Lab 8: Grouping Sets and Pivoting Data

Retrieving Regional Sales Totals

AdventureWorks sells products to customers in multiple country/regions around the world.

An existing report uses the query provided in the editor to return total sales revenue grouped by country/region and state/province.

Instructions

Modify the query so that the results include a grand total for all sales revenue and a subtotal for each country/region in addition to the state/province subtotals that are already returned. Make sure to use the aliases provided, and default column names elsewhere.

```
SELECT a.CountryRegion, a.StateProvince, SUM(soh.TotalDue) AS Revenue

FROM SalesLT.Address AS a

JOIN SalesLT.CustomerAddress AS ca

ON a.AddressID = ca.AddressID

JOIN SalesLT.Customer AS c

ON ca.CustomerID = c.CustomerID

JOIN SalesLT.SalesOrderHeader as soh

ON c.CustomerID = soh.CustomerID

GROUP BY ROLLUP(a.CountryRegion, a.StateProvince)

ORDER BY a.CountryRegion, a.StateProvince;
```

Retrieving Regional Sales Totals (2)

The solution to the previous exercise has been included on the right, with some additions.

Instructions

- Modify your query to include a column named `Level` that indicates at which level in the total, country/region, and state/province hierarchy the revenue figure in the row is aggregated.

- For example, the grand total row should contain the value 'Total', the row showing the subtotal for United States should contain the value 'United States Subtotal', and the row showing the subtotal for California should contain the value 'California Subtotal'.
- Make sure to use the aliases provided, and default column names elsewhere.

```
SELECT a.CountryRegion, a.StateProvince,

IIF(GROUPING_ID(a.CountryRegion) = 1 AND GROUPING_ID(a.StateProvince) = 1, 'Total',
IIF(GROUPING_ID(a.StateProvince) = 1, a.CountryRegion + ' Subtotal', a.StateProvince + '
Subtotal')) AS Level,

SUM(soh.TotalDue) AS Revenue

FROM SalesLT.Address AS a

JOIN SalesLT.CustomerAddress AS ca

ON a.AddressID = ca.AddressID

JOIN SalesLT.Customer AS c

ON ca.CustomerID = c.CustomerID

JOIN SalesLT.SalesOrderHeader as soh

ON c.CustomerID = soh.CustomerID

GROUP BY ROLLUP(a.CountryRegion, a.StateProvince)

ORDER BY a.CountryRegion, a.StateProvince;
```

Retrieving Regional Sales Totals (3)

Again, the solution to the previous exercise has been provided, so you can take it from there!

Instructions

Extend your query to include a grouping for individual cities. Make sure to use the aliases provided, and default column names elsewhere.

```
SELECT a.CountryRegion, a.StateProvince, a.City,

CHOOSE (1 + GROUPING_ID(a.CountryRegion) + GROUPING_ID(a.StateProvince) +
GROUPING_ID(a.City),

    a.City + ' Subtotal', a.StateProvince + ' Subtotal',

    a.CountryRegion + ' Subtotal', 'Total') AS Level,

SUM(soh.TotalDue) AS Revenue
```

```
FROM SalesLT.Address AS a

JOIN SalesLT.CustomerAddress AS ca

ON a.AddressID = ca.AddressID

JOIN SalesLT.Customer AS c

ON ca.CustomerID = c.CustomerID

JOIN SalesLT.SalesOrderHeader as soh

ON c.CustomerID = soh.CustomerID

GROUP BY ROLLUP(a.CountryRegion, a.StateProvince, a.City)

ORDER BY a.CountryRegion, a.StateProvince, a.City;
```

Retrieving Customer Sales By Category

AdventureWorks products are grouped into categories, which in turn have parent categories (defined in the `SalesLT.vGetAllCategories` view).
AdventureWorks customers are retail companies, and they may place orders for products of any category. The revenue for each product in an order is recorded as the `LineTotal`value in the `SalesLT.SalesOrderDetail` table.

Instructions

Retrieve a list of customer company names together with their total revenue for each parent category in `Accessories`, `Bikes`, `Clothing`, and `Components`. Make sure to use the aliases provided, and default column names elsewhere.

```
SELECT * FROM

(SELECT cat.ParentProductCategoryName, cust.CompanyName, sod.LineTotal

 FROM SalesLT.SalesOrderDetail AS sod

 JOIN SalesLT.SalesOrderHeader AS soh ON sod.SalesOrderID = soh.SalesOrderID

 JOIN SalesLT.Customer AS cust ON soh.CustomerID = cust.CustomerID

 JOIN SalesLT.Product AS prod ON sod.ProductID = prod.ProductID

 JOIN SalesLT.vGetAllCategories AS cat ON prod.ProductcategoryID = cat.ProductCategoryID) AS catsales

PIVOT (SUM(LineTotal) FOR ParentProductCategoryName

IN ([Accessories], [Bikes], [Clothing], [Components])) AS pivotedsales

ORDER BY CompanyName;
```