Summary Generator Program Overview

It is a public class used to that gets data from the database for given time period and generates summary which includes insights Customer information, Product information, Supplier information. It generates well formatted XML to store this data.

Queries:

- Customer Information:
  - *with order_data as (select OrderID, sum(Quantity*UnitPrice) as order_value from orderdetails group by OrderID),*
  - *customer_info as (select CustomerID, count(orders.OrderID) as num_orders, sum(order_value) as order_total_val from orders join order_data using(OrderID) where OrderDate between <start date> and <end date> group by CustomerID)*
  - *select CustomerID, num_orders, order_total_val, ContactName as customer_name, Address as street_address,City, Region, PostalCode, Country from customers join customer_info using(CustomerID);*

  This query uses two subqueries "order_data" and "customer_info" and performs join on orderdetails, orders and customers

- Product Information:
  - *with product_info as (select ProductID, ProductName,CategoryID, SupplierID, CategoryName from products natural join categories),*
  - *supplier_info as (select ProductID, ProductName, CategoryName, CompanyName from suppliers join product_info using(SupplierID)),*
  - *product_sales as (select ProductID, OrderID, Sum(Quantity) as units_sold, Sum(Quantity*orderdetails.UnitPrice)as total_value from orderdetails join orders using(OrderID) where OrderDate between <start date> and <end date> group by ProductID)*
  - *Select ProductID, ProductName, CategoryName, CompanyName, units_sold, total_value from product_sales join supplier_info using(ProductID);*

  This query uses three subqueries "product_info" , "supplier_info" and "product_sales" and performs join on products, categories, suppliers, orderdetails, orders .

- Supplier Information:
  - *with supplier_info as (select SupplierID, CompanyName, Address, City, Region, PostalCode, Country, ProductID, ProductName from suppliers join products using(SupplierID)),*
  - *product_sales as(select ProductID, OrderID, Sum(Quantity) as units_sold, Sum(Quantity*orderdetails.UnitPrice) as total_value from orderdetails join orders*

using(OrderID)where OrderDate between <start date> and <end date> group by ProductID)

- *select SupplierID,CompanyName, Address, City, Region, PostalCode, Country, sum(units_sold), sum(total_value) from product_sales join supplier_info using(ProductID) group by SupplierID;*

This query uses two subqueries "supplier_info" and "product_sales" and performs join on orderdetails, orders, suppliers and products

❖ **Approach used:**

1. After connecting with database using JDBC driver.
2. Perform queries that are explained above.
3. Once result set for each query is obtained.
4. Each result set is saved in global variables.
5. For customer information list of hashmap is used.
6. Hashmap of <String, ArrayList> to store product information.
7. Hashmap of <String, Hashmap> to store supplier information.
8. These global variables are then accessed by xml generator.
9. XML is generated with well-indented format.

❖ Why this solution is ready to be deployed:

1. **Scalable:** Variable length data structures are used to save data from the queries which means that even if database gets scaled to any further number of rows, summary generator can handle it without any issue.
2. **Flexibility:** There is no hard coding for any query or data processing that means any new field can be queried. There is also use of subqueries that means more subqueries can be added to enhance the results.
3. **XML Generation:** XML elements are added dynamically and its generated in fully indented format which meets the requirement well, so this will serve the purpose well.
4. **Performance:** Implemented solution is time-efficient as the data is stored globally and is easily accessible and even loops are run on dynamic data-structure. It can work well in the company

Submitted by:

Prabhjot Kaur
B00843735