**PROJECT PART II**

**Course:** Principles of Database Systems

**Section:** CS-GY 6083 – INET SUMMER 2022

**Submission Date:** 26th August 2022

**Team members:**

1. **Niko Ganev (ng2451)**
2. **Bhargav Makwana (bm3125)**
3. **Jash Merchant (jjm9801)**

**Table of Contents**

1. Summary
2. Technology Stack
3. Logical and Relational ERD
4. Data tables
5. Demo
6. Features
7. Lessons learned
8. SQL queries and results

**Summary**

In this project we implemented a new information system for the insurance company **We Do Secure (WDS).** This system is based on a relational database with 12 tables which securely stores the data for users and admins and allow users to apply for insurance policies as well as pay their invoices.

The goal of our application is to increase the number of users reached by WDS through an online portal that can be found online by anyone looking for insurance and convert these potential users into actual paying customers thanks to the easy process and appealing User Interface.

Besides the increase in the number of applications and general awareness of the WDS brand, our solution aims at differentiating WDS from their competition by guaranteeing data security as well as improving worker productivity through simple workflows for both customers and insurance agents.

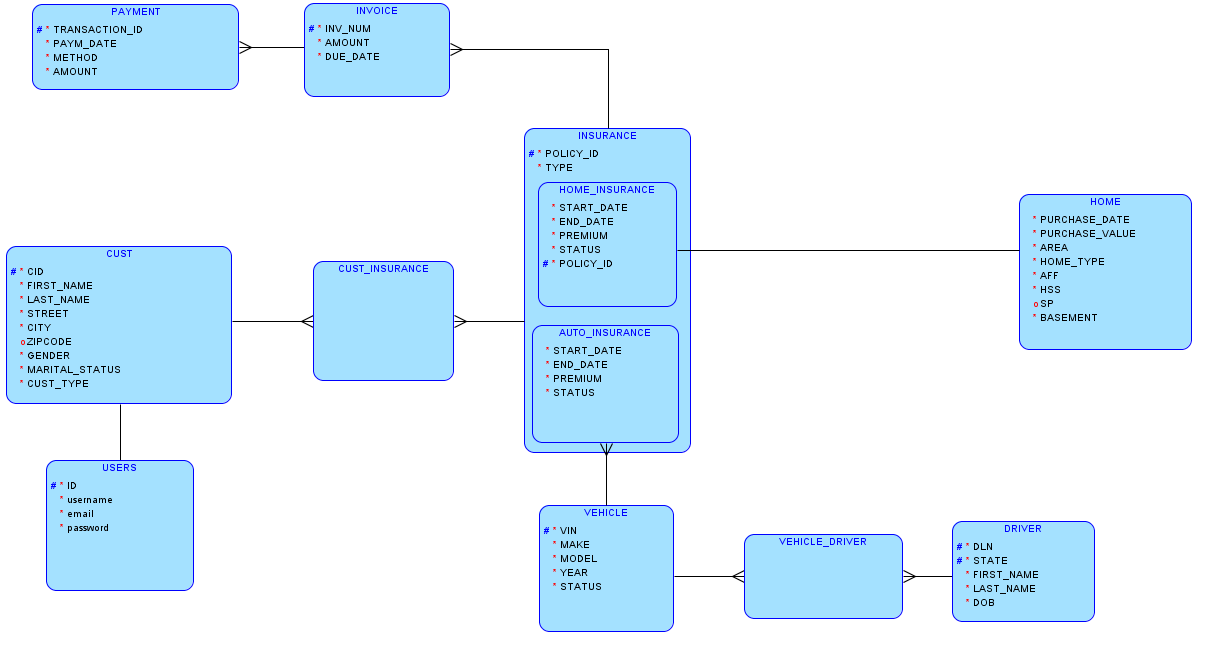
The admin panel allows the insurance agents to monitor their own performance and manage their clients effortlessly empowering an optimal experience for all parties.

**Technology Stack**

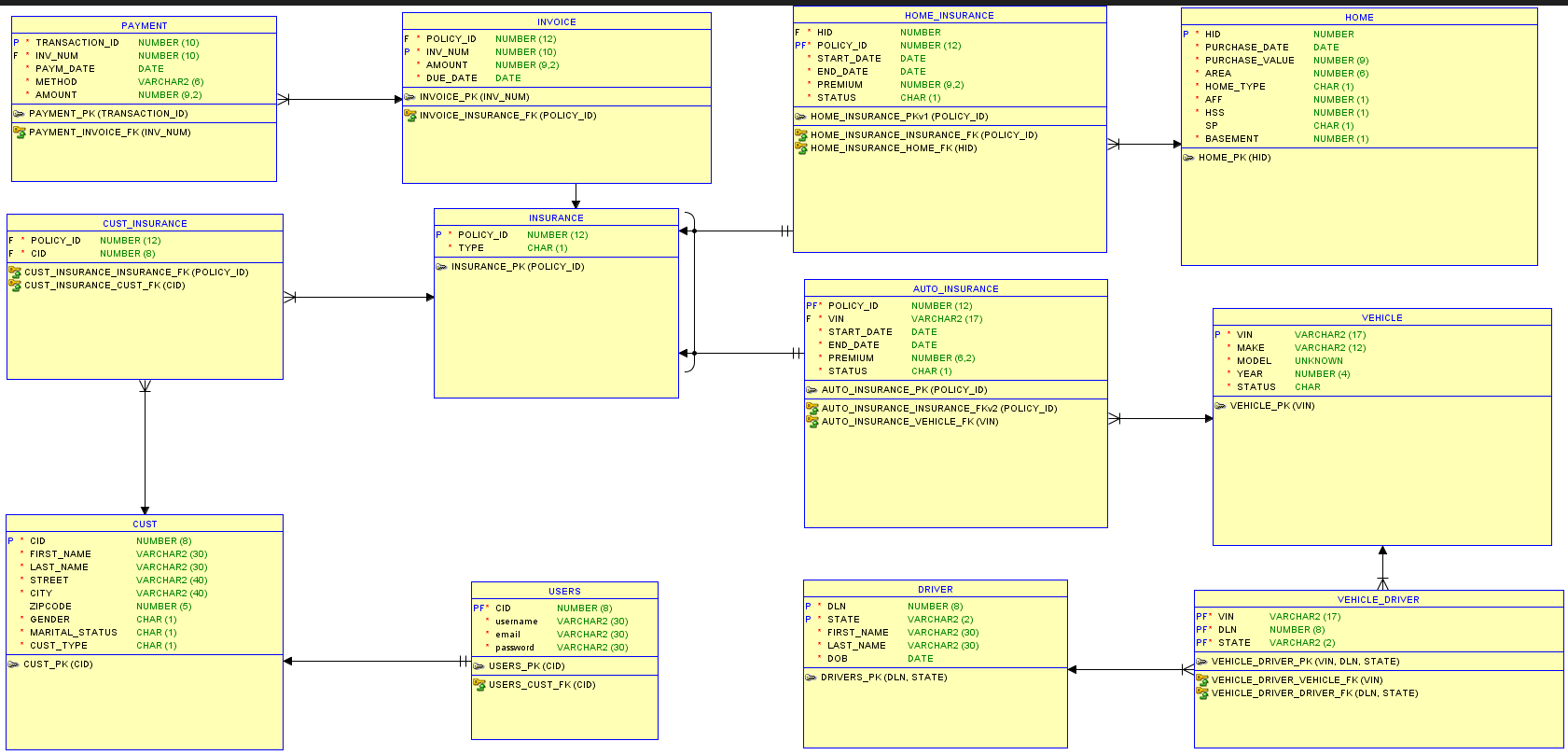
Ours is a mobile-responsive full-stack web application developed for users that want to enrol for Home and Insurance policies, using HTML, CSS, Bootstrap and JavaScript on the frontend while, Python, Flask and sqlite3 on the backend, and is deployed on Heroku (<https://pds-project.herokuapp.com/login>)

**Logical and Relational ERD**

**Logical ERD**



**Relational ERD**

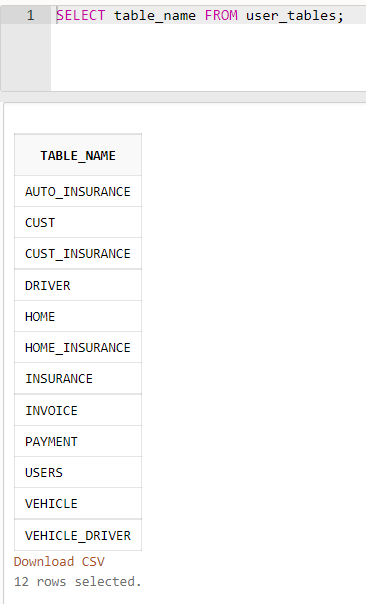


**Assumptions:**

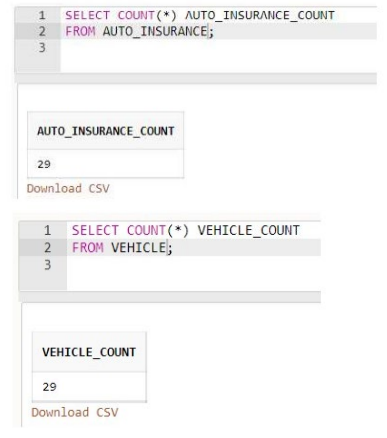
1. A customer can enrol in more than one insurance and an insurance contract can have more than one person on contract such as husband and wife and hence customer to insurance becomes a many to many relationship requiring an intersect table which we have named cust\_insurance.
2. Similarly, we have considered vehicle and driver to be a many to many relationship needing vehicle\_driver as an intersect table.
3. Insurance supertype possess two subtypes - Home insurance and Auto insurance.
4. A single insurance can have multiple invoices generated and each invoice could be paid in several installments.
5. Address is a composite attribute resolved into street, zip code and city.
6. Name is a composite attribute resolved into first and last name

**Data Tables**

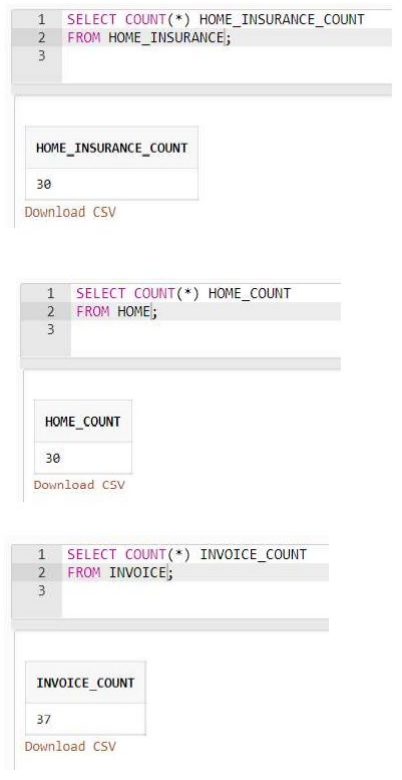
**Tables:**

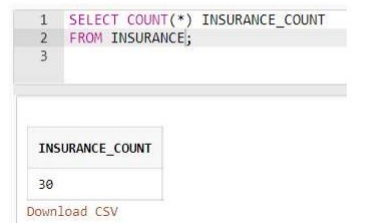


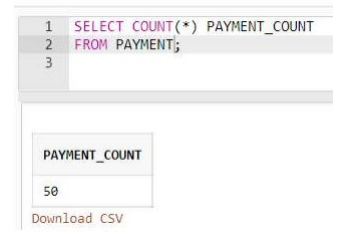
****

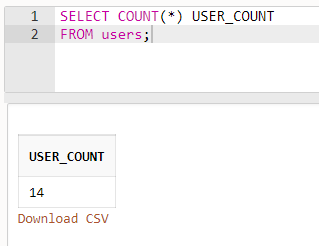
****

****

****

****

****

****

**Demo**

1. **Login**

**Graphical user interface, website

Description automatically generated**

1. Graphical user interface, website

   Description automatically generated**Registration**
2. **Home**

Graphical user interface, text, application

Description automatically generated

1. **Auto Insurance Form**

**Graphical user interface, application

Description automatically generated**

1. **Home Insurance Form**

**Graphical user interface, application

Description automatically generated**

1. **My Policies page**

**Graphical user interface

Description automatically generated**

1. **Admin Panel**

**Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated**

1. **Admin Panel (Customers List)**

**A screenshot of a computer

Description automatically generated**

1. **Forgot Password**

**Graphical user interface, application

Description automatically generated**

**Features**

* Users can perform **CRUD (C-Create, R-Read, U-Update, D-Delete)**
* SQL injection prevention using SQLAlchemy library
* Cross site scripting prevention
* Concurrency
* Mobile-responsive UI
* User registration **(C)**
* Password encryption using bcrypt library **(C)**
* Home Insurance application **(C)**
* Auto Insurance application **(C)**
* Frontend and backend error validations **(R)**
* Allow only unique usernames **(R)**
* User login **(R)**
* Graph to display number of users registered each day **(R)**
* My policy page for users to view their respective policies and pay invoice **(R)**
* Admin panel to view users list **(R)**
* Customers section under admin panel to view users who successfully converted to customers by enrolling in a policy **(R)**
* Reset password feature by emailing a unique link **(U)**
* Delete users and corresponding customer entries **(D)**
* Deployed to Heroku (<https://pds-project.herokuapp.com/login>)

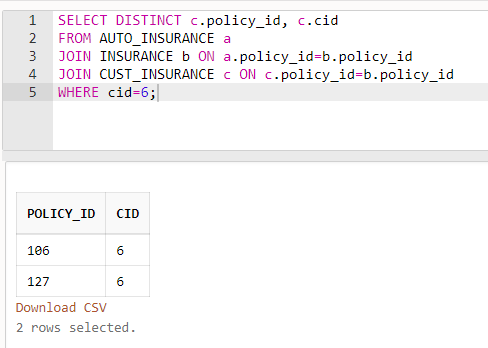
**Lessons learned**

In the span of the entire project, that is from understanding the Business case to the part of developing a prototype application we learned a lot of useful things jotted down below:

* Understanding the Business requirements and developing an ER model based on it has its own challenges, like what kind of relationship best serves the purpose (1:M, M:N, etc.).
* We enjoyed and the team building and coordination process, realized the importance of a team and its members.
* We learned that implementing an actual application from designing schema is actually fascinating. While it has its own challenges pertaining to the tech stack used, it also helps one to learn the tech skills as well has improve debugging skills.
* The project timeline and deadline play an important role in restricting/scaling the scope of the application.
* I learned how to use Github effectively and realized how important it is while working on a team project.
* I believe we could have done even better if we started working on the project a little early with same seriousness as there were a lot of other things that we wanted to implement and improve but due to time constraints we couldn’t.
* Got to learn a lot too from the teammates, each one of us had different speciality so we focussed on that and worked for a common goal and fortunately things went well.

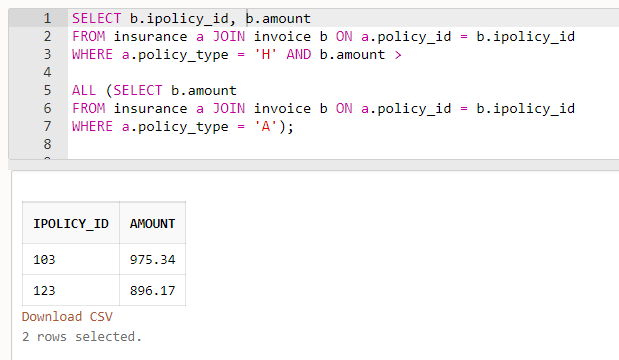
**SQL Queries and results**

**Q1)**

****

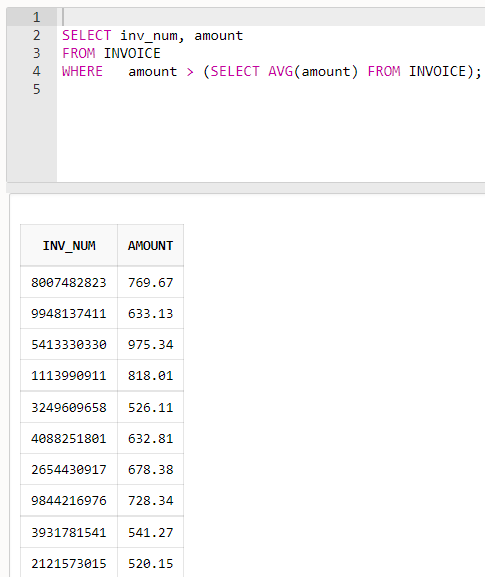
This query returns all insurance policies for customer with id 6.

**Q2)**

****

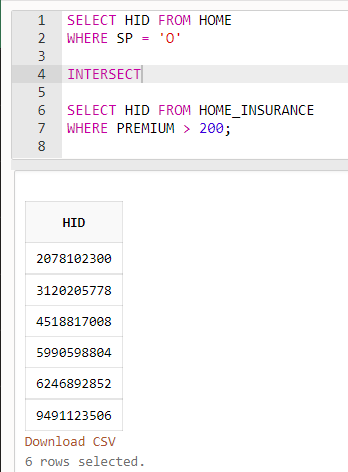
This query returns all home insurance policies that are more expensive than all of the auto insurance policies.

**Q3**



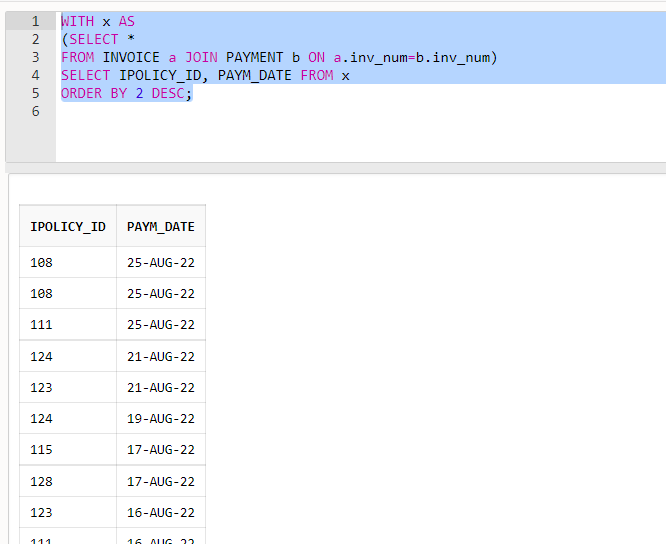
This query returns all invoices that are above the average invoice amount.

**Q4)**

****

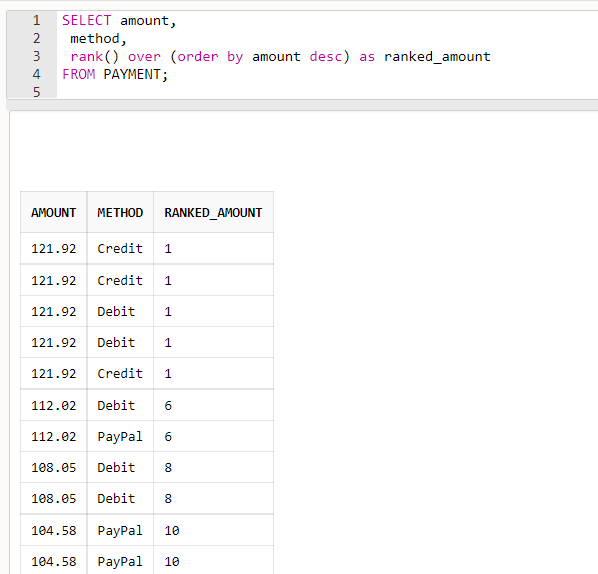
This query returns the ids of all homes with an outdoor pool paying more than 200 dollars in home insurance premium.

**Q5)**

****

This query returns the IDs and payment date of policies which received a payment the most recently.

**Q6)**

****

This insurance returns the top payment amounts along with the method of payment allowing to look for corelation between payment amount and method.