MySQL高级-Day01回顾

- **SQL查询总结**

```
3、select ...聚合函数 from 表名
1、where ...
2、group by ...
4、having ...
5、order by ...
6、limit ...;
```

- **聚合函数（铁三角之一）**

avg(...) sum(...) max(...) min(...)
count(字段名) # 空值NULL不会被统计

- **group by（铁三角之二）**

给查询结果进行分组
如果select之后的字段名和group by之后的字段不一致,则必须对该字段进行聚合处理(聚合函数)

- **having语句（铁三角之三）**

对查询的结果进行进一步筛选
注意
1、having语句通常和group by语句联合使用,过滤由group by语句返回的记录集
2、where只能操作表中实际存在字段,having可操作由聚合函数生成的显示列

- **distinct**

select distinct 字段1,字段2 from 表名;

- **查询时做数学运算**

select 字段1*2,字段2+5 from 表名;

update 表名 set attack=attack*2 where 条件;

- **索引(BTree)**

优点： 加快数据检索速度
缺点： 占用物理存储空间,需动态维护,占用系统资源
SQL命令运行时间监测

mysql>show variables like '%pro%';

1、开启： mysql> set profiling=1;
2、查看： mysql> show profiles;
3、关闭： mysql> set profiling=0;

- **普通(MUL)、唯一(UNI,字段值不能重复,可为NULL)**

创建

index(字段名),index(字段名)
unique(字段名),unique(字段名)
create [unique] index 索引名 on 表名(字段名);

查看

desc 表名;

show index from 表名\G;

Non_Unique:1 :index

Non_Unique:0 :unique

删除

drop index 索引名 on 表名; (只能一个一个删)

MySQL高级-Day02笔记

外键（foreign key）

- **定义**

  让当前表字段的值在另一个表的范围内选择

- **语法**

```
foreign key(参考字段名)
references 主表(被参考字段名)
on delete 级联动作
on update 级联动作
```

- **使用规则**

1、主表、从表字段数据类型要一致
2、主表被参考字段： KEY的一种，一般为主键

- **示例**

表1、缴费信息表(财务)

```
id    姓名       班级       缴费金额
1    唐伯虎    AID1903      300
2    点秋香    AID1903      300
3    祝枝山    AID1903      300
```

表2、学生信息表(班主任) -- 做外键关联

```
stu_id    姓名     缴费金额
  1      唐伯虎     300
  2      点秋香     300
```

- **删除外键**

```
alter table 表名 drop foreign key 外键名;
•外键名 : show create table 表名;
```

- **级联动作**

```
cascade
•数据级联删除、更新(参考字段)
restrict(默认)
•从表有相关联记录,不允许主表操作
set null
•主表删除、更新,从表相关联记录字段值为NULL
```

- **已有表添加外键**

```
alter table 表名 add foreign key(参考字段) references 主表(被参考字段) on delete 级联动作 on update 级联动作
```

嵌套查询(子查询)

定义

把内层的查询结果作为外层的查询条件

语法格式

```
select ... from 表名 where 条件(select ....);
```

示例

```
1、把攻击值小于平均攻击值的英雄名字和攻击值显示出来

2、找出每个国家攻击力最高的英雄的名字和攻击值(子查询)
```

多表查询

sql脚本资料：join_query.sql

```
mysql -uroot -p123456
mysql>source /home/tarena/join_query.sql
```

```
create database if not exists db1 character set utf8;
use db1;

create table if not exists province(
id int primary key auto_increment,
pid int,
pname varchar(15)
)default charset=utf8;

insert into province values
(1, 130000, '河北省'),
(2, 140000, '陕西省'),
```

```
(3, 150000, '四川省'),
(4, 160000, '广东省'),
(5, 170000, '山东省'),
(6, 180000, '湖北省'),
(7, 190000, '河南省'),
(8, 200000, '海南省'),
(9, 200001, '云南省'),
(10,200002,'山西省');

create table if not exists city(
id int primary key auto_increment,
cid int,
cname varchar(15),
cp_id int
)default charset=utf8;

insert into city values
(1, 131100, '石家庄市', 130000),
(2, 131101, '沧州市', 130000),
(3, 131102, '廊坊市', 130000),
(4, 131103, '西安市', 140000),
(5, 131104, '成都市', 150000),
(6, 131105, '重庆市', 150000),
(7, 131106, '广州市', 160000),
(8, 131107, '济南市', 170000),
(9, 131108, '武汉市', 180000),
(10,131109, '郑州市', 190000),
(11,131110, '北京市', 320000),
(12,131111, '天津市', 320000),
(13,131112, '上海市', 320000),
(14,131113, '哈尔滨', 320001),
(15,131114, '雄安新区', 320002);

create table if not exists county(
id int primary key auto_increment,
coid int,
coname varchar(15),
copid int
)default charset=utf8;

insert into county values
(1, 132100, '正定县', 131100),
(2, 132102, '浦东新区', 131112),
(3, 132103, '武昌区', 131108),
(4, 132104, '哈哈', 131115),
(5, 132105, '安新县', 131114),
(6, 132106, '容城县', 131114),
(7, 132107, '雄县', 131114),
(8, 132108, '嘎嘎', 131115);
```

- **笛卡尔积**

```
select 字段名列表 from 表名列表;
```

- **多表查询**

```
select 字段名列表 from 表名列表 where 条件;
```

- **示例**

```
1、显示省和市的详细信息
    河北省    石家庄市
    河北省    廊坊市
    湖北省    武汉市

2、显示 省 市 县 详细信息
```

连接查询

- **内连接（结果同多表查询，显示匹配到的记录）**

```
select 字段名 from  表1 inner join 表2 on 条件 inner join 表3 on 条件;
eg1 : 显示省市详细信息

eg2 : 显示 省 市 县 详细信息
```

- **左外连接**

以 左表 为主显示查询结果

```
select 字段名 from 表1 left join 表2 on 条件 left join 表3 on 条件;
eg1 : 显示 省 市 详细信息（要求省全部显示）
```

- **右外连接**

用法同左连接,以右表为主显示查询结果

```
select 字段名 from 表1 right join 表2 on 条件 right join 表3 on 条件;
```

数据导入

==掌握大体步骤==

==source 文件名.sql==

作用

把文件系统的内容导入到数据库中
语法（方式一）

load data infile "文件名"
into table 表名
fields terminated by "分隔符"
lines terminated by "\n"

示例

scoretable.csv文件导入到数据库db2的表

```
1、将scoretable.csv放到数据库搜索路径中
   mysql>show variables like 'secure_file_priv';
         /var/lib/mysql-files/
   Linux: sudo cp /home/tarena/scoreTable.csv /var/lib/mysql-files/
2、在数据库中创建对应的表
  create table scoretab(
  rank int,
  name varchar(20),
  score float(5,2),
  phone char(11),
  class char(7)
  )charset=utf8;
3、执行数据导入语句
load data infile '/var/lib/mysql-files/scoreTable.csv'
into table scoretab
fields terminated by ','
lines terminated by '\n'
4、练习
  添加id字段,要求主键自增长,显示宽度为3,位数不够用0填充
  alter table scoretab add id int(3) zerofill primary key auto_increment first;
```

语法（方式二）

source 文件名.sql

数据导出

作用

将数据库中表的记录保存到系统文件里

语法格式

select ... from 表名
into outfile "文件名"
fields terminated by "分隔符"
lines terminated by "分隔符";

练习

```
1、把sanguo表中英雄的姓名、攻击值和国家三个字段导出来,放到 sanguo.csv中

2、将mysql库下的user表中的 user、host两个字段的值导出到 user2.txt, 将其存放在数据库目录下
```

注意

```
1、导出的内容由SQL查询语句决定
2、执行导出命令时路径必须指定在对应的数据库目录下
```

表的复制

==1、表能根据实际需求复制数据==

==2、复制表时不会把KEY属性复制过来==

语法

```
create table 表名 select 查询命令;
```

练习

```
1、复制sanguo表的全部记录和字段,sanguo2

2、复制sanguo表的 id,name,country 三个字段的前3条记录,sanguo4
```

注意

复制表的时候不会把原有表的 KEY 属性复制过来

复制表结构
create table 表名 select 查询命令 where false;

锁（自动加锁和释放锁）

==全程重点，理论和锁分类及特点==

目的

解决客户端并发访问的冲突问题

锁类型分类

```
读锁(共享锁): select 加读锁之后别人不能更改表记录,但可以进行查询
写锁(互斥锁、排他锁): 加写锁之后别人不能查、不能改
```

锁粒度分类

表级锁： myisam
行级锁： innodb

今日作业

1、把 /etc/passwd 文件的内容导入到数据库的表中

```
tarena:x:1000:1000:tarena,,,:/home/tarena:/bin/bash
```

2、Day01的md文件中的外键及查询作业题