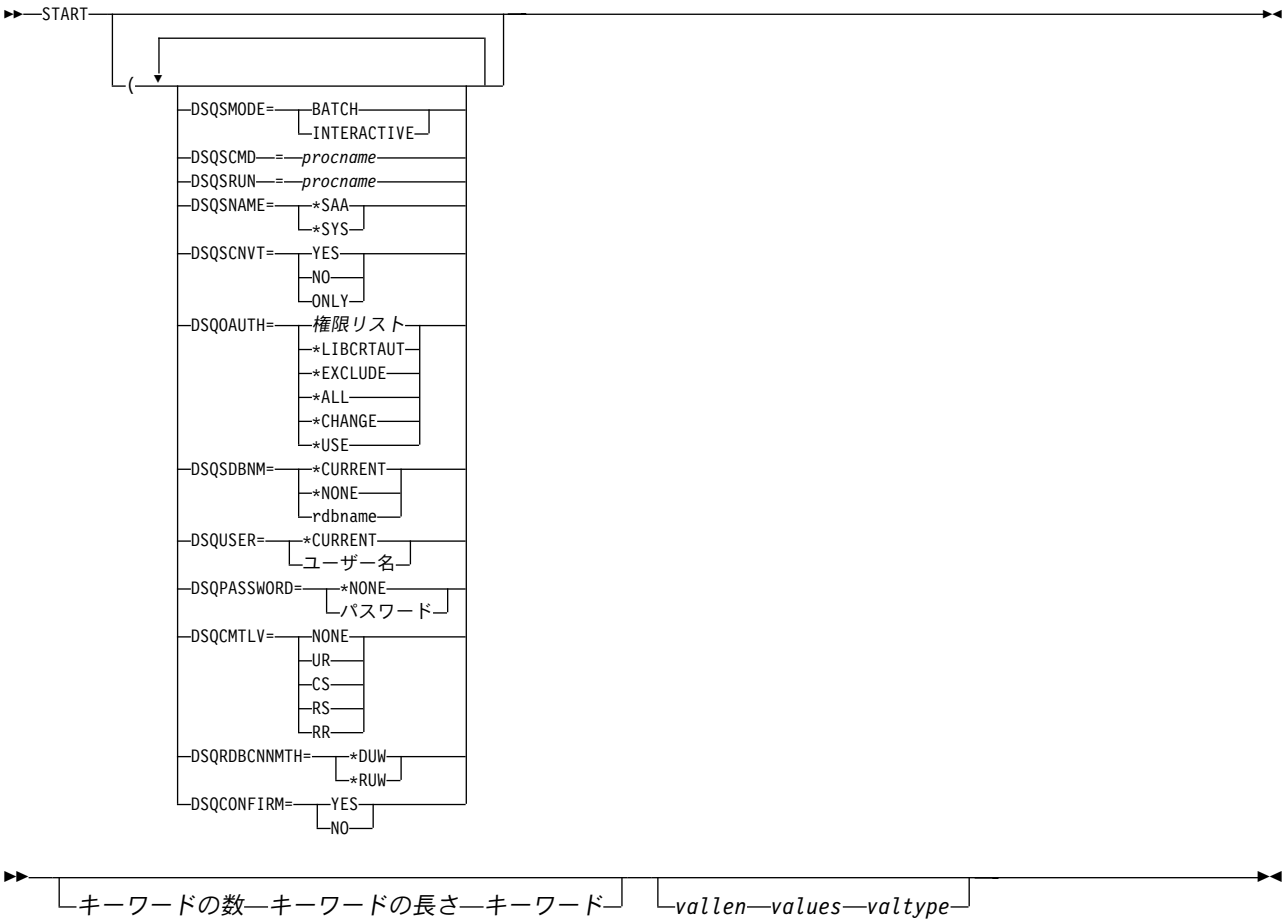
```
"SET  GLOBAL  (STARTAL='''A'''"
"SET  GLOBAL  (ENDAL='''J'''"
"RUN  QUERY  EMPREPORT"
"SET  GLOBAL   (STARTAL='''K'''"
"SET  GLOBAL   (ENDAL='''Z'''"
"RUN  QUERY  EMPREPORT"
```

EMPREPORT Query は、 Query 管理機能プログラムの開始 (STRQMQRY) CL コマンドによって対話方式で実行することもできます。 SELECT ステートメントが実行される前に、変数の STARTAL および ENDAL に関するプロンプトが出されます。

Query 管理機能での START

START コマンドは、Query 管理機能のインスタンスを開始するためのインターフェースを提供します。このコマンドは、呼び出し可能インターフェースを介して出された場合にのみ有効です。 START コマンドによって、Query 管理機能セッションを開始する方法を示す値を指定できます。

START



Query 管理機能での START の拡張パラメーター・リスト

キーワードの数
この呼び出しで渡されるキーワードの数。

キーワードの長さ
指定される各キーワードの長さ。

キーワード
セットされている START コマンドのキーワードの名前。

Query 管理機能の START コマンドでは、次のキーワードが使用されます。

- DSQSMODE は、後続のコマンドが出された時点での Query 管理機能の操作のモードを指定します。有効なオプションは次のとおりです。

  INTERACTIVE

  Query 管理機能の処理時に画面を表示できるようにします。 RUN QUERY コマンドの結果として生成された報告書は、いずれも画面に表示されます。応答を必要とする確認メッセージがあれば、応答できるように画面に表示されます。

  BATCH

  Query 管理機能の処理時に画面は表示されません。 応答を必要とするメッセージは、エラーになります。その他のすべてのメッセージは、ジョブ・ログに送られます。

START コマンドの DSQSMODE 変数にセットされたキーワード値は、 Query コマンド・プロシージャーを介して DSQSMODE 変数にセットされているキーワード値を一時変更します。

- DSQSCMD は、Query 管理機能セッションを開始するために使用されるファイルの名前です。このプロシージャーの中で使用できる唯一のステートメントのタイプは、 SET GLOBAL コマンドです。
  START コマンドで DSQSNAME キーワードが指定されていない場合、 Query コマンド・プロシージャーを見つけるために *SAA 規則が使用されます。それ以外の場合は、DSQSNAME キーワードによってセットされた命名規則が使用されます。 START コマンドで DSQSCMD キーワードを指定しない場合は、Query 管理機能は、デフォルト・プロシージャーである DSQSCMDP を探索して、それを実行します。デフォルト・プロシージャーが見つからない場合でも、START コマンドが失敗することはありません。 START コマンドのパラメーターとして指定する以外には、ここだけに DSQ データ変数をセットできます。
- DSQSRUN は、初期化の開始後に処理される、Query 管理機能プロシージャーを命名します。有効な手続きコマンドすべてが許可されます。
- DSQSNAME は、Query コマンドおよび SQL Query を処理する際に使用する命名規則です。詳細については、 3 ページの『Query 管理機能の Query オブジェクトの命名規則』を参照してください。 START コマンドの DSQSNAME にセットされたキーワード値は、 Query コマンド・プロシージャーで DSQSNAME にセットされたキーワード値を一時変更します。

  ***SAA** SQL 命名規則が使用されます。コマンドまたは Query プロシージャーで指定される修飾 Query オブジェクト名は、その形式が `'database.object'` となります。

  ***SYS** コマンドまたは Query プロシージャーで指定される修飾 Query オブジェクト名は、その形式が `'library/object'` となります。

- DSQSCNVT は、DB2 UDB for iSeries Query 管理機能オブジェクトが見つからない場合に、Query 管理機能が Query for iSeries 定義オブジェクトを検索するかどうかを示します。 Query 定義の中の情報は、コマンドで使用する一時 Query 管理機能オブジェクトを作成するために用いられます。

  たとえば、コマンド `RUN QUERY MYLIB/QRY1` では、Query 管理機能に、 QRY1 という名前の Query 管理機能プログラム・オブジェクトを探索するように指示しています。このオブジェクトが見つからないと、Query 管理機能は、Query 定義オブジェクトを探索し、その中の情報を使って Query を実行します。

  YES DB2 UDB for iSeries Query 管理機能オブジェクトが見つからない場合、Query 管理機能は Query for iSeries 定義オブジェクトを検索します。

  NO DB2 UDB for iSeries Query 管理機能オブジェクトが見つからなくても、Query 管理機能は Query for iSeries 定義オブジェクトを検索しません。

  ONLY Query 管理機能は Query for iSeries 定義オブジェクトの検索だけを実行します。

- DSQOAUTH は、Query 管理機能が作成するすべてのオブジェクトに与えられる権限です。 Query コマンド・プロシージャーまたは START コマンドの DSQOAUTH キーワードに値をセットすることによって、 Query インスタンスの処理時に作成されたすべてのオブジェクトに関するデフォルトの共通権限を指定することができます。 指定できる値は次のとおりです。

  ***LIBCRTAUT**
  オブジェクトに対する権限は、そのオブジェクトが作成されているライブラリーの CRTAUT パラメーターに指定された値と同じです。 CRTAUT パラメーターが変更されても、その新しい値は既存のオブジェクトの権限に影響しません。

  ***CHANGE**
  変更権限によって、他のユーザーはオブジェクトに対して、所有者に限定されているもの、またはオブジェクト存在権限およびオブジェクト管理権限によって制御されているものを除いて、す

べての操作を実行することができます。ユーザーは、オブジェクトを削除したり、それを新しい所有者に転送することを除いて、 Query オブジェクトを任意の方法で変更または使用することができます。

***ALL** 全権限によって、所有者に限定されているものまたは権限リスト管理権に制御されているものを除いて、他のユーザーはオブジェクトに対してすべての操作を実行することができます。オブジェクトを新しい所有者に転送することを除いて、 Query オブジェクトを任意に処理する (それを消去することも含む) ことができます。

***EXCLUDE**
EXCLUDE 権限は、他のユーザーが Query オブジェクトを処理できないようにします。特定タイプの権限が与えられていない限り、その所有者以外のユーザーが Query オブジェクトを使用することはできません。

***USE** 使用権限は、他のユーザーが Query オブジェクトを実行、エクスポート、または印刷できるようにし、インポートまたは保管することはできないようにします。

権限リスト名
権限リストの名前を指定した場合、 Query オブジェクトを使用するユーザーの能力を制御するために、その権限が使用されます。 詳細については、*機密保護解説書* (SD88-5027) を参照してください。

Query コマンド・プロシージャーを用いて権限を指定しなかった場合は、他のユーザーは Query オブジェクトに対する *EXCLUDE アクセスを認められることになります。

- DSQSDBNM は、 Query インスタンスにおいて Query 管理機能が開始するすべての SQL 操作の対象となるリモート・データベースです。 START コマンドまたは Query コマンド・プロシージャーでこのキーワードを指定しない場合には、 Query インスタンスと関連する接続が、START コマンドの実行時の CURRENT SERVER です。指定できる値は次のとおりです。

  ***CURRENT**
  Query インスタンスでは、CURRENT SERVER と関連づけられた接続を継承します。DSQSDBNM キーワードは CURRENT SERVER のリモート・データベース名にセットされます。

  注: リレーショナル・データベース・ディレクトリーには、システムがアクセスできるすべてのリモート・データベースおよびローカル・データベースの名前が入っています。

  ローカル・データベース名がリレーショナル・データベース・ディレクトリーの中にない場合には、このオプションは *NONE にセットされます。デフォルトは *CURRENT です。

  ***NONE**
  接続はローカル・データベース・マネージャーに対して行われます。ローカル・データベースの項目が、リレーショナル・データベース・ディレクトリーの中に存在している必要はありません。

  rdbname
  リモート・データベース名を表します。この名前により、分散リレーショナル・データベース・アーキテクチャー (DRDA) を用いてアクセスできるデータベースが識別されます。 rdbname は、18 文字までの長さにすることができます。この名前は、英字の大文字 (A〜Z)、数値 (0〜9)、または下線 (_) で構成しなければなりません。最初の文字は英字で、リレーショナル・データベース・ディレクトリーの中に該当のリモート・データベース名の項目がなければなりません。

START コマンドの DSQSDBNM 変数についてセットされたキーワード値は、 Query コマンド・プロシージャーにより DSQSDBNM 変数についてセットされたすべてのキーワード値を一時変更することになります。

- DSQSDBNM キーワードでリモート・データベースの名前を指定した場合、 DSQUSER および DSQPASSWORD は、そのリモート・データベースで使用されるユーザー ID およびパスワードを指定します。これらのキーワードのいずれかを指定したい場合でも、両方を指定しなければなりません。これらのキーワードを指定しない場合には、デフォルト・ユーザー ID は *CURRENT に、そしてデフォルト・パスワードは *NONE になります。 DSQUSER のキーワードも DSQPASSWORD のキーワードも、 Query コマンド・プロシージャーによってセットすることはできません。 DSQPASSWORD 変数および値は、グローバル変数プールには保管されません。
- DSQCMTLV は、セッションの実行時に使用されるコミットメント制御のレベルを指定するために用いられるキーワードです。デフォルトは NONE です。このキーワードをその他の値にセットした場合、Query 管理機能はコミットメント制御のもとですべての SQL ステートメントを実行します。 NONE 以外のコミットメント・レベルでセッションを実行する場合には、COMMIT および ROLLBACK SQL ステートメントを実行することができます。 ERASE TABLE CPI コマンドは、Query インスタンスの DSQCMTLV の値と関連するコミットメント制御レベルとなります。

  注: SAVE DATA AS コマンドのコミットメント制御レベルは、常に NONE となります。

  DSQCMTLV キーワードについて指定することができる値は次のとおりです。

  NONE コミットメント制御は使用されません。この値は *NONE と同じです。

  UR 更新された行だけが、トランザクションの終わりまでロックされます。この値は *CHG と同じです。

  CS カーソルが位置づけられている行は、カーソル位置が変更されるまでロックされます。更新された行は、トランザクションの終わりまでロックされます。この値は *CS と同じです。

  RS 選択されたすべての行が、トランザクションの終わりまでロックされます。更新された行は、トランザクションの終わりまでロックされます。この値は *ALL と同じです。

  RR 作業単位 (UOW) が完了するまで、選択されたすべての行がロックされます。排他ロックに加えて、分離レベル RR (逐次化可能) でアプリケーション・プロセスを実行すると、読み取るすべての行に対して少なくとも共用ロックを獲得します。ロックにより、アプリケーション・プロセスは、並行するアプリケーション・プロセスから完全に分離されます。そのため、1 つの作業単位内で Query を繰り返すと、必ず同じ結果になります。

  Query コマンド・プロシージャーによりセットされた DSQCMTLV 値は、 START コマンドで DSQCMTLV についてセットされたキーワード値によって一時変更されます。
- DSQRDBCNNMTH は、セッションにおいて使用する接続管理方式を指示します。デフォルトは *DUW です。 DSQRDBCNNMTH には、次の値を指定できます。

  ***DUW** 複数のリレーショナル・データベースに接続できます。これは、DSQRDBCNNMTH のデフォルトです。他のリレーショナル・データベースに連続して START または CONNECT コマンドを出しても、前の接続は切断されません。ある接続から別の接続に切り替えるためには、SET CONNECTION コマンドを使用できます。これによって、読み取り専用接続となることがあります。同じデータベースに連続して CONNECT コマンドを出すと、失敗となります。ただし、同じデータベースに連続して START コマンドを出すことはできます。

  ***RUW** リレーショナル・データベースに対する接続が 1 つだけ許されます。他のリレーショナル・データベースに対して連続して START または CONNECT コマンドを出すと、新しい接続が設定

される前に、前の接続が切断されます。同じデータベースに連続して START または CONNECT コマンドを出しても、現行接続は変わりません。

注: 以前の接続はすべて切断されます。詳細については、 24 ページの『Query 管理機能での CONNECT』を参照してください。

- DSQCONFIRM は、CONFIRM パラメーターを伴う DB2 for iSeries Query 管理機能コマンドに対するデフォルトを指定するためのキーワードです。たとえば、DSQCONFIRM を NO にセットしておくと、ERASE QUERY コマンドに CONFIRM キーワードを指定しない場合は、確認処理が行われません。DSQCONFIRM には、次の値を指定できます。

  YES ERASE、IMPORT または EXPORT コマンドに CONFIRM キーワードを指定しないと、確認処理が行われます。

  NO この場合、確認処理は行われません。

  値の長さ

  キーワード値を含むプログラム・ストレージの長さ。

  値

  キーワード値を含むプログラム・ストレージ域。

  値のタイプ

  キーワード値を含むストレージ域のデータ・タイプ。

Query 管理機能での START の例

START コマンドを使用したプログラムの例については、 『第 7 章 Query 管理機能での呼び出し可能インターフェース』を参照してください。

Query 管理機能での START Query コマンド・プロシージャー

Query 管理機能の初期設定の一部として実行される Query コマンド・プロシージャーの名前を指定するためには、START コマンドの DSQSCMD キーワードを使用してください。このプロシージャーを使用して、アプリケーション固有のユーザー変数をセットすることもできます。

DSQSCMD パラメーターに使用される Query コマンド・プロシージャーは、 START コマンド自身以外でユーザーが DSQ 変数をセットできる唯一の場所です。

注: START コマンドで指定される DSQ キーワードは、 DSQSCMD でセットされる同じ DSQ 値を常に一時変更します。たとえば、

```
START(DSQSCMD=MYPROC DSQSNAME=*SAA...

      MYPROC
      SET GLOBAL(DSQSNAME=*SYS...
```

この結果は、DSQSNAME=*SAA になります。

Query コマンド・プロシージャーを使用して設定できるのは、以下の DSQ 変数です。

- DSQSMODE
- DSQSRUN
- DSQOAUTH
- DSQSNAME
- DSQCONFIRM

- DSQAPRNM
- DSQSDBNM
- DSQCMTLV
- DSQRDBCNNMTH
- DSQSCNVT

Query コマンド・プロシージャーでセットされたその他の DSQ 変数は、すべて無視されます。ユーザー提供のプロシージャーでセットされていないすべての DSQ 変数には、デフォルトが適用されます。ユーザーが Query コマンド・プロシージャーを使用して設定できる DSQ 変数に使用可能なパラメーターは、次のとおりです。

DSQSMODE = INTERACTIVE | BATCH
このパラメーターは、後続のコマンドが出された時の Query 管理機能の操作モードを指示します。有効なオプションは次のとおりです。

INTERACTIVE
Query 管理機能の処理中に画面を表示することが可能になります。 RUN QUERY コマンドの結果として生成されたすべての報告書が画面に表示されます。応答を要求する確認メッセージが画面に表示され、そのメッセージに応答することができます。

BATCH
Query 管理機能の処理中に画面は表示されません。応答を要求するメッセージは、すべてがエラーとなります。その他のすべてのメッセージは、ジョブ・ログに送られます。

DSQSRUN = *Query* プロシージャー名
Query プロシージャー名は、初期設定が開始された後に実行される Query 管理機能プロシージャーの名前を指定します。

DSQSRUN パラメーターが START コマンドに指定されておらず、 DSQSRUN 変数が Query コマンド・プロシージャーでセットされていない場合には、初期設定プロシージャーは実行されません。

DSQOAUTH = *CHANGE、*EXCLUDE、*USE、*ALL、または権限リスト名
DSQOAUTH が Query コマンド・プロシージャーによってセットされない場合には、デフォルトとして *EXCLUDE が使用されます。

DSQSNAME = *SYS | *SAA
このパラメーターは、 Query 管理機能コマンドの処理時に使用される命名規則の名前を指定します。

***SYS** コマンドまたは Query 管理機能プロシージャーに修飾名を指定するために、 library/object の形式を使用します。

***SAA** SQL 命名規則が使用されます。コマンドまたは Query 管理機能プロシージャーに修飾名を指定するために、 database.object の形式を使用します。

DSQSNAME パラメーターが START コマンドに指定されておらず、 DSQSNAME 変数が Query コマンド・プロシージャーでセットされていない場合には、デフォルトの命名規則として *SAA が使用されます。

DSQCONFIRM = YES | NO
このキーワードは、確認処理が許されるコマンド (IMPORT、EXPORT、および SAVE DATA) に CONFIRM の指定がない場合のデフォルトの値を指定します。この DSQ 変数が Query コマンド・プロシージャーに指定されていない場合のデフォルトは、 DSQCONFIRM=YES です。

START

DSQSDBNM = *CURRENT | *NONE | リモート・データベース名

このキーワードは、Query インスタンスにおいて Query 管理機能が開始するすべての SQL 操作が対象とするリモート・データベースを指定します。 START コマンドまたは Query コマンド・プロシージャーでこのキーワードを指定しない場合には、 Query インスタンスと関連する接続が、START コマンドの実行時の CURRENT SERVER です。指定できる値は次のとおりです。

DSQCMTLV = NONE | UR | CS | RS | RR

このキーワードは、セッション中に使用するコミットメント制御のレベルを指定するために使用します。デフォルトは NONE です。このキーワードをその他の値にセットした場合、Query 管理機能はコミットメント制御のもとですべての SQL ステートメントを実行します。 NONE 以外のコミットメント・レベルでセッションを実行する場合には、COMMIT および ROLLBACK SQL ステートメントを実行することができます。 ERASE TABLE CPI コマンドは、Query インスタンスの DSQCMTLV の値と関連するコミットメント制御レベルとなります。

DSQSCNVT = YES | NO | ONLY

このキーワードは、Query 管理オブジェクト情報が使用できない場合に、 Query 情報および書式情報を Query for iSeries 定義 (QRYDFN) から引き出せるかどうかを示します。このキーワードに NO を指定すると、EXPORT、 PRINT、または RUN コマンドに指定された Query または書式が見つからなかった場合、コマンドはエラーで終了します。

EXPORT、PRINT、または RUN コマンドに指定された Query または書式が見つからなかった場合に、 Query 管理機能に Query for iSeries 情報を使用するように要求するためには、このキーワードに YES を指定してください。 このキーワードが START コマンドに指定されていない場合のデフォルトは、DSQSCNVT=NO です。

このキーワードに ONLY を指定すると、該当するタイプの Query 管理機能オブジェクトの有無にかかわらず、 QRYDFN オブジェクトが見つからなかった場合には、コマンドはエラーで終了します。

Query 管理機能でのコマンド・プロシージャーの例

以下は、プロダクトに組み込まれたデフォルトのプロシージャーの内容の例です。

```
'SET GLOBAL (DSQSMODE=BATCH'
'SET GLOBAL (DSQOAUTH=*EXCLUDE'
'SET GLOBAL (DSQSNAME=*SAA'
'SET GLOBAL (DSQCONFIRM=YES'
```

Query 管理機能での CL コマンド

以下の制御言語 (CL) コマンドは、Query 管理機能を処理する場合や、Query 管理機能の報告書を作成するアプリケーションを書くときに、一般に使用されるコマンドです。これらの CL コマンドの使用法については、*CL プログラミング* (SD88-5038) を参照してください。

Query 管理機能での ANZQRY (照会の分析)

照会の分析 (ANZQRY) コマンドを使用すると、Query 管理機能での変換の問題について Query for iSeries 定義 (QRYDFN) オブジェクトを分析することができます。 Query 管理機能の戻す診断メッセージには、分析された QRYDFN オブジェクトから得られた Query および書式情報の使用における、Query for iSeries と Query 管理機能との間の潜在的な相違点が示されます。完了メッセージには、見つかった問題のうち最高の重大度の問題が示されます。

Query 管理機能での CRTQMFORM (Query 管理機能書式の作成)

Query 管理機能書式の作成 (CRTQMFORM) コマンドによって、指定された情報源から Query 管理機能書式を作成することができます。書式は、報告書を印刷または表示する時に DATA をフォーマット設定する (Query の実行による) 方法を定義します。書式情報はソース・ファイル・メンバー・レコードにエンコードされます。

Query 管理機能での CRTQMQRY (Query 管理機能プログラムの作成)

Query 管理機能プログラムの作成 (CRTQMQRY) コマンドによって、指定された情報源から Query を作成することができます。 1 つの Query は、変数の置換値を入れることができる単一の SQL ステートメントです。 Query は、1 つのソース・ファイル・メンバーの中で複数レコードに分割することができます。

Query 管理機能での DLTQMFORM (Query 管理機能書式の削除)

Query 管理機能書式の削除 (DLTQMFORM) コマンドによって、既存の Query 管理機能書式をライブラリーから削除することができます。総称書式名を使用して、複数の書式をライブラリーまたはライブラリーのリストから削除することもできます。

Query 管理機能での DLTQMQRY (Query 管理機能プログラムの削除)

Query 管理機能プログラムの削除 (DLTQMQRY) コマンドによって、既存の Query 管理機能プログラム・オブジェクトをライブラリーから削除することができます。総称 Query 名を使用して、複数の Query をライブラリーまたはライブラリーのリストから削除することもできます。

Query 管理機能での RTVQMFORM (Query 管理機能書式の検索)

Query 管理機能書式の検索 (RTVQMFORM) コマンドによって、 Query 管理機能書式 (QMFORM) オブジェクトから、エンコードされた書式ソース・レコードを検索することができます。ソース・レコードは、編集が可能なソース・ファイル・メンバーに入れられます。

また、指定された QMFORM が存在していない場合は、書式ソース・レコードを QRYDFN オブジェクトから検索することもできます。

Query 管理機能での RTVQMQRY (Query 管理機能プログラムの検索)

Query 管理機能プログラムの検索 (RTVQMQRY) コマンドによって、 SQL ソース・ステートメントを Query 管理機能プログラム (QMQRY) オブジェクトから検索することができます。ソース・レコードは、編集が可能なソース・ファイル・メンバーに入れられます。

また、指定された QMQRY オブジェクトが存在していない場合は、 QMQRY ソース・レコードを QRYDFN オブジェクトから検索することもできます。

STRQMPRC (Query 管理機能プロシージャーの開始)

Query 管理機能プロシージャーの開始 (STRQMPRC) コマンドによって、ソース・ファイルにメンバーとして保管された Query 管理機能プロシージャーを実行することができます。

STRQMQRY (Query 管理機能プログラムの開始)

Query 管理機能プログラムの開始 (STRQMQRY) コマンドによって、既存の Query 管理機能プログラムを実行することができます。 Query は、Query 管理機能プログラムに保管された SQL ステートメントを実行します。 SQL SELECT ステートメントの実行によって収集された DATA は、表示、印刷、または別のデータベース・ファイルに保管することができます。

START

また、指定された QMQRY オブジェクトが存在していない場合は、 QRYDFN オブジェクトから SQL ステートメントを引き出すこともできます。

WRKQMFORM (Query 管理機能書式の処理)

Query 管理機能書式の処理 (WRKQMFORM) コマンドは、ユーザー指定の Query 管理機能書式名のサブセットの Query 管理機能書式のリストを表示します。このリストから、Query 管理機能書式と関連したいくつかの機能を使用することができます。

WRKQMQRY (Query 管理機能プログラムの処理)

Query 管理機能プログラムの処理 (WRKQMQRY) コマンドは、ユーザー指定の Query 管理機能プログラム名のサブセットからの Query 管理機能プログラムのリストを表示します。このリストから、Query 管理機能プログラムに関連したいくつかの機能を使用することができます。

第 5 章 Query 管理機能のプロシージャー

同じ Query 管理機能コマンドを使用して、報告書の作成を繰り返している場合があります。このような場合には、1 つのプロシージャー を作成し、これらのステップをまとめて実行することを考えてください。プロシージャーを使用すれば、 1 組の Query 管理機能コマンドをたった 1 つの RUN コマンドで実行することができます。

プロシージャーによって、アプリケーションに柔軟性を持たせることもできます。名前付きプロシージャーを実行するためのアプリケーションを作成することができます。アプリケーション・プログラムの変更を行わずに、いつでも新しい状況に適合するようにプロシージャーを更新または調整することができます。

Query 管理機能でのプロシージャーの作成

プロシージャーを使用して、一連の Query 管理機能 CPI コマンドを 1 つの RUN コマンドで実行することができます。共通のプログラム部分について 1 つのプロシージャーを作成し、一連のコマンドではなく 1 つのコマンドでそのプロシージャーを呼び出すことができます。

プロシージャーを作成する場合には、以下の規則に注意してください。

- プロシージャーは、ソース・ファイル・メンバーです。
- プロシージャーには、Query 管理機能コマンドとブランク行を入れることができます。 (ブランク行はコマンドの処理には影響しません。) プロシージャーには任意に、H レコードおよびコメント用の V レコードを入れることもできます。コメント・レコードは、プロシージャーのインポート時にテキスト記述子として使用することができます。
- 各 Query 管理機能コマンドには、大文字の英字だけを使用しなければなりません。
- すべてのコマンドは、アポストロフィまたは引用符で囲む必要があります。コマンドに引用符が含まれる場合、内部の引用符は例 2 に示すように、 2 つの連続するアポストロフィ (')、または引用符 (“”) で表されます。詳細については、『付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用』を参照してください。
- プロシージャーには、別のプロシージャーまたは Query を実行する RUN コマンドを含めることができます。
- 1 つのコマンドは、1 行の 79 文字までに限定されています。
- プロシージャー行の幅は、ソース・ファイル・レコードの幅に制限されます。
- プロシージャー行上の Query コマンドの幅は、プロシージャー解析を行った後で 256 文字に制限されます。プロシージャー解析では、先行ブランクおよび後書きブランクが除去され、アポストロフィおよび引用符が削減されます。

Query 管理機能でのプロシージャーの作成例 1

```
/*H QM4 01 P 01 E V W E R 01 03 90/3/19 14:27 */
/*V 1001 014 Monthly report */
/* This produces the monthly reports. */
'RUN QUERY A'                 /* PAYROLL */
'PRINT REPORT (PRINTER=PRT01'
'RUN QUERY B'                 /* ACCTS RECEIVABLE */
'PRINT REPORT (PRINTER=PRT01'
```

H レコードおよび V レコードの形式については、 『第 8 章 Query 管理機能でのオブジェクトのエクスポートとインポート』で説明します。 V レコードのテキスト (Monthly report) は、 IMPORT PROC コマンドでソース・ファイル・メンバーのテキスト記述をセットするために使用されます。

Query 管理機能でのプロシージャーの作成例 2

```
'SAVE DATA AS LASTWKDATA (COMMENT='Last weeks'data'
'IMPORT FORM REPT4 FROM MYLIB/FORMS(REPT4) (CONFIRM=YES'
'SET GLOBAL (TBLENAME=MYFILE'
'SET GLOBAL (CMPVAL2='''Joe A. Customer'''
'RUN QUERY REPT4QRY (FORM=REPT4'
'SAVE DATA AS LASTWKDATA'
'PRINT REPORT'
```

Query 管理機能でプロシージャーを作成するためのステップ

MYPROC と呼ばれるプロシージャーを作成するには、次のように行います。

1. ソース・メンバーの MYPROC を編集して、次のような情報を追加します。

   ```
   /*H QM4 01 P 01 E V W E R 01 03 90/3/19 14:27 */
   /*V 1001 014 Monthly report */
   /* This produces the monthly reports. */
   'RUN QUERY A'                 /* PAYROLL */
   'PRINT REPORT (PRINTER=PRT01'
   'RUN QUERY B'                 /* ACCTS RECEIVABLE */
   'PRINT REPORT (PRINTER=PRT01'
   ```

2. メンバー MYPROC を保管します。

これで、プロシージャーを実行する準備ができました。

プロシージャーを使用する場合、次の規則が適用されます。

- プロシージャーの中に 12 までのプロシージャーをネストすることができますが、各プロシージャーは呼び出し側の特性を持つようになります。したがって、 RUN QUERY コマンドを実行したばかりのプロシージャーで処理される PRINT コマンドは、その RUN QUERY コマンドによるデータを印刷することになります。
- 再帰ネストは許されません。すなわち、プロシージャーが自分自身のプロシージャーを実行する RUN PROC を含むことはできません。
- プロシージャー中の Query コマンドは、引用符 (″) またはアポストロフィ (') で区切らなければなりません。
- Query 管理機能のプロシージャーは高水準言語の構成を持たないので、 GET コマンドは機能しません。プロシージャー中の GET コマンドは、メッセージをジョブ・ログに送ります。このメッセージには、変数名と値が入っています。
- Query 管理機能は、SET コマンドの変数値すべてを文字ストリングとして扱います。したがって、整数の変数をプロシージャーに設定することはできません。
- プロシージャーには必要に応じて、 H レコードとその直後にコメントの V レコードを含めることができます。
- プロシージャーには必要に応じて、コメントを入れることができます。コメントは、そのプロシージャーで行われる処理を記述するのに使用します。コメントを、複数の行にまたがらせたり、また他のコメントの中にネストすることはできません。コメントは、その始めに **/*** を、終わりに ***/** を付けて区切ります。

| iSeries システムにおいて、各コマンドの処理は、そのプロシージャーが処理される特定のインスタンスの
| モードによって異なります。

Query 管理機能のプロシージャーは、ソース物理ファイルのメンバーです。 Query 管理機能では、RUN PROC、IMPORT PROC、EXPORT PROC、PRINT PROC、および ERASE PROC のコマンドで特定のメンバーを指定することができます。この指定は、メンバーを Query オブジェクト名の一部として指定できるようにすることによって行われます。メンバー名は間にブランクがないように括弧で区切って、 Query オブジェクト名の後に続けなければなりません。

オブジェクト名の一部としてメンバーを指定しない場合には、ファイルの最初のメンバーが指定されたものと想定されます。 ERASE PROC が出され、しかも複数のメンバーが存在する場合には、最初のメンバーだけが削除されます。メンバーの指定がなく、しかも IMPORT PROC 処理の一環としてメンバーが作成される場合、そのメンバーにはプロシージャー・ファイルの名前が与えられます。

Query 管理機能のプロシージャーでのユーザー対話

Query ユーザーとの対話は、Query 管理機能の対話状態によって異なります。この状態は、START コマンドの起動パラメーター DSQSMODE によって制御されます (47 ページの『Query 管理機能での START』を参照)。対話の状態を認める場合、プロシージャーの使用に際してさらに考慮すべき事項があります。たとえば、ERASE コマンドで CONFIRM=YES を指定するか、あるいは IMPORT コマンドで DISPLAY=YES を指定すると、プロシージャーが失敗することがあります。

複数の Query が入っているプロシージャーが実行される場合、それぞれの Query 処理に応じてフォーマット設定された報告書画面が表示されます。それから報告書をページ送りすることができます。報告書を終了すると、プロシージャーに制御権が戻され、次のステートメントが処理されます。

Query 管理機能でのプロシージャー対話

プロシージャーを処理する方式によって、各コマンドの処理中に生じることについては、 『第 4 章 Query 管理機能でのコマンド』を参照してください。

- プロシージャーは別のプロシージャーの中から呼び出すことができます。これはネストと呼ばれています。他のプロシージャーの中にあるプロシージャーは、最初のプロシージャーで指定されたパラメーターを使用します。したがって、RUN QUERY コマンドを実行したばかりのプロシージャーで処理される PRINT コマンドは、その RUN QUERY コマンドのデータを印刷することになります。
- Query 管理機能で使用が許されているネスト・レベルは 12 です。
- 再帰ネストは許されません。たとえば、PROCEDURE A にコマンド RUN PROC A を入れることはできません。
- GET コマンドはプロシージャー内では機能しません。プロシージャーの中に GET コマンドがあると、変数の名前および値を入れるジョブ・ログに通知メッセージが送られることになります。
- Query 管理機能は、SET コマンドのすべての変数値を文字ストリングとして扱います。

Query 管理機能でのプロシージャー・オブジェクト

Query プロシージャーはソース物理ファイルです。 Query 管理機能では、メンバーを Query オブジェクト名の一部として指定できるため、 RUN PROC、 IMPORT PROC、 EXPORT PROC、 PRINT PROC、および ERASE PROC コマンドで特定のメンバーを指定できます。メンバー名は間にブランクがないように括弧で区切って、 Query オブジェクト名の後に続けなければなりません。次の例では、各コマンドをどのように変更すれば、Query 管理機能で特定のメンバーを指せるかを示しています。

```
RUN PROC MYLIB/MYPROCS(MYMEMBER)
PRINT PROC MYLIB/MYPROCS(MYMEMBER)
IMPORT PROC MYPROCS(MYMEMBER) FROM QQMQRYSRC
EXPORT PROC MYLIB/MYPROCS(MYMEMBER) TO QQMQRYSRC(MYMEMBER)
```

メンバーをオブジェクト名の一部として指定しない場合は、常にファイルの最初のメンバーと見なされます。 ERASE PROC が出され、複数のメンバーが存在している場合、最初のメンバーだけが削除されます。メンバーを指定しないで、しかも IMPORT PROC 処理の一部として作成したい場合、メンバーは PROC ファイル名と同じ名前で作成されます。

RUN PROC または PRINT PROC コマンドの実行では、Query 管理機能は、ソース・ファイルの全体を処理します。 PROC ファイルがインポートまたはエクスポートで作成される場合は、79 文字のデータ長で作成されます。したがって、それより大きいレコード長のファイルからのインポートまたはエクスポートでは、切り捨てが行われます。

Query プロシージャーは、次の方法で実行することができます。

1. STRQMPRC CL コマンドを出す。
2. Query 管理機能の呼び出し可能プログラミング・インターフェースを介して、 RUN PROC Query コマンドを実行する。
3. DB2 UDB for iSeries Query Manager の Query ステートメントのポップアップで RUN PROC を実行する。
4. Query 管理機能呼び出し可能プログラミング・インターフェースの START コマンドの DSQSCMD キーワードにプロシージャー名を指定する。

Query 管理機能でのプロシージャー・エラーの処理

重大度が FAILURE のエラーが起こると、プロシージャーの処理は停止し、プロシージャーの完了コードにそのエラーが示されます。プロシージャーの処理中に出されたメッセージは、すべてジョブ・ログに入れられ、要約メッセージが連絡域に戻されます。

Query 管理機能でのプロシージャー・エラーのカテゴリー

Query 管理機能プロシージャーでは、次のエラーのカテゴリーが可能です。

- Query 管理機能プロシージャー中のコマンドが使用できない。
- Query 管理機能プロシージャー中のコマンドが正しくない。
- Query 管理機能プロシージャー中のストリングが正しくない。
- Query 管理機能プロシージャーでは再帰は使用できない。
- プロシージャーの最大ネスト・レベルを超えた。 15 のネスト・レベルが使用可能です。

第 6 章 Query 管理機能での報告書の書式

この章では、書式の作成方法と、Query 管理機能の報告書作成機能について説明します。 FORM に指定されたフォーマット設定情報を用いて Query の結果をフォーマット設定し、報告書を作成します。

Query 管理機能でアプリケーションが FORM を使用する方法

アプリケーションは、エクスポート FORM を直接変更または作成することによって、 FORM を作成または変更することができます。

アプリケーションを使用して、Query 管理機能から既存の FORM をエクスポートして変更し、その FORM をインポートし、報告書をフォーマット設定することができます。ただし、FORM はそのつどエクスポートする必要はありません。アプリケーションは、エクスポートされた既存の FORM にアクセスし、変更して、報告書作成のために、その変更した書式を Query 管理機能にインポートすることができます。

FORM の一部 (たとえば、列情報用の T レコードおよび R レコードを続けた見出し (H) レコード) だけをインポートすることもできます。 FORM フィールドの残りの部分は、デフォルトで充てんすることができます。

書式の作成方法については、次のセクション、Query 管理機能での書式の作成を参照してください。

Query 管理機能での書式の作成

ユーザーは、書式 (form) で指定した情報を使用して、 Query の結果をフォーマット設定することによって、報告書を作成することができます。アプリケーションでは、エクスポートされた書式を直接、変更または作成することによって、書式を作成または変更することができます。

アプリケーションを使用して、既存の書式を Query 管理機能からエクスポートし、変更し、変更したその書式をインポートして報告書をフォーマット設定することができます。そのつど、書式をエクスポートする必要はありません。アプリケーションは、エクスポートされた既存の書式にアクセスし、変更して、報告書作成のために、その変更した書式を Query 管理機能にインポートすることができます。

また、特定の書式フィールドをデフォルトで充てんできるソース仕様から書式をインポートすることもできます。フォーマット設定する列ごとに、その後に 1 つの T レコードと 1 つの R レコードが続く見出し (H) レコードだけを使用することができます。書式フィールドの残りの部分は、Query 管理のデフォルトで充てんされます。これによって、ユーザーは、すべての書式属性に値を入力しなくても書式全体を作成することができます。 QUERY 管理機能書式は、DB2 UDB for iSeries Query マネージャーを使用して作成することも可能です。

Query 管理機能でのデフォルト書式の作成

1. レコード長を 162 桁にして、ライブラリー MYLIB1 のソース・ファイル TESTFORM にソース・メンバー DEFAULT を作成することによって、外部化書式オブジェクトを生成するためのテンプレートを作成します。これを行うには、

   OS/400 コマンド行に、次のように入力して、

   ```
   CRTSRCPF MYLIB1/TESTFORM RCDLEN(162) MBR(DEFAULT)
   ```

   実行キーを押します。
2. メンバー DEFAULT を編集して、次の情報を追加します。

```
H QM4 05 F 03 E V W E R 01 03 90/12/31 09:21
T 1110 001 000
R
E
```

このデフォルトの書式がエクスポートされた時点で、日時はその当日の日時になります。

3. メンバー DEFAULT を保管します。
4. MYLIB1 にデフォルトの書式オブジェクトを作成するには、 Query 管理機能書式の作成 (CRTQMFORM) コマンドを使用して、作成したソース・ファイル・メンバーをインポートします。 OS/400® コマンド行に、次のように入力して、

   ```
   CRTQMFORM QMFORM(MYLIB1/DEFAULT)
   SRCFILE(MYLIB1/TESTFORM) SRCMBR(DEFAULT)
   ```

   実行キーを押します。
5. Query 管理機能書式の検索 (RTVQMFORM) CL コマンドを使用して、 CRTQMFORM コマンドの結果作成された Query 管理機能書式をエクスポートします。これを最も簡単に行うには、次のようにします。
   a. F9 を押して、直前のコマンドを検索します。
   b. CRT を RTV に置き換えます。

      コマンド全体は次のようになります。

      ```
      RTVQMFORM QMFORM(MYLIB1/DEFAULT)
      SRCFILE(MYLIB1/TESTFORM) SRCMBR(DEFAULT)
      ```
   c. 実行キーを押します。

これで、メンバー DEFAULT には、実行時にセットされる属性を除くすべての書式属性のデフォルトを含む完全な書式が入ります。その後、メンバー DEFAULT を編集してフィールド属性を変更し、さらに列を追加することができます。

デフォルトは、指定されない情報に対して提供されるものです。書式がインポートされる時に提供されるデフォルトもあります。その他のデフォルト (データ・タイプや列見出しなど) は、実行時に提供され、処理された Query の結果のデータによって異なります。

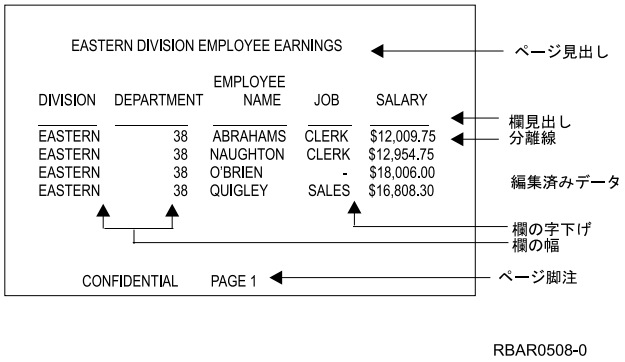
書式にエンコードするキーワードは、大文字の英字でなければなりません。テキスト・フィールド (見出し、フッター、および最終テキスト) は、大文字でも小文字でもかまいません。

一般に、書式オブジェクトのフィールドは、次の機能上のカテゴリーによってグループ化されます。

- 切れ目
- 列
- 最終
- オプション
- ページ

Query 管理機能でのフォーマット設定の用語

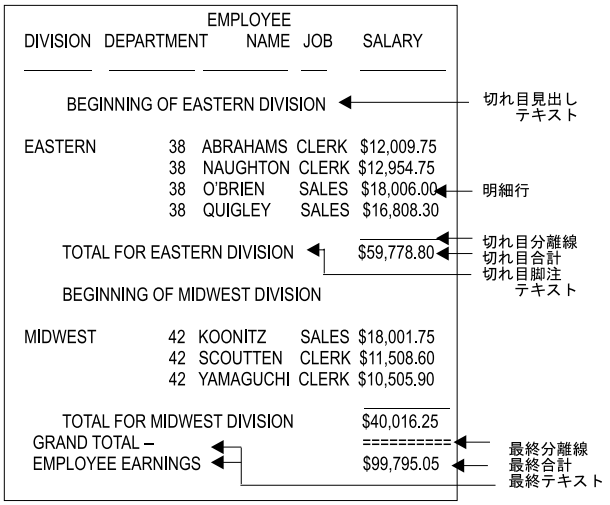
FORM で使用可能なすべてのオプションを理解するために、 63 ページの図 8 および図 9 では、いくつかの FORM オプションが形式化された報告書に与える効果を示してあります。



RBAR0508-0

図 8. 報告書の基本部分

“編集済みデータ” は、データベースからの情報を関連編集コードに従って表示したものです。



RBAR0509-0

図 9. 制御レベルが 1 つある報告書の基本部分

FORM オブジェクトには、報告書を記述するフィールドが入れられます。 Query 管理機能は、最大 255 列の情報をサポートします。 Query 管理機能には、データベースから使用できるデータは 1 行あたり 32 KB までという制限があります。
“使用目的” 以外のすべてのフィールドにはデフォルトが用意されていますが、そのデフォルトは実行された Query の結果のデータによって異なります。

FORM で使用されるキーワードは大文字の英字でなければなりません。テキスト・フィールド (見出し、フッター、および最終テキスト) には、大文字も小文字も使用することができます。

Query 管理機能での DBCS データ

FORM での 2 バイト文字セット (DBCS) データの使用法の詳細については、 『付録 A. Query 管理機能での DBCS データ』を参照してください。

Query 管理機能での COLUMN フィールド

次のトピックについては、Query 管理機能の COLUMN フィールドで取り上げられています。

Query 管理機能での列のデータ・タイプ

このフィールドは、報告書の列のデータ・タイプを表しています。使用可能な値は、NUMERIC、 CHAR、GRAPHIC、および DATE/TIME です。 OS/400 では、GRAPHIC と DATE/TIME はサポートされていません。 DB2 UDB for iSeries Query 管理機能の報告書の各列は、1 組のフィールド値で記述されます。FORM で *n* 番目の列の値が、Query によって選択された *n* に適用されます。以下では、列フィールドの定義に使用できるフィールドについて説明します。列フィールドでの 2 バイト文字セット (DBCS) データの使用についての詳細は、『付録 A. Query 管理機能での DBCS データ』 を参照してください。

表 3 には、*列* フィールドの属性について、デフォルトおよび使用できる値を示してあります。

表 3. 列フィールドのデフォルト

属性	デフォルト	使用できる値
列見出し	表での列見出し	1 ～ 62 文字 (最大 8 文字の下線を含む)
使用目的	—	AVG、 MIN、 MAX、 SUM、 COUNT、 BREAK1 ～ BREAK6、 AVERAGE、 MINIMUM、 MAXIMUM、 OMIT
字下げ	2	0 ～ 999
幅	使用されるデータ・タイプにより異なる。	1 ～ 32,767 (SBCS)
データ・タイプ	表でのフィールドのデータ・タイプ	CHAR、NUMERIC
編集コード- 数値	表でのフィールドの編集コード	E、D、I、J、K、L、P
編集コード- 文字	C	C、CW、CT
順序	列番号	1 ～ 999
編集コード- 日付、時刻	実行時	TDY、 TDM、 TDD、 TDYA、 TOMA、 TDDA、 TTS、 TTC、 TTA、 TTAN、 TTU、 TSI

Query 管理機能での列の見出し

このフィールドは、報告書の中の列の見出しを表します。

この見出しは、62 文字までの長さとすることができます。

見出しには下線文字を組み込み、それを使って、複数行見出しのための新しい行を示すことができます。Query 管理機能は、1 つの見出しで最高 8 つまでの下線を処理します。先行下線および後書き下線によって、列見出しの前および後にブランク・セグメントが作成されます。

たとえば、“AMOUNT_LAST_INCREASE” の列見出しの場合は、次のような 3 行の列見出しになります。

```
 AMOUNT
  LAST
INCREASE
```

下線が連続している場合は (その位置にかかわらず) ブランク行が生成されます。

下線の規則によって、列見出しに下線文字が表示されるのを防止できることに注意してください。この規則の唯一の例外は、8 文字を超える下線が現れた場合です。この場合には、余分な下線は見出しの最後の行の一部として印刷されます。

複数行の見出しを指定した場合には、Query 管理機能は、最も長い行のスペースに合わせて、それより短い行は自動的に中央にそろえます。文字データ用の見出しは自動的に左寄せされ、数値データ用の見出しは自動的に右寄せされます。データの位置調整は、列の幅の範囲内で行われます。幅の指定には、このフィールドの最長セグメントの長さを反映させる必要があります。

このフィールドの文字数が、幅で指定された文字数より多い場合、列に指定された幅に合わせてフィールドが切り捨てられます。

列見出しを指定しないと、Query 管理機能が実行時のデフォルトを提供します。該当する列のデータベース定義によって列見出しを設けてはならないことが示されている場合を除き、ブランク見出しの書式をインポートして列を見出しなしで表示することはできません。

Query 管理機能での列の使用目的

このフィールドは、形式化された報告書の明細行の列の用途を決めるためのものです。

各列には、それぞれ 1 つだけ*使用目的* を指定することができます。ある列に複数の使用目的を指定したい場合は、Query の中で複数回その列を選択し、 FORM の列ごとに使用目的コードを定義しなければなりません。

使用目的のオプションは、以下のとおりです。

[ブランク]
報告書に組み込まれる列。

OMIT
報告書から除外する列。

AVERAGE、COUNT、FIRST、LAST、MAXIMUM、MINIMUM、SUM
これらのキーワードは、列の中のデータを合計する統計的な使用目的の名前を指定します。これを使用した結果は、切れ目合計または最終合計に示されます。

COLUMN フィールド

使用目的コード	定義
AVERAGE (または AVG)	列での平均値
COUNT	列での非ヌル値の数
FIRST	列の中の最初の値
LAST	列の中の最後の値
MAXIMUM (または MAX)	列の中の最大値
MINIMUM (または MIN)	列の中の最小値
SUM	列での非ヌル値の合計

AVERAGE および SUM は数値データにしか使用できませんが、 COUNT、 FIRST、 LAST、 MAXIMUM、および MINIMUM は文字データにも数値データにも使用することができます。

MAXIMUM および MINIMUM を使用して文字を比較する場合には、短い方のストリングにはブランクが埋め込まれ、内部 2 進コードに基づいてストリングが比較されます。たとえば、文字ストリング “ab” は文字ストリング “aaa” より大きいことになります。 1 文字ごとの特別な処理はありません。あるシステムから別のシステムに移送可能なアプリケーションで MAXIMUM および MINIMUM を使用する場合には、マシンが異なれば照合順序も異なる場合があることに注意してください。 EBCDIC および ASCII は文字の照合が同じではないため、異なるシステムで MAXIMUM または MINIMUM を使用すると別の結果が生じる可能性があります。

総計的な使用目的の AVERAGE、COUNT、FIRST、LAST、MAXIMUM、 MINIMUM、および SUM には、次の規則が適用されます。

1. 統計オーバーフローが起こると、そのフィールドの値の代わりに、列の幅いっぱいに “>>>>” が表示されます。
2. 列幅が小さすぎるために集計を表示できない場合は、そのフィールドの値はその列の幅に対して “*****” で示されます。

BREAK1

BREAK1 は、列を第 1 レベルすなわち最高位の*制御の切れ目* として指定するために使用される値です。制御の切れ目は、列の値が変わるブレーク・ポイントです。

たとえば、従業員が部課番号および職種別に分類されて並べられている場合、 BREAK1 を使用してある部課の全従業員の給与を合計し、 BREAK2 を使用して部課内の職種別に給与を合計することができます。職種が異なる行が読み取られるたびに、BREAK2 は該当するデータと合計を生成して表示します。部課番号が異なる行が読み取られるたびに、 BREAK2 および BREAK1 は該当する両方のデータのセットを生成して表示します。

各切れ目合計が表示される前に、統計的な使用目的で表示されたすべての列の下に、一連のハイフン (“-”) からなる 1 行が入れられます。 この行は、FORM の “オプション” 部分のオプションによって消去することができます。通常、切れ目データの各セットの後に 1 つのブランク行が入れられます。

統計的な使用目的 (AVERAGE、 COUNT、 FIRST、 LAST、 MAXIMUM、 MINIMUM、 SUM) は、制御の切れ目で使用することができます。たとえば、合計データは、使用目的が SUM の場合にはすべての列の小計として表示され、使用目的が AVERAGE の場合には各列の平均として表示されます。

切れ目の一部として印刷されるデータは、このセクションで後述する切れ目定義によって決まります。

切れ目のレベル番号は、連続している必要はありません。この点から、制御の切れ目番号は絶対的なものではありません。制御の切れ目 1、3、および 5 は指定せずに、2、4、および 6 を指定することも

できます。しかし、制御の切れ目テキストの割り当ては絶対的なものとして行われます。すなわち、制御の切れ目 2 のテキストは 2 番目の制御の切れ目ではなく、常に制御の切れ目番号 2 に関連付けられます。

複数の列を同じ BREAKn 値に割り当てることができます。この場合、Query 管理機能が制御の切れ目を決定する必要があれば、Query 管理機能は、同じ制御の切れ目レベルを持つすべての列を、単一の連続した列と見なします。このことは、LAST_NAME と FIRST_NAME が別個の列として保管されている場合に特に有用です。 同じように、MONTH、DAY、および YEAR がそれぞれ別個の列である場合にも、このことが必要になります。

制御の切れ目の使用時には、列のデータが正しい順序になっていることが必要です。データを順序正しくするためには、報告書を作成する SELECT で ORDER BY を使用しなければなりません。

切れ目を指定しても、それによって列の再順序づけが自動的に行われるわけではありませんが、切れ目テキストは、通常すべての合計列の左側に表示されます。したがって、IBM は、「順序」値を使用して報告書上で切れ目列は左側に、合計列は右側に表示することをお勧めします。

BREAK2

BREAK2 は、列を 2 次レベルの制御の切れ目として指定するために使用される使用目的値です。BREAK2 が定義されている列 (複数の場合もある) に変更があったか、または BREAK1 が生成された時に、BREAK2 は自動的に生成されます。 BREAK1 は BREAK2 を生成させますが、BREAK2 が BREAK1 を生成させることはありません。

BREAK3 ～ BREAK6

BREAK3 ～ BREAK6 は、レベル 3 ～ 6 の切れ目に対する制御列の名前を指定する値です。

Query 管理機能での列の字下げ

このフィールドは、1 行の中の列の相対位置を表します。列と次のいずれかのものとの間のブランク文字数を単位として使用します。

- 前の列の右端
- 画面または用紙の左端

字下げは *n* (0 <= *n* <= 999) とすることができます。デフォルトの形式では、Query 管理機能は字下げを 2 に初期設定します。

Query 管理機能での列の幅

列の出力幅です。 列見出しおよびデータを表示するために確保される文字スペース数を指定します。この幅より大きい名前は切り捨てられます。 Query 管理機能では、幅フィールドは、数値のみとして定義します。最大幅は 1 バイト文字で 32767 です。表示したい値の長さが列の幅を超えている場合には、その値が数値データであれば一連のアスタリスク (****) で置き換えられ、文字データであれば右側を切り捨てられます。表示幅を変更し報告書を表示し直すことによって、所要の結果を得ることができます。日付、時刻、タイム・スタンプの各データは、形式化された報告書では文字データのように取り扱われます。次の規則も適用されます。

- 列見出しおよび列明細は、列の中で左寄せされます。
- 書式で指定された列の幅が、フォーマット設定された日付、時刻、またはタイム・スタンプの幅よりも小さい場合には、日付、時刻、またはタイム・スタンプの右側が切り捨てられます。
- 書式で指定された列の幅が、フォーマット設定された日付、時刻、またはタイム・スタンプの幅よりも大きい場合には、日付、時刻、またはタイム・スタンプの右側にブランクが埋め込まれます。
- ヌルの日付、時刻、またはタイム・スタンプのフィールドは、左寄せされたダッシュ (-) としてフォーマット設定されます。

COLUMN フィールド

幅値を指定しないと、Query 管理機能が実行時のデフォルトを提供します。

Query 管理機能での列のデータ・タイプ

このフィールドは、照会された表内の対応する列に含まれているデータのタイプを表します。データ・タイプ・オプションには次のものがあります。

CHARACTER
表内の列の中のデータは文字です。

NUMERIC
DATA 内の列の中のデータは数値です。数値データの形式には、2 進、パック、ゾーン、および浮動小数点があります。

DATE、TIME、TIMEST
この列の中のデータは日付、時刻、またはタイム・スタンプです。

GRAPHIC
この列の中のデータは DBCS グラフィックです (『付録 A. Query 管理機能での DBCS データ』を参照)。

Query 管理機能での列の編集コード

編集コードを使用して、文字データおよび数値データを、表示するための形式に設定します。 表 4 には、**日付**データ用の編集コードが示されています。

注: x は日付の区切り文字を指定する場所を示します。これは任意の特殊文字 (ブランクを含む) とすることができますが、文字や数字を指定することはできません。たとえば、ダッシュ (-) を使用することができます。

表 4. Query 管理機能 CPI の日付編集コード

編集コード	形式	例
TDYx	YYYYxMMxDD	TDY/ ==> 1987/01/31
TDMx	MMxDDxYYYY	TDM- ==> 01-31-1987
TDDx	DDxMMxYYYY	TDD ==> 31 01 1987
TDYAx	YYxMMxDD	TDYA/ ==> 87/01/31
TDMAx	MMxDDxYY	TDMA- ==> 01-31-87
TDDAx	DDxMMxYY	TDDA. ==> 31.01.87

表 5 には、**時刻**データ用の編集コードが示されています。

表 5. Query 管理機能 CPI の時刻編集コード

編集コード	形式	注	例
TTSx	HHxMMxSS	秒を含む	TTS. ==> 13.42.35
TTCx	HHxMMxSS	秒を含み、12 時間制	TTC: ==> 01:42:35
TTAx	HHxMM	省略形 (秒を含まない)	TTA, ==> 13,42
TTAN	HHMM	省略形 (区切り文字なし)	TTAN ==> 1342
TTUx	HHxMM AM または PM	米国様式	TTU: ==> 01:42 PM

69 ページの表 6 には、TIMESTAMP データ用の唯一の編集コードが示されています。マイクロ秒を除いた、タイム・スタンプ形式のすべてを表示するためには、幅フィールドの値を少なくとも 19 としなければなりません。幅フィールドの値が 26 より小さい場合には、それより後の桁は切り捨てられます。

表 6. Query 管理機能 CPI のタイム・スタンプ編集コード

編集コード	形式	例
TSI	yyyy-mm-dd-hh.mm.ss.nnnnnn	1987-01-21-13.42.19.123456

文字データ用には、以下の編集コードがあります。 DBCS グラフィック編集コードについては、『付録 A. Query 管理機能での DBCS データ』を参照してください。

C このコードは値の表示を変更しません。値が列内の 1 行に収まらない場合は、Query 管理機能は、列の幅に合わせてそのテキストを切り捨てます。 C は、文字データに対するデフォルトです。

CW

値の表示の変更は行いませんが、その値が列内の 1 行に収まらない場合、Query 管理機能は、列の幅に応じてテキストを折り返します。すなわち、列の終わりでデータを切り捨てるのではなく、Query 管理機能は列内の 1 行にできるだけ多くのデータを入れてから、次の行にデータを続けます。

CW 編集コードは、DBCS と 1 バイト文字データの混在している列に使用することができます。

CT

このコードは値の表示を変更しません。ただし、値が列内の 1 行に収まらない場合は、Query 管理機能はその列内のテキストに応じて列を折り返します。すなわち、列の終わりでデータを切り捨てるのではなく、Query 管理機能はできるだけ多くのデータを 1 行に入れ、ブランクが見つかったらそこで行を中断し、次の行にデータを続けます。データ・ストリングが長過ぎて列に収まらず、しかもブランクが入っていない場合は、Query 管理機能は、ブランクが見つかってそこでテキストを折り返せるまでは、幅によってデータを折り返します。

CT 編集コードは、DBCS と 1 バイト文字が混在する列に使用することができます。 Query 管理機能は、1 バイト文字または 2 バイト文字のブランクを 1 つ見つけると、その行を中断します。

数値データ用には、以下の編集コードがあります。

E 科学技術計算表記法で数字を表示します。たとえば、数値 -1234.56789 は -1.234E+03 として表示されます。報告書には最高 15 桁までの、表示可能な数値を入れることができます。常に 1 つのスペースが先行符号用に確保されていますが、正の数値の場合には表示されません。 E の後には常に 1 つの符号と少なくとも 2 桁の数字が続きます。3 桁まで表示されます。

D、I、J、K、L、および P

このコードは、数値を先行ゼロ、負記号、千単位の区切り記号、通貨記号、およびパーセント記号をさまざまに組み合わせて、10 進表記法で表示します。下記の表に例示してあります。

各コードには、小数点以下の桁数を示す 0 ～ 31 の数値を続けることができます。この数を指定しない場合は、小数点以下の桁数はゼロと見なされます。使用できるスペースに小数点以下の桁が収まりきらない場合は、数値は丸められます。スペースより小数点以下の桁数が少ない場合は、ゼロが埋め込まれます。

70 ページの図 10 には、数値編集コードによる数値 -1234567.885 のフォーマット設定方法が示されています。この例では、次のように想定しています。

- 幅は 15
- 小数点を表す文字はピリオド (.)
- 1000 の位の値の切れ目はコンマ (,)
- 一般的な丸めを使用 (四捨五入)
- D2 の通貨記号は、左側のドル記号 ($)
- 負の値の標識は、マイナス記号 (-)。後書きの負の標識はありません。

COLUMN フィールド

上記のパラメーターは、オペレーティング・システムによって異なり、システムまたはユーザーのプロファイルで指定することができます。アプリケーションを別の環境に移動する場合には、パラメーターがその環境間で同意義になるようにしてください。

編集コード	先行ゼロ	負符号	1000 の位の区切り文字	通貨記号	% 記号	-1234567.885 の表示
E	なし	あり	なし	なし	なし	-1.23456789E+06
D2	なし	あり	あり	あり	なし	-$1,234,567.89
I3	あり	あり	なし	なし	なし	-0001234567.885
J2	あり	なし	なし	なし	なし	000001234567.89
K3	なし	あり	あり	なし	なし	-1,234,567.885
L2	なし	あり	なし	なし	なし	-1234567.89
P2	なし	あり	あり	なし	あり	-1,234,567.89%

図 10. 編集コードの使用法

編集の値を指定しないと、Query 管理機能によって実行時デフォルトが提供されます。

Query 管理機能での列の順序

“順序” を指定して、生成された報告書の列を一定の順序に並べることができます。

“順序” の値の評価には、次の規則が適用されます。

- FORM の *n* 番目の列のデフォルトは *n* です。
- 値は 1 ～ 999 の任意の番号にすることができます。
- 番号が連続している必要はありません。
- “順序” 番号が同じ列は、書式に表示される順序と同じ順序で報告書に表示されます。

Query 管理機能での列の実行時デフォルト

Query の実行で抽出したデータ列を報告書用にフォーマット設定する必要があるとき、Query 管理機能は、データ・タイプ、列見出し、編集、および幅の各値として、システム提供のデフォルトを使用します。この状態は次の場合に生じます。

- 書式が指定されなかったか、あるいは抽出データについてデフォルトの書式を参照するために *SYSDFT が指定された場合。
- 指定された書式に報告書のフォーマット設定に必要な情報が入っていないか、あるいはブランク値または列表フィールドの脱落に関する警告付きで書式をインポートした場合。

注: 日付列または時刻列が含まれている Query で SAVE DATA AS コマンドを使用した場合には、日付および時刻のフィールドと関連するデフォルトの情報は、ユーザーと関連する情報であり、照会された元の表列と関連するデフォルトの情報**ではありません**。

Query 管理機能での列の見出しデフォルト

システムに対してフィールドを定義する場合、ファイル・データ用の列見出しのデフォルトを設定することができます。対話式データ定義ユーティリティー (IDDU) の使用時に他のテキストが指定されていない限り、フィールド名を使用してください。

明示の列見出しの入っていない書式を使用してデフォルト報告書を作成するときは、データベースからの列見出しが使用されます。

- データベースに列見出しがない場合は、選択した名前がデフォルトの列見出しになります。
- Query が列名によって列を選択する場合は、その列名がデフォルトの列見出しになります。
- Query がシステム列名を使用して列を選択する場合は、システム列名がデフォルトの列見出しになります。

選択した列がシステム列名と同じ明示の列見出しを持っている場合は、その列名が列見出しとして使用されます。

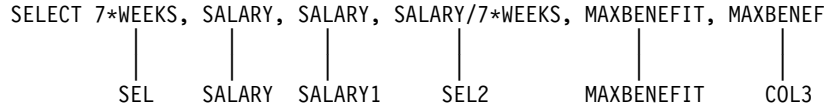
列見出しがデフォルトとならないようなファイルを定義することができます。計算済みデータおよびデフォルトが設定されていないファイル・フィールドの列見出しのデフォルトは、選択された列の実行時に生成される列固有の名前のセットから取られます。これらは、SAVE DATA 要求があった場合に新しいファイルを作成するために使用される列の名前となります。

これらの名前を作成するために、Query 管理機能は選択された列を、最初から最後まで順に処理します。選択された *n* 番目の列の固有の名前は、次のような方法によって作成されます。

1. ファイル (表) 定義 (または計算されたフィールドの場合は SEL) の列名が前に作成されたいずれの名前とも一致しない場合、それが作成される名前として使用されます。
2. 一致した名前が長すぎて次の使用可能番号に追加できない場合、その番号を COL に追加して名前を作成します。
3. 一致した名前が次の使用可能番号に追加できる長さである場合、その番号を列名に追加して名前を作成します。

注: 固有の名前にするために列の名前または COL に追加される番号は、最初が 1 で、次の番号は 2 であり、以下同様になります。

次の例は、特定の SELECT リストのために作成された固有の名前を示しています。



列名が重複するときも、番号の付加によって固有の列名になります。番号の付加によって固有の名前が 30 文字を超えてしまう場合は、最後の 5 文字が切り捨てられ、5 桁の番号が付け加えられます。たとえば、

```
SELECT ThisIsABigLongColumnNameExampl, ThisIsABigLongColumnNameExampl
```

この場合、次のような列名になります。

```
SELECT ThisIsABigLongColumnNameExampl, ThisIsABigLongColumnNameE00001
```

列の一部に前に定義された列見出しのデフォルトを持つものがある場合、形式化された報告書の列見出しは必ずしも固有のものにはなりません。このことは、名前により指定される書式にも *SYSDFT 書式にも該当します。

Query 管理機能での列の編集デフォルト

| 編集のデフォルトは、Query for iSeries など、システム上の他のデータ表示プロダクトで使用されるものと
| 同じになるように意図されています。文字データは変更されないか、または切り捨てられます (Query 管理
| 機能 CPI 編集コード C など)。浮動小数点データには、科学技術計算表記法が使用されます (Query 管理
| 機能 CPI 編集コード E など)。通常、編集デフォルトには数値フィールドに対応する Query 管理機能
| CPI 表示がなく、編集語、編集記述、および RPG 編集コードを組み込むことができます。

ファイル (表) データのデフォルトは、フィールド (列) がシステムに対して定義される時に設定されます。システム・レベルの値が計算済みデータの編集を決定します。これらの値は、変換可能メッセージから取り入れられ、必要な時にはいつでも、デフォルトの編集記述を作成するために使用されます。

また、変更可能システム・レベルの値も、数値データに適用される編集を変更することができます。たとえば、QDECFMT システム値を使用して、RPG 編集コード J に提供される編集を変更します。

| デフォルトの日付として、次の規則に従って最適の SQL 形式が使用されます。

| *表 7. デフォルトの SQL 形式*

| **OS/400 書式** | **SQL**
| *MDY | MM/DD/YYYY (USA)
| *YMD | YYYY-MM-DD (ISO)
| *DMY | DD.MM.YYYY (EUR)
| *JUL | YYYY-MM-DD (ISO)

| 使用される日付区切り記号は、ジョブの日付区切り記号にかかわらず、上記のデフォルトの SQL 形式にな
| ります。

データベース接続が同種である場合には、時刻にはジョブの時刻区切り記号が使用されます。データベース接続が異種であり、ジョブの時刻区切り記号がコロンである場合にはコロンが使用されます。それ以外の場合は時刻区切り記号はピリオドになります。

Query 管理機能での列の幅のデフォルト

幅のデフォルトは、列の中に表示したいすべてのものに十分なスペースを用意するように意図されています。特定の列の場合に、デフォルトは次のうちの最大のものになります。

- 列見出しの最長セグメントの長さ
- 編集済みデータの幅 (SUM 使用目的の総計値を列に表示しなければならない場合、生データの長さに 3 を加えて調整したもの)
- 9 の長さ (COUNT 使用目的の総計値をその列に表示しなければならない場合)

列見出しのデフォルトとして使用される列名は固有のものです。デフォルトの名前を固有にする必要があるときは、Query 管理機能は、列名の最後に接尾部として番号を加えます。この場合、その列名は最初に現れた時はそのままです。その後、この同じ列名が現れると、それらにはすべて番号が追加されます。番号は、すべての列名で順次割り当てられます。たとえば、

```
SELECT ID, ID, DEPT, JOB, ID, DEPT
```

この場合、デフォルトの見出しは次のようになります。

```
        ID  ID1  DEPT  JOB  ID2  DEPT3
```

デフォルトの列名を固有にすることができない場合、Query 管理機能は列名に対して COL*n* を割り当てます。たとえば、

```
SELECT ABCDEFGHIJKLMNOPQR, ABC, ABCDEFGHIJKLMNOPQR
```

この場合、デフォルトの見出しは次のようになります。

```
ABCDEFGHIJKLMNOPQR  ABC  COL1
```

Query 管理機能での PAGE フィールド

以下のフィールドは、報告書上の見出しおよびフッターを指定するために使用されます。 表 8 は、ページ・フィールドの属性に対するデフォルトおよび使用できる値を示しています。

表 8. ページ・フィールドのデフォルト

属性	デフォルト	使用できる値
見出しの前のブランク行数	0	0 ～ 999
フッターの前のブランク行数	2	0 ～ 999
見出しの後のブランク行数	2	0 ～ 999
フッターの後のブランク行数	0	0 ～ 999
見出しテキストの位置合わせ	CENTER	LEFT、CENTER、RIGHT
フッター・テキストの位置合わせ	CENTER	LEFT、CENTER、RIGHT

Query 管理機能でのページ・フィールドの見出し / フッターの前のブランク行

これらのフィールドはページ見出しまたはページ・フッターの前のブランク行の数を指示するフィールドであり、数値で指定しなければなりません。 0 ～ 999 の任意の数値を使用することができます。ページ見出しのデフォルトは *0*、ページ・フッターのデフォルトは *2* になります。各ページの見出しに先行して、常にページ排出が行われます。 “見出しの前のブランク行数” フィールドは、見出しとページの最上部との間のブランク行数を制御します。 “フッターの前のブランク行数” フィールドは、報告書本体と最初のフッター行との間のブランク行数を制御します。

ブランク行数は、ページ上に印刷される行数のカウントに含まれます。

Query 管理機能でのページ・フィールドの見出し / フッターの後のブランク行

これらのフィールドは、ページ見出しまたはページ・フッターの後のブランク行の数を指示します。 Query 管理機能では、このフィールドは数値のみとして定義されます。 0 ～ 999 の任意の数値を使用することができます。ページ見出しのデフォルトは *2* で、ページ・フッターのデフォルトは *0* になります。 “見出しの後のブランク行数” フィールドは、最後の見出し行と報告書本体との間のブランク行数を制御します。 “フッターの後のブランク行数” フィールドは、次のものを制御します。

- 最後のフッター行とページの終わりの間のブランク行数
- 最後のフッター行と、“日付および時刻” または “ページ番号” (あるいはその両方) が入っている行

ブランク行数は、ページ上に印刷される行数のカウントに含まれます。

“フッターの後のブランク行数” は、 “フッターの前のブランク行数” よりも優先します。報告書本体の後に余分なスペースが残されている報告書ページでは、 “フッターの後のブランク行数” の値が正しい行数となるように、余分なブランク行が挿入されます。

Query 管理機能でのページ・フィールドの見出しテキスト行

見出しテキスト行には、最大 999 の見出しテキストが含まれます。

Query 管理機能でのページ・フィールドの見出しテキスト行の行の値

行の値は見出しテキストの行位置を示します。

たとえば、行番号 1 および 5 のテキスト行を組み込んだ場合、テキストは報告書で次のように表示されます。

```
行番号 1 のテキスト
[ブランク行]
[ブランク行]
[ブランク行]
行番号 5 のテキスト
```

テキスト行番号 1 を指定し、その指定を後で繰り返した場合、行番号 1 は最後の指定が優先し、最初の指定は取り消されます。

Query 管理機能でのページ・フィールドの見出しテキストの位置合わせフィールド

位置合わせフィールドは、報告書行の中のページ見出しテキストの位置を制御します。次の値を使用することができます。

RIGHT
テキストを右にそろえます。

LEFT テキストを左にそろえます。

CENTER
テキストを中央にそろえます。

見出しのデフォルトは *CENTER* です。

Query 管理機能でのページ・フィールドの見出しテキスト・フィールド

これらのフィールドには、報告書のページ見出しとして表示されるテキストを入力できます。それぞれのテキスト行に最大 55 文字まで入力することができます。

テキスト行 *n* が非ブランク・テキストの最大行である場合、 *n* はフォーマット設定される行数を指示します。このことは、フォーマット設定された行の中に完全なブランク行が生じる結果になる場合にもあてはまります。

ページ見出しには、次の 4 つのタイプの特殊変数を含めることができます。見出しテキストの中でそのような変数を識別するために、変数の前にアンパーサンド (&) を付けてコーディングしなければなりません。

&col ここで、*col* は、列名、システム列名、または列番号です。 *&col* は、指定された列のページの最初の値に割り当てられます。

列の折り返しが指定されている場合を除き、変数 &col は、編集コードの指定に従ってフォーマット設定されます。その後、編集コードの指定は無視されます。必要に応じて、編集されたデータは、該当の列の幅に合わせて切り捨てられます。

列の名前は、SQL から戻される列の名前です。これは英大文字でのみ指定することができます。計算される列または複写される列の列名を判別するには、 71 ページの『Query 管理機能での列の見出しデフォルト』を参照してください。

列名が DATE、TIME、PAGE であるか、または、列名に $、#、@ または引用符が入っているときは、その列は列番号で参照しなければなりません。

列の番号は、 SQL ステートメント Query またはプロンプト Query (指示照会) から戻される列の順序によって決まります。順序フィールド内の順序によって列の番号が決まることはありません。 *&col* はページの切れ目ごとにセットされます。

&DATE

Query 管理機能は、この値を現在の日付に置き換えます。

&TIME

Query 管理機能は、この値を現在の時刻に置き換えます。 OS/2 では、現在の時刻は活動状態プロファイルの編集コード、またはシステムの国または地域のコードに基づいた形式になります。

&PAGE

Query 管理機能は、この値を現在のページ番号に置き換えます。ページ番号の形式は、1 ～ 9999 の範囲の 4 桁の番号で、先行のゼロは消去されます。 9999 に達した後は、カウンターは折り返して 0 に戻り、後続のページごとに数値が増加し、先行ゼロの消去は行いません。

報告書が印刷される時に、フォーマット設定仕様に従ってフォーマット設定されたページ見出しが、各ページの最上部に表示されます。列の折り返しが指定されている場合を除き、変数 *&col* のフォーマット設定は編集コードの指定に従って行われます。ある列について列の折り返しが指定されている場合でも、データがテキストにフォーマット設定される時には無視されます。

すべての変数は報告書の作成時に分析解決されます。

Query 管理機能でのページ・フィールドのフッター・テキスト行

フッター・テキストには、最大 999 のフッター・テキスト行が含まれます。

Query 管理機能でのページ・フィールドのフッター・テキスト行の行の値

行の値はフッター・テキスト行の行位置を示します。

たとえば、行番号 1 および 5 のテキスト行を組み込んだ場合、テキストは報告書で次のように表示されます。

```
行番号 1 のテキスト
[ブランク行]
[ブランク行]
[ブランク行]
行番号 5 のテキスト
```

テキスト行番号 1 を指定し、その指定を後で繰り返した場合、行番号 1 は最後の指定が優先し、最初の指定は取り消されます。

Query 管理機能でのページ・フィールドのフッター・テキストの位置合わせフィールド

位置合わせフィールドは、報告書行の中でのページ・フッター・テキストの位置を制御します。次の値を使用することができます。

RIGHT

テキストを右にそろえます。

LEFT テキストを左にそろえます。

CENTER

テキストを中央にそろえます。

PAGE フィールド

フッターのデフォルト値は、*center* です。

Query 管理機能でのページ・フィールドのフッター・テキスト・フィールド

これらのフィールドによって、報告書に表示するフィールドのテキストを入力できます。それぞれのテキスト行に最大 55 文字まで入力することができます。見出しテキスト・フィールドの項で説明した変数を使用することができます。報告書のフォーマット設定時に、変数は適切な値に置き換えられます。指定された列について、変数 &col がページの最後の値に割り当てられます。報告書が印刷される時に、フォーマット設定仕様に従ってフォーマット設定されたページ・フッターが、各ページの最下部に表示されます。列の折り返しが指定されている場合を除き、変数 *&col:* のフォーマット設定は編集コードの指定に従って行われます。列の折り返しを指定している場合でも、データをテキストにフォーマット設定する時には無視されます。形式化されたページ・フッターは、表示報告書の最下部に 1 回表示されます。

テキスト行 *n* が非ブランク・テキストの最大行である場合、 *n* はフォーマット設定される行数を指示します。このことは、フォーマット設定された行の中に完全なブランク行が生じる結果になる場合にもあてはまります。

Query 管理機能での FINAL TEXT フィールド

以下のフィールドは、報告書に表示される最終テキストを指定するために使用されます。 表 9 は最終テキスト・フィールドの属性について、デフォルトおよび使用できる値を示しています。

表 9. 最終テキスト・フィールドのデフォルト

属性	デフォルト	使用できる値
改ページ	NO	YES、NO
最終合計行表示	1	1 ～ 999 または NONE
テキストの前のブランク行数	0	0 ～ 999 または BOTTOM
最終テキストの位置合わせ	RIGHT	LEFT、CENTER、RIGHT

Query 管理機能での最終テキスト・フィールドの改ページ

このフィールドは、印刷時にページを改めて報告書の後続部分 (最終テキスト) を印刷するかどうかを指示します。デフォルトは *no* です。改ページに YES を指定した場合には、最終テキストは新しいページにフォーマット設定されます。

Query 管理機能で最終テキスト・フィールドの行に最終合計を書き込む

このフィールドは、最終合計をフォーマット設定するかどうか、その場合に報告書最終テキスト内の縦方向のどの位置にそれを位置づけるかを指示します。 1 ～ 999 の数値または NONE を使用することができます。 NONE は最終合計データを表示しないことを指示します。デフォルトは *1* です。

1 ～ 12 の値は、合計データをフォーマット設定しなければならない最終テキスト内の相対行番号を指示します。これは、厳密に縦方向の配置になります。行内の横方向の配置については、最終合計は常に合計している列の下にフォーマット設定されます。合計の使用目的とともに列の幅が指定されていない場合には、この値は無視されます。

Query 管理機能での最終テキスト・フィールドのテキストの前のブランク行

このフィールドは、報告書の本体と最終テキストの最初の行との間のブランク行の数を指示します。 0 ～ 999 の任意の数値を使用することができます。デフォルトは *0* です。 BOTTOM を指定することもできます。これは最終テキストを印刷ページの最下部に位置づけます。 BOTTOM によって、最終テキストがそのページのページ・フッター・テキストの指定位置の直前に位置するように、何行かのブランク行が挿入されます。現在のページに最終テキストのための十分なスペースがない場合には、最終テキストは次のページの最下部に入れられます。

Query 管理機能での最終テキスト行の行フィールド

このフィールドは最終テキスト行の行位置を指示します。

たとえば、行番号 1 および 5 のテキスト行を組み込んだ場合、テキストは報告書で次のように表示されます。

```
行番号 1 のテキスト
[ブランク行]
[ブランク行]
[ブランク行]
行番号 5 のテキスト
```

テキスト行番号 1 を指定し、その指定を後で繰り返した場合、行番号 1 は最後の指定が優先し、最初の指定は取り消されます。

Query 管理機能での最終テキスト・フィールドの位置合わせフィールド

このフィールドは、報告書行内で最終テキストの位置を制御します。これは左マージンと最初の合計列との間の位置合わせを指します。報告書に最終合計データが含まれない場合は、表示または印刷される報告書の幅全体での位置合わせを指します。次の値を使用することができます。

RIGHT
テキストを右にそろえます。

LEFT テキストを左にそろえます。

CENTER
テキストを中央にそろえます。

最終テキストのデフォルトは *RIGHT* です。関連した最終テキストがない場合には、位置合わせの値は無視されます。

Query 管理機能での最終テキスト・フィールドの最終テキスト行

最終テキストに使用できる行数は 999 行です。これらの行上に何を表示するかを指定します。各テキスト行には最大 55 文字を入力することができます。*&col:* が、使用できるただ一つの変数です。変数 &col は、指定された列の最後のレコードの最後の値に割り当てられます。列の折り返しが指定されている場合を除き、変数 &col は、編集コードの指定に従ってフォーマット設定されます。その後、編集コードの指定は無視されます。必要に応じて、編集されたデータは、該当の列の幅に合わせて切り捨てられます。

テキスト行 *n* が非ブランク・テキストの最大行である場合、 *n* はフォーマット設定される行数を指示します。このことは、フォーマット設定された行の中に完全なブランク行が生じる結果になる場合にもあてはまります。

FINAL TEXT フィールド

最終合計表示行 の値が *m* で、かつ *m* > *n* である場合、報告書には *m* 行の最終行がフォーマット設定されます。

Query 管理機能での BREAK フィールド

切れ目レベル 1 ～ 6 に関する情報を指定することができます。また、適切な FORM フィールド番号を選択して、エクスポート FORM を変更または指定することもできます。 『第 8 章 Query 管理機能でのオブジェクトのエクスポートとインポート』を参照してください。切れ目レベルのそれぞれについて、別個のフィールド番号があります。切れ目レベルのそれぞれについて、同様の方法で指定します。オプションの各セットは相互に独立しています。

表 10 は、切れ目フィールドの属性に対するデフォルトおよび使用できる値を示しています。

表 10. 切れ目フィールドのデフォルト

属性	デフォルト	使用できる値
切れ目で改ページ	NO	YES、NO
フッターで改ページ	NO	YES、NO
列見出しの繰り返し	NO	YES、NO
見出しの前のブランク行数	0	0 ～ 999
フッターの前のブランク行数	0	0 ～ 999 または BOTTOM
見出しの後のブランク行数	0	0 ～ 999
フッターの後のブランク行数	1	0 ～ 999
切れ目合計表示行	1	1 ～ 999 または NONE
切れ目見出しテキストの位置合わせ	LEFT	LEFT、CENTER、RIGHT
切れ目フッター・テキストの位置合わせ	RIGHT	LEFT、CENTER、RIGHT

Query 管理機能での切れ目フィールドにおける、切れ目で改ページ / フッターで改ページ

これらのフィールドは、報告書の後続部分を新しいページで始めるかどうか指示します。デフォルトは、どちらの場合も *NO* です。「切れ目で改ページ」フィールドに YES を指定した場合には、切れ目のメンバー行が新しいページにフォーマット設定されます。切れ目見出しを指定した場合には、それが新しいページの切れ目メンバー行の前に入れられます。「フッターで改ページ」フィールドに YES を指定した場合には、切れ目フッターが次のページにフォーマット設定されます (フッターが存在している場合)。

Query 管理機能での切れ目フィールドの列見出しの繰り返し

このフィールドは、特定の切れ目レベルのメンバー行の上で列見出しを繰り返すかどうかを指示します。*NO* がデフォルトです。

報告書をページングまたは印刷する時には、列見出しが常に画面またはページの最上部に現れます。特定の切れ目の列見出しの繰り返しに Yes を指定すると、このような列見出しに加えて、切れ目の先頭に別の見出しのセットが表示されます。これは、切れ目見出しテキストの有無にかかわらず行われます。しかし、切れ目が印刷ページの最上部で始まる場合には、切れ目メンバー行の前の列見出しのセットだけがフォーマット設定されます。

Query 管理機能での切れ目フィールドにおける、見出し / フッターの前のブランク行

これらのフィールドは、切れ目見出しまたは切れ目フッターの前に表示されるブランク行数を指示します。切れ目見出しが指定されていない場合は、このフィールドの値は切れ目メンバー行の前のブランク行数になります。使用することができる値は 0 ～ 999 の数です。見出しとフッターのどちらの場合もデフォルトは *ゼロ* です。

見出しの前のブランク行数フィールドには、数値だけを入れることができます。

切れ目フッターの場合には、BOTTOM を指定することもできます。印刷報告書にのみ適用可能な BOTTOM によって、切れ目フッターの位置は印刷報告書の現在のページの最下部になります。 BOTTOM によって、テキストがそのページのページ・フッター・テキストの指定位置の直前に位置するように、何行かのブランク行が挿入されます。このことは、次の行を次のページに印刷しなければならないため、ページ排出が起こることも意味しています。

Query 管理機能での切れ目フィールドにおける、見出し / フッターの後のブランク行

これらのフィールドは、切れ目見出しまたは切れ目フッターの後のブランク行数を指示します。切れ目見出しが指定されていない場合には、このフィールドの値はデフォルトで、切れ目メンバー行の後のブランク行数になります。フッターが指定されていない場合には、切れ目フッターは切れ目メンバー行の後のブランク行に組み込まれます。 0 ～ 999 の任意の数値を使用することができます。見出しのデフォルトは *0*、フッターのデフォルトは *1* になります。

Query 管理機能での切れ目フィールドの切れ目合計表示行

このフィールドは、切れ目合計をフォーマット設定するかどうかを指示します。また設定する場合には、切れ目フッター・テキストに対する相対的な位置をどこに決めるかを指示します。値には 1 ～ 999 または NONE を使用することができます。 NONE は、その切れ目については切れ目合計情報を表示しないことを指示します。デフォルトは *1* です。使用される数値は、切れ目合計とともに表示される切れ目フッター・テキストの行番号に対応します。

この配置は、厳密に縦方向の配置です。行内の横方向の配置については、切れ目合計は常に合計されている列の下にフォーマット設定されます。合計の使用目的とともに列の幅が指定されていない場合、合計をとる列がないためこの値は無視されます。

切れ目見出しテキスト行

見出しテキスト行フィールドには、最大 999 の見出しテキスト行が含まれます。

Query 管理機能での切れ目見出しテキスト行の値

行の値は見出しテキストの行位置を示します。

たとえば、行番号 1 および 5 のテキスト行を組み込んだ場合、テキストは報告書で次のように表示されます。

```
行番号 1 のテキスト
[ブランク行]
[ブランク行]
[ブランク行]
行番号 5 のテキスト
```

BREAK フィールド

テキスト行番号 1 を指定し、その指定を後で繰り返した場合、行番号 1 は最後の指定が優先し、最初の指定は取り消されます。

Query 管理機能での切れ目見出しテキストの位置合わせフィールド

このフィールドは、報告書行内の切れ目見出しテキストの位置を制御します。次の値を使用することができます。

RIGHT
テキストを右にそろえます。

LEFT テキストを左にそろえます。

CENTER
テキストを中央にそろえます。

デフォルトは *LEFT* です。位置合わせは、表示または印刷される報告書の全体の幅に基づいて行われます。

Query 管理機能での切れ目見出しの切れ目見出しテキスト

切れ目見出しテキストには、1 行あたり 55 文字を使用することができます。変数として使用することができるのは *&col:* だけです。変数 *&col:* が、指定された切れ目グループの最初の値に割り当てられます。

列の折り返しが指定されている場合を除き、変数 &col は、編集コードの指定に従ってフォーマット設定されます。その後、編集コードの指定は無視されます。必要に応じて、編集されたデータは、該当の列の幅に合わせて切り捨てられます。

テキスト行 *n* が非ブランク・テキストの最大行である場合、 *n* はフォーマット設定される行数を指示します。このことは、フォーマット設定された行の中に完全なブランク行が生じる結果になる場合にもあてはまります。

Query 管理機能での切れ目フッター・テキスト行

フッター・テキスト行には、最大 999 のフッター・テキスト行が入ります。

Query 管理機能での切れ目フッター・テキスト行の値

行の値はフッター・テキスト行の行位置を示します。

たとえば、行番号 1 および 5 のテキスト行を組み込んだ場合、テキストは報告書で次のように表示されます。

```
行番号 1 のテキスト
[ブランク行]
[ブランク行]
[ブランク行]
行番号 5 のテキスト
```

テキスト行番号 1 を指定し、その指定を後で繰り返した場合、行番号 1 は最後の指定が優先し、最初の指定は取り消されます。

Query 管理機能での切れ目フッター・テキストの位置合わせフィールド

このフィールドは、報告書行内のページ・フッター・テキストの位置を制御します。次の値を使用することができます。

RIGHT
テキストを右にそろえます。

LEFT テキストを左にそろえます。

CENTER
テキストを中央にそろえます。

デフォルトは *RIGHT* です。位置合わせは、左側の最初の文字と最初の合計列との間のスペースを指します。報告書に切れ目合計データが含まれない場合は、表示または印刷される報告書の幅全体での位置合わせを指します。

Query 管理機能での切れ目フッターの切れ目フッター・テキスト・フィールド

変数として使用できるのは、*&col* だけです。変数 *&col* は、指定された列の切れ目グループの最後の値に割り当てられます。 1 行あたり 55 文字まで使用することができます。切れ目フッター・テキストであると見なされるのは、初めて非ブランク・フィールドが後に続いているブランク・フィールド (ただし、このフィールドは含めない) までの行です。

テキスト行 *n* が非ブランク・テキストの最大行である場合、 *n* はフォーマット設定される行数を指示します。このことは、フォーマット設定された行の中に完全なブランク行が生じる結果になる場合にもあてはまります。

切れ目合計表示行 の値が *m* で、かつ *m* > *n* である場合、報告書には *m* 行の切れ目行がフォーマット設定されます。

Query 管理機能での OPTIONS フィールド

オプション・フィールドを使用すれば、各種の報告書フォーマット設定オプションを指定することができます。

表 11 は、オプション・フィールドの属性に対するデフォルトと、使用できる値を示しています。

表 11. オプション・フィールドのデフォルト

属性	デフォルト	使用できる値
明細行間隔	1	1 ～ 4
切れ目列の一括表示	YES	YES、NO
デフォルトの切れ目テキスト	YES	YES、NO
列の折り返し行の同一ページ表示	YES	YES、NO
列見出し区切り記号	YES	YES、NO
切れ目合計区切り記号	YES	YES、NO
最終合計区切り記号	YES	YES、NO

Query 管理機能でのオプション・フィールドの明細行間隔

このフィールドは、報告書の各明細行の間に要求する間隔を指示します。このフィールドに入力することができるのは数値だけです。受け入れ可能な値は 1、2、3、または 4 です。 1 は 1 行間隔、2 は 2 行間隔などとなります。デフォルトは *1* です。

Query 管理機能でのオプション・フィールドの切れ目列の一括表示

切れ目の使用目的コードを列の 1 つに割り当てる場合には、このフィールドを使用して、切れ目列の値を報告書に表示する時点を決定します。デフォルトは *YES* であり、切れ目列の値は変更された時にだけ表示されます。このフィールドが NO であると、切れ目列の値は報告書のすべての明細行に表示されます。

Query 管理機能でのオプション・フィールドのデフォルト切れ目テキスト

このフィールドは、報告書にデフォルトの切れ目テキストを組み込むことを要求するかどうかを指示します。デフォルトは *YES* です。切れ目合計行にマークを付けるためには、デフォルト切れ目テキストを使用してください。切れ目合計行には、それぞれの切れ目レベルによって 1 つまたは複数のアスタリスクが付きます。つまり、レベル番号が最も大きい切れ目レベル・テキストには 1 つのアスタリスク、 2 番目に大きいレベル番号のテキストには 2 つのアスタリスクというようになります。たとえば、報告書に 2 つの制御の切れ目 (BREAK2 と BREAK4) がある場合、Query 管理機能は、レベル 4 の切れ目を示すマークとしてアスタリスクを 1 つ、レベル 2 の切れ目を示すマークとしてアスタリスクを 2 つ生成します。

Query 管理機能でのオプション・フィールドの列の折り返し行の同一ページ表示

報告書に 1 つまたは複数の列に列折り返しを指定した場合、このフィールドを使用して、折り返し列を 2 ページに分割してよいかどうかを決定します。このフィールドのデフォルトは *YES* です。これにより、(折り返される列がページの長さより短ければ) その折り返される列が 2 つのページに分割されることが防止されます。このフィールドを NO にすると、折り返された列を別のページ間で分割することができます。

Query 管理機能でのオプション・フィールドの列見出し区切り記号

このフィールドは、列見出し区切り記号 (破線) を報告書に表示するかどうかを指示します。デフォルトは *YES* です。

Query 管理機能でのオプション・フィールドの切れ目合計区切り記号

このフィールドは、切れ目合計区切り記号 (破線) を報告書に表示するかどうかを指示します。デフォルトは *YES* です。合計列がない時に指定された値が YES の場合、ブランク区切り行が生成されます。

Query 管理機能でのオプション・フィールドの最終合計区切り記号

このフィールドは、最終合計区切り記号 (等号を連ねた線) を報告書に表示するかどうかを指示します。デフォルトは *YES* です。合計列がない時に指定された値が YES の場合、ブランク区切り行が生成されます。

第 7 章 Query 管理機能での呼び出し可能インターフェース

DB2 UDB for iSeries Query 管理機能の呼び出し可能インターフェース (CI) の提供する機能により、アプリケーション・プログラムは、Query 管理機能インターフェースを呼び出して、Query 管理機能の各機能を実行できます。 DB2 UDB for iSeries Query 管理機能が完了すると、戻りコードと状況情報が呼び出しプログラムに戻されます。 Query 管理機能では、CI は、ILEC、C、COBOL および RPG の各言語で使用できます。

CI は次の要素で構成されています。

- Query 管理機能 CI マクロ

  Query 管理機能のマクロ命令は、Query 管理機能 CI モジュールを呼び出すアプリケーション・プログラムをコンパイルする時に使用される、組み込みファイルとマクロ・ファイルで構成されます。これらのファイルには連絡域の構造の宣言、および連絡域の構造の更新とアクセスに必要なすべての固定情報が入っています。また、これらは Query 管理機能 CI モジュールに対する異なったプログラミング言語からの標準インターフェースを提供します。このインターフェースは、プログラミング言語と照会機能との間で、プログラム変数の共通の記憶とアクセスを行うものです。 Query 管理機能がサポートする言語ごとに、 1 つの Query 管理機能 CI マクロまたは組み込みファイルが用意されています。

  表 12 は、Query 管理機能で使用できるそれぞれのマクロ組み込みパッケージです。マクロ組み込みは、表 13 の Query 管理機能で示されているように、組み合わせパッケージでも使用できます。

表 12. マクロ組み込みパッケージ

言語	ライブラリー	ファイル	メンバー
ILEC	QCLE	H	DSQCOMMC
COBOL	QLBL	QILBINC	DSQCOMMB
RPG	QRPG	QIRGINC	DSQCOMMR

表 13. 組み合わせマクロ組み込みパッケージ

言語	ライブラリー	ファイル	メンバー
ILE C	QSYSINC	H	DSQCOMMC
OPM COBOL	QSYSINC	QLBLSRC	DSQCOMMB
OPM RPG	QSYSINC	QRPGSRC	DSQCOMMR

  これらの組み込みファイルまたはマクロ・ファイルを使用するアプリケーション・プログラムをコンパイルする前に、コンパイラーによって使用されるデフォルトの組み込みファイルにメンバーをコピーしてください。これにより、組み込みファイルをライブラリー名またはファイル名で修飾しないでアプリケーション・プログラムで使用することができるようになり、より高度な移植性が維持されます。

  ライブラリー名およびファイル名で組み込みファイルを修飾するアプリケーション・プログラムをコーディングすることができます。これにより、最新バージョンの組み込みファイルを持つプログラムをコンパイルすることができます。

- Query 管理機能

  Query および報告書作成サービスを提供します。

- Query 管理機能 CI モジュール

インターフェースが提供する、Query 管理機能の各機能へのアクセスを可能にするモジュールです。これらのモジュールは次のとおりです。

DSQCICE
拡張パラメーター・リスト用の ILEC および C 言語インターフェース・モジュール

DSQCIC
非拡張パラメーター・リスト用の ILEC および C 言語インターフェース・モジュール

DSQCIB
COBOL 言語インターフェース・モジュール

DSQCIR
RPG 言語インターフェース・モジュール

Query 管理機能での呼び出し可能インターフェースの説明

呼び出し可能インターフェース (CI) は、プログラミング言語が Query 管理機能コマンドを実行するために使用できるインターフェースです。すべての Query 管理機能コマンドが、この CI を介してサポートされます。

Query 管理機能コマンドを実行するためには、プログラムは呼び出しを出して、プログラムと Query 管理機能との間で通信を開始します。この呼び出しは、Query 管理機能提供ルーチンに対して行われます。

呼び出しプログラムは、最初に開始の呼び出しを出した後、 1 つまたは複数の Query 管理機能コマンドを出すことができます。処理する各 Query 管理機能コマンドごとに、Query 管理機能提供ルーチンに対する呼び出しが必要です。 Query 管理機能コマンドが完了するたびに、その呼び出しの処理についての情報が戻りコードとして呼び出し元に戻されます。コマンドの処理についてのその他の情報は、CI によって収集されて共用変数内に保管されます。制御が呼び出しアプリケーションに戻ると、これらの変数を参照することができるようになります。

プログラムが Query 管理機能の各機能を使用する必要がなくなったら、そのプログラムと Query 管理機能との通信を終了するための呼び出しを出します。この呼び出しは、Query 管理機能提供ルーチンに対して行われます。

以上の処理には、次のような考慮事項があります。

- Query 管理機能に対する呼び出しがアプリケーションに戻るのは、コマンドの処理が完了した時だけです。
- 呼び出しを処理していない時には、CI は静止状態になっています。
- アプリケーションへの通信は、すべて変数プールまたはインターフェース連絡域に記憶された変数データおよび戻りコードを介して行われます。
- コマンドは大文字の英字でコーディングしなければなりません。
- 渡されるコマンドの長さは、最大 256 バイトまででなければなりません。

次の図は、CI が構造全体のどこに収まるかを示しています。

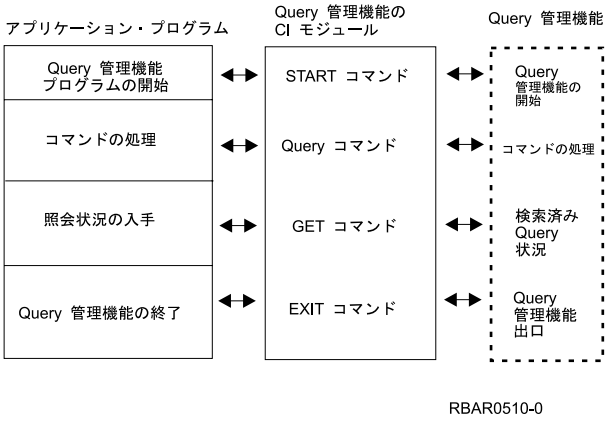


図 11. 呼び出し可能インターフェースの図解

アプリケーションが CI を介して Query 管理機能コマンドを正常に処理した場合は、その結果は、通常、画面表示をしないでコマンドをオンラインに処理した場合と同じになります。

Query 管理機能 CI は、サポートされている言語ごとに固有のインターフェース通信マクロを準備しています。通信マクロには、場合に応じて次の定義が入っています。

1. インターフェース連絡域 (DSQCOMM)
2. 戻りコード
3. Query 管理機能に対する呼び出しインターフェース

Query 管理機能におけるインターフェース連絡域 (DSQCOMM)

Query 管理機能の CI 連絡域は、すべての CI 呼び出しで必要です。 CI 連絡域用の記憶域は、Query 管理機能 CI を使用するプログラムによって割り振られます。

START コマンドは、Query 管理機能の固有なインスタンスを設定します。 START コマンド処理の一部として、CI 連絡域が Query 管理機能によって更新されます。**アプリケーション・プログラムは、絶対に CI 連絡域を変更してはなりません。** START コマンドの後に続く呼び出しは、すべて Query 管理機能のインスタンスに対応する CI 連絡域のアドレスを渡さなければなりません。正しい連絡域を指示するのは、ユーザー・プログラムの責任です。

CI 連絡域は CI 通信マクロによって記述されます。サポートされる言語ごとに固有な通信マクロがあります。アプリケーションに移植性を持たせるためには、値は等価値によってではなく、変数名によって参照しなければなりません。この値はシステム間で異なる場合があるためです。

CI 連絡域 DSQCOMM には次の情報が含まれますが、呼び出しプログラムでこれを変更しては**なりません**。

- 戻りコード

  コマンドが実行された後の Query 管理機能の状況を示します。
- インスタンス ID

  START コマンドの処理過程で Query 管理機能が設定する ID です。
- 完了メッセージ ID

  コマンドがユーザー端末で出された場合、ユーザー端末に表示されるメッセージのメッセージ ID が入ります。
- Query メッセージ ID

コマンドの結果、Query 処理が行われた場合、Query メッセージのメッセージ ID が入ります。これは、ジョブ・ログで表示されるはずのメッセージのメッセージ ID です。このメッセージ ID は環境によって異なります。アプリケーションのデバッグに役立てるために用意されたものであり、移植可能なアプリケーションにおいては、これに依存すべきではありません。

- エラーの場合の START コマンド・パラメーター

  パラメーター・エラーのために START コマンドが失敗した場合には、エラーのあったパラメーターが入ります。
- 取り消し標識

  Query 管理機能によるコマンドの実行中に、ユーザーがそのコマンド処理を取り消したかどうかを示します。
- Query の派生

  Query 情報が Query for iSeries 定義から派生したかどうかを示します。
- 書式の派生

  書式情報が Query for iSeries 定義から派生したかどうかを示します。

Query 管理機能 CI の戻りコード

戻りコードは、Query 管理機能 CI に対する各呼び出しの後で戻されます。戻りコードの値は、CI 通信マクロの中に記述されます。アプリケーションに移植性を持たせるためには、値は等価値によってではなく、変数名によって参照しなければなりません。この値はシステム間で異なる場合があるためです。

CI からの戻りコードには、次のものがあります。

- 要求は正常に処理されました。
- コマンドは処理されましたが、警告条件を伴っています。
- コマンドは正しく処理されませんでした。
- 重大エラー: 該当のインスタンスについて、Query 管理機能セッションは終了しました。

各戻りコードの定義については、93 ページの『Query 管理機能 CI での C 言語インターフェース』、101 ページの『Query 管理機能 CI での COBOL 言語インターフェース』、および 107 ページの『Query 管理機能 CI での RPG 言語インターフェース』を参照してください。

Query 管理機能 CI の戻り変数

CI から制御が戻る時、Query 管理機能コマンドの完了についての情報の入った変数がセットされます。呼び出しプログラムは、変数プールからこの戻り変数を入手することができます。

この変数は、GET コマンドを使用して名前によって記号的に参照し、Query 管理機能変数プールから入手します。

Query 管理機能 CI のコマンド・メッセージ変数

対話式ユーザーによってコマンドが開始されると、正常に完了したか、あるいは処理中にエラーが起こったかを示すメッセージが画面に表示されます。アプリケーションもコマンド・メッセージ変数を介して、これと同じ情報を使用することができます。 CI を使用して Query 管理機能コマンドを処理するたびに、その完了時に次のコマンド・メッセージ変数が提供されます。

DSQCIMNO

メッセージ番号が入ります。このメッセージ番号は DSQCOMM 域にも戻されます。

DSQCIMSG
対話式にユーザーに表示される第 1 レベル・メッセージのテキストが入ります。

Query 管理機能 CI の Query メッセージ変数

Query 管理機能の Query の処理時にエラーが起きた場合、問題分析に役立つ Query メッセージが作成されることがあります。たとえば、RUN QUERY コマンドの処理中にエラーが起きたとします。この場合、詳細説明が提供されることがあります。 Query 管理機能メッセージ変数は、次のものから成ります。

DSQCIQNO
メッセージ番号が入ります。このメッセージ番号は DSQCOMM 域にも戻されます。

DSQCIQMG
エンド・ユーザーに対話式に表示される第 1 レベル・メッセージのテキストが入ります。

DSQCISQL
DB2 Universal Database for iSeries からの SQL 戻りコードがあれば入ります。

Query 管理機能 CI のコマンド構文拡張

Query 管理機能が COBOL などの高水準言語に対する変数サポートを提供するためには、Query 管理機能は、呼び出し側のプログラム・ストレージにアクセスする必要があります。 (Query 管理機能の変数サポートの説明については、『Query 管理機能 CI の拡張変数サポート』を参照してください。) DB2 UDB for iSeries Query 管理機能コマンドの拡張によって、呼び出し側のプログラム・ストレージに対するアクセスを必要とするコマンドが使用できるようになります。このコマンド拡張は、Query 管理機能コマンド・オプションを指定するための別の方法です。コマンド拡張は GET、SET、および START コマンドに使用されます。これらのコマンドをサポートするために、ユーザー・プログラム域に対するアクセスが必要となるためです。

Query 管理機能 CI の拡張変数サポート

変数とは、Query 管理機能の中の名前のついたエンティティーであり、値を割り当てることができるものです。拡張変数サポートによって、アプリケーションで Query 管理機能内にグローバル変数を定義することができます。

変数は SQL Query の中で置換値として使用したり、また CI を使用時に使用することができます。変数が作成された後は、Query 管理機能セッションの間中、そのセッションで使用することができます。アプリケーション定義の変数に加えて、Query 管理機能は、プロダクト変数のセットも維持します。これらの変数は、SQL Query および CI でも使用することができます。

グローバル変数の設定例については、『付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用』を参照してください。

Query 管理機能 CI での変数の作成

変数は SQL Query 処理中に最初に参照される時点で、実行時に暗黙に作成されます。また、SET GLOBAL コマンドを使用して特定の値にセットされる時には、明示的に作成されます。変数が暗黙に作成される場合、変数値が存在するのは RUN QUERY コマンドの処理中だけです。

Query 管理機能 CI での変数の参照

変数への参照は、GET GLOBAL コマンドを使用して SQL Query またはユーザー・プログラムに変数名を指定することによって行います。 SQL Query で変数名を参照する場合は、Query 管理機能がそれを変数と認識できるように、変数名の接頭部にアンパーサンド (&) を付ける必要があります。たとえば、

拡張変数サポート

```
SELECT * FROM &TNAME
```

値 &TNAME は、Query 管理機能変数と見なされます。

Query 管理機能 CI での変数名

呼び出し可能インターフェースにまたがる SQL Query で変数を使用する場合、次の規則が適用されます。

- 名前には、以下の文字を含めることができます。
  - **英字**。英字とは任意の 1 バイト文字です (A〜Z、または各言語のアルファベット文字)。
  - **アラビア数字** (0〜9)
  - **下線** (_)
- 変数名は英字で始めなければなりません (SQL Query で使用される変数名の例外については下記を参照してください)。
- 名前は長さが 18 文字以下でなければなりません。
- SQL Query で使用される変数名は、前にアンパーサンド (&) を付け、その次の文字を 1 バイト文字セットの英字にしなければなりません。アンパーサンドは名前として許される 30 文字には含まれません。アンパーサンド文字は、変数名の始めを区切るためのものであることに注意してください。変数名に複数のアンパーサンドを使用することはできません。アンパーサンドごとに区切られて、別の変数名が始まることになるためです。
- ユーザー定義の変数を DSQ で始めてはなりません。

以下は有効な変数名です。

```
SQL Query では                GET/SET コマンドでは
--------------------          -------------------------
&I_OWE_YOU                    I_OWE_YOU
&MYVAR123                     MYVAR123
&THIS_IS_A_BIG_NAM            THIS_IS_A_BIG_NAME
```

Query 管理機能 CI での変数値

変数値には、文字または整数を使用することができます。次の規則があります。

Query 管理機能 CI での文字変数

- 文字変数値は、55 文字までの長さの任意の値で構成されます。
- 文字変数をより小さい文字フィールドへ GET することが可能です。その文字ストリングは 55 文字より後は切り捨てられます。
- 文字変数をより大きい文字フィールドへ GET することが可能です。その文字ストリングは左寄せされ、ブランクが埋め込まれます。 C ストリングの終わりにあるヌル文字は移動されません。
- GET が C 言語の呼び出し可能インターフェースを介して実行された場合には、ストリングの終わりに C のヌル文字が挿入されます。

Query 管理機能 CI での整変数

- 整変数値は、4 バイトの長さでなければなりません。 4 バイト以外の長さの整数の SET または GET を試みた場合、結果はエラーになります。
- 整変数値は符号付きと見なされます。
- 値を SQL Query で使用するときは、SQL 規則に従わなければなりません。

- 整数値は、SQL ステートメントへの置き換えの前に、前後にブランクのない文字ストリングに変換されます。暗黙の結果フィールドの幅が必要である場合、あるいは SQL ステートメントで結果フィールドの定義中に変数置換を使用している場合には、整変数は使用しないでください。

Query 管理機能 CI での定義変数

Query 管理機能は、ユーザー・プログラムに有用なグローバル変数を提供します。 Query 管理機能変数の現行セットを使用して、Query 管理機能の環境および特定のオブジェクトの現在の状況を判別することができます。 Query 管理機能変数は、ユーザー、プログラム、またはプロシージャーによって変更することはできません。これらの変数のサブセットは、 START コマンドに指定された Query コマンド・プロシージャーを使用してセットすることができます (47 ページの『Query 管理機能での START』を参照)。

Query 管理機能では、以下の変数を使用することができます。ここでは、変数の長さは最大長で示してあります。

DSQAAUTH

現行の接続許可 ID。この名前には、ジョブが実行されているユーザー・プロファイルの名前が入っています。

タイプ 文字

長さ 10

値 -

DSQOAUTH

Query コマンドによって作成されたオブジェクトに与えられるデフォルトのオブジェクト共通権限。値の説明については、 47 ページの『Query 管理機能での START』を参照してください。

タイプ 文字

長さ 10

値

- *LIBCRTAUT
- *EXCLUDE
- *ALL
- *USE
- *CHANGE
- 権限リスト名

DSQSNAME

使用される命名規則。この値の説明については、 47 ページの『Query 管理機能での START』を参照してください。

タイプ 文字

長さ 4

値

- *SAA
- *SYS

DSQAPRNM

現行のデフォルト・プリンター。

拡張変数サポート

タイプ 文字

長さ 10

値

- *SAME
- *JOB
- プリンター装置名

DSQCATTN

最後のコマンド取り消し標識。

タイプ 文字

長さ 3

値

- YES
- NO

DSQCISQL

最後の SQL の戻りコード。

タイプ 整数

長さ 4

値 Information Center の SQL プログラミング 概念というトピックを参照してください。

DSQSQLST

SQL の状態。 IBM リレーショナル・データベース・プロダクト用の拡張 SQL 戻りコード。

タイプ 文字

長さ 5

値 Information Center の SQL プログラミング 概念というトピックの中の SQLSTATES についての付録を参照してください。

DSQAROWS

データ用に取り出された、現行の行数。

タイプ 整数

長さ 4

値 0 ～ 最大行数

DSQAROWC

現行のデータが完了。

タイプ 文字

長さ 3

値

- YES
- NO

DSQSMODE

現行の処理モード。

タイプ 文字

長さ 11

値

- BATCH
- INTERACTIVE

DSQCONFIRM

処理のデフォルトの確認。

タイプ 文字

長さ 3

値

- YES
- NO

DSQSCNVT

Query 定義から派生した情報の使用が可能。

タイプ 文字

長さ 4

値

- NO
- YES
- ONLY

DSQCIMNO

Query メッセージ ID。連絡域の Query メッセージ行に戻される値と同じ値です。

タイプ 文字

長さ 8

値 -

DSQCIQNO

メッセージ ID。連絡域の完了メッセージ行に戻される値と同じ値です。

タイプ 文字

長さ 8

値 -

DSQCIMSG

対話式にユーザーに表示されるメッセージのテキストが入ります。

タイプ 文字

長さ 55

値 -

DSQCIQMG

対話式でユーザーに表示される Query メッセージ・テキストが入っています。

タイプ 文字

拡張変数サポート

長さ 55

値 -

DSQSDBNM

すべての SQL 操作の対象となるリモート・データベースの名前。

タイプ 文字

長さ 18

値

- *CURRENT
- *NONE
- rdbname

DSQUSER

リモート・データベースと一緒に使用するユーザー ID。

タイプ 文字

長さ 10

値

- *CURRENT
- ユーザー名

DSQCMTLV

セッション・コミットメント制御レベルを指定。

タイプ 文字

長さ 4

値

- NONE
- UR
- CS
- RS (DB2™ および SQL/DS に出力する時に RS は RR に変更され、 OS/400 システムに戻る時に、RR は再び変更されて RS になります。)
- RR

Query 管理機能 CI でのコミットメント制御

コミットメント制御は、データベース・ファイル操作のグループ分けの方法です。データベースの変更が 1 つの単位にグループ化されます。コミット・コマンドまたはロールバック・コマンドにより、この単位を保管または除去することができます。コミットメント制御のレベルを指定すると、 1 つの単位として処理したいデータベース変更のグループの大きさを指定することになります。コミットメント制御は、START コマンドの DSQCMTLV キーワードを使用して指定します。次のタイプがあります。

- NONE— コミットメント制御は使用されません。
- UR— 更新された行は、トランザクションの終わりまで一緒にロックされます。
- CS— カーソルが位置づけられている行は、カーソル位置が変更されるまでロックされます。
- RS— 選択されたすべての行が、トランザクションの終わりまでロックされます。

- RR— 選択されたすべての行が、作業単位 (UOW) の終わりまでロックされます。 これにより、作業単位の中で読み取りを繰り返すと、必ず同じ結果が戻されます。

注: SAVE DATA AS コマンドのコミットメント制御レベルは、常に NONE となります。

コミットメント制御についての詳細は、*バックアップおよび回復の手引き* (SD88-5008) を参照してください。

Query 管理機能における HLL プログラムを使用した CI へのアクセス

次の高水準言語を使用して、 Query 管理機能の呼び出し可能インターフェース (CI) にアクセスすることができます。

- C/400*
- COBOL/400*
- RPG/400*

Query 管理機能 CI での C 言語インターフェース

Query 管理機能の呼び出し可能インターフェースは、通常の “C” 関数呼び出しを使用してアクセスされます。各関数呼び出しの正確な記述は、呼び出し可能インターフェース “C” 通信組み込みファイル DSQCOMMC に用意されています。通信組み込みファイル DSQCOMMC は、各オペレーティング・システムごとに固有なファイルです。 Query 管理機能には、DSQCIC と DSQCICE の 2 つの外部サブルーチン呼び出しが用意されています。 DSQCIC は、プログラム変数をアクセスする必要のない Query 管理機能コマンドを処理するために使用されます。 DSQCICE は、DSQCICE をアクセスする必要のあるコマンドを処理するために使用されます。

次のコマンドでは、DSQCICE 機能を使用しなければなりません。

START

SET GLOBAL

GET GLOBAL

他のすべての Query 管理機能コマンドは、DSQCIC 機能を使用して指定する必要があります。

Query 管理機能 CI での C 言語の DSQCOMMC の例

94 ページの図 12 は、 Query 管理機能 CI の C 通信マクロの OS/400 バージョンを示しています。

C 言語インターフェース

```
/*******************************************************************/
/*                                                                 */
/*  NAME:  dsqcommc.h                                              */
/*                                                                 */
/*  MODULE-TYPE: IBM C/400 Query Management Interface include file */
/*                                                                 */
/*  PROCESSOR:  C                                                  */
/*                                                                 */
/*  DESCRIPTION:                                                   */
/*      This include file contains the declarations needed         */
/*      by a C application program for interfacing                 */
/*      with the query management callable interface.              */
/*      query management is the OS/400 implementation of the       */
/*      Systems Application Architecture Query Callable            */
/*      Programming Interface.                                     */
/*                                                                 */
/*  Copyright: 5728-SS1 (C) COPYRIGHT IBM CORP. 1989               */
/*                                                                 */
/*******************************************************************/

/*******************************************************************/
/* Callable Interface Constants and Structures                     */
/*******************************************************************/

/* return code values for DSQ_RETURN_CODE                          */
#define DSQ_SUCCESS        0   /* successful running of the request  */
#define DSQ_WARNING        4   /* normal completion with warnings    */
#define DSQ_FAILURE        8   /* command did not process correctly  */
#define DSQ_SEVERE        16   /* severe error;  Query session       */
                               /* ended.                             */

/* Variable data types */
#define DSQ_VARIABLE_CHAR "CHAR"    /* unsigned character data type  */
#define DSQ_VARIABLE_FINT "FINT"    /* long integer type             */

/* Cancel indicator                                                */
#define DSQ_CANCEL_YES    "1"       /* Yes it was canceled.          */
#define DSQ_CANCEL_NO     "0"       /* No, it was not canceled.      */

/* Derived query/form indicator                                    */
#define DSQ_DERIVED_YES   "1"       /* Yes it was derived from QRYDFN*/
#define DSQ_DERIVED_NO    "0"       /* No, it was not derived        */

/* Yes/No indicator.  This indicator can be used to test the values  */
/*   returned for the following global variables:                  */
/*      DSQCATTN - Last command cancel indicator.                  */
/*      DSQAROWC - Current data completed indicator.               */
/*                                                                 */
#define DSQ_YES           "1"       /* Yes                           */
#define DSQ_NO            "0"       /* No                            */

/* misc defines                                                    */
#define DSQ_TRUE           1        /* indicates TRUE                */
#define DSQ_FALSE          0        /* indicates FALSE               */
#define DSQ_MATCH          0        /* match indicator               */
```

図 12. DSQCOMMC の例 (1/2)

```
/* define the Communication Area structure */
struct dsqcomm
  {
  unsigned long dsq_return_code;      /* function return code         */
  unsigned long dsq_instance_id;      /* instance id for this session */
  unsigned char dsq_reserve1[44]; /* reserved space -       */
                                      /* not for application use      */
  unsigned char dsq_message_id[8];  /* completion message id*/
  unsigned char dsq_q_message_id[8];     /* query message id*/
  unsigned char dsq_start_parm_error[8];       /* start parm*/
  unsigned char dsq_cancel_ind[1];   /* command canceled by */
                                       /* Control-Break (1=yes,0=no)*/
  unsigned char dsq_reserve2[17];       /* reserved space - */
                                        /*not for application use   */
  unsigned char dsq_query_derived[1];     /* query used was */
                                       /*derived from OS/400 *QRYDFN */
  unsigned char dsq_form_derived[1];/*form used was derived */
                                        /*   from OS/400 *QRYDFN    */
  unsigned int  dsq_delete_env;         /* flag used by QM to       */
                                        /*  control the QM          */
                                        /*  environment.            */
  unsigned char dsq_reserve3[924];    /* Reserve area 3     */
                                        /* application use          */
  } ;

/*********************************************************************/
/* Callable Interface External Function/Routine Definition           */
/*********************************************************************/

/* pragma definitions */
#define dsqcice DSQCICE
#define dsqcic DSQCIC
#pragma linkage(DSQCIC, OS)
#pragma linkage(DSQCICE, OS)

/* prototype for DSQCICE  */
extern void dsqcice  (
             struct dsqcomm *,      /* Communication Area            */
             signed long  *,        /* command length                */
             char *,                /* command                       */
             signed long *,         /* number of parms               */
             signed long *,         /* keyword lengths               */
             char *,                /* keywords                      */
             signed long *,         /* data lengths                  */
             void *,                /* data                          */
             char *);               /* data value type               */

/* prototype for DSQCIC   */
extern void dsqcic   (
             struct dsqcomm *,      /* Communication Area            */
             signed long  *,        /* command length                */
             char *);               /* command                       */
```

図 12. DSQCOMMC の例 (2/2)

Query 管理機能 CI の C 変数サポート

入力文字ストリングである C 変数 (コマンド・ストリング、および START と SET コマンド変数を含む) の場合は、ユーザーは 1 つのヌル値で終わっている区域を渡さなければなりません。変数の長さにもそのヌル値を含めなければなりません。 Query 管理機能に渡される変数の長さを入手するには、長さ機能を使用する必要があります。ヌル値 (X'00') は文字ストリングの終わりを示します。

C 言語インターフェース

出力文字ストリングである C 変数 (GET コマンドによりセットされた値を含む) の場合は、Query 管理機能は、その記憶域からユーザーの指定記憶域にデータを移動し、ストリングの終わりにヌル標識をセットします。文字ストリングがユーザーのストレージに収まらない場合には、警告メッセージが出され、データは右側を切り捨てられます。データ・ストリングの終わりには必ずヌル標識が入れられます。

Query 管理機能 CI の C 用 DSQCIC 関数構文

```
dsqcic(&communication_area,&command_length,&command_string );
```

ここで、

- *連絡域* は、構造 DSQCOMM です。
- *コマンドの長さ* は、コマンド・ストリングの長さです。

  長さは長整数として指定します。
- *コマンド・ストリング* は、処理したい Query 管理機能のコマンドです。

  コマンド・ストリングは文字タイプの配列として指定します。

Query 管理機能 CI の C 用 DSQCICE 関数構文

```
dsqcice (&communication_area,&command_length,&command_string,
         &number_of_parameters,&variable_length,&variable,
         &value_length,&value,&value_type);
```

ここで、

- *連絡域* は、構造 DSQCOMM です。
- *コマンドの長さ* は、コマンド・ストリングの長さです。

  コマンドの長さは長整数として指定します。
- *コマンド・ストリング* は、処理したい Query 管理機能のコマンドを指定する文字ストリングへのポインターです。

  コマンド・ストリングは文字タイプの配列として指定します。
- *パラメーター数* はコマンド変数の数です。

  変数の数は長整数として指定します。
- *変数長* は指定された各変数名の長さです。

  変数名 (1 つまたは複数) の長さは、長整数タイプの変数または変数配列として指定します。
- *変数* は、Query 管理機能変数名です (1 つまたは複数)。

  変数名ストリングは符号のない文字タイプの配列として指定します。
- *値の長さ* は、変数に対応する各値の長さです。

  対応する値の長さは、長整数タイプの変数または変数配列として指定します。
- *値* は各変数に対応する値です。

  値ストリングは、符号のない文字タイプの配列または長整数タイプの変数または変数配列として指定します。このタイプは VTYPE パラメーターに指定します。
- *値タイプ* は、値ストリング VALUE の Query 管理機能データ・タイプを示します。

  値タイプ・ストリングには、Query 管理機能通信マクロで提供される次の値の 1 つが入ります。

  DSQ_VARIABLE_CHAR は、値ストリングが符号のない文字タイプであることを示します。

  DSQ_VARIABLE_FINT は、値ストリングが長整数タイプであることを示します。

Query 管理機能 CI の C 用インターフェース連絡域 (DSQCOMM)

Query 管理機能インターフェース連絡域は、通信マクロ DSQCOMMC の一部です。インターフェース連絡域は、DSQCOMM という名前の構造タイプとして記述されます。

CI 連絡域 DSQCOMM には次の情報が含まれますが、呼び出しプログラムによってこれを変更しては***なりません***。

dsq_return_code (無符号長整数)
コマンドが実行された後の Query 管理機能の処理状況を示す整数

dsq_instance_ID (無符号長整数)
START コマンドの処理過程で Query 管理機能が設定する ID です。

dsq_reserve1 (44 文字)
将来使用するために確保されています。

dsq_message_id (8 文字)
完了メッセージ ID

dsq_q_message_id (8 文字)
Query メッセージ ID

dsq_start_parm_error (8 文字)
パラメーター・エラーによって START が正常に実行されなかった場合のエラーのあるパラメーター

dsq_cancel_ind (1 文字)
コマンド取り消し標識。これは、Query 管理機能がコマンドを実行していた時に、ユーザーがそのコマンドの処理を取り消したかどうかを示します。

dsq_cancel_yes (文字= 1)

dsq_cancel_no (文字= 0)

dsq_reserve2 (23 文字)

dsq_Query_derived (1 文字)
使用された Query 情報が Query for iSeries 用定義から派生したかどうかを示します。

dsq_form_derived (1 文字)
使用された書式情報が Query for iSeries 用定義から派生したかどうかを示します。

dsq_reserve2 (17 文字)

dsq_reserve3 (156 文字)

Query 管理機能 CI での C 言語インターフェース用戻りコード

戻りコードは、Query 管理機能 CI に対する各呼び出しの後で戻されます。戻りコード値はデータ・インターフェース・マクロによって記述されます。アプリケーションに移植性を持たせるためには、値は等価値によってではなく、変数名によって参照しなければなりません。この値はシステム間で異なる場合があるためです。

“dsq_return_code” の**戻りコード**の値には、次のものがあります。

C 言語インターフェース

DSQ_SUCCESS
要求は正常に処理されました。

DSQ_WARNING
正常に完了しましたが、警告が出されました。

DSQ_FAILURE
コマンドは正しく処理されませんでした。

DSQ_SEVERE
重大エラー: 該当のインスタンスについて、Query 管理機能セッションは終了しました。 Query 管理機能セッションが終了したため、このインスタンス ID を使用して、Query 管理機能に対する追加の呼び出しを行うことはできません。

Query 管理機能の C 言語 Query CI プログラム例

99 ページの図 13 は、Query 管理機能の CI 用に作成された C 言語プログラムの例です。

```
/*********************************************************************/
/* Sample Program:  DSQABFC                                          */
/* C Version of the Query Management Callable Interface              */
/*********************************************************************/

/*********************************************************************/
/* Include standard and string "C" functions                         */
/*********************************************************************/
#include <string.h>
#include <stdlib.h>

/*********************************************************************/
/* Include and declare query interface communications area           */
/*********************************************************************/
#include <DSQCOMMC.H>

int main()
     {

 struct dsqcomm communication_area;        /* DSQCOMM from include */

 /*********************************************************************/
 /* Query interface command length and commands                       */
 /*********************************************************************/
 signed long command_length;
 static char start_query_interface[] = "START";
 static char set_global_variables[] = "SET GLOBAL";
 static char run_query[] = "RUN QUERY Q1";
 static char print_report[] = "PRINT REPORT (FORM=F1";
 static char end_query_interface[] = "EXIT";

 /*********************************************************************/
 /* Query command extension, number of parameters and lengths         */
 /*********************************************************************/
 signed long number_of_parameters;    /* number of variables       */
 signed long keyword_lengths[10];     /* lengths of keyword names  */
 signed long data_lengths[10];        /* lengths of variable data  */

 /*********************************************************************/
 /* Variable data type constants                                      */
 /*********************************************************************/
 static char char_data_type[] = DSQ_VARIABLE_CHAR;
 static char int_data_type[]  = DSQ_VARIABLE_FINT;

 /*********************************************************************/
 /* Keyword parameter and value for START command                     */
 /*********************************************************************/
 static char start_keywords[] = "DSQSCMD";
 static char start_keyword_values[] = "USERCMD1";
```

図 13. C プログラムの例 (1/3)

```
/*****************************************************************/
/* Keyword parameter and values for SET command                  */
/*****************************************************************/
#define SIZE_VAL 8
char set_keywords [3][SIZE_VAL];   /* Parameter name array       */
signed long set_values[3];         /* Parameter value array      */

/*****************************************************************/
/* MAIN PROGRAM                                                  */
/*****************************************************************/

/*****************************************************************/
/* Start a Query Interface Session                               */
/*****************************************************************/
    number_of_parameters = 1;
    command_length = sizeof(start_query_interface);
    keyword_lengths[0] = sizeof(start_keywords);
    data_lengths[0] = sizeof(start_keyword_values);
    dsqcice(&communication_area,
            &command_length,
            &start_query_interface[0],
            &number_of_parameters,
            &keyword_lengths[0],
            &start_keywords[0],
            &data_lengths[0],
            &start_keyword_values[0],
            &char_data_type[0]);

/*****************************************************************/
/* Set numeric values into query using SET command               */
/*****************************************************************/
    number_of_parameters = 3;
    command_length = sizeof(set_global_variables);
    strcpy(set_keywords[0],"MYVAR01");
    strcpy(set_keywords[1],"SHORT");
    strcpy(set_keywords[2],"MYVAR03");
    keyword_lengths[0] = SIZE_VAL;
    keyword_lengths[1] = SIZE_VAL;
    keyword_lengths[2] = SIZE_VAL;
    data_lengths[0] = sizeof(long);
    data_lengths[1] = sizeof(long);
    data_lengths[2] = sizeof(long);
    set_values[0] = 20;
    set_values[1] = 40;
    set_values[2] = 84;
    dsqcice(&communication_area,
            &command_length,
            &set_global_variables[0],
            &number_of_parameters,
            &keyword_lengths[0],
            &set_keywords[0][0],
            &data_lengths[0],
            &set_values[0],
            &int_data_type[0]);
```

図 13. C プログラムの例 (2/3)

```
  /*****************************************************************/
  /* Run a Query                                                   */
  /*****************************************************************/
      command_length = sizeof(run_query);
      dsqcic(&communication_area,&command_length,&run_query [0]);

  /*****************************************************************/
  /* Print the results of the query                                */
  /*****************************************************************/
      command_length = sizeof(print_report);
      dsqcic(&communication_area,&command_length,&print_report[0]);

  /*****************************************************************/
  /* End the query interface session                               */
  /*****************************************************************/
      command_length = sizeof(end_query_interface);
      dsqcic(&communication_area,&command_length,&end_query_interface[0]);
      exit(0);
 }
```

図 13. C プログラムの例 (3/3)

Query 管理機能 CI での COBOL 言語インターフェース

Query 管理機能 CI は、通常の COBOL 関数呼び出しを使用することによってアクセスされます。各関数呼び出しの正確な記述は、Query 管理機能の COBOL 通信マクロ DSQCOMMB で提供されています。通信マクロ DSQCOMMB は、各オペレーティング・システムごとに固有なものです。 Query 管理機能では、すべての Query 管理機能コマンドを実行するために実行する DSQCIB という名前の外部サブルーチンが用意されています。 DSQCIB への呼び出しで渡されるパラメーターによって、プログラム変数が渡されるかどうかが決められます。プログラム変数は、次の Query 管理機能のコマンドで渡さなければなりません。

START

SET GLOBAL

GET GLOBAL

他の Query 管理機能コマンドは、プログラム変数を指定しません。

Query 管理機能 CI での DSQCIB 関数構文

```
CALL DSQCIB USING DSQCOMM, CMDLTH, CMDSTR.
```

ここで、

- *DSQCOMM* は構造 DSQCOMM です。
- *CMDLTH* はコマンド・ストリング CMDSTR の長さです。

  この長さは整数 "PIC 9(8)" 変数として指定されます。
- *CMDSTR* は、処理したい Query 管理機能のコマンドです。

  このコマンド・ストリングは、CMDLTH によって指定された長さの文字ストリングとして指定されます。

Query 管理機能 CI の COBOL 用 DSQCIB 拡張関数構文

```
CALL DSQCIB USING
            DSQCOMM CMDLTH CMDSTR
            PNUM VNLTH VNAME VLTH VALUE VTYPE.
```

COBOL 言語インターフェース

ここで、

- *DSQCOMM* は構造 DSQCOMM です。
- *CMDLTH* はコマンド・ストリング CMDSTR の長さです。

  この長さは整数 "PIC 9(8)" 変数として指定されます。
- *CMDSTR* は、処理したい Query 管理機能のコマンドです。

  このコマンド・ストリングは、CMDLTH によって指定された長さの文字ストリングとして指定されます。
- *PNUM* はコマンド変数の数です。

  PNUM は整数 "PIC 9(8)" 変数として指定されます。
- *VNLTH* は指定された各変数名の長さです。

  変数名 (1 つまたは複数) の長さは、整数 "PIC 9(8)" 変数または変数配列として指定されます。
- *VNAME* は、Query 管理機能変数名です (1 つまたは複数)。

  変数名ストリングは、長さが VNLTH によって指定されたものと同じ文字または文字の構造として指定されます。すべての文字が同じ長さであれば、文字の配列を使用することができます。
- *VLTH* は各変数に対応する値の長さです。

  対応する値の長さは、整数 "PIC 9(8)" 変数または変数配列として指定されます。
- *VALUE* は各変数に対応する値です。

  値ストリングは、1 つの文字、文字の構造、整数 "PIC 9(8)" 変数または変数配列として指定されます。このタイプは VTYPE パラメーターに指定します。
- *VTYPE* は、値ストリング VALUE の Query 管理機能データ・タイプを示します。

  値タイプ・ストリングには、Query 管理機能通信マクロで提供される次の値の 1 つが入ります。

  DSQ-VARIABLE-CHAR は、値が文字であることを示します。

  DSQ-VARIABLE-FINT は、値が "PIC 9(8)" であることを示します。

Query 管理機能 CI の COBOL 用インターフェース連絡域 (DSQCOMM)

Query 管理機能インターフェース連絡域は、通信マクロ DSQCOMMB の一部です。インターフェース連絡域は、DSQCOMM という名前の構造として記述されています。

インターフェース連絡域には、表 14 に示されている情報が入っています。この情報は、呼び出しプログラムによって変更してはなりません。

表 14. DSQCOM のプログラミング情報

変数	タイプ	長さ (バイト数)	説明
DSQRET	2 進数	4 バイト	コマンドが実行された後の Query 管理機能の処理状況を示す整数
DSQINS	2 進数	4 バイト	START コマンドの処理時に Query 管理機能によってセットされる ID
DSQRES	文字	44 バイト	将来使用するために確保されています。
DSQMSG	文字	8 バイト	完了メッセージ ID
DSQQMG	文字	8 バイト	Query メッセージ ID
DSQSPE	文字	8 バイト	パラメーター・エラーによって START が正常に実行されなかった場合のエラーのあるパラメーター

表 14. DSQCOM のプログラミング情報 (続き)

変数	タイプ	長さ (バイト数)	説明
DSQCNL	文字	1 バイト	コマンド取り消し標識。これは、Query 管理機能がコマンドを実行していた時に、ユーザーがそのコマンドの処理を取り消したかどうかを示します。 DSQCLY ″VALUE 1″ DSQCLN ″VALUE 0″
DSQRS2	文字	17 バイト	将来使用するために確保されています。
DSQQDR	文字	1 バイト	Query は Query for iSeries 用の QRYDFN から派生しています。 DSQDRY ″VALUE 1″ DSQDRN ″VALUE 0″
DSQFDR	文字	1 バイト	書式は Query for iSeries 用の QRYDFN から派生しています。 DSQDRY ″VALUE 1″ DSQDRN ″VALUE 0″
DSQRS3	文字	156 バイト	将来使用するために確保されています。
DSQRS4	文字	256 バイト	将来使用するために確保されています。
DSQRS5	文字	256 バイト	将来使用するために確保されています。
DSQRS6	文字	256 バイト	将来使用するために確保されています。

Query 管理機能 CI の COBOL 用戻りコード

戻りコードは、Query 管理機能 CI に対する各呼び出しの後で戻されます。戻りコード値はデータ・インターフェース・マクロによって記述されます。アプリケーションに移植性を持たせるためには、値は等価値によってではなく、変数名によって参照しなければなりません。この値はシステム間で異なる場合があるためです。

“DSQ-RETURN-CODE” の**戻りコード**値には、次のものがあります。

DSQ-SUCCESS
要求は正常に処理されました。

DSQ-WARNING
正常に完了しましたが、警告が出されました。

DSQ-FAILURE
コマンドは正しく処理されませんでした。

DSQ-SEVERE
重大エラー: 該当のインスタンスについて、Query 管理機能セッションは終了しました。 Query 管理機能セッションが終了したため、このインスタンス ID を使用して、Query 管理機能に対する追加の呼び出しを行うことはできません。

Query 管理機能の COBOL Query CI プログラム例

104 ページの図 14 は、Query 管理機能の CI 用に作成された COBOL 言語プログラムの例です。

```
*****************************************************************
*   The following is a VS COBOL II version of the query
*   callable interface *** DSQABFCO **.
*****************************************************************
 IDENTIFICATION DIVISION.
 PROGRAM-ID.  DSQABFCO.
   DATE-COMPILED.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
*************************
* Copy DSQCOMMB definition - contains query interface variables
*************************
   COPY DSQCOMMB.

* Query interface commands
 01  STARTQI      PIC X(5)  VALUE "START".
 01  SETG         PIC X(10) VALUE "SET GLOBAL".
 01  QUERY        PIC X(12) VALUE "RUN QUERY Q1".
 01  REPT         PIC X(21) VALUE "PRINT REPORT (FORM=F1".
 01  ENDQI        PIC X(4)  VALUE "EXIT".
* Query command length
 01  QICLTH       PIC 9(8) USAGE IS COMP-4.
* Number of variables
 01  QIPNUM       PIC 9(8) USAGE IS COMP-4.
* Keyword variable lengths
 01  QIKLTHS.
   03   KLTHS     PIC 9(8) OCCURS 10 USAGE IS COMP-4.
* Value Lengths
 01  QIVLTHS.
   03   VLTHS     PIC 9(8) OCCURS 10 USAGE IS COMP-4.
* Start Command Keyword
 01  SNAMES.
   03  SNAME1   PIC X(7) VALUE "DSQSCMD".
* Start Command Keyword Value
 01  SVALUES.
   03  SVALUE1   PIC X(8) VALUE "USERCMD1".
* Set GLOBAL Command Variable Names to set
 01  VNAMES.
   03  VNAME1   PIC X(7) VALUE "MYVAR01".
   03  VNAME2   PIC X(5) VALUE "SHORT".
   03  VNAME3   PIC X(7) VALUE "MYVAR03".
* Variable value parameters
 01  VVALUES.
   03   VVALS     PIC 9(8) OCCURS 10 USAGE IS COMP-4.
 01   TEMP     PIC 9(8)           USAGE IS COMP-4.
```

図 14. COBOL プログラムの例 (1/2)

```
 PROCEDURE DIVISION.
*
* Start a query interface session
     MOVE 0 TO QICLTH.
     INSPECT STARTQI TALLYING QICLTH FOR CHARACTERS.
     MOVE 0 TO TEMP.
     INSPECT SNAME1 TALLYING TEMP FOR CHARACTERS.
     MOVE TEMP TO KLTHS(1).
     MOVE 0 TO TEMP.
     INSPECT SVALUE1 TALLYING TEMP FOR CHARACTERS.
     MOVE TEMP TO VLTHS(1).
     MOVE 1 TO QIPNUM.
     CALL DSQCIB  USING DSQCOMM, QICLTH, STARTQI,
                        QIPNUM, QIKLTHS, SNAMES,
                        QIVLTHS, SVALUES, DSQ-VARIABLE-CHAR.
*
* Set numeric values into query variables using SET GLOBAL command
     MOVE 0 TO QICLTH.
     INSPECT SETG TALLYING QICLTH FOR CHARACTERS.
     MOVE 0 TO TEMP.
     INSPECT VNAME1 TALLYING TEMP FOR CHARACTERS.
     MOVE TEMP TO KLTHS(1).
     MOVE 0 TO TEMP.
     INSPECT VNAME2 TALLYING TEMP FOR CHARACTERS.
     MOVE TEMP TO KLTHS(2).
     MOVE 0 TO TEMP.
     INSPECT VNAME3 TALLYING TEMP FOR CHARACTERS.
     MOVE TEMP TO KLTHS(3).
     MOVE 4 TO VLTHS(1).
     MOVE 4 TO VLTHS(2).
     MOVE 4 TO VLTHS(3).
     MOVE 20 TO VVALS(1).
     MOVE 40 TO VVALS(2).
     MOVE 84 TO VVALS(3).
     MOVE 3 TO QIPNUM.
     CALL DSQCIB  USING DSQCOMM, QICLTH, SETG,
                        QIPNUM, QIKLTHS, VNAMES,
                        QIVLTHS, VVALUES, DSQ-VARIABLE-FINT.
*
* Run a Query
     MOVE 0 TO QICLTH.
     INSPECT QUERY TALLYING QICLTH FOR CHARACTERS.
     CALL DSQCIB USING DSQCOMM, QICLTH, QUERY.
*
* Print the results of the query
     MOVE 0 TO QICLTH.
     INSPECT REPT TALLYING QICLTH FOR CHARACTERS.
     CALL DSQCIB USING DSQCOMM, QICLTH, REPT.
*
* End the query interface session
     MOVE 0 TO QICLTH.
     INSPECT ENDQI TALLYING QICLTH FOR CHARACTERS.
     CALL DSQCIB USING DSQCOMM, QICLTH, ENDQI.
     STOP RUN.
```

図 14. COBOL プログラムの例 (2/2)

Query 管理機能の COBOL Query CI プログラム例 2

106 ページの図 15 は、 OS/400 環境に合うように調整された Query 管理機能 CI の COBOL 通信マクロの例です。

```
*****************************************************************
*                                                               *
*  NAME:  DSQCOMMB                                              *
*                                                               *
*  MODULE-TYPE: IBM COBOL/400 Query Management Interface        *
*               include file                                    *
*                                                               *
*  PROCESSOR: COBOL                                             *
*                                                               *
*  DESCRIPTION:                                                 *
*      This include file contains the declarations needed       *
*      by a COBOL/400 application program for interfacing       *
*      with the query management callable interface.            *
*      query management is the OS/400 implementation of the     *
*      Systems Application Architecture Query Callable          *
*      Programming Interface.                                   *
*                                                               *
*  Copyright: 5728-SS1 (C) COPYRIGHT IBM CORP. 1989             *
*                                                               *
*****************************************************************

* Structure declare for communications area
  01 DSQCOMM.
     03 DSQ-RETURN-CODE      PIC 9(8) USAGE IS BINARY VALUE 0.
*                             * Function return code       *
     03 DSQ-INSTANCE-ID      PIC 9(8) USAGE IS BINARY VALUE 0.
*                             * Identifier from START cmd  *
     03 DSQ-RESERVE1         PIC X(44).
*                             * Reserved area              *
     03 DSQ-MESSAGE-ID       PIC X(8).
*                             * Completion message id      *
     03 DSQ-Q-MESSAGE-ID     PIC X(8).
*                             * Query message ID           *
     03 DSQ-START-PARM-ERROR PIC X(8).
*                             * START parameter in error   *
     03 DSQ-CANCEL-IND       PIC X(1).
*                             * 1 = Command canceled       *
*                             * 0 = Command not canceled   *
     03 DSQ-RESERVE2         PIC X(17).
*                             * Reserved space -- not for  *
*                             * application use            *
```

図 15. DSQCOMMB の例 (1/2)

```
     03 DSQ-QUERY-DERIVED    PIC X(1).
*                              * 1 = Query was derived from *
*                              *      OS/400 QRYDFN         *
*                              * 0 = Query was not derived  *
*                              *      from OS/400 QRYDFN    *
     03 DSQ-FORM-DERIVED     PIC X(1).
*                              * 1 = Form was derived from  *
*                              *      OS/400 QRYDFN         *
*                              * 0 = Form was not derived   *
*                              *      from OS/400 QRYDFN    *
     03 DSQ-DELETE-ENV       PIC 9(8) USAGE IS BINARY VALUE 0.
*                              * Flag used to delete env.   *
     03 DSQ-RESERVE3         PIC X(924).
*                              * Reserved space -- not for  *
*                              * application use            *


*  Return code values for DSQ-RETURN-CODE
  01 DSQ-SUCCESS             PIC 9(8) USAGE IS BINARY VALUE 0.
  01 DSQ-WARNING             PIC 9(8) USAGE IS BINARY VALUE 4.
  01 DSQ-FAILURE             PIC 9(8) USAGE IS BINARY VALUE 8.
  01 DSQ-SEVERE              PIC 9(8) USAGE IS BINARY VALUE 16.

*  Callable Interface program name
  01 DSQCIB                  PIC X(7) VALUE "QQXMAIN".

*  Values for variable type on CALL parameter
  01 DSQ-VARIABLE-CHAR       PIC X(4) VALUE "CHAR".
  01 DSQ-VARIABLE-FINT       PIC X(4) VALUE "FINT".

*  Values for query/form derived field in communications area
  01 DSQ-DERIVED-NO          PIC X(1) VALUE "0".
  01 DSQ-DERIVED-YES         PIC X(1) VALUE "1".

*  Values for the cancel indicator field in communications area
  01 DSQ-CANCEL-YES         PIC X(1) VALUE "1".
  01 DSQ-CANCEL-NO          PIC X(1) VALUE "0".

*  Yes/No indicator.  This indicator can be used
*   to test the values
*   returned for the following global variables:
*     DSQCATTN - Last command cancel indicator.
*     DSQAROWC - Current data completed indicator.
*
  01 DSQ-YES                 PIC X(1) VALUE "1".
  01 DSQ-NO                  PIC X(1) VALUE "0".
```

図 15. DSQCOMMB の例 (2/2)

Query 管理機能 CI での RPG 言語インターフェース

Query 管理機能の呼び出し可能インターフェースは、通常の RPG 関数呼び出しを使用してアクセスされます。各関数呼び出しの正確な記述は、Query 管理機能の RPG 通信マクロ組み込みメンバー DSQCOM で提供されています。 Query 管理機能では、すべての Query 管理機能コマンドを実行するために実行する、DSQCIR という名前の外部サブルーチンが用意されています。 DSQCIR への呼び出しで渡されるパラメーターにより、プログラム変数が渡されるかどうかが決まります。プログラム変数は、次の Query 管理機能のコマンドで渡さなければなりません。

START

SET GLOBAL

GET GLOBAL

RPG 言語インターフェース

他の Query 管理機能コマンドは、プログラム変数を指定しません。

Query 管理機能 CI の RPG 用 DSQCIR 関数構文

```
C                   CALL DSQCIR
C                   PARM           DSQCOM
C                   PARM           CMDLTH
C                   PARM           CMDSTR
```

ここで、

- *DSQCOM* は構造 DSQCOM です。
- *CMDLTH* はコマンド・ストリング CMDSTR の長さです。
  この長さは、4 バイトの 2 進数フィールドとして指定されます。
- *CMDSTR* は、処理したい Query 管理機能のコマンドです。
  このコマンド・ストリングは、CMDLTH によって指定された長さの文字ストリングとして指定されます。

Query 管理機能 CI の RPG 用 DSQCIR 拡張関数構文

```
C                   CALL DSQCIR
C                   PARM           DSQCOM
C                   PARM           CMDLTH
C                   PARM           CMDSTR
C                   PARM 1         PNUM
C                   PARM           KLTH
C                   PARM           KWORD
C                   PARM           VLTH
C                   PARM           VALUE
C                   PARM           VTYPE
```

ここで、

- *DSQCOM* は構造 DSQCOM です。
- *CMDLTH* はコマンド・ストリング CMDSTR の長さです。
  この長さは、4 バイトの 2 進数フィールドとして指定されます。
- *CMDSTR* は、処理したい Query 管理機能のコマンドです。
- このコマンド・ストリングは、CMDLTH によって指定された長さの文字ストリングとして指定されます。
- *PNUM* はコマンド・キーワードの数です。
  PNUM は 4 バイトの 2 進数フィールドとして指定されます。
- *KLTH* は指定される各キーワードの長さです。
  このキーワードの長さは、4 バイトの 2 進数フィールドとして指定されます。
- *KWORD* は、Query 管理機能キーワード (複数のこともある) です。
  キーワード・ストリングは、KLTH で指定されたものと同じ文字または文字の構造として指定されます。すべての文字が同じ長さであれば、文字の配列を使用することができます。
- *VLTH* はキーワードと対応する各値の長さです。
  対応する値の長さは、4 バイトの 2 進数フィールドとして指定されます。
- *VALUE* は各キーワードに対応する値です。
  値のストリングは、1 つの文字、文字の構造、または 4 バイトの 2 進数フィールドとして指定されます。このタイプは VTYPE パラメーターに指定します。

- *VTYPE* は、値ストリング VALUE の Query 管理機能データ・タイプを示します。

  値タイプ・ストリングには、Query 管理機能通信組み込みメンバーで提供される次の値の 1 つが入ります。

  DSQVCH は、値が文字であることを示します。

  DSQVIN は、値が整数 (4 バイトの 2 進数) であることを示します。

Query 管理機能 CI の RPG 用インターフェース連絡域 (DSQCOMMR)

Query 管理機能インターフェース連絡域は、通信組み込みメンバー DSQCOMMR の一部です。このインターフェース連絡域は、DSQCOM という名前の構造として記述されます。

CI 連絡域 DSQCOM には次の情報が入っていますが、これを呼び出しプログラムで変更しては**なりません**。

DSQRET 2 進数 (4 バイト)
コマンドが実行された後の Query 管理機能の処理状況を示す整数

DSQINS 2 進数 (4 バイト)
START コマンドの処理過程で Query 管理機能が設定する ID です。

DSQRS1 文字 (44 バイト)
将来使用するために確保されています。

DSQMSG 文字 (8 バイト)
完了メッセージ ID

DSQQMG 文字 (8 バイト)
Query メッセージ ID

DSQSPE 文字 (8 バイト)
パラメーター・エラーによって START が正常に実行されなかった場合のエラーのあるパラメーター

DSQCNL 文字 (1 バイト)
コマンド取り消し標識。これは、Query 管理機能がコマンドを実行していた時に、ユーザーがそのコマンドの処理を取り消したかどうかを示します。

DSQCLY "VALUE 1"

DSQCLN "VALUE 0"

DSQQDR 文字 (1 バイト)
使用された Query 情報が Query for iSeries 用定義から派生したかどうかを示します。

DSQDRY "VALUE 1" — オブジェクトが派生した

DSQDRN "VALUE 0" — オブジェクトは派生しなかった

DSQFDR 文字 (1 バイト)
使用された書式情報が Query for iSeries 用定義から派生したかどうかを示します。

DSQDRY "VALUE 1" — オブジェクトが派生した

DSQDRN "VALUE 0" — オブジェクトは派生しなかった

DSQRS2 文字 (23 バイト)
将来使用するために確保されています。

DSQRS3 文字 (156 バイト)
将来使用するために確保されています。

Query 管理機能 CI の RPG 用戻りコード

戻りコードは、Query 管理機能 CI に対する各呼び出しの後で戻されます。戻りコードの値は、データ・インターフェースによって記述されます。アプリケーションに移植性を持たせるためには、値は等価値によってではなく、変数名によって参照しなければなりません。この値はシステム間で異なる場合があるためです。 DSQRET の**戻りコード**の値は次のとおりです。

DSQSUC
要求は正常に処理されました。

DSQWAR
正常に完了しましたが、警告が出されました。

DSQFAI
コマンドは正しく処理されませんでした。

DSQSEV
重大エラー: 該当のインスタンスについて、Query 管理機能セッションは終了しました。 Query 管理機能セッションが終了したため、このインスタンス ID を使用して、Query 管理機能に対する追加の呼び出しを行うことはできません。

Query 管理機能 CI の RPG 言語 Query の例

図 16 は、Query 管理機能 CI 用に作成された RPG 言語プログラムの例です。

```
     ****************************************************************
     *                                                              *
     *           SAMPLE RPG PROGRAM USING QUERY INTERFACE           *
     *           ----------------------------------------           *
     *                                                              *
     *  1) Include member DSQCOMMMR contains the communications     *
     *     area to be passed to the query interface.                *
     *  2) Command name lengths, command names,                     *
     *     variable name lengths, variable names,                   *
     *     variable value lengths, variable values,                 *
     *     are loaded as compile time arrays.                       *
     *  3) It is necessary to pass all interface lengths and        *
     *     numeric variable information in binary format.           *
     *                                                              *
     ****************************************************************
    H
     *
     * Compile time arrays of command   name   lengths  and  values
     *                         variable  name   lengths  and  values
     *                         variable content lengths  and  values
     *
    E                    CNL     1   7  9 0 CNV    25     commands
    E                    VNL     1   3  9 0 VNV     7     variable names
    E                    VCL     1   3  9 0 VCV     9 0   variable values
     *
    I            DS
    I                                    B   1  280CNL
    I            DS
    I                                    B   1  120VNL
    I            DS
    I                                    B   1  120VCL
    I            DS
    I                                    B   1   40BINARY
```

図 16. RPG プログラムの例 (1/3)

```
 *
 * Pull in the communications area
 *
I/COPY DSQCOMMR
 *
 * Start a query interface session:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM           CNL,1            command length
C                     PARM           CNV,1            START
C                     PARM 1         BINARY           # keywords
C                     PARM           CNL,6            keyword length
C                     PARM           CNV,6            DSQSCMD
C                     PARM           CNL,7            value length
C                     PARM           CNV,7            USERCMD1
C                     PARM DSQVCH    DATA    4        CHAR
 *
 * Set numeric values into query variables using SET GLOBAL command:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM           CNL,2            command length
C                     PARM           CNV,2            SET GLOBAL
C                     PARM 3         BINARY           # variables
C                     PARM           VNL              name lengths
C                     PARM           VNV              name values
C                     PARM           VCL              variable lengths
C                     PARM           VCV              variable values
C                     PARM DSQVIN    DATA             FINT
 *
 * Run a query:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM           CNL,3            command length
C                     PARM           CNV,3            RUN QUERY Q1
 *
 * Print the results of the query:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM           CNL,4            command length
C                     PARM           CNV,4            PRINT REPORT
 *                                                      (FORM=F1
 * End the query interface session:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM           CNL,5            command length
C                     PARM           CNV,5            EXIT
 *
C                     SETON                      LR
 *
```

図 16. RPG プログラムの例 (2/3)

```
**  CNL/CNV           Command lengths and command values
000000005START
000000010SET GLOBAL
000000012RUN QUERY Q1
000000021PRINT REPORT (FORM=F1
000000004EXIT
000000007DSQSCMD
000000008USERCMD1
**  VNL/VNV           Variable name lengths and variable names
000000007MYVAR01
000000007MYVAR02
000000007MYVAR03
**  VCL/VCV           Variable value lengths and variable values
000000004000000020
000000004000000040
000000004000000084
```

図 16. RPG プログラムの例 (3/3)

Query 管理機能 CI の RPG 言語 Query の例 2

図 17 は、Query 管理機能 CI の RPG 通信組み込みメンバーの例です。このバージョンの通信組み込みメンバーは、 OS/400 システムの環境に合わせて調整されています。

```
I*****************************************************************
I*                                                               *
I*  NAME:  DSQCOMMR                                              *
I*                                                               *
I*  MODULE-TYPE: IBM RPG/400 Query Management Include File       *
I*                                                               *
I*  PROCESSOR: RPG                                               *
I*                                                               *
I*  DESCRIPTION:                                                 *
I*      This include file contains the declarations needed       *
I*      by an RPG/400 application program for interfacing        *
I*      with the query management callable interface.            *
I*      Query Management is the OS/400 implementation of the     *
I*      Systems Application Architecture Query Callable          *
I*      Programming Interface.                                   *
I*                                                               *
I*  Copyright: 5728-SS1 (C) COPYRIGHT IBM CORP. 1989             *
I*                                                               *
I*****************************************************************
```

図 17. DSQCOMMR の例 (1/3)

```
I*****************************************************************
I*               QUERY INTERFACE INCLUDE                         *
I*                                                               *
I* DSQCOM Definition, contains QUERY interface variables:        *
I*                                                               *
I*           DSQRET     - Status of QUERY processing            *
I*           DSQINS     - QUERY identifier                      *
I*           DSQRS1     - Reserved                              *
I*           DSQMSG     - Completion message-ID                 *
I*           DSQQMG     - QUERY message ID                      *
I*           DSQSPE     - START fail parameter error            *
I*           DSQCNL     - Command cancel indicator              *
I*           DSQQDR     - Query was derived from OS/400 QRYDFN  *
I*           DSQFDR     - Form was derived from OS/400 QRYDFN   *
I*           DSQDEN     - Environment deletion indicator        *
I*           DSQRS2     - Reserved                              *
I*           DSQRS3     - Reserved                              *
I*           DSQRS4     - Reserved                              *
I*           DSQRS5     - Reserved                              *
I*           DSQRS6     - Reserved                              *
I*                                                               *
I*****************************************************************
IDSQCOM      DS     I                                     B   1   40DSQRET
I                                    B   5   80DSQINS
I                                        9  52 DSQRS1
I                                       53  60 DSQMSG
I                                       61  68 DSQQMG
I                                       69  76 DSQSPE
I                                       77  77 DSQCNL
I                                       78  94 DSQRS2
I                                       95  95 DSQQDR
I                                       96  96 DSQFDR
I                                       97 100 DSQDEN
I                                      101 356 DSQRS3
I                                      357 612 DSQRS4
I                                      613 868 DSQRS5
I                                      8691024 DSQRS6
I*
```

図 17. DSQCOMMR の例 (2/3)

RPG 言語インターフェース

```
I* DSQRET - DSQ return code meanings
I*                     SUCCESS       --     value  0
I*                     WARNING       --     value  4
I*                     FAILURE       --     value  8
I*                     SEVERE        --     value 16

I*
I              0                      C         DSQSUC
I              4                      C         DSQWAR
I              8                      C         DSQFAI
I              16                     C         DSQSEV
I*
I* DSQCNL - DSQ cancel indicator meanings
I*                     CANCEL YES    --     value '1'
I*                     CANCEL NO     --     value '0'
I*
I              '1'                    C         DSQCLY
I              '0'                    C         DSQCLN
I*
I* DSQQDR/DSQFDR - DSQ QRYDFN derivation indicator meanings
I*                     DERIVED YES   --     value '1'
I*                     DERIVED NO    --     value '0'
I*
I              '1'                    C         DSQDRY
I              '0'                    C         DSQDRN
I*
I* DSQYES/DSQNO  - DSQ constants for the values returned
I*                   for the following global variables:
I*                     DSQCATTN - Last command cancel indicator.
I*                     DSQAROWC - Current data completed indicator.
I*
I              '1'                    C         DSQYES
I              '0'                    C         DSQNO
I*     I* DSQCIR - Interface program call name definition
I*
I              'QQXMAIN'              C         DSQCIR
I*
I* DSQVCH - contains constant value 'CHAR'
I* DSQVIN - contains constant value 'FINT'
I*
I              'CHAR'                 C         DSQVCH
I              'FINT'                 C         DSQVIN
I*
I*           END OF DSQCOM QUERY INCLUDE
I************************************************************
```

図 17. DSQCOMMR の例 (3/3)

サブプログラムを使用した Query 管理機能の CI へのアクセス

サブプログラムを使用して、 Query 管理機能呼び出し可能インターフェース (CI) にアクセスすることができます。これらのサブプログラムによって、 CI にアクセスするために必要なデータの大部分をユーザーが取り扱わなくてすむようになります。このセクションでは、サブプログラムおよび Query 処理でサブプログラムを使用する方法について説明します。

ここでは、より一般的な実行される機能を表す 7 つのサブプログラムを列挙して説明します。ユーザーの特定の環境で、一般に使用される他の機能がある場合には、これとは別のサブプログラムを作成してください。

サブプログラムを使用する場合には、次の点を考慮してください。

- 呼び出しプログラムでサブプログラムを 1 回だけ呼び出す場合、または呼び出しの頻度が少ない場合、呼び出しプログラムに戻った時にそのサブプログラムを終了してください。他のプログラムの呼び出しの詳細については、*RPG/400 使用者の手引き* を参照してください。
- CI が開始された後は、インスタンス ID が割り振られます。したがって、データ構造 DSQCOM がプログラムからプログラムへ渡されます。このインスタンス ID は、そのセッション中のアクセスのたびに CI に渡すことが必要です。
- これらのサブプログラムを使用するアプリケーション・プログラムが、サブプログラムが作成された iSeries システムとは異なるシステムで実行される場合には、プログラムを実行する iSeries システムに、これらのサブプログラムのオブジェクト・コード・バージョンが必要になります。
- これらのサブプログラムを使用するアプリケーション・プログラムを iSeries システム以外のシステムで実行する場合には、同じ機能を実行するプログラムのオブジェクト・コード・バージョンをそのシステムに作成しなければなりません。

  注: この章で説明されるコードは RPG/400® 言語で書かれており、 iSeries 以外のシステムで必ずしもコンパイルできるとは限りません。
- この章に示されているコードは RPG で書かれていますが、同様の機能を他のプログラミング言語で開発することができます。また、RPG/400 サブプログラムがコンパイル済みであれば、 COBOL プログラムからそれにアクセスすることもできます。

次のセクションでは、一般に使用される Query 管理機能の各機能を実行するためのサブプログラムの使用法を説明します。それぞれの説明は、Query 管理機能タスクを実行するために作成されるサブプログラムの例についてのものです。

Query 管理機能 CI での START サブプログラム

キーワード DSQSMODE、DSQSCMD、DSQSRUN および DSQSNAME の値が、このプログラムに 132 文字のストリングとして渡されます。最初の 33 文字は DSQSMODE キーワード値、次の 33 文字は DSQSCMD キーワード値、その次の 33 文字は DSQSRUN キーワード値、そして最後の 33 文字は DSQSNAME キーワード値を表します。タイプするキーワード値は左寄せしてください。キーワード値を使用しない場合であっても、 33 桁のブランク文字のストリングとして渡すことが必要です。

START サブプログラムは、渡されたキーワード値ストリングを読み取り、ブランク値がないかどうかをテストして、値の長さを計算します。さらに、開始コマンド、キーワード、およびキーワード値を必要な長さの 1 つのストリングにまとめて、プログラム式インターフェースを呼び出します。インターフェースが開始されると、START サブプログラムは、制御権を呼び出しプログラムに戻すことによって終了します。

```
 ****************************************************************
 *                                                              *
 *         START COMMAND CPI QUERY INTERFACE HANDLER            *
 *         ----------------------------------------             *
 *                                                              *
 *  1) Include member DSQCOMMR contains the communications      *
 *     area to be passed to the query management interface.     *
 *  2) This program handles the START CPI QM interface          *
 *     command. It reads the DSQ keywords information to be     *
 *  3) The keyword information is passed to this program in the *
 *     form of 4 values which are the 4 keyword values to be    *
 *     passed to query management. This program calculates      *
 *     the length of each keyword name and keyword value and    *
 *     strings the necessary information into arrays for        *
 *     passing to the query management callable interface.      *
 *                                                              *
 ****************************************************************
H
 *
E                    LTH     1   4  9 0 KEY     8   lengths of k/wds
E                    STA         4 33               k/wd vals passed
E                    KEL         4  9 0             keyword lengths
E                    KEN        30  1               keyword names
E                    VAL         4  9 0             value lengths
E                    VAV        81  1               value values
E                    TST        33  1               test value length
 *
I            DS
I                                    B   1   40BIN1
I                                    B   5   80BIN2
I                                    B   9  240KEL
I                                    B  25  400VAL
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed start command keyword values:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
C                     PARM           STA              keywords passed
 *
 * prepare keyword name lengths, names, value lengths, values
 *
C                     Z-ADD1         Y       20       initialize
C                     Z-ADD1         W       20       counters
C                     Z-ADD0         KEL              and numeric
C                     Z-ADD0         VAL              arrays
```

図 18. START サブプログラムの例 (1/2)

```
     *
     C           V         DOUEQ4                          look at each
     C                     ADD  1         V       10       passed keyword
     C           STA,V     COMP *BLANKS                   50value & process
     C           *IN50     IFEQ '0'                        if not blank
     *
     C                     ADD  1         X       10       keyword name
     C                     MOVE LTH,V     KEL,X            lengths array
     *
     C           ' '       LOKUPKEN,Y                     60 string keyword
     C                     MOVE KEY,V     WORK1   8          name into
     C                     MOVEAWORK1     KEN,Y              names array
     *
     C                     MOVEASTA,V     TST                find keyword
     C                     Z-ADD1         Z       20         value length
     C           ' '       LOKUPTST,Z                     61 and move
     C   61                SUB  1         Z                  to keyword
     C  N61                Z-ADD33        Z                  value lengths
     C                     Z-ADDZ         VAL,X              array
     *
     C           ' '       LOKUPVAV,W                     62 string keyword
     C                     MOVE STA,V     WORK2  33          value into
     C                     MOVEAWORK2     VAV,W              values array
     *
     C                     END
     C                     END
     *
     * start the query interface session:
     *
     C                     CALL DSQCIR
     C                     PARM           DSQCOM           comms area
     C                     PARM 5         BIN1             command length
     C                     PARM 'START'   CHAR1   5        START
     C                     PARM X         BIN2             # keywords
     C                     PARM           KEL              keyword lengths
     C                     PARM           KEN              keyword names
     C                     PARM           VAL              value lengths
     C                     PARM           VAV              values
     C                     PARM DSQVCH    CHAR2   4        CHAR
     *
     C                     MOVE '1'       *INLR
     *
**  start DSQ keyword name lengths and names loaded as compile time array
000000008DSQSMODE
000000007DSQSCMD
000000007DSQSRUN
000000008DSQSNAME
```

図 18. START サブプログラムの例 (2/2)

Query 管理機能 CI での SETC サブプログラム

SETC サブプログラムは、CI に渡される文字値について SET GLOBAL 変数の機能を実行します。 SETC サブプログラムは、一度に 1 つの変数を処理します。変数名および値は、このプログラムに 2 つの別々のパラメーターとして渡されます。この名前は 10 文字までの長さ、値は 20 文字までの長さとすることができます。

このサブプログラムは必要な長さを計算し、情報を 1 つのストリングにまとめて、プログラム式インターフェースを呼び出します。変数は CHAR データ・タイプにセットされ、次に制御権が呼び出しプログラムに戻されます。

注: SETC サブプログラムは、セッション中に何度も呼び出されることがあるため、実行後も終了しません。

```
 ****************************************************************
 *                                                              *
 *          SET GLOBAL COMMAND (CHARACTER VARIABLE)             *
 *                 CPI QUERY INTERFACE HANDLER                  *
 *          ------------------------------------                *
 *                                                              *
 *  1) Include member DSQCOMMR contains the communications      *
 *     area to be passed to the Query Interface.                *
 *  2) This program handles the SET GLOBAL Query Interface      *
 *     command for variable values to be passed to the          *
 *     interface as CHAR type.
 *  3) It reads the variable name and value, calculates the     *
 *     length of each, and passes the information to Query      *
 *     Management.                                              *
 *  4) The program handles one variable at a time, the length   *
 *     of the variable name can be a maximum of 10 characters   *
 *     and the length of the variable value can be a maximum    *
 *     of 20 characters.                                        *
 *                                                              *
 ****************************************************************
H
 *
E                    TNL        10  1              test name length
E                    TVL        20  1              test value length
 *
I              'SET GLOBAL'          C         CMD
 *
I            DS
I                                    B   1   40BIN1
I                                    B   5   80BIN2
I                                    B   9  120BIN3
I                                    B  13  160BIN4
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed variable name and value:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
C                     PARM           VARNAM 10        variable name
C                     PARM           VARVAL 20        variable value
 *
 * calculate the variable name length and variable value length:
 *
C                     MOVEAVARNAM    TNL
C                     Z-ADD1         X       20         X = last
C           ' '       LOKUPTNL,X                    60  non blank
C   60                SUB  1         X                  character
C  N60                Z-ADD10        X                  in name
```

図 19. SETC サブプログラムの例 (1/2)

```
 *
C                     Z-ADD20        Y       20         if value
C           VARVAL    IFNE *BLANKS                      blank pass
C                     MOVEAVARVAL    TVL                20 blanks
C           AGAIN     TAG
C           ' '       COMP TVL,Y                   61   Y = last
C   61                SUB  1         Y                  non blank
C   61                GOTO AGAIN                        character
C                     END                               in value
 *
 * set the global variables:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM 10        BIN1             command length
C                     PARM CMD       CHAR1  10        SET GLOBAL
C                     PARM 1         BIN2             # variables
C                     PARM X         BIN3             var name length
C                     PARM           VARNAM           variable name
C                     PARM Y         BIN4             var value lngth
C                     PARM           VARVAL           variable value
C                     PARM DSQVCH    CHAR2   4        CHAR
 *
C                     RETRN
```

図 19. SETC サブプログラムの例 (2/2)

Query 管理機能 CI での SETA サブプログラム

SETA サブプログラムは、アポストロフィで囲まれる文字値について SET GLOBAL 変数の機能を実行して、それを CI に渡します。この機能は、データ項目を固定情報文字値 (DEPT='ACCT') と比較する Query を作成する時に必要です。変数名および値は、このプログラムに 2 つの別々のパラメーターとして渡されます。この名前は 10 文字までの長さ、値は 20 文字までの長さとすることができます。

このサブプログラムは値をアポストロフィで囲み、必要な長さを計算し、情報を 1 つのストリングにまとめて、プログラム式インターフェースを呼び出します。変数は CHAR データ・タイプにセットされ、次に制御権が呼び出しプログラムに戻されます。

注: SETA サブプログラムは、セッション中に何度も呼び出されることがあるため、実行後も終了しません。

```
*****************************************************************
*                                                               *
*       SET GLOBAL COMMAND (APOSTROPHE ENCLOSED CHARACTER       *
*             VARIABLE) CPI QUERY INTERFACE HANDLER             *
*       ------------------------------------------------        *
*                                                               *
*  1) Include member DSQCOMMR contains the communications       *
*     area to be passed to the query management interface.      *
*  2) This program handles the SET GLOBAL interface             *
*     command for variable values to be enclosed in             *
*     apostrophes and passed to the interface as CHAR type.     *
*  3) It reads the variable name and value, calculates the      *
*     length of each, encloses the value in apostrophes,        *
*     and passes the information to query management.           *
*  4) The program handles one variable at a time, the length    *
*     of the variable name can be a maximum of 10 characters    *
*     and the length of the variable value can be a maximum     *
*     of 20 characters.                                         *
*                                                               *
*****************************************************************
H
 *
E                    TNL        10  1               test name length
E                    TVL        22  1               test value length
 *
I              'SET GLOBAL'          C         CMD
 *
I            DS
I                                    B   1   40BIN1
I                                    B   5   80BIN2
I                                    B   9  120BIN3
I                                    B  13  160BIN4
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed variable name and value:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
C                     PARM           VARNAM 10        variable name
C                     PARM           VARVAL 20        variable value
 *
 * calculate the variable name length and variable value length:
 *
C                     MOVEAVARNAM    TNL
C                     Z-ADD1         X       20         X = last
C           ' '       LOKUPTNL,X                    60  non blank
C   60                SUB  1         X                  character
C  N60                Z-ADD10        X                  in name
```

図 20. SETA サブプログラムの例 (1/2)

```
 *
C                     MOVE ''      TVL,1              set up first
C                     MOVEAVARVAL    TVL,2              apostrophe
C                     Z-ADD21        Y       20
C           AGAIN     TAG                               Y = last
C           ' '       COMP TVL,Y                    61  blank
C   61                SUB  1         Y                  character
C   61                GOTO AGAIN
C                     ADD  1         Y                  set up last
C                     MOVE ''      TVL,Y              apostrophe
 *
 * set the global variables:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM             comms area
C                     PARM 10        BIN1               command length
C                     PARM CMD       CHAR1  10          SET GLOBAL
C                     PARM 1         BIN2               # variables
C                     PARM X         BIN3               var name length
C                     PARM           VARNAM             variable name
C                     PARM Y         BIN4               var value lngth
C                     PARM           TVL                variable value
C                     PARM DSQVCH    CHAR2   4          CHAR
 *
C                     RETRN
```

図 20. SETA サブプログラムの例 (2/2)

Query 管理機能 CI での SETN サブプログラム

SETN サブプログラムは小数点を持ち、後書き符号が挿入される数値 (2 進数以外) について SET GLOBAL 変数の機能を実行して、それを CI に渡します。この機能は、数値データ項目を固定情報値 (AMOUNT=525.30-) と比較する Query を作成する時に必要です。

変数名、変数値、および小数点以下の桁数は、このプログラムに 3 つの別々のパラメーターとして渡されます。この名前は 10 文字までの長さの文字、値は 15 文字の長さの数値、および小数点以下の桁数は 2 桁の長さとすることができます。値と小数点以下の桁数は、標準の数値データとして渡されなければなりません (渡される前に左寄せされない)。このサブプログラムは、指定された場合に小数点を挿入し、数値が負であれば負符号を追加し、必要な長さを計算し、情報を 1 つのストリングにまとめて、プログラム式インターフェースを呼び出します。変数は CHAR データ・タイプにセットされ、制御権が呼び出しプログラムに戻されます。

