Names: Eugene Chang, Sanho Lee

NYU IDs: N17404284, N15250101

Course Section Number: CSCI-GA.2433-001 - Fall 2023

Date: 12/19/2023

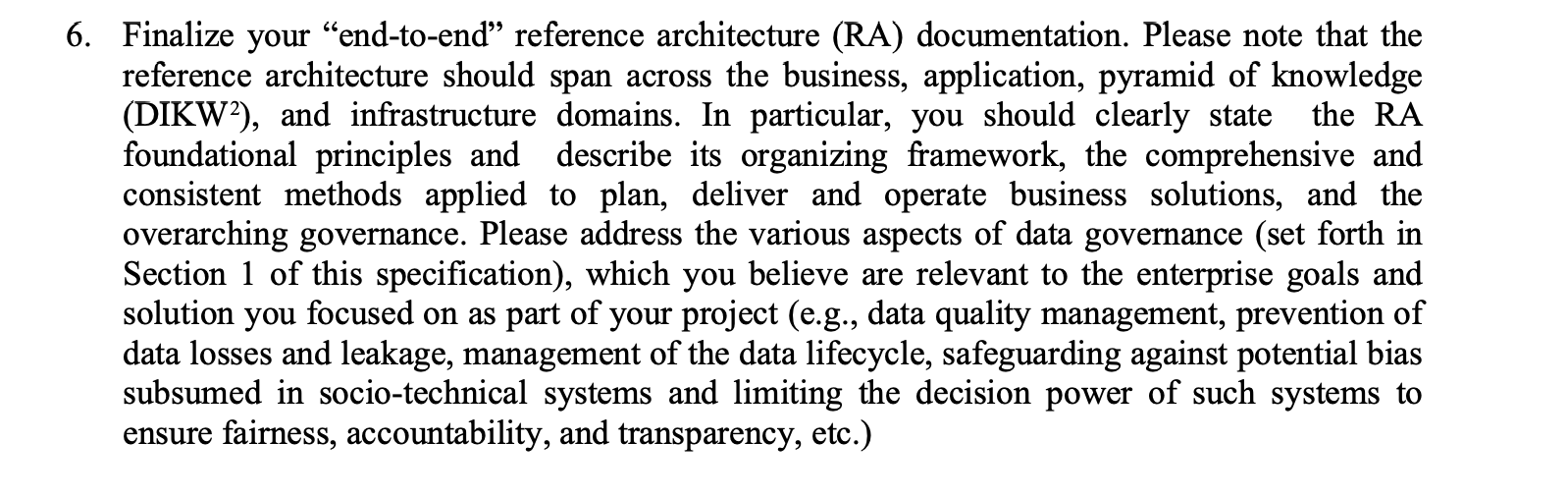
**EDA Project Part 4**

**Goals for Part 4:**

* Solidify machine learning algorithm and learning method
  + Supervised Learning
    - We have to create sample data manually that is “accurate” to train on
  + Unsupervised
    - Really hard since dataset is not realistic
* Create a basic website based on the User Case Diagram we mentioned above
  + Takes in user input as a form
  + Sends request to the server that runs the Machine Learning algorithm
    - Server is connected to Azure SQL server
  + Returns a response with a predicted insurance cost for this user
* Publish everything on GitHub at the end
  + Gif demo of the app
  + Instructions on how to run server and website on local host

Website Application:

* Framework: Flask
  + Lightweight and efficient
  + Good for a simple web application like this one
  + Python → able to integrate machine learning model



Our EDA Model Description: **Car Insurance Company Model.**

**Entity-Relationship (ER) Model Documentation**

**Entity Types and Attributes:**

**COMPANY**

* CompanyCode (Primary Key)
* CompanyName
* Email
* Phone

**CUSTOMER**

* CustomerSSN (Primary key)
* CastLastName
* CustFirstName
* CustomerDOB
* CustomerGender
* Email
* Phone
* CustomerLisc

**VEHICLE**

* VIN (Primary key)
* Brand
* Model
* Year
* LicensePlateNumber
* Mileage
* VehicleType
* CustomerSSN (Foreign Key)

**CONTRACT**

* ContractID (Primary key)
* CoverageType (plan)
* MaximumCoverage
* StartDate
* EndDate
* MonthlyPrice
* CompanyID (Foreign key)
* VIN (Foreign Key)
* CustomerSSN (Foreign key)
* AgentsSSN ( Foreign key)

**CLAIM**

* ClaimID (Primary key)
* AccidentDate
* AccidentDesc
* ClaimAmount
* Status
* CustomerSSN ( Foreign key)
* ContractID ( Foreign key)

**AGENT**

* AgentSSN (Primary key)
* CompanyCode(Foreign key)
* AgentLastName
* AgentFirstName
* AgentDOB
* Email
* Phone

**PAYMENT**

* PaymentID
* PaymentAmount
* PaymentMethod
* DateOfPayment
* CustomerSSN (Foreign key)
* ContractID ( Foreign key)

**ADDRESS (weak)**

* AddressLine1 (Primary key)
* Zip (Primary key)
* CustomerSSN (Foreign key)
* AgentSSN (Foreign key)
* AddressLine2
* State
* City

**DRIVING\_HISTORY (weak)**

* CustomerSSN ( Primary Key, Foreign Key)
* TrafficViolations
* Accidents
* DrivingExperience

**PROFILE (Used for Machine Learning Algorithm)**

**Relationship Types and Attributes:**

COMPANY\_CONTRACT : MANAGE

* Cardinality ratio:
  + One-to-many (1:N)
  + COMPANY participation (0,N)
  + CONTRACT participation (1,1)
* Attributes:
  + CompanyID
  + ContractID
* Explanation:
  + Each company can offer multiple insurance contracts, but they can also offer no contracts.
  + One contract must be linked to a company in order to exist

COMPANY\_AGENT : EMPLOY

* Cardinality ratio:
  + One-to-many(1:N)
  + Company participation (0,N)
  + Agent participation (1,1)
* Attributes:
  + CompanyCode
  + AgentSSN
* Explanation
  + Companies may employ multiple agents
  + Each agent must only be employed by one company

CUSTOMER\_CONTRACT : BUY

* Cardinality ratio:
  + One-to-many (1:N)
  + CUSTOMER participation (1, N)
  + CONTRACT participation (1, 1)
* Attributes:
  + CustomerSSN
  + ContractID
* Explanation:
  + Each customer may purchase multiple contracts but must purchase at least one contract to be a customer
  + Each contract can only be assigned to one customer

CUSTOMER\_CLAIM : FILES

* Cardinality ratio:
  + One-to many (1:N)
  + CUSTOMER participation (0, N)
  + CLAIM participation (1,1)
* Attributes:
  + CustomerSSN
  + ClaimID
* Explanation:
  + Customers may file zero to multiple claims
  + Each claim can only be filed by one customer

CUSTOMER\_PAYMENT : PAYS

* Cardinality ratio:
  + One-to-many(1:N)
  + CUSTOMER participation (1,N)
  + PAYMENT participation(1,1)
* Attributes:
  + CustomerSSN
  + PAYMENTID
* Explanation:
  + A customer must have a payment, but can have multiple payments.
  + A payment should be assigned to one customer.

CUSTOMER\_ADDRESS : LIVES

* Cardinality ratio:
  + One-to-many(1:N)
  + CUSTOMER participation (1,N)
  + ADDRESS participation(1,1)
* Attributes:
  + CustomerSSN
  + AddressLine1
  + ZIP
* Explanation:
  + A customer must have an address, but can have multiple addresses.
  + An address should be assigned to one customer.

CUSTOMER\_VEHICLE: OWN

* Cardinality ratio:
  + One-to-many(1:N)
  + CUSTOMER participation ( 0,N)
  + VEHICLE participation(1,1)
* Attributes:
  + CustomerSSN
  + VIN
* Explanation:
  + A customer can have multiple vehicles, but doesn’t matter if they have none.
  + A vehicle must be assigned to one customer.

CUSTOMER\_DRIVING\_HISTORY: HISTORY\_OF ( weak)

* Cardinality ratio:
  + One-to-One(1:1)
  + CUSTOMER participation ( 0,1)
  + DRIVING\_HISTORY participation(1,1)
* Explanation:
  + A customer can have one Driving history.
  + A Driving history must be assigned to one customer.

VEHICLE\_CONTRACT : ASSIGNED\_TO

* Cardinality ratio:
  + One-to-one (1:1)
  + VEHICLE participation (1,1)
  + CONTRACT participation (1,1)
* Attributes:
  + VIN
  + ContractID
* Explanation:
  + Each vehicle can only be assigned to one insurance contract
  + Each contract can only cover one vehicle

CONTRACT\_CLAIM : ASSOCIATED\_WITH

* Cardinality ratio:
  + One-to-many (1:N)
  + CONTRACT participation (0,N)
  + CLAIM participation (1,1)
* Attributes:
  + ContractID
  + ClaimID
* Explanation:
  + Each contract can be associated with zero to multiple claims
  + Each claim can only be linked to a single contract

AGENT\_CONTRACT : SELL

* Cardinality ratio:
  + One-to-many (1:N)
  + AGENT participation (0,N)
  + CONTRACT participation ( 1,1)
* Attributes:
  + AgentSSN
  + ContractID
* Explanation:
  + Each Agent can sell many contracts, but it does not matter if they sell none.
  + Each contract can only cover one agent.

