MySQL第二部分讲义

一 动手做（TODO）

（一） DML数据操作语言（TODO 11个任务）

> **DML用于更新改变数据库中的数据，主要包括INSERT、UPDATE和DELETE语句。 其中，INSERT语句用于将数据插入到数据库中，UPDATE语句用于更新数据库中已经存在的数据，DELETE语句则用于删除数据库中已经存在的数据。**

1. 插入数据--INSERT 操作

(1) 默认输入一条完整记录，并且输入值的顺序和表的字段顺序一致，使用格式1

语法格式1：INSERT INTO 表名 VALUES（值1，值2，值3，值4.....）；
注意：字符串和日期类型需要用单引号

- [x] 任务1：向班级表t_class里插入一条完整记录

```
INSERT INTO t_class VALUES(1, 'Java全栈1018班');
```

(2) 输入一条不完整的记录，就需要用到格式2，明确的指定出字段名和值，并且前后顺序一致 语法格式2：INSERT INTO 表名 （字段名1，字段名2，字段名3，字段名4） VALUES（值 1，值2，值3，值4）；

- [x] 任务2： 向学生表里插入一个不完整记录。

```
INSERT INTO t_student (sid,sname)   VALUES('yc002','张华');
注意：非空字段必须选择插入数据
```

(3) 在插入中使用默认值DEFAULT
如果有默认值，就会使用默认值；如果没有默认值就使用NULL

- [x] 任务3： 向学生表里插入一条记录，性别使用默认值插值。

```
INSERT INTO  t_student(sid,sname,sex)  VALUES('yc010','王小二', DEFAULT);
```

（4） 一次插入多条记录 语法格式3：INSERT INTO 表名 （字段名1，字段名2，字段名3，字段名4） VALUES（值1，值2，值3，值4），（值 1，值2，值3，值4），（值 1，值2，值3，值4）...;

- [x] 任务4： 向学生表里插入多条记录。

```
INSERT INTO  t_student(sid,sname,sex)  VALUES('yc110','张三', '男'),('yc111','李四',
'男'),('yc112','王五', '女');
```

（5） 如果主键设置了自动增长，主键可以插入NULL,实际插入表里的数据的主键值已经自动增长，有值了。但是语法不会因为主键插入的NULL而报错。

- [x] 任务5：向班级表t_class里插入一条记录(班级表的主键cno设置了auto_increment)

```
INSERT INTO t_class VALUES(NULL, 'Java全栈1017班');
```

(6) 批量插入数据，将一个表的数据批量插入到另一个表中 语法格式4：INSERT INTO 表名 （字段名1，字段名2...） SELECT 字段名1，字段名2 FROM 表名

- [x] 任务6： 将学生表t_student的数据插入到t_stu表中。

```
INSERT INTO  t_stu(s_id,s_name,s_sex)  SELECT sno,sname,sex FROM t_student ;
```

注意:t_student表的sno,sname,sex和t_stu的s_id,s_name,s_sex的数据类型要一一对应上。

2. 修改更新数据--UPDATE操作

- 语法格式1—更新单列： （更新了所有行的某一列）
  UPDATE 表名 SET 字段名 = 新值;

- [x] 任务7、将所有学生的名字改成张三峰。

```
UPDATE  t_student SET  sname = '张三峰';
```

- 语法格式2—更新多列：UPDATE 表名 SET 字段名1 = 新值1，字段名2 = 新值2，字段名3 = 新值3;

- [x] 任务8、将所有学生的电话改成13245678900，邮箱改成67888995@qq.com

```
UPDATE  t_student  SET  telephone='13245678900', email = '67888995@qq.com' ;
```

注意：对于更新所有行记录(风险大，不建议使用，还有比如所有员工设置密码)

- 语法格式3—更新指定记录：UPDATE 表名 SET 字段名 = 新值 WHERE 条件;

- [x] 任务9 更新t_student表中学号为"yc1004"的学生的记录(姓名改为"王祖蓝",手机号改为"18406072416")。

```
UPDATE t_student
SET sname='王祖蓝',telephone='18406072416'
WHERE  sno='yc1004';
```

