多表查询

使用表的别名进行多表查询

如:SELECT a.id,a.name,a.address,b.math,b.english,b.chinese FROM tb\_demo065 a,tb\_demo065\_tel b WHERE a.id=b.id AND b.id='$\_POST[textid]'

SQL语言中，可以通过两种方式为表指定别名

第一种是通过关键字AS指定,如

SELECT a.id,a.name,a.address,b.math,b.english,b.chinese FROM tb\_demo065 AS a,tb\_demo065\_tel AS b WHERE a.id=b.id

第二种是在表名后直接加表的别名实现

SELECT a.id,a.name,a.address,b.math,b.english,b.chinese FROM tb\_demo065 a,tb\_demo065\_tel b WHERE a.id=b.id

使用表的别名应注意以下几点

(1)别名通常是一个缩短了的表名，用于在连接中引用表中的特定列，如果连接中的多个表中有相同的名称列存在，必须用表名或表的别名限定列名

(2)如果定义了表的别名就不能再使用表名

例如：

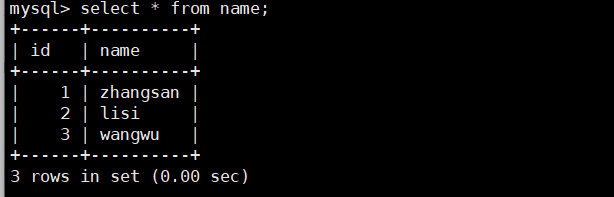
MariaDB [a]> create table name(id int,name varchar(10));

MariaDB [a]> create table age(id int,age int);

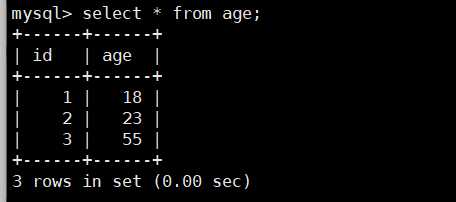
MariaDB [a]> insert into name values (1,'zhangsan') , (2,'lisi'), (3,'wangwu');

MariaDB [a]> insert into age values (1,'18'), (2,'23'), (3,'55');

MariaDB [a]> select \* from name;

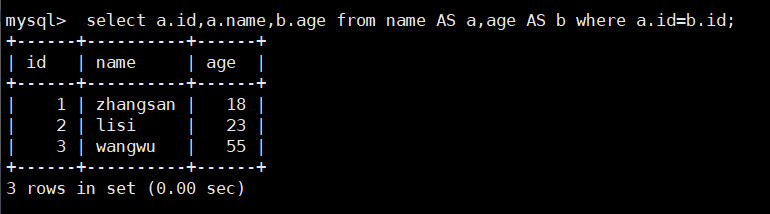


MariaDB [a]> select \* from age;



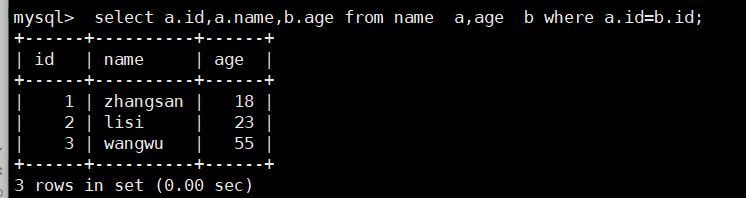
要求直接查出对应ID的每个人的年龄

MariaDB [a]> select a.id,a.name,b.age from name AS a,age AS b where a.id=b.id;



或者

mysql> select a.id,a.name,b.age from name a,age b where a.id=b.id;



