School of Computer Science, UPES, Dehradun.

A

LABORATORY FILE

On

DATABASE MANAGEMENT SYSTEM (DBMS) LAB

B.TECH. -III Semester

AUG. – NOV.- 2024.

Submitted by:
Name: Harsh Vardhan Saini
SAP ID: 500120369
Roll No: R2142230056
Batch: 2

Experiment 10

To understand the concepts of Views.

Aim:

To create and manage an employee database, including functionality for inserting, updating, and deleting employee records, as well as creating views for simplified data access.

Problem Statement:

The task involves setting up a database to track employee information, such as names, dates of birth, salaries, and department affiliations. The goal is to facilitate efficient data retrieval and manipulation, while also creating views that allow for more straightforward access to specific employee data.

Theory:

Relational databases organize data in structured tables, enabling relationships and efficient querying. SQL (Structured Query Language) is used for defining and manipulating this data. Key concepts include:

- Tables: Used to store data in rows and columns.
- Primary Keys: Unique identifiers for records in a table.
- Views: Virtual tables created from SQL queries that simplify access to data.
- Data Manipulation: Using `INSERT`, `UPDATE`, and `DELETE` commands to modify data in the database.

Commands Used:

1. Database and Table Creation:
2. Data Insertion:
3. Creating Views:
4. Modifying Table Structure:

5. Updating Records via a View:

6. Deleting Records via a View:

7. Creating Another View:

8. Viewing the Data:

9. Dropping a View:

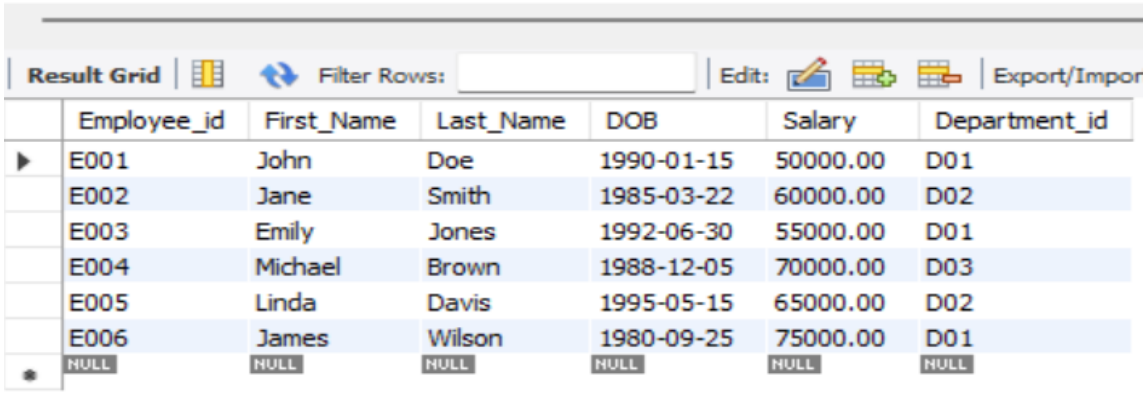
Results:

```
1   -- Ayush Vashishth
2   -- 500119331
3
4   CREATE DATABASE exp10;
5   USE exp10;
6
7   -- Creating the EMPLOYEES Table
8   CREATE TABLE EMPLOYEES (
9       Employee_id CHAR(10) PRIMARY KEY,
10      First_Name CHAR(30) NOT NULL,
11      Last_Name CHAR(30) NOT NULL,
12      DOB DATE,
13      Salary DECIMAL(10, 2) NOT NULL,  -- Using DECIMAL to handle salaries with two decimal places
14      Department_id CHAR(10)
15  );
16
17  -- Inserting values into the EMPLOYEES table
18  INSERT INTO EMPLOYEES (Employee_id, First_Name, Last_Name, DOB, Salary, Department_id) VALUES
19  ('E001', 'John', 'Doe', '1990-01-15', 50000.00, 'D01'),
20  ('E002', 'Jane', 'Smith', '1985-03-22', 60000.00, 'D02'),
21  ('E003', 'Emily', 'Jones', '1992-06-30', 55000.00, 'D01'),
22  ('E004', 'Michael', 'Brown', '1988-12-05', 70000.00, 'D03'),
23  ('E005', 'Linda', 'Davis', '1995-05-15', 65000.00, 'D02'),
24  ('E006', 'James', 'Wilson', '1980-09-25', 75000.00, 'D01');
25
26  -- Creating a View named emp_view
27  CREATE VIEW emp_view AS
28  SELECT Employee_id, Last_Name, Salary, Department_id
```

```
29      FROM EMPLOYEES;
30
31      -- You cannot directly insert into a view like this unless you are inserting into an updatable view that maps directly to a base table.
32      -- Remove the insert into view since it will cause errors.
33
34      -- If you need to modify the Salary column to allow NULL values, you'd do the following:
35      ALTER TABLE EMPLOYEES MODIFY Salary DECIMAL(10, 2) NULL;
36
37      -- Now, you can insert a row with a NULL salary
38      INSERT INTO EMPLOYEES (Employee_id, First_Name, Last_Name, DOB, Salary, Department_id) VALUES
39      ('E007', 'Chris', 'Anderson', NULL, NULL, 'D01');
40
41      -- Update operations on the View (affects the base table EMPLOYEES)
42      UPDATE emp_view
43      SET Salary = 80000.00
44      WHERE Employee_id = 'E001';
45
46      -- Delete an employee from the view (and consequently from the EMPLOYEES table)
47      DELETE FROM emp_view
48      WHERE Employee_id = 'E003';
49
50      SELECT * FROM emp_view;
51      -- Dropping the emp_view
52      DROP VIEW emp_view;
53
54      -- Create a View named salary_view to show annual salary for employees in Department D02
55      CREATE VIEW salary_view AS
56      SELECT Employee_id, Last_Name, Salary * 12 AS Annual_Salary
```

```
56      SELECT Employee_id, Last_Name, Salary * 12 AS Annual_Salary
57      FROM EMPLOYEES
58      WHERE Department_id = 'D02';
59
60      -- View the salary_view
61      SELECT * FROM salary_view;
```

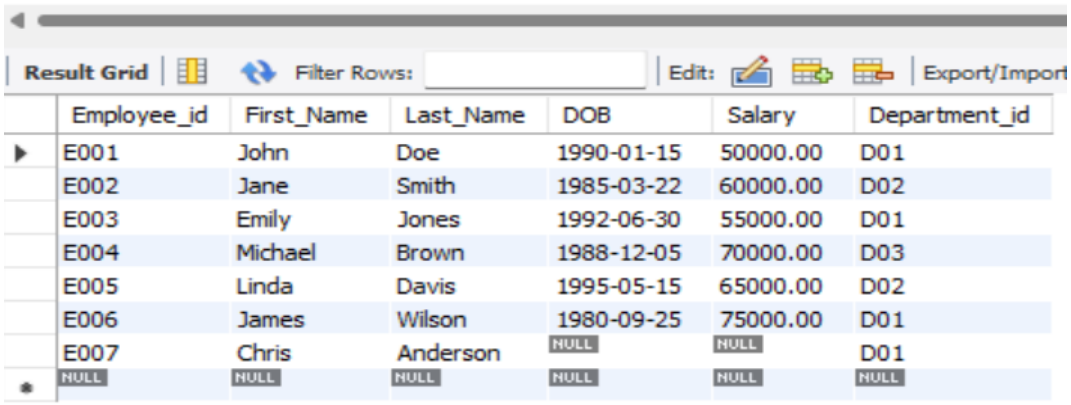
```
1 •  SELECT * FROM exp10.employees;
```



Result Grid | Filter Rows: | Edit: | Export/Impor

Employee_id	First_Name	Last_Name	DOB	Salary	Department_id
E001	John	Doe	1990-01-15	50000.00	D01
E002	Jane	Smith	1985-03-22	60000.00	D02
E003	Emily	Jones	1992-06-30	55000.00	D01
E004	Michael	Brown	1988-12-05	70000.00	D03
E005	Linda	Davis	1995-05-15	65000.00	D02
E006	James	Wilson	1980-09-25	75000.00	D01
NULL	NULL	NULL	NULL	NULL	NULL

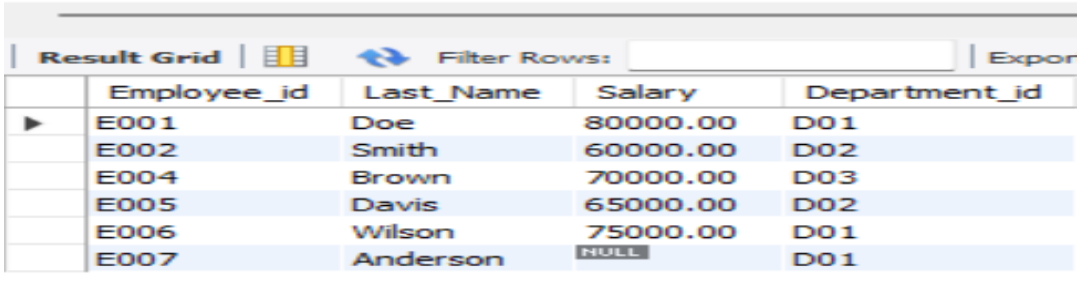
```
1 •  SELECT * FROM exp10.employees;
```



Result Grid | Filter Rows: | Edit: | Export/Import

Employee_id	First_Name	Last_Name	DOB	Salary	Department_id
E001	John	Doe	1990-01-15	50000.00	D01
E002	Jane	Smith	1985-03-22	60000.00	D02
E003	Emily	Jones	1992-06-30	55000.00	D01
E004	Michael	Brown	1988-12-05	70000.00	D03
E005	Linda	Davis	1995-05-15	65000.00	D02
E006	James	Wilson	1980-09-25	75000.00	D01
E007	Chris	Anderson	NULL	NULL	D01
NULL	NULL	NULL	NULL	NULL	NULL

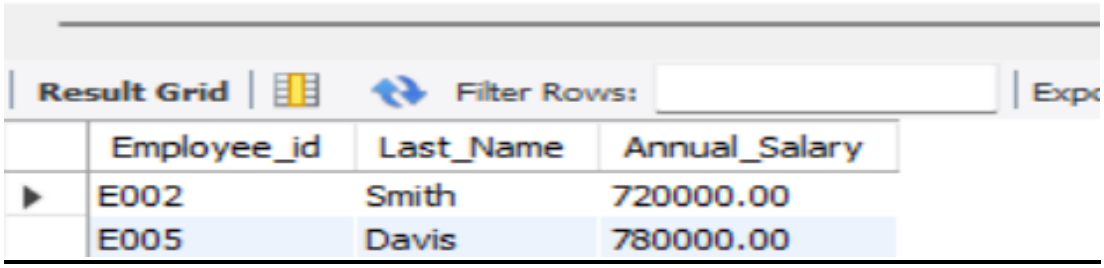
```
1 •  SELECT * FROM exp10.emp_view;
```



Result Grid | Filter Rows: | Expor

Employee_id	Last_Name	Salary	Department_id
E001	Doe	80000.00	D01
E002	Smith	60000.00	D02
E004	Brown	70000.00	D03
E005	Davis	65000.00	D02
E006	Wilson	75000.00	D01
E007	Anderson	NULL	D01

```
1 •    SELECT * FROM exp10.salary_view;
```



Result Grid | Filter Rows: | Expo

	Employee_id	Last_Name	Annual_Salary
▶	E002	Smith	720000.00
	E005	Davis	780000.00

Conclusion:

The SQL code effectively sets up a relational database to manage employee information, including functionalities for inserting, updating, and deleting records. The creation of views enhances data accessibility by allowing users to easily retrieve specific employee data without dealing with the underlying table directly.

The use of a view for annual salaries provides a clear example of how to present calculated data, facilitating reporting and analysis. Overall, this database design supports efficient employee management and can be expanded further with additional features such as more complex views or stored procedures for automated reporting.

Future enhancements could include adding indexes for faster querying, more detailed employee attributes, or implementing stored procedures for common operations to streamline data management tasks.