Linux 的 Database-SQL-RDBMS HOWTO 文件 (PostgreSQL 物件關聯資料庫系統)

作者：Al Dev (Alavoor Vasudevan) *alavoor@yahoo.com*
譯者：曾達康 *a9504480@graduate.hku.hk*

二零零一年一月二日，41.0 版

本文件是一份介紹如何在 Unix 系統快捷地建立一個 SQL 資料庫引擎 (Database Engine) 及前端工具的「實用指南」。它也討論國際標準的 ANSI/ISO SQL 語言及探討在互聯網中「開放式開發」環境 ("open development" environment) 開發 SQL 資料庫引擎的優點。它是關於如何在 Unix 系統中建立 PostgreSQL 這個既可用作應用程式資料庫伺服器 (Application Database Server)，又可用作萬維網資料庫伺服器 (Web Database Server) 的新一代物件關聯式資料庫。PostgreSQL 嘗試實踐今後的國際 ISO / ANSI SQL 標準。本文件也提供關於資料庫的介面程式，如前端圖像介面、應用程式高速開發 (Rapid Application Development) 工具、ODBC、JDBC 驅動程式、C、C++、Java、Perl 程式語言介面和網上資料庫工具。這裡提供的資料適用於所有 Unix / Windows NT 平台及其他資料庫，對資料庫、SQL 語言及 PostgreSQL 的新手非常有用。這文件也有一個 SQL 的導引，解釋 SQL 的語法，對初學者非常有用。有經驗的用家會覺得這是一份有用的參考資料。對學生來說，這處提供的資料會讓他們得到 PostgresSQL 關聯式資料庫的源程式碼，從而學習如何編寫一個物件關聯式 SQL 資料庫。

目錄

1 引言

本文件的主要目標是提供一個關於快捷地設定 PostgreSQL 的 URL 的詳盡名單，和提倡開放原始碼系統，如 PostgreSQL、Linux 的好處。

世界上的每一部電腦都需要一個資料庫來存取資料。你使用電腦主要理由是用來儲存、 讀取和處理資料，而且它們要很快完成工作，從而節省你的時間。同時，系統必須簡單、 完整、快速、可靠、經濟和易用。資料庫是最重要的系統， 因為它儲存了世界上每一間公司的關鍵資料。世上的每一項工業都需要一個資料庫系統。 沒有的話，電訊、汽車、銀行、航空公司等工業都不能有效運作。 最流行的資料庫系統都跟從國際標準組織 (International Standard Organisation, ISO) 的規格和 ANSI SQL（美國）標準。業界常用的最新規格是 ISO/ANSI SQL 1992。下一個標準是 SQL 1998/99，又名 SQL-3，這仍在開發中。流行的資料庫，如 Oracle，Sybase 和 Informix 都是根據這些標準，或嘗試實施這些標準。

沒有如 ANSI/ISO SQL 的標準，用戶要一次建立一套適用於所有資料庫系統的應用軟件會非常困難。客戶希望一次過使用 ISO SQL，ODBC，JDBC 發展一個應用軟件而在世上所有的資料庫系統應用。

世上自由而支援 ISO SQL、ANSI SQL/98、SQL/92 和 ANSI SQL/89 的關聯式資料庫 (RDBMS) 中，最流行的是 PostgreSQL。PostgreSQL 是新一代的物件關聯式資料庫， 以全面符合 ISO/ANSI 等 SQL 標準為目標。免費的 RDBMS 中，PostgreSQL 是唯一同時支援物件資料庫和 SQL 的。本文件會告訴你如何安裝這資料庫、 如何建立網上資料庫、應用軟件資料庫、前端圖像介面和介面程式。 我强烈建議你所寫的資料庫軟件必須百分百符合 ISO/ANSI SQL、ODBC、JDBC 等標準，這樣，使得你的程式能移植到不同的資料庫，如 PostgreSQL、Oracle、 Sybase、Informix 等。

你可從 PostgreSQL 得到最高質素及非常豐富的功能，因為它依從「開放原始碼開發模式」 (Open Source Code development model)。開放原始碼模式會給你完整的源程式碼， 開發過程在互聯網上由有極大量的人手進行。 目前的趨勢顯示未來大部分的軟件開發工作會在世界通行的所謂「資訊高速公路

」(Information Super-Highway) 上進行。未來數年，互聯網會爆炸性地成長，這樣會促成業界更多使用 PostgreSQL。

把統計學、數學和科學的原理應用於軟件質素，我們只會在一個如 PostgreSQL 般，把原程式碼開放給經資訊高速公路連繫在一起的大量頭腦的系統下，才能得到最高質素的軟件。正所謂集思廣益。開放原始碼的模式也可防止重覆工作，非常經濟，節省發佈時間及合符有關優化國家和地球資源的現代經濟定律。只要有人寫好了一個軟件，你便不需要再來一次。你不會把寶貴的時間浪費於早已做好的工作。一寸光陰一寸金，你要好好利用，因為人一天只有八小時工作！在我們踏入廿一世紀時，我們取得所需軟件的方法將會改變。每人的第一選擇都將會是如 PostgreSQL、Linux 等開放原始碼系統。

購買可執行的軟件不代表你擁有軟件。原程式碼是珍貴的資產，二元檔並無價值。購買軟件可能會成為歷史。你只需要買好的硬件，花錢在硬件上而從互聯網上取得軟件是值得的。重點是工作量大的電腦硬件。硬件才是真正跑動的馬匹，軟件只是鞭策牠。電腦硬件的複雜程度使得全球只有 6 個國家表現出設計及生產電腦晶片和硬件的能力。這是一種先進的科技，過程非常複雜，資金密集，設廠及生產 0.18（甚至小於 0.18）微米科技的機器投資龐大。在一小片晶片上便有數以百萬計的半導體和電路擠在一起。Applied Material、AMD、英特爾、Cyrix、日立、IBM 及其他工司花費大量人-年 (man-years) 在高科技，如晶片設計、微電子 (Micro-electronics) 及納電子 (Nano-electronics) 的研究上。微米代表百萬分之一米 (10^-6)，納米代表十億分之一米 (10^-9)。現今微電子技術中，0.35 微米的使用鋁作導體，0.25 微米的使用銅。在不久的將來，用銅的 0.10 微米技術，甚致納電子技術將會應用於電腦晶片。因為銅導電能力較强，鋁導體將會被淘汰。在刻蝕晶片 (photolithography) 的過程中，極高頻的紫外線、X-射線或電子束會被用於刻蝕 (etch) 特徵大小 (feature size) 小於 0.15 微米的電路。今後二十年，矽片將會被分子電腦 (molecular computers) 和生物晶片 (bio chips) 取代。它們的速度將會比矽片快數十億倍！分子是一組原子。而微小的原子組成你在世上所見的所有物件。分子電腦會用物質的分子來當超快的電子開關制。開制 (ON) 代表 1、關閉 (OFF) 代表 0。世上所有電腦程式都是根據二元數（數字 1 和 0）運作。

下表列出晶片技術的進步及未來發展趨勢。

未來晶片能力的進展

****

項目/年份	1997	1999	2001	2003	2012	2020	2030
特徵大小（微米）	0.25	0.18	0.15	0.13	0.05	< 0.00001	原子
晶片大小（毫米）	200	300	300	300	450	生物/分子	量子
最低運行電壓	1.8-2.5	1.5-1.8	1.2-1.5	1.2-1.5	0.5-0.6	< 0.001	極小
最大功率消耗	70	90	110	130	175	600	極小
運行頻率 (MHz)	750	1,250	1,500	2,100	10,000	> 50,000	-----
DRAM 容量	256 MB	1 GB	2 GB	4 GB	256 GB	> 1000GB	-----

如你所見，硬件才重要和需要高科技，軟件則需要人手，但所用的科技就比較簡單。

另一方面，世上每國家都會發展軟件。事實上， 世上任何一個擁有一部低價電腦的人都可編寫軟件。

Oracle，Informix，Sybase，IBM DB2 (Unix) 等資料庫，都是用 "C" 語言寫成， 二元檔則是用編譯器產生，這樣便可售給用戶了。Oracle、Sybase、Informix 資料庫是百分百的 C 程式！！

十四年來，人們已在 PostgreSQL 上做了很多工作，從頭創造另一個符合 ANSI/ISO SQL 的資料庫系統並不合符常理。在 PostgreSQL 的源程式上加上欠缺的功能或將之加强， 之後立刻使用，有很大好處。

據估計，對「互聯網產品」的需求將會以指數級數 (exponentially) 增長，因為它可維繫一群高質素、低價而龐大數量的使用者及開發人員。 不使用「互聯網產品」的國家會錯過這場「全球互聯網革命」而被其他國家大大拋離。 這是因為互聯網本身就是世上最大的「軟件公司」，也是大型的軟件「發電機」！

1.1 量子 (Quantum) 電腦——量子物理學很有用

如你在前表「未來晶片能力的進展」所見，在大約 2030 年，PostgreSQL 一類資料庫系統會在量子電腦運行。量子電腦根據原子粒子的特性，如旋轉 (spin) 方向來產生狀態。例如，在向上旋轉 (spin is up) 時，一夥粒子可被看作“一”，向下旋轉時可被看作“零”。原子和其核心可出現重疊 (superposition) 的狀態，使得一、零和其間的數值可同時表達。籍著干擾原子的旋轉， 量子位 (qubit) 可放在一起，令它們可當一個整體來運作， 使得非線性計算 (nonlinear computational) 能力遙遙領先現代超級電腦！！ 在原子層面，量子物理學有助理解原子粒子的表現。

2 物理定律適用於電腦軟件！

這章會說明科學在不同事物，如軟件、宇宙、原子、能量甚至你自己的創造過程中扮演如何重要的角色。也會說明為何科學知識在使用科學的產品前非常重要。

黃金定律是—「你不可用一件來歷不明的產品！！」這定律適用於所有東西— 資料庫系統、電腦系統、作業系統、宇宙甚至你的身體！即是說， 你要得到系統完整的源程式碼和資料。了解人體和人體內的原子的運作非常重要，因為 PostgreSQL、微軟視窗 95 等都是人造的。

創造是很重要的一步。使用科學物件的人必須知道它如何產生。這甚至適用於電腦系統和 PostgreSQL。大多數人都沒有科學知識，因此不知道如微軟視窗 NT/95、Oracle、 人體和宇宙等系統如何產生。很多人不知道什麼創造宇宙和微軟視窗 NT/95 和它們內部是什麼。複雜的系統以簡單的組成部分建成——有千千萬萬個宇宙產生了， 每個宇宙都是千千萬萬個超星團 (super-cluster) 組成，每個超星團都是千千萬萬個星系 (galaxy) 組成，每個星系都是千千萬萬夥恒星 (star) 組成，有些恒星系統中有行星 (planet)，而行星是數以十億計的原子組成。（世界歷史中， 只在古印度有一個人創造了一個宇宙，但在現代社會再沒有發生。 世上不少國家都正嘗試創造宇宙。）創造宇宙是一件極為先進的科技，比炸中廣島和長崎，造成恐怖的破壞的原子彈先進得多。 現代核武既細小又强勁，只需一個這些核彈投入太平洋即可使地球消失！ 武器的變化無窮！！在古印度的戰場中就用過核武和其他更強力非凡的武器！ 當艾伯特・愛因斯坦 (Albert Eienstein)（一個二十世紀 00 年代的科學家）說核武可使大城市人間蒸發時，沒有人相信他。 今時今日也沒有人相信人力可創造宇宙。

微軟視窗九五之類軟件是以 C 和組合語言寫成，只用到 1 和 0。我們所住的這類宇宙是由其他空間中兩粒不相似而適合的原子組合撞擊而成。（有趣的事在粒子撞擊之前剛好發生了） 人體是由兩粒不相似而合適的細胞組合撞擊而成！！ （有趣的事在細胞撞擊之前剛好發生了）人類遺傳了宇宙的特性。你所在的宇宙以前並不存在——宇宙中的原子並不存在，甚至時間都不存在！ 宇宙在大爆炸時出生，開始擴張並持續成長。直至現在宇宙仍在擴張！！ 有人在一個名為‘**Brahma**’的宇宙創造了你所在的宇宙。 知識是這個宇宙之母！！ ‘Brahma’在你生活的嬰兒宇宙誕生之前已經愛上了‘知識之母

’！！ 那過程和你出生的經過差不多！沒有來自知識之母的的‘基因’，要寫一個小型的 ’C’ 程式也沒可能！（譯按：完全不明白這一段想説什麼，更不知怎樣譯才符合原意， 大家如果知道，請來函通知。）

有朝一日我們的宇宙會關上（一聲巨響之後），其中的所有原子會完全毀滅和消失！

人類實在是有可能創造宇宙的。可以創造的宇宙總數是無限大， 可以創造的作業系統的總數也是無限大！！有數以百萬計的宇宙， 可分為三大類。無限數量的宇宙和無限變化的多維原子 (multi-dimensional atoms) 崩塌成為數個基本維宇宙 *(primary-dimensional-universe)*。 非常先進的數學也支持這理論。

科學和統計學定律支持開放原始碼，如 PostgreSQL 和 Linux。互聯網的速度與日俱增，也越來越可靠，開放原始碼的動量 (momentum) 會迅速增加。另外，如果統計學及科學定律正確的話， 人們更注意科學和當無知的人開始學習科學，封閉式程式遲早會在世上消失。

發展 PostgreSQL 這類計劃需要能量和時間等資源，因此，PostgreSQL 是能量和時間的產品。能量和時間只能在科學上解釋，所以物理學和 PostgreSQL、Linux 之類軟件專案是有直接關聯的。科學（物理）定律適用於任何時間空間，任何行為， 甚至軟件開發。

物理學即使在你説話（聲波）、步行（地面和腳之間的摩擦力）、 看書和編寫軟件時都發生作用。世上所有科學都植根於數學，包括 PostgreSQL。PostgreSQL 使用了數學的一個小分枝‘現代代數’(Modern Algebra)。現代代數處理‘集合論’(Set Theory)、‘關係代數’(Relational Algebra)、群組 (Groups)、環 (Rings)、收集 (Collections)、集 (Sets)、聯合 (Unions)、交錯 (Intersections)、排斥 (Exclusions)、域 (Domains)、列 (Lists) 等等。

PostgreSQL 這類產品之所以存在，便是因為能量和時間。 而質量和能量是同一事物！質能相同的事實在一百年前仍未為人所知！ 即使在今時今日世人也不知道互聯網是最大的軟件「發電機」 和世上最大的「軟件公司」。

腦細胞在工作（編寫程式）時會消耗能量，把來自食物的化學能轉化為電能及熱能。 即使在你閱讀這段文字時，你的腦細胞便會消耗燃料及使用小量能量。也就是説， 人腦是一台熱力學上的機器 (thermodynamic heat engine)。正因為這樣， 熱力學的定律也適用於人腦，也就間接地影響到 PostgreSQL 一類軟件。

世上可有無限種顏色、電腦語言、晶片設計和理論，但不可能有一種完美的顏色、 電腦語言、設計和系統！你所能得到的只是近乎完美的顏色（波長）、系統、 資料庫或理論！大自然就像一個萬花筒一有無限維數的空間， 其他空間粒子的變限變化，但它們全部結合為小數的空間，反之亦然。

把世上數百萬網民的能量結合起來，使得建立一個近乎完美的系統 （包括資料庫軟件）變得有可能。孤掌難鳴，但籍著網絡把大量人手集合起來， 他們的總能量會非常龐大，而且可集中於建立一個近乎完美的系統。

能量的單位是焦耳 (Joules)，千焦耳或公斤，時間以秒或小時量度。功率 (power) 是能量除以時間，單位是瓦或千瓦。

```
         每人的能量 = y 焦耳
或以質量來説
         每人的能量 = y 公克
質量和能量之間的轉換因數是 E = m * c * c’’c’ 是光速而 ’m’ 是質量。
         時間 = 8 小時（這是一個常數，因為一人一天只有 8 小時）
         功率 = 能量 / 時間
              = (y / (8 * 60 * 60)) 瓦
         世界的總功率 = n * (y / (8 * 60 * 60)) 瓦
n = 參與計劃的總人數。
```

從以上方程式，很明顯增加 'n' 會大幅改善產品質素。n 越大功率也越大（用千瓦表示）。 你可以想像全互聯網有多少能量（以千焦耳計算） 和功率（以千瓦計算）專注於 Linux 和 PostgreSQL 一類系統！

很明顯，互聯網可連繫很多人，也就是說互聯網有大量能量和時間， 可用比軟件公司更短的時間產生更高質素的軟件。即使是微軟、IBM 等大公司也不可違抗物理定律，而必會向物理定律投降。

結論是：因為科學的定律，‘開於原始碼’系統，像 PostgreSQL、Linux 會戰勝‘封閉原始碼’系統，和必會比它們優異。這是有科學根據的。 人類不應浪費時間創造太多重覆的軟件產品。

3 PostgreSQL 是什麼？

PostgreSQL 是一個自由的資料庫，給予你完整的程式碼，是一個以 ANSI SQL 1998, 92, 89 規格為目標的物件關聯式資料庫系統，可在不同的硬件平台和操作系統中運行。 PostgreSQL 的最終目的是百分百符合 ANSI/ISO SQL 及成為世界第一的開放通用資料庫。

今天，PostgreSQL 是世上最先進的系統。令人詫異的是，很多商業的資料庫在質素、 特性和能力各方面都追不上它！！PostgreSQL 是地球上很多國家的合作成果，是一個和國際太空站 (International Space Station) 相似的專案。在未來的數十年，PostgreSQL 仍會是最佳的資料庫系統，因為它是一個開放原始碼系統。

PostgreSQL 背後的基本想法是一只要一個模組的程式寫好了， 你就不應浪費時間重新寫一次，哪管只花千分之一秒！！

Informix Universal server（1997 年推出）的基礎是 PostgreSQL 的較舊版本，因為 Informix 收購了 Illustra Inc. 及整合到 Informix。Illustra 資料庫是根據 Postgres（PostgreSQL 的前身）寫成的。

PostgreSQL 是 POSTGRES 資料庫管理系統的改良版，一個新一代 DBMS 的研究原型 (prototype)。在保留 POSTGRES 強大的資料模型及豐富的資料型態的同時， 它以一個擴充了的 SQL 子集來取代 PostQuel 查詢語言。

PostgreSQL 是由一群參加 PostgreSQL 開發通信論壇的互聯網開發人員進行的。 目前的協調者為 Marc G. Fournier。

- *scrappy@postgreSQL.org*

這隊伍會負責今後所有 PostgreSQL 發展工作。當然，資料庫的用戶自己也是 PostgreSQL 的開發人員！這開發過程是分散在互聯網上大量資料庫用家的。

PostgreSQL 1.01 的作者是 Andrew Yu 和 Jolly Chen。PostgreSQL 的前身 Postgres 的程式碼，是很多大學畢業生、本科生及程式設計員在 University of California，Berkeley 的 Michael Stonebraker 教授的指導下的產品。

已有數以百萬套 PostgreSQL 被安裝作資料庫伺服器， 網上資料庫伺服器和應用程式資料庫伺服器。它是一個很先進的物件關聯式資料庫 （ORDBMS）。

PostgreSQL 可在 Soloris、SunOS、HPUX、AIX、Linux、Irix、Digital Unix、BSDi、 NetBSD、FreeBSD、SCO Unix、NEXTSTEP、Unixware 及所有其他的 Unix 執行。移植到視窗 95/NT 的工作已以 Cygnus cygwin32 套件完成。

本文所述的 PostgreSQL 和有關物件受 University of California, Berkeley 的版權保護。

3.1 白皮書

PostgreSQL 資料簡介：

- 題目： PostgreSQL SQL RDBMS 資料庫（物件關聯資料庫管理系統）
- 目前版本：7.0.1
- 年齡： PostgreSQL 十五歲。自 1985 年開始開發
- 作者： 十五年來互聯網上數以百萬計的大學和公司

PostgreSQL 的白皮書在 `<http://www.greatbridge.com>` 找到。

4 哪個較好？PostgreSQL 還是 MySQL？

4.1 PostgreSQL 打敗 Oracle、IBM DB2、MS SQL server 和其他！！

PostgreSQL 在表現、速度、擴展性 (scalability) 和 可信性 (reliability) 各方面的標準測試中打敗 Oracle 8（和 8i）、IBM DB2、MS SQL server、Sybase、 Interbase 和 MySQL！

到 `<http://www.aldev.8m.com>` 或 `<http://aldev.webjump.com>` 閱讀基準測試結果。

（譯按：有關該測試結果，有不少人提出質疑，請參閱 `<http://www.devshed.com/BrainDump/MySQL_Benchmarks/>`）

4.2 MySQL 和其他重覆的 RDBMSes

MySQL 是另一個開放原始碼的 SQL 伺服器，但它不支援交易 (Transaction)。它適用於非常小的資料庫，而不支援先進的 SQL 功能。PostgreSQL 則是一個企業級的資料庫，支援交易和幾乎所有 SQL 結構。PostgreSQL 比商業的資料庫，如 Oracle、Sybase 和 Informix 等先進得多。PostgreSQL 支援非常先進的鎖定機制 (locking mechanism) 和很多在商業資料庫中找不到的先進功能！！

在不久將來，MySQL 的開發將會停止，因為 MySQL 也以 ANSI SQL 為目標，是一個重覆的產品。我們要最先進和成熟的開放原始碼 SQL 伺服器，而拋掉所有其他的，因為我們沒有很多時間（來對付多個 RDBMS）！事實上，你連只對付一個如 PostgreSQL 般强勁的 SQL 伺服器的時間也沒有。所有 MySQL 用者都會轉移到 PostgreSQL。而且，MySQL 是一個‘類商業’(quasi-commercial) 的產品，和 PostgreSQL 不同，後者開放原始碼，無需授權費。既生瑜，何生亮。有了 PostgreSQL，就不需再有另一個 SQL 資料庫系統了。

如 MySQL 一類的重覆產品使得用家無所適從，分散資源。 一個“近乎完美”的系統應該是獨一無二的系統， 世上每一個人都應為它工作！！重覆的產品弊多於利，因此，絕不值得分散資源。 這已在商業的資料庫發生，如 Oracle、Sybase、Informix 和微軟 SQL server。這樣會使得用者分裂，而且產品互不兼容。我卻想把 ***SQL*** 伺服器的原始碼給你控制！！！

你不需要上百個資料庫系統，你只需要一個最好的資料庫伺服器，它就是‘PostgreSQL’。

警告：根據一份如 ANSI SQL 的規格，人們可造出無限個資料庫系統！！

MySQL 沒有而 PostgreSQL 支援的功能如下—

- 交易 (Transactions)
- 內儲程序 (Stored Procedures)
- 激發 (Triggers)（update，insert 和 delete）
- 物件導向資料庫 (Object oriented databases)
- 先進的封鎖系統 (Locking systems)，在多用戶、多交易環境下的並行管理 (concurrency management)。
- 子查詢 (Sub-queries)
- 伺服器端游標 (Server-side cursors)
- 查詢快取 (Query caching)
- 資料庫封鎖 (Locking of databases)
- 較好的連結支援（JOIN、UNION、MINUS、INTERSECT、outer join）
- 很多其他功能——多不勝數。

MySQL 可在 `<http://www.tcx.se>` 找到。
（譯按：MySQL 已改以 GPL 發行。）

5 何處取得 ?

你可從以下地方購買 Redhat Linux 光碟、Debian Linux 光碟或 Slackware Linux 光碟，從中找到 PostgreSQL 套件（原始碼和二元碼）：

- Linux System Labs 網址： `<http://www.lsl.com/>`（7 美元）
- CheapBytes Inc 網址： `<http://www.cheapbytes.com/>`（7 美元）
- Debian 主網址： `<http://www.debian.org/vendors.html>`

PostgreSQL 組織也售賣‘PostgreSQL CDROM’，內含完整的原始碼，許多 Unix 操作系統的可執行檔及完整的文件。

- 從主網頁取得 PostgreSQL 光碟： `<http://www.postgresql.org>` 30（美元）

PostgreSQL 只含二元檔的發行版本 (distribution)：

- PostgreSQL RPMs 的維護人是 Lamer Owen，在 *lamar.owen@wgcr.org* 找到。
- PostgreSQL 源 RPM 和二元 RPM `<http://www.ramifordistat.net/postgres>`。
- PostgreSQL 源 RPM 和二元 RPM `<http://www.postgresql.org>`。點選“Latest News”和 Redhat RPMs。
- PostgreSQL 源 RPM 和二元 RPM `<http://www.redhat.com/pub/contrib/i386/>`。ftp 站台在 `<ftp://ftp.redhat.com/pub/contrib/i386/>`

- Solaris, HPUX, AIX, IRIX, Linux 的二元檔站台： `<ftp://ftp.postgresql.org/pub/bindist>`

萬維網網站：

- 主網站： `<http://www.postgresql.org/>`
- 次網站： `<http://logical.thought.net/postgres95/>`
- `<http://www.itm.tu-clausthal.de/mirrors/postgres95/>`
- `<http://s2k-ftp.cs.berkeley.edu:8000/postgres95/>`
- `<http://xenium.pdi.net/PostgreSQL/>`
- `<http://s2k-ftp.cs.berkeley.edu:8000/postgres95/>`

以下為 ftp 站：

- 主 FTP: `<ftp://ftp.postgresql.org/pub>`
- 次 FTP: `<ftp://ftp.chicks.net/pub/postgresql>`
- `<ftp://ftp.emsi.priv.at/pub/postgres/>`
- `<ftp://ftp.itm.tu-clausthal.de/pub/mirrors/postgres95>`
- `<ftp://rocker.sch.bme.hu/pub/mirrors/postgreSQL>`
- `<ftp://ftp.jaist.ac.jp/pub/dbms/postgres95>`
- `<ftp://ftp.luga.or.at/pub/postgres95>`
- `<ftp://postgres95.vnet.net:/pub/postgres95>`
- `<ftp://ftpza.co.za/mirrors/postgres>`
- `<ftp://sunsite.auc.dk/pub/databases/postgresql>`
- `<ftp://ftp.task.gda.pl/pub/software/postgresql>`
- `<ftp://xenium.pdi.net/pub/PostgreSQL>`

PostgreSQL 的源程式碼也可從 sunsite unc 的映射站得到（全球有大約一千個）。它是在 Redhat Linux 發行版本 (distribution) 的 /pub/contrib/i386/postgresql.rpm 檔中。

- 要知道有什麼映射站，請到 `<ftp://sunsite.unc.edu>`

6 PostgreSQL 快速安裝指引

這節會幫你在小於五分鐘內快速安裝和啓動這資料庫。

6.1 安裝和測試

安裝、測試、確認和執行 PostgreSQL 的步驟 以 root 身份簽入

```
# cd /mnt/cdrom/RedHat/RPMS
# man rpm
# ls postgre*.rpm
# rpm -qpl postgre*.rpm | less （觀看檔案名單）
# rpm -qpi postgre*.rpm （觀看套件資料）
# cat /etc/passwd | grep postgres
```

注意：如果你看到一個‘postgres’使用者，你可能要備份和清除 postgres 的家目錄 postgres 和刪除這名用者，或把他改名為‘postgres2’之類。安裝時需要乾乾淨淨。

```
:
# rpm -i postgre*.rpm （必須安裝所有套件，即客戶端、開發、資料和主套件來使
pgaccess 正常工作）
# man chkconfig
# chkconfig --add postgresql  （用來在開機時開始 pg。）
# /etc/rc.d/init.d/postgresql start （啓動 postgresql）
# man xhost
# xhost +  （讓 pgaccess 使用顯示裝置）
# su - postgres
bash$ man createdb
bash$ createdb mydatabase
bash$ man psql
bash$ psql mydatabase
……在 psql 中可按上 / 下箭咀來編輯歷史記錄或 \s

bash$ export DISPLAY=<主機名>:0.0
bash$ man pgaccess
bash$ pgaccess mydatabase
```

現在你可以在 pgaccess 或 psql 高速下達各種 SQL 指令了！！

```
bash$ cd /usr/doc/postgresql*
```

在此閱讀所有 FAQ，使用者、程式設計員、管理手冊和指引。

6.2 PostgreSQL RPMs

此外，請看 `<http://www.ramifordistat.net/postgres>` 的 *"Installation Steps"*。

PostgreSQL RPM 由 Lamar Owen 維護，位於 *lamar.owen@wgcr.org* 在 `<http://www.postgresql.org>` 可找到更多 PostgreSQL 的資料。

6.3 Maximum RPM

熟習 RedHat RPM 套件管理員以管理 PostgreSQL 安裝。在 `<http://www.RPM.org>` 下載‘Maximum RPM’一書，找尋檔名 maximum-rpm.ps.gz。 在 Linux 使用 gv 指令閱讀它—

```
# gv maximum-rpm.ps.gz
```

有一個 rpm2deb 可把 RPM 套件轉換為 Debian Linux 套件

6.4 例子 RPM

測試 PostgreSQL 的各種介面需要例子。從以下地方安裝 PostgreSQL 例子目錄—

- Linux 光碟 - postgresql-*examples.rpm
- postgresql-*examples.rpm `<http://www.aldev.8m.com>` 或 `<http://aldev.webjump.com>`
- PostgreSQL 原程式碼 postgresql*.src.rpm，在 examples、testing 或 tutorial 目錄尋找。

6.5 測試 PyGreSQL—Python 介面

安裝 examples 套件，請參閱 6.4 (例子 RPM)，之後—

```
bash$ cd /usr/lib/pgsql/python
bash$ createdb thilo
bash$ psql thilo
thilo=> create table test (aa char(30), bb char(30) );
thilo=> \q
bash$ /usr/bin/python
>>> import _pg
>>> db = _pg.connect('thilo', 'localhost')
>>> db.query("INSERT INTO test VALUES ('ping', 'pong')")
>>> db.query("SELECT * FROM test")
eins|zwei
----+----
ping|pong
(1 row)
>>>CTRL+D
bash$
……好像行了—現在正式安裝它
bash$ su - root
# cp /usr/lib/pgsql/python/_pg.so /usr/lib/python1.5/lib-dynload
```

6.6 測試 Perl—Perl 介面

安裝 examples 套件，請參閱 6.4 (例子 RPM)，之後—

```
root# chown -R postgres.postgres /var/lib/pgsql/examples
bash$ cd /var/lib/pgsql/examples/perl5
bash$ perl ./example.pl
```

注意：如果以上指令失敗了請做如此做：全域變數 @INC 要在包括 site_perl 目錄中的 Pg.pm 模組，所以要如下使用 -I 選項。

```
bash$ perl -I/usr/lib/perl5/site_perl/5.004/i386-linux-thread ./example.pl
```

……你的 perl 已在存取 PostgreSQL 資料庫！！

要使用 perl 介面，請參閱 example.pl 檔。

6.7 測試 libpq, libpq++ 介面

安裝 examples 套件，請參閱 6.4 (例子 RPM)，之後—

```
root# chown -R postgres.postgres /var/lib/pgsql/examples
bash$ cd /var/lib/pgsql/examples/libpq
bash$ gcc testlibpq.c -I/usr/include/pgsql -lpq
bash$ export PATH=$PATH:.
bash$ a.out

bash$ cd /var/lib/pgsql/examples/libpq++
bash$ g++ testlibpq0.cc -I/usr/include/pgsql -I/usr/include/pgsql/libpq++
-lpq++ -lpq -lcrypt
bash$ ./a.out （注意：不須理會錯誤訊息—如下）
> create table foo (aa int, bb char(4));
No tuples returned...
status = 1
Error returned: fe_setauthsvc: invalid name: , ignoring...
> insert into foo values ('4535', 'vasu');
No tuples returned...
status = 1
Error returned: fe_setauthsvc: invalid name: , ignoring...
> select * from foo;
aa   |bb   |
-----|-----|
4535 |vasu |
Query returned 1 row.
>
>CTRL+D
bash$
```

……你直接用 C/C++ 介面來存取 PostgreSQL 資料庫！！

6.8 測試 Java 介面

安裝 examples 套件，請參閱 6.4 (例子 RPM) 和安裝以下東西。

- 從 `<ftp://ftp.redhat.com/pub/contrib/i386>` 或 `<http://www.blackdown.org>` 取得 JDK jdk-*glibc*.rpm
- 從 `<ftp://ftp.redhat.com/pub/contrib/i386>` 取得 postgresql-jdbc-*.rpm

```
root# chown -R postgres.postgres /var/lib/pgsql/examples
bash$ cd /var/lib/pgsql/examples/jdbc
bash$ echo $CLASSPATH
 --> 應顯示 CLASSPATH=/usr/lib/pgsql/jdbc7.0-1.2.jar:.:/home/java/jdk1.2.2/lib:/usr/lib/pgsql:/usr/lib/p

而 jdbc*.jar 的版本號碼要正確。
*.jar 檔要放在 /usr/lib/pgsql 和  /usr/libjdk*/lib 目錄中。

bash$ export CLASSPATH=/usr/lib/pgsql/jdbc7.0-1.2.jar:.:/home/java/jdk1.2.2/lib:/usr/lib/pgsql:/usr/lib

編輯所有 psql.java 檔，把‘package’的行變為評論。
bash$ javac psql.java
bash$ java psql jdbc:postgresql:template1 postgres < password >[1] select * from pg_tables;
tablename       tableowner      hasindexes      hasrules
pg_type postgres        true    false   false
pg_attribute    postgres        true    false   false
[2]
CTRL+C
bash$
```

……你直接用 Java 介面來存取 PostgreSQL 資料庫！

6.9 測試 ecpg 介面

安裝 examples 套件，請參閱 6.4 (例子 RPM)，之後—

```
root# chown -R postgres.postgres /var/lib/pgsql/examples
bash$ cd /var/lib/pgsql/examples/ecpg
bash$ ecpg test1.pgc -I/usr/include/pgsql
bash$ cc test1.c -I/usr/include/pgsql -lecpg -lpq -lcrypt
bash$ createdb mm
bash$ ./a.out
```

……你直接用嵌入式 SQL 來存取 PostgreSQL 資料庫！

6.10 測試 SQL 的例子－自訂資料型態和函數

安裝 examples 套件，請參閱 6.4 (例子 RPM)，之後－

```
root# chown -R postgres.postgres /var/lib/pgsql/examples
bash$ cd /var/lib/pgsql/examples/sql
未寫好…
```

6.11 測試 Tcl/TK 介面

Tck/Tk 的例子是 pgaccess 程式。 使用一個文字編輯器閱讀 /usr/bin/pgaccess 檔案－

```
bash$ view /usr/bin/pgaccess
bash$ export DISPLAY=<你電腦的主機名>:0.0
bash$ createdb mydb
bash$ pgaccess mydb
```

6.12 測試 ODBC 介面

1. 在 `<http://www.insightdist.com/psqlodbc/>` 取得 win32 pgsql odbc 驅動程式
2. 另外請參閱 /usr/lib/libpsqlodbc.a

6.13 測試 MPSQL Motif-worksheet 介面

在 `<http://www.mutinybaysoftware.com>` 取得 RPM。

6.14 確認

要確認 PostgreSQL 的高品質，執行 Regression 測試套件： 以 root 身份簽入

```
# rpm -i postgresql*test.rpm
閱讀 README 檔或安裝源程式，它已有 regress 目錄
# rpm -i postgresql*.src.rpm
# cd /usr/src/redhat/SPECS
# more postgresql*.spec （以察看需要安裝哪些 RPM 套件）
# rpm -bp postgresql*.spec （…這樣會準備該套件）

Regression 測試需要 Makefiles 和類似 *fmgr*.h 的標頭檔，可用以下指令
產生－
# rpm --short-circuit -bc postgresql*.spec （走捷徑！）
看到‘make -C common  SUBSYS.o’後按 CTRL-C 終止工作。這時候 configure
已經成功，所有 Makefiles 和標頭檔也已產生。你不再需要讓它繼續下去。
# cd /usr/src/redhat/BUILD
```

```
# chown -R postgres postgresql*
# su - postgres
bash$ cd /usr/src/redhat/BUILD/postgresql-6.5.3/src/test/regress
bash$ more README
bash$ make clean; make all runtest
bash$ more regress.out
```

6.15 緊急除錯

有時除錯嵌補會在主要的推出 PostgreSQL 的主要版本之後出現。你可依據自己需要，自行進行修補。請跟從以下步驟： 到 postgresql 程式碼的目錄

```
# rpm -i postgresql*.src.rpm
# cd /usr/src/postgresql6.5.3
# man patch
# patch -p0 < patchfile
# make clean
# make
```

嵌補檔是在

- PostgreSQL 嵌補： <ftp://ftp.postgresql.org/pub/patches>

7 快速開始指引

請參考6 (快速安裝指引)一章。

7.1 資料庫建立、丟棄、改名

你可使用使用者友善的 GUI，名為‘pgaccess’來建立和丟棄資料庫，也可用指令列‘psql’工具。

```
如你以 root 身份登入，變身為用者‘postgres’：
# xhost + （讓 pgaccess 使用顯示裝置）
# su - postgres
bash$ man createdb
bash$ createdb mydatabase
bash$ man psql
bash$ psql mydatabase
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s

bash$ export DISPLAY=<主機名>:0.0
bash$ man pgaccess
bash$ pgaccess mydatabase
```

你現在可以在 psql 或 pgaccess 高速下達 SQL 指令。

要丟棄資料庫的話：

```
bash$ man dropdb
bash$ man destroydb  （在較舊版本的 pgsql 中使用）
bash$ dropdb <資料庫名稱>
```

也可以在一個 SQL 連接期間用以下指令消滅一個資料庫：

```
> drop database <資料庫名稱>
```

要更改資料庫名稱，請參閱7.7 (備份和還原)一節

7.2 建立和丟棄用戶

要建立新用戶，以 unix 用戶‘postres’簽入，你可用簡單易用的 GUI 工具‘pgaccess’來增減用戶。

```
bash$ man pgaccess
bash$ pgaccess <資料庫名稱>
```

擊選“Users”tab，再擊選 Object|New 或 Object|Delete

你也可用指令行命令稿 (script)。名為‘createuser’的命令稿啓動 psql。

```
bash$ man createuser
bash$ createuser <用戶名稱>
bash$ createuser -h host -p port -i userid <用戶名稱>
```

要丟棄一個 postgres 用戶，使用命令稿‘destroyuser’。

```
bash$ man dropuser
bash$ man destroyuser （在較舊版本的 pgsql 中使用）
bash$ destroyuser
```

7.3 建立和丟棄群組

目前沒有簡單的介面可用來設定用戶群組。你要自行在 **pg_group** 表格中插入或更新紀錄。如：

```
bash$ su - postgres
bash$ psql <資料庫名稱>
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s

psql=> insert into pg_group (groname, grosysid, grolist)
psql=> values ('posthackers', '1234', '{5443, 8261}' );
INSERT 58224
psql=> grant insert on foo to group posthackers;
CHANGE
psql=>
```

pg_group 中的欄位為： **groname** 群組名稱。這名稱必須全為字母或數字，不可有底線或標點符號。

grosysid 群組代碼。這是一個 int4，每個群組都要獨一無二。

grolist 屬於此群組的 **pg_user** 代碼名單。是一個 int4[]。

要刪除群組：

```
bash$ su - postgres
bash$ psql <資料庫名稱>
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s


psql=> delete from pg_group where groname = 'posthackers';
```

7.4 建立、編輯和丟棄表格

你可使用使用者友善的 GUI，名為‘pgaccess’，或指令列‘psql’工具來建立、編輯和丟棄資料庫表格。

```
bash$ man pgaccess
bash$ pgaccess <資料庫名稱>
```

擊選 Table | New | Design 按鈕。

```
bash$ man psql
bash$ psql <資料庫名稱>
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s
```

在 psql 提示下，鍵入標準的 SQL 指令，如‘create table’、‘alter table’或‘drop table’來處理表格。

7.5 建立、編輯和丟棄表格中的紀錄

你可使用使用者友善的 GUI，名為‘pgaccess’，或指令列‘psql’工具來建立、編輯和丟棄資料庫表格中的紀錄。

```
bash$ man pgaccess
bash$ pgaccess <資料庫名稱>
```

擊選 Table | < 選一個表格 > | Open 按鈕。

```
bash$ man psql
bash$ psql <資料庫名稱>
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s
```

在 psql 提示下，鍵入標準的 SQL 指令，如‘insert into table_name’、‘update table_name’或‘delete from table_name’來處理表格。

7.6 改變目前的資料庫

你可使用使用者友善的 GUI，名為‘pgaccess’，或指令列‘psql’工具來改變目前的資料庫。

```
bash$ man pgaccess
bash$ pgaccess <資料庫名稱>
```

擊選 Database | Open 按鈕。

```
bash$ man psql
bash$ psql <資料庫名稱>
…在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s

psql=> connect <資料庫名稱> <user>
```

7.7 備份與還原資料庫

PostgreSQL 提供了兩個工具來備份你的系統：**pg_dump** 備份一個資料庫，**pg_dumpall** 一次過備份所有資料庫。

```
bash$ su - postgres
bash$ man pd_dump
bash$ pd_dump <資料庫名稱> > database_name.pgdump
```

還原則：

```
bash$ cat database_name.pgdump | psql <資料庫名稱>
```

這技考可用於搬移資料庫或為資料庫改名。

警告：每個資料庫都要定時備份。因為 PostgreSQL 自行管理它在檔案系統中的檔案，所以不要倚賴系統備份來作資料庫備份。 沒有人能保證那些檔案在還原後會處於一個可用而一致的狀態。

備份大型資料庫：因為 PostreSQL 容許表格大過系統的最大檔案容量，把一個表格輸出為一個檔案會有問題， 因為所產生的檔案可能會超出系統的限制。因為 **pg_dump** 會寫到 **stdout**，你可使用標準的 unix 工具來解決這可能的問題：

使用壓縮了的輸出：

```
bash$ pg_dump <資料庫名稱> | gzip > filename.dump.gz
```

用以下方法重新載入：

```
bash$ createdb <資料庫名稱>
bash$ gunzip -c filename.dump.gz | psql <資料庫名稱>
```

或

```
bash$ cat filename.dump.gz | gunzip | psql <資料庫名稱>
```

使用 split：

```
bash$ pg_dump <資料庫名稱> | split -b 1m - filename.dump.
```

注意：以上指令中，在 filename.dump 後有一點 (.)！！你可用以下方法重新載入：

```
bash$ man createdb
bash$ createdb <資料庫名稱>
bash$ cat filename.dump.* | pgsql <資料庫名稱>
```

當然，檔案名稱（filename）和 **pg_dump** 輸出的內容無需和資料庫名稱配合。此外，還原了的資料庫可有任何新名。 因此這方法適用於為資料庫改名。

要把所有 PostgreSQL 輸出，使用 **pg_dumpall**

```
bash$ man pg_dumpall
bash$ pg_dumpall -o > db.out
要重新載入：
bash$ psql -e template1 < db.out
```

7.8 資料庫保安

請參閱10 (PostgreSQL 保安)一節。

7.9 線上求助

你必須知道 PostgreSQL 的線上求助設施，因為它會為你節省很多時間，讓你快速取得資料。

參閱不同指令，如 createdb、createuser 等等的線上 man pages。

```
bash$ man createdb
```

也請參閱 psql 的線上求助訊息，在 psql 提示鍵入 \h

```
bash$ psql mydatabase
psql> \h
提示：在 psql 按上 / 下箭咀來編輯歷史紀錄或用 \s
```

7.10 建立激發 (Triggers) 和內儲程序 (Stored Procedures)

要建立激發或內儲程序，先在你使用的資料庫執行‘createlang’命令稿來安裝‘plpgsql’。如果你想預設使用它，把它安裝在‘template1’，因為以後建立的資料庫都是 template1 的仿裝品。參閱‘createlang’網頁或位於 /usr/doc/postgresql-7.0.2/user/index.html 的 User Guide。

```
bash$ man createlang
bash$ createdb mydb
bash$ export PGLIB=/usr/lib/pgsql
bash$ createlang plpgsql mydb
bash$ createlang plpgsql template1
```

也請參考 6.4 (例子 RPM) 中激發和內儲程序的例子。Examples RPM 中的其中一個例子：

```
create function tg_pfield_au() returns opaque as '
begin
    if new.name != old.name then
        update PSlot set pfname = new.name where pfname = old.name;
    end if;
    return new;
end;
' language 'plpgsql';

create trigger tg_pfield_au after update
    on PField for each row execute procedure tg_pfield_au();
```

另一個激發程式碼例子：

```
create trigger check_fkeys_pkey_exist
        before insert or update on fkeys
        for each row
        execute procedure
        check_primary_key ('fkey1', 'fkey2', 'pkeys', 'pkey1', 'pkey2');
```

你必須安裝 TEST 套件— postgresql-test-7.0.2-2.rpm 和閱讀在 /usr/lib/pgsql/test/regress/sql 中的 SQL 命令稿例子。

要顯示資料庫中激發的名單—

```
bash$ psql mydb
psql=> \?
psql=> \dS
psql=> \d pg_trigger
psql=> select tgname from pg_trigger order by tgname;
```

要顯示資料庫中函數和內儲程序的名單—

```
bash$ psql mydb
psql=> \?
psql=> \dS
psql=> \d pg_proc
psql=> select proname, prosrc from pg_proc order by proname;
psql=> \df
```

7.11 PostgreSQL 文件

有關其他問題，請參閱 PostgreSQL 的說明書，它們的資料十分全面。PostgreSQL 文件已在套件中。請閱讀‘User's Guide’、‘Programmer's Guide’、‘Administrator's Guide’和其他說明書。

8 PostgreSQL 支援超過 200 Gig 的超級資料庫

早已有很多大公司將 PostgreSQL 用於大型的資料庫。在此提供以下技考：

8.1 中央處理器種類—32 位元或 64 位元

32 位元處理器的電腦在資料庫超過 5 GigaByte 時表現會迅速下降。你可用 32 位元處理器執行 30 gig 的資料庫，但表現會下降。32 位元機器受到最多 2 GB 記憶體、2 GB 檔案系統及操作系統其他方面的限制。使用 SGI、IBM、或 HP 為 Linux 特製的檔案系統或 ext3-fs 來在 32 位元機器上支援大過 2 GB 的檔案大小。

對極大型的資料庫來說，我强烈建議你使用 64 位元處理器，如 Digital Alpha、Sun Ultra-sparc 64 位元處理器、SGI 64 位元處理器、Intel Mercel IA-64 處理器、HPUX 64 位元電腦、IBM 64 位元電腦。使用 64 位元的處理器來編譯 PostgreSQL，它就可以處理大型的資料庫及查詢。查詢大型表格和資料庫的

速度會比 32 位元的電腦快數倍。64 位元電腦的優點是你有很大的記憶位址 (memory addressing space) 和操作系統可處理很大的檔案系統、為大型資料庫提供較佳表現、 支援更多記憶體、能力更强……

（譯按：2.4 的 Linux 核心已突破 2 GB 檔案大小的限制，並支援最多 64 GB 記憶體。）

8.2 多處理器

要使用大型的資料料，你最好使用內有 4、16 或 32 夥處理器的 SMP 電腦。另一方面，你也可使用 4 至 5 部單處理器電腦，而把你的資料庫分割 (partition) 為 4 至 5 個分開的資料庫，每個在一部電腦運行。每個處理器都以高速 （100MBits）的乙以太網絡卡連接。例如，你的資料庫中有二百個表格，你可把它們分為四個資料庫，每個有五十個表格。這樣，你把工作平均分配給 4 部分開的電腦。這是一個比四處理器電腦便宜的方案。 你可使用‘跨資料庫的查詢’、在區域網絡使用 NFS、為唯讀表格‘CREATE VIEW’來完成這工作。每個處理器都可「見到」所有資料庫，即全部 200 個表格。在未來，PostgreSQL 會支援‘跨資料庫的查詢’（已經在 TODO 名單），在將推出的 7.1 版出現。例如，使用代號 a、b 作表格名稱作的跨資料庫查詢會類似——

```
select a.col1, a.col2, b.col4, b.col7
from
    database1.my_tablea a, database2.my_tableb b
where
    a.col1 = b.col3 and
    a.col4 = b.col9;

update my_tablea
set
    col1 =  b.col2
from
    database1.my_tablea a, database2.my_tableb b
where
    a.col4 = b.col9;
```

8.3 複製 (Replication) 伺服器

為大型企業/商業公司而設的複製伺服器由 PostgreSQL Inc 作商業販賣。你可使用複製伺服器來提供重覆的資料及高度可得性 (availability)。複製伺服器是一件複雜、先進的產品，因此需要金錢。

9 憑什麼相信 PostgreSQL？Regression 測試套件為用戶建立信心

有賴「物理學定律」，我們可以科學地檢證 PostgreSQL 是否真的依從 ISO/ANSI SQL 規格工作。為方便測試 PostgreSQL，Regression 測試套件（src/test/regress）已包含在發行版本 (distribution) 中， 它容許你的電腦檢證標準的 SQL 操作及 PostgreSQL 的延伸能力。 測試套件中已包含了數百個 SQL 測試程式。

你應用電腦的高速來驗證 PostgreSQL，而不是用人腦。 電腦可以人腦千萬倍甚至數以億倍的速度來進行 Regression 測試。現代電腦可在很短時間內執行數十億個 SQL 測試。在不久將來，電腦會比人腦快數以兆倍。因此， 使用電腦來測試電腦的表現是合理的。

如有需要，你也可自行增加測試。如果你認為它會對其你互聯網用家有幫助， 你也可把它上載到 PostgreSQL 的主網站。Regression 測試套件幫助用戶建立對 PostgreSQL 的信心及信任，它也幫助生產系統迅速設立 PostgreSQL。

Regression 測試套件可被視為開發人員與最終用戶間相互同意的一份“非常有力”的技術文件。PostgreSQL 在開發過程及推出軟件之前廣泛使用 Regression 測試套件來保證質素。

PostgreSQL 的能力可籍 Regression 測試套件直接反映出來。如果一種功能、 語法或特性在 Regression 測試中存在，它就被支援，所有其他沒有列出的 PostgreSQL 就可能不支援！！你可以自行驗證及加入 Regression 測試套件中。

10 資料庫保安

資料庫保安問題需在多個不同層面探討：

- 資料庫檔案保護。資料庫的所有檔案都不准 *postgres* 超級用戶以外的用戶讀取。
- 據預設值，從用戶連接到資料庫伺服器只能透過本機的 UNIX 插座 (socket)，不是 TCP/IP 插座。伺服器需以 -i 選項啓動才能容許非本機客戶連接。
- 客戶連接可以 IP 地址和 / 或用者名稱加以限制，只需修改 **$PG_DATA** 中的 **pg_hba.conf** 檔。
- 可籍外部套件認證客戶連接。
- 每位 Postgres 用戶都有一個使用者名稱和選擇性地擁有一個密碼。 預設值是用戶沒有權限寫入不是自己建立的資料庫。
- 用戶可被編配到群組中，能否存取一個表格可以群組特權來限制。

10.1 用戶認證

認證是後端伺服器和 postmaster 確保要求存取資料的用戶是否確為其人的過程。所有啓動 Postgres 的用者都與認證是後端伺服器和 postmaster **pg_user** 類別比較，以確定他們有權做想做的工作。 而用者真實身分的辨別工作以多個不同方法完成：

- 在使用者 **shell** 中： 在使用者 shell 直接啓動的後端伺服器記下用者的（有效）使用者代號 (effective user-id)，之後作一次 **setuid** 來化身為 **postgres**。 有效使用者識別號會被用作存取控制的基礎。再沒有其他認證工作。
- 來自網絡： 如果 Postgres 系統以分散式運行，任何人都可存取 postmaster 行程 (process) 的互聯網 TCP 埠。資料庫管理員需設定 **$PGDATA** 目錄中的 **pg_hda.conf** 檔，來指定不同主機連接不同資料庫時所用的認證系統。

  請參閱 **pg_hba.conf(5)** （man 5 pg_hba.conf） 以了解不同認證系統。當然，根據主機來進行認證在 Unix 也不是毫無破碇。有決心的入侵者也可以偽裝自己所用的主機。這些保安問題不在 Postgres 的控制範圍。

10.2 依據主機控制存取

依據主機控制存取即 PostgreSQL 用於決定什麼客戶可存取資料庫和他們如何確認身份的過程。每個資料庫都有一個名為 **pg_hba.conf** 的檔案，在它的 **$PGDATA** 目錄中，它控制誰能存取資料庫。每個存取資料庫的用戶都要在 **pg_hba.conf** 中有相應的紀錄。否則所有來自該客戶的連接都會以**"User authemtication failed"** 錯誤訊息被拒。

請參閱 **pg_hba.conf(5)** 的 man page（man 5 pg_hba.conf）。

pg_hba.conf 檔的基本格式是一組紀錄，一行一個。空行或以 hash 字母（#）開始的行則不理會。一個紀錄由一些用空格和 / 或 Tab 分開的欄位組成。

從客戶來的連接可經 Unix 插座或互聯網插座（即 TCP/IP）。來自 Unix 插座的連接用以下格式的記錄控制：

```
local database authentication method
```

而

database 指定本紀錄用於哪個資料庫。**all** 代表它適用於所有資料庫。

authentication method 指定用戶在使用 Unix 插座時用來認證的方法。不同的方法容後詳述。

來自互聯網插座的連接用以下格式的記錄控制：

```
host database TCP/IP-address TCP/IP-mask authentication method
```

TCP/IP 地址會 邏輯上 *and'ed* 到指定的 TCP/IP 罩和客戶的 TCP/IP 地址。如兩者相等，這紀錄會用於這個連接。如果一個連接符合超過一個紀錄， 檔案中的第一個紀錄會被使用。不論 TCP/IP 地址還是 TCP/IP 罩都是以分點的十進數格式 (dotted decimal notation) 表達。如果一個連接和所有紀錄都不符，reject 認證方法將會被使用。（參閱10.3 (認證方法)）

10.3 認證方法

Unix 和 TCP/IP 插座都支援以下認證方法：

- **trust** 無條件地容許連接。
- **reject** 無條件地拒絕連接。
- **crypt** 問客戶端用者的密碼。它會經加密（使用 crypt(3)）和與 pg_shadow 表格的密碼比較。如果密碼相符，便容許連接。
- **password** 問客戶端用者的密碼。它會原裝傳送和與 pg_shadow 表格的密碼比較。如果密碼相符，便容許連接。一個選擇性的密碼檔可在 password 關鍵字後出現。這樣密碼的配對會用這檔案，而不是 **pg_shadow** 檔案。請參閱 **pg_passwd**。

以下認證方法只受 TCP/IP 插座支援：

- **krb4** 用 Kerberos V4 來認證用戶。
- **krb5** 用 Kerberos V4 來認證用戶。

- **ident** 用客戶端的 ident 伺服器來認證用戶（RFC 1413）。在 **ident** 關鍵字後還可選擇性地指明一個映射，讓 ident 用戶名稱對應到 Postgres 用戶名稱。對應的關係儲存在 **$PGDATA/pg_ident.conf** 檔中。

這裡有一些例子：

```
# 信任所有來自 Unix 插座的連接。
local   trust
# 信任所有這部電腦經 TCP/IP 來的連接。
host    all 127.0.0.1   255.255.255.255     trust
# 我們討厭這部電腦。
host    all 192.168.0.10    255.255.255.0       reject
# 這部電腦不懂加密，所以我們要純文字的密碼。
host    all 192.168.0.3 255.255.255.0       password
# 這群電腦的其他成員都要提供加密了的密碼。
host    all 192.168.0.0 255.255.255.0       crypt
```

10.4 存取控制

Postgres 提供了限制其他用家存取自己資料的機制。

- 資料庫超級用家 **(Database superusers)** 資料庫超級用家（即 **pg_user.usesuper** 設定為對的用家）自動通過以下兩項之外的所有存取控制：**pg_user.usecatupd** 沒有設定為對的話不准手動更新系統目錄，摧毀系統目錄（或修改它的綱目 (schema)）則永不准許。
- 存取特權 **(Access Privilege)** 存取特權是用來限制讀、寫和設定類別 (class) 規則的權力，用 SQL **grant/revoke(1)** 來決定。
- 移除類別和修改綱目 毀滅或修改現存類別的結構，如修改、丟棄表格，和丟棄索引，只能由類別的主人執行。 如上所述，這類動作不許用於系統目錄。

10.5 經 SSH 的保安 TCP/IP 連接

你可用 **ssh** 來加密客戶和 Postgres 伺服器間的網絡連接。做得正確的話，這樣可形成一個保安的網絡連接。

ssh 的文件提供了大部份開始時需要的資料。請參閱 `<http://www.heimhardt.de/htdocs/ssh.html>` 以加深了解。逐步解釋只需兩步。

經 **ssh** 形成一條保安通道： 逐步解釋只需兩步。

- 產生一條到後端電腦的通道，如下：

```
ssh -L 3333:wit.mcs.anl.gov:5432 postgres@wit.mcs.anl.gov
```

- **-L** 參數後的第一個數字，**33333**，是你這方通道的埠號。第二個數字， **5432**，是另一方的終點一後端的埠號。兩個埠數之間的名稱或地址屬於伺服器。 **ssh** 最後一個參數中用家自行選擇是否加上的用戶名稱也是。 沒有用戶名稱的話，**ssh** 會嘗試使用登入客戶端電腦的用戶名稱。 你可用任何伺服器電腦接受的用戶名稱，不一定要用和 PostgreSQL 有關的。

- 你現在已開始了一個 **ssh** 時段，你把可你所用的電腦的 Postgres 客戶連接到你前一步所定的埠號。如你使用 **psql**，你要另一個介面，因為之前所用的一個已被 **ssh** 佔據。

```
psql -h localhost -p 3333 -d mpw
```

- 注意你要用 **-h** 參數使得你的客戶用 TCP 插座而不是 Unix 插座。如果你用 **5432** 作通道的終點你可略去埠的參數。

10.6 Kerberos 認證

Kerberos 是一個業界標準的保安認證系統，適用於經過公用網絡的分散式系統。

如何取得 Kerberos 認證系統並不隨同 Postgres 散佈。不同版本的 Kerberos 多以可供選擇的軟件的形式自作業系統廠商得到。此外，原程式碼的發行版本 (distribution) 可自 MIT Project Athena 得到。

注意：即使你使廠商提供了 `Kerberos`，你仍可能想用 `MIT`
版，因為一些廠商的版本被故意弄壞以致不能和 `MIT` 版本合作。

有關 Kerberos 的查詢可提交你的廠商或 MIT Project Athena。注意 FAQLs（常問問題名單，Frequently-Asked Questions Lists) 會被定時寄至 Kerberos 通信論壇（寄信去訂閱），和 USENET 新聞組。

安裝： 安裝 Kerberos 在 Kerberos Installation Notes 中已詳細說明。要確定伺服器鍵碼檔 (key file)（**srvtab** 或 **keytab**）可被 postgres 用戶讀到。籍設定 **src/Makefile.global** 檔的 KRBVERS 變數，Postgres 或其客戶程式可編譯為使用 MIT Kerberos 通訊協定的第四或五版。你也可改變 Postgres 期望找到相關函式庫、標頭檔和自己的伺服器鍵碼檔的位置。編譯完後， Postgres 要登記為一個 Kerberos 服務。請參閱 Kerberos Operations Notes 和相關 manual pages 以取得有關登記服務的詳情。

操作： 安裝之後，Postgres 在各方面應如一般 Kerberos 服務運作。要知道認證的使用方法，請參閱 *PostgreSQL User's Guide* 有關 **postmaster** 和 **psql** 的章節。

在 Kerberos 第五版掛釣 (hooks)，對用戶和服務命名作了以下假設（也請參閱以下表格）。

- 用戶的主要名稱 (principal names，anames) 第一個部份中假設含有 Unix / Postgres 用戶名稱。
- Postgres 服務假設有兩個部份，服務名稱和主機名稱。後者如同第四版的格式 （即除去所有域名的部分）。

```
                    表格：Kerberos 參數例子
 ------------------------------------------------------
 參數          例子
 ------------------------------------------------------
 user         frew@S2K.ORG
 user         aoki/HOST=miyu.S2K.Berkeley.EDU@S2K.ORG
 host         postgres_dbms/ucbvax@S2K.ORG
 ------------------------------------------------------
```

11 PostgreSQL 的前端 GUI 工具（圖像使用者界面）

網頁瀏覽器在未來將會成為最流行的前端 GUI。建議你把所有「祖傳」的視窗 95/NT 程式改寫為在網上執行的程式。

你要使用網上應用程式伺服器如 17.3 ()（使用 Java）或 17.4 ()（使用 Python）或 17.5 ()。

佳網上命令稿 (script) 編寫（或編譯）語言是 21 (PHP+Zend 編譯器)。**PHP** 功能十分強勁，因為它把 Perl、Java、C++ 和 Javascript 的功能結合到一種語言中，而且可於所有作業系統—Unixes 和視窗 NT/95 執行。

排名以喜歡的程度為先後，最佳工具為

- Enhydra 在 17.3 () 加上 Borland Java JBuilder Linux 版 `<http://www.inprise.com>`
- Zope 在 17.4 ()
- OpenACS 在 17.5 ()
- PHP 命令稿和 Zend 編譯器 21 (PHP+Zend 編譯器)
- X-Designer 支援 C++, Java 和 MFC `<http://www.ist.co.uk/xd>`
- Windows95 和 Unix 版的 QT 在 `<http://www.troll.no>` 和 `<ftp://ftp.troll.no>`
- Code Crusader 在 Linux cdrom 中，根據 MetroWorks Code Warrior 寫成的免費軟件 `<http://www.kaze.stetson.edu/cdevel/code_crusader/about.html>`
- MetroWorks 的 Code Warrior `<http://www.metrowerks.com>`
- GNU Prof C++ IDE，來自 *(Redhat)* `<http://www.redhat.com>` Cygnus `<http://www.cygnus.com>`
- Borland C++ Builder Linux 版 `<http://www.inprise.com>`
- Borland Java JBuilder Linux 版 `<http://www.inprise.com>`

以最佳為先的語言選擇—

1. Java，但它的程式執行得很慢，而且要授權費。C++ 比 Java 快五倍！！
2. Python（強大的物件導向命令稿語言）。
3. PHP 萬維網伺服器命令稿加上 HTML、DHTML、Javascript 客戶端命令稿及 Java-Applet。
4. Perl 命令稿語言使用 Perl-Qt 或 Perl-Tk 13 (Perl 資料庫介面)
5. Omnipresent 和 Omnipotent 語言 C++ (GNU g++)：
    - Fast CGI（以 C++ 寫成）加上 Javascript/Java-Applet 作為網頁前端介面
    - GNU C++ 加 QtEZ 或 QT
    - GNU C++ 加 Lesstif 或 Motif。

還有其他工具—PostgreSQL 發行版本 (distribution) 中有一個名為‘pgTcl’的 Tcl/Tk 介面程式庫。Tcl/Tk 有一個名為 SpecTcl 的綜合開發環境。

- Lesstif Motif 工具 `<ftp://ftp.redhat.com/pub/contrib/i386/lesstif*.rpm>`
- Vibe Java/C++ 在 `<http://www.LinuxMall.com/products/00487.html>`
- JccWarrior `<ftp://ftp.redhat.com/pub/contrib/i386/jcc*.rpm>`
- Tcl/Tk `<http://www.scriptics.com>`
- 名為 INCR 的 Tcl 物件導向延伸 `<http://www.tcltk.com>`
- Visual TCL 網站 `<http://www.neuron.com>`
- Visual TCL Redhat rpm 在 `<ftp://ftp.redhat.com/pub/contrib/i386/visualtcl*.rpm>`
- `<http://sunscript.sun.com/>`
- `<http://sunscript.sun.com/TclTkCore/>`
- `<ftp://ftp.sunlabs.com/pub/tcl/tcl8.0a2.tar.Z>`
- Java FreeBuilder `<ftp://ftp.redhat.com/pub/contrib/i386/free*.rpm>`
- SpecTCL `<ftp://ftp.redhat.com/pub/contrib/i386/spec*.rpm>`
- PostgreSQL 的 Java RAD 工具 Kanchenjunga `<http://www.man.ac.uk/~whaley/kj/kanch.html>`
- Applixware 工具 `<http://www.redhat.com>`（譯按：Redhat 早已停售 Applix 產品。）
- XWPE X Window Programming Environment `<http://www.identicalsoftware.com/xwpe/>` 或 `<ftp://ftp.rpi.edu/~payned/xwpe>` `<ftp://ftp.redhat.com/pub/contrib/i386/xwpe*.rpm>`
- XWB X Window Work Bench `<ftp://ftp.redhat.com/pub/contrib/i386/xwb*.rpm>`
- NEdit `<ftp://ftp.redhat.com/pub/contrib/i386/nedit*.rpm>`

你也可在視窗 95 中使用 Borland C++ Builder、Delphi、Borland JBuilder、PowerBuilder 經 ODBC/JDBC 驅動程式連接到 unix 電腦的 PostgreSQL。

12 PostgreSQL 的介面驅動程式

12.1 PostgreSQL 的 ODBC 驅動程式

ODBC 代表由微軟制定的‘Open DataBase Connectivity’。它是從不同廠商的不同資料庫存取資料的流行介面。使用 ODBC 驅動程式寫成的軟件保證可用於不同資料庫，如 PostgreSQL、Oracle、Sybase 和 Informix 等。

- *PostODBC* `<http://www.insightdist.com/psqlodbc>` 已經包括在發行版本中。請參閱主網頁 `<http://www.postgresql.org>`。它已包括在 PostgreSQL 光碟中。
- 在 `<http://www.iodbc.org>` 的開放原始碼 ODBC 專案。
- `<http://www.openlinksw.com>` Open Link Software Corporation 售賣 PostgreSQL 和其他資料庫的 ODBC 驅動程式。她也送出免費的 ODBC (名額有限）。

- Insight 的 PostgreSQL ODBC `<http://www.insightdist.com/psqlodbc>` 這是 PostODBC 的官方網站。
- FreeODBC 套件 `<http://www.ids.net/~bjepson/freeODBC/>`。這是一個免費的 ODBC。
- PostgreSQL 的 ODBC 32 Explorer `<http://members.nbci.com/anhr>`

12.2 PostgreSQL 的 UDBC 驅動程式

UDBC 是一種獨立於驅動程式管理員 (driver managers) 和 DLL 支援的靜態 ODBC 版本，用來直接把資料庫連接能力嵌入到應用軟件中。

- `<http://www.openlinksw.com>` Open Link Software Corporation 出售 PostgreSQL 和其他資料庫的 UDBC 驅動程式。Open Link 也送出免費的 UDBC (名額有限）。

12.3 PostgreSQL 的 JDBC 驅動程式

JDBC 代表‘Java DataBase Connectivity’。Java 是一種昇陽 (Sun Microsystems) 所開發的獨立於平台的程式語言。Java 程式設計師被鼓勵使用 JDBC 來撰寫資料庫程式，以增强程式在不同資料庫，如 PostgreSQL、Oracle、Informix 等之間的移植能力。如果你寫 Java 程式，你可在以下網站取得 PostgreSQL 的 JDBC 驅動程式。

JDBC 驅動程式已包括在 PostgreSQL 發行版本的 postgresql-jdbc*.rpm 套件中。

- `<http://www.demon.co.uk/finder/postgres/index.html>` 昇陽 Java 的 PostgreSQL 連接
- `<ftp://ftp.ai.mit.edu/people/rst/rst-jdbc.tar.gz>`
- `<http://www.openlinksw.com>` Open Link Software Corporation 出售 PostgreSQL 和其他資料庫的 JDBC 驅動程式。Open Link 也送出免費的 JDBC (名額有限）。
- JDBC 英國網站 `<http://www.retep.org.uk/postgres>`
- JDBC FAQ 網站 `<http://eagle.eku.edu/tools/jdbc/faq.html>`

JDBC 的網頁、導引和 FAQ 在：

- JDBC HOME `<http://splash.javasoft.com/jdbc>`
- JDBC 導引 `<http://www.javasoft.com/products/jdk/1.1/docs/guide/jdbc>`
- JDBC FAQ `<http://javanese.yoyoweb.com/JDBC/FAQ.txt>`

請參考 6.8 (測試 Java 介面) 一節。

12.4 PostgreSQL 的 Java

Java 程式設計師會覺得這些對他們很有用。

- `<ftp://ftp.redhat.com/pub/contrib/i386>` 看 postgresql-jdbc-*.rpm
- `<http://www.blackdown.org>`

請參考 6.8 (測試 Java 介面) 一節。

13 PostgreSQL 的 Perl 資料庫介面 (Database Interface, DBI)

13.1 PostgreSQL 的 PERL 介面

PERL 是‘Practical Exptraction and Report Language’的簡寫。 世上所有軟硬件平台都可使用 Perl。你所在視窗 95/NT、蘋果麥金塔 iMac、所有 Unix （Solaris、HPUX、AIX、Linux、Irix、SCO 等等）、大型電腦 MVS、桌面的 OS/2、OS/400、Amdahl UTS 和其他很多電腦。Perl 甚至可在很多不流行和不為人知的操作系統和硬件執行！！所以，在看到 Perl 在一個罕見的操作系統上執行時，不必大驚小怪。你可以想像到 Perl 的使用者和開發人員的數量。和“C”語言相似，Perl 已有很長壽命，而在未來數千年仍大行其道！Perl 的執行速度比 Java 快十倍，有時甚至比“C”快。Java 是一個非常複雜的系統，既有虛擬機器，又有直譯器，使得它非常緩慢、不穩定和不可靠。 Perl 則簡潔、快速，更是物件導向。

PostgreSQL 的 Perl 介面已包括在 PostgreSQL 的安裝套件中。請參看 src/pgsql_perl5 目錄。

- Pgsql_perl5 電郵聯絡地址： *E.Mergl@bawue.de*
- Perl 網頁 `<http://www.perl.com/perl/index.html>`
- Perl 導引，請在 `<http://reference.perl.com/>` 參看導引目錄
- Perl FAQ 在 `<http://www.yahoo.com/Computers_and_Internet/Programming_Languages/Perl/>`
- 從 `<http://www.perl.com/CPAN/modules/by-module/CPAN>` 取得所有 Perl 模組之母，打下‘/usr/bin/cpan’、‘man CPAN’和數以千計的模組 `<http://www.perl.com/CPAN-local/modules/by-module>`。
- Perl GUI 使用者介面 Perl-Qt rpm： `<ftp://ftp.redhat.com/pub/contrib/i386>` 也請看 PerlQt-1.06-1.i386.rpm
- Perl GUI 使用者介面 Perl-Qt： `<http://www.accessone.com/~jql/perlqt.html>`
- Perl GUI 使用者介面 Perl-XForms： `<ftp://ftp.redhat.com/pub/contrib/i386>`，尋找 Xforms4Perl-0.8.4-1.i386.rpm
- Perl GUI 使用者介面 Perl-Tk： `<ftp://ftp.redhat.com/pub/contrib/i386>`
- Perl GUI kits： `<http://reference.perl.com/query.cgi?ui>`
- Perl 資料庫介面： `<http://reference.perl.com/query.cgi?database>`

- Perl 到 "C" 的翻譯工具： `<http://www.perl.com/CPAN-local/modules/by-module/B/>` 尋找 Compiler-a3.tar.gz
- 把 Perl 編譯為可執行檔。Perl2Exe 是一個可把 perl 命令稿轉為可執行檔的的指令列工具。`<http://www.indigostar.com/perl2exe.htm>`
- Bourne shell 到 Perl 的翻譯工具： `<http://www.perl.com/CPAN/authors/id/MERLYN/sh2perl-0.02.tar.gz>`
- awk 轉為 perl 的 a2p 和 sed 轉為 perl 的 s2p 已包括在 PERL 散佈中。
- 也請看 comp.lang.perl.* 等有關 PERL 的新聞組

13.2 Perl 資料庫介面 DBI

13.2.1 何謂 DBI？

Perl 資料庫介面（DBI）是 Perl 語言的一個資料庫存取應用程式介面（API）。Perl DBI API 規格定義了一組函數、變數和慣例，以提供一個一致而獨立於實際所用資料庫的介面。 資料庫驅動程式（Perl DBI）的發起人為一定數量的商業資料庫引擎制定標準， 所以你只需花很少工夫便可從 Oracle 轉到 PostgreSQL。

