https://www.sql-practice.com/

SQL-Practice Website

SQL DATABASE

SQL SCHEMA

TABLES:

patients		
Key Type	Column Name	Data Type
Primary Key	patient_id	INT
	first_name	TEXT
	last_name	TEXT
	gender	CHAR(1)
	birth_date	DATE
	city	TEXT
Foreign Key	province_id	CHAR(2)
	allergies	TEXT
	height	INT
	Weight	INT

admissions		
Key Type	Column Name	Data Type
Foreign Key	patient_id	INT
	admission_date	DATE
	discharge_date	DATE
	diagnosis	TEXT
Foreign Key	attending_doctor_id	INT

doctors		
Key Type	Column Name	Data Type
Foreign Key	doctor_id	INT
	first_name	TEXT
	last_name	TEXT
	specialty	TEXT

province_names		
Key Type	Column Name	Data Type
Foreign Key	province_id	CHAR(2)
	province_name	TEXT

Questions & Answers

Easy Questions

Questions 1:

Show first name, last name, and gender of patients who's gender is 'M'

Solution:

```
SELECT first_name, last_name, gender

FROM patients

WHERE gender= &amp;apos;M&amp;apos;;
```

Questions 2:

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

Solution:

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

Questions 3:

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

Solution:

```
SELECT
 first_name
FROM patients
WHERE first_name LIKE &amp;apos;C%&amp;apos;;
```

or,

```
SELECT first_name
FROM patients
WHERE substring (first_name, 1, 1) = &amp;apos;C&amp;apos;;
```

Questions 4:

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

Solution:

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

or,

```
SELECT
  first_name,
  last_name
FROM patients
WHERE weight >= 100 AND weight <= 120;
```

Questions 5:

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

Solution:

```
UPDATE patients
SET allergies = &amp;apos;NKA&amp;apos;
WHERE allergies IS NULL;
```

Questions 6:

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

Solution:

```
SELECT (first_name || &amp;apos; &amp;apos; || last_name) AS full_name
```

```
FROM patients;
```

or,

```
SELECT
  CONCAT (first_name, &apos; &apos;, last_name) AS full_name
FROM patients;
```

Questions 7:

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

Example: 'Ontario' instead of 'ON'

Solution:

```
SELECT patients.first_name, patients.last_name,
province_names.province_name
FROM patients
JOIN province_names
WHERE patients.province_id = province_names.province_id
GROUP BY patient_id;
```

Questions 8:

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

Solution:

```
SELECT COUNT (patient_id) AS total_patients
FROM patients
WHERE birth_date LIKE &apos;2010%&apos;;
```

or,

```
SELECT count(first_name) AS total_patients
FROM patients
```

```
WHERE
  birth_date >= &apos;2010-01-01&apos;
  AND birth_date <= &apos;2010-12-31’;
```

Questions 9:

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

Solution:

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

or,

```
SELECT
  first_name,
  last_name,
  height
FROM patients
ORDER BY height DESC
LIMIT 1;
```

or,

```
SELECT
  first_name,
```

```
  last_name,
  height
FROM patients
WHERE height = (
    SELECT max(height)
    FROM patients
  );
```

Questions 10:

Show all columns for patients who have one of the following patient_ids:

1,45,534,879,1000

Solution:

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

Questions 11:

Show the total number of admissions.

Solution:

```
SELECT COUNT (admission_date) AS total_addmissions
FROM admissions;
```

Questions 12:

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

Solution:

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

Questions 13:

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

Solution:

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

Questions 14:

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

Solution:

```
SELECT DISTINCT city as unique_cities
FROM patients
WHERE province_id = &amp;apos;NS&amp;apos;;
```

or,

```
SELECT city
FROM patients
GROUP BY city
HAVING province_id = &amp;apos;NS&amp;apos;;
```

Questions 15:

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

Solution:

```
SELECT first_name, last_name, birth_date

FROM patients

WHERE height > 160 AND weight >70;
```

Questions 16:

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

Solution:

```
SELECT first_name, last_name, allergies

FROM patients

WHERE city = &apos;Hamilton&apos; AND allergies IS NOT NULL;
```

Questions 17:

Based on cities where our patient lives in, write a query to display the list of unique cities starting with a vowel (a, e, i, o, u). Show the result order in ascending by city.

