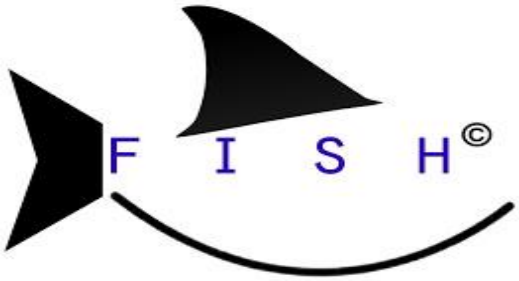


F.I.S.H.

Project Design Document

10/20/2020

Team Member: Zengxiaoran Kang, Yifei Liang, Yuqi Sun, Taowei Ji

Class Section: CSCE 315 501 Section

Table of Contents

1. Introduction

This section provides an overview of the setup of Project 2 - Database, which involves the development of the Graphic User Interface, business analytic platform, and a SQL-based backend support.

The Project Design Document describes design goals and considerations, provides a high-level overview of the system architecture, and describes the data design associated with the system, as well as the graphic interface and operational scenarios.

The name of our team is Fish and the project name is Project 2 – database. It is the first group assignment requested by Programming Studio CSCE 315-501 section lectured by Professor Yunsock Choe.

The project can be substantially decompose to three major sections. The first section is the Graphic User Interface (GUI) which creates with Java Swing and AWT packages and responsible for interacting with users' demands and queries. The second is the backend enginee which is hosted on MySQL database schema. The database we use, adventureworks, is offered by Professor Yunsock Choe. The last section is the Dashboard specifically designed for busiess analysis with support from XChart and maven. The team operates changes and collaborate over Github cloud, and communicate effectively via Zoom channel. We have managed to complete this amazing project in a month.

2. General Overview and Key Functionalities

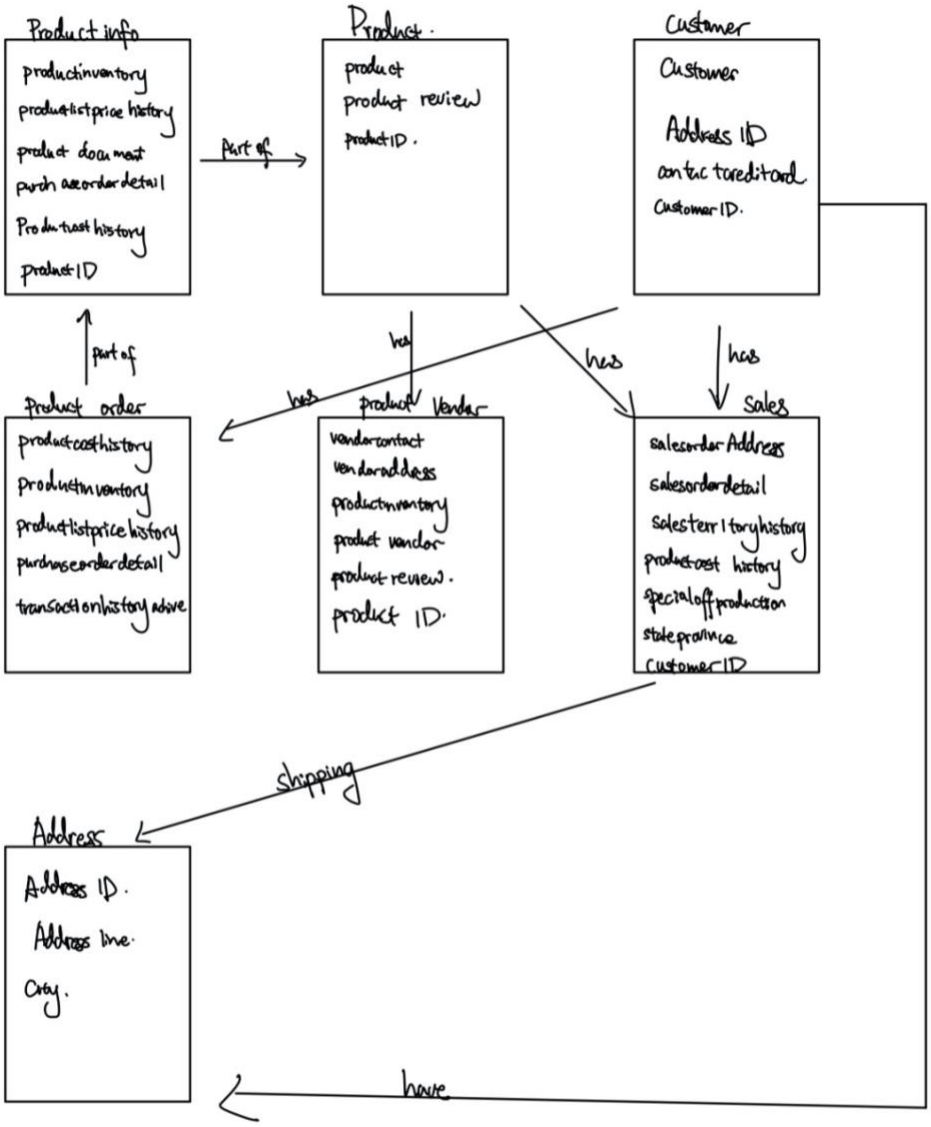
2.1 General Overview

This section contains the general overview of the key functionality that the database client will provide, an ER-diagram of the major subset (this has to be substantial) of the database that our database client will support, and list of major SQL queries that the client will employs.

2.2 Entity Relationship Diagram

Describe relationships among the adventurework database.

https://app.creately.com/diagram/c4XyIDO58Ua/view



2.3 Key Functionalities

Key functionality our database client will provide user-interface are listed below:

- jdb-show-related-tables <table-name>
- jdb-show-all-primary-keys function
- jdb-find-column <column-name>
- jdb-search-path <table1> <table2>
- jdb-search-and-join <table1> <table2>
- jdb-get-view <view-name> '(' < sql query > ')'
- jdb-stat <table> (or <view-name>) <column_name>
- jdb-show-head <table name> <number of rows print out
- jdb-product-to-where <city
- jdb-info-of-subtotalitem <the first number of the ordering
- jdb-delete-view <viewName
- jdb-see-views
- jdb-show-tables
- jdb-join-table
- jdb-show-columns-of-a-table
- Custom sql command
- Insert INTO
    - Select
        - Where
        - Group by
        - Join
        - Select
    - Value
- Update
- Delete
- Create
    - Table
    - View
    - Index
- Drop
- Alter
- Union
- Join
    - Left join
    - Right join
    - Inner Join
- NULL
- Aggregate functions
    - Count
    - Sum
    - Avg
- Group by
- Order by

3. SQL Queries Design

3.1 Goals and Guidelines

The project requires at least 20 different kinds of queries, and denotes that most queries must involve multiple tables (JOIN). All the visualization including diagram of the database and visualization tool for business analysis need to be deployed dynamically.

3.2 Development Methods

The system of our project is rooted from SQL database and centralized around Java GUI. We start from developing a simple prototype - JDBC terminal interface. The Terminal is able to perform foundmental operation such as interacting with the adventurework SQL database. Then, we extract the JDBC terminal interface out from the JDBC class and implement a new graphical interface with Java Swing and AWT. This process involves advanced strategry over object-oriented techqniue which allows the frontend Java GUI to communicate with the backend JDBC class. Finally, we take a step further by adding the Dashboard to Java GUI. To create new jdb customize function, the developer needs to initialize a class function in JDBC, and test it with JDBC terminal class to validify its robustness. Next, the developer can add the drop down box option inside GUI's JCombobox, and connect it with JDBC through JDBC object reference.

3.3 Examples

- Most queries must involve multiple tables (JOIN)
  - Join Customer and Customer address table

```
SELECT * From Customer
 Join CustomerAddress On(Customer.CustomerID =
CustomerAddress.CustomerId)
```

  -

	PasswordSalt	CustomerID	AddressID	AddressType
p+KwQhl2fJL7w=	1KjXYs4=	1	832	Main Office
+CRgbvJltknw=	fs1ZGhY=	2	833	Main Office
+CRgbvJltknw=	fs1ZGhY=	2	297	Shipping

  - Getting real address by joining on address table

```
SELECT * From Customer
 Join CustomerAddress On(Customer.CustomerID =
CustomerAddress.CustomerId) Join Address
on(CustomerAddress.AddressId = Address.AddressID)
```

  -

sType	AddressID	AddressLine1	AddressLine2	City	StatePro
ffice	832	2251 Elliot Avenue	NULL	Seattle	Washingt
ffice	833	3207 S Grady Way	NULL	Renton	Washingt
g	297	7943 Walnut Ave	NULL	Renton	Washingt

- Aggregate functions such as COUNT, SUM, AVG, etc. must be included at least 5 times.
  - How many items with ListPrice more than $1000 have been sold?

```
Database changed
mysql> SELECT COUNT(*) AS Total From salesorderdetail JOIN product ON salesorder
detail.productID = product.productid WHERE product.listprice > 1000;
+-------+
| Total |
+-------+
| 29382 |
+-------+
```

  - How many products in ProductCategory 'what' have been sold to an address in 'somewhere'?

```
SELECT
  SUM(SalesOrderDetail.OrderQty)
FROM
  ProductCategory
  JOIN
    Product
    ON ProductCategory.ProductCategoryID = Product.ProductCategoryID
  JOIN
    SalesOrderDetail
    ON Product.ProductID = SalesOrderDetail.ProductID
  JOIN
    SalesOrderHeader
    ON SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesorderID
  JOIN
    Address
    ON SalesOrderHeader.ShipToAddressID = Address.AddressID
WHERE
  Address.City = 'somewhere'
  AND ProductCategory.Name = 'what';
```

SUM(SalesOrde..
1924

  - Show the best selling item by total value

```
SELECT
  Product.Name,
  SUM(SalesOrderDetail.OrderQty * SalesOrderDetail.UnitPrice) AS Value
FROM
  Product
  JOIN
    SalesOrderDetail
    ON Product.ProductID = SalesOrderDetail.ProductID
GROUP BY
  Product.Name
ORDER BY
  Value DESC;
```

Name	Value
Touring-1000 Blue, 60	37191.44
Mountain-200 Black, 42	37178.73
Mountain-200 Black, 38	35801.74
Road-350-W Yellow, 48	33509.58
Touring-1000 Yellow, 60	23745.32
Touring-1000 Blue, 50	22887.04
Mountain-200 Silver, 42	20879.85
Road-350-W Yellow, 40	20411.8

- Find the number of what product ordered by CompanyName ‘what company’’

```
SELECT
  SUM(SalesOrderDetail.OrderQty) As Total
FROM
  SalesOrderDetail
  JOIN
    Product
    ON SalesOrderDetail.ProductID = Product.ProductID
  JOIN
    SalesOrderHeader
    ON SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesOrderID
  JOIN
    Customer
    ON SalesOrderHeader.CustomerID = Customer.CustomerID
WHERE
  Product.Name = 'product name'
  AND Customer.CompanyName = 'company name';
```

SUM(SalesOrde..
9

- Show the best selling item by average value

```
SELECT
  Product.Name,
  AVG(SalesOrderDetail.OrderQty * SalesOrderDetail.UnitPrice) as Value
FROM
  Product
  JOIN
    SalesOrderDetail
    ON Product.ProductID = SalesOrderDetail.ProductID
GROUP BY
  Product.Name
ORDER BY
  Value DESC;
```

Name	Value
Mountain-200 Black, 42	9294.6825
Road-350-W Yellow, 48	8377.395
Road-350-W Yellow, 42	7943.59
Touring-1000 Blue, 60	7438.288
Road-250 Black, 48	7330.05
Road-250 Black, 44	6597.045
Mountain-200 Black, 38	5966.956667

- GROUP BY statement must be used at least 2 times.
  - List the CompanyName for James alphabetically
  - ```
    SELECT CompanyName
    FROM Customer
    WHERE FirstName='James'
    GROUP BY CompanyName;
    ```
CompanyName
Family Cycle Store
Leather Seat Factory
Out-of-the-Way Hotels
Refined Department Stores
Road-Way Mart
Timely Shipping Service
  - Group the product using productID
  - ```
    SELECT ProductID, Name
    FROM Product
    GROUP BY ProductID, Name
    ```
ProductID	Name
680	HL Road Frame - Black, 58
706	HL Road Frame - Red, 58
707	Sport-100 Helmet, Red
708	Sport-100 Helmet, Black
709	Mountain Bike Socks, M
710	Mountain Bike Socks, L
711	Sport-100 Helmet, Blue

- ORDER BY statement must be used at least 2 times.
  - Order by orderdate for all row in SalesorderHeader

```
SELECT *
  FROM SalesOrderHeader JOIN SalesOrderDetail
  Order by
  orderdate
```

  -

SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status
71923	1	Tue, 01 Jun 2004 00:00:00 GMT	Sun, 13 Jun 2004 00:00:00 GMT	Tue, 08 Jun 2004 00:00:00 GMT	5
71885	1	Tue, 01 Jun 2004 00:00:00 GMT	Sun, 13 Jun 2004 00:00:00 GMT	Tue, 08 Jun 2004 00:00:00 GMT	5
71816	1	Tue, 01 Jun 2004	Sun, 13 Jun	Tue, 08 Jun 2004	5

  - Order by customerID for all row in SalesorderHeader

```
SELECT *
  FROM SalesOrderHeader JOIN SalesOrderDetail
  Order by
  CustomerID
```

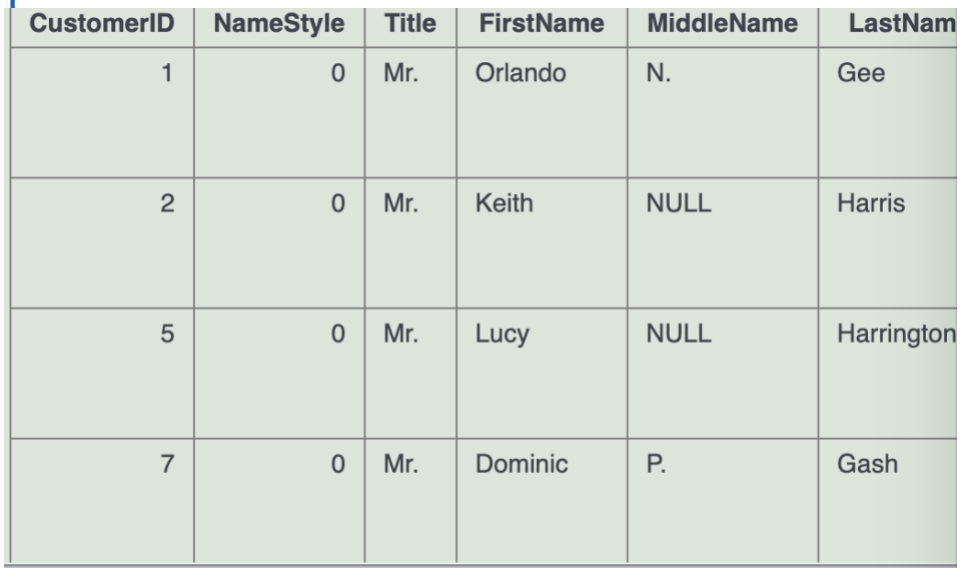
  -

Result:

SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status
71915	1	Tue, 01 Jun 2004 00:00:00 GMT	Sun, 13 Jun 2004 00:00:00 GMT	Tue, 08 Jun 2004 00:00:00 GMT	5
71915	1	Tue, 01 Jun 2004 00:00:00 GMT	Sun, 13 Jun 2004 00:00:00	Tue, 08 Jun 2004 00:00:00 GMT	5

- WHERE clause is used to filter records and extract only those records that fulfill a specified condition.
  - Find all male customers

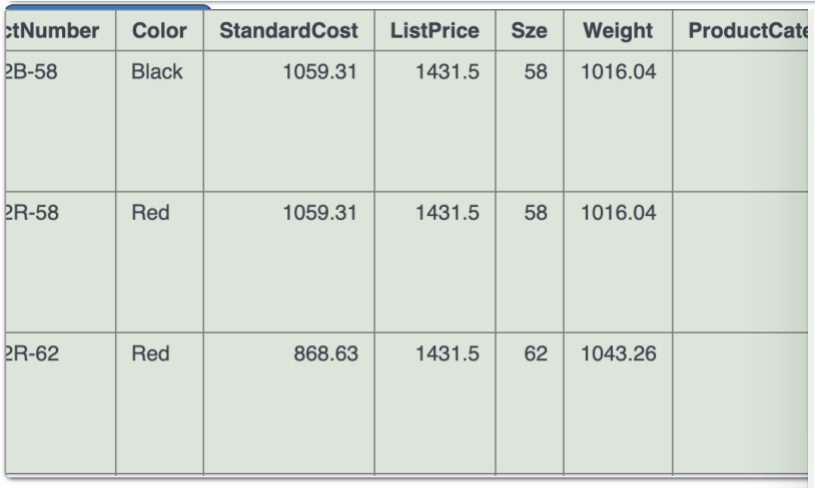
```
Select *
From Customer
Where Title = 'Mr.'
```

  - 

CustomerID	NameStyle	Title	FirstName	MiddleName	LastNam
1	0	Mr.	Orlando	N.	Gee
2	0	Mr.	Keith	NULL	Harris
5	0	Mr.	Lucy	NULL	Harrington
7	0	Mr.	Dominic	P.	Gash

  - Find all product with weight greater than 100

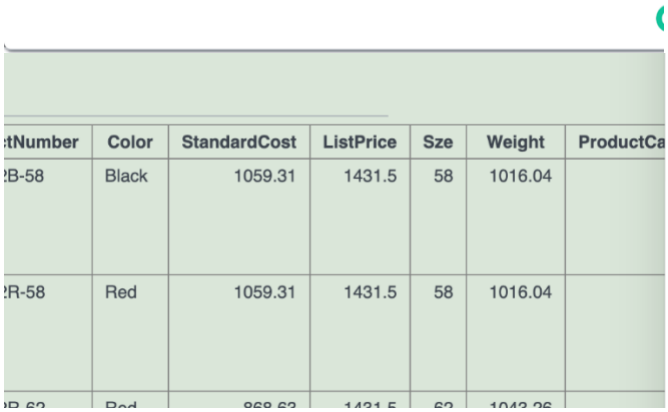
```
Select *
From Product
Where weight >100
```

