wo<http://www.cnblogs.com/yunf/archive/2011/04/12/2013448.html>

## Mac安装

### Mac

下载dmg格式版本后，安装

参考文档http://blog.csdn.net/adwfcu/article/details/7720848

### 启动和停止命令

sudo /Library/StartupItems/MySQLCOM/MySQLCOM start，然后输入你的系统管理员密码即可。

sudo /Library/StartupItems/MySQLCOM/MySQLCOM stop，然后输入你的系统管理员密码即可。

【注意：Mac Yesemite 10.10 通过上诉命令失败，可以利用launchctl自动启mysql】

Launchctl的作用和Linux的Cronttab一样，参考http://my.oschina.net/huyuaning/blog/334584

如果下面这种方式启动，在10.10上用10.8和10.9都关系不大。我用的就是10.8版本

1、进入/Library/LaunchDaemons目录

2、sudo touch com.mysql.plist

3、vim com.mysql.plist, 增加下面文字

<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<plist version="1.0">

<dict>

<key>Label</key>

<string>com.mysql</string>

<key>ProgramArguments</key>

<array>

<string>/usr/local/mysql/bin/mysqld\_safe</string>

</array>

<key>RunAtLoad</key>

<true/>

<key>KeepAlive</key>

<false/>

</dict>

</plist>

如果文件直接拷贝过来的，要修改这个文件必须属于root用户，wheel组，用chown修改

sudo chown root:wheel com.mysql.plist

4、查看文件格式是否有问题

sudo plutil -lint com.mysql.plist

5、最后调用一下launchctl命令，加载这个plist,

launchctl load -w com.mysql.plist

如果不想开机启动，则在当前目录下输入

launchctl unload -w com.mysql.plist

### 更改mysql root账户密码

/usr/local/mysql/bin/mysqladmin -u root password 新密码

终端中输入命令：/usr/local/mysql/bin/mysql即可。

或者直接：mysql –uroot –p123456

### Mac下MySQL卸载方法

http://www.cnblogs.com/TsengYuen/archive/2011/12/06/2278574.html

Mysql客户端

http://xmong.iteye.com/blog/1911351

# Mysql语句

## 基本

### 启停数据库

启动

D:\30SCM\MyCat\MySQl\mysql-advanced-5.6.18-winx64\bin>mysqld –console

停止

D:\30SCM\MyCat\MySQl\mysql-advanced-5.6.18-winx64\bin>mysqladmin -uroot shutdown

1. [client] # 与mysql连接的客户端（其中包括我们编写的应用程序）

default-character-set = utf8

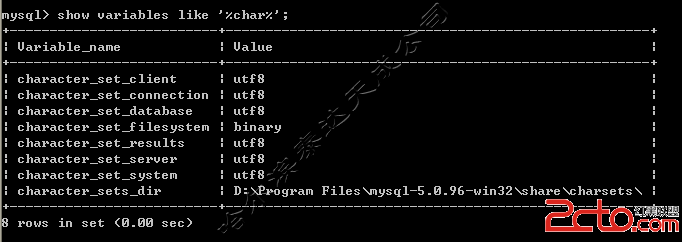
2. [mysqld] # 服务器相关设置

character\_set\_server= utf8 # 兼容V5.5设置

default-collation= utf8\_general\_ci

查看字符集

show variables like '%char%';



### 连接数据数据库

本地连接

mysql -uroot–p

远程主机

mysql -u root -p -h 192.168.0.41

### 创建、查看、删除数据库

查看系统中存在的数据库

show databases;

创建数据库

CREATE DATABASE ***dbname***

创建数据库[指定编码]

GBK: create database test2 DEFAULT CHARACTER SET gbk COLLATE gbk\_chinese\_ci;

UTF8: CREATE DATABASE `test2` DEFAULT CHARACTER SET utf8 COLLATE utf8\_general\_ci;

选择要操作的数据库

use***dbname***

删除数据库

drop database ***dbname***

### 备份sql server

--- 创建 备份数据的 device

USE master

EXEC sp\_addumpdevice 'disk', 'testBack', 'c:\mssql7backup\MyNwind\_1.dat'

--- 开始 备份

BACKUP DATABASE pubs TO testBack

### 创建新表

create table tabname(col1 type1 [not null] [primary key],col2 type2 [not null],..)

根据已有的表创建新表：

A：create table tab\_new like tab\_old (使用旧表创建新表)

B：create table tab\_new as select col1,col2… from tab\_old definition only

### 【显示建表语句】

show create table tabname;

### 删除新表

drop table tabname

### 增加一个列

Alter table tabname add column col type

注：列增加后将不能删除。DB2中列加上后数据类型也不能改变，唯一能改变的是增加varchar类型的长度。

### 添加主键： Alter table tabname add primary key(col)

删除主键： Alter table tabname drop primary key(col)

### 创建索引：create [unique] index idxname on tabname(col….)

删除索引：drop index idxname

注：索引是不可更改的，想更改必须删除重新建。

### 创建视图：create view viewname as select statement

删除视图：drop view viewname

### 几个简单的基本的sql语句

选择：select \* from table1 where 范围

插入：insert into table1(field1,field2) values(value1,value2)

删除：delete from table1 where 范围

更新：update table1 set field1=value1 where 范围

查找：select \* from table1 where field1 like ’%value1%’ ---like的语法很精妙，查资料!

