SQL project

Submitted by

ANIMESH HALDER

Project Problem Statement:

You are hired by a chain of online retail stores **"Reliant retail limited"**. 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.

1st part(Q1-Q6) comes under **SQLite** and queries should be executed in **DB Browser**. (Database used- New Orders.db)

2nd part(Q7-Q10) comes under **MYSQL** and the queries should be executed in **MYSQL**. (SQL Script used - new Orders.sql)

1. Write a query to Display the product details (product_class_code, product_id, product_desc, product_price,) as per the following criteria and sort them in descending order of category: a. If the category is 2050, increase the price by 2000 b. If the category is 2051, increase the price by 500 c. If the category is 2052, increase the price by 600. Hint: Use case statement. no permanent change in table required. (60 ROWS) [NOTE: PRODUCT TABLE]

Execution finished without errors.
Result: 60 rows returned in 19ms

At line 1:

```
SELECT P.PRODUCT_CLASS_CODE, P.PRODUCT_ID,
P.PRODUCT_DESC,P.product_price,
CASE P.PRODUCT_CLASS_CODE
WHEN 2050 THEN P.product_price + 2000 -- to increase price of 2050 product
WHEN 2051 THEN P.product_price + 500 -- to increase price of 2051 product
WHEN 2052 THEN P.product_price + 600 -- to increase price of 2052 product
ELSE P.PRODUCT_PRICE
END 'CALCULATED_PRICE'
FROM PRODUCT P INNER JOIN PRODUCT_CLASS PC
ON P.PRODUCT_CLASS_CODE = PC.PRODUCT_CLASS_CODE
ORDER BY P.PRODUCT_CLASS_CODE DESC;
```

	PRODUCT_CLASS_CODE	PRODUCT_ID	PRODUCT_DESC	PRODUCT_PRICE	CALCULATED_PRICE
1	3002	99991	Dell Targus Synergy 2.0 Backpack	999	999
2	3002	99992	Tom Clancy's Ghost Recon: Future ...	999	999
3	3002	99993	Nokia 1280 (Black)	999	999
4	3001	99994	HP Deskjet 2050 All-in-One - J510a ...	3749	3749
5	3001	99995	LG MS-2049UW Solo Microwave	4800	4800
6	3001	99996	Nokia Asha 200 (Graphite)	4070	4070
7	3000	99999	Samsung Galaxy Tab 2 P3100	19300	19300
8	3000	99998	Nikon Coolpix L810 Bridge	14987	14987
9	3000	99997	Sony Xperia U (Black White)	16499	16499
10	3000	99990	Quanta 4 Port USB Hub	500	500

2. Write a query to display (product_class_desc, product_id, product_desc, product_quantity_avail) and Show inventory status of products as below as per their available quantity: a. For Electronics and Computer categories, if available quantity is <= 10, show 'Low stock', 11 <= qty <= 30, show 'In stock', >= 31, show 'Enough stock' b. For Stationery and Clothes categories, if qty <= 20, show 'Low stock', 21 <= qty <= 80, show 'In stock', >= 81, show 'Enough stock' c. Rest of the categories, if qty <= 15 – 'Low Stock', 16 <= qty <= 50 – 'In Stock', >= 51 – 'Enough stock' For all categories, if available quantity is 0, show 'Out of stock'. Hint: Use case statement. (60 ROWS) [NOTE: TABLES TO BE USED – product, product_class]

	PRODUCT_CLASS_DESC	PRODUCT_ID	PRODUCT_DESC	PRODUCT_QUANTITY_AVAIL	INVENTORY_STATUS
1	Electronics	221	Cybershot DWC-W325 Camera	5	Low stock
2	Electronics	202	Sams 192 L4 Single-door Refrigerator	15	In stock
3	Electronics	203	Jocky Speaker Music System HT32	19	In stock
4	Electronics	201	Sky LED 102 CM TV	30	In stock
5	Toys	204	Cricket Set for Boys	10	Low stock
6	Toys	208	Doll House	12	Low stock

Execution finished without errors.
Result: 60 rows returned in 10ms

At line 1:

```
SELECT PC.PRODUCT_CLASS_DESC, P.PRODUCT_ID, P.PRODUCT_DESC, P.PRODUCT_QUANTITY_AVAIL,
--For Electronics and Computer categories
CASE
WHEN P.PRODUCT_CLASS_CODE in (2050,2053) THEN
CASE
WHEN P.PRODUCT_QUANTITY_AVAIL =0 THEN 'Out of stock'
WHEN P.PRODUCT_QUANTITY_AVAIL <=10 THEN 'Low stock'
WHEN (P.PRODUCT_QUANTITY_AVAIL >=11 and P.PRODUCT_QUANTITY_AVAIL <=30) THEN 'In stock'
WHEN P.PRODUCT_QUANTITY_AVAIL >=31 THEN 'Enough stock'
END
--For Stationery and Clothes categories
WHEN P.PRODUCT_CLASS_CODE in (2052,2056) THEN
CASE
WHEN P.PRODUCT_QUANTITY_AVAIL =0 THEN 'Out of stock'
WHEN P.PRODUCT_QUANTITY_AVAIL <=20 THEN 'Low stock'
WHEN (P.PRODUCT_QUANTITY_AVAIL >=21 and P.PRODUCT_QUANTITY_AVAIL <=80) THEN 'In stock'
WHEN P.PRODUCT_QUANTITY_AVAIL >=81 THEN 'Enough stock'
END
--Rest of the products
ELSE
CASE
WHEN P.PRODUCT_QUANTITY_AVAIL =0 THEN 'Out of stock'
WHEN P.PRODUCT_QUANTITY_AVAIL <=15 THEN 'Low stock'
WHEN (P.PRODUCT_QUANTITY_AVAIL >=16 and P.PRODUCT_QUANTITY_AVAIL <=50) THEN 'In stock'
WHEN P.PRODUCT_QUANTITY_AVAIL >=51 THEN 'Enough stock'
END
END INVENTORY_STATUS
FROM PRODUCT P INNER JOIN PRODUCT_CLASS PC
ON P.PRODUCT_CLASS_CODE = PC.PRODUCT_CLASS_CODE
ORDER BY P.PRODUCT_CLASS_CODE, P.PRODUCT_QUANTITY_AVAIL ASC;
```

3. Write a query to show the number of cities in all countries other than USA & MALAYSIA, with more than 1 city, in the descending order of CITIES. (2 rows) [NOTE: ADDRESS TABLE]

Execution finished without errors.
Result: 2 rows returned in 7ms

At line 1:

```
SELECT A.COUNTRY, COUNT(CITY) COUNT_OF_CITIES
FROM ADDRESS A
GROUP BY A.COUNTRY
HAVING A.COUNTRY NOT IN ('USA','Malaysia') AND COUNT(CITY) > 1
ORDER BY COUNT(CITY) DESC;
```

	COUNTRY	COUNT_OF_CITIES
1	India	26
2	Singapore	6

4. Write a query to display the customer_id,customer full name ,city,pincode,and order details (order id, product class desc, product desc, subtotal(product_quantity * product_price)) for orders shipped to cities whose pin codes do not have any 0s in them. Sort the output on customer name and subtotal. (52 ROWS) [NOTE: TABLE TO BE USED - online_customer, address, order_header, order_items, product, product_class]

Execution finished without errors.
Result: 52 rows returned in 24ms

At line 1:

```
SELECT OC.CUSTOMER_ID,(OC.CUSTOMER_FNAME||' '||OC.CUSTOMER_LNAME) AS CUSTOMER_FULL_NAME,A.CITY,
A.PINCODE,OI.ORDER_ID,
PC.PRODUCT_CLASS_DESC,P.PRODUCT_DESC,(OI.PRODUCT_QUANTITY*P.PRODUCT_PRICE) AS SUBTOTAL
FROM
ONLINE_CUSTOMER OC
INNER JOIN ADDRESS A ON OC.ADDRESS_ID = A.ADDRESS_ID
INNER JOIN ORDER_HEADER OH ON OH.CUSTOMER_ID = OC.CUSTOMER_ID
INNER JOIN ORDER_ITEMS OI ON OI.ORDER_ID = OH.ORDER_ID
INNER JOIN PRODUCT P ON P.PRODUCT_ID = OI.PRODUCT_ID
INNER JOIN PRODUCT_CLASS PC ON PC.PRODUCT_CLASS_CODE = P.PRODUCT_CLASS_CODE
WHERE OH.ORDER_STATUS='Shipped' AND A.PINCODE NOT LIKE '%0%'
ORDER BY CUSTOMER_FULL_NAME, SUBTOTAL;
```

	CUSTOMER_ID	CUSTOMER_FULL_NAME	CITY	PINCODE	ORDER_ID	PRODUCT_CLASS_DESC	PRODUCT_DESC	SUBTOTAL
1	30	Anita Kohli	Amherst	14228	10059	Electronics	Cybershot DWC-W325 Camera	5300
2	19	Bharti Subhash	Dharmapuri	635897	10054	Clothes	Infant Sleepwear Blue	500
3	19	Bharti Subhash	Dharmapuri	635897	10034	Bags	Women Hand Bag	1600
4	19	Bharti Subhash	Dharmapuri	635897	10034	Kitchen Items	Phils Wah Collection Juicer JM12	2029
5	19	Bharti Subhash	Dharmapuri	635897	10054	Bags	HP ODC Laptop Bag 15.5	3390
6	10	Bidhan C.Roy	Hosur	635235	10070	Stationery	4M Post It Pad 3.5	70

5. Write a Query to display product id,product description,totalquantity(sum(product quantity) for a given item whose product id is 201 and which item has been bought along with it maximum no. of times. Display only one record which has the maximum value for total quantity in this scenario. (USE SUB-QUERY)(1 ROW)[NOTE : ORDER_ITEMS TABLE,PRODUCT TABLE]

Execution finished without errors.
Result: 1 rows returned in 7ms

At line 1:

```
SELECT OI.PRODUCT_ID,
P.PRODUCT_DESC,
SUM(OI.PRODUCT_QUANTITY) AS TOTAL_QUANTITY
FROM ORDER_ITEMS OI
INNER JOIN PRODUCT P ON P.PRODUCT_ID = OI.PRODUCT_ID
WHERE OI.ORDER_ID IN ( SELECT DISTINCT ORDER_ID FROM ORDER_ITEMS OI_S WHERE PRODUCT_ID = 201)
AND OI.PRODUCT_ID != 201
GROUP BY OI.PRODUCT_ID
ORDER BY TOTAL_QUANTITY DESC
LIMIT 1;
```

	PRODUCT_ID	PRODUCT_DESC	TOTAL_QUANTITY
1	218	Shell Fingertip Ball Pen	30

6. Write a query to display the customer_id,customer name, email and order details (order id, product desc,product qty, subtotal(product_quantity * product_price)) for all customers even if they have not ordered any item.(225 ROWS) [NOTE: TABLE TO BE USED - online_customer, order_header, order_items, product]

Execution finished without errors.
Result: 225 rows returned in 54ms

At line 1:

```
SELECT OC.CUSTOMER_ID,
(OC.CUSTOMER_FNAME ||' '|| OC.CUSTOMER_LNAME) AS CUSTOMER_FULL_NAME,
OC.CUSTOMER_EMAIL,
OH.ORDER_ID,P.PRODUCT_DESC,
OI.PRODUCT_QUANTITY,
(OI.PRODUCT_QUANTITY*P.PRODUCT_PRICE) AS SUBTOTAL
FROM ONLINE_CUSTOMER OC
INNER JOIN ADDRESS A ON OC.ADDRESS_ID = A.ADDRESS_ID
LEFT JOIN ORDER_HEADER OH ON OC.CUSTOMER_ID = OH.CUSTOMER_ID
LEFT JOIN ORDER_ITEMS OI ON OH.ORDER_ID = OI.ORDER_ID
LEFT JOIN PRODUCT P ON OI.PRODUCT_ID = P.PRODUCT_ID
LEFT JOIN PRODUCT_CLASS PC ON P.PRODUCT_CLASS_CODE = PC.PRODUCT_CLASS_CODE;
```

	CUSTOMER_ID	CUSTOMER_FULL_NAME	CUSTOMER_EMAIL	ORDER_ID	PRODUCT_DESC	PRODUCT_QUANTITY	SUBTOTAL
1	1	Jennifer Wilson	jen_w@gmail.com	10001	Sky LED 102 CM TV	1	35000
2	1	Jennifer Wilson	jen_w@gmail.com	10001	Infant Sleepwear Blue	3	750
3	1	Jennifer Wilson	jen_w@gmail.com	10001	Samsung Galaxy On6	1	14000
4	1	Jennifer Wilson	jen_w@gmail.com	10001	Foldable Premium Chair	1	4000
5	1	Jennifer Wilson	jen_w@gmail.com	10011	OnePlus 6 Smart Phone	1	32500
6	1	Jennifer Wilson	jen_w@gmail.com	10011	Samsung Galaxy On6	2	28000

7. Write a query to display carton id, (len*width*height) as carton_vol and identify the optimum carton (carton with the least volume whose volume is greater than the total volume of all items (len * width * height * product_quantity)) for a given order whose order id is 10006, Assume all items of an order are packed into one single carton (box). (1 ROW) [NOTE: CARTON TABLE]

```
USE ORDERS;
SHOW TABLES;

SELECT
        C.CARTON_ID, (C.LEN*C.WIDTH*C.HEIGHT) as CARTON_VOL

FROM ORDERS.CARTON C
    WHERE (C.LEN*C.WIDTH*C.HEIGHT)>=  (
                                        SELECT
                                        SUM(P.LEN*P.WIDTH*P.HEIGHT*PRODUCT_QUANTITY) AS VOL
                                        FROM ORDERS.ORDER_HEADER OH
                                        INNER JOIN ORDERS.ORDER_ITEMS OI ON OH.ORDER_ID = OI.ORDER_ID
                                        INNER JOIN ORDERS.PRODUCT P ON OI.PRODUCT_ID = P.PRODUCT_ID
                                        WHERE OH.ORDER_ID =10006
                                      )
ORDER BY (C.LEN*C.WIDTH*C.HEIGHT) ASC
LIMIT 1
```

Result Grid

CARTON_ID	CARTON_VOL
40	1215000000

8. Write a query to display details (customer id,customer fullname,order id,product quantity) of customers who bought more than ten (i.e. total order qty) products with credit card or Net banking as the mode of payment per shipped order. (6 ROWS) [NOTE: TABLES TO BE USED - online_customer, order_header, order_items,]

```sql
SELECT
        OC.CUSTOMER_ID,
        CONCAT(OC.CUSTOMER_FNAME,' ',OC.CUSTOMER_LNAME) AS CUSTOMER_FULL_NAME,
        OH.ORDER_ID,
        OI.PRODUCT_QUANTITY,
        SUM(OI.PRODUCT_QUANTITY) AS TOTAL_ORDER_QUANTITY

FROM ORDERS.ONLINE_CUSTOMER OC
INNER JOIN ORDERS.ORDER_HEADER OH ON OH.CUSTOMER_ID = OC.CUSTOMER_ID
INNER JOIN ORDERS.ORDER_ITEMS OI ON OI.ORDER_ID = OH.ORDER_ID
        WHERE OH.ORDER_STATUS = 'Shipped'
                GROUP BY OH.ORDER_ID
                        HAVING TOTAL_ORDER_QUANTITY > 10
                                ORDER BY TOTAL_ORDER_QUANTITY;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CUSTOMER_ID	CUSTOMER_FULL_NAME	ORDER_ID	PRODUCT_QUANTITY	TOTAL_ORDER_QUANTITY
6	Anita Goswami	10010	1	11
5	Ramya Ravinder	10003	1	12
17	Prasad Shetty	10043	10	12
10	Bidhan C.Roy	10015	1	13
6	Anita Goswami	10017	2	14
6	Anita Goswami	10006	1	15
2	Jackson Davis	10002	2	16
2	Jackson Davis	10012	1	23
7	Ashwathi Bhatt	10008	5	25
11	Vikas Jha	10018	15	40
3	Komal Choudhary	10007	20	53

9. Write a query to display the order_id, customer id and cutomer full name of customers starting with the alphabet "A" along with (product_quantity) as total quantity of products shipped for order ids > 10030. (5 ROWS) [NOTE: TABLES TO BE USED - online_customer, order_header, order_items]

```sql
SELECT
        OH.ORDER_ID,
        OC.CUSTOMER_ID,
        CONCAT(CUSTOMER_FNAME,' ',CUSTOMER_LNAME) AS CUSTOMER_FULL_NAME,
        SUM(OI.PRODUCT_QUANTITY) AS TOTAL_PRODUCT_QUANTITY

FROM ORDERS.ONLINE_CUSTOMER OC
INNER JOIN ORDERS.ORDER_HEADER OH ON OH.CUSTOMER_ID = OC.CUSTOMER_ID
INNER JOIN ORDERS.ORDER_ITEMS OI ON OI.ORDER_ID = OH.ORDER_ID
            WHERE OH.ORDER_STATUS = 'Shipped' AND OH.ORDER_ID > 10030 AND OC.CUSTOMER_FNAME LIKE 'A%'
                GROUP BY OH.ORDER_ID
                        ORDER BY ORDER_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ORDER_ID	CUSTOMER_ID	CUSTOMER_FULL_NAME	TOTAL_PRODUCT_QUANTITY
10032	7	Ashwathi Bhatt	7
10039	14	Avinash Dutta	1
10059	30	Anita Kohli	1
10068	51	Ahmad Bin Gh Azali	3
10069	23	Anna Pinnock	4

Result Grid

Form Editor

10. Write a query to display product class description ,total quantity (sum(product_quantity),Total value (product_quantity * product price) and show which class of products have been shipped highest(Quantity) to countries outside India other than USA? Also show the total value of those items. (1 ROWS)[NOTE:PRODUCT TABLE,ADDRESS TABLE,ONLINE_CUSTOMER TABLE,ORDER_HEADER TABLE,ORDER_ITEMS TABLE,PRODUCT_CLASS TABLE]

```
SELECT
        PC.PRODUCT_CLASS_CODE,
        PC.PRODUCT_CLASS_DESC,
        SUM(OI.PRODUCT_QUANTITY) AS TOTAL_QUANTITY,
        SUM(OI.PRODUCT_QUANTITY*P.PRODUCT_PRICE) AS TOTAL_VALUE

FROM ORDERS.ORDER_ITEMS OI
INNER JOIN ORDERS.ORDER_HEADER OH ON OH.ORDER_ID = OI.ORDER_ID
INNER JOIN ORDERS.ONLINE_CUSTOMER OC ON OC.CUSTOMER_ID = OH.CUSTOMER_ID
INNER JOIN ORDERS.PRODUCT P ON P.PRODUCT_ID = OI.PRODUCT_ID
INNER JOIN ORDERS.PRODUCT_CLASS PC ON PC.PRODUCT_CLASS_CODE = P.PRODUCT_CLASS_CODE
INNER JOIN ORDERS.ADDRESS A ON A.ADDRESS_ID = OC.ADDRESS_ID
        WHERE OH.ORDER_STATUS ='Shipped' AND A.COUNTRY NOT IN('India','USA')
            GROUP BY PC.PRODUCT_CLASS_CODE,PC.PRODUCT_CLASS_DESC
                ORDER BY TOTAL_QUANTITY DESC
                        LIMIT 1;
```

Result Grid

PRODUCT_CLASS_CODE	PRODUCT_CLASS_DESC	TOTAL_QUANTITY	TOTAL_VALUE
2056	Stationery	13	2138.00

Result 1

Read Only