

How to Draw an ERD for the Employee Database

A Beginner's Guide to Visualizing Database Structures

Using MySQL Workbench

Table of Contents

1. What is an ERD?	3
2. Core Concepts for Beginners	3
3. Step-by-Step: Generating an ERD in MySQL Workbench	5
4. Thinking Like a Database Designer	7
5. Interpreting the Final Diagram	9
6. Conclusion	10

1. What is an ERD?

An **Entity-Relationship Diagram (ERD)** is a visual blueprint of a database. Think of it like a map that shows all the main places (the tables) and the roads that connect them (the relationships). It helps you understand how different pieces of information in the database are related to each other without having to look at the raw data.

For a beginner, it's the fastest way to see the 'big picture' of the database you are working with.

2. Core Concepts for Beginners

Before we start, let's learn three simple terms you will see in an ERD:

1. Entities

An **entity** is a real-world object or concept. In our case, an entity is simply a **table**. The Employee Database has 6 entities: ``employees``, ``departments``, ``salaries``, etc.

2. Attributes

An **attribute** is a property or characteristic of an entity. For a database, attributes are the **columns** in a table. For the ``employees`` entity, the attributes are ``emp_no``, ``first_name``, ``last_name``, etc.

3. Primary Keys (PK)

A **Primary Key** is a special attribute that uniquely identifies each record in a table. No two rows can have the same primary key. Think of it like a student ID number—every student has one, and it's unique.

■ **How to Spot Them:** In MySQL Workbench, Primary Keys are often marked with a yellow key icon. They are frequently named with ``_id`` or ``_no``, like ``emp_no`` or ``dept_no``.

3. Step-by-Step: Generating an ERD in MySQL Workbench

MySQL Workbench has a powerful built-in tool that can create an ERD for you automatically. This process is called '**Reverse Engineering**'.

- 1 **Open MySQL Workbench** and connect to your local database instance (the same one where you installed the `employees` database).
- 2 From the top menu, click on **Database**, then select **Reverse Engineer...**
- 3 A wizard will open. On the first screen, your active connection should already be selected. Simply click **Next**.
- 4 The wizard will connect to your database and show a success message. Click **Next**.
- 5 You will see a list of all databases (schemas) on your server. Select the **employees** database from the list and click **Next**.
- 6 The wizard will retrieve the database objects. Once it's done, click **Next**.
- 7 Now you'll see a list of tables to include in the diagram (e.g., `employees`, `departments`). Make sure all 6 are checked, then click **Execute**.
- 8 The wizard will now generate the ERD. This might take a moment. Once it's finished, click **Next**, and then **Finish**.
- 9 **Congratulations!** A new tab will open showing the complete ERD for the `employees` database.

■ **Tip:** You can drag the tables around in the diagram to rearrange them for better readability.

4. Thinking Like a Database Designer

The automatic generator is great, but a true understanding comes from learning to spot the key components yourself. Let's practice finding the entities and their primary keys.

Task 1: Identify the Main Entities

In the MySQL Workbench Navigator (the left sidebar), expand the `employees` schema. You will see a 'Tables' dropdown. The 6 tables listed there are our main entities.

employees	departments	dept_emp
dept_manager	salaries	titles

Task 2: Find the Primary Keys (The Unique IDs)


Now, let's find the primary key for our two main tables. You can do this by expanding a table in the Navigator, then expanding 'Columns'. The column with a  **key icon** next to it is the Primary Key.

Table Name	Primary Key (PK)	What it means
employees	`emp_no`	The unique ID number for each employee.
departments	`dept_no`	The unique ID number for each department.

Task 3: Discover the Relationships

Relationships are formed when one table's primary key appears in another table. Look at the `dept_emp` table's columns. You will see it contains both `emp_no` and `dept_no`. This table exists only to connect, or 'link', the `employees` table with the `departments` table. It tells us which employees have worked in which departments.

This is called a **linking table** and it is a sign of a 'many-to-many' relationship (many employees can be in one department, and one employee could have been in many departments over time).

5. Interpreting the Final Diagram

When you look at the ERD you generated, you will see:

- **Boxes:** Each box represents a table (an entity). The title of the box is the table name.
- **Lines:** The lines connecting the boxes show the relationships between them.
- **Key Icons (PK):** Inside each box, one column will have a 'PK' (Primary Key) icon. This is the unique identifier for that table.
- **Red Icons (FK):** Some columns will have a red 'FK' (Foreign Key) icon. A Foreign Key is a primary key from one table that has been added to another table to link them. For example, `emp_no`` is a PK in `employees`` but an FK in `salaries``.
- **Crow's Feet:** The symbols at the end of the lines describe the relationship. A single dash means 'one', and a three-pronged symbol (like a crow's foot) means 'many'. A line connecting `departments`` and `dept_emp`` shows that one department can have many entries in the `dept_emp`` table.

Example: Find the line between the `employees`` table and the `salaries`` table. It shows that ONE employee can have MANY salaries recorded over time. This is a classic 'one-to-many' relationship.

6. Conclusion

You have now successfully generated an ERD and learned the basic principles of how it is constructed! This diagram is your map to navigating the `employees`` database. Before writing a complex query, it's always a good idea to look at the ERD to understand how the tables you need are connected.

Keep practicing by exploring the relationships between the other tables. Happy querying!