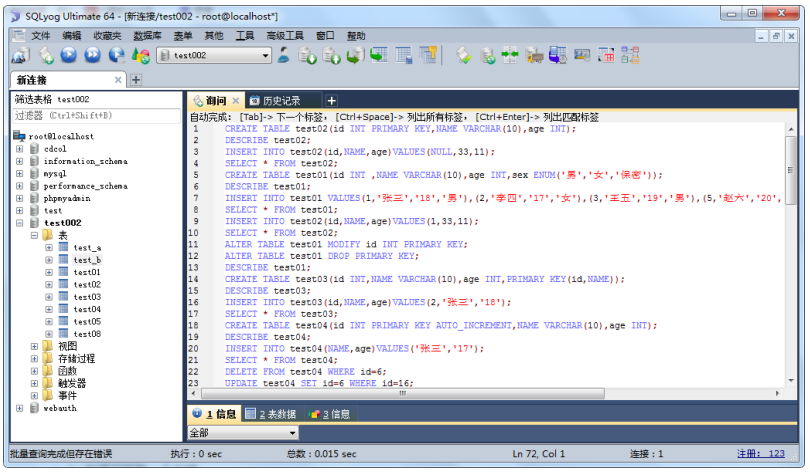
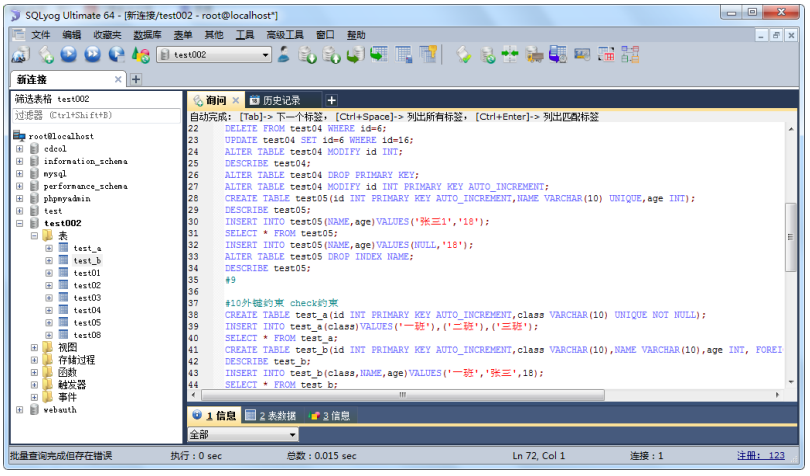
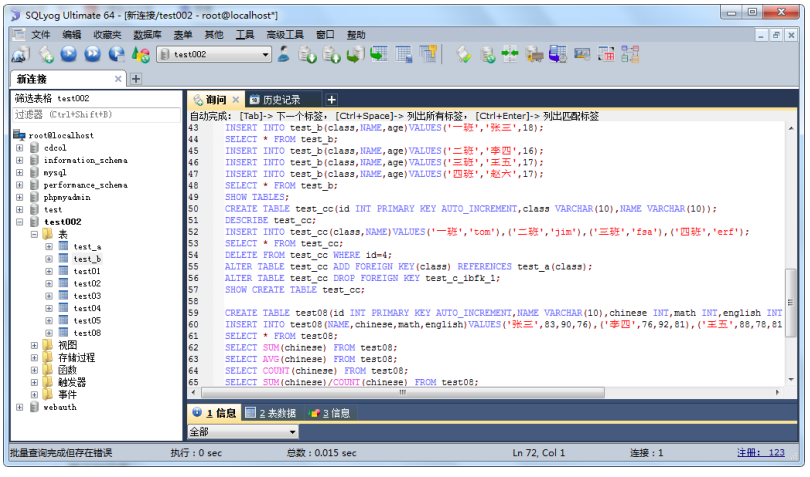
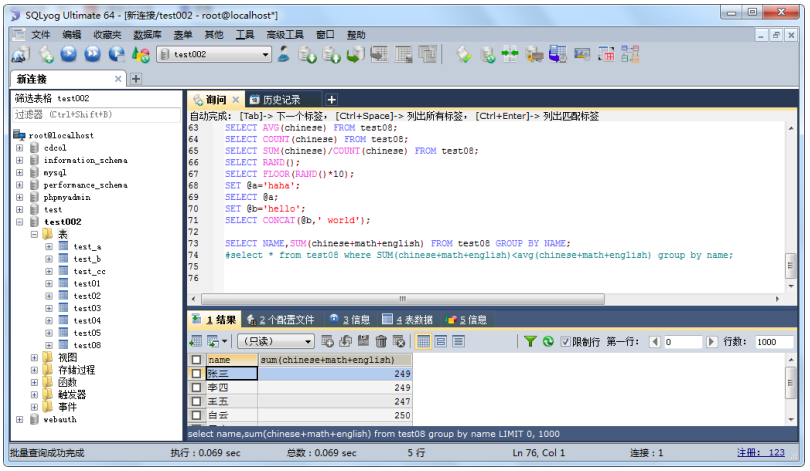
数据库2

- 
SQLyog Ultimate 64 - [新连接/test002 - root@localhost*]
文件 编辑 收藏夹 数据库 表单 其他 工具 高级工具 窗口 帮助
新连接
筛选表格 test002
过滤器 (Ctrl+Shift+B)
root@localhost
cdcol
information_schema
mysql
performance_schema
phpmyadmin
test
test002
表
test_a
test_b
test01
test02
test03
test04
test05
test08
视图
存储过程
函数
触发器
事件
webauth
询问
历史记录
自动完成: [Tab]-> 下一个标签, [Ctrl+Space]-> 列出所有标签, [Ctrl+Enter]-> 列出匹配标签
1 CREATE TABLE test02(id INT PRIMARY KEY,NAME VARCHAR(10),age INT);
2 DESCRIBE test02;
3 INSERT INTO test02(id,NAME,age)VALUES(NULL,33,11);
4 SELECT * FROM test02;
5 CREATE TABLE test01(id INT ,NAME VARCHAR(10),age INT,sex ENUM('男','女','保密'));
6 DESCRIBE test01;
7 INSERT INTO test01 VALUES(1,'张三','18','男'),(2,'李四','17','女'),(3,'王五','19','男'),(5,'赵六','20',
8 SELECT * FROM test01;
9 INSERT INTO test02(id,NAME,age)VALUES(1,33,11);
10 SELECT * FROM test02;
11 ALTER TABLE test01 MODIFY id INT PRIMARY KEY;
12 ALTER TABLE test01 DROP PRIMARY KEY;
13 DESCRIBE test01;
14 CREATE TABLE test03(id INT,NAME VARCHAR(10),age INT,PRIMARY KEY(id,NAME));
15 DESCRIBE test03;
16 INSERT INTO test03(id,NAME,age)VALUES(2,'张三','18');
17 SELECT * FROM test03;
18 CREATE TABLE test04(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10),age INT);
19 DESCRIBE test04;
20 INSERT INTO test04(NAME,age)VALUES('张三','17');
21 SELECT * FROM test04;
22 DELETE FROM test04 WHERE id=6;
23 UPDATE test04 SET id=6 WHERE id=16;
1 信息
2 表数据
3 信息
全部
批量查询完成但存在错误
执行：0 sec
总数：0.015 sec
Ln 72, Col 1
连接：1
注册: 123

- 
SQLyog Ultimate 64 - [新连接/test002 - root@localhost*]
文件 编辑 收藏夹 数据库 表单 其他 工具 高级工具 窗口 帮助
新连接
筛选表格 test002
过滤器 (Ctrl+Shift+B)
root@localhost
cdcol
information_schema
mysql
performance_schema
phpmyadmin
test
test002
表
test_a
test_b
test01
test02
test03
test04
test05
test08
视图
存储过程
函数
触发器
事件
webauth
询问
历史记录
自动完成: [Tab]-> 下一个标签, [Ctrl+Space]-> 列出所有标签, [Ctrl+Enter]-> 列出匹配标签
22 DELETE FROM test04 WHERE id=6;
23 UPDATE test04 SET id=6 WHERE id=16;
24 ALTER TABLE test04 MODIFY id INT;
25 DESCRIBE test04;
26 ALTER TABLE test04 DROP PRIMARY KEY;
27 ALTER TABLE test04 MODIFY id INT PRIMARY KEY AUTO_INCREMENT;
28 CREATE TABLE test05(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10) UNIQUE,age INT);
29 DESCRIBE test05;
30 INSERT INTO test05(NAME,age)VALUES('张三1','18');
31 SELECT * FROM test05;
32 INSERT INTO test05(NAME,age)VALUES(NULL,'18');
33 ALTER TABLE test05 DROP INDEX NAME;
34 DESCRIBE test05;
35 #9
36
37 #10外键约束 check约束
38 CREATE TABLE test_a(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10) UNIQUE NOT NULL);
39 INSERT INTO test_a(class)VALUES('一班'),('二班'),('三班');
40 SELECT * FROM test_a;
41 CREATE TABLE test_b(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10),NAME VARCHAR(10),age INT, FOREI
42 DESCRIBE test_b;
43 INSERT INTO test_b(class,NAME,age)VALUES('一班','张三',18);
44 SELECT * FROM test_b;
1 信息
2 表数据
3 信息
全部
批量查询完成但存在错误
执行：0 sec
总数：0.015 sec
Ln 72, Col 1
连接：1
注册: 123

