Foreign Keys in MySQL

A **foreign key** is a column that creates a **link between two tables**. It ensures that the value in one table **must match** a value in another table.

This is used to maintain **data integrity** between related data.

Why Use Foreign Keys?

Imagine this scenario:

You have a `users` table. Now you want to store each user's address. Instead of putting address columns inside the `users` table, you create a separate `addresses` table, and link it to `users` using a **foreign key**.

Creating a Table with a Foreign Key

Let's create an `addresses` table where each address belongs to a user.

```
CREATE TABLE addresses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    street VARCHAR(255),
    city VARCHAR(100),
    state VARCHAR(100),
    pincode VARCHAR(10),
    FOREIGN KEY (user_id) REFERENCES users(id)
);
```

Explanation:

- `user_id` is a **foreign key**.
- It references the `id` column in the `users` table.
- This ensures that every address must be linked to a valid user.

Dropping a Foreign Key

To drop a foreign key, you need to know its **constraint name**. MySQL auto-generates it if you don't specify one, or you can name it yourself:

```
CREATE TABLE addresses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(id)
);
```

To drop it:

```
ALTER TABLE addresses
DROP FOREIGN KEY fk_user;
```

Adding a Foreign Key Later (Using ALTER)

Suppose the foreign key was **not defined during table creation**. You can add it later using `ALTER TABLE`:

```
ALTER TABLE addresses
ADD CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(id);
```

Adding `ON DELETE` Action

By default, if you delete a user that has related addresses, MySQL will throw an error. You can control this behavior with `ON DELETE`.

Example with `ON DELETE CASCADE`:

If you want addresses to be automatically deleted when the user is deleted:

```
CREATE TABLE addresses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    street VARCHAR(255),
    city VARCHAR(100),
    state VARCHAR(100),
    pincode VARCHAR(10),
    CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);
```

Or alter it later:

```
ALTER TABLE addresses
ADD CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE;
```

Other ON DELETE Options

ON DELETE Option	Behavior
`CASCADE`	Deletes all related rows in child table
`SET NULL`	Sets the foreign key to NULL in the child table
`RESTRICT`	Prevents deletion of parent if child exists (default)

Summary

- Foreign keys **connect tables** and enforce **valid references**.
- You can create them inline or with `ALTER TABLE`.
- You can drop them by name.
- Use `ON DELETE` to control what happens when the parent row is deleted.