13.2.2 PostgreSQL 的 DBD 驅動程式

在你安裝 DBD PostgreSQL （驅動程式）前，你要先安裝 DBI，從以下地方取得 DBI 驅動程式：

- 先從 `<http://www.perl.com/CPAN/modules/by-module/CPAN>` 取得所有 Perl 模組之母，打‘/usr/bin/cpan’、‘man CPAN’和數以千計的模組 `<http://www.perl.com/CPAN-local/modules/by-module>`。
- DBI 模組 `<http://www.perl.com/CPAN-local/modules/by-module/DBI>`
- DBI 模組 `<http://www.symbolstone.org/technology/perl/DBI>`
- DBI FAQ `<http://www.symbolstone.org/technology/perl/DBI/doc/faq.html>`
- Perl DBI 的參考文件 `<http://www.symbolstone.org/technology/perl/DBI>`
- DBI 通信論壇 `<http://www.fugue.com/dbi>`
- Perl 資料庫參考文件 `<http://www.perl.com/reference/query.cgi?section=database>`
- 下載 DBI rpm （注意：可能是舊版） `<http://rpmfind.net/linux/rpm2html/search.php?query=DBI>`

在此取得 DBD-Pg

- 先從 `<http://www.perl.com/CPAN/modules/by-module/CPAN>` 取得所有 Perl 模組之母，打‘/usr/bin/cpan’、‘man CPAN’和數以千計的模組 `<http://www.perl.com/CPAN-local/modules/by-module>`。

- DBD 模組 `<http://www.perl.com/CPAN-local/modules/by-module/DBD>`，找 DBD-pg 檔或 *DBD* `<http://www.perl.com/CPAN/modules/by-module/DBD>`。
- Comprehensive Perl Archive Network CPAN `<http://www.perl.com/CPAN>` 在這處選‘Database’（在 Search 盒子上方）， 擊選‘Go’按鈕。
- 預先編譯了的視窗 NT/2000 套件可在 `<http://www.edmund-mergl.de/export/DBD-Pg.zip>` 得到
- 下載 DBD rpm（注意：可能是舊版）`<http://rpmfind.net/linux/rpm2html/search.php?query=DBD>`
- Perl 模組（數以千計）`<http://www.perl.com/CPAN-local/modules/by-module>`。

13.2.3 DBI 的技術支援

- 請將意見或錯誤報告連同 perl -v 和 perl -V 的輸出、PostgreSQL 的版本、DBD-pg 的版本、DBI 的版本寄至 *E.Mergl@bawue.de*

13.2.4 DBI 文件

這裡有一些 DBI 的資料來源。

POD 文件： POD 是嵌入到 perl 程式中用以“即場”解釋程式碼的一段文件， 用以給程式設計師和模組的用家提供有用的資料。DBI 和驅動程式的 POD 正越來越流行，要閱讀有關文件，請使用以下指令。

```
DBI 規格的 PO 可用以下指令閱讀
        $ perldoc DBI

使用結合到 DBD::Oracle 的 Oraperl 模擬層的人可用以下指令閱讀如何用 Oraperl
介面編寫程式：
        $ perldoc Oraperl

DBD::mSQL 模組的用家可籍以下指令閱讀一些該驅動程式的獨家函數 (private
functions) 和特異功能的資料：
        $ perldoc DBD::mSQL

POD 文件中也包含常見問題。要閱讀的話請輸入：
        $ perldoc DBI::FAQ

POD 的一般資料—如何撰寫 POD，及整體的 POD 哲學，可籍以下指令閱讀：
        $ perldoc perlpod
```

安裝了 Tk 模組的用家可能會對一個名為 tkpod，使用 Tk 的 POD 閱讀器有興趣。它會把 POD 編排到一個方便及可閱讀的形式。

也請看看

- 來自 DBI 的通信論壇的資料 `<http://www.symbolstone.org/technology/perl/DBI/tidbits>`

- DBI Perl Journal 網頁 `<http://www.tpj.com>`
- “DBperl”這文章在一九九六年十一月的“Dr. Dobbs Journal”發表。
- “The Perl5 Database Interface”，一本由 Alligator Descartes 寫成，由 O’Reilly and Associates 出版的書。

用家可能參加的通信論壇為：

- 通信論壇 `<http://www.fugue.com/dbi>`
- **dbi-announce** 電郵至 *dbi-announce-request@fugue.com* 以 ’subscribe’ 為正文內容
- **dbi-dev** 開發人員專用 Email: *dbi-dev-request@fugue.com* 以 ’subscribe’ 為正文內容
- **dbi-users** 一般討論 Email: *dbi-users-request@fugue.com* 以 ’subscribe’ 為正文內容
- 美國通信存庫 `<http://outside.organic.com/mail-archives/dbi-users/>`
- 歐洲通信存庫 `<http://www.rosat.mpe-garching.mpg.de/mailing-lists/PerlDB-Interest>`

13.2.5 視窗 95/NT 有沒有 DBI 支援？

DBI 和 DBD::Oracle 的 Win32 版本已成為 DBI 的標準部分。因此，比 DBI 0.81 高的版本應該會正常運作。透過 ODBC，你可以存取微軟 Access 和 SQL-Server 資料庫。在 DBI-0.79（及以後）有一個 Win32::ODBC DBI 模擬層。它名為 DBI::W32ODBC。你需要使用 Win32::odbc 模組。

- Win32 DBI `<http://www.symbolstone.org/technology/perl/DBI>`
- Win32 ODBC `<http://www.roth.net>`
- 微軟 SQL server 的 Perl 介面 `<http://www.algonet.se/~sommar/mssql>`

13.2.6 商業支援及訓練

PERL CLINIC : Perl Clinic 以合約形式為 Perl、DBI、DBD::Oracle 和 Oraperl 提供商業支援。這些支援是由 DBI 作者 Tim Bunce 工作的公司提供的。欲知詳情，請看：

- 支援 `<http://www.perlclinic.com>`
- 支援 `<http://www.perldirect.com>`
- 培訓 `<http://www.westlake.com/training>`

13.3 測試 Perl 介面

請參閱6.6 (測試 Perl—PostgreSQL 介面)一節

14 PostgreSQL 管理工具

14.1 PGACCESS - 管理 PostgreSQL 的 GUI 工具

PgAccess 是 PostgreSQL 的 Tcl/Tk 介面。它已經包括在 PostgreSQL 的發行套件中。你可能想從這網頁找出它有沒有新版本：

- `<http://www.flex.ro/pgaccess>`
- 如有任何意見、改良的建議，電郵： *teo@flex.ro*

pgaccess 的用法

```
# man xhost
# xhost +
# su - postgres
bash$ man pgaccess
bash$ export DISPLAY=<主機名>:0.0
bash$ pgaccess mydatabase
```

PgAccess 的特色

PgAccess 視窗—Main window, Table builder, Table(query) view, Visual query builder.

Tables

- 開啓表格 (tables) 查閱，最多二百個紀錄（可在 perferences 菜單更改）
- 籍拖曳垂直格線 (grid line) 重整大小（在列表空間 (table space) 進行比在列表目錄 (table header) 好）
- 單格中自動換行—每個表格的配置獨立儲存
- 匯入/匯出其他檔案格式（SDF、CSV）
- 過濾功能（輸入如 price>3.14 的過濾器）
- 排序功能（自行輸入需排序的欄位）
- 即場編輯
- 改良了的表格產生助理 (table generator assistant)
- 改良了的欄位編輯

Queries

- 定義、編輯和儲存“用家定義查詢”(user defined queries)
- 把查詢儲存作為景觀 (views)
- 查詢的執行

- 觀看 select 形式的查詢結果
- 刪除和更改查詢名稱
- 具拖曳和放下功能的 Visual query builder。如果你有安裝 Netscape Navigator 的 Tcl/Tk 插件，你可在此看到它工作：（譯按：不知在哪？！）

序列 (sequence)

- 定義、刪除和檢查序列

函數 (Function)

- 使用 SQL 語言定義、檢查和刪除函數

將來還會有以下改進

- 表格設計（增加欄位、更名等等）
- 函數定義
- 報告產生器
- 基本的命令稿編寫功能

有關 LIBGTCL 的資料

你會用到 PostgreSQL 到 Tcl 的介面程式庫 libpgtcl，所謂的 Tcl/Tk 可‘載入’模組。libpgtcl 和原始碼位於 PostgreSQL 的 /src/interfaces/libpgtcl 目錄。特定來說，你需要一個可在 Tcl/Tk‘載入’的 libpgtcl 程式庫。技術上來說，它和普通的 PostgreSQL 可載入目的檔並不相同，因為 libpgtcl 是一堆目的檔 (object files) 的集合。這檔案在 Linux 名為 libpgtcl.so。你可在以上網址下載一個為 Linux i386 系統預先編譯了的版本。只需把這檔案拷貝到系統程式庫目錄（ /usr/lib） 即可。如有疑難，其中一個解決方法是刪除原始碼中有關載入 libpgtcl.so 的一句和使用 pgwish（或 wishpg）來載入 pgaccess.tcl ，而非 wish，因為這 wish 已鏈結 libpgtcl 函式庫。

如果你在編譯 pgaccess 目錄時遇到 crypt not found 的問題，請使用 -lcrypt。

14.2 PostgreSQL 圖型介面查詢工具 GtkSQL

GtkSQL 是一個圖型介面查詢工具（像 PostgreSQL 的 psql）。它以 GNU GPL 發行。使用 Gtk+ 1.2.3 和 PostgreSQL 6.3 開發而成。

GtkSQL 的主網站為 `<http://gtksql.sourceforge.net>`。

它的主要特色包括：

1. 多個 SQL 暫存區
2. SQL 關鍵字、表格名稱和欄位自動完成
3. 簡易顯示表格定義
4. PostgreSQL 和 MySQL 支援（而且容易加上其他資料庫）

GtkSQL 目前的版本為 v. 0.3。你可在 `<https://sourceforge.net/project/?form_grp=533>` 下載原始碼。

14.3 視窗中的 PostgreSQL 互動式查詢工具（WISQL 或 MPSQL）

MPSQL 提供給使用者一個圖像 SQL 介面控制 PostgreSQL。MPSQL 與 Oracle 的 SQL Worksheet 或微軟 SQL Server 的查詢工具 WISQL 類似。它有一個漂亮的 GUI 和指令記錄。你也可以剪下及貼上。它還有其他有助提高生產力的功能。

- `<http://www.troubador.com/~keidav/index.html>`
- 電郵： *keidav@whidbey.com*
- `<http://www.ucolick.org/~de/>` 在 tcl_syb/wisql.html 檔中
- `<http://www.troubador.com/~keidav/index.html>`
- 電郵： *de@ucolick.org*

14.4 名為 PSQL 的 PostgreSQL 互動式查詢工具（ISQL）

ISQL 是供文字指令行終端機使用的。這已包括在發行版本中，名為 psql。和 Sybase ISQL、Oracle SQLplus 十分相似。在 Unix 指令提示輸入‘psql’會出現 psql> 提示。

```
bash# su - postgres
bash$ man psql
bash$ psql mydatabase
打 \h 以閱讀指令的求助訊息
```

非常使用者友善和易用。 可在外框命令稿 (shell script) 中使用。

14.5 MPMGR—PostgreSQL 的資料庫管理工具

MPMGR 為 PostgreSQL 是供一個圖像管理介面。你可在以下地方找到它：

- `<http://www.mutinybaysoftware.com/>`
- Email: *keidav@mutinybaysoftware.com*
- `<http://www.troubador.com/~keidav/index.html>`
- 電郵： *keidav@whidbey.com*
- `<http://www.ucolick.org/~de>` in file tcl_syb/wisql.html
- PostgreSQL 的 WISQL `<http://www.ucolick.org/~de/Tcl/pictures>`
- 電郵： *de@ucolick.org*

14.6 PgAdmin、PhpPgAdmin 工具

- 視窗 95/NT 的 PgAdmin 工具 視窗 95/NT 的 PostgreSQL 資料庫設計工具 `<http://www.pgadmin.freeserve.co.uk>`
- 網上運作的管理工具—PostgreSQL 用的 PhpPgAdmin 在 `<http://www.phpwizard.net/projects/phpPgAdmin>`

14.7 PgBash - SQL shell 工具

PgBash 的功能類似 psql。而且，PgBash 使用 bash 的代號 (alias)、函數、歷史編輯創造出一個靈活的互動操作環境，提供有用的功能。

PgBash 的主網頁在 <http://www.psn.co.jp/PostgreSQL/pgbash/index-e.html>

PgBash 是一個包括 PostgreSQL“直接 SQL”或“嵌入式 SQL” ("direct SQL" or the "embedded SQL") 介面的 shell，籍改良 bash（目前最新版為 2.03）shell 而來。PgBash 可用作登入 shell、sub-shell（從 shell 啓動的 shell) 和 shell 程式。

在此，直接 SQL 有把結果直接輸出到標準輸出的能力。而嵌入式 SQL 有把提取出來的結果設定到 shell 變數的能力。它們則再經 shell 命令稿 (script) 語言處理。

SQL 語句（以分數作結束字完）被當作一個 shell 指令處理，我們可以執行 SQL 語句並用於管道 (pipeline)、改向和背景工作選項。此外，使用 exec_sql 指令，我們可用不同的選項來執行 SQL 語句。實際執行 SQL 的例子如下。

```
prompt> /usr/local/bin/pgbash                ……啓動 pgbash
pgbash> connect to db2@xxx.com user sakaida; ……連接到資料庫
pgbash> select * from test limit 100; | more ……使用管道
pgbash> select * from test; > /tmp/sel.dat & ……使用改向和背景工作
pgbash> addr='Osaka'
pgbash> insert into test values(
>  111,'name',                               ……可以換行
> '$addr'                                    ……使用 shell 變數
> );
pgbash> connect to db3@yyy.com user postgres;
pgbash> set connection db2;                  ……把目前的資料庫改為 db2
pgbash> select * from test;                  ……自 db2 的 test 表格選擇
pgbash> exec_sql -d db3 "select * from test3"……連接到 db3
pgbash> ls
pgbash> begin;
pgbash> declare cur cursor for select * from test;
pgbash> fetch in cur into :AA,:BB;           ……設定 shell 變數
pgbash> echo "AA=$AA, BB=$BB"
pgbash> end;
pgbash> fc fetch                             ……編輯歷史紀錄和執行
pgbash> !echo                                ……再次 echo
pgbash> disconnect all                       ……關閉所有連接
pgbash> exit                                 ……結束 pgbash
```

14.8 PostgreSQL 的 Webmin 工具

webmin 工具（透過可以是保安的網頁來管理 Unix 電腦）的最新版本（0.82 版）有一個 PostgreSQL 模組。你可使用這模組來新增用戶、群組、資料庫、表格，也可查閱表格。

你可在 <http://www.webmin.com/webmin> 找到 webmin。

15 PostgreSQL 用的中央處理器

參閱文件 <http://metalab.unc.edu/LDP/HOWTO/CPU-Design-HOWTO.html> 以得到可用於 PostgreSQL 的中央處理器名單，本文也會提供有關中央處理器的詳情。

下中央處理器 (CPUs)（32 位元和 64 位元）可執行 PostgreSQL。它們都運行 Linux。

- 處理器主網站是：Google 搜尋引擎 CPU 站“Computers>Hardware>Components>Microprocessors” <http://directory.google.com/Top/Computers/Hardware/Components/Microprocessors>

以下為 GNU/GPL 開放原始碼處理器名單：

- 開放原始碼處理器網站—Google 搜尋 "Computers>Hardware>Open Source" <http://directory.google.com/Top/Computers/Hardware/Open_Source>
- OpenRISC 1000 Free 32-bit 處理器 IP 核心，與專斷的 ARM 和 MIPS 競爭，位於 <http://www.opencores.org/cores/or1k>
- OpenRISC 2000 位於 <http://www.opencores.org>
- 歐洲太空總署的 ESA-32bit 和 ESA-64bit 處理器“LEON” Sparc <http://www.estec.esa.nl/wsmwww/leon>
- GNU/GPL Freedom 64-bit F-CPU <http://www.f-cpu.org> 或 <http://f-cpu.tux.org> 映射站台位於 <http://www.f-cpu.de>
- STM 32 位元，雙向超純量 (2-way superscalar) RISC 處理器 <http://www.asahi-net.or.jp/~uf8e-itu>
- 以 Verilog 或 VRML 寫成的免費微處理器和 DSP IP 核心 <http://www.cmosexod.com>
- 可使開發加快的免費硬件核心 <http://www.scrap.de/html/opencore.htm>
- Opencores org—開放原始碼、自由 IP core <http://www.opencores.org>
- Linux 開放硬件和免費 EDA 系統 <http://opencollector.org>
- ARM 處理器 <http://www.arm.com/Documentation>
- Cogent 處理器 <http://www.cogcomp.com>

以下為商業處理器名單：

- 俄羅斯的 E2k 64-bit CPU（很快的處理器！！！）網站：<http://www.elbrus.ru/roadmap/e2k.html> Elbrus 已和美國昇陽微系統結為合作伙伴。
- 韓國三星 (Samsung) 64 位元處理器，來自 DEC Alpha。 <http://www.samsungsemi.com> Alpha-64bit 處理器在 <http://www.alpha-processor.com> 三星和美國的 Compaq 正合作開發 Alpha 處理器。
- 英特爾 (Intel) IA 64 <http://developer.intel.com/design/ia-64>
- 全美達 (Transmeta) crusoe 處理器，和不久將來的全美達 64 位元處理器。 <http://www.transmeta.com>

- Sun Ultra-sparc 64 位元處理器 `<http://www.sun.com>` 或 `<http://www.sunmicrosystems.com>`
- MIPS RISC 處理器 `<http://www.mips.com>`
- SGI Graphics MIPS Architecture 中央處理器 `<http://www.sgi.com/processors>`
- IDT MIPS Architecture 中央處理器 `<http://www.idt.com>`
- IBM Power PC（摩托羅拉） `<http://www.motorola.com/SPS/PowerPC/index.html>`
- 摩托羅拉嵌入式處理器。根據 PowerPC、M-CORE、ColdFire、M68k 或 M68HC 核心 `<http://www.mot-sps.com>`
- M68k 或 M68K 核心 `<http://www.mot-sps.com>`
- 日立 SuperH 64-bit RISC 處理器 SH7750 `<http://www.hitachi.com>` 如果一次買一萬個，每個只售四十美元。
- Fujitsu 64-bit 處理器 `<http://www.fujitsu.com>`
- HAL (California) Super-Sparc 64-bit 處理器 `<http://www.hal.com>` 也和 Sun 的 sparc 結構相容。
- Pyramid Technologies 的 Seimens Pyramid 中央處理器
- Intel X86 系列 32-bit 中央處理器，奔騰、賽揚等等。
- AMDs X86 系列 32-bit 中央處理器，K-6、Athlon 等等。
- National's Cyrix X86 處理器 32-bit CPUs Cyrix 等等。
- 其他國家（台灣、韓國，日本）的其他中央處理器？？請告訴我……

其他重要的處理器網站為—

- 廿四小時全球處理器新聞 `<http://www.newsnow.co.uk/cgi/NewsNow/NewsLink.htm?Theme=Processors>`
- 電腦結構網詁位於 `<http://www.cs.wisc.edu/~arch/www>`
- ARM 處理器 `<http://www.arm.com/Documentation>`
- 偉大的處理器 `<http://www.cs.uregina.ca/~bayko/cpu.html>`
- Microdesign 資源 `<http://www.mdronline.com>`

16 只使用一個顯示器 (monitor) 來設定多台 PostgreSQL 電腦

如果你不想花錢於硬件的選擇器，你可使用 **VNC (Vitual Network Computing)** 技術，來自電訊業巨人 AT & T。 VNC 是 GPL 的，是一個自由軟件。籍著 VNC，你可在一個沒有顯示器的電腦執行 PostgreSQL 程式，而在遠端電腦的顯示器顯示出來！！不過，那些電腦必須以 Ethernet 網絡介面卡連接。VNC 可在 `<http://www.uk.research.att.com/vnc>` 找到。（譯按：VNC 是使用 TCP/IP，理論上只要有 IP 地址便可在遠端電腦連接，不一定要 Ethernet。不過速度可能很慢。）

你可疊起多台電腦主機及只用一個顯示器，而用一個 KWM（鍵盤、影像、顯示器） (Keyborad, Video, Monitor) （譯按：M 是否當為滑鼠？）選擇器來選擇所用電腦。 這不但節省地方，減少混亂，也節省顯示器、鍵盤和滑鼠（每份總值 100 至 500 美元）。

籍著選擇器，你可疊高多台 PostgreSQL 伺服器（開發、測試、生產）、互聯網伺服器、 ftp 伺服器、內聯網伺服器、電郵伺服器、新聞伺服器在一個大櫃。這選擇器也可控制視窗 95/NT 和 OS/2 電腦。

查看下列網址：

- DataComm Warehouse Inc，電話 1-800-328-2261。他們提供所有品種的電腦硬件 `<http://www.warehouse.com>` 4 埠手動 KVM 選擇器 (PS/2) 大約 $89.99。 Part No. DDS1354
- Network Technologies Inc `<http://www.networktechinc.com/servswt.html>` (120 美元/PC 8 埠) 的清單中包括 'Server Switches' 和 'Video only switches'
- Scene Double Inc, 英國 `<http://www.scene.demon.co.uk/qswitch.htm>`
- Cybex corporation `<http://www.cybex.com>`
- Raritan Inc `<http://www.raritan.com>`
- RealStar Solutions Inc `<http://www.real-star.com/kvm.htm>`
- Belkin Inc `<http://www.belkin.com>`
- Better Box Communications Ltd. `<http://www.betterbox.com/info.html>`
- 到附近的硬件商鋪查詢“Server Switch”或稱“KVM Auto Switches”。

在 yahoo 搜尋器中找出更多有“Server Switches”或“KVM Switches”的公司。

每個 PostgreSQL 資料庫伺服器最好用一部專用的 unix 電腦，以增強表現。任何其他程式/行程都不要在這電腦上執行。 閱讀你所住地方的報章的商業版找出有哪些商鋪售賣 Intel 電腦和 13 吋單色螢幕（十分廉宜的螢幕）。你只需要硬件，而不需微軟視窗 / DOS。資料庫伺服器不需要彩色顯示器，因為你可在一個彩色的個人電腦上遙控管理。

你可在網上商店買到單純的電腦硬件。你可在網上拍賣中找到好的價錢。

- 網上商店與拍賣場 `<http://www.egghead.com>`
- 網上商店 `<http://www.buy.com>`
- 拍賣場 `<http://www.ubid.com>`

在以下地方取得 RedHat（或其他發行版本）Linux 的光碟：

- Linux System Labs 網址： `<http://www.lsl.com/>` 7 美元
- Cheap Bytes Inc 網址： `<http://www.cheapbytes.com/>` 7 美元

確保你購買的硬件被 RedHat Linux 支援。購買前在 RedHat 的 ftp 站找出獲推薦的硬件，如 SCSI 配接器和顯示卡。用大約六百美元便可得到一台强勁的 Intel 電腦，用 RedHat Linux 來執行 PostgreSQL。從視窗 95、OS/2、Unix Motif、瀏覽器（如 Redbaron、Opera、Netscape 及其他廿多個）經 odbc/jdbc/perl/tcl 來連接 PostgreSQL。（瀏覽器正迅速成為標準的 GUI 客戶 (client)。）

使用 KWM 選擇器，你可只用一個顯示器和一個鍵盤來控制多台主機。

17 PostgreSQL 的萬維網應用程式伺服器

有些應用程式伺服器可和 PostgreSQL 合作，開放原始碼還是商業版本都有。 著名的開放原始碼萬維網應用程式伺服器有以 Perl 為基礎的 SmartWorker、WIRM、 Velocigen、Enhydra（Java）和 Zope（Python），商業的則有 IBM Websphere、BEA Weblogic。

建議你使用保安的萬維網伺服器如 Apache + mod_ssl + OpenSSL。在 `<http://www.c2.net/products/sh3>` 參閱 Redhat StrongHold 保安伺服器的資料。

萬維網應用程式伺服器可以所支援的程式語言分類。 你必須使用一個以你最喜歡的語言為基礎的萬維網應用程式伺服器。

萬維網應用程式伺服器的分類如下：

- 以 PERL 語言為基礎
- 以 PHP 語言為基礎（和 PERL 相似，也有點似 Java）
- 以 Python 語言為基礎（物件導向命令稿語言）
- 以 Java 語言為基礎（Sun Microsystems Java）
- 以 Tcl 語言為基礎（Tcl/TK—稱為“Tickle”命令稿語言
- 以 C++ 語言為基礎（C++ 和 CORBA）

17.1 PERL 萬維網應用程式伺服器

和“C”語言相似，Perl 語言已有很長壽命，而且 Perl 在未來一段很長時間還會被廣泛應用！在某些操作上，Perl 比 Java 快三倍（但 Java 在某些操作上也比 Perl 快）。 Java 是一個非常複雜的系統，既有虛擬機器，又有直譯器，使得它非常緩慢、不穩定和不可靠。 Perl 則簡潔、快速，更是物件導向。

此外，Perl 程式可輕易地編譯，以進一步改良表現。使用 Perl2Exe 這個指令列工具來把 perl 命令稿轉為可執行檔 `<http://www.indigostar.com/perl2exe.htm>`。

PERL 可使用以下萬維網應用程式伺服器

- Mason `<http://www.masonhq.com>` 是一個功能强勁，以 Perl 為基礎的網頁開發和傳送引擎。使用 Mason，你可在你的 HTML 中嵌入 Perl 碼，又可自共享、可重用的元件建立網頁。
- BingoX `<http://opensource.cnation.com/projects/BingoX>` 是一個開放原始碼、物件導向的萬維網應用架構，用 mod_perl 編寫，以大幅減少建立大型動態、資料庫驅動的網站和應用程式所需時間。 SmartWorkder 是 Perl 類別的集合，容許你的萬維網應用程式成為真正的應用程式， 而不是 HTML 樣本中隨意嵌入程式碼。 SmartWorker `<http://www.smartworker.org>`
- Apache-Perl 整合計劃 mod_perl 使得完全以 Perl 寫成 Apache 模組有可能。而且，嵌入到伺服器的持久直譯器 (persistent interpreter) 避免了啓動外部直譯器的麻煩和 Perl 的啓動時間。

  參閱 `<http://perl.apache.org>` 和 `<http://modperl.sourcegarden.org>` 的 **mod_perl_garden** 計劃。
- Apache::ASP `<http://www.apache-asp.org>` 把 Active Server Pages 移植到 Apache 萬維網伺服器，使用 Perl 作為主機的命令稿語言。Apache::ASP 容許開發人員創造包括環節管理 (session management) 功能和嵌入 Perl 碼的動態網上應用程式。另外， 它還有很多强勁的延伸功能，包括 XML tablibs、XSLT 顯現 (XSLT rendering)、還有不在原本 ASP API 的新事件。

- WIRM（Web Interface Repository Manager）是一個以 Perl 為基礎的應用程式伺服器，提供一個高階的編程環境以開發網上資訊系統 (web information systems)。WIRM 包括一個物件關聯式資料庫和一套用作視覺化 (visualizing)、整合和分析異質 (heterogeneous) 多媒體資料的 Perl 介面。WIRM 提供工具替多媒體資料庫產生對內容敏感的景觀 (context-sensitive views)， 使開發人員可以快速開發在內容和表達方式適應不同類型用家物態網頁。 參閱 `<http://www.wirm.org>`

- EmbPerl `<http://perl.apache.org/embperl>` 讓你可在 HTML 文件中嵌入 Perl 程式碼。使用 Perl 表示你可使用一種精心泡製的語言，在 WWW 早已廣泛利用。你也可用數以百計早寫好的的 Perl 模組—包括 DBI—來存取越來越多資料庫系統。

- ePerl `<http://www.engelschall.com/sw/eperl>` 直譯一個加上了 Perl 5 程式的 ASCII 檔，只處理 Perl 5 程式碼而不理會純 ASCII 數據。它可用不同方法運作：作為一個獨立的 Unix 濾器或把 Perl 5 模組整合到一般的檔案產生工作，亦可作為一種强勁的萬維網伺服器命令稿語言， 來產生動態 HTML 頁。

- XPP `<http://opensource.cnation.com/projects/XPP>` 代表‘XPP Parses Perl’或‘XPML Page Parser’，是一個快速兼有效的 HTML 分析器，可分析名為 XPML 頁的動態 HTML 頁中嵌入了的 perl和類似 HTML 的標籤。

- Gamla—一個以 Perl 為基礎的 RAD 和應用程式伺服器。gamla 計劃的目的是創造一個應用程式快速開發（RAD）工具和一個以 Perl 為基礎的萬維網應用程式伺服器。所有 Gamla 計劃產生的原程式碼都會在公衆領域 (public domain) 釋放。

  Gamla 在 `<http://gamla.iglu.org.il>`

- AxKit `<http://www.axkit.org>` 是一個 Apache（和 mod_perl）和 XML 應用程式伺服器。它能即時把 XML 轉換為任何格式，如 HTML、WAP 或純文字。 轉換過程可使用 W3C 標準技考，或自行設計的程式碼。AxKit 也使用內置的 Perl 直譯器來提供有趣而强勁的技考作 XML 轉換之用。

  AxKit 强調的是把內容和表達方法分開。AxKit 的管線 (pipelining) 技巧容許內容分不同階段轉換為可表現的格式，容許部分平台以不同方式看到數據。 AxKit 容許網頁設計員集中精力於網站設計，內容開發者純粹為內容工作， 而網站管理員也能在其强項盡其所能。

Perl 的商業萬維網伺服器

- Zelerate AllCommerce `<http://www.zelerate.org/html/eng/home.shtml>` 是一個商業、內容、客戶和關係管理系統。這個高表現、可延展的互聯網應用程式以 Perl 寫成，及用了一個後端資料庫。

- VelociGen 服務 XML、資料庫或即時送到的動態內容，速度和靜態 HTML 一樣—達到 CGI 的六十倍，而你不需修改你的應用程式。VelociGen 也藉伺服器端 XML 標籤、意外保護 (crash protection) 和多部電腦間負載平衡，使開發更簡單。

  VelociGen 可完美地加入到任何平台上的任何萬維網伺服器， 增强伺服器的表現和加快以動態內容主導的網站的反應。VelociGen 可同時處理大量要求，比 Java Servlets 快十倍或比 Cold Fusion 快四倍。

  Velocigen `<http://www.binevolve.com/velocigen>`

17.2 PHP 萬維網應用程式伺服器

PHP 可用以下萬維網應用程式伺服器

- Midgard PHP 萬維網應用程式伺服器以 PHP 命令稿語言為基礎，而 PHP 執行極為快速一比 Java 快。Midgrad 的主網站為 `<http://www.midgard-project.org>`。PHP 可用 Zend 編譯暨優化器編譯 `<http://www.zend.com>`。PHP 執行得很快，而速度比 Java 快五至十倍。

  參閱 D (Midgard 安裝) 和 PHP HOWTO `<http://www.linuxdoc.org/HOWTO/PHP-HOWTO.html>`

- Ariadne `<http://www.muze.nl/software/ariadne>` 是一個萬維網應用系統。 它包含一個完整的架構，用 PHP 輕鬆地開發和管理萬維網應用程式。 這系統使用一個模組化的方式，所有交易都使用抽象的介面。 這做法為更改系統工作或加上新功能帶來極大的自由，而且不需重新編寫其他部分。

17.3 Lutris Corp "Enhydra Enterprise" (Java)

Enhydra 支援 PostgreSQL 資料庫。

Enhydra 是一個極為流行的 Java/XML/J2EE 萬維網應用程式伺服器，由 Lutris Corporation 開發。它是全球最好的 Java/XML 萬維網應用程式伺服器。它支援 EJB、Servlets、JSP、JNDI、JDBC、JTA、CORBA、XMLC/Rocks、DODS 和國際化。它已為很多歐美的 Fortune 500 公司採用。French Telecom 一類公司正直接贊助 Enhydra。它以百分百純正 Java 寫成，可於 `<http://www.enhydra.org>` 取得。Enhydra 是一個開放原始碼專案， 但它以商業形式由 Lutris Corp 售賣和支援。請參閱 `<http://www.lutris.com>`。

在 *PostgreSQL with Enhydra* `<http://www.enhydra.org/software/documentation/enhydra/NewApp-DODS-Tutorial-PGSQL.html>` 閱讀如何設定 PostgreSQL 和 Enhydra 的指導，也請參閱 *Setup database with Enhydra* `<http://www.enhydra.org/software/documentation/enhydra/Enhydra-NewApp-DODS-Tutorial.htm>` 以了解如何為 Enhydra 設定資料庫。

你可用 Borland 的 JBuilder 和 Enhydra 合作。JBuilder 在 `<http://www.inprise.com>` 找到。

也請於 `<http://www.linuxdoc.org/HOWTO/Enterprise-Java-for-Linux-HOWTO.html>` 參閱 Enterprise Java HOWTO。

17.4 Zope（Python）

Python 漸漸成為極流行的「純」物件導向命令稿語言。Zope 是一個萬維網應用程式伺服器，提供了 PostgreSQL 介面。Zope 可在 `<http://www.zope.org>` 得到 Python 可在 `<http://www.python.org>` 找到。

17.5 OpenACS（Tcl 語言）

OpenACS (Open ArsDigita Community System) `<http://openacs.org>` 是一個用來建立可擴展、社群主導 (community-oriented) 網頁應用程式的先進工具箱。它依靠 AOLserver，一個網頁 / 應用程式伺服器，和 PostgreSQL，一個真正符合 ACID 的 RDBMS。它們是兩個和開放原始碼授權下的免費而高質的產品。

ACS 由 ArsDigita 編寫， `<http://www.arsdigita.com>`，它們的 ACS (ArsDigita Community System) 嘗試盡量獨立於資料庫，雖然它原本以 Oracle 為基礎（因此 Open ACS 要花時間進行移植）。

請參閱 `<http://www.appserver-zone.com>`。

17.6 C++, CORBA 萬維網應用程式伺服器

- PortalSphere 萬維網應用程式伺服器以 C++ 寫成，在 Unix（和 Linux）為最高速度和穩定度運行。PortalSphere 嚴格地遵從 CORBA 標準，既支援標準的 HTTP 通訊協定和 IIOP 點對點協定，以提供超高速的主從連接。 和所有著名的資料庫直接結合，這些特色使 PortalSphere 快如閃電， 而且獨家擁有在互聯網上支援實時事件的能力。 PortalSphere 可比 HTTP/CGI 快一百倍。直接連接到大部分著名的資料庫，可延展至支援超過一萬個用戶連接。

  到 `<http://www.portalsphere.com/overview.html>` 了解 PortalSphere。

- FlashPoint C++、C、PERL 萬維網應用程式伺服器計劃的目前是在一個多流 (multi-thread) 的環境支援高速網上應用程式服務、支援多樣的開發語言，包括 C 和 C++、和支援在很多其它環境下難以執行的軟件工程方式。它可和 Apache 一起工件，視乎你的需要，甚至可取代它。參閱 `<http://www.bouldersoftware.com/products/flashpoint>` 和下載 *FlashPoint Redhat RPM* `<http://www.bouldersoftware.com/products/flashpoint/download.html>` 套件。

- "C Server Pages" `<http://cserverpages.20m.com>` 是一個有效率和可延展的應用程式伺服器，以 C++寫成，使它擁有處理以 C++ 寫成的伺服器頁和嵌入了動態元素的樣版的能力。你可用你喜歡的方式，或兩種都用。 你可用 C++ 建立自己的商業物件。你的網頁可以是市場上任何 ORB 的 CORBA 客戶。

17.7 萬維網應用程式伺服器目錄

參閱 萬維網應用程式伺服器 `<http://198.85.71.76/html.html>` 目錄，內有一份“黃頁”。

18 PostgreSQL 的應用程式和工具

18.1 PostgreSQL 的 4GL 萬維網資料庫程式—AppGEN 開發系統

AppGEN 可在此下載

- `<http://www.man.ac.uk/~whaley/ag/appgen.html>`
- `<ftp://ftp.mcc.ac.uk/pub/linux/ALPHA/AppGEN>`.

