**NAME :- ANIKET SANJAYKUMAR BIYANI**

**DATA ENGINEERING BATCH -1**

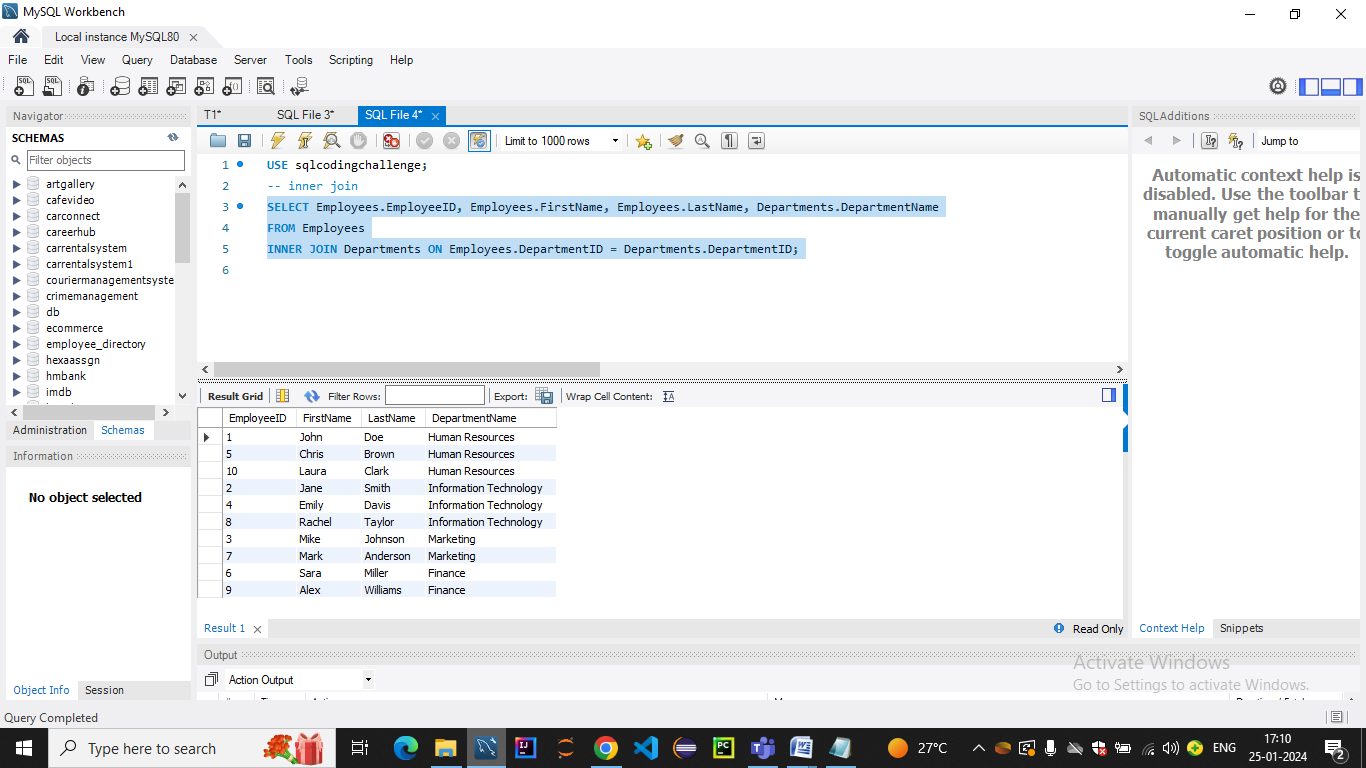
**SQL CODING CHALLENGE**

**Q2** **Execute all the join with examples.**

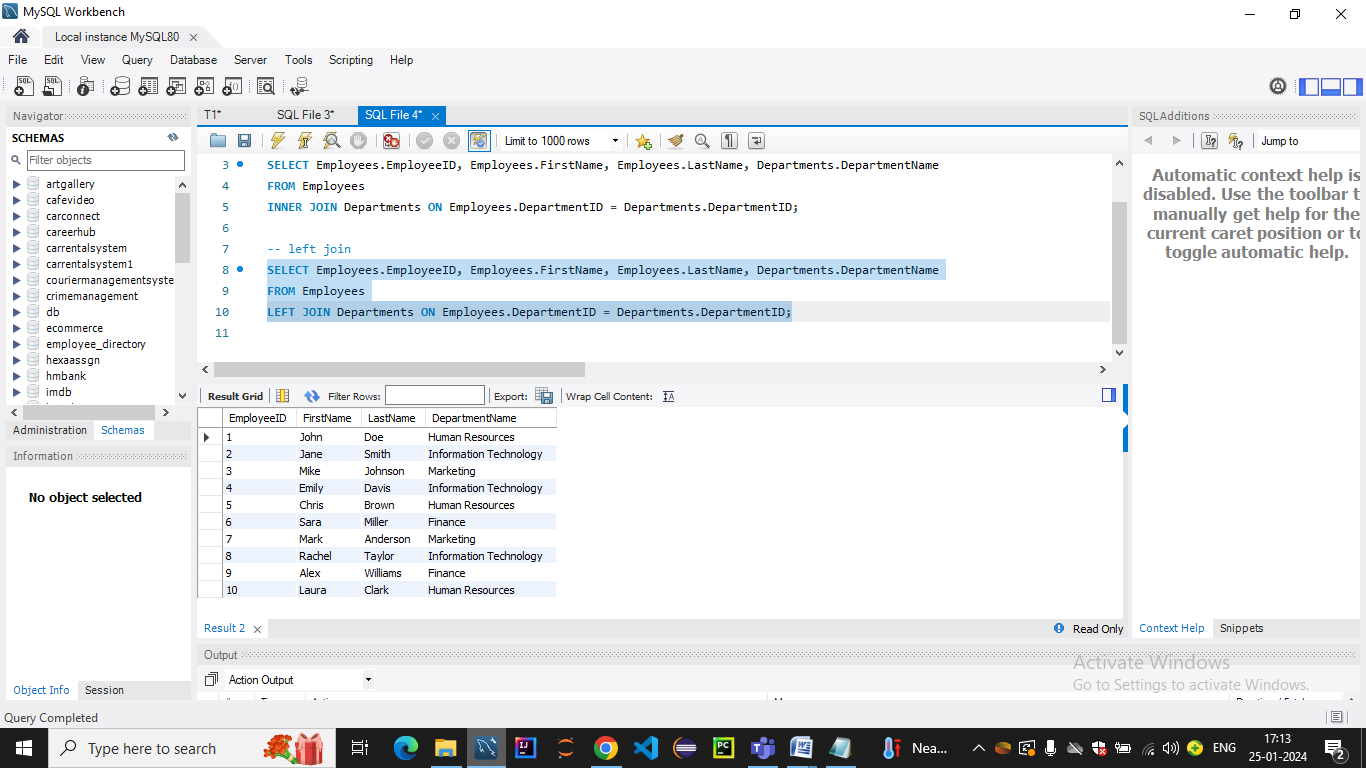
* ***THEORY :-***
* In relational databases, a join is a SQL operation that combines rows from two or more tables based on a related column between them.
* The purpose of a join is to retrieve data from multiple tables in a single query, allowing you to create a result set that includes columns from both tables.
* By using joins, you can create a single result set that incorporates data from different tables.
* Joins are a key aspect of SQL queries when it comes to retrieving, analyzing, and reporting data from multiple tables.

***TYPES OF JOINS :-***

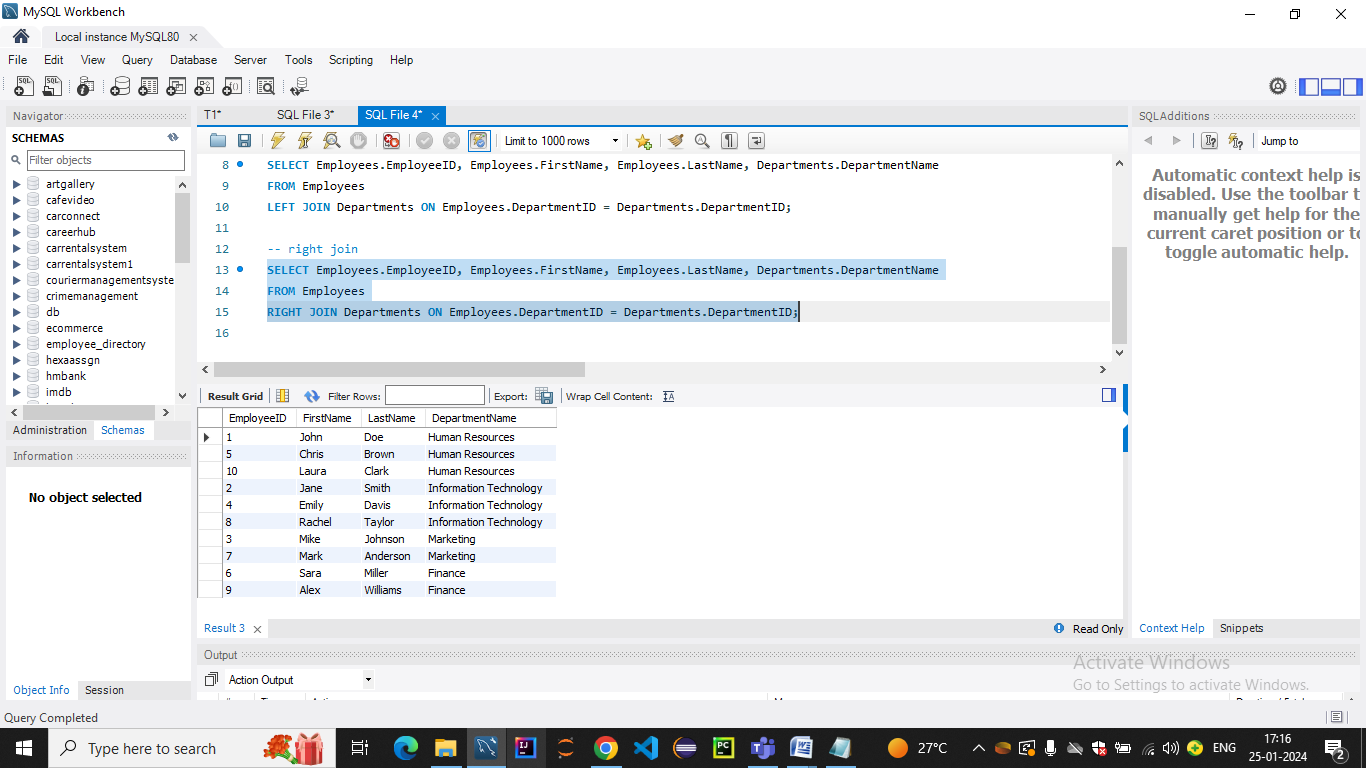
* INNER JOIN:
* An INNER JOIN returns only the rows where there is a match in both tables based on the specified join condition



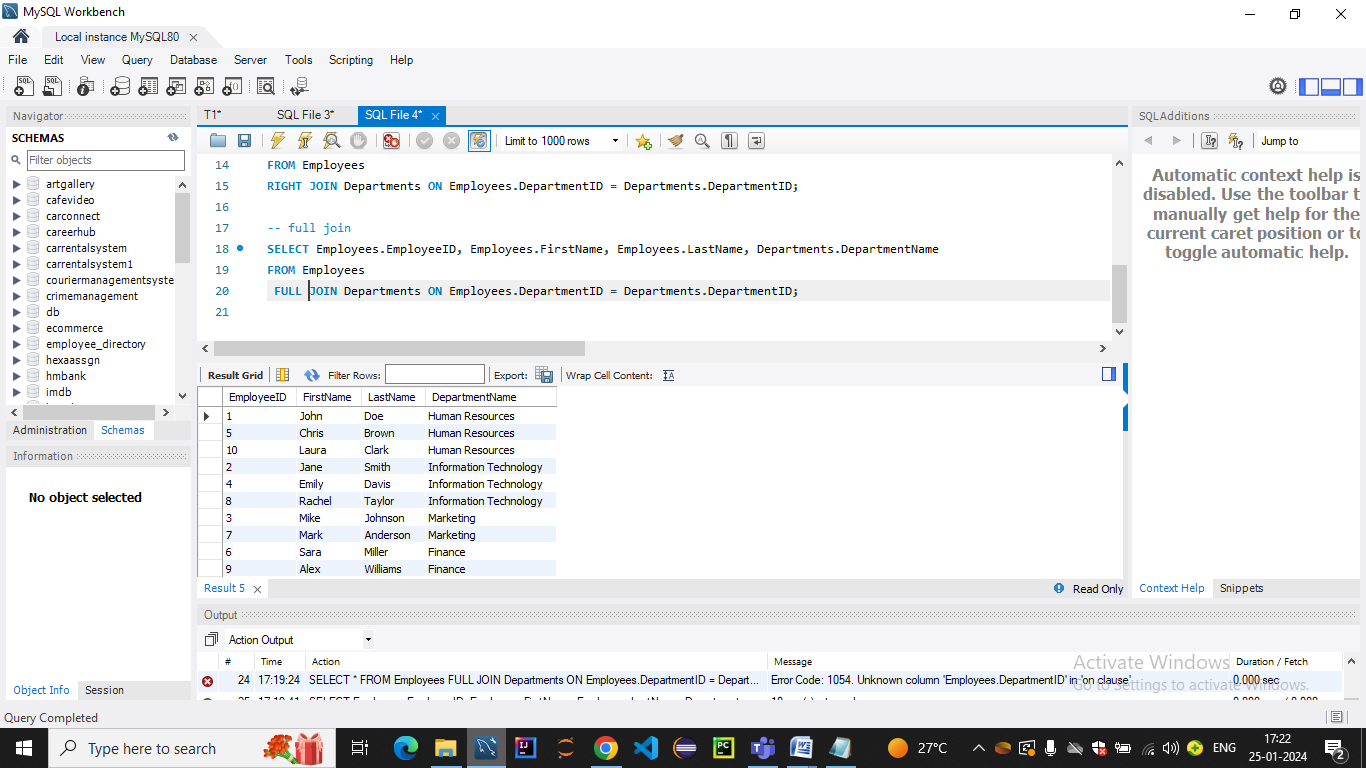
* *EXPLANATION :-*
* This query selects the EmployeeID, FirstName, LastName, and DepartmentName for employees who belong to a department.
* The INNER JOIN is performed on the DepartmentID column, ensuring that only matching records from both tables are included in the result.
* LEFT JOIN (or LEFT OUTER JOIN):
* A LEFT JOIN returns all rows from the left table and the matched rows from the right table .
* If there is no match, NULL values are returned for columns from the right table.



* *Explanation:*
* This query selects the EmployeeID, FirstName, LastName, and DepartmentName for all employees, even those without a matching department.
* The LEFT JOIN ensures that all rows from the left table (Employees) are included, and if there's no match in the right table (Departments), the columns from the right table will have NULL values.
* RIGHT JOIN (orRIGHT OUTER JOIN):
* A RIGHT JOIN returns all rows from the right and the matched rows from the left table (
* If there is no match, NULL values are returned for columns from the left table.

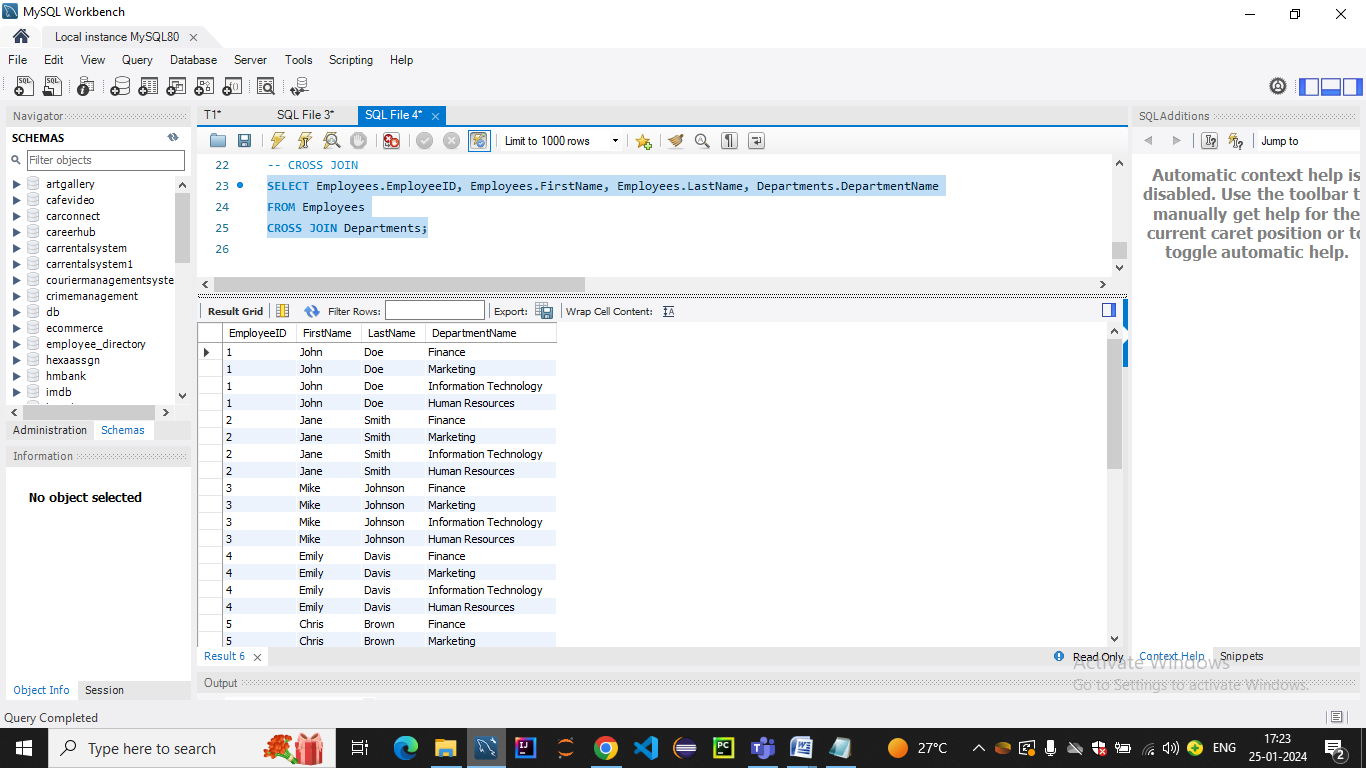


* *Explanation:*
* This query selects the EmployeeID, FirstName, LastName, and DepartmentName for all departments, even those without any employees.
* The RIGHT JOIN ensures that all rows from the right table (Departments) are included, and if there's no match in the left table (Employees), the columns from the left table will have NULL values.
* FULL JOIN (or FULL OUTER JOIN):
* A FULL JOIN returns all rows when there is a match in either the left or right table. If there is no match, NULL values are returned for columns from the table without a match

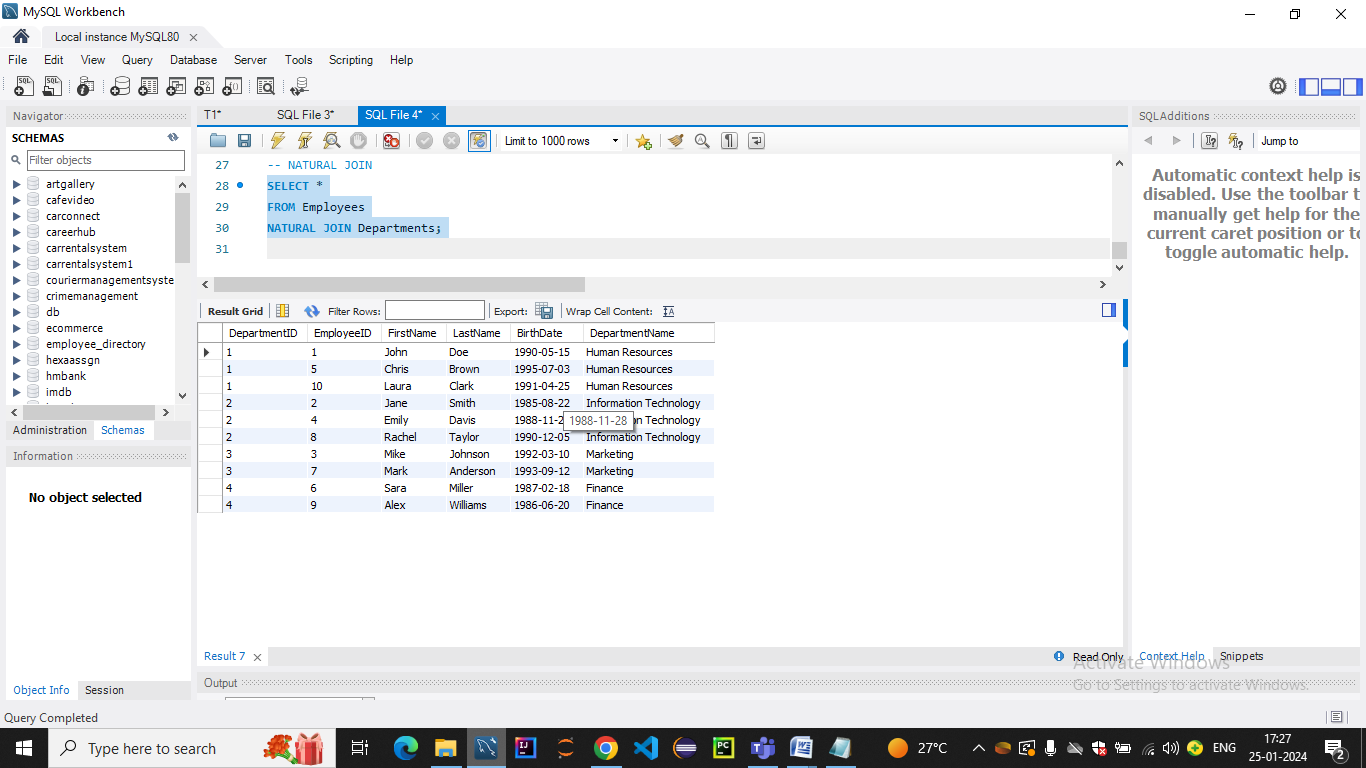


* *Explanation:*
* This query selects the EmployeeID, FirstName, LastName, and DepartmentName for all employees and departments.
* The FULL JOIN ensures that all rows from both tables are included, and if there's no match in either the left or right table, the columns from the table without a match will have NULL values.
* CROSS JOIN :

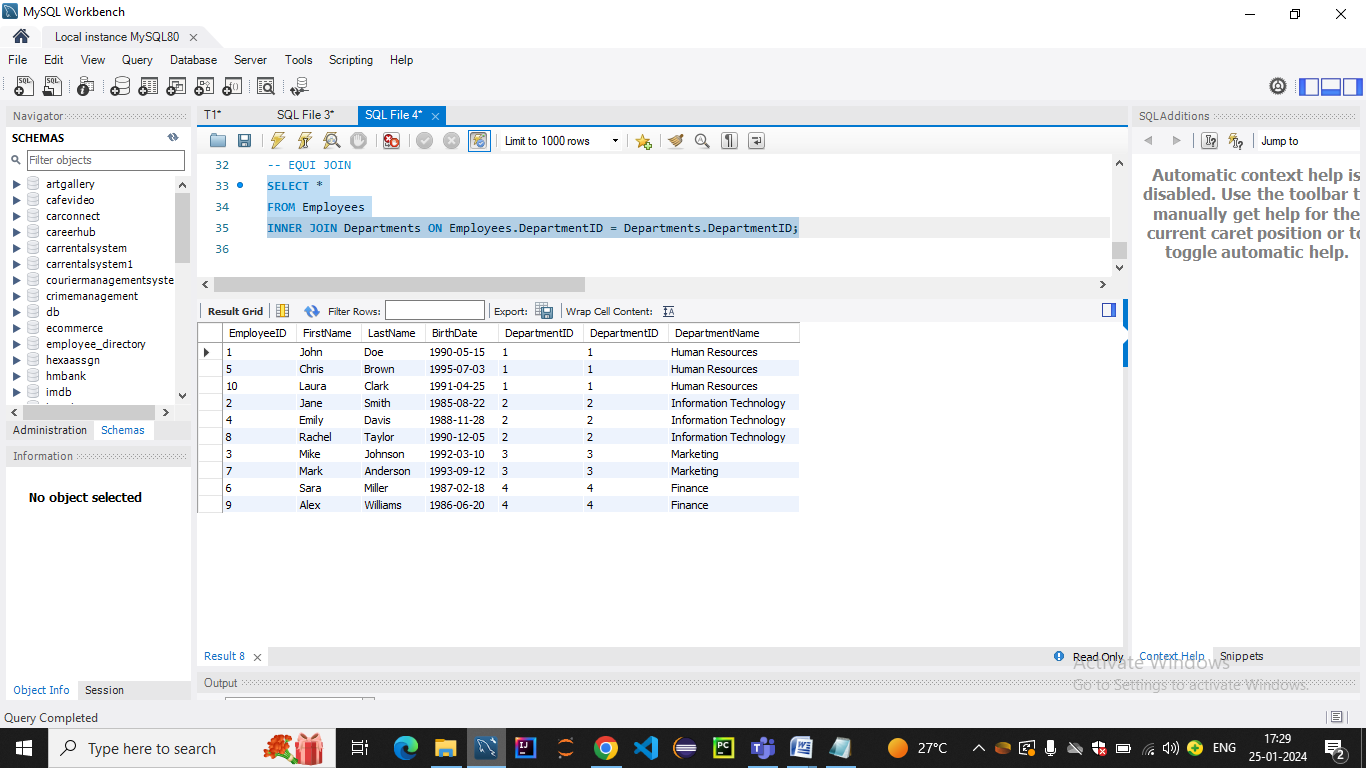
A CROSS JOIN returns the Cartesian product of the two tables, meaning it produces all possible combinations of rows from both tables.



* *Explanation:*
* query selects the EmployeeID, FirstName, LastName, and DepartmentName for all possible combinations of employees and departments.
* The CROSS JOIN does not require a join condition, and it results in a combination of each row from the left table with each row from the right table.
* NATURAL JOIN :
* A NATURAL JOIN is a type of join that automatically matches columns with the same name in both tables. It eliminates the need to explicitly specify the columns to join on.
* *Explanation:*
* This query performs a NATURAL JOIN between the Employees and Departments tables.
* It implicitly matches the columns with the same name (in this case, DepartmentID) in both tables.



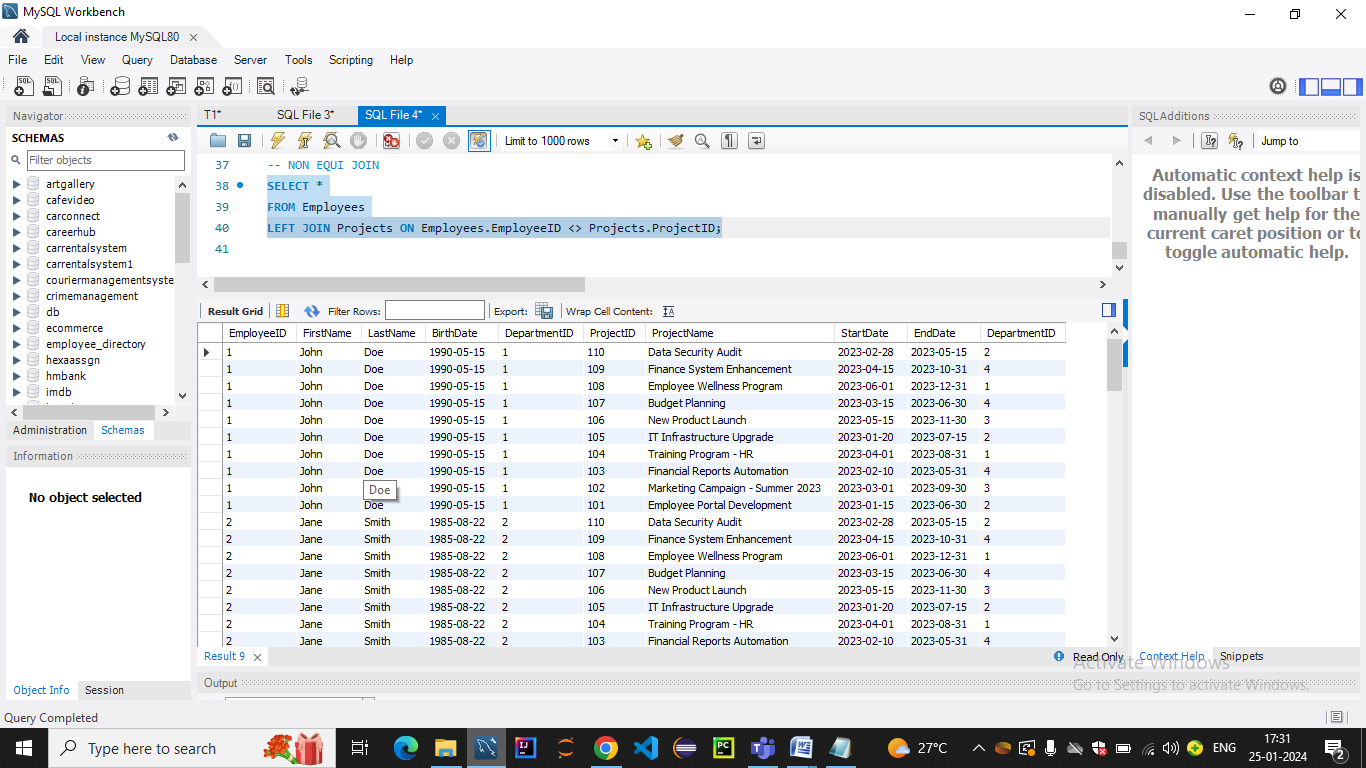
* EQUI JOIN:
* An EQUI JOIN is a general term for any join that uses equality conditions (e.g., =) to match rows between tables.
* This includes INNER JOIN, LEFT JOIN, and RIGHT JOIN when they are based on equality conditions.



*Explanation:*

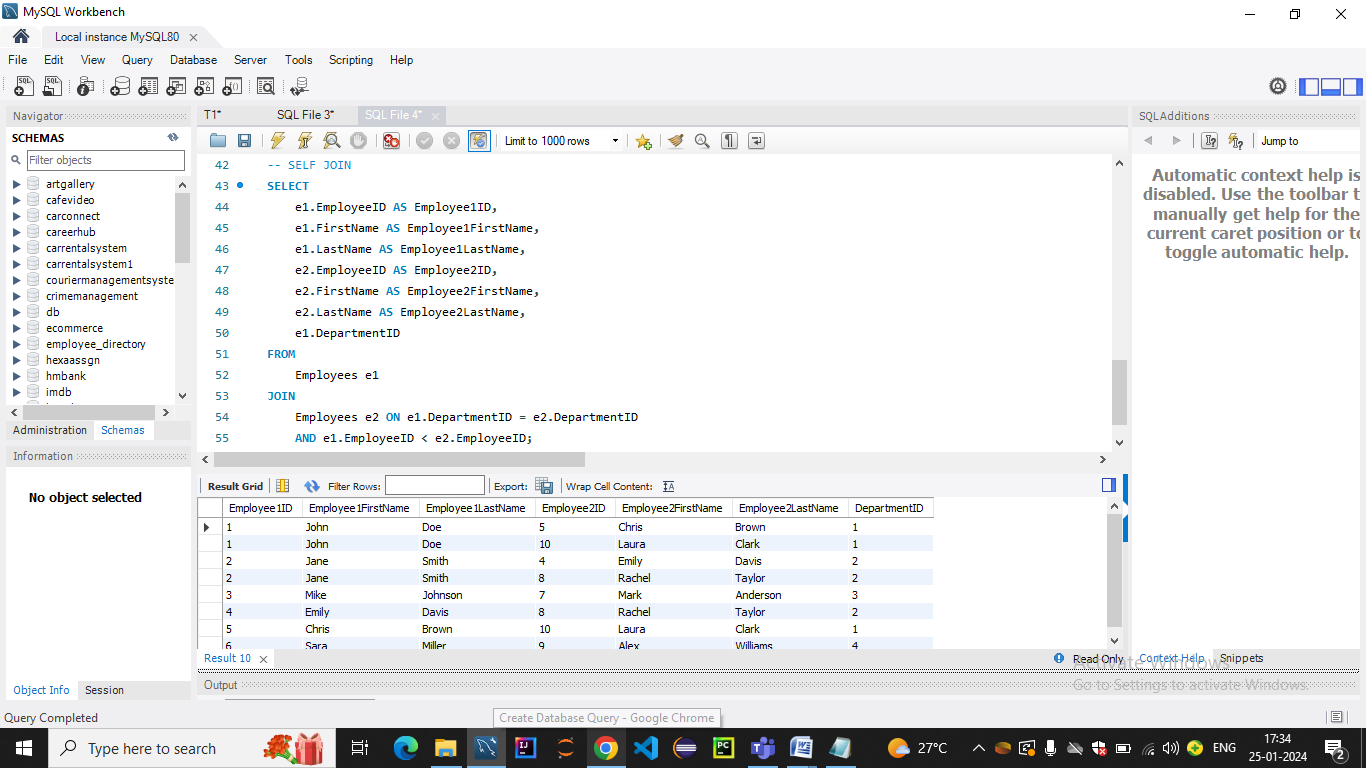
This query uses an EQUI JOIN (INNER JOIN) with the condition Employees.DepartmentID = Departments.DepartmentID to match rows between the Employees and Departments tables.

* NON-EQUI JOIN:
* A NON-EQUI JOIN involves matching rows between tables based on conditions other than equality (e.g., using operators like <, >, <=, >=, or non-equality operators like <> or !=).



*Explanation:*

This query uses a NON-EQUI JOIN (LEFT JOIN) with the condition Employees.EmployeeID <> Projects.ProjectID to match rows where the EmployeeID is not equal to ProjectID.

* SELF JOIN:
* A SELF JOIN is a join where a table is joined with itself. This is useful when you have hierarchical data or when you want to compare rows within the same table.
* 
* *Explanation:*
* This query will return pairs of employees who share the same department, avoiding duplicates and self-matches.