排序：select \* from table1 order by field1,field2 [desc]

总数：select count as totalcount from table1

求和：select sum(field1) as sumvalue from table1

平均：select avg(field1) as avgvalue from table1

最大：select max(field1) as maxvalue from table1

最小：select min(field1) as minvalue from table1

### 几个高级查询运算词

A： UNION 运算符

UNION 运算符通过组合其他两个结果表（例如 TABLE1 和 TABLE2）并消去表中任何重复行而派生出一个结果表。当 ALL 随 UNION 一起使用时（即 UNION ALL），不消除重复行。两种情况下，派生表的每一行不是来自 TABLE1 就是来自 TABLE2。

B： EXCEPT 运算符

EXCEPT 运算符通过包括所有在 TABLE1 中但不在 TABLE2 中的行并消除所有重复行而派生出一个结果表。当 ALL 随 EXCEPT 一起使用时 (EXCEPT ALL)，不消除重复行。

C： INTERSECT 运算符

INTERSECT 运算符通过只包括 TABLE1 和 TABLE2 中都有的行并消除所有重复行而派生出一个结果表。当 ALL 随 INTERSECT 一起使用时 (INTERSECT ALL)，不消除重复行。

注：使用运算词的几个查询结果行必须是一致的。

### 使用外连接

A、left （outer） join：

左外连接（左连接）：结果集几包括连接表的匹配行，也包括左连接表的所有行。

SQL: select a.a, a.b, a.c, b.c, b.d, b.f from a LEFT OUT JOIN b ON a.a = b.c

B：right （outer） join:

右外连接(右连接)：结果集既包括连接表的匹配连接行，也包括右连接表的所有行。

C：full/cross （outer） join：

全外连接：不仅包括符号连接表的匹配行，还包括两个连接表中的所有记录。

### 分组:Group by:

  一张表，一旦分组完成后，查询后只能得到组相关的信息。

 组相关的信息：（统计信息） count,sum,max,min,avg  分组的标准)

    在SQLServer中分组时：不能以text,ntext,image类型的字段作为分组依据

 在selecte统计函数中的字段，不能和普通的字段放在一起；

### 对数据库进行操作

 分离数据库： sp\_detach\_db; 附加数据库：sp\_attach\_db 后接表明，附加需要完整的路径名

14.如何修改数据库的名称:

sp\_renamedb 'old\_name', 'new\_name'

### 修改用户名密码

update mysql.user set password =password('123456') where user = 'root';

flush privileges;

### 时间格式

注意：与Oracle不同mysql的timestamp不支持毫秒。

并且格式上：

Java：“yyyy-MM-dd HH:mm:ss.SSS”

Oracle不区分大小写: 'YYYY-MM-DD HH24:MI:SS.FF3';

Mysql与Unix相识：'%Y-%m-%d %H:%i:%s'

一、TIMESTAMP

显示格式：YYYY-MM-DD HH:MM:SS

时间范围:[ '1970-01-01 00:00:00'到'2037-12-31 23:59:59']

TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  在创建新记录和修改现有记录的时候都对这个数据列刷新。

TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  在创建新记录的时候把这个字段设置为当前时间，但以后修改时，不再刷新它。

TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  在创建新记录的时候把这个字段设置为0，以后修改时刷新它。

TIMESTAMP DEFAULT ‘yyyy-mm-dd hh:mm:ss’ ON UPDATE CURRENT\_TIMESTAMP  在创建新记录的时候把这个字段设置为给定值，以后修改时刷新它

1、TIMESTAMP列不为空时，默认值可以为“0000-00-00 00:00:00”，但不能为null。

2、一个表可以存在多个TIMESTAMP列，但一个表只有一个TIMESTAMP类型的字段可以在默认值或者UPDATE部分用CURRENT\_TIMESTAMP，即设置为数据更新而改变为数据库系统当前值。

3、TIMESTAMP列的默认值是CURRENT\_TIMESTAMP常量值。当纪录数据发生变化的时候，TIMESTAMP列会自动将其值设定为CURRENT\_TIMESTAMP。

4、TIMESTAMP列创建后的格式是：

ALTER TABLE `course`

ADD COLUMN `birthday`  timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP;

ALTER TABLE `course`

ADD COLUMN `birthday`  timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' ON UPDATE CURRENT\_TIMESTAMP ;

ALTER TABLE `course`

ADD COLUMN `birthday`  timestamp NULL AFTER `cname`;

二、DATETIME

显示格式：YYYY-MM-DD HH:MM:SS

时间范围:[ '1000-01-01 00:00:00'到'9999-12-31 23:59:59']

三、DATE

显示格式：YYYY-MM-DD

时间范围：['1000-01-01'到'9999-12-31']

四、日期格式转换

1、字符串转日期

select STR\_TO\_DATE('2013-01-29 13:49:18', '%Y-%m-%d %H:%i:%s')

2、日期转字符串

select DATE\_FORMAT('2013-01-29 13:49:18', '%Y-%m-%d %H:%i:%s')

五、日期的中常用的年月日时分秒星期月份等获取方法

select TIMESTAMP('2013-01-29 13:50:27');

select DATE('2013-01-29 13:50:27');

select YEAR('2013-01-29 13:50:27');

select MONTH(('2013-01-29 13:50:27');

select WEEK('2013-01-29 13:50:27');

select DAY('2013-01-29 13:50:27');

select TIME('2013-01-29 13:50:27');

select CURTIME();

select CURDATE();

select CURRENT\_DATE;

select CURRENT\_TIME;

select CURRENT\_TIMESTAMP;

select NOW()

六、日期的运算：

SELECT DATE\_ADD('2013-01-29 13:50:27', INTERVAL 1 DAY);

                -> '2013-01-30 13:50:27'

SELECT DATE\_ADD('2013-01-29 13:50:27', INTERVAL 1 HOUR);

                -> '2013-01-29 14:50:27'

SELECT DATE\_ADD('2013-01-29 13:50:27', INTERVAL 1 MONTH);

               -> '2013-02-28 13:50:27'

## 状态监控

### QPS

QPS(每秒 Query 量 ) 里 的 QPS 实际上是指 MySQL Server 每秒执行的 Query总量,在 MySQL 5.1.30 及以下版本可以通过 Questions 状态值每秒内的变化量来近似表示,而从 MySQL 5.1.31 开始,则可以通过 Queries 来表示。Queries 是在 MySQL 5.1.31 才新增的状态变量。主要解决的问题就是 Questions 状态变量并没有记录存储过程中所执行的 Query(当然,在无存储过程的老版本 MySQL 中则不存在这个区别),而 Queries 状态变量则会记录。二者获取方式:  
QPS = Questions(or Queries) / Seconds  
获取所需状态变量值:  
SHOW /\*!50000 GLOBAL \*/ STATUS LIKE 'Questions'  
SHOW /\*!50000 GLOBAL \*/ STATUS LIKE 'Queries'

