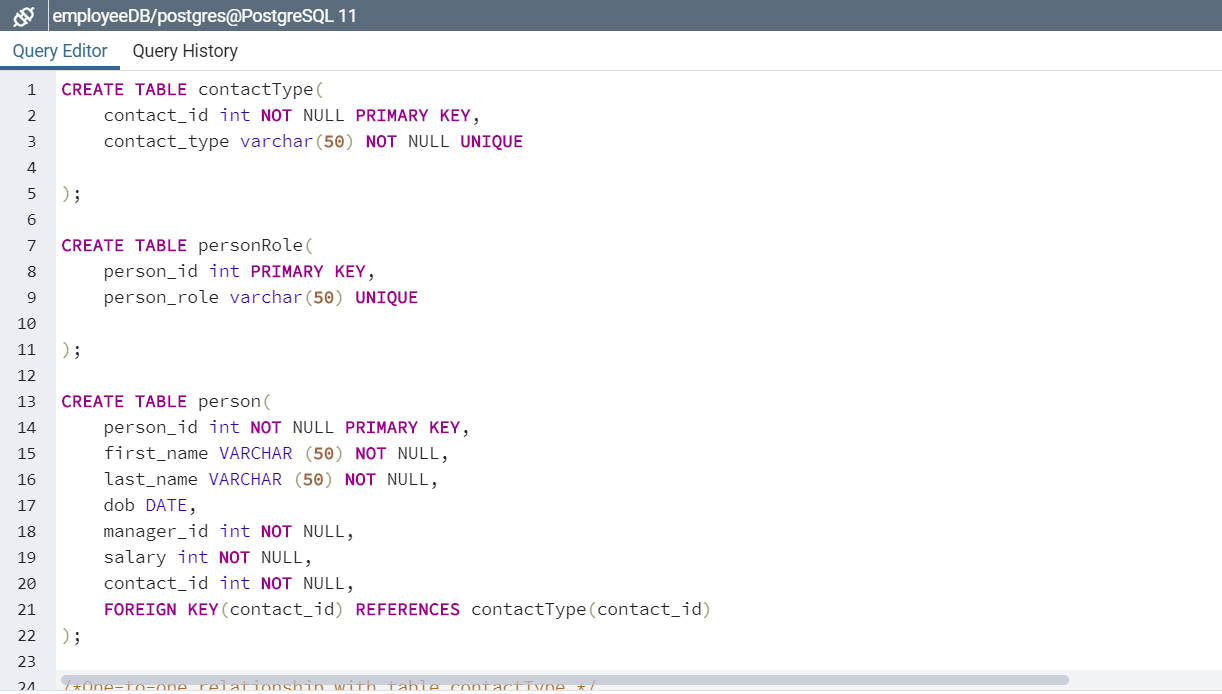
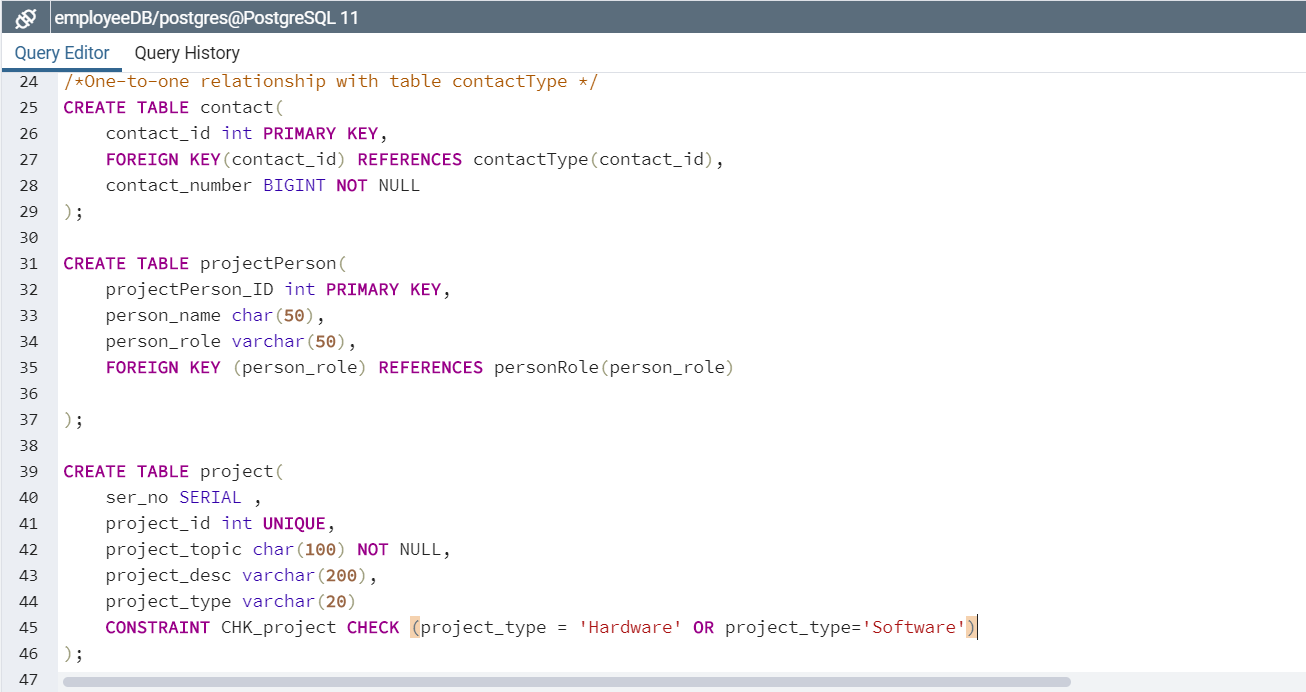
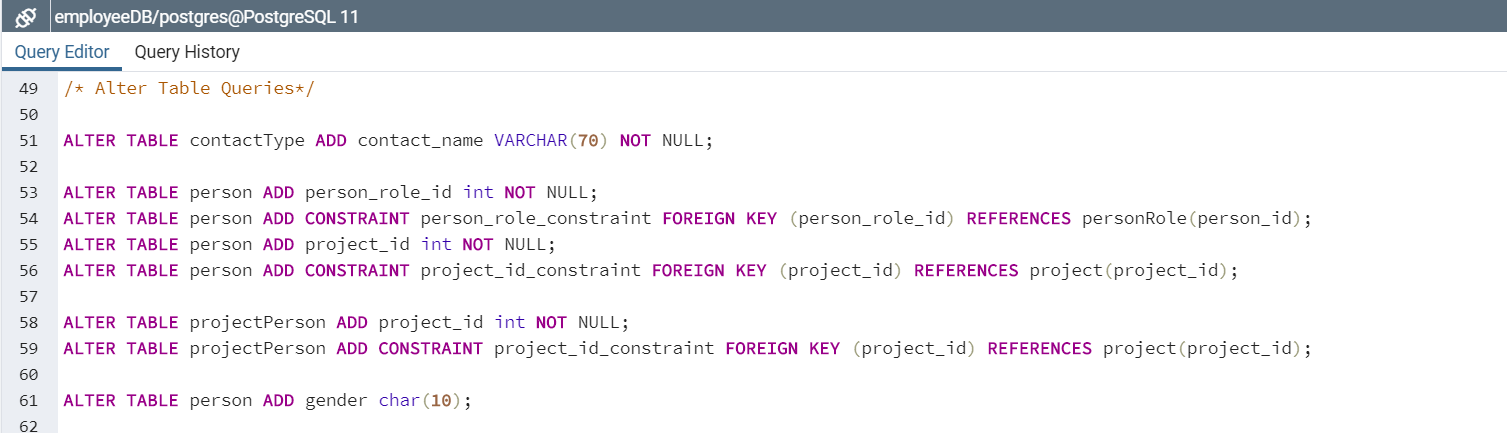
**Whole Practice Assignment SQL**

1. Create employeeDB.
2. Create Tables and link them using Foreign keys.

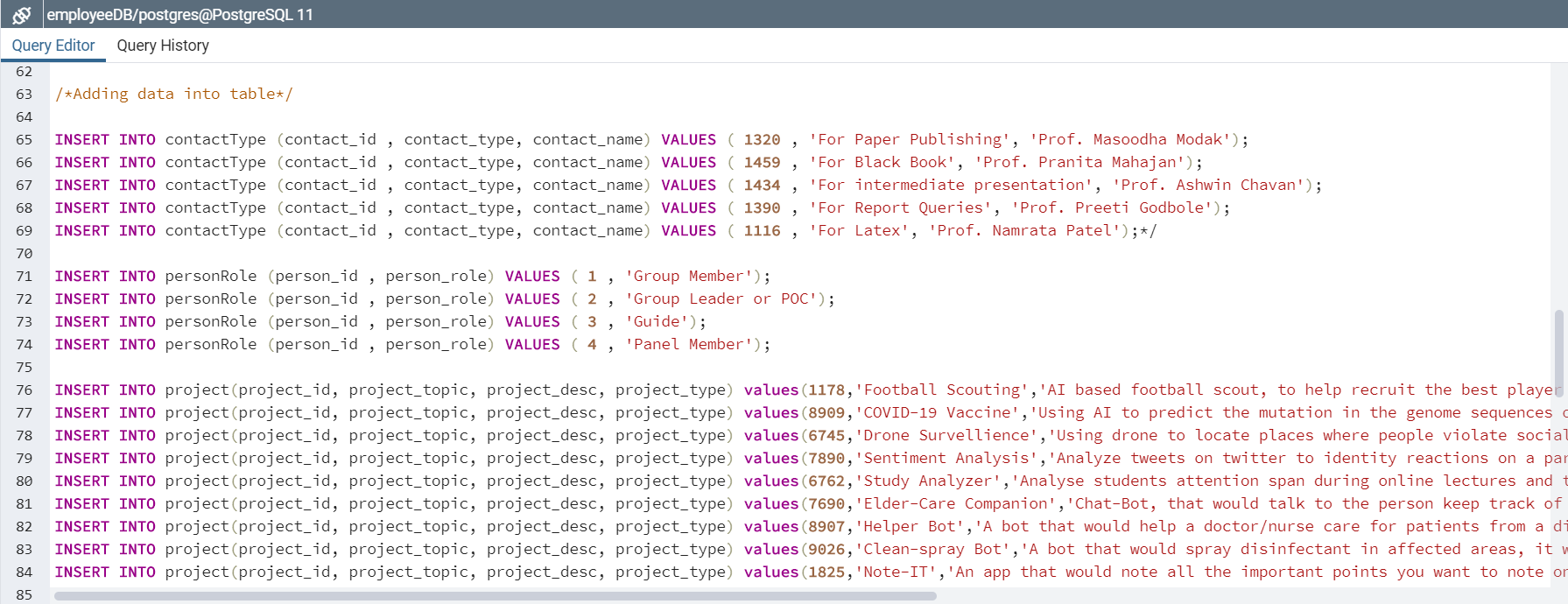


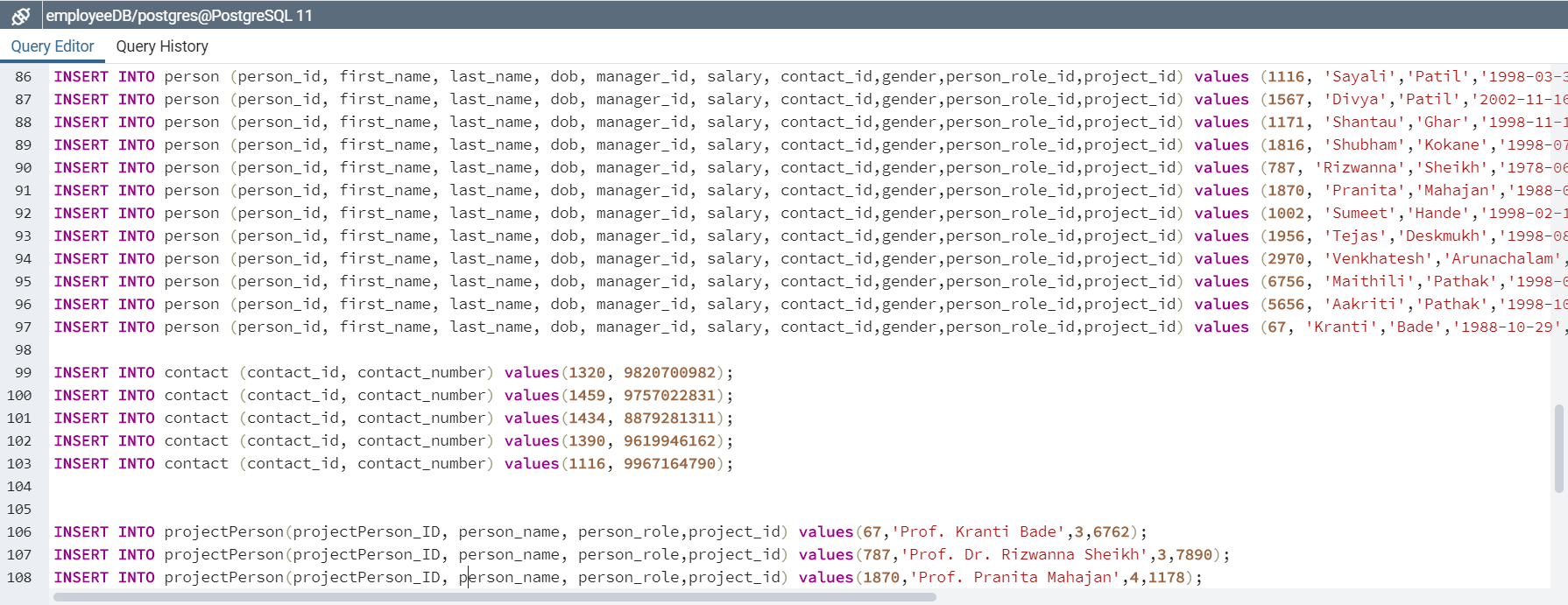


1. Tables Successfully created.
2. Alter Table Queries.

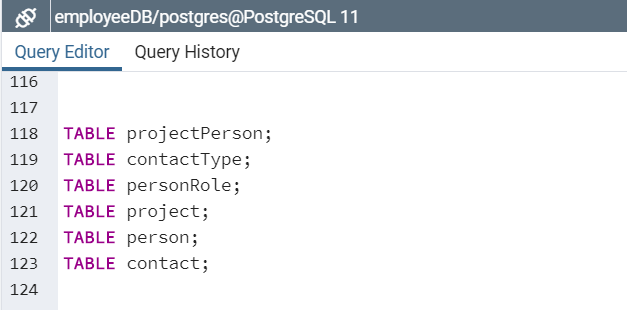


1. Adding data into Tables.

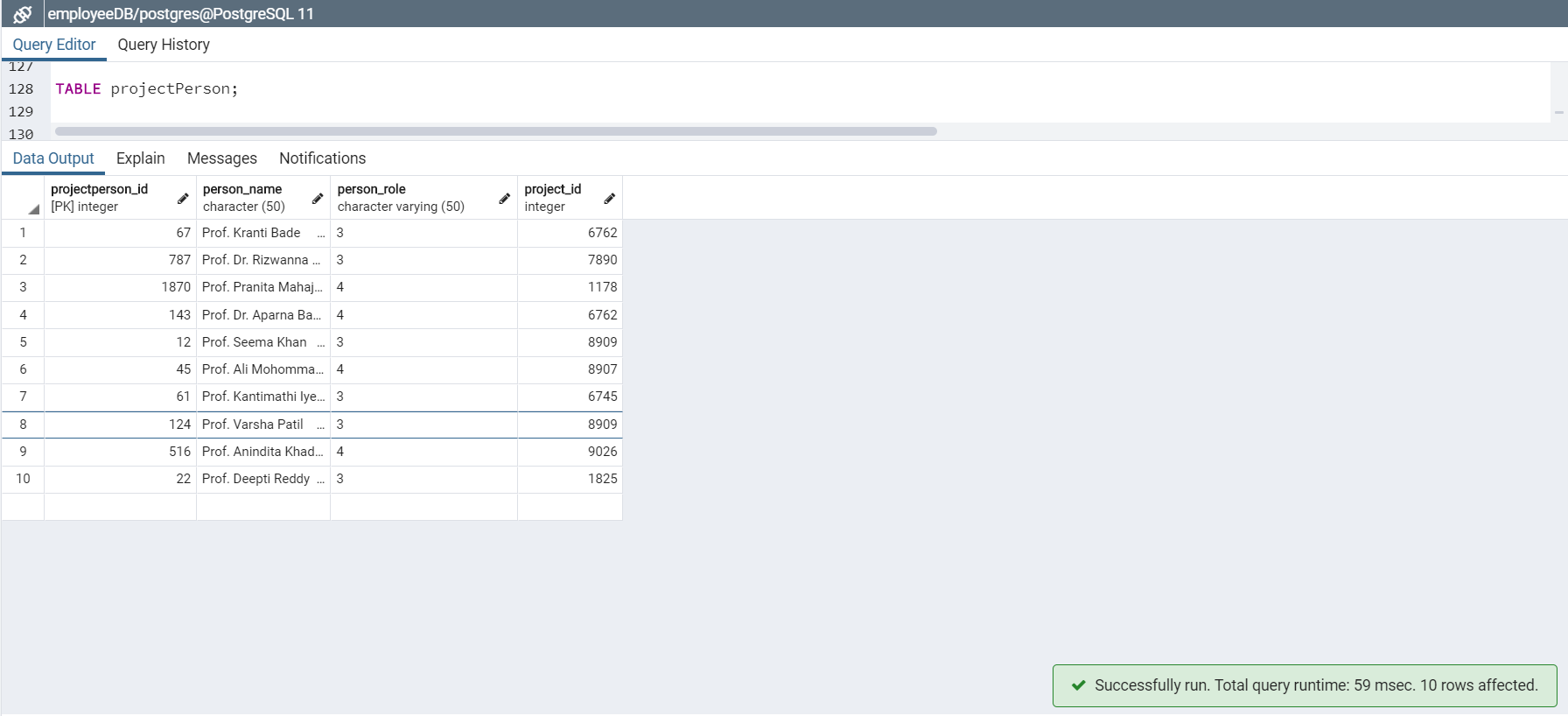


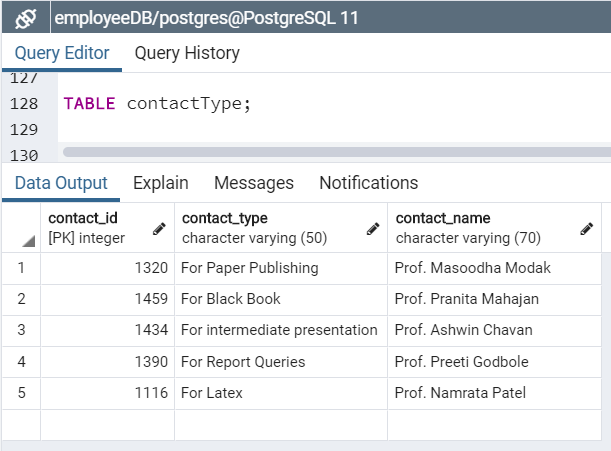


