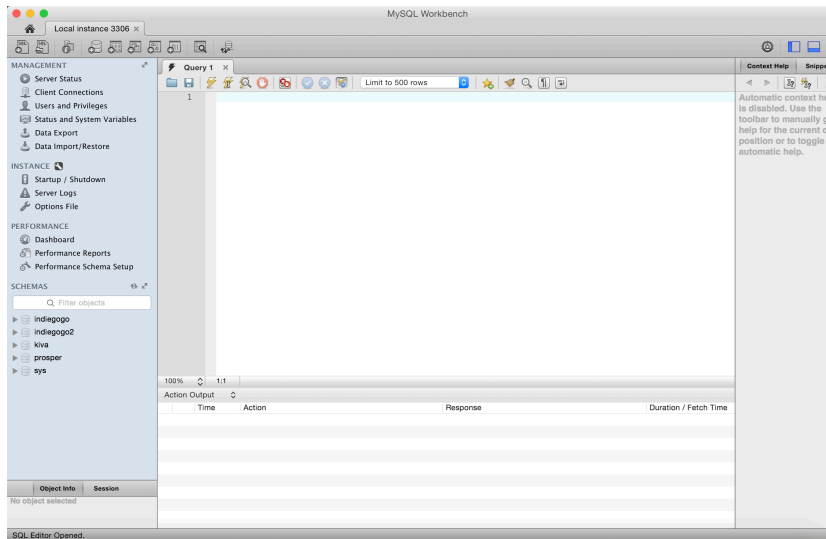


Create a simple database with MySQL

1. Connect the MySQL server through MySQL Workbench

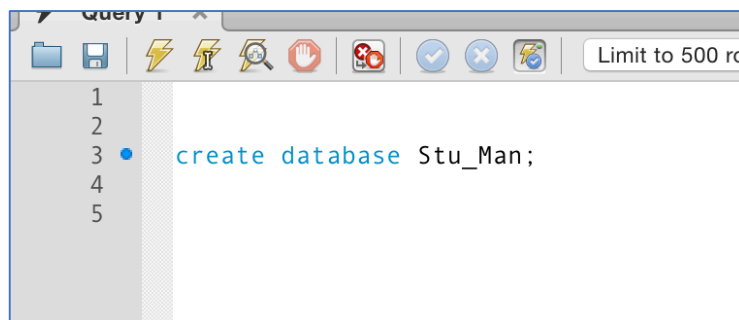


You can achieve many database operations by typing the SQL language into the Query panel, such as creating a database, creating tables, selecting, updating, dropping tables or databases. This document is going to create a simple database instance for student management system in university.

2. Create a database

type the following SQL and click the execute (the third icon), Stu_Man is the name of created database. You will find the created database on the 'schemas' panel (left, if not, please refresh).

create database Stu_Man;



3. Create tables

MySQL is a typical relational database, which contains two basic concepts, entity and Relation (ER). Both entity and relation are described as tables, which are organized into rows and columns. All the data referenced by an attribute are in the same domain and conform to the same constraints.

