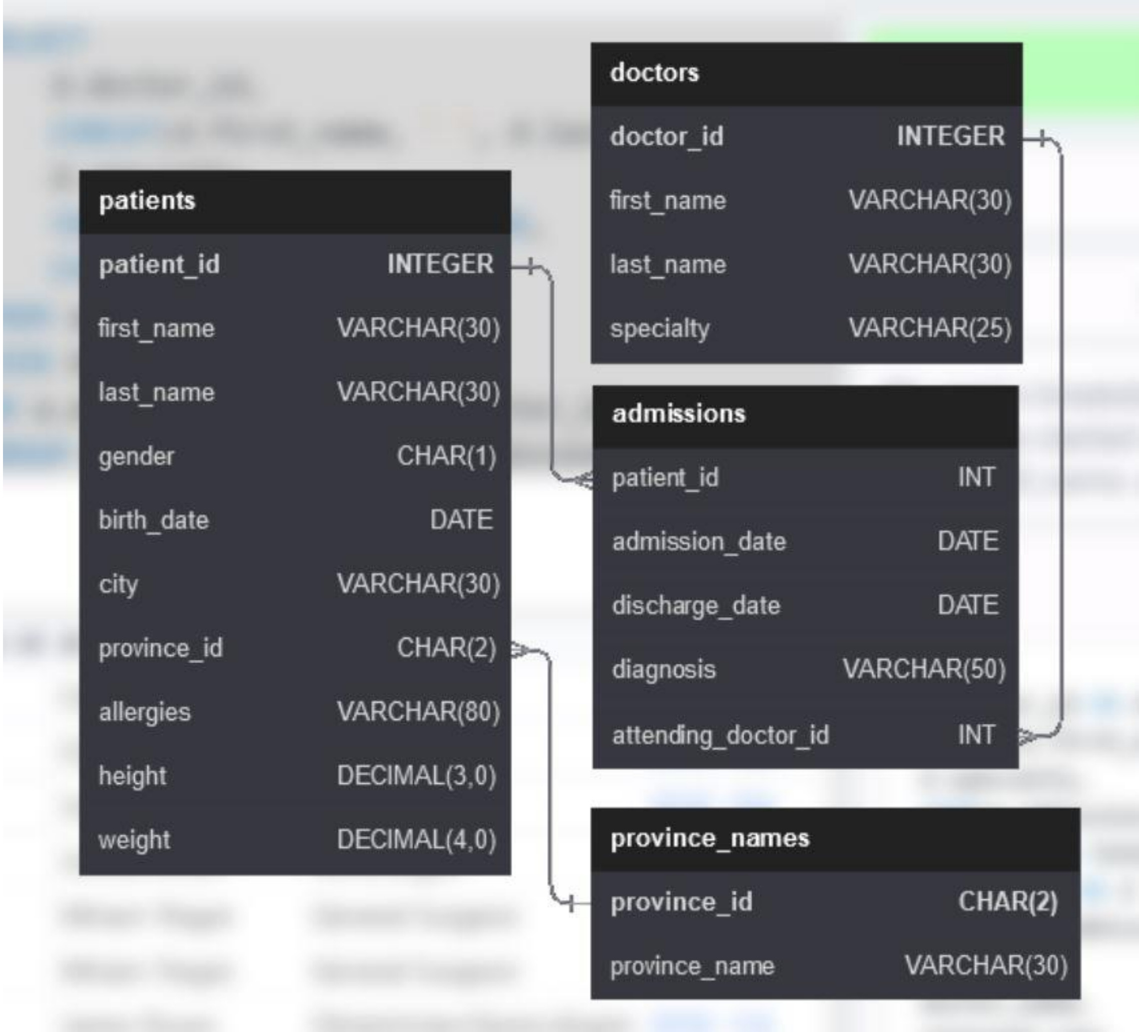
A database ‘hospital’ is provided and consists of four tables.



doctors	
doctor_id	INTEGER
first_name	VARCHAR(30)
last_name	VARCHAR(30)
specialty	VARCHAR(25)

patients	
patient_id	INTEGER
first_name	VARCHAR(30)
last_name	VARCHAR(30)
gender	CHAR(1)
birth_date	DATE
city	VARCHAR(30)
province_id	CHAR(2)
allergies	VARCHAR(80)
height	DECIMAL(3,0)
weight	DECIMAL(4,0)

admissions	
patient_id	INT
admission_date	DATE
discharge_date	DATE
diagnosis	VARCHAR(50)
attending_doctor_id	INT

province_names	
province_id	CHAR(2)
province_name	VARCHAR(30)

Also questions of different difficulty levels are given. All data was taken from https://www.sql-practice.com/.

Question #1
Show first name, last name, and gender of patients whose gender is 'M'.

```
SELECT first_name, last_name, gender
FROM patients
WHERE gender='M';
```

Question #2
Show first name and last name of patients who does not have allergies. (null)

```
SELECT first_name, last_name
FROM patients
WHERE allergies is NULL;
```

Question #3

Show first name of patients that start with the letter 'C'

```
SELECT first_name
FROM patients
WHERE first_name LIKE 'C%';
```

Question #4

Show first name and last name of patients that weight within the range of 100 to 120 (inclusive)

```
SELECT first_name, last_name
FROM patients
WHERE weight BETWEEN 100 AND 120;
```

Question #5

Update the patients table for the allergies column. If the patient's allergies is null then replace it with 'NKA'

```
UPDATE patients
SET allergies='NKA'
WHERE allergies IS NULL;
```

Question #6

Show first name and last name concatenated into one column to show their full name.

```
SELECT CONCAT(first_name, ' ', last_name) full_name
FROM patients;
```

Question #7

Show first name, last name, and the **full** province name of each patient.

```
SELECT patients.first_name, patients.last_name, province_names.province_name
FROM patients
JOIN province_names
```

```
ON patients.province_id=province_names.province_id;
```

Question #8

Show how many patients have a birth_date with 2010 as the birth year.

```
SELECT COUNT(*)
FROM patients
WHERE YEAR(birth_date)=2010;
```

Question #9

Show the first_name, last_name, and height of the patient with the greatest height.

```
SELECT first_name, last_name, MAX(height)
FROM patients;
```

Question #10

Show all columns for patients who have one of the following patient_ids: 1,45,534,879,1000

```
SELECT *
FROM patients
WHERE patient_id=1 OR patient_id=45 OR patient_id=534 OR patient_id=879
OR patient_id=1000;
```

Or:

```
SELECT *
FROM patients
WHERE patient_id IN (1, 45, 534, 879, 1000);
```

Question #11

Show the total number of admissions

```
SELECT COUNT(*) FROM admissions;
```

Question #12

Show all the columns from admissions where the patient was admitted and discharged on the same day.

```
SELECT *
FROM admissions
WHERE admission_date=discharge_date;
```

Question #13

Show the patient id and the total number of admissions for patient_id 579.

```
SELECT patient_id, COUNT(admission_date) total_admissions
FROM admissions
WHERE patient_id=579;
```

Question #14

Based on the cities that our patients live in, show unique cities that are in province_id 'NS'?

```
SELECT DISTINCT city
FROM patients
WHERE province_id='NS';
```

Question #15

Write a query to find the first_name, last name and birth date of patients who has height greater than 160 and weight greater than 70

```
SELECT first_name, last_name, birth_date
FROM patients
WHERE height>160 AND weight>70;
```

Question #16

Write a query to find list of patients first_name, last_name, and allergies where allergies are not null and are from the city of 'Hamilton'

```
SELECT first_name, last_name, allergies
FROM patients
WHERE allergies IS NOT null AND city='Hamilton';
```

Question #17

Show unique first names from the patients table which only occurs once in the list.

For example, if two or more people are named 'John' in the first_name column then don't include their name in the output list. If only 1 person is named 'Leo' then include them in the output.

```
SELECT first_name
FROM patients
GROUP BY first_name
HAVING COUNT(first_name)=1;
```

Question #18

Show patient_id and first_name from patients where their first_name start and ends with 's' and is at least 6 characters long.

```
SELECT patient_id, first_name
FROM patients
WHERE SUBSTR(first_name, 1, 1) LIKE 'S%' AND SUBSTR(first_name, -1) LIKE '%s' AND LEN(first_name)>=6;
```

Question #19

Show patient_id, first_name, last_name from patients whos diagnosis is 'Dementia'.

Primary diagnosis is stored in the admissions table.

```
SELECT p.patient_id, first_name, last_name
FROM patients p
JOIN admissions a
USING(patient_id)
WHERE diagnosis='Dementia';
```

Question #20

Display every patient's first_name.
Order the list by the length of each name and then by alphabetically.

```
SELECT first_name
FROM patients
ORDER BY LEN(first_name), first_name;
```

Question #21

Show the total amount of male patients and the total amount of female patients in the patients table.
Display the two results in the same row.

```
SELECT DISTINCT
(SELECT COUNT(gender) FROM patients WHERE gender='M') male,
(SELECT COUNT(gender) FROM patients WHERE gender='F') female
FROM patients;
```

Question #22

Show first and last name, allergies from patients which have allergies to either 'Penicillin' or 'Morphine'. Show results ordered ascending by allergies then by first_name then by last_name.

```
SELECT first_name, last_name, allergies
FROM patients
WHERE allergies IN ('Penicillin', 'Morphine')
ORDER BY allergies, first_name, last_name;
```

Question #23

Show patient_id, diagnosis from admissions. Find patients admitted multiple times for the same diagnosis.

```
SELECT patient_id, diagnosis
FROM admissions
GROUP BY patient_id, diagnosis
HAVING COUNT(*)>1;
```

Question #24

Show the city and the total number of patients in the city.
Order from most to least patients and then by city name ascending.

```
SELECT city, COUNT(patient_id) total_patients
FROM patients
GROUP BY city
ORDER BY total_patients DESC, city ASC;
```

Question #25

Show first name, last name and role of every person that is either patient or doctor. The roles are either "Patient" or "Doctor"

```
SELECT first_name, last_name, 'patient' role
FROM patients
UNION ALL
SELECT first_name, last_name, 'doctor' role
FROM doctors;
```

Question #26

Show all allergies ordered by popularity. Remove NULL values from query.

```
SELECT allergies, COUNT(allergies) total_allergies
FROM patients
WHERE allergies IS NOT NULL
GROUP BY allergies
ORDER BY total_allergies DESC;
```

Question #27

Show all patient's first_name, last_name, and birth_date who were born in the 1970s decade. Sort the list starting from the earliest birth_date.

```
SELECT first_name, last_name, birth_date
FROM patients
WHERE YEAR(birth_date) BETWEEN 1970 AND 1979
ORDER BY birth_date;
```

Question #28

We want to display each patient's full name in a single column. Their last_name in all upper letters must appear first, then first_name in all lower case letters. Separate the last_name and first_name with a comma. Order the list by the first_name in decending order
EX: SMITH,jane

```
SELECT CONCAT(UPPER(last_name), ',', LOWER(first_name)) full_name
FROM patients
```

```
ORDER BY first_name DESC;
```

Question #29

Show the province_id(s), sum of height; where the total sum of its patient's height is greater than or equal to 7,000.

```
SELECT province_id, SUM(height) total_height
FROM patients
GROUP BY province_id
HAVING total_height>=7000;
```

Question #30

Show the difference between the largest weight and smallest weight for patients with the last name 'Maroni'

```
SELECT MAX(weight)-MIN(weight)
FROM patients
WHERE last_name='Maroni';
```

Question #31

Show all of the days of the month (1-31) and how many admission_dates occurred on that day. Sort by the day with most admissions to least admissions.

```
SELECT DAY(admission_date) day, COUNT(*) total_admissions
FROM admissions
GROUP BY DAY(admission_date)
ORDER BY total_admissions DESC;
```

Question #32

Show all columns for patient_id 542's most recent admission_date.

```
SELECT *
FROM admissions
WHERE patient_id=542
ORDER BY admission_date DESC LIMIT 1;
```

Question #33

Show patient_id, attending_doctor_id, and diagnosis for admissions that match one of the two criteria:
1. patient_id is an odd number and attending_doctor_id is either 1, 5, or 19.
2. attending_doctor_id contains a 2 and the length of patient_id is 3 characters.

```
SELECT patient_id, attending_doctor_id, diagnosis
FROM admissions
WHERE (patient_id%2=1 AND attending_doctor_id IN (1, 5, 19))
OR (attending_doctor_id LIKE '%2%' AND LEN(patient_id)=3);
```

Question #34

Show first_name, last_name, and the total number of admissions attended for each doctor.

Every admission has been attended by a doctor.

```
SELECT d.first_name, d.last_name, COUNT(attending_doctor_id)
total_admissions
FROM doctors d
JOIN admissions a
ON a.attending_doctor_id=d.doctor_id
GROUP BY a.attending_doctor_id;
```

Question #35

For each doctor, display their id, full name, and the first and last admission date they attended.

```
SELECT d.doctor_id, CONCAT(d.first_name, ' ', d.last_name) full_name,
MIN(a.admission_date) first_admission, MAX(a.admission_date) last_admission
FROM doctors d
JOIN admissions a
ON d.doctor_id=a.attending_doctor_id
GROUP BY d.doctor_id;
```

Question #36

Display the total amount of patients for each province. Order by descending.

```
SELECT pro.province_name, COUNT(pa.patient_id) total_patients
FROM patients pa
JOIN province_names pro
ON pa.province_id=pro.province_id
```

```
GROUP BY pro.province_name
ORDER BY total_patients desc;
```

Question #37
For every admission, display the patient's full name, their admission diagnosis, and their doctor's full name who diagnosed their problem.

```
SELECT CONCAT(p.first_name, ' ', p.last_name) full_patient_name, a.diagnosis,
CONCAT(d.first_name, ' ', d.last_name) full_dr_name
FROM patients p
JOIN admissions a
ON p.patient_id=a.patient_id
JOIN doctors d
ON a.attending_doctor_id=d.doctor_id;
```

Question #38
Display the first name, last name and number of duplicate patients based on their first name and last name.

Ex: A patient with an identical name can be considered a duplicate.

```
SELECT first_name, last_name, COUNT(*)
FROM patients
GROUP BY first_name, last_name
HAVING COUNT(*) > 1;
```

Question #39
Display patient's full name,
height in the units feet rounded to 1 decimal,
weight in the unit pounds rounded to 0 decimals,
birth_date,
gender non abbreviated.
Convert CM to feet by dividing by 30.48.
Convert KG to pounds by multiplying by 2.205.

```
SELECT CONCAT(first_name, ' ', last_name) full_name, ROUND(height/30.48, 1) height, ROUND(weight*2.205, 0) weight, birth_date,
CASE
WHEN gender='M' THEN 'Male'
ELSE 'Female' END gender
FROM patients;
```

Question #40
Show patient_id, first_name, last_name from patients whose does not have any records in the admissions table. (Their patient_id does not exist in any admissions.patient_id rows.)

```
SELECT patient_id, first_name, last_name
FROM patients
WHERE patient_id NOT IN (SELECT patient_id FROM admissions);
```

Question #41
Show all of the patients grouped into weight groups.
Show the total amount of patients in each weight group.
Order the list by the weight group decending.
For example, if they weight 100 to 109 they are placed in the 100 weight group, 110-119 = 110 weight group, etc.

```
SELECT COUNT(*) total_patients,
CASE
	WHEN weight BETWEEN 0 AND 9 THEN 0
	WHEN weight BETWEEN 10 AND 19 THEN 10
	WHEN weight BETWEEN 20 AND 29 THEN 20
	WHEN weight BETWEEN 30 AND 39 THEN 30
	WHEN weight BETWEEN 40 AND 49 THEN 40
	WHEN weight BETWEEN 50 AND 59 THEN 50
	WHEN weight BETWEEN 60 AND 69 THEN 60
	WHEN weight BETWEEN 70 AND 79 THEN 70
	WHEN weight BETWEEN 80 AND 89 THEN 80
	WHEN weight BETWEEN 90 AND 99 THEN 90
	WHEN weight BETWEEN 100 AND 109 THEN 100
	WHEN weight BETWEEN 110 AND 119 THEN 110
	WHEN weight BETWEEN 120 AND 129 THEN 120
	WHEN weight BETWEEN 130 AND 139 THEN 130
	WHEN weight BETWEEN 140 AND 149 THEN 140
	ELSE null
	  END weight_group
FROM patients
GROUP BY
CASE
	WHEN weight BETWEEN 0 AND 9 THEN 0
	WHEN weight BETWEEN 10 AND 19 THEN 10
	WHEN weight BETWEEN 20 AND 29 THEN 20
	WHEN weight BETWEEN 30 AND 39 THEN 30
	WHEN weight BETWEEN 40 AND 49 THEN 40
```

```
        WHEN weight BETWEEN 50 AND 59 THEN 50
        WHEN weight BETWEEN 60 AND 69 THEN 60
        WHEN weight BETWEEN 70 AND 79 THEN 70
        WHEN weight BETWEEN 80 AND 89 THEN 80
        WHEN weight BETWEEN 90 AND 99 THEN 90
        WHEN weight BETWEEN 100 AND 109 THEN 100
        WHEN weight BETWEEN 110 AND 119 THEN 110
        WHEN weight BETWEEN 120 AND 129 THEN 120
        WHEN weight BETWEEN 130 AND 139 THEN 130
        WHEN weight BETWEEN 140 AND 149 THEN 140
        ELSE null
          END
ORDER BY weight_group DESC;
```

Question #42

Show patient_id, weight, height, isObese from the patients table.
Display isObese as a boolean 0 or 1.
Obese is defined as weight(kg)/(height(m)^2) >= 30.
weight is in units kg.
height is in units cm.

```
SELECT patient_id, weight, height,
CASE
          WHEN weight/POWER(height/100.0,2) >=30 THEN 1
          ELSE 0
END AS obesity
FROM patients;
```

Question #43

Show patient_id, first_name, last_name, and attending doctor's specialty.
Show only the patients who has a diagnosis as 'Epilepsy' and the doctor's first name is 'Lisa'

Check patients, admissions, and doctors tables for required information.

```
SELECT p.patient_id, p.first_name, p.last_name, d.specialty
FROM patients p
JOIN admissions a
ON p.patient_id=a.patient_id
JOIN doctors d
ON a.attending_doctor_id=d.doctor_id
WHERE a.diagnosis='Epilepsy' AND d.first_name='Lisa';
```

Question #44

All patients who have gone through admissions, can see their medical documents on our site. Those patients are given a temporary password after their first admission. Show the patient_id and temp_password.

The password must be the following, in order:
1. patient_id
2. the numerical length of patient's last_name
3. year of patient's birth_date

```
SELECT DISTINCT patient_id,
CONCAT(patient_id,LEN(p.last_name),YEAR(p.birth_date)) temp_password
FROM patients p
JOIN admissions a
USING(patient_id);
```

Question #45

Each admission costs $50 for patients without insurance, and $10 for patients with insurance. All patients with an even patient_id have insurance.

Give each patient a 'Yes' if they have insurance, and a 'No' if they don't have insurance. Add up the admission_total cost for each has_insurance group.

```
WITH insurance_table AS (
  SELECT
  CASE
WHEN patient_id%2=0 THEN 'yes'
ELSE 'no'
END AS has_insurance,
CASE
WHEN patient_id%2=0 THEN 10
ELSE 50
END AS cost_after_insurance
FROM admissions)

SELECT has_insurance, SUM(cost_after_insurance) total_cost_after_insurance
FROM insurance_table
GROUP BY has_insurance;
```

Question #46

Show the provinces that has more patients identified as 'M' than 'F'. Must only show full province_name.

```
SELECT province_name FROM
(SELECT province_id, COUNT(*) male
FROM patients
WHERE gender='M'
GROUP BY province_id, gender) male_count
JOIN
(SELECT province_id, count(*) female
FROM patients
WHERE gender='F'
GROUP BY province_id, gender) female_count
USING (province_id)
JOIN province_names
USING (province_id)
WHERE male>female;
```

Question #47

We are looking for a specific patient. Pull all columns for the patient who matches the following criteria:
- First_name contains an 'r' after the first two letters.
- Identifies their gender as 'F'
- Born in February, May, or December
- Their weight would be between 60kg and 80kg
- Their patient_id is an odd number
- They are from the city 'Kingston'

```
SELECT * FROM patients
WHERE first_name LIKE '__r%'
AND gender='F'
AND (MONTH(birth_date) IN (2, 5, 12))
AND (weight BETWEEN 60 AND 80)
AND patient_id%2=1
AND city='Kingston';
```

Question #48

Show the percent of patients that have 'M' as their gender. Round the answer to the nearest hundreth number and in percent form.

```
SELECT CONCAT(ROUND(100*((SELECT CAST(COUNT(*) as float) FROM
patients)
- CAST(COUNT(*) AS float))/(SELECT CAST(COUNT(*) AS float) FROM
patients), 2), '%') male_count FROM patients
WHERE gender='F';
```

Question #49
For each day display the total amount of admissions on that day. Display the amount changed from the previous date.

```
WITH admissions_table AS (
  SELECT admission_date, COUNT(*) admissions_per_day
  FROM admissions
  GROUP BY admission_date)

SELECT admission_date, admissions_per_day,
admissions_per_day-LAG(admissions_per_day)    OVER    (ORDER    BY
admission_date) admission_count_diff
FROM admissions_table;
```

Question #50
Sort the province names in ascending order in such a way that the province 'Ontario' is always on top.

```
SELECT province_name
FROM province_names
ORDER BY
CASE WHEN province_name='Ontario' THEN 1
ELSE 2
END,
province_name ASC;
```

Question #51
We need a breakdown for the total amount of admissions each doctor has started each year. Show the doctor_id, doctor_full_name, specialty, year, total_admissions for that year.

```
SELECT
```

```
d.doctor_id,
CONCAT(d.first_name, ' ', d.last_name) dr_full_name,
d.specialty,
YEAR(a.admission_date) year,
COUNT(*) total_admissions
FROM admissions a
JOIN doctors d
ON a.attending_doctor_id=d.doctor_id
GROUP BY doctor_id, year(a.admission_date);
```