如下：

mysql> SHOW /\*!50000 GLOBAL \*/ STATUS LIKE 'Questions';

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| Questions | 8367 |

+---------------+-------+

1 row in set (0.00 sec)

mysql> select NOW();

+---------------------+

| NOW() |

+---------------------+

| 2014-09-04 09:40:46 |

+---------------------+

1 row in set (0.00 sec)

mysql> SHOW /\*!50000 GLOBAL \*/ STATUS LIKE 'Questions';

+---------------+-------+

| Variable\_name | Value |

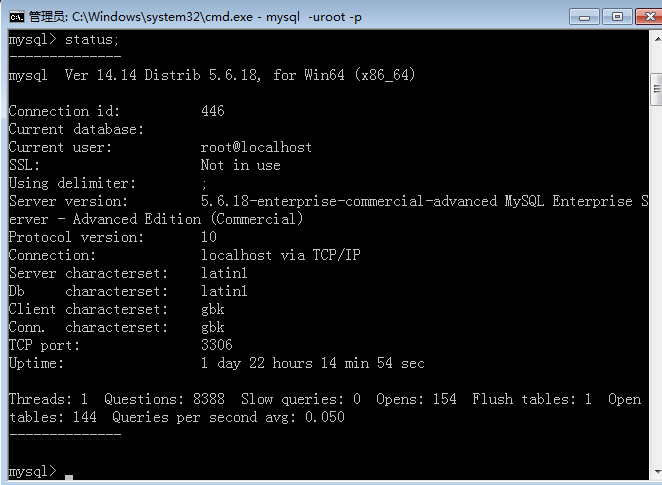
+---------------+-------+

| Questions | 8369 |

+---------------+-------+

1 row in set (0.00 sec)

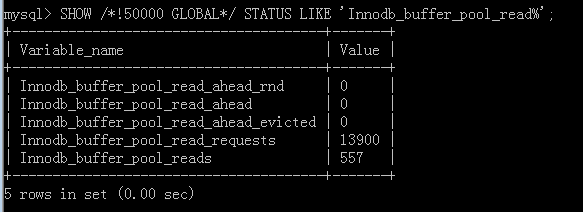
或者用status查看



### Innodb Buffer 命中率

这里 Innodb Buffer 所指的是 innodb\_buffer\_pool,也就是用来缓存 Innodb 类型表的数据和索引的内存空间。类似 Key buffer,我们同样可以根据 MySQL Server 提供的相应状态值计算出其命中率:  
innodb\_buffer\_read\_hits=(1-Innodb\_buffer\_pool\_reads/Innodb\_buffer\_pool\_read\_requests) \* 100%

SHOW /\*!50000 GLOBAL\*/ STATUS



### Thread Cache 命中率

Thread Cache 命中率:Thread Cache 命中率能够直接反应出我们的系统参数，thread\_cache\_size 设置的是否合理。一个合理的 thread\_cache\_size 参数能够节约大量创建新连接时所需要消耗的资源。Thread Cache 命中率计算方式如下:

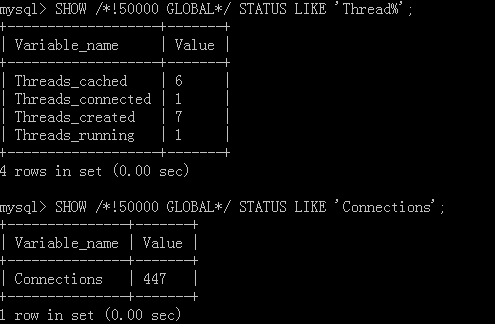
Thread\_cache\_hits = (1 - Threads\_created / Connections) \* 100%

获取所需状态变量值:

SHOW /\*!50000 GLOBAL\*/ STATUS LIKE 'Thread%';

SHOW /\*!50000 GLOBAL\*/ STATUS LIKE 'Connections';

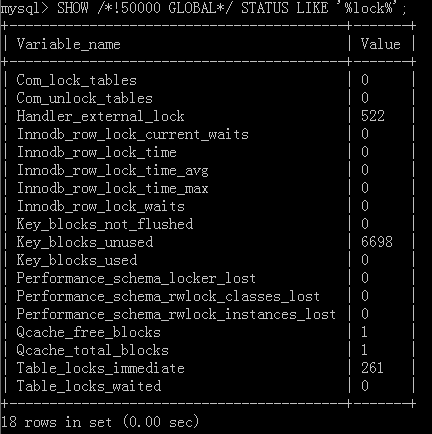
正常来说,Thread Cache 命中率要在 90% 以上才算比较合理。



### 锁定状态

锁定状态包括表锁和行锁两种,我们可以通过系统状态变量获得锁定总次数,锁定造成其他线程等待的次数,以及锁定等待时间信息。

SHOW /\*!50000 GLOBAL\*/ STATUS LIKE '%lock%';



可以清晰的了解到系统整体的锁定是否严重。如当 Table\_locks\_waited 与Table\_locks\_immediate 的比值较大,则说明我们的表锁造成的阻塞比较严重,可能需要调整 Query 语句,或者更改存储引擎,亦或者需要调整业务逻辑等。

而 Innodb\_row\_lock\_waits 较大,则说明 Innodb 的行锁也比较严重,且影响了其他线程的正常处理。同样需要查找出原因并解决。造成 Innodb 行锁严重的原因可能是 Query 语句所利用的索引不够合理(Innodb 行锁是基于索引来锁定的),造成间隙锁过大等。

### 复制延时量

复制延时量将直接影响了 Slave 数据库处于不一致状态的时间长短。如果我们是通过 Slave 来提供读服务,就不得不重视这个延时量。我们可以通过在 Slave 节点上执行"SHOW SLAVE STATUS"命 令, Seconds\_Behind\_Master 项

取的值来了解 Slave 当前的延时量(单位:秒)

### Tmp table 状况

Tmp Table 的状况主要是用于监控 MySQL 使用临时表的量是否过多,是否有临时表过大而不得不从内存中换出到磁盘文件上。临时表使用状态信息可以通过如下方式获得:

SHOW /\*!50000 GLOBAL\*/ STATUS LIKE 'Created\_tmp%';

| Created\_tmp\_disk\_tables | 1|

Created\_tmp\_tables|46|

从上面的状态信息可以了解到系统使用了 46 次临时表,其中有 1 次临时表比较大,无法在内存中完成,而不得不使用到磁盘文件。如果 Created\_tmp\_tables 非常大 ,则可能是系统中排序操作过多,或者是表连接方式不是很优化。而如果是Created\_tmp\_disk\_tables 与 Created\_tmp\_tables 的比率过高,如超过 10%,则我们需要考虑是否 tmp\_table\_size 这个系统参数所设置的足够大。当然,如果系统内存有限,也就没有太多好的解决办法了。

### show engine innodb status \G

show engine innodb status \G

这里的[\G 用法：查询结果按列打印](http://www.cnblogs.com/samren/archive/2012/09/15/2686869.html)

### MYSQL查询优化：profile功能

<http://www.cnblogs.com/adforce/archive/2012/06/02/2532287.html>

这里还需要注意一点就是，需要安装profile模块才能实现。

1、不过版本要在5.0.37之后。(SHOW PROFILES and SHOW PROFILE were added in MySQL 5.0.37.)

SELECT **@@profiling**;

来查看是否已经启用profile，如果profilng值为0，可以通过

SET profiling = **1**;

来启用。启用profiling之后，我们执行一条查询语句，比如：

select count(\*) from roi\_summary;

然后show profiles查看如下：

+----------------+------------+----------------------------------+

| Query\_ID | Duration | Query |

+----------------+------------+----------------------------------+

| **1** | **0.00021500** | select **@@profiling** |

| **2** | **0.05522700** | select count(\*) from roi\_summary |

+----------------+------------+----------------------------------+

2 rows in set (0.00 sec)

其中ID为5的语句是刚执行的查询语句

2、变量profiling是用户变量，每次都得重新启用。

### 通过genaral\_log跟踪SQL语句

1. 检查genaral\_log是否已经打开

show global variables like '%log%';

|  |
| --- |
| mysql> show global variables like '%log%';  +-----------------------------------------+-------------------------------------+  | Variable\_name | Value |  +-----------------------------------------+-------------------------------------+  | back\_log | 250 |  | binlog\_cache\_size | 32768 |  | binlog\_checksum | CRC32 |  | binlog\_direct\_non\_transactional\_updates | OFF |  | binlog\_format | STATEMENT |  | binlog\_max\_flush\_queue\_time | 0 |  | binlog\_order\_commits | ON |  | binlog\_row\_image | FULL |  | binlog\_rows\_query\_log\_events | OFF |  | binlog\_stmt\_cache\_size | 32768 |  | binlogging\_impossible\_mode | IGNORE\_ERROR |  | expire\_logs\_days | 0 |  | general\_log | ON |  | general\_log\_file | mysql.log |  | innodb\_api\_enable\_binlog | OFF |  | innodb\_flush\_log\_at\_timeout | 1 |  | innodb\_flush\_log\_at\_trx\_commit | 1 | |

1. 如果没有打开，修改mysql的配置文件：   
   添加下面几行：

shell>vi /usr/local/mysql/my.cnf

log-output=FILE

general-log=1

general\_log\_file="mysql.log"

slow-query-log=1

slow\_query\_log\_file="mysql-slow.log"

long\_query\_time=10

然后重启数据库：*启动和停止命令*

3、然后读取general-log就可以跟踪SQL了

sudo tail -f /usr/local/mysql/data/mysql.log

## 二、提升

### 1、复制表(只复制结构,源表名：a 新表名：b) (Access可用)

法一：select \* into b from a where 1<>1（仅用于SQlServer）

法二：select top 0 \* into b from a

### 2、拷贝表(拷贝数据,源表名：a 目标表名：b) (Access可用)

insert into b(a, b, c) select d,e,f from b;

### 3、跨数据库之间表的拷贝(具体数据使用绝对路径) (Access可用)

insert into b(a, b, c) select d,e,f from b in ‘具体数据库’ where 条件

例子：..from b in '"&Server.MapPath(".")&"\data.mdb" &"' where..

### 4、子查询(表名1：a 表名2：b)

select a,b,c from a where a IN (select d from b ) 或者: select a,b,c from a where a IN (1,2,3)

### 5、显示文章、提交人和最后回复时间

select a.title,a.username,b.adddate from table a,(select max(adddate) adddate from table where table.title=a.title) b

### 6、外连接查询(表名1：a 表名2：b)

select a.a, a.b, a.c, b.c, b.d, b.f from a LEFT OUT JOIN b ON a.a = b.c

### 7、在线视图查询(表名1：a )

select \* from (SELECT a,b,c FROM a) T where t.a> 1;

### 8、between的用法,between限制查询数据范围时包括了边界值,not between不包括

select \* from table1 where time between time1 and time2

select a,b,c, from table1 where a not between 数值1 and 数值2

### 9、in 的使用方法

select \* from table1 where a [not] in (‘值1’,’值2’,’值4’,’值6’)

### 10、两张关联表，删除主表中已经在副表中没有的信息

delete from table1 where not exists ( select \* from table2 where table1.field1=table2.field1 )

### 11、四表联查问题：

select \* from a left inner join b on a.a=b.b right inner join c on a.a=c.c inner join d on a.a=d.d where .....

### 12、日程安排提前五分钟提醒

SQL: select \* from 日程安排 where datediff('minute',f开始时间,getdate())>5

### 13、一条sql 语句搞定数据库分页

select top 10 b.\* from (select top 20 主键字段,排序字段 from 表名 order by 排序字段 desc) a,表名 b where b.主键字段 = a.主键字段 order by a.排序字段

具体实现：

关于数据库分页：

  declare @start int,@end int

  @sql  nvarchar(600)

  set @sql=’select top’+str(@end-@start+1)+’+from T where rid not in(select top’+str(@str-1)+’Rid from T where Rid>-1)’

  exec sp\_executesql @sql

注意：在top后不能直接跟一个变量，所以在实际应用中只有这样的进行特殊的处理。Rid为一个标识列，如果top后还有具体的字段，这样做是非常有好处的。因为这样可以避免 top的字段如果是逻辑索引的，查询的结果后实际表中的不一致（逻辑索引中的数据有可能和数据表中的不一致，而查询时如果处在索引则首先查询索引）

### 14、前10条记录

select top 10 \* form table1 where 范围

### 15、选择在每一组b值相同的数据中对应的a最大的记录的所有信息(类似这样的用法可以用于论坛每月排行榜,每月热销产品分析,按科目成绩排名,等等.)

select a,b,c from tablename ta where a=(select max(a) from tablenametb where tb.b=ta.b)

### 16、包括所有在 TableA 中但不在 TableB和TableC 中的行并消除所有重复行而派生出一个结果表

(select a from tableA ) except (select a from tableB) except (select a from tableC)

### 17、随机取出10条数据

select top 10 \* from tablename order by newid()

### 18、随机选择记录

select newid()

### 19、删除重复记录

1),delete from tablename where id not in (select max(id) from tablename group by col1,col2,...)

2),select distinct \* into temp from tablename

  delete from tablename

  insert into tablename select \* from temp

评价： 这种操作牵连大量的数据的移动，这种做法不适合大容量但数据操作

3),例如：在一个外部表中导入数据，由于某些原因第一次只导入了一部分，但很难判断具体位置，这样只有在下一次全部导入，这样也就产生好多重复的字段，怎样删除重复字段

alter table tablename

--添加一个自增列

add  column\_bint identity(1,1)

 delete from tablename where column\_b not in(

select max(column\_b)  from tablename group by column1,column2,...)

alter table tablename drop column column\_b

### 20、列出数据库里所有的表名

select name from sysobjects where type='U' // U代表用户

### 21、列出表里的所有的列名

select name from syscolumns where id=object\_id('TableName')

### 22、列示type、vender、pcs字段，以type字段排列，case可以方便地实现多重选择，类似select 中的case。

select type,sum(case vender when 'A' then pcs else 0 end),sum(case vender when 'C' then pcs else 0 end),sum(case vender when 'B' then pcs else 0 end) FROM tablename group by type

显示结果：

type vender pcs

电脑 A 1

电脑 A 1

光盘 B 2

光盘 A 2

手机 B 3

手机 C 3

### 23、初始化表table1

TRUNCATE TABLE table1

### 24、选择从10到15的记录

select top 5 \* from (select top 15 \* from table order by id asc) table\_别名 order by id desc

## 三、技巧

### 1、1=1，1=2的使用，在SQL语句组合时用的较多

“where 1=1” 是表示选择全部    “where 1=2”全部不选，

如：

if @strWhere !=''

begin

set @strSQL = 'select count(\*) as Total from [' + @tblName + '] where ' + @strWhere

end

else

begin

set @strSQL = 'select count(\*) as Total from [' + @tblName + ']'

end

我们可以直接写成

错误！未找到目录项。

set @strSQL = 'select count(\*) as Total from [' + @tblName + '] where 1=1 安定 '+ @strWhere 2、收缩数据库

--重建索引

DBCC REINDEX

DBCC INDEXDEFRAG

--收缩数据和日志

DBCC SHRINKDB

DBCC SHRINKFILE

### 3、压缩数据库

dbccshrinkdatabase(dbname)

### 4、转移数据库给新用户以已存在用户权限

exec sp\_change\_users\_login 'update\_one','newname','oldname'

go

### 5、检查备份集

RESTORE VERIFYONLY from disk='E:\dvbbs.bak'

### 6、修复数据库

ALTER DATABASE [dvbbs] SET SINGLE\_USER

GO

DBCC CHECKDB('dvbbs',repair\_allow\_data\_loss) WITH TABLOCK

GO

ALTER DATABASE [dvbbs] SET MULTI\_USER

GO

### 7、日志清除

SET NOCOUNT ON

DECLARE @LogicalFileNamesysname,

 @MaxMinutes INT,

 @NewSize INT

USE tablename -- 要操作的数据库名

SELECT  @LogicalFileName = 'tablename\_log', -- 日志文件名

@MaxMinutes = 10, -- Limit on time allowed to wrap log.

 @NewSize = 1  -- 你想设定的日志文件的大小(M)

Setup / initialize

DECLARE @OriginalSizeint

SELECT @OriginalSize = size

 FROM sysfiles

 WHERE name = @LogicalFileName

SELECT 'Original Size of ' + db\_name() + ' LOG is ' +

 CONVERT(VARCHAR(30),@OriginalSize) + ' 8K pages or ' +

 CONVERT(VARCHAR(30),(@OriginalSize\*8/1024)) + 'MB'

 FROM sysfiles

 WHERE name = @LogicalFileName

CREATE TABLE DummyTrans

 (DummyColumn char (8000) not null)

DECLARE @Counter    INT,

 @StartTime DATETIME,

 @TruncLog   VARCHAR(255)

SELECT @StartTime = GETDATE(),

 @TruncLog = 'BACKUP LOG ' + db\_name() + ' WITH TRUNCATE\_ONLY'

DBCC SHRINKFILE (@LogicalFileName, @NewSize)

EXEC (@TruncLog)

-- Wrap the log if necessary.

WHILE @MaxMinutes> DATEDIFF (mi, @StartTime, GETDATE()) -- time has not expired

 AND @OriginalSize = (SELECT size FROM sysfiles WHERE name = @LogicalFileName)

 AND (@OriginalSize \* 8 /1024) > @NewSize

 BEGIN -- Outer loop.

SELECT @Counter = 0

 WHILE   ((@Counter < @OriginalSize / 16) AND (@Counter < 50000))

 BEGIN -- update

 INSERT DummyTrans VALUES ('Fill Log') DELETE DummyTrans

 SELECT @Counter = @Counter + 1

 END

 EXEC (@TruncLog)

 END

SELECT 'Final Size of ' + db\_name() + ' LOG is ' +

 CONVERT(VARCHAR(30),size) + ' 8K pages or ' +

 CONVERT(VARCHAR(30),(size\*8/1024)) + 'MB'

 FROM sysfiles

 WHERE name = @LogicalFileName

DROP TABLE DummyTrans

SET NOCOUNT OFF

### 8、更改某个表

exec sp\_changeobjectowner 'tablename','dbo'

### 9、存储更改全部表

CREATE PROCEDURE dbo.User\_ChangeObjectOwnerBatch

@OldOwner as NVARCHAR(128),

@NewOwner as NVARCHAR(128)

AS

DECLARE @Name    as NVARCHAR(128)

DECLARE @Owner   as NVARCHAR(128)

DECLARE @OwnerName   as NVARCHAR(128)

DECLARE curObject CURSOR FOR

select 'Name'    = name,

   'Owner'    = user\_name(uid)

from sysobjects

where user\_name(uid)=@OldOwner

order by name

OPEN   curObject

FETCH NEXT FROM curObject INTO @Name, @Owner

WHILE(@@FETCH\_STATUS=0)

BEGIN

if @Owner=@OldOwner

begin

   set @OwnerName = @OldOwner + '.' + rtrim(@Name)

   exec sp\_changeobjectowner @OwnerName, @NewOwner

end

-- select @name,@NewOwner,@OldOwner

FETCH NEXT FROM curObject INTO @Name, @Owner

END

close curObject

deallocate curObject

GO

### 10、SQL SERVER中直接循环写入数据

declare @iint

set @i=1

while @i<30

begin

    insert into test (userid) values(@i)

    set @i=@i+1

end

案例：

有如下表，要求就裱中所有沒有及格的成績，在每次增長0.1的基礎上，使他們剛好及格:

 Name     score

 Zhangshan 80

 Lishi       59

 Wangwu      50

 Songquan 69

while((select min(score) from tb\_table)<60)

begin

update tb\_table set score =score\*1.01

where score<60

if  (select min(score) from tb\_table)>60

  break

 else

    continue

end

## 数据开发-经典

### 1.按姓氏笔画排序:

Select \* From TableName Order By CustomerName Collate Chinese\_PRC\_Stroke\_ci\_as //从少到多

### 2.数据库加密:

select encrypt('原始密码')

select pwdencrypt('原始密码')

select pwdcompare('原始密码','加密后密码') = 1--相同；否则不相同 encrypt('原始密码')

select pwdencrypt('原始密码')

select pwdcompare('原始密码','加密后密码') = 1--相同；否则不相同

### 3.取回表中字段:

declare @list varchar(1000),

@sqlnvarchar(1000)

select @list=@list+','+b.name from sysobjectsa,syscolumns b where a.id=b.id and a.name='表A'

set @sql='select '+right(@list,len(@list)-1)+' from 表A'

exec (@sql)

### 4.查看硬盘分区:

EXEC master..xp\_fixeddrives

### 5.比较A,B表是否相等:

if (select checksum\_agg(binary\_checksum(\*)) from A)

     =

    (select checksum\_agg(binary\_checksum(\*)) from B)

print '相等'

else

print '不相等'

### 6.杀掉所有的事件探察器进程:

DECLARE hcforeach CURSOR GLOBAL FOR SELECT 'kill '+RTRIM(spid) FROM master.dbo.sysprocesses

WHERE program\_name IN('SQL profiler',N'SQL 事件探查器')

EXEC sp\_msforeach\_worker '?'

### 7.记录搜索:

开头到N条记录

Select Top N \* From 表

-------------------------------

N到M条记录(要有主索引ID)

Select Top M-N \* From 表 Where ID in (Select Top M ID From 表) Order by ID   Desc

----------------------------------

N到结尾记录

Select Top N \* From 表 Order by ID Desc

案例

例如1：一张表有一万多条记录，表的第一个字段 RecID 是自增长字段， 写一个SQL语句，找出表的第31到第40个记录。

 select top 10 recid from A where recid not  in(select top 30 recid from A)

分析：如果这样写会产生某些问题，如果recid在表中存在逻辑索引。

 select top 10 recid from A where……是从索引中查找，而后面的select top 30 recid from A则在数据表中查找，这样由于索引中的顺序有可能和数据表中的不一致，这样就导致查询到的不是本来的欲得到的数据。

解决方案

1， 用order by select top 30 recid from A order by ricid 如果该字段不是自增长，就会出现问题

2， 在那个子查询中也加条件：select top 30 recid from A where recid>-1

例2：查询表中的最后以条记录，并不知道这个表共有多少数据,以及表结构。

set @s = 'select top 1 \* from T   where pid not in (select top ' + str(@count-1) + ' pid  from  T)'

print @s      exec  sp\_executesql  @s

### 9：获取当前数据库中的所有用户表

select Name from sysobjects where xtype='u' and status>=0

### 10：获取某一个表的所有字段

select name from syscolumns where id=object\_id('表名')

select name from syscolumns where id in (select id from sysobjects where type = 'u' and name = '表名')

两种方式的效果相同

### 11：查看与某一个表相关的视图、存储过程、函数

select a.\* from sysobjects a, syscomments b where a.id = b.id and b.text like '%表名%'

### 12：查看当前数据库中所有存储过程

select name as 存储过程名称 from sysobjects where xtype='P'

### 13：查询用户创建的所有数据库

select \* from master..sysdatabases D where sid not in(select sid from master..syslogins where name='sa')

或者

select dbid, name AS DB\_NAME from master..sysdatabases where sid<> 0x01

### 14：查询某一个表的字段和数据类型

select column\_name,data\_type from information\_schema.columns

where table\_name = '表名'

### 15：不同服务器数据库之间的数据操作

--创建链接服务器

exec sp\_addlinkedserver   'ITSV ', ' ', 'SQLOLEDB ', '远程服务器名或ip地址 '

exec sp\_addlinkedsrvlogin  'ITSV ', 'false ',null, '用户名 ', '密码 '

--查询示例

select \* from ITSV.数据库名.dbo.表名

--导入示例

select \* into 表 from ITSV.数据库名.dbo.表名

--以后不再使用时删除链接服务器

exec sp\_dropserver  'ITSV ', 'droplogins '

--连接远程/局域网数据(openrowset/openquery/opendatasource)

--1、openrowset

--查询示例

select \* from openrowset( 'SQLOLEDB ', 'sql服务器名 '; '用户名 '; '密码 ',数据库名.dbo.表名)

--生成本地表

select \* into 表 from openrowset( 'SQLOLEDB ', 'sql服务器名 '; '用户名 '; '密码 ',数据库名.dbo.表名)

--把本地表导入远程表

insert openrowset( 'SQLOLEDB ', 'sql服务器名 '; '用户名 '; '密码 ',数据库名.dbo.表名)

select \*from 本地表

--更新本地表

update b

set b.列A=a.列A

 from openrowset( 'SQLOLEDB ', 'sql服务器名 '; '用户名 '; '密码 ',数据库名.dbo.表名)as a inner join 本地表 b

on a.column1=b.column1

--openquery用法需要创建一个连接

--首先创建一个连接创建链接服务器

exec sp\_addlinkedserver   'ITSV ', ' ', 'SQLOLEDB ', '远程服务器名或ip地址 '

--查询

select \*

FROM openquery(ITSV,  'SELECT \*  FROM 数据库.dbo.表名 ')

--把本地表导入远程表

insert openquery(ITSV,  'SELECT \*  FROM 数据库.dbo.表名 ')

select \* from 本地表

--更新本地表

update b

set b.列B=a.列B

FROM openquery(ITSV,  'SELECT \* FROM 数据库.dbo.表名 ') as a

inner join 本地表 b on a.列A=b.列A

--3、opendatasource/openrowset

SELECT   \*

FROM   opendatasource( 'SQLOLEDB ',  'Data Source=ip/ServerName;User ID=登陆名;Password=密码 ' ).test.dbo.roy\_ta

--把本地表导入远程表

insert opendatasource( 'SQLOLEDB ',  'Data Source=ip/ServerName;User ID=登陆名;Password=密码 ').数据库.dbo.表名

select \* from 本地表

SQL Server基本函数

SQL Server基本函数

1.字符串函数 长度与分析用

1,datalength(Char\_expr) 返回字符串包含字符数,但不包含后面的空格

2,substring(expression,start,length) 取子串，字符串的下标是从“1”，start为起始位置，length为字符串长度，实际应用中以len(expression)取得其长度

3,right(char\_expr,int\_expr) 返回字符串右边第int\_expr个字符，还用left于之相反

4,isnull( check\_expression , replacement\_value )如果check\_expression為空，則返回replacement\_value的值，不為空，就返回check\_expression字符操作类

5,Sp\_addtype 自定義數據類型

例如：EXEC sp\_addtype birthday, datetime, 'NULL'

6,set nocount {on|off}

使返回的结果中不包含有关受 Transact-SQL 语句影响的行数的信息。如果存储过程中包含的一些语句并不返回许多实际的数据，则该设置由于大量减少了网络流量，因此可显著提高性能。SET NOCOUNT 设置是在执行或运行时设置，而不是在分析时设置。

SET NOCOUNT 为 ON 时，不返回计数（表示受 Transact-SQL 语句影响的行数）。

SET NOCOUNT 为 OFF 时，返回计数

常识

在SQL查询中：from后最多可以跟多少张表或视图：256

在SQL语句中出现 Order by,查询时，先排序，后取

在SQL中，一个字段的最大容量是8000，而对于nvarchar(4000),由于nvarchar是Unicode码。

SQLServer2000同步复制技术实现步骤

一、 预备工作

1.发布服务器,订阅服务器都创建一个同名的windows用户,并设置相同的密码,做为发布快照文件夹的有效访问用户

--管理工具

--计算机管理

--用户和组

--右键用户

--新建用户

--建立一个隶属于administrator组的登陆windows的用户（SynUser）

2.在发布服务器上,新建一个共享目录,做为发布的快照文件的存放目录,操作:

我的电脑--D:\ 新建一个目录,名为: PUB

--右键这个新建的目录

--属性--共享

--选择"共享该文件夹"

--通过"权限"按纽来设置具体的用户权限,保证第一步中创建的用户(SynUser) 具有对该文件夹的所有权限

--确定

3.设置SQL代理(SQLSERVERAGENT)服务的启动用户(发布/订阅服务器均做此设置)

开始--程序--管理工具--服务

--右键SQLSERVERAGENT

--属性--登陆--选择"此账户"

--输入或者选择第一步中创建的windows登录用户名（SynUser）

--"密码"中输入该用户的密码

4.设置SQL Server身份验证模式,解决连接时的权限问题(发布/订阅服务器均做此设置)

企业管理器

--右键SQL实例--属性

--安全性--身份验证

--选择"SQL Server 和 Windows"

--确定

5.在发布服务器和订阅服务器上互相注册

企业管理器

--右键SQL Server组

--新建SQL Server注册...

--下一步--可用的服务器中,输入你要注册的远程服务器名 --添加

--下一步--连接使用,选择第二个"SQL Server身份验证"

--下一步--输入用户名和密码（SynUser）

--下一步--选择SQL Server组,也可以创建一个新组

--下一步--完成

6.对于只能用IP,不能用计算机名的,为其注册服务器别名（此步在实施中没用到）

 (在连接端配置,比如,在订阅服务器上配置的话,服务器名称中输入的是发布服务器的IP)

开始--程序--Microsoft SQL Server--客户端网络实用工具

--别名--添加

--网络库选择"tcp/ip"--服务器别名输入SQL服务器名

--连接参数--服务器名称中输入SQL服务器ip地址

--如果你修改了SQL的端口,取消选择"动态决定端口",并输入对应的端口号

二、 正式配置

1、配置发布服务器

打开企业管理器，在发布服务器（B、C、D）上执行以下步骤:

(1) 从[工具]下拉菜单的[复制]子菜单中选择[配置发布、订阅服务器和分发]出现配置发布和分发向导

(2) [下一步] 选择分发服务器 可以选择把发布服务器自己作为分发服务器或者其他sql的服务器（选择自己）

(3) [下一步] 设置快照文件夹

采用默认\\servername\Pub

(4) [下一步] 自定义配置

可以选择:是,让我设置分发数据库属性启用发布服务器或设置发布设置

否,使用下列默认设置（推荐）

(5) [下一步] 设置分发数据库名称和位置 采用默认值

(6) [下一步] 启用发布服务器 选择作为发布的服务器

(7) [下一步] 选择需要发布的数据库和发布类型

(8) [下一步] 选择注册订阅服务器

(9) [下一步] 完成配置

2、创建出版物

发布服务器B、C、D上

(1)从[工具]菜单的[复制]子菜单中选择[创建和管理发布]命令

(2)选择要创建出版物的数据库，然后单击[创建发布]

(3)在[创建发布向导]的提示对话框中单击[下一步]系统就会弹出一个对话框。对话框上的内容是复制的三个类型。我们现在选第一个也就是默认的快照发布(其他两个大家可以去看看帮助)

(4)单击[下一步]系统要求指定可以订阅该发布的数据库服务器类型,

SQLSERVER允许在不同的数据库如 orACLE或ACCESS之间进行数据复制。

但是在这里我们选择运行"SQL SERVER 2000"的数据库服务器

(5)单击[下一步]系统就弹出一个定义文章的对话框也就是选择要出版的表

注意: 如果前面选择了事务发布 则再这一步中只能选择带有主键的表

(6)选择发布名称和描述

(7)自定义发布属性 向导提供的选择:

是 我将自定义数据筛选,启用匿名订阅和或其他自定义属性

否 根据指定方式创建发布 （建议采用自定义的方式）

(8)[下一步] 选择筛选发布的方式

(9)[下一步] 可以选择是否允许匿名订阅

1)如果选择署名订阅,则需要在发布服务器上添加订阅服务器

方法: [工具]->[复制]->[配置发布、订阅服务器和分发的属性]->[订阅服务器] 中添加

否则在订阅服务器上请求订阅时会出现的提示:改发布不允许匿名订阅

如果仍然需要匿名订阅则用以下解决办法

[企业管理器]->[复制]->[发布内容]->[属性]->[订阅选项] 选择允许匿名请求订阅

2)如果选择匿名订阅,则配置订阅服务器时不会出现以上提示

(10)[下一步] 设置快照 代理程序调度

(11)[下一步] 完成配置

当完成出版物的创建后创建出版物的数据库也就变成了一个共享数据库

有数据

srv1.库名..author有字段:id,name,phone,

srv2.库名..author有字段:id,name,telphone,adress

要求：

srv1.库名..author增加记录则srv1.库名..author记录增加

srv1.库名..author的phone字段更新，则srv1.库名..author对应字段telphone更新

--\*/

--大致的处理步骤

--1.在 srv1 上创建连接服务器,以便在 srv1 中操作 srv2,实现同步

exec sp\_addlinkedserver 'srv2','','SQLOLEDB','srv2的sql实例名或ip'

exec sp\_addlinkedsrvlogin 'srv2','false',null,'用户名','密码'

go

--2.在 srv1 和 srv2 这两台电脑中,启动 msdtc(分布式事务处理服务),并且设置为自动启动

。我的电脑--控制面板--管理工具--服务--右键 Distributed Transaction Coordinator--属性--启动--并将启动类型设置为自动启动

go

--然后创建一个作业定时调用上面的同步处理存储过程就行了

企业管理器

--管理

--SQL Server代理

--右键作业

--新建作业

--"常规"项中输入作业名称

--"步骤"项

--新建

--"步骤名"中输入步骤名

--"类型"中选择"Transact-SQL 脚本(TSQL)"

--"数据库"选择执行命令的数据库

--"命令"中输入要执行的语句: execp\_process

--确定

--"调度"项

--新建调度

--"名称"中输入调度名称

--"调度类型"中选择你的作业执行安排

--如果选择"反复出现"

--点"更改"来设置你的时间安排

然后将SQL Agent服务启动,并设置为自动启动,否则你的作业不会被执行

设置方法:

我的电脑--控制面板--管理工具--服务--右键 SQLSERVERAGENT--属性--启动类型--选择"自动启动"--确定.

--3.实现同步处理的方法2,定时同步

--在srv1中创建如下的同步处理存储过程

create procp\_process

as

--更新修改过的数据

update b set name=i.name,telphone=i.telphone

from srv2.库名.dbo.authorb,authori

where b.id=i.id and

(b.name <> i.name or b.telphone<>i.telphone)

--插入新增的数据

insert srv2.库名.dbo.author(id,name,telphone)

select id,name,telphone from author i

where not exists(

select \* from srv2.库名.dbo.author where id=i.id)

--删除已经删除的数据(如果需要的话)

delete b

from srv2.库名.dbo.author b

where not exists(

select \* from author where id=b.id)

go