IE 6700

Case Study

Group 25

Team Information:

Archit Raj, Sai Vinay Teja Jakku

1. Business Problem Definition & Requirements:

Real estate is an ever-changing and growing market. There is always a need to bridge the gap between the consumers and suppliers of the market. In this project, we attempt to close that gap by building a database that helps store, retrieve, and analyze information about the key commodity of the market i.e Houses. We do this by creating 4 tables with Renter, Owner, House, and Broker Company as key parameters. We draw relations between these parameters and their attributes and build a database that can give access to good-quality affordable housing and can even be used to strengthen the knowledge base for policy evaluation.

The following table contains how a user could link the relationship between the entities. A renter has to contact the broker company in order to get the data for the house and the owner.

In the implementation following rules have to be kept in mind:

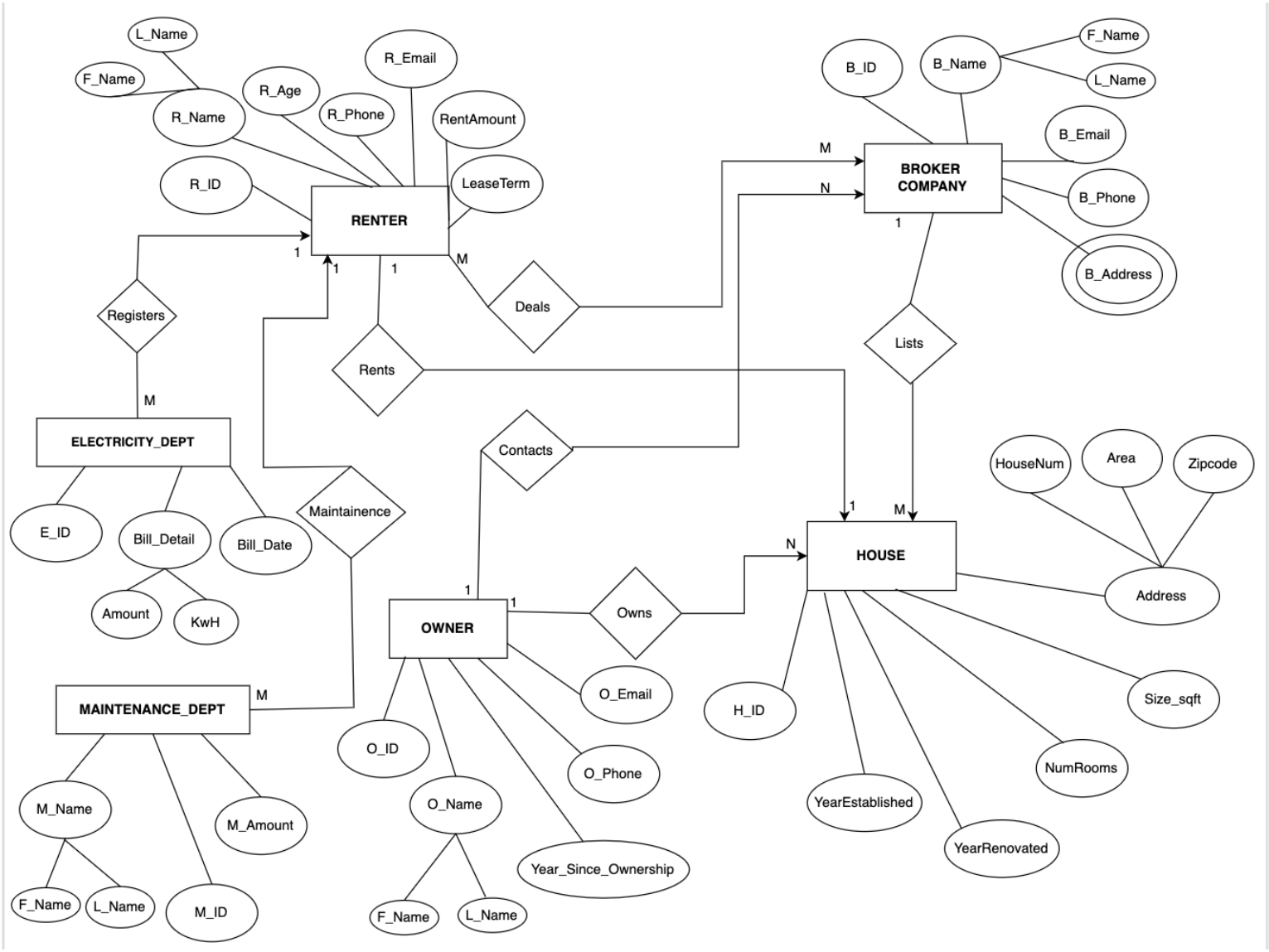
- Broker companies can have access to multiple houses whereas houses can have access to only one broker company.

- Owners can own multiple houses and can have access to more than one broker each.
- Renters can deal with many broker companies and many broker companies can deal with many other renters.
- Houses can have access to at least one broker company, one renter, and one owner.
- Maintenance worker can work with many houses
- Electricity department makes bills for all the houses

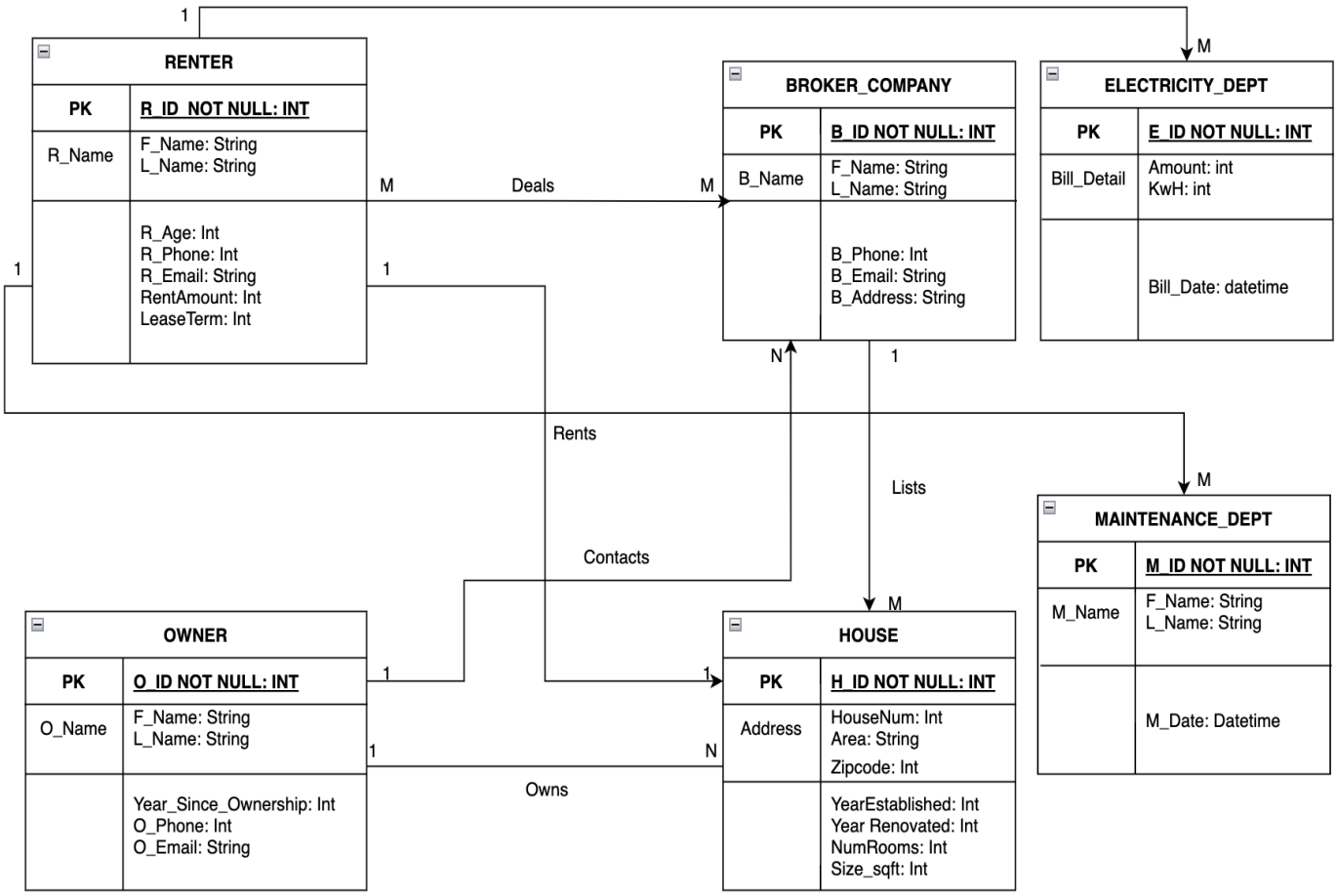
2. Design:

To create the database that meets the requirements of the problem statement, we first need to identify the required tables such as Renter, Owner, House, Broker Company.
All the details of the house are being stored by the broker company, such as an address, size, number of rooms, etc. These details are being provided by the owner of the house. Every renter can deal with the broker company by providing them their details such as rent amount, term of the lease, along with their personal details such as email and phone so that they can be contacted back by the broker company.

3. Conceptual Data Model - Enhanced Entity-Relationship (EER):



4. Conceptual Data Model - Unified Modeling Language (UML):



5. Relational Model:

Primary Key: Bold || Foreign Key: Underlined and Italics

Renter(**R_ID,** F_Name, L_Name, R_Age, R_Phone, R_Email, RentAmount, LeaseTerm, *B_ID*, *H_ID*)

For Foreign Keys:

B_ID refers to the B_ID taken from B_ID in Broker_Company

H_ID refers to the H_ID taken from H_ID in House

BrokerCompany(**B_ID**, F_Name, L_Name, B_Phone, B_Email, B_Address, *R_ID*, *O_ID*, *H_ID*)

For Foreign Keys:

R_ID refers to the R_ID taken from R_ID in Renter

O_ID refers to the O_ID taken from O_ID in Owner

H_ID refers to the H_ID taken from H_ID in House

Owner(**O_ID**, F_Name, L_Name, Year_Since_Ownership, O_Phone, O_Email, *B_ID*, *H_ID*)

For Foreign Keys:

O_ID refers to the O_ID taken from O_ID in Owner

H_ID refers to the H_ID taken from H_ID in House

House(**H_ID**, HouseNum, Area, Zipcode, YearEstablished, NumRooms, Size_sqft, *O_ID*, *R_ID*, *B_ID*)

For Foreign Keys:

O_ID refers to the O_ID taken from O_ID in Owner

R_ID refers to the R_ID taken from R_ID in Renter

R_ID refers to the R_ID taken from R_ID in Renter

Electricity_Dept(**E_ID**, Amount, KwH, Bill_Date, Renter_*R_ID*)

For Foreign Keys:

Renter_R_ID refers to the R_ID taken from R_ID in Renter

Maintenance_Dept(**M_ID**, M_Name(F_Name, L_Name), M_Date, Renetr_*R_ID*)

For Foreign Keys:

Renter_R_ID refers to the R_ID taken from R_ID in Renter

6. Implementation of Relation Model via MySQL and NoSQL:

A. MySQL Implementation:

The database was created in MySQL and the following queries were performed:

Query1: Display renters who paid a rent amount $2500 with a Lease Term of 2 years.

```
SELECT F_NAME, L_NAME, R_ID, Rent_Amount,
Lease_Term FROM Renter
WHERE Rent_Amount > 2500 AND Lease_Term = 2;
```

F_NAME	L_NAME	R_ID	Rent_Amount	Lease_Term
Donny	Banner	4	2900	2
Roger	Francis	5	2590	2
Virgil	Pinto	7	3600	2
Loren	De Silva	9	3500	2
Olivia	Phil	14	3100	2
NULL	NULL	NULL	NULL	NULL

Query2: Display the name of renters whose house was established after 2000.

```
SELECT r.F_Name, r.L_Name, r.Rent_Amount
FROM Renter r
LEFT JOIN House h
ON r.House_H_ID = h.H_ID
WHERE h.YearEstablished > 2000;
```

F_Name	L_Name	Rent_Amount
Loren	De Silva	3500
Ben	Perkins	2850
Joe	Porter	2100
Donny	Banner	2900
Carrol	Acosta	2890

Query3. Display the details of brokers with houses which are not occupied by the renters.

```
SELECT *
FROM Broker_Company
WHERE Broker_Company.B_ID
IN (SELECT Broker_Company_B_ID FROM House WHERE H_ID NOT IN (SELECT Renter.House_H_ID
FROM RENTER));
```

B_ID	F_Name	L_Name	B_Phone	B_Email	B_Address
6	Devan	Nitzsche	7174901742	edavis@gmail.com	68754 Clementine Haven
7	Jaylon	Littel	7907867712	hilbert80@yahoo.com	564 Delfina Springs Suite 892
8	Alf	Gutkowski	8961894922	yschulist@hotmail.com	320 Amelia Heights Suite 986
9	Archibald	Morar	8789696198	annabelle75@yahoo.com	568 Dayna Lane
NULL	NULL	NULL	NULL	NULL	NULL

Query4. Display renter's details of those who pay below the average electricity bill.

```
SELECT r.F_Name, r.L_Name,
r.R_Phone, r.R_Email, e.Amount
FROM Renter r
LEFT JOIN Electricity_Dept e
ON r.R_ID = e.Renter_R_ID
WHERE e.Amount < (SELECT
AVG(Amount) FROM Electricity_Dept);
```

F_Name	L_Name	R_Phone	R_Email	Amount
Virgil	Pinto	8689675663	virgil@gmail.com	150
Irving	Tran	2522352643	irtran@gmail.com	110
Carrol	Acosta	8924100411	carrol@gmail.com	110
Joe	Porter	8738971874	joe@gmail.com	150
Edurardo	Xavi	3523665223	xavi@gmail.com	124
Roger	Francis	7896896812	roger@gmail.com	120
Jean	Paul	4523523676	jpaul@gmail.com	120
Olivia	Phil	8908142218	phil@gmail.com	124
Irving	Tran	2522352643	irtran@gmail.com	110
Stephen	Wright	2414214654	steph@gmail.com	120
Erwin	Klause	7907907456	kerwin@gmail.com	110
Carrol	Acosta	8924100411	carrol@gmail.com	100
Joe	Porter	8738971874	joe@gmail.com	150
Donny	Banner	7240971904	don@gmail.com	125
Edurardo	Xavi	3523665223	xavi@gmail.com	155
Roger	Francis	7896896812	roger@gmail.com	110

Query5. List top 3 brokers with the maximum number of renters.

```
SELECT b.B_ID, b.F_name, b.L_name,
b.B_phone, COUNT(r.R_ID) AS No_of_Renters
FROM Broker_company b
JOIN House h
ON b.b_id = h.Broker_Company_B_iD
JOIN Renter r
ON r.House_H_ID = h.H_ID
GROUP BY b.B_iD
ORDER BY No_of_Renters DESC
LIMIT 3;
```

B_ID	F_name	L_name	B_phone	No_of_Renters
1	Ansel	Lind	2585470601	3
2	Jose	Bruen	3268089101	3
9	Archibald	Morar	8789696198	2

Query 6. Display the houses which haven't been renovated in a year since they were established.

```
SELECT H_ID, H_num, Area
FROM House
WHERE (YearRenovated-YearEstablished) > 1;
```

H_ID	H_num	Area
1	22	Harvard St
2	150	12 Stoneholm St
3	150	Huntington Ave
5	1	12 Stanton St
6	309	W 3rd St
8	318	Train St
10	67	Topliff St
11	1244	Irving St
12	309	Mass Ave
13	19	Fairmont St
14	47	Boylston St
15	124	Blue Hill Dr
16	123	Regina Rd
17	67	Park Dr
18	456	Clarendon St
19	28	Harvard St
20	127	Huntington Ave

Query7. Display the Owner information with most no.of houses to rent

```
SELECT o.O_ID, o.F_name, o.L_name, o.O_phone
FROM Owner o
JOIN House h
ON o.House_H_ID = h.H_ID
GROUP BY o.O_ID
ORDER BY COUNT(h.H_ID) DESC
LIMIT 5;
```

O_ID	F_name	L_name	O_phone
1	Brose	Chaperlin	9071412498
2	Paten	Conring	3898787334
3	John	Di Franceschi	7678512311
4	Niccolo	Arnoud	8789712352
5	Chen	Bryden	4124523566

Query 8. Display the houses with at least 3 bedrooms and ready to rent

```
SELECT *
FROM House h
WHERE H_ID
IN (SELECT H_ID FROM House WHERE H_ID NOT IN (SELECT Renter.House_H_ID FROM RENTER))
AND NumRooms >= 3;
```

H_ID	H_Num	Area	Zipcode	YearEstablished	YearRenovated	NumRooms	Size_sqft	Broker_Company_B_ID
9	3	Washington St	02118	2016	NULL	3	1770	9
12	309	Mass Ave	02115	2000	2020	3	1800	8
16	123	Regina Rd	02124	1991	2010	4	1981	6
18	456	Clarendon St	02215	1988	2012	3	1650	7
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

B. NoSQL Implementation:

The 6 tables(Renter, House, Broker_Company, Owner, Electricity_Dept, Maintenance) were implemented in the MongoDB as 6 different collections. Then, later on, the team worked on certain NoSQL's MongoDB following queries:

1. **Find Renters who are aged above 30 and have a lease term of 2 years.**

```
db.Renter.find({ $and:[{"Lease_Term":"2"},{"R_Age":{$gt:30}}]}).pretty()
```

```
{ _id: ObjectId("6264b4ea803de902c5346321"),
  R_ID: 9,
  F_Name: 'Loren',
  L_Name: 'De Silva',
  R_Age: 37,
  R_Phone: '2459471919',
  R_Email: 'desilva@gmail.com',
  Rent_Amount: '3500',
  Lease_Term: '2',
  House_H_ID: '4' }
{ _id: ObjectId("6264b4ea803de902c5346323"),
  R_ID: 11,
  F_Name: 'Edurardo',
  L_Name: 'Xavi',
  R_Age: 51,
  R_Phone: '3523665223',
  R_Email: 'xavi@gmail.com',
  Rent_Amount: '1670',
  Lease_Term: '2',
  House_H_ID: '5' }
{ _id: ObjectId("6264b4ea803de902c534631f"),
  R_ID: 7,
  F_Name: 'Virgil',
  L_Name: 'Pinto',
  R_Age: 35,
  R_Phone: '8689675663',
  R_Email: 'virgil@gmail.com',
  Rent_Amount: '3600',
  Lease_Term: '2',
  House_H_ID: '2' }
Atlas atlas-ewcr7d-shard-0 [primary] HouseRenterData>
```

2. **List the total Amount of Electricity bill, KwH sorted by Renter_ID.**

```
db.Electricity_Dept.aggregate([
  {$match: {}},
```

```
{ _id: 1, total_Amount: 220, total_KwH: 4000 }
{ _id: 2, total_Amount: 300, total_KwH: 5740 }
{ _id: 3, total_Amount: 450, total_KwH: 9000 }
{ _id: 4, total_Amount: 305, total_KwH: 6000 }
{ _id: 5, total_Amount: 230, total_KwH: 3780 }
{ _id: 6, total_Amount: 210, total_KwH: 2600 }
{ _id: 7, total_Amount: 330, total_KwH: 5000 }
{ _id: 8, total_Amount: 320, total_KwH: 5860 }
{ _id: 9, total_Amount: 350, total_KwH: 5780 }
{ _id: 10, total_Amount: 400, total_KwH: 7200 }
{ _id: 11, total_Amount: 279, total_KwH: 3800 }
{ _id: 12, total_Amount: 290, total_KwH: 6200 }
{ _id: 13, total_Amount: 370, total_KwH: 6468 }
{ _id: 14, total_Amount: 304, total_KwH: 6300 }
{ _id: 15, total_Amount: 550, total_KwH: 7800 }
Atlas atlas-ewcr7d-shard-0 [primary] HouseRenterData>
```

{$group: {_id: "$Renter_R_ID", total_Amount: {$sum: "$Amount"}, total_KwH: {$sum: "$KwH"}}},
{$sort: {_id: 1, total_Amount:1}}])

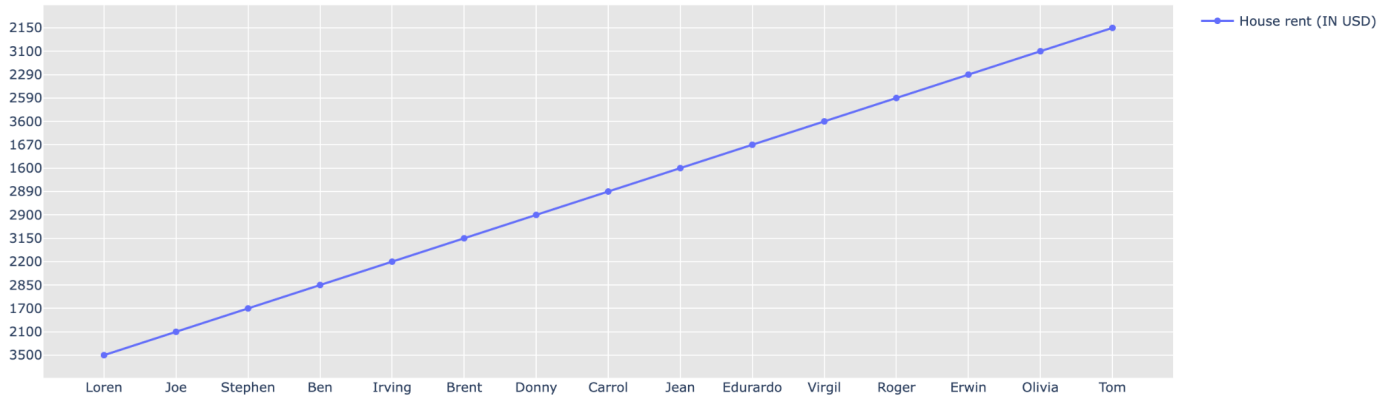
3. **Write the query for the number of houses that have a number of rooms greater than 4 or at least Size of the house more than 2000 sq ft.**
db.House.find({ $or:[{"NumRooms":{$gt:4}}, {"Size_sqft":{$gt:2000}}]}).pretty()

```
> db.House.find({ $or:[{"NumRooms":{$gt:4}}, {"Size_sqft":{$gt:2000}}]}).pretty()
< { _id: ObjectId("6268a9de803de902c53463ae"),
    H_ID: 2,
    H_Num: 150,
    Area: '12 Stoneholm St',
    Zipcode: '02115',
    YearEstablished: 1975-01-01T00:00:00.000Z,
    YearRenovated: 2005-01-01T00:00:00.000Z,
    NumRooms: 4,
    Size_sqft: 2250,
    Broker_Company_B_ID: 9 }
  { _id: ObjectId("6268a9de803de902c53463b0"),
    H_ID: 4,
    H_Num: 527,
    Area: 'Mass Ave',
    Zipcode: '02118',
    YearEstablished: 2016-01-01T00:00:00.000Z,
    YearRenovated: 1970-01-01T00:00:00.000Z,
    NumRooms: 4,
    Size_sqft: 2350,
    Broker_Company_B_ID: 4 }
  { _id: ObjectId("6268a9de803de902c53463b4"),
    H_ID: 8,
    H_Num: 318,
    Area: 'Train St',
    Zipcode: '02112',
    YearEstablished: 2003-01-01T00:00:00.000Z,
    YearRenovated: 2017-01-01T00:00:00.000Z,
    NumRooms: 5,
    Size_sqft: 2950,
    Broker_Company_B_ID: 7 }
  { _id: ObjectId("6268a9de803de902c53463bd"),
    H_ID: 17,
    H_Num: 67,
    Area: 'Park Dr',
    Zipcode: '02215',
    YearEstablished: 2002-01-01T00:00:00.000Z,
    YearRenovated: 2021-01-01T00:00:00.000Z,
    NumRooms: 5,
    Size_sqft: 2700,
    Broker_Company_B_ID: 9 }
Atlas atlas-ewcr7d-shard-0 [primary] HouseRenterData>
```

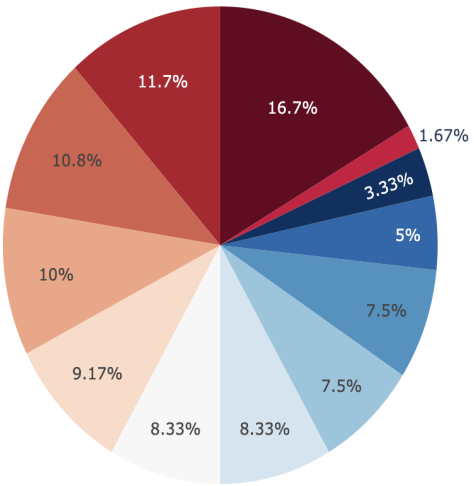
7. Database Visualization through Python:

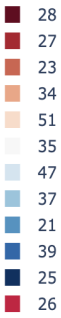
The database is accessed using Python's PyMongo library to perform visualization of analyzed data as shown below. The connection of the MongoDB database to Python is done using PyMongo and MongoClient, followed by executing the collections to the dataframes using Pandas library and using Matplotlib and Plotly to plot the graphs for the analytics.

Graph 1: Rent Amount per Renters

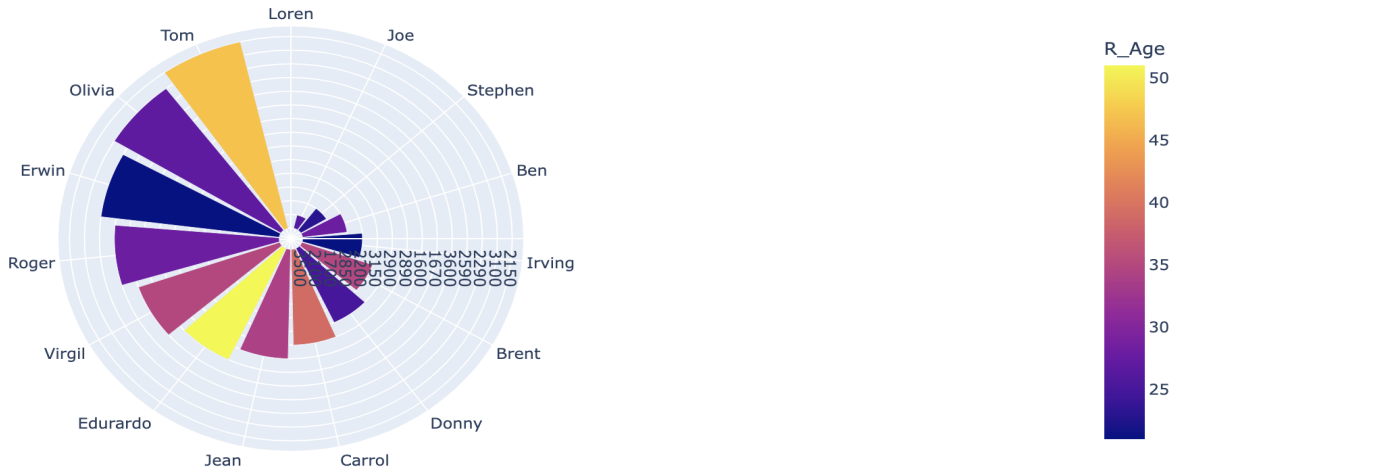


Graph 2: Renter Age Distribution Pie:

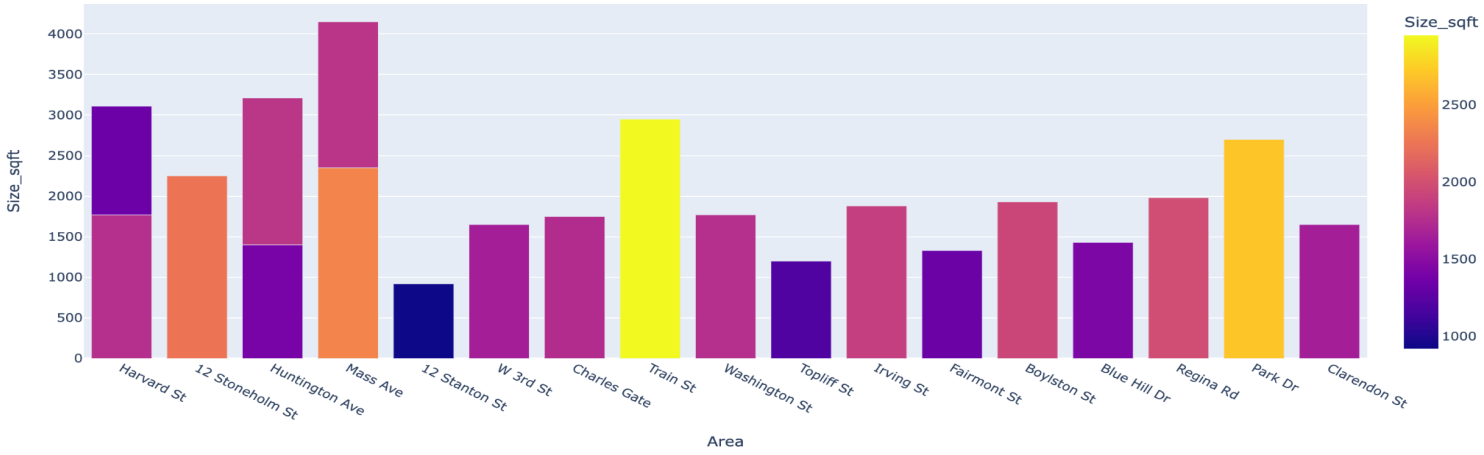




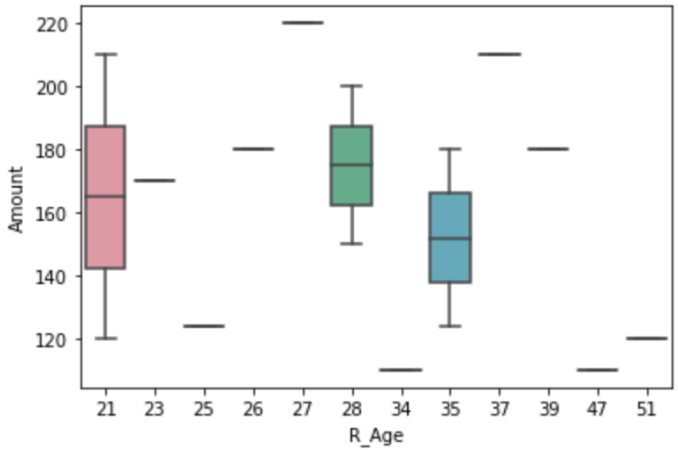
Graph 3: Proportion of Rent Amount per Renter vs Age:



Graph 4: Area of the House vs Size per Sqft:



Graph 5: Renter Age vs Amount of Electricity Paid:



8. Conclusion: Outcomes and shortcomings of the Projects

1. Project Outcomes:

- We have successfully implemented our conceptual model in a SQL database and converted that to a NoSQL database.
- We have demonstrated database access within the application written in Python and performed and visualized useful analytics.

2. Shortcomings:

- Having implemented multiple layers of database languages, MySQL, NoSQL, and then using Python libraries for getting the insights would keep it highly unstable for the long run.
- Focusing more on the MongoDB platform would help the organization in gaining proper control of its whereabouts, by utilizing a cloud database setup.
- For key insights, the company could shift more towards BI tools like Tableau, PowerBI, or Figma.