BRENT BUENARTE
Student ID: 010681356
VDM2 – VDM2 TASK 1: DATA ANALYSIS

Data Analysis Business Report

Summarize one real-world written business report that can be created from the DVD Dataset:

With the number amount of gathered unstructured data, there are multiple possibilities, or ways, these data can be compiled or manipulated, setting the goals for the company to examine and achieve. One of which is generating revenue or sales for stores. Data is gathered and used to understand which film category generates the most sales per store, this will allow the company:

1. Increase inventory stocks to favored film categories and pull least favored categories.
2. Generate the number of sales per category and determine its overall price.
3. Oversee other stores and examine, which film genre generates revenue per store.

To understand, two sets of tables will be provided to benefit a business and create report. A table showcasing a detailed field of information to view field changes, such as price changes, additional stores being placed, etc. In conclusion, the main goal is to provide the top five film categories per store and determine its revenue and inventory.

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

For the detailed table that will showcase all the necessary fields for detailed information, the specific fields that will be used will be:

- name
- title
- amount
- address
- first_name

- last_name

For the summary table, I've included the following specific fields to showcase the quick view of the table for business reports:

- address_location
- name
- category_overall_price
- category_sales_count

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

The tables that will be used to provide data necessary for the detailed table and summary tables are film_category, film, inventory, rental, payment, customer, staff, store, address, city, and country.

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.

The amount in the detailed table will be converted from an INT data type into a VARCHAR type in the summary table and merge with a "$" string. This field will be called category overall price. This so the field and string can be concatenated together to form a new field.

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

The detailed table section can be used to display the name of the movie and its sales. For business to know what movie and category to pull or keep in the shelves, an number of generated sales per category is shown in the table, including the total amount of revenue it produced.

The summary table section showcases the top-rated film categories per store. If a new store were to be added and create a report on its first month sale, it will be shown in a quick summary under this table and showcase which genre of film generates high volume of sales.

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

In a real-world situation, this report should be refreshed every month or quarterly, especially because according to Stephen Follows Film Data and Education, films released by Hollywood are generally more than 100 movies a year, or about 25 movies per quarterly, so as new movies are added to the inventory, generating sales would be very different per month. After the accumulated sales every month would total the number of sales generated in a year per category.

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

```
CREATE OR REPLACE FUNCTION int_var_convert()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN

--Clear out Summary Table
DELETE FROM summarytableset;
--Concat address, city, and country to clean out data layout and make it
easily readible
INSERT INTO summarytableset (
        SELECT
                address AS "address_location",
                name,
                concat('$', TO_CHAR(SUM(amount)::numeric, '9999999999.99'))
AS "category_overall_price",
                COUNT(name) AS "category_sales_count"
        FROM detailedtableset
        GROUP BY address_location, name
        ORDER BY address_location, category_overall_price DESC
        );
RETURN NEW;
END;$$
```

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

```
/* Creation of detailedtableset, IF NOT EXISTS applied */
DROP TABLE IF EXISTS detailedTableSet;
CREATE TABLE IF NOT EXISTS detailedtableset (
	name VARCHAR(50),
	title VARCHAR(50),
	amount DECIMAL(10,2),
	address VARCHAR(50),
	first_name VARCHAR(50),
	last_name VARCHAR(50)
);

/* Creation of summarytableset, IF NOT EXISTS applied */
DROP TABLE IF EXISTS summarytableset;
CREATE TABLE IF NOT EXISTS summarytableset (
	address_location VARCHAR(50),
	name VARCHAR(20),
	category_overall_price VARCHAR(20),
	category_sales_count INT
);
```

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.

```
/* Pulls data from the database and casted into detailed table */
INSERT INTO detailedtableset (
	name,
	title,
	amount,
	address,
	first_name,
	last_name
)
SELECT
	name,
	title,
	amount,
	address,
	customer.first_name,
	customer.last_name
FROM category
INNER JOIN film_category
	ON category.category_id = film_category.category_id
INNER JOIN film
	ON film_category.film_id = film.film_id
INNER JOIN inventory
	ON film.film_id = inventory.film_id
INNER JOIN rental
	ON inventory.inventory_id = rental.inventory_id
INNER JOIN payment
	ON rental.rental_id = payment.rental_id
INNER JOIN customer
```

```
       ON payment.customer_id = customer.customer_id
INNER JOIN staff
       ON payment.staff_id = staff.staff_id
INNER JOIN store
       ON staff.store_id = store.store_id
INNER JOIN address
       ON store.address_id = address.address_id
INNER JOIN city
       ON address.city_id = city.city_id
INNER JOIN country
       ON city.country_id = country.country_id;
```

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.

```
/* This will be applied ONLY when data is inserted into the database and
into the detailed table set*/

CREATE TRIGGER alter_summary
AFTER INSERT ON detailedtableset
--only execution on a new row without reruning data query
FOR EACH STATEMENT
EXECUTE PROCEDURE int_var_convert()
```

Provide an original store procedure in a text format that can be used to refresh 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 new_detailed_table()
       LANGUAGE plpgsql
       AS $$
       BEGIN
              DELETE FROM detailedtableset;
              INSERT INTO detailedtableset (
                     name,
                     title,
                     amount,
                     address,
                     first_name,
                     last_name
              )
              SELECT
                     name,
                     title,
                     amount,
                     address,
                     customer.first_name,
```

```
                customer.last_name
            FROM category
            INNER JOIN film_category
                ON category.category_id = film_category.category_id
            INNER JOIN film
                ON film_category.film_id = film.film_id
            INNER JOIN inventory
                ON film.film_id = inventory.film_id
            INNER JOIN rental
                ON inventory.inventory_id = rental.inventory_id
            INNER JOIN payment
                ON rental.rental_id = payment.rental_id
            INNER JOIN customer
                ON payment.customer_id = customer.customer_id
            INNER JOIN staff
                ON payment.staff_id = staff.staff_id
            INNER JOIN store
                ON staff.store_id = store.store_id
            INNER JOIN address
                ON store.address_id = address.address_id
            INNER JOIN city
                ON address.city_id = city.city_id
            INNER JOIN country
                ON city.country_id = country.country_id;
    END;$$

--to call stored procedure
CALL new_detailed_table()
```

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

An employer, to keep the data updated and structured correctly, would use an external tool job scheduling application, such as *pgAgent* for PostgreSQL to create stored procedure to allow the procedure, function and trigger, either every month or quarterly.

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.

For outside sources, I only used two:

- Determine how many films are produced each year. With knowing the average movies created and shown per year, it can be used to create an idea of how many are produced and shown per month, thus adding it in the inventory and prepare to analyze generated sales of movies and its categories.
- Concatenating a string type and integer type, and realizing it if it's an acceptable approach to convert the integer type into string.

Stephen Follows. "How Many Films Are Released Each Year?" Stephen Follows, 6 July 2021, stephenfollows.com/how-many-films-are-released-each-year/

Transformation Code Tutorial: https://www.w3schools.blog/to-char-function-oracle