第4章 使用 SQL 语句存取数据

本节讲述如何使用 SQL 语句来连接数据库和检索数据。

在示例中，我们给出了要输入的语句，（大多数情况下）后面跟着在对样本数据库发出该语句时将显示的结果。注意，虽然我们用大写字母显示语句，但也可以用大小写字母混合来输入这些语句（用单引号 (') 或双引号 ('') 括起来的语句除外）。

DB2 通用数据库包括的 SAMPLE 数据库由几个表组成，如第71页的『附录A. 样本数据库表』中所列。可使用“第一步骤”安装启动板来创建此数据库。还可从命令行创建 SAMPLE 数据库。参见 *SQL Reference* 以了解详情。

注意，DB2 通用数据库包含其它样本数据库，它们演示“数据仓库中心”和“OLAP Starter 工具箱”功能。此书中的示例仅使用一般 SAMPLE 数据库。

视数据库设置方式的不同，您可能必须通过给所使用的表名加上模式名和一个句点作为前缀来限定这些表名。对于本书中的示例，假定缺省模式为 USERID。所以可以将表 ORG 引用为 USERID.ORG。可询问管理员这样做是否有必要。

本章包括:

- 连接数据库
- 调查错误
- 选择列和选择行
- 将行进行排序和除去重复行
- 运算次序
- 使用表达式来计算值
- 给表达式命名
- 从多个表中选择数据
- 使用子查询
- 使用函数
- 分组

连接数据库

您必须先与数据库连接，才能使用 SQL 语句来查询或操作该数据库。CONNECT 语句使数据库连接与用户名相关联。

例如，要连接 SAMPLE 数据库，在 DB2 命令行处理器中输入下列命令:

```
CONNECT TO SAMPLE USER USERID USING PASSWORD
```

（确保选择的是在服务器系统上有效的用户 ID 和口令）。

在此示例中，USER 的值为 USERID，USING 的值为 PASSWORD。

下列信息告诉您连接成功:

```
   数据库连接信息

 数据库产品              = DB2/NT 7.1.0
 SQL 权限 ID             = USERID
 本地数据库别名          = SAMPLE
```

一旦连接上，就可以开始操作数据库。有关连接的详情，参考 *SQL Reference* 中的 CONNECT 语句。

调查错误

每当在任何示例中出现输入错误时，或者在执行 SQL 语句期间出错时，数据库管理程序会返回错误信息。错误信息由信息标识符、简要说明以及 SQLSTATE 组成。

SQLSTATE 错误是 DB2 系列产品的公共错误码。SQLSTATE 错误符合 ISO/ANSI SQL92 标准。

例如，如果在 CONNECT 语句中用户 ID 或口令不正确，则数据库管理程序将返回信息标识符 SQL1403N 以及 SQLSTATE 08004 。该信息如下:

```
SQL1403N 提供的用户名和／或口令不正确。  SQLSTATE=08004
```

可以通过输入一个问号 (?)，然后输入信息标识符或 SQLSTATE 来获取关于错误信息的详情:

```
? SQL1403N
```

或

```
? SQL1403
```

或

```
? 08004
```

注意，错误 SQL1403N 的说明中倒数第二行表明 SQLCODE 为 -1403。 SQLCODE 为产品特定的错误码。以 N（通知）或 C（严重）结尾的信息标识符表示一个错误，并且具有负 SQLCODE。以 W（警告）结尾的信息标识符表示一个警告，并且具有正 SQLCODE。

选择列

使用 SELECT 语句从表中选择特定的列。在该语句中指定用逗号分隔的列名列表。此列表称为*选择列表*。

下列语句从 SAMPLE 数据库的 ORG 表中选择部门名称 (DEPTNAME) 和部门号 (DEPTNUMB):

```
SELECT DEPTNAME, DEPTNUMB
   FROM ORG
```

以上语句产生下列结果:

```
DEPTNAME       DEPTNUMB
-------------- --------
Head Office          10
New England          15
Mid Atlantic         20
South Atlantic       38
Great Lakes          42
Plains               51
Pacific              66
Mountain             84
```

通过使用星号 (*) 可从表中选择所有列。下一个示例列出了 ORG 表中的所有的列和行:

```
SELECT *
   FROM ORG
```

此语句产生下列结果:

```
DEPTNUMB DEPTNAME       MANAGER DIVISION   LOCATION
-------- -------------- ------- ---------- -------------
      10 Head Office        160 Corporate  New York
      15 New England         50 Eastern    Boston
      20 Mid Atlantic        10 Eastern    Washington
      38 South Atlantic      30 Eastern    Atlanta
      42 Great Lakes        100 Midwest    Chicago
      51 Plains             140 Midwest    Dallas
      66 Pacific            270 Western    San Francisco
      84 Mountain           290 Western    Denver
```

选择行

要从表中选择特定行，在 SELECT 语句之后使用 WHERE 子句指定要选择的行必须满足的条件。从表中选择行的标准是*搜索条件*。

搜索条件由一个或多个*谓词*组成。谓词指定关于某一行是真或是假（或未知）的条件。可使用下列基本谓词在 WHERE 子句中指定条件:

谓词	功能
x = y	x 等于 y
x <> y	x 不等于 y
x < y	x 小于 y
x > y	x 大于 y
x <= y	x 小于或等于 y
x >= y	x 大于或等于 y
IS NULL/IS NOT NULL	测试空值

在构造搜索条件时，要注意只对数字数据类型执行算术运算，并只在相容数据类型之间进行比较。例如，不能将文本字符串与数字进行比较。

如果正在根据字符值来选择行，则该值必须用单引号括起来（例如，`WHERE JOB = 'Clerk'`），并且输入的每个字符值必须与数据库中的完全一样。如果数据值在数据库中是小写的，而您用大写形式来输入它，将不选择行。如果正在根据数字值来选择行，该值不能用引号括起来（例如，`WHERE DEPT = 20`）。

下列示例只从 STAFF 表中选择部门 20 的行:

```
SELECT DEPT, NAME, JOB
   FROM STAFF
   WHERE DEPT = 20
```

此语句产生下列结果:

```
DEPT   NAME      JOB
------ --------- -----
    20 Sanders   Mgr
    20 Pernal    Sales
    20 James     Clerk
    20 Sneider   Clerk
```

下一个示例使用 AND 来指定多个条件。可以指定任意多个条件。该示例从 STAFF 表中选择部门 20 中的职员:

```
SELECT DEPT, NAME, JOB
   FROM STAFF
   WHERE JOB = 'Clerk'
   AND DEPT = 20
```

此语句产生下列结果:

```
DEPT   NAME      JOB
------ --------- -----
    20 James     Clerk
    20 Sneider   Clerk
```

未在其中输入值且不支持缺省值的列中出现空值。将值特别设置为空值的地方也可以出现空值。空值只能在定义为支持空值的列中出现。 第9页的『创建表』中讨论了在表中定义和支持空值。

使用谓词 IS NULL 和 IS NOT NULL 来检查空值。

下列语句列出了佣金未知的雇员:

```
SELECT ID, NAME
   FROM STAFF
   WHERE COMM IS NULL
```

此语句产生下列结果:

```
ID     NAME
------ ---------
    10 Sanders
    30 Marenghi
    50 Hanes
   100 Plotz
   140 Fraye
   160 Molinare
   210 Lu
   240 Daniels
   260 Jones
   270 Lea
   290 Quill
```

值零与空值不相同。下列语句选择表中佣金为零的每个人:

```
SELECT ID, NAME
   FROM STAFF
   WHERE COMM = 0
```

因为样本表中的 COMM 列中没有零值，所以返回的结果集为空。

下一个示例选择 STAFF 表中 YEARS 的值大于 9 的所有行:

```
SELECT NAME, SALARY, YEARS
   FROM STAFF
   WHERE YEARS > 9
```

此语句产生下列结果:

```
NAME      SALARY    YEARS
--------- --------- ------
Hanes      20659.80     10
Lu         20010.00     10
Jones      21234.00     12
Quill      19818.00     10
Graham     21000.00     13
```

将行进行排序

您可能想要信息按特定次序返回。使用 ORDER BY 子句将信息按一个或多个列中的值进行排序。

下列语句显示部门 84 中按雇用年数排序的雇员:

```
SELECT NAME, JOB, YEARS
   FROM STAFF
   WHERE DEPT = 84
   ORDER BY YEARS
```

此语句产生下列结果:

```
NAME      JOB   YEARS
--------- ----- ------
Davis     Sales      5
Gafney    Clerk      5
Edwards   Sales      7
Quill     Mgr       10
```

指定 ORDER BY 作为整个 SELECT 语句中的最后一个子句。在此子句中命名的列可以是表达式或表的任何列。 ORDER BY 子句中的列名不必在选择列表中指定。

可通过在 ORDER BY 子句中显式指定 ASC 或 DESC 将行按升序或降序进行排序。如果既未指定 ASC, 也未指定 DESC, 则自动按升序将行进行排序。下列语句按雇用年数以降序显示部门 84 中的雇员:

```
SELECT NAME, JOB, YEARS
   FROM STAFF
   WHERE DEPT = 84
   ORDER BY YEARS DESC
```

此语句产生下列结果:

```
NAME      JOB   YEARS
--------- ----- ------
Quill     Mgr       10
Edwards   Sales      7
Davis     Sales      5
Gafney    Clerk      5
```

可以按字符值以及数字值将行进行排序。下列语句按姓名字母顺序显示部门 84 的雇员:

```
SELECT NAME, JOB, YEARS
   FROM STAFF
   WHERE DEPT = 84
   ORDER BY NAME
```

此语句产生下列结果:

```
NAME      JOB   YEARS
--------- ----- ------
Davis     Sales      5
Edwards   Sales      7
Gafney    Clerk      5
Quill     Mgr       10
```

除去重复行

当使用 SELECT 语句时，您可能不想要返回重复信息。例如，STAFF 有一个其中多次列出了几个部门号的 DEPT 列，以及一个其中多次列出了几个工作说明的 JOB 列。

要消除重复行，在 SELECT 子句上使用 DISTINCT 选项。例如，如果将 DISTINCT 插入该语句，则部门中的每项工作仅列出一次:

```
SELECT DISTINCT DEPT, JOB
   FROM STAFF
   WHERE DEPT < 30
   ORDER BY DEPT, JOB
```

此语句产生下列结果:

```
DEPT   JOB
------ -----
    10 Mgr
    15 Clerk
    15 Mgr
    15 Sales
    20 Clerk
    20 Mgr
    20 Sales
```

DISTINCT 已消除了在 SELECT 语句中指定的一组列中所有包含重复数据的行。

运算次序

考虑运算次序是很重要的。一个子句的输出是下一个子句的输入，如下面列表中所述。第25页的『给表达式命名』中给出一个要考虑其中运算次序的示例。

下列运算顺序不一定是 DB2 代码内执行运算的方法。这个简单解释仅允许以更直观的方式考虑查询。运算顺序如下:

1. FROM 子句
2. WHERE 子句
3. GROUP BY 子句
4. HAVING 子句
5. SELECT 子句
6. ORDER BY 子句

使用表达式来计算值

*表达式*是包括在语句中的计算或函数。下列语句计算，如果部门 38 中每个雇员都收到 $500 的奖金，则每人的工资将是多少:

```
SELECT DEPT, NAME, SALARY + 500
   FROM STAFF
   WHERE DEPT = 38
   ORDER BY 3
```

此结果为:

```
DEPT   NAME      3
------ --------- ----------------
    38 Abrahams          12509.75
    38 Naughton          13454.75
    38 Quigley           17308.30
    38 Marenghi          18006.75
    38 O'Brien           18506.00
```

注意第三列的列名是一个数字。这是一个系统生成的数字，因为 SALARY+500 未指定列名。以后此数字在 ORDER BY 子句中用来表示第三列。第25页的『给表达式命名』论及如何给表达式取有意义的名称。

可使用基本算术运算符加 (+)、减 (−)、乘 (*) 和除 (/) 来构成算术表达式。

这些运算符可对几种不同类型操作数的数值进行运算，其中某些操作数为:

- 列名（例如在 RATE*HOURS 中）
- 常数值（例如在 RATE*1.07 中）
- 标量函数（例如在 LENGTH(NAME) + 1 中）。

给表达式命名

可选的 AS 子句允许您给表达式指定有意义的名称，这就使得以后再引用该表达式更容易。可使用 AS 子句为选择列表中的任何项提供名称。

下列语句显示其工资加佣金少于 $13,000 的所有雇员。表达式 SALARY + COMM 命名为 PAY:

```
SELECT NAME, JOB, SALARY + COMM AS PAY
   FROM STAFF
   WHERE (SALARY + COMM) < 13000
   ORDER BY PAY
```

此语句产生下列结果:

```
NAME      JOB   PAY
--------- ----- ----------
Yamaguchi Clerk   10581.50
Burke     Clerk   11043.50
Scoutten  Clerk   11592.80
Abrahams  Clerk   12246.25
Kermisch  Clerk   12368.60
Ngan      Clerk   12714.80
```

通过使用 AS 子句，可以在 ORDER BY 子句中引用特定的列名而不是系统生成的数字。在此示例中，我们在 WHERE 子句中将 (SALARY + COMM) 与 13000 进行比较，而不是使用名称 PAY。这是运算次序的结果。在给定 (SALARY + COMM) 名称 PAY 之前计算 WHERE 子句的值，原因是 SELECT 子句在 WHERE 子句后执行。因此，不能在该谓词中使用 PAY。

从多个表中选择数据

可使用 SELECT 语句从两个或多个表中生成包含信息的报告。这通常称为*连接*。例如，可以连接 STAFF 和 ORG 表中的数据以形成一个新表。要连接两个表，在 SELECT 子句中指定想要显示的列，在 FROM 子句中指定表名，在 WHERE 子句中指定搜索条件。 WHERE 子句是可选的。

下一个示例使每个经理的姓名与部门名称关联。需要从两个表中选择信息，因为雇员信息（STAFF 表）和部门信息（ORG 表）是独立存储的。下列查询分别选择 STAFF 和 ORG 表的 NAME 和 DEPTNAME 列。搜索条件将选择范围缩小为 MANAGER 列中的值与 ID 列中的值相同的行:

```
SELECT DEPTNAME, NAME
   FROM ORG, STAFF
   WHERE MANAGER = ID
```

图3 演示如何比较两个不同表中的列。加框线的值表示满足搜索条件的匹配项。

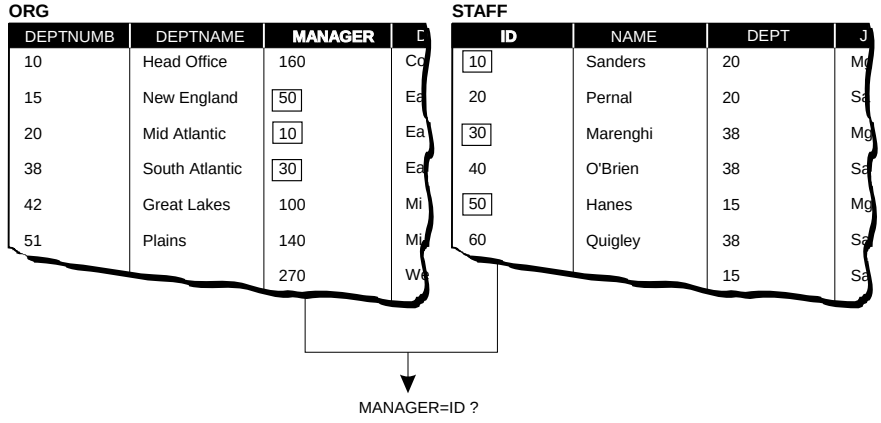

图 3. 从 STAFF 和 ORG 表中选择

SELECT 语句产生下列结果:

```
DEPTNAME       NAME
-------------- ---------
Mid Atlantic   Sanders
South Atlantic Marenghi
New England    Hanes
Great Lakes    Plotz
Plains         Fraye
Head Office    Molinare
Pacific        Lea
Mountain       Quill
```

该结果列出了每个经理的姓名和他或她的部门。

使用子查询

在编写 SQL SELECT 语句时，可在 WHERE 子句中放置附加的 SELECT 语句。每个附加的 SELECT 启动一个子查询。

然后，子查询本身又可包括其结果代入原始查询的 WHERE 子句的另一子查询。另外，WHERE 子句可将子查询包括在多个搜索条件中。子查询可引用与主查询中所使用的不同的表和列。

下列语句从 ORG 表中选择 STAFF 表中其 ID 为 280 的雇员的分部和位置:

```
SELECT DIVISION, LOCATION
   FROM ORG
   WHERE DEPTNUMB = (SELECT DEPT
                        FROM STAFF
                        WHERE ID = 280)
```

在处理语句时，DB2 首先确定子查询的结果。因为 ID 为 280 的雇员在部门 66，所以此示例的子查询的结果为 66。然后，最终结果从其 DEPTNUMB 列具有值 66 的 ORG 表的行中得出。最终结果是:

```
DIVISION   LOCATION
---------- -------------
Western    San Francisco
```

当使用子查询时，数据库管理程序计算该子查询并将结果值直接代入 WHERE 子句。

在第38页的『相关子查询』中进一步讨论了子查询。

使用函数

本节简要介绍了将用于全书示例的函数。*数据库函数*是一组输入数据值和一个结果值之间的关系。

函数可以是*内部*的或*用户定义*的。 DB2 通用数据库提供很多内部函数和预安装的用户定义函数。

可找到 SYSIBM 模式的内部函数，而预安装的用户定义函数为 SYSFUN 模式。SYSIBM 和 SYSFUN 是保留模式。

内部函数和预安装的用户定义函数始终不能满足所有的用户需求。因此，应用程序开发者可能需要创建自己的一套专门针对他们的应用程序的函数。用户定义函数使这成为可能，例如通过扩展 DB2 通用数据库的范围以包括定制的商业或科学函数。这在第66页的『用户定义函数』中进一步讨论。

列函数

*列函数*对列中的一组值进行运算以得到单个结果值。下列就是一些列函数的示例。要获取完整列表，参考 *SQL Reference*。

AVG 返回某一组中的值除以该组中值的个数的和

COUNT 返回一组行或值中行或值的个数

MAX 返回一组值中的最大值

MIN 返回一组值中的最小值

下列语句从 STAFF 表中选择最高工资:

```
SELECT MAX(SALARY)
   FROM STAFF
```

此语句从 STAFF 样本表中返回值 22959.20。

下一个示例选择其收入超过平均收入但在公司的年数少于平均年数的雇员姓名和工资。

```
SELECT NAME, SALARY
   FROM STAFF
   WHERE SALARY > (SELECT AVG(SALARY) FROM STAFF)
   AND YEARS < (SELECT AVG(YEARS) FROM STAFF)
```

此语句产生下列结果:

```
 NAME      SALARY
--------- ---------
Marenghi   17506.75
Daniels    19260.25
Gonzales   16858.20
```

在以上示例的 WHERE 子句中，在*子查询*中说明了列函数，而不是直接实现列函数（例如: WHERE SALARY > AVG(SALARY)）。不能在 WHERE 子句中说明列函数。这是由运算次序导致的结果。将 WHERE 子句考虑为在 SELECT 子句之前进行计算。因此，当正在计算 WHERE 子句时，列函数没有对该组值的存取权。稍后由 SELECT 子句选择这组值。

可使用 DISTINCT 元素作为列函数自变量的一部分，以在应用函数之前消除重复值。因此，COUNT(DISTINCT WORKDEPT) 计算不同部门的个数。

标量函数

*标量函数*对一个单一值进行某个运算以返回另一个单一值。下列就是一些由 DB2 通用数据库提供的标量函数的示例。

ABS 返回数的绝对值

HEX 返回值的十六进制表示

LENGTH 返回自变量中的字节数（对于图形字符串则返回双字节字符数。）

YEAR 抽取日期时间值的年份部分

有关标量函数的详细列表和说明，参考 *SQL Reference*。

下列语句返回 ORG 表中的部门名称以及其每个名称的长度:

```
SELECT DEPTNAME, LENGTH(DEPTNAME)
   FROM ORG
```

此语句产生下列结果:

```
DEPTNAME       2
-------------- -----------
Head Office             11
New England             11
Mid Atlantic            12
South Atlantic          14
Great Lakes             11
Plains                   6
Pacific                  7
Mountain                 8
```

注意，由于未使用 AS 子句给 LENGTH(DEPTNAME) 取一个有意义的名称，所以第二列中出现系统生成的数字。

表函数

*表函数*返回表的列，类似于由单一 CREATE TABLE 语句创建的表。

表函数仅可用于 SQL 语句的 FROM 子句。

DB2 通用数据库中当前唯一支持的表函数是 SQLCACHE_SNAPSHOT。

SQLCACHE_SNAPSHOT

将 DB2 动态 SQL 语句高速缓存的瞬象的结果作为表返回。

分组

DB2 通用数据库具有基于表的特定列对数据进行分析的能力。

可根据 GROUP BY 子句中定义的分组结构来组织行。最简单的格式为，一个*组*就是一组行，每一组在 ″GROUP BY″ 列中都具有完全相同的值。SELECT 子句中的列名必须为分组列或列函数。列函数对 GROUP BY 子句定义的每个组返回一个值。每一组由结果集中的单一行表示。下列示例产生一个列出了每个部门号的最高工资的结果:

```
SELECT DEPT, MAX(SALARY) AS MAXIMUM
   FROM STAFF
   GROUP BY DEPT
```

此语句产生下列结果:

```
DEPT   MAXIMUM
------ ---------
    10  22959.20
    15  20659.80
    20  18357.50
    38  18006.00
    42  18352.80
    51  21150.00
    66  21000.00
    84  19818.00
```

注意，计算的是每个部门（由 GROUP BY 子句定义的组）而不是整个公司的 MAX(SALARY)。

将 WHERE 子句与 GROUP BY 子句一起使用

分组查询可以在形成组和计算列函数之前具有消除非限定行的标准 WHERE 子句。必须在 GROUP BY 子句之*前*指定 WHERE 子句。例如:

```
SELECT WORKDEPT, EDLEVEL, MAX(SALARY) AS MAXIMUM
   FROM EMPLOYEE
   WHERE HIREDATE > '1979-01-01'
   GROUP BY WORKDEPT, EDLEVEL
   ORDER BY WORKDEPT, EDLEVEL
```

结果为:

```
WORKDEPT EDLEVEL MAXIMUM
-------- ------- -----------
D11           17    18270.00
D21           15    27380.00
D21           16    36170.00
D21           17    28760.00
E11           12    15340.00
E21           14    26150.00
```

注意，在 SELECT 语句中指定的每个列名也在 GROUP BY 子句中提到。未在这两个地方提到的列名将产生错误。GROUP BY 子句对 WORKDEPT 和 EDLEVEL 的每个唯一组合各返回一行。

在 GROUP BY 子句之后使用 HAVING 子句

可将限定条件应用于各个组，以便 DB2 仅对满足条件的组返回结果。为此，*在* GROUP BY 子句后面包含一个 HAVING 子句。 HAVING 子句可包含一个或多个用 AND 和 OR 连接的谓词。每个谓词将组特性（如 AVG(SALARY)）与下列之一进行比较:

- 该组的另一个特性

  例如:

```
HAVING AVG(SALARY) > 2 * MIN(SALARY)
```

- 常数

  例如:

```
HAVING AVG(SALARY) > 20000
```

例如，下列查询查找雇员数超过 4 的部门的最高和最低工资:

```
SELECT WORKDEPT, MAX(SALARY) AS MAXIMUM, MIN(SALARY) AS MINIMUM
   FROM EMPLOYEE
   GROUP BY WORKDEPT
   HAVING COUNT(*) > 4
   ORDER BY WORKDEPT
```

此语句产生下列结果:

```
WORKDEPT MAXIMUM     MINIMUM
-------- ----------- -----------
D11         32250.00    18270.00
D21         36170.00    17250.00
E11         29750.00    15340.00
```

有可能（虽然很少见）查询有 HAVING 子句但没有 GROUP BY 子句。在此情况下，DB2 将整个表看作一个组。因为该表被看作是单个组，所以最多可以有一个结果行。如果 HAVING 条件对整个表为真，则返回选择的结果（该结果必须整个由列函数组成）；否则不返回任何行。