注: SETN サブプログラムは、セッション中に何度も呼び出されることがあるため、実行後も終了しません。

```
*****************************************************************
*                                                               *
*      SET GLOBAL COMMAND (NUMERIC - NON BINARY INTEGER)        *
*                 CPI QUERY INTERFACE HANDLER                   *
*      ------------------------------------------------         *
*                                                               *
*  1) Include member DSQCOMMR contains the communications       *
*     area to be passed to the query management interface.      *
*  2) This program handles the SET GLOBAL interface             *
*     command for variable values to be passed to the           *
*     interface as numeric data CHAR type.                      *
*  3) It reads the variable name and value, calculates the      *
*     length of each, inserts the decimal point and leading     *
*     negative sign (if required) and passes the information    *
*     to query management.                                      *
*  4) The program handles one variable at a time, the length    *
*     of the variable name can be a maximum of 10 characters    *
*     and the length of the variable value can be a maximum     *
*     of 15 numeric digits (plus sign and decimal point).       *
*                                                               *
*****************************************************************
H
 *
E                    TNL        10  1               test name length
E                    TVL        17  1               variable value
 *
I              'SET GLOBAL'          C         CMD
 *
I            DS
I                                    B   1   40BIN1
I                                    B   5   80BIN2
I                                    B   9  120BIN3
I                                    B  13  160BIN4
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed variable name and value:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
C                     PARM           VARNAM 10        variable name
C                     PARM           VARVAL 150       variable value
C                     PARM           VARDEC  20       decimal places
 *
 * calculate the variable name length:
 *
C                     MOVEAVARNAM    TNL
C                     Z-ADD1         X       20         X = last
C           ' '       LOKUPTNL,X                    60  non blank
C   60                SUB  1         X                  character
C  N60                Z-ADD10        X                  in name
```

図 21. SETN サブプログラムの例 (1/2)

```
 *
 * set up the variable with decimal point and leading minus sign:
 *
C                     MOVE *BLANKS   TVL              Clear array
C                     MOVE VARVAL    VARCHA 15        Setup as alpha
C           VARVAL    COMP 0                     61  Negative value
C   61                MLLZO'8'       VARCHA           so strip sign
C*
C           VARDEC    IFEQ 0                          * Processing
C                     MOVEAVARCHA    TVL,3            * if value
C   61                MOVE '-'       TVL,2            * has no
C                     GOTO PASS                       * decimals
C                     END                             *
C*
C                     MOVEAVARCHA    TVL,2            * Processing
C   61                MOVE '-'       TVL,1            * if value
C                     Z-ADD16        Y       20       * has
C                     Z-ADD17        Z       20       * decimals
C           AGAIN     TAG                             *
C                     MOVE TVL,Y     TVL,Z            * Move each
C                     SUB  1         VARDEC           * array
C           VARDEC    IFNE 0                          * position
C                     SUB  1         Y                * over one
C                     SUB  1         Z                * place until
C                     GOTO AGAIN                      * decimal
C                     END                             * location
C                     MOVE '.'       TVL,Y            * is reached
C*
C           PASS      TAG
 *
 * set the Global Variables:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM 10        BIN1             command length
C                     PARM CMD       CHAR1  10        SET GLOBAL
C                     PARM 1         BIN2             # variables
C                     PARM X         BIN3             var name length
C                     PARM           VARNAM           variable name
C                     PARM 17        BIN4             var value lngth
C                     PARM           TVL              variable value
C                     PARM DSQVCH    CHAR2   4        CHAR
 *
C                     RETRN
```

図 21. SETN サブプログラムの例 (2/2)

Query 管理機能 CI での RUNQ サブプログラム

RUNQ サブプログラムは、RUN QUERY インターフェースを活動化します。 Query 名と書式名は、このプログラムに 42 文字のストリングとして渡されます。最初の 21 文字が Query 名となり、後半の 21 文字が書式名となります。

Query 名と書式名は左寄せしなければなりません。書式が使用されていない場合であっても、ストリングの 22～42 桁目はブランク文字として渡すことが必要です。

RUNQ サブプログラムは、渡された Query および書式の名前を読み取り、ブランク値がないかどうかをテストし、長さを計算し、RUN QUERY コマンドをフォーマット設定して、プログラム式インターフェースを呼び出します。 Query が開始された後に、RUNQ サブプログラムは制御権をその呼び出しプログラムに戻します。

注: RUNQ サブプログラムは、セッション中に何度も呼び出されることがあるため、実行後も終了しません。

```
 ****************************************************************
 *                                                              *
 *        RUN QUERY COMMAND CPI QUERY INTERFACE HANDLER         *
 *        ---------------------------------------------         *
 *                                                              *
 *  1) Include member DSQCOMMR contains the communications      *
 *     area to be passed to the query management interface.     *
 *  2) This program handles the RUN QUERY interface command.    *
 *     It reads the passed query name and form information,     *
 *     reformats it, then passes the information to query       *
 *     management.                                              *
 *                                                              *
 ****************************************************************
H
 *
E                    VAL        59  1               value to pass
 *
IRUNQ        DS
I                                       1  21 QNAM
I                                      22  42 FNAM
I            DS
I                                    B   1   40BIN1
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed run query command information:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
C                     PARM           RUNQ             query and form
 *
 * prepare the run query command:
 *
C                     MOVEA'RUN'     VAL              Set up RUN
C                     MOVEA'QUERY'   VAL,5            QUERY and
C                     MOVEAQNAM      VAL,11           Query name
C                     Z-ADD12        X       20       Set array index
```

図 22. RUNQ サブプログラムの例 (1/2)

```
 *
C           FNAM      IFNE *BLANKS                  Only if form
C           ' '       LOKUPVAL,X                 60Find next blank
C                     ADD  1         X              leave a space &
C                     MOVEA'(FORM='  VAL,X          insert (FORM=
C           ' '       LOKUPVAL,X                 60Find next blank
C                     MOVEAFNAM      VAL,X          form name
C                     END
 *
C           ' '       LOKUPVAL,X                 61Find last blank
C   61                SUB  1         X              Last non blank
C  N61                Z-ADD59        X              No blanks left
 *
 * process the run query command:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM         comms area
C                     PARM X         BIN1           command length
C                     PARM           VAL            command
 *
C                     RETRN
```

図 22. RUNQ サブプログラムの例 (2/2)

Query 管理機能 CI での RUNP サブプログラム

RUNP サブプログラムは、RUN PROC インターフェースを活動化します。プロシージャー名は、このプログラムに 33 文字のストリングとして渡されます。プロシージャー名は左寄せしなければなりません。
RUNP サブプログラムは、渡されたプロシージャー名を読み取り、長さを計算し、RUN PROC コマンドをフォーマット設定して、プログラム式インターフェースを呼び出します。プロシージャーが開始された後に、RUNP サブプログラムは制御権をその呼び出しプログラムに戻します。

注: RUNP サブプログラムは、セッション中に何度も呼び出されることがあるため、実行後も終了しません。

RPG 言語インターフェース

```
     ****************************************************************
     *                                                              *
     *         RUN PROC COMMAND CPI QUERY INTERFACE HANDLER         *
     *         -------------------------------------------          *
     *                                                              *
     *  1) Include member DSQCOMMR contains the communications      *
     *     area to be passed to the query management interface.     *
     *  2) This program handles the RUN PROC interface command.     *
     *     It reads the passed procedure name and form information, *
     *     reformats it, calculates the length, then passes the     *
     *     information to query management.                         *
     *                                                              *
     ****************************************************************
    H
     *
    E                    VAL        42  1               value to pass
     *
    I            DS
    I                                    B   1   40BIN1
    I/COPY BPLIB/QRPGSRC,DSQCOMMR
     *
     * receive the passed run procedure command information:
     *
    C           *ENTRY    PLIST
    C                     PARM           DSQCOM           comms area
    C                     PARM           RUNP   33        procedure name
     *
     * prepare the run procedure command:
     *
    C                     MOVEA'RUN'     VAL              Set up RUN
    C                     MOVEA'PROC'    VAL,5            PROC and
    C                     MOVEARUNP      VAL,10           Procedure name
     *
    C                     Z-ADD11        X       20       Set array index
    C           ' '       LOKUPVAL,X                     60Find last blank
    C   60                SUB  1         X                Last non blank
    C  N60                Z-ADD42        X                No blanks left
     *
     * process the run procedure command:
     *
    C                     CALL DSQCIR
    C                     PARM           DSQCOM           comms area
    C                     PARM X         BIN1             command length
    C                     PARM           VAL              command
     *
    C                     RETRN
```

図 23. RUNP サブプログラムの例

Query 管理機能 CI での EXIT サブプログラム

EXIT サブプログラムには、DSQCOM 連絡域に対する追加のパラメーターは必要ありません。このサブプログラムが呼び出されると、Query インターフェースを終了し、それ自体も終了して、このサブプログラムを呼び出したプログラムに制御権が戻されます。

```
 ****************************************************************
 *                                                              *
 *           EXIT COMMAND CPI QUERY INTERFACE HANDLER           *
 *           ----------------------------------------           *
 *                                                              *
 *  1) Include member DSQCOMMR contains the communications      *
 *     area to be passed to the query management interface.     *
 *  2) This program handles the EXIT interface command.         *
 *     It passes to query management the command length         *
 *     and command.                                             *
 *                                                              *
 ****************************************************************
H
 *
I            DS
I                                    B   1   40BIN1
I/COPY BPLIB/QRPGSRC,DSQCOMMR
 *
 * receive the passed communications area:
 *
C           *ENTRY    PLIST
C                     PARM           DSQCOM           comms area
 *
 * call the interface and end the session:
 *
C                     CALL DSQCIR
C                     PARM           DSQCOM           comms area
C                     PARM 4         BIN1             command length
C                     PARM 'EXIT'    DATA    4        command
 *
C                     MOVE '1'       *INLR            end program
```

図 24. EXIT サブプログラムの例

第 8 章 Query 管理機能でのオブジェクトのエクスポートとインポート

特定の Query 管理機能オブジェクトは、エディターまたはアプリケーションで処理するために、またはある環境から別の環境に移送するために、エクスポートすることができます。オブジェクトのエクスポートは、同じ Query プロダクトによっても、または異なる Query プロダクトによっても行うことができます。*QMQRY オブジェクトに関連付けられているソート・シーケンス表の内容を、エクスポートすることはできません。 以下のセクションでは、Query、プロシージャー、およびオブジェクトをエクスポートする場合の形式について説明しています。

注: 属性 PROMPT を持つ Query 管理機能プログラムは、属性 SQL を持つ Query と同じ方式でエクスポートされます。プロンプト Query (指示照会) については、*Query Manager ご使用の手引き* を参照してください。

Query 管理機能の一般オブジェクト形式

このセクションでは、多くのシステムに共通している Query 管理機能の形式を説明します。その他の形式、すなわち、各 Query 管理機能オブジェクトに固有の形式については、後のセクションで説明します。

Query 管理機能の外部化されたオブジェクトのコメント

オブジェクト・コメントは、オンラインまたは印刷コピーで表示された時点で外部化オブジェクトになります。 Query 管理機能の外部化書式オブジェクト、Query、またはプロシージャー・オブジェクトには、オブジェクトコメントを 1 つ入れることができます。このコメントはフィールド番号が 1001 で、長さが最大 50 までの V レコードとして指定しなければなりません。 50 文字より長いコメントは、超えた部分が切り捨てられます。コメントは、外部化されたオブジェクトをエクスポートする時に、そのオブジェクト内に生成されます。コメント・レコードが存在する場合、そのコメント・レコードは H レコードの直後に続けなければなりません。 H レコード・タイプ・フィールドは次のいずれかでなければなりません。

- 書式の場合は F
- Query の場合は Q
- プロシージャーの場合は P

インポート時 に、どのテキスト記述が Query 管理機能オブジェクトで使用されるかは、次のようにして決められます。

- COMMENT= オプションが指定されている場合、このオプションの値が使用されます。
- メンバー中にメンバー・テキストがある場合、そのメンバー・テキストが使用されます。
- オブジェクト中にコメントがある場合、そのコメントが使用されます。
- コメントが存在しない場合、テキスト記述はブランクになります。

エクスポート時 に、外部化 Query 管理機能オブジェクトにどのテキスト記述が書かれるかは、次のようにして決められます。

- COMMENT= オプションが指定されている場合、このオプションの値が使用されます。
- COMMENT= オプションが指定されていない場合、Query 管理機能オブジェクトの中のテキスト記述が使用されます。

Query オブジェクトまたはプロシージャー・オブジェクトの場合に、エンコードされた形式の一部として使用できるのは、H レコードおよび V レコードだけです。 T、R、E、および * など、その他のレコード・タイプを使用すると、予測できない結果になります。

コメントを複数行にまたがって使用することはできません。また他のコメントの中にコメントを使用することもできません。コメントは '/*' で始まり、'*/' で終ります。

プロシージャーは、オブジェクト・コメントおよびユーザー・コメントを持っています。プロシージャー内のオブジェクト・コメントはコメント区切り文字の内側になければならず、開始コメントとレコード ID の間にブランクを入れることはできません。ユーザー・コメントは実行時には無視されます。

次の例は、1 つのオブジェクト・コメントといくつかのユーザー・コメントを持つプロシージャーの例です。

```
/*H QM4 01 P 01 E V W E R 01 03 90/07/24 13:30 */
/*V 1001 016 SALES PROCEDURE */
'IMPORT QUERY SALES FROM QRYSRC'
'RUN QUERY SALES'  /*Total sales query*/
'PRINT REPORT'  /*Management report*/
```

次の例は、Query 内のオブジェクト・コメントです。

```
H QM4 01 Q 01 E V W E R 01 03 90/06/38 01:25
V 1001 011 SALES QUERY
SELECT SALARY FROM SALESFILE WHERE
SALARY > 50000
```

Query 管理機能の外部形式

オブジェクトの形式を以下に簡単に説明します。後続のセクションで、オブジェクトのそれぞれの形式について説明してあります。

プロシージャー
パネル形式

SQL 照会
パネル形式

書式 エンコードされた形式

Query 管理機能のパネル形式

この形式は、オブジェクトが一連のテキスト・ストリングとして入っている多くの固定長レコードで構成されます。レコード内では、オブジェクトには特別なフォーマット設定は行われません。アプリケーションによって構成された状態、または端末から入力された状態のままです。

この形式で書き出されたオブジェクトは、次の属性を持っています。

- 79 バイトの論理レコード長
- 固定長レコード形式

Query 管理機能のエンコードされた形式

この外部形式によって、単純なパネル形式オブジェクトよりも多くの構造を含む、Query 管理機能オブジェクトにアクセスすることができます。この形式は、FORM オブジェクトの場合にだけ使用してください。

Query 管理機能のエンコードされた形式のサイズ

エンコードされた形式では、レコード長は 150 文字以下でなければなりません。

Query 管理機能で基本エンコード形式を形成するレコード

基本エンコード形式を形成するレコードの形式について、以下で説明していきます。その他のエンコード形式レコード・タイプ (特定の Query 管理機能オブジェクトに関連する) については、後続のセクションで個々のオブジェクトの外部形式を扱う際に説明します。

各レコード・タイプごとに、その目的、実際の内容、形式、およびその使用に関する一連の注を記述してあります。レコードの説明では、レコードの各フィールドごとに使用可能な値について、正確で完全なリストを示してあります。これらのフィールド (特に見出しレコードの) には、単一の値しか入れることができないものもあります。これは意図的に行っているもので、Query 管理機能の今後のリリースでは、そのようなフィールドに別の値を入れられるようになることを意味している場合がしばしばあります。

エンコードされた形式のレコード・タイプの基本セットは、次のとおりです。

レコード・タイプ	記述名
“H”	『Query 管理機能の見出し (“H”) レコード』
“V”	133 ページの『Query 管理機能の値 (“V”) レコード』
“T”	135 ページの『Query 管理機能の表記述 (“T”) レコード』
“R”	137 ページの『Query 管理機能の表行 (“R”) レコード』
“E”	139 ページの『Query 管理機能のオブジェクトの終わり (“E”) レコード』
“*”	139 ページの『Query 管理機能のアプリケーション・データ (“*”) レコード』

ここに定義されていないフィールドの使用を許すようにするには、アプリケーションを作成する必要があります。すなわち、定義されていないか、または理解されないフィールドおよびレコードは、アプリケーションでは無視しなければなりません。そのようなフィールドを作っておくと、アプリケーションは、今後の Query 管理機能のリリースで生成されるオブジェクトを使用し、実行することができます。このような設計によって、サポートされない機能を含むオブジェクトの使用も許されることになります。

Query 管理機能の見出し (“H”) レコード

H レコードは、外部化されたオブジェクトの内容を識別するものです (オブジェクトは、オンライン表示またはハード・コピー印刷されると、外部化されたオブジェクトになります)。 H レコードには、オブジェクトの特性およびファイル形式を記述する情報が含まれます。

次の図は、見出しレコードの内容を要約したものです。

```
    ....+....1....+....2....+....3....+....4....+

形式  :   Hdppp rr t oo f u s n a cc ii yy/mm/dd hh:mm
               |  | |  | | | | | |  |  |        |     |
              d  d d  d d d d d d  d  d        d     d

          |_____________________________________|
                             |
                             |
                        固定形式

ここで:   H         オブジェクトの見出しレコードであることを示す
          d         このレコード専用のデータ・フィールド区切り文字
                    -- ブランク 1 つ
          ppp       プロダクト ID:
                      QM4
          rr        オブジェクトが作成されたプロダクト・レベル:
                      03、04、05 など
          t         このファイルのオブジェクトのタイプ:
                      F は書式、Q は Query、P はプロシージャー
          oo        指定のタイプのオブジェクトが作成された時点の
                    プロダクト・オブジェクト・レベル:
                      01、02、03 など
          f         このファイルの中のオブジェクトの書式:
                      E エンコードされた形式
          u         オブジェクトの状況を示す:
                      E エラーが含まれる場合
                      W 警告が含まれる場合
                      V 有効な場合
          s         オブジェクトに組み込まれているサブセットを示す:
                      W オブジェクト全体の場合
```

図 25. 見出しレコードの説明 (1/2)

```
          n         エクスポート・オブジェクトの言語を示す:
                      E 英語
          a         項目に対する処理:
                      R オブジェクト置換
          cc        各接続レコードの始めにある制御域の長さ
                     (1 バイトのレコード・タイプを含む):
                      01 の場合
          ii        "V" および "T" タイプ・レコードに
                    指定される整数長フィールドの長さ:
                      03 すべてのオブジェクトの場合
          yy/mm/dd  日付スタンプ
          hh:mm     タイム・スタンプ

例:       H QM4 03 F 03 E V W E R 01 03 89/09/23 15:21

           ("オブジェクト・レベル" 3 での QM4 FORM ファイル。
           エンコードされた形式で作成され、エラーおよび警告なし。全体が英語。
           完全な置換が使用可能。制御域は 1 バイト。
           整数長フィールドは 3 バイト)
```

図 25. 見出しレコードの説明 (2/2)

次の図には、見出しレコードの中の各フィールドの位置および内容を要約してあります。この図で使用されているフィールド名は、見出しレコード全体を説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
H	01	H	要	
d	02	(ブランク)	不要	(入力には適用されない)
ppp	03-05	QM4	要	
rr	07-08	03	不要	(入力には使用されない)
t	10	F (書式) Q (Query) P (プロシージャー)	要	
oo	12-13	03	不要	現行オブジェクト・レベル
f	15	E	要	
u	17	E W V	不要	(入力には使用されない)
s	19	W	不要	(入力には使用されない)
n	21	E (英語)	不要	(入力には使用されない)
a	23	R	要	
cc	25-26	01 (書式)	不要	(入力には使用されない)
ii	28-29	03	不要	(入力には使用されない)
yy/mm/dd	31-38	(日付)	不要	(入力には使用されない)
hh:mm	40-44	(時刻)	不要	(入力には使用されない)

図 26. 見出しレコードのフィールド

図 26 の注:

- 見出しレコードが外部ファイルの最初のレコードでなければなりません。
- レコードが固定形式の長さより短い場合、指定されないままのフィールド (またはフィールドの一部) はブランクと見なされます。
- オブジェクト・レベル (“oo”) は、オブジェクトの外部化された形式の変更を示すために使用されます。Query 管理機能の特定のレベルで外部オブジェクトが変更されると、そのオブジェクトのレベル番号は大きくなります。アプリケーションはこの番号を使用して、オブジェクトのレコードの形式を判別することができます。
- 制御域 (長さが “cc”) は、エンコードされた形式のレコード (見出しレコードを除く) のそれぞれの始めにあって、指定したレコードに関する制御情報の入る固定域です。この制御域には、レコード・タイプおよびレコード継続標識などの情報が入れられます。
- 見出しレコードには、エンコードされた形式を将来拡張するためのサブセット、形式、アクション、制御域の長さ、および整数長フィールドが含まれています。
- 将来、追加のフィールドが見出しレコードの終わりに追加されます。

Query 管理機能の値 (“V”) レコード

V レコードを使用して、オブジェクトの中の単一表ではないフィールド (FORM OPTIONS フィールドなど) の値を提供します。そこには固有のフィールド番号、そのフィールドの値、およびその長さが入っています。

次の図は、V レコードの内容を要約したものです。

```
様式:      Vc..cdf..f l..l v..v
                      |   |   |
                      d   d   d

          |__| |__________|
            |          |
            |          |
          制御域    レコード情報

ここで、    V          値レコードであることを示す
            c..c       このレコードの制御域の残りの部分
                       record
                         (将来使用するために確保されている)
            d          このレコード専用のデータ・フィールド区切り文字
                       -- ブランク 1 つ
            f..f       フィールド番号
                         例: 1201、1509
            l..l       データ値の長さを示す:
                         数値の代りに * を使用すると、データ値が
                         レコードの終わりで区切られることを示す

            v..v       データ値 (印刷可能な書式)

例:         V 1511 004 NONE

              (書式フィールド 1511、長さ 4 バイト、値 'NONE')
```

図 27. 値レコードの説明

次の図は、V レコードの中の各フィールドの位置および内容を要約したものです。図で使用されているフィールド名は、V レコード全体を説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

	制御域			
フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
V	01	V	要	
c..c	02	(x)	適用外	
x	02	(ブランク)	適用外	

図 28. 値レコードのフィールド (1/2)

		レコードのその他の部分		
フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
d	+01	(ブランク)	不要	
f..f	+02-05	1001-9999	要	
l..l	+07-09	* 000-999	要	
v..v	+11-終わり	(データ)	不要	(ブランク)

図 28. 値レコードのフィールド (2/2)

134 ページの図 28 の注:

- 省略されたデータ値 (レコードの終わりなど) または “*” にだけ続くブランクは、ヌル (ブランクと同じ) 値がこのフィールドに適用されることを暗黙に示します。
- フィールドを明示的にブランクにセットするためには、そのフィールドには正の長さを指定し、ブランク・データ値が入っていなければなりません。
- フィールドは、オブジェクトが次のような場合に更新された時にデフォルトにセットされます。
  - 指定された長さがゼロの場合
  - 長さが指定されていない場合
- Query 管理機能は、フィールド長が 0 であることを見つけると、このフィールドにデフォルトがセットされることを示す警告を出します。
- 指定された長さが提供データ値よりも短いと、Query 管理機能は、指定された長さの方を使用し、警告メッセージを出します。
- 指定された長さが提供データ値よりも長いと、Query 管理機能は、レコードの終わりを超えたデータ値のセットは行わず、警告メッセージを出します。
- IBM では、将来の V レコードの使用 (たとえば、意味のあるブランクを示すために明示的に長さを指定できる)、および V レコードの拡張に備えて、長さフィールドを残しています。

Query 管理機能の表記述 (“T”) レコード

T レコードは、後に続く値の表の内容および形式を記述します。 T レコードの内容によって、この表のすべての R (行) レコードの内容が決まります。 T レコードは、記述されている表 (その固有の表番号によって)、表に含まれる列 (その固有のフィールド番号によって)、列が現れる順序、および列の値の長さを示します。

次の図は、T レコードの内容を要約したものです。

```
様式:      Tc..cdt..t n..n m..m f..f l..l f..f l..l  ...
                     |    |    |    |    |    |    |
                     d    d    d    d    d    d    d

          |__|              |____| |_____| ...
            |                   |       |
            |                   |       |
          制御域              フィールドと長さを組にしたものの繰り返し


ここで、     T          表記述レコードであることを示す。
            c..c       このレコードの制御域の残りの部分
                       次のもので構成される
                         x
                       ここで、
                         x      ブランク (将来使用するために確保されている)
            d          このレコード専用のデータ・フィールド区切り文字
                       -- ブランク 1 つ
            t..t       表番号
                         例: 1110、2710
            n..n       この表の中の行 ("R" レコード) の数
                         数値の代りに * を使用すると、データ値が
                         に続くすべての "R" レコードから構成される
                         ことを示す
            m..m       この表の中の列 (フィールドと長さ) 数
                         例: 003、006
            f..f       この列のフィールド番号
                         例: 1113、2712
            l..l       この列のデータ値の長さ
                         例: 005、012

例:         T 1110   5   2 1112   7 1113  18

              (書式表 1110。長さが 7 の列 1112 と
              長さが 18 の列 1113 の 2 列を、5 行含む)
```

図 29. 表レコードの説明

次の図は、T レコードの中の各フィールドの位置および内容を要約したものです。図で使用されているフィールド名は、T レコード全体を説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

制御域

フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
T	01	T	要	
c..c	02	(x)	適用外	
x	02	(ブランク)	適用外	

図 30. 表レコードのフィールド (1/2)

レコードのその他の部分

フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
d	+01	(ブランク)	不要	
t..t	+02-05	1001-9999	要	
n..n	+07-09	* 000-999	要	
m..m	+11-13	000-999	要	
f..f	+15-18 +24-27	1001-999	要	
l..l	+20-22 +29-31 など	000-999	要	

図 30. 表レコードのフィールド (2/2)

136 ページの図 30 の注:

- T レコードに続く R レコードの数が T レコードで指定された行数カウントに正確に一致しない場合、エラー条件が生じます。
- f..f/l..l の対の数は、指定の表の列の数に制限されます。
- 列の数は列フィールドの数と長さに一致していなければなりません。一致していない場合には、警告メッセージが出され、 T レコード内のフィールド番号 / 列データ値の長さの数が使用されます。
- f..f/l..l の対の順序は任意です。
- T レコードの直後のすべての R レコード (つまり、 1 つの表に関連したレコード) には、各列について T レコードに指定された正確な長さの値が入っていなければなりません。レコードが暗黙の長さより短い場合には、ブランクまたはブランクが埋め込まれた値となります。
- Query 管理機能は、長さ 0 (または指定のない) の列を、オブジェクトの更新時にデフォルトにセットします。
- Query 管理機能は、指定された長さが 0 の列について、その列がデフォルトにセットされることを示す警告メッセージを出します。
- ゼロ行の表 (またはファイルに組み込まれていない表) は、長さがゼロの列を表に適用するのと同じ効果があります。 Query 管理機能は、すべての列をそのデフォルトにセットします。
- 列フィールドをブランクにセットするためには、 T レコードにその列の正の長さ、R レコードにはブランクの値がなければなりません。

Query 管理機能の表行 (“R”) レコード

R レコードは、現行表内の単一行に 1 組の値を提供するために使用されます。このレコードは、値を関連した T レコードによって記述されたとおりの順番に並べたリストで構成されます。 R レコードは、T レコードに指定されているとおりのデータ値の位置および長さの記述に正確に一致していなければなりません。

次の図は、R レコードの内容を要約したものです。

```
様式:       Rc..cdv..v v..v v..v ...
                       |    |    |
                       d    d    d

            |__||_____________| ...
              |        |
              |        |
            制御域     値のリスト

ここで、    R          表行レコードであることを示す
            c..c       このレコードの制御域の残りの部分
                       次のもので構成される
                       x
                       ここで、
                       x      ブランク (将来使用するために確保されている)
            d          このレコード専用のデータ・フィールド区切り文字
                       -- ブランク 1 つ
            v..v       この行と列のデータ値 (印刷可能形式)

例:         R       2 SALARY

              (書式 1 列目の値は '       2'、長さは 7。
              2 列目の値は 'SALARY'。
              長さは T レコードに少なくとも 6 が指定されていると
              見なされる)
```

図 31. 行レコードの説明

次の図は、R レコードの中の各フィールドの位置および内容を要約したものです。図で使用されているフィールド名は、R レコード全体を説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

制御域

フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
R	01	R	要	
c..c	02	(x)	適用外	
x	02	(ブランク)	適用外	

図 32. 行レコードのフィールド (1/2)

レコードのその他の部分

フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
d	+01	(ブランク)	不要	
v..v	+02-xx +(xx+2)-yy +(yy+2)-zz など	(データ)	不要	(ブランク)

図 32. 行レコードのフィールド (2/2)

図 32 の注:

- R レコードは、別の R レコードまたは T レコードの直後になければなりません。
- v..v の値の数は、関連した T レコードの記述と正確に一致していなければなりません。
- 関連した T レコード中の長さがゼロのデータ値は、オブジェクトのこの行および列に値が適用されないことを示します。したがって、それはデフォルトにセットされます。ただし、T レコードにフィールドが存在していることによって、このフィールドのための余分な区切り文字が R レコードに入っていることが必要となります。長さの値をゼロにすると、 R レコード内で 1 つの区切り文字に別の区切り文字が続くことになります。

Query 管理機能のオブジェクトの終わり (“E”) レコード

E レコードは、オブジェクトの終わりを区切るために使用されます。

次の図は、E レコードの内容を要約したものです。

```
                ....+....1....+....2....+....3....+....

様式:           Ec..c
                |__|
                  |
                  |
                制御域

ここで、        E          オブジェクトの終わりレコードであることを示す
                c..c       このレコードの制御域の残りの部分
                           (将来使用するために確保されている)

例:             E
```

図 33. オブジェクトの終わりレコードの説明

次の図は、E レコードの中の各フィールドの位置および内容を要約したものです。図で使用されているフィールド名は、E レコード全体を説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

		制御域		
フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
E	01	E	要	
c..c	02	(x)	適用外	
x	02	(ブランク)	適用外	

図 34. オブジェクトの終わりレコードのフィールド

図 34 の注:

- E レコードは、外部ファイルの最後のレコードでなければなりません。
- レコードが固定形式の長さより短い場合、指定されないままのフィールド (またはフィールドの一部) はブランクと見なされます。

Query 管理機能のアプリケーション・データ (“*”) レコード

アプリケーション・データ・レコードによって、指定したオブジェクトと関連するユーザー自身のデータを、外部ファイルに組み込むことができます。これらは、ファイル内のオブジェクトを詳細に説明するためのコメント・レコードとして使用するように選択することもできます。

次の図は、アプリケーション・データ・レコードの内容を要約したものです。

```
形式:       *v..v

ここで、    *          アプリケーション・データ・レコードであることを示す
            v..v       適応業務プロシージャーによって作成されるデータ値
                       (印刷可能形式が望ましい)

例:         * これは DEPT によってグループ化する書式です。

              (ファイルのコメント・レコード)
```

図 35. アプリケーション・データ・レコードの説明

次の図は、* レコードの中の各フィールドの位置および内容を要約したものです。図で使用されているフィールド名は、 * レコード全体について説明した前述の図で定義したものです。オブジェクト固有の値については、必要に応じて注があります。

	制御域			
フィールド	桁	使用できる値	入力の必要	入力がブランクの場合のデフォルト
*	01	*	要	
v..v	02-終わり	(データ)	不要	(入力には使用されない)

図 36. アプリケーション・データ・レコードのフィールド

図 36 の注:

- アプリケーション・データ・レコードは、見出しレコードの前以外の外部ファイルの任意の位置に表示することができます。
- レコードの形式の妥当性検査以外では、アプリケーション・データ・レコードは無視され、入力処理に影響しません。
- レコードが固定形式の長さより短い場合、指定されないままのフィールド (またはフィールドの一部) はブランクと見なされます。

Query 管理機能の EXPORT ファイルおよび IMPORT ファイルに関する考慮事項

次にファイルのエクスポートおよびインポートに関する考慮事項を示します。

- Query 管理機能によって、複数メンバーのソース・ファイルに対してエクスポートおよびインポートを行うことができます。
- 各レコードに必要なソース・シーケンス*番号* および*日付* フィールドのための 12 桁を考慮した、レコード長を持つソース・ファイルを作成しなければなりません。
- Query 管理機能は、既存のソース・ファイルへのエクスポート時に、そのファイルに十分なデータ長がなかった場合には、データを切り捨てて警告メッセージを生成します。次の状況が切り捨ての原因となる可能性があります。
  - データ長が 79 バイトより小さいファイルへ SQL Query オブジェクトまたはプロシージャー・オブジェクトをエクスポートする場合。 Query 管理機能は、SQL ステートメントまたは Query コマンドの一部が切り捨てられた場合には、切り捨て警告メッセージを作成します。

– データ長が不十分なファイルへ Query 管理機能書式オブジェクトをエクスポートする場合。使用可能な最小のデータ長は 150 バイトです (これは、エクスポートされるエンコードされた形式の書式レコードである、列表 R レコードの最大長に基づいています)。 Query 管理機能がエクスポートの結果としてファイルを作成する場合には、 150 バイトのデータ長を持つファイルを作成します。したがって、150 バイトのデータ長が必要になります。 Query 管理機能は、データの切り捨てが起こった時に、切り捨て警告メッセージを出してきます。 書式のいくつかの部分はオプションであるため、実際に必要なデータ長は 150 バイトより小さくなります。

- ソース物理ファイルのメンバーが置き換えられている場合には、確認メッセージが送られます (EXPORT コマンドに CONFIRM=YES が指定されている場合)。すでに存在しているソース物理ファイルの新しいメンバーの作成中は、このメッセージは送られません。
- Query 管理機能は、SQL Query オブジェクトまたはプロシージャー・オブジェクトのインポート時には、ファイル中の 79 桁目より後のすべての列を無視します。 列が無視された場合には、メッセージが生成されます。
- Query 管理機能は、書式オブジェクトのインポート時には、ファイル中の 150 桁目より後のすべての列を無視します。 列が無視された場合には、メッセージが生成されます。
- OS/400 システムは、可変レコード長のファイルはサポートしていません。したがって、インポートのために iSeries システムへ転送する前またはその転送中に、エンコードされた形式で外部化書式オブジェクトが入っているファイルを固定レコード形式に変換しなければなりません。オブジェクトのエクスポート時には、Query 管理機能が固定長レコード形式のファイルを作成します。このレコードの意味のあるデータの終わりから、そのレコードの終わりまでブランクが埋め込まれます。したがって、固定形式の外部化書式をサポートしていないプロダクトをインポートする場合には、その前にファイルを可変長レコード形式に変換しなければなりません。
- Query 管理機能は、 SQL Query オブジェクトまたはプロシージャー・オブジェクトのインポート時に、入力ファイルのデータ長が 79 バイトより小さい場合には、各レコードの内部表現の 79 桁目 (その桁も含む) までブランクを埋め込みます。引用符に囲まれた DBCS 混用ストリングが行に含まれる場合には、この埋め込みはそのストリング内に行われ、予期しない結果になることがあります。
- 外部化 SQL Query またはプロシージャーを含むソース物理ファイルは、ソース物理ファイルについて iSeries システムで使用可能な任意のサイズにすることができます。

  注: ソース物理ファイルはシステムで使用可能な任意のサイズにすることができますが、インポート時にその最大許容限界を超えた場合には、結果として切り捨てられることがあります。インポートの制約についての詳細は、 SQL 解説書を参照してください。
- IMPORT FORM の実行中に、COLUMNS.DATATYPE フィールドが DATE、 TIME、または TIMESTAMP であり、COLUMNS.USAGE フィールドが AVG または SUM である場合には、使用目的およびデータ・タイプが不適合であることを示すメッセージが表示されます。 COLUMNS.USAGE は使用されず、IMPORT 処理は警告を伴って終了します。
- 同じことが RUN QUERY または PRINT REPORT の実行中に適用された場合、メッセージが表示され、RUN QUERY または PRINT REPORT は正しく実行されません。 (外部化書式でインポート時に COLUMNS.DATATYPE が指定されていない場合、この状態が生じます。)

Query 管理機能の未確定の日付および時刻用リテラル

SQL ステートメントに示される日付および時刻用のリテラルは、いずれかの SQL 形式とするか、あるいは有効な OS/400 の日付または時刻の形式でなければなりません。 142 ページの表 15 は、SQL 時刻形式を示しています。

表 15. 時刻データ・タイプの表示形式

形式名	省略形	時刻形式	例
国際規格	ISO	HH.MM.SS	13.30.05
IBM USA 標準規格	USA	HH:MM am または pm	1:30 pm
IBM 欧州標準規格	EUR	HH.MM.SS	13.30.05
日本工業規格 (西暦)	JIS	HH:MM:SS	13:30:05

表 16 は、SQL ステートメントで使用できる SQL 日付形式を示しています。

表 16. 日付データ・タイプの表示形式

形式名	省略形	日付形式	例
国際規格	ISO	YYYY-MM-DD	1987-10-12
IBM USA 標準規格	USA	MM/DD/YYYY	10/12/1987
IBM 欧州標準規格	EUR	DD.MM.YYYY	12.10.1987
日本工業規格 (西暦)	JIS	YYYY-MM-DD	1987-10-12

国または地域に特有の日時形式は、間違って解釈されることがあります。 Query にグローバル変数が含まれており、日付形式がジョブとは異なっていると、この状態が生じます。このため、Query 管理機能では、これらのあいまいな日付および時刻リテラルが使用されると、それを認識して、エラー内容を説明するメッセージを出して、失敗します。問題を修正する方法を示すメッセージが表示されます。

Query 管理機能では、Query のソース仕様のインポート時にはあいまいな日付および時刻リテラルは認識されません。あいまいな表現とならないように、Query のソース仕様で適切に文書化しておいてください。

注: ジョブの日付および時刻の形式または区切り記号は、 Query インスタンスを開始した後では変更してはなりません。 IMPORT および RUN QUERY コマンドでは、 START コマンドの実行時に有効であった日付および時刻情報が使用されます。

Query 管理機能の可変長フィールド

可変長フィールドは 2 バイトからなり、その後にデータが続きます。最大長は 32,740 です。可変長フィールドには、文字 (1 バイト文字セット (SBCS) またはブラケット 2 バイト文字セット (DBCS))、グラフィック DBCS、または 16 進数を使用することができます。 DBCS データに関する考慮事項および最大長については、
221 ページの『付録 A. Query 管理機能での DBCS データ』を参照してください。

制御の切れ目処理でフィールドを比較する場合、後続のブランクは重要ではありません。長さの異なる 2 つのストリングで後続のブランクの数だけが異なっている場合には、この 2 つのストリングは等しくなります。

最大長が 32,740 (ヌル値が使用可能な場合は 32,739) である可変長フィールドを定義することができます。割り振られた長さをこのフィールドに指定することもできます。パフォーマンスを向上させるためには、この最適の長さをその割り振られた長さとして定義してください。多くのレコードはこの長さか、またはそれより短い長さであり、固定データ記憶域に記憶されます。レコードがこの割り振られた長さより長い場合には、可変長データ記憶域に記憶されます。レコードはすべて正しく記憶され、補助記憶装置への余分な読み込みは長いレコードの場合にのみ必要となります。

Query 管理機能の画面形式

画面形式を使用する外部化プロシージャー・オブジェクトおよび SQL Query オブジェクトの詳細については、『第 5 章 Query 管理機能のプロシージャー』 および 『第 2 章 Query 管理機能での照会機能』を参照してください。

Query 管理機能のエンコードされた形式

Query 管理機能では、外部化書式オブジェクトについてのみエンコードされた形式が使用されます。このセクションでは、Query 管理機能によって使用されるエンコードされた形式の各レコード・タイプを説明しています。 Query 管理機能では、本書で定義されていないフィールドも使用することができます。

注: 認識できないフィールドはインポート時に失われ、その後のエクスポート時にも表示されることはありません。

Query 管理機能での書式オブジェクトのインポート

書式オブジェクトをエンコードされた形式で Query 管理機能にインポートする時には、次の規則が適用されます。

- H レコードがファイルの最初のレコードでなければなりません。
- ファイル内で E レコードの前に見つかった H、V、T、R、および * 以外のレコード・タイプは無視されます。
- 未知のレコード・タイプが見つかった場合には、警告メッセージが出されます。
- E レコードの後のレコードは無視されます。
- 列表の T レコードは、見出しレコードまたはコメントの V レコードの直後になければならず、表の行数の数値カウントを含んでいなければなりません (* 行カウントは使用できません)。
- Query 管理機能は、H レコード内の制御域の長さ (cc) フィールドは無視します。
- Query 管理機能は、すべての書式オブジェクト・レコードの制御域の長さの値を 01 と想定します。
- Query 管理機能は、各書式オブジェクト・レコードに指定された区切り文字の値を使用します。 したがって、非ブランク区切り文字も使用可能です。
- H レコードは列に固有のものであるため、そのレコードの区切り文字の値は無視されます。
- 次のフィールドは、書式オブジェクトがインポートされた時に大文字でなければなりません。
  - すべてのレコードのレコード ID
  - 見出しレコード内にある次のもの
    - プロダクト ID (QRW、QMF*、QM4 など)
    - オブジェクトのタイプ (F)
    - オブジェクトの形式 (E)
    - アクション (R)
  - 列表用の R レコードの中のデータ・タイプ値 (NUMERIC、CHAR)。
  - すべてのオブジェクト・キーワードおよび置換変数。
- 重複して現れるデータ値または表があった場合には、 1 つの例外を除いて、それが前に現れた設定値を一時変更することになります。 Query 管理機能では、新しいオブジェクトがそのオブジェクトについて設定された規則に反している場合でも、前の設定値は一時変更されません。たとえば、書式について指定された列の数は、最初の列表が処理された後は変更することができません。
- 切れ目情報を表示するために、元の形式と新しい形式を組み合わせることはできません。
- 入力ファイルに含まれていないオブジェクトの値は、そのデフォルトにセットされます。

Query 管理機能での列表の明細

書式には、基礎となるデータのためのすべての列が入っていなければなりません。書式とデータの不一致は、RUN または PRINT の実行時にその書式が適用されるまで検出されません。

列表全体が読み込まれる時には、まだ指定されていないフィールドはデフォルトにセットされません。デフォルトは実行時に適用され、照会中の表からのファイル定義で定義されている値となります。列表フィールドが脱落したままインポートされた書式をエクスポートした場合には、結果として、同じ列表フィールドが脱落した外部化書式となります。 Query 管理機能では、実行時にデフォルトが適用される場合、OS/400 の編集コードが考慮されてそれが使用されます。これらのフィールドが指定されていない場合には、インポートおよびエクスポート時に警告メッセージが生成されます。

Query 管理機能では、列表が複数あっても問題ありません。列の数を変更するような表の追加は認められません。

Query 管理機能は、GRAPHIC データ・タイプを指定した書式のインポートをサポートしていますが、SQL 規則および DB2 UDB for iSeries はこのデータ・タイプをサポートしていません。 Query 管理機能ではこのデータ・タイプが入っている書式オブジェクトのインポートが可能ですが、この書式を適用しようとすると、結果として実行時にデータと書式のミスマッチ・エラーが起こります。サポートされていないデータ・タイプを指定するオブジェクトがインポートされた場合には、警告メッセージが生成されます。認識できないかまたは正しくない列表の値は無視されて、値はデフォルトと見なされます。

Query 管理機能での書式オブジェクトのエクスポート

Query 管理機能では、インポートされたオブジェクトで使用されている区切り文字にかかわらず、ブランク区切り文字が使用されます。 インポート時にデフォルトとされた情報が、列以外のすべての報告書セクションについてエクスポートされます。 Query 管理機能は、切れ目情報を記述するための新しい形式を使用して、書式オブジェクトをエクスポートします。

Query 管理機能のレコード形式の規則

入力 (インポート) の場合:

1. ファイルは、可変長レコードでも固定長レコードでも構成できます。
2. 最小許容論理レコード長は 23 で、必要な見出しレコード形式および内容に基づくものです。
3. レコード・タイプ文字 (H、V、T、R、E、および *) は、各レコードの 1 桁目になければなりません。
4. すべてのレコードの最初の “cc” バイトは、制御情報用に確保されるもので、したがって固定形式が必要です (“cc” はオブジェクトによって異なります)。
5. すべてのデータ・オブジェクト (フィールド番号、長さ、および値を含む) は、正確に 1 つの区切り文字をその前後に付ける必要があります。

   この規則には次のような 2 つの例外があります。 1) レコードの終わりは区切り文字として数えられます。 2) ゼロ長 (ヌル) の値は、“存在しない” 値を囲む 1 対の区切り文字を必要とします。
6. 入力で必要なすべてのフィールドは、妥当性を検査されます。
7. 単一のデータ値または表が重複して現れた場合、前の設定値が無効になります。ただし、次の場合は例外です。すなわち、特定のオブジェクトについて確立されている規則に対して新しいオブジェクトが違反している場合は、Query 管理機能は、前の設定値を無効にしません。たとえば、最初の列表が処理された後では、書式について用意されている列の数は変更できないなどです。
8. 入力ファイルに組み込まれていないオブジェクトの部分は、デフォルトにセットされます。

9. 数値の長さ、表番号、およびフィールド番号は、先行ゼロまたは先行ブランク (あるいはその両方) を付けて指定することができますが、後書きブランク (単一のブランク区切り文字以外) で埋め込むことはできません。レコード内の位置で右寄せにしなければなりません。
10. 非数値長 (* 指定) は、整数長値 (見出しレコードで指定) の現在の長さになるまで、後書きブランクで埋め込まなければなりません。
11. オブジェクト・フィールド値がオブジェクト・パネルのデータ入力フィールドより短い場合、後書きブランクを埋め込みます。
12. オブジェクト・フィールド値がオブジェクト・パネルのデータ入力フィールドより長い場合は、切り捨てられます。
13. FORM ファイルの形式がエラーの場合、IMPORT は打ち切られます。たいていの場合、エラーおよびファイル内のエラー位置を、単一メッセージで説明します。
14. ファイル内に E レコードに続くレコードがあっても、これは無視されます。
15. 入力ファイルに E レコードが入っていない場合、ファイルの終わりがオブジェクトの終わりを暗黙指定すると見なされます。

出力 (エクスポート) の場合:

1. 表番号およびフィールド番号は、すべて 4 桁の数値です。
2. すべての長さは 3 桁の長さ (見出しレコードに指定) になるように、先行ゼロを付けて書かれます。
3. ブランク文字は、すべてのレコード内で使用されるデータ・オブジェクト区切り文字です。
4. 区切り文字が各レコードの最終文字となることはありません。
5. すべての予約フィールドは、ブランクを用いて表示されます。
6. MVS™ のもとで、QMF™ で事前に割り振られたデータ・セットにエクスポートする場合、データ・セットのレコード形式は可変 (V) でなければならず、またそのデータ・セットの論理レコード長はエクスポートされるオブジェクトの実際の最大出力論理レコード長を満たすものでなければなりません。出力データ・セットのレコード長が事前に割り振りされた最大レコード長よりも大きい場合には、エラーが生じます。
7. 表列は外部化書式では、数値フィールド番号順に表示されます。ただし、列見出しフィールドは例外で、最後になります。
8. E レコードはターゲット・ファイルの最後のレコードとして表示されます。

Query 管理機能の特定の Query オブジェクトの形式

以下のセクションには、Query 管理機能オブジェクトのリストと、外部化された形式の例を挙げてあります。

『Query 管理機能の外部化 FORM 形式』

158 ページの『Query 管理機能の外部化 PROC 形式および QUERY 形式』

Query 管理機能の外部化 FORM 形式

FORM (書式) は、Query 管理機能によって、エンコードされた形式に外部化されます。この形式の仕様については、すでに 129 ページの『Query 管理機能の一般オブジェクト形式』で説明してあります。

FORM オブジェクトでは、130 ページの『Query 管理機能のエンコードされた形式』で説明した H、V、T、R、および E レコードを使用します。

特定の Query オブジェクトの形式

基本エンコード形式との相違、および書式に固有のその他の考慮事項について、以下で説明します。

- 入力 FORM には、基礎データのすべてが含まれなければなりません。
- COLUMNS 表は、見出しレコードまたは V レコードに続いて指定される最初の部分でなければなりません (しかも、この表が後で重複して生じることによって、サイズが変更されてはなりません)。
- COLUMNS 表には、TABLE 内の行数の数値カウントが組み込まれなければなりません ("*" 行カウント指定ではなく)。
- COLUMNS 表全体が読み込まれると、指定されていないフィールドはデフォルトにセットされます。
- COLUMNS.DATA_TYPE 列は常に書き出されます。

147 ページの図 37 は FORM オブジェクトについて説明したフィールドが、Query 管理機能 (QMFORM) オブジェクトの作成に使用する外部化書式で表すとどうなるかを示しています。テキスト・フィールドの場合を除いて、指定できるフィールド値は大文字で指定しなければなりません。 図 37 に示してあるデフォルトは、インポート時に適用され、エクスポートされた書式のソース仕様に示されます。テキスト表の場合は次のようになります。

- カウント範囲に示された 0 は、その表がインポート時にソースに存在していなくてもよいことを示します。インポート時に存在していなければ、エクスポートされた書式のソース仕様にも存在しません。
- カウント範囲に示された最大値は、*行* 値として使用できる最大の値です。最小値は 1 です。
- テキスト行番号は、インポートのためには番号順になっている必要はありませんが、エクスポート・ソース内では番号順に示されます。
- 非ブランク・テキストを持つインポート行の最高番号の行の前にギャップがある場合、そのギャップを埋めるためにブランク・テキストを持つ番号付きの行が追加されます。

レコード・タイプ	テーブル番号	フィールド番号	説明	カウント範囲	インポート時のデフォルト
V		1001	オブジェクト・コメント		
			**********		
			* 列 *		
			**********		
T	1110		列のフィールド	1-255	
		1112	--列のデータ・タイプ		-
		1113	--列の見出し		-
		1114	--列の使用目的		-
		1115	--列のインデント		2
		1116	--列の幅		-
		1117	--列の編集		-
		1118	--列の順序		n
			(たとえば、5 番目の R レコードの場合、n は 5)		
			**********		
			* ページ *		
			**********		
V		1201	見出しの前のブランク行の数		0
V		1202	見出しの後のブランク行の数		2
T	1210		ページ見出しテキスト	0-999	
		1212	--ページ見出し行番号		-
		1213	--ページ見出しの位置合わせ		CENTER
		1214	--ページ見出しテキスト		-
V		1301	フッターの前のブランク行数		
V		1302	フッターの後のブランク行数		
T	1310		ページ・フッター・テキスト表	0-999	
		1312	--ページ・フッター行番号		-
		1313	--ページ・フッターの位置合わせ		CENTER
		1314	--ページ・フッター・テキスト		-
			**********		
			* 最終 *		
			**********		
V		1401	最終テキストに対する改ページ		NO
V		1402	最終合計表示行		1
V		1403	最終テキストの前のスキップ数		1
T	1410		最終テキスト	0-999	
		1412	--最終テキスト行番号		-
		1413	--最終テキスト位置合わせ		RIGHT
		1414	--最終テキスト		-

図 37. エンコード (外部化) FORM フィールドの要約 (1/2)

```
                    ************************************************
                    *                    オプション                *
                    ************************************************
V           1501    明細行の間隔                                 1
V           1502    切れ目列の一括表示                           YES
V           1503    省略時の切れ目テキスト                       YES
V           1505    列の折り返し行の同一ページ表示               YES
V           1507    列見出し区切り記号                           YES
V           1508    切れ目合計区切り記号                         YES
V           1510    最終合計区切り記号                           YES

                    ************************************************
                    *                     切れ目                   *
                    ************************************************
V           3080    切れ目レベル標識                             -
V           3101    改ページ後切れ目見出し                       NO
V           3102    列見出しの繰り返し                           NO
V           3103    見出しの前のブランク行の数                   0
V           3104    見出しの後のブランク行の数                   0
T    3110           切れ目見出し表                 0-5
V           3112      --切れ目見出し行番号                       -
V           3113      --切れ目見出しの位置合わせ                 LEFT
V           3114      --切れ目見出しテキスト                     -
V           3201    改ページ後切れ目フッター                     NO
V           3202    切れ目合計表示行                             1
V           3203    フッターの前のブランク行の数                 0
V           3204    フッターの後のブランク行の数                 1
T    3210           切れ目フッター表               0-5
V           3212      --切れ目フッター行番号                     -
V           3213      --切れ目フッター位置合わせ                 RIGHT
V           3214      --切れ目フッター・テキスト                 -
```

図 37. エンコード (外部化) FORM フィールドの要約 (2/2)

149 ページの図 38 は、エクスポートされた Query 管理機能の例です。書式を編集し、その後でインポートすることにより、報告書を所要の形式に設定することができます。レコード・タイプおよびフィールド番号のリストを参照して、フィールド番号を特定の報告書属性に一致させてください。書式の特定部分の意味を説明するために、有効なコメント・レコードである * レコードが使用されています。 この例に示されているのは、R レコード (表) および V レコードを持つ特定の T レコードです。書式の残りの部分の同じタイプのレコードにも、同様の解釈が適用されます。

```
H QM4 01 F 01 E V W E R 01 03 90/3/19 14:27
* 'H' レコードは、上記のように、ファイルの最初のレコードでなければなりません。
* 注記レコードがある場合を除き、列表は 'H' レコードの直後になければ
* なりません。
* T レコードは、R レコードの中で後に続く情報を記述します。
* フィールド番号 '1110' は、表を列表として識別します。
*  |    '005' は、'T' レコードの後に 5 つの列 (R レコード) が続くことを
*  |     |  意味します。
*  |     |   '006' は、T レコードの中にフィールド番号とフィールドの対が
*  |     |    |   6 個あることを意味します。
*  |     |    |   '1112' で始まるのは R レコードの中の値を記述する
*  |     |    |   フィールド番号とフィールドの対です。
*  |     |    |   たとえば、'1112' は 'データ・タイプ'に対応し、
*  |     |    |    |   (フィールド番号の表を参照)
*  |     |    |    |   長さが 8 です。
*  |     |    |    |   列の字下げを変更したい場合は、フィールド番号の
*  |     |    |    |     表を調べれば、列字下げ ID が
*  |     |    |    |     '1115' であることがわかります。
*  |     |    |    |                  '1112' から順にフィールド番号をカウント
*  |     |    |    |                   |    すると、'1115' は一連のフィールドの
*  |     |    |    |                   |    長さの対の 3 番目で、フィールドの幅
*  |     |    |    |                   |    は 6 であることがわかります。
*  |     |    |    |                   |
*  |___  |__  |__  |_____              |_____
*  |  |  |  |  |  |     |             |     |
T 1110 005 006 1112 008 1114 007 1115 006 1116 005 1117 005 1113 040
*                  列字下げの値は、R レコードの 3 番目のフィールドであり、
*                   |    すべての列に 3 が入っています。
*                   |    値は左寄せされ、ブランク区切り文字によって
*                   |    区切られています。
*                   |_
*                   |_|
R CHAR              3      6     C     NAME
R CHAR     BREAK1   3      6     C     DEPARTMENT
R NUMERIC  SUM      3      6     L     YEARS
R NUMERIC  SUM      3      6     L     SALARY
R NUMERIC  SUM      3      6     L     COMMISSION
* 'V' レコードは、内の単一属性を記述します。
* フィールド番号 '1201' は、ページ見出し属性の前のブランク行数です。
*  |
*  |     値の長さは 1 です。
*  |     |  値は 1 です。
*  |     |  |
*  |__   |_ |_
*  |  |  |  ||_|
V 1201 001 1
V 1202 001 2
```

図 38. 外部化書式の例 (1/4)

特定の Query オブジェクトの形式

```
* 次の表は、ページ見出しテキストの説明です。
* 使用されるフィールドは、それぞれ、行番号、位置合わせ、テキストです。
* これは、ページ見出しテキストです。
T 1210 005 003 1212 004 1213 006 1214 055
R 1    CENTER ********************************************
R 2    CENTER **** &ID                          &DATE    ****
R 3    CENTER ****           COMPANY REPORT              ****
R 4    CENTER ****                                       ****
R 5    CENTER ********************************************
*     数値の位置に、代わりに '*' を入れると、レコードの残りの部分を
*     |  データ値の長さとして使用することになります。
*     |
*     |_|
V 1301 *   1
V 1302 *   2
T 1310 003 003 1312 004 1313 006 1314 055
* これは、ページ・フッター・テキストです。
R 1    CENTER XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
R 2    CENTER XXXXXXXX Internal Use Only XXXXXXXX
R 3    CENTER XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
V 1401 003 YES
V 1402 001 2
T 1410 005 003 1412 004 1413 006 1414 055
R 1    LEFT   ************************************************
R 2    LEFT   ******* &PAGE  &TIME  &DATE             ********
R 3    LEFT   *******          END OF REPORT           ********
R 4    LEFT   *******                                  ********
R 5    LEFT   ************************************************
V 1501 *   1
V 1502 003 YES
V 1503 003 YES
V 1505 003 YES
V 1507 003 YES
V 1508 003 YES
V 1510 003 YES
* 次のセクションは、新しい形式を使用する切れ目情報を示して
* います。
*          '3080' V レコードの値は、次の '3080' V レコード
*           |  が見つかるまでの切れ目情報のすべてに適用
*           |  される切れ目レベルを示します。
*         |_|  この例では、切れ目レベルは 1 です。
V 3080 001 1
V 3101 002 NO
V 3102 002 NO
V 3103 001 0
V 3104 001 0
T 3110 001 003 3112 004 3113 006 3114 055
R  1   CENTER BREAK 1 HEADING
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
T 3210 003 003 3212 004 3213 006 3214 055
```

図 38. 外部化書式の例 (2/4)

```
R  1    CENTER ****************************************
R  2    CENTER **** BREAK 1 FOOTING Employee ID=&ID   ****
R  3    CENTER ****************************************
* 切れ目レベル 2 の情報
V 3080 001 2
V 3101 003 YES
V 3102 003 YES
V 3103 001 0
V 3104 001 0
T 3110 002 003 3112 004 3113 006 3114 055
R 1    CENTER BREAK 2 HEADING
R 2    CENTER ---------------
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
T 3210 003 003 3212 004 3213 006 3214 055
R 1    CENTER ****************************************
R 2    CENTER **** BREAK 2 FOOTING    Employee ID=&1 ****
R 3    CENTER ****************************************
* 切れ目レベル 3 の情報
V 3080 001 3
V 3101 003 YES
V 3102 003 YES
V 3103 001 0
V 3104 001 0
T 3110 002 003 3112 004 3113 006 3114 055
R 1    CENTER BREAK 3 HEADING
R 2    CENTER ---------------
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
T 3210 003 003 3212 004 3213 006 3214 055
R 1    CENTER ****************************
R 2    CENTER **** BREAK 3 FOOTING    ****
R 3    CENTER ****************************
* 切れ目レベル 4 の情報
V 3080 001 4
V 3101 003 YES
V 3102 003 YES
V 3103 001 0
V 3104 001 0
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
T 3210 003 003 3212 004 3213 006 3214 055
```

