Alexandra Beno D191 PA

Student ID 011095454

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.

The databases will be polled to determine which actors are the most popular among customers by seeing which are rented the most often.

A1. Identify the specific fields that will be included in the detailed table and the summary table of the report.

The data in this report will be taken from many different tables. From the film table, the film_id and title will be used. From film_actor, actor_id and film_id. From actor, actor_id, first_name, and last_name. From rental, rental_id and inventory_id. From inventory, inventory_id and film_id. The full_name field is a combination of the first_name and last_name fields from the actor table. The full_name field and a new field called times_rented that stores an integer are included in the summary table, while all other fields (and the full_name field again) are included in the detailed table.

A2. Describe the types of data fields used for the report.

The data in this report is comprised of many different data types. The film_id, actor_id, rental_id, and inventory_id fields contain integers. The title field contains a varchar with a max size of 255 characters. The first_name and last_name fields contain varchars with a max size of 45 characters. The full_name field generated by the function in A4 is a varchar with a max size of 91.

A3. 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 data in this report will be taken from many different tables. This includes the film table, the film_id table, the film_actor table, the actor table, the inventory table, and the rental table.

A4. 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).

In the actor table, the First Name and Last Name of each actor are separated into two different fields. These will be combined into one with a custom transformation with a user-defined function in order to make the names easier to read and reduce time spent cross-referencing different fields. (EX: “FirstName” + “Lastname” = “FirstName LastName”)

A5. Explain the different business uses of the detailed table section and the summary table section of the report.

The detailed table provides information about every rental and which film was being rented. This level of granularity allows the user to perform forecasting and predictions, as well as examine trends that may not be obvious from the summary table.

The summary table will provide a quick summary of the rentals made by customers. The top and worst performing actors will be ranked, allowing the business to see which films have actors that are better to have on display and which films should be rotated out for other offerings.

A6. Explain how frequently your report should be refreshed to remain relevant to stakeholders.

The data should be updated as frequently as possible to ensure accurate record-keeping. However, to ensure business accuracy, it should be refreshed at a minimum of once a month so that the business is not making decisions based on data that is out of date.

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

```
CREATE OR REPLACE FUNCTION first_plus_last (f VARCHAR(45), l VARCHAR(45))

RETURNS VARCHAR(91)

LANGUAGE plpgsql

AS $$
```

```
DECLARE
	r VARCHAR(91);
BEGIN
	r = f || ' ' || l;
	RETURN r;
END;
$$;
```

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

```
CREATE OR REPLACE FUNCTION generate_tables()
RETURNS VOID
LANGUAGE plpgsql
AS $$
BEGIN
	CREATE TABLE detailed_report (
		film_id INT,
		title VARCHAR(255),
		actor_id SMALLINT,
		full_name VARCHAR(91),
		rental_id INT,
		inventory_id int
	);
	CREATE TABLE summary_report (
		full_name VARCHAR(91),
		times_rented INTEGER
```

```
        );
END;
$$;
```

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.

```
--This is a part of the procedure detailed in part F
        SELECT
                t1.film_id,
                t1.title,
                t2.actor_id,
                first_plus_last( t3.first_name, t3.last_name ),
                t5.rental_id,
                t4.inventory_id
        FROM
                film t1
        INNER JOIN
                film_actor t2 ON t1.film_id = t2.film_id
        INNER JOIN
                actor t3 ON t2.actor_id = t3.actor_id
        INNER JOIN
                inventory t4 ON t1.film_id = t4.film_id
        INNER JOIN
                rental t5 ON t4.inventory_id = t5.inventory_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 update_summary()

RETURNS TRIGGER

LANGUAGE plpgsql

AS $$

BEGIN

        DELETE FROM summary_report;

        INSERT INTO summary_report(full_name, times_rented)

        SELECT

                full_name,

                COUNT ( * ) AS Occurrences

                FROM detailed_report

                GROUP BY full_name

                ORDER BY Occurrences DESC;

        RETURN NEW;

END;

$$;


CREATE TRIGGER trigger_update

AFTER INSERT OR UPDATE OR DELETE ON detailed_report

EXECUTE FUNCTION update_summary();
```

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.

```
--The summary table is updated when the detailed table is updated thanks to the trigger
detailed in part E

CREATE OR REPLACE PROCEDURE fill_tables()

LANGUAGE plpgsql

AS $$

BEGIN

        DELETE FROM detailed_report;

        INSERT INTO detailed_report (film_id, title, actor_id, full_name, rental_id,
inventory_id)

        SELECT

                t1.film_id,

                t1.title,

                t2.actor_id,

                first_plus_last( t3.first_name, t3.last_name ),

                t5.rental_id,

                t4.inventory_id

        FROM

                film t1

        INNER JOIN

                film_actor t2 ON t1.film_id = t2.film_id

        INNER JOIN

                actor t3 ON t2.actor_id = t3.actor_id

        INNER JOIN

                inventory t4 ON t1.film_id = t4.film_id

        INNER JOIN

                rental t5 ON t4.inventory_id = t5.inventory_id;

END;
```

```
$$;
```

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

pgAgent is a relevant job scheduling tool that can be used to automate the stored procedure. As an addon for pgAdmin, it is built to work with the software the business is already using, and will therefor be easy to implement.

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=02d058a9-30f3-4341-bcb1-b35900ec4d8d

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 to support my submission

I. Demonstrate professional communication in the content and presentation of your submission.