Problem Statement:

How can organizations optimize employee compensation structures (salaries, overtime pay) to balance cost efficiency and employee satisfaction while minimizing attrition, based on departmental and positional trends?

Salaries and overtime costs are significant components of organizational expenses. Through analyzing salary growth over time, payroll cost trends, and compensation's impact on attrition, this project aims to identify opportunities to optimize compensation structures while maintaining employee satisfaction and retention.

Project Overview:

This project focuses on dissecting payroll data to unveil insights that can help organizations streamline their compensation strategies. By examining various facets of payroll, such as departmental salary distributions and overtime impacts, the project aims to propose actionable strategies that align employee compensation with corporate financial goals and employee retention targets.

The approach involves querying a comprehensive payroll database to analyze trends and patterns. The insights derived will guide adjustments in the compensation structure to enhance cost management while maintaining or improving employee satisfaction and loyalty. The end goal is to deliver a framework that supports sustainable payroll practices that foster a motivated and efficient workforce.

Analysis

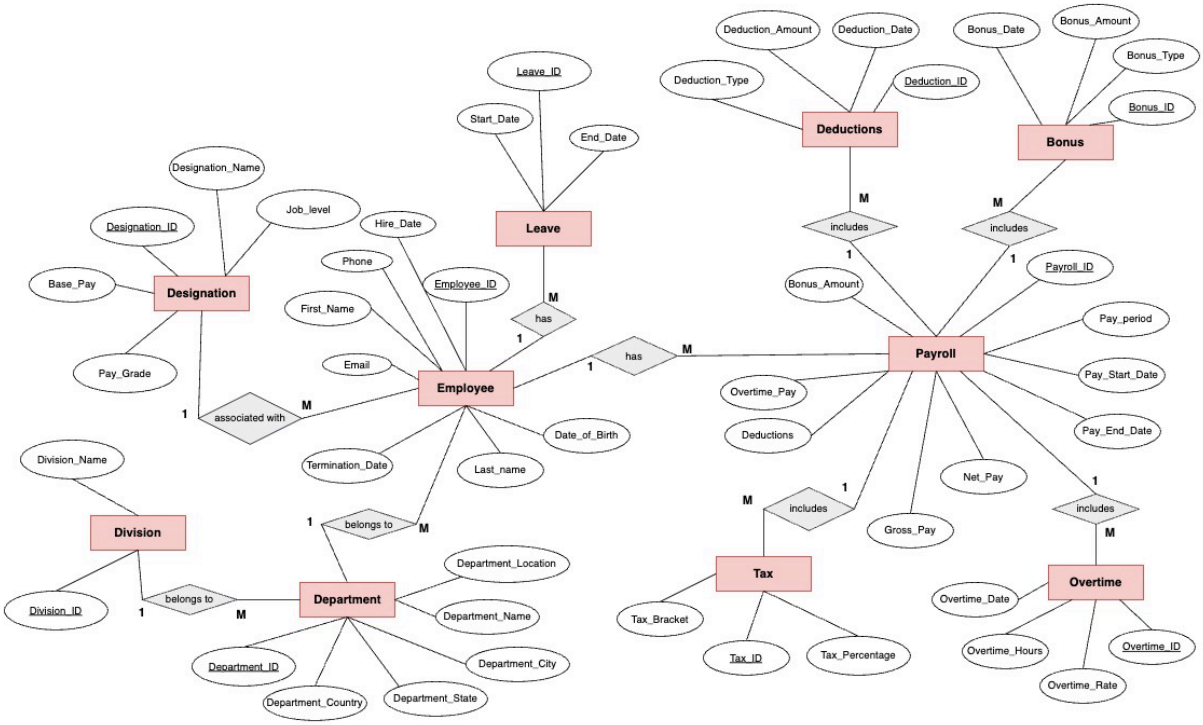
Payroll Cost Analysis: This analysis focuses on understanding the total payroll costs by examining how different departments and job positions accumulate costs. It identifies significant cost drivers and explores opportunities for adjustments in the compensation structure that could lead to more sustainable financial practices.

Overtime Analysis :The overtime analysis assesses the impact of overtime work on payroll expenses. It identifies departments or positions that incur substantial overtime costs and evaluates whether these costs are justified by output or if they signal inefficiencies in workforce management, such as under-staffing or poor process optimization.

