**Julio Pochet - SQL Inner Joins**

**SQL Inner Joins – How They Work**

When working with databases, we often need to combine data from multiple tables. That’s where **INNER JOIN** comes in. It pulls records that match in both tables and leaves out anything that doesn’t have a match. Let’s look at a couple of simple examples.

**Example 1: Customers & Orders**

Imagine we have two tables:

**Customers Table**

| **customer\_id** | **name** | **city** |
| --- | --- | --- |
| 1 | Alice | New York |
| 2 | Bob | Chicago |
| 3 | Charlie | Los Angeles |

**Orders Table**

| **order\_id** | **customer\_id** | **product** |
| --- | --- | --- |
| 101 | 1 | Laptop |
| 102 | 2 | Phone |
| 103 | 4 | Tablet |

Now, let’s say we want to find customers who have placed orders. Using an **INNER JOIN**, we can link these two tables by customer\_id:

SELECT customers.customer\_id, customers.name, orders.order\_id, orders.product

FROM customers

INNER JOIN orders ON customers.customer\_id = orders.customer\_id;

**Results:**

| **customer\_id** | **name** | **order\_id** | **product** |
| --- | --- | --- | --- |
| 1 | Alice | 101 | Laptop |
| 2 | Bob | 102 | Phone |

Here’s what happened:

* **Alice and Bob showed up** because they had matching orders in the orders table.
* **Charlie got left out** because he hasn’t placed an order.
* **Order 103 is ignored** because customer ID 4 doesn’t exist in the customer’ table.

**Alternative Query for the Same Results**

We could also use a WHERE EXISTS clause instead of INNER JOIN:

SELECT c.customer\_id, c.name, o.order\_id, o.product

FROM customers c

WHERE EXISTS (

SELECT 1 FROM orders o WHERE o.customer\_id = c.customer\_id

);

**Example 2: Employees & Departments**

**Employees Table**

| **employee\_id** | **name** | **department\_id** |
| --- | --- | --- |
| 1 | John | 101 |
| 2 | Sarah | 102 |
| 3 | Mark | 103 |

**Departments Table**

| **department\_id** | **department\_name** |
| --- | --- |
| 101 | HR |
| 102 | IT |
| 104 | Finance |

Now, let’s say we want to find employees who belong to departments that exist in the departments table:

SELECT employees.employee\_id, employees.name, departments.department\_name

FROM employees

INNER JOIN departments ON employees.department\_id = departments.department\_id;

**Results:**

| **employee\_id** | **name** | **department\_name** |
| --- | --- | --- |
| 1 | John | HR |
| 2 | Sarah | IT |

Here’s what happened:

* **John and Sarah showed up** because their department IDs matched existing ones.
* **Mark is left out** because department 103 isn’t listed in the departments’ table.

**Alternative Query for the Same Results**

We could use WHERE IN to get the same outcome:

SELECT employee\_id, name, department\_id

FROM employees

WHERE department\_id IN (SELECT department\_id FROM departments);

**Final Thoughts**

**INNER JOIN** is a great way to get matching data from two tables, but sometimes **subqueries** like WHERE EXISTS or WHERE IN can give the same results. The best option depends on the situation.

What do you guys think? Do you prefer **JOINs or subqueries** when working with databases? Have you ever run into issues where missing data causes problems? Let’s discuss it!