IBM® DB2® Universal Database

應用程式開發手冊

版本 7

SC40-0493-01

IBM® DB2® Universal Database

應用程式開發手冊

版本 7

SC40-0493-01

使用此資訊及其支援的產品之前，請先閱讀第395頁的『附錄E. 注意事項』下的一般資訊。

目錄

歡迎使用 DB2 應用程式開發

註: 位於行之左邊距的修訂列 "|"，指示自本書首次隨 DB2 Universal Database 版本 7 出版後對該行文字所做的新增或修改。章節標題旁的修訂列 "|" 指示該章節包含經修訂的文字。

本序文提供您使用 DB2 應用程式開發所需要入門資訊，特別是 DB2 Developer's Edition 產品。

『DB2 Developer's Edition』一節，提供針對您的特定開發需求安裝資訊。此資訊會協助您決定如何從 IBM DB2 Universal Developer's Edition 版本 7，或 IBM DB2 Personal Developer's Edition 版本 7 來安裝 DB2。

第ix頁的『DB2 應用程式開發書籍』一節，說明 DB2 檔案庫中使用在應用程式開發的一些主要書籍。

第x頁的『DB2 程式設計介面』一節，描述 DB2 應用程式開發的主要程式設計介面概念。

第xix頁的『DB2 特性』一節，說明 DB2 應用程式開發可供您使用的主要特性。

DB2 Developer's Edition

DB2 Universal Database 提供了兩個用於應用程式開發的產品資料包： DB2 Personal Developer's Edition 和 DB2 Universal Developer's Edition。 Personal Developer's Edition 提供可在 OS/2、Linux 及 Windows 32 位元作業系統中執行的 DB2 Universal Database 及 DB2 Connect Personal Edition 產品。 DB2 Universal Developer's Edition 提供了適用於這些平台及 AIX、HP-UX、PTX、Silicon Graphics IRIX、及 Solaris** Operating Environment** 上的 DB2 產品。

使用這些產品隨附的軟體，您可以開發及測試在一個作業系統上執行的應用程式，並存取同一個或不同作業系統上的資料庫。例如，您可以建立在 Windows NT 作業系統上執行的應用程式，也可以存取 UNIX 平台 (如 AIX) 上的資料庫。請參閱您的產品授權合約，以了解 Developer's Edition 產品的使用條款與條件。

Personal Developer's Edition 包含幾個 CD-ROM，這些 CD-ROM 含有您開發及測試應用程式需要的所有程式碼。在每一個包裝盒內，您會看到下列項目：

- 適用於 OS/2、Linux 及 Windows 作業系統的 DB2 Universal Database 產品 CD-ROM。每一個 CD-ROM 均內含適用於受支援作業系統的 DB2 伺服器、

Administration Client、Application Development Client 及 Run-Time Client。這些光碟僅供您作爲應用程式測試之用。如果您需要安裝及使用資料庫，請購買 Universal Database 產品，以取得有效的產品授權。

- DB2 Connect Personal Edition
- DB2 出版品 CD-ROM，內含 PDF 格式的 DB2 書籍
- DB2 Extenders (僅限 OS/2 及 Windows)
- DB2 XML Extender (僅限 Windows)
- DB2 OLAP Starter Kit (僅限 Windows)。在安裝 OLAP Starter Kit 之前，您必須先安裝 DB2 伺服器。
- VisualAge for Java，Entry Edition

Universal Developer's Edition 內含可用於 DB2 支援之所有作業系統的 CD-ROM，包含下列內容：

- DB2 Universal Database Personal Edition、Workgroup Edition 及 Enterprise Edition
- DB2 Connect Personal Edition 及 DB2 Connect Enterprise Edition
- 適用於所有平台的 Administration Client。這些從屬站含有管理資料庫的工具，如「控制中心」及「事件分析程式」。這些從屬站亦可讓您在任何系統上執行應用程式。
- 適用於所有平台的 Application Development Client。這些從屬站有應用程式開發工具、範例程式及標頭檔。每一個 DB2 AD Client 包括開發應用程式所需的所有項目。若需 AD Client 的詳細資訊，請參閱第4頁的『關於 DB2 Application Development Client』。
- 適用於所有平台的 Run-Time Client。您可以在任何系統上從 Run-Time Client 執行應用程式。 Run-Time Client 沒有 Administration Client 的某些功能，如「DB2 控制中心」和「事件分析程式」，因此所需的空間較小。
- DB2 Satellite Edition
- DB2 Extenders
- DB2 XML Extender
- DB2 OLAP Starter Kit
- Net.Data
- VisualAge for Java，Professional Edition (OS/2 及 Windows)
- Websphere Studio
- Websphere Application Server，Standard Edition
- Query Management Facility (試用後再購買)

此外，兩種 Developer's Edition 均含有對您在開發應用程式上有幫助的其它軟體。本軟體會時常更新，並附有使用的授權合約。

安裝資訊

每一個 DB2 產品 CD-ROM 都含有安裝資訊，您可以從 CD-ROM 中直接以 HTML 檢視。隨附有適用於受支援平台的*快速入門*書籍，說明安裝 DB2 伺服器、Administration Client、Application Development Client 及 Run-Time Client 的方法。

您也可在*安裝與架構補充資料*取得其它資訊。本書僅可從 Client CD-ROM 中檢視。

您要的書籍會位於您的使用之語言的次目錄中。CD-ROM README.TXT 檔會告知您書籍檔在 CD-ROM 中的位置。

執行您的瀏覽器，並按一下書籍次目錄中的 `index.htm` 檔。即會出現 *快速入門* 及 *安裝與架構補充資料* 書籍的次目錄。

db2i2 *DB2 for OS/2 快速入門*

db2ix *DB2 for UNIX 快速入門*

db2i6 *DB2 for Windows 快速入門*

db2iy *安裝與架構補充資料*

DB2 出版品 CD-ROM 含有本產品隨附的所有 DB2 書籍的 PDF 檔。您可以使用 Adobe Acrobat Reader，直接自 CD-ROM 檢視這些書籍。適用您語言的 DB2 書籍是位於適當的語言次目錄中。若需安裝資訊，您可以存取*快速入門*及*安裝與架構補充資料*書籍。其檔名皆以上述列示的字元組起首。

若需列印 PDF 檔的相關指示，請參閱第385頁的『列印 PDF 書籍』。

請參閱 CD-ROM 上的 README.TXT 檔，以取得如何存取書籍檔案的完整詳細資訊。

DB2 應用程式開發書籍

DB2 檔案庫說明於第377頁的『附錄D. 使用 DB2 檔案庫』中。DB2 書籍分成兩大種類：一種提供有關管理 DB2 資料庫的資訊，另一種提供有關 DB2 應用程式開發的資訊。某些書籍則同時提供這兩種資訊。身爲 DB2 應用程式開發者，您會發現這兩類都是您所要參考的 DB2 書籍。不過，您和本節的主要焦點是在 DB2 檔案庫的主要應用程式開發書籍上。

其中有兩本主要書籍可使用在應用程式設計上。 *CLI Guide and Reference* 主要討論 DB2 CLI 應用程式設計，而 *Application Development Guide* 則是討論 DB2 CLI 以外各種不同種類的 DB2 程式設計。這兩本書籍也包含參考資料。

您可以在本書*應用程式開發手冊*中找到設置開發環境，以及編譯、鏈結與執行應用程式所需的資訊。

有關 SQL 陳述式及函數的語法，請參閱 *SQL Reference*。

被視爲 DB2 檔案庫管理類別的兩本書籍，也是您應用程式設計上重要的參考書。*Administrative API Reference* 內含管理 DB2 資料庫使用的所有管理功能之詳細資訊。您會發現在應用程式中同時使用 DB2 API 和 SQL 陳述式或單獨使用 DB2 API，非常方便。 *Command Reference* 內含所有 DB2 指令 (SQL 陳述式除外) 的詳細資訊，以及說明如何使用「DB2 命令行處理器 (CLP)」。

若要解決環境設置或程式開發上的問題，請參閱 *Troubleshooting Guide*。它可在您洽詢「DB2 客戶服務中心」時，協助判斷錯誤來源、從問題回復及使用偵錯工具程式。 *訊息參考手冊*內含 DB2 錯誤訊息的完整列示與說明，它是您在除錯應用程式時非常有用的一個參照。

這些和其它 DB2 檔案庫的書籍，以及適合您 DB2 環境的線上資訊，可提供您開發 DB2 應用程式所需的資訊。有關並不完全與 DB2 相關的開發資訊，請參閱廠商所提供關於您所使用編譯器、直譯器或其它開發工具的文件。

DB2 程式設計介面

您可以使用數種不同的程式設計介面，管理或存取 DB2 資料庫。 方法是：

1. 使用 DB2 API 來執行管理功能，如備份及復置資料庫。
2. 將靜態與動態 SQL 陳述式嵌入您的應用程式中。
3. 在您的應用程式中編碼 DB2 CLI 函數呼叫，來呼叫動態 SQL 陳述式。
4. 開發 Java 應用程式和 applet，呼叫 Java Database Connectivity 應用程式設計介面 (JDBC API)。
5. 開發符合「資料存取物件 (DAO)」與「遠端資料物件 (RDO)」規格的 Microsoft Visual Basic 與 Visual C++ 應用程式，以及使用「物件連結和內嵌資料庫橋站 (OLE DB)」的「ActiveX 資料物件 (ADO)」應用程式。
6. 使用 IBM 或其它廠商工具 (例如 Net.Data、Excel、Perl)，及 ODBC 一般使用者工具 (例如 Lotus Approach 及其程式設計語言 LotusScript) 來開發應用程式。

應用程式存取 DB2 資料庫的方法視所要開發的應用程式類型而定。例如，如果您要的是資料登錄應用程式，可以選擇將靜態 SQL 陳述式嵌入應用程式中。如果您要的是可在全球資訊網 (WWW) 執行查詢的應用程式，可以選擇 Net.Data、Perl 或 Java。

使用內含的 SQL 陳述式

「結構化查詢語言 (SQL)」是一種用來在 DB2 資料庫中存取及操作資料的資料庫介面語言。您可以將 SQL 陳述式嵌入應用程式中，讓它們執行 SQL 支援的任何作業，例如擷取或儲存資料。您可以利用 DB2 來以 C/C++、COBOL、FORTRAN、Java (SQLJ)，以及 REXX 程式設計語言來編碼您內含的 SQL 應用程式。

內含 SQL 陳述式的應用程式稱作主程式。用來建立主程式的程式設計語言稱作主語言。主程式和主語言因爲可加入 SQL 陳述式，所以用這種方法來定義。

若是靜態 SQL 陳述式，您在編譯之前會知道 SQL 陳述式類型，及表格與直欄名稱。唯一不知道的是陳述式搜尋或更新的特定資料值。您可以用主語言變數來代表那些值。在執行應用程式之前，您必須先前置編譯、連結，然後編譯靜態 SQL 陳述式。 靜態 SQL 最好在其統計值不會有大幅變更的資料庫上執行。否則，陳述式很快就會過期。

相反的，動態 SQL 陳述式則是您應用程式在運作時所建置和執行的陳述式。交談式應用程式可提示一般使用者要求輸入 SQL 陳述式的關鍵部份，例如要搜尋的表格與直欄名稱，是一個不錯的動態 SQL 範例。應用程式會在執行時建立 SQL 陳述式，然後將陳述式提出來進行處理。

您可以撰寫具有靜態 SQL 陳述式、動態 SQL 陳述式或兩者皆有的應用程式。

通常，靜態 SQL 陳述式較適合具有預設異動的高效能應用程式。預約系統是這類應用程式一個不錯的範例。

通常，動態 SQL 陳述式非常適合對快速變更的資料庫執行的應用程式，在執行時需要指定這種資料庫的異動。交談式查詢介面是這類應用程式一個不錯的範例。

當您將 SQL 陳述式嵌入應用程式時，您必須將應用程式前置編譯及連結到資料庫，步驟如下：

1. 建立具有內含的 SQL 陳述式之程式的來源檔。
2. 連接資料庫，然後對每一個來源檔進行前置編譯。

   前置編譯器可將每一個來源檔內的 SQL 陳述式轉換成 DB2 執行 API，以呼叫資料庫管理程式。前置編譯器也會在資料庫中產生一個存取資料包，另外如果您指定要建立連結檔案也會產生連結檔案。

存取資料包含有 DB2 最佳化工具選擇使用於您應用程式中的靜態 SQL 陳述式的存取計劃。存取計劃包含資料庫管理程式以最佳化工具所決定的最有效方式，來執行靜態 SQL 陳述式時所需的資訊。若是動態 SQL 陳述式，最佳化工具會在您執行應用程式時建立存取計劃。

連結檔案含有 SQL 陳述式及其它建立存取資料包所需的資料。您可以使用連結檔案於稍後重新連結應用程式，不需要先對應用程式進行前置編譯。重新連結會產生最適合目前資料庫狀況的存取計劃。如果前置編譯過的應用程式將存取不同的資料庫，則您必須重新連結您的應用程式。 如果資料庫統計值自前次連結後有變動，您應重新連結您的應用程式。

3. 使用主語言編譯器來編譯修改過的來源檔 (及其它不含 SQL 陳述式的檔案)。
4. 將目的檔與 DB2 和主語言檔案庫鏈結，即產生可執行的程式。
5. 如果在前置編譯時未建立存取資料包，或者要存取不同的資料庫，請連結此連結檔案以建立存取資料包。
6. 執行應用程式。此應用程式可使用資料包內的存取計劃來存取資料庫。

內含的 SQL for Java (SQLJ)

DB2 Java 內含的 SQL (SQLJ) 支援是由 DB2 AD Client 提供。有了 DB2 SQLJ 支援和 DB2 JDBC 支援，您便可建置和執行 SQLJ applet、應用程式及儲存程序。其中含有靜態 SQL，並使用連結到 DB2 資料庫的內含 SQL 陳述式。

有關 DB2 SQLJ 支援的詳細資訊，請造訪下列網頁：

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

使用 DB2 CLI

DB2 CLI 是一種程式設計介面，您的 C 和 C++ 應用程式可使用它來存取 DB2 資料庫。 DB2 CLI 以 Microsoft Open Database Connectivity (ODBC) 規格及 ISO CLI 標準爲基礎。因爲 DB2 CLI 是以業界標準爲基礎，所以已經熟悉這些資料庫介面的應用程式設計師，在學習上更加能夠得心應手。

在使用 DB2 CLI 時，您的應用程式會將作爲函數引數的動態 SQL 陳述式傳給資料庫管理程式處理。因此，DB2 CLI 是嵌入動態 SQL 的另一個選擇。

您也可以在 CLI、ODBC 或 JDBC 應用程式中，將 SQL 陳述式當作靜態 SQL 執行。 CLI/ODBC/JDBC「靜態剪影」特性可以讓應用程式的一般使用者在許多情況下，以靜態 SQL 取代動態 SQL 的使用。若需其餘相關資訊，請參閱：

```
http://www.ibm.com/software/data/db2/udb/staticcli
```

您可以建置 ODBC 應用程式而不使用 ODBC 驅動程式管理程式，或鏈結您的應用程式與 UNIX 中的 `libdb2`，及 OS/2 及 Windows 32 位元作業系統中的

db2cli.lib，在任何平台中只使用 DB2 的 ODBC 驅動程式。 DB2 CLI 範例程式會示範此動作。範例程式位於 UNIX 中的 sqllib/samples/cli，及 OS/2 及 Windows 32 位元作業系統中的 %DB2PATH%\samples\cli。

您不需要前置編譯或連結 DB2 CLI 應用程式，因爲它們使用 DB2 提供的一般存取資料包。您只要編譯及鏈結您的應用程式即可。

然而，在您的 DB2 CLI 或 ODBC 應用程式可以存取 DB2 資料庫之前， DB2 DB2 AD Client 隨附的 DB2 CLI 連結檔案必須連接到每一個要存取的 DB2 資料庫。雖然第一次執行陳述式時會自動執行這個動作，但我們建議由資料庫管理員從要存取 DB2 資料庫之每一平台的其中一個從屬站來連結這些連結檔案。 有關連結指示，請參閱第42頁的『連結』。

例如，假設有 OS/2、AIX 和 Windows 95 從屬站，各要存取兩個 DB2 資料庫。管理者應該對要存取的每一個資料庫從其中一個 OS/2 從屬站來連結這些連結檔案。然後，管理者應該從要被存取的每一個資料庫上的其中一個 AIX 從屬站來連結這些連結檔案。最後，管理者再對 Windows 95 從屬站執行如上的動作。

DB2 CLI 對內含的動態 SQL

您可以使用內含的動態 SQL 陳述式或 DB2 CLI 來開發動態應用程式。在這兩種方式下，SQL 陳述式在執行時期已備妥及處理。每一個方法各有列示如下的優點。

DB2 CLI 優點

可移轉性

DB2 CLI 應用程式使用一組標準函數，將 SQL 陳述式傳遞給資料庫。在執行 DB2 CLI 應用程式之前，您所要做的是編譯及鏈結它們。相反的，您必須在執行內含的 SQL 應用程式之前，先對它們進行前置編譯、編譯，再將它們連結到資料庫。此處理可將應用程式有效地連結到特定的資料庫。

不需連結

您不需要分別將 DB2 CLI 應用程式連結到所要存取的每一個資料庫。您只需要將 DB2 CLI 提供的連結檔案一次連結到所有 DB2 CLI 應用程式即可。如此可大幅降低管理應用程式所耗費的時間。

擴充提取與輸入

DB2 CLI 函數可讓您將資料庫中多重列擷取到含有單一呼叫的陣列。也可讓您使用輸入變數陣列來多次執行 SQL 陳述式。

介面與目錄一致

資料庫系統包含一些目錄表格，這些目錄表格具有資料庫及其使用者的相關資訊。這些目錄表格依使用的系統而有不同。DB2 CLI 提供一致的介面

來查詢有關元件的目錄資訊，這些元件如表格、直欄、外部和主要鍵，以及使用者專用權。這可讓您的應用程式免於因資料庫伺服器不同版次而變更目錄，以及因不同資料庫伺服器之間所產生的差異。您不需要對特定的伺服器或產品版本撰寫目錄查詢。

擴充資料轉換

DB2 CLI 可在 SQL 和 C 資料類型之間自動轉換資料。例如，若提取任何 SQL 資料類型成爲 C 字元資料類型，則會將它轉換成字串表示。如此可讓 DB2 CLI 適用於交談式查詢應用程式。

無廣域資料區

若使用 DB2 CLI，則毋需應用程式控制的複雜廣域資料區 (例如 SQLDA 和 SQLCA)，這通常爲內含的 SQL 應用程式所需。而 DB2 CLI 會自動配置及控制必要的資料結構，以及提供 handle 讓應用程式參照它們。

擷取儲存程序的結果集

DB2 CLI 應用程式可擷取儲存程序 (位於伺服器) 產生的多重列及結果集。

可捲動的游標

DB2 CLI 支援可與陣列輸出一起使用的伺服器端可捲動游標。這在 GUI 應用程式中非常有用，您可以使用 Page Up、Page Down、 Home 和 End 按鍵來顯示捲動框內的資料庫資訊。您可以宣告游標爲可捲動的，然後在一或多個列的結果集間來回移動。您亦可藉由從現行列、結果集的開頭或結尾，或先前標示的特定列，指定偏移來提取橫列。

內含的動態 SQL 優點

所有的 DB2 CLI 使用者均共用相同的專用權。內含的 SQL 提供的優點爲：藉由對資料包的特定使用者授與執行專用權，可具有更多的機密保護。

內含的 SQL 不僅支援 C 和 C++。對於偏好以另一種語言來編碼應用程式的使用者，這會是一個優點。

動態 SQL 通常會與靜態 SQL 更加一致。如果您已知道如何規劃靜態 SQL，則將它移至動態 SQL 就不會像移至 DB2 CLI 那樣困難了。

使用 Java Database Connectivity (JDBC)

DB2 的 Java 支援包括 JDBC，它是一種廠商中立動態 SQL 介面，透過標準化 Java 方法提供應用程式的資料存取權。 JDBC 類似 DB2 CLI，您不需要前置編譯或連結 JDBC 程式。利用廠商中立標準，JDBC 應用程式提供增加可移轉性。使用 JDBC 撰寫應用程式僅適用動態 SQL。

JDBC 對於透過網際網路存取 DB2 資料庫特別有用。使用 Java 程式設計語言，您可以開發 JDBC Applet 和應用程式，利用網路連接來存取及操作遠端 DB2 資料庫內的資料。您亦可建立位於伺服器的 JDBC 儲存程序、存取資料庫伺服器，以及將資訊傳回呼叫儲存程序的遠端從屬站應用程式。

JDBC API 類似 CLI/ODBC API，可提供從 Java 程式碼存取資料庫的標準方法。您的 Java 程式碼會把作爲方法引數的 SQL 陳述式傳至 DB2 JDBC 驅動程式。此驅動程式處理來自您從屬站 Java 程式碼的 JDBC API 呼叫。

Java 的可移轉性可讓您將 DB2 存取權釋放給多重平台上的從屬站，僅需要一個 Java 型 Web 瀏覽器即可。

Java 應用程式依賴 DB2 從屬站來連接 DB2。從桌上管理程式或命令行中啓動您的應用程式，就像啓動任何其它應用程式一般。 DB2 JDBC 驅動程式會處理來自您應用程式的 JDBC API 呼叫，並使用從屬站連接將要求告知伺服器，以及接收結果。

Java Applet 不需要 DB2 從屬站連接。通常，您會將 Applet 嵌入「超文字標示語言 (HTML)」網頁中。

只要在從屬站機器上有一個 Java 啓用的 Web 瀏覽器或 Applet 檢視器，您就可以執行 applet。當載入您的 HTML 頁時，瀏覽器會將 Java Applet 下載至您的機器，然後下載 Java 類別檔和 DB2 的 JDBC 驅動程式。當您的 Applet 呼叫 JDBC API 連接 DB2 時， JDBC 驅動程式會透過常駐在 web 伺服器的 JDBC Applet 伺服器，建立與 DB2 資料庫的個別網路連接。

有關 DB2 JDBC 支援的詳細資訊，請造訪下列網頁：

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

使用 DB2 API

您的應用程式可能需要執行某些資料庫管理作業，如建立、啓動、備份或復置資料庫。 DB2 提供很多的 API，供您在您的應用程式 (包括內含的 SQL 與 DB2 CLI 應用程式) 中執行這些作業。這可讓您將相同的管理功能規劃到應用程式中，您可以利用 DB2 伺服器管理工具來執行，如第xxiv頁的『DB2 Universal Database 工具』中所述。

此外，您可能需要執行只能透過 DB2 API 執行的特定作業。例如，您可能要擷取錯誤訊息文字，讓應用程式可將它顯示給一般使用者查閱。若要擷取訊息，您必須使用「取得錯誤訊息 API」。

使用「ActiveX 資料物件」(ADO) 及「遠端資料物件」(RDO)

您可以撰寫 Microsoft Visual Basic 和 Microsoft Visual C++ 資料庫應用程式，確認「資料存取物件 (DAO)」與「遠端資料物件 (RDO)」規格。DB2 亦支援使用 Microsoft OLE DB to ODBC Bridge 的「ActiveX 資料物件」(ADO) 應用程式。

「ActiveX 資料物件 (ADO)」可讓您透過 OLE DB 提供者，撰寫應用程式來存取及操作資料庫伺服器內的資料。 ADO 的主要優點是快速開發、容易使用及少用磁碟。

「遠端資料物件 (RDO)」提供透過 ODBC 存取遠端資料來源的資訊模式。 RDO 提供一組物件，使容易連接資料庫、執行查詢和儲存程序、操作結果，以及 COMMIT 伺服器變更。它是特別設計來存取遠端 ODBC 相關資料來源，以及使其便於使用 ODBC，而不需要複雜的應用程式碼。

使用 Perl 介面

DB2 支援「Perl 資料庫介面 (DBI)」規格，可透過 DBD::DB2 驅動程式進行資料存取。 DB2 Universal Database Perl DBI 網站的網址如下：

```
http://www.ibm.com/software/data/db2/perl/
```

並且包含最新的 DBD::DB2 驅動程式以及相關資訊。

使用 ODBC 一般使用者工具

您亦可使用 ODBC 一般使用者工具，如 Lotus Approach、 Microsoft Access 和 Microsoft Visual Basic，來建立應用程式。ODBC 工具提供您開發應用程式的另一個選擇，它比使用高階程式設計語言在操作上更爲簡單。

Lotus Approach 提供兩個方法來存取 DB2 資料。您可以使用圖形介面來執行查詢、開發報告及分析資料。或者，您可以使用 LotusScript 來開發應用程式，LotusScript 是一個功能完整的物件導向程式設計語言，包含物件、事件、方法和內容的寬陣列，以及內建程式編輯器。

用來建置 Web 應用程式的工具

DB2 Universal Database 支援所有的主要「網際網路」標準，是用於 Web 的理想資料庫。結合關聯式資料庫的延展性及可用性的性質，它具有存取記憶體的速度，方便網際網路搜尋及複雜文字對應。DB2 Universal Database 支援 WebSphere、Java 及 XML Extender，以便您部署電子商務應用程式。

DB2 Universal Developer's Edition 的數個工具提供 Web 賦能支援。VisualAge for Java 是整合的開發環境 (IDE)，可讓您建置、測試 Java 應用程式，以及將它們部署到 WebSphere Application Server 及 DB2 Universal Database。WebSphere Studio

是一個工具套件，將網站開發的全部工作整合成一個普通介面。WebSphere Application Server Standard Edition 爲電子商務應用程式提供了良好的部署環境。其元件可讓您快速且輕鬆地建置及部署個人化動態 Web 內容。

VisualAge for Java

VisualAge for Java Professional Edition (在 OS/2 及 Windows 上隨 DB2 Universal Developer's Edition 提供) 之特性及效能的增進，使建立可縮放且功能強大的電子商務應用程式的工作較以往更容易。它與 IBM WebSphereApplication Server、WebSphere Studio 及 DB2 Universal Database 的緊密整合可縮短開發時間，增進生產力，同時也使對企業資料的存取更簡便且安全。

IBM Data Access JavaBeans (VisualAge for Java 的選用安裝特性) 可讓應用程式開發者輕鬆存取啓用 JDBC 的關聯式資料庫。com.ibm.db 資料包中的 IBM Data Access JavaBeans 包括可簡化對關聯式資料庫存取的類別，並提供以下增強的特性：

- 快取查詢結果
- 透過結果快取更新
- 查詢參數支援
- Meta 資料支援

WebSphere Studio

WebSphere Studio 是一個工具套件，它將網站開發的全部工作整合成一個普通介面。WebSphere Studio 使合作建立、組譯、發佈及維護動態交談式 Web 應用程式更容易。Studio 由 Workbench、Page Designer、Remote Debugger 及精靈組成，並包含附屬 Web 開發產品 (如 Macromedia Flash、Fireworks、Freehand 及 Director) 的試用版。WebSphere Studio 可讓您隨心所欲創建交談式網站，以支援您的進階業務功能。

WebSphere Application Server Standard Edition (由 DB2 Universal Developer's Edition 提供) 是 WebSphere Studio 的一個元件。它將伺服器端商業應用程式的可移轉性與 Java 技術的效能及管理性相結合，爲設計基於 Java 的 Web 應用程式提供了一個綜合性平台。它可與企業資料庫及異動系統進行有效地互動。您可以在與 WebSphere Application Server 相同的機器、或在另一台 Web 伺服器上執行 DB2 伺服器。

WebSphere Application Server Advanced Edition (不在 DB2 Universal Developer's Edition 中) 提供對 Enterprise JavaBean 應用程式的額外支援。 DB2 Universal Database 由 WebSphere Application Server Advanced Edition 提供，用作管理伺服

器儲存庫。它對依照 Sun Microsystems 的「EJB 規格」所建置的應用程式提供伺服器功能，從而支援將 Web 應用程式整合到非 Web 商業系統中。

XML Extender

Extensible Markup Language (XML) 是用於應用程式間資料交換之已接受的標準技術。XML 文件是一種人類可讀的標示文件。其文字由字元資料及標記標示組成。標記標示可由文件的作者定義。「文件類型定義 (DTD)」可用來宣告標記定義及限制。DB2 XML Extender (隨 DB2 Universal Developer's Edition 及 Windows 上的 Personal Developer's Edition 提供) 爲程式提供了使用 SQL 延伸功能操作 XML 資料的機制。

DB2 XML Extender 引入了三種新的資料類型：`XMLVARCHAR`、`XMLCLOB` 及 `XMLFILE`。Extender 提供的 UDF 可儲存、取出及更新單一或多重直欄及表格中的 XML 文件。可使用位置路徑在整個 XML 文件、或根據結構式元件進行搜尋，搜尋所使用的是 Extensible Stylesheet Language Transformation (XSLT) 及 XPath for XML Path Language 的子集。

爲便於將 XML 文件儲存成一組直欄，DB2 XML Extender 提供了一個管理工具，可協助設計者進行 XML 到關聯式資料庫的對映。「文件存取定義 (DAD)」可用來維護 XML 文件的結構式及對映資料。DAD 被定義及儲存成 XML 文件，易於操作及理解。新的儲存程序可用來撰寫或分解該文件。

有關 DB2 XML Extender 的詳細資訊，請造訪：

```
http://www.ibm.com/software/data/db2/extenders/xmlext/index.html
```

MQSeries 賦能

隨 DB2 Universal Database 提供的一組 MQSeries 功能，可讓 DB2 應用程式與非同步傳訊作業相互作用。這表示 MQSeries 支援可用於以任何 DB2 支援之程式設計語言撰寫的應用程式。

在基本架構中，MQSeries 伺服器與 DB2 Universal Database 位於同一資料庫伺服器機器上。可透過 DB2 伺服器使用 MQSeries 功能，進而存取其它的 MQSeries 應用程式。多個 DB2 從屬站可透過資料庫同時存取 MQSeries 功能。MQSeries 作業可讓 DB2 應用程式與其它 MQSeries 應用程式進行非同步通信。例如，新功能所提供的簡單方法，可讓 DB2 應用程式將資料庫事件發佈到遠端 MQSeries 應用程式、透過選用的 MQSeries Workflow 產品起始工作流程，或者利用選用的 MQSeries Integrator 產品與現存的應用程式資料包通信。

Net.Data

Net.Data 可讓網際網路和企業內部網路透過您的 web 應用程式存取 DB2 資料。它利用 web 伺服器介面 (API)，提供比通用閘道介面 (CGI) 應用程式更高的效能。Net.Data 支援使用 Java、REXX、Perl 和 C++ 語言的從屬站端處理程序以及伺服器端處理程序。 Net.Data 提供條件性邏輯和豐富的巨集語言。同時也提供了 XML 支援，可讓您產生 XML 標示，作爲您 Net.Data 巨集的輸出，而無需手動輸入標示。您也可指定 XML 樣式表 (XSL)，用來格式化及顯示產生的輸出。Net.Data 網頁爲：

```
http://www.ibm.com/software/data/net.data/
```

DB2 特性

DB2 包含各種在伺服器上執行的特性，您可以用來補充或擴充您的應用程式。當使用 DB2 特性時，您不需要撰寫自己的程式碼來執行相同作業。 DB2 也可讓您在伺服器儲存部份程式碼，而不必將所有程式碼都保留在從屬站應用程式。如此可增加執行的效能以及方便維護。

有一些特性可保護資料及定義資料之間的關係。也有一些物件導向特性可建立具彈性的進階應用程式。您可以利用一個以上的方法來使用某些特性。例如，限制讓您保護資料及定義資料值之間的關係。下列是某些主要的 DB2 特性：

- 限制
- 使用者定義類型 (UDT) 和大型物件 (LOB)
- 使用者定義的函數 (UDF)
- 觸發函式
- 儲存程序

若要決定是否使用 DB2 特性，請注意下列各點：

應用程式獨立性

您可以將應用程式與它處理的資料獨立分開。利用在資料庫上執行的 DB2 特性，可讓您維護及變更資料相關邏輯，而不影響您的應用程式。如果您必須變更此邏輯，則只需要在一個地方變更它，那就是伺服器，而不是存取資料的每一個應用程式。

效能 藉由在伺服器儲存及執行應用程式組件，可讓您更快速地執行應用程式。一般而言，這可以將部份處理程序移至功能更強大的伺服器機器，減少從屬站應用程式與該伺服器之間的網路壅塞。

應用程式需求

您的應用程式可以有其它應用程式沒有的專用邏輯。例如，如果您的應用程式依照不適合其它應用程式的特定次序來處理資料登錄錯誤，則您要撰寫自己的程式碼來處理這個狀況。

在某些情況下，您可決定將 DB2 特性用於伺服器上，因爲這些特性可供數個應用程式使用。在其它情況下，您可決定將邏輯保留在應用程式中，因爲它僅供您的應用程式使用。

限制

若要保護資料及定義資料之間的關係，請務必定義一些商業規則。這些規則定義適用於表格中直欄的資料值，或一個以上表格中直欄之間的關係。

DB2 利用資料庫系統，提供一些限制作爲實施那些規則的方法。藉由使用資料庫系統來實施商業規則，您就不必在應用程式中撰寫程式碼來實施它們了。不過，如果只要將一個商業規則引用到一個應用程式，則您應該在應用程式將該規則編碼，而不必使用廣域資料庫限制。

DB2 提供下列各種限制：

1. NOT NULL 限制
2. UNIQUE 限制
3. PRIMARY KEY 限制
4. FOREIGN KEY 限制
5. CHECK 限制

您可使用 SQL 陳述式 CREATE TABLE 和 ALTER TABLE 來定義這些限制。

使用者定義類型 (UDT) 和大型物件 (LOB)

資料庫中每一個資料元素各儲存在表格的一個直欄內，而每一個直欄都定義了一個資料類型。資料類型對您放入直欄中的值類型以及可對其執行的作業加以限制。例如，整數直欄只能含有固定範圍內的數字。 DB2 包括一組具有已定義性質和行爲的內建式資料類型：字串、數字、日期時間值、大型物件、空字元、圖形字串、二進位字串及資料鏈結。

不過，有時候內建式資料類型可能無法滿足您應用程式的需求。 DB2 提供使用者定義類型 (UDT)，供您定義應用程式需要的各種資料類型。

UDT 以內建式資料類型爲基礎。當您定義 UDT 時，同時定義了對 UDT 有效的作業。例如，您可定義以 DECIMAL 資料類型爲基礎的 MONEY 資料類型。不過，以 MONEY 資料類型而言，您只能容許加法和減法運算，但不能容許乘法和除法運算。

「大型物件 (LOB)」可讓您儲存及操作資料庫內大型且複雜的資料物件：這些物件如音效、視訊、影像及大型文件。

UDT 和 LOB 的組合提供您更多的功能。您不再受限於僅使用 DB2 提供的內建資料類型來表現您的商業資料，及擷取該資料的語意。您可使用 UDT 來定義進階應用程式的大型複雜資料結構。

除了擴充內建式資料類型之外，UDT 還提供了幾個其它優點：

支援應用程式的物件導向程式設計

您可將類似物件組合成相關的資料類型。這些類型都有一個名稱、一個內部表示法及一個特定的行爲。藉由使用 UDT，您可以將新類型的名稱及其內部表示的方法通知 DB2。 LOB 是新類型的其中一個可能的內部表示法，也是最適合大型複雜資料結構的表示法。

透過完全符合與囊封而達到的資料完整性

「完全符合」可保證只有定義在特殊類型的函數和作業能引用到該類型。囊封技術可確保 UDT 的行爲受函數及其可用運算子的限制。在 DB2 中，UDT 的行爲可經由使用者定義函數 (UDF) 的形式提供，撰寫這些 UDF 可符合大範圍的使用者需求。

將效能整合到資料庫管理程式

因爲 UDT 是以相同於內建式資料類型的內部方式表示，所以它們使用與內建式資料類型相同的程式碼來執行內建函數、比較運算子、索引及其它函數。例外的是，利用 LOB 的 UDT 無法與比較運算子和索引一起使用。

儲存程序

通常，應用程式可透過網路來存取資料庫。如果要傳回很多資料，則會導致效能不佳。儲存程序可在資料庫伺服器上執行。從屬站應用程式可呼叫儲存程序，然後執行資料庫存取，而不必透過網路傳回不必要的資料。儲存程序只傳回從屬站應用程式需要的結果。

使用儲存程序，您可以獲得許多好處：

減少網路壅塞

將許多 SQL 陳述式一起分組，以減少網路壅塞。典型的應用程式需要每一個 SQL 陳述式在網路上往返兩次。將 SQL 陳述式分組，會變成每一組陳述式在網路上往返兩次，使應用程式的效能提高。

僅存取伺服器上的特性

儲存程序可存取僅在伺服器上執行的指令，例如 LIST DATABASE DIRECTORY 和 LIST NODE DIRECTORY；儲存程序具有增加伺服器機器的記憶體和磁碟空間的優點；也可以存取安裝在伺服器上的其它軟體。

實施商業規則

您可使用儲存程序來定義數個應用程式共用的商業規則。除了使用限制和觸發函式之外，這是定義商業規則的另一個方法。

當某應用程式呼叫儲存程序時，它會根據儲存程序中定義的規則，以一致的方式處理資料。如果您需要變更規則，只要在儲存程序中變更一次即可，不必在每一個應用程式中呼叫儲存程序。

使用者定義函數 (UDF)

SQL 提供的內建式功能可能無法滿足您所有應用程式的需求。爲了可讓您擴充這些功能，DB2 提供了使用者定義函數 (UDF) 的支援。您可以使用 Visual Basic、C/C++ 或 Java 來執行任何 SQL 陳述式內的作業，傳回單一純量值或表格。

UDF 提供您很大的彈性。它們可從資料庫傳回單一純量值作爲選取列示部份，或從非資料庫來源傳回整個表格，例如試算表。

UDF 提供一個方法來標準化您的應用程式。藉由實施一組共用的使用者定義函數，許多應用程式可利用相同方式來處理資料，以確保結果的一致性。

使用者定義函數也支援應用程式的物件導向程式設計。 UDF 提供摘要方式，讓您定義用來執行資料物件作業的方法。並且，UDF 還提供了囊封技術，讓您控制物件基礎資料的存取權，保護它免於被直接竄改及可能的毀損。

OLE DB 表格函數

Microsoft OLE DB 是一組 OLE/COM 介面，提供應用程式對儲存於不同資訊來源的資料相同的存取權。 DB2 Universal Database 簡化了 OLE DB 應用程式的建立，方法是讓您定義一些存取 OLE DB 資料來源的表格函數。您可以在經由 OLE DB 介面來呈現資料的資料來源上，執行包括 GROUP BY、JOIN 和 UNION 的作業。例如，您可定義 OLE DB 表格函數從 Microsoft Access 資料庫或 Microsoft Exchange 通訊錄傳回表格，然後建立一個將 OLE DB 表格函數內資料與 DB2 資料庫內資料完整組合的報告。

利用 OLE DB 表格函數，藉由提供內建式存取權給任何 OLE DB 提供者，來減少您的應用程式開發工作。以 C、Java 和 OLE 自動化表格函數而言，開發人員必須實施表格函數，而在 OLE DB 表格函數的情形下，同屬的內建式 OLE DB 消費者則以 OLE DB 提供者作爲介面來擷取資料。您只需要登記一種語言類型 OLE DB 的表格函數，然後以 OLE DB 提供者和相關列集作爲資料來源參考。您不需要執行任何 UDF 程式設計來利用任何 OLE DB 表格函數。

OLE 自動化 UDF 及儲存程序

OLE (物件連結和內嵌) 自動化是 Microsoft Corporation 的 OLE 2.0 架構組件。透過 OLE 自動化，您的應用程式無論是以哪一種語言撰寫，都可以將它們的內容

和方法呈現在 OLE 自動化物件中。其它應用程式，例如 Lotus Notes 或 Microsoft Exchange，則可利用這些內容與方法的優點，透過 OLE 自動化將這些物件加以整合。

DB2 for Windows 32 位元作業系統，利用 UDF 及儲存程序，提供對 OLE 自動化物件的存取權。欲存取 OLE 自動化物件及呼叫其方法， 您必須將物件的方法登記爲 UDF 或儲存程序。然後，DB2 應用程式就可以藉由呼叫 UDF 或儲存程序來呼叫方法。UDF 可以是純量或表格函數。

例如，您可以開發一個應用程式，在使用某產品 (例如 Microsoft Excel) 建立的試算表中查詢資料。若要這樣做，則您要開發一個 OLE 自動化表格函數，從工作表中擷取資料，然後將它傳回 DB2。然後，DB2 可開始處理資料、執行線上分析處理程序 (OLAP)，再將查詢結果傳回您的應用程式。

觸發函式

觸發函式定義一組在指定表格上執行刪除、插入或更新作業的動作。當執行這種 SQL 作業時，即稱觸發函式啓動。觸發函式可在 SQL 作業之前或之後啓動。您可利用 SQL 陳述式 CREATE TRIGGER 來定義觸發函式。

您可在執行更新或插入之前以下列幾個方法來使用觸發函式：

- 要在資料庫中實際更新或插入一些值之前，先檢查或修改它們。這對於必須將使用者看到的資料轉換成某些內部資料庫格式時很有用。
- 執行以使用者定義函數編碼的其它非資料庫作業。

同樣地，您可在執行更新或插入之後，以下列幾個方法來使用觸發函式：

- 更新其它表格中的資料。這對於維護資料間關係或保存審核追蹤資訊很有用。
- 檢查表格或其它表格中其它資料。當參照整合性限制不適合時或表格核對限制僅限於檢查目前的表格時，這對於確定資料完整性很有用。
- 執行以使用者定義函數編碼的非資料庫作業。這對於在發出警示或更新資料庫外部資訊時很有用。

使用觸發函式您可以獲得許多好處：

加快應用程式開發

觸發函式儲存在資料庫中，可供全部應用程式使用。它可讓您不必爲每一個應用程式編碼相同的函數。

廣域實施商業規則

觸發函式僅須定義一次，即可供使用觸發函式管理的資料之所有應用程式使用。

便於維護

需要在資料庫中做的變更只需執行一次，而不必在使用觸發函式在每一個應用程式中進行。

DB2 Universal Database 工具

您可使用各種不同的工具來開發應用程式。 DB2 Universal Database 提供下列工具來協助您撰寫和測試應用程式中的 SQL 陳述式，以及協助您監督它們的效能：

註: 並非所有工具都適用於每一個平台。

控制中心

一種顯示資料庫物件 (例如資料庫、表格和資料包) 及其相互間關係的圖形介面。「控制中心」可用來執行一些管理作業，像配置系統、管理目錄、備份及回復系統、排定工作行程，以及管理媒體。

「控制中心」包括下列機能：

命令中心

可用來在交談式視窗中輸入 DB2 指令和 SQL 陳述式，以及在結果視窗中查看執行結果。您可上下捲動來查看結果及將輸出存檔。

Script 中心

可用來建立 Script，儲存並於稍後呼叫。這些 Script 可包括 DB2 指令、SQL 陳述式或作業系統指令。您可排定 Script 以無人式執行。您可執行這些工作一次，或將它們設定在重複的排程上執行。重複排程對於備份之類的作業特別有用。

異動日誌

可用來檢視下列類型的資訊：與擱置執行、執行或已完成執行工作相關的所有可用資訊；復原歷程日誌；警示日誌；及訊息日誌。您亦可使用異動日誌來複查無人式執行的工作結果。

警示中心

可用來監控系統，對可能發生的問題提早警告，或自動執行一些動作來解決問題。

工具設定

可用來變更「控制中心」、「警示中心」及「抄寫」的設定。

效能監督程式

「效能監督程式」是「控制中心」的可安裝選項，它是一種圖形介面，能針對 DB2 系統提供綜合效能資料收集，以及檢視、報告、分析及警示功能。「效能監督程式」可用來進行效能調整。

您可選擇監督 snapshot 或事件。「Snapshot 監督程式」可讓您依指定的時間間隔擷取時間點資訊。「事件監督程式」可讓您在事件發生期間 (例如連接) 記錄效能資訊。

Visual Explain

Visual Explain 是「控制中心」的可安裝選項，它是一種圖形介面，可讓您分析及調整 SQL 陳述式，包括檢視 SQL 陳述式最佳化工具選擇的存取計劃。

儲存程序建置器

註: 本節是對「儲存程序建置器」線上說明及*Application Development Guide*之「IBM DB2 儲存程序建置器」一章的補充。

「DB2 儲存程序建置器」是一種 GUI 型工具，支援快速開發 DB2 儲存程序。它提供範圍從工作站到 OS/390 的 DB2 系列單一開發環境。在 Windows 32 位元作業系統上，可以從這些常用的應用程式開發工具中啓動它：Microsoft Visual Studio、Microsoft Visual Basic 及 IBM VisualAge for Java，或從 IBM DB2 Universal Database 程式群組中以個別應用程式的方式來啓動。還可以藉由在指令行執行 `db2spb` 指令，在所有受支援的平台上啓動它。

配置您的環境

若要在您的 AIX、Solaris 或 Windows 系統上用「儲存程序建置器」建置 SQL 程序，需要先配置用於 SQL 程序的 DB2 環境。請遵循第93頁的『第5章 開發 SQL 程序』中的平台設置指示。

若要在您的 AIX、Solaris 或 Windows 系統上用「儲存程序建置器」建置 Java 儲存程序，需要先配置用於 Java 的 DB2 環境。請遵循第60頁的『設定環境』中的平台設置指示。

針對 Solaris 上用「儲存程序建置器」建置的 Java 儲存程序，必須將環境變數 `JAVA_HOME` 設定到 Java 安裝所在的路徑 (即包含 `/bin` 及 `/lib` 目錄的目錄中)。如果您安裝的 Java Development Kit 及 Java Runtime Environment 與 DB2 提供的不同，也需要在 AIX 上這樣做。

「儲存程序建置器」支援使用 Java 1.2 建置 Java 儲存程序 (僅適用於 Windows NT 及 Windows 2000 平台)。利用 Java 1.2 中的雙向語言支援，「儲存程序建置器」支援諸如阿拉伯文及希伯來文之類的雙向語言。

若要用「儲存程序建置器」匯出 Java 儲存程序，請：

1. 在儲存程序資料夾上按一下滑鼠右鍵，再按「匯出 Java 儲存程序」以開啓「匯出 Java 儲存程序」視窗。
2. 選取您想要匯出的儲存程序，並將其移至「選取的儲存程序」直欄。
3. 選取您喜好的選項，並按一下「確定」。

在 OS/390 上建置儲存程序

「DB2 儲存程序建置器」支援在 DB2 for OS/390 版本 6 及更新版本上開發 Java 儲存程序，並支援在 DB2 for OS/390 版本 7 伺服器上建置 SQL 程序。您可以建立新的儲存程序或變更現存的儲存程序。

在 OS/390 上順利完成建置儲存程序 (它將在「工作負荷管理程式 (WLM)」應用程式環境中執行) 之後，「DB2 儲存程序建置器」會自動復新 WLM 位址空間。

當「DB2 儲存程序建置器」連接至 DB2 OS/390 版本 5 及更新版時，如果您使用精靈插入儲存程序時沒有指出 WLM 環境選項，則所產生的程式碼會包含下列文字：NO WLM ENVIRONMENT。此行程式碼會使儲存程序按預期在 SPAS 位址空間中執行。此修正程式可解決「DB2 儲存程序建置器」版本 6 及更新版中存在的問題。

修正程式出現後產生的程式碼如下：

```
CREATE PROCEDURE SYSPROC.Proc2 (  )
    RESULT SETS 1
    LANGUAGE SQL
    MODIFIES SQL DATA
    COLLID TEST
    NO WLM ENVIRONMENT
    ASUTIME NO LIMIT
    RUN OPTIONS 'NOTEST(ALL,*,,VADTCPIP&9.112.14.91:*)'
-----------------------------------------------------------------
-- SQL Stored Procedure
-----------------------------------------------------------------
P1: BEGIN
    -- Declare cursor
    DECLARE cursor1 CURSOR WITH RETURN FOR
        SELECT * FROM SYSIBM.SYSPROCEDURES;

    -- Cursor left open for client application
    OPEN cursor1;

END P1
```

有關「DB2 for OS/390 儲存程序建置器」的詳細資訊，請造訪以下網站：

http://www-4.ibm.com/software/data/db2/os390/spb/exciting

設定建置選項

使用 AIX、Solaris 及 Windows 平台上的「DB2 儲存程序建置器」，可以設定所有 SQL 程序的建置選項。這些建置選項包括下列編譯器及前置編譯器 DB2 登錄變數：

- DB2_SQLROUTINE_PREPOPTS
- DB2_SQLROUTINE_COMPILER_PATH
- DB2_SQLROUTINE_COMPILE_COMMAND
- DB2_SQLROUTINE_KEEP_FILES

雖然可以使用 db2set 指令設定這些登錄變數，但使用「儲存程序建置器」可免去實際存取資料庫伺服器以發出指令，或停止後再重新啓動伺服器以使變更生效的麻煩。

若要開啓「SQL 儲存程序建置選項」視窗，請在您專案檢視的資料庫連線上按一下滑鼠右鍵，再按「SQL 儲存程序建置選項」。若需設定這些選項的詳細資訊，請參閱「DB2 儲存程序」線上說明。

MQSeries 與 OLE DB 的表格 UDF

「DB2 儲存程序建置器」提供精靈，協助您建立 MQSeries 及 OLE DB 的表格 UDF。使用「建立 OLE DB 表格 UDF」精靈，可以存取 OLE DB 資料提供者。精靈會建立 OLE 表格 UDF 及選用的表格檢視。可使用「建立 MQSeries 表格 UDF」精靈，建立帶選項表格檢視的表格 UDF，以存取 MQSeries 訊息並以列表格式剖析資料。

除錯儲存程序

將 AIX、Solaris 及 Windows 上的除錯 SQL 程序直接整合到「DB2 儲存程序建置器」上。在除錯非隔離 (受信任) 的「SQL 程序」時，KEEPDARI 資料庫管理程式配置選項可設成「是」或「否」；而在除錯隔離 (未受信任) 的「SQL 程序」時，必須設成「是」(預設值)。若需使用整合除錯器的附加資訊，請參閱「儲存程序建置器」說明。

若要在 AIX、Solaris 及 Windows 平台上將遠端除錯功能用於 Java 及 C 儲存程序，需要安裝 IBM Distributed Debugger。IBM Distributed Debugger 內含在 Visual Age for Java Professional Edition CD 中。除錯器從屬站僅在 Windows 平台上執行。AIX、Solaris 及 Windows 爲受支援的伺服器平台。使用「儲存程序建置器」

內建的 SQL 除錯功能，可以除錯 AIX、Solaris 及 Windows 的本端及遠端「SQL 儲存程序」。若要在 OS/390 平台上除錯 SQL 程序，必須具備 IBM C/C++ Productivity Tools for OS/390 R1 產品。有關 IBM C/C++ Productivity Tools for OS/390 R1 的詳細資訊，請造訪以下網站：

```
http://www.ibm.com/software/ad/c390/pt/
```

已知問題、限制及可行方案

- 「SQL 程序」目前在 Windows 98 上並未受支援。
- 對於 Java 儲存程序而言，JAR ID、類別名稱及方法名稱不能包含非 ASCII 字元。
- 在 AS/400 上，必須將以下 V4R4 PTF 套用到 OS/400 V4R4：

  ```
  - SF59674
  - SF59878
  ```

- 當從資料庫復置儲存程序時，字元次類型爲 FOR MIXED DATA 或 FOR SBCS DATA 的儲存程序參數不在編輯器窗格的原始碼中顯示。
- 目前，從資料庫擷取 Java 原始碼時會發生問題。擷取期間，程式碼中的說明無法正確顯示。這將影響正在非 ASCII 字碼頁中工作，以及其從屬站及伺服器在不同字碼頁的「DB2 儲存程序建置器」使用者。
- 將 Java Development Kit 1.1.8 或 Java Runtime Environment 1.1.8 與繁體中文語言環境搭配使用時，會發生問題。「儲存程序建置器」程式的圖形 (包括功能表、編輯器文字、訊息等) 將無法正常顯示。解決方案是變更檔案 `font.properties.zh_TW` (其位於以下目錄之一或兩者中)：

  ```
  sqllib/java/jdk/lib
  sqllib/java/jre/lib
  ```

  將：

  ```
  monospaced.0=\u7d30\u660e\u9ad4,CHINESEBIG5_CHARSET,NEED_CONVERTED
  ```

  變更爲：

  ```
  monospaced.0=Courier New,ANSI_CHARSET
  ```

第1章 簡介

本書提供設定環境來開發 DB2 應用程式時所需要的資訊，並提供逐步指示，協助您在此環境中編譯、鏈結和執行這些應用程式。本書說明如何在下列平台中，使用 DB2 Application Development (DB2 AD) Client for DB2 Universal Database 版本 7 來建置應用程式：

- AIX
- HP-UX
- Linux
- OS/2
- PTX
- Silicon Graphics IRIX
- Solaris 作業環境
- Windows 32 位元作業系統

註:

1. DB2 for NUMA-Q 支援 PTX 作業系統。
2. Windows 32 位元作業系統包括 Windows NT、Windows 95、Windows 98 及 Windows 2000。每當本書提及 Windows 32 位元作業系統時，均暗喻所有這些作業系統，但系統網路架構 (SNA) 支援及 REXX 支援的情況除外。它們僅在 Windows NT 及 Windows 2000 中受支援。

若要開發應用程式，您可使用下列程式設計介面：

DB2 應用程式設計介面 (DB2 API)
提供管理功能以管理 DB2 資料庫。

DB2 Call Level Interface (DB2 CLI)
是以 X/Open CLI 規格爲基礎的可呼叫 SQL 介面，與 Microsoft 公司的 Open Database Connectivity (ODBC) 介面相容。

內含的 SQL
在程式中直接使用已編碼的 SQL 陳述式，必須前置編譯該陳述式才能轉換成執行函數呼叫。

內含的 SQL for Java (SQLJ)
在產生的設定檔中使用 SQL 陳述式，這些陳述式經過前置編譯而且自行設定爲執行時函數呼叫，它們會輪流提供介面給資料庫管理程式。

Java Database Connectivity (JDBC)
是動態的 SQL API for Java。JDBC API 包括在已支援平台上提供使用的 Java Development Kits 中。

關於不同程式設計介面的其它資訊，請參閱：

- *Application Development Guide* 討論如何編碼和設計一些應用程式，這些應用程式使用內含的 SQL、內含的 SQL for Java 和 Java Database Connectivity (JDBC) 來存取 DB2 系列伺服器。亦說明使用者定義函數 (UDF)。
- *CLI Guide and Reference* 說明如何編碼和設計使用 DB2 Call Level Interface 和 ODBC 的應用程式。
- *Administrative API Reference* 討論如何編碼和設計使用「DB2 應用程式設計介面」的應用程式。

您會發現下列書籍對於提供進一步相關資訊很有用，例如詳細的產品安裝和設定：

- *DB2 for OS/2 快速入門*，說明如何在 OS/2 伺服器及從屬站工作站上，安裝資料庫管理程式及 DB2 Application Development Client。
- *DB2 for UNIX 快速入門*，說明如何在 UNIX 伺服器及從屬站工作站上，安裝資料庫管理程式及 DB2 Application Development Client。
- *DB2 for Windows 快速入門*，說明如何在 Windows 32 位元作業系統的伺服器及從屬站工作站上，安裝資料庫管理程式及 DB2 Application Development Client。
- *Command Reference*，說明如何使用 DB2「命令行處理器 (CLP)」和所有 DB2 指令。
- *Troubleshooting Guide*，協助您解決與 DB2 從屬站和伺服器相關的應用程式開發問題，以及協助您解決與資料庫管理和連接方面的作業相關的問題。

關於 DB2 文件檔案庫的完整列示，請參閱第377頁的『附錄D. 使用 DB2 檔案庫』。

註: 本書的範例以「現狀」提供，不具任何保證。使用者 (而非 IBM) 要承擔品質、效能及任何錯誤修復的全部風險。

適用對象

如果您想在其中一個目前支援 DB2 Universal Database 版本 7 的平台上開發程式，建議您使用本書。本書說明您的程式如何使用 DB2 API 管理 DB2 資料庫，以及如何使用 DB2 CLI、內含的 SQL、SQLJ 和 JDBC 來存取 DB2 資料庫。

使用本書之前，您必須先熟悉在此平台上要使用的一種或多種支援的程式設計語言。這些語言列示於第6頁的『平台支援的軟體』中。

如何使用本書

本書的設計方式可讓您輕易存取開發應用程式所需的資訊。各章的分組如下：

- 第 1 到第 3 章：各章含有所有平台的一般性及介紹性資訊。
- 第 4 到第 5 章：各章含有所有平台的特定程式設計資訊。
- 第 6 到第 13 章：各章含有某一平台的特定程式設計資訊。

所有的應用程式開發者均應閱讀前三章，然後依據個人使用的作業系統及程式設計語言，讀取「建置應用程式」一章，其中含有開發者需要的特定程式設計資訊。

這些附錄提供關於不同主題的重要附加資訊。

強調標示慣例

本書使用下列慣例：

斜體字 指出下列其中一個：

- 新術語的介紹
- 使用者所提供的名稱及值
- 另一個資訊來源的參照
- 一般的重點

大寫字體

指出下列其中一個：

- 資料庫管理程式資料類型
- 欄位名稱
- 關鍵字
- SQL 陳述式

範例文字

指出下列其中一個：

- 編寫的例子及程式碼片段
- 指令
- 輸出的例子，類似於系統所顯示的內容
- 特別資料值的範例
- 系統訊息的例子
- 檔案及目錄名稱
- 指示您輸入的資訊

粗體 強調。

關於 DB2 Application Development Client

註: Application Development Client 在舊版的 DB2 中稱爲 DB2 Software Development Kit (DB2 SDK) Client。

DB2 Application Development (DB2 AD) Client 提供開發應用程式所需的工具及環境，存取施行 DRDA 的 DB2 伺服器及應用程式伺服器。

您可以使用安裝的 DB2 AD Client 來建置及執行 DB2 應用程式。也可以在這些 DB2 從屬站上執行 DB2 應用程式：

- DB2 Run-Time Client
- DB2 Administration Client

請參閱第33頁的『第2章 設定』，取得關於如何設置您的程式環境的資訊。

本書說明要使用於各平台的 DB2 AD Client，包括下列各項：

- **C/C++、COBOL 及 Fortran 的前置編譯器** (假設該平台支援這些語言；詳細資訊，請參閱第6頁的『平台支援的軟體』)。
- **內含的 SQL 應用程式支援**，包括程式設計檔案庫、併入檔及程式碼範例。
- **DB2 Call Level Interface (DB2 CLI) 應用程式支援**，包括程式設計檔案庫、併入檔及程式碼範例，以開發易於移轉到 ODBC 及使用 ODBC SDK 編譯的應用程式。您可以從 Microsoft for Windows 32 位元作業系統，及許多其它支援平台的不同廠商處取得 ODBC SDK。就 Windows 32 位元作業系統而言，DB2 從屬站包含一個 ODBC 驅動程式，該驅動程式支援使用 Microsoft ODBC SDK 開發的應用程式。就所有其它平台而言，DB2 從屬站包含一個選擇性安裝的 ODBC 驅動程式，該驅動程式支援可以使用該平台的 ODBC SDK 開發的應用程式。僅有 DB2 Clients for Windows 32 位元作業系統含有 ODBC 驅動程式管理程式。

- **DB2 Java Enablement**，包括 DB2 Java Database Connectivity (DB2 JDBC) 支援（可用於開發 Java 應用程式及 Applet），及 DB2 內含的 SQL for Java (DB2 SQLJ) 支援（可用於開發 Java 內含的 SQL 應用程式及 Applet）。
- **Java Development Kit (JDK) 1.1.8 及 Java Runtime Environment (JRE) 1.1.8**，由 IBM 所推出，與 DB2 for AIX 及 DB2 for Windows 32 位元作業系統一起安裝，並隨附在 DB2 for OS/2 中。
- **ActiveX Data Objects (ADO) 及「物件連結和內嵌」 (OLE) 自動化 UDF 及「儲存程序」**適用於 Windows 32 位元作業系統，含有在 Microsoft Visual Basic 及 Microsoft Visual C++ 中執行的程式碼範例。此外，還有在 Microsoft Visual Basic 中已施行且含 Remote Data Objects (RDO) 的程式碼範例。
- **物件連結和內嵌資料庫 (OLE DB) 表格函數** ，適用於 Windows 32 位元作業系統。
- **DB2 儲存程序建置器 (SPB)**，適用於 AIX、Solaris 及 Windows 32 位元作業系統。這是一種 GUI 型的工具，可支援快速開發DB2 儲存程序。它提供範圍從工作站到 OS/390 的 DB2 系列單一開發環境。在 Windows 上，可以從下列這些受歡迎的應用程式開發工具來啓動它： Microsoft Visual Studio、Microsoft Visual Basic 及 IBM VisualAge for Java，或從 IBM DB2 Universal Database 程式群組以個別應用程式方式來啓動。在 AIX 及 Solaris 上，可以使用 `db2spb` 指令來啓動它。
- **交談式 SQL** 經由「命令中心」或「命令行處理器 (CLP)」，產生 SQL 陳述式的原型，或對資料庫執行特別查詢。
- **一組記錄的 API**，可讓其它應用程式開發工具直接在它們的產品內執行 DB2 的前置編譯器支援。例如，在 AIX 及 OS/2 中，IBM COBOL 會使用此介面。您可以從匿名 FTP 站台 (ftp://ftp.software.ibm.com) 取得「前置編譯器服務程式 API」組的相關資訊。 PostScript 檔 prepapi.psbin 是在 /ps/products/db2/info 目錄中。此檔案是二進位格式。如果您無法存取此電子論壇而想要取得此份文件的拷貝，則可以依照「服務資訊夾頁」中的說明，自「IBM 服務中心」訂購。
- **SQL92 及 MVS 一致旗號控制**，它會識別應用程式中不符合 ISO/ANSI SQL92 Entry Level 標準的內含 SQL 陳述式，或識別 DB2 for OS/390 不支援的內含 SQL 陳述式。如果您將工作站上所開發的應用程式移轉到另一個平台，「旗號控制」會經由顯示語法不相容，來節省您的時間。請參閱 *Command Reference*，取得關於 PRECOMPILE PROGRAM 指令中 SQLFLAG 選項的資訊。

支援的伺服器

您可以使用 DB2 AD 從屬站來開發將在特定平台上執行的應用程式。不過，應用程式可以在下列平台伺服器存取遠端資料庫：

- DB2 for AIX
- DB2 for HP-UX
- DB2 for Linux
- DB2 for OS/2
- DB2 for NUMA-Q
- DB2 for SCO UnixWare 7
- DB2 for Solaris
- DB2 for Windows NT
- 遵守 DRDA 的應用系統伺服器，如：
  - DB2 for OS/390
  - DB2 for AS/400
  - DB2 for VSE & VM (先前稱爲 SQL/DS for VM 及 VSE)
  - 遵守 DRDA 的應用系統伺服器，它們來自資料庫廠商而非 IBM。

註:

1. DB2 for NUMA-Q 支援 PTX 作業系統。
2. DB2 for SCO UnixWare 7 僅適用於 DB2 版本 5.2。

平台支援的軟體

本段落將列出 DB2 對本書所描述的平台支援的編譯器及相關軟體。編譯器資訊假定您將使用該平台的 DB2 前置編譯器，而非使用可能已內建在所列出的編譯器之一的前置編譯器支援。關於您的作業系統支援的通信產品的資訊，請參閱您的作業系統的*快速入門*這本書。

若需最新的 DB2 編譯器資訊及相關的軟體更新，請造訪下列 DB2 應用程式開發網頁：

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

註:

1. **DB2 版本注意事項**也許會有支援平台的更新編譯器及作業系統資訊。您可以在您的產品 CD-ROM 中，從下列路徑取得一般文字及 HTML 格式的「版本注意事項」，其中 `<language_directory>` 是您使用的語言目錄，而 `index.htm` 是主要的 HTML 檔：

   一般文字檔：

   doc/<language_directory>/release.txt (UNIX)

Doc\<language_directory>\release.txt (OS/2 及 Windows)

HTML 檔：

doc/<language_directory>/db2ir/index.htm (UNIX)

Doc\<language_directory>\db2ir\index.htm (OS/2 及 Windows)

2. **Fortran 和 REXX**。DB2 不會增強 Fortran 和 REXX 的特性超過 DB2 Universal Database 版本 5.2 中的這些語言的支援層次。
3. **Fortran** 在 DB2 版本 7 中不提供 Fortran 範例程式。
4. **HP-UX**。如果您是從 HP-UX 版本 10 或更早的版本將 DB2 移轉到 HP-UX 版本 11，您的 DB2 程式必須以 HP-UX 版本 11 上的 DB2 重新執行前置編譯 (如果它們併入內含的 SQL)，且必須重新編譯。包括所有 DB2 應用程式、儲存程序、使用者定義函數及使用者跳出程式都需進行此項作業。並且，在 HP-UX 版本 11 上編譯的 DB2 程式不能在 HP-UX 版本 10 或更早版本上執行。在 HP-UX 版本 10 編譯及執行的 DB2 程式可從遠端連接到 HP-UX 版本 11 伺服器。
5. **Micro Focus COBOL**。透過 DB2 版本 2.1.1 或更早版本來進行前置編譯，並以 Micro Focus COBOL 進行編譯的任何現存的應用程式應該以現行版本的 DB2 重新進行前置編譯，然後以 Micro Focus COBOL 重新進行編譯。如果這些以 IBM 前置編譯器的較早版本開發的應用程式未重新進行前置編譯，則在發生異常終止時，資料庫可能會損毀。
6. **Perl**。列印時，Perl Database Interface (Perl DBI) 的 DB2 UDB 驅動程式 (DBD::DB2) 版次 0.75 適用於 AIX、HP-UX、Linux、Solaris 及 Windows NT。您可以從下列網址下載最新的驅動程式：

   `http://www.ibm.com/software/data/db2/perl`

7. **REXX**。IBM Object REXX for Windows NT/95 不再隨附於 DB2 中。若需取得 Object REXX 的相關資訊，請造訪：

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

8. **PHP**。目前，可以使用 PHP 從 Web 應用程式存取 DB2。PHP 是內含在 HTML 中的跨平台伺服器端 Script 語言。它支援使用統一 ODBC 存取方法對 DB2 的存取，在該方法中，用戶級的 PHP 使用 ODBC 呼叫與 DB2 通信。與標準的 ODBC 不同，使用「統一 ODBC」方法，通信會直接到達 DB2 CLI 層，而不透過 ODBC 層。如需 PHP 與 DB2 搭配使用的其它資訊，請搜尋 DB2 支援網站：

   `www.ibm.com/software/data/db2/udb/winos2unix/support`

AIX

DB2 for AIX 支援下列作業系統：

AIX/6000

版本 4.2.1 (僅適用於 32 位元)

版本 4.3.3 及更新的版本

PowerPC 上的 AIX 5

註: 除非特別說明，否則這些版本均適用於 32 位元及 64 位元。

DB2 for AIX 支援下列程式設計語言及編譯器：

C IBM C for AIX 版本 3.6.6 (僅適用於 32 位元)

IBM C for AIX 版本 3.6.6.3

IBM C for AIX 版本 4.4

IBM C for AIX 版本 5.0

註: 除非特別說明，否則這些版本均適用於 32 位元及 64 位元。

C++ IBM C Set++ for AIX 版本 3.6.6 (僅適用於 32 位元)

IBM C Set++ for AIX 版本 3.6.6.3

IBM VisualAge C++ 版本 4.0

IBM VisualAge C++ 版本 5.0

註: 除非特別說明，否則這些版本均適用於 32 位元及 64 位元。

COBOL

IBM COBOL Set for AIX 版本 1.1

Micro Focus COBOL 版本 4.0.20 (PRN 12.03 或更新版本) for AIX 4.2.1

Micro Focus COBOL 版本 4.1.10 (PRN 13.04 或更新版本) for AIX 4.2.1

Micro Focus Server Express 1.0.00 (PRN 13.08 或更新版本) for AIX 4.3.3 及更新版本

Fortran

IBM XL Fortran for AIX 版本 4.1 (僅適用於 32 位元) 及 5.1.0 (32 位元及 64 位元)

Java 對於 AIX 4.2.1 與 AIX 4.3.3 及更新版本：IBM 推出的 Java Development Kit (JDK) 版本 1.1.8 及 Java Runtime Environment (JRE) 版本 1.1.8 for AIX (與 DB2 一同安裝)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。

對於 AIX 4.3.3 及更新版本：IBM 推出的 Java Development Kit (JDK) 版本 1.2.2 及 Java Runtime Environment (JRE) 版本 1.2.2 for AIX。需要 JDK 的開發日期爲 2000 年 4 月 17 日或之後。

Perl Perl 5.004_04、DBI 0.93 (請參閱上面的備註，以取得 DB2 UDB 驅動程式 DBD::DB2 的相關資訊)

REXX IBM AIX REXX/6000 AISPO 產品編號：5764-057

IBM Object REXX for AIX 版本 1.1

REXXSAA 4.00

註: REXX 支援僅適用於 32 位元

HP-UX

DB2 for HP-UX 支援下列作業系統：

HP-UX
版本 11 及 11i

DB2 for HP-UX 支援下列程式設計語言及編譯器：

C HP C Compiler 版本 A.11.00.03

C++ HP aC++ 版本 A.03.25

COBOL
Micro Focus COBOL 版本 4.1

Fortran
HP Fortran/9000 版本 10.0

HP-UX F77 B.11.00.01

Java Hewlett-Packard 推出的 HP-UX Developer's Kit for Java 版次 1.1.8

Hewlett-Packard 推出的 HP-UX Developer's Kit for Java 版次 1.2.2

Perl Perl 5.004_04、DBI 0.93 (請參閱上面的備註，以取得 DB2 UDB 驅動程式 DBD::DB2 的相關資訊)

Linux

DB2 for Linux for Intel x86 (32 位元架構) 支援下列作業系統環境：

Linux 核心程式版本 2.2.12 或更新版本、glibc 版本 2.1.2 或更新版本、libstdc++ 版本 2.9.0、rpm (需要安裝) 及 pdksh 套裝軟體

DB2 for Linux on S/390 支援下列作業系統環境：

Linux 核心程式版本 2.2.16 或更新版本、glibc 2.1.3 或更新版本、libstdc++ 版本 6.1、rpm (需要安裝)、pdksh 套裝軟體，以及下列一項：SuSE Linux 版本 7.0 (S/390 版) 或 Turbolinux Server 6 (zSeries 及 S/390 版)

DB2 for Linux 支援下列程式設計語言及編譯器：

C GNU/Linux gcc 版本 egcs-2.91.66 (egcs-1.1.2 版次)

C++ GNU/Linux g++ 版本 egcs-2.91.66 (egcs-1.1.2 版次)

Java IBM Developer Kit 及 Runtime Environment for Linux 版本 1.1.8 (需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後)。

IBM Developer Kit 及 Runtime Environment for Linux 版本 1.2.2

IBM Developer Kit 及 Runtime Environment for Linux 版本 1.3

Perl Perl 5.004_04、DBI 0.93 (請參閱上面的備註，以取得 DB2 UDB 驅動程式 DBD::DB2 的相關資訊)

OS/2

DB2 for OS/2 支援下列作業系統：

OS/2 WARP 3.0、WARP 4.0 及 WARP 4.5

DB2 for OS/2 支援下列程式設計語言：

C/C++

IBM VisualAge C++ for OS/2 版本 3.6.5 及 4.0

註: 請自下列網址下載適用於這些 VisualAge 編譯器版本的最新 FixPak：

```
http://www.ibm.com/software/ad/vacpp/service/csd.html
```

這些 VisualAge C++ 編譯器將來服務支援的限制，請參閱下列網站的新聞部份：

```
http://www.ibm.com/software/ad/vacpp
```

COBOL

IBM VisualAge COBOL for OS/2 版本 2.0

Micro Focus COBOL 版本 4.0.20

FORTRAN

WATCOM FORTRAN 77 32 版本 10.5

Java IBM 推出的 Java Development Kit (JDK) 版本 1.1.8 及 Java Runtime

Environment (JRE) 版本 1.1.8 for OS/2 (與 DB2 一同提供)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。

REXX IBM Procedures Language 2/REXX (屬於 OS/2 的一部份)

PTX

DB2 for NUMA-Q 支援下列作業系統：

PTX 版本 4.5.1 及更新版本

DB2 for NUMA-Q 支援下列程式設計語言及編譯器：

C ptx/C 版本 4.5

C++ ptx/C++ 版本 5.2

Java ptx/JSE 版本 3.0

Silicon Graphics IRIX

DB2 for Silicon Graphics IRIX 支援下列作業系統：

Silicon Graphics IRIX
版本 6.2 及更新的版本

DB2 for Silicon Graphics IRIX 支援下列程式設計語言及編譯器：

C MIPSpro C Compiler 7.2

C++ MIPSpro C++ 7.2

Fortran
MIPSpro Fortran-77 7.2

Java Silicon Graphics 公司推出的 Java2 Software Development Kit 版本 1.2.2 (JDK 1.2.2)

註: Silicon Graphics IRIX 支援僅適用於 32 位元

Solaris

DB2 for Solaris 支援下列作業系統：

Solaris
版本 2.6 (僅適用於 32 位元)

Solaris 7 及 Solaris 8 (32 位元及 64 位元)

DB2 for Solaris 支援下列程式設計語言及編譯器：

C Forte/WorkShop C 版本 4.2 (僅適用於 32 位元)、5.0、6 及 6.1 (32 位元及 64 位元)

註: 這些編譯器版本通常稱爲 ″SPARCompiler″。

C++ Forte/WorkShop C++ 版本 4.2 (僅適用於 32 位元)、5.0、6 及 6.1 (32 位元及 64 位元)

註: 這些編譯器版本通常稱爲 ″SPARCompiler″。

COBOL

Micro Focus COBOL Server Express 版本 1.0.00

Fortran

SPARCompiler Fortran 版本 4.2 及 5.0

Java Sun Microsystems 推出的 Java Development Kit (JDK) 版本 1.1.8 及 1.2.2 for Solaris

Perl Perl 5.004_04、DBI 0.93 (請參閱上面的備註，以取得 DB2 UDB 驅動程式 DBD::DB2 的相關資訊)

Windows 32 位元作業系統

DB2 for Windows 32 位元作業系統支援下列各項：

Microsoft Windows NT

版本 4.0 與服務程式封包 4 或更新版本

Microsoft Windows Millennium Edition

Microsoft Windows 2000

Microsoft Windows 98

Microsoft Windows 95

版本 4.00.950 或更新版本

DB2 for Windows 32 位元作業系統支援下列程式設計語言：

Basic Microsoft Visual Basic 版本 4.2 及版本 5.0 (沒有提供此語言的 DB2 前置編譯器)

C/C++

Microsoft Visual C++ 版本 5.0 和 6.0

IBM VisualAge C++ for Windows 版本 3.6.5 及 4.0

註: 請自下列網址下載適用於這些 VisualAge 編譯器版本的最新 FixPak：

http://www.ibm.com/software/ad/vacpp/service/csd.html

這些 VisualAge C++ 編譯器將來服務支援的限制，請參閱下列網站的新聞部份：

http://www.ibm.com/software/ad/vacpp

COBOL

Micro Focus COBOL 版本 4.0.20

Micro Focus COBOL Net Express 版本 3.0

IBM VisualAge COBOL 版本 2.0

REXX IBM Object REXX for Windows NT/95 版本 1.1 (請參閱前面的備註)

Java IBM 推出的 Java Development Kit (JDK) 1.1.8 及 Java Runtime Environment (JRE) 1.1.8 for Win32 (可選擇與 DB2 一起安裝)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。

Sun Microsystems 推出的 Java Development Kit (JDK) 1.2.2 for Win32

IBM 推出的 Java Development Kit (JDK) 1.2.2 for Win32。需要 JDK 的開發日期爲 2000 年 4 月 17 日或之後。

Microsoft Software Developer's Kit for Java 版本 3.1

Perl Perl 5.004_04、DBI 0.93 (提供 Windows NT 版，請參閱上面的備註，以取得 DB2 UDB 驅動程式 DBD::DB2 的相關資訊)

範例程式

註:

1. 本節針對 DB2 支援的所有平台，說明程式設計語言的範例程式。並非所有範例程式均已移植到所有平台或支援的程式設計語言。
2. Db2 範例程式僅以「現狀」提供，不具任何保證。使用者 (而非 IBM) 要承擔品質、效能及任何錯誤修復的全部風險。

範例程式隨附於 DB2 Application Development (DB2 AD) Client。您可以使用這些範例程式作爲模板，建立您自己的應用程式。

每一種支援語言的範例程式副檔名不同，而在每一種語言中，內含的 SQL 與非內含的 SQL 程式之範例程式副檔名也不同。某一語言中，程式群組的副檔名也會不同。這些不同的範例檔的副檔名均在下列表格中加以分類：

按語言分類的範例副檔名

第15頁的表1.

