Jonathan Bartolo

D326 Performance Assessment

7/1/2024

*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 useful business report that can be created from the DVD Dataset is a report on which actor’s movies bring in the most revenue.

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

The following fields will be included in the *detailed* table:

Actor\_id, first\_name, last\_name, film\_id, title, rental\_date, return\_date, payment\_amount

The following fields will be included in the *summary* table:

First\_name, last\_name, total\_revenue

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

The types of data fields that will be used for the detailed table are as follows: Actor\_id as an integer which identifies each actor, first\_name as variable length string containing the first name of an actor, last\_name as a variable length string containing the last name of an actor, film\_id as an integer which identifies a film, title as a variable length string containing the title of the film, rental\_date as a date identifying when the rental occurred, return\_date as a date identifying when the rental was returned, payment\_amount as a decimal identifying how much was paid

The types of data fields that will be used for the summary table are as follows:

First\_name as a variable length string containing the first name of the actor, last\_name as a variable length string containing the last name of the actor, total\_revenue as a decimal containing the total revenue generated by the actor’s movies

*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 detailed table will use the film table for film\_id and title, film\_actor table for actor\_id, actor table for first\_name and last\_name, rental table for rental\_date and return\_date, and payment table for payment\_amount.

The summary table will source from the detailed table.

*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).*

Rental\_date and return\_date will be transformed from timestamp to date to enhance readability.

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

The *detailed* table section provides insights on individual transactions and can be used to determine which movies are rented more frequently so more stock should be ordered.

The *summary* table section provides a higher level of insight on which actor’s movies generate the most revenue. This information can be used for corporate planning and trend analysis so that when actors have new movies being released, corporate can have an idea on how popular a rental it will be.

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

My report should be refreshed on a monthly basis. This allows for strategic planning and trend analysis

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

CREATE OR REPLACE FUNCTION ConvertTimestampToDate(ts TIMESTAMP)

RETURNS DATE AS $$

BEGIN

RETURN DATE(ts);

END;

$$ LANGUAGE plpgsql;

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

CREATE TABLE Detailed (

Actor\_id INT,

First\_name VARCHAR(45),

Last\_name VARCHAR(45),

Film\_id INT,

Title VARCHAR(255),

Rental\_date DATE,

Return\_date DATE,

Payment\_amount DEC(5,2)

);

CREATE TABLE Summary (

First\_name VARCHAR(45),

Last\_name VARCHAR(45),

Total\_revenue DEC(8,2)

);

*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.*

INSERT INTO Detailed

SELECT a.actor\_id, a.first\_name, a.last\_name, i.film\_id, f.title, ConvertTimestampToDate(r.rental\_date), ConvertTimestampToDate(r.return\_date), p.amount

FROM rental r

JOIN payment p ON r.rental\_id = p.rental\_id

JOIN inventory i ON r.inventory\_id = i.inventory\_id

JOIN film f ON i.film\_id = f.film\_id

JOIN film\_actor fa ON f.film\_id = fa.film\_id

JOIN actor a ON fa.actor\_id = a.actor\_id;

*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 OR REPLACE FUNCTION summary\_trigger()

RETURNS TRIGGER AS $$

BEGIN

DELETE FROM summary;

INSERT INTO summary SELECT first\_name, last\_name, SUM(payment\_amount)

FROM detailed

GROUP BY first\_name, last\_name;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER new\_summary

AFTER INSERT

ON detailed

FOR EACH STATEMENT

EXECUTE PROCEDURE summary\_trigger();

*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.*

CREATE OR REPLACE PROCEDURE refresh\_tables()

AS $$

BEGIN

DELETE FROM detailed;

DELETE FROM summary;

INSERT INTO Detailed

SELECT a.actor\_id, a.first\_name, a.last\_name, i.film\_id, f.title, ConvertTimestampToDate(r.rental\_date), ConvertTimestampToDate(r.return\_date), p.amount

FROM rental r

JOIN payment p ON r.rental\_id = p.rental\_id

JOIN inventory i ON r.inventory\_id = i.inventory\_id

JOIN film f ON i.film\_id = f.film\_id

JOIN film\_actor fa ON f.film\_id = fa.film\_id

JOIN actor a ON fa.actor\_id = a.actor\_id;

RETURN;

END;

$$ LANGUAGE plpgsql;

CALL refresh\_tables();

SELECT \* FROM detailed;

SELECT \* FROM summary;

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

pgAgent is a job scheduling tool that can be installed in pgAdmin and works with Postgres databases.

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

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=59cbe63d-fe0d-4977-8d95-b1a80151d117

*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.*

No sources were used.