|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 01 | **DATA DEFINITION LANGUAGE, DATA MANIPULATION LANGUAGE COMMANDS** |
| **DATE** | **:** 15.02.2024 |

**AIM**

To define and manage the structure of the database objects using Data Definition language commands and Data Manipulation commands in Oracle database software using SQLPlus tool.

**SYNTAX**

1. CREATE TABLE table\_name (column1 datatype(size) , column2 datatype(size) … column datatype(size));
2. INSERT INTO table\_name VALUES (values1, values2,… valuesn);
3. ALTER TABLE tablename
4. MODIFY column\_name datatype(size);
5. RENAME COLUMN old\_column\_name TO new\_column\_name;
6. ADD column\_name datatype(size);
7. DROP column\_name;

iv) SELECT [\*][column(s)] FROM table\_name;

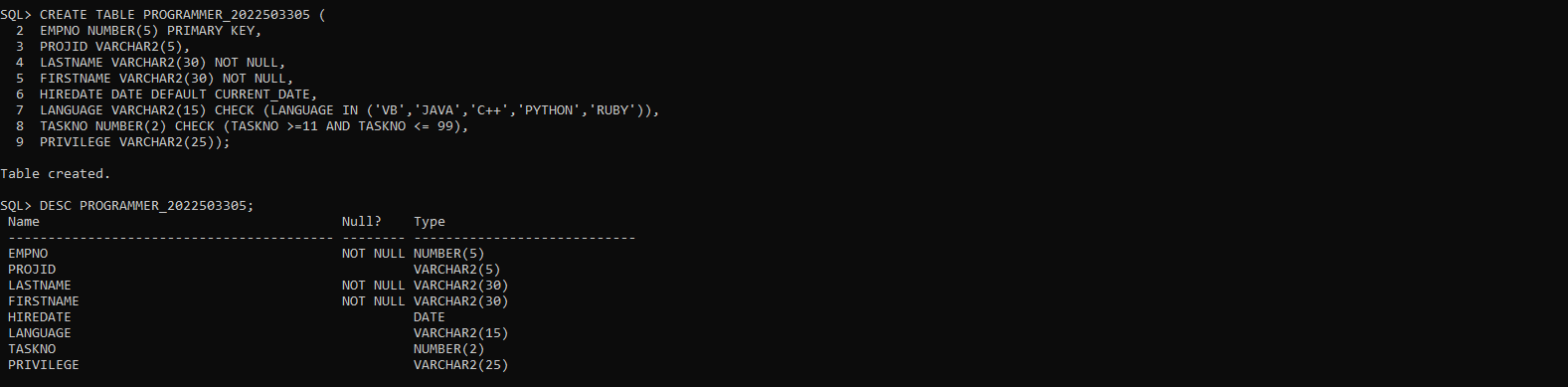
v) DESC table\_name;

vi) UPDATE table\_name SET column\_name = values;

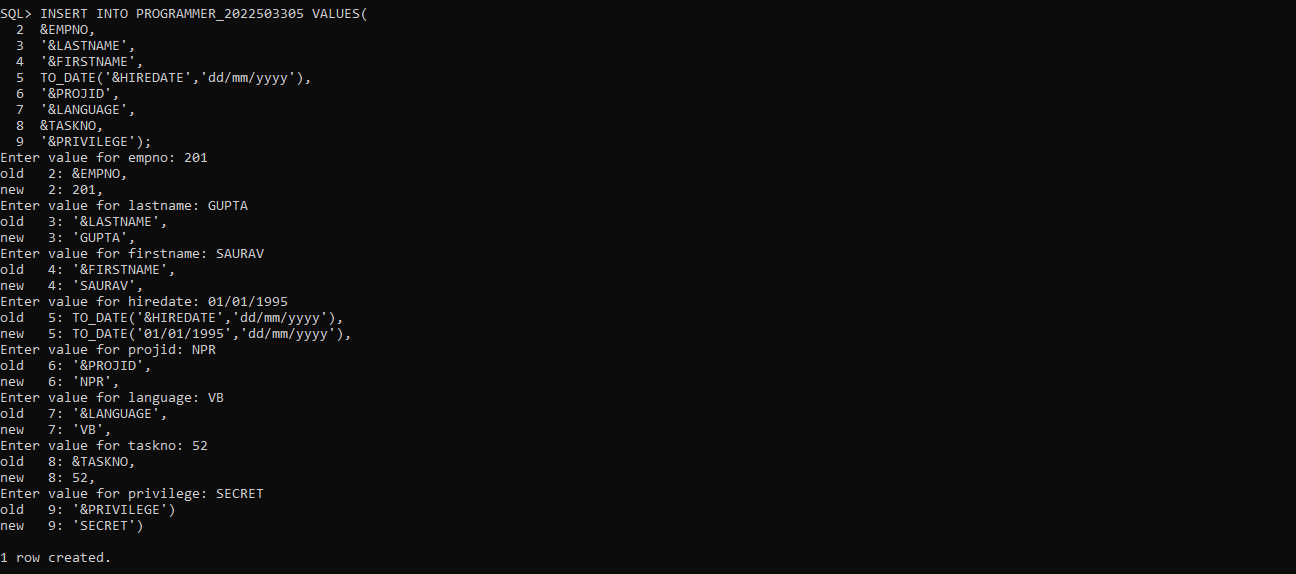
vii) RENAME TABLE old\_table\_name TO new\_table\_name;

viii) TRUNCATE TABLE table\_name;

**OUTPUT**



2.



**QUESTIONS**

1. Create the table with the constraint specified
2. Insert the values using static and dynamic method.

3. Insert an employee with

EmpNo 896,Last Name=’Dilip’FirstName=’Kumar’HireDate=’08-JAN-1997’

projid=’Rnc’

Language=Python

TaskNo=12

Privilige=secret.

Check the integrity constraint violation and report the same.

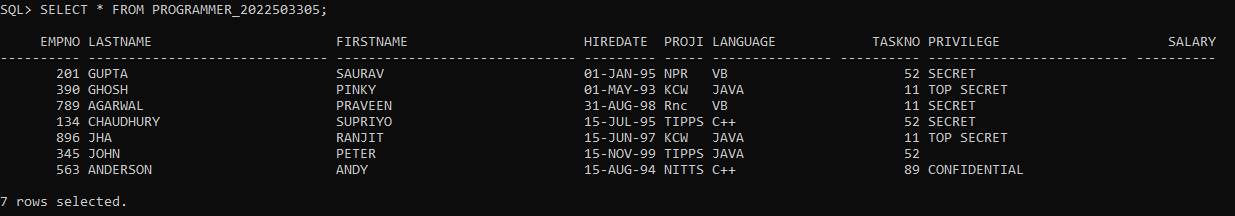
1. Add the column salary. Max size 10 and 4 places of decimal
2. Modify the column salary. Max size is 12 and 4 places of decimal.
3. Update the column salary based on privilege
4. Modify the column EmpNo. Change the maximum size to 10. Report if any error occurs.
5. Modify the column Privilege. Set the constraint as NOT NULL. Report if any error occurs.
6. Rename the column Language to Prog\_language
7. Update the language for an employee with EmpNo 896to "Python"
8. Increase the salary of all employees by 10%
9. Set the hire date to the current date for all employees hired after 08/30/98
10. Update the Privilegeto TopSecretfor employees hired between1/1/98 to 1/1/2000
11. Update the Language for employees whose last names containsvowelsto "Go":
12. Update theLanguagecolumn to "Swift" for rows where theLastNamecolumn has a length of exactly 5 characters
13. Drop the column EmpNo. Check the integrity constraint violation and report the same.
14. The column salaryin the PROGRAMMER Table is no longer needed. Delete the column.

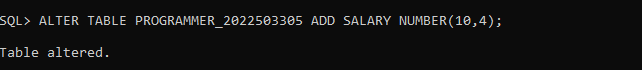


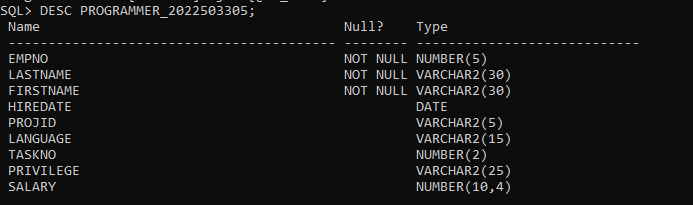
` 3.



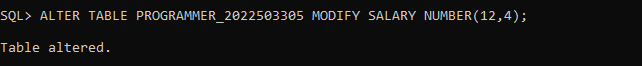
1. Create a table Department with Dno(charactertype), dname (character of 25) , location (character of 25)
2. Insert three rows with dno D101, D102, D103 and include other field values as your choice.
3. Add a column deptid in the programmer table as foreign key.
4. Insert the deptid value as D105 in Programmer table and validate the same. Report if any error occurs.
5. Create a duplicate table with the same structure and dataHint:create table new\_table asselect \* from existing\_table;
6. Delete all the entries in the table but retain the structure of the table.
7. Drop the table Department as it not needed. Report if any constraint violation occurs.
8. Drop the table programmeras it not needed. Report if any constraint violation occurs.

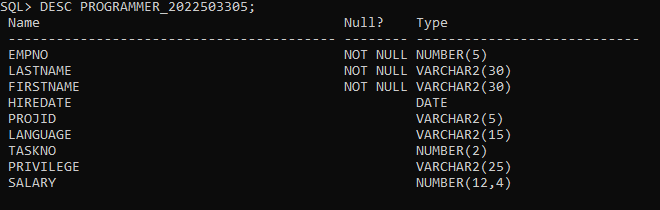


4. 

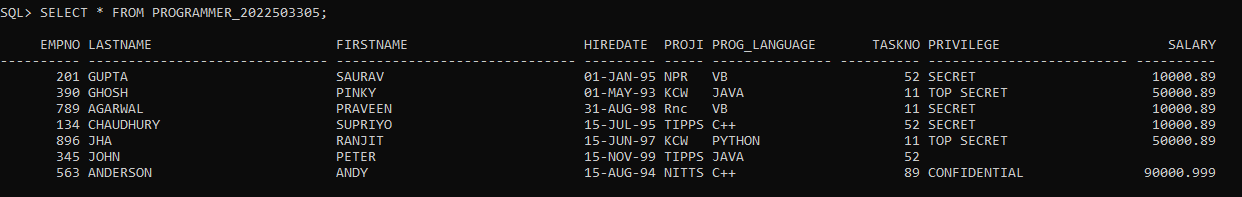


5.

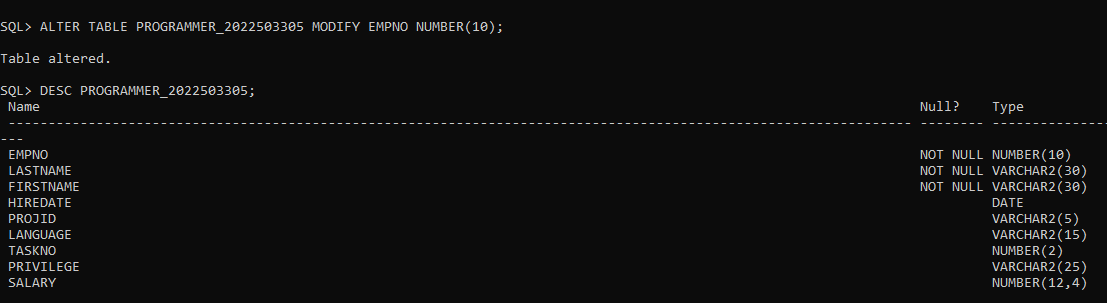




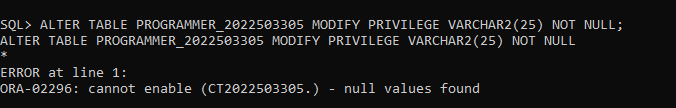
6.



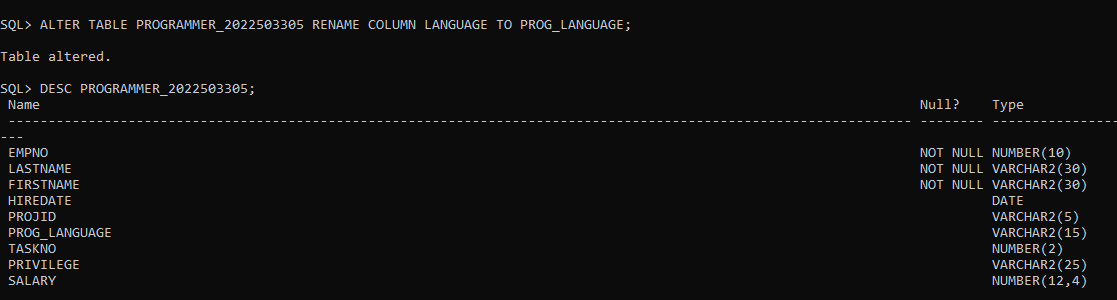
7.



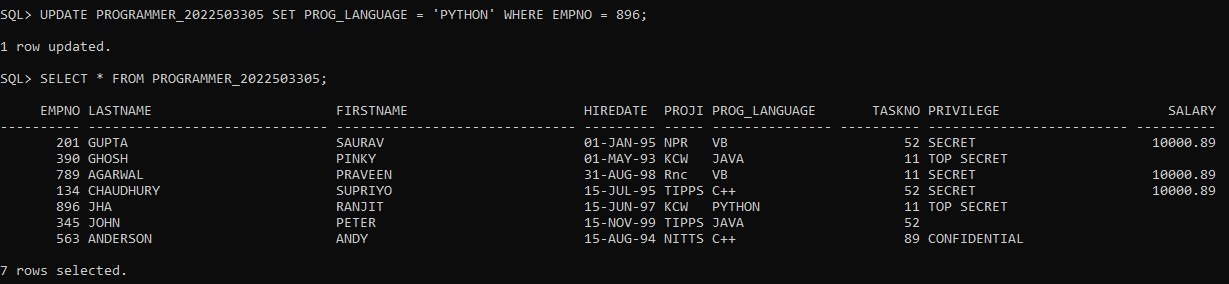
8.



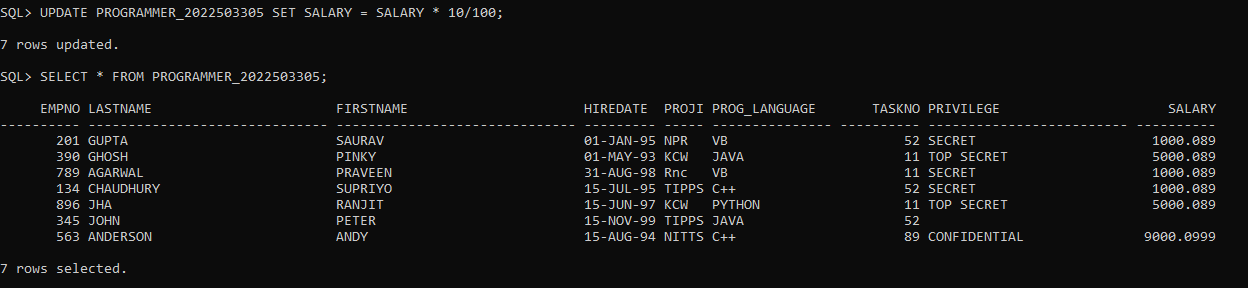
9.



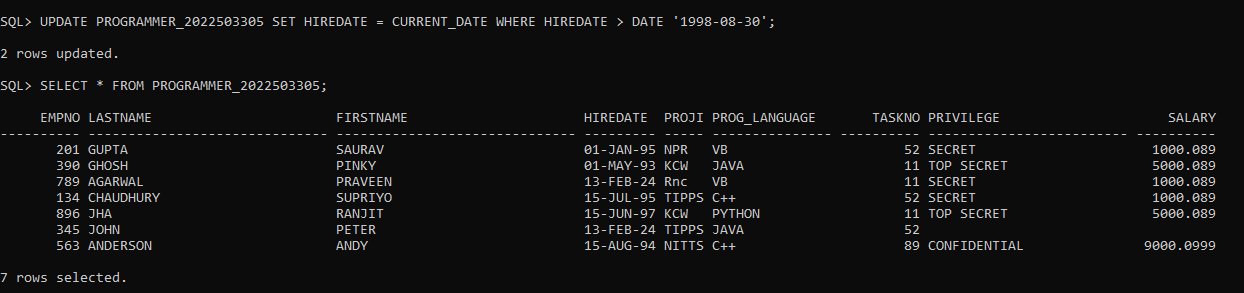
10.



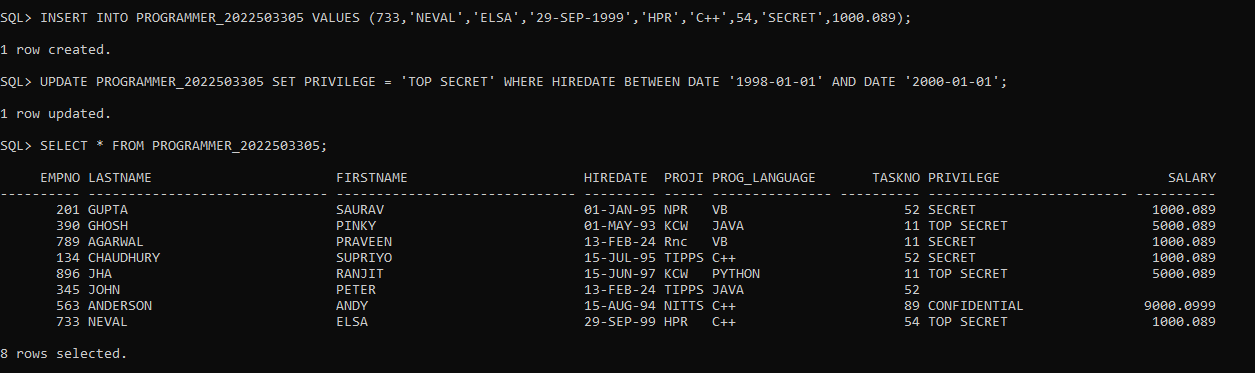
11.



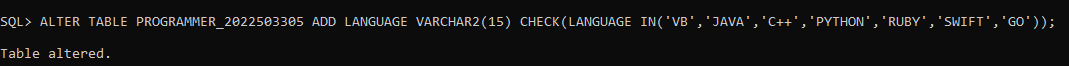
12.

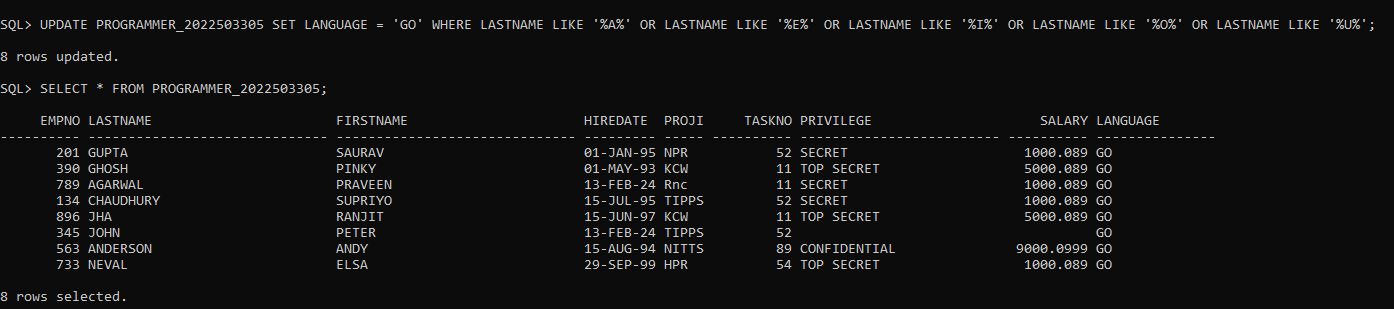


13.

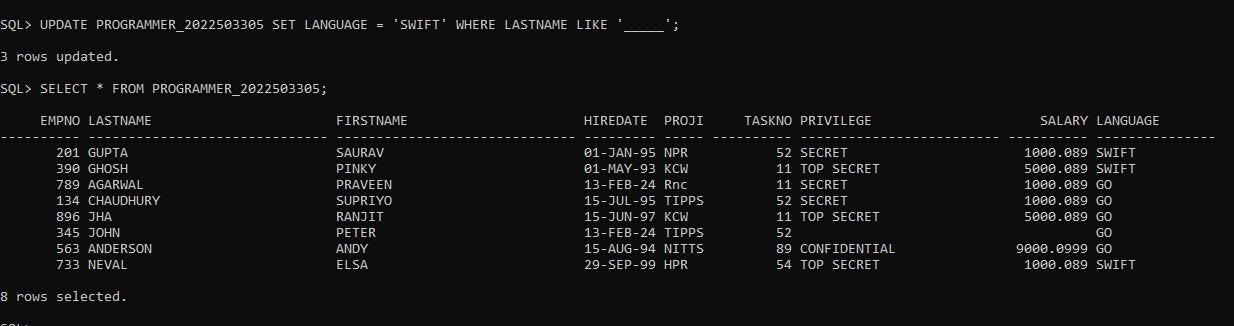


14.

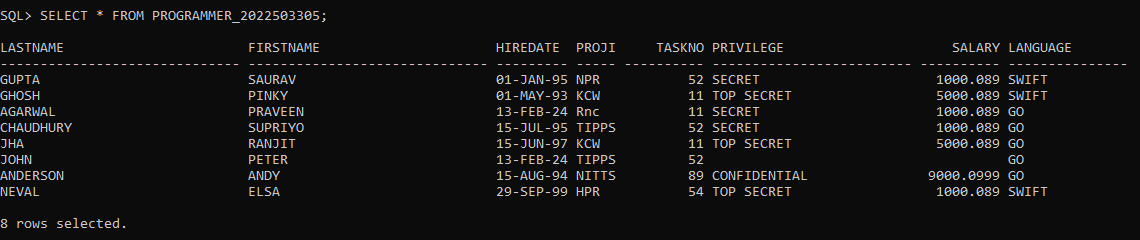




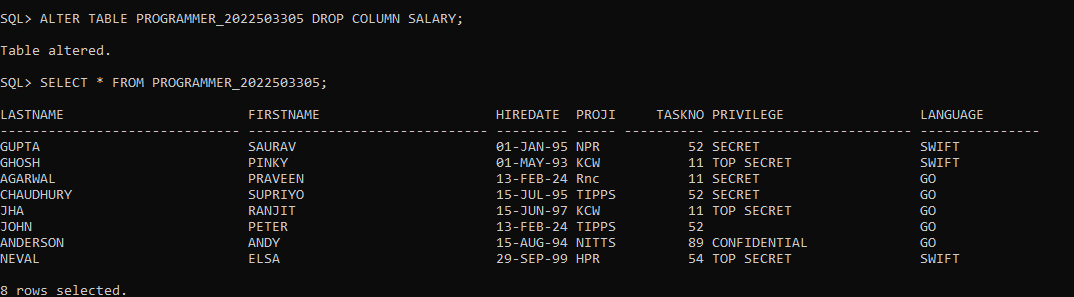
15.



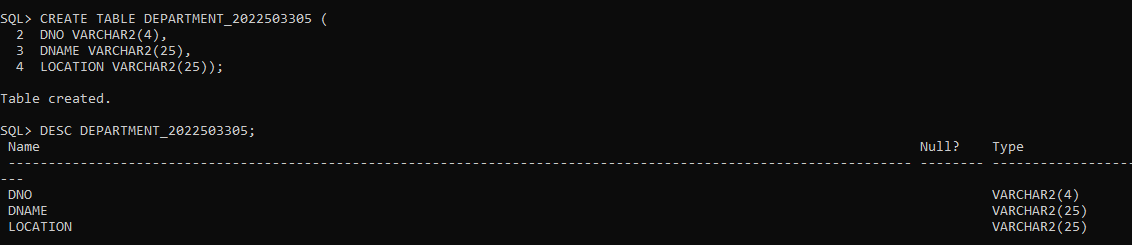
16.



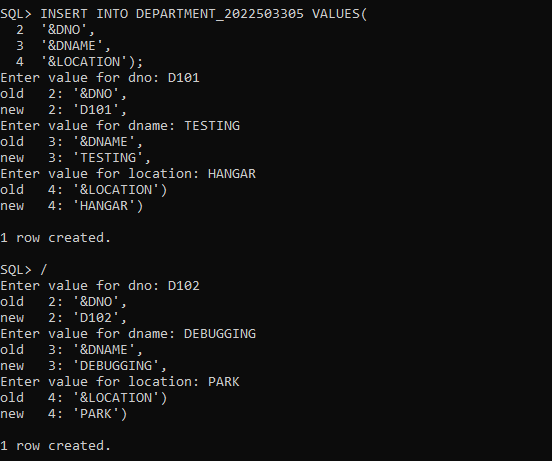
17.

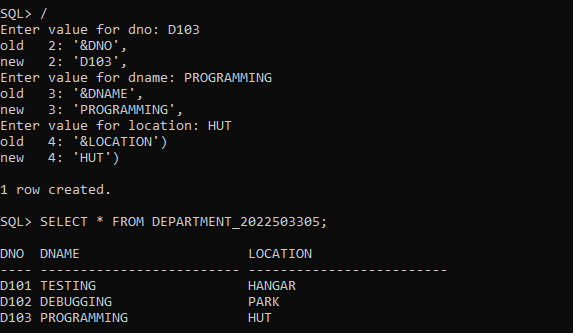


18.

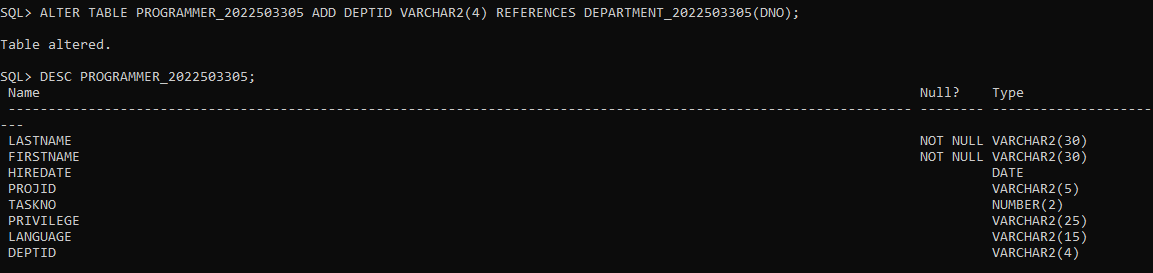


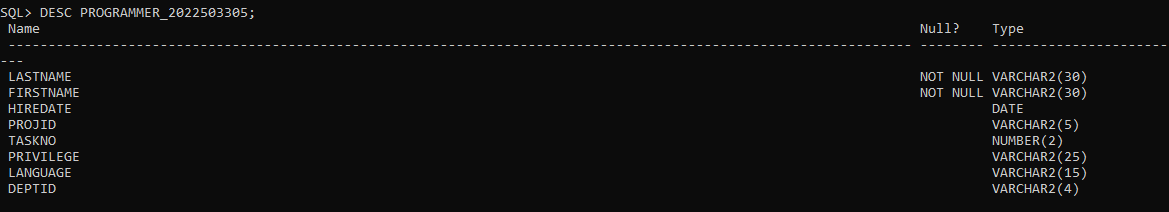
19

. 

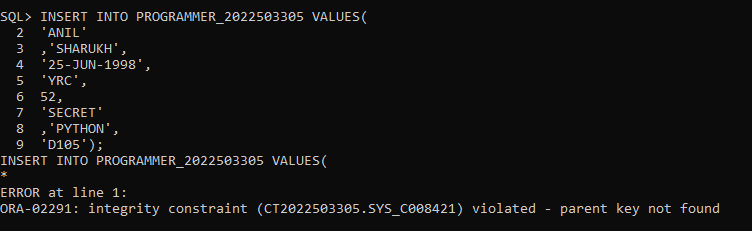


20.

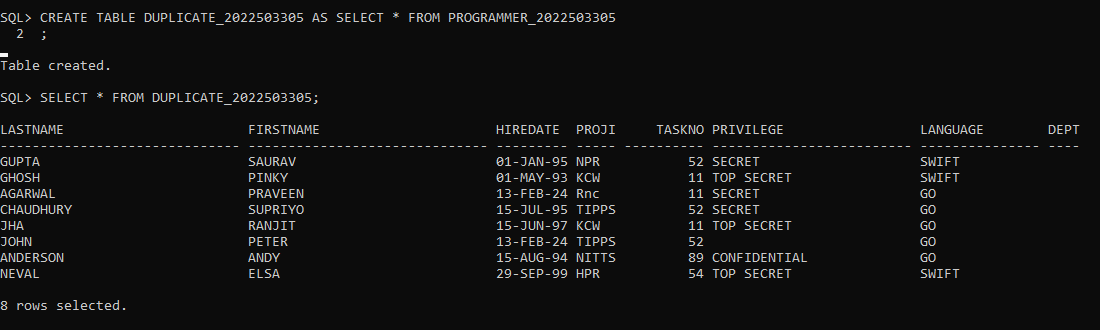




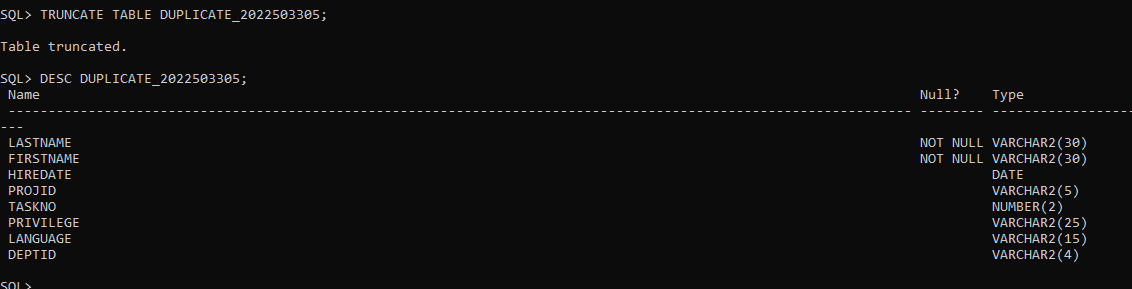
21.



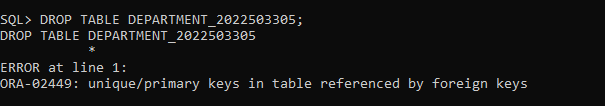
22.



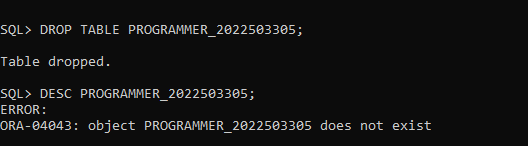
23.



24.



25.



**RESULT**

Thus, managing the structure of the database objects using Data Definition language commands has been executed successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 02 | **ER DIAGRAM** |
| **DATE** | **:** 19.02.2024 |

**AIM**

To draw the ER diagram for the given scenarios.

**PROCEDURE:**

**STEP 1:** Go to draw.io website in you search engine.

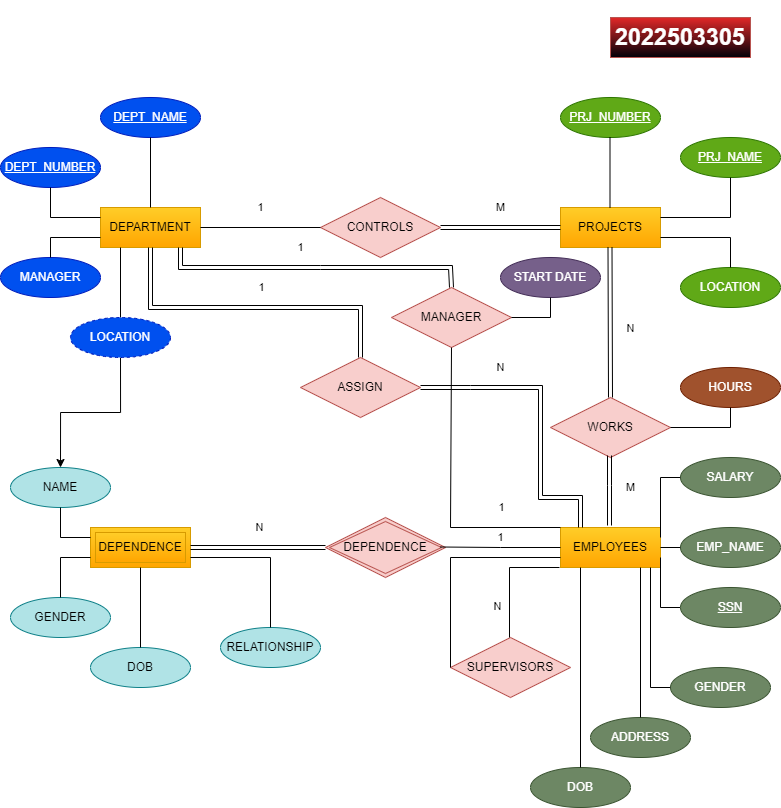
**STEP 2:** Place the necessary entity relationship shapes such as rectangle, oval etc.

**STEP 3:** Name the symbols.

**STEP 4:** Connect the symbols using the connections in the menu bar.

**STEP 5:** Add some styles to the entity relationship shapes and the add some text styles.

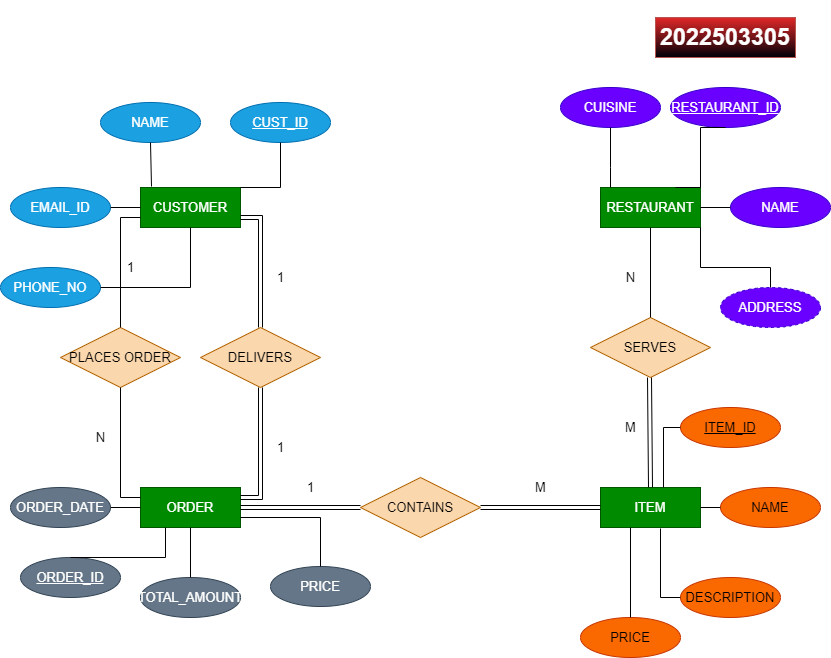
**STEP 6:** Screenshot the output

****

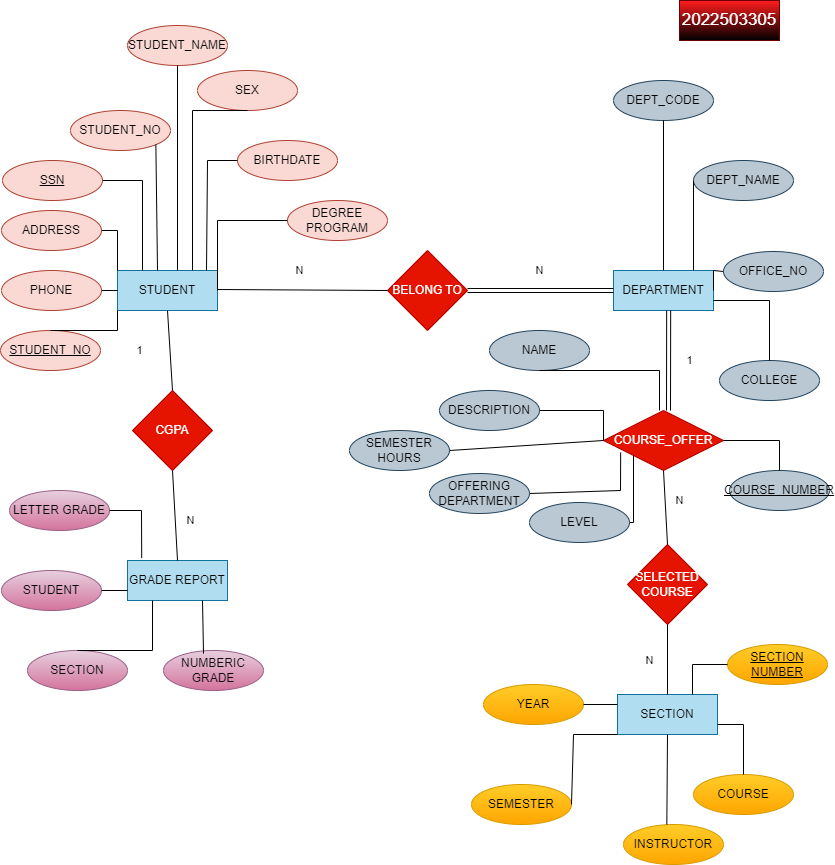
**QUESTIONS**

1. The company is organized into departments. Each department has a name, an unique number, and a particular employee who manages the department. We keep track of the start date when the employee began managing the dept. A dept may have several locations. A dept controls a number of projects, each of which has a name, unique number, andsinglelocation.Westoreeachemployee’sname,SSN,address,salary,sex,and DOB.

An employee is assigned to one dept but may work onseveralprojects,whicharenotnecessarycontrolledbythesamedept.Wekeeptrackofthenumberofhoursperweekthat an employee works for each project. We also keep track ofthe direct supervisor of each employee. We want to keep trackof the dependents of each employee for insurance purpose. We keep each dependent’s first name sex, DOB, and relationship totheemployee.

****

1. Customer represents individual users who place orders on Swiggy. Each customer is defined by CustomerID, Name, Email, and Phone Number. Each customer can place multiple orders, but each order is placed by exactly one customer.
2. Restaurant represents the restaurants partnered with Swiggy. Each restaurant is described by the RestaurantID, Name, Address, Cuisine. Each restaurant can serve multiple food items, but each food item is served by exactly one restaurant.
3. Order represents individual orders placed by customers. Each order is identified by a unique OrderID and has attributes such as Order Date and Total Amount. Each order can have one delivery,and each delivery corresponds to exactly one order.
4. Item represents individual food items available for order. Each item is described by the ItemID and has attributes such as Name, Description and Price. Each order can contain multiple items, and each item can be part of multiple orders.

****

1. ConsiderthefollowingsetofrequirementsforaUNIVERSITYdatabasethatisused to keep track of students 'transcripts.
2. The university keeps track of each student's name, student number, social security number, current address and phone, permanent address and phone, birthdate, sex, class (freshman,sophomore,...,graduate),majordepartment,minordepartment (if any), and degree program (B.A., B.S., ..., Ph.D.).Some user applications need to refer to the city, state, and zip of the student's permanent address, and to the student's last name. Both social security number and student number have unique values for each student.
3. Each department is described by a name, department code, office number, office phone, and college. Both name and code have unique values for each department.
4. Each course has a course name, description, course number, number of semester hours, level, and offering department. The value of course number is unique for each course.
5. Each section has an instructor, semester, year, course, and section number. The section number distinguishes different sections of the same course that are taught during the same semester/year; its values are 1, 2, 3, ..., up to the number of sections taught during each semester.
6. A grade report has a student, section, letter grade, and numeric grade

(0,1,2,3,4 for F, D, C, B, A, respectively).

**RESULT**

Thus, the has been drawn and the necessary styles has been given successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 03 | **DATA QUERY LANGUAGE** |
| **DATE** | **:** 20.02.2024 |

**AIM**

To retrieve the records in the tables from the database using the Data Query Language.

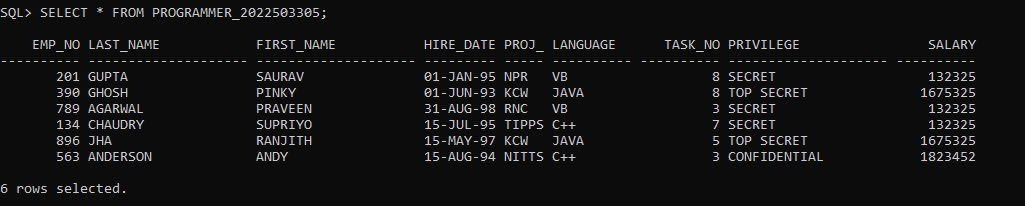
**SYNTAX**

1. SELECT \* FROM table\_name;
2. SELECT DISTINCT column\_names FROM table\_name;
3. SELECT columnname,COUNT(column\_name) FROM table\_name GROUP BY column\_name;
4. SELECT column\_names FROM table\_name;
5. SELECT column\_name AS another\_column\_name FROM table\_name;
6. SELECT column\_name FROM table\_name WHERE column\_name = value;

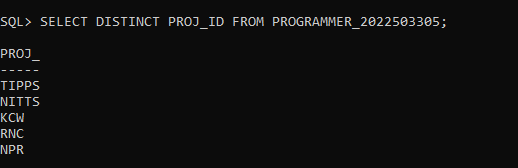
**QUESTIONS**

1. Write a query to retrieve all columns from the 'Programmer' table.
2. Write a query to retrieve the unique ProjId present in the 'Programmer' table
3. Write a query to retrieve total number of employees in the 'Programmer' table."
4. Write a query to retrieve the 'LastName' and 'FirstName' columns from the 'Programmer' table."
5. Write a query to calculate the annual salary for each employee in the 'Programmer' table by multiplying the 'salary' column by 12.
6. Write a query to provide an alias 'full\_name' for the concatenation of 'FirstName' and 'LastName' columns in the 'Programmer' table."
7. Write a query to concatenate the 'FirstName' and 'LastName' columns with a space in between, and provide the result as 'Employee Name' from the 'Programmer' table."
8. Write a query to retrieve all rows from the 'Programmer' table where the 'Privilege' is 'Top Secret'."
9. Write a query to retrieve all rows from the 'Programmer' table where the 'HireDate' falls between '1995-01-01' and '1998-12-31'.

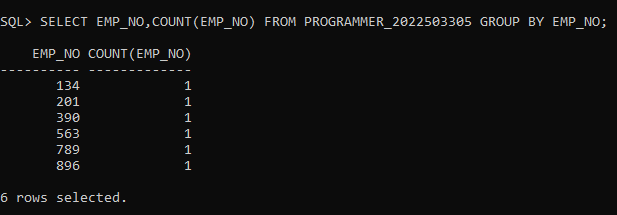
I.



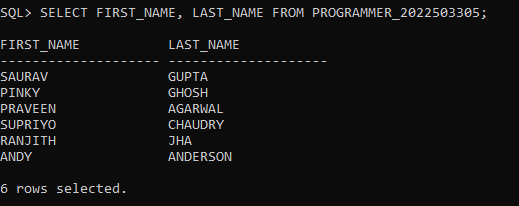
II.



III.

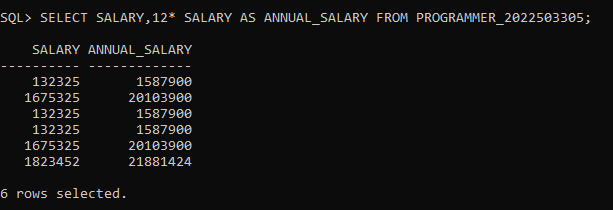


IV.

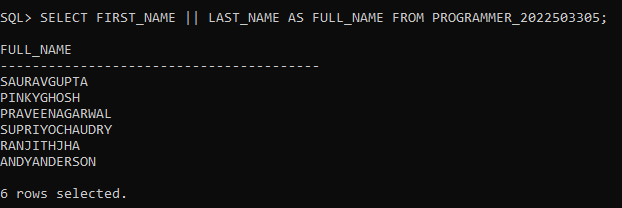


1. Write a query to retrieve all rows from the 'Programmer' table where the 'HireDate' falls in the month of ‘August'.
2. Write a query to retrieve all rows from the 'Programmer' table where the 'HireDate' falls in the month of ‘August', ’November’.
3. Write a query to retrieve all rows from the 'Programmer' table where the 'HireDate' falls in the month according to the user input.
4. Write a query to retrieve all rows from the 'Programmer' table where the 'salary' is greater than 50000.
5. Write a query to retrieve all rows from the 'Programmer' table where the 'salary' is greater than 50000 and the 'ProjId' is not ‘TIPPS’.
6. Write a query to retrieve all rows from the 'Programmer' table where the 'Language' is either 'Java' or 'C++'.
7. Write a query to retrieve all rows from the 'Programmer' table where the 'Privilege' is NULL."
8. Write a query to calculate the count of distinct entries of 'ProjId' column in the 'Programmer' table.
9. Write a query to retrieve details of employees from the 'Programmer' table who joined in either January, February, or March.
10. Retrieve the details of employees from the 'Programmer' table whose last name starts with 'S'.

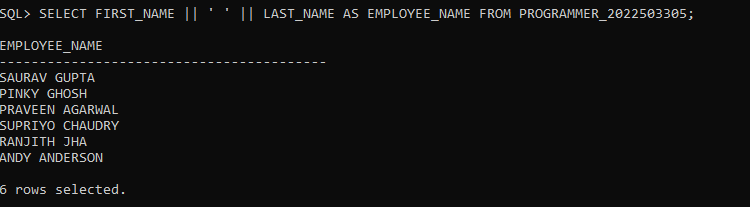
V.



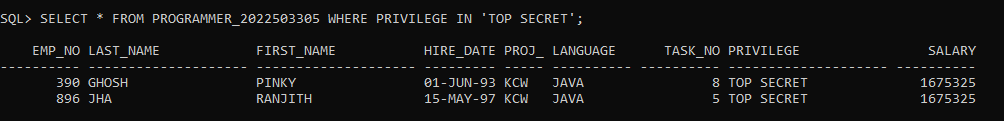
VI.



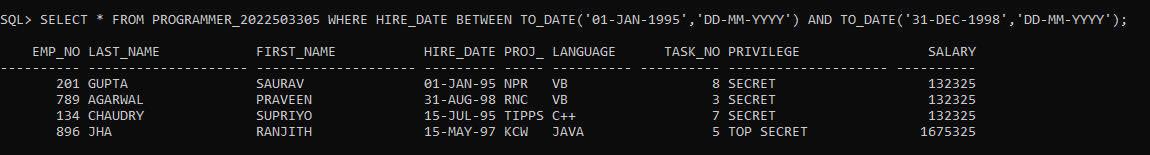
VII.



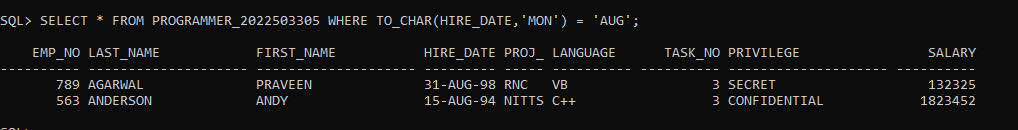
VIII.



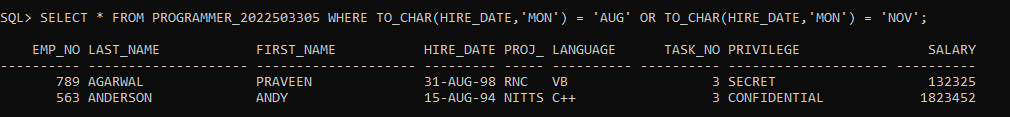
IX.



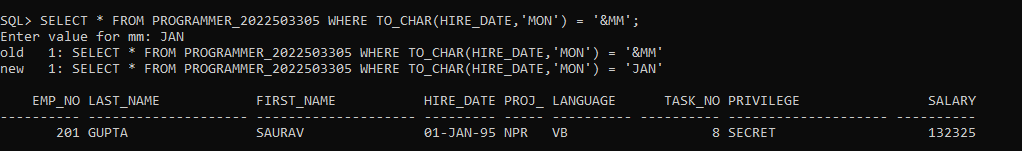
X.



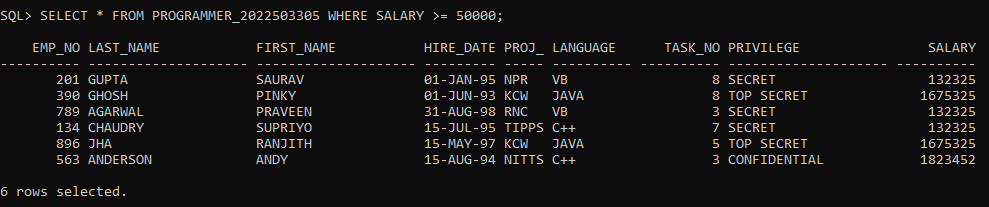
XI.



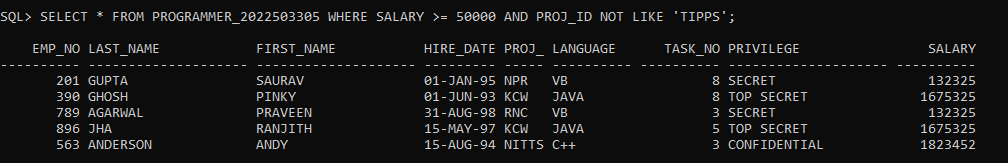
XII



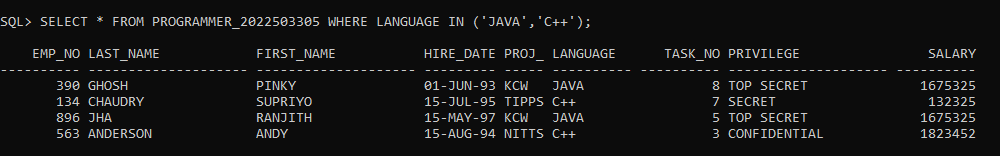
XIII



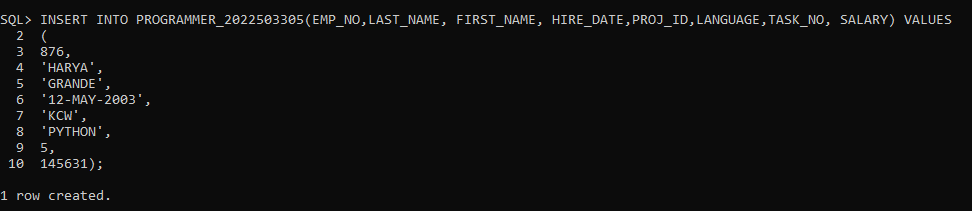
XIV.



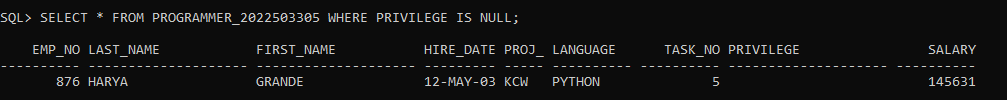
XV.



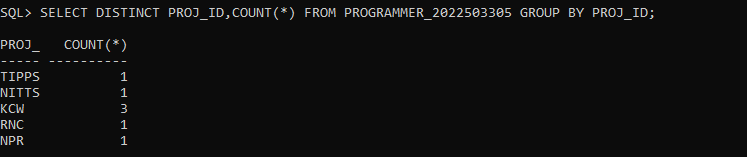
XVI - I



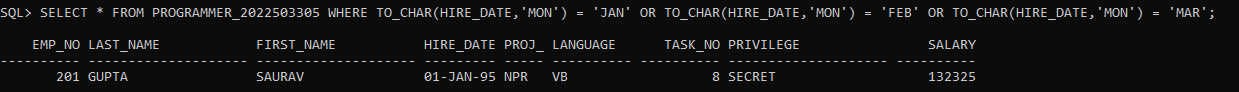
XVI - II.



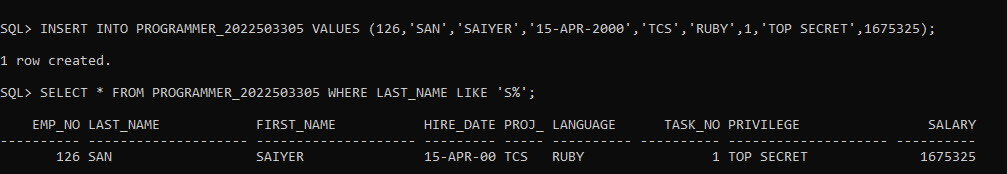
XVII.



XVIII.



XIX.



**RESULT**

Thus, retrieving the records in the tables from the database using the Data Query Language has been executed successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 04 | **ER - MAPPING** |
| **DATE** | **:** 15.02.2024 |

**AIM**

To map the ER – Relation for the given ER – Diagram.

**PROCEDURE**

**STEP 1:** Explore the given ER- Relational diagram.

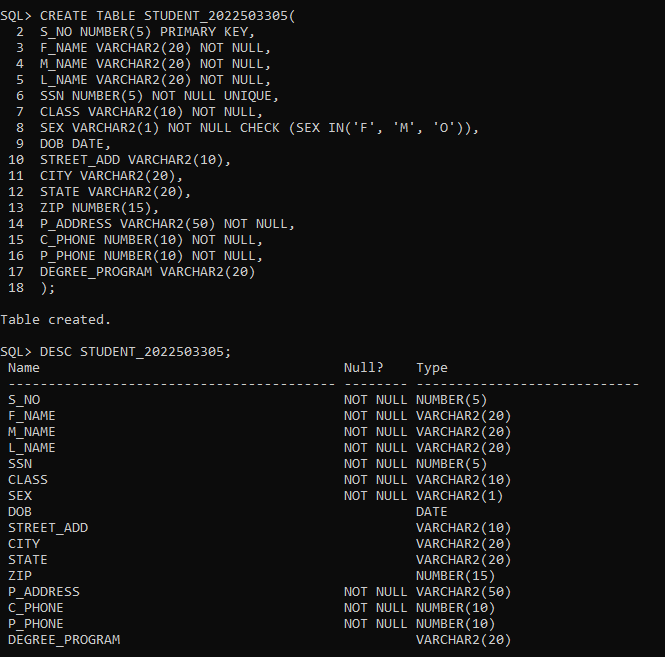
**STEP 2:** Find the tables in the diagram.

**STEP 3:** Create the tables with required constraints.

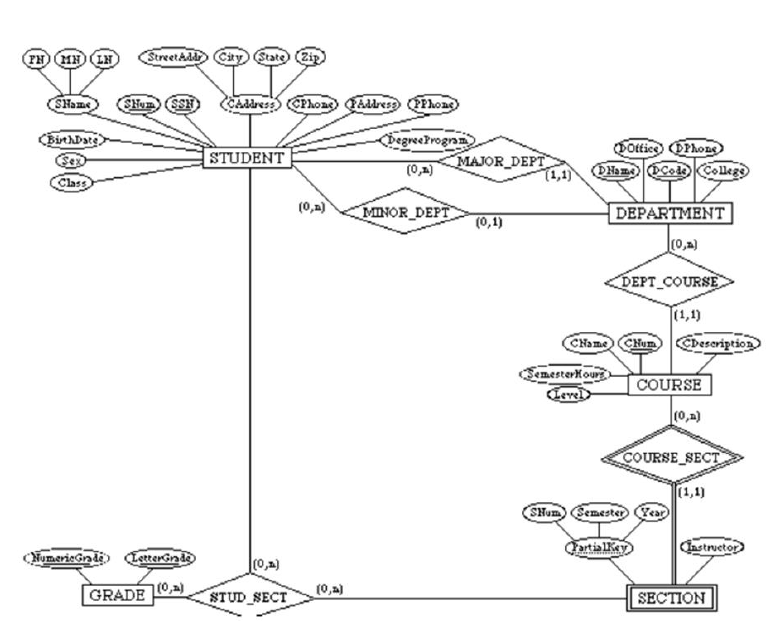
**STEP 4:** Use keys such as UNIQUE KEY, FOREIGN KEY, PRIMARY KEY.

**STEP 5:** Screenshot the outputs.

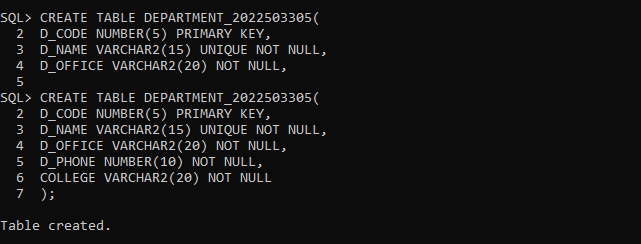
CREATING STUDENT TABLE

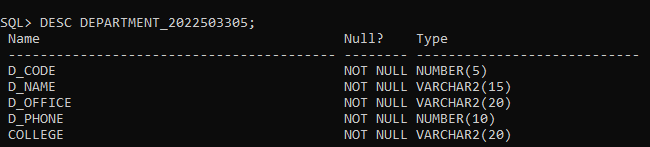


**QUESTIONS**

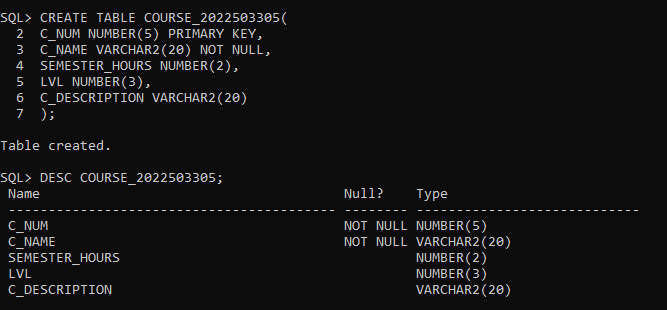


CREATING DEPARTMENT TABLE

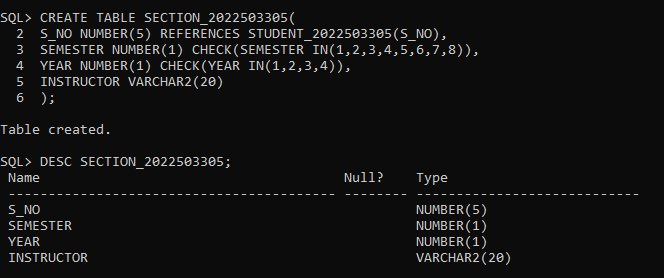




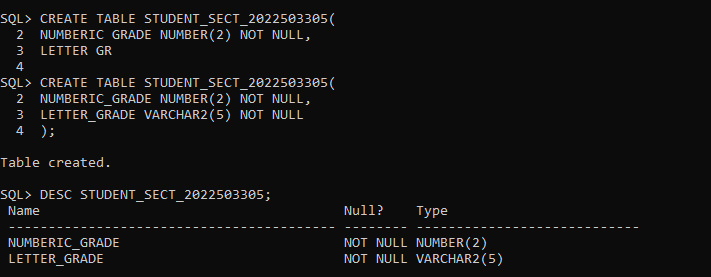
CREATING COURSE TABLE



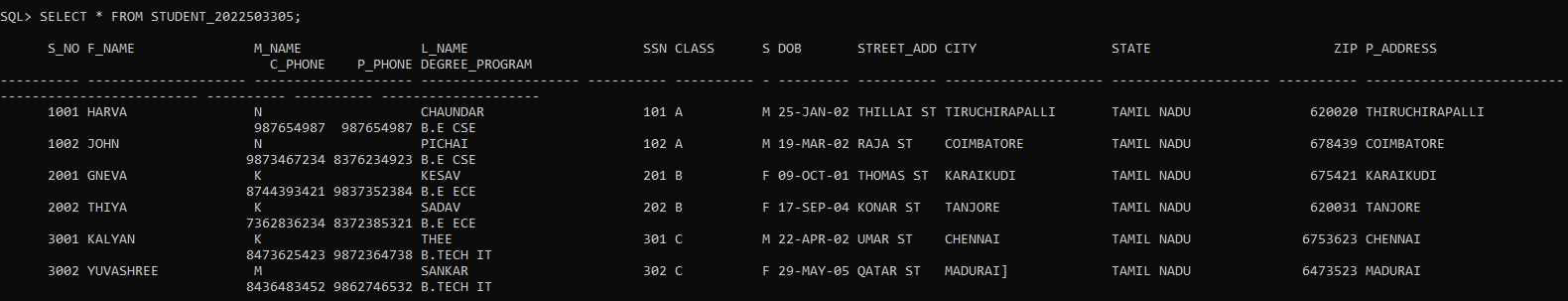
CREATING SECTION TABLE HAVING FOREIGN KEY

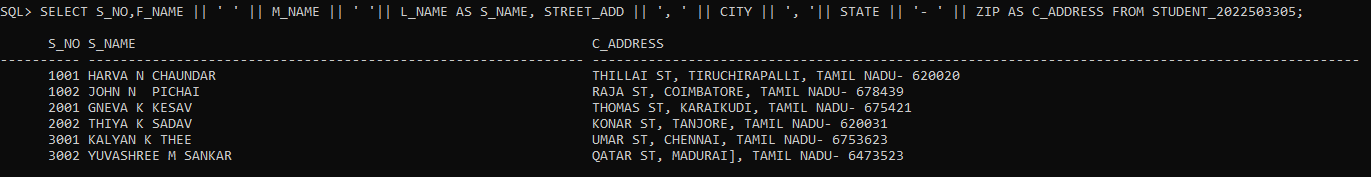


CREATING STUDENT SECTION TABLE

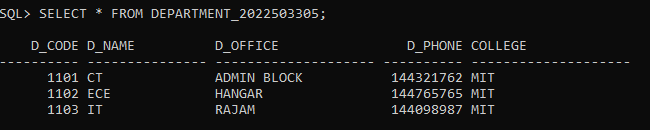


AFTER INSERTING VALUES IN THE STUDENT TABLE

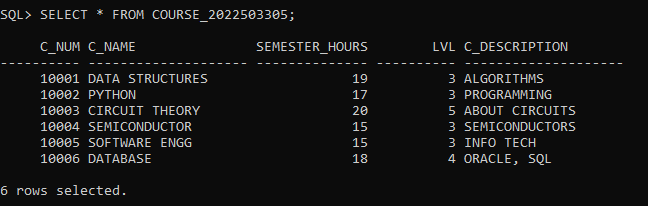




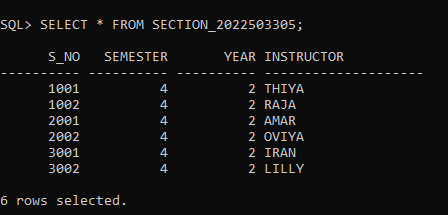
AFTER INSERTING VALUES IN THE DEPARTMENT TABLE



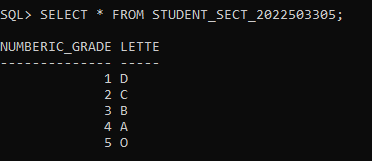
AFTER INSERTING VALUES IN THE COURSE TABLE



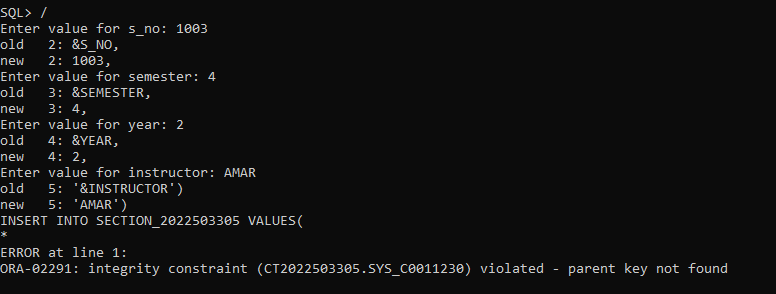
AFTER INSERTING VALUES IN THE SECTION TABLE



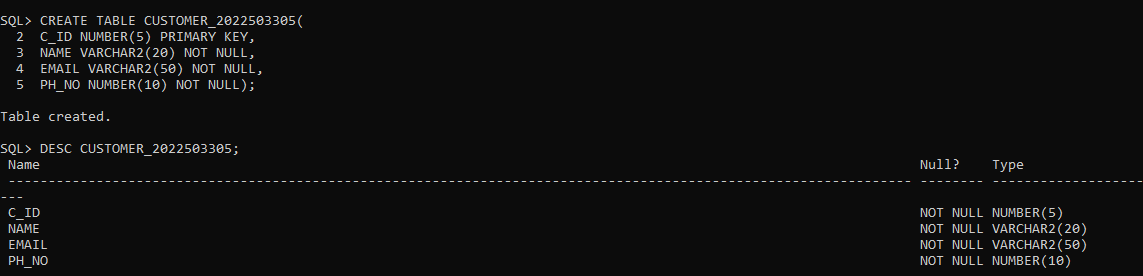
AFTER INSERTING VALUES IN THE STUDENT\_SECT TABLE



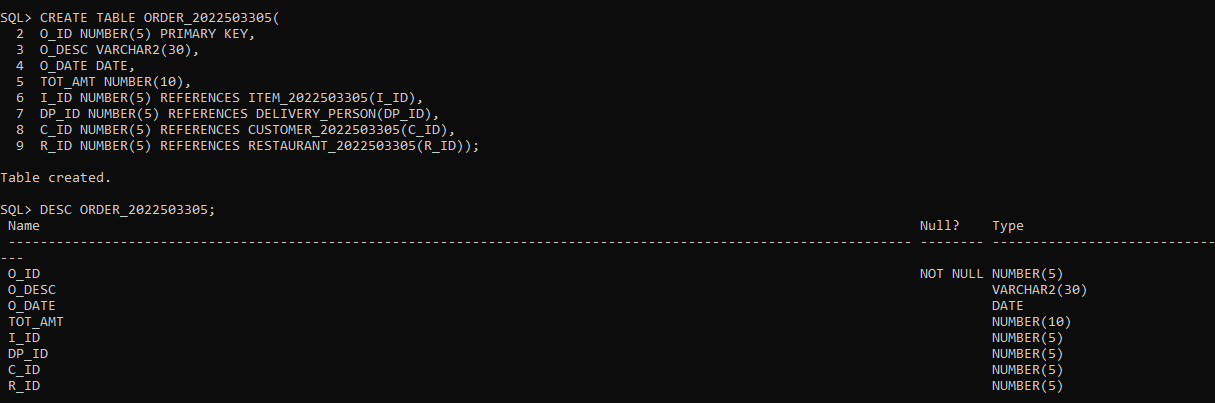
VIOLATING THE INTERGRITY CONSTRAINT BY ADDING THE UNKNOWN VALUE TO THE CHILD TABLE WHICH IS NOT AVAILABLE IN THE PARENT TABLE



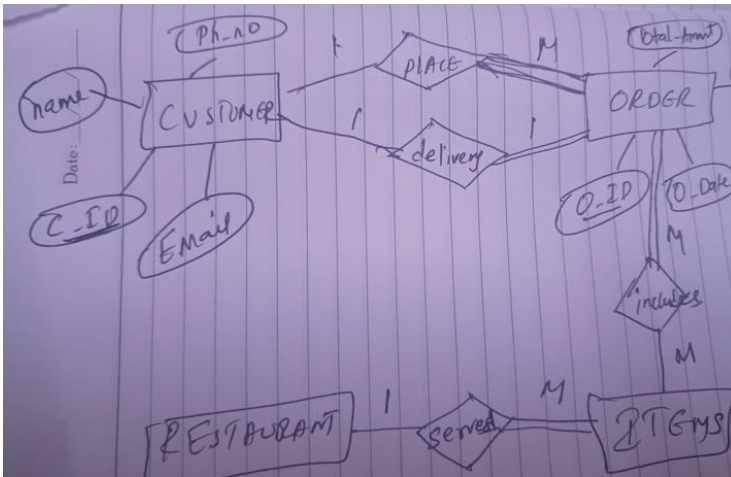
CREATING THE CUSTOMER TABLE



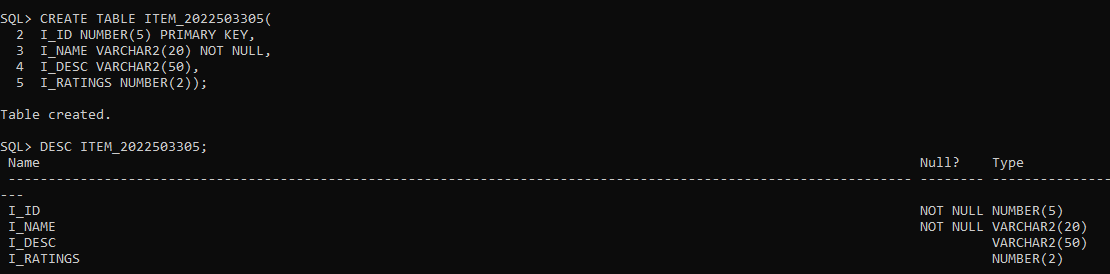
CREATING THE ORDER TABLE



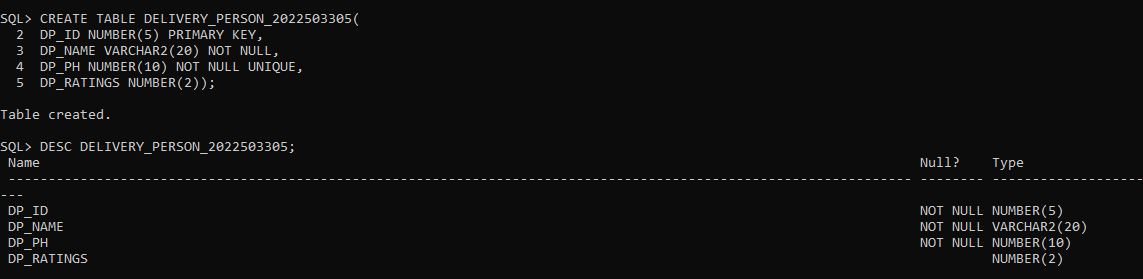
2.



CREATING THE ITEM TABLE



CREATING THE DELIVERY PERSONS TABLE



CREATING THE RESTAURANT TABLE

**RESULT**

Thus, ER-relational mapping for the given ER-Diagram has been done successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 05 | **DATA QUERY FUNCTIONS** |
| **DATE** | **:** 15.02.2024 |

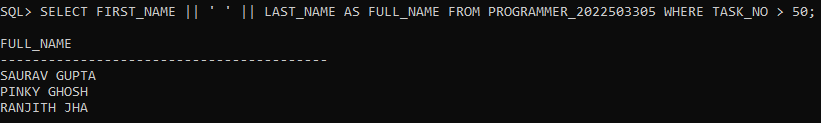
**AIM**

To retrieve the records in the tables from the database using Data Query Functions.

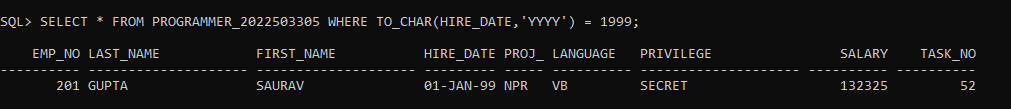
**SYNTAX**

1. SELECT column\_name1 || column\_name2 AS name FROM table\_name WHERE condition;
2. SELECT \* FROM table\_name TO\_CHAR(column\_name, YYYY/MM/DD) = value
3. SELECT DISTINCT column\_name FROM table\_name;
4. SELECT \* FROM table\_name;
5. SELECT column\_name FROM table\_name where LENGTH(column\_name) condtion;
6. SELECT column\_name FROM table\_name WHERE condition;
7. SELECT column\_name ORDER BY column\_name ASC/DES FETCH FIRST ROW ONLY;
8. SELECT column\_name FROM table\_name WHERE column\_name = TO\_DATE(‘date’) value;
9. SELECT \* FROM table\_name WHERE SUBSTR(column\_name,st\_range,end\_range) value;
10. SELECT \* FROM table\_name WHERE LOWER(column\_name) LIKE ‘%value’;
11. SELECT \* FROM table\_name WHERE LOWER(column\_name) LIKE ‘value%’;
12. SELECT \* FROM table\_name WHERE LOWER(column\_name) LIKE ‘\_value\_’;
13. SELECT \* FROM table\_name WHERE LOWER(column\_name) LIKE ‘\_value’;
14. SELECT \* FROM table\_name WHERE LOWER(column\_name) LIKE ‘value\_’;

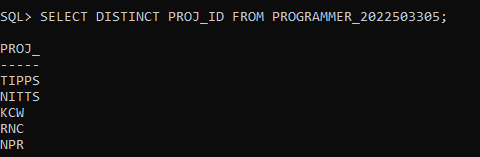
1.



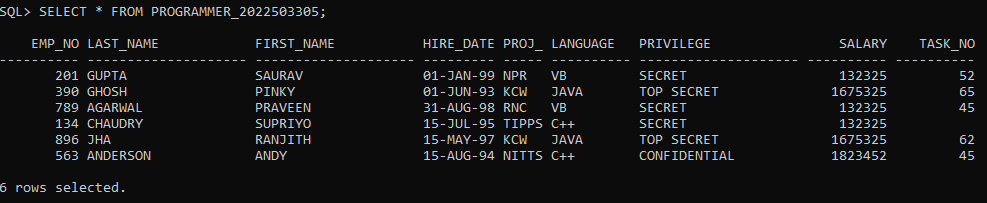
2.



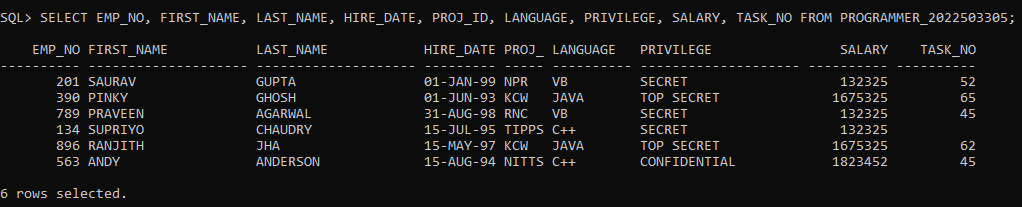
3.



4.



5.



**QUESTIONS**

1. Report the employee full name who has the Task no greater than 50.

2.Report the employee details who hadhired in the year 1999.

3.Report the employee details whose ProjId is TIPPS and Programming language is Java

4.Report on unique ProjIdvalues from the PROGRAMMER table.

5.Use different way to return all columns and rows of data from the PROGRAMMER table6.Report the employee details whose length of full name is greater than 10 letters.

7.Report the employee ProjId and Task no Whose Privilege is Secret but starts at position 5.

8.Report the employee who has hired earlier than all others

9.List the employee project details whose name is “JOHN PETER” and “GHOSH PINKY”.

10.Report the project details of an employee by reading their empno at run time.

11.Obtain the information about the employee who has recruited 30 days before 13/06/95.

12.Find the details of the employee whose first letter of full name and first letter of last name starts with same character.

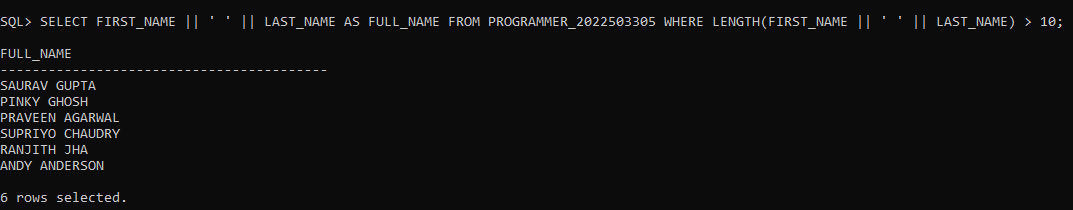
13.Find the details of the employee whose first letter of full name and first letter of last name not starts with same character.

14.Find the details of the employee with FIRST\_NAME values that do NOT begin with the letter “A” or those that do NOT comply with a COMMISSION greater than 2% percent.

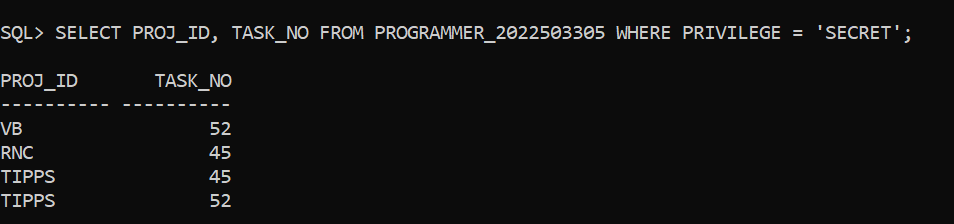
15.Find the details of the employee whose lastname contains “u” letter and had thesalary greater than 5000 bearing taskno 89 or 52 or 10 whose commission calculated should not be null.

16.Report the details of the employee whose hire date is in the month that starts with ‘J’(i.e January or june)

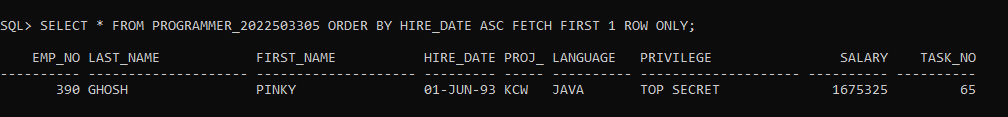
6.



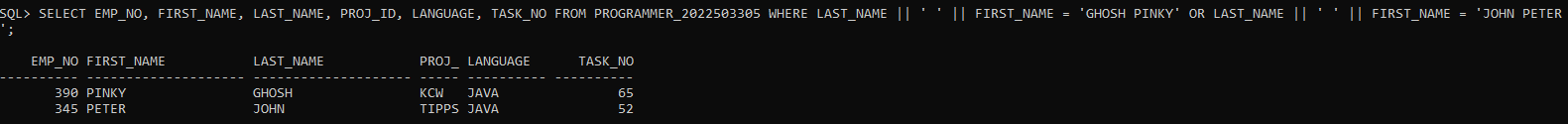
7.



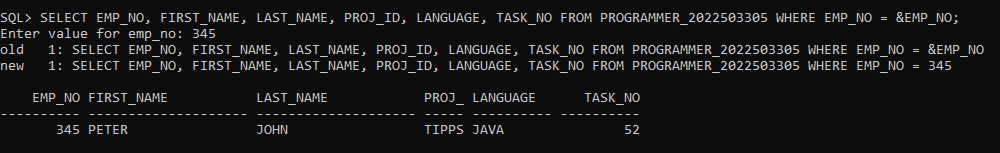
8.



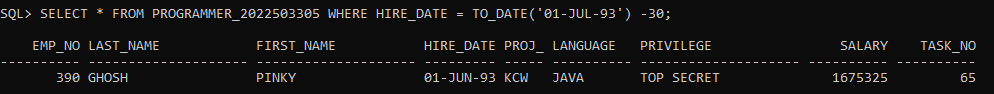
9.



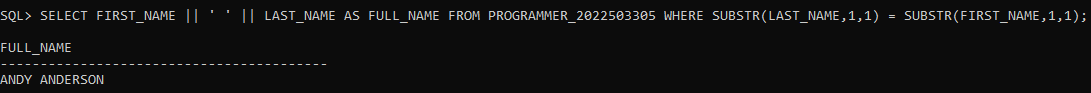
10.



11.



12.



17.Query the ProjId from Programmer table.i.e ., The Project Idfor the <FirstName> <LastName> ‘s projectis: <ProjId >, where the ProjId being concatenated to the statement so as to create the single expression aliased as ProjectDescription. The literal "'s projectis:" is then concatenated with this text, and the ProjIdfield is concatenated after that.

Example:Project Description“The projectId for the Saurav Gupta’s Projectis: NPR.”

18.Find the the number of days for which staff were employed in a job.Create an alias for the expression column in your query using Days Employed.

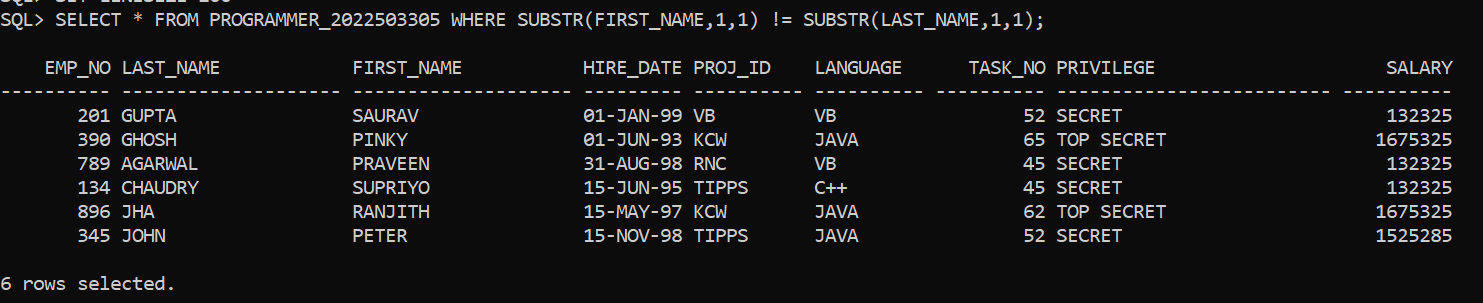
19.Write an SQL query to determine the number of years each staff member has been employed in a job. Display the employee number (EmpNo), project ID (ProjId), hire date (HireDate), and create an alias for the expression column representing the number of years employed using the name 'Years Employed'. Consider that there are 365 days in a year.

20.Display the salary of each employee in the following format: '$9.999.000'.

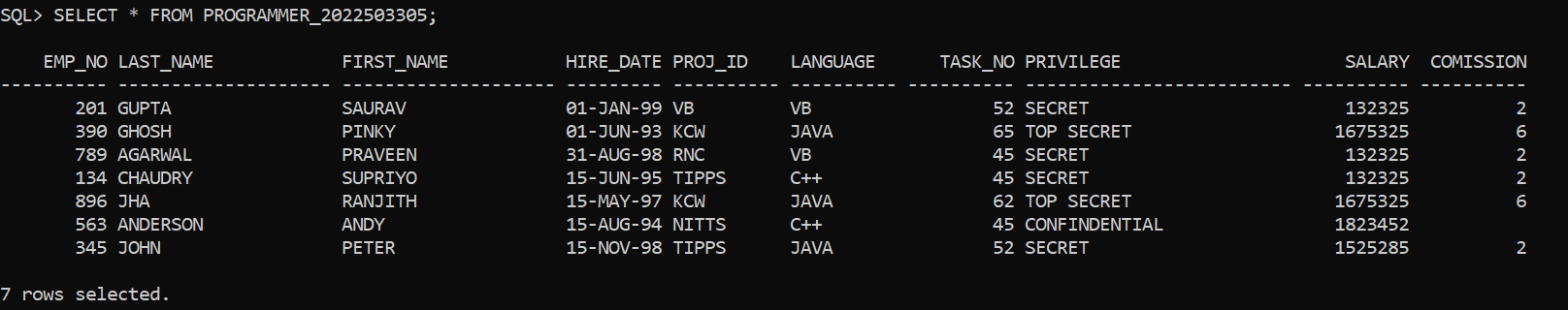
21.Create an SQL query to show the hire date of employees, with the month expressed in full name format.

22.Write an SQL query to display employee details, showing the hire date in the 'yyyy' format.

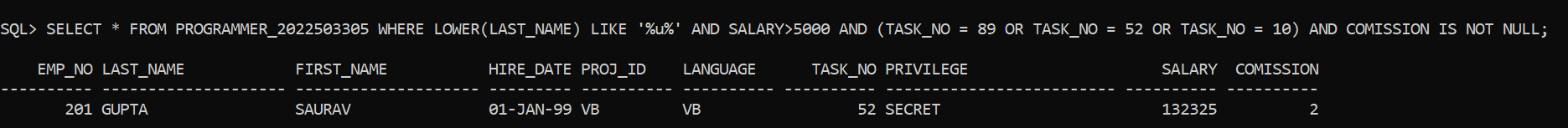
13.



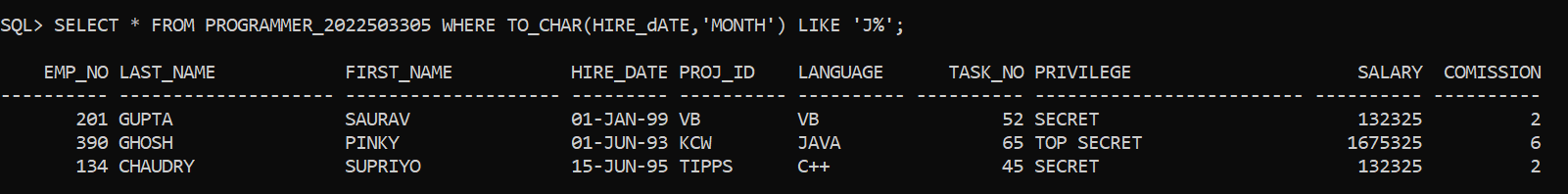
14.



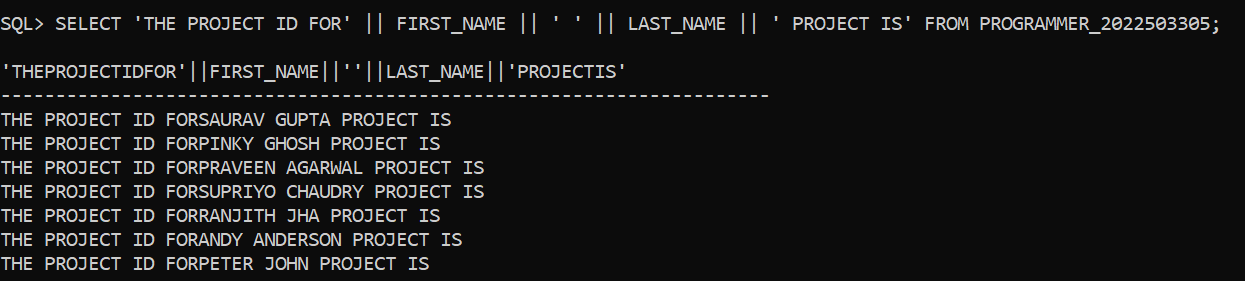
15.



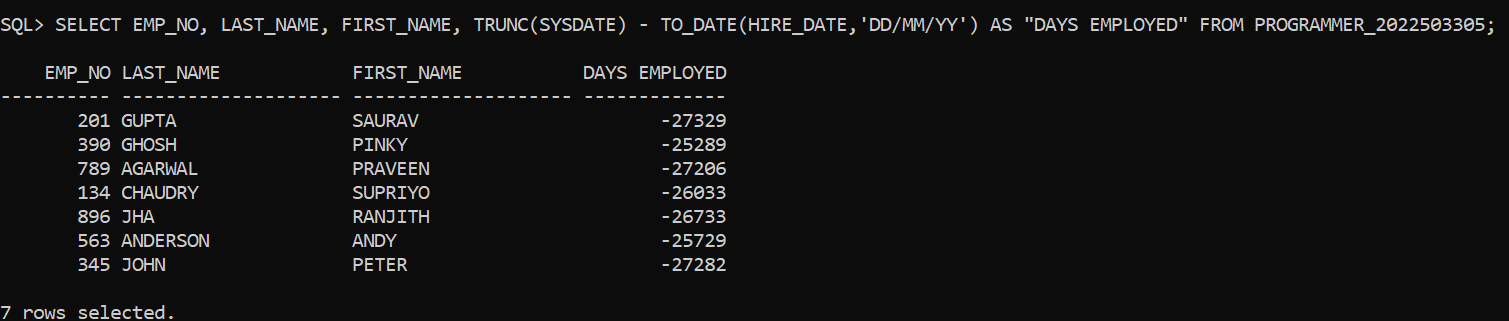
16.



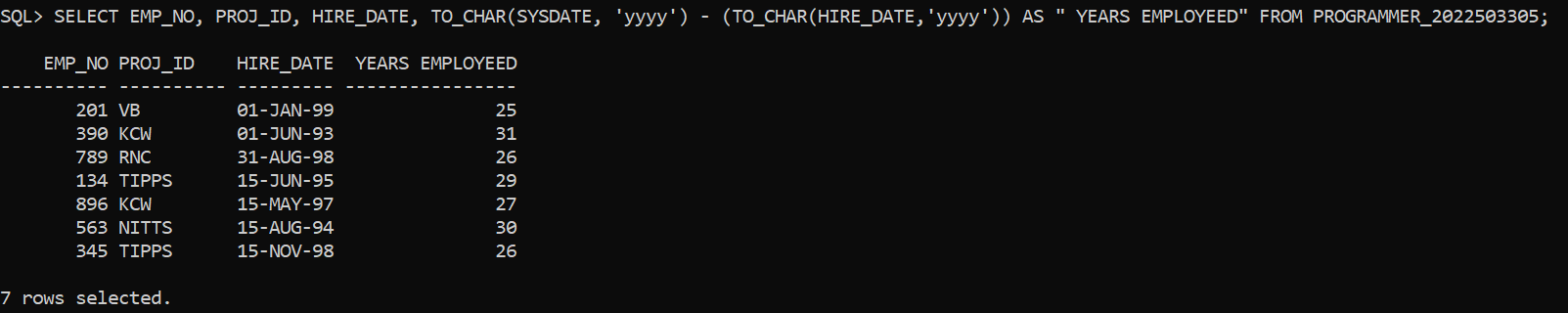
17.



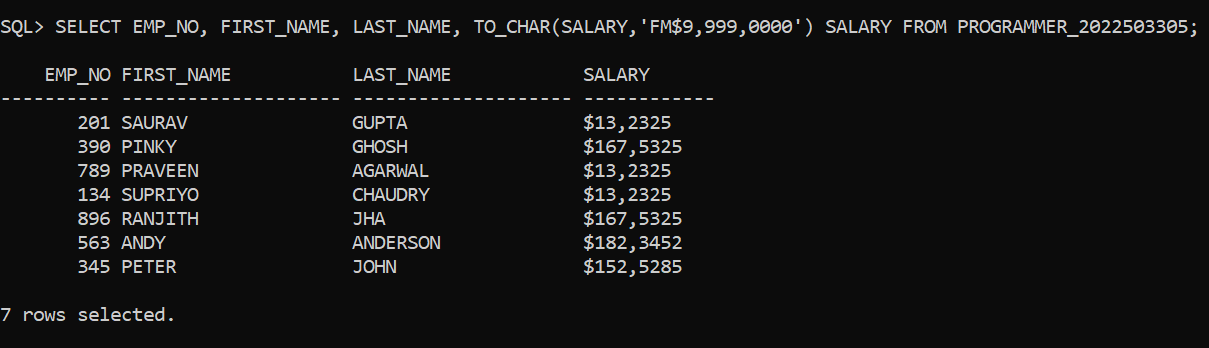
18.



19.



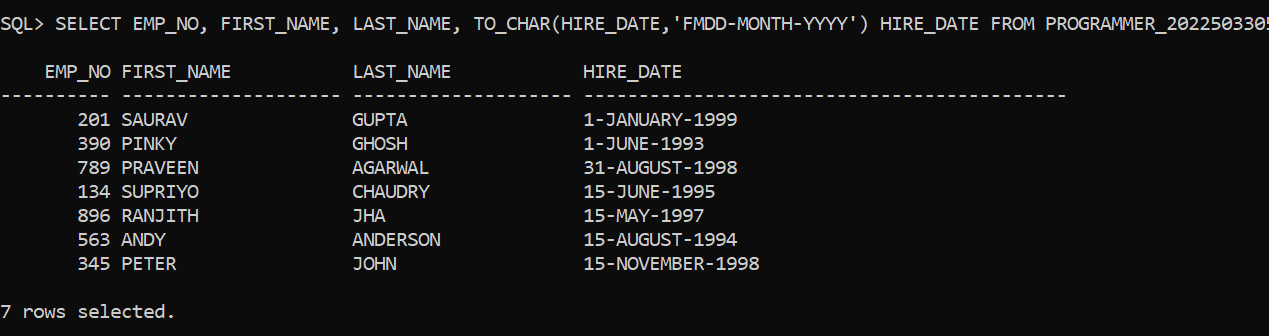
20.



21.



22.



**RESULT**

Thus, retrieving the records in the tables from the database using Data Query Functions has been done successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 06 | **VIEWS** |
| **DATE** | **:** 13.03.2024 |

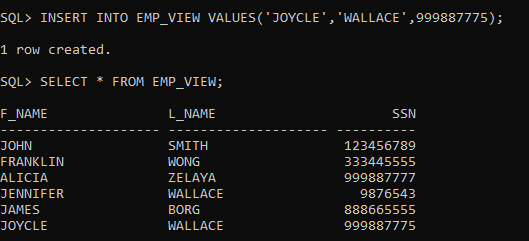
**AIM**

To simplify the complex queries, enhance the data and provide data abstraction using the logical representation by Views.

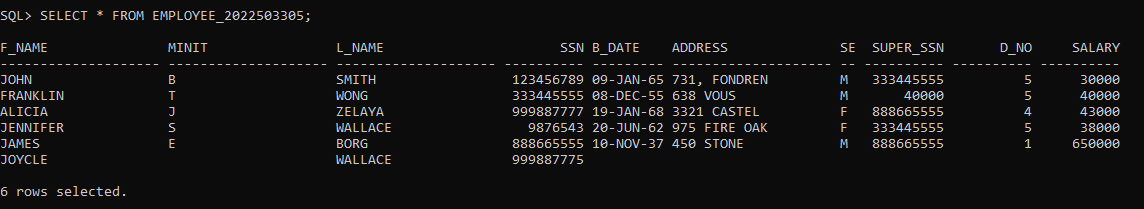
**SYNTAX**

1. CREATE VIEW view\_name AS SELECT column\_name FROM table\_name;
2. INSERT INTO view\_name VALUES(value1, value2, value3… valuen);
3. UPDATE view\_name SET column\_name = value WHERE column\_name condition;
4. SELECT \* FROM view\_name;
5. DROP VIEW view\_name;
6. DESC view\_name;

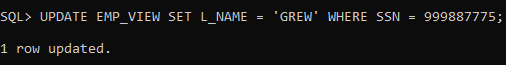
1.



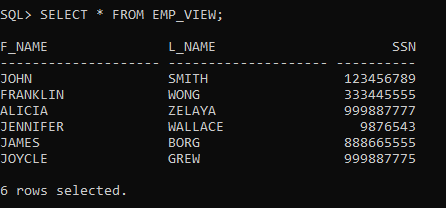
2.



3.

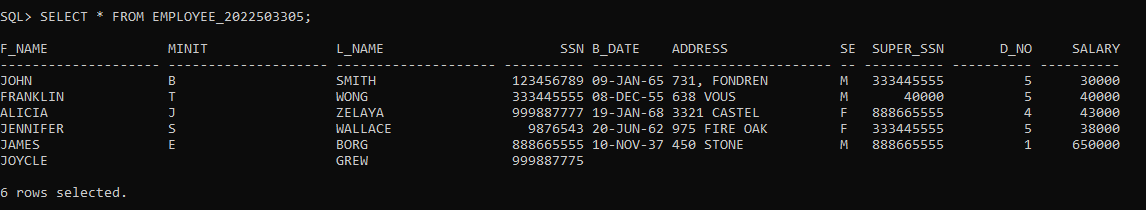


1. CHANGES IN VIEW TABLE

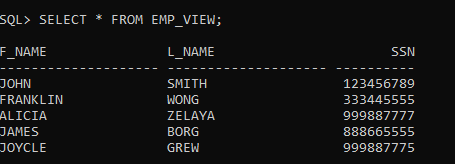


**QUESTION**

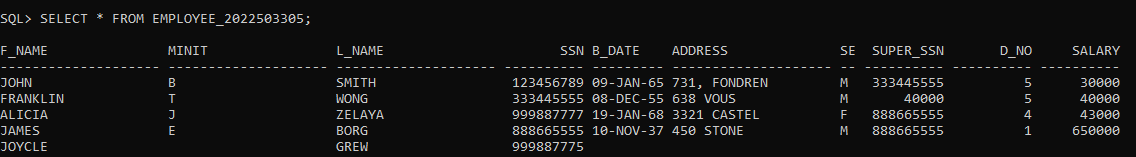
1. 1.Insert , update, delete on simple views that reflects views and base table.
2. Insert, delete, update on simple views that violate the constraint such as primary key. Note: primary key of a table is a foreign key on other table.
3. Insert, update, delete on complex views that reflects views and base table
4. Drop view and perform query on base table and vice versa.
5. Drop column in base table after creation of view with the respective column and perform select query on view.
6. Rename column in base table after creation of view with the respective column and perform select query on view.
7. Rename table in base table after creation of view and perform select query on view.
8. Modify data type of a column in base table after creation of view with the respective column and perform select query on view.
9. Create a view from existing view.
10. CHANGES IN BASE TABLE.



1. CHANGES IN VIEW TABLE

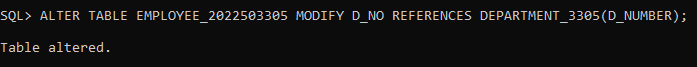


1. CHANGES IN BASE TABLE



2.

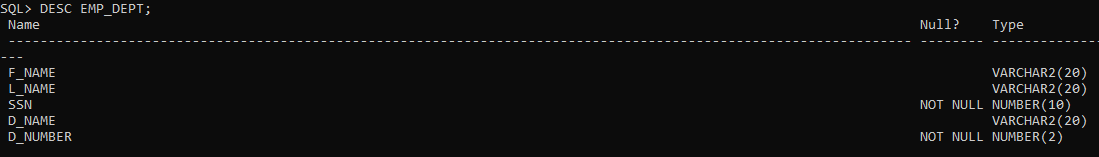
SETTING FOREIGN KEY



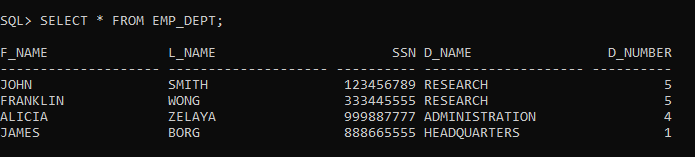
CREATING VIEW FOR BOTH THE TABLES AS EMP\_VIEW

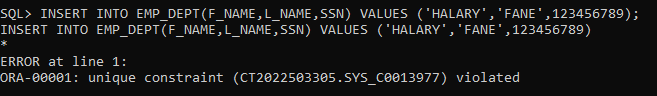


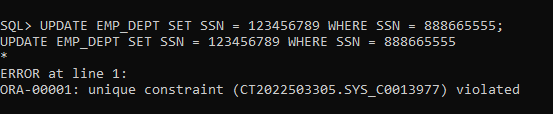
DESCRIBING THE VIEW



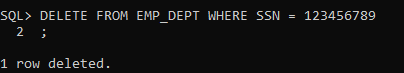
RETRIVING DATA FROM THE VIEW



 INSERTING IN EMP\_DEPT TABLE

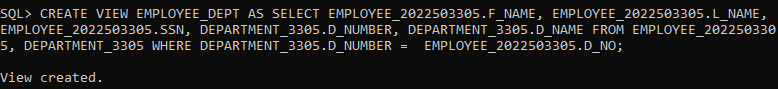
 UPDATING IN EMP\_DEPT TABLE

DELETING IN EMP\_DEPT TABLE

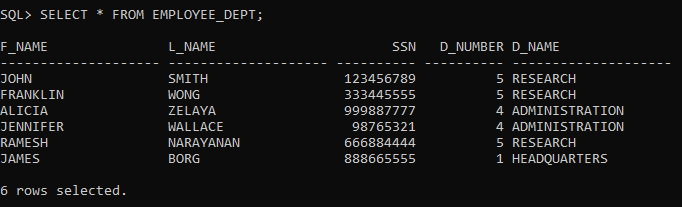


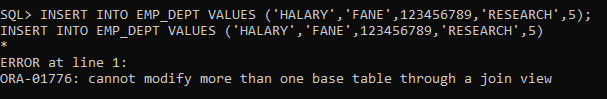
3.

CREATING A COMPLEX VIEW

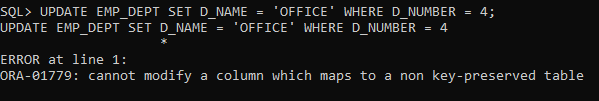


RETRIVING DATA FROM THE VIEW

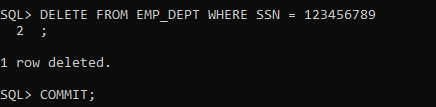


 INSERTING IN COMPLEX VIEW TABLE

UPDATING IN COMPLEX VIEW TABLE



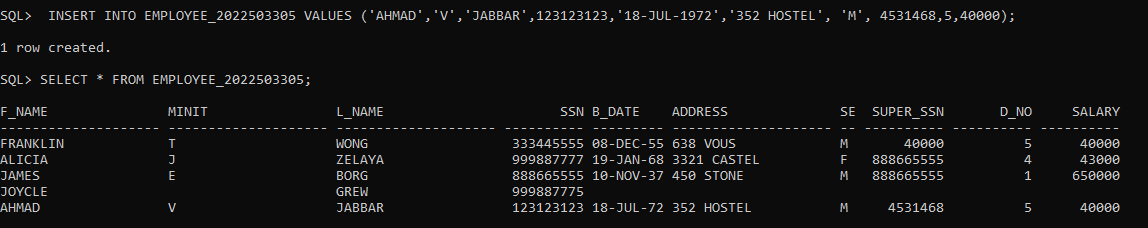
DELETING IN COMPLEX VIEW TABLE



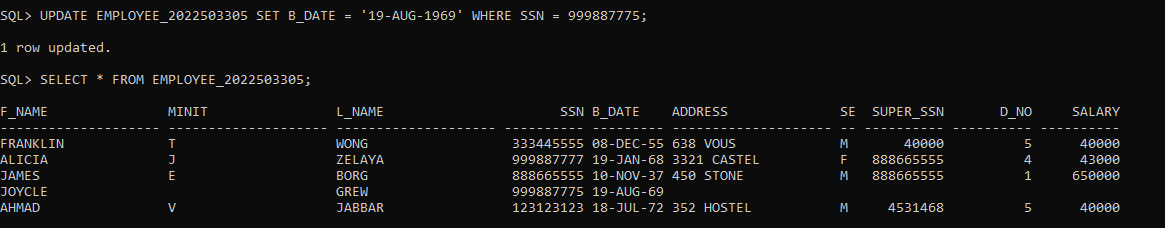
4.

DROPING THE VIEW

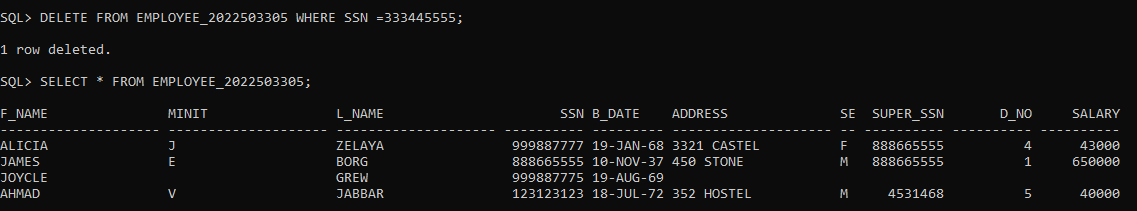


 INSERTING INTO THE BASE TABLE

UPDATING THE BASE TABLE



DELETING A ROW IN THE BASE TABLE

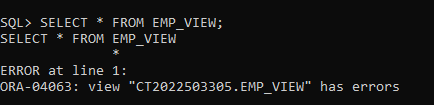


5.

DROPING THE RESPECTIVE COLUMN WHICH IS AVAILABLE IN BOTH TABLES

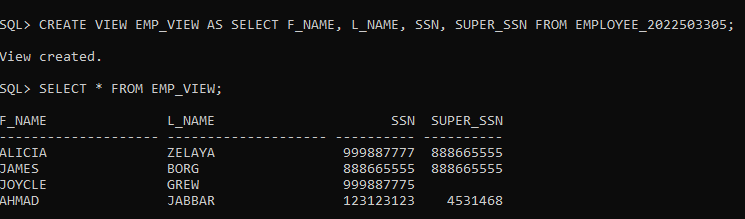


PERFORMING SELECT QUERY AFTER DROPING THE RESPECTIVE COLUMN

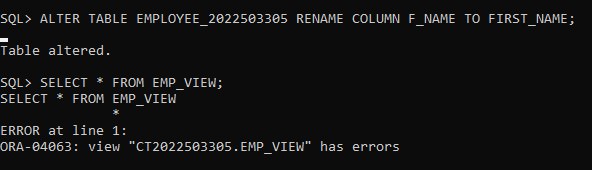


6.

CREATING THE VIEW

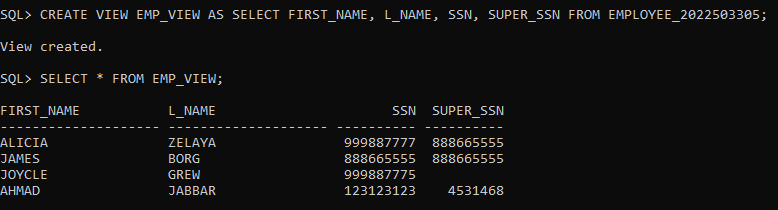


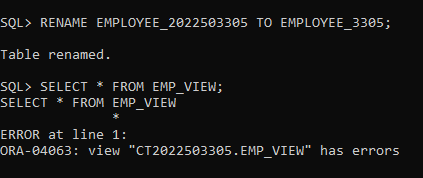
CHANGING THE COLUMN NAME OF THE BASE TABLE



7.

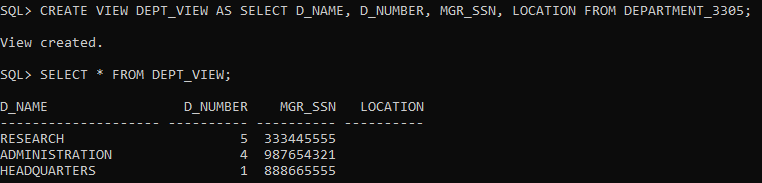
CREATING THE VIEW



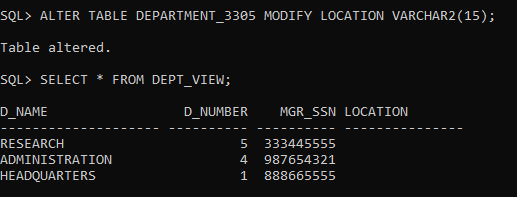
 RENAMING THE BASE TABLE AND PERFORMING THE SELECT QUERY

8.

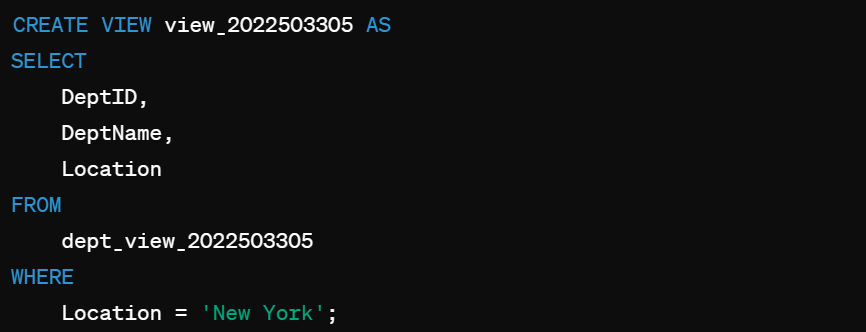
CREATING THE VIEW



MODIFYING THE DATATYPE OF THE BASE TABLE AND PERFORMING THE SELECT QUERY



9.



**RESULT**

Thus, simplification the complex queries, enhancing the data and providing data abstraction using the logical representation by Views has been done successfully.

|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 07 | **JOINS** |
| **DATE** | **:** 02.04.2024 |

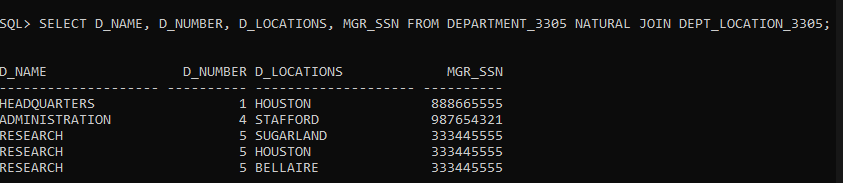
**AIM**

To practice the queries on joins such as natural join, cross join, non equii join, self join.

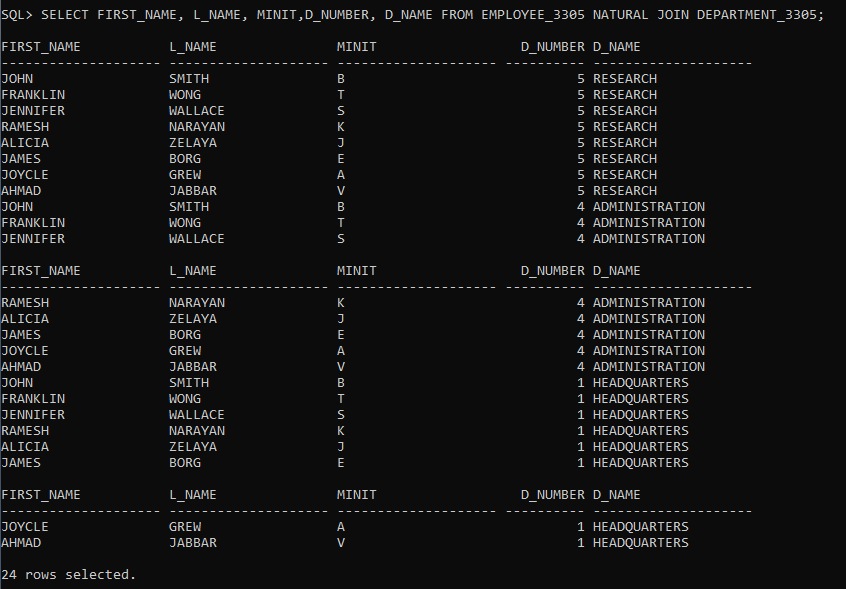
**SYNTAX**

1. SELECT \* FROM table\_name1 NATURAL JOIN table\_name2;
2. SELECT \* FROM table\_name1 JOIN table\_name2 ON condition;
3. SELECT \* FROM table\_name1 JOIN table\_name2 ON condtion WHERE condition;
4. SELECT \* FROM table\_name1 INNER JOIN table\_name2 ON condition;
5. SELECT \* FROM table\_name1 LEFT JOIN table\_name2 ON condition;
6. SELECT \* FROM table\_name1 RIGHT JOIN table\_name2 ON condtion;

1.



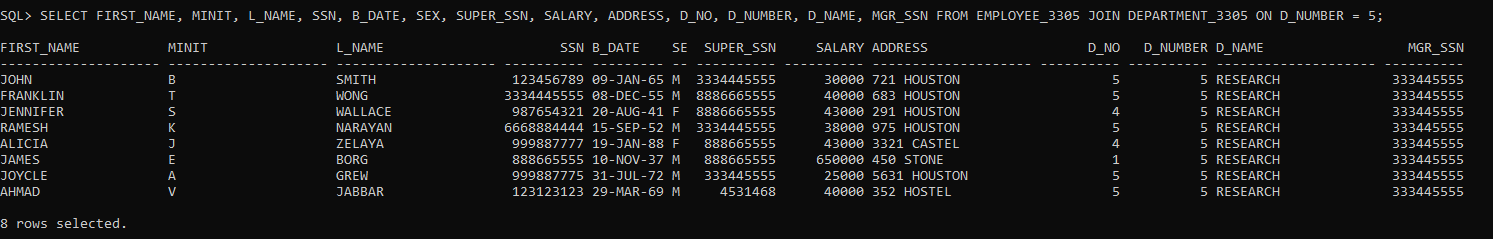
2.



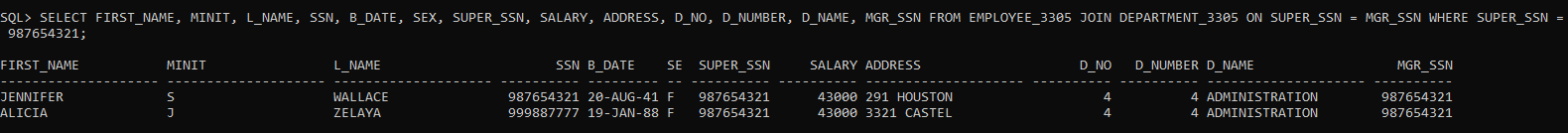
**QUESTIONS**

1. Find the department detailssuch as department name, department number, location and manager
2. Find the employeeand his/herdepartment details
3. Find the employee and department details for the department “Research”
4. Find the employee and department details for the manager id 98754321.
5. Find the employee and department details for the department location ‘Houston’
6. Add a new employee without assigning his dept details.
7. Add a new department dnumber=3and dlocation=’NEW YORK’
8. Find the employee and dept details who wereassigned to any of the department.
9. Find the department details that has employee assigned.
10. Find the employee and manager nameof each departmentOuter join(LEFT /RIGHT/FULL , IS NULL , IS NOT NULL)
11. Find the employee who has not assigned to any department.
12. .Find the DEPARTMENT that has no employee.
13. Find the employee who has not assigned to any department.
14. Find all employeeswho areeither assigned or not assigned to any department.
15. Find departmentdetails that has either employees or no employees

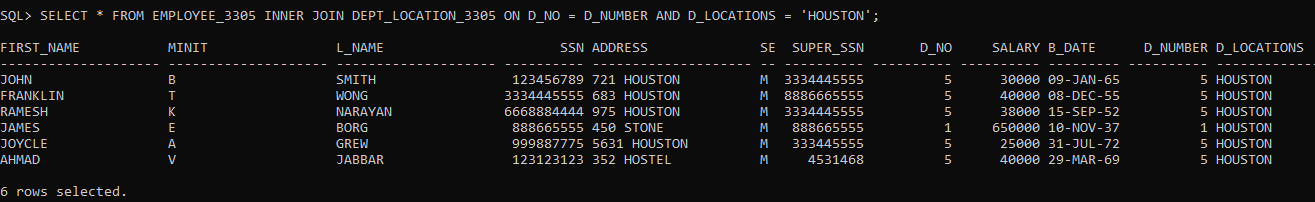
3.



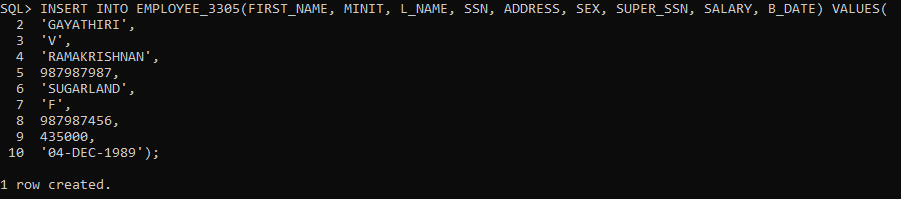
4.

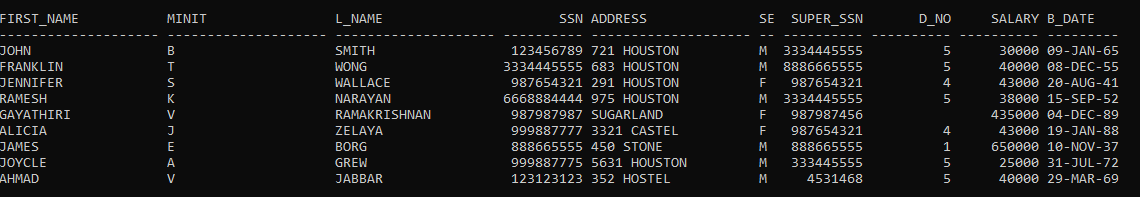


5.

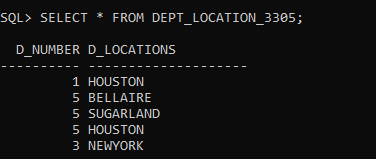


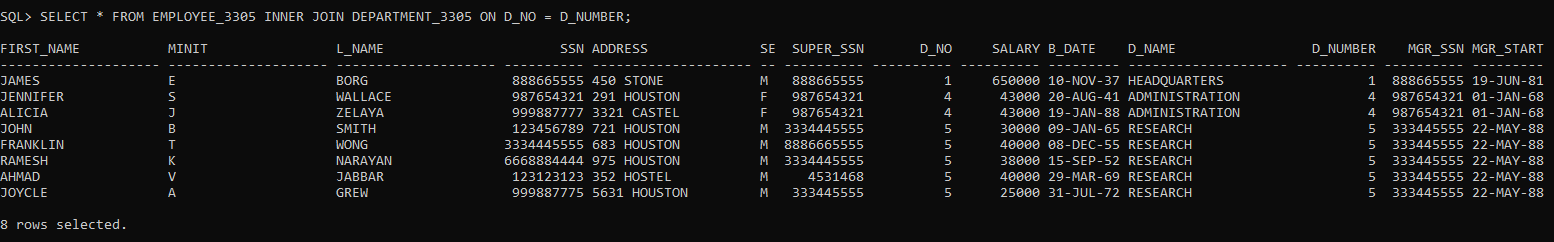
1. Add a new employee without assigning his dept details.



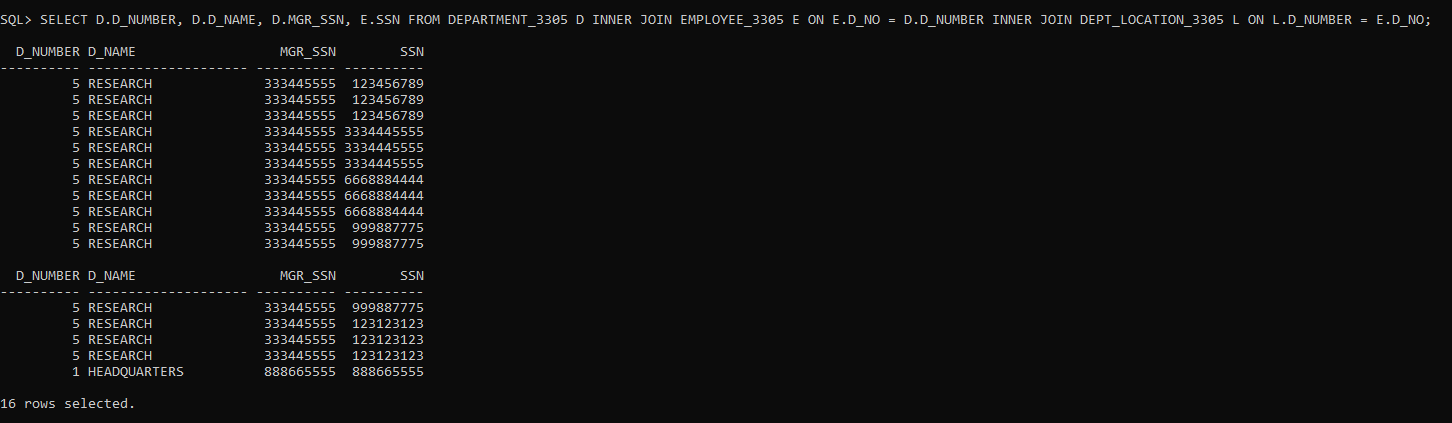


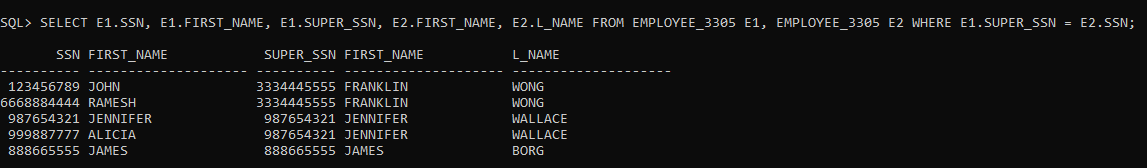
1. Add a new department dnumber = 3and dlocation=’NEW YORK’



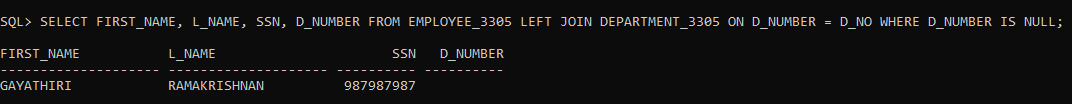
 6.

7.



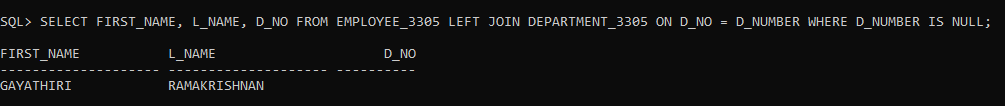
 8.

9.

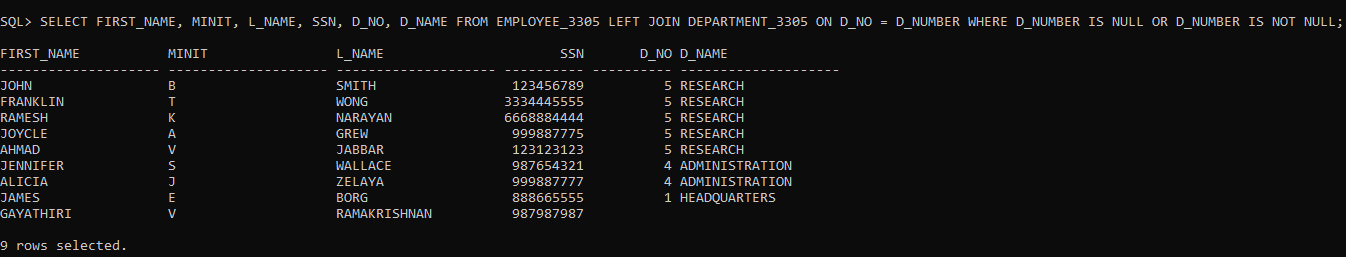


 10.

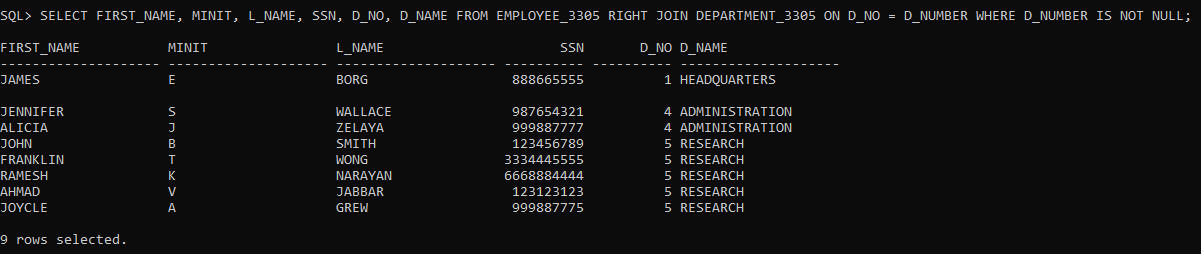
11.



12.



13.



|  |  |  |
| --- | --- | --- |
| **EX NO** | **:** 08 | **AGGREGATE FUNCTIONS – GROUP BY, ORDER BY** |
| **DATE** | **:** 02.04.2024 |