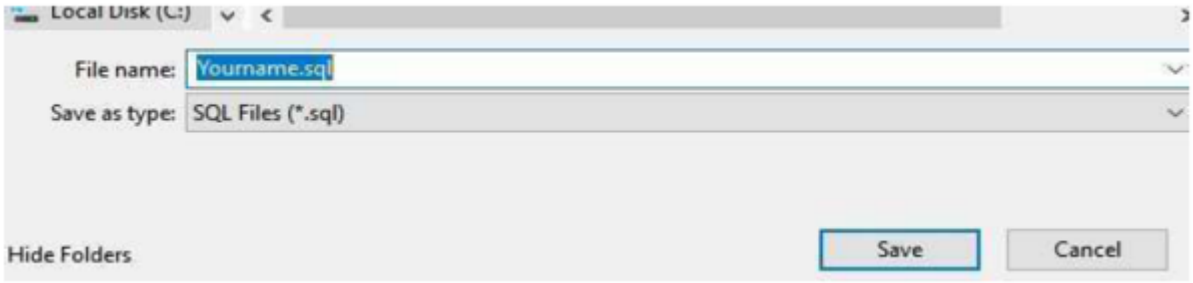
SQL Project

Each question is of 5 marks (5*10=50 marks)

How to save files?

MySQL(.SQL): After executing all the commands/answers, click on the file menu on the top left and save as Yourname.sql as shown below:



Order Management Schema details

This document captures the scenario of simple order management functionality of an online retail store.

Typical purchase scenario: A customer places an order for N products specifying quantity for each line item of the order. Every product belongs to a product class (or category). All products ordered in one order, are shipped to customer's address (in India or Outside) by a shipper in one shipment. Order can be paid using either Cash, Credit Card or Net Banking.

There can be customers who may not have placed any order. Few customers would have cancelled their orders (As a whole order, no cancellation of individual item allowed). Few orders may be 'In process' status. There can also be products that were never purchased.

Shippers use optimum sized cartons (boxes) to ship an order, based on the total volume of all products and their quantities. Dimensions of each product (L, W, H) is also stored in the database. To keep it simple, all products of an order are put in one single appropriately sized carton for shipping.

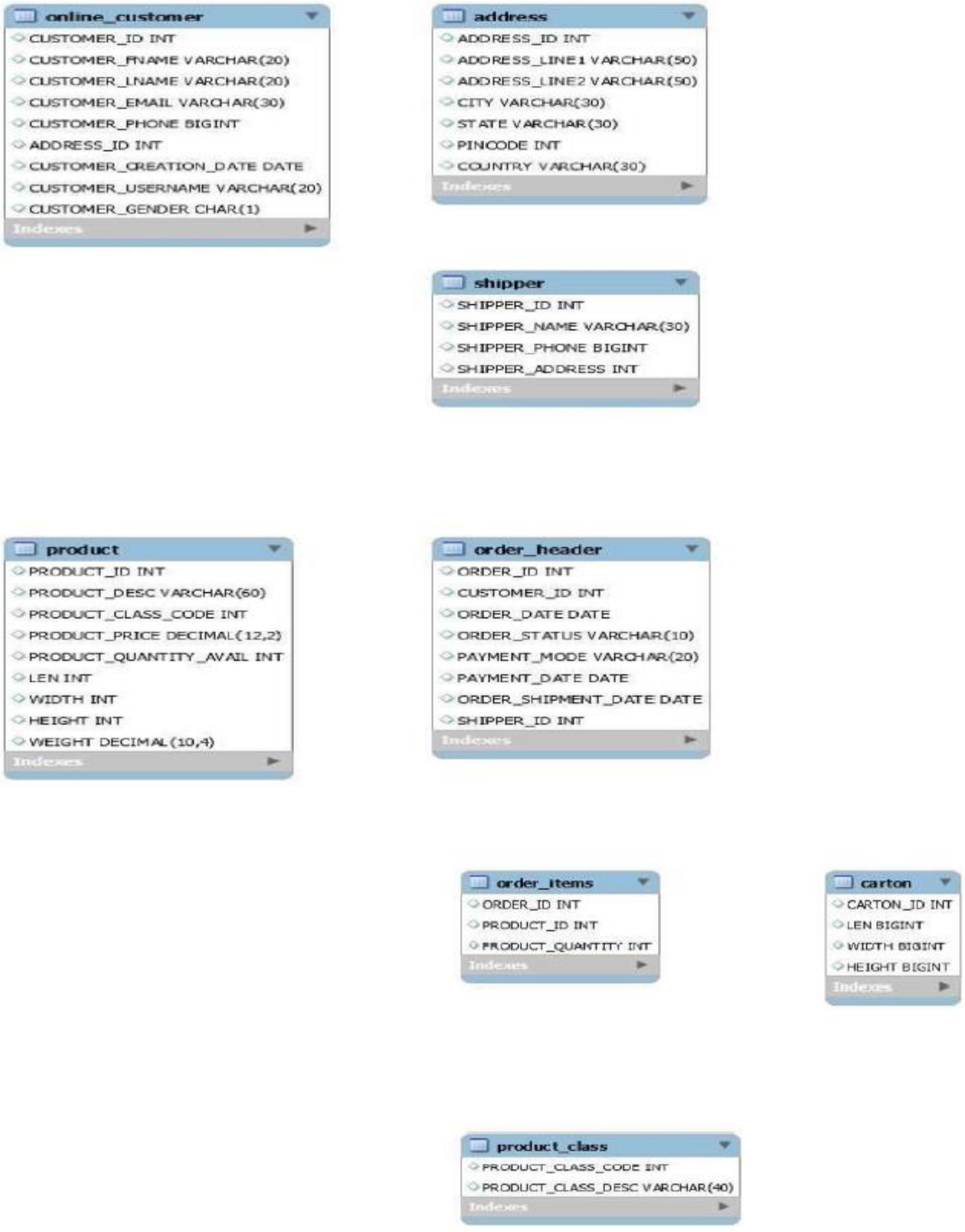
Project MYSQL

You are hired by a chain of online retail stores "RRL". They provided you with orders database and seek answers to the following queries as the results from these queries will help the company in making data driven decisions that will impact the overall growth of the online retail store.

All the questions come under MYSQL and the queries should be executed in MYSQL.

(SQL script- orders.sql)

ER Diagram



The above schema (ER diagrams are just for your reference) to understand what each table contains.

Only .SQL file needs to be submitted, no Business Report is required.

The best practice to understand the tables is to run each of them one by one and try to understand each table columns and its values. Also, try to compare tables to understand if there is any primary key / common columns on which we can join and get some more understanding about data.

For example:

What does Product_Quanity_Available in product table mean? Is it the total products (including orders placed i.e. sold (product_quantity in order_items table) and not sold?
How can we figure out this?
Let's run a query to see all the columns in product table, then run a query to see all columns of order_items table. Now, join both the tables and check for different conditions like if product_quantity_available equal to; less than or greater than product_quantity. Here, we found that all the conditions are satisfied which assures us that product_quantity_available contains only those products which are currently available excluding the once which are sold.
This is one way to understand the tables as in Industry we might not have complete information.

Q1. Write a query to display **customer_id**, **customer full name** with their title (**Mr/Ms**), both **first name and last name** are in **upper case**, **customer_email**, **customer_creation_year** and **display customer's category** after applying below categorization rules:

i. if CUSTOMER_CREATION_DATE year <2005 then category A

ii. if CUSTOMER_CREATION_DATE year >=2005 and <2011 then category B

iii. if CUSTOMER_CREATION_DATE year>= 2011 then category C

Expected 52 rows in final output.

[Note: TABLE to be used - ONLINE_CUSTOMER TABLE]

Hint:Use CASE statement. create customer_creation_year column with the help of customer_creation_date, no permanent change in the table is required.
(*Here don't UPDATE or DELETE the columns in the table nor CREATE new tables for your representation. A new column name can be used as an alias for your manipulation in case if you are going to use a CASE statement.)*

Q2. Write a query to display the following information for the **products which have not been sold: product_id, product_desc, product_quantity_avail, product_price,**
inventory values (product_quantity_avail * product_price), New_Price after applying discount as per below criteria. Sort the output with respect to decreasing value of Inventory_Value.

i) If Product Price > 20,000 then apply 20% discount

ii) If Product Price > 10,000 then apply 15% discount

iii) if Product Price =< 10,000 then apply 10% discount

Expected 13 rows in final output.

[NOTE: TABLES to be used - PRODUCT, ORDER_ITEMS TABLE]

Hint: Use CASE statement, no permanent change in table required.
(*Here don't UPDATE or DELETE the columns in the table nor CREATE new tables for your representation. A new column name can be used as an alias for your manipulation in case if you are going to use a CASE statement.)*

Q3. Write a query to display **Product_class_code, Product_class_desc, Count of Product type in each product class, Inventory Value (p.product_quantity_avail*p.product_price).** Information should be displayed for only those **product_class_code which have more than 1,00,000 Inventory Value**. Sort the output with respect to decreasing value of Inventory_Value.

Expected 9 rows in final output.

[NOTE: TABLES to be used - PRODUCT, PRODUCT_CLASS]

Hint: 'count of product type in each product class' is the count of product_id based on product_class_code.

Q4. Write a query to display **customer_id, full name, customer_email, customer_phone and country of customers** who have cancelled all the orders placed by them.

Expected 1 row in the final output

[NOTE: TABLES to be used - ONLINE_CUSTOMER, ADDRESSS, OREDER_HEADER]

Hint: USE SUBQUERY

Q5. Write a query to display **Shipper name, City to which it is catering, num of customer catered by the shipper in the city , number of consignment delivered to that city for Shipper DHL**

Expected 9 rows in the final output

[NOTE: TABLES to be used - SHIPPER, ONLINE_CUSTOMER, ADDRESSS, ORDER_HEADER]

Hint: The answer should only be based on Shipper_Name -- DHL. The main intent is to find the number of customers and the consignments catered by DHL in each city.

Q6. Write a query to display **product_id, product_desc, product_quantity_avail, quantity sold and show inventory Status of products** as per below condition:

a. For Electronics and Computer categories,

> if sales till date is Zero then show 'No Sales in past, give discount to reduce inventory',
>
> if inventory quantity is less than 10% of quantity sold, show 'Low inventory, need to add inventory',
>
> if inventory quantity is less than 50% of quantity sold, show 'Medium inventory, need to add some inventory',
>
> if inventory quantity is more or equal to 50% of quantity sold, show 'Sufficient inventory'

b. For Mobiles and Watches categories,

> if sales till date is Zero then show 'No Sales in past, give discount to reduce inventory',
>
> if inventory quantity is less than 20% of quantity sold, show 'Low inventory, need to add inventory',
>
> if inventory quantity is less than 60% of quantity sold, show 'Medium inventory, need to add some inventory',
>
> if inventory quantity is more or equal to 60% of quantity sold, show 'Sufficient inventory'

c. Rest of the categories,

if sales till date is Zero then show 'No Sales in past, give discount to reduce inventory',

if inventory quantity is less than 30% of quantity sold, show 'Low inventory, need to add inventory',

if inventory quantity is less than 70% of quantity sold, show 'Medium inventory, need to add some inventory',

if inventory quantity is more or equal to 70% of quantity sold, show 'Sufficient inventory'

Expected 60 rows in final output

[NOTE: (USE CASE statement) ; TABLES to be used - PRODUCT, PRODUCT_CLASS, ORDER_ITEMS]

Hint: quantity sold here is product_quantity in order_items table.
You may use multiple case statements to show inventory status (Low stock, In stock, and Enough stock) that meets both the conditions i.e. on products as well as on quantity.
The meaning of the rest of the categories, means products apart from electronics, computers, mobiles, and watches.

Q7. Write a query to display **order_id and volume of the biggest order** (in terms of volume) that can fit **in carton id 10 .**

Expected 1 row in final output

[NOTE: TABLES to be used - CARTON, ORDER_ITEMS, PRODUCT]

Hint: First find the volume of carton id 10 and then find the order id with products having total volume less than the volume of carton id 10

Q8. Write a query to display **customer id, customer full name, total quantity and total value (quantity*price) shipped** where mode of payment is **Cash** and **customer last name** starts with **'G'**

Expected 2 rows in final output

[NOTE: TABLES to be used - ONLINE_CUSTOMER, ORDER_ITEMS, PRODUCT, ORDER_HEADER]

Q9. Write a query to display **product_id, product_desc and total quantity of products** which are sold together **with product id 201** and are **not shipped to city Bangalore and New Delhi.**
Expected 5 rows in final output
[NOTE: TABLES to be used - ORDER_ITEMS, PRODUCT, ORDER_HEADER, ONLINE_CUSTOMER, ADDRESS]

*Hint: Display the output in **descending order with respect to the sum of product_quantity**. (USE SUB-QUERY) In final output show only those products , product_id's which are sold with 201 product_id (201 should not be there in output) and are shipped except Bangalore and New Delhi*

Q10. Write a query to display the **order_id, customer_id and customer fullname, total quantity of products shipped** for **order ids which are even** and shipped to address where **pincode is not starting with "5"**

Expected 15 rows in final output

[NOTE: TABLES to be used - ONLINE_CUSTOMER, ORDER_HEADER, ORDER_ITEMS, ADDRESS]