Anthony Hart  
000956500  
Advanced Data Management – D191

Business Report

A. 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.

* One real-world business report that can be generated from the DVD Dataset is *Lost Monthly Revenue from Inactive Customers by Store*, which would show the average monthly spend of inactive customers at each store. The business could then perform market research on those customers, asking why they are no longer actively renting DVDs from their business, potentially addressing a larger issue at the organization (such as too costly of late fees, poor experience, or lack of desired inventory) or attempt to regain them as customers.

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

* Summary Table
  + Customer : store\_id (int)
  + Calculation : num\_inactive\_customers (int)
  + Calculation : avg\_spend\_per\_month (double precision)
  + Calculation : num\_payments (int)
* Detailed Table
  + Customer : store\_id (int)
  + Customer : customer\_id (int)
  + Customer: active (int)
  + Calculation : full\_name (varchar)
  + Customer : email (varchar)
  + Customer : create\_date (date)
  + Payment : amount (double precision)
  + Payment : payment\_date (timestamp)
  + Payment : payment\_id (int)

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

* The types of fields in the reports are int, double precision, varchar, date, and timestamp.
* Int is used for countable data.
* Double precision is used for fields with decimal points, such as currency.
* Varchar is used for textual data, such as name and email.
* Date is used for dates.
* Timestamps are used to determine the exact moment something occurred, such as a payment.

3. 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 Customer and Payment tables will provide details for both types of reports.

4. 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).

* The detail table will require a concatenation of the customer first\_name and last\_name. This is for easier readability of the name.

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

* The summary table section is designed to give the management team insight into more strategic data. For example, it’s helpful to know if a specific store has a high number of inactive users, or simply a high amount of lost monthly revenue. This information will help the stakeholders determine necessary actions to take, along with priorities on those actions.
* The detailed table, though, is designed to answer a question in further detail. For example, a possible follow-up question to the Summary Report might be “what are the last purchase dates for each of the inactive customers”? The Detailed Report would provide this information to address drill-down questions such as this.

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

* The report should be updated anytime a customer’s *active* flag is changed, either from TRUE to FALSE or visa-versa. If a customer goes from active to inactive, the business will want to include their details in the report. If the opposite happens, the business will want to exclude their data, as it’s no longer relevant to the question being posed against the data.

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

SELECT

    c.first\_name || ' ' || c.last\_name AS full\_name,

INTO inactive\_customers

FROM customer c

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

-- Generate Detailed Report table --

-- Report details all payments made by inactive customers --

DROP TABLE IF EXISTS inactive\_customers;

SELECT

    c.store\_id,

    c.customer\_id,

    c.active,

    c.first\_name || ' ' || c.last\_name AS full\_name,

    c.email,

    c.create\_date,

    p.amount,

    p.payment\_date,

    p.payment\_id

INTO inactive\_customers

FROM customer c

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

WHERE active = 0

ORDER BY c.store\_id, c.customer\_id, p.payment\_date;

-- Generate Summary Report table --

-- Report indicates average loss of revenue per month due to inactive customers at each store --

DROP TABLE IF EXISTS inactive\_customers\_by\_store;

SELECT

    store\_id,

    COUNT(DISTINCT customer\_id) AS num\_inactive\_customers,

    SUM(amount)/COUNT(DISTINCT DATE\_TRUNC('month', payment\_date))/num\_inactive\_customers AS avg\_spend\_per\_month,

    COUNT(store\_id) AS num\_payments

INTO inactive\_customers\_by\_store

FROM inactive\_customers

GROUP BY store\_id

* The lines INTO *inactive\_customers* and INTO *inactive\_customers\_by\_store* are the commands which create the tables. This code drops the table, selects the data, then creates the table and enters all data into the table at the same time.

D. 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.

SELECT

    c.store\_id,

    c.customer\_id,

    c.active,

    c.first\_name || ' ' || c.last\_name AS full\_name,

    c.email,

    c.create\_date,

    p.amount,

    p.payment\_date,

    p.payment\_id

INTO inactive\_customers

FROM customer c

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

WHERE active = 0

ORDER BY c.store\_id, c.customer\_id, p.payment\_date;

* The code above extracts the data directly from the customer and payment tables, then adds it into the newly created *inactive\_customers* table.

E. 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.

CREATE TRIGGER inactive\_customer\_update

    AFTER INSERT OR DELETE ON inactive\_customers

    FOR EACH STATEMENT

    EXECUTE PROCEDURE refresh\_inactive\_customers\_by\_store();

* In the code above, *inactive\_customers* is the detailed 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.

-- Refresh data in the DETAIL report by clearing all data and re-inserting it --

CREATE OR REPLACE FUNCTION refresh\_inactive\_customers()

RETURNS TRIGGER LANGUAGE plpgsql AS $$

BEGIN

    DELETE FROM inactive\_customers;

    INSERT INTO inactive\_customers

    SELECT

        c.store\_id,

        c.customer\_id,

        c.active,

        c.first\_name || ' ' || c.last\_name AS full\_name,

        c.email,

        c.create\_date,

        p.amount,

        p.payment\_date,

        p.payment\_id

    FROM customer c

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

    WHERE active = 0

    ORDER BY c.store\_id, c.customer\_id, p.payment\_date;

    RETURN NEW;

END;

$$;

-- Create a trigger to refresh inactive\_customers after each update or delete --

CREATE TRIGGER customer\_update\_or\_delete

    AFTER INSERT UPDATE OR DELETE ON customer

    FOR EACH STATEMENT

    EXECUTE PROCEDURE refresh\_inactive\_customers();

-- Refresh data in the SUMMARY report by clearing all data and re-insterting it--

CREATE OR REPLACE FUNCTION refresh\_inactive\_customers\_by\_store()

RETURNS TRIGGER LANGUAGE plpgsql AS $$

BEGIN

    DELETE FROM inactive\_customers\_by\_store;

    INSERT INTO inactive\_customers\_by\_store

    SELECT

        store\_id,

        COUNT(DISTINCT customer\_id) AS num\_inactive\_customers,

        SUM(amount)/COUNT(DISTINCT DATE\_TRUNC('month', payment\_date))/num\_inactive\_customers AS avg\_loss\_per\_month,

        COUNT(store\_id) AS num\_payments

    FROM inactive\_customers

    GROUP BY store\_id;

    RETURN NEW;

END;

$$;

-- Create a trigger to refresh the SUMMARY report table after each insert or delete --

CREATE TRIGGER inactive\_customer\_update

    AFTER INSERT OR DELETE ON inactive\_customers

    FOR EACH STATEMENT

    EXECUTE PROCEDURE refresh\_inactive\_customers\_by\_store();

* This code is split into two separate stored procedures. The first procedure executes based on changes made to the *customer* table, which then updates the *inactive\_customers* table. The second procedure runs when the *inactive\_customers* table (or detailed report) is updated, which then updates the *inactive\_customers\_by\_store* table (or summary report).
* Both procedures work in conjunction with each other, and it’s written such that other stored procedures can be run based on updates to the *inactive\_customers* table, assuming additional reports may be created. This allows scalability, as well as easier maintenance since the procedures are broken down into specific functions.

G. Provide a Panopto video recording that includes the presenter and a vocalized demonstration of the functionality of the code used for the analysis.

* The video has been uploaded to the folder *Advanced Data Management D191 | D326 (Student Creators) [assignments]*. [Here is a quick link](https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=8b99d711-7313-48a2-a3d3-b074003bec5f) to the video.

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.

* I did not use any third-party code, in-text citations, or references.