  - 

ctNumber	Color	StandardCost	ListPrice	Sze	Weight	ProductCate
2B-58	Black	1059.31	1431.5	58	1016.04	
2R-58	Red	1059.31	1431.5	58	1016.04	
2R-62	Red	868.63	1431.5	62	1043.26	

- Find Product size > 50 and weight >100

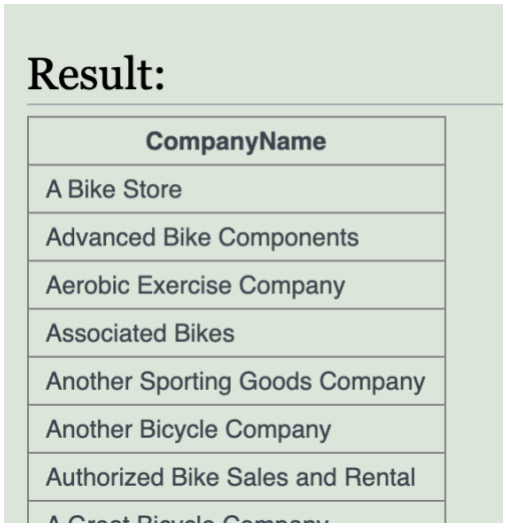
```
Select *
From Product
Where weight >100 and sze > 50
```



tNumber	Color	StandardCost	ListPrice	Sze	Weight	ProductCa
B-58	Black	1059.31	1431.5	58	1016.04	
R-58	Red	1059.31	1431.5	58	1016.04	

- LIKE operator must be used at least 1 time.
  - List the CompanyName start with ‘a’

```
SELECT CompanyName
  FROM Customer
 WHERE CompanyName LIKE 'a%';
```

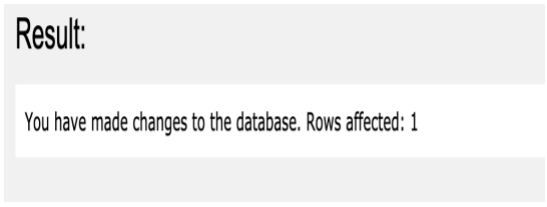


Result:

CompanyName
A Bike Store
Advanced Bike Components
Aerobic Exercise Company
Associated Bikes
Another Sporting Goods Company
Another Bicycle Company
Authorized Bike Sales and Rental

- Update
  - Update the first name and email for customer 1(ID)

```
UPDATE Customer
SET FirstName='Kim', EmailAddress='12345@mail.com'
WHERE CustomerID=1;
```



Result:

You have made changes to the database. Rows affected: 1

- Alter
  - Remove the Phone column from Customer table

```
ALTER TABLE Customer
DROP COLUMN Phone;
```

  -
  - Add Sex and Size column to Customer tabnle

```
ALTER TABLE CUSTOMERS ADD SEX char(1),SIZE char(1);
```

  -
- Drop
  - Remove entire CustomerAddress table

```
DROP TABLE CustomerAddress;
```

  -
- Delete
  - Delete all companies name starting with ‘a’ in Customers table’s CompanyName column

```
DELETE FROM Customers WHERE CompanyName like 'a%';
```

  -
- Null
  - Find any customer that does not have an email address

```
SELECT FirstName, LastName, EmailAddress
FROM Customer
WHERE EmailAddress IS NULL;
```

  -

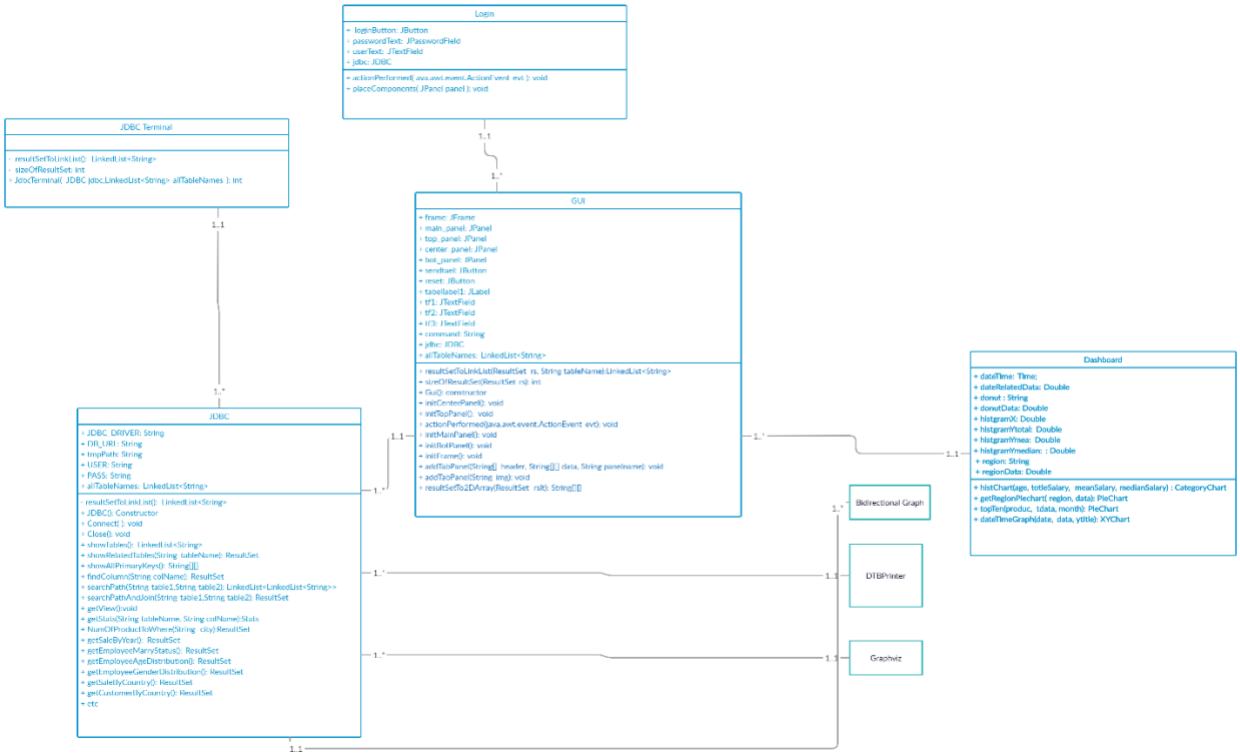
Result:

FirstName	LastName	EmailAddress

Show what the answer should be...

4. Overall System Design

This section will illustrate a thorough and logical view of the overall system design and organization, indicating whether the processing system is distributed or centralized.



4.1 JDBC-based database connectivity

JDBC class is the essential bridge to serve the connectivity between SQL database and other Java programs. JDBC manages connection to the database, issuing queries and commands, and transforming feedback from the database into a meaningful image to the GUI and Dashboard. JDBC class is used inside GUI class as an object reference variable.

4.2 Graphic User Interface (GUI)

Java GUI serves the purpose of interacting the application with users/customers. The Java GUI will take in different inputs from the users with the menu displayed above (i.e. demonstrated in section 6 operational scenario). The Java GUI will package all necessary parameters, interpret the requests and translate them into a set of SQL queries, and eventually forward the information to the JDBC class.

After that, depending on the interactive commands, the Java GUI will retrieve the data feedback from JDBC and display the content visually to the users.

There are static parsing functions inside GUI designed to extract useful information from the database after issuing queries. Since the database is protected under read-only mode, most of the commands or queries will return a JTable displayed on the bottom of GUI.

The class have built-in error handlers that will safely manage any potential errors and exceptions that occur when invalid inputs are given before the queries being sent.

4.2.1 Login / Security Architecture

Login class is used to verify user credential information before giving them access to the GUI system. After entering correct credential, the Login class will open up the GUI class and initialize JDBC connectivity simultaneously.

4.3 Dashboard

Dashboard class is a visualization platform that serves to assist business analysis for the customers. Dashboard uses XChart, which is a light-weight and convenient library for plotting data. Its focus is on simplicity and ease-of-use, requiring only two lines of code to save or display a basic default chart. The GUI will first generate data by retrieving information from the database, and stores a set of images dynamically. Then, the GUI class will read from stored images and displayed to users.

4.4 JDBC Terminal

JDBC Terminal is a protype developed on earlier stage that equip basic command line input functionality which allows developers to validify new query methods before moving on to next stage of production in GUI.

4.5 Bidirectional Graph

The Bidrectional Graph class is an online resource that is used to support the construction of jdb-join-table and jdb-find-path customized commands. This class will reconstruct the relationship schema in SQL database to adjacency matrix standard and perform Breadth First Search (BFS) to find the shortest path available between two tables via primary and foreign keys. This class is only used inside JDBC class to serve computation on graphical analysis.

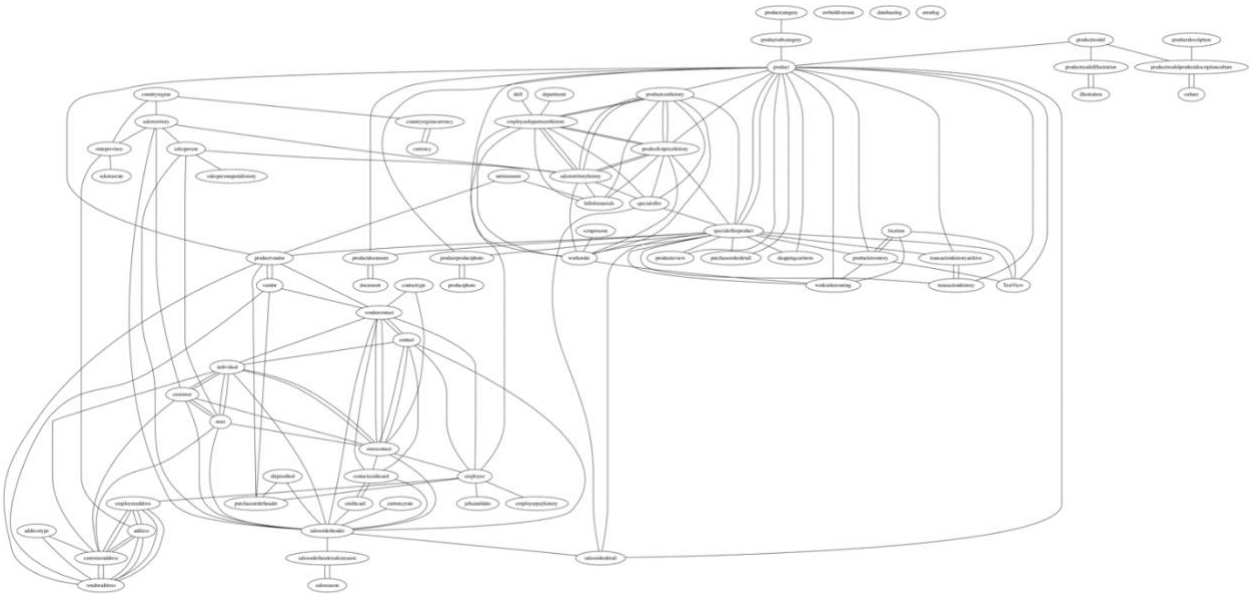
4.6 DTB Printer

DTB Printer class is a github resource that is used to print out ASCII presentation on the content of SQL query. This class is used in JDBC Terminal statically to help developers test their experimental methods.

4.7 Graphviz

Graphviz is open source graph visualization software. Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. The users require install this dependency in oder to enjoy the full functionality of our project. This class being used in JDBC class and its product image is displayed in GUI class. The image below is

an overview of our database.



5. Major Milestones

5.1 MS1 – Business / Assignment Requirements

Time: 09/20 Sunday 11:59 p.m.

- Construct outline and objective goal of Database project
- Target a set of main features filling out imaginary customer's needs
- List out a timeline of major milestones to pace up the process of production

5.2 MS2 - Database Design

Time: 09/22 Tuesday 11:59 p.m.

- Construct outline and objective goal of Database project
- Target a set of main features filling out imaginary customer's needs
- List out a timeline of major milestones to pace up the process of production

5.2.1 Setup Environment for File and Database Structures

Time: 09/24

- Being able to send queries and extract raw data from SQL database
- Initiate a simple Java GUI server and connect a few functionality with the JDBC based client.

5.3 MS3 - Data Conversion

Time: 09/27 Sunday 11:59 p.m.

- Generalize ways to request queries to SQL database.
- Able to send and retrieve data from SQL database through JDBC.

5.4 MS4 - User Interface

Time: 10/03 Saturday 11:59 p.m.

- Design a proper layout for Java GUI and construct a prototype interface.
- Create a small example set of JUnits

5.4.1 Inputs

Time: 10/04 Sunday 11:59 p.m.

- Setup the prototype of Java GUI and connect GUI class with JDBC class
- Create parsing functionality in GUI
- Adding Graphiviz key functionality

5.4.2 Outputs

Time: 10/07 Wednesday 11:59 p.m.

- Ensure the input validation and handle potential errors and exceptions
- Provide GUI-based interface to interact with JDBC with full functionality

5.5 MS5 - Dashboard

Time: 10/14 Wednesday 11:59 p.m.

- Evolve DB GUI to have simple data analytics functions.
- Import third-party graphing/charting libraries

5.6 MS6 – Desgin Document and Final Submission

Time: 10/16 Friday 11:59 p.m.

- Review and rectify original design documentation.
- Testing out each functionality
- Report any unintentional results and fix them accordingly

Time: 10/18 Sunday 11:19 p.m.

- Finish Individual report
- Submit final project

6. Operational Scenarios

This secition describe the general functionality of the system from the users' perspectives.The scenarios tie together all parts of the system, the users, and other entities by describing how they interact.

The GUI interaction box has a drop down box at the top, in which the users can select the functions from the drop down box and depending on the function, they may or may not need to input different conditions for the function.

To clear all the input in the text box, press reset.

To send the input into gui, press send

After sending all the parameters that are needed by the function, press send to get the result back. If one of the parameters is entered incorrectly, there will be a pop up window that shows what is wrong with the user input.

You can also change the view of the GUI by select the lower right corner drop down box.

Functions:

jdb-show-related-tables <table-name>: This takes in one input, and the functions shows the the entered table's related tables, which are the tables it is connected to on the graph.

jdb-show-all-primary-keys function : this function shows all the table's primary keys

jdb-find-column <column-name>: This takes in one input, and it shows all the column

jdb-search-path <table1> <table2>: This takes in two input, and it shows the path between two tables if they are connected

jdb-search-and-join <table1> <table2>: This takes in two input, first it will find the path between two tables, then it will join the table together.

jdb-get-view <view-name> '(' < sql query > ')': This takes in two input, will create a view using the sql query.

jdb-stat <table> (or <view-name>) <column_name>: This takes in two input, where it will print out a historgram for the data and find the min max of the data.

jdb-slice <tableName> <index1> <index2>: This will take in three input where it will print the table from line index1 to index2

jdb-show-head <table name> <number of rows print out>: This will take in two input where it will print the first number of rows of the table.

jdb-product-to-where <city>: this will print the number of product of in the city

jdb-info-of-subtotalitem <the first number of the ordering>: This will have 1 input and it print the product ordering that start with the number

jdb-delete-view <viewName>: this delets the view

jdb-see-views: this will print all views created

jdb-show-tables: this will show all the tables

jdb-draw: this will print the table relation graph

jdb-join-table: this will join together all the tables the user enter

jdb-show-columns-of-a-table: this will show the table's columns depending on the input

Custom sql command: enter a custom sql command and a table will be print out.

7. References

This section provides a list of remarks and links that the team employ as references during the developing process. Most are used in the implementation for JDBC, Gui, and Dashboard classes.

1. Bidirectional Tables - https://www.geeksforgeeks.org/graph-and-its-representations/
2. SQL Query jdb-stats - https://www.geeksforgeeks.org/calculate-median-in-mysql/
3. Design Document Template - https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjxkI-20L_sAhVJ2qwKHbgyAkMQFjATegQIChAC&url=https%3A%2F%2Fwww.cms.gov%2Fresearch-statistics-data-and-systems%2Fcms-information-technology%2Fxlc%2Fdownloads%2Fsystemdesigndocument.docx%23%3A~%3Atext%3DThe%2520System%2520Design%2520Document%2520(SDD)%2520describes%2520how%2520the%2520functional%2520and%2Cdocumented%2520in%2520the%2520Logical%2520Data&usg=AOvVaw0K6mjaN8CPeX1m6KaQlqZX
4. DTB Table Printer - https://github.com/htorun/dbtableprinter
5. SQL Tutorial - https://www.w3schools.com/sql/default.asp
6. AdventureWorks Sandbox - https://sqlzoo.net/wiki/AdventureWorks
7. Java Swing Tutorial - https://www.youtube.com/watch?v=HXV3zeQKqGY&t=6619s
8. jsplitpanel - https://stackoverflow.com/questions/1879091/jsplitpane-setdividerlocation-problem
9. Tabbed Panel - https://docs.oracle.com/javase/tutorial/uiswing/components/tabbedpane.html
10. Nested combination layout - https://stackoverflow.com/questions/5621338/how-to-add-jtable-in-jpanel-with-null-layout#:~:text=You%20can%20make%20use%20of,To%20add%20JTable%20to%20JPanel.&text=JPanel%20panel%20%3D%20new%20JPanel()%3B,add(scrollPane%2C%20BorderLayout.
11. Result set to Array - https://stackoverflow.com/questions/20021139/converting-resultset-to-multidimensional-string-array
12. Implementation of the close button in JPanel - https://stackoverflow.com/questions/24634047/closeable-jtabbedpane-alignment-of-the-close-button
13. Xchart implementation - https://knowm.org/open-source/xchart/xchart-example-code/

Appendix A: Record of Changes

Appendix A documents a table of major changes during the development process.

Table 1 - Record of Changes

MileStone Mark	Date	Author/Owner	Description of Change
MS3	*10/03/2020*	*Zengxiaoran Kang*	*Extract JDBC Terminal class out from the JDBC class and elevate an upper level of abstraction on the Backend Development*
MS3	*10/05/2020*	*Yifei Liang*	*Modify the return type and parameters of all functions in the JDBC class in order to satisfy the new level of abstraction*
MS4	*10/14/2020*	*Zengxiaoran Kang & Yuqi Sun*	*Rewrite and format the original design dococument in a professional setting*