Table of Contents

Data Types

Store

Attribute	Data Type	Nullable
StoreNumber	String	Not Null
StreetAddress	String	Not Null
PhoneNumber	String	Not Null
Restaurant	Boolean	Not Null
SnackBar	Boolean	Not Null

Childcare

Attribute	Data Type	Nullable
TimeLimit	Integer	Not Null

City

Attribute	Data Type	Nullable
Name	String	Not Null
State	String	Not Null
Population	Long integer	Not Null

Product

Attribute	Data Type	Nullable
PID	String	Not Null
Name	String	Not Null
RetailPrice	float	Not Null

Category

Attribute	Data Type	Nullable
Name	String	Not Null

Table of Contents

Date

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

Holiday

Attribute	Data Type	Nullable
Name	String	Not Null

AdCampaign

Attribute	Data Type	Nullable
Description	String	Not Null

Discount

Attribute	Data Type	Nullable
DiscountPrice	Float	Not Null

Sale

Attribute	Data Type	Nullable
Quantity	Integer	Not Null

Business Logic Constraints

1. Childcare time limit can only be chosen from a value set.
2. Date must be ranging from 1990/01/01 to the current date.
3. Store attribute “PhoneNumber” must be of correct phone number format
4. The Discount attribute “DiscountPrice” must be smaller than the “RetailPrice” of the owner product

Table of Contents

5. The Sale Quantity must be >0
6. Retail price will be invalid if discount price is in effective
7. If a product is discounted, it is for the same price in all stores
8. Stores are not allowed to have discount independently
9. All products are available and sold at all stores
10. Each Sales item is the aggregate sale for one product in one store on one day.
11. City population must be a positive integer

Task Decomposition and Abstract Code

Main Menu

Task Decomp

Lock types: Read-only.
Number of locks: 6.
Enabling Condition: trigger by successfully open the LSRS DBMS.
Frequency: Moderate, depended on the usage of managers.
Consistency (ACID): Not critical. Order is not critical.
Subtask: No mother task needed. No decomposition needed.

Abstract Code

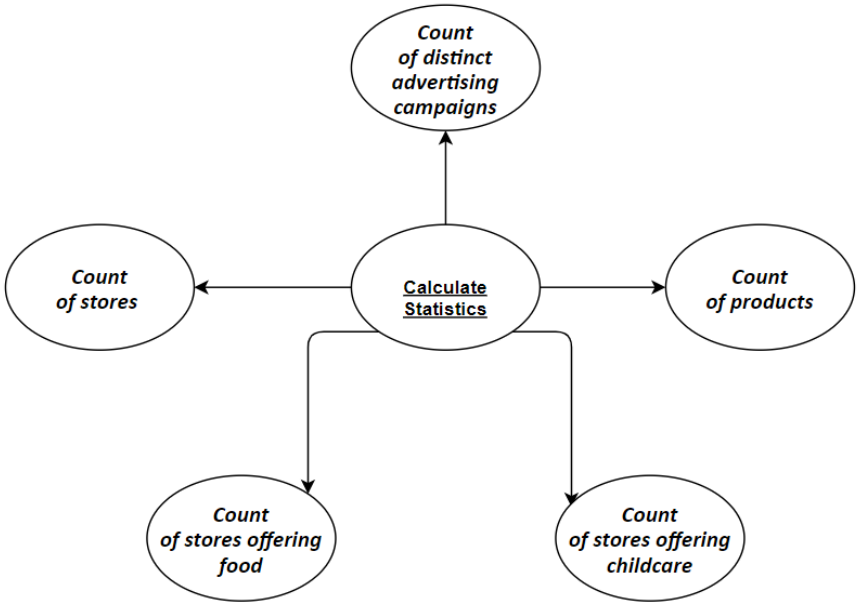
- Show "**Calculate Statistics**" button.
- Show "**View/Edit Holidays**" button.
- Show "**View/Update City Profile**" button.
- Show the buttons of "***Query all Category***", "***Actual versus Predicted Revenue for Couches and Sofas***", "***Store Revenue by Year by State***", "***Analyze Outdoor Furniture on Groundhog Day***", "***State with Highest Volume for each Category***", "***Revenue by Population***", "***Childcare Sales Volume***", "***Analyze Sales by Restaurant Existence***", and "***Count Sold Products during and outside Advertising Campaign***".
- Exit "**Main Menu**" when closing this application.

Table of Contents

Calculate Statistics

Task Decomp

Lock types: read-only
Number of locks: 4 lookups of stores, products, childcare, adcampaign.
Enabling Condition: None.
Frequency: High. The count statistic task is triggered whenever the application is used.
Consistency (ACID): Not critical. Order is not critical.
Subtask: All subtasks must be done under the mother task. They can be done in parallel and the order is not important.



Abstract Code

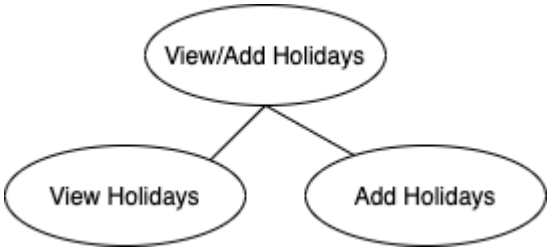
- Run the **Calculate Statistics**: Query for the information forms about Stores, Products, and Advertising Campaign from the database.
    - Find all the stores in the form; sum the number of stores; display the sum of stores.
    - Find the stores using offering childcare in the Store form; sum the number of these stores; display the sum of the stores offering childcare.
    - Find all the products in the Product form; sum the number of the products; display the sum of the products.
    - Find all advertising campaign in the AdCampaign form; sum the number of advertising campaign; display this sum.

Table of Contents

View/Add Holidays

Task Decomp

Lock Types: Read-only and write-only on Holiday.
Number of Locks: 2 locks are needed. 1 read-only for View Holidays task and 1 write-only for Add Holidays task.
Enabling Conditions: triggered by clicking ***View/Add Holidays*** button from **Main Menu**
Frequency: writing is low and reading is moderate
Consistency(ACID): is not critical, even if the Holiday is being edited by the user while another user is viewing them.
Subtasks: Mother task is required before the subtask of adding Holidays



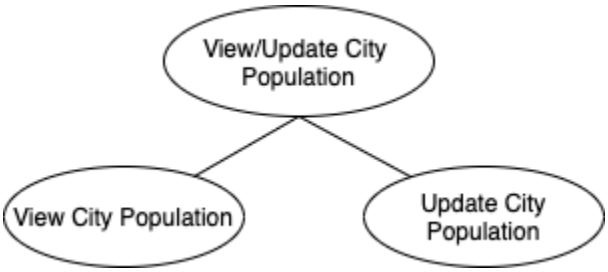
Abstract Code

- Show "***View Holidays***" and "***Edit Holidays***" tabs
- Click "***View Holidays***" button:
    - Display all holiday names and dates from Holiday
    - User can select a range of dates and the specific holidays are displayed
- Click "***Add Holidays***" button:
    - User inputs a holiday name and its date.
    - If the date is invalid, display an error message as "Invalid date".
    - If the date is valid:
        - If Holiday.Name is empty for that Date, then add the user-input and display a "Successfully added" message.
        - If Holiday.Name is nonempty and the user-input is not a substring of the Holiday.Name, then concatenate the holiday names. Display a "Successfully added" message.
        - Otherwise make no change and display "Holiday already existed" message.
- When ready, user can go back to the **Main Menu**.

View/Update City Population

Task Decomp

LockTypes: 1 read task View City Population; 1 write task Update City Population
Number of Locks: 2 - one ready; one write
Enabling Conditions: user click “View City Population”
Frequencies: Update is low and View is moderate
Consistency: not critical, even if one user is updating the city population, it doesn’t have to be viewed at the same time
Subtasks: Mother task is needed.



Abstract Code:

- User clicked on ***View Population*** on **Main menu**
- Display View Population page
    - User pick a *state* and *city name* in dropdown list
    - Run the View Population task: query the population in City where *city name* and *state* equals to the selection by the user
    - Display population on **Main Menu**
- User clicked on ***Edit Population*** on ***View Population***
    - User select a *state* and *city name* in the dropdown list
    - User input an integer for *NewPopulation*
    - If data validation is successful for *NewPopulation* then:
    - Run the Edit Population task: insert new *NewPopulation* into City.
- When ready, user selects next action from choices in **Main Menu**.

Report 1: Query All Categories

Task Decomp

Lock Types: Read-only on the Category, Product.
Number of Locks: 2
Enabling Conditions: when the “Query All Categories” button is clicked
Frequency: Moderate
Consistency (ACID): Not critical
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

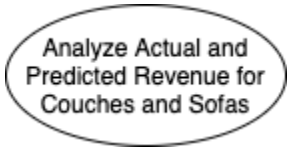
- User clicks on ***Query All Categories*** button from **Main Menu**
- Run the **Query All Categories** task: calculate statistics information about all the categories.
    - List all Categories
    - For each Category, find:
        - if it has Products;
        - the Category name;
        - total number of Products in it;
        - the minimum regular Product.RetailPrice of all Products in it;
        - the average regular Product.RetailPrice of all Products in it: record the sum of all regular Product.RetailPrices, and divide the sum by the number of all Product.regularPrices.
        - the maximum regular Product.RetailPrice of all Products in it;
    - Sort and display the Category statistics results based on Category name in ascending order
- When ready, user selects next action from choices in **Main Menu**.

Report 2: Actual Versus Predicted Revenue for Couches and Sofas

Task Decomp

LockTypes: 5 read only lookups on Sale, Product, Category, Discount, Date
Number of Locks: 5
Enabling Conditions: user click ***Actual Versus Predicted Revenue for Couches and Sofas Report*** button
Frequencies: Moderate
Consistency (ACID): Not critical
Subtasks: Mother task is not needed. No decomposition needed.

Table of Contents



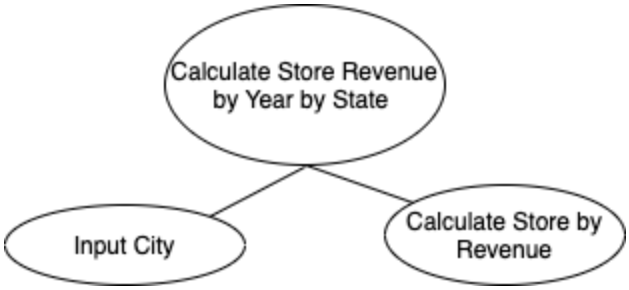
Abstract Code:

- User clicked on ***Actual Versus Predicted Revenue for Couches and Sofas Report*** button from **Main Menu**
- For each PID in Product
    - Save *RetailPrice*
    - If Category equals to Couches and Sofas
        - Query Sale, for each sale:
        - Save *Quantity*
        - If Date has Discount:
            - Save *DiscountPrice*
            - Calculate ActualRevenue using *DiscountPrice* and *Quantity*
            - Calculate *ModifedQuantity* using *Quantity*
            - Calculate PredictedRevenue using *ModifedQuantity* and *RetailPrice*
        - Else:
            - Calculate *ActualRevenue* using *RetailPrice* and *Quantity*
            - Calculate *PredictedRevenue* using *RetailPrice* and *Quantity*
- If RevenueDifference is greater than $5000 or smaller than -$5000, Show Report
- Return to **Main Menu**

Report 3: Store Revenue by Year by State

Task Decomp

LockTypes:6 read only lookups; Sale, Product, Category, Discount, Date, City
Number of Locks: 6
Enabling Conditions: user click ***Store Revenue by Year by State Report*** button
Frequencies: Moderate
Consistency: Not critical
Subtasks: Mother task is needed. Task decomposition is needed. InputCity subtask must be executed before CalculateStoreRevenue subtask.



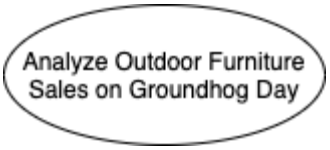
Abstract Code:

- User clicked on ***Store Revenue by Year by State Report*** button on **Main Menu**
- User select *State* in dropdown box
- For each city in City:
    - If *city.state* == *State* selected by user:
        - Save city.id
        - Query store in Store
        - For each store:
            - Query Sale, save *Quantity*
            - Query Product, save *RetailPrice*
            - Query Date, save *Year, Month, Day*
                - ➢ If date has Discount on *Year, Month, Day*:
                    - ✓ Save discount price
                    - ✓ Revenue += discount price * quantity
                - ➢ Else:
                    - ✓ Revenue += retail price * quantity
- Sort results by year in ascending order
- Sort results by Revenue in descending order if user click “Revenue” on report

Report 4: Analyze Outdoor Furniture Sales on Groundhog Day

Task Decomp

Lock Types: Read-only on Sale, Date, Category, Product
Number of Locks: 4
Enabling Conditions: when user clicks the ***Analyze Outdoor Furniture Sales on Groundhog Day*** button
Frequency: Low, because the LEOFURN team does not need to prove the idea of “customers begin thinking about the warm spring weather ahead” many times
Consistency: Not critical
Subtask: mother task is not needed. No decomposition is needed.



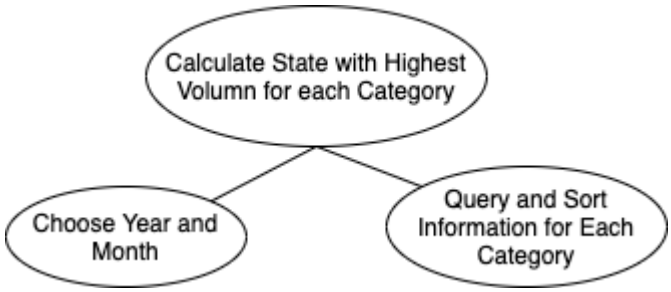
Abstract Code

- User clicks on ***Query Outdoor Furniture on Groundhog Day*** from **Main Menu**
- Run the **Query Outdoor Furniture on Groundhog Day** task: calculate statistics information about outdoor furniture sold per year
    - List all Date.year that have Sales
    - For each Date.year, find/return:
        - the Date.year;
        - in the outdoor furniture Category, the total number of Products sold that year by looking through Sale; save the result as sum_sale;
        - the average number of units sold per day: divide sum_sale by the number of Date.days for the Date.year.
        - the total number of units sold on Groundhog Day of that year;
    - Sort the results based on the year in ascending order, and display if the average number of units sold per day is larger than the total number of units sold on Groundhog Day of that year.
- When ready, user selects next action from choices in **Main Menu**.

Report 5: State with Highest Volume for each Category

Task Decomp

Lock Types: Read-only on the City, Store, Sale, Date, Product, Category
Number of Locks: Six
Enabling Conditions: when user clicks the “Calculate State with Highest Volume for each category” button
Frequency: Moderate - all 2 subtasks have the same frequency.
Consistency: non critical
Subtask: mother task is needed and it coordinates subtasks. “Choose Year and Month” must be done before “Query and sort information for each category” is processed.



Abstract Code

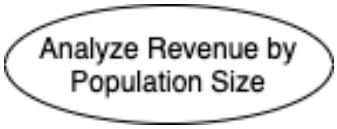
- User clicks on ***Calculate State with Highest Volume for each Category*** button from **Main Menu**

- Run the **Calculate State with Highest Volume for each Category** task: find the state with the greatest number of units for each category for the chosen year and month
    - User chooses Year and Month from the dropdown menu
    - List all Categories
    - For each Category, find:
        - the Category name;
        - the State that sold the highest number of Products in that Category, linked through City, Store, Sale, Product, Category;
        - the number of Products that were sold by Stores in that State.
    - Sort and display the results by Category name in ascending order.
- When ready, user selects next action from choices in **Main Menu**.

Report 6: Revenue by Population

Task Decomp

Lock Types: Read only
Number of Locks: 5 read-only locks of City,Product, Discount, Store, Sale and Date information.
Enabling Conditions: triggered by clicking ***Revenue by Population*** button from **Main Menu**.
Frequency: Moderate
Consistency(ACID): Not critical.
Subtasks: No mother task needed. No decomposition needed.



Abstract Code

- User clicked on ***Revenue by Population*** button from **Main Menu**
- Run the **Summarize Revenue by Population Size**: Query for the information forms about all City.Population, Product.RetailPrice, Discount.DiscountPrice, Store, Sale and Date
    - Group City by Population into Small (population <3,700,000) Medium (>=3,700,000 and <6,700,000 and =9,000,000).
    - Calculate the total revenue for a store within a given year:
        - If the product wat sold on a date without discount: Store.Revenue = Product.RetailPrice * Sale.Quantity
        - If the product was sold with a discount: Store.Revenue = Discount.DiscountPrice*Sale.Quantity
    - Adding up the total Revenues for all stores located in a City;
    - Adding up the total revenues for all cities within a population size;
    - Do the above calculations for all years.

Table of Contents

    - Order the revenue results
        - Row (Years): oldest to newest
        - Columns (Population Size): smallest to largest
- When ready, user selects ***Close*** button and go back to **Main Menu.**

Report 7: Childcare Sales Volume

Task Decomp

Lock Types: Read-only
Number of Locks: 5 read-only locks of Childcare, Product, Discount, Store, Sale and Date information.
Enabling Conditions: triggered by successfully clicking ***Childcare Sales Volume*** button from **Main Menu**.
Frequency: Moderate.
Consistency(ACID): Not critical.
Subtasks: No mother task needed. No decomposition needed.

Abstract Code

- User clicked on ***Childcare Sales Volume*** button from **Main Menu**
- Run the **Calculate Sales Volume by Childcare**: Query for the information forms about all Childcare.TimeLimit, Product.RetailPrice, Discount.DiscountPrice, Store, Sale and Date
    - Categorize Childcare into different levels (eg, No Childcare, Low, Medium and High) according to its time limit. Group Stores by their Childcare category.
    - Calculate the total revenue for a store within a given month:
        - If the product was sold on a date without discount:
          Store.Revenue = Product.RetailPrice * Sale.Quantity
        - If the product was sold with a discount:
          Store.Revenue = Discount.DiscountPrice*Sale.Quantity
    - Adding up the total Revenues for all stores within a Childcare category
    - Do the above calculations for 12 recent months.
    - Order the revenue results:
          Row (Months): oldest to newest
          Columns (Level of Childcare service): smallest to highest

- When ready, user selects ***Close*** button and go back to **Main Menu.**

Table of Contents

Report 8: Analyze Sales by Restaurant Existence

Task Decomp

Lock Type: Read-only lock
Number of Locks: We need 4 read-only locks to access 4 different entities, which are Sale, Store, Product and Category.
Enabling Conditions: user clicks the ***Analyze Sales by Restaurant Existence*** button on **Main Menu.**
Frequency: Moderate
Consistency (ACID): Not critical.
Subtasks: No mother task. Decomposition not needed.

Abstract Code

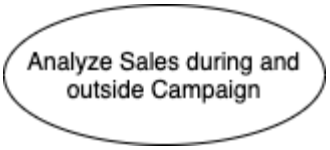
- Click on the ***Analyze Sales by Restaurant Existence*** button on **Main Menu**
- Calculate and generate result
    - Query the Sales.Quantity, Store.Restaurant (if restaurant exists or not), Product.PID, and Category.Name of all sales record
    - Group data by Store.Restaurant and Category.Name, sum all Sales.Quantity in a group
    - Sort result by non-restaurant before restaurant
    - Sort result by category name ascending
- When ready, click ***Close*** to go back to **Main Menu**

Report 9: Count Sold Products during and outside Advertising Campaign

Task Decomp

Lock types: Read-only
Number of locks: 5, **Adcampaign**, Sale, Product, Date and Discount
Enabling Condition: trigger by successfully click "***Count Sold Products during and outside Advertising*** **Campaign**" on the **Main Menu**.
Frequency: Moderate
Consistency (ACID): Not critical. Order is not critical.
Subtask: No mother task needed. No decomposition needed.

Table of Contents



Abstract Code

- User clicks **on Count *Sold Products during and outside Advertising Campaign*** button from **Main Menu**:
- Run the ***Count Sold Products during and outside Advertising Campaign***: Query for the information forms about all Products, Date, Sales, Adcampaign.
    - Find the products where the product has sold and discounted from Sales and Date; display the PID, product name and quantity.
    - Find the discount_date with AdCampaign in Date form as *Dis_Adv*;
        - Find the product in Sales form using Dis_Adv;
        - Find the product ID(PID) and sold quantity;
        - Sum the quantity as Sold_During_Campaign
    - Find the discount_date without AdCampaign in Date form as Dis_NoAdv;
        - Find the product in Sales form using Dis_NoAdv;
        - Find product ID(PID) and sold quantity;
        - Sum the quantity as Sold_Outside_Campaign
    - Combine the Sold_During_Campaign and Sold_Outside_Campaign using PID;
        - Display the product name using PID;
        - Subtract the Sold During Campaign from Sold Outside Campaign; Display the result as Difference.
    - Sort the list as Difference descending; Display top 10 and bottom 10 rows from the list to this report
- Return to the **Main Menu** when closing this report.