**DE – Assignment**

1. Write a query to return Territory and corresponding Sales Growth. Compare growth between periods Q4-2021 vs Q3-2021. If Territory (say T123) has Sales worth 100 in Q3-2021 and Sales worth 110 in Q4-2021, then the Sales Growth will be 10% [ i.e. = ((110 - 100)/100) \* 100 ] Output the ID of the Territory and the Sales Growth. Only output these territories that had any sales in both quarters.  
     
   Tables : fct\_customer\_sales , map\_customer\_territory  
     
   **fct\_customer\_sales**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Cust\_id | Varchar |
| Prod\_sku\_id | Varchar |
| Order\_date | Datetime |
| Order\_value | Int |
| Order\_id | Varchar |

**Map\_customer\_territory**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Cust\_id | Varchar |
| Territory\_id | Varchar |

1. Definition of *Frequent* Customer: A Customer who has transacts on the platform atleast once in every 5 days since last transaction

**Table:**

**“**

CREATE TABLE SALES (

ORDER\_ID INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

CUSTOMER\_ID INT,

ORDER\_VALUE NUMERIC(10,2),

ORDER\_DATE DATETIME

);

INSERT INTO SALES

(ORDER\_ID, CUSTOMER\_ID, ORDER\_VALUE, ORDER\_DATE) VALUES

(10001,90001, 10000, '2022-02-01 09.00.00'),

(10002,90001, 10000, '2022-02-03 09.00.00'),

(10003,90001, 10000, '2022-02-07 09.00.00'),

(10004,90001, 20000, '2022-02-09 09.00.00'),

(10005,90001, 20000, '2022-02-14 09.00.00'),

(10006,90001, 10000, '2022-02-14 09.00.00'),

(10007,90001, 10000, '2022-02-17 09.00.00'),

(10009,90001, 80000, '2022-02-21 09.00.00'),

(100020,90001, 10000, '2022-02-23 09.00.00'),

(100021,90001, 10000, '2022-02-28 09.00.00'),

(10010,90002, 10000, '2022-02-01 09.00.00'),

(10013,90002, 30000, '2022-02-09 09.00.00'),

(10014,90002, 10000, '2022-02-14 09.00.00'),

(10015,90002, 10000, '2022-02-14 09.00.00'),

(10016,90002, 70000, '2022-02-17 09.00.00'),

(10017,90002, 10000, '2022-02-21 09.00.00'),

(10019,90002, 10000, '2022-02-28 09.00.00');

select \* FROM SALES

“

Write a SQL query for below questions:

1. Find which customers are *Frequent.*
2. Evaluate cumulative sum of ORDER\_VALUE for each customer in ascending order of ORDER\_DATE

Output Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ORDER\_ID** | **CUSTOMER\_ID** | **ORDER\_VALUE** | **ORDER\_DATE** | **CUM\_SUM** |
| 10001 | 90001 | 10000 | 2022-02-01 | 10000 |
| 10002 | 90001 | 10000 | 2022-02-03 | 20000 |
| 10003 | 90001 | 10000 | 2022-02-07 | 30000 |
| 10004 | 90001 | 20000 | 2022-02-09 | 50000 |
| 10005 | 90001 | 20000 | 2022-02-14 | 70000 |
| 10006 | 90001 | 10000 | 2022-02-14 | 80000 |
| 10007 | 90001 | 10000 | 2022-02-17 | 90000 |
| 10009 | 90001 | 80000 | 2022-02-21 | 170000 |
| 10010 | 90002 | 10000 | 2022-02-01 | 10000 |
| 10013 | 90002 | 30000 | 2022-02-09 | 40000 |
| 10014 | 90002 | 10000 | 2022-02-14 | 50000 |
| 10015 | 90002 | 10000 | 2022-02-14 | 60000 |
| 10016 | 90002 | 70000 | 2022-02-17 | 130000 |
| 10017 | 90002 | 10000 | 2022-02-21 | 140000 |
| 10019 | 90002 | 10000 | 2022-02-28 | 150000 |
| 100020 | 90001 | 10000 | 2022-02-23 | 160000 |
| 100021 | 90001 | 10000 | 2022-02-28 | 170000 |

1. Order IDs which constitute Top 80 percentile basis Order\_Value
2. Create a coupon\_flag which becomes active on alternate transactions, signifying availability of coupon. Assume coupon\_flag is 1 (Active) on first transaction, find number of days an offer was valid for each customer.
3. Consider the flight dataset attached. Write a Python code block to find all the travel options a passenger can take, along with flight details for the input Delhi (origin) to Mumbai (destination).



For example: If the passenger choose to travel from Agra to Pune. The result must be like-

* Agra – Pune >> 3 hours >> 1 Stop (Surat) >> Flight Layover

1. Write a python program to flatten a nested JSON to list all the available nic into dataframe. Use below JSON data for reference.

{

"count": 13,

"virtualmachine": [

{

"id": "1082e2ed-ff66-40b1-a41b-26061afd4a0b",

"name": "test-2",

"displayname": "test-2",

"securitygroup": [

{

"id": "9e649fbc-3e64-4395-9629-5e1215b34e58",

"name": "test",

"tags": []

}

],

"nic": [

{

"id": "79568b14-b377-4d4f-b024-87dc22492b8e",

"networkid": "05c0e278-7ab4-4a6d-aa9c-3158620b6471"

},

{

"id": "3d7f2818-1f19-46e7-aa98-956526c5b1ad",

"networkid": "b4648cfd-0795-43fc-9e50-6ee9ddefc5bd",

"traffictype": "Guest"

}

],

"hypervisor": "KVM",

"affinitygroup": [],

"isdynamicallyscalable": false

}

]

}

1. Design a compute resource for given problem statement. The Marketing team are running campaigns online and all the user experiences are captured via Google Analytics which is replicated into Google BigQuery. The BI analyst has gathered all the sales history data into Snowflake data warehouse. How and where can the analyst combine these two datasets in order to identify possible leads for the business?

You need to suggest a platform or compute instance where both the sources can be queried together and tables can be joined to find out recent sellers visiting the advertisements surfing through all the buying options.