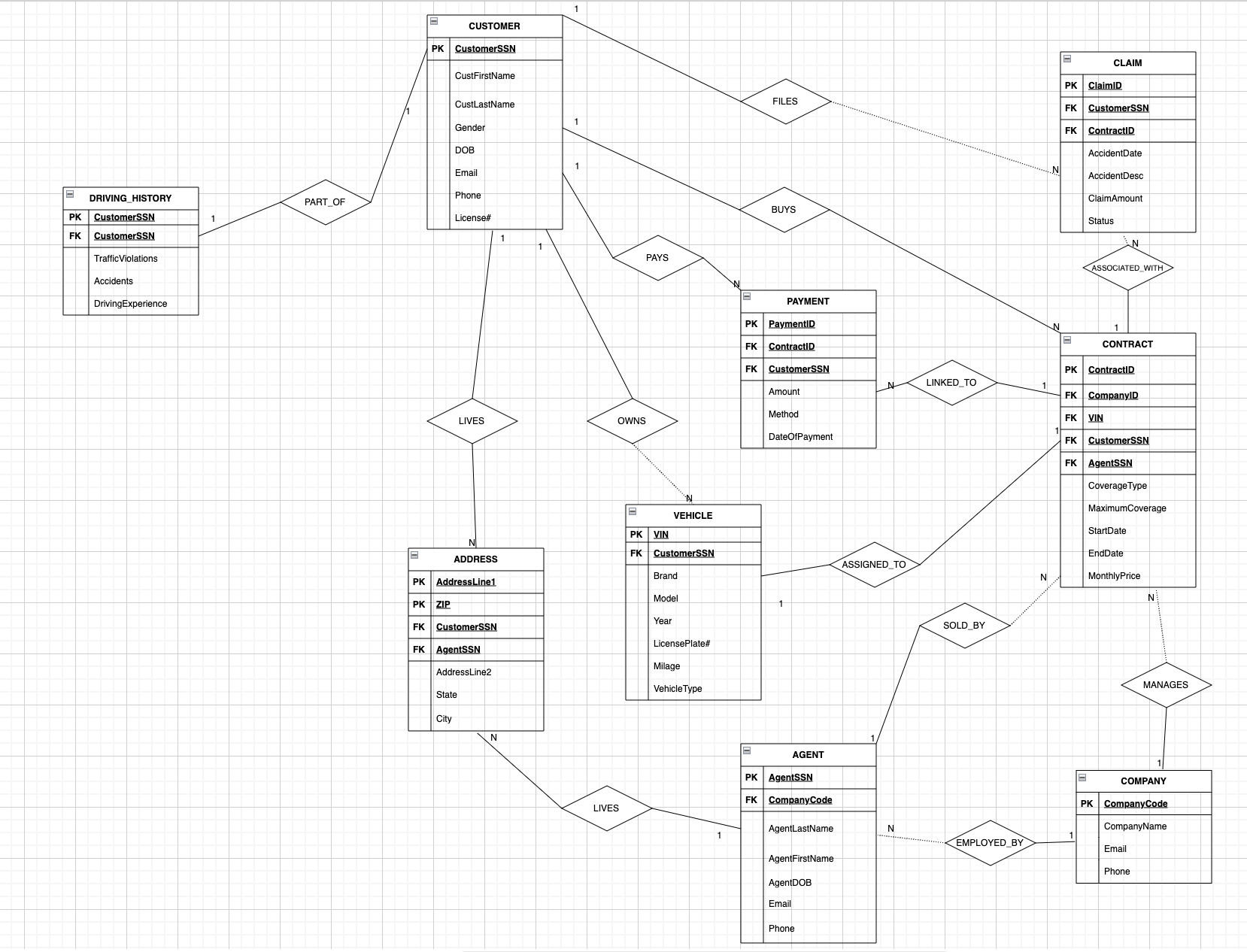
AGENT\_ADDRESS : LIVES

* Cardinality ratio:
  + One-to-many (1:N)
  + AGENT participation (1,N)
  + ADDRESS participation (1,1)
* Attributes:
  + AgentSSN
  + AddressLine1
  + ZIP
* Explanation:
  + An agent must have an address, but can have multiple addresses.
  + An address should be assigned to one agent.

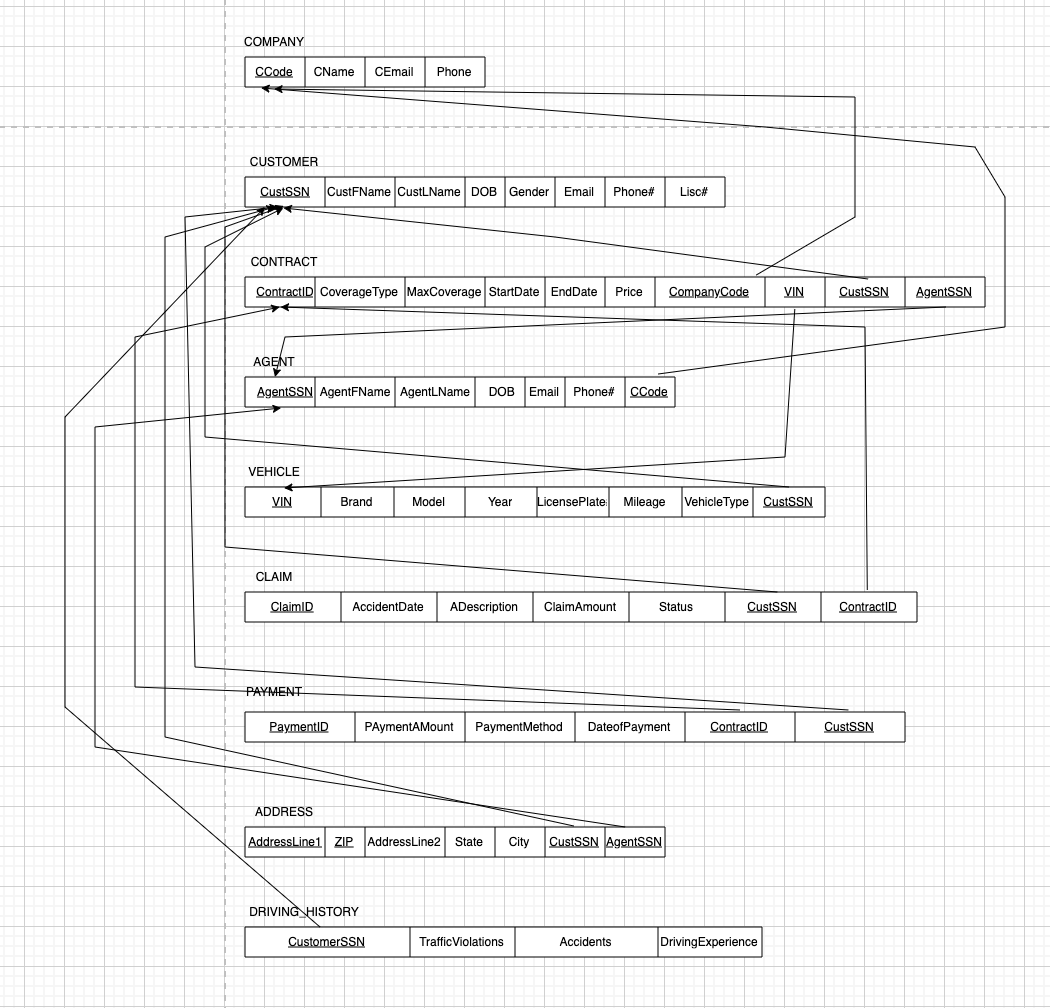
CONTRACT\_PAYMENT : LINKED\_TO

* Cardinality:
  + One-to-many (1:N)
  + PAYMENT participation (1,1)
  + CONTRACT participation (1,N)
* Attributes:
  + PaymentID
  + ContractID
* Explanation:
  + Each payment can only pay for one contract
  + Contracts can be paid by multiple payments over the course of its term

**ER - Diagram**

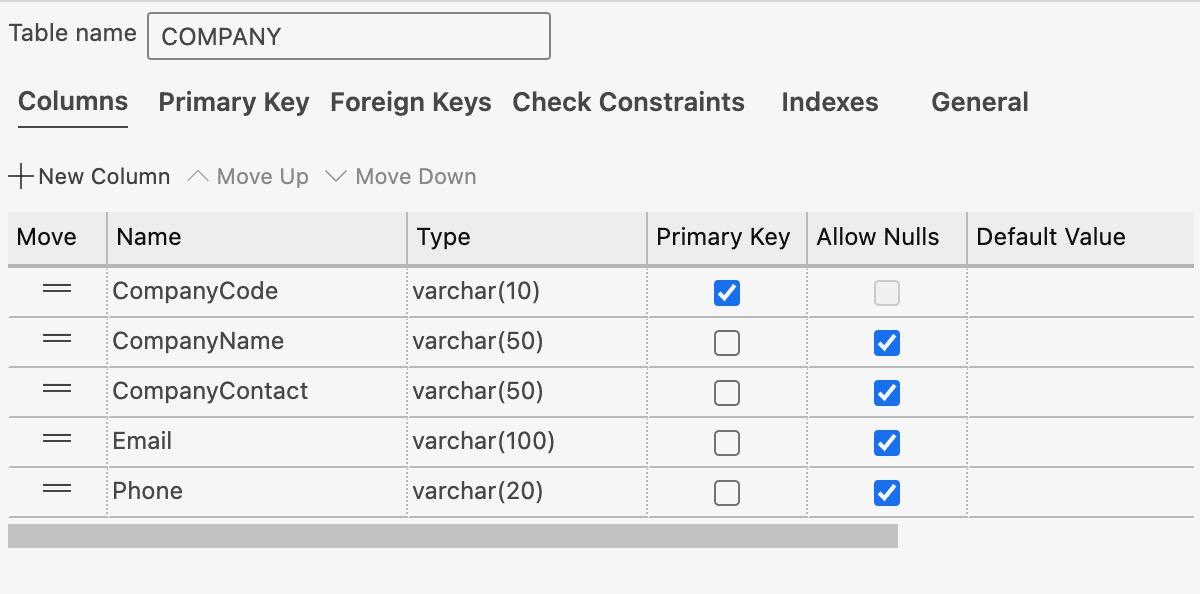


**Relational Schema Mapping**

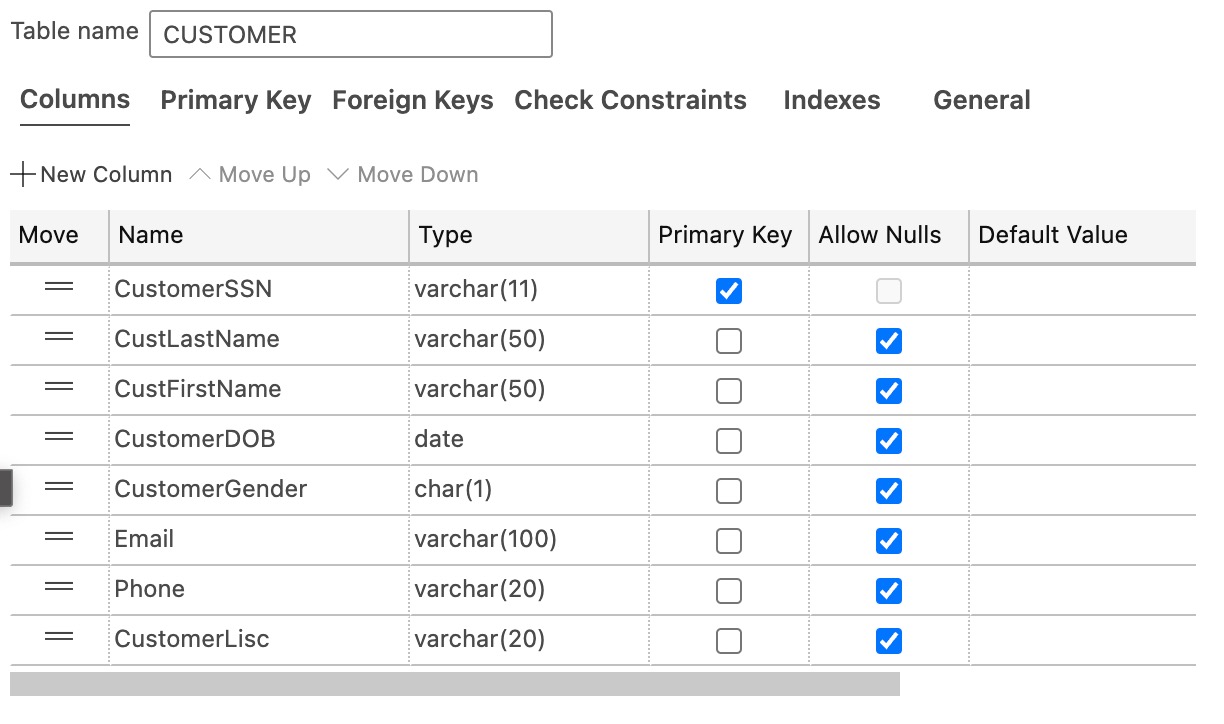
****

**Physical Model Design**

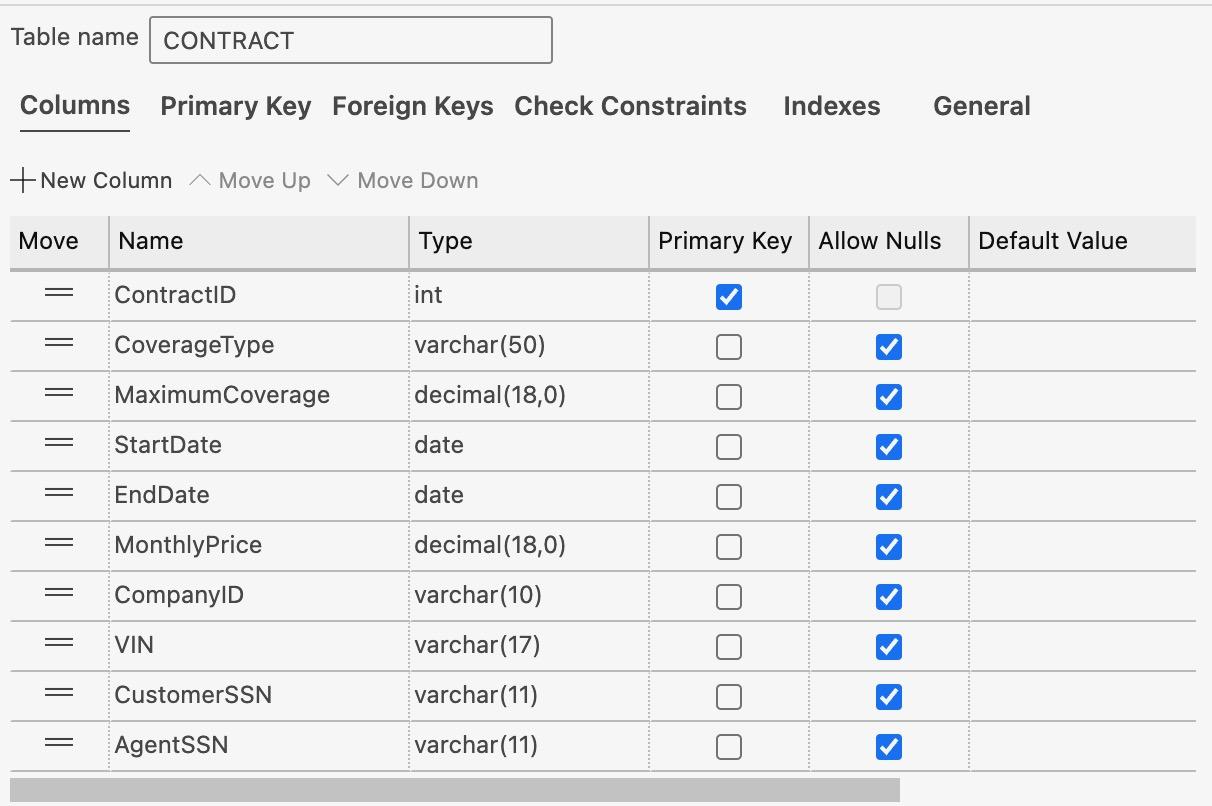
**TABLE COMPANY**

****

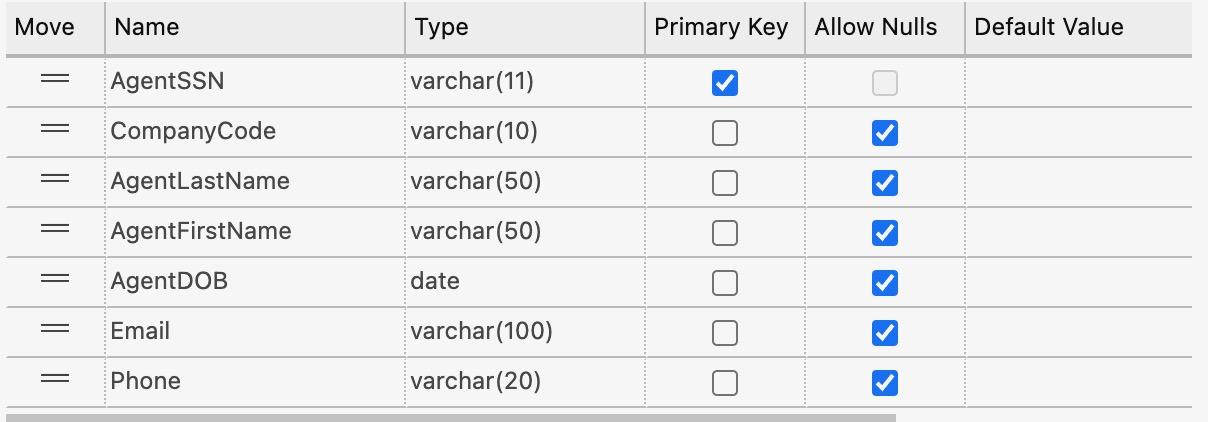
**TABLE CUSTOMER**

****

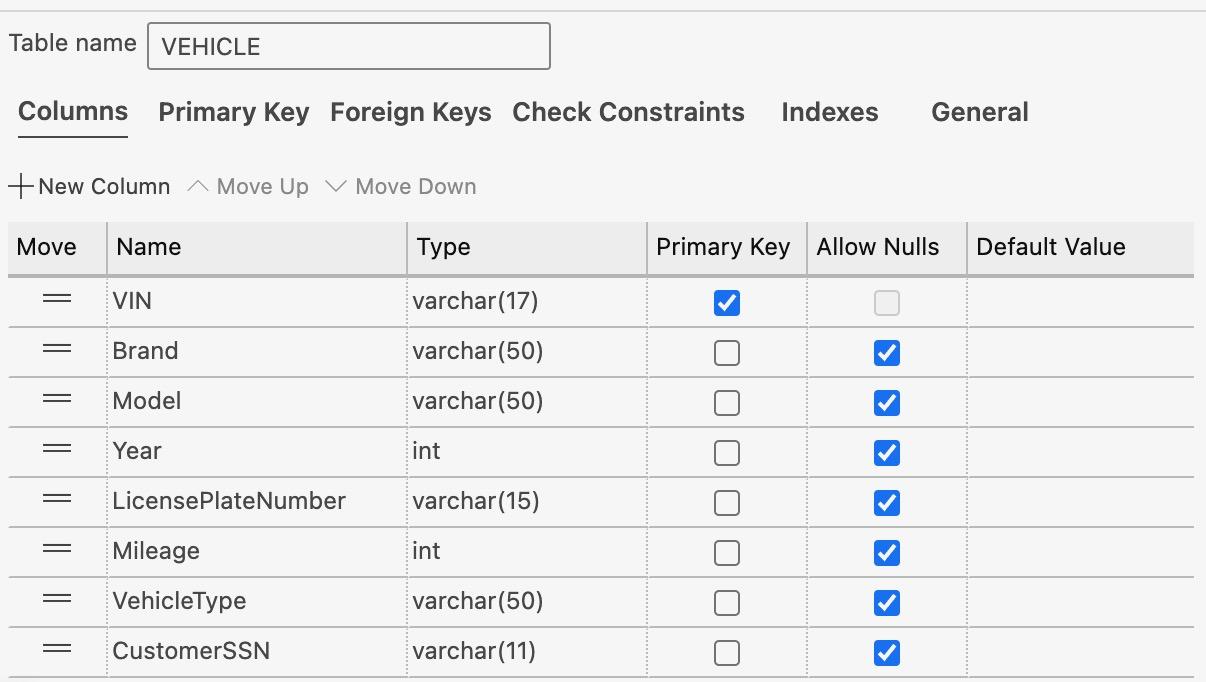
**TABLE CONTRACT**

****

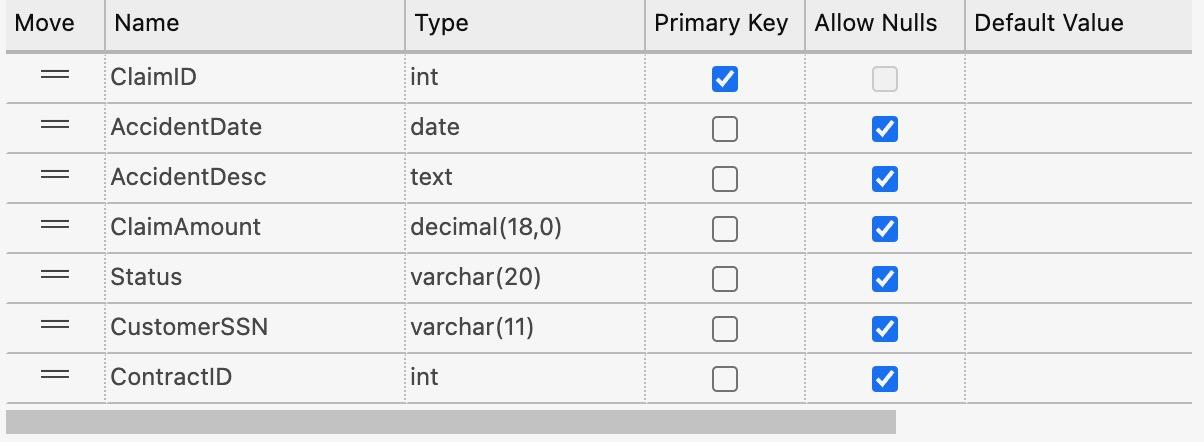
**TABLE AGENT**

****

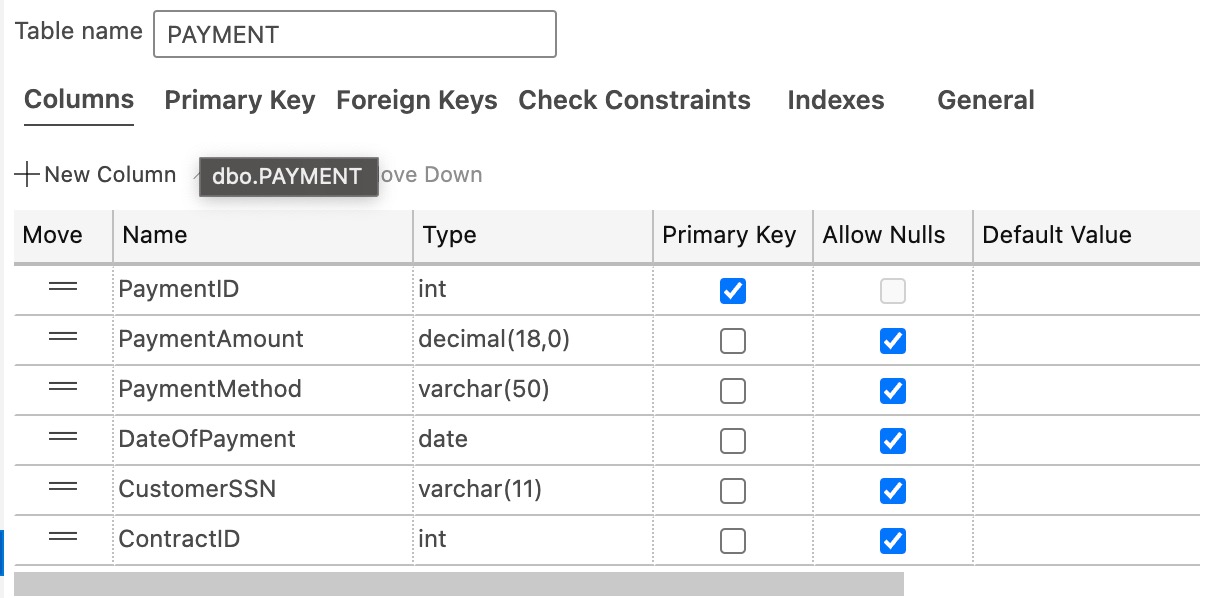
**TABLE VEHICLE**

****

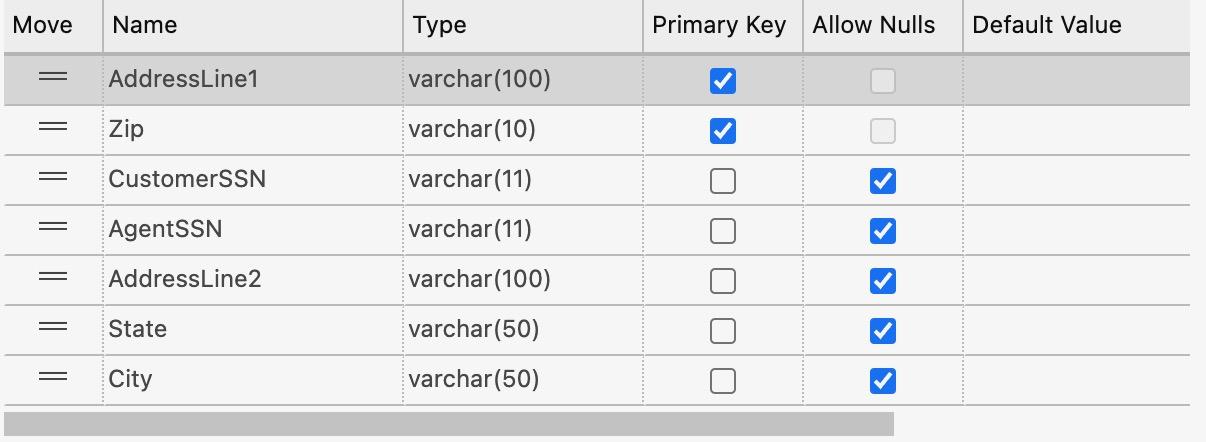
**TABLE CLAIM**

****

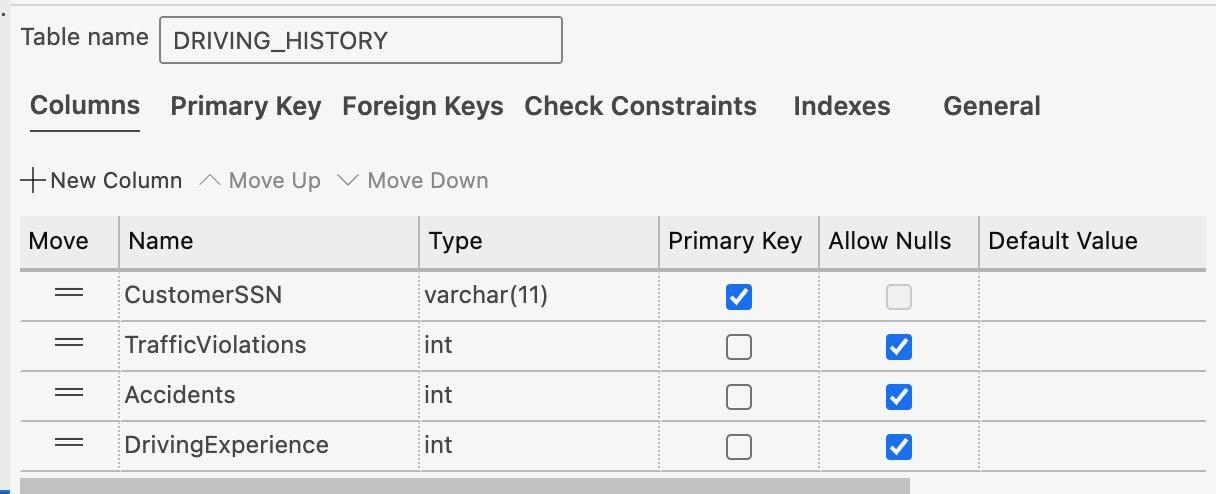
**TABLE PAYMENT**

****

**TABLE ADDRESS**

****

**TABLE DRIVING\_HISTORY**

****

Now, let me show you how I leveraged this data to implement a machine learning algorithm. We designed a machine learning model using the XGBoost algorithm, aimed at predicting the monthly premium for insurance customers based on various factors.

In the development of our predictive model for car insurance premiums using the XGBoost algorithm, a critical step involved the creation of a robust and representative dataset. Given the necessity for a comprehensive set of data to effectively train a supervised learning model, we employed Mockaroo, a sophisticated data generation tool, to populate our dataset with mock data. This approach enabled us to simulate a wide array of scenarios and customer profiles, thereby enhancing the diversity and richness of our training data.

To align our dataset with the requirements of a supervised learning system, we focused on ensuring that our target variable, the monthly insurance premium, was accurately represented and calculated based on various influencing factors. To achieve this, we implemented a SQL stored procedure, CalculatePremiums, designed to compute the monthly premium for each profile in our database.

This procedure was meticulously crafted to reflect realistic insurance premium calculation practices. It initiates by setting a base premium amount, to which adjustments are made based on several key factors:

* Accident History: An additional charge of $30 is applied for each recorded accident, recognizing the increased risk associated with customers who have a history of accidents.
* Traffic Violations: Traffic violations also contribute to the premium calculation, with $15 added for each violation, acknowledging their significance in risk assessment.
* Age Consideration: The age of the customer plays a pivotal role, with an extra $25 added for customers either below 25 years or above 65 years of age. This adjustment is based on the statistical risk variance associated with these age groups.
* Driving Experience: Driving experience is factored in to reward safer, more experienced drivers. For every year of driving experience beyond five years, a deduction of $10 is made from the premium.
* Mileage: Lastly, the annual mileage is considered, with the premium increasing proportionally to the distance driven, at a rate of $1 for every 1,000 miles.

However, since the Premium Fee could go down to negative values due to the factor of Driving Experience, we set the minimum value to 25 dollars to avoid negative values.

This procedure is executed annually to update the MonthlyPremium attribute in the Profile table, ensuring that our dataset remains dynamic and reflective of current risk assessments.

**Here's an overview of the steps and concepts involved:**

1. Data Loading and Preprocessing:

Data Loading: We loaded insurance data from a CSV file (profile.csv). This dataset includes information about customers and their driving history.

Feature Selection: We selected relevant features for predicting monthly premiums: Age, CustomerGender, Mileage, TrafficViolations, Accidents, and DrivingExperience.

Data Transformation: The date of birth (DOB) was converted into Age by calculating the difference between the current year and the year of birth. This transformation makes the feature more relevant and easier to use in prediction.

Categorical and Numerical Processing: We used StandardScaler to normalize numerical features and OneHotEncoder to encode categorical features. This step is crucial to handle different types of data and make them suitable for the machine learning model.

2. Model Selection and Training:

XGBoost Algorithm: We chose the XGBoost regressor for this task. XGBoost stands for eXtreme Gradient Boosting, a powerful and efficient implementation of gradient boosted decision trees designed for speed and performance.

Pipeline Creation: A pipeline was created to streamline the preprocessing and modeling process. This pipeline includes the preprocessing steps and the XGBoost model.

Data Splitting: The data was split into training and testing sets to train the model and evaluate its performance.

3. Hyperparameter Tuning:

GridSearchCV: To optimize the model, we used GridSearchCV, which systematically works through multiple combinations of parameter tunes, cross-validating as it goes to determine which parameters yield the best performance.

Parameter Grid: A set of parameters (n\_estimators, max\_depth, learning\_rate) was defined for the XGBoost model. GridSearchCV explored different combinations of these parameters to find the best configuration.

4. Model Evaluation:

Mean Squared Error (MSE): The model's performance was evaluated using MSE, a common regression metric that measures the average squared difference between the predicted and actual values.

Actual vs. Predicted Comparison: We printed the actual and predicted premiums for the first 10 instances from the test set for a quick evaluation.

5. User Interaction for Predictions:

Prediction Function: We developed a function, predict\_premium, which prompts the user to input feature values for a new customer and uses the trained model to predict their monthly premium.

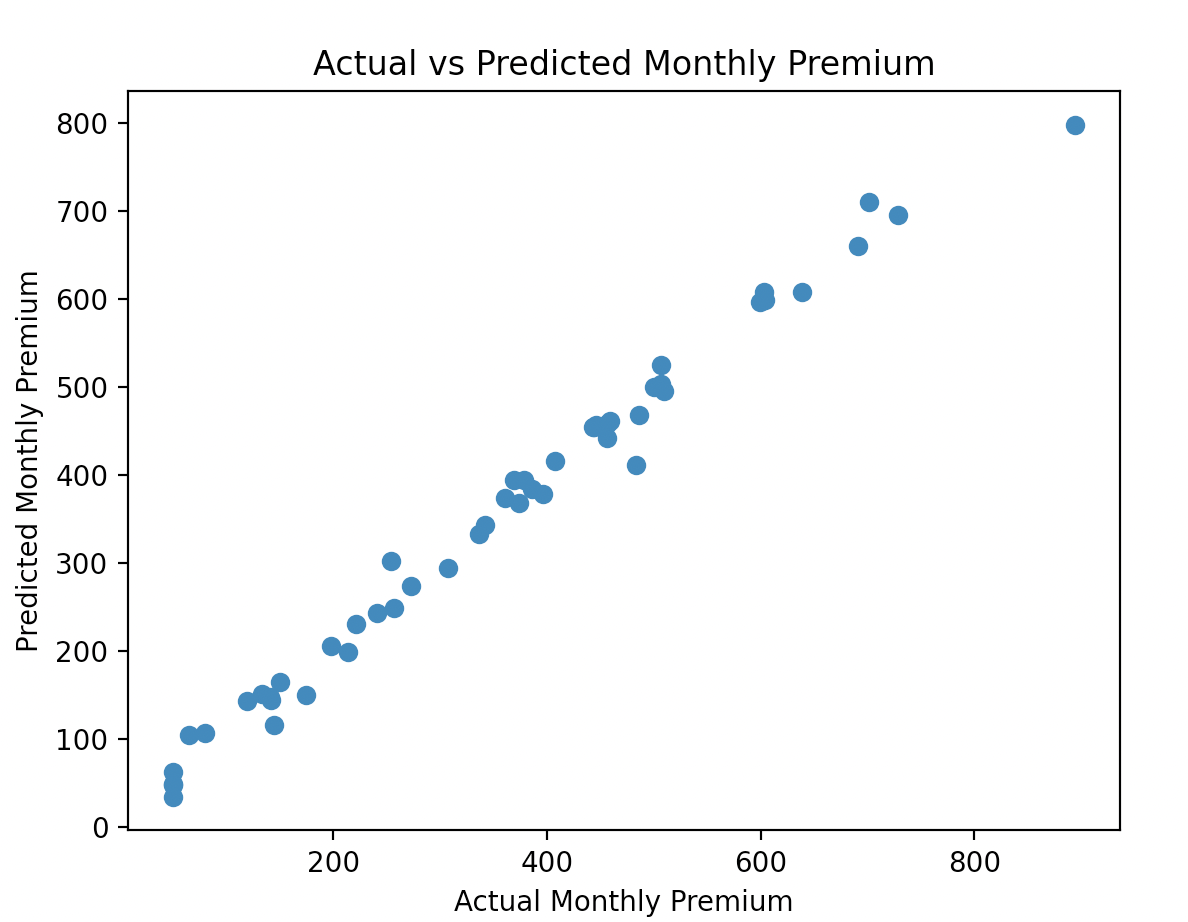
Interactive Prediction: This function allows for real-time interaction and demonstrates how the model can be applied in a practical, user-facing scenario.

6. Machine Learning Algorithm - XGBoost:

Gradient Boosting Framework: XGBoost is part of the gradient boosting family of machine learning algorithms. It builds multiple decision trees sequentially, where each tree tries to correct the errors made by the previous ones.

Strengths of XGBoost:

* Handles a wide range of data types, relationships, and distributions.
* Highly efficient, scalable, and works well with large datasets.
* Provides regularization options to prevent overfitting.
* Often delivers superior performance compared to other algorithms.



**Final Summary:**

This model, leveraging XGBoost and a robust preprocessing pipeline, exemplifies a practical application of machine learning in insurance premium prediction. It's important to remember that the model's accuracy heavily relies on the quality and representativeness of the training data, and further tuning and validation might be necessary for production use.

In our quest to redefine the car insurance landscape, our vision is firmly rooted in becoming a market leader driven by technological innovation and customer-centricity. At the heart of this vision lies the commitment to deliver personalized and competitively priced insurance policies, achieved through the cutting-edge application of machine learning technologies, particularly the **XGBoost** algorithm. This approach not only optimizes risk assessment but also significantly enhances customer experience and operational efficiency.

The guiding principles of our endeavor revolve around four key pillars: unwavering customer-centricity, robust innovation in data utilization, steadfast transparency and integrity, and the agility to embrace technological advancements. These principles are not just ideals but are actively manifested in every facet of our business operations and strategic decisions.

Our policies and guidelines are meticulously crafted to uphold data security and privacy, complying with stringent regulations. We recognize the profound impact of AI in our industry, which is why responsible usage of XGBoost and other AI technologies is at the forefront of our ethical considerations. We are committed to a paradigm of continuous improvement, constantly refining our models and strategies in light of new data, customer feedback, and evolving market trends.

The project-specific guidelines for our car insurance model emphasize the innovative application of the XGBoost algorithm for precise premium prediction. This is complemented by a user-friendly digital platform, facilitating effortless policy management for our customers. Integration with existing systems is seamless, ensuring a unified and efficient operational workflow. Importantly, scalability and performance monitoring are integral, ensuring that our solutions not only meet current demands but are also future-proof.

Looking ahead, we envision our architecture's adaptability to extend beyond car insurance, encompassing other insurance products and catering to diverse global markets. We are keen on incorporating emerging technologies like IoT for enhanced real-time data collection and blockchain for secure, transparent transactions.

In essence, the alignment of our car insurance model with this visionary architecture is a testament to our commitment to revolutionize insurance offerings. The XGBoost machine learning model is a cornerstone of this alignment, embodying our principles of data-driven decision-making and innovation. This architecture is not just a blueprint for current endeavors but is a versatile and dynamic foundation for future growth and technological integration.The project's core concepts, rooted in leveraging machine learning for insightful data analysis and decision-making, have broad applicability across various sectors. The future expansion of these ideas can lead to significant advancements in operational efficiency, risk management, customer experience, and strategic planning in multiple industries.

Our Website Descriptions and demonstrations will be on the github link.

<https://github.com/egnechng/DB-FinalProject>