-DBMS Winter '21

Vaccination Database

Team *Teekakaran*

Team Members and Contribution

1. Aman - 2019013 - defining entities and relations , writing relational algebra queries, SQL queries
2. Hardik Garg - 2019040 - Defining mini world and functional requirements, ER diagram, relational schema diagram, coding database in mysql, SQL queries
3. Rohan Arora - 2019323 - defining entities and relations, ER diagram, relational schema diagram, writing SQL queries
4. Abhishek Chaturvedi - 2019401 - defining entities and relations, ER diagram, relational schema diagram, writing SQL queries
5. Harsh Kumar Agarwal - 2019423 - defining entities and relations, writing relational algebra queries, SQL queries

Introduction (Mini World)

After nearly a year of hard work and efforts, several pharmaceutical companies have come up with a vaccine in order to combat a country wide pandemic which has infected millions of people. The vaccine is good news, however, now the major challenge in front of the Health Ministry is to distribute these vaccines and streamline the entire process. This involves many tasks -

- Approving vaccines by pharmaceutical companies by studying their data and getting approval from professional medical authorities
- Procuring the vaccines from these companies on a contractual or requirement basis
- Distributing license to health care centers to allow them to vaccinate people

- Shipment of vaccines, either through their own means or letting the pharmaceutical company handle it
- Prioritizing the distribution of vaccine - frontline healthcare workers and senior citizens should be vaccinated first
- Allowing all citizens to register themselves for the vaccination program and allotting them a date and health care center for the same

We aim to build a **Vaccination Database** in order to streamline the entire process by storing all the relevant details related to the vaccine, distributors, health care centers, medical professionals, vaccine stocks and types, citizen details and many more aspects.

The database will be available as a public portal for the citizens to get themselves registered and other stakeholders (medical professionals, Health Ministry, vaccine distributors etc) will have some additional privileges based on their requirement with partial/complete viewing/editing access, as needed.

Functional Requirements

The database shall fulfill the following functional requirements -

- The Health Ministry can use it to approve vaccines by consulting medical professionals, procure them from the vaccine distributors, handle their distribution to the health care center, allow citizens to register themselves for the vaccination
- The medical professionals can use it to view the details of their patients (like medical history) and recommend them vaccine accordingly. They can approve or disapprove vaccines which can be stored as an attribute in their table. Also, they can view the details of health care centers for which they are responsible
- The health care centers can view the details of vaccines available, their prices and time of arrival in order to meet the requirements. They can view patient details in order to avoid complications during vaccination and can allot a date to a patient for getting vaccinated

- The vaccine distributor can view the details of patients that have already been vaccinated in order to collect more data regarding the efficacy of their vaccines and can forward this data to their research team. They can contact the health centers or the Health Ministry regarding vaccine contracts and shipments as well.
- The citizens can view vaccine details like side effects and precautions to be taken. They can view health center details so that they can choose one according to their convenience and get it approved. They can view medical professional details so that they can choose which one to visit based on their requirement.

WEEK-1

Identifying Stakeholders

Stakeholders

1. Health Ministry - Complete editing and viewing rights
2. Medical Professionals - Partial editing and complete viewing rights
3. Health Care Centers - Partial editing and complete viewing rights
4. Vaccine Distributors - Partial editing and viewing rights
5. Citizens - Partial editing and viewing rights

Roles of Stakeholders

1. **Health Ministry -** They serve as the admin of the database system. They have complete editing and viewing rights over the database. They are solely responsible for managing the database.

2. **Medical Professionals -** They are responsible for prescribing a particular vaccine to the patients based on their requirements and can modify the data records of the patients and monitor them accordingly. They have complete viewing rights to stay in the loop of things however, partial editing rights.

3. **Health Care Centers -** They are responsible for providing the vaccine to the patients as per requirement. They can check the distributor tables and place orders for vaccines as per the demand. They can also check patient records just like medical professionals before giving vaccines to patients. They have complete viewing rights and partial editing rights.

4. **Vaccine Distributors -** They are responsible for getting their vaccines approved and shipping them to health care centers as per the demand. They have partial editing and viewing rights as they cannot see the complete details of every patient. However, they can see a collective percentage indicating the efficacy of their vaccine.

5. **Citizens -** They can view the vaccine options available to them and book vaccination slots at health care centers after consulting with medical professionals (if needed). They have partial viewing and editing access since they will be editing their own details only.

WEEK-2

Stakeholder Queries and Entities

Queries Associated with Stakeholders

1. **Health Ministry**
    a. Create database and table commands
    b. Add/edit/modify in the database tables
    c. Can view all the available vaccine records and their efficacy
    d. Can view all citizens who signed up for vaccination
    e. Add/Remove health care centers (analogous to giving licenses physically)
    f. Check vaccine stocks and prices with vaccine distributors

2. **Medical Professionals**
    a. Display the details of all health care centers
    b. Get details of all types of vaccines sorted by their efficacy
    c. Check if a particular patient has some allergy with respect to a particular vaccine before prescribing
    d. Get the details of patient and update their medical record for future reference (if a vaccine needs multiple shots, update the details)
    e. Get details of all health centers sorted by the type of vaccine most used in each (to get a demographic idea)
    f. Check if a patient has been vaccinated before or not
    g. Compare the results of the same vaccine across different health centers to gather statistical data and advise the Health Ministry

3. **Health Care Centers**
    a. Display the details of patients, prior allergies or medical conditions regarding a particular vaccine

b. View the stock of particular vaccine and order for more to meet the demand
c. Keep a record of efficacy of the vaccine to forward to other stakeholders
d. Allot a slot to a patient for vaccination
e. Display list of patients sorted by age to prioritize senior citizens first for vaccination
f. Sort vaccine suppliers by proximity of location and price. This will help them procure vaccines in a timely and cost effective manner

4. Vaccine Distributors

a. Display the details of all health care centers using the vaccine sorted by number of orders
b. Display details of efficacy of vaccine as reported by health care centers and figure out manufacturing logistics (more effective vaccine is manufactured more)
c. Keep a track of which vaccines have received legal licenses by government and have been approved
d. Maintain record of the production costs of vaccine and a profit/loss field
e. Get sales records of rival vaccine distributors from the health care center records
f. Keep a track of side effects of their vaccine (severity of side effects is tracked using numeric values)

5. Citizens

a. Add and sign up themselves on the portal for vaccination
b. View all the available vaccine choices to them (efficacy will not be visible)
c. View the details of medical professionals to consult (a citizen with specific medical conditions will seek consultation from the medical professional of that field)
d. See active/recovered cases at each health center (so that they can take proper measures while visiting)
e. Book a slot and center for vaccination (final verdict shall rest with the center)

f. Display health center details
g. Modify their medical details after getting vaccinated once (for types which require multiple shots)

Entities and their Attributes

1. **Health Ministry-** Record of employees, health care centres, vaccines, vaccine distributors, citizens.
2. **Manufacturing Company-**
    a. Name
    b. Company ID
    c. Location
    d. Type of Vaccine
    e. Capacity
    f. Employees
    g. Quantity Manufactured
    h. Orders Remaining
3. **Vaccine-**
    a. Name
    b. Vaccine ID
    c. Type
    d. Manufacturer
    e. Manufacturing Date
    f. Expiry Date
    g. Storage Requirement
    h. Price
4. **Medical Professionals-**
    a. ID
    b. Name
    c. Date of Birth
    d. Type
    e. Health Care Centre associated with

5. **Health Care Centres-**
    a. Centre ID
    b. Name
    c. Location
    d. Record of Medical Professionals
    e. Record of Patients admitted
    f. Capacity
    g. Type of vaccines stored
    h. Number of vaccines available
6. **Vaccine Distributors (and manufacturers)**-
    a. Distributor ID
    b. Location
    c. Capacity
    d. Type of vaccines distributed
    e. Quantity Delivered
    f. Quantity to Deliver
    g. Delivery Cost
    h. Health Care centres distributed to
7. **Citizens-**
    a. Citizen ID
    b. Name
    c. Gender
    d. Date of Birth
    e. Address
    f. Medical History

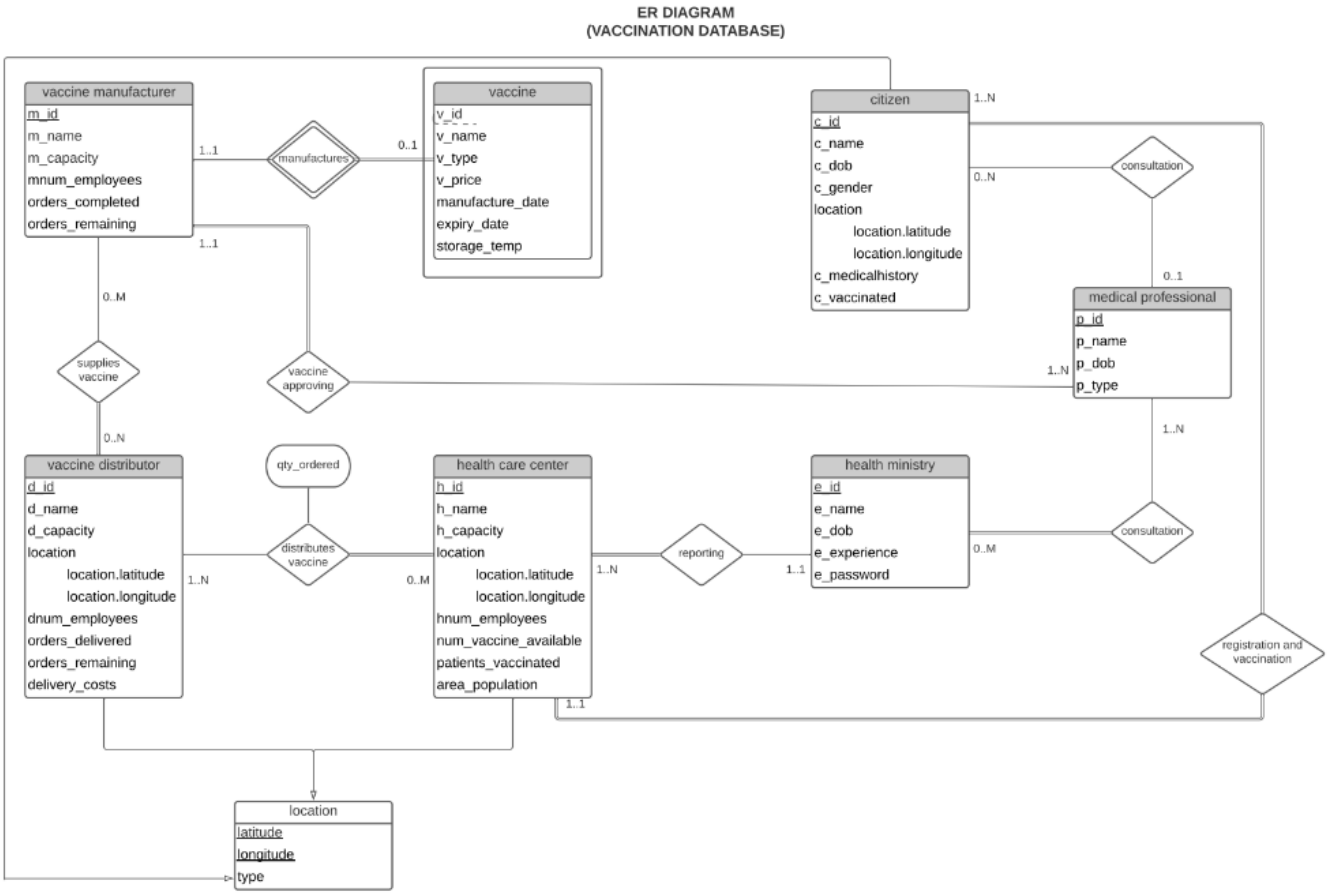
Relationships Between Entities

1. Vaccine Distributor **distributes** Vaccine
2. Vaccine Manufacturer **manufactures** Vaccine
3. Citizens **register for** Vaccine
4. Health Ministry **approves** Vaccine
5. Health Care Center **allots slots to** Citizens
6. Citizens **consult** Medical Professionals
7. Medical Professionals **inspect** Vaccine
8. Health Care Center **orders** Vaccine
9. Health Ministry **procures** vaccines from manufacturing company
10. Distributors **deliver** vaccines to health care centres
11. Health care centre **vaccinate** citizens
12. Health Ministry **guides** the distributors for efficient delivery of vaccines
13. Health care centres **update** the status to health ministry regularly

WEEK-3 and Week-4

ER diagrams and Relationship schemas

ER Diagram

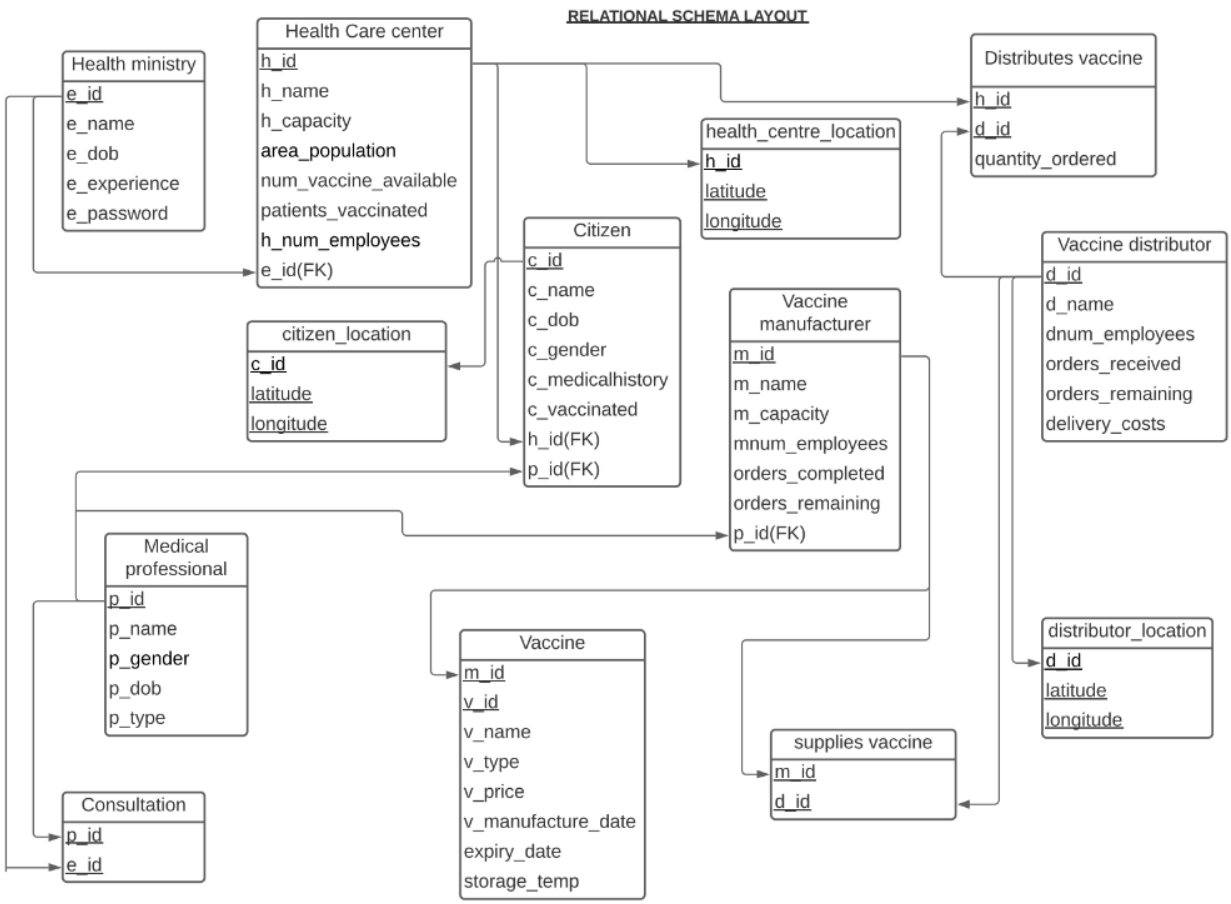


Reasoning Behind Some Implementations -

1. **Weak Entity -** Vaccine is chosen as a weak entity which depends on the vaccine manufacturer, because no manufacturer means no vaccine and for convenience we have assumed that 1 manufacturer can make only 1 vaccine (practically reasonable as well since it takes a lot of time, effort and testing to make even 1 vaccine)

2. **Making Location a Separate Entity -** Based on feedback given by Sir in class demos, we made location a separate entity and had location as a composite attribute (latitude and longitude) in vaccine_distributor, health_care_centre and citizen entities.
3. **Making separate entities for vaccine manufacturer -** Here, vaccine manufacturer acts as the vaccine supplier for the vaccine distributor which orders from the available manufacturers. We did not keep a location entity for it because it is assumed that the distributors already have ties with the closest and best manufacturers and also to simplify the database (there are already 3 location tables). Also, the vaccine manufacturer is not a separate stakeholder because it is considered to be a subset of the largest vaccine distributors. For example - there are 50 distributors out of which 10 have their manufacturing plants and all 50 distributors can order from these 10 plants.
4. **Regarding Quantity of Vaccines -** It is assumed that whenever vaccines are delivered/supplies/transported, they are in units of lakhs/thousands and therefore, the price indicated is w.r.t the whole unit of vaccines and not 1 atomic piece of vaccine.
5. **Modifications in Entities and Relations -** In Week-2 we wrote down all entities and relations that we could think of, however, when we prepared the ER diagram, had feedback from Sir and TA during interaction sessions, we modified them from time to time. The two main reasons for this were - to avoid redundant data (age and dob mean the same thing, so we kept only dob) and to maintain referential integrity constraints (defining primary keys, foreign keys and composite keys)

Relational Schema Diagram



The ER diagram is converted to Relational Schema diagram according to the rules discussed in the lectures, introducing foreign keys and making tables for relations, wherever applicable.

All diagrams are created using lucid chart

WEEK-4 and Week-5

Coding Database and Populating Tables

Layout and Code for all the relations (done using DataGrip) -

1. Health Ministry

```
   CREATE TABLE health_ministry (
e_id INT AUTO_INCREMENT,
e_name VARCHAR(30) NOT NULL,
e_dob DATE NOT NULL,
e_experience INT,
e_password VARCHAR(30),
PRIMARY KEY (e_id)
);
```

Field	Type	Null	Key	Default	Extra
e_id	int	NO	PRI	<null>	auto_increment
e_name	varchar(30)	NO		<null>	
e_dob	date	NO		<null>	
e_experience	int	YES		<null>	
e_password	varchar(30)	YES		<null>	

2. Medical Professional

```
   CREATE TABLE medical_professional (
      p_id INT AUTO_INCREMENT,
      p_name VARCHAR(30) NOT NULL,
      p_gender varchar(1) NOT NULL,
      p_dob DATE NOT NULL,
      p_type varchar(30) NOT NULL,
```

```
    PRIMARY KEY (p_id)
);
```

Field	Type	Null	Key	Default	Extra
p_id	int	NO	PRI	<null>	auto_increment
p_name	varchar(30)	NO		<null>	
p_gender	varchar(1)	NO		<null>	
p_dob	date	NO		<null>	
p_type	varchar(30)	NO		<null>	

3. **Ministry Consultation**

```
CREATE TABLE ministry_consultation (
    p_id INT,
    e_id INT,
    FOREIGN KEY (p_id) REFERENCES  medical_professional
(p_id) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (e_id) REFERENCES health_ministry (e_id) ON
DELETE CASCADE ON UPDATE CASCADE,
    PRIMARY KEY (p_id, e_id)
);
```

Field	Type	Null	Key	Default	Extra
p_id	int	NO	PRI	<null>	
e_id	int	NO	PRI	<null>	

4. **Health Care Centre**

```
CREATE TABLE health_care_centre (
    h_id INT AUTO_INCREMENT,
    h_name varchar(30) NOT NULL,
    h_capacity INT NOT NULL DEFAULT 100 CHECK (h_capacity
>=50),
    area_population INT NOT NULL DEFAULT 200,
```

```
    patients_vaccinated INT DEFAULT 0,

    e_id INT,

    num_vaccine_available INT NOT NULL DEFAULT 100,

    h_num_employees INT DEFAULT 100 CHECK (h_num_employees
>= 50),

    PRIMARY KEY (h_id),

    FOREIGN KEY (e_id) REFERENCES health_ministry (e_id) ON
DELETE CASCADE ON UPDATE CASCADE

);
```

Field	Type	Null	Key	Default	Extra
h_id	int	NO	PRI	<null>	auto_increment
h_name	varchar(30)	NO		<null>	
h_capacity	int	NO		100	
area_population	int	NO		200	
patients_vaccinated	int	YES		0	
e_id	int	YES	MUL	<null>	
num_vaccine_available	int	NO		100	
h_num_employees	int	YES		100	

5. Citizen

```
CREATE TABLE citizen (

    c_id INT AUTO_INCREMENT,

    c_name VARCHAR(30) NOT NULL,

    c_dob DATE NOT NULL,

    c_gender varchar(1) NOT NULL,

    c_medicalhistory VARCHAR(200),

    c_vaccinated INT NOT NULL,

    h_id INT,

    p_id INT,

    FOREIGN KEY (h_id) REFERENCES health_care_centre (h_id)
ON DELETE CASCADE ON UPDATE CASCADE,
```

```
   FOREIGN KEY (p_id) REFERENCES health_ministry (e_id) ON
DELETE CASCADE ON UPDATE CASCADE,

   PRIMARY KEY (c_id)

);
```

Field	Type	Null	Key	Default	Extra
c_id	int	NO	PRI	<null>	auto_increment
c_name	varchar(30)	NO		<null>	
c_dob	date	NO		<null>	
c_gender	varchar(1)	NO		<null>	
c_medicalhistory	varchar(200)	YES		<null>	
c_vaccinated	int	NO		<null>	
h_id	int	YES	MUL	<null>	
p_id	int	YES	MUL	<null>	

6. Vaccine Manufacturer

```
CREATE TABLE vaccine_manufacturer (

   m_id INT AUTO_INCREMENT,

   m_name VARCHAR(30) NOT NULL,

   m_capacity INT DEFAULT 2000,

   m_num_employees INT DEFAULT 200 CHECK
(m_num_employees>=100),

   orders_completed INT NOT NULL,

   orders_remaining INT NOT NULL,

   p_id INT,

   FOREIGN KEY (p_id) REFERENCES medical_professional
(p_id) ON DELETE CASCADE ON UPDATE CASCADE,

   PRIMARY KEY (m_id)

);
```

Field	Type	Null	Key	Default	Extra
m_id	int	NO	PRI	<null>	auto_increment
m_name	varchar(30)	NO		<null>	
m_capacity	int	YES		2000	
m_num_employees	int	YES		200	
orders_completed	int	NO		<null>	
orders_remaining	int	NO		<null>	
p_id	int	YES	MUL	<null>	

7. **Vaccine Distributor**

```
CREATE TABLE vaccine_distributor (
   d_id INT AUTO_INCREMENT,
   d_name VARCHAR(30) NOT NULL,
   d_num_employees INT DEFAULT 100 CHECK
(d_num_employees>=50),
   orders_received INT NOT NULL,
   orders_remaining INT NOT NULL,
   delivery_costs FLOAT(12,2) NOT NULL,
   PRIMARY KEY (d_id)
);
```

Field	Type	Null	Key	Default	Extra
d_id	int	NO	PRI	<null>	auto_increment
d_name	varchar(30)	NO		<null>	
d_num_employees	int	YES		100	
orders_received	int	NO		<null>	
orders_remaining	int	NO		<null>	
delivery_costs	float(12,2)	NO		<null>	

8. **Vaccine**

```
CREATE TABLE vaccine (
   m_id INT,
   v_id INT,
   v_name VARCHAR(30) NOT NULL,
   v_type VARCHAR(30) NOT NULL,
```

```
    v_price FLOAT(12,2) NOT NULL,
    v_manufacture_date DATE NOT NULL,
    v_expiry_date DATE NOT NULL,
    storage_temp FLOAT(5,2) NOT NULL,
    FOREIGN KEY (m_id) REFERENCES vaccine_manufacturer
(m_id) ON DELETE CASCADE ON UPDATE CASCADE,
    PRIMARY KEY (m_id, v_id)
);
```

Field	Type	Null	Key	Default	Extra
m_id	int	NO	PRI	<null>	
v_id	int	NO	PRI	<null>	
v_name	varchar(30)	NO		<null>	
v_type	varchar(30)	NO		<null>	
v_price	float(12,2)	NO		<null>	
v_manufacture_date	date	NO		<null>	
v_expiry_date	date	NO		<null>	
storage_temp	float(5,2)	NO		<null>	

9. Vaccine Supply

```
CREATE TABLE vaccine_supply (
    m_id INT,
    d_id INT,
    FOREIGN KEY (m_id) REFERENCES vaccine_manufacturer
(m_id) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (d_id) REFERENCES vaccine_distributor (d_id)
ON DELETE CASCADE ON UPDATE CASCADE,
    PRIMARY KEY (m_id, d_id)
);
```

Field	Type	Null	Key	Default	Extra
m_id	int	NO	PRI	<null>	
d_id	int	NO	PRI	<null>	

10. Vaccine Distribution

```
CREATE TABLE vaccine_distribution (

   h_id INT,

   d_id INT,

   qty_ordered_T INT,

   FOREIGN KEY (h_id) REFERENCES health_care_centre (h_id)
ON DELETE CASCADE ON UPDATE CASCADE,

   FOREIGN KEY (d_id) REFERENCES vaccine_distributor (d_id)
ON DELETE CASCADE ON UPDATE CASCADE,

   PRIMARY KEY (h_id, d_id)

);
```

Field	Type	Null	Key	Default	Extra
h_id	int	NO	PRI	<null>	
d_id	int	NO	PRI	<null>	
qty_ordered_T	int	YES		<null>	

11. Health Centre Location

```
CREATE TABLE health_centre_location (

   h_id INT,

   latitude FLOAT (6,3) NOT NULL,

   longitude FLOAT (6,3) NOT NULL,

   FOREIGN KEY (h_id) REFERENCES health_care_centre (h_id)
ON DELETE CASCADE ON UPDATE CASCADE,

   PRIMARY KEY (h_id, latitude, longitude)

);
```

Field	Type	Null	Key	Default	Extra
h_id	int	NO	PRI	<null>	
latitude	float(6,3)	NO	PRI	<null>	
longitude	float(6,3)	NO	PRI	<null>	

12. Distributor Location

```
CREATE TABLE distributor_location (
    d_id INT,
    latitude FLOAT (6,3) NOT NULL,
    longitude FLOAT (6,3) NOT NULL,
    FOREIGN KEY (d_id) REFERENCES vaccine_distributor (d_id)
ON DELETE CASCADE ON UPDATE CASCADE,
    PRIMARY KEY (d_id, latitude, longitude)
);
```

Field	Type	Null	Key	Default	Extra
d_id	int	NO	PRI	<null>	
latitude	float(6,3)	NO	PRI	<null>	
longitude	float(6,3)	NO	PRI	<null>	

13. Citizen Location

```
CREATE TABLE citizen_location (
    c_id INT,
    latitude FLOAT (6,3) NOT NULL,
    longitude FLOAT (6,3) NOT NULL,
    FOREIGN KEY (c_id) REFERENCES citizen (c_id) ON DELETE
CASCADE ON UPDATE CASCADE,
    PRIMARY KEY (c_id, latitude, longitude)
);
```

Field	Type	Null	Key	Default	Extra
c_id	int	NO	PRI	<null>	
latitude	float(6,3)	NO	PRI	<null>	
longitude	float(6,3)	NO	PRI	<null>	

Populating Tables with Data

All the tables were populated with meaningful random data generated using mockaroo.com ensuring the referential integrity constraints defined. The number of entries per table is as follows -

1. Health Ministry - 100
2. Health Care Centre - 50
3. Ministry Consultation - 100
4. Health Care Centre - 100
5. Citizen - 1000
6. Vaccine Manufacturer - 10
7. Vaccine Distributor - 50
8. Vaccine - 10
9. Vaccine Supply - 50
10. Vaccine Distribution - 100
11. Health Care Centre Location - 50
12. Vaccine Distributor Location - 50
13. Citizen Location - 1000

Week-6

Writing Queries in SQL

SQL queries for each stakeholder were written and executed as follows -

1. Health Ministry

(A) This query returns the details of ministry employees who were in-charge of vaccines which got manufactured the latest.

```
SELECT e_id, e_name, v_name, v_manufacture_date FROM
health_ministry, vaccine WHERE v_manufacture_date =
(SELECT MAX(v_manufacture_date) FROM vaccine) AND e_id
IN (SELECT e_id FROM ministry_consultation WHERE p_id
IN (SELECT p_id FROM vaccine_manufacturer WHERE m_id
IN (SELECT m_id FROM vaccine WHERE v_manufacture_date
= (SELECT MAX(v_manufacture_date) FROM vaccine))));
```

e_id	e_name	v_name	v_manufacture...
3	Emmie	Pyrithions Zinc	2020-12-06
26	Kelcie	Pyrithions Zinc	2020-12-06
72	Katherina	Pyrithions Zinc	2020-12-06
97	Humphrey	Pyrithions Zinc	2020-12-06

(B) This query returns the details of ministry employees who are in charge of health centres with the maximum area population

```
SELECT health_ministry.e_id, e_name, h_id, h_name,
area_population FROM health_ministry,
health_care_centre WHERE area_population IN (SELECT
MAX(area_population) FROM health_care_centre) AND
```

```
health_ministry.e_id = (SELECT health_care_centre.e_id
FROM health_care_centre WHERE area_population=(SELECT
MAX(area_population) FROM health_care_centre));
```

e_id	e_name	h_id	h_name	area_population
64	Tremayne	37	Itzkin health centre	498

(C) This query selects all those employees whose password does not contain a number and hence is insecure, and orders them in non-decreasing order of the length of password

```
SELECT e_id, e_name, e_password FROM health_ministry
WHERE e_password NOT REGEXP '.*\\d+.*' ORDER BY
LENGTH(e_password);
```

e_id	e_name	e_password
11	Jaquelyn	IiifNJ
18	Robbyn	dGViER
4	Fee	aFRXhp
8	Dagny	phrJLnb
19	Terri	przLtsZ
89	Brett	fvVjLNd
9	Panchito	tzenXjih
23	Daisy	TjzYAeqy
36	Cordey	asLqFcYJ
81	Rebel	LcfJIuRW

(D) This query prints all ministry employees with their names and ids along with the number of medical professionals they are consulting and orders them in non-decreasing order of medical professionals

```
SELECT health_ministry.e_id, health_ministry.e_name,
count(p_id) as p_ids FROM health_ministry,
ministry_consultation WHERE health_ministry.e_id =
ministry_consultation.e_id group by e_id having
count(p_id)>=2 ORDER BY count(p_id) DESC;
```

e_id	e_name	p_ids
53	Carlye	6
84	Karalynn	4
73	Sanderson	3
30	Emelda	3
39	Shaun	3
46	Theobald	3
60	Dru	3
52	Deeyn	2
87	Brod	2

(E) This query selects details of all ministry employees such that the distributor they are supervising has the maximum number of pending orders

```
SELECT e_id, e_name, e_experience, orders_remaining
FROM health_ministry, vaccine_distributor WHERE
orders_remaining IN (SELECT MAX(orders_remaining) FROM
vaccine_distributor) AND e_id IN (SELECT e_id FROM
health_care_centre WHERE h_id IN (SELECT h_id FROM
vaccine_distribution WHERE d_id IN (SELECT d_id FROM
vaccine_distributor WHERE orders_remaining IN (SELECT
MAX(orders_remaining) FROM vaccine_distributor))));
```

e_id	e_name	e_experience	orders_remaining
89	Brett	9	497

2. Medical Professional

(A) This query displays details of all medical professionals along with the citizens they have advised

```
SELECT * FROM medical_professional, citizen WHERE
medical_professional.p_id=citizen.p_id;
```

c_id	c_name	c_dob	c_gender	c_medicalhistory	c_vaccinated	h_id
83	Archibald Goodbur	1937-01-22	M	weak immunity	2	1
105	Arlen Fuchs	1923-12-18	M	gall bladder removed	2	83
141	Marion De Minico	1989-12-07	F	gall bladder removed	0	83
142	Ozzie Hazell	1977-06-01	M	weak muscle and nerves	2	77
219	Jermayne Parsonage	1936-11-30	M	gall bladder removed	2	27
244	Clotilda Yelyashev	2002-11-19	F	heart patient	0	82
282	Terra Sackett	1936-02-24	F	breathing issues	2	49
322	Vladamir Hauxley	1961-02-27	M	no medical history	1	43
445	Tobit Burniston	1938-08-13	M	no medical history	2	66
496	Daveen Lerwill	1915-05-03	F	weak immunity	0	3
508	Merilyn Goodrick	1929-05-15	F	undergone kidney transplant	0	46
553	Evangeline Roskeilly	1988-04-09	F	weak immunity	2	99
720	Henderson Hackinge	2006-11-11	M	undergone LASIK surgery	1	30

(B) This query selects medical professionals and the citizens they have advised in non-increasing order of citizens advised by each professional

```
SELECT citizen.p_id, p_name , COUNT(c_id) AS num_patients
FROM medical_professional, citizen WHERE citizen.p_id =
medical_professional.p_id GROUP BY p_id ORDER BY
COUNT(c_id) DESC;
```

p_id	p_name	num_patients
35	Karisa	27
23	Nico	27
47	Mart	27
46	Ced	27
9	Lincoln	26
4	Fawnia	25
7	Garrott	25
18	Waldemar	25
19	Ase	24
24	Dayle	23
22	Giusto	23
12	Gerti	23
40	Eldridge	23

(C) This query details of all medical professionals who are lungs or eye specialist and the citizens they have advised

```
SELECT medical_professional.p_id, p_name, p_type, c_id,
c_name FROM medical_professional, citizen WHERE
medical_professional.p_id = citizen.p_id AND p_type LIKE
'%eye%' OR '%lungs%';
```

p_id	p_name	p_type	c_id	c_name
7	Garrott	eyes	75	Onfre Skally
7	Garrott	eyes	101	Mignon Lonnon
7	Garrott	eyes	137	Sianna Peerless
7	Garrott	eyes	146	Artemus Charlesworth
7	Garrott	eyes	156	Giacobo Coulthard
7	Garrott	eyes	185	Arv Raisbeck
7	Garrott	eyes	186	Dalia McVittie
7	Garrott	eyes	308	Rena Saunier
7	Garrott	eyes	310	Cross Seilmann
7	Garrott	eyes	316	Elston Scholig
7	Garrott	eyes	434	Jaye Rosen
7	Garrott	eyes	448	Adrea O'Hern
7	Garrott	eyes	483	Sig Tieraney

(D) This query selects the details of all medical professionals and the senior citizens (>=60 yrs) they have advised along with the speciality of that professional arranged in non-increasing order of number of senior citizens per professional

```
SELECT citizen.p_id, p_name, p_type, COUNT(c_id) AS
num_patients FROM medical_professional, citizen WHERE
citizen.p_id=medical_professional.p_id AND
TIMESTAMPDIFF(YEAR, c_dob, CURDATE()) >=60 GROUP BY
citizen.p_id ORDER BY COUNT(c_id) DESC;
```

p_id	p_name	p_type	num_patients
32	Cloris	brain	17
19	Ase	lungs	16
16	Nestor	eyes	16
7	Garrott	eyes	16
12	Gerti	allergy	14
23	Nico	allergy	14
22	Giusto	liver	14
45	Marchall	brain	13
9	Lincoln	heart	13
4	Fawnia	liver	13
40	Eldridge	kidney	13
6	Lem	allergy	12

(E) This query selects the details of all medical professionals and vaccine manufacturers which have >= 150 orders still remaining

```
SELECT vaccine_manufacturer.p_id, p_name, m_id, m_name,
orders_remaining FROM vaccine_manufacturer,
medical_professional WHERE medical_professional.p_id =
vaccine_manufacturer.p_id AND orders_remaining>=150 ORDER
BY orders_remaining DESC;
```

p_id	p_name	m_id	m_name	orders_remaining
26	Bay	3	pain relief biotech	200
23	Nico	4	headache formula biotech	182
44	Vanny	8	Morphine sulfate biotech	180
4	Fawnia	1	Antabuse biotech	152

3. Health Care Centre

(A) Select all health care centres where vaccines available are less than the citizens in that area who have not been vaccinated yet

```
SELECT * FROM health_care_centre WHERE
num_vaccine_available <
area_population-patients_vaccinated;
```

h_id	h_name	h_capacity	area_population	patients_vaccinated	e_id	num_
1	Winman health centre	869	367	176	100	
3	MacPhee health centre	675	391	130	98	
5	Caville health centre	549	314	141	96	
6	Haydney health centre	665	464	64	95	
7	Pavey health centre	486	417	67	94	
11	Hartfleet health centre	871	339	251	90	
12	Yukhin health centre	623	343	94	89	
14	Eastmond health centre	126	280	115	87	
15	Klimp health centre	659	395	28	86	
16	Loxley health centre	537	387	169	85	
19	Vize health centre	807	455	180	82	

(B) Select all those health centres who have vaccinated more patients than the average of all health centres combined

```
SELECT h_name FROM health_care_centre WHERE
patients_vaccinated > (SELECT AVG(patients_vaccinated) FROM
health_care_centre);
```

h_name
McElvine health centre
Spinige health centre
Hartfleet health centre
McCandie health centre
Brocks health centre
Follan health centre
Dundridge health centre
Simonelli health centre
Finn health centre
Gymlett health centre
Belloch health centre
Nann health centre

(C) Select all citizens registered with the “Finn Health Centre”

```
SELECT * from citizen as C1, health_care_centre as C2 WHERE
C1.h_id = C2.h_id AND LOWER(C2.h_name)='finn health
centre';
```

c_id	c_name	c_dob	c_gender	c_medicalhistory	c_vaccinated	C1.h_id	p_id
103	Joana Crayker	1913-05-08	F	weak muscle and nerves	2	27	3
219	Jermayne Parsonage	1936-11-30	M	gall bladder removed	2	27	1
246	Merralee Matisse	1927-12-30	F	gall bladder removed	1	27	5
295	Berkly Chastaing	1997-08-28	M	heart patient	0	27	4
343	Bay Baroux	2001-11-30	M	breathing issues	0	27	1
362	Archy Probart	1917-04-15	M	undergone kidney transplant	1	27	3
499	Marion Bysshe	1971-01-31	F	undergone LASIK surgery	2	27	2
597	Levin Purcell	2008-03-06	M	no medical history	1	27	3
623	Nelie Finnick	1992-11-05	F	no medical history	2	27	3
722	Phillipe Ginley	1960-09-28	M	no medical history	0	27	4
736	Laura Torrie	1946-09-23	F	heart patient	2	27	2
756	Betty Mcmanaman	2006-11-27	F	breathing issues	2	27	1
827	Dewitt Carley	1931-08-20	M	weak muscle and nerves	0	27	4

(D) Select details of health care centre with the minimum capacity

```
SELECT * from health_care_centre WHERE h_capacity = (SELECT
MIN(h_capacity) FROM health_care_centre);
```

h_id	h_name	h_capacity	area_population	patients_vaccinated	e_id	num_vacci
30	Smallpeice health centre	100	173	17	71	

(E) Select details of all health centres, corresponding citizens and distance between them in ‘km’ such that the name of health centres starts with ‘L’ and printed in non-decreasing order of distance between them

```
SELECT health_care_centre.h_id, health_care_centre.h_name,
citizen.c_id, citizen.c_name,
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude))/1000 AS distance FROM
```

```
health_care_centre, citizen, citizen_location,
health_centre_location WHERE citizen.h_id =
health_care_centre.h_id AND citizen.h_id =
health_centre_location.h_id AND citizen.c_id =
citizen_location.c_id AND h_name LIKE 'L%' ORDER BY
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude));
```

h_id	h_name	c_id	c_name	distance
16	Loxley health centre	568	Nikolia Winterflood	1907.9992787523704
38	Lillie health centre	21	Adolphus Maggiori	2406.7960571807957
16	Loxley health centre	829	Valli Philson	2505.401830731838
16	Loxley health centre	107	Sindee Carnalan	2592.3999123827602
38	Lillie health centre	226	Birdie Rossiter	2748.428047000507
16	Loxley health centre	146	Artemus Charlesworth	3327.376887104121
90	Lange health centre	954	Fionnula Joppich	3593.2437653846655
97	Learned health centre	405	Arlinda Letchford	3870.8657920480828
90	Lange health centre	615	Magda Marlor	3876.233037087223
97	Learned health centre	296	Rois Medcraft	4090.0630963537697
38	Lillie health centre	77	Lucinda Aynscombe	4390.807342260873
97	Learned health centre	65	Evvie Kimmings	4492.238004593901

4. Vaccine Distributor (and Manufacturer)

(A) This query selects all vaccines which were manufactured first

```
SELECT * FROM vaccine WHERE v_manufacture_date = (SELECT
MIN(v_manufacture_date) FROM vaccine);
```

m_id	v_id	v_name	v_type	v_price	v_manufacture_date	v_expiry_date	storage_t
9	2	Ibuprofen	nasal spray	170833.28	2020-03-14	2022-07-23	
10	1	Hydrocortisone Acetate	syrup	599146	2020-03-14	2022-09-17	

(B) This query selects all those vaccines which are prices higher than the average price of all vaccines

```
SELECT * FROM vaccine WHERE v_price > (SELECT AVG(v_price)
FROM vaccine);
```

m_id	v_id	v_name	v_type	v_price	v_manufacture_date	v_expiry_date
2	9	levothyroxine sodium	nasal spray	555320.5	2020-06-02	2022-12-12
4	7	Titanium Dioxide	nasal spray	582129.94	2020-09-01	2022-04-30
6	5	CYCLOBENZAPRINE HYDROCHLORIDE	nasal spray	562746.3	2020-04-23	2022-09-16
7	4	RANOLAZINE	nasal spray	573610.94	2020-11-20	2022-11-11
10	1	Hydrocortisone Acetate	syrup	599146	2020-03-14	2022-09-17

(C) This query selects all vaccines whose storage temperature is < 0 degree celsius and the type is syringe

```
SELECT * FROM vaccine WHERE storage_temp < 0 AND v_type
LIKE '%syringe%';
```

m_id	v_id	v_name	v_type	v_price	v_manufacture_date	v_expiry_date
3	8	Lidocaine Hydrochloride	syringe	362193.9	2020-11-02	2022-09-24

(D) This query selects details of all vaccine manufacturers whose orders remaining are more than they have completed

```
SELECT * FROM vaccine_manufacturer WHERE orders_remaining >
orders_completed;
```

m_id	m_name	m_capacity	m_num_employees	orders_completed	orders_remaining	p_id
1	Antabuse biotech	265	430	109	152	
2	SF biotech	692	329	67	138	2
3	pain relief biotech	576	184	142	200	2
4	headache formula biotech	521	281	89	182	2
8	Morphine sulfate biotech	120	184	53	180	4
9	Ananassa Resina biotech	563	479	47	123	4

(E) This query selects details of all distributors which have a quantity of > 50 ordered by their corresponding health care centre

```
SELECT * FROM vaccine_distribution AS d1,
vaccine_distributor AS d2 WHERE d1.d_id=d2.d_id AND
d1.qty_ordered_T > 50;
```

h_id	d1.d_id	qty_ordered_T	d2.d_id	d_name	d_num_employees	orders_received	orders_r
2	5	82	5	Temp	56	481	
2	44	84	44	Viva	837	96	
4	48	64	48	Tin	721	245	
5	30	77	30	Stringtough	518	102	
6	12	61	12	Y-find	440	227	
6	20	81	20	Overhold	695	341	
7	41	71	41	Matsoft	588	131	
11	23	72	23	Zoolab	145	162	
11	35	81	35	Rank	622	465	
11	44	72	44	Viva	837	96	
12	20	73	20	Overhold	695	341	
12	34	99	34	Prodder	572	52	

5. Citizens

(A) This query returns the list of citizens with some specific allergies which can interfere with the vaccination process.

```
SELECT citizen.c_id, citizen.c_name, citizen.c_dob ,
citizen.c_gender FROM citizen WHERE citizen.c_vaccinated=0
AND LOWER(citizen.c_medicalhistory) LIKE '%allerg%';
```

c_id	c_name	c_dob	c_gender
33	Hettie Jeary	2001-06-10	F
41	Suellen Webling	1981-08-26	F
65	Evvie Kimmings	1919-12-24	F
69	Colleen McPhillips	1950-12-23	F
71	Glennie De Laspee	1940-10-15	F
76	Helen Nerger	1928-04-07	F
82	Margarette Coleborn	1974-07-26	F
100	Roxane Cowans	1997-11-02	F
212	Ravi Soame	2010-04-15	M
230	Pierre Clare	1957-06-18	M
232	Cam Binstead	1949-12-17	M
235	Raffaello Silverthorne	1941-05-02	M

(B) This query sorts and returns the list of unvaccinated citizens in decreasing order of age, allowing the senior citizens to be treated earlier.

```
SELECT citizen.c_id, citizen.c_name, citizen.c_dob,
citizen.c_gender FROM citizen WHERE citizen.c_vaccinated=0
ORDER BY TIMESTAMPDIFF(YEAR, c_dob, CURDATE()) DESC;
```

c_id	c_name	c_dob	c_gender
572	Corabelle Ovington	1910-04-14	F
718	Ingar Oswick	1910-10-19	M
891	Adelind Andryunin	1911-01-14	F
132	Symon Stidson	1911-07-23	M
717	Gil Klimuk	1911-06-21	M
853	Jobye Verbeek	1912-02-01	F
494	Mead Gorbell	1912-04-02	F
780	Jim Abella	1912-04-03	M
707	Catriona Drei	1913-05-11	F
40	Marika Garbutt	1914-05-30	F
569	Farlie Houselee	1914-08-09	M
936	Angeli Agass	1915-01-21	M
981	Lyon Winterburn	1914-10-06	M

(C) This query sorts and displays the areas which have more number of unvaccinated citizens i.e. sorts the list in decreasing order of number of unvaccinated citizens in an area.

```
SELECT h_id, count(c_id) as unvaccinated FROM citizen WHERE
c_vaccinated=0 GROUP BY h_id ORDER BY count(c_id) DESC;
```

h_id	unvaccinated
6	9
7	9
29	6
25	6
72	6
9	6
59	6

(D) This query displays the distance of senior citizen with heart ailments who have been vaccinated at least once from their health care centre

```
SELECT health_care_centre.h_id, health_care_centre.h_name,
citizen.c_id, citizen.c_name,
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude))/1000 AS distance FROM
health_care_centre, citizen, citizen_location,
health_centre_location WHERE citizen.h_id =
health_care_centre.h_id AND citizen.h_id =
health_centre_location.h_id AND citizen.c_id =
citizen_location.c_id AND TIMESTAMPDIFF(YEAR, c_dob,
CURDATE()) >=60 AND c_vaccinated>=1 AND
LOWER(c_medicalhistory) LIKE '%heart%' ORDER BY distance;
```

h_id	h_name	c_id	c_name	distance
72	Mathivon health centre	402	Christyna Pahlsson	522.2254484315147
33	Murkin health centre	871	Stanislaus Crollman	2268.07663506304
96	Bigrigg health centre	166	Merwyn Lemmer	3102.889125612346
55	Pezey health centre	325	Salim Romayn	5210.183328582501
89	Ioselevich health centre	540	Luca Louthe	5832.5997199948615
51	Bendin health centre	675	Patrick Brownett	6198.23052343543
22	Rider health centre	188	Prinz Likely	7812.5103480636135
60	Ginty health centre	366	Vito Willimot	8025.893634060536
66	McCard health centre	102	Josiah Tatlock	8103.583413316944
64	Cotman health centre	278	Maryrose Maly	8630.834628695859
77	Cossar health centre	733	Fanchon de Grey	8741.654412199281

(E) This query displays the ratio of vaccinated to non vaccinated citizens who were born in or after 1st January 2000.

```
SELECT (SELECT COUNT(*) FROM citizen WHERE c_vaccinated>=1
AND c_dob > '2000-01-01') / COUNT(*) AS ratio from citizen
WHERE c_dob > '2000-01-01';
```

ratio
0.6918

Note: The outputs contained many more rows and columns than shown in screenshot but they could not fit in the page, hence only a part is shown in some cases.

Ideas used in Queries

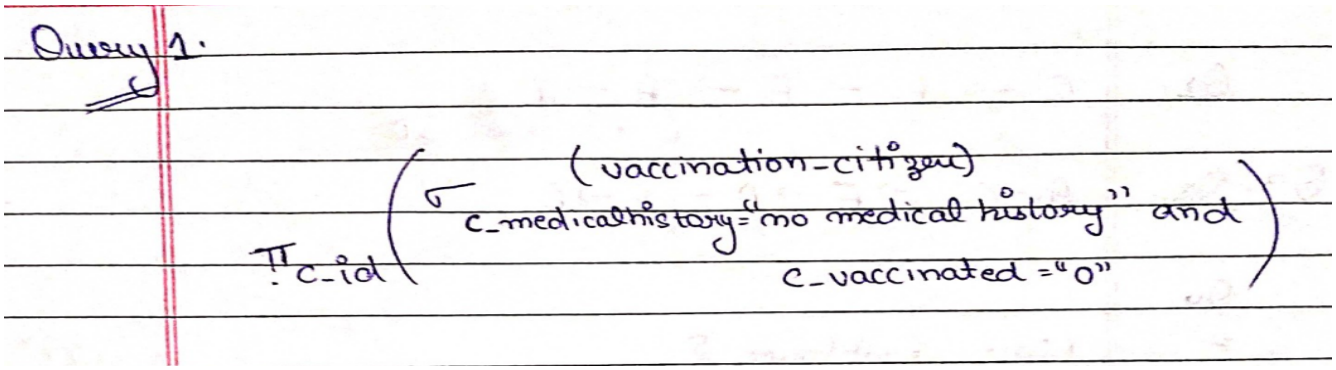
1. **Regular Expressions -** To detect the presence of a number in a password
2. **Age Calculation -** Used *TIMESTAMP* function of mysql to calculate the age from dob
3. **Distance Calculation -** Used ST_DISTANCE_SPHERE function of mysql to calculate distance between two points, given their latitudes and longitudes

Week-7,8,9

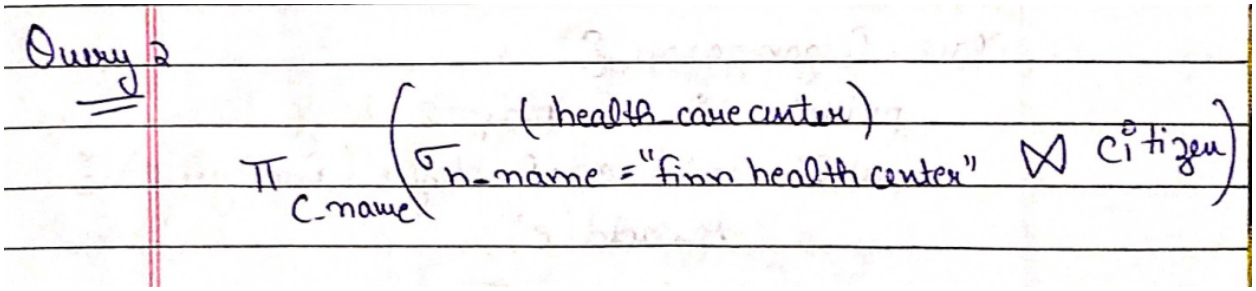
Mid Sem Evaluation and Relational Algebraic Queries

Relational Algebraic Queries

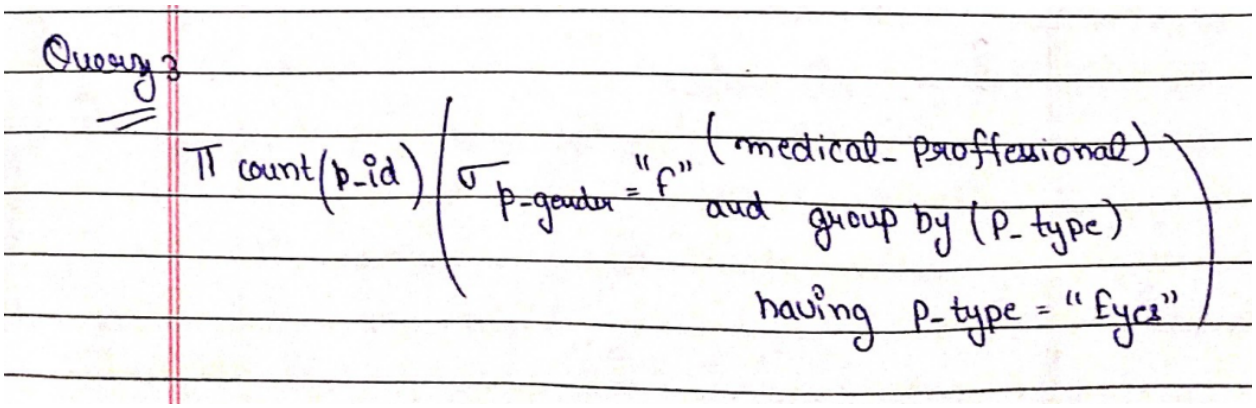
QUERY1 :: List the c_id of all citizens who registered themselves for vaccination and having no Medical history and are not vaccinated yet.



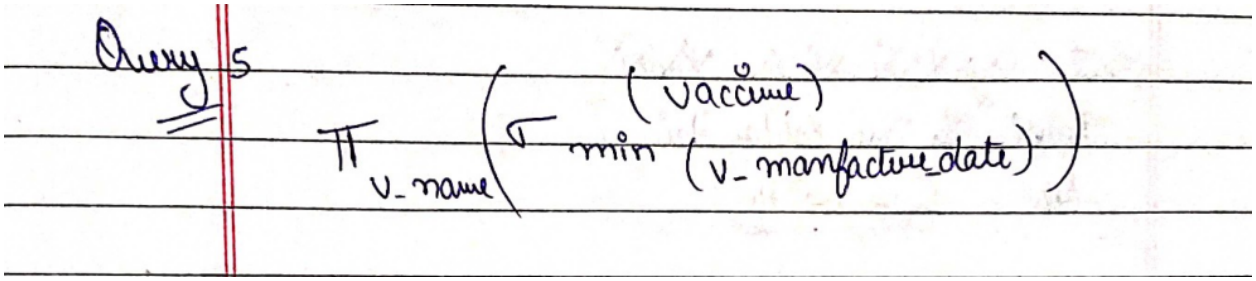
QUERY2 :: List the name of all citizens registered in "Finn Health Center".



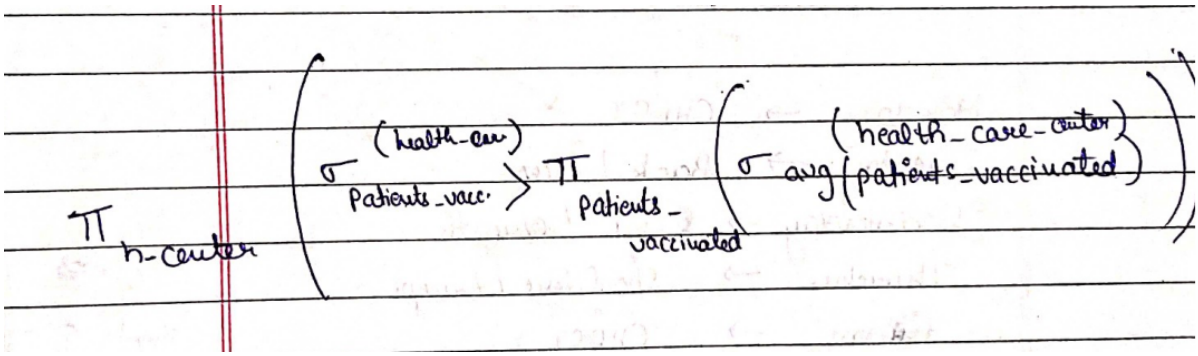
Query3 :: Mention the number of female medical professionals specialized in heart diseases.



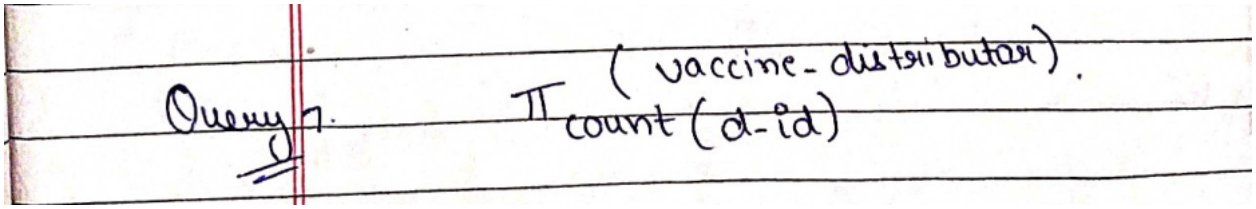
QUERY5 :: Name the first launched vaccine available for the vaccination process.



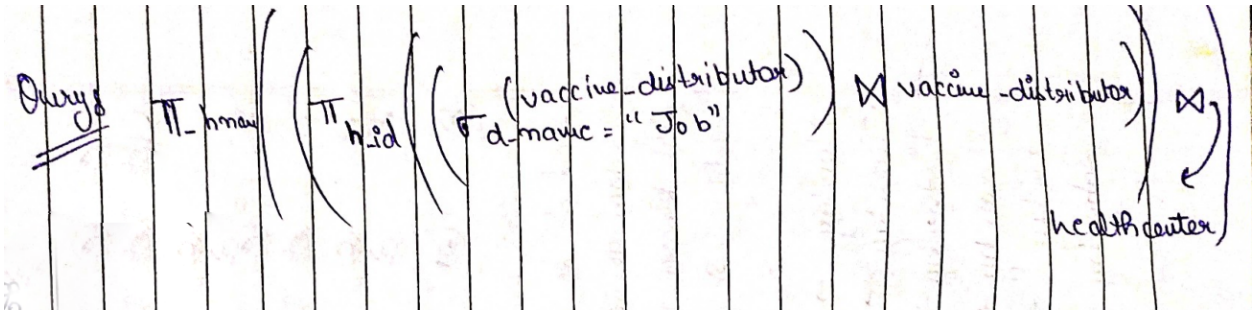
QUERY6 :: Find Health Centers name who vaccinated patients greater than the average vaccinated patients of the entire organisation.



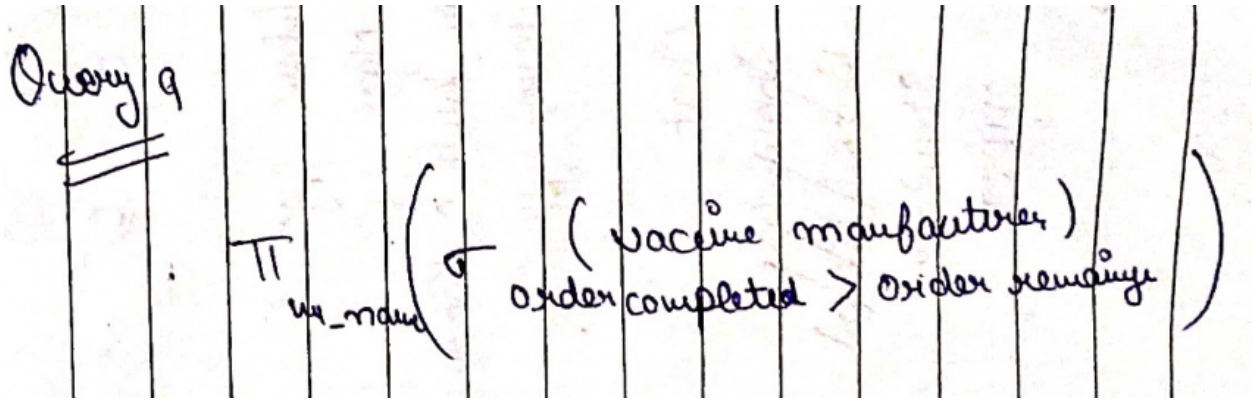
Query7 :: Count the total no. of distributors available for the delivery of vaccines.



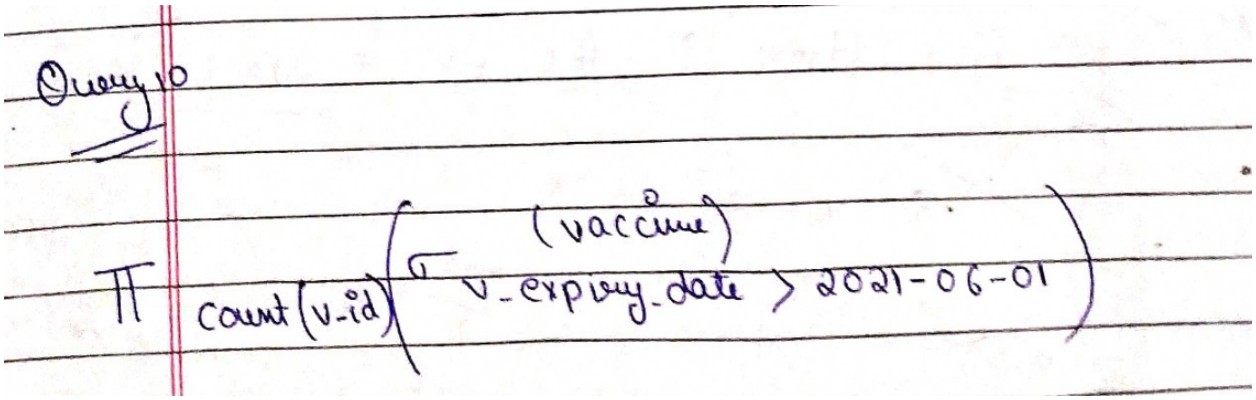
Query8 :: List the name of Health centers who got delivery of vaccine from distributor named "Jon"



Query9 :: Find the name of the vaccine manufacturer having more completed orders than remaining orders.



Query10 :: Count the number of vaccines available which will not expire till 2021-06-01.



Week-10

Indexing and mySQL Queries

Indexing

The commands written for indexing are as follows -

```
1. CREATE INDEX c_id ON citizen(c_id);
2. CREATE INDEX c_id ON citizen_location(c_id);
3. CREATE INDEX d_id ON distributor_location(d_id);
4. CREATE INDEX h_id ON health_care_centre(h_id);
5. CREATE INDEX h_id ON health_centre_location(h_id);
6. CREATE INDEX e_id ON health_ministry(e_id);
7. CREATE INDEX p_id ON medical_professional(p_id);
8. CREATE INDEX index1 ON ministry_consultation(p_id, e_id);
9. CREATE INDEX index2 ON vaccine_distribution(h_id, d_id);
10.  CREATE INDEX d_id ON vaccine_distributor(d_id);
11.  CREATE INDEX m_id ON vaccine_manufaturers(m_id);
12.  CREATE INDEX index3 ON vaccine_supply(m_id, d_id);
```

mySQL Queries

Query1. This query returns the details of ministry employees who are in charge of health centres with the maximum area population

```
SELECT health_ministry.e_id, e_name, h_id, h_name,
area_population FROM health_ministry, health_care_centre WHERE
area_population IN (SELECT MAX(area_population) FROM
health_care_centre) AND health_ministry.e_id = (SELECT
health_care_centre.e_id FROM health_care_centre WHERE
area_population IN (SELECT MAX(area_population) FROM
health_care_centre));
```

Query 2. This query prints all ministry employees with their names and ids along with the number of medical professionals they are consulting and orders them in non-decreasing order of medical professionals

```
SELECT health_ministry.e_id, health_ministry.e_name, count(p_id)
as p_ids FROM health_ministry, ministry_consultation WHERE
health_ministry.e_id =  ministry_consultation.e_id group by e_id
having count(p_id)>=2 ORDER BY count(p_id) DESC;
```

Query3. This query returns the details of ministry employees who were in-charge of vaccines which got manufactured the latest.

```
SELECT e_id, e_name, v_name, v_manufacture_date FROM
health_ministry, vaccine WHERE v_manufacture_date

= (SELECT MAX(v_manufacture_date) FROM vaccine) AND e_id IN
(SELECT e_id FROM ministry_consultation WHERE p_id IN (SELECT
p_id FROM vaccine_manufacturer

WHERE m_id IN (SELECT m_id FROM vaccine WHERE v_manufacture_date
= (SELECT MAX(v_manufacture_date) FROM vaccine))));
```

Query 4. This query selects the details of all medical professionals and the senior citizens (>=60 yrs) they have advised along with the speciality of that professional arranged in non-increasing order of number of senior citizens per professional

```
SELECT citizen.p_id, p_name, p_type, COUNT(c_id) AS num_patients
FROM medical_professional, citizen WHERE
citizen.p_id=medical_professional.p_id AND TIMESTAMPDIFF(YEAR,
c_dob, CURDATE()) >=60 GROUP BY citizen.p_id ORDER BY
COUNT(c_id) DESC;
```

Query 5. This query details of all medical professionals who are lungs or eye specialist and the citizens they have advised

```
SELECT medical_professional.p_id, p_name, p_type, c_id, c_name
FROM medical_professional, citizen WHERE
medical_professional.p_id = citizen.p_id AND p_type LIKE '%eye%'
OR '%lungs%';
```

Query6. This query displays details of all medical professionals along with the citizens they have advised.

```
SELECT * FROM medical_professional, citizen WHERE
medical_professional.p_id=citizen.p_id;a
```

Query 7. Select details of all health centres, corresponding citizens and distance between them in 'km' such that the name of health centres starts with 'L' and printed in non-decreasing order of distance between them

```
SELECT health_care_centre.h_id, health_care_centre.h_name,
citizen.c_id, citizen.c_name,
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude))/1000 AS distance FROM
health_care_centre, citizen, citizen_location,
health_centre_location WHERE citizen.h_id =
health_care_centre.h_id AND citizen.h_id =
health_centre_location.h_id AND citizen.c_id =
citizen_location.c_id AND h_name LIKE 'L%' ORDER BY
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
```

```
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude));
```

Query 8. Select details of the health care centre with the minimum capacity.

```
SELECT * from health_care_centre WHERE h_capacity = (SELECT
MIN(h_capacity) FROM health_care_centre);
```

Query 9. Select all those health centres who have vaccinated more patients than the average of all health centres combined

```
SELECT h_name FROM health_care_centre WHERE patients_vaccinated
> (SELECT AVG(patients_vaccinated) FROM health_care_centre);
```

Query 10. This query selects all those vaccines which are prices higher than the average price of all vaccines

```
SELECT * FROM vaccine WHERE v_price > (SELECT AVG(v_price) FROM
vaccine);
```

Query 11. This query selects all vaccines whose storage temperature is < 0 degree celsius and the type is syringe

```
SELECT * FROM vaccine WHERE storage_temp < 0 AND v_type LIKE
'%syringe%';
```

Query12.This query selects details of all distributors which have a quantity of > 50 ordered by their corresponding health care centre

```
SELECT * FROM vaccine_distribution AS d1,
vaccine_distributor AS d2 WHERE d1.d_id=d2.d_id AND
d1.qty_ordered_T > 50;
```

Query 13.This query sorts and displays the areas which have more number of unvaccinated citizens i.e. sorts the list in decreasing order of number of unvaccinated citizens in an area.

```
SELECT h_id, count(c_id) as unvaccinated FROM citizen WHERE
c_vaccinated=0 GROUP BY h_id ORDER BY count(c_id) DESC;
```

Query 14.This query displays the distance of senior citizen with heart ailments who have been vaccinated at least once from their health care centre

```
SELECT health_care_centre.h_id, health_care_centre.h_name,
citizen.c_id, citizen.c_name,
ST_DISTANCE_SPHERE(POINT(citizen_location.longitude,
citizen_location.latitude ),
POINT(health_centre_location.longitude,
health_centre_location.latitude))/1000 AS distance FROM
health_care_centre, citizen, citizen_location,
health_centre_location WHERE citizen.h_id =
health_care_centre.h_id AND citizen.h_id =
health_centre_location.h_id AND citizen.c_id =
citizen_location.c_id AND TIMESTAMPDIFF(YEAR, c_dob, CURDATE())
```

```
>=60 AND c_vaccinated>=1 AND LOWER(c_medicalhistory) LIKE
'%heart%' ORDER BY distance;
```

Query15.This query sorts and returns the list of unvaccinated citizens in decreasing order of age, allowing the senior citizens to be treated earlier.

```
SELECT citizen.c_id, citizen.c_name, citizen.c_dob,
citizen.c_gender FROM citizen WHERE citizen.c_vaccinated=0
ORDER BY TIMESTAMPDIFF(YEAR, c_dob, CURDATE()) DESC;
```

Week-11

Embedded SQL Queries

Embedded SQL Queries

1. This query selects details of all ministry employees such that the distributor they are supervising has the maximum number of pending orders

```
SELECT e_id, e_name, e_experience, orders_remaining
FROM health_ministry, vaccine_distributor WHERE
orders_remaining IN (SELECT MAX(orders_remaining) FROM
vaccine_distributor) AND e_id IN (SELECT e_id FROM
health_care_centre WHERE h_id IN (SELECT h_id FROM
vaccine_distribution WHERE d_id IN (SELECT d_id FROM
vaccine_distributor WHERE orders_remaining IN (SELECT
MAX(orders_remaining) FROM vaccine_distributor))));
```

2. This query returns the details of ministry employees who were in-charge of vaccines which got manufactured the latest.

```
SELECT e_id, e_name, v_name, v_manufacture_date FROM
health_ministry, vaccine WHERE v_manufacture_date =
(SELECT MAX(v_manufacture_date) FROM vaccine) AND e_id
IN (SELECT e_id FROM ministry_consultation WHERE p_id
IN (SELECT p_id FROM vaccine_manufacturer WHERE m_id
IN (SELECT m_id FROM vaccine WHERE v_manufacture_date
= (SELECT MAX(v_manufacture_date) FROM vaccine))));
```

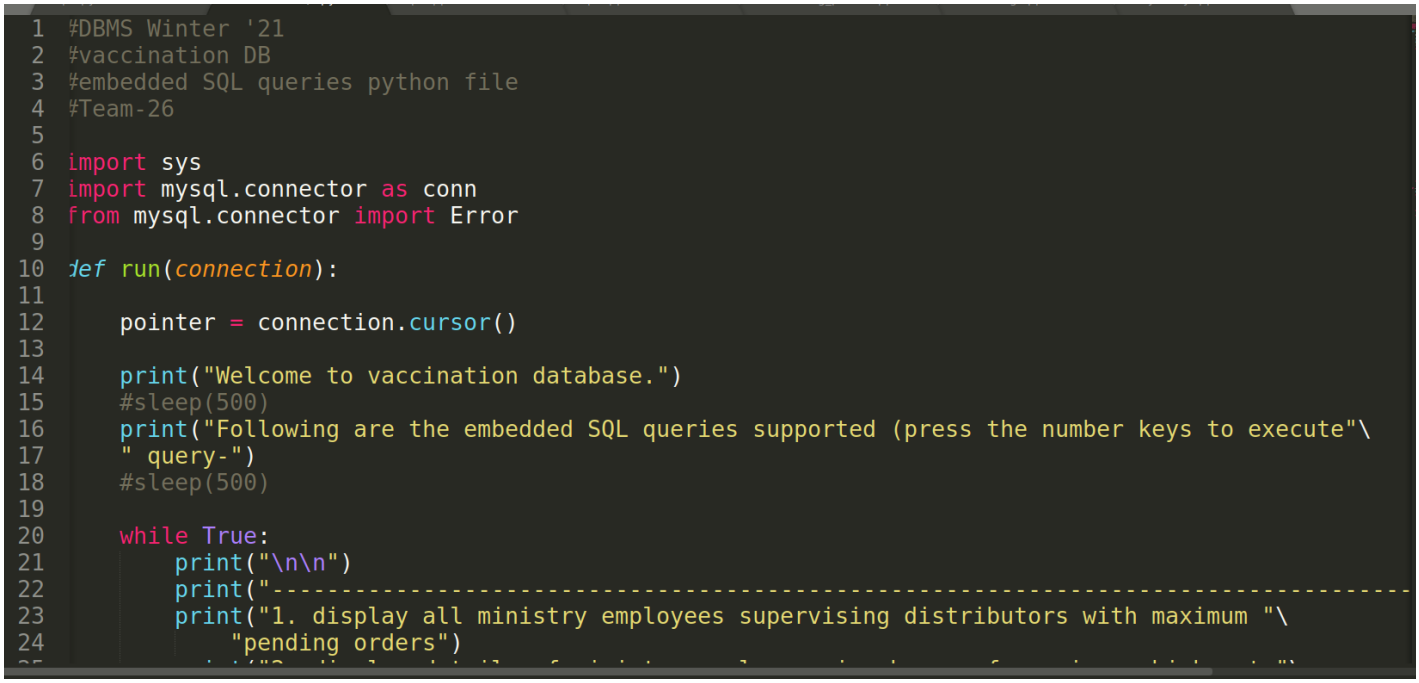
3. This query selects the details of all medical professionals and vaccine manufacturers which have >= 150 orders still remaining

```
SELECT vaccine_manufacturer.p_id, p_name, m_id, m_name,
orders_remaining FROM vaccine_manufacturer,
medical_professional WHERE medical_professional.p_id =
vaccine_manufacturer.p_id AND orders_remaining>=150 ORDER
BY orders_remaining DESC;
```

4. Sorts unvaccinated citizens in decreasing order of age

```
SELECT citizen.c_id, citizen.c_name, citizen.c_dob,
citizen.c_gender FROM citizen WHERE citizen.c_vaccinated=0
ORDER BY TIMESTAMPDIFF(YEAR, c_dob, CURDATE()) DESC;
```

Screenshots



```
#DBMS Winter '21
#vaccination DB
#embedded SQL queries python file
#Team-26

import sys
import mysql.connector as conn
from mysql.connector import Error

def run(connection):

    pointer = connection.cursor()

    print("Welcome to vaccination database.")
    #sleep(500)
    print("Following are the embedded SQL queries supported (press the number keys to execute"\
    " query-")
    #sleep(500)

    while True:
        print("\n\n")
        print("-------------------------------------------------------------------------------------------")
        print("1. display all ministry employees supervising distributors with maximum "\
            "pending orders")
```

```
        print("1. display all ministry employees supervising distributors with maximum "\
            "pending orders")
        print("2. display details of ministry employees in charge of vaccines which got "\
            "manufactured the latest")
        print("3. show all medical professionals and vaccine manufactureres having >=150 "\
            "orders remaining")
        print("4. display unvaccinated citizens in decreasing order of their ages")
        print("5. exit the program")

        choice = int(input())

        if choice==5:
            break

        elif choice==4:
            query = "select citizen.c_id, citizen.c_name, citizen.c_dob, citizen.c_gender"\
            " FROM citizen WHERE citizen.c_vaccinated=0 ORDER BY TIMESTAMPDIFF(YEAR, "\
            "c_dob, CURDATE()) DESC;"

            pointer.execute(query)
            result = pointer.fetchall()

            print("query executed successfully\n")
            print("(citizen_id, citizen_name, citizen_dob, citizen_gender)\n\n")
```

```
            print("(citizen_id, citizen_name, citizen_dob, citizen_gender)\n\n")
            for value in result:
                print(value)

        elif choice==3:
            query = "select vaccine_manufacturer.p_id, p_name, m_id, m_name, "\
            "orders_remaining FROM vaccine_manufacturer, medical_professional WHERE "\
            "medical_professional.p_id = vaccine_manufacturer.p_id AND orders_remaining>=150"\
            " ORDER BY orders_remaining DESC;"
            pointer.execute(query)
            result = pointer.fetchall()

            print("query executed successfully\n")
            print("(medical_professional_id, medical_professional_name, manufacturer_id,"\
            "manufacturer_name, orders_remaining)\n\n")
            for value in result:
                print(value)

        elif choice==2:
            query = "select e_id, e_name, v_name, v_manufacture_date FROM health_ministry,"\
            "vaccine WHERE v_manufacture_date = (SELECT MAX(v_manufacture_date) FROM vaccine) "\
            "AND e_id IN (SELECT e_id FROM ministry_consultation WHERE p_id IN (SELECT "\
            "p_id FROM vaccine_manufacturer WHERE m_id IN (SELECT m_id FROM vaccine WHERE "\
            "v_manufacture_date = (SELECT MAX(v_manufacture_date) FROM vaccine))));"
```

```
            pointer.execute(query)
            result = pointer.fetchall()

            print("query executed successfully\n")
            print("(employee_id, employee_name, vaccine_name, manufacture_date)\n\n")
            for value in result:
                print(value)

        elif choice==1:
            query = "select e_id, e_name, e_experience, orders_remaining FROM "\
            "health_ministry, vaccine_distributor WHERE orders_remaining IN (SELECT "\
            "MAX(orders_remaining) FROM vaccine_distributor) AND e_id IN (SELECT e_id "\
            "FROM health_care_centre WHERE h_id IN (SELECT h_id FROM vaccine_distribution"\
            " WHERE d_id IN (SELECT d_id FROM vaccine_distributor WHERE orders_remaining "\
            "IN (SELECT MAX(orders_remaining) FROM vaccine_distributor))));"

            pointer.execute(query)
            result = pointer.fetchall()

            print("query executed successfully\n")
            print("(employee_id, employee_name, employee_experience, orders_remaining)\n\n")
            for value in result:
                print(value)
```

```
            else:
                break

    pointer.close()
    connection.close()
    return

if __name__ == "__main__":

    print("connecting to database...")

    try:
        connection = conn.connect(host='localhost',
                                  database='vaccination',
                                  user='kone',
                                  password='Klol4365!')

        run(connection)
        #connection.close()

    except Error:
        print("error while connecting", Error)

    finally:

        print("database closed")
        exit(0)
```

Week-12-13

Creating User Permissions (Grants)

Following users were created as shown, with their respective permissions -

1. ```
   create user manufacturer@localhost identified by 'EndsemDemo@123';
   ```

   Permissions

   1. ```
      GRANT select, insert, delete, update on vaccine_manufacturer to manufacturer@localhost;
      ```
   2. ```
      GRANT select, insert, delete, update on vaccine to manufacturer@localhost;
      ```
   3. ```
      GRANT select on medical_professional to manufacturer@localhost;
      ```
   4. ```
      GRANT select, insert, delete, update on vaccine_supply to manufacturer@localhost;
      ```
   5. ```
      GRANT select on vaccine_distributor to manufacturer@localhost;
      ```

2. ```
   create user vdistributor@localhost identified by 'EndsemDemo@123';
   ```

   Permissions

   1. ```
      GRANT select, insert, delete, update on vaccine_distributor to distributor@localhost;
      ```
   2. ```
      GRANT select, insert, delete, update on distributor_location to distributor@localhost;
      ```
      ```

3. GRANT select on health_care_centre to distributor@localhost;
4. GRANT select, insert, delete, update on vaccine_distribution to distributor@localhost;
5. GRANT select, insert, delete, update on vaccine_supply to distributor@localhost;

3. create user citizen@localhost identified by 'EndsemDemo@123';

Permissions

1. GRANT select, insert, delete, update on citizen to citizen@localhost;
2. GRANT select, insert, delete, update on citizen_location to citizen@localhost;
3. GRANT select on medical_professional to citizen@localhost;
4. GRANT select on health_care_centre to citizen@localhost;

4. create user ministry_employee@localhost identified by 'EndsemDemo@123';

Permission

1. GRANT select, insert, delete, update on health_ministry to ministry_employee@localhost;
2. GRANT select, insert, delete, update on health_care_centre to ministry_employee@localhost;
3. GRANT select, insert, delete, update on ministry_consultation to ministry_employee@localhost;

5. `create user health_centre_staff@localhost identified by 'EndsemDemo@123';`

Permission

1. `GRANT select, insert, delete, update on health_care_centre to health_centre_staff@localhost;`
2. `GRANT select, insert, delete, update on health_centre_location to health_centre_staff@localhost;`
3. `GRANT select on health_ministry to health_centre_staff@localhost;`
4. `GRANT select, insert, delete, update on citizen to health_centre_staff@localhost;`
5. `GRANT select, insert, delete, update on vaccine_distribution to health_centre_staff@localhost;`

Week-13-14

UI Creation

A user interface supporting 4 queries was created using tkinter library available in python. User can enter new ministry employee details and view some other details according to the three other queries.

Code Screenshots

```
#DBMS Winter '21
#vaccination DB
#UI file
#Team-26

import sys
import mysql.connector as conn
from mysql.connector import Error
from tkinter import *
from PIL import ImageTk, Image
from tkinter import scrolledtext


#connect to db
#establish connection with db

try:
        connection = conn.connect(host='localhost',
                                  database='vaccination',
                                  user='kone',
                                  password='Klol4365!')

        pointer = connection.cursor()
        #run(connection)
        #connection.close()

except Error:
```

```
except Error:
        print("error while connecting", Error)
        exit(0)


class MyWindow:

    def __init__(self, win):

        self.toplabel = Label(win, text = 'COVID-19 DATA APPLICATION', fg = "white", bg = "black",
        self.mainlabel = Label(win, text = 'TEEKA-KARAN', fg = "black", bg = "white", font = "Helv
        self.mainlabel.place(x = 350, y = 70)
        self.toplabel.place(x = 10, y = 10)
        self.query = Label(win, text = 'Input', font = "Helvetica 16 bold")
        self.result = Label(win, text = 'Result', font = "Helvetica 16 bold")
        self.text_area1 = Entry(bd = 1)
        self.text_area1.place(x = 220, y = 150)
        self.query.place(x = 100, y = 150)


        self.exitt = Button(win, text='exit application', command=self.exitt, fg = "black", bg = '
        self.exitt.place(x = 250, y = 300)

        self.query1 = Button(win, text='enter new employee', command=self.query1, fg = "black", bg
        self.query1.place(x = 100, y = 220)

        self.query2 = Button(win, text='heart patient, senior citizens, vaccinated >=1 times', cor
```

```
        self.query1 = Button(win, text='enter new employee', command=self.query1, fg = "black", bg
        self.query1.place(x = 100, y = 220)

        self.query2 = Button(win, text='heart patient, senior citizens, vaccinated >=1 times', cor
        self.query2.place(x = 475, y = 220)

        self.query3 = Button(win, text='vaccines with price > avg price', command=self.query3, fg
        self.query3.place(x = 475, y = 150)

        self.query4 = Button(win, text='show employees grouped by experience', command=self.query4
        self.query4.place(x = 475, y = 290)

        self.result.place(x = 20, y = 300)
        self.text_area2 = scrolledtext.ScrolledText(win, width = 95, height = 10, font = ("Times N
        self.text_area2.grid(column = 0, pady = 10, padx = 10)
        self.text_area2.place(x = 10, y = 350)


    def query1(self):
        a,b,c,d = self.text_area1.get().split(',')
        query = "insert into health_ministry(e_name, e_dob, e_experience, e_password) values('" +
        pointer.execute(query)
        connection.commit()
        print("executed successfully")
        #self.text_area2.append(str(a)+" "+b+" "+c+" "+d))
```

```
    def query2(self):
        query="select health_care_centre.h_id, health_care_centre.h_name, citizen.c_id, "\
        "citizen.c_name, ST_DISTANCE_SPHERE(POINT(citizen_location.longitude, "\
        "citizen_location.latitude ), POINT(health_centre_location.longitude, "\
        "health_centre_location.latitude))/1000 AS distance FROM health_care_centre, "\
        "citizen, citizen_location, health_centre_location WHERE citizen.h_id = "\
        "health_care_centre.h_id AND citizen.h_id = health_centre_location.h_id AND "\
        "citizen.c_id = citizen_location.c_id AND TIMESTAMPDIFF(YEAR, c_dob, CURDATE())"\
        " >=60 AND c_vaccinated>=1 AND LOWER(c_medicalhistory) LIKE '%heart%' ORDER BY "\
        "distance;"

        pointer.execute(query)
        result = pointer.fetchall()

        ans="\nh_id, h_name, c_id, c_name, distance\n"

        for value in result:
            ans=ans+str(value)+"\n"

        self.text_area2.insert(END, ans)

```

```
    def query3(self):

        query="select * FROM vaccine WHERE v_price > (SELECT AVG(v_price) FROM vaccine);"

        pointer.execute(query)
        result = pointer.fetchall()

        ans="\nm_id, v_id, v_name, v_type, v_price, v_manufacture_date, expiry date, storage_temp

        for value in result:
            ans=ans+str(value)+"\n"

        self.text_area2.insert(END, ans)

    def query4(self):
        query="select count(*) as num_emp, e_experience FROM health_ministry group by e_experience

        pointer.execute(query)
        result = pointer.fetchall()

        ans="\nnum_emp, experience\n"

        for value in result:
            ans=ans+str(value)+"\n"

        self.text_area2.insert(END, ans)

```

```
118
119             for value in result:
120                 ans=ans+str(value)+"\n"
121
122             self.text_area2.insert(END, ans)
123
124         def exitt(self):
125             pointer.close()
126             connection.close()
127             exit(0)
128
129
130     root = Tk()
131
132     canv = Canvas(root, width = 1000, height = 600, bg='white')
133     canv.grid(row = 2, column = 3)
134
135     img = ImageTk.PhotoImage(Image.open("bg.jpeg"))
136     canv.create_image(0, 0, anchor = NW, image = img)
137
138
139     mywin = MyWindow(root)
140
141     root.title('HEALTH MINISTRY')
142
143     root.mainloop()
144
```

Output