启动mysql监视器，指令 mysql -u 用户名 -p密码

注意：-p和密码之间没有空格，

更换密码指令 mysqladmin -u root PASSWORD 新密码

显示数据库一览 show databases;

删除数据库 drop databases;

创建数据库 create database 数据库名；

创建新用户并赋予其对数据库的操作权限 grant all privileges on 数据库.数据表 to user@localhost inentified by ‘密码’

赋予了从本地连接数据库的用户user能对数据库的数据表所有对象进行操作的全部权限

指定使用数据库：use 数据库名；

查看现在使用的数据库 select database();

注意：必须有（）

显示所有的表 show tables;

**显示表结构** desc 表名；

显示创建表语法 show create table 表名；

删除表 drop table 表名；

**插入数据** insert into 表名 （列名1，列名2.。。。。。）values （数据1，数据2.。。。。。）

注意：如果插入的数据中涵盖所有列，可以省略所有列名部分，当插入指定数值时，两边是一一对应的，如果个数不符则执行失败。对于不可为空的列，如果该列不为空，且没有设置默认值，如果插入数据时没指定数值，则插入失败。不想设置值的列可以用null（不加引号）

**Sql语句规则**：语句必须以分号结束，保留的关键字不区分大小写，可自由地加入空白或换行符。

**更新指令** update 表名 set 列名1=值1，列名2=值2，… where XXXX;

注意：返回true或者false

**删除**表中记录 delete from 表名 where 条件表达式

注意：如果省略了条件，则删除表中所有数据，但不高效，可以使用truncate table；

**Null 条件**

检索null 时不能使用=运算符，可以使用is null

**Order by 排序**

Select \* from table order by `id` DESC，`birth` DESC;

注意：多个关键字排序时，按照表达式顺序排列优先级，排序方式省略时默认正序ASC。

在mysql中 null被当作最小值看。

**Limit**

Limit [开始位置，数量]，开始位置默认从0开始，也就是说 limit 2 和limit 0，2意义相同

注意：原则上limit语句是要与order by同时使用的。如果没有指明排序方式，数据库会按照随机\*顺序取出，每次数据会不一样。

**数据分组**

以特定的关键字对记录进行整理称为分组化，使用group by语句

**修改表结构ALTER TABLE命令**

**改变列的数据类型，**例如可以将定义为varchar类的列改变为可以容纳大量字符的text类型等，注意修改不能使用的类型时会出错，灵位有时会出现原来数据变成乱码，或者一部分数据消失。因此如果表中原来有数据时，一般最好在对表结构进行修改前的备份

**改变列的类型**：alter table 表名 MODIFY 列名 数据类型（大小）；

例如：ALTER TABLE visitor MODIFY name VARCHAR(30); #改变name字段类型为字符串30

注意：如果列中存在数据了，是不适合进行数据类型的转换的

**追加新列**

将新列追加到表最后：alter table 表名 add 列名 数据类型 默认值等

例如：ALTER TABLE ` visitor`

ADD `contract\_account` varchar(20) NOT NULL DEFAULT '' COMMENT '备注 ',

例如：ALTER TABLE visitor ADD old INT(5)；

将新列追加到最前面，使用关键字first：ALTER TABLE visitor ADD old INT(5) FIRST；

追加到任意列后面，使用关键字 after 列名：ALTER TABLE visitor ADD old INT(5) AFTER name；

例如：alter table `pay\_pwdfree\_contract` ADD `digest` varchar(32) NOT NULL DEFAULT '' COMMENT '备注' AFTER `pay\_channel`;

改变列的名称和类型

语法：ALTER TABLE 表名 CHANGE 修改前的列名 修改后的列名 修改后的类型；

例如：alter table vistor change birth birthday DATE;

删除列

语法： ALTER TABLE 表名 DROP 列名；

例如：alter table pay\_pwdfree\_contract drop digest;

注意：删除列时，原来保存的数据、相关的索引也将被删除，但对表中的其他列没影响；

复制表的列构造和数据

语法：CREATE TABLE 新表名 SELECT \* FROM 旧表名；

例如：create table `pay\_pwdfree\_copy` select \* from `pay\_pwdfree\_contract`;

