Contents

Data Types

AdminUser

Attribute	Data Type	Nullable
Email	String	Not Null
UserType	String	Not Null
Password	String	Not Null

CasualUser

Attribute	Data Type	Nullable
Email	String	Not Null
UserType	String	Not Null
Password	String	Not Null
Birthday	Date	Not Null
PhoneNumber	String	Not Null
FirstName	String	Not Null
LastName	String	Not Null
Age	Integer	Null

Adopter

Attribute	Data Type	Nullable
Email	String	Not Null
FirstName	String	Not Null
LastName	String	Not Null
Street	String	Not Null
City	String	Not Null
State	String	Not Null
Zip	String	Not Null
HouseholdSize	Integer	Not Null
PhoneNumber	String	Not Null

AdoptionApplication

Attribute	Data Type	Nullable
ApplicationState	Integer 0 – PENDINGAPPROVAL 1 – APPROVED	Not Null

	2 – REJECTED 3 – MATCHED 4 - ADOPTED	
ApplicationDate	Date	Not Null
DateApprovedOrRejected	Date	Null

Dog

Attribute	**Data Type**	**Nullable**
DogID	Integer	Not Null
Name	String	Not Null
Sex	String	Not Null
SurrenderDate	Date	Not Null
AgeWhenSurrender	Integer	Not Null
AlterationStatus	Boolean	Not Null
AdoptionState	String	Not Null
Description	String	Null

Breed

Attribute	**Data Type**	**Nullable**
BreedName	String	Not Null

AdoptionDetails

Attribute	**Data Type**	**Nullable**
AdoptionDetailID	Integer	Not Null
AdoptionDate	Date	Not Null
AdoptionFee	Integer	Not Null

Microchip

Attribute	**Data Type**	**Nullable**
MicrochipID	String	Not Null

Assumption: Microchip ID is as same as Microchip value which is global unique across all vendors and Microchip value is actual content of Microchip ID.

MicrochipVendor

Attribute	Data Type	Nullable
ManufacturerName	String	Not Null

Surrender

Attribute	Data Type	Nullable
SurrenderID	Integer	Not Null
SurrenderType	String	Not Null
PhoneNumber	String	Null

Expense

Attribute	Data Type	Nullable
Date	Date	Not Null
VendorName	String	Not Null
Amount	Float	Not Null

ExpenseCategory

Attribute	Data Type	Nullable
Category	String	Not Null

SystemConfig

Attribute	Data Type	Nullable
ConfigName	String	Not Null
ConfigValue	String	Null

Business Logic Constraints

- Only one user logged in at a time.
- Volunteers cannot perform adoptions.
- Expenses can only be entered by a volunteer above 18 years old.
- Only Executive Director can approve/reject adoption applications, enter adoption information, and view reports.
- Can hold only 15 dogs at a time by default.
- The number of dogs taken in cannot exceed the shelter capacity.
- If and only if “Unknown” or “Mixed” are selected, the breed of the animal can be updated. It can only be updated if the dog's sex is "Unknown". This can only be updated after surrender if the dog was unaltered when it is surrendered.
- A dog cannot be adopted until it has been altered.
- A dog may only have one microchip.
- Each dog can incur only one expense associated with a specific vendor on a given date.
- The surrender phone number will only be visible to the Executive Director, after entry by a volunteer.
- Each adopter can submit multiple adoption applications, but only one per day
- Each adoption application can only be matched to a single dog.
- If the volunteer has a milestone birthday (an age that is a multiple of 10), this should be indicated as the Executive Director will include a token of appreciation in the email, such as a gift card code.

Task Decomposition(TD) and Abstract Code(AC) for Tasks

Login

Task Decomp

Lock Types: Read-only on AdminUser table or CasualUser table and SystemConfig table
Number of Locks: Several different schema constructs are needed
Enabling Conditions: None
Frequency: Around 10 logins per day
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed

Abstract Code

- User enters *email* ('$Email'), *password* ('$Password') input fields.
- If data validation is successful for both username and password input fields, then:
    - When ***Enter*** button is clicked:
    - If User record is found but User.password != '$Password':
        - Go back to **Login** form, with error message.
    - Else
        - If User.UserType = “Volunteer”
            - If in SystemConfig table has SystemConfig.ConfigName = “loggedUser” and ConfigValue != '$Email' && ConfigValue != NULL
                - Go back to **Login** form, show error message (Only one volunteer user can log in at the same time).
            - Else

    - Store login information as session variable *Email* (‘$Email’).
    - Set the value of SystemConfig.ConfigValue = ‘$Email’ for SystemConfig.ConfigName = “loggedUser”
    - Go to **View Dashboard** form.
- Else email and password input fields are invalid, display **Login** form, with error message.

Logout

Task Decomp

Lock Types: Write-only on SystemConfig table
Number of Locks: Single
Enabling Conditions: User has logged in
Frequency: Same as **Login** task
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is **Login**. No decomposition needed

Abstract Code

- User clicks ***Log Out*** button on **Dog Dashboard**.
- Set the value of SystemConfig.ConfigValue = NULL for SystemConfig.ConfigName = “loggedUser”
- Go to **Login** form.

View Dashboard

Task Decomp

Lock Types: Read only on Dog table and SystemConfig table.
Number of Locks: Two
Enabling Conditions: Trigger by successful login.
Frequency: Daily.
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Retrieve all dogs currently in the shelter from the Dog table.
- Display dog details in columns: Dog.DogID, Dog.Name, Dog.Breed, Dog.Sex, Dog.AlterationStatus, Dog.AgeWhenSurrender, Dog.AdoptionState, ordered by Dog.SurrenderDate (oldest to newest).
- Implement filter mechanism to filter dogs by Dog.AdoptionState.
- Create clickable list where clicking on a dog's name opens the **Dog Detail** form.
- If User clicks on a dog in the list:
    - Jump to the **View Dog Details** task.
- Calculate and display the number of currently available spaces in the shelter using the Dog and SystemConfig tables.
- Show ***"Add Adoption Application"*** tab.
- Upon clicking the ***Add Adoption Application*** button:
    - Jump to the **Add Adoption** task.
- If the shelter is not full:
    - Show ***"Add Dog"*** tab.
    - Upon clicking the ***Add Dog*** button:
        - Jump to the **Add Dog** task.
- Display ***Log Out*** button
- If User clicks ***Log Out*** button
    - Jump to the **Logout** task

View Dog Details

Task Decomp

Lock Types: Read-only on Dog, Microchip, Surrender and Expense tables
Number of Locks: Several different schema constructs are needed
Enabling Conditions: User selects a dog from the Dog Dashboard
Frequency: High – Expected frequent use by volunteers and the Executive Director
Consistency (ACID): not critical, order is not critical.
Subtasks: All tasks must be done but can be done in parallel. Mother task is required to coordinate subtasks. Order is not necessary.

Abstract Code

- User selects a dog from the **Dog Dashboard**, system records *DogID* ('$DogID').
- Find dog where Dog.DogID = '$DogID', record Dog.SurrenderID as *SurrenderID('$SurrenderID')*.
- Display Dog.Name, Dog.Breed, Dog.Sex, Dog.AgeWhenSurrender, Dog.AlterationStatus, Dog.Description, and Dog.SurrenderDate.
- If the dog has a microchip (Dog.MicrochipID exists):
    - Display Dog.MicrochipID and MicrochipVendor.ManufacturedName.
- Find expenses from Expense table, grouped by Expense.Category, calculate total expense per category
- Display Expense.Date, Expense.VendorName, Expense.Amount, total expense per category
- if exist Surrender.PhoneNumber where Surrender.SurrenderID = '$SurrenderID'
    - Display Surrender.PhoneNumber

- Display Surrender.SurrenderType
- If the dog is adoptable and the user is the Executive Director, display the "***Add Adoption***" button.

Update Dog Details

Task Decomp

Lock Types: Write lock on Dog table
Number of Locks: Single locks on Dog and Microchip tables
Enabling Conditions: User selects a dog from the Dog Dashboard
Frequency: High – Expected frequent use by volunteers and the Executive Director
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

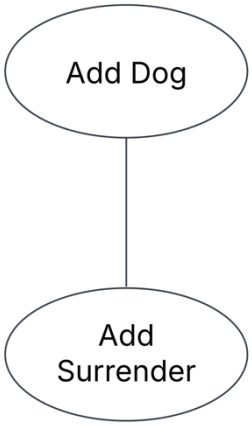
Abstract Code

- User clicks on the ***View Dog Details*** button on **Dog Dashboard**
- Form with prefilled dog info pops up (Use **View Dog Details** task).
- User can edit free text field – *sex* ('$Sex') (if "unknown"), *microchipID* ('$MicrochipID') (if not present), *breed* ('$Breed') (only if "Unknown" or "Mixed" are the current breed), and *alteration status* ('$AlterationStatus') (only if unaltered)
- Once ***submit*** button is clicked, update command is done on Dog table
    - Dog.Sex = '$Sex'
    - Dog.MicrochipID= '$MicrochipID'
    - Dog.Breed= '$Breed'

- Dog. AlterationStatus= '$AlterationStatus'

Add Dog

Task Decomp



Lock Types: Write Lock on Dog and Surrender.

Number of Locks: 2
Enabling Conditions:
When a dog is surrendered and the current number of unadopted dogs in the shelter is less than shelter's maximum capacity, a logged-in user can add information for both the dog and the surrender concurrently.
Frequency:
Varies depending on the number of unadopted dogs in the shelter, the shelter's maximum capacity, and the number of incoming dogs surrendered.
Consistency (ACID):

- Order is not critical.
- When a dog is surrendered, both the dog's details and surrender record are consistently updated.
- If the surrender is from the local animal control department, its phone number must be updated.

Subtasks:

- All tasks must be completed in order, as the created instances need to be referenced or linked.
- Subtask is **Add Surrender** and Inputs for subtasks are collected in the current mother task.

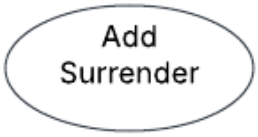
Abstract Code

- User is logged in and entered into **Dashboard.**
- If space is available (current number of dogs in shelter is less than shelter's maximum capacity)
    - Display ***Add Dog*** button.
    - Click ***Add Dog*** button.
    - Jump to **Add Dog** Form page.
    - User inputs for the following attributes for Dog:
        - *name* ('$name'), *sex* ('$sex'), *alterrationStatus* ('$alterrationStatus'), *ageWhenSurrendered* ('$ageWhenSurrendered') in months and breed ('$breed') from dropdown list.
            - Note that above attributes cannot be empty.
        - *description* ('$description'), *MicrochipID* ('$MicrochipID ') and *MicrochipVendor* ('$MicrochipVendor ') from dropdown list.
            - Note that above attribute could be empty.
    - User inputs for the following attributes for Surrender:
        - Choose *surrenderType*('$surrenderType') from dropdown list. Its either Individual or Local Animal Control Department
        - *phoneNumber* ('$phoneNumber')
            - Phone number cannot be empty if surrenderType selected as Local Animal Control Department.
    - When ***Add*** button is clicked after inputs.
    - Check the validity of all input attributes
        - If all input attributes are valid.
            - Insert new Dog into database using inputs attributes listed above and Dog.SurrenderDate = $(Today's date), Dog.AdoptionState = *NotAdopted*(default).
            - Dog.DogID is auto generated by database after insert.
            - Run subtask task **Add Surrender.**

- Else
    - Go back to **Add Dog** form and display error messages for those invalid attributes.

Add Surrender

Task Decomp



Lock Types: Write Lock on Surrender
Number of Locks: Single
Enabling Conditions:
Same as **Add Dog** as above
Frequency:
Same as **Add Dog** as above
Consistency (ACID):

- Order is not critical.
- When a dog is surrendered, both the dog's details and surrender records are consistently updated.
- If the surrender is from the local animal control department, its phone number must be updated.

Subtasks:

- Mother Task is **Add Dog.**
- No subtasks and No decomposition needed.

Abstract Code

- Run mother task **Add Dog** with all valid input attributes from user.
- With input attributes, insert new Surrender into database using $(surrenderType) and $(phoneNumber). Surrender.ID is auto generated.
- Link the new Surrender to new Dog that is added in mother task **Add Dog.**

- Jump to the **Dog Details** using Dog.DogID generated in mother task **Add Dog.**

Add Adoper

Task Decomp

Lock Types: Write Lock on Adopter table

Number of Locks: Single

Enabling Conditions: Logged-in user is the Executive Director

Frequency: Low – Occurs as needed when new adopters are registered

Consistency (ACID): Important to ensure proper record creation for adoption

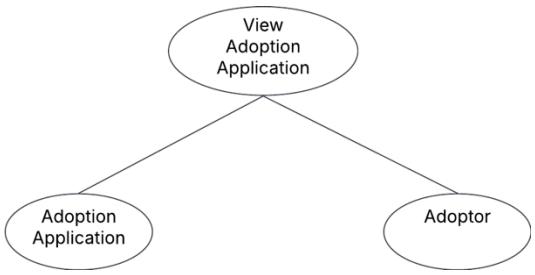
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- User logs in as Executive Director.
- Click on "*Add Adopter*" button.
- Display a form to enter adopter details: Adopter.First Name, Adopter.Last Name, Adopter.Adress, Adopter.Household,Size, Adopter.Email, Adopter.PhoneNumber.
- Validate the fields.
- If validation succeeds
    - Insert a new record into the Adopter table.
    - Display a confirmation message: "Adopter added successfully.

