

FoxPRO 2.5 程序设计讲座 (六)

罗 辉 (湖南省双峰工行) 何丹鸿 (湖南省娄底人行)

第八讲 结构化查询和多库查询

查询即在数据库或表中选定记录和字段的请求。Foxpro2.5 提供两种途径对数据库记录进行查询: SQL 结构化查询和用全屏幕编辑命令、索引、浏览变化窗口等命令和技术构造的程序查询。

一、RQBE 和结构化查询

SQL 即结构化查询语言的缩写,它是用于和数据库对话的标准化命令。Foxpro2.5 支持 SQL,确切地说,部分支持 SQL,主要的就是支持 SELECT 命令。

SELECT 命令从表中检索信息。它可直接在命令窗口输入,也可在实例关系查询 RQBE 窗口交互式地建立。

1. RQBE 查询构造器及查询命令

Foxpro 的 RQBE 窗口允许用 Foxpro 界面风格交互式建立 SQL 查询,使用诸如复选框、按钮及无线按钮等 Foxpro 对象,可标识被查询的数据库文件、指定文件的关联条件、选择输出字段、指定输出标准、依次输出、定向输出、保存 SQL 的构造等,其实质是构造 SELECT 命令并可执行的窗口,最适宜于初学者学习 SQL。它构造的查询默认保存在扩展名为.QPR 文件中,编译后的文件扩展名是.QPX。

Foxpro2.5 的 SQL 如何进行查询,充其量是如何有效地构造 SELECT 命令。SELECT 命令可从一个或多个数据库/表中查询数据。SELECT 的命令格式是:

```
SELECT [ALL  DISTINCT][<别名>.]<条目> [AS <
栏目名>]
[,<别名>.]<条目> [AS <栏目名>]...]
FROM <库/表> [<局部别名>][<库/表> [<局部
别名>]...]
[[INTO <目的>] [TO FILE <文件名>
[ADDITIVE]
```

```
TO PRINTER [PROMPT] TO SCREEN]]
[PREFERENCE <资源名>][NOCONSOLE] [PLAIN]
[NOWAIT]
[WHERE <关联条件> [AND <关联条件> ...]
[AND OR <过滤条件> [AND OR <过滤条件>
...]]]
[GROUP BY <分组栏> [, <分组栏> ...]]
[HAVING <过滤条件>][UNION [ALL] <SELECT 命
令>]
[ORDER BY <排序条目>[ASC DESC],<排序条目
>[ASC DESC]...]
```

当你用 SELECT 提出一个查询时,FoxPro 解释这一查询并从数据库/表中查询指定的数据。Foxpro 使用 SELECT 命令就象使用其它 FoxPro 命令一样毫无二致,且在执行 SELECT 命令之前,并不需要事先将数据库打开。

2. 指定要查询的条目

SELECT 命令必须标明在查询结果中要显示的条目,如字段名、常量或表达式等。默认为 ALL,显示所有的字段域。每一条目将在查询结果中产生一系列结果。如果有几个条目同名,必须前置数据库别名加以区分。用 DISTINCT 保证查询结果唯一。

查询结果输出时,系统默认为查询结果头建立一行栏目名。栏目名的确定遵循如下规则:

- (1)如果条目是一个唯一的字段名,则栏目名即该字段名。
- (2)如果有多个条目有同样的名字,则输出栏目名用该名字带 A、B...显示。
- (3)如果条目是表达式,则输出为 EXP A、EXP B、EXP C 等。
- (4)如果条目包含了统计函数如 COUNT()或 SUM()等,则输出栏目名命名为 CNT A 或 SUM B。可以用 AS 为每一条目另外指定用于结果输出的栏

目名。当<条目>是一个表达式、或者要给该栏以一个有意义的名字时,这是十分有用的。<栏目名>必须是字符串,而且该字符串不需用引号括起来。但是在字段名中不能出现的字符如空格等,在这里也不能出现;而且十分遗憾的是,栏目名不能是汉字。

【例 8.1】以大写方式显示在 CUSTOMER.DBF 中全部城市的名称且该栏目显示为 CityList。

```
SELECT UPPER(city) AS CityList FROM
customer
```