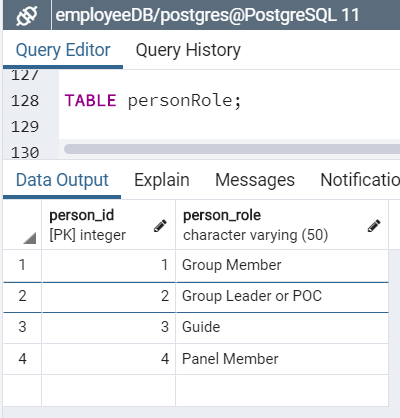
1. Print Tables.

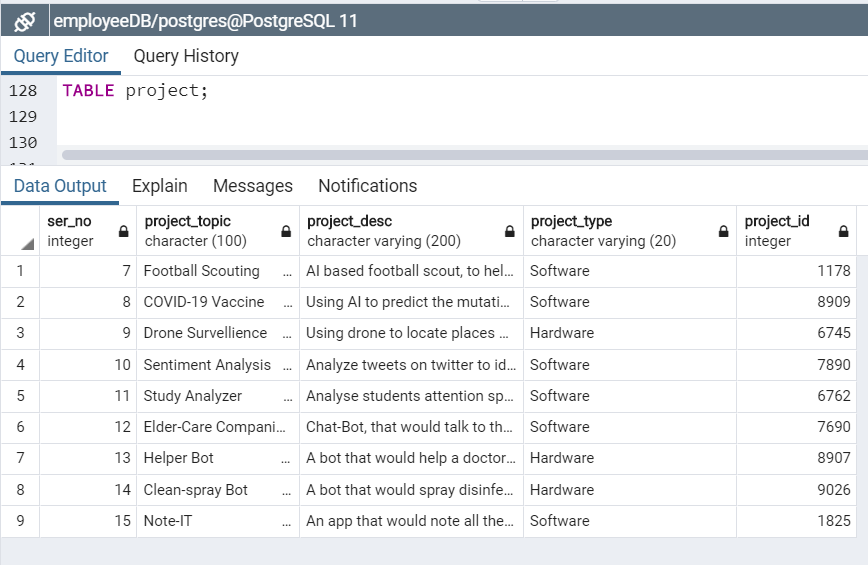


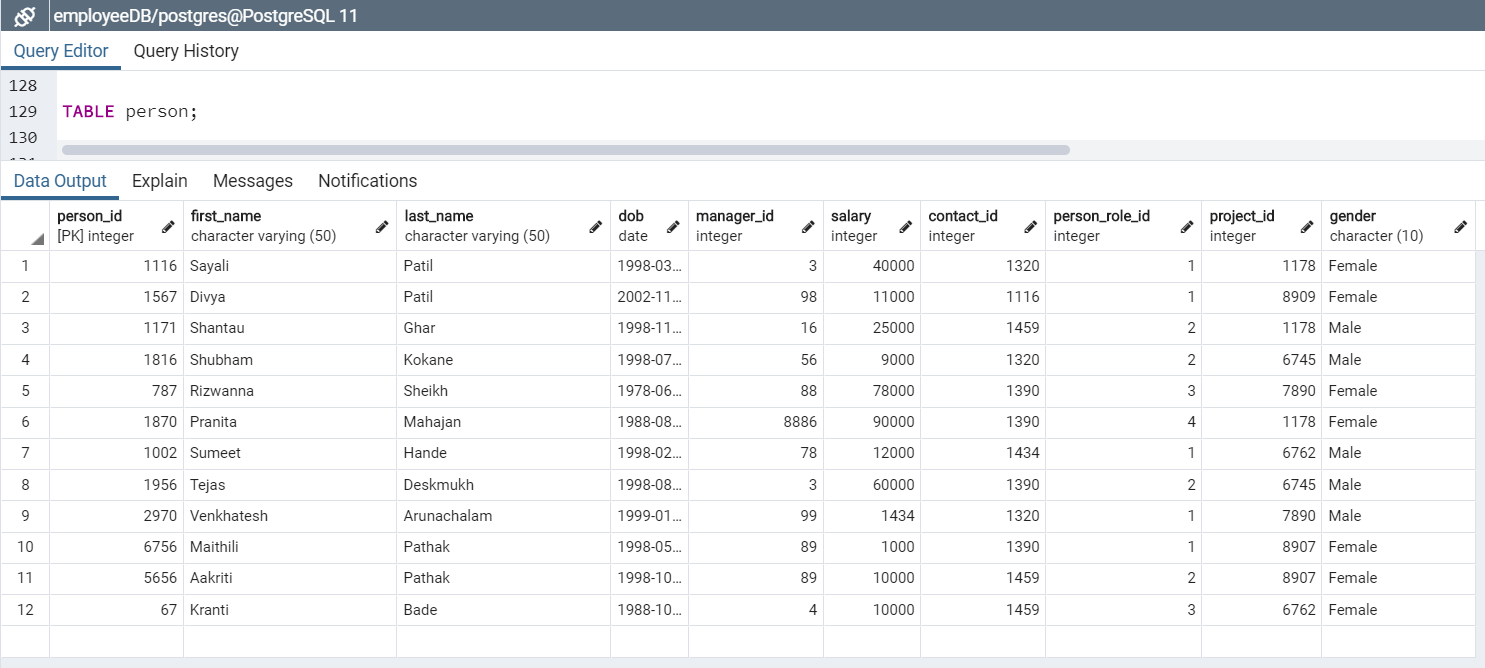
1. Output



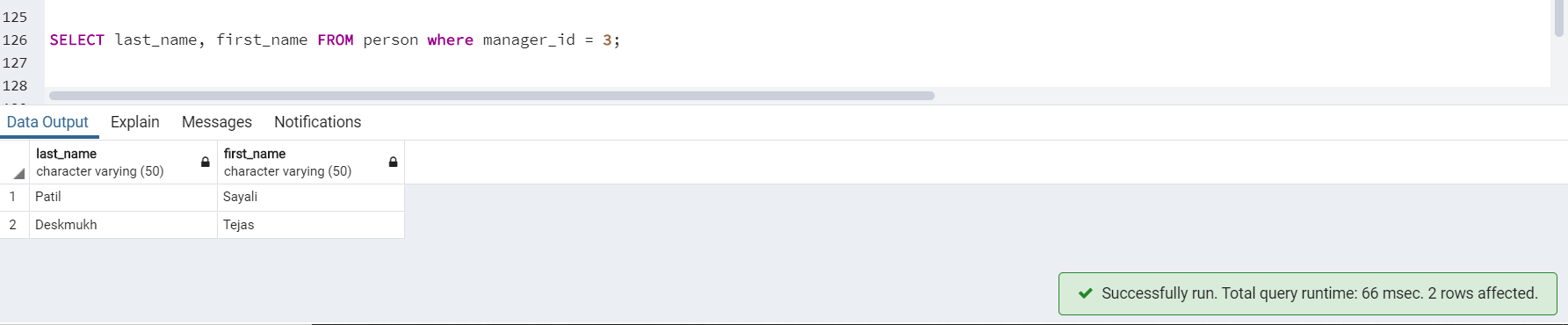






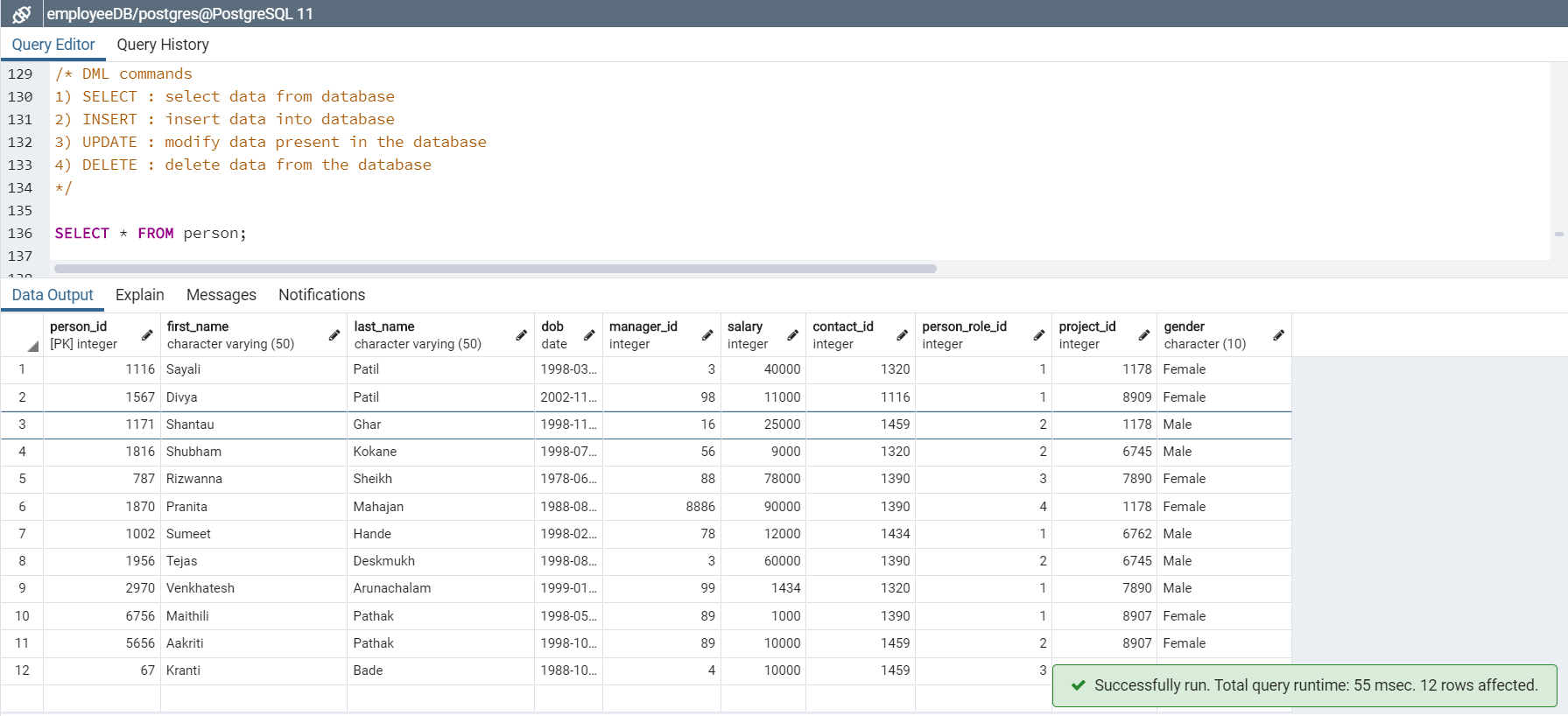


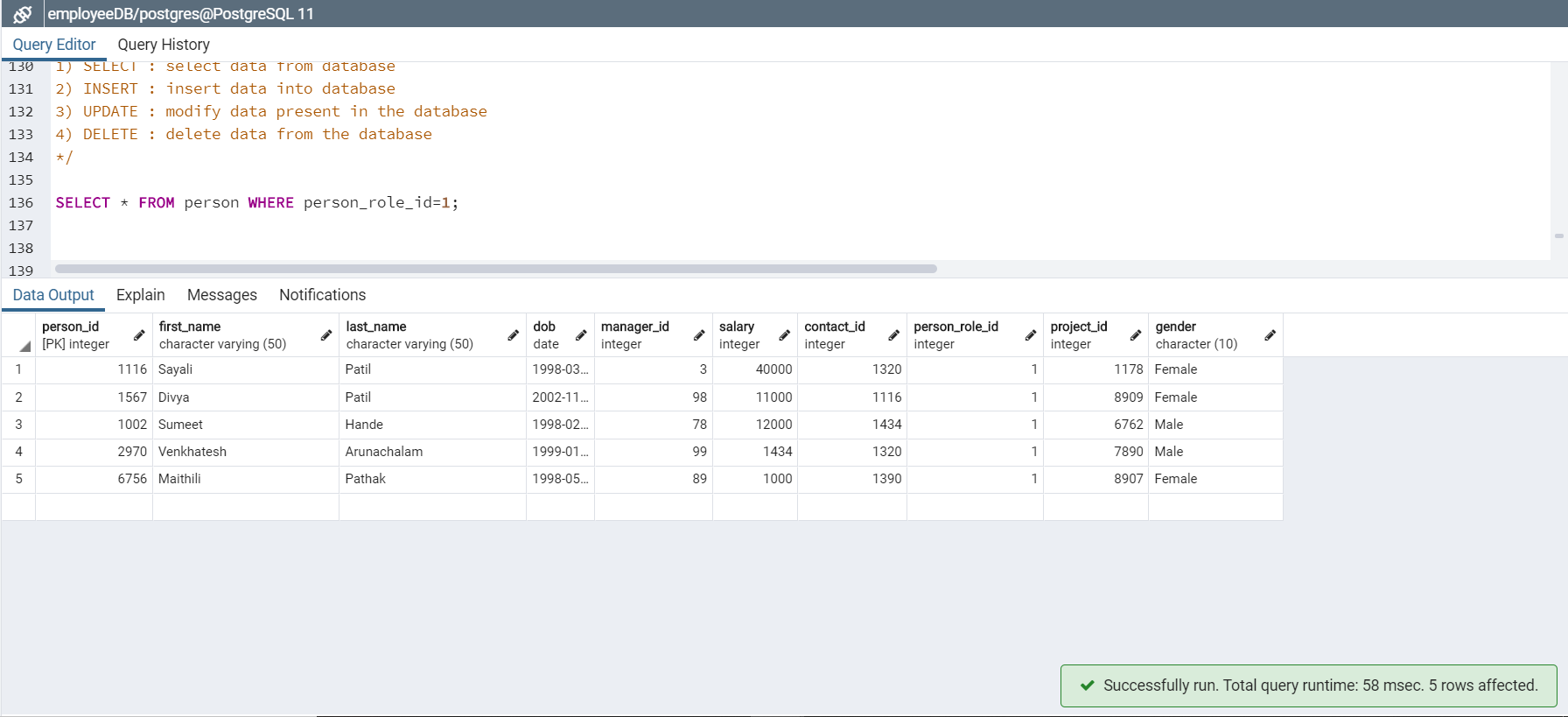
1. Query to display last\_name and first\_name for manager\_id = 3