View Adoption Application

Task Decomp



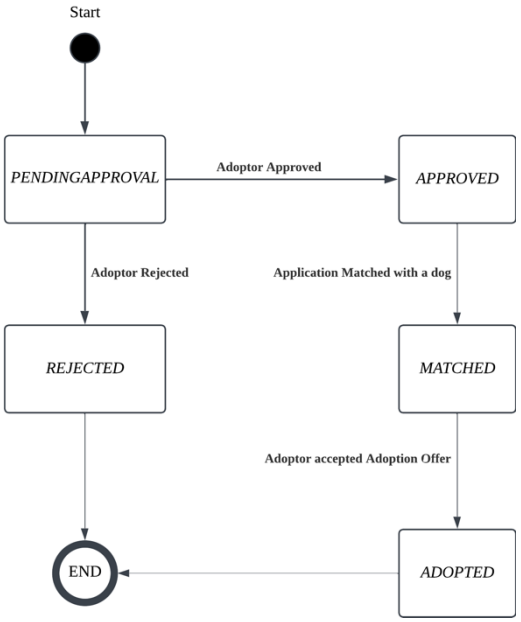
Lock Types: Read-only on AdoptionApplication table and Adopter table
Number of Locks: 2
Enabling Conditions: Logged User is Executive Director and ***Add Adoption Application*** button is clicked.

Frequency: Around 3-5 views per weekday
Consistency (ACID): not critical, order is not critical.
Subtasks:

- Mother Task is not needed. No decomposition needed.
- Optional Subtask is **View Applicant Details** if user click a particular adoption application.

Abstract Code:

- User is logged in and role is Executive Director.
- ***View Adoption Application*** button is clicked.
- Find all **Adoption Applications** whose AdoptionApplication.ApplicationState = “PendingApproval”
    - Note for States Transitions as below



- Display a list of Adoption Applications with applicationID, applicationDate and applicant's email under section **PendingApproval.**
- When user clicked a particular ***Adoption Application*** *in the list*
    - Find the applicant who applied the Adoption Application from **ADOPER** database using applicant's email.
    - Jump into **Adoption Application Details** and display applicant's details using Adoper.name, Adoper.address, Adoper.householdSize, Adoper.Email and Adoper.PhoneNumber and 2 buttons are displayed, ***Approve*** and ***Reject***.
    - Executive Director could click either ***Approve*** button or ***Reject*** button to approve or reject application.

Add Adoption Application

Task Decomp

Lock Types: Write on AdoptionApplication table and read on Adoper table.
Number of Locks: Single
Enabling Conditions: Logged User is Volunteer, click on add adoption table

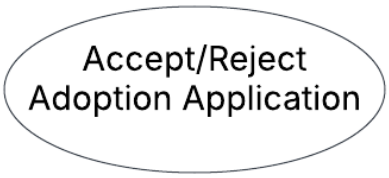
Frequency: Medium - Add adoption applications is not common

Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task: **Add Adoper** task

Abstract Code:

- User is logged in.
- User click ***add adoption application***
- User select an Adopter from Adopter database.
- User hits the submit button
- New AdoptionApplication are written to AdoptionApplication table with AdoptionApplication.ApplicationDate = $(Today's Date).

Accept/Reject Adoption Application



Lock Types: Write Lock on AdoptionApplication.
Number of Locks: Single
Enabling Conditions:

- Run mother task **View Adoption Application** and the Adoption Application detail is displayed/clicked where AdoptionApplication.applicationState = ***PENDINGAPPROVAL***.
- Logged in user is Executive Director

Frequency: Varies depends on Executive board's workload. Normally it occurs 2-3 times per day.
Consistency (ACID): not critical, order is not critical.
Subtasks:

- Mother task is **View Adoption Application.**
- No subtask.

Abstract Code

- Run mother task **View Adoption Application.**
- A particular adoption application whose state is equal to ***PENDINGAPPROVAL*** is

clicked by Executive Director.

- Jump into **Adoption Application Details** and display adoption application detail with $(Email) and $(applicationDate) and 2 buttons, ***APPROVE*** and ***REJECT***.
- Executive Director viewed the adoption application details and clicked any of the button once.
    - If Executive Director clicked ***APPROVE*** button,
    - Update AdoptionApplication.applicationState = APPROVED for AdoptionApplication instance having Adoper.Email) = $('Email') and AdoptionApplication.applicationDate) = $('applicationDate')
    - If Executive Director clicked ***REJECT*** button,
        - Update AdoptionApplication.applicationState = REJECTED for AdoptionApplication instance having Adoper.Email) = $('Email') and AdoptionApplication.applicationDate) = $('applicationDate')
    - Update AdoptionApplication. DateApprovedOrRejected = $('Today's date')
    - Jump back to **View Adoption Application** with confirmation message.

Search for Adopers

Task Decomp

Lock Types: Read-lock on Adopter
Number of Locks: Single
Enabling Conditions: One open AdoptionDetails and one available dog
Frequency: Low – Adoption events occur infrequently
Consistency (ACID): Critical; all updates must be atomic
Subtasks: Mother task is **Match Adoption Application**.

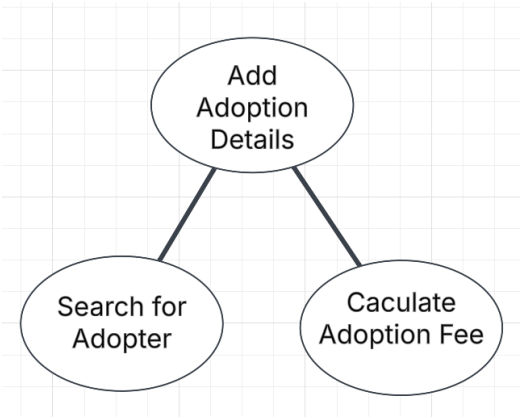
Abstract Code

- Perform mother task **Match Adoption Application**.
- User input ('$name') for searching the adopter.

- Find all Adopters whose Adopter.lastName contains ('$name')
- Display all adopters in the list including Adoper.name, Adoper.address, Adoper.householdSize, Adoper.Email and Adoper.PhoneNumber.

Add Adoption Details

Task Decomp



Lock Types: Read/write lock on Dog, write lock on AdoptionDetails and read lock on AdoptionApplication, Adopter tables
Number of Locks: Several different schema constructs are needed
Enabling Conditions: Once an adopter accepts the offer and logged in user is Executive Director
Frequency: Low – Adoption events occur infrequently
Consistency (ACID): Critical; all updates must be atomic
Subtasks: All tasks must be done and cannot be done in parallel. Mother task is required to coordinate subtasks. Order is necessary.

Abstract Code

- User click ***Add Adoption*** button on **View Dog Details** page
- Record current *dog ID* ('$DogID'), *dog name* ('$DogName')
- User enters *adopter name*('$AdopterName') input field
- Search for adopter by last name (Use task: **Search for Adopter**, pass parameter: '$ AdopterName')
- If at least one adopter matched:

- User selects the adopter from the search results
- Display the latest adoption application where AdoptionApplication.Date= MAX(AdoptionApplication.Date), record *application date*('$ApplicationDate') and Adoper Email('$AdopterEmail')
- Compute the adoption fee based on recorded expenses (Use task: **Calculate Adoption Fee**, pass parameter: '$DogID'), record *adoption fee* ('$AdoptionFee')
- User is then prompted to enter the *adoption date* ('$AdoptionDate')
- Display a confirmation screen listing the dog's name '$DogName', the adopter's contact information Adopter.Email, the adoption fee '$AdoptionFee'(if the dog belongs to any Terrier breed, and is named "Sideways", append "(waived)" after fee), and adoption date '$AdoptionDate'.
- If user click "confirm":
  - Update related information:

    AdoptionDetails.DogID = '$DogID',

    AdoptionDetails.AdoperEmail = '$AdopterEmail',

    AdoptionDetails.AdoptionDate = '$AdoptionDate',

    AdoptionDetails.AdoptionFee = '$AdoptionFee',

    Dog.AdoptionState = "Adopted",
- Else:
  - Back to **Adopter Select** page

- Else:
  - Show error message (No adopter matched)

Match Adoption Application

Task Decomp

Lock Types: Write lock on AdoptionApplication and Dog tables

Number of Locks: two locks, depending on the application and dog involved

Enabling Conditions: User is Executive: Director, adoption application is approved

Frequency: Low – Generally happens once per application

Consistency (ACID): Critical to ensure the dog is properly matched to the adoption application

Subtasks: Mother task is **View Dog Details**. No decomposition needed.

