第5章

表数据操作

本章要点

(1) 插入数据。

(2) 修改数据。

(3) 删除数据。

对表进行插入数据、修改数据和删除数据等操作,分别使用数据操纵语言 DML 中的插入语句 INSERT、修改语句 UPDATE 和删除语句 DELETE 来进行。本章介绍插入数据、修改数据、删除数据等内容。

学生信息数据库 stusys 中的学生表 student、课程表 course、成绩表 score、教师表 teacher、讲课表 lecture 的样本数据,参见附录 B 学生信息数据库 stusys 的表结构和样本数据。

5.1 插入数据

下面介绍 INSERT 语句、REPLACE 语句和插入查询结果语句。

5.1.1 为表的所有列插入数据

向数据库的表插入一行或多行数据,使用 INSERT 语句,其基本语法格式如下。

语法格式:

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO]table_name [(col_name , … )]
    VALUES({EXPR| DEFAULT}, … ),( … ), …
    |
```

说明:

(1) table_name:需要插入数据的表名。

(2) col_name:列名,插入列值的方法有两种:

① 不指定列名:必须为每个列都插入数据,且值的顺序必须与表定义的列的顺序一一对应,且数据类型相同;

② 指定列名:只需要为指定列插入数据。

(3) VALUES 子句:包含各列需要插入的数据清单,数据的顺序要与列的顺序相对应。

下面举例说明给表的所有列插入数据时,列名可以省略。设 student 表、student1 表和 student2 表已创建,其表结构参见附录 B。

【例 5.1】 向 student1 表插入一条记录('196001','董明霞','女','1999-05-02','通信',48)。

在 MySQL 命令行客户端输入如下 SQL 语句：

```
mysql > INSERT INTO student1
    -> VALUES ('196001','董明霞','女','1999-05-02','通信',48);
```

执行结果：

```
Query OK, 1 row affected (0.06 sec)
```

使用 SELECT 语句查询插入的数据。

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
|196001 |董明霞  |女     |1999-05-02 |通信      |48 |
+-------+--------+-------+-----------+----------+---+
1 row in set (0.00 sec)
```

可以看出插入全部列的数据成功，在插入语句中，已省略列名表，只有插入值表，且插入值的顺序和表定义的列的顺序相同。

如果插入值的顺序和表定义的列的顺序不同，在插入全部列时，则不能省略列名表，参见下例。

【例 5.2】 向 student1 表插入一条记录，学号为“196002”，姓名为“李茜”，专业为“通信”，总学分 52，性别为“女”，出生日期为“1998-07-25”。

```
mysql > INSERT INTO student1 (sno, sname, speciality, tc, ssex, sbirthday)
    ->     VALUES('196002','李茜','通信',52,'女','1998-07-25');
```

执行结果：

```
Query OK, 1 row affected (0.18 sec)
```

使用 SELECT 语句查询插入的数据：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
|196001 |董明霞  |女     |1999-05-02 |通信      |48 |
|196002 |李茜    |女     |1998-07-25 |通信      |58 |
+-------+--------+-------+-----------+----------+---+
2 rows in set (0.08 sec)
```

5.1.2　为表的指定列插入数据

为表的指定列插入数据，在插入语句中，只给出了部分列的值，其他列的值为表定义时

的默认值，或允许该列取空值。

【例 5.3】 向 student1 表插入一条记录，学号为“196004”，姓名为“周俊文”，性别为“男”，取默认值，出生日期为“1998-03-10”，专业为空值，总学分为 50 分。

```
mysql > INSERT INTO student1 (sno, sname, sbirthday, tc)
    -> VALUES('196004','周俊文', '1998 - 03 - 10', 50);
```

执行结果：

```
Query OK, 1 row affected (0.06 sec)
```

使用 SELECT 语句查询插入的数据：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
|196001 |董明霞  |女     |1999 - 05 - 02|通信   |48 |
|196002 |李茜    |女     |1998 - 07 - 25|通信   |52 |
|196004 |周俊文  |男     |1998 - 03 - 10|NULL   |50 |
+-------+--------+-------+-----------+----------+---+
3 rows in set (0.00 sec)
```

5.1.3 插入多条记录

插入多条记录时，在插入语句中，只需指定多个插入值列表，插入值列表之间用逗号隔开。

【例 5.4】 向 student 表插入样本数据，共 6 条记录，参见附录 B。

```
mysql > INSERT INTO student
    -> VALUES('191001','刘清泉','男','1998 - 06 - 21','计算机',52),
    -> ('191002','张慧玲','女','1999 - 11 - 07','计算机',50),
    -> ('191003','冯涛','男','1999 - 08 - 12','计算机',52),
    -> ('196001','董明霞','女','1999 - 05 - 02','通信',48),
    -> ('196002','李茜','女','1998 - 07 - 25','通信',52),
    -> ('196004','周俊文','男','1998 - 03 - 10','通信',50);
```

执行结果：

```
Query OK, 6 rows affected (0.03 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

使用 SELECT 语句查询插入的数据：

```
mysql > SELECT * FROM student;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
```

```
|196001   |刘清泉   |男      |1998-06-21 |计算机     |52   |
|196002   |张慧玲   |女      |1999-11-07 |计算机     |50   |
|191003   |冯涛     |男      |1999-08-12 |计算机     |52   |
|196001   |董明霞   |女      |1999-05-02 |通信       |48   |
|196002   |李茜     |女      |1998-07-25 |通信       |52   |
|196004   |周俊文   |男      |1998-03-10 |通信       |50   |
+-------+--------+-------+-----------+---------+---+
6 rows in set (0.00 sec)
```

5.1.4 REPLACE 语句

REPLACE 语句的语法格式与 INSERT 语句基本相同，当存在相同的记录时，REPLACE 语句可以在插入数据之前将与新记录冲突的旧记录删除，使新记录能够正常插入。

【例 5.5】 对 student1 表，重新插入记录('196002','李茜','女','1998-07-25','通信',52)。

```
mysql > REPLACE INTO student1 VALUES
    -> ('196002','李茜','女','1998-07-25','通信',52);
```

执行结果：

```
Query OK, 1 row affected (0.04 sec)
```

5.1.5 插入查询结果语句

将已有表的记录快速插入当前表中，使用 INSERT INTO…SELECT…语句。其中，SELECT 语句返回一个查询结果集，INSERT 语句将这个结果集插入指定表中。

语法格式：

```
INSERT[INTO] table_name 1 (column_list1)
    SELECT(column_list2) FROM table_name e2 WHERE (condition)
```

其中，table_name 1 是待插入数据的表名，column_list1 是待插入数据的列名表；table_name 2 是数据来源表名，column_list2 是数据来源表的列名表；column_list2 列名表必须和 column_list1 列名表的列数相同，且数据类型匹配；condition 指定查询语句的查询条件。

【例 5.6】 向 student2 表插入 student 表的记录。

```
mysql > INSERT INTO student2
    -> SELECT * FROM student;
```

执行结果：

```
Query OK, 6 rows affected (0.06 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

5.2 修改数据

修改表中的一行或多行记录的列值使用 UPDATE 语句。

语法格式：

```
UPDATE table_name
    SET column1 = value1[, column2 = value2, … ]
    [WHERE < condition >]
```

说明：

（1）SET 子句：用于指定表中要修改的列名及其值，column1，column2，…为指定修改的列名，value1，value2，…为相应的指定列修改后的值。

（2）WHERE 子句：用于限定表中要修改的行，condition 指定要修改的行满足的条件，若语句中不指定 WHERE 子句，则修改所有行。

注意：UPDATE 语句修改的是一行或多行中的列。

5.2.1 修改指定记录

修改指定记录需要通过 WHERE 子句指定要修改的记录满足的条件。

【例 5.7】 在 student1 表中，将学生周俊文的出生日期改为“1999-03-10”。

```
mysql > UPDATE student1
    - >     SET sbirthday = '1999 - 03 - 10'
    - >     WHERE sname  = '周俊文';
```

执行结果：

```
Query OK, 1 row affected (0.07 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

使用 SELECT 语句查询修改指定记录后的数据：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
|196001 |董明霞  |女     |1999 - 05 - 02|通信   |48 |
|196002 |李茜    |女     |1998 - 07 - 25|通信   |52 |
|196004 |周俊文  |男     |1998 - 03 - 10|NULL   |50 |
+-------+--------+-------+-----------+----------+---+
3 rows in set (0.00 sec)
```

5.2.2 修改全部记录

修改全部记录不需要指定 WHERE 子句。

【例 5.8】 在 student1 表中，将所有学生的学分增加 2 分。

```
mysql > UPDATE student1
    - >     SET tc = tc + 2;
```

执行结果：

```
Query OK, 3 rows affected (0.10 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

使用 SELECT 语句查询修改全部记录后的数据：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+--------+---------+--------+------------+----------+---+
|sno     |sname    |ssex    |sbirthday   |speciality|tc |
+--------+---------+--------+------------+----------+---+
|196001  |董明霞   |女      |1999-05-02  |通信      |50 |
|196002  |李茜     |女      |1998-07-25  |通信      |54 |
|196004  |周俊文   |男      |1999-03-10  |NULL      |52 |
+--------+---------+--------+------------+----------+---+
3 rows in set (0.00 sec)
```

5.3　删 除 数 据

删除表中的一行或多行记录使用 DELETE 语句。

语法格式：

```
DELETE FROM table_name
    [WHERE < condition >]
```

其中，table_name 是要删除数据的表名，WHERE 子句是可选项，用于指定表中要删除的行，condition 指定删除条件，若省略 WHERE 子句，则删除所有行。

注意：DELETE 语句删除的是一行或多行。如果删除所有行，表结构仍然存在，即存在一个空表。

5.3.1　删除指定记录

删除指定记录需要通过 WHERE 子句指定表中要删除的行所满足的条件。

【例 5.9】 在 student1 表中，删除学号为“196004”的行。

```
mysql > DELETE FROM student1
    ->     WHERE sno = '196004';
```

执行结果：

```
Query OK, 1 row affected (0.02 sec)
```

使用 SELECT 语句查询删除一行后的数据：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
+-------+--------+-------+-----------+----------+---+
|sno    |sname   |ssex   |sbirthday  |speciality|tc |
+-------+--------+-------+-----------+----------+---+
|196001 |董明霞  |女     |1999-05-02 |通信      |50 |
|196002 |李茜    |女     |1998-07-25 |通信      |54 |
+-------+--------+-------+-----------+----------+---+
2 rows in set (0.00 sec)
```

5.3.2 删除全部记录

删除全部记录有两种方式：一种方式是通过 DELETE 语句并省略 WHERE 子句，则删除表中所有行，仍保留表的定义在数据库中。另一种方式是通过 TRUNCATE 语句，则删除原来的表并重新创建一个表。

1. DELETE 语句

省略 WHERE 子句的 DELETE 语句，用于删除表中所有行，而不删除表的定义。

【例 5.10】 在 student1 表中，删除所有行。

```
mysql > DELETE FROM student1;
```

执行结果：

```
Query OK, 2 rows affected (0.07 sec)
```

使用 SELECT 语句进行查询：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
Empty set (0.00 sec)
```

2. TRUNCATE 语句

TRUNCATE 语句用于删除原来的表并重新创建一个表，而不是逐行删除表中记录，执行速度比 DELETE 语句快。

语法格式：

```
TRUNCATE[TABLE] table_name
```

其中，table_name 是要删除全部数据的表名。

【例 5.11】 在 student 表中，删除所有行。

```
mysql > TRUNCATE student;
```

执行结果：

```
Query OK, 0 rows affected (0.21 sec)
```

使用 SELECT 语句进行查询：

```
mysql > SELECT * FROM student1;
```

查询结果：

```
Empty set (0.01 sec)
```

5.4 小 结

本章主要介绍了以下内容。

(1) 插入数据的语句有：INSERT 语句、REPLACE 语句和插入查询结果语句。

INSERT 语句用于向数据库的表插入一行或多行数据，可为表的所有列插入数据，也可为表的指定列插入数据和插入多行数据。

当存在相同的记录时，REPLACE 语句可以在插入数据之前将与新记录冲突的旧记录删除，使新记录能够正常插入。

将已有表的记录快速插入当前表中，可以使用 INSERT INTO…SELECT…语句。

(2) 修改表中的一行或多行记录的列值使用 UPDATE 语句。

修改指定记录需要通过 WHERE 子句指定要修改的记录满足的条件，修改全部记录不需要指定 WHERE 子句。

(3) 删除表中的一行或多行记录使用 DELETE 语句。

删除指定记录需要通过 DELETE 语句的 WHERE 子句指定表中要删除的行所满足的条件。

删除全部记录有两种方式：一种方式是通过 DELETE 语句并省略 WHERE 子句，则删除表中所有行，仍保留表的定义在数据库中。另一种方式是通过 TRUNCATE 语句，则删除原来的表并重新创建一个表。

习 题 5

一、选择题

5.1 表数据操作的基本语句不包括(　　)。

A. INSERT　　B. DROP

C. UPDATE　　D. DELETE

5.2 删除表的全部记录采用(　　)。

A. DELETE　　B. TRUNCATE

C. A 和 B 选项　　D. INSERT

5.3 以下语句无法添加记录的是(　　)。

A. INSERT INTO…UPDATE…　　B. INSERT INTO…SELECT…

C. INSERT INTO…SET…　　D. INSERT INTO…VALUES…

5.4 快速清空表中的记录可采用(　　)语句。

A. DELETE　　B. TRUNCATE

C. CLEAR TABLE　　D. DROP TABLE

5.5 (　　)字段可以采用默认值。

A. 出生日期　　B. 姓名

C. 专业　　D. 学号

二、填空题

5.6 插入数据的语句有_____语句和 REPLACE 语句。

5.7 将已有表的记录快速插入当前表中，可以使用_____ 语句。

5.8 插入数据时不指定列名，要求必须为每个列都插入数据，且值的顺序必须与表定义的列的顺序 _____。

5.9 VALUES 子句包含了_____需要插入的数据，数据的顺序要与列的顺序相对应。

5.10 为表的指定列插入数据，在插入语句中，除给出了部分列的值外，其他列的值为表定义时的默认值或允许该列取 _____。

5.11 当存在相同的记录时，REPLACE 语句可以在插入数据之前将与新记录冲突的旧记录 _____，使新记录能够正常插入。

5.12 插入多条记录时，在插入语句中只需指定多个插入值列表，插入值列表之间用_____隔开。

5.13 修改表中的一行或多行记录的_____ 使用 UPDATE 语句。

5.14 修改指定记录需要通过 WHERE 子句指定要修改的记录满足的_____。

5.15 删除全部记录有两种方式：一种方式是通过 DELETE 语句并省略 WHERE 子句，另一种方式是通过_____语句。

三、问答题

5.16 简述插入数据所使用的语句。

5.17 比较插入列值使用的两种方法：不指定列名和指定列名。

5.18 修改数据有哪两种方法？

5.19 比较删除数据使用的两种方法：删除指定记录和删除全部记录。

5.20 删除全部记录有哪两种方式？各有何特点？

四、应用题

5.21 向课程表(course)插入样本数据，参见附录 B。

5.22 使用 INSERT INTO…SELECT…语句，将 course 表的记录快速插入 course1 表中。

5.23 采用三种不同的方法，向 course2 表插入数据。

(1) 省略列名表，插入记录('1004','数据库系统',4)。

(2) 不省略列名表，插入课程号为“1017”、学分为 3 分、课程名为“操作系统”的记录。

(3) 插入课程号为“4002”，课程名为“数字电路”，学分为空的记录。

5.24 在 course1 表中，将课程名“操作系统”改为“计算机网络”。

5.25 在 course1 表中，将课程号 1201 的学分改为 3 分。

5.26 在 course1 表中，删除课程名为“高等数学”的记录。

5.27 采用两种不同的方法，删除表中的全部记录。

(1) 使用 DELETE 语句，删除 course1 表中的全部记录。

(2) 使用 TRUNCATE 语句，删除 course2 表中的全部记录。

5.28 分别向成绩表(score)、教师表(teacher)、讲课表(lecture)插入样本数据，参见附录 B。