create database comptest;

USE comptest;

CREATE TABLE customer(

custID INT auto\_increment not null,

fname varchar(100),

lname varchar(100),

customerType varchar(15),

PRIMARY KEY(custID)

);

CREATE TABLE laptop(

laptop\_type varchar(30) not null,

cost decimal(10,2),

PRIMARY KEY (laptop\_type)

);

create table online\_customer(

online\_id int auto\_increment,

custID int,

primary key(online\_id),

foreign key (custID) references customer(custID)

);

/\*we split online\_customer and online\_customer\_details into two seperate tables because it was difficult to enter fake data

tp populate the table as well as the custID at the same time\*/

CREATE TABLE online\_customer\_details(

online\_id INT,

card\_num int unique auto\_increment,

email VARCHAR(100),

password VARCHAR(20),

PRIMARY KEY(online\_id),

FOREIGN KEY (online\_id) REFERENCES online\_customer(online\_id)

) auto\_increment = 100000000;

/\* when a purchase is made from a branch it should be added to the branch purchase table (may be able to use a trigger here)\*/

create table company\_inventory(

laptop\_serial\_num int auto\_increment,

laptop\_type varchar(30),

branch\_num int,

primary key(laptop\_serial\_num),

foreign key (laptop\_type) references laptop(laptop\_type) on delete cascade

) auto\_increment = 1000000;

CREATE TABLE branch1(

laptop\_serial\_num int,

PRIMARY KEY(laptop\_serial\_num),

FOREIGN KEY (laptop\_serial\_num) REFERENCES company\_inventory(laptop\_serial\_num) on delete cascade

) ;

CREATE TABLE branch2(

laptop\_serial\_num int,

PRIMARY KEY(laptop\_serial\_num),

FOREIGN KEY (laptop\_serial\_num) REFERENCES company\_inventory(laptop\_serial\_num) on delete cascade

) ;

CREATE TABLE branch3(

laptop\_serial\_num int,

PRIMARY KEY(laptop\_serial\_num),

FOREIGN KEY (laptop\_serial\_num) REFERENCES company\_inventory(laptop\_serial\_num) on delete cascade

) ;

CREATE TABLE branch1\_purchase(

purchase\_id INT auto\_increment,

laptop\_serial\_num INT,

laptop\_type varchar(30),

purchase\_date datetime,

custID INT,

primary key(purchase\_id),

FOREIGN KEY (laptop\_type) REFERENCES laptop(laptop\_type),

FOREIGN KEY (custID) REFERENCES customer(custID )

);

CREATE TABLE branch2\_purchase(

purchase\_id INT auto\_increment,

laptop\_serial\_num INT,

laptop\_type varchar(30),

purchase\_date datetime,

custID INT,

primary key(purchase\_id),

FOREIGN KEY (laptop\_type) REFERENCES laptop(laptop\_type),

FOREIGN KEY (custID) REFERENCES customer(custID )

);

CREATE TABLE branch3\_purchase(

purchase\_id INT auto\_increment,

laptop\_serial\_num INT,

laptop\_type varchar(30),

purchase\_date datetime,

custID INT,

primary key(purchase\_id),

FOREIGN KEY (laptop\_type) REFERENCES laptop(laptop\_type),

FOREIGN KEY (custID) REFERENCES customer(custID )

);

drop table if exists cart\_Details;

CREATE TABLE cart\_Details(

online\_id int,

laptop\_serial\_num int,

date\_added datetime,

PRIMARY KEY(laptop\_serial\_num, online\_id),

FOREIGN KEY (laptop\_serial\_num) REFERENCES company\_inventory(laptop\_serial\_num) on delete cascade,

FOREIGN KEY (online\_id) REFERENCES online\_customer(online\_id) on delete cascade

);

drop table if exists online\_purchase;

CREATE TABLE online\_purchase(

online\_purchase\_id INT auto\_increment not null,

online\_id INT,

laptop\_serial\_num int,

purchase\_date DATE,

PRIMARY KEY(online\_purchase\_id),

FOREIGN KEY (online\_id) REFERENCES cart\_Details(online\_id) on delete cascade

);

drop table if exists online\_receipt;

CREATE TABLE online\_receipt(

tracking\_number INT auto\_increment not null,

online\_purchase\_id INT,

total\_cost decimal(10,2),

PRIMARY KEY(tracking\_number),

FOREIGN KEY (online\_purchase\_id) REFERENCES online\_purchase(online\_purchase\_id)

);

create table stars(

star\_rating int,

star\_meaning varchar(100),

primary key(star\_rating)

);

drop table if exists review;

CREATE TABLE review(

review\_id int auto\_increment,

online\_id INT,

laptop\_type VARCHAR(30),

star\_rating INT,

PRIMARY KEY(review\_id),

FOREIGN KEY (online\_id) REFERENCES online\_customer(online\_id),

FOREIGN KEY (laptop\_type) REFERENCES laptop(laptop\_type),

FOREIGN KEY (star\_rating) REFERENCES stars(star\_rating)

);

CREATE TABLE credit\_card\_status(

card\_num int,

online\_id int,

card\_status VARCHAR(20),

PRIMARY KEY(card\_num),

FOREIGN KEY(card\_num)REFERENCES online\_customer\_details(card\_num) ,

foreign key (online\_id) references online\_customer\_details(online\_id)

);

CREATE VIEW bank\_database AS

SELECT customer.fname,customer.lname, credit\_card\_status.card\_num,credit\_card\_status.card\_status

FROM customer join online\_customer\_details join online\_customer join credit\_card\_status

ON customer.custID = online\_customer.custID AND

online\_customer.online\_id = online\_customer\_details.online\_id and

online\_customer\_details.card\_num = credit\_card\_status.card\_num;

/\*drop table if exists warehouse;

CREATE TABLE warehouse(

warehouse\_id INT auto\_increment not null,

laptop\_type int,

quantity int,

PRIMARY KEY(warehouse\_id),

FOREIGN KEY (laptop\_type) REFERENCES laptop(laptop\_type)

);\*/

DELIMITER //

CREATE TRIGGER update\_online\_customer

AFTER INSERT on customer

FOR EACH ROW

BEGIN

INSERT INTO online\_customer(custID) VALUES(new.custID);

END //

DELIMITER ;

DELIMITER $$

create trigger add\_to\_branch /\*adds laptops to assigned branch table\*/

after insert on company\_inventory

for each row begin

if new.branch\_num = '1' then

insert into branch1(laptop\_serial\_num) values(new.laptop\_serial\_num) ;

end if;

if new.branch\_num = '2' then

insert into branch2(laptop\_serial\_num) values(new.laptop\_serial\_num) ;

end if;

if new.branch\_num = '3' then

insert into branch3(laptop\_serial\_num) values(new.laptop\_serial\_num) ;

end if;

END $$

DELIMITER ;

DELIMITER $$

create trigger check\_card /\*checks if credit card is valid before purchase\*/

after insert on cart\_details

for each row

begin

if (select distinct credit\_card\_status.card\_status from credit\_card\_status join cart\_details on credit\_card\_status.online\_id = new.online\_id) = 'under'

then

insert into online\_purchase(laptop\_serial\_num, online\_id) values(new.laptop\_serial\_num, new.online\_id);

end if;

end $$

delimiter ;

DELIMITER $$

create trigger online\_purchase\_to\_branch /\*adds purchases to branch purchase tables\*/

after insert on online\_purchase

for each row

begin

if exists (select new.laptop\_serial\_num from company\_inventory) then

if (select distinct company\_inventory.branch\_num from company\_inventory join online\_purchase where company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num) = 1 then

insert into branch1\_purchase(laptop\_serial\_num, custID, laptop\_type) values(new.laptop\_serial\_num, (select distinct customer.custID from customer join online\_customer join online\_purchase on

customer.custID = online\_customer.custID and online\_customer.online\_id = new.online\_id), (select distinct company\_inventory.laptop\_type from company\_inventory join online\_purchase on company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num));

end if;

if (select distinct company\_inventory.branch\_num from company\_inventory join online\_purchase where company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num) = 2 then

insert into branch2\_purchase(laptop\_serial\_num, custID, laptop\_type) values(new.laptop\_serial\_num, (select distinct customer.custID from customer join online\_customer join online\_purchase on

customer.custID = online\_customer.custID and online\_customer.online\_id = new.online\_id), (select distinct company\_inventory.laptop\_type from company\_inventory join online\_purchase on company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num));

end if;

if (select distinct company\_inventory.branch\_num from company\_inventory join online\_purchase where company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num) = 3 then

insert into branch3\_purchase(laptop\_serial\_num, custID, laptop\_type) values(new.laptop\_serial\_num, (select distinct customer.custID from customer join online\_customer join online\_purchase on

customer.custID = online\_customer.custID and online\_customer.online\_id = new.online\_id), (select distinct company\_inventory.laptop\_type from company\_inventory join online\_purchase on company\_inventory.laptop\_serial\_num = new.laptop\_serial\_num));

end if;

delete from company\_inventory where laptop\_serial\_num = new.laptop\_serial\_num;

end if;

end $$

delimiter ;

delimiter $$

create procedure search\_for\_laptop(in laptopSerialNum int) /\*this allows user to search for a particular laptop serial number\*/

begin

select branch\_num from company\_inventory where laptop\_serial\_num = laptopSerialNum;

end $$

delimiter ;

delimiter $$

create procedure most\_laptops(in laptopType varchar(30)) /\*this gives branch with most laptops of particular type\*/

begin

SELECT branch\_num, laptop\_type, COUNT(laptop\_type) AS amount

FROM company\_inventory

where laptop\_type = laptopType

GROUP BY laptop\_type

ORDER BY amount DESC

LIMIT 1;

end $$

delimiter ;

delimiter $$

create procedure most\_laptop\_sales(in branchName varchar(30)) /\*gives most laptop sales of selected branch\*/

begin

set @t1 = concat(

'SELECT laptop\_type, COUNT(\*) AS sales FROM ', branchName, ' GROUP BY laptop\_type ORDER BY sales DESC LIMIT 1');

prepare stmt3 from @t1;

execute stmt3;

deallocate prepare stmt3;

end $$

delimiter ;

delimiter $$

create procedure most\_branch\_sales() /\*gives the amount of sales for each branch, as well as branch with most sales\*/

begin

select (select count(\*) from branch1\_purchase) as branch1,

(select count(\*) from branch2\_purchase) as branch2,

(select count(\*) from branch3\_purchase) as branch3,

(select greatest(branch1, branch2, branch3)) as most\_sales;

end $$

delimiter ;

/\*bug - customer id will appear more than once if they buy from more than 1 branch\*/

delimiter $$

create procedure customer\_total\_spent(in n int) /\*gives customers that have spent more than a particular amount\*/

begin

select distinct branch2\_purchase.custID, sum(laptop.cost) as total\_spent from branch2\_purchase join laptop

where branch2\_purchase.laptop\_type = laptop.laptop\_type

group by custID

having total\_spent > n

union

select distinct branch1\_purchase.custID, sum(laptop.cost) as total\_spent from branch1\_purchase join laptop

where branch1\_purchase.laptop\_type = laptop.laptop\_type

group by custID

having total\_spent > n

union

select distinct branch3\_purchase.custID, sum(laptop.cost) as total\_spent from branch3\_purchase join laptop

where branch3\_purchase.laptop\_type = laptop.laptop\_type

group by custID

having total\_spent > n;

end $$

delimiter ;

Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Data Format | Field Size | Description | Example |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |