# Task 9. SQL Scripts for Table Generation and Initial Data

## Step 1: Write the SQL Script for Table Creation

Create a file named create_tables.sql that contains the SQL commands to create the required tables and relationships.

**create_tables.sql:**

```sql
-- Create User table
CREATE TABLE users (
    id CHAR(36) PRIMARY KEY,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    is_admin BOOLEAN DEFAULT FALSE
);

-- Create Place table
CREATE TABLE places (
    id CHAR(36) PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    price DECIMAL(10, 2) NOT NULL,
    latitude FLOAT NOT NULL,
    longitude FLOAT NOT NULL,
    owner_id CHAR(36),
    FOREIGN KEY (owner_id) REFERENCES users(id) ON DELETE CASCADE
);

-- Create Review table
CREATE TABLE reviews (
    id CHAR(36) PRIMARY KEY,
    text TEXT NOT NULL,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    user_id CHAR(36),
    place_id CHAR(36),
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,
    FOREIGN KEY (place_id) REFERENCES places(id) ON DELETE CASCADE,
    UNIQUE (user_id, place_id)
);

-- Create Amenity table
CREATE TABLE amenities (
    id CHAR(36) PRIMARY KEY,
    name VARCHAR(255) UNIQUE NOT NULL
);

-- Create Place_Amenity association table
CREATE TABLE place_amenity (
    place_id CHAR(36),
    amenity_id CHAR(36),
    PRIMARY KEY (place_id, amenity_id),
    FOREIGN KEY (place_id) REFERENCES places(id) ON DELETE CASCADE,
    FOREIGN KEY (amenity_id) REFERENCES amenities(id) ON DELETE CASCADE
);
```

---

## Step 2: Write the SQL Script for Initial Data Insertion

Create another file named insert_initial_data.sql to insert initial data into the tables.

**insert_initial_data.sql:**

```sql
-- Insert administrator user
INSERT INTO users (id, first_name, last_name, email, password, is_admin)
VALUES (
    '36c9050e-ddd3-4c3b-9731-9f487208bbc1',
    'Admin',
    'HBnB',
    'admin@hbnb.io',
    '$2b$12$WZqhkmHErIMTl7YKJ/RdfXEbrNKg9XyFo7Csh5RgN5tQ8qGHyjfD2', -- bcrypt hash of 'admin1234'
```

```
   TRUE
);

-- Insert initial amenities
INSERT INTO amenities (id, name) VALUES (UUID(), 'WiFi');
INSERT INTO amenities (id, name) VALUES (UUID(), 'Swimming Pool');
INSERT INTO amenities (id, name) VALUES (UUID(), 'Air Conditioning');
```

- The bcrypt hash for the password admin1234 can be generated using Python:

```python
from bcrypt import hashpw, gensalt
print(hashpw(b"admin1234", gensalt()).decode())
```

---

## Step 3: Test the SQL Scripts

**Load and execute the scripts using SQLite or another database:**

1. **Create the database and execute the creation script:**

```bash
sqlite3 development.db < create_tables.sql
```

2. **Insert the initial data:**

```bash
sqlite3 development.db < insert_initial_data.sql
```

3. **Verify the database schema and data:**
    - Open the database using SQLite:

```bash
sqlite3 development.db
```

- List tables:

```sql
.tables
```

- Check the data:

```sql
SELECT * FROM users;
SELECT * FROM amenities;
```

---

## Step 4: Test CRUD Operations

Manually test CRUD operations (Create, Read, Update, Delete) using SQL queries to verify the database functionality.

**Examples:**

1. **Create a new user:**

```sql
INSERT INTO users (id, first_name, last_name, email, password, is_admin)
VALUES ('new-user-uuid', 'John', 'Doe', 'john.doe@example.com', '$2b$12$examplehash', FALSE);
```

2. **Retrieve all users:**

```sql
SELECT * FROM users;
```

3. **Update a user:**

```sql
UPDATE users SET email = 'john.new@example.com' WHERE id = 'new-user-uuid';
```

4. **Delete a user:**

```sql
DELETE FROM users WHERE id = 'new-user-uuid';
```

---

## Step 5: Automate Execution (Optional)

Write a shell script to automate the execution of SQL scripts:

```bash
#!/bin/bash

DB_FILE="development.db"

# Remove the old database
```

```
if [ -f $DB_FILE ]; then
    rm $DB_FILE
fi


# Create the database schema
sqlite3 $DB_FILE < create_tables.sql


# Insert initial data
sqlite3 $DB_FILE < insert_initial_data.sql
```

echo "Database has been recreated and populated with initial data."

Save this as reset_db.sh and make it executable:

```bash
chmod +x reset_db.sh
./reset_db.sh
```

---

**Expected Outcome**

1. The following tables are created:
   - users
   - places
   - reviews
   - amenities
   - place_amenity
2. The database contains:
   - An admin user with the provided credentials.
   - Three initial amenities (WiFi, Swimming Pool, Air Conditioning).
3. All relationships (e.g., foreign keys and unique constraints) are functional and enforce the database integrity.