1. Write down a sample requirement for a database.

2. Identify nouns from the requirement statement. There will be a table for each noun.

3. Identify possible attributes for each table.

4. Identify relationships for all the tables.

5. Assesses the tables and fields and adjusts them if needed and implement the design.

## Step 1: Problem Statement

The data science company requires a centralized database system. It should streamline project tracking, employee information, customer relations, and financial records, while also enabling easy access to sales and expense data. Additionally, it should support the documentation and organization of future plans and initiatives, ensuring seamless coordination and decision-making across the organization's operational aspects.

## Step 2: Identifying Tables

The data science company requires a centralized database system. It should streamline **project** tracking, **employee** information, **customer** relations, and **financial records**, while also enabling easy access to **sales** and **expense** data. Additionally, it should support the documentation and organization of **future plans and initiatives**, ensuring seamless coordination and decision-making across the organization's operational aspects.

**Extension:** It is very common that a company has different departments to operate smoothly. So there will a few **departments** in the company from the problem statement. Also, I can further sub-divided projects under **tasks** for employees.

**Table Names**

1. Departments
2. Employees
3. Projects
4. Tasks
5. Customers
6. Sales
7. Expenses
8. Future Plans

## Step 3: Identifying Attributes

**Departments:**

Department Number<br>
Department Name

**Employees:**

Employee ID
First Name
Last Name
Email
Phone Number
Department Number
Position
Start Date

**Projects:**

Project ID
Project Name
Start Date
End Date
Status
Department Number

**Tasks:**

Task ID
Project ID
Employee ID
Due Date
Status

**Customers:**

Customer ID
Customer Name
Email
Phone Number
Address
Project ID

**Sales:**

Sale ID
Customer ID
Date
Sale’s Price

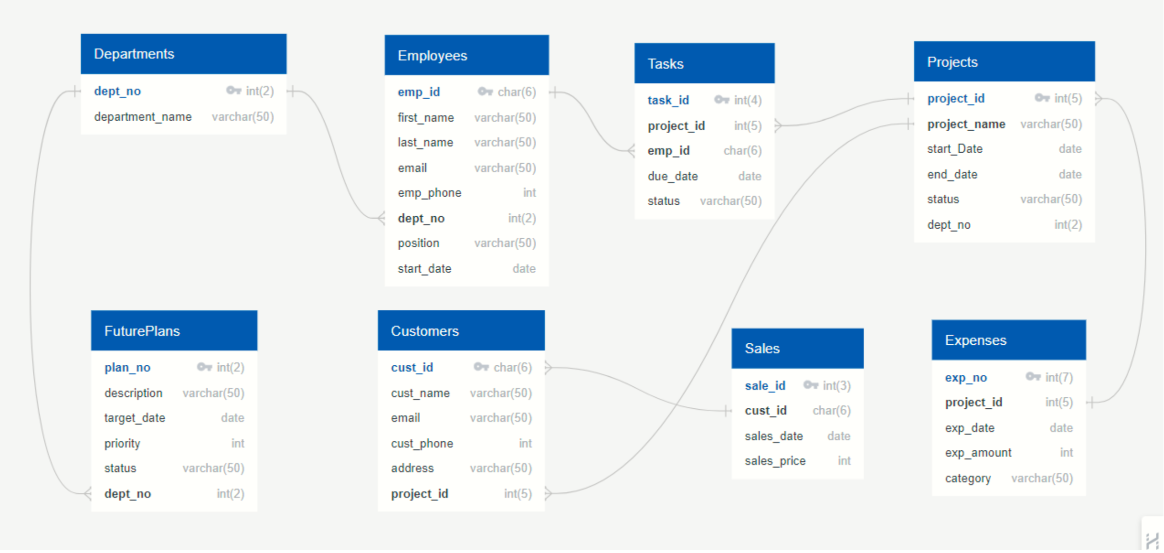
**Expenses:**

Expense Number
Project ID
Date
Amount
Category

**Future Plans:**

Plan Number
Description
Target Date
Priority
Status
Department Number

## Step 4: Relationships build up



| Departments | |
|---|---|
| dept_no | int(2) |
| department_name | varchar(50) |

| Employees | |
|---|---|
| emp_id | char(6) |
| first_name | varchar(50) |
| last_name | varchar(50) |
| email | varchar(50) |
| emp_phone | int |
| dept_no | int(2) |
| position | varchar(50) |
| start_date | date |

| Tasks | |
|---|---|
| task_id | int(4) |
| project_id | int(5) |
| emp_id | char(6) |
| due_date | date |
| status | varchar(50) |

| Projects | |
|---|---|
| project_id | int(5) |
| project_name | varchar(50) |
| start_Date | date |
| end_date | date |
| status | varchar(50) |
| dept_no | int(2) |

| FuturePlans | |
|---|---|
| plan_no | int(2) |
| description | varchar(50) |
| target_date | date |
| priority | int |
| status | varchar(50) |
| dept_no | int(2) |

| Customers | |
|---|---|
| cust_id | char(6) |
| cust_name | varchar(50) |
| email | varchar(50) |
| cust_phone | int |
| address | varchar(50) |
| project_id | int(5) |

| Sales | |
|---|---|
| sale_id | int(3) |
| cust_id | char(6) |
| sales_date | date |
| sales_price | int |

| Expenses | |
|---|---|
| exp_no | int(7) |
| project_id | int(5) |
| exp_date | date |
| exp_amount | int |
| category | varchar(50) |

## **Step 5: Implimentation (using MySQL)**

```
-- creation database
DROP DATABASE IF EXISTS dsc_database;
CREATE DATABASE dsc_database;
USE dsc_database;

-- departments table
CREATE TABLE departments (
   dept_no INT(2),
   department_name VARCHAR(50),
   PRIMARY KEY (dept_no)
);

-- employees table
CREATE TABLE employees (
   emp_id CHAR(6),
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   email VARCHAR(50),
   emp_phone INT,
   dept_no INT(2),
   position VARCHAR(50),
   start_date DATE,
   PRIMARY KEY (emp_id),
   CONSTRAINT fk_dept_no_departments FOREIGN KEY (dept_no) REFERENCES departments(dept_no)
);

-- projects table
CREATE TABLE projects (
   project_id INT(5),
   project_name VARCHAR(50),
   start_date DATE,
   end_date DATE,
   status VARCHAR(50),
   dept_no INT(2),
   PRIMARY KEY (project_id)
);

-- tasks table
CREATE TABLE tasks (
   task_id INT(4),
   project_id INT(5),
   emp_id CHAR(6),
   due_date DATE,
   status VARCHAR(50),
   PRIMARY KEY (task_id),
   CONSTRAINT fk_tasks_project_id FOREIGN KEY (project_id) REFERENCES projects(project_id),
   CONSTRAINT fk_tasks_emp_id FOREIGN KEY (emp_id) REFERENCES employees(emp_id)
);
```

```
-- customers table
CREATE TABLE customers (
   cust_id CHAR(6),
   cust_name VARCHAR(50),
   email VARCHAR(50),
   cust_phone INT,
   address VARCHAR(50),
   project_id INT(5),
   PRIMARY KEY (cust_id),
   CONSTRAINT fk_customers_project_id FOREIGN KEY (project_id) REFERENCES
projects(project_id)
);

-- sales table
CREATE TABLE sales (
   sale_id INT(3),
   cust_id CHAR(6),
   sales_date DATE,
   sales_price INT,
   PRIMARY KEY (sale_id),
   CONSTRAINT fk_sales_cust_id FOREIGN KEY (cust_id) REFERENCES customers(cust_id)
);

-- expenses table
CREATE TABLE expenses (
   exp_no INT(7),
   project_id INT(5),
   exp_date DATE,
   exp_amount INT,
   category VARCHAR(50),
   PRIMARY KEY (exp_no),
   CONSTRAINT fk_expenses_project_id FOREIGN KEY (project_id) REFERENCES
projects(project_id)
);

-- futurePlans table
CREATE TABLE futurePlans (
   plan_no INT(3),
   description VARCHAR(50),
   target_date DATE,
   priority INT,
   status VARCHAR(50),
   dept_no INT(2),
   PRIMARY KEY (plan_no),
   CONSTRAINT fk_dept_no_futurePlans FOREIGN KEY (dept_no) REFERENCES
departments(dept_no)
);
```