図 38. 外部化書式の例 (3/4)

```
R 1    CENTER ****************************
R 2    CENTER **** BREAK 4 FOOTING    ****
R 3    CENTER ****************************
* 切れ目レベル 5 の情報
V 3080 001 5
V 3101 003 YES
V 3102 003 YES
V 3103 001 0
V 3104 001 0
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
* 切れ目レベル 6 の情報
V 3080 001 6
V 3101 003 YES
V 3102 003 YES
V 3103 001 0
V 3104 001 0
V 3201 002 NO
V 3202 002 NO
V 3203 001 0
V 3204 001 0
T 3210 003 003 3212 004 3213 006 3214 055
R 1    CENTER ++++++++++++++++++++++++
R 2    CENTER +++ BREAK 6 FOOTING +++
R 3    CENTER ++++++++++++++++++++++++
* 'E' レコードは、ファイルの最終レコードです。
* 'E' レコードの後のレコードは無視されます。
E
```

図 38. 外部化書式の例 (4/4)

外部化書式オブジェクトを編集して、報告書の形式を変更することができます。外部化書式のレイアウトは、エンコードされた形式であり、レコード・タイプおよびフィールド番号 ID を使用して書式を表します。外部化書式オブジェクトの各フィールド番号 ID は、それぞれ報告書中の別の属性を表します。外部化書式を変更した後で、それをインポートして変更内容を有効にする必要があります。

153 ページの図 39 は、エンコードされた書式フィールドの記述名を示しています。

レコード タイプ	表 番号	フィールド 番号	説明
V		1001	オブジェクト・コメント
			************************************
			*** 報告書の列セクション ***
			************************************
T	1110		列フィールド
		1112	--列のデータ・タイプ
		1113	--列見出し
		1114	--列の使用目的
		1115	--列の字下げ
		1116	--列の幅
		1117	--列の編集
		1118	--列の順序
			************************************
			*** 報告書のページ・セクション ***
			************************************
V		1201	見出しの前のブランク行数
V		1202	見出しの後のブランク行数
T	1210		ページ見出しテキスト表
		1212	--ページ見出し行番号
		1213	--ページ見出しの位置合わせ
		1214	--ページ見出しテキスト
V		1301	フッターの前のブランク行数
V		1302	フッターの後のブランク行数
T	1310		ページ・フッター・テキスト表
		1312	--ページ・フッター行番号
		1313	--ページ・フッター位置合わせ
		1314	--ページ・フッター・テキスト

図 39. エンコードされた形式の書式フィールドの記述名 (1/2)

```
                        ************************************
                        ***  報告書の最終セクション        ***
                        ************************************
V                1401   最終テキストに対する改ページ
V                1402   最終合計行表示
V                1403   最終テキスト前のスキップ行数


T      1410              最終テキスト表
                 1412           --最終テキスト行番号
                 1413           --最終テキスト位置合わせ
                 1414           --最終テキスト

                        ********************************************
                        **報告書のオプション・フィールド・セクション**
                        ********************************************
V                1501   明細行の間隔
V                1502   切れ目列の一括表示
V                1503   省略時の切れ目テキスト
V                1505   列の折り返し行の同一ページ表示
V                1507   列見出し区切り記号
V                1508   切れ目合計区切り記号
V                1510   最終合計区切り記号
```

図 39. エンコードされた形式の書式フィールドの記述名 (2/2)

エンコードされたオブジェクトの中に、切れ目情報のための新しい形式が入っています。元の形式を使用する書式をサポートするために、Query 管理機能は、切れ目情報を記述する元の形式と新しい形式の両方をサポートしています。 2 つの形式を組み合わせて使用することは許されないため、それを試みた場合は、インポート要求は終了します。すべての書式は、新しい形式を使用してエクスポートされます。

図 40 は、切れ目レベルを示すために切れ目レベル標識 (フィールド番号 3080 の V レコード) を準備する新しい形式を説明しています。各切れ目レベル標識に続くすべての切れ目情報が、 3080 の V レコード内の切れ目レベルの値に適用されます。

新しい形式では、1 組のフィールド番号を使用して、切れ目見出しおよびフッター情報が記述されます。これによって、将来、多数の切れ目レベルがサポートされるような拡張をより効率的に行うことができます。

レコード タイプ	表 番号	フィールド 番号	説明
			************************************
			報告書の切れ目フィールド・セクション
			************************************
V		3080	切れ目レベル標識
V		3101	改ページ後の切れ目見出し
V		3102	列見出しの繰り返し
V		3103	見出しの前のブランク行数
V		3104	見出しの後のブランク行数
T	3110		切れ目見出し表
V		3112	--切れ目見出し行番号
V		3113	--切れ目見出しの位置合わせ
V		3114	--切れ目見出しテキスト
V		3201	改ページ後切れ目フッター
V		3202	切れ目合計表示行
V		3203	フッターの前のブランク行数
V		3204	フッターの後のブランク行数
T	3210		切れ目フッター表
V		3212	--切れ目フッター行番号
V		3213	--切れ目フッター位置合わせ
V		3214	--切れ目フッター・テキスト

図 40. エンコードされた切れ目情報に望ましい形式

156 ページの図 41 は、エンコードされたオブジェクト内の切れ目情報を表すための元の形式を説明しています。この形式では、切れ目情報の各属性ごとに固有のフィールド番号が使用されます。この形式を新しい形式と組み合わせて使用することはできません。

特定の Query オブジェクトの形式

レコード タイプ	表 番号	フィールド 番号	説明
			************************************
			報告書の切れ目フィールド・セクション
			************************************
V		1601	切れ目 1: 改ページ後見出し
V		1602	切れ目 1: 列見出しの繰り返し
V		1603	切れ目 1: 見出しの前のブランク行数
V		1604	切れ目 1: 見出しの後のブランク行数
T	1610		切れ目 1: 見出し表
V		1612	--切れ目 1: 見出し行番号
V		1613	--切れ目 1: 見出しの位置合わせ
V		1614	--切れ目 1: 見出しテキスト
V		1701	切れ目 1: 改ページ後切れ目フッター
V		1702	切れ目 1: 切れ目合計表示行
V		1703	切れ目 1: フッターの前のブランク行数
V		1704	切れ目 1: フッターの後のブランク行数
T	1710		切れ目 1: フッター表
V		1712	--切れ目 1: フッター行番号
V		1713	--切れ目 1: フッターの位置合わせ
V		1714	--切れ目 1: フッター・テキスト
V		1801	切れ目 2: 改ページ後見出し
V		1802	切れ目 2: 列見出しの繰り返し
V		1803	切れ目 2: 見出しの前のブランク行数
V		1804	切れ目 2: 見出しの後のブランク行数
T	1810		切れ目 2: 見出し表
V		1812	--切れ目 2: 見出し行番号
V		1813	--切れ目 2: 見出しの位置合わせ
V		1814	--切れ目 2: 見出しテキスト
V		1901	切れ目 2: 改ページ後切れ目フッター
V		1902	切れ目 2: 切れ目合計表示行
V		1903	切れ目 2: フッターの前のブランク行数
V		1904	切れ目 2: フッターの後のブランク行数
T	1910		切れ目 2: フッター表
V		1912	--切れ目 2: フッター行番号
V		1913	--切れ目 2: フッターの位置合わせ
V		1914	--切れ目 2: フッター・テキスト
V		2001	切れ目 3: 改ページ後見出し
V		2002	切れ目 3: 列見出しの繰り返し
V		2003	切れ目 3: 見出しの前のブランク行数
V		2004	切れ目 3: 見出しの後のブランク行数
T	2010		切れ目 3: 見出し表
V		2012	--切れ目 3: 見出し行番号
V		2013	--切れ目 3: 見出しの位置合わせ
V		2014	--切れ目 3: 見出しテキスト

図 41. エンコードされた制御の切れ目情報の元の形式 (1/3)

```
V                 2101     切れ目 3: 改ページ後切れ目フッター
V                 2102     切れ目 3: 切れ目合計表示行
V                 2103     切れ目 3: フッターの前のブランク行数
V                 2104     切れ目 3: フッターの後のブランク行数
T        2110               切れ目 3: フッター表
V                 2112     --切れ目 3: フッター行番号
V                 2113     --切れ目 3: フッターの位置合わせ
V                 2114     --切れ目 3: フッター・テキスト

V                 2201     切れ目 4: 改ページ後見出し
V                 2202     切れ目 4: 列見出しの繰り返し
V                 2203     切れ目 4: 見出しの前のブランク行数
V                 2204     切れ目 4: 見出しの後のブランク行数
T        2210               切れ目 4: 見出し表
V                 2212     --切れ目 4: 見出し行番号
V                 2213     --切れ目 4: 見出しの位置合わせ
V                 2214     --切れ目 4: 見出しテキスト

V                 2301     切れ目 4: 改ページ後切れ目フッター
V                 2302     切れ目 4: 切れ目合計表示行
V                 2303     切れ目 4: フッターの前のブランク行数
V                 2304     切れ目 4: フッターの後のブランク行数
T        2310               切れ目 4: フッター表
V                 2312     --切れ目 4: フッター行番号
V                 2313     --切れ目 4: フッターの位置合わせ
V                 2314     --切れ目 4: フッター・テキスト

V                 2401     切れ目 5: 改ページ後見出し
V                 2402     切れ目 5: 列見出しの繰り返し
V                 2403     切れ目 5: 見出しの前のブランク行数
V                 2404     切れ目 5: 見出しの後のブランク行数
T        2410               切れ目 5: 見出し表
V                 2412     --切れ目 5: 見出し行番号
V                 2413     --切れ目 5: 見出しの位置合わせ
V                 2414     --切れ目 5: 見出しテキスト

V                 2501     切れ目 5: 改ページ後切れ目フッター
V                 2502     切れ目 5: 切れ目合計表示行
V                 2503     切れ目 5: フッターの前のブランク行数
V                 2504     切れ目 5: フッターの後のブランク行数
T        2510               切れ目 5: フッター表
V                 2512     --切れ目 5: フッター行番号
V                 2513     --切れ目 5: フッターの位置合わせ
V                 2514     --切れ目 5: フッター・テキスト
```

図 41. エンコードされた制御の切れ目情報の元の形式 (2/3)

特定の Query オブジェクトの形式

```
V                   2601     切れ目 6: 改ページ後見出し
V                   2602     切れ目 6: 列見出しの繰り返し
V                   2603     切れ目 6: 見出しの前のブランク行数
V                   2604     切れ目 6: 見出しの後のブランク行数
T         2610              切れ目 6: 見出し表
V                   2612     --切れ目 6: 見出し行番号
V                   2613     --切れ目 6: 見出しの位置合わせ
V                   2614     --切れ目 6: 見出しテキスト

V                   2701     切れ目 6: 改ページ後切れ目フッター
V                   2702     切れ目 6: 切れ目合計表示行
V                   2703     切れ目 6: フッターの前のブランク行数
V                   2704     切れ目 6: フッターの後のブランク行数
T         2710              切れ目 6: フッター表
V                   2712     --切れ目 6: フッター行番号
V                   2713     --切れ目 6: フッターの位置合わせ
V                   2714     --切れ目 6: フッター・テキスト
```

図 41. エンコードされた制御の切れ目情報の元の形式 (3/3)

Query 管理機能の外部化 PROC 形式および QUERY 形式

PROC オブジェクトおよび SQL QUERY オブジェクトは、 130 ページの『Query 管理機能のパネル形式』で説明したパネル形式で外部化されます (エクスポートされて保管されます)。

Query 管理機能のソート・シーケンスについての IMPORT Query の考慮事項

IMPORT QUERY を使用して SQL Query オブジェクトを作成する場合、Query 管理機能はソート・シーケンス・オプションと言語 ID をサポートします。

ソート・シーケンスと言語 ID は、IMPORT QUERY CPI コマンドおよび Query のソース仕様に指定することができます。 IMPORT QUERY CPI コマンドに指定した値は、Query ソース仕様に指定した値に優先します。

IMPORT コマンドには、SRTSEQ と LANGID の 2 つの別個のオプションがあります。これらは、Query ソース仕様では別個の V レコードです。 Query 管理機能を使用すると、コマンドで 1 つの属性を指定し、ソース仕様で 1 つの属性を指定することができます。

Query 管理機能のエラー処理と警告条件

他のシステムから外部化された Query をインポートできるようにするため、 Query 管理機能では、ソース仕様の不一致を許容できるようになっています。 LANGID または SRTSEQ がソース・メンバーに指定されている場合、以下の V レコード形式エラーが起きる可能性があります。

- 値の長さの指定がゼロか、または指定がない
- 指定された長さが、データの値よりも短い
- 指定された長さが、データの値よりも長い
- LANGID と SRTSEQ に対して、認識できない特殊な値がソース仕様に指定されている

より多くの特殊値をサポートする高水準システムから Query をエクスポートすると、次の V レコード形式エラーが起こる可能性があります。

- 言語 ID がサポートされていない

より多くの言語 ID をサポートするシステムから Query をエクスポートすると、次のような V レコード形式エラーが起こる可能性があります。

- 変換テーブル名または言語 ID 名の形式が正しくない

Query 管理機能での失敗条件

V レコード形式エラーとしてフラグが付けられるエラーはいずれも、それがコマンドにある場合は、失敗する条件になります。たとえば、ステートメント IMPORT QUERY FROM Y (SRTSEQ=*INVALID は、次の理由で失敗します。

- 変換テーブルが見つからない
- 変換テーブル名が修飾されているとき、そのライブラリーが見つからない
- 指定の変換テーブルを使用する権限が与えられていない
- 変換テーブルの入っているライブラリーに対する権限が与えられていない

Query 管理機能のソート・シーケンスについての EXPORT QUERY の考慮事項

EXPORT QUERY CPI コマンドには変更ありません。ただし、EXPORT QUERY の処理は変更されています。新しく 2 つの V レコードが、Query ソース・ファイル・メンバーにエクスポートされます。 V レコード・タイプ 5001 は、ソート・シーケンス・オプションです。 V レコード・タイプ 5002 は、言語 ID です。

```
V 5001 010 *JOB
V 5002 003 ENG
```

新しい V レコードは、コメント / 説明用の V レコード 1001 の直後に、数値順に入れられます。エクスポートされる情報は、Query を作成するために使用する情報です。ユーザーが指定したソート・シーケンス表がまだ存在しているかどうかの検査は、行われません。

Query for iSeries 用 *QRYDFN オブジェクトを SQL ステートメントに変換するためにエクスポートを行うと、Query 管理機能は、そのソート・シーケンスと言語 ID である V レコードをソース・ファイル・メンバーにエクスポートします。 *QRYDFN に定義された SRTSEQ と LANGID は、次のオプションとしてソース・ファイルにエクスポートされます。

1= 16 進

4= 変換テーブル

5= システム・ソート・シーケンス

次のいずれかのオプションを使用する *QRYDFN オブジェクトが定義されると、

2= Query for iSeries 用言語

3= シーケンスの定義

次のようになります。

- ソート・シーケンスとしては、*HEX が常にエクスポートされます。
- LANGID V レコードはエクスポートされません。

Query 管理機能の外部化 Query の説明

Query 管理機能では、Query の中にソート・シーケンスと言語 ID を指定することができます。 図 42 は、Query において、H レコードの後に入れることのできる V レコードのリストです。

レコード・タイプ	テーブル番号	フィールド番号	説明	カウント範囲	インポート時のデフォルト
V		1001	オブジェクト・コメント		
V		5001	ソート・シーケンス・オプション *JOBRUN *JOB *HEX *LANGIDSHR *LANGIDUNQ *LIBL/表名 *CURLIB/表名 ライブラリー名/表名		*JOBRUN
V		5002	言語 ID *JOBRUN *JOB 言語 ID		*JOBRUN

図 42. 外部化 Query フィールドの要約

Query 管理機能がこれらのレコードを正しく V レコードとして解釈するためには、 H レコードが該当のソース・メンバーの最初のレコードでなければなりません。 V レコードは、H レコードの直後になければなりません。その間に、ブランクのレコードや V レコード以外のレコードがあってはなりません。 V レコードの順序は、自由です。エラーでない限り、各タイプの最後のレコードが使用されます。最後のレコードがエラーの場合は、そのタイプの中で、前にある有効な V レコードが使用されます。

有効なソート・シーケンス・オプションの V レコードが見つからない場合、デフォルト設定が使用されます。また、有効な言語 ID の V レコードが見つからない場合も、デフォルトが使用されます。有効でない V レコードのそれぞれに対して、ジョブ・ログに V レコード警告が送られ、インポートは警告を伴いますが完了します。

IMPORT QUERY コマンドに SRTSEQ または LANGID パラメーターを指定する場合は、そのコマンドの値がソース仕様の値に優先します。 SRTSEQ または LANGID が IMPORT QUERY CPI コマンドのオプションとして指定されている場合、次の事項があてはまります。

- ソース・メンバーの対応するオプションの V レコードは無視される。
- ソース・メンバーの対応するオプションの V レコードは検査されない。

コマンドやソース仕様で SRTSEQ または LANGID パラメーターを指定しない場合、パラメーター SRTSEQ と LANGID はデフォルトの *JOBRUN になります。

第 9 章 Query 管理機能での分散リレーショナル・データベース・アーキテクチャー (DRDA)

Query 管理機能の DRDA 機能を使用すると、アプリケーションは、複数のリモート・データベースにアクセスして、同期をとってそれらのデータベースに対しコミットを行い、ロールバックすることができます。

Query 管理機能は、次の 2 つのタイプの接続管理方式をサポートしています。

- リモート作業単位 (RUW)
- 分散作業単位 (DUW)

複数のデータベース接続を選択するには、DUW 接続管理を使用します。 1つのデータベースに接続することを選択する場合は、RUW 接続管理を使用します。 Query 管理機能の START コマンドで DSQRDBCNNMTH キーワードを使用することによって、使用する接続管理の方式を選ぶことができます。また、DSQRDBCNNMTH は、 START コマンドの DSQSCMD キーワードで指定される Query コマンド・プロシージャーの中でもセットすることができます。 DSQRDBCNNMTH キーワードについての説明は、 47 ページの『Query 管理機能での START』を参照してください。接続管理方式は、また、 RDBCNNMTH パラメーターを使用して、 CL コマンドの STRQMQRY と STRQMPRC で指定することもできます。

リモート・データベース名またはローカル・データベース名は、 CONNECT および SET CONNECTION コマンドを使用して、または、 START コマンドの DSQSDBNM キーワードを使用して、指定することができます。 DSQSDBNM キーワードの詳細については、 47 ページの『Query 管理機能での START』を参照してください。 CONNECT コマンドおよび SET CONNECTION コマンドについての詳細は、 24 ページの『Query 管理機能での CONNECT』および 45 ページの『Query 管理機能での SET CONNECTION』を参照してください。

CONNECT コマンドおよび SET CONNECTION コマンドを使用するか、または、START コマンドの DSQDBNM キーワードを使用して、アプリケーションは指定のデータベースに接続されます。接続情報は Query インスタンスに関連づけられます。 START コマンドまたは CONNECT コマンドを用いてリモート・データベースが指定されていない場合には、 START コマンドの実行時の接続情報には現行サーバーが示されます。

注: RUN QUERY、ERASE TABLE、および SAVE DATA AS コマンドはすべて、この接続に対して行われます。現在の接続が Query インスタンスに関連した接続と同じでない場合には、 RUN QUERY、ERASE TABLE、または SAVE DATA AS の各コマンドは正常に実行されません。

Query 管理機能 DRDA のリモート作業単位 (RUW)

RUW を使用した場合は、現在の Query 管理機能がそのリモート・データベースに接続しているのと同じになります。その接続管理は、現在行われているのと同じです。 RUW では、1 つのリレーショナル・データベースに対して行える接続は 1 つだけです。コミットとロールバックは、その 1 つの接続に適用されます。

Query 管理機能 DRDA の分散作業単位 (DUW)

これは、Query 管理機能のデフォルトの接続管理方式です。 DUW 接続管理方式は、RUW 接続管理方式よりもはるかに強力です。 DUW では、複数のリレーショナル・データベース接続を維持することができます。コミットとロールバックは、同期化された方式で複数のシステムに対して行われます。

Query 管理機能 DRDA の接続管理のステートメント

Query 管理機能は、次の接続ステートメントをサポートしています。

- CONNECT
- COMMIT
- DISCONNECT
- RELEASE
- SET CONNECTION

Query 管理機能プログラムでは、COMMIT コマンドだけが使用できます。これらのコマンドについては、21 ページの 『第 4 章 Query 管理機能でのコマンド』に詳しい説明があります。

Query 管理機能 DRDA の接続管理

CONNECT および RELEASE ステートメントは、接続の保留または解放の状態を制御するものです。解放状態とは、次のコミット操作が正常に行われた時点で、その接続が切断される条件を指します。解放状態は、保留されている切断と考えることができます。ロールバックは、接続にはどのような影響も与えません。保留状態は、次のコミット操作で接続が切断されないことを意味します。接続は、CONNECT ステートメントによって保留状態に置かれます。 RELEASE ステートメントによって、接続は、保留状態から解放状態に移ります。解放状態にある接続を保留状態に戻したり、置いたりすることはできません。すなわち、ロールバックが出された時、またはコミットの結果がロールバックになった時には、接続は、作業単位の境界を超えて、解放状態のままになっています。

接続の状態が保留であるか解放であるかに関係なく、接続は、現行状態または休止状態にもなっています。*現行状態* とは、この状態の間に実行される SQL ステートメントのために、この接続が使用されるような接続の状態を指します。*休止状態* とは、接続が中断されている状態を指します。接続が休止状態にあるときは、SQL ステートメントは、コミットとロールバックの場合にのみその接続を使用することができます。 SET CONNECTION と CONNECT ステートメントを使用すると、指名のリレーショナル・データベースに対する既存の接続が休止状態に置かれているか、または休止状態のままになっている時に、その接続を現行状態に変更します。一時点で、ただ 1 つの接続だけが現行状態になっていることができます。休止接続が同じ作業単位で現行状態になると、すべてのロック、カーソル、および準備されたステートメントは、その接続が現行状態であった時に最後に使用された状態に復元されます。

DISCONNECT ステートメントは、指定された接続を切断します。一度リレーショナル・データベースへの接続が切断されると、 SQL ステートメントでそのリレーショナル・データベースを対象とする必要が出てきた時は、アプリケーションはそのリレーショナル・データベースに再度接続しなければなりません。保護会話には、RELEASE ステートメントを使用しなければなりません。

Query 管理機能 DRDA の会話タイプ

保護

保護会話は、リモート・システムのリレーショナル・データベースへの接続に使用されます。保護会話では、2 フェーズのコミット・プロトコルが使用され、万一障害が起きても、リモート・システムで行われた更新は、他のリモートまたはローカル・リソースに対する更新と同期化されます。

無保護
無保護会話は、リモート・システムのリレーショナル・データベースへの接続に使用されます。この場合は、障害が起きると、そのリモート・システムで行われた更新は他のリモート・リソースまたはローカル・リソースに対する更新と同期化されません。

ローカル
会話は使用されません。この接続は、ローカル・リレーショナル・データベースに対するものです。 2 フェーズのコミット・プロトコルが使用され、万一障害が起きても、ローカル・システムで行われた更新は他のリモート・リソースまたはローカル・リソースに対する更新と同期化されます。

***ARDPGM**
接続は、アプリケーション・リクエスター・ドライバー (ARD) プロシージャーがアクセスするリレーショナル・データベースに対するものになります。リレーショナル・データベースを対象とする SQL 要求は、リレーショナル・データベース・ディレクトリー・レベルで指定される ARD プログラムで処理されます。

Query 管理機能 DRDA の読み取り専用接続

YES
接続は読み取り専用です。コミットメント制御のもとで実行すると、この接続では更新を行うことはできません。

NO
接続は読み取り専用ではありません。この接続では更新を行うことができます。この接続で更新を行う場合、接続の会話タイプがローカルまたは保護であると、この作業単位では、読み取り専用ではなく、かつ、会話タイプがローカルまたは保護であるその他のすべての接続に対しても、更新ができます。そうでない場合、すなわち、この接続で更新を行う場合、その会話タイプが無保護であると、この作業単位ではこの接続に対してのみ、更新を行うことができます。

Query 管理機能 DRDA の状況

HLD
保留状態は、次のコミット操作で接続が切断されないことを意味します。接続は、CONNECT ステートメントによって保留状態に置かれます。

RLS
解放状態は、次の正常なコミット操作で、その接続が切断されることを意味します (ロールバックは接続に影響しません)。接続は、RELEASE ステートメントによって、保留状態から解放状態になります。解放状態にある接続を、保留状態にすることはできません。

Query 管理機能 DRDA の接続管理方式の考慮事項

接続管理方式は、Query 管理機能の CONNECT コマンドの持つ語義に関係し、前の接続の切断に影響を与えます。 RUW の接続管理方式の場合に CONNECT を出すと、RUW は新しい接続を設定する前に、前の接続 (1 つまたは複数) を切断します。一方、DUW 接続管理方式の場合に CONNECT を出すと、前の接続はいずれも切断されません。

Query 管理機能 DRDA の START コマンドでの DSQRDBCNNMTH キーワードの使用

このキーワードは、Query 管理機能で使用されている接続管理方式を指定します。

DSQRDBCNNMTH の有効なオプションは、次のとおりです。

***DUW**

複数のリレーショナル・データベースに接続できます。これは、この変数のデフォルトです。別のリレーショナル・データベースに連続して START または CONNECT コマンドを出しても、前の接続の切断にはなりません。ある接続から別の接続への切り替えは、SET CONNECTION で行います。これによって、読み取り専用接続となることがあります。

同じデータベースに対して連続して CONNECT コマンドを出すと、失敗となります。ただし、同じデータベースへの連続の START コマンドは許されます。

***RUW**

リレーショナル・データベースに対する接続が 1 つだけ許されます。別のリレーショナル・データベースに連続して START または CONNECT コマンドを出すと、新しい接続が設定される前に、前の接続がすべて切断されます。同じデータベースに連続して START または CONNECT コマンドを出しても、現行接続は変わりません。

Query 管理機能 DRDA の START コマンドでの DSQSDBNM キーワードの使用

START コマンドで DSQSDBNM キーワードを使用すると、 Query インスタンスはユーザーが指定したリモート・データベースに接続されます。このキーワードは、Query インスタンスの処理中に Query 管理機能によって開始されるすべての SQL 操作の対象となるリモート・データベースを指示します。このキーワードが START コマンドで指定されず、しかも CONNECT コマンドが使用されない場合には、 Query インスタンスと関連づけられた接続が、 START コマンドの実行時に CURRENT SERVER となります。これは、継承接続です。接続情報はこの Query インスタンスに関連づけられます。 RUN QUERY、ERASE TABLE、および SAVE DATA AS の Query 管理機能コマンドは、すべてこの接続に対して行われます。

DSQSDBNM キーワードの有効なオプションは次のとおりです。

***CURRENT**

インスタンスは CURRENT SERVER と関連した接続を継承します。そのリモート・データベース名がリレーショナル・データベース・ディレクトリーにある場合には、 DSQSDBNM キーワードは CURRENT SERVER にセットされます。そのリモート・データベース名がリレーショナル・データベース・ディレクトリーにない場合には、 DSQSDBNM キーワードは *NONE にセットされます。デフォルトは *CURRENT です。

***NONE**

接続はローカル・データベース・マネージャーに対して行われます。ローカル・データベースはリレーショナル・データベース・ディレクトリーの中にある必要はありません。ローカル・データベース名がリレーショナル・データベース・ディレクトリーにある場合には、DSQSDBNM キーワードは該当する名前にセットされます。ローカル・データベース名がリレーショナル・データベース・ディレクトリーにない場合には、DSQSDBNM キーワードは *NONE にセットされます。

rdbname

リモート・データベース名は、 DRDA を用いてアクセスできるデータベースを識別するためのものです。 リモート・データベース名を指定する場合は、 START コマンドの DSQUSER と DSQPASSWORD のキーワードを使用して、リモート・データベースに対するユーザー識別とパスワードを指定することができます。

Query 管理機能の DRDA と活動化グループ

それぞれの Query インスタンスは、関連のデータベース接続と、関連の活動化グループとを持っています。各活動化グループは、それに関連する接続を 1 つまたは複数、持つことができます。 Query 管理機能によって、アプリケーションは、その Query インスタンスの関連活動化グループに関連する接続を管理することができます。この接続は、RUW または DUW のいずれかの接続管理方式を使用して、管理するこ

とができます。 RUW 接続管理方式を使用すると接続が 1 つだけ維持され、一方、 DUW 接続管理方式を使用すると複数の接続を維持することができます。

アプリケーション・プログラムは、そのプログラムに関連付けられている活動化グループが Query 管理機能インスタンスにも関連付けられている場合にのみ、その Query 管理機能インスタンスを使用できます。複数の活動化グループが、1 つの Query 管理機能インスタンスを共用することはできません。ある活動化グループにおいて Query 管理機能インスタンスを介して行われる処理は、他の活動化グループでの Query 管理機能インスタンスによって行われる処理に影響を与えることはありません。

Query 管理機能の DRDA と活動化グループについての考慮事項

ILE C/400 を使用して作成したアプリケーション・プログラムだけを、デフォルト以外の活動化グループに関連付けることができます。 DB2 UDB for iSeries Query Manager の機能は、デフォルトの活動化グループに関連付けられます。 Query 管理機能の CL コマンドもすべてデフォルトの活動化グループで実行されます。

CONNECT コマンドを処理するために Query 呼び出し可能プログラミング・インターフェースを呼び出すアプリケーション・プログラムは、呼び出しスタックに残っていなければなりません。それがない場合には、アプリケーション・プログラムが終了した時点で暗黙の切断が起こります。すべての RUN QUERY および ERASE TABLE コマンドは、その Query インスタンスに関連する接続である、現行接続を対象としています。

Query 管理機能 DRDA のデフォルトの活動化グループ

図 43 は、デフォルト活動化グループに関連する 2 つのプログラムの間に起こることを示しています。どちらのプログラムも ILE C/400 プログラムではありません。

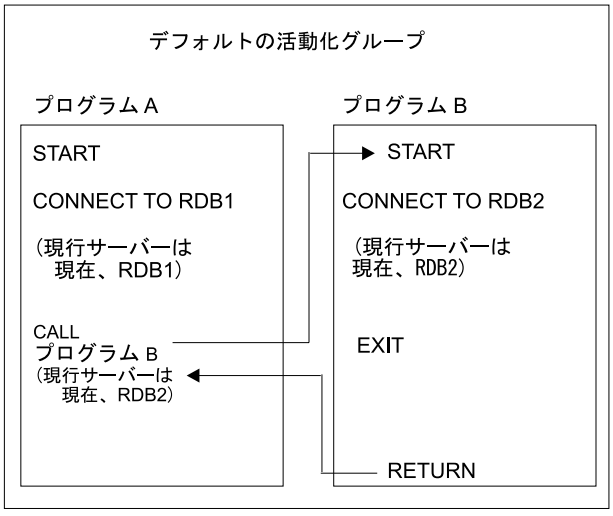


図 43. デフォルト活動化グループで実行しているプログラム

Query 管理機能 DRDA の非デフォルト活動化グループ

図 44 は、同じ活動化グループで実行しているプログラムの対話を示しています。プログラム A とプログラム B は、ともに活動化グループ 2 で実行されます。プログラム B がプログラム A に戻ると、現行サーバーは RDB2 にセットされます。

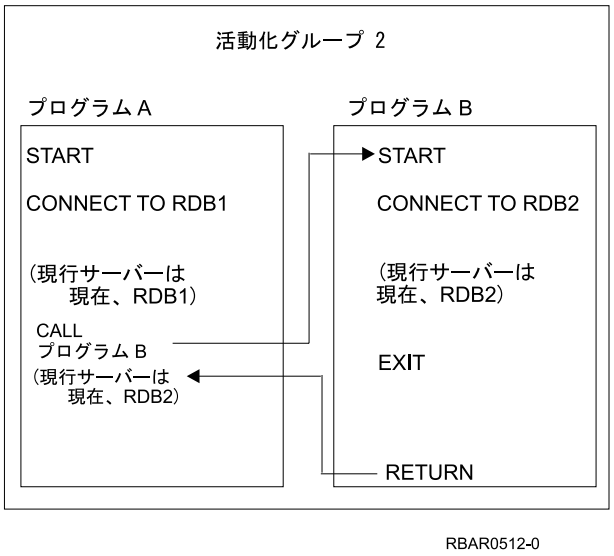


図 44. 同じ活動化グループで実行しているプログラム

Query 管理機能 DRDA のデフォルトおよび非デフォルト活動化グループ

図 45 は、デフォルトおよび非デフォルトの活動化グループで実行しているプログラムの対話を示しています。プログラム A とプログラム B は、異なる活動化グループで実行されます。プログラム B がプログラム A に戻った後も、プログラム A に関連する現行サーバーは、RDB1 のままになっています。

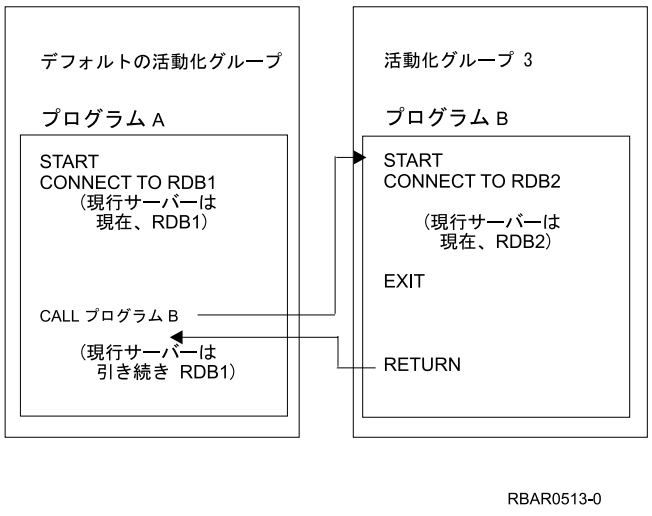


図 45. 異なる活動化グループで実行しているプログラム

Query 管理機能 DRDA の 2 つの非デフォルト活動化グループ

図 46 は、異なる活動化グループで実行されるプログラムの対話を示しています。プログラム A とプログラム B は、異なる活動化グループで実行されます。プログラム B がプログラム A に戻った後も、プログラム A に関連する現行サーバーは、RDB1 のままになっています。

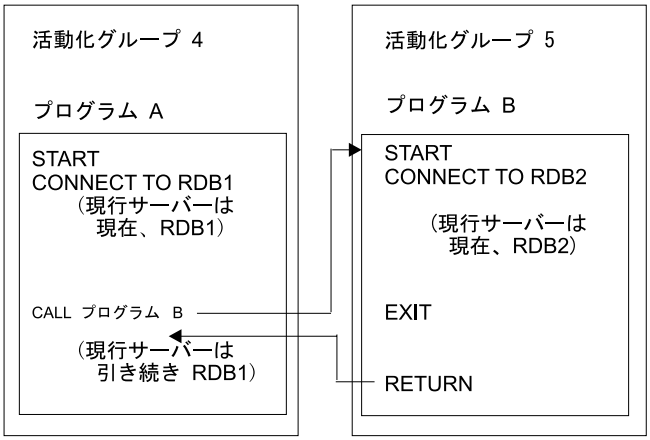


図 46. 異なる非デフォルト活動化グループで実行しているプログラム

Query 管理機能での DRDA に関するコマンドの考慮事項

以下のコマンドには、DRDA で使用される場合に特殊な考慮事項があります。

『Query 管理機能 DRDA の SAVE DATA AS』

『Query 管理機能 DRDA のその他のコマンド』

Query 管理機能 DRDA の SAVE DATA AS

SAVE DATA AS コマンドをリモート接続に使用することはできません。ただし、SAVE DATA AS コマンドの前に、リモート接続をローカル接続に切り換えることによって、リモート Query の結果をローカルに保管することができます。 RUW 接続管理方式では、ローカル接続を設定するために、 CONNECT RESET コマンドを使用できます。また、DUW 接続管理方式では、前にローカル接続がない場合は、CONNECT RESET コマンドを使用してローカル接続を設定することができます。前にローカル接続がある場合は、SET CONNECTION コマンドを使用する必要があります。

Query 管理機能 DRDA のその他のコマンド

IMPORT、EXPORT、PRINT、および RUN PROC コマンドで参照される Query オブジェクト、ソース・ファイル、および印刷ファイルは、接続がリモートであるかどうかに関係なく、常にローカルで検索されます。 IMPORT、EXPORT、PRINT、 RUN PROC、GET、および SET コマンドを実行する場合、アプリケーション・プロセスは接続可能な状態である必要はありません。

Query 管理機能 DRDA のコミットメント制御

コミットメント制御とは、処理しているデータベースに対して行うことができる更新のレベルのことです。Query 管理機能セッションに持たせたいコミットメント制御の種類を指定することができます。

コミットメント制御は、START コマンドの DSQCMTLV キーワードを使用して指定します。デフォルトは NONE です。このキーワードに NONE 以外の値を設定すると、Query 管理機能において、コミットメント制御のもとで (RUN QUERY CPI コマンドを使用して) すべてのセッション SQL ステートメントを実行することができます。次に、Query 管理機能の COMMIT コマンドまたは COMMIT と ROLLBACK の SQL ステートメントを実行できます。

注: SAVE DATA AS コマンドは、コミットメント制御で実行することはできません。

DSQCMTLV キーワードには次の値を指定することができます。

NONE
コミットメント制御を使用しないことを指示します。これは *NONE 分離レベルと同じです。

UR
更新された行だけがトランザクションの終わりまでロックされることを指定します。これは *CHG 分離レベルと同じです。

CS
カーソルが位置づけられている行は、カーソル位置が変更されるまでロックされることを指定します。更新された行は、トランザクションの終わりまでロックされます。これは *CS 分離レベルと同じです。

RS
選択または更新されたすべての行がトランザクションの終わりまでロックされることを指定します。これは *ALL 分離レベルと同じです。

RR
選択または更新されたすべての行が作業単位 (UOW) の終わりまでロックされることを指定します。これにより、作業単位 (UOW) の中で読み取りを繰り返すと、必ず同じ結果が戻されます。これは、遂次化可能分離レベルと同じです。

START コマンドの DSQCMTLV 用にセットされるキーワードは、 Query コマンド・プロシージャーによって DSQCMTLV 変数用にセットされたキーワードの値を一時変更します。デフォルトの活動化グループを必ず使用する場合は、コミットメント制御処理は、ジョブ・レベルのコミット定義を使用し、バージョン 2 リリース 3 モディフィケーション 0 より前の OS/400 の場合と同じように機能します。

Query 管理機能 DRDA での ILE C/400 の考慮事項

Query 管理機能をコミットメントの制御下で実行する場合は、コミット定義が使用されます。 1 つのプロセスで、複数のコミット定義を開始させることができます。コミット定義は、ジョブまたは特定の活動化グループのいずれかに関連付けられます。デフォルトの活動化グループに関連するアプリケーション・プログラムは、ジョブ・レベルのコミット定義を使用します。

Query 管理機能の COMMIT コマンドまたは COMMIT SQL ステートメントをアプリケーション・プログラムで実行するときは、次のようになります。

- その活動化グループ・レベルのコミット定義に関連する処理だけがコミットされます。
- コミットメント制御下のアプリケーション・プログラムで行われる処理は、すべてコミットされます。

- 同じ活動化グループのアプリケーション・プログラムで実行される処理は、コミットされます。別の活動化グループに関連するアプリケーション・プログラムで行われる処理は、コミットされません。
- 解放状態になっている接続は、すべて切断されます。

Query 管理機能 DRDA の非デフォルト活動化グループのコミットメント制御について

次の場合には、Query インスタンスが暗黙に開始すると、活動化グループ・レベルのコミット定義が始まります。

- アプリケーション・プログラムがコミットメント制御を使用して Query インスタンスを開始している
- ジョブ・レベルのコミット定義が開始されていない

次のすべてに該当する場合は、Query インスタンスを暗黙に開始すると、ジョブ・レベルのコミットメント定義が使用されます。

- アプリケーション・プログラムが非デフォルト活動化グループに関連付けられている
- アプリケーション・プログラムがコミットメント制御を使用して Query インスタンスを開始している
- ジョブ・レベルのコミット定義は開始されている
- 活動化グループ・レベルのコミット定義は開始されていない

アプリケーション・プログラムにおける Query 管理機能の COMMIT コマンドまたは COMMIT SQL ステートメントの実行は、活動化グループ・レベルのコミット定義に関連する処理をコミットするだけです。デフォルト活動化グループの他のアプリケーション・プログラムで行われる処理は、コミットされます。デフォルト活動化グループのその他のアプリケーション・プログラムで行われる処理は、ジョブ・レベルのコミット定義の開始後に活動化グループのコミット定義が開始した場合にだけ、コミットされます。

デフォルト活動化グループに対するコミットメント制御について

アプリケーション・プログラムがデフォルト活動化グループに関連付けられ、しかも

- アプリケーション・プログラムがコミットメント制御を使用して Query インスタンスを開始している場合には、
- ジョブ・レベルのコミット定義が開始しているかどうかには関係なく、 Query インスタンスの暗黙の開始によって、ジョブ・レベルのコミット定義が使用されます。

アプリケーション・プログラムで Query 管理機能の COMMIT コマンドまたは COMMIT SQL ステートメントを実行すると、そのジョブ・レベルのコミット定義に関連する処理のすべてがコミットされます。デフォルト活動化グループの他のアプリケーション・プログラムで行われる処理は、コミットされます。非デフォルト活動化グループに関連するアプリケーション・プログラムで行われる処理は、コミットされません。

コミットメント制御と活動化グループについてさらに詳しくは、*バックアップおよび回復の手引き* (SD88-5008) を参照してください。

Query 管理機能 DRDA でのリモート処理と長い列名

バージョン 3 リリース 1 より前のリリースのリモート iSeries システムの表から、10 文字を超える列名を選択しようとする Query の実行は失敗します。「SQL0107 - &1 too long Maximum 10 characters」のエラーが出ます。

iSeries 以外のシステムに対して 10 文字を超える列名を選択する Query を実行した時は、列名全体が報告書に表示され、データの出力ファイルへの保管においてもそのまま保持されます。データを出力ファイルに保管する時は、システム列名としては、名前の最初の 10 文字が使用されます。

第 10 章 Query 管理機能でのコード化文字セット ID (CCSID)

CCSID は、エンコード方式、および、文字セットとコード・ページの 1 対または複数の対を固有のものとして識別する、2 バイトの (無符号) 整数です。 CCSID が付加されたデータは、同じ文字セットを使用する異なる言語において、データが同じに見えるように変換することができます。コード・ページが異なる場合には、変換しないとデータが同じに見えない場合があります。これは、ある言語の文字が別の言語では別の文字のように見える 16 進値がデータに含まれていることがあるためです。

Query 管理機能での CCSID のインポート処理

Query 管理機能プログラムまたは書式がインポートされる場合には、インポートするファイルの CCSID を用いてマーク付けされ、CCSID 変換は行われません。 Query 管理機能プロシージャーがインポートされ、インポート先のファイルが存在していない場合には、ジョブの CCSID を用いてファイルが作成されます。ファイルが存在しており、そのファイルの CCSID がインポート元の CCSID と一致しない場合には、プロシージャーはインポート先のファイルの CCSID に変換されます。

Query 管理機能での CCSID のエクスポート処理

Query 管理機能プログラム、書式、またはプロシージャーのエクスポート時に、該当するソース・ファイルが存在していない場合には、ジョブの CCSID を用いてソース・ファイルが作成されます。 Query 管理機能プログラム、書式、またはプロシージャーが異なる CCSID を持っている場合には、ソース・ファイルはエクスポート先のソース・ファイルの CCSID に変換されます。

Query 管理機能での CCSID の印刷処理

Query 管理機能プログラム、書式、またはプロシージャーが印刷される場合には、それぞれがジョブの CCSID に変換されます。 MIN、MAX、あるいは BREAKn を使用する書式から報告書を印刷する場合、その報告書をフォーマット設定するために、Query 実行に使用したソート・シーケンスを別の CCSID に変換することが必要になります。ユーザーが 65535 以外のジョブ CCSID を持っている場合は、MIN、MAX および BREAKn の処理を行う前に、ソート・シーケンス表をそのジョブ CCSID に変換しておかなければなりません。ユーザーが 65535 のジョブ CCSID を持っている場合は、ソート・シーケンス表を、MIN、MAX および BREAKn の処理を行う列の CCSID に変換しなければなりません。この変換の結果は、印刷される報告書の形式に影響します。

Query 管理機能での CCSID のソート・シーケンス処理

ソート・シーケンス表をデータの比較に使用する場合には、そのソート・シーケンス順序表は、常にデータの CCSID に変換されます。 CCSID 変換が正しく行われれば、表示および印刷される報告書は正しく作成されます。次の状況では、問題が起こることがあります。

- 置換文字を使用した変換

  ソース CCSID 内の文字の中には、表の CCSID 変換時に CCSID で表せない文字があります。このような場合には、変換したソート・シーケンス表を作成する時に置換文字が使用されます。ソート・シーケンス表で認識されないすべての文字はみな同じ重みを持つので、外見上矛盾する比較がフォーマット設定時に起こることがあります。通常のデータの CCSID 変換でも、未知の文字はすべて同じ置換値に変換されます。
- 変換の失敗

MIN、MAX および BREAKn 処理を行うのに適した CCSID にソート・シーケンス表を変換できないと、報告書はフォーマット設定されますが、次のメッセージがジョブ・ログに送られます。

```
QWM1723 - Cannot convert the sort sequence for use on column ColName.
```

このエラーは、MIN、MAX または BREAKn の処理を必要とする報告書の場合にのみ生じます。MIN、MAX または BREAKn を使用する列の明細値は、この変換エラーを表示しないで、ソート・シーケンスを使用します。フィールドと合計のすべての出力は、その列の幅一杯の疑問符 (**'?'**) の行に置き換えられます。

Query 管理機能でのその他の考慮事項

異なる CCSID を持つソース・ファイルにプロシージャーをインポートおよびエクスポートする場合には、そのプロシージャーはまずジョブの CCSID に変換され、次にソース・ファイルの CCSID に変換されます。この余分の変換を避けるためには、インポートまたはエクスポートを行わずに、もう一方のファイルにプロシージャーをコピーしてください。

Query 管理機能書式は、報告書に表示される時点ではジョブの CCSID に変換されません。このため、書式の一部が認識不能になる場合があります。

Query 管理機能報告書が表示または印刷される場合には、データはジョブの CCSID に変換されます。SAVE DATA AS コマンドを使用してデータを保管する時に、該当するファイルが存在していない場合には、データの元となるファイルの CCSID を使用してファイルが作成されます。元のファイルと異なる CCSID を持つ既存のファイルにデータが保管される場合には、このデータはデータの保管先のファイルの CCSID に変換されます。データが表示された後に保管される場合には、このデータは表示される時点でジョブの CCSID に変換され、保管時にファイルの CCSID に変換されます。この余分の変換を避けるためには、RUN QUERY コマンドに DISPLAY=NO を指定するか、あるいは STRQMQRY CL コマンドの出力パラメーターに *OUTFILE を指定してください。

Query 管理機能プログラム、書式、およびプロシージャーは、印刷時にジョブ CCSID に変換されます。

第 11 章 DB2 UDB for iSeries Query 管理機能の考慮事項

この章では、Query 管理機能が他のシステム機能とどのように対話するかを説明し、これらの機能を使用する時に役立ついくつかの手法を説明します。

『Query 管理機能での一時変更に関する考慮事項』

176 ページの『Query 管理機能のその他のヒントおよび手法』

191 ページの『Query 管理機能の実行時環境』

191 ページの『Query 管理機能の処理限界』

192 ページの『Query 管理機能のリリース相互間の考慮事項』

Query 管理機能での一時変更に関する考慮事項

データベース・ファイル一時変更 (OVRDBF) コマンドによって指定される一時変更を使用すれば、参照を別のファイルに変更することができます。以下のセクションでは、Query 管理機能がいろいろなタイプのファイルを処理する時の、一時変更の処理方法に関するいくつかの考慮事項について説明します。

Query 管理機能の表およびビュー

RUN QUERY コマンドの実行中に構造化照会言語 (SQL) ステートメントの中で参照される表およびビューの一時変更に関する考慮事項は、SQL で使用されるものと同じです。一時変更を指定した場合には、次のパラメーターが処理されます。

- TOFILE
- MBR
- SEQONLY
- LVLCHK
- INHWRT
- WAITRCD

SQL では、RUN QUERY の前の OVRDBF コマンドの MBR キーワードに所要のメンバーを指定することによって、 Query 管理機能プログラムの最初のメンバー以外のメンバーを処理することができます。

Query が MBR(*ALL) の一時変更があるファイルからメンバーを選択した場合には、その Query は正常に実行されません。 SQL ステートメントでの一時変更の使用に関する詳細は、 Information Center の SQL プログラミング 概念というトピックを参照してください。

Query 管理機能の ERASE TABLE コマンドで参照される表

ERASE TABLE コマンドでは、一時変更は無視されます。

Query 管理機能の SAVE DATA AS で参照される表およびビュー

OVRDBF CL コマンドを使用すれば、コマンドに指定された表またはビュー以外のファイルを処理するように、Query 管理機能に指示することができます。 OVRDBF コマンドの TOFILE キーワードに指定されたファイルが存在していない場合には、一時変更は無視されます。

SAVE DATA AS コマンドを出す前に、 OVRDBF コマンドの MBR キーワードに所要のメンバーを指定することによって、ファイルの最初のメンバー以外のメンバーにデータを保管することができます。

MBR(*ALL) の一時変更があるファイルに対して SAVE DATA AS コマンドを出した場合には、そのコマンドは正常に実行されません。

一時変更を指定した場合には、SAVE DATA AS コマンドで次のパラメーターが処理されます。

- TOFILE
- MBR
- SEQONLY
- LVLCHK
- INHWRT
- WAITRCD

Query 管理機能でのソース・ファイルの IMPORT および EXPORT

IMPORT または EXPORT コマンドによって参照される、ソース・ファイルの一時変更が処理されます。

ソース・ファイルに一時変更を指定した場合には、IMPORT または EXPORT コマンドで次のパラメーターが処理されます。

- TOFILE
- MBR

MBR(*ALL) の一時変更があるソース・ファイルからの IMPORT が可能です。この IMPORT では、各メンバーの各レコードが処理されます。メンバーは作成された順に読み込まれます。 IMPORT 完了メッセージには、インポート時に処理された最初のメンバーの名前だけがリストされます。

ソース・ファイルに MBR(*ALL) の一時変更がある場合には、そのファイルの EXPORT は正常に実行されません。

EXPORT が一時変更のあるファイル名を参照していて、一時変更の対象となるファイルが存在していない場合には、 Query 管理機能がファイルを作成します。そのファイルには、OVRDBF コマンドの TOFILE キーワードに指定された名前と同じ名前が付けられます。たとえば、Query 管理機能の呼び出しの前に、次の CL コマンドが実行されたとします。

```
OVRDBF FILE(XYZ) TOFILE(MYLIB/MYFILE)
```

ファイル MYFILE は MYLIB には存在していません。次の Query 管理機能コマンドにより、結果として、ライブラリー MYLIB に MYFILE という名前のソース物理ファイルが作成されることになります。

```
EXPORT QUERY MYQUERY TO XYZ
```

一時変更とともに指定されるメンバー名は、コマンドに指定されたメンバー名より優先します。たとえば、Query 管理機能の呼び出しの前に、次の CL コマンドが実行されたとします。

```
OVRDBF FILE(XYZ) TOFILE(MYLIB/MYFILE) MBR(MEMBER2)
```

次の Query 管理機能コマンドを出すと、結果として Query の MYQUERY のソースが、ライブラリー MYLIB のファイル MYFILE のメンバー MEMBER2 に入れられます。

```
EXPORT QUERY MYQUERY TO XYZ(MEMBER1)
```

Query 管理機能の Query プロシージャー

RUN PROC、ERASE PROC、PRINT PROC、IMPORT PROC、または EXPORT PROC コマンドで Query プロシージャーとして参照されるファイルの一時変更は処理されません。 RUN PROC によって実行中のプロシージャー中の Query コマンドによって処理される、その他のファイルの一時変更は処理されます。IMPORT PROC および EXPORT PROC コマンドに指定されたソース・ファイルの一時変更は処理されます。

Query 管理機能は、実行中のプロシージャーと同じ名前のファイルについては、プロシージャーが実行されている間は、そのファイルについての一時変更は処理することができません。この規則は、次のものに適用されます。

- ソース・ファイルがプロシージャー・ファイルと同じ名前である場合の、 IMPORT PROC または EXPORT PROC コマンドのソース・ファイル。
- 同じ名前のプロシージャーの中で実行されるか、あるいは同じ名前のプロシージャーの中でネストされたプロシージャーに対して実行される、 IMPORT または EXPORT コマンドのソース・ファイル。
- コマンドが同じ名前のプロシージャーの中で実行されるか、あるいは同じ名前のプロシージャーの中でネストされたプロシージャーに対して実行される場合に、 SAVE DATA AS コマンドで参照されるファイル。
- RUN Query が同じ名前のプロシージャーの中で実行されるか、あるいは同じ名前のプロシージャーの中でネストされたプロシージャーに対して実行される場合に、その Query の中の SQL ステートメントによって参照されるファイル。

次の例は、PROC ステートメントの一時変更がどのように処理されるかを示します。

- 次の CL コマンドが Query 管理機能の呼び出しの前に実行されたとします。

```
OVRDBF FILE(XYZ) TOFILE(MYLIB/MYFILE)
OVRDBF FILE(ABC) TOFILE(MYLIB/MYFILE)
```

- 次のコマンドの結果として、上記の CL コマンドによってファイル ABC がファイル MYFILE に一時変更されたとしても、 MYLIB のプロシージャー ABC が処理されます。

```
RUN PROC MYLIB/ABC
PRINT PROC MYLIB/ABC
ERASE PROC MYLIB/ABC
```

- 次の IMPORT コマンドは、一時変更が Query プロシージャーについては処理されておらず、それがソース・ファイルに対するものではないため、 MYLIB のソース・ファイル MYFILE から MYLIB のプロシージャー ABC をインポートします。

```
IMPORT PROC MYLIB/ABC FROM MYLIB/XYZ
```

- 次の EXPORT コマンドは、一時変更が Query プロシージャーについては処理されていないため、MYLIB のソース・ファイル ABC に MYLIB のプロシージャー ABC をインポートします。ソース・ファイルに指定されたファイルは Query プロシージャーと同じであるため、一時変更は処理されていません。

```
EXPORT PROC MYLIB/ABC(MEMBER1) TO MYLIB/ABC(MEMBER2)
```

- QUERY1 と呼ばれる Query には、次の SQL ステートメントが入っています。

```
SELECT * FROM MYLIB/ABC A1, MYLIB/XYZ A2 WHERE A1.X=B1.X
```

そして、ファイル MYLIB/ABC には、RUN QUERY QUERY1 というコマンドが入っています。次の RUN PROC コマンドは、MYLIB の Query プロシージャー ABC を実行します。この Query が実行される時に、MYLIB のファイル ABC および MYLIB の MYFILE からデータが選択されます。 Query プロシージャー ABC の処理中は、ファイル ABC に対する一時変更は処理されません。

```
RUN PROC MYLIB/ABC
```

一時変更の詳細については、 データベース・プログラミングおよびファイル管理を参照してください。

Query 管理機能のその他のヒントおよび手法

このセクションでは、Query 管理機能の特別なアプリケーションを説明し、他のプロダクトおよびシステム機能を使用して Query 管理機能をより容易に処理できる方法を示します。ヒントおよび手法の多くは、Query for iSeries オブジェクトからの情報の使用が関係します。したがって、それらのヒントおよび手法を使用する前に、『第 12 章 Query 管理機能での Query for iSeries 定義情報の使用』の情報を熟知している必要があります。

Query 管理機能でのオブジェクトの印刷

Query 管理機能オブジェクトを印刷する場合には、ソース・ファイル・メンバーを作成してそれを編集します。作成したメンバーを使用して印刷処理を完了するためには、次の指示を使用します。

- セッションの開始時に作成されるソース・ファイル・メンバーに次のステートメントを入れて、Query(QMQRY) オブジェクトを印刷します。

  ```
  'PRINT QUERY ライブラリー名/Query 名 (PRINTER= プリンター名'
  ```

  次に、プロシージャー・メンバーに対して Query 管理機能プロシージャーの開始 (STRQMPRC) CL コマンドを実行して、Query 管理機能プログラムの内容を印刷します。
- セッションの開始時に作成されるソース・ファイル・メンバーに次のステートメントを入れて、書式 (QMFORM) オブジェクトを印刷します。

  ```
  'PRINT FORM ライブラリー名/書式名 (PRINTER= プリンター名'
  ```

  次に、Query 管理機能プロシージャーの開始 (STRQMPRC) CL コマンドを実行して、 Query 管理機能書式の内容を印刷します。
- セッションの開始時に作成されるソース・ファイル・メンバーに次のステートメントを入れて、プロシージャー (QMPROC) オブジェクトを印刷します。

  ```
  'PRINT PROC ライブラリー名/プロシージャー名 (PRINTER= プリンター名'
  ```

  次に、プロシージャー・メンバーに対して Query 管理機能プロシージャーの開始 (STRQMPRC) CL コマンドを実行して、Query 管理機能プロシージャーの内容を印刷します。
- セッションの開始時に作成されるソース・ファイル・メンバーに次のステートメントのいずれか (または両方) を入れて、Query for iSeries QRYDFN オブジェクトを印刷します。

  ```
  'PRINT QUERY ライブラリー名/Query 名 (PRINTER= プリンター名'
  または
  'PRINT FORM ライブラリー名/書式名 (PRINTER= プリンター名'
  ```

  次に、プロシージャー・メンバーに対して ALWQRYDFN=YES キーワードを持つ Query 管理機能プロシージャーの開始 (STRQMPRC) CL コマンドを実行して、 QRYDFN オブジェクトの Query または書式部分を印刷します。

Query 管理機能で QRYDFN を使用するための STRQMQRY のデフォルトの変更

Query 管理機能で使用される Query および書式情報を開発して維持するために、 WRKQRY コマンドを使用したい場合には、STRQMQRY コマンドのコピーを使用することが便利です。このコマンドは、名前を

指定するだけで QRYDFN オブジェクトが実行されるようなデフォルトを使用するように変更されています。たとえば、次のコマンドを入力することによって、ジョブの現行ライブラリーにコマンド STRQRYDFN を作成することができます。

```
CRTDUPOBJ OBJ(STRQMQRY) FROMLIB(QSYS) OBJTYPE(*CMD) TOLIB(*CURLIB)
          NEWOBJ(STRQRYDFN)

CHGCMDDFT CMD(*CURLIB/STRQRYDFN) NEWDFT('QMFORM(*QMQRY)')

CHGCMDDFT CMD(*CURLIB/STRQRYDFN) NEWDFT('ALWQRYDFN(*ONLY)')
```

現行ライブラリー (*CURLIB) の QRY1 を実行するためには、 STRQRYDFN *CURLIB/QRY1 とタイプするか、あるいは STRQRYDFN QRY1 とだけタイプしてください。

Query 管理機能での QRYDFN オブジェクトの使用に関する情報の表示

QRYDFN オブジェクトから情報が派生した時に問題があった場合にとられるシステムの処置を読み取るためには、次のコマンド・ストリングを使用して、 Query 管理機能の変換メッセージを表示してください。

```
DSPMSGD RANGE(QWM2301 QWM2399) DETAIL(*BASIC)
```

表示されるメッセージには、とられるシステムの処置によっては予想外の結果となることについての警告が入っていることがあり、あるいはそのメッセージによって診断された種類の問題を回避または発生を最小限にする方法が示唆される場合があります。

Query for iSeries を使用するグローバル変数を含む Query の定義

Query for iSeries は、他の QRYDFN オブジェクトと同じ方法で実行することができない従属 QRYDFN オブジェクトを使用するデータまたはテキストのマージ機能をサポートしています。これらのオブジェクトは、レコード選択テストに変数 (従属値) が含まれる点で異なっています。これらの変数に正しい値を割り当てた場合には、Query 管理機能を使用して、このような Query を実行することができます。

Query に関する情報が従属 QRYDFN オブジェクトから派生する時に、従属値がグローバル変数に変換されます。たとえば、 :T01.cusnam は &T01_CUSNAM になります。 (Query を実行する STRQMQRY コマンドの使用時に SETVAR パラメーターに値を指定しなかった場合には、T01_CUSNAM の値を求めるプロンプトが出されます)。

派生した SELECT ステートメントの WHERE 文節に所要の値を挿入したい場合には、 SETVAR パラメーターを使用してください。たとえば、T01_CUSNAM 変数に、値 “Smith” や “%Apt%” のような住所を指定することができます。 (グローバル変数の設定例については、 241 ページの『付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用』を参照してください。)

CL プログラムおよびコマンドを作成して、プロンプトの処理を改善し、可能な選択項目を用意し、さらに入力される値の制限または妥当性検査を行うことができます。

Query 管理機能で既存の QMQRY 用の QMFORM を作成するための Query for iSeries の使用

システム・デフォルト (*SYSDFT) 書式による既存の QMQRY オブジェクトの実行では、必要なフォーマット設定結果が作成されない場合には、Query for iSeries を使用して、そのシステム・デフォルトに基づく書式情報を定義することができます。次のステップは、システムのデフォルトに基づく書式情報を定義するための手順を示しています。

1. QMQRY オブジェクトを実行し、データを表に保管します。
2. WRKQRY のオプション 1 (作成) を使用して、QRYDFN オブジェクトを定義します。

a. データが保管されている表をファイルの選択項目として指定します。 (この表の定義は一時変更のためのデフォルトを提供します。)
b. 使用可能ではあっても Query for iSeries ではサポートされていない次の機能の使用について、考慮してください。
   - ページ・テキストにおける & フィールド挿入変数
   - 下線文字で終わる & フィールド挿入変数
   - 切れ目または最終テキストでの切れ目フィールド挿入変数以外の変数
   - 数字以外の文字で終わる & 列 # 挿入変数
c. ここで定義したものを QRYDFN オブジェクトとして保管します。

3. 元の Query を実行する時に新しい QRYDFN オブジェクトからの書式情報を使用するように Query 管理機能に要求するか、あるいは書式情報を検索しそれを使用して、 QMQRY オブジェクトで使用するための QMFORM を作成します。

Query 管理機能でのサイズを超えた単一レコードからのデータの表示

列見出し形式で表示するには広すぎる単一レコードの複数列報告書を作成する QRYDFN オブジェクトがあった場合には、報告書全体を表題付きで表示する書式を作成することができます。次のステップを使用して、新しい書式を作成します。

1. WRKQRY コマンドを使用して、QRYDFN オブジェクトを変更します。
   a. すべての報告書列について 0 の長さを指定します。
   b. 適切に調整された表題を持つページ見出しテキストを定義して、変数を挿入します。 3 行までが使用可能です。
   c. 所要のページ・フッター・テキストを使用します。
2. 書式のソースを検索して必要な調整を加えます (たとえば、追加のページ見出しまたはフッター・テキスト行、左寄せの位置合わせ、または行送り)。
3. 調整したソースから Query 管理機能書式オブジェクトを作成します。
4. ここで作成された書式を使用する Query を実行して、完全な報告書を表示します。

次は、表題付き書式で表示される単一レコード報告書の例です。

```
Attn (tele) . . . :  Howard Jones (218-485-0162)
Account name  . . :  International Milling Company
Address . . . . . :  4126 Kettering Memorial Parkway
City, state . . . :  Fort Wayne, In.
Zip . . . . . . . :  46815

Invoice # . . . . :  B12358-9
Date shipped  . . :  03/27/90
Hauler  . . . . . :  Dave (3-7809)
```

Query 管理機能の PDM オプションでの Query 管理機能または CL コマンドの使用

QMQRY、QMFORM、または QRYDFN オブジェクトのリストを処理する時には、プログラム開発管理機能 (PDM) の使用が便利な場合があります。リスト項目の前にオプション・コードをタイプする時に、選択した置き換えライブラリーおよびオブジェクト名の後に CL コマンド (たとえば、 STRQMQRY または ANZQRY) を実行するオプションを定義してください。たとえば、

- オプション SQ を次のように定義します。

```
STRQMQRY &L/&N QMFORM(*QMQRY) ALWQRYDFN(*ONLY)
```

次に、SQ を使用し、デフォルトが QMFORM(*SYSDFT) および ALWQRYDFN(*NO) に一時変更されることは気にせずに、選択された QRYDFN オブジェクトから派生する Query 情報および書式情報を使用して報告書を表示します。

- オプション Z を次のように定義します。

```
ANZQRY   &L/&N 99
```

次に、QRYDFN オブジェクトのリスト中のすべての名前の前に Z をタイプして、完了メッセージを入手します。これらのメッセージを調べて、満足のゆく Query 管理機能の使用のために、どの QRYDFN オブジェクトが調整を必要としているかを見つけます。

ユーザーが開発した CL コマンドを実行したり、あるいは Query 管理機能オブジェクトに作用するユーザーが開発した CL プログラムを呼び出すために、他のオプションを定義することもできます。

Query 管理機能で QRYDFN オブジェクトを永続的に変換する CL プログラムの作成

QRYDFN オブジェクトから Query 管理機能オブジェクトへの変換操作が頻繁に実行される場合には、そのような変換を行うための CL プログラムを作成することができます。そのプログラムのためのパラメーターを定義して、オブジェクト名およびその他の変数を指定します。

180 ページの図 47 は、QRYDFN 情報を Query 管理機能オブジェクトに変換するプログラムのためのソースの例です。このプログラムでは、QRYDFN オブジェクトは *LIBL の中で探索され、また Query 管理機能オブジェクトは *CURLIB に入れられるものと仮定されています。この例では、QRYDFN オブジェクトから Query for iSeries によって作成される報告書、および変換されたオブジェクトから Query 管理機能によって作成される報告書が示されています。要求が取り消されなければ、変換されたオブジェクトはプログラムによって QTEMP から *CURLIB にコピーされます。

```
 Columns . . . :    1  68           Edit                         USRLIB/QCLSRC
 SEU==>                                                             MIGRATE
 FMT **  ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+...
        *************** Beginning of data *************************************
0001.00 PGM PARM(&OBJ)
0002.00 DCL VAR(&OBJ) TYPE(*CHAR) LEN(10)
0003.00 CRTSRCPF QTEMP/QQMQRYSRC 91
0004.00 MONMSG MSGID(CPF7302)
0005.00 CRTSRCPF QTEMP/QQMFORMSRC 162
0006.00 MONMSG MSGID(CPF7302)
0007.00 RUNQRY *LIBL/&OBJ OUTPUT(*)
0008.00 RTVQMQRY *LIBL/&OBJ QTEMP/QQMQRYSRC &OBJ ALWQRYDFN(*YES)
0009.00 MONMSG MSGID(QWM2701)
0010.00 CRTQMQRY QTEMP/&OBJ QTEMP/QQMQRYSRC &OBJ
0011.00 MONMSG MSGID(QWM2701)
0012.00 RTVQMFORM *LIBL/&OBJ QTEMP/QQMFORMSRC &OBJ ALWQRYDFN(*YES)
0013.00 MONMSG MSGID(QWM2701)
0014.00 CRTQMFORM QTEMP/&OBJ QTEMP/QQMFORMSRC &OBJ
0015.00 MONMSG MSGID(QWM2701)
0016.00 STRQMQRY QMQRY(&OBJ) QMFORM(*QMQRY)
0017.00 MONMSG MSGID(QWM2701 QWM2703) EXEC(RETURN)
0018.00 DLTQMQRY QMQRY(*CURLIB/&OBJ)
0015.00 MONMSG MSGID(CPF2105)
0019.00 CRTDUPOBJ OBJ(&OBJ) FROMLIB(QTEMP) OBJTYPE(*QMQRY) TOLIB(*CURLIB)
0020.00 DLTQMFORM QMFORM(*CURLIB/&OBJ)
0015.00 MONMSG MSGID(CPF2105)
0021.00 CRTDUPOBJ OBJ(&OBJ) FROMLIB(QTEMP) OBJTYPE(*QMFORM) TOLIB(*CURLIB)
0022.00 ENDPGM
        ****************** End of data ***************************************
```

図 47. 永続変換プログラムのための CL ソース

Query 管理機能でのフィールド値に対する Query 処理

特定のファイルのフィールドで使用される値の順序正しいリストを表示するために、総称 Query を作成することができます。 Query が実行される時に、ライブラリー、ファイル、およびフィールド名をグローバル変数にセットすることができます。次の例は、総称 Query を作成する SELECT ステートメントです。

```
SELECT DISTINCT &FIELD FROM &LIBRARY/&FILE ORDER BY 1
```

このステートメントから作成された QMQRY オブジェクトを実行して、下記の SETVAR パラメーターに指定されるリストを入手してください (ここでは、作成される QMQRY オブジェクトの名前は qryvalues であり、 dept という名前のフィールドを持つ staff という名前のデータベース・ファイルがライブラリー testdata の中にあるものと仮定しています)。

```
STRQMQRY qryvalues SETVAR((FIELD dept) (LIBRARY testdata) (FILE staff))
```

変数をセットする時にレコード選択テストを追加して、値のサブセットを入手します。

```
STRQMQRY qryvalues SETVAR((FIELD dept) (LIBRARY testdata)
         (FILE 'staff where dept > 50'))
```

変数をセットする時にアスタリスク (*) を使用して、重複したレコードのないすべての列を表示します。

```
STRQMQRY qryvalues SETVAR((FIELD '*' ) (LIBRARY testdata) (FILE staff))
```

単純な CL プロンプト・プログラムおよびコマンドを作成して、グローバル変数に対する値の指定をより容易にします。次のようにタイプして、所要のリストが得られるようにこれをセットアップします。

```
q testdata/staff dept
```

ソース入力ユーティリティー (SEU) を使用して Query のソースを編集していてテストに使用可能な値を参照したい場合には、このコマンドの使用が役立ちます。 SEU を使用すれば、システムまたはユーザー定義のコマンドを入力するためのウィンドウを要求することができます。

Query 管理機能での Query への変数値の受け渡し

グローバル変数名は必ずしも意味のある名前である必要はないため、値を求めるプロンプトを受け取ったユーザーは、何をタイプすべきかが分からない場合があります。この場合に、意味のあるプロンプトを出し、タイプされた値の妥当性を検査する CL プログラムおよびコマンドを作成することができます。 図 48 および図 49 は、プログラムの作成および Query 用のコマンドの作成に使用できるソース・ステートメントを示しています。これは、指定されたファイルの最初のメンバー内の指定フィールドの値の順序正しいリストを表示するためのものです。 CL プログラムをこのプログラム・ソースから作成し、コマンドがコマンド・ソースから作成される時に、このプログラムをコマンド処理プログラムとして指定してください。

```
0001.00 PGM PARM(&FILE &FIELD)
0002.00 DCL VAR(&FILE) TYPE(*CHAR) LEN(20)
0003.00 DCL VAR(&LIB) TYPE(*CHAR) LEN(10)
0004.00 DCL VAR(&TABLE) TYPE(*CHAR) LEN(10)
0005.00 DCL VAR(&FIELD) TYPE(*CHAR) LEN(10)
0006.00 CHGVAR &LIB %SUBSTRING(&FILE 11 10)
0007.00 CHGVAR &TABLE %SUBSTRING(&FILE 1 10)
0008.00 STRQMQRY MYLIB/QRYVALUES SETVAR((LIBRARY &LIB)(FILE &TABLE)(FIELD &FIELD))
0009.00 ENDPGM
```

図 48. グローバル変数プロンプト・プログラムの CL ソース

```
                                                                  ******
0001.00  Q:          CMD        PROMPT('Query 列値')
0002.00              PARM       KWD(FILE) TYPE(Q1) MIN(1) MAX(1) +
0003.00                           PROMPT(' テーブル名 ')
0004.00              PARM       KWD(FIELD) TYPE(*CHAR) LEN(10) +
0005.00                           PROMPT(' 列名 ')
0006.00  Q1:         QUAL       TYPE(*NAME) LEN(10) MIN(1)
0007.00              QUAL       TYPE(*NAME) LEN(10) +
0008.00                         DFT(*LIBL) +
0009.00                         SPCVAL(*LIBL (*CURLIB *CURLIB)) +
0010.00                         PROMPT(' コレクション ')
```

図 49. グローバル変数プロンプト・コマンドの CL ソース

次の図は、ユーザーが開発した変数値を渡すために必要なプロンプト画面の例です。

```
                        Query 列値 (Q)

選択項目を入力して、実行キーを押してください。
テーブル名 . . . . . .                  名前
  コレクション . . . . . .     *LIBL      名前、*LIBL、*CURLIB
カラム名 . . . . . .                    文字値

                                                                  終り
F3=終了    F4=プロンプト    F5=最新表示    F12=取り消し
F13=この画面の使用法       F24=キーの続き
```

Query 管理機能での列見出しのない列の定義

列に見出しを入れないためには、最上部の見出し行の左端に *NONE を指定します。この見出し行は、対話式データ定義ユーティリティー (IDDU) のもとで定義を処理する時、または WRKQRY コマンドを使用して Query 管理機能を適用する Query for iSeries QRYDFN オブジェクトを定義する時に表示されるものです。エンコードされた書式ソースの列見出しとして、*NONE を指定することもできます。いずれの場合にも、列見出し区切り記号を報告書全体から除去するか、あるいはすべての列見出しに *NONE を指定しない限り、列に区切り記号が残ります。列見出し区切り記号を除去するためには、書式のソースの中から適切なフィールドを検索して編集し、書式を作成し直します。

ISQL 開発の Query をフォーマット設定するための Query 管理機能の使用

サポートされている SQL データベース機能のいずれかを使用する Query を開発するために、構造化照会言語 (SQL) を対話式に使用することができます。 SQL より上位にある対話式構造化照会言語 (ISQL) プロダクトを使用すれば、 SQL コマンドを対話式で実行することができます。これらの機能には、副照会、スカラー機能、GROUP BY ステートメント、および Query for iSeries のプロンプトによるインターフェースを介しては使用できないその他の機能があります。以下のステップは、ISQL で作成された SELECT ステートメントを QMQRY オブジェクトに入れる方法、およびこの QMQRY オブジェクトを実行して得られる表示または印刷出力をフォーマット設定するために Query 管理機能で使用することができる情報を、 Query for iSeries を使用して定義する方法を説明しています。

1. SQL 開始 (STRSQL) コマンドを指定します。
   a. ISQL を使用して、所要の Query を開発します。
   b. データベース (コレクション) を作成して Query からの出力を受け取るか、あるいは前に作成されたコレクションを使用します。
   c. セッション用の出力装置が前に作成されたコレクション内のデータベース・ファイルとなるように変更します。この Query の目的を記述する名前 (たとえば、QRYPURPOSE) を使用します。
   d. Query を実行し直して、ファイル (表) QRYPURPOSE を作成します。

e. Query セッションをメンバー QRYPURPOSE として保管します。ファイルおよびライブラリー名はユーザーが指定するので、後でセッションを編集して、 Query 管理機能プログラムのソースとして使用できることに留意してください。

f. ISQL セッションを終了します。

2. SEU 開始 (STRSEU) コマンドを指定して、保管されたセッションを編集します。
   a. この Query を定義する SELECT ステートメントが入っていないすべての行を除去します。
   b. 必要な任意のコメントを追加します。
   c. 必要に応じて、SELECT ステートメント中の該当する要素をグローバル変数に置き換えます。
   d. SEU を終了して、変更されたメンバーを保管します。
3. Query 管理機能プログラム作成 (CRTQMQRY) コマンドを使用して、メンバー QRYPURPOSE から QMQRY オブジェクトの QRYPURPOSE を作成します。
4. WRKQRY コマンドを指定して、*作成* オプションを選択します。
   a. ISQL セッションで作成されたファイル QRYPURPOSE を選択します。
   b. 報告書列のフォーマット設定の一時変更を指定します。表示されるデフォルトは、報告書を表示するために使用される ISQL と同じですが、 *SYSDFT 書式によって QRYPURPOSE を実行した場合は、Query 管理機能で使用されるものとは同じでありません。表示されたデフォルトを使用したい場合には、それらの値が QRYDFN オブジェクトと一緒に保管されるように、 Query for iSeries がそれらを一時変更として取り扱うようにします。 (列見出しに変更があると、元のデフォルトに戻った場合であっても、デフォルトは一時変更されたものと見なされます。このことは、長さ、小数点以下の桁数、および数値編集にもあてはまります。)
   c. 編集コード J (数値)、J¥ (通貨値)、および M (数値 ID) を使用して、指定した小数点以下の桁数のすべての一時変更が組み込まれる Query 管理機能編集コードに変換できるような編集を定義します。
   d. 報告書が読みやすくなると考えられる特別なフォーマット設定があれば、それを追加します。たとえば、QRYPURPOSE に保管される SELECT ステートメントには、総計スカラー機能として定義された列の下に最終テキストおよび総合計が現れるように定義することができます。
   e. フォーマット設定選択項目を QRYDFN オブジェクトの QRYPURPOSE として保管します。
5. オプションで、QRYDFN QRYPURPOSE から書式のソースを検索し、それを使用して QMFORM QRYPURPOSE を作成します。
6. STRQMQRY コマンドを使用して、Query の QRYPURPOSE を実行します。 QMFORM(*QMQRY) を使用し、QRYDFN QRYPURPOSE から QMFORM を作成しなかった場合には、 QMFORM(*QMQRY) を使用し、ALWQRYDFN(*ONLY) を指定して、 QRYDFN オブジェクトから派生したフォーマット設定情報を強制的に使用するようにします。

図 50 は、ISQL 開発の Query を示しています。

```
                                        DB2 Query 管理機能 OS/400

Query  . . . . . :   MAXSALARY
  ライブラリー . . :     USRLIB
テキスト . . . . . :
SEQNBR *...+....1....+....2....+....3....+....4....+....5....+....6....+....7...
000001 select dept,max(salary) from testdata/staff group by dept
                                          * * * *   ソースの終り   * * * *
```

図 50. ISQL 開発の QMQRY プログラム・オブジェクトの印刷例

次の画面は、 183 ページの図 50 の Query から作成されたフォーマット設定済み報告書を示しています。この報告書は、ISQL 出力ファイル定義から作成された QRYDFN オブジェクトから派生した書式情報を、Query 管理機能で使用して作成されます。

```
                              報告書の表示
Query . . . . .:   USRLIB/MAXSALARY          幅    . . .:      71
書式  . . . . .:   USRLIB/MAXSALARY          桁      . .:       1
制御  . . . .
行     |...+....1....+....2....+....3....+....4....+....5....+....6....+....7.
                                    最大
                          部門      給与
                    ------------  ------------
000001
000002
000003                      10     $22,959.20
000004                      15     $20,659.80
000005                      20     $18,357.50
000006                      38     $18,006.00
000007                      42     $18,352.80
000008                      51     $21,150.00
000009                      66     $21,000.00
000010                      84     $19,818.00
000011                            ============
000012       全体の最大給与:
000013                             $22,959.20
                                                                    続く...
   F3=終了      F12=取消し      F19=左        F20=右         F21=分割
```

Query 管理機能での参照制約付きの ISQL 報告書処理選択の使用

参照制約の能力と機能を使用する環境で ISQL を使用する時は、次の点に留意してください。

- 選択された表が参照制約の場合は、この制約は、ISQL *OUTFILE を使用して作成される出力ファイルには加えられません。
- SELECT ステートメントの結果が既存のファイルへの出力となる場合は、出力表に違反となる参照制約があれば、*OUTFILE の処理は失敗します。このエラーは、既存のファイルが変更された後で、はじめて検出されます。
- SELECT ステートメントの結果が従属ファイルまたは親ファイルへの出力となる場合、設定されたまたは使用可能になっている制約が検査保留であれば、検査保留エラーが起こる可能性があります。検査保留エラーは既存のファイルを変更する前に、見つけなければなりません。

Query 管理機能で最終合計に表題をスタックするためのテキスト挿入変数の使用

次の図は、合計値をスタックした表題付きの最終レベルの合計報告書の例です。この例は、最終合計値を複数の表示画面または印刷出力ファイルの異なる列に分散して示すのではなく、 1 ページ内に任意の順序で示すことができることを表しています。

```
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
       (Salary analysis for 35 employees in department 10)

  Minimum.....:$10,506
  Maximum.....:$22,959
  Average.....:$16,676
  Total.......:$583,647

/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
```

図 51. カバー・ページおよび見出しテキスト挿入としての最終レベルの合計値

この報告書は、次の特性を持つ単一の QRYDFN オブジェクトから Query 管理機能によって作成されたものです。

- 合計専用の出力書式
- 切れ目フィールドが選択されていない
- 最終レベルの合計が抑止されていない
- 合計が選択されている
- 挿入として使用されるすべての合計フィールドに対して長さ 0 が指定されている
- カバー・ページおよびページ見出しテキストに、出力列番号の &# 参照によって示される合計値挿入の配置で所要の表題および見出しが含まれる

注: Query for iSeries を使用して QRYDFN を実行しようとした場合、0 にセットされているすべてのフィールド幅が診断され、報告書は作成されません。

Query 管理機能での表レイアウトとテキストの組み合わせの使用

テキスト挿入は表レイアウトと組み合わせて使用することができます。次の例を作成するために、前述のヒントで説明した特性を持つ QRYDFN オブジェクトについてレコード選択テストが定義されています。出力を特定の顧客 (グローバル変数を使用するために実行時に指定された) に限定するために、レコード選択テストが定義され、ラベルの定義に使用するカバー・ページ・テキストおよび最終テキストの挿入を行うために、 MIN および COUNT 関数が定義されています。

```
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
                      (Orders Inquiry)

 Herman B. Wannamaker
 3124 Melrose Ave - Apt 35
 Gooseneck, NY  55945


       TOTAL           AVG           MAX           MIN
      Charges         Price         Price         Price
--------------  ------------  ------------  ------------
    $3,859.72        $79.54      $1024.89         $3.50

Number of transactions:  35
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
```

図 52. 合計表を用いた最終レベルのテキスト挿入

Query 管理機能での複数レベルの合計専用の QRYDFN の変換

次に示す 2 つの図は、切れ目合計と最終合計の両方を表示した合計専用の報告書です。 Query 管理機能報告書は、より読みやすい書式で情報を示しています。これは、Query 定義の QRYDFN オブジェクトの SQL 列関数から得られる最低レベルの合計、および 2 番目の列使用目的から得られるその他の合計を使用して作成されています。この報告書は次のようにして作成されました。すべてのレベルを縮小し最終合計を抑止するように元の QRYDFN のコピーを変更し、それを Query として使用するために保管しました。もう 1 つのコピーを最低レベルの合計を抑止するように変更し、それを書式として使用するために保管しました。次に、QRYDFN オブジェクトの使用を許した状態で、STRQMQRY をこの Query および書式に適用しました。全体の平均は、実際は平均値の平均であることに注意してください。

```
                     Salary Report Summary, 1989
                  AVERAGE       MINIMUM      MAXIMUM
Job      Years    Salary        Salary       Salary
-----   -------   ---------     ---------    ---------
CLERK         0   $12,655.98    $11,508.60   $13,504.60
              1   $10,988.00    $10,988.00   $10,988.00
              3   $12,689.78    $12,009.75   $13,369.80
              4   $12,258.50    $12,258.50   $12,258.50
              5   $12,769.35    $12,508.20   $13,030.50
              6   $12,482.95    $10,505.90   $14,460.00
              8   $14,252.75    $14,252.75   $14,252.75
                  -----------   -----------  ------------
Overall CLERK:
                  $12,585.33    $10,505.90   $14,460.00

MANAGER       5   $18,383.50    $17,506.75   $19,260.25
              6   $21,150.00    $21,150.00   $21,150.00
              7   $19,889.83    $18,352.80   $22,959.20
              9   $18,555.50    $18,555.50   $18,555.50
             10   $20,162.60    $19,818.00   $20,659.80
             12   $21,234.00    $21,234.00   $21,234.00
                  -----------   -----------  ------------
  Overall MANAGER:
                  $19,895.80    $17,506.75   $22,959.20

SALES         0   $16,808.30    $16,808.30   $16,808.30
              4   $16,858.20    $16,858.20   $16,858.20
              5   $15,454.50    $15,454.50   $15,454.50
              6   $18,488.08    $18,001.75   $19,456.50
              7   $17,333.78    $16,502.83   $17,844.00
              8   $18,171.25    $18,171.25   $18,171.25
              9   $18,674.50    $18,674.50   $18,674.50
             13   $21,000.00    $21,000.00   $21,000.00
                  -----------   -----------  -----------
Overall SALES:
                  $17,848.36    $15,454.50   $21,000.00

                  ===========   ===========  ===========
Overall:
                  $16,679.64    $10,505.90   $22,959.20
06/18/90  09:50:21
```

図 53. SQL 列関数に適用された書式使用例

```
06/21/90  14:23:04  Salary Report Summary, 1989  ページ    1
Job    Years       Salary
CLERK      0
           平均値 $12,655.98
           最小値 $11,508.60
           最大値 $13,504.60

CLERK      1
           平均値 $10,988.00
           最小値 $10,988.00
           最大値 $10,988.00

CLERK      3
           平均値 $12,689.78
           最小値 $12,009.75
           最大値 $13,369.80

CLERK      4
           平均値 $12,258.50
           最小値 $12,258.50
           最大値 $12,258.50

CLERK      5
           平均値 $12,769.35
           最小値 $12,508.20
           最大値 $13,030.50

CLERK      6
           平均値 $12,482.95
           最小値 $10,505.90
           最大値 $14,460.00

CLERK      8
           平均値 $14,252.75
           最小値 $14,252.75
           最大値 $14,252.75

CLERK
           Overall CLERK:
           平均値 $12,612.61
           最小値 $10,505.90
           最大値 $14,460.00

MANAGER    5
           平均値 $18,383.50
           最小値 $17,506.75
           最大値 $19,260.25

MANAGER    6
           平均値 $21,150.00
           最小値 $21,150.00
           最大値 $21,150.00
```

図 54. 切れ目レベルが複数個ある報告書 - Query for iSeries (1/3)

```
06/21/90  14:23:04  Salary Report Summary, 1989  ページ    2
Job    Years       Salary
MANAGER    7
           平均値 $19,889.83
           最小値 $18,352.80
           最大値 $22,959.20
MANAGER    9
           平均値 $18,555.50
           最小値 $18,555.50
           最大値 $18,555.50
MANAGER   10
           平均値 $20,162.60
           最小値 $19,818.00
           最大値 $20,659.80

MANAGER   12
           平均値 $21,234.00
           最小値 $21,234.00
           最大値 $21,234.00

MANAGER
           Overall MANAGER:
           平均値 $19,805.80
           最小値 $17,506.75
           最大値 $22,959.20

SALES      0
           平均値 $16,808.30
           最小値 $16,808.30
           最大値 $16,808.30

SALES      4
           平均値 $16,858.20
           最小値 $16,858.20
           最大値 $16,858.20

SALES      5
           平均値 $15,454.50
           最小値 $15,454.50
           最大値 $15,454.50

SALES      6
           平均値 $18,488.08
           最小値 $18,001.75
           最大値 $19,456.50

SALES      7
           平均値 $17,333.78
           最小値 $16,502.83
           最大値 $17,844.00
```

図 54. 切れ目レベルが複数個ある報告書 - Query for iSeries (2/3)

```
06/21/90  14:23:04  Salary Report Summary, 1989  ページ    3
Job    Years       Salary
SALES      8
           平均値 $18,171.25
           最小値 $18,171.25
           最大値 $18,171.25

SALES      9
           平均値 $18,674.50
           最小値 $18,674.50
           最大値 $18,674.50

SALES     13
           平均値 $21,000.00
           最小値 $21,000.00
           最大値 $21,000.00

SALES
           Overall SALES:
           平均値 $17,869.36
           最小値 $15,454.50
           最大値 $21,000.00

           Overall:
           平均値 $16,675.64
           最小値 $10,505.90
           最大値 $22,959.20
   * * *  報告書の終わり * * *
```

図 54. 切れ目レベルが複数個ある報告書 - Query for iSeries (3/3)

Query 管理機能での切れ目レベル合計グループのソートおよび部分設定

切れ目レベル合計報告書を作成する QRYDFN オブジェクトまたは QMQRY オブジェクトがあれば (SQL ステートメントに SQL 列関数および GROUP BY 文節が含まれる)、QMQRY オブジェクト用にソースを作成して不要なグループを除外するために列関数の値を使用し、残りのグループを合計に基づいて順序付けることができます。これは、HAVING および ORDER BY 文節を検索されたソースの中に編集することによって行うことができます。次のステートメントは、口座番号の集合について、口座番号順の当座貸越し合計および回数のリストを作成します。

```
SELECT ACCTNUM, COUNT(*), SUM(OVRDRFT) FROM ACCTINFO/OVRDRFTS
GROUP BY ACCTNUM  ORDER BY ACCTNUM
```

次のステートメントは許容限度内の当座貸越し合計を持つ口座番号を除外し、残りを当座貸越し合計 (降順) および当座貸越し回数 (昇順) によって順番に並べます。

```
SELECT ACCTNUM, COUNT(*), SUM(OVRDRFT) FROM ACCTINFO/OVRDRFTS
GROUP BY ACCTNUM  HAVING SUM(OVRDRFT) > 100
ORDER BY 3 DESC, 2, ACCTNUM
```

Query 管理機能での SQL 機能の追加

| QRYDFN オブジェクトを変更するか、または、 QRYDFN オブジェクトから取り出した Query 管理機能
| ソース・メンバーを編集することによって、 Query for iSeries ではサポートされていない SQL 機能を追
| 加することができます。次のリストは、追加の SQL 機能を入手する方法を説明しています。

| • 次のような許されているが Query for iSeries ではサポートされていない機能を定義するためには、
| Query 処理 (WRKQRY) コマンドを使用します。
| – ページ・テキストにおける & フィールド挿入変数

- 下線文字で終わる & フィールド挿入変数
- 切れ目テキストまたは最終テキスト内の切れ目フィールド挿入変数以外のもの
- 数字以外の文字で終わる & 列 # 挿入変数

- ソース入力ユーティリティー開始 (STRSEU) コマンドを使用して、 WRKQRY コマンドを用いて定義することができない機能を追加するために、検索したソースを次のように変更します。
  1. QRYDFN オブジェクトまたは Query 管理機能プログラム (QMQRY) オブジェクトまたは Query 管理機能書式 (QMFORM) オブジェクトからソースを検索します。これには、 ALWQRYDFN(*YES) パラメーターを持つ RTVQMQRY および FORM CL コマンドを使用します。
  2. そのソースを編集して所要の機能を追加します (下記の『Query 管理機能で追加可能な SQL 機能』を参照)。
  3. 編集されたソースを使用して、QMQRY オブジェクトまたは QMFORM オブジェクトを作成します。これは、以降の Query 管理機能要求の中で、 CRTQMQRY および CRTQMFORM CL コマンドを用いて参照することができます。
- SQL 機能を追加するために、DB2 UDB for iSeries Query Manager プロダクトを使用して、 QMQRY または QMFORM オブジェクトを変更します。詳細は、*Query Manager ご使用の手引き* を参照してください。

Query 管理機能で追加可能な SQL 機能

次の機能を Query 管理機能プログラムのソースに追加することができます。

- ALL レコードに代わる DISTINCT レコード
- アスタリスク (*) を使用するすべてのフィールドの選択
- FROM 文節の SQL 命名規則
- SQL 式または述部テスト値としての列関数またはスカラー関数
- 探索条件修飾子としての NOT
- GROUP BY および HAVING 文節
- 括弧と結合子の併用

次の機能を Query 管理機能書式のソースに追加することができます。

- 文字フィールド編集
- 数値編集のためのいろいろな Query 管理機能編集コード
- 明示的な列幅の制御
- FIRST および LAST の列関数
- 10 個以上の切れ目列の使用能力
- 書式定義内の再順序付け
- 報告書内の行送り
- 合計行の配置 (2 行目以外で)
- 切れ目見出しテキスト
- (OS/400 の限度を超える) 追加のテキスト行
- テキスト行の位置合わせ
- フレームの制御 (たとえば、区切り記号、デフォルト切れ目テキスト、または罫線など)

Query ソースの編集用の SQL 構文および書式ソースの編集用のエンコードされた書式レイアウトの詳細については、「SQL 解説書」を参照してください。

Query 管理機能の実行時環境

パフォーマンス上の理由により、Query 管理機能はコマンドの完了時にその実行時環境を終了しません。その結果、次の Query 管理機能のコマンドが出された際に開始が早まります。これを実行するために、少量のストレージが保存されます。実行時環境を停止させるためには、次のコマンドを使用してください。

```
ENDEPMENV QQXCPIENV
```

Query 管理機能の処理限界

Query 管理機能では、ユーザーの希望通りには報告書を処理できない場合があります。次に、Query 管理機能による処理の限界について説明します。

Query 管理機能コマンド

呼び出し可能インターフェース上のコマンド・ストリングは、256 バイトに制限されています。引用符を除去する前のプロシージャー中のコマンド・ストリングも、256 バイトに制限されています。

1 つのコマンドには、1000 個を限度としてキーワードおよび変数を指定することができます。これは、コマンド・ストリングの一部として指定されたキーワードまたは変数に、拡張インターフェースを介して指定されたキーワードまたは変数を加えた合計です。同じキーワードまたは変数が重複していた場合には、その重複分も限度内の 1 つとしてカウントされます。

Query 管理機能での SQL Query

ブランクとコメントを除いた後の SQL Query ステートメントのサイズは、 32 KB マイナス 1 バイトに制限されています。

Query 管理機能での外部化 Query

外部化された Query を構成するソース・ファイルについては、次のような限界が存在しています。

- レコードの幅が 79 バイトより大きい場合には、 79 桁目より後の桁にあるデータは無視されます。
- 最大 211929 行までのソース・テキストを指定することができます。

Query 管理機能の外部化書式

外部化された書式を構成するソース・ファイルについては、次のような限界が存在しています。

- レコードの幅が 150 バイトより大きい場合には、 150 桁目より後の桁にあるデータは無視されます。
- Query 管理機能では、外部化された書式オブジェクト内の重複した情報セクションが許されており、ファイルに MBR(*ALL) の一時変更が可能であるため、処理することができる外部化された書式内のソース・レコードの数にはほとんど限界というものがありません。

Query 管理機能のインスタンス

任意の一時点に活動状態にある処理またはジョブについて、最大 25 までの Query 管理機能インスタンスを指定することができます。

Query 管理機能のグローバル変数

各 Query 管理機能インスタンスについて、最大 1000 個までの固有のグローバル変数をセットすることができます。

Query 管理機能のプロシージャー制限

Query 管理機能は、Query プロシージャーの実行時および印刷時には任意のファイル幅をサポートします。Query プロシージャーのエクスポート時およびインポート時に、コマンドの対象となるソース・ファイルが作成される場合には、それが 79 バイトの幅で作成されます。コマンドの対象となるファイルがすでに存在していて、その幅が元のファイルより小さい場合には、データが切り捨てられることがあります。対象となるファイルの幅が元のファイルより大きい場合には、データが失われることはなく、レコードにはブランクが埋め込まれます。

Query 管理機能のリリース相互間の考慮事項

属性が PROMPT の Query 管理機能プログラムは、前のリリースで有効な SQL 機能だけを使用する場合、OS/400 のバージョン 2 リリース 2 以前のリリースで使用することができます。 PROMPT の属性を持つ Query 管理機能プログラムを、前のリリース用に保管することはできません。 SQL Query 管理機能の SQL 機能への変換を使用して、Query を SQL Query に変換する必要があります。その後で前のリリース用にこれを保管することができるようになります。 Query は、正常に実行するために変更しなければならないことがあります。

第 12 章 Query 管理機能での Query for iSeries 定義情報の使用

Query for iSeries 定義には、次のものに共通する機能のための仕様が入っています。

- Query for iSeries
- Query 管理機能 CPI Query

DB2 UDB for iSeries Query 管理機能では、Query for iSeries 定義 (QRYDFN) オブジェクトに保管されているこの情報を使用して、報告書を作成することができます。この章では、QRYDFN オブジェクトに入っている情報の DB2 UDB for iSeries Query 管理機能による使用を制限する方法、および、可能な最良の結果を得るための方法について説明します。

注: Query for iSeries QRYDFN オブジェクトの使用に関する詳細については、 176 ページの『Query 管理機能のその他のヒントおよび手法』を参照してください。

Query 管理機能は、Query for iSeries が作成した QRYDFN オブジェクトから、Query を実行するための情報、および報告書をフォーマット設定するための情報を取り出すことができます。 Query 管理機能プログラム (QMQRY) オブジェクトおよび書式 (QMFORM) オブジェクトへの変換は必要ありません。ユーザー作成のプログラムからこの機能を利用するための情報については、 CPI START コマンドの DSQSCNVT パラメーターを参照してください。対話式または CL プログラムからこの機能を利用するための情報については、次の CL コマンドの ALWQRYDFN パラメーターを参照してください。

- STRQMPRC - プロシージャー (保管された CPI コマンド手順) を実行する
- STRQMQRY - Query を実行して、データを保管するか、または報告書をフォーマット設定する
- RTVQMQRY - 編集可能な Query 管理機能プログラムのソースを取り出す
- RTVQMFORM - 編集可能な Query 管理機能書式のソースを取り出す

| QRYDFN オブジェクト内に指定および保管できる機能には、Query 管理機能がサポートする機能に変換できないものや、正しく変換できないものがあります。変換に関する問題が起こった場合、SELECT ステートメントの長さが 32KB を超えたため Query の情報を引き出して使用することができない場合を除いて、Query 管理機能は情報を引き出して使用し、実施した処置 (切り捨てなど) に関して警告を出しません。ANZQRY コマンドは、変換問題を処理する方法を示唆するメッセージおよびオンライン・ヘルプ情報の書式を分析します。

変換上の問題が存在しない場合でも、情報を引き出して作成した出力は予想と異なることがあり、受け入れられないこともあります。これは、指定できない機能に対して Query 管理機能が異なるデフォルトを与えたり、正常に変換された選択項目を異なった方法で使用するからです。次のセクションでは、QRYDFN オブジェクトに保管された情報を使用して最良の結果を得る方法や、Query 管理機能の出力と Query for iSeries からの出力を比較した時に予想される相違点について詳細に述べています。

QRYDFN オブジェクトに保管された情報では、 Query 管理機能の機能に対して完全にアクセスすることができず、また QRYDFN オブジェクトから情報を取り出すことのほうが、 Query 管理機能オブジェクトから情報を取り出すことよりも効率的でないため、多くのユーザーは QRYDFN オブジェクトを、対応する QMQRY および QMFORM オブジェクトに変換することを選びます。 QRYDFN オブジェクトから情報を検索する (エクスポートする) 方法、およびそれを使用して Query 管理機能オブジェクトを作成する (インポートする) 方法については、 204 ページの『QRYDFN オブジェクトからの DB2 UDB for iSeries Query 管理機能オブジェクトの作成』を参照してください。

Query 管理機能での QRYDFN の変換

コマンド処理のために Query 管理機能プログラム (QMQRY) オブジェクトまたは Query 管理機能書式 (QMFORM) オブジェクトが必要になった場合、DB2 UDB for iSeries Query 管理機能は、ライブラリーまたはライブラリー・リストの中から、指定された名前と一致する名前のオブジェクトを探します。 DB2 UDB for iSeries Query 管理機能に対して、この探索をスキップするように強制することもできますし、ライブラリーまたはライブラリー・リストから指定の名前を持つ Query for iSeries 定義 (QRYDFN) を探索するように強制することもできます。探索に合致する QRYDFN オブジェクトが見つかると、処理に必要な Query 情報または書式情報はこのオブジェクトから引き出され、このことが起こったことを示すメッセージおよびコードが戻されます。この情報を引き出す方法の詳細については、この章の他のトピックを参照してください。

DB2 UDB for iSeries Query 管理機能は、情報の取り出し過程で何らかの問題が起きたかどうかには関係なく、 QRYDFN オブジェクトからの情報を使用します。 QRYDFN オブジェクトが使用された時には、欠陥についてのまたは機能上の相違点に関するメッセージは生成されません。

取り出した情報は、要求が完了した時に破棄されます。情報を DB2 UDB for iSeries Query 管理機能オブジェクトに永続的に変換するためには、 QRYDFN オブジェクトから Query または書式のソースを取り出し、それを使用してそのタイプの DB2 UDB for iSeries Query 管理機能オブジェクトを作成します。

QRYDFN オブジェクトへの DB2 UDB for iSeries Query 管理機能の適用

DB2 UDB for iSeries Query 管理機能は、通常、DB2 UDB for iSeries Query 管理機能オブジェクトからの情報だけを使用します。しかし、DB2 UDB for iSeries Query 管理機能書式または Query 情報が使用できない場合は、 DB2 UDB for iSeries Query 管理機能に Query for iSeries の情報を使用するように要求することができます。また、DB2 UDB for iSeries Query 管理機能書式または Query 情報は使用しないようにすることもできます。

START コマンドに次のどちらかを指定します。

- DSQSCNVT=YES
- DSQSCNVT=ONLY

次の CL コマンドを使用する場合、

- STRQMPRC (Query 管理機能プロシージャーの開始)
- STRQMQRY (Query 管理機能 QUERY の開始)
- RTVQMQRY (Query 管理機能 QUERY の検索)
- RTVQMFORM (Query 管理機能書式の検索)

ALWQRYDFN(*YES) または ALWQRYDFN(*ONLY) を指定して、DSQSCNVT 値をセットします。

注: DSQSCNVT の値が *ONLY である時は、 Query 管理機能は QRYDFN オブジェクトだけを探すことによって、 QMQRY または QMFORM キーワードに指定された名前を解決します。

次に示す CPI コマンドをプログラム (START コマンドをコーディング) またはプロシージャー (STRQMPRC コマンドをコーディング) でコーディングし、直接 QRYDFN オブジェクトに適用することができます。

- RUN QUERY *myqrydfn*
- RUN QUERY *myqrydfn* (FORM=*myqrydfn*

- RUN QUERY *myqrydfn* (FORM=*myqrydfn2*
- RUN QUERY *myqmqry* (FORM=*myqrydfn*
- RUN QUERY *myqrydfn* (FORM=*myqmform*
- PRINT REPORT (FORM=*myqrydfn*
- PRINT QUERY *myqrydfn*
- PRINT FORM *myqrydfn*
- EXPORT QUERY *myqrydfn*
- EXPORT FORM *myqrydfn*

注: *myqrydfn* および *myqrydfn2* は、 DSQSCNVT 値が ONLY ではなく YES である場合、使用するオブジェクトを探索する時に Query 管理機能が同じ名前の Query オブジェクトまたは書式オブジェクトを見つけないように、固有の名前でなければなりません。

DB2 UDB for iSeries Query 管理機能の処理中に Query for iSeries 定義を使用したくない場合は、 DB2 UDB for iSeries Query 管理機能がデフォルトを使用できるようにしてください。 DB2 UDB for iSeries Query 管理機能のデフォルトは、START コマンドでは DSQSCNVT=NO、また、STRQMPRC、STRQMQRY、RTVQMQRY、または RTVQMFORM CL コマンドでは ALWQRYDFN (*NO) です。あるいは、DB2 UDB for iSeries Query 管理機能が QRYDFN オブジェクトを使用しないようにするために、Query for iSeries 定義が入っているライブラリーを、DB2 UDB for iSeries Query 管理機能の検索するライブラリーまたはライブラリー・リストから除外する方法も使用できます。

システム/36 環境で作成された Query に、DB2 UDB for iSeries Query 管理機能を直接適用することはできません。ただし、システム/36 Query 変換 (CVTS36QRY) コマンドを使用して、システム/36 Query を Query for iSeries 定義に変換することはできます。その後で、システム/36 Query から変換された QRYDFN オブジェクトから、DB2 UDB for iSeries Query 管理機能情報を取り出すことができます。

Query 管理機能での QRYDFN 変換に関する考慮事項

QRYDFN に指定されている選択項目のすべてを完全に変換することは、不可能であると考えられます。Query for iSeries 機能の中には、DB2 UDB for iSeries Query 管理機能の報告書や Query で使用できないものもあり、あるいは、似ていても、変換すれば脱落やゆがみが起きるものもあります。QRYDFN から派生した情報は、QMQRY または QMFORM として使用できないことがあります。作成した報告書またはデータ・レコード出力は、明白な欠陥がある場合もあり、あるいは Query for iSeries によって作成されたものとは相違がありすぎて、同じ目的には使用できない場合もあります。(STRQMQRY コマンドを RUNQRY コマンドの代わりに使用した結果が受け入れられない可能性のある状況のリストについては、206 ページの『Query 管理機能での RUNQRY コマンドに代わる STRQMQRY コマンドの使用』を参照してください。) しかし、Query for iSeries 定義を処理して簡単な調整を行い、受け入れ不能な相違点を除去することもできます。

Query 管理機能での報告書の相違点

次に示す図は、同一の Query for iSeries 定義からの情報を使用して Query for iSeries が作成した印刷報告書と、 Query 管理機能が作成した印刷報告書とを対比した報告書ページの例です。この定義は、情報を引き出して使用した時に起こる可能性があることを示すために選び出しただけであって、多くの Query for iSeries 定義で必ずしもこのようなことが起こるわけではありません。

```
/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥¥/¥/¥/¥
  08/11/90  12:25:21                                 ページ 1

DEPT           SALARY        COMM      Commission
                                       as percent of salary
                                       ____________________
  10        19,260.25         .00                 .00
            20,010.00         .00                 .00
            21,234.00         .00                 .00
            22,959.20         .00                 .00

   Dept 10 summary information
   最小値  19,260.25          .00
   最大値                     .00

  15       12,258.50       110.10                 .01
           12,508.20       206.60                 .02
           16,502.83     1,152.00                 .07
           20,659.80          .00                 .00

    Dept 15 summary information
    最小値  12,258.50         .00
    最大値               1,152.00

    Year 1988 - salary and commission survey
    最小値  12,258.50         .00
    最大値               1,152.00

* * * 報告書の終わり * * *

/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥
```

図 55. 調整前の Query for iSeries の出力

```
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\\/\/\/\/\/\/\/\/\/\/\
                                                    Commission
                                                as percent of salary
          MIN           MIN       MAX
 DEPT     SALARY        COMM      COMM                ______________
------    --------      -------    ------  --------------------------
  10      19,260.25      .00       .00      .000000000000000000000000
          20,010.00      .00       .00      .000000000000000000000000
          21,234.00      .00       .00      .000000000000000000000000
          22,959.20      .00       .00      .000000000000000000000000
        ------------ --------  --------
Dept 10
          19,260.25      .00       .00

   15     12,258.50    110.10    110.10    .008981523024839907003303 8
          12,508.20    206.60    206.60    .016517164739930605522777 0
          16,502.83  1,152.00  1,152.00    .069806208995669227641562 0
          20,659.80       .00       .00    .000000000000000000000000 0
        ------------ --------- ---------
Dept 15
          12,258.50       .00   1,152.00

        ============ ========= ==========
Year 19
          12,258.50       .00   1,152.00


          Page    1        *************************  IBM
08/11/90  11:58:42                                            1
\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
```

図 56. 調整前の Query 管理機能の出力

次に示す図は、QRYDFN オブジェクトに以下のような小さな変更を行った後の Query for iSeries の印刷出力と、 Query 管理機能の印刷出力とを対比した報告書ページの例です。

- 最初の列の前のスペースを 3 に一時変更した
- 結果フィールドのサイズに長さ 2 および小数点以下の桁数 2 を指定した
- 列の見出しテキストから下線を除去した
- 切れ目テキストおよび最終テキストを圧縮した
- ページ・フッター・テキストを圧縮し、&PAGE 変数を除去した

```
/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥¥/¥/¥/¥
  08/11/90  12:25:21                    ページ 1

 DEPT          SALARY         COMM      Commission
                                     as percent of salary
   10      19,260.25           .00               .00
           20,010.00           .00               .00
           21,234.00           .00               .00
           22,959.20           .00               .00

   Dept 10 :
   最小値  19,260.25           .00
   最大値                      .00

  15       12,258.50        110.10               .01
           12,508.20        206.60               .02
           16,502.83      1,152.00               .07
           20,659.80           .00               .00

    Dept 15 :
    最小値  12,258.50          .00
    最大値                1,152.00

    Year 1988 :
    最小値  12,258.50          .00
    最大値                1,152.00

* * * 報告書の終わり * * *


/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥/¥
```

図 57. 調整後の Query for iSeries の出力

```
\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\

             MIN           MIN          MAX        Commission
   Dept     SALARY         COMM         COMM    as percent of salary
-------    ---------    ---------   ----------  --------------------
     10    19,260.25         .00          .00                    .00
           20,010.00         .00          .00                    .00
           21,234.00         .00          .00                    .00
           22,959.20         .00          .00                    .00
          -----------   ---------  -----------
Dept 10:
            19,260.25        .00          .00

     15     12,258.50     110.10       110.10                   .01
            12,508.20     206.60       206.60                   .02
            16,502.83   1,152.00     1,152.00                   .07
            20,659.80        .00          .00                   .00
           -----------  ---------   -----------
Dept 15:
            12,258.50        .00     1,152.00

           ===========  =========  ============
Year 1988:
            12,258.50        .00     1,152.00


08/11/90 12:25:04                                                1
/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
```

図 58. 調整後の Query 管理機能の出力

DB2 UDB for iSeries Query 管理機能を使用して定期的に特定の Query を実行する場合は、満足できる結果を得るために、次のことを行ってください。

1. Query 分析 (ANZQRY) コマンドを使用して QRYDFN を分析し、作成されるすべての診断メッセージを読み取ります。 ANZQRY 診断テキスト警告に適切に応答します。この警告は Query 管理機能プログラムの開始 (STRQMQRY) コマンドを使用する前に取る必要のある処置に関するものです。
2. STRQMQRY コマンドを使用して、出力を入念に調べてください。未診断障害を見つけたり、ANZQRY 診断メッセージに無視できるものがある場合があります。
3. Query 処理 (WRKQRY) コマンドを使用して、QRYDFN オブジェクトを表示します。定義画面を検査して、 201 ページの『Query 管理機能での QRYDFN オプションの指針の適用』に示してある互換性の指針から逸脱しているものがないか調べてください。*変更* オプションを使用して調整し、出力を改良してください。
4. Query for iSeries と DB2 UDB for iSeries Query 管理機能との相違点のリスト (203 ページの『Query for iSeries と DB2 UDB for iSeries Query 管理機能の相違点』) を参照して、適切な処置をとってください。

Query 管理機能での QRYDFN の分析

DB2 UDB for iSeries Query 管理機能で使用するための情報を取り出す時に起こり得る問題を見つけるためには、 Query 分析 (ANZQRY) コマンドを使用して、Query for iSeries 定義を検査してください。 ANZQRY コマンドは、分析した QRYDFN オブジェクトから取り出される情報を Query for iSeries および DB2 UDB for iSeries Query 管理機能で使用する場合の潜在的な相違点について、詳しく示す診断メッセージを戻します。

完了メッセージは、最高レベルの重大度の潜在的問題が見つかったことを示します。重大度コードが 10 より大きい場合、メッセージ・ヘルプには、メッセージで示されているシステム処置を含め、見過ごしやすく重大な結果となる可能性のある問題についての警告が含まれていることがあります。

低いレベルの重大度コードであっても、派生した情報の使用時に重大な問題が生じないことを意味するとは限らず、高レベルの重大度コードが QRYDFN を使用しなければならないことを意味するものでもありません。 ANZQRY コマンドを使用する時には、次のことを考慮してください。

- 分析処理は指定ファイル内の情報を検査しますが、データベース・ファイルの一時変更については有効なものでも無視します。 QRYDFN オブジェクトに保管されている情報が、ファイル定義で見つかった情報と互換性があるかどうかについての検査は行われません。
- 分析処理は、SELECT ステートメントが大きくなりすぎるなどの、変換中に起こる可能性のあるエラーを予測しません。
- 分析処理は、実行中に起こる可能性のある次のようなエラーを予測しません。
  - 構造化照会言語 (SQL) の構文エラー (受け入れ不能な式または 10 進数の使用)
  - SQL データ・エラー (フィールドの脱落、不一致タイプの比較)
  - 大きすぎるフォーマット設定済み報告書の幅
  - 指定の使用目的または編集に不適当なフィールド・データ・タイプ
- 分析処理は、Query for iSeries と DB2 UDB for iSeries Query 管理機能との間の、次のような相違は無視します。
  - 変換可能な最終合計テキストの脱落
  - 列見出し行の先行ブランクの脱落
  - 長すぎる列見出しの位置合わせの違い
  - 列見出し行の下線文字での中断
  - 合計の前のスペースへの切れ目テキストまたは最終テキストの非拡張
  - 最後のデータ列を超える切れ目テキストまたは最終テキストの非拡張
  - 同一行上の合計タイプの相違
  - 結果フィールド・サイズのデフォルトの相違
  - 結果位取りおよび精度の計算のアルゴリズムの相違
  - 特定タイプのデータに関する計算による値の相違
  - 2 バイト文字セット (DBCS) 文字データ比較の処理の相違
  - 固有の形式の日付処理の相違 (DB2 UDB for iSeries Query 管理機能は、ファイル定義で指定された *MDY(2 桁の年) などの OS/400 の日付形式は無視し、SAA 形式の 1 つを使用する)
- 重大度コードでは、分析処理で生成されたメッセージの数は考慮に入れません。診断された問題自体が実際には重大度コードが示すほど重大ではないこともあるため、誤解されることもあります。たとえば、10 進データ・エラーを無視しても、そのことがある特定の Query にとっては問題ではないとしても、数値フィールドのない Query に対してさえも、重大度 30 の問題であるとして診断されています。

Query 管理機能での出力の検査

出力を検査する以外には、障害および受け入れ不能な相違点を検出する方法がない場合があります。ANZQRY によって診断した問題の実際の重大度を評価する方法についても同じです。比較可能なコマンド出力を得るには、DB2 UDB for iSeries Query 管理機能では STRQMQRY コマンドを、 Query for iSeries では RUNQRY コマンドを使用してください。

派生した Query を実行する時には、次のような相違点がないかどうかを調べてください。

- *LAST メンバーまたは名前で指定されたメンバーから生じたものでないレコード
- 不一致レコードの省略
- 列の脱落、または列順序の相違
- 余分な合計機能のための列の追加
- レコード順序の相違、または選択項目セットの相違
- 結果フィールド・データ列の長さまたは精度の相違
- 結果フィールドの値の相違、または予測しない数値オーバーフロー
- 合計または平均の値の相違、または予測しない数値オーバーフロー
- 数値計算のゼロによる除算エラー

派生した書式を使用して報告書を表示または印刷する時には、次のような相違点がないかどうかを調べてください。

- テキストの切り捨て
- 未解決のテキスト挿入変数
- 編集の相違
- 報告書の列の幅の相違
- 見出しまたはフッターの位置合わせの相違
- 行の折り返しなし
- カバー・ページなし
- ページ見出しにページ番号または日付および時刻の情報なし

Query 管理機能での QRYDFN オプションの指針の適用

DB2 UDB for iSeries Query 管理機能で使用する目的で Query for iSeries 定義を作成または変更する時は、次の定義画面の使用法の指針に従うと、起こりうる問題の多くを回避することができます。新しい QRYDFN オブジェクトの作成、または既存のオブジェクトの変更を行うためには、WRKQRY コマンドを使用し、オプション・フィールドには DB2 UDB for iSeries Query 管理機能との互換性を保つために推奨されている値を入れてください。 DB2 UDB for iSeries Query 管理機能で使用するために QRYDFN オブジェクトに次のオプションの値を指定する場合は、下記の指針を使用してください。

- ファイル選択項目の指定
  - 単一形式のファイルだけを選択する
  - *FIRST メンバーだけを選択する
  - 結合のタイプを指定する
  - 一致したレコード結合だけを使用する
- 結果フィールドの定義
  - 式に除算が含まれる場合サイズの一時変更を使用する
  - サイズの一時変更は数値列のフォーマット設定の制御のためだけに使用する
  - 該当する行でテキストを分割したい場合だけ、列見出しに下線文字を使用する
  - データ位置合わせに依存する列見出しを使用しない
  - 列見出しの中で複数行の数値を使用しない
  - 列見出し区切り文字のための行を使用しない
  - 文字フィールドのサイズを小さくするためには SUBSTR (フォーマット設定の一時変更ではない) を使用する

- フィールドの選択および順序付け
  - 特定のフィールドを選択する
  - 255 個を超えるフィールドを選択しない
- レコードの選択
  - || または SUBSTR を使用して定義された結果フィールドのテストに LIKE を使わない
  - グローバル変数が必要な箇所では従属値の構文を使用する
- ソート・フィールドの選択
  - EBCDIC 順序付けが受け入れ可能でない限り、文字フィールドのソートは行わない
- 照合順序の選択
  - EBCDIC 順序を選択する
- 報告書列のフォーマット設定の指定
  - 合計の左側に切れ目テキストまたは最終テキストのためのスペースを残す
  - ファイル・フィールドの長さを一時変更する場合、列見出しを一時変更する
  - 該当する行でテキストを分割したい場合だけ、列見出しに下線文字を使用する
  - データ位置合わせに依存する列見出しを使用しない
  - 列見出しの中で複数行の数値を使用しない
  - 列見出し区切り文字のための行を使用しない
  - 切れ目フィールドを省略しない
  - 数値ファイルのフィールド・サイズを一時変更する場合には、編集を一時変更する
- 数値フィールド編集の定義
  - 編集コードまたは数値編集選択項目を使用する
- 編集コードの指定
  - 編集コードは J または M だけを使用する
  - アスタリスク充てんオプションに修飾子を使用しない
  - 編集コードに M を指定した浮動通貨記号に修飾子を使用しない
- 数値フィールド編集の記述
  - Query 管理機能編集コードに変換することができる記述を使用する
  - ゼロの値の抑止を要求しない
  - アスタリスク充てんを要求しない
  - 単一の先行ゼロを要求しない
- 報告書合計機能の選択
  - 余分な機能のための列を追加したくない場合には、1 フィールド当たり複数の合計機能を要求しない
  - その機能のための列を追加したくない場合には、切れ目フィールドに合計機能を要求しない
- 報告書の切れ目の定義
  - 明細レコードを省略したい場合、結果フィールドに対して分割を行わない
  - 明細レコードを省略したい場合、1 つのレベルだけを定義する
- 報告書の切れ目のフォーマット設定
  - 最初の報告書フィールドに合計機能がある場合、切れ目テキストを使用しない
  - 切れ目テキスト変数に対して報告書フィールド名を使用する
  - *最終合計* デフォルトを置き換えるための最終テキストを定義する

  - 他のレベルを定義してある時に明細レコードを省略したい場合、レベル 0 の合計を省略する
- 出力タイプおよび出力書式の選択
  - 合計だけのデータベース・ファイル出力を作成する Query を定義しない
  - CL コマンドまたはプロシージャーに実行時の装置オプションを指定する
  - 必要ならば、行送りを使用する
  - 出力が最終合計だけの場合を除き、カバー・ページは定義しない
  - 行折り返しを使用しない
  - ページ見出しテキストまたはフッター・テキストとして 55 文字を超える文字を使用しない
  - ページ・テキスト変数に対して報告書フィールド名を使用する
- 処理オプションの指定
  - 数値オーバーフローの切り捨てを要求しない
  - 10 進データ・エラーの無視を要求しない
  - 置換文字の無視以外を要求しない

Query for iSeries と DB2 UDB for iSeries Query 管理機能の相違点

Query for iSeries と Query 管理機能の相違点について知っておくことによって、 Query for iSeries 定義オブジェクトから派生した情報からより良い結果を得ることができる場合があります。

- Query 管理機能が使用するデフォルト印刷書式幅は Query for iSeries が使用するものより小さい幅になります。Query for iSeries が保持していた、ある 1 つの印刷書式幅に基づく報告書に関して並行報告書を作成することができます。
- 他の Query によって作成されたファイル用に定義された Query は、これら他の Query から派生した Query 管理機能書式情報によって作成されたファイルでは使用できない場合があります。このことは、結果フィールドが出力ファイル定義に含まれる場合に特に起こりやすくなります。
- 認識されない小数点区切り文字が式または値に含まれている場合、Query for iSeries は定義を実行することができません。しかし、有効な小数点区切り文字が QDECFMT システム値によって指示される区切り文字に変換されるため、Query 管理機能はそのような定義を実行することができます。
- Query 管理機能データベース処理では、位取りなしの値の除算後、小数点以下の桁は切り捨てられます。たとえば、2/3 の計算結果は 0 です。このため、変換処理では式の中の +、-、/、*、または (が後に続く各数値定数には、必ず小数点区切り文字が含まれるようにします。その結果、式の列のサイズは小数点以下が 15 桁になります。位取りなしの数値フィールド (定数なし) の除算を行う式では、おそらく予想外の結果になります。数値結果フィールド名を SUBSTR 関数の長さまたはオフセットとして、あるいは DATE 関数の引き数として使用すると、エラーを起こす場合があります。
- SQL が正整数であるとしている個所に、小数点で区切られた結果フィールドの名前または番号を使用すると、実行を妨げるエラーが起こる場合があります。
- Query 管理機能データベース処理の場合の 2 バイト文字セット (DBCS) データの規則は、Query for iSeries に適用されるものと異なります。 DBCS 混用ストリング以外のものをテストするために、連結ストリングを使用してはなりません。
- 長すぎる書式テキストは切り捨てられるため、変換処理ではブランクのストリングを圧縮します。Query 管理機能は列および合計値配置によって決められている限度を超えるテキストの拡張を許していないため、テキストは失われたままです。
- Query for iSeries は、日付、時刻、およびタイム・スタンプのフィールドに指定された任意の固有のフォーマット設定を使用しますが、Query 管理機能では、代わりに類似した SQL 形式を使用します。

- 合計のみの出力の作成を目的とする QRYDFN オブジェクトが適切に定義されている場合、 Query 管理機能は Query 情報を、GROUP BY 文節のある SQL SELECT ステートメントおよび列または列関数のグループ化を含むフィールド選択リストに変換します (報告書の切れ目制御フィールドおよび選択された合計機能に対し)。報告書の制御の切れ目が定義されていなければ、列関数だけが使用されます。
  QRYDFN オブジェクトがこのような変換に不適切に定義されていれば、明細レコードは省略されません。次のいずれかに該当する場合、SQL 合計変換は実行を妨げるエラーを起こします。
  - ファイル・フィールド参照を伴わない結果フィールドに対する COUNT 以外の合計機能
  - SQL で許される幅より広い文字フィールド用の MIN または MAX
  - SQL で許される幅より広い、切れ目フィールドの合計サイズ
  - SQL で許される幅より広い、切れ目列と合計列の合計サイズ
- Query 管理機能からの合計専用出力を、以前の Query for iSeries が同じ QRYDFN オブジェクトを実行して作成したファイルに追加することはできません。これは、Query for iSeries はレベルおよびオーバーフローのフィードバック情報を入れるための余分なフィールドを含めてファイル形式を生成するためで、 Query 管理機能はそのような情報を提供せず、そのような情報のためのスペースも用意しません。
- ジョブの CCSID ではなく、QRYDFN の CCSID が検出されたオブジェクトの CCSID になります。

注: 変換の詳細については、 209 ページの『Query 管理機能での変換の詳細』を参照してください。

QRYDFN オブジェクトからの DB2 UDB for iSeries Query 管理機能オブジェクトの作成

以下のステップは、QRYDFN を永続的に DB2 UDB for iSeries Query 管理機能の書式オブジェクトおよび DB2 UDB for iSeries Query 管理機能プログラム・オブジェクトに変換する代表的な方法です。

1. 選択した QRYDFN に対して ANZQRY コマンドを実行します。メッセージが出され、オブジェクトを DB2 UDB for iSeries Query 管理機能オブジェクトに変換する前に、そのオブジェクトを調整するように勧告されます。
2. 1 つは外部化された Query 用、もう 1 つは外部化された書式用に別々の原始ファイル・メンバーを作成します。
3. 参照されるファイル定義に何らかの変更がある場合、WRKQRY コマンドを使用して、変更およびその解決された QRYDFN オブジェクトの保管を行います。このコマンドを使用して、変換を改善するためにオブジェクトを変更したり、あるいは、 Query for iSeries で許容されてはいるがサポートはされていない機能を追加することもできます。
4. CL コマンドまたは DB2 UDB for iSeries Query 管理機能コマンドを使用して (たとえば、RTVQMQRY または EXPORT QUERY)、使用できる情報を抽出します。
5. Query for iSeries プロンプト・インターフェースからは使用できない機能を追加したい場合には、外部化されたオブジェクトを編集します。
6. CL コマンドまたは DB2 UDB for iSeries Query 管理機能コマンド (たとえば、 CRTQMQRY または IMPORT QUERY) を使用して DB2 UDB for iSeries Query 管理機能オブジェクトを作成します。

205 ページの図 59 は、 Query 管理機能プログラム・オブジェクトまたは書式オブジェクトとして使用するために、 Query for iSeries QRYDFN がどのようにして変換されるかを示したものです。

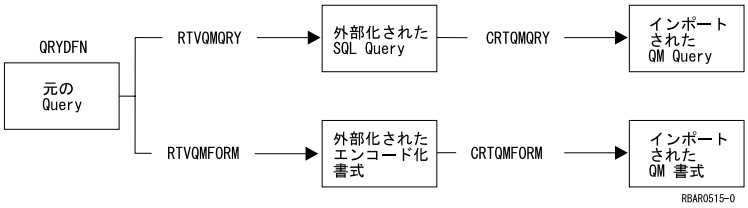


図 59. 変換のデータ・フロー

注:

1. QRYDFN オブジェクトから取り出した書式にはブランクの列項目が含まれていることがあり、その結果、QMFORM オブジェクトを作成するためにその書式を使用した時に警告が出されることがあります。これらの警告は無視することができます。
2. QRYDFN オブジェクトから取り出される Query ソースは、SQL SELECT ステートメントです。FROM 文節は、システム命名規則を使用します。

図 60 は、Query for iSeries QRYDFN オブジェクトを DB2 UDB for iSeries Query 管理機能オブジェクトに変換するための、 CL コマンドの使用の例です。

```
                                                    作業例...
                                                     オプション 2 からの出口で
                                                     MYLIB 内の QRY1 を
                                                     QTEMP 内に QRY1 として保管
                                                     (変更)
                                                           |
                                                           |
                                                           |
                                                           |
                                                           |
コマンドをタイプして、実行キーを押してください。                  |
> crtsrcpf qtemp/qqmqrysrc 91                             |
> crtsrcpf qtemp/qqmformsrc 162                           |
> wrkqry ___________________________________________________|
> rtvqmqry qtemp/qry1 qtemp/qqmqrysrc qry1
> RTVQMQRY コマンドは検索した情報を用いて完了
> strseu qtemp/qqmqrysrc qry1_____________________________
> crtqmqry qry1/qqmqrysrc qry1                            |
> rtvqmform qtemp/qry1 qtemp/qqmformsrc qry1              |
> RTVQMFORM command completed using derived information.  |
> crtqmform qry1 qtemp/qqmformsrc qry1                    |
===>                                                       |
                                                           |
                                                    編集例...
                                                     FROM 文節の library/object 名を
                                                     datase.object 名に変更
```

図 60. QRYDFN を変換するための CL コマンド手順の例

CL コマンドを使用して Query for iSeries QRYDFN オブジェクトを DB2 UDB for iSeries Query 管理機能オブジェクトに変換する別の例については、 179 ページの『Query 管理機能で QRYDFN オブジェクトを永続的に変換する CL プログラムの作成』を参照してください。

Query 管理機能での RUNQRY コマンドに代わる STRQMQRY コマンドの使用

Query 管理機能プログラムの開始 (STRQMQRY) コマンドおよび Query 実行 (RUNQRY) コマンドの両方を、事前に作成した QRYDFN オブジェクトの中の仕様に従って、フォーマット設定された報告書またはデータベース・ファイル出力を作成するために使用できます。次の例は、実行するオブジェクトが QRYDFN オブジェクトの時の、各コマンドについての最小パラメーター要件を示しています。

```
RUNQRY mylib/myqrydfn STRQMQRY mylib/myqrydfn  QMFORM(*QMQRY) ALWQRYDFN(*YES)
```

注: Query 管理機能は STRQMQRY コマンドと RUNQRY コマンドの両方に対し、 Query for iSeries *QRYDFN オブジェクトにおける代替照合順序表をサポートしています。

STRQMQRY コマンドを使用する理由には次のようなものがあります。

- STRQMQRY を使用すると、多くの報告書の体裁が改善されます。
- STRQMQRY は QRYDFN 情報の使用制限を軽減します。 RUNQRY と違い、STRQMQRY は次のものを使用した場合にも正常に完了します。
  - エラーがあるままで保管されていた QRYDFN オブジェクト
  - バッチ・モードでの従属 Query
  - 別々のオブジェクトからの書式情報および Query 情報
  - システムで定義されていないファイルの選択が含まれる、 QRYDFN オブジェクトから派生した書式情報
  - QDECFMT システム値によって示された小数点区切り文字以外の小数点区切り文字のある数値定数を含む、QRYDFN オブジェクトから派生した Query 情報
- STRQMQRY では、大きなファイルからの合計のみの情報の報告書作成がより早くなることがあります。このことを可能にするのは、検索した書式内の合計機能を使用するのではなく、検索した Query 内の SQL GROUP BY 文節または列関数を使用する変換モード (SQL 合計) です。ただし、より効率良く合計値を作成し、明細レコードを省略するこの方法は、特別な特性を持つ QRYDFN の場合にのみ使用可能です。 203 ページの『Query for iSeries と DB2 UDB for iSeries Query 管理機能の相違点』の項を参照してください。

RUNQRY の代わりに STRQMQRY を使用すると、得られる結果が受け入れ不可能であったり、あるいは適正な結果を得るために何らかの処置が必要になることがあります。まず、次の項目を検討してください。

- STRQMQRY を使用した時に mylib の中に myqrydfn という名前の QMQRY または QMFORM がある場合、ALWQRYDFN パラメーター値として *ONLY が指定されていなければ、Query の実行または報告書のフォーマット設定のための情報は、 QRYDFN オブジェクトからではなく、そのオブジェクトから取り出されます。 *YES を指定して、Query 管理機能のオブジェクトと一緒に QRYDFN オブジェクトを使用しなければならない場合、どちらかのオブジェクトの名前を変更するか、あるいはどちらかのオブジェクトを移動させなければなりません。
- Query データベースはメンバーの指定をサポートしないため、 *FIRST メンバーではなく所要のメンバーを使用するためには、ファイル一時変更を使用することが必要になることがあります (173 ページの『Query 管理機能での一時変更に関する考慮事項』の項を参照)。
- どちらのコマンドにも OUTFILE パラメーターがありますが、 STRQMQRY には *RUNOPT のデフォルトはありません。表示出力以外の出力が必要な場合、STRQMQRY には QRYDFN オブジェクトからのデフォルトが使用できないため、OUTFILE パラメーターおよびその関連パラメーターを指定しなければなりません。

- QRYDFN オブジェクトが従属 Query である場合、従属値を実行時にセットしなければなりません。RUNQRY コマンドについては、レコード選択 (RCDSLT) パラメーターを使用してください。このためには対話方式が必要で、従属値を指定するために*レコード選択* プロンプトが表示されます。STRQMQRY の場合、従属値の名前がグローバル変数に変換され、その変数を SETVAR パラメーターを用いてセットします。 SETVAR パラメーターを用いてすべてのグローバル変数に値を指定すると、その Query はバッチ・モードで実行することができます。対話モードでは、STRQMQRY は INQUIRY メッセージを使用して、 SETVAR パラメーターの使用によって前にセットされなかったグローバル変数についてプロンプトを出します。しかし、これらのメッセージには予期された値についての情報は含まれません。したがって、印刷された Query 定義を手元に用意しておく、あるいは、使用可能な選択項目を表示させてその中から所要の値を入力するようにというプロンプトを出す STRQMQRY ベースの簡単なコマンドを作成する、などの方法をとるのはよいことです。
- RUNQRY 処理では、Query の保管後にファイル定義に行われた変更を動的に調整します。 STRQMQRY 処理では調整しません。 STRQMQRY を使用するためには、その前に QRYDFN オブジェクトを保管し直さなければならないことがあります。
- SQL 変換モードが明細レコードの省略のために使用された場合、どの COUNT 合計も COUNT (*) に変換されます。ヌル値はカウントされます。ヌル値をカウントしたくない場合には、ヌル値を持つレコードを除外するレコード選択テスト (ISNOT NULL) を追加してください。
- 日付に、日付データ・タイプではなく文字列 (フィールド) を使用する場合は、以下の 2 つの間に相違が生じることがあります。
  - INSERT 処理を使用する SQL
  - 副選択を指定して INSERT 処理を使用する SQL

  SELECT を指定せずに INSERT 処理を使用すると、ユーザーのプロファイルの日付形式が使われます。SELECT を指定して INSERT 処理を使用すると、SQL は、その値を出力用と見なして、 SQL 形式 (ddmmyyyy) で Query 管理機能に値を提供します。

  これを避けるための理想的な方法は、日付には日付タイプの列を使用することです。文字タイプの列を使用しなければならない場合は、ユーザー・プロファイルで必ず ddmmyyyy 日付形式を指定してください。
- Query for iSeries の結果の許容度が、次のいずれかに依存している場合、Query 管理機能の結果は受け入れられないことになると思われます。項目ごとに、許容できない場合に取られるシステムの処置を括弧に入れて示しています。
  - フィールド選択の動的分析解決 (保管されていたフィールド・リストを使用する)
  - 複数形式のファイルからのデータ (出力を生成しない)
  - 1 次ファイルとの不一致結合 (一致結合を実行する)
  - 1 次ファイルとの一致結合 (一致結合を実行する)
  - 式の位取りおよび精度の制御 (デフォルトを使用して計算する)
  - 256 個以上のフィールド選択および余分な機能選択 (最初の 255 個の選択項目だけを使用する)
  - || または SUBSTR を伴う結果式の LIKE テスト (出力は作成されない)
  - 非 16 進数照合 (代替照合は行わない)
  - デフォルトの数値編集による小数点以下の桁数の一時変更 (小数点以下の桁数はデフォルト)
  - n 個の数字または文字をフォーマット設定するための長さの一時変更 (数字桁数または文字数はデフォルト)
  - デフォルトの数値編集による長さの一時変更 (デフォルトの長さを使用する)
  - デフォルトの列見出しによる長さの一時変更 (デフォルトの長さを使用する)

- デフォルトの小数点以下の桁数による編集の一時変更 (列の値の小数点以下の桁数は 0 として編集される)
- 日付および時刻の数値編集一時変更 (実行時のデフォルトを使用する)
- 数値編集の一時変更 (編集コード K を使用する)
- 同一の列のフィールドに対する複数の合計機能 (それぞれの合計機能ごとに別の列を追加する)
- 切れ目フィールド列の省略 (切れ目フィールド列は省略されない)
- 切れ目フィールドに対する合計機能 (合計値を保持するために独立した列を追加する)
- 最初または唯一の列に合計がある場合の非 SQL 合計のための切れ目テキスト (切れ目テキストは表示されない)
- 最初または唯一の列に合計がある場合の非 SQL 合計のための最終テキスト (最終テキストは表示されない)
- 複数レベルの切れ目合計のある合計のみの出力 (明細レコードは省略されない)
- 切れ目合計および最終合計のある合計のみの出力 (明細レコードは省略されない)
- 切れ目制御に使用される結果フィールドのある合計のみの出力 (明細レコードは省略されない)
- フィールド用に非ヌル値のカウントのある合計のみの出力 (明細レコードは省略されるが、すべてのレコードがカウントされる)
- 結果フィールド・リテラル用の COUNT 以外の列関数 (SQL 合計用に出力は作成されない)
- SQL の許容限度を超える広い文字フィールドに関する MAX 列関数 (SQL 合計用に出力は作成されない)
- SQL の許容限度を超える広い文字フィールドに関する MIN 列関数 (SQL 合計用に出力は作成されない)
- SQL の許容限度を超える広い GROUP BY 用の切れ目フィールドの合計サイズ (SQL 合計用に出力は作成されない)
- SQL の許容限度を超える広い切れ目列および合計列の合計サイズ (SQL 合計用に出力は作成されない)
- 行の折り返し (行の折り返しを行わず、並行印刷ファイルを作成する)
- カバー・ページ・テキスト (カバー・ページは印刷しない)
- 55 文字より広いページ・テキスト (ページ・テキストを切り捨てる)
- 数値オーバーフローの切り捨て (数値オーバーフローを丸める)
- 10 進データ・エラーの無視 (エラーを無視せず、データを訂正しない)
- 文字置換をエラーとして扱う (文字置換を無視する)
- 10 文字を超える名前の参照表を参照するために、オブジェクト名を使う (オブジェクト名はシステムの命名規則に合致していなければならない)

- 203 ページの『Query for iSeries と DB2 UDB for iSeries Query 管理機能の相違点』にリストされている Query for iSeries と DB2 UDB for iSeries Query 管理機能との相違点を考慮し、適切な処置をとることによって、よりよい結果を得ることができます。たとえば、デフォルトが異なっているため、使用する印刷用紙幅を Query for iSeries で使用するものと同じにすることができます。

Query 管理機能での変換の詳細

図 61 は、Query 定義仕様をプロンプトで要求するために使用する Query 処理画面と、 Query 管理機能プログラム・オブジェクトおよび Query 管理機能書式オブジェクトの対応するセクションとの間の関係を示しています。 DB2 UDB for iSeries Query 管理機能がサポートしている構造化照会言語 (SQL) の機能のうちのいくつかは、 図 61 に示されていません (サブリストおよびある種のスカラー関数など)。これらは、プロンプトによるインターフェースでは使用できませんが、SQL では使用できます。

Query 処理画面	**Query 管理機能オブジェクト**
ファイル選択項目の指定	
- ファイル、ライブラリー	SELECT FROM
- メンバー	適用されない
- 形式	適用されない
- ファイル ID	SELECT FROM
	SELECT リスト、ORDER BY
	(フィールド修飾子として使用)
結合タイプの指定	
- 結合のタイプ	適用されない
ファイル結合方法の指定	
- フィールド__テスト__フィールド	SELECT WHERE
結果フィールドの定義	
- フィールド__ (列見出しなしの場合)	列: 列見出し、幅
- 式__	SELECT リスト、WHERE
	(フィールドの置換)
- 列見出し__ (一時変更なしの場合)	列: 列見出し、幅
- 長さ__	列: 編集、幅
- 小数点以下の桁数	列: 編集、幅
フィールドの選択と順序付け	
- 順序__フィールド	SELECT リスト
レコードの選択	
- AND/OR__フィールド__テスト__値	SELECT WHERE
分類フィールドの選択	
- 分類パリティー__A/D__フィールド	SELECT ORDER BY
	SELECT GROUP BY
	SELECT HAVING
照合順序の選択	
- 照合順序オプション	適用されない
- 表、ライブラリー	適用されない
照合順序の定義	
- 順序__文字	適用されない
報告書の列のフォーマット設定の指定	
- フィールド__	
	列: 順序
	列: データ・タイプ
- 列の間隔__	列: 字下げ
- 列見出し__ (一時変更のみ)	列: 列見出し、幅
- 長さ__ (一時変更のみ)	列: 使用目的、編集、幅
(0) (OMIT)	
- 小数点以下の桁数__ (一時変更のみ)	列: 編集、幅
(#) (#)	
- 編集 (一時変更のみ)	列: 編集、幅
(変換不能な数値の編集) (K)	
数値フィールド編集の定義	
- 編集オプション	適用されない
数値フィールド編集の記述	
- 小数点など	列: 編集、幅
(変換可能なものの説明)	(対応するコード)

図 61. Query 処理画面と DB2 UDB for iSeries Query 管理機能オブジェクトの関係 (1/3)

Query 処理画面	Query 管理機能オブジェクト
日付 / 時刻フィールド編集の記述	
- 日付 / 時刻区切り記号	適用されない
編集コードの指定	
- 編集コード、任意指定修飾子	列: 編集、幅
(J)	(K)
(J$)	(D)
(M)	(L)
編集語の指定	
- 編集語	適用されない
- 合計用の編集語	適用されない
報告書合計機能の選択	
- オプション__フィールド	列: 使用目的、幅
(1=合計)	(SUM)
(2=平均)	(AVERAGE)
(3=最小)	(MINIMUM)
(4=最大)	(MAXIMUM)
(5=カウント)	(COUNT)
	(FIRST)
	(LAST)
報告書切れ目の定義	
- 切れ目レベル__分類パリティー__フィールド	列: 使用目的
(1-6)	(BREAK1-BREAK6)
報告書切れ目のフォーマット設定 (レベル 0)	
	最終 : 改ページ後最終テキスト
- 合計の抑止	最終: 最終合計表示行
(Y=Yes、 N=No (テキストがある場合))	(NONE、2)
	最終: フッターの前のブランク行数
- 切れ目テキスト	最終: 最終フッター・テキスト
(最初で唯一の行)	(行 1)
	(行 2-12)
報告書切れ目のフォーマット設定 (レベル 1-6)	
- 新しいページへスキップ	切れ目: 改ページ後見出し
	切れ目: 列見出しの繰り返し
	切れ目: 見出しの前のブランク行数
	切れ目: 見出しの後のブランク行数
	切れ目: 改ページ後フッター
	切れ目: フッターの前のブランク行数
	切れ目: フッターの後のブランク行数
- 合計の抑止	切れ目: 合計表示行
(Y=Yes、 N=No (テキストがある場合))	(NONE、2)
	切れ目: 切れ目見出しテキスト
- 切れ目テキスト	切れ目: 切れ目フッター・テキスト
(最初で唯一の行)	(行 1)
	(行 2-5)
出力タイプおよび出力書式の選択	
- 出力タイプ	適用されない
- 出力の形式	適用されない
- 行の折り返し	適用されない

図 61. Query 処理画面と DB2 UDB for iSeries Query 管理機能オブジェクトの関係 (2/3)

Query 処理画面		Query 管理機能オブジェクト
印刷装置出力の定義		
- 印刷装置		適用されない
- 用紙サイズ		適用されない
- 開始行		適用されない
- 終了行		適用されない
- 行間隔		オプション: 明細行の間隔
(1-3)	(1-3)	
	(4)	
		オプション: 切れ目列の一括表示
		オプション: 省略時の切れ目テキスト
		オプション: 列折り返し行の同一ページ表示
		オプション: 列見出し区切り記号
		オプション: 切れ目合計区切り記号
		オプション: 最終合計区切り記号
- 定義の印刷		適用されない
スプール出力の定義		
- 出力のスプール		適用されない
- 用紙タイプ		適用されない
- コピーの部数		適用されない
- 保留		適用されない
カバー・ページの指定		
- カバー・ページの印刷		適用されない
- カバー・ページ・テキスト (5 行まで)		適用されない
ページ見出しおよびページ・フッターの指定		
- 標準ページ見出しの印刷		
		ページ: 見出しの前のブランク行数
		ページ: 見出しの後のブランク行数
- ページ見出し		ページ: ページ見出しテキスト
(行 1-3)	(行 1-3)	
	(行 4-5)	
		ページ: フッターの前のブランク行数
		ページ: フッターの後のブランク行数
- ページ・フッター		ページ: ページ・フッター・テキスト
(最初で唯一の行)	(行 1)	
	(行 2-5)	
データベース・ファイル出力の定義		
- ファイル、ライブラリー、メンバー		適用されない
- ファイル内のデータ		適用されない
- 権限 (新しいファイルの場合)		適用されない
- テキスト (新しいファイルの場合)		適用されない
- 定義の印刷		適用されない
処理オプションの指定		
- 四捨五入の使用		適用されない
- 10 進データ・エラーの無視		適用されない

図 61. Query 処理画面と DB2 UDB for iSeries Query 管理機能オブジェクトの関係 (3/3)

変換の結果、以下の処置が取られることがありますが、これは予測できません。最良の結果を得るためには、これらの処置について理解しておく必要があります。

- 式またはテスト値の中の文字はストリング定数でない限り、すべて大文字に変換されます。これには区切り文字付き名前の文字も含まれます。
- フィールド名として使用される SQL 予約語は、派生した SELECT ステートメントの中では、引用符で囲まれます。
- 各結果フィールド名は、SELECT リストまたはレコード選択テストでは、対応する式で置き換えられます。結果フィールド名は他の結果フィールドの式の中で使用することができるため、置換は再帰的に行われます。 ORDER BY 文節の中の結果フィールド名は、列番号で置き換えられます。
- 結合テストが存在している場合、結合テストが WHERE 文節を開始するために使用される、AND 演算子を用いてレコード選択テストが追加されます。

- レコード選択テストでは、従属値はグローバル変数に変換されます。たとえば、従属値 :t01.″collection″ は、&T01_COLLECTION に変換されます。
- 有効な小数点区切り文字は、派生した SELECT ステートメントの中の数値定数の Query 小数形式 (QDECFMT) システム値で示される区切り文字に変換されます。
- QUERY 管理機能処理は、位取りのない値を除算した場合、小数点以下の桁を切り捨てます。たとえば、2 を 3 で割った場合の値は 0 になります。このため、派生した SELECT ステートメントの中の式では、 QDECFMT システム値によって示される小数点区切り文字が小数点区切り文字のない (かつ +、-、*、(、または / が続く) すべての数値の後に置かれます。
- 列が結果フィールドであるか、または列見出し一時変更がある場合を除き、合計機能なしの列では、*列見出し* フィールドはブランクのままです。見出しが定義されない場合には、フィールド名はデフォルトになります。
- 見出しテキストの中の下線文字は置き換え文字に置き換えられることはありません。 1_9_9_0 のような単一の行見出しは、変換によって変更されず、派生した書式情報が使用される時には、4 行の見出しが生成されます。

  次の処置が行われることもあります。
  - 列見出しがないことを示す *NONE は、単一の下線文字に変換されます。あるいは、合計機能列では、該当する値について Query for iSeries が使用する表題に変換されます。
  - 最後の非ブランク行を含めすべての行から、先行ブランクおよび後書きブランクを除去し、単一の下線文字で結果セグメントを結合することによって、列見出しストリングを作成します。
  - 合計機能を指定した列の見出しを作成する場合は、列見出しストリングを Query for iSeries が該当値に使用する表題に追加します。結果ストリングが 62 文字より長い場合には、ストリングは切り捨てられます。
  - 合計機能の表題に追加する見出しストリングが存在していない場合、フィールド名が使用されます。フィールド定義の一部としてそのフィールドに定義されている見出しは無視されます。
- 次のいずれかの場合、編集値の文字部分はブランクのままにされます。
  - その列には COUNT 合計値だけが入れられる場合。
  - サイズの一時変更がない場合。ただし、その列がサイズが指定されている結果フィールドである場合を除きます。
  - 一時変更編集が定義されておらず、列が結果フィールドではなく、数値の小数点以下の桁数の一時変更がある場合。
- 小数点以下の桁数の一時変更が、文字、日付、時刻、またはタイム・スタンプ・フィールドを指示している場合、DB2 UDB for iSeries Query 管理機能は、 *C* 編集コードを使用します。編集の一時変更に対して QUERY 管理機能 CPI Query 編集コードとより一致するものが判別できないか、あるいは一時変更が存在せず列が結果フィールド用である場合には、数値フィールドには *K* 編集コードが使用されます。
- *編集* オプションの選択項目で示された一時変更の編集タイプだけが検討されます。
- RPG 編集コードの編集に対するシステム・デフォルトの効力は無視されます。 J、通貨記号付きの J、および M の編集コードは、必ずそれぞれ K、D、および L に変換されます。
- 比較が行われる時、実際の編集に使用されない編集の記述選択項目 (たとえば、使用される通貨記号がない時の左側または右側の通貨記号) は無視されます。
- 文字部分がブランクまたは C の場合、*編集* 値の数値部分はブランクのままにされます。この値は、使用する小数点以下の桁数が、一時変更 (非ゼロ長一時変更) として保管されている、または結果フィールドの定義の一部として保管されている小数点以下の桁数から判別できない場合も、ブランクのままにされます。

- QRYDFN オブジェクトに保管されている情報が列に表示されるすべての幅の要件を指定していない場合には、列の幅の値はブランクのままにされます。列フォーマット設定の長さの一時変更は幅に影響しますが、それによって決まるのは、編集済みデータより幅が広いものが他に列内になく (見出しセグメントまたはカウント合計値 9,999,999 など)、一時変更の長さがデータベース処理によって想定されているデータ幅より小さい場合の、フォーマット設定された桁数または小数点以下の桁数だけです。
- ページ・テキスト、切れ目テキスト、および最終テキストは次のように調整されます。
  - 連続したブランクのストリングは 1 つのブランクに圧縮されます。
  - フィールド参照挿入変数は列参照変数に変換され、特殊変数 (&time、 &date、 &page など) はすべて大文字に変換されます。アンパーサンド (&) で始まるストリングは、 & の後の文字がブランクまたは数字ではなく、しかもそのストリングが次の文字の 1 つで終わっている場合に限り、テキスト挿入変数として認識されます。
    - ブランク
    - スラッシュ
    - コロン
    - ダッシュ
    - アンパーサンド
    - 下線
    - DBCS シフトアウト文字

    フィールド参照を列参照に変換する時点で、選択されたフィールドはすべて (報告書順で) 考慮されます。
  - 分析解決されたテキストが 55 文字を超える場合、最初の 56 桁の最初のブランクまたはアンパーサンドの位置で切り捨てられます。ブランクまたはアンパーサンドが見つからない場合、そのテキストは使用されません。
  - 切り捨てがクローズされているため、DBCS ストリングは開いたままとなります。
- Query 定義の CCSID、および式またはレコード選択の中のリテラルのフォーマット設定属性は、 DB2 UDB for iSeries Query 管理機能に渡されます。この情報は変換が実行されるジョブからのものではありません。

第 13 章 Query 管理機能での制御言語インターフェース

制御言語 (CL) を介して Query 管理機能の管理機能を使用し、報告書を生成するための単純なアプリケーションを作成することができます。コマンド、制御言語 (CL) プログラム、Query、および書式を定義することによって、意味のある報告書を生成するために必要な情報を求めるプロンプトをユーザーに対して出すことができます。

Query 管理機能での QMQRY オブジェクトおよび QMFORM オブジェクトの作成

Query (QMQRY オブジェクト) は、SQL ステートメントを使用して定義することができます。 図 62 は、SALARYQ2 という名前の印刷済み Query の例です。

```
                              DB2 Query 管理機能 OS/400

Query . . . . . . . . . . . . : SALARYQ2
  ライブラリー  . . . . . . . :   EXAMPLE
分類順序  . . . . . . . . . . : 分類順序テーブル
   分類順序テーブル . . . . . : QASCII
      ライブラリー  . . . . . :   QSYS
言語識別コード  . . . . . . . : ENU
テキスト  . . . . . . . . . . : Sales Query
SEQNBR |...+....1....+....2....+....3....+....4....+....5....+....6....
000001 SELECT DEPT,NAME,ID,JOB,YEARS,SALARY,COMM
000002        FROM TESTDATA/STAFF
000003        WHERE DEPT = &COND1 AND SALARY < &COND2
000004        ORDER BY DEPT,SALARY
                                * * * * *   ソースの終り    * * *
```

図 62. Query テストの SELECT ステートメント

注: ユーザー定義のソート・シーケンス表を使用するように *QMQRY オブジェクトを定義した後は、そのソート・シーケンス表への変更は、その Query が変更され、その Query を実行する時点でのみ有効になります。 *QMQRY オブジェクトに関連するソート・シーケンス表の内容を、 PRINT または EXPORT することはできません。

QMQRY オブジェクトの作成に関する詳細については、9 ページの『Query 管理機能での Query の作成』を参照してください。

書式 (QMFORM オブジェクト) を定義して、生成される報告書に含まれる情報を指定することができます。

QMFORM オブジェクトの作成に関する詳細については、61 ページの『Query 管理機能での書式の作成』を参照してください。

Query 管理機能での数字変換用の CL プログラムの例

図 63 の制御言語プログラムは、 図 62 に示した Query を使用しています。

```
                                    SEU SOURCE LISTING
ソース・ファイル . . . . . EXAMPLE/SOURCE
メンバー . . . . . . . . . CLPGM
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...
  100 PGM PARM(&VAR1 &VAR2 &VAR3 &VAR4)
  200 DCL &VAR1 *CHAR LEN(6)
  300 DCL &VAR2 *CHAR LEN(6)
  400 DCL &VAR3 *CHAR LEN(6)
  500 DCL &VAR4 *CHAR LEN(10)
  600 STRQMQRY QMQRY(EXAMPLE/SALARYQ2) QMFORM(EXAMPLE/&VAR4) +
  700          OUTPUT(&VAR3)                                 +
  800          SETVAR((COND1 &VAR1) (COND2 &VAR2))
  900 ENDPGM
                           * * * *  ソース仕様の終わり  * * * *
```

図 63. CL プログラムのソース・ファイル

注: STRQMQRY 内で、複数の変数と長い SQL ステートメントと一緒に STRQMQRY を使用すると、予期しない結果を招く場合があります。長い SQL ステートメントは、55 文字のセグメントに分割して、複数の変数に保管できます。しかし、これらのセグメントの切れ目は必ず文字でなければなりません。ステートメントをブランクで分割すると、CL コマンド・プロセッサーはブランクを切り捨てます。変数が連結されたときに、切り捨てられたスペースは置き換えられないので、 SQL ステートメントが失敗することがあります。ただし、CL コマンド・プロセスは、セグメントの先頭にあるブランクをそのままにします。

図 64 は、この CL プログラム例を実行するためのコマンドを示しています。

```
                                    SEU SOURCE LISTING
ソース・ファイル . . . . . EXAMPLE/SOURCE
メンバー . . . . . . . . .  DEPTREP1
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...
  200 CMD   PROMPT('部門報告書')
  300             PARM      KWD(VAR1) TYPE(*CHAR) LEN(6) RSTD(*YES) +
  301                         DFT(10) VALUES(10 20 30 40 50) +
  400                         PROMPT('報告書作成部門')
  500             PARM      KWD(VAR2) TYPE(*CHAR) LEN(6) DFT(100000) +
  600                         PROMPT('給与の上限')
  700             PARM      KWD(VAR3) TYPE(*CHAR) LEN(6) RSTD(*YES) +
  701                         DFT(*) VALUES(* *PRINT) PROMPT('出力 +
  800                         先 (*/*PRINT)')
  900             PARM      KWD(VAR4) TYPE(*CHAR) LEN(10) RSTD(*YES) +
  901                         DFT(SALARYF2) VALUES(BYDEPT BYDIVISION +
 1000                         SALARYF2) PROMPT('報告書名')
                           * * * *  ソース仕様の終わり  * * * *
```

図 64. CL コマンドのソース・ファイル

コマンド EXAMPLE/DEPTREP を入力し、プロンプトを得るための F4 キーを押した場合には、次の画面が表示されます。

```
                        部門報告書 (DEPTREP)

選択項目を入力して、実行キーを押してください。

報告書作成部門 . . . . . .  10          10, 20, 30, 40, 50
給与の上限 . . . . . .  100000      文字値
出力先 (*/*PRINT) . . . . .  *PRINT       *, *PRINT
報告書名 . . . . . .  SALARYF2     BYDEPT, BYDIVISION, SALARYF2

                                                                  終り
F3=終了    F4=プロンプト    F5=最新表示    F12=取り消し
F13=この画面の使用法       F24=キーの続き
```

このジョブのデフォルトを使用するためには、実行キーを押してください。

図 65 に示されている報告書には、部門 10 の給与額 (SALARY) および勤続年数 (YEARS) に関する部門平均も表示されます。

```
     DEPT  NAME              ID  JOB        YEARS      SALARY       COMM
---------  ---------  ---------  -----  ---------  -----------  ---------
      10   DANIELS         240   MGR           5    19,260.25        .00
           LU              210   MGR          10    20,010.00        .00
           JONES           260   MGR          12    21,234.00        .00
           MOLINARE        160   MGR           7    22,959.20        .00
                                                --------  -----------
                             Dept Avg:          9    20,865.86
```

図 65. CL プログラムの報告書の例

Query 管理機能での文字変数用の QMQRY オブジェクトおよび QMFORM オブジェクトの作成

Query (QMQRY オブジェクト) の定義は、数値変数用のオブジェクト定義と同じ方法で行いますが、1 つだけ違いがあります。文字変数には、さらに多くの引用符が必要です。 図 66 は、SALARYQ3 という名前の印刷された Query の例です。

```
                          DB2 Query 管理機能 OS/400

Query  . . . . . :   SALARYQ3
  ライブラリー . . :     EXAMPLE
テキスト . . . . . :   TEST QUERY
SEQNBR |...+....1....+....2....+....3....+....4....+....5....+....6....
000001 SELECT JOB,DEPT,NAME,ID,YEARS,SALARY,COMM
000002        FROM TESTDATA/STAFF
000003        WHERE JOB = &COND1 AND SALARY < &COND2
000004        ORDER BY JOB,SALARY
                              * * * * *  ソースの終り  * * *
```

図 66. Query テストの SELECT ステートメント

注: ユーザー定義のソート・シーケンス表を使用するように *QMQRY オブジェクトを定義した後は、そのソート・シーケンス表への変更は、その Query が変更され、その Query を実行する時点でのみ有効になります。 *QMQRY オブジェクトに関連するソート・シーケンス表の内容を、 PRINT または EXPORT することはできません。

QMQRY オブジェクトの作成に関する詳細については、9 ページの『Query 管理機能での Query の作成』を参照してください。

書式 (QMFORM オブジェクト) を定義して、生成される報告書に含まれる情報を指定することができます。

QMFORM オブジェクトの作成に関する詳細については、61 ページの『Query 管理機能での書式の作成』を参照してください。

Query 管理機能での文字変換用の CL プログラムの例

図 67 の制御言語プログラムは、図 66 に示した Query を使用しています。

```
|                                      SEU SOURCE LISTING
|   ソース・ファイル . . . . . EXAMPLE/SOURCE
|   メンバー . . . . . . . . . CLPGM
|   SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...
|     100 PGM PARM(&VAR1 &VAR2 &VAR3 &VAR4)
|     200 DCL &VAR1 *CHAR LEN(6)
|     300 DCL &VAR2 *CHAR LEN(6)
|     400 DCL &VAR3 *CHAR LEN(6)
|     500 DCL &VAR4 *CHAR LEN(10)
|     600 DCL &CHARJOB *CHAR LEN(10)
|     700 CHGVAR    VAR(&CHARJOB) VALUE('''' *TCAT &VAR1 *TCAT '''')
|     800 STRQMQRY QMQRY(EXAMPLE/SALARYQ3) QMFORM(EXAMPLE/&VAR4) +
|     900          OUTPUT(&VAR3)                                 +
|    1000          SETVAR((COND1 &CHARJOB) (COND2 &VAR2))
|    1100 ENDPGM
|                              * * * *  ソース仕様の終わり  * * * *
```

図 67. CL プログラムのソース・ファイル

図 68 は、この CL プログラム例を実行するためのコマンドを示しています。

```
|                                      SEU SOURCE LISTING
|    ソース・ファイル . . . . . EXAMPLE/SOURCE
|    メンバー . . . . . .  JOBREP1
|    SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...
|      200 CMD   PROMPT('ジョブ報告書')
|      300              PARM      KWD(VAR1) TYPE(*CHAR) LEN(6) RSTD(*YES) +
|      301                          DFT(MGR) VALUES(MGR SALES CLERK) +
|      400                          PROMPT('報告書作成ジョブ・カテゴリー')
|      500              PARM      KWD(VAR2) TYPE(*CHAR) LEN(6) DFT(100000) +
|      600                          PROMPT('給与の上限')
|      700              PARM      KWD(VAR3) TYPE(*CHAR) LEN(6) RSTD(*YES) +
|      701                          DFT(*) VALUES(* *PRINT) PROMPT('出力 +
|      800                          先 (*/*PRINT)')
|      900              PARM      KWD(VAR4) TYPE(*CHAR) LEN(10) RSTD(*YES) +
|      901                          DFT(SALARYF3) VALUES(BYJOB BYDIVISION  +
|     1000                          SALARYF3) PROMPT('報告書名')
|                                 * * * *  ソース仕様の終わり  * * * *
```

図 68. CL コマンドのソース・ファイル

コマンド EXAMPLE/DEPTREP を入力し、プロンプトを得るための F4 キーを押した場合には、次の画面が表示されます。

```
                          ジョブ報告書 (JOBREP)

選択項目を入力して、実行キーを押してください。

報告書ジョブ・カテゴリー   . . .   MGR          MGR, SALES, CLERK
給与の上限 . . . . . .  100000       文字値
出力先 (*/*PRINT) . . . . . .  *PRINT       *, *PRINT
報告書名 . . . . . .  SALARYF2     BYJOB, BYDIVISION, SALLARYF2

                                                                  終り
F3=終了    F4=プロンプト    F5=最新表示    F12=取り消し
F13=この画面の使用法       F24=キーの続き
```

このジョブのデフォルトを使用するためには、実行キーを押してください。

図 69 の報告書には、ジョブ MGR の給与額 (SALARY) と経験年数 (YEARS) に関する部門の値が表示されています。

```
                                                                    COMM
JOB       DEPT  NAME           ID   YEARS       SALARY      COMM   ---------
-----  -------  --------- -------  ------  -----------  --------        .00
MGR         38  MARENGHI       30       5    17,506.75       .00
            42  PLOTZ         100       7    18,352.80       .00
            20  SANDERS        10       7    18,357.50       .00
            66  LEA           270       9    18,555.50       .00
            10  DANIELS       240       5    19,260.25       .00
            84  QUILL         290      10    19,818.00       .00
            10  LU            210      10    20,010.00       .00
            15  HANES          50      10    20,659.80       .00
            51  FRAYE         140       6    21,150.00       .00
            10  JONES         260      12    21,234.00       .00
            10  MOLINARE      160       7    22,959.20       .00
                                     ------  -----------
                          Job Avg:       8    19,805.80
```

図 69. CL プログラムの報告書の例

付録 A. Query 管理機能での DBCS データ

この付録では、DB2 UDB for iSeries Query 管理機能における 2 バイト文字セット (DBCS) データの使用法について説明します。 DBCS データの使用法は、いくつかの点で 1 バイト文字セット (SBCS) データの使用法と異なります。 DBCS データを使用するには、DBCS を使用できるハードウェアおよびソフトウェアが必要です。

Query 管理機能における DBCS データとは

2 バイト文字セットとは、それぞれの文字が 2 バイトで表される文字のセットです。日本語、中国語、および韓国語などの言語では、1 バイト (256 のコード点) では表現しきれない多くの文字があり、2 バイト文字セットが必要となります。

各文字を表すために 1 バイトを使用する文字セットを、1 バイト文字セットと呼びます。英語、ドイツ語、およびフランス語などの言語は 1 バイト文字セットです。カタカナは、文字を内部的に 1 バイトで表すことができるため、 1 バイト文字セットで表すこともできます。

この付録で*混合* データと呼んでいる場合は、 1 バイト文字および 2 バイト文字の両方のデータのストリングが 1 つのデータ・フィールドに入っていることを意味します。

DBCS データには、*シフト文字付き DBCS* とグラフィック *DBCS* の 2 つのタイプがあります。

シフト文字付き DBCS データ:
シフト文字付き DBCS データとは、データベースに保管される場合に、そのデータの前に 1 バイトのシフトアウト (SO) 文字 (16 進 '0E')、後ろに 1 バイトのシフトイン (SI) 文字 (16 進 '0F') が付けられている DBCS データです。 SO 文字は DBCS データの始まりを示し、SI 文字は DBCS データの終わりを示します。データに SBCS と DBCS の両方が混用されている場合は、混用ストリング内の DBCS データには、前に SO 文字が、後に SI 文字が付きます。

グラフィック DBCS データ:
グラフィック DBCS データは、その前後に SO 文字および SI 文字を付けずにデータベースに保管される DBCS データです。グラフィック DBCS データは、固定長でも可変長でもかまいません。

DB2 UDB for iSeries は、SO 文字および SI 文字を含む、バイト 数として指定された長さで、シフト文字付き DBCS データを維持します。データベースに記憶できるシフト文字付き DBCS データの最大長は、32,766 です。

データベースは 文字 数で指定される長さで、グラフィック DBCS を維持します。したがって、グラフィック・フィールドとしてデータベースで使用されるバイト数は、文字数の 2 倍になります。データベースに記憶できるグラフィック DBCS データの最大長は、16,383 です。可変長グラフィック DBCS データの場合は、最大長が 16,370 になります。

Query 管理機能で印刷および表示される DBCS データ

DBCS データを含む DB2 UDB for iSeries Query 管理機能オブジェクトは、DBCS を使用できる表示装置があるかどうかには関係なく、表示することができます。

DBCS フィールドは、印刷または表示される時に、SO と SI の文字で囲まれます。これは、そのジョブが DBCS を処理できるかどうかには関係なく行われます。

シフト文字付きデータのデフォルトの報告書幅は、 SO と SI の文字を含んだデータのバイト数の長さになります。グラフィック DBCS データのデフォルトの報告書幅は、文字数の 2 倍に 2 (SO と SI のため) を加えた幅になります。

DB2 UDB for iSeries Query 管理機能は、SO 文字を列の最初の文字として位置決めします。 SO 文字は、字下げ域には入れられません (224 ページの『Query 管理機能の FORM での DBCS の使用』の INDENT を参照)。このため、SBCS 列見出しが使用された場合には位置合わせの失敗が起こりますが、 DBCS 列見出しの場合は、位置合わせは正しいものになります。 図 70 は、DBCS データを使用した報告書の一部です。 <> 文字は、SO 文字と SI 文字を表します。これらの文字は、印刷または表示された報告書ではブランクになっています。この例では、データに DBCS の英字を使っています。最初の列は DBCS の列見出しを持ち、2 番目の列は SBCS の列見出しを持ちます。

```
<E M P L O Y E E >
<N A M E >                   JOB
---------------------------  -------------
<J O H N   S H E R M A N   > <M G R      >
<B O B   S C H R A M S K I > <C L E R K   >
<J U D I T H   B A R T A   > <S A L E S   >
```

図 70. DBCS データのデフォルト報告書の例

印刷または表示される報告書で、DBCS データ・タイプのフィールドが印刷ファイルまたは表示パネルで分割される結果になった場合、ジョブが DBCS を使用できるか否かによって、報告書に差が生じます。ジョブが DBCS を使用できる場合は、SO および SI 文字が付加され、分割された列セグメントを囲みます。 図 71 は、ジョブが DBCS 使用可能である場合に、 DBCS 列が印刷ページ間で分割される状態を示しています。 DBCS を使用できないジョブの場合は、 SO 文字や SI 文字が分割された列セグメントを囲むことはありません。 図 71 では、報告書の幅は 46、PRINT REPORT 幅は 24 になります。 223 ページの図 72 は、ジョブが DBCS を使用できない場合に、印刷ページ間で DBCS 列が分割される状態を示しています。報告書の中の文字 # は、DBCS 文字の半分を表します。

```
スプール・ファイル 1:          |スプール・ファイル 2:
                   1(1)      |                        1(2)
                             |
 <E M P L O Y E E>           |
 <N A M E>                   |           JOB
 ----------------------      | --------- --------------
 <J O H N   S H E R M >      | <A N   >  <M G R      >
 <B O B   S C H R A M >      | <S K I >  <C L E R K  >
 <J U D I T H   B A R >      | <T A   >  <S A L E S  >
 <K A R E N   R A N D >      | <      >  <S A L E S  >
```

図 71. 印刷報告書の分割の例 (DBCS 使用可能)

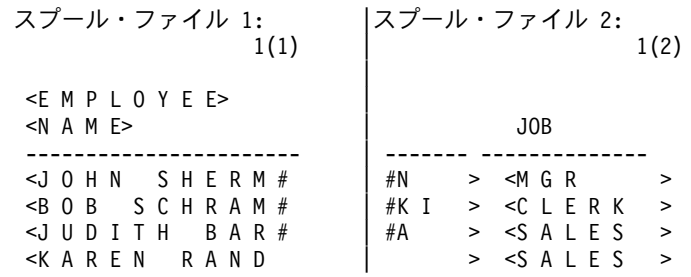


図 72. 印刷報告書の分割の例 (DBCS が使用可能でない場合)

Query 管理機能の DBCS データで使用されるデータ・タイプ

2 つの基本タイプ、CHARACTER または GRAPHIC のいずれかとしてデータを保管する列を定義することにより、データベースに DBCS データを保管することができます。

- シフト文字付きの DBCS データは、CHARACTER データ・タイプの列に入れなければなりません。シフト文字付き DBCS データと SBCS データを混用することができます。シフト文字付き DBCS 定数 (次の注を参照) も、CHARACTER データ・タイプの列に入れる必要があります。シフト文字付き DBCS 定数は、次のようにその前後に 1 バイトのアポストロフィが付けられたシフト文字付き DBCS データから構成されます。

  ```
  '<D1D2D3>'
  ```

  ここで、D1、D2 および D3 は、2 バイト文字を表します。列の中の項目は、すべて同じ長さでなければなりません。
- グラフィック DBCS ストリングは、GRAPHIC として定義された列に入れなければなりません。グラフィック DBCS 定数 (次の注を参照) も、GRAPHIC データ・タイプの列に入れる必要があります。グラフィック DBCS 定数は、次のように、その前後に 1 バイトのアポストロフィが付けられたシフト文字付き DBCS データと、その前に付く G とから構成されます。

  ```
  G'<D1D2D3>'
  ```

  グラフィック DBCS データと SBCS データを混用することはできません。グラフィックとして定義された列には、固定長または可変長のいずれのグラフィック DBCS データも入れることができます。

  注: 上述の定数は、派生列を選択する時に SQL ステートメントで使用されるものです。派生列とは、SQL 式として定義されているものです。たとえば、次の式です。

  ```
  SELECT G'<A B C >' || Column1, '<D B C S >' FROM Tablename
  ```

これらの列の最大長についての詳細は、SQL 解説書を参照してください。

DB2 UDB for iSeries Query 管理機能での DBCS データの使用

次のセクションでは、DB2 UDB for iSeries Query 管理機能において、DBCS データを使用する場合と SBCS データを使用する場合との*相違点* を述べます。

224 ページの『Query 管理機能の入力フィールドでの DBCS データの使用』

224 ページの『Query 管理機能の Query での DBCS データの使用』

224 ページの『Query 管理機能の FORM での DBCS の使用』

227 ページの『Query 管理機能での DBCS データの保管』

228 ページの『DB2 UDB for iSeries Query 管理機能コマンドでの DBCS グローバル変数の使用』

228 ページの『Query 管理機能での DBCS データのエクスポート』

229 ページの『Query 管理機能での DBCS データのインポート』

229 ページの『Query 管理機能での DBCS 報告書の印刷』

Query 管理機能の入力フィールドでの DBCS データの使用

DB2 UDB for iSeries Query 管理機能のすべての入力フィールドでは、名前を除いて、DBCS データを使用することができます。

Query 管理機能の Query での DBCS データの使用

以下は、シフト文字付き DBCS、SBCS と DBCS の混用、またはグラフィック DBCS のいずれでもかまいません。

- 置換値
- 文字データ・タイプ・フィールド内の区切り文字付きストリング
- 注記

グラフィック・データ・タイプは、グラフィック DBCS データだけを含むことができます。

SQL キーワードは英語でなければなりません。

Query 管理機能の FORM での DBCS の使用

FORM オブジェクトは、SELECT ステートメントで Query を実行した結果のデータのためのフォーマット設定の仕様を含んでいます。ここでは、FORM 仕様がどのように DBCS データを扱うかについて説明します。

FORM パネルの以下のテキストに対して、シフト付き DBCS データ、 SBCS と DBCS の混用データ、およびグラフィック DBCS データを使用することができます。

- 制御レベル・テキスト
- ページ・テキスト
- 最終テキスト

FORM 仕様は、次のように DBCS データを処理します。

USAGE
: 列の用途を指定します。FORM の USAGE は、SBCS 文字でなければなりません。

INDENT
: 列の左側のブランク・スペース。このフィールドを使用して、列とその前の列または左マージンとの間を区切ります。報告書にブランク・スペースとして示される DBCS の SO 文字および SI 文字のための、特別な考慮事項について、 221 ページの『Query 管理機能で印刷および表示される DBCS データ』を参照してください。

WIDTH

列の幅にしたい広さを指定します。列のデータ・タイプが GRAPHIC の場合、 COLUMNS.WIDTH はグラフィック DBCS の文字数として解釈されます。その他のデータ・タイプでは、バイト数として解釈されます。

列にシフト文字付き DBCS データが含まれている場合、幅のバイト数は、 COLUMNS.WIDTH の値に 2 (SO 文字と SI 文字) を加えた値になります。シフト文字付き DBCS データの最大の列幅は、32,766 バイトです。報告書には、SO 文字と SI 文字のために余分な桁をとっておく必要があります。余分な桁をとっておかないと、データを表示できない場合があります。 WIDTH が 4 個のスペースより小さいと (これは 1 桁の DBCS 文字を表示するのに必要な最小スペース)、その列はアスタリスク (***) で満たされます。

列にグラフィック・データが含まれている場合は、幅のバイト数は、 COLUMNS.WIDTH の 2 倍に 2 (グラフィック DBCS データが表示または印刷される時に追加される SO 文字と SI 文字のため) を加えた値になります。グラフィック DBCS データに対して FORM に指定できる最大幅は、 16,383 文字 (可変長 DBCS データの場合は、16,370) です。 COLUMNS.DATATYPE に GRAPHIC を指定すると、データを囲む SO 文字と SI 文字用に余分な幅を加える必要がなくなります。これは、DB2 UDB for iSeries Query 管理機能が処理するからです。

注: DB2 UDB for iSeries Query 管理機能は書式内に COLUMNS.DATATYPE を必要としないため、 SBCS データにも DBCS データにも同じ書式を適用することができます。報告書の幅 (バイト数) は、それぞれの場合で異なります。

DATATYPE

照会された表内の対応する列に入れられるデータのタイプを指定します。シフト文字付き DBCS データの COLUMNS.DATATYPE は、CHARACTER でなければなりません。グラフィック DBCS の COLUMNS.DATATYPE は、GRAPHIC でなければなりません。

EDIT EDIT コードは、列の中の値の区切り方法を決めます。

これらのコードは、1 バイト文字で書式に入力しなければなりません。

- **C** は、文字データ・タイプ列の編集コードです。値の表示は変更されません。
- **CW** は、文字データ列を折り返す場合の編集コードです。値の表示は変更されません。ただし、値が列内の 1 行に収まらない場合、列の幅に応じてテキストが折り返されます。つまり、データを列の終わりで切り捨てる代わりに、列内の 1 行にできるだけ多くのデータを入れてから、データを折り返して次の行に続けます。
- **CT** は、列テキストに応じて、文字データの列を折り返す場合の編集コードです。値の表示には変更を行いませんが、値が列内の 1 行に収まらない場合には、その列は列内のテキストに応じて折り返されます。すなわち、データを列の終わりで切り捨てる代わりに、列内の 1 行にできるだけ多くのデータを入れ、1 バイトのブランクがある個所で行を中断し、そこでデータを折り返して次の行に続けます。データのストリングが長過ぎて列に収まらず、しかも 1 バイトのブランクがないと、 1 バイト・ブランクが見つかるまでデータを幅一杯で折り返します。

  CT 編集コードを DBCS と SBCS の混用データを含む列に使用する場合には、その列の最小の幅は 4 になります。
- **G** は、グラフィック・タイプとして定義されたデータの列の編集コードです。値の表示は変更されません。
- **GW** は、グラフィック・データを折り返したい場合の編集コードです。値の表示には変更を行いませんが、値が列内の 1 行に収まらない場合には、 DB2 UDB for iSeries Query 管理機能はその列は列内のテキストに応じて折り返されます。すなわち、DB2 UDB for iSeries Query 管理機

能は、データを列の終わりで切り捨てる代わりに、列内の 1 行にできるだけ多くのデータを入れ、そこでデータを折り返して次の行に続けます。

図 73 の報告書の例は、最初の 4 列について、 COLUMNS.WIDTH が 1、2、3 および 4 の書式を使用して、グラフィック DBCS データを表示した結果を示したものです。 5 番目の列は、データを記述しています。列見出しは、SBCS データです。最初の 4 列の COLUMNS.EDIT は、GW です。実行された SQL ステートメントは、SELECT COL1、COL1、COL1、COL1、Desc FROM SAMPLE です。 COL1 という名前の列が VARGRAPHIC(10) として定義されています。

```
1      22       333        4444         Description
----   ------   -------    ----------   ----------------------------
<>     <>       <>         <>           LENGTH(COL1)=0
<1 >   <1 >     <1 >       <1 >         LENGTH(COL1)=1
<2 >   <2 2 >   <2 2 >     <2 2 >       LENGTH(COL1)=2
<2 >
<3 >   <3 3 >   <3 3 3 >   <3 3 3 >     LENGTH(COL1)=3
<3 >   <3 >
<3 >
<4 >   <4 4 >   <4 4 4 >   <4 4 4 4 >   LENGTH(COL1)=4
<4 >   <4 4 >   <4 >
<4 >
<4 >
<5 >   <5 5 >   <5 5 5 >   <5 5 5 5 >   LENGTH(COL1)=5
<5 >   <5 5 >   <5 5 >     <5 >
<5 >   <5 >
<5 >
<5 >
<6 >   <6 6 >   <6 6 6 >   <6 6 6 6 >   LENGTH(COL1)=6
<6 >   <6 6 >   <6 6 6 >   <6 6 >
<6 >   <6 6 >
<6 >
<6 >
<6 >
<7 >   <7 7 >   <7 7 7 >   <7 7 7 7 >   LENGTH(COL1)=7
<7 >   <7 7 >   <7 7 7 >   <7 7 7 >
<7 >   <7 7 >   <7 >
<7 >   <7 >
<7 >
<7 >
<7 >
<8 >   <8 8 >   <8 8 8 >   <8 8 8 8 >   LENGTH(COL1)=8
<8 >   <8 8 >   <8 8 8 >   <8 8 8 8 >
<8 >   <8 8 >   <8 8 >
<8 >   <8 8 >
<8 >
<8 >
<8 >
<8 >
<9 >   <9 9 >   <9 9 9 >   <9 9 9 9 >   LENGTH(COL1)=9
<9 >   <9 9 >   <9 9 9 >   <9 9 9 9 >
<9 >   <9 9 >   <9 9 9 >   <9 >
<9 >   <9 9 >
<9 >   <9 >
<9 >
<9 >
<9 >
<9 >
```

図 73. GRAPHIC データに GW を使用した報告書

Query 管理機能でのデータ切り捨ての処理方法

DB2 UDB for iSeries Query 管理機能は、DBCS 文字の分割を避ける方法として、フィールド境界または画面境界で、表示される DBCS データを切り捨てます。切り捨てられた行の文字を表示するには、ページ送りが必要です。

シフトインおよびシフトアウトの文字は、幅または列内に含まれ、報告書の字下げまたはマージンには入れられません。

Query 管理機能での DBCS データの保管

DB2 UDB for iSeries Query 管理機能では、Query によって選択された DBCS データの保管方法として、次の方法をサポートしています。

1. DB2 UDB for iSeries Query 管理機能コマンド: SAVE DATA AS
2. CL コマンド: STRQMQRY OUTPUT(*OUTFILE)

SAVE DATA AS コマンドを使用して DBCS データを保管する場合、次の規則が適用されます。

- DBCS 択一列は、DBCS 択一および DBCS 混用の列に保管できます。
- 列のすべてのフィールドがシフト文字付き DBCS ストリングである場合のみ、 DBCS 択一列を DBCS 専用列に保管することができます。
- 列のフィールドがいずれもシフト文字付き DBCS フィールドではない場合のみ、 DBCS 択一列を CHAR (非 DBCS) 列に保管することができます。
- DBCS 専用列は、DBCS 専用列、DBCS 混用列、および DBCS 択一列に保管できます。
- DBCS 専用列は、CHAR 列に保管できません。
- DBCS 混用列は、DBCS 混用列に保管することができます。
- 列のすべてのフィールドがシフト文字付き DBCS フィールドである場合のみ、 DBCS 混用列を DBCS 専用列に保管することができます。
- 列のすべてのフィールドがシフト文字付き DBCS フィールドまたは SBCS フィールドのいずれかである場合のみ、 DBCS 混用列を DBCS 択一列に保管することができます。これは、ストリングが混用である場合は許されません。たとえば、値 X'F10EF1F10F' を持つ列は、DBCS 択一列に保管することはできません (0E は DBCS のシフトアウト、また、0F はシフトインのシフト文字です)。
- CHAR フィールドは、DBCS 混用フィールドおよび DBCS 択一フィールドに保管することができます。
- CHAR フィールドは、DBCS 専用フィールドに保管することはできません。
- グラフィック DBCS データは、グラフィック DBCS データだけに保管することができます。その他のデータ・タイプをグラフィック DBCS データに保管することはできません。
- 固定長および可変長のグラフィック DBCS データ・タイプに関する長さの規則は、固定長および可変長の文字データ・タイプの長さの規則と同じです。ただし、有効な組み合わせを判別する場合には、グラフィック DBCS データの長さは文字数でカウントされ、文字データの長さはバイト数でカウントされる点を考慮に入れなければなりません。
  - 保管されるデータの定義された長さが、保管先の定義された長さ以下である場合は、固定長のグラフィック DBCS データは、固定長のグラフィック DBCS データに保管することができます。
  - 保管されるデータの定義された長さが、保管先の定義された長さ以下である場合は、固定長のグラフィック DBCS データは、可変長のグラフィック DBCS データに保管することができます。
  - 保管されるデータの定義された長さが、保管先の定義された長さ以下である場合は、可変長のグラフィック DBCS データは、可変長のグラフィック DBCS データに保管することができます。

– 保管されるデータの定義された長さが、保管先の定義された長さ以下である場合は、可変長のグラフィック DBCS データは、固定長のグラフィック DBCS データに保管することができます。

DB2 UDB for iSeries Query 管理機能コマンドでの DBCS グローバル変数の使用

DB2 UDB for iSeries Query 管理機能グローバル変数は、CHAR および INTEGER だけが使用できます。DB2 UDB for iSeries Query 管理機能は、GRAPHIC タイプのグローバル変数はサポートしません。 SQL ステートメントでグラフィック DBCS 定数を代用する必要がある場合には、使用するグローバル変数は、CHAR タイプにセットしなければなりません。 DB2 UDB for iSeries SQL では、SQL ステートメントでグラフィック DBCS 定数しか許されないため、定数が次の形式であれば、

```
G'<A B C D E >'
```

次の文字ストリングを変数プールにセットしておかなければなりません。

```
"G'<A B C D E >'"
```

たとえば、次の Query を実行する場合、

```
SELECT * FROM Table WHERE (ColumnName=&ComparisonValue
```

そして、ColumnName という名前の列が GRAPHIC タイプである場合、グローバル変数 ComparisonValue の値は、グラフィック DBCS 定数でなければなりません。次の例は、SET GLOBAL コマンドを Query プロシージャーの中に入れる方法を示しています。それぞれの SET GLOBAL コマンドは機能的には同じもので、 SQL ステートメントが正しく作成されるようにグローバル変数をセットしています。

```
"  SET GLOBAL (ComparisonValue=G'<A B C D E >'  "
"  SET GLOBAL (ComparisonValue=""G'<A B C D E >'""  "
'  SET GLOBAL (ComparisonValue="G'<A B C D E >'"  '
```

これらの 3 つのすべてについて、実行するための SQL ステートメントは次のものです。

```
SELECT * FROM Table WHERE (ColumnName=G'<A B C D E >'
```

グローバル変数のセットについては、 『付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用』にさらに例が載っています。

CL コマンド

次の例は、上の Query を正常に実行するために STRQMQRY SETVAR() をどのようにコードしなければならないかを示しています。

```
STRQMQRY QMQRY(QueryName) SETVAR(('ComparisonValue' 'G'<A B C D E >''))
```

Query 管理機能での DBCS データのエクスポート

グラフィックおよび可変グラフィックとして定義されたデータは、エクスポートすることができます。

エクスポートされるデータの見出しレコードのデータ・タイプ・コードは、次のとおりです。

VARGRAPHIC の場合は、464

GRAPHIC の場合は、468

シフト文字付き DBCS データを含む文字として定義されたデータは、エクスポートすることができます。グラフィックとして定義されたデータも、エクスポートできます。

エクスポート・データの列幅は、データ内の DBCS 文字数であり、これは、データを保管するのに使われるバイト数の半分です。列データは、データベースから取り出されたものが、正確にそのままデータ・レコードに保管されます。

Query 管理機能での DBCS データのインポート

DBCS データは、Query、プロシージャー、および書式でインポートすることができます。 DBCS Query およびプロシージャーをこのようにインポートする場合には、レコード長が 79 バイトを超えてはなりません。

Query 管理機能での DBCS 報告書の印刷

DBCS データを使用している時に、報告書が複数のページに分割される場合、報告書の 2 ページ目およびそれ以降の印刷は、そのページの左側から 4 バイト目の位置から再開されます。

付録 B. DB2 UDB for iSeries Query 管理機能インターフェースの例

この付録には、報告書を作成するために、 Query 管理機能の呼び出し可能インターフェースを使用する方法を説明してあります。プロシージャーまたはプログラムの作成方法を示す、 RPG と COBOL のサンプル・プログラムが用意されています。この例をユーザーのシステムで使用して、 DB2 UDB for iSeries Query 管理機能の呼び出し可能インターフェースの使い方を学ぶことができます。 Query 管理機能に対する CL インターフェースの例については、 『第 13 章 Query 管理機能での制御言語インターフェース』を参照してください。

Query 管理機能での報告書の作成

図 74 は、Query 管理機能を使用して、事前に定義された Query 管理機能書式オブジェクト (QMFORM) と、 Query 管理機能プログラム・オブジェクト (QMQRY) を使ったデータの選択と報告書の作成の例を示しています。報告書の作成には、SAMP1 プログラムを実行します。

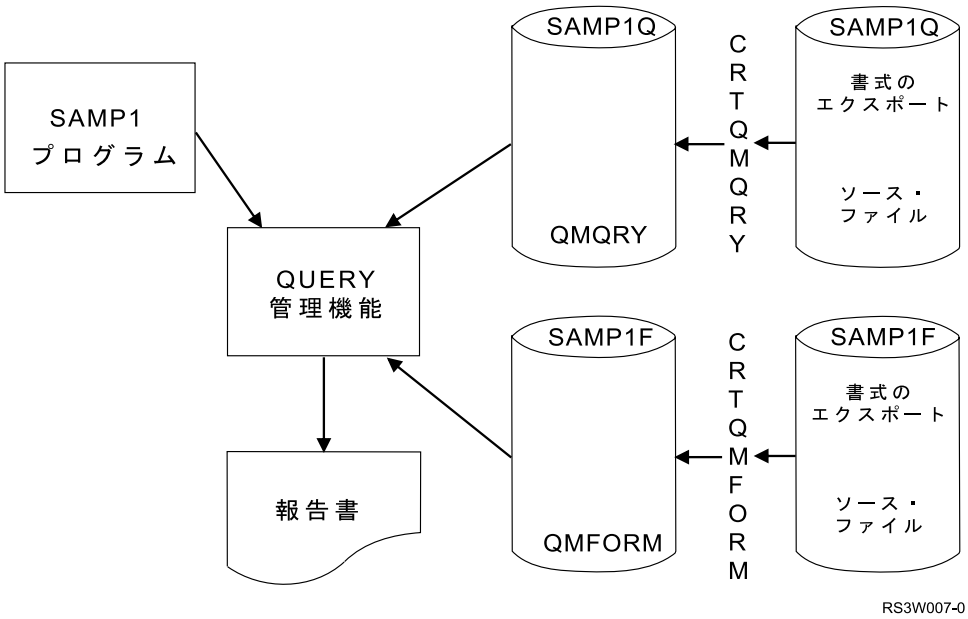


図 74. Query 管理機能を使用した報告書作成の概要

この例では、システムにはデータベース・ファイル WKPAY があり、この中には、社員名 (EMPLOYEE NAME)、社員番号 (EMPLOYEE NUMBER)、週あたりの勤務時間 (WEEKLY HOURS WORKED)、時間あたりの給与額 (HOURLY RATE OF PAY) の情報が入っています。 SAMP1 プログラムを実行する前に、QMQRY オブジェクトと QMFORM オブジェクトをシステムに作成してください。これには、まず、Query 管理機能プログラム・ソース (QMQRYSRC) ファイルと、 Query 管理機能書式ソース (QMFORMSRC) ファイルに、 2 つのオブジェクトのためのソースを作成することから始めます。次に、Query 管理機能プログラムの作成 (CRTQMQRY) コマンドと、 Query 管理機能書式の作成 (CRTQMFORM) コマンドを使用して、これらを QMQRY と QMFORM の形式にインポートします。232 ページの図 75 は、WKPAY ファイルのためのデータ記述仕様 (DDS) です。

```
     *
     * weekly payroll details
     *
    A                                     UNIQUE
    A          R PAYR                     TEXT('Weekly Pay Record')
    A            EMPNO          5         COLHDG('Employee' 'Number')
    A            NAME          20         COLHDG('Employee' 'Name')
    A            HOURS          5S 2      COLHDG('Hours' 'Worked')
    A            RATE           5S 2      COLHDG('Hourly' 'Rate')
    A            WKAMT          5S 2      COLHDG('Weekly' 'Pay')
    A          K EMPNO
```

図 75. WKPAY ファイル用のデータ記述仕様

図 76 は、この例で使用される Query です。

```
SELECT NAME, HOURS, RATE, WKAMT FROM BPLIB/WKPAY
ORDER BY NAME
```

図 76. Query ソース選択ステートメント

注: このソース・ファイルの最大長は、91 文字 (行番号に 6 文字、日付に 6 文字、データに 79 文字) です。サンプル・プログラムが出す Query コマンドの START には *SYS を使用するオプションが指定されているので、 SQL ステートメントは OS/400 の標準に適合するように編成されます。

図 77 は、SAMP1 プログラムの実行結果です。

```
DATE: 11/21/89          WEEKLY PAY REPORT         PAGE:    1

                          HOURS         HOURLY         PAY
EMPLOYEE NAME             WORKED        RATE          AMOUNT

ANDERSON, W                38.50          6.23        100.00
COLLINS, R                 41.50          5.40        400.00
GREEN, C                   40.00          7.10        300.00
SMITH, T                   42.00          5.30        200.00

                                           TOTAL    1,000.00
```

図 77. SAMP1 の報告書の結果

Query 管理機能のサンプル・プログラム

RPG (233 ページの図 78) および COBOL (235 ページの図 79) で準備されているサンプル・プログラムは、どちらも同じ機能を実行します。これらのプログラムは、一度に 1 レコードずつ読み取り、週間給与総額を計算してから、計算済みの情報でファイルを更新するという方法で、 WKPAY ファイルを処理します。すべてのレコードを更新すると、インターフェース・プログラム QQXMAIN に対する呼び出しによって Query 管理機能が開始され、 START コマンドおよび連絡域が渡されます。 START コマンドは、このインターフェース・セッションではシステムの命名規則 (*SYS) を使用し、デフォルト (*SAA) は使用しないことを指定しています。

呼び出し可能インターフェースを呼び出し、RUN コマンドを渡してください。次に、PRINT コマンドを使用して、結果を印刷します。 EXIT コマンドを使用してインターフェースを終了し、制御を呼び出しプログラムに戻して、プログラムを終了します。

Query 管理機能のデータをインターフェースに渡す時には、コマンドおよびパラメーターの長さを整数 (2 進数) 形式にする必要があります。データをこの形式で渡すことができるような構造が、プログラムでセットアップされています。

| コンパイル時には、連絡域が入っているモジュールがプログラムに組み込まれます。これを使用して、
| Query 管理機能とユーザー・プログラムの間で操作状況をやりとりします。 Query の状況についてのイン
| ターフェース・プログラム名および標準フィールド名は、組み込みモジュールの中でも定義されます。シス
| テム相互間で Query アプリケーションを転送することができるように、必要な場合にはいつでもこれらの
| 名前をアプリケーション・プログラムで使用してください。

Query 管理機能での RPG プログラムの例

図 78 は、WKPAY ファイルを処理する RPG のサンプル・プログラムです。

```
 ****************************************************************
 *                                                              *
 *         SAMPLE 1 RPG PROGRAM USING QUERY INTERFACE           *
 *         ------------------------------------------           *
 *                                                              *
 *  1) Include member DSQCOMMR contains the communications      *
 *     area to be passed to the query management interface.     *
 *  2) The WKPAY weekly payroll details are read and the hours  *
 *     worked are multiplied by the hourly rate to calculate    *
 *     the weeks pay. The file is then updated with the weekly  *
 *     pay amount.                                              *
 *  3) Once all the records in the WKPAY file are updated then  *
 *     the interface is started and a query report              *
 *     printed using the just updated file.                     *
 *                                                              *
 ****************************************************************
H
FWKPAY   UF  E                    DISK
 *
E                    COM     1   4 25               interface cmds
 *
I            DS
I                                    B   1   40BIN1
I                                    B   5   80BIN2
I                                    B   9  120BIN3
I                                    B  13  160BIN4
I/COPY QRPG/QIRGINC,DSQCOMMR
 *
 * Update the Weekly Pay file with weekly earnings:
 *
C           *IN50     DOUEQ'1'
C                     READ WKPAY                    50 EOF
C  N50      HOURS     MULT RATE      WKAMT     H      calculate pay
C  N50               UPDATPAYR                       update pay file
C                     END
 *                                                    ensure all
C                     FEOD WKPAY                      changes done
```

図 78. RPG プログラムの例 (1/2)

```
     *                                                    in storage
     * Start the query interface session:
     *
    C                     CALL DSQCIR
    C                     PARM           DSQCOM          comms area
    C                     PARM 5         BIN1            command length
    C                     PARM           COM,1           START
    C                     PARM 1         BIN2            # keywords
    C                     PARM 8         BIN3            keyword length
    C                     PARM 'DSQSNAME'DATA8   8       keyword
    C                     PARM 4         BIN4            value length
    C                     PARM '*SYS'    DATA4   4       value
    C                     PARM DSQVCH    TYPE    4       CHAR
     *
     * Run the query:
     *
    C                     CALL DSQCIR
    C                     PARM           DSQCOM          comms area
    C                     PARM 16        BIN1            command length
    C                     PARM           COM,2           RUN QUERY
     *                                                     SAMP1Q
     * Print the results of the query:
     *
    C                     CALL DSQCIR
    C                     PARM           DSQCOM          comms area
    C                     PARM 25        BIN1            command length
    C                     PARM           COM,3           PRINT REPORT
     *                                                    (FORM=SAMP1F
     * End the query interface session:
     *
    C                     CALL DSQCIR
    C                     PARM           DSQCOM          comms area
    C                     PARM 4         BIN1            command length
    C                     PARM           COM,4           EXIT
     *
    C                     MOVE '1'       *INLR           end the program
     *
**    commands loaded as compile time array
START
RUN QUERY SAMP1Q
PRINT REPORT (FORM=SAMP1F
EXIT
```

図 78. RPG プログラムの例 (2/2)

Query 管理機能での COBOL プログラムの例

235 ページの図 79 は、WKPAY ファイルを処理する COBOL のサンプル・プログラムです。

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.     SAMP1.
 DATE-COMPILED.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION
 SOURCE-COMPUTER.    IBM-AS400.
 OBJECT-COMPUTER.    IBM-AS400.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT PAY-FILE
         ASSIGN TO DISK-WKPAY
         FILE STATUS IS PAY-FILE-STATUS.
 DATA DIVISION.
 FILE SECTION.
 FD  PAY-FILE LABEL RECORDS STANDARD.
 01  PAY-REC.
     COPY DDS-PAYR OF WKPAY.
 WORKING-STORAGE SECTION.
*****************************************************************
*                                                               *
*         SAMPLE 1 COBOL PROGRAM USING QUERY INTERFACE          *
*         ------------------------------------------            *
*                                                               *
*  1) Include member DSQCOMMB contains the communications       *
*     area to be passed to the query management interface.      *
*                                                               *
*  2) The WKPAY weekly payroll details are read and the hours   *
*     worked are multiplied by the hourly rate to calculate     *
*     the weeks pay. The file is then updated with the weekly   *
*     pay amount.                                               *
*                                                               *
*  3) Once all the records in the WKPAY file are updated,       *
*     the interface is started and a query report               *
*     printed using the file just updated.                      *
*                                                               *
*****************************************************************

* Include the Communications area

   COPY DSQCOMMB OF QLBL-QILBINC.

 * Query Interface Commands

  77   START-CMD           PIC X(5)   VALUE "START".
  77   KEYWORD-NAME        PIC X(8)   VALUE "DSQSNAME".
  77   NAME-VALUE          PIC X(4)   VALUE "*SYS".
  77   RUN-QUERY-CMD       PIC X(16)  VALUE "RUN QUERY SAMP1Q".
  77   PRINT-CMD           PIC X(25)
                           VALUE "PRINT REPORT (FORM=SAMP1F".
  77   EXIT-CMD            PIC X(4)   VALUE "EXIT".
```

図 79. COBOL プログラムの例 (1/3)

```
77   ONE                 PIC 9(8) USAGE IS BINARY VALUE 1.
77   FOUR                PIC 9(8) USAGE IS BINARY VALUE 4.
77   FIVE                PIC 9(8) USAGE IS BINARY VALUE 5.
77   EIGHT               PIC 9(8) USAGE IS BINARY VALUE 8.
77   SIXTEEN             PIC 9(8) USAGE IS BINARY VALUE 16.
77   TWENTY-FIVE         PIC 9(8) USAGE IS BINARY VALUE 25.

77   PAY-FILE-STATUS     PIC XX.

01   FILE-END            PIC X VALUE SPACE.
     88 END-OF-FILE      VALUE "E".

01   FILE-ERROR-INFO.
     05  OP-NAME         PIC X(7).
     05  FILLER          PIC X(20) VALUE " ERROR ON FILE WKPAY".
     05  FILLER          PIC X(18) VALUE " - FILE STATUS IS ".
     05  STATUS-VALUE    PIC XX.

PROCEDURE DIVISION.
DECLARATIVES.
FILE-ERROR SECTION.
    USE AFTER STANDARD ERROR PROCEDURE ON PAY-FILE.
FILE-ERROR-PARA.
    MOVE PAY-FILE-STATUS TO STATUS-VALUE.
    DISPLAY "FILE PROCESSING ERROR".
    DISPLAY FILE-ERROR-INFO.
    DISPLAY "PROCESSING ENDED DUE TO FILE ERROR".
    STOP RUN.
END DECLARATIVES.

FILE-UPDATE SECTION.
OPEN-FILE.
    MOVE "OPEN" TO OP-NAME.
    OPEN I-O PAY-FILE.
    PERFORM READ-PAY-FILE THRU UPDATE-PAY-FILE
         UNTIL END-OF-FILE.
READ-PAY-FILE.
    MOVE "READ" TO OP-NAME.
    READ PAY-FILE
         AT END SET END-OF-FILE TO TRUE
         MOVE "CLOSE" TO OP-NAME
         CLOSE PAY-FILE
         PERFORM PROCESS-QUERY.
UPDATE-PAY-FILE.
    MULTIPLE HOURS BY RATE GIVING WKAMT ROUNDED.
    MOVE "UPDATE" TO OP-NAME.
    REWRITE PAY-REC.

* Query Interface command and parameter lengths

PROCESS-QUERY SECTION.
START-INTERFACE.
    CALL DSQCIB USING DSQCOMM, FIVE, START-CMD,
                      ONE, EIGHT, KEYWORD-NAME,
                      FOUR, NAME-VALUE, DSQ-VARIABLE-CHAR.
```

図 79. COBOL プログラムの例 (2/3)

```
RUN-QUERY.
    CALL DSQCIB USING DSQCOMM, SIXTEEN, RUN-QUERY-CMD.
PRINT-REPORT.
    CALL DSQCIB USING DSQCOMM, TWENTY-FIVE, PRINT-CMD.
EXIT-INTERFACE.
    CALL DSQCIB USING DSQCOMM, FOUR, EXIT-CMD.
    STOP RUN.
```

図 79. COBOL プログラムの例 (3/3)

Query 管理機能での Query および書式ソース

233 ページの図 78 と 235 ページの図 79 の、RPG と COBOL のサンプル・プログラムは、報告書を作成するために Query と書式のソース・ファイルを使用しています。 図 80 は、サンプル・プログラムで参照している Query ソース・ファイル (SAMP1Q) です。

```
SELECT NAME, HOURS, RATE, WKAMT FROM BPLIB/WKPAY
ORDER BY NAME
```

図 80. サンプルの Query ソース・ファイル

図 81 は、サンプル・プログラムで参照しているソース (SAMP1F) です。

```
H QM4 01 F 01 E E   E R 01 03 89/11/20 15:51
T 1110 004 005 1112 007 1115 006 1116 005 1118 003 1113 015
R CHAR    2      30    1   Employee_Name
R NUMERIC 2      8     2   Hours_Worked
R NUMERIC 2      8     3   Hourly_Rate
R NUMERIC 2      8     4   Weekly_Pay
E
```

図 81. サンプルのソース

Query 管理機能での Query および書式の印刷出力

図 82 は、サンプル・プログラムで参照している Query のソース・ファイル (SAMP1Q) に対して、コマンド PRINT QUERY SAMP1Q を実行した結果の印刷出力です。

```
                                    DB2 Query 管理機能 OS/400

Query  . . . . . :   SAMP1Q
  ライブラリー . . :     BPLIB
テキスト . . . . . :   SAA Query
SEQNBR *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8...
000001 SELECT NAME, HOURS, RATE, WKAMT FROM BPLIB/WKPAY
000002 ORDER BY NAME
                          * * * * *   ソースの終り   * * * * *
```

図 82. Query ソース・ファイルの印刷出力

図 83 は、サンプル・プログラムで参照している書式のソース (SAMP1F) に対して、コマンド PRINT FORM SAMP1F を実行した結果の印刷出力です。

```
書式 . . . . . :   SAMP1F
  ライブラリー . . :     BPLIB
テキスト . . . . . :   サンプル 1 プログラムのための書式レイアウト
                                                    列情報
番号     見出し           使用目的          タイプ          字下げ        幅       編集      SEQ
  1      Employee_Name                      CHAR              2         30                   1
  2      Hours_Worked                       NUMERIC           2          8                   2
  3      Hourly_Rate                        NUMERIC           2          8                   3
  4      Weekly_Pay                         NUMERIC           2          8                   4
                                                    ページ情報
見出しテキスト. . . . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . . :   2
フッター・テキスト  . . . . . . . . . . . . . :   NO
フッターの前のブランク行数  . . . . . . . . . :   2
フッターの後のブランク行数  . . . . . . . . . :   0
                                                    最終情報
最終テキスト  . . . . . . . . . . . . . . . . :   NO
最終テキストの改ページ  . . . . . . . . . . . :   NO
最終合計の行                . . . . . . . . . . :   1
テキストの前のブランク行数  . . . . . . . . . :   0
                                                    制御レベル情報
制御レベル番号  . . . . . . . . . . . . . . . :   1
この制御レベル番号の列  . . . . . . . . . . . :   NONE
見出しテキスト. . . . . . . . . . . . . . . . :   NO
見出しの改ページ  . . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . . :   0
列見出しの反復  . . . . . . . . . . . . . . . :   NO
フッター・テキスト  . . . . . . . . . . . . . :   NO
フッターの改ページ  . . . . . . . . . . . . . :   NO
フッターの前のブランク行数  . . . . . . . . . :   0
フッターの後のブランク行数  . . . . . . . . . :   1
制御レベル合計の行  . . . . . . . . . . . . . :   1
                                                    制御レベル情報
制御レベル番号  . . . . . . . . . . . . . . . :   2
この制御レベル番号の列  . . . . . . . . . . . :   NONE
見出しテキスト. . . . . . . . . . . . . . . . :   NO
見出しの改ページ  . . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . . :   0
列見出しの反復  . . . . . . . . . . . . . . . :   NO
```

図 83. 書式のソースの印刷出力 (1/3)

```
書式 . . . . . :   SAMP1F
  ライブラリー . . :     BPLIB
テキスト . . . . . :   サンプル 1 プログラムのための書式レイアウト
フッター・テキスト  . . . . . . . . . . . . :   NO
フッターの改ページ  . . . . . . . . . . . . :   NO
フッターの前のブランク行数  . . . . . . . . :   0
フッターの後のブランク行数  . . . . . . . . :   1
制御レベル合計の行  . . . . . . . . . . . . :   1
                                                 制御レベル情報
制御レベル番号  . . . . . . . . . . . . . . :   3
この制御レベル番号の列  . . . . . . . . . . :   NONE
見出しテキスト. . . . . . . . . . . . . . . :   NO
見出しの改ページ  . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . :   0
列見出しの反復  . . . . . . . . . . . . . . :   NO
フッター・テキスト  . . . . . . . . . . . . :   NO
フッターの改ページ  . . . . . . . . . . . . :   NO
フッターの前のブランク行数  . . . . . . . . :   0
フッターの後のブランク行数  . . . . . . . . :   1
制御レベル合計の行  . . . . . . . . . . . . :   1
                                                 制御レベル情報
制御レベル番号  . . . . . . . . . . . . . . :   4
この制御レベル番号の列  . . . . . . . . . . :   NONE
見出しテキスト. . . . . . . . . . . . . . . :   NO
見出しの改ページ  . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . :   0
列見出しの反復  . . . . . . . . . . . . . . :   NO
フッター・テキスト  . . . . . . . . . . . . :   NO
フッターの改ページ  . . . . . . . . . . . . :   NO
フッターの前のブランク行数  . . . . . . . . :   0
フッターの後のブランク行数  . . . . . . . . :   1
制御レベル合計の行  . . . . . . . . . . . . :   1
                                                 制御レベル情報
制御レベル番号  . . . . . . . . . . . . . . :   5
この制御レベル番号の列  . . . . . . . . . . :   NONE
見出しテキスト. . . . . . . . . . . . . . . :   NO
見出しの改ページ  . . . . . . . . . . . . . :   NO
見出しの前のブランク行数  . . . . . . . . . :   0
見出しの後のブランク行数  . . . . . . . . . :   0
列見出しの反復  . . . . . . . . . . . . . . :   NO
フッター・テキスト  . . . . . . . . . . . . :   NO
フッターの改ページ  . . . . . . . . . . . . :   NO
```

図 83. 書式のソースの印刷出力 (2/3)

```
                                DB2 Query 管理機能 OS/400

書式 . . . . . :   SAMP1F
  ライブラリー . . :     BPLIB
テキスト . . . . . :   サンプル 1 プログラムのための書式レイアウト
フッターの前のブランク行数  . . . . . :   0
フッターの後のブランク行数  . . . . . :   1
制御レベル合計の行  . . . . . :   1
                                                制御レベル情報
制御レベル番号  . . . . . :   6
この制御レベル番号の列  . . . . . :   NONE
見出しテキスト. . . . . :   NO
見出しの改ページ  . . . . . :   NO
見出しの前のブランク行数  . . . . . :   0
見出しの後のブランク行数  . . . . . :   0
列見出しの反復  . . . . . :   NO
フッター・テキスト  . . . . . :   NO
フッターの改ページ  . . . . . :   NO
フッターの前のブランク行数  . . . . . :   0
フッターの後のブランク行数  . . . . . :   1
制御レベル合計の行  . . . . . :   1
                                                  オプション情報
明細行の間隔  . . . . . :   1
制御レベル列の制御  . . . . . :   YES
省略時の制御レベルのテキスト  . . . . . :   YES
列折り返し行のページまでの保存  . . . . . :   YES
列見出し区切り記号  . . . . . :   YES
制御レベル合計区切り記号  . . . . . :   YES
最終合計区切り記号  . . . . . :   YES
              * * * * *  コ ン ピ ュ ー タ ー 印 刷 出 力 の 終 り  * * * * *
```

図 83. 書式のソースの印刷出力 (3/3)

付録 C. Query 管理機能でのグローバル変数設定時の引用符およびアポストロフィの使用

プロシージャーまたはプログラムの中でグローバル変数を設定する時に、引用符またはアポストロフィをいくつ使用したらよいか、判断に迷うことがあるかもしれません。使用する DB2 UDB for iSeries Query 管理機能の機能によって、使用しなければならない引用符とアポストロフィの数は大きく変わります。

システムは、種々のレベルの解析を行わなければならないので、引用符またはアポストロフィのセットが必要になります。引用符が正しい個数だけ使用されていないと、Query は正しく実行されません。

一般的な規則は、次のとおりです。

> 引用符またはアポストロフィのセットが他の引用符またはアポストロフィのセットの中にあって、そのタイプ (引用符またはアポストロフィ) がすべて同じであれば、それらを保持するには内側のセットを二重にしなければなりません。

次の例は、SQL ステートメントの中でリテラルとして使用されるグローバル変数の設定時に、引用符とアポストロフィがどのように使用されるかを示したものです。この例では、中に組み込まれた引用符またはアポストロフィが置換変数の中にあると、どうなるかを示しています。

置換変数を要求する次の SQL ステートメントが使用されるとします。

```
SELECT * FROM CUSTINFO
WHERE CUSTNAME=&CUSTNAME
```

以下の報告書が作成されます。

```
CUSTNAME                   CUSTID
---------------   ---------------
O'Malley                      450
```

変数置換を行って、報告書を作成する実際の SQL ステートメントは、次のものです。

```
SELECT * FROM CUSTINFO
       WHERE CUSTNAME='O''Malley'
```

Query 管理機能での疑問符およびアポストロフィの Query グローバル変数プール規則

次の値が変数 CUSTNAME 用にグローバル変数プールにセットされます。

```
'O''Malley'
```

SQL ステートメントの規則に適合させるために、中に入っている引用符は二重になっています。
CUSTNAME は文字ストリング・データ・タイプなので、比較値はアポストロフィで囲まれています。

Query 管理機能での疑問符およびアポストロフィの CL コマンド規則

これを実行するために入力される CL コマンドは、次のものです。

```
STRQMQRY  QMQRY(CUSTQRY)  SETVAR((CUSTNAME ' ''O''''Malley'' ')))
```

CL コマンドの規則に適合させるために、変数値の中に入っている引用符を二重にし、ストリング全体をアポストロフィで囲んであります。

Query 管理機能での疑問符およびアポストロフィのメッセージ・プロンプト規則

Query を対話式に実行していて、変数 CUSTNAME がセットされていないと、 QWM1913 のメッセージ・プロンプトが出ます。このメッセージ・プロンプトに対して入力する値は次のとおりです。

```
                    プログラム・メッセージの表示

変数 CUSTNAME に値を入力して、実行キーを押してください。

応答を入力して、実行キーを押してください。
  応答 . . .  'O''Malley'_________________




F3=終了    F12=取消し
```

Query 管理機能での疑問符およびアポストロフィの高水準言語プログラミング規則

Query コマンドが高水準言語プログラミングから呼び出し可能インターフェース (短バージョン) を介して入力されると、その設定は次のようになります。

```
SET GLOBAL (CUSTNAME=' ''O''''Malley'' '
```

Query コマンドのストリング変数値規則に合わせるために、変数値の中に入れるアポストロフィは二重にし、ストリング全体をアポストロフィで囲みます。次は、高水準言語の例です。

- RPG

  ストリングのセットアップの例については、 120 ページの図 20 を参照してください。

- COBOL

  ```
  MOVE "SET GLOBAL (CUSTNAME=' ''O''''Malley'' '" TO COMMAND-STRING;
  ```

- C

  ```
  strcpy(command_string,"SET GLOBAL (CUSTNAME=' ''O''''Malley'' '");
  ```

Query 管理機能での疑問符およびアポストロフィの Query プロシージャーの使用

Query コマンドが Query プロシージャーを介して入力される場合は、その変数は次のようにセットされます。

```
'SET GLOBAL (CUSTNAME='' ''''O''''''''Malley'''' ''    '
```

SET GLOBAL コマンドをプロシージャーから実行する場合は、呼び出し可能インターフェース・コマンド行に適用される規則がすべて適用されます。プロシージャーからコマンドを入力するには、コマンド・ストリングの中に組み込まれるすべてのアポストロフィを二重にし、しかもストリング全体をアポストロフィで囲む必要があります。

Query 管理機能で変数を単純化する方法

次の方法を使用して、変数の設定を単純化することができます。

1. C、COBOL または RPG で呼び出し可能インターフェースの拡張バージョンを使用して、グローバル変数値をセットします。正確な変数値をストリングにタイプします。フィールドを区切るため、または文字値を指示するために必要なもの以外には、余分な引用符またはアポストロフィは必要ありません。
   - COBOL

     ```
     MOVE "'O''Malley'" TO SET-VALUE;
     ```

   - C

     ```
     strcpy(set_value,"'O''Malley'");
     ```

2. アポストロフィと同じように引用符 (") を使用します。

   呼び出し可能インターフェースの Query コマンドには、次を使用します。

   ```
   SET GLOBAL (CUSTNAME="'O''Malley'"
   ```

   コマンド・ストリング内の変数値を引用符で区切る場合は、その値の中で使用するアポストロフィは、二重にする必要はありません。

   ```
   "SET GLOBAL (CUSTNAME="" 'O''Malley' "" "
   ```

   SET GLOBAL コマンドをプロシージャーから実行する場合は、呼び出し可能インターフェースのコマンド行に適用される規則がすべて適用されます。プロシージャーからコマンドを入力するには、コマンド・ストリングは、引用符 (") またはアポストロフィ (') で囲む必要があります。引用符で囲む時は、ストリングの中で使用する引用符はすべて二重にしなければなりません。アポストロフィで囲む場合は、ストリングの中のすべてのアポストロフィを二重にしてください。

付録 D. Query 管理機能でのソート・シーケンスの例

この付録は、複数の言語環境でのソート・シーケンスの例を示しています。ここでは、Query および報告書の例は、すべて STAFF 表に対して実行されたものです。スタッフ表 (STAFF) には、従業員の名前や業務などの情報が入っています。スタッフ表は次のようになっています。

表 17. STAFF 表

ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
10	Sanders	20	Mgr	7	18357.50	0
20	Pernal	20	Sales	8	18171.25	612.45
30	Merenghi	38	MGR	5	17506.75	0
40	OBrien	38	Sales	6	18006.00	846.55
50	Hanes	15	Mgr	10	20659.80	0
60	Quigley	38	SALES	00	16808.30	650.25
70	Rothman	15	Sales	7	16502.83	1152.00
80	James	20	Clerk	0	13504.60	128.20
90	Koonitz	42	sales	6	18001.75	1386.70
100	Plotz	42	mgr	6	18352.80	0

ソート・シーケンスを使用して、以下のことを行うことができます。

- ソート
- グループ
- 結合
- レコードの選択
- 最小値および最大値の判別
- SBCS 文字データでの報告書の制御の切れ目作成

以下の例では、Query および結果の報告書は、2 進文字コード・ソート・シーケンス (*HEX)、共用重みソート・シーケンス (*LANGIDSHR)、または固有重みソート・シーケンス (*LANGIDUNQ) を使用しています。

Query 管理機能でのソートの例

次の SQL ステートメントによって、報告書は、 表 17 の JOB (業務) の列の値を使用してソートされます。

```
SELECT * FROM STAFF ORDER BY JOB
```

246 ページの図 84 は、*HEX ソート・シーケンスを使用したソートの結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   _______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

            ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
         -------  ---------  -------  -----  -------  -----------  ---------
000001      100   Plotz          42   mgr          7    18,352.80        .00
000002       90   Koonitz        42   sales        6    18,001.75   1,386.70
000003       80   James          20   Clerk        0    13,504.60     128.20
000004       10   Sanders        20   Mgr          7    18,357.50        .00
000005       50   Hanes          15   Mgr         10    20,659.80        .00
000006       30   Marenghi       38   MGR          5    17,506.75        .00
000007       20   Pernal         20   Sales        8    18,171.25     612.45
000008       40   OBrien         38   Sales        6    18,006.00     846.55
000009       70   Rothman        15   Sales        7    16,502.83   1,152.00
000010       60   Quigley        38   SALES        0    16,808.30     650.25
******  * * * * *  データの終わり  * * * * *



   F3=終了      F12=取消し      F19=左        F20=右         F21=分割
```

*図 84. SRTSEQ=*HEX： ソート・シーケンスなしのソートを示す報告書の例*

JOB 列の値は、大文字と小文字の混用であることに注意してください。ここには、**Mgr**、**MGR**、および **mgr** という語があります。行は、JOB 列の値によってソートされますが、大文字の **MGR** は、小文字の **mgr** とは異なる値として扱われています。したがって、**Mgr**、**MGR**、および **mgr** は、隣接の行には表示されていません。また、**sales** は、 **Mgr** より小さい値として扱われています。

図 85 は、共用重みソート・シーケンスを使用したソートの結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   _______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

            ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
         -------  ---------  -------  -----  -------  -----------  ---------
000001       80   James          20   Clerk        0    13,504.60     128.20
000002       10   Sanders        20   Mgr          7    18,357.50        .00
000003       30   Marenghi       38   MGR          5    17,506.75        .00
000004       50   Hanes          15   Mgr         10    20,659.80        .00
000005      100   Plotz          42   mgr          7    18,352.80        .00
000006       20   Pernal         20   Sales        8    18,171.25     612.45
000007       40   OBrien         38   Sales        6    18,006.00     846.55
000008       60   Quigley        38   SALES        0    16,808.30     650.25
000009       70   Rothman        15   Sales        7    16,502.83   1,152.00
000010       90   Koonitz        42   sales        6    18,001.75   1,386.70
******  * * * * *  データの終わり  * * * * *



   F3=終了      F12=取消し      F19=左        F20=右         F21=分割
```

*図 85. SRTSEQ=*LANGIDSHR, LANGID=ENU： 共用重みソート・シーケンスを使用したソートを示す報告書の例*

行は、JOB 列の値によってソートされています。小文字は大文字と同じ値として扱われています。 246 ページの図 85 では、大文字と小文字の値 (**mgr**、 **Mgr**、および **MGR**) がグループになっていることに注意してください。

図 86 は、固有重みソート・シーケンスを使用したソートの結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

             ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
          -------  ---------  -------  -----  -------  -----------  ---------
000001       80   James          20   Clerk        0    13,504.60     128.20
000002      100   Plotz          42   mgr          7    18,352.80        .00
000003       10   Sanders        20   Mgr          7    18,357.50        .00
000004       50   Hanes          15   Mgr         10    20,659.80        .00
000005       30   Marenghi       38   MGR          5    17,506.75        .00
000006       90   Koonitz        42   sales        6    18,001.75   1,386.70
000007       20   Pernal         20   Sales        8    18,171.25     612.45
000008       40   OBrien         38   Sales        6    18,006.00     846.55
000009       70   Rothman        15   Sales        7    16,502.83   1,152.00
000010       60   Quigley        38   SALES        0    16,808.30     650.25
******  * * * * *  データの終わり  * * * * *



   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 86. SRTSEQ=*LANGIDUNQ, LANGID=ENU： 固有重みソート・シーケンスを使用したソートを示す報告書の例*

行は、JOB 列の値によってソートされています。これは、各国語辞書で使用されているソート方法です。小文字と大文字は固有の値として扱われますが、小文字と大文字が隣接したソート・シーケンスになるような重みが付けられています。このソート・シーケンスでは、小文字は大文字の前になります。

Query 管理機能でのレコード選択の例

次の SQL ステートメントによって、JOB 列に **MGR** の値が入っているレコードを示す報告書が作成されます。

```
SELECT * FROM STAFF WHERE JOB=MGR
```

248 ページの図 87 は、*HEX ソート・シーケンスによるレコード選択の結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

             ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
          -------  ---------  -------  -----  -------  -----------  ---------
000001       30   Marenghi       38   MGR          5    17,506.75        .00
******  * * * * *  データの終わり  * * * * *









   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 87. SRTSEQ=*HEX： ソート・シーケンスを使用したレコード選択を示す報告書の例*

図 87 では、JOB 列の選択基準に合致する行が選択されています。小文字の **mgr** は、大文字の **MGR** とは異なる値として扱われています。 **Mgr** と **mgr** という値の行は、選択されていません。 **MGR** という値の行だけが選択されています。

図 88 は、共用重み文字列順序の使用によるレコード選択の結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

             ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
          -------  ---------  -------  -----  -------  -----------  ---------
000001       10   Sanders        20   Mgr          7    18,357.50        .00
000002       30   Marenghi       38   MGR          5    17,506.75        .00
000003       50   Hanes          15   Mgr         10    20,659.80        .00
000004      100   Plotz          42   mgr          7    18,352.80        .00
******  * * * * *  データの終わり  * * * * *









   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 88. SRTSEQ=*LANGIDSHR, LANGID=ENU： 共用重みソート・シーケンスを使用したレコード選択を示す報告書の例*

248 ページの図 88 では、大文字と小文字を同じ値として扱い、 JOB 列のレコード選択基準に合致する行を選択しています。したがって、**mgr**、**Mgr**、および **MGR** のすべてが選択されています。

図 89 は、共用重み文字列順序によるレコード選択の結果です。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

             ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
          -------  ---------  -------  -----  -------  -----------  ---------
000001       30   Marenghi       38   MGR          5    17,506.75        .00
******  * * * * *  データの終わり  * * * * *









   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 89. SRTSEQ=*LANGIDUNQ, LANGID=ENU： 固有重みソート・シーケンスを使用したレコード選択を示す報告書の例*

図 89 では、小文字と大文字はそれぞれ固有の値として扱われ、したがって、小文字の **mgr** は、大文字の **MGR** と同じ値としては扱われません。したがって、**MGR** の行だけが選択されています。

Query 管理機能での報告書の切れ目の例

次の SQL ステートメントを使用すると、JOB 列の値を使ってソートされた報告書が作成されます。

```
SELECT * FROM STAFF ORDER BY JOB
```

250 ページの図 90 の報告書は、*HEX ソート・シーケンスによる報告書の切れ目を示しています。 JOB 列には BREAK1 が、SALARY 列には SUM が使用されています。

```
     ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
-------  ---------  -------  -----  -------  -----------  ---------
   100   Plotz          42   mgr          7    18,352.80        .00
                                            -----------
                                       *    18,352.80

    90   Koonitz        42   sales        6    18,001.75   1,386.70
                                            -----------
                                       *    18,001.75

    80   James          20   Clerk        0    13,504.60     128.20
                                            -----------
                                       *    13,504.60

    10   Sanders        20   Mgr          7    18,357.50        .00
    50   Hanes          15               10    20,659.80        .00
                                            -----------
                                       *    39,017.30

    30   Marenghi       38   MGR          5    17,506.75        .00
                                            -----------
                                       *    17,506.75

    20   Pernal         20   Sales        8    18,171.25     612.45
    40   OBrien         38                6    18,006.00     846.55
    70   Rothman        15                7    16,502.83   1,152.00
                                            -----------
                                       *    52,680.08

    60   Quigley        38   SALES        0    16,808.30     650.25
                                            -----------
                                       *    16,808.30
```

*図 90. SRTSEQ=*HEX： ソート・シーケンスを使用した報告書の切れ目を示す報告書の例*

図 90 では、行は、JOB 列の値による報告書の切れ目でグループ分けされています。この時、値の一部が大文字または小文字であるかどうかも、基準として使われています。大文字の **MGR** は、大文字と小文字を混用した **Mgr** と同じ値としては扱われず、したがって、同じ切れ目レベルには入れられていません。

251 ページの図 91 は、共用重みソート・シーケンスによる報告書の切れ目を示しています。

```
      ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
-------  ---------  -------  -----  -------  -----------  ---------
     80   James          20   Clerk        0    13,504.60     128.20
                                              -----------
                                           *    13,504.60

     10   Sanders        20   Mgr          7    18,357.50        .00
     30   Marenghi       38                5    17,506.75        .00
     50   Hanes          15               10    20,659.80        .00
    100   Plotz          42                7    18,352.80        .00
                                              -----------
                                           *    74,876.85

     20   Pernal         20   Sales        8    18,171.25     612.45
     40   OBrien         38                6    18,006.00     846.55
     60   Quigley        38                0    16,808.30     650.25
     70   Rothman        15                7    16,502.83   1,152.00
     90   Koonitz        42                6    18,001.75   1,386.70
                                              -----------
                                           *    87,490.13
```

*図 91. SRTSEQ=*LANGIDSHR, LANGID=ENU： 共用重みソート・シーケンスによる報告書の切れ目を示す報告書の例*

図 91 では、行は、JOB 列の値による切れ目でグループ化されています。小文字は大文字と同じ値として扱われています。したがって、これらの値 (**mgr**、**Mgr**、および **MGR**) はすべて同じ切れ目レベルに入れられています。

252 ページの図 92 は、固有重みソート・シーケンスによる報告書の切れ目を示しています。

252 ページの図 92 では、行は、JOB 列の値によってソートされています。小文字と大文字は固有の値として扱われますが、小文字と大文字が隣接したソート・シーケンスになるような重みが付けられています。このソート・シーケンスでは、小文字は大文字の前になります。行を切れ目レベルにグループ化する時、同じ値をもつ行だけがまとめられます。

```
     ID  NAME          DEPT  JOB      YEARS       SALARY      COMM
-------  ---------  -------  -----  -------  -----------  ---------
    80   James          20   Clerk        0    13,504.60     128.20
                                             -----------
                                         *    13,504.60

   100   Plotz          42   mgr          7    18,352.80        .00
                                             -----------
                                         *    18,352.80

    10   Sanders        20   Mgr          7    18,357.50        .00
    50   Hanes          15   Mgr         10    20,659.80        .00
                                             -----------
                                         *    39,017.30

    30   Marenghi       38   MGR          5    17,506.75        .00
                                             -----------
                                         *    17,506.75

    90   Koonitz        42   sales        6    18,001.75   1,386.70
                                             -----------
                                         *    18,001.75

    20   Pernal         20   Sales        8    18,171.25     612.45
    40   OBrien         38   Sales        6    18,006.00     846.55
    70   Rothman        15   Sales        7    16,502.83   1,152.00
                                             -----------
                                         *    52,680.13

    60   Quigley        38   SALES        0    16,808.30     650.25
                                             -----------
                                         *    16,808.30
```

*図 92. SRTSEQ=*LANGIDUNQ, LANGID=ENU： 固有重みソート・シーケンスによる報告書の切れ目を示す報告書の例*

Query 管理機能でのグループ化の例

次の SQL ステートメントを使用すると、JOB 列の値ごとに合計データがまとめられます。

```
SELECT JOB, SUM(SALARY) FROM STAFF GROUP BY JOB ORDER BY JOB
```

253 ページの図 93 は、*HEX ソート・シーケンスによる Query のグループ化を示しています。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

         JOB                               SUM ( SALARY )
         -----  ----------------------------------------
000001   mgr                                  18,352.80
000002   sales                                18,001.75
000003   Clerk                                13,504.60
000004   Mgr                                  39,017.30
000005   MGR                                  17,506.75
000006   Sales                                52,680.08
000007   SALES                                16,808.30
******  * * * * *  データの終わり  * * * * *




   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 93. SRTSEQ=*HEX： ソート・シーケンスを使用したグループ化を示す報告書の例*

図 93 では、JOB 列の値によって報告書に切れ目が付けられ、行がグループ化されています。 **MGR** は、**Mgr** と同じ値としては扱われないので、これらは同じ切れ目レベルには入れられていません。

図 94 は、共用ソート・シーケンスによるグループ化を示しています。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

         JOB                               SUM ( SALARY )
         -----  ----------------------------------------
000001   Clerk                                13,504.60
000002   Mgr                                  74,876.85
000003   Sales                                87,490.13
******  * * * * *  データの終わり  * * * * *









   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 94. SRTSEQ=*LANGIDSHR, LANGID=ENU： 共用重みソート・シーケンスによるグループ化を示す報告書の例*

図 94 では、行は、JOB 列の値による切れ目でグループ化されています。小文字は大文字と同じ値として扱われています。したがって、これらのすべての値 (**mgr**、**Mgr**、および **MGR**) が同じグループに入れられています。

図 95 は、固有重みソート・シーケンスによるグループ化を示しています。

```
                              報告書の表示
Query . . . . .:   *                        幅  . . . .:      71
書式  . . . . .:   *                        桁  . . . .:       1
制御 . . . . . .   ______
行     ....+....1....+....2....+....3....+....4....+....5....+....6....

         JOB                                SUM ( SALARY )
         -----  ----------------------------------------
000001   Clerk                                 13,504.60
000002   mgr                                   18,352.80
000003   Mgr                                   39,017.30
000004   MGR                                   17,506.75
000005   sales                                 18,001.75
000006   Sales                                 52,680.13
000006   SALES                                 16,808.30
******  * * * * *  データの終わり  * * * * *




   F3=終了       F12=取消し       F19=左         F20=右          F21=分割
```

*図 95. SRTSEQ=*LANGIDUNQ, LANGID=ENU： 固有重みソート・シーケンスによるグループ化を示す報告書の例*

図 95 では、行は、JOB 列の値によってソートされています。小文字と大文字はそれぞれ固有の値として扱われますが、それらが隣接したソート・シーケンスになるように重みが付けられています。小文字は、大文字の前にソートされます。したがって、順序上、値 **mgr** は、値 **Mgr** の前に現れ、また、値 **Mgr** は、値 **MGR** の前になります。

行を切れ目レベルにグループ化する時、同じ値をもつ行だけがまとめられます。

Query 管理機能での切れ目合計の使用

253 ページの図 93 では、JOB 列の値によって報告書に切れ目が付けられ、行がグループ化されています。大文字の **MGR** は、大文字と小文字を混用している **Mgr** と同じ値としては扱われていません。したがって、**MGR** と **Mgr** は、同じ切れ目レベルにはグループ化されていません。

付録 E. 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものであり、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、プログラムまたはサービスの操作性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。実施権の許諾については、下記の宛先に書面にてご照会ください。

〒106-0032 東京都港区六本木 3 丁目 2-31
IBM World Trade Asia Corporation
Intellectual Property Law & 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で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

本書は定期的に見直され、必要な変更 (たとえば、技術的に不適切な表現や誤植など) 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。

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 Corporation
Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901-7829
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

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にお願いします。

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、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名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。お客様は、IBM 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。

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的創作物にも、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.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

商標

以下は、IBM Corporation の商標です。

C/400
DB2
Distributed Relational Database Architecture
DRDA
e (logo) IBM
IBM
iSeries
MVS
オペレーティング・システム/400
OS/400
QMF
RPG/400
System/36
SQL/DS

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。

他の会社名、製品名およびサービス名などはそれぞれ各社の商標または登録商標です。

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]

[カ行]

［サ行］

[タ行]

［ナ行］

［ハ行］

[マ行]

［ヤ行］

［ラ行］

B

C

D

E

F

G

H

I

K

L

N

O

P

Q

R

S

T

U

V

W

［特殊文字］

Printed in Japan

SC88-4018-00

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