- **注意：**

> 1）数字不用单引号，日期和字符必须使用单引号
> 2）更新数据时，数据必须与列的数据类型匹配
> 3）更新数据时，数据必须要满足约束规则

3. 删除数据--DELETE操作

- 格式1—删除所有记录：
  DELETE FROM 表名;

  - [x] 任务10 删除t_student的所有数据

```
DELETE FROM t_student;
```

- 格式2—删除指定记录：DELETE FROM 表名 WHERE 条件;

  - [x] 任务11 删除学号为yc1001的学生记录

```
DELETE FROM t_student WHERE sno='yc1001' ;
```

（二） TCL事务控制语言（TODO 2个任务）

- 任务1：事务处理的方法1：用 BEGIN, ROLLBACK, COMMIT来实现

```
mysql> use myschool;
Database changed
mysql> CREATE TABLE transaction_test( id int(5) primary key ) engine=innodb;  # 创建数据表
Query OK, 0 rows affected (0.04 sec)

mysql> select * from transaction_test;
Empty set (0.01 sec)

mysql> begin;  # 开始事务
Query OK, 0 rows affected (0.00 sec)

mysql> insert into transaction_test values(5);
Query OK, 1 rows affected (0.01 sec)

mysql> insert into transaction_test values(6);
Query OK, 1 rows affected (0.00 sec)

mysql> commit; # 提交事务
Query OK, 0 rows affected (0.01 sec)

mysql>  select * from transaction_test;
+------+
| id   |
+------+
| 5    |
| 6    |
+------+
2 rows in set (0.01 sec)
```

```
mysql> begin;    # 开始事务
Query OK, 0 rows affected (0.00 sec)

mysql>  insert into transaction_test values(7);
Query OK, 1 rows affected (0.00 sec)

mysql> rollback;   # 回滚
Query OK, 0 rows affected (0.00 sec)

mysql>   select * from transaction_test;   # 因为回滚所以数据没有插入
+------+
| id   |
+------+
| 5    |
| 6    |
+------+
2 rows in set (0.01 sec)

mysql>
```

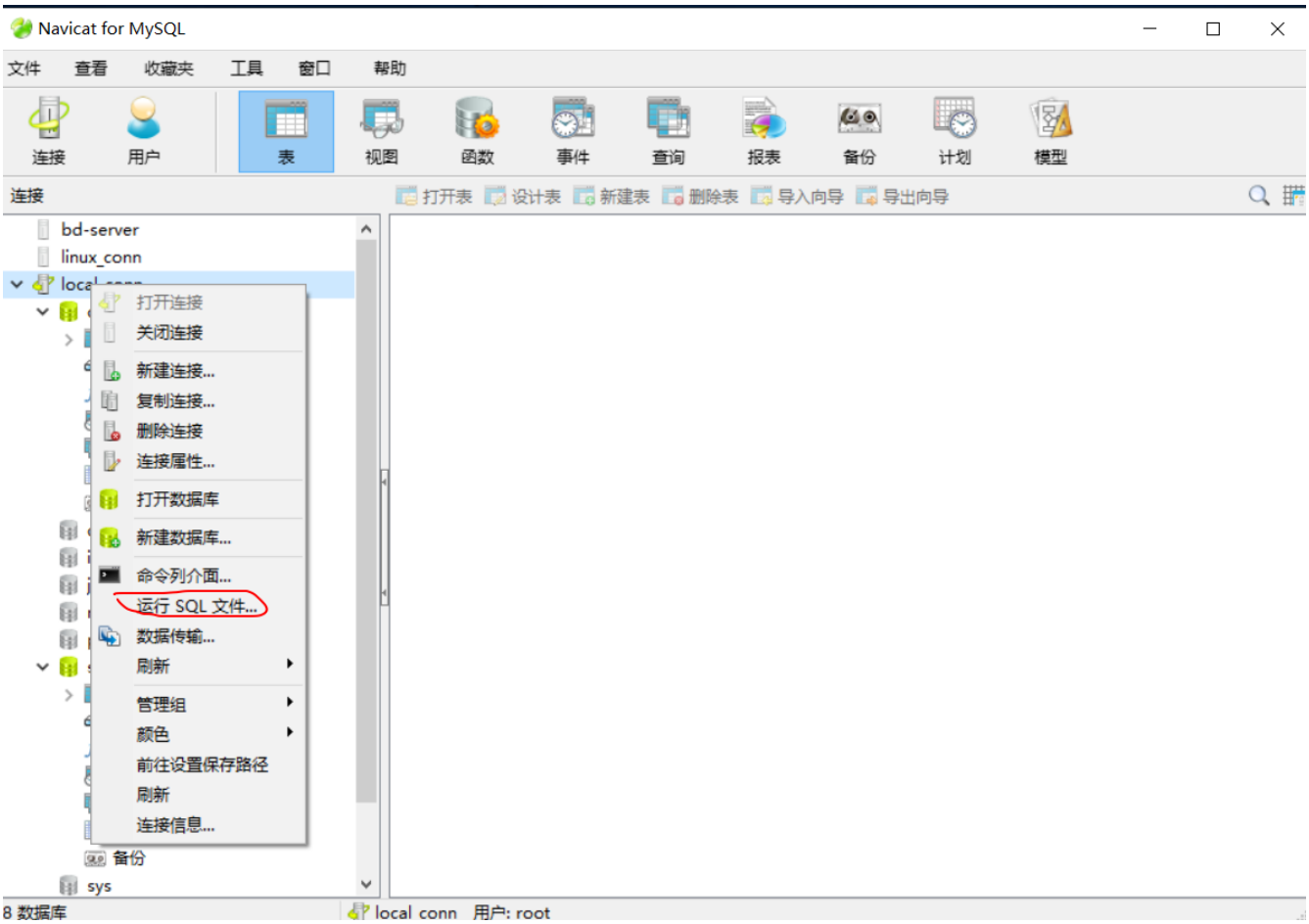
- 任务2 事务处理的第二种方法，直接用 SET 来改变 MySQL 的自动提交模式。

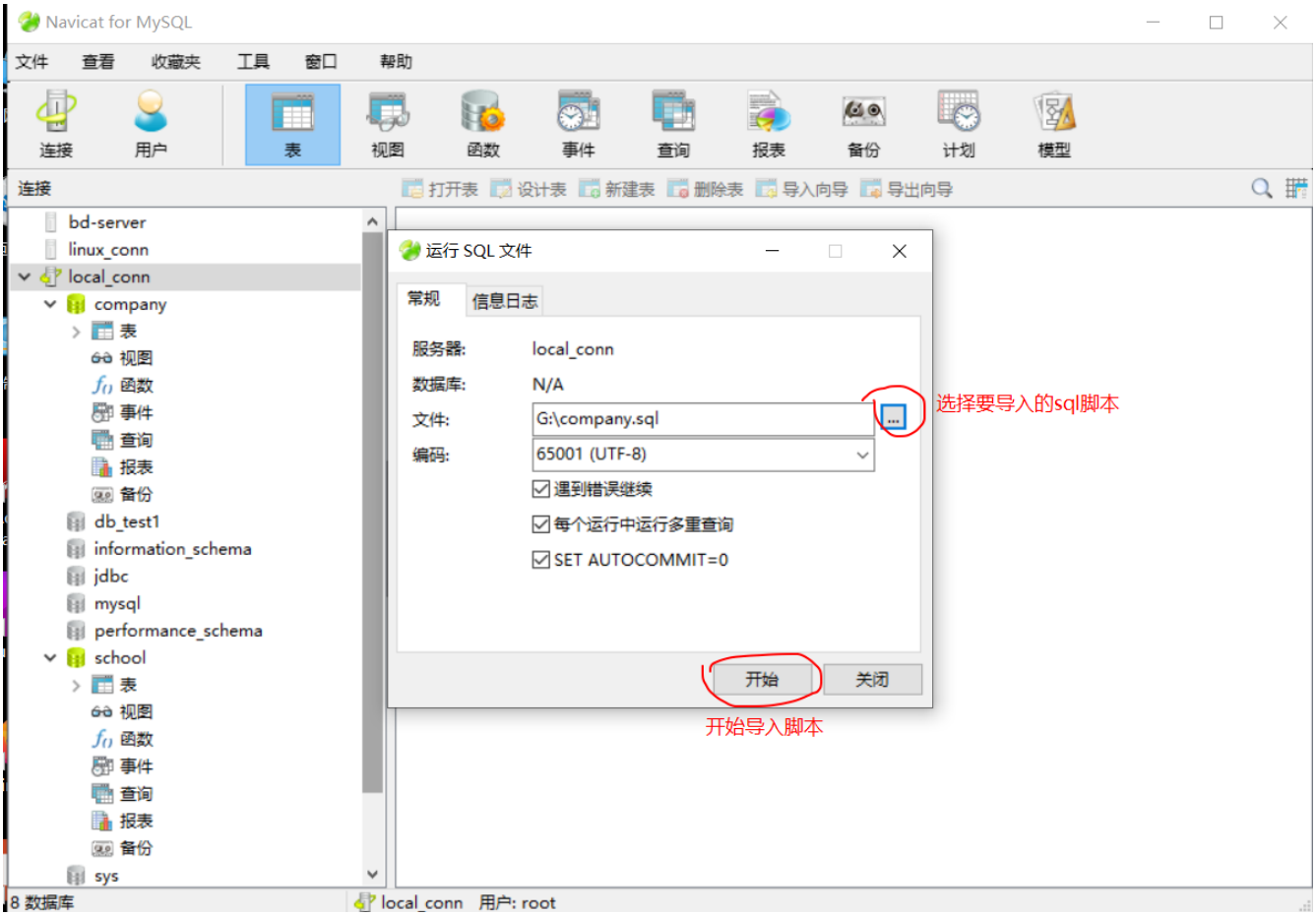
```
mysql> use myschool;
Database changed
mysql> CREATE TABLE transaction_test2( id int(5) primary key ) engine=innodb;  # 创建数据表
Query OK, 0 rows affected (0.04 sec)

mysql> select * from transaction_test;
Empty set (0.01 sec)

mysql> SET AUTOCOMMIT=0   ##禁止自动提交
Query OK, 0 rows affected (0.00 sec)

mysql> insert into transaction_test values(6);
Query OK, 1 rows affected (0.00 sec)

mysql> commit; # 提交事务
Query OK, 0 rows affected (0.01 sec)

mysql>  select * from transaction_test;
+------+
| id   |
+------+
| 5    |
| 6    |
+------+
2 rows in set (0.01 sec)
```

（三） 如何使用source命令导入SQL脚本?

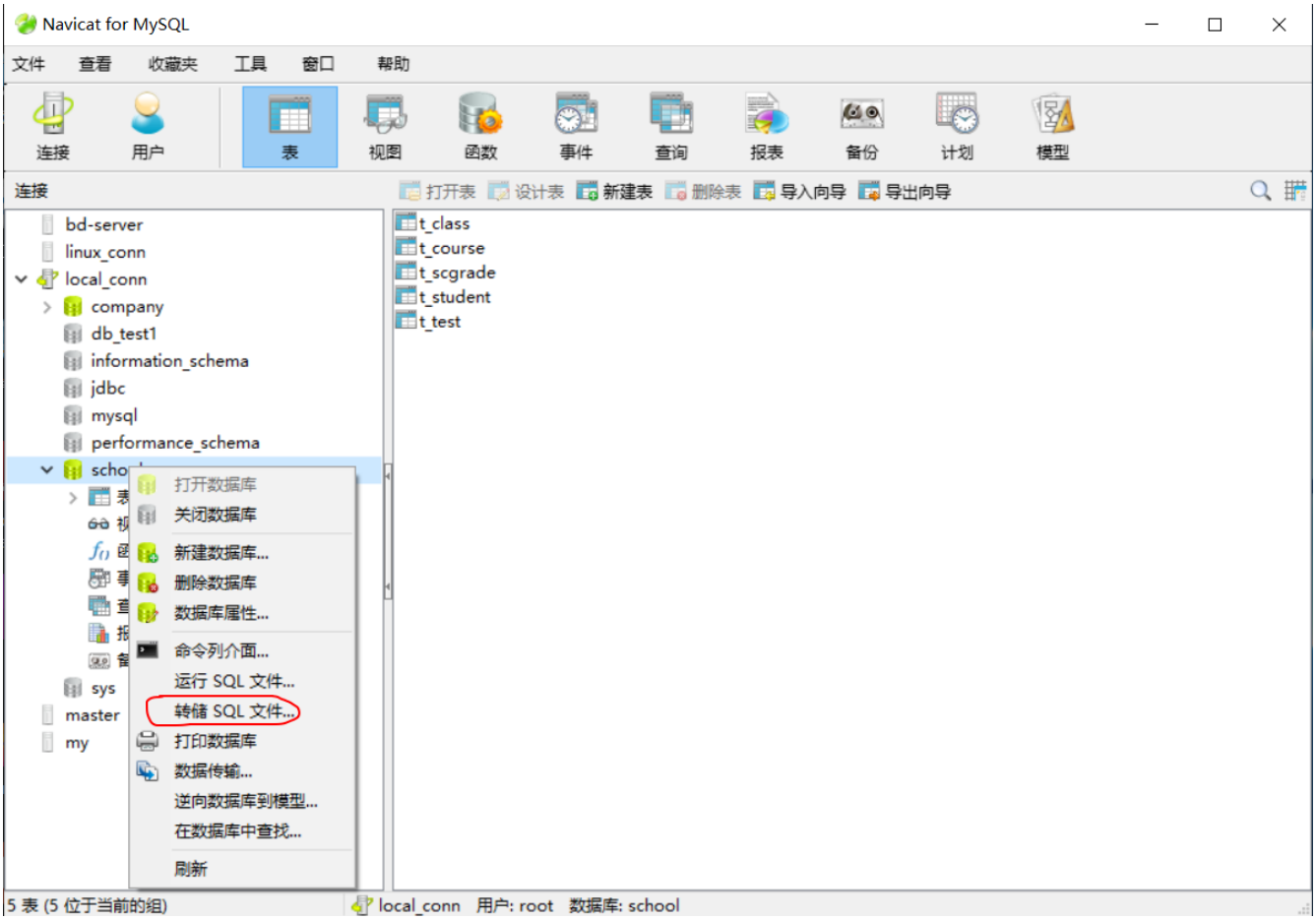
> 在cmd窗口登录MySQL后输入下面命令 命令格式： source 盘符：\脚本名称.sql 注意：不能加分号结尾 例如： source company.sql

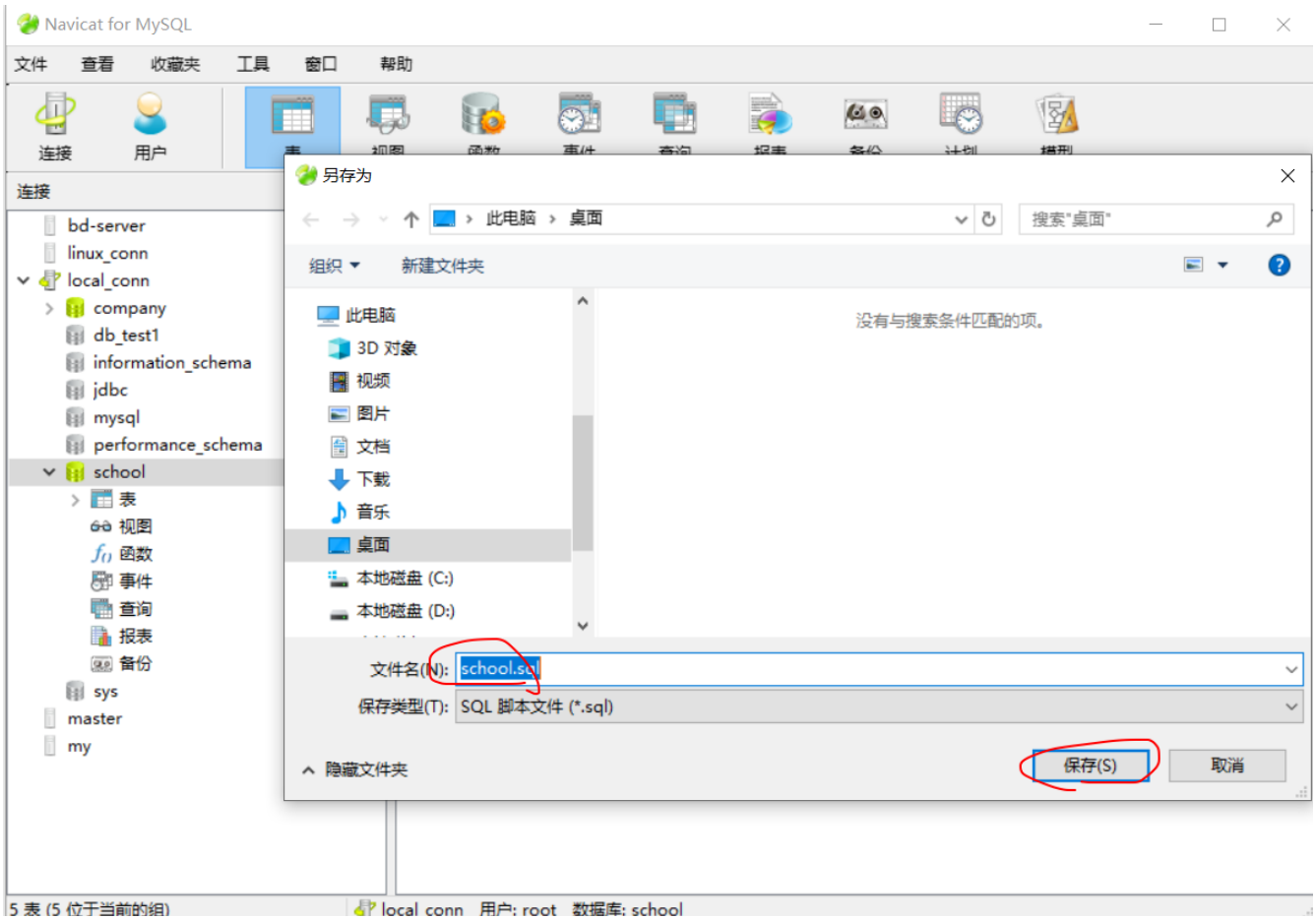
（四） 如何使用Navicat导入SQL脚本?

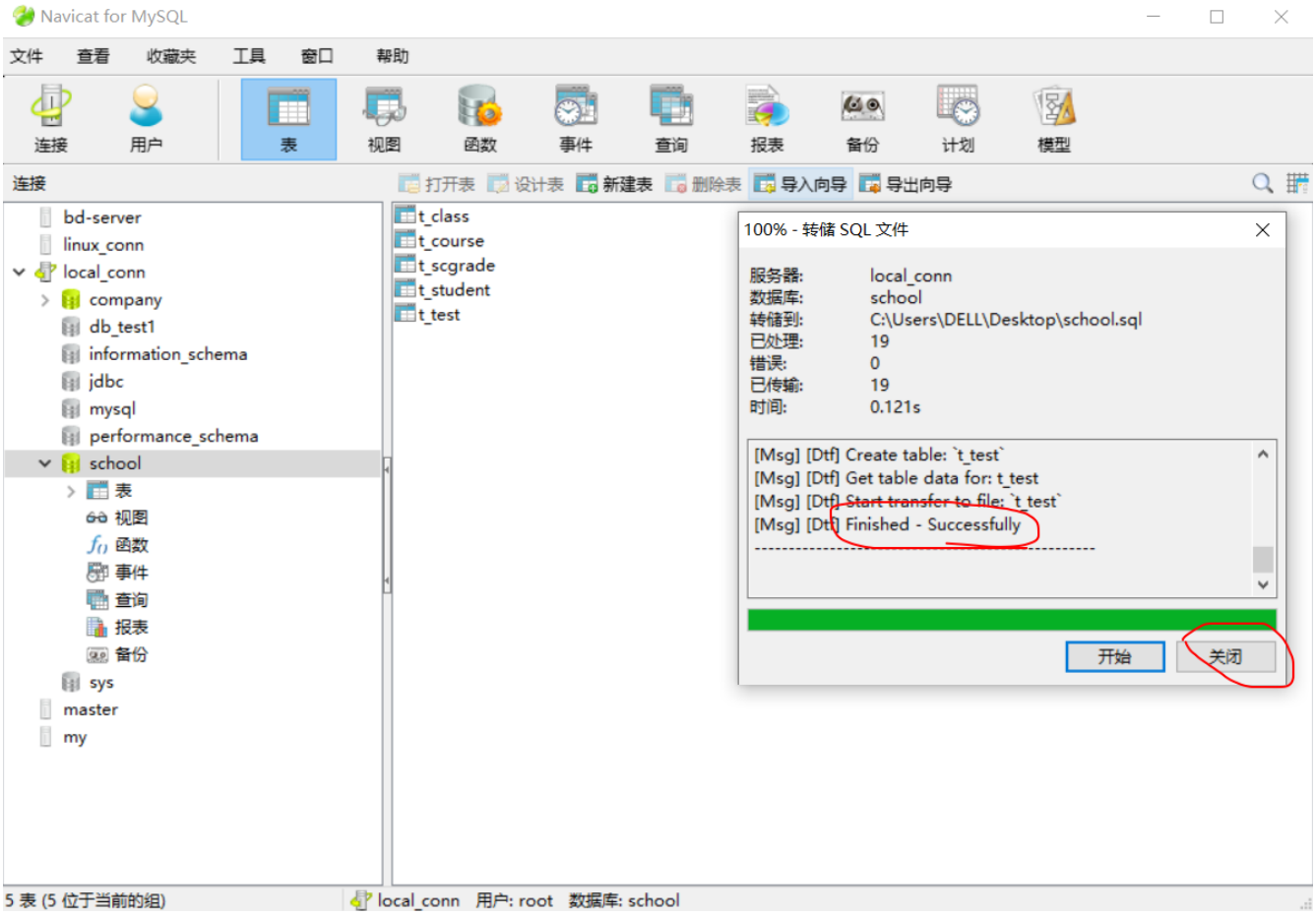




（五） 如何使用Navicat导出SQL脚本（备份数据）？


Navicat for MySQL
文件 查看 收藏夹 工具 窗口 帮助
连接 用户 表 视图 函数 事件 查询 报表 备份 计划 模型
连接
打开表 设计表 新建表 删除表 导入向导 导出向导
bd-server
linux_conn
local_conn
company
db_test1
information_schema
jdbc
mysql
performance_schema
school
sys
master
my
t_class
t_course
t_scgrade
t_student
t_test
打开数据库
关闭数据库
新建数据库...
删除数据库
数据库属性...
命令列介面...
运行 SQL 文件...
转储 SQL 文件...
打印数据库
数据传输...
逆向数据库到模型...
在数据库中查找...
刷新
5 表 (5 位于当前的组)
local_conn 用户: root 数据库: school


Navicat for MySQL
文件 查看 收藏夹 工具 窗口 帮助
连接 用户
另存为
此电脑 › 桌面
搜索"桌面"
组织 新建文件夹
此电脑
3D 对象
视频
图片
文档
下载
音乐
桌面
本地磁盘 (C:)
本地磁盘 (D:)
没有与搜索条件匹配的项。
文件名(N): school.sql
保存类型(T): SQL 脚本文件 (*.sql)
隐藏文件夹
保存(S)
取消
5 表 (5 位于当前的组)
local_conn 用户: root 数据库: school



二 理解并口述（技术点和面试题）

【口述1】.DELETE与TRUNCATE的区别？ （面试题）

> 相同点：DELETE FROM 表名 与 TRUNCATE TABLE 表名都可以删除数据表里数据，删除数据后表结构还在
> 不同点：

（1）. TRUNCATE TABLE属于DDL语言，自动提交 ；而DELETE FROM 属于DML语言，如果开启事务手动提交，DML语言需要COMMIT提交。

（2）. DELETE 可以带WHERE条件删除指定记录，但是TRUNCATE不可以删除指定记录，而是全部删除。

（3）. DELETE删除数据后不释放表空间，但是TRUNCATE删除数据后会释放数据所在的表空间，但索引等所占用的表空间不会释放，需要通过命令去释放

（4）. 删除数据表所有数据时，TRUNCATE 的操作并不记录到日志，TRUNCATE TABLE 的速度比DELETE快

（5）. DELETE 删除数据后只要没有COMMIT，就可以回滚ROLLBACK，但是使用TRUNCATE TABLE不会对事务有影响，TRUNCATE 删除数据后不可以ROLLBACK回滚

（6）. TRUNCATE可以删除视图，但是DELETE不可以删除视图

（7）. 使用TRUNCATE TABLE重新设置AUTO_INCREMENT计数器，DELETE删除数据后，不会重新计数。

【口述2】什么是事务?

- MySQL事务主要用于处理操作量大，复杂度高的数据。比如说，在人员管理系统中，你删除一个人员，你既需要删除人员的基本资料，也要删除和该人员相关的信息，如信箱，文章等等，这样，这些数据库操作语句就构成一个事务！
- 在 MySQL 中只有使用了 Innodb 数据库引擎的数据库或表才支持事务。
- 事务处理可以用来维护数据库的完整性，保证成批的SQL语句要么全部执行，要么全部不执行。
- 事务用来管理 insert,update,delete 语句

- 事务是对数据库操作的逻辑单位，在一个事务中可以包含一条或多条DML （数据操纵语言）、DDL （数据定义语言）和DCL （数据控制语言）语句，这些语句组成一个逻辑整体。
- 在关系数据库中，一个事务可以是一条SQL语句，一组SQL语句或整个程序。
- 事务是恢复和并发控制的基本单位。

【口述3】事务的操作?

- 事务的执行只有两种结果：要么全部执行，把数据库带入一个新的状态，要么全部不执行，对数据库不做任何修改。
- 对事务的操作有两个：提交（ COMMIT ）和回滚（ ROLLBACK ）。

> **注意：在 MySQL 命令行的默认设置下，事务都是自动提交的，即执行 SQL 语句后就会马上执行 COMMIT 操作。因此要显式地开启一个事务务须使用命令 BEGIN 或 START TRANSACTION，或者执行命令 SET AUTOCOMMIT=0，用来禁止使用当前会话的自动提交。**
>
> MYSQL 事务处理主要有两种方法： 1、用 BEGIN, ROLLBACK, COMMIT来实现 BEGIN 开始一个事务 ROLLBACK 事务回滚 COMMIT 事务确认 2、直接用 SET 来改变 MySQL 的自动提交模式: SET AUTOCOMMIT=0 禁止自动提交 SET AUTOCOMMIT=1 开启自动提交

【口述4】事务的特性ACID（面试题）

一般来说，事务是必须满足4个条件（ACID）：：原子性（Atomicity，或称不可分割性）、一致性（Consistency）、隔离性（Isolation，又称独立性）、持久性（Durability）。

- 事务的特性ACID：
  1. 原子性（Atomicity):事务必须是原子工作单元，一个事务（transaction）中的所有操作，对于其数据修改，要么全部执行，要么全都不执行。
  2. 一致性（Consistency):事务在完成时，必须使所有的数据都保持一致状态。在事务开始之前和事务结束以后，数据库的完整性没有被破坏。这表示写入的资料必须完全符合所有的预设规则，这包含资料的精确度、串联性以及后续数据库可以自发性地完成预定的工作。
  3. 隔离性(Isolation):由并发事务所作的修改必须与任何其它并发事务所作的修改隔离。 数据库允许多个并发事务同时对其数据进行读写和修改的能力，隔离性可以防止多个事务并发执行时由于交叉执行而导致数据的不一致。事务隔离分为不同级别，包括读未提交（Read uncommitted）、读提交（read committed）、可重复读（repeatable read）和串行化（Serializable）。
  4. 持久性（Durability):事务完成之后，它对系统的影响是永久性的。事务处理结束后，对数据的修改就是永久的，即便系统故障也不会丢失。
- 在企业应用中，多个事务并发所可能存在的问题如下：
  1. 脏读：一个事务读到另一个事务未提交的更新数据。
  2. 不可重复读：一个事务两次读同一行数据，可是这两次读到的数据不一样。
  3. 幻读：一个事务执行两次查询，但第二次查询比第一次查询多出了一些数据行。
  4. 丢失更新：撤销一个事务时，把其他事务已提交的更新的数据覆盖了。

三 课后任务（TODO）

任务：完成"驱动项目"UOL联合开放实验室信息管理项目系统所有表的数据插入，添加数据，每个数据表至少插入5-10条数据，事务设置为禁止自动提交，使用事务提交。