按程式群組分類的範例副檔名
第15頁的表2.

下表收集按類型分類的範例程式：

DB2 API 範例程式，不具有內含的 SQL
第17頁的表3.

DB2 API 內含的 SQL 範例程式
第20頁的表4.

內含的 SQL 範例程式，不具有 DB2 API
第22頁的表5.

使用者定義的函數範例程式
第23頁的表6

DB2 CLI 範例程式
第24頁的表7.

Java JDBC 範例程式
第25頁的表8.

Java SQLJ 範例程式
第26頁的表9.

SQL 程序範例程式
第27頁的表10.

ActiveX 資料物件、遠端資料物件及 Microsoft 異動伺服器範例程式
第28頁的表11.

物件連結和內嵌 (OLE) 自動化範例程式
第29頁的表12.

物件連結和內嵌資料庫 (OLE DB) 表格函數
第30頁的表13.

命令行處理器 (CLP) 範例程式
第31頁的表14.

日誌管理使用者跳出程式
第31頁的表15.

註:

1. 第20頁的表4包含的程式具有 DB2 API 及內含的 SQL 陳述式。關於所有的 DB2 API 範例程式，請參閱第17頁的表3及第20頁的表4。 關於所有內含的 SQL 範例程式 (除了 Java SQLJ 以外)，請參閱第20頁的表4及第22頁的表5。

2. UDF 的第23頁的表6範例程式不含 DB2 CLI UDF 程式。關於這些程式，請參閱第24頁的表7。

表 1. 按語言分類的範例副檔名

語言	目錄	內含的 SQL 程式	非內含的 SQL 程式
C	samples/c samples/cli (CLI 程式)	.sqc	.c
C++	samples/cpp	.sqC (UNIX) .sqx (Windows & OS/2)	.C (UNIX) .cxx (Windows & OS/2)
COBOL	samples/cobol samples/cobol_mf	.sqb	.cbl
JAVA	samples/java	.sqlj	.java
REXX	samples/rexx	.cmd	.cmd

表 2. 按程式群組分類的範例副檔名

範例群組	目錄	副檔名
ADO, RDO, MTS	samples\ADO\VB (Visual Basic) samples\ADO\VC (Visual C++) samples\RDO samples\MTS	.bas .frm .vbp (Visual Basic) .cpp .dsp .dsw (Visual C++)
CLP	samples/clp	.db2
OLE	samples\ole\msvb (Visual Basic) samples\ole\msvc (Visual C++)	.bas .vbp (Visual Basic) .cpp (Visual C++)
OLE DB	samples\oledb	.db2
SQL 程序	samples/sqlproc	.db2 .c .sqc (從屬站應用程式)
使用者跳出	samples/c	.cad (OS/2) .cadsm (UNIX & Windows) .cdisk (UNIX & Windows) .ctape (UNIX) .cxbsa (UNIX & Windows)

註:

目錄定界符號

在 UNIX 上，爲 /。 在 OS/2 及 Windows 平台爲 \。在這些表格中，除非僅 Windows 及/或 OS/2 上提供此目錄，否則即使用這些 UNIX 定界符號。

副檔名 在只存在一個副檔名的表格中，提供給範例使用。

內含的 SQL 程式

需要前置編譯，但 REXX 內含的 SQL 程式除外，因爲該程式在執行時會解譯內含的 SQL 陳述式。

IBM COBOL 範例

只有在 AIX、OS/2 及 Windows 32 位元作業系統的 `cobol` 次目錄中提供。

Micro Focus Cobol 範例

只有在 AIX、HP-UX、OS/2、Solaris 作業環境及 Windows 32 位元作業系統的 `cobol_mf` 次目錄中提供。

Java 範例

包括 Java Database Connectivity (JDBC) applet、應用程式及儲存程序、內含的 SQL for Java (SQLJ) applet、應用程式和儲存程序，以及 Java UDF。所有支援的 DB2 平台上皆提供 Java 範例。

REXX 範例

只有在 AIX、OS/2 及 Windows NT 作業系統才提供。

CLP 範例

執行 SQL 陳述式的命令行處理器 script。

OLE 範例

適用於 Microsoft Visual Basic 及 Microsoft Visual C++ 的「物件連結和內嵌」 (OLE)，只在 Windows 32 位元作業系統上提供。

ADO、RDO 及 MTS 範例

包括 Microsoft Visual Basic 和 Microsoft Visual C++ 中的 ActiveX 資料物件範例，以及 Microsoft Visual Basic 中的「遠端資料物件」和「Microsoft 異動伺服器」範例，只在 Windows 32 位元作業系統上才提供。

使用者跳出程式範例

用來保存及擷取資料庫日誌檔的「日誌管理使用者跳出程式」。 這些檔案必須以 `.c` 副檔名加以更名，並編譯爲 C 語言程式。

您可以在已安裝 DB2 的目錄的 `samples` 次目錄中，找到範例程式。 每一種支援的語言均有一個次目錄。 下列範例告訴您如何在每一個支援的平台上，找出以 C 或 C++ 撰寫而成的範例。

- **在 UNIX 平台上**。

  您可以在您的資料庫案例目錄下的 `sqllib/samples/c` 中，找到內含的 SQL 及 DB2 API 程式的 C 原始程式； DB2 CLI 程式的 C 原始程式是在

sqllib/samples/cli 中。關於範例表格中範例程式的其它資訊，請參閱 DB2 案例下適當 samples 次目錄中的 README 檔案。 README 檔將含有任何未列示在本書中的附加範例。

- **On OS/2 及 Windows 32 位元作業系統**。

  您可以在 DB2 安裝目錄下的 %DB2PATH%\samples\c 中，找到內含的 SQL 及 DB2 API 程式的 C 原始程式； DB2 CLI 程式的 C 原始程式是在 %DB2PATH%\samples\cli 中。 變數 %DB2PATH% 決定 DB2 的安裝所在。根據安裝 DB2 的磁碟機而定，%DB2PATH% 將指向 *drive*:\sqllib。關於範例表格中程式的附加資訊，請參閱適當 %DB2PATH%\samples 次目錄中的 README 檔案。README 檔將含有任何未列示在本書中的附加範例。

範例程式目錄在大多數平台上是唯讀的。在您可以變更或開發範例程式之前，請將它們複製到您的工作目錄中。

DB2 API 不具有內含的 SQL 範例

表 3. DB2 API 範例程式，不具有內含的 SQL

範例程式	內含 API
backrest	• sqlbftcq - 提取表格空間配置區查詢 • sqlbstsc - 設定表格空間配置區 • sqlfudb - 更新資料庫架構 • sqlubkp - 備份資料庫 • sqluroll - Rollforward 資料庫 • sqlurst - 回復資料庫
checkerr	• sqlaintp - 取得錯誤訊息 • sqlogstt - 取得 SQLSTATE 訊息
cli_info	• sqleqryi - 查詢從屬站資訊 • sqleseti - 設定從屬站資訊
client	• sqleqryc - 查詢從屬站 • sqlesetc - 設定從屬站
d_dbconf	• sqleatin - 連接 • sqledtin - 分離 • sqlfddb - 取得資料庫架構預設值

表 3. DB2 API 範例程式，不具有內含的 SQL (繼續)

範例程式	內含 API
d_dbmcon	• sqleatin - 連接 • sqledtin - 分離 • sqlfdsys - 取得資料庫管理程式架構預設值
db_udcs	• sqleatin - 連接 • sqlecrea - 建立資料庫 • sqledrpd - 捨棄資料庫
db2mon	• sqleatin - 連接 • sqlmon - 取得/更新監督程式開關 • sqlmonss - 取得 Snapshot • sqlmonsz - 估計 sqlmonss() 輸出緩衝區所需的大小 • sqlmrset - 重設監督程式
dbcat	• sqlecadb - 編目資料庫 • sqledcls - 關閉資料庫目錄掃描 • sqledgne - 取得下一個資料庫目錄登錄 • sqledosd - 開啓資料庫目錄掃描 • sqleuncd - 取消編目資料庫
dbcmt	• sqledcgd - 變更資料庫註解 • sqledcls - 關閉資料庫目錄掃描 • sqledgne - 取得下一個資料庫目錄登錄 • sqledosd - 開啓資料庫目錄掃描 • sqleisig - 安裝信號處理程式
dbconf	• sqleatin - 連接 • sqlecrea - 建立資料庫 • sqledrpd - 捨棄資料庫 • sqlfrdb - 重設資料庫架構 • sqlfudb - 更新資料庫架構 • sqlfxdb - 取得資料庫架構
dbinst	• sqleatcp - 連接及變更通行碼 • sqleatin - 連接 • sqledtin - 分離 • sqlegins - 取得案例

表 3. DB2 API 範例程式，不具有內含的 SQL (繼續)

範例程式	內含 API
dbmconf	• sqleatin - 連接 • sqledtin - 分離 • sqlfrsys - 重設資料庫管理程式架構 • sqlfusys - 更新資料庫管理程式架構 • sqlfxsys - 取得資料庫管理程式架構
dbsnap	• sqleatin - 連接 • sqlmonss - 取得 Snapshot
dbstart	• sqlepstart - 啓動資料庫管理程式
dbstop	• sqlefrce - 強迫終止應用程式 • sqlepstp - 停止資料庫管理程式
dcscat	• sqlegdad - 編目 DCS 資料庫 • sqlegdcl - 關閉 DCS 目錄掃描 • sqlegdel - 解編 DCS 資料庫 • sqlegdge - 取得資料庫的 DCS 目錄登錄 • sqlegdgt - 取得 DCS 目錄登錄 • sqlegdsc - 開啓 DCS 目錄掃描
dmscont	• sqleatin - 連接 • sqlecrea - 建立資料庫 • sqledrpd - 捨棄資料庫
ebcdicdb	• sqleatin - 連接 • sqlecrea - 建立資料庫 • sqledrpd - 捨棄資料庫
migrate	• sqlemgdb - 移轉資料庫
monreset	• sqleatin - 連接 • sqlmrset - 重設監督程式
monsz	• sqleatin - 連接 • sqlmonss - 取得 Snapshot • sqlmonsz - 估計 sqlmonss() 輸出緩衝區所需的大小

表 3. DB2 API 範例程式，不具有內含的 SQL (繼續)

範例程式	內含 API
nodecat	• sqlectnd - 編目節點 • sqlencls - 關閉節點目錄掃描 • sqlengne - 取得下一個節點目錄登錄 • sqlenops - 開啓節點目錄掃描 • sqleuncn - 解編節點
restart	• sqlerstd - 重新啓動資料庫
setact	• sqlesact - 設定帳戶字串
setrundg	• sqlesdeg - 設定執行時階次
sws	• sqleatin - 連接 • sqlmon - 取得/更新監督程式開關
utilapi	• sqlaintp - 取得錯誤訊息 • sqlogstt - 取得 SQLSTATE 訊息

DB2 API 內含的 SQL 範例

表 4. DB2 API 內含的 SQL 範例程式

範例程式	內含 API
asynrlog	• sqlurlog - 非同步讀取日誌
autocfg	• db2AutoConfig -- 自動架構 • db2AutoConfigMemory -- 自動架構可用記憶體 • sqlfudb -- 更新資料庫架構 • sqlfusys -- 更新資料庫管理程式架構 • sqlesetc -- 設定從屬站 • sqlaintp -- SQLCA 訊息
dbauth	• sqluadau - 取得授權
dbstat	• sqlureot - 重組表格 • sqlustat - Runstats
expsamp	• sqluexpr - 匯出 • sqluimpr - 匯入

表 4. DB2 API 內含的 SQL 範例程式 (繼續)

範例程式	內含 API
impexp	• sqluexpr - 匯出 • sqluimpr - 匯入
loadqry	• db2LoadQuery - 載入查詢
makeapi	• sqlabndx - 連結 • sqlaprep - 前置編譯程式 • sqlepstp - 停止資料庫管理程式 • sqlepstr - 啓動資料庫管理程式
rebind	• sqlarbnd - 重新連結
rechist	• sqlubkp - 備份資料庫 • sqluhcls - 關閉回復歷程檔掃描 • sqluhgne - 取得下一個回復歷程檔登錄 • sqluhops - 開啓回復歷程檔掃描 • sqluhprn - 刪除回復歷程檔 • sqluhupd - 更新回復歷程檔
tabscont	• sqlbctcq - 關閉表格空間配置區查詢 • sqlbftcq - 提取表格空間配置區查詢 • sqlbotcq - 開啓表格空間配置區查詢 • sqlbtcq - 表格空間配置區查詢 • sqlefmem - 可用記憶體
tabspace	• sqlbctsq - 關閉表格空間查詢 • sqlbftpq - 提取表格空間查詢 • sqlbgtss - 取得表格空間統計值 • sqlbmtsq - 表格空間查詢 • sqlbotsq - 開啓表格空間查詢 • sqlbstpq - 單一表格空間查詢 • sqlefmem - 可用記憶體
tload	• sqluexpr - 匯出 • sqluload - 載入 • sqluvqdp - 停止表格的表格空間

表 4. DB2 API 內含的 SQL 範例程式 (繼續)

範例程式	內含 API
tspace	• sqlbctcq - 關閉表格空間配置區查詢 • sqlbctsq - 關閉表格空間查詢 • sqlbftcq - 提取表格空間配置區查詢 • sqlbftpq - 提取表格空間查詢 • sqlbgtss - 取得表格空間統計值 • sqlbmtsq - 表格空間查詢 • sqlbotcq - 開啓表格空間配置區查詢 • sqlbotsq - 開啓表格空間查詢 • sqlbstpq - 單一表格空間查詢 • sqlbstsc - 設定表格空間配置區 • sqlbtcq - 表格空間配置區查詢 • sqlefmem - 可用記憶體
utilemb	• sqlaintp - 取得錯誤訊息 • sqlogstt - 取得 SQLSTATE 訊息

不具有 DB2 API 的內含 SQL 範例

表 5. 不具有 DB2 API 的內含 SQL 範例程式

範例程式名稱	程式說明
adhoc	示範將以交談方式處理 SQL 指令的動態 SQL 及 SQLDA 結構。 SQL 指令是由使用者輸入的，而對應於 SQL 指令的輸出將被傳回。
advsql	示範如何使用進階的 SQL 表示式，如 CASE、 CAST 及純量全選。
blobfile	示範如何操控「二進位大型物件 (BLOB)」，方法是從範例資料庫中讀取 BLOB 值，再置於檔案中。可以使用外部檢視器來顯示此檔案的內容。
columns	示範如何使用已利用動態 SQL，所處理過的游標。 此程式會在您要的綱目名稱下，列出 SYSCAT.COLUMNS 的結果集。
cursor	示範如何使用靜態 SQL 來使用游標。
delet	示範將從資料庫中刪除項目的靜態 SQL。
dynamic	示範如何使用動態 SQL 來使用游標。
joinsql	示範如何使用進階 SQL 結合表示式。
largevol	示範在分割環境中處理的平行查詢，以及如何使用 NFS 檔案系統，來自動合併結果集。只有 AIX 才提供。
lobeval	示範 LOB 定位器的用途，以及延遲實際 LOB 資料的評估。

表 5. 不具有 DB2 API 的內含 SQL 範例程式 (繼續)

範例程式名稱	程式說明
lobfile	示範 LOB 檔案 handle 的用途。
lobloc	示範 LOB 定位器的用途。
lobval	示範 LOB 的用途。
openftch	示範如何使用靜態 SQL 來提取、更新及刪除橫列。
recursql	示範進階 SQL 遞迴查詢的使用方式。
sampudf	示範如何實作「使用者定義類型」(UDT) 及「使用者定義函數」(UDF) 來修改表格登錄。本程式中宣告的所有 UDF 都是原始 UDF。
spclient	一種從屬站應用程式，會在 spserver 共用檔案庫中呼叫儲存程序。
spcreate.db2	一種 CLP Script，含有 CREATE PROCEDURE 陳述式以登記由 spserver 程式所建立的儲存程序。
spdrop.db2	一種 CLP Script，含有取消登記由 spserver 程式建立的儲存程序所需的 DROP PROCEDURE 陳述式。
spserver	示範儲存程序的伺服器程式。從屬站程式是 spclient。
static	示範靜態 SQL 如何擷取資訊。
tabsql	示範進階 SQL 表格表示式的使用方式。
tbdefine	示範建立及捨棄表格。
thdsrver	示範使用 POSIX 緒 API 來建立和管理緒。本程式維護環境定義儲存池。 有一個 generate_work 函數從 main 開始執行，並建立由工作者引線所執行的動態 SQL 陳述式。可使用環境定義之後，會建立及分派一個緒來執行指定工作。 所產生的工作包含陳述式，可刪除 sample 資料庫的 STAFF 或 EMPLOYEE 表格的登錄。 此程式僅適用 UNIX 平台。
trigsql	示範如何使用進階 SQL 觸發函式及限制。
udfcli	示範呼叫由 udfsrv 程式所建立的使用者定義函數 (UDF)，然後儲存在伺服器中以存取 sample 資料庫中的表格。
updat	示範如何使用靜態 SQL 來更新資料庫。
varinp	示範如何使用參數記號，將變數輸入「內含的動態 SQL」陳述式呼叫中。

使用者定義的函數範例

表 6. 使用者定義的函數範例程式

範例程式名稱	程式說明
DB2Udf.java	一種 Java UDF，示範許多作業，包括整數除法、操作「字元大型物件」(CLOB)，以及使用 Java 案例變數。
udfsrv.c	使用使用者定義的函數 ScalarUDF 建立檔案庫，以存取範例資料庫表格。

表 6. 使用者定義的函數範例程式 (繼續)

範例程式名稱	程式說明
UDFsrv.java	示範如何使用 Java 使用者定義的函數 (UDF)。

DB2 CLI 範例

表 7. DB2 Universal Database 的 CLI 範例程式

範例程式名稱	程式說明
共同公用程式檔	
utilcli.c	CLI 範例中所使用的公用程式函數。
utilapi.c	呼叫 DB2 API 的公用程式函數。
應用程式層次 - 處理 DB2 及 CLI 的應用程式層次範例。	
apinfo.c	如何取得及設定應用程式層次資訊。
aphndls.c	如何配置及釋放 handle。
apsqlca.c	如何使用 SQLCA 資料。
安裝映像檔層次 - 處理 DB2 及 CLI 的安裝映像檔層次範例。	
ilinfo.c	如何取得及設定安裝層次資訊 (如 CLI 驅動程式的版本)。
案例層次 - 處理 DB2 及 CLI 之案例層次的範例。	
ininfo.c	如何取得及設定案例層次資訊。
資料庫層次 - 處理 DB2 中資料庫物件的範例。	
dbconn.c	如何連接資料庫及切斷與資料庫的連接。
dbinfo.c	如何在資料庫層次上取得及設定資訊。
dbmconn.c	如何連接多個資料庫及切斷與資料庫的連接 (使用 DB2 API 來建立及捨棄次要資料庫)。
dbmuse.c	如何使用多個資料庫執行異動 (使用 DB2 API 來建立及捨棄次要資料庫)。
dbnative.c	如何將含有 ODBC escape 子句的陳述式轉換爲資料來源特定格式。
dbuse.c	如何使用資料庫物件。
dbusemx.sqc	如何使用與內含的 SQL 連結的單一資料庫。
表格層次 - 處理 DB2 中表格物件的範例。	
tbconstr.c	如何使用表格限制。
tbconstr.c	如何建立、變更及除去表格。
tbinfo.c	如何在表格層次上取得及設定資訊。
tbmod.c	如何修改表格中的資訊。
tbread.c	如何讀取表格中的資訊。
資料類型層次 - 處理資料類型的範例。	

表 7. DB2 Universal Database 的 CLI 範例程式 (繼續)

範例程式名稱	程式說明
dtinfo.c	如何取得資料類型的相關資訊。
dtlob.c	如何讀取及寫入 LOB 資料。
dtudt.c	如何建立、使用及捨棄使用者定義的特殊類型。
UDF 層次 - 示範使用者定義函數的範例。	
udfcli.c	呼叫 udfsrv.c 中使用者定義函數的從屬站應用程式。
udfsrv.c	udfcli.c 範例呼叫的使用者定義函數 ScalarUDF。
儲存程序層次 - 示範 CLI 中儲存程序的範例。	
spcreate.db2	用來發出 CREATE PROCEDURE 陳述式，CLP Script。
spdrop.db2	從型錄中捨棄儲存程序的 CLP Script。
spclient.c	用來呼叫在 spserver.c 中宣告之伺服器函數的從屬站程式。
spserver.c	在伺服器上建置及執行的儲存程序函數。
spcall.c	用來呼叫任何儲存程序的程式。

註: samples/cli 目錄中的其他檔案包括：

- README - 列示全部範例檔。
- makefile - 全部檔案的 Makefile
- 應用程式及儲存程序的建置檔案

Java 範例

表 8. Java Database Connectivity (JDBC) 範例程式

範例程式名稱	程式說明
DB2Appl.java	使用呼叫使用者的專用權來查詢範例資料庫的 JDBC 應用程式。
DB2Applt.java	一種 JDBC Applet，可使用 JDBC Applet 驅動程式來查詢資料庫。使用 DB2Applt.html 中指定的使用者名稱、通行碼、伺服器及埠號參數。
DB2Applt.html	內含 Applet 範例程式 DB2Applt 的 HTML 檔。它需要以伺服器及使用者資訊來自行設定。
DB2UdCli.java	一種 Java 從屬站應用程式，可呼叫 Java 使用者定義的函數 DB2Udf。
Dynamic.java	示範使用動態 SQL 的游標。
MRSPcli.java	此爲從屬站程式，可呼叫伺服器程式 MRSPsrv。程式示範 Java 儲存程序傳回的多重結果集。
MRSPsrv.java	此爲伺服器程式，由從屬站程式 MRSPcli 來呼叫。程式示範 Java 儲存程序傳回的多重結果集。
Outcli.java	一種 Java 從屬站應用程式，可呼叫 SQLJ 儲存程序 Outsrv。
PluginEx.java	一種 Java 程式，可將新的功能表項目及工具列按鈕新增至 DB2 Web 控制中心。

表 8. Java Database Connectivity (JDBC) 範例程式 (繼續)

範例程式名稱	程式說明
Spclient.java	一種 JDBC 從屬站應用程式，可呼叫 Spserver 儲存程序類別中的 PARAMETER STYLE JAVA 儲存程序。
Spcreate.db2	一種 CLP Script，含有 CREATE PROCEDURE 陳述式，可以將 Spserver 類別中所含的方法登記爲儲存程序。
Spdrop.db2	一種 CLP Script，含有取消登記 Spserver 類別所含的儲存程序時所需的 DROP PROCEDURE 陳述式。
Spserver.java	示範 PARAMETER STYLE JAVA 儲存程序的 JDBC 程式。從屬站程式是 Spclient.java。
UDFcli.java	一種 JDBC 從屬站應用程式，可呼叫 Java 使用者定義函數的檔案庫 UDFsrv 中的函數。
UseThrds.java	顯示如何使用緒來非同步執行 SQL 陳述式 (JDBC 版本的 CLI 範例 async.c)。
V5SpCli.java	一種 Java 從屬站應用程式，可呼叫 DB2GENERAL 儲存程序 V5Stp.java。
V5Stp.java	示範在伺服器中更新 EMPLOYEE 表格的 DB2GENERAL 儲存程序，並將新的薪資及員工名冊傳回從屬站。從屬站程式是 V5SpCli.java。
Varinp.java	示範如何使用參數記號，將變數輸入「內含的動態 SQL」陳述式呼叫中。

表 9. 內含的 SQL for Java (SQLJ) 範例程式

範例程式名稱	程式說明
App.sqlj	使用靜態 SQL，從範例資料庫的 EMPLOYEE 表格中擷取及更新資料。
Applt.sqlj	一種 applet，可使用 JDBC Applet 驅動程式來查詢資料庫。 使用 Applt.html 中指定的使用者名稱、通行碼、伺服器及埠號參數
Applt.html	內含 Applet 範例程式 Applt 的 HTML 檔。它需要以伺服器及使用者資訊來自行設定。
Cursor.sqlj	示範使用靜態 SQL 的疊代。
OpF_Curs.sqlj	Openftch 程式的類別檔。
Openftch.sqlj	示範如何使用靜態 SQL 來提取、更新及刪除橫列。
Outsrv.sqlj	示範使用 SQLDA 結構的儲存程序。此程序在 sample 資料庫的 STAFF 表格中，將員工的中間薪資填入 SQLDA 內。在資料庫處理完畢後 (尋找中間薪資)，儲存程序將已填入的 SQLDA 及 SQLCA 狀態傳回 JDBC 從屬站應用程式 Outcli 。
Stclient.sqlj	一種 SQLJ 從屬站應用程式，可以呼叫 SQLJ 儲存程序程式 Stserver 建立的 PARAMETER STYLE JAVA 儲存程序。
Stcreate.db2	一種 CLP Script，含有 CREATE PROCEDURE 陳述式，可以將 Stserver 類別中所含的方法登記爲儲存程序。
Stdrop.db2	一種 CLP Script，含有取消登記 Stserver 類別中所含的儲存程序時所需的 DROP PROCEDURE 陳述式。

表 9. 內含的 SQL for Java (SQLJ) 範例程式 (繼續)

範例程式名稱	程式說明
Stserver.sqlj	，示範 PARAMETER STYLE JAVA 儲存程序 SQLJ 程式。從屬站程式是 Stclient.sqlj 。
Static.sqlj	使用靜態 SQL 來擷取資訊。
Stp.sqlj	一種儲存程序，可更新伺服器上的 EMPLOYEE 表格，將新的薪資及報酬資訊傳回 JDBC 從屬站程式 StpCli。
UDFclie.sqlj	一種從屬站應用程式，可呼叫 Java 使用者定義函數的檔案庫 UDFsrv 中的函數。
Updat.sqlj	使用靜態 SQL 來更新資料庫。

SQL 程序範例

表 10. SQL 程序範例程式

範例程式名稱	程式說明
basecase.db2	UPDATE_SALARY 程序會提升 ″sample″ 資料庫中 ″staff″ 表格的 ″empno″ IN 參數所識別的員工薪資。程序會根據使用 ″rating″ IN 參數的 CASE 陳述式，判斷提升幅度。
basecase.sqc	呼叫 UPDATE_SALARY 程序。
baseif.db2	UPDATE_SALARY_IF 程序會提升 ″sample″ 資料庫中 ″staff″ 表格的 ″empno″ IN 參數所識別的員工薪資。程序會根據使用 ″rating″ IN 參數的 IF 陳述式，判斷提升幅度。
baseif.sqc	呼叫 UPDATE_SALARY_IF 程序。
dynamic.db2	CREATE_DEPT_TABLE 程序會使用動態 DDL 來建立新的表格。表格的名稱取決於程序的 IN 參數值。
dynamic.sqc	呼叫 CREATE_DEPT_TABLE 程序。
iterate.db2	ITERATOR 程序會使用 FETCH 迴圈，從 ″department″ 表格中擷取資料。如果 ″deptno″ 直欄的值不是 ’D11’，則會將修改的資料插入 ″department″ 表格。如果 ″deptno″ 直欄的值是 ’D11’，則 ITERATE 陳述式會將控制串流傳回 LOOP 陳述式的開頭。
iterate.sqc	呼叫 ITERATOR 程序。
leave.db2	LEAVE_LOOP 程序會計算在 ″not_found″ 條件 handler 呼叫 LEAVE 陳述式之前，在 LOOP 陳述式中執行的 FETCH 作業數。 LEAVE 陳述式會讓控制串流跳出迴圈，並完成儲存程序。
leave.sqc	呼叫 LEAVE_LOOP 程序。
loop.db2	LOOP_UNTIL_SPACE 程序會計算在游標擷取到一列值爲空格 (’ ’) 的直欄 ″midinit″ 之前，在 LOOP 陳述式中執行的 FETCH 作業數。 loop 陳述式會讓控制串流跳出迴圈，並完成儲存程序。

表 10. SQL 程序範例程式 (繼續)

範例程式名稱	程式說明
loop.sqc	呼叫 LOOP_UNTIL_SPACE 程序。
nestcase.db2	BUMP_SALARY 程序會使用巢狀 CASE 陳述式，提升 ″sample″ 資料庫中 ″staff″ 表格的 dept IN 參數所識別的部門員工薪資。
nestcase.sqc	呼叫 BUMP_SALARY 程序。
nestif.db2	BUMP_SALARY_IF 程序會使用巢狀 IF 陳述式，提升 ″sample″ 資料庫中 ″staff″ 表格的 dept IN 參數所識別的部門員工薪資。
nestif.sqc	呼叫 BUMP_SALARY_IF 程序。
repeat.db2	REPEAT_STMT 程序會計算在游標無法再擷取更多橫列之前，在 repeat 陳述式中執行的 FETCH 作業數。條件 handler 會讓控制串流跳出重複的迴圈，並完成儲存程序。
repeat.sqc	呼叫 REPEAT_STMT 程序。
rsultset.c	呼叫 MEDIAN_RESULT_SET 程序，顯示中間薪資，然後顯示 SQL 程序所產生的結果集。此從屬站是使用 CLI API 撰寫的，可以接受結果集。
rsultset.db2	MEDIAN_RESULT_SET 程序會取得 ″sample″ 資料庫中 ″staff″ 表格的 ″dept″ IN 參數所識別的部門員工之中間薪資。中間值會被指派爲薪資 OUT 參數，並傳回 ″rsultset″ 從屬站。然後程序會開啓 WITH RETURN 游標，傳回其薪資大於中間值的員工結果集。程序會將結果集傳回從屬站。
spserver.db2	此 CLP Script 中的 SQL 程序會示範基本錯誤處理常式、巢狀儲存程序呼叫，並將結果集傳回從屬站應用程式或呼叫應用程式。您可以使用 CLI 範例目錄中的 ″spcall″ 應用程式，來呼叫程序。也可以使用 ″spclient″ 應用程式 (位於 C 及 CPP 範例目錄)，呼叫沒有傳回結果集的程序。
whiles.db2	DEPT_MEDIAN 程序會取得 ″sample″ 資料庫中 ″staff″ 表格的 ″dept″ IN 參數所識別的部門員工之中間薪資。中間值會被指派給薪資 OUT 參數，並傳回 ″whiles″ 從屬站。然後，whiles 從屬站會列印中間薪資。
whiles.sqc	呼叫 DEPT_MEDIAN 程序。

ADO、RDO 及 MTS 範例

表 11. ADO、RDO 及 MTS 範例程式

範例程式名稱	程式說明
Bank.vbp	一種 RDO 程式，可建立及維護銀行分行的資料，能夠在客戶帳戶上執行異動。此程式可以利用使用者指定的資料庫，因爲包含 DDL 來建立必要的表格，供應用程式用來儲存資料。
Blob.vbp	此 ADO 程式示範如何擷取 BLOB 資料。從 sample 資料庫的 emp_photo 表格中擷取及顯示圖片。此程式亦可使用本端檔案中的影像來取代 emp_photo 表格中的影像。

表 11. ADO、RDO 及 MTS 範例程式 (繼續)

範例程式名稱	程式說明
BLOBAccess.dsw	此範例示範如何使用 Microsoft Visual C++ 來高亮度標示 ADO/Blob 存取。類似於 Visual Basic 範例 Blob.vbp。BLOB 範例具有兩個 main 函數： 1. 從範例資料庫中讀取 BLOB，顯示在螢幕上。 2. 從檔案中讀取 BLOB，並插入資料庫中。(匯入)
Connect.vbp	此 ADO 程式將建立連接物件，並連接至 sample 資料庫。完成之後，程式將切斷連結，並且結束。
Commit.vbp	此應用程式示範如何使用 ADO 的「自動 COMMIT」/「手動 COMMIT」特性。程式在 sample 資料庫的 EMPLOYEE 表格中，查詢員工編號及姓名。使用者可以選擇使用「自動 COMMIT」或「手動 COMMIT」模式來連接資料庫。在「自動 COMMIT」模式中，使用者在記錄上的所有變動，將會在資料庫上自動更新。在「手動 COMMIT」模式中，使用者在做任何變更之前，必須先執行異動作業。異動之後的變更，可以透過回捲作業來還原。確認異動之後，即永久地儲存變更。結束程式會自動還原變更。
db2com.vbp	此 Visual Basic 專案示範如何使用 Microsoft 異動伺服器來更新資料庫。專案會建立從屬站程式 db2mts.vbp 所用的伺服器 DLL，以及四個類別模組： • UpdateNumberColumn.cls • UpdateRow.cls • UpdateStringColumn.cls • VerifyUpdate.cls 此程式中，sample 資料庫內會建立一個暫時表格 DB2MTS。
db2mts.vbp	此爲從屬站程式的 Visual Basic 專案，使用 Microsoft 異動伺服器來呼叫由 db2com.vbp 所建立的伺服器 DLL。
Select-Update.vbp	此 ADO 程式執行的函數與 Connect.vbp 相同，但另外提供 GUI 介面。使用者可透過此介面，檢視、更新及刪除 sample 資料庫中 ORG 表格的資料。
Sample.vbp	此 Visual Basic 專案透過 ADO 使用 Keyset 游標，提供圖形式使用者介面給 sample 資料庫中的所有資料。
VarCHAR.dsp	一個 Visual C++ 程式，使用 ADO 將 VarChar 資料當作文字欄位來存取。提供圖形式使用者介面，供使用者用來檢視及更新 sample 資料庫中 ORG 表格的資料。

物件連結和內嵌範例

表 12. 物件連結和內嵌 (OLE) 範例程式

範例程式名稱	程式說明
sales	示範有關 Microsoft Excel 銷售試算表的向上捲動查詢（使用 Visual Basic）。
names	查詢 Lotus Notes 通訊錄 (使用 Visual Basic)。

表 12. 物件連結和內嵌 (OLE) 範例程式 (繼續)

範例程式名稱	程式說明
inbox	透過 OLE/Messaging 查詢 Microsoft Exchange 信箱電子郵件訊息（使用 Visual Basic）。
invoice	以電子郵件附件的形式傳送 Microsoft Word 發票文件的 OLE 自動化使用者定義的函數 (使用 Visual Basic)。
bcounter	一種 OLE 自動化使用者定義的函數，示範使用案例變數的 scratch pad (在 Visual Basic 中執行)。
ccounter	計數器 OLE 自動化使用者定義的函數（使用 Visual C++）。
salarysrv	計算 sample 資料庫 STAFF 表格內中間薪資的 OLE 自動化儲存程序 (使用 Visual Basic)。
salarycltvc	一種 Visual C++ DB2 CLI 範例，會呼叫 Visual Basic 儲存程序 salarysrv。
salarycltvb	一種 Visual Basic DB2 CLI 範例，會呼叫 Visual Basic 儲存程序 salarysrv。
salsvado	一種 OLE 自動化儲存程序 (在 32 位元 Visual Basic 及 ADO 中執行)，會藉由計算新建立的表格 STAFF2 中的中間薪資來示範輸出參數，並藉由從該表格擷取薪資來示範結果集。
salclado	一種 Visual Basic 從屬站，會呼叫 Visual Basic 儲存程序 salsvado。
testcli	一種 OLE 自動化內含的 SQL 從屬站應用程式，會呼叫儲存程序 tstsrv (在 Visual C++ 中執行)。
tstsrv	一種 OLE 自動化儲存程序，會示範在從屬站及儲存程序間不同的傳送類型 (在 Visual C++ 中執行)。

表 13. 物件連結和內嵌資料庫 (OLE DB) 表格函數

範例程式名稱	程式說明
jet.db2	Microsoft.Jet.OLEDB.3.51 提供者
mapi.db2	MAPI 的 INTERSOLV 連接 OLE DB
msdaora.db2	Oracle 的 Microsoft OLE DB 提供者
msdasql.db2	ODBC 驅動程式的 Microsoft OLE DB 提供者
msidxs.db2	Microsoft OLE DB 索引伺服器提供者
notes.db2	Notes 的 INTERSOLV 連接 OLE DB
sampprov.db2	Microsoft OLE DB 範例提供者
sqloledb.db2	SQL 伺服器的 Microsoft OLE DB 提供者

命令行處理器範例

表 14. 命令行處理器 (CLP) 範例程式。

範例檔名	檔案說明
const.db2	建立一個具有 CHECK CONSTRAINT 子句的表格。
cte.db2	示範一個常用的表格表示式。 示範這個進階 SQL 陳述式的對等範例程式爲 tabsql。
flt.db2	示範遞迴查詢。 示範這個進階 SQL 陳述式的對等範例程式爲 recursql。
join.db2	示範表格的外部結合。 示範這個進階 SQL 陳述式的對等範例程式爲 joinsql。
stock.db2	示範如何使用起始動作定義。 示範這個進階 SQL 陳述式的對等範例程式爲 trigsql。
testdata.db2	使用 DB2 內建函數 (如 RAND() 及 TRANSLATE())，將具有隨機產生的測試資料表格的移入。
thaisort.db2	這個 Script 特別爲泰語使用者撰寫。泰語是按照語音順次序排序，需要進行前排序/交換主要母音及其子音作業，也需要進行後排序作業，才能以正確的排列順序來檢視資料。 此檔案是藉由建立 UDF 函數前排序及後排序，並建立一個表格來執行泰語排序作業；然後再依表格呼叫函數來排列表格資料順序。 若要執行此程式，首先您必須從 C 來源檔 udf.c 建立使用者定義的函數程式 udf。

日誌管理使用者跳出程式範例

表 15. 日誌管理使用者跳出範例程式。

範例檔名	檔案說明
db2uext2.cadsm	這是一個利用 ADSTAR DSM (ADSM) API 來保存及擷取資料庫日誌檔的範例「使用者跳出程式」。 此範例提供呼叫的審核追蹤 (每一個選項儲存在不同的檔案中)，包括時間戳記和已接收參數。 本範例也提供發生錯誤的呼叫之錯誤追蹤，包括時間戳記與用來判斷問題所在之已隔離出錯誤的字串。 您可以停用這些選項。 該檔案必須更名爲 db2uext2.c 並編譯爲 C 程式。 在 UNIX 及 Windows 32 位元作業系統上提供。 OS/2 版本是 db2uexit.cad。 **註:** 在 AIX 上使用層次 3.1.6 及以上之 ADSM API Client 的應用程式必須使用 xlc_r 或 xlC_r 編譯器呼叫來建立，而不是使用 xlc 或 xlC，即使應用程式爲單緒。這會確保檔案庫提供安全的緒。如果您擁有一個使用 non-thread-safe 之檔案庫編譯的應用程式，則可以引用 Fixtest IC21925E 或與您的應用程式提供者連絡。Fixtest 可在 index.storsys.ibm.com 匿名 FTP 伺服器上找到。這會使 ADSM API 層次退回到 3.1.3。
db2uexit.cad	這是 OS/2 版本的 db2uext2.cadsm。該檔案必須更名爲 db2uexit.c 並編譯爲 C 程式。
db2uext2.cdisk	此爲範例「使用者跳出程式」，利用所在特定平台的系統複製指令。此程式可保存、擷取資料庫日誌檔，並提供呼叫的審核追蹤 (每一個選項儲存在不同的檔案中) ，包括時間戳記和收到的參數。本範例也提供發生錯誤的呼叫之錯誤追蹤，包括時間戳記與用來判斷問題所在之已隔離出錯誤的字串。 您可以停用這些選項。 該檔案必須更名爲 db2uext2.c 並編譯爲 C 程式。 在 UNIX 及 Windows 32 位元作業系統上提供。

表 15. 日誌管理使用者跳出範例程式。 (繼續)

範例檔名	檔案說明
db2uext2.ctape	此爲範例「使用者跳出程式」，利用所在特定 UNIX 平台的系統磁帶指令。 程式可以保存及擷取資料庫日誌檔。系統磁帶指令的所有限制即爲此使用者跳出程式的限制。此範例提供呼叫的審核追蹤 (每一個選項儲存在不同的檔案中)，包括時間戳記和已接收參數。本範例也提供發生錯誤的呼叫之錯誤追蹤，包括時間戳記與用來判斷問題所在之已隔離出錯誤的字串。 您可以停用這些選項。 該檔案必須更名爲 db2uext2.c 並編譯爲 C 程式。 只適用於 UNIX 平台。
db2uext2.cxbsa	這是一個利用 XBSA API 來「保存」及「擷取」資料庫日誌檔案的範例「使用者跳出程式」。該範例提供呼叫的審核追蹤 (每一個選項儲存在不同的檔案中)，包括時間戳記和已接收參數。本範例也提供發生錯誤的呼叫之錯誤追蹤，包括時間戳記與用來判斷問題所在之已隔離出錯誤的字串。 您可以停用這些選項。 該檔案必須更名爲 db2uext2.c 並編譯爲 C 程式。 只適用於 UNIX 及 Windows 平台。

第2章 設定

若要建立適當的環境來建置及執行 DB2 應用程式，您需要適當地設置下列項目：

1. 編譯器或直譯器
2. DB2 (資料庫管理程式、DB2 AD Client 及從屬站連接)
3. 作業系統環境
4. DB2 範例資料庫 (可選用的)

檢查編譯器或直譯器環境

爲了開發 DB2 程式，您必須使用作業系統支援的其中一種程式設計語言的編譯器或直譯器，請參閱第6頁的『平台支援的軟體』中的列示。最好先建立一個非 DB2 應用程式，確定現存的編譯器或直譯器環境已設定妥當。 然後，如果您遇到任何問題，請參閱您的編譯器或直譯器所附的文件。

設置 DB2 環境

若要設定 DB2 環境，下列項目必須安裝且可以使用：

- 具有適合於您的環境的資料庫案例的伺服器上的資料庫管理程式。如果您需要有關資料庫案例的資訊，請參閱第367頁的『附錄A. 關於資料庫管理程式案例』。
- DB2 AD Client，其位於您要開發應用程式的從屬站或伺服器工作站上。
- 對遠端伺服器的連接 (若您在從屬站工作站上開發)。

更新「資料庫管理程式架構檔」

此檔案包含用於開發應用程式的重要設定值。您可以輸入下列指令以變更這些設定值：

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

且可以輸入下列指令以檢視設定值：

```
db2 get dbm cfg
```

請參閱 *Command Reference* ，取得如何使用這些指令的其餘相關資訊。

- 對於儲存程序，關鍵字 KEEPDARI 的預設值是 yes。這會讓儲存程序處理保持在作用中。如果您要開發儲存程序，則可能要多測試幾次將相同的儲存程序檔案庫。此預設設定可能會干擾檔案庫的重新載入。在開發儲存程序時，最好將此關鍵字的值變更爲 no，然後在您準備好要載入最新版的儲存程序時，再將它變更回 yes。
- 若是 Java，請更新 JDK11_PATH 關鍵字。請參閱第60頁的『設定環境』，取得詳細資訊。

關於安裝及設置的詳細資訊，請參閱作業系統的*快速入門*書籍。

確定上述項目已安裝且正常運作之後，您可以根據下列其中一段的步驟，設定作業系統環境：

- 『設定 OS/2 環境』
- 第35頁的『設定 UNIX 環境』
- 第36頁的『設定 Windows 32 位元作業系統環境』

在您設定作業系統環境後，您也許想建立本書範例程式所使用的範例資料庫。若要建立資料庫，請參閱第40頁的『建立、編目及連結範例資料庫』。

設定 OS/2 環境

大部份 OS/2 編譯器使用環境變數，控制各種選項。您可以在 CONFIG.SYS 檔案中設定這些變數，或建立指令檔來設定變數。

CONFIG.SYS

在 CONFIG.SYS 檔案中設定環境變數的優點，就是一旦輸入後，每次啓動電腦時系統便會，自動設定它們。

指令檔

在指令檔中設定環境變數的優點，就是具有捷徑及彈性來使用許多編譯器。缺點是您必須在啓動每一個程式設計階段作業時，執行指令檔。

若執行指令檔來設定環境變數，則必須在設定環境變數的相同視窗中，建置您的應用程式。若在另一個視窗中建置應用程式，則使用的選項將不同於第一個視窗中設定的選項。

在安裝 DB2 AD Client 時，在CONFIG.SYS 檔中會插入下列陳述式：

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

本書中的指令檔假設此陳述式已存在。如果您在安裝 DB2 AD Client 之後編輯 CONFIG.SYS 檔，請確定此陳述式未被移除。

另外，下列環境變數由 DB2 自動更新：

- PATH，包括目錄 %DB2PATH%\bin
- LIBPATH，包括目錄 %DB2PATH%\dll

關於 DB2 更新的 Java 環境變數，請參閱第68頁的『OS/2』。

此外，若使用下列其中一種程式設計語言， CONFIG.SYS 檔案中必須包含適當的陳述式：

C/C++ set INCLUDE=%DB2PATH%\include;%INCLUDE%

FORTAN

```
set FINCLUDE=%DB2PATH%\include;%FINCLUDE%
```

IBM COBOL

```
set SYSLIB=%SYSLIB%;%DB2PATH%\include\cobol_a
```

Micro Focus COBOL

```
set COBCPY=%DB2PATH%\include\cobol_mf;%COBCPY%
```

在 OS/2 上，除了 DB2PATH 及 DB2INSTPROF 之外，在 CONFIG.SYS 中不應再定義任何 DB2 環境變數。所有 DB2 變數應該於 DB2「案例設定檔登記」中定義在整體層次、案例層次或案例節點層次 (平行修訂版)。 使用 db2set.exe 指令來設定、修改及列示變數。

註: 如果您設定了 DB2INSTDEF 登記變數，則不需要 DB2INSTANCE。如果未設定 DB2INSTANCE，則它會定義所使用的預設案例名稱。

設定 UNIX 環境

您需要設定環境變數，方可存取當安裝資料庫管理程式時，所建立的資料庫案例。 *DB2 for UNIX 快速入門*一書提供有關設定環境變數的一般資訊。本節提供設定環境變數來存取資料庫案例的特定指示。

每一個資料庫管理程式都有兩個案例檔案， db2profile 及 db2cshrc，其中包含用來設定該案例環境變數的 Script。根據所使用的 shell，輸入下列指令來執行 script：

對於 bash 或 Korn shell 而言：

```
. $HOME/sqllib/db2profile
```

對 C shell 而言：

```
source $HOME/sqllib/db2cshrc
```

其中 $HOME 是案例擁有者的起始目錄。

爲了方便起見，您可在 .profile 或 .login 檔中併入這個指令，以便登入時自動執行。

空白檔案 sqllib/userprofile 及 sqllib/usercshrc 是在案例建立期間建立的，以容許使用者放置他們自己的案例環境設定值。在任何 DB2 FixPak 或未來版本安裝的案例更新 (db2iupdt) 期間，均不會對這些檔案加以修改。如果您不想要 db2profile 或 db2cshrc Script 中的新環境設定，可以使用相對應的「使用者」Script 來置換它們 (在 db2profile 或 db2cshrc Script 的結尾有「使用者」Script 的具體叫法)。在案例移轉期間 (db2imigr)，會將使用者 Script 複製過去，以便繼續使用您的環境修改。這些使用者 Script 僅在 DB2 版本 7 及以後的版本中可用。

根據您所在的 UNIX 平台，DB2 案例建立期間會自動更新下列環境變數：

AIX:

- PATH，包括許多 DB2 目錄，其中包括 sqllib/bin
- LIBPATH，包括目錄 sqllib/lib

HP-UX:

- PATH，包括許多 DB2 目錄，其中包括 sqllib/bin
- SHLIB_PATH，包括目錄 sqllib/lib

Linux、PTX 及 Solaris：

- PATH，包括許多 DB2 目錄，其中包括 sqllib/bin
- LD_LIBRARY_PATH，包括目錄 sqllib/lib

Silicon Graphics IRIX:

- PATH，包括許多 DB2 目錄，其中包括 sqllib/bin
- LD_LIBRARY_PATH，包括目錄 sqllib/lib (o32 物件類型應用程式所需要的)
- LD_LIBRARYN32_PATH，包括目錄 sqllib/lib32 (n32 物件類型應用程式所需要的)

關於 DB2 更新的 Java 環境變數，請參閱第60頁的『設定環境』。

設定 Windows 32 位元作業系統環境

在 Windows NT 或 Windows 2000 中安裝 DB2 AD Client 時，安裝程式會更新架構登記的環境變數 INCLUDE、LIB、PATH、DB2PATH 及 DB2INSTANCE。預設案例是 DB2。

關於 DB2 更新的 Java 環境變數，請參閱第74頁的『Windows 32 位元作業系統』。

在 Windows 95、Windows 98 或 Windows Millennium Edition 上安裝 DB2 AD Client 時，安裝程式會更新autoexec.bat 檔。

您可以置換這些環境變數，設定機器或目前登入使用者的值。若要置換這些值，請使用下列其中一項：

- Windows NT 控制台
- Windows 2000 控制台
- Windows 95、Windows 98，或 Windows Millennium Edition 指令視窗
- Windows 95、Windows 98，或 Windows Millennium Edition autoexec.bat 檔

註:

1. 變更這些環境變數時需要特別小心。**不要**變更 DB2PATH 環境變數。
2. 在指令中使用變數 %DB2PATH% 時，請以雙引號括住完整路徑，如 set LIB="%DB2PATH%\lib";%LIB%。在 DB2 版本 7 中，此變數的預設安裝值是 \Program Files\sqllib，其中含有空格，所以如果不使用雙引號會造成錯誤。

可以更新這些環境變數，以便在 Windows 32 位元作業系統上執行更多程式。此外，爲了執行 DB2 應用程式，您必須採取下列特定步驟：

- 在建置 C 或 C++ 程式時，您必須確定 INCLUDE 環境變數含有 %DB2PATH%\INCLUDE 作爲第一個目錄。

  例如，Microsoft Visual C++ 編譯器環境設定檔 Vc\bin\vcvars32.bat 具有下列指令：

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

  欲以 DB2 使用此檔案，請先將會設定 %DB2PATH%\INCLUDE 路徑的 %INCLUDE%，從列示的尾端移到開頭，如下所示：

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

- 建置 Micro Focus COBOL 程式時，請設定 COBCPY 環境變數去指向 %DB2PATH%\INCLUDE\cobol_mf。
- 建置 IBM COBOL 程式時，請設定 SYSLIB 環境變數去指向 %DB2PATH%\INCLUDE\cobol_a。
- 確定 LIB 環境變數是指向 %DB2PATH%\lib，作法如下：

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

- 確定遠端資料庫的伺服器上已設定 DB2COMM 環境變數。

- 確定伺服器及從屬站上已啓動機密保護服務程式，以進行 SERVER 身分驗證，視 CLIENT 身分驗證的層次而定。 若要啓動機密保護服務程式，請使用 NET START DB2NTSECSERVER 指令。

註:

1. 所有 DB2 環境變數皆可定義於使用者環境中，或設定成登記變數。請參閱 *Administration Guide*，取得登記變數的相關資訊。請參閱*Command Reference*，取得 db2set 指令的相關資訊。
2. DB2INSTANCE 應該只定義在使用者環境層次。若使用 DB2INSTDEF 登記變數，以便於未設定時可以定義預設的案例名稱，則不再需要 DB2INSTANCE。
3. Windows NT 或 Windows 2000 環境上的資料庫管理程式是一種 Windows NT 服務或 Windows 2000 服務，因此，只要已順利啓動服務程式，不論是否發生其它問題，都不會傳回錯誤或警告。這表示執行 db2start 或 NET START 指令時，即使無法啓動通信子系統，亦不傳回任何警告。因此，使用者恆應檢查 Windows NT 或 Windows 2000 事件日誌或 DB2DIAG.LOG，找出這些指令在執行時可能發生的錯誤。

啓用伺服器上的通信

本節解釋如何連接 DB2 Universal Database 伺服器。

開始安裝、編目及連結 sample 資料庫之前，您應該先確定伺服器是否正常運作，並架構爲支援所要編目的通訊協定。 在伺服器上執行下列動作：

1. 確定已設定 db2comm 環境變數。例如，如果要使用 TCP/IP，請輸入：

   ```
   db2set DB2COMM=tcpip
   ```

   並確定已架構了 TCP/IP 支援的通訊協定。

   關於在服務程式檔中新增 TCP/IP 設定的指示，請參閱平台的 *快速入門* 書籍。
2. 啓動資料庫案例，輸入：

   ```
   db2start
   ```

必須自從屬站執行公用程式對範例資料庫的連結。若需其餘資訊，請參閱第42頁的『連結』。

Windows NT 及 Windows 2000

在 DB2 for Windows NT 或 DB2 for Windows 2000 的生產系統中，您必須將資料庫案例啓動成服務程式。步驟如下：

- 如果使用的是通訊協定，請確定已在 Windows NT 或 Windows 2000 控制台的「系統環境變數」區段中設定了 db2comm 環境變數。

- 啓動機密保護服務程式。此動作可以自動完成 (請參閱下面的備註)，或您可以使用下列指令來自行啓動此服務程式：

  ```
  NET START DB2NTSECSERVER
  ```

- 輸入下列指令，啓動案例：

  ```
  db2start
  ```

自動啓動「機密保護服務程式」。 一般而言，唯一需要將機密保護服務程式設定成自動啓動的時機，就是工作站扮演連接至伺服器的 DB2 從屬站，而伺服器已架構成具有「從屬站身分驗證」的功能。若要自動啓動機密保護服務程式，請執行下列動作：

Windows NT

1. 按一下「開始」按鈕。
2. 按一下「設定」。
3. 按一下「控制台」。
4. 在「控制台」中，按一下「服務程式」。
5. 在「服務程式」視窗中，強調顯示「DB2 安全伺服器」。
6. 若未出現「已啓動」和「自動」設定，請按一下「啓動」。
7. 按一下「自動」。
8. 按一下「確定」。
9. 重新啓動機器，使設定生效。

Windows 2000

1. 按一下「開始」按鈕。
2. 按一下「設定」。
3. 按一下「控制台」。
4. 按一下「管理工具」。
5. 按一下「服務程式」。
6. 在「服務程式」視窗中，強調顯示「DB2 安全伺服器」。
7. 如果沒有列出「已啓動」及「自動」設定，請在頂端的功能表上按一下「動作」。
8. 按一下「內容」。
9. 請確定您是在「一般事項」標籤中。
10. 從「啓動類型」下拉式功能表中，選取「自動」。
11. 按一下「確定」。
12. 重新啓動機器，使設定生效。

建立、編目及連結範例資料庫

若要使用 DB2 提供的範例程式，您必須在伺服器工作站上建立 sample 資料庫。請參閱 *SQL Reference* 取得 sample 資料庫內容的列示。

若要使用遠端從屬站來存取伺服器的 sample 資料庫，您必須在從屬站工作站上編目 sample 資料庫。

另外，若要使用遠端從屬站來存取伺服器的 sample 資料庫，而伺服器上執行不同版本的 DB2 或不同的作業系統，則需要將資料庫公用程式 (包括 DB2 CLI) 連結到 sample 資料庫。

建立

欲建立 sample 資料庫，您必須具有 SYSADM 權限。關於 SYSADM 權限的詳細資訊，請參閱作業系統的*快速入門*書籍。

若要建立資料庫，請在伺服器上執行下列動作：

1. 確定 db2sampl (用來建立 sample 資料庫的程式) 的位置在您的路徑中。db2profile 或 db2cshrc 檔案會將 db2sampl 放入您的路徑中，因此除非改變，否則位置保持不變。
   - 在 UNIX 伺服器上，db2sampl 位於：

     ```
     $HOME/sqllib/bin
     ```

     其中 $HOME 是 DB2 案例擁有者的起始目錄
   - 在 OS/2 及 Windows 上，db2sampl 位於：

     ```
     %DB2PATH%\bin
     ```

     其中 %DB2PATH% 即是 DB2 的安裝所在
2. 請確定 DB2INSTANCE 環境變數已設定爲您要建立 sample 資料庫的案例名稱。如果未設定，您可以使用下列指令來設定：
   - 在 UNIX 平台上：

     您可以對 bash 或 Korn shell 進行設定，請輸入：

     ```
     DB2INSTANCE=instance_name
     export DB2INSTANCE
     ```

     對 C shell 執行此動作輸入下列指令：

     ```
     setenv DB2INSTANCE instance_name
     ```
   - 在 OS/2 及 Windows 上，輸入：

     ```
     set DB2INSTANCE=instance_name
     ```

其中 *instance_name* 即是資料庫案例的名稱

3. 在輸入 db2sampl 後緊接著輸入您要建立範例資料庫的位置，來建立 sample 資料庫。 在 UNIX 平台上，此爲 *path*，輸入方式如下：

   ```
   db2sampl path
   ```

   在 OS/2 及 Windows 上，這是 *drive*，其輸入方式如下：

   ```
   db2sampl drive
   ```

   如果您未指定路徑或磁碟機，則安裝程式將範例表格安裝在資料庫管理程式架構檔中的 DFTDBPATH 參數所指定的預設路徑或磁碟機中。如果您需要有關架構檔的資訊，請參閱 *Administration Guide*。

   資料庫與在其中建立它的案例具有相同的身分驗證類型。如果您需要有關在建立資料庫案例時，指定身分驗證的詳細資訊，請參閱*快速入門* 一書

在主電腦或 AS/400 伺服器上建立

如果您要對主電腦伺服器 (如 DB2 for OS/390 或 AS/400 伺服器) 執行範例程式，則必須建立一個資料庫，其中包含了 *SQL Reference* 中所述的範例表格。您可以參照範例程式 expsamp，其中使用 STAFF 和 ORG 表格，示範如何使用 API，在 DB2 Connect 資料庫之間匯入及匯出表格和表格資料。

註: 工作站上的 DB2 與主電腦系統上的 DB2 存在一些 SQL 語法及 DB2 指令方面的差異。當存取 DB2 for OS/390 或 DB2 for AS/400 上的資料庫時，請確定您的程式使用這些資料庫系統支援的 SQL 陳述式及前置編譯/連結選項。

若要建立資料庫，請：

1. 使用 db2sampl，在 DB2 伺服器案例中建立 sample 資料庫。
2. 連接 sample 資料庫。
3. 將範例表格匯出到檔案中。
4. 連接 DB2 Connect 資料庫。
5. 建立範例表格。
6. 匯入範例表格。

若您需要匯出及匯入檔案的相關資訊，請參照 *Data Movement Utilities Guide and Reference*。若需關於與資料庫連接及建立表格的資訊，請參閱 *SQL Reference*。

編目

若要從遠端從屬站來存取伺服器的 sample 資料庫，您必須在從屬站工作站上編目 sample 資料庫。

您不需要對伺服器工作站上的 sample 資料庫進行編目，因爲當您建立時，即已進行編目。

編目會以從屬站應用程式想要存取的資料庫名稱，來更新從屬站工作站上的資料庫目錄。 當處理從屬站要求時，資料庫管理程式會使用已編目的名稱，來尋找資料庫，並與它連接。

快速入門 一書提供資料庫編目的一般資訊。本節會提供關於 sample 資料庫編目的特定指示。

若要從遠端從屬站工作站中，將範例資料庫編入目錄中，請輸入：

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

其中 *nodename* 即是伺服器節點的名稱

快速入門 一書說明如何將節點編目，作爲設置通訊協定的一部份您也須先將遠端節點編入目錄中，方可與資料庫連接。

連結

如果您要從執行不同版本的 DB2，或在不同作業系統上執行 DB2 的遠端從屬站，存取伺服器上的 sample 資料庫，則您需要將資料庫公用程式 (包括 DB2 CLI) 連結至 sample 資料庫。

連結作業會建立資料包，於應用程式執行時，供資料庫管理程式用來存取資料庫。可以對前置編譯期間建立的連結檔案，指定 BIND 指令，明確地完成連結作業。

Command Reference 提供了關於連結資料庫公用程式的一般資訊。本節說明將資料庫公用程式連結到 sample 資料庫的特定指示。

隨著您使用的從屬站工作站平台，連結資料庫公用程式的方法有所不同。

在 OS/2 從屬站工作站上：

1. 輸入下列指令，連接 sample 資料庫：

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

   其中 *userid* 及 *password* 是 sample 資料庫所在位置的案例之使用者 ID 及通行碼。

   透過此指令，DB2 自動將公用程式連結到資料庫，因此，使用者不必明確地進行連結。

2. 結束「命令行處理器」，檢查連結訊息檔案 bind.msg，驗證是否順利完成連結。

在 UNIX 從屬站工作站上：

1. 輸入下列指令，連接 sample 資料庫：

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

   其中 *userid* 及 *password* 是 sample 資料庫所在位置的案例之使用者 ID 及通行碼。

2. 經由輸入下列指令，使公用程式與資料庫連結：

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

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

   其中 *BNDPATH* 是連結檔案所在的路徑，如 $HOME/sqllib/bnd，其中 $HOME 是 DB2 案例擁有者的起始目錄

3. 經由檢查連結訊息檔 bind.msg 及 cli.msg，驗證是否已完成連結。

在執行 Windows 32 位元作業系統的從屬站工作站上：

1. 從「開始」功能表中，選取「程式集」。
2. 在「程式」功能表中，選取「IBM DB2」。
3. 在「IBM DB2」功能表中，選取「DB2 指令」視窗。

   此時出現命令視窗。

4. 輸入下列指令，連接 sample 資料庫：

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

   其中 *userid* 及 *password* 是 sample 資料庫所在位置的案例之使用者 ID 及通行碼。

5. 經由輸入下列指令，使公用程式與資料庫連結：

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

   其中 %DB2PATH% 是指安裝 DB2 的路徑。

6. 結束命令視窗，檢查連結訊息檔案 bind.msg，驗證是否成功完成連結。

適用於所有平台

如果您已在遵守 DRDA 的應用系統伺服器上，建立了 sample 資料庫，請指定下列 .lst 檔的其中一個，而不要指定 db2ubind.lst。

ddcsmvs.lst
適用 DB2 for OS/390

ddcsvm.lst
適用於 DB2 for VM

ddcsvse.lst
適用於 DB2 for VSE

ddcs400.lst
適用於 DB2 for AS/400

作業系統的 *快速入門* 一書，提供有關連結資料庫公用程式的一般資訊。

下一步驟

一旦您的環境設置完畢，您就可以開發您的 DB2 應用程式。下列章節討論範例程式，以及告訴您如何編譯、鏈結及執行它們。首先閱讀第45頁的『第3章 開發 DB2 應用程式的一般資訊』，再接著閱讀所要建置的應用程式後面的特定章節。

若使用 Java 來設計程式，請參閱第59頁的『第4章 開發 Java Applet 及應用程式』。

若需用 SQL 程序來設計程式，請參閱第93頁的『第5章 開發 SQL 程序』。

若使用 DB2 API、DB2 CLI 及內含的 SQL 來設計程式，請參閱平台的「建置應用程式」。

若需進一步資訊，請參照下列書籍：

- 若需使用內含的 SQL、JDBC 及 SQLJ 執行應用程式，及使用者定義函數 (UDF) 的應用程式相關資訊，請參閱 *Application Development Guide* 。
- 若需使用 DB2 CLI 或 ODBC 的應用程式相關資訊，請參閱 *CLI Guide and Reference* 。
- 若需 DB2 API 應用程式的相關資訊，請參閱 *Administrative API Reference* 。

第3章 開發 DB2 應用程式的一般資訊

本章中的資訊適用於一個以上的作業系統。大部份的主題適用於 DB2 的主要支援平台。

關於最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

建置和執行 DB2 程式的一般要點

1. 應用程式環境：
   - OS/2：如果您是使用指令檔，而非在 CONFIG.SYS 檔中設定環境變數，則您必須在相同的視窗中建置您的應用程式。
   - UNIX：您必須在設定環境變數的 Shell 中建置及執行 DB2 應用程式。 視您所使用的 Shell 而定，您可以執行 `db2profile` 或 `db2cshrc` 來完成此動作。
   - Windows 32 位元作業系統：您必須在 DB2 命令視窗中建置您的應用程式。關於其它資訊，請參閱第33頁的『第2章 設定』。
2. 若要建置含有內含的 SQL 的 DB2 程式或執行 DB2 程式，必須啓動伺服器上的資料庫管理程式。若要啓動資料庫管理程式，您需要 SYSADM（系統管理）權限。關於 SYSADM 權限的資訊，請參閱 *快速入門*。

   在伺服器上輸入下列指令來啓動資料庫管理程式（若尚未執行此程式）：

   ```
   db2start
   ```
3. 當您要建置生產的應用程式時，在可執行檔內所建置的 DB2 執行時間路徑應是安裝路徑，而不是開發應用程式的本端 DB2 案例的路徑。 本書的設計目的在於告訴您如何在開發環境中建置應用程式，因此，附有 UNIX 中 `sqllib/include` 及 `sqllib/lib` 的案例副本，及 OS/2 與 Windows 32 位元作業系統中 `%DB2PATH%\include` 及 `%DB2PATH%\lib` 案例副本的說明。

4. 變更或建置範例程式之前，建議將您要使用的語言範例從 UNIX 上的 sqllib/samples，或從 OS/2 或 Windows 32 位元作業系統上的 %DB2PATH%\samples 複製到自己的工作目錄。這樣可讓您保留原始範例，供未來參照之用。

建置檔、make 檔和錯誤檢查公用程式

DB2 提供一建置工具陣列，以滿足程式開發的需求。這些工具使您容易建置提供的範例程式，這些範例程式示範範圍廣泛的 DB2 功能。這些工具也可讓您建置自己的資料庫程式。就每一個支援的編譯器而言，DB2 提供建置檔、make 檔和錯誤檢查公用程式，在範例目錄中有提供它們和範例程式。本節解譯如何使用這些工具。

建置檔

下列各章使用一些含有編譯和鏈結指令的檔案，以便使用支援的平台編譯器建置程式。這些檔案在 OS/2 上稱爲指令檔，在 UNIX 上稱爲 Script 檔，在 Windows 32 位元作業系統上稱爲批次檔。我們一般稱這些檔案爲建置檔。

DB2 在支援的平台上提供各種語言的建置檔，以便使用其中已建置的各類型程式，每一種語言的範例程式都在相同的目錄中。表16 列出在所有支援平台上各種語言的建置檔，但在 Windows 32 位元作業系統上的 C++ 除外。檔案的副檔名均已被省略。若是 OS/2，副檔名是 .cmd；若是 Windows 32 位元作業系統，副檔名是 .bat。若是 UNIX 平台，則沒有副檔名。

對於 Windows 32 位元作業系統，有兩個支援的 C++ 編譯器：Microsoft Visual C++ 及 IBM VisualAge C++。對於這些編譯器，在每一個建置檔名稱的 "bld" 後會分別插入 "m" 或 "v"，但 bldclisp 的名稱除外，會變成 bldmclis 或 bldvclis。這些建置檔均列示在第47頁的表17中。.bat 副檔名已被省略。

表 16. DB2 建置檔

建置檔	建置的程式類型
bldsqlj	Java 內含的 SQLJ 應用程式。
bldsqljs	Java 內含的 SQLJ 儲存程序。
bldcli	DB2 CLI 應用程式，附有或沒有內含的 SQL。
bldapi	DB2 CLI 應用程式，附有或沒有內含的 SQL，需要鏈結內含 DB2 API 的 utilapi 公用程式檔，以建立及捨棄資料庫。
bldclisp	DB2 CLI 儲存程序 (非內含的 SQL)。
bldapp	應用程式，附有或沒有內含的 SQL。
bldsrv	內含的 SQL 儲存程序。

表 16. DB2 建置檔 (繼續)

建置檔	建置的程式類型
bldudf	使用者定義函數 (UDF)。
bldmt	多緒內含的 SQL 應用程式 (僅適用於受支援 UNIX 平台中的 C/C++)。
bldevm	事件監督程式範例程式 `evm` (僅適用於 AIX、OS/2 及 Windows 32 位元作業系統)。

表 17. Windows 32 位元作業系統的 C/C++ 建置檔

建置檔	建置的程式類型
bldmcli	Microsoft Visual C++ DB2 CLI 應用程式，附有或沒有內含的 SQL。
bldvcli	VisualAge C++ DB2 CLI 應用程式，附有或沒有內含的 SQL。
bldmapi	Microsoft Visual C++ DB2 CLI 應用程式，附有或沒有內含的 SQL，需要鏈結內含 DB2 API 的 `utilapi` 公用程式檔，以建立及捨棄資料庫。
bldvapi	VisualAge C++ DB2 CLI 應用程式，附有或沒有內含的 SQL，需要鏈結內含 DB2 API 的 `utilapi` 公用程式檔，以建立或捨棄資料庫。
bldmclis	Microsoft Visual C++ DB2 CLI 儲存程序 (非內含的 SQL)。
bldvclis	VisualAge C++ DB2 CLI 儲存程序 (非內含的 SQL)。
bldmapp	Microsoft Visual C++ 應用程式，附有或沒有內含的 SQL。
bldvapp	VisualAge C++ 應用程式，附有或沒有內含的 SQL。
bldmsrv	Microsoft Visual C++ 內含的 SQL 儲存程序。
bldvsrv	VisualAge C++ 內含的 SQL 儲存程序。
bldmudf	Microsoft Visual C++ 使用者定義函數 (UDF)。
bldvudf	VisualAge C++ 使用者定義函數 (UDF)。
bldmevm	事件監督程式範例程式 evm，具有 Microsoft Visual C++ 編譯器。
bldvevm	事件監督程式範例程式 evm，具有 VisualAge C++ 編譯器。

本書記錄這些建置檔的原因，是因爲它們非常清楚地示範 DB2 建議的編譯和鏈結選項，以便使用支援的編譯器來建置不同種類的程式。通常有許多其它編譯和鏈結選項，使用者可自由地試驗它們。請參閱提供的編譯和鏈結選項的編譯器文件。除了建置和範例程式以外，軟體開發者也可使用建置檔建置自己的程式。使用者可修改範例程式，將之作爲模版以協助他們的程式設計開發。

爲了方便起見，建置檔是設計成使用編譯器容許的檔名來建置來源檔。這方面不像 make 檔，make 檔會將程式名稱寫到檔案內。建置檔在 UNIX 上使用 $1 變數，在 OS/2 和 Windows 32 位元作業系統上使用 %1 變數，替代內部的程式名稱。其它類似命名的變數替代可能需要的其它引數。建置檔能夠快速而容易地試驗，因爲每一個建置檔都適合特定種類的程式建置，例如 DB2 API、DB2 CLI、內含的 SQL、儲存程序或使用者定義的函數。只要編譯器支援某類型建置檔要建置的特定類型程式就會提供該種類建置檔。

由建置檔所產生的物件及可執行檔會在每一次程式建置時自動改寫，即使來源檔沒有修改也一樣。這不像 makefile。它這表示軟體開發者可以重建現存的程式，而不用刪除之前的物件及可執行檔，或修改來源檔。

建置檔含有範例資料庫的預設設定。若使用者正在存取另一個資料庫，則只要提供另一個參數就可改寫預設值。若他們持續使用其它資料庫，他們可能需要將這個資料庫名稱（取代 sample）寫在建置檔本身之內。

內含的 SQL 程式所使用的建置檔會呼叫另一個檔案 embprep，該檔案中含有內含的 SQL 程式的前置編譯及連結步驟。這些步驟視內含的 SQL 程式的建置位置而定也許需要選用的參數使用者 ID 及通行碼。

如果軟體開發者是在資料庫所在的伺服器案例中建置程式，則使用者 ID 及通行碼，則對兩者而言是共同的，也就不需要提供。換句話說，如果軟體開發者是在不同的案例中，如從遠端存取伺服器資料庫的從屬站機器，則必須提供這些參數。 embprep 檔亦供 make 檔使用。請參閱下面的 『make 檔』，取得此檔案的其餘相關資訊。

最後軟體開發者可依需要修改建置檔。除了在建置檔中變更資料庫名稱以外（如上述），軟體開發者可輕易將其它參數寫在檔案內，變更編譯和鏈結選項，或變更預設 DB2 案例路徑。建置檔的簡單、直接及特定的本質使您能夠輕易地依需要修改它們。

make 檔

支援的編譯器的每一個範例目錄，包含一個用來建置提供的範例程式的 make 檔。make 檔會建置大部份編譯器所隨附的 DB2 範例程式。對於每一個範例程式編譯時使用的許多一般元素，它都使用變數。 make 檔的語法和其指令的輸出，在某些重要方面與提供的建置檔不同。 make 指令容易使用而且功能強：

make <program_name>
: 編譯和鏈結指定的程式。

make all
: 編譯和鏈結列示在 make 檔的所有程式。

make clean

刪除列示在 make 檔的全部程式的全部中間檔（如目的檔）。

make cleanall

刪除列示在 make 檔的全部程式的全部中間檔和可執行檔。

不像建置檔，make 檔不會改寫它列示之程式的現存中間檔和可執行檔。使用 make all 指令可更快建立部份這些檔案的可執行檔（若其它檔案已有可執行檔），因爲 make all 會忽略這些檔案。但它也需要 make clean 及 make cleanall 指令，以便在不需要現存目的檔和可執行檔時除去它們。

make 檔可用於程式開發。make 檔比建置檔不容易使用，但如果您要的是 make 指令的能力與便利的話，那麼不妨考慮使用 make 檔。要將新程式加入現存 make 檔，請使用類似現存範例程式作爲模板來編寫程式登錄的語法。請注意：此語法與 DB2 API、DB2 CLI 和內含的 SQL 程式以及儲存程序和 UDF 不同。

以下是如何使用提供的 make 檔的範例。這個 make 檔（來自 AIX 上的 samples/cli 目錄）使用 IBM C 編譯器，在 AIX 上建置全部提供的 DB2 CLI 範例。本範例將示範如何將儲存程序程式 spserver 建置到從屬站應用程式 spclient 呼叫的共用檔案庫。

使用 makefile 之前，您必須編輯內含於此檔案的下列變數：

DB 使用的資料庫。預設是設定爲 sample。

UID 使用的使用者 ID。預設是不設定任何值。

PWD 代表 UID 使用者 ID 的通行碼。預設是不設定任何值。

在 AIX 上，DB2 CLI makefile 會定義變數 DB2PATH、 CC、COPY、ERASE、 CFLAGS 及 LIBS，如下所示：

```
   # Set DB2PATH to where DB2 will be accessed.
   # The default is the instance path.
   DB2PATH=$(HOME)/sqllib

   CC = cc
COPY = cp
ERASE = rm -f

   # The required compiler flags
   CFLAGS= -I$(DB2PATH)/include

   # The required libraries
   LIBS= -L$(DB2PATH)/lib -Wl,-rpath,$(DB2PATH)/lib -ldb2
```

makefile 會在編譯儲存程序 spserver 時使用這些變數，並將共用檔案庫複製到函數次目錄：

```
spserver : utilcli.o
        $(CC) -o spserver spserver.c utilcli.o $(CFLAGS) $(LIBS) \
        -H512 -T512 -bE:spserver.exp -e outlanguage
        $(ERASE) $(DB2PATH)/function/spserver
        $(COPY) spserver $(DB2PATH)/function/spserver
```

對於內含的 SQL 程式，make 檔會呼叫 embprep 檔，其中含有這些內含的 SQL 程式的前置編譯及連結步驟。使該檔案成爲個別的檔案，供每一個內含的 SQL 程式呼叫，以省去在 make 檔內重複這些步驟的手續。此檔案會使用使用者 ID、通行碼及資料庫名稱的參數，連接伺服器上的資料庫。 makefile 會在呼叫 embprep 時，將這些值傳送給它。

資料庫變數 DB 依預設是寫在 sample 資料庫內，如果使用其它資料庫的話，使用者可以變更該變數。使用者 ID 和通行碼變數UID 及 PWD，依預設不設定爲任何值。若使用者已經在與伺服器資料庫相同的案例中工作的話，便不需要使用這些選用性參數。然而，如果不是這種情況，例如，如果使用者從從屬站機器遠端連接伺服器，則使用者可以修改 makefile，方法爲提供適當的值給 UID 及 PWD 變數，然後這些值會自動傳送到 embprep 前置編譯及連結檔案。下列是在 Windows NT 中 Micro Focus COBOL make 檔呼叫 embprep 檔的範例，以建置內含的 SQL 應用程式 updat：

```
updat.cbl : updat.sqb
        embprep updat $(DB) $(UID) $(PWD)
updat.obj : updat.cbl
        $(CC) updat.cbl ;
updat : updat.obj checkerr.obj
        $(LINK) updat.obj checkerr.obj $(LIBS)
```

錯誤檢查公用程式

DB2 AD Client 提供數個公用程式檔。這些檔案有一些函數，可以錯誤檢查及列印錯誤資訊。但有一例外狀況，即 CLI 公用程式檔 utilapi.c，該檔案會呼叫 DB2 API 以建立及捨棄資料庫。在範例目錄中提供每一種語言版本的公用程式檔。在與某一應用程式一起使用時，錯誤檢查公用程式檔可提供有用的錯誤資訊，以便更易於除錯 DB2 程式。大部份的錯誤檢查公用程式會使用 DB2 API GET SQLSTATE MESSAGE 及 GETERROR MESSAGE 以取得適當的 SQLSTATE 及 SQLCA 資訊，這些資訊與程式執行時所發生的問題相關。 DB2 CLI 公用程式檔 utilcli.c 不使用這些 DB2 API；相反地，它使用同等的 DB2 CLI 陳述式。所有錯誤檢查公用程式印出的說明性錯誤訊息，讓軟體開發者能夠迅速瞭解問題。

有些 DB2 程式如儲存程序和使用者定義的函數，不必使用這些公用程式。 Java 也不需要使用這些公用程式，因爲在發生異常狀況時，SQLException 物件會被排除。

以下是 DB2 支援的不同程式設計語言的編譯器，所使用的錯誤檢查公用程式檔：

checkerr.cbl
供 COBOL 程式使用

utilcli.c
供 CLI 程式使用

utilapi.c
供 C 非內含的 SQL 程式使用

utilemb.sqc
供 C 內含的 SQL 程式使用

utilapi.C
供 C++ 非內含的 SQL 程式使用。

utilemb.sqC
供 C++ 內含的 SQL 程式使用

要使用公用程式函數之前，必須先編譯公用程式檔，然後編譯在建立目標程式的可執行檔時鏈結的目的檔。在 samples 目錄中的 makefile 及建置檔，可供需要錯誤檢查公用程式的程式執行上述作業。

下列範例示範如何在 DB2 程式中使用錯誤檢查公用程式。 utilemb.h 標頭檔會定義 EMB_SQL_CHECK 巨集，該巨集可替代函數 SqlInfoPrint() 及 TransRollback()：

```
#define EMB_SQL_CHECK( MSG_STR )           \
   if( SqlInfoPrint( MSG_STR, &sqlca, __LINE__, __FILE__)  != 0 )
TransRollback( );
```

SqlInfoPrint() 會檢查 SQLCODE 旗號。它會將此旗號指示的特定錯誤之任何相關可用資訊列印出來。它也會指向原始程式中發生錯誤的位置。 TransRollback() 可讓公用程式檔安全地在發生錯誤的異動後執行失敗回復。它需要內含的 SQL 陳述式，連接資料庫並執行失敗後回復。下列範例說明 C++ 程式 cursor 如何使用巨集呼叫程式函數，並提供 SqlInfoPrint() 函數的 MSG_STR 參數值 "DECLARE CURSOR"：

```
Cursor::Fetch () {
   EXEC SQL DECLARE c1 CURSOR FOR
            SELECT name, dept FROM staff WHERE job='Mgr'
            FOR UPDATE OF job;
   EMB_SQL_CHECK("DECLARE CURSOR") ;
```

EMB_SQL_CHECK 巨集會確保如果 DECLARE 陳述式失效，則異動仍可安全地失敗後回復，並列印適當的錯誤訊息。

鼓勵軟體開發者在建立自己的 DB2 程式時使用和建置這些錯誤檢查公用程式。

Java Applet 和應用程式

欲建置 Java Applet 及應用程式，您應在所有的平台中遵循相同的步驟，本書中有說明此資訊的章節，即第59頁的『第4章 開發 Java Applet 及應用程式』。本章分成幾節來提供每一個平台需要的特定安裝資訊。除了 DB2 安裝資訊以外，第33頁的『第2章 設定』 還有提供本安裝資訊。

Java 這一章說明如何建置那些使用 JDBC 驅動程式的 JDBC 程式，以及說明如何建置那些除了使用 JDBC 驅動程式以外還使用 Java 支援的內含的 SQL 的 SQLJ 程式。本章說明如何建置 JDBC 和 SQLJ Applet、應用程式和儲存程序。本章也說明如何建置那些無法包含 JDBC 或 SQLJ 陳述式的 Java 使用者定義的函數。

samples 目錄包含一個 Java makefile 和一些 SQLJ 程式的建置檔。因爲在命令行上建置 JDBC 程式很容易，所以沒有爲這些程式提供建置檔。

DB2 API 應用程式

DB2 AD Client 含有呼叫 DB2 API 的範例程式。本書稍後有幾章「建置應用程式」，會就該章中的平台，說明如何使用 DB2 AD Client 隨附的 DB2 應用程式建置檔，建置支援編譯器的範例程式。您也可以使用所提供的 makefile。 make 檔和建置檔顯示可使用的編譯器選項。在適當的章節中，會爲每一個平台的支援編譯器定義這些選項。您可能需要修改選項，以符合您的環境。

本書使用下面範例程式，來示範使用支援的程式設計語言建置和執行 DB2 API 應用程式的步驟。您遵循的步驟會依環境而不同：

client
: 示範下列 API 的使用：SET CLIENT 和 QUERY CLIENT

關於全部 DB2 API 範例程式的詳細說明，請參閱第14頁的『範例程式表格』。

支援的 DB2 API 範例程式的來源檔位於下列目錄的適當程式設計語言次目錄：sqllib/samples (UNIX) 和 %DB2PATH%\samples（ OS/2 和 Windows 32 位元作業系統）。

建置範例程式之後，可用它們來作爲模板以來建立您自己的應用程式。您可使用提供的 make 檔或建置檔建置 DB2 API 程式。

註: 在撰寫您的 API 應用程式時，所有 API 輸入結構都應 memset 爲 0，讓未明確設定的結構元素都起始設定爲 0。這在就下一層次版本的 API，重新編譯已編碼的應用程式時，是很重要的。在重新編譯時，會使用結構的新定義，但可能所有元素並未起始設定。 呼叫 memset 可確保所有元素都已起始設定。下列範例會說明輸入資料結構 pLoadInStruct memset 爲 0：

```
memset( pLoadInStruct, 0, sizeof(pLoadInStruct) );
```

DB2 Call Level Interface (CLI) 應用程式

DB2 AD Client 隨附使用 DB2 Call Level Interface (DB2 CLI) 函數呼叫的範例程式。您可研究範例以瞭解如何在應用程式中使用這些函數呼叫來存取 DB2 資料庫。

假設您使用 ODBC SDK（DB2 不含 ODBC SDK）重新編譯此應用程式，並假設在應用程式平台有 ODBC 驅動程式管理程式，便可以將符合 ODBC 的 DB2 CLI 應用程式移轉到 ODBC 之下工作。

範例程式、建置檔和一個 `makefile` 併入 UNIX 上的目錄 `sqllib/samples/cli`，或併入 OS/2 和 Windows 32 位元作業系統上的 `%DB2PATH%\samples\cli`。您可能需要依環境來修改建置檔和 `makefile` 中的編譯器選項。

下列範例是本書中所使用的範例程式，這些範例程式示範建置和執行 DB2 CLI 應用程式的步驟（您遵循的步驟可能因環境而不同）：

`tbinfo` 示範如何在表格層次上取得及設定資訊。

`dbusemx`
示範如何使用單一資料庫連結內含的 SQL。

`dbmconn`
示範如何連接多重資料庫，及如何切斷與資料庫的連接。

`spclient`
是主/從範例的從屬站程式；伺服器程式是 `spserver`。

`spserver`
是主/從範例的伺服器程式；從屬站程式是 `spclient`。

`udfcli` 使用由使用者定義的函數 `udfsrv` 所建立的函數。

關於全部 DB2 CLI 範例程式的更詳細說明，請參閱第24頁的表7。*CLI Guide and Reference* 說明如何使用 DB2 CLI 運作的範例。

內含的 SQL 應用程式

註: 此處不討論內含的 SQL for Java (SQLJ)，但在第59頁的『第4章 開發 Java Applet 及應用程式』有完整討論。

DB2 AD Client 包括了一些範例程式，其中內含 SQL 陳述式。本書稍後有幾章「建置應用程式」，會就該章中的平台，說明如何使用 DB2 AD Client 隨附的建

置檔，建置支援編譯器的範例程式。您也可以使用所提供的 makefile。 make 檔和建置檔顯示可使用的編譯器選項。在適當的章節中，會爲每一個平台的支援編譯器定義這些選項。您可能需要修改選項，以符合您的環境。

執行某建置檔以建置有內含的 SQL 的範例程式時，該建置檔會執行下列步驟：
- 與資料庫連接。
- 前置編譯您的來源檔。
- 使您的連結檔與資料庫連結。
- 切斷與資料庫的連接。
- 編譯及鏈結您的來源檔。

下列是本書中所使用的範例程式，示範使用支援的程式設計語言來建置和執行內含 SQL 應用程式的步驟。您遵循的步驟會依環境而不同：

updat 使用靜態 SQL 來更新資料庫。

下列範例是用來示範使用 C 及 C++ 所建置的儲存程序及使用者定義函數 (UDF) 的內含 SQL 從屬站應用程式：

spclient
是從屬站程式，示範呼叫儲存程序；伺服器程式是 spserver。

spserver
是伺服器程式，示範儲存程序；從屬站程式是 spclient。

udfcli 使用使用者定義的函數檔案庫 udfsrv 中的 ScalarUDF 函數。

下列範例是用來示範使用 COBOL 所建置的儲存程序及使用者定義函數 (UDF)：

outcli 是從屬站程式，示範呼叫儲存程序；伺服器程式是 outsrv。

outsrv 是伺服器程式，示範儲存程序；從屬站程式是 outcli。

calludf
呼叫使用者定義的函數檔案庫 udf 中的函數。

關於這些範例程式的詳細說明，請參閱第14頁的『範例程式表格』。

支援的這些範例程式的來源檔位於下列目錄的適當程式設計語言次目錄： UNIX 上的 sqllib/samples，以及 OS/2 和 Windows 32 位元作業系統上的 %DB2PATH%\samples。

建置範例程式之後，可用它們來作爲模板以來建立您自己的應用程式。可使用您自己的 SQL 陳述式，來修改範例程式，而做到這一點。您可使用提供的 makefile 或建置檔建置程式。

第13頁的『範例程式』列出所有範例程式。 *Application Development Guide* 說明有內含的 SQL 之範例如何運作。

儲序程序

儲存程序已建置並儲存在伺服器上。可以使用從屬站應用程式遠端存取這些儲存程序。然後儲存程序會在伺服器資料庫中執行本端處理程序，並將結果傳回從屬站。這可減少網路壅塞，並增進整體效能。

儲存程序可在隔離或非隔離狀況下執行。未隔離的儲存程序與資料庫管理程式在相同的位址空間上執行；隔離的儲存程序則在不同於資料庫管理程式的位址空間上執行。相較之下，前者的效能較高。使用非隔離的儲存程序，必須承擔使用者程式碼會損壞資料庫控制結構的風險。因此您應該只在需要最大效能時，才執行非隔離的儲存程序。在以未被隔離方式執行這些程式之前，請先確定已徹底測試過它們。請參閱 *Application Development Guide*，取得詳細資訊。

本書中所示範的儲存程序均儲存在伺服器的路徑 `sqllib/function` 下。對於呼叫的程序符合共用檔案庫名稱的 DB2DARI 參數樣式儲存程序而言，此位置表示已隔離儲存程序。如果不要隔離此種類型的儲存程序，您需要將它移至 `sqllib/function/unfenced` 目錄。至於其它所有類型的 DB2 儲存程序，您需要在呼叫程式中使用 CREATE FUNCTION 陳述式，指出是否被隔離。關於建立和使用不同類型的 DB2 儲存程序的完整討論，請參閱 *Application Development Guide* 中的「儲存程序」一章。

下列範例程式是用來示範在伺服器上使用「SQL 程序」建置及儲存儲存程序檔案庫的步驟：

spserver.db2
: 是一個 CLP script，含有用來在伺服器中建立共用檔案庫的「SQL 程序」。它可以使用 CLP `call` 指令、或以 C、C++ 及 CLI 目錄中的 `spclient` 應用程式來呼叫。

下列範例程式是用來示範使用 C 及 C++ 在伺服器上建置及儲存儲存程序檔案庫的步驟：

spserver
: 是主/從範例的伺服器程式；從屬站程式是 `spclient`。

下列範例程式是用來示範使用 Java 在伺服器上建置及儲存儲存程序檔案庫的步驟：

Spserver
: 是主/從範例的伺服器程式；從屬站程式是 `Spclient`。

下列範例程式是用來示範使用 COBOL 在伺服器上建置及儲存儲存程序檔案庫的步驟：

outsrv 是主/從範例的伺服器程式；從屬站程式是 outcli。

使用者定義的函數 (UDF)

使用者定義的函數可讓您依需要撰寫自己的 SQL 擴充。像儲存程序，使用者定義的函數儲存在伺服器上供從屬站應用程式存取。 UDFs 不含內含的 SQL 陳述式。

本書使用下列範例程式示範在伺服器上建置和儲存 UDF 檔案庫的步驟：

udfsrv 建立使用者定義函數 (UDF) 的檔案庫。僅供 C 及 C++ 使用。從屬站應用程式 udfcli 會呼叫這些函數。

udf 建立使用者定義函數 (UDF) 的檔案庫。僅供 COBOL 使用。從屬站應用程式 calludf 會呼叫這些函數。

UDFsrv 建立使用者定義函數 (UDF) 的檔案庫。僅供 Java 使用。 UDFcli 及 UDFclie 分別是 JDBC 及 SQLJ 從屬站應用程式，會呼叫這些函數。

多緒的應用程式

在支援的 UNIX 平台上，DB2 支援 C 及 C++ 多緒應用程式。這些應用程式可讓使用者有數個同時處理程序，同時執行數個緒的操作。這樣可以處理非同步事件，並建立事件導向應用程式，而無需重新排序以輪詢綱目。下列範例程式是用來示範 DB2 多緒應用程式的建置：

thdsrver
示範緒的建立及管理。

UDF 及儲存程序的 C++ 考慮事項

在 C++ 中函數名稱可能會 '超載'。若同名的兩個函數有不同引數，則這兩個函數可共存，如下列指令：

```
int func( int i )
```

及

```
int func( char c )
```

依預設，C++ 編譯器會將函數名稱 '類型裝飾' 或 '識別編碼'。這表示該引數類型名稱會附加至它們的函數名稱進行解析，與先前兩個範例的 func__Fi 和 func__Fc 一樣。在每一個平台上識別編碼的名稱會不同，所以無法移轉明確使用識別編碼名稱的程式碼。

在 OS/2 及 Windows 32 位元作業系統上，type-decorated 函數名稱可以在 .obj (物件) 檔中決定。

在 OS/2 及 Windows 上使用 VisualAge C++ 編譯器，便可以使用 cppfilt 指令從 .obj (物件) 檔中決定 type-decorated 函數名稱，如下所示：

```
cppfilt -b /p myprog.obj
```

其中 myprog.obj 是您的程式物件檔。

在 Windows 上使用 Microsoft Visual C++ 編譯器，便可以使用 dumpbin 指令在 .obj (物件) 檔中決定 type-decorated 函數名稱，如下所示：

```
dumpbin /symbols myprog.obj
```

其中 myprog.obj 是您的程式物件檔。

在 UNIX 平台上，可以使用 nm 指令，從 .o (物件) 檔或從共用檔案庫，決定 type-decorated 函數名稱。本指令會產生大量輸出，所以建議透過 grep pipe 此輸出，以尋找正確行，如下所示：

```
nm myprog.o | grep myfunc
```

其中 myprog.o 是程式目的檔，myfunc 是程式來源檔中的函數。

所有這些指令產生的輸出中，有一行附有已識別編碼的函數名稱。例如，在 UNIX 上，此行類似下列：

```
myfunc__FPlT1PsT3PcN35|     3792|unamex|          | ...
```

一旦您已由上述指令之一取得已識別編碼的函數名稱，便可以在適當的指令中使用該名稱。下面即說明如何使用自上述 UNIX 範例中取得之已識別編碼的函數名稱，執行作業。在 OS/2 或 Windows 上取得之已識別編碼的函數名稱，其使用方法是一樣的。

在使用 CREATE FUNCTION 登記 UDF 時，EXTERNAL NAME 子句必須指定已識別編碼的函數名稱：

```
CREATE FUNCTION myfunco(...) RETURNS...
       ...
       EXTERNAL NAME '/whatever/path/myprog!myfunc__FPlT1PsT3PcN35'
       ...
```

同樣地，在呼叫儲存程序時，CALL 函數也必須指定已識別編碼的函數名稱：

```
CALL 'myprog!myfunc__FPlT1PsT3PcN35' ( ... )
```

如果您的儲存程序或 UDF 檔案庫未含有超載的 C++ 函數名稱，您可以使用 extern "C" 的選項，來迫使編譯器將它視爲不使用類型裝飾的函數名稱。 (請注

意，您可以一直超載指定給 UDF 的 SQL 函數名稱，因爲 DB2 會依據它所採用的名稱及參數，來解析將呼叫的檔案庫函數。)

```
#include <string.h>
#include <stdlib.h>
#include "sqludf.h"

/*------------------------------------------------------------------------*/
/* function fold: output = input string is folded at point indicated      */
/*                         by the second argument.                        */
/*         inputs: CLOB,                  input string                    */
/*                 LONG                   position to fold on             */
/*         output: CLOB                   folded string                   */
/*------------------------------------------------------------------------*/
extern "C" void fold(
      SQLUDF_CLOB       *in1,                    /* input CLOB to fold       */
   ...
   ...
}
/* end of UDF: fold */

/*------------------------------------------------------------------------*/
/* function find_vowel:                                                   */
/*             returns the position of the first vowel.                   */
/*             returns error if no vowel.                                 */
/*             defined as NOT NULL CALL                                   */
/*         inputs: VARCHAR(500)                                           */
/*         output: INTEGER                                                */
/*------------------------------------------------------------------------*/
extern "C" void findvwl(
      SQLUDF_VARCHAR    *in,                     /* input smallint           */
   ...
   ...
}
/* end of UDF: findvwl */
```

在這個例子中，編譯器不會以類型裝飾 UDF `fold` 及 `findvwl`，而且應該使用它們的純名稱，將它們登記在 CREATE FUNCTION 陳述式中。同樣地，如果以 `extern "C"` 編寫 C++ 儲存程序，將在 CALL 陳述式中使用它的未裝飾函數名稱。

第4章 開發 Java Applet 及應用程式

您可以使用適合平台的 Java Development Kit (JDK) 開發 Java 程式，以存取 DB2 資料庫。 JDK 包括 Java Database Connectivity (JDBC)，這是一種動態 SQL API for Java。

在 DB2 從屬站和伺服器上，所提供的 DB2 JDBC 支援是作爲 Java Enablement 選項的一部份。有了這項支援，您可以建置和執行 JDBC 應用程式和 Applet。這些只含有動態 SQL，並使用 Java 呼叫介面來將 SQL 陳述式傳送到 DB2。

所提供的 DB2 Java 內含的 SQL (SQLJ) 支援是作爲 DB2 AD Client 的一部份。除了 DB2 JDBC 支援外，有了 DB2 SQLJ 支援之後，您可建置和執行 SQLJ Applet 和應用程式。其中含有靜態 SQL，並使用連結到 DB2 資料庫的內含 SQL 陳述式。

DB2 AD Client 提供的 SQLJ 支援包括：

- DB2 SQLJ 轉換程式 `sqlj`，以 Java 原始陳述式取代 SQLJ 程式中的內含的 SQL 陳述式，並產生序列設定檔，此設定檔含有關於 SQLJ 程式中的 SQL 作業之資訊。
- DB2 SQLJ 設定檔自訂程式 `db2profc`，它前置編譯儲存在序列設定檔中的 SQL 陳述式，自行設定它們成爲執行函數呼叫，以及在 DB2 資料庫中產生資料包。關於 `db2profc` 指令的詳細資訊，請參閱 *Command Reference*。
- DB2 SQLJ 設定檔印表機 `db2profp`，它以純文字方式列印 DB2 自訂版的設定檔內容。

欲執行 DB2 Java 應用程式，您必須安裝及呼叫提供原生緒支援的 Java Virtual Machine (JVM)。欲執行使用原生緒的 Java 應用程式，您可以在指令中使用 `-native` 選項。例如，欲執行 Java 範例應用程式 `App.class`，您可以使用下列指令：

```
java -native App
```

您也可以指定原生緒作爲某些 Java Virtual Machine 的預設緒支援。本章中的資訊均假設原生緒支援爲預設值。請參閱您的 JVM 文件，取得在系統中設定原生緒爲預設值的相關指示。

欲執行 DB2 Java Applet，您可以呼叫提供原生緒或綠色緒支援的 Java Virtual Machine。

關於以 Java 程式設計 DB2 的其它資訊，請參閱 *Application Development Guide* 的 ″Programming in Java″ 這一章。

關於最新的 DB2 Java 更新資訊，請造訪下列網頁：

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

設定環境

如果您正在受支援的平台上使用 IBM JDK 1.1.8 來開發 SQLJ 程式，則需要 JDK 的開發日期爲 1999 年 11 月 24 日 (或之後)。否則，在編譯期間可能會遇到 JNI 錯亂 (panic) 錯誤。

如果您正在受支援的平台上使用 IBM JDK 1.2 來開發 SQLJ 程式，則需要 JDK 的開發日期爲 2000 年 4 月 17 日 (或之後)。否則，在編譯期間可能會遇到「無效 Java」類型錯誤。

若要測試您的 JDBC 環境，您可以使用 sqllib\samples\java (Windows) 或 sqllib/samples/java (UNIX) 中的範例檔案 db2JDBCVersion.java。db2JDBCVersion 程式檢查目前使用之 DB2 JDBC 驅動程式的版本，以及 JDBC 環境是否已爲其正確設定。此程式不可用於 OS/2。

AIX

欲在具有 DB2 JDBC 支援的 AIX 上開發 Java 應用程式，您需要在您的開發機器上安裝及架構下列項目：

1. 下列其中一項：
   - 對於 AIX 4.2.1 與 AIX 4.3.3 及更新版本：IBM 推出的 Java Development Kit (JDK) 版本 1.1.8 及 Java Runtime Environment (JRE) 版本 1.1.8 for AIX (與 DB2 一同安裝)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。
   - 對於 AIX 4.3.3 或更新版本：IBM 推出的 Java Development Kit (JDK) 版本 1.2 及 Java Runtime Environment (JRE) 版本 1.2 for AIX。需要 JDK 的開發日期爲 2000 年 4 月 17 日或之後。

   (請參照 http://www.ibm.com/software/data/db2/java。)
2. DB2 Java Enablement，隨附於適用 AIX 從屬站及伺服器的 DB2 Universal Database 版本 7 中。

若要執行 DB2 Java 儲存程序或 UDF，您必須在伺服器上更新 DB2 資料庫管理程式架構，以便併入 JDK 安裝在該機器上的路徑。您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk11
```

其中 /home/db2inst/jdk11 是安裝 JDK 的路徑。

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，請輸入：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk12
```

其中 /home/db2inst/jdk12 是安裝 JDK 1.2 的路徑。

註: 實際上，JDK11_PATH 用於指出任何層次 JDK 的位置。JDK 是 1.1 還是 1.2 由 DB2 登錄變數 DB2_USE_JDK12 的設定控制。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

爲了在具有 DB2 JDBC 支援的 AIX 上執行 JDBC 和 SQLJ 程式，所以在資料庫管理程式檔 db2profile 和 db2cshrc 中有包括用來更新 AIX Java 環境的指令。建立 DB2 案例之後，修改 .profile 及/或 .cshrc，使得 CLASSPATH 包括：

- "." （現行目錄）
- 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

使用帶有 Java 應用程式的 JDBC 2.0 驅動程式

JDBC 1.2 驅動程式在所有的作業系統 (其中可用 JDBC 1.2 驅動程式) 中仍是預設驅動程式。 欲利用 JDBC 2.0 的新功能，您必須安裝 JDK 1.2 支援。執行利用 JDBC 2.0 新功能的應用程式之前，您必須藉由從 sqllib/java12 目錄發出 usejdbc2 指令來設定環境。如果想要您的應用程式恆使用 JDBC 2.0 驅動程式，請考慮執行下列動作：

- 若爲 Bash 或 Korn Shell，請將下列一行新增到您的登入設定檔中 (如 .profile) 或您的 Shell 起始設定 Script (如 .bashrc 或 .kshrc)：

  ```
  . sqllib/java12/usejdbc2
  ```

- 若爲 C Shell，請將下列一行新增到您的 .cshrc Script 中：

  ```
  source sqllib/java12/usejdbc2.csh
  ```

因爲 usejdbc2 及 usejdbc2.csh Script 使用 db2profile 環境設定，所以請確定此指令放置在執行 db2profile 的指令之後。

欲切換回 JDBC 1.2 驅動程式，請從 sqllib/java12 目錄執行下列指令：

- 若爲 bash 或 korn shell：

  ```
  . usejdbc1
  ```

- 若爲 C shell：

  ```
  source usejdbc1.csh
  ```

使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，您必須爲您案例的隔離使用者 ID 設定環境。預設隔離使用者 ID 是 db2fenc1。若要設定隔離使用者 ID 的環境，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=1
```

欲切換回 JDBC 1.2 驅動程式支援，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=
```

HP-UX

欲在具有 DB2 JDBC 支援的 HP-UX 上開發 Java 應用程式，您需要在您的開發機器上安裝及架構下列項目：

1. 下列其中一項：
   - Hewlett-Packard 推出的 HP-UX Developer's Kit for Java 版次 1.1.8 或更新的版本
   - Hewlett-Packard 推出的 HP-UX Developer's Kit for Java 版次 1.2.2

   (請參照 http://www.ibm.com/software/data/db2/java)
2. DB2 Java Enablement，隨附於適用 HP-UX 從屬站及伺服器的 DB2 Universal Database 版本 7 中。

註: Java UDF 及儲存程序需要 HP-UX Developer's Kit for Java 版次 1.2.2。

若要執行 DB2 Java 儲存程序或 UDF，您也必須更新伺服器上的 DB2 資料庫管理程式架構，納入該機器上安裝 JDK 的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk12
```

其中 /home/db2inst/jdk12 是安裝 JDK 1.2.2 的路徑。

註: 實際上，JDK11_PATH 用於指出任何層次 JDK 的位置。JDK 是 1.1 還是 1.2 由 DB2 登錄變數 DB2_USE_JDK12 的設定控制。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

若要執行 Java2 儲存程序，請確定共用檔案庫路徑與下列內容類似：

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

其中 $JAVADIR 是 Java2 SDK 的位置。

註: 如果您正在使用 HotSpot Java Virtual Machine 並接收到「HotSpot Virtual Machine 錯誤」，則您可以考慮停用 HotSpot JVM。您可以透過下列兩種方法之一來達成此目的：

1. 編輯檔案 sqllib/bin/db2profc 並將下列行：

   ```
   java COM.ibm.db2.sqlj.DB2SQLJInstaller "$@"
   ```

   變更爲：

   ```
   java -classic COM.ibm.db2.sqlj.DB2SQLJInstaller "$@"
   ```

2. 藉由編輯 $JAVADIR/bin/.java_wrapper 的行 103，將 ″vmtype=hotspot″ 變更爲 ″vmtype=classic″，停用作爲預設 JVM 的 HotSpot (這會影響使用 Java 的所有應用程式，而不僅是 DB2)。這樣，如果使用者不想使用 HotSpot JVM，他們將需指定 ″-hotspot″ 作爲 Java 的引數，例如：″java -hotspot <program_name>″。

爲了在具有 DB2 JDBC 支援的 HP-UX 上執行 JDBC 和 SQLJ 程式，資料庫管理程式檔 db2profile 和 db2cshrc 中已包括用來更新 HP-UX Java 環境的指令。建立 DB2 案例之後，修改 .profile 及/或 .cshrc，使得：

1. THREADS_FLAG 設定爲 ″native″。
2. CLASSPATH 包括：
   - ″.″（現行目錄）
   - 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

使用帶有 Java 應用程式的 JDBC 2.0 驅動程式

JDBC 1.2 驅動程式在所有的作業系統 (其中可用 JDBC 1.2 驅動程式) 中仍是預設驅動程式。 欲利用 JDBC 2.0 的新功能，您必須安裝 JDK 1.2 支援。執行利用 JDBC 2.0 新功能的應用程式之前，您必須藉由從 sqllib/java12 目錄發出 usejdbc2 指令來設定環境。如果想要您的應用程式恆使用 JDBC 2.0 驅動程式，請考慮執行下列動作：

- 若爲 Bash 或 Korn Shell，請將下列一行新增到您的登入設定檔中 (如 .profile) 或您的 Shell 起始設定 Script (如 .bashrc 或 .kshrc)：

```
. sqllib/java12/usejdbc2
```

- 若爲 C Shell，請將下列一行新增到您的 .cshrc Script 中：

```
source sqllib/java12/usejdbc2.csh
```

請確定此指令放置在執行 db2profile 的指令之後，原因是 usejdbc2 及 usejdbc2.csh Script 使用 db2profile 環境設定。

欲切換回 JDBC 1.2 驅動程式，請從 sqllib/java12 目錄執行下列指令：

- 若爲 bash 或 korn shell：

```
. usejdbc1
```

- 若爲 C shell：

```
source usejdbc1.csh
```

使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，您必須爲您案例的隔離使用者 ID 設定環境。預設隔離使用者 ID 是 db2fenc1。若要設定隔離使用者 ID 的環境，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=1
```

欲切換回 JDBC 1.2 驅動程式支援，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=
```

Linux

若要在具有 DB2 JDBC 支援的 Linux 上建置 Java 應用程式，您必須在開發機器上安裝和架構下列各項：

1. 下列其中一項：
   - IBM Developer Kit 及 Runtime Environment for Linux 版本 1.1.8 (需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後)。
   - IBM Developer Kit 及 Runtime Environment for Linux 版本 1.2
   - IBM Developer Kit 及 Runtime Environment for Linux 版本 1.3

   (請參閱 http://www.ibm.com/software/data/db2/java)
2. DB2 Java Enablement，隨附於適用 Linux 從屬站及伺服器的 DB2 Universal Database 版本 7 中。

若要執行 DB2 Java 儲存程序或 UDF，您也必須更新伺服器上的 DB2 資料庫管理程式架構，納入該機器上安裝 JDK 的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /usr/local/jdk118
```

其中 /usr/local/jdk118 是安裝 JDK 的路徑。

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，請輸入：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk12
```

其中 /home/db2inst/jdk12 是安裝 JDK 1.2 的路徑。

註: 實際上，JDK11_PATH 用於指出任何層次 JDK 的位置。JDK 是 1.1 還是 1.2 由 DB2 登錄變數 DB2_USE_JDK12 的設定控制。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

若要執行 Java 儲存程序或使用者定義的函數，則 Linux 執行時間鏈結器必須能夠存取某些 Java 共用程式庫。您可以將 Java 共用程式庫的位置新增到 /etc/ld.so.conf，或在 /usr/lib 目錄中建立到程式庫的符號鏈結。

如果您決定將 Java 共用程式庫的位置新增到 /etc/ld.so.conf，您必須以 root 身分執行下列指令，復新執行時間鏈結器快取記憶體：

```
bash# ldconfig
```

如果您決定在 /usr/lib 中建立程式庫的符號鏈結，則鏈結的程式庫清單對於 IBM Developer Kit for Java 的不同版本而不同。

若爲 IBM Developer Kit for Java 的版本 1.1.8，需要符號鏈結指向：

```
libjava.so
libjitc.so
libmath.so
libzip.so
```

若爲 IBM Developer Kit for Java 的版本 1.2 或 1.3，需要符號鏈結指向：

```
libjava.so
libjvm.so
libhpi.so
```

爲了在具有 DB2 JDBC 支援的 Linux 中執行 JDBC 及 SQLJ 程式，在資料庫管理程式檔 db2profile 及 db2cshrc 中已併入了更新 Linux Java 環境的指令。建立 DB2 案例之後，修改 .bashrc、.profile 及/或 .cshrc，使得：

1. THREADS_FLAG 設定爲 ″native″。
2. CLASSPATH 包括：
   - ″.″ （現行目錄）
   - 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

使用帶有 Java 應用程式的 JDBC 2.0 驅動程式

JDBC 1.2 驅動程式在所有的作業系統 (其中可用 JDBC 1.2 驅動程式) 中仍是預設驅動程式。 欲利用 JDBC 2.0 的新功能，您必須安裝 JDK 1.2 支援。執行利用 JDBC 2.0 新功能的應用程式之前，您必須藉由從 sqllib/java12 目錄發出 usejdbc2 指令來設定環境。如果想要您的應用程式恆使用 JDBC 2.0 驅動程式，請考慮執行下列動作：

- 若爲 Bash 或 Korn Shell，請將下列一行新增到您的登入設定檔中 (如 .profile) 或您的 Shell 起始設定 Script (如 .bashrc 或 .kshrc)：

  ```
  . sqllib/java12/usejdbc2
  ```

- 若爲 C Shell，請將下列一行新增到您的 .cshrc Script 中：

  ```
  source sqllib/java12/usejdbc2.csh
  ```

請確定此指令放置在執行 db2profile 的指令之後，原因是 usejdbc2 及 usejdbc2.csh Script 使用 db2profile 環境設定。

欲切換回 JDBC 1.2 驅動程式，請從 sqllib/java12 目錄執行下列指令：

- 若爲 bash 或 korn shell：

  ```
  . usejdbc1
  ```

- 若爲 C shell：

  ```
  source usejdbc1.csh
  ```

使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，您必須爲您案例的隔離使用者 ID 設定環境。預設隔離使用者 ID 是 db2fenc1。若要設定隔離使用者 ID 的環境，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=1
```

欲切換回 JDBC 1.2 驅動程式支援，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=
```

OS/2

若要在具有 DB2 JDBC 支援的 OS/2 上建置 Java 應用程式，您必須在開發機器上安裝和架構下列各項：

1. IBM 推出的 OS/2 版 Java Development Kit (JDK) 版本 1.1.8 及 Java Runtime Environment (JRE) 版本 1.1.8 (隨附於 DB2)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。(請參照 http://www.ibm.com/software/data/db2/java。)

   註: 部份訊息不會顯示於執行在 09/99 之前發佈之 JDK 1.1.8 版本的 OS/2 上。請確定您擁有最新的 JDK 版本 1.1.8。

2. DB2 Java Enablement，隨附於適用 OS/2 從屬站及伺服器的 DB2 Universal Database 版本 7 中。

JDK 必須安裝在 HPFS 磁碟機上，以容許長檔名的副檔名大於三個字元，例如 .java。Java 工作目錄也必須在 HPFS 磁碟機。若要在 OS/2 伺服器執行 Java 儲存程序或 UDF，DB2 必須安裝在伺服器的 HPFS 磁碟機，使得儲存程序或 UDF .class 檔能夠置於 sqllib\function 目錄。

若要執行 DB2 Java 儲存程序或 UDF，您必須在伺服器上更新 DB2 資料庫管理程式架構，以便併入 JDK 安裝在該機器上的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH c:\jdk11
```

其中 c:\jdk11 是有安裝 JDK 的路徑。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

若要在具有 DB2 JDBC 支援的 OS/2 中執行 JDBC 及 SQLJ 程式，在安裝 DB2 時，系統會自動更新 CLASSPATH 環境變數以包含：

- "." （現行目錄）
- 檔案 sqllib\java\db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib\java\sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib\java\runtime.zip
```

PTX

若要在具有 DB2 JDBC 支援的 PTX 中建置 Java 應用程式，您必須在您的開發機器中安裝及架構下列各項：

1. ptx/JSE 版本 3.0，相當於 Sun Microsystem 的 JDK 1.2 (請參照 http://www.ibm.com/software/data/db2/java)。
2. DB2 Java Enablement，隨附於適用 NUMA-Q 的 DB2 Universal Database 版本 7 中。

若要更新 DB2 Java 儲存程序或 UDF，您也必須更新伺服器中的 DB2 資料庫管理程式架構，以併入該機器中安裝 ptx/JSE 的路徑。您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /opt/jse3.0
```

其中 /opt/jse3.0 是安裝 ptx/JSE 的路徑。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

爲了在具有 DB2 JDBC 支援的 PTX 中執行 JDBC 及 SQLJ 程式，在資料庫管理程式檔 db2profile 及 db2cshrc 中已併入了更新 PTX Java 環境的指令。建立 DB2 案例之後，修改 .profile 及/或 .cshrc，使得 CLASSPATH 包括：

- "." （現行目錄）
- 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

Silicon Graphics IRIX

若要在具有 DB2 JDBC 支援的 Silicon Graphics IRIX 上建置 Java 應用程式，您必須在開發機器上安裝和架構下列各項：

1. Silicon Graphics 公司推出的 Java2 Software Development Kit 版本 1.2 (JDK 1.2)。 (請參照 `http://www.ibm.com/software/data/db2/java`)。
2. DB2 Java Enablement，隨附於適用 Silicon Graphics IRIX 從屬站的 DB2 Universal Database 版本 7 中。

註: 在 Silicon Graphics IRIX 中的 SQLJ 應用程式僅能以 o32 物件類型建置。欲將 Java 預設物件類型變更爲 o32，請使用下列指令將環境變數 `SGI_ABI` 設定爲具有 Korn shell：

```
export SGI_ABI=-o32
```

並使用下列指令設定爲具有 C shell：

```
setenv SGI_ABI -o32
```

當開發帶有 o32 物件類型的 SQLJ 應用程式，並使用 JDK 1.2 的 Java JIT 編譯器時，如果 SQL 轉換程式因分區段錯誤而失敗，請嘗試使用此指令關閉 JIT 編譯器：

```
export JAVA_COMPILER=NONE
```

在 Silicon Graphics IRIX 上開發 Java SQLJ 程式需要 JKD 1.2。

DB2 for Silicon Graphics IRIX 只適用於從屬站。若要執行 DB2 應用程式和 Applet，以及建置 DB2 內含的 SQL 應用程式和 Applet，您必須從從屬站機器存取伺服器機器上的 DB2 資料庫。伺服器機器會執行另一個作業系統。關於架構從屬站對伺服器通信的資訊，請參閱 *DB2 for UNIX 快速入門*。

另外，由於您將從在不同作業系統上執行的遠端從屬站存取伺服器上的資料庫，所以必須連結資料庫公用程式 (包括 DB2 CLI) 到資料庫。關於詳細資訊，請參閱第42頁的『連結』。

若要執行 DB2 Java 儲存程序或 UDF，您也必須更新伺服器上的 DB2 資料庫管理程式架構，納入該機器上安裝 JDK 的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk11
```

其中 /home/db2inst/jdk11 是安裝 JDK 的路徑。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

爲了在具有 DB2 JDBC 支援的 Silicon Graphics IRIX 中執行 JDBC 及 SQLJ 程式，在資料庫管理程式檔 db2profile 及 db2cshrc 中已併入了更新 Silicon Graphics IRIX Java 環境的指令。建立 DB2 案例之後，修改 .profile 及/或 .cshrc，使得 CLASSPATH 包括：

- "." （現行目錄）
- 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

註: 在 Silicon Graphics IRIX 上，連接環境定義 close() 方法可能會導致設陷。其解決之道，是在垃圾收集期間讓連接環境定義自動結束。

Solaris

欲在具有 DB2 JDBC 支援的 Solaris 作業環境中建置 Java 應用程式，您必須在開發機器中安裝及架構下列各項：

1. Sun Microsystems 推出的 Solaris 版 Java Development Kit (JDK) 版本 1.1.8 或 1.2 (請參照 http://www.ibm.com/software/data/db2/java)。
2. DB2 Java Enablement，隨附於適用 Solaris 從屬站及伺服器的 DB2 Universal Database 版本 7 中。

若要執行 DB2 Java 儲存程序或 UDF，您必須在伺服器上更新 DB2 資料庫管理程式架構，以便併入 JDK 安裝在該機器上的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH /usr/java
```

其中 /usr/java 是安裝 JDK 的路徑。

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，請輸入：

```
db2 update dbm cfg using JDK11_PATH /home/db2inst/jdk12
```

其中 /home/db2inst/jdk12 是安裝 JDK 1.2 的路徑。

註: 實際上，JDK11_PATH 用於指出任何層次 JDK 的位置。JDK 是 1.1 還是 1.2 由 DB2 登錄變數 DB2_USE_JDK12 的設定控制。

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 JDK11_PATH 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

註: 在 Solaris 作業環境中，部份的 Java Virtual Machine 在執行於 ″setuid″ 環境中的程式內，施行的成效不佳。隔離儲存程序及使用者定義的函數程序 db2dari 及 db2udf 通常爲了機密保護而作爲 ″setuid nobody″ 安裝。這表示隔離 Java 使用者定義的函數及儲存程序可能無法啓動，並導致 SQLCODE -4300。

變更 sqllib/adm 中 db2dari 及 db2udf 程式上的許可權是一個可行方案。輸入下列指令會讓 JVM 版本爲隔離 UDF 及儲存程序工作：

```
chmod 0555 db2dari db2udf
```

對於伺服器端動態載入 Java 直譯器，還存在其他問題。即使 jdk11_path 架構參數設定正確，含有 Java 直譯器的共用檔案庫 libjava.so 也可能無法載入。db2diag.log 檔案中的訊息可能在 SQLCODE -4300 或 SQLCODE -4301 錯誤之後指出這一點。使用 ″setuid″ 程式之 dlopen 函數，於執行時間載入共用程式庫的 Solaris 機密保護限制，會產生這個問題。另一種可行方案是，您可爲 /usr/lib 中的必要 JVM 共用檔案庫建立符號鏈結，您可使用類似下列的指令 (依機器上安裝 Java 的路徑而改變)：

```
ln -s /usr/java/lib/*.so /usr/lib
```

爲了具有 DB2 JDBC 支援的 Solaris 作業系統中執行 JDBC 及 SQLJ 程式，在資料庫管理程式檔 db2profile 及 db2cshrc 中已併入了更新 Solaris Java 環境的指令。建立 DB2 案例之後，修改 .profile 及/或 .cshrc，使得：

1. THREADS_FLAG 設定爲 ″native″。
2. CLASSPATH 包括：

- "." （現行目錄）
- 檔案 sqllib/java/db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib/java/runtime.zip
```

使用帶有 Java 應用程式的 JDBC 2.0 驅動程式

JDBC 1.2 驅動程式在所有的作業系統 (其中可用 JDBC 1.2 驅動程式) 中仍是預設驅動程式。 欲利用 JDBC 2.0 的新功能，您必須安裝 JDK 1.2 支援。執行利用 JDBC 2.0 新功能的應用程式之前，您必須藉由從 sqllib/java12 目錄發出 usejdbc2 指令來設定環境。如果想要您的應用程式恆使用 JDBC 2.0 驅動程式，請考慮執行下列動作：

- 若爲 Bash 或 Korn Shell，請將下列一行新增到您的登入設定檔中 (如 .profile) 或您的 Shell 起始設定 Script (如 .bashrc 或 .kshrc)：

  ```
  . sqllib/java12/usejdbc2
  ```

- 若爲 C Shell，請將下列一行新增到您的 .cshrc Script 中：

  ```
  source sqllib/java12/usejdbc2.csh
  ```

請確定此指令放置在執行 db2profile 的指令之後，原因是 usejdbc2 及 usejdbc2.csh Script 使用 db2profile 環境設定。

欲切換回 JDBC 1.2 驅動程式，請從 sqllib/java12 目錄執行下列指令：

- 若爲 bash 或 korn shell：

  ```
  . usejdbc1
  ```

- 若爲 C shell：

  ```
  source usejdbc1.csh
  ```

使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，您必須爲您案例的隔離使用者 ID 設定環境。預設隔離使用者 ID 是 db2fenc1。若要設定隔離使用者 ID 的環境，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=1
```

欲切換回 JDBC 1.2 驅動程式支援，請從 CLP 發出下列指令：

```
db2set DB2_USE_JDK12=
```

Windows 32 位元作業系統

若要在具有 DB2 JDBC 支援的 Windows 32 位元平台上建置 Java 應用程式，您必須在開發機器上安裝和架構下列各項：

1. 下列其中一項：
   - IBM 推出的 Java Development Kit (JDK) 1.1.8 及 Java Runtime Environment (JRE) 1.1.8 for Win32 (選用性地與 DB2 一起安裝)。需要 JDK 的開發日期爲 1999 年 11 月 24 日或之後。
   - Sun Microsystems 推出的 Java Development Kit (JDK) 1.2 for Win32。
   - IBM 推出的 Java Development Kit (JDK) 1.2 for Win32。需要 JDK 的開發日期爲 2000 年 4 月 17 日或之後。
   - Microsoft Software Developer's Kit for Java 版本 3.1。

   (請參照 `http://www.ibm.com/software/data/db2/java`。)
2. DB2 Java Enablement，隨附於適用 Windows 32 位元作業系統從屬站及伺服器的 DB2 Universal Database 版本 7 中。

若要執行 DB2 Java 儲存程序或 UDF，您必須在伺服器上更新 DB2 資料庫管理程式架構，以便併入 JDK 安裝在該機器上的路徑。 您可在伺服器命令行上輸入下列指令來執行此動作：

```
db2 update dbm cfg using JDK11_PATH c:\jdk11
```

其中 `c:\jdk11` 是有安裝 JDK 的路徑。

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，請輸入：

```
db2 update dbm cfg using JDK11_PATH c:\jdk12
```

其中 `c:\jdk12` 是有安裝 JDK 1.2 的路徑。

註: 實際上，JDK11_PATH 用於指出任何層次 JDK 的位置。JDK 是 1.1 還是 1.2 由 DB2 登錄變數 `DB2_USE_JDK12` 的設定控制。

註: 如果安裝 JDK 的路徑之目錄名稱中有一或多個空格，請以單引號括住路徑。例如：

```
db2 update dbm cfg using JDK11_PATH 'c:\Program Files\jdk11'
```

您可在伺服器上輸入下列指令，檢查 DB2 資料庫管理程式架構，以查驗 `JDK11_PATH` 欄位的值是否正確：

```
db2 get dbm cfg
```

您可能需要將輸出重新導向至某檔案以方便檢視。JDK11_PATH 欄位出現在靠近輸出起始處。關於這些指令的詳細資訊，請參閱 *Command Reference*。

欲在具有 DB2 JDBC 支援的 Windows 平台上執行 JDBC 及 SQLJ 程式，在安裝 DB2 時，系統會自動更新 CLASSPATH 以併入：

- ″.″（現行目錄）
- 檔案 sqllib\java\db2java.zip

若要建置 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib\java\sqlj.zip
```

若要執行 SQLJ 程式，則也會更新 CLASSPATH 來包括下面檔案：

```
sqllib\java\runtime.zip
```

若指定哪一個 Java Development Kit 要用於 DB2 SQLJ，DB2 會在 Windows 32 位元作業系統安裝環境變數 DB2JVIEW。它適用於全部 DB2 SQLJ 指令（db2profc、db2profp、profdb、profp 和 sqlj ）。

若使用預設值 ″DB2JVIEW=0″，或沒有設定 DB2JVIEW，會使用 Sun JDK；亦即若呼叫 ″profp″，它會當做 ″java sqlj.runtime.profile.util.ProfilePrinter″ 來執行。若 ″DB2JVIEW=1″，會使用 Microsoft SDK for Java；亦即若呼叫 ″profp″，它會當做 ″jview sqlj.runtime.profile.util.ProfilePrinter″ 來執行。

使用帶有 Java 應用程式的 JDBC 2.0 驅動程式

JDBC 1.2 驅動程式在所有的作業系統 (其中可用 JDBC 1.2 驅動程式，包括 Windows 32 位元) 中仍是預設驅動程式。 欲利用 JDBC 2.0 的新功能，您必須安裝適用您平台的 JDBC 2.0 驅動程式及 JDK 1.2 支援。欲安裝 Windows 32 位元作業系統的 JDBC 2.0 驅動程式，請從 sqllib\java12 目錄輸入 usejdbc2 指令。此指令會執行下列作業：

- 建立 1.22 驅動程式檔的 sqllib\java11 目錄
- 備份 JDBC 1.2 驅動程式檔到 sqllib\java11 目錄
- 從 sqllib\java12 目錄複製 JDBC 2.0 驅動程式檔到適當的目錄

欲切換回 JDBC 1.2 驅動程式，請從 sqllib\java12 目錄執行 usejdbc1 批次檔。

執行 usejdbc1 或 usejdbc2 之前，請確定已停止下列服務：

- DB2 JDBC Applet Server
- DB2 JDBC Applet Server - 控制中心
- IBM WS AdminServer (僅 IBM WebSphere 應用程式伺服器可用)

若要確定您已順利設定 JDBC 驅動程式版本，您需確定所有的檔案已無誤地複製到 sqllib\java 及 sqllib\bin 目錄。如果發出 usejdbc1 或 usejdbc2 指令之後獲得下列其中一項訊息：

「存取遭拒。」

或

「因為檔案正在由另一程序使用，所以程序無法存取該檔案。」

其可能的原因是一或多個上述服務仍在執行。跳至「Windows 服務」視窗並停止服務 (如果已啓動)，然後嘗試重新執行 usejdbc1 或 usejdbc2 指令。

如果仍出現錯誤，請發出指令 db2stop force 並再試。如果不見效，請重新啓動您的系統，以確定已停止上述程序，然後重新執行 usejdbc1 或 usejdbc2 指令。

使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式

若要使用帶有 Java 儲存程序及 UDF 的 JDBC 2.0 驅動程式，您必須藉由執行下列步驟來設定環境：

1. 在 sqllib\java12 目錄中發出下列指令：

   ```
   usejdbc2
   ```

2. 從 CLP 發出下列指令：

   ```
   db2set DB2_USE_JDK12=1
   ```

欲切換回 Java UDF 及儲存程序的 JDBC 1.2 驅動程式支援，請執行下列步驟：

1. 在 sqllib\java12 目錄中發出下列指令：

   ```
   usejdbc1
   ```

2. 從 CLP 發出下列指令：

   ```
   db2set DB2_USE_JDK12=
   ```

設定您 IBM JDK 環境的範例批次檔

下列指令可以放置在批次檔中，以使用 IBM Java Development Kit 設定您的 Java 環境。請確定您作出了所有必需的路徑變更來適應您的特定環境。其他受支援的 JDK 可使用類似的指令。

下列指令可讓範例批次檔設定 IBM JDK 1.1.8 環境：

```
set JDKPATH=D:\JAVA\IBMjdk118
set PATH=%JDKPATH%\bin;%PATH%
set CLASSPATH=.;%JDKPATH%\lib\classes.zip;%CLASSPATH%
db2 update dbm cfg using JDK11_PATH %JDKPATH%
```

```
db2set DB2_USE_JDK12=0
db2 terminate
db2stop
db2start
```

下列指令可讓範例批次檔設定 IBM JDK 1.2 環境。set CLASSPATH 指令僅用於解除設定 JDK 1.1.8 (如果已設定)。如果先前未設定 JDK 1.1.8，則此指令將無效：

```
set JDKPATH=D:\JAVA\IBMjdk122
set PATH=%JDKPATH%\bin;%PATH%
set CLASSPATH=%CLASSPATH:D:\JAVA\IBMjdk118\lib\classes.zip;=%
db2 update dbm cfg using JDK11_PATH %JDKPATH%
db2set DB2_USE_JDK12=1
db2 terminate
db2stop
db2start
```

Java 範例程式

DB2 提供範例程式 (用在下列各節中) 來示範建立和執行只用動態 SQL 的 JDBC 程式，以及使用靜態 SQL 的 SQLJ 程式。在 UNIX 平台上，Java 範例位於 sqllib/samples/java。在 OS/2 和 Windows 32 位元作業系統上，這些範例是位於 sqllib\samples\java。範例目錄也包含 README、 makefile 和建置檔。

Java makefile 會建置所有提供的範例程式。它需要相同的 make 可執行程式，Java Development Kits 通常不提供該程式。關於詳細資訊，請參閱 makefile 本文開頭的註解。部份 Java makefile 指令與其它語言不同： make clean 指令會除去所有由 java 編譯器產生的檔案，如 .class 檔。make cleanall 指令除去這些檔案和 SQLJ 轉換程式產生的檔案。

在命令行上很容易就能建置 JDBC 程式，所以沒有包括它們的建置檔。有提供兩個 SQLJ 建置檔：bldsqlj 建置 SQLJ Applet 和應用程式； bldsqljs 建置 SQLJ 儲存程序。第80頁的『SQLJ 程式』這一節有示範這些建置檔。

OS/2

在 OS/2 上，工作目錄必須在 HPFS 磁碟機。在安裝期間，如果 DB2 安裝程式偵測出目標磁碟機是 FAT，則它將兩個檔案 javasamp.exe 及 README 檔案放置在 sqllib\samples\java 目錄中。若要使用 Java 範例程式，請將 javasamp.exe 移到您的 HPFS 工作目錄，並執行此可執行程式。這將 Java 範例程式解壓縮到目錄中。如果安裝程式偵測出目標磁碟機是 HPFS，則它在安裝程序期間，將 Java 範例檔案解壓縮到 sqllib\samples\java 目錄中。

JDBC 程式

Applets

DB2Applt 示範動態 SQL Java Applet，其使用 JDBC Applet (或 ″net″) 驅動程式存取 DB2 資料庫。

透過在命令行上輸入的指令來建置和執行此 Applet：

1. 確定 DB2 機器（伺服器或從屬站）有安裝及執行 web 伺服器。
2. 根據檔案的指示修改 DB2Applt.html 檔。
3. 在 DB2Applt.html 中指定的 TCP/IP 埠上啓動 JDBC Applet 伺服器，例如，若在 DB2Applt.html 中，您指定 param name=port value='6789'，然後您會輸入下列指令：

   ```
   db2jstrt 6789
   ```

   註: 請確定連接字串中的 JDBC 埠號是建議的預設值。如果您能確定號碼與另一埠號不衝突，則可變更此項。不要使用資料庫埠號 ″50000″。
4. 使用此指令編譯 DB2Applt.java 以產生檔案 DB2Applt.class：

   ```
   javac DB2Applt.java
   ```
5. 請確定您的 Web 瀏覽器可以存取您的工作目錄。如果無法存取工作目錄，請將 DB2Applt.class 和 DB2Applt.html 複製到可存取的目錄中。
6. 將 OS/2 或 Windows 32 位元作業系統上的檔案 sqllib\java\db2java.zip，或將 UNIX 上的 sqllib/java/db2java.zip 複製到與 DB2Applt.class 和 DB2Applt.html 相同的目錄。
7. 在從屬站機器上，啓動 Web 瀏覽器（它必須支援 JDK 1.1）並載入 DB2Applt.html。

步驟 (1)、(5) 及 (7) 的另一個方法是在從屬站機器的工作目錄中輸入下列指令，使用附於 Java Development Kit 的 Applet 檢視器來進行作業：

```
Appletviewer DB2Applt.html
```

您也可使用 Java makefile 建置此程式。

應用程式

DB2Appl 示範動態 SQL Java 應用程式，其使用 JDBC 應用程式 (或 ″app″) 驅動程式存取 DB2 資料庫。

透過在命令行上輸入的指令來建置和執行此應用程式：

1. 使用此指令編譯 DB2Appl.java 以產生檔案 DB2Appl.class：

```
javac DB2Appl.java
```

2. 使用此指令對應用程式執行 java 直譯器：

```
java DB2Appl
```

您也可使用 Java makefile 建置此程式。

儲存程序的從屬站應用程式

Spclient 是一個從屬站應用程式，其使用 JDBC 應用程式驅動程式呼叫 Java 儲存程序類別 Spserver。在建置及執行此從屬站應用程式之前，請在伺服器中建置儲存程序類別。請參閱『儲存程序』。

透過在命令行上輸入的指令建置和執行此從屬站程式：

1. 使用下列指令編譯 Spclient.java ，以產生檔案 Spclient.class：

```
javac Spclient.java
```

2. 使用下列指令在從屬站程式上執行 Java 直譯器：

```
java Spclient
```

您也可使用 Java makefile 建置此程式。

使用者定義函數的從屬站應用程式

UDFcli 是從屬站程式，它使用 JDBC 應用程式驅動程式，在使用者定義的函數伺服器程式 UDFsrv 中，呼叫已執行的使用者定義的函數。建置和執行此從屬站應用程式之前，請在伺服器建置使用者定義的函數程式 UDFsrv。請參閱第89頁的『使用者定義的函數 (UDF)』。

透過在命令行上輸入的指令建置和執行此從屬站程式：

1. 使用此指令編譯 UDFcli.java 以產生檔案 UDFcli.class：

