Pharmacy Database Report

1. Introduction

The "Pharmacy Database" is designed to manage the essential operations of a pharmacy. It includes information about companies, medicines, patients, doctors, staff, and transactions such as prescriptions and purchases. This document outlines the database structure, table relationships, and sample data entries.

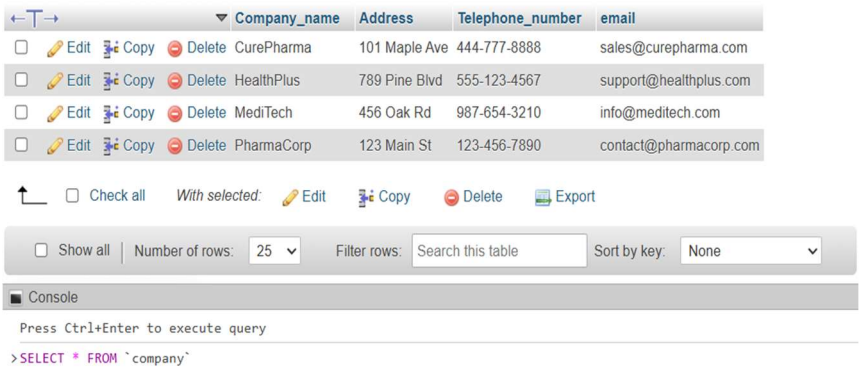
---

2. Database Schema

Database Name: pharmacy_data_base

Tables Overview

1. **company:**
    - Stores company details.
    - **Primary Key:** Company_name



	Company_name	Address	Telephone_number	email
Edit Copy Delete	CurePharma	101 Maple Ave	444-777-8888	sales@curepharma.com
Edit Copy Delete	HealthPlus	789 Pine Blvd	555-123-4567	support@healthplus.com
Edit Copy Delete	MediTech	456 Oak Rd	987-654-3210	info@meditech.com
Edit Copy Delete	PharmaCorp	123 Main St	123-456-7890	contact@pharmacorp.com

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

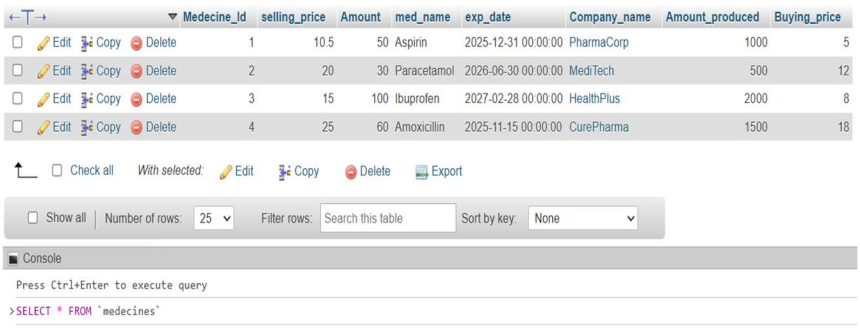
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `company`
```

```
create table company( Company_name varchar(100),
Address  varchar(100),
Telephone_number varchar(100),
email varchar(100),
constraint primcompany primary key(Company_name));
```

2. **medecines:**
    - Contains information about medicines.
    - **Primary Key:** Medecine_Id
    - **Foreign Key:** Company_name references company(Company_name)



	Medecine_Id	selling_price	Amount	med_name	exp_date	Company_name	Amount_produced	Buying_price
Edit Copy Delete	1	10.5	50	Aspirin	2025-12-31 00:00:00	PharmaCorp	1000	5
Edit Copy Delete	2	20	30	Paracetamol	2026-06-30 00:00:00	MediTech	500	12
Edit Copy Delete	3	15	100	Ibuprofen	2027-02-28 00:00:00	HealthPlus	2000	8
Edit Copy Delete	4	25	60	Amoxicillin	2025-11-15 00:00:00	CurePharma	1500	18

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

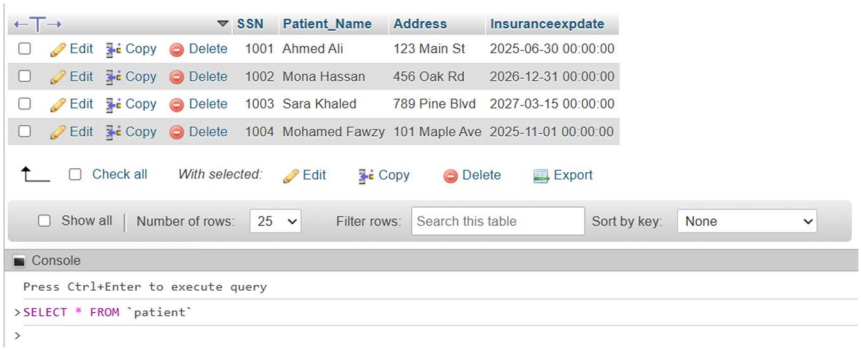
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `medecines`
```

```
create table Medecines( Medecine_Id int,
selling_price float,
Amount int,
med_name varchar(100),
exp_date datetime,
Company_name varchar(100),
Amount_produced float,
Buying_price float,
constraint primmedecines primary key(Medecine_Id),
constraint foreignk foreign key(Company_name) references company(Company_name)
);
```

3. **patient:**
    - Holds patient records.
    - **Primary Key:** SSN



	SSN	Patient_Name	Address	Insuranceexpdate
Edit Copy Delete	1001	Ahmed Ali	123 Main St	2025-06-30 00:00:00
Edit Copy Delete	1002	Mona Hassan	456 Oak Rd	2026-12-31 00:00:00
Edit Copy Delete	1003	Sara Khaled	789 Pine Blvd	2027-03-15 00:00:00
Edit Copy Delete	1004	Mohamed Fawzy	101 Maple Ave	2025-11-01 00:00:00

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

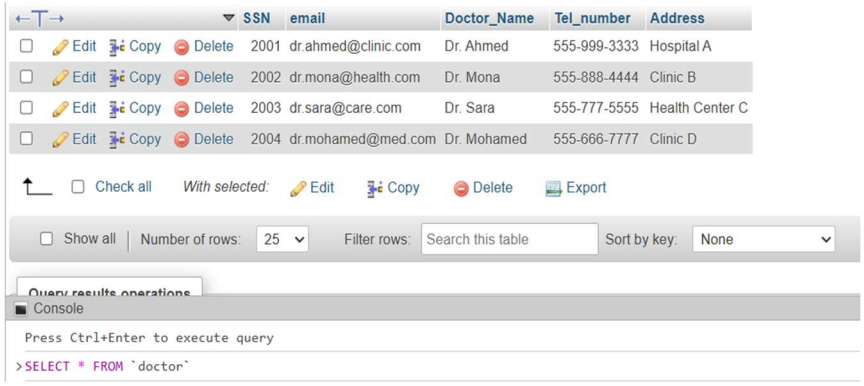
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `patient`
>
```

```
create table Patient ( SSN int,
Patient_Name varchar(100),
Address varchar(100),
Insuranceexpdate datetime,
constraint primpat primary key(SSN)
);
```

4. **doctor:**
    - Stores doctor details.
    - **Primary Key:** SSN



	SSN	email	Doctor_Name	Tel_number	Address
Edit Copy Delete	2001	dr.ahmed@clinic.com	Dr. Ahmed	555-999-3333	Hospital A
Edit Copy Delete	2002	dr.mona@health.com	Dr. Mona	555-888-4444	Clinic B
Edit Copy Delete	2003	dr.sara@care.com	Dr. Sara	555-777-5555	Health Center C
Edit Copy Delete	2004	dr.mohamed@med.com	Dr. Mohamed	555-666-7777	Clinic D

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Query results operations

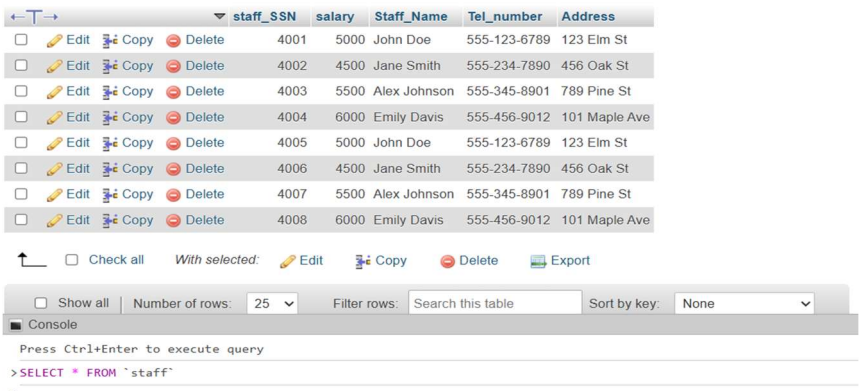
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `doctor`
```

```
create table Doctor ( SSN int,
email varchar(100),
Doctor_Name varchar(100),
Tel_number varchar(100),
Address varchar(100),
constraint primpat primary key(SSN)
);
```

5. **staff:**
    - Maintains staff information.
    - **Primary Key:** staff_SSN



	staff_SSN	salary	Staff_Name	Tel_number	Address
Edit Copy Delete	4001	5000	John Doe	555-123-6789	123 Elm St
Edit Copy Delete	4002	4500	Jane Smith	555-234-7890	456 Oak St
Edit Copy Delete	4003	5500	Alex Johnson	555-345-8901	789 Pine St
Edit Copy Delete	4004	6000	Emily Davis	555-456-9012	101 Maple Ave
Edit Copy Delete	4005	5000	John Doe	555-123-6789	123 Elm St
Edit Copy Delete	4006	4500	Jane Smith	555-234-7890	456 Oak St
Edit Copy Delete	4007	5500	Alex Johnson	555-345-8901	789 Pine St
Edit Copy Delete	4008	6000	Emily Davis	555-456-9012	101 Maple Ave

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

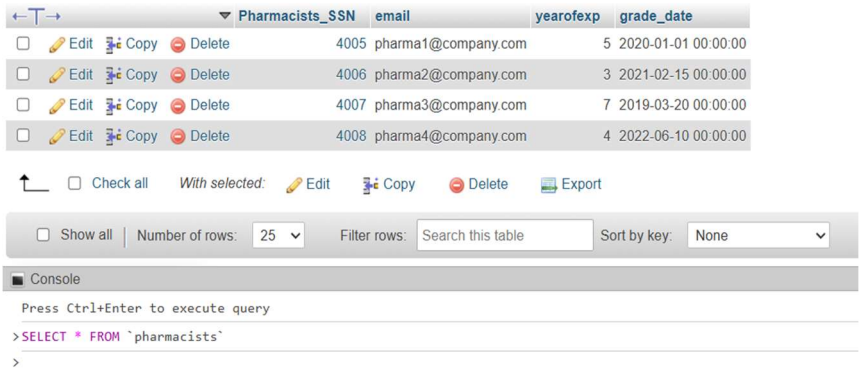
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `staff`
>
```

```
CREATE TABLE `staff` (
  `staff_SSN` int NOT NULL,
  `salary` float DEFAULT NULL,
  `Staff_Name` varchar(100) DEFAULT NULL,
  `Tel_number` varchar(100) DEFAULT NULL,
  `Address` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`staff_SSN`)
);
```

6. **pharmacists:**
    - Stores pharmacist records.
    - **Primary Key:** Pharmacists_SSN
    - **Foreign Key:** Pharmacists_SSN references staff(staff_SSN)



	Pharmacists_SSN	email	yearofexp	grade_date
Edit Copy Delete	4005	pharma1@company.com	5	2020-01-01 00:00:00
Edit Copy Delete	4006	pharma2@company.com	3	2021-02-15 00:00:00
Edit Copy Delete	4007	pharma3@company.com	7	2019-03-20 00:00:00
Edit Copy Delete	4008	pharma4@company.com	4	2022-06-10 00:00:00

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

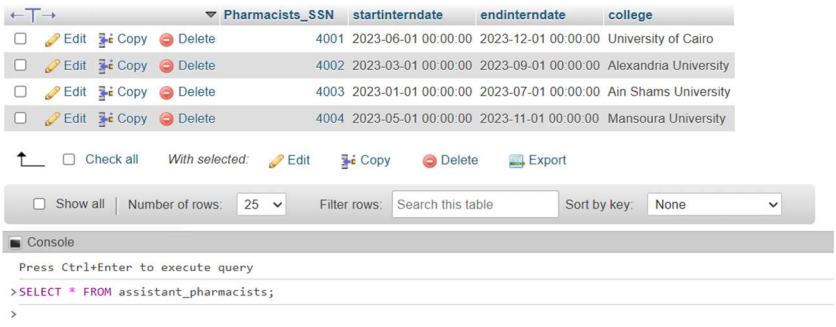
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `pharmacists`
>
```

```
create table Pharmacists ( Pharmacists_SSN int,
email varchar(100),
yearofexp int,
grade_date datetime,
constraint foreign key (Pharmacists_SSN) references staff (staff_SSN),
constraint primpat primary key(Pharmacists_SSN)
);
```

7. **assistant_pharmacists:**
    - Tracks assistant pharmacists.
    - **Primary Key:** Pharmacists_SSN
    - **Foreign Key:** Pharmacists_SSN references staff(staff_SSN)



	Pharmacists_SSN	startinterndate	endinterndate	college
Edit Copy Delete	4001	2023-06-01 00:00:00	2023-12-01 00:00:00	University of Cairo
Edit Copy Delete	4002	2023-03-01 00:00:00	2023-09-01 00:00:00	Alexandria University
Edit Copy Delete	4003	2023-01-01 00:00:00	2023-07-01 00:00:00	Ain Shams University
Edit Copy Delete	4004	2023-05-01 00:00:00	2023-11-01 00:00:00	Mansoura University

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

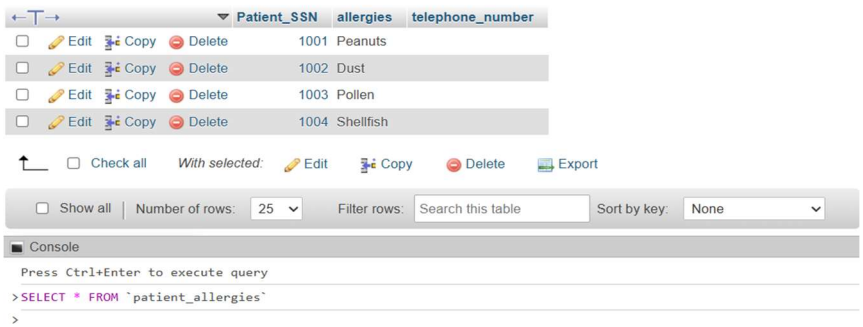
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM assistant_pharmacists;
>
```

```
CREATE TABLE Assistant_Pharmacists (
    Pharmacists_SSN INT,
    startinterndate DATETIME,
    endinterndate DATETIME,
    college VARCHAR(100),
    CONSTRAINT forphar FOREIGN KEY (Pharmacists_SSN) REFERENCES staff (staff_SSN),
    CONSTRAINT primpat PRIMARY KEY (Pharmacists_SSN)
);
```

8. **patient_allergies:**
    - Lists patient allergies.
    - **Primary Key:** (Patient_SSN, telephone_number)
    - **Foreign Key:** Patient_SSN references patient(SSN)



	Patient_SSN	allergies	telephone_number
Edit Copy Delete	1001	Peanuts	
Edit Copy Delete	1002	Dust	
Edit Copy Delete	1003	Pollen	
Edit Copy Delete	1004	Shellfish	

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

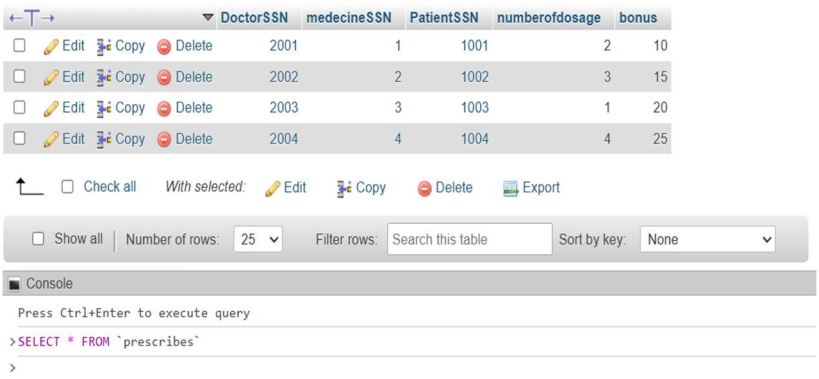
Console

Press Ctrl+Enter to execute query

```
>SELECT * FROM `patient_allergies`
>
```

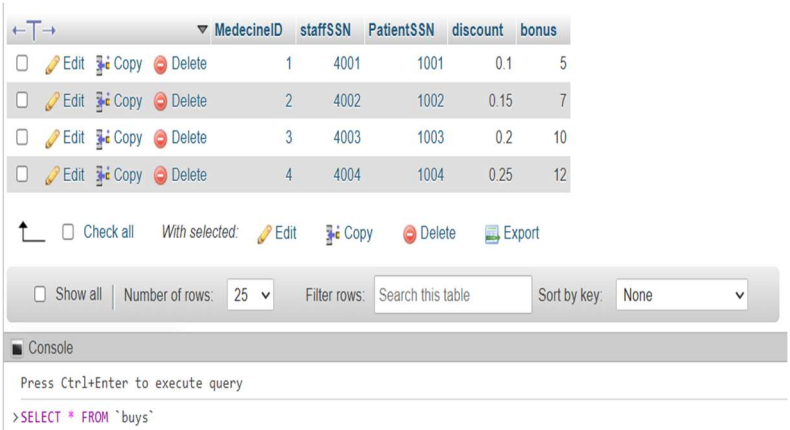
```
create table Patient_allergies ( Patient_SSN int,
allergies varchar(100),
telephone_number VARCHAR(15),
constraint forpattel foreign key (Patient_SSN) references patient (SSN),
constraint primpat primary key(Patient_SSN,telephone_number)
);
```

9. **prescribes:**
    - Tracks medicine prescriptions.
    - **Primary Key:** (DoctorSSN, medecineSSN, PatientSSN)
    - **Foreign Keys:**
        - DoctorSSN references doctor(SSN)
        - medecineSSN references medecines(Medecine_Id)
        - PatientSSN references patient(SSN)



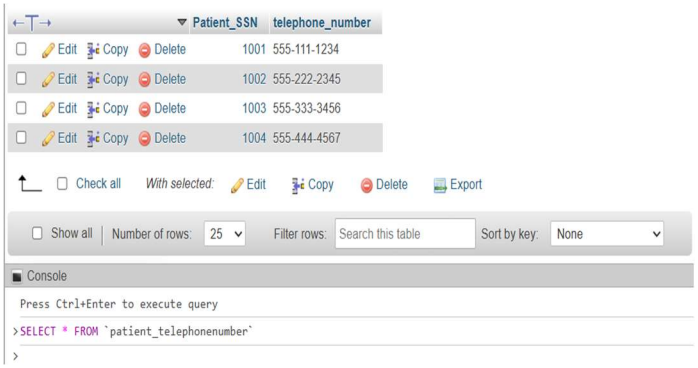

```
create table prescribes
(DoctorSSN int,
 medecineSSN int,
PatientSSN int,
numberofdosage int,
bonus int,
constraint forkeyper foreign key (DoctorSSN) references doctor(SSN),
constraint forkey2per foreign key (medecineSSN) references medecines (Medecine_Id),
constraint forkey3per foreign key (PatientSSN) references patient (SSN),
constraint primkyper primary key(DoctorSSN,medecineSSN,PatientSSN));
```

10. **buys:**
    - Records medicine purchases.
    - **Primary Key:** (staffSSN, MedecineID, PatientSSN)
    - **Foreign Keys:**
        - staffSSN references staff(staff_SSN)
        - MedecineID references medecines(Medecine_Id)
        - PatientSSN references patient(SSN)




```
create table buys
(MedecineID int,
 staffSSN int,
PatientSSN int,
discount float,
bonus int,
constraint forkeybuy foreign key (staffSSN) references staff(staff_SSN),
constraint forkey2buy foreign key (MedecineID) references medecines (Medecine_Id),
constraint forkey3buy foreign key (PatientSSN) references patient (SSN),
constraint primkybuy primary key(staffSSN,MedecineID,PatientSSN));
```

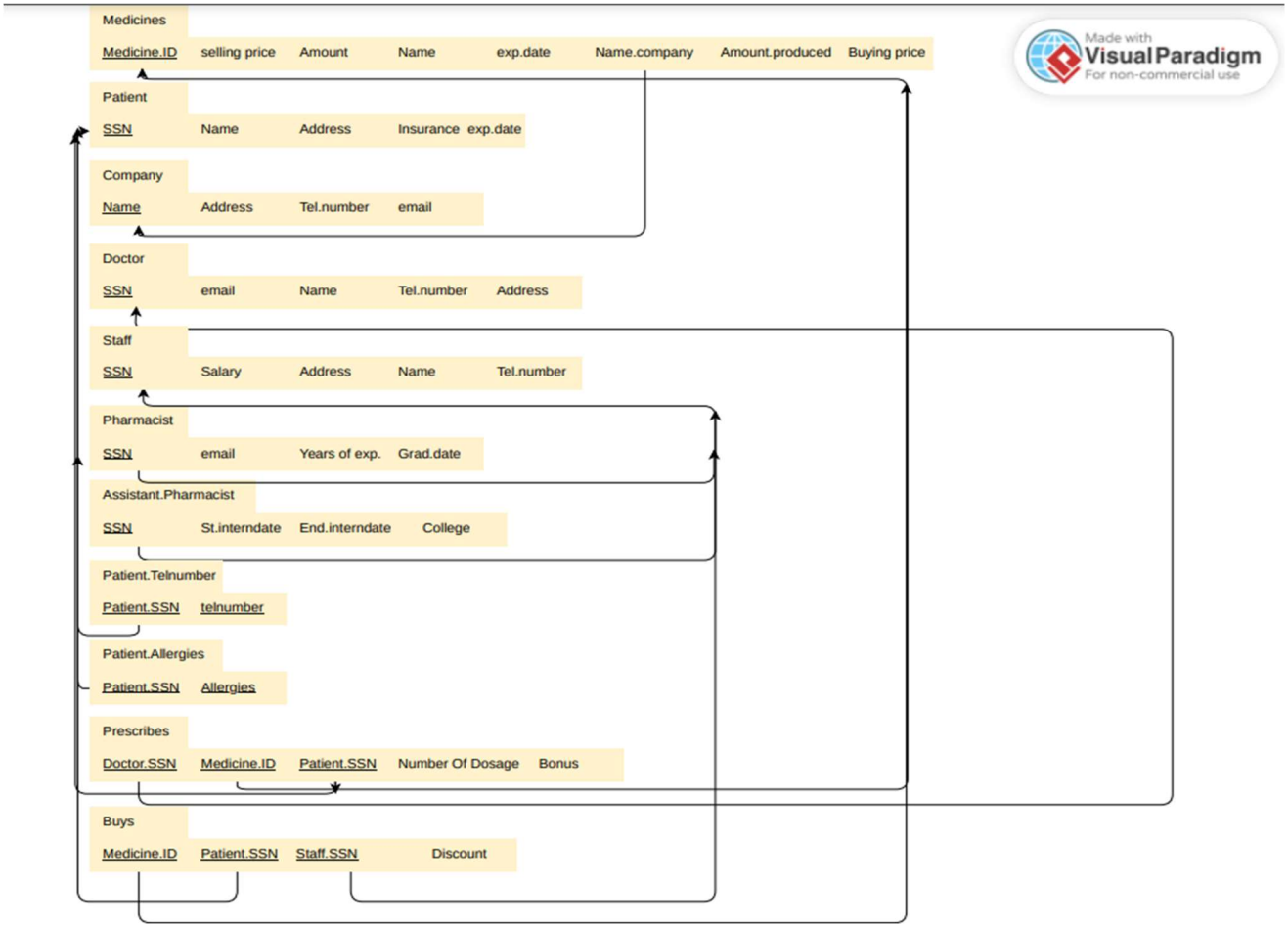
11. **patient_telephonenumber:**
    - Tracks patients' phone numbers.
    - **Primary Key:** (Patient_SSN, telephone_number)
    - **Foreign Key:** Patient_SSN references patient(SSN)



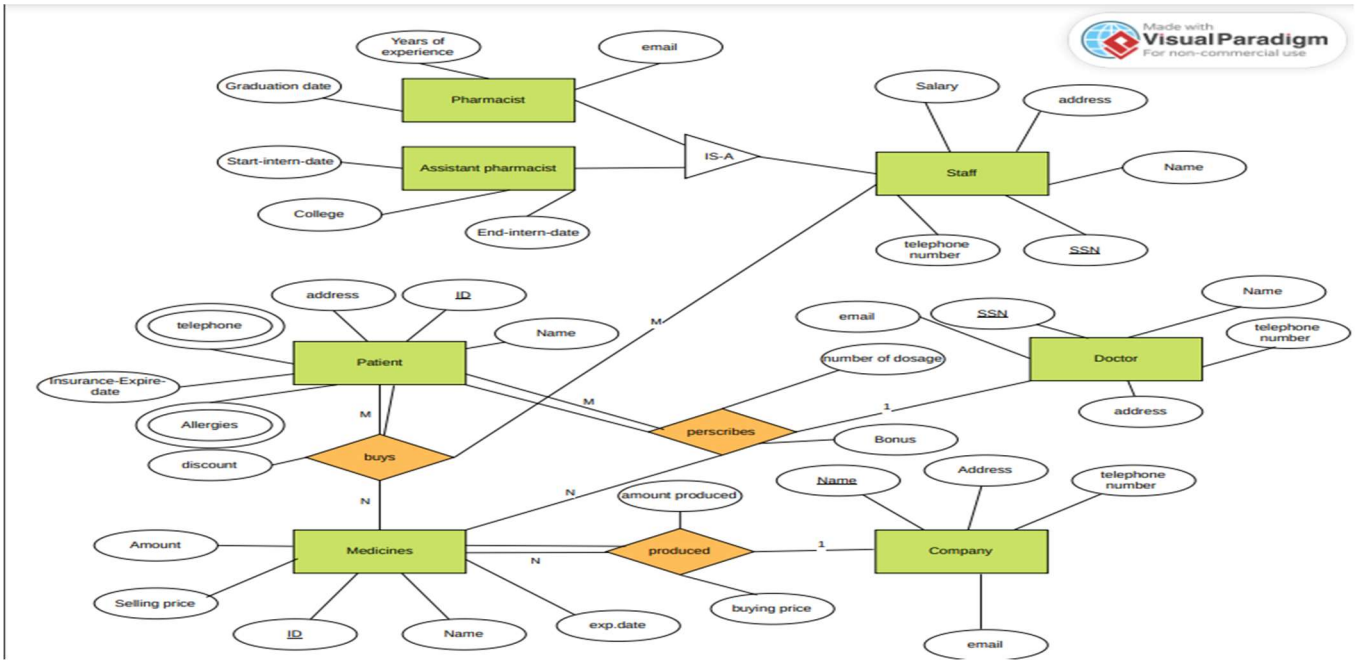

```
CREATE TABLE `patient_telephonenumber` (
  `Patient_SSN` int NOT NULL,
  `telephone_number` varchar(100) NOT NULL,
  PRIMARY KEY (`Patient_SSN`,`telephone_number`),
  CONSTRAINT `fk_patient_telephonenumber` FOREIGN KEY (`Patient_SSN`) REFERENCES `patient` (`SSN`)
);
```

---

3. ***Relational Mapping*:**



4. Entity Relationship Diagram (ERD)



---

5. Business Rules:

1. **Prescription Verification:** Only licensed pharmacists can approve prescriptions, making sure they check for potential interactions and correct dosages.

2. **Inventory and Expiry Management:** Automatically reorders items when stock is low and blocks any expired batches from being dispensed.

3. **Age and Dosage Limits:** Controls access to medications based on age and monitors dosage limits according to the patient's profile.

4. **Interaction and Allergy Warnings:** Issues alerts when there's a risk of adverse reactions or allergies.

5. **Insurance and Cost Checks:** Verifies insurance coverage and notifies patients if they need to cover any additional costs.

6. **Recall Alerts:** Blocks recalled medicines to prevent them from being dispensed. With these rules and connections, the DBMS can ensure safe, efficient, and patient-centered pharmacy operations.

---

6. *Conclusion:*

The "Pharmacy Database" provides a comprehensive system for managing pharmacy operations, enhancing efficiency, and maintaining accurate records. Its structured schema and interlinked tables ensure seamless data management and retrieval.