但当 SELECT 中使用了 PLAIN 子句,将禁止显示栏目名。不过如存在 INTO,则 PLAIN 的作用会被忽略。

用作条目的对象必须是在 FROM 子句里标明的数据库中的字段名;或者是一个常量说明符,其中的常量出现在查询结果的每一行相应位置上;或者是一个表达式,它可以是一个用户自定义函数名。

不过,虽然在 SELECT 子句中可使用用户自定义函数作为条目使用获得便利,但也要考虑如下约束情况:

(1)SELECT 的操作速度可能要受到用户自定义函数的执行速度的制约。因而需大量调用的用户自定义函数最好用 API 或用 C 语言编写。

(2)在 SELECT 中调用的用户自定义函数中改变 FoxPro 输入输出环境或数据库环境是很不安全的,通常其结果是非预期的。

(3)向 SELECT 调用的用户自定义函数传送数据,最好通过函数调用的参数表实现。

(4)如果你试验并发现一个非法的操作在 Foxpro 某一版本上正常工作了,这并不能保证在未来版本的 Foxpro 下仍能正常工作。

除了这些局限性之外,你可以随意的在 SELECT 中使用用户自定义函数。

下面有关字段操作的函数可出现在条目中,当做条目使用:

AVG(<字段>)••数值型数据字段栏的平均值。

COUNT(<字段>)•字段栏中值的数目。COUNT(*)对输出的行数进行统计。

MIN(<字段>)••栏中最小值。

MAX(<字段>)••栏中最大值。

SUM(<字段>)••数值数据栏的总和。

这些函数不能嵌套使用。

3.指定数据源

可以用 FROM 子句列出查询将从中搜索信息的数据库/表。如在当前目录或 Foxpro 路径中没有指定的数据库/表,系统将打开 Open 对话框以便你指定文件位置。

如果查询中用到的数据库/表原先并未打开,查询后它将保持打开在某一工作区中。

也可以为数据库/表指定一个局部别名作为 SELECT 命令中临时使用的名字。如果指定了局部别名,则在整个 SELECT 命令中必须使用局部别名取代该数据库名所在位置。局部别名不影响 Foxpro 环境,它在查询之后自动放弃。

【例 8.2】不显示记录重复的数据,其中数据库 customer、invoices 被分别赋局部别名 a、b,它们在 SELECT 命令之后自动放弃。

```
SELECT DISTINCT a.company, b.idate, b.itotal ;
FROM customer a, invoices b ;
WHERE a.cno = b.cno
```

4.指定查询结果输出去向

在 RQBE 窗口中,可指定 SELECT 命令查询结果的输出去向。结果可以送到 BROWSE 窗口、数组、文本文件、打印机、Foxpro 主窗口、报表生成器、游标、数据库/表、Foxpro 的 FoxGraph 图形处理器。

在 SELECT 命令中使用 INTO 子句决定查询结果是否存到数组、游标、数据库/表;如果使用 TO 子句而没用 INTO 子句,则可将查询结果存到一个文本文件或到打印机或到 Foxpro 主窗口中。

如果使用了 INTO 子句,则在屏幕上无输出显示;当 SELECT 中同时带有 INTO 和 TO 子句,则 TO 子句将被忽略。

如果不包含 INTO 和 TO 子句,则查询结果将显示在一个 Browse 窗口中。如果查询结果被送到一个 Browse 窗口,可用 PREFERENCE 子句保存 Browse 窗口的属性和参数到 FOXUSER 资源文件以便以后直接引用。如果 SELECT 是用在程序中,当查询结果被保存在一个 Browse 窗口时,可用 NOWAIT 使在打开

Browse 窗口后不停止程序的执行。如包含 INTO, 则 NOWAIT 被忽略。

INTO 子句可使用如下格式之一:

(1) INTO ARRAY <数组名> --- 指定结果存到数组。

(2) INTO CURSOR <游标名> --- 存储结果到一个游标, 游标中的查询结果可以用 BROWSE 窗口浏览, 或送图形处理器进行图形处理, 或作为报表生成器的数据来源。如果你指明游标名为一个已打开的数据库名, FoxPro 将关闭该数据库, 并产生一个同名的游标, 且如果 SET SAFETY 是 OFF 的话, 将没有警告。执行 SELECT 之后, 这个临时游标仍保持打开和活动, 但它是只读的。一旦你关闭了该临时游标, 它自动消失。

当储存结果到了游标后, 在 SELECT 命令之后紧跟着一条将结果传送到 Foxpro 的图形处理器 FoxGraph 的命令: DO (GENGRAPH), 系统将查询结果自动送往 FoxGraph.EXE 软件, 并直接启动该程序进行图形输出。其中系统参数 GENGRAPH 保存了图形驱动程序的执行路径。

当储存结果到了游标后, 在 SELECT 命令之后紧跟着一条建立报表的命令: CREATE REPORT <报表名> FROM <游标名>, 系统将查询结果自动送往报表文件, 可用 REPORT FROM <报表名> 命令输出报表。

(3) INTO DBF <库> TABLE <表> [将结果储存到一个数据库/表中。如果指明的数据库/表已打开, FoxPro 将关闭它然后重新打开。其默认扩展名为 .DBF。SELECT 执行后, 这一数据库保持打开和活动。

不管怎样, 当在 SET TALK ON 之后执行 SELECT 命令, FoxPro 显示查询时间和查询结果的记录数。系统变量 TALLY 储存了结果的记录数。

【例 8.3】将两个表中匹配的数据存储到第三个表中。

```
SELECT a.company, b.idate, b.itotal ;
FROM customer a, invoices b ;
WHERE a.cno = b.cno ;
INTO TABLE custinv.dbf &&数据库 custinv.dbf 将保持打开在当前工作区
```

BROWSE &&浏览新库中的数据
CLOSE DATABASES

TO 子句也有三种使用格式:

(1) TO FILE <文件> [ADDITIVE] --- 结果存到一个文本文件。加 ADDITIVE 表示不覆盖原文件, 查询结果从原文件尾开始存放。

(2) TO PRINTER [PROMPT] --- 结果直接送打印机。在 FoxPro for Windows 中, 如果查询结果指向到一个打印机, 你可包含可选子句 PROMPT 在打印开始前显示一个对话框。在这一对话框中你可进行打印机或打印设置。

(3) TO SCREEN --- 结果在 Foxpro 主窗口上显示。

用 NOCONSOLE 禁止查询结果被显示, 它用于不带 TO 的命令中。如存在 INTO 子句, NOCONSOLE 被忽略。

5. 记录选择和子查询

往往需要选定数据库中特定的一组记录进行查询, 而不是对库中所有记录进行查询; 有时还要对有关联的多个库/表进行联合查询。从一个或多个库/表中选择满足某一条件的一组记录, 可用 WHERE 子句带一定选择条件来实现。这在 RQBE 窗口是通过选定规则 (Selecting Criteria) 完成选择条件的设计的。

WHERE 子句可使用 <关联条件> 和 <过滤条件> 两类选择条件。

<关联条件> 连接在 FROM 中出现的两个库表。用逻辑运算符可将多个关联条件组合起来, 以实现多个库/表的连接; <过滤条件> 选择合乎要求的记录, 同样也可将多个过滤条件组合起来完成复杂的记录组选择。

【例 8.4】显示发货单的日期不迟于 05/08/90 的记录。

```
SELECT a.company, b.idate, b.itotal ;
FROM customer a, invoices b ;
WHERE a.cno = b.cno ; &&关联条件
AND b.idate < {05/08/90} &&过滤条件
```

如果查询中的两个库表未进行连接, 则第一个库/表中每一个记录将被连接到第二个库表中的每一个记录, 这样查询结果将很长, 往往也不是我们要查询的结

果。

构造 WHERE 子句需注意两点:

(1)在关联条件中要仔细使用 (DELETED(), RECNO(), EOF(), FOUND(), RECCOUNT()等)等支持别名和工作区名的函数,一不小心,它们极易产生不期望的结果。要强调的一点是:SELECT 不使用你的工作区。

(2)用字段空值进行记录的连接要特别小心。例如,用邮政编码字段 ZIP 连接 CUSTOMER.ZIP 和 INVOICE.ZIP, CUSTOMER 含了 100 条空邮政编码的记录,而 INVOICE 含了 400 条空邮政编码,则查询结果中由于这些空值将产生 40,000 额外的记录。这时应使用 EMPTY()函数以屏蔽空值记录的出现。

Foxpro2.5 可执行子查询,即子 SELECT 语句的查询结果可成为父 SELECT 的查询条件。同级的 WHERE 可同时出现不嵌套的多个子查询。

在过滤条件中,除了可使用标准比较操作符外,还可附加七个特定的关键字:

(1)ALL--与集内所有的值进行比较,条件全部满足,则返回真。

【例 8.5】`taxrate < ALL (SELECT taxrate FROM ; customer WHERE state = "WA")`

子查询得到州名 STATE 为 "WA" 的所有 TAXRATE 值,作为父 SELECT 的比较条件:选择小于所有已被选择的 TAXRATE 值的 TAXRATE。

(2)SOME / ANY 与集内所有的值进行比较,条件(至少)满足一次,则返回真。

【例 8.6】`taxrate < ANY (SELECT taxrate FROM ; customer WHERE state = "WA")`

(3)BETWEEN--值落在指定范围,则返回真。

【例 8.7】`customer.taxrate BETWEEN 5.50 AND 6.00`

(4)EXISTS--集内不为空,则返回真。

【例 8.8】显示与 INVOICES 中邮政编码匹配的 CUSTOMER.DBF 中的所有公司名。

`SELECT company FROM customer a WHERE ;`

`EXISTS (SELECT * FROM offices b WHERE a.zip = b.zip)`

(5)IN--值在集内,则返回真。

【例 8.9】`customer.zip NOT IN("98052", "98072", "98034")`

(6)LINK--允许用通配符来过滤。可使用的通配符:百分号%,通配任意个字符;下划线,通配一个字符。

【例 8.10】`customer.state NOT LIKE "%WA%"`,选择州名中不含有 "WA" 的所有记录。又如匹配模式 "F X%",它匹配所有以 F 开头、第二个字符任意、第三个字符为 X、以后可以是任意个字符序列的值,因此它匹配 "FOX"、"FOXBASE"、"FPXASAS" 等字符串。

6.分组处理查询结果

对查询结果分组进行处理,可用 GROUP BY 对记录分组,还可用 HAVING 选择特定的分组进行处理。

用 GROUP BY 子句对查询结果进行分组, <分组栏>是一标准的数据库字段名、或含有 SQL 字段操作函数的计算字段、或一个指示保存查询结果的数据库/表的栏号(最左栏号为 1)。

如果你分组数据并只想选择那些满足指定过滤条件的记录,应使用 HAVING 和 GROUPBY 子句。HAVING 在分组上和 WHERE 类似,但 HAVING 的过滤条件不能用于子查询。

【例 8.11】按邮政编码对客户数进行统计,且仅限于 ZIP 大于 10000 的记录。

`SELECT zip,COUNT(zip) AS COMPANYYN ; UM FROM customer ;`

`GROUP BY zip HAVING zip > 10000`

没有 GROUP BY 的 HAVING 子句功能上相当于 WHERE 子句。但如果 HAVING 子句未使用 SQL 的字段操作函数的话,用 WHERE 的处理速度将更快。

7.查询结果的联合

用 UNION 将两个 SELECT 查询结果组合到一起。默认时,它组合查询结果并剔除重复的结果,可用 ALL 关键字防止剔除重复结果。

使用 UNION 必须满足如下条件:不能用 UNION 组合子查询;两个 SELECT 查询必须输出同样多的栏

数;两个 SELECT 中查询结果的每一栏目必须同数据类型和宽度;只有最后的 SELECT 查询才能使用用于指示输出顺序的 ORDER BY 子句,而且 ORDER BY 将影响整个输出结果。

8.指定查询结果的顺序

使用 ORDER BY 将查询结果排序。其中的排序条目必须是如下两者之一:要么是 FROM 子句标明的数据库/表中的一个字段,而且它是主 SELECT 子句中的条目;要么是一个指明结果库表中的条目位置的数字表达式。

你还可使用 DESC / ASC 指明按降 / 升序排序。

如果不使用 ORDER BY 子句特别说明顺序的话,查询结果无序。

9.查询的执行和修改

通过 RQBE 窗口构造的查询或直接编辑的查询命令可存储到一个默认为 .QPR 后缀的文件中。

用 DO 命令可执行这个文件重现查询结果。通过在命令窗口直接输入 DO <查询文件.QPR> 或通过系统菜单上的 RUN 命令可完成这一工作。

有时还要求对已保存的查询文件进行修改,则可在命令窗口中输入 MODIFY QUERY <查询文件.QPR> 激活 RQBE 窗口对该查询进行修改和重新存盘。

10.Foxpro2.5 的另外两条标准 SQL 命令

Foxpro2.5 提供了另外两条标准 SQL 命令:CREATE TABLE(非交互式建立数据库命令)和 INSERT INTO(不需打开数据库直接向库尾插入记录命令)。其使用简单方便,请参阅有关资料。

11.SQL 与 Rushmore

用户一旦选择了 SELECT,就没有必要牢记在通常情况下的那些规则了,SQL 会自行启用 Rushmore 技术,决定在什么时候需要对一个查询进行优化,并自动完成优化查询,用户甚至不需要打开数据库或相应的索引文件,如果 SQL 觉得需要索引文件,它会为自己建立临时的索引文件。

二、多库查询

简单的应用,其信息可用单个数据库来表示。但实

际工作中的信息,用一个数据库来统一表示有关联的信息往往是不现实的。因此在进行数据库设计时,必须合理组织数据,规范信息的表示,用适当的多个数据库来表示一个完整的应用系统所涉及的所有信息,以尽量降低数据的冗余,保证数据的一致性和完整性。因此,有效地对多个库同时进行查询等操作是一个很实际的问题。

正如上一节所介绍的,用 SQL 可实现多库查询。但是有时 SQL 也显得过于不灵活,有时更希望通过编程直接进行多库的关联查询。

1.用公共字段建立关联

应用系统对多个数据库的查询操作,必须先给这些数据库间建立一种关联,从而有效地组织要查询的信息。通过库间的公共字段关联多个库,建立一个相关表进行有效数据的查询和处理。

SET RELATION 用来在两个数据库(父库和子库)间建立关联。在建立关联之前,父库必须在当前工作区打开,而子库在另外的工作区打开。

关联建立之后,在父库中移动记录指针,子库中的记录指针也将移到相关联的记录上。如果子库中无关联记录存在,则其记录指针将移到子库尾。

相关表必须有一个公共字段。例如,假定父表(CUSTOMER.DBF)容纳客户信息,它包括名字、地址和一个唯一的客户号字段 CNO;子表(INVOICES.DBF)包含了销售信息,它也包含了一个客户号字段 CNO。可基于公共字段客户号 CNO 用 SET RELATION 将两个库间建立关联。但在此之前,子库必须基于公共字段进行了索引。建立关联之后,你根据给定的客户号将父库 CUSTOMER 中的记录指针移到某一记录时,子库中的记录指针也将自动移到了具有同样客户号的记录上。

子库关于公共字段的索引可以是单入口索引(.IDX)、独立复合索引(.CDX)。如果是复合索引,必须用 SET ORDER 指定子库正确的主索引标志 TAG。

如果用 SET RELATION 用非数字的关联表达式将父库与一个未打开索引的子库建立关联,将显示一个警告信息:"Database isn't ordered";但如果关联表达式是数字的,则系统认为是按记录号在父子库间建立关联的:在父库中移动记录指针到某一记录,则子库的记录指

针将移到与父库有相同记录号的记录上。

也可用一条 SET RELATION 命令带 INTO 用逗号分隔的关系列表将一个父库与多个子库建立关联。

当在用 SET RRELATION 命令建立新的关联关系时,如果当前工作区已存在某些关联了,而想在建立新的关联时,仍保持原来的关联,则可在 SET RELATION 命令后附加 ADDITIVE 子句达到目的。

可用函数 RELATION()返回将两个库建立关联的关联表达式。

用 SET RELATION TO 不带关联表达式,删除当前工作区的所有关联关系。用命令 SET RELATION OFF 则可以删除指定的父子关联。

2.处理相关表中一对多关联关系

用 SET RELATION 命令仅在父子库间建立一对一的关联。在父子库建立关联后,当父库记录指针移动时,子库指针将移到与父库中当前记录匹配的的第一个记录上,但它只完成一对一的关联,当子库中有多个记录与父库中一个记录相匹配时,用这种方法将无法检索到子库中剩下的匹配记录。而 SET SKIP TO 命令就是为解决这种一对多问题而提供的,当在父库中记录指针移动到一个新记录之前,子库的记录指针将遍历匹配的的记录,父记录指针将保持在同一记录上。在子库记录指针移到最后一个匹配记录时,在父工作区执行 SKIP 命令,父库记录指针移到一个新记录上,而子库记录指针将相应移到第一个匹配记录上。这时,再执行一次 SKIP 命令,系统自动查看子库中是否还有匹配的的记录,如果有,子库记录指针移到下一个匹配的的记录上,而父库记录指针保持不变。从而保证对所有匹配的的记录进行处理。

用 SET SKIP TO 不带任何参数将取消当前工作区中的一对多关联,但仍保持一对一关联。可以翻看在前面的《数据库浏览命令 BROWSE 的使用》举例。

3.一对多关联数据的浏览

一对多关联数据的显示对不同的查看命令略有不同。当仅用 SET RELATION 命令建立一对一关联时,用 LIST、DISPLAY、BROWSE 等命令同时显示相关联的信息时,将只显示父库记录和子库中第一个匹配的记录的有关信息,对剩下的匹配记录将不显示,好象他们根

本不存在一样。

当用 SET SKIP 命令建立了一对多关联后,使用 LIST、DISPLAY 命令同时显示相关联数据时,在显示每一个匹配的子库记录有关信息的同时,父库当前记录的有关信息将在同一行重复显示;而用 BROWSE 命令浏览代替 LIST 或 DISPLAY 命令后,BROWSE 对父库记录则不重复显示而只显示一次。它对父子库记录分组显示,即在 BROWSE 窗口中将以一个父库记录和匹配的多个子库记录为一组的分组方式显示有关信息。对显示在 Browse 窗口的父库的某一记录,子库中相关的多个记录都可以显示出来。父记录与子库中第一个匹配的的记录一起显示,随后的其它匹配的的记录则显示在父记录和第一个匹配的子记录之后的列上。这时,在 BROWSE 窗口中可快速在每组信息间移动:在父库记录所属的字段栏中,在按住 Ctrl 键的同时,按 ↑ 或 ↓ 键在组间上下移动一个分组的信息,即上下移动一个父记录。子库中无匹配记录时,父库记录指针才可移到另一个新记录上。【例 8.12】本例在 CUSTOMER.DBF 和 INVOICES.DBF 库间建立关联,显示一对一的关联数据。继而建立一对多关联,并同时显示两个库中一对多关联数据。然后撤消一对多关联,为每个库打开了相应的 Browse 窗口,可观察在父库中移动记录时,子库记录指针的移动情况。

```
CLOSE DATABASES
USE customer IN 1          &&父库
USE invoices IN 2         &&子库
SELECT invoices
SET ORDER TO TAG cno      &&指定主索引
SELECT customer
SET RELATION TO cno INTO invoices &&建立关联
LIST OFF ALL Name,Company,Invoices.Payment
BROWSE FIELDS Name,Company,Invoices.Payment
SET SKIP TO invoices      &&建立一对多关联
LIST OFF ALL Name,Company,Invoices.Payment
BROWSE FIELDS Name,Company,Invoices.Payment
SET SKIP TO               &&撤消一对多关联
SELECT invoices           &&打开两个 Browse 窗口
BROWSE NOWAIT            &&在父库窗口移动指针
SELECT customer           &&子库指针将相应移动
BROWSE NOWAIT
```