```
javac UDFcli.java
```

2. 使用下列指令在從屬站程式上執行 Java 直譯器：

```
java UDFcli
```

您也可使用 Java makefile 建置此程式。

儲存程序

Spserver 示範使用 JDBC 應用程式驅動程式的動態 SQL PARAMETER STYLE JAVA 儲存程序。在伺服器上編譯和儲存儲存程序。從屬站應用程式呼叫時，它們會存取伺服器資料庫並將資訊傳回從屬站應用程式。

欲在命令行上輸入指令，以在伺服器中建置及執行此程式：

1. 請使用下列指令編譯 Spserver.java，以產生檔案 Spserver.class：

   ```
   javac Spserver.java
   ```

2. 複製 Spserver.class 到 OS/2 及 Windows 32 位元作業系統的 sqllib\function 目錄，或到 UNIX 的 sqllib/function 目錄。
3. 下一步，在伺服器上執行 Spcreate.db2 script，將儲存程序編目。首先，連接資料庫：

   ```
   db2 connect to sample
   ```

   如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

   ```
   db2 -td@ -vf Spdrop.db2
   ```

   然後，使用下列指令編目儲存程序：

   ```
   db2 -td@ -vf Spcreate.db2
   ```

4. 然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請將共用檔案庫的檔案模式設爲「執行」，以便 DB2 案例可以存取檔案庫。
5. 編譯並執行 Spclient 從屬站應用程式以存取儲存程序類別。請參閱第79頁的『儲存程序的從屬站應用程式』。

您也可使用 Java makefile 建置此程式。

SQLJ 程式

註: 欲使用 UNIX、OS/2 及 Windows 32 位元作業系統的 IBM Java Development Kit 建置及執行 SQLJ 程式，您必須使用下列適用於您作業系統的指令，關閉 JDK 的 JIT 編譯器：

OS/2 及 Windows:

```
SET JAVA_COMPILER=NONE
```

UNIX: `export JAVA_COMPILER=NONE`

建置檔 bldsqlj 含有一些建置 SQLJ Applet 或應用程式的指令。在 UNIX 上，這是一個 Script 檔。在 OS/2 上，它是指令檔 bldsqlj.cmd，在 Windows 上，它是批次檔 bldsqlj.bat。此指令的內容和批次檔內容相同，而且會先出現本版本，然後出現 UNIX Script 檔。後面的 Applet 和應用程式建置區段會往後參照這些建置檔。

註: 附於 DB2 的 SQLJ 轉換程式將編譯轉換的 .java 檔成爲 .class 檔。因此，本區段中的建置檔不使用 java 編譯器。

在 OS/2 和 Windows 32 位元作業系統的下列建置檔中，第一個參數 %1 指定來源檔名稱。第二個參數 %2 指定您想要與其連接的資料庫的名稱。第三個參數 %3 指定資料庫的使用者 ID，而 %4 則指定通行碼。只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。若沒有提供資料庫名稱，程式會使用預設 sample 資料庫。

```
@echo off
rem bldsqlj -- OS/2 and Windows 32-bit operating systems
rem Builds a Java embedded SQL (SQLJ) program.
rem Usage: bldsqlj prog_name [ db_name [ userid password ]]

if "%1" == "" goto error

rem Translate and compile the SQLJ source file
rem and bind the package to the database.
if "%2" == "" goto case1
if "%3" == "" goto case2
if "%4" == "" goto error
goto case3
:case1
sqlj %1.sqlj
   db2profc -url=jdbc:db2:sample -prepoptions="package using %1" %1_SJProfile0
   goto continue
:case2
   sqlj     -url=jdbc:db2:%2 %1.sqlj
   db2profc -url=jdbc:db2:%2 -prepoptions="package using %1" %1_SJProfile0
   goto continue
:case3
   sqlj     -url=jdbc:db2:%2 -user=%3 -password=%4  %1.sqlj
   db2profc -url=jdbc:db2:%2 -user=%3 -password=%4 -prepoptions="package using %1"
%1_SJProfile0
   goto continue
:continue

goto exit

:error
echo Usage: bldsqlj prog_name [ db_name [ userid password ]]

:exit

@echo on
```

bldsqlj 的轉換程式及前置編譯選項
sqlj SQLJ 轉換程式（也編譯此程式）。
%1.sqlj SQLJ 來源檔。
%1.java 來自 SQLJ 來源檔的已轉換 Java 檔。
db2profc DB2 for Java 設定檔自訂程式。
-url 指定一個 JDBC URL 建立資料庫連接，如 jdbc:db2:sample。
-user 指定使用者 ID（選用性參數）。
-password 指定通行碼（選用性參數）。
-prepoptions 使用字串 "package using %1" 指定資料庫的資料包名稱， %1 是 SQLJ 來源檔名稱。
%1_SJProfile0 指定程式的序列化設定檔。

在下面 UNIX Script 檔中，第一個參數 $1 指定來源檔名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。 資料庫名稱、使用者 ID 及通行碼均爲可選用的。若沒有提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldsqlj script file -- UNIX platforms
# Builds a Java embedded SQL (SQLJ) sample
# Usage: bldsqlj <prog_name> [ <db_name> [ <userid> <password> ]]

# Translate and compile the SQLJ source file
# and bind the package to the database.
if (($# < 2))
then
   sqlj $1.sqlj
   db2profc -url=jdbc:db2:sample -prepoptions="package using $1" $1_SJProfile0
elif (($# < 3))
then
   sqlj     -url=jdbc:db2:$2 $1.sqlj
   db2profc -url=jdbc:db2:$2 -prepoptions="package using $1" $1_SJProfile0
else
   sqlj     -url=jdbc:db2:$2 -user=$3 -password=$4  $1.sqlj
   db2profc -url=jdbc:db2:$2 -user=$3 -password=$4 -prepoptions="package using $1"
     $1_SJProfile0
fi
```

bldsqlj 的轉換程式及前置編譯選項	
sqlj	SQLJ 轉換程式（也編譯此程式）。
$1.sqlj	SQLJ 來源檔。
$1.java	來自 SQLJ 來源檔的已轉換 Java 檔。
db2profc	DB2 for Java 設定檔自訂程式。
-url	指定一個 JDBC URL 建立資料庫連接，如 `jdbc:db2:sample`。
-user	指定使用者 ID（選用性參數）。
-password	指定通行碼（選用性參數）。
-prepoptions	使用字串 `"package using $1"` 指定資料庫的資料包名稱， `$1` 是 SQLJ 來源檔名稱。
$1_SJProfile0	指定程式的序列化設定檔。

Applets

`Applt` 示範一個存取 DB2 資料庫的 SQLJ Applet

使用建置檔 `bldsqlj` 建置此 Applet，然後執行它：

1. 確定 DB2 機器（伺服器或從屬站）有安裝及執行 web 伺服器。
2. 根據檔案的指示修改 `Applt.html` 檔。
3. 在您在 `Applt.html` 中指定的 TCP/IP 埠上啓動 JDBC Applet 伺服器。 例如，若在 `Applt.html` 中您指定 `param name=port value='6789'`，那麼您應該會輸入下列指令：

   ```
   db2jstrt 6789
   ```

   註: 請確定連接字串中的 JDBC 埠號是建議的預設值。如果您能確定號碼與另一埠號不衝突，則可變更此項。不要使用資料庫埠號 ″50000″。

4. 使用下列指令來建置 Applet：

   ```
   bldsqlj Applt [ <db_name> [ <userid> <password> ]]
   ```

其中選用性參數 <db_name> 可讓您存取另一個資料庫而不是預設 sample 資料庫。若要存取的資料庫位於不同案例，例如若從遠端從屬站機器存取伺服器，則需要選用性參數 <userid> 和 <password>。

5. 請確定您的 Web 瀏覽器可以存取您的工作目錄。如果無法存取工作目錄，請複製下列檔案到可存取的目錄：

   ```
   Applt.html                          Applt.class
   Applt_Cursor1.class                 Applt_Cursor2.class
   Applt_SJProfileKeys.class           Applt_SJProfile0.ser
   ```

6. 將 OS/2 和 Windows 32 位元作業系統上的檔案 sqllib\java\db2java.zip 和 sqllib\java\runtime.zip，或將 UNIX 上的 sqllib/java/db2java.zip 和 sqllib/java/runtime.zip，複製到與其它 Applt 檔相同的目錄。
7. 在從屬站機器上，啓動 Web 瀏覽器 (它必須支援 JDK 1.1) 並載入 Applt.html。

步驟 (1)、(5) 及 (7) 的另一個方法是在從屬站機器的工作目錄中輸入下列指令，使用附於 Java Development Kit 的 Applet 檢視器來進行作業：

```
Appletviewer Applt.html
```

您也可使用 Java makefile 建置此程式。

應用程式

App 示範一個存取 DB2 資料庫的 SQLJ 應用程式

若要使用建置檔 bldsqlj 建置此應用程式，請輸入此指令：

```
bldsqlj App [ <db_name> [ <userid> <password> ]]
```

其中選用性參數 <db_name> 可讓您存取另一個資料庫而不是預設 sample 資料庫。若要存取的資料庫位於不同案例，例如若從遠端從屬站機器存取伺服器，則需要選用性參數 <userid> 和 <password>。

使用此指令對應用程式執行 Java 直譯器：

```
java App
```

您也可使用 Java makefile 建置此程式。

儲存程序的從屬站程式

Stclient 是從屬站程式，其使用 JDBC 應用程式驅動程式呼叫 SQLJ 儲存程序類別 Stserver。在建置及執行此從屬站應用程式之前，請在伺服器中建置儲存程序類別。請參閱第85頁的『儲存程序』。

若要使用建置檔 bldsqlj 建置此從屬站程式，請輸入此指令：

```
bldsqlj Stclient [ <db_name> [ <userid> <password> ]]
```

其中選用性參數 <db_name> 可讓您存取另一個資料庫而不是預設 sample 資料庫。若要存取的資料庫位於不同案例，例如若從遠端從屬站機器存取伺服器，則需要選用性參數 <userid> 和 <password>。

使用此指令對從屬站應用程式執行 Java 直譯器：

```
java Stclient
```

您也可使用 Java makefile 建置此程式。

使用者定義函數的從屬站程式

UDFclie 是從屬站程式，它使用 JDBC 應用程式驅動程式，在伺服器程式 UDFsrv 中，呼叫已執行的使用者定義的函數。建置和執行此從屬站應用程式之前，請在伺服器建置 UDFsrv 程式。請參閱第89頁的『使用者定義的函數 (UDF)』。

若要使用建置檔 bldsqlj 建置此 SQLJ 從屬站程式，請輸入此指令：

```
bldsqlj UDFclie [ <db_name> [ <userid> <password> ]]
```

其中選用性參數 <db_name> 可讓您存取另一個資料庫而不是預設 sample 資料庫。若要存取的資料庫位於不同案例，例如若從遠端從屬站機器存取伺服器，則需要選用性參數 <userid> 和 <password>。

使用此指令對從屬站應用程式執行 Java 直譯器：

```
java UDFclie
```

您也可使用 Java makefile 建置此程式。

儲存程序

建置檔 bldsqljs 含有一些建置 SQLJ 儲存程序的指令。在 UNIX 上，這是一個 Script 檔。在 OS/2 上，它是指令檔 bldsqljs.cmd，在 Windows 上，它是批次檔 bldsqljs.bat。此指令的內容和批次檔內容相同，而且會先出現本版本，然後出現 UNIX Script 檔。

在 OS/2 和 Windows 32 位元作業系統的下列建置檔中，第一個參數 %1 指定來源檔名稱。第二個參數 %2 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的同一案例中，所以不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。 若沒有提供資料庫名稱，程式會使用預設 sample 資料庫。

```
@echo off
rem bldsqljs -- OS/2 and Windows 32-bit operating systems
rem Builds a Java embedded SQL (SQLJ) stored procedure
rem Usage: bldsqljs prog_name [ db_name ]

if "%1" == "" goto error

rem Translate and compile the SQLJ source file
rem and bind the package to the database.
if "%2" == "" goto case1
goto case2
:case1
sqlj %1.sqlj
   db2profc -url=jdbc:db2:sample -prepoptions="package using %1" %1_SJProfile0
   goto continue
:case2
   sqlj     -url=jdbc:db2:%2 %1.sqlj
   db2profc -url=jdbc:db2:%2 -prepoptions="package using %1" %1_SJProfile0
:continue

rem Copy the *.class and *.ser files to the 'function' directory.
copy %1*.class "%DB2PATH%\function"
copy %1*.ser "%DB2PATH%\function"

goto exit
:error
echo Usage: bldsqljs prog_name [ db_name ]
:exit
@echo on
```

bldsqljs 的轉換程式及前置編譯選項

sqlj SQLJ 轉換程式（也編譯此程式）。

%1.sqlj
SQLJ 來源檔。

%1.java
來自 SQLJ 來源檔的已轉換 Java 檔。

db2profc
DB2 for Java 設定檔自訂程式。

-url 指定一個 JDBC URL 建立資料庫連接，如 `jdbc:db2:sample`。

-prepoptions
使用字串 `"package using %1"` 指定資料庫的資料包名稱， `%1` 是 SQLJ 來源檔名稱。

%1_SJProfile0
指定程式的序列化設定檔。

在下面 UNIX Script 檔中，第一個參數 $1 指定來源檔名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的同一案例中，所以不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。 若沒有提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldsqljs script file -- UNIX platforms
# Builds a Java embedded SQL (SQLJ) stored procedure
# Usage: bldsqljs <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Translate and compile the SQLJ source file
# and bind the package to the database.
if (($# < 2))
then
   sqlj $1.sqlj
   db2profc -url=jdbc:db2:sample -prepoptions="package using $1" $1_SJProfile0
else
   sqlj     -url=jdbc:db2:$2 $1.sqlj
   db2profc -url=jdbc:db2:$2 -prepoptions="package using $1" $1_SJProfile0
fi

# Copy the *.class and *.ser files to the 'function' directory.
rm -f $DB2PATH/function/$1*.class
rm -f $DB2PATH/function/$1*.ser
cp $1*.class $DB2PATH/function
cp $1*.ser $DB2PATH/function
```

bldsqljs 的轉換程式及前置編譯選項
sqlj SQLJ 轉換程式（也編譯此程式）。
$1.sqlj SQLJ 來源檔。
$1.java 來自 SQLJ 來源檔的已轉換 Java 檔。
db2profc DB2 for Java 設定檔自訂程式。
-url 指定一個 JDBC URL 建立資料庫連接，如 jdbc:db2:sample。
-prepoptions 使用字串 "package using $1" 指定資料庫的資料包名稱， $1 是 SQLJ 來源檔名稱。
$1_SJProfile0 指定程式的序列化設定檔。

Stserver 示範 PARAMETER STYLE JAVA 儲存程序，使用 JDBC 應用程式驅動程式存取 DB2 資料庫。在伺服器上編譯和儲存儲存程序。從屬站應用程式呼叫時，它們會存取伺服器資料庫並將資訊傳回從屬站應用程式。

欲使用建置檔 bldsqljs 建置此儲存程序類別：

1. 請輸入下列指令：

   ```
   bldsqljs Stserver [ <db_name> ]
   ```

   其中選用性參數 <db_name> 可讓您存取另一個資料庫而不是預設 sample 資料庫。

2. 下一步，在伺服器上執行 Stcreate.db2 script，將儲存程序編目。首先，連接資料庫：

   ```
   db2 connect to sample
   ```

   如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

   ```
   db2 -td@ -vf Stdrop.db2
   ```

   然後，使用下列指令編目儲存程序：

   ```
   db2 -td@ -vf Stcreate.db2
   ```

3. 然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請將共用檔案庫的檔案模式設爲「執行」，以便 DB2 案例可以存取檔案庫。

4. 編譯及執行 Stclient 從屬站應用程式，呼叫儲存程序。請參閱第84頁的『儲存程序的從屬站程式』。

您也可使用 Java makefile 建置此程式。

使用者定義的函數 (UDF)

UDFsrv 示範 Java 使用者定義的函數。由於 UDF 程式不含 SQL 陳述式，所以 Java UDF 程式無法包含 SQLJ 陳述式。在伺服器上建立 UDFsrv 檔案庫之後，從屬站應用程式可存取該檔案庫。 DB2 提供 JDBC 從屬站應用程式 UDFcli，以及提供 SQLJ 從屬站應用程式 UDFclie。這兩個應用程式可用來存取 UDFsrv 檔案庫。

透過在命令行上輸入的指令在伺服器建置和執行 UDF 程式：

1. 使用此指令編譯 UDFsrv.java 以產生檔案 UDFsrv.class：

   ```
   javac UDFsrv.java
   ```

2. 複製 UDFsrv.class 到 OS/2 及 Windows 32 位元作業系統的 sqllib\function 目錄，或到 UNIX 的 sqllib/function 目錄。
3. 若要存取 UDFsrv 檔案庫，您可使用 JDBC 或 SQLJ 從屬站應用程式。若要編譯和執行 JDBC 從屬站應用程式 UDFcli，請參閱第79頁的『使用者定義函數的從屬站應用程式』。若要編譯和執行 SQLJ 從屬站應用程式 UDFclie，請參閱第85頁的『使用者定義函數的從屬站程式』。

UDFcli 和 UDFclie 包含 CREATE FUNCTION SQL 陳述式，您可使用這些陳述式向資料庫登記內含於 UDFsrv 的 UDF。UDFcli 和 UDFclie 也包含一些使用 UDF 的 SQL 陳述式（一旦它們被登記之後）。

DB2 Java Applets 的一般要點

1. Java Applet 使用的 db2java.zip 檔案與 JDBC Applet 伺服器處於相同的 FixPak 層次很重要。在正常情況下，db2java.zip 從 JDBC Applet 伺服器正在執行的 Web 伺服器載入。這會確保相配。不過，如果您設定的 Java Applet 是從不同的位置載入 db2java.zip，則會發生不符情況。在兩個檔案之間使 FixPak 層次相符在連接時間嚴格執行。如果偵測出不符，則連接遭拒，並且從屬站接收到下列其中一項異常狀況：
   - 如果 db2java.zip 是 DB2 版本 7 FixPak 2 或更新版本：

     ```
     COM.ibm.db2.jdbc.DB2Exception：[IBM][JDBC 驅動程式]
     CLI0621E  未支援的 JDBC 伺服器設定。
     ```

   - 如果 db2java.zip 在 FixPak 2 之前：

```
COM.ibm.db2.jdbc.DB2Exception：[IBM][JDBC 驅動程式]
CLI0601E  已結束無效陳述式 handle 或陳述式。
SQLSTATE=S1000
```

如果發生不符情況，則 JDBC Applet 伺服器在 jdbcerr.log 檔案中記載下列其中一項訊息：

- 如果 JDBC Applet 伺服器是 DB2 版本 7 FixPak 2 或更新版本：

```
jdbcFSQLConnect：JDBC Applet Server 及從屬站 (db2java.zip)
版本不相配。無法繼續連接。, einfo= -111
```

- 如果 JDBC Applet 伺服器在 FixPak 2 之前：

```
jdbcServiceConnection()：接收到無效要求。, einfo= 0
```

若要測試您的 JDBC 環境，您可以使用 sqllib\samples\java (Windows) 或 sqllib/samples/java (UNIX) 中的範例檔案 db2JDBCVersion.java。db2JDBCVersion 程式檢查目前使用之 DB2 JDBC 驅動程式的版本，以及 JDBC 環境是否已爲其正確設定。此程式不可用於 OS/2。

2. 就由數個 Java 類別組成的大型 JDBC 或 SQLJ Applet 而言，您可選擇將它的類別備份至單一 JAR 檔。就 SQLJ Applet 而言，您也必須將序列設定檔及其類別放置一處。若選擇這樣做，請將 JAR 檔新增至 "Applet" 標籤中的 archive 參數。關於詳細資訊，請參閱 JDK 版本 1.1 文件。

   就 SQLJ Applet 而言： 有些瀏覽器尚未支援從一個與 Applet 相關的資源檔載入序列化物件。例如，使用那些瀏覽器載入 Applet Applt 時會出現下列錯誤訊息：

   ```
   java.lang.ClassNotFoundException: Applt_SJProfile0
   ```

   其解決之道，是使用一個公用程式將序列化設定檔轉換成以 Java 類別格式儲存的設定檔。此公用程式是一個稱爲 sqlj.runtime.profile.util.SerProfileToClass 的 Java 類別。它將序列化設定檔資源檔視爲輸入，並產生一個含有設定檔的 Java 類別作爲輸出。您可使用下列其中一個指令來轉換設定檔：

   ```
   profconv Applt_SJProfile0.ser
   ```

   或

   ```
   java sqlj.runtime.profile.util.SerProfileToClass Applt_SJProfile0.ser
   ```

   結果會建立類別 Applt_SJProfile0.class。使用 .class 格式的設定檔取代 Applet 使用的 .ser 格式的所有設定檔，問題就會解決。

3. 您可能需要將檔案 db2java.zip （就 SQLJ Applet 而言，也包括檔案 runtime.zip）放在一個由數個可從網站下載的 Applet 共用的目錄。這些檔案位於 OS/2 和 Windows 32 位元作業系統上的 sqllib\java 目錄，以及位於 UNIX 上的 sqllib/java 目錄。您可能需要將 codebase 參數新增至 HTML 檔中的 "Applet" 標籤以識別此目錄。關於詳細資訊，請參閱 JDK 版本 1.1 文件。
4. 自從 DB2 版本 5.2 之後，信號處理常式已新增至 JDBC Applet 伺服器（聆聽者）db2jd，使它更加健全。因此使用者無法使用 CTRL-C 指令來結束 db2jd。因此，終止聆聽者的唯一方法是結束此處理。
5. 關於在 web 伺服器上執行 DB2 Java Applet 的資訊 (尤其是在 Domino Go Webserver)，請參閱：

   http://www.ibm.com/software/data/db2/db2lotus/gojava.htm

第5章 開發 SQL 程序

本章提供建置「DB2 SQL 程序」的詳細資訊。

「DB2 SQL 程序」範例程式，在 UNIX 平台上是位於 sqllib/samples/sqlproc 目錄中，在 OS/2 及 Windows 32 位元作業系統上是位於 %DB2PATH%\samples\sqlproc 目錄中。若需範例程式的說明，請參閱第27頁的表10。

設定環境、建立及呼叫 SQL 程序之方法範例

本章詳細說明如何在所有受支援的平台上設定「SQL 程序」環境、建立「SQL 程序」，以及在這些環境中呼叫這些程序。本導引章節提供的範例，說明在 Windows NT 或 Windows 2000 環境中，如何使用 Microsoft Visual C++ 版本 6.0 編譯器來進行這三種作業。您可以透過作出適當的變更 (在本章其餘部份中有詳細說明)，將這些步驟套用於所有受支援的環境中。

藉由剪下下列指令，並將其貼上到批次檔中，然後在 DB2CLP 視窗中執行之，即可執行 Microsoft Visual C++ 版本 6.0 編譯器的環境設定指令。請確保對於特定的環境作出所有必要的變更 (其中包括路徑設定)：

```
@echo on
   rem Setting the SQL PROCEDURE environment:
   db2set DB2_SQLROUTINE_COMPILER_PATH="c:\Micros~1\vc98\bin\vcvars32.bat"
   db2set DB2_SQLROUTINE_COMPILE_COMMAND="cl -Od -W2 /TC -D_X86_=1
            -I%DB2PATH%\include SQLROUTINE_FILENAME.c /link -dll
     -def:SQLROUTINE_FILENAME.def /out:SQLROUTINE_FILENAME.dll
            %DB2PATH%\lib\db2api.lib"
@echo off
```

一旦您的環境設定完畢，即可在 DB2CLP 視窗中建立並呼叫「SQL 程序」。以下爲使用 whiles.db2 範例檔案的範例：

1. 將「SQL 程序」範例檔案複製到工作目錄中：

   ```
   C:\> xcopy /I c:\sqllib\samples\sqlproc c:\sqlproc
   ```

2. 跳至工作目錄：

   ```
   C:\> cd c:\sqlproc
   ```

3. 連接至資料庫：

   ```
   C:\sqlproc> db2 connect to sample
   ```

4. 執行下列 whiles.db2 Script：

   ```
   C:\sqlproc> db2 -td@ -vf whiles.db2
   ```

5. 請使用程序名稱、IN 參數值及 OUT 參數的 "?"，輸入 CALL 指令：

   ```
   C:\sqlproc> db2 "call dept_median (51,?)"
   ```

   您應接收到此結果：

   ```
   MEDIANSALARY: 1.76545000000000e+004
   "DEPT_MEDIAN" RETURN_STATUS: "0"
   ```

   若您在執行上述指令時，收到下列錯誤訊息：

   ```
   SQL0805N  找不到資料包 "NULLID.SQLLD202"。SQLSTATE=51002
   ```

   則您可能需要將您的 DB2 公用程式重新連結到資料庫。請從 SQLLIB\bnd 目錄中發出下列指令：

   ```
   C:\SQLLIB\bnd> DB2 bind @db2ubind.lst blocking all grant public
   C:\SQLLIB\bnd> DB2 bind @db2cli.lst blocking all grant public
   ```

   並重新執行 CALL 指令。

設定 SQL 程序環境

下列是關於在第33頁的『第2章 設定』中設定 DB2 環境指示的附加說明。

註:

1. 在 OS/2 FAT 檔案系統上，「SQL 程序」的綱目名稱限制爲八個字元以下。綱目名稱若長於八個字元，則必須使用 HPFS 檔案系統。
2. 在 UNIX 系統上，僅有在得到 .fenced 檔案擁有者的許可權之後，才可執行隔離的 SQL 常式。請確定，.fenced 檔案的擁有者亦擁有該檔案所在的 $HOME/sqllib/adm 目錄。另外，無論何時變更 .fenced 檔案的擁有者，皆請對下列目錄 (若存在此目錄) 的檔案許可權作出適當的變更：

   UNIX

   ```
   $HOME/sqllib/function/routine/sqlproc/<db_name>/<schema_name>/tmp
   ```

   OS/2 及 Windows

   ```
   %DB2PATH%\function\routine\sqlproc\<db_name>\<schema_name>\tmp
   ```

   其中 <*db_name*> 及 <*schema_name*> 是用來建立 SQL 程序的資料庫及綱目。

若需 SQL 程序支援，您必須在伺服器上安裝 Application Development Client。若需安裝 Application Development Client 的相關資訊，請參照適用您的平台的 *快速入門* 一書。若需您平台中 DB2 支援的 C 及 C++ 編譯器的相關資訊，請參閱第 6頁的『平台支援的軟體』。

需使用兩個變數來完成編譯器的配置：

DB2_SQLROUTINE_COMPILER_PATH
藉由向編譯器二進位、程式庫及併入檔提供路徑，可設定該編譯器的環境變數。

DB2_SQLROUTINE_COMPILE_COMMAND
指定 DB2 用來編譯爲 SQL 程序而建立之 C 檔案的完整指令。

您可使用 db2set 指令，亦可在「儲存程序建置器」中使用「SQL 儲存程序建置選項」對話框，來設定這些 DB2 登錄變數值。若使用「SQL 儲存程序建置選項」對話框，無需實際存取資料庫伺服器，亦無需重新啓動資料庫伺服器，即可使變更生效。

設定編譯器環境變數

在 OS/2、Windows 及 UNIX 作業系統上配置環境有不同的規則。在部份情況下，無需進行任何配置；而在其它情況下，必須將 DB2_SQLROUTINE_COMPILER_PATH DB2 登錄變數，配置爲指向可適當設定環境變數的可執行 Script。必須執行 db2start 指令，以使設定生效。以下內容說明如何設定受支援平台的編譯器環境變數：

OS/2 對於 IBM VisualAge C++ for OS/2 版本 3.6：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\ibmcxxo\bin\setenv.cmd"
```

對於 IBM VisualAge C++ for OS/2 版本 4：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\ibmcpp40\bin\setenv.cmd"
```

註: 在使用這些指令時，假定已在 c: 磁碟機上安裝有 C++ 編譯器。如有必要，可變更磁碟機或路徑，以反映 C++ 編譯器在系統中的位置。

UNIX 在您首次編譯儲存程序時，DB2 會產生一可執行的 Script 檔 $HOME/sqllib/function/routine/sr_cpath (包含有編譯器環境變數的預設值)。若此預設值不適合您的編譯器，您可以編輯此檔案。另外，您亦可設定 DB2_SQLROUTINE_COMPILER_PATH DB2 登錄變數，以包含其它可執行 Script 的完整路徑名稱 (此可執行 Script 可指定您需要的設定值)。

Windows

在 Windows 32 位元作業系統中，若將您的編譯器環境變數設定爲 SYSTEM 變數，則無需進行配置。否則，請將 DB2_SQLROUTINE_COMPILER_PATH DB2 登錄變數設定爲如下內容：

對於 Microsoft Visual C++ 版本 5.0：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\devstudio\vc\bin\vcvars32.bat"
```

對於 Microsoft Visual C++ 版本 6.0：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\Micros~1\vc98\bin\vcvars32.bat"
```

對於 IBM VisualAge C++ for Windows 版本 3.6：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\ibmcxxw\bin\setenv.bat"
```

對於 IBM VisualAge C++ for Windows 版本 4：

```
db2set DB2_SQLROUTINE_COMPILER_PATH="c:\ibmcppw40\bin\setenv.bat"
```

註: 在使用這些指令時，假定已在 c: 磁碟機上安裝有 C++ 編譯器。如有必要，可變更磁碟機或路徑，以反映C++ 編譯器在您系統中的位置。

自訂編譯指令

若安裝有 Application Development Client，則會提供預設編譯指令，它可至少用於每個伺服器平台所支援的編譯器之一：

AIX IBM C Set++ for AIX 版本 3.6.6

HP-UX

HP aC++ 版本 A.03.25

Linux

GNU/Linux g++

OS/2 IBM VisualAge C++ for OS/2 版本 3.6.5

PTX ptx/C++ 版本 5.2

Solaris

Forte/WorkShop C++ 版本 4.2、5.0、6 及 6.1

Windows NT 及 Windows 2000

Microsoft Visual C++ 版本 5.0 及 6.0

若要使用其它編譯器，或自訂預設指令，您必須對 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數進行如下設定：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=<compile_command>
```

其中 <compile_command> 爲 C 或 C++ 編譯指令，包括建立儲存程序所必需的選項及參數。必須執行 db2start 指令，以使設定生效。

在編譯指令中，請使用關鍵字 SQLROUTINE_FILENAME 取代所產生的 SQC、C、PDB、DEF、EXP、訊息日誌及共用程式庫檔案的檔名。只有對 AIX，才使用關鍵字 SQLROUTINE_ENTRY 取代進入點名稱。以下爲受支援伺服器平台上之 C 或 C++ 編譯器的 DB2_SQLROUTINE_COMPILE_COMMAND 預設值。

AIX 若使用 IBM C for AIX 版本 3.6.6，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=xlc -H512 -T512 \
-I$HOME/sqllib/include SQLROUTINE_FILENAME.c -bE:SQLROUTINE_
FILENAME.exp \
-e SQLROUTINE_ENTRY -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -lc
-ldb2
```

若使用 IBM C Set++ for AIX 版本 3.6.6，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=xlC -H512 -T512 \
-I$HOME/sqllib/include SQLROUTINE_FILENAME.C -bE:SQLROUTINE_
FILENAME.exp \
-e SQLROUTINE_ENTRY -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -lc
-ldb2
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

註: 欲在 AIX 上編譯 64 位元 SQL 程序，請新增 -q64 選項到上述指令中。

若使用 IBM VisualAge C++ for AIX 版本 4，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld"
```

若您未在 vacbld 指令之後指定配置檔，則 DB2 會在首次嘗試建立 SQL 程序時，建立以下預設配置檔：

```
$HOME/sqllib/function/routine/sqlproc.icc
```

您可以在設定 DB2_SQLROUTINE_COMPILE_COMMAND 的 DB2 登錄值時，指定自己的配置檔：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld $HOME/sqllib/function/sqlproc.
icc"
```

HP-UX

若使用 HP C Compiler 版本 A.11.00.03，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=cc +DAportable +ul -Aa +z \
-I$HOME/sqllib/include -c SQLROUTINE_FILENAME.c; \
ld -b -o SQLROUTINE_FILENAME SQLROUTINE_FILENAME.o \
-L$HOME/sqllib/lib -ldb2
```

若使用 HP aC++ 版本 A.03.25，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=aCC +DAportable +u1 +z -ext \
-I$HOME/sqllib/include SQLROUTINE_FILENAME.C -b \
-o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -ldb2
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

Linux

若使用 GNU/Linux gcc，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=cc \
  -I$HOME/sqllib/include SQLROUTINE_FILENAME.c \
-shared -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -ldb2
```

若使用 GNU/Linux g++，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=g++ \
-I$HOME/sqllib/include SQLROUTINE_FILENAME.C \
-shared -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib -ldb2
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

PTX 若使用 ptx/C 版本 4.5，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=cc -KPIC \
  -I$HOME/sqllib/include SQLROUTINE_FILENAME.c \
-G -o SQLROUTINE_FILENAME.so -L$HOME/sqllib/lib -ldb2 ; \
cp SQLROUTINE_FILENAME.so SQLROUTINE_FILENAME
```

若使用 ptx/C++ 版本 5.2，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=c++ -KPIC \
-D_RWSTD_COMPILE_INSTANTIATE=0
-I$HOME/sqllib/include SQLROUTINE_FILENAME.C \
-G -o SQLROUTINE_FILENAME.so -L$HOME/sqllib/lib -ldb2 ; \
cp SQLROUTINE_FILENAME.so SQLROUTINE_FILENAME
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

OS/2 若使用 IBM VisualAge C++ for OS/2 版本 3.6.5，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="icc -Ge- -Gm+ -W2
-I%DB2PATH%\include SQLROUTINE_FILENAME.c
/B\"/NOFREE /NOI /ST:64000\" SQLROUTINE_FILENAME.def
%DB2PATH%\lib\db2api.lib"
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

若使用 IBM VisualAge C++ for OS/2 版本 4，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld"
```

若您未在 vacbld 指令之後指定配置檔，則 DB2 會在首次嘗試建立 SQL 程序時，建立以下預設配置檔：

```
%DB2PATH%\function\routine\sqlproc.icc
```

若您要使用自己的配置檔，則可在設定 DB2_SQLROUTINE_COMPILE_COMMAND 的 DB2 登錄值時，指定自己的配置檔：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld
%DB2PATH%\function\sqlproc.icc"
```

Solaris

若使用 Forte/WorkShop C 版本 4.2 及 5.0，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=cc -xarch=v8plusa -Kpic \
  -I$HOME/sqllib/include SQLROUTINE_FILENAME.c \
-G -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib \
-R$HOME/sqllib/lib -ldb2
```

若使用 Forte/WorkShop C++ 版本 4.2 及 5.0，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=CC -xarch=v8plusa -Kpic \
-I$HOME/sqllib/include SQLROUTINE_FILENAME.C \
-G -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib \
-R$HOME/sqllib/lib -ldb2
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

註:

1. 將編譯器選項 `-xarch=v8plusa` 新增至預設編譯器指令，是爲了避免編譯器在與 `libdb2.so` 鏈結時，不產生有效可執行檔的問題。
2. 若要在 Solaris 上編譯 64 位元 SQL 程序，請除去 `-xarch=v8plusa` 選項，並將 `-xarch=v9` 選項新增至上述指令中。

Windows NT 及 Windows 2000

註: Windows 95、Windows 98、或 Windows Millennium Edition 不支援 SQL 程序。

若使用 Microsoft Visual C++ 版本 5.0 及 6.0，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=cl -Od -W2 /TC -D_X86_=1
-I%DB2PATH%\include SQLROUTINE_FILENAME.c /link -dll
  -def:SQLROUTINE_FILENAME.def /out:SQLROUTINE_FILENAME.dll
  %DB2PATH%\lib\db2api.lib
```

若尚未設定 DB2_SQLROUTINE_COMPILE_COMMAND DB2 登錄變數，則此爲預設編譯指令。

若使用 IBM VisualAge C++ for Windows 版本 3.6.5，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="ilib /GI
SQLROUTINE_FILENAME.def & icc -Ti -Ge- -Gm+ -W2
-I%DB2PATH%\include SQLROUTINE_FILENAME.c
/B\"/ST:64000 /PM:VIO /DLL\" SQLROUTINE_FILENAME.exp
%DB2PATH%\lib\db2api.lib"
```

若使用 IBM VisualAge C++ for Windows 版本 4，則爲：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld"
```

若您未在 vacbld 指令之後指定配置檔，則 DB2 會在首次嘗試建立 SQL 程序時，建立以下預設配置檔：

```
%DB2PATH%\function\routine\sqlproc.icc
```

若您要使用自己的配置檔，則可在設定 DB2_SQLROUTINE_COMPILE_COMMAND 的 DB2 登錄值時，指定自己的配置檔：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND="vacbld
%DB2PATH%\function\sqlproc.icc"
```

欲傳回預設編譯器選項，請使用下列指令，將 DB2_SQLROUTINE_COMPILE_COMMAND 的 DB2 登錄值設定爲 NULL 值：

```
db2set DB2_SQLROUTINE_COMPILE_COMMAND=
```

保留中間檔

當您發出 CREATE PROCEDURE 陳述式時，DB2 會建立許多中間檔，如果 DB2 順利完成陳述式，這些檔案就會被刪除。 如果 SQL 程序沒有如預期地執行，可能檢查 DB2 所建立的 SQC、C 及訊息日誌檔會有助於解決問題。欲保留 DB2 在順利執行 CREATE PROCEDURE 陳述式期間內所建立的檔案，您必須將 DB2_SQLROUTINE_KEEP_FILES DB2 登錄變數的值設定爲 ″1″、″y″ 或 ″yes″，如下列指令所示：

```
db2set DB2_SQLROUTINE_KEEP_FILES=1
```

在未能順利建立 SQL 程序時，可能留下一些中間檔，您須手動刪除之。這些檔案位於下列目錄中：

UNIX

```
$HOME/sqllib/function/routine/sqlproc/<db_name>/<schema_name>/tmp
```

OS/2 及 Windows

```
%DB2PATH%\function\routine\sqlproc\<db_name>\<schema_name>\tmp
```

其中 <*db_name*> 及 <*schema_name*> 是用來建立 SQL 程序的資料庫及綱目。

自訂前置編譯及連結選項

您可以設定 DB2_SQLROUTINE_PREPOPTS DB2 登錄變數，自行設定前置編譯及連結選項。這些選項不能在程序層次上自行設定。欲指定 SQL 程序的自訂前置編譯選項，請使用下列指令，將 DB2 前置編譯器所使用的前置編譯選項列示放入 DB2 登錄：

```
db2set DB2_SQLROUTINE_PREPOPTS=options
```

其中 *options* 指定 DB2 前置編譯器所使用的前置編譯選項列示。僅接受下列選項：

```
BLOCKING {UNAMBIG | ALL | NO}
DATETIME {DEF | USA | EUR | ISO | JIS | LOC}
DEGREE {1 | degree-of-parallelism | ANY}
DYNAMICRULES {BIND | RUN}
EXPLAIN {NO | YES | ALL}
EXPLAINSNAP {NO | YES | ALL}
FEDERATED {NO | YES}
INSERT {DEF | BUF}
ISOLATION {CS |RR |UR |RS |NC}
QUERYOPT optimization-level
SYNCPOINT {ONEPHASE | TWOPHASE | NONE}
```

備份及復置

在建立 SQL 程序時，若動態鏈結程式庫 (DLL) 小於 2 百萬位元組，則所產生的共用 DLL 亦會保留於型錄表格中。在備份及復置資料庫時，將會使用保留於型錄表格中的版本，來備份及復置任何所產生之共用 DLL 小於 2 百萬位元組的 SQL 程序。若您 SQL 程序所產生的共用 DLL 大於 2 百萬位元組，請確保您亦執行資料庫備份及復置以及檔案系統備份及復置。否則，您須藉由使用 syscat.procedures 型錄表格中的來源，來手動重建該 DLL。

註: 在資料庫回復時，將會移除隸屬於回復資料庫之檔案系統上的所有 SQL 程序可執行檔。若將索引建立配置參數 indexrec 設定爲 RESTART，則將從型錄

表格中取出所有 SQL 程序可執行檔，並在下次連線時放回到檔案系統中。否則，將在首次執行 SQL 程序時，取出 SQL 可執行檔。

將可執行檔放回到以下目錄中：

UNIX $HOME/sqllib/function/routine/sqlproc/<*database_name*>

OS/2 及 Windows

%DB2PATH%\function\routine\sqlproc\<*database_name*>

其中 <*database_name*> 代表建立 SQL 程序時所使用的資料庫。

註: 若在復置作業之後，首次嘗試連線到資料庫時，傳回以下內容：

```
SQL2048N 在存取物件 "SQL PROCEDURE FILES" 時發生錯誤。
原因碼："7"。
```

則必需使用 db2stop 來停止 DB2，並使用 db2start 重新啓動之。

建立 SQL 程序

「DB2 命令行處理器」Script 在 UNIX 上是位於 sqllib/samples/sqlproc 目錄，在 OS/2 及 Windows 上則是位於 %DB2PATH%\samples\sqlproc 目錄 (那些 Script 檔均以 .db2 副檔名結尾)，它們會在伺服器上執行建立儲存程序的 CREATE PROCEDURE 陳述式。 每一個 CLP Script 都有一個名稱相同的對應從屬站應用程式檔，其副檔名爲 .sqc 或 .c。

在執行 CREATE PROCEDURE CLP Script 之前，請使用下列指令連接範例資料庫：

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

其中 *userid* 及 *password* 是 sample 資料庫所在位置的案例之使用者 ID 及通行碼。

欲執行 rsultset.db2 Script 檔中所含的 CREATE PROCEDURE 陳述式，請輸入下列指令：

```
db2 -td@ -vf rsultset.db2
```

現在，您可以呼叫 SQL 程序，如下面『呼叫 SQL 程序』所述。

呼叫 SQL 程序

您可以使用命令行處理器 (CLP) call 指令，或經由建置從屬站應用程式來呼叫 SQL 程序。

使用 CALL 指令

欲使用 call 指令，您必須輸入儲存程序名稱，以及任何 IN 或 INOUT 參數、'?' 作爲每個 OUT 參數的位置保留符號。

首先，請遵循第102頁的『建立 SQL 程序』中的步驟，建立 SQL 程序。

欲呼叫 SQL 程序，您必須先連接資料庫：

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

其中 *userid* 及 *password* 是 sample 資料庫所在位置的案例之使用者 ID 及通行碼。

儲存程序的參數已在程式來源檔中儲存程序的 CREATE PROCEDURE 陳述式中給定。例如，在來源檔 whiles.db2 中，DEPT_MEDIAN 程序的 CREATE PROCEDURE 陳述式會開始：

```
CREATE PROCEDURE DEPT_MEDIAN
(IN deptNumber SMALLINT, OUT medianSalary DOUBLE)
```

若要呼叫此程序，您必須在 IN 參數 deptNumber 中放入一個有效的 SMALLINT 值，並在 OUT 參數中放入問號 '?'。您可以從範例資料庫的對應表格，或檢查從屬站呼叫程式來源檔，取得一有效值作爲使用的值。在 whiles.sqc 中，則會使用值 ″51″：

```
printf("Use CALL with Host Variables to invoke the Server Procedure "
   "named %s\n", procname);
dept = 51;                      /* get median for dept. 51 */
```

請使用程序名稱、IN 參數值及 OUT 參數值 (問號 '?')，輸入 call 指令。程序的參數必須以括弧括住，且必須使用雙引號，如下所示：

```
db2 "call dept_median (51, ?)"
```

您應會接收到此結果：

```
MEDIANSALARY: 1.76545000000000e+004
"DEPT_MEDIAN" RETURN_STATUS: "0"
```

若您在呼叫上述指令時，收到下列錯誤訊息：

```
SQL0805N  找不到資料包 "NULLID.SQLLD202"。SQLSTATE=51002
```

則您可能需要將您的 DB2 公用程式重新連結到資料庫。請從 SQLLIB\bnd 目錄中發出下列指令：

```
C:\SQLLIB\bnd> DB2 bind @db2ubind.lst blocking all grant public
C:\SQLLIB\bnd> DB2 bind @db2cli.lst blocking all grant public
```

並重新執行 CALL 指令。

在使用 call 指令時，請記住下列要點：

- 結果直欄的長度最大值爲 1023 個字元。
- 呼叫的儲存程序必須已定義在型錄中。

OS/2 DB2 CLI 從屬站應用程式

指令檔 bldcli.cmd 位於 %DB2PATH%\samples\sqlproc 中，含有建置 SQL 程序的 DB2 CLI 從屬站應用程式之指令。請參閱第221頁的『DB2 CLI 應用程式』，取得有關 bldcli.cmd 的詳細資訊。

欲從來源檔 rsultset.c 建置 DB2 CLI 從屬站應用程式 rsultset，請輸入：

```
bldcli rsultset
```

此指令會建立可執行檔 rsultset。

欲呼叫儲存程序，請輸入可執行檔名稱、您要連接的資料庫名稱、及資料庫案例的使用者 ID 與通行碼，以執行範例從屬站應用程式：

```
rsultset database userid password
```

OS/2 內含的 SQL 從屬站應用程式

指令檔 bldapp.cmd 位於 %DB2PATH%\samples\sqlproc 中，含有可以建置 SQL 程序之內含的 SQL 從屬站應用程式的指令。請參閱第227頁的『DB2 API 及內含的 SQL 應用程式』，取得有關 bldapp.cmd 的詳細資訊。

欲從來源檔 basecase.sqc 建置內含的 SQL 從屬站應用程式 basecase，請輸入指令檔名稱、可執行檔名稱、您要連接的資料庫、及資料庫案例的使用者 ID 及通行碼：

```
bldapp basecase database userid password
```

結果產生可執行檔 basecase。

若要呼叫儲存程序，請輸入下列指令來執行從屬站應用程式：

```
basecase database userid password
```

UNIX DB2 CLI 從屬站應用程式

Script 檔 bldcli 位於 sqllib/samples/sqlproc 中，含有可以建立 SQL 程序之 DB2 CLI 從屬站應用程式的指令。若需關於 bldcli Script 檔的詳細資訊，請參閱關於 UNIX 平台的「建置應用程式」一章中「DB2 CLI 應用程式」一節。

欲從來源檔 rsultset.c 建置 DB2 CLI 從屬站應用程式 rsultset，請輸入：

```
bldcli rsultset
```

此指令會建立可執行檔 rsultset。

欲呼叫儲存程序，請輸入可執行檔名稱、您要連接的資料庫名稱、及資料庫案例的使用者 ID 與通行碼，以執行範例從屬站應用程式：

rsultset *database userid password*

UNIX 內含的 SQL 從屬站應用程式

Script 檔 bldapp 位於 sqllib/samples/sqlproc 中，含有建置 SQL 程序之內含的 SQL 從屬站應用程式的指令。若需關於 bldapp Script 檔的詳細資訊，請參閱關於 UNIX 平台的「建置應用程式」一章中「DB2 API 及內含的 SQL 應用程式」一節。

欲從來源檔 basecase.sqc 建置內含的 SQL 從屬站應用程式 basecase，請輸入 Script 檔名稱、可執行檔名稱、您要連接的資料庫、及資料庫案例的使用者 ID 與通行碼：

```
bldapp basecase database userid password
```

結果產生可執行檔 basecase。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

basecase *database userid password*

Windows DB2 CLI 從屬站應用程式

%DB2PATH%\samples\sqlproc 目錄中含有兩個建置檔，可以建置 DB2 CLI 從屬站應用程式：bldmcli 適用於 Microsoft Visual C++ 編譯器，而 bldvcli 適用於 IBM VisualAge C++ 編譯器。若需關於 bldmcli 的詳細資訊，請參閱第328頁的『DB2 CLI 應用程式』。若需關於 bldvcli 的詳細資訊，請參閱第342頁的『DB2 CLI 應用程式』。

欲從來源檔 rsultset.c 建置 DB2 CLI 從屬站應用程式 rsultset，請根據您使用的編譯器，輸入下列其中之一：

```
bldmcli rsultset
```

或

```
bldvcli rsultset
```

這些指令會建立可執行檔 rsultset。

欲呼叫儲存程序，請輸入可執行檔名稱、您要連接的資料庫名稱、及資料庫案例的使用者 ID 與通行碼，以執行範例從屬站應用程式：

rsultset *database userid password*

Windows 內含的 SQL 從屬站應用程式

%DB2PATH%\samples\sqlproc 目錄含有兩個建置檔，可以建置內含的 SQL 從屬站應用程式：bldmapp 適用於 Microsoft Visual C++ 編譯器，且 bldvapp 適用於 IBM VisualAge C++ 編譯器。若需有關 bldmapp 的詳細資訊，請參閱第333頁的『DB2 API 及內含的 SQL 應用程式』。若需有關 bldvapp 的詳細資訊，請參閱第347頁的『DB2 API 及內含的 SQL 應用程式』。

欲從來源檔 basecase.sqc 建置內含的 SQL 從屬站應用程式 basecase，請輸入 Script 檔名稱、可執行檔名稱、您要連接的資料庫、及資料庫案例的使用者 ID 與通行碼。根據您使用的編譯器，此指令會是下列其中之一：

```
bldmapp basecase database userid password
```

或

```
bldvapp basecase database userid password
```

結果產生可執行檔 basecase。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

basecase *database userid password*

分送編譯後的 SQL 程序

註: 若使用 DB2 版本 7 FixPak 3 之前版建立資料庫，則在這些資料庫的 DB2 伺服器之間分送編譯後的 SQL 程序時，您必須啓用 DB2，以取出並安裝之。若要進行如此操作，請於編譯後之 SQL 程序的每台來源或目的地 DB2 伺服器上，輸入下列指令：

```
db2updv7 -d <database_name>
```

對於使用 DB2 版本 7 FixPak 3 或以後版本建立的資料庫而言，在建立資料庫期間，會以隱含方式啓用 DB2。

當您定義 SQL 程序時，即將其轉換爲 C 程式，並在依據目標資料庫進行預先編譯及連結之後，對之進行編譯及鏈結，以建立共用程式庫。在編譯及鏈結步驟

中，需要在資料庫伺服器機器上安裝有 C 或 C++ 編譯器。然而，一旦您定義了 SQL 程序，您即可以編譯後的形式，將其分送到使用同一作業系統及同一 DB2 版本，卻不一定存取 C 或 C++ 編譯器的 DB2 資料庫伺服器機器。出現這些狀況時，DB2 可容許使用者以編譯後的形式，從一資料庫中取出 SQL 程序，並將其以編譯後的形式安裝於其它伺服器機器的另一資料庫中。

DB2 可提供指令行介面，亦可提供程式設計介面，以進行取出及安裝作業。指令行介面包含兩個 CLP 指令：GET ROUTINE 及 PUT ROUTINE。程式設計介面包含兩個內建儲存程序：GET_ROUTINE_SAR 及 PUT_ROUTINE_SAR。若需指令行介面的其餘相關資訊，請參照*Command Reference*。若需程式設計介面的其餘相關資訊，請參照*SQL Reference*。

若要將編譯後的 SQL 程序從一資料庫伺服器分送到另一資料庫伺服器，請執行下列步驟：

1. 建置應用程式，其中包括定義爲該應用程式之一部份的 SQL 程序。
2. 在測試程序之後，可藉由發出 GET ROUTINE 指令或呼叫 GET_ROUTINE_SAR 儲存程序，取出每個程序的編譯後版本，並放入其它檔案中。將這些檔案複製到您的分送媒體中 (如有必要)。
3. 藉由發出 PUT ROUTINE 指令，或呼叫 PUT_ROUTINE_SAR 儲存程序，使用在上個步驟中建立的檔案，在每台伺服器上安裝每個程序之編譯後的版本。每台資料庫伺服器必須擁有相同的作業系統及 DB2 層次。

   註: 爲使 PUT ROUTINE 指令或 PUT_ROUTINE_SAR 儲存程序可正常執行，必須在目標機器上安裝 Application Development Client。

第6章 開發 AIX 應用程式

本章提供在 AIX 上建置應用程式的詳細資訊。在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

關於 AIX 的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

註: 欲使用本章中的建置檔案來建置 64 位元應用程式，您可以解除每一個建置檔中指示指令的註解，或以下列指令設定 64 位元物件模式環境：

```
export OBJECT_MODE=64
```

重要注意事項

本節提供 AIX 專用資訊，以便在各種受支援的編譯器上建置 DB2 應用程式。包括：

- 安裝及執行 IBM 與 Micro Focus COBOL
- 儲存程序與 UDF 的進入點

- 儲存程序與 CALL 陳述式
- UDF 與 CREATE FUNCTION 陳述式

安裝及執行 IBM 與 Micro Focus COBOL

因爲 AIX 載入儲存程序及解析它們之間的檔案庫參照的方式，所以就有關於應該如何安裝 COBOL 的需求。當 COBOL 程式在執行期間，載入一個共用檔案庫 (儲存程序) 時，這些需求即變成一個因素。

當載入一個儲存程序時，也須載入它所參照的檔案庫鏈。 當 AIX 搜尋一個僅被您的程式間接參照的檔案庫時，它必須使用已編入檔案庫中，用來參照它的路徑，而這個路徑是由語言供應商 (IBM COBOL 或 Micro Focus COBOL) 所提供的。這個路徑非常可能不是編譯器安裝所在的同一個路徑。如果找不到鏈中的檔案庫，儲存程序的載入將失敗，而且您將收到 SQLCODE -10013。

欲確保這種情況不會發生，請在您想要放置編譯器的位置上安裝它，然後從安裝目錄中對所有語言檔案庫，建立與 /usr/lib (這是一個當需要載入一個檔案庫時，幾乎恆會搜尋的目錄) 的符號式鏈結。 您可以使檔案庫與 sqllib/function (儲存程序目錄) 產生鏈結，但這僅對一個資料庫案例有作用；/usr/lib 對機器上的所有使用者均有作用。強烈建議您不要複製檔案庫；當檔案庫的多個拷貝存在時，這個建議尤其適用於 Micro Focus COBOL。

Micro Focus COBOL 的範例符號式鏈結如下 (假定它安裝在 /usr/lpp/cobdir 中)：

```
[1]> su root
[2]> cd /usr/lib
[1]> ln -sf /usr/lpp/cobdir/coblib/*.a .
```

對於您需要設定的特定路徑，請查看您的編譯器文件。

儲存程序與 UDF 的進入點

儲存程序即是存取資料庫並傳回資訊給您的從屬站應用程式的程式。使用者定義的函數 (UDF) 即是您自己的純量函數或表格函數。在伺服器上編譯儲存程序與 UDF，在伺服器上的共用檔案庫中預存與執行它們。 編譯儲存程序與 UDF 時會建立這些共用檔案庫。

每一個共用檔案庫都有一個進入點，從伺服器呼叫該進入點即可存取共用檔案庫中的程序。 AIX 上的 IBM C 編譯器可讓您在檔案庫中指定任何匯出的函數名稱作爲預設進入點。 僅在儲存程序呼叫或 CREATE FUNCTION 陳述式中指定檔案庫名稱時，才會呼叫這個函數。可透過鏈結步驟中的 -e 選項，做到這一點。 例如：

```
-e funcname
```

使 funcname 成爲預設進入點。關於這個選項如何與 CREATE FUNCTION 陳述式產生關係的資訊，請參閱第112頁的『UDF 和 CREATE FUNCTION 陳述式』。

在其他 UNIX 平台上，沒有如此的機制存在，所以 DB2 假設預設進入點與檔案庫本身同名。

AIX 需要您提供一個匯出檔，這個檔案會指定檔案庫中哪一些廣域函數可從其外部呼叫。 這個檔案必須包括檔案庫中所有儲存程序及/或使用者定義的函數的名稱。其他 UNIX 平台僅匯出檔案庫中的所有廣域函數。這是 AIX 匯出檔的例子：

```
#! outsrv export file
outsrv
```

匯出檔 outsrv.exp 會列出儲存程序 outsrv。鏈結器使用 outsrv.exp，建立共用檔案庫 outsrv，來包含同名的儲存程序。

註: 在開發共用檔案庫之後，通常會將它複製到 DB2 將從其中存取它的目錄內。當嘗試取代儲存程序或使用者定義的函數共用檔案庫時，您應該執行 /usr/sbin/slibclean，來清除 AIX 共用檔案庫快取記憶體，或是從目標目錄中除去檔案庫，然後從來源目錄中，將檔案庫複製到目標目錄中。不然，複製作業可能會失敗，因爲 AIX 保有所參照的檔案庫的快取記憶體，且不容許檔案庫被改寫。

AIX 編譯器文件有關於匯出檔的進一步資訊。

儲存程序與 CALL 陳述式

Application Development Guide 描述如何編寫您的儲存程序。 *SQL Reference* 描述如何在資料庫的位置中，使用 CALL 陳述式來呼叫您的儲存程序。本節說明如何以您在 CALL 陳述式中所提供的資訊，對列出的儲存程序進行編譯及鏈結。

當您編譯及鏈結您的程式時，您可以用兩種方式，來識別函數：

- 使用 -e 選項。

  例如，您可以在鏈結步驟中指定：

  ```
  -e modify
  ```

  這指出已鏈結檔案庫的預設進入點爲函數 modify。

  如果您將鏈結目錄 /u/mydir/procs 中的檔案庫 mystored，且您想要使用上述所指定的預設進入點 modify，請編寫如下的 CALL 陳述式：

  ```
  CALL '/u/mydir/procs/mystored'
  ```

檔案庫 mystored 將載入至記憶體中，而且函數 modify 將被 DB2 挑選，作爲預設進入點，並執行它。

- 使用一個使用 -bE: 選項指定的匯出檔。

  一般說來，當您在您的檔案庫中具有多個儲存程序，且您想要存取附加的函數，作爲儲存程序時，您將會使用這個鏈結選項。

  欲繼續上述的例子，假定檔案庫 mystored 含有三個儲存程序： modify (上述的儲存程序)、 remove 及 add。您將 modify 識別爲上述的預設進入點，並經由在匯出檔中併入 remove 及 add，在鏈結步驟中指出它們是附加的進入點。

  在鏈結步驟中，您指定：

  ```
  -bE:mystored.exp
  ```

  識別匯出檔 mystored.exp。

  匯出檔會是儲存程序函數列示，其中最先列出的是預設進入點：

  ```
   modify
remove
add
  ```

  最後，儲存程序的這兩個 CALL 陳述式 (呼叫 remove 及 add 函數) 的編寫方式如下：

  ```
  CALL '/u/mydir/procs/mystored!remove'
  ```

  及

  ```
  CALL '/u/mydir/procs/mystored!add'
  ```

UDF 和 CREATE FUNCTION 陳述式

Application Development Guide 會描述如何編寫您的 UDF。 *SQL Reference* 則描述如何使用 CREATE FUNCTION 陳述式，向 DB2 登記您的 UDF。本節解譯編譯和鏈結 UDF 之間的關係，以及您在 CREATE FUNCTION 陳述式中的 EXTERNAL NAME 子句上提供的資訊。

當您編譯及鏈結您的程式時，您可以用兩種方式，來識別函數：

- 使用 -e 選項。

  例如，您可以在鏈結步驟中指定：

  ```
  -e modify
  ```

  這指出已鏈結檔案庫的預設進入點爲函數 modify。

如果您正在鏈結目錄 /u/mydir/procs 中的檔案庫 myudfs，而且想要使用上述指定的預設進入點 modify，請在您的 CREATE FUNCTION 函數中指定：

```
EXTERNAL NAME '/u/mydir/procs/myudfs'
```

DB2 會挑選檔案庫 myudfs 的預設進入點，亦即是函數 modify。

- 使用一個使用 -bE: 選項指定的匯出檔。

  一般說來，當您在您的檔案庫中具有多個 UDF，且您想要存取附加的函數，作爲 UDF，您將會使用這個鏈結選項。

  繼續上述的例子，假定檔案庫 myudfs 含有三個 UDF：上述的 modify、remove 及 add。您將 modify 識別爲上述的預設進入點，並經由在匯出檔中併入 remove 及 add，在鏈結步驟中指出它們是附加的進入點。

  在鏈結步驟中，您指定：

  ```
  -bE:myudfs.exp
  ```

  ，它會識別匯出檔 myudfs.exp。

  匯出檔的樣子如下：

```
* additional entry points for myudfs
#!
remove
add
```

  最後， UDF 的這兩個 CREATE FUNCTION 陳述式使用 remove 及 add 函數將含有這些 EXTERNAL NAME 子句：

  ```
  EXTERNAL NAME '/u/mydir/procs/myudfs!remove'
  ```

  及

  ```
  EXTERNAL NAME '/u/mydir/procs/myudfs!add'
  ```

IBM C

本節解譯如何使用具有下列 DB2 介面的 IBM C：

- DB2 CLI
- DB2 API
- 內含的 SQL

DB2 CLI 應用程式

sqllib/samples/cli 中的 Script 檔 bldcli 含有一些建置 DB2 CLI 程式的指令。參數 $1 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名指示，則會呼叫 embprep script 來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- AIX
# Builds a CLI program with IBM C.
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
fi

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the error-checking utility.
xlc $CFLAGS_64 -I$DB2PATH/include -c utilcli.c

# Compile the program.
xlc $CFLAGS_64 -I$DB2PATH/include -c $1.c

# Link the program.
xlc $CFLAGS_64 -o $1 $1.o utilcli.o -L$DB2PATH/lib -ldb2
```

編譯和鏈結 bldcli 的選項
編譯選項：
xlc IBM C 編譯器。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如：$HOME/sqllib/include
-c 僅執行編譯；沒有任何鏈結。此 Script 檔有不同的編譯及鏈結步驟。
鏈結選項：
xlc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。
-o $1 指定可執行的程式。
$1.o 指定目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldcli tbinfo
```

結果會產生可執行檔 tbinfo。 您可輸入這個可執行檔名稱來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中含有 CLI 範例程式的說明，指出使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldcli script 間的唯一不同處。請參閱第113頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果會產生可執行檔 dbmconn。您可輸入這個可執行檔名稱來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

sqllib/samples/cli 中的 Script 檔 bldclisp 含有一些建置 DB2 CLI 儲存程序的指令。參數 $1 會指定您來源檔的檔名；$2 會指定共用檔案庫進入點的儲存程序函數。

```
#! /bin/ksh
# bldclisp script file -- AIX
# Builds a CLI stored procedure in IBM C.
# Usage: bldclisp <prog_name> [ <entry_point> ]
```

```
# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the error-checking utility.
xlc $CFLAGS_64 -I$DB2PATH/include -c utilcli.c

# Compile the program.
xlc $CFLAGS_64 -I$DB2PATH/include -c $1.c

# Link the program.
xlc $CFLAGS_64 -o $1 $1.o utilcli.o -L$DB2PATH/lib \
  -ldb2 -lm -H512 -T512 -bE:$1.exp -e $2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldclisp 的編譯選項和鏈結選項

編譯選項：

xlc IBM C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

bldclisp 的編譯選項和鏈結選項
鏈結選項：
xlc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。
-o $1 指定可執行的程式。
$1.o 指定目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
-lm 與計算檔案庫鏈結。
-H512 指定輸出檔對齊。
-T512 指定輸出檔文字區段的開始位址。
-bE:$.exp 指定匯出檔。匯出檔含有儲存程序的列示。
-e $2 指定共用檔案庫的預設進入點。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，請輸入建置檔檔名、程式名稱及共用檔案庫進入點的儲存程序函數名稱：

```
bldclisp spserver outlanguage
```

Script 檔會將儲存程序複製到 sqllib/function 目錄。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，您可以下列指令將儲存程序編目：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置完成共用檔案庫 spserver，您就可以建置 CLI 從屬站應用程式 spclient，在共用檔案庫中呼叫儲存程序。

您可以使用 script 檔 bldcli 建置 spclient。詳細資訊，請參閱第113頁的『DB2 CLI 應用程式』。

欲存取共用檔案庫，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample，或其別名，或其它的資料庫名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

在 sqllib/samples/c 中的建置檔 bldapp 含有可以建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- AIX
# Builds a C application program.
# Usage:  bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
```

```
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# If embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.c error-checking utility.
  xlc $CFLAGS_64 -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  xlc $CFLAGS_64 -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
xlc $CFLAGS_64 -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  xlc $CFLAGS_64 -o $1 $1.o utilemb.o -ldb2 -L$DB2PATH/lib
else
  # Link the program with utilapi.o
  xlc $CFLAGS_64 -o $1 $1.o utilapi.o -ldb2 -L$DB2PATH/lib
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

xlc IBM C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。編譯及鏈結是不同的步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
xlc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。
-o $ 指定可執行的程式。
$1.o 指定程式目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果不是內含的 SQL 程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。
-ldb2 與資料庫管理程式檔案庫鏈結。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c 建置 DB2 API 非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqc 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果會產生一個可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

sqllib/samples/c 中的 script 檔 bldsrv 含有建置儲存程序的指令。Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2，會指定共用檔案庫進入點的儲存程序函數。第三個參數 $3 指定您要連接的資料庫名稱。因爲儲存程序必須建置在資料庫常駐的同一案例上，所以不需要任何使用者 ID 及通行碼的參數。

只有前兩個參數 (來源檔名稱及進入點) 是必要的。資料庫名稱是可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldsrv script file -- AIX
# Builds a C stored procedure
# Usage: bldsrv <prog_name> <entry_point> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $3

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the program.
xlc $CFLAGS_64 -I$DB2PATH/include -c $1.c

# Link the program using the export file $1.exp,
# creating shared library $1 with entry point $2.
```

```
xlc $CFLAGS_64 -o $1 $1.o -ldb2 -L$DB2PATH/lib -H512 -T512 -bE:$1.exp
-e $2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

xlc IBM C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。編譯及鏈結是不同的步驟。

鏈結選項：

xlc 使用編譯器作爲鏈結器的前端。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-o $1 將輸出檔指定爲共用檔案庫檔案。

$1.o 指定儲存程序目的檔。

-ldb2 與 DB2 檔案庫鏈結。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。

-H512 指定輸出檔對齊。

-T512 指定輸出檔文字區段的開始位址。

-bE:$1.exp
指定匯出檔。匯出檔含有儲存程序的列示。

-e $1 指定共用檔案庫的預設進入點。

請參閱您的編譯器文件，取得其他編譯器選項。

如果連接的是 sample 資料庫，則若欲從來源檔 spserver.sqc 建置範例程式 spserver，請輸入建置檔名稱、程式名稱及共用檔案庫進入點的儲存程序函數名稱：

```
bldsrv spserver outlanguage
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver outlanguage database
```

Script 檔會將儲存程序複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置共用檔案庫 spserver 完成，您即可建置存取共用檔案庫的從屬站應用程式 spclient。

您可以使用 script 檔 bldapp 建置 spclient。請參閱第119頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
spclient database userid password
```

其中

database
: 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample，或其別名，或其它的資料庫名稱。

userid 爲有效的使用者 ID。

password
: 爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

sqllib/samples/c 中的 script 檔 bldudf 含有建置 UDF 的指令。像儲存程序一樣地編譯 UDF。 它們無法含有 SQL 陳述式。這表示要建置 UDF 程式，您不能連接資料庫、前置編譯及連結程式。

第一個 $1 指定您的來源檔的名稱。第二個參數 $2，會指定共用檔案庫進入點的儲存程序函數。 Script 檔使用來源檔 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- AIX
# Builds a C UDF library
# Usage: bldudf <prog_name> <entry_point>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the program.
xlc $CFLAGS_64 -I$DB2PATH/include -c $1.c

# Link the program.
xlc $CFLAGS_64 -o $1 $1.o -ldb2 -ldb2apie -L$DB2PATH/lib -H512 -T512 -bE:$1.exp
-e $2
# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

xlc IBM C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

bldudf 的編譯及鏈結選項
鏈結選項： **xlc** 使用編譯器作爲鏈結器的前端。 **$CFLAGS_64** 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。 **-o $1** 將輸出檔指定爲共用檔案庫檔案。 **$1.o** 指定共用檔案庫目的檔。 **-ldb2** 與資料庫管理程式檔案庫鏈結。 **-ldb2apie** 與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。 **-L$DB2PATH/lib** 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。 **-H512** 指定輸出檔對齊。 **-T512** 指定輸出檔文字區段的開始位址。 **-bE:$1.exp** 指定匯出檔。匯出檔含有 UDF 的列示。 **-e $2** 指定共用檔案庫的預設進入點。 請參閱您的編譯器文件，取得其他編譯器選項。關於建立 UDF 的其它資訊，請參閱第112頁的『UDF 和 CREATE FUNCTION 陳述式』。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入建置檔名稱、程式名稱及共用檔案庫進入點的 UDF 函數：

```
bldudf udfsrv ScalarUDF
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。會提供此程式的 DB2 CLI 和內含的 SQL 版本。

您可以在 sqllib/samples/cli 中使用 script 檔 bldcli，從來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。關於詳細資訊，請參閱第113頁的『DB2 CLI 應用程式』。

您可以在 sqllib/samples/c 中使用 script 檔 bldapp，從來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。請參閱第119頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

欲呼叫 UDF，請輸入可執行檔名稱以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

必須編譯 AIX 版本 4 及 5 上的 C 多緒應用程式，並且鏈結 xlc_r 編譯器而不是 xlc 編譯器，或者鏈結 xlC_r 編譯器而不是 xlC 編譯器 (就 C++ 而言)。_r 版本會爲多緒編譯設定適當的前置處理器定義，並爲鏈結器提供適當的緒檔案庫名稱。

您可從編譯器文件中取得使用多緒編譯器前端設定編譯器及鏈結旗號的附加資訊。

在 sqllib/samples/c 中的 script 檔 bldmt 含有建置內含的 SQL 多緒程式的指令。第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。參數 $3 指定資料庫的使用者 ID，而 $4 則指定通行碼。 只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- AIX
# Builds a C multi-threaded embedded SQL program.
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the program.
xlc_r $CFLAGS_64 -I$DB2PATH/include -c $1.c
```

```
# Link the program.
xlc_r $CFLAGS_64 -o $1 $1.o -L$DB2PATH/lib -ldb2
```

除了前面討論的 xlc_r 編譯器以及剝有鏈結的公用程式檔這兩點以外，其它編譯及鏈結選項均和內含的 SQL 檔 bldapp 所使用的選項相同。關於這些選項的資訊，請參閱第119頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqc 建置多緒範例程式 thdsrver，請輸入下列指令：

```
bldmt thdsrver
```

結果會產生一個可執行檔 thdsrver。欲對 sample 資料庫執行可執行檔，請輸入可執行檔名稱：

```
thdsrver
```

IBM C Set++

本節包含下列主題：

- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)
- 多緒的應用程式

DB2 API 及內含的 SQL 應用程式

在 sqllib/samples/cpp 中的建置檔 bldapp，含有可建置 DB2 API 及內含 SQL 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- AIX
# Builds a C++ application program.
# Usage:  bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
```