注意：**不会复制表的主键、索引、表注释等配置,表的配置字符格式等都是默认值**

复制表的列构造

语法：CREATE TABLE 新表名 LIKE 旧表名；

注意：复制结构和原表一样，包括主键、索引、引擎、字符格式、表注释等

复制数据

向一个表中复制另一个表中的全部数据

语法：INSERT INTO 表名 select \* from 含有数据的表

删除表

语法： DROP TABLE 表名；

也可以在表删除语句湖面追加IF EXISTS 语句，删除先判断表是否存在，存在才删除

语法：DROP TABLE IF EXISTS表名；

事务处理（transaction）：将多个更新命令作为一个整体来执行，从而保证数据整合性的机制。

确认表使用可什么储存引擎，必须使用 SHOW CREATE TABLE 表名；查看ENGINE的值，默认引擎是MyISAM，但该引擎不支持事务

小技巧：使用mysql监视器时，可以使用\G替换；可以使数据显示更调理，select 语句用的较多。

变更存储引擎

语法：ALTER TABLE 表名 ENGINE=新引擎； #常用MyISAM、InnoDB

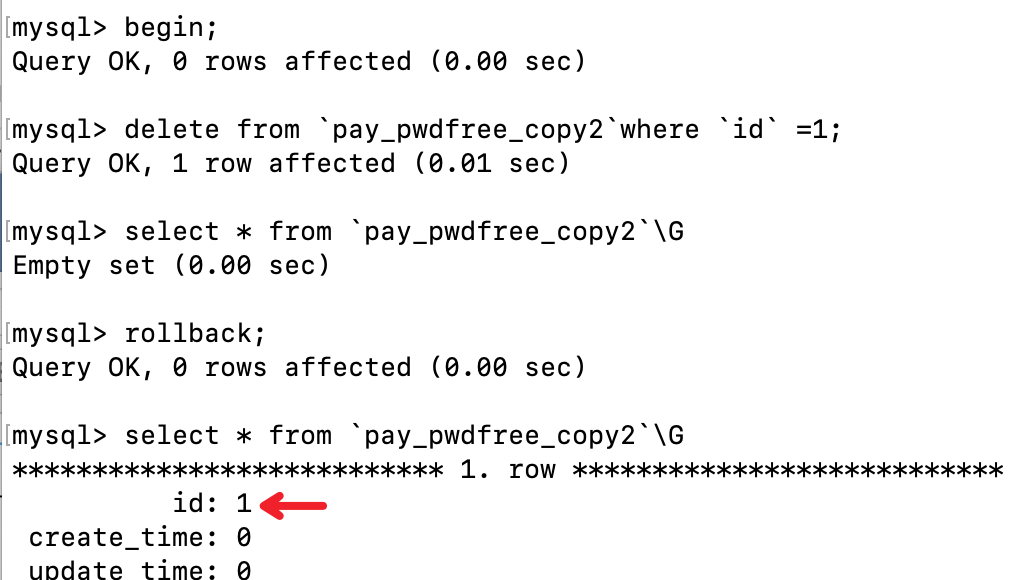
使用事务处理功能时，主要设计3个重要命令：

BEGIN；或者 START TRANSACTION； 声明事务处理开始

COMMIT； 提交整个事务

ROLLBACK；回滚到事务开始状态

例如：



可以看到最后表数据恢复到初始状态，如果将rollback换成commit 那么删除的处理就被提交，不可恢复。

锁的种类

按照使用目的可以分为共享锁定与排他锁定

死锁：是两个不同的事务处理在相互等待对方释放锁定，永远也不可能解除锁定的一种状态，为了防止死锁现象的发生，应尽量对同一对象数据按照相同的顺序进行操作。

如果处理事务时发生停单或者硬件错误，数据库在错误排除后的第一次启动时进行roll forward（前滚）的处理，具体做法是从日志中抽出从最后的检查点到错误发生时间点的事务处理，然后重新执行一次，这样数据库就恢复到错误发生前的状态了。

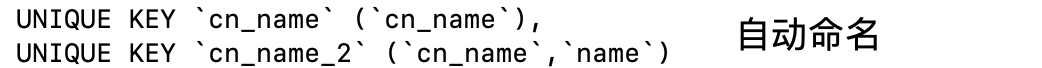
**MySQL中创建索引的两种方式**

* CREATE INDEX index\_name ON table\_1(filed\_1);
* ALTER TABLE table\_1 ADD INDEX(field\_1);

主要区别：

1.使用CREATE创建索引必须提供索引名称，而ALTER方式可以不提供则MySQL会自动为你创建索引名称，如果想使用ALTER方式自定义创建索引名称，可以类似如下SQL语句：

mysql> ALTER TABLE table\_1 ADD INDEX index\_name(field\_1,field\_2);



2.CREATE 方式创建索引，一次只能创建一个索引。ALTER方式可以一次创建多个索引，类似如下SQL语句：

mysql> ALTER TABLE table\_1 ADD PRIMARY KEY(id), ADD INDEX index\_name\_1(field\_1,field\_2), ADD INDEX index\_name\_2(field\_3,field\_4,field\_5);

3.ALTER独有创建PRIMARY KEY的特性。

创建/删除唯一索引

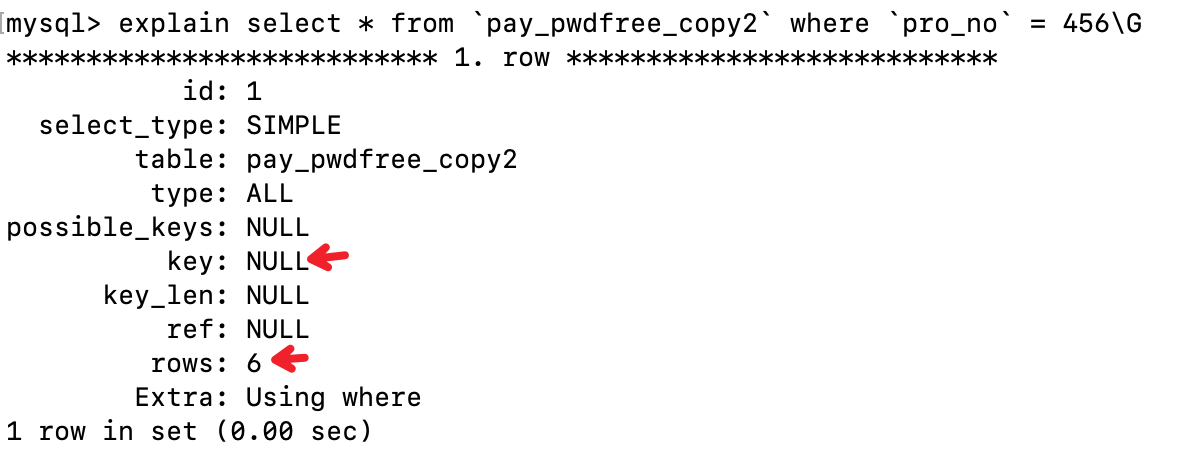
ALTER TABLE `table\_name` ADD UNIQUE *[index\_name]* ( `column` );

ALTER TABLE `table\_name` DROP INDEX index\_name

使用EXPLAIN分析索引使用情况

语法：EXPLAIN select 语句；

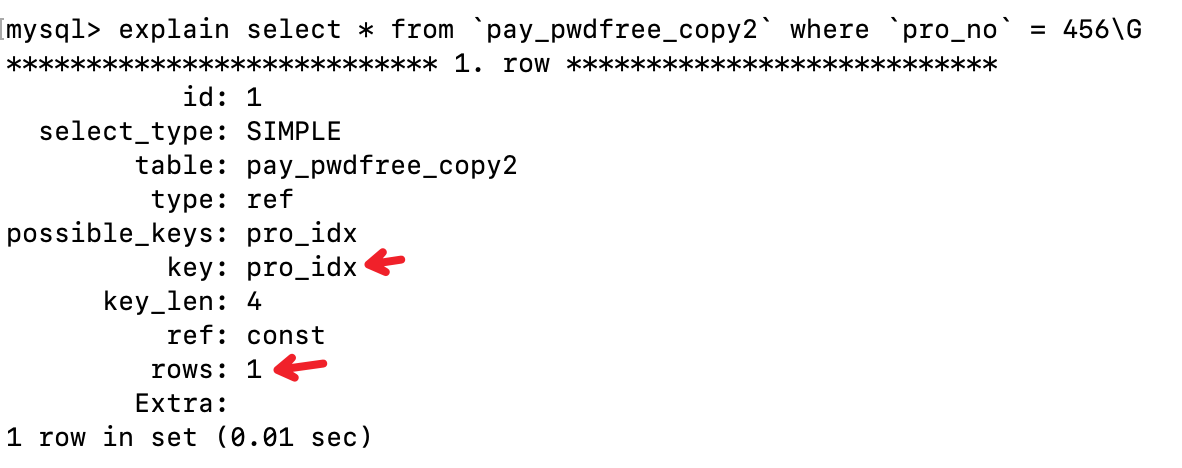
注意：返回结果注意查看key和rows，Key是检索时使用的索引的名称，rows是检索时遍历的记录数，下面是没有给pro\_no加索引的情况下查询，结果是key 为空，没用到索引，row的数值也比较高（一共7条数据）



给pro\_no添加索引：



显示结果用到了新索引，rows也变为1，效率大大提高



注意：如果遍历的次数与创建索引前变化不大，说明创建索引时选择的列名不合理。

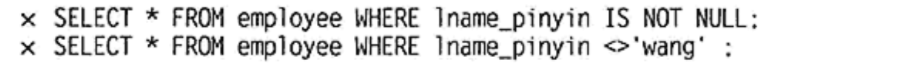
一些无法使用索引的情况：

* Like语句中后方一致和部分一致不能用索引，例如：

Select \* from `table` where `name` like ‘%w%’;

Select \* from `table` where `name` like ‘%w’;

* 使用了is not null 、<>比较运算符的场合，例如



* 对列使用了运算/函数的场合：

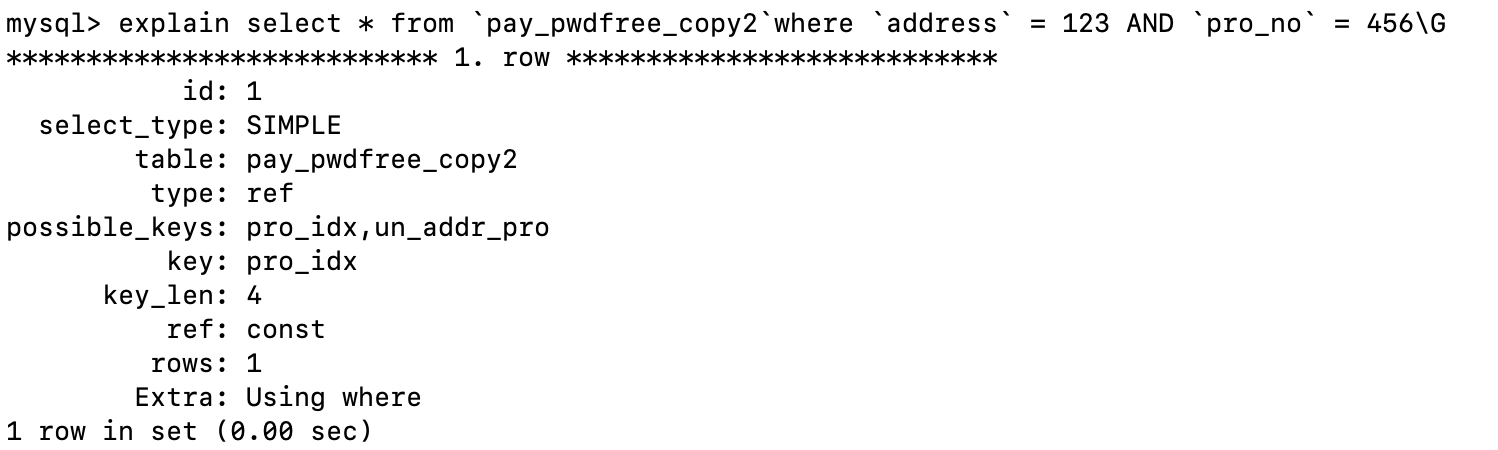


注：year函数时从日期中取出年份

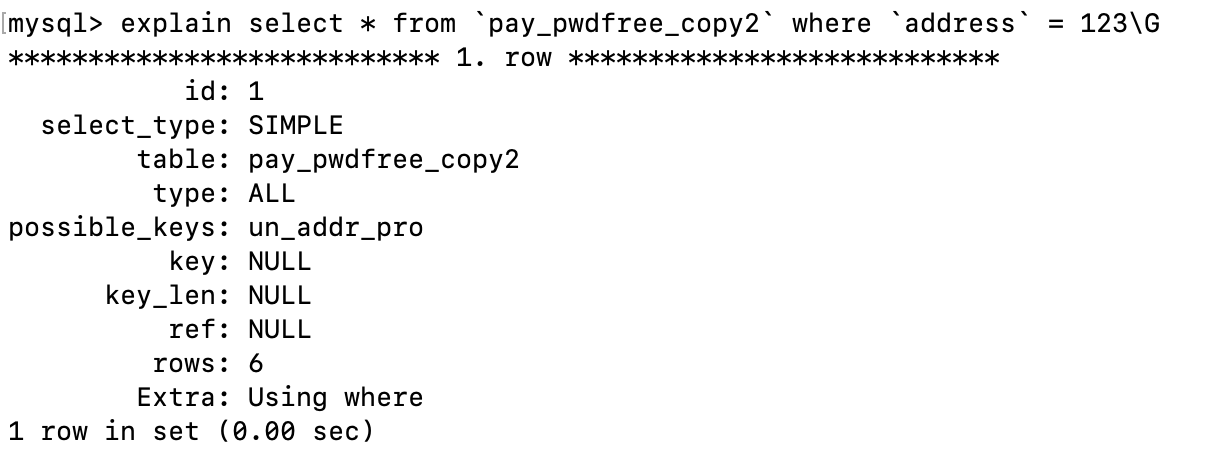
* 复合索引的第一列没有包含在where条件语句中

当创建一个复合索引，包括（name、age），~~当我们检索name，~~或者同时检索name and age时，可以用到复合索引，但是如果不包括第一个name，只检索age，或者 name or age 时，复合索引不起作用。

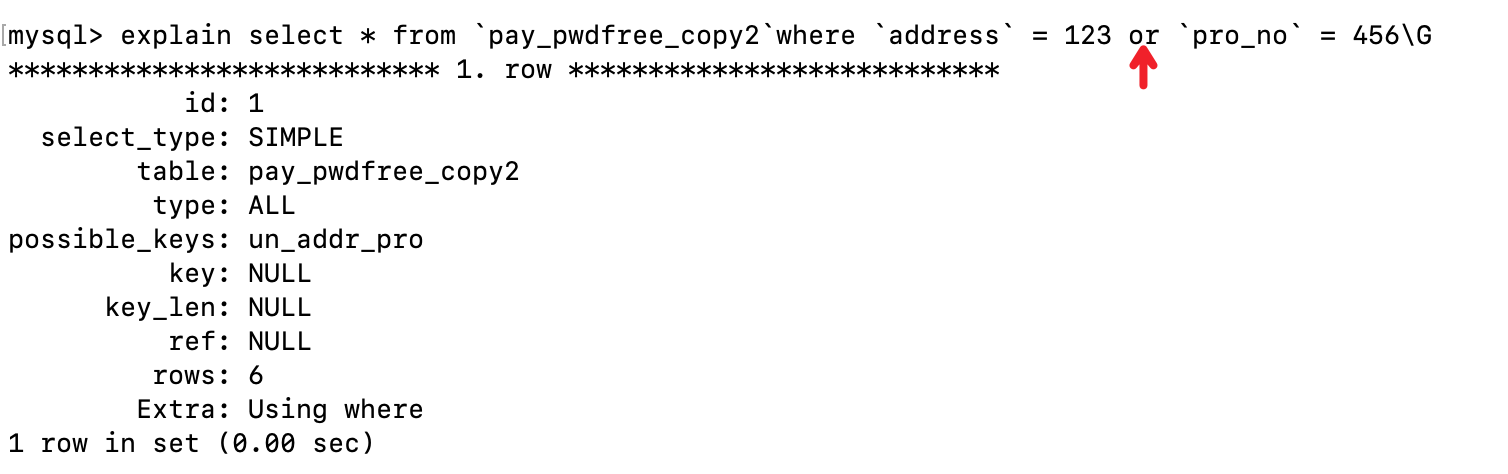
例如：可使用复合索引



不可以使用复合索引



由于使用了or，需要单独检索



数据库的整体备份与恢复

数据库的设置、表与列的定义以及数据等，所有的数据库信息都可以输出到文件中，这个过程被称作转储（dump）。对mysql数据库进行转储操作时，可以使用mysqldump命令。Mysqldump命令会将数据库结构与数据都以sql语句的形式输出到文件中。也就是说，在输出的文件中会出现[CREATE TABLE]这样的创建表语句，以及[INSERT INTO]这样数据插入的语句。

备份语法：

mysqldump -u 用户名 -p密码 数据库名 > 输出文件名

注意和连接数据库一样，-p和密码之间没空格

例如：

# mysqldump -u root -proot platform > platform.db.bak

使用转储文件进行数据库恢复

恢复前，必须有存在的数据库接受数据信息，也就是在其他服务器恢复数据库之前，需要创建数据库。

例如，首先使用mysqladmin命令来创建数据库，其次对mysql命令使用输入重定向来恢复数据库：

mysql> create database platform\_huifu

# mysql -u root -proot platform\_huifu < platform.db.bak