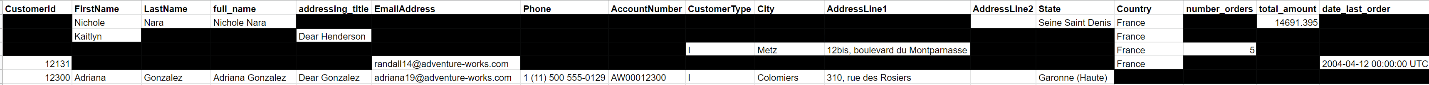
**1.1** You’ve been tasked to create a detailed **overview of all individual customers** (these are defined by customerType = ‘I’ and/or stored in an individual table). Write a query that provides:

* **Identity information** : CustomerId, Firstname, Last Name, FullName (First Name & Last Name).
* **An Extra column** called addressing\_title i.e. (Mr. Achong), if the title is missing - Dear Achong.
* **Contact information** : Email, phone, account number, CustomerType.
* **Location information** : City, State & Country, address.
* **Sales**: number of orders, total amount (with Tax), date of the last order.

Copy only the top 200 rows from your written select ordered by total amount (with tax).

* **Hint:** Few customers have multiple addresses, to avoid duplicate data take their latest available address by choosing max(AddressId)
* Result Hint:



WITH

  sales AS(

  SELECT

    CustomerID,

    ROUND(SUM(TotalDue), 2) AS total\_amount,

    COUNT(SalesOrderID) AS orders\_count,

    DATE(MAX(OrderDate)) AS last\_order\_date

  FROM

    `adwentureworks\_db.salesorderheader`

  GROUP BY

    CustomerID ),

  address\_id AS(

  SELECT

    CustomerID,

    MAX(AddressID) AS AddressID

  FROM

    `adwentureworks\_db.customeraddress`

  GROUP BY

    CustomerID )

SELECT

  individual.CustomerID AS customer\_id,

  contact.Firstname AS first\_name,

  contact.LastName AS last\_name,

  contact.Firstname || "" "" || contact.LastName AS full\_name,

  IFNULL(contact.Title, 'Dear') || ' ' || contact.LastName AS addressing\_title,

  contact.EmailAddress AS email,

  contact.phone,

  customer.AccountNumber AS account\_number,

  customer.CustomerType AS customer\_type,

  address.city,

  state.name AS state,

  country.name AS country,

  CONCAT(COALESCE(address.AddressLine1, ''), COALESCE((' ' || address.AddressLine2), '')) AS address\_line,

  sales.orders\_count,

  sales.total\_amount,

  sales.last\_order\_date

FROM

  `adwentureworks\_db.individual` individual

LEFT JOIN

  `adwentureworks\_db.contact` contact

ON

  individual.ContactID = contact.ContactID

LEFT JOIN

  `adwentureworks\_db.customer` customer

ON

  individual.CustomerID = customer.CustomerID

LEFT JOIN

  address\_id

ON

  individual.CustomerID = address\_id.CustomerID

LEFT JOIN

  `adwentureworks\_db.address` address

ON

  address\_id.AddressID = address.AddressID

LEFT JOIN

  `adwentureworks\_db.stateprovince` state

ON

  address.StateProvinceID = state.StateProvinceID

LEFT JOIN

  `adwentureworks\_db.countryregion` country

ON

  state.CountryRegionCode = country.CountryRegionCode

LEFT JOIN

  sales

ON

  individual.CustomerID = sales.CustomerID

ORDER BY

  sales.total\_amount DESC

LIMIT

  200

**1.2** Business finds the original query valuable to analyze customers and now want to get the data from the first query for the **top 200 customers with the highest total amount (with tax) who have not ordered for the last 365 days**. How would you identify this segment?

Hints:

* You can use temp table, cte and/or subquery of the 1.1 select.
* Note that the database is old and the current date should be defined by finding the latest order date in the orders table.

WITH

  sales AS(

  SELECT

    CustomerID,

    ROUND(SUM(TotalDue), 2) AS total\_amount,

    COUNT(SalesOrderID) AS orders\_count,

    DATE(MAX(OrderDate)) AS last\_order\_date,

    DATE\_DIFF( (

      SELECT

        MAX(DATE(OrderDate))

      FROM

        `adwentureworks\_db.salesorderheader` ), DATE(MAX(OrderDate)), DAY) AS diff\_from\_current\_date

  FROM

    `adwentureworks\_db.salesorderheader`

  GROUP BY

    CustomerID ),

  address\_id AS(

  SELECT

    CustomerID,

    MAX(AddressID) AS AddressID

  FROM

    `adwentureworks\_db.customeraddress`

  GROUP BY

    CustomerID )

SELECT

  individual.CustomerID AS customer\_id,

  contact.Firstname AS first\_name,

  contact.LastName AS last\_name,

  contact.Firstname || "" "" || contact.LastName AS full\_name,

  IFNULL(contact.Title, 'Dear') || ' ' || contact.LastName AS addressing\_title,

  contact.EmailAddress AS email,

  contact.phone,

  customer.AccountNumber AS account\_number,

  customer.CustomerType AS customer\_type,

  address.city,

  state.name AS state,

  country.name AS country,

  CONCAT(COALESCE(address.AddressLine1, ''), COALESCE((' ' || address.AddressLine2), '')) AS address\_line,

  sales.orders\_count,

  sales.total\_amount,

  sales.last\_order\_date

FROM

  `adwentureworks\_db.individual` individual

LEFT JOIN

  `adwentureworks\_db.contact` contact

ON

  individual.ContactID = contact.ContactID

LEFT JOIN

  `adwentureworks\_db.customer` customer

ON

  individual.CustomerID = customer.CustomerID

LEFT JOIN

  address\_id

ON

  individual.CustomerID = address\_id.CustomerID

LEFT JOIN

  `adwentureworks\_db.address` address

ON

  address\_id.AddressID = address.AddressID

LEFT JOIN

  `adwentureworks\_db.stateprovince` state

ON

  address.StateProvinceID = state.StateProvinceID

LEFT JOIN

  `adwentureworks\_db.countryregion` country

ON

  state.CountryRegionCode = country.CountryRegionCode

LEFT JOIN

  sales

ON

  individual.CustomerID = sales.CustomerID

WHERE

  diff\_from\_current\_date >= 365 -- more

  OR equal TO because the CURRENT date IS included TO the count AS well

ORDER BY

  sales.total\_amount DESC

LIMIT

  200

**1.3** Enrich your original 1.1 SELECT by creating a **new column in the view that marks active & inactive customers based on whether they have ordered anything during the last 365 days**.

* Copy only the top 500 rows from your written select ordered by CustomerId desc.

WITH

  sales AS(

  SELECT

    CustomerID,

    ROUND(SUM(TotalDue), 2) AS total\_amount,

    COUNT(SalesOrderID) AS orders\_count,

    DATE(MAX(OrderDate)) AS last\_order\_date,

    DATE\_DIFF( (

      SELECT

        MAX(DATE(OrderDate))

      FROM

        `adwentureworks\_db.salesorderheader` ), DATE(MAX(OrderDate)), DAY) AS diff\_from\_current\_date

  FROM

    `adwentureworks\_db.salesorderheader`

  GROUP BY

    CustomerID ),

  address\_id AS(

  SELECT

    CustomerID,

    MAX(AddressID) AS AddressID

  FROM

    `adwentureworks\_db.customeraddress`

  GROUP BY

    CustomerID )

SELECT

  individual.CustomerID AS customer\_id,

  contact.Firstname AS first\_name,

  contact.LastName AS last\_name,

  contact.Firstname || "" "" || contact.LastName AS full\_name,

  IFNULL(contact.Title, 'Dear') || ' ' || contact.LastName AS addressing\_title,

  contact.EmailAddress AS email,

  contact.phone,

  customer.AccountNumber AS account\_number,

  customer.CustomerType AS customer\_type,

  address.city,

  state.name AS state,

  country.name AS country,

  CONCAT(COALESCE(address.AddressLine1, ''), COALESCE((' ' || address.AddressLine2), '')) AS address\_line,

  sales.orders\_count,

  sales.total\_amount,

  sales.last\_order\_date,

  CASE

    WHEN sales.diff\_from\_current\_date >= 365 THEN 'Inactive'

    ELSE 'Active'

END

  AS customer\_status

FROM

  `adwentureworks\_db.individual` individual

LEFT JOIN

  `adwentureworks\_db.contact` contact

ON

  individual.ContactID = contact.ContactID

LEFT JOIN

  `adwentureworks\_db.customer` customer

ON

  individual.CustomerID = customer.CustomerID

LEFT JOIN

  address\_id

ON

  individual.CustomerID = address\_id.CustomerID

LEFT JOIN

  `adwentureworks\_db.address` address

ON

  address\_id.AddressID = address.AddressID

LEFT JOIN

  `adwentureworks\_db.stateprovince` state

ON

  address.StateProvinceID = state.StateProvinceID

LEFT JOIN

  `adwentureworks\_db.countryregion` country

ON

  state.CountryRegionCode = country.CountryRegionCode

LEFT JOIN

  sales

ON

  individual.CustomerID = sales.CustomerID

ORDER BY

  customer\_id DESC

LIMIT

  500

**1.4** Business would like to extract data on **all active customers from North America. Only customers that have either ordered no less than 2500 in total amount (with Tax) or ordered 5 + times** should be presented.