```
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  xlC $CFLAGS_64 -I$DB2PATH/include -c utilemb.C
else
  # Compile the utilapi.C error-checking utility.
  xlC $CFLAGS_64 -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
xlC $CFLAGS_64 -I$DB2PATH/include -c $1.C

if [[ -f $1".sqC" ]]
then
  # Link the program with utilemb.o
  xlC $CFLAGS_64 -o $1 $1.o utilemb.o -ldb2 -L$DB2PATH/lib
else
  # Link the program with utilapi.o
  xlC $CFLAGS_64 -o $1 $1.o utilapi.o -ldb2 -L$DB2PATH/lib
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

xlC IBM C Set++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。編譯及鏈結是不同的步驟。

bldapp 的編譯及鏈結選項
鏈結選項： **xlC** 使用編譯器作爲鏈結器的前端。 **$CFLAGS_64** 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。 **-o $1** 指定可執行的程式。 **-o $1** 指定程式目的檔。 **utilapi.o** 包括非內含的 SQL 程式之 API 公用程式目的檔。 **utilemb.o** 包括內含的 SQL 程式之內含的 SQL 公用程式目的檔。 **-ldb2** 與資料庫管理程式檔案庫鏈結。 **-L$DB2PATH/lib** 指定 DB2 執行程式共用檔案庫的位置。例如：`$HOME/sqllib/lib`。若未指定 `-L` 選項，編譯器會假設下列路徑 `/usr/lib:/lib`。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.C` 建置非內含的 SQL 範例程式 `client`，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 `client`。 您可以輸入下面的指令，對 `Sample` 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqC` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果會產生一個可執行檔 `updat`。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』中，有關建置 C++ 儲存程序的資訊。

在 sqllib/samples/cpp 中的 script 檔 bldsrv，含有建置儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2，會指定共用檔案庫進入點的儲存程序函數。第三個參數 $3 指定您要連接的資料庫名稱。既然儲存程序必須建置在資料庫常駐的同一案例中，則您不需要使用者 ID 及通行碼的參數。

只有前兩個參數 (來源檔名稱及進入點) 是必要的。資料庫名稱是可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldsrv script file -- AIX
# Builds a C++ stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
  LFLAGS_64=-X64
else
  CFLAGS_64=
  LFLAGS_64=
fi
```

```
# Compile the program.
xlC $CFLAGS_64 -I$DB2PATH/include -c $1.C

# Link using export file $1.exp, creating shared library $1
makeC++SharedLib $LFLAGS_64 -p 1024 -o $1 $1.o -L$DB2PATH/lib -ldb2 -E $1.exp

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

xlC IBM C Set++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。編譯及鏈結是不同的步驟。

鏈結選項：

makeC++SharedLib
具有固定建構元的儲存程序的鏈結器 Script。

$LFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-X64" 值；否則，即不含任何值。

-p 1024
將優先順序設定爲任意值 1024。

-o $1 將輸出檔指定爲共用檔案庫檔案。

$1.o 指定程式目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。

-ldb2 與資料庫管理程式檔案庫鏈結。

-E $1.exp
指定匯出檔。匯出檔含有儲存程序的列示。

-e $2 指定共用檔案庫進入點。

請參閱您的編譯器文件，取得其他編譯器選項。

如果連接的是 sample 資料庫，如欲從來源檔 spserver.sqC 建置範例程式 spserver，則請輸入建置檔名稱、程式名稱及共用檔案庫進入點的儲存程序函數名稱：

```
bldsrv spserver outlanguage
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver outlanguage database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了共用檔案庫 spserver，您就可以建置呼叫檔案庫中儲存程序的從屬站應用程式 spclient。

您可以使用 script 檔 bldapp 建置 spclient。請參閱第128頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample，或其別名，或其它的資料庫名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數，然後將輸出傳回從屬站應用程式。

使用者定義函數 (UDF)

註: 關於建置 C++ UDF 的資訊，請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』。

在 sqllib/samples/cpp 中的 script 檔 bldudf 含有建置 UDF 的指令。UDF 無法包含內含的 SQL 陳述式。因此，要建置 UDF 程式，不需要連接資料庫。前置編譯及連結程式。

參數 $1 指定來源檔的名稱。參數 $2 指定共用檔案庫進入點的使用者定義的函數。 Script 檔使用來源檔 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- AIX
# Builds a C++ UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
  LFLAGS_64=-X64
else
  CFLAGS_64=
  LFLAGS_64=
fi

# Compile the program.
if [[ -f $1".c" ]]
then
  xlC $CFLAGS_64 -I$DB2PATH/include -c $1.c
elif [[ -f $1".C" ]]
then
  xlC $CFLAGS_64 -I$DB2PATH/include -c $1.C
fi

# Link using export file $1.exp, creating shared library $1
makeC++SharedLib $LFLAGS_64 -p 1024  -o $1 $1.o -L$DB2PATH/lib -ldb2
-ldb2apie -E $1.exp
```

```
# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

xlC IBM C Set++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-q64" 值；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

鏈結選項：

makeC++SharedLib
具有固定建構元的儲存程序的鏈結器 Script。

$LFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-X64" 值；否則，即不含任何值。

-p 1024
將優先順序設定爲任意值 1024。

-o $1 將輸出檔指定爲共用檔案庫檔案。

$1.o 指定程式目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。

-ldb2 與資料庫管理程式檔案庫鏈結。

-ldb2apie
與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。

-E $1.exp
指定匯出檔。匯出檔含有儲存程序的列示。

請參閱您的編譯器文件，取得其他編譯器選項。關於建立 UDF 的其它資訊，請參閱第112頁的『UDF 和 CREATE FUNCTION 陳述式』。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入建置檔名稱、程式名稱及共用檔案庫進入點的 UDF 函數：

```
bldudf udfsrv ScalarUDF
```

Script 檔會複製 UDF 到伺服器的路徑 sqllib/function 中。

必要時，設定 UDF 的檔案模式，以便 DB2 案例可以執行它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。您可以在 sqllib/samples/cpp 中使用 script 檔 bldapp，從來源檔 udfcli.sqC 建置 udfcli 程式。請參閱第128頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

欲呼叫 UDF，請輸入可執行檔名稱以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會呼叫 udfsrv 檔案庫中的 ScalarUDF 函數。

多緒的應用程式

必須編譯 AIX 版本 4 及 5 上的 C++ 多緒應用程式，並且鏈結 xlC_r 編譯器而不是 xlC 編譯器，或者鏈結 xlc_r 編譯器而不是 xlc 編譯器 (就 C 而言)。_r 版本會爲多緒編譯設定適當的前置處理器定義，並爲鏈結器提供適當的緒檔案庫名稱。

您可從編譯器文件中取得使用多緒編譯器前端設定編譯器及鏈結旗號的附加資訊。

在 sqllib/samples/cpp 中的 script 檔 bldmt，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。參數 $3 指定資料庫的使用者 ID，而 $4 則指定通行碼。 只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- AIX
# Builds a C++ multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true
```

```
if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-q64
else
  CFLAGS_64=
fi

# Compile the program.
xlC_r $CFLAGS_64 -I$DB2PATH/include -c $1.C

# Link the program.
xlC_r $CFLAGS_64 -o $1 $1.o -L$DB2PATH/lib -ldb2
```

除了前面討論的 xlC_r 編譯器以及沒有鏈結的這兩點公用程式檔以外，編譯及鏈結選項均與內含的 SQL script 檔 bldapp 中所使用的選項相同。關於這些選項的資訊，請參閱第128頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqC 建置多緒範例程式 thdsrver，請輸入下列指令：

```
bldmt thdsrver
```

結果會產生一個可執行檔 thdsrver。欲對 sample 資料庫執行可執行檔，請輸入可執行檔名稱：

```
thdsrver
```

VisualAge C++ 版本 4.0

此 VisualAge C++ 編譯器適用於 AIX、OS/2 及 Windows 32 位元作業系統。本節中的資訊適用於所有這些平台。

VisualAge C++ 編譯器與本書中述及的其它編譯器不同。若要使用 VisualAge C++ 版本 4.0 來編譯程式，您必須先製作一個架構檔。 請參閱附於編譯器的文件以瞭解如何製作架構檔。

DB2 爲您可使用 VisualAge C++ 編譯器建置的不同類型 DB2 程式提供一些架構檔。若要使用某 DB2 架構檔，您必須先對想要編譯的程式名稱設定環境變數。然後使用 VisualAge C++ 提供的指令編譯該程式。以下是 DB2 提供的架構檔，以及一些說明如何能夠使用這些架構檔來編譯程式的章節：

cli.icc
: DB2 CLI 架構檔。關於詳細資訊，請參閱第138頁的『DB2 CLI 應用程式』。

cliapi.icc
: DB2 CLI 與 DB2 API 架構檔。關於詳細資訊，請參閱第141頁的『DB2 CLI 應用程式與 DB2 API』。

clis.icc

DB2 CLI 儲存程序架構檔。關於詳細資訊，請參閱第142頁的『DB2 CLI 儲存程序』。

api.icc

DB2 API 架構檔。關於詳細資訊，請參閱第144頁的『DB2 API 應用程式』。

emb.icc

內含的 SQL 架構檔。關於詳細資訊，請參閱第146頁的『內含的 SQL 應用程式』。

stp.icc

內含的 SQL 儲存程序架構檔。關於詳細資訊，請參閱第148頁的『內含的 SQL 儲存程序』。

udf.icc

使用者定義的函數架構檔。關於詳細資訊，請參閱第151頁的『使用者定義函數 (UDF)』。

DB2 CLI 應用程式

架構檔 cli.icc，在 AIX上位於 的 sqllib/samples/cli 中，在 OS/2 及 Windows 32 位元作業系統上位於 %DB2PATH%\samples\cli 中，可讓您建置 DB2 CLI 程式。

```
// cli.icc configuration file for DB2 CLI applications
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export CLI=prog_name'
// To use on OS/2 and Windows, enter: 'set CLI=prog_name'
// Then compile the program by entering: 'vacbld cli.icc'

if defined( $CLI )
{
  prog_name = $CLI
}
else
{
  error "Environment Variable CLI is not defined."
}

infile = prog_name".c"
utilcli = "utilcli.c"

if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( libsearchpath, db2path"/lib" ),
```

```
                incl( searchPath, db2path"/include" )
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
{
  db2path     = $DB2PATH
  outfile     = prog_name".exe"
  group lib   = "db2cli.lib"
  option opts = link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
}

option opts
{
  target type(exe) outfile
  {
    source infile
    source utilcli
    source lib
  }
}
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

欲使用架構檔，從來源檔 tbinfo.c 建置 DB2 CLI 範例程式 tbinfo，請執行下列動作：

1. 輸入下列指令將 CLI 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export CLI=tbinfo
   ```

   在 OS/2 及 Windows 上：

   ```
   set CLI=tbinfo
   ```

2. 若工作目錄有 cli.ics 檔（使用 cli.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 cli.ics 檔：

   ```
   rm cli.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 cli.ics 檔。

3. 編譯範例程式，請輸入：

   ```
   vacbld cli.icc
   ```

   註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

結果會產生可執行檔 tbinfo。 您可以輸入可執行檔名稱，以執行該程式：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

在 AIX 上以 embprep 檔、在 OS/2 上以 embprep.cmd 檔或在 Windows 32 位元作業系統上以 embprep.bat 檔對程式進行前置編譯後，您可以使用架構檔。embprep 檔案會前置編譯來源檔，並連結程式到資料庫。在 AIX 上，您可以使用 cli.icc 架構檔來編譯經過前置編譯的檔案。在 OS/2 及 Windows 上，必須使用 cliapi.icc 檔案，而非 cli.icc 檔案，因爲內含的 SQL 應用程式需要 cliapi.icc 將 db2api.lib 檔案庫鏈入。

有三種方式可以從來源檔 dbusemx.sqc 前置編譯內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   embprep dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   embprep dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   embprep dbusemx database userid password
   ```

結果會產生經過前置編譯的 C 檔案 dbusemx.c。

前置編譯後，可以使用 AIX 上 cli.icc 檔案，及 OS/2 與 Windows 上的 cliapi.icc 檔案編譯 C 檔案，如下所示：

1. 輸入下列指令將 CLI 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export CLI=dbusemx
   ```

   在 OS/2 及 Windows 上：

   ```
   set CLIAPI=dbusemx
   ```

2. 如果您的工作目錄中存在 cli.ics 或 cliapi.ics 檔案 (使用cli.icc 或 cliapi.icc 檔案建置另一個程式來產生)，則請使用此指令刪除cli.ics 或 cliapi.ics 檔案：

   ```
   rm cli.ics
   ```

   或

   ```
   rm cliapi.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 cli.ics 或 cliapi.ics 檔案。

3. 編譯範例程式，請輸入：

   ```
   vacbld cli.icc
   ```

或

```
vacbld cliapi.icc
```

註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。 第24頁的表7 中含有 CLI 範例程式的說明，指出使用 DB2 API 的範例。架構檔 cliapi.icc 在 AIX 上位於 sqllib/samples/cli，在 OS/2 及 Windows 32 位元作業系統上位於 %DB2PATH%\samples\cli，可讓您建置 DB2 CLI 程式與 DB2 API。

此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 cli.icc 架構檔間唯一不同處。

欲從來源檔 dbmconn.c 建置 DB2 CLI 範例程式 dbmconn，請執行下列動作：

1. 輸入下列指令，將 CLIAPI 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export CLIAPI=dbmconn
   ```

   在 OS/2 及 Windows 上：

   ```
   set CLIAPI=dbmconn
   ```

2. 如果您在工作目錄中有 cliapi.ics 檔 (使用 cliapi.icc 檔建置另一個程式所產生的)，請使用下列指令刪除 cliapi.ics 檔：

   ```
   rm cliapi.ics
   ```

   現存的 cliapi.ics 檔 (將由您要再次建置的同一程式所產生的) 不必刪除。

3. 編譯範例程式，請輸入：

   ```
   vacbld cliapi.icc
   ```

註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

結果會產生可執行檔 dbmconn。您可以輸入可執行檔名稱，以執行該程式：

```
dbmconn
```

DB2 CLI 儲存程序

架構檔 clis.icc，在 AIX 上位於 sqllib/samples/cli，在 OS/2 及 Windows 32 位元作業系統上位於 %DB2PATH%\samples\cli，可讓您建置 DB2 CLI 儲存程序。

```
// clis.icc configuration file for DB2 CLI stored procedures
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export CLIS=prog_name'
// To use on OS/2 and Windows, enter: 'set CLIS=prog_name'
// Then compile the program by entering: 'vacbld clis.icc'

if defined( $CLIS )
{
  prog_name = $CLIS
}
else
{
  error "Environment Variable CLIS is not defined."
}

infile = prog_name".c"
utilcli = "utilcli.c"
expfile = prog_name".exp"

if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
  cpcmd       = "cp"
  funcdir     = db2path"/function"
}
else /* if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ ) */
{
  db2path     = $DB2PATH
  outfile     = prog_name".dll"
  if defined( $__TOS_WIN__ )
  {
    expfile = prog_name"v4.exp"
  }
  group lib   = "db2cli.lib"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
```

```
  cpcmd       = "copy"
  funcdir     = db2path"\\function"
}

option opts
{
  target type(dll) outfile
  {
    source infile
    source utilcli
    source lib
  }
}

if defined( $__TOS_AIX__ )
{
  rmcmd       = "rm -f"
  run after rmcmd " " funcdir "/" outfile
}

run after cpcmd " " outfile " " funcdir
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

欲使用架構檔，從來源檔 spserver.c 建置 DB2 CLI 儲存程序 spserver，請執行下列動作：

1. 輸入下列指令將 CLIS 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export CLIS=spserver
   ```

   在 OS/2 及 Windows 上：

   ```
   set CLIS=spserver
   ```

2. 若工作目錄有 clis.ics 檔（使用 clis.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 clis.ics 檔：

   ```
   rm clis.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 clis.ics 檔。

3. 編譯範例程式，請輸入：

   ```
   vacbld clis.icc
   ```

   註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

儲存程序會複製到伺服器，在 AIX 上會複製到路徑 sqllib/function，在 OS/2 及 Windows 32 位元作業系統上則到路徑 %DB2PATH%\function。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，以資料庫所在位置的案例使用者 ID 及通行碼連接資料庫：

```
db2 connect to sample userid password
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了儲存程序 spserver 後，您就可以建置呼叫儲存程序的 CLI 從屬站應用程式 spclient。您可以使用架構檔 cli.icc 建置 spclient。關於詳細資訊，請參閱第138頁的『DB2 CLI 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 應用程式

架構檔 api.icc 在 AIX 上位於 sqllib/samples/c 及 sqllib/samples/cpp，在 OS/2 及 Windows 32 位元作業系統上位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\c，可讓您以 C 或 C++ 建置 DB2 API 程式。

```
// api.icc configuration file for DB2 API programs
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export API=prog_name'
// To use on OS/2 and Windows, enter: 'set API=prog_name'
// Then compile the program by entering: 'vacbld api.icc'
```

```
if defined( $API )
{
  prog_name = $API
}
else
{
  error "Environment Variable API is not defined."
}

infile = prog_name".c"
util   = "utilapi.c"

if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
{
  db2path     = $DB2PATH
  outfile     = prog_name".exe"
  group lib   = "db2api.lib"
  option opts = link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
}

option opts
{
  target type(exe) outfile
  {
    source infile
    source util
    source lib
  }
}
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

若要使用架構檔，從來源檔 client.c 建置 DB2 API 範例程式 client，請執行下列動作：

1. 輸入下列指令將 API 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export API=client
   ```

在 OS/2 及 Windows 上：

```
set API=client
```

2. 若工作目錄有 api.ics 檔（使用 api.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 api.ics 檔：

   ```
   rm api.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 api.ics 檔。

3. 編譯範例程式，請輸入：

   ```
   vacbld api.icc
   ```

   註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

結果會產生一個可執行檔 client。 您可以輸入可執行檔名稱，以執行該程式：

```
client
```

內含的 SQL 應用程式

架構檔 emb.icc 在 AIX 中位於 sqllib/samples/c 及 sqllib/samples/cpp，在 OS/2 及 Windows 32 位元作業系統中位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，可讓您以 C 或 C++ 建置 DB2 內含的 SQL 應用程式。

```
// emb.icc configuration file for embedded SQL applications
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export EMB=prog_name'
// To use on OS/2 and Windows, enter: 'set EMB=prog_name'
// Then compile the program by entering: 'vacbld emb.icc'

if defined( $EMB )
{
  prog_name = $EMB
}
else
{
  error "Environment Variable EMB is not defined."
}

// To connect to another database, replace "sample"
// For user ID and password, update 'user' and 'passwd'
// and take out the comment in the line: 'run before "embprep "'
dbname = "sample"
user   = ""
passwd = ""

// Precompiling the source program file
run before "embprep " prog_name " " dbname // " " user " " passwd

infile = prog_name".c"
util = "utilemb.sqc"
```

```
if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
{
  db2path     = $DB2PATH
  outfile     = prog_name".exe"
  group lib   = "db2api.lib"
  option opts = link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
}

option opts
{
  target type(exe) outfile
  {
    source infile
    source util
    source lib
  }
}
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

若要使用架構檔，從來源檔 updat.sqc 建置內含的 SQL 應用程式 updat，請執行下列動作：

1. 輸入下列指令將 EMB 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export EMB=updat
   ```

   在 OS/2 及 Windows 上：

   ```
   set EMB=updat
   ```

2. 若工作目錄有 emb.ics 檔（使用 emb.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 emb.ics 檔：

   ```
   rm emb.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 emb.ics 檔。

3. 編譯範例程式，請輸入：

   ```
   vacbld emb.icc
   ```

註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

結果會產生一個可執行檔 updat。 您可以輸入可執行檔名稱，以執行該程式：

```
updat
```

內含的 SQL 儲存程序

架構檔 stp.icc 在 AIX 中位於 sqllib/samples/c 及 sqllib/samples/cpp，在 OS/2 及 Windows 32 位元作業系統中位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，可讓您以 C 及 C++ 建置 DB2 內含的 SQL 儲存程序。

```
// stp.icc configuration file for embedded SQL stored procedures
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export STP=prog_name'
// To use on OS/2 and Windows, enter: 'set STP=prog_name'
// Then compile the program by entering: 'vacbld emb.icc'

if defined( $STP )
{
  prog_name = $STP
}
else
{
  error "Environment Variable STP is not defined."
}

// To connect to another database, replace "sample"
// For user ID and password, update 'user' and 'passwd'
// and take out the comment in the line: 'run before "embprep "'
dbname = "sample"
user   = ""
passwd = ""

// Precompiling the source program file
run before "embprep " prog_name " " dbname // " " user " " passwd

infile = prog_name".c"
expfile = prog_name".exp"

if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
  cpcmd       = "cp"
  funcdir     = db2path"/function"
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
```

```
{
  db2path     = $DB2PATH
  outfile     = prog_name".dll"
  if defined( $__TOS_WIN__ )
  {
    expfile = prog_name"v4.exp"
  }
  group lib   = "db2api.lib"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
  cpcmd       = "copy"
  funcdir     = db2path"\\function"
}

option opts
{
  target type(dll) outfile
  {
    source infile
    source lib
  }
}

if defined( $__TOS_AIX__ )
{
  rmcmd       = "rm -f"
  run after rmcmd " " funcdir "/" outfile
}

run after cpcmd " " outfile " " funcdir
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

欲使用架構檔，從來源檔 spserver.sqc 建置內含的 SQL 儲存程序 spserver，請執行下列動作：

1. 輸入下列指令將 STP 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export STP=spserver
   ```

   在 OS/2 及 Windows 上：

   ```
   set STP=spserver
   ```

2. 若工作目錄有 stp.ics 檔（使用 stp.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 stp.ics 檔：

   ```
   rm stp.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 stp.ics 檔。

3. 編譯範例程式，請輸入：

```
vacbld stp.icc
```

註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

儲存程序會複製到伺服器，在 AIX 上會複製到路徑 sqllib/function，在 OS/2 及 Windows 32 位元作業系統上則到路徑 %DB2PATH%\function。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了儲存程序 spserver 後，您就可以建置從屬站應用程式 spclient，呼叫儲存程序。您可以使用架構檔 emb.icc 建置 spclient。詳細資訊，請參閱第146頁的『內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

架構檔 udf.icc，在 AIX 中位於 sqllib/samples/c 及 sqllib/samples/cpp，在 OS/2 及 Windows 32 位元作業系統中位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，可讓您以 C 及 C++ 建置使用者定義函數。

```
// udf.icc configuration file for user-defined functions
// for VisualAge C++ Version 4.0
// To use on AIX, enter: 'export UDF=prog_name'
// To use on OS/2 and Windows, enter: 'set UDF=prog_name'
// Then compile the program by entering: 'vacbld udf.icc'

if defined( $UDF )
{
  prog_name = $UDF
}
else
{
  error "Environment Variable UDF is not defined."
}

infile = prog_name".c"
expfile = prog_name".exp"

if defined( $__TOS_AIX__ )
{
  // Set db2path to where DB2 will be accessed.
  // The default is the standard instance path.
  db2path     = $HOME"/sqllib"
  outfile     = prog_name
  group lib   = "libdb2.a", "libdb2apie.a"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"/lib" ),
                incl( searchPath, db2path"/include" )
  cpcmd       = "cp"
  funcdir     = db2path"/function"
}
else // if defined( $__TOS_OS2__ ) | defined( $__TOS_WIN__ )
{
  db2path     = $DB2PATH
  outfile     = prog_name".dll"
  if defined( $__TOS_WIN__ )
  {
    expfile = prog_name"v4.exp"
  }
  group lib   = "db2api.lib", "db2apie.lib"
  option opts = link( exportList, expfile ),
                link( libsearchpath, db2path"\\lib" ),
                incl( searchPath, db2path"\\include" )
  cpcmd       = "copy"
  funcdir     = db2path"\\function"
}

option opts
{
```

```
  target type(dll) outfile
  {
    source infile
    source lib
  }
}
if defined( $__TOS_AIX__ )
{
  rmcmd       = "rm -f"
  run after rmcmd " " funcdir "/" outfile
}

run after cpcmd " " outfile " " funcdir
```

VisualAge C++ 版本 4.0 會依據安裝的作業系統，定義下列其中一項環境變數：__TOS_AIX__、__TOS_OS2__、 __TOS_WIN__。

欲使用架構檔，從來源檔 udf.c 建置使用者定義的函數程式 udfsrv，請執行下列動作：

1. 輸入下列指令將 UDF 環境變數設定爲程式名稱：

   在 AIX 上：

   ```
   export UDF=udfsrv
   ```

   在 OS/2 及 Windows 上：

   ```
   set UDF=udfsrv
   ```

2. 若工作目錄有 udf.ics 檔（使用 udf.icc 檔建置另一個程式產生該檔案），請使用此指令刪除 udf.ics 檔：

   ```
   rm udf.ics
   ```

   不必刪除針對您再次建置的相同程式而產生的現存 udf.ics 檔。

3. 編譯範例程式，請輸入：

   ```
   vacbld udf.icc
   ```

   註: VisualAge C++ 版本 4.0 提供 vacbld 指令。

UDF 檔案庫會複製到路徑 sqllib/function 中的伺服器。

必要時請設定使用者定義的函數的檔案模式，使 DB2 案例能執行它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。會提供此程式的 DB2 CLI 和內含的 SQL 版本。

您可以使用架構檔 cli.icc，從來源檔 udfcli.c (在 AIX 中位於 sqllib/samples/cli，在 OS/2 及 Windows 32 位元作業系統中位於 %DB2PATH%\samples\cli) 建置 DB2 CLI udfcli 程式。關於詳細資訊，請參閱第138頁的『DB2 CLI 應用程式』。

您可以使用架構檔 emb.icc，從來源檔 udfcli.sqc (在 AIX 中位於 sqllib/samples/c，在 OS/2 及 Windows 32 位元作業系統中位於 %DB2PATH%\samples\cli) 建置內含的 SQL udfcli 程式。詳細資訊，請參閱第146頁的『內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入可執行檔名稱以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

IBM COBOL Set for AIX

本節包含下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

如果您想要開發內含 SQL 及 DB2 API 呼叫的應用程式，而且您將使用 IBM COBOL Set for AIX 編譯器，請牢記下列幾點：

- 當您使用命令行處理器指令 db2 prep，來前置編譯您的應用程式時，請使用 target ibmcob 選項。
- 請勿在您的來源檔中使用 Tab 鍵字元。
- 您可以在您的來源檔的第一行中，使用 PROCESS 及 CBL 關鍵字，來設定編譯選項。
- 如果您的應用程式僅內含 SQL，且沒有任何 DB2 API 呼叫，則您不需要使用 pgmname(mixed) 編譯選項。如果您使用 DB2 API 呼叫，則您必須使用 pgmname(mixed) 編譯選項。
- 如果您使用的是 IBM COBOL Set for AIX 編譯器的「System/390 主電腦資料類型支援」功能，則應用程式的 DB2 併入檔會位於下列目錄：

  ```
  $HOME/sqllib/include/cobol_i
  ```

  若使用提供的 Script 檔建置 DB2 範例程式，必須變更在這些 Script 檔中指定的併入檔路徑以指向 cobol_i 目錄而不是 cobol_a 目錄。

若不是使用 IBM COBOL Set for AIX 編譯器的「System/390 主電腦資料類型支援」特性，或使用此編譯器的舊版本，則應用程式的 DB2 併入檔會位於下列目錄：

```
$HOME/sqllib/include/cobol_a
```

指定 COPY 檔名，來併入 .cbl 副檔名，方式如下：

```
COPY "sql.cbl".
```

DB2 API 及內含的 SQL 應用程式

在 sqllib/samples/cobol 中的建置檔 bldapp，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- AIX
# Builds an IBM COBOL application program
# Usage:  bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqb" ]]
then
  embprep $1 $2 $3 $4
fi

# Compile the checkerr.cbl error checking utility.
cob2 -qpgmname\(mixed\) -qlib -I$DB2PATH/include/cobol_a \
     -c checkerr.cbl

# Compile the program.
cob2 -qpgmname\(mixed\) -qlib -I$DB2PATH/include/cobol_a \
     -c $1.cbl

# Link the program.
cob2 -o $1 $1.o checkerr.o -ldb2 -L$DB2PATH/lib
```

bldapp 的編譯及鏈結選項
編譯選項：
cob2 IBM COBOL Set 編譯器
-qpgmname\(mixed\) 指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。
-qlib 指示編譯器處理 COPY 陳述式。
-I$DB2PATH/include/cobol_a 指定 DB2 併入檔的位置。例如：$HOME/sqllib/include/cobol_a。
-c 僅執行編譯；沒有任何鏈結。編譯及鏈結是不同的步驟。
鏈結選項：
cob2 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行的程式。
$1.o 指定程式目的檔。
checkerr.o 包括錯誤檢查的公用程式目的檔。
-ldb2 與資料庫管理程式檔案庫鏈結。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.cbl 建置非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生一個可執行檔 client。您可以輸入下面的指令，對 Sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果會產生一個可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

在 sqllib/samples/cobol 中的 script 檔 bldsrv，含有建置儲存程序的指令。Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的相同案例中，所以不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

Script 檔使用來源檔名稱 $1 作爲共用檔案庫名稱和共用檔案庫的進入點。若所建置的儲存程序的進入點函數名稱與來源檔名稱不同，您可修改 Script 檔以接受進入點的另一個參數。建議您將資料庫參數更名爲 $3。然後您可將進入點鏈結選項變更爲 -e $2，而且在執行 Script 檔時在命令行上指定其它參數。

```
#! /bin/ksh
# bldsrv script file -- AIX
# Builds an IBM COBOL stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the checkerr.cbl error checking utility.
cob2 -qpgmname\(mixed\) -qlib -I$DB2PATH/include/cobol_a \
```

```
     -c checkerr.cbl

# Compile the program.
cob2 -qpgmname\(mixed\) -qlib -c -I$DB2PATH/include/cobol_a $1.cbl

# Link the program using the export file $1.exp
# creating shared library $1 with entry point $1.
cob2 -o $1 $1.o checkerr.o -H512 -T512 -e $1 -bE:$1.exp \
     -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory of the DB2
instance.
# This assumes the user has write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

cob2 IBM COBOL Set 編譯器

-qpgmname\(mixed\)
指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。

-qlib 指示編譯器處理 COPY 陳述式。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-I$DB2PATH/include/cobol_a
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include/cobol_a。

bldsrv 的編譯及鏈結選項
鏈結選項：
cob2 使用編譯器來鏈結編輯。
-o $1 將輸出檔指定爲共用檔案庫檔案。
$1.o 指定儲存程序目的檔。
checkerr.o 包括錯誤檢查的公用程式目的檔。
-H512 指定輸出檔對齊。
-T512 指定輸出檔文字區段的開始位址。
-e $1 指定共用檔案庫的預設進入點。
-bE:$1.exp 指定匯出檔。匯出檔含有儲存程序的列示。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
-ldb2 與資料庫管理程式檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 outsrv.sqb 建置範例程式 outsrv，並連接範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接的是另一個資料庫，亦請併入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將儲存程序複製到伺服器的路徑 sqllib/function 中。

必要時，請設定儲存程序的檔案模式，以便從屬站應用程式可以存取它。

一旦您開發儲存程序 outsrv，您就可以開發從屬站應用程式 outcli，來呼叫儲存程序。您可以使用 script 檔 bldapp 建置 outcli。請參閱第154頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 outsrv，並在伺服器資料庫中執行名稱相同的儲存程序函數，然後將輸出傳回從屬站應用程式。

Micro Focus COBOL

本節包含下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

如果您想要開發內含 SQL 及 DB2 API 呼叫的應用程式，而且您將使用 Micro Focus COBOL 編譯器，請牢記下列幾點：

- 當您使用命令行處理器指令 db2 prep，來前置編譯您的應用程式時，請使用 target mfcob 選項 (預設值)。
- 若要使用內建式前置編譯器前端、執行時直譯器或 Animator 除錯器 (具有 Micro Focus COBOL 版本 4.0 及 4.1)，請依下列方式執行 Micro Focus 提供的 mkrts 指令，將 DB2 Generic API 進入點新增至 Micro Focus 執行時模組 rts32：
  1. 以 root 身份登入。
  2. 使用下列目錄提供的引數來執行 mkrts：

     ```
     /usr/lpp/db2_07_01/lib/db2mkrts.args
     ```

  註: 上述進入點不套用至 Micro Focus Server Express。
- 您必須在 Micro Focus COBOL 環境變數 COBCPY 中，併入 DB2 COBOL COPY 檔案目錄。COBCPY 環境變數會指定 COPY 檔的位置。Micro Focus COBOL 的 DB2 COPY 檔常駐在資料庫案例目錄下的 sqllib/include/cobol_mf 中。

  若要加入此目錄，請輸入下列指令：

  ```
  export COBCPY=$COBCPY:$HOME/sqllib/include/cobol_mf
  ```

  註: 您可能想要在 .profile 檔中設定 COBCPY。

DB2 API 及內含的 SQL 應用程式

在 sqllib/samples/cobol_mf 中的建置檔 bldapp，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- AIX
# Builds a Micro Focus COBOL application program
# Usage:  bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqb" ]]
then
  embprep $1 $2 $3 $4
fi

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$DB2PATH/include/cobol_mf:$COBCPY

# Compile the checkerr.cbl error checking utility.
cob -c -x checkerr.cbl

# Compile the program.
cob -c -x $1.cbl

# Link the program.
cob -x -o $1 $1.o checkerr.o -ldb2 -ldb2gmf -L$DB2PATH/lib
```

bldmfapi 的編譯選項和鏈結選項	
編譯選項：	
cob	COBOL 編譯器。
-c	僅執行編譯；沒有任何鏈結。
-x	產生可執行程式。

bldmfapi 的編譯選項和鏈結選項
鏈結選項：
cob 使用編譯器作爲鏈結器的前端。
-x 產生可執行程式。
-o $1 指定可執行的程式。
$1.o 指定程式目的檔。
-ldb2 鏈結 DB2 檔案庫。
-ldb2gmf 鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.cbl 建置非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生一個可執行檔 client。您可以輸入下面的指令，對 Sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果會產生一個可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註:

1. 在 AIX 4.2.1 上，使用 Micro Focus 4.1 編譯器建置儲存程序之前，請執行下列指令：

   ```
   db2stop
   db2set DB2LIBPATH=$LIBPATH
   db2set DB2ENVLIST="COBDIR LIBPATH"
   db2set
db2start
   ```

   請確定 db2stop 會停止資料庫，且LIBPATH 已在您的 shell 環境中正確地設定。最後一個發出的 db2set 指令可顯示您的設定值：請確定 DB2LIBPATH 及 DB2ENVLIST 已正確地設定。

2. 在 AIX 版本 4 平台上使用的部份較新版 Micro Focus COBOL 編譯器，無法用來建立靜態鏈結的儲存程序。因此，make 檔和 Script 檔 bldsrv，已調適爲容許建立動態鏈結的儲存程序。

   爲了使遠端從屬站應用程式能順利呼叫此動態鏈結的儲存程序，必須在儲存程序所在的伺服器上呼叫 Micro Focus COBOL 常式 cobinit() 之後，才能執行儲存程序。執行 make 檔或 script 檔 bldsrv 時，即建立可完成上述動作的外層程式（wrapper program）。然後再鏈結儲存程序碼形成儲存程序共用檔案庫。由於使用此外層程式的關係，爲了讓從屬站應用程式能夠呼叫儲存程序 x，它必須呼叫 x_wrap 而不是 x。

   本節稍後將說明外層程式的明細。

在 sqllib/samples/cobol_mf 中的 script 檔 bldsrv，含有建置儲存程序的指令。Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的相同案例中，所以不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。如果沒有提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

Script 檔使用來源檔名稱 $1 作爲共用檔案庫名稱和共用檔案庫的進入點。若所建置的儲存程序的進入點函數名稱與來源檔名稱不同，您可修改 Script 檔以接受進入點的另一個參數。建議您將資料庫參數更名爲 $3。然後您可將進入點鏈結選項變更爲 -e $2，而且在執行 Script 檔時在命令行上指定其它參數。

```
#! /bin/ksh
# bldsrv script file -- AIX
# Builds a Micro Focus COBOL stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$DB2PATH/include/cobol_mf:$COBCPY

# Compile the program.
cob -c -x $1.cbl

# Create the wrapper program for the stored procedure.
wrapsrv $1

# Link the program using export file ${1}_wrap.exp
# creating shared library $1 with entry point ${1}_wrap.
cob -x -o $1 ${1}_wrap.c $1.o -Q -bE:${1}_wrap.exp -Q "-e $1" \
-Q -bI:$DB2PATH/lib/db2g.imp -ldb2gmf -L$DB2PATH/lib

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

cob	COBOL 編譯器。
-c	僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
-x	產生可執行程式。

bldsrv 的編譯及鏈結選項

鏈結選項：

cob　使用編譯器來鏈結編輯。

-x　產生可執行程式。

-o $1　指定可執行的程式。

-o ${1}_wrap.c
指定外層程式。

$1.o　指定程式目的檔。

-Q -bE:${1}_wrap.exp
指定匯出檔。匯出檔含有儲存程序進入點列示。如果儲存程序稱爲 x，則它的進入點是 x_wrap。

-Q "-e $1"
指定共用檔案庫的預設進入點。

-Q -bI:$DB2PATH/lib/db2g.imp
提供 DB2 應用程式檔案庫的進入點的列示。

-ldb2gmf
鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。

請參閱您的編譯器文件，取得其他編譯器選項。

外層程式 wrapsrv 會導致在執行儲存程序前，先呼叫 Micro Focus COBOL 常式 cobinit()。它的內容如下。

```
#! /bin/ksh
# wrapsrv script file
# Creates the wrapper program for Micro Focus COBOL stored procedures
# Usage: wrapsrv <stored_proc>

# Note: The client program calls "<stored_proc>_wrap" not "<stored_proc>"

# Create the wrapper program for the stored procedure.
cat << WRAPPER_CODE > ${1}_wrap.c
#include <stdio.h>
void cobinit(void);
int $1(void *p0, void *p1, void *p2, void *p3);

int main(void)
{
  return 0;
}

int ${1}_wrap(void *p0, void *p1, void *p2, void *p3)
{
  cobinit();
  return $1(p0, p1, p2, p3);
}
WRAPPER_CODE
# Create the export file for the wrapper program
echo $1_wrap > ${1}_wrap.exp
```

欲從來源檔 outsrv.sqb 建置範例程式 outsrv，如果是連接到範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

必要時，請設定共用檔案庫的檔案模式，以便從屬站應用程式可以存取它。

一旦建置了儲存程序 outsrv，您就可以建置從屬站應用程式 outcli，呼叫該儲存程序。您可以使用 script 檔 bldapp 建置 outcli。請參閱第160頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可以存取共用檔案庫 outsrv，並在伺服器資料庫中執行名稱相同的儲存程序函數。然後將輸出傳回從屬站應用程式。

結束儲存程序

開發儲存程序時，請使用下列陳述式來結束儲存程序：

```
move SQLZ-HOLD-PROC to return-code.
```

透過這個陳述式，儲存程序即會正確地傳回至從屬站應用程式。 當本端 COBOL 從屬站應用程式呼叫儲存程序時，這點特別重要。

REXX

您不必前置編譯或連結 REXX 程式。

欲在 AIX 上執行 DB2 REXX/SQL 程式，您必須設定 LIBPATH 環境變數，來包括 DB2 安裝目錄下的 sqllib/lib。

輸入：

```
export LIBPATH=$LIBPATH:/lib:/usr/lib:/usr/lpp/db2_07_01/sqllib/lib
```

在 AIX 上，您的應用程式檔案可以具有任何副檔名。您可以使用下列兩種方法之一，來執行您的應用程式：

1. 在 Shell 命令提示字元中，輸入 rexx *名稱*，其中*名稱*爲您的 REXX 程式的名稱。
2. 如果您的 REXX 程式的第一行含有 "魔術數字" (#!)，並識別 REXX/6000 直譯器常駐的目錄，則您可以經由在 Shell 命令提示字元中輸入您的 REXX 程式，來執行它。例如，如果 REXX/6000 直譯器檔案位在 /usr/bin 目錄中，併入下列作爲您的 REXX 程式的第一行：

   ```
   #! /usr/bin/rexx
   ```

   然後，經由在 Shell 命令提示字元中輸入下列指令，使程式成爲可執行檔：

   ```
   chmod +x name
   ```

   經由在 Shell 命令提示字元中，輸入您的 REXX 程式的檔名來執行它。

REXX 範例程式位在目錄 sqllib/samples/rexx 中。欲執行 REXX 範例程式 updat.cmd，請執行下列一項：

- 將 "#! /usr/bin/rexx" 這一行新增至程式來源檔頂端（若這一行不在該檔案頂端的話）。然後輸入下列指令直接執行程式：

  ```
  updat.cmd
  ```

- 輸入下列指令指定 REXX 直譯器和程式：

  ```
  rexx updat.cmd
  ```

關於 REXX 和 DB2 的進一步資訊，請參閱 *Application Development Guide* 的 ″Programming in REXX″ 這一章。

第7章 建置 HP-UX 應用程式

本章提供如何在 HP-UX 中建置 DB2 應用程式的詳細資訊。在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。 如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

關於 HP-UX 的最新 DB2 應用程式開發更新組件，請造訪下列 DB2 應用程式開發網頁：

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

註:

1. +DAportable 選項是用在 DB2 建置檔及 make 檔的編譯及鏈結步驟中。此選項會產生在 PA_RISC 1.1 及 2.0 工作站與伺服器中可相容的程式碼。使用此選項，會付出輕微的效能成本。欲增進效能，您可以從 sqllib/samples 目錄的建置檔及 make 檔中除去 +DAportable 選項。沒有了此選項，在建置 HP-UX 程式時，您會得到一個類似下列的警告：

   ```
   (Warning) At least one PA 2.0 object file (<filename>.o) was detected.
   The linked object may not run on a PA 1.x system.
   ```

   其中 <filename> 是正在編譯的程式檔。

   除非您有 PA_RISC 1.1 或 2.0 系統，否則不會顯示此警告。

2. 如果您是從 HP-UX 版本 10 或更早的版本將 DB2 移轉到 HP-UX 版本 11，您的 DB2 程式必須以 HP-UX 版本 11 上的 DB2 重新執行前置編譯 (如果它們併入內含的 SQL)，且必須重新編譯。包括所有 DB2 應用程式、儲存程序、使用者定義函數及使用者跳出程式都需進行此項作業。並且，在 HP-UX 版

本 11 上編譯的 DB2 程式不能在 HP-UX 版本 10 或更早版本上執行。在 HP-UX 版本 10 編譯及執行的 DB2 程式可從遠端連接到 HP-UX 版本 11 伺服器。

3. 對於 64 位元的應用程式，請在範例目錄中使用 64 位元建置 Script。在該處，Script 與 32 位元 Script 同名，只是將 ″64″ 新增到 Script 名稱的結尾。

HP-UX C

本節包含下列主題：

- DB2 CLI 應用程式
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)
- 多緒的應用程式

DB2 CLI 應用程式

sqllib/samples/cli 中的 Script 檔 bldcli 含有一些建置 DB2 CLI 程式的指令。

註: 對於 64 位元 DB2 CLI 程式，請使用 bldcli64 Script。

參數 $1 指定來源檔名稱。 這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名指示，則會呼叫 embprep script 來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- HP-UX
# Builds a CLI program with HP-UX C.
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]
# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
fi

# Compile the error-checking utility.
cc +DAportable -Aa +e -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc +DAportable -Aa +e -I$DB2PATH/include -c $1.c

# Link the program.
cc +DAportable -o $1 $1.o utilcli.o -L$DB2PATH/lib -ldb2
```

bldcli 的編譯選項和鏈結選項

編譯選項：

cc 使用 C 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

-Aa 使用 ANSI 標準模式。

+e 當在 ANSI C 模式中進行編譯時，啓用 HP 附加價值特性。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：`$HOME/sqllib/include`

-c 僅執行編譯；沒有任何鏈結。 編譯及鏈結是不同的步驟。

bldcli 的編譯選項和鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
+DAportable 使用在 PA_RISC 1.x 及 2.0 工作站與伺服器中相容的的程式碼。
-o $1 指定可執行的程式。
-o $1.o 指定目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。 例如，$HOME/sqllib/lib
-ldb2 與資料庫管理程式檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式tbinfo，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 tbinfo。您可輸入這個可執行檔名稱來執行此可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。

註: 對於具有 DB2 API 的 64 位元 DB2 CLI 程式，請使用 bldapi64 Script。

bldapi 檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。 這是此檔案與 bldcli script 間的唯一不同處。 請參閱第170頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果產生可執行檔 dbmconn。您可輸入這個可執行檔名稱來執行此可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

sqllib/samples/cli 中的 Script 檔 bldclisp 含有一些建置 DB2 CLI 儲存程序的指令。

註: 對於 64 位元的 DB2 CLI 儲存程序，請使用 bldclisp64 Script。

參數 $1 指定來源檔名稱。 只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldclisp script file -- HP-UX
# Builds a CLI stored procedure in HP-UX C.
# Usage: bldclisp <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the error-checking utility.
cc +DAportable +u1 +z -Aa +e -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc +DAportable +u1 +z -Aa +e -I$DB2PATH/include -c $1.c

# Link the program.
ld -b -o $1 $1.o utilcli.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldclisp 的編譯選項和鏈結選項

編譯選項：

cc C 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

+u1 容許未調整的資料存取。僅在您使用未調整的資料時才使用。

+z 產生與位置無關的程式碼。

-Aa 使用 ANSI 標準模式 (僅適用於 C 編譯器)。

+e 當在 ANSI C 模式中進行編譯時，啓用 HP 附加價值特性。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：`$HOME/sqllib/include`

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

鏈結選項：

ld 使用鏈結器來鏈結編輯。

-b 建立共用檔案庫，而非正常的可執行檔。

-o $1 指定可執行檔。

$1.o 指定目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： `-L$HOME/sqllib/lib`。如果您未指定 `-L` 選項，將使用 `/usr/lib:/lib`。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldclisp spserver
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置存取共用檔案庫的 CLI 從屬站應用程式 spclient。

您可以使用 script 檔 bldcli 建置 spclient。詳細資訊，請參閱第170頁的『DB2 CLI 應用程式』。

欲存取共用檔案庫，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

term>
: 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
: 爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver 並在伺服器資料庫中執行許多儲存程序函數，然後將輸出傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp 位於 sqllib/samples/c 中，含有建置 DB2 應用程式的指令。

註: 對於 64 位元的 DB2 應用程式，請使用 bldapp64 Script。

第一個參數 $1 指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- HP-UX
# Builds a C application program
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.c error-checking utility.
  cc +DAportable -Aa +e -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  cc +DAportable -Aa +e -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc +DAportable -Aa +e -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  cc +DAportable -o $1 $1.o utilemb.o -L$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o
  cc +DAportable -o $1 $1.o utilapi.o -L$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項
編譯選項： **cc** C 編譯器。 **+DAportable** 產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。 **-Aa** 使用 ANSI 標準模式 (僅適用於 C 編譯器)。 **+e** 當在 ANSI C 模式中進行編譯時，啓用 HP 附加價值特性。 **-I$DB2PATH/include** 指定 DB2 併入檔的位置。 例如：-I$DB2PATH/include **-c** 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
鏈結選項： **cc** 使用編譯器來鏈結編輯。 **+DAportable** 使用在 PA_RISC 1.x 及 2.0 工作站與伺服器中相容的的程式碼。 **-o $1** 指定可執行檔。 **$1.o** 指定程式目的檔。 **utilemb.o** 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。 **utilapi.o** 如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。 **-L$DB2PATH/lib** 指定 DB2 執行程式共用檔案庫的位置。 例如： -L$DB2PATH/lib。 如果您未指定 -L 選項，將使用 /usr/lib:/lib。 **-ldb2** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c 建置 DB2 API 非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生可執行檔 client。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqc 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

```
bldapp updat database
```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

```
bldapp updat database userid password
```

結果產生可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

```
updat
```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

```
updat database
```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

```
updat database userid password
```

內含的 SQL 儲存程序

Script 檔 bldsrv，位於 sqllib/samples/c 中，含有可建置內含的 SQL 儲存程序的指令。

註: 對於 64 位元內含的 SQL 儲存程序，請使用 bldsrv64 Script。

bldsrv Script 會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。 既然儲存程序必須在資料庫常駐的同一案例中建置，您就不需要指定使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldsrv script file -- HP-UX
# Builds a C stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the program.
```

```
cc +DAportable +u1 +z -Aa +e -I$DB2PATH/include -c $1.c

# Link the program to create a shared library
ld -b -o $1 $1.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

+u1 容許未調整的資料存取。僅在您使用未調整的資料時才使用。

-Aa 使用 ANSI 標準模式 (僅適用於 C 編譯器)。

+z 產生與位置無關的程式碼。

+e 當在 ANSI C 模式中進行編譯時，啓用 HP 附加價值特性。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：-I$DB2PATH/include。

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

鏈結選項：

ld 使用鏈結器來鏈結編輯。

-b 建立共用檔案庫，而非正常的可執行檔。

-o $1 將輸出檔指定爲共用檔案庫檔案。

$1.o 指定程式目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： $HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqc 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient 以存取該檔案庫。

您可以使用 script 檔 bldapp 建置 spclient。請參閱第176頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

term>
: 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
: 爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

在 sqllib/samples/c 中的 script 檔 bldudf 含有建置 UDF 的指令。

註: 對於 64 位元的 UDF，請使用 bldudf64 Script。

像儲存程序一樣地編譯 UDF。但是，它們不能包含內含的 SQL 陳述式。這表示若要建置 UDF 程式並不需要連接資料庫、前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- HP-UX
# Builds a C UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
cc +DAportable +u1 +z -Aa +e -I$DB2PATH/include -c $1.c

# Link the program and create a shared library.
ld -b -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

+u1 容許未調整的資料存取。僅在您使用未調整的資料時才使用。

-Aa 使用 ANSI 標準模式 (僅適用於 C 編譯器)。

+z 產生與位置無關的程式碼。

+e 當在 ANSI C 模式中進行編譯時，啓用 HP 附加價值特性。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldudf 的編譯及鏈結選項

鏈結選項：
ld　使用鏈結器來鏈結編輯。
-b　建立共用檔案庫，而非正常的可執行檔。
-o $1　將輸出檔指定爲共用檔案庫檔案。
$1.o　指定程式目的檔。
-L$DB2PATH/lib
　指定 DB2 執行程式共用檔案庫的位置。 例如： $HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2　與 DB2 檔案庫鏈結。
-ldb2apie
　與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。會提供此程式的 DB2 CLI 和內含的 SQL 版本。

您可以在 sqllib/samples/cli 中使用 script 檔 bldcli，從來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。詳細資訊，請參閱第170頁的『DB2 CLI 應用程式』。

您可以在 sqllib/samples/c 中使用 script 檔 bldapp，從來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。請參閱第176頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫 UDF，請輸入這個可執行檔名稱來執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

註: HP-UX 提供 POSIX 緒檔案庫及 DCE 緒檔案庫。DB2 支援使用 POSIX 緒檔案庫的多緒應用程式。

HP-UX 上的多緒應用程式必須定義 _REENTRANT，才能執行編譯。 HP-UX 文件建議以 -D_POSIX_C_SOURCE=199506L 進行編譯。如此也可確保已定義 _REENTRANT。應用程式也需要鏈結 -lpthread。

Script 檔 bldmt，位於 sqllib/samples/c中，含有建置內含的 SQL 多緒程式的指令。

註: 對於 64 位元內含的 SQL 多緒程式，請使用 bldmt64 Script。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。 第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。 資料庫名稱、使用者 ID 及通行碼均爲可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- HP-UX
# Builds a C multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# Compile the program.
cc +DAportable -Aa -I$DB2PATH/include -D_POSIX_C_SOURCE=199506L -c $1.c

# Link the program
cc +DAportable -o $1 $1.o -L$DB2PATH/lib -ldb2 -lpthread
```

除了前面討論的 -D_POSIX_C_SOURCE=199506L 編譯選項及 -lpthread 鏈結選項，沒有 +e 編譯選項及公用程式檔案鏈結之外，其它的編譯及鏈結選項都與 bldapp 檔所使用的選項相同。關於這些選項的資訊，請參閱第176頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqc 建置範例程式 thdsrver，請輸入下列指令：

```
bldmt thdsrver
```

結果產生一個可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入這個可執行檔名稱：

```
thdsrver
```

HP-UX C++

本節含有下列主題：

- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)
- 多緒的應用程式

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/cpp中，含有建置 DB2 應用程式的指令。

註: 對於 64 位元的應用程式，請使用 bldapp64 Script。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一的必要參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- HP-UX
# Builds a C++ application program
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
  ./embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  aCC +DAportable -Aa -ext -I$DB2PATH/include -c utilemb.C
else
  # Compile the utilapi.C error-checking utility.
  aCC +DAportable -Aa -ext -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
aCC +DAportable -Aa -ext -I$DB2PATH/include -c $1.C
```

```
if [[ -f $1".sqC" ]]
then
  # Link the program with utilemb.o.
  aCC +DAportable -o $1 $1.o utilemb.o -L$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o.
  aCC +DAportable -o $1 $1.o utilapi.o -L$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

aCC HP aC++ 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

-Aa 開啓 ANSI C++「標準」特性。

-ext 容許不同 C++ 副檔名包含「很長的」支援。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

鏈結選項：

aCC 使用 HP aC++ 編譯器作爲鏈結器的前端。

+DAportable
使用在 PA_RISC 1.x 及 2.0 工作站與伺服器中相容的的程式碼。

-o $1 指定可執行檔。

$1.o 指定程式目的檔。

utilemb.o
如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。

utilapi.o
如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： $HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.C 建置非內含的 SQL DB2 API 範例程式 client，請輸入：

```
bldapp client
```

結果產生可執行檔 client。您可以輸入下面的指令，對 Sample 資料庫執行可執行檔：

client

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqC 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ 儲存程序的相關資訊。

Script 檔 bldsrv，位於 sqllib/samples/cpp 中，含有建置內含的 SQL 儲存程序的指令。

註: 對於 64 位元內含的 SQL 儲存程序，請使用 bldsrv64 Script。

bldsrv Script 會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的同一案例中，所以並不需要使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。 資料庫名稱是可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。

Script 檔會使用來源檔名稱 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- HP-UX
# Builds a C++ stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
./embprep $1 $2

# Compile the program. First ensure it is coded with extern "C".
aCC +DAportable +u1 -Aa +z -ext -I$DB2PATH/include -c $1.C

# Link the program to create a shared library.
aCC +DAportable -b -o $1 $1.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

aCC HP aC++ 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

+u1 容許未調整的資料存取。

-Aa 開啓 ANSI C++「標準」特性。

+z 產生與位置無關的程式碼。

-ext 容許不同 C++ 副檔名包含 "很長的" 支援。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$DB2PATH/include

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldsrv 的編譯及鏈結選項

鏈結選項：

aCC 使用 HP aC++ 編譯器作爲鏈結器的前端。

+DAportable
使用在 PA_RISC 1.x 及 2.0 工作站與伺服器中相容的的程式碼。

-b 建立共用檔案庫，而非正常的可執行檔。

-o $1 指定可執行檔。

$1.o 指定程式目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： -L$DB2PATH/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqC 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient，呼叫共用檔案庫中的儲存程序。

您可以使用 script 檔 bldapp 建置 spclient。請參閱第184頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

```
spclient database userid password
```

其中

term>
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。儲存程序會將輸出傳回從屬站應用程式。

使用者定義函數 (UDF)

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ UDF 的相關資訊。

在 sqllib/samples/c 中的 script 檔 bldudf 含有建置 UDF 的指令。

註: 對於 64 位元的 UDF，請使用 bldudf64 Script。

使用者定義的程式無法有內含的 SQL 陳述式。這表示若要建置 UDF 程式並不需要連接資料庫、前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- HP-UX
# Builds a C or C++ UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
if [[ -f $1".c" ]]
then
  aCC +DAportable +u1 -ext -Aa +z -I$DB2PATH/include -c $1.c
elif [[ -f $1".C" ]]
then
  aCC +DAportable +u1 -ext -Aa +z -I$DB2PATH/include -c $1.C
fi

# Link the program.
aCC +DAportable -b -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

aCC HP aC++ 編譯器。

+DAportable
產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。

+u1 容許未調整的資料存取。

-ext 容許不同 C++ 副檔名包含 ″很長的″ 支援。

-Aa 開啓 ANSI C++ 標準特性。

+z 產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$DB2PATH/include

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldudf 的編譯及鏈結選項

鏈結選項：

aCC 使用 HP aC++ 編譯器作爲鏈結器的前端。

-b 建立共用檔案庫，而非正常的可執行檔。

-o $1 指定可執行檔。

$1.o 指定程式目的檔。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： -L$DB2PATH/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

-ldb2apie
與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。您可以在 sqllib/samples/cpp 中使用 script 檔 bldapp，從來源檔 udfcli.sqC 建置 udfcli 程式。請參閱第184頁的『DB2 API 及內含的 SQL 應用程式』，取得詳細資訊。

若要呼叫 UDF，請輸入這個可執行檔名稱來執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會呼叫 udfsrv 檔案庫中的 ScalarUDF 函數。

多緒的應用程式

註: HP-UX 提供 POSIX 緒檔案庫及 DCE 緒檔案庫。在 HP-UX 上 DB2 僅支援使用 POSIX 緒檔案庫的多緒應用程式。

對 HP-UX C++ 而言，必須爲多緒應用程式使用 -D_HPUX_SOURCE，以便定義 rand_r。也必須定義 RWSTD_MULTI_THREAD 及 _REENTRANT，並使用 +W829。應用程式也需要鏈結 -lpthread。

Script 檔 bldmt 位於 sqllib/samples/cpp 中，含有建置內含的 SQL 多緒程式的指令。

註: 對於 64 位元內含的 SQL 多緒程式，請使用 bldmt64 Script。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。 第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。 資料庫名稱、使用者 ID 及通行碼均爲可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- HP-UX
# Builds a C++ multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
./embprep $1 $2 $3 $4

# Compile the program.
aCC +DAportable -ext -I$DB2PATH/include \
   -D_HPUX_SOURCE -DRWSTD_MULTI_THREAD -D_REENTRANT +W829 -c $1.C

# Link the program
aCC -o $1 $1.o -L$DB2PATH/lib -ldb2 -lpthread
```

除了前面討論的選項，及沒有公用程式檔鏈結的之外，所使用的其它編譯及鏈結選項也用於內含的 SQL Script 檔 bldapp。關於這些選項的資訊，請參閱第184頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqC 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生一個可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入這個可執行檔名稱：

```
thdsrver
```

Micro Focus COBOL

本節包含下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

如果您想要開發內含 SQL 及 DB2 API 呼叫的應用程式，而且您將使用 Micro Focus COBOL 編譯器，請牢記下列幾點：

- 當您使用命令行處理器指令 `db2 prep`，來前置編譯您的應用程式時，請使用 `target mfcob` 選項 (預設值)。
- 若要使用內建式前置編譯器前端、執行時直譯器或 Animator 除錯器，您必須執行 Micro Focus 提供的 `mkrts` 指令，新增 DB2 Generic API 進入點到 Micro Focus 執行時模組 `rts32`。您也必須執行 `mkcheck`，更新檢查檔。如果沒有執行，就會收到 SQLGSTRT 的 173 錯誤。

  在執行 `mkrts` 及 `mkcheck` 前，必須以下列步驟設定 `COBOPT`：

  1. 以 root 身份登入。
  2. 在目錄 $COBDIR/src/rts 中輸入：

     ```
     COBOPT=/opt/IBMdb2/V7.1/lib/db2mkrts.args; export COBOPT
     ksh mkrts
     mv $COBDIR/rts32 $COBDIR/rts32.orig
     cp rts32 $COBDIR/rts32
     ```

  3. 您也必須重建 `check` 可執行檔，Hewlett-Packard 將該檔案隨附在本產品中。如果沒有重建位在 `$COBDIR` 目錄中的 `check` 可執行檔，則使用 `cob -C SQL` 執行的編譯嘗試會失效，並附有執行時系統 173 錯誤，因爲 DB2 前置處理器會呼叫 DB2 檔案庫。欲重建 `check`，您應將 `$COBDIR` 目錄下的 `src/sql` 目錄變更爲根目錄，並執行 `mkcheck` script。一旦 script 完成，您就必須將產生的 `check` 可執行檔移到 `$COBDIR` 目錄。在目錄 `$COBDIR/src/sql` 中輸入：

     ```
     COBOPT=/opt/IBMdb2/V7.1/lib/db2mkrts.args; export COBOPT
     ksh mkcheck
     mv $COBDIR/check $COBDIR/check.orig
     cp check $COBDIR/check
     ```

  現在，您可以使用下列目錄中提供的引數來執行 `mkrts`：

  ```
  /opt/IBMdb2/V7.1/lib/db2mkrts.args
  ```

- 您必須在 Micro Focus COBOL 環境變數 COBCPY 中，併入 DB2 COBOL COPY 檔案目錄。COBCPY 環境變數會指定 COPY 檔的位置。 Micro Focus COBOL 的 DB2 COPY 檔常駐在資料庫案例目錄下的 `sqllib/include/cobol_mf` 中。

  若要加入此目錄，請輸入下列指令：

  ```
  export COBCPY=$COBCPY:/opt/IBMdb2/V7.1/include/cobol_mf
  ```

  註: 您可能想要在 `.profile` 檔中設定 COBCPY。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp 位於 sqllib/samples/cobol_mf 中，含有建置 DB2 API 及內含的 SQL 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- HP-UX
# Builds a Micro Focus COBOL application program
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqb" ]]
then
  embprep $1 $2 $3 $4
fi

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# Compile the checkerr.cbl error checking utility.
cob +DAportable -cx checkerr.cbl

# Compile the program.
cob +DAportable -cx $1.cbl

# Link the program.
cob +DAportable -x $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```

bldapp 的編譯及鏈結選項	
編譯選項：	
cob	Micro Focus COBOL 編譯器。
+DAportable	產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。
-cx	編譯至物件模組。

bldapp 的編譯及鏈結選項

鏈結選項：

cob 使用編譯器作爲鏈結器的前端。

+DAportable
使用在 PA_RISC 1.x 及 2.0 工作站與伺服器中相容的的程式碼。

-x 指定可執行程式。

$1.o 包括程式目的檔。

checkerr.o
包括公用程式目的檔以便檢查錯誤。

-L$DB2PATH/lib
指定 DB2 執行程式共用檔案庫的位置。 例如： `$HOME/sqllib/lib`。

-ldb2 鏈結 DB2 檔案庫。

-ldb2gmf
鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.cbl` 建置非內含的 SQL 範例程式 `client`，請輸入：

```
bldapp client
```

結果產生一個可執行檔 `client`。您可以輸入下面的指令，對 `Sample` 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqb` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 `updat`。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 `sample` 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

Script 檔 bldsrv 位於 sqllib/samples/cobol_mf 中，含有建置內含的 SQL 儲存程序的指令。Script 檔會將儲存程序編入伺服器上的共用檔案庫中，以便可透過從屬站應用程式來呼叫它。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2， 指定您想要與其連接的資料庫的名稱。 既然儲存程序必須建置在資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。若未提供資料庫名稱，程式會使用預設 sample 資料庫。 Script 檔會使用來源檔名稱 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- HP-UX
# Builds a Micro Focus COBOL stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$COBCPY:$DB2PATH/include/cobol_mf

# Compile the program.
cob +DAportable +z -cx $1.cbl

# Link the program.
ld -b -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2gmf \
   -L$COBDIR/coblib -lcobol -lcrtn

# Copy the shared library to the sqllib/function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項
編譯選項： **cob** COBOL 編譯器。 **+DAportable** 產生在 PA_RISC 1.x 及 2.0 工作站與伺服器中可相容的的程式碼。 **+z** 產生與位置無關的程式碼。 **-cx** 編譯至物件模組。
鏈結選項： **ld** 使用鏈結器來鏈結編輯。 **-b** 建立共用檔案庫，而非正常的可執行檔。 **-o $1** 指定可執行檔。 **$1.o** 包括程式目的檔。 **-L$DB2PATH/lib** 指定 DB2 執行程式共用檔案庫的位置。 例如： `$HOME/sqllib/lib` **-ldb2** 鏈結 DB2 共用檔案庫。 **-ldb2gmf** 鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。 **-L$COBDIR/coblib** 指定 COBOL 執行期檔案庫的位置。 **-lcobol** 鏈結 COBOL 檔案庫。 **-lcrtn** 鏈結 crtn 檔案庫。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `outsrv.sqb` 建置範例程式 `outsrv`，如果是連接到範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將儲存程序複製到 `sqllib/function` 目錄。

必要時，請設定儲存程序的檔案模式，以便從屬站應用程式可以存取它。

一旦您開發儲存程序 `outsrv`，您就可以開發從屬站應用程式 `outcli`，來呼叫儲存程序。您可以使用 script 檔 `bldapp` 建置 `outcli`。請參閱第194頁的『DB2 API 及內含的 SQL 應用程式』 ，取得詳細資訊。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可以存取儲存程序檔案庫 outsrv，在伺服器資料庫中執行名稱相同的儲存程序函數，然後將輸出傳回從屬站應用程式。

結束儲存程序

當您開發您的儲存程序時，請使用下列陳述式，來結束您的儲存程序：

```
move SQLZ-HOLD-PROC to return-code.
```

透過這個陳述式，儲存程序即會正確地傳回至從屬站應用程式。

第8章 開發 Linux 應用程式

本章提供如何在 Linux 上建置應用程式的詳細資訊。在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。 如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

若要取得 Linux 版的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

Linux C

本節包含下列主題：

- DB2 CLI 應用程式
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義函數 (UDF)
- 多緒的應用程式

DB2 CLI 應用程式

sqllib/samples/cli 中的 Script 檔 bldcli，包含用來建置 DB2 CLI 程式的指令。參數 $1 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名說明，則會呼叫 embprep script 來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- Linux
# Builds a CLI program with Linux C
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
embprep $1 $2 $3 $4
fi

# Compile the error-checking utility.
cc -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc -I$DB2PATH/include -c $1.c

# Link the program.
cc -o $1 $1.o utilcli.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
```

bldcli 的編譯選項和鏈結選項
編譯選項：
cc C 編譯器。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如： $HOME/sqllib/include
-c 僅執行編譯；沒有任何鏈結。 Script 檔有不同的編譯及鏈結步驟。

bldcli 的編譯選項和鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-Wl,-rpath,$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 tbinfo。您可以輸入下列可執行檔名稱來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。 第24頁的表7 中包含 CLI 範例程式的說明，指示使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。 這是此檔案與 bldcli script 間的唯一不同處。 請參閱第199頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入下列可執行檔名稱來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

sqllib/samples/cli 中的 Script 檔 bldclisp，包含用來建置 DB2 CLI 儲存程序的指令。參數 $1 指定來源檔名稱。

```
#! /bin/ksh
# bldclisp script file -- Linux
# Builds a CLI stored procedure in Linux C.
# Usage: bldclisp <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the error-checking utility.
cc -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc -I$DB2PATH/include -c $1.c

# Link the program.
cc -o $1 $1.o utilcli.o -shared -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
```

```
# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldclisp 的編譯選項和鏈結選項
編譯選項： **cc** C 編譯器。 **-I$DB2PATH/include** 指定 DB2 併入檔的位置。例如： $HOME/sqllib/include。 **-c** 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。
鏈結選項： **cc** 使用編譯器作爲鏈結器的前端。 **-o $1** 指定可執行檔。 **$1.o** 併入程式目的檔。 **utilcli.o** 包括公用程式目的檔以便檢查錯誤。 **-shared** 建立共用檔案庫。 **-L$DB2PATH/lib** 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。 **-Wl,-rpath,$DB2PATH/lib** 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。 **-ldb2** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，請輸入：

```
bldclisp spserver
```

Script 檔會將共用檔案庫複製到 sqllib/function 目錄。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了共用檔案庫 spserver，您就可以建置 CLI 從屬站應用程式 spclient，呼叫共用檔案庫。

您可以使用 script 檔 bldcli 建置 spclient。詳細資訊，請參閱第199頁的『DB2 CLI 應用程式』。

欲存取共用檔案庫，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/c 中，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。 這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯及連結 script embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Linux
# Builds a C application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
embprep $1 $2 $3 $4
  # Compile the utilemb.c error-checking utility.
  cc -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  cc -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o.
  cc -o $1 $1.o utilemb.o -L$DB2PATH/lib \
   -Wl,-rpath,$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o.
  cc -o $1 $1.o utilapi.o -L$DB2PATH/lib \
   -Wl,-rpath,$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： `$HOME/sqllib/include`

-c 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 指定目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：`$HOME/sqllib/lib`。如果您未指定 `-L` 選項，將使用 `/usr/lib:/lib`。
-Wl,-rpath,$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：`$HOME/sqllib/lib`。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.c` 建置 DB2 API 非內含的 SQL 範例程式 `client`，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 `client`。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqc` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 `updat`。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

Script 檔 bldsrv，位於 sqllib/samples/c 中，含有可建置內含的 SQL 儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。 第二個參數 $2， 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須在資料庫常駐的同一案例中建置，所以就不需要指定使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldsrv script file -- Linux
# Builds a C stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the program.
cc -I$DB2PATH/include -c $1.c

# Link the program and create a shared library
cc -shared -o $1 $1.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2

# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項
編譯選項：
cc C 編譯器。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如： $HOME/sqllib/include
-c 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
-shared 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-Wl,-rpath,$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqc 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient 以存取該檔案庫。

您可以使用 script 檔 bldapp，建置 spclient。詳細資訊，請參閱第204頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

Script 檔 bldudf，位於 sqllib/samples/c 中，含有建置 UDF 的指令。UDF 不內含 SQL 陳述式。 所以要建置 UDF 程式並不需要連接資料庫，或前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔也會使用這個來源檔名稱，來當作共用檔案庫名稱 。

```
#! /bin/ksh
# bldudf script file -- Linux
# Builds a C UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
cc -I$DB2PATH/include -c $1.c
```

```
# Link the program and create a shared library.
cc -o $1 $1.o -shared -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： $HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。

鏈結選項：

cc 使用編譯器作爲鏈結器的前端。

-o $1 指定可執行檔。

$1.o 併入程式目的檔。

-shared
建立共用檔案庫。

-L$DB2PATH/lib
指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-Wl,-rpath,$DB2PATH/lib
指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。

