SQL Joins

SQL Joins Overview

What are SQL Joins?

- Operations that combine rows from two or more tables
- Based on related columns between tables
- Essential for retrieving data from normalized databases

Table Aliases

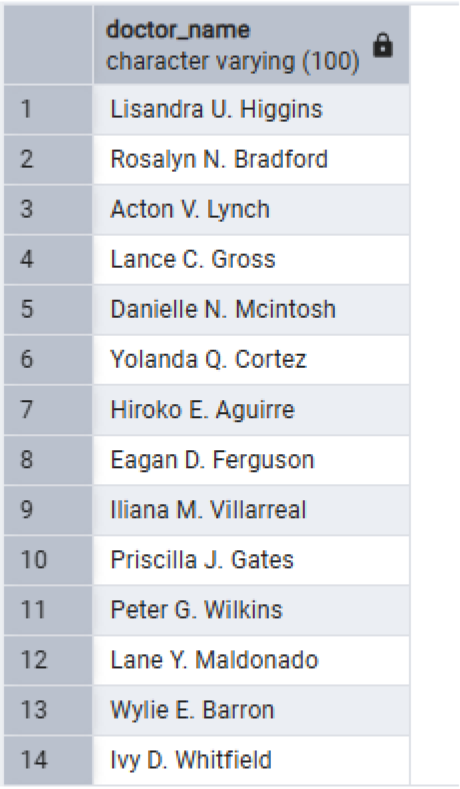
- SQL aliases are used to give a table, or a column in a table, a temporary name.
- Aliases are often used to make column names more readable.
- An alias only exists for the duration of that query.
- An alias is created with the AS keyword.

Syntax:

```sql
SELECT doc_name AS doctor_name
FROM doctors;
```

We can also skip the AS keyword and get the same result:

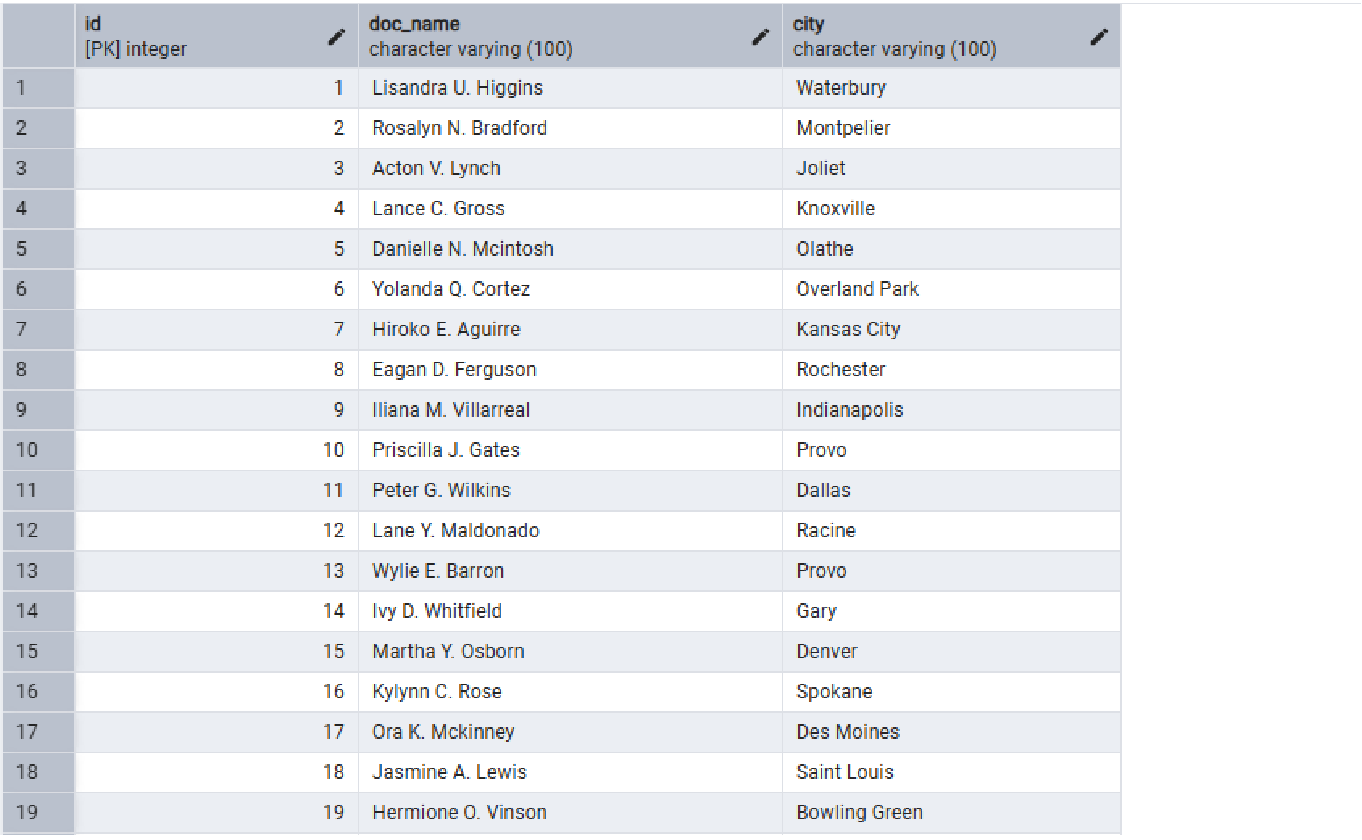
```sql
SELECT doc_name doctor_name FROM doctors
```



	doctor_name character varying (100)
1	Lisandra U. Higgins
2	Rosalyn N. Bradford
3	Acton V. Lynch
4	Lance C. Gross
5	Danielle N. Mcintosh
6	Yolanda Q. Cortez
7	Hiroko E. Aguirre
8	Eagan D. Ferguson
9	Iliana M. Villarreal
10	Priscilla J. Gates
11	Peter G. Wilkins
12	Lane Y. Maldonado
13	Wylie E. Barron
14	Ivy D. Whitfield

