Name: Zhang, Zejun Date: 11/22/2023

NYU ID: zz4140

Course Section Number: CSCI-GA-2433

Project Part 2

**Total in points** (100 points total):

**Professor’s Comments**:

**The modification based on Part I:**

1. The relation between ContractPremium and its two parts: Associate Commission and Production Credit. These two parts were two entities, and I changed them to two attributes of the ContractPremium, this is more appropriate way to understand the business logic, since these two attributes are not related to any other entities.
2. The Serve relation was an n…n relation in the part one, and I changed it to 1…n relation. This simplified business logic, and lower the risks of data inconsistency in the future.
3. The Benefit relation was n…n relation in the part one, and it is canceled in this part, since this part does not fit in the business logics.

**The data lake:**

1. A data lake is a storage solution that allows organizations to store large amounts of structured, semi-structured, and unstructured data. The main advantage of a data lake over a traditional data warehouse is its flexibility and scalability. It can store raw data without pre-defined data models, thus giving data analysts the flexibility to query and analyze data.
2. Key features of a data lake:
   1. Flexibility: The ability to store various types of data.
   2. Scalability: Easy to scale as the amount of data increases.
   3. Low cost: Usually based on low cost storage solutions.
3. This business database can be good to fit into a data lake, since there are different kind of data within this insurance company’s database. For instance, we have structured data like customer’s basic information, and we also have some unstructured data like the pictures of invoice detail and claim detail.
4. In this data lake, I use relational database system to manage all the data. All those unstructured data was stored on cloud, and they are stored in the RDBMS in the form of URL.
5. Since we do not have essential data to start the project, so I faked some data based on the instruction of Professor Franchitti in the lecture. The faked data was generated from Mockaroo.com, which is a website that can be used to generated various random data. However, this data-generating tool does not guarantee unique data, so I applied necessary data cleaning processes for all the primary keys. Then, based on the schema (figure 1), import all the foreign keys into the database.
6. Future data: Based on the database I created now, all the structured data we might generate in the future can be easily added into the .csv file we currently have. For those unstructured data, we could upload to cloud and generate the URLs, then put them into corresponding .csv files.

A diagram of a company

Description automatically generated with medium confidenceFigure 1

**The Schema:**

1. Customer:

This entity stores basic information about customers.

Key attributes:

1. SSN: this is the id of each customer, and it’s primary key.
2. Account\_name: this is the foreign key from Account entity.

For the fake data of this part, I created 1000 rows different data.

1. Account:

This entity stores basic information about accounts.

Key attributes:

1. Account\_name: this is the id for each account, and it’s primary key.

This part of fake data contain 100 rows different accounts.

1. AccountMember:

This entity stores the information between account and customer, i.e. which account does each customer belong to.

Key attributes:

1. SSN: this is the foreign key from Customer entity.
2. Account\_name: this is the foreign key from Account entity.

Since there are 1000 different customers, there are 1000 entries within AccountMember now.

1. Claims:

This entity holds information about claims made by customers.

Key attributes:

1. Claim#: Claim number, a unique identifier.
2. SSN: this is the foreign key from Customer entity.

1000 rows data to start.

1. Invoice:

This entity is for invoices generated for payments.

Key attributes:

1. SSN: this is the foreign key from Customer entity.
2. Invoice#: Invoice number, a unique identifier.

1000 rows data to start.

1. BAccount Name:

This entity represents billing accounts.

Key attributes:

1. BAccount\_name: Name of the billing account. Primary key.

100 rows data

1. Account\_BillingAccount:

This is the relation between account and baccount

Key attributes:

1. BAccount\_name: this is the foreign key from Account\_BillingAccount entity.
2. Account\_name: this is the foreign key from Account entity.
3. Account\_Admin:

This entity represents administrators for accounts.

Key attributes:

1. Admin\_name: Name of the administrators. Primary key.

100 rows data

1. Acc\_acctadmin:

This is the relation between account and account\_admin.

Key attributes:

1. Admin\_name: this is the foreign key from Account\_Admin entity.
2. Account\_name: this is the foreign key from Account entity.
3. ManagerContract:

This is the information of each contract between customers and associats.

Key attributes:

1. Contract#: Contract number, a unique identifier.
2. Associate\_SSN: this is the foreign key from Associate entity.
3. Account\_name: this is the foreign key from Account entity.

1000 rows data

1. Contractbenefit:

This entity holds the benefits associated with contracts.

Key attributes:

1. Benefit\_name: Name of the benefit. Primary key.
2. Contract#: this is the foreign key from ManagerContract entity.
3. ContractPremium:

This entity represents premiums for each contract.

Key attributes:

1. PremiumCode: A unique code for the premium. Primary key.
2. Benefit\_name: this is the foreign key from Contractbenefit entity.
3. Associate:

This entity represents associates who may sell products or manage contracts.

Key attributes:

1. Associate\_SSN: Social Security Number of the associate.
2. Product:

This entity represents products/line of products that can be sold.

Key attributes:

1. ProductID: Unique identifier for the product.
2. Sell:

This is the relation between associates and products.

Key attributes:

1. ProductID: this is the foreign key from Product entity.
2. Associate\_SSN: this is the foreign key from Associate entity.

**The ML Part:**

In the insurance company’s business process, getting quotes is the first step that a customer would like to do. However, the contract benefit premium is based on multiple features that we obtain from customers. Therefore, I would like to design a machine learning model to predict(set) an appropriate quote for each customer based on their information.

The data I am going to use could not be faked by myself, so I looked for Kaggle and find this: Medical Insurance Premium(<https://www.kaggle.com/datasets/tejashvi14/medical-insurance-premium-prediction/data>). This includes several key information from a customer, such as age, diabetes, blood pressure problems, height, weight, etc.. I would like to use multiple ways to assess the importance of each feature, and set them an appropriate proportion of the final quote.