Solution:

```
SELECT DISTINCT city

FROM patients

WHERE (city like &apos;a%&apos;) OR

   (city like &apos;e%&apos;) OR

   (city like &apos;i%&apos;) OR

   (city like &apos;o%&apos;) OR

   (city like &apos;u%&apos;)
```

```
ORDER BY city ASC;
```

Medium Questions

Questions 1:

Show unique birth years from patients and order them by ascending.

Solution:

```
SELECT YEAR (birth_date) AS birth_year

FROM patients

GROUP BY YEAR (birth_date);
```

Questions 2:

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.

Solution:

```
SELECT first_name

FROM patients

GROUP BY first_name

HAVING COUNT (first_name) = 1;
```

or,

```
SELECT first_name
FROM (
   SELECT
    first_name,
    count(first_name) AS occurrencies
   FROM patients
   GROUP BY first_name )
WHERE occurrencies = 1;
```

Questions 3:

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

Solution:

```
SELECT patient_id, first_name
FROM patients
WHERE first_name LIKE &amp;apos;s____%s&amp;apos;;
```

or,

```
SELECT
 patient_id,
 first_name
FROM patients
WHERE
 first_name LIKE &amp;apos;s%s&amp;apos;
 AND len(first_name) >= 6;
```

or,

```
SELECT
```

```
  patient_id,
  first_name
FROM patients
WHERE
  first_name like &apos;s%&apos;
  AND first_name like &apos;%s&apos;
  AND len(first_name) >= 6;
```

Questions 4:

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

Primary diagnosis is stored in the admissions table.

Solution:

```
SELECT
  patients.patient_id,
  patients.first_name,
  patients.last_name
FROM patients
  JOIN admissions on patients.patient_id = admissions.patient_id
  WHERE diagnosis = &apos;Dementia&apos;;
```

or,

```
SELECT
  patient_id,
```

```
 first_name,
 last_name
FROM patients
WHERE patient_id IN (
  SELECT patient_id
  FROM admissions
  WHERE diagnosis = &apos;Dementia&apos;
 );
```

Questions 5:

Display every patient's first_name.

Order the list by the length of each name and then by alphabetically.

Solution:

```
SELECT first_name
FROM patients
ORDER BY LENGTH (first_name), first_name ASC;
```

Questions 6:

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.

Solution:

```
SELECT SUM (gender==&apos;M&apos;) AS male_count,
SUM (gender==&apos;F&apos;) AS female_count
FROM patients;
```

or,

```
SELECT
  (SELECT COUNT (*) FROM patients WHERE gender=&apos;M&apos;)
AS male_count,
  (SELECT COUNT (*) FROM patients WHERE gender=&apos;F&apos;)
AS female_count;
```

Questions 7:

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.

Solution:

```
SELECT first_name, last_name, allergies
FROM patients
WHERE allergies=&apos;Penicillin&apos; OR allergies =
&apos;Morphine&apos;
ORDER BY allergies, first_name, last_name ASC;
```

or,

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

```
  allergies,
  first_name,
  last_name;
```

or,

```
SELECT
  first_name,   last_name,  allergies
FROM
  patients
WHERE
  allergies = &apos;Penicillin&apos; OR allergies = &apos;Morphine&apos;
ORDER BY
  allergies ASC,
  first_name ASC,
  last_name ASC;
```

Questions 8:

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

Solution:

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

Questions 9:

Show the city and the total number of patients in the city.

Order from most to least patients and then by city name ascending.

Solution:

```
SELECT city, COUNT (patient_id) AS num_patients

FROM patients

GROUP BY city

ORDER BY COUNT (patient_id) DESC, city ASC;
```

Questions 10:

Show first name, last name and role of every person that is either patient or doctor.

The roles are either "Patient" or "Doctor".

Solution:

```
SELECT first_name, last_name, &amp;apos;Patient&amp;apos; AS role FROM patients

UNION ALL

SELECT first_name, last_name, &amp;apos;Doctor&amp;apos; AS role FROM doctors;
```

Questions 11:

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

Solution:

```
SELECT allergies, COUNT (*) AS total_diagnosis

FROM patients

WHERE allergies IS NOT NULL

GROUP BY allergies

ORDER BY total_diagnosis DESC;
```

or,

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

or,

```
SELECT
  allergies,
  COUNT (allergies) AS total_diagnosis
FROM patients
GROUP BY allergies
HAVING
  allergies IS NOT NULL
ORDER BY total_diagnosis DESC;
```

Questions 12:

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.

Solution:

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

or,

```
SELECT
  first_name,   last_name,  birth_date
FROM patients
WHERE
  birth_date >= &apos;1970-01-01&apos; AND birth_date < &apos;1980-01-01&apos;
ORDER BY birth_date ASC;
```

or,

```
SELECT
  first_name,
  last_name,
  birth_date
FROM patients
WHERE YEAR (birth_date) LIKE &apos;197%&apos;
ORDER BY birth_date ASC;
```

Questions 13:

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

Solution:

```
SELECT UPPER (last_name) ||&apos;,&apos;|| LOWER (first_name)
AS new_name_format
FROM patients
```

```
ORDER BY first_name DESC;
```

or,

```
SELECT
  CONCAT (UPPER (last_name), &apos;,&apos;, LOWER (first_name))
AS new_name_format
FROM patients
ORDER BY first_name DESC;
```

Questions 14:

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.

Solution:

```
SELECT province_id, SUM (height) AS sum_height
FROM patients
GROUP BY province_id
HAVING SUM (height)>=7000
ORDER BY sum_height DESC;
```

Questions 15:

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

Solution:

```
SELECT MAX (weight) - MIN (weight) AS weight_delta
FROM patients
WHERE last_name = &apos;Maroni&apos;;
```

Questions 16:

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.

Solution:

```
SELECT DAY (admission_date) AS day_number,
        COUNT (admission_date) AS number_of_admissions
FROM admissions
GROUP BY DAY (admission_date)
ORDER BY number_of_admissions DESC;
```

Questions 17:

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

Solution:

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

or,

```
SELECT *
FROM admissions
WHERE patient_id = 542
GROUP BY patient_id
HAVING
  admission_date = MAX (admission_date);
```

or,

```
SELECT *
FROM admissions
WHERE
  patient_id = &apos;542&apos;
  AND admission_date = (
    SELECT MAX (admission_date)
    FROM admissions
    WHERE patient_id = &apos;542&apos;
  );
```

or,

```
SELECT *
FROM admissions
GROUP BY patient_id
HAVING
  patient_id = 542
  AND MAX (admission_date);
```

Questions 18:

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.

Solution:

```
SELECT
 patient_id,
 attending_doctor_id,
 diagnosis
FROM admissions
WHERE
 (
 attending_doctor_id IN (1, 5, 19)
 AND patient_id % 2 != 0
 )
 OR
 (
 LEN (patient_id)=3
 AND
 attending_doctor_id LIKE &apos;%2%&apos;
 );
```

Questions 19:

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

Every admission has been attended by a doctor.

Solution:

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

or,

```
SELECT
  first_name,
  last_name,
  COUNT (*)
FROM
  doctors p,
  admissions a
where
  a.attending_doctor_id = p.doctor_id
GROUP BY p.doctor_id;
```

Questions 20:

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

Solution:

```
SELECT
  doctors.doctor_id,
  doctors.first_name||&amp;apos; &amp;apos;||doctors.last_name AS full_name,
  MIN (admissions.admission_date) AS first_admission_date,
  MAX (admissions.admission_date) AS last_admission_date
FROM doctors
  JOIN admissions on doctors.doctor_id = admissions.attending_doctor_id
GROUP BY admissions.attending_doctor_id;
```

Questions 21:

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

Solution:

```
SELECT
  province_names.province_name,
  COUNT (patients.patient_id) AS patient_count
FROM province_names
  JOIN patients ON patients.province_id = province_names.province_id
GROUP BY patients.province_id
ORDER BY patient_count DESC;
```

Questions 22:

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

Solution:

```
SELECT
  p.first_name || &amp;apos; &amp;apos; || p.last_name as patients_name,
  a.diagnosis,
  d.first_name||&amp;apos; &amp;apos;||d.last_name as doctor_name
FROM admissions a
JOIN patients p ON p.patient_id   = a.patient_id
JOIN doctors d ON d.doctor_id = a.attending_doctor_id;
```

Questions 23:

Display the number of duplicate patients based on their first_name and last_name.

Solution:

```
SELECT first_name, last_name, COUNT (*) AS num_of_duplicates
FROM patients
GROUP BY first_name, last_name
HAVING COUNT (first_name and last_name) = 2;
```

Questions 24:

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.

Solution:

```
SELECT
 first_name || &apos; &apos; || last_name AS patient_name,
 ROUND (height / 30.48, 1) AS &apos;height "Feet"&apos;,
 ROUND (weight * 2.205, 0) AS &apos;weight "Pound"&apos;,
 birth_date,
 CASE
  WHEN gender = &apos;M&apos; THEN &apos;MALE&apos;
  ELSE &apos;FEMALE&apos;
 END AS gender
FROM patients;
```

Hard Questions

Questions 1:

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 descending.

For example, if they weight 100 to 109 they are placed in the 100 weight group, 110-119 = 110 weight group, etc.

Solution:

```
SELECT
  COUNT (patient_id) AS patients_in_group,
  FLOOR (weight/10)*10 AS weight_group
FROM patients
GROUP BY weight_group
ORDER BY weight_group DESC;
```

Or,

```
SELECT
  TRUNCATE (weight, -1) AS weight_group,
  COUNT (*)
FROM patients
GROUP BY weight_group
ORDER BY weight_group DESC;
```

Or,

```
SELECT
  COUNT (patient_id),
```

```
  weight - weight % 10 AS weight_group
FROM patients
GROUP BY weight_group
ORDER BY weight_group DESC;
```

Questions 2:

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.

Solution:

```
SELECT
  patient_id,
  weight,
  height,
  CASE
  WHEN (weight/power((height*.01),2))>=30 THEN  1
  ELSE 0
  END AS isObese
FROM patients;
```

Or,

```
SELECT
  patient_id,
  weight,
  height,
```

```
  weight / POWER (CAST (height AS float) / 100, 2) >= 30 AS obese
FROM patients;
```

Questions 3:

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.

Solution:

```
SELECT
  p.patient_id as patient_id,
  p.first_name as patient_first_name,
  p.last_name as patient_last_name,
  d.specialty as attending_doctor_speciality
FROM admissions a
  JOIN patients p ON p.patient_id = a.patient_id
  JOIN doctors d ON a.attending_doctor_id = d.doctor_id
WHERE a.diagnosis = &amp;apos;Epilepsy&amp;apos; AND d.first_name = &amp;apos;Lisa&amp;apos;;
```

Or,

```
SELECT
  ph1.specialty
FROM patients AS pa
  JOIN (
```

```
  SELECT *
  FROM admissions AS a
    JOIN doctors AS ph ON a.attending_doctor_id = ph.doctor_id
) AS ph1 USING (patient_id)
WHERE
  ph1.diagnosis = &apos;Epilepsy&apos;
  AND ph1.first_name = &apos;Lisa&apos;;
```

Or,

```
SELECT
  a.patient_id,
  a.first_name,
  a.last_name,
  b.specialty
FROM
  patients a,
  doctors b,
  admissions c
```

```
WHERE
  a.patient_id = c.patient_id
  AND c.attending_doctor_id = b.doctor_id
  AND c.diagnosis = &apos;Epilepsy&apos;
  AND b.first_name = &apos;Lisa&apos;;
```

Or,

```
WITH patient_table AS (
    SELECT
      patients.patient_id,
      patients.first_name,
      patients.last_name,
      admissions.attending_doctor_id
    FROM patients
      JOIN admissions ON patients.patient_id = admissions.patient_id
    WHERE
      admissions.diagnosis = &apos;Epilepsy&apos;
  )
```

```
SELECT
  patient_table.patient_id,
  patient_table.first_name,
  patient_table.last_name,
  doctors.specialty
FROM patient_table
  JOIN doctors ON patient_table.attending_doctor_id = doctors.doctor_id
WHERE doctors.first_name = &apos;Lisa&apos;;
```

Questions 4:

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

Solution:

```
SELECT
  DISTINCT p.patient_id,
  CONCAT (
    p.patient_id,
    LEN (p.last_name),
```

```
    YEAR (p.birth_date)
  ) AS temp_password
FROM patients p
  JOIN admissions a ON a.patient_id = p.patient_id;
```

Or,

```
SELECT
  DISTINCT p.patient_id,
  p.patient_id || FLOOR (LEN (last_name)) || FLOOR (YEAR (birth_date))
AS temp_password
FROM patients p
  JOIN admissions a ON p.patient_id = a.patient_id
```

Or,

```
SELECT
  pa.patient_id,
  ad.patient_id || FLOOR (LEN (pa.last_name)) ||
FLOOR (YEAR (pa.birth_date)) AS temp_password
FROM patients pa
```

```
  JOIN admissions a ON p.patient_id = a.patient_id
GROUP BY pa.patient_id;
```

Questions 5:

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.

Solution:

```
SELECT
  CASE
    WHEN patient_id % 2 = 0 THEN &amp;apos;Yes&amp;apos;
    ELSE &amp;apos;No&amp;apos;
  END AS has_insurance,
  SUM (
    CASE
      WHEN patient_id % 2 = 0 THEN 10
      ELSE 50
```

```
    END
  ) AS cost_after_insurance
FROM admissions
GROUP BY has_insurance;
```

Or,

```
SELECT &apos;No&apos; AS has_insurance, COUNT (*) * 50 AS cost
FROM admissions WHERE patient_id % 2 = 1 GROUP BY has_insurance
UNION
SELECT &apos;Yes&apos; AS has_insurance, COUNT (*) * 10 AS cost
FROM admissions WHERE patient_id % 2 = 0 GROUP BY has_insurance;
```

Or,

```
SELECT has_insurance,
  CASE
    WHEN has_insurance = &apos;Yes&apos; THEN COUNT (has_insurance) *
10
    ELSE COUNT (has_insurance) * 50
  END AS cost_after_insurance
FROM (
```

```
  SELECT
    CASE
      WHEN patient_id % 2 = 0 THEN &apos;Yes&apos;
      ELSE &apos;No&apos;
    END AS has_insurance
  FROM admissions)
GROUP BY has_insurance;
```

Or,

```
SELECT has_insurance, SUM (admission_cost) AS admission_total
FROM
(
  SELECT patient_id,
  CASE WHEN patient_id % 2 = 0 THEN &apos;Yes&apos;
ELSE &apos;No&apos; END AS has_insurance,
  CASE WHEN patient_id % 2 = 0 THEN 10 ELSE 50 END AS admission_cost
  FROM admissions)
GROUP BY has_insurance;
```

Questions 6:

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

Solution:

```
SELECT pr.province_name
FROM province_names pr
```

```
  JOIN patients pt ON pr.province_id = pt.province_id
GROUP BY province_name
HAVING
  SUM (CASE WHEN gender = &apos;M&apos; THEN  1 ELSE 0 END) >
SUM (CASE WHEN gender = &apos;F&apos; THEN 1 ELSE 0 END);
```

Or,

```
SELECT province_name
FROM (
   SELECT
     province_name,
     SUM (gender = &apos;M&apos;) AS n_male,
     SUM (gender = &apos;F&apos;) AS n_female
   FROM patients pa
     JOIN province_names pr ON pa.province_id = pr.province_id
   GROUP BY province_name
 )
WHERE n_male > n_female;
```

Or,

```
SELECT pr.province_name
FROM patients AS pa
  JOIN province_names AS pr ON pa.province_id = pr.province_id
```

```
GROUP BY pr.province_name
HAVING
  SUM (gender = &apos;M&apos;) > SUM (gender = &apos;F&apos;);
```

Or,

```
SELECT province_name
FROM patients p
  JOIN province_names r ON p.province_id = r.province_id
GROUP BY province_name
HAVING
  SUM (CASE WHEN gender = &apos;M&apos; THEN 1 ELSE -1 END) > 0;
```

Or,

```
SELECT pr.province_name
FROM patients AS pa
  JOIN province_names AS pr ON pa.province_id = pr.province_id
GROUP BY pr.province_name
HAVING
  COUNT (CASE WHEN gender = &apos;M&apos; THEN 1 END) > COUNT (*) *
0.5;
```

Questions 7:

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'

Solution:

```
SELECT *

FROM patients

WHERE (first_name LIKE &amp;apos;__r%&amp;apos;)

  AND (gender = &amp;apos;F&amp;apos;)

  AND (MONTH (birth_date) = 2 OR MONTH (birth_date) = 5 OR MONTH
(birth_date) = 12)

  AND (weight BETWEEN 60 AND 80)

  AND (patient_id % 2 = 1)

  AND (city = &amp;apos;Kingston&amp;apos;);
```

Or,

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

Questions 8:

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

Solution:

```
SELECT
 CONCAT (ROUND (  (
     SUM (CASE
WHEN gender = &apos;M&apos; THEN 1
ELSE NULL
END) / CAST (
```

```
COUNT (*) AS float) * 100), 2), &apos;%&apos;)
AS percent_of_male_patients
FROM patients;
```

Or,

```
SELECT CONCAT (ROUND (
    (
      SELECT COUNT (*)
      FROM patients
      WHERE gender = &apos;M&apos;
    ) / CAST (COUNT (*) AS float),4) * 100, &apos;%&apos;)
AS percent_of_male_patients
FROM patients;
```

Or,

```
SELECT
  ROUND (100 * AVG (gender = &apos;M&apos;), 2) || &apos;%&apos;
AS percent_of_male_patients
FROM patients;
```

Or,

```
SELECT
   CONCAT (ROUND (SUM (gender=&apos;M&apos;) / CAST (COUNT (*)
AS float), 4) * 100, &apos;%&apos;)
```

```
FROM patients;
```

Questions 9:

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

Solution:

```
SELECT admission_date,
  COUNT (admission_date) AS admission_day,
  (COUNT (*) – LAG (COUNT (*)) OVER (ORDER BY admission_date))
AS admissions_change
FROM admissions
GROUP BY admission_date;
```

Or,

```
WITH admission_counts_table AS (
  SELECT admission_date, COUNT (patient_id) AS admission_count
  FROM admissions
  GROUP BY admission_date
  ORDER BY admission_date DESC
)
SELECT
  admission_date,
  admission_count,
```

admission_count – LAG (admission_count) OVER (ORDER BY admission_date) AS admission_count_change

FROM admission_counts_table;

Questions 10:

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

Solution:

```
SELECT   province_name

FROM province_names

ORDER BY (CASE WHEN province_name=&amp;apos;Ontario&amp;apos; THEN 0
ELSE 1 END),

province_name ASC;
```

Or,

```
SELECT province_name

FROM province_names

ORDER BY

 (NOT province_name = &amp;apos;Ontario&amp;apos;),

 province_name;
```

Or,

```
SELECT province_name

FROM province_names

ORDER BY

 province_name = &amp;apos;Ontario&amp;apos; DESC,

 province_name;
```

Questions No:

Solution: