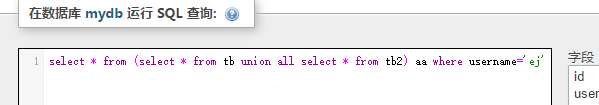
1. 在数据库中有个小坑，如果你使用了union all查出出来后的结果，然后再进行查询的话，一定要给表一个别名，不然会报错；union 和 union all 两个的本质都是从两个表的结果集后在汇总到一张表，但列数行数要一样，而且如果列名不设置，那他会默认第一表的列名，他们两个的区别就是union 会自动去除重复的，浪费磁盘资源，union all 会比较快的查询出结果出来。(注意可以union 的两个表可以括号起来查询出结果集之后再来union的，这样就怎样去判断条件都是没事的了)



1. 如果两个列的行数要合并起来，比如：每个列各3行，合起来共3行的，可以用concat 或者 group\_concat ，然后如果合起来共6行的，那就是用union 或者 union all



1. 在创建表的时候可以采用 : create table tb\_name engine=archive as select \* from my\_table; 然后就会把my\_table 的字段和内容都弄到了 tb\_name ；
2. 引擎 MyISAM、Archive、InnoDB 这三个引擎中，MyISAM 是用于非常频繁的查询 ；Archive是用于存档，因为其压缩了比MyISAM少73%空间，比InnoDB少86%空间，Archive一般都是用于存储日志和历史资料的查询；InnoDB 一般用于更新和并发高的查询，支持事物和行。
3. 在创建索引的时候要注意付出的代价，如果查询的数据时候，索引会自动 进行判断，如果索引找到匹配后，然后就会进行代价判断，如果代价估计数据大于本表的15%以上，就不使用索引了，此时索引的代价就更大。 索引使用总共有 查询的时候mysql自动索引 ；还有另外一个可以强制使用：select \* from tb force index(index\_email)
4. **查询优化：**

如果你使用了 name like “%s”; 这个时候是不会使用到索引的，因为%是通配符，进行全局匹配搜索。

索引能解决锁对整个表的锁定，比如:

创建一个索引 create index idx\_name on tb(‘name’);

在session开始一个窗口：

Begin;

Select \* from tb where name=’张三’;

这个时候在session2 开始一个窗口：

Begin;

Select \* from tb where name=’李四’;

这样就不会因为一个行的事务而影响整个表。

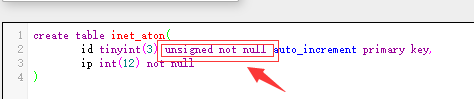
1. 如果在数据过大的时候，比如超过10G，可以采用分区：partition,可以在创建表的时候：

**Create table user( id tinyint(3) not null primary key, name char(30) not null comment ‘姓名’)**

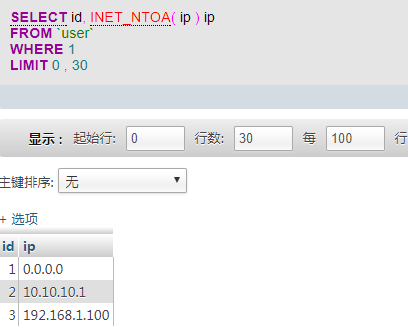
**partition by range(id)( partition id01 values less than (10), partition id02 values less than (20), partition id03 values less than maxvalue);**  或者你可以用list来创建也可以，或者你也可以采用 Hash() 函数来随机分配，固定分区，创建分区，如： **create table tb( a int(10), b int(19)) partition by hash(id); partitions 4; hash会自动创建4个分区，然后会取模%来随机分配(注意 分区4 的partitions 有个s);**

当然你也可以add partition 增加分区，也可以drop partition了。

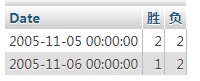
1. 查询分区的内容使用 select \* from tb\_name partition (id01); 其中id01 是自定义的分区名；(**注意这里的mysql版本要求是 5.6以上才可以**).
2. unsigned 要在 not null 前面才行：

