Add Relations

Relations connect tables together by linking data. A relation uses a Foreign Key to reference another table's Primary Key.

What is a Foreign Key?

A Foreign Key (FK) is a column in one table that references the Primary Key of another table.

Example: Users and Orders

users table:

• id (Primary Key)

• username

• email

orders table:

• id (Primary Key)

• user_id (Foreign Key - references users.id)

• total

• created_at

When you create an order, user_id links to a user in the users table. The user_id in orders must exist as an id in users.

Why Use Relations?

Relations ensure data integrity. You cannot:

• Create an order for a user that doesn't exist

• Delete a user if they have orders (depending on ON DELETE setting)

• Have orphaned data (order with no user)

How to Create a Relation

Step 1: Open Add Relation Dialog

Click the [Add Relation] button in toolbar OR press Ctrl+R

Step 2: Select Parent Table

Select the main table being referenced.

In users and orders example:

• Parent Table: users (the main table)

Parent table is usually the one you want to protect.

Step 3: Select Child Table

Select the table with the foreign key.

In users and orders example:

• Child Table: orders (the dependent table)

Child table has the foreign key column.

Step 4: Select Foreign Key Column

Choose which column in the child table references the parent.

In users and orders example:

• Foreign Key Column: user_id (in orders table)

This column will contain an id that matches users.id.

Step 5: Choose ON DELETE Action

Decide what happens when a parent row is deleted.

CASCADE - Delete Related Rows

Delete related rows in child table when parent deleted.

What happens:

• Delete user with id 5

• All orders with user_id 5 are also deleted

• Use when child data depends on parent

When to use:

• Orders must have a user

• Comments must have a post

• Images must have a gallery

SET NULL - Keep Child Rows

Set foreign key to NULL when parent deleted.

What happens:

• Delete user with id 5

• All orders with user_id 5 become user_id NULL

• Requires child FK column to be nullable

• Use when you want to keep history

When to use:

• Keep order history even if user deleted

• Keep comments even if author deleted

• Preserve historical records

RESTRICT - Prevent Deletion

Prevent deletion if foreign key exists.

What happens:

• Try to delete user with id 5

• Operation blocked because orders exist

• User cannot be deleted until orders are removed

• Safe but restrictive

When to use:

- Safety critical operations
- Prevent accidental deletions
- Enforce business rules

Complete Example: Orders to Users

Setup

Parent Table: users

• Contains: id, username, email

Child Table: orders

• Contains: id, user_id, total, created_at

Create Relation

• Parent Table: users

• Child Table: orders

• FK Column: user_id

• ON DELETE: CASCADE

Meaning

Each order must belong to a user. When user is deleted, all their orders are deleted too.

Result on Canvas

A line is drawn from users table to orders table showing the relationship.

Multiple Relations Example

You can have many relations in one database.

E-Commerce Database:

users table: id, username, email

products table: id, name, price

orders table: id, user_id (FK to users), created_at

order_items table: id, order_id (FK to orders), product_id (FK to products), quantity

Relations:

1. users -> orders (user_id, CASCADE)
2. orders -> order_items (order_id, CASCADE)
3. products -> order_items (product_id, SET NULL)

Foreign Key Naming Convention

Name foreign key columns consistently.

Good Names:

✓ user_id (references users table)

✓ product_id (references products table)

✓ category_id (references categories table)

✓ author_id (references authors table)

Bad Names:

✗ fk (unclear what it references)

✗ fk1, fk2, fk3 (numbered, confusing)

✗ uid (too abbreviated)

✗ user (could be confused with username)

Important Rules

Both tables must exist before creating relation

• Create users table first

• Create orders table second

• Then create relation

Foreign key column must exist

• Add user_id column to orders table

• Then create relation to users

Primary key type must match

• If users.id is INT, orders.user_id must be INT

• Types must be same for proper linking