- 
SQLyog Ultimate 64 - [新连接/test002 - root@localhost*]
文件 编辑 收藏夹 数据库 表单 其他 工具 高级工具 窗口 帮助
新连接
筛选表格 test002
过滤器 (Ctrl+Shift+B)
root@localhost
cdcol
information_schema
mysql
performance_schema
phpmyadmin
test
test002
表
test_a
test_b
test01
test02
test03
test04
test05
test08
视图
存储过程
函数
触发器
事件
webauth
询问
历史记录
自动完成: [Tab]-> 下一个标签, [Ctrl+Space]-> 列出所有标签, [Ctrl+Enter]-> 列出匹配标签
43 INSERT INTO test_b(class,NAME,age)VALUES('一班','张三',18);
44 SELECT * FROM test_b;
45 INSERT INTO test_b(class,NAME,age)VALUES('二班','李四',16);
46 INSERT INTO test_b(class,NAME,age)VALUES('三班','王五',17);
47 INSERT INTO test_b(class,NAME,age)VALUES('四班','赵六',17);
48 SELECT * FROM test_b;
49 SHOW TABLES;
50 CREATE TABLE test_cc(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10),NAME VARCHAR(10));
51 DESCRIBE test_cc;
52 INSERT INTO test_cc(class,NAME)VALUES('一班','tom'),('二班','jim'),('三班','fsa'),('四班','erf');
53 SELECT * FROM test_cc;
54 DELETE FROM test_cc WHERE id=4;
55 ALTER TABLE test_cc ADD FOREIGN KEY(class) REFERENCES test_a(class);
56 ALTER TABLE test_cc DROP FOREIGN KEY test_c_ibfk_1;
57 SHOW CREATE TABLE test_cc;
58
59 CREATE TABLE test08(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10),chinese INT,math INT,english INT
60 INSERT INTO test08(NAME,chinese,math,english)VALUES('张三',83,90,76),('李四',76,92,81),('王五',88,78,81
61 SELECT * FROM test08;
62 SELECT SUM(chinese) FROM test08;
63 SELECT AVG(chinese) FROM test08;
64 SELECT COUNT(chinese) FROM test08;
65 SELECT SUM(chinese)/COUNT(chinese) FROM test08;
1 信息
2 表数据
3 信息
全部
批量查询完成但存在错误
执行：0 sec
总数：0.015 sec
Ln 72, Col 1
连接：1
注册: 123



- 约束：在数据类型的基础上，对插入表中的数据进行进一步限制
- 常见的约束：
  - 主键约束：被主键约束的字段，具有非null且唯一的属性。表中有主键约束后，可以极大提示查询效率
  - 自增约束：自增约束只修饰主键，如果主键是可序列号的值，在不给主键插入值时，主键的值自动增长
  - 唯一约束：被唯一约束修饰的字段，不能出现重复的值
  - 非null约束：被非null约束修饰的字段，在不插入值时，取默认值
  - 默认值约束：被默认值约束的字段，在不插入值时，取默认值
  - 外键约束：涉及多个表，副表中某个字段的取值范围，被主表已有的数据约束
  - check约束：在8版本才生效。可以根据自定义字段取值范围来约束字段
- 主键约束：
- 关键字：primary key
- *一张表只能有一个主键
- 创建主键约束
  - 建表时创建
    - 命令公式：CREATE TABLE 表名(字段 字段数据类型 PRIMARY KEY,...);

```
CREATE TABLE test02(id INT PRIMARY KEY,NAME VARCHAR(10),age INT);
DESCRIBE test02;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	(NULL)	
NAME	varchar(10)	YES		(NULL)	
age	int(11)	YES		(NULL)	

  - 对已有表格追加主键约束：
    - 前提条件：被追加的字段，本身的数据就满足非null且唯一
    - 命令公式：ALTER TABLE 表名 MODIFY 字段 字段数据类型 PRIMARY KEY;

```
27  ALTER TABLE test01 MODIFY id INT PRIMARY KEY;
28  DESCRIBE test01;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	
NAME	varchar(10) 11B	YES		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	
sex	enum('?',... 18B	YES		(NULL) 0K	

- 删除主键约束（通常不会有这种操作）：ALTER TABLE 表名 DROP PRIMARY KEY;

```
29  ALTER TABLE test01 DROP PRIMARY KEY;
30  DESCRIBE test01;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO		(NULL) 0K	
NAME	varchar(10) 11B	YES		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	
sex	enum('?',... 18B	YES		(NULL) 0K	

- *联合主键：通常出现在非常多字段的表中
  - 创建联合主键公式：CREATE TABLE 表名(字段1 数据类型,字段2 数据类型,...,PRIMARY KEY(字段1,字段2,...));

```
31  CREATE TABLE test03(id INT,NAME VARCHAR(10),age INT,PRIMARY KEY(id,NAME));
32  DESCRIBE test03;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	0 1B	
NAME	varchar(10) 11B	NO	PRI	0B	
age	int(11) 7B	YES		(NULL) 0K	

  - 对已有表追加联合主键：ALTER TABLE 表名 ADD PRIMARY KEY(字段1,字段2,...);

```
36  ALTER TABLE test03 ADD PRIMARY KEY(age,NAME);
37  DESCRIBE test03;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	YES		(NULL) 0K	
NAME	varchar(10) 11B	NO	PRI	0B	
age	int(11) 7B	NO	PRI	0 1B	

- 自增约束：专门修饰主键约束
  - 自增的规律：消费最新的序列号
- 关键字：auto_increment
- 创建自增约束
  - 建表时创建：create table 表名(字段 字段数据类型 primary key auto_increment,...);

```
40  CREATE TABLE test04(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10),age INT);
41  DESCRIBE test04;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

  - 对已有表追加：
    - 如果表中已有主键，需要先删除主键

    - alter table 表名 modify 字段 数据类型 primary key auto_increment,...);

```
ALTER TABLE test04 MODIFY id INT PRIMARY KEY AUTO_INCREMENT;
DESCRIBE test04;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

- 删除自增约束：直接修改主键数据类型即可：ALTER TABLE 表名 modify 主键 主键数据类型;

```
ALTER TABLE test04 MODIFY id INT;
DESCRIBE test04;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	0 1B	
NAME	varchar(10) 11B	YES		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

- 唯一约束
- 关键字：unique
- 创建唯一约束
    - 建表时创建：create table 表名(字段 字段数据类型 unique,...);

```
CREATE TABLE test05(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10) UNIQUE,age INT);
DESCRIBE test05;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES	UNI	(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

    - 对已有表追加：
        - 前提条件：被追加的字段本身就符合唯一的特性
        - alter table 表名 modify 字段 数据类型 unique;

```
ALTER TABLE test05 MODIFY age INT UNIQUE;
DESCRIBE test05;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES	UNI	(NULL) 0K	
age	int(11) 7B	YES	UNI	(NULL) 0K	

    - 删除唯一约束：alter table 表名 drop index 字段;

```
ALTER TABLE test05 DROP INDEX age;
DESCRIBE test05;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES	UNI	(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

- 非null约束
- 关键字：not null
- 创建非null约束

- 建表时创建：create table 表名(字段 字段数据类型 not null,...);

```
65  CREATE TABLE test06(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10) NOT NULL,age INT);
66  DESCRIBE test06;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	NO		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

- 对已有表追加：alter table 表名 modify 字段 数据类型 not null;

```
70  ALTER TABLE test06 MODIFY age INT NOT NULL;
71  DESCRIBE test06;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	NO		(NULL) 0K	
age	int(11) 7B	NO		(NULL) 0K	

- 删除非null约束：相当于修改数据类型 alter table 表名 modify 字段 数据类型;

```
72  ALTER TABLE test06 MODIFY age INT;
73  DESCRIBE test06;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	NO		(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

- **如果表中没有主键，其中有一个字段同时具有非null且唯一，会被默认推举为主键

```
77  ALTER TABLE test06 MODIFY NAME VARCHAR(10) UNIQUE NOT NULL;
78  DESCRIBE test06;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO		0 1B	
name	varchar(10) 11B	NO	PRI	(NULL) 0K	
age	int(11) 7B	YES		(NULL) 0K	

-

- 默认值约束
- 关键字：defalut
- 创建默认值约束
  - 建表时创建：create table 表名(字段 字段数据类型 default 值,...);

```
82  CREATE TABLE test07(id INT PRIMARY KEY AUTO_INCREMENT,NAME VARCHAR(10) DEFAULT '张三',age INT);
83  DESCRIBE test07;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES		张三 6B	
age	int(11) 7B	YES		(NULL) 0K	

  - 对已有表追加：alter table 表名 modify 字段 数据类型 default 值;

```
83  ALTER TABLE test07 MODIFY age INT DEFAULT 18;
84  DESCRIBE test07;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type	Null	Key	Default	Extra
id	int(11) 7B	NO	PRI	(NULL) 0K	auto_increment
NAME	varchar(10) 11B	YES		张三 6B	
age	int(11) 7B	YES		18 2B	

- 删除默认值约束：相当于修改数据类型alter table 表名 modify 字段 数据类型;

```
84  ALTER TABLE test07 MODIFY age INT;
85  DESCRIBE test07;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type		Null	Key	Default		Extra
id	int(11)	7B	NO	PRI	(NULL)	0K	auto_increment
NAME	varchar(10)	11B	YES		张三	6B	
age	int(11)	7B	YES		(NULL)	0K	

-
- 外键约束
- 关键字：foreign key references
- 创建外键约束
  - 前提条件：
    - 主表被引用的字段必须具有非null且唯一的属性
    - 数据库的引擎需要支持事务
      - 数据库的引擎：查询和写入数据的方式
        - innodb:支持事务。数据的读写相对较慢
        - myisam:不支持事务。数据的读写相对较快
      - 事务：可以理解成是否支持自定义逻辑
  - 建表时创建：命令公式：create table 表名(字段 数据类型,...,foreign key(字段) references 主表(指定字段));

```
91  CREATE TABLE test_a(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10) UNIQUE NOT NULL);
92  INSERT INTO test_a(class)VALUES('一班'),('二班'),('三班');
93  SELECT * FROM test_a;
94  CREATE TABLE test_b(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10),NAME VARCHAR(10)
95  ,age INT, FOREIGN KEY(class) REFERENCES test_a(class));
96  DESCRIBE test_b;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type		Null	Key	Default		Extra
id	int(11)	7B	NO	PRI	(NULL)	0K	auto_increment
class	varchar(10)	11B	YES	MUL	(NULL)	0K	
NAME	varchar(10)	11B	YES		(NULL)	0K	
age	int(11)	7B	YES		(NULL)	0K	

  - 对已有表追加：alter table 表名 add foreign key (字段) references 主表(指定字段));

```
11  CREATE TABLE test_c(id INT PRIMARY KEY AUTO_INCREMENT,class VARCHAR(10),NAME VARCHAR(10));
12  DESCRIBE test_c;
13  DESCRIBE test_a;
14  ALTER TABLE test_c ADD FOREIGN KEY (class) REFERENCES test_a(class);
15  DESCRIBE test_c;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Field	Type		Null	Key	Default		Extra
id	int(11)	7B	NO	PRI	(NULL)	0K	auto_increment
class	varchar(10)	11B	YES	MUL	(NULL)	0K	
name	varchar(10)	11B	YES		(NULL)	0K	

- 删除外键约束：先查询外键的名称
  - 查询表的外键名称：show create table 表名;

```
SHOW CREATE TABLE test_c;
CREATE TABLE `test_c` (
  `id` INT(11) NOT NULL AUTO_INCREMENT,
  `class` VARCHAR(10) DEFAULT NULL,
  `name` VARCHAR(10) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `class` (`class`),
  CONSTRAINT `test_c_ibfk_1` FOREIGN KEY (`class`) REFERENCES `test_a` (`class`)
) ENGINE=INNODB DEFAULT CHARSET=latin1
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读)

Table	Create Table
test_c	CREATE TABLE `test_c` (`id` int(11) NOT NULL AUTO_INCREMENT, `cl

- alter table 表名 DROP FOREIGN KEY 外键约束名；

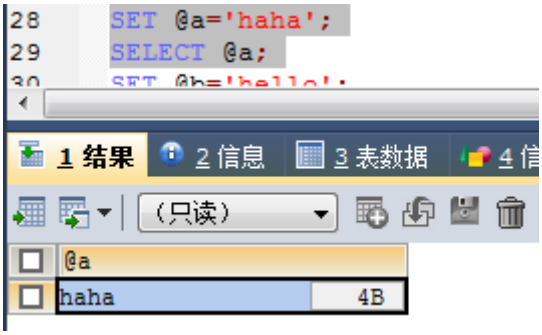
```
ALTER TABLE test_c DROP FOREIGN KEY test_c_ibfk_1;
DESCRIBE test_c;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

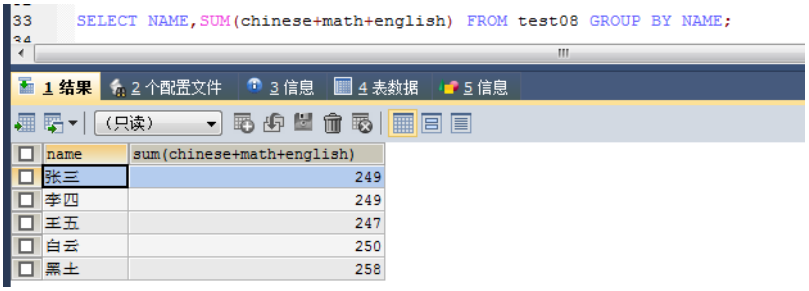
(只读)

Field	Type		Null	Key	Default		Extra
id	int(11)	7B	NO	PRI	*(NULL)*	0K	auto_increment
class	varchar(10)	11B	YES	MUL	*(NULL)*	0K	
name	varchar(10)	11B	YES		*(NULL)*	0K	

-
- check约束
- 关键字：check
- check约束可以指定某个或多个字段的取值范围
- 命令格式：create table 表名(字段 数据类型,...,check(某个字段的取值范围));
-
- 聚合函数
  - 聚合：在mysql中，通过函数计算得到的结果会以数据包的方式返回
  - 函数：一组实现某种功能的代码，可以通过调用函数的名称，传入适当的参数得到相应的结果
- mysql中常见的函数
  - sum():求和
  - avg():求平均值
  - max()/min():获取最大/最小值
  - count():统计、计数
  - rand():生成一个0-1之间的随机数
  - floor():向下取整
  - concat():拼接多个字符串



- 分组查询：也叫聚合后的发散
- 为什么要分组查询：分组查询一般用于统计数据，使用分组能让汇总结果一目了然。
- 关键字：group by 字段
  - 根据分组的字段，一定是表中已有的字段



-
- 排序
- 关键字：order by 字段
  - 正序：asc，默认正序
  - 倒序：desc
  -
- 分页查询：通常配合排序使用
- 关键字：limit m,n
  - m:从第几行开始展示查询结果。0对应第一行，1对应第二行，以此类推
  - n：代表从m行开始，展示多少行
  - #查询数学成绩第二名

```
SELECT math FROM test08 ORDER BY math DESC LIMIT 1,1;
```

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

math
92

-
- 子查询：查询语句的条件，也是一个查询语句
  - #查询数学成绩最高的人的信息

```
SELECT MAX(math) FROM test08;
SELECT * FROM test08 WHERE math=93;
SELECT * FROM test08 WHERE math=(SELECT MAX(math) FROM test08);
```

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

id	NAME	chinese	math	english
4	白云	75	93	82

-
- 字段去重：只显示不重复的数据
  - 关键字：distinct

```
SELECT DISTINCT NAME FROM test08;
```

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

name
张三
李四
王五
白云
黑土

- 报错的原因
  - where尝试拿出一个字段和某个值进行比较，但整个字段原表中并不存在

```
SELECT NAME,SUM(chinese+math+english) FROM test08 GROUP BY NAME
WHERE SUM(chinese+math+english)<(SELECT AVG(chinese+math+english) FROM test08);
```

1 信息 | 2 表数据 | 3 信息

```
1 queries executed, 0 success, 1 errors, 0 warnings

查询：select name,sum(chinese+math+english) from test08 group by name where SUM(chinese+math+english)<(select
avg(chinese+math+english...

错误代码： 1064
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use
near 'where SUM(chinese+math+english)<(select avg(chinese+math+english) from test08) L' at line 2

执行耗时   : 0 sec
传送时间   : 0 sec
总耗时     : 0 sec
```

  - 解决办法：将where改写成having
- having：也是用于条件筛选

```
SELECT NAME,SUM(chinese+math+english) FROM test08 GROUP BY NAME
HAVING SUM(chinese+math+english)<(SELECT AVG(chinese+math+english) FROM test08);
```

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

name	sum(chinese+math+english)
张三	249
李四	249
王五	247
白云	250

- having和where的区别
  - where运算优先级较高，having优先级较低
  - where针对是原表中已有的字段，having针对的是分组后的再筛选
  - where只能在group by之前使用，而having只能在group by使用