

Table of Contents

[Table of Contents](#)

[Data Types](#)

[Business Logic Constraints](#)

[Task Decomposition with Abstract Code](#)

[View Stats \(Main Menu\)](#)

[View/Add Holidays \(Holiday Maintenance Form\)](#)

[Edit City Population \(City Population Update Form\)](#)

[View Category Report \(Report 1\)](#)

[View Actual vs Predicted Revenue for Couches & Sofas \(Report 2\)](#)

[View Store Revenue by Year by State Report \(Report 3\)](#)

[View Outdoor Furniture on Groundhog Day Report \(Report 4\)](#)

[View State with Highest Volume for each Category \(Report 5\)](#)

[View Revenue by Population \(Report 6\)](#)

[View Childcare Sales Volume \(Report 7\)](#)

[View Restaurant Impact on Category Sales \(Report 8\)](#)

[View Advertising Campaign Analysis \(Report 9\)](#)

Data Types

STORE

Attribute	Data Type	Nullable
StoreNo	Integer	Not Null
PhoneNo	String	Not Null
Address	String	Not Null

CITY

Attribute	Data Type	Nullable
State	String	Not Null
CityName	String	Not Null
Population	Integer	Not Null

CHILDCARE

Attribute	Data Type	Nullable
Limit	Integer	Not Null

SOLD

Attribute	Data Type	Nullable
Qty	Integer	Not Null
TotalSale	Decimal	Not Null

DATE

Attribute	Data Type	Nullable
Date	Date	Not Null
HolidayName	List<String>	Null

HAS_DISCOUNT

Attribute	Data Type	Nullable
DiscountPrice	Decimal	Not Null

AD_CAMPAIGN

Attribute	Data Type	Nullable
Description	String	Not Null

PRODUCT

Attribute	Data Type	Nullable
PID	Integer	Not Null
Name	String	Not Null
Price	Decimal	Not Null

CATEGORY

Attribute	Data Type	Nullable
Name	String	Not Null

Business Logic Constraints

- SOLD will reference PRODUCT Price unless there is a DiscountPrice for that PRODUCT on that Date
- Adding a new HolidayName adds to the current list of holiday names for that Date
- An AD_CAMPAIGN is considered Active if it exists for that date
- Only one HolidayName can be added to a Date at a time
- Only one City Population can be updated at a time
- The Advertising Campaign Analysis only includes products that we sold. This means we are excluding products that may have been on discount and had a campaign but were never sold.

Task Decomposition with Abstract Code

View Stats (Main Menu)

Task Decomposition

Lock Types: Read-only lookups of STORE, PRODUCT, and AD_CAMPAIGN

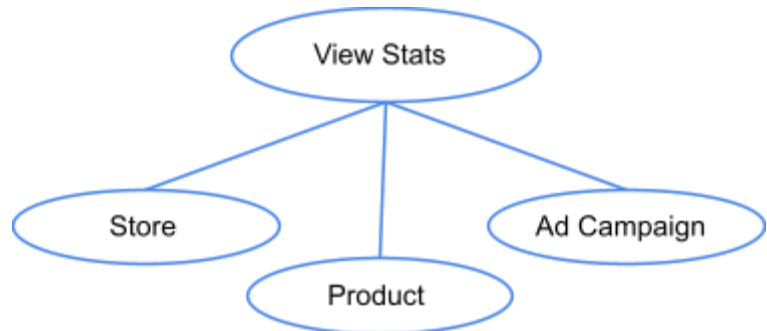
Number of Locks: Three different schema constructs are needed namely STORE, PRODUCT, and AD_CAMPAIGN

Enabling Conditions: Triggered by opening application or returning to main menu

Frequency: Low, and all have the same frequency

Consistency (ACID): is not critical, is only a lookup and is a snapshot in time

Subtasks: All tasks must be done but can be done in parallel. Order is not necessary. Mother task is required to coordinate subtasks – must be decomposed into subtasks



Abstract Code

- Show ***“Store Revenue by Year by State Report”, “Store Revenue by Year by State Report”, “Store Revenue by Year by State Report”, “Outdoor Furniture on Groundhog Day Report”, “State with Highest Volume for each Category Report”, “Revenue by Population Report”, “Childcare Sales Volume Report”, “Restaurant Impact on Category Sales Report”, and “Advertising Campaign Analysis Report”*** links.
- Show ***Holidays*** and ***City Population*** links
- Run the **ViewStats** task:
 - Run the **Store** task:
 - Count and display total of STORE instances.
 - Count and display total of STORE instances that have FOOD_SERVICE
 - Count and display total of STORE that Offers Childcare
 - Run **Product** task:
 - Count and display all PRODUCT instances
 - Run **Ad Campaign** task:
 - Count and display total of distinct AD_CAMPAIGNS
- Upon:
 - Click ***Holidays*** link - Jump to the **View/Add Holidays** task
 - Click ***City Population*** link - Jump to the **Edit City Population** task
 - Click ***Category Report*** link - Jump to the **View Category** task
 - Click ***Actual vs. Predicted Revenue for Couches & Sofas*** link - Jump to the **Actual vs. Predicted Revenue for Couches & Sofas** task
 - Click ***Store Revenue by Year by State*** link - Jump to the **View Store Revenue by Year by State** task
 - Click ***Outdoor Furniture on Groundhog Day*** link - Jump to the **View Outdoor Furniture on Groundhog Day** task

Phase 1 Report | CS 6400 - Spring 2021 | Team 029

- Click ***State with Highest Volume for each Category*** link - Jump to the **View State with Highest Volume for each Category** task
- Click ***Revenue by Population*** link - Jump to the **View Revenue by Population** task
- Click ***Childcare Sales Volume*** link - Jump to the **View Childcare Sales Volume** task
- Click ***Restaurant Impact on Category Sales*** link - Jump to the **View Restaurant Impact on Category Sales** task
- Click ***Advertising Campaign Analysis*** link - Jump to the **View Advertising Campaign Analysis** task

View/Add Holidays (Holiday Maintenance Form)

Task Decomposition

Lock Types: Lookups of DATE. Read and insert locks needed.

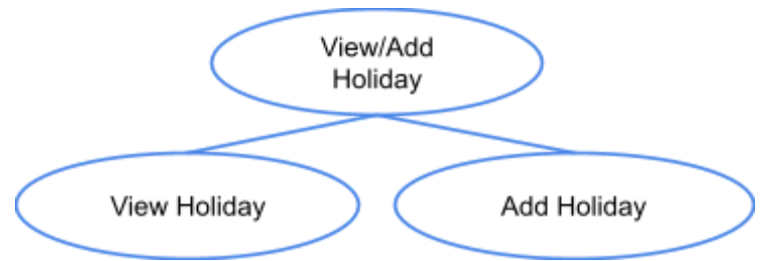
Number of Locks: One schema construct required.

Enabling Conditions: Activated when user goes into the holiday interface

Frequency: Low

Consistency (ACID): Consistency is critical as reports and stores are dependent on the holiday information for accuracy of analytics

Subtasks: Mother task is needed to coordinate subtasks. Order is not necessary.



Abstract Code

- User clicked on **Holidays** link from the **Main Menu**:
- Run the **View/Add Holidays** task:
 - Run **View Holiday** task: query for list of holidays
 - Find all the dates that don't have HolidayName as null using Date and HolidayName from DATE
 - Display each Date and all values in HolidayName associated with that date as a separate line.
- Display **Add Holiday** button at the bottom of the page along with *Date* input field and *Holiday Name* input field.
- If user clicked on **Add Holiday** button from the **Holiday Maintenance** form:
 - Data Validation: User can't enter string for date datatype in Date. User can't enter a HolidayName that already exists for that date.
 - User enters Date and HolidayName into corresponding input fields
 - If data validation is successful for both Date and HolidayName then:
 - Write a new entry to the database
 - If HolidayName exists for the Date specified, add HolidayName to existing values
 - Else add HolidayName to Date
 - Return to **Holiday Maintenance** form and Display confirmation message
 - Else HolidayName and Date are invalid, display **Holiday Maintenance** form, with error message

Edit City Population (City Population Update Form)

Task Decomposition



Lock Types: Lookups and edits of City. Read and update locks needed for CityName, State, and Population.

Number of Locks: One schema construct involved, City

Enabling Conditions: Activated when user goes into the city population interface

Frequency: Low

Consistency (ACID): Consistency isn't critical

Subtasks: Mother task isn't needed. Tasks don't need to be coordinated or executed at the same time.

Abstract Code

- User clicked on **City Population Update** link from the **Main Menu**:
- Display form with input fields for *CityName*, and *Updated Population* along with **Update Population** button
- User enters *CityName* and *Updated Population*
- Data Validation: User can only enter int datatype for Population. User can't enter int for string datatype in CityName. Population can't be updated to a null or blank value.
- Upon user clicking on **Update Population** button:
 - If data validation is correct for *CityName* and *Updated Population* then:
 - Run the **Edit City Population** task:
 - Update Population for the given CityName using CITY
 - Display CityName, State, and Population with confirmation message
 - Else CityName and Population is invalid, return to **City Population Update** form and display error

View Category Report (Report 1)

Task Decomposition

Lock types: Read only lookups of Category and Product


Number of Locks: Two schema constructs needed, Category and Product

Enabling conditions: Trigger when user clicks View Category Report link

Frequency: Low

Consistency (ACID): Same consistency for both lookups, consistency isn't critical

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



View Category Report

Abstract Code

- User clicked on **Category Report** link from the Main Menu:
- Run the **View Category Report** task:
- Find the Name for each CATEGORY
- For each CATEGORY:
 - Calculate the number of PRODUCTS in the CATEGORY; Display the number of PRODUCTS
 - Find the retail Price of the PRODUCTS in the CATEGORY
 - Calculate the average retail Price of the PRODUCTS in the CATEGORY; Display the average retail Price of the PRODUCTS
 - Find the minimum Price of the PRODUCTS in the CATEGORY; Display the minimum Price of the PRODUCTS
 - Find the maximum Price of the PRODUCTS in the CATEGORY; Display the maximum Price of the PRODUCTS
- Sort ascending the CATEGORY values by Name
- If user clicks **Main Menu** button, return to Main Menu

View Actual vs Predicted Revenue for Couches & Sofas (Report 2)

Task Decomposition

Lock types: Lookup read only of PRODUCT, CATEGORY, SOLD, DATE

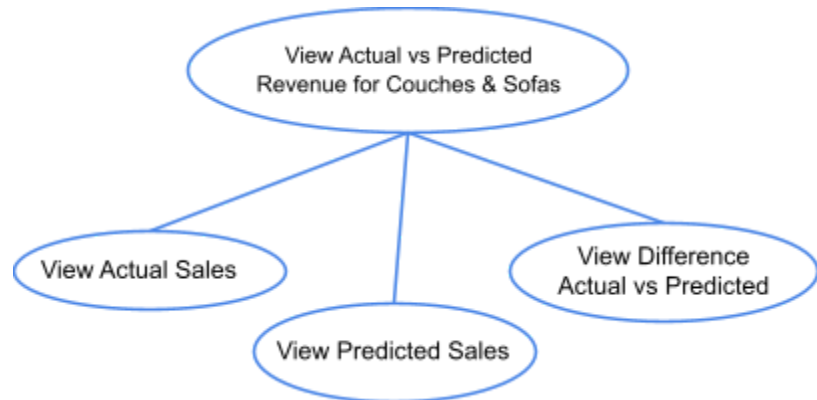
Number of Locks: Four schema constructs of PRODUCT, CATEGORY, SOLD and DATE needed

Enabling conditions: Trigger when user clicks View Actual versus Predicted Revenue for Couches and Sofas link

Frequency: Low, same frequency

Consistency (ACID): View of Actual and View Predicted Sales are done before View Difference Actual vs. Predicted

Subtasks: Order is important in the subtasks. Mother task required to coordinate the subtasks



Abstract Code

- User clicked on **Actual versus Predicted Revenue** link from the **Main Menu**:
- Run the **View Actual versus Predicted Revenue for Couches & Sofas** Task:
- Get PID, Name and Price from PRODUCT for all products within the CATEGORY "Couches and Sofas"
- For each PRODUCT
 - Run the **View Actual Sales** task:
 - Calculate the total number of units SOLD
 - Get the Date when the PRODUCT was at discount
 - Get the discount price of the product
 - Calculate the total number of units of the PRODUCT where the Date of SOLD is equal to the Date of HAS_DISCOUNT name it 'TotalNumberUnitDiscount'
 - Calculate the total total number of units in SOLD with retail Price the Date of SOLD is not equal to the Dates of HAS_DISCOUNT name it 'TotalNumberUnitWithoutDiscount'
 - Calculate the actual revenue: multiply the total number of units sold at retail Price by the retail Price plus the total number of units sold at DiscountPrice by their DiscountPrice
 - Run the **View Predicted Sales** task:
 - Multiply 'TotalNumberUnitDiscount' by 0.75 name 'NewPredictedSales'
 - Add the 'TotalNumberUnitWithoutDiscount' and 'NewPredictedSales', name it 'TotalPredictedSales'
 - Multiply 'TotalPredictedSales' by Price name it 'PredictedRevenue'
 - Run the **View Difference Actual vs Predicted** task:
 - Calculate the difference between 'ActualRevenue' and 'PredictedRevenue' name it 'Difference Actual vs Predicted'

Phase 1 Report | CS 6400 - Spring 2021 | Team 029

- Display, sorted in descending order by 'Difference Actual vs Predicted', PID, NAME, Price, total number of units sold, total number of units sold at discount price, the total revenue, the predicted revenue and 'Difference Actual vs Predicted' for each product where absolute value of 'Difference Actual vs Predicted' is > 5000
- If user clicks **Main Menu** button, return to **Main Menu**

View Store Revenue by Year by State Report (Report 3)

Task Decomposition

Lock Types: Read-only lookups of CITY, STORE, SOLD, PRODUCT and DATE.

Number of Locks: Up to 5 schema constructs are needed.

Enabling Conditions: Trigger when user clicks View Store Revenue by Year by State link

Frequency: Low, all have the same frequency.

Consistency (ACID): Low, same frequency.

Subtasks: Order is important in the subtasks.

Mother task required to coordinate the subtasks.



Abstract Code

- User clicked on **View Store Revenue by Year by State** link from the **Main Menu**:
- Run the **Select State** task:
- Query for all unique States in CITY in the database to be displayed in the UI
- User selects State from dropdown menu in the UI and clicks **Generate Report** button
- Upon clicking **Generate Report** button:
 - For each Year from DATE_SOLD
 - Find all CITY that have selected State
 - For each STORE in CITY:
 - Display StoreNo, Address and CityName
 - For each DATE_SOLD
 - Calculate TotalAmount: multiply Quantity by PRODUCT Price or Discount Price (if there is one available on this date)
 - Calculate Revenue: Add the TotalAmount calculated for each DATE_SOLD
 - Display the revenue
 - Sort by year in ascending order and then by revenue in descending order
- If user clicks **Main Menu** button, return to **Main Menu**

View Outdoor Furniture on Groundhog Day Report (Report 4)

Task Decomposition

Lock Types: Read-only lookup from SOLD, DATE, PRODUCT, CATEGORY


Number of Locks: Up to four schema constructs are required

Enabling Conditions: Trigger when user clicks Outdoor Furniture on Groundhog Day link

Frequency: Low

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

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



View Outdoor Furniture
on Groundhog Day

Abstract Code

- User clicked on ***Outdoor Furniture on Groundhog Day*** link from **Main Menu**:
- Run the ***View Outdoor Furniture on Groundhog Day*** task: query total and average quantity sold per year and on February 2.
 - Find DATE, CATEGORY, and SOLD
 - For each year in Date:
 - Find and sum total quantity within SOLD of all PRODUCTS that belong to 'Outdoor Furniture' in CATEGORY
 - Calculate average daily quantity by dividing total annual quantity sold by 365
 - Find number of units sold on Feb. 2
 - Sort list by year in ascending order
 - Display number of units sold on Feb. 2 versus average daily quantity
- If user clicks ***Main Menu*** button, return to **Main Menu**

View State with Highest Volume for each Category (Report 5)

Task Decomposition

Lock Types: Lookup DATE, CATEGORY including their Name, SOLD, STORE and CITY in which they are located. All these are Read-only.

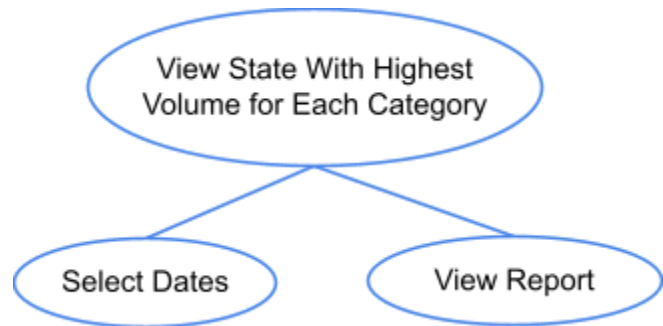
Number of Locks: 5 different schema constructs need to be accessed to retrieve the information.

Enabling Conditions: Triggered when user clicks View State with Highest Volume for each Category link. At least one date must exist in the database. A user must select a year and a month before generating the report.

Frequency: ViewDates and GenerateReport have around the same frequency.

Consistency (ACID): is not critical.

Subtasks: Mother task is needed to coordinate subtasks. Order is important. Subtasks cannot be done in parallel.



Abstract Code

- User clicked on **View State With Highest Volume of Sales for each Category** link from **Main Menu**:
- Run **View State With Highest Volume for Each Category** task:
 - Run the **Select Dates** task: query for all available DATE_SOLD in the database to be displayed in the UI
 - User selects Month and Year from dropdown menu in the UI and clicks **Generate Report** button
 - Run **View Report** task:
 - For each CATEGORY:
 - Display the CATEGORY Name
 - Find each PRODUCT_SOLD on each DATE_SOLD that corresponds to the Month and Year provided by the user that belongs to the CATEGORY
 - For each PRODUCT_SOLD find which STORE_SOLD it
 - Add all SOLD Quantity values for each STORE that belongs to the same CITY
 - Add the results obtained for each CITY that have the same State and name it "TotalNumberSold".
 - Display the State name with maximum "TotalNumberSold" and the its corresponding "TotalNumberSold"
 - If two states have the same "TotalNumberSold" then duplicate the category in the final report for each of them
 - Sort the list of categories by name in ascending order.
- If user clicks **Main Menu** button, return to **Main Menu**

View Revenue by Population (Report 6)

Task Decomposition

Lock Types: Read-only lookup of total sales from SOLD, DATE, CITY, PRODUCT


Number of Locks: Up to four schema constructs are required

Enabling Conditions: Triggered when user clicks Revenue by Population link

Frequency: Low

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

Subtasks: Mother Task is not required. Task decomposition is not required.



View Revenue by
Population

Abstract Code

- User clicked on **Revenue by Population** link from **Main Menu**:
- Run the **View Revenue by Population** task: queries TotalSale by City population categories and compare per annum
- Find SOLD, DATE, CITY, PRODUCT
- For each year in DATE:
 - For each CITY:
 - If population is < 3,699,999; "Population_category" is "Small"
 - ELIF population is < 6,699,999; "Population_category" is "Medium"
 - ELIF population is < 8,999,999; "Population_category" is "Large"
 - Else: "Population_category" is "Extra Large"
 - Return population category corresponding to that City population
 - For each DATE_SOLD
 - Calculate TotalAmount: multiply Quantity by PRODUCT Price or Discount Price (if there is one available on this date)
 - Calculate Revenue: Add the TotalAmount calculated for each DATE_SOLD
 - Display the revenue
 - Sum Revenue for all CITY in each Population Category
- Sort table by year ascending (i.e., oldest at the top, newest at the bottom)
- Sort columns by population category (i.e., smallest to extra large)
- If user clicks **Main Menu** button, return to **Main Menu**

View Childcare Sales Volume (Report 7)

Task Decomposition

Lock Types: Read-only lookup for DATE, STORE, CHILDCARE, SOLD and PRODUCT.


Number of Locks: up to 5 schema constructs need to be accessed.

Enabling Conditions: Triggered when user clicks View Childcare Sales link.

Frequency: Low.

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

Subtasks: Mother Task is not required. Task decomposition is not required.



View Childcare Sales Volume

Abstract Code

- User clicked on **Childcare Sales Volume** link from the MainMenu:
- Run the **View Childcare Sales Volume** task:
 - Find the last 12 months worth of sales by checking DATE_SOLD
 - For each month in DATE_SOLD:
 - Find each STORE_SOLD where STORE does not OFFERS offer CHILDCARE:
 - For each DATE_SOLD
 - Calculate TotalAmount: multiply Quantity by PRODUCT Price or Discount Price (if there is one available on this date)
 - Add all TotalAmount of all STOREs and display this value as “Total Sales of Stores Without Childcare”
 - Find each STORE_SOLD where STORE offers CHILDCARE:
 - For each OFFERS Limit:
 - For each DATE_SOLD
 - Calculate TotalAmount: multiply Quantity by PRODUCT Price or Discount Price (if there is one available on this date)
 - Add all TotalAmount of all STOREs and display this value as “Total Sales of Stores that offer Childcare with <Limit>”
- If user clicks **Main Menu** button, return to Main Menu

View Restaurant Impact on Category Sales (Report 8)

Task Decomposition

Lock Types: Read-only lookups of CATEGORY, RESTAURANT, STORE and SOLD


Number of Locks: Up to three different schema constructs are needed

Enabling Conditions: Triggered when user clicks View Restaurant Impact on Category Sales link

Frequency: Low.

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

Subtasks: Mother task is not required – does not need to be decomposed into subtasks.



View Restaurant Impact on
Category Sales

Abstract Code

- User clicked on ***Restaurant Impact on Category Sales*** link from the **MainMenu**:
- Run the **View Restaurant Impact on Category Sales** task:
 - For each CATEGORY:
 - Display Name
 - Find each STORE_SOLD where STORE has FOOD_SERVICE RESTAURANT:
 - Add Quantity from SOLD of all STOREs and display this value as “Quantity Sold for Restaurant”
 - Find each STORE_SOLD where STORE does not have FOOD_SERVICE RESTAURANT:
 - Add all Quantity of all STOREs and display this value as “Quantity Sold for Non-Restaurant”
 - Sort the list of categories by name in ascending order
 - For each CATEGORY group by “Quantity Sold for Non-Restaurant” and “Quantity Sold for Restaurant”. Sort by Non-Restaurant first.
- If user clicks ***Main Menu*** button, return to **Main Menu**

View Advertising Campaign Analysis (Report 9)

Task Decomposition

Lock Types: Read-only lookup of PRODUCT, SOLD, AD_CAMPAIGN and DATE.


Number of Locks: Up to 4 schema constructs needed.

Enabling Conditions: Triggered when user clicks View Advertising Campaign Analysis link.

Frequency: Low.

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

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



View Advertising
Campaign Analysis

Abstract Code

- User clicked on **Advertising Campaign Analysis** link from **Main Menu**:
- Run the **View Advertising Campaign Analysis** task:
 - Find each PRODUCT that HAS_DISCOUNT and for each of them:
 - Display the Product PID and Name
 - Find each PRODUCT_SOLD
 - Add the SOLD Quantity of all PRODUCT_SOLD where DATE has an AD_CAMPAIGN and display this value as “Sold During Campaign”
 - Add the SOLD Quantity of all PRODUCT_SOLD where DATE does not have AD_CAMPAIGN and display this value as “Sold Outside Campaign”
 - Compute the difference between the values “Sold During Campaign” and “Sold Outside Campaign” and display it as “Difference”
 - Sort the products by Difference in descending order
 - Display the top 10 and bottom 10 products only from the resulting list of products
- If user clicks **Main Menu** button, return to **Main Menu**