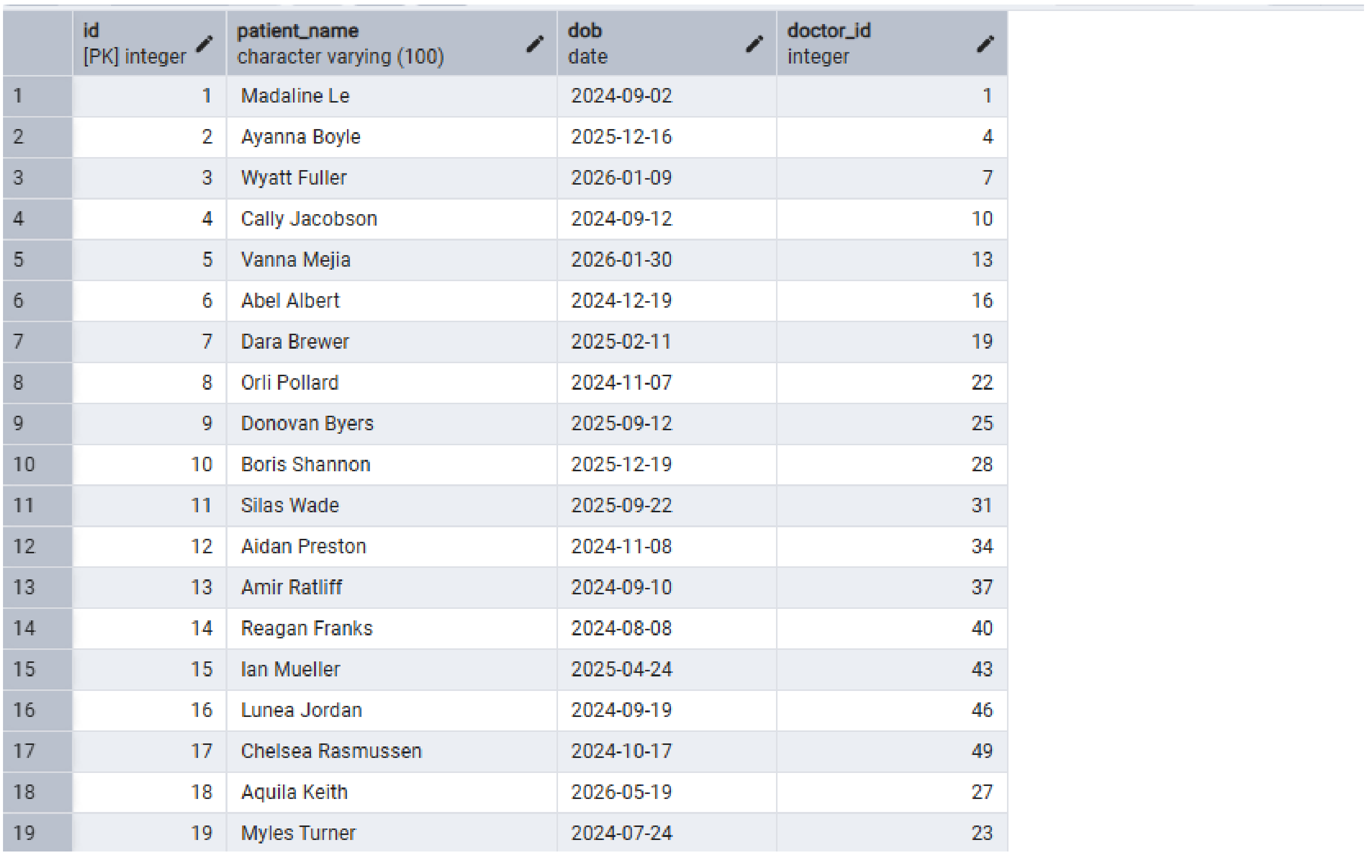
Sample Data

DOCTORS



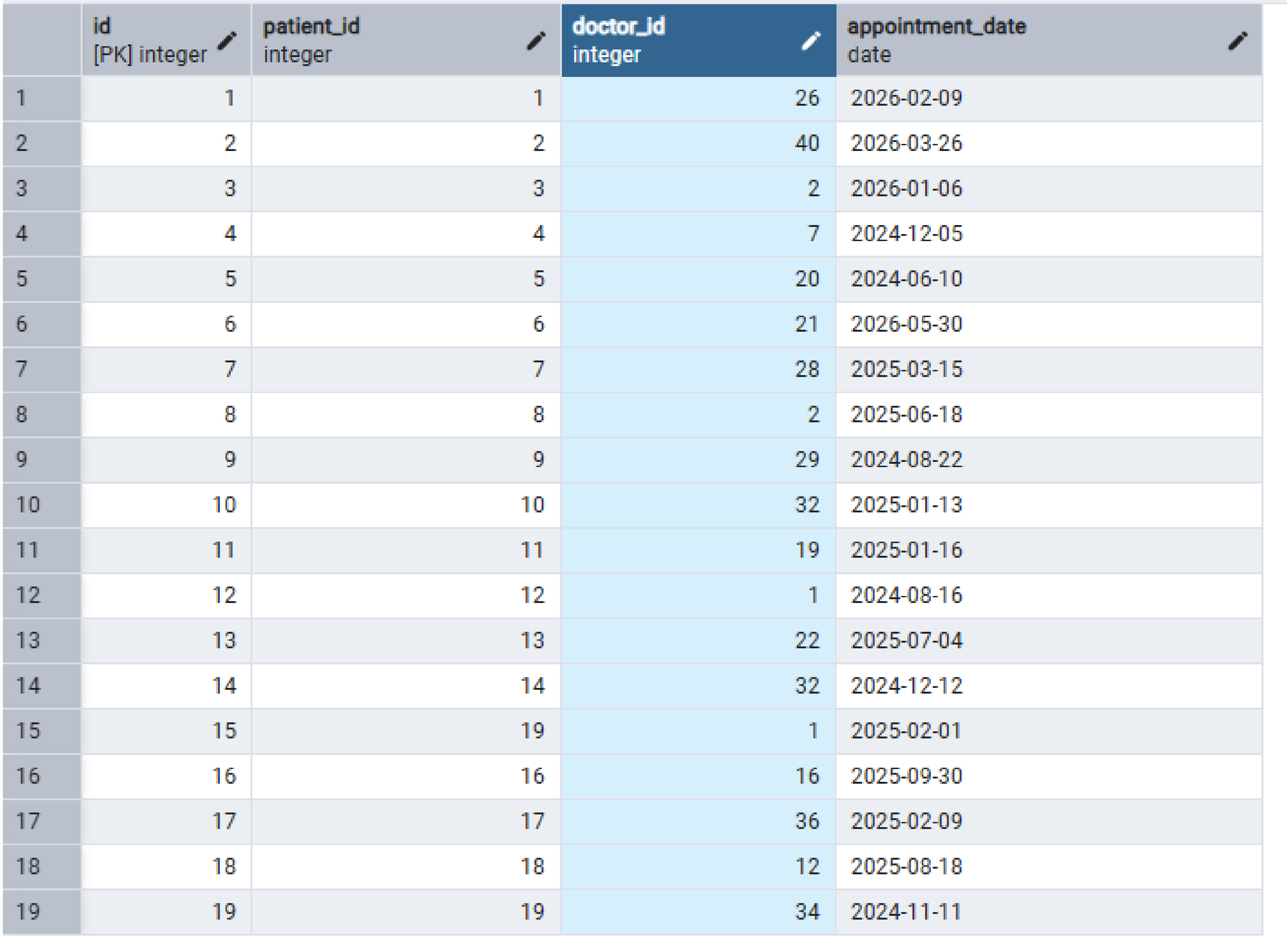
	id [PK] integer	doc_name character varying (100)	city character varying (100)
1	1	Lisandra U. Higgins	Waterbury
2	2	Rosalyn N. Bradford	Montpelier
3	3	Acton V. Lynch	Joliet
4	4	Lance C. Gross	Knoxville
5	5	Danielle N. Mcintosh	Olathe
6	6	Yolanda Q. Cortez	Overland Park
7	7	Hiroko E. Aguirre	Kansas City
8	8	Eagan D. Ferguson	Rochester
9	9	Iliana M. Villarreal	Indianapolis
10	10	Priscilla J. Gates	Provo
11	11	Peter G. Wilkins	Dallas
12	12	Lane Y. Maldonado	Racine
13	13	Wylie E. Barron	Provo
14	14	Ivy D. Whitfield	Gary
15	15	Martha Y. Osborn	Denver
16	16	Kylynn C. Rose	Spokane
17	17	Ora K. Mckinney	Des Moines
18	18	Jasmine A. Lewis	Saint Louis
19	19	Hermione O. Vinson	Bowling Green

PATIENTS



	id [PK] integer	patient_name character varying (100)	dob date	doctor_id integer
1	1	Madaline Le	2024-09-02	1
2	2	Ayanna Boyle	2025-12-16	4
3	3	Wyatt Fuller	2026-01-09	7
4	4	Cally Jacobson	2024-09-12	10
5	5	Vanna Mejia	2026-01-30	13
6	6	Abel Albert	2024-12-19	16
7	7	Dara Brewer	2025-02-11	19
8	8	Orli Pollard	2024-11-07	22
9	9	Donovan Byers	2025-09-12	25
10	10	Boris Shannon	2025-12-19	28
11	11	Silas Wade	2025-09-22	31
12	12	Aidan Preston	2024-11-08	34
13	13	Amir Ratliff	2024-09-10	37
14	14	Reagan Franks	2024-08-08	40
15	15	Ian Mueller	2025-04-24	43
16	16	Lunea Jordan	2024-09-19	46
17	17	Chelsea Rasmussen	2024-10-17	49
18	18	Aquila Keith	2026-05-19	27
19	19	Myles Turner	2024-07-24	23

APPOINTMENTS



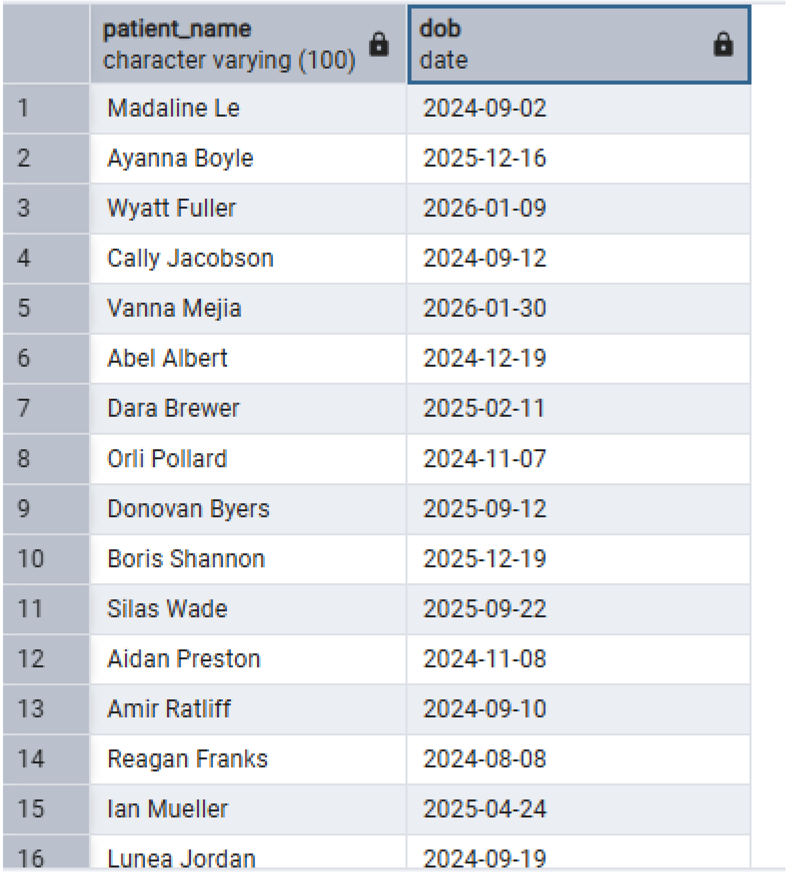
	id [PK] integer	patient_id integer	doctor_id integer	appointment_date date
1	1	1	26	2026-02-09
2	2	2	40	2026-03-26
3	3	3	2	2026-01-06
4	4	4	7	2024-12-05
5	5	5	20	2024-06-10
6	6	6	21	2026-05-30
7	7	7	28	2025-03-15
8	8	8	2	2025-06-18
9	9	9	29	2024-08-22
10	10	10	32	2025-01-13
11	11	11	19	2025-01-16
12	12	12	1	2024-08-16
13	13	13	22	2025-07-04
14	14	14	32	2024-12-12
15	15	19	1	2025-02-01
16	16	16	16	2025-09-30
17	17	17	36	2025-02-09
18	18	18	12	2025-08-18
19	19	19	34	2024-11-11

INNER JOIN

Returns: Only matching rows from both tables

```sql
SELECT patient_name, dob
FROM patients
INNER JOIN doctors on patients.doctor_id = doctors.id;
```

Result:



	patient_name character varying (100)	dob date
1	Madaline Le	2024-09-02
2	Ayanna Boyle	2025-12-16
3	Wyatt Fuller	2026-01-09
4	Cally Jacobson	2024-09-12
5	Vanna Mejia	2026-01-30
6	Abel Albert	2024-12-19
7	Dara Brewer	2025-02-11
8	Orli Pollard	2024-11-07
9	Donovan Byers	2025-09-12
10	Boris Shannon	2025-12-19
11	Silas Wade	2025-09-22
12	Aidan Preston	2024-11-08
13	Amir Ratliff	2024-09-10
14	Reagan Franks	2024-08-08
15	Ian Mueller	2025-04-24
16	Lunea Jordan	2024-09-19

This shows patients that have doctors assigned to them.

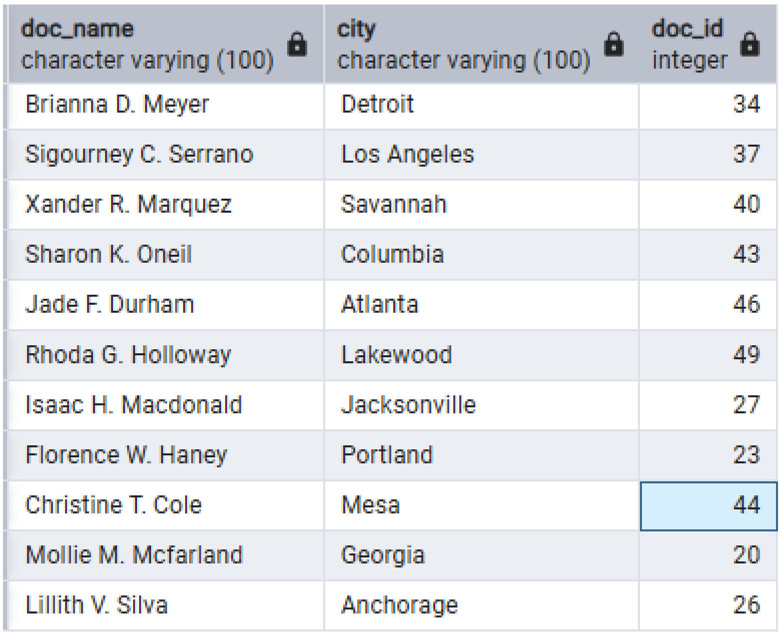
INNER is the default join type for JOIN, so when you write JOIN the parser actually writes INNER JOIN.

LEFT JOIN

Returns: All rows from left table + matching rows from right

```sql
SELECT patient_name, dob, patients.id as p_id
FROM patients
LEFT JOIN doctors on patients.doctor_id = doctors.id;
```

Result:



doc_name character varying (100)	city character varying (100)	doc_id integer
Brianna D. Meyer	Detroit	34
Sigourney C. Serrano	Los Angeles	37
Xander R. Marquez	Savannah	40
Sharon K. Oneil	Columbia	43
Jade F. Durham	Atlanta	46
Rhoda G. Holloway	Lakewood	49
Isaac H. Macdonald	Jacksonville	27
Florence W. Haney	Portland	23
Christine T. Cole	Mesa	44
Mollie M. Mcfarland	Georgia	20
Lillith V. Silva	Anchorage	26

All doctors included, even without assigned patients

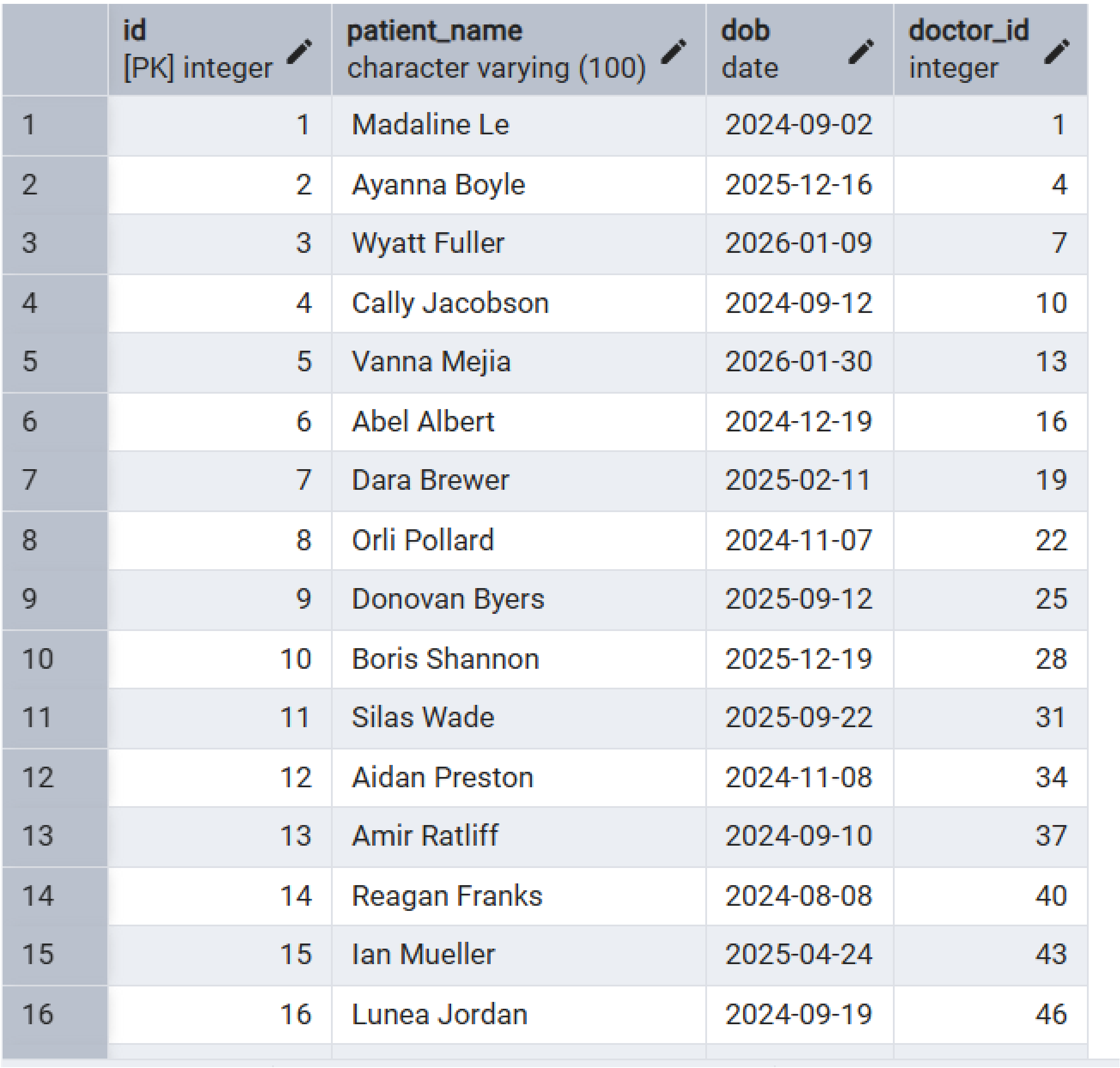
Example 2

Using the current hospital_db:

Running

```
SELECT * FROM patients;
```

Fetches the data from `patients` table resulting to:

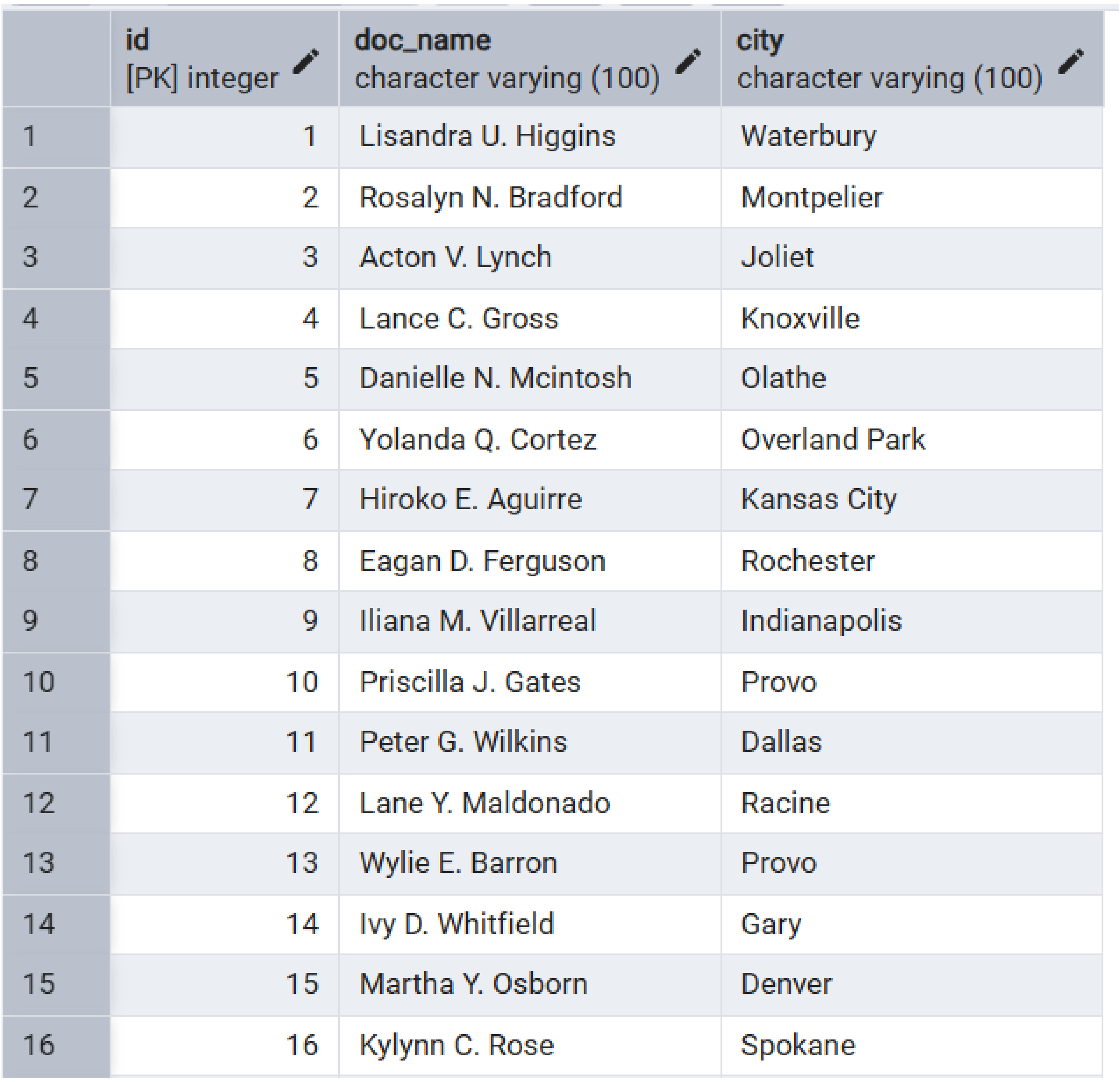


	id	patient_name	dob	doctor_id
	[PK] integer	character varying (100)	date	integer
1	1	Madaline Le	2024-09-02	1
2	2	Ayanna Boyle	2025-12-16	4
3	3	Wyatt Fuller	2026-01-09	7
4	4	Cally Jacobson	2024-09-12	10
5	5	Vanna Mejia	2026-01-30	13
6	6	Abel Albert	2024-12-19	16
7	7	Dara Brewer	2025-02-11	19
8	8	Orli Pollard	2024-11-07	22
9	9	Donovan Byers	2025-09-12	25
10	10	Boris Shannon	2025-12-19	28
11	11	Silas Wade	2025-09-22	31
12	12	Aidan Preston	2024-11-08	34
13	13	Amir Ratliff	2024-09-10	37
14	14	Reagan Franks	2024-08-08	40
15	15	Ian Mueller	2025-04-24	43
16	16	Lunea Jordan	2024-09-19	46

Running

```sql
SELECT * FROM doctors;
```

Fetches the data from `doctors` table resulting to:

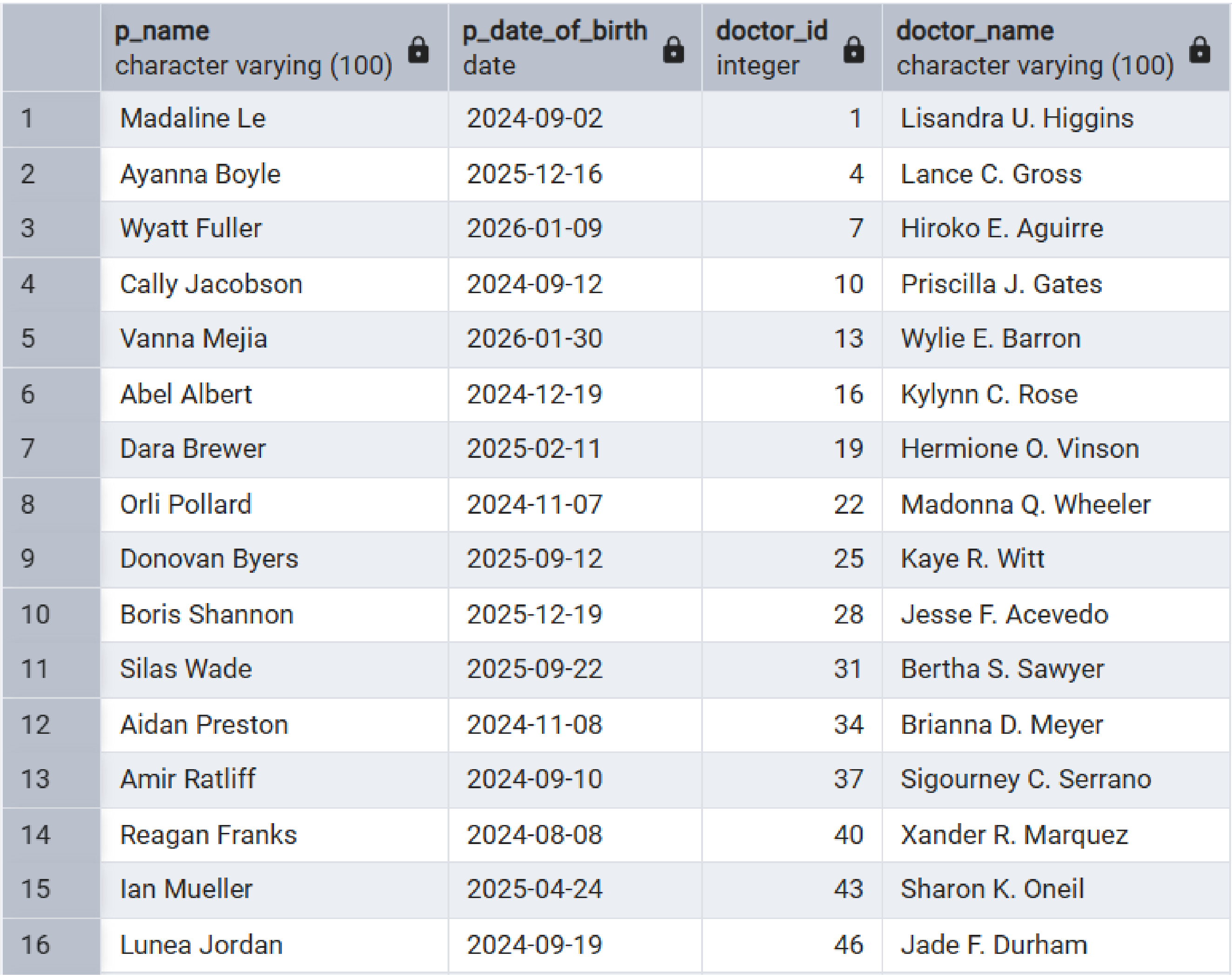


	id [PK] integer	doc_name character varying (100)	city character varying (100)
1	1	Lisandra U. Higgins	Waterbury
2	2	Rosalyn N. Bradford	Montpelier
3	3	Acton V. Lynch	Joliet
4	4	Lance C. Gross	Knoxville
5	5	Danielle N. Mcintosh	Olathe
6	6	Yolanda Q. Cortez	Overland Park
7	7	Hiroko E. Aguirre	Kansas City
8	8	Eagan D. Ferguson	Rochester
9	9	Iliana M. Villarreal	Indianapolis
10	10	Priscilla J. Gates	Provo
11	11	Peter G. Wilkins	Dallas
12	12	Lane Y. Maldonado	Racine
13	13	Wylie E. Barron	Provo
14	14	Ivy D. Whitfield	Gary
15	15	Martha Y. Osborn	Denver
16	16	Kylynn C. Rose	Spokane

PERFORMING a LEFT JOIN

```
SELECT p.patient_name AS P_Name,
p.dob P_Date_of_Birth,
d.id AS doctor_ID, d.doc_name Doctor_Name
FROM patients p
LEFT JOIN
doctors d
ON p.doctor_id = d.id;
```

Resulting to:



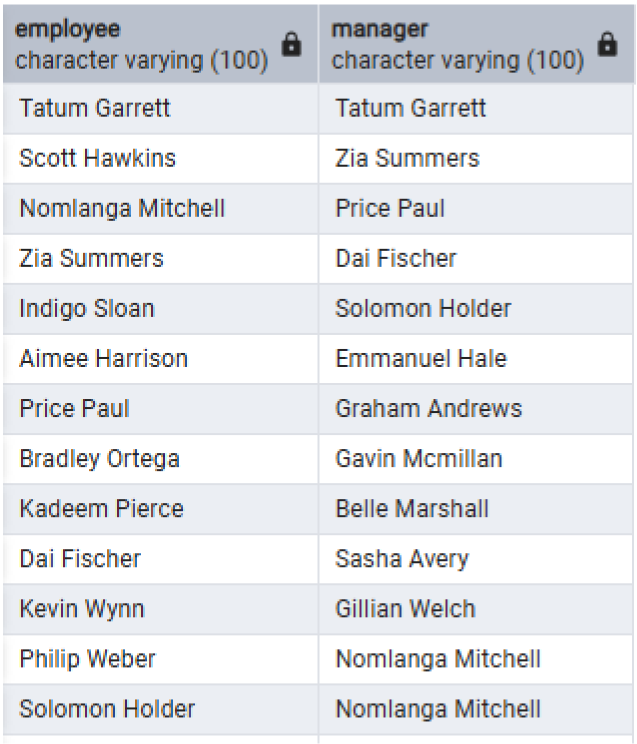
	p_name character varying (100)	p_date_of_birth date	doctor_id integer	doctor_name character varying (100)
1	Madaline Le	2024-09-02	1	Lisandra U. Higgins
2	Ayanna Boyle	2025-12-16	4	Lance C. Gross
3	Wyatt Fuller	2026-01-09	7	Hiroko E. Aguirre
4	Cally Jacobson	2024-09-12	10	Priscilla J. Gates
5	Vanna Mejia	2026-01-30	13	Wylie E. Barron
6	Abel Albert	2024-12-19	16	Kylynn C. Rose
7	Dara Brewer	2025-02-11	19	Hermione O. Vinson
8	Orli Pollard	2024-11-07	22	Madonna Q. Wheeler
9	Donovan Byers	2025-09-12	25	Kaye R. Witt
10	Boris Shannon	2025-12-19	28	Jesse F. Acevedo
11	Silas Wade	2025-09-22	31	Bertha S. Sawyer
12	Aidan Preston	2024-11-08	34	Brianna D. Meyer
13	Amir Ratliff	2024-09-10	37	Sigourney C. Serrano
14	Reagan Franks	2024-08-08	40	Xander R. Marquez
15	Ian Mueller	2025-04-24	43	Sharon K. Oneil
16	Lunea Jordan	2024-09-19	46	Jade F. Durham

SELF JOIN

Purpose: Table joins with itself (hierarchical data)

```
SELECT e.employee_name AS employee,
       m.employee_name AS manager
FROM employees AS e
LEFT JOIN employees AS m ON e.manager_id = m.id;
```

Sample EMPLOYEES data:



employee character varying (100)	manager character varying (100)
Tatum Garrett	Tatum Garrett
Scott Hawkins	Zia Summers
Nomlanga Mitchell	Price Paul
Zia Summers	Dai Fischer
Indigo Sloan	Solomon Holder
Aimee Harrison	Emmanuel Hale
Price Paul	Graham Andrews
Bradley Ortega	Gavin Mcmillan
Kadeem Pierce	Belle Marshall
Dai Fischer	Sasha Avery
Kevin Wynn	Gillian Welch
Philip Weber	Nomlanga Mitchell
Solomon Holder	Nomlanga Mitchell

Common Uses:

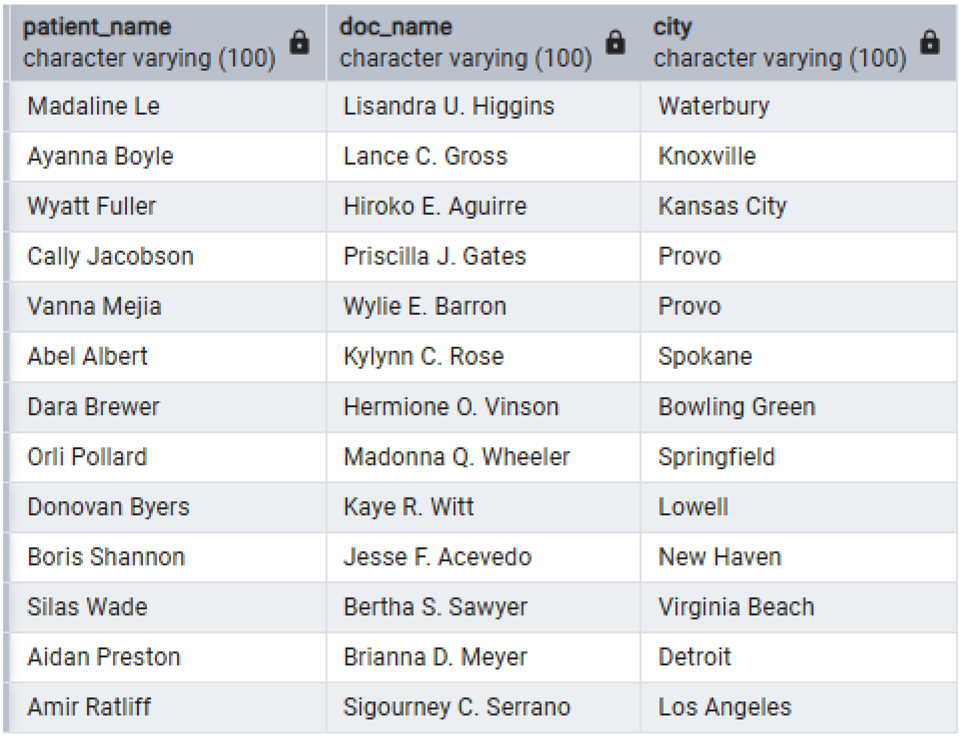
- Employee-manager relationships
- Doctor referral chains
- Patient family relationships

FULL OUTER JOIN

Returns: All rows from both tables

```
SELECT p.patient_name, d.doc_name, d.city
FROM patients AS p
FULL OUTER JOIN doctors AS d ON p.doctor_id = d.id;
```

Result:

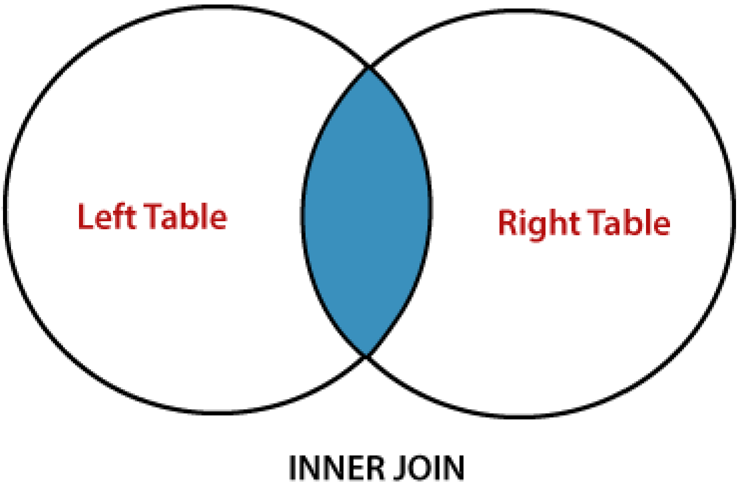


patient_name character varying (100)	doc_name character varying (100)	city character varying (100)
Madaline Le	Lisandra U. Higgins	Waterbury
Ayanna Boyle	Lance C. Gross	Knoxville
Wyatt Fuller	Hiroko E. Aguirre	Kansas City
Cally Jacobson	Priscilla J. Gates	Provo
Vanna Mejia	Wylie E. Barron	Provo
Abel Albert	Kylynn C. Rose	Spokane
Dara Brewer	Hermione O. Vinson	Bowling Green
Orli Pollard	Madonna Q. Wheeler	Springfield
Donovan Byers	Kaye R. Witt	Lowell
Boris Shannon	Jesse F. Acevedo	New Haven
Silas Wade	Bertha S. Sawyer	Virginia Beach
Aidan Preston	Brianna D. Meyer	Detroit
Amir Ratliff	Sigourney C. Serrano	Los Angeles

Everything from both tables - includes doctors with no patients

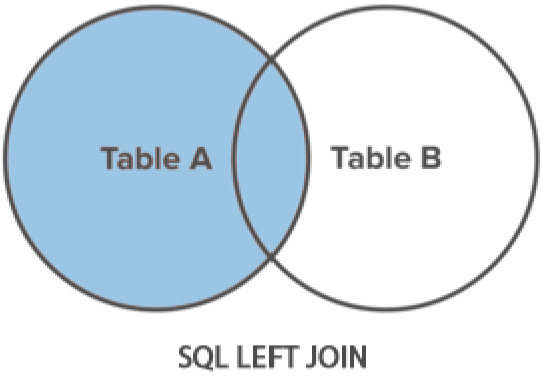
---

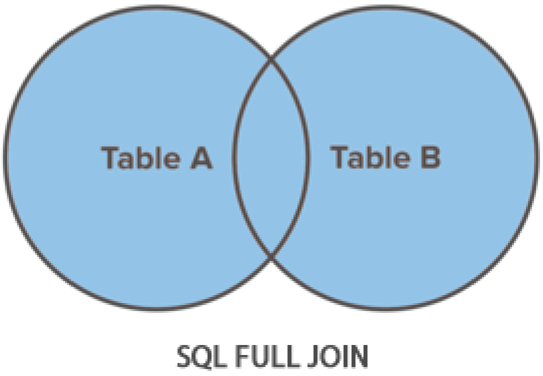
Visual Comparison



INNER JOIN: (Only overlap)

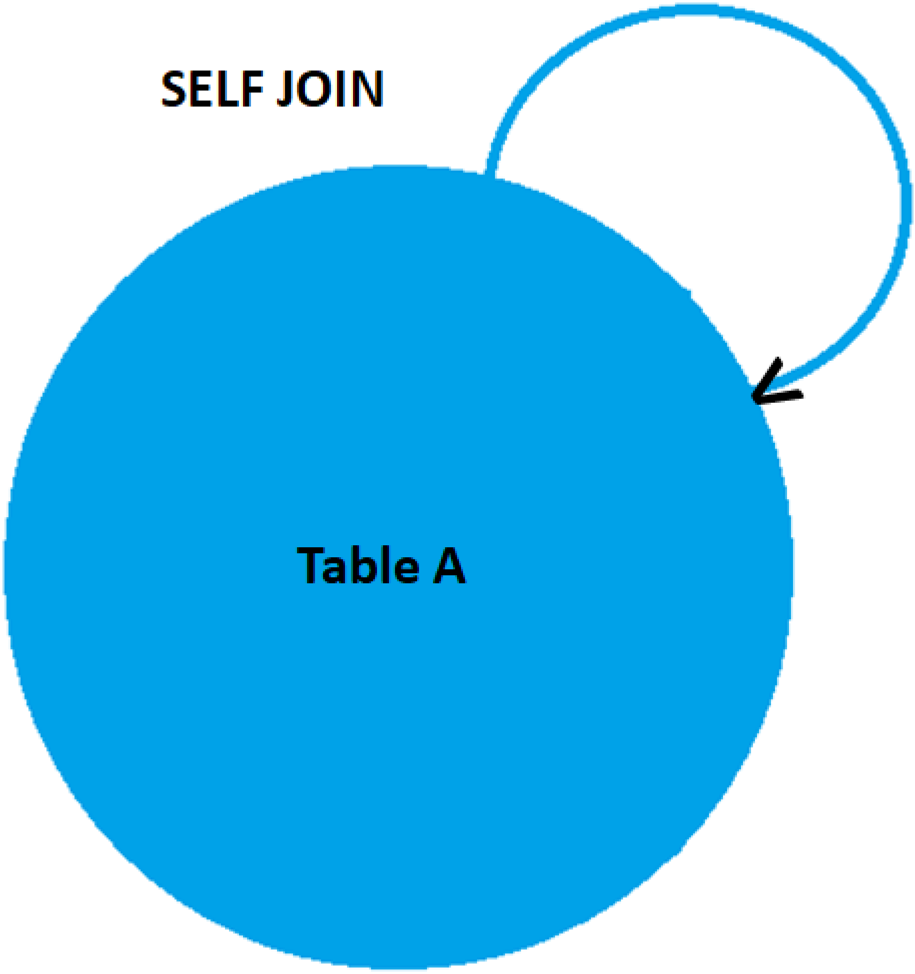
LEFT JOIN: ●◐ (All left + overlap)





FULL OUTER JOIN: ●○ (Everything)

SELF JOIN: ●⟲ (Table to itself)



Best Practices

Do:

- Use table aliases consistently
- Index JOIN columns
- Choose appropriate JOIN type
- Filter data early with WHERE

Avoid:

- Missing JOIN conditions
- Wrong JOIN type for requirements
- Ignoring NULL handling
- Poor column indexing

Real-World Applications

INNER JOIN: Active relationships only

- Patients with assigned doctors
- Appointments with valid patient-doctor pairs

LEFT JOIN: Include all from primary table

- All patients (with/without doctors)
- All appointments (even with missing data)

SELF JOIN: Hierarchical data

- Employee management structure
- Doctor referral networks
- Patient family relationships

FULL OUTER JOIN: Data reconciliation

- Patient-doctor assignment audits
- System data comparisons
- Complete database reviews

Key Takeaways

- **Table aliases** improve readability
- **INNER JOIN** = matches only
- **LEFT JOIN** = all from left + matches
- **SELF JOIN** = table references itself
- **FULL OUTER JOIN** = everything from both

Happy Coding Champ!