Shipt Coding Exercise

Answers: By Rahul Singh (rahpsing@iu.edu)

Database organization (SQL Queries) :

```
create table Shipt_Customer (
id number NOT NULL primary key,
email varchar2(40),
first_name varchar2(20),
last_name varchar2(20));

create table Shipt_Category (
id number NOT NULL primary key,
category_name varchar2(40));

create table Shipt_Order (
id number NOT NULL primary key,
order_date timestamp,
order_total double precision,
order_status varchar2(20),
customer_id number,
foreign key (customer_id) references Customer(id));

create table Shipt_Product (
id number NOT NULL primary key,
product_name varchar2(20),
quantity double precision,
price_per_unit double precision);

create table Product_Category (
product_id number NOT NULL,
category_id number NOT NULL,
primary key(product_id,category_id)
);

create table Order_Products (
order_id number NOT NULL,
product_id number NOT NULL,
quantity number NOT NULL,
primary key(order_id,product_id)
);
```

Simple insert statements :

```
insert into Shipt_Customer values(1,"rahul_singh919@yahoo.com","Rahul","Singh");
insert into Shipt_Category values(1,"Electronics");
insert into Shipt_Category values(2,"Hardware");
insert into Shipt_Order values(1,"2-20-2017",205,"In transit",1);
insert into Shipt_Product values(1,"LEDs",52,5.00);
insert into Product_Category values(1,1);
```

insert into Order_Products(1,1);

Question 3) SQL QUERY is as follows:

```
select cust.id as customer_id, cust.first_name as customer_first_name, pc.CATEGORY_ID as
category_id,sc.category_name, sum(op.quantity) as number_purchased
from Shipt_Customer cust join Shipt_Order SOR on cust.id = SOR.customer_id
join  Order_Products op on SOR.id = op.order_id
join Product_Category pc on pc.product_id  = op.product_id
join Shipt_Category sc on pc.CATEGORY_ID = sc.ID
group by cust.id, cust.first_name, pc.CATEGORY_ID, sc.category_name;
```

Question 4) Refer to class ShiptFetchDataImpl in the project submitted in package

com.shipt.service.serviceImpl.ShiptFetchDataImpl

Question 5) Refer to class ShiptFetchDataImpl in the project submitted in package

com.shipt.service.serviceImpl.ShiptFetchDataImpl

Question 6) Refer to class ShiptFetchDataImpl in the project submitted in package

com.shipt.service.serviceImpl.ShiptFetchDataImpl

Non-implementation questions:

We want to give customers the ability to create lists of products for one-click ordering of bulk items. How would you design the tables, what are the pros and cons of your approach?

Tables: Create a new table to hold the active carts for each customer. Exactly 1 active cart for each customer and not more. Once an order has been placed, the entries shall be moved to the order table and the cart entry shall be removed. (Status marked as invalid)

```
Create table Cart (
id number primary key,
customer_Id number,
total_amount double precision,
status bit default 0,
foreign key (customer_id) references Customer(id)
);

Create table Cart_Items (
cart_id number,
product_id number,
primary key(cart_id,product_id)
);
```

Pros: We are maintaining an active list of carts so that a customer can add items to his cart and come back later and simply click the “place order” but and all his items that were once selected get ordered.

Cons: Extra storage for a list of active carts.

If Shipt knew exact inventory of stores, and when facing a high traffic and limited supply of particular item, how do you distribute the inventory among customers checking out?

(Look for class Inventory in package com.shipt.util in the project file submitted)

I would have an Inventory class/object that would store the exact numbers of each item available. The method that would update the inventory once a customer chooses to order an item shall be ‘synchronized’. Thus we are ensured that even in case of heavy traffic the inventory won’t go corrupt and we shall be able to provide the first come first serve scenario.

Example :

Item is available in inventory:

Customer chooses to add item in cart but not order at that time:

Strategy 1 :While adding an item to cart a simple check shall be performed to see if the items are available or not. However, the count of items shall not be updated unless the customer eventually places the order. That way, we avoid the contention of resources from customers who simply book items in their cart and leave it idle.

Strategy 2 : Better yet, we could update the count when a customer adds an item to the cart but define a grace period within which, the item, if not ordered shall automatically be added back to the inventory (just updating the available count) while still being available in the customer’s cart.

Finally, on placing the order at a later time a check shall be made if the item was available or not and the count shall be updated in the inventory accordingly.