

OMSCS CS6400 - Spring 2019

Project Phase 1 – Report

Team 051

[Table of Contents:](#)

S&E Data Warehouse Data Type

[Data Types](#)

S&E Data Warehouse Constraints

[Business Logic Constraints](#)

Task Decomposition with Abstract Code:

[Main Menu/Dashboard](#)

[Holidays](#)

[Managers](#)

[Cities](#)

[Manufacturer' Product Report](#)

[Category Report](#)

[Actual versus Predicted Revenue for GPS units](#)

[Store Revenue](#)

[Air Conditioners on Groundhog Day](#)

[State with Highest Volume for each Category](#)

[Revenue by Population](#)

Data Types:

Store

Attribute	Data Type	Nullable
Store number	string	Not Null
Phone number	string	Not Null
Street address	string	Not Null

Manager

Attribute	Data Type	Nullable
First Name	string	Not Null
Middle Name	string	Not Null
Last Name	string	Not Null
Email address	string	Not Null
If active	Boolean	Not Null

City

Attribute	Data Type	Nullable
Name	string	Not Null
State	string	Not Null
Population	integer	Not Null

Manufacturer

Attribute	Data Type	Nullable
Name	string	Not Null
Maximum discount	float	Null

Product

Attribute	Data Type	Nullable
PID	integer	Not Null
Name	string	Not Null
Retail price	float	Not Null

Category

Attribute	Data Type	Nullable
Name	string	Not Null

Date

Attribute	Data Type	Nullable
Year	string	Not Null
Month	string	Not Null
Day	string	Not Null

Holiday

Attribute	Data Type	Nullable
Name	string	Not Null

Sale price

Attribute	Data Type	Nullable
Sale price	float	Not Null

Sold quantity

Attribute	Data Type	Nullable
Sold quantity	integer	Not Null

Business Logic Constraints

Store

1. A store may have more than one manager.
2. Some stores may not have a manager assigned to them.

Manager

1. A manager may manage more than one store.
2. Managers may become inactive if they stop being an S&E employee.
3. All active managers must be assigned to a store.

City

1. Multiple stores may locate in the same city.

Product

1. Products have a numeric unique identifier (PID).
2. Every product has a retail *price*. The retail price is in effect unless there is a sale.
3. The same product could go on sale multiple times with different sale prices.
4. Product sale price must be lower than retail price.
5. If a maximum discount is specified by the manufacturer, then no product can be placed on sale for less than (1-maximum discount) of the retail price.
6. If a maximum discount is not specified by the manufacturer, no product can be discounted more than 90% of retail.

Category

1. Each category has a unique name.
2. Every product must be in at least one category.

Manufacturer

1. Each product is related to a single manufacturer.
2. Multiple products could be made by the same manufacturer.
3. Each product is assigned one or more categories.

Main Menu/Dashboard

Task Decomp

Lock Types: Lookup Store, Manufacturer, Product and Manager records, all are Read-only.

Number of Locks: four

Enabling Conditions: None

Frequency: About 50 times per day.

Consistency (ACID): not critical, order is not critical.

Subtasks: Mother Task is not needed. No decomposition needed.

Abstract code:

- **View statistics information**
- Show “**View Manufacturer’s Product Report**”, “**View Category Report**”, “**View Actual versus Predicted Revenue Report**”, “**View Store Revenue Report**”, “**View Air Conditioner Sold on Groundhog Day**”, “**View Sate with Highest Volume for each Category**”, “**View Revenue by Population**”, “**Managers**”, “**Holidays**” and “**Cities**” tabs
- User click on **View Manufacturer’s Product Report** button in **Main Menu** - Jump to the **Product Report 1** task
- User click **View Category Report** button in **Main Menu** - Jump to the **Product Report 2** task
- User click **View Actual versus Predicted Revenue Report** button- Jump to the **Produce Report 3** task
- User click **View Store Revenue Report** button- Jump to the **Produce Report 4** task
- User click **View Air Conditioner Sold on Groundhog Day** button- Jump to the **Produce Report 5** task
- User click **View Sate with Highest Volume for each Category** button- Jump to the **Produce Report 6** task
- User click **View Revenue by Population** button- Jump to the **Produce Report 7** task
- User click “**View Managers**” button, jump to **View/List Managers**
- User click “**Holidays**” button, jump to **View Holiday Information**
- User click “**Cities**” button, jump to **View City Infomation**

Task: View statistics information

Lock Types: Lookup Store, Manufacturer, Product and Manager records, all are Read-only.

Number of Locks: Four schema constructs are involved

Enabling Conditions: None

Frequency: About 50 times per day (displayed on **Main Menu**).

Consistency (ACID): not critical, order is not critical

Subtasks: Mother Task is not needed. No decomposition needed.



View Statistics
Information

Abstract code:

- Retrieve STORE record; count and display “the count of stores”
- Retrieve MANUFACTURER record; count and display “the count of manufacturers”
- Retrieve PRODUCT record; count and display “the count of products”
- Retrieve MANAGER record; count and display “the count of managers”

Holidays

Task Decomp

Task: View Holiday Information

Lock Types: Read on Holiday record.


Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency event.

Consistency (ACID): not critical, order is not critical

Subtasks: None



View Holiday
Information

Abstract code:

- User clicks on **Holidays** Button in the Main Menu
- Retrieve and display list of all holiday information:
 - Holiday name
 - Date

Task: Add Holiday Information

Lock Types: Read on Holiday record, read/write on Holiday record.

Number of Locks: Single


Enabling Conditions: None

Frequency: low frequency event.

Consistency (ACID): not critical, original holiday information could be viewed if the holiday is being edit

Supertask: None

Subtasks: None



Add Holiday
Information

Abstract code:

- User clicks **Add Holiday** Button in the Holidays form
- Display additional room for another holiday, including two input fields: *name* and *date*
- User input holiday name and date
- User click **Save** button
 - If user input a name that already exists in original list:
 - Pops error message "Holiday has existed"
 - If user input date already exist in original list:
 - Pops error message "Holiday has existed"
 - Else: insert a record into the Holiday table

Managers

Task Decomp

Task: View/List Managers

Lock Types: Read only on Manager record.

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency.

Consistency (ACID): not critical, it is okay if Manager records are updated while being viewed by other users

Supertask: None

Subtasks: None



Abstract code:

- User clicks **Managers** Button in the **Main Menu**
- Sort and display list of all managers by last name ascending
 - For each manager record, display:
 - Manager name
 - Manager E-mail
 - If active
 - Store Number of the store he/she is assigned

Task: Edit Managers

Lock Types: update/insert/delete on the Manager records

Number of Locks: Single

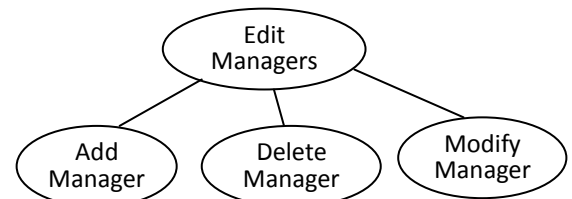
Enabling Conditions: None

Frequency: low frequency

Consistency (ACID): not critical, it is okay if Manager records are updated while being viewed by other users

Supertask: None

Subtasks: Add Manager, Modify Manager, Delete Manager



Abstract code:

- User click **Edit Managers** button in **Managers**
- View/List Managers, Populate Manager name dropdown
- When a button is clicked, then do the following:
 - If **Add Manager** button is clicked: **Add Manager; View/List Managers**
 - If **Delete Manager** button is clicked; **Delete Manager; View/List Managers**
 - If **Modify Manager** button is clicked; **Modify Manager; View/List Managers**
 - If **Cancel** button is clicked; go to **View/List Managers** for current records

Task: Add Manager

Lock Types: insert on the Manager records

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency

Consistency (ACID): not critical, original manager records can be viewed if new manager is being added.



Supertask: Edit Managers

Subtasks: None

Abstract code:

- User click **Add Manager** button in Edit Managers
- Display additional room for another manager, which contains:
 - Name* input field
 - Email* input field
 - A dropdown menu option for *Store Number*
- User enter manager name, email-address, and store number assigned
- User click **Save** button, the record is insert in the database

Task: Delete Manager

Lock Types: delete on the Manager records

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency

Consistency (ACID): not critical, original manager records can be viewed if new manage is being deleted.

Supertask: Edit Managers

Subtasks: None

Delete
Manager

Abstract code:

- User click **Delete Manager** button in Edit Managers
- If the to-be-delete manger is currently assigned to one or more stores:
 - Pops error message "A manger has to be unassigned from all stores before deleted"
- Else: The specific manager record will be deleted from the database

Task: Modify Manager

Lock Types: update on the Manager records

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency

Consistency (ACID): not critical, original manager records can be viewed if new manage is being edited.

Supertask: Edit Managers

Subtasks: None

Modify
Manager

Abstract code:

- User click **Modify Manager** button in Edit Managers
- User may change manager's email address, active status, delete or add assigned store number
- User click **Save** button, the record is updated in the database

Cities

Task Decomp

Task: View city information

Lock Types: Read on City record.

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency event.

Consistency (ACID): not critical, order is not critical

Supertask: None

Subtasks: None

View city
Information

Abstract code:

- User clicks ***Cities*** Button in the **Main Menu**
- Retrieve and display list of all city information:
 - City name
 - State
 - Population
- Adjacent to each row in form is *Update* Buttons

Task: Update city population

Lock Types: Read/write on City record.

Number of Locks: Single

Enabling Conditions: None

Frequency: low frequency event.

Consistency (ACID): not critical, order is not critical

Supertask: None

Subtasks: None

Update city
population

Abstract code:

- User clicks Update Button on the **Cities** form
 - City population field entry change to blank.
 - User enters population into the population field on form
 - User click Save button, update the record in database
-

Manufacturer' Product Report

Task Decomp

Task: Produce Report 1 (Manufacturer's Product Report)

Lock Types: Lookup Manufacturer, Product and Sale Price record, all are Read-only.

Number of Locks: 3

Enabling Conditions: None

Frequency: Almost same frequency as main menu.

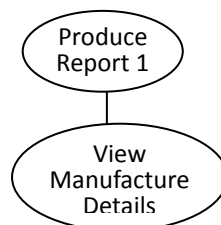
Consistency (ACID): not critical, order is not critical.

Supertask: client query from Main Menu

Subtasks: lookup Manufacture Detail information upon specific manufacturer selectd by client

Abstract code:

- User clicked on **View Manufacturer's Product Report** button from **Main Menu**
- Retrieve a complete list of all manufacturer records with their corresponding Product records
- For each manufacturer, retrieve the manufacturer's name, **COUNT** the total number of the products offered by the manufacturer; find the **average** retail price of all the manufacturer's products, the **maximum** retail price and the **minimum** retail price.
- Sort the list of manufacturer entries by average retail price descending
- Display Manufacturer's Product Report, for each manufacturer, display:
 - Manufacturer's name
 - Total number of products offered by the manufacturer
 - Average retail price
 - Minimum retail price
 - Maximum retail price
 - Upon click View Details, execute the **VIEW Manufacturer Details** task



Task: View Manufacture Details

Lock Types: Lookup Manufacturer, Product and Sale Price record, all are Read-only.

Number of Locks: Three

Enabling Conditions: None

Frequency: User Detail and Menu Options have the same frequency.

Consistency (ACID): not critical, order is not critical.

Supertask: client query from Main Menu

Abstract code:

- User clicked on the **Update Buttons** button in manufacturer table in **Report 1**
- For the specific manufacturer selected by the user, retrieve and display manufacturer name, and maximum discount of its products, the total number of the products offered by the manufacturer; average retail price, the maximum retail price and the minimum

retail price in the header

- Sort the list of the products offered by the manufacturer by retail price descending
- For each product offered by the manufacturer, display:
 - Product ID
 - Name
 - Retail price
 - Category (concatenated)

Category Report

Task Decomp

Produce
Report 2

Task: Produce Report 2 (Category Report)

Lock Types: Lookup Manufacturer, Product and Sale Price record, all are Read-only.

Number of Locks: Three

Enabling Conditions: None

Frequency: Almost same frequency as main menu

Consistency (ACID): not critical, order is not critical.

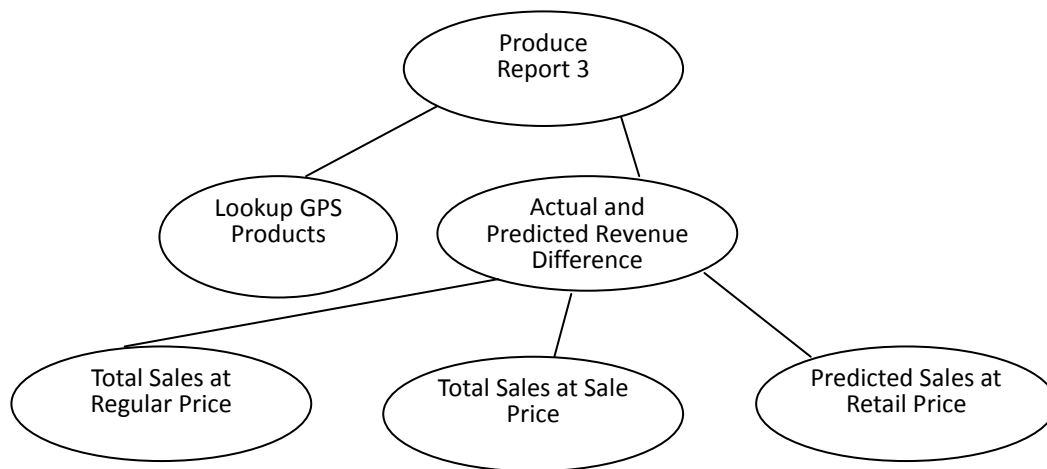
Supertask: client query from Main Menu

Abstract code:

- User clicked on **View Category Report** button from **Main Menu**
- For each category:
 - Retrieve and display category name
 - Count and display the total number of products in the category
 - Retrieve and display average retail price of all the products in the category
 - Find and display total number of unique manufacturers offering products in the category

Actual versus Predicted Revenue for GPS units

Task Decomp



Task: Produce Report 3

Lock Types: Read-only lookups of Category Name, Product Name and PID, Product Sold Date, Product Sold Quantity, Product Retail Price and Product Sale Price.

Number of Locks: Several different schema constructs are needed.

Enabling Conditions: None.

Frequency: Almost same frequency as main menu.

Consistency (ACID): is not critical, even if the product information is being edited by the user while a friend is looking at it.

Subtasks: none

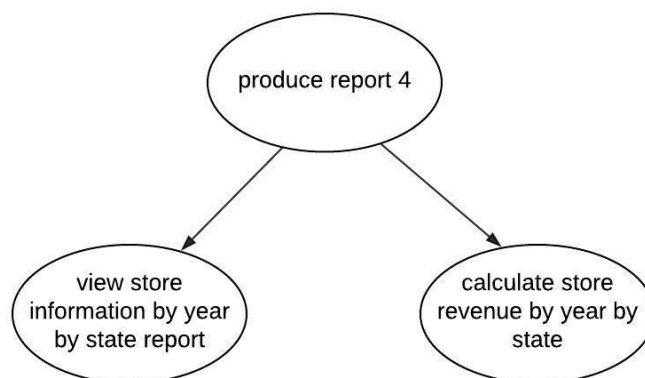
Abstract code:

- User clicked on **Actual versus Predicted Revenue** button after selecting **GPS Products** drop-down filter from **Main Menu**:
- Run the **Actual versus Predicted Revenue** task: query for the difference between the actual revenue and the predicted revenue.
- Find all Products belonging to Category Name of GPS.
- For each Product in GPS Category:
 - Display Product ID (PID), Product Name, Retail Price
 - Find all Dates when Product was sold in Retail Price.
 - Find Retail Quantity sold on Dates of Retail Price.
 - Calculate Retail Revenue of Product sold in Retail Price by multiplying Retail Price and Retail Quantity.
 - Find all Dates when Product was sold in Sale Price.
 - Find Sale Quantity sold on Dates of Sale Price.
 - Calculate Sale Revenue of Product sold in Sale Price by multiplying Sale Price and Sale Quantity.
 - Update Sale Price to Retail Price for transactions occurred on Sale Dates.
 - Update Sale Quantity to Predicted Quantity by formula: Predicted Quantity = Sale Quantity * 0.75
 - Calculate Predicted Revenue of Product sold in Sale Price by multiplying

- Retail Price and Predicted Quantity.
 - Calculate the total number of GPS products sold at Retail Price by summing up Retail Quantity.
 - Calculate the total number of GPS products sold at Sale Price by summing up Sale Quantity
 - Calculate the total number of GPS products sold by adding up the above two total numbers.
 - Display total number of units ever sold, the total number of units sold at Sale Price, the total number of units sold at Retail Price.
 - Sum up Retail Revenue of all Retail Products sold in Retail Price.
 - Sum up Sale Revenue of all Sale Products sold in Sale Price.
 - Sum up Predicted Revenue of all Sale Products if sold in Retail Price
 - Calculate Total Actual Revenue of GPS Products by adding up Retail Revenue and Sale Revenue.
 - Calculate Total Predicted Revenue of GPS Products by adding up Retail Revenue and Predicted Revenue.
 - Calculate the difference between total Actual Revenue and total Predicted Revenue.
- For each Product in GPS Category:
 - Display Total Actual Revenue, Total Predicted Revenue and their difference if the difference is larger than \$5000 (positive or negative) with the difference sorted in descending order.

Store Revenue

Task Decomp



Task: Produce Report 4

Lock Types: read only

Number of Locks: Several different schema constructs are needed

Enabling Conditions: None

Frequency: Almost same frequency as main menu

Consistency (ACID): not critical

Subtasks: None

Abstract code:

- User clicks **View Store Revenue Report** button
- For each distinct state in the drop down box:
- Find all the cities related to the state, display the city name.
- For each store related to the cities, display the store number, street address.
- Find the quantity of product sold with retail price and the quantity of product on sale.
- Find all the sold products, Display the year of the date that the products are sold.
- Find the quantity of product sold with retail price and the quantity of product on sale.
- Calculate revenue of each product in the store by:
- Each product revenue= *quantity* of product sold with retail price X retail price + *quantity* of product on sale X sale price.
- Sum up all the product revenue in each store to get the total revenue of each store.
- The report is ordered by year, then by revenue in descending order.

Air Conditioners on Groundhog Day

Task decomp

Task: Produce Report 5

Lock Types: Read-only look up on date, product

Number of Locks: Single

Enabling Conditions: None

Frequency: Almost same frequency as main menu

Consistency (ACID): not critical, order is not critical.

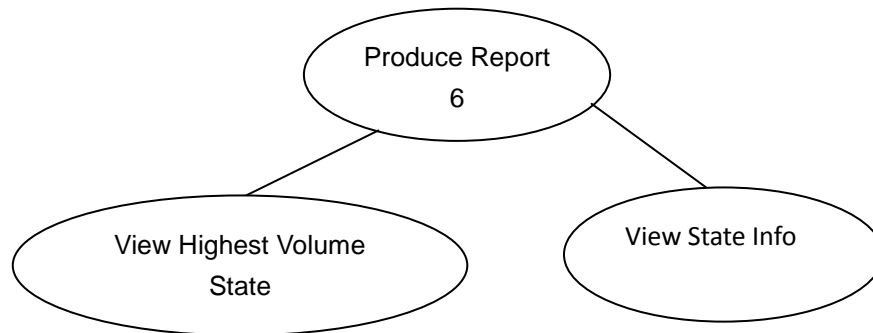
Subtasks: None

Abstract Code

- User click **View Air Conditioner Sold on Groundhog Day** button on **Main Menu**
- Filter products to leave those in Air Conditioner Category
- Group the sold information by year, sum up all the quantities to get the total number of items sold that year.
 - Get average number of units sold per day by divide the total number by 365.
- Group the sold information by each year's Groundhog Day, sum up all the quantities to get total number of units sold at that day.
- Join those two information above by year.
 - Sort on the year in ascending order

State with Highest Volume for each Category

Task Decomp



Task: Produce Report 6

Lock Types: Read-only look up on date, category, product, city, store, manager.

Number of Locks: Several different schema constructs are needed

Enabling Conditions: None

Frequency: Almost same frequency as main menu

Consistency (ACID): not critical, order is not critical.

Subtasks: View Highest Volume State, View State Info

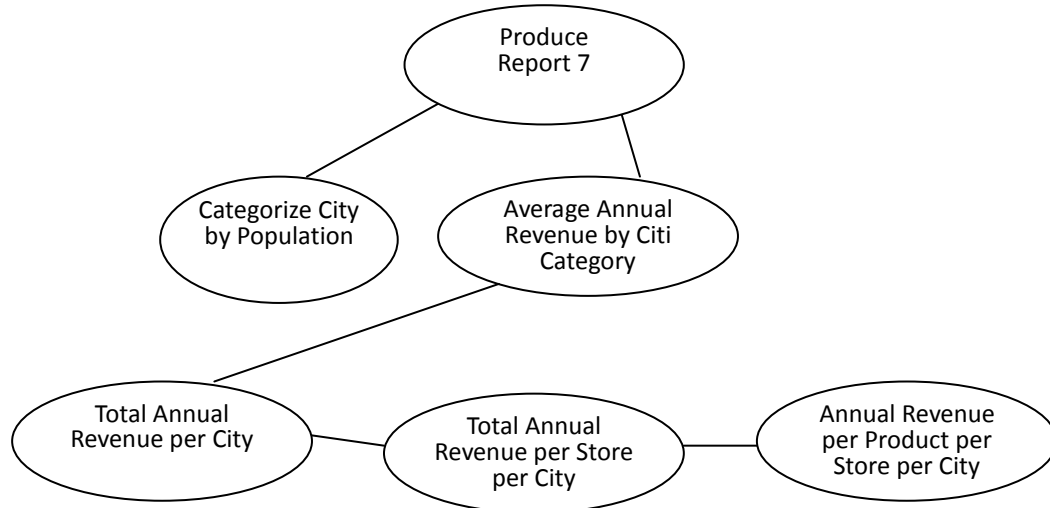
Abstract Code

- User click ***View Sate with Highest Volume for each Category*** button on **Main Menu**
- User input *Year* and *Month* into input fields and click *look up* button.
- Do **View Highest Volume State task**:
 - Filter sold date with user input's year and month.
 - Group the product by category, and for each category:
 - Group the store by state, sum up the quantity to get total sale volume for that sate
 - Find the state with the maximum volume, return the state name and the volume
 - Sort by category name in ascending order.
 - Attach a hyperlink to each *row* of the main report
 - Upon user click a *row*:
 - Do **View State Info**:
 - Get the category, year/month, state information from the main menu
 - Find all store in the that state
 - For each store:
 - Find city and managers belongs to this store.
 - Return store ID, address, city, and every unique manager as a row
 - Sort the store by ID in ascending order.

- Display the date with the header (category, year/month, state)

Revenue by Population

Task Decomp



Task: Produce Report 7

Lock Types: Read-only lookups of City Population, Store Number, Product ID, Product Sold Date, Product Sold Quantity, Product Retail Price and Product Sale Price.

Number of Locks: Several different schema constructs are needed.

Enabling Conditions: None.

Frequency: Almost same frequency as main menu

Consistency (ACID): Consistency is not critical, even if City Population is being edited while a user is looking up the Revenue by Population.

Subtasks: None.

Abstract Code

- User clicked on **Revenue by Population** button from **Main Menu**:
- Run the **Revenue by Population** task: query for information about average annual revenue for specific city population categories.
 - Find all unique Cities available in database by looking up both City Name and Store Number (ID) of Stores located in the City.
 - For each unique City:
 - Find City Population.
 - Categorize City based on Population. City categories: Small (population <3,700,000), Medium (population >=3,700,000 and <6,700,000), Large (population >=6,700,000 and <9,000,000) and Extra Large (population >=9,000,000).
 - Find Store Number (ID) of all Stores located in the City.
 - Find all unique Years available in database.
 - For each Year:
 - For each City:
 - For each Store in the City:

- Find all Dates when Products were sold in Retail Price.
- Find Product Quantity sold on Dates of Retail Price.
- Calculate Revenue of each Product sold in Retail Price in the Store by multiplying Retail Price and Product Quantity.
- Sum up Retail Revenue of all Products sold in Retail Price.
- Find all Dates when Products were sold in Sale Price.
- Find Product Quantity sold on Dates of Sale Price.
- Calculate Revenue of each Product sold in Sale Price in the Store by multiplying Sale Price and Product Quantity.
- Sum up Sale Revenue of all Products sold in Sale Price.
- Calculate total Revenue of Products in the Store of the City of the Year by adding up both Retail Revenue and Sale Revenue.
 - Sum up total Revenue of all Stores in the City of the Year.
- Sum up Revenue of all Cities of the Year of each City Category.
- Calculate average Revenue of a City Category in the Year by dividing total Revenue by number of Cities within a City Category.
- Sort results by Year (from oldest to newest) first and then by City Size Category (from smallest to largest).
- Display result in table.