**ADBMS LAB 3**

**NAME : Shivani Shridhar Gole**

**NUID : 002770631**

Q1. Write a SQL query using "FOR JSON PATH" and AdventureWorks2017 to get the total sales a salesperson has made. Use TotalDue in SalesOrderHeader to calculate the total sales. Return the data in the format described below.   
 Use the provided format only for formatting purposes. Don't use the data contained in the format for validation purposes.   
 Submit the code.

SQL query for generating the JSON file

select SalesPersonID,

CAST(ROUND(SUM(TotalDue),0) AS INT) AS TotalSales

FROM Sales.SalesOrderHeader

GROUP BY SalesPersonID

ORDER BY SalesPersonID ASC

FOR JSON PATH;

 

Cosmos DB query

/\*\*Cosmos DB sql query to find the sum,avg,min,max of TotalSales\*\*/

SELECT

SUM(sales.TotalSales) AS SUMOFTOTALSALES,

AVG(sales.TotalSales) AS AVERAGEOFTOTALSALES,

MAX(sales.TotalSales) AS MAXOFTOTALSALES,

MIN(sales.TotalSales) AS MINOFTOTALSALES

FROM sales

Graphical user interface, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Q2. Write a SQL query using "FOR JSON PATH" and AdventureWorks2017 to get the products an order contains. Return the data in the format described below. Return data only for orders in the sales order id range between 43660 and 43680.   
 Use the provided format only for formatting purposes. Don't use the data contained in the format for validation purposes.   
 Submit the code.   
   
[{"SalesOrderID":43660,   
 "Products":[{"ProductID":762,"OrderQty":1},   
 {"ProductID":758,"OrderQty":1}]},   
 {"SalesOrderID":43661,   
 "Products":[{"ProductID":745,"OrderQty":1},   
 {"ProductID":743,"OrderQty":1},   
 {"ProductID":747,"OrderQty":2},   
 {"ProductID":712,"OrderQty":4},   
 {"ProductID":715,"OrderQty":4},   
 {"ProductID":742,"OrderQty":2},   
 {"ProductID":775,"OrderQty":3},   
 {"ProductID":778,"OrderQty":2},   
 {"ProductID":711,"OrderQty":2},   
 {"ProductID":741,"OrderQty":2},   
 {"ProductID":776,"OrderQty":4},   
 {"ProductID":773,"OrderQty":2},   
 {"ProductID":716,"OrderQty":2},   
 {"ProductID":777,"OrderQty":2},   
 {"ProductID":708,"OrderQty":5}]}

Import the generated data into the Cosmos DB SQL API database.   
Write a SQL query for the Cosmos DB SQL API to get the number of unique products contained in each order.

SQL query for generating the JSON file

SELECT h.SalesOrderID,

(SELECT ProductID, OrderQty

FROM Sales.SalesOrderDetail

WHERE SalesOrderID = h.SalesOrderID

FOR JSON PATH)

AS Products

FROM

Sales.SalesOrderHeader AS h

WHERE SalesOrderID BETWEEN 43660 AND 43680

ORDER BY h.SalesOrderID

FOR JSON PATH;

Cosmos DB query

/\*\*Cosmos DB query for finding the count of unique products\*\*/

select sales.SalesOrderID, count(1) as UNIQUEPRODUCTCOUNT

from sales JOIN products in sales.Products

GROUP BY sales.SalesOrderID

ORDER BY sales.SalesOrderID DESC

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Q3 Write a SQL query using "FOR JSON PATH" and AdventureWorks2017 to get the top 3 products in a sales territory and a product's total sold quantity. Return the data in the format described below. TotalProductQuantity is the total sold quantity of a product.   
The top 3 products have the 3 highest total sold quantities in a territory. If there is a tie, the tie needs to be retrieved. The product quantity of an order is stored in SalesOrderDetail.   
Use the provided format only for formatting purposes. Don't use the data contained in the format for validation purposes.   
Submit the code.   
  
[{"TerritoryID":1,   
 "Top3Products":[{"ProductID":870,"TotalProductQuantity":947},   
 {"ProductID":712,"TotalProductQuantity":887},   
 {"ProductID":711,"TotalProductQuantity":794}]},   
 {"TerritoryID":2,   
 "Top3Products":[{"ProductID":715,"TotalProductQuantity":549},   
 {"ProductID":712,"TotalProductQuantity":514},   
 {"ProductID":711,"TotalProductQuantity":423}]}

Import the generated data into a Cosmos DB SQL API database. Submit a screenshot of importing results.   
Write a SQL query for the Cosmos DB SQL API to get the totals of TotalProductQuantity regardless of the sales territory for each product. Submit the code and a screenshot of the executing results.

SQL query for generating the JSON file

/\*\*Top 3 products in each Territory\*\*/

select TerritoryID,

(select top 3

ProductID,

SUM(OrderQty) as TotalProductQuantity

from Sales.SalesOrderDetail o

join sales.SalesOrderHeader h on o.SalesOrderID = h.SalesOrderID

join sales.SalesTerritory t1 on h.TerritoryID = t.TerritoryID

where t1.TerritoryID = t.TerritoryID

GROUP BY ProductID

ORDER BY TotalProductQuantity DESC

FOR JSON PATH)

as Top3Products

from sales.SalesTerritory t

ORDER BY TerritoryID

FOR JSON PATH;

 

Cosmos DB query

/\*\*Cosmos db SQL API query to find sum of TotalProductQuantity of all territories\*\*/

SELECT sum(sales.TotalProductQuantity) as SumofTotalProductQuantity

FROM shiv JOIN sales in shiv.Top3Products

GROUP BY shiv.Territory

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated