第三章 关系数据库标准语言 SQL

1. 答:

（1）综合统一。 SQL 语言集数据定义语言 DDL、数据操纵语言 DML、数据控制语言 DCL 的功能于一体。

（2）高度非过程化。用 SQL 语言进行数据操作，只要提出“做什么”，而无须指明“怎么做”，因此无需了解存取路径，存取路径的选择以及 SQL 语句的操作过程由系统自动完成。

（3）面向集合的操作方式。SQL 语言采用集合操作方式，不仅操作对象、查找结果可以是元组的集合，而且一次插入、删除、更新操作的对象也可以是元组的集合。

（4）以同一种语法结构提供两种使用方式。SQL 语言既是自含式语言，又是嵌入式语言。作为自含式语言，它能够独立地用于联机交互的使用方式，也能够嵌入到高级语言程序中，供程序员设计程序时使用。

（5）语言简捷，易学易用。

解析:

详细的可参考《概论》书上 3.1.2。注意不要仅仅背这些特点，关键是要通过具体的练习，使用 SQL 语句来理解这些特点。

2. 答:

SQL 的数据定义功能包括定义表、定义视图和定义索引。

SQL 语言使用 CREATE TABLE 语句建立基本表，ALTER TABLE 语句修改基本表定义，DROP TABLE 语句删除基本表;

使用 CREATE INDEX 语句建立索引， DROP INDEX 语句删除索引;

使用 CREATE VIEW 命令建立视图，DROP VIEW 语句删除视图。

3. 答:

建 S 表：S(SNO，SNAME，STATUS，CITY);

```
CREATE TABLE  S
( SNO     CHAR(3),
 SNAME  CHAR(10),
 STATUS  CHAR(2),
 CITY   CHAR(10));
```

建 P 表：P(PNO，PNAME，COLOR，WEIGHT);

```
CREATE TABLE P
( PNO     CHAR(3),
 PNAME  CHAR(10),
 COLOR  CHAR(4),
 WEIGHT  INT);
```

建 J 表：J(JNO，JNAME，CITY);

```
CREATE TABLE J
( JNO    CHAR(3),
 JNAME  CHAR(10),
 CITY    CHAR(10));
```

建 SPJ 表：SPJ(SNO，PNO，JNO，QTY)；

```
CREATE TABLE SPJ
( SNO  CHAR(3),
 PNO  CHAR(3),
 JNO  CHAR(3),
 QTY  INT);
```

4. 答：

（1）

```
SELECT  SNO
FROM   SPJ
WHERE  JNO='J1';
```

（2）

```
SELECT  SNO
FROM    SPJ
WHERE  JNO='J1' AND  PNO='P1';
```

（3）

```
SELECT  SNO          /*这是嵌套查询*/
FROM    SPJ
WHERE  JNO='J1'
  AND  PNO  IN       /*找出红色零件的零件号码 PNO */
       (SELECT  PNO
        FROM   P      /*从 P 表中找*/
        WHERE  COLOR='红');
```

或

```
SELECT  SNO
FROM   SPJ，P        /*这是两表连接查询*/
WHERE  JNO='J1'     /*这是复合条件连接查询*/
         AND  SPJ.PNO=P.PNO  AND  COLOR='红'；
```

（4）

解析：

读者可以对比第 2 章习题 5 中用 ALPHA 语言来完成该查询的解答：

GET W (J.JNO): ¬∃SPJX(SPJX .JNO=J.JNO ∧

∃SX (SX.SNO=SPJX .SNO ∧ SX .CITY='天津' ∧

∃PX (PX .PNO=SPJX .PNO ∧ PX .COLOR='红'))

如果大家理解了有关该题的解析说明，那么本题的解答可以看成是把关系演算用 SQL 来表示的过程。

```
SELECT  JNO          /*这种解法是使用多重嵌套查询*/
FROM   J             /*注意：从 J 表入手，以包含那些*/
```

```
WHERE  NOT  EXISTS        /*尚未使用任何零件的工程号*/
      ( SELECT  *
        FROM   SPJ
        WHERE  SPJ.JNO=J.JNO
            AND SNO IN
                ( SELECT  SNO    /*天津供应商的 SNO*/
                  FROM   S
                  WHERE  CITY='天津' )
            AND  PNO  IN         /*红色零件的 PNO*/
                ( SELECT  PNO
                  FROM   P
                  WHERE  COLOR='红' ) );
```

或

```
SELECT  JNO
FROM    J
WHERE  NOT EXISTS
        (SELECT  *
         FROM   SPJ, S, P     /*这里的子查询是一个多表连接*/
         WHERE  SPJ.JNO=J.JNO  AND  SPJ.SNO=S.SNO
           AND  SPJ.PNO=P.PNO  AND  S.CITY='天津'
           AND  P.COLOR='红');
```

(5)

解析:

本查询的解析可以参考第二章第 5 题，用 ALPHA 语言的逻辑蕴函来表达。

上述查询可以抽象为:要求这样的工程 x，使 $(\forall y)\ p \rightarrow q$ 为真。即:

对于所有的零件 y，满足逻辑蕴函 $p \rightarrow q$:

P 表示谓词: “供应商 S1 供应了零件 y”

q 表示谓词: “工程 x 选用了零件 y”

即: 只要“供应商 S1 供应了零件 y”为真，则“工程 x 选用了零件 y”为真。

逻辑蕴函可以转换为等价形式:

$(\forall y)p \rightarrow q \equiv \neg (\exists y (\neg(p \rightarrow q)) \equiv \neg (\exists y (\neg(\neg p \vee q) \equiv \neg \exists y(p \wedge \neg q)$

它所表达的语义为: 不存在这样的零件 y，供应商 S1 供应了 y，而工程 x 没有选用 y。

用 SQL 语言表示如下:

```
SELECT  DISTINCT  JNO
FROM   SPJ  SPJZ
WHERE  NOT EXISTS        /*这是一个相关子查询 */
       ( SELECT  *        /*父查询和子查询均引用了 SPJ 表*/
         FROM  SPJ  SPJX   /*用别名 SPJZ、SPJX 将父查询*/
         WHERE  SNO='S1'   /*与子查询中的 SPJ 表区分开*/
           AND NOT EXISTS
                (SELECT  *      /*用别名 SPJY 与父查询*/
                 FROM  SPJ SPJY   /*中的 SPJ 表区分开*/
```

```
                    WHERE  SPJY.PNO=SPJX.PNO
                      AND SPJY.JNO=SPJZ.JNO））；
```

5. 答:

（1）

```
SELECT  SNAME, CITY
FROM    S;
```

（2）

```
SELECT  PNAME, COLOR, WEIGHT
FROM   P;
```

（3）

```
SELECT  JNO
FROM   SPJ
WHERE  SNO=‘S1’;
```

（4）

```
SELECT  P.PNAME, SPJ.QTY
FROM   P, SPJ
WHERE  P.PNO=SPJ.PNO
      AND SPJ.JNO='J2';
```

（5）

```
SELECT  DISTINCT  PNO
FROM   SPJ
WHERE  SNO  IN
       （SELECT  SNO
          FROM   S
          WHERE  CITY='上海'）；
```

（6）

```
SELECT  JNAME
FROM    J, SPJ, S
WHERE   J. JNO=SPJ. JNO
   AND  SPJ. SNO=S.SNO
   AND  S.CITY='上海';
```

或

```
SELECT  JNAME
FROM   J
WHERE  JNO IN
           (SELECT  JNO
            FROM   SPJ, S
            WHERE  SPJ. SNO=S.SNO
          AND S.CITY='上海');
```

（7）

```
SELECT  JNO
FROM   J
WHERE  NOT EXISTS
```

```
                (SELECT *
                 FROM SPJ
                 WHERE SPJ.JNO=J.JNO
                   AND SNO IN
                         (SELECT SNO
                          FROM S
                          WHERE CITY='天津'));
```

或

```
SELECT JNO
FROM J
WHERE NOT EXISTS
          (SELECT *
           FROM SPJ, S
           WHERE SPJ.JNO=J.JNO
            AND SPJ.SNO=S.SNO
            AND S.CITY='天津');
```

（8）

```
UPDATE P
SET   COLOR='蓝'
WHERE COLOR='红';
```

（9）

```
UPDATE SPJ
SET    SNO='S3'
WHERE SNO='S5'
    AND JNO='J4'
    AND PNO='P6';
```

（10）

```
DELETE
FROM SPJ
WHERE SNO='S2';

DELETE
FROM S
WHERE SNO='S2';
```

解析：

注意删除顺序，应该先从 SPJ 表中删除供应商 S2 所供应零件的记录，然后从 S 表中删除 S2。

（11）

```
INSERT INTO SPJ(SNO, JNO, PNO, QTY)  /*INTO 子句中指明列名*/
VALUES (S2，J6，P4，200);            /*插入的属性值与指明列要对应*/
```

或

```
INSERT INTO SPJ        /*INTO 子句中没有指明列名*/
  VALUES (S2，P4，J6，200);   /*插入的记录在每个属性列上有值*/
                              /*并且属性列要和表定义中的次序一致*/
```

6.答:

基本表是本身独立存在的表，在 SQL 中一个关系就对应一个表。

视图是从一个或几个基本表导出的表。视图本身不独立存储在数据库中，是一个虚表。即数据库中只存放视图的定义而不存放视图对应的数据，这些数据仍存放在导出视图的基本表中。视图在概念上与基本表等同，用户可以如同基本表那样使用视图，可以在视图上再定义视图。

7.答:

（1）视图能够简化用户的操作。

（2）视图使用户能以多种角度看待同一数据。

（3）视图对重构数据库提供了一定程度的逻辑独立性。

（4）视图能够对机密数据提供安全保护。

（详细解释参见《概论》第四版 3.6.4）

8.答:

不是。视图是不实际存储数据的虚表，因此对视图的更新，最终要转换为对基本表的更新。因为有些视图的更新不能唯一地有意义地转换成对相应基本表的更新，所以，并不是所有的视图都是可更新的。

如《概论》第四版 3.6.1[例 6]中的视图 S_G（学生的学号及他的平均成绩）

```
CREAT VIEW S_G(Sno，Gavg)
  AS SELECT Sno，AVG(Grade) /*设 SC 表中“成绩”列 Grade 为数字型*/
     FROM SC
     GROUP BY Sno;
```

要修改平均成绩，必须修改各科成绩，而我们无法知道哪些课程成绩的变化导致了平均成绩的变化。

9. 答:

基本表的行列子集视图一般是可更新的。如《概论》第四版 3.6.1 中的[例 1]。

若视图的属性来自集函数、表达式，则该视图肯定是不可以更新的。

如《概论》第四版 3.6.1[例 6]中的 S_G 视图。

10. 答: （略）

解析：不同的系统对视图更新的规定是不同的，读者必须了解你所用系统对视图更新的规定。

11.答: 创建视图:

```
CREATE VIEW    V_SPJ AS
SELECT SNO, PNO, QTY
FROM   SPJ
WHERE JNO=
          (SELECT JNO
          FROM J
          WHERE JNAME='三建');
```

对该视图查询：

（1）找出三建工程项目使用的各种零件代码及其数量。

```
SELECT  PNO, QTY
FROM   V_SPJ;
```

（2）找出供应商 S1 的供应情况。

```
SELECT  PNO, QTY    /* S1 供应三建工程的零件号和对应的数量*/
FROM    V_SPJ
WHERE   SNO='S1';
```