

SQL foundations

by Nicola Orecchini, 31/07/2025

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Agenda

Introduction to relational data model

SQL basic operations

Deep dive on JOIN

Agenda

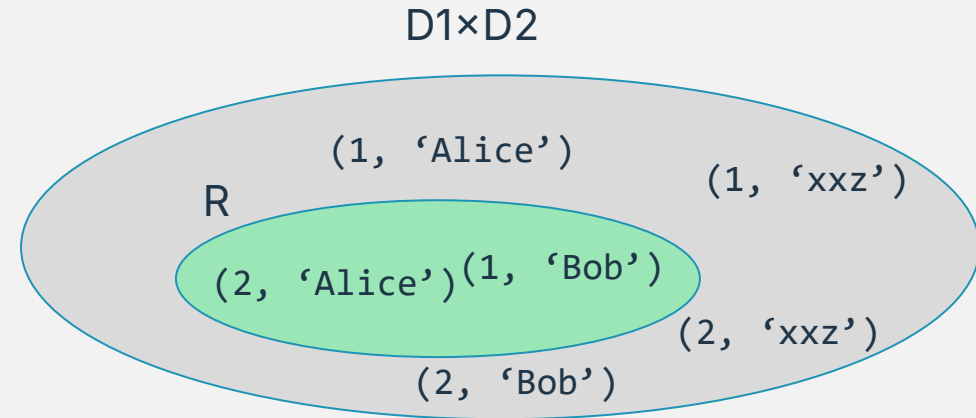
- Introduction to relational data model
- SQL basic operations
- Deep dive on JOIN

A relational database is a database that stores data in tables (called *relations*)



$D_1 \times D_2 \times \dots \times D_n$ are domains: sets of possible values for a variable. Each domain corresponds to a database column type, e.g.:

- $D_1 = \mathbb{Z}$ (integers for id)
- $D_2 = \text{strings}$ (for name)



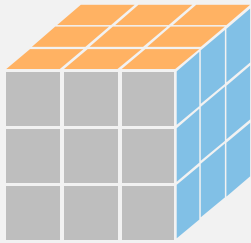
A relation R is a subset of the cartesian product $D_1 \times D_2 \times \dots \times D_n$:
 $R \subseteq D_1 \times D_2 \times \dots \times D_n$

Each element of R is a n -tuple (d_1, d_2, \dots, d_n) where $d_i \in D_i$.

So, a relation corresponds to the concept of table, in a database. A table is thus a set of tuples

The word «relational» in relational databases refers to the fact that data is stored in structured tables (relations), and not to the fact that tables have relationships between them (e.g., keys)

Users can manipulate tables through Structured Query Language, a language that allows multiple operations



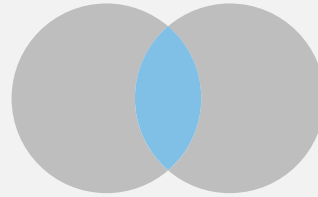
Filters



Selection



Projection



Set operations



Union



Intersection



Exclusion



Relational operations



Cartesian
Product



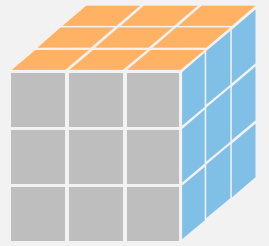
Join

Agenda

Introduction to relational data model

➤ SQL basic operations

Deep dive on JOIN



Filters allow to take only specific rows or columns of a table

σ

Selection

What it does

Considers only rows of a table that meet a specified condition

π

Projection

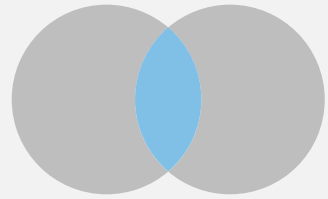
Considers only columns of a table as specified by a list of names

Example

```
SELECT *  
FROM customer  
WHERE name='bob'
```

```
SELECT name, surname  
FROM customer
```

Set operations combine or compare the results of multiple queries that return the same column structure



U

Union

∩

Intersection

\

Exclusion

What it does

Starting from 2 (or more) tables, creates a new table containing all records of the 1st and all those of the 2nd. The 2 tables must be made of the same type of tuples

Starting from 2 (or more) tables, creates a new table containing only records that are present in both tables. The 2 tables must be made of the same type of tuples

Starting from 2 (or more) tables, creates a new table containing only records that are in the first but not in the second. The 2 tables must be made of the same type of tuples

Example

```
SELECT name, surname
FROM customer
UNION
SELECT name, surname
FROM staff
```

```
SELECT name, surname
FROM customer
INTERSECT
SELECT name, surname
FROM staff
```

```
SELECT name, surname
FROM customer
EXCEPT
SELECT name, surname
FROM staff
```


Relational operations combine information from two or more tables by matching rows based on a condition



Cartesian product

What it does

Given 2 (or more) tables, creates a new table whose records are tuples obtained by combining a record of the first table with one of the second, until all possible pairs have been generated



Join

Given 2 (or more) tables, calculates the Cartesian product between them, and then filters only tuples where a specific condition (specified by the user) is met. Then, depending on the join type, special extra records can be returned

Example

```
SELECT *  
FROM customer, staff
```

```
SELECT *  
FROM customer  
JOIN staff ON  
customer.cust_id=staff.st_id
```

Agenda

Introduction to relational data model

SQL basic operations

➤ Deep dive on JOIN

Joins are of 2 types: inner & outer

Inner join

What it does

A Cartesian product (also called "Cross join") between 2 (or more) tables in which only combinations that fulfil a given predicate are retained

Examples (variants are equivalent)

```
-- "Classic" ANSI JOIN syntax
SELECT *
FROM customer c
JOIN staff s ON c.cust_id=s.staff_id
```

```
-- "Old" syntax using a "CROSS JOIN"
SELECT *
FROM customer c, staff s
WHERE c.cust_id=s.staff_id
```

Outer join

An inner join with extra records. These extra records are rows from either the LEFT, the RIGHT, or both (FULL) tables, for which no rows satisfying the predicate were found in the inner join results

```
SELECT *
FROM customer
LEFT JOIN staff ON
customer.cust_id=staff.st_id
```

```
SELECT *
FROM customer c
JOIN staff s ON c.cust_id=s.st_id

UNION

SELECT rows_not_matched.*, NULL, ..., NULL
FROM (
    SELECT rows_not_matched.*
    FROM customer c

    EXCEPT

    SELECT c.*
    FROM customer c
    JOIN staff s ON c.cust_id=s.st_id
) rows_not_matched
```

Inner Join is a filtered Cartesian product

orders

Order_id	Customer_id	Order_date
1003	AZ501	2025-07-03
1004	BB223	2025-07-03
1005	CX987	2025-07-03

orders-products

Order_id	product_id	quantity
1003	Xxx705	2
1003	Xxx102	1
1003	Xxx258	1
1004	Xxx258	3



Order_id	Customer_id	Order_date	Order_id	product_id	quantity
1003	AZ501	2025-07-03	1003	Xxx705	2
1003	AZ501	2025-07-03	1003	Xxx102	1
1003	AZ501	2025-07-03	1003	Xxx258	1
1003	AZ501	2025-07-03	1004	Xxx258	3
1004	BB223	2025-07-03	1003	Xxx705	2
1004	BB223	2025-07-03	1003	Xxx102	1
1004	BB223	2025-07-03	1003	Xxx258	1
1004	BB223	2025-07-03	1004	Xxx258	3
1005	CX987	2025-07-03	1003	Xxx705	2
1005	CX987	2025-07-03	1003	Xxx102	1
1005	CX987	2025-07-03	1003	Xxx258	1
1005	CX987	2025-07-03	1004	Xxx258	3



WHERE orders.order_id=orders-products.order_id

Order_id	Customer_id	Order_date	Order_id	product_id	quantity
1003	AZ501	2025-07-03	1003	Xxx705	2
1003	AZ501	2025-07-03	1003	Xxx102	1
1003	AZ501	2025-07-03	1003	Xxx258	1
1004	BB223	2025-07-03	1004	Xxx258	3

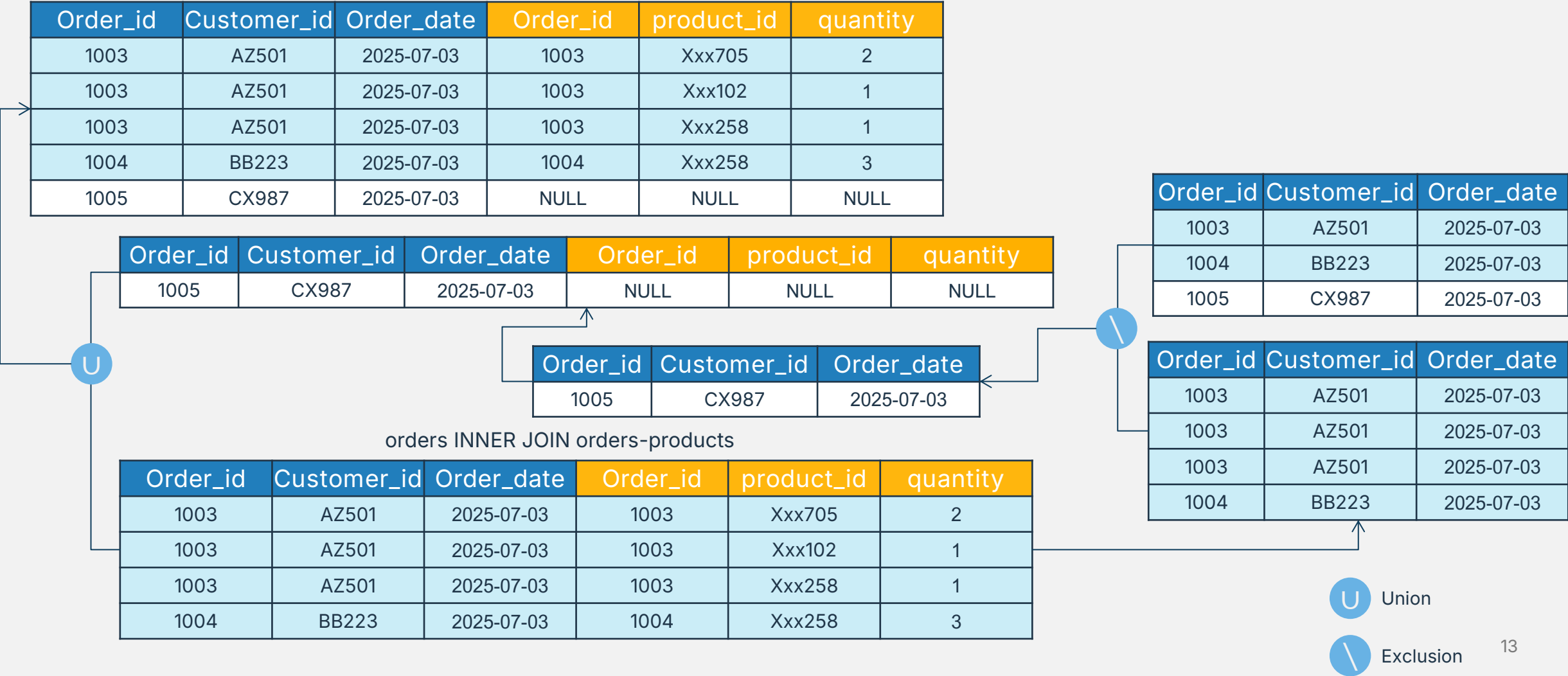


Cross product



Selection

Left Join is an Inner Join unioned with unmatched records from the left table padded with NULLs



Joins: ON vs WHERE clause

```
INSERT INTO existing  
SELECT * FROM incoming  
WHERE tech_date > '2025-  
07-01'
```

- implicit vs explicit
- ordine
- types (left, inner, ...)

Null-safe equality

```
INSERT INTO existing  
SELECT * FROM incoming  
WHERE tech_date > '2025-  
07-01'
```

Window over partition by

```
INSERT INTO existing  
SELECT * FROM incoming  
WHERE tech_date > '2025-  
07-01'
```


Row number, Rank, Dense Rank

```
INSERT INTO existing  
SELECT * FROM incoming  
WHERE tech_date > '2025-07-01'
```



Nicola Orecchini

nicolaorecchini@gmail.com

Join is a Selection on a Cartesian product

orders

Order_id	Customer_id	Order_date
1003	AZ501	2025-07-03
1004	BB223	2025-07-03
1005	CX987	2025-07-03
1006	DE658	2025-07-03
1007	GT514	2025-07-03

orders-products

Order_id	product_id	quantity
1003	Xxx705	2
1003	Xxx102	1
1003	Xxx258	1
1004	Xxx698	3
1005	Xxx417	4
1005	Xxx417	8
1006	Xxx417	2
1007	Xxx546	1
1007	xxx147	5

orders X orders-products

Order_id	product_id	quantity	Order_id	product_id	quantity
1003	AZ501	2025-07-03	1003	Xxx705	2
			1003	Xxx102	1
			1003	Xxx258	1
			1004	Xxx698	3
			1005	Xxx417	4
			1005	Xxx417	8
			1006	Xxx417	2
			1007	Xxx546	1
			1007	xxx147	5
1004	BB223	2025-07-03	1003	Xxx705	2
			1003	Xxx102	1
			1003	Xxx258	1
			1004	Xxx698	3
			1005	Xxx417	4
			1005	Xxx417	8
			1006	Xxx417	2
			1007	Xxx546	1
			1007	xxx147	5
...					