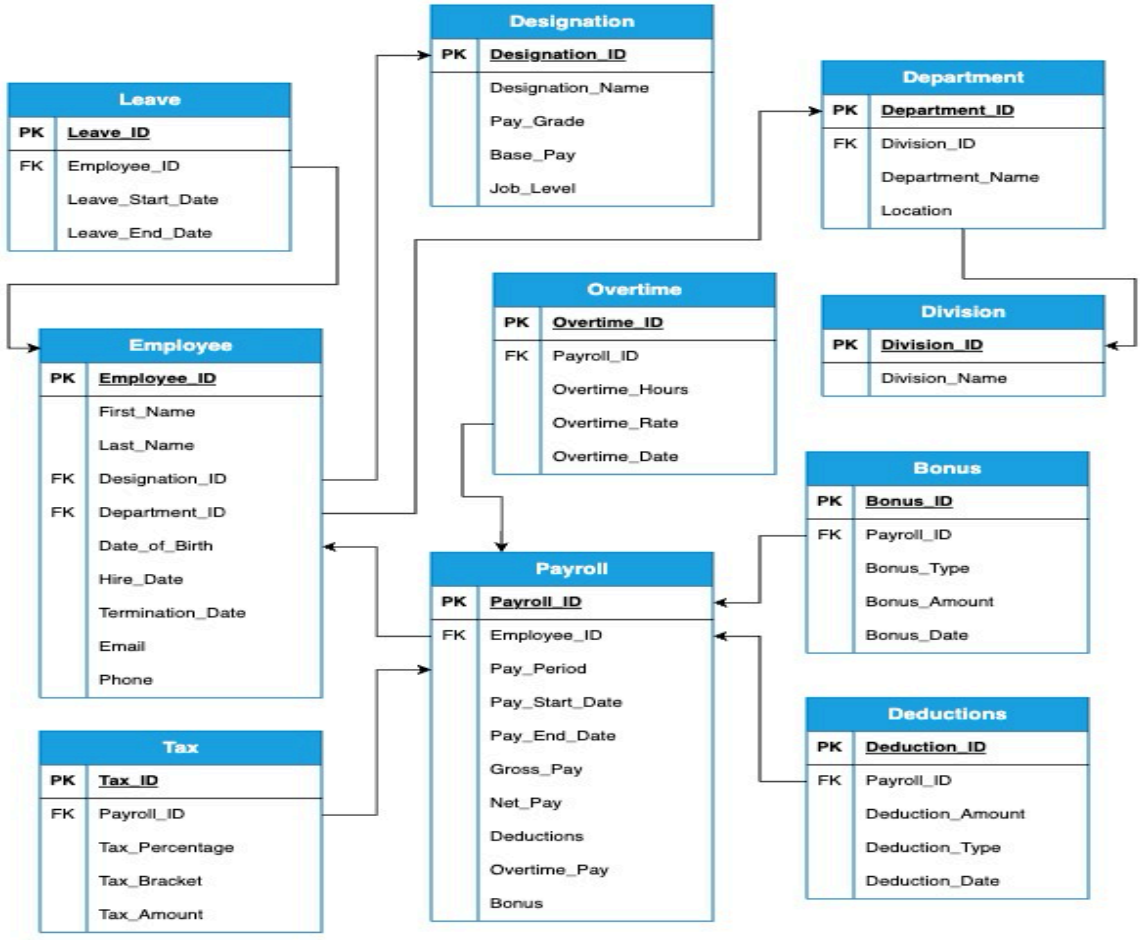
Salary Growth Over Time: This segment tracks how salaries have evolved within the organization over a specified period. The goal is to identify trends in salary increments and compare them across departments and positions to determine if salary changes are in alignment with market trends and company performance.

Employee Attrition and Compensation:Understanding the relationship between compensation and employee turnover is crucial for refining retention strategies. This analysis examines whether there are patterns in the compensation packages of employees who leave versus those who stay, aiming to pinpoint compensation-related factors that might influence attrition rates.

