Practical 5 Part II

# What is a Join?

A **JOIN** combines records from two or more tables using a related column.

# Types of Joins:

1. **INNER JOIN** → Returns only matching records.
2. **LEFT JOIN** → Returns all records from the left table and matching records from the right table.
3. **RIGHT JOIN** → Returns all records from the right table and matching records from the left table.
4. **FULL OUTER JOIN** → Returns all records from both tables (not available in MySQL).
5. **CROSS JOIN** → Returns the Cartesian product of both tables.
6. **SELF JOIN** → Joins a table to itself.
7. **Customer Table**

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| customer\_id | NUMBER (PK) | PRIMARY KEY, AUTO-INCREMENT |
| name | VARCHAR2(100) | NOT NULL |
| email | VARCHAR2(100) | UNIQUE |
| phone | VARCHAR2(15) | NOT NULL |
| address | VARCHAR2(255) | NULLABLE |

1. **Product Table**

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| product\_id | NUMBER (PK) | PRIMARY KEY |
| name | VARCHAR2(100) | NOT NULL |
| category | VARCHAR2(50) | NOT NULL |
| price | DECIMAL(10,2) | NOT NULL |
| stock\_quantity | INT | NOT NULL |

1. **Order\_Details Table**

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| order\_id | NUMBER (PK) | PRIMARY KEY |
| customer\_id | NUMBER (FK) | FOREIGN KEY REFERENCES  Customer(customer\_id) |
| order\_date | DATE | NOT NULL |
| total\_amount | DECIMAL(10,2) | NOT NULL |

1. **Order\_Item Table**

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |
| order\_id | NUMBER (FK) | FOREIGN KEY REFERENCES  Order\_Details(order\_id) |
| product\_id | NUMBER (FK) | FOREIGN KEY REFERENCES  Product(product\_id) |
| quantity | INT | NOT NULL |
| subtotal | DECIMAL(10,2) | NOT NULL |

1. **Employee Table**

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraints** |

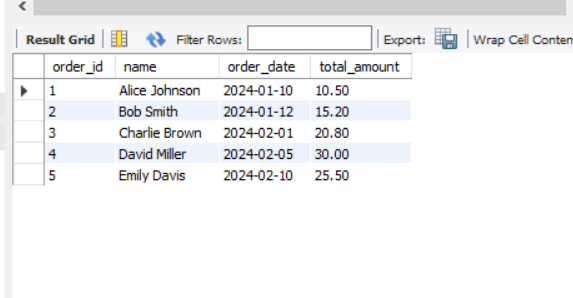
|  |  |  |
| --- | --- | --- |
| employee\_id | NUMBER (PK) | PRIMARY KEY |
| name | VARCHAR2(100) | NOT NULL |
| role | VARCHAR2(50) | NOT NULL |
| salary | DECIMAL(10,2) | NOT NULL |
| hire\_date | DATE | NOT NULL |

**Examples of Joins**

# INNER JOIN: Get order details with customer names

SELECT o.order\_id, c.name, o.order\_date, o.total\_amount FROM Order\_Details o

INNER JOIN Customer c ON o.customer\_id = c.customer\_id;



# INNER JOIN: Retrieve product names and their order quantities

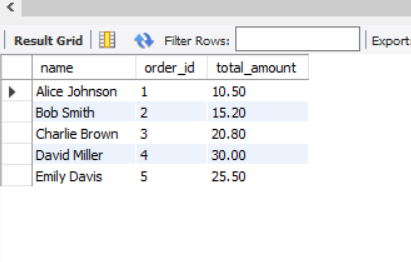
SELECT p.name, oi.quantity FROM Order\_Item oi

INNER JOIN Product p ON oi.product\_id = p.product\_id;

# LEFT JOIN: Get all customers and their orders (including those who never ordered)

SELECT c.name, o.order\_id, o.total\_amount FROM Customer c

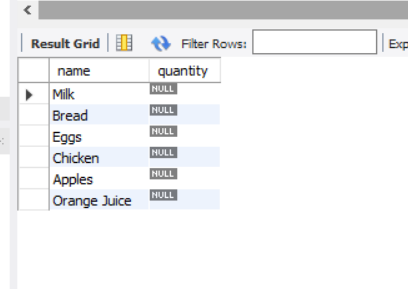
LEFT JOIN Order\_Details o ON c.customer\_id = o.customer\_id;



# LEFT JOIN: Retrieve all products and their order details (including those not ordered yet)

SELECT p.name, oi.quantity FROM Product p

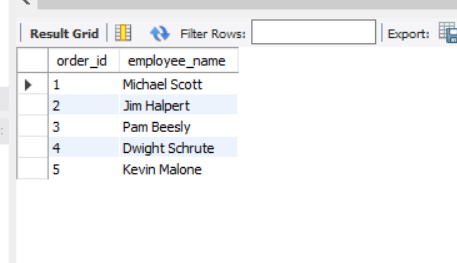
LEFT JOIN Order\_Item oi ON p.product\_id = oi.product\_id;



# RIGHT JOIN: Get all orders with or without employee assigned

SELECT o.order\_id, e.name AS employee\_name FROM Order\_Details o

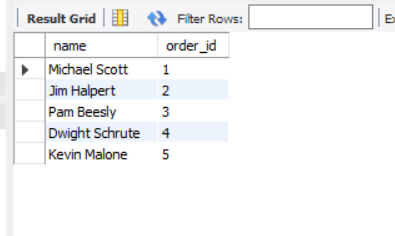
RIGHT JOIN Employee e ON o.customer\_id = e.employee\_id;



# RIGHT JOIN: Retrieve employees who processed orders

SELECT e.name, o.order\_id FROM Employee e

RIGHT JOIN Order\_Details o ON e.employee\_id = o.customer\_id;



# FULL OUTER JOIN: Get all customers and orders (Oracle SQL only)

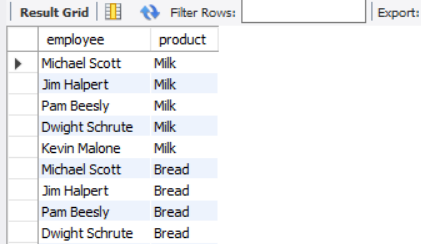
SELECT c.name, o.order\_id, o.total\_amount FROM Customer c

FULL OUTER JOIN Order\_Details o ON c.customer\_id = o.customer\_id;

# CROSS JOIN: Show all possible employee-product assignments

SELECT e.name AS employee, p.name AS product FROM Employee e

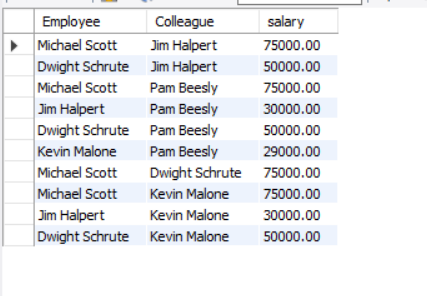
CROSS JOIN Product p;



# SELF JOIN: Find employees earning more than their colleagues

SELECT e1.name AS Employee, e2.name AS Colleague, e1.salary FROM Employee e1

JOIN Employee e2 ON e1.salary > e2.salary;



# 

# SELF JOIN: Find employees working under the same manager

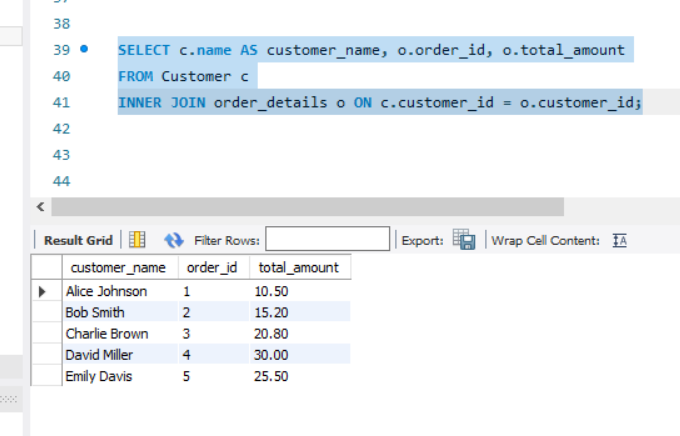
SELECT e1.name AS Employee, e2.name AS Manager FROM Employee e1

JOIN Employee e2 ON e1.role = 'Cashier' AND e2.role = 'Manager';

# 

**Joins Tasks**

1. Retrieve **customer names** along with their orders.



1. Show **product names** and their **order quantities**.
2. List all customers and their orders (**including those who never ordered**).

# Retrieve all products and their order details (including those not ordered yet).

1. Find employees who have **processed orders**.