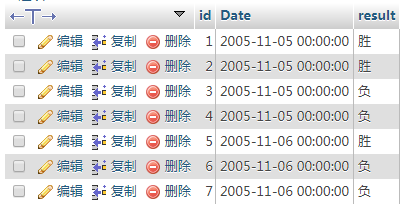


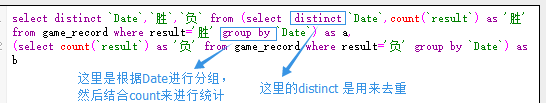
10、在设置要存储IP特殊字段值时要采用，如：ip int(12) unsigned not null,这个时候一定要有 unsigned 不然，128以上的ip段不能正常保存，出现值不一样，insert 数据的时候要用 inet\_aton(‘192.168.1.1’)这样来保存，然后select的时候用 inet\_ntoa(ip) 来读取。



1. 如果遇到下列情况，而且是一张表，或者多张表，可以用 迪尔卡集 来操作：

结果：

表：

使用sql语句: 

1. 行锁(Innodb)，表锁(myisam,memory)，页锁(BDB)浅析：

1、行锁优点：(可以并发处理，一般行锁还是用的比较多)

储存在内存中，运行速度快；  
 能进行针对行的数据进行锁，并出现异常时，可以小数据进行回滚；

行锁缺点：

消耗内存，占据的内存比较大，不适应做那些经常查询的，读操作；

I/O需要的量多，而且容易造成死锁；

2、表锁相对其他两个优点：(可以并发处理，用concurrent\_insert 设置)

现在的myisam的表锁都是默认状态2 (就是可以在读的同时，并发向尾部写入)但在涉及到金额的时候，要加上lock table table\_name write(或者read) 来确保数据的完整性，不然一个表完成了数据操作，另外一个表可能失败，导致数据错误；

除了命令执行SET LOW\_PRIORITY\_UPDATES=1，降低写(更新)的优先级，还可以降低insert、delete的优先级，还可以使用系统提供的 max\_write\_lock\_count设置一个合适的值，来降低写的优先级，让一些读操作有机会获得锁；实在复杂的查询最好安排在闲时，比如夜间；

表锁一般用于读操作，而且严格条件下可以进行更新操作；

不会造成死锁，加锁速度快；

3、BDB页锁是基于这两个锁之间，反应最快的还是表锁。

加锁 Lock table table\_name write(或者 read);

释放锁 Unlock tables;

4、在行锁InnoDB中，行锁的引用是基于索引id的，在数据库中要begin 启用事务,commit 结束事务，其中也可以利用rollback;在提交commit前进行回滚；在行锁中可以使用

Select \* from tb where id=1 for update;来进行排他锁

Select \* from tb where id=1 lock in share mode;来进行共享锁

但记得，都基于事务的开始和id索引

但是在要对表进行复杂操作或者太多行数据更新的话，InnoDB才采用表锁，太多这种操作就建议使用MyISAM表了。

表锁可以采用：

SET AUTOCOMMIT=0;

LOCK TABLES tb WRITE;

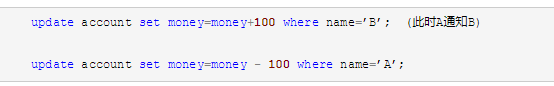
Insert into table\_name …..;

UNLOCK TABLES;

5、间隙锁：就是进行范围查询的时候不匹配的值，成为“间隙(GAP)”,mysql的 InnoDB 会给其加“间隙锁”；如果是在并发的情况下，“间隙锁”会容易造成等待锁太久，所以在业务逻辑尽量用相等的来查询；

6、

脏读：读取未提交的数据，比如转钱 a转了b，然后a 通知b查询，b查询到钱了，但第二条数据 a还没执行，事务还没提交，则实际上 b 也还没收到钱，因为事务没提交，会回滚；



事务的隔离性有四个特点：(结合了Lock锁来执行，不然会在另外个用户可以读取其他事务正在执行的事情的行，虽然没读取带未提交的事务，但不能体现出事务的隔离性，之所以能读到其他事务正在操作行，表，页（但未读取未提交的数据）的数据，是因为Innodb有RedoLog 和 Undo Log 两个日志来实现事务的隔离性；

Redo 是存储执行完成后的事务的数据，也就是commit 后，记录文件是ib\_logfile0 ib\_logfile1；

Undo 是记录未提交commit 事务的数据，默认情况下是写入表空间的；

)

1. 可串行化(Serializable):可避免脏读、不可重复读、幻读的发生；
2. 可重复读(repeatable read):可避免脏读、幻读的发生；
3. 已提交读(Read commited):避免脏读的发生；
4. 未提交读:(Read uncommitted):最低级别，任何情况都无法保证；

可以用show variables like ‘%iso%’ 来查看当前状态，然后可以用 set tx\_isolation=’隔离级别名称（比如：read-uncommitted）’;

1. varchar 和 char 的区别：

Varchar:储存的字符有645个 ，并在255字符内储存时，额外多一个字符，用于少更新，并不知道长度的，或者大于255长度的；

Char：储存字符有255个，固定长度，一般用于身份证，手机号码，姓名等近似值长度的字符串；

8、datetime 、timestamp、date、time的区别：

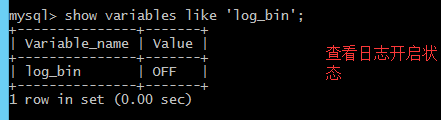
Datetime：不受时区影响，以1000-01-01 到 9999-12-12 这样的格式范围储存；

Timestamp：受时区影响，以1970-01-01 到 2038-12-12 这样的格式范围显示，并且行更新后，第一个的timestamp会自动更新；

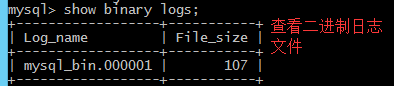
Date： 是显示 YYYY-mm-dd 这样的格式，一般用来保存生日(建议用这种，不要用字符串来用)；

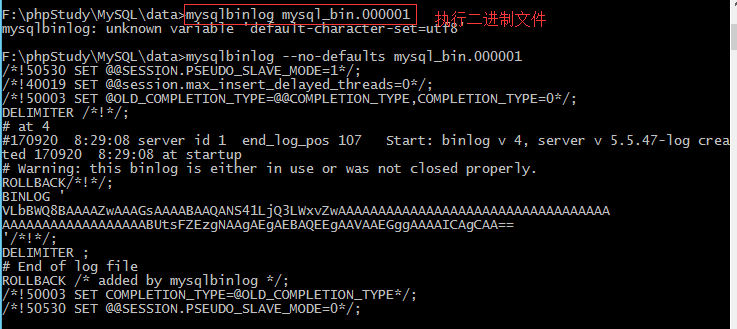
Time: 是显示HH:mm:ss 这样的格式来显示

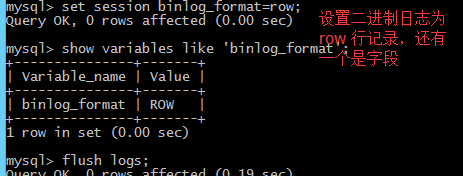
1. 二进制日志



在mysqld配置项下面加上log\_bin=mysql\_bin，然后重启就可以了







记得在MySQL5.7以后开始，有个binlog\_row\_image=full,[minimal] 来进行控制储存二进制日志来记录是修改一列，但保存20列(就一张表里的，无关的也记录)；也有只记录修改列的；设置此参数

set binlog\_row\_image ='minimal';

1. 主从服务器：
2. 在DB服务器上建立复制账号：

CREATE USER ‘repl’ @’IP段’ identified by ‘password’;

GRANT REPLICATION SLAVE ON \*.\* TO ‘repl’ @’IP段’;

1. 复制配置步骤：

配置主数据库服务器：

bin\_log = mysql-bin

server\_id = 100

配置从数据库服务器：

bin\_log = mysql-bin

server\_id = 101

relay\_log = mysql-relay-bin（这里是中心日志的名字）

log\_slave\_update = on [可选]

read\_only = on [可选]

1. 初始化从服务器数据：

Mysqldump --master-data=2-single-transaction

Xtrabackup --slave-info

1. 启动复制连路

CHANGE MASTER TO MASTER\_HOST = ‘master\_host\_ip’,

MASTER\_USER = ’repl’,

MASTER\_PASSWORD= ‘password’,

MASTER\_LOG\_FILE=’mysql\_log\_file\_name’,

MASTER\_LOG\_POS=4;