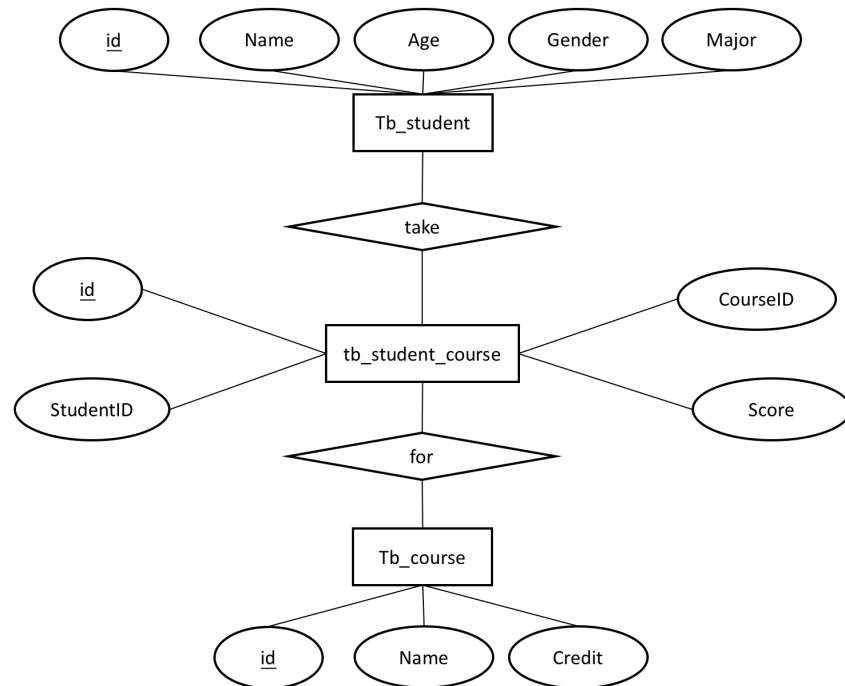
This instance contains three tables, that are tb_student (student information), tb_course (course information), and tb_student_course (the scores of student for course). We can describe them using the following ER-Diagram (https://en.wikipedia.org/wiki/Entity-relationship_model)

Rectangles: entities,

Diamonds: relationships

Circles: attributes

Underline means this attribute is the primary key which is the uniquely identifying attribute.



Create these tables by executing the following SQLs:

```
create table Stu_Man.tb_student
```

```
(
    id int(11) not null auto_increment primary key,
    Name varchar(32) not null,
    Age int default 0,check(Age>0 and Age<=100),
    Gender boolean default 0,check(Gender=0 or Gender=1),
    Major varchar(32) not null
);
```

```
create table Stu_Man.tb_course
```

```
(
    id int(5) not null auto_increment primary key,
    Name varchar(32) not null,
```

Credit int(2)

);

create table Stu_Man.tb_student_course

(

id int(5) not null auto_increment primary key,

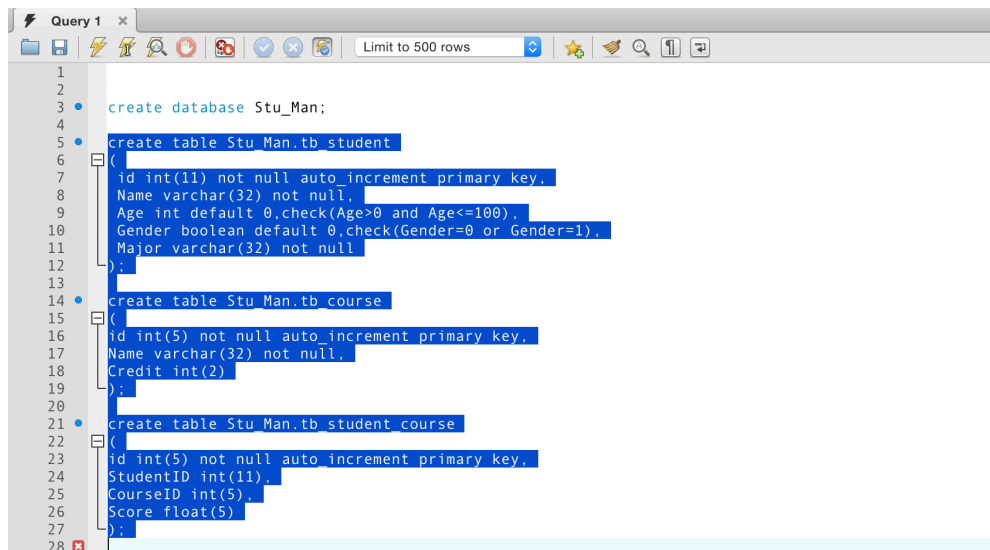
StudentID int(11),

CourseID int(5),

Score float(5)

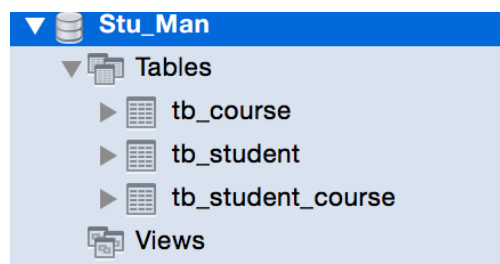
);

tips: you can execute specific SQLs with selecting them, otherwise workbench will execute all the current SQLs in the query panel.



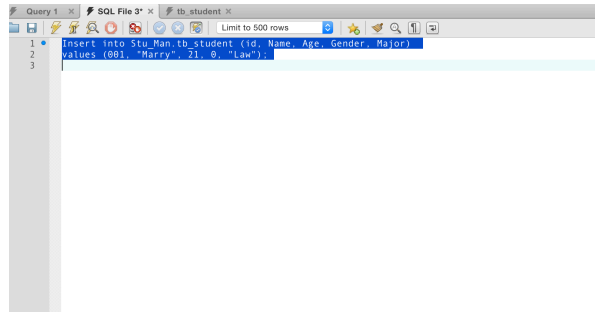
```
1  
2  
3 • create database Stu_Man;  
4  
5 • create table Stu_Man.tb_student  
6 (  
7   id int(11) not null auto_increment primary key,  
8   Name varchar(32) not null,  
9   Age int default 0, check(Age>0 and Age<=100),  
10  Gender boolean default 0, check(Gender=0 or Gender=1),  
11  Major varchar(32) not null  
12 );  
13  
14 • create table Stu_Man.tb_course  
15 (  
16  id int(5) not null auto_increment primary key,  
17  Name varchar(32) not null,  
18  Credit int(2)  
19 );  
20  
21 • create table Stu_Man.tb_student_course  
22 (  
23  id int(5) not null auto_increment primary key,  
24  StudentID int(11),  
25  CourseID int(5),  
26  Score float(5)  
27 );  
28
```

you will see your created tables (refresh):



4.Import or insert (add) data into tables

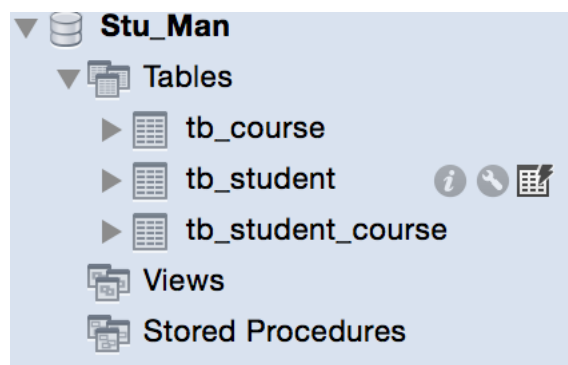
Insert by executing SQL:



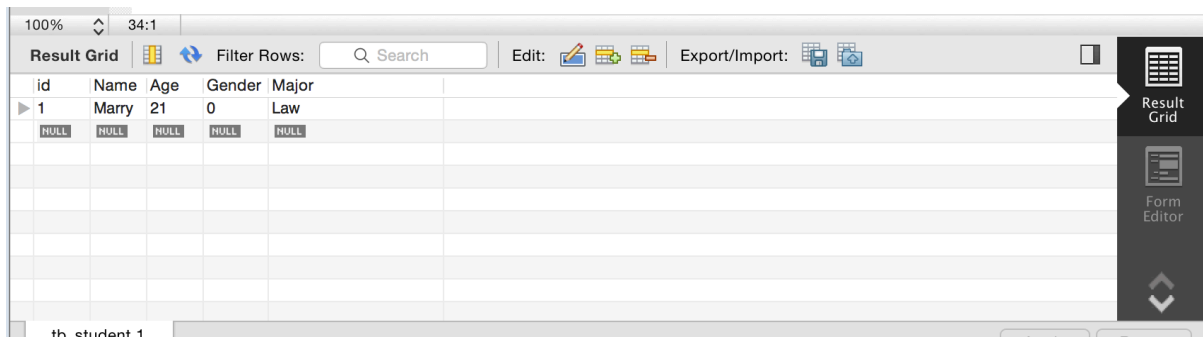
Insert into Stu_Man.tb_student (id, Name, Age, Gender, Major) values (001, 'Marry', 21, 0, 'Law');

or using the visual grid:

select the table you want to edit, and click the grid icon (third) or right click the mouse and then “select...”;



you will see the current data of this table, such as the record we just inserted by executing SQL:

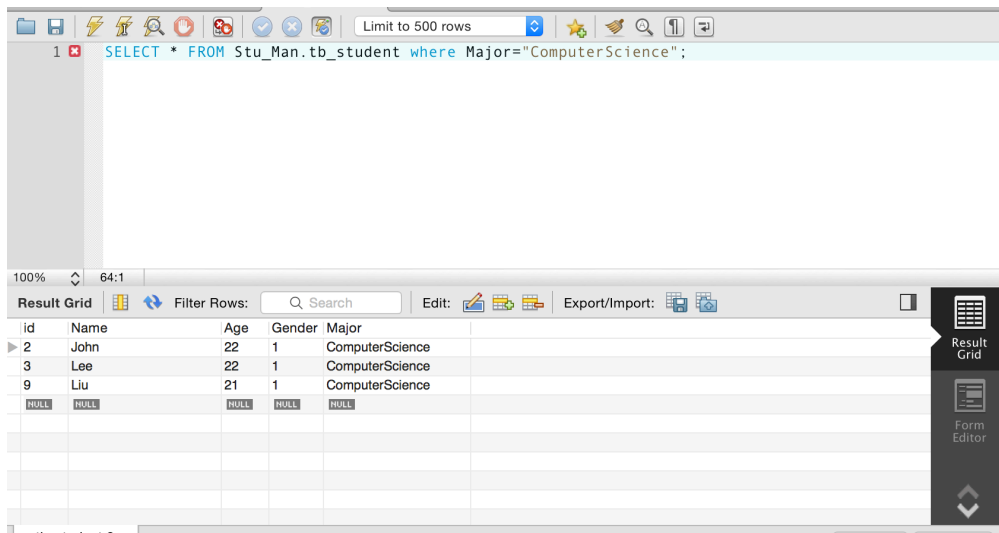


you can achieve data edit (edit, insert, delete) or export and import data (many data formats are legal, such as .SQL, .CSV and so on).

5. One most important operation: “Select”

if I want to find out the students whose majors are “ComputerScience”:

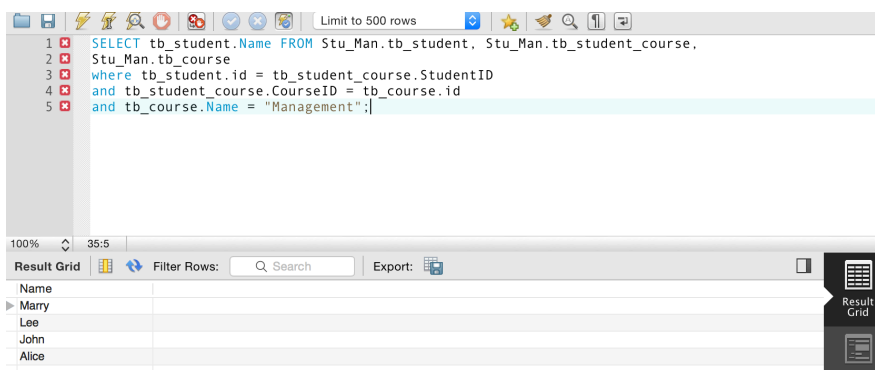
*SELECT * FROM Stu_Man.tb_student where Major="ComputerScience";* * means selecting all the attributes;



If I just want to know their names:

SELECT Name FROM Stu_Man.tb_student where Major="ComputerScience";

Selecting from multiple tables with union: such as selecting students who are studying “Management” course



*SELECT tb_student.Name FROM Stu_Man.tb_student, Stu_Man.tb_student_course, Stu_Man.tb_course
where tb_student.id = tb_student_course.StudentID
and tb_student_course.CourseID = tb_course.id
and tb_course.Name = "Management";*

6. Advanced functions (statistics), group by and roll up

(<http://dev.mysql.com/doc/refman/5.7/en/group-by-modifiers.html>)

If I want to know the number of students for each course:

```

1 select Stu_Man.tb_student_course.CourseID,
2 count(distinct(Stu_Man.tb_student_course.StudentID))
3 from Stu_Man.tb_student_course
4 group by tb_student_course.CourseID;

```

CourseID	count(distinct(Stu_Man.tb_student_course.S...
1	5
2	4
3	1

```

select Stu_Man.tb_student_course.CourseID,
count(distinct(Stu_Man.tb_student_course.StudentID))
from Stu_Man.tb_student_course
group by tb_student_course.CourseID;

```

If I want to know the number of students for each course and also the number of all the students:

```

1 select Stu_Man.tb_student_course.CourseID,
2 count(distinct(Stu_Man.tb_student_course.StudentID))
3 from Stu_Man.tb_student_course
4 group by tb_student_course.CourseID with rollup;

```

CourseID	count(distinct(Stu_Man.tb_student_course.S...
1	5
2	4
3	1
NULL	5

```

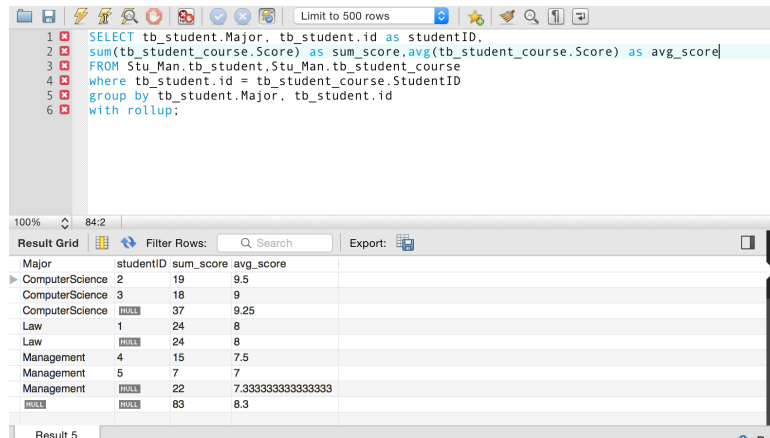
select Stu_Man.tb_student_course.CourseID,
count(distinct(Stu_Man.tb_student_course.StudentID))
from Stu_Man.tb_student_course
group by tb_student_course.CourseID with rollup;

```

the difference is the last row in the second selection. The GROUP BY clause permits a WITH ROLLUP modifier that causes extra rows to be added to the summary output. These rows represent higher-level (or super-aggregate) summary operations. ROLLUP thus enables you to answer questions at multiple levels of analysis with a single query. It can be used, for example, to provide support for OLAP (Online Analytical Processing) operations.

Another example:

If I want to know the total scores, average scores for each student and the students from each major



```
SELECT tb_student.Major, tb_student.id as studentID,  
sum(tb_student_course.Score) as sum_score, avg(tb_student_course.Score) as avg_score  
FROM Stu_Man.tb_student, Stu_Man.tb_student_course  
where tb_student.id = tb_student_course.StudentID  
group by tb_student.Major, tb_student.id  
with rollup;
```

