Drop database if exists E\_commerce;

Create database E\_commerce;

use E\_commerce;

CREATE TABLE

IF NOT EXISTS supplier (

SUPP\_ID int primary key,

SUPP\_NAME varchar(50) NOT NULL,

SUPP\_CITY varchar(50),

SUPP\_PHONE varchar(10) NOT NULL

);

CREATE TABLE

IF NOT EXISTS customer (

CUS\_ID INT NOT NULL,

CUS\_NAME VARCHAR(20) NOT NULL,

CUS\_PHONE VARCHAR(10) NOT NULL,

CUS\_CITY varchar(30) NOT NULL,

CUS\_GENDER CHAR,

PRIMARY KEY (CUS\_ID)

);

CREATE TABLE

IF NOT EXISTS category (

CAT\_ID INT NOT NULL,

CAT\_NAME VARCHAR(20) NOT NULL,

PRIMARY KEY (CAT\_ID)

);

CREATE TABLE

IF NOT EXISTS product (

PRO\_ID INT NOT NULL,

PRO\_NAME VARCHAR(20) NOT NULL DEFAULT "Dummy",

PRO\_DESC VARCHAR(60),

CAT\_ID INT NOT NULL,

PRIMARY KEY (PRO\_ID),

FOREIGN KEY (CAT\_ID) REFERENCES CATEGORY (CAT\_ID)

);

CREATE TABLE

IF NOT EXISTS supplier\_pricing (

PRICING\_ID INT NOT NULL,

PRO\_ID INT NOT NULL,

SUPP\_ID INT NOT NULL,

SUPP\_PRICE INT DEFAULT 0,

PRIMARY KEY (PRICING\_ID),

FOREIGN KEY (PRO\_ID) REFERENCES PRODUCT (PRO\_ID),

FOREIGN KEY (SUPP\_ID) REFERENCES SUPPLIER (SUPP\_ID)

);

CREATE TABLE

IF NOT EXISTS `order` (

ORD\_ID INT NOT NULL,

ORD\_AMOUNT INT NOT NULL,

ORD\_DATE DATE,

CUS\_ID INT NOT NULL,

PRICING\_ID INT NOT NULL,

PRIMARY KEY (ORD\_ID),

FOREIGN KEY (CUS\_ID) REFERENCES CUSTOMER (CUS\_ID),

FOREIGN KEY (PRICING\_ID) REFERENCES SUPPLIER\_PRICING (PRICING\_ID)

);

CREATE TABLE

IF NOT EXISTS rating (

RAT\_ID INT NOT NULL,

ORD\_ID INT NOT NULL,

RAT\_RATSTARS INT NOT NULL,

PRIMARY KEY (RAT\_ID),

FOREIGN KEY (ORD\_ID) REFERENCES `order` (ORD\_ID)

);

INSERT INTO SUPPLIER VALUES(1,"Rajesh Retails","Delhi",'1234567890');

INSERT INTO SUPPLIER VALUES(2,"Appario Ltd.","Mumbai",'2589631470');

INSERT INTO SUPPLIER VALUES(3,"Knome products","Banglore",'9785462315');

INSERT INTO SUPPLIER VALUES(4,"Bansal Retails","Kochi",'8975463285');

INSERT INTO SUPPLIER VALUES(5,"Mittal Ltd.","Lucknow",'7898456532');

INSERT INTO CUSTOMER VALUES(1,"AAKASH",'9999999999',"DELHI",'M');

INSERT INTO CUSTOMER VALUES(2,"AMAN",'9785463215',"NOIDA",'M');

INSERT INTO CUSTOMER VALUES(3,"NEHA",'9999999999',"MUMBAI",'F');

INSERT INTO CUSTOMER VALUES(4,"MEGHA",'9994562399',"KOLKATA",'F');

INSERT INTO CUSTOMER VALUES(5,"PULKIT",'7895999999',"LUCKNOW",'M');

INSERT INTO CATEGORY VALUES( 1,"BOOKS");

INSERT INTO CATEGORY VALUES(2,"GAMES");

INSERT INTO CATEGORY VALUES(3,"GROCERIES");

INSERT INTO CATEGORY VALUES (4,"ELECTRONICS");

INSERT INTO CATEGORY VALUES(5,"CLOTHES");

INSERT INTO PRODUCT VALUES(1,"GTA V","Windows 7 and above with i5 processor and 8GB RAM",2);

INSERT INTO PRODUCT VALUES(2,"TSHIRT","SIZE-L with Black, Blue and White variations",5);

INSERT INTO PRODUCT VALUES(3,"ROG LAPTOP","Windows 10 with 15inch screen, i7 processor, 1TB SSD",4);

INSERT INTO PRODUCT VALUES(4,"OATS","Highly Nutritious from Nestle",3);

INSERT INTO PRODUCT VALUES(5,"HARRY POTTER","Best Collection of all time by J.K Rowling",1);

INSERT INTO PRODUCT VALUES(6,"MILK","1L Toned MIlk",3);

INSERT INTO PRODUCT VALUES(7,"Boat EarPhones","1.5Meter long Dolby Atmos",4);

INSERT INTO PRODUCT VALUES(8,"Jeans","Stretchable Denim Jeans with various sizes and color",5);

INSERT INTO PRODUCT VALUES(9,"Project IGI","compatible with windows 7 and above",2);

INSERT INTO PRODUCT VALUES(10,"Hoodie","Black GUCCI for 13 yrs and above",5);

INSERT INTO PRODUCT VALUES(11,"Rich Dad Poor Dad","Written by RObert Kiyosaki",1);

INSERT INTO PRODUCT VALUES(12,"Train Your Brain","By Shireen Stephen",1);

INSERT INTO SUPPLIER\_PRICING VALUES(1,1,2,1500);

INSERT INTO SUPPLIER\_PRICING VALUES(2,3,5,30000);

INSERT INTO SUPPLIER\_PRICING VALUES(3,5,1,3000);

INSERT INTO SUPPLIER\_PRICING VALUES(4,2,3,2500);

INSERT INTO SUPPLIER\_PRICING VALUES(5,4,1,1000);

INSERT INTO SUPPLIER\_PRICING VALUES(6,12,2,780);

INSERT INTO SUPPLIER\_PRICING VALUES(7,12,4,789);

INSERT INTO SUPPLIER\_PRICING VALUES(8,3,1,31000);

INSERT INTO SUPPLIER\_PRICING VALUES(9,1,5,1450);

INSERT INTO SUPPLIER\_PRICING VALUES(10,4,2,999);

INSERT INTO SUPPLIER\_PRICING VALUES(11,7,3,549);

INSERT INTO SUPPLIER\_PRICING VALUES(12,7,4,529);

INSERT INTO SUPPLIER\_PRICING VALUES(13,6,2,105);

INSERT INTO SUPPLIER\_PRICING VALUES(14,6,1,99);

INSERT INTO SUPPLIER\_PRICING VALUES(15,2,5,2999);

INSERT INTO SUPPLIER\_PRICING VALUES(16,5,2,2999);

INSERT INTO `ORDER` VALUES (101,1500,"2021-10-06",2,1);

INSERT INTO `ORDER` VALUES(102,1000,"2021-10-12",3,5);

INSERT INTO `ORDER` VALUES(103,30000,"2021-09-16",5,2);

INSERT INTO `ORDER` VALUES(104,1500,"2021-10-05",1,1);

INSERT INTO `ORDER` VALUES(105,3000,"2021-08-16",4,3);

INSERT INTO `ORDER` VALUES(106,1450,"2021-08-18",1,9);

INSERT INTO `ORDER` VALUES(107,789,"2021-09-01",3,7);

INSERT INTO `ORDER` VALUES(108,780,"2021-09-07",5,6);

INSERT INTO `ORDER` VALUES(109,3000,"2021-09-10",5,3);

INSERT INTO `ORDER` VALUES(110,2500,"2021-09-10",2,4);

INSERT INTO `ORDER` VALUES(111,1000,"2021-09-15",4,5);

INSERT INTO `ORDER` VALUES(112,789,"2021-09-16",4,7);

INSERT INTO `ORDER` VALUES(113,31000,"2021-09-16",1,8);

INSERT INTO `ORDER` VALUES(114,1000,"2021-09-16",3,5);

INSERT INTO `ORDER` VALUES(115,3000,"2021-09-16",5,3);

INSERT INTO `ORDER` VALUES(116,99,"2021-09-17",2,14);

INSERT INTO RATING VALUES(1,101,4);

INSERT INTO RATING VALUES(2,102,3);

INSERT INTO RATING VALUES(3,103,1);

INSERT INTO RATING VALUES(4,104,2);

INSERT INTO RATING VALUES(5,105,4);

INSERT INTO RATING VALUES(6,106,3);

INSERT INTO RATING VALUES(7,107,4);

INSERT INTO RATING VALUES(8,108,4);

INSERT INTO RATING VALUES(9,109,3);

INSERT INTO RATING VALUES(10,110,5);

INSERT INTO RATING VALUES(11,111,3);

INSERT INTO RATING VALUES(12,112,4);

INSERT INTO RATING VALUES(13,113,2);

INSERT INTO RATING VALUES(14,114,1);

INSERT INTO RATING VALUES(15,115,1);

INSERT INTO RATING VALUES(16,116,0);

-------------------------

select

CO.cus\_gender as `Gender`,

count(CO.cus\_gender) as `NoOfCustomers`

from(

select

c.cus\_id,

c.cus\_name,

c.cus\_gender

from

customer c

inner join `order` o

on

c.cus\_id = o.CUS\_ID

where

o.ORD\_AMOUNT >=3000

group by

c.cus\_id

) AS CO

group by

CO.cus\_gender

------------------------------------------

#4 – Display the total number of customers based on gender who have placed individual orders of worth at least Rs.3000.

select

o.\*,

p.pro\_name,

p.pro\_desc

from

`order` o

inner join supplier\_pricing sp

inner join product p on (

o.pricing\_id = sp.pricing\_id

and sp.pro\_id = p.pro\_id

)

where

o.cus\_id = 2

---------------------------------

#5 - Display all the orders along with product name ordered by a customer having Customer\_Id = 2

select

s.\*,

NoOfProducts

from

supplier s

inner join (

select

supp\_id,

count(pro\_id) as NoOfProducts

from

supplier\_pricing

group by

supp\_id

having

NoOfProducts > 1

) as sp on s.supp\_id = sp.supp\_id

----------------------

#6 – Display the Supplier details who can supply more than one product.

select \* from supplier\_pricing;

select

pro\_id,

min(supp\_price) as Minimum\_Price\_Of\_Product

from

supplier\_pricing

group by

pro\_id;

select

p.cat\_id,

p.pro\_name

from

product p;

-- PRODUCT table inner join prev query from supplier\_pricing

select

p.cat\_id,

p.pro\_name,

sp.pro\_id,

Minimum\_Price\_Of\_Product

from

product p

inner join (

select

pro\_id,

min(supp\_price) as Minimum\_Price\_Of\_Product

from

supplier\_pricing

group by

pro\_id

) as SP on sp.pro\_id = p.pro\_id;

select

c.cat\_name as Category\_Name,

P\_SP.cat\_id as Category\_ID,

min(Minimum\_Price\_Of\_Product)

from

category c

inner join(

select

p.cat\_id,

p.pro\_name,

sp.pro\_id,

Minimum\_Price\_Of\_Product

from

product p

inner join (

select

pro\_id,

min(supp\_price) as Minimum\_Price\_Of\_Product

from

supplier\_pricing

group by

pro\_id

) as SP on sp.pro\_id = p.pro\_id

) as P\_SP on c.cat\_id = P\_SP.cat\_id

group by

c.cat\_id

--------------------------

#7- Find the least expensive product from each category and print the table with category id, name, product name and price of the product.

-- Display the Id and Name of the Product ordered after "2021-10-05".

select

p.pro\_id,

p.pro\_name

from

`order` o

inner join supplier\_pricing sp

inner join product p on (

o.pricing\_id = sp.pricing\_id

and sp.pro\_id = p.pro\_id

and o.ord\_date >= '2021-10-05'

)

group by

p.pro\_id

---------------------------------

#8- Display the Id and Name of the Product ordered after “2021-10-05”.

-- Display customer name and gender whose name start or end with character 'A'.

-- like operator

select

cus\_name,

cus\_gender

from customer

where

cus\_name like 'A%'

or cus\_name like '%A'

-----------------------------------

#9- Display customer name and gender whose names start or end with character 'A'.

select

supp\_id as SUPPLIER\_ID,

supp\_name as SUPPLIER\_NAME,

AverageRating,

CASE

WHEN AverageRating = 5 THEN 'Excellent Service'

WHEN AverageRating > 4 THEN 'Good Service'

WHEN AverageRating > 2 THEN 'Average Service'

ELSE 'Poor Service'

END as Type\_of\_Service

from

(

select

s.supp\_id,

s.supp\_name,

avg(r.rat\_ratstars) as AverageRating

from

rating r

inner join `order` o

inner join supplier\_pricing sp

inner join supplier s on (

r.ord\_id = o.ord\_id

and o.pricing\_id = sp.pricing\_id

and sp.supp\_id = s.supp\_id

)

group by

supp\_id

order by

supp\_id

) as R\_O\_SP\_S;

#10 – Create a stored procedure to display supplier id, name, Rating (Average rating of all the products sold by every customer) and Type\_of\_Service. For Type\_of\_Service, If rating = 5, print “Excellent Service”, If rating > 4 print “Good Service”, If rating > 2print “Average Service” else print “Poor Service”. Note that there should be one rating per supplier.

CREATE DEFINER=`root`@`localhost` PROCEDURE `ListSupplierRatingDetails`()

BEGIN

select

supp\_id as SUPPLIER\_ID,

supp\_name as SUPPLIER\_NAME,

AverageRating,

CASE

WHEN AverageRating = 5 THEN 'Excellent Service'

WHEN AverageRating > 4 THEN 'Good Service'

WHEN AverageRating > 2 THEN 'Average Service'

ELSE 'Poor Service'

END as Type\_of\_Service

from

(

select

s.supp\_id,

s.supp\_name,

avg(r.rat\_ratstars) as AverageRating

from

rating r

inner join `order` o

inner join supplier\_pricing sp

inner join supplier s on (

r.ord\_id = o.ord\_id

and o.pricing\_id = sp.pricing\_id

and sp.supp\_id = s.supp\_id

)

group by

supp\_id

order by

supp\_id

) as R\_O\_SP\_S;

END

---------------

call ListSupplierRatingDetails();