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
- 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
- Sale ID
- Project ID

Sales:
- Sale ID
- Customer ID
- Date
- Sale’s Price

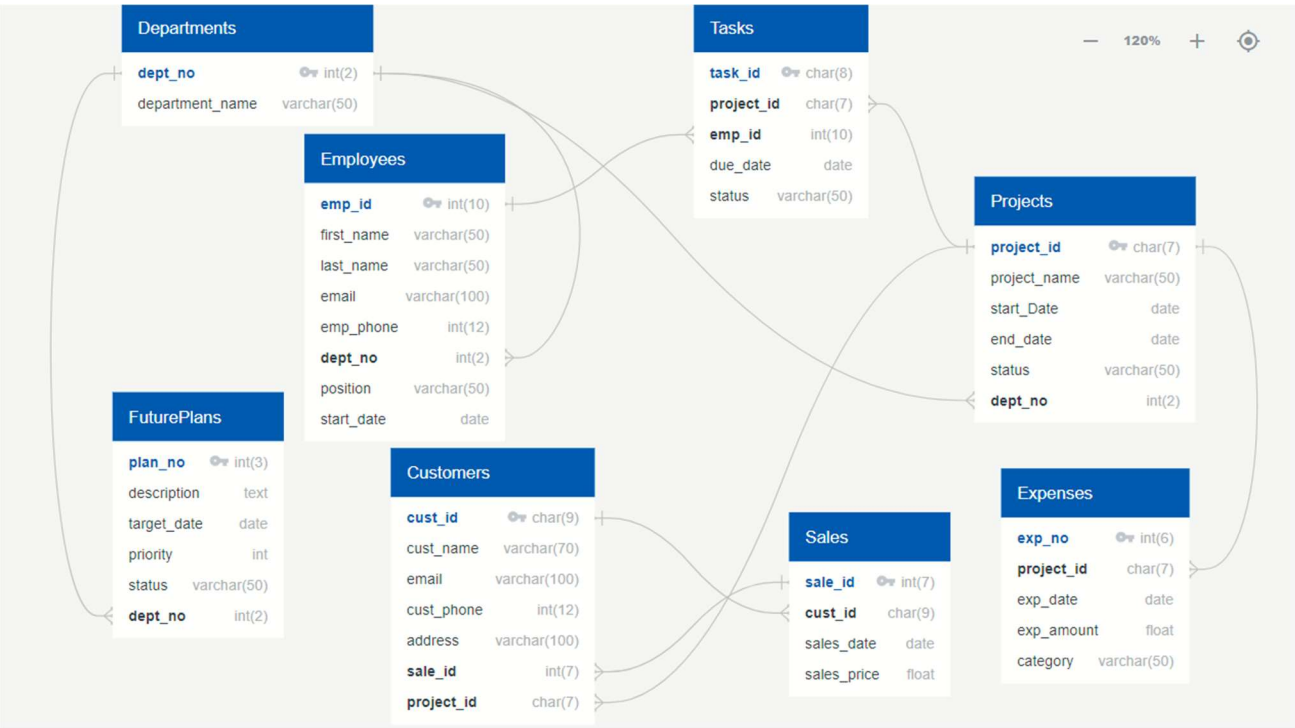
Expenses:

Expense Number
Project ID
Date
Amount
Category

Future Plans:

Plan Number
Description
Target Date
Priority
Status
Department Number

Step 4: Relationships build up



Step 5: Implimentation (using MySQL)

```
-- MySQL Workbench 8.0 CE
-- Data Science Company Database (dsc_database)
-- My GitHub Link: https://github.com/kawserabdullah

-- Creat Database
DROP DATABASE IF EXISTS dsc_database;
CREATE DATABASE dsc_database;
USE dsc_database;

-- Departments Table
CREATE TABLE `Departments` (
    `dept_no` int(2)  NOT NULL ,
    `department_name` varchar(50)  NOT NULL ,
    PRIMARY KEY (
        `dept_no`
    )
);

-- Employees Table
CREATE TABLE `Employees` (
    `emp_id` int(10)  NOT NULL ,
    `first_name` varchar(50)  NOT NULL ,
    `last_name` varchar(50)  NOT NULL ,
    `email` varchar(100)  NOT NULL ,
    `emp_phone` int(12)  NOT NULL ,
    `dept_no` int(2)  NOT NULL ,
    `position` varchar(50)  NOT NULL ,
    `start_date` date  NOT NULL ,
    PRIMARY KEY (
        `emp_id`
    )
);

-- Projects Table
CREATE TABLE `Projects` (
    `project_id` char(7)  NOT NULL ,
    `project_name` varchar(50)  NOT NULL ,
    `start_Date` date  NOT NULL ,
    `end_date` date  NOT NULL ,
    `status` varchar(50)  NOT NULL ,
    `dept_no` int(2)  NOT NULL ,
    PRIMARY KEY (
        `project_id`
    )
);
```

-- Tasks Table

```
CREATE TABLE `Tasks` (
   `task_id` char(8)  NOT NULL ,
   `project_id` char(7)  NOT NULL ,
   `emp_id` int(10)  NOT NULL ,
   `due_date` date  NOT NULL ,
   `status` varchar(50)  NOT NULL ,
   PRIMARY KEY (
      `task_id`
   )
);
```

-- Customers Table

```
CREATE TABLE `Customers` (
   `cust_id` char(9)  NOT NULL ,
   `cust_name` varchar(70)  NOT NULL ,
   `email` varchar(100)  NOT NULL ,
   `cust_phone` int(12)  NOT NULL ,
   `address` varchar(100)  NOT NULL ,
   `sale_id` int(7)  NOT NULL ,
   `project_id` char(7)  NOT NULL ,
   PRIMARY KEY (
      `cust_id`
   )
);
```

-- Sales Table

```
CREATE TABLE `Sales` (
   `sale_id` int(7)  NOT NULL ,
   `cust_id` char(9)  NOT NULL ,
   `sales_date` date  NOT NULL ,
   `sales_price` float  NOT NULL ,
   PRIMARY KEY (
      `sale_id`
   )
);
```

-- Expenses Table

```
CREATE TABLE `Expenses` (
   `exp_no` int(6)  NOT NULL ,
   `project_id` char(7)  NOT NULL ,
   `exp_date` date  NOT NULL ,
   `exp_amount` float  NOT NULL ,
   `category` varchar(50)  NOT NULL ,
   PRIMARY KEY (
      `exp_no`
   )
);
```

-- FuturePlans Table

```
CREATE TABLE `FuturePlans` (
   `plan_no` int(3)  NOT NULL ,
   `description` text  NOT NULL ,
   `target_date` date  NOT NULL ,
   `priority` int  NOT NULL ,
   `status` varchar(50)  NOT NULL ,
   `dept_no` int(2)  NOT NULL ,
   PRIMARY KEY (
      `plan_no`
   )
);
```

-- Relationships Build up using PK and FK

```
ALTER TABLE `Employees` ADD CONSTRAINT `fk_Employees_dept_no` FOREIGN
KEY(`dept_no`)
REFERENCES `Departments` (`dept_no`);

ALTER TABLE `Projects` ADD CONSTRAINT `fk_Projects_dept_no` FOREIGN KEY(`dept_no`)
REFERENCES `Departments` (`dept_no`);

ALTER TABLE `Tasks` ADD CONSTRAINT `fk_Tasks_project_id` FOREIGN KEY(`project_id`)
REFERENCES `Projects` (`project_id`);

ALTER TABLE `Tasks` ADD CONSTRAINT `fk_Tasks_emp_id` FOREIGN KEY(`emp_id`)
REFERENCES `Employees` (`emp_id`);

ALTER TABLE `Customers` ADD CONSTRAINT `fk_Customers_sale_id` FOREIGN KEY(`sale_id`)
REFERENCES `Sales` (`sale_id`);

ALTER TABLE `Customers` ADD CONSTRAINT `fk_Customers_project_id` FOREIGN
KEY(`project_id`)
REFERENCES `Projects` (`project_id`);

ALTER TABLE `Sales` ADD CONSTRAINT `fk_Sales_cust_id` FOREIGN KEY(`cust_id`)
REFERENCES `Customers` (`cust_id`);

ALTER TABLE `Expenses` ADD CONSTRAINT `fk_Expenses_project_id` FOREIGN
KEY(`project_id`)
REFERENCES `Projects` (`project_id`);

ALTER TABLE `FuturePlans` ADD CONSTRAINT `fk_FuturePlans_dept_no` FOREIGN
KEY(`dept_no`)
REFERENCES `Departments` (`dept_no`);
```