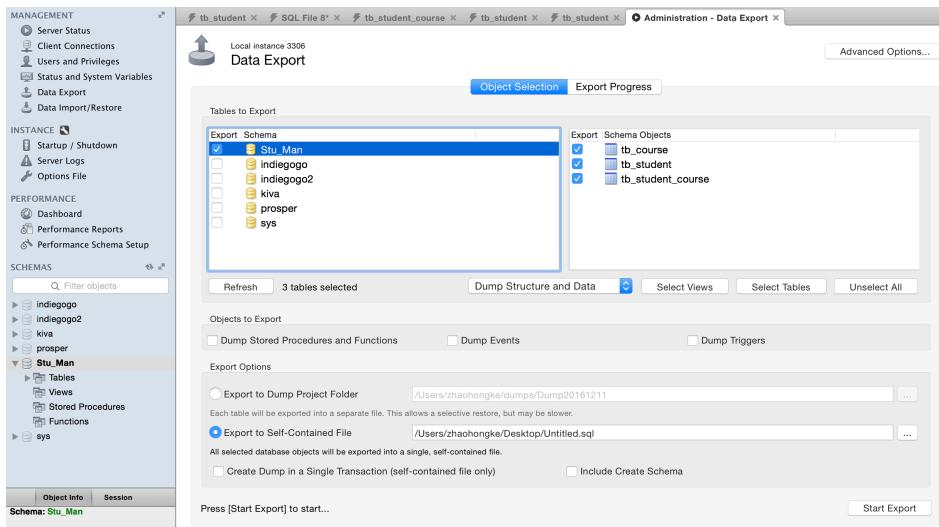
**More complicated SQLs (please refer <http://www.w3schools.com/sql/>,
<http://dev.mysql.com/doc/refman/5.7/en/examples.html>)**

The referred database instance and another more complicated database for e-commerce

Download the example database in SQL format. You can import the complete database instance directly without building step by step.

Export database or a certain table:

Click the ‘Data Export’, and then select the target database or table, click the “Export to Self-Contained File” option, and finally click the “Start Export” button.

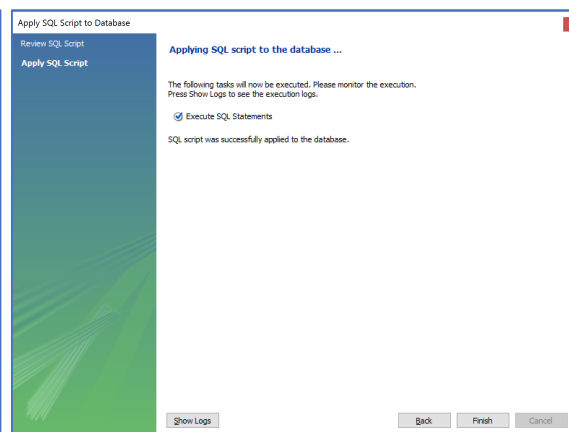
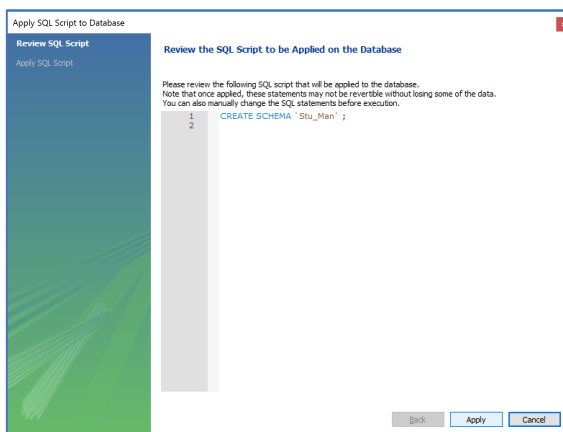
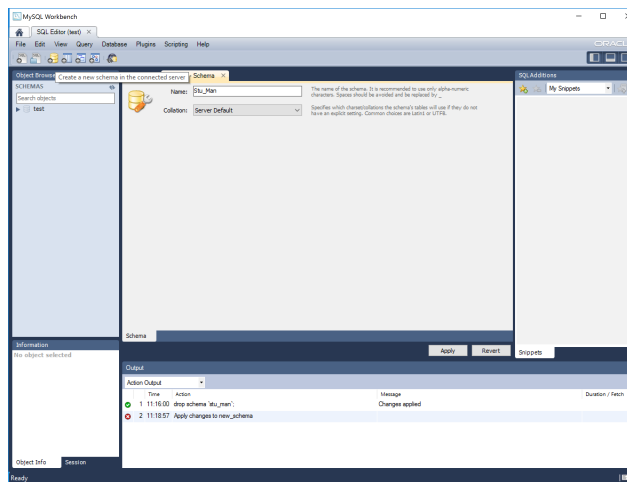


