# 实验1-数据库定义与操作语言实验

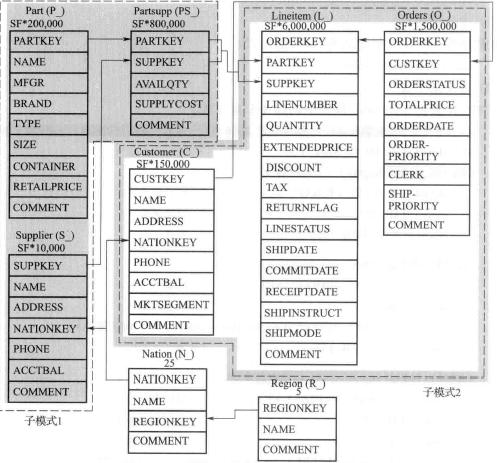
## 实验环境

Windows 10

MySQL 8.0

## 数据库定义实验

本实验建立TPC-H数据库模式。TPC-H数据库模式由零件表（Part）、供应商表（Supplier）、零件供应商表（Partsupp）、顾客表（Customer）、国家表（Nation）、地区表（Region）、订单表（Orders）和订单明细表（Lineitem）8个基本表组成。



TPC-H数据库模式…….

子模式1：。。。

子模式2：。。。

1. 定义数据库

采用中文字符集创建名为TPCH的数据库。“IF NOT EXISTS”语句判断TPCH数据库是否存在，如果没有则创建；设置数据库字符集的语句是“DEFAULT CHARACTER SET GBK COLLATE GBK\_CHINESE\_CI”，教材中“ENCODING = ‘GBK’”的语法有误。

CREATE DATABASE IF NOT EXISTS TPCH DEFAULT CHARACTER SET GBK COLLATE GBK\_CHINESE\_CI;

1. 定义模式

在数据库TPCH中创建名为Sales的模式。

CREATE SCHEMA IF NOT EXISTS Sales DEFAULT CHARACTER SET GBK COLLATE GBK\_CHINESE\_CI;

在MySQL中，模式和数据库是等价的，因此执行这条语句后“SHOW DATABASES;”会发现有一个名为“Sales”的数据库。

1. 定义基本表

在Sales模式中创建8个基本表。

1. 设置当前作用域的数据库为Sales，基本表就会自动创建在Sales模式下。

USE Sales;

1. 地区表

CREATE TABLE IF NOT EXISTS Region ( --地区表

regionkey INTEGER PRIMARY KEY, --地区编号（主键）

name CHAR(25), --地区名称

comment VARCHAR(152) --备注

);

1. 国家表

CREATE TABLE IF NOT EXISTS Nation ( --国家表

nationkey INTEGER PRIMARY KEY, --国家编号（主键）

name CHAR(25), --国家名称

regionkey INTEGER REFERENCES Region (regionkey), --地区编号（外键）

comment VARCHAR(152) --备注

);

1. 供应商表

CREATE TABLE IF NOT EXISTS Supplier ( --供应商基本表

suppkey INTEGER PRIMARY KEY, --供应商编号（主键）

name CHAR(25), --供应商名称

address VARCHAR(40), --供应商地址

nationkey INTEGER REFERENCES Nation (nationkey), --国家编号（外键）

phone CHAR(15), --供应商电话

accbal REAL, --供应商账户余额

comment VARCHAR(101) --备注

);

1. 零件表

CREATE TABLE IF NOT EXISTS Part ( --零件基本表

partkey INTEGER PRIMARY KEY, --零件编号（主键）

name VARCHAR(55), --零件名称

mfgr CHAR(25), --制造厂

brand CHAR(10), --品牌

type VARCHAR(25), --零件类型

size INTEGER, --尺寸

container CHAR(10), --包装

retailprice REAL, --零售价格

comment VARCHAR(23) --备注

);

1. 零件供应商表

CREATE TABLE IF NOT EXISTS PartSupp ( --零件供应联系表

partkey INTEGER REFERENCES Part (partkey), --零件编号（外键）

suppkey INTEGER REFERENCES Supplier (suppkey),--零件供应商编号（外键）

availqty INTEGER, --可用数量

supplycost REAL, --供应价格

comment VARCHAR(199), --备注

PRIMARY KEY (partkey , suppkey) --主键，表级约束

);

1. 顾客表

CREATE TABLE IF NOT EXISTS Customer ( --顾客表

custkey INTEGER PRIMARY KEY, --顾客编号（主键）

name VARCHAR(25), --顾客名字

address VARCHAR(40), --地址

nationkey INTEGER REFERENCES Nation (nationkey), --国籍编号

phone CHAR(15), --电话

acctbal REAL, --账户余额

mktsegment CHAR(10), --市场分区

comment VARCHAR(117) --备注

);

1. 订单表

CREATE TABLE IF NOT EXISTS Orders ( --订单表

orderkey INTEGER PRIMARY KEY, --订单编号（主键）

custkey INTEGER REFERENCES Customer (custkey), --顾客编号（外键）

orderstatus CHAR(1), --订单状态

totalprice REAL, --总金额

orderdate DATE, --日期

orderpriority CHAR(15), --优先级

clerk CHAR(15), --记账员

shippriority INTEGER, --运输优先级

comment VARCHAR(79) --备注

);

1. 订单明细表

CREATE TABLE IF NOT EXISTS Lineitem ( --订单明细表

orderkey INTEGER REFERENCES Orders (orderkey), --订单编号

partkey INTEGER REFERENCES Part (partkey), --零件编号

suppkey INTEGER REFERENCES Supplier (suppkey), --供应商编号

linenumber INTEGER, --订单明细编号

quantity REAL, --数量

extendedprice REAL, --订单明细价格

discount REAL, --折扣[0.00, 1.00]

tax REAL, --税率[0.00， 0.08]

returnflag CHAR(1), --退货标记

linestatus CHAR(1), --订单明细状态

shipdate DATE, --装运日期

commitdate DATE, --委托日期

receipdate DATE, --签收日期

shipinstruct CHAR(25), --装运说明

shipmode CHAR(10), --装运方式

comment VARCHAR(44), --备注

PRIMARY KEY (orderkey , linenumber),

FOREIGN KEY (partkey , suppkey)

REFERENCES PartSupp (partkey , suppkey)

);

## 数据基本查询实验

1. 导入数据

加载数据时遇到“—secure-file-priv”错误，使用“SHOW GLOBAL VARIABLES LIKE ‘%secure%’;”发现secure-file-priv为NULL，即不允许导入导出数据，需要更改MySQL服务默认配置文件my.ini中的secure-file-priv选项为“”（空字符串），以表示允许任意目录的导入导出，或者设置为某个目录的路径。

USE Sales;

LOAD DATA INFILE 'region.tbl' REPLACE INTO TABLE Sales.Region FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'nation.tbl' REPLACE INTO TABLE Sales.Nation FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'supplier.tbl' REPLACE INTO TABLE Sales.Supplier FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'part.tbl' REPLACE INTO TABLE Sales.Part FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'partsupp.tbl' REPLACE INTO TABLE Sales.PartSupp FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'customer.tbl' REPLACE INTO TABLE Sales.Customer FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

LOAD DATA INFILE 'orders.tbl' REPLACE INTO TABLE Sales.Orders FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

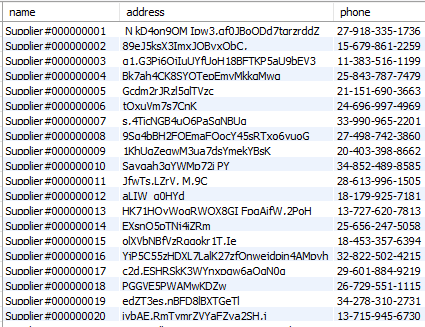
LOAD DATA INFILE 'lineitem.tbl' REPLACE INTO TABLE Sales.Lineitem FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';

1. 单表查询（投影）

查询供应商的名称、地址和联系电话。

SELECT name, address, phone FROM Supplier;

此命令查询了Supplier表中的所有元组，并打印其name、address、phone属性。部分结果如下：

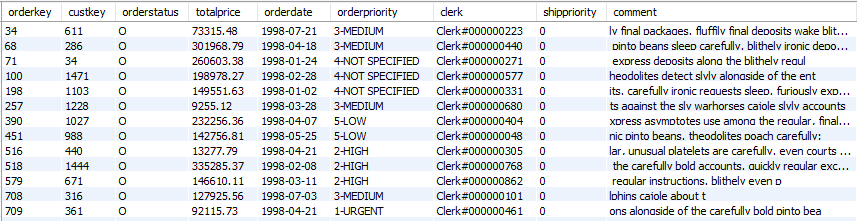


1. 单表查询（选择）

查询最近一周内提交的总价大于1000元的订单的编号、顾客编号等订单的所有信息。由于表中日期是很久之前的，因此假设CURRENT\_DATE为DATE("1997-12-25")。

SELECT \* FROM Sales.Orders WHERE DATE("1997-12-25") - orderdate >= 0 AND DATE("1997-12-25") - orderdate < 7 AND totalprice > 1000;

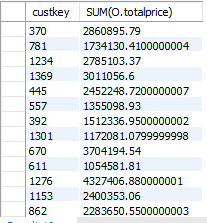
结果如下：



1. 不带分组过滤条件的分组统计查询

SELECT C.custkey, SUM(O.totalprice) FROM customer C, orders O

WHERE C.custkey = O.custkey GROUP BY C.custkey;

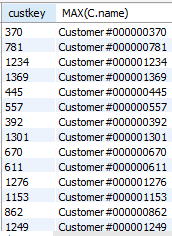


1. 带分组过滤条件的分组统计查询

SELECT C.custkey, MAX(C.name) FROM customer C, orders O

WHERE C.custkey = O.custkey

GROUP BY C.custkey HAVING AVG(O.totalprice) > 1000;



1. 单表自身连接查询

SELECT F.suppkey, F.name, F.address

FROM supplier F, supplier S WHERE F.nationkey = S.nationkey

AND S.name = 'Supplier#000000009';



1. 两表连接查询（普通连接）

SELECT P.name, P.mfgr, P.retailprice, PS.supplycost

FROM part P, partsupp PS

WHERE P.retailprice > PS.supplycost;



1. 两表连接查询（自然连接）

SELECT P.name, P.mfgr, P.retailprice, PS.supplycost

FROM part P, partsupp PS

WHERE P.partkey = PS.partkey AND P.retailprice > PS.supplycost;

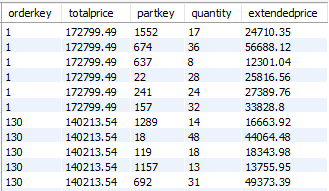


1. 三表连接查询

SELECT O.orderkey, O.totalprice, L.partkey, L.quantity, L.extendedprice

FROM customer C, orders O, lineitem L

WHERE C.custkey = O.custkey AND O.orderkey = L.orderkey AND C.name = 'Customer#000000370';



## 数据更新实验

1. INSERT基本语句（插入全部列的数据）

INSERT INTO customer VALUES(2030, '张三', '北京市', 40, '010-51001199', 0.00, 'Northeast', 'vip customer');

按教材上的代码输入会发生错误：Error Code: 1062. Duplicate entry '30' for key 'PRIMARY'。这是因为cust\_key是主键，而cust\_key为30的客户已经在customer表中。

SELECT \* FROM customer WHERE name = '张三';



1. INSERT基本语句（插入部分列的数据）

教材中(partkey, suppkey)为(479, 1)，外键引用错误，因为PartSupp中没有这个主键，修改为(1, 27)。

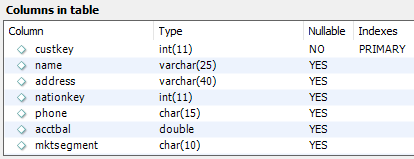
INSERT INTO lineitem(orderkey, Linenumber, partkey, suppkey, quantity, shipdate)

VALUES (862, ROUND(RAND() \* 100.0), 1, 27, 10, '2012-3-6');

1. 批量数据INSERT语句
   1. 创建一个新的顾客表，把所有中国国籍顾客插入到新的顾客表中。

“WITH NO DATA”的方法语法错误。

CREATE TABLE NewCustomer LIKE customer;

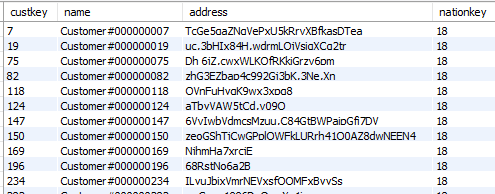


INSERT INTO newcustomer

SELECT C.\*

FROM customer C, nation N

WHERE C.nationkey = N.nationkey AND N.name = 'CHINA';



* 1. 创建一个顾客购物统计表，记录每个顾客极其购物总数和总价等信息。

CREATE TABLE ShoppingStat(

    custkey INTEGER,

quantity REAL,

totalprice REAL

);

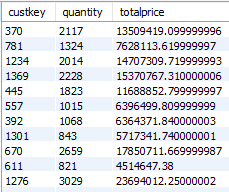
INSERT INTO shoppingstat

SELECT C.custkey, SUM(L.quantity), SUM(O.totalprice)

FROM customer C, orders O, lineitem L

WHERE C.custkey = O.custkey AND O.orderkey = L.orderkey

GROUP BY C.custkey;



* 1. 倍增零件表的数据，多次重复执行，直到总记录数达到50万行为止。

INSERT INTO part

SELECT partkey + (SELECT COUNT(\*) FROM part),

     name, mfgr, brand, type, size, container, retailprice, comment

FROM part;

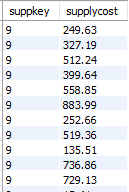
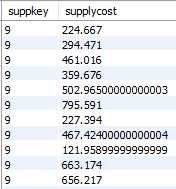


1. UPDATE语句（修改部分记录的部分列值）

UPDATE partsupp

SET supplycost = supplycost \* 0.9

WHERE suppkey = (SELECT suppkey FROM supplier WHERE name = 'Supplier#000000009');

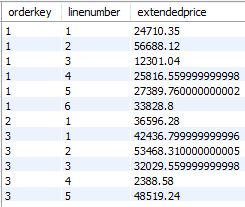
1. UPDATE语句（利用一个表中的数据修改另外一个表中的数据）

教材又错了。

UPDATE lineitem L, part P

SET L.extendedprice = P.retailprice \* L.quantity

WHERE L.partkey = P.partkey;



1. DELETE基本语句（删除给定条件的所有记录）

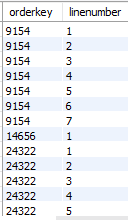
DELETE FROM lineitem

WHERE orderkey IN (SELECT orderkey FROM orders O, customer C

                WHERE O.custkey = C.custkey AND C.name = 'Customer#000000001');

DELETE FROM orders

WHERE custkey = (SELECT custkey FROM customer WHERE name = 'Customer#000000001');

## 视图实验

1. 创建视图（省略视图列名）
2. 创建视图（不能省略列名的情况）
3. 创建视图（WITH CHECK OPTION）
4. 可更新的视图
5. 不可更新的视图
6. 删除视图（RESTRICT/CASCADE）

## 索引实验

1. 创建唯一索引
2. 创建函数索引
3. 创建复合索引
4. 创建聚簇索引
5. 创建Hash索引
6. 修改索引名称
7. 分析某个SQL查询语句执行时是否使用了索引
8. 验证索引效率

## 实验总结