1. ***Summarize one real-world written business report that can be created from the DVD Dataset from the “Labs on Demand Assessment Environment and DVD Database” attachment.***

This report will show the payments made / amounts per customers’ postal code. This information is critical for targeting marketing efforts / strategies towards areas with higher or lower sales. The detailed report will list every transaction by postal code, customer, and amount. The summary report will show total amount, number of transactions and average payment all grouped by the postal code.

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

Detailed table fields: postal\_code (from address), concatenated first\_name and last\_name (from customer), payment\_date (from payment), and amount (from payment)

Summary table: postal\_code (from address, total\_amount by sum of amount (from payments) grouped by postal code, number\_of\_payments counted by payment\_id (from payment) grouped by postal code.

1. ***Describe the types of data fields used for the report.***

The data types that will be used for this report include INTEGERS, VARCHAR, DATE, and DECIMAL (for currency values).

1. ***Identify* at least*two specific tables from the given dataset that will provide the data necessary for the detailed table section and the summary table section of the report.***

The three tables that provide the necessary data are address, customer, and payment.

1. ***Identify*at least*one field in the detailed table section that will require a custom transformation with a user-defined function and explain why it should be transformed (e.g., you might translate a field with a value of N to No and Y to Yes).***

One field that will be transformed in the detailed table will be the concatenated first and last name. This is solely for readability and convenience for whomever will be running the query.

1. ***Explain the different business uses of the detailed table section and the summary table section of the report.***

The detailed table section shows customer behavior, trends, and patterns at a micro-level. This can help personalizing marketing strategies. The detail report can also show anomalies or outliers because every transaction is listed. Which also can ensure data integrity or accuracy for the aggregated data.

The summary table section summarizes this data into a high-level overview and paints a picture of the raw data. This table will help with the strategic decision-making behind the marketing towards customers. Stakeholders can use this table to look at long time trends and patterns.

While the detailed table is crucial in providing raw data, it may not provide as high level of review as the summary table. And while the summary table does provide this high level overview with trends and patterns, it lacks the details that the detailed table provides.

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

For strategic market planning, this report should be run monthly. This ensures the stakeholders are receiving current and accurate data. This also for fast and accurate marketing towards customers.

1. ***Provide original code for function(s) in text format that perform the transformation(s) you identified in part A4.***

--Creating first and last name concatenation

CREATE OR REPLACE FUNCTION get\_full\_name(p\_first\_name VARCHAR, p\_last\_name VARCHAR)

RETURNS VARCHAR AS $$

BEGIN

    RETURN p\_first\_name || ' ' || p\_last\_name;

END;

$$ LANGUAGE plpgsql;

1. ***Provide original SQL code in a text format that creates the detailed and summary tables to hold your report table sections.***

--Creating detailed table

DROP TABLE IF EXISTS detailed\_payment\_report;

CREATE TABLE detailed\_payment\_report (

    postal\_code VARCHAR(10),

    customer\_name VARCHAR(100),

    amount DECIMAL (5, 2),

    payment\_date DATE

);

--Crating summary table

DROP TABLE IF EXISTS summary\_payment\_report;

CREATE TABLE summary\_payment\_report (

    postal\_code VARCHAR(10),

    total\_amount DECIMAL(10, 2),

    number\_of\_payments INT

);

1. ***Provide an original SQL query in a text format that will extract the raw data needed for the detailed section of your report from the source database.***

--Inserting raw data into detailed table section

INSERT INTO detailed\_payment\_report (postal\_code, customer\_name, amount, payment\_date)

    SELECT

        a.postal\_code,

        get\_full\_name(c.first\_name, c.last\_name),

        p.amount,

        p.payment\_date

    FROM payment p

    JOIN customer c ON p.customer\_id = c.customer\_id

    JOIN address a ON c.address\_id = a.address\_id;

1. ***Provide original SQL code in a text format that creates a trigger on the detailed table of the report that will continually update the summary table as data is added to the detailed table.***

--Populating summary table

INSERT INTO summary\_payment\_report (postal\_code, total\_amount, number\_of\_payments)

    SELECT

        postal\_code,

        SUM(amount) AS total\_amount,

        COUNT(\*) AS number\_payments

    FROM

        detailed\_payment\_report

    GROUP BY

        postal\_code;

--Trigger function - when new rows are added to detailed table then trigger function

--will update data on summary table

CREATE OR REPLACE FUNCTION refresh\_summary\_table()

RETURNS TRIGGER

LANGUAGE plpgsql

AS $$

BEGIN

  -- Delete existing records from the summary table

  DELETE FROM summary\_payment\_report;

  -- Repopulate the summary table based on the current state of the detailed table

  INSERT INTO summary\_payment\_report (postal\_code, number\_of\_payments, total\_amount)

  SELECT

    postal\_code,

    COUNT(\*) AS number\_of\_payments,

    SUM(amount) AS total\_amount

  FROM

    detailed\_payment\_report

  GROUP BY

    postal\_code;

  RETURN NEW;

END;

$$;

CREATE TRIGGER update\_summary\_after\_insert

AFTER INSERT

ON detailed\_payment\_report

FOR EACH STATEMENT

EXECUTE FUNCTION refresh\_summary\_table();

***F.  Provide an original stored procedure in a text format that can be used to refresh the data in*both *the detailed table and summary table. The procedure should clear the contents of the detailed table and summary table and perform the raw data extraction from part D.***

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

A job scheduling tool that can be used to automate this stored procedure could be pgAgent. pgAgent and pgAdmin would be installed and configured to execute scheduled jobs. This could be scheduled as monthly, or changed depending on the needs of the business.

--Procedure to refresh both detailed and summary tables

CREATE OR REPLACE PROCEDURE refresh\_report\_data()

LANGUAGE plpgsql

AS $$

BEGIN

    -- Deleting existing tables

    DELETE FROM detailed\_payment\_report;

    DELETE FROM summary\_payment\_report;

    -- Repopulating the detailed\_payment\_report table

    INSERT INTO detailed\_payment\_report (postal\_code, customer\_name, amount, payment\_date)

    SELECT

        a.postal\_code,

        get\_full\_name(c.first\_name, c.last\_name),

        p.amount,

        p.payment\_date

    FROM

        payment p

    JOIN customer c ON p.customer\_id = c.customer\_id

    JOIN address a ON c.address\_id = a.address\_id;

    -- Repopulating the summary\_payment\_report table based on the new detailed table

    INSERT INTO summary\_payment\_report (postal\_code, total\_amount, number\_of\_payments)

    SELECT

        postal\_code,

    SUM(amount) AS total\_amount,

        COUNT(\*) AS number\_of\_payments

    FROM

        detailed\_payment\_report

    GROUP BY

        postal\_code;

END;

$$;

***H.  Acknowledge all utilized sources, including any sources of third-party code, using in-text citations and references. If no sources are used, clearly declare that no sources were used to support your submission.***

None were used.