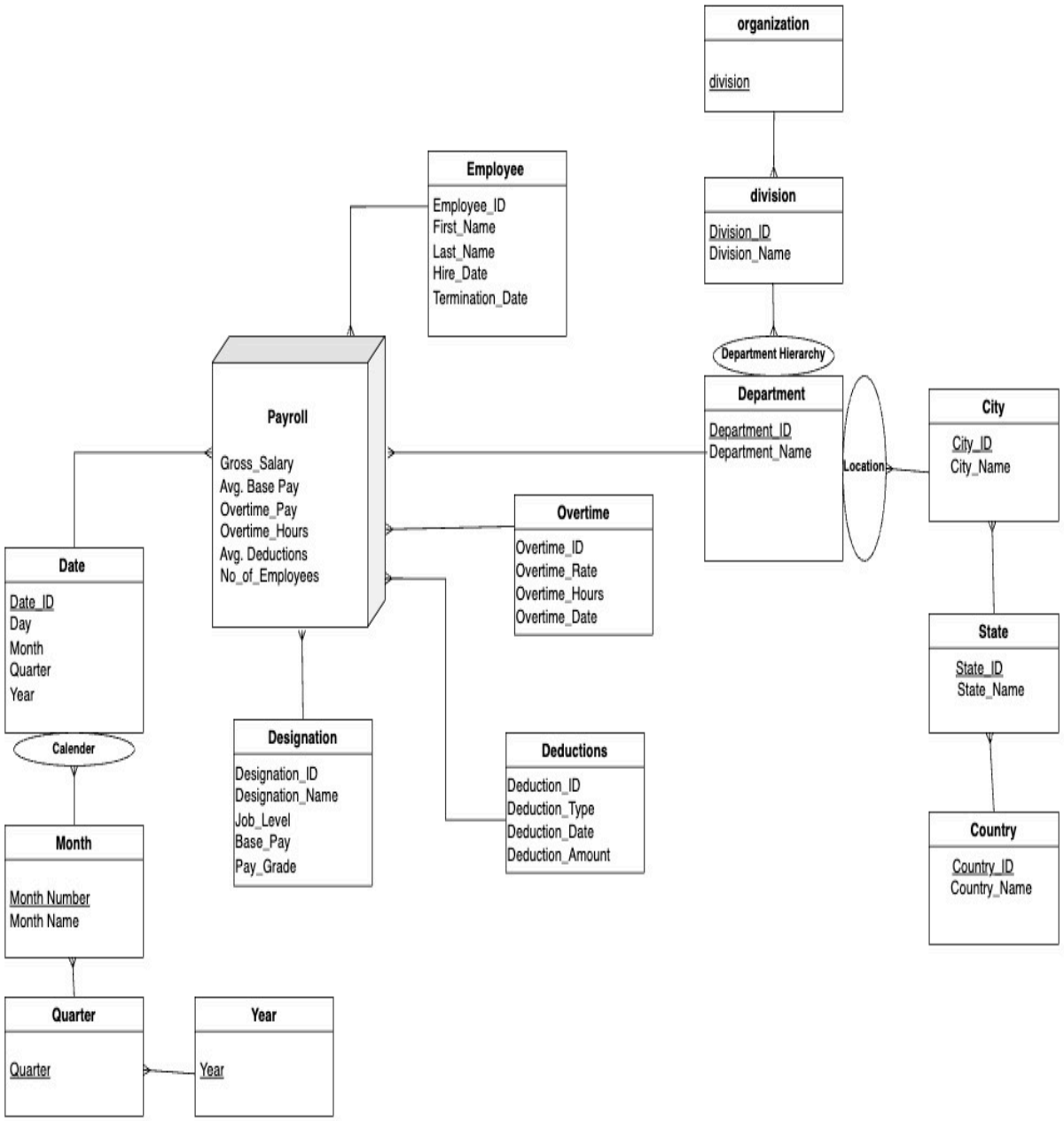
Entity Relationship Model



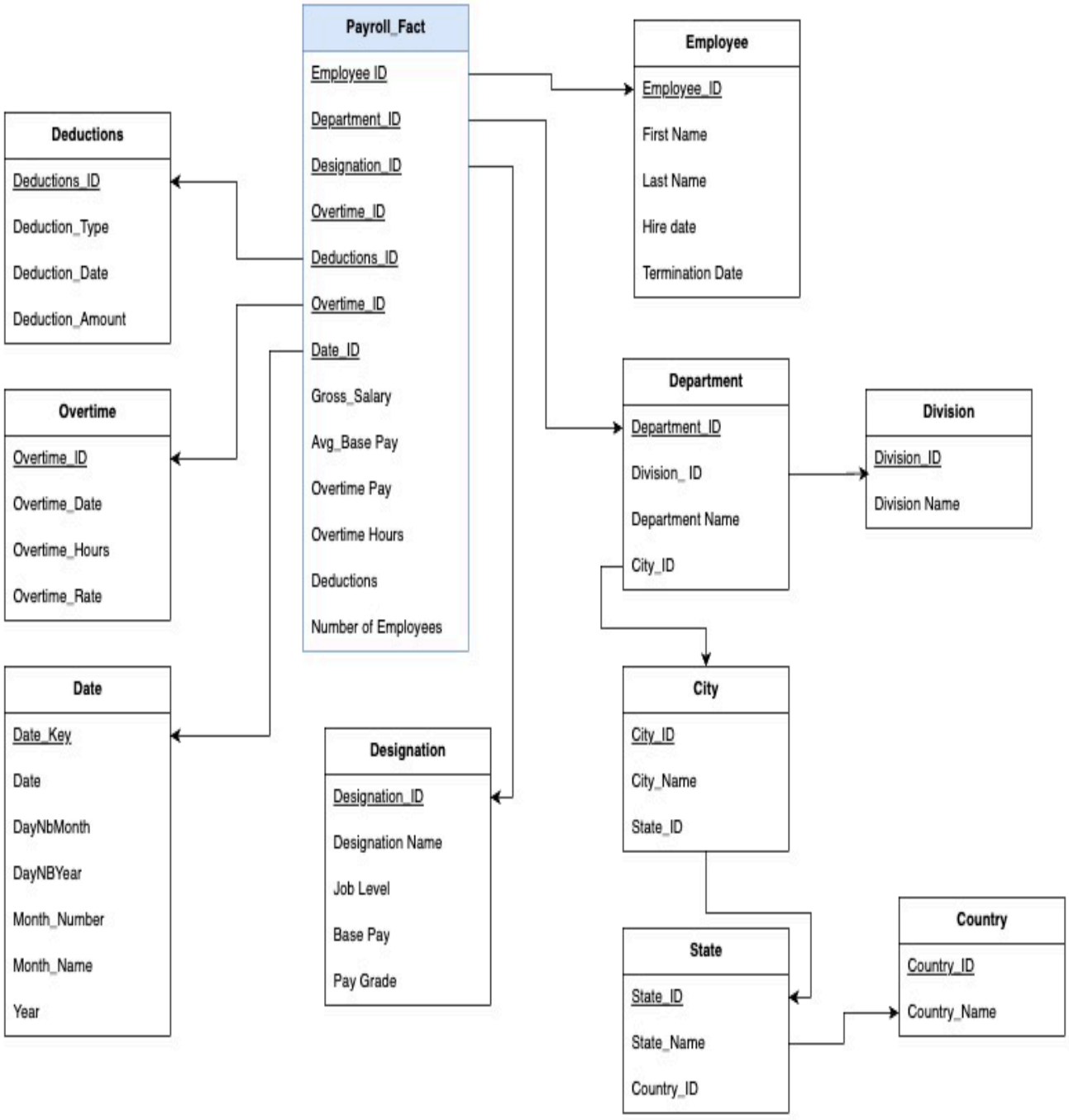
OLTP Relational Model



Conceptual DW Model



Relational OLAP Model



Primary Events

1. Overtime Event (Overtime Table)

- **Description:** This event logs overtime hours worked by employees and the pay associated with it.
- **Key Attributes:**
    - **Overtime_ID:** Unique identifier for overtime entries.
    - **Employee_ID:** The employee who worked overtime.
    - **Overtime_Hours:** Total hours of overtime.
    - **Overtime_Rate:** Rate at which overtime is paid.
    - **Overtime_Pay:** Total pay for the overtime period.
    - **Overtime_Date:** Date of overtime entry.

2. Deduction Event (Deductions Table)

- **Description:** This event captures various deductions made from employee salaries, such as social security, insurance, or loan repayments.
- **Key Attributes:**
    - **Deduction_ID:** Unique identifier for the deduction entry.
    - **Employee_ID:** Employee subjected to the deduction.
    - **Deduction_Amount:** Total deduction amount.
    - **Deduction_Date:** Date when the deduction was applied.
    - **Deduction_Type:** Type of deduction (e.g., tax, insurance).

OLAP Operations

1. Roll-Up

Operation: Aggregates data to a higher level in the hierarchy.

Query: Payroll <- ROLLUP (Payroll, Employee -> Department -> Division, SUM(Gross_Salary))

Result <- DICE (Payroll, Time.Month = 'December')

2. Drill-Down

Operation: Drills down to a more granular level of data.

Query: Payroll <- DRILLDOWN (Division -> Department -> Employee, SUM(Net_Salary))

Result <- SLICE (Time.Year = '2023')

3. Slice

Operation: Filters the data for a particular dimension.

Query: Result <- SLICE (Payroll, Time.Month = 'March')

4. Dice

Operation: Filters data based on multiple conditions.

Query: Result <- DICE (Payroll, Employee.Employee_Type = 'Full-Time' AND Department.Department_Name = 'IT')

5. Pivot (Rotate)

Operation: Reorients the data to view it from different perspectives.

Query: Pivot (Payroll, SUM(Overtime_Pay) BY Time.Quarter, Employee.Employee_Type)

6. Aggregation

Operation: Summarizes data using an aggregation function.

Query: Aggregate (Payroll, SUM(Gross_Salary), AVG(Tax_Percentage) BY Department)

7. Ranking

Operation: Ranks data based on certain metrics.

Query: Rank (Payroll, Gross_Salary BY Employee_ID DESC)

8. Rollup

Operation: Aggregate payroll data by month and department to summarize Gross Salary.

Query: Payroll <- ROLLUP (Payroll, Time.Month -> Department, SUM(Gross_Salary))

Result <- DICE (Payroll, Time.Year = '2023')

DB Implementation

Country Table

```
CREATE TABLE Country (
    Country_ID INT PRIMARY KEY,
    Country_Name VARCHAR(255)
);
```

State Table

```
CREATE TABLE State (
    State_ID INT PRIMARY KEY,
```

```
    State_Name VARCHAR(255),
    Country_ID INT,
    FOREIGN KEY (Country_ID) REFERENCES Country(Country_ID)
);
```

City Table

```
CREATE TABLE City (
    City_ID INT PRIMARY KEY,
    City_Name VARCHAR(255),
    State_ID INT,
    FOREIGN KEY (State_ID) REFERENCES State(State_ID)
);
```

Division Table

```
CREATE TABLE Division (
    Division_ID INT PRIMARY KEY,
    Division_Name VARCHAR(255)
);
```

Department Table

```
CREATE TABLE Department (
    Department_ID INT PRIMARY KEY,
    Division_ID INT,
    Department_Name VARCHAR(255),
    City_ID INT,
    FOREIGN KEY (Division_ID) REFERENCES Division(Division_ID),
    FOREIGN KEY (City_ID) REFERENCES City(City_ID)
);
```

Designation Table

```
CREATE TABLE Designation (
    Designation_ID INT PRIMARY KEY,
    Designation_Name VARCHAR(255),
    Job_Level VARCHAR(255),
    Base_Pay DECIMAL(10,2),
    Pay_Grade VARCHAR(255)
);
```

Employee Table

```
CREATE TABLE Employee (
    Employee_ID INT PRIMARY KEY,
    First_Name VARCHAR(255),
    Last_Name VARCHAR(255),
    Hire_Date DATE,
    Termination_Date DATE,
    Department_ID INT,
    Designation_ID INT,
    FOREIGN KEY (Department_ID) REFERENCES Department(Department_ID),
    FOREIGN KEY (Designation_ID) REFERENCES Designation(Designation_ID)
);
```

Date Table

```
CREATE TABLE Date (
    Date_Key INT PRIMARY KEY,
    Date DATE,
    DayNbMonth INT,
    DayNbYear INT,
    Month_Number INT,
    Month_Name VARCHAR(255),
```

```
    Year INT
);
```

Overtime Table

```
CREATE TABLE Overtime (
    Overtime_ID INT PRIMARY KEY,
    Overtime_Date DATE,
    Overtime_Hours INT,
    Overtime_Rate DECIMAL(10,2)
);
```

Deductions Table

```
CREATE TABLE Deductions (
    Deductions_ID INT PRIMARY KEY,
    Deduction_Type VARCHAR(255),
    Deduction_Date DATE,
    Deduction_Amount DECIMAL(10,2)
);
```

Payroll_Fact Table

```
CREATE TABLE Payroll_Fact (
    Employee_ID INT,
    Department_ID INT,
    Designation_ID INT,
    Overtime_ID INT,
    Deductions_ID INT,
    Date_ID INT,
    Gross_Salary DECIMAL(10,2),
    Avg_Base_Pay DECIMAL(10,2),
```

```
    Overtime_Pay DECIMAL(10,2),
    Deductions DECIMAL(10,2),
    Number_of_Employees INT,
    PRIMARY KEY (Employee_ID, Date_ID),
    FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID),
    FOREIGN KEY (Department_ID) REFERENCES Department(Department_ID),
    FOREIGN KEY (Designation_ID) REFERENCES Designation(Designation_ID),
    FOREIGN KEY (Overtime_ID) REFERENCES Overtime(Overtime_ID),
    FOREIGN KEY (Deductions_ID) REFERENCES Deductions(Deductions_ID),
    FOREIGN KEY (Date_ID) REFERENCES Date(Date_Key)
);
```

Postgres Implementaion Screenshot: