Assessment Melita

Contents

[Assessment Brief and Summary 1](#_Toc140068620)

[Table creation 1](#_Toc140068621)

[Assessment Questions 2](#_Toc140068622)

[Conclusion and recommendations 6](#_Toc140068623)

# Assessment Brief and Summary

This assessment calls for some SQL Query and data validations. 3 tables were provided such as orders, products and date. The relationship adopted was a star schema connecting a One-to-Many relationship between Products and Orders table using ProductID and also One-to-Many relationship between date to orders table. Although there were some data integrity and consistency issues, efforts were made to clean the historical data. Also, a data validation script was written to avoid future occurrence of data integrity.

# Table creation

To better query the data, a testdb was created and individual tables using the below query.

CREATE DATABASE testdb;

Also, individual tables for orders, products and date tables were created. The query adopted Posgresql platform because it is an open source.

-- 1. Creating the tables

* Order table

CREATE TABLE Orders (

OrderID INT,

OrderDate DATE,

ProductID INT,

CustomerID INT,

SalesAmount DECIMAL(10,2),

Quantity INT

);

* Products table

CREATE TABLE Products (

ProductID INT,

ProductName VARCHAR(255),

ProductCategory VARCHAR(255)

);

* Date Table

CREATE TABLE Date (

Date DATE,

Weekday VARCHAR(255),

Year INT,

Month VARCHAR(255),

Day INT,

Holiday INT

);

# Assessment Questions

1. **Write a query to produce total sales quantity and amount per EACH day.**

SELECT

OrderDate,

SUM(Quantity) AS TotalQuantity,

SUM(SalesAmount) AS TotalAmount

FROM Orders

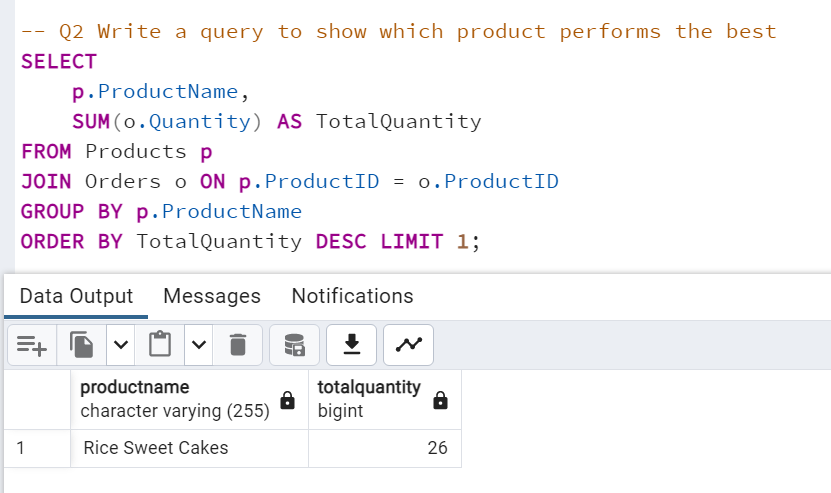
GROUP BY OrderDate;

A screenshot of a computer

Description automatically generated

1. **Write a query to show which product performs the best.**

To answer this question, there is a need to join order and product table on ProductID



SELECT

p.ProductName,

SUM(o.Quantity) AS TotalQuantity

FROM Products p

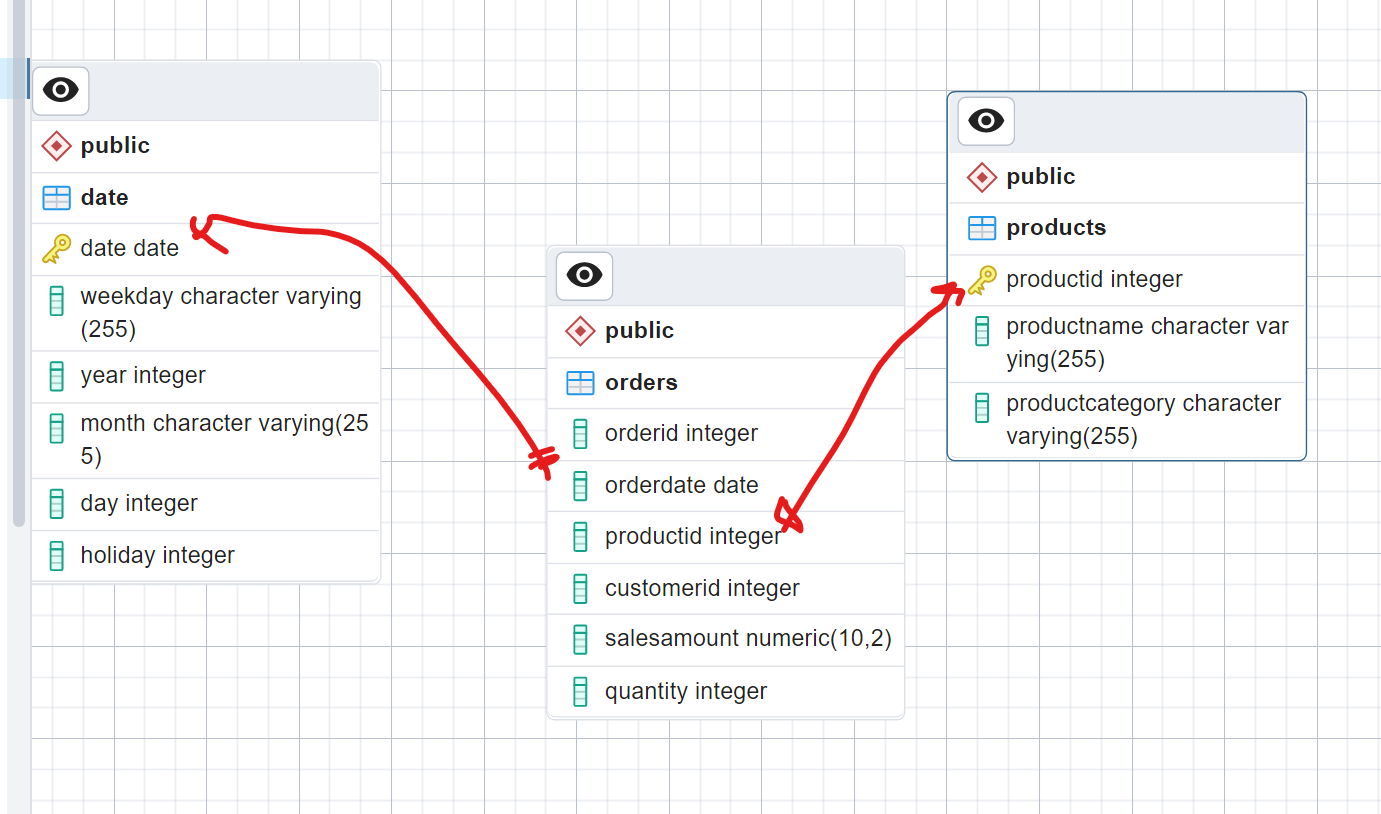
JOIN Orders o ON p.ProductID = o.ProductID

GROUP BY p.ProductName

ORDER BY TotalQuantity DESC LIMIT 1;

1. **Draw an ERD for this database.**

An ERD typically consists of entities (tables), relationships between entities, and attributes (columns) within each entity. The ERD for your database would have three entities: Orders, Products, and Date. Orders would have a one-to-many relationship with both Products and Date, indicating that each order is associated with one product and one date. Both Products and Date would have their own attributes specific to their respective tables.



1. **If you had to add a table to the database to improve the business, what would it be and why?**

A potential addition to the tables could be a "Customers" table. This table could store information about each customer, such as customer ID, name, contact details, and any other relevant information. Including a Customers table would enable better customer management, personalized marketing, and analysis of customer behavior.

1. **Are there any issues in the data provided that you would like to highlight? How would you resolve them, and how will you go around it to avoid this issue from occurring in the future?**

Duplicate order IDs with different product IDs in the Order table, quantities, and sales amounts. This could lead to inconsistencies and confusion in data analysis. To resolve this issue, one could either update the data to ensure each order ID is unique or add a primary key column to the Order table. Additionally, implementing data validation rules and constraints during data entry can help prevent such issues in the future.

Also, there were null values in the productId of the order table. I will write a data validation query such as NOT NULL in the query so that that NULL entries would not be accepted in the ProductID field of the Order table.

1. **Write a query to extract the second-largest order.**

A screenshot of a computer

Description automatically generated

SELECT \* FROM Orders

ORDER BY OrderID DESC

LIMIT 1 OFFSET 1;

1. **What is the difference between UNION and UNION ALL?**

The UNION operator is used to combine the result sets of two or more SELECT statements into a single result set, removing duplicate rows. On the other hand, UNION ALL also combines result sets but does not remove duplicate rows. UNION ALL includes all rows from each SELECT statement, even if there are duplicates.

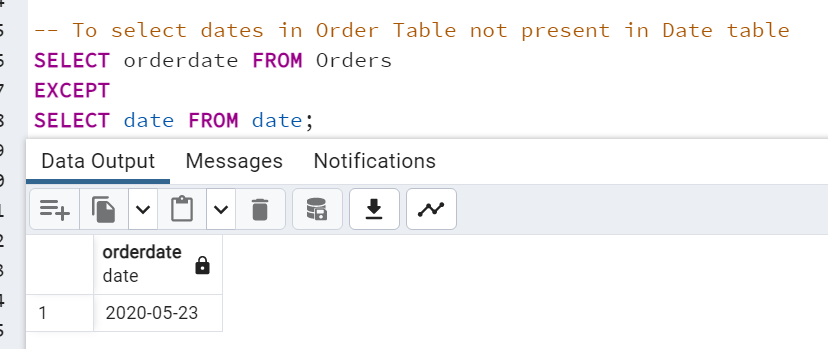
1. **How do you apply EXCEPT in the presented data?**

The EXCEPT operator is not available in all database systems, but if it is supported, it is used to retrieve distinct rows from the left-hand SELECT statement that are not present in the right-hand SELECT statement. In the provided data, let's assume we want to find the order dates that do not have a matching record in the date table. The query would look like this:

SELECT OrderDate FROM Orders

EXCEPT

SELECT date FROM date;



*This shows a major data integrity issue as there were order on the 23 of May, 2020 which was not captured in the date table.*

1. **What are Data Quality Rules? Can you give an example of how to apply one to the given examples?**

Data Quality Rules are guidelines or constraints applied to data to ensure its accuracy, consistency, and reliability. These rules help maintain high-quality data within a database. An example rule for the given examples could be ensuring that all product names in the Orders table exist in the Product table.

Here's an example query to identify any orders with invalid product names:

SELECT \* FROM Orders o

WHERE NOT EXISTS (

SELECT \* FROM Products p WHERE o.ProductID = p.ProductID );

A screenshot of a computer

Description automatically generated

1. **What is the difference between Hbase and Relational database?**

HBase is a NoSQL database that provides a distributed, scalable, and high-performance storage system. It is designed to handle large volumes of structured and semi-structured data. HBase uses a key-value store model and is typically used for handling big data workloads, especially in the context of Apache Hadoop.

A relational database, on the other hand, is based on the relational model and uses tables with predefined schemas and relationships between tables. It is suitable for structured data and provides ACID (Atomicity, Consistency, Isolation, Durability) properties. Relational databases excel at handling complex transactions, enforcing data integrity, and supporting SQL queries.

1. **What are the common data challenges?**

Common data challenges include data quality issues (incomplete, inconsistent, or inaccurate data), data integration difficulties (combining data from multiple sources), data security and privacy concerns, data governance and compliance, and managing big data and scalability.

1. **What is partitioning?**

Partitioning is a technique used in databases to divide a large table into smaller, more manageable pieces called partitions. Each partition is stored separately and can be accessed or processed independently. Partitioning provides benefits such as improved query performance, reduced storage requirements, and simplified data maintenance. Common partitioning strategies include range partitioning, hash partitioning, and list partitioning, based on criteria like specific ranges of values, hash functions, or predefined lists.

# Conclusion and recommendations

The above report showed that there is a need to improve data integrity, consistency and completeness in order to get more accurate insights out of a data. The query also displays some ways of detecting these issues of data integrity and adequately resolving them by writing a data validation query like constraints in SQL. A Primary key must be NOT NULL so that null data is not allowed into the database. It is therefore recommended that data be checked on a regular basis to affirm its integrity and completeness.