**数据库实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实 验 人：** | **凌国明** | | **学 号：** | **21307077** | **日 期：** | **2023年9月28日** |
| **院（系）：** | **计算机学院** | | | **专业（班级）：** | **计算机科学与技术** | |
| **联系方式：** |  | | |  |  | |
| **实验题目：** | | **1.5 视图** | | | | |

* + - 1. **实验目的**

熟悉 SQL 语言有关视图的操作，能够熟练使用 SQL 语句来创建需要的视图，定义数据库外模式，并能使用所创建的视图实现数据管理。

* + - 1. **实验内容和要求**

针对给定的数据库模式，以及相应的应用需求，创建视图和带 with check option的视图，并验证视图 with check option 选项的有效性。理解和掌握视图执行原理，掌握可更新视图和不可更新视图的区别。

* + - 1. **实验重点和难点**

实验重点：创建视图

实验难点：可更新视图和不可更新视图的区别，with check option 的验证。

* + - 1. **实验工具**

MySQL、SQL Server、Navicat

* + - 1. **实验过程**

（1）创建视图（省略视图列名）

创建一个‘上海黎顺服装经营部’供应商供应的零件视图 V\_DLMU\_PEATSUPP1，要求列出供应零件的编号，可用数量，零售价格，供应价格和备注等信息。

执行以下SQL语句：

CREATE VIEW V\_DLMU\_PEATSUPP1 AS

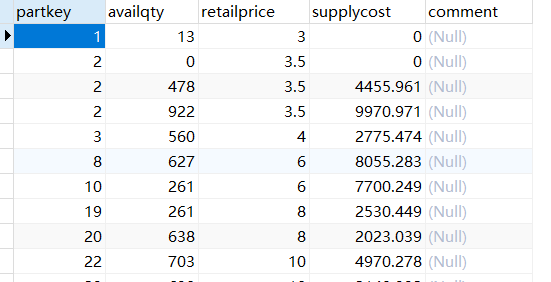
SELECT p.partkey, ps.availqty, p.retailprice, ps.supplycost, p.comment

FROM part p

JOIN partsupp ps ON p.partkey = ps.partkey

查询该视图后截屏如下：

SELECT \* FROM V\_DLMU\_PARTSUPP1;



（2）创建视图（不能省略视图列名的情况）

创建一个视图‘V\_CustAvgOrder’，按顾客统计平均每个订单的购买金额和零件数量，要求输出顾客编号，姓名，平均购买金额和平均购买零件数量。

执行以下SQL语句：

CREATE VIEW V\_CustAvgOrder AS

SELECT c.custkey, c.NAME, AVG(o.totalprice), AVG(l.quantity) AS avgquantity

FROM orders o

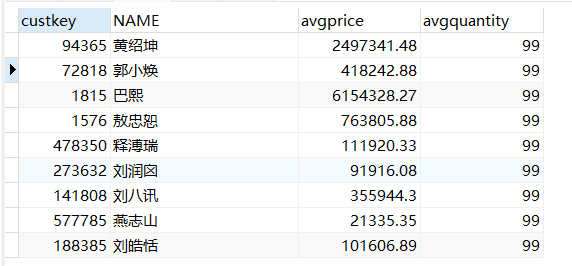
JOIN lineitem l ON o.orderkey = l.orderkey

JOIN customer c ON o.custkey = c.custkey

GROUP BY o.custkey

查询该视图中 AVGQUANTITY>98 的所有信息后部分截屏如下：

select \* from V\_CustAvgOrder where avgquantity >98;



（3）创建视图（with check option）

使用 with check option，创建一个‘上海黎顺服装经营部’供应商供应的零件视图 V\_DLMU\_PARTSUPP2，要求列出供应零件的编号、可用数量和供应价格等信息。然后通过该视图分别增加、删除和修改一条‘上海黎顺服装经营部’零件供应记录，验证 with check option 是否起作用。

1 创建视图

执行以下SQL语句：

CREATE VIEW V\_DLMU\_PEATSUPP2 AS

SELECT p.partkey, ps.availqty, ps.supplycost

FROM part p

JOIN partsupp ps ON p.partkey = ps.partkey

WITH CHECK OPTION

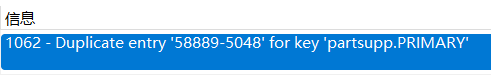
2 向该视图增加零件供应记录

* 情况1（values(58889,5048,704,77760)）：失败

执行以下SQL语句：

INSERT INTO partsupp(partkey, suppkey, availqty, supplycost)

VALUES(58889,5048,704,77760)



* 情况2（values(58889,1,704,77760)）：成功

执行以下SQL语句：

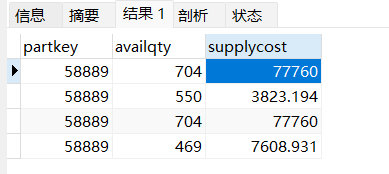
INSERT INTO partsupp(partkey, suppkey, availqty, supplycost)

VALUES(58889,1,704,77760)



select \* from V\_DLMU\_PARTSUPP2 **WHERE partkey = 58889**

查询该视图后截屏如下：



向该视图增加零件供应记录成功！

3 修改该视图的零件供应价格（set supplycost=12 where suppkey=1）

执行以下SQL语句：

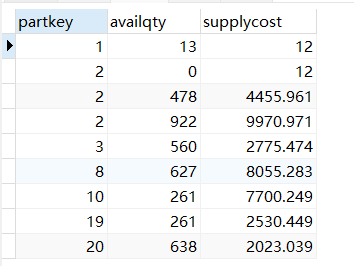
UPDATE partsupp

SET supplycost = 12

WHERE suppkey = 1

查询该视图后截屏如下：

select \* from V\_DLMU\_PARTSUPP2



修改该视图的零件供应记录成功！

4 删除该视图中零件编号大于100的供应记录（where suppkey=1 and partkey>100）

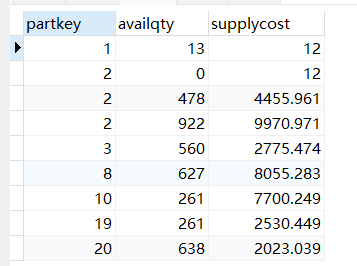
执行以下SQL语句：

DELETE FROM partsupp

WHERE suppkey = 1 AND partkey > 100

查询该视图后截屏如下：

select \* from V\_DLMU\_PARTSUPP2



删除该视图的零件供应记录成功！

说明：我们在创建视图时指定了 witch check option 关键字，这也就是说，**更新后的每一条数据仍然要满足创建视图时指定的 where 条件**。因为向视图增加记录，删除记录以及更新视图的零件供应记录都不与当初的 where 条件冲突，因此执行成功了。 即可验证 with check option 起了作用。

（4）可更新的视图（行列子集视图）

创建一个‘上海黎顺服装经营部’供应商供应的零件视图 V\_DLMU\_PEATSUPP3，要求列出供应零件的编号、可用数量和供应价格等信息。然后通过该视图分别增加、删除和修改一条‘上海黎顺服装经营部’零件供应记录，验证该视图是否是可更新的，并比较上述“（3）创建视图”实验任务与本任务结果有何异同？

1 创建视图

执行以下SQL语句：

CREATE VIEW V\_DLMU\_PEATSUPP3 AS

SELECT p.partkey, ps.availqty, ps.supplycost

FROM part p

JOIN partsupp ps ON p.partkey = ps.partkey

2 向该视图增加零件供应记录

* 情况1（values(58889,5048,704,77760)）：成功，但不能通过视图查询到

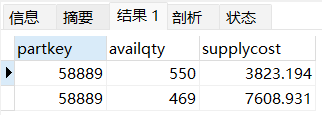
执行以下SQL语句：

INSERT INTO partsupp(partkey, suppkey, availqty, supplycost)

VALUES(58889,5048,704,77760)

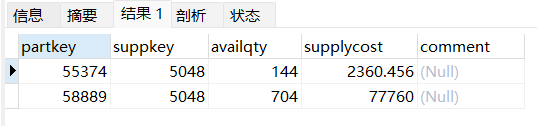
查询该视图后截屏如下：（左边插入前，右边插入后）

select \* from V\_DLMU\_PARTSUPP3 WHERE partkey = 58889

查询截屏如下：

select \* from PARTSUPP where suppkey=5048



* 情况2（values(58889,1,704,77760)）：成功，并且通过视图可以查询到

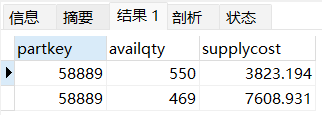
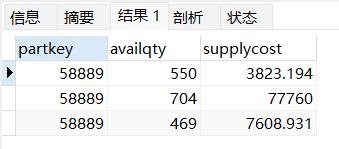
执行以下SQL语句：

INSERT INTO partsupp(partkey, suppkey, availqty, supplycost)

VALUES(58889,5048,704,77760)

查询该视图后截屏如下：（左边为插入前，右边为插入后）

select \* from V\_DLMU\_PARTSUPP3 WHERE partkey = 58889

向该视图增加零件供应记录成功！

3 修改该视图的零件供应价格（set supplycost=20 where suppkey=1）

执行以下SQL语句：

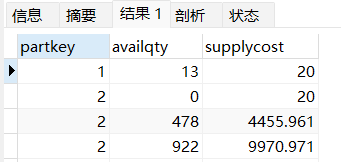
UPDATE partsupp

SET supplycost=20

WHERE suppkey=1

查询该视图后截屏如下：

select \* from V\_DLMU\_PARTSUPP3



修改该视图的零件供应记录成功！

4、删除该视图中零件编号大于100的供应记录（where suppkey=1 and partkey>100）

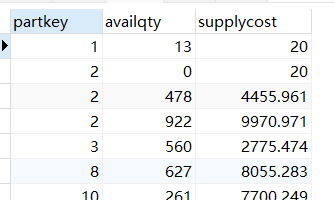
执行以下SQL语句：

DELETE FROM partsupp

WHERE suppkey=1 AND partkey>100

查询该视图后截屏如下：

select \* from V\_DLMU\_PARTSUPP3



删除该视图的零件供应记录成功！

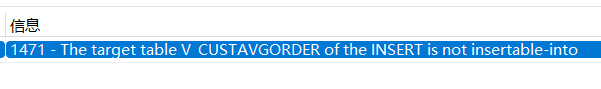
（5）不可更新的视图

（2）中创建的视图是可更新的吗？通过 SQL 更新语句加以验证，并说明原因。

答：

insert into V\_CUSTAVGORDER values(100000,NULL,20,2000);

执行该语句的反馈信息截屏如下：



SQL中视图不可插入是因为视图本身是虚拟的表，它是由一个或多个基本表的查询结果组成的，而不是实际存储数据的对象。因此，插入操作无法直接应用于视图，因为无法确定数据应该插入到哪个基本表中。要插入数据，需要直接操作基本表而不是视图。

（6）删除视图（ restrict/Cascade）

创建顾客订购零件明细视图 V\_CustOrd,要求列出顾客编号、姓名、购买零件数，金额， 然后在该视图的基础上，再创建（2）的视图 V\_CustAvOrder1,然后使用 restrict 选项删除 视图 V\_CustOrd,观察现象并解释原因。利用 cascade 选项删除视图 V\_CustOrd,观察现象并检查 V\_CustAvgOrder 是否存在，解释原因。

1、创建顾客订购零件明细视图 V\_CustOrd

执行以下SQL语句：

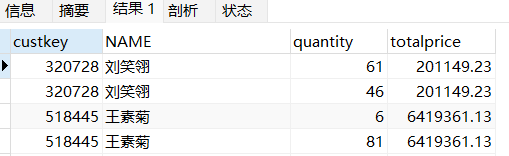
CREATE VIEW V\_CustOrd AS

SELECT c.custkey, c.NAME, quantity, totalprice

FROM orders o

JOIN lineitem l ON o.orderkey = l.orderkey

JOIN customer c ON o.custkey = c.custkey



2、基于视图 V\_CustOrd 创建视图 V\_CustAvOrder1

执行以下SQL语句：

CREATE VIEW V\_CustAvOrder1 AS

SELECT custkey, NAME, AVG(quantity), AVG(totalprice)

FROM V\_CustOrd

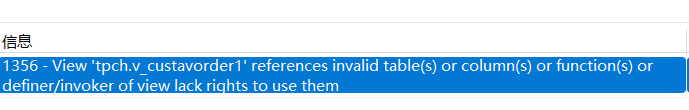
GROUP BY custkey



3、使用 restrict 选项删除视图 V\_CustOrd

执行以下SQL语句：

DROP VIEW V\_CUSTORD RESTRICT;



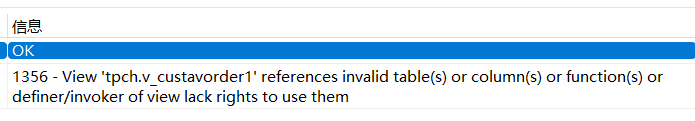
4、使用 cascade 选项删除视图 V\_CustOrd

先做“1、创建顾客订购零件明细视图 V\_CustOrd”

drop view V\_CUSTORD cascade;

select \* from V\_CUSTAVGORDER1;

执行该语句的反馈信息截屏如下：



视图是虚拟的表，它是一个或多个基本表的查询结果的引用，而不是实际存储数据的对象。V\_CustOrd引用数据库中的基本表，V\_CUSTAVGORDER1引用V\_CustOrd。如果基本表被删除，则无法访问V\_CustOrd；同理，如果V\_CustOrd被删除，无法访问V\_CUSTAVGORDER1

* + - 1. **与实验结果相关的文件**

无

* + - 1. **实验总结**

在这次实验中，我深入了解了SQL语言中与视图相关的操作。实验的主要目的是使我们熟悉SQL语句的视图创建和管理，以及理解数据库外模式的定义和使用。

首先，我们针对给定的数据库模式，使用SQL语句创建了多个视图，这有助于将数据库中的数据按照不同的需求进行组织和呈现。我们还学习了如何使用"with check option"选项来保证视图数据的完整性，这对于数据管理非常重要。

在实验过程中，我深入理解了视图的执行原理，特别是了解了可更新视图和不可更新视图之间的区别。可更新视图允许我们对其进行修改、插入和删除操作，而不可更新视图则不允许这些操作，这是因为视图的定义可能包含了复杂的查询逻辑。

实验的难点之一是验证"with check option"选项的有效性，确保视图数据的完整性。这需要仔细检查视图的定义和与之相关的基本表，确保插入或修改的数据符合视图的条件。

总的来说，通过这次实验，我不仅学会了如何创建和管理SQL视图，还深入了解了视图的特性以及如何保证数据的一致性和完整性。这对于数据库管理和数据分析都是非常有用的技能。