In the output for these customers divide their address line into two columns, i.e.:

| **AddressLine1** | **address\_no** | **Address\_st** |
| --- | --- | --- |
| '8603 Elmhurst Lane' | 8603 | Elmhurst Lane |

Order the output by country, state and date\_last\_order.

WITH

  sales AS(

    SELECT

      CustomerID,

      ROUND(SUM(TotalDue), 2) AS total\_amount,

      COUNT(SalesOrderID) AS orders\_count,

      DATE(MAX(OrderDate)) AS last\_order\_date,

      DATE\_DIFF(

        (

        SELECT

          MAX(DATE(OrderDate))

        FROM

          `adwentureworks\_db.salesorderheader`

        ),

        DATE(MAX(OrderDate)), DAY) AS diff\_from\_current\_date

    FROM

      `adwentureworks\_db.salesorderheader`

    GROUP BY

      CustomerID

  ),

  address\_id AS(

  SELECT

    CustomerID,

    MAX(AddressID) AS AddressID

  FROM

    `adwentureworks\_db.customeraddress`

  GROUP BY

    CustomerID

  ),

  main\_table AS(

  SELECT

    individual.CustomerID AS customer\_id,

    contact.Firstname AS first\_name,

    contact.LastName AS last\_name,

    contact.Firstname || "" "" || contact.LastName AS full\_name,

    IFNULL(contact.Title, 'Dear') || ' ' || contact.LastName AS addressing\_title,

    contact.EmailAddress AS email,

    contact.phone,

    customer.AccountNumber AS account\_number,

    customer.CustomerType AS customer\_type,

    address.city,

    state.name AS state,

    country.name AS country,

    address.AddressLine1 AS address\_line1,

    address.AddressLine2 AS address\_line2,

    sales.orders\_count,

    sales.total\_amount,

    sales.last\_order\_date,

    CASE

      WHEN sales.diff\_from\_current\_date >= 365 THEN 'Inactive'

      ELSE 'Active'

    END AS customer\_status

  FROM

    `adwentureworks\_db.individual` individual

  LEFT JOIN

    `adwentureworks\_db.contact` contact

  ON

    individual.ContactID = contact.ContactID

  LEFT JOIN

    `adwentureworks\_db.customer` customer

  ON

    individual.CustomerID = customer.CustomerID

  LEFT JOIN

    address\_id

  ON

    individual.CustomerID = address\_id.CustomerID

  LEFT JOIN

    `adwentureworks\_db.address` address

  ON

    address\_id.AddressID = address.AddressID

  LEFT JOIN

    `adwentureworks\_db.stateprovince` state

  ON

    address.StateProvinceID = state.StateProvinceID

  LEFT JOIN

    `adwentureworks\_db.countryregion` country

  ON

    state.CountryRegionCode = country.CountryRegionCode

  LEFT JOIN

    sales

  ON

    individual.CustomerID = sales.CustomerID

  )

SELECT

  \*,

  CASE

    WHEN address\_line1 LIKE '%Box %' THEN 'Not provided' -- if instead of address, a post office box information is provided, it is not applicable to split it to address number and street

    ELSE CAST(REPLACE(LEFT(address\_line1, STRPOS(address\_line1, ' ') - 1), ',', '') AS STRING) -- removing any commas from the address\_no with REPLACE

  END AS address\_no,

  CASE

    WHEN address\_line1 LIKE '%Box %' THEN 'Not provided'

    ELSE SUBSTR(address\_line1, STRPOS(address\_line1, ' ') + 1)

  END AS address\_str

FROM main\_table

WHERE

  customer\_status ='Active' AND

  country IN ('United States', 'Canada') AND -- checked that there are 6 countries in the main table (SELECT country FROM main\_table GROUP BY 1), and two of them are in North America

  (total\_amount >= 2500 OR

  orders\_count >= 5)

