Healthcare Analytics with SQL

Analyst : Prasanna Kumar

Project Needs and Findings

- Analyze healthcare data
- Focusing on extracting meaningful insights abou
  - patients, doctors,
  - appointments,
  - diagnoses, and
  - treatments using advanced SQL technique
- Approach:
  - All types of JOIN function in SQL.
  - Combining aggregate functions and condit

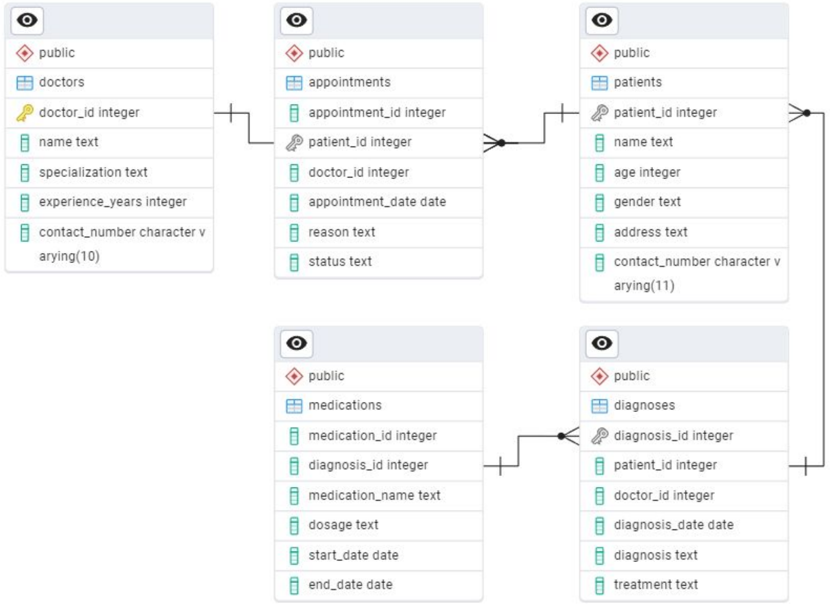
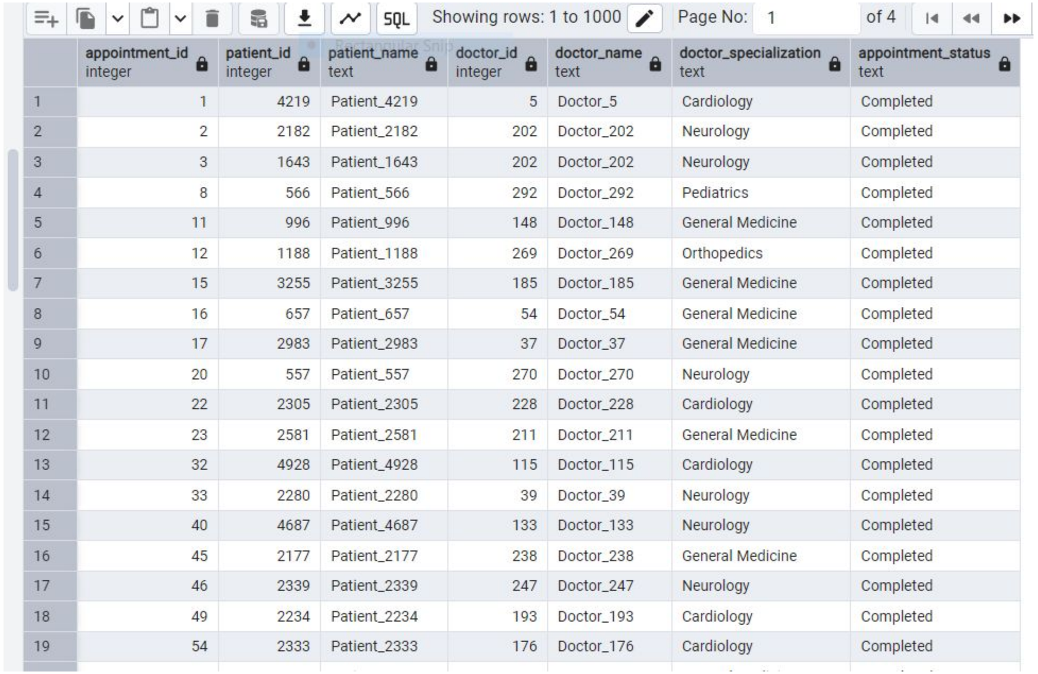


Fig 1.1: ER Diagram

Inner and Equi Joins

- Write a query to fetch details of all completed appointments, including the patient's name, doctor's name, and specialization.

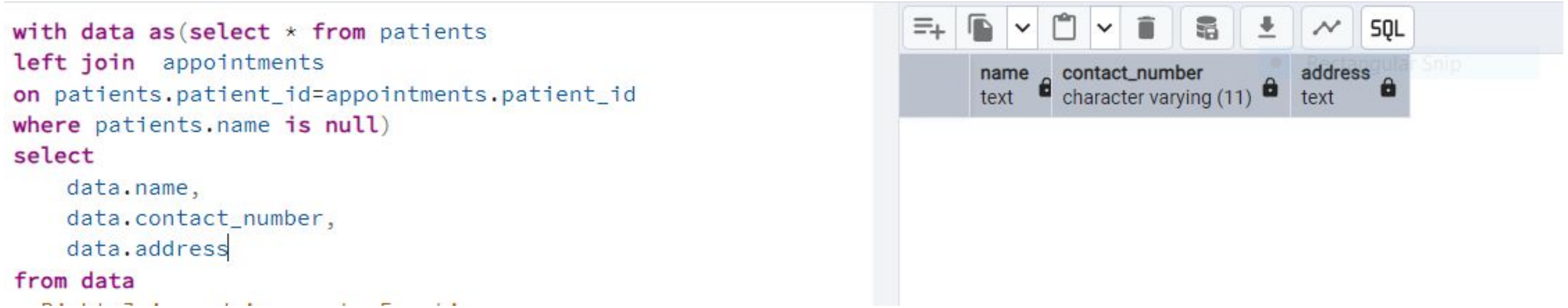
```sql
with  Doctor_detail as(
select appointments.appointment_id,
       appointments.patient_id,
       doctors.doctor_id,
       doctors.name as doctor_name,
       doctors.specialization as doctor_specialization,
       appointments.status as appointment_status
from appointments
inner join doctors
on appointments.doctor_id=doctors.doctor_id
where appointments.status = 'Completed'
)
Select
       Doctor_detail.appointment_id,
       Doctor_detail.patient_id,
       patients.name as Patient_name,
       Doctor_detail.doctor_id,
       Doctor_detail.doctor_name,
       Doctor_detail.doctor_specialization,
       Doctor_detail.appointment_status
from
       Doctor_detail
inner join
       patients
on
       Doctor_detail.patient_id=patients.patient_id;
```



	appointment_id integer	patient_id integer	patient_name text	doctor_id integer	doctor_name text	doctor_specialization text	appointment_status text
1	1	4219	Patient_4219	5	Doctor_5	Cardiology	Completed
2	2	2182	Patient_2182	202	Doctor_202	Neurology	Completed
3	3	1643	Patient_1643	202	Doctor_202	Neurology	Completed
4	8	566	Patient_566	292	Doctor_292	Pediatrics	Completed
5	11	996	Patient_996	148	Doctor_148	General Medicine	Completed
6	12	1188	Patient_1188	269	Doctor_269	Orthopedics	Completed
7	15	3255	Patient_3255	185	Doctor_185	General Medicine	Completed
8	16	657	Patient_657	54	Doctor_54	General Medicine	Completed
9	17	2983	Patient_2983	37	Doctor_37	General Medicine	Completed
10	20	557	Patient_557	270	Doctor_270	Neurology	Completed
11	22	2305	Patient_2305	228	Doctor_228	Cardiology	Completed
12	23	2581	Patient_2581	211	Doctor_211	General Medicine	Completed
13	32	4928	Patient_4928	115	Doctor_115	Cardiology	Completed
14	33	2280	Patient_2280	39	Doctor_39	Neurology	Completed
15	40	4687	Patient_4687	133	Doctor_133	Neurology	Completed
16	45	2177	Patient_2177	238	Doctor_238	General Medicine	Completed
17	46	2339	Patient_2339	247	Doctor_247	Neurology	Completed
18	49	2234	Patient_2234	193	Doctor_193	Cardiology	Completed
19	54	2333	Patient_2333	176	Doctor_176	Cardiology	Completed

Left Join with Null Handling

- **Task:** Retrieve all patients who have never had an appointment. Include their name, contact details, and address in the output.

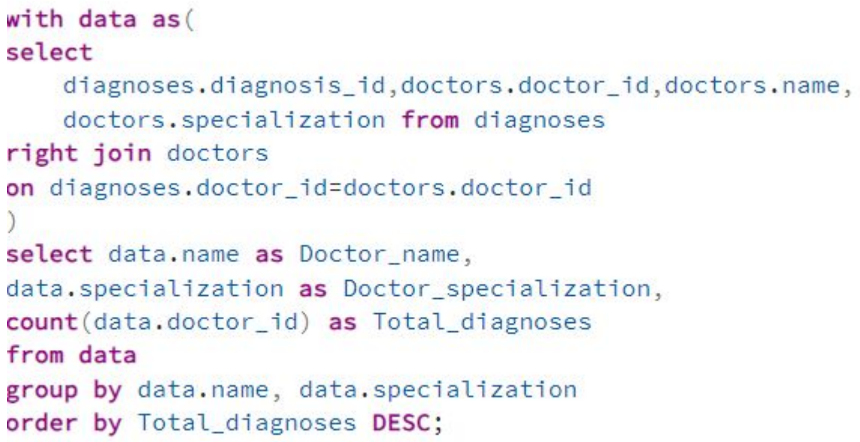


```
with data as(select * from patients
left join  appointments
on patients.patient_id=appointments.patient_id
where patients.name is null)
select
    data.name,
    data.contact_number,
    data.address
from data
```

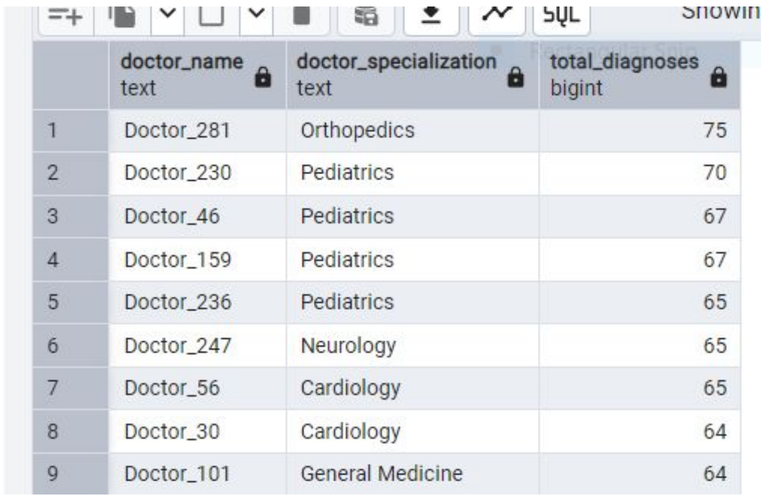
```
name    contact_number            address
text    character varying (11)    text
```

Right Join and Aggregate Functions

- **Task:** Find the total number of diagnoses for each doctor, including doctors who haven't diagnosed any patients. Display the doctor's name, specialization, and total diagnoses.



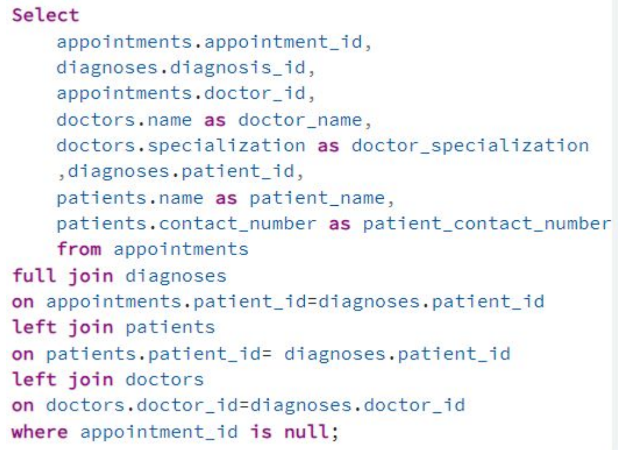
```
with data as(
select
    diagnoses.diagnosis_id,doctors.doctor_id,doctors.name,
    doctors.specialization from diagnoses
right join doctors
on diagnoses.doctor_id=doctors.doctor_id
)
select data.name as Doctor_name,
data.specialization as Doctor_specialization,
count(data.doctor_id) as Total_diagnoses
from data
group by data.name, data.specialization
order by Total_diagnoses DESC;
```



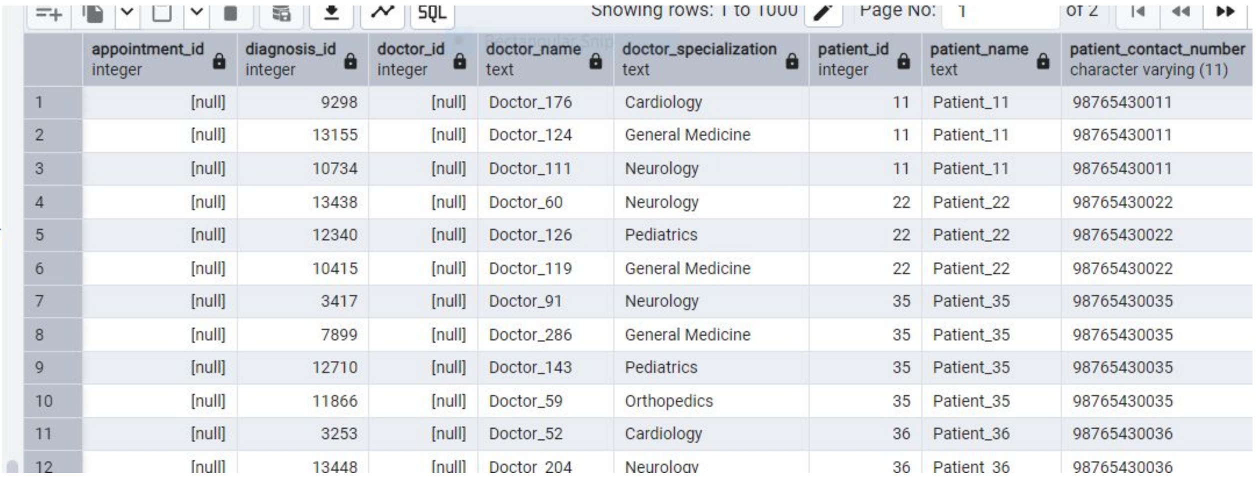
	doctor_name text	doctor_specialization text	total_diagnoses bigint
1	Doctor_281	Orthopedics	75
2	Doctor_230	Pediatrics	70
3	Doctor_46	Pediatrics	67
4	Doctor_159	Pediatrics	67
5	Doctor_236	Pediatrics	65
6	Doctor_247	Neurology	65
7	Doctor_56	Cardiology	65
8	Doctor_30	Cardiology	64
9	Doctor_101	General Medicine	64

Full Join for Overlapping Data

- **Task:** Write a query to identify mismatches between the appointments and diagnoses tables. Include all appointments and diagnoses with their corresponding patient and doctor details.



```
Select
    appointments.appointment_id,
    diagnoses.diagnosis_id,
    appointments.doctor_id,
    doctors.name as doctor_name,
    doctors.specialization as doctor_specialization
    ,diagnoses.patient_id,
    patients.name as patient_name,
    patients.contact_number as patient_contact_number
    from appointments
full join diagnoses
on appointments.patient_id=diagnoses.patient_id
left join patients
on patients.patient_id= diagnoses.patient_id
left join doctors
on doctors.doctor_id=diagnoses.doctor_id
where appointment_id is null;
```

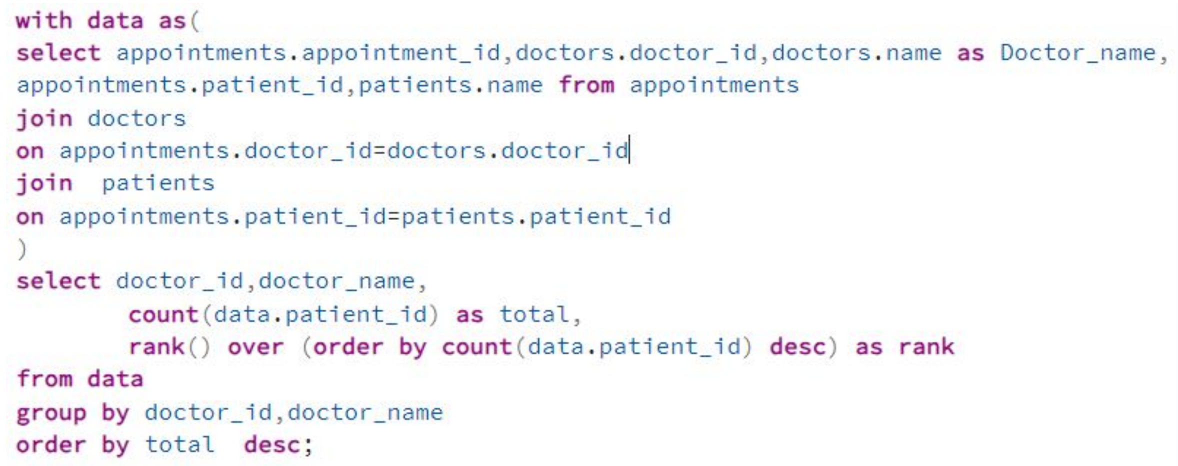


Showing rows: 1 to 1000 Page No: 1 of 2

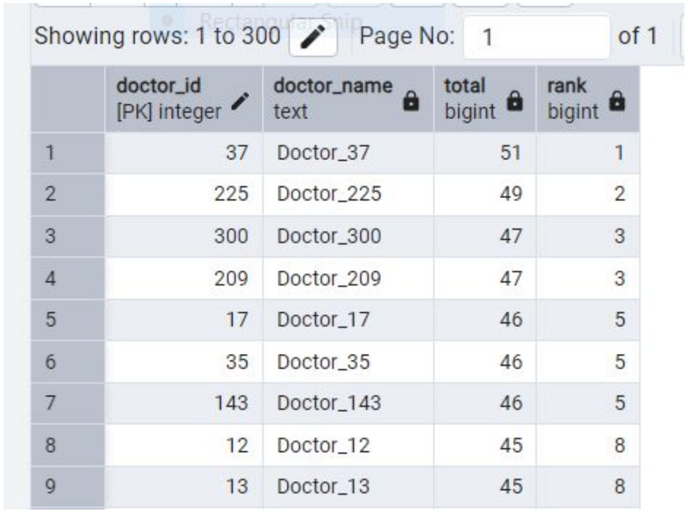
	appointment_id integer	diagnosis_id integer	doctor_id integer	doctor_name text	doctor_specialization text	patient_id integer	patient_name text	patient_contact_number character varying (11)
1	[null]	9298	[null]	Doctor_176	Cardiology	11	Patient_11	98765430011
2	[null]	13155	[null]	Doctor_124	General Medicine	11	Patient_11	98765430011
3	[null]	10734	[null]	Doctor_111	Neurology	11	Patient_11	98765430011
4	[null]	13438	[null]	Doctor_60	Neurology	22	Patient_22	98765430022
5	[null]	12340	[null]	Doctor_126	Pediatrics	22	Patient_22	98765430022
6	[null]	10415	[null]	Doctor_119	General Medicine	22	Patient_22	98765430022
7	[null]	3417	[null]	Doctor_91	Neurology	35	Patient_35	98765430035
8	[null]	7899	[null]	Doctor_286	General Medicine	35	Patient_35	98765430035
9	[null]	12710	[null]	Doctor_143	Pediatrics	35	Patient_35	98765430035
10	[null]	11866	[null]	Doctor_59	Orthopedics	35	Patient_35	98765430035
11	[null]	3253	[null]	Doctor_52	Cardiology	36	Patient_36	98765430036
12	[null]	13448	[null]	Doctor_204	Neurology	36	Patient_36	98765430036

Window Functions (Ranking and Aggregation)

- **Task:** For each doctor, rank their patients based on the number of appointments in descending order.



```sql
with data as(
select appointments.appointment_id,doctors.doctor_id,doctors.name as Doctor_name,
appointments.patient_id,patients.name from appointments
join doctors
on appointments.doctor_id=doctors.doctor_id
join  patients
on appointments.patient_id=patients.patient_id
)
select doctor_id,doctor_name,
        count(data.patient_id) as total,
        rank() over (order by count(data.patient_id) desc) as rank
from data
group by doctor_id,doctor_name
order by total  desc;
```

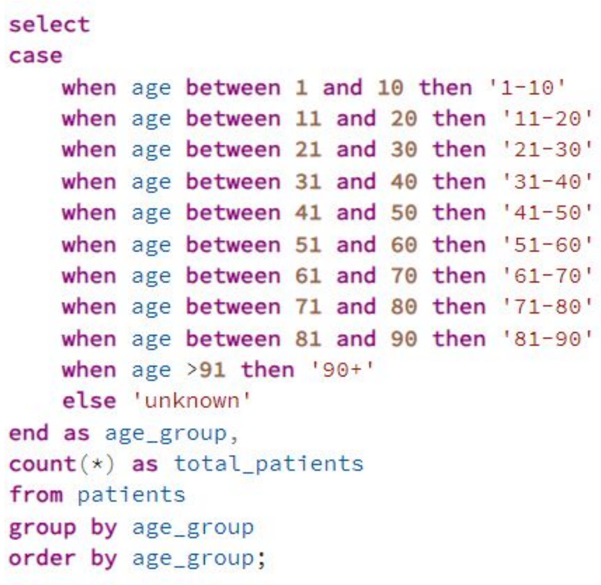


Showing rows: 1 to 300 Page No: 1 of 1

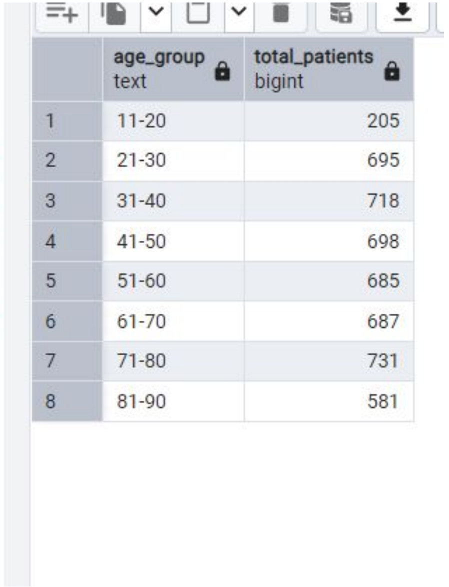
	doctor_id [PK] integer	doctor_name text	total bigint	rank bigint
1	37	Doctor_37	51	1
2	225	Doctor_225	49	2
3	300	Doctor_300	47	3
4	209	Doctor_209	47	3
5	17	Doctor_17	46	5
6	35	Doctor_35	46	5
7	143	Doctor_143	46	5
8	12	Doctor_12	45	8
9	13	Doctor_13	45	8

Conditional Expressions

- **Task:** Write a query to categorize patients by age group (e.g., 18-30, 31-50, 51+). Count the number of patients in each age group.



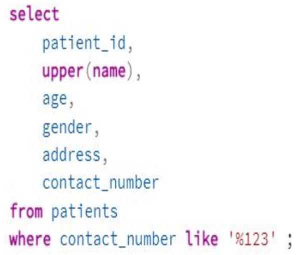
```
select
case
    when age between 1 and 10 then '1-10'
    when age between 11 and 20 then '11-20'
    when age between 21 and 30 then '21-30'
    when age between 31 and 40 then '31-40'
    when age between 41 and 50 then '41-50'
    when age between 51 and 60 then '51-60'
    when age between 61 and 70 then '61-70'
    when age between 71 and 80 then '71-80'
    when age between 81 and 90 then '81-90'
    when age >91 then '90+'
    else 'unknown'
end as age_group,
count(*) as total_patients
from patients
group by age_group
order by age_group;
```



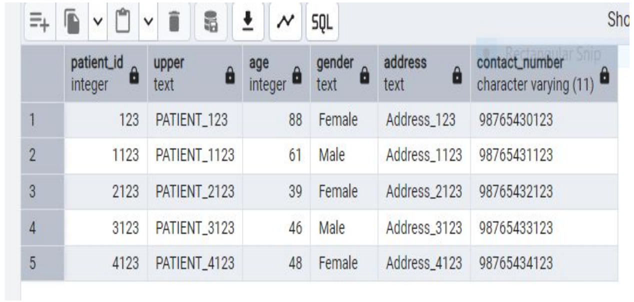
	age_group text	total_patients bigint
1	11-20	205
2	21-30	695
3	31-40	718
4	41-50	698
5	51-60	685
6	61-70	687
7	71-80	731
8	81-90	581

Numeric and String Functions

- **Task:** Retrieve a list of patients whose contact numbers end with "1234" and display their names in uppercase.



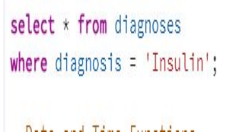
```sql
select
    patient_id,
    upper(name),
    age,
    gender,
    address,
    contact_number
from patients
where contact_number like '%123' ;
```



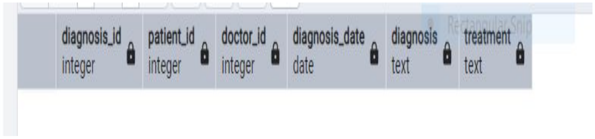
	patient_id integer	upper text	age integer	gender text	address text	contact_number character varying (11)
1	123	PATIENT_123	88	Female	Address_123	98765430123
2	1123	PATIENT_1123	61	Male	Address_1123	98765431123
3	2123	PATIENT_2123	39	Female	Address_2123	98765432123
4	3123	PATIENT_3123	46	Male	Address_3123	98765433123
5	4123	PATIENT_4123	48	Female	Address_4123	98765434123

Subqueries for Filtering

- **Task:** Find patients who have only been prescribed "Insulin" in any of their diagnoses.



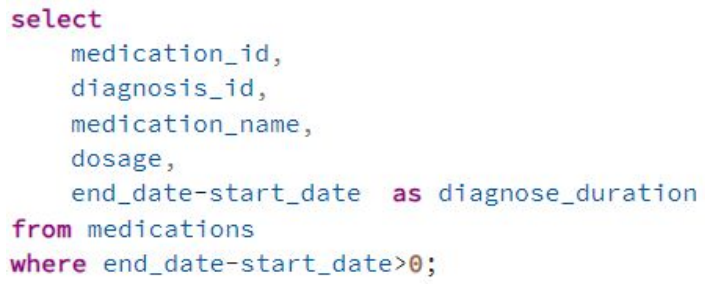
```sql
select * from diagnoses
where diagnosis = 'Insulin';
```



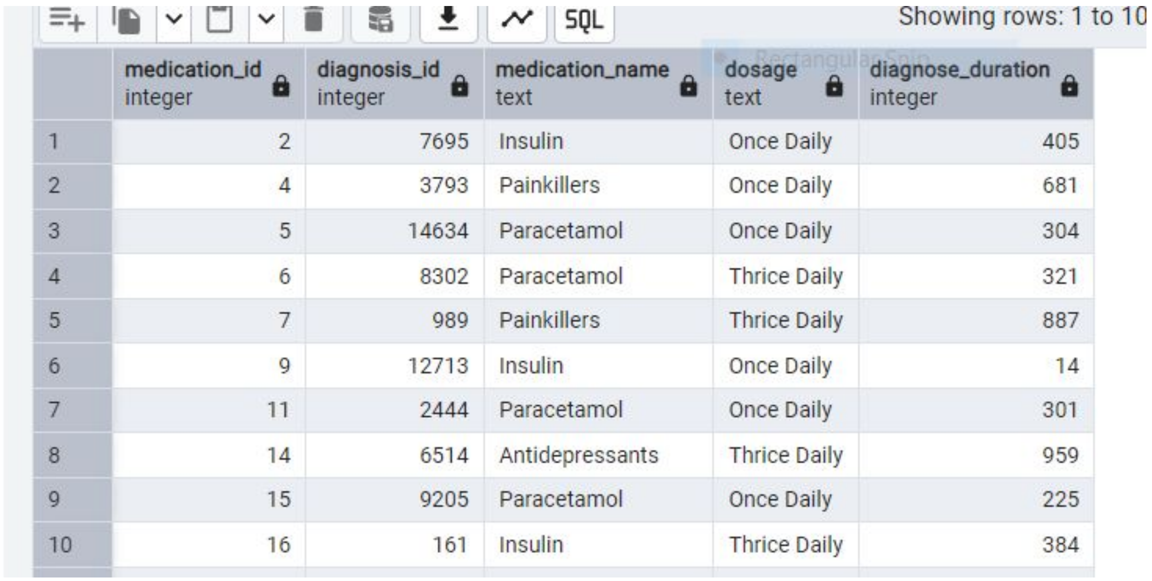
	diagnosis_id integer	patient_id integer	doctor_id integer	diagnosis_date date	diagnosis text	treatment text

Date and Time Functions

- **Task:** Calculate the average duration (in days) for which medications are prescribed for each diagnosis.



```
select
    medication_id,
    diagnosis_id,
    medication_name,
    dosage,
    end_date-start_date  as diagnose_duration
from medications
where end_date-start_date>0;
```



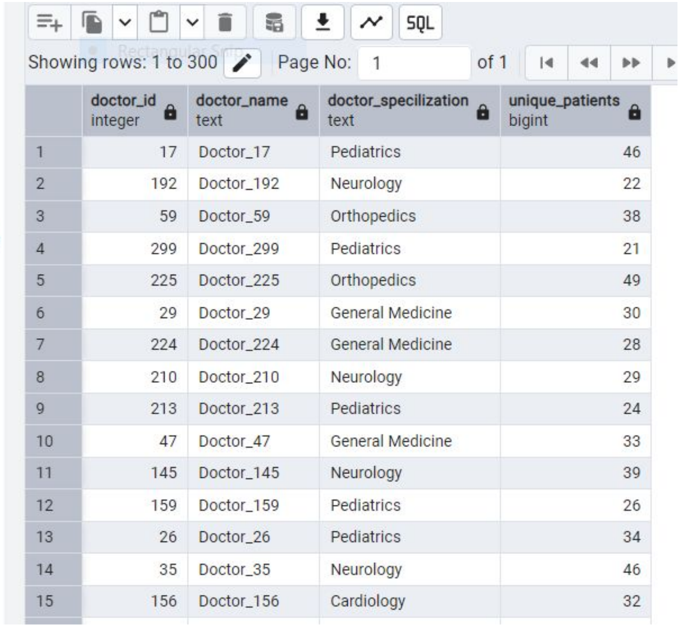
Showing rows: 1 to 10

	medication_id integer	diagnosis_id integer	medication_name text	dosage text	diagnose_duration integer
1	2	7695	Insulin	Once Daily	405
2	4	3793	Painkillers	Once Daily	681
3	5	14634	Paracetamol	Once Daily	304
4	6	8302	Paracetamol	Thrice Daily	321
5	7	989	Painkillers	Thrice Daily	887
6	9	12713	Insulin	Once Daily	14
7	11	2444	Paracetamol	Once Daily	301
8	14	6514	Antidepressants	Thrice Daily	959
9	15	9205	Paracetamol	Once Daily	225
10	16	161	Insulin	Thrice Daily	384

Complex Joins and Aggregation

- **Task:** Write a query to identify the doctor who has attended the most unique patients. Include the doctor's name, specialization, and the count of unique patients.

```
with data as(
select
    appointments.doctor_id,
    doctors.name as doctor_name,doctors.specialization as doctor_specilization,
    appointments.patient_id, patients.name from appointments
join doctors
on appointments.doctor_id=doctors.doctor_id
join patients
on appointments.patient_id=patients.patient_id
GROUP BY patients.name, appointments.doctor_id, doctors.name, doctors.specialization,
    appointments.patient_id)
select
    data.doctor_id, data.Doctor_name,
    data.doctor_specilization ,
    count(data.patient_id) as Unique_patients
from
    data
group by
    data.doctor_id, data.doctor_name,
    data.doctor_specilization ;
```



Showing rows: 1 to 300 Page No: 1 of 1

	doctor_id integer	doctor_name text	doctor_specilization text	unique_patients bigint
1	17	Doctor_17	Pediatrics	46
2	192	Doctor_192	Neurology	22
3	59	Doctor_59	Orthopedics	38
4	299	Doctor_299	Pediatrics	21
5	225	Doctor_225	Orthopedics	49
6	29	Doctor_29	General Medicine	30
7	224	Doctor_224	General Medicine	28
8	210	Doctor_210	Neurology	29
9	213	Doctor_213	Pediatrics	24
10	47	Doctor_47	General Medicine	33
11	145	Doctor_145	Neurology	39
12	159	Doctor_159	Pediatrics	26
13	26	Doctor_26	Pediatrics	34
14	35	Doctor_35	Neurology	46
15	156	Doctor_156	Cardiology	32