## 一.启动与停止

1.在services.msc中选择自动手动禁止

2.在mysql自带的命令行中直接输入密码

3.以管理员身份运行命令提示符：

⑴停止服务：net stop mysql80

先启动再登陆

⑵启动服务：net start mysql80

⑶登陆：

①mysql -uroot -p123456(直接显示密码)

②mysql -uroot -p（回车）此时输入密码是密文

⑷退出：quit

⑸由于配置环境变量时录入的是mysql8.0，因此启动5.7时需要加入5.7的端口号：mysql -uroot -P13306 -p（回车）输入密码

⑹登陆其他计算机的mysql：mysql -uroot -P3306 -hlocalhost（ip地址）-p

⑺查看版本信息：mysql -V或者mysql -version

或者登陆后：select version();

## 二.使用命令

清屏：system cls(在win系统命令行是cls)

查看端口号：show global variables like 'port';

1.查看数据库：show databases;

不管在哪个位置都能查看，使用数据库之后也能查看

2.创建数据库：create database 库名;

删除数据库：drop database 库名；（在文件夹中用鼠标删除不起作用，必须用命令行）

3.在创建表之前先使用：use 库名;

使用某库之后查看所有表名：show tables;

4.创建表：create table 表名(id int,name varchar(15));

5.查看某个库的所有表名：

show tables from test001;

6.导入SQL文件

先创建并使用一个数据库，然后导入

source 路径（不要加分号，不然会打不开文件）

7.选择库后再查看表中数据：select \* from 表名;

8.查看表的结构：desc 表名；（是结构，不是数据内容）

## 三.DQL简单查询(select)

1.查询一个字段：select 字段名 from 表名；

（也是需要先选择数据库）

⑴select后面跟字段名或者对象，是对象的话查询结果会显示一个字段(值全为搞对象)，原来的表有多少行就会显示多少行

2.查询两个字段：字段名之间用,隔开

3.查询所有字段：可以把每个字段名都写上，也可以用\*

⑴实际开发不建议使用\*，效率低

⑵显示的顺序按查询的顺序来

4.给查询的字段起别名：select 字段名,字段名,原字段名 as 新字段名 from 表名；

⑴只是当前显示的字段名改变了，不改变表的字段名，select不会进行修改，as可以省略

⑵起别名的时候不能直接给别名加空格，有空格的话可以用单或双引号将别名括起来

⑶在数据库中，字符串用单引号，因为双引号不标准，在oracle数据库中不能用

5.字段可以使用数学表达式：

⑴select sal\*12 from emp;

⑵select sal\*12 as yearsal from emp;（直接显示新字段名）

⑶select sal\*12 as '年薪' from emp;

## 四.DQL条件查询

1.列出其中一个字段符合条件的一些字段：

select 字段名,字段名,字段名 from 表名 where 字段名 判断条件；

⑴例：

①字段名＜，≤，>，≥，=，<>或!=

②between \_ and \_ (两个值之间并且左小右大闭区间，等同于≥ \_ and ≤ \_)

③is null，is not null（在数据库里null不是一个值，是空，什么也没有）

④and（并且，取交集），or（取并集）

and和or同时出现，and优先级高（假如and在后面，也是先执行and周围的语句，类似\*），因此需要加括号

⑤字段名 in ('值','值',…)，相当于多个or将值连接起来。not in（不在值当中的数据）

## 五.DQL模糊查询

like称为模糊查询，支持%或\_匹配

1.%：匹配任意多个字符

select 字段名,字段名…from 表名 where 字段名 like '%O';

①名字中含有O：%O%

②以O结尾：%O

③以O开始：O%

2.\_：匹配任意一个字符

字段名,字段名…from 表名 where 字段名 like '\_O%';

①第二个字母是O:\_O%

②第三个：\_\_O%

3.查找\_：用转义字符\\_

## 六.DQL查询：排序

1.排序单个字段

⑴select 字段名,字段名,字段名 from 表名 order by 字段名;默认是升序

⑵指定降序：order by 字段名 desc;

⑶指定升序：order by 字段名 asc;（asc可以不写）

2.排序多个字段

⑴假如一个字段里有相同大小的值，再按照其他字段里名字升序排序：select 字段名,字段名,字段名 from 表名 order by 字段名1 asc, 字段名2 asc;

3.根据字段位置排序：select 字段名,字段名,字段名 from 表名 order by 数字；（这个数字是按第几个字段排序）不建议这样查询

4.综合排序（加上条件查询）排序在最后执行：

select 字段名,字段名 from 表名 where 字段名 判断条件 order by 字段名;

## 七.DQL数据处理函数

1.又被称为单行处理函数，特点：一个输入对应一个输出。有多少个输入对应多少个输出，把每个输入输出了再执行下个输入输出

⑴lower()转换小写：select lower(ename) from emp；（起别名：select lower(ename) as ename from emp;）

⑵upper()转换成大写

⑶length()取长度

⑷trim()去空格:select ename,sal from emp where ename = trim(' king')

⑸substr截子串(字段名,起始下标,截取的长度)：select substr(ename,1,1) from emp;（起始下标从1开始，从0开始不会出结果）

①模糊查询：select ename from emp where ename like 'A%';

select ename from emp where substr(ename, 1, 1) = 'A';

②小写select concat(substr(ename,1,1),lower(substr(ename,2,length(ename)-1))) as ename from emp;concat()拼接字符串

⑹str\_to\_data将字符串转化成日期

⑺data\_format格式化日期

⑻round()四舍五入：select round(1234.567,1) from emp;（1是保留1位小数，0保留整数，-1保留到十位）

⑼rand()生成随机数：默认0.多的小数

select rand() from emp;

100以内的随机数：select round(rand()\*100,0)

⑽format设置千分位

(11)ifnull：

①select sal + comm from emp;最后结果，null参与的数学运算，结果全为null

②ifnull(数据，被当做的值)：意思是如果数据为null，就把它当做别的值，比如0

select (sal + ifnull(comm,0))\*12 as yearsa from emp;

(12)case 字段名 when 条件1 then 选项1 when 条件2 then 选项2 else 默认值 end

2.相对的还有多行处理函数，又称为分组函数，必须分组了才能使用分组函数

特点：多个输入对应一个输出

默认整张表为一组

⑴sum()求和：select sum(sal) from emp;

⑵min()/max()最小/大值

⑶avg()平均值

⑷count()计数：

①count(具体字段)：统计该字段所有不为null的元素的总数

②count(\*)：统计行数（SQL语句中不存在某一行的值全为null）

⑸①分组函数会自动忽略null，不需要提前对null进行处理

②分组函数不能直接使用在where子句中

③所有的分组函数可以组合在一起用：select max(sal),min(sal),count(comm),sum(sal),avg(sal) from emp;

## 八.DQL分组查询

1.关键字展示顺序：

select，from，join，on，where，group by，having，order by，limit

关键字执行顺序：

from，on，join，where，group by，分组函数(可以使用select后面的别名)，having，select，order by，limit

⑴分组函数不能使用where，因为where执行了才会执行group by分组。

而select后面可以跟分组函数，因为select在group by后面执行，而没写group by的时候默认整张表是一组

⑵①按照岗位分组，然后对工资求和(对每个不同岗位的薪资分别求和)：

select sum(sal) from emp group by job;

②select后面不能跟其他字段名，只能跟group by 里参与分组的字段以及分组函数，因为不能按照分组对其进行归纳，可以在mysql执行，但无意义，并且不能在oracle执行

③找出每个部门的最高薪资：

select deptno,max(sal) from emp group by deptno;

④找出每个部门里不同岗位的最高薪资：

select deptno,jobmax(sal) from emp group by deptno,job;

⑤找出每个部门的最高薪资，要求显示大于等于3000的：

Ⅰ.select deptno,max(sal) from emp where sal ≥ 3000 group by deptno;

Ⅱ.（可以使用having对分完组的数据进行处理，having不能单独使用，不能代替where，必须和group by联合使用）select deptno,max(sal) from emp group by deptno havong max ≥ 3000;效率低，优先选择where

⑥用不了where只能用having的

找出每个部门平均薪资高于2500的：

select deptno,avg(sal) from emp group by deptno having avg(sal) ≥ 2500;

⑦找出每个岗位的平均薪资，要求显示大于1500的，除manager岗位之外，要求按照降序排列：

select job,avg(sal) from emp where job != 'manager' group by job having avg(sal) >= 1500 order by desc;

## 九.DQL--distinct去重

distinct给查询结果去重

1.连接一个字段名去重：select distinct deptno from emp;

2.连接多个字段，distinct放在所有字段名前面，字段间每行取并集，然后去重：select distinct job,deptno from emp;

3.统计工作岗位数量：select count(distinct job) from emp;

## 十.DQL连接查询(内,外,多张)

1.从一张表中单独查询，称为单表查询。跨表查询，多张表联合起来查询数据，被称为连接查询。

2.连接查询分类：

⑴根据年代：SQL92，SQL99

⑵根据表连接方式：内连接，外连接，全连接。

内连接A∩B：等值连接，非等值连接，自连接。

外连接：左外连接A∩B∪A，右外连接A∩B∪B。

全连接A∪B：full (outer) join

3.当两张表没有条件限制时进行连接查询，最终结果的行数是两张表行数的乘积，称为笛卡尔积现象。

4.避免笛卡尔积现象

⑴正确语句：select ename,dname from emp,dept where emp.deptno = dept.deptno;

最终查询结果是14条，匹配次数没有减少，只不过进行了筛选。

⑵效率更高：select emp.ename,dept.dname from emp,dept where emp.deptno = dept.deptno;

⑶或者给表起别名（92语法，等值连接）：select e.ename,d.dname from emp e,dept d where emp.deptno = dept.deptno;

5.内连接之等值连接（99语法）：select e.ename,d.dname from emp e join dept d on e.deptno = d.deptno;

⑴优点：表连接的条件是独立的，没用where，之后可以再添加where

⑵具体语法：select 别名1.字段名1,别名2.字段名2 from 表名1 别名1 inner join 表名2 别名2 on 连接条件 where 筛选条件;

⑶inner（内连接）可以省略

6.找出每个员工的薪资等级，要求显示员工名，薪资，薪资等级（非等值连接）：

select e.ename,e.sal,s.grade from emp e join salgrade s on e.sal between s.losal and s.hisal;

7.查询员工的上级领导，要求显示员工名和对应的领导名（自连接）：

select a.ename,b.ename from emp a join emp b on a.mgr = b.empno;

8.内连接的特点：完全能够匹配上条件的查询

select e.ename,d.dname from emp e right join dept d on e.deptno = d.deptno;

表之间没有主次关系

9.例，外连接（右）：select e.ename,d.dname from emp e right (outer) join dept d on e.deptno = d.deptno;

⑴在等值连接之后，返回所有匹配的结果，即使左表中没有能与右表连接的值，依然返回右表的所有行

⑵任何一个右（左）连接都有左（右）连接的写法。表之间有主次，右连接的主表是右表

⑶outer可以省略

⑷左右连接写法可以互换，left左边的表名或right右边的表名是主表

10.查询每个员工的上级领导，要求显示所有员工的名字和领导名（左连接）：select a.ename,b.ename from emp a left join emp b on a.mgr = b.empno;

11.多张表连接：一条语句中可以出现内外连接混合。

⑴语法：

select…

from

a

join

b

on

ab连接条件

join

c

on

ac连接条件

join

d

on

ad连接条件;

⑵例，

①找出每个员工的部门名称以及薪资等级:

select e.ename,e.sal,d.dname,s.grade from emp e join dept d on e.deptno = d.deptno join salgrade s on e.sal between s.losal and s.hisal;

②select \* from emp e join dept d on e.deptno = d.deptno join salgrade s on e.sal between s.losal and s.hisal;

③显示员工名，上级领导，员工部门，薪资，薪资等级：

select e.ename,a.ename,d.dname,e.sal,s.grade from emp e join dept d on e.deptno = d.deptno join salgrade s on e.sal between s.losal and hisal left join emp a on e.mgr = a.empno;（不加left的话，由于King没有上级领导，主表显示不全）

④员工的上级领导的上级领导：select a.ename,a.mgr,b.ename,b.mgr,c.ename,c.mgr,d.ename,d.mgr from emp a left join emp b on a.mgr = b.empno left join emp c on b.mgr = c.empno left join emp d on c.mgr = d.empno;

## 十一.子查询

1.select语句中嵌套select语句，被嵌套的select语句为子查询

2.子查询可以出现在select,from,where后面

3.(where)找出大于最低工资的员工和薪资:

select ename,sal from emp where sal > (select min(sal) from emp);

4.from后面的子查询是将子句的查询结果当做一张表。可以给子查询的表起别名，并且里面含有函数的字段名也要起别名，以防识别错误。

找出每个岗位平均工资的薪资等级：

select t.\*,s.grade from (select job,avg(sal) as avgsal from emp group by job) t join salgrade s on t.avgsal between s.losal and s.hisal;

5.(select)找出每个员工的部门名称，要求显示员工名，部门名:select e.ename,e.deptno,(select d.dname from dept d where e.deptno = d.deptno) as dname from emp e;也可以用等值连接。这个子查询的子句只能返回一个结果，不然会报错。

## 十二.DQL—union合并查询结果集

1.union效率比表连接高。

2.union是将两张表直接拼接在一起，减少了表连接过程的匹配次数。

3.多表连接时，匹配次数是相乘，而用union拼接时，匹配次数是相加。

4.例：

select ename,job from emp where job in ('manager','salesman');

等同于select ename,job from emp where job = 'manager' union select ename,job from emp where job = 'salesman';

5.注意两个集合的结果字段数应该相同，即上面是之前是两个，union后面也应该是两个，字段名可以不同。但在oracle里要求前后结果合并时数据类型也相同。

## 十三.DQL—limit(分页)

1.limit 起始下标,长度。起始下标可以省略，这样是直接从0开始取

2.MySQL中limit在order by后面执行

3.取出工资排名在3到9的员工：

select ename,sal from emp order by sal limit 2,7;

4.分页：

第pageNo页：limit (PageNo - 1) \* pageSize,pageSize

## 十四.DDL建表

1.create table 表名(

字段名1 数据类型1,

字段名2 数据类型2,

字段名3 数据类型3

);

表名：建议以t\_或者tbl\_开始，可读性强

表名和字段名都属于标识符

2.数据类型

⑴varchar：可变长度的字符串，最长255

⑵char：定长字符串，最长255

⑶int：整型，最长11

⑷bignt：长整型，相当于long

⑸float：单精度浮点数

⑹double：双精度浮点数

⑺date：短日期

⑻datetime：长日期

⑼clob：字符大对象，最多4G，例如存储一篇文章

⑽blob：二进制大对象，存储图片，视频，声音，需要使用IO流。

(11)decimal()：准确的小数值，m是数字总个数(65)，不算负号，d是小数点后个数(30)

3.删除表：drop table 表名;

优化使用：drop table if exists 表名;(防止报错)

4.例：

create table t\_student(

num int,

name varchar(32),

sex char(1) default 'm',

age int(3),

birth date

email varchar(255));

5.快速建表(将查询结果当做一张表创建)：

create table emp2 as select \* from emp;

## 十五.DML—insert插入

1.insert into 表名(字段名1,字段名2,字段名3) values(值1,值2,值3);

⑴字段名可以不按照表本来的顺序写，但是值必须和前面字段名的顺序对应。

⑵字段名和值数量上必须对应，可以不和表的列数对应，没有给某个字段指定值时默认写上NULL。

⑶可以在建表时某个字段名，数据类型的后面加上默认值：表名(字段名 数据类型 default 默认值);

⑷插入时字段名可以省略（等于都写上了），但是值必须按照表的顺序全部写上。

2.接单行处理函数的date和datetime

create table t\_user(

id int,

name varchar(32),

birth date,

create\_time datetime

);

(长日期datetime默认格式：%Y-%m-%d %h:%i:%s)

insert into t\_user values(001,'spray','2000-06-22','2022-2-28 17:01:07');

⑴如果create\_time的格式是date或者time，插入的时候会截取，时分秒或者年月日不会被插入成功。

⑵在MySQL中获取系统当前时间(datetime)：now()函数

insert into t\_user values(002,'dream','2000-9-15',now());

3.插入多条记录：insert into t\_user(id,name,birth,create\_time) values(第一行值),(二值),(三值);

要一一对应。

4.将查询结果插入一张表中（查询结果要符合要插入的表的结构）：

insert into dept select \* from dept;

## 十六.—update修改,delete删除,truncate(DDL)删除

1.update语法：update 表名 set 字段名1=值1,字段名2=值2,字段名3=值3 where 条件;

不加条件限制会导致所有数据更新。

update t\_user set name =' qwe',create\_time = now() where id = 001

2.delete语法：delete from 表名 where id = 001;

不加条件会将表里所有的数据都给删除，变成空表。

3.delete快速删除表中数据(不是删表，这种删除方式慢,效率低)：

delete from dept\_bak;

原理：数据所占物理空间不会释放。

优点：支持回滚，后悔了可以再回复数据。

删除前必须先start transaction;删除之后才能恢复。

4.truncate(不能回滚，但效率非常高，删除所有数据)：

truncate table dept\_bak;

## 十七.DDL表结构的增删改

1.修改表结构的话，后台程序的代码也需要大量修改，非常麻烦，一般不会修改

用alter，属于DDL。

2.语法：alter table 表名 add

## 十八.DDL—constraint约束

1.在创建表的时候给表中的字段加上约束，约束是为了保证表中的数据有效。

2.包括：

非空约束not null，

唯一性约束unique，

主键约束primary key，简称PK，

外键约束foreign key，简称FK，

检查约束check（MySQL不支持，Oracle支持）

3.not null：约束的字段不能为null。例如HTML提交的表单姓名不能为空。

没有表级约束，只有列级约束。

create table t\_vip(

id int,

name varvhar(255) not null

);

insert into t\_vip(id,name) values(001,'spray'),(002,'dream');

insert inti t\_vip(id) values(003);(这个会报错：)

4.unique：约束的字段不能重复，但可以为null(null不算重复)。

drop table if exists t\_vip;

create table t\_vip(

id int,

name varchar(255) unique(列级约束)

);

insert into t\_vip(id,name) values(001,'spray'),(002,'dream');

insert into t\_vip values(003,'spray');(不能重复，会报错：Duplicate entry 'spray' for key 't\_vip.name'）。

5.联合唯一性：

需求：name和Email两个字段联合起来具有唯一性，即两个字段的一行值取并集之后和别的行的值比较唯一。

称为表级约束。需要使用unique(字段名,字段名,字段名)。

drop table if exists t\_vip;

create table t\_vip(

id int,

name varchar(255),

email varchar(255),

unique(name,email)

);

insert into t\_vip(id,name)

values(001,'zhangsan','1@qq.com'),

(002,'zhangsan','2@qq.com');

insert into t\_vip values(003,'zhangsan','2@qq.com');(报错：Duplicate entry 'zhangsan-1@qq.com' for key 't\_vip.name')

6.not null和unique可以联合使用，逻辑上取并集。

在MySQL中该字段就成为主键字段，Oracle中不一样。

7.primary key(主键约束，也叫主键)：

⑴主键字段：添加了主键约束的字段。

主键值：主键字段中的每一个值叫做主键值。

⑵主键值是每一行记录的唯一标识，是每一行的身份证号。

任何一张表都应该有主键，否则表无效。

主键特征：not null + unique(值不能是null，不能重复)

⑶例：

drop table if exists t\_vip;

create table t\_vip(

id int primary key,(列级约束,单一主键)

name varchar(255)

);

insert into t\_vip(id,name) values(001,'spray'),(002,'dream');

insert into t\_vip values(001,'spray');(报错：Duplicate entry '1' for key 't\_vip.PRIMARY')

⑷表级约束：在字段名最后加上primary(id)

drop table if exists t\_vip;

create table t\_vip(

id int,

name varchar(255)

primary key(id)

);

insert into t\_vip(id,name) values(001,'spray'),(002,'dream');

⑸两个字段联合起来做主键primary key(id,name)，叫做复合主键，逻辑和第5条一样。在实际开发中不建议使用复合主键。

⑹主键只能有一个，不能给两个字段分别添加主键约束。

⑺主键值建议使用int,bight,char

主键值一般都是数字，定长的

⑻主键除了单一主键和复合主键外，还可以分为自然主键和业务主键。

自然主键：主键值是自然数，和业务没关系

业务主键：主键值和业务紧密关联，例如银行卡账号作为主键值。

开发中使用自然主键，只要不重复就行。

⑼用auto\_increment自增：

drop table if exists t\_vip;

create table t\_vip(

id int primary key auto\_increment,

name varchar(255)

);

insert into t\_vip(name) values('spray'),('dream');

insert into t\_vip(name) values('spray'),('dream');

insert into t\_vip(name) values('spray'),('dream');

insert into t\_vip(name) values('spray'),('dream');

insert into t\_vip(name) values('spray'),('dream');

8.foreign key(外键约束)

⑴保护子表的正确性。

下表的cno字段需要添加外键约束。

⑵被引用的是父表，引用的是子表。

创建表时先创建父表，

删除表时先删子表，

插入数据时先插父表。

⑶例：

drop table if exists t\_student;

drop table if exitst t\_class;

create table t\_class(

classno int primary key,

classname varchar(255)

);

create table t\_student(

no int primary key auto\_increment,

name varchar(255),

cno int,

foreign key(cno) references t\_class(classno)

);

insert into t\_class(classno,classname) values(100,'abc'),(101,'def');

insert into t\_student(name,cno) values('a',100),('b',100),('c',101),('d',101);

insert into t\_student values(5,'e',102);(报错：Cannot add or update a child row: a foreign key constraint fails (`test001`.`t\_student`, CONSTRAINT `t\_student\_ibfk\_1` FOREIGN KEY (`cno`) REFERENCES `t\_class` (`classno`)))

⑷被外键引用的父表的字段不一定要是主键，但必须具有unique唯一性约束。这样外键引用的时候才有价值。外键可以是null。

## 十九.存储引擎

1.概念：表存储/组织数据的方式

不同的存储引擎，表存储数据的方式不同

2.show create table t\_student;(展示创建表的时候的SQL语句)

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

| Table | Create Table |

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

| t\_student | CREATE TABLE `t\_student` (

`no` int NOT NULL AUTO\_INCREMENT,

`name` varchar(255) DEFAULT NULL,

`cno` int DEFAULT NULL,

PRIMARY KEY (`no`),

KEY `cno` (`cno`),

CONSTRAINT `t\_student\_ibfk\_1` FOREIGN KEY (`cno`) REFERENCES `t\_class` (`classno`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci |

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

3.在建表的最后右括号后边用ENGING来指定存储引擎，CHARSET来指定表的字符编码方式

MySQL默认存储引擎是InnoDB

默认字符编码是utf8mb4

4.查看当前MySQL支持哪些存储引擎：

show engines \G

5.关于MySQL常用的存储引擎(面试重点)：MyISAM，InnoDB，MEMORY

## 二十.TCL事务

1.事务就是一个完整的业务逻辑，比如A像B转账，事务就是DML语句一起成功或一起失败，不可再分。

2.只有DML语句(insert,delete,update)才和事务有关，涉及安全问题。做某件事时需要多条DML语句才能完成，所以需要事务。

3.⑴事务开启：

DML语句

DML语句

DML语句

DML语句

事务结束;

⑵事务执行过程中每一条DML都会记录到事务性活动的日志文件中，事务执行过程可以提交事务也可以回滚事务。

⑶提交事务：清空事务日志文件，将数据全部彻底持久化到数据库表中。标志着事务结束，并且是全部成功的结束。

⑷回滚事务：将之前所有的DML全部撤销，会清空事务日志文件，标志着事务结束，并且是全部失败的结束。

4.提交事务：commit

回滚:rollback

默认情况下是自动提交事务的，每执行一条DML自动提交一次，回滚只能回滚到上一次的提交点。

5.关闭MySQL的自动提交机制：执行start transaction;

然后再执行插入，删除，修改语句，这样再回滚就能恢复到上一次提交点。

（回滚了之后再插入好像不能回滚成功。）

commit;提交事务，提交了之后就只能回滚到提交点。

6.事务的特性：

A：原子性，事务是最小的工作单元，不可再分。

C：一致性，在同一个事务中，所有操作必须同时成功，或者同时失败。

I：隔离型，A事务和B事务具有一定的隔离。

相当于多线程并发访问同一张表。

D：持久性，事务最终结束的一个保障。事务提交相当于将没有保存到硬盘上的数据保存到硬盘上。

## 二十一.事务隔离级别

1.4个级别：

⑴读未提交：read uncommitted(最低隔离级别)

事务A可以读取事务B未提交的数据，脏读现象，称读到了脏数据。这种级别是理论。

并且一个事务在修改数据的时候不允许另一个事务同时修改，但可以读取

⑵读已提交：read committed

事务A只能提取到事务B提交之后的数据，解决了脏读现象，读取到的数据是真实的，但不可重复读取数据。意思是B可重复提交，但A每次在B提交之后读取，读取的是不同的数据。

Oracle数据库默认隔离级别是读已提交。

⑶可重复读：repeatable read

永远读取到的是事务开启后的数据，只要事务不结束，都是那个数据。(但是在同一个命令行窗口读取的数据会随着DML更新)

事务A开启之后，即使事务B修改了，提交了，读取的数据还是没有变。解决了不可重复读取数据的问题，但可能会出现读取到的数据是幻像，不真实。

MySQL默认事务隔离级别是可重复读。

例如银行总账执行一条select语句，可能需要执行很久，需要让这期间的存取款对数据没有影响。

⑷序列化/串行化：serializable(最高隔离级别)

效率最低，表示事务排队，不能并发。每次读取到的都是最真实的。

## 二十二.事务隔离级别演示

提交事务后数据才会更新到硬盘，这时其他地方才能读取到。没有提交的话只有自己能读到。

1.设置全局事务隔离级别为读未提交:

set global transaction isolation level read uncommitted;

2.查询全局级的当前隔离级别:

8.0之后版本:select @@transaction\_isolation;

之前版本的用老师文档上的。

3.注意，设置完隔离级别之后必须重进MySQL才会生效。

4.read uncommitted演示：

⑴首先改变全局事物隔离级别:

set global transaction isolation level read uncommitted;

然后重开两个窗口。

⑵A和B使用test001数据库，并开启事务执行start transaction;

⑶B插入一行数据，再用A来查询，发现B没提交A也读到了数据。

5.read committed：

⑴set global transaction isolation level read committed;

然后重开两个窗口。

⑵use test001; start transaction;

⑶B插入一行数据，A查询不到，Bcommit;之后A能查到。

6.repeatable read：

⑴set global transaction isolation level repeatable read;

然后重开两个窗口。

⑵A和B使用test001数据库，并开启事务执行start transaction;

⑶B插入几行数据，再用A来查询，发现B提交了A也读不到数据。

7.serializable：

⑴set global transaction isolation level repeatable read;

然后重开两个窗口。

⑵A和B使用test001数据库，并开启事务执行start transaction;

⑶B插入几行数据，再用A来查询，发现A的光标在下一行不动，查不到结果，这时候在B执行commit;A的查询结果才会出来。

## 二十三.索引原理

1.索引是在数据库表的字段上添加，是为了提高查询效率的机制。一张表的一个字段可以添加一个索引，多个字段联合起来也可以添加。

索引相当于一本书的目录，为了缩小扫描范围的机制。

索引是数据库优化的重要手段，优化时优先考虑的是索引。

2.MySQL在查询方面主要有两种方式，一种是全表扫描，二是根据索引检索。

3.主键上会自动添加索引对象，另外如果有unique约束的字段也会自动创建索引对象。

当创建了索引对象后，会自动形成一个树的结构。

4.在数据库当中，任何一张表的任何一条记录在硬盘存储上都有一个硬盘的物理存储编号。

5.在MySQL中，索引是一个单独的对象，不同的存储引擎以不同的形式存在，在MyISAM中，索引存储在一个.MYI文件中。在InnoDB中存储在一个逻辑名称叫做tablesapce的当中。在MYMERY中索引被存储在内存当中。

6.索引在MySQL中是以自平衡二叉树的形式存在。

7.通过索引对象idIndex查找，定位到要查找的id值，得出物理地址=0x某某某，然后select \* from t\_user where 物理地址=0x某某某;直接定位到那条记录。

8.什么时候添加索引：

⑴数据量庞大(到底有多庞大，需要测试)

⑵该字段经常被扫描，比如出现在where后面

⑶该字段很少进行DML操作，因为DML之后索引需要重新排序。

⑷建议不要随意添加索引，因为索引也是需要维护的。建议通过主键和有唯一性约束的字段进行查询，效率高。

9.类型：单一索引，复合索引，主键索引，唯一性索引。越唯一效率越高。

## 二十四.索引的创建删除查询

1.创建：create index emp\_ename\_index on emp(ename);

给emp表的ename字段添加索引对象，起名emp\_ename\_index

2.删除：drop index emp\_ename\_index on emp;

删除emp表上的emp\_ename\_index索引对象

3.查看一个SQL语句是否使用了索引进行检索：

explain select \* from emp where ename = 'KING';(解释这条SQL语句)

## 二十五.索引失效

但是8.0并没有失效？

1.select \* from emp where ename like '%T';

因为模糊查询中以%开头了，不能通过索引进行检索。应尽量避免模糊查询的时候用%开始。

2.使用or的时候，如果其中一边有一个字段没有索引，另一个字段上的索引也会失效：

explain select \* from emp where ename = 'KING' or 'manager';

不建议使用or。但是用union就不会失效。

3.使用复合索引的时候，没有用左侧的列查找。

复合索引，两个字段或多个字段联合起来添加一个索引：crrate emp\_job\_sal\_index on emp(job,sal);

索引没有失效：explain select \* from emp where job = 'manager';

失效：explain select \* from emp where sal = 800;

## 二十六.视图

原表删除之后，不能对view操作了？

1.view：站在不同的角度去看待同一份数据

2.创建：create view dept\_bak\_view as select \* from dept\_bak;

3.删除：drop view dept\_bak\_view;

4.只有DQL语句才能以view的形式创建，as后面必须是DQL语句。

5.可以面向视图对象进行增删改查，对视图对象进行增删改查(CRUD)会导致原表被操作！！！

⑴insert into dept\_bak\_view(deptno,dname,loc) values(60,'sales','beijing');dept\_bak表中的数据变了。

⑵create view emp\_dept\_view as select e.ename,e,sal,d.dname from emp\_bak e join dept d on e.deptno = d.deptno;

update emp\_dept\_view set sal = 1000 where dname = 'accounting';

select \* from emp\_bak;

这样也会变。

6.视图对象的作用：相当于引用，函数复用。修改时只需要修改视图对象所映射的SQL语句。

7.C：create

R：retrive

U：update

D：delete

## 二十七.DBA命令

1.新建用户，授权，回收权限

2.导出：在windows的DOS命令窗口输入，

mysqldump bjpowernode>D:\bjpowernode.sql -uroot -p340915将bjpowernode数据库导出到指定位置。

报错：⑴Data里没有bjpowernode时：

mysqldump: Got error: 1049: Unknown database 'bjpowernode' when selecting the database

⑵有那个数据库的时候：mysqldump: Got error: 1045: Access denied for user 'root'@'localhost' (using password: YES) when trying to connect

不过两种都成功了。(密码随便写居然也成功了？)

3.导入：在MySQL数据服务器上，

source D:\bjpowernode;

4.导出某数据库下指定的表：mysqldump bjpowernode emp>D:\bjpowernode.sql -uroot -p340915

## 二十八.数据库设计范式(面试重点)

数据库表的设计依据，可以避免数据的冗余，空间浪费。

1.第一范式：要求任何一张表必须有主键，每一个字段原子性不可再分。

2.第二范式：要求所有非主键字段完全依赖主键，不要产生部分依赖主键。建立在第一范式的基础之上。

3.第三范式：要求所有非主键字段直接依赖主键，不要产生传递依赖。建立在第二范式的基础之上。

4.最终目的是为了客户需求，有时候会拿冗余换速度。表和表的连接次数越多，效率越低。

并且对于开发人员来说，SQL语句编写难度会降低。

## 二十九.注意事项

1.电脑设置的5.7的端口号是13306

8.0的是3306

而navicat只能连13306，即服务器版本5.7

本机IP地址填写127.0.0.1或localhost

2..sql文件被称为SQL脚本文件，在其中编写SQL语句，执行SQL脚本文件时，该文件中的所有语句会全部执行

在MySQL中执行SQL脚本文件：source 绝对路径

## 三十.题目

1.取得每个部门最高薪资的人员名称：

select

e.ename,e.deptno, a.maxsal

from emp e

join (

select deptno,group\_concat(ename),max(sal) as maxsal from emp group by deptno

) a

on e.deptno = a.deptno and e.sal = a.maxsal;

2.找出薪资在部门的平均薪资之上的人员名称：select

a.deptno,a.avgsal,e.ename,e.sal

from emp e

join (

select deptno,avg(sal) as avgsal

from emp

group by deptno

) a

on e.deptno = a.deptno and e.sal > a.avgsal;

3.取得部门中所有人的平均的薪资等级：

⑴理解为部门中所有人的平均薪资的薪资等级：

select

c.ename,c.deptno,b.sgrade,b.aavgsal

from emp c

join (

select

a.deptno as adeptno,s.grade as sgrade,a.avgsal as aavgsal

from salgrade s

join (

select deptno,avg(sal) as avgsal

from emp

group by deptno

) a

on a.avgsal between s.losal and s.hisal

) b

on c.deptno = b.adeptno

order by c.deptno;

⑵理解为每个部门的薪资的等级的平均值（不用显示员工名字）：

select e.deptno,avg(s.grade)

from emp e

join salgrade s

on e.sal between s.losal and s.hisal

group by e.deptno;

4.不用组函数取得最大的薪水：

⑴降序,分页：

select sal

from emp

order by sal desc

limit 1;

⑵表的自连接：

select sal

from emp

where sal not in

(select distinct(e.sal)

from emp e

join emp

on e.sal < a.sal

);

5.找出平均薪资最高的部门编号：

⑴降序,分页：

select deptno,avg(sal)

from emp

group by deptno

order by avg(sal)

limit 1;

⑵先找出最高的平均薪资，然后根据部门编号分组求平均薪资，最后显示最高的平均薪资和分组后的平均薪资相等的部门编号：

select deptno

from emp

group by deptno

having

avg(sal) =

(

select max(a.avgsal)

from (

select deptno,avg(sal) as avgsal

from emp

group by deptno

) a

);

6.取得平均薪水最高的部门名称：

select dname

from dept

where

deptno = (

select deptno

from emp

group by deptno

having

avg(sal) =

(

select max(a.avgsal)

from (

select deptno,avg(sal) as avgsal

from emp

group by deptno

) a

)

);

7.求平均薪水等级最低的部门名称：

select dname

from dept

where

deptno = (

select deptno

from dept

group by deptno

order by avg(sal)

limit 1

);

8.取得比普通员工的最高薪水还要高的领导人姓名。(普通员工是员工代码没有在mgr上出现的）：

先给mgr去重，然后找出empno不在mgr里的，然后换成max(sal)，最后再找出薪水比它高的：

select ename,sal

from emp

where

sal > (

select max(sal)

from emp

where

empno not in (

select distinct mgr

from emp

where mgr is not null

)

);

9.取得薪水最高的前5名员工：

select ename,sal from emp order by sal desc limit 5;

10取得薪水最高的第6到第10名员工：

select ename,sal from emp order by sal desc limit 5, 5;

11.取得最后入职的5名员工：

select ename,hiredate from emp order by hiredate deac limit 5;

12.取得每个薪水等级有多少个员工：

先找出每个员工的工作岗位，然后分组，求数量：

select s.grade,count(\*)

from emp e

join salgrade s

on e.sal between s.losal and hisal

group by s.grade;

13.面试题：有 3 个表 S(学生表)，C（课程表），SC（学生选课表）

S（SNO，SNAME）代表（学号，姓名）

C（CNO，CNAME，CTEACHER）代表（课号，课名，教师）

SC（SNO，CNO，SCGRADE）代表（学号，课号，成绩）

问题：

⑴找出没选过“黎明”老师的所有学生姓名。

⑵列出 2 门以上（含2 门）不及格学生姓名及平均成绩。

⑶即学过 1 号课程又学过 2 号课所有学生的姓名。

⑴

⑵

⑶

14.列出所有员工及领导的姓名：

select e.ename,b.ename from emp e left join emp b on e.mgr = b.empno;

15.列出受雇日期早于其直接上级的所有员工编号，姓名，部门名称：

select a.empno,a.ename,b.ename,d.deptno,d.name

from emp a

left join emp b

on a.mgr = b.empno

left join dept d

on a.deptno = d.deptno

where a.hiredate < b.hiredate;

16.