Database Management

CS 4342 / CS5342

Spring 2023 Semester

Assignment 3

“The Archivers”

Paul Djangang

Leila Martinez

Angel Otero Montanez

S. M. Mustaquim

Mohammad Ariful Islam Khan

March 30, 2023

Table of Contents

1. SCOPE

The aim of the project is to design and implement a database system to manage the tourism industry in El Paso County. The system will register and manage tourist businesses that require a license to operate in the county. Each business in the database has a unique name and unique TIN[1]. The database will store if a business is registered in the county, and each business' name, type, phone number, email address. A business registered in the county can request a business license, each of which has a unique license number; the database will store each license's issue date, expiration date, status, and request number. The request number is unique.

When a business is removed from the system, the license is also removed. Licenses will be approved by agents that have a unique ID and a unique office number. The database will also store each agent's official name, and office information such as multiple office phone numbers, address, and email.

The database will store each business' employees information such as their unique SSN[2], position, payment, name, birth date and gender. We will keep track of the number of employees for each business. Each business will have an owner, or multiple owners whose personal information such as name, gender, birth date, home address, phone number, and unique email will be stored in the database. Owners can have multiple businesses. Business location information such as their unique location code, address, multiple phone numbers, if it's rented or owned, the start date of renting, the owner, and ADA accessibility compatibility will be stored in the database. A business can provide different types of activities. Each activity has a unique name, activity type, activity description, risk type, and cost associated to the activity. When a business is removed from the system, the activities provided by that business are also removed.

The system will provide data analytics features, which will be available for El Paso County to retrieve. These features will include the inquiry of number of workers per business, number of licenses issued and canceled, the average time to approve and issue a license, the business distribution in the region, the business per tourism type, the business with more than N workers, the number of workers per business, the business with activities that have a high level of risk, and the business located per zip code, and the information of specific business owners.

2. REQUIREMENTS

R1. The system shall retrieve the number of employees for a particular business

R2. The system shall retrieve the payment information for a particular position

R3. The system shall retrieve all locations for a particular business

R4. The system shall List all businesses with their owners

R5. The system shall show the total payment for each type of business

R6. The system shall list all businesses and their activities

R7. The system shall find businesses with expired licenses

R8. The system shall show the number of employees per business

R9. The system shall list activities with risk type 'Medium' or higher

R10. The system shall find the most expensive activity per business

3. ASSUMPTIONS

- The email address of each owner is unique.
- A license can be issued to only one registered business.
- An agent can approve multiple licenses, but each license can be approved by only one agent.
- Each business location has a unique location code and can be rented by only one business at a time.
- The system will store data only for businesses that require a license to operate in El Paso County.
- The system will store data only for agents who approve licenses for registered businesses in El Paso County.
- The system will store data only for employees who work for registered businesses in El Paso County.

4. ENTITY-RELATIONSHIP DIAGRAM

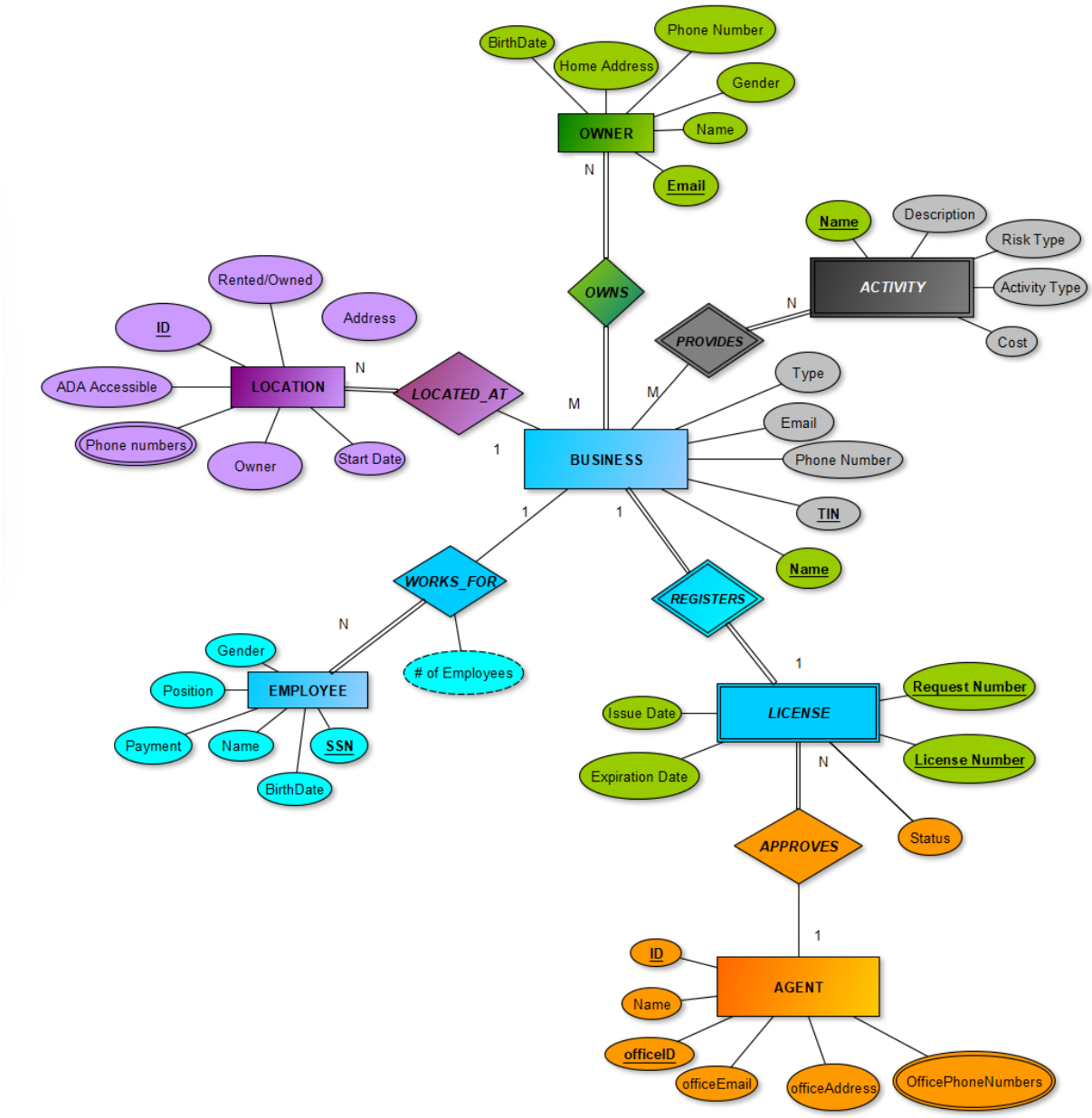


Figure 1. Entity/Relationship Diagram for Tourism Industry

Purple – Arif
Black – Angel
Orange – Leila
Green – Paul
Blue – Mustaquim

5. RELATIONAL MODEL

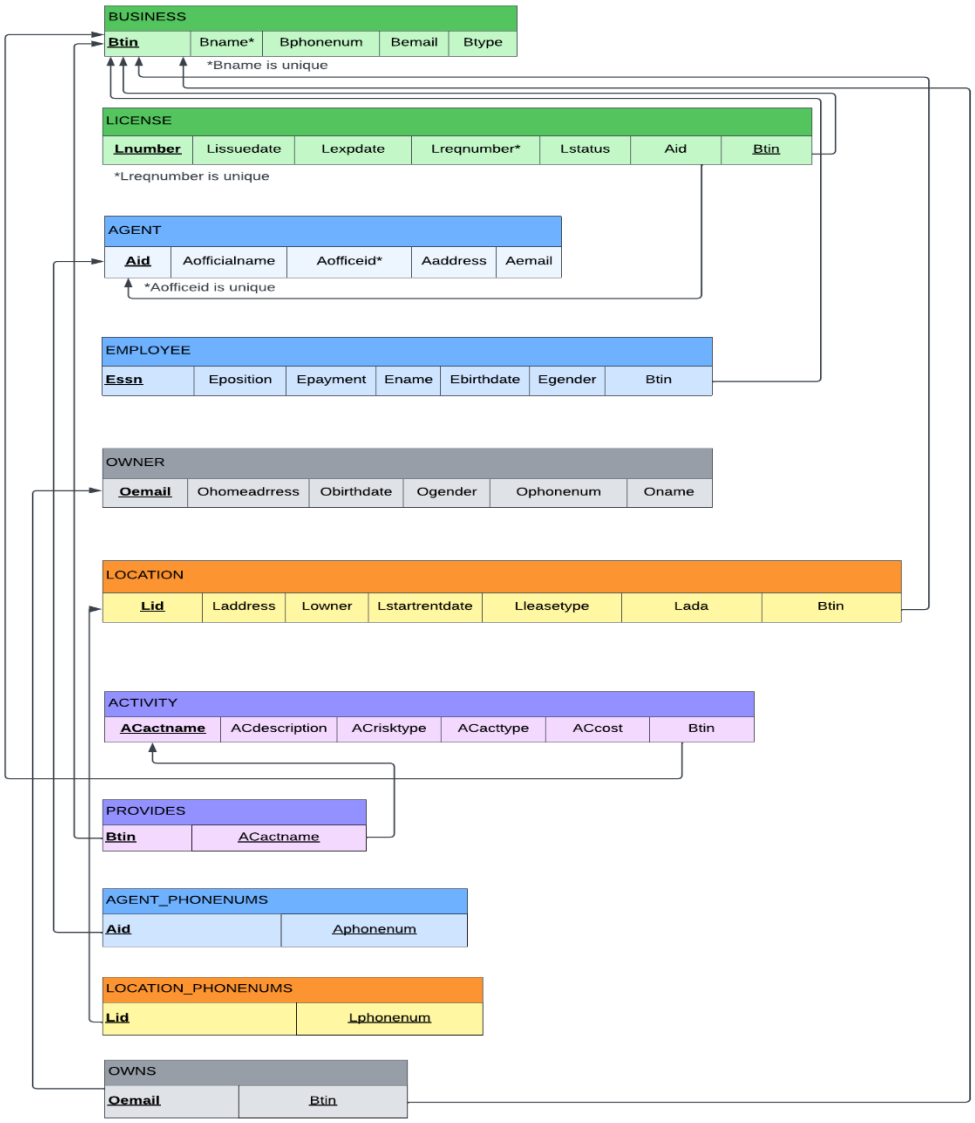


Figure 2. Entity/Relationship Diagram for Tourism Industry

Business and License tables – Paul
Agent, Employee, Agent_PhoneNums – Mustaquim
Owner, Owns – Angel
Location, Location_PhoneNums – Leila
Activity, Provides – Arif

6. NORMALIZED SCHEMA

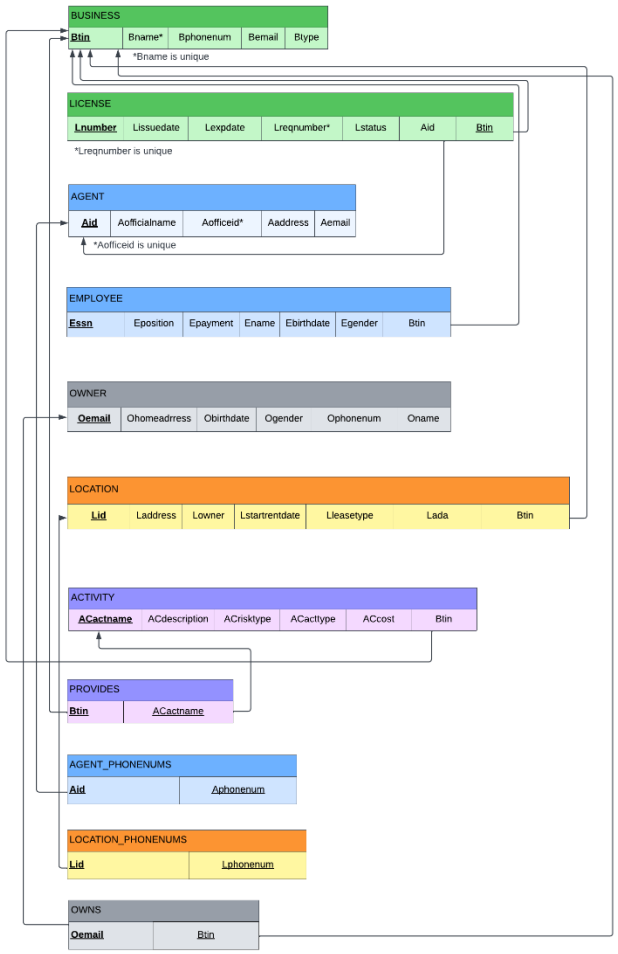


Figure 3. Normalized Schema for Tourism Industry

Business and License tables – Paul
Agent, Employee, Agent_PhoneNums – Mustaquim
Owner, Owns – Angel
Location, Location_PhoneNums – Leila
Activity, Provides – Arif

BUSINESS

Functional dependencies:

| Btin | Bname | Bphonenum | Bemail | Btype |

FD1: {Btin} -> {Btin, Bname, Bphonenum, Bemail, Btype}

Normalization:

- The BUSINESS relation is in First Normal Form (1NF) because all its attributes are atomic.
- The BUSINESS relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key Btin.
- The BUSINESS relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

LICENSE

Functional dependencies:

| Lnumber | Btin | Lissuedate | Lexpdate | Lreqnumber | Lstatus | Aid |

FD1: {Lnumber, Btin} -> {Lissuedate, Lexpdate, Lreqnumber, Lstatus}

Normalization:

- The LICENSE relation is in First Normal Form (1NF) because all its attributes are atomic.
- The LICENSE relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key {Lnumber, Btin}.
- The LICENSE relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

AGENT

Functional dependencies:

| Aid | Aofficialname | Aofficeid | Aaddress| Aemail|

FD1: {Aid} -> {Aid, Aofficialname, Aofficeid, Aaddress, Aemail }

Normalization:

- The AGENT relation is in First Normal Form (1NF) because all its attributes are atomic.
- The AGENT relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key Aid.
- The AGENT relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

EMPLOYEE

Functional dependencies:

| Essn | Eposition| Epayment| Ename| Ebirthdate | Egender | Btin |

FD1: {Essn} -> {Essn, Eposition, Epayment, Ename, Ebirthdate, Egender, Btin}

Normalization:

- The EMPLOYEE relation is in First Normal Form (1NF) because all its attributes are atomic.
- The EMPLOYEE relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key Essn.
- The EMPLOYEE relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

OWNER

Functional dependencies:

| Oemail | Ohomeaddress| Obirthdate| Ogender| Ophonenum| Oname |

FD1: {Oemail} -> {Oemail, Ohomeaddress, Obirthdate, Ogender, Ophonenum, Oname }

Normalization:

- The OWNER relation is in First Normal Form (1NF) because all its attributes are atomic.
- The OWNER relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key Oemail.
- The OWNER relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

LOCATION

Functional dependencies:

| Lid | Laddress| Lowner| Lstartrentdate| Lleasetype| Lada| Btin|

FD1: {Lid} -> {Lid, Laddress, Lowner, Lstartrentdate, Lleasetype, Lada, Btin}

Normalization:

- The LOCATION relation is in First Normal Form (1NF) because all its attributes are atomic.
- The LOCATION relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key Lid.
- The LOCATION relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

ACTIVITY

Functional dependencies:

| ACactname | ACdescription | ACrisktype| ACacttype | ACcost | Btin|

FD1: {ACactname} -> {Acdescription, ACrisktype, ACacttype, ACcost, Btin}

Normalization:

- The ACTIVITY relation is in First Normal Form (1NF) because all its attributes are atomic.
- The ACTIVITY relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key ACactname.
- The ACTIVITY relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key.

PROVIDES

Functional dependencies:

| Btin | Acactname|

FD1: {Btin, ACactname} -> {Btin, ACactname}

Normalization:

- The PROVIDES relation is in First Normal Form (1NF) because all its attributes are atomic.
- The PROVIDES relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key {Btin, ACactname}; there are no non-prime attributes.
- The PROVIDES relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key; there are no non-prime attributes.

AGENT_PHONENUMS

Functional dependencies:

| Aid | Aphonenum|

FD1: {Aid, Aphonenum} -> {Aid, Aphonenum}

Normalization:

- The ACTIVITY relation is in First Normal Form (1NF) because all its attributes are atomic.
- The ACTIVITY relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key {Aid, Aphonenum}; there are no non-prime attributes.
- The ACTIVITY relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key; there are no non-prime attributes.

LOCATION_PHONENUMS

Functional dependencies:

| Lid | Lphonenum|

FD1: {Lid, Lphonenum} -> {Lid, Lphonenum}

Normalization:

- The LOCATION_PHONENUMS relation is in First Normal Form (1NF) because all its attributes are atomic.
- The LOCATION_PHONENUMS relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key {Lid, Lphonenum}; there are no non-prime attributes.
- The LOCATION_PHONENUMS relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key; there are no non-prime attributes.

OWNS

Functional dependencies:

| Oemail | Btin|

FD1: {Oemail, Btin} -> {Oemail, Btin}

Normalization:

- The OWNS relation is in First Normal Form (1NF) because all its attributes are atomic.
- The OWNS relation is in Second Normal From (2NF) because all the non-prime attributes depend fully on the primary key {Oemail, Btin}; there are no non-prime attributes.
- The OWNS relation is in Third Normal From (3NF) because none of the non-prime attributes depend transitively on the primary key; there are no non-prime attributes.

7. Database Schema in MySQL

Creating Tables

List the CREATE TABLE statements in your document.

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SHOW TABLES ;
+--------------------------+
| Tables_in_s23_mjv_team12 |
+--------------------------+
| activity                 |
| agent                    |
| agent_phonenums          |
| business                 |
| employee                 |
| license                  |
| location                 |
| location_phonenums       |
| owner                    |
| owns                     |
| provides                 |
| testgrouptable           |
+--------------------------+
12 rows in set (0.0509 sec)
```

```sql
CREATE TABLE Business (
  Btin INT PRIMARY KEY,
  Bname VARCHAR(255) UNIQUE,
  Btype VARCHAR(255),
  Bphonenum VARCHAR(255),
  Bemail VARCHAR(255)
);

CREATE TABLE License (
  Lnumber INT PRIMARY KEY,
  Lissuedate DATE,
  Lexpdate DATE,
  Lstatus VARCHAR(255),
  Lreqnumber INT UNIQUE,
  Btin INT,
  FOREIGN KEY (Btin) REFERENCES Business(Btin)
);

CREATE TABLE Agent (
  Aid INT PRIMARY KEY,
  Aofficialname VARCHAR(255),
  Aofficeid INT UNIQUE,
  Aaddress VARCHAR(255),
  Aemail VARCHAR(255)
);
```

```sql
CREATE TABLE Employee (
  Ess INT PRIMARY KEY,
  Eposition VARCHAR(255),
  Epayment DECIMAL(10, 2),
  Ename VARCHAR(255),
  Ebirthdate DATE,
  Egender ENUM('M', 'F', 'Other'),
  Btin INT,
  FOREIGN KEY (Btin) REFERENCES Business(Btin)
);

CREATE TABLE Owner (
  Oemail VARCHAR(255) PRIMARY KEY,
  Oname VARCHAR(255),
  Ogender ENUM('M', 'F', 'Other'),
  Obirthdate DATE,
  Ohomeaddress VARCHAR(255),
  Ophonenum VARCHAR(255)
);

CREATE TABLE Location (
  Lid INT PRIMARY KEY,
  Laddress VARCHAR(255),
  Lowner VARCHAR(255),
  Lstartrentdate DATE,
  Lleasetype VARCHAR(255),
  Lada VARCHAR(255),
  Btin INT,
  FOREIGN KEY (Btin) REFERENCES Business(Btin)
);
```

```sql
CREATE TABLE Activity (
  ACactname VARCHAR(255) PRIMARY KEY,
  ACdescription TEXT,
  ACrisktype VARCHAR(255),
  ACacttype VARCHAR(255),
  Accost DECIMAL(10, 2),
  Btin INT,
  FOREIGN KEY (Btin) REFERENCES Business(Btin)
);

CREATE TABLE Provides (
  Btin INT,
  ACname VARCHAR(255),
  PRIMARY KEY (Btin, ACname),
  FOREIGN KEY (Btin) REFERENCES Business(Btin),
  FOREIGN KEY (ACname) REFERENCES Activity(ACactname
);

CREATE TABLE Agent_Phonenums (
  Aid INT,
  Aphonenum VARCHAR(255),
  PRIMARY KEY (Aid, Aphonenum),
  FOREIGN KEY (Aid) REFERENCES Agent(Aid)
);
```

```sql
CREATE TABLE Location_Phonenums (
  Lid INT,
  Lphonenum VARCHAR(255),
  PRIMARY KEY (Lid, Lphonenum),
  FOREIGN KEY (Lid) REFERENCES Location(Lid)
);

CREATE TABLE Owns (
  Oemail VARCHAR(255),
  Btin INT,
  PRIMARY KEY (Oemail, Btin),
  FOREIGN KEY (Oemail) REFERENCES Owner(Oemail),
  FOREIGN KEY (Btin) REFERENCES Business(Btin)
);
```

Figure 4. Creating the Tables

8. Database Records

Inserting Values

Include a sample of database records (at least 3 per table) representative of the domain you are modeling in your team's database in the CS servers.

```sql
-- Business
INSERT INTO Business (Btin, Bname, Btype, Bphonenum, Bemail) VALUES
(4, 'Happy Cafe', 'Cafe', '555-111-4567', 'happycafe@example.com'),
(5, 'Book Barn', 'Bookstore', '555-111-5678', 'bookbarn@example.com'),
(6, 'Pet Palace', 'Pet Store', '555-111-6789', 'petpalace@example.com');

-- License
INSERT INTO License (Lnumber, Lissuedate, Lexpdate, Lstatus, Lreqnumber, Btin) VALUES
(4, '2022-02-01', '2023-02-01', 'Active', 1004, 4),
(5, '2022-05-01', '2023-05-01', 'Active', 1005, 5),
(6, '2022-09-01', '2023-09-01', 'Active', 1006, 6);

-- Agent
INSERT INTO Agent (Aid, Aofficialname, Aofficeid, Aaddress, Aemail) VALUES
(4, 'Emily Wilson', 104, '1100 Main St', 'emilywilson@example.com'),
(5, 'Richard Thompson', 105, '2100 Market St', 'richardthompson@example.com'),
(6, 'Patricia Jackson', 106, '3100 Park Ave', 'patriciajackson@example.com');
```

```sql
-- Employee
INSERT INTO Employee (Ess, Eposition, Epayment, Ename, Ebirthdate, Egender, Btin) VALUES
(4, 'Barista', 28000.00, 'Dan Miller', '1992-03-01', 'M', 4),
(5, 'Sales Associate', 26000.00, 'Eva Garcia', '1993-07-15', 'F', 5),
(6, 'Animal Care Specialist', 30000.00, 'Frank Lee', '1989-11-10', 'M', 6);

-- Owner
INSERT INTO Owner (Oemail, Oname, Ogender, Obirthdate, Ohomeaddress, Ophonenum) VALUES
('owner4@example.com', 'Susan Moore', 'F', '1968-02-01', '1700 Main St', '555-123-7890'),
('owner5@example.com', 'Peter Collins', 'M', '1973-07-15', '1800 Market St', '555-123-8901'),
('owner6@example.com', 'Deborah Edwards', 'F', '1985-11-10', '1900 Park Ave', '555-123-9012');

-- Location
INSERT INTO Location (Lid, Laddress, Lowner, Lstartrentdate, Lleasetype, Lada, Btin) VALUES
(4, '123 Cafe St', 'Sara Brown', '2022-02-01', 'Long term', '12346', 4),
(5, '456 Bookstore Ave', 'Paul Green', '2022-05-01', 'Short term', '23457', 5),
(6, '789 Pet Rd', 'Kim White', '2022-09-01', 'Long term', '34568', 6);
```

```sql
-- Agent_Phonenums
INSERT INTO Agent_Phonenums (Aid, Aphonenum) VALUES
(2, '555-111-0004'),
(3, '555-111-0005'),
(3, '555-111-0006');

-- Location_Phonenums
INSERT INTO Location_Phonenums (Lid, Lphonenum) VALUES
(2, '555-222-0004'),
(3, '555-222-0005'),
(3, '555-222-0006');

-- Owns
INSERT INTO Owns (Oemail, Btin) VALUES
('owner1@example.com', 2),
('owner2@example.com', 3),
('owner3@example.com', 1);
```

```sql
-- Agent_Phoenums
INSERT INTO Agent_Phonenums (Aid, Aphonenum) VALUES
(2, '555-111-0004'),
(3, '555-111-0005'),
(3, '555-111-0006');

-- Location Phonenums
INSERT INTO location_Phonenums (Lid, Lphonenum) VALUES
(2, '555-222-0004'),
(3, '555-222-0005'),
(3, '555-222-0006');

-- Owns
INSERT INTO Owns (Oemail, Btin) VALUES
('owner1@example.com', 2),
('owner2@example.com', 3),
('owner3@example.com', 1);
```

```sql
-- Activity
INSERT INTO Activity (ACactname, ACdescription, ACrisktype, ACacttype, Accost, Btin) VALUES
('Coffee Brewing', 'Preparing coffee beverages', 'Low', 'Food Service',30.00, 4)
('Book Sorting', 'Organizing and categorizing books', 'Low', 'Retail', 25.00, 5),
('Pet Grooming', 'Grooming and styling pets', 'Medium', 'Pet Care', 45.00, 6);

-- Provides
INSERT INTO Provides (Btin, ACname) VALUES
(4, 'Coffee Brewing'),
(5, 'Book Sorting'),
(6, 'Pet Grooming');
```

Show Tables

Business

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM business;
+------+----------------+-------------+--------------+---------------------------+
| Btin | Bname          | Btype       | Bphonenum    | Bemail                    |
+------+----------------+-------------+--------------+---------------------------+
|    1 | Tech Solutions | IT Services | 555-111-1234 | techsolutions@example.com |
|    2 | Green Grocer   | Grocery     | 555-111-2345 | greengrocer@example.com   |
|    3 | City Gym       | Fitness     | 555-111-3456 | citygym@example.com       |
+------+----------------+-------------+--------------+---------------------------+
```

License

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM license
                                                        -> ;
+---------+------------+------------+---------+------------+------+------+
| Lnumber | Lissuedate | Lexpdate   | Lstatus | Lreqnumber | Aid  | Btin |
+---------+------------+------------+---------+------------+------+------+
|       1 | 2022-01-01 | 2023-01-01 | Active  | 1001       | NULL |    1 |
|       2 | 2022-04-01 | 2023-04-01 | Active  | 1002       | NULL |    2 |
|       3 | 2022-08-01 | 2023-08-01 | Active  | 1003       | NULL |    3 |
+---------+------------+------------+---------+------------+------+------+
```

Agent

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM agent;
+-----+---------------+-----------+--------------+-----------------------+
| Aid | Aofficialname | Aofficeid | Aaddress     | Aemail                |
+-----+---------------+-----------+--------------+-----------------------+
|   1 | John Smith    |       101 | 123 Main St  | johnsmith@example.com |
|   2 | Jane Doe      |       102 | 456 Market St | janedoe@example.com  |
|   3 | James Brown   |       103 | 789 Park Ave | jamesbrown@example.com |
+-----+---------------+-----------+--------------+-----------------------+
```

Employee

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM employee;
+-----+--------------------+----------+---------------+------------+---------+------+
| Ess | Eposition          | Epayment | Ename         | Ebirthdate | Egender | Btin |
+-----+--------------------+----------+---------------+------------+---------+------+
|   1 | Software Developer | 60000.00 | Alice Johnson | 1990-01-01 | F       |    1 |
|   2 | Cashier            | 24000.00 | Bob Jones     | 1995-02-15 | M       |    2 |
|   3 | Fitness Instructor | 35000.00 | Carol Davis   | 1988-06-10 | F       |    3 |
+-----+--------------------+----------+---------------+------------+---------+------+
```

Owner

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM owner;
+--------------------+--------------+---------+------------+---------------+--------------+
| Oemail             | Oname        | Ogender | Obirthdate | Ohomeaddress  | Ophonenum    |
+--------------------+--------------+---------+------------+---------------+--------------+
| owner1@example.com | George Martin | M       | 1975-01-01 | 1600 Main St  | 555-123-4567 |
| owner2@example.com | Paula White  | F       | 1980-05-15 | 1200 Market St | 555-123-5678 |
| owner3@example.com | Linda Taylor | F       | 1972-12-20 | 1000 Park Ave | 555-123-6789 |
+--------------------+--------------+---------+------------+---------------+--------------+
```

Location

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM location;
+-----+-----------------+------------+----------------+------------+--------+------+
| Lid | Laddress        | Lowner     | Lstartrentdate | Lleasetype | Lada   | Btin |
+-----+-----------------+------------+----------------+------------+--------+------+
|   1 | 123 Tech St     | John Doe   | 2022-01-01     | Long term  | 12345  |    1 |
|   2 | 456 Grocery Ave | Jane Smith | 2022-04-01     | Short term | 23456  |    2 |
|   3 | 789 Gym Rd      | Jim Brown  | 2022-08-01     | Long term  | 34567  |    3 |
+-----+-----------------+------------+----------------+------------+--------+------+
```

Activity

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM activity;
+---------------------+---------------------------------------+-----------+-----------+--------+------+
| ACactname           | ACdescription                         | ACrisktype | ACacttype | Accost | Btin |
+---------------------+---------------------------------------+-----------+-----------+--------+------+
| Group Fitness Classes | Leading group fitness classes       | Medium     | Fitness   |  60.00 |    3 |
| Network Security    | Securing computer networks            | Medium     | IT        | 200.00 |    1 |
| Produce Handling    | Handling and maintaining fresh produce | Low        | Retail    |  40.00 |    2 |
+---------------------+---------------------------------------+-----------+-----------+--------+------+
```

Provides

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM provides;
+------+----------------------+
| Btin | ACname               |
+------+----------------------+
|    3 | Group Fitness Classes |
|    1 | Network Security     |
|    2 | Produce Handling     |
+------+----------------------+
```

Agent_PhoneNums

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM agent_phonenums;
+-----+--------------+
| Aid | Aphonenum    |
+-----+--------------+
|   2 | 555-111-0004 |
|   3 | 555-111-0005 |
|   3 | 555-111-0006 |
+-----+--------------+
```

Location_PhoneNums

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM Location_PhoneNums;
+-----+--------------+
| Lid | Lphonenum    |
+-----+--------------+
|   2 | 555-222-0004 |
|   3 | 555-222-0005 |
|   3 | 555-222-0006 |
+-----+--------------+
```

Owns

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT * FROM owns;
+--------------------+------+
| Oemail             | Btin |
+--------------------+------+
| owner3@example.com |    1 |
| owner1@example.com |    2 |
| owner2@example.com |    3 |
+--------------------+------+
```

Figure 6. Showing the Tables

9. SQL Queries

Requirement Queries

Include the MySQL queries required to satisfy your functional requirements. Trace back to each of your functional requirements and show how you satisfy them. Note that one query may satisfy more than one functional requirement.

1. **(LEILA)** To retrieve the number of employees for a particular business:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT COUNT(*) FROM Employee JO
IN business on employee.Btin=business.Btin where business.Bname='Tech Solutions';
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
```

Figure 9.1

2. **(LEILA)** To retrieve the payment information for a particular position:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT Epayment FROM Employee WH
ERE Eposition='Cashier';
+----------+
| Epayment |
+----------+
| 24000.00 |
+----------+
```

Figure 9.2

3. **(MUSTAQUIM)** To retrieve all locations for a particular business:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT location.laddress FROM Lo
cation natural JOIN business WHERE business.Bname = 'Tech Solutions';
+-------------+
| laddress    |
+-------------+
| 123 Tech St |
+-------------+
```

Figure 9.3

4. **(MUSTAQUIM)** List all businesses with their owners:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Bname, o.Oname
                                                        -> FROM Business b
                                                        -> JOIN Owns ON b.Btin = Owns.Btin
                                                        -> JOIN Owner o ON Owns.Oemail = o.
Oemail;
+----------------+---------------+
| Bname          | Oname         |
+----------------+---------------+
| City Gym       | Paula White   |
| Green Grocer   | George Martin |
| Tech Solutions | Linda Taylor  |
+----------------+---------------+
```

Figure 9.4

5. **(ANGEL)** Show the total payment for each type of business:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Btype, SUM(e.Epayment)
AS TotalPayment
                                                          -> FROM Business b
                                                          -> JOIN Employee e ON b.Btin = e.Bt
in
                                                          -> GROUP BY b.Btype;
+-------------+--------------+
| Btype       | TotalPayment |
+-------------+--------------+
| IT Services |     60000.00 |
| Grocery     |     24000.00 |
| Fitness     |     35000.00 |
+-------------+--------------+
```

Figure 9.5

6. **(ANGEL)** List all businesses and their activities:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Bname, a.ACactname
                                                          -> FROM Business b
                                                          -> JOIN Provides p ON b.Btin = p.Bt
in
                                                          -> JOIN Activity a ON p.ACname = a.
ACactname;
+---------------+-----------------------+
| Bname         | ACactname             |
+---------------+-----------------------+
| City Gym      | Group Fitness Classes |
| Tech Solutions | Network Security      |
| Green Grocer  | Produce Handling      |
+---------------+-----------------------+
3 rows in set (0.0758 sec)
```

Figure 9.6

7. **(PAUL)** Find businesses with expired licenses:

```
3 rows in set (0.0430 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Bname, l.Lexpdate
                                                          -> FROM Business b
                                                          -> JOIN License l ON b.Btin = l.Bti
n
                                                          -> WHERE l.Lexpdate = '2023-01-01';

+---------------+------------+
| Bname         | Lexpdate   |
+---------------+------------+
| Tech Solutions | 2023-01-01 |
+---------------+------------+
1 row in set (0.0534 sec)
```

Figure 9.7

8. **(PAUL)** Show the number of employees per business:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Bname, COUNT(e.Ess) AS
NumberOfEmployees
                                                          -> FROM Business b
                                                          -> JOIN Employee e ON b.Btin = e.Bt
in
                                                          -> GROUP BY b.Bname;
+---------------+-------------------+
| Bname         | NumberOfEmployees |
+---------------+-------------------+
| City Gym      |                 1 |
| Green Grocer  |                 1 |
| Tech Solutions |                 1 |
+---------------+-------------------+
```

Figure 9.8

9. **(ARIF)** List activities with risk type 'Medium' or higher:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT ACactname, ACdescription
                                                          -> FROM Activity
                                                          -> WHERE ACrisktype IN ('Medium', '
High');
+----------------------+------------------------------+
| ACactname            | ACdescription                |
+----------------------+------------------------------+
| Group Fitness Classes | Leading group fitness classes |
| Network Security      | Securing computer networks    |
+----------------------+------------------------------+
```

Figure 9.9

10. **(ARIF)** Find the most expensive activity per business:

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > SELECT b.Bname, a.ACactname, a.A
ccost
                                                          -> FROM Business b
                                                          -> JOIN Provides p ON b.Btin = p.Bt
in
                                                          -> JOIN Activity a ON p.ACname = a.
ACactname
                                                          -> WHERE (b.Btin, a.Accost) IN (
                                                          ->     SELECT Btin, MAX(Accost)
                                                          ->     FROM Activity
                                                          ->     GROUP BY Btin
                                                          -> );
+---------------+----------------------+--------+
| Bname         | ACactname            | Accost |
+---------------+----------------------+--------+
| City Gym      | Group Fitness Classes |  60.00 |
| Tech Solutions | Network Security      | 200.00 |
| Green Grocer  | Produce Handling     |  40.00 |
+---------------+----------------------+--------+
3 rows in set (0.0423 sec)
```

Figure 9.10

10. Views

1. **(Paul)** ActiveLicenses: A view that shows all active licenses.

```
CREATE VIEW ActiveLicenses AS
SELECT B.Bname, L.Lnumber, L.Lissuedate, L.Lexpdate
FROM License L
JOIN Business B ON B.Btin = L.Btin
WHERE L.Lstatus = 'Active';
```

```
Query OK, 0 rows affected (0.1468 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > select * From ActiveLicenses;
+----------------------+---------+------------+------------+
| Bname                | Lnumber | Lissuedate | Lexpdate   |
+----------------------+---------+------------+------------+
| Tech Solutions       |       1 | 2022-01-01 | 2023-01-01 |
| Green Grocer         |       2 | 2022-04-01 | 2023-04-01 |
| City Gym             |       3 | 2022-08-01 | 2023-08-01 |
| ABC Corporation      |     101 | 2021-01-01 | 2022-01-01 |
| XYZ Inc.             |     102 | 2021-02-01 | 2023-02-01 |
| Sunshine Bakery      |     104 | 2021-04-01 | 2022-04-01 |
| Johnson & Sons       |     105 | 2021-05-01 | 2023-05-01 |
| Bright Ideas         |     106 | 2021-06-01 | 2022-06-01 |
| ABC Realty           |     108 | 2021-08-01 | 2023-08-01 |
| Innovative Solutions |     109 | 2021-09-01 | 2022-09-01 |
| Happy Paws           |     111 | 2021-11-01 | 2023-11-01 |
| Sunny Skies Travel   |     112 | 2021-12-01 | 2022-12-01 |
+----------------------+---------+------------+------------+
```

2. **(Angel)** BusinessActivities: A view that shows all businesses and their associated activities.

```
CREATE VIEW BusinessActivities AS
SELECT B.Bname, A.ACactname
FROM Provides P
JOIN Business B ON B.Btin = P.Btin
JOIN Activity A ON A.ACactname = P.ACname;
```

```
Query OK, 0 rows affected (0.1942 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > select * From BusinessActivities;
+------------------+------------------------+
| Bname            | ACactname              |
+------------------+------------------------+
| Sunshine Bakery  | Bakery                 |
| Johnson & Sons   | Construction Services  |
| Bright Ideas     | Content Marketing      |
| City Gym         | Fitness Classes        |
| City Gym         | Group Fitness Classes  |
| Smith & Co.      | Legal Services         |
| Tech Solutions   | Mobile App Development |
| Tech Solutions   | Network Security       |
| City Gym         | Personal Training      |
| Happy Paws       | Pet Grooming           |
| Green Grocer     | Produce Handling       |
| ABC Corporation  | Retail Sales           |
| Bright Ideas     | Social Media Management |
| Acme Enterprises | Software Development   |
| Tech Solutions   | Web Development        |
+------------------+------------------------+
```

3. **(Leila)** EmployeeCountByBusiness: A view that displays the number of employees for each business.

```
CREATE VIEW EmployeeCountByBusiness AS
SELECT B.Bname, COUNT(*) AS EmployeeCount
FROM Employee E
JOIN Business B ON B.Btin = E.Btin
GROUP BY B.Bname;
```

```
Query OK, 0 rows affected (0.1412 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > select * From EmployeeCountByBusiness;
+----------------------+---------------+
| Bname                | EmployeeCount |
+----------------------+---------------+
| Tech Solutions       |             1 |
| Green Grocer         |             1 |
| City Gym             |             1 |
| ABC Corporation      |             1 |
| XYZ Inc.             |             1 |
| Acme Enterprises     |             1 |
| Sunshine Bakery      |             1 |
| Johnson & Sons       |             1 |
| Bright Ideas         |             1 |
| Global Logistics     |             1 |
| ABC Realty           |             1 |
| Innovative Solutions |             1 |
| Smith & Co.          |             1 |
| Happy Paws           |             1 |
| Sunny Skies Travel   |             1 |
+----------------------+---------------+
```

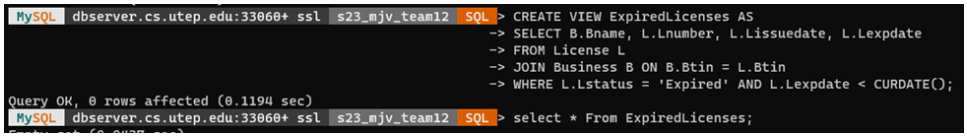
4. **(Mustaquim)** BusinessOwnerInfo: A view that shows business owner information for each business.

```
CREATE VIEW BusinessOwnerInfo AS
SELECT B.Bname, O.Oname, O.Oemail
FROM Owns OW
JOIN Owner O ON O.Oemail = OW.Oemail
JOIN Business B ON B.Btin = OW.Btin;
```

```
Query OK, 0 rows affected (0.1744 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > select * From  BusinessOwnerInfo;
+----------------------+-----------------+---------------------------+
| Bname                | Oname           | Oemail                    |
+----------------------+-----------------+---------------------------+
| Tech Solutions       | Ava Davis       | ava.davis@hotmail.com     |
| Tech Solutions       | Linda Taylor    | owner3@example.com        |
| Green Grocer         | Daniel Kim      | daniel.kim@yahoo.com      |
| Green Grocer         | George Martin   | owner1@example.com        |
| City Gym             | David Lee       | david.lee@hotmail.com     |
| City Gym             | Paula White     | owner2@example.com        |
| ABC Corporation      | Emily Chen      | emily.chen@gmail.com      |
| XYZ Inc.             | Jennifer Garcia | jennifer.garcia@yahoo.com |
| Acme Enterprises     | John Smith      | john.smith@gmail.com      |
| Sunshine Bakery      | Julia Davis     | julia.davis@hotmail.com   |
| Johnson & Sons       | Karen Williams  | karen.williams@gmail.com  |
| Bright Ideas         | Maria Perez     | maria.perez@gmail.com     |
| Global Logistics     | Michael Brown   | michael.brown@hotmail.com |
| ABC Realty           | Olivia Nguyen   | olivia.nguyen@gmail.com   |
| Innovative Solutions | Richard Martinez| richard.martinez@yahoo.com|
+----------------------+-----------------+---------------------------+
15 rows in set (0.0905 sec)
```

5. **(Arif)** ExpiredLicenses: A view that shows all expired licenses.

```
CREATE VIEW ExpiredLicenses AS
SELECT B.Bname, L.Lnumber, L.Lissuedate, L.Lexpdate
FROM License L
JOIN Business B ON B.Btin = L.Btin
WHERE L.Lstatus = 'Expired' AND L.Lexpdate < CURDATE();
```



11. Procedures

1. **(Angel)** AddBusiness: A procedure to add a new business.

```
DELIMITER //
CREATE PROCEDURE AddBusiness(IN p_Bname VARCHAR(255),
IN p_Btype VARCHAR(255), IN p_Bphonenum VARCHAR(255),
IN p_Bemail VARCHAR(255))
BEGIN
  INSERT INTO Business(Bname, Btype, Bphonenum, Bemail)
  VALUES (p_Bname, p_Btype, p_Bphonenum, p_Bemail);
END //
DELIMITER ;
```

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER //
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > CREATE PROCEDURE AddBusiness(IN p_Bname VARCHAR(255), IN p_Btype
 VARCHAR(255), IN p_Bphonenum VARCHAR(255), IN p_Bemail VARCHAR(255))
                                                        -> BEGIN
                                                        ->   INSERT INTO Business(Bname, Btype, Bphonenum, Bemail)
                                                        ->   VALUES (p_Bname, p_Btype, p_Bphonenum, p_Bemail);
                                                        -> END //
Query OK, 0 rows affected (0.1986 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER ;
```

2. **(Paul)** UpdateLicenseStatus: A procedure to update the status of a license.

```
DELIMITER //
CREATE PROCEDURE UpdateLicenseStatus(IN p_Lnumber INT,
IN p_Lstatus VARCHAR(255))
BEGIN
  UPDATE License
  SET Lstatus = p_Lstatus
  WHERE Lnumber = p_Lnumber;
END //
DELIMITER ;
```

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER //
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > CREATE PROCEDURE UpdateLicenseStatus(IN p_Lnumber INT, IN p_Lsta
tus VARCHAR(255))
                                                        -> BEGIN
                                                        ->   UPDATE License
                                                        ->   SET Lstatus = p_Lstatus
                                                        ->   WHERE Lnumber = p_Lnumber;
                                                        -> END //
Query OK, 0 rows affected (0.1761 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER ;
```

3. **(Arif)** AddEmployee: A procedure to add a new employee to a business.

```
DELIMITER //
CREATE PROCEDURE AddEmployee(IN p_Ess INT, IN p_Eposition VARCHAR(255),
IN p_Epayment DECIMAL(10, 2),
IN p_Ename VARCHAR(255),
IN p_Ebirthdate DATE,
IN p_Egender ENUM('M', 'F', 'Other'),
IN p_Btin INT)
BEGIN
  INSERT INTO Employee(Ess, Eposition, Epayment, Ename, Ebirthdate, Egender, Btin)
  VALUES (p_Ess, p_Eposition, p_Epayment, p_Ename, p_Ebirthdate, p_Egender, p_Btin);
END //
DELIMITER ;
```

```
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER //
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > CREATE PROCEDURE AddEmployee(IN p_Ess INT, IN p_Eposition VARCHA
R(255), IN p_Epayment DECIMAL(10, 2), IN p_Ename VARCHAR(255), IN p_Ebirthdate DATE, IN p_Egender ENUM('M', 'F', 'Other'), IN p
_Btin INT)
                                                    -> BEGIN
                                                    ->   INSERT INTO Employee(Ess, Eposition, Epayment, Ename, Ebirthda
te, Egender, Btin)
                                                    ->   VALUES (p_Ess, p_Eposition, p_Epayment, p_Ename, p_Ebirthdate,
 p_Egender, p_Btin);
                                                    -> END //
Query OK, 0 rows affected (0.2254 sec)
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER ;
```

4. **(Mustaquim)** AssignActivityToBusiness: A procedure to assign an activity to a business.

```
DELIMITER //
CREATE PROCEDURE AssignActivityToBusiness(IN p_Btin INT, IN p_ACname VARCHAR(255))
BEGIN
  INSERT INTO Provides(Btin, ACname)
  VALUES (p_Btin, p_ACname);
END //
DELIMITER ;
```

```
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER //
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > CREATE PROCEDURE AssignActivityToBusiness(IN p_Btin INT, IN p_AC
name VARCHAR(255))
                                                    -> BEGIN
                                                    ->   INSERT INTO Provides(Btin, ACname)
                                                    ->   VALUES (p_Btin, p_ACname);
                                                    -> END //
Query OK, 0 rows affected (0.2138 sec)
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER ;
```

5. **(Paul)** AddOwnerToBusiness: A procedure to add an owner to a business.

```
DELIMITER //
CREATE PROCEDURE AddOwnerToBusiness(IN p_Oemail VARCHAR(255), IN p_Btin INT)
BEGIN
  INSERT INTO Owns(Oemail, Btin)
  VALUES (p_Oemail, p_Btin);
END //
DELIMITER ;
```

```
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER //
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > CREATE PROCEDURE AddOwnerToBusiness(IN p_Oemail VARCHAR(255), IN
p_Btin INT)
                                                    -> BEGIN
                                                    ->   INSERT INTO Owns(Oemail, Btin)
                                                    ->   VALUES (p_Oemail, p_Btin);
                                                    -> END //
Query OK, 0 rows affected (0.2030 sec)
MySQL  dbserver.cs.utep.edu:33060+ ssl  s23_mjv_team12  SQL > DELIMITER ;
```

6. **(Leila)** numBusiness: A procedure to count how many businesses in Table Business

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER $
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > CREATE PROCEDURE numBusiness(IN bussIn char(50))
                                                        -> BEGIN
                                                        -> SELECT COUNT(*) FROM business;
                                                        -> END $
Query OK, 0 rows affected (0.2398 sec)
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > DELIMITER ;
```

```
MySQL dbserver.cs.utep.edu:33060+ ssl s23_mjv_team12 SQL > CALL numBusiness('El Paso');
+----------+
| COUNT(*) |
+----------+
|       16 |
+----------+
1 row in set (0.0478 sec)
```

12. Triggers

(Done in IBM DB2)

***Trigger 1 (Paul Djangang) :** will prevent new businesses from being added if they have the same phone number as an existing business. This ensures that each business has a unique phone number in the **Business** table.

```
DELIMITER $$

CREATE TRIGGER business_before_insert
BEFORE INSERT ON Business
FOR EACH ROW
BEGIN
    DECLARE cnt INT;
    SELECT COUNT(*) INTO cnt FROM Business WHERE Bphonenum = NEW.Bphonenum;
    IF cnt > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Error: Duplicate phone number not allowed.';
    END IF;
END;
$$

DELIMITER ;
```

Let's run the following command:

```
INSERT INTO Business (Btin, Bname, Btype, Bphonenum, Bemail)
VALUES (17, 'Test Business', 'Test Type', '555-111-1234', 'test@test.com');
```

Since the phone number '555-111-1234' already exists in the **Business** table, the trigger **business_before_insert** would interrupt the insertion and raise an error.

```
Error: Duplicate phone number not allowed.
```

***Trigger 2 (Angel Otero Montanez) :** Ensure the license expiration date (**Lexpdate**) is always later than the issue date (**Lissuedate**).

```
DELIMITER //

CREATE TRIGGER license_date_check_before_insert
BEFORE INSERT ON License
FOR EACH ROW
BEGIN
    IF NEW.Lexpdate <= NEW.Lissuedate THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Error: License expiration date must be later than the issue date.';
    END IF;
END; //

DELIMITER ;
```

Let’s run the following command:

```
INSERT INTO License (Lnumber, Lissuedate, Lexpdate, Lstatus, Lreqnumber, Aid, Btin)
VALUES (114, '2023-04-01', '2023-03-01', 'Active', 1014, NULL, 2);
```

when you try to insert a new license where the expiration date is not later than the issue date, the trigger will prevent the **INSERT** operation and raise an error:

```
Error: License expiration date must be later than the issue date.
```

***Trigger 3 (Leila Martinez) :** Ensures the **Aofficeid** is a positive number when inserting a new agent into the **Agent** table.

```
DELIMITER //

CREATE TRIGGER agent_officeid_check_before_insert
BEFORE INSERT ON Agent
FOR EACH ROW
BEGIN
    IF NEW.Aofficeid <= 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Error: Agent office ID must be a positive number.';
    END IF;
END; //

DELIMITER ;
```

Let's run the following command:

```
INSERT INTO Agent (Aid, Aofficialname, Aofficeid, Aaddress, Aemail)
VALUES (49, 'Test Agent', 0, 'Test Address', 'test@test.com');
```

when you try to insert a new agent with a non-positive office ID, the trigger will prevent the **INSERT** operation and raise an error:

```
Error: Agent office ID must be a positive number.
```

***Trigger 4 (Mohammad Ariful Islam Khan)** : prevents employees under the age of 16 from being added to the **Employee** table:

```
DELIMITER //

CREATE TRIGGER check_age_before_insert
BEFORE INSERT ON Employee
FOR EACH ROW
BEGIN
    IF TIMESTAMPDIFF(YEAR, NEW.Ebirthdate, CURDATE()) < 16 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Error: Employee must be at least 16 years old.';
    END IF;
END; //

DELIMITER ;
```

Let's run the following command:

```
INSERT INTO Employee (Ess, Eposition, Epayment, Ename, Ebirthdate, Egender, Btin)
VALUES (100000014, 'Test Position', 30000, 'Test Name', '2020-01-01', 'M', 1);
```

when you try to insert an employee under the age of 16, the trigger will prevent the **INSERT** operation and raise an error:

```
Error: Employee must be at least 16 years old.
```

Trigger 5 (S M Mustaquim): checks if the **Lstartrentdate** is not in the future

```
DELIMITER //

CREATE TRIGGER check_lease_start_date_before_insert_update
BEFORE INSERT ON Location
FOR EACH ROW
BEGIN
    IF NEW.Lstartrentdate > CURRENT_DATE THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Error: Lease start date cannot be in the future.';
    END IF;
END; //

DELIMITER ;
```

Let's run the following command:

```
INSERT INTO Location
(Lid, Laddress, Lowner, Lstartrentdate, Lleasetype, Lada, Btin)
VALUES
(19, '300 Future Ave', 'Time Traveler', '2023-12-01', 'Long term', '12345', 1);
```

Since the **Lstartrentdate** ('2023-12-01') is in the future (assuming the current date is before '2023-12-01'), this INSERT statement would violate the trigger condition, and MySQL will raise an error:

```
Error: Lease start date cannot be in the future.
```

13. Reports

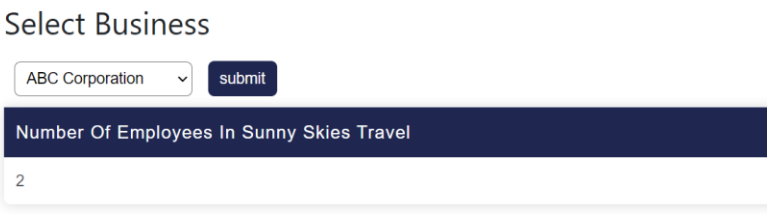


Figure 13.1

The system shall show the number of employees per business

Business	Number Of Employees
ABC Corporation	4
ABC Realty	3
Acme Enterprises	3
Bright Ideas	3
City Gym	4
Global Logistics	3
Green Grocer	1
Happy Paws	2
Innovative Solutions	3
Johnson & Sons	3
Smith & Co.	3
Sunny Skies Travel	2
Sunshine Bakery	3
Tech Solutions	5
XYZ Inc.	3

Figure 13.2

The system shall retrieve the payment information for a particular position

Figure 13.3

The system shall retrieve all locations for a particular business

ABC Corporation submit

Locations For Bright Ideas
135 Maple Lane

Figure 13.4

The system shall List all businesses with their owners

Business	Owner
Tech Solutions	Ava Davis
Tech Solutions	Linda Taylor
Green Grocer	Daniel Kim
Green Grocer	George Martin
City Gym	David Lee
City Gym	Paula White
ABC Corporation	Emily Chen
XYZ Inc.	Jennifer Garcia
Acme Enterprises	John Smith
Sunshine Bakery	Julia Davis
Johnson & Sons	Karen Williams
Bright Ideas	Maria Perez
Global Logistics	Michael Brown
ABC Realty	Olivia Nguyen
Innovative Solutions	Richard Martinez

Figure 13.5

The system shall show the total payment for each type of business

Business Type	Total Payment
IT Services	300000.00
Grocery	24000.00
Fitness	185000.00
Retail	230000.00
Technology	47000.00
Manufacturing	132000.00
Food & Beverage	43000.00
Construction	55000.00
Marketing	105000.00
Transportation	39000.00
Real Estate	120000.00
Consulting	68000.00
Legal	41000.00
Pet Services	56000.00
Travel	81000.00

Figure 13.6

The system shall list all businesses and their activities

Business	Activity Type
Sunshine Bakery	Bakery
Johnson & Sons	Construction Services
Bright Ideas	Content Marketing
City Gym	Fitness Classes
City Gym	Group Fitness Classes
Smith & Co.	Legal Services
Tech Solutions	Mobile App Development
Tech Solutions	Network Security
City Gym	Personal Training
Happy Paws	Pet Grooming
Green Grocer	Produce Handling
ABC Corporation	Retail Sales
Bright Ideas	Social Media Management
Acme Enterprises	Software Development
Tech Solutions	Web Development

Figure 13.7

The system shall find businesses with expired licenses

Businesses With Expired Licenses	Expiration Date
Tech Solutions	2023-01-01

Figure 13.8

The system shall list activities with risk type 'Medium' or higher

Activities With Meduim To High Risk	Description
Construction Services	Construction and renovation services for residential and commercial properties
Group Fitness Classes	Leading group fitness classes
Legal Services	Legal representation for individuals and businesses in a variety of areas including litigation, real estate, and corporate law
Mobile App Development	Design and development of mobile applications for businesses and organizations
Network Security	Securing computer networks
Personal Training	One-on-one fitness training with a certified personal trainer
Software Development	Design and development of custom software solutions for businesses and organizations

Figure 13.9

The system shall find the most expensive activity per business

Business	Activity	Cost
Sunshine Bakery	Bakery	5000.00
Johnson & Sons	Construction Services	10000.00
Bright Ideas	Content Marketing	1000.00
City Gym	Group Fitness Classes	60.00
Smith & Co.	Legal Services	250.00
Tech Solutions	Mobile App Development	2500.00
Happy Paws	Pet Grooming	30.00
Green Grocer	Produce Handling	40.00
ABC Corporation	Retail Sales	10000.00
Acme Enterprises	Software Development	5000.00

Figure 13.10

14. Requirements Tracing

Requirement	Addressed By	Created By
R1. The system shall retrieve the number of employees for a particular business	Query 9.1 Interface Figure 13.1	LEILA
R2. The system shall retrieve the payment information for a particular position	Query 9.2 Interface Figure 13.2	LEILA
R3. The system shall retrieve all locations for a particular business	Query 9.3 Interface Figure 13.3	MUSTAQUIM
R4. The system shall List all businesses with their owners	Query 9.4 Interface Figure 13.4	MUSTAQUIM
R5. The system shall show the total payment for each type of business	Query 9.5 Interface Figure 13.5	ANGEL
R6. The system shall list all businesses and their activities	Query 9.6 Interface Figure 13.6	ANGEL
R7. The system shall find businesses with expired licenses	Query 9.7 Interface Figure 13.7	PAUL
R8. The system shall show the number of employees per business	Query 9.8 Interface Figure 13.8	PAUL
R9. The system shall list activities with risk type 'Medium' or higher	Query 9.9 Interface Figure 13.9	ARIF
R10. The system shall find the most expensive activity per business	Query 9.10 Interface Figure 13.10	ARIF

15. Graphical User Interface (GUI)

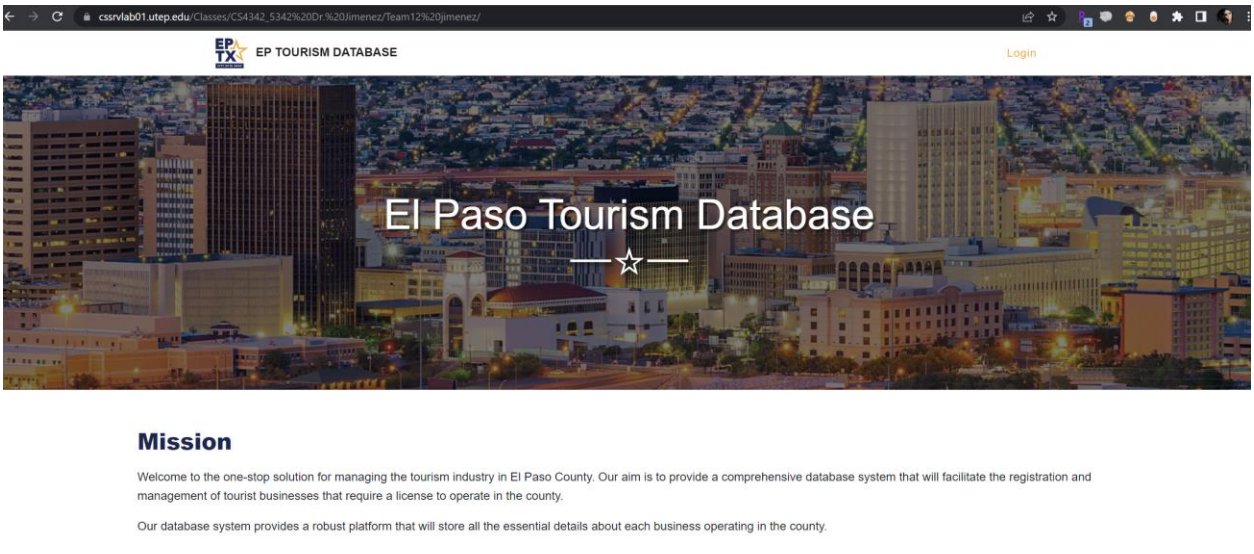


Figure 13.1 Main Website

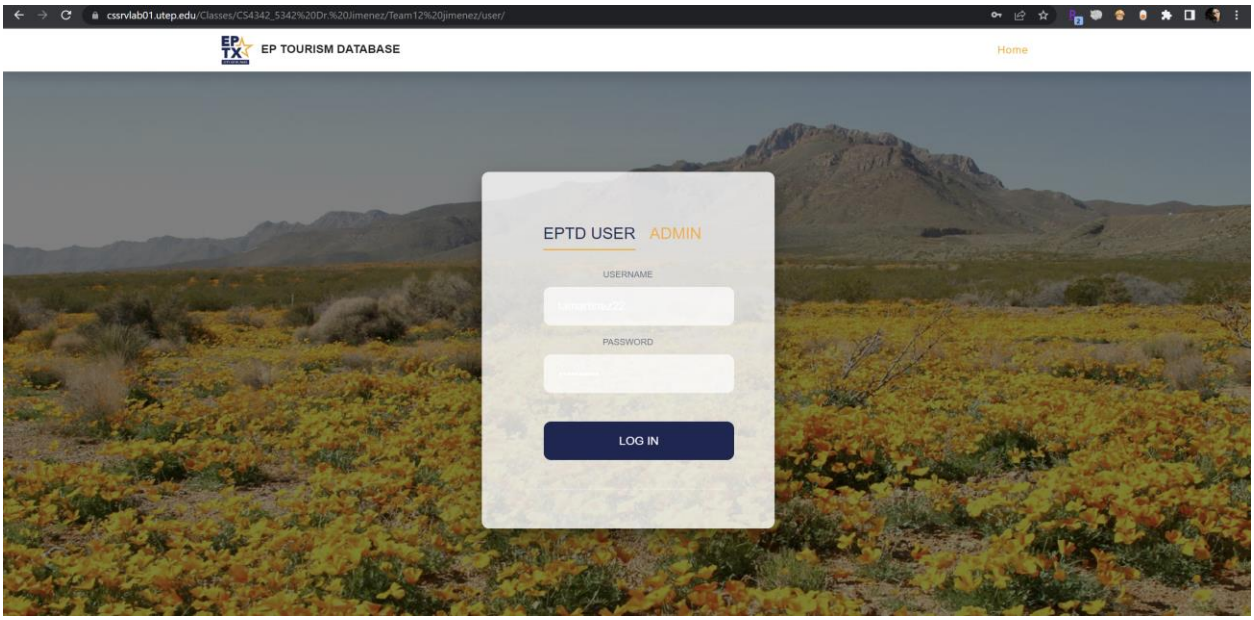
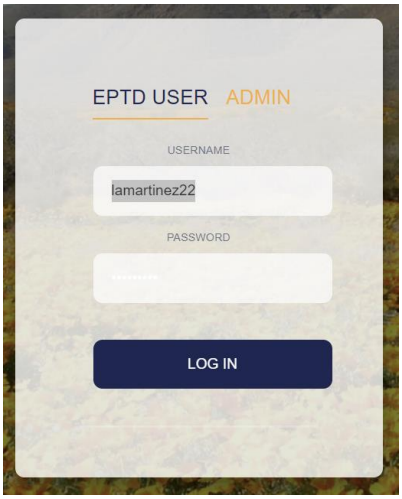


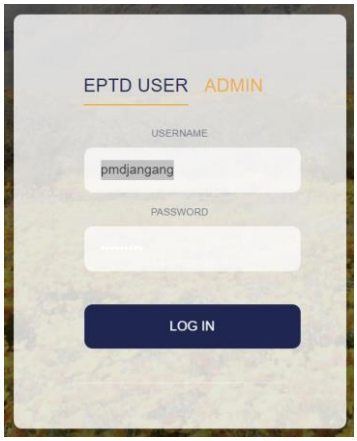
Figure 13.2 Log in

16. IIS Web Server and GUI

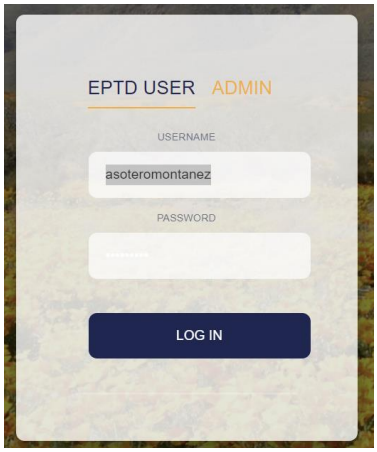
a. Log in for Leila



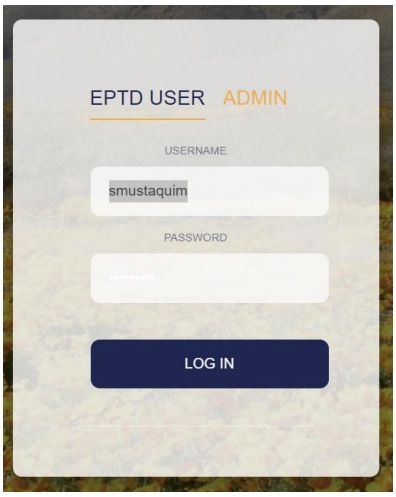
b. Log in for Paul



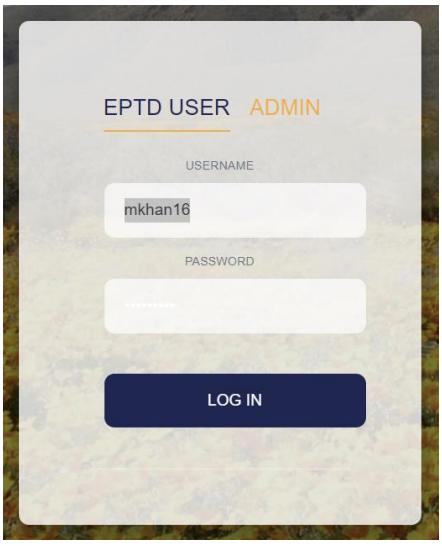
c. Log in for Angel



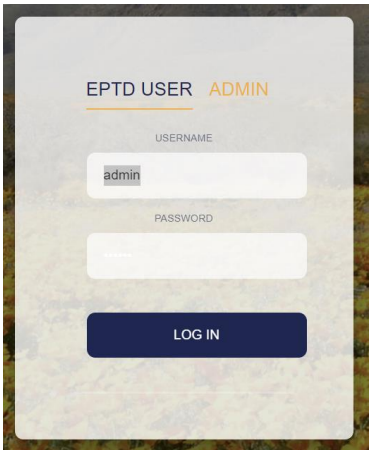
d. Log in for Mustaquim



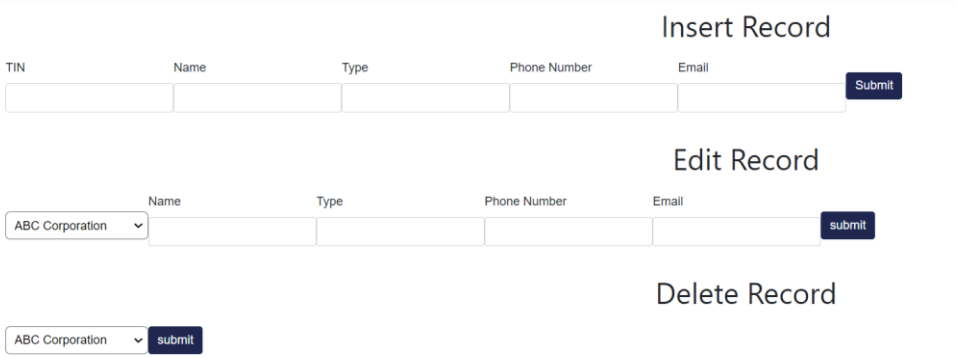
e. Log in for Arif



f. Log in for Admin



f. Add, delete, update



17. References

1. Elmasri, R., & Navathe, S. (2016). Fundamentals of Database Systems. Pearson.
2. Dr. Maria and Dr. Villanueva

APPENDIX A. ATTRIBUTION INFORMATION

Leila

- Contributed to overall Scope.
- Contributed to adding requirements 8 and 9.
- Contributed to creating Agent entity and attributes to the ER.
- Contributed to creating Location and Location_PhoneNums in the Relational Model
- Contributed to creating the Normalization for tables Location and Location_PhonenUms and adding their description.
- Contributed to creating Location and Location_PhoneNums Tables on MySQL
- Contributed to INSERTING values into Location and Location_PhoneNums tables on MySQL
- Contributed by doing QUERIES for requirements 1 and 2.
- Contributed by doing php first 2 requirements

Mustaquim

- Contributed to overall Scope.
- Contributed to adding requirements 6 and 10.
- Contributed to creating Employee entity and attributes to the ER
- Contributed to creating Agent, Agent_PhoneNums, and Employee in the Relational Model
- Contributed to creating the Normalization for tables Agent, Agent_PhoneNums, and Employee and adding their description.
- Contributed to creating Agent, Agent_PhoneNums, and Employee Tables on MySQL
- Contributed to INSERTING values into Agent, Agent_PhoneNums, and Employee tables on MySQL
- Contributed by doing QUERIES for requirements 3 and 4.
- Contributed by doing php for requirements 3 and 4

Angel

- Contributed to overall Scope.
- Contributed to adding requirements 7 and 3.
- Contributed to creating Activity entity and attributes to the ER
- Contributed to creating Owner and Owns tables in the Relational Model
- Contributed to creating the Normalization for tables Owner and Owns, and adding their description.
- Contributed to creating Owner and Owns Tables on MySQL
- Contributed to INSERTING values into Owner and Owns tables on MySQL
- Contributed by doing QUERIES for requirements 5 and 6.
- Contributed by doing the teams php and individual php
- Contributed by doing php for requirements 5 and 6

Paul

- Contributed to overall Scope.

- Contributed to adding requirements 4 and 1.
- Contributed to creating Owner entity and attributes to the ER
- Contributed to creating Business and License tables in the Relational Model
- Contributed to creating the Normalization for tables Business and License; and adding their description.
- Contributed to creating Business and License Tables on MySQL
- Contributed to INSERTING values into Business and License tables on MySQL
- Contributed by doing QUERIES for requirements 7 and 8.
- Contributed by doing php for requirements 7 and 8

Arif

- Contributed to overall Scope.
- Contributed to adding requirements 2 and 5.
- Contributed to creating Activity entity and attributes to the ER
- Contributed to creating Owner and Owns tables in the Relational Model
- Contributed to creating the Normalization for tables Owner and Owns, and adding their description.
- Contributed to creating Owner and Owns Tables on MySQL
- Contributed to INSERTING values into Owner and Owns tables on MySQL
- Contributed by doing QUERIES for requirements 9 and 10.
- Contributed by doing the individual php
- Contributed by doing php for requirements 9 and 10

IN THE TEAM JOURNAL

Each team member should reply to the entry saying: I agree with the contribution of my teammates stated in this journal entry.

Leila Martinez: I agree with the contribution of my teammates stated in this journal entry.

Paul Djangang : I agree with the contribution of my teammates stated in this journal entry.

S M Mustaquim: I agree with the contribution of my teammates stated in this journal entry.

Angel Otero Montanez: I agree with the contribution of my teammates stated in this journal entry.

Mohammad Ariful Islam Khan: I agree with the contribution of my teammates stated in this journal entry.