-ldb2 與 DB2 檔案庫鏈結。

-ldb2apie
與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，設定 UDF 的檔案模式，以便 DB2 案例可以執行它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。已提供 DB2 CLI 及此程式的內含 SQL 版本。

您可以在 sqllib/samples/cli 中使用 script 檔 bldcli，從來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。詳細資訊，請參閱第199頁的『DB2 CLI 應用程式』。

您可以在 sqllib/samples/c 中使用 script 檔 bldapp，從來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。詳細資訊，請參閱第204頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入下列可執行檔名稱，執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

使用 Linux C 的多緒應用程式必須使用 -D_REENTRANT 進行編譯，並使用 -lpthread 鏈結。

Script 檔 bldmt，位於 sqllib/samples/c中，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。 第二個參數 $2， 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。 資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldmt script file -- Linux
# Builds a C multi-threaded embedded SQL program.
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# Compile the program.
cc -I$DB2PATH/include -c $1.c -D_REENTRANT

# Link the program.
cc -o $1 $1.o -lpthread -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
```

除了前面討論的 -D_REENTRANT 及 -lpthread 選項，及沒有鏈結的公用程式檔以外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第204頁的『DB2 API 及內含的 SQL 應用程式』。

欲從來源檔 thdsrver.sqc 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

Linux C++

本節涵蓋下列主題：

- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義函數
- 多緒的應用程式

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/cpp 中，含有建置範例 C++ 程式的指令。

第一個參數 $1 指定您的來源檔的名稱。 這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數以前置編譯並連結檔案 embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Linux
# Builds a C++ application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  g++ -I$DB2PATH/include -c utilemb.C
else
  # Compile the utilapi.C error-checking utility.
  g++ -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
g++ -I$DB2PATH/include -c $1.C

if [[ -f $1".sqC" ]]
then
  # Link the program with utilemb.o
  g++ -o $1 $1.o utilemb.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o
  g++ -o $1 $1.o utilapi.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

g++ C++ 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： `$HOME/sqllib/include`

-c 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
g++ 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：`$HOME/sqllib/lib`。如果您未指定 `-L` 選項，將使用 `/usr/lib:/lib`。
-Wl,-rpath,$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：`$HOME/sqllib/lib`。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.C` 建置非內含的 SQL 範例程式 `client`，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 `client`。 您可以輸入下面的指令，對 `Sample` 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqC` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 `updat`。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ 儲存程序的相關資訊。

Script 檔 bldsrv，位於 sqllib/samples/cpp 中，含有建置內含的 SQL 儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。因爲儲存程序必須建置在資料庫常駐的同一案例中，所以您不需要使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。 資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。 Script 檔會使用來源檔名稱 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- Linux
# Builds a C++ stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the program.
g++ -I$DB2PATH/include -c $1.C

# Link the program and create a shared library.
g++ -shared -o $1 $1.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
```

```
# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項
編譯選項： **g++** C++ 編譯器。 **-I$DB2PATH/include** 指定 DB2 併入檔的位置。例如： $HOME/sqllib/include **-c** 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。
鏈結選項： **g++** 使用編譯器作爲鏈結器的前端。 **-shared** 建立共用檔案庫。 **-o $1** 指定可執行檔。 **$1.o** 併入程式目的檔。 **-L$DB2PATH/lib** 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。 **-Wl,-rpath,$DB2PATH/lib** 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。 **-ldb2** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqC 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient，呼叫共用檔案庫中的儲存程序。

您可以使用 script 檔 bldapp，建置 spclient。詳細資訊，請參閱第212頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。儲存程序會將輸出傳回從屬站應用程式。

使用者定義函數 (UDF)

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ UDF 的相關資訊。

Script 檔 bldudf，位於 sqllib/samples/cpp 中，含有建置 UDF 的指令。UDF 不內含 SQL 陳述式。 這表示若要建置 UDF 程式，您不需要連接資料庫、前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔使用此來源檔名稱作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- Linux
# Builds a C++ UDF library
# Usage: bldudf <prog_name>
```

```
# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
if [[ -f $1".c" ]]
then
g++ -I$DB2PATH/include -c $1.c

elif [[ -f $1".C" ]]
then
g++ -I$DB2PATH/include -c $1.C
fi

# Link the program.
g++ -o $1 $1.o -shared -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

g++ C++ 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： `$HOME/sqllib/include`。

-c 僅執行編譯；沒有任何鏈結。 此 Script 檔有不同的編譯及鏈結步驟。

bldudf 的編譯及鏈結選項
鏈結選項：
g++ 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
-shared 建立共用檔案庫。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-Wl,-rpath,$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
-ldb2apie 與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。您可以在 sqllib/samples/cpp 中使用 script 檔 bldapp，從來源檔 udfcli.sqC 建置 udfcli 程式。關於詳細資訊，請參閱第212頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入下列可執行檔名稱，執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會呼叫 udfsrv 檔案庫中的 ScalarUDF 函數。

多緒的應用程式

使用 Linux C++ 的多緒應用程式必須使用 -D_REENTRANT 進行編譯，並使用 -lpthread 鏈結。

Script 檔 bldmt，位於 sqllib/samples/cpp 中，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。 第二個參數 $2， 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。 資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldmt script file -- Linux
# Builds a C++ multi-threaded embedded SQL program.
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
embprep $1 $2 $3 $4

# Compile the program.
g++ -D_REENTRANT -I$DB2PATH/include -c $1.C

# Link the program.
g++ -lpthread -o $1 $1.o -L$DB2PATH/lib -Wl,-rpath,$DB2PATH/lib -ldb2
```

除了前面討論的 -D_REENTRANT 及 -lpthread 選項，及沒有鏈結的公用程式檔以外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第212頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqC 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

第9章 開發 OS/2 應用程式

本章提供在 OS/2 上建置應用程式的詳細資訊。在指令檔中，以 db2 開頭的指令即爲「命令行處理器 (CLP)」指令。如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

若要取得 OS/2 版的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

註: 含有使用者定義 SQLDA 的複合 SQL 陳述式，不容許使用在 OS/2 的 16 位元應用程式中。

IBM VisualAge C++ for OS/2 版本 3

本節包括下列主題：

- DB2 CLI 應用程式
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)

註: VisualAge C++ 編譯器用於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 目錄中所提供的 C 及 C++ 範例程式。其相同的指令檔也在這兩個目錄中。視指令檔的副檔名而定，檔案中含有接受 C 或 C++ 來源檔的指令。

DB2 CLI 應用程式

%DB2PATH%\samples\cli 中的指令檔 bldcli，包含用來建置 DB2 CLI 程式的指令。參數 %1 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數$3 指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名說明，則會呼叫 embprep 指令檔來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
@echo off
rem  bldcli command file - OS/2
rem  Builds a CLI program with IBM VisualAge C++.
rem  Usage: bldcli prog_name [ db_name [ userid password ]]

if exist "%1.sqc" call embprep %1 %2 %3 %4
if exist "%1.sqx" call embprep %1 %2 %3 %4
if "%1" == "" goto error

rem Compile the error-checking utility.
icc -C+ -O- -Ti+ utilcli.c

rem Compile the program.
if exist "%1.sqx" goto cpp
icc -C+ -O- -Ti+ %1.c
goto link_step
:cpp
icc -C+ -O- -Ti+ %1.cxx

rem Link the program.
:link_step
ilink /NOFREE /NOI /DEBUG /ST:64000 /PM:VIO %1.obj utilcli.obj,%1.exe,NUL,db2cli.
lib;
goto exit
:error
echo Usage: bldcli prog_name [ db_name [ userid password ]]
:exit
@echo on
```

bldcli 的編譯選項和鏈結選項	
編譯選項：	
icc	IBM VisualAge C++ 編譯器。
-C+	僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
-O-	沒有最佳化。關閉最佳化，更容易使用除錯器。
-Ti+	建立除錯器資訊

bldcli 的編譯選項和鏈結選項

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

/NOFREE
沒有可用格式。

/NOI 沒有不限字體。強迫區分大小寫識別字。

/DEBUG
包括除錯資訊。

/ST:64000
指定至少 64 000 的堆疊大小。

/PM:VIO
讓程式能夠在 OS/2 視窗中執行。

%1.obj
包括目的檔。

utilcli.obj
包括公用程式目的檔以便檢查錯誤。

%1.exe
指定可執行檔。

NUL 接受預設值。

db2cli.lib
鏈結 DB2 CLI 檔案庫。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 tbinfo.exe。您可以輸入下列可執行檔名稱 (不含副檔名) 來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果產生可執行檔 dbusemx.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例中的 sample 資料庫，則只要輸入可執行檔名稱 (不需要副檔名)：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，使用 DB2 API 的範例。

位於 %DB2PATH%\samples\cli 中的，指令檔 bldapi，含有使用具 DB2 API 指令所建置的 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldcli 指令檔間的唯一不同處。請參閱第221頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn, 請輸入：

```
bldapi dbmconn
```

結果產生可執行檔 dbmconn.exe。您可以輸入下列可執行檔名稱 (不含副檔名) 來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

%DB2PATH%\samples\cli 中的指令檔 bldclisp，包含用來建置 CLI 儲存程序的指令。指令檔將儲存程序建置成伺服器上的 DLL。

參數 %1 指定來源檔名稱。指令檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem  bldclisp command file - OS/2
rem  Builds a CLI stored procedure using the IBM VisualAge C++ compiler.
rem  Usage: bldclisp <prog_name>
```

```
if "%1" == "" goto error

rem Compile the error-checking utility.
icc -C+ -Ti+ -Ge- -Gm+ -W2 utilcli.c
rem Compile the program.
if exist "%1.cxx" goto cpp
icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.c
goto link_step
:cpp
icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.cxx

:link_step
rem Link the program and produce a DLL.
ilink /NOFREE /MAP /NOI /DEBUG /ST:64000 %1.obj utilcli.obj,%1.dll,,db2cli.
lib,%1.def;
rem Copy the stored procedure DLL to the 'function' directory
copy %1.dll %DB2PATH%\function

goto exit
:error
echo Usage: bldclisp prog_name
:exit
@echo on
```

bldclisp 的編譯選項和鏈結選項

編譯選項：

icc	IBM VisualAge C++ 編譯器。
-C+	僅執行編譯；沒有任何鏈結。指令檔有不同的編譯及鏈結步驟。
-Ti+	建立除錯器資訊。
-Ge-	建置 .DLL 檔案。使用靜態鏈結版的執行程式庫。
-Gm+	鏈結多工檔案庫。
-W2	輸出警告、錯誤、嚴重的及無法復原的錯誤訊息。

bldclisp 的編譯選項和鏈結選項

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

/NOFREE
沒有可用格式。

/MAP 建立對映檔。

/NOI 沒有不限字體。強迫區分大小寫識別字。

/DEBUG
包括除錯資訊。

/ST:64000
指定至少 64000 的堆疊大小。

%1.obj
包括目的檔。

%1.dll
建立動態鏈結檔案庫。

db2cli.lib
鏈結 DB2 CLI 檔案庫。

%1.def
模組定義檔。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，請輸入：

```
bldclisp spserver
```

Script 檔會將共用檔案庫複製到伺服器的路徑 %DB2PATH%\function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦建置了共用檔案庫 spserver，您就可以建置 CLI 從屬站應用程式 spclient，呼叫共用檔案庫中的儲存程序。

您可以使用指令檔 bldcli 建置 spclient。關於詳細資訊，請參閱第221頁的『DB2 CLI 應用程式』。

欲存取共用檔案庫，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

指令檔 bldapp.cmd，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置 DB2應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL的程式而言，這是唯一必要的參數。建置內含的 SQL程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2 指定您要連接的資料庫名稱；第三個參數 %3 指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL程式，bldapp 會傳送參數到前置編譯及鏈結指令檔，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldapp command file -- OS/2
rem Builds a VisualAge C++ application program
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

if exist "%1.sqx" goto embedded
if exist "%1.sqc" goto embedded
goto non_embedded

:embedded
rem Precompile and bind the program.
call embprep %1 %2 %3 %4
rem Compile the program.
if exist "%1.cxx" goto cpp_embedded
icc -c utilemb.c
```

```
icc -C+ -O- -Ti+ %1.c
goto link_embedded
:cpp_embedded
icc -c utilemb.cxx
icc -C+ -O- -Ti+ %1.cxx
goto link_embedded

:non_embedded
rem Compile the program.
if exist "%1.cxx" goto cpp
icc -c utilapi.c
icc -C+ -O- -Ti+ %1.c
goto link_non_embedded
:cpp
icc -c utilapi.cxx
icc -C+ -O- -Ti+ %1.cxx
goto link_non_embedded

rem Link the program.
:link_embedded
ilink /NOFREE /NOI /DEBUG /ST:64000 /PM:VIO %1.obj utilemb.obj,,,db2api;
goto exit
:link_non_embedded
ilink /NOFREE /NOI /DEBUG /ST:64000 /PM:VIO %1.obj utilapi.obj,,,db2api;
:exit
@echo on
```

bldapp 的編譯及鏈結選項

編譯選項：

icc	IBM VisualAge C++ 編譯器。
-C+	僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
-O-	沒有最佳化。關閉最佳化，更容易使用除錯器。
-Ti+	建立除錯器資訊

bldapp 的編譯及鏈結選項

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

/NOFREE
沒有可用格式。

/NOI 沒有不限字體。強迫區分大小寫識別字。

/DEBUG
包括除錯資訊。

/ST:64000
指定至少 64000 的堆疊大小。

/PM:VIO
讓程式能夠在 OS/2 視窗中執行。

utilemb.obj
如果是內含的 SQL程式，則有內含 SQL 公用程式的目的檔，可執行錯誤檢查。

utilapi.obj
如果不是內含的 SQL程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。

db2api
與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.c` 建置 DB2 API 非內含的 SQL範例程式 `client`, 請輸入：

```
bldapp client
```

結果產生可執行檔 `client.exe`。

欲執行可執行檔，請輸入可執行檔名稱 (不需要副檔名)：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqc` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 `updat.exe`。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例中的 sample 資料庫，則只要輸入可執行檔名稱 (不需要副檔名)：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

指令檔 bldsrv，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置內含的 SQL 儲存程序的指令。指令檔將儲存程序編譯成伺服器上的 DLL。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在與資料庫常駐的相同案例中，因此就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

指令檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldsrv command file -- OS/2
rem Builds a VisualAge C++ stored procedure
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2

rem Compile the program.
if exist "%1.cxx" goto cpp
icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.c
goto link_step
:cpp
icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.cxx

:link_step
rem Link the program.
ilink /NOFREE /NOI /DEBUG /ST:64000 %1.obj,%1.dll,,db2api,%1.def;

rem Copy the stored procedure to the %DB2PATH%\function directory.
copy %1.dll %DB2PATH%\function
@echo on
```

bldsrv 的編譯及鏈結選項
編譯選項： **icc** IBM VisualAge C++ 編譯器。 **-C+** 僅執行編譯；沒有任何鏈結。指令檔有不同的編譯及鏈結步驟。 **-Ti+** 建立除錯器資訊。 **-Ge-** 建置 .DLL 檔案。使用靜態鏈結版的執行程式庫。 **-Gm+** 鏈結多工檔案庫。 **-W2** 輸出警告、錯誤、嚴重的及無法復原的錯誤訊息。
鏈結選項： **ilink** 使用 ilink 鏈結器來鏈結編輯。 **/NOFREE** 沒有可用格式。 **/NOI** 沒有不限字體。強迫區分大小寫識別字。 **/DEBUG** 包括除錯資訊。 **/ST:64000** 指定至少 64000 的堆疊大小。 **%1.dll** 建立動態鏈結檔案庫。 **db2api** 與 DB2 檔案庫鏈結。 **%1.def** 模組定義檔。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqc 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

指令檔會將共用檔案庫複製到伺服器的路徑 %DB2PATH%\function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient，存取共用檔案庫。

您可以使用指令檔 bldapp 建置 spclient。關於詳細資訊，請參閱第227頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

指令檔 bldudf，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置 UDF 的指令。

UDF 無法包含內含的 SQL陳述式。因此，欲建置 UDF 程式，您不用連接到資料庫去做前置編譯及連結動作。

指令檔使用一個參數 %1，用來指定來源檔的名稱。使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem  bldudf command file -- OS/2
rem  Builds a VisualAge C++ user-defined function (UDF)
rem  Usage: bldudf <prog_name>

if "%1" == "" goto error
```

```
rem Compile the program.
if exist "%1.cxx" goto cpp
icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.c
goto link_step
:cpp
rem icc -C+ -Ti+ -Ge- -Gm+ -W2 %1.cxx

:link_step
rem Link the program.
ilink /NOFREE /MAP /NOI /DEBUG /ST:64000 %1.obj,%1.dll,,db2api db2apie,%1.def;

rem Copy the UDF to the %DB2PATH%\function directory
copy %1.dll %DB2PATH%\function

goto exit
:error
echo Usage: bldudf prog_name
:exit
@echo on
```

bldudf 的編譯及鏈結選項

編譯選項：

icc	IBM VisualAge C++ 編譯器。
-C+	僅執行編譯；沒有任何鏈結。指令檔有不同的編譯及鏈結步驟。
-Ti+	建立除錯器資訊。
-Ge-	建置 .DLL 檔案。使用靜態鏈結版的執行程式庫。
-Gm+	鏈結多工檔案庫。
-W2	輸出警告、錯誤、嚴重的及無法復原的錯誤訊息。

bldudf 的編譯及鏈結選項

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

/NOFREE
沒有可用格式。

/MAP 建立對映檔。

/NOI 沒有不限字體。強迫區分大小寫識別字。

/DEBUG
包括除錯資訊。

/ST:64000
指定至少 64000 的堆疊大小。

%1.dll
建立動態鏈結檔案庫。

db2api
與 DB2 檔案庫鏈結。

db2apie
鏈結 DB2 API 引擎檔案庫。

%1.def
模組定義檔。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到伺服器的路徑 %DB2PATH%\function 中。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。已提供 DB2 CLI 及此程式的內含 SQL 版本。

您可以使用位於指令檔 bldcli.cmd，將 %DB2PATH%\samples\cli 中的來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。詳細資訊，請參閱第221頁的『DB2 CLI 應用程式』。

您可以使用位於指令檔 bldapp，將 %DB2PATH%\samples\c 中的來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。關於詳細資訊，請參閱第227頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入可執行檔名稱 (不需要副檔名) 以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

IBM VisualAge C++ for OS/2 版本 4.0

VisualAge C++ 版本 4 編譯器的應用程式建置資訊，亦可用於 AIX、OS/2 及 Windows 32 位元作業系統。請參閱第137頁的『VisualAge C++ 版本 4.0』取得此資訊。

IBM VisualAge COBOL for OS/2

本節包括下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

這些主題可協助您在 DB2 中使用 IBM VisualAge COBOL 編譯器。

建立連結檔案的實例

使用 DB2 for OS/2 及 IBM Cobol 來建立應用程式時， DB2 前置編譯器通常無法建立連結檔案。這是因爲 OS/2 內的檔案 handle 限制。

修正程式會讓 OS/2 容許在執行編譯的機器上有更多的檔案 handle。這一行：

```
SET SHELLHANDLESINC=20
```

應該在安裝 DB2 for OS/2 的機器上，插入 CONFIG.SYS 檔案內。另外，您也可以在編譯時使用 NODATA 選項 (此爲 IBM Cobol 選項)。

內含的 SQL 及 DB2 API 呼叫

如果要開發的應用程式包含內含的 SQL 及 DB2 API 呼叫，且您使用 IBM VisualAge COBOL 編譯器，請牢記下列幾點：

- 當您使用命令行處理器指令 db2 prep 來前置編譯應用程式時，請使用 target ibmcob 選項。
- 請勿在您的來源檔中使用 Tab 鍵字元。

- 您可以在來源檔中使用 PROCESS 及 CBL 關鍵字，設定編譯選項。關鍵字只能放在直欄 8 至 72 中。
- 如果您的應用程式僅內含 SQL，且沒有任何 DB2 API 呼叫，則您不需要使用 pgmname(mixed) 編譯選項。如果您使用 DB2 API 呼叫，則您必須使用 pgmname(mixed) 編譯選項。
- 如果您使用的是 IBM VisualAge COBOL 編譯器的「System/390 主電腦資料類型支援」特性，則您應用程式的 DB2 併入檔會位於下列目錄中：

  ```
  %DB2PATH%\include\cobol_i
  ```

  如果您是使用提供的指令檔建置 DB2 範例程式，則指令檔中所指定的併入檔路徑必須變更爲指向 cobol_i 目錄，而非指向 cobol_a 目錄。

  如果您「不」是使用 IBM VisualAge COBOL 編譯器的「System/390 主電腦資料類型支援」特性，或您使用的是此編譯器的舊版本，則您應用程式的 DB2 併入檔會位於下列目錄中：

  ```
  %DB2PATH%\include\cobol_a
  ```

  指定 COPY 檔名，來併入 .cbl 副檔名，方式如下：

  ```
  COPY "sql.cbl".
  ```

內含的 SQL 應用程式

指令檔 bldapp.cmd，位於 %DB2PATH%\samples\cobol 中，含有建置 DB2 應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL的程式而言，這是唯一必要的參數。建置內含的 SQL程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2 指定您要連接的資料庫名稱；第三個參數 %3 指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL程式，bldapp 會傳送參數到前置編譯及鏈結指令檔，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldapp command file -- OS/2
rem Builds a VisualAge COBOL application program
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

rem If an embedded SQL program, precompile and bind it.
if exist "%1.sqb" goto prepbind
goto compile_step
:prepbind
call embprep %1 %2 %3 %4
```

```
:compile_step
rem Compile the checkerr error checking utility.
cob2 -c -g -qpgmname(mixed) -qlib -I%DB2PATH%\include\cobol_a checkerr.cbl

rem Compile the program.
cob2 -c -g -qpgmname(mixed) -qlib -I%DB2PATH%\include\cobol_a %1.cbl

rem Link the program.
ilink %1.obj checkerr.obj db2api.lib /ST:64000 /PM:VIO /NOI /DEBUG
@echo on
```

bldapp 的編譯及鏈結選項

編譯選項：

cob2 IBM VisualAge COBOL 編譯器。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-g 包括除錯資訊。

-qpgmname(mixed)
指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。

-qlib 指示編譯器處理 COPY 陳述式。

-I*路徑* 指定 DB2 併入檔的位置。例如： `-I%DB2PATH%\include\cobol_a`。

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

checkerr.obj
包括錯誤檢查公用程式目的檔。

db2api.lib
與 DB2 檔案庫鏈結。

/ST:64000
指定至少 64000 的堆疊大小。

/PM:VIO
讓程式能夠在 OS/2 視窗中執行。

/NOI 鏈結時不要不限字體。

/DEBUG
包括除錯資訊。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.cbl` 建置非內含的 SQL範例程式 `client`，請輸入：

```
bldapp client
```

結果產生可執行檔 `client.exe`。您可以輸入可執行檔名稱 (不需要副檔名)，以對 `sample` 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果要存取同一案例中的 sample 資料庫，則只要輸入可執行檔名稱 (不需要副檔名)：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

指令檔 bldsrv，位於 %DB2PATH%\samples\cobol 中，含有建置儲存程序的指令。指令檔將儲存程序編譯成伺服器上的 DLL。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在資料庫常駐的同一案例中，就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

指令檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldsrv command file -- OS/2
rem Builds a VisualAge COBOL stored procedure
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2
```

```
rem Compile the program.
cob2 -c -g -qpgmname(mixed) -qlib -I%DB2PATH%\include\cobol_a %1.cbl

rem Link the program.
ilink %1.obj checkerr.obj %1.def db2api.lib /ST:64000 /PM:VIO /NOI /DEBUG

rem Copy stored procedure to the %DB2PATH%\function directory.
copy %1.dll %DB2PATH%\function
@echo on
```

bldsrv 的編譯及鏈結選項

編譯選項：

cob2 IBM VisualAge COBOL 編譯器。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-g 包括除錯資訊。

-qpgmname(mixed)
指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。

-qlib 指示編譯器處理 COPY 陳述式。

-I*路徑* 指定 DB2 併入檔的位置。例如： `-I%DB2PATH%\include\cobol_a`。

鏈結選項：

ilink 使用 ilink 鏈結器來鏈結編輯。

checkerr.obj
包括錯誤檢查公用程式目的檔。

%1.def
模組定義檔。

db2api.lib
與 DB2 檔案庫鏈結。

/ST:64000
指定至少 64000 的堆疊大小。

/PM:VIO
讓程式能夠在 OS/2 視窗中執行。

/NOI 鏈結時不要不限字體。

/DEBUG
包括除錯資訊。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `outsrv.sqb` 建置範例程式 `outsrv`，並連接範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接的是另一個資料庫，亦請併入資料庫名稱：

```
bldsrv outsrv database
```

指令檔會使用與範例程式的同一目錄中所含的模組定義檔 outsrv.def，建置 DLL。指令檔會複置伺服器路徑 %DB2PATH%\function 中的儲存程序 DLL outsrv.dll。

必要時，請設定 DLL 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 DLL outsrv，您就可以建置從屬站應用程式 outcli，存取 DLL。您可以使用指令檔 bldapp 建置 outcli。關於詳細資訊，請參閱第236頁的『內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取 DLL outsrv，並在伺服器資料庫中執行名稱相同的儲存程序函數，然後將輸出傳回從屬站應用程式。

Micro Focus COBOL

本節包括下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

DB2 不支援 Micro Focus COBOL 編譯器所附的 link386 鏈結器。若要鏈結 DB2 Micro Focus COBOL 程式，您必須使用 IBM 編譯器產品提供的 ilink 鏈結器。本節的 Script 檔中使用的 cbllink 指令，會呼叫 ilink 鏈結器。

使用 Micro Focus COBOL 編譯器來建置含有內含的 SQL 及 DB2 API 呼叫的應用程式時，請牢記下列幾點：

- 當您在使用命令行處理器指令 db2 prep 前置編譯您的應用程式時，請使用預設的 target mfcob 選項。
- 確定 LIB 環境變數指向 %DB2PATH%\lib，例如：

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

- Micro Focus COBOL 的 DB2 COPY 位於 %DB2PATH%\include\cobol_mf。設定 COBCPY 環境變數來包括目錄，例如：

  ```
  set COBCPY=%DB2PATH%\include\cobol_mf;%COBCPY%
  ```

對所有 DB2 應用程式設計介面的呼叫，必須使用呼叫慣例 8 來執行。 DB2 COBOL 前置編譯器會自動將 CALL-CONVENTION 子句插入 SPECIAL-NAMES 段落中。若 SPECIAL-NAMES 段落不存在，DB2 COBOL 會另外建立，如下所示：

```
Identification Division
Program-ID. "static".
special-names.
    call-convention 8 is DB2API.
```

同時，若有呼叫 DB2 API，前置編譯器會在 ″call″ 關鍵字後面自動加入符號 DB2API，用來識別呼叫慣例。 每當前置編譯器從內含的 SQL 陳述式中建立 DB2 API 執行呼叫，就會進行這種動作。

若 DB2 API 呼叫出現在未經過前置編譯的應用程式中，您應該在應用程式中自行建立一個如同上述的 SPECIAL-NAMES 段落。若直接呼叫 DB2 API，則需要自行在 ″call″ 關鍵字後面加上 DB2API 符號。

DB2 API 及內含的 SQL 應用程式

指令檔 bldapp，位於 %DB2PATH%\samples\cobol_mf 中，含有建置 DB2 應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL的程式而言，這是唯一必要的參數。建置內含的 SQL程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2 指定您要連接的資料庫名稱；第三個參數 %3 指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL程式，bldapp 會傳送參數到前置編譯及鏈結指令檔，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldapp command file -- OS/2
rem Builds a Micro Focus COBOL application program
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]
```

```
rem If an embedded SQL program, precompile and bind it.
if exist "%1.sqb" goto prepbind
goto compile_step
:prepbind
call embprep %1 %2 %3 %4

:compile_step
rem Compile the error-checking utility.
cobol checkerr.cbl;

rem Compile the program.
cobol %1.cbl;

rem Link the program.
cbllink %1.obj checkerr.obj db2api.lib db2gmf32.lib
@echo on
```

bldapp 的編譯及鏈結選項
編譯選項： **cobol**　Micro Focus COBOL 編譯器。
鏈結選項： **cbllink** 使用鏈結器來鏈結編輯。 **checkerr.obj** 包括錯誤檢查公用程式目的檔。 **db2api.lib** 鏈結 DB2 API 檔案庫。 **db2gmf32.lib** 鏈結 M. F. COBOL 的 DB2 例外處理程式檔案庫。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.cbl 建置非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生可執行檔 client.exe。您可以輸入可執行檔名稱 (不需要副檔名)，以對 sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果要存取同一案例中的 sample 資料庫，則只要輸入可執行檔名稱 (不需要副檔名)：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

指令檔 bldsrv，位於 %DB2PATH%\samples\cobol_mf 中，含有建置內含的 SQL 儲存程序的指令。指令檔將儲存程序編譯成伺服器上的 DLL。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在資料庫常駐的同一案例中，就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。指令檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldsrv command file -- OS/2
rem Builds a Micro Focus COBOL stored procedure
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2

rem  Compile the stored procedure.
cobol %1.cbl;

rem  Link the stored procedure and create a shared library.
cbllink /d %1.obj db2api.lib db2gmf32.lib
```

```
rem Copy the stored procedure to the %DB2PATH%\function directory.
copy %1.dll %DB2PATH%\function
@echo on
```

bldsrv 的編譯及鏈結選項
編譯選項： **cobol**　Micro Focus COBOL 編譯器。
鏈結選項： **cbllink** 使用 Micro Focus COBOL 鏈結器來鏈結編輯。 **/d**　建立 .dll 檔案。 **db2api.lib** 包括 DB2 API 檔案庫。 **db2gmf32.lib** 鏈結 M. F. COBOL 的 DB2 例外處理程式檔案庫。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 outsrv.sqb 建置範例程式 outsrv，如果是連接到範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv outsrv database
```

鏈結器採用使用者未指定的預設進入點。 /d 選項用來建立 .dll 檔案，以便建置儲存程序。指令檔會複置伺服器路徑 %DB2PATH%\function 中的儲存程序 DLL outsrv.dll。

必要時，請設定 DLL 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 DLL outsrv，您就可以建置存取它的從屬站應用程式 outcli。您可以使用指令檔 bldapp 建置 outcli。關於詳細資訊，請參閱第241頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取 DLL outsrv，並在伺服器資料庫中執行名稱相同的儲存程序函數。然後將輸出傳回從屬站應用程式。

REXX

您不必編譯或連結 REXX 程式。

在 OS/2 上，您的應用程式副檔名必須是 .cmd。建立之後，您可以直接在作業系統指令提示下執行應用程式。

OS/2 REXX 程式必須在第一行的第一欄中包含註解，以便與批次指令有所區分：

```
/* Any comment will do. */
```

REXX 範例程式位於 %DB2PATH%\samples\rexx 目錄中。若要執行範例 REXX 程式 updat，請輸入：

```
updat
```

關於 REXX 及 DB2 的進一步資訊，請參閱 *Application Development Guide* 的 ″REXX 程式設計″ 這一章。

第10章 開發 PTX 應用程式

本章提供在 PTX 上使用 DB2 for NUMA-Q 建置應用程式的詳細資訊。 在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。 如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書

若要取得 PTX 版最新的 DB2 應用程式開發更新組件，請造訪下列網頁：

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

ptx/C

本節含有下列主題：

- DB2 CLI 應用程式
- DB2 CLI 應用程式與 DB2 API
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義函數 (UDF)
- 多緒的應用程式

DB2 CLI 應用程式

Script 檔 bldcli，位於 sqllib/samples/cli 中，含有建置 DB2 CLI 程式的指令。參數 $1 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名說明，則會呼叫 embprep script 來前置編譯此程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- PTX
# Builds a DB2 CLI program
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
fi

# Compile the error-checking utility.
cc -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc -I$DB2PATH/include -c $1.c

# Link the program.
cc -o $1 $1.o utilcli.o -L$DB2PATH/lib -ldb2
```

bldcli 的編譯及鏈結選項

編譯選項：

cc 使用 C 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： $HOME/sqllib/include

-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。

bldcli 的編譯及鏈結選項
鏈結選項： **cc** 使用編譯器作爲鏈結器的前端。 **-o $1** 指定可執行檔程式。 **$1.o** 包括程式目的檔。 **utilcli.o** 包括公用程式目的檔以便檢查錯誤。 **-L$DB2PATH/lib** 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如，$HOME/sqllib/lib **-ldb2** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 tbinfo。您可以輸入可執行檔名稱，執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldcli script 間的唯一不同處。請參閱第247頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入可執行檔名稱，執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

Script 檔 bldclisp，位於 sqllib/samples/cli 中，含有建置 DB2 CLI 儲存程序的指令。參數 $1 指定來源檔名稱。

```
#! /bin/ksh
# bldclisp script file -- PTX
# Builds a DB2 CLI stored procedure
# Usage: bldclisp <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the error-checking utility.
cc -KPIC -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc -KPIC -I$DB2PATH/include -c $1.c

# Link the program.
cc -G -o $1 $1.o utilcli.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldclisp 的編譯選項和鏈結選項

編譯選項：

cc C 編譯器。

-KPIC 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。

鏈結選項：

cc 使用編譯器作爲鏈結器的前端。

-G 建立共用檔案庫。

-o $1 指定可執行檔。

$1.o 包括程式目的檔。

utilcli.o
包括公用程式目的檔以便檢查錯誤。

-L$DB2PATH/lib
指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，請輸入：

```
bldclisp spserver
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置存取共用檔案庫的 CLI 從屬站應用程式 spclient。

您可以使用 script 檔 bldcli 建置 spclient。關於詳細資訊，請參閱第247頁的『DB2 CLI 應用程式』。

欲存取共用檔案庫，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

database
: 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
: 爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver, 並在伺服器資料庫中執行許多儲存程序函數，然後將輸出傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

建置檔 bldapp，位於 sqllib/samples/c，含有建置 DB2 API 及內含的 SQL 程式的指令。

第一個參數 $1，指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL DB2 API 程式唯一需要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- PTX
# Builds a C application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to the location where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# if an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.c error-checking utility.
  cc -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  cc -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  cc -o $1 $1.o utilemb.o -L$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o
  cc -o $1 $1.o utilapi.o -L$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項
編譯選項：
cc C 編譯器。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如：$HOME/sqllib/include
-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 包括程式目的檔。
util.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c 建置非內含的 SQL範例程式 client，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqc 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

Script 檔 bldsrv，位於 sqllib/samples/c 中，含有建置內含的 SQL儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1，指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須與資料庫常駐的同一案例中建置，您就不需要指定使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldsrv script file -- PTX
# Builds a C stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the program.
cc -KPIC -I$DB2PATH/include -c $1.c

# Link the program and create a shared library.
cc -G -o $1 $1.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項
編譯選項：
cc C 編譯器。
-KPIC 爲共用檔案庫產生與位置無關的程式碼。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如：-I$DB2PATH/include
-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
-G 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 包括程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqc 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient 以存取該檔案庫。

您可以使用 Script 檔 bldapp 建置 spclient。關於詳細資訊，請參閱第253頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

Script 檔 bldudf，位於 sqllib/samples/c 中，含有建置 UDF 的指令。UDF 不包含內含的 SQL陳述式。因此，若要建置 UDF 程式，您不必連接至資料庫，或前置編譯和連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- PTX
# Builds a C user-defined function library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
cc -KPIC -I$DB2PATH/include -c $1.c

# Link the program and create a shared library.
cc -G -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

-KPIC 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。

鏈結選項：

cc 使用編譯器作爲鏈結器的前端。

-G 建立共用檔案庫。

-o $1 指定可執行檔。

$1.o 包括程式目的檔。

-L$DB2PATH/lib
指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-ldb2 與 DB2 檔案庫鏈結。

-ldb2apie
與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。已提供 DB2 CLI 及此程式的內含 SQL 版本。

您可以在 sqllib/samples/cli 中使用 script 檔 bldcli，從來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。關於詳細資訊，請參閱第247頁的『DB2 CLI 應用程式』。

您可以在 sqllib/samples/c 中使用 script 檔 bldapp，從來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。關於詳細資訊，請參閱第253頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入可執行檔名稱，以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

使用 ptx/C 的多緒應用程式必須以 -Kthread 進行編譯及鏈結。

Script 檔 bldmt，位於 sqllib/samples/c中，含有建置內含的 SQL多緒程式的指令。

第一個參數 $1，指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- PTX
# Builds a C multi-threaded embedded SQL program.
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to the location where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4
```

```
# Compile the program.
cc -Kthread -I$DB2PATH/include -c $1.c

# Link the program.
cc -Kthread -o $1 $1.o -L$DB2PATH/lib -ldb2
```

除了前面討論的 -Kthread 選項，及沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第253頁的『DB2 API 及內含的 SQL 應用程式』。

欲從來源檔 thdsrver.sqc 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

ptx/C++

本節含有下列主題：

- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義函數 (UDF)
- 多緒的應用程式

DB2 API 及內含的 SQL 應用程式

建置檔 bldapp，位於 sqllib/samples/cpp，含有建置 DB2 API 及內含的 SQL 程式的指令。

第一個參數 $1，指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL DB2 API 程式唯一需要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- PTX
# Builds a C++ application program
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to the location where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# if an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  c++ -I$DB2PATH/include -D_RWSTD_COMPILE_INSTANTIATE=0 -c utilemb.C
else
  # Compile the utilapi.C error-checking utility.
  c++ -I$DB2PATH/include -D_RWSTD_COMPILE_INSTANTIATE=0 -c utilapi.C
fi

# Compile the program.
c++ -I$DB2PATH/include -D_RWSTD_COMPILE_INSTANTIATE=0 -c $1.C

if [[ -f $1".sqC" ]]
then
  # Link the program with utilemb.o
  c++ -o $1 $1.o utilemb.o -L$DB2PATH/lib -ldb2 -lseq
else
  # Link the program with utilapi.o
  c++ -o $1 $1.o utilapi.o -L$DB2PATH/lib -ldb2 -lseq
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

c++ C++ 編譯器。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include

-D_RWSTD_COMPILE_INSTANTIATE=0
請勿個體化 rogue wave 類別。

-c 僅執行編譯，不執行鏈結。 編譯及鏈結是不同的步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
c++ 使用編譯器作爲鏈結器的前端。
-o $1 指定可執行檔。
$1.o 包括程式目的檔。
utilemb.o 如果是內含的 SQL程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果是非內含的 SQL程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
-lseq 與 Sequent 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.C 建置 DB2 API 非內含的 SQL範例程式 client，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqC 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例中的另一個資料庫，請加上資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例中的資料庫，請加上資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ 儲存程序的相關資訊。

Script 檔 bldsrv，位於 sqllib/samples/cpp 中，含有建置內含的 SQL儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1，指定您的來源檔的名稱。這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以附加的選用參數 $2，會指定您要連接的資料庫名稱。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。既然儲存程序必須建置在與資料庫常駐的同一案例中，就不需要使用者 ID 及通行碼的任何附加參數。 Script 檔 bldsrv 會傳送參數到前置編譯及連結檔案，embprep。

來源檔名稱 $1 亦被用於共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- PTX
# Builds a C++ stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Compile the program. First ensure it is coded with extern "C".
c++ -KPIC -I$DB2PATH/include -D_RWSTD_COMPILE_INSTANTIATE=0 -c $1.C

# Link the program and create a shared library.
c++ -G -o $1 $1.o -L$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1.so $DB2PATH/function/$1
```

bldsrv 的編譯及鏈結選項
編譯選項：
c++ C++ 編譯器。
-KPIC 爲共用檔案庫產生與位置無關的程式碼。
-I$DB2PATH/include 指定 DB2 併入檔的位置。例如：-I$DB2PATH/include
-D_RWSTD_COMPILE_INSTANTIATE=0 請勿個體化 rogue wave 類別。
-c 僅執行編譯，不執行鏈結。 本書假定編譯及鏈結是分開的步驟。
鏈結選項：
c++ 使用編譯器作爲鏈結器的前端。
-G 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 包括程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqC 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient，呼叫共用檔案庫中的儲存程序。

您可以使用 Script 檔 bldapp 建置 spclient。關於詳細資訊，請參閱第260頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

database
爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。儲存程序會將輸出傳回從屬站應用程式。

使用者定義函數 (UDF)

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』，取得建置 C++ UDF 的相關資訊。

Script 檔 bldudf，位於 sqllib/samples/cpp 中，含有建置 UDF 的指令。UDF 不包含內含的 SQL陳述式。因此，若要建置 UDF 程式，您不必連接至資料庫，或前置編譯和連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- PTX
# Builds a C++ user-defined function library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Compile the program.
c++ -KPIC -I$DB2PATH/include  -D_RWSTD_COMPILE_INSTANTIATE=0 -c $1.c

# Link the program and create a shared library.
c++ -G -o $1 $1.o -L$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1.so $DB2PATH/function/$1
```

bldudf 的編譯及鏈結選項

編譯選項：

c++ C++ 編譯器。

-KPIC 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-D_RWSTD_COMPILE_INSTANTIATE=0
請勿個體化 rogue wave 類別。

-c 僅執行編譯，不執行鏈結。 本書假定編譯及鏈結是分開的步驟。

bldudf 的編譯及鏈結選項
鏈結選項：
c++ 使用編譯器作爲鏈結器的前端。
-G 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 包括程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-ldb2 與 DB2 檔案庫鏈結。
-ldb2apie 與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔將 UDF 複製到路徑 sqllib/function 中的伺服器。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。您可以 script 檔 bldapp，從在 sqllib/samples/cpp 中使用來源檔 udfcli.sqC 建置 udfcli 程式。關於詳細資訊，請參閱第260頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF，請輸入可執行檔名稱，以執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會呼叫 udfsrv 檔案庫中的 ScalarUDF 函數。

多緒的應用程式

使用 ptx/C++ 的多緒應用程式必須以 -Kthread 進行編譯及鏈結。

Script 檔 bldmt, 位於 sqllib/samples/cpp 中，含有建置內含的 SQL多緒程式的指令。

第一個參數 $1，指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式會使用預設的 sample 資料庫。

```
#! /bin/ksh
# bldmt script file -- PTX
# Builds a C++ multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to the location where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# Compile the program.
c++ -Kthread -I$DB2PATH/include -D_RWSTD_COMPILE_INSTANTIATE=0 -c $1.C

# Link the program.
c++ -Kthread -o $1 $1.o -L$DB2PATH/lib -ldb2 -lseq
```

除了前面討論的 -Kthread 選項，及沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第260頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqC 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

第11章 開發 Silicon Graphics IRIX 應用程式

本章提供在 Silicon Graphics IRIX 中建置 DB2 應用程式的詳細資訊。在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

若要取得 Silicon Graphics IRIX 的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

DB2 for Silicon Graphics IRIX 只適用於從屬站。若要執行 DB2 應用程式及建置 DB2 內含的 SQL 應用程式，您必須由從屬站機器存取伺服器機器上的 DB2 資料庫。伺服器機器會執行另一個作業系統。關於架構從屬站對伺服器通信的資訊，請參閱 *DB2 for UNIX 快速入門*。

另外，由於您將從在不同作業系統上執行的遠端從屬站存取伺服器上的資料庫，所以必須連結資料庫公用程式 (包括 DB2 CLI) 到資料庫。關於詳細資訊，請參閱第42頁的『連結』。

DB2 檔案庫支援

Silicon Graphics IRIX 提供三種不同且不相容的物件類型：o32 (預設值)、n32 (新的 32 物件類型) 及 64 (64 物件類型)。DB2 不支援 64, 但支援 o32 及 n32 物件類型。

作業系統有兩種不同且不相容的緒 API 版本：sproc 介面及 POSIX 緒 API。DB2 支援 POSIX 緒 API。

使用 sproc 介面的應用程式可以使用非緒版本的 DB2 檔案庫 libdb2，它不提供安全的緒。小心使用 sproc 介面，因爲 libdb2 不具 sproc 安全性。

爲了容納此部份的功能，DB2 提供下列檔案庫支援：

lib/libdb2.so
不具有緒的 o32

lib/libdb2_th.so
具有 POSIX 緒的 o32

lib32/libdb2.so
不具有緒的 n32

lib32/libdb2_th.so
具有 POSIX 緒的 n32

若要使用 n32 物件類型，程式必須使用 -n32 選項來編譯及鏈結，亦必須鏈結 lib32/libdb2.so 或 lib32/libdb2_th.so 檔案庫。若要使用預設的 o32 物件類型，程式必須鏈結 lib/libdb2.so 或 lib/libdb2_th.so 檔案庫，但不啓用 -n32 選項。

註: 欲使用本章中所述的建置檔來建置 n32 物件類型應用程式，請解除指示指令的註解。

MIPSpro C

本節解譯如何在下列 DB2 介面類型上使用 MIPSpro C：
- DB2 CLI
- DB2 API
- 內含的 SQL

DB2 CLI 應用程式

sqllib/samples/cli 中的 Script 檔 bldcli，包含用來建置 DB2 CLI 程式的指令。參數 $1 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2，指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，且 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名說明，則會呼叫 embprep script 來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- Silicon Graphics IRIX
# Builds a CLI program with MIPSpro C.
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]
```

```
# Set DB2PATH to where DB2 will be accessed.
# The default is the instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
fi

# To compile with n32 object support, uncomment the following line.
# IRIX_OBJECT_MODE=-n32

if [ "$IRIX_OBJECT_MODE" = "-n32" ] ; then
  # Link with db2 n32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib32
else
  # Link with db2 o32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib
fi

# Compile the error-checking utility.
cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c $1.c

# Link the program.
cc $IRIX_OBJECT_MODE -o $1 $1.o utilcli.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH
-lm -ldb2
```

bldcli 的編譯選項和鏈結選項

編譯選項：

cc
使用 C 編譯器。

$IRIX_OBJECT_MODE
如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$HOME/sqllib/include

-c
僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldcli 的編譯選項和鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
$IRIX_OBJECT_MODE 如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2_LIBPATH 指定在鏈結時 DB2 固定及共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-rpath $DB2_LIBPATH 指定在執行時 DB2 共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 tbinfo。您可以輸入可執行檔名稱、資料庫名稱及資料庫所在位置的案例之使用者 ID 及通行碼，以執行可執行檔：

```
tbinfo database userid password
```

建置及執行內含的 SQL 應用程式

欲從來源檔 dbusemx.sqc 建置 dbusemx，請併入資料庫的參數，及含有資料庫的案例之使用者 ID 與通行碼的參數：

```
bldcli dbusemx database userid password
```

結果會產生可執行檔 dbusemx。欲執行內含的 SQL 應用程式，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
dbusemx database userid password
```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。 這是此檔案與 bldcli script 間的唯一不同處。 請參閱第270頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入可執行檔名稱、資料庫名稱及資料庫所在位置的案例之使用者 ID 及通行碼，以執行可執行檔：

```
dbmconn database userid password
```

儲存程序的 DB2 CLI 從屬站應用程式

儲存程序是存取資料庫並傳回資訊給從屬站應用程式的程式。您要在伺服器上編譯及儲存儲存程序。此伺服器在另一個平台上執行。

欲在支援 DB2 的平台伺服器中建置 DB2 CLI 儲存程序 spserver，請參照本書中有關該平台的「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了儲存程序 spserver，您就可以從來源檔 spclient.c 使用 Script 檔 bldcli 建置呼叫儲存程序 spclient 的從屬站應用程式。詳細資訊，請參閱第270頁的『DB2 CLI 應用程式』。

您可以輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以呼叫儲存程序：

```
spclient database userid password
```

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

UDF 的 DB2 CLI 從屬站應用程式

使用者定義的函數 (UDF) 是您自己在伺服器上編譯及儲存的純量函數和表格函數。此伺服器在另一個平台上執行。 欲在支援 DB2 的平台伺服器中建置使用者定義的

函數程式 udfsrv，請參照本書中有關該平台的「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了 udfsrv，您就可以從 sqllib/samples/cli 中的來源檔 udfcli.c，使用 DB2 CLI Script 檔 bldcli，建置 DB2 CLI 從屬站應用程式 udfcli 來呼叫它。詳細資訊，請參閱第270頁的『DB2 CLI 應用程式』。

欲呼叫 UDF 程式，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以執行呼叫應用程式：

```
udfcli database userid password
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp, 位於 sqllib/samples/c 中，含有建置 DB2 應用程式的指令。第一個參數 $1 指定您的來源檔的名稱。這是唯一必要的參數，也是沒有內含的 SQL 的 DB2 API 程式唯一需要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Silicon Graphics IRIX
# Builds a C application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile with n32 object support, uncomment the following line.
# IRIX_OBJECT_MODE=-n32

if [ "$IRIX_OBJECT_MODE" = "-n32" ] ; then
  # Link with db2 n32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib32
else
  # Link with db2 o32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib
fi

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
```

```
  # Compile the utilemb.c error-checking utility.
  cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  cc $IRIX_OBJECT_MODE -o $1 $1.o utilemb.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH
-lm -ldb2
else
  # Link the program with utilapi.o
  cc $IRIX_OBJECT_MODE -o $1 $1.o utilapi.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH
-lm -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

cc 使用 C 編譯器。

$IRIX_OBJECT_MODE
如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$HOME/sqllib/include

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
$IRIX_OBJECT_MODE 如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。
-L$DB2_LIBPATH 指定在鏈結時 DB2 固定及共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-rpath $DB2_LIBPATH 指定在執行時 DB2 共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。
-lm 與計算檔案庫鏈結。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c 建置 DB2 API 非內含的 SQL 範例程式 client, 請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。

欲執行可執行檔，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
client database userid password
```

建置及執行內含的 SQL 應用程式

欲從來源檔 updat.sqc 建置範例程式 updat，請併入資料庫的參數，及含有資料庫的案例之使用者 ID 與通行碼的參數：

```
bldapp updat database userid password
```

結果是可執行檔 updat。欲對 sample 資料庫執行可可執行檔，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
updat database userid password
```

儲存程序的內含 SQL 從屬站應用程式

儲存程序是存取資料庫並傳回資訊給從屬站應用程式的程式。您要在伺服器上編譯及儲存儲存程序。此伺服器在另一個平台上執行。

欲在支援 DB2 的平台伺服器上建置內含的 SQL 儲存程序 spserver，請參照本書中有關該平台的「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了儲存程序 spserver，您就可以建置呼叫儲存程序的從屬站應用程式。您可以使用 Script 檔 bldapp，從來源檔 spclient.sqc 建置 spclient。關於詳細資訊，請參閱第274頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫儲存程序，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以執行從屬站應用程式：

```
spclient database userid password
```

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF) 的從屬站應用程式

使用者定義的函數 (UDF) 是您自己在伺服器上編譯及儲存的純量函數和表格函數。此伺服器在另一個平台上執行。 欲在支援 DB2 的平台伺服器上建置使用者定義的函數程式 udfsrv，請參照本書中有關該平台的「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了 udfsrv，您就可以使用 Script 檔 bldapp，從 sqllib/samples/c 中的 udfcli.sqc 來源檔建置內含的 SQL 從屬站應用程式 udfcli 來呼叫它。關於詳細資訊，請參閱第274頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF 程式，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以執行呼叫應用程式：

```
udfcli database userid password
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

Silicon Graphics IRIX 上的多緒應用程式，需要使用 -ldb2_th 及 -lpthread 鏈結選項，來鏈結 DB2 檔案庫的 POSIX 緒版本之 o32 或 n32 物件類型。

在 sqllib/samples/c 中的 script 檔 bldmt 含有建置內含的 SQL 多緒程式的指令。第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2，指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。

```
#! /bin/ksh
# bldmt script file -- Silicon Graphics IRIX
# Builds a C multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile with n32 object support, uncomment the following line.
# IRIX_OBJECT_MODE=-n32

if [ "$IRIX_OBJECT_MODE" = "-n32" ] ; then
  # Link with db2 n32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib32
else
  # Link with db2 o32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib
fi

# Precompile and bind the program.
embprep $1 $2 $3 $4

# Compile the program.
cc $IRIX_OBJECT_MODE -I$DB2PATH/include -c $1.c

# Link the program.
cc $IRIX_OBJECT_MODE -o $1 $1.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH -lm -ldb2_th
-lpthread
```

除了前面討論的 -ldb2_th 及 -lpthread 鏈結選項，與沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第274頁的『DB2 API 及內含的 SQL 應用程式』。

欲從來源檔 thdsrver.sqc 建置範例程式 thdsrver，請併入資料庫的參數，及含有資料庫的案例之使用者 ID 及通行碼的參數：

```
bldmt thdsrver database userid password
```

結果產生可執行檔 thdsrver。

欲執行可執行檔，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
thdsrver database userid password
```

MIPSpro C++

本節包括下列主題：

- DB2 API 及內含的 SQL 應用程式
- 多緒的應用程式

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp, 位於 sqllib/samples/cpp中，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 應用程式而言，這是唯一的必要參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。

```
#! /bin/ksh
# bldapp script file -- Silicon Graphics IRIX
# Builds a C++ application program
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile with n32 object support, uncomment the following line.
# IRIX_OBJECT_MODE=-n32

if [ "$IRIX_OBJECT_MODE" = "-n32" ] ; then
  # Link with db2 n32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib32
else
  # Link with db2 o32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib
fi

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  CC $IRIX_OBJECT_MODE -I$DB2PATH/include -c utilemb.C
else
  # Compile the utilapi.c error-checking utility.
```

```
  CC $IRIX_OBJECT_MODE -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
CC $IRIX_OBJECT_MODE -I$DB2PATH/include -c $1.C

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  CC $IRIX_OBJECT_MODE -o $1 $1.o utilemb.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH
-lm -ldb2
else
  # Link the program with utilapi.o
  CC $IRIX_OBJECT_MODE -o $1 $1.o utilapi.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH
-lm -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

CC 使用 C++ 編譯器。

$IRIX_OBJECT_MODE
如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。 例如：$HOME/sqllib/include

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldapp 的編譯及鏈結選項

鏈結選項：

CC 使用編譯器作爲鏈結器的前端。

$IRIX_OBJECT_MODE
如果 'IRIX_OBJECT_MODE=-n32' 未加上註解，則含有 "-n32"；否則，即不含任何值。

-o $1 指定可執行檔。

$1.o 併入程式目的檔。

utilemb.o
如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。

utilapi.o
如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。

-L$DB2_LIBPATH
指定在鏈結時 DB2 固定及共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。如果您未指定 -L 選項，將使用 /usr/lib:/lib。

-rpath $DB2_LIBPATH
指定在執行時 DB2 共用檔案庫的位置。若是 o32 物件類型，它指向：$DB2PATH/lib；若是 n32 物件類型，它指向：$DB2PATH/lib32。

-lm 與計算檔案庫鏈結。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 updat.sqC 建置範例程式 updat，請併入資料庫的參數，及含有資料庫的案例之使用者 ID 及通行碼的參數：

```
bldapp updat database userid password
```

結果產生可執行檔 updat，欲執行可執行檔，則請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
updat database userid password
```

儲存程序的內含 SQL 從屬站應用程式

儲存程序是存取資料庫並傳回資訊給從屬站應用程式的程式。您要在伺服器上編譯及儲存儲存程序。此伺服器在另一個平台上執行。

欲在支援 DB2 的平台伺服器上建置內含的 SQL 儲存程序 spserver，請參照本書中有關該平台「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了儲存程序 spserver，您就可以建置呼叫儲存程序的從屬站應用程式。您可以使用 Script 檔 bldapp，從來源檔 spclient.sqC 建置 spclient。關於詳細資訊，請參閱第279頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫儲存程序，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以執行從屬站應用程式：

```
spclient database userid password
```

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。儲存程序會將輸出傳回從屬站應用程式。

UDF 的內含 SQL 從屬站應用程式

使用者定義函數 (UDF) 指您在伺服器上編譯及儲存的純量函數。 此伺服器在另一個平台上執行。 欲在支援 DB2 的平台伺服器中建置使用者定義的函數程式 udfsrv，請參照本書中有關該平台的「建置應用程式」一章。至於 DB2 從屬站可存取的其它伺服器，請參閱第6頁的『支援的伺服器』。

一旦您建置了 udfsrv，您可以使用 Script 檔 bldapp，從 sqllib/samples/cpp 中的 udfcli.sqC 來源檔，建置呼叫它的內含的 SQL 從屬站應用程式 udfcli。關於詳細資訊，請參閱第279頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫 UDF 程式，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼，以執行呼叫應用程式：

```
udfcli database userid password
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

Silicon Graphics IRIX 上的多緒應用程式，需要使用 -ldb2_th 及 -lpthread 鏈結選項，來鏈結 DB2 檔案庫的 POSIX 緒版本之 o32 或 n32 物件類型。

在 sqllib/samples/cpp 中的 script 檔 bldmt，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2，指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。

```
#! /bin/ksh
# bldmt script file -- Silicon Graphics IRIX
# Builds a C++ multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile with n32 object support, uncomment the following line.
# IRIX_OBJECT_MODE=-n32

if [ "$IRIX_OBJECT_MODE" = "-n32" ] ; then
  # Link with db2 n32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib32
else
  # Link with db2 o32 object type libraries.
  DB2_LIBPATH=$DB2PATH/lib
fi

# Precompile and bind the program.
embprep $1 $2 $3 $4

# Compile the program.
CC $IRIX_OBJECT_MODE -I$DB2PATH/include -c $1.C

# Link the program.
CC $IRIX_OBJECT_MODE -o $1 $1.o -L$DB2_LIBPATH -rpath $DB2_LIBPATH -lm -ldb2_th
-lpthread
```

除了前面討論的 `-ldb2_th` 及 `-lpthread` 鏈結選項，與沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項都與內含的 SQL Script 檔 `bldapp` 所使用的相同。關於這些選項的資訊，請參閱第279頁的『DB2 API 及內含的 SQL 應用程式』。

欲從來源檔 `thdsrver.sqC` 建置範例程式 `thdsrver`，請併入資料庫的參數，及含有資料庫的案例之使用者 ID 及通行碼的參數：

```
bldmt thdsrver database userid password
```

結果產生可執行檔 `thdsrver`。

欲對 `sample` 資料庫執行可可執行檔，請輸入可執行檔名稱、資料庫名稱、及含有資料庫的案例之使用者 ID 及通行碼：

```
thdsrver database userid password
```

第12章 開發 Solaris 應用程式

本章提供在 Solaris 作業環境中建置應用程式的詳細資訊。在 script 檔中，以 db2 開頭的指令即爲命令行處理器 (CLP) 指令。 如果您需要有關 CLP 指令的詳細資訊，請參閱 *Command Reference* 一書。

若需 Solaris 作業環境的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

註:

1. -mt 多緒選項會依照 Solaris 作業環境中施行緒的方式，而用於 DB2 建置檔及 make 檔的鏈結步驟中。使用此選項，須付出一些效能成本。如果最佳效能是您的考量，則您可以試著不使用此選項，而使用非緒 libdb2.so 檔案庫來鏈結您的應用程式。 然而，如果未使用 -mt 切換，則在執行應用程式時，可能會出現下列應用程式錯誤訊息：

   ```
   libc internal error: _rmutex_unlock: rmutex not held
   ```

   或，應用程式會在沒有任何錯誤訊息的情況下中止。
2. 欲使用本章中述及的建置檔來建置 64 位元應用程式，請解除指示指令的註解。

Forte/WorkShop C

註: Forte/WorkShop C 以前稱爲 ″SPARCompiler C″。

本節包括下列主題：

- 使用 -xarch=v8plusa 選項
- DB2 CLI 應用程式
- DB2 CLI 應用程式與 DB2 API
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)
- 多緒的應用程式

使用 -xarch=v8plusa 選項

使用 Sun WorkShop C 及 C++ Compilers 時，如果您在執行可執行檔時遇到問題，收到如下錯誤：

1. 行 1 處存在語法錯誤：'(' 爲非預期的
2. ksh：<應用程式名稱>：無法執行 (其中應用程式名稱爲編譯之可執行檔的名稱)

則您可能遇到的問題是：編譯器在與 libdb2.so 鏈結時未產生有效的可執行檔。一個修正的建議是將 -xarch=v8plusa 選項新增到您的編譯及鏈結指令。例如，當編譯範例應用程式 dynamic.sqc 時：

```
embprep dynamic sample
embprep utilemb sample
cc -c utilemb.c -xarch=v8plusa -I/export/home/db2inst1/sqllib/include
cc -o dynamic dynamic.c utilemb.o -xarch=v8plusa -I/export/home/db2inst1/sqllib/include \
    -L/export/home/db2inst1/sqllib/lib -R/export/home/db2inst1/sqllib/lib -l db2
```

註:

1. 如果您是在 Solaris 上使用「SQL 程序」，且是在透過 DB2_SQLROUTINE_COMPILE_COMMAND 側錄變數使用您自己的編譯字串，請確保您併入了 -xarch=v8plusa 編譯器選項。Forte/WorkShop C 的預設編譯指令包括此選項：

   ```
   db2set DB2_SQLROUTINE_COMPILE_COMMAND=cc -xarch=v8plusa -Kpic \
     -I$HOME/sqllib/include SQLROUTINE_FILENAME.c \
   -G -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib \
   -R$HOME/sqllib/lib -ldb2
   ```

2. 若要在 Solaris 上編譯 64 位元 SQL 程序，請除去 -xarch=v8plusa 選項，並將 -xarch=v9 選項新增至上述指令中。

DB2 CLI 應用程式

sqllib/samples/cli 中的 Script 檔 bldcli，包含用來建置 DB2 CLI 程式的指令。參數 $1 指定來源檔名稱。

對沒有內含的 SQL 的 CLI 程式而言，這是唯一的必要參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

如果程式有內含的 SQL，並以 .sqc 副檔名說明，則會呼叫 embprep script 來前置編譯程式，並產生附有副檔名 .c 的程式檔。

```
#! /bin/ksh
# bldcli script file -- Solaris
# Builds a DB2 CLI program.
# Usage: bldcli <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
fi

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the error-checking utility.
cc $CFLAGS_64 -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc $CFLAGS_64 -I$DB2PATH/include -c $1.c

# Link the program.
cc $CFLAGS_64 -o $1 $1.o utilcli.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2
```

bldcli 的編譯選項和鏈結選項
編譯選項： **cc** 使用 C 編譯器。 **$CFLAGS_64** 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。 **-I$DB2PATH/include** 指定 DB2 併入檔的位置。例如： `$HOME/sqllib/include` **-c** 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。
鏈結選項： **cc** 使用編譯器作爲鏈結器的前端。 **$CFLAGS_64** 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。 **-o $1** 指定可執行檔程式。 **$1.o** 併入程式目的檔。 **utilcli.o** 包括公用程式目的檔以便檢查錯誤。 **-L$DB2PATH/lib** 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如，`$HOME/sqllib/lib` **-R$DB2PATH/lib** 指定在執行時 DB2 共用檔案庫的位置。例如，`$HOME/sqllib/lib` **-ldb2** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `tbinfo.c` 建置範例程式 `tbinfo`，請輸入：

```
bldcli tbinfo
```

結果產生可執行檔 `tbinfo`。您可以輸入下列可執行檔名稱來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

在 sqllib/samples/cli 中的 Script 檔 bldapi 內含一些指令，可以使用 DB2 API 建置 DB2 CLI 程式。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldcli script 間的唯一不同處。請參閱第287頁的『DB2 CLI 應用程式』，取得 bldapi 及 bldcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入下列可執行檔名稱來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

sqllib/samples/cli 中的 Script 檔 bldclisp，包含用來建置 DB2 CLI 儲存程序的指令。參數 $1 指定來源檔名稱。

```
#! /bin/ksh
# bldclisp script file -- Solaris
# Builds a DB2 CLI stored procedure.
# Usage: bldclisp <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the error-checking utility.
cc $CFLAGS_64 -Kpic -I$DB2PATH/include -c utilcli.c

# Compile the program.
cc $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.c

# Link the program.
cc $CFLAGS_64 -G -o $1 $1.o utilcli.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldclisp 的編譯選項和鏈結選項

編譯選項：

cc C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-Kpic 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。 本書假定編譯及鏈結是分開的步驟。

bldclisp 的編譯選項和鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilcli.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
-G 建立共用檔案庫。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置範例程式 spserver，請輸入：

```
bldclisp spserver
```

Script 檔會將儲存程序複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 spserver 後，您就可以建置呼叫儲存程序的 CLI 從屬站應用程式 spclient。

您可以使用 script 檔 bldcli 建置 spclient。詳細資訊，請參閱第287頁的『DB2 CLI 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/c 中，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Solaris
# Builds a C application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true
```

```
if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqc" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.c error-checking utility.
  cc $CFLAGS_64 -I$DB2PATH/include -c utilemb.c
else
  # Compile the utilapi.c error-checking utility.
  cc $CFLAGS_64 -I$DB2PATH/include -c utilapi.c
fi

# Compile the program.
cc $CFLAGS_64 -I$DB2PATH/include -c $1.c

if [[ -f $1".sqc" ]]
then
  # Link the program with utilemb.o
  cc $CFLAGS_64 -o $1 $1.o utilemb.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2
else
  # Link the program with utilapi.o
  cc $CFLAGS_64 -o $1 $1.o utilapi.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： `$HOME/sqllib/include`

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果不是內含的 SQL 程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c 建置 DB2 API 非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。

欲執行可執行檔，請輸入可執行檔檔名：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqc 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

```
bldapp updat database userid password
```

結果是可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

Script 檔 bldsrv，位於 sqllib/samples/c 中，含有建置內含的 SQL 儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldsrv script file -- Solaris
# Builds a C stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi
```

```
# Compile the program.
cc $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.c

# Link the program and create a shared library
cc $CFLAGS_64 -G -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-Kpic 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：`-I$DB2PATH/include`

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

鏈結選項：

cc 使用編譯器作爲鏈結器的前端。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-G 建立共用檔案庫。

-o $1 指定可執行檔。

$1.o 併入程式目的檔。

-L$DB2PATH/lib
指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：`$HOME/sqllib/lib`。如果您未指定 `-L` 選項，將使用 `/usr/lib:/lib`。

-R$DB2PATH/lib
指定在執行時 DB2 共用檔案庫的位置。例如：`$HOME/sqllib/lib`。

-ldb2 與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqc 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將儲存程序複製到 sqllib/function 目錄。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 spserver，您就可以建置呼叫儲存程序的從屬站應用程式 spclient。

您可以使用 Script 檔 bldapp 建置 spclient。關於詳細資訊，請參閱第292頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

Script 檔 bldudf，位於 sqllib/samples/c 中，含有建置 UDF 的指令。UDF 不包含內含的 SQL 陳述式。所以建置 UDF 程式，不需要連接資料庫，或是前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- Solaris
# Builds a C UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the program.
cc $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.c

# Link the program and create a shared library.
cc $CFLAGS_64 -G -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

cc C 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-Kpic 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldudf 的編譯及鏈結選項
鏈結選項：
cc 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9"；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
-ldb2apie 與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。
-G 建立共用檔案庫。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。已提供 DB2 CLI 及此程式的內含 SQL 版本。

您可以在 sqllib/samples/cli 中使用 script 檔 bldcli，從來源檔 udfcli.c 建置 DB2 CLI udfcli 程式。詳細資訊，請參閱第287頁的『DB2 CLI 應用程式』。

您可以使用 Script 檔 bldapp，從 sqllib/samples/c 中的來源檔 udfcli.sqc 建置內含的 SQL udfcli 程式。關於詳細資訊，請參閱第292頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫 UDF，請輸入下列可執行檔名稱，執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會從 udfsrv 檔案庫呼叫 ScalarUDF 函數。

多緒的應用程式

在 Solaris 上使用 Forte/WorkShop C 的多緒應用程式，必須使用 -mt 來編譯及鏈結。如此可傳送 -D_REENTRANT 到前置處理器，以及傳送 -lthread 到鏈結器。POSIX 緒也需要將 -lpthread 傳送到鏈結器。此外，使用編譯器選項 -D_POSIX_PTHREAD_SEMANTICS 還容許函數的 POSIX 變項，例如 getpwnam_r()。

Script 檔 bldmt，位於 sqllib/samples/c中，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldmt script file -- Solaris
# Builds a C multi-threaded embedded SQL program.
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the program.
cc $CFLAGS_64 -mt -D_POSIX_PTHREAD_SEMANTICS -I$DB2PATH/include -c $1.c

# Link the program.
cc $CFLAGS_64  -mt -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -lpthread
```

除了前面討論的 -mt、-D_POSIX_PTHREAD_SEMANTICS 及 -lpthread 選項，與沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項均與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第292頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqc 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。 若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

對於擁有許多連線的多緒程式，下列核心程式參數可能要設定成預設值以外的值。這些參數僅在需要這些參數的多緒程式爲區域應用程式時需要重設。

semsys:seminfo_semume
可由任何一個程序使用之號誌 (semaphore) 還原結構的限制

shmsys:shminfo_shmseg
任何一個程序均可以建立之共用記憶體區段數目的限制。

這些參數會在 /etc/system 檔案中加以設定。將以下內容用作設定這些值的指南。對於每個區域性連接，DB2 均將使用一個號誌及一個共用記憶體來通信。假設多緒應用程式爲區域應用程式，且擁有 X 條到 DB2 的連接，則該應用程式 (程序) 將需要 X 個共用記憶體及 X 個號誌還原結構來與 DB2 進行通信。所以，這兩個核心程式「參數」的值均應設定成 X + 10 (加上的 10 提供安全邊距)。

Forte/WorkShop C++

註: Forte/WorkShop C++ 以前稱爲 SPARCompiler C++

本節包括下列主題：

- 使用 -xarch=v8plusa 選項
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)
- 多緒的應用程式

使用 -xarch=v8plusa 選項

使用 Sun WorkShop C 及 C++ Compilers 時，如果您在執行可執行檔時遇到問題，收到如下錯誤：

1. 行 1 處存在語法錯誤：'(' 爲非預期的
2. ksh：<應用程式名稱>：無法執行 (其中應用程式名稱爲編譯之可執行檔的名稱)

則您可能遇到的問題是：編譯器在與 libdb2.so 鏈結時未產生有效的可執行檔。一個修正的建議是將 -xarch=v8plusa 選項新增到您的編譯及鏈結指令。例如，當編譯範例應用程式 dynamic.sqC 時：

```
embprep dynamic sample
embprep utilemb sample
CC -c utilemb.C -xarch=v8plusa -I/export/home/db2inst1/sqllib/include
CC -o dynamic dynamic.C utilemb.o -xarch=v8plusa -I/export/home/db2inst1/sqllib/include \
   -L/export/home/db2inst1/sqllib/lib -R/export/home/db2inst1/sqllib/lib -l db2
```

註:

1. 如果您是在 Solaris 上使用「SQL 程序」，且是在透過 DB2_SQLROUTINE_COMPILE_COMMAND 側錄變數使用您自己的編譯字串，請確保您併入了 -xarch=v8plusa 編譯器選項。Forte/WorkShop C++ 的預設編譯指令包括此選項：

   ```
   db2set DB2_SQLROUTINE_COMPILE_COMMAND=CC -xarch=v8plusa -Kpic \
     -I$HOME/sqllib/include SQLROUTINE_FILENAME.c \
   -G -o SQLROUTINE_FILENAME -L$HOME/sqllib/lib \
   -R$HOME/sqllib/lib -ldb2
   ```

2. 欲在 Solaris 上編譯 64 位元 SQL 程序，請拿掉 -xarch=v8plusa 選項，並將 -xarch=v9 選項新增到上面的指令。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/cpp 中，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一的必要參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Solaris
# Builds a C++ application program.
# Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]
```

```
# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqC" ]]
then
  embprep $1 $2 $3 $4
  # Compile the utilemb.C error-checking utility.
  CC $CFLAGS_64 -I$DB2PATH/include -c utilemb.C
else
  # Compile the utilapi.C error-checking utility.
  CC $CFLAGS_64 -I$DB2PATH/include -c utilapi.C
fi

# Compile the program.
CC $CFLAGS_64 -I$DB2PATH/include -c $1.C

if [[ -f $1".sqC" ]]
then
  # Link the program with utilemb.o
  CC $CFLAGS_64 -o $1 $1.o utilemb.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -mt
else
  # Link the program with utilapi.o
  CC $CFLAGS_64 -o $1 $1.o utilapi.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -mt
fi
```

bldapp 的編譯及鏈結選項

編譯選項：

CC C++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如： $HOME/sqllib/include

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldapp 的編譯及鏈結選項
鏈結選項：
CC 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
utilemb.o 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.o 如果是非內含的 SQL 程式，則有 DB2 API 公用程式物件檔，可執行錯誤檢查。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.C 建置非內含的 SQL DB2 API 範例程式 client，請輸入：

```
bldapp client
```

結果會產生一個可執行檔 client。 您可以輸入下面的指令，對 Sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqC 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

```
bldapp updat database userid password
```

結果是可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』 中有關建置 C++ 儲存程序的相關資訊。

Script 檔 bldsrv, 位於 sqllib/samples/cpp 中，含有建置內含的 SQL 儲存程序的指令。 Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在資料庫常駐的同一案例中，則您不需要使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

Script 檔會使用來源檔名稱 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- Solaris
# Builds a C++ stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true
```

```
if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the program.
  CC $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.C

# Link the program and create a shared library
CC $CFLAGS_64 -G -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -mt

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項

編譯選項：

CC C++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-Kpic 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：`-I$DB2PATH/include`

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldsrv 的編譯及鏈結選項
鏈結選項：
CC 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。
-G 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.sqC 建置範例程式 spserver，如果連接的是 sample 資料庫，請輸入：

```
bldsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv spserver database
```

Script 檔會將共用檔案庫複製到伺服器的路徑 sqllib/function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了共用檔案庫 spserver，您就可以建置從屬站應用程式 spclient，呼叫共用檔案庫中的儲存程序。

您可以使用 Script 檔 bldapp 建置 spclient。關於詳細資訊，請參閱第302頁的『DB2 API 及內含的 SQL 應用程式』。

欲呼叫共用檔案庫中的儲存程序，請輸入下列指令以執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可存取共用檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。儲存程序會將輸出傳回從屬站應用程式。

使用者定義函數 (UDF)

註: 請參閱第56頁的『UDF 及儲存程序的 C++ 考慮事項』 中建置 C++ UDF 的相關資訊。

Script 檔 bldudf，位於 sqllib/samples/cpp 中，含有建置 UDF 的指令。UDF 不包含內含的 SQL 陳述式。所以建置 UDF 程式，不需要連接資料庫，或是前置編譯及連結程式。

參數 $1 指定來源檔名稱。 Script 檔會使用來源檔名稱，作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldudf script file -- Solaris
# Builds a C++ UDF library
# Usage: bldudf <prog_name>

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
```

```
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the program.
if [[ -f $1".c" ]]
then
  CC $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.c
elif [[ -f $1".C" ]]
then
  CC $CFLAGS_64 -Kpic -I$DB2PATH/include -c $1.C
fi

# Link the program and create a shared library.
CC $CFLAGS_64 -G -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -ldb2apie

# Copy the shared library to the sqllib/function subdirectory.
# Note: the user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldudf 的編譯及鏈結選項

編譯選項：

CC C++ 編譯器。

$CFLAGS_64
如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。

-Kpic 爲共用檔案庫產生與位置無關的程式碼。

-I$DB2PATH/include
指定 DB2 併入檔的位置。例如：$HOME/sqllib/include。

-c 僅執行編譯；沒有任何鏈結。 此 script 檔有不同的編譯及鏈結步驟。

bldudf 的編譯及鏈結選項
鏈結選項：
CC 使用編譯器作爲鏈結器的前端。
$CFLAGS_64 如果 'BUILD_64BIT=true' 未加上註解，則含有 "-xarch=v9" ；否則，即不含任何值。
-G 建立共用檔案庫。
-o $1 指定可執行檔。
$1.o 併入程式目的檔。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，將使用 /usr/lib:/lib。
-R$DB2PATH/lib 指定在執行時 DB2 共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
-ldb2apie 與「DB2 API 引擎檔案庫」鏈結，以容許使用 LOB 定位器。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數程式 udfsrv，請輸入：

```
bldudf udfsrv
```

Script 檔會將 UDF 複製到 sqllib/function 目錄。

必要時，請設定 UDF 的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。您可以在 sqllib/samples/cpp 中使用 script 檔 bldapp，從來源檔 udfcli.sqC 建置 udfcli 程式。關於詳細資訊，請參閱第302頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫 UDF，請輸入下列可執行檔名稱，執行範例呼叫應用程式：

```
udfcli
```

呼叫應用程式會呼叫 udfsrv 檔案庫中的 ScalarUDF 函數。

多緒的應用程式

在 Solaris 上使用 Forte/WorkShop C++ 的多緒應用程式，必須使用 -mt 來編譯及鏈結。 如此可傳送 -D_REENTRANT 到前置處理器，以及傳送 -lthread 到鏈結器。POSIX 緒也需要將 -lpthread 傳送到鏈結器。此外，使用編譯器選項 -D_POSIX_PTHREAD_SEMANTICS 還容許函數的 POSIX 變項，例如 getpwnam_r()。

Script 檔 bldmt, 位於 sqllib/samples/cpp 中，含有建置內含的 SQL 多緒程式的指令。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。第三個參數 $3，指定資料庫的使用者 ID，而 $4 則指定通行碼。只需要第一個參數 (來源檔名稱)。資料庫名稱、使用者 ID 及通行碼均爲可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

```
#! /bin/ksh
# bldmt script file -- Solaris
# Builds a C++ multi-threaded embedded SQL program
# Usage: bldmt <prog_name> [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2 $3 $4

# To compile 64 bit programs, uncomment the following line.
# BUILD_64BIT=true

if [ "$BUILD_64BIT" != "" ]
then
  CFLAGS_64=-xarch=v9
else
  CFLAGS_64=
fi

# Compile the program.
CC $CFLAGS_64 -mt -D_POSIX_PTHREAD_SEMANTICS -I$DB2PATH/include -c $1.C

# Link the program.
CC $CFLAGS_64 -mt -o $1 $1.o -L$DB2PATH/lib -R$DB2PATH/lib -ldb2 -lpthread
```

除了前面討論的 -mt、-D_POSIX_PTHREAD_SEMANTICS 及 -lpthread 選項，與沒有公用程式檔鏈結的之外，其它的編譯及鏈結選項均與內含的 SQL Script 檔 bldapp 所使用的相同。關於這些選項的資訊，請參閱第302頁的『DB2 API 及內含的 SQL 應用程式』。

若要從來源檔 thdsrver.sqC 建置範例程式 thdsrver，請輸入：

```
bldmt thdsrver
```

結果產生可執行檔 thdsrver。 若要對 sample 資料庫執行可執行檔，請輸入下列指令：

```
thdsrver
```

對於擁有許多連接的多緒程式，下列核心程式參數可能要設定成預設值以外的值。如果需要這些參數的多緒程式爲區域應用程式，則僅需重設這些參數。

semsys:seminfo_semume
可由任何一個程序使用之號誌還原結構的限制

shmsys:shminfo_shmseg
任何一個程序均可以建立之共用記憶體區段數目的限制。

這些參數會在 /etc/system 檔案中加以設定。將以下內容用作設定這些值的指南。對於每個區域性連接，DB2 均將使用一個號誌及一個共用記憶體來通信。假設多緒應用程式爲區域應用程式，且擁有 X 條到 DB2 的連接，則該應用程式 (程序) 將需要 X 個共用記憶體及 X 個號誌還原結構來與 DB2 進行通信。所以，這兩個核心程式「參數」的值均應設定成 X + 10 (加上的 10 提供安全邊距)。

Micro Focus COBOL

本節包括下列主題：

- 使用編譯器
- DB2 API 及 DB2 內含的應用程式
- 內含的 SQL 儲存程序

使用編譯器

如果您想要開發內含 SQL 及 DB2 API 呼叫的應用程式，而且您將使用 Micro Focus COBOL 編譯器，請牢記下列幾點：

- 當您使用命令行處理器指令 db2 prep，來前置編譯您的應用程式時，請使用 target mfcob 選項 (預設值)。
- 您必須在 Micro Focus COBOL 環境變數 COBCPY 中，併入 DB2 COBOL COPY 檔案目錄。COBCPY 環境變數會指定 COPY 檔的位置。 Micro Focus COBOL 的 DB2 COPY 檔常駐在資料庫案例目錄下的 sqllib/include/cobol_mf 中。

  若要包括目錄，請輸入：

  ```
  export COBCPY=$COBCPY:$HOME/sqllib/include/cobol_mf
  ```

註: 您可能想要在 .profile 檔中設定 COBCPY。

DB2 API 及內含的 SQL 應用程式

Script 檔 bldapp，位於 sqllib/samples/cobol_mf 中，含有建置 DB2 應用程式的指令。

第一個參數 $1 指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。 建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 $2， 指定您要連接的資料庫名稱；第三個參數 $3，指定資料庫的使用者 ID，而 $4 指定通行碼。

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
#! /bin/ksh
# bldapp script file -- Solaris
# Builds a Micro Focus COBOL application program
# Usage: bldapp [ <db_name> [ <userid> <password> ]]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# If an embedded SQL program, precompile and bind it.
if [[ -f $1".sqb" ]]
then
  embprep $1 $2 $3 $4
fi

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$DB2PATH/include/cobol_mf:$COBCPY

# Compile the checkerr.cbl error-checking utility.
cob -cx checkerr.cbl

# Compile the program.
cob -cx  $1.cbl

# Link the program.
cob -x $1.o checkerr.o -L$DB2PATH/lib -ldb2 -ldb2gmf
```

bldapp 的編譯及鏈結選項	
編譯選項：	
cob	Micro Focus COBOL 編譯器。
-cx	編譯至物件模組。

bldapp 的編譯及鏈結選項
鏈結選項：
cob 使用編譯器作爲鏈結器的前端。
-x 指定可執行程式。
$1.o 併入程式目的檔。
checkerr.o 包括公用程式目的檔以便檢查錯誤。
-L$DB2PATH/lib 指定在鏈結時 DB2 固定及共用檔案庫的位置。例如：$HOME/sqllib/lib。
-ldb2 與 DB2 檔案庫鏈結。
-ldb2gmf 鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.cbl 建置非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生可執行檔 client。您可以輸入下面的指令，對 Sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果是可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

註:

1. 在 Solaris 上建置 Micro Focus 儲存程序前，請執行下列指令：

   ```
   db2stop
   db2set DB2LIBPATH=$LD_LIBRARY_PATH
   db2set DB2ENVLIST="COBDIR LD_LIBRARY_PATH"
   db2set
   db2start
   ```

   請確定 db2stop 會停止資料庫。最後發出的 db2set 指令是爲了檢查您的設定值；請確定 DB2LIBPATH 及 DB2ENVLIST 的設定正確。

2. 在 Solaris 上所使用的部份最新版本的 Micro Focus COBOL 編譯器不能用來建立靜態鏈結的儲存程序。因此， make 檔和 Script 檔 bldsrv, 已調適爲容許建立動態鏈結的儲存程序。

   爲了使遠端從屬站應用程式能順利呼叫此動態鏈結的儲存程序，必須在儲存程序所在的伺服器上呼叫 Micro Focus COBOL 常式 cobinit() 之後，才能執行儲存程序。執行 make 檔或 script 檔 bldsrv 時，即建立可完成上述動作的外層程式（wrapper program）。然後再鏈結儲存程序碼形成儲存程序共用檔案庫。由於使用此外層程式的關係，爲了讓從屬站應用程式能夠呼叫儲存程序 x，它必須呼叫 x_wrap 而不是 x。

   本節稍後將說明外層程式的明細。

在 sqllib/samples/cobol_mf 中的 script 檔 bldsrv，含有建置儲存程序的指令。Script 檔會編譯儲存程序，並將它放入可被從屬站應用程式呼叫的共用檔案庫。

第一個參數 $1 指定您的來源檔的名稱。第二個參數 $2 指定您想要與其連接的資料庫的名稱。既然儲存程序必須建置在資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

Script 檔會使用來源檔名稱 $1 作爲共用檔案庫名稱。

```
#! /bin/ksh
# bldsrv script file -- Solaris
# Builds a Micro Focus COBOL stored procedure
# Usage: bldsrv <prog_name> [ <db_name> ]

# Set DB2PATH to where DB2 will be accessed.
# The default is the standard instance path.
DB2PATH=$HOME/sqllib

# Precompile and bind the program.
embprep $1 $2

# Set COBCPY to include the DB2 COPY files directory.
export COBCPY=$DB2PATH/include/cobol_mf:$COBCPY

# Compile the program.
cob -cx  $1.cbl

# Create the wrapper program for the stored procedure.
wrapsrv $1

# Link the program creating shared library $1 with main entry point ${1}_wrap
cob -x -o $1 ${1}_wrap.c $1.o -Q -G -L$DB2PATH/lib -ldb2 -ldb2gmf

# Copy the shared library to the sqllib/function subdirectory.
# The user must have write permission to this directory.
rm -f $DB2PATH/function/$1
cp $1 $DB2PATH/function
```

bldsrv 的編譯及鏈結選項
編譯選項：
cob COBOL 編譯器。
-cx 編譯至物件模組。

bldsrv 的編譯及鏈結選項
鏈結選項：
cob 使用編譯器來鏈結編輯。
-x 產生可執行程式。
-o $1 指定可執行檔程式。
${1}_wrap.c 指定外層程式。
$1.o 指定程式目的檔。
-Q
-G
-L$DB2PATH/lib 指定 DB2 執行程式共用檔案庫的位置。例如：$HOME/sqllib/lib。如果您未指定 -L 選項，則編譯器將採用下列路徑： /usr/lib:/lib。
-ldb2 鏈結 DB2 檔案庫。
-ldb2gmf 鏈結 Micro Focus COBOL 的 DB2 例外處理程式檔案庫。
請參閱您的編譯器文件，取得其他編譯器選項。

外層程式 wrapsrv 會導致在執行儲存程序前，先呼叫 Micro Focus COBOL 常式 cobinit()。它的內容如下。

```
#! /bin/ksh
# wrapsrv script file
# Creates the wrapper program for Micro Focus COBOL stored procedures
# Usage: wrapsrv <stored_proc>

# Note: The client program calls "<stored_proc>_wrap" not "<stored_proc>"

# Create the wrapper program for the stored procedure.
cat << WRAPPER_CODE > ${1}_wrap.c
#include <stdio.h>
void cobinit(void);
int $1(void *p0, void *p1, void *p2, void *p3);

int main(void)
{
  return 0;
}

int ${1}_wrap(void *p0, void *p1, void *p2, void *p3)
{
  cobinit();
  return $1(p0, p1, p2, p3);
}
WRAPPER_CODE
```

欲從來源檔 outsrv.sqb 建置範例程式 outsrv，如果是連接到範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將儲存程序複製到伺服器的路徑 sqllib/function 中。

必要時，請設定儲存程序的檔案模式，以便從屬站應用程式可以存取它。

一旦您建置了儲存程序 outsrv，您就可以建置呼叫儲存程序的從屬站應用程式 outcli。您可以使用 script 檔 bldapp 建置 outcli。關於詳細資訊，請參閱第313頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password

爲有效的通行碼。

從屬站應用程式可以存取儲存程序檔案庫 `outsrv`，在伺服器資料庫中執行名稱相同的儲存程序函數，然後將輸出傳回從屬站應用程式。

結束儲存程序

開發儲存程序時，請使用下列陳述式來結束儲存程序：

```
move SQLZ-HOLD-PROC to return-code.
```

透過這個陳述式，儲存程序即會正確地傳回至從屬站應用程式。 當本端 COBOL 從屬站應用程式呼叫儲存程序時，這點特別重要。

第13章 開發 Windows 32 位元作業系統的應用程式

本章提供關於在 Windows 32 位元作業系統建置應用程式的詳細資訊。在批次檔中，開頭是 db2 的指令是「命令行處理器」(CLP) 指令。若需要關於 DB2 指令的其它資訊，請參閱 *Command Reference*。

關於 Windows 32 位元作業系統的最新 DB2 應用程式開發更新組件，請造訪下列網頁：

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

註:

1. 在 Windows 32 位元作業系統中的所有應用程式，包括內含的 SQL 及未內含的 SQL，都必須在 DB2 命令視窗中建置，而不是自作業系統指令提示建置。
2. 在程式中所使用的任何路徑名稱，若含有變數 %DB2PATH%，均應以雙引號括住，如： "%DB2PATH%\function"。就好像 Windows 32 位元作業系統中的 DB2 版本 7.1 的預設安裝路徑爲 \Program Files\sqllib，其中含有空格如果沒有使用雙引號，就會出現錯誤訊息，如： ″the syntax of the command is incorrect″。在本章中，這類的路徑名稱如果是當作指令或程式碼範例的一部份，則就會以雙引號括住。

WCHARTYPE CONVERT 前置編譯選項

WCHARTYPE 前置編譯選項使用 wchar_t 資料類型，處理多位元組格式或寬字元格式的圖形資料。 *Application Development Guide* 有提供本選項的其它資訊。

就 DB2 for Windows 32 位元作業系統而言，WCHARTYPE CONVERT 選項支援使用 Microsoft Visual C++ 編譯器編譯的應用程式。不過若應用程式在與資料庫字碼頁不同的字碼頁中將資料插入 DB2 資料庫，請勿對此編譯器使用 CONVERT 選項。 DB2 通常會在此狀況下執行字碼頁轉換；不過 Microsoft C 執行環境不處理某些雙位元組字元的字元替代。 這樣會導致執行轉換錯誤。

WCHARTYPE CONVERT 選項不支援使用 IBM VisualAge C++ 編譯器編譯的應用程式。 就本編譯器而言，請使用 WCHARTYPE 的 NOCONVERT 選項。透過 NOCONVERT 選項，在應用程式與資料庫管理程式之間就不會發生隱含字元轉換。從資料庫管理程式收送資料圖形主變數中的資料作爲不變的「雙位元組字集」(DBCS) 字元。

若需要將圖形資料從多位元組格式轉換成寬字元格式，請使用 wcstombs() 函數。例如：

```
wchar_t widechar[200];
wchar_t mb[200];
wcstombs((char *)mb,widechar,200);

EXEC SQL INSERT INTO TABLENAME VALUES(:mb);
```

同樣地，您可使用 mbstowcs() 函數從多位元組格式轉換成寬字元格式。

若應用程式靜態連結 C 執行程式庫，請不要從應用程式發出 setlocale() 呼叫，因爲這樣會導致 C 執行轉換錯誤。若應用程式動態連結 C 執行程式庫，那麼就可以使用 setlocale()。對於儲存程序來說也是一樣的情況。

物件連結和內嵌資料庫 (OLE DB) 表格函數

DB2 支援 OLE DB 表格函數。就這些函數而言，除了建立 CREATE FUNCTION DDL 以外不需要建置應用程式。DB2 在 %DB2PATH%\samples\oledb 目錄中提供 OLE DB 表格函數範例檔。這些是「命令行處理器」(CLP) 檔。可執行下列步驟建置它們：

1. db2 connect to database_name
2. db2 -t -v -f file_name.db2
3. db2 terminate

其中 database_name 是正在連接的資料庫，file_name 是副檔名爲 .db2 的 CLP 檔之名稱。

關於 OLE DB 表格函數的完整說明，請參閱 *Application Development Guide*。

Microsoft Visual Basic

註: DB2 AD Client for Windows 32 位元作業系統不提供適用於 Microsoft Visual Basic 的前置編譯器。

本節包含下列主題：

- ActiveX 資料物件 (ADO)
- 遠端資料物件 (RDO)
- 物件連結和內嵌 (OLE) 自動化

ActiveX 資料物件 (ADO)

「ActiveX 資料物件」(ADO) 可讓您撰寫應用程式以便透過 OLE DB 提供者存取和操作資料庫伺服器中的資料。 ADO 的主要優點是快速、容易使用、記憶體超負荷量低以及少用磁碟。

若要對 Microsoft Visual Basic 使用 ADO，您必須建立 ADO 類型檔案庫的參照。請執行下列各項：

1. 從「專案」功能表選取「參照」
2. 勾選「Microsoft ActiveX Data Objects <version_number> Library」方框
3. 按一下「確定」。

其中 <version_number> 是 ADO 檔案庫的現行版本。

完成此參照後，可以透過「VBA 物件瀏覽器」和「IDE 編輯器」存取 ADO 物件、方法和內容。

完整 Visual Basic 程式包含表格和其它圖形元素，您必須在 Visual Basic 環境中才能檢視它。以下是一些 Visual Basic 指令，這些指令作爲程式的一部份來存取 DB2 sample 資料庫（以 ODBC 來建立目錄）：

建立連接：

```
Dim db As Connection
Set db = New Connection
```

設定本端游標檔案庫提供的從屬站端游標：

```
db.CursorLocation = adUseClient
```

設定此提供者，使 ADO 使用 Microsoft ODBC Driver，然後在沒有使用者 ID/ 通行碼的情況下開啓資料庫 ″sample″；亦即使用現行使用者：

```
db.Open "SAMPLE"
```

建立記錄集：

```
Set adoPrimaryRS = New Recordset
```

使用 select 陳述式以填入記錄集：

```
adoPrimaryRS.Open "select EMPNO,LASTNAME,FIRSTNME,MIDINIT,EDLEVEL,JOB
from EMPLOYEE Order by EMPNO", db
```

程式設計師可從這裡使用 ADO 方法存取資料，例如移至下一個記錄集：

```
adoPrimaryRS.MoveNext
```

刪除記錄集中的現行記錄：

```
adoPrimaryRS.Delete
```

同時程式設計師可執行下列指令存取個別欄位：

```
Dim Text1 as String
Text1 = adoPrimaryRS!LASTNAME
```

DB2 在 %DB2PATH%\samples\ADO\VB 目錄中提供 Visual Basic ADO 範例程式。

遠端資料物件 (RDO)

「遠端資料物件」(RDO) 提供一個資訊模式以便透過 ODBC 存取遠端資料來源。RDO 提供一組物件，這些物件使您容易連接資料庫、執行查詢和儲存程序、操作結果和確定伺服器的變更。它專門用來存取遠端 ODBC 關聯式資料原始檔，使您更容易使用 ODBC 而不需要複雜應用程式碼，而且是存取一個隨 ODBC 驅動程式出現的關聯式資料庫的主要方法。 RDO 透過 Open Database Connectivity (ODBC) API 和驅動程式管理員執行一個薄的程式碼層 (thin code layer)，驅動程式管理員會使用最少的工作站資源來建立連線、建立結果集及游標，並執行複雜的程序。

若要對 Microsoft Visual Basic 使用 RDO，您必須建立 Visual Basic 專案的參照。請執行下列各項：

1. 從「專案」功能表選取「參照」
2. 勾選「Microsoft Remote Data Object <Version Number>」方框
3. 按一下「確定」。

其中 <version_number> 是現行 RDO 版本。

完整 Visual Basic 程式包含表格和其它圖形元素，您必須在 Visual Basic 環境中才能檢視它。以下是一些 Visual Basic 指令，這些指令作爲 DB2 程式的一部份，該程式連接至 sample 資料庫，開啓一個記錄集（選取 EMPLOYEE 表格的全部直欄），然後在訊息視窗逐一顯示員工姓名：

```
Dim rdoEn As rdoEngine
Dim rdoEv As rdoEnvironment
Dim rdoCn As rdoConnection
Dim Cnct$
Dim rdoRS As rdoResultset
Dim SQLQueryDB As String
```

指定連接字串：

```
Cnct$ = "DSN=SAMPLE;UID=;PWD=;"
```

設定 RDO 環境：

```
Set rdoEn = rdoEngine
Set rdoEv = rdoEn.rdoEnvironments(0)
```

連接至資料庫：

```
Set rdoCn = rdoEv.OpenConnection("", , , Cnct$)
```

指定記錄集的 SELECT 陳述式：

```
SQLQueryDB = "SELECT * FROM EMPLOYEE"
```

開啓記錄集並執行查詢：

```
Set rdoRS = rdoCn.OpenResultset(SQLQueryDB)
```

當位置不在記錄集結尾時，顯示表格中 LASTNAME、FIRSTNME 的 「訊息框」，一次顯示一位員工：

```
While Not rdoRS.EOF
MsgBox rdoRS!LASTNAME &", " &rdoRS!FIRSTNME
```

移至記錄集的下一列：

```
rdoRS.MoveNext
Wend
```

結束程式：

```
rdoRS.Close
rdoCn.Close
rdoEv.Close
```

DB2 在 %DB2PATH%\samples\RDO 目錄中提供 Visual Basic RDO 範例程式。

物件連結和內嵌 (OLE) 自動化

本節以 Microsoft Visual Basic 說明「物件連結和內嵌」(OLE) 自動化 UDF，以及說明存取儲存程序的範例 OLE 自動化控制器。

您可使用任何語言執行 OLE 自動化 UDF 和儲存程序（因爲 OLE 與語言無關），方法是顯示 OLE 自動化伺服器的方法，然後以 UDF 向 DB2 登記這些方法。支援開發 OLE 自動化伺服器的應用程式開發環境包含下列項目的某些版本：Microsoft Visual Basic、Microsoft Visual C++、Microsoft Visual J++、Microsoft FoxPro、 Borland Delphi 、Powersoft PowerBuilder 和 Micro Focus COBOL。 另外，可以透過 OLE 自動化來存取對於 OLE 而言是適當包裝的 Java bean 物件（例如 Microsoft Visual J++）。

您必須參考適當應用程式開發環境的文件，以取得關於開發 OLE 自動化伺服器的進一步資訊。關於使用 OLE 自動化來設計 DB2 的其它詳細資訊，請參閱 *Application Development Guide*。

OLE 自動化 UDF 及儲存程序

Microsoft Visual Basic 支援建立 OLE 自動化伺服器。 透過將某類別模組新增至 Visual Basic 專案，在 Visual Basic 中建立新型物件。 將公用次程序新增至此類別模組來建立方法。可以將這些公用程序登記到 DB2 爲 OLE 自動化 UDF 及儲存程序。請參閱 Microsoft Visual Basic 手冊 *Creating OLE Servers, Microsoft Corporation, 1995*，以及參閱 Microsoft Visual Basic 提供的 OLE 範例，以取得關於建立和建置 OLE 伺服器的其它文件。

DB2 在 Microsoft Visual Basic 中提供 OLE 自動化 UDF 及儲存程序自我包含的範例，這些範例位於目錄 `%DB2PATH%\samples\ole\msvb` 中。若需建置及執行 OLE 自動化 UDF 及儲存程序範例的相關資訊，請參閱 `%DB2PATH%\samples\ole` 中的 `README` 檔。

Microsoft Visual C++

本節包括下列主題：

- ActiveX 資料物件 (ADO)
- 物件連結和內嵌 (OLE) 自動化
- DB2 CLI 應用程式
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)

註: Visual C++ 編譯器是用於 `%DB2PATH%\samples\c` 及 `%DB2PATH%\samples\cpp` 目錄中所提供的 C 及 C++ 範例程式。其相同的批次檔亦位於這兩個目錄中。視指令檔的副檔名而定，檔案中含有接受 C 或 C++ 來源檔的指令。本節中所使用的批次檔是用來示範建置程式，但前兩個主題： 「ActiveX 資料物件 (ADO)」「物件鏈結及內含 (OLE) 自動化」中的除外。

ActiveX 資料物件 (ADO)

執行下列變更後，可以像一般 C++ 程式一樣地編譯那些使用 Visual C++ 的 DB2 ADO 程式。

若要使 C++ 原始程式當作 ADO 程式來執行，您可將下列匯入陳述式置於原始程式檔頂端：

```
#import "C:\program files\common files\system\ado\msado<VERSION NUMBER>.dll" \
no_namespace \
rename( "EOF", "adoEOF")
```

其中 <VERSION NUMBER> 是 ADO 程式庫的版本號碼。

編譯此程式後，使用者必須驗證 msado<VERSION NUMBER>.dll 是否在指定的路徑。另一個驗證方法是將 C:\program files\common files\system\ado 新增至環境變數 LIBPATH，然後在來源檔中使用這個較短的匯入陳述式：

```
#import <msado<VERSION NUMBER>.dll> \
no_namespace \
rename( "EOF", "adoEOF")
```

這是用於 DB2 範例程式 BLOBAccess.dsp 的方法。

有了這個 IMPORT 陳述式，DB2 程式就具有 ADO 程式庫的存取權。您現在可以像編譯其它程式一樣地編譯 Visual C++ 程式。若您也使用另一個程式設計介面，例如 DB2 API 或 DB2 CLI，請參閱本章適當的節次以瞭解關於建置程式的其它資訊。

DB2 在 %DB2PATH%\samples\ADO\VC 目錄中提供 Visual C++ ADO 範例程式。

物件連結和內嵌 (OLE) 自動化

本節以 Microsoft Visual C++ 說明「物件連結和內嵌」(OLE) 自動化 UDF，以及說明儲存程序的範例 OLE 自動化控制器。

您可使用任何語言執行 OLE 自動化 UDF 和儲存程序（因爲 OLE 與語言無關），方法是顯示 OLE 自動化伺服器的方法，然後以 UDF 向 DB2 登記這些方法。支援開發 OLE 自動化伺服器的應用程式開發環境包含下列項目的某些版本：Microsoft Visual Basic、Microsoft Visual C++、Microsoft Visual J++、Microsoft FoxPro、 Borland Delphi 、Powersoft PowerBuilder 和 Micro Focus COBOL。 另外，可以透過 OLE 自動化來存取對於 OLE 而言是適當包裝的 Java bean 物件（例如 Microsoft Visual J++）。

您必須參考適當應用程式開發環境的文件，以取得關於開發 OLE 自動化伺服器的進一步資訊。關於使用 OLE 自動化來設計 DB2 的其它詳細資訊，請參閱 *Application Development Guide*。

OLE 自動化 UDF 及儲存程序

Microsoft Visual C++ 支援建立 OLE 自動化伺服器。可以使用 Microsoft Foundation Classes 和 Microsoft Foundation Class 應用程式精靈，伺服器或執行爲 Win32 應用程式。伺服器可以是 DLL 或 EXE。關於進一步資訊，請參閱 Microsoft Visual C++ 文件和 Microsoft Visual C++ 提供的 OLE 範例。關於建置 Visual C++ UDF for DB2 的資訊，請參閱第339頁的『使用者定義函數 (UDF)』。若需使用 DB2 CLI 建置 Visual C++ 儲存程序的相關資訊，請參閱第331頁的『DB2 CLI 儲存程序』。若需建置 DB2 的 Visual C++ 內含的 SQL 儲存程序之相關資訊，請參閱第336頁的『內含的 SQL 儲存程序』。

DB2 提供在 Microsoft Visual C++ 中 OLE 自動化 UDF 及儲存程序的自我包含的範例，這些範例是位於目錄 `%DB2PATH%\samples\ole\msvc` 中。若需建置及執行 OLE 自動化 UDF 及儲存程序範例的相關資訊，請參閱 `%DB2PATH%\samples\ole` 中的 `README` 檔。

DB2 CLI 應用程式

`%DB2PATH%\samples\cli` 中的批次檔 `bldmcli.bat` 含有一些建置 DB2 CLI 程式的指令。

參數 `%1` 指定來源檔名稱。

這是唯一必要的參數，也是沒有內含的 SQL 的 CLI 程式唯一需要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 `%2`，指定您要連接的資料庫名稱；第三個參數 `%3`，指定資料庫的使用者 ID，且 `%4` 指定通行碼

如果程式含有內含的 SQL，並以 `.sqc` 或 `.sqx` 副檔名說明，則會呼叫 `embprep` 批次檔來前置編譯程式，並分別產生具有 `.c` 或 `.cxx` 副檔名的程式檔。

```
@echo off
rem  bldmcli batch file - Windows 32-bit Operating Systems
rem  Builds a CLI program with Microsoft Visual C++.
rem  Usage: bldmcli prog_name [ db_name [ userid password ]]

if exist "%1.sqc" call embprep %1 %2 %3 %4
if exist "%1.sqx" call embprep %1 %2 %3 %4

rem Compile the error-checking utility.
cl -Z7 -Od -c -W1 -D_X86=1 -DWIN32 utilcli.c

rem  Compile the program.
```

```
if exist "%1.sqx" goto cpp
cl -Z7 -Od -c -W1 -D_X86=1 -DWIN32 %1.c
goto link_step
:cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx

rem  Link the program.
:link_step
link -debug:full -debugtype:cv -OUT:%1.exe %1.obj utilcli.obj db2cli.lib
@echo on
```

bldmcli 的編譯選項和鏈結選項

編譯選項：

cl Microsoft Visual C++ 編譯器。

-Z7 產生 C7 樣式察碼偵錯程式資訊。

-Od 停用最佳化。使用一個關閉最佳化的除錯器比較容易。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-W1 設定警告層次。

-D_X86_=1
在 Intel 電腦上執行 Windows 32 位元作業系統需要的編譯器選項。

-DWIN32
Windows 32 位元作業系統需要的編譯器選項。

鏈結選項：

link 使用 32 位元鏈結器來鏈結編輯。

-debug:full
包括除錯資訊。

-debugtype:cv
指示除錯器類型。

-OUT:%1.exe
指定可執行檔。

%1.obj 包括目的檔。

utilcli.obj
包括公用程式目的檔以便檢查錯誤。

db2cli.lib
與 DB2 CLI 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldmcli tbinfo
```

結果產生可執行檔 tbinfo。您可以輸入下列可執行檔名稱來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldmcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldmcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldmcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

Script 檔 bldmapi，位於 sqllib/samples/cli 中，含有使用 DB2 API 建置 DB2 CLI 程式的指令。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldmcli 批次檔間的唯一不同處。請參閱第328頁的『DB2 CLI 應用程式』，取得 bldmapi 及 bldmcli 兩者共同的編譯及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldmapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入下列可執行檔名稱來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

%DB2PATH%\samples\cli 中的批次檔 bldmclis.bat 含有一些建置 CLI 儲存程序的指令。此批次檔對伺服器上的 DLL 建置此儲存程序。

參數 %1 指定來源檔名稱。批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldmclis.bat file - Windows 32-bit Operating Systems
rem Builds a CLI stored procedure using the Microsoft Visual C++ compiler.
rem Usage: bldmclis prog_name

if "%1" == "" goto error

rem  Compile the program.
if exist "%1.cxx" goto cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c  utilcli.c
goto link_step
:cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx  utilcli.c

rem  Link the program.
:link_step
link -debug:full -debugtype:cv -dll -out:%1.dll %1.obj utilcli.obj db2cli.lib
-def:%1.def
rem Copy the stored procedure DLL to the 'function' directory
copy %1.dll "%DB2PATH%\function"

goto exit
:error
echo Usage: bldmclis prog_name
:exit
@echo on
```

bldmclis 的編譯選項和鏈結選項
編譯選項： **cl** Microsoft Visual C++ 編譯器。 **-Z7** 產生 C7 樣式察碼偵錯程式資訊。 **-Od** 停用最佳化。使用一個關閉最佳化的除錯器比較容易。 **-c** 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。 **-W2** 設定警告層次。 **-D_X86_=1** 在 Intel 電腦上執行 Windows 32 位元作業系統需要的編譯器選項。 **-DWIN32** Windows 32 位元作業系統需要的編譯器選項。
鏈結選項： **link** 使用 32 位元鏈結器來鏈結編輯。 **-debug:full** 包括除錯資訊。 **-debugtype:cv** 指示除錯器類型。 **-OUT:%1.dll** 建置 .DLL 檔。 **%1.obj** 包括目的檔。 **utilcli.obj** 包括公用程式目的檔以便檢查錯誤。 **db2cli.lib** 與 DB2 CLI 檔案庫鏈結。 **-def:%1.def** 使用模組定義檔。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置 spserver 儲存程序，請輸入：

```
bldmclis spserver
```

批次檔會使用與 CLI 範例程式在相同目錄中的模組定義檔 spserver.def，建置儲存程序。批次檔會將儲存程序 DLL spserver.dll 複製到伺服器的路徑 %DB2PATH%\function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 spserver 後，您就可以建置呼叫儲存程序的 CLI 從屬站應用程式 spclient。

您可以使用 Script 檔 bldmcli 建置 spclient。關於詳細資訊，請參閱第328頁的『DB2 CLI 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

批次檔 bldmapp.bat，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置內含的 SQL 程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個附加的選用參數：第二個參數 %2，指定您要連接的資料庫名稱；第三個參數 %3，指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL 程式，bldmapp 會傳送參數到前置編譯並鏈結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldmapp.bat -- Windows 32-bit operating systems
rem Builds a Microsoft Visual C++ application program
rem Usage: bldmapp prog_name [ db_name [ userid password ]]

if exist "%1.sqx" goto embedded
if exist "%1.sqc" goto embedded
goto non_embedded

:embedded
rem Precompile and bind the program.
call embprep %1 %2 %3 %4
rem  Compile the program.
if exist "%1.cxx" goto cpp_emb
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c utilemb.c
goto link_embedded
:cpp_emb
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx utilemb.cxx
rem  Link the program.
:link_embedded
link -debug:full -debugtype:cv -out:%1.exe %1.obj utilemb.obj db2api.lib
goto exit

:non_embedded
rem  Compile the program.
if exist "%1.cxx" goto cpp_non
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c utilapi.c
goto link_non_embedded
:cpp_non
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx utilapi.cxx
rem  Link the program.
:link_non_embedded
link -debug:full -debugtype:cv -out:%1.exe %1.obj utilapi.obj db2api.lib
:exit
@echo on
```

bldmapp 的編譯及鏈結選項
編譯選項： **cl** Microsoft Visual C++ 編譯器。 **-Z7** 產生 C7 樣式察碼偵錯程式資訊。 **-Od** 停用最佳化。使用一個關閉最佳化的除錯器比較容易。 **-c** 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。 **-W2** 設定警告層次。 **-D_X86_=1** 在 Intel 電腦上執行 Windows 32 位元作業系統需要的編譯器選項。 **-DWIN32** Windows 32 位元作業系統需要的編譯器選項。
鏈結選項： **link** 使用 32 位元鏈結器來鏈結編輯。 **-debug:full** 包括除錯資訊。 **-debugtype:cv** 指示除錯器類型。 **-out:%1.exe** 指定檔名。 **%1.obj** 包括目的檔 **utilemb.obj** 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。 **utilapi.obj** 如果不是內含的 SQL 程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。 **db2api.lib** 與 DB2 檔案庫鏈結。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c (位於 %DB2PATH%\samples\c) 或從來源檔 client.cxx (位於 %DB2PATH%\samples\cpp) 建置 DB2 API 非內含的 SQL 範例程式 client，請輸入：

```
bldmapp client
```

結果產生可執行檔 client.exe。您可在命令行輸入下列可執行檔名稱（無副檔名）來執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從 C 來源檔 updat.sqc (位於 %DB2PATH%\samples\c) 或從 C++ 來源檔 updat.sqx (位於 %DB2PATH%\samples\cpp) 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldmapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldmapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldmapp updat database userid password
   ```

結果產生可執行檔 updat.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

批次檔 bldmsrv.bat，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置內含的 SQL 儲存程序的指令。此批次檔對伺服器上的 DLL 建置此儲存程序。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2，指定您想要與其連接的資料庫的名稱。 既然儲存程序必須建置在與資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只需要第一個參數 (來源檔名稱)。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldmsrv.bat -- Windows 32-bit operating systems
rem Builds a Microsoft Visual C++ stored procedure
rem Usage: bldmsrv prog_name [ db_name ]

rem Precompile and bind the program.
call embprep %1 %2

rem  Compile the program.
if exist "%1.cxx" goto cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c
goto link_step
:cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx

:link_step
rem  Link the program.
link -debug:full -debugtype:cv -out:%1.dll -dll %1.obj db2api.lib -def:%1.def

rem Copy the stored procedure DLL to the 'function' directory
copy %1.dll "%DB2PATH%\function"
@echo on
```

bldmsrv 的編譯及鏈結選項

編譯選項：

cl Microsoft Visual C++ 編譯器。

-Z7 產生 C7 樣式察碼偵錯程式資訊。

-Od 停用最佳化。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-W2 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。

-D_X86_=1
在 Intel 電腦上執行 Windows 32 位元作業系統需要的編譯器選項。

-DWIN32
Windows 32 位元作業系統需要的編譯器選項。

bldmsrv 的編譯及鏈結選項
鏈結選項： **link** 使用鏈結器來鏈結編輯。 **-debug:full** 包括除錯資訊。 **-debugtype:cv** 指示除錯器類型。 **-out:%1.dll** 建置 .DLL 檔。 **%1.obj** 包括目的檔。 **db2api.lib** 與 DB2 檔案庫鏈結。 **-def:%1.def** 模組定義檔。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從 C 來源檔 spserver.sqc 或 C++ 來源檔 spserver.sqx 建置 spserver 儲存程序 DLL，請輸入：

```
bldmsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldmsrv spserver database
```

批次檔會使用與範例程式在相同目錄中的模組定義檔 spserver.def 來建置 DLL。批次檔會將 DLL spserver.dll 複製到伺服器的路徑 %DB2PATH%\function。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 DLL spserver，您就可以建置呼叫它的從屬站應用程式 spclient。

您可以使用 Script 檔 bldmapp 建置 spclient。關於詳細資訊，請參閱第333頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序 DLL spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

批次檔 bldmudf，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，含有建置 UDF 的指令。

UDF 無法包含內含的 SQL 陳述式。因此，若要建置 UDF 程式，您不必連接至資料庫去前置編譯和連結程式。

批次檔使用參數 %1，該參數指定來源檔名稱。它使用來源檔名稱 %1 代表 DLL 名稱。

```
@echo off
rem bldmudf.bat -- Windows 32-bit operating systems
rem Builds a Microsoft Visual C++ user-defined function (UDF).
rem Usage: bldmudf udf_prog_name

if "%1" == "" goto error

rem  Compile the program.
if exist "%1.cxx" goto cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.c
goto link_step
:cpp
cl -Z7 -Od -c -W2 -D_X86_=1 -DWIN32 %1.cxx

:link_step
```

```
rem  Link the program.
link -debug:full -debugtype:cv -dll -out:%1.dll %1.obj db2api.lib db2apie.lib -def:%1.def
rem Copy the UDF DLL to the 'function' directory
copy %1.dll "%DB2PATH%\function"

goto exit
:error
echo Usage: bldmudf prog_name
:exit
@echo on
```

bldmudf 的編譯及鏈結選項

編譯選項：

cl Microsoft Visual C++ 編譯器。

-Z7 產生 C7 樣式察碼偵錯程式資訊。

-Od 停用最佳化。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-W2 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。

-D_X86_=1
在 Intel 電腦上執行 Windows 32 位元作業系統需要的編譯器選項。

-DWIN32
Windows 32 位元作業系統需要的編譯器選項。

bldmudf 的編譯及鏈結選項

鏈結選項：

link 使用鏈結器來鏈結編輯。

-debug:full
包括除錯資訊。

-debugtype:cv
指示除錯器類型。

-dll 建立 DLL。

-out:%1.dll
建置 .DLL 檔。

%1.obj 包括目的檔。

db2api.lib
與 DB2 檔案庫鏈結。

db2apie.lib
與 DB2 API Engine 檔案庫鏈結。

-def:%1.def
模組定義檔。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udfsrv.c 建置使用者定義的函數 udfsrv，請輸入：

```
bldmudf udfsrv
```

批次檔會使用與範例程式在相同目錄中的模組定義檔 udfsrv.def，建置使用者定義的函數。批次檔會將使用者定義的函數 DLL udfsrv.dll 複製到伺服器的路徑 %DB2PATH%\function 中。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。提供 DB2 CLI 以及本程式的內含的 SQL C 以及 C++ 版本。

您可以使用批次檔 bldmcli，從 udfcli.c 來源檔 (位於 %DB2PATH%\samples\cli) 建置 DB2 CLI udfcli 程式。關於詳細資訊，請參閱第328頁的『DB2 CLI 應用程式』。

您可以使用批次檔 bldmapp，從 udfcli.sqc 來源檔 (位於 %DB2PATH%\samples\c) 建置內含的 SQL C udfcli 程式。關於詳細資訊，請參閱第333頁的『DB2 API 及內含的 SQL 應用程式』。

您可以使用批次檔 bldmapp，從來源檔 udfcli.sqx (位於 %DB2PATH%\samples\cpp) 建置內含的 SQL C++ udfcli 程式。關於詳細資訊，請參閱第333頁的『DB2 API 及內含的 SQL 應用程式』。

若要執行 UDF，請輸入下列指令：

```
udfcli
```

呼叫應用程式會從 udfsrv DLL 呼叫 ScalarUDF 函數。

IBM VisualAge C++ 版本 3.5

本節包括下列主題：

- DB2 CLI 應用程式
- DB2 CLI 儲存程序
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序
- 使用者定義的函數 (UDF)

註: VisualAge C++ 編譯器用於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 目錄中所提供的 C 及 C++ 範例程式。 相同的批次檔已放入這兩個目錄中。視指令檔的副檔名而定，檔案中含有接受 C 或 C++ 來源檔的指令。

DB2 CLI 應用程式

%DB2PATH%\samples\cli 中的批次檔 bldvcli.bat 含有一些以 IBM VisualAge C++ 建置 DB2 CLI 程式的指令。

參數 %1 指定來源檔名稱。

對沒有內含的 SQL 的 CLI 程式而言，這是唯一的必要參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2，指定您要連接的資料庫名稱；第三個參數 %3，指定資料庫的使用者 ID，且 %4 指定通行碼

如果程式有內含的 SQL，並以 .sqc 或 .sqx 副檔名表示，則會呼叫 embprep 批次檔去前置編譯該程式，並分別產生附有 .c 或 .cxx 副檔名的程式檔。

```
@echo off
rem  bldvcli batch file - Windows 32-bit Operating Systems
rem  Builds a CLI program with IBM VisualAge C++.
rem  Usage: bldvcli prog_name

if exist "%1.sqc" call embprep %1 %2 %3 %4
if exist "%1.sqx" call embprep %1 %2 %3 %4
```

```
rem Compile the error-checking utility.
icc -c -Ti -W1 /I"%DB2PATH%\include" utilcli.c

rem  Compile the program.
if exist "%1.sqx" goto cpp
icc -c -Ti -W1 /I"%DB2PATH%\include" %1.c
goto link_step
:cpp
icc -c -Ti -W1 /I"%DB2PATH%\include" %1.cxx

rem  Link the program.
:link_step
ilink /MAP /DEBUG /ST:32000 /PM:VIO %1.obj utilcli.obj db2cli.lib
@echo on
```

bldvcli 的編譯選項和鏈結選項

編譯選項：

icc IBM VisualAge C++ 編譯器。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-Ti 產生除錯器資訊。

-W1 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。

鏈結選項：

ilink 使用資源鏈結器來鏈結編輯。

/MAP 產生對映檔。

/DEBUG 包括除錯資訊。

/ST:32000
指定至少 32 000 的堆疊大小。

/PM:VIO
可讓程式在視窗或全螢幕中執行。

%1.obj 包括目的檔。

utilcli.obj
包括公用程式目的檔以便檢查錯誤。

db2cli.lib
與 DB2 CLI 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 tbinfo.c 建置範例程式 tbinfo，請輸入：

```
bldvcli tbinfo
```

結果產生可執行檔 tbinfo。您可以輸入下列可執行檔名稱來執行可執行檔：

```
tbinfo
```

建置及執行內含的 SQL 應用程式

有三種方法可以從來源檔 dbusemx.sqc 建置內含的 SQL 應用程式 dbusemx：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldvcli dbusemx
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldvcli dbusemx database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldvcli dbusemx database userid password
   ```

結果會產生可執行檔 dbusemx。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   dbusemx
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   dbusemx database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   dbusemx database userid password
   ```

DB2 CLI 應用程式與 DB2 API

DB2 含有 CLI 範例程式，這些程式會使用 DB2 API 來建立及捨棄資料庫，以便示範在一個以上的資料庫中使用 CLI 函數。第24頁的表7 中的 CLI 範例程式說明，表示使用 DB2 API 的範例。

Script 檔 bldvapi，位於 sqllib/samples/cli 中，含有以 DB2 API 建置 DB2 CLI 程式的指令。此檔案可在含有 DB2 API 的 utilapi 公用程式檔中執行編譯及鏈結，以建立及捨棄資料庫。這是此檔案與 bldvcli 批次檔間的唯一不同處。請參閱第342頁的『DB2 CLI 應用程式』，取得 bldvapi 及 bldvcli 共同的鏈結及鏈結選項之相關資訊。

欲從來源檔 dbmconn.c 建置範例程式 dbmconn，請輸入：

```
bldvapi dbmconn
```

結果會產生可執行檔 dbmconn。您可以輸入下列可執行檔名稱來執行可執行檔：

```
dbmconn
```

DB2 CLI 儲存程序

%DB2PATH%\samples\cli 中的批次檔 bldvclis.bat 含有些建置 CLI 儲存程序的指令。此批次檔對伺服器上的 DLL 建置此儲存程序。

參數 %1 指定來源檔名稱。批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldvclis.bat file - Windows 32-bit Operating Systems
rem Builds a CLI stored procedure using the IBM VisualAge C++ compiler
rem Usage: bldvclis prog_name

if "%1" == "" goto error

rem  Compile the program.
if exist "%1.cxx" goto cpp
icc -c+ -Ti -Ge- -Gm+ -W1 %1.c  utilcli.c
goto link_step
:cpp
icc -c+ -Ti -Ge- -Gm+ -W1 %1.cxx  utilcli.c

:link_step
rem Import the library and create an export file.
rem The function name in the .def file must be decorated to be consistent
rem with the function name in the .map file. Typically, this is done by
rem prepending "_" and appending "@" and the number of bytes of arguments,
rem as in: "@16". In spserverva.def, for example, the IBM VisualAge C++
rem compiler requires "EXPORTS _outlanguage@16" and not "EXPORTS outlanguage".
ilib /GI %1va.def

rem Link the program and produce a DLL.
ilink /ST:64000 /PM:VIO /MAP /DLL %1.obj utilcli.obj %1va.exp db2cli.lib

rem Copy the stored procedure DLL to the 'function' directory
copy %1.dll "%DB2PATH%\function"

goto exit
:error
echo Usage: bldvclis prog_name
:exit
@echo on
```

bldvclis 的編譯選項和鏈結選項
編譯選項：
icc IBM VisualAge C++ 編譯器。
-c+ 僅執行編譯；沒有任何鏈結。此批次檔有不同的編譯及鏈結步驟。
-Ti 產生除錯器資訊。
-Ge- 建置 .DLL 檔。使用靜態鏈結的執行程式庫版本。
-Gm+ 與多工檔案庫鏈結。
-W1 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。
鏈結選項：
ilink 使用資源鏈結器來鏈結編輯。
/ST:64000 指定至少 64 000 的堆疊大小。
/PM:VIO 可讓程式在視窗或全螢幕中執行。
/MAP 產生對映檔。
/DLL 建置 .DLL 檔。
%1.obj 包括目的檔。
%1.exp 包括 VisualAge 匯出檔。
db2cli.lib 與 DB2 CLI 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 spserver.c 建置 spserver 儲存程序，請輸入：

```
bldvclis spserver
```

批次檔會使用與 CLI 範例程式在相同目錄中的模組定義檔 spserverva.def，建置儲存程序。 批次檔會將儲存程序 DLL spserver.dll 複製到伺服器的路徑 %DB2PATH%\function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 spserver 後，您就可以建置呼叫儲存程序的 CLI 從屬站應用程式 spclient。

您可以使用批次檔 bldvcli 建置 spclient。關於詳細資訊，請參閱第342頁的『DB2 CLI 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

spclient *database userid password*

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序檔案庫 spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

DB2 API 及內含的 SQL 應用程式

批次檔 bldvapp.bat，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，含有建置 DB2 應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2，指定您要連接的資料庫名稱；第三個參數 %3，指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL 程式，bldvapp 會傳送參數到前置編譯及鏈結批次檔，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldvapp.bat -- Windows 32-bit operating systems
rem Builds a VisualAge C++ application program
```

```
rem Usage: bldvapp prog_name [ db_name [  userid password ]]

if exist "%1.sqx" goto embedded
if exist "%1.sqc" goto embedded
goto non_embedded

:embedded
rem Precompile and bind the program.
call embprep %1 %2 %3 %4
rem  Compile the program.
if exist "%1.cxx" goto cpp_emb
icc -c -Ti -W1 %1.c utilemb.c
goto link_embedded
:cpp_emb
icc -c -Ti -W1 %1.cxx utilemb.cxx
rem  Link the program.
:link_embedded
ilink /MAP /DEBUG /ST:32000 /PM:VIO %1.obj utilemb.obj db2api.lib
goto exit

:non_embedded
rem  Compile the program.
if exist "%1.cxx" goto cpp_non
icc -c -Ti -W1 %1.c utilapi.c
goto link_non_embedded
:cpp_non
icc -c -Ti -W1 %1.cxx utilapi.cxx
rem  Link the program.
:link_non_embedded
ilink /MAP /DEBUG /ST:32000 /PM:VIO %1.obj utilapi.obj db2api.lib
:exit
@echo on
```

bldvapp 的編譯及鏈結選項

編譯選項：

icc	IBM VisualAge C++ 編譯器。
-c	僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
-Ti	產生除錯器資訊。
-W1	輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。

bldvapp 的編譯及鏈結選項
鏈結選項：
ilink 使用資源鏈結器來鏈結編輯。
/MAP 產生對映檔。
/DEBUG 包括除錯資訊。
/ST:32000 指定至少 32 000 的堆疊大小。
/PM:VIO 可讓程式在視窗或全螢幕中執行。
%1.obj 包括目的檔。
utilemb.obj 如果是內含的 SQL 程式，則有內含的 SQL 公用程式目的檔，可執行錯誤檢查。
utilapi.obj 如果不是內含的 SQL 程式，則有 DB2 API 公用程式目的檔，可執行錯誤檢查。
db2api.lib 與 DB2 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.c (位於 %DB2PATH%\samples\c) 或從來源檔 client.cxx (位於 %DB2PATH%\samples\cpp) 建置 DB2 API 非內含的 SQL 範例程式 client，請輸入：

```
bldvapp client
```

結果產生可執行檔 client.exe。您可在命令行輸入下列可執行檔名稱（無副檔名）來執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從 C 來源檔 updat.sqc (位於 %DB2PATH%\samples\c) 或從 C++ 來源檔 updat.sqx (位於 %DB2PATH%\samples\cpp) 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldvapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldvapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldvapp updat database userid password
   ```

結果產生可執行檔 updat.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

批次檔 bldvsrv.bat，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp，含有建置內含的 SQL 儲存程序的指令。批次檔將儲存程序編譯成 DLL，然後把它儲存在伺服器。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2，指定您想要與其連接的資料庫的名稱。 既然儲存程序必須建置在與資料庫常駐的同一案例中，就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldvsrv.bat -- Windows 32-bit operating systems
rem Builds a VisualAge C++ stored procedure
rem Usage: bldvsrv prog_name [ db_name ]

rem Precompile and bind the program.
call embprep %1 %2

rem  Compile the program.
if exist "%1.cxx" goto cpp
icc -c+ -Ti -Ge- -Gm+ -W1 %1.c
goto link_step
:cpp
icc -c+ -Ti -Ge- -Gm+ -W1 %1.cxx

:link_step
rem Import the library and create a definition file.
```

```
rem The function name in the .def file must be decorated to be consistent
rem with the function name in the .map file. Typically, this is done by
rem prepending "_" and appending "@" and the number of bytes of arguments,
rem for example,"@16". In spserverva.def,the IBM VisualAge C++ compiler requires
rem "EXPORTS _outlanguage@16" and not "EXPORTS outlanguage".
ilib /GI %1va.def

rem Link the program and produce a DLL.
ilink /ST:64000 /PM:VIO /MAP /DLL %1.obj %1va.exp db2api.lib

rem Copy the Stored Procedure DLL to the 'function' directory.
copy %1.dll "%DB2PATH%\function"
@echo on
```

bldvsrv 的編譯及鏈結選項

編譯選項：

icc IBM VisualAge C++ 編譯器。

-c+ 僅執行編譯；沒有任何鏈結。此批次檔有不同的編譯及鏈結步驟。

-Ti 產生除錯器資訊。

-Ge- 建置 .DLL 檔。使用靜態鏈結的執行程式庫版本。

-Gm+ 與多工檔案庫鏈結。

-W1 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。

鏈結選項：

ilink 使用資源鏈結器來鏈結編輯。

/ST:64000
指定至少 64 000 的堆疊大小。

/PM:VIO
可讓程式在視窗或全螢幕中執行。

/MAP 產生 MAP 檔。

/DLL 建置 .DLL 檔。

%1.obj 包括目的檔。

%1va.exp
VisualAge 匯出檔。

db2api.lib
與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從 C 來源檔 spserver.sqc 或 C++ 來源檔 spserver.sqx 建置 spserver 儲存程序 DLL，請輸入：

```
bldvsrv spserver
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldmsrv spserver database
```

批次檔會使用與範例程式在相同目錄中的模組定義檔 spserverva.def，建置儲存程序。批次檔會將儲存程序 DLL spserver.dll 複製到伺服器的路徑 %DB2PATH%\function 中。

下一步，在伺服器上執行 spcreate.db2，將儲存程序編目。首先，連接資料庫：

```
db2 connect to sample
```

如果儲存程序之前就已編目過，您可以下列指令捨棄它們：

```
db2 -td@ -vf spdrop.db2
```

然後，使用下列指令編目儲存程序：

```
db2 -td@ -vf spcreate.db2
```

然後，停止資料庫後再重新啓動，以辨識新的共用檔案庫。必要時，請設定共用檔案庫的檔案模式，以便 DB2 案例可以存取檔案庫。

一旦您建置了儲存程序 DLL spserver，您就可以建置呼叫它的從屬站應用程式 spclient。

您可以使用 Script 檔 bldvapp 建置 spclient。關於詳細資訊，請參閱第347頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
spclient database userid password
```

其中

資料庫 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取儲存程序 DLL spserver，並在伺服器資料庫中執行許多儲存程序函數。輸出會傳回從屬站應用程式。

使用者定義函數 (UDF)

批次檔 bldvudf，位於 %DB2PATH%\samples\c 及 %DB2PATH%\samples\cpp 中，含有建置 UDF 的指令。

UDF 無法包含內含的 SQL 陳述式。因此，欲建置 UDF 程式，您不需要連接資料庫，或前置編譯及連結程式。

參數 %1 指定來源檔名稱。批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldvudf.bat -- Windows 32-bit operating systems
rem  Builds a VisualAge C++ user-defined function (UDF)
rem Usage: bldvudf program_name

if "%1" == "" goto error

rem  Compile the program.
if exist "%1.cxx" goto cpp
icc -Ti -c+ -Ge- -Gm+ -W1 %1.c
goto link_step
:cpp
icc -Ti -c+ -Ge- -Gm+ -W1 %1.cxx

:link_step
rem Generate an import library and export file using a definition file.
rem Function(s) in the .def file are prepended with an underscore, and
rem appended with the @ sign and number of bytes of arguments (in decimal).
rem Parameters of less than four bytes are rounded up to four bytes.
rem Structure size is rounded up to a multiple of four bytes.
rem For example, function fred prototyped as: "int fred(int, int, short);"
rem would appear as:  "_fred@12" in the .def file.
rem These decorated function names can also be found in %1.map
rem after running the following ilink command without %1va.exp.
ilib /gi %1va.def

rem Link the program to a dynamic link library
ilink /ST:64000 /PM:VIO /MAP /DLL %1.obj %1va.exp db2api.lib db2apie.lib

rem Copy the UDF DLL to the 'function' directory.
copy %1.dll "%DB2PATH%\function"

goto exit
:error
echo Usage: bldvudf prog_name
:exit
@echo on
```

bldvudf 的編譯及鏈結選項
編譯選項：
icc IBM VisualAge C++ 編譯器。
-Ti 產生除錯器資訊。
-c+ 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。
-Ge- 建置 .DLL 檔。使用靜態鏈結的執行程式庫版本。
-Gm+ 與多工檔案庫鏈結。
-W1 輸出警告、錯誤以及嚴重和無法復原的錯誤訊息。
鏈結選項：
ilink 使用資源鏈結器來鏈結編輯。
/ST:64000 指定至少 64000 的堆疊大小
/PM:VIO 可讓程式在視窗或全螢幕中執行。
/MAP 產生 MAP 檔。
/DLL 建置 .DLL 檔。
%1.obj 包括目的檔。
%1va.exp 包括 VisualAge 匯出檔。
db2api.lib 與 DB2 檔案庫鏈結。
db2apie.lib 與 DB2 API Engine 檔案庫鏈結。
請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 udf.c 建置使用者定義的函數 udfsrv，請輸入：

```
bldvudf udfsrv
```

批次檔會使用與範例程式在相同目錄中的模組定義檔 udfsrv.def，建置使用者定義的函數。批次檔會將使用者定義的函數 DLL udfsrv.dll 複製到伺服器的路徑 %DB2PATH%\function 中。

一旦您建置了 udfsrv，您就可以建置呼叫它的從屬站應用程式 udfcli。提供 DB2 CLI 以及本程式的內含的 SQL C 以及 C++ 版本。

您可以使用批次檔 bldvcli，從 udfcli.c 來源檔 (位於 %DB2PATH%\samples\cli) 建置 DB2 CLI udfcli 程式。關於詳細資訊，請參閱第342頁的『DB2 CLI 應用程式』。

您可以使用批次檔 bldvapp，從 udfcli.sqc 來源檔 (位於 %DB2PATH%\samples\c) 建置內含的 SQL C udfcli 程式。關於詳細資訊，請參閱第347頁的『DB2 API 及內含的 SQL 應用程式』。

您可以使用批次檔 bldvapp，從 udfcli.sqx 來源檔 (位於 %DB2PATH%\samples\cpp) 建置內含的 SQL C++ udfcli 程式。關於詳細資訊，請參閱第347頁的『DB2 API 及內含的 SQL 應用程式』。

若要執行 UDF，請輸入下列指令：

```
udfcli
```

呼叫應用程式會從 udfsrv DLL 呼叫 ScalarUDF 函數。

IBM VisualAge C++ 版本 4.0

VisualAge C++ 版本 4 編譯器的應用程式建置資訊，亦可用於 AIX、OS/2 及 Windows 32 位元作業系統。請參閱第137頁的『VisualAge C++ 版本 4.0』 取得此資訊。

IBM VisualAge COBOL

本節包括下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

若開發含有內含的 SQL 和 DB2 API 呼叫的應用程式，而且使用 IBM VisualAge COBOL 編譯器，請記住以下幾點：

- 使用命令行處理器指令 db2 prep 前置編譯應用程式時，請使用 target ibmcob 選項（預設選項）。
- 請勿在您的來源檔中使用 Tab 鍵字元。
- 您可在來源檔中使用 PROCESS 和 CBL 關鍵字以設定編譯選項。只在直欄 8 到 72 中加入這些關鍵字。

- 如果您的應用程式僅內含 SQL，且沒有任何 DB2 API 呼叫，則您不需要使用 pgmname(mixed) 編譯選項。如果您使用 DB2 API 呼叫，則您必須使用 pgmname(mixed) 編譯選項。
- 如果您使用的是 IBM VisualAge COBOL 編譯器的「System/390 主電腦資料類型支援」特性，則您應用程式的 DB2 併入檔會位於下列目錄中：

  ```
  %DB2PATH%\include\cobol_i
  ```

  如果您是使用隨附的批次檔來建置 DB2 範例程式，則在批次檔中所指定的併入檔路徑必須變更爲指向 cobol_i 目錄，而非指向 cobol_a 目錄。

  如果您「不」是使用 IBM VisualAge COBOL 編譯器的「System/390 主電腦資料類型支援」特性，或您使用的是此編譯器的舊版本，則您應用程式的 DB2 併入檔會位於下列目錄中：

  ```
  %DB2PATH%\include\cobol_a
  ```

  指定 COPY 檔名，來併入 .cbl 副檔名，方式如下：

  ```
  COPY "sql.cbl".
  ```

DB2 API 及內含的 SQL 應用程式

批次檔 bldapp.bat，位於 %DB2PATH%\samples\cobol 中，含有建置 DB2 應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2，指定您要連接的資料庫名稱；第三個參數 %3，指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL 程式，bldapp 會傳送參數到前置編譯及連結檔案，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldapp.bat  -- Windows 32-bit operating systems
rem Builds a VisualAge COBOL application program
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

rem If an embedded SQL program, precompile and bind it.
if not exist "%1.sqb" goto compile_step
call embprep %1 %2 %3 %4

:compile_step
rem Compile the error-checking utility.
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include\cobol_a" checkerr.cbl

rem  Compile the program.
```

```
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include\cobol_a" %1.cbl

rem  Link the program.
cob2 %1.obj checkerr.obj db2api.lib
@echo on
```

bldapp 的編譯及鏈結選項

編譯選項：

cob2 IBM VisualAge COBOL 編譯器。

-qpgmname(mixed)
指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。

-c 僅執行編譯；沒有任何鏈結。本書假定編譯及鏈結是分開的步驟。

-qlib 指示編譯器處理 COPY 陳述式。

-I*路徑* 指定 DB2 併入檔的位置。例如： `-I"%DB2PATH%\include\cobol_a"`.

checkerr.cbl
編譯錯誤檢查公用程式。

鏈結選項：

cob2 使用編譯器來鏈結編輯。

checkerr.obj
包括錯誤檢查公用程式目的檔。

db2api.lib
與 DB2 檔案庫鏈結。

請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `client.cbl` 建置非內含的 SQL 範例程式 `client`，請輸入：

```
bldapp client
```

結果產生可執行檔 `client.exe`。您可以輸入下列可執行檔名稱 (不含副檔名)，對 `sample` 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 `updat.sqb` 建置內含的 SQL 應用程式 `updat`：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生是可執行檔 updat。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例上的 sample 資料庫，則只要輸入可執行檔檔名：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

批次檔 bldsrv.bat，位於 %DB2PATH%\samples\cobol 中，含有建置內含的 SQL 儲存程序的指令。批次檔將儲存程序編譯成伺服器上的 DLL。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2，指定您想要與其連接的資料庫的名稱。 既然儲存程序必須建置在與資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldsrv.bat  -- Windows 32-bit operating systems
rem Builds a VisualAge COBOL stored procedure
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2

rem  Compile the stored procedure.
cob2 -qpgmname(mixed) -c -qlib -I"%DB2PATH%\include\cobol_a" %1.cbl

rem  Link the stored procedure and create a shared library.
ilib /nol /gi:%1 %1.obj
ilink /free /nol /dll db2api.lib %1.exp %1.obj iwzrwin3.obj

rem Copy stored procedure to the %DB2PATH%\function directory.
copy %1.dll "%DB2PATH%\function"
@echo on
```

bldsrv 的編譯及鏈結選項
編譯選項： **cob2** IBM VisualAge COBOL 編譯器。 **-qpgmname(mixed)** 指示編譯器允許 CALL 具有混合字體名稱的檔案庫進入點。 **-c** 僅執行編譯；沒有任何鏈結。此批次檔有不同的編譯及鏈結步驟。 **-qlib** 指示編譯器處理 COPY 陳述式。 **-I***路徑* 指定 DB2 併入檔的位置。例如： `-I"%DB2PATH%\include\cobol_a".`
鏈結選項： **ilink** 使用 IBM VisualAge COBOL 鏈結器。 **/free** 自由格式。 **/nol** 沒有商標。 **/dll** 使用原始程式名稱來建立 DLL。 **db2api.lib** 與 DB2 檔案庫鏈結。 **%1.exp** 併入匯出檔。 **%1.obj** 併入程式目的檔。 **iwzrwin3.obj** 併入 IBM VisualAge COBOL 提供的目的檔。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 `outsrv.sqb` 建置範例程式 `outsrv`，並連接範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接的是另一個資料庫，亦請併入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將儲存程序複製到伺服器的路徑 `sqllib/function` 中。

必要時，設定儲存程序的檔案模式，以便 DB2 案例可執行它。

一旦您開發儲存程序 `outsrv`，您就可以開發從屬站應用程式 `outcli`，來呼叫儲存程序。您可以使用批次檔 `bldapp` 建置 `outcli`。關於詳細資訊，請參閱第356頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式可以存取儲存程序檔案庫 outsrv，在伺服器資料庫中執行名稱相同的儲存程序函數，然後將輸出傳回從屬站應用程式。

Micro Focus COBOL

本節包括下列主題：

- 使用編譯器
- DB2 API 及內含的 SQL 應用程式
- 內含的 SQL 儲存程序

使用編譯器

如果您想要開發內含 SQL 及 DB2 API 呼叫的應用程式，而且您將使用 Micro Focus 編譯器，請牢記下列幾點：

- 當您在使用命令行處理器指令 db2 prep 前置編譯應用程式時，請使用預設的 target mfcob 選項。
- 確定 LIB 環境變數指向 %DB2PATH%\lib，例如：

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

- Micro Focus COBOL 的 DB2 COPY 位於 %DB2PATH%\include\cobol_mf。 設定 COBCPY 環境變數來包括目錄，例如：

  ```
  set COBCPY="%DB2PATH%\include\cobol_mf;%COBCPY%"
  ```

對所有 DB2 應用程式設計介面，都必須使用呼叫慣例 74 來執行。 DB2 COBOL 前置編譯器會自動將 CALL-CONVENTION 子句插入 SPECIAL-NAMES 段落中。若 SPECIAL-NAMES 段落不存在，DB2 COBOL 會另外建立，如下所示：

```
Identification Division
Program-ID. "static".
special-names.
    call-convention 74 is DB2API.
```

同時，若有呼叫 DB2 API，前置編譯器會在 ″call″ 關鍵字後面自動加入符號 DB2API，用來識別呼叫慣例。 每當前置編譯器從內含的 SQL 陳述式中建立 DB2 API 執行呼叫，就會進行這種動作。

若 DB2 API 呼叫出現在未經過前置編譯的應用程式中，您應該在應用程式中自行建立一個如同上述的 SPECIAL-NAMES 段落。若直接呼叫 DB2 API，則需要自行在 ″call″ 關鍵字後面加上 DB2API 符號。

DB2 API 及內含的 SQL 應用程式

批次檔 bldapp，位於 %DB2PATH%\samples\cobol_mf 中，含有建置 DB2 應用程式的指令。

第一個參數 %1，指定您的來源檔的名稱。對沒有內含的 SQL 的程式而言，這是唯一必要的參數。建置內含的 SQL 程式需要連接到資料庫，所以也提供您三個選用參數：第二個參數 %2，指定您要連接的資料庫名稱；第三個參數 %3，指定資料庫的使用者 ID，且 %4 指定通行碼

對於內含的 SQL程式，bldapp 會傳送參數到前置編譯及連結批次檔，embprep。如果沒有提供任何資料庫名稱，則會使用預設的 sample 資料庫。如果建置程式的案例與資料庫所在位置的案例不同時，才需要使用者 ID 及通行碼參數。

```
@echo off
rem bldapp.bat  -- Windows 32-bit operating systems
rem Builds a Micro Focus Cobol application program
rem Usage: bldapp <prog_name> [ <db_name> [ <userid> <password> ]]

rem If an embedded SQL program, precompile and bind it.
if not exist "%1.sqb" goto compile_step
call embprep %1 %2 %3 %4

:compile_step
rem Compile the error-checking utility.
cobol checkerr.cbl;

rem  Compile the program.
cobol %1.cbl;

rem  Link the program.
cbllink -l %1.obj checkerr.obj db2api.lib
@echo on
```

bldapp 的編譯及鏈結選項
編譯選項： **cobol**　Micro Focus COBOL 編譯器。

bldapp 的編譯及鏈結選項
鏈結選項： **cbllink** 使用鏈結器來鏈結編輯。 **-l** 鏈結 lcobol 檔案庫。 **checkerr.obj** 鏈結錯誤檢查公用程式目的檔。 **db2api.lib** 鏈結 DB2 API 檔案庫。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 client.cbl 建置非內含的 SQL 範例程式 client，請輸入：

```
bldapp client
```

結果產生可執行檔 client.exe。您可以輸入下列可執行檔名稱 (不含副檔名)，對 sample 資料庫執行可執行檔：

```
client
```

建置及執行內含的 SQL 應用程式

有三種方式可以從來源檔 updat.sqb 建置內含的 SQL 應用程式 updat：

1. 如果連接同一案例上的範例資料庫，請輸入：

   ```
   bldapp updat
   ```

2. 如果連接同一案例上的不同資料庫，亦請輸入資料庫名稱：

   ```
   bldapp updat database
   ```

3. 如果連接另一案例上的資料庫，亦請輸入資料庫案例的使用者 ID 及通行碼：

   ```
   bldapp updat database userid password
   ```

結果產生可執行檔 updat.exe。

有三種方式可以執行此內含的 SQL 應用程式：

1. 如果存取同一案例中的 sample 資料庫，則只要輸入可執行檔名稱 (不需要副檔名)：

   ```
   updat
   ```

2. 如果存取同一案例上的另一個資料庫，請輸入可執行檔名稱及資料庫名稱：

   ```
   updat database
   ```

3. 如果存取另一案例上的資料庫，請輸入可執行檔名稱、資料庫名稱及資料庫案例的使用者 ID 及通行碼：

   ```
   updat database userid password
   ```

內含的 SQL 儲存程序

批次檔 bldsrv，位於 %DB2PATH%\samples\cobol_mf 中，含有建置內含的 SQL 儲存程序的指令。批次檔將儲存程序編譯成伺服器上的 DLL。

第一個參數 %1，指定您的來源檔的名稱。第二個參數 %2，指定您想要與其連接的資料庫的名稱。 既然儲存程序必須建置在與資料庫常駐的相同案例中，就不需要任何使用者 ID 及通行碼的參數。

只有第一個參數是必要的，即來源檔名稱。資料庫名稱是可選用的。如果未提供任何資料庫名稱，則程式將使用預設的資料庫 sample。

批次檔會使用來源檔名稱 %1 作爲 DLL 名稱。

```
@echo off
rem bldsrv.bat  -- Windows 32-bit operating systems
rem Builds a Micro Focus Cobol stored procedure
rem Usage: bldsrv <prog_name> [ <db_name> ]

rem Precompile and bind the program.
call embprep %1 %2

rem  Compile the stored procedure.
cobol %1.cbl /case;

rem  Link the stored procedure and create a shared library.
cbllink /d %1.obj db2api.lib

rem Copy the stored procedure to the %DB2PATH%\function directory.
copy %1.dll "%DB2PATH%\function"
@echo on
```

bldsrv 的編譯及鏈結選項
編譯選項：
cobol Micro Focus COBOL 編譯器。
/case 防止外部符號被轉換成大寫字體。

bldsrv 的編譯及鏈結選項
鏈結選項： **cbllink** 使用 Micro Focus COBOL 鏈結器來鏈結編輯。 **/d** 建立 .dll 檔案。 **db2api.lib** 鏈結 DB2 API 檔案庫。 請參閱您的編譯器文件，取得其他編譯器選項。

欲從來源檔 outsrv.sqb 建置範例程式 outsrv，如果是連接到範例資料庫，請輸入：

```
bldsrv outsrv
```

如果連接其它的資料庫，亦請輸入資料庫名稱：

```
bldsrv outsrv database
```

Script 檔會將 DLL 複製到伺服器的路徑 sqllib/function 中。

必要時，請設定 DLL 的檔案模式，以便從屬站程式可以存取它。

一旦您建置了 DLL outsrv，您就可以建置呼叫它的從屬站應用程式 outcli。您可以使用批次檔 bldapp 建置 outcli。關於詳細資訊，請參閱第361頁的『DB2 API 及內含的 SQL 應用程式』。

若要呼叫儲存程序，請輸入下列指令來執行範例從屬站應用程式：

```
outcli database userid password
```

其中

term> 爲您想要與其連接的資料庫的名稱。 此名稱可以是 sample 或它的遠端別名或其它名稱。

userid 爲有效的使用者 ID。

password
爲有效的通行碼。

從屬站應用程式會存取 DLL outsrv，並在伺服器資料庫中執行名稱相同的儲存程序函數。然後將輸出傳回從屬站應用程式。

Object REXX

Object REXX 是 REXX 語言的物件導向版本。物件導向功能已加入傳統的 REXX 中，但未改變其原有的函數及指令。 Object REXX 直譯器已經過進一步改良，提供下列額外的支援：

- 類別、物件及方法
- 傳訊及多型性
- 單一及多重繼承

Object REXX 與傳統的 REXX 完全相容。本節中提及 REXX 時，表示所有 REXX 版本，包括 Object REXX。

您不必前置編譯或連結 REXX 程式。

在 Windows NT 上，REXX 程式的開頭不需要有註解。不過，基於可移轉性的理由，每一個 REXX 程式在第一行的第一欄中，最好還是以註解做開頭。如此可分辨程式與其它平台上的批次指令：

```
/* Any comment will do. */
```

REXX 範例程式位於 `%DB2PATH%\samples\rexx` 目錄中。若要執行範例 REXX 程式 `updat`，請執行下列動作：

1. 如果它尚未執行，請經由輸入下列指令，來啓動伺服器上的資料庫管理程式：

   ```
   db2start
   ```

2. 輸入：

   ```
   rexx updat.cmd
   ```

關於 REXX 及 DB2 的進一步資訊，請參閱 *Application Development Guide* 的「REXX 程式設計」這一章。

附錄A. 關於資料庫管理程式案例

DB2 支援在同一台機器上有多個資料庫管理程式案例。資料庫管理程式案例具有它自己的架構檔、目錄及資料庫。

每一個資料庫管理程式案例可以管理數個資料庫。不過給定的資料庫僅屬於一個案例。 圖1 顯示此關係。

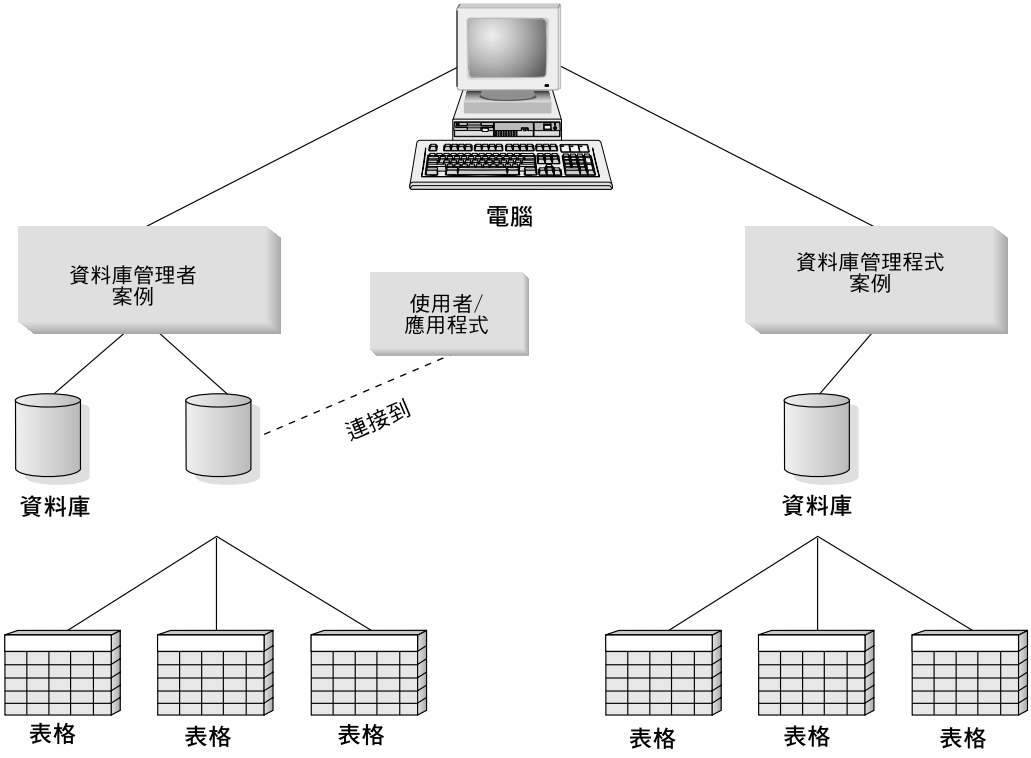

圖 1. 資料庫管理程式案例

資料庫管理程式案例使您能夠在同一台機器上具有多個資料庫環境。 例如，您可以同時擁有一個用於開發的資料庫管理程式案例及另一個用於生產的案例。

就 UNIX 伺服器而言，您就可以在不同資料庫管理程式案例上，具有不同的 DB2 版本。 例如，您可以讓某一個資料庫管理程式者執行 DB2 Universal Database 版

本 6.1，並讓另一個執行 DB2 Universal Database 版本 7.1。然而，在版本層次中，僅支援一個版次及修正層次。例如， DB2 版本 5.0 及 DB2 版本 5.2 不能並存於 UNIX 伺服器中。

若使用 OS/2、Windows NT 及 Windows 2000 伺服器，您必須在每一個資料庫管理者案例中使用相同的 DB2 版本、版次及修正層次。 您不能讓一個資料庫管理程式案例執行 DB2 Universal Database 版本 6.1，並讓另一個案例執行 DB2 Universal Database 版本 7.1。

您需要知道每一個您使用的案例的下列項目：

案例名稱

就 UNIX 平台而言，這是指您在建立資料庫管理程式案例時指定的有效使用者名稱。

就 OS/2、Windows NT 及 Windows 2000 而言，這是一個英數字串，最多 8 個字元。 在安裝期間內，會建立一個名爲 "DB2" 的案例。

案例目錄

案例位在其中的起始目錄。

就 UNIX 平台而言，案例目錄是 `$HOME/sqllib`，其中 $HOME 是案例擁有者的起始目錄。

就 OS/2、Windows NT 及 Windows 2000 而言，案例目錄爲 `%DB2PATH%\`*instance_name*。變數 `%DB2PATH%` 決定 DB2 的安裝所在。端視 DB2 安裝在哪一個磁碟機上，`%DB2PATH%` 將指向 *drive*`:\sqllib`。

OS/2、Windows NT 及 Windows 2000 上的案例路徑是依據下列之一所建立的：

`%DB2PATH%\%DB2INSTANCE%` (例如， `C:\SQLLIB\DB2`)

或是，如果定義 DB2INSTPROF：

`%DB2INSTPROF%\%DB2INSTANCE%` (例如， `C:\PROFILES\DB2`)

DB2INSTPROF 環境變數是用在 OS/2、Windows NT 及 Windows 2000 上，以支援在從屬站機器僅具有唯讀存取權的網路磁碟機上執行 DB2。 假如是這樣的話，會設定 DB2 指向 *drive*`:\sqllib`，而且設定 DB2INSTPROF 指向本端路徑（例如，`C:\PROFILES`），該路徑會包含全部案例特定資訊如目錄和架構，因爲 DB2 對這些檔案需要具有更新權。

關於建立和管理資料庫管理程式案例的資訊，請參閱您平台的 *快速入門* 書籍。

附錄B. 移轉您的應用程式

當您從版本 2 或更高版本的 DB2、DB2 Client Application Enabler 或 DB2 SDK 升級成 DB2 Universal DataBase 版本 7 時，您的資料庫及節點目錄會自動移轉。若要從 DB2 版本 1 開始移轉，必須先移轉成 DB2 Universal Database 版本 5。才能從版本 5 再移轉成版本 7。若要移轉現存資料庫，請使用 *Administration Guide* 中說明的工具。

註:

1. **DB2 版本 7.1 及 7.2**。這些版本安裝在 UNIX 系統上的相同目錄中。例如，在 AIX 上，DB2 版本 7.1 或 DB2 版本 7.2 的安裝路徑均爲 `/usr/lpp/db2_07_01/lib`。如果本書指的是 DB2 版本 7，則參照 DB2 版本 7.1 及 DB2 版本 7.2 這兩個版本。
2. **HP-UX**。如果從 HP-UX 版本 10 或更舊版本移轉 DB2 到 HP-UX 版本 11，則 DB2 程式必須以 HP-UX 版本 11 上的 DB2 重新進行前置編譯 (如果 DB2 程式包括內含的 SQL)，而且必須重新編譯。包括所有 DB2 應用程式、儲存程序、使用者定義函數及使用者跳出程式都需進行此項作業。並且，在 HP-UX 版本 11 上編譯的 DB2 程式不能在 HP-UX 版本 10 或更早版本上執行。在 HP-UX 版本 10 編譯及執行的 DB2 程式可從遠端連接到 HP-UX 版本 11 伺服器。
3. **Linux**. DB2 不支援從 DB2 Universal Database for Linux 版本 5.2 (測試版) 的移轉。
4. **Micro Focus COBOL**。透過 DB2 版本 2.1.1 或更早版本來進行前置編譯，並以 Micro Focus COBOL 進行編譯的任何現存的應用程式應該以現行版本的 DB2 重新進行前置編譯，然後以 Micro Focus COBOL 重新進行編譯。如果這些以 IBM 前置編譯器的較早版本開發的應用程式未重新進行前置編譯，則在發生異常終止時，資料庫可能會損毀。
5. 移轉的相關資訊，請參閱 *Administration Guide* 的「Appendix D. Incompatibilities Between Releases」，及針對您平台的 *快速入門* 一書的「安置規劃」及「DB2 安裝後的移轉作業」部份。

註: 下一節以及「問題」和「狀況」兩節，只適用於 UNIX 平台。

如果您有 DB2 版本 1、DB2 版本 2、DB2 版本 5 或 DB2 版本 6.1 的應用程式，並且要在同一部機器中同時執行前一版及 DB2 版本 7 的資料庫案例，則您必須對環境作某些變更。若要決定需作何種變更，請回答下列問題後，複查「狀況」這一節，查看是否有任何狀況適合您的情況。

此處以 AIX 系統來解釋所提出的要點。相同概念適用其它 UNIX 平台，但其細節可能不同，例如環境變數和特定指令。如果您不熟悉您作業系統上的這些細節，請參閱 *Administration Guide* 或 *DB2 for UNIX 快速入門* 一書中，「Planning for Installation」一章的 「Migrating from Previous Versions of DB2」一節。

問題

問題 1: 如何將前一版的應用程式鏈結到 DB2 從屬站執行程式庫，例如， AIX 中的 libdb2.a？

若要決定可執行檔的內含共用檔案庫搜尋路徑，請使用下列其中一個系統指令：

AIX /usr/bin/dump -H *executable_filename*

HP-UX

/usr/bin/chatr *executable_filename*

Linux

/usr/bin/objdump -p *executable_filename*

PTX /usr/bin/dump -Lv *executable_filename*

Silicon Graphics IRIX

/bin/elfdump -Lv *executable_filename*

Solaris

/usr/bin/dump -Lv *executable_filename*

其中 *executable_filename* 是應用程式的可執行檔名稱。

下列爲 AIX 應用程式的 DB2 版本 1 的範例傾出報表：

```
---------------------------------------------------------------------------

  dbcat:

    ***Loader Section***
                        Loader Header Information
  VERSION#         #SYMtableENT     #RELOCent        LENidSTR
  0x00000001       0x00000012       0x00000029       0x00000064

  #IMPfilID        OFFidSTR         LENstrTBL        OFFstrTBL
  0x00000004       0x000003bc       0x00000077       0x00000420


    ***Import File Strings***
  INDEX  PATH                          BASE                MEMBER
  0      /usr/lpp/db2_01_01_0000/lib:/usr/lpp/xlC/lib:/usr/lib:/lib
```

```
1                                       libc.a              shr.o
2                                       libC.a              shr.o
3                                       libdb2.a            shr.o

-------------------------------------------------------------------------
```

行 0 (零) 顯示可執行檔尋找所鏈結之共用檔案庫時搜尋的目錄路徑。行 1、2 和 3 顯示應用程式所鏈結的共用檔案庫。

根據應用程式的建立方式，您會看到下列不同路徑：/usr/lpp/db2_01_01_0000/lib 、INSTHOME/sqllib/lib (其中 INSTHOME 是資料庫案例擁有者的起始目錄) 或只有 /usr/lib:/lib 組合。

問題 2：如何在系統上架構 DB2 執行程式庫？

在安裝 DB2 版本 1、2、5、6.1 或 7 其中之一時，有一個選用性步驟，可建立系統預設的共用檔案庫路徑 /usr/lib 與 DB2 安裝路徑 (其中含有 DB2 從屬站執行程式庫) 之間的符號鏈結。

不同 DB2 版本的安裝路徑如下：

版本 1

/usr/lpp/db2_01_01_0000/lib

版本 2

/usr/lpp/db2_02_01/lib

版本 5

/usr/lpp/db2_05_00/lib

版本 6.1

/usr/lpp/db2_06_01/lib

版本 7

/usr/lpp/db2_07_01/lib

無論使用的是哪一種版本，執行時共用檔案庫皆命名爲 libdb2.a。

任何時候，只能以其中一個版本的檔案庫作爲預設值。 DB2 提供此預設值，讓您建立應用程式時，該應用程式不必依據特定的 DB2 版本。

問題 3：是否需要在環境中指定不同的搜尋路徑？

您可以在 AIX 上使用 LIBPATH 環境變數，在 HP-UX 上使用 SHLIB_PATH 環境變數，以及在 SCO UnixWare 7、Silicon Graphics IRIX 和 Solaris 上使用

LD_LIBRARY_PATH 環境變數，置換於應用程式中設定的共用檔案庫搜尋路徑 Linux、 PTX、Silicon Graphics IRIX 及 Solaris。

註: 對於在 Silicon Graphics IRIX 中的 n32 物件類型應用程式，請使用 LD_LIBRARYN32_PATH 環境變數。

您可以使用「問題 1」回覆所提供適合您平台的系統指令，來查看檔案庫搜尋路徑。

狀況

解決上面的問題後，您可能需要變更環境。請閱讀下面所列示的狀況。如果其中一個狀況適用於您的情況，請依該狀況作必要的變更。

狀況 1：如果版本 6.1 應用程式從 AIX 預設共用檔案庫路徑 /usr/lib/libdb2.a 載入共用檔案庫，且

- 如果 /usr/lib/libdb2.a 與 /usr/lpp/db2_06_01/lib/libdb2.a 間有符號鏈結, 且資料庫伺服器是 DB2 Universal Database for AIX 版本 7，請執行下列其中一項：
  - 變更符號鏈結以指向：

    ```
    /usr/lpp/db2_07_01/lib/libdb2.a
    ```

    DB2 for UNIX 快速入門 中有設定檔案庫間鏈結的相關資訊。若您是 root 使用者，您可以使用 ″db2ln″ 指令變更鏈結，如下所示：

    ```
    /usr/lpp/db2_07_01/cfg/db2ln
    ```

  - 設定 LIBPATH 環境變數以指向 /usr/lpp/db2_07_01/lib 或 INSTHOME/sqllib/lib，其中 INSTHOME 是版本 7 DB2 案例擁有者的起始目錄。
  - 在應用程式 (從屬站) 案例與伺服器案例之間架構 TCP/IP 連接。關於架構 TCP/IP 的資訊，請參閱 *安裝與架構補充資料*。
- 如果 /usr/lib/libdb2.a 與 /usr/lpp/db2_07_01/lib/libdb2.a 間有符號鏈結，且資料庫伺服器是 DB2 版本 6.1，則在應用程式 (從屬站) 案例與伺服器案例間架構一個 TCP/IP 連接。關於架構 TCP/IP 的資訊，請參閱 *安裝與架構補充資料*。

狀況 2：如果版本 6.1 應用程式從 DB2 版本 6.1 案例擁有者的 $HOME 路徑 ($HOME/sqllib/lib/libdb2.a) 載入共用檔案庫，且資料庫伺服器是 DB2 Universal Database for AIX 版本 7，請執行下列其中一項：

- 將應用程式案例移轉成與資料庫伺服器案例相同的版本。

- 設定 LIBPATH 環境變數以指向 /usr/lpp/db2_07_01/lib 或 INSTHOME/sqllib/lib，其中 INSTHOME 是版本 7 案例擁有者的起始目錄。
- 在應用程式 (從屬站) 案例與伺服器案例之間架構 TCP/IP 連接。關於架構 TCP/IP 的資訊，請參閱 *安裝與架構補充資料*。

狀況 3：如果版本 6.1 應用程式從 DB2 版本 6.1 安裝路徑 (/usr/lpp/db2_06_01/lib/libdb2.a) 載入共用檔案庫，且資料庫伺服器是 DB2 Universal Database for AIX 版本 7，請執行下列其中一項：

- 設定 LIBPATH 環境變數以指向 /usr/lpp/db2_07_01/lib 或 INSTHOME/sqllib/lib，其中 INSTHOME 是資料庫案例擁有者的起始目錄。
- 在應用程式 (從屬站) 案例與伺服器案例之間架構 TCP/IP 連接。關於架構 TCP/IP 的資訊，請參閱 *安裝與架構補充資料*。

狀況 4：如果版本 6.1 應用程式從 DB2 Universal Database for AIX 版本 7 安裝路徑 (/usr/lpp/db2_07_01/lib/libdb2.a) 載入共用檔案庫，且資料庫伺服器是 DB2 版本 6.1，則在應用程式 (從屬站) 案例與伺服器案例間架構 TCP/IP 連接。關於架構 TCP/IP 的資訊，請參閱 *安裝與架構補充資料*。

其它移轉注意事項

開發應用程式時，請考量下列各點。這些注意事項會讓您的應用程式更具可攜性：

- 對 UNIX 而言，在應用程式中只使用預設路徑 /usr/lib:/lib。在 OS/2 及 Windows 32 位元作業系統中，請使用下列指令確定 LIB 環境變數是指向 %DB2PATH%\lib：

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

  並且在預設路徑及您使用的 DB2 版本之間建立符號鏈結。確定鏈結是連到您應用程式所需要的最小 DB2 層次。關於設定鏈結的資訊，請參閱關於您平台的*快速入門*書籍。
- 如果您的應用程式需要特定的 DB2 版本，請在您的應用程式中編寫所指定 DB2 版本的路徑。例如，如果您的 AIX 應用程式需要 DB2 版本 5，請編寫 /usr/lpp/db2_05_00/lib。通常，您不需要這麼做。
- 當您在建置生產而非內部開發的應用程式時， 應用程式中的路徑不應指向 UNIX 中 sqllib/lib 目錄的案例擁有者拷貝，或指向 OS/2 及 Windows 32 位元作業系統的 %DB2PATH%\lib 目錄。這會讓應用程式高度依賴特定使用者名稱和環境。

- 一般而言，不要使用 LIBPATH 環境變數，或 Windows 32 位元作業系統的 LIB 環境變數，變更特定環境下的搜尋路徑。此變數會置換在環境中執行的應用程式所指定的搜尋路徑。 這些應用程式可能找不到所需的檔案庫或檔案。
- 在 DB2 Universal Database 版本 6.1 及 7 中，含有字串語意的所有字元陣列項目都是 char 類型，而非其它變項，例如 unsigned char。您使用 DB2 Universal Database 版本 6.1 或版本 7 編寫的任何應用程式都應遵循此慣例。

  如果您具有使用 unsigned char 的 DB2 版本 1 應用程式，則編譯器會發出警告或錯誤訊息，因爲版本 1 應用程式中的 unsigned char，與版本 6.1 或版本 7 函數原型中的 char 之間發生類型衝突。 如果發生這種情形，請使用編譯器選項 `-DSQLOLDCHAR` 來解決問題。
- 請參照 *SQL Reference*，取得 DB2 Universal Database 版本 7 及舊版 DB 2 間的不相容列示。請參照 *Administrative API Reference*，取得 DB2 Universal Database 版本 7 及舊版 DB2 間 API 不相容的列示。

附錄C. 問題與解決方案

當建立或執行您的應用程式時，您可能會遇到下列問題：

- 從屬站或伺服器問題，如在開發期間或執行您的應用程式時，無法與資料庫連接。
- 作業系統問題，如在開發期間，找不到檔案。
- 在開發期間發生的編譯器選項問題。
- 在開發期間或執行您的應用程式時，所發生的語法問題及程式編寫問題。

您可以使用下列資訊來源，來解決這些問題：

建置檔案

對於連接資料庫、前置編譯、編譯、鏈結及連結等建置問題，您可以使用本書提供的建置檔案，察看可用的命令行處理器指令及編譯器選項。

編譯器文件

適用於未被建置 Script 檔所涵蓋的編譯器選項問題。

應用程式開發手冊

請參閱 *Application Development Guide*，取得語法及其他程式編寫問題。

CLI Guide and Reference

關於 CLI 程式的語法、CLI 追蹤機能、架構關鍵字及程式撰寫問題，請參閱 *CLI Guide and Reference*。

SQL Reference

請參閱 *SQL Reference*，取得 SQL 陳述式及函數的語法相關資訊。

SQLCA 資料結構

如果您的應用程式發出 SQL 陳述式，或是呼叫資料庫管理程式 API，它必須經由檢查 SQLCA 資料結構，來檢查是否有錯誤狀況發生。

SQLCA 資料結構會在 SQLCODE 及 SQLSTATE 欄位中，傳回錯誤資訊。在每一個 SQL 陳述式執行後，且在大多數資料庫管理程式 API 呼叫後，資料庫管理程式即會更新結構。

您的應用程式可以擷取及列印錯誤資訊，或是在螢幕上顯示它。請參閱 *Application Development Guide*，取得詳細資訊。

線上錯誤訊息

不同的 DB2 元件，包括資料庫管理程式、資料庫管理公用程式、安裝與架構程序，以及命令行處理器，會產生線上錯誤訊息。這些訊息的每一個都

具有唯一字首，以及緊接在字首後面的 4 或 5 位數訊息碼。訊息碼後面顯示單一字母，指出錯誤的嚴重性。

您可以經由輸入下列指令，使用命令行處理器來查看訊息說明：

```
db2 "? xxxnnnn"
```

其中 xxx 是訊息字首，而 nnnn 是訊息號碼。 您必須加上引號。

關於 DB2 錯誤訊息的完整列示與說明，請參閱*訊息參考手冊*。

偵錯工具程式與錯誤日誌

這些是針對其他資訊來源仍無法解決的建置問題或執行問題。診斷工具包括追蹤機能、系統日誌及錯誤日誌。 DB2 會依據優先順序及發生的前後，將錯誤及警告狀況放在錯誤日誌中。請參閱 *Troubleshooting Guide*，取得詳細資訊。也有一個特別適用於爲了除錯 CLI 程式的 CLI 追蹤機能。關於詳細資訊，請參閱 *CLI Guide and Reference*。

附錄D. 使用 DB2 檔案庫

DB2 Universal Database 檔案庫是由線上說明、手冊 (PDF 及 HTML)及 HTML 格式的範例程式所組成。本節將描述此檔案庫所提供的資訊，以及存取此檔案庫的方法。

若要取得線上產品資訊，您可以使用「資訊中心」。相關資訊，請參閱第390頁的『用資訊中心來存取資訊』。您可以在 Web 上檢視作業資訊、疑難排解資訊、範例程式及 DB2 資訊。

DB2 PDF 檔案與列印的書籍

DB2 資訊

下列表格將 DB2 書籍分成四類：

DB2 手冊與參考資訊

這些書籍包含所有平台的一般 DB2 資訊。

DB2 安裝與架構資訊

這些書籍適用於特定平台上的 DB2。例如，針對各個不同的作業平台*快速入門*如 OS/2、 Windows、UNIX 等的書籍。

HTML 格式的跨平台範例程式

這些範例爲 HTML 版的範例程式，會隨 Application Development Client 一起安裝。這些範例爲參考用資訊，並不會取代實際的程式。

版本注意事項

這些檔案包含 DB2 書籍中未包含的最新資訊。

您可以從產品 CD-ROM 中，直接檢視 HTML 格式的安裝手冊、版次注意事項及教學指導。大部份的書籍以 HTML 格式存在產品 CD-ROM 中，以供檢視，而以 Adobe Acrobat (PDF) 格式存在 DB2 出版品 CD-ROM 中，供檢視與列印。您也可以從 IBM 訂購印刷的書籍；請參閱 第386頁的『訂購印刷書籍』。下表會列出可以訂購的書籍。

在 OS/2 及 Windows 平台上，您可以在 sqllib\doc\html 目錄中安裝 HTML 檔案。DB2 資訊會轉換爲不同的語言；然而，不是所有資訊都可以轉換成每一種語言。該資訊無特定語言版本時，則提供英文資訊

在 UNIX 平台中，您可以在 doc/%L/html 目錄中安裝多種語言版本的 HTML 檔案，其中 *%L* 代表語言環境。若需其餘相關資訊，請參照適當的*快速入門*書籍。

您可以使用不同方式，取得 DB2 書籍及存取資訊：

- 第389頁的『檢視線上資訊』
- 第393頁的『搜尋線上資訊』
- 第386頁的『訂購印刷書籍』
- 第385頁的『列印 PDF 書籍』

表 18. DB2 資訊

名稱	說明	書號 PDF 檔名	HTML 目錄
	DB2 手冊與參考資訊		
Administration Guide	*Administration Guide: Planning* 提供資料庫概念的綜覽、設計事項的相關資訊 (如邏輯及實體資料庫設計) 及高可用性的討論。 *Administration Guide: Implementation* 提供施行事項的相關資訊，如施行您的設計、存取資料庫、審核、備份及回復。 *Administration Guide: Performance* 提供資料庫環境及應用程式效能評估及調整的相關資訊。 您可洽北美服務中心，訂購這三本英文版的 *Administration Guide*，書號爲 SBOF-8934。	SC09-2946 db2d1x70 SC09-2944 db2d2x70 SC09-2945 db2d3x70	db2d0
Administrative API Reference	說明您可以用來管理資料庫的 DB2 應用程式設計介面 (API) 及資料結構。本書也解釋如何從應用程式呼叫 API。	SC09-2947 db2b0x70	db2b0
應用程式開發手冊	提供環境安裝資訊以及逐步的指示，教您如何在 Windows、OS/2 及 UNIX 平台上，編譯、鏈結及執行 DB2 應用程式。	SC40-0493 db2axx70	db2ax
APPC、CPI-C 與 SNA Sense Codes	提供有關您使用 DB2 Universal Database 產品時，可能會遇到之 APPC、CPI-C 及 SNA 感應碼的一般資訊。 只提供 HTML 格式。	沒有書號 db2apx70	db2ap

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
Application Development Guide	解釋如何使用內含的 SQL 或 Java (JDBC 及 SQLJ) 開發存取 DB2 資料庫的應用程式。討論主題包含在分段的環境中，或使用聯合系統撰寫儲存程序、撰寫使用者定義功能、建立使用者定義類型、使用觸發函式及開發應用程式。	SC09-2949 db2a0x70	db2a0
CLI Guide and Reference	說明如何使用 DB2 CLI 這個可呼叫的 SQL 介面 (與 Microsoft ODBC 規格相容) 來發展可存取 DB2 資料庫的應用程式。	SC09-2950 db2l0x70	db2l0
Command Reference	解釋如何使用「命令行處理器」，並說明您可以用來管理資料庫的 DB2 指令。	SC09-2951 db2n0x70	db2n0
連接環境補充資料	提供有關如何使用 DB2 for AS/400、 DB2 for OS/390、DB2 for MVS 或 DB2 for VM 作爲使用 DB2 Universal Database 伺服器的 DRDA 應用程式要求程式的設定及參考資料。本書亦詳細說明如何使用 DRDA 應用程式伺服器與 DB2 Connect 應用程式要求程式。 僅提供 HTML 及 PDF 格式。	沒有書號 db2h1x70	db2h1
Data Movement Utilities Guide and Reference	解釋如何使用 DB2 公用程式，如匯入、匯出、載入、AutoLoader 及 DPROP，以便利資料的移動。	SC09-2955 db2dmx70	db2dm
資料倉儲中心管理手冊	提供使用「資料倉儲中心」，如何開發及維護資料倉儲的相關資訊。	SC40-4096 db2ddx70	db2dd
資料倉儲中心 Application Integration Guide	提供相關資訊，協助程式設計師整合應用程式與「資料倉儲中心」及「資訊型錄管理程式」。	SC26-9994 db2adx70	db2ad
DB2 Connect User's Guide	提供有關 DB2 Connect 產品的概念、程式設計及一般使用資訊。	SC09-2954 db2c0x70	db2c0
DB2 Query Patroller Administration Guide	提供 DB2 Query Patroller 系統的作業概觀、特定的作業及管理資訊與作業資訊，供管理圖形式使用者介面公用程式使用。	SC09-2958 db2dwx70	db2dw

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
DB2 Query Patroller User's Guide	說明如何使用 DB2 Query Patroller 的工具及功能。	SC09-2960 db2wwx70	db2ww
名詞解釋	提供在 DB2 及其元件中所使用的術語定義。 提供 HTML 格式及在 *SQL Reference* 中讀取。	沒有書號 db2t0x70	db2t0
Image, Audio, 與 Video Extenders 管理與程式設計手冊	提供有關 DB2 擴充元的一般資訊，並提供有關管理及架構映像檔、音效及影像 (IAV) 擴充元，及有關利用 IAV 擴充元進行程式設計的資訊。它包含了參考資料、診斷資訊 (附有訊息) 及範例。	SC40-0525 dmbu7x70	dmbu7
資訊型錄管理程式管理手冊	提供有關管理資訊型錄的指引。	SC40-0497 db2dix70	db2di
資訊型錄管理程式 Programming Guide and Reference	提供「資訊型錄管理程式」的架構介面定義。	SC26-9997 db2bix70	db2bi
資訊型錄管理程式使用手冊	提供使用「資訊型錄管理程式」使用者介面的相關資訊。	SC40-0498 db2aix70	db2ai
安裝與架構補充資料	指引您規劃、安裝及設定特定平台的 DB2 從屬站。此補充資訊也包含了連結、設定從屬站及伺服器通信、 DB2 GUI 工具、DRDA AS、分散式安裝、架構分散式要求及存取不同資料來源等的相關資訊。	GC40-0480 db2iyx70	db2iy
訊息參考手冊	列示由 DB2、資訊型錄管理程式及資料倉儲中心所發出的訊息與訊息碼，並說明您應採取的動作。 您可洽北美服務中心，訂購這兩種英文版的訊息參考手冊，書號爲 SBOF-8932。	第一冊 GC40-0491 db2m1x70 第二冊 GC40-0492 db2m2x70	db2m0
OLAP Integration Server Administration Guide	解釋如何使用 OLAP Integration Server 的「管理管理程式」(Administration Manager) 元件。	SC27-0787 db2dpx70	無

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
OLAP Integration Server Metaoutline User's Guide	解釋如何使用標準 OLAP Metaoutline 介面 (不使用「Metaoutline 輔助程式」) 建立及移入 OLAP Metaoutline。	SC27-0784 db2upx70	無
OLAP Integration Server Model User's Guide	解釋如何利用標準「OLAP 模型介面」 (而不使用「模型輔助程式」) 來建立 OLAP 模型。	SC27-0783 db2lpx70	無
OLAP 安裝與使用手冊	提供「OLAP 起始者套件」(OLAP Starter Kit) 的架構及設定資訊。	SC40-0520 db2ipx70	db2ip
OLAP Spreadsheet Add-in for Excel 使用手冊	說明如何使用 Excel 試算表程式來分析 OLAP 資料。	SC40-0548 db2epx70	db2ep
OLAP Spreadsheet Add-in for Lotus 1-2-3 使用手冊	說明如何使用 Lotus 1-2-3 試算表程式來分析 OLAP 資料。	SC40-0547 db2tpx70	db2tp
Replication 指南與參考手冊	提供 DB2 所附之「IBM 抄寫工具」的規劃、架構、管理及使用資訊。	SC40-0499 db2e0x70	db2e0
Spatial Extender 使用與參考手冊	提供有關安裝、架構、管理、程式設計及疑難排解 Spatial Extender 的資訊。亦提供空間資料概念的重要說明，並附有 Spatial Extender 的特定參考資料 (訊息及 SQL)。	SC40-0527 db2sbx70	db2sb
SQL 入門	介紹 SQL 概念，並提供許多建構及作業的範例。	SC40-0494 db2y0x70	db2y0
SQL Reference, 第一冊及第二冊	敘述 SQL 語法、語意與語言的規則。本書也包含版本間不相容處、產品限制及目錄畫面等相關資訊。 您可洽北美服務中心，訂購這兩本英文版的 *SQL 參考手冊*，書號爲 SBOF-8933。	第一冊 SC09-2974 db2s1x70 第二冊 SC09-2975 db2s2x70	db2s0
System Monitor Guide and Reference	敘述如何收集關於資料庫與資料庫管理程式的各種資訊。本書解釋如何使用該資訊來了解資料庫活動、增進效能並判斷問題產生的原因。	SC09-2956 db2f0x70	db2f0

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
Text Extender 管理與程式設計手冊	提供有關 DB2 擴充元的一般資訊，並附有關管理及架構 Text Extender，及有關使用 Text Extender 進行程式設計的資訊。它包含了參考資料、診斷資訊 (附有訊息) 及範例。	SC40-0526 desu9x70	desu9
Troubleshooting Guide	協助您判斷錯誤的來源、從問題中回復，以及透過「DB2 客戶服務」的諮詢來使用診斷工具。	GC09-2850 db2p0x70	db2p0
新特性介紹	說明 DB2 Universal Database 版本 7 中的新特性、功能及加強功能。	SC40-0495 db2q0x70	db2q0
	DB2 安裝與架構資訊		
DB2 Connect Enterprise Edition for OS/2 與 Windows 快速入門	提供在 OS/2 及 Windows 32 位元作業系統上， DB2 Connect Enterprise Edition 的規劃、移轉、安裝與架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0479 db2c6x70	db2c6
DB2 Connect Enterprise Edition for UNIX 快速入門	提供在 UNIX 系列平台上， DB2 Connect Enterprise Edition 的規劃、移轉、安裝、架構及作業資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0478 db2cyx70	db2cy
DB2 Connect Personal Edition 快速入門	提供在 OS/2 及 Windows 32 位元作業系統上， DB2 Connect Personal Edition 的規劃、移轉、安裝、架構及作業資訊。本書亦包含所有支援從屬站的安裝及設定資訊。	GC40-0486 db2c1x70	db2c1
DB2 Connect Personal Edition Quick Beginnings for Linux	提供在所有支援 Linux 分送式系統上， DB2 Connect Personal Edition 的規劃、安裝、移轉及架構資訊。	GC09-2962 db2c4x70	db2c4
DB2 Data Links Manager 快速入門	提供 DB2 Data Links Manager 在 AIX 及 Windows 32 位元作業系統上的規劃、安裝、架構及作業資訊。	GC40-0485 db2z6x70	db2z6
DB2 Enterprise - Extended Edition for UNIX 快速入門	提供 DB2 Enterprise - Extended Edition 在 UNIX 系列平台上的規劃、安裝及架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0483 db2v3x70	db2v3

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
DB2 Enterprise - Extended Edition for Windows 快速入門	提供 DB2 Enterprise - Extended Edition 在 Windows 32 位元作業系統上的規劃、安裝及架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0482 db2v6x70	db2v6
DB2 for OS/2 快速入門	提供 OS/2 作業系統上之 DB2 Universal Database Personal Edition 的規劃、安裝、移轉及架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0487 db2i2x70	db2i2
DB2 for UNIX 快速入門	提供 UNIX 平台上 DB2 Universal Database Personal Edition 的規劃、安裝、移轉及架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0489 db2ixx70	db2ix
DB2 for Windows 快速入門	提供 DB2 Universal Database 在 Windows 32 位元作業系統上的規劃、安裝、移轉及架構資訊。本書亦包含了許多支援從屬站的安裝及設定資訊。	GC40-0490 db2i6x70	db2i6
DB2 Personal Edition 快速入門	提供 DB2 Universal Database Personal Edition 在 OS/2 及 Windows 32 位元作業系統上的規劃、安裝、移轉及架構資訊。	GC40-0488 db2i1x70	db2i1
DB2 Personal Edition Quick Beginnings for Linux	提供 DB2 Universal Database Personal Edition 在所有支援 Linux 分散式系統上的規劃、安裝、移轉及架構資訊。	GC09-2972 db2i4x70	db2i4
DB2 Query Patroller 安裝手冊	提供有關 DB2 Query Patroller 的安裝資訊。	GC40-0481 db2iwx70	db2iw
DB2 Warehouse Manager 安裝手冊	提供有關倉儲代理程式、倉儲轉換程式及「資訊型錄管理程式」的安裝資訊。	GC40-0521 db2idx70	db2id
	HTML 格式的跨平台範例程式		

表 18. DB2 資訊 (繼續)

名稱	說明	書號 PDF 檔名	HTML 目錄
HTML 格式的範例程式	以 HTML 格式提供在 DB2 支援的所有平台上，程式設計語言的範例程式。範例程式僅供參考。並非所有程式設計語言皆有範例可用。只有在安裝了 DB2 Application Development Client 時，才能使用 HTML 範例。 若需程式的其餘相關資訊，請參照*應用程式開發手冊*。	沒有書號	db2hs
	版本注意事項		
DB2 Connect 版本注意事項	提供 DB2 Connect 書籍中未包含的最新資訊。	請參閱備註 #2。	db2cr
DB2 安裝注意事項	提供 DB2 書籍中未包括的最新安裝特定資訊。	僅附於產品 CD-ROM 中。	
DB2 版本注意事項	提供 DB2 書籍中未包含的所有 DB2 產品及特性的最新資訊。	請參閱備註 #2。	db2ir

註:

1. 檔名中第六個位置上的字元 *x* 表示書籍的語言版本。例如，檔名 `db2d0e70` 會識別英文版的 *Administration Guide*，而檔名 `db2d0f70` 則識別同一本書的法文版。下列字母會用在檔名的第六個位置上，以表示語言版本：

語言	識別字
巴西葡萄牙文	b
保加利亞文	u
捷克文	x
丹麥文	d
荷蘭文	q
英文	e
芬蘭文	y
法文	f
德文	g
希臘文	a
匈牙利文	h
義大利文	i
日文	j
韓文	k
挪威文	n

波蘭文	p
葡萄牙文	v
俄文	r
簡體中文	c
斯洛維尼亞文	l
西班牙文	z
瑞典文	s
繁體中文	t
土耳其文	m

2. 「版本注意事項」中可取得 DB2 書籍中未包含的最新資訊 (有兩種檔案格式，HTML 及 ASCII)。而 HTML 版本則可以從「資訊中心」及產品 CD-ROM 中取得。欲檢視 ASCII 檔：
   - 在 UNIX 平台上，請參閱 Release.Notes 檔案。此檔案是位在 DB2DIR/Readme/*%L* 目錄中，其中 *%L* 代表語言環境名稱，而 DB2DIR 代表：
     - /usr/lpp/db2_07_01 (在 AIX 上)
     - /opt/IBMdb2/V7.1 (在 HP-UX、PTX、Solaris、及 Silicon Graphics IRIX 上)
     - /usr/IBMdb2/V7.1 (在 Linux 上)。
   - 在其它平台上，請參閱 RELEASE.TXT 檔案。這個檔案位在產品安裝的目錄中。在 OS/2 平台上，您可以按兩下 **IBM DB2** 資料夾，然後按兩下**版本注意事項**圖示。

列印 PDF 書籍

如果您想擁有印妥的書籍副本，您可以列印 DB2 出版品 CD-ROM 上的 PDF 檔。利用 Adobe Acrobat Reader，您可以列印整本書或只列印特定範圍的頁數。若需檔案庫中各書籍的檔名，請參閱第378頁的表18。

您可以從 Adobe 網站 (http://www.adobe.com) 取得最新版本的 Adobe Acrobat Reader。

DB2 出版品 CD-ROM 中已包含 PDF 檔案，其副檔名爲 PDF。欲存取 PDF 檔：

1. 插入 DB2 出版品 CD-ROM。在 UNIX 系列平台上，裝載 DB2 出版品 CD-ROM。請參照您的*快速入門*一書，取得裝載程序。
2. 啓動 Acrobat Reader。
3. 從下列其中一個位置開啓想要的 PDF 檔：
   - 在 OS/2 及 Windows 平台上：

     x:\doc*language* 目錄，其中 *x* 代表 CD-ROM 光碟機，且 *language* 表示兩個字元的國碼，代表您所使用的語言 (例如，EN 代表英文)。

- 在 UNIX 平台上：

  CD-ROM 中的 */cdrom*/doc/*%L* 目錄，其中 */cdrom* 代表 CD-ROM 的裝載點，且 *%L* 代表想要的語言環境名稱。

您也可以從 CD-ROM 中將 PDF 檔複製到本端或本端磁碟機中，並從該處讀取檔案。

訂購印刷書籍

您可以利用書號銷售單 (SBOF) 各別或整組 (僅限北美洲) 訂購印刷 DB2 書籍。欲訂購書籍，請聯絡您的 IBM 授權經銷商或業務代表，如您在美國，請撥 1-800-879-2755，如在加拿大，請撥 1-800-IBM-4YOU。您也可以從出版品網頁 (http://www.elink.ibmlink.ibm.com/pbl/pbl) 訂購書籍。

有兩組書籍可供訂購。SBOF-8935 提供 DB2 Warehouse Manager 的參照及使用資訊。SBOF-8931 提供所有其它 DB2 Universal Database 產品及特性的參照及使用資訊。每一張 SBOF 的內容均列示在下列表格中：

表 19. 訂購印刷書籍

SBOF 編號	訂購書籍	
SBOF-8931	• Administration Guide: Planning • Administration Guide: Implementation • Administration Guide: Performance • Administrative API Reference • 應用程式開發指南 • Application Development Guide • CLI Guide and Reference • Command Reference • Data Movement Utilities Guide and Reference • 資料倉儲中心管理手冊 • Data Warehouse Center Application Integration Guide • DB2 Connect User's Guide • 安裝與架構補充資料 • Image, Audio, 與 Video Extenders 管理與規畫手冊 • 訊息參考手冊，第一冊與第二冊	• OLAP Integration Server Administration Guide • OLAP Integration Server Metaoutline User's Guide • OLAP Integration Server Model User's Guide • OLAP Integration Server User's Guide • OLAP 安裝與使用手冊 • OLAP Spreadsheet Add-in for Excel 使用手冊 • OLAP Spreadsheet Add-in for Lotus 1-2-3 使用手冊 • Replication 指南與參考手冊 • Spatial Extender Administration and Programming Guide • SQL 入門 • SQL Reference, Volumes 1 and 2 • System Monitor Guide and Reference • Text Extender 管理與程式設計 • Troubleshooting Guide • 新特性介紹
SBOF-8935	• 資訊型錄管理程式管理手冊 • 資訊型錄管理程式使用手冊 • Information Catalog Manager Programming Guide and Reference	• Query Patroller Administration Guide • Query Patroller User's Guide

DB2 線上文件

存取線上說明

所有 DB2 元件都有線上說明。下列表格將描述各種類型的說明。

說明類型	內容	如何存取...
指令說明	解釋命令行處理器中指令的語法。	從交談模式的命令行處理器中，輸入： `? command` 其中 *command* 代表某一關鍵字或整個指令。 例如， `? catalog` 將顯示所有 CATALOG 指令的說明，至於 `? catalog database` 則會顯示 CATALOG DATABASE 指令的說明
從屬站架構輔助程式說明 命令中心說明 控制中心說明 資料倉儲中心說明 事件分析程式說明 資訊型錄管理程式說明 衛星管理中心說明 Script 中心說明	說明您可以在視窗或筆記本中執行的作業。說明包含您必須知道的概觀及先決條件資訊，並說明如何使用視窗或筆記本控制項。	在視窗或筆記本中按一下**說明**按鈕，或按 **F1** 鍵。

訊息說明	說明訊息的原因，及所有您應採取的動作。	從交談模式的命令行處理器中，輸入：
		`? XXXnnnnn`
		其中 *XXXnnnnn* 代表有效的訊息 ID。
		例如， `? SQL30081` 將顯示關於 SQL30081 訊息的說明
		欲一次一個螢幕，檢視訊息說明，請輸入：
		`? XXXnnnnn | more`
		欲將訊息說明儲存在檔案中，請輸入：
		`? XXXnnnnn > filename.ext`
		其中 *filename.ext* 代表您要儲存訊息說明的檔案。

說明類型	內容	如何存取...
SQL 說明	解釋 SQL 陳述式的語法。	從交談模式的命令行處理器中，輸入： `help` *`statement`* 其中 *statement* 代表 SQL 陳述式。 例如，help SELECT 會顯示有關 SELECT 陳述式的說明。 **註:** UNIX 型的平台上沒有 SQL 說明。
SQL 陳述式說明	解釋 SQL 陳述式及類別碼。	從交談模式的命令行處理器中，輸入： `?` *`sqlstate`* 或 `?` *`class code`* 其中 *sqlstate* 代表有效的五位數 SQL 狀態，且 *class code* 代表 SQL 狀態的前兩位數。 例如， `? 08003` 將顯示 08003 SQL 陳述式的說明，至於 `? 08` 則將顯示 08 類別碼的說明

檢視線上資訊

隨本產品所附的書籍軟本均爲超文字標記語言 (HTML) 格式。軟本格式可讓您搜尋或瀏覽資訊，並提供相關資訊的超文字鏈結。它同時也使得您的整個環境中，更易於共用此檔案庫。

您可以用符合 HTML 3.2 版規格的任何瀏覽器，來檢視線上書籍或範例程式。

欲檢視線上手冊或範例程式：

- 如果您正在執行 DB2 管理工具，請使用「資訊中心」。
- 在瀏覽器上按一下**檔案 —>開啓網頁**。您所開啓的網頁含有 DB2 資訊的說明及鏈結：
  - 在 UNIX 平台上，開啓下列網頁：

    `INSTHOME``/sqllib/doc/%L/html/index.htm`

    其中 *%L* 代表語言環境名稱。
  - 在其它平台上，開啓下列網頁：

    `sqllib\doc\html\index.htm`

    路徑位在 DB2 安裝所在的磁碟機上。

如果您尚未安裝「資訊中心」，您可以按兩下**DB2 資訊**圖示來開啓網頁。視您將使用的系統而定，圖示將位於主要產品資料夾或「Windows 啓動」功能表中。

安裝 Netscape 瀏覽器

如果您尚未安裝 Web 瀏覽器，您可以從產品包裝盒中的 Netscape CD-ROM 中安裝 Netscape。若須有關如何安裝的詳細指示，請執行下列：

1. 插入 Netscape CD-ROM。
2. 僅限於 UNIX 系列平台上，裝載 CD-ROM。請參照您的*快速入門*一書，取得裝載程序。
3. 若須安裝指示，請參照 CDNAV*nn*.txt 檔，其中 *nn* 代表您的兩個字元的語言識別字。檔案是位在 CD-ROM 的根目錄中。

用資訊中心來存取資訊

「資訊中心」可讓您迅速地存取 DB2 產品資訊。具有 DB2 管理工具的所有平台皆有「資訊中心」。

您可以按兩下「資訊中心」圖示，開啓「資訊中心」。視您使用的系統而定，圖示會位在主產品資料夾的「資訊」資料夾或 Windows **開始**功能表中。

您也可以在 DB2 Windows 平台上使用工具列及**說明**功能表來存取「資訊中心」。

「資訊中心」提供六種類型的資訊。按一下適當的標籤，可以查閱所提供的該類型的主題。

作業 列出您可以 DB2 執行的作業。

參照 DB2 參考資料、如關鍵字、指令及 API。

書籍 DB2 書籍。

疑難排解 錯誤訊息的種類及其回復動作。

範例程式 隨 DB2 Application Development Client 所附的範例程式。如果您未安裝 DB2 Application Development Client，則不會顯示此標籤。

Web 全球資訊網 (WWW) 上的 DB2 資訊。欲存取這個資訊，您必須從您的系統中與 Web 連接。

當您在其中一個列示中選取某個項目時，「資訊中心」即會啓動檢視器，來顯示資訊。檢視器可以是系統說明檢視器、編輯器或 Web 瀏覽器，視您選取的資訊種類而定。

「資訊中心」提供尋找特性，所以您可以搜尋特定主題而無需瀏覽列示。

若需全文搜尋，則請遵循「資訊中心」中的超文字鏈結，進入**搜尋 DB2 線上資訊**搜尋套表。

通常 HTML 搜尋伺服器會自動啓動。如果 HTML 資訊的搜尋無法運作，您可能必須使用下列方法之一，啓動搜尋伺服器：

在 Windows 中

按一下**開始**，並選取**程式集 —> IBM DB2 —> 資訊 —> 啓動 HTML 搜尋伺服器**。

在 OS/2 中

按兩下 **DB2 for OS/2** 資料夾，然後按兩下**啓動 HTML 搜尋伺服器**圖示。

如果您在搜尋 HTML 資訊時遭遇任何其它問題，請參考版本注意事項。

註: 在 Linux、PTX 及 Silicon Graphics IRIX 環境中，無法使用「搜尋」功能。

使用 DB2 精靈

精靈在每一項作業中，可以逐步協助您完成特定的管理作業。您可以經由控制中心及從屬站架構輔助程式來使用精靈。下列會列出精靈並說明其目的。

註: 「建立資料庫」、「建立索引」、「架構多位置更新」及「效能架構」精靈均可在已分割的資料庫環境中使用。

精靈	協助您...	如何存取...
新增資料庫	在從屬工作站上將資料庫編目。	從「從屬站架構輔助程式」中，按一下**新增**。
備份資料庫	備份計畫的決定、建立及排程。	在「控制中心」中，對著您要備份的資料庫按一下滑鼠右鍵，然後選取**備份 —> 資料庫 - 使用精靈**。
架構多位置更新	架構多位置更新、分散式異動或兩階段確定。	在「控制中心」中，在**資料庫**資料夾上按一下滑鼠右鍵，然後選取**多位置更新**。
建立資料庫	建立一個資料庫，並執行某些基本架構作業。	在「控制中心」中，在**資料庫**資料夾上按一下滑鼠右鍵，然後選取**建立 —> 資料庫 - 使用精靈**。
建立表格	選取基本資料類型，及建立表格的主要鍵。	在「控制中心」中，在**表格**圖示上按一下滑鼠右鍵，然後選取**建立 —> 表格 - 使用精靈**。

精靈	協助您...	如何存取...
建立表格空間	建立新的表格空間。	在「控制中心」中，在**表格空間**圖示上按一下滑鼠右鍵，然後選取**建立 —> 表格空間 - 使用精靈**。
建立索引	針對您所有的查詢來建議您要建立及捨棄哪些索引。	在「控制中心」中，在**索引**圖示上按一下滑鼠右鍵，然後選取**建立 —> 索引 - 使用精靈**。
效能架構	藉由更新架構參數調整資料庫效能，以符合您業務上的需求。	在「控制中心」中，對著您要調整的資料庫按一下滑鼠右鍵，然後選取**使用精靈架構效能**。 至於在分割的資料庫環境中，則在「資料庫分割區」畫面中，對著您要調整的第一個資料庫分割區按一下滑鼠右鍵，然後選取**使用精靈架構效能**。
復置資料庫	錯誤發生後回復資料庫。它會幫助您了解使用哪一個備份及要回轉哪些日誌。	在「控制中心」中，對著您要復置的資料庫按一下滑鼠右鍵，然後選取**復置 —> 資料庫 - 使用精靈**。

設定文件伺服器

根據預設值，DB2 資訊會安裝到您的本端系統上。這表示每一位必須存取 DB2 資訊的人，必須安裝相同的檔案。欲將 DB2 資訊儲存在單一位置中，請執行下列步驟：

1. 在您的本端系統中，從 \sqllib\doc\html 複製所有的檔案及次目錄到 Web 伺服器。每一本書都有它自己的次目錄，其中包含了所有構成該書的必要 HTML 及 GIF 檔。請確定目錄結構沒有改變。
2. 架構 Web 伺服器，在新的位置搜尋檔案。若需相關資訊，請參照*安裝與架構補充資料* 中的「NetQuestion 附錄」。
3. 如果使用 Java 版本的「資訊中心」，您可以對所有的 HTML 檔指定一個基礎 URL。您應該使用該 URL 取得書籍列示。
4. 當您可以檢視書籍檔案時，您可以在經常查閱的主題上加上書籤。您也許會想要將下列網頁加上書籤：
   - 書籍列示
   - 經常使用之書籍的目錄
   - 經常參考的文章，如「變更表」主題
   - 「搜尋」表格

若需如何從中央電腦上使用 DB2 Universal Database 線上文件檔的相關資訊，請參照*安裝與架構補充資料*中的「NetQuestion 附錄」。

搜尋線上資訊

欲在 HTML 檔中尋找資訊，請使用下列方法之一：

- 按一下頂端訊框中的**搜尋**。使用搜尋表格頁面，來尋找特定主題。在 Linux、PTX 或 Silicon Graphics IRIX 環境中無法使用此功能。
- 按一下頂端訊框中的**索引**。使用索引，來找出書籍中的特定主題。
- 顯示目錄或說明或 HTML 書籍的索引，然後使用 Web 瀏覽器的尋找功能，找尋書中的特定主題。
- 使用 Web 瀏覽器的書籤功能，來迅速地回到特定主題。
- 使用「資訊中心」的搜尋功能，來找出特定主題。詳細資訊，請參閱第390頁的『用資訊中心來存取資訊』。

附錄E. 注意事項

而在其它國家中，IBM 不見得有提供本書中所提的各項產品、服務或功能。要知道在您所在之區是否可用到這些產品與服務時，請向當地的 IBM 服務代表查詢。本書在提及 IBM 產品、程式或服務時，不表示或暗示只能使用 IBM 的產品、程式或服務。只要未侵犯 IBM 的智慧財產權，任何功能相當的產品、程式或服務都可以取代 IBM 的產品、程式或服務。不過，其它非 IBM 產品、程式、或服務在運作上的評價與驗證，其責任屬於使用者。

在這本書或文件中可能包含著 IBM 所擁有之專利或專利申請案。本書使用者並不享有前述專利之任何授權。 您可以用書面方式來查詢授權，來函請寄到：

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

若要查詢有關二位元組 (DBCS) 資訊的特許權限事宜，請聯絡您國家的 IBM 智慧財產部門，或者用書面方式寄到：

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

下列段落若與該國之法律條款抵觸，即視為不適用： IBM 僅以現狀提供本書，而不提供任何明示或默示之保證 (包括但不限於可售性或符合特定效用的保證)。若有些地區在某些交易上並不允許排除上述保證，則該排除無效。

本書中可能會有技術上或排版印刷上的訛誤。因此，IBM 會定期修訂；並將修訂後的內容納入新版中。同時，IBM 得隨時改進並 (或) 變動本書中所提及的產品及 (或) 程式。

本書對於非 IBM 網站的援引只是爲了方便而提供，並不對這些網站作任何認可。該些網站上的內容並非本 IBM 產品內容的一部份，用戶使用該網站時應自行承擔風險。

當您提供資訊給 IBM 時，您即授權予 IBM 以其認爲適當的方式來使用或分送資訊，而不必對您負起任何責任。

本程式之獲授權者若希望取得相關資料，以便使用下列資訊者可洽詢 IBM。其下列資訊指的是： (1) 獨立建立的程式與其它程式 (包括此程式) 之間更換資訊的方式 (2) 相互使用已交換之資訊方法。若有任何問題請聯絡：

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

上述資料的取得有其特殊要件，在某些情況下必須付費方得使用。

本書所描述的任何程式及其所有可用的授權著作是由 IBM 所提供，並受到「IBM 客戶合約」、「國際程式授權合約」或雙方之間任何同等合約條款之規範。

此間所含之任何效能資料，皆是得自控制的環境之下；因此不同作業環境之下所得的結果，可能會有很大的差異。部份測量可能是在開發中的系統上執行，因此不保證可以從一般的系統獲致相同的結果。甚至有部份的測量，是利用插補法而得的估計值，其實際結果可能會有所不同。本書的使用者應根據其特有的環境，驗證出適用的資料。

本書所提及之非 IBM 產品資訊，係一由產品的供應商，或其出版的聲明或其它公開管道取得。IBM 並未測試過這些產品，也無法確認這些非 IBM 產品的執行效能、相容性、或任何對產品的其它主張是否完全無誤。如果您對非 IBM 產品的性能有任何的疑問，請逕向該產品的供應商查詢。

有關 IBM 未來動向的任何陳述，僅代表 IBM 的目標而已，並可能於未事先聲明的情況下有所變動或撤回。

本書中含有日常商業活動所用的資料及報告範例。爲了提供完整的說明，這些範例包括個人、公司、廠牌和產品的名稱。這些名稱全屬虛構，若與任何公司的名稱和住址雷同，純屬巧合。

著作權授權：

本書包含原始語言的範例應用程式，用以說明各種作業平台上的程式設計技術。您可以基於研發、使用、銷售或散佈符合作業平台 (用於執行所撰寫的範例程式) 之應用程式設計介面的應用程式等目的，以任何形式複製、修改及散佈這些範例程式，而無需付費給 IBM。但這些範例皆未經過完整的測試。因此，IBM 不會保證或暗示這些程式的穩定性、服務能力或功能。

這些範例程式或是任何衍生著作的每一份拷貝或任何部份，都必須具有下列的著作權聲明：

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

商標

下列術語 (以星號 (*) 標示) 是 IBM 公司在美國、其它國家或兩者的商標。

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

下列術語是其它公司的商標或註冊商標：

Microsoft、Windows、Windows NT 是微軟公司的商標或註冊商標。

Java 以及所有與 Java 有關的商標與標章，以及 Solaris 是 Sun Microsystems, Inc. 在美國、其它國家或兩者的商標。

Tivoli 與 NetView 是 Tivoli Systems Inc. 在美國、其它國家或兩者的商標。

UNIX 是 X/Open Company Limited 在美國、其它國家或兩者的註冊商標，須經該公司授權始可使用。

其它公司、產品或服務名稱 (以兩顆星號 (*) 標示) 可能是其它公司的商標或服務標誌。

索引

索引順序以中文字，英文字，及特殊符號之次序排列。

〔三劃〕

〔四劃〕

〔五劃〕

〔六劃〕

〔七劃〕

〔八劃〕

〔十七劃〕

A

B

C

D

E

F

H

J

L

M

N

O

P

R

S

U

W

洽詢 IBM

當您有技術上的問題時，請在洽詢「DB2 客戶支援中心」之前，仔細閱讀並執行*疑難排解指南*所建議的動作。該指南會告訴您必須預先準備的資訊，協助「DB2 客戶支援中心」提供更完善的服務。

若要取得 DB2 Universal Database 產品的相關資訊，或是訂購該系列產品，請洽詢當地 IBM 分公司的業務代表，或是 IBM 授權的軟體經銷商。

如果您住在美國當地，請撥下列一組電話號碼：

- 1-800-237-5511，客戶支援中心
- 1-888-426-4343，取得可用服務選項的資訊

產品資訊

如果您住在美國當地，請撥下列一組電話號碼：

- 1-800-IBM-CALL (1-800-426-2255) 或 1-800-3IBM-OS2 (1-800-342-6672)，訂購產品或取得一般資訊。
- 1-800-879-2755，訂購出版品。

http://www.ibm.com/software/data/

DB2 World Wide Web 頁面將提供關於新聞、產品說明、教育課程以及其他種種的現行 DB2 資訊。

http://www.ibm.com/software/data/db2/library/

DB2 Product and Service Technical Library 可讓您存取常見的問題、修正程式、書籍，以及最新的 DB2 技術資訊。

註: 這項資訊可能只會以英文表示。

http://www.elink.ibmlink.ibm.com/pbl/pbl/

International Publications 訂購網站會提供書籍的訂購資訊。

http://www.ibm.com/education/certify/

IBM 網站中的 Professional Certification Program 會提供包括 DB2 在內之各種 IBM 產品的認證測試資訊。

ftp.software.ibm.com

以匿名方式登入。您可以在目錄 `/ps/products/db2` 中找到 DB2 及其它產品的相關示範程式、修訂程式、資訊及工具。

comp.databases.ibm-db2, bit.listserv.db2-l

使用者可以利用這些 Internet 新聞群組討論 DB2 產品的使用經驗。

在 Compuserve 上：GO IBMDB2

輸入此項指令，即可存 IBM DB2 Family 論壇。所有 DB2 產品均可透過這些論壇取得支援。

關於如何聯絡美國以外的 IBM 的資訊，請參閱 IBM Software Support Handbook 的附錄 A。若要存取本文件，請造訪下列網頁：http://www.ibm.com/support/，然後選取接近網頁底端的 IBM Software Support Handbook 鏈結。

註: 在某些國家中，IBM 授權的代理商應該洽詢它們的產品支援體系，而不是洽詢「IBM 支援中心」。

IBM® DB2® Universal Database

SC40-0493-01

應用程式開發手冊

版本 7

折疊線

台北市 110 基隆路一段 206 號

大中華研發中心
臺灣國際商業機器股份有限公司
軟體國際部
啟

寄件人 姓名：
地址：

廣告回信
台灣北區郵政管理局登記
北台字第0587號
（免貼郵票）

寄

折疊線

讀者意見表

爲使本書盡善盡美，本公司極需您寶貴的意見；懇請您使用過後，撥冗填寫下表，惠予指教。

請於下表適當空格內，填入記號（∨）；我們會在下一版中，作適當修訂，謝謝您的合作!

評估項目	評　估　意　見		備　註
正確性	內容說明與實際程序是否符合	□是□否	
	參考書目是否正確	□是□否	
一致性	文句用語及風格，前後是否一致	□是□否	
	實際畫面訊息與本書所提之畫面訊息是否一致	□是□否	
完整性	是否遺漏您想知道的項目	□是□否	
	字句、章節是否有遺漏	□是□否	
術語使用	術語之使用是否恰當	□是□否	
	術語之使用，前後是否一致	□是□否	
可讀性	文句用語是否通順	□是□否	
	有否不知所云之處	□是□否	
內容說明	內容說明是否詳盡	□是□否	
	例題說明是否詳盡	□是□否	
排版方式	本書的形狀大小，版面安排是否方便使用	□是□否	
	字體大小，顏色編排，是否有助於閱讀	□是□否	
目錄索引	目錄內容之編排，是否便於查考	□是□否	
	索引語錄之排定，是否便於查考	□是□否	
	※評估意見爲"否"者，請於備註欄說明。		

其他：（篇幅不夠時，請另紙說明。）

上述改正意見，一經採用，本公司有合法之使用及發佈權利，特此聲明。

SC40-0493-01