Abstract Code

- Search for available dogs (AdoptionState = 'Not Adopted', AlterationStatus = True).
- User selects a dog and clicks "**Start *Match***" button.
- Display adoption application from AdoptionApplication whose AdoptionApplication.applicationState = APPROVED.
- Executive Director selects an approved adoption application and confirm selection:
    - Update AdoptionApplication.ApplicationState = 'Matched'.
    - Link AdoptionApplication to Dog.DogID.
    - Display confirmation: "Application matched to DogID $DogID."

Add Expense

Task Decomp

Lock Types: Write lock on Expense table
Number of Locks: Single lock
Enabling Conditions: User must be a volunteer over 18 years old
Frequency: Moderate – Expenses are entered as they occur
Consistency (ACID): Important to prevent duplicate expense
Subtasks: Mother Task is **View Dog Details**. No decomposition needed

Abstract Code

- User selects "***Add Expense***" button on **Dog Detail** screen.
- Record current *dog ID* ('$DogID')
- Query User.Age by '$UserID' stored in HTTP Session, validate if User.Age ≥ 18.
    - If not, display error message and return.
- User enters *vendor name*('$VendorName'), *amount*('$Amount'), *category*('$Category'), *date*('$Date')
- Check for duplicate expense:
    - If an expense already exists for the same Expense.VendorName = '$VendorName', and Expense.Date = '$Date', display an error message.
- Insert new expense record into Expense table with '$DogID', '$VendorName', '$Amount', '$Category', and '$Date'.
- Display confirmation message and update total expenses on **Dog Detail** screen.

Calculate Adoption Fee

Lock Types: Read only on Expense. Write on ApplicationDetails
Number of Locks: Two
Enabling Conditions: Logged User is Executive Director. Currently adding a new dog available for adoption
Frequency: Moderate
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother task is **Add Adoption Details**. No decomposition needed.

Abstract Code

- When Adopter Sign back the Adoption contract, User will need to add Adoption details associated with a dog with '$DogID'.
- Adoption Fee field will be calculated for the dog.
- Find all Expenses from Expense Table associated with Dog.DogID = ('$DogID')
- Sum up all expenses by its value and notes as ('$Adoption Fee')
- If the dog is brought in my animal control, Update '$Adoption Fee' = ('$Adoption Fee') x 1.10.
- If the dog is a terrier, and named sideways, the adoption fee is waived.
- Else, Update ('$Adoption Fee') = ('$Adoption Fee') x 1.25.

View Expense Analysis

Task Decomp

Lock Types: Read only on Expense
Number of Locks: Single
Enabling Conditions: Logged User is Executive Director.
Frequency: Moderate
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.[i]

Abstract Code

- Executive Director select ***View Expense Analysis*** button
- Find all Expenses from Expense group by Expenses.VendorName.
- Calculate the sum of the amounts spent from the Expenses table, group the results by Vendor Name and name it as Total Spent. Order the results in descending order based on the total amount spent.
- Display results in a chart format with columns: Expenses.VendorName, Total Spent.

View Animal Control Report

Task Decomp

Lock Types: Read only, Surrender, AdoptionDetail and Expense Tables are needed
Number of Locks: Three

Enabling Conditions: User must be Executive Director

Frequency: Monthly, or as needed for administrative reporting

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

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

Abstract Code

- Login Verification-Ensure the user is logged in as the Executive Director before proceeding.
- Executive Director selects "***View Animal Control Report***" *Button.*
- Fetch Animal Control Data:

  -Query the Surrender table to get the count of dogs surrendered by animal control in the last 6 months, including the current month.

  -Query the Adoption table to get data on dogs that were adopted after being in the shelter for 60+ days during the same period.

  - Query the Expense table to calculate the total expenses for the adopted dogs.
- Display Aggregated Report:

  -Show the number of dogs surrendered by animal control for each month.

-Display the count of dogs adopted after 60+ days in the shelter.

-Show the total expenses for the dogs adopted in each month.

- Drill-Down Functionality:

  -Allow drill-downs for each category

  -Animal Control Surrenders Drill Down: Show Dog.DogID, Dog.Breed, Dog.Sex, Dog.AlterationStatus, Dog.MicrochipID, Dog.SurrenderDate.

  -Long Stay Adoptions Drill Down:

  -Show Dog. Dog ID, Dog. Breed, Dog. Sex, Dog. Microchip ID, Dog. Surrender Date in Rescue.

- Adoption Expenses Drill Down:

  -Show Dog.DogID, Dog.Breed, Dog.Sex, Dog.MicrochipID, AdoptionDetails. AdoptionDate, Dog.SurrenderDate, Expense.Amount

  Animal Control Indicator, Total Expenses.

- Sorting:

  -Animal Control Surrenders Drill Down: Sort by Dog ID ascending.

  -Long Stay Adoptions Drill Down: Sort by Days in Rescue descending, then Dog ID descending.

  -Adoption Expenses Drill Down: Sort by Dog ID ascending.

- Display Results:

  Show the aggregated data and drill-downs for the last 6 months+, reflecting real-time data for the current month.

View Monthly Adoption Report

Task Decomp

- **Lock Types**: Read only, Dog,Surrender, AdoptionDetail and Expense Tables are needed.
- **Number of Locks**: Three different schema constructs are needed.
- **Enabling Conditions**: User must be Executive Director
- **Frequency**: Monthly
- **Consistency (ACID)**: Not critical, order is not critical
- **Subtasks**: Mother Task is not needed. No decomposition needed.

Abstract Code

- Login Verification-Ensure the user is logged in as the Executive Director before proceeding.
- Executive Director selects *"**Monthly Adoption Report**"* Button.
- For each ($'month') in the past 12 months. Perform the following queries order by ($'month') ASC
    - Net profit = Sum of AdoptionDetails.adoptionFee where AdoptionDetails. Adoptiondate = ($'month') - Sum of all Expense.amount where Expense.date = ($'month')
    - Display count of Dog.surrenderDate = ($'month') and the count of AdoptionDetails.AdoptionDate = ($'month').
    - Show all breeds if the following are not zero
        - Find ($'breed') where count numbers of Dog.surrenderDate = ($'month') or AdoptionDetails.AdoptionDate = ($'month') is greater than 0
            - Show breeds and count order by name of breeds(A-Z)

View Birthday Celebration Report

Task Decomp

Lock Types: Read only CasualUser table
Number of Locks: Single
Enabling Conditions: Logged User is Executive Director.
Frequency: Monthly
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Executive Director selects ***"View Birthday Celebration Report"*** Button.
- Perform query: Find volunteer details (first name, last name, email, birthdate) from CasualUser table.
- Calculate the current age of each volunteer in the current month.
- Check if any volunteers have birthdays in the current month. If no birthdays are found in the current month:
    - Display message: "No volunteer birthdays this month!"
- Else:
    - For each volunteer, if the age is a multiple of 10:
        - Display report with columns from CasualUser: First Name, Last Name, Email, Milestone Birthday (Yes).
    - Else:
        - Display report with columns from CasualUser: First Name, Last Name, Email, Milestone Birthday (No).
- Store each monthly report.
- If User enters month ('$Month') and year ('$Year') input fields (current and previous year only):
    - Display the corresponding report.

Search for Volunteers

Task Decomp

Lock Types: Read only on CasualUser
Number of Locks: Single
Enabling Conditions: Logged User is Executive Director.
Frequency: Moderate
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User enters volunteer name ('$Volunteername') input field.
- Show ***Search*** tab.
- If data validation is successful for volunteer name input field, then:
    - When ***Search*** button is clicked:
        - Perform query: Find CasualUser.FirstName, CasualUser.LastName, CasualUser.Email, CasualUser.PhoneNumber of users where the CasualUser.firstName or CasualUser.lastName (in lowercase) matches the lowercase version of the specified volunteer name. Then, order the results by last name in ascending order and, within the same last name, by first name in ascending order.
        - If results are found:
            - Display results in a table format with columns: CasualUser.FirstName, CasualUser.LastName, CasualUser.Email Address, CasualUser.PhoneNumber.
        - Else:
            - Display search form with error message: "No results found for the entered name."
- Else:

Display search form with error message: "Invalid input. Please enter a valid name."