Robert Gal

Advanced data management

A. Summarize one real-world business:

One of the main components to a successful business is to understand customers' spending behaviors and instrument strategies to encourage returning customers with more spending. One of the strategies is to implement a membership system based on the spending level from renting activities in our store, which gives customers rewards and special discounts. By doing so, we can encourage returning customers and attract new ones through promoting the membership system. This will stabilize our business and grow into the future.

1. Identify the specific fields that will be included in the detailed and the summary table of the report:

The report will include two tables.

Detailed : customer id, first name, last name, email, amount, payment date.

Summary: customer id, full name, total amount and member type.

2. Describe the types of data fields used for the report.

Detailed table:

customer id : INT (INTEGER) – Numeric type

first name : VARCHAR(30) – Character type

last name: VARCHAR(30) – Character type

email: VARCHAR(90) - Character type

amount: DECIMAL(5, 2) – Numeric type

payment date: DATE – Temporal data type

Summary table:

customer id :   INT (INTEGER) –   Numeric type

full name : VARCHAR(60)   – Character type

total amount: DECIMAL(5, 2) – Numeric type

member type: VARCHAR(10) – Character type

3. Identify at least two specific tables from the given dataset:

This report will use two specific tables, the customer and the payment table.

4. Identify one field that will require a custom transformation and explain why:

Customers first name and last name are transformed to full name in the summary table. This will better display all names of this report to our stakeholders. The amount is turned into ’total amount’ in order to show the total amount of spending. A new column ’member type’ is created, which transforms the total amount into ’member type’. Our store will have three member types: Gold, Silver, and Bronze, based on the level of ’total amount’. Gold membership type includes customers who spent at least 200 or more; Silver membership includes spending amount between 100 and 199; Bronze membership includes spending amount between 25 and 99. The report shows the total amount and the implementation of the new membership types to the stakeholders. This will be the new strategic plan to keep returning customers while attracting new ones maximizing the profit.

5. Explain the different business uses of the detailed and the summary section:

The report from the detailed table store every single transaction activities from our customers in the company. It will be usefull to see our customers spending behaviors in detail. The report from the summary table shows the total amount spending from each customer and their membership status. It will help to monitor our new membership system for the business.

6. Explain how frequently your report should be refreshed to remain relevant to stakeholders.

The report will refresh once a month . It will show to the stakeholders how effective our new membership system works. The analysis will inform us how we can move to the right direction with our business decisions.

B. Provide original code for function(s) in text format that perform the tranformation(s):

```
CREATE FUNCTION amount_summary_update()

RETURNS TRIGGER

LANGUAGE plpgsql

AS $$

BEGIN

DELETE FROM summary;

INSERT INTO summary(

SELECT

      customer_id,

      CONCAT(first_name,' ',last_name) AS full_name,

      sum(amount) AS total_amount,

     'bronze' AS type

FROM detailed

GROUP BY customer_id, full_name

HAVING sum(amount) BETWEEN 25 and 99

UNION

SELECT
```

```
            customer_id,
            CONCAT(first_name,' ',last_name) AS full_name,
            sum(amount) AS total_amount,
           'silver' AS type
FROM detailed
GROUP BY customer_id, full_name
HAVING sum(amount) BETWEEN 100 and 199
UNION
SELECT
            customer_id,
            CONCAT(first_name,' ',last_name) AS full_name,
            sum(amount) AS total_amount,
           'gold' AS type
FROM detailed
GROUP BY customer_id, full_name
HAVING sum(amount) >= 200
ORDER BY total_amount DESC
);

RETURN NEW;
END $$
```

C. Provide original SQL code in text format that creates the detailed and summary tables:

```
CREATE TABLE detailed(
        customer_id INT,
        first_name VARCHAR(30),
        last_name VARCHAR(30),
        email VARCHAR(90),
        amount DECIMAL(5,2),
        payment_date DATE
);



CREATE TABLE summary(
        customer_id INT,
        full_name VARCHAR(60),
        total_amount DECIMAL(5,2),
        member_type VARCHAR (10)
);
```

D. Provide an original SQL query in text format that will extract the raw data needed for the detailed section of your report:

```
INSERT INTO detailed(
	customer_id,
	first_name,
	last_name,
	email,
	amount,
	payment_date
)
SELECT
	customer_id,
	first_name,
	last_name,
	email,
	SUM(amount) AS total_amount,
	CAST(payment_date as date)
FROM customer
JOIN payment USING (customer_id)
GROUP BY customer_id, payment_date
ORDER BY payment_date DESC;
```

E. Provide original SQL code in a text format that creates a trigger on the detailed table:

```
CREATE TRIGGER update_summary

AFTER INSERT ON detailed

FOR EACH STATEMENT

EXECUTE PROCEDURE amount_summary_update()
```

F. Provide an original stored procedure in a text format that can be used to refresh the data in both the detailed and the summary tables:

```
CREATE PROCEDURE update_report()

LANGUAGE plpgsql

AS $$

BEGIN

DELETE FROM detailed;

INSERT INTO detailed(

        customer_id,

        first_name,

        last_name,

        email,

        amount,

        payment_date
)

SELECT

        customer_id,

        first_name,

        last_name,

        email,

        SUM(amount) AS total_amount,

        CAST(payment_date as date)

FROM customer

JOIN payment USING (customer_id)

GROUP BY customer_id, payment_date

ORDER BY payment_date DESC;

END $$
```

F1. Identify a relevant job scheduling tool that can be used to automate the stored procedure:

pgAgent is one of the external tool can be used for automate stored procedures. This tool can help to schedule refresh our report once a month or as frequently as required.

Calling the store procedure manually also will refresh our tables:

```
CALL update_report()
```

H: I did not use any third-party code or sources for this assignment.