

**Example of Creating Tables with Foreign Key:**

1. **Creating the departments Table:**

CREATE TABLE departments (

dept\_id INT PRIMARY KEY,

department\_name VARCHAR(100)

);

Creating the employees Table with a Foreign Key:

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

employee\_name VARCHAR(100),

dept\_ref INT,

FOREIGN KEY (dept\_ref) REFERENCES departments(dept\_id)

);

**Explanation:**

* dept\_id in the departments table is the primary key.
* dept\_ref in the employees table is the foreign key that references dept\_id in the departments table.

INSERT INTO departments (dept\_id, department\_name) VALUES (1, 'HR');

INSERT INTO employees (employee\_id, employee\_name, dept\_ref) VALUES (1, 'John Doe', 1);

If dept\_ref references a dept\_id that doesn’t exist in departments, it will fail:

INSERT INTO employees (employee\_id, employee\_name, dept\_ref) VALUES (2, 'Jane Doe', 99); -- Fails

**Summary:**

* Ensure the dept\_id column exists in the referenced table (departments).
* Make sure dept\_id is a primary or unique key in departments.
* Verify that the data types of the foreign key and referenced key columns match.

To retrieve the **department name** for each employee, you can perform a **JOIN** between the employees table and the departments table. This will allow you to match the dept\_ref column in the employees table with the dept\_id column in the departments table, so you can display the corresponding department name.

SELECT e.employee\_name, d.department\_name

FROM employees e

JOIN departments d ON e.dept\_ref = d.dept\_id;

**Explanation:**

* **employees e**: This refers to the employees table and gives it an alias (e) for easier reference.
* **departments d**: This refers to the departments table and gives it an alias (d).
* **JOIN**: This joins the two tables based on the condition that dept\_ref in the employees table matches dept\_id in the departments table.
* **e.employee\_name and d.department\_name**: These columns are selected to display each employee's name along with their corresponding department name.
* **Example:**
* If you have the following data:
* **departments table:**

| **dept\_id** | **department\_name** |
| --- | --- |
| 1 | HR |
| 2 | Sales |
| 3 | Marketing |

* **employees table:**

| **employee\_id** | **employee\_name** | **dept\_ref** |
| --- | --- | --- |
| 1 | John Doe | 1 |
| 2 | Jane Smith | 2 |
| 3 | Sam Green | 3 |

* **Result of the Query:**

| **employee\_name** | **department\_name** |
| --- | --- |
| John Doe | HR |
| Jane Smith | Sales |
| Sam Green | Marketing |

* This query joins both tables and retrieves the corresponding department name for each employee.

**Explanation of INNER JOIN:**

* An **INNER JOIN** returns only the rows where there is a match in both tables (i.e., where employees.dept\_ref matches departments.dept\_id).
* If there is no matching record in one of the tables, the row will not appear in the result set.

In this case:

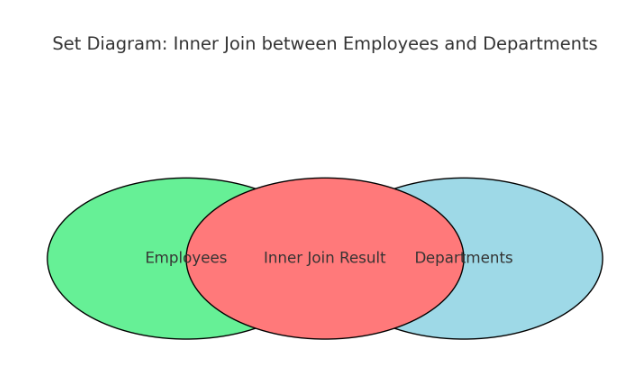
* It retrieves only employees that have a valid department (i.e., where dept\_ref in the employees table matches an existing dept\_id in the departments table).
* If an employee has a dept\_ref value that does not exist in the departments table, that employee would not appear in the result.

**Explanation of the Diagram:**

* **Employees Table**: Contains employee information with employee\_id as the primary key (PK) and dept\_ref as the foreign key (FK) that references the departments table.
* **Departments Table**: Contains department information with dept\_id as the primary key (PK).
* **Inner Join Result**: Represents the result of the inner join operation, showing the employee\_name and the department\_name for employees that have matching department records.

**Inner Join Logic:**

* An **Inner Join** returns only the rows where there is a match between dept\_ref in the Employees table and dept\_id in the Departments table. If an employee does not have a matching department, they will not appear in the result set.



**1. INNER JOIN (Intersection)**

* **Definition**: Returns only the rows where there is a match in both tables.
* **Set Diagram**: The intersection (overlapping area) of the two sets.

SELECT e.employee\_name, d.department\_name

FROM employees e

INNER JOIN departments d ON e.dept\_ref = d.dept\_id;

This query retrieves only the employees who have a valid department.