ORDER BY

  country,

  state,

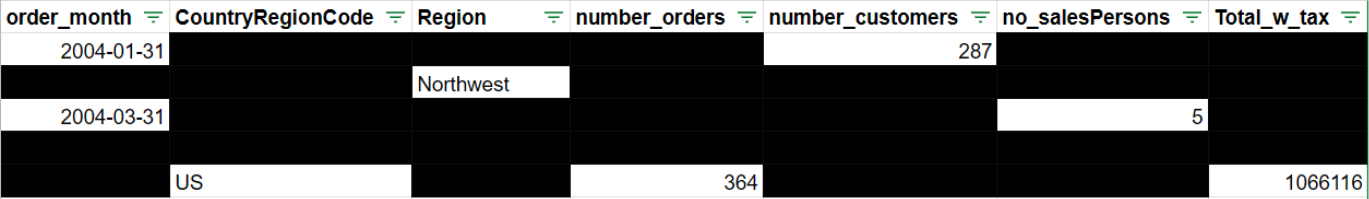
  last\_order\_date

**2. Reporting Sales’ numbers**

* **Main tables to start from:** salesorderheader.

**2.1** Create a query of monthly sales numbers in each Country & region. Include in the query a number of orders, customers and sales persons in each month with a total amount with tax earned. Sales numbers from all types of customers are required.

* Result Hint:



SELECT

  LAST\_DAY(DATE(soh.OrderDate), MONTH) AS order\_month,

  s.CountryRegionCode,

  s.Name AS Region,

  COUNT(DISTINCT soh.SalesOrderID) AS number\_orders,

  COUNT(DISTINCT soh.CustomerID) AS number\_customers,

  COUNT(DISTINCT soh.SalesPersonID) AS no\_salespersons,

  ROUND(SUM(soh.TotalDue), 0) AS total\_w\_tax

FROM

  `tc-da-1.adwentureworks\_db.salesorderheader`soh

LEFT JOIN

  `tc-da-1.adwentureworks\_db.salesterritory`s

USING

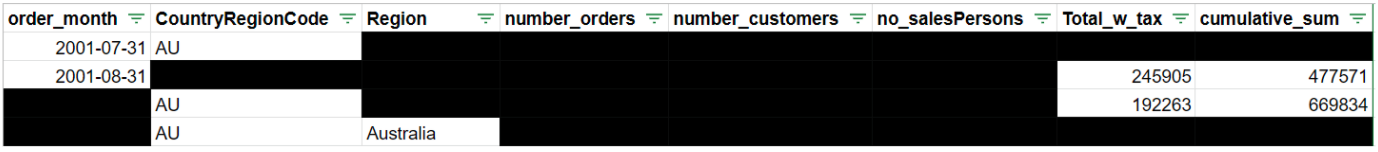
  (TerritoryID)

GROUP BY

  ALL;

**2.2** Enrich 2.1 query with the cumulative\_sum of the total amount with tax earned per country & region.

* Hint: use CTE or subquery.
* Result Hint:



WITH

  MonthlySales AS (

  SELECT

    LAST\_DAY(DATE(soh.OrderDate), MONTH) AS order\_month,

    s.CountryRegionCode,

    s.Name AS Region,

    COUNT(DISTINCT soh.SalesOrderID) AS number\_orders,

    COUNT(DISTINCT soh.CustomerID) AS number\_customers,

    COUNT(DISTINCT soh.SalesPersonID) AS no\_salespersons,

    ROUND(SUM(soh.TotalDue), 0) AS total\_w\_tax

  FROM

    `tc-da-1.adwentureworks\_db.salesorderheader`soh

  LEFT JOIN

    `tc-da-1.adwentureworks\_db.salesterritory`s

  USING

    (TerritoryID)

  GROUP BY

    ALL )

SELECT

  \*,

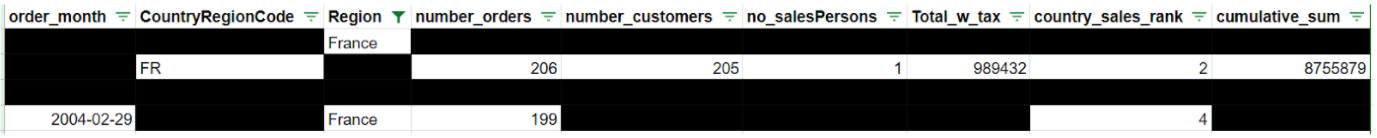
  SUM(total\_w\_tax) OVER (PARTITION BY CountryRegionCode, Region ORDER BY order\_month) AS cumulative\_sum

FROM

  MonthlySales;

**2.3** Enrich 2.2 query by adding ‘sales\_rank’ column that ranks rows from best to worst for each country based on total amount with tax earned each month. I.e. the month where the (US, Southwest) region made the highest total amount with tax earned will be ranked 1 for that region and vice versa.

* Result Hint (with region filtered on France):



WITH

  MonthlySales AS (

  SELECT

    LAST\_DAY(DATE(soh.OrderDate), MONTH) AS order\_month,

    s.CountryRegionCode,

    s.Name AS Region,

    COUNT(DISTINCT soh.SalesOrderID) AS number\_orders,

    COUNT(DISTINCT soh.CustomerID) AS number\_customers,

    COUNT(DISTINCT soh.SalesPersonID) AS no\_salespersons,

    ROUND(SUM(soh.TotalDue), 0) AS total\_w\_tax

  FROM

    `tc-da-1.adwentureworks\_db.salesorderheader` soh

  LEFT JOIN

    `tc-da-1.adwentureworks\_db.salesterritory` s

  USING

    (TerritoryID)

  GROUP BY

    ALL )

SELECT

  \*,

  SUM(total\_w\_tax) OVER (PARTITION BY CountryRegionCode, Region ORDER BY order\_month) AS CumulativeTotal,

  RANK() OVER (PARTITION BY CountryRegionCode, Region ORDER BY total\_w\_tax DESC) AS country\_sales\_rank

FROM

  MonthlySales;

**2.4** Enrich 2.3 query by adding taxes on a country level:

* As taxes can vary in country based on province, the needed column is ‘**mean\_tax\_rate**’ -> average tax rate in a country.
* Also, as not all regions have data on taxes, you also want to be transparent and show the **‘perc\_provinces\_w\_tax’** -> a column representing the percentage of provinces with available tax rates for each country (i.e. If US has 53 provinces, and 10 of them have tax rates, then for US it should show 0,19)
* **Hint:** If a state has multiple tax rates, choose the higher one. Do not double count a state in country average rate calculation if it has multiple tax rates.
* **Hint:** Ignore the isonlystateprovinceFlag rate mechanic, it is beyond the scope of this exercise. Treat all tax rates as equal.
* Result Hint (with region filtered on US):



WITH

  sales\_data AS(

  SELECT

    LAST\_DAY(DATE(sales.OrderDate), MONTH) AS order\_month,

    territory.CountryRegionCode AS country\_code,

    territory.Name AS region,

    COUNT(sales.SalesOrderID) AS order\_count,

    COUNT(DISTINCT sales.CustomerID) AS customer\_count,

    COUNT(DISTINCT sales.SalesPersonID) AS salesperson\_count,

    ROUND(SUM(sales.TotalDue), 0) AS total\_sales\_with\_tax

  FROM

    `adwentureworks\_db.salesorderheader` sales

  LEFT JOIN

    `adwentureworks\_db.salesterritory` territory

  USING

    (TerritoryID)

  GROUP BY

    ALL ),

  state\_tax\_data AS(

  SELECT

    state\_province.CountryRegionCode AS country\_code,

    state\_province.StateProvinceID AS state\_id,

    MAX(tax.TaxRate) AS tax\_rate

  FROM

    `adwentureworks\_db.stateprovince` state\_province

  LEFT JOIN

    `adwentureworks\_db.salestaxrate` tax

  USING

    (StateProvinceID)

  GROUP BY

    ALL ),

  tax\_stats AS(

  SELECT

    country\_code,

    ROUND(AVG(tax\_rate), 2) AS mean\_tax\_rate,

    ROUND(COUNT(tax\_rate)/COUNT(state\_id), 2) AS perc\_provinces\_w\_tax

  FROM

    state\_tax\_data

  GROUP BY

    ALL )

SELECT

  sales\_data.\*,

  SUM(total\_sales\_with\_tax) OVER (PARTITION BY country\_code, region ORDER BY order\_month) AS cumulative\_sales\_with\_tax,

  RANK() OVER (PARTITION BY country\_code, region ORDER BY total\_sales\_with\_tax DESC) AS sales\_rank,

  tax\_stats.mean\_tax\_rate,

  tax\_stats.perc\_provinces\_w\_tax

FROM

  sales\_data

LEFT JOIN

  tax\_stats

USING

  (country\_code)

ORDER BY

  mean\_tax\_rate DESC