Jason Whitby
000755904
D191 Advanced Data Management
PA (VDM1)

Business Report Summary

A. Summarize one real-world business report that can be created from the attached Data Sets and Associated Dictionaries.

There are several different approaches to the output of reports that one could surmise from the given datasets and information inside the ‘dvdrental’ database. I focused on an aspect of patron retention that focused around discerning which customers were ‘frequent fliers’ of renting DVDs. Using that knowledge, an incentive program could be utilized to then keep them coming back to use the service to increase the bottom line for the business. Using this incentive initiative we would also make it known to those that were not recurring patrons to begin renting DVDs more often to get the perks that them recurring ones would. This would definitely help in increasing profits.

1. Describe the data used for the report.

   I used the ‘customer’ and tables from the ‘dvdrental’ database to make my summary table for the business report. The customer list in would be the specific information that would be necessary to target for the incentive program. This allowed to see whom in particular returned to rent the most often.

   The information from the number of rentals and how often would be used to emphasize which customers would be the best to target for the incentive campaign. This information could then be used to decide if there is something that can be done about the non-recurrent customers. Something such as finding a budget to develop some advertising around the incentive campaign to alert them to its benefits.

2. Identify two or more specific tables from the given dataset that will provide the data necessary for the detailed and the summary sections of the report.

   There were only two tables that were necessary to achieve the desired summary table, those tables being the ‘customer’ and ‘rental’ tables. These two tables hold all of the necessary information such as the rental IDs for movies that are linked by customer IDs to determine how often customers rent movies.

3. Identify the specific fields that will be included in the detailed and the summary sections of the report.

   The following specific fields were needed for the details section of the report:

   1. rental_id
   2. rental_date
   3. return_date

4. staff_id
5. customer_id
6. first_name
7. last_name
8. email

The following fields were used for the summary section for the inference:

1. customer_name
2. email
3. customer_id (COUNT)

4. Identify one field in the detailed section that will require a custom transformation and explain why it should be transformed. For example, you might translate a field with a value of ‘N’ to ‘No’ and ‘Y’ to ‘Yes’.

   In the case of pulling the necessary information from the customer table there was an issue with having the first and last names separated by column. For the summary table to be displayed more concisely, there needed to be a transformation to concatenate the two columns together to look better and take up less space. The transformation would put the last name first and the first name last (last, first).

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

   The details section of the report will be used to keep track of the transactional rental data in the stores. This is a more specific dataset that will allow deeper diving into a very detailed look. From what specific customers are renting to how often they are renting it and returning it.

   The summary section is a scoped down version of the details section. This would allow for a general overview that can be pushed into a graph or dashboard for stakeholders. Using the summary section for this purpose would allow for the ability to see where the budget needs to allocated for initiatives.

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

   Given the current customer count, refreshing this report should not happen too quickly. Not something such as weekly, but more on a monthly basis to keep the information relevant. If it is done too quickly it will not be able to take into account a larger trend. This could lead to faulty information that would cause advertising initiatives to go to the wrong people at the wrong time wasting capital.

F1. Explain how the stored procedure can be run on a schedule to ensure data freshness.

The stored procedure can be run on demand whenever it is deemed necessary to do so to gather more frequent information. Or, it can be chosen to run less frequently to change the trend line for what the stakeholders are trying to discern. Another way that this could be done a schedule would be to use a cron job in Unix based systems, or with a Windows scheduler. This would allow for the summary table to just be updated on a very stringent timeline.

---B. CREATE TABLES

```
--this will create an empty table that is called details
--that will retain information for the details section of the
--business section of the report

DROP TABLE IF EXISTS details;
CREATE TABLE details(
customer_id integer,
first_name varchar(45),
last_name varchar (45),
email varchar (45),
rental_id integer,
rental_date timestamp,
return_date timestamp,
staff_id integer);

--this displays all of the currently empty information
--in the details table
SELECT * FROM  details;
SELECT COUNT(rental_id) FROM details;
```

---CREATION OF THE EMPTY SUMMARY TABLE---

```
--this creates the table named summary that will contain
--the information for the summarized section of the report
DROP TABLE  IF EXISTS summary;
CREATE TABLE summary(
customer_name varchar(95),
email varchar(90),
customer_count integer);
```

---C. DATA INSERTION FOR DETAILS TABLE---

```
--this will pull in the information from the customer and
--rental tables into the details table
INSERT INTO details(
customer_id,
first_name,
last_name,
```

```
email,
rental_id,
rental_date,
return_date,
staff_id )
SELECT c.customer_id, c.first_name, c.last_name, c.email,
r.rental_id, r.rental_date, r.return_date, r.staff_id
FROM rental AS r
INNER JOIN customer AS c ON c.customer_id = r.customer_id;

--this will show the now inserted data from the customer and rental
--tables inside of the details table
SELECT * FROM details;


---D. FUNCTION CREATION---

--summary table update

--1: empties out the summary table
CREATE OR REPLACE FUNCTION refresh_summary()
RETURNS TRIGGER AS $$
BEGIN
--2: refreshes the summary table with the data that is in details table
DELETE FROM summary;
INSERT INTO summary(
SELECT
concat_ws(',', last_name, first_name) AS customer_name, email, COUNT(customer_id)
FROM details
GROUP BY customer_id, customer_name, email
ORDER BY count(customer_id)DESC
LIMIT 100);
RETURN NEW;
END; $$
LANGUAGE plpgsql


---E. TRIGGER CREATION---

--this trigger runs the refresh_summary function when there is an INSERT INTO
--in the details table
CREATE TRIGGER refresh_summary
AFTER INSERT ON details
FOR EACH STATEMENT
EXECUTE PROCEDURE refresh_summary();


---F. PROCEDURE CREATION---
CREATE OR REPLACE PROCEDURE table_refresh()
```

```
LANGUAGE plpgsql
AS $$
BEGIN

DELETE FROM details;

INSERT INTO details(
customer_id,
first_name,
last_name,
email,
rental_id,
rental_date,
return_date,
staff_id )
SELECT c.customer_id, c.first_name, c.last_name, c.email,
r.rental_id, r.rental_date, r.return_date, r.staff_id
FROM rental AS r
INNER JOIN customer AS c ON c.customer_id = r.customer_id;
END; $$

--procedure call
CALL table_refresh();

--get results
SELECT * FROM details;
SELECT * FROM summary;
```