Import database or tables:

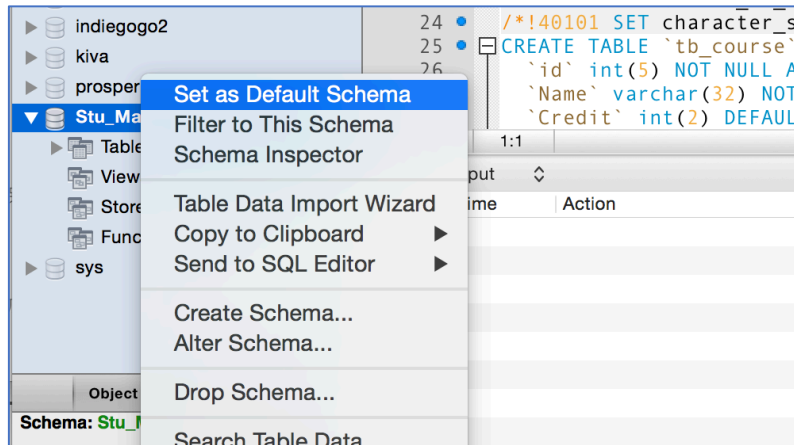
1.Download the referred database instance for student management

https://mega.nz/#!upZCUQBI!Wb9cwiv5yZ_r0Nkns-SLBJWcdyAQc6UM6CvZPZ7YwQw

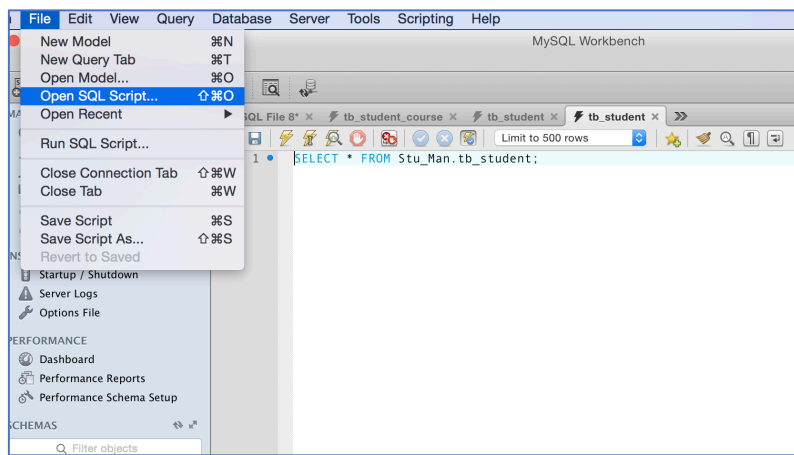
2.Create a new schema for the database, can be named as “Stu_Man”



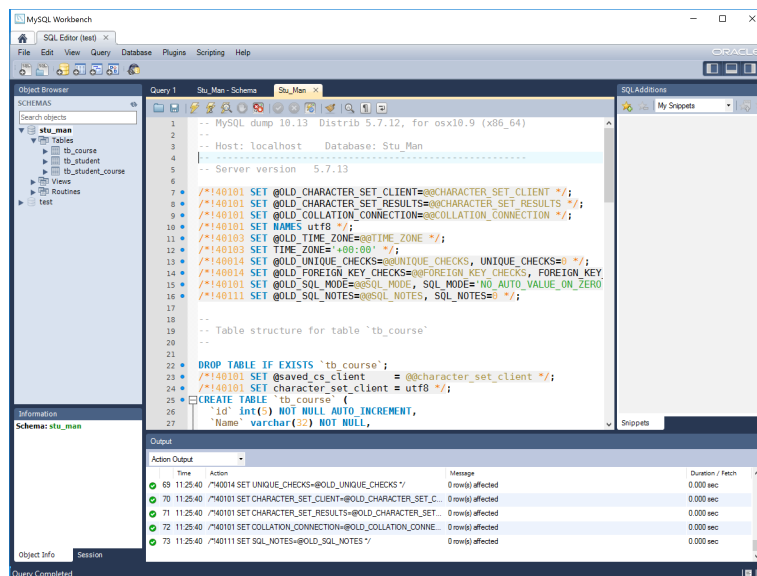
3. Set the created database as the default schema



4. Open the downloaded SQL file



5. Execute this SQL script and you will find the database on the left (refresh schemas).



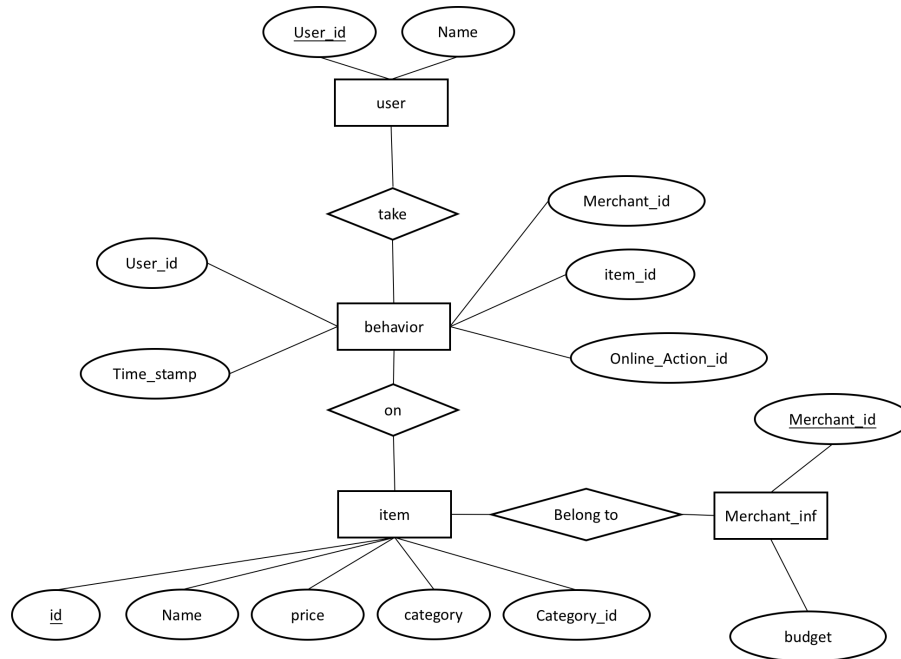
A more complicated database instance for e-commerce:

1.Download the complicated instance

<https://mega.nz/#!XtJigL6T!TOc-xg4e3bVXYXwoOiuMfKjxYBgmkLTjOqnmyrfnPjg>

and import this database in the similar way (above).

2.ER-diagram for this data



3.Data description

this database contains four tables, i.e., user, item, merchant_inf and behavior

Table name	user	item	Merchant_inf	behavior
#column	2	5	2	5
#row	49	100	247	980

Some attribute descriptions:

User table: the user information

User_id,

Name, (the user name, which is private and invisible)

Item table: the item information

Id,

Name,

Price,

Category,
Category_id,

Merchant_inf table: the merchant information

Merchant_id,
Budget, budget constraints imposed on the merchant

Behavior table: users' behaviors, such as "click or buy" some items

User_id,
Merchant_id,
Time_stamp, the time of behavior
Item_id,
Online_Action_id, "0" denotes "click" while "1" for "buy"