Hospital Database: SQL Project

Objective

The objective of this project is to strengthen SQL skills by solving a wide range of query problems using a realistic hospital database schema. The exercises cover beginner to advanced topics such as filtering, JOINs, aggregation, subqueries, string manipulation, conditional logic, and analytic functions. The goal is to build confidence and readiness for real-world data analysis tasks and interviews.

Tools Used

- **Language: SQL**
- **Dataset: Hospital Management System**
- **Tables: patients, admissions, doctors, province_names.**

Full project on Github: https://github.com/Souvik362/Hospital-Database-SQL-Project

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";
```

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;
```

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

```
select first_name
from patients
where first_name like "C%";
```

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;
```

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;
```

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

```
select concat(first_name, " ", last_name)
from patients;
```

7. **Show first name, last name, and the full province name of each patient.**
   Example: 'Ontario' instead of 'ON'

```
select p.first_name, p.last_name, pn.province_name
from patients p
join province_names pn
on p.province_id = pn.province_id;
```

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

```
select count(patient_id)
from patients
where year(birth_date)=2010;
```

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

```
select first_name, last_name, height
from patients
where height= (select max(height)from patients);
```

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 in (1,45,534,879,1000);
```

11. **Show the total number of admissions**

```
select count(patient_id)
from admissions;
```

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;
```

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

```
select patient_id, count(admission_date)
from admissions
where patient_id= 579;
```

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

```
select distinct p.city
from patients p
join province_names pn
on p.province_id = pn.province_id
where p.province_id = "NS";
```

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;
```

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 not allergies is null and city = "Hamilton";
```

Medium

---

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

```
select distinct (year( birth_date))
from patients
order by 1;
```

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.

```
select first_name
from patients
group by 1
having count(first_name)=1;
```

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.**

```
select patient_id, first_name
from patients
where first_name like "s%s" and len(first_name)>= 6;
```

4. **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, p.first_name, p.last_name
from patients p
join admissions a
on p.patient_id= a.patient_id
where a.diagnosis= "Dementia";
```

5. **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), 1;
```

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.

```
select
count(case when gender = "M" then 1 end) as male_patients,
count(case when gender = "F" then 1 end) as female_patients
from patients;
```

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.**

```
select first_name, last_name, allergies
from patients
where allergies= "Penicillin" or allergies= "Morphine"
order by 3,1,2;
```

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

```
select patient_id, diagnosis
from admissions
group by 1,2
having count(*)>1;
```

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.

```
select city, count(patient_id)
from patients
group by 1
order by 2 desc,1;
```

10. **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" as role
from patients
union all
select first_name, last_name, "doctor" as role
from doctors;
```

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

```
select allergies, count(allergies)
from patients
where not allergies is null
group by 1
order by 2 desc;
```

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.**

```
select first_name, last_name,birth_date
from patients
where year(birth_date) between 1970 and 1979
order by 3;
```

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

```
select concat(upper(last_name),",", lower(first_name))
from patients
order by first_name desc;
```

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.**

```
select province_id, sum(height)
from patients
group by 1
having sum(height) >=7000;
```

15. **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";
```

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.**

```
select day(admission_date), count(patient_id)
from admissions
group by 1
order by 2 desc;
```

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

```
select patient_id, max(admission_date), discharge_date,
diagnosis,attending_doctor_id
from admissions
where patient_id=542
group by patient_id;
```

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.

```
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);
```

19. **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(a.attending_doctor_id)
from doctors d
join admissions a
on d.doctor_id= a.attending_doctor_id
group by d.doctor_id;
```

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

```
select a.attending_doctor_id, concat(d.first_name," ", d.last_name),
min(admission_date), max(admission_date)
from admissions a
join doctors d
on a.attending_doctor_id= d.doctor_id
group by 1;
```

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

```
select pn.province_name, count(p.patient_id)
from patients p
join province_names pn
on p.province_id= pn.province_id
group by p.province_id
order by 2 desc;
```

22. **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) AS patient_full_name,
 a.diagnosis, CONCAT(d.first_name, " ", d.last_name) AS doctor_full_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;
```

23. **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 1,2
having count(*)>1;
```

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.

```
select concat(first_name, " ", last_name) as full_name, round((height/30.48),1) as
height,
round(weight*2.205), birth_date, case when gender= "M" then "Male" else
"Female" end as gender
from patients;
```

25. **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 p.patient_id, p.first_name, p.last_name
FROM patients p
LEFT JOIN admissions a ON p.patient_id = a.patient_id
where a.patient_id is null;
```

26. **Display a single row with max_visits, min_visits, average_visits where the maximum, minimum and average number of admissions per day is calculated. Average is rounded to 2 decimal places.**

```
SELECT MAX(daily_visits) AS max_visits, MIN(daily_visits) AS min_visits,
ROUND(AVG(daily_visits), 2) AS average_visits
FROM (
SELECT admission_date, COUNT(*) AS daily_visits
FROM admissions
GROUP BY admission_date) AS daily_admissions;
```

Hard

---

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 decending.**

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

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

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

```
select patient_id, weight, height,
case when weight/power(height/100.0,2)>=30 then 1
else 0
end as isobese
from patients;
```

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.

```
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 like "%Lisa%";
```

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

```
select p.patient_id,
concat(p.patient_id,len(p.last_name),year(p.birth_date))
from patients p
join admissions a on p.patient_id= a.patient_id
group by 1;
```

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.

```
with insurance as(select case when patient_id%2=0 then "Yes"
else "No"
end as "has_Insurance"
from admissions),
cost_after_insurance as (select has_insurance, case when has_insurance="Yes"
then 10
else 50
end as "cost"
from insurance)

select has_insurance, sum(cost) as cost_after_insurance
from cost_after_insurance
group by 1;
```

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

```
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 = 'M' THEN 1 END) > COUNT(
CASE WHEN gender = 'F' THEN 1 END);
```

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'

```
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!=0
and city= "Kingston";
```

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

```
with male_count as
(select count(gender) as male
 from patients
 where gender="M"),
 total_patients as
(select count(*) as total_patients
 from patients)

 select CONCAT(ROUND(m.male * 100.0 / tp.total_patients, 2), '%') AS
percentage
 from male_count m, total_patients tp;
```

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

```
WITH daily_admissions AS (
 SELECT
 admission_date,
 COUNT(patient_id) AS admissions_day
 FROM admissions
 GROUP BY admission_date)
 SELECT
 admission_date,
 admissions_day,
 admissions_day - LAG(admissions_day, 1) OVER (ORDER BY admission_date)
AS    change_from_previous_day
FROM daily_admissions;
```

10. **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 0
 ELSE 1
 END,
 province_name;
```

11. **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) as doctor_name,
d.specialty as specialty,
year(a.admission_date) as year,
count(a.patient_id) as admissions
from admissions a
join doctors d
on a.attending_doctor_id= d.doctor_id
group by 1,2,3,4
order by 1,4,5 desc;
```

Conclusion:

This project helped me sharpen my SQL skills across real-world hospital data scenarios. From writing basic SELECT queries to tackling complex logic using CTEs and analytical functions, this was a great deep-dive into relational databases.

If you're hiring for data roles or working on a project where SQL skills are essential, I'd love to connect and contribute!

Email: souvikroy362@gmail.com
Linkedin: https://www.linkedin.com/in/souvik-roy-89964591/