Health Insurance Claim Database Design

IE 6700 SEC 02

USE CASE STUDY REPORT

Group 24

Bardia Mouhebat

Venkat Navneeth Burla

I. Executive Summary

The primary goal of this study was to design and implement a relational database that is industry ready for use in the health insurance industry for use by insurance and health care providers who have repeatedly complained about how the claim data input process that they are required to do is an extremely time-consuming process, and when looking at the problem as a whole, there is enormous amounts of data.

The creation of a relational database would assist insurance carriers in keeping track of all data in one location. It would assist to avoid data duplication and manual data entry, which would save a lot of time. The database was created with the claim process in mind. The EER and UML diagrams were modelled, and then the conceptual model was mapped to a relational model, with primary and foreign keys identified. This database was then developed in a MySQL environment, and a prototype with two tables and one relationship was created in a MongoDB database to better understand how this database would work in a NoSQL context.

The database was successfully created and subsequently connected to Python, allowing for the comparison of numerous details, some of which are given in the following sections.

II.Problem Introduction

Health insurance claim process can be confusing for a lot of people. We want to create a database to store all the information about health insurance claim process so that both the people and business owners can have a better view of the process. We want to store information about insurance company, health plan, members, claims, procedures, healthcare providers and doctors.

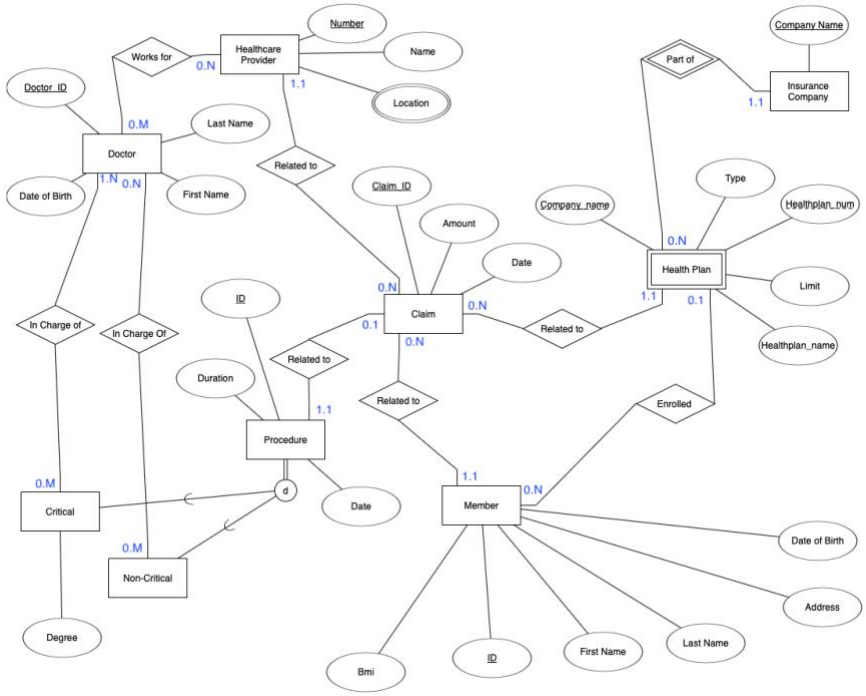
Requirements:

- For each of them we want to store the following information:
- For insurance company we want to store the company name which uniquely identifies the insurance company
- For health plan we want to store the limit, type and health plan number. Health plan number combined with the company name can uniquely identify the health plan
- For members we want to store the first name, last name, address, BMI, date of birth and member ID which uniquely identifies each member
- For claims we want to store the date, amount and claim ID which uniquely identifies each claim.
- For healthcare providers we want to store the name, location and healthcare provider number which uniquely identifies the healthcare provider. Also, each healthcare rovider can have multiple locations
- For doctors we want to store the first name, last name, date of birth and doctor ID which uniquely identifies each doctor
- For procedure we want to store the duration, date and procedure ID which uniquely identifies each procedure. Also, procedure are either critical or non-critical, and for critical procedures we want to store the degree

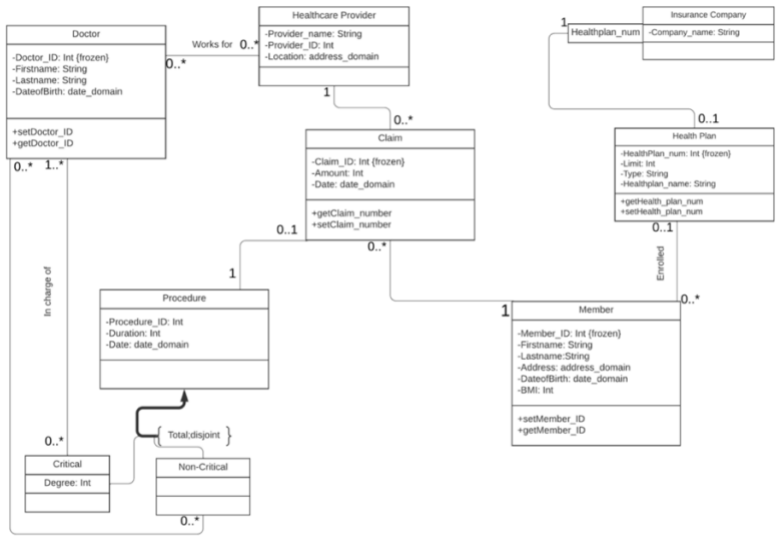
- Each health plan should have one and only one insurance company; an insurance company can have zero to multiple health plans
- Each member can be enrolled in one health plan or not be enrolled in any health plan at all; each health plan can have zero to multiple members
- Each claim should have exactly one member, one health plan, one healthcare provider and one procedure.
- Each member can have zero to multiple claims
- Each healthcare provider can have zero to multiple claims
- Each health plan can have zero to multiple claims
- Each procedure can have zero or one claim
- Each healthcare provider can have zero to multiple doctors; a doctor can work for zero to multiple healthcare providers
- Critical procedures should have at least one doctor and can have multiple doctors; a doctor can be in charge of zero to multiple critical procedures
- Non-critical procedures can have zero to multiple doctors; a doctor can be in charge of zero to multiple non-critical procedures

III.Conceptual Data Modeling

1.Conceptual data model (EER)



2.Conceptual data model (UML)



3.Relational Model

Primary Key- Bold **Foreign Key- *Italicized***

- **Insurance Company** (company_name)
- **Health plan** **(healthplan_num, healthplanname, type, limit,** *company_name*)

  Foreign key company_name refers to company_name in Insurance Company; NULL NOT ALLOWED

- **Member** (**member_ID**, firstname, lastname, address, date of birth, BMI, *healthplan_num*) Foreign key healthplan_num refers to healthplan_num in Health Plan; NULL ALLOWED
- **Claim** **(**claim_id**,** amount, date, *member_id, provider_num, procedure_id, healthplan_num*)
  Foreign key member_id refers to member_id in Member; NULL NOT ALLOWED
  Foreign key provider_num refers to provider_num in Provider; NULL NOT ALLOWED
  Foreign key procedure_id refers to procedure_id in critical_procedure and non_critical procedure; NULL NOT ALLOWED
  Foreign key healthplan_num refers to healthplan_num in Health plan; NULL NOT ALLOWED
- **Provider** (provider_num, name)
- **Provider_loc**(location, *provider_num*) Foreign key provider_num refers to provider_num in Provider; NULL NOT ALLOWED
- **Non_critical_procedure** (procedure_id, duration, date)
- **Critical_procedure** (procedure_id, duration, date, degree)
- **Doctor**(doctor_id, firstname, lastname, date of birth)
- **Works_for**(*provider_num*, *doctor_id*)
  Foreign key provider_num referes to provider_num in Provider; NULL ALLOWED
  Foreign key doctor_id referes to doctor_id in Doctor; NULL ALLOWED
- **In charge of critical** (*doctor_id*, *procedure_id*)
  Foreign key doctor_id refers to doctor_id in Doctor; NULL NOT ALLOWED
  Foreign key procedure_id refers to procedure_id in critical_procedure; NULL ALLOWD
- In charge of non-critical(*doctor_id*, *procedure_id*)
  Foreign key doctor_id refers to doctor_id in Doctor; NULL ALLOWED
  Foreign key procedure_id refers to procedure_id in non-critical_procedure; NULL ALLOWED

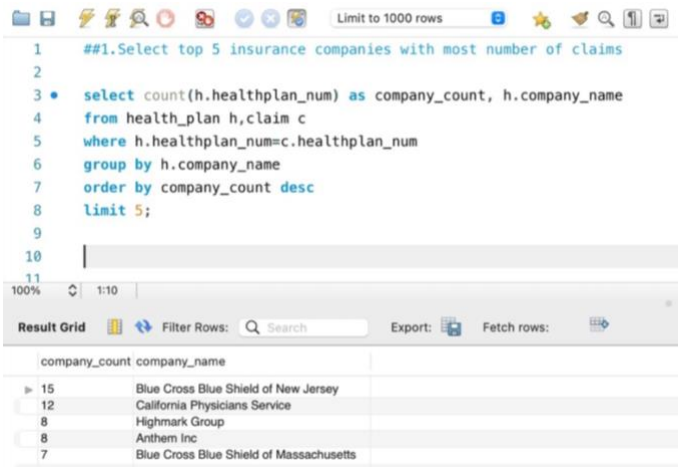
IV.Implementation of Relation Model via MySQL and NoSQL

1.My SQL Implementation:

A database was created using MySql and the following queries were performed.

1. Select top 5 insurance companies with most number of claims

```
select count(h.healthplan_num) as company_count,h.company_name
from health_plan h,claim c
where h.healthplan_num=c.healthplan_num
group by h.company_name
order by company_count desc
limit 5;
```



2. Select the average BMI of all the members who are born after 1980.

```
select avg(bmi)
from member
where date_of_birth > '1980-01-01';
```

3. Select all the doctors firstname and lastname, who where in charge of critical procedure '90'.

```
select firstname, lastname
from doctor
where doctor_id in (select d.doctor_id from doctor d, in_charge_of_critical ic
where d.doctor_id=ic.doctor_id and ic.critical_procedure_id='90');
```

```
17    ##3. Select all the doctors firstname and lastname, who where in charge of critical procedure '90'.
18
19 •  select firstname, lastname
20    from doctor
21    where doctor_id in (select d.doctor_id from doctor d, in_charge_of_critical ic
22    where d.doctor_id=ic.doctor_id and ic.critical_procedure_id='90');
23
```

100% 67:22

Result Grid | Filter Rows: Search | Export:

firstname	lastname
Laila	Friese
Alden	Puccio
Dona	Mccammon

4. Select all the doctors who work for at least 3 healthcare providers.

Solution a ➔ select d.firstname, d.lastname, d.doctor_id
from doctor d
where(select count(*) from works_for w
where d.doctor_id=w.doctor_id group by w.doctor_id) >=3;

Solution b ➔ select firstname, lastname, doctor_id
from doctor
where doctor_id in(
select doctor_id from works_for
group by doctor_id
having count(*) >= 3);

Limit to 1000 rows

```
24    ##4. Select all the doctors who work for at least  3 healthcare providers.
25 •  select firstname, lastname, doctor_id
26    from doctor
27    where doctor_id in(
28    select doctor_id from works_for
29    group by doctor_id
30    having count(*) >= 3);
31
```

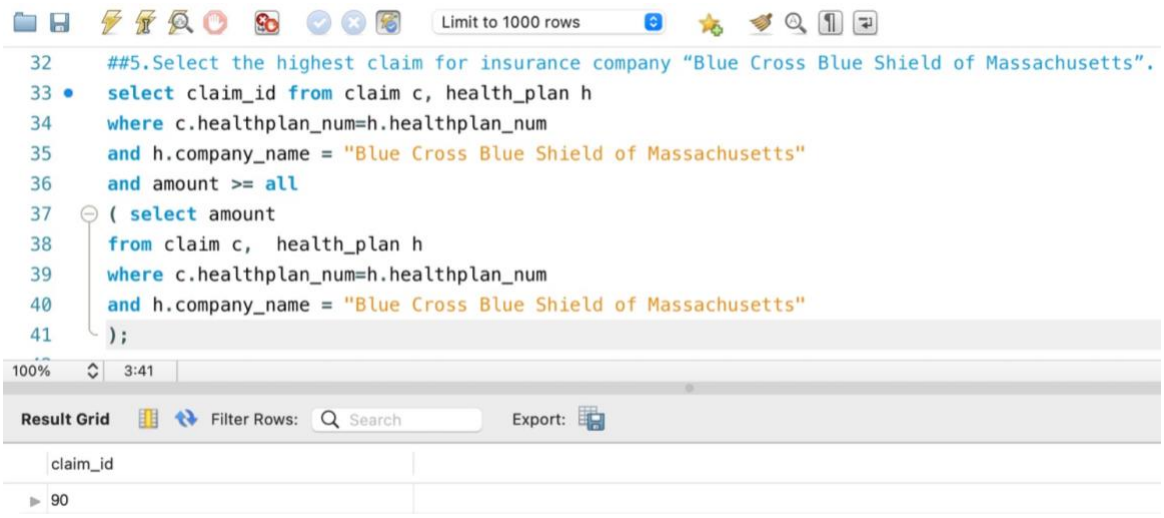
100% 23:30

Result Grid | Filter Rows: Search | Edit: | Export/Import:

firstname	lastname	doctor_id
Laila	Friese	1232122
Alden	Puccio	1232132
Teena	Wurst	1232252
Lanie	Witmer	1232262
Alyson	Vahle	1232402
NULL	NULL	NULL

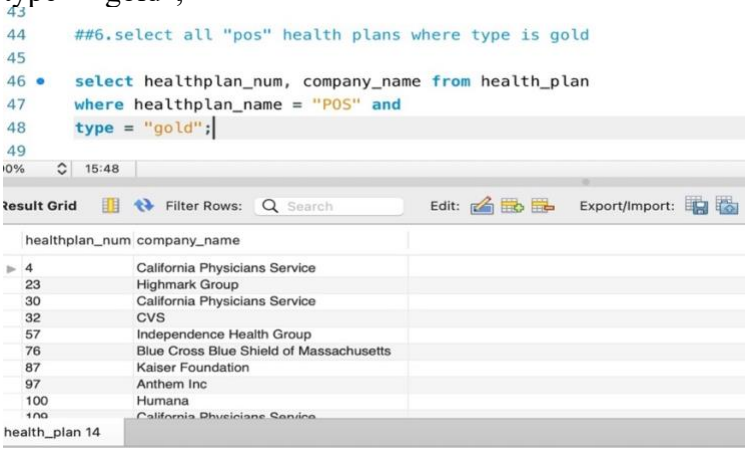
5.Select the highest claim for insurance company “Blue Cross Blue Shield of Massachusetts”.

```
select claim_id from claim c, health_plan h
where c.healthplan_num=h.healthplan_num
and h.company_name = "Blue Cross Blue Shield of Massachusetts"
and amount >= all
( select amount
from claim c,  health_plan h
where c.healthplan_num=h.healthplan_num
and h.company_name = "Blue Cross Blue Shield of Massachusetts"
);
```



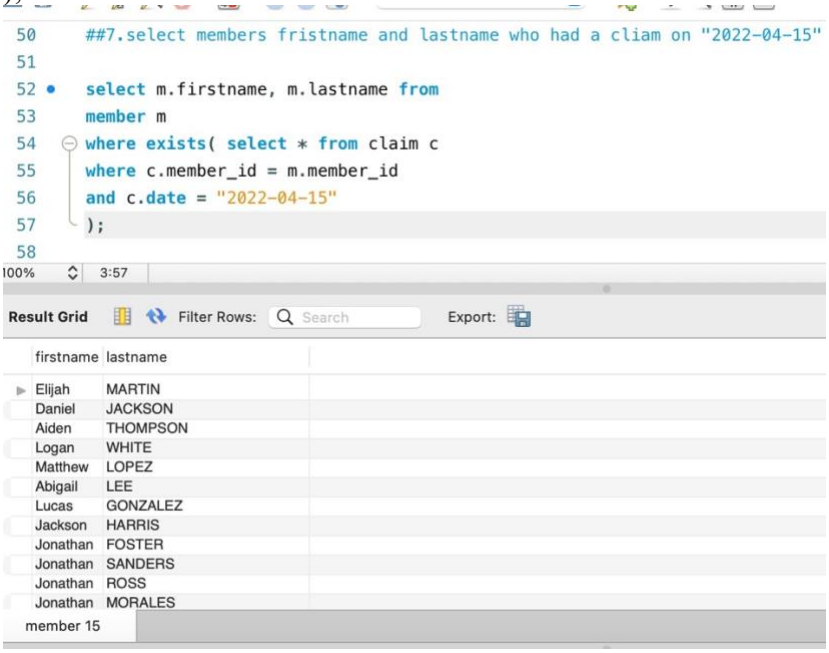
6. select all "pos" health plans where type is gold

```
select healthplan_num, company_name from health_plan
where healthplan_name = "POS" and
type = "gold";
```



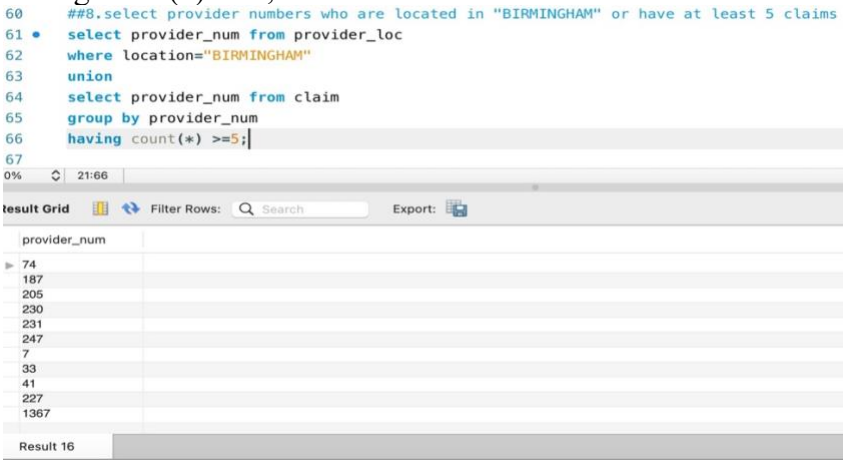
7.select members fristname and lastname who had a claim on "2022-04-15"

```
select m.firstname, m.lastname from
member m
where exists( select * from claim c
where c.member_id = m.member_id
and c.date = "2022-04-15"
);
```



8.select provider numbers who are located in "BIRMINGHAM" or have at least 5 claims

```
select provider_num from provider_loc
where location="BIRMINGHAM"
union
select provider_num from claim
group by provider_num
having count(*) >=5;
```



9.select all the doctors who are not in charge of a critical procedure

select lastname from doctor
where doctor_id not in (select doctor_id from in_charge_of_critical);

```
69    ##9.select all the doctors who are not in charge of a critical procedure
70 •  select lastname from doctor
71    where doctor_id not in (select doctor_id from in_charge_of_critical);
72
```

100% 70:71

Result Grid | Filter Rows: Search | Export:

lastname
Gabourel
Pasquale
Westergard
Vahle
Lukowski

10.show min,max and average amount of claims

select min(amount) as min_amount, max(amount) as max_amount, avg(amount) as avg_amount from claim;

```
73    ##10.show min,max and average amount of claims
74 •  select min(amount) as min_amount, max(amount) as max_amount, avg(amount) as avg_amount
75    from claim;
76
```

100% 12:75

Result Grid | Filter Rows: Search | Export:

min_amount	max_amount	avg_amount
51	992	572.1354

2.No SQL Implementation:

The following tables were created and the corresponding queries were implemented

1.Insurance company table
2.Health plan table
3.Member
4.Provider

1. **Average limit of each company's health plans**

```
db.healthplan.aggregate(
   [
      {$group: {_id:"$company_name", average :{$avg: "$limit"}}},
      {$sort:{average:-1}}

      ]);
```

```
//Average limit of each company's health plans

db.healthplan.aggregate(
   [
      {$group: {_id:"$company_name", average :{$avg: "$limit"}}},
      {$sort:{average:-1}}

      ]);
```

Run

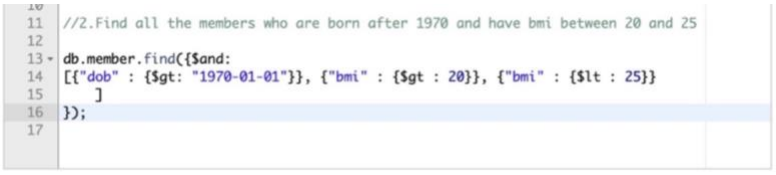
Result

```
{ "_id" : "Kaiser Foundation", "average" : 5000 }
{ "_id" : "United Health", "average" : 4666.666666666667 }
```

---

2. **Find all the members who are born after 1970 and have bmi between 20 and 25**

```
db.member.find({$and:
[{"dob" : {$gt: "1970-01-01"}}, {"bmi" : {$gt : 20}}, {"bmi" : {$lt : 25}}
   ]
});
```



```
//2.Find all the members who are born after 1970 and have bmi between 20 and 25

db.member.find({$and:
[{"dob" : {$gt: "1970-01-01"}}, {"bmi" : {$gt : 20}}, {"bmi" : {$lt : 25}}
   ]
});
```

Run

Result

```
{ "_id" : 1043, "firstname" : "Emma", "lastname" : "SMITH", "address" : "Lingard L
{ "_id" : 1050, "firstname" : "Sophia", "lastname" : "JONES", "address" : "1 Brood
{ "_id" : 1060, "firstname" : "William", "lastname" : "RODRIGUEZ", "address" : "Th
```

---

3. F**ind the top three health plans with highest average of bmi**

```
db.member.aggregate(
  [
    {$group: {_id:"$healthplan_num", avg_bmi :{$avg: "$bmi"}}},
    {$sort:{avg_bmi:-1}},
    {$limit:3}
    ]);
```

```
18 //3.Find the top three health plans with highest average of bmi
19
20 db.member.aggregate(
21     [
22         {$group: {_id:"$healthplan_num", avg_bmi :{$avg: "$bmi"}}},
23         {$sort:{avg_bmi:-1}},
24         {$limit:3}
25         ]);
26
27
```

Run

Result

```
{ "_id" : 8, "avg_bmi" : 35.5 }
{ "_id" : 26, "avg_bmi" : 29.3 }
{ "_id" : 69, "avg_bmi" : 27.6 }
```

4. Find all the health plans where name is “PPO”, type is “silver” and company_name is “Kaiser Foundation”

```
db.healthplan.find({$and:
[{"healthplan_name" : "PPO"}, {"type" : "silver"}, {"company_name" : "Kaiser Foundation"}
   ]
});
```

```
26
27 //4.Find all the health plans where name is “PPO”, type is “silver” and company_name is
28
29 db.healthplan.find({$and:
30 [{"healthplan_name" : "PPO"}, {"type" : "silver"}, {"company_name" : "Kaiser Foundation"
31     ]
32 });
33
34
```

Run

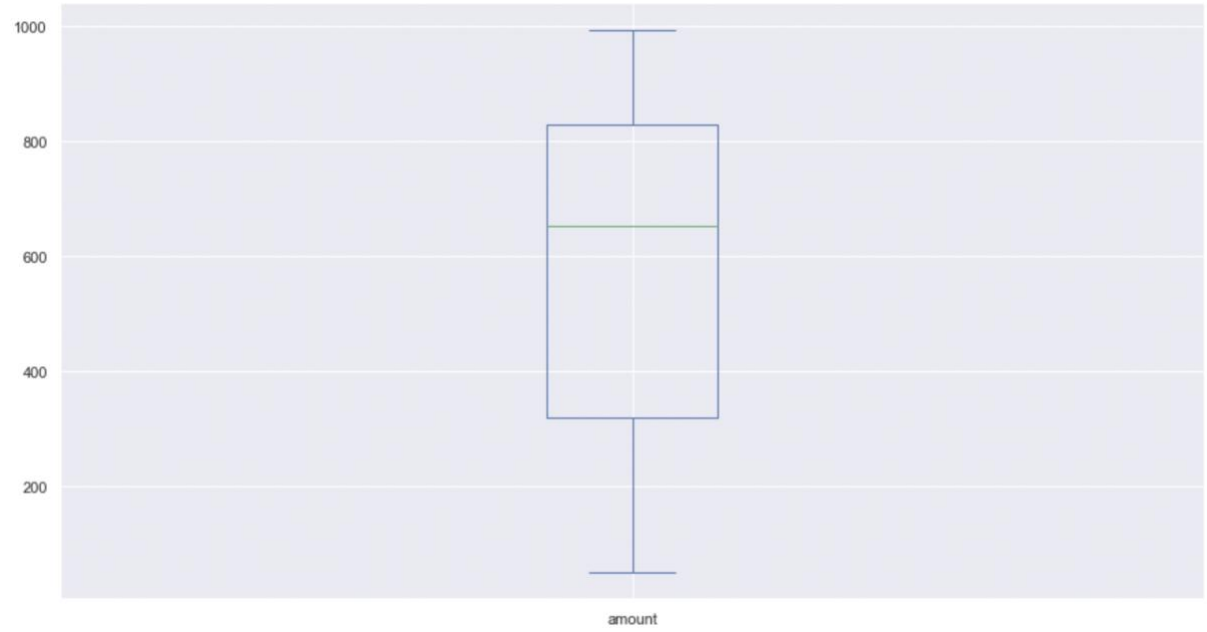
Result

```
{ "_id" : 87, "healthplan_name" : "PPO", "type" : "silver", "limit" : 5000, "compa
{ "_id" : 98, "healthplan_name" : "PPO", "type" : "silver", "limit" : 5000, "compa
```

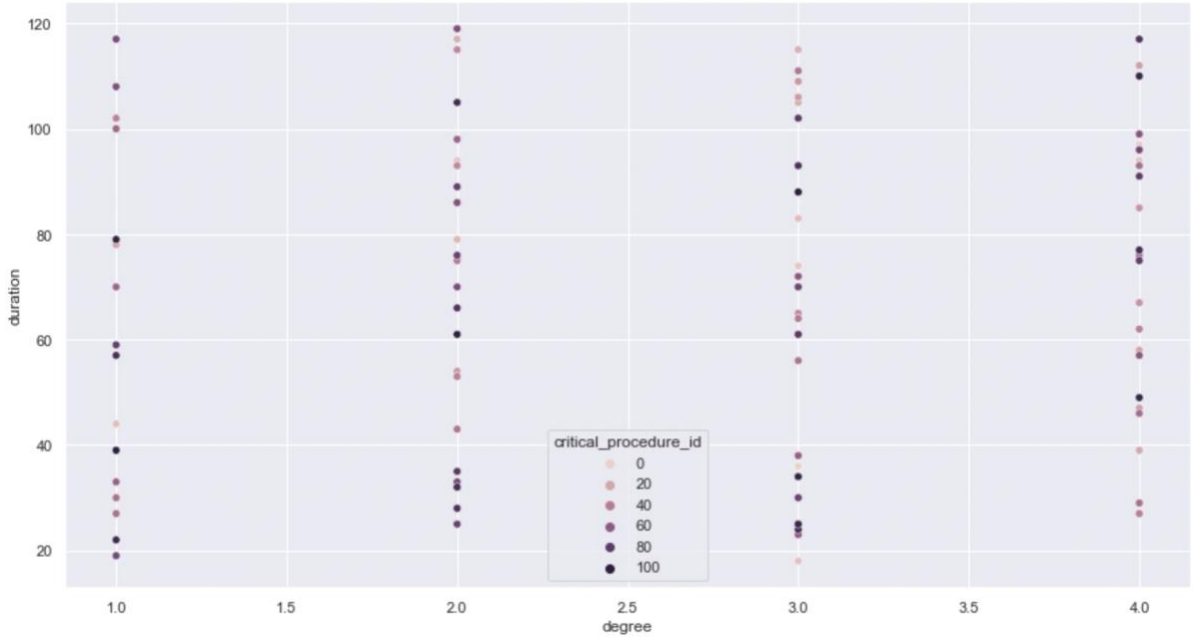
V.Database Access via Python

The Insurance database was accessed using python and after successfully connected, the data was fetched and some plots were plotted for analysis. MySQL connection using Python was established using mysql.connector, followed by cursor.excecute to run and fetchall from query. The required tables are fetched and converted to a dataframe using pandas library and using matplotlib graphs were plotted for analysis.

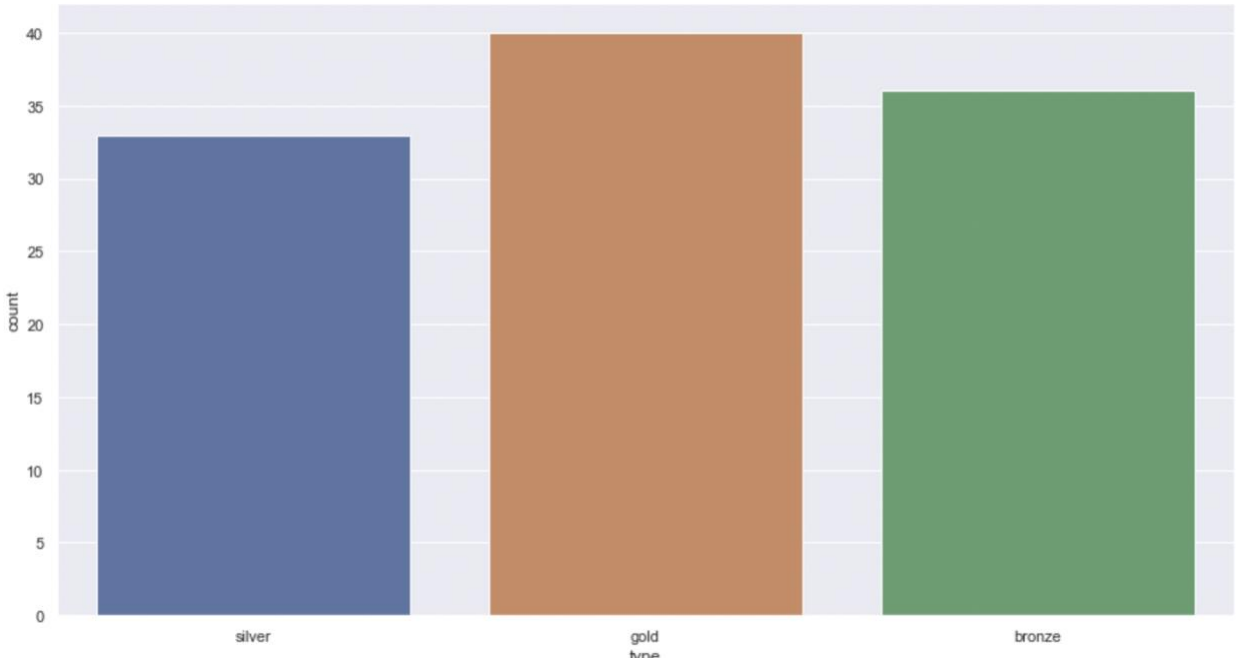
Graph 1: Box plot for amount of claims



Graph 2: Scatter plot for degree vs duration



Graph 3: Count plot for type of Insurance plan



VI.Summary and Recommendation

The MySQL-based Health Insurance Claim Database is an industry-ready relational database that may be implemented in the insurance sector. It will reduce confusion among patients and health care professionals during the claim process and will provide excellent analytics capabilities, a portion of which is demonstrated in this report using Python.

In terms of data quality, there is also room for improvement. When analysing data, care must be taken because there is a risk that the data will become damaged and that the conclusions will not be as expected. As a result, having a database is always useful and would avert a lot of problems.

Implementing data governance controls on the database to assure data quality would be an improvement, as the submitted insurance claim data is reused by other providers to enter claim details into the database of a specific patient. The health plan should be linked to a patient id, which, once entered, should allow the health care provider to choose which operation to do and read all of the patient's attributes automatically.