1 关系型数据库系统

关系型数据库由数据库、存储和管理数据库中数据的软件，以及显示数据并使用用户能够与数据库系统进行交互的应用程序组成。

目前大多数数据库系统都是关系数据库系统，它们都是基于关系数据模型，有三个要素：**结构**、**完整性**、**语言**。结构定义了数据的表示，完整性给出了一些对于数据的约束，语言提供了访问和操纵数据的手段。

表的一行表示一条记录，称为一个元组；表的一列表示该记录中一个属性的值，一列称为一个属性。

完整性约束

一般来说，有三种类型约束：域约束、主键约束和外键约束。域约束和主键约束是内部关联型约束，意味着每个约束只涉及一个关系，外键约束是相互关联型的，意味着一个涉及多个关系约束。

域约束规定一个属性的允许值。

超键是一个属性或一组属性，它唯一标识了一个关系。也就是说没有量过记录具有相同的超键值。

键是一个最小的超键，一个关系可以有几个键，每个键都称为一个候选键。主键是由数据库设计者指定的一个用来标识一个关系中的记录的候选键。

外键约束：在关系数据库中，数据是相互关联的。关系中的记录是相互关联的，而不同关系中的记录通过他们的共同属性也是相互关联的。简单来说，共同属性就是外键，外键约束定义了关系之间的关系。

复习题

1.不再累述
2.不再累述
3.主键只能有一个，外键可以有多个。
4.在同一关系中，外键必须是主键。
5.一个外键需要与它的引用主键不一定具有相同名字（**存疑**）
6.外键的值可以为空。

2 SQL

在 Mysql 上创建用户

在 windows 中，输入命令 net stop mysql 可以终止它，输入命令 net start mysql 可以启动

它。

创建数据库

```
create database basename
```

这个语句是创建一个名为 basename 的数据库。

创建和删除表

```
create table Course(
    courseId char(5),
    subjectId char(4) not null,
    courseNumber integer,
    title varchar(50) not null,
    numOfCredits integer,
    primary key(courseId)
    );
```

删除表为 drop table Course。

为了防止出错，可以用文本文档写好代码，然后保存为 sql 结尾的文件。在命令行中使用 sourse dir/sqlname.sql 来导入 sql 文件。

简单插入、更新和删除

①在表中插入一条记录的语法是：

```
insert into tableName[(column1,column2,…,column)] values(values1,values2,…,valuen);
```

注意，列名是可选的。尽管列名有默认值，但是如果省略了列名，必须输入记录中所有列的值。

②更新表格的语法是：

```
update tableName
set column1 = newValue1[,其它属性]
[where condition];
```

③删除记录的语法是：

```
delete from tableName
[where condition];
```

若全部删除，则是 delete from tableName;

简单查询

```
select column-list
from table-list
[where condition];
```

select 子句列出所选定的列，from 子句指定查询所涉及的表。可选的 where 子句指明

选择执行的条件。

比较运算符和布尔运算符

比较运算符

运算符	描述	运算符	描述	运算符	描述
=	等于	<	小于	>	大于
<>或!=	不等于	<=	小于等于	>=	大于等于

布尔运算符

运算符	描述	运算符	描述	运算符	描述
not	逻辑非	and	逻辑与	or	逻辑或

注意：要选择表中所有属性，只需要使用一个星号（*）即可。

操作符 like、between-and 和 is null

like 是一个用于模式匹配的操作符。检验字符串 s 是否含有模式 p 的语法是：

s like p 或 s not like p

在模式 p 中可以使用通配符%和_。%匹配零个或多个字符，_与 s 中的任意单字符匹配。

v between v1 and v2 等价于 v>=v1 and v<=v2,。v not between v1 and v2 等价于 v<v1 or v>v2。

is null 检查 v 是否为 null。

列的别名

slesect columnName [as] alias

显示记录

①显示互不相同的记录

利用关键字 distinct，可以用于去除输出重复的元组。例如

select distinct subjectId as “Subject ID”

from Course;

当 select 子句中条目多余一项时，关键字 distinct 可以查找所有条目中相异记录。

②显示排好序的记录

SQL 提供对输出结果排序的 order by 子句，语法如下：

select column-list

from table-list

[where condition]

[order by columns-to-be-sorted];

在这个语法结构中，columns-to-be-sorted 指定要排序的一列或多了。默认情况下是升序。要按降序排列，要在 columns-to-be-sorted 后面附加关键字 **desc**。当指定多列时，**先按第一列对各行排序，然后对第一列具有相同值的行再按第二列排序，以此类推**。

联结表

举例：列出学生 Jacob Smith 所学的课程。要完成这个查询，需要将表 Student 和表 Enrollment 联结起来。

```
select distinct lastName, firstName,coursed
from Student Enrollment
where Student.ssn = Enrollment.ssn and
lastName = 'Smith' and firstName = 'Jacob';
```

3 JDBC

在使用 MySQL8.0 的情况下，Java 连接 MySQL 与以前的版本还有少许不一样。其主要区别在于以下几点：

（1）驱动程序名改为了：**com.mysql.cj.jdbc.Driver**。

（2）在写 URLs 的时候，要写成这样的格式：**jdbc:mysql://hostname/dbname?useSSL=false&serverTimezone=UTC**。也就是说将 SSL 关闭，将服务器时区设置为 UTC。

4 PreparedStatement

PreparedStatement 借口用于执行含有或不含有参数的静态 SQL 语句。PreparedStatement 对象是用 Connection 借口中的 preparedStatement 方法创建的。例如用一个 insert 创建一个 PreparedStatement 对象：

```
PreparedStatement preparedStatement = connection.prepareStatement("insert into Student(firstName,mi,lastName) " + "values(?,?,?)");
```

5 CallableStatement

CallableStatement 可以执行 SQL 存储过程。这个进程可能会有 IN/OUT/IN OUT 参数。当调用过程时，参数 IN 接受传递给过程的值；**在进程结束后，参数 OUT 返回一个值，但是当调用过程时，它不包含任何一个值**。当过程被调用时，IN OUT 参数包含传递给过程的值，在完成之后返回一个值。

既可以调用过程，也可以调用函数。它继承了 PreparedStatement。此外它提供了注册

OUT 参数的方法以及从 OUT 参数获取值的方法。

在调用 SQL 进程之前，需要使用合适的 setter 方法将值传给 IN 和 IN OUT 参数，使用 registerOutParameter 来注册 OUT 和 IN OUT 参数。

5 获取元数据

可以使用 DatabaseMetaData 接口获取数据库的元数据，例如数据库的 URL、用户名、JDBC 驱动程序名称等。格式如下：

```
DatabaseMetaData dbMetaData = connection.getMetaData();
```

ResultSetMetaData 接口哦用于获取到结果集合的元数据，如表的列数和列名等。格式如下：

```
ResultSetMetaData rsData = resultSet.getMetaData();
```