AppGEN 是一種高階第四代程式語言和應用程式產生器 (application generator)，用以產生以萬維網為基礎的程式。這些程式一般在互聯網或公司內聯網使用。 AppGen 程式以合乎 Common Gateway Interface (CGI) 標準的 C 命令稿 (C scripts) 寫成，大部分互聯網伺服器都可支援。

要使用 AppGen，你需要：

PostgreSQL，關聯式資料庫系統

支援 CGI 的互聯網伺服器，如 NCSA 的 HTTPD

一個 ANSI C 編譯器，如 GCC

AppGEN 包括以下 Unix （Linux） 可執行程式：

- defgen，它會根據邏輯資料結構產生一個基礎的樣板程式。 這些程式可加入、更新、刪除和尋找資料庫中的紀錄，又可自動保持參考整合性 (referential integrity)。
- appgen，AppGEN 的編譯程式。它會把 AppGEN 的源程式碼編譯為可由 CGI 執行的 C 原始碼及可即時在網上使用 HTML 格式文件。
- dbf2sql，把 dBase III 相容的 .dbf 檔轉換為可執行的 SQL 命令稿的工具程式。這使得大部分儲存在 DOS/視窗資料庫套件的資料可移植到一個 SQL 伺服器，如 PostgreSQL。
- 此外，AppGEN 收集了一些會在執行期間使用的 HTML 文件、GIF 檔和 Java 小程式。當然，如所有好的軟件般，它也包括完整的源程式碼。

作者 Andrew Whaley 可以以下途徑接觸：

- *andrew@arthur.smuht.nwest.nhs.uk*

18.2 PostgreSQL 的互聯網介面—DBENGINE

dbengine 是 Ingo Ciechowski 所寫的一個隨插即用的 PostgreSQL 互聯網介面。它在

- `<http://www.cis-computer.com/software/dbengine/>`

有關 **DBENGINE：** dbengine 是互聯網與 Postgres95 之間的介面，只需花數分鐘， 它便可存取任何現存的資料庫。

PHP 讓你在文件中編寫類似 Perl 的語言，但不是真的 Perl，AppGEN 和 WDB-P95 需要你為每一個資料庫建立設定檔 (configuration files)——聽起來， 你似乎都要先學一類新的假語言 (meta language) 才能啓用。

和其他工具不同，你不需要學任何特別的程式或命令稿語言來使用 dbengine。也不會有每個資料庫都有設定檔的情況，所以你不需要熟習這些新結構。 不過，如果你想用到 dbengine 的完整功能，學習 Perl 會是一個好主意。

整個系統可籍處理一個附加的資料庫設定好， 它會包含有關如何理解你的資料庫存取的資料。你甚至可以指定虛假欄 (virtual fields)，用以在資料顯示在螢幕之前作出即時計算。

18.3 PostgreSQL 的 Apache 伺服器模組—NeoSoft NeoWebScript

Apache 是一個有名的互聯網伺服器。一個 PostgreSQL 到 Apache 的介面模組位於—

- `<http://www.neosoft.com/neowebscript/>`

NeoWebScript 是一程容許你把不同複雜程度的程式嵌入到 HTML 檔程式語言。

當有人要求一個包含 NeoWebScript 的 HTML 網頁時，支援 NeoWebScript 的伺服器便執行嵌入了的命令稿 (script)，產生一個含有自訂內容的網頁。

NeoWebScript 是直接在網頁的 HTML 碼編寫功能强大、 伺服器為基礎的互動程式的一個方法，高速、安全、易學。籍著 NeoWebScript，即使對新手來説，計數器 (counter)、電郵表格 (email forms)、牆紙、簽名簿 (guest books)、訪客紀錄都變得簡單。請看看 NeoWebScript 對自己、PERL 和 JavaScript 作出的比較。

果你想在你的互聯網伺服器安裝 NeoWebScript， 你的網絡管理員在開始前需先閱讀他們的 Sysop FAQ。《Theory of Operations》會解釋 NeoWebScript 如何運作，《Installation》會逐步説明。《Management》關於設定和執行伺服器的事宜，《Test》讓你確定 NeoWebScript 運作正常，《Troubleshooting》和伺服器的問題有關。

在自己的 ISP、內聯網或外聯網使用 NeoWebScript 費用全免， 你會在登記下載時得到完全的執照。不過， 如果你想把它嵌入到自己的產品或在商業的伺服器（如 SSL）使用，它總值 99 美元。

NeoWebScript 是一個容許你在網頁中嵌入 Tcl/Tk 語言作為命令稿工具的 Apache 模組。它由 Karl Lehenbauer，NeoSoft 的 Chief Technical Officer 發明，並由 Neosoft 的程式設計師和技術撰稿員編寫文件、改良及加强功能。

Apache 是世上最流行的互聯網伺服器，經調查的網站中有六十八巴仙使用。

Tcl/Tk 是由 Dr. John Ousterhout 開發的一種强勁、自由、跨平台的命令稿語言。 他説過“Tck/Tk”容許軟件開發人員比使用以 C 或 C++ 為基礎的工具快十倍完成工作。 它也是一種偉大的膠水語言 (glue language)，使得現存的程式共同工作、 更圖像化和可用於互聯網。”

NeoSoft 創辦人及 Chief Technical Officer，Karl Lehenbauer 在開始時便已參與 Tcl/Tk 開發。他也和 Mark Diehkans 一起創作了 Extended Tcl，又名為 TclX 或 NeoSoft Tcl，延伸這種語言的功能。很多 Tcl 的核心指令都來自 Extended Tcl，而由 Dr. Ousterhout 加入到核心語言中。

NeoSoft Inc., 1770 St. James Place, Suite 500, Houston, TX 77056 USA

18.4 HTML 的伺服器端延伸 HEITML 和一個 PostgreSQL 的 4GL 語言

heitml 是另一個 postgres 與萬維網間的介面。欲知詳情，請接觸

```
Helmut Emmelmann H.E.I. Informationssyteme GmbH
Wimpfenerstrasse 23 Tel. 49-621-795141
68259 Mannheim Germany Fax. 49-621-795161
```

- Mr.Helmut Emmelmann 的電郵地址 *emmel@h-e-i.de*
- Heitml 主網頁 `<http://www.heitml.com>`
- Heitml 次網頁 `<http://www.h-e-i.deom>`

Heitml 同時是 HTML 的伺服器端延伸和一種 4GL 語言。網頁編寫人員可以 HTML 的風格用類似 HTML 標籤編寫程式。

Heitml（音 "Hi"-TML）是 HTML 的延伸及一個全面的第四代語言， 它容許以互聯網為基礎的程式直接利用 SQL 資料庫中的資料，而不需使用複雜的 CGI 命令稿。

Heitml 在伺服器端延伸 HTML，動態地把“.hei”檔轉換為 HTML 格式， 使它與任何瀏覽器兼容。它包括大家熟悉、易用的 HTML 語法， 及提供大量預先開發了的標籤和程式庫來應附以往由 CGI 進行的工作。像 XML 般，heitml 容許使用者自定標籤。籍著 heitml，使用者自定的標籤可翻譯為 HTML 及傳送給瀏覽器。

Heitml 同時以 HTML 設計者和專業的程式設計師為目標。HTML 設計師可使用 heitml 標籤來建立動態的網頁，存取 SQL 資料庫，或建立完整的萬維網應用程式。計數器、 登記資料庫 (registration databases)、搜尋器表格、電郵表格、階層式選單 (hierachical menus) 都可輕鬆地籍著在大量元件函式庫 (Component Libraries) 預建的類似 HTML 的標籤產生。

對程式設計師來說，heitml 在 HTML 中嵌入了完整的第四代程式語言，

```
    (如 <if>, <while>, 和 <let> 標籤)，
```

和對含整數、實數、布林值、字串和元組 (tuple) 的表達式强勁運算能力。 元組的參考語法 (reference semantics) 就像物件導向語言，及儲存在堆疊中。Heitml 變數，包括所有在堆疊中的複雜資料結構，在 Session Mode 中可在不同網頁間維持。 你也可以自訂標籤或環境標籤 (environment tags)，甚至重新定義 HTML 標籤。

heitml 使得以下事情有可能

——以具結構及模組化 (structured and modular) 方式開發網頁，大幅減低維護困擾。

——開發有智慧及互動的網頁，內容可動態地適應用者所需。

——無需設計程式便可顯示 SQL 資料庫的內容，你只需要使用預先 定義的“dba”系列標籤。

——使用 Session Variables 開發複雜的資料庫及目錄購物程式。

Heitml 可經 CGI 介面在 Linux 的任何互聯網伺服器中執行，而在 Apache 中可使用 apache API 使得執行速度特別快（避免了 CGI 的工作）。目前，它支援 mSQL（第一及二版）、PostgreSQL（第六版）、mySQL、及 the yard databases。heitml 可在 Linux、BSDi、Solaris 和 SunOS 執行，以及在視窗 NT 加上 CGI 和 ISAPI、ODBC、視窗 95 中使用。

heitml（Linux 版）對研究、非商業性和個人用途是免費的。商業網站需附出授權費。 設有試用期限而功能完整的試用版可免費下載。（不過要注意，每個你開發的“.hei” 網頁都會印出一個訊息指出它是非商用版本。登記之後你會收到一個鍵碼， 使你可關掉這訊息而無需重新安裝。）

heitml（音“Hi”-TML）籍著可自行定義的標籤和完整的程式設計能力大幅延伸及增强 HTML 的功能。這使動態的內容和資料庫程式可在 HTML 世界中出現，而不需要 CGI 或外部手稿或程式語言。也就是說，一個 HTML 作者可在網頁中嵌入應用程式。 這只需要使用新的標籤，而不需 CGI 或程式設計。另一方面， 進階使用者或程式設計師可創造及設計强勁的標籤庫。這做法使得 heitml 對 HTML 新手和專業程式設計師共時適用。heitml 在萬維網伺服器上執行及動態地產生 HTML，所以 heitml 和互聯網標準兼容，也和任何瀏覽器兼容。 它一方面為用者提供完整的資料庫存取功能，另一方面使用者無需理會任何不必要的 CGI 複雜性。heitml 是根據有關編譯器構造和交易 (transaction) 系統的最新研究成果而開發的。

heitml 頁的開發過程和 HTML 頁一樣，都是使用文字編輯器或 HTML 編輯器，也如常地放在萬維網伺服器上。不過，新的網頁可包括動態的 heitml 標籤和使用標籤庫。你可使用這些標籤存取資料庫、產生動態內容、寄出電郵， 甚至可以開發强勁的應用程式，如登記資料庫 (registration database) 和購物系統。

HTML 新手和專業程式設計師一樣會為能夠高速而簡易地設計精采的應用程式， 如我們的互動簽名簿，而不需使用複雜而難學的 CGI 命令稿而驚訝。這工作只需使用 dba 標籤庫中的工具便可做到。

heitml 包括不同種類的標籤庫，用以產生簽名簿、資料庫維護程式、 可延伸的查詢表格、强勁的電郵表格或網頁的階層式選單 (hierarchic menu)。這些工具已可使用，你只需在網頁上加上對應的標籤。

作為一個有經驗的程式設計師，你可全面使用 heitml 的持久動態元組架構 (persistent dynamic tuple architecture)：heitml 不單是有動態型態 (dynamic typing)、全功能的表達式計算、遞迴函數和豐富的參數傳遞特色的命稿語言， 也籍動態元組架構自動保持任何大小的 session 資料。

18.5 PostgreSQL 的 America On-line AOL 萬維網伺服器

AOLserver 2.3 版是一個免費的商業萬維網伺服器，它可連接到 PostgreSQL。欲知詳情，請看

- AOL 萬維網伺服器主網頁 `<http://www.aolserver.com>`
- Philip Greenspun 的 AOLserver 介紹 `<http://photo.net/wtr/aolserver/introduction-1.html>`

AOLserver 是一個快速，全多流處理 (multithreaded)，支援 Tcl 的萬維網伺服器。但不止於此，它還是一個全面支援資料庫的萬維網開發平台。

使用 AOLserver，你可有公用 (pooled) 的 PostgreSQL（和其他 RDBMS）連接，由不同執行緒 (threads) 分享。AOLserver 有 Tcl 和 C API，讓你可開發强勁的動態網頁。所有這些東西都在 1995 年開始。它以 APL (AOLserver Public License) 或 GPL 授權，因此是全自由的軟件。

對網頁來說，Tcl API 最有用。AOLserver 有一組强勁的 Tcl 呼叫，例如 ns_sendmail（用來發出電郵）、ns_httpget（用來取回一個 URL）、ns_schedule（類以 cron 的功能，用以排定程式在特定時間執行）等等。你也可用 Tcl API 來輕易擴展 AOLserver 的功能。每一個 AOLserver 的虛擬伺服器都可擁有私家的 Tcl 命令稿“函式庫”，由 AOLserver 解釋，並可讓該虛擬伺服器中任何網頁使用得到。

你可用三種方法為 AOLserver 開發網頁：

- 純 HMTL
- .tcl 頁——即可用 ns_write 呼叫來傳回 HTML 的 tcl 程式。
- .adp 頁——AOL Dynamic Pages。你以純 HTML 開發網頁，但你可用 <% <%=

雖然 AOLserver 是一個偉大的的互聯網伺服器，擁有優良的架構， 它的光芒在於它連接資料庫的能力。AOLserver 擁有自己的資料庫抽象層 (abstraction layer)，讓你完全不需修改程式而連接到不同 RDBMS。這些 RDBMS 的連接是公用 (pooled)、持續 (persistent) 的，由不用的執行緒分享。 這容許很快的連結和有效運用資源。

AOLserver 擁有所有主要的 RDBMS 的驅動程式：PostgreSQL、Oracle、Sybase、 Informix、Illustra、Solid、Interbase 和 MySQL。

18.6 PostgreSQL 的疑難 / 計劃追蹤系統程式工具

它在

- `<http://www.homeport.org/~shevett/pts/>`

18.7 把 dbase 的 dbf 檔轉換至 PostgreSQL

dbf2msql 程式和 mSQL 及 PostgreSQL 合作愉快。你可在以下方找到

- `<ftp://ftp.nerosworld.com/pub/SQL/dbf2sql/>`
- `<ftp://ftp.postgresql.org/pub/incoming/dbf2pg-3.0.tar.gz>`

這程式由 Faculty of Electrical Engineering TU Delft, NL Computer Architecture and Digital Technique section 的 Maarten Boekhold 寫成。

- *M.Boekhold@et.tudelft.nl*

你也可使用 python 的方法 (method) 在讀入 dbf 檔及載入到 postgres 資料庫。

- See `<http://www.python.org>`

18.8 把微軟 MDB 資料庫轉換至 PostgreSQL

計劃中的 MDB Tools 是一組函式庫和公具程式，用於把微軟 Access 資料庫（mdb 檔）匯出至多用戶的資料庫，如 Oracle、Sybase、DB2、Informix、MySQL、PostgreSQL 之類。

- 從 `<http://mdbtools.sourceforge.net>` 取得 MDB Tools
- 通信論壇 `<http://lists.sourceforge.net/mailman/listinfo/mdbtools-dev>`

18.9 Zeos 客戶

“Zeos”是一個用來開發和管理資料庫應用程式的軟件產品，用於開放原始碼 SQL 伺服器一MySQL、PostgreSQL 和 InterBase `<http://www.zeos.dn.ua/eng/index.html>`

18.10 Java 的 Report Writer

Generic Report Writer 是一個目錄驅動的報告編寫器。它不是拖一放 (drag-and-drop) 介面。可用於 PostgreSQL、MySQL 和 Access。它也很可能可用於其他你有 Type 4 JDBC version 1 驅動程式的資料庫。它在 `<http://www.geocities.com/SiliconValley/Ridge/4280/GenericReportWriter/grwhome.html>` 。

19 資料庫設計工具一實體關係圖 (Entity Relation Tool) 工具

"DeZign for databases" (`<http://www.heraut.demon.nl/dezign/index.html>`) 是一個使用實體關係圖的的資料庫開發工具。它以視像形式支援實體和關係的排列， 並自動為大部分流行產生資料庫方案 (schema)。

“DeZign for databases”在設計時使用自動外鍵轉移 (automatic foregin key migration ?)，成功從單一的規範支援邏輯和實質數據等級 (data level)。 多款顯示設定包括實體/主鍵/連外鍵的屬性/不連外鍵的屬性。“DeZign for databases” 也支援定義域 (domain)（用戶自定資料型態）。

DeZign 產生的報告可將複雜的設計以簡化了的形式提交予不同等級的管理層。 你只需按一下滑鼠便可產生報告、資料字典 (datadictionaries) 和資料庫。 它支援以下資料庫：Oracle、Interbase、IBM DB2

、Sybase、微軟 Access (95/97/2000)、微軟 SQL Server、Paradox、dBase、Informix、SQL-Anywhere、MySQL 和 PostgreSQL。

Heraut“DeZign for databases”在（ `<http://www.heraut.demon.nl>`）。

20 PostgreSQL 的互聯網資料庫設計/完成工具－EARP

- `<http://www.oswego.edu/Earp>`
- `<ftp://ftp.oswego.edu>` 的‘pub/unix/earp’目錄。

20.1 EARP 是什麼？

是 David Dougherty 的“Easily Adjustable Response Program”。EARP 是一個利用 PostgreSQL 資料庫系系的網上資料庫設計/實施工具。功能包括：

- 一個視覺化的設計系統。
- 一個 sendmail 介面。（可收發電郵）
- 一個增強了的保安機制。
- 一個 cgi 驅動程式。

20.2 施行

EARP 主要施行一個在 http 監控程式 (daemon) 下執行的 CGI 二元檔來存取資料庫伺服器。所有設計工具都建立在驅動程式中，除在網上， 再沒有地方需要作出設計。那些工具本身需要一個支援圖像的瀏覽器， 所設計出來的物件獨立於編寫方法，而是根據個人好惡。

20.3 它如何工作？

EARP 的其中一個主要特色是它使用物件導向的方法來產生用以連接資料庫的 html 頁。大部分頁面都包含多個物件。每個物件都以某些工具產生及取得一個名字， 這些物件會被頁面工具 (page tool) 以一個可呼叫的順序 (callable sequence) 連結在一起。物件可在其他頁面中重用。有關 HTML、查詢、讀取使用者輸入、 可延伸的查詢和輸入物件格式化 (Extendable Formatting of Query and Input objects)、把現有物件連結成其他物件的基本工具皆有提供。 更先進工具包括電郵工具和多流查詢工具。

EARP 的另一個特色是先進的保安。有多種方法限制存取 EARP 系統的不同部分。為達到先進的保安，EARP 檢查每一個連接， 以確定每個連接人員的身分 (ids) 及群組 (groups)。不同部分的存取獨立定義， 最終的組合決定結果能否存取 EARP 的一個特定部分。此外， 實施這些保安特色只需一個支援基本（或更佳）使用者認證 (user authentication) 的 http 伺服器。

20.4 何處取得 EARP？

EARP 可籍匿名 (anonymous) ftp 取得：

- `<ftp://ftp.oswego.edu>` 的 ’pub/unix/earp’ 目錄。

21 PHP Hypertext PreProcessor—PostgreSQL 的伺服器端 html 嵌入命令稿語言

萬維網介面工具位於：

- `<http://www.php.net>`
- `<http://www.vex.net/php>`

PHP 也有一個名為 Zend 的編譯器，可大幅改善表現。首先你在開發、測試和除錯時使用 PHP 命令稿語言。當準備好正式使用時使用 Zend 編譯器產生可迅速執行的執行檔。

前身為 Professional Home Pages（PHP）而現在稱為 PHP Hypertext Pre-Processor

- 很多國定都有映射站台，名為 www.COUNTRYCODE.php.net
- `<http://www.fe.de.php.net>`
- `<http://www.sk.php.net>`
- `<http://php.iquest.net/>`
- 問題可電郵到： *rasmus@lerdorf.on.ca*

PHP 是一個伺服器端 html 嵌入命令稿語言。它讓你在你的 .HTML 檔中加入簡單的命令稿，像 JavaScript 般。不過，和 JavaScript 不同，PHP 獨立於瀏覽器。JavaScript 是一種客戶端的 html 嵌入命令稿語言，而 PHP 是伺服器端語言。PHP 的概念和 Netscape 的 LiveWire Pro 產品相似。如果你喜歡快速更新又包括源程式碼的自由軟件，你大概會喜歡 PHP 3。

- 支援 PostgreSQL 的程式碼由 Adam Sussman 寫成 *asussman@vidya.com*

21.1 主要特色

- 標準的 CGI，FastCGI 和 Apache 模組支援— 作為一個標準的 CGI 程式，PHP 可在執行任何 Unix 互聯網伺服器的任何 Unix 電腦上安裝。加上對新 FastCGI 標準的支援，PHP 可從中得到速度上的改善。作為一個 Apache 模組， PHP 變成 CGI 程式外一個强勁及快如閃電的選擇。
- 存取記錄 (Access Logging)— 籍著 PHP 的存取紀錄功能，用者可自行維護命中數字 (hit counting) 及作出紀錄。它完全不利用系統中央的紀錄檔，而且即時監察存取過程。Log Viewer Script 提供了個別用者擁有的頁面的存取摘要。此外， 本套件可被設定為在每頁產生一個顯示存取資料的註腳。本頁的結尾就有一個例子。
- 存取控制 (Access Control)— 一個以萬維網為基礎的內建設定畫面處理取控制設定。 用者可以為某人擁有的所有或個別網頁建立規則，限制誰人能及他能怎樣閱讀有關網頁。 根據用者的網域、瀏覽器、電郵地址甚至來源文件 (referring document)，網頁可以密碼保護、完全看不到、禁止紀錄 (logging disabled) 和做其他事。
- PostgreSQL 支援— Postgres 是一個先進的自由 RDBMS。利用 PHP，用者可直接在 .HTML 檔中嵌入 PostgreSQL 的「SQL 查詢」。

- RFC-1867 檔案上傳支援— 檔案上傳是 Netscape 2.0 的新功能。它讓用者把檔案上傳到萬維網伺服器。 使得這事行可的原因，是實際的 Mime 解碼由 PHP 提供，它可提供額外的框架， 在收到上傳的檔案後做些有用的事。
- HTTP 為基礎的認證控制— PHP 可用來在 Apache 萬維網伺服器中建立自定的以 HTTP 為基礎的認證機制。
- 變數、列陣、相聯列陣 (Associative Arrays)— PHP 支援變數形態、列陣、甚至如 Perl 般的相聯列陣。它們可籍 GET 或 POST 方法在網頁間傳遞。
- 條件式、While 迴圈— PHP 支援一種類似 C 的全功能命令稿語言。你可使用 if/then/elseif/else/endif 條件式、while 迴圈和 switch/case 敘述來指引你的 html 頁的顯示的邏輯流向。
- 延伸正規表達式— 正規表達式常被用於樣式匹配、樣式替換 (pattern substitutions) 和一般的字串處理。PHP 支援所有常用的正規表達式運作。
- 原始 HTTP 標頭控制— 要進行高階網站設計，使網頁視情形傳送自定 HTTP 標頭的能力是必須的。 一個慣常用法是傳送一個 Location: URL 標頭來把客戶端傳送到其他 URL。它也可用於關掉快取或處理網頁頁首上的“last update”標頭。
- 即時 GIF 圖像產生— PHP 支援 Thomas Boutell 的 GD 圖像函式庫，使它可即時產生 GIF 圖像。
- ISP“安全模式”支援— PHP 支援一個獨有的“安全模式”(Safe Mode)，使得在同一伺服器上有多個用戶使用 PHP 命令稿也沒安全問題。
- 新版本的 PHP 加入了很多新功能，請參閱主網頁 `<http://www.php.net>`
- 它是自由的！ 最後一個必要的特性。這套件是完全自由的。它以 GNU/GPL 授權，讓你可為任何目的使用這軟件，包括商業用途。欲知詳情，請參看 GNU General Public License 文件。

21.2 PHP—簡史

初出現時，PHP 是一個以 Perl 寫成小巧簡單的 cgi 包裝器 (wrapper)。這套件最先名為 Personal Home Page Tools，後來變為 Personal Home Page Construction Kit。

另有一工具是寫來簡化在網頁中嵌入 SQL 查詢的工作。它基本上是另一個 CGI 包裝器，能分析 (parsed) SQL 查詢及簡化建立根據這些查詢的表格和列表的工作。 這工具名為 FI (Form Interpreter)。

PHP/FI 2.0 版完全重寫這兩套件，將之合而為一。它進化至可把簡單電腦語言嵌入 HTML 檔中。它籍著把所簡單的命令稿直接嵌入 HTML 檔中，消除大量小型 Perl cgi 程式的需要。這樣會加快你的網頁，因為 fork 多次 Perl 的困擾已經消除。 它也簡化了管理大型網站的工作，因為你可把網頁的所有部分放在一個 html 檔中。加上多種資料庫的支援，可使用資料庫的網頁的開發變得簡單。 很多人覺得嵌入查詢比分別建立 HTML 和 CGI 檔簡單得多。

PHP/FI 已改名為 PHP。

21.3 那麼，我能用 PHP 做什麼？

如果你使用 PHP，你首先會注意到它在頁尾加上這網頁曾被瀏覽過多少次的資料 （如果你在在編譯時選取了 access logging）。這只是 PHP 所做的一小部分。 它的另一個重要角色是作為一個表格直譯 cgi 程式 (form interpreter cgi)，這也就是首名中 FI 的意思。例如，如果你在一個網頁中建立了一個表格， 你需要一個程式來處理表格上的資料。即使你只想把資料傳送到另一網頁，你也需要 cgi 程式的幫助。PHP 使得取得和處理表格的資料極為容易。

21.4 一個簡單的例子

假設你有一張表格：

```
<FORM ACTION="/cgi-bin/php.cgi/~userid/display.html" METHOD=POST>
<INPUT TYPE="text" name="name">
<INPUT TYPE="text" name="age">
<INPUT TYPE="submit">
<FORM>
```

你的 display.html 便可包括類似的東西：

```
< ?echo "Hi $ name, you are $ age years old!<p>" >
```

就是這樣簡單！PHP 自動為每個表格輸入欄位產生一個變數。之後你便可在 ACTION URL 檔中使用這些變數。

在你找出如何使用變數之後的步驟就是在網頁中試用一些邏輯流程標籤。例如， 如果你想根據用者輸入顯示不同訊息，你可使用 if/else 邏輯。在上面的例子， 我們可改變 display.html，使得不同東西會根據用戶輸入顯示出來：

```
<?
    if($age>50);
        echo "Hi $name, you are ancient!<p>";
    elseif($age>30);
        echo "Hi $name, you are very old!<p>";
    else;
        echo "Hi $name.";
    endif;
>
```

PHP 提供一種非常强勁的命令稿語言，能做到的事比以上簡單的例子多很多。欲知詳情， 請參閱有關 PHP Script Language 的章節。

你也可使用 PHP 來設定有什麼人能看到你的網頁。這可使用一個內建的設定畫面做到。 例如，籍著它你可指定只有某些網域的人可看到你的網頁， 或建立一些規則來用密碼保護某些網頁。欲知詳情，請參閱有關 Access Control 的章節。

PHP 也有能力接收任何依照 RFC-1867 規定的瀏覽器上傳的檔案。 這特色使用者能上傳文字或二元檔。籍著 PHP 的存取控制和邏輯功能， 你可全面控制准許誰上傳及上傳後做什麼。欲知詳情，請參閱有關 File Upload 的章節。

PHP 支援 PostgreSQL 資料庫套件。它支援在 .HTML 檔中嵌入 SQL 查詢。

PHP 也支援 mysql 資料庫套件。它支援在 .HTML 檔中嵌入 SQL 查詢。

21.5 CGI 改向 (Redirection)

21.5.1 Apache 1.0.x 摘要

使用 Apache 伺服器的 cgi 改向模組是執行 PHP 的一個好方法。 請注意你無須擔心改向模組，如果你使用 PHP 的 Apache 模組版本 (Apache module version)。有兩個這類改向模組。一個是 Dave Andersen 開發的：

- *angio@aros.net*

它可在以下地方找到

- `<ftp://ftp.aros.net/pub/util/apache/mod_cgi_redirect.c>`

另一個已包括在 Apache 中，名為 mod_actions.c。這些模組極為相似， 只是在用法上有少許差別。兩個都測試過，能和 PHP 順利合作。

查看 Apache 的文件關於如何加入一個模組。一般來說，你把模組名稱加到一個名為 Configuration 的檔案中。如果你使用 mod_actions 模組，你要加的一句是：

Module action_module mod_actions.o

如果你使用 mod_cgi_redirect 模組，加上以下一句：

Module cgi_redirect_module mod_cgi_redirect.o

重新編譯你的 httpd 及安裝它。要設定 cgi 改向你需要在你的 mime.types 檔中建立新的 mime 類型，或在你的 srm.conf 檔中使用 AddType 指令來加上 mime 類型。所加上的 mime 類型應類似：

```
application/x-httpd-php phtml
```

如果你使用 mod_actions.c 模組，你需在 srm.conf 檔中加上以下一行：

```
Action application/x-httpd-php /cgi-bin/php.cgi
```

如果你使用 mod_cgi_redirect.c 模組，你需在 srm.conf 檔中加上以下一行：

```
CgiRedirect application/x-httpd-php /cgi-bin/php.cgi
```

不要同時使用 mod_actions.c 和 mod_cgi_redirect.c。

安裝及正確設定了這些 cgi 改向模組後，你只需把延伸檔名定為 .phtml， 便可以指定這檔案由 PHP 分析。此外，如果你在 srm.conf 檔的 DirectoryIndex 設定行加上 index.phtml，最上層的網頁便會自動由 php 分析，如果你的 index 檔名為 index.phtml。

21.5.2 Netscape HTTPD

你可籍 Netscape Server CGI Redirection 模組來指定自動把對某些延伸檔名的檔案的要求改向給 PHP。這模組可在 PHP 網頁的 File Archieves 中找到。套件中的 README 明確地解釋了如何設定它來用於 PHP。

21.5.3 NCSA HTTPD

目前，NCSA 並不支援模組，所以，要使用 cgi 改向的話，你要修改伺服器的原始碼。 一個對 NCSA 1.5 做這工作的嵌補在在 PHP 檔案存庫中找到。

21.6 在指令行執行 PHP

如果你建立了 CGI 版本的 PHP，你只需在指令行輸入 php.cgi filename，而filename 是你想分析的檔案。你可建立獨立的 PHP 命令稿，只需把命令稿 (script) 的第一行寫為類似：

```
#!/usr/local/bin/php.cgi -q
```

“-q”阻止印出 HTTP 標頭。喜歡的話，你可不用這選擇。

21.7 PHPGem 套件

PHPGem 是一個 PHP 命令稿，用以加速編寫使用表格的 PHP 命令稿。它可用於不同的 SQL 伺服器，如 PostgreSQL、MySQL、mSQL、ODBC 和 Adabas。 你輸入表格欄位的描述和參數（欄位名稱、欄位的 on/off searching (?) 等等）， PHPGem 輸出另一個使用那些表格的 PHP 命令稿（檢視/加/修改/刪除/重覆內容和搜索）。 PHPGem 可使用多層巢狀表格 (multi-level nested tables, ??)。PHPGem 容許你指定每個表格每個欄位每人旳存取控制。PHPGem 也支援影像。

PHPGem 在 `<http://sptl.org/phpgem>` 找到。

22 PostgreSQL 的 Python 介面

Python 是一個直譯的物件導向命令稿語言。它使用簡易（輕巧的語法、 簡單直接的句子），有很多用於建立 GUI 的延伸、萬維網介面等等。 一個有智慧的瀏覽器（類似 HotJava）正在開發中（一九九五年十一月）， 這會給予程式設計員很多機會。Python 的版權持有人為荷蘭阿姆斯特丹的 Stichting S Mathematisch Centrum，可自由散佈。它包括物件、類別 (classes)、模組、和例外 (exceptions) 的動態載入。使用 C 語言來將其介面加入新系統函式庫是直接的工作，使 Python 在自行設定時容易使用。Python 是一個擁有 X 介面的高等命令稿語言。在 Linux 光碟中的 Python 套件包括大部分標準 Python 模組，以及用作 Tk 介面的 Tix widget 模組。

PyGreSQL 是一個用作 PostgreSQL 資料庫的介面。它嵌入了 PostgreSQL 查詢函式庫，使得在 Python 手稿中使用強勁的 PostgreSQL 特性十分簡單。PyGreSQL 由 D’Arcy J.M. Cain 和 Pascal Andre 寫成。

- PyGreSQL 的新網站 `<http://www.druid.net/pygresql/>`
- D’Arcy 所維護的 `<http://www.druid.net/~darcy/>`
- 舊網站在 `<ftp://ftp.via.ecp.fr/pub/python/contrib/Database/PyGres95.README>`
- D’Arcy J.M. Cain *darcy@druid.net*
- Pascal Andre *andre@chimay.via.ecp.fr*
- Pascal Andre *andre@via.ecp.fr*

22.1 如何取得 PyGres？

不同套件的主網站在：

- Python `<ftp://ftp.python.org:/pub/www.python.org/1.5/python1.5b2.tar.gz>`

- PyGreSQL `<ftp://ftp.druid.net/pub/distrib/PyGreSQL-2.1.tgz>`
- 舊網站 `<ftp://ftp.via.ecp.fr/pub/python/contrib/Database/PyGres95-1.0b.tar.gz>`

無論如何，你應嘗試尋找最接近你的映射站台。請參閱資料來源來找這些站台。PygreSQL 也會在 Python 和 PostgreSQL 站台的 contrib 目錄找到。

22.2 資料和支援

如果你需要這些套件的資料，請查看以下網頁：

- Python： `<http://www.python.org/>`
- PostgreSQL： `<http://epoch.cs.berkeley.edu:8000/postgres95/index.html>`
- PyGreSQL `<ftp://ftp.druid.net/pub/distrib/PyGreSQL-2.1.tgz>`
- PyGreSQL 舊網站： `<http://www.via.ecp.fr/via/products/pygres.html>`

支援：

- PyGreSQL 的通信論壇。要參加的話，你可電郵至： *majordomo@vex.net* 在正文中寫 “subscribe pygresql name@domain” ，把 “name@domain” 改為你自己的電郵地址。
- Python 的新聞組： newsgroup comp.lang.python
- PyGreSQL : 如有臭蟲報告、意見、注釋，請找 Andre *andre@via.ecp.fr*

22.3 測試 Python 介面

請參考 6.5 (測試 Python—PostgreSQL 介面) 一節。

23 PostgreSQL 和萬維網的閘路 (gateway) — WDB-P95

23.1 有關 wdb-p95

WDB-P95 - 一個由 J. Douglas Dunlop 開發的 PostgreSQL 資料庫萬維網介面 它在：

- J Rowe 的新 WDB 在 `<http://www.lava.net/beowulf/programming/wdb>`
- 新版本的 WWW-WDB 在 `<http://www.eol.ists.ca/~dunlop/wdb-p95/>`
- 有問題或想加入通信論壇 *dunlop@eol.ists.ca*

這是一個修改了的 wdb-1.3a2，提供一個 PostgreSQL 到萬維網的閘路。 這版本也需要能處理 HTML 表格的瀏覽器。原本的 wdb 無此要求，但要還原也不難。

你可嘗試 CASI Tape 和 Image Query。你可看看其 Form Definition File (FDF)，它也用於產生 CASI Tape 和 Image Query，它包含了兩個列表的聯結 (join)。

這版本包含所有需要用於安裝和執行以 WDB-P95 為 PostgreSQL 介面的檔案。要把這系統移植到其他資料庫應頗為簡單—如果它支援標準的 SQL 和擁有 Perl 介面。

23.2 PostgreSQL 伺服器、pgperl 和 httpd 是否需在同一電腦？

否－PostgreSQL 不需在同一電腦。因 WDB-P95 由 httpd 監控程式 (daemon) 呼叫，它們要在同一部電腦。又因為 WDB-P95 使用 Pg.pm，pgperl 也要在同一電腦。 Pgperl 是以 libpq 函式庫寫成，所以它可存取網絡上任何 PostgreSQL 伺服器，就如其他 PostgreSQL 客戶。如下所述：

(萬維網客戶 (Netscape)) => (HTTP 伺服器 (NCSA's http) + WDB-P95 + pgperl + libpq)=> (PostgreSQL 伺服器)

括號 () 代表電腦。

每部電腦可是不同類型：NT、SUN、HP……但你需要在你打算執行 WDB-P95 的電腦的 libpq 介面函式庫，因為你要用它來編譯 pgperl。（這系統設計成會使用 HTML 表格，所以最好有一新近的萬維網客戶端。）

24 "C", "C++", ESQL/C 語言介面和 PostgreSQL 的逐位運算元 (Bitwise Operators)

24.1 "C" 介面

它已包含在發行版本中，名為‘libpq’。和 Oracle OCI、Sybase DB-lib、Informix CLI 函式庫相似。

24.2 "C++" 介面

它已包含在發行版本中，名為‘libpq++’。 請參考 6.7 (測試 libpq, libpq++ 介面) 一節。

24.3 ESQL/C

PostgreSQL ESQL/C 'Embedded C Pre-compiler' 類似 Oracle Pro*C、Informix ESQL/C。PostgreSQL ESQL/C 是一種 SQL 應用程式介面 (API)，使 C 程式設計師能自行設計可擁有管理資料庫的能力的應用程式。PostgreSQL ESQL/C 容許你使用你所熟悉的第三代電腦語言而又能利用到結構化查詢語言 (SQL) 的優點。

ESQL/C 包含以下軟件：

- ESQL/C 函式庫提供具資料庫伺服器存取能力的 C 函式。
- ESQL/C 標頭檔提供對 ESQL/C 程式有用的資料結構、常數和巨集的定義。
- ESQL/C 預處理器，一個把包含 SQL 句子的 C 檔轉換為可執行檔的源程式預處理器。

  （譯按：應為把含嵌入式 SQL 的 C 檔案轉換為標準的 C 檔案，並可由 gcc 編譯為可執行檔。另外，PostgreSQL 的 SQL 嵌入式 SQL 預處理器應名為 ecpg。）

它在

- PostgreSQL 的 ESQL/C 已包含在發行版本中。
- 主網站 `<ftp://ftp.lysator.liu.se/pub/linus>`

- 電郵： *linus@epact.se*

請參考 6.9 (測試 ecpg 介面) 一節。

使用 Vim 彩色編輯器編寫‘ecpg’檔（*.pgc），你要這樣做：

```
bash$ su - postgres
bash$ mkdir $HOME/vim
建立一個名為‘$HOME/vim/myfilestypes.vim’的檔案，內有以下數行

    " myfiletypefile
        au! BufRead,BufNewFile *.pgc    set filetype=esqlc
```

你應該有一個 $HOME/.gvimrc 檔。否則請自己製造一個，請參考在 `<http://metalab.unc.edu/LDP/HOWTO/Vim-HOWTO.html>` 的 Vim HOWTO 文件。

在 $HOME/.gvimrc 加上以下一行

```
let myfiletypefile = "~/vim/myfiletypes.vim"
```

現在如果你用

```
bash$ gvim sample.pgc
```

編輯檔案，你可得到依語法突出內文 (syntax highlight) 的顏色。

24.4 PostgreSQL 的逐位元運算元

逐位元運算元由 Nicolas Moldavsky 寫成

- *nico@overnet.com.ar*

實踐逐位元運算（AND、OR、XOR、位元補數 (bitwise complement)）的 pgsql“C”函式。可經匿名 FTP 得到：

- `<ftp://ftp.overnet.com.ar/pub/utils/linux/bitpgsql.tgz>`

包括了 Linux 的 Makefile。

25 PostgreSQL 的日文假名碼 (Kanji Code)

它在以下網址：

- `<ftp://ftp.sra.co.jp/pub/cmd/postgres/>`

26 移植 PostgreSQL 到視窗 95 / 視窗 NT

視窗 NT 版的 PostgreSQL 二元檔可在以下地方找到：

- 視窗 NT PostgreSQL 二元檔 `<http://www.askesis.nl>`
- `<http://www.postgresql.org>`

下載可執行檔，將它解壓，並從第十三步開始跟從26.6 (安裝 PostgreSQL)。

如果你想重新編譯原程式碼，請跟從以下指示。移植到視窗 NT 要使用 Cygnus cygwin32 套件，它包括視窗 NT/95 的 gcc、gmake。

- Cygwin 32 套件在 `<http://www.cygnus.com/misc/gnu-win32>` 找到

到此網頁取得 cdk.exe 檔（gnu-win32 的自我解壓檔）（譯按：此工具已改名為 cygwin，網址為 `<http://sourceware.cygnus.com>`。）

26.1 NT 版的作者

PostgreSQL NT 版的作者為－

- Daniel Horak *horak@mmp.plzen-city.cz*
- Joost Kraaijeveld *JKraaijeveld@askesis.nl*
- Kevin Lo *kevlo@FreeBSD.org*
- NT 版的網頁在 `<http://www.freebsd.org/~kevlo/postgres/portNT.html>`

26.2 安裝 Cygwin 套件

1. 下載 `<ftp://go.cygnus.com/pub/sourceware.cygnus.com/cygwin/latest/full.exe>`
2. 執行 full.exe，安裝到 c:\Unix\Root 目錄。
3. 執行 Cygwin，鍵入‘mount –help’以閱讀文件。你可用 -f 選項强制掛上。之後執行“umount / " 和“mount c:\Unix\Root /”

26.3 設定 Bash 視窗

安裝 Cygwin 套件後，請進行以下工作以設定工作環境：

1. 安裝 Vi 編輯器‘Vim’。請參閱 `<http://metalab.unc.edu/LDP/HOWTO/Vim-HOWTO.html>`

2. cygwin bash 的預定視窗是一個廿四行，底色為黑色的視窗。要設定 bash 視窗的底色和大小，按開始->控制台->MS DOS 主控台，把底色設定為黑色， 大小改為七十行。

（或）右擊視窗標題列，改變內容。

3. 編輯 c:\cygnus\cywinb20 中的 cygnus.bat 和作以下設定－

```
set HOME=c:\cygnus\cygwinb20
bash --login
```

及編輯 /.bash_profile 加上這幾行

```
set -o vi
PATH=$PATH:/usr/local/bin:/usr/bin
export PATH
```

4. 要啓用指令列歷史編輯，鍵入—

bash$ set -o vi

使用 l、k、j、h，你可用 vi 指令編輯指令列歷史。你可重覆和修改之前的指令，以節省打字時間。

5. 你可用以下指令掛上磁碟機 / 目錄—

```
bash$ umount /
bash$ mount "c:\cygnus"  /
bash$ mount "c:\cygnus\cygwin-b20\postgres" /usr/local/pgsql
```

6. 用以下指令閱讀線上求助訊息—

```
bash$ mount --help
bash$ ls --help
```

26.4 安裝 Andy Piper 工具

1. 到 `<ftp://ftp.xemacs.org/pub/xemacs/aux/>` 下載 cygwin-b20-local.tar.bz2 到 c:/Unix/Root 目錄。
2. cd c:/Unix/Root; bunzip2 cygwin-b20-local.tar.bz2
3. tar -xvf cygwin-b20-local.tar
4. cd /local/bin; sh check_cygwin_setup.sh
5. 完成第四步後，你會看到以下訊息：

```
You don’t have /bin would you like to mount cygwin as /bin?"
[ y/n ]
```

選‘n’，其他問題則選‘y’。

6. mount c:/Unix/Root/cygwin-b20/H-i586-cygwin32/i586-cygwin32/bin /bin
7. cd c:/Unix/Root/cygwin-b20/H-i586-cygwin32/i586-cygwin32; mkdir libexec share man etc sbin info
8. cp -R /local/{ bin,libexec,share,man,etc,sbin,info,include }

26.5 安裝 Ludovic Lange 的 Cygwin32 IPC 套件

1. 到 `<http://www.multione.capgemini.fr/tools/pack_ipc>` 下載 cygwin32_ipc-1.03.tgz 到 c:/Unix/Root 目錄。
2. tar -zxvf cygwin32_ipc-1.03.tgz
3. cd cygwin32_ipc-1.03/src and run 'make'
4. mkdir -p c:/usr/local/{bin,include,lib,include/sys}

```
cp /cygwin32_ipc-1.03/bin/* c:/usr/local/bin
cp /cygwin32_ipc-1.03/include/sys/* c:/usr/local/include/sys
cp /cygwin32_ipc-1.03/lib/* c:/usr/local/lib
cp c:/usr/local/bin/* /bin
cp c:/Unix/Root/cygwin-b20/H-i586-cygwin32/bin/* /bin
```

5. mount c:/usr/local/bin /usr/local/bin

```
mount c:/usr/local/include /usr/local/include
mount c:/usr/local/lib /usr/local/lib
cp /local/lib/* /usr/local/lib
```

26.6 安裝 PostgreSQL

1. 下載最新的 PostgreSQL 源程式碼
2. Postgres 把所有檔案當作二元檔，所以會遇到 lf/cf 問題，我們要做第二至五步：

```
mkdir -p c:/Postgres/{Source,Binary}
mkdir c:/Postgres/Binary/pgsql
mkdir -p /usr/src/pgsql
mkdir -p /usr/local/pgsql
```

3. 把 Postgres 源程式碼拷貝到 c:/Postgres/Source 目錄，之後 tar -zxvf postgresql-6.5.3.tar.gz
4. mv postgresql-6.5.3 pgsql
5. 現在掛上各個目錄—

```
mount -b c:/Postgres/Binary/pgsql /usr/local/pgsql
mount c:/Postgres/Source/pgsql /usr/src/pgsql
mount c:/Unix/Root/cygwin-b20/share /sw/cygwin-b20/share
```

6. mkdir -p /usr/local/pgsql/{bin,include,lib,data}
7. cd /usr/src/pgsql/src/win32
8. 拷貝標頭檔—

```
cp un.h c:/Unix/Root/cygwin-b20/H-i586-cygwin32/i586-cygwin32/include/sys
cp endian.h c:/Unix/Root/cygwin-b20/H-i586-cygwin32/i586-cygwin32/include
cp tcp.h c:/Unix/Root/cygwin-b20/H-i586-cygwin32/i586-cygwin32/include/netinet
```

9. ln -s /usr/local/lib /usr/src/pgsql/src/backend/libpostgres.a
10. cd /usr/src/pgsql/src, 之後執行 './configure'
11. make > make.txt 2>&1
12. make install > make.install.txt 2>&1
13. cp /usr/local/pgsql/lib/pq.dll /usr/local/pgsql/bin
14. 在 make install 之後你要把 bin 和 lib 目錄中的文字檔轉換，以清除 cr/lf 和 eof 之類東西。
15. 在 / 目錄使用任何編輯器產生 .bashrc 如下：

```
PATH=$PATH:/usr/local/pgsql/bin:/usr/local/bin
PGDATA=/usr/local/pgsql/data
PGLIB=/usr/local/pgsql/lib
LD_LIBRARY_PATH=/usr/local/pgsql/lib:/usr/local/lib
export LD_LIBRARY_PATH PATH PGDATA PGLIB
```

16. source /.bashrc, 之後執行 'initdb –username=xxxx' 要注意資料庫系統的擁有人不可以是 root/administrator。
17. 編輯 /usr/local/pgsql/data/pg_hba.conf 檔，如：

```
host        all     163.17.11.109   255.255.255.0   trust
```

18. ipc-daemon.exe&
19. postmaster -i&
20. 執行 ' psql -h host_name template1'

27 通信論壇

27.1 PostgreSQL 的電郵戶口

以下地方取得免費電郵

- 在 Yahoo `<http://www.yahoo.com>` 擊選 e-mail
- In Lycos `<http://www.lycos.com>` 擊選 new e-mail accounts
- In hotmail `<http://www.hotmail.com>` 擊選 new e-mail accounts

訂閱 PostgreSQL 通信論壇，而 Yahoo 有一個附加特色，可為 PostgreSQL 電郵建立獨立的目錄，免致擾亂你其他的電郵。擇選目錄 Email- > Options- > Filters 及揀選 seperate folder for email。 你可用這個電郵戶口在世界任何地方收發電郵，只要你能上網。

如果你有其他電郵，你可用“Mail Filters”來自動把 PostgreSQL 郵件放到獨立的目錄，以免混亂。

27.2 英文通信論壇

請參閱主網頁的 Mailing Lists 項目：

- `<http://www.postgresql.org/>`
- 把問題電郵至： *pgsql-questions@postgresql.org*
- 開發人員 *pgsql-hackers@postgresql.org*
- 有關移植的問題 *pgsql-ports@postgresql.org*
- 文件問題 *pgsql-docs@postgresql.org*

你會在一日之內從電郵收到答案或回覆。
你也可訂閱通訊論壇。要訂閱或停止訂閱，電郵至

- *pgsql-questions-request@postgresql.org*
- *pgsql-hackers-request@postgresql.org*
- *pgsql-ports-request@postgresql.org*
- *pgsql-docs-request@postgresql.org*

訊息的正文只可有一行

subscribe
（或）

unsubscribe

27.3 通信論壇存庫

通信論壇也以 html 格式存庫在以下地方：

- 經 MHonarc 再經 WWW 的每日目錄 `<http://www.postgresql.org/mhonarc/pgsql-questions>`
- `<ftp://ftp.postgresql.org>` 目錄 /pub/majordomo

PostgreSQL 主網頁中也有一個專為 pgsql 問題而設的搜尋引擎。

27.4 西班牙文通信論壇

目前有一個「非官方」的西班牙文 PostgreSQL 通信論壇。要訂閱的話，用者需電郵至：

- *majordomo@tlali.iztacala.unam.mx*

訊息的正文只可有一行

inscripcion pgsql-ayuda

28 文件和參考書

28.1 用戶指引和說明書

PostgreSQL 發行版本中已包括以下各項的 postscript、HTML 格式 和 unix man-pages。它們在 /usr/doc/postgresql* 目錄中。如果你能上網， 你可在以下地方找到下列文件： `<http://www.postgresql.org/docs>`

- "Installation Guide"
- "User Guide" for PostgreSQL
- "Implementation Guide" 詳細說明 PostgreSQL 資料庫的內部運作
- 線上說明書
- HTML 格式的線上說明書
- Postscript 格式的線上說明書，用以列印

28.2 線上文件

- 內建資料型態和運算元的名單和解說

  `A 是否 PSQL 指令的一部分`

- 支援的 SQL 關鍵字名單

  `在源程式 /tools 目錄中有一個命令稿 (script)`

- 支援的句語名單

  `使用 psql \h 指令`

- PostgreSQL 的基本的關聯式資料庫概念（實施方法）和數個線上例子（查詢）

  ```
  看 src/test 中的 regression 測試，請參閱 <ref name="例子 RPM"
  id="Examples RPM">。你會找到 regress/sql 和 suite/*.sql 目錄。
  ```

- PostgreSQL 指導

  `SQL 指導稿件在 src/tutorial 目錄中`

  也請參閱本文件附錄乙 B (附錄乙) 的“初學者 SQL 指導”。

28.3 有用的參考書

- "Understanding the New SQL: A Complete Guide" - Jim Melton 和 Alan R.Simon 著

  `Morgan Kaufman Publisher 是其中一本最好的 SQL 書。這本書以 SQL92 為準。`

- "A Guide to THE SQL STANDARD" - C.J.Date 著

  `Addison-Wesley Publishing company 也是一本好書。有關 SQL 的名作。`

- SQL - The Standard Handbook, 一九九二年十一月

  `Stephen Cannan` 和 `Gerard Otten`
  `McGraw-Hill Book Company Europe , Berkshire, SL6 2QL, England`

- SQL Instant Reference, 1993

  `Martin Gruber, Technical Editor: Joe Celko`
  `SYBEX Inc.  2021 Challenger Drive Alameda, CA 94501`

- C.J.Date, "An introduction to Database Systems" (第六版), Addison-Wesley, 1995, ISBN 0-201-82458-2

  這本是資料庫管理系統的聖經。這本書詳細說明正規化 (normalization)、SQL、恢復 (recovery)、並行性 (concurrency)、保安、整合性 (integrity)、對原本的關聯模式的延伸、現今的話題如主從系統和物件導向模式。也包括大量參考資料以供深入研究。向大部分用戶推薦。

- Stefan Stanczyk, "Theory and Practice of Relational Databases", UCL Press Ltd, 1990, ISBN 1-857-28232-9

  這本書詳細說明關聯式資料庫、關係代數 (relational algebra)、關係演算 (calculus) 和正規化的理論。但它不包含現實世界的問題和較複雜的例子。向大部分用戶推薦。

- "The Practical SQL Handbook" 第三版， Addison Wesley Developers Press ISBN 0-201-44787-8

  向大部分用戶推薦。

- Michael Stonebraker, "Readings in Database Systems", Morgan Kaufmann, 1988, ISBN 0-934613-65-6

  這是過去數年有關資料庫的論文集。它不是給一般用者看，而是進階的學生（研究生）或資料庫開發人員。

- C.J.Date, "Relational Database - Selected Readings", Addison-Wesley, 1986, ISBN 0-201-14196-5

  這是過去數年有關資料庫的論文集。它不是給一般用者看，而是進階的學生（研究生）或資料庫開發人員。

- Nick Ryan 和 Dan Smith, "Database Systems Engineering", International Thomson Computer Press, 1995, ISBN 1-85032-115-9

  這本書深入說明存取方法和儲存技考。

- Bipin C. Desai, "An introduction to Database Systems", West Publishing Co., 1990, ISBN 0-314-66771-7

  它不是給一般用者看，而是進階的學生（研究生）或資料庫開發人員。

- Joe Celko "INSTANT SQL Programming"

  `Wrox Press Ltd.`
  `Unit 16, 20 James Road, Tyseley`
  `Birmingham, B11 2BA, England`
  `1995`

- Michael Gorman "Database Management Systems: Understanding and Applying Database"

```
Technology
QED and John Wiley
1991
```

- Michael Gorman "Enterprise Database for a Client/Server Environment" QED and John Wiley

  說明建立經由 `repository metamodels` 的主從資料庫應用程式的要求和 `ANSI`
  標準 `SQL 1993` 的應用。

還有數以百計有關 SQL 的書！到書店看看吧。

28.4 ANSI/ISO SQL 規格文件—SQL 1992, SQL 1998

ANSI/ISO SQL 規格文件可在下列網站找到：

- `<http://www.naiua.org/std-orgs.html>`
- `<http://www.ansi.org/docs>` 掣動檔案 cat_c.html 及尋找“Database SQL”。
- SQL92 標準 `<http://www.jcc.com>` 掣動檔案 sql_stnd.html。
- ANSI/ISO SQL 規格 `<http://www.contrib.andrew.cmu.edu/~shadow/sql.html>` 你會在此找到 SQL 的參考資料。

28.5 ANSI/ISO SQL 1992 的語法

請參閱本文件的附錄甲 A (附錄甲)

28.6 ANSI/ISO SQL 1998 的語法

SQL 1998 (SQL 3) 規格仍在開發中。參看本文件的“電子途徑取得 SQL3 草案” 28.9 ()

28.7 初學者 SQL 導引

參看本文附錄乙 B ()

28.8 SQL92 的時態延伸 (Temporal Extension)

- SQL92 的時態延伸的文件 `<ftp://FTP.cs.arizona.edu/tsql/tsql2/>`
- Temporal SQL-3 規格 `<ftp://FTP.cs.arizona.edu/tsql/tsql2/sql3/>`

這目錄中包括一個 SQL-92 語言規格的時態延伸。這種新語言被名為 TSQL2。

這裡的語言規格是這語言的最終版本。

可直接聯絡 TSQL2 設計委員會的主席：Richard T.Snodgrass, Department of Computer Science, University of Arizona, Tucson, AZ 85721,

- *rts@cs.arizona.edu*

TSQL2 Language Design Committee 的電郵地址和入會詳情可在語言規格最後的 一章找到。

這目錄中的內容如下。

spec.dvi,.ps TSQL2 語言規格，一九九四年九月出版

bookspec.ps TSQL2 語言規格，內容如在 TSQL2 的書般，一九九五年十月 出版（請參閱下文）

sql3 提交給 ANSI 和 ISO SQL3 籌委會的修訂建議

和語言規格一起，這裡還收集了一些有關設計上的決定的評論、提供例子、 和研究如何實踐這語言。這些評論原本是給 TSQL2 Language Design Committee 的建議。它們現在有另一個用處：提供有關 TSQL2 構造的例子、 推動設計時作出的大量決定、比較 TSQL2 和過去十五年有人提議過的大量語言。 必須强調，請些評論目前不是 TSQL2 語言規格的一部分，而是對它的補充和發展。 這語言規格已是 TSQL2 的最終決定。

這些評論，加上語言規格、數個索引和其他支援材料，已經集結成書：

Snodgrass, R.T., editor, The TSQL2 Temporal Query Language, Kluwer Academic Publishers, 1995, 674+xxiv 頁。

書中有一個節錄了的評價，完整的評價在此目錄中的 eval.ps 提供。

tl2tsql2.pl 檔是一個把獲接受的時態邏輯 (temporal logic) 翻譯為 TSQL2 的 prolog 程式。這程式由 Michael Boehlen 寫成。

- *boehlen@iesd.auc.dk*

各位可接觸他以取得一份解釋這翻譯過程的論文。這是這程式的舊版本。新版本在

- `<http://www.cs.auc.dk/general/DBS/tdb/TimeCenter/Software>`

(the TimeDB and Tiger systems).

28.9 第零部分—取得 ISO/ANSI SQL 文件

這文章說明如何（合法地）取得一份 SQL-92 標準和怎樣取得一份“現時的”SQL3 草案。

這標準版權所有， ANSI 標準的持有人為 ANSI，ISO 標準的持有人為 ISO。

現時有兩（2）個 SQL 標準，分別由 ANSI 和 ISO 出版。這兩份標準每個字都一模一樣， 除了一些小節，如文件標題、頁首、短句“International Standard”和“American Standard”之類。

購買 SQL-92 標準

The ISO standard, ISO/IEC 9075:1992, Information Technology - Database Languages - SQL，目前（一九九三年三月）在 ANSI 有存貨及可得到：

```
American National Standards Institute
1430 Broadway
New York, NY 10018 (USA)
Phone (sales): +1.212.642.4900
```

價錢 US$230.00。ANSI 版，ANSI X3.135-1992, American National Standard for Information Systems - Database Language SQL，目前並無存貨， 但應該可在一九九三年三月下旬至五月上旬間得到。估計它的價錢是 US$225.00。

果你從 ANSI 購買任何一份文件，你需要給 7% 的手續費（即大約 US$9.10）。海外郵寄肯定會更貴。ANSI 要求所有單據都伴隨一份公司訂單的複印本， 或者你可以美元寄一張美國的銀行的支票，他們會在結了賬後寄給你。（例外情況：如果你的組織是 ANSI 的公司會員 (corporate member)，ANSI 會把文件寄給你而向公司收錢。）

在美國國外，ISO 標準也可從作為 ISO (International Organization for Standardization) 或 IEC (International Electrotechnical Commission) 會員的本土國營機構（國家的標準組織）購買。國營機構及其地址的名單可從 ANSI 或其他國營機構取得。它們也可從 ISO 得到：

```
International Organization for Standardization
Central Secretariat
1, rue de Varembi
CH-1211 Genhve 20
Switzerland
```

如果你想以更方便快捷的方法訂購這標準，你需要附出代價。你可從以下組織以 US$308.00 訂購 ISO/IEC 9075:1992, Information Technology - Database Languages - SQL：

```
Global Engineering Documents
2805 McGaw Ave
Irvine, CA 92714 (USA)
USA
Phone (works from anywhere): +1.714.261.1455
Phone (only in the USA): (800)854-7179
```

我不知道這是否包括貨運費，但我估計（起碼）國際貨運會額外收費。 他們會頗為快捷地把文件寄給你，甚至會接受“主要的信用卡”。Global 還沒有 ANSI 版，或他們沒有價錢或估計時間（但我預計它會在 ANSI 出版後數週之內出現，價錢接近 US$300.00）。

購買一份 SQL3 草案 (Working Draft)

你可從 ANSI X3 Secretariat, CBEMA (Computer and Business Equipment Manufacturers Association) 購買一份 SQL3 草案的複印本。他們意圖確保能提供“最近”版本的 SQL3 草案，並以 US$60.00 至 US$65.00 出售。你可以以下途徑接觸 CBEMA。

```
CBEMA, X3 Secretariat
Attn: Lynn Barra
1250 Eye St.
Suite 200
Washington, DC 20005 (USA)
```

Lynn Barra 也可以電話號碼 +1.202.626.5738 接觸以要求一份，不過郵寄似乎較有禮貌。

電子徑取得 SQL3 草案

最新版本（以寫本文時為準）的 SQL3（ANSI 和 ISO）草案（和所有附件） 可在以下站台以“anonymous ftp”或“ftpmail”得到：

```
gatekeeper.dec.com
```

在

```
/pub/standards/sql/
```

這目錄中有多個檔案，包括 PostScript 檔和純文字檔（格式不很美觀， 但可在螢幕上無須特別軟件閱讀）。

一般來說，你可找到類似名稱的檔案：

```
sql-bindings-mar94.ps
sql-bindings-mar94.txt
sql-cli-mar94.ps
sql-cli-mar94.txt
sql-foundation-mar94.ps
sql-foundation-mar94.txt
sql-framework-mar94.ps
sql-framework-mar94.txt
sql-psm-mar94.ps
sql-psm-mar94.txt
```

完成新版本的文件後，“mar94”會改變來說明新的出版日期（例如“aug94”是估計在 “mar94”之後的版本）。

另外，為照顧不能從 FTP 取得檔案名單的讀者，我們在目錄中放置了一個名為

```
ls
```

的檔案。這檔案（神奇！）包括了目錄內的檔案名單。

從 Ftp 直接獲得檔案

這是使用 FTP 的例子。它特別說明如何連接到 gatekeeper.dec.com， 到達放置這些檔案的目錄，及把檔案傳送到你的電腦。要注意的的電腦要有上網功能。 簽入名稱是‘ftp’，密碼則是你的電郵地址（即所謂‘anonymous ftp’）。指令‘type binary’是用來確保收到的檔案中沒有位元被截去。‘get’一次取得一個檔案。 在命令稿 (script) 中的注解會在角括號內 < 即是如此 >。

```
% ftp gatekeeper.dec.com
Connected to gatekeeper.dec.com.
220- *** /etc/motd.ftp ***
     Gatekeeper.DEC.COM is an unsupported service of DEC Corporate Research.
     <…等一會…>
220 gatekeeper.dec.com FTP server (Version 5.83 Sat ... 1992) ready.
Name (gatekeeper.dec.com:<你的登入名稱>): ftp  <anonymous 也可>
331 Guest login ok, send ident as password.
Password: <鍵入你的電郵地址>
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.  <什麼也沒所謂>
Using binary mode to transfer files.
ftp> cd pub/standards/sql
250 CWD command successful.
ftp> dir
```

```
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 9529
-r--r--r--  1 root     system     357782 Feb 25 10:18 x3h2-93-081.ps
-r--r--r--  1 root     system     158782 Feb 25 10:19 x3h2-93-081.txt
-r--r--r--  1 root     system     195202 Feb 25 10:20 x3h2-93-082.ps
-r--r--r--  1 root     system      90900 Feb 25 10:20 x3h2-93-082.txt
-r--r--r--  1 root     system    5856284 Feb 25 09:55 x3h2-93-091.ps
-r--r--r--  1 root     system    3043687 Feb 25 09:57 x3h2-93-091.txt
226 Transfer complete.
ftp> type binary
200 Type set to I.
ftp> get x3h2-93-082.txt
200 PORT command successful.
150 Opening BINARY mode data connection for x3h2-93-082.txt (90900 bytes).
226 Transfer complete.
90900 bytes received in 0.53 seconds (166.11 Kbytes/s)
ftp> quit
% <這檔案已成為你的電腦中的 x3h2-93-082.txt>
```

沒有 ftp 支援的情況下取得檔案

Digital Equipment Corporation 和數間其他公司相似，提供 ftp email 服務。它可能需要數天才有回應，但它為不能直接上網的人提供了等同 ftp 的服務。伺服器的地址為：

ftpmail@decwrl.dec.com

以下命令稿會取得最新版本的 SQL3 文件的 PostScript 版：

```
reply joe.programmer@imaginary-corp.com
connect gatekeeper.dec.com anonymous
binary
compress
uuencode
chdir /pub/standards/sql
get x3h2-93-091.ps
quit
```

第一句指令告訴伺服器要把檔案傳給你，你需要把 “joe.programmer@imaginary-corp.com”換為你的電郵地址。這例子中的檔案 x3h2-93-091.ps 會以 uuencode 了的 compress 檔格式，分成三十四封電郵回覆。 如果你的環境沒有提供重組檔案的工具，你需要用下列命令稿以純文字模式取得檔案：

```
reply joe.programmer@imaginary-corp.com
connect gatekeeper.dec.com anonymous
chdir /pub/standards/sql
get x3h2-93-091.ps
quit
```

但請注意，這 .ps 檔大概會被分為超過七十份來回覆！

要取得任何不是 x3h2-93-091.ps 的檔案，只需把“x3h2-93-091.ps” 換為你想要的檔案的名稱。要取得所有檔案的名單，把“x3h2-93-091.ps”換為 “dir”。

28.10 第一部分—ISO/ANSI SQL 當前情況

這章是有關 SQL 標準和其當前情況的資料來源。

前情況：

目前正在進行開發工作，以改進 SQL 為為計算上完整的語言 (computationally complete language)，以可用於持久、複雜的物件。這包括：一般化和專門化等級 (generalization and specialization hierarchies)、多級繼承 (multiple inheritance)、自訂資料形態、激發 (triggers) 和判斷 (assertions)、知識庫系統支援 (support for knowledge based systems)、迴遞查詢式 (recursive query expression) 和附加資料管理工具。它也包括抽象資料型態 (ADTs)、物件識別字 (object identifiers)、方法 (methods)、繼承 (inheritance)、多型 (polymorphism)、包裝 (encapsulation)、 和所有其他與物件資料管理有關的功能的規格。

一九九六年秋季，SQL3 數個部分進行了一次 ISO CD 不記名投票。這數部分是 SQL/Framework、SQL/Foundation 和 SQL/Bindings。一如所料， 投票結果是在收到約九百個評論下遭否決。一月下旬舉行了一次 ISO DBL 編輯會議 (editing meeting)，處理了大量在評論中提及或另外提交的問題解決方法。 由於該次會議未能處理所有評論，會議將會延長。 編輯會議定於一九九七年七月在倫敦完成。

據估計，在七月的編輯會議之後，會再提出對有關那些部分的 SQL 進行最後的 CD 投票。最後 CD 過程會花大約六個月，之後會有 DBL 編輯會議、DIS 投票和快捷的 IS 投票。

ISO 的程序在 SQL/92 之後改變了，所以 SQL 委員會仍在摸索這過程的詳情。

如果事事順利，SQL3 的這些部分會在 1998 年後期成為官方 ISO/IEC 標準，不過時間非常緊迫。

在一九九三年，ANSI 和 ISO 開發委員會決定把之後的 SQL 發展分為多部分的標準。這些部分包括：

- Part 1: Framework A non-technical description of how the document is structured.
- Part 2: Foundation The core specification, including all of the new ADT features.
- Part 3: SQL/CLI The Call Level Interface.
- Part 4: SQL/PSM The stored procedures specification, including computational completeness.
- Part 5: SQL/Bindings The Dynamic SQL and Embedded SQL bindings taken from SQL-92.
- Part 6: SQL/XA An SQL specialization of the popular XA Interface developed by X/Open
- Part 7:SQL/TemporalAdds time related capabilities to the SQL standards.

在美國，SQL3 的整體同時當作 ANSI 本土 (Domestic, "D") 專案和 ISO 專案處理。目前估計完成時間是一九九九年。

SQL/CLI 和 SQL/PSM 現正以最高速度處理，以作為 SQL-92 的修訂。在美國， 它們只被當作國際 (International, "I") 專案處理。 SQL/CLI 在一九九五年完成了。SQL/PSM 應該會在 1996 年未段完成。

除了 SQL3 外，還有一些附加的計劃：

- SQL/MM An ongoing effort to define standard multi-media packages using the SQL3 ADT capabilities.
- Remote Data Access (RDA)

標準委員會和過程 (Standards Committee and Process)

事實上，世上有多個 SQL 標準委員會。ISO 有一部分是一個國際 SQL 標準小組。多個國家都有以 SQL 為重點的委員會。這些國家（通常）派出代表參與 ISO/IEC JTC1/SC 21/WG3 DBL 會議。主動參與 ISO SQL 標準制定的國家是：

- 澳洲
- 巴西
- 加拿大
- 法國
- 德國
- 日本
- 韓國
- 荷蘭
- 英國
- 美國

NIST 確認 (NIST Validation)

在美國，SQL 實踐的確認工作由 National Institute of Standards and Training (NIST) 進行。NIST 目前有一套入門級 (entry level) SQL-92 的測試套件。NIST 確認要求的詳情被定義為一項 Federal Information Processing Standard (FIPS)。目前對 SQL 的要求定義於 FIPS 127-2。本文件的 Postscript 和文字模式可從 NIST 取得。目前通過 SQL 確認的產品名單也可在 NIST 取得。

標準 SQL 刊物和文件 (Standard SQL Publications and Articles)

SQL 標準有兩個版本。兩個都可從 ANSI 取得：

- ISO/IEC 9075:1992, "Information Technology — Database Languages — SQL"
- ANSI X3.135-1992, "Database Language SQL"

除封面和對其他標準的參考外，SQL 標準的兩個版本完全一樣。 兩個版本都可從以下地方取得：

```
American National Standards Institute
1430 Broadway
New York, NY 10018
USA
Phone (sales): +1.212.642.4900
```

除 SQL-92 標準外，現在還有一份技術正誤表 (Technical Corrigendum)（除錯）：

```
* Technical Corrigendum 1:1994 to ISO/IEC 9075:1992
```

TC 1 應該可從 ANSI 取得。TC 1 只有 ISO 版本——它適用於 ISO 和 ANSI 版本的 SQL-92。

除標準外，也有關於 1992 SQL 標準的書。這些書提供一個比實際標準易理解的描述。

有關的標準

SQL 社群也對多個其他標準有興趣。這節包括一些有關這些努力的指針。 當更多資料在網上出現後，這個名單也會更長。

- SQL Environments (FIPS 193)
- Next Generation Repository Systems (X3H4) - 一份呼籲參與 "Developing Standards for the Next Generation Repository Systems" 的新聞稿。

28.11 第二部分—ISO/ANSI SQL 基金 (ISO/ANSI SQL Foundation)

SQL3 努力的一個重要部分在 SQL 基金文件中：

- 基本的 SQL/PSM 能力（來自 SQL/PSM-92）
- 新的資料型態
- 激發 (Triggers)
- 子表格 (Subtables)
- 抽像資料型態 (Abstract Data Types, ADT)
- 物件導向能力

對物件導向能力有數個先決要求：

- 定義複雜運作的能力
- 把複雜的運作儲存在資料庫中
- 呼叫外部程序，有些不能用 SQL 做到的運作，或需要和外面溝通

這些能力已定義為 SQL/PSM 的一部分

現在有大量工作於煉製 SQL-3 物件模型，使它和 ODMG 提出的物件模型看齊。這努力在 X3H2 和 ISO DBL 文章中描述過：容納 SQL3 和 ODMG。在 SQL3/OQL Merger 最近的更新版也可得到。

SQL3 時間表

有關 SQL3 的工作正在進行，但最終標準還有待數年後才出現。

- 國際性投票把 SQL3 基礎從草案提升為委員草案 (Committee Draft, CD) 在一九九六日秋季進行。
- 估計投票會帶來大量評論
- 將會需要第二次 CD 投票
- Draft International Standard 投票多半會在 1998 年中進行
- 國際標準可在一九九九年中完成

ANSI 版的標準的編排也差不多。

28.12 第三部分—ISO/ANSI SQL 呼叫級介面 (Call Level Interface)

SQL/CLI 是一種對 SQL 資料庫的呼叫級介面。它設計為支援包裹 (shrink-wrapped) 的應用程式。CLI 原本由 SQL Access Group (SAG) 的一個小組委員會創造。 SAG/CLI 規格在 1992 年以微軟 Open DataBase Connectivity (ODBC) 規格公佈。一九九三年， SAG 把 CLI 提交到 ANSI 和 ISO SQL 委員會中。(SQL Access Group 現已結合到 X/Open consortium 中。)

SQL/CLI 提供了以下事項的國際標準：

- 獨立於實踐方法的 CLI 來存取 SQL 資料庫
- 主從工具 (Client-server tools) 可輕易籍動態鏈結函式庫存取資料庫
- 支援及鼓勵豐富的主從工具

SQL/CLI 時間表

對標準的開發過程，SQL/CLI 正以驚人的速度處理中。

- SQL/CLI 是對 1992 SQL 標準（SQL-92）的補遺
- 一九九五年完成，成為 ISO 標準
- ISO/IEC 9075-3:1995 Information technology – Database languages – SQL – Part 3: Call-Level Interface (SQL/CLI)
- 目前 SQL/CLI 正努力加上 SQL3 功能

28.13 第四部分— ISO/ANSI SQL 持久儲存模組 (Persistent Stored Modules)

SQL/PSM 籍以下功能擴充 SQL：

- 程序語言 (Procedural language) 延伸
- 複句 (Multi-statement) 和儲存程序
- 外部函數與程序呼叫 (External function and procedure calls)

除作為有價值的應用程式開發工具，SQL/PSM 為 SQL3 提供了物件導向能力的基礎支援。

複句和內存程序

複句和內存程序為主從環境提供多種好處：

- 表現—由於內存程序可執行多句 SQL 語句，與客戶經網絡的溝通可減少。
- 保安—用者可以有權利經內存程序更新一個或多個表格，但沒權直接更新。
- 分享程式碼—不需每個存取資料庫客戶工具都重寫和重試內存程序的程式碼。
- 控制—為應用程式的執行提供單一的定義和控制。

程序語言延伸

程序語言籍流向控制和其他編程技巧為 SQL 加上傳統程式語言的功能。

流向控制語句

- If-then-else
- 迴圈結構 (Looping constructs)
- 例外處理 (Exception handling)
- Case 語句
- Begin-End 區塊

程序語言延伸還包括其他編程技巧：

- 變數宣告
- Set 語句，用於賦值 (assignment)
- 取得行程 (process) 和狀態的資料

此外，所有傳統 SQL 語言都可包含在多句的程序中。

外部函數與程序呼叫

一個經常在大量資料庫產品的 wish lists 中出現，而有部分實踐到的特色， 是呼叫用者所寫，在資料庫軟件外的程序。

- 容許某個電腦或應用程式加上自己的資料庫函數
- 可在整個資料庫應用程式使用

這功能的好處是它讓資料庫（因此包括資料庫應用程式）使用豐富的程序和函數， 豐富得不能由一個標準組織定出。

SQL/PSM 時間表

SQL/PSM 發展迅速：

- SQL/PSM 是 SQL-92 的補遺
- 有關把 SQL/PSM 從國際標準的草案變為國際標準的國際投票在一九九六年一月完成。
- 一九九六年五月的編輯會議未能解決所有評論
- PSM 編輯會議安排於一九九六年九月三十日至十月四日期間重開
- 安排緊迫，但 PSM 有可能在一九九六年間出版
- 官方任命為： ISO/IEC DIS 9075-4:199? Information technology – Database languages – SQL – Part 4: SQL Persistent Stored Modules (SQL/PSM)
- 現正為 SQL3 加上 SQL/PSM 支援工作。

28.14 第五部分一 ISO/ANSI SQL/結合 (Bindings)

為方便參考，程式語言的結合被抽出成獨立的文件。目前的版本即是 SQL-92 的動態和嵌入結合 (dynamic and embedded bindings) 的節錄。

式語言結合仍有不同的問題有待解決。

對傳統程式語言來說，資料型態和 SQL-92 的可以互相對應。不過， SQL 物件和程式語言變數之間的對應必須定義。

對物件導向語言來說，目前的 SQL 資料型態、SQL 物件模型和物件導向程式語言之間的對應關係必須定義。

物件模型需在公開這些之前穩定下來。

語言結合完成時會成為 SQL3 的一部分。

28.15 第六部分一ISO/ANSI SQL XA 介面專門化 （SQL/XA）

這規格會為總交易管理員 (global Transaction Manager) 和 SQL 資源管理員 (SQL Resource Manager) 之間應用程式介面 (API) 提供標準。它會根據 ISO/IEC 10026 的內容，“分散交易處理 (Distributed Transaction Processing）” 標準化函數呼叫，SQL 資源管理員會用以支援分二階段的確定 (two-phase commit)。基礎的文件是自經 X/Open 批準而來自一份 X/Open 出版物，它清楚的指明 xa_close, xa_commit, xa_complete, xa_end, xa_forget, xa_open, xa_prepare, xa_recover, xa_rollback 和 xa_start 等多個函數，以 SQL 資料型態的輸入和輸出參數的語法。

ISO 目前正嘗試盡快追上 (fast-tract) X/Open XA 規格。 這個過程原封不動採用一個目前的業界規格。以 ISO SC21，JTC 1 等級的 XA 盡快追上投票在一九九五年四月二十七日開始，一九九五年十月二十七日結束。如果 XA 規格經 75% 票數和 2/3 的 JTC 1 p-members 批準，它會成為一個國際標準。如果經盡快追上投票批準，SQL/XA 可在 1996 年成為標準。

28.16 第七部分一時態 ISO/ANSI SQL

時態 SQL 處理和時間有關的資料。其概念是在查詢資料時利用它在一個特定時間的情況。 時態 SQL 是一份由 Rick Snodgrass 在一九九四年十二月所寫，描述這概念的文件。

X3 宣佈確認一個新計劃，ISO/IEC 9075 Part 7: SQL/Temporal 是一份關於 SQL/Temporal 的新聞稿

```
------------------------------------------------------------------------
                              Temporal SQL
                              ************
Rick Snodgrass（TSQL2 委員會主席）
31-Dec-1994
```

有人質疑過 SQL 3 對附加時間支援的問題（如在 DBL R10-75 提議，要求 SQL 有新的部分來處理時間性的資料庫）。其論點是抽象資料形態 (ADT's) 足以支援時態。 在這資訊性的項目，我使用具體例子來論證擁有抽象資料形態的欄位不能處理時態查詢。 特別來說，很多時間性的查詢不是很難用 SQL 模擬，就是需在過程語言中使用嵌入式 SQL。其他選擇在 TSQL2，一個對 SQL-92 的時態延伸中提出了。

28.16.1 導言

有效時間 (Valid-time) 的支援超出一個時態 ADT 的能力。使用後者， 一個欄位被指定為時態領域，如 DATE 或 INTERVAL（稍後會有例子）。籍著有效時間， 表格的每行隨著時間改變，因為現實改變了。聯繫到一個有效時間表格的某行的時間戳記 (timestamp)，會被查詢語言理解為該行中各欄數值的組合有效的時間。 這個無須言明的時間戳記使得查詢可以簡單易明地表達出來。

28.16.2 個案研究—儲存現時資料

The University of Arizona 的 Office of Appointed Personnel 在資料庫中有些資料，包括僱員名字、目前薪金和目前職位。 這些可用一個簡單的表格表達出來。

```
Employee(Name, Salary, Title)
```

使用這個表格尋找一個僱員的薪金十分簡單。

```
SELECT Salary
FROM Employee
WHERE Name = 'Bob'
```

現在 OAP 想再記錄出生日期。為些，在表格中加了一欄，產生以下綱目 (schema)：

```
Employee(Name, Salary, Title, DateofBirth DATE)
```

找尋一名僱員的出生日期個尋找薪金類似。

```
SELECT DateofBirth
FROM Employee
WHERE Name = 'Bob'
```

28.16.3 個案研究—儲存歷史資料

OAP 想把受僱歷史電腦化。為此，他們加上兩欄， 一欄指出表示此行的資料何時開始有效，另一指出資料何時失效。

Employee (Name, Salary, Title, DateofBirth, Start DATE, Stop DATE)

以資料模型 (data model) 說，這些新欄位和 DateofBirth 毫無分別。不過， 它們的存在產生多種結果。

28.16.4 個案研究—投映 (Projection)

要找出僱員目前的薪金會較為困難。

```
SELECT Salary
FROM Employee
WHERE Name = 'Bob' AND Start <= CURRENT_DATE AND CURRENT_DATE <= Stop
```

這句查詢比之前的複雜。元凶很明顯是新增的兩欄。OAP 想發出薪金歷史給各員工。 特別地，有需要找出每一名僱員在收受一個薪酬的最長時間。不幸地，用 SQL 不能做到。一名僱員在薪酬調整期間職銜可有多次變更。

```
Name     Salary  Title              DateofBirth    Start           Stop
----     ------  -----              -----------    -----           ----
Bob      60000   Assistant Provost  1945-04-09     1993-01-01      1993-05-30
Bob      70000   Assistant Provost  1945-04-09     1993-06-01      1993-09-30
Bob      70000   Provost            1945-04-09     1993-10-01      1994-01-31
Bob      70000   Professor          1945-04-09     1994-02-01      1994-12-31
```

圖 1

要注意有三行 Bob 的薪金停留在 $70,000。所以，結果應該是 Bob 有兩行。

```
Name     Salary  Start           Stop
----     ------  -----           ----
Bob      60000   1993-01-01      1993-05-30
Bob      70000   1993-06-01      1994-12-31
```

另一方法是把薪金和職級資料印給用者，而由用者找出他的薪酬何時變更。 這方法既不吸引，亦不實際。另一方法是盡量使用 SQL。

```
CREATE TABLE Temp(Salary, Start, Stop)
AS      SELECT Salary, Start, Stop
        FROM Employee;
```

repeat

```
        UPDATE Temp T1
        SET (T1.Stop) = (SELECT MAX(T2.Stop)
                         FROM Temp AS T2
                         WHERE T1.Salary = T2.Salary AND T1.Start < T2.Start
                                AND T1.Stop >= T2.Start AND T1.Stop < T2.Stop)
        WHERE EXISTS (SELECT *
                      FROM Temp AS T2
                      WHERE T1.Salary = T2.Salary AND T1.Start < T2.Start
                                AND T1.Stop >= T2.Start AND T1.Stop < T2.Stop)
        until no rows updated;

DELETE FROM Temp T1

WHERE EXISTS (SELECT *
              FROM Temp AS T2
              WHERE T1.Salary = T2.Salary
                        AND ((T1.Start > T2.Start AND T1.Stop <= T2.Stop)
                        OR (T1.Start >= T2.Start AND T1.Stop < T2.Stop))
```

迴圈中找出重疊或相鄰而需要合併的時段。在最壞的情況，迴圈會執行 log N 次，N 指一連串重疊或相鄰的同薪金紀錄的數量。讀者可自行模擬查詢在以上例子的執行， 以驗證它的正確性。

第三種方法是使用 SQL 為表格開啓一個游標 (cursor)。為每個薪金管理一個各段落的鏈表 (linked list)。這鏈表開始時需為空白。

```
DECLARE emp_cursor CURSOR FOR
```

```
             SELECT Salary, Title, Start, Stop
             FROM Employee;
     OPEN emp_cursor;
     loop:
             FETCH emp_cursor INTO :salary, :start, :stop;
             if no-data returned then goto finished;
             find position in linked list to insert this information;
             goto loop;
     finished:
     CLOSE emp_cursor;
```

重覆至鏈表結束，印出日期和薪金。

如果游標是 ORDER BY Start，鏈表並非必要。

無論如何，一句簡單的查詢是不易用 SQL-92 的功能表達出這目的。使用 TSQL2 的話這查詢會很簡單。

```
             SELECT Salary
             FROM Employee
```

28.16.5 案例研究－連結 (JOIN)

一個更激進的方法是把綱目重整，把薪金、職級、出生日期資料分開， 從而避開抽取薪酬歷史時遇到的問題。（為方便計，我們以後的討論中不再考慮出生日期。）

```
             Employee1 (Name, Salary, Start DATE, Stop DATE)
             Employee2 (Name, Title, Start DATE, Stop DATE)
```

Employee1 表格如下：

```
     Name    Salary  Start          Stop
     ----    ------  -----          ----
     Bob     60000   1993-01-01     1993-05-30
     Bob     70000   1993-06-01     1993-12-31
```

這是 Employee2 表格。

```
     Name    Title                  Start          Stop
     ----    ------                 -----          ----
     Bob     Assistant Provost      1993-01-01     1993-09-30
     Bob     Provost                1993-10-01     1994-01-31
     Bob     Professor              1994-02-01     1994-12-31
```

改變之後，取得僱員薪金資料變得簡單。

```
             SELECT Salary, Start, Stop
             FROM Employee1
             WHERE Name = 'Bob'
```

但如果 OAP 想一個表格中包括薪金、職位的時段（即 OAP 想一個像圖 1 形式的表格）？一個方法是印出兩個表格，而讓用者找出有關組合。另一方法是完全使用 SQL。不幸地，這查詢要分析 Employee1 每行和 Employee2 的重疊情況，有四種可能出現。

```
SELECT Employee1.Name, Salary, Dept, Employee1.Start, Employee1.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee2.Start <= Employee1.Start AND Employee1.Stop < Employee2.Stop
UNION
SELECT Employee1.Name, Salary, Dept, Employee1.Start, Employee2.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee1.Start >= Employee2.Start AND Employee2.Stop < Employee1.Stop
        AND Employee1.Start < Employee2.Stop
UNION
SELECT Employee1.Name, Salary, Dept, Employee2.Start, Employee1.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee2.Start > Employee1.Start AND Employee1.Stop < Employee2.Stop
        AND Employee2.Start < Employee1.Stop
UNION
SELECT Employee1.Name, Salary, Dept, Employee2.Start, Employee2.Stop
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
     AND Employee2.Start > Employee1.Start AND Employee2.Stop < Employee1.Stop
```

正確找出所有情形是一件具挑戰性的工作。使用 TSQL2，進行一個具時態的連結就如人們所期望。

```
SELECT Employee1.Name, Salary, Dept
FROM Employee1, Employee2
WHERE Employee1.Name = Employee2.Name
```

28.16.6 個案研究—聚集 (AGGREGATES)

現在 OAP 被問及最高薪酬是什麼？加上時間前，這很簡單。

```
SELECT MAX(Salary)
FROM Employee
```

現在加上了薪金歷史，我們要找出不同時間最高薪金的歷史。問題當然是 SQL 並不提供具時態的聚集。簡單的方法是印出有關資料，用人手找出最大值。 另一個方法是取巧，把一個當時的 (snapshot) 聚集查詢轉化為不使用聚集的查詢， 再轉化為時態聚集。那個不使用聚集的查詢用於找出不大於它本身數值的薪金。

```
SELECT Salary
FROM Employee AS E1
WHERE NOT EXISTS (SELECT *
                  FROM Employee AS E2
                  WHERE E2.Salary > E1.Salary)
```

把這個查詢變為時態查詢並不顯淺。以下是一種方法。

```
CREATE TABLE Temp (Salary, Start, Stop)
AS      SELECT Salary, Start, Stop
        FROM Employee;
INSERT INTO Temp
        SELECT T.Salary, T.Start, E.Start
        FROM Temp AS T, Employee AS E
        WHERE E.Start >= T.Start AND E.Start < T.Stop AND E.Salary > T.Salary;

INSERT INTO Temp
        SELECT T.Salary, T.Stop, E.Stop
        FROM Temp AS T, Employee AS E
        WHERE E.Stop > T.Start AND E.Stop <= T.Stop AND E.Salary > T.Salary;
DELETE FROM Temp T
WHERE EXISTS (SELECT *
              FROM Employee AS E
              WHERE ((T.Start => E.Start AND T.Start < E.Stop)
                        OR (E.Start >= T.Start AND E.Start < T.Stop))
                    AND E.Salary > T.Salary;
```

這方法建立輔助表格。我們在這個表格中加上兩個時段中重疊部分的上、下部分。之後我們把所有子查詢中找出的紀錄刪除，因此影響到 NOT EXISTS。最後我們從輔助表格中找出最長的時段，像之前我們找出薪金資訊般。如大家所料，這樣的 SQL 句子執行時效率非常低，因為那些複雜的巢狀不等查詢。

第三個方法是盡量少用 SQL，而在主機上所用的語言中用游標找出最長的歷史。

使用 TSQL2 再次簡單直接。

```
        SELECT MAX(Salary)
        FROM Employee
```

28.16.7 摘要

大部分資料庫應用程式都需要處理隨時間改變的資料。有效時間在 SQL 並不支援。很多時態查詢不是很難以 SQL 模擬，就是需要把 SQL 嵌入到程序語言中，因為 SQL 在資料模型和查詢語言結構中缺乏對有效時間的支援。

另一方面，我們示範出加上有效時間支援需要對 DBMS 施行作出一些改變，但能戲劇性地簡化一些查詢，並使其他查詢變得有可能。將來也可於儲存結構、索引方法的最佳化 (optimization)。而最佳化的策略可導致表現大幅提升。

隨著 SQL3 的一個新部分支援隨時間改變的資料，我們可以開始處理這類應用，改良 SQL3 處理時態資料的能力。

```
----------------------------------------------------------------------------
           Accredited Standards Committee* X3, Information Technology
NEWS RELEASE
```

文件編號：　　`PR/96-0002`

```
回覆：       Barbara Bennett at bbennett@itic.nw.dc.us

             X3 宣佈認可一個這專案, ISO/IEC 9075 Part 7:  SQL/Temporal

Washington D.C., 1996 年一月
-----------------------------------------------------------------------------
```

– Accredited Standards Committee X3, Information Technology 宣佈認可一個有關 SQL/Temporal 支援的新專案，ISO/IEC 9075 Part 7，並把有關工作交予 Technical Committee X3H2, Database。這個建議中的標準訂明將出現的 SQL3 標準的一個新部分，如第七部分 Temporal SQL 為 SQL 的一個延伸，在 SQL 資料庫環境支援存取和處理具時態的資料。下次 X3H2 會議定於一九九六年三月十一至十四日在堪薩斯 (Kansas) 舉行。

如有查詢，可郵寄往：

Chairman of X3H2,
Dr. Donald R. Deutsch,
Sybase, Inc., Suite 800,
6550 Rock Spring
Drive, Bethesda, MD 20817.
Email: deutsch@sybase.com.

現正尋求可能的專利和其他恰當的事項（版權、商標）。這把有關資料寄交：

X3 Secretariat at
1250 Eye Street
NW, Suite 200,
Washington DC 20005.
Email: x3sec@itic.nw.dc.us
FAX: (202)638-4922.

28.17 第八部分—ISO/ANSI SQL 多媒體 (SQL/MM)

一個為建立用於多媒體應用程式的 SQL 物件函式庫的新 ISO/IEC 國際標準化計劃在一九九三年初期獲批準進行。這個新標準化行動名為 SQL Multimedia (SQL/MM)，它會用草擬中的 SQL3 規格中的抽象資料型態（ADT）規格和目標指定 SQL ADT 定義套件。

SQL/MM 目標為標準化科學和工程、全文和文件處理、管理多媒體物件，如影像、聲音、 動畫、音樂和錄像的物件函數庫。它多數會提出一種用於其他 JTC1 標準組織（如有關文件的 SC18、影像的 SC24、照片和動作圖像的 SC29）定義的物件的 SQL 語言結合 (binding)。

SQL/MM 的專案計劃指出它會成為一份多部分標準，包含不定數量的部分。 第一部分會是其他部分如何建成的框架。其他每個部分會專注於一個特定的 SQL 應用程式套件。以下為一九九四年八月存在的 SQL/MM 部分：

- Part 1: Framework A non-technical description of how the document is structured.
- Part 2: Full Text Methods and ADTs for text data processing. About 45 pages.
- Part 3: Spatial Methods and ADTs for spatial data management. About 200 pages with active contributions from Spatial Data experts from 3 national bodies.

- Part 4: General Purpose Methods and ADTs for complex numbers, Facilities include trig and exponential functions, vectors, sets, etc. Currently about 90 pages.

有多個有關空間和地理資料的標準化的工作在進行中：

- ANSI X3L1 - Geographic Information Systems. Unisys 的 Mark Ashworth of Unisys 是 X3L1 和 ANSI X3H2 間的聯絡人。 他也是 SQL/MM 草稿第一、三和四部分的編輯。
- ISO TC 211 - Geographic information/Geomatics

29 PostgreSQL 的技術支援

如果你有任何技術上的問題式遇到困難，你可電郵至：

- *pgsql-questions@postgresql.org*
- 新聞組 `<comp.databases.postgresql.general>`
- 新聞組 `<comp.databases.postgresql.hackers>`
- 新聞組 `<comp.databases.postgresql.doc>`
- 新聞組 `<comp.databases.postgresql.bugs>`
- 新聞組 `<linux.postgres>`
- 其他通信論壇 `<http://www.postgresql.org>`

及預期在一日內收到回覆。因為互聯網產品的用戶範圍很廣，用戶又互相幫助， 互聯網可輕易為數以百萬計用戶提供技術支援。電郵支援比電話支援方便得多， 因為你可剪下和貼上錯誤訊息、程式輸出等等，又可輕易傳送到通信論壇 / 新聞組。

29.1 商業支援

PostgreSQL 組織也會向公司出售技術支援，所得收益會用於維護世界各地的多個映射站台 （萬維網和 ftp）。也會用來印製文件、指引和書籍以幫助顧客。他們的網址為 `<http://www.postresql.org>`。

另一間名為‘Great Bridge Corporation’正進行 PostgreSQL 的開發、銷售和支援。他們的網頁為 `<http://www.greatbridge.com>`。它是一間由‘Landmark Communications corp’和其他投資公司設立的公共公司，專門向世界大企業大公司和售賣和支援 PostgreSQL。

你也可在 Redhat、Anderson、WGS（Work Group Solutions）等專業顧問行得到幫助。 你可找他們幫助，因為他們有很好的 C、C++ 專家（ PostgreSQL 以 C 寫成）。

- Redhat 公司一資料庫顧問部 `<http://www.redhat.com>`
- Work Group Solutions `<http://www.wgs.com>`
- Anderson Consulting `<http://www.ac.com>`

30 經濟和商業方面

商業資料庫繳交很多税項如聯邦、州、銷售、傭用、社會保障、醫療保險税、員工健康保障、其他各程員工福利、市場推廣和廣告費。所有這些費用都不直接用於資料庫的開發，不用於改善資料庫的品質和技術。當你購買一個商業資料庫，一部分的金額浪費於税款、研究開發成本等地方。

而且，商業資料庫需支付租金和購買 Unix 機器，方裝和維護。所有這些成本都轉嫁給用戶。

和商業資料庫相比，PostgreSQL 擁有無需直接税項的優點，因為它是互聯網製造。一大群人為 PostgreSQL 的開發作出貢獻。例如，以一個假設的例子，如果在美國有一百萬間公司，而每間都捐十美元（對 PostgreSQL 來説物超所值），每間便得到一千萬元！！這就是互聯網上軟件開發的魔術。

目前，PostgreSQL 源程式有大約 250,000 行 C、C++ 程式。如果每行 C 程式碼值二美元，今天整個 PostgreSQL 便值五十萬美元！

很多公司已開發大量 C、C++ 程式以作自用。因此，引入 PostgreSQL 源程式碼和與互聯網上其他公司合作會對公司帶來大量利益，節省時間和人力。

31 其他資料庫的名單

以下列出其他 Unix、Linux 的 SQL 資料庫。

- 點選及到達 Applications->databases. `<http://www.caldera.com/tech-ref/linuxapps/linapps.html>`
- 點選及到達 Applications->databases. `<http://www.xnet.com/~blatura/linapps.shtml>`
- 資料庫資源 `<http://linas.org/linux/db.html>` 這是由 Linas Vepstas 寫成的。 *linas@fc.net*
- 自由資料庫名單 `<http://cuiwww.unige.ch:80/~scg/FreeDB/FreeDB.list.html>`
- Browne 的 RDBMS 名單 `<http://www.hex.net/~cbbrowne/rdbms.html>` 由 Christopher B. Browne 寫成。 *cbbrowne@hex.net*
- SAL 關聯式 RDBMS 名單 `<http://SAL.KachinaTech.COM/H/1/>`
- SAL 的物件導向 DBMS 名單 `<http://SAL.KachinaTech.COM/H/2/>`
- SAL's 的工具和其他資料庫名單 `<http://SAL.KachinaTech.COM/H/3/>`
- ACM SIGMOD 公眾可取得的資料庫軟件索引 `<http://bunny.cs.uiuc.edu/sigmod/databaseSoftware/>`

32 萬維網搜尋提示

互聯網非常廣闊，擁有大量軟件和下方有一個資訊海洋。它在全世界以每年百分之三百的速度增長。據估計，全球有大約一千萬個網站！

要搜尋資料，你需要雅虎、網景、Lycos 等搜尋引擎。去到雅虎，點選 search。使用過濾選項 (filtering options) 來收窄你的搜尋要求。預設的搜尋動件是 "Intelligent search"，這比較一般化及列出所有可能性。點選"Options"選擇 "EXACT phrase"搜尋、"AND"搜尋、"OR"搜尋等等。這樣你可

更快找到你想要的訊息。此外，在 search 目錄下，有 radio buttons 用以選擇搜尋新聞組、互聯網站或雅虎網站。

33 結論

在研究過所有自由和可取得源程式碼的資料庫後，發現到只有 PostgreSQL 是世上最成熟、最多人使用和全面的自由 SQL RDMS（物件關聯式）。

PostgreSQL 非常吸引，因為有很多工作已完成。它有 ODBC 和 JDBC 驅動程式，籍著這些，可以編寫獨立於資料庫的應用程式。用 ODBC 或 JDBC 驅動程式寫的 PostgreSQL 應用程式很容易移植到其他資料庫，如 Oracle、Sybase 和 Informix，反之亦然。

你可能會問「為什麼用 PostgreSQL？」答案是因為由零開始， 開發一個資料庫消耗大量時間，選擇一個滿足以下條件的資料庫系統合乎常識：

一個

- 可取得原程式碼—必須是開放程式碼系統
- 沒有附加授權字句 (license string)、沒有擁有權字句（譯按：PostgreSQL 是 GPL，並非無授權協議。如果沒有授權字句，使用它便違法了！）
- 可在互聯網散佈
- 已開發多年
- 滿足 ISO/ANSI SQL 92（和 SQL 89）等標準
- 可滿足未來的需要，如 SQL 3（SQL 98）
- 擁有先進功能

的資料庫系統，而只有 PostgreSQL 滿足所有這些條件，在這情況下是一套適當的軟件。 你可能會說‘PostgreSQL’是一個怪名。但我的看法是—為何改名？這世界會永遠使用 “PostgreSQL”！！

34 FAQ—有關 PostgreSQL 的問題

請參閱一般性、Linux 和 Irix 的最新版 FAQ：

- `<http://www.postgresql.org/docs/faq-english.shtml>`

35 本文件的其他格式

本文件以十一種不同的格式出版——DVI、Postscript、LaTeX、LyX、GNU-info、HTML、RTF （Rich Text Format）、純文字、Unix man pages 和 SGML。

- 你可在 `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/other-formats/>` 取得以 HTML、DVI、Postscript 或 SGML 格式寫成的單一 tar.gz 檔。 （譯按：sunsite.unc.edu 早已改名為 metalab.unc.edu。）

- 純文字檔在： `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO>`
- 其他語言的翻譯，如法文、德文、西班牙文、中文、日文位於 `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO>` 我對翻譯到任何其他語言的幫助無任歡迎。

本文件以一個名為“SGML-Tools”的工具寫成，它可在以下地方找到： `<http://www.sgmltools.org>` 編譯程式後你會得到類似以下的指令：

- sgml2html databasehowto.sgml （產生 HTML 檔）
- sgml2rtf databasehowto.sgml （產生 RTF 檔）
- sgml2latex databasehowto.sgml （產生 LaTeX 檔）

LaTeX 文件可轉換為 PHP 檔，只需使用 **sgml2latex**（和 dvips）產生 Postscript 輸出，及用 Acrobat **distill**（ `<http://www.adobe.com>`）處理該輸出：

```
bash$ man sgml2latex
bash$ sgml2latex filename.sgml
bash$ man dvips
bash$ dvips -o filename.ps filename.dvi
bash$ distill filename.ps
bash$ man ghostscript
bash$ man ps2pdf
bash$ ps2pdf input.ps output.pdf
bash$ acroread output.pdf &
```

你也可用 Ghostscript 指令 **ps2pdf**。ps2pdf 提供了 Adobe 和 Acrobat Distiller 產品的幾乎所有功能：它把 PostScript 檔轉為 Portable Document Format (PDF) 檔。**ps2pdf** 使用了一個細小的命令稿（批次檔），它啓動 Ghostscript，挑選一個名為 **pdfwrite** 的特殊“輸出設備”。要使用 ps2pdf，pdfwrite 設備需在編譯 Ghostscript 時在 makefile 中指定；詳情請參閱 Ghostscript 編譯的文件。

本文件位於：

- `<http://sunsite.unc.edu/LDP/HOWTO/PostgreSQL-HOWTO.html>`

你也可在以下映射站台找到本文件：

- `<http://www.caldera.com/LDP/HOWTO/PostgreSQL-HOWTO.html>`
- `<http://www.WGS.com/LDP/HOWTO/PostgreSQL-HOWTO.html>`
- `<http://www.cc.gatech.edu/linux/LDP/HOWTO/PostgreSQL-HOWTO.html>`
- `<http://www.redhat.com/linux-info/ldp/HOWTO/PostgreSQL-HOWTO.html>`
- 其他在你附近（以網絡位址來說）的站台可在 `<http://sunsite.unc.edu/LDP/hmirrors.html>` 找到，選擇一個網站和到 /LDP/HOWTO/PostgreSQL-HOWTO.html 目錄。

（譯按：本文件的中文〔大五碼〕版可在 `<http://www.linux.org.tw/CLDP/HOWTO/>` 找到。而要產生中文文件，你需要 bg5sgmltools，可在 `<http://www.linux.org.tw/CLDP/>` 找到。）

要觀看 dvi 格式的文件，使用 xdvi 程式。在 Redhat Linux 中，你可透過 ControlPanel | Applications | Publishing | TeX 按鈕找到 tetex-xdvi*.rpm 套件，當中即為 xdvi 程式。

要閱讀 dvi 文件，執行指令：

```
xdvi -geometry 80x90 howto.dvi
man xdvi
```

和使用滑鼠調整視窗大小。瀏覽時使用箭頭鍵、Page Up、Page Down 鍵，你也可用 'f', 'd', 'u', 'c', 'l', 'r', 'p', 'n' 字母來移上、移下、置中、到前後頁等等。 要關掉 expert menu，按 'x'。

你可用程式 'gv' (ghostview) 或 'ghostscript' 來閱讀 postscript 檔。在 RedHat Linux 中，ghostscript 程式是在 ghostscript*.rpm 套件，gv 則在 gv*.rpm。你可透過 ControlPanel | Applications | Graphics 按鈕找到它們。gv 程式比 ghostscript 易用得多。ghostscript 和 gv 也可在其他平台，如 OS/2、視窗 95 及 NT 得到，你也可在這些平台閱讀本文件。

- 在 `<http://www.cs.wisc.edu/~ghost>` 取得 ghostscript 的視窗 95、OS/2 和其他作業系統的版本。

要閱讀 postscript 文件，執行指令—

```
gv howto.ps
ghostscript howto.ps
```

注意：本文件很長，如果你想印出這 postscript 檔，大約需要 113 頁。（譯按：gv 是 ghostscript 的一個前端界面，所以即使你只打算用 gv，你也要安裝後者。此外，在 OS/2 和視窗中並沒有 gv，只有 GSView，一個和 gv 功能差不多的程式。詳情請參關 `<http://www.cs.wisc.edu/~ghost>`。）

你可使用 Netscape Navigator，Microsoft Internet Explorer，Redhat Baron Web Browser 或任何其他的萬維網瀏覽器來閱讀 HTML 格式的文件。

你可用 LyX，LaTeX 的一個 X-Window 前端介面來閱讀 LaTeX 和 LyX 檔。

36 版權和授權

Copyright Al Dev (Alavoor Vasudevan) 1997-2000.

Copyright policy is GNU/GPL as per LDP (Linux Documentation project). LDP is a GNU/GPL project. Additional restrictions are - you must retain the author's name, email address and this copyright notice on all the copies. If you make any changes or additions to this document then you should intimate all the authors of this document.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. In no event shall the author/authors of this document be liable for any damages whatsoever (including without limitation, special, incidental, consequential, or direct/indirect damages for personal injury, loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of this document.

Author/authors offers no warranties or guarantees on fitness, usability, merchantability of this document. Brands, companies and product names mentioned in this document are trademarks or registered trademarks of their respective holders. Please refer to individual copyright notices of brands, companies and products mentioned in this document. It is your responsibility to read and understand the copyright notices of the organisations/companies/products/authors mentioned in this document before using their respective information.

A 附錄甲－ANSI/ISO SQL 1992 語法

這檔案中包括這語言的 BNF 以深度為先的樹狀模掃 (depth-first tree traversal)，大約在 27-AUG-1992 11:03:41.64 完成。這個特定版本所用的 BNF 是只用 ANSI 和 SQL2 的。

```
<SQL terminal character> ::=
      <SQL language character>
    | <SQL embedded language character>

<SQL language character> ::=
      <simple Latin letter>
    | <digit>
    | <SQL special character>

<simple Latin letter> ::=
      <simple Latin upper case letter>
    | <simple Latin lower case letter>

<simple Latin upper case letter> ::=
          A | B | C | D | E | F | G | H | I | J | K | L | M | N | O
    | P | Q | R | S | T | U | V | W | X | Y | Z

<simple Latin lower case letter> ::=
          a | b | c | d | e | f | g | h | i | j | k | l | m | n | o
    | p | q | r | s | t | u | v | w | x | y | z

<digit> ::=
    0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<SQL special character> ::=
      <space>
    | <double quote>
    | <percent>
    | <ampersand>
    | <quote>
    | <left paren>
    | <right paren>
    | <asterisk>
    | <plus sign>
    | <comma>
    | <minus sign>
    | <period>
```

```
    | <solidus>
    | <colon>
    | <semicolon>
    | <less than operator>
    | <equals operator>
    | <greater than operator>
    | <question mark>
    | <underscore>
    | <vertical bar>

<space> ::= !! <EMPHASIS>（使用中的字集所用的空白字元）

<double quote> ::= "

<percent> ::= %

<ampersand> ::= &

<quote> ::= ’

<left paren> ::= (

<right paren> ::= )

<asterisk> ::= *

<plus sign> ::= +

<comma> ::= ,

<minus sign> ::= -

<period> ::= .

<solidus> ::= /

<colon> ::= :

<semicolon> ::= ;

<less than operator> ::= <

<equals operator> ::= =

<greater than operator> ::= >

<question mark> ::= ?

<underscore> ::= _

<vertical bar> ::= |
```

```
<SQL embedded language character> ::=
      <left bracket>
    | <right bracket>

<left bracket> ::= [

<right bracket> ::= ]

<token> ::=
      <nondelimiter token>
    | <delimiter token>

<nondelimiter token> ::=
      <regular identifier>
    | <key word>
    | <unsigned numeric literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>

<regular identifier> ::= <identifier body>

<identifier body> ::=
    <identifier start> [ ( <underscore> | <identifier part> )... ]

<identifier start> ::= <EMPHASIS>（!! 參看文法規則）

<identifier part> ::=
      <identifier start>
    | <digit>

<key word> ::=
      <reserved word>
    | <non-reserved word>

<reserved word> ::=
      ABSOLUTE | ACTION | ADD | ALL
    | ALLOCATE | ALTER | AND
    | ANY | ARE
    | AS | ASC
    | ASSERTION | AT
    | AUTHORIZATION | AVG
    | BEGIN | BETWEEN | BIT | BIT_LENGTH
    | BOTH | BY
    | CASCADE | CASCADED | CASE | CAST
    | CATALOG
    | CHAR | CHARACTER | CHAR_LENGTH
    | CHARACTER_LENGTH | CHECK | CLOSE | COALESCE
    | COLLATE | COLLATION
    | COLUMN | COMMIT
```

```
| CONNECT
| CONNECTION | CONSTRAINT
| CONSTRAINTS | CONTINUE
| CONVERT | CORRESPONDING | COUNT | CREATE | CROSS
| CURRENT
| CURRENT_DATE | CURRENT_TIME
| CURRENT_TIMESTAMP | CURRENT_USER | CURSOR
| DATE | DAY | DEALLOCATE | DEC
| DECIMAL | DECLARE | DEFAULT | DEFERRABLE
| DEFERRED | DELETE | DESC | DESCRIBE | DESCRIPTOR
| DIAGNOSTICS
| DISCONNECT | DISTINCT | DOMAIN | DOUBLE | DROP
| ELSE | END | END-EXEC | ESCAPE
| EXCEPT | EXCEPTION
| EXEC | EXECUTE | EXISTS
| EXTERNAL | EXTRACT
| FALSE | FETCH | FIRST | FLOAT | FOR
| FOREIGN | FOUND | FROM | FULL
| GET | GLOBAL | GO | GOTO
| GRANT | GROUP
| HAVING | HOUR
| IDENTITY | IMMEDIATE | IN | INDICATOR
| INITIALLY | INNER | INPUT
| INSENSITIVE | INSERT | INT | INTEGER | INTERSECT
| INTERVAL | INTO | IS
| ISOLATION
| JOIN
| KEY
| LANGUAGE | LAST | LEADING | LEFT
| LEVEL | LIKE | LOCAL | LOWER
| MATCH | MAX | MIN | MINUTE | MODULE
| MONTH
| NAMES | NATIONAL | NATURAL | NCHAR | NEXT | NO
| NOT | NULL
| NULLIF | NUMERIC
| OCTET_LENGTH | OF
| ON | ONLY | OPEN | OPTION | OR
| ORDER | OUTER
| OUTPUT | OVERLAPS
| PAD | PARTIAL | POSITION | PRECISION | PREPARE
| PRESERVE | PRIMARY
| PRIOR | PRIVILEGES | PROCEDURE | PUBLIC
| READ | REAL | REFERENCES | RELATIVE | RESTRICT
| REVOKE | RIGHT
| ROLLBACK | ROWS
| SCHEMA | SCROLL | SECOND | SECTION
| SELECT
| SESSION | SESSION_USER | SET
| SIZE | SMALLINT | SOME | SPACE | SQL | SQLCODE
| SQLERROR | SQLSTATE
| SUBSTRING | SUM | SYSTEM_USER
```

```
    | TABLE | TEMPORARY
    | THEN | TIME | TIMESTAMP
    | TIMEZONE_HOUR | TIMEZONE_MINUTE
    | TO | TRAILING | TRANSACTION
    | TRANSLATE | TRANSLATION | TRIM | TRUE
    | UNION | UNIQUE | UNKNOWN | UPDATE | UPPER | USAGE
    | USER | USING
    | VALUE | VALUES | VARCHAR | VARYING | VIEW
    | WHEN | WHENEVER | WHERE | WITH | WORK | WRITE
    | YEAR
    | ZONE

<non-reserved word> ::=

      ADA
    | C | CATALOG_NAME
    | CHARACTER_SET_CATALOG | CHARACTER_SET_NAME
    | CHARACTER_SET_SCHEMA | CLASS_ORIGIN | COBOL | COLLATION_CATALOG
    | COLLATION_NAME | COLLATION_SCHEMA | COLUMN_NAME | COMMAND_FUNCTION
    | COMMITTED
    | CONDITION_NUMBER | CONNECTION_NAME | CONSTRAINT_CATALOG | CONSTRAINT_NAME
    | CONSTRAINT_SCHEMA | CURSOR_NAME
    | DATA | DATETIME_INTERVAL_CODE
    | DATETIME_INTERVAL_PRECISION | DYNAMIC_FUNCTION
    | FORTRAN
    | LENGTH
    | MESSAGE_LENGTH | MESSAGE_OCTET_LENGTH | MESSAGE_TEXT | MORE | MUMPS
    | NAME | NULLABLE | NUMBER
    | PASCAL | PLI
    | REPEATABLE | RETURNED_LENGTH | RETURNED_OCTET_LENGTH | RETURNED_SQLSTATE
    | ROW_COUNT
    | SCALE | SCHEMA_NAME | SERIALIZABLE | SERVER_NAME | SUBCLASS_ORIGIN
    | TABLE_NAME | TYPE
    | UNCOMMITTED | UNNAMED

<unsigned numeric literal> ::=
      <exact numeric literal>
    | <approximate numeric literal>

<exact numeric literal> ::=
      <unsigned integer> [ <period> [ <unsigned integer> ] ]
    | <period> <unsigned integer>

<unsigned integer> ::= <digit>...

<approximate numeric literal> ::= <mantissa> E <exponent>

<mantissa> ::= <exact numeric literal>

<exponent> ::= <signed integer>
```

```
<signed integer> ::= [ <sign> ] <unsigned integer>

<sign> ::= <plus sign> | <minus sign>

<national character string literal> ::=
    N <quote> [ <character representation>... ] <quote>
      [ ( <separator>... <quote> [ <character representation>... ] <quote> )... ]

<character representation> ::=
      <nonquote character>
    | <quote symbol>

<nonquote character> ::= !! <EMPHASIS>（!! 參看文法規則）

<quote symbol> ::= <quote><quote>

<separator> ::= ( <comment> | <space> | <newline> )...

<comment> ::=
    <comment introducer> [ <comment character>... ] <newline>

<comment introducer> ::= <minus sign><minus sign>[<minus sign>...]

<comment character> ::=
      <nonquote character>
    | <quote>

<newline> ::= !! <EMPHASIS>（實踐方法定義的行末指示）


<bit string literal> ::=
    B <quote> [ <bit>... ] <quote>
      [ ( <separator>... <quote> [ <bit>... ] <quote> )... ]

<bit> ::= 0 | 1

<hex string literal> ::=
    X <quote> [ <hexit>... ] <quote>
      [ ( <separator>... <quote> [ <hexit>... ] <quote> )... ]

<hexit> ::= <digit> | A | B | C | D | E | F | a | b | c | d | e | f

<delimiter token> ::=
      <character string literal>
    | <date string>
    | <time string>
    | <timestamp string>
    | <interval string>
    | <delimited identifier>
    | <SQL special character>
    | <not equals operator>
```

```
    | <greater than or equals operator>
    | <less than or equals operator>
    | <concatenation operator>
    | <double period>
    | <left bracket>
    | <right bracket>

<character string literal> ::=
    [ <introducer><character set specification> ]
    <quote> [ <character representation>... ] <quote>
      [ ( <separator>... <quote> [ <character representation>... ] <quote> )... ]

<introducer> ::= <underscore>

<character set specification> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <user-defined character repertoire name>
    | <standard universal character form-of-use name>
    | <implementation-defined universal character form-of-use name>

<standard character repertoire name> ::= <character set name>

<character set name> ::= [ <schema name> <period> ]
      <SQL language identifier>

<schema name> ::=
    [ <catalog name> <period> ] <unqualified schema name>

<catalog name> ::= <identifier>

<identifier> ::=
    [ <introducer><character set specification> ] <actual identifier>

<actual identifier> ::=
      <regular identifier>
    | <delimited identifier>

<delimited identifier> ::=
    <double quote> <delimited identifier body> <double quote>

<delimited identifier body> ::= <delimited identifier part>...

<delimited identifier part> ::=
      <nondoublequote character>
    | <doublequote symbol>

<nondoublequote character> ::= <EMPHASIS>（!! 參看文法規則）

<doublequote symbol> ::= <double quote><double quote>
```

```
<unqualified schema name> ::= <identifier>

<SQL language identifier> ::=
    <SQL language identifier start>
       [ ( <underscore> | <SQL language identifier part> )... ]

<SQL language identifier start> ::= <simple Latin letter>

<SQL language identifier part> ::=
      <simple Latin letter>
    | <digit>

<implementation-defined character repertoire name> ::=
    <character set name>

<user-defined character repertoire name> ::= <character set name>

<standard universal character form-of-use name> ::=
    <character set name>

<implementation-defined universal character form-of-use name> ::=
    <character set name>

<date string> ::=
    <quote> <date value> <quote>

<date value> ::=
    <years value> <minus sign> <months value>
        <minus sign> <days value>

<years value> ::= <datetime value>

<datetime value> ::= <unsigned integer>

<months value> ::= <datetime value>

<days value> ::= <datetime value>

<time string> ::=
    <quote> <time value> [ <time zone interval> ] <quote>

<time value> ::=
    <hours value> <colon> <minutes value> <colon> <seconds value>

<hours value> ::= <datetime value>

<minutes value> ::= <datetime value>

<seconds value> ::=
      <seconds integer value> [ <period> [ <seconds fraction> ] ]
```

```
<seconds integer value> ::= <unsigned integer>

<seconds fraction> ::= <unsigned integer>

<time zone interval> ::=
    <sign> <hours value> <colon> <minutes value>

<timestamp string> ::=
    <quote> <date value> <space> <time value>
        [ <time zone interval> ] <quote>

<interval string> ::=
    <quote> ( <year-month literal> | <day-time literal> ) <quote>

<year-month literal> ::=
      <years value>
    | [ <years value> <minus sign> ] <months value>

<day-time literal> ::=
      <day-time interval>
    | <time interval>

<day-time interval> ::=
    <days value>
      [ <space> <hours value> [ <colon> <minutes value>
        [ <colon> <seconds value> ] ] ]

<time interval> ::=
      <hours value> [ <colon> <minutes value> [ <colon> <seconds value> ] ]
    | <minutes value> [ <colon> <seconds value> ]
    | <seconds value>

<not equals operator> ::= <>

<greater than or equals operator> ::= >=

<less than or equals operator> ::= <=

<concatenation operator> ::= ||

<double period> ::= ..

<module> ::=
    <module name clause>
    <language clause>
    <module authorization clause>
    [ <temporary table declaration>... ]
    <module contents>...

<module name clause> ::=
    MODULE [ <module name> ]
```

```
      [ <module character set specification> ]

<module name> ::= <identifier>

<module character set specification> ::=
    NAMES ARE <character set specification>

<language clause> ::=
    LANGUAGE <language name>

<language name> ::=
    ADA | C | COBOL | FORTRAN | MUMPS | PASCAL | PLI

<module authorization clause> ::=
      SCHEMA <schema name>
    | AUTHORIZATION <module authorization identifier>
    | SCHEMA <schema name>
          AUTHORIZATION <module authorization identifier>

<module authorization identifier> ::=
    <authorization identifier>

<authorization identifier> ::= <identifier>

<temporary table declaration> ::=
    DECLARE LOCAL TEMPORARY TABLE
        <qualified local table name>
      <table element list>
      [ ON COMMIT ( PRESERVE | DELETE ) ROWS ]

<qualified local table name> ::=
    MODULE <period> <local table name>

<local table name> ::= <qualified identifier>

<qualified identifier> ::= <identifier>

<table element list> ::=
      <left paren> <table element> [ ( <comma> <table element> )... ] <right paren>

<table element> ::=
      <column definition>
    | <table constraint definition>

<column definition> ::=
    <column name> ( <data type> | <domain name> )
    [ <default clause> ]
    [ <column constraint definition>... ]
    [ <collate clause> ]

<column name> ::= <identifier>
```

```
<data type> ::=
      <character string type>
           [ CHARACTER SET <character set specification> ]
    | <national character string type>
    | <bit string type>
    | <numeric type>
    | <datetime type>
    | <interval type>

<character string type> ::=
      CHARACTER [ <left paren> <length> <right paren> ]
    | CHAR [ <left paren> <length> <right paren> ]
    | CHARACTER VARYING <left paren> <length> <right paren>
    | CHAR VARYING <left paren> <length> <right paren>
    | VARCHAR <left paren> <length> <right paren>

<length> ::= <unsigned integer>

<national character string type> ::=
      NATIONAL CHARACTER [ <left paren> <length> <right paren> ]
    | NATIONAL CHAR [ <left paren> <length> <right paren> ]
    | NCHAR [ <left paren> <length> <right paren> ]
    | NATIONAL CHARACTER VARYING <left paren> <length> <right paren>
    | NATIONAL CHAR VARYING <left paren> <length> <right paren>
    | NCHAR VARYING <left paren> <length> <right paren>

<bit string type> ::=
      BIT [ <left paren> <length> <right paren> ]
    | BIT VARYING <left paren> <length> <right paren>

<numeric type> ::=
      <exact numeric type>
    | <approximate numeric type>

<exact numeric type> ::=
      NUMERIC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DECIMAL [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | DEC [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | INTEGER
    | INT
    | SMALLINT

<precision> ::= <unsigned integer>

<scale> ::= <unsigned integer>

<approximate numeric type> ::=
      FLOAT [ <left paren> <precision> <right paren> ]
    | REAL
    | DOUBLE PRECISION
```

```
<datetime type> ::=
      DATE
    | TIME [ <left paren> <time precision> <right paren> ]
          [ WITH TIME ZONE ]
    | TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]
          [ WITH TIME ZONE ]

<time precision> ::= <time fractional seconds precision>

<time fractional seconds precision> ::= <unsigned integer>

<timestamp precision> ::= <time fractional seconds precision>

<interval type> ::= INTERVAL <interval qualifier>

<interval qualifier> ::=
      <start field> TO <end field>
    | <single datetime field>

<start field> ::=
    <non-second datetime field>
        [ <left paren> <interval leading field precision> <right paren> ]

<non-second datetime field> ::= YEAR | MONTH | DAY | HOUR
    | MINUTE

<interval leading field precision> ::= <unsigned integer>

<end field> ::=
      <non-second datetime field>
    | SECOND [ <left paren> <interval fractional seconds precision> <right paren> ]

<interval fractional seconds precision> ::= <unsigned integer>

<single datetime field> ::=
      <non-second datetime field>
          [ <left paren> <interval leading field precision> <right paren> ]
    | SECOND [ <left paren> <interval leading field precision>
          [ <comma> <interval fractional seconds precision> ] <right paren> ]

<domain name> ::= <qualified name>

<qualified name> ::=
    [ <schema name> <period> ] <qualified identifier>

<default clause> ::=
      DEFAULT <default option>

<default option> ::=
      <literal>
```

```
    | <datetime value function>
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | NULL

<literal> ::=
      <signed numeric literal>
    | <general literal>

<signed numeric literal> ::=
    [ <sign> ] <unsigned numeric literal>

<general literal> ::=
      <character string literal>
    | <national character string literal>
    | <bit string literal>
    | <hex string literal>
    | <datetime literal>
    | <interval literal>

<datetime literal> ::=
      <date literal>
    | <time literal>
    | <timestamp literal>

<date literal> ::=
    DATE <date string>

<time literal> ::=
    TIME <time string>

<timestamp literal> ::=
    TIMESTAMP <timestamp string>

<interval literal> ::=
    INTERVAL [ <sign> ] <interval string> <interval qualifier>

<datetime value function> ::=
      <current date value function>
    | <current time value function>
    | <current timestamp value function>

<current date value function> ::= CURRENT_DATE

<current time value function> ::=
      CURRENT_TIME [ <left paren> <time precision> <right paren> ]

<current timestamp value function> ::=
      CURRENT_TIMESTAMP [ <left paren> <timestamp precision> <right paren> ]
```

```
<column constraint definition> ::=
    [ <constraint name definition> ]
    <column constraint>
      [ <constraint attributes> ]

<constraint name definition> ::= CONSTRAINT <constraint name>

<constraint name> ::= <qualified name>

<column constraint> ::=
      NOT NULL
    | <unique specification>
    | <references specification>
    | <check constraint definition>

<unique specification> ::=
    UNIQUE | PRIMARY KEY

<references specification> ::=
    REFERENCES <referenced table and columns>
      [ MATCH <match type> ]
      [ <referential triggered action> ]

<referenced table and columns> ::=
     <table name> [ <left paren> <reference column list> <right paren> ]

<table name> ::=
      <qualified name>
    | <qualified local table name>

<reference column list> ::= <column name list>

<column name list> ::=
    <column name> [ ( <comma> <column name> )... ]

<match type> ::=
      FULL
    | PARTIAL

<referential triggered action> ::=
      <update rule> [ <delete rule> ]
    | <delete rule> [ <update rule> ]

<update rule> ::= ON UPDATE <referential action>

<referential action> ::=
      CASCADE
    | SET NULL
    | SET DEFAULT
    | NO ACTION
```

```
<delete rule> ::= ON DELETE <referential action>

<check constraint definition> ::=
    CHECK
        <left paren> <search condition> <right paren>

<search condition> ::=
      <boolean term>
    | <search condition> OR <boolean term>

<boolean term> ::=
      <boolean factor>
    | <boolean term> AND <boolean factor>

<boolean factor> ::=
    [ NOT ] <boolean test>

<boolean test> ::=
    <boolean primary> [ IS [ NOT ]
          <truth value> ]

<boolean primary> ::=
      <predicate>
    | <left paren> <search condition> <right paren>

<predicate> ::=
      <comparison predicate>
    | <between predicate>
    | <in predicate>
    | <like predicate>
    | <null predicate>
    | <quantified comparison predicate>
    | <exists predicate>
    | <unique predicate>
    | <match predicate>
    | <overlaps predicate>

<comparison predicate> ::=
    <row value constructor> <comp op>
        <row value constructor>

<row value constructor> ::=
       <row value constructor element>
    | <left paren> <row value constructor list> <right paren>
    | <row subquery>

<row value constructor element> ::=
      <value expression>
    | <null specification>
    | <default specification>
```

```
<value expression> ::=
      <numeric value expression>
    | <string value expression>
    | <datetime value expression>
    | <interval value expression>

<numeric value expression> ::=
      <term>
    | <numeric value expression> <plus sign> <term>
    | <numeric value expression> <minus sign> <term>

<term> ::=
      <factor>
    | <term> <asterisk> <factor>
    | <term> <solidus> <factor>

<factor> ::=
    [ <sign> ] <numeric primary>

<numeric primary> ::=
      <value expression primary>
    | <numeric value function>

<value expression primary> ::=
      <unsigned value specification>
    | <column reference>
    | <set function specification>
    | <scalar subquery>
    | <case expression>
    | <left paren> <value expression> <right paren>
    | <cast specification>

<unsigned value specification> ::=
      <unsigned literal>
    | <general value specification>

<unsigned literal> ::=
      <unsigned numeric literal>
    | <general literal>

<general value specification> ::=
      <parameter specification>
    | <dynamic parameter specification>
    | <variable specification>
    | USER
    | CURRENT_USER
    | SESSION_USER
    | SYSTEM_USER
    | VALUE
```

```
<parameter specification> ::=
    <parameter name> [ <indicator parameter> ]

<parameter name> ::= <colon> <identifier>

<indicator parameter> ::=
    [ INDICATOR ] <parameter name>

<dynamic parameter specification> ::= <question mark>

<variable specification> ::=
    <embedded variable name> [ <indicator variable> ]

<embedded variable name> ::=
    <colon><host identifier>

<host identifier> ::=
      <Ada host identifier>
    | <C host identifier>
    | <COBOL host identifier>
    | <Fortran host identifier>
    | <MUMPS host identifier>
    | <Pascal host identifier>
    | <PL/I host identifier>

<Ada host identifier> ::= !! <EMPHASIS>（參看文法規則）

<C host identifier> ::=
    !! <EMPHASIS>（參看文法規則）

<COBOL host identifier> ::= !! <EMPHASIS>（參看文法規則）

<Fortran host identifier> ::= !! <EMPHASIS>（參看文法規則）

<MUMPS host identifier> ::= !! <EMPHASIS>（參看文法規則）

<Pascal host identifier> ::= !! <EMPHASIS>（參看文法規則）

<PL/I host identifier> ::= !! <EMPHASIS>（參看文法規則）

<indicator variable> ::=
    [ INDICATOR ] <embedded variable name>

<column reference> ::= [ <qualifier> <period> ] <column name>

<qualifier> ::=
      <table name>
    | <correlation name>

<correlation name> ::= <identifier>
```

```
<set function specification> ::=
      COUNT <left paren> <asterisk> <right paren>
    | <general set function>

<general set function> ::=
      <set function type>
          <left paren> [ <set quantifier> ] <value expression> <right paren>

<set function type> ::=
    AVG | MAX | MIN | SUM | COUNT

<set quantifier> ::= DISTINCT | ALL

<scalar subquery> ::= <subquery>

<subquery> ::= <left paren> <query expression> <right paren>

<query expression> ::=
      <non-join query expression>
    | <joined table>

<non-join query expression> ::=
      <non-join query term>
    | <query expression> UNION  [ ALL ]
          [ <corresponding spec> ] <query term>
    | <query expression> EXCEPT [ ALL ]
          [ <corresponding spec> ] <query term>

<non-join query term> ::=
      <non-join query primary>
    | <query term> INTERSECT [ ALL ]
          [ <corresponding spec> ] <query primary>

<non-join query primary> ::=
      <simple table>
    | <left paren> <non-join query expression> <right paren>

<simple table> ::=
      <query specification>
    | <table value constructor>
    | <explicit table>

<query specification> ::=
    SELECT [ <set quantifier> ] <select list> <table expression>

<select list> ::=
      <asterisk>
    | <select sublist> [ ( <comma> <select sublist> )... ]

<select sublist> ::=
      <derived column>
```

```
    | <qualifier> <period> <asterisk>

<derived column> ::= <value expression> [ <as clause> ]

<as clause> ::= [ AS ] <column name>

<table expression> ::=
    <from clause>
    [ <where clause> ]
    [ <group by clause> ]
    [ <having clause> ]

<from clause> ::= FROM <table reference>
    [ ( <comma> <table reference> )... ]

<table reference> ::=
      <table name> [ [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ] ]
    | <derived table> [ AS ] <correlation name>
          [ <left paren> <derived column list> <right paren> ]
    | <joined table>

<derived column list> ::= <column name list>

<derived table> ::= <table subquery>

<table subquery> ::= <subquery>

<joined table> ::=
      <cross join>
    | <qualified join>
    | <left paren> <joined table> <right paren>

<cross join> ::=
    <table reference> CROSS JOIN <table reference>

<qualified join> ::=
    <table reference> [ NATURAL ] [ <join type> ] JOIN
      <table reference> [ <join specification> ]

<join type> ::=
      INNER
    | <outer join type> [ OUTER ]
    | UNION

<outer join type> ::=
      LEFT
    | RIGHT
    | FULL

<join specification> ::=
```

```
      <join condition>
    | <named columns join>

<join condition> ::= ON <search condition>

<named columns join> ::=
    USING <left paren> <join column list> <right paren>

<join column list> ::= <column name list>

<where clause> ::= WHERE <search condition>

<group by clause> ::=
    GROUP BY <grouping column reference list>

<grouping column reference list> ::=
    <grouping column reference>
        [ ( <comma> <grouping column reference> )... ]

<grouping column reference> ::=
    <column reference> [ <collate clause> ]

<collate clause> ::= COLLATE <collation name>

<collation name> ::= <qualified name>

<having clause> ::= HAVING <search condition>

<table value constructor> ::=
    VALUES <table value constructor list>

<table value constructor list> ::=
    <row value constructor> [ ( <comma> <row value constructor> )... ]

<explicit table> ::= TABLE <table name>

<query term> ::=
      <non-join query term>
    | <joined table>

<corresponding spec> ::=
    CORRESPONDING [ BY <left paren> <corresponding column list> <right paren> ]

<corresponding column list> ::= <column name list>

<query primary> ::=
      <non-join query primary>
    | <joined table>

<case expression> ::=
      <case abbreviation>
```

```
    | <case specification>

<case abbreviation> ::=
      NULLIF <left paren> <value expression> <comma>
            <value expression> <right paren>
    | COALESCE <left paren> <value expression>
            ( <comma> <value expression> )... <right paren>

<case specification> ::=
      <simple case>
    | <searched case>

<simple case> ::=
    CASE <case operand>
      <simple when clause>...
      [ <else clause> ]
    END

<case operand> ::= <value expression>

<simple when clause> ::= WHEN <when operand> THEN <result>

<when operand> ::= <value expression>

<result> ::= <result expression> | NULL

<result expression> ::= <value expression>

<else clause> ::= ELSE <result>

<searched case> ::=
    CASE
      <searched when clause>...
      [ <else clause> ]
    END

<searched when clause> ::= WHEN <search condition> THEN <result>

<cast specification> ::=
    CAST <left paren> <cast operand> AS
        <cast target> <right paren>

<cast operand> ::=
      <value expression>
    | NULL

<cast target> ::=
      <domain name>
    | <data type>

<numeric value function> ::=
```

```
      <position expression>
    | <extract expression>
    | <length expression>

<position expression> ::=
    POSITION <left paren> <character value expression>
        IN <character value expression> <right paren>

<character value expression> ::=
      <concatenation>
    | <character factor>

<concatenation> ::=
    <character value expression> <concatenation operator>
        <character factor>

<character factor> ::=
    <character primary> [ <collate clause> ]

<character primary> ::=
      <value expression primary>
    | <string value function>

<string value function> ::=
      <character value function>
    | <bit value function>

<character value function> ::=
      <character substring function>
    | <fold>
    | <form-of-use conversion>
    | <character translation>
    | <trim function>

<character substring function> ::=
    SUBSTRING <left paren> <character value expression> FROM <start position>
                [ FOR <string length> ] <right paren>

<start position> ::= <numeric value expression>

<string length> ::= <numeric value expression>

<fold> ::= ( UPPER | LOWER )
     <left paren> <character value expression> <right paren>

<form-of-use conversion> ::=
    CONVERT <left paren> <character value expression>
        USING <form-of-use conversion name> <right paren>

<form-of-use conversion name> ::= <qualified name>
```

```
<character translation> ::=
    TRANSLATE <left paren> <character value expression>
        USING <translation name> <right paren>

<translation name> ::= <qualified name>

<trim function> ::=
    TRIM <left paren> <trim operands> <right paren>

<trim operands> ::=
    [ [ <trim specification> ] [ <trim character> ] FROM ] <trim source>

<trim specification> ::=
      LEADING
    | TRAILING
    | BOTH

<trim character> ::= <character value expression>

<trim source> ::= <character value expression>

<bit value function> ::=
    <bit substring function>

<bit substring function> ::=
    SUBSTRING <left paren> <bit value expression> FROM <start position>
        [ FOR <string length> ] <right paren>

<bit value expression> ::=
      <bit concatenation>
    | <bit factor>

<bit concatenation> ::=
    <bit value expression> <concatenation operator> <bit factor>

<bit factor> ::= <bit primary>

<bit primary> ::=
      <value expression primary>
    | <string value function>

<extract expression> ::=
    EXTRACT <left paren> <extract field>
        FROM <extract source> <right paren>

<extract field> ::=
      <datetime field>
    | <time zone field>

<datetime field> ::=
      <non-second datetime field>
```

```
    | SECOND

<time zone field> ::=
      TIMEZONE_HOUR
    | TIMEZONE_MINUTE

<extract source> ::=
      <datetime value expression>
    | <interval value expression>

<datetime value expression> ::=
      <datetime term>
    | <interval value expression> <plus sign> <datetime term>
    | <datetime value expression> <plus sign> <interval term>
    | <datetime value expression> <minus sign> <interval term>

<interval term> ::=
      <interval factor>
    | <interval term 2> <asterisk> <factor>
    | <interval term 2> <solidus> <factor>
    | <term> <asterisk> <interval factor>

<interval factor> ::=
    [ <sign> ] <interval primary>

<interval primary> ::=
      <value expression primary> [ <interval qualifier> ]

<interval term 2> ::= <interval term>

<interval value expression> ::=
      <interval term>
    | <interval value expression 1> <plus sign> <interval term 1>
    | <interval value expression 1> <minus sign> <interval term 1>
    | <left paren> <datetime value expression> <minus sign>
          <datetime term> <right paren> <interval qualifier>

<interval value expression 1> ::= <interval value expression>

<interval term 1> ::= <interval term>

<datetime term> ::=
      <datetime factor>

<datetime factor> ::=
      <datetime primary> [ <time zone> ]

<datetime primary> ::=
      <value expression primary>
    | <datetime value function>
```

```
<time zone> ::=
    AT <time zone specifier>

<time zone specifier> ::=
      LOCAL
    | TIME ZONE <interval value expression>

<length expression> ::=
      <char length expression>
    | <octet length expression>
    | <bit length expression>

<char length expression> ::=
    ( CHAR_LENGTH | CHARACTER_LENGTH )
        <left paren> <string value expression> <right paren>

<string value expression> ::=
      <character value expression>
    | <bit value expression>

<octet length expression> ::=
    OCTET_LENGTH <left paren> <string value expression> <right paren>

<bit length expression> ::=
    BIT_LENGTH <left paren> <string value expression> <right paren>

<null specification> ::=
    NULL

<default specification> ::=
    DEFAULT

<row value constructor list> ::=
    <row value constructor element>
        [ ( <comma> <row value constructor element> )... ]

<row subquery> ::= <subquery>

<comp op> ::=
      <equals operator>
    | <not equals operator>
    | <less than operator>
    | <greater than operator>
    | <less than or equals operator>
    | <greater than or equals operator>

<between predicate> ::=
    <row value constructor> [ NOT ] BETWEEN
      <row value constructor> AND <row value constructor>

<in predicate> ::=
```

```
    <row value constructor>
      [ NOT ] IN <in predicate value>

<in predicate value> ::=
      <table subquery>
    | <left paren> <in value list> <right paren>

<in value list> ::=
    <value expression> ( <comma> <value expression> )...

<like predicate> ::=
    <match value> [ NOT ] LIKE <pattern>
      [ ESCAPE <escape character> ]

<match value> ::= <character value expression>

<pattern> ::= <character value expression>

<escape character> ::= <character value expression>

<null predicate> ::= <row value constructor>
    IS [ NOT ] NULL

<quantified comparison predicate> ::=
    <row value constructor> <comp op> <quantifier> <table subquery>

<quantifier> ::= <all> | <some>

<all> ::= ALL

<some> ::= SOME | ANY

<exists predicate> ::= EXISTS <table subquery>

<unique predicate> ::= UNIQUE <table subquery>

<match predicate> ::=
    <row value constructor> MATCH [ UNIQUE ]
        [ PARTIAL | FULL ] <table subquery>

<overlaps predicate> ::=
    <row value constructor 1> OVERLAPS <row value constructor 2>

<row value constructor 1> ::= <row value constructor>

<row value constructor 2> ::= <row value constructor>

<truth value> ::=
      TRUE
    | FALSE
    | UNKNOWN
```

```
<constraint attributes> ::=
      <constraint check time> [ [ NOT ] DEFERRABLE ]
    | [ NOT ] DEFERRABLE [ <constraint check time> ]

<constraint check time> ::=
      INITIALLY DEFERRED
    | INITIALLY IMMEDIATE

<table constraint definition> ::=
    [ <constraint name definition> ]
    <table constraint> [ <constraint attributes> ]

<table constraint> ::=
      <unique constraint definition>
    | <referential constraint definition>
    | <check constraint definition>

<unique constraint definition> ::=
            <unique specification> even in SQL3)
    <unique specification>
      <left paren> <unique column list> <right paren>

<unique column list> ::= <column name list>

<referential constraint definition> ::=
    FOREIGN KEY
        <left paren> <referencing columns> <right paren>
      <references specification>

<referencing columns> ::=
    <reference column list>

<module contents> ::=
      <declare cursor>
    | <dynamic declare cursor>
    | <procedure>

<declare cursor> ::=
    DECLARE <cursor name> [ INSENSITIVE ] [ SCROLL ] CURSOR
      FOR <cursor specification>

<cursor name> ::= <identifier>

<cursor specification> ::=
    <query expression> [ <order by clause> ]
      [ <updatability clause> ]

<order by clause> ::=
    ORDER BY <sort specification list>
```

```
<sort specification list> ::=
    <sort specification> [ ( <comma> <sort specification> )... ]

<sort specification> ::=
    <sort key> [ <collate clause> ] [ <ordering specification> ]

<sort key> ::=
      <column name>
    | <unsigned integer>

<ordering specification> ::= ASC | DESC

<updatability clause> ::=
    FOR
        ( READ ONLY |
          UPDATE [ OF <column name list> ] )

<dynamic declare cursor> ::=
    DECLARE <cursor name> [ INSENSITIVE ] [ SCROLL ] CURSOR
        FOR <statement name>

<statement name> ::= <identifier>

<procedure> ::=
    PROCEDURE <procedure name>
        <parameter declaration list> <semicolon>
      <SQL procedure statement> <semicolon>

<procedure name> ::= <identifier>

<parameter declaration list> ::=
      <left paren> <parameter declaration>
          [ ( <comma> <parameter declaration> )... ] <right paren>
    | <parameter declaration>...

<parameter declaration> ::=
      <parameter name> <data type>
    | <status parameter>

<status parameter> ::=
    SQLCODE | SQLSTATE

<SQL procedure statement> ::=
      <SQL schema statement>
    | <SQL data statement>
    | <SQL transaction statement>
    | <SQL connection statement>
    | <SQL session statement>
    | <SQL dynamic statement>
    | <SQL diagnostics statement>
```

```
<SQL schema statement> ::=
      <SQL schema definition statement>
    | <SQL schema manipulation statement>

<SQL schema definition statement> ::=
      <schema definition>
    | <table definition>
    | <view definition>
    | <grant statement>
    | <domain definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>
    | <assertion definition>

<schema definition> ::=
    CREATE SCHEMA <schema name clause>
      [ <schema character set specification> ]
      [ <schema element>... ]

<schema name clause> ::=
      <schema name>
    | AUTHORIZATION <schema authorization identifier>
    | <schema name> AUTHORIZATION
          <schema authorization identifier>

<schema authorization identifier> ::=
    <authorization identifier>

<schema character set specification> ::=
    DEFAULT CHARACTER
        SET <character set specification>

<schema element> ::=
      <domain definition>
    | <table definition>
    | <view definition>
    | <grant statement>
    | <assertion definition>
    | <character set definition>
    | <collation definition>
    | <translation definition>

<domain definition> ::=
    CREATE DOMAIN <domain name>
        [ AS ] <data type>
      [ <default clause> ]
      [ <domain constraint>... ]
      [ <collate clause> ]

<domain constraint> ::=
```

```
    [ <constraint name definition> ]
    <check constraint definition> [ <constraint attributes> ]

<table definition> ::=
    CREATE [ ( GLOBAL | LOCAL ) TEMPORARY ] TABLE
        <table name>
      <table element list>
      [ ON COMMIT ( DELETE | PRESERVE ) ROWS ]

<view definition> ::=
    CREATE VIEW <table name> [ <left paren> <view column list>
                                  <right paren> ]
      AS <query expression>
      [ WITH [ <levels clause> ] CHECK OPTION ]

<view column list> ::= <column name list>

<levels clause> ::=
    CASCADED | LOCAL

<grant statement> ::=
   GRANT <privileges> ON <object name>
     TO <grantee> [ ( <comma> <grantee> )... ]
       [ WITH GRANT OPTION ]

<privileges> ::=
      ALL PRIVILEGES
    | <action list>

<action list> ::= <action> [ ( <comma> <action> )... ]

<action> ::=
      SELECT
    | DELETE
    | INSERT [ <left paren> <privilege column list> <right paren> ]
    | UPDATE [ <left paren> <privilege column list> <right paren> ]
    | REFERENCES [ <left paren> <privilege column list> <right paren> ]
    | USAGE

<privilege column list> ::= <column name list>

<object name> ::=
      [ TABLE ] <table name>
    | DOMAIN <domain name>
    | COLLATION <collation name>
    | CHARACTER SET <character set name>
    | TRANSLATION <translation name>

<grantee> ::=
      PUBLIC
    | <authorization identifier>
```

```
<assertion definition> ::=
    CREATE ASSERTION <constraint name> <assertion check>
      [ <constraint attributes> ]

<assertion check> ::=
    CHECK
        <left paren> <search condition> <right paren>

<character set definition> ::=
    CREATE CHARACTER SET <character set name>
        [ AS ]
      <character set source>
      [ <collate clause> | <limited collation definition> ]

<character set source> ::=
      GET <existing character set name>

<existing character set name> ::=
      <standard character repertoire name>
    | <implementation-defined character repertoire name>
    | <schema character set name>

<schema character set name> ::= <character set name>

<limited collation definition> ::=
    COLLATION FROM <collation source>

<collation source> ::=
      <collating sequence definition>
    | <translation collation>

<collating sequence definition> ::=
      <external collation>
    | <schema collation name>
    | DESC <left paren> <collation name> <right paren>
    | DEFAULT

<external collation> ::=
    EXTERNAL <left paren> <quote> <external collation name> <quote> <right paren>

<external collation name> ::=
      <standard collation name>
    | <implementation-defined collation name>

<standard collation name> ::= <collation name>

<implementation-defined collation name> ::= <collation name>

<schema collation name> ::= <collation name>
```

```
<translation collation> ::=
    TRANSLATION <translation name>
        [ THEN COLLATION <collation name> ]

<collation definition> ::=
    CREATE COLLATION <collation name> FOR
        <character set specification>
      FROM <collation source>
        [ <pad attribute> ]

<pad attribute> ::=
      NO PAD
    | PAD SPACE

<translation definition> ::=
    CREATE TRANSLATION <translation name>
      FOR <source character set specification>
        TO <target character set specification>
      FROM <translation source>

<source character set specification> ::= <character set specification>

<target character set specification> ::= <character set specification>

<translation source> ::=
      <translation specification>

<translation specification> ::=
      <external translation>
    | IDENTITY
    | <schema translation name>

<external translation> ::=
    EXTERNAL <left paren> <quote> <external translation name> <quote> <right paren>

<external translation name> ::=
      <standard translation name>
    | <implementation-defined translation name>

<standard translation name> ::= <translation name>

<implementation-defined translation name> ::= <translation name>

<schema translation name> ::= <translation name>

<SQL schema manipulation statement> ::=
      <drop schema statement>
    | <alter table statement>
    | <drop table statement>
    | <drop view statement>
    | <revoke statement>
```

```
    | <alter domain statement>
    | <drop domain statement>
    | <drop character set statement>
    | <drop collation statement>
    | <drop translation statement>
    | <drop assertion statement>

<drop schema statement> ::=
    DROP SCHEMA <schema name> <drop behavior>

<drop behavior> ::= CASCADE | RESTRICT

<alter table statement> ::=
    ALTER TABLE <table name> <alter table action>

<alter table action> ::=
      <add column definition>
    | <alter column definition>
    | <drop column definition>
    | <add table constraint definition>
    | <drop table constraint definition>

<add column definition> ::=
    ADD [ COLUMN ] <column definition>

<alter column definition> ::=
    ALTER [ COLUMN ] <column name> <alter column action>

<alter column action> ::=
      <set column default clause>
    | <drop column default clause>

<set column default clause> ::=
    SET <default clause>

<drop column default clause> ::=
    DROP DEFAULT

<drop column definition> ::=
    DROP [ COLUMN ] <column name> <drop behavior>

<add table constraint definition> ::=
    ADD <table constraint definition>

<drop table constraint definition> ::=
    DROP CONSTRAINT <constraint name> <drop behavior>

<drop table statement> ::=
    DROP TABLE <table name> <drop behavior>

<drop view statement> ::=
```

```
    DROP VIEW <table name> <drop behavior>

<revoke statement> ::=
    REVOKE [ GRANT OPTION FOR ]
        <privileges>
        ON <object name>
      FROM <grantee> [ ( <comma> <grantee> )... ] <drop behavior>

<alter domain statement> ::=
    ALTER DOMAIN <domain name> <alter domain action>

<alter domain action> ::=
      <set domain default clause>
    | <drop domain default clause>
    | <add domain constraint definition>
    | <drop domain constraint definition>

<set domain default clause> ::= SET <default clause>

<drop domain default clause> ::= DROP DEFAULT

<add domain constraint definition> ::=
    ADD <domain constraint>

<drop domain constraint definition> ::=
    DROP CONSTRAINT <constraint name>

<drop domain statement> ::=
    DROP DOMAIN <domain name> <drop behavior>

<drop character set statement> ::=
    DROP CHARACTER SET <character set name>

<drop collation statement> ::=
    DROP COLLATION <collation name>

<drop translation statement> ::=
    DROP TRANSLATION <translation name>

<drop assertion statement> ::=
    DROP ASSERTION <constraint name>

<SQL data statement> ::=
      <open statement>
    | <fetch statement>
    | <close statement>
    | <select statement: single row>
    | <SQL data change statement>

<open statement> ::=
    OPEN <cursor name>
```

```
<fetch statement> ::=
    FETCH [ [ <fetch orientation> ] FROM ]
      <cursor name> INTO <fetch target list>

<fetch orientation> ::=
      NEXT
    | PRIOR
    | FIRST
    | LAST
    | ( ABSOLUTE | RELATIVE ) <simple value specification>

<simple value specification> ::=
      <parameter name>
    | <embedded variable name>
    | <literal>

<fetch target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<target specification> ::=
      <parameter specification>
    | <variable specification>

<close statement> ::=
    CLOSE <cursor name>

<select statement: single row> ::=
    SELECT [ <set quantifier> ] <select list>
      INTO <select target list>
        <table expression>

<select target list> ::=
    <target specification> [ ( <comma> <target specification> )... ]

<SQL data change statement> ::=
      <delete statement: positioned>
    | <delete statement: searched>
    | <insert statement>
    | <update statement: positioned>
    | <update statement: searched>

<delete statement: positioned> ::=
    DELETE FROM <table name>
      WHERE CURRENT OF <cursor name>

<delete statement: searched> ::=
    DELETE FROM <table name>
      [ WHERE <search condition> ]

<insert statement> ::=
```

```
    INSERT INTO <table name>
      <insert columns and source>

<insert columns and source> ::=
      [ <left paren> <insert column list> <right paren> ]
            <query expression>
    | DEFAULT VALUES

<insert column list> ::= <column name list>

<update statement: positioned> ::=
    UPDATE <table name>
      SET <set clause list>
        WHERE CURRENT OF <cursor name>

<set clause list> ::=
    <set clause> [ ( <comma> <set clause> )... ]

<set clause> ::=
    <object column> <equals operator> <update source>

<object column> ::= <column name>

<update source> ::=
      <value expression>
    | <null specification>
    | DEFAULT

<update statement: searched> ::=
    UPDATE <table name>
      SET <set clause list>
      [ WHERE <search condition> ]

<SQL transaction statement> ::=
      <set transaction statement>
    | <set constraints mode statement>
    | <commit statement>
    | <rollback statement>

<set transaction statement> ::=
    SET TRANSACTION <transaction mode>
        [ ( <comma> <transaction mode> )... ]

<transaction mode> ::=
      <isolation level>
    | <transaction access mode>
    | <diagnostics size>

<isolation level> ::=
    ISOLATION LEVEL <level of isolation>
```

```
<level of isolation> ::=
      READ UNCOMMITTED
    | READ COMMITTED
    | REPEATABLE READ
    | SERIALIZABLE

<transaction access mode> ::=
      READ ONLY
    | READ WRITE

<diagnostics size> ::=
    DIAGNOSTICS SIZE <number of conditions>

<number of conditions> ::= <simple value specification>

<set constraints mode statement> ::=
    SET CONSTRAINTS <constraint name list>
        ( DEFERRED | IMMEDIATE )

<constraint name list> ::=
      ALL
    | <constraint name> [ ( <comma> <constraint name> )... ]

<commit statement> ::=
    COMMIT [ WORK ]

<rollback statement> ::=
    ROLLBACK [ WORK ]

<SQL connection statement> ::=
      <connect statement>
    | <set connection statement>
    | <disconnect statement>

<connect statement> ::=
    CONNECT TO <connection target>

<connection target> ::=
      <SQL-server name>
        [ AS <connection name> ]
          correspondence with Tony Gordon)
        [ USER <user name> ]
    | DEFAULT

<SQL-server name> ::= <simple value specification>

<connection name> ::= <simple value specification>

<user name> ::= <simple value specification>

<set connection statement> ::=
```

```
    SET CONNECTION <connection object>

<connection object> ::=
      DEFAULT
    | <connection name>

<disconnect statement> ::=
    DISCONNECT <disconnect object>

<disconnect object> ::=
      <connection object>
    | ALL
    | CURRENT

<SQL session statement> ::=
      <set catalog statement>
    | <set schema statement>
    | <set names statement>
    | <set session authorization identifier statement>
    | <set local time zone statement>

<set catalog statement> ::=
    SET CATALOG <value specification>

<value specification> ::=
      <literal>
    | <general value specification>

<set schema statement> ::=
    SET SCHEMA <value specification>

<set names statement> ::=
    SET NAMES <value specification>

<set session authorization identifier statement> ::=
    SET SESSION AUTHORIZATION
        <value specification>

<set local time zone statement> ::=
    SET TIME ZONE
        <set time zone value>

<set time zone value> ::=
      <interval value expression>
    | LOCAL

<SQL dynamic statement> ::=
      <system descriptor statement>
    | <prepare statement>
    | <deallocate prepared statement>
    | <describe statement>
```

```
    | <execute statement>
    | <execute immediate statement>
    | <SQL dynamic data statement>

<system descriptor statement> ::=
      <allocate descriptor statement>
    | <deallocate descriptor statement>
    | <set descriptor statement>
    | <get descriptor statement>

<allocate descriptor statement> ::=
    ALLOCATE DESCRIPTOR <descriptor name>
       [ WITH MAX <occurrences> ]

<descriptor name> ::=
    [ <scope option> ] <simple value specification>

<scope option> ::=
      GLOBAL
    | LOCAL

<occurrences> ::= <simple value specification>

<deallocate descriptor statement> ::=
    DEALLOCATE DESCRIPTOR <descriptor name>

<set descriptor statement> ::=
    SET DESCRIPTOR <descriptor name>
        <set descriptor information>

<set descriptor information> ::=
      <set count>
    | VALUE <item number>
        <set item information> [ ( <comma> <set item information> )... ]

<set count> ::=
    COUNT <equals operator> <simple value specification 1>

<simple value specification 1> ::= <simple value specification>

<item number> ::= <simple value specification>

<set item information> ::=
    <descriptor item name> <equals operator> <simple value specification 2>

<descriptor item name> ::=
      TYPE
    | LENGTH
    | OCTET_LENGTH
    | RETURNED_LENGTH
    | RETURNED_OCTET_LENGTH
```

```
    | PRECISION
    | SCALE
    | DATETIME_INTERVAL_CODE
    | DATETIME_INTERVAL_PRECISION
    | NULLABLE
    | INDICATOR
    | DATA
    | NAME
    | UNNAMED
    | COLLATION_CATALOG
    | COLLATION_SCHEMA
    | COLLATION_NAME
    | CHARACTER_SET_CATALOG
    | CHARACTER_SET_SCHEMA
    | CHARACTER_SET_NAME

<simple value specification 2> ::= <simple value specification>

<item number> ::= <simple value specification>

<get descriptor statement> ::=
    GET DESCRIPTOR <descriptor name> <get descriptor information>

<get descriptor information> ::=
      <get count>
    | VALUE <item number>
        <get item information> [ ( <comma> <get item information> )... ]

<get count> ::=
    <simple target specification 1> <equals operator>
         COUNT

<simple target specification 1> ::= <simple target specification>

<simple target specification> ::=
      <parameter name>
    | <embedded variable name>

<get item information> ::=
    <simple target specification 2> <equals operator> <descriptor item name>>

<simple target specification 2> ::= <simple target specification>

<prepare statement> ::=
    PREPARE <SQL statement name> FROM <SQL statement variable>

<SQL statement name> ::=
      <statement name>
    | <extended statement name>

<extended statement name> ::=
```

```
    [ <scope option> ] <simple value specification>

<SQL statement variable> ::= <simple value specification>

<deallocate prepared statement> ::=
    DEALLOCATE PREPARE <SQL statement name>

<describe statement> ::=
      <describe input statement>
    | <describe output statement>

<describe input statement> ::=
    DESCRIBE INPUT <SQL statement name> <using descriptor>

<using descriptor> ::=
    ( USING | INTO ) SQL DESCRIPTOR <descriptor name>

<describe output statement> ::=
    DESCRIBE [ OUTPUT ] <SQL statement name> <using descriptor>

<execute statement> ::=
    EXECUTE <SQL statement name>
      [ <result using clause> ]
      [ <parameter using clause> ]

<result using clause> ::= <using clause>

<using clause> ::=
      <using arguments>
    | <using descriptor>

<using arguments> ::=
    ( USING | INTO ) <argument> [ ( <comma> <argument> )... ]

<argument> ::= <target specification>

<parameter using clause> ::= <using clause>

<execute immediate statement> ::=
    EXECUTE IMMEDIATE <SQL statement variable>

<SQL dynamic data statement> ::=
      <allocate cursor statement>
    | <dynamic open statement>
    | <dynamic fetch statement>
    | <dynamic close statement>
    | <dynamic delete statement: positioned>
    | <dynamic update statement: positioned>

<allocate cursor statement> ::=
    ALLOCATE <extended cursor name> [ INSENSITIVE ]
```

```
        [ SCROLL ] CURSOR
      FOR <extended statement name>

<extended cursor name> ::=
    [ <scope option> ] <simple value specification>

<dynamic open statement> ::=
    OPEN <dynamic cursor name> [ <using clause> ]

<dynamic cursor name> ::=
      <cursor name>
    | <extended cursor name>

<dynamic fetch statement> ::=
    FETCH [ [ <fetch orientation> ] FROM ] <dynamic cursor name>
        <using clause>

<dynamic close statement> ::=
    CLOSE <dynamic cursor name>

<dynamic delete statement: positioned> ::=
    DELETE FROM <table name>
      WHERE CURRENT OF
          <dynamic cursor name>

<dynamic update statement: positioned> ::=
    UPDATE <table name>
      SET <set clause>
          [ ( <comma> <set clause> )... ]
        WHERE CURRENT OF
            <dynamic cursor name>

<SQL diagnostics statement> ::=
    <get diagnostics statement>

<get diagnostics statement> ::=
    GET DIAGNOSTICS <sql diagnostics information>

<sql diagnostics information> ::=
      <statement information>
    | <condition information>

<statement information> ::=
    <statement information item> [ ( <comma> <statement information item> )... ]

<statement information item> ::=
    <simple target specification> <equals operator> <statement information item name>

<statement information item name> ::=
      NUMBER
    | MORE
```

```
    | COMMAND_FUNCTION
    | DYNAMIC_FUNCTION
    | ROW_COUNT

<condition information> ::=
    EXCEPTION <condition number>
      <condition information item> [ ( <comma> <condition information item> )... ]

<condition number> ::= <simple value specification>

<condition information item> ::=
    <simple target specification> <equals operator> <condition information item name>

<condition information item name> ::=
      CONDITION_NUMBER
    | RETURNED_SQLSTATE
    | CLASS_ORIGIN
    | SUBCLASS_ORIGIN
    | SERVER_NAME
    | CONNECTION_NAME
    | CONSTRAINT_CATALOG
    | CONSTRAINT_SCHEMA
    | CONSTRAINT_NAME
    | CATALOG_NAME
    | SCHEMA_NAME
    | TABLE_NAME
    | COLUMN_NAME
    | CURSOR_NAME
    | MESSAGE_TEXT
    | MESSAGE_LENGTH
    | MESSAGE_OCTET_LENGTH

<embedded SQL host program> ::=
      <embedded SQL Ada program>
    | <embedded SQL C program>
    | <embedded SQL COBOL program>
    | <embedded SQL Fortran program>
    | <embedded SQL MUMPS program>
    | <embedded SQL Pascal program>
    | <embedded SQL PL/I program>

<embedded SQL Ada program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL C program> ::=
      !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL COBOL program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL Fortran program> ::=
    !! <EMPHASIS>(See the Syntax Rules.)
```

```
<embedded SQL MUMPS program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL Pascal program> ::=
    !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL PL/I program> ::= !! <EMPHASIS>(See the Syntax Rules.)

<embedded SQL declare section> ::=
      <embedded SQL begin declare>
        [ <embedded character set declaration> ]
        [ <host variable definition>... ]
      <embedded SQL end declare>
    | <embedded SQL MUMPS declare>

<embedded SQL begin declare> ::=
    <SQL prefix> BEGIN DECLARE SECTION
        [ <SQL terminator> ]

<SQL prefix> ::=
      EXEC SQL
    | <ampersand>SQL<left paren>

<SQL terminator> ::=
      END-EXEC
    | <semicolon>
    | <right paren>

<embedded character set declaration> ::=
    SQL NAMES ARE <character set specification>

<host variable definition> ::=
      <Ada variable definition>
    | <C variable definition>
    | <COBOL variable definition>
    | <Fortran variable definition>
    | <MUMPS variable definition>
    | <Pascal variable definition>
    | <PL/I variable definition>

<Ada variable definition> ::=
    <Ada host identifier> [ ( <comma> <Ada host identifier> )... ] :
    <Ada type specification> [ <Ada initial value> ]

<Ada type specification> ::=
      <Ada qualified type specification>
    | <Ada unqualified type specification>

<Ada qualified type specification> ::=
      SQL_STANDARD.CHAR [ CHARACTER SET
         [ IS ] <character set specification> ]
          <left paren> 1 <double period> <length> <right paren>
```

```
    | SQL_STANDARD.BIT
          <left paren> 1 <double period> <length> <right paren>
    | SQL_STANDARD.SMALLINT
    | SQL_STANDARD.INT
    | SQL_STANDARD.REAL
    | SQL_STANDARD.DOUBLE_PRECISION
    | SQL_STANDARD.SQLCODE_TYPE
    | SQL_STANDARD.SQLSTATE_TYPE
    | SQL_STANDARD.INDICATOR_TYPE

<Ada unqualified type specification> ::=
      CHAR
          <left paren> 1 <double period> <length> <right paren>
    | BIT
          <left paren> 1 <double period> <length> <right paren>
    | SMALLINT
    | INT
    | REAL
    | DOUBLE_PRECISION
    | SQLCODE_TYPE
    | SQLSTATE_TYPE
    | INDICATOR_TYPE

<Ada initial value> ::=
    <Ada assignment operator> <character representation>...

<Ada assignment operator> ::= <colon><equals operator>

<C variable definition> ::=
      [ <C storage class> ]
      [ <C class modifier> ]
      <C variable specification>
    <semicolon>

<C storage class> ::=
      auto
    | extern
    | static

<C class modifier> ::= const | volatile

<C variable specification> ::=
      <C numeric variable>
    | <C character variable>
    | <C derived variable>

<C numeric variable> ::=
    ( long | short | float | double )
      <C host identifier> [ <C initial value> ]
            [ ( <comma> <C host identifier> [ <C initial value> ] )... ]
```

```
<C initial value> ::=
    <equals operator> <character representation>...

<C character variable> ::=
    char [ CHARACTER SET
             [ IS ] <character set specification> ]
      <C host identifier>
        <C array specification> [ <C initial value> ]
        [ ( <comma> <C host identifier>
          <C array specification>
                 [ <C initial value> ] )... ]

<C array specification> ::=
    <left bracket> <length> <right bracket>

<C derived variable> ::=
      <C VARCHAR variable>
    | <C bit variable>

<C VARCHAR variable> ::=
    VARCHAR [ CHARACTER SET [ IS ]
        <character set specification> ]
        <C host identifier>
            <C array specification> [ <C initial value> ]
          [ ( <comma> <C host identifier>
              <C array specification>
                      [ <C initial value> ] )... ]

<C bit variable> ::=
    BIT <C host identifier>
        <C array specification> [ <C initial value> ]
      [ ( <comma> <C host identifier>
        <C array specification>
                   [ <C initial value> ] )... ]

<COBOL variable definition> ::=
    (01|77) <COBOL host identifier> <COBOL type specification>
      [ <character representation>... ] <period>

<COBOL type specification> ::=
      <COBOL character type>
    | <COBOL bit type>
    | <COBOL numeric type>
    | <COBOL integer type>

<COBOL character type> ::=
    [ CHARACTER SET [ IS ]
          <character set specification> ]
    ( PIC | PICTURE ) [ IS ] ( X [ <left paren> <length> <right paren> ] )...

<COBOL bit type> ::=
```

```
    ( PIC | PICTURE ) [ IS ]
        ( B [ <left paren> <length> <right paren> ] )...

<COBOL numeric type> ::=
    ( PIC | PICTURE ) [ IS ]
      S <COBOL nines specification>
    [ USAGE [ IS ] ] DISPLAY SIGN LEADING SEPARATE

<COBOL nines specification> ::=
      <COBOL nines> [ V [ <COBOL nines> ] ]
    | V <COBOL nines>

<COBOL nines> ::= ( 9 [ <left paren> <length> <right paren> ] )...

<COBOL integer type> ::=
      <COBOL computational integer>
    | <COBOL binary integer>

<COBOL computational integer> ::=
    ( PIC | PICTURE ) [ IS ] S<COBOL nines>
      [ USAGE [ IS ] ] ( COMP | COMPUTATIONAL )

<COBOL binary integer> ::=
    ( PIC | PICTURE ) [ IS ] S<COBOL nines>
      [ USAGE [ IS ] ] BINARY

<Fortran variable definition> ::=
    <Fortran type specification>
    <Fortran host identifier>
        [ ( <comma> <Fortran host identifier> )... ]

<Fortran type specification> ::=
      CHARACTER [ <asterisk> <length> ]
          [ CHARACTER SET [ IS ]
                <character set specification> ]
    | BIT [ <asterisk> <length> ]
    | INTEGER
    | REAL
    | DOUBLE PRECISION

<MUMPS variable definition> ::=
    ( <MUMPS numeric variable> | <MUMPS character variable> )
        <semicolon>

<MUMPS numeric variable> ::=
    <MUMPS type specification>
      <MUMPS host identifier> [ ( <comma> <MUMPS host identifier> )... ]

<MUMPS type specification> ::=
      INT
    | DEC
```

```
          [ <left paren> <precision> [ <comma> <scale> ] <right paren> ]
    | REAL

<MUMPS character variable> ::=
    VARCHAR <MUMPS host identifier> <MUMPS length specification>
      [ ( <comma> <MUMPS host identifier> <MUMPS length specification> )... ]

<MUMPS length specification> ::=
    <left paren> <length> <right paren>

<Pascal variable definition> ::=
    <Pascal host identifier> [ ( <comma> <Pascal host identifier> )... ] <colon>
      <Pascal type specification> <semicolon>

<Pascal type specification> ::=
      PACKED ARRAY
          <left bracket> 1 <double period> <length> <right bracket>
        OF CHAR
          [ CHARACTER SET [ IS ]
                <character set specification> ]
    | PACKED ARRAY
          <left bracket> 1 <double period> <length> <right bracket>
        OF BIT
    | INTEGER
    | REAL
    | CHAR [ CHARACTER SET
                              [ IS ] <character set specification> ]
    | BIT

<PL/I variable definition> ::=
    (DCL | DECLARE)
        (   <PL/I host identifier>
          | <left paren> <PL/I host identifier>
                [ ( <comma> <PL/I host identifier> )... ] <right paren> )
    <PL/I type specification>
    [ <character representation>... ] <semicolon>

<PL/I type specification> ::=
      ( CHAR | CHARACTER ) [ VARYING ]
          <left paren><length><right paren>
          [ CHARACTER SET
                [ IS ] <character set specification> ]
    | BIT [ VARYING ] <left paren><length><right paren>
    | <PL/I type fixed decimal> <left paren> <precision>
          [ <comma> <scale> ] <right paren>
    | <PL/I type fixed binary> [ <left paren> <precision> <right paren> ]
    | <PL/I type float binary> <left paren> <precision> <right paren>

<PL/I type fixed decimal> ::=
      ( DEC | DECIMAL ) FIXED
    | FIXED ( DEC | DECIMAL )
```

```
<PL/I type fixed binary> ::=
      ( BIN | BINARY ) FIXED
    | FIXED ( BIN | BINARY )

<PL/I type float binary> ::=
      ( BIN | BINARY ) FLOAT
    | FLOAT ( BIN | BINARY )

<embedded SQL end declare> ::=
    <SQL prefix> END DECLARE SECTION
        [ <SQL terminator> ]

<embedded SQL MUMPS declare> ::=
    <SQL prefix>
      BEGIN DECLARE SECTION
        [ <embedded character set declaration> ]
        [ <host variable definition>... ]
      END DECLARE SECTION
    <SQL terminator>

<embedded SQL statement> ::=
    <SQL prefix>
      <statement or declaration>
    [ <SQL terminator> ]

<statement or declaration> ::=
      <declare cursor>
    | <dynamic declare cursor>
    | <temporary table declaration>
    | <embedded exception declaration>
    | <SQL procedure statement>

<embedded exception declaration> ::=
    WHENEVER <condition> <condition action>

<condition> ::=
    SQLERROR | NOT FOUND

<condition action> ::=
    CONTINUE | <go to>

<go to> ::=
    ( GOTO | GO TO ) <goto target>

<goto target> ::=
      <host label identifier>
    | <unsigned integer>
    | <host PL/I label variable>

<host label identifier> ::= !!<EMPHASIS>（參看文法規則）
```

```
<host PL/I label variable> ::= !!<EMPHASIS>（參看文法規則）

<preparable statement> ::=
      <preparable SQL data statement>
    | <preparable SQL schema statement>
    | <preparable SQL transaction statement>
    | <preparable SQL session statement>
    | <preparable implementation-defined statement>

<preparable SQL data statement> ::=
      <delete statement: searched>
    | <dynamic single row select statement>
    | <insert statement>
    | <dynamic select statement>
    | <update statement: searched>
    | <preparable dynamic delete statement: positioned>
    | <preparable dynamic update statement: positioned>

<dynamic single row select statement> ::= <query specification>

<dynamic select statement> ::= <cursor specification>

<preparable dynamic delete statement: positioned> ::=
   DELETE [ FROM <table name> ]
      WHERE CURRENT OF <cursor name>

<preparable dynamic update statement: positioned> ::=
   UPDATE [ <table name> ]
      SET <set clause list>
      WHERE CURRENT OF <cursor name>

<preparable SQL schema statement> ::=
      <SQL schema statement>

<preparable SQL transaction statement> ::=
      <SQL transaction statement>

<preparable SQL session statement> ::=
      <SQL session statement>

<preparable implementation-defined statement> ::=
    !! <EMPHASIS>(See the Syntax Rules.)

<direct SQL statement> ::=
    <directly executable statement> <semicolon>

<directly executable statement> ::=
      <direct SQL data statement>
    | <SQL schema statement>
    | <SQL transaction statement>
```

```
    | <SQL connection statement>
    | <SQL session statement>
    | <direct implementation-defined statement>

<direct SQL data statement> ::=
      <delete statement: searched>
    | <direct select statement: multiple rows>
    | <insert statement>
    | <update statement: searched>
    | <temporary table declaration>

<direct select statement: multiple rows> ::=
    <query expression> [ <order by clause> ]

<direct implementation-defined statement> ::=
    !!<EMPHASIS>(See the Syntax Rules)

<SQL object identifier> ::=
    <SQL provenance> <SQL variant>

<SQL provenance> ::= <arc1> <arc2> <arc3>

<arc1> ::= iso | 1 | iso <left paren> 1 <right paren>

<arc2> ::= standard | 0 | standard <left paren> 0 <right paren>

<arc3> ::= 9075

<SQL variant> ::= <SQL edition> <SQL conformance>

<SQL edition> ::= <1987> | <1989> | <1992>

<1987> ::= 0 | edition1987 <left paren> 0 <right paren>

<1989> ::= <1989 base> <1989 package>

<1989 base> ::= 1 | edition1989 <left paren> 1 <right paren>

<1989 package> ::= <integrity no> | <integrity yes>

<integrity no> ::= 0 | IntegrityNo <left paren> 0 <right paren>

<integrity yes> ::= 1 | IntegrityYes <left paren> 1 <right paren>

<1992> ::= 2 | edition1992 <left paren> 2 <right paren>

<SQL conformance> ::= <low> | <intermediate> | <high>

<low> ::= 0 | Low <left paren> 0 <right paren>

<intermediate> ::= 1 | Intermediate <left paren> 1 <right paren>
```

```
<high> ::= 2 | High <left paren> 2 <right paren>
```

B 附錄乙—初學者 SQL 導引

B.1 PostgreSQL 導引

SQL 導引也包含在 PostgreSQL 中散佈。SQL 導引手稿在 src/tutorial 目錄中。

B.2 互聯網 URL 指標

初學者的 SQL 導引可在以下地方找到

- Jim Hoffman 的指導 `<http://w3.one.net/~jhoffman/sqltut.htm>`
- Carnegie Mellon Univ `<http://www.heinz.cmu.edu/project/dbms>` 在此擊選‘technical’-> ‘SQL_examples.html’和其他。
- Concord Univ `<http://www.cs.concordia.ca/Course_Notes/oracle/browser/node1.html>`

有意見或疑難？電郵至

- Jim Hoffman *jhoffman@one.net*

John Hoffman 也推薦以下網址：

- SQL Reference `<http://www.contrib.andrew.cmu.edu/~shadow/sql.html>`
- Ask the SQL Pro `<http://www.inquiry.com/techtips/thesqlpro/>`
- SQL Pro's Relational DB Useful Sites `<http://www.inquiry.com/techtips/thesqlpro/usefulsites.html>`
- Programmer's Source `<http://infoweb.magi.com/~steve/develop.html>`
- DBMS Sites `<http://info.itu.ch/special/wwwfiles>` 到此取得 compdb.html 檔
- DB Ingredients `<http://www.compapp.dcu.ie/databases/f017.html>`
- Web Authoring `<http://www.stars.com/Tutorial/CGI/>`
- Computing Dictionary `<http://wfn-shop.princeton.edu/cgi-bin/foldoc>`
- DBMS Lab/Links `<http://www-ccs.cs.umass.edu/db.html>`
- SQL FAQ `<http://epoch.CS.Berkeley.EDU:8000/sequoia/dba/montage/FAQ>` Go here and see file SQL_TOC.html
- SQL Databases `<http://chaos.mur.csu.edu.au/itc125/cgi/sqldb.html>`
- RIT Database Design Page `<http://www.it.rit.edu/~wjs/IT/199602/icsa720/icsa720postings.html>`

- Database Jump Site `<http://www.pcslink.com/~ej/dbweb.html>`
- Programming Tutorials on the Web `<http://www.eng.uc.edu/~jtilley/tutorial.html>`
- Development Resources `<http://www.ndev.com/ndc2/support/resources.htp>`
- Query List `<http://ashok.pair.com/sql.htm>`
- IMAGE SQL Miscellaneous `<http://jazz.external.hp.com/training/sqltables/main.html>`
- Internet Resource List `<http://www.eit.com/web/netservices.html>`

B.3 線上 SQL 指導

到以下網頁取得線上 SQL 指導

- SQL 初學課程 `<http://sqlcourse.com>`
- SQL 進階課程 `<http://sqlcourse2.com>`

C 附錄丙—Linux 快速安裝指引

如果你計劃在 Linux 使用 PostgreSQL，而在安裝 Linux 時需要幫助，請參考在這附錄中指示的地方。它們包含以下主題：

- Salient Features of Linux - Why Linux is better as a database server when compared with Windows 95/NT
- 10 minutes Linux Quick Install Instructions
- Microsoft-Linux Analogy List
- Quick Steps to Recompile the Linux Kernel

- 主網頁 `<http://www.aldev.8m.com>`
- 映射站台 `<http://aldev.webjump.com>`
- 映射站台 `<http://www.homepages.infoseek.com/~aldev1/index.html>`
- 映射站台 `<http://www3.bcity.com/aldev/>`
- 映射站台 `<http://www.members.spree.com/technology/aldev/>`

D 附錄丁—安裝 Midgard

來自 `<http://www.midgard-project.org/download/binaries>` 的 RPMS 目前不包括 PostgreSQL 支援，所以你要從原始碼 tar.gz 檔安裝。

下載 Midgard 原始碼 tar.gz 檔，閱讀 INSTALL.REDHAT 檔—

```
bash# cd midgard-lib-1.4beta6
bash# ./configure --prefix=/usr/local --with-mysql=/usr/local --includedir=/usr/include/mysql --with-mi
bash# make
bash# make install
bash# ldconfig -v | grep -i midga
```

拷貝櫃頭檔，以防 make install 不這樣做…

```
bash# cp *.h /usr/local/include
```

```
bash# cd ../mod_midgard-1.4beta5c
bash# ./configure --prefix=/usr/local --with-mysql=/usr/local --includedir=/usr/include/mysql --with-mi
bash# make
bash# make install
# 修改 apache 設定為 /usr/……
#modify apache line to correct /usr/.....
bash# vi /etc/httpd/conf/httpd.conf   （或在  /etc/apache/httpd.conf）
bash# /etc/init.d/apache restart
# 要重新啓動 apache！！！
```

```
bash# cd ../midgard-php-1.4beta6
bash# ./configure '--with-apxs' '--with-mysql' '--with-pgsql' '--with-midgard' --prefix=/usr/local --wi
```

```
bash# gvim Makefile
```

在 INCLUDE 變數中加上 -I/usr/include/pgsql。

也要如下在 $(APXS) 指令中加上 $(INCLUDE)一

```
libphp3.so: mod_php3.c libmodphp3-so.a  pcrelib/libpcre.a midgard/libphpmidgard.a
    -@test -f ./mod_php3.c || test -L ./mod_php3.c || $(LN_S) $(srcdir)/mod_php3.c ./mod_php3.c
    -@test -f ./mod_php3.c || test -h ./mod_php3.c || $(LN_S) $(srcdir)/mod_php3.c ./mod_php3.c
    $(APXS) -c -o libphp3.so  -I$(srcdir) $(INCLUDE) -I. -I/usr/local/include -I/usr/lib/glib/include
$(APXS_LDFLAGS)
```

```
bash# make
bash# make install
# 修改 apache 設定為 /usr/……
# 加上如下數行一
    LoadModule php4_module        modules/libphp4.so
            AddModule mod_php4.c
    LoadModule php4_module        lib/apache/libphp4.so

    <IfModule mod_php4.c>
        AddType application/x-httpd-php4 .php4
        AddType application/x-httpd-php4 .php
        AddType application/x-httpd-php4-source .phps
        AddType application/x-httpd-php .php
    </IfModule>
```

```
bash# vi /etc/httpd/conf/httpd.conf   （或在 /etc/apache/httpd.conf）

bash# /etc/init.d/apache restart
# 要重新啓動 apache！！！
```

D.1 測試 Midgard PHP 伺服器

要測試安裝是否成功請這樣做一 在你的文件根目錄中建立一個檔案。我通常稱它為 info.php，並在其間加上一行：

```
<?phpinfo()?>
```

在你的瀏覽器中載入：http://localhost/info.php

你會見到一頁漂亮的撮要，列出你的設定中所有資料。 你大概不會把這檔案放在在生產用的伺服器中，但為了開發時除錯和一般資料， 它會很方便。

D.2 保安 OpenSSL

你可能要在 `<ftp://ftp.deva.net/pub/sources/crypto/rsaref20.1996.tar.Z>` 取行 RSA 套裝軟件以啓用 SSL 加密。也請看看 Linux 光碟的 OpenSSL `<http://www.openssl.org>`。

如果你不想啓動 SSL（或你遇到困難），下載原始碼 Apache-Midgard RPM 和修改 *.spec 檔和刪去 SSL，再重見 RPM。