MySQL事务处理简述

特点：

1、MySQL中只有innodb引擎才能使用事务

2、事务处理可以用来维护数据的完整性

3、事务用于管理insert、update、delete语句对select无效

MySQL事务的几大特性：

1、原子性，事务处理过程中，要嘛全部成功，要嘛全部失败，失败后整个事务将回滚至事务开始前的状态，不会对数据进行修改

2、数据的一致性，事务开始至事务结束，需保证数据没有被破坏，如果不符合设定的规则，事务将自动回退到事务执行前的状态

最简单的例子：两个账户间转账，两个账户的总额初始为2k，不管后面两个账户间如何转账，最后相加的总额还是2k

3、隔离性，数据库同时可以多个事务一起跑，如果不具备隔离，可能会对数据交叉，从而导致数据的损坏

4、持久性，事务执行成功后，数据将写入磁盘

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

事务隔离级别：

读未提交：read uncommitted

读已提交：read committed

可重复读：repeatable read

串行化： serializable

隔离级别介绍：

read uncommitted 读取未提交的内容

事务a未提交的数据，事务b依然能读取到，这种情况被称为“脏读”，这种读出来的数据叫“脏数据”，该隔离级别最低，一般不使用

read committed 读取提交内容

该级别下，任何事务只能读取到已经提交的数据，这种隔离级别也支持所谓的不可重复读（Nonrepeatable Read），也就是说，如果有其它事务重新提交了该数据结果，那么第二次重新查询，该数据结果将发生改变

repeatable read 可重读

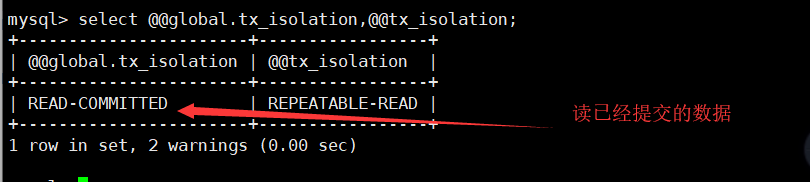
MySQL默认事务隔离级别，该级别下，事务a和事务b同时并发查询同一个数据时，会看到相同的结果，但如果此时事务a已经重新提交了数据，事务b再重新查询，将会发现查询到的数据还是之前未提交的状态，该情况被称为“幻读”，目前新版本的mysql使用innodb引擎已经很好的解决了该幻读

serializable 串行化

事务与事务之间完全隔离，不同事务间不能同一时间对相同数据进行操作，这种级别下，可能导致大量的超时现象，但隔离效果最好

修改事务隔离级别命令：

mysql> set global transaction isolation level read committed; //全局的

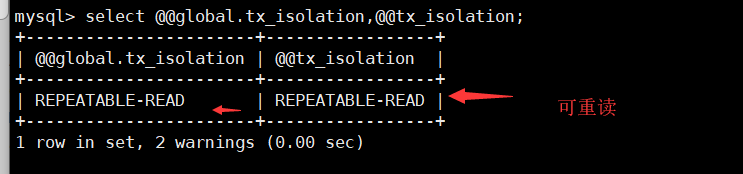


mysql> set session transaction isolation level read committed; //当前会话

mysql> set global transaction isolation level repeatable read ; //修改为重复读

查询事务隔离级别：

mysql> select @@global.tx\_isolation,@@tx\_isolation; //全局的 & 当前会话的一起修改



MySQL事务处理

mysql可使用两种办法执行事务：

1、通过begin手动执行

begin 开始事务

rollback 回滚事务

commit 确认事务

2、直接将默认自动提交关闭

set autocommit=0 关闭事务自动提交

set autocommit=1 开启事务自动提交

set autocommit=0 关闭事务自动提交

begin 开始事务

rollback 回滚事务

commit 提交事务

savepoint 标识符 创建一个事务标识符，可回滚到指定标识符位置

release savepoint 标识符 删除一个标识符

rollback to 标识符 回滚到指定标识符位置

set transaction 修改事务隔离级别

读未提交：read uncommitted

读已提交：read committed

可重复读：repeatable read

串行化：serializable

事务执行实例：

mysql > begin; //开启事务

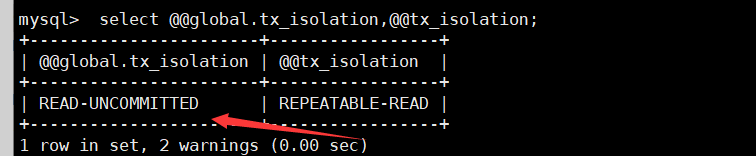
mysql > commit; //提交事务

mysql > rollback; //回滚事务

脏读实验：

mysql> set global transaction isolation level read uncommitted; 设置读取未提交的内容

mysql> select @@global.tx\_isolation,@@tx\_isolation;



事务a

mysql> begin；

在事务a还未提交时事务b：

mysql> select \*

创建事务，并在中途创建一个标识符：

mysql> begin;

mysql> savepoint a;

mysql> rollback to a;

如果要删除一个标识符：

mysql> release savepoint a;

===============================================================================

mysql视图

1、什么是视图

通过其它一个或多个表获取数据，虚拟存在的一张表，它是作为一个select语句保存的（虚拟表）

2、用来干嘛？

视图，优点多多

1.简单；使用视图表的用户不需要进行复杂的操作，查询到的数据都是给予这张表的用户提前预设好的，也就是说所看到的数据就是使用者需要的

2.安全；因为视图使用者只能看到他该看到的数据，其它数据是无法进行操作的，我们之前所学的权限控制，仅能针对表的控制而无法针对表中的某一列进行控制，而视图刚好弥补了这一点

3.数据独立；一旦视图的结构确定了，可以屏蔽表结构变化对用户的影响，源表增加列对视图没有影响；源表修改列名，则可以通过修改视图来解决，不会造成对访问者的影响

总结：使用视图是为了提高数据的安全性和查询效率

视图创建实例：

创建视图格式：

create view 视图名称 (视图字段) as select语句;

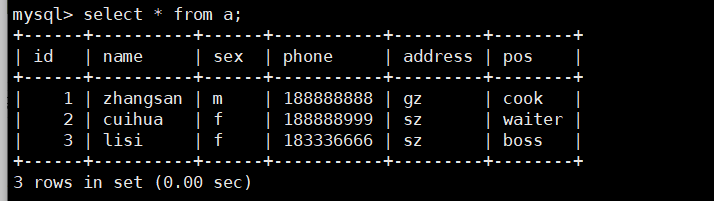
基于单表的视图创建：

创建物理表：

mysql> create table a (id int,name varchar(20),sex char(1),phone int,address varchar(20),pos varchar(20));

mysql> insert into a values(1,'zhangsan','m' ,'188888888','gz' ,'cook'),(2,'cuihua','f', '188888999','sz','waiter'),(3,'lisi','f','183336666','sz','boss');

mysql> select \* from a;

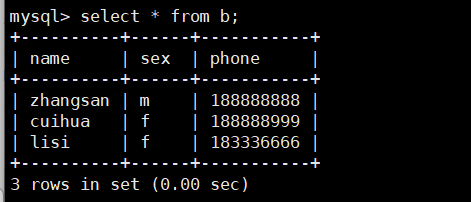


基于物理表a，创建一个视图b：

create view b(name,sex,phone) as select name,sex,phone from a;



mysql> select \* from b;



基于多表的视图创建：

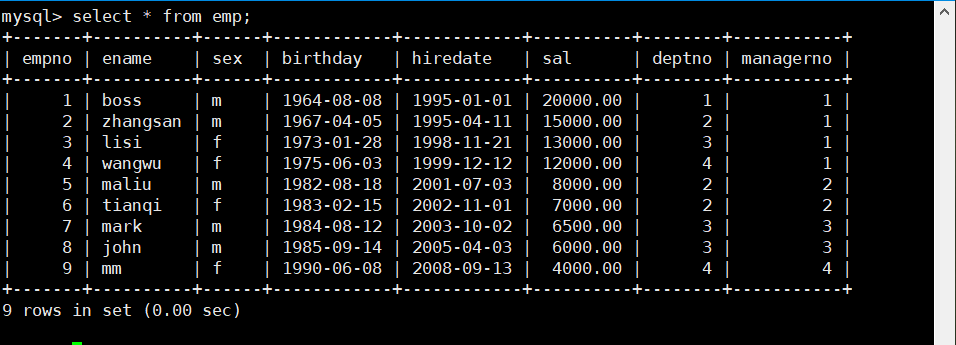
mysql> create table emp (empno int(1),ename char(20),sex char(1),birthday date,hiredate date,sal decimal(10,2),deptno tinyint(1),managerno tinyint);

mysql> create table emp1 (deptno tinyint(1),dname char(20),loc char (5));

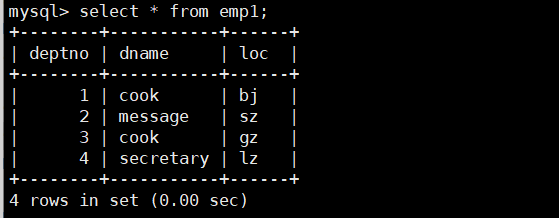
mysql> insert into emp values(1,'boss','m','1964-08-08','1995-01-01','20000','1','1'),(2,'zhangsan','m','1967-04-05','1995-04-11','15000','2','1'),(3,'lisi','f','1973-01-28','1998-11-21','13000','3','1'),(4,'wangwu','f','1975-06-03','1999-12-12','12000','4','1'),(5,'maliu','m','1982-08-18','2001-07-03','8000','2','2'),(6,'tianqi','f','1983-02-15','2002-11-01','7000','2','2'),(7,'mark','m','1984-08-12','2003-10-02','6500','3','3'),(8,'john','m','1985-09-14','2005-04-03','6000','3','3'),(9,'mm','f','1990-06-08','2008-09-13','4000','4','4');

mysql> insert into emp1 values ('1','cook','bj'),('2','message','sz'),('3','cook','gz'),('4','secretary','lz');

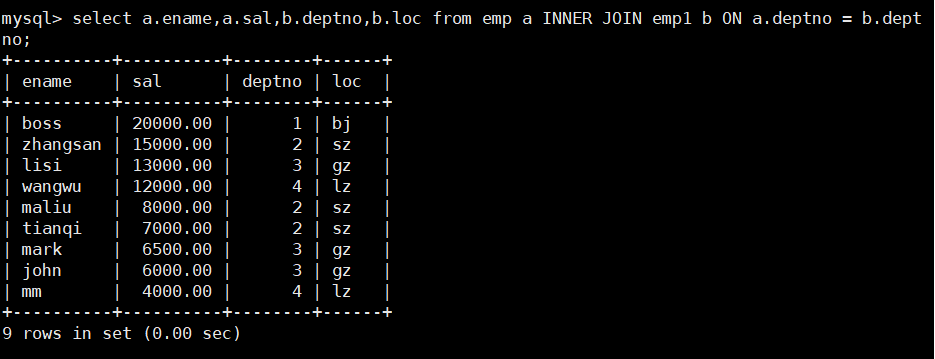
mysql> select \* from emp;



mysql> select \* from emp1;



根据上方的emp和emp1两个物理表，如何创建出一个视图？需求内容如下：



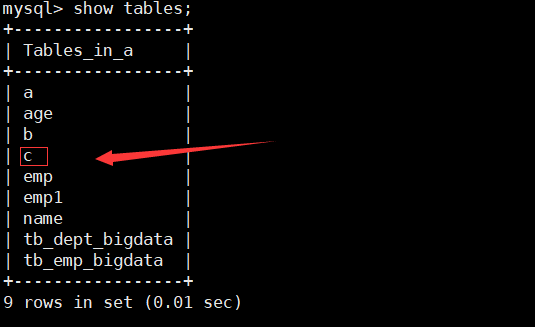
如果在上面的条件再加一个条件，找出所有女员工的信息又该如何操作？

先多表查询看下效果，

mysql> select a.ename,a.sal,b.deptno,b.loc from emp a INNER JOIN emp1 b ON a.deptno = b.deptno;

然后创建视图

mysql> create view c(ename,sal,deptno,loc) as select a.ename,a.sal,b.deptno,b.loc from emp a INNER JOIN emp1 b ON a.deptno = b.deptno;



如果要删除视图：

mysql> drop view 视图名称;