**Design: Database**

Student: Muralidhar Garapati

Project: Insurance application

Choice of database technology: Relational database. I most likely will be using Postgres database. Currently, I am using sequelize ORM and Postgres database in my local implementation.

Justification: This application requires Insured’s information, policy information (such as Policy start date, Policy end date), vehicle information, coverages information to be stored and retrieved whenever needed. All these pieces of information are related, for example, there is no policy without an Insured showing interest to take a policy, once he/she is interested then the policy information needs to be saved followed by the risk (vehicle) the carrier is willing to cover. A vehicle cannot exist on its own without a policy record. So, there is a dependency between these entities and choosing the relational DB is the right choice.

Also, if insured wants to make changes to the policy the users of the application should be able to retrieve the Policy information which will come from various related tables and updates can happen on them.

As relational database provides the ability to create meaningful information by joining the tables, it fits perfectly for this application. One use case is, at an enterprise level this will allow to generate reports either using front end or by querying the database directly by joining related tables. Few examples of such reports are, how much premium the carrier has written each month or how many policies a particular user has created within a range of days etc.

**Tables Information**

Information will be stored in these tables.

“InsuredAccount” – contains insured’s information

Columns are

Id primary key

firstName

lastName

dob

gender

email

“Address” – address of the insured will be stored in this table

Columns are

Id primary key

addrLine1

addrLine2

city

state

zipcode

accountID foreign key of InsuredAccount.

“AutoPolicy” – this is the main entity that establishes the contract between carrier and insured.

Columns are

Id primary key

policyNumber

policyStart

policyEnd

premium

status – status of the policy such as “inforce”, “waiting UW approval”, “UW rejected”

requireUWApprovalInd – indicates if UW approval is needed for this policy

isUWApprovedInd – indicates if Underwriter approved the policy

accountID foreignkey ID of the InsuredAccount for which policy record got created

userID foreignkey ID of user that created the policy

“User” – user of the application

Columns are

id primary key

firstname

lastname

username

roleName

“Vehicle” – the vehicle of the insured that is being covered

Columns are

Id primary key

year

vin

make

model

mileage

policyID foreignKey ID of the policy this vehicle is related to

“Coverage” – protection taken by the insured

Columns are

Id primary key

bodilyInjuryCovInd

propertyDmgCovInd

medPayCovInd

collisionCovInd

policyID foreignKey ID of the policy this coverage is related to

**Please visit the next page** to view the Entity-Relationship diagram detailing the relationships between the tables and the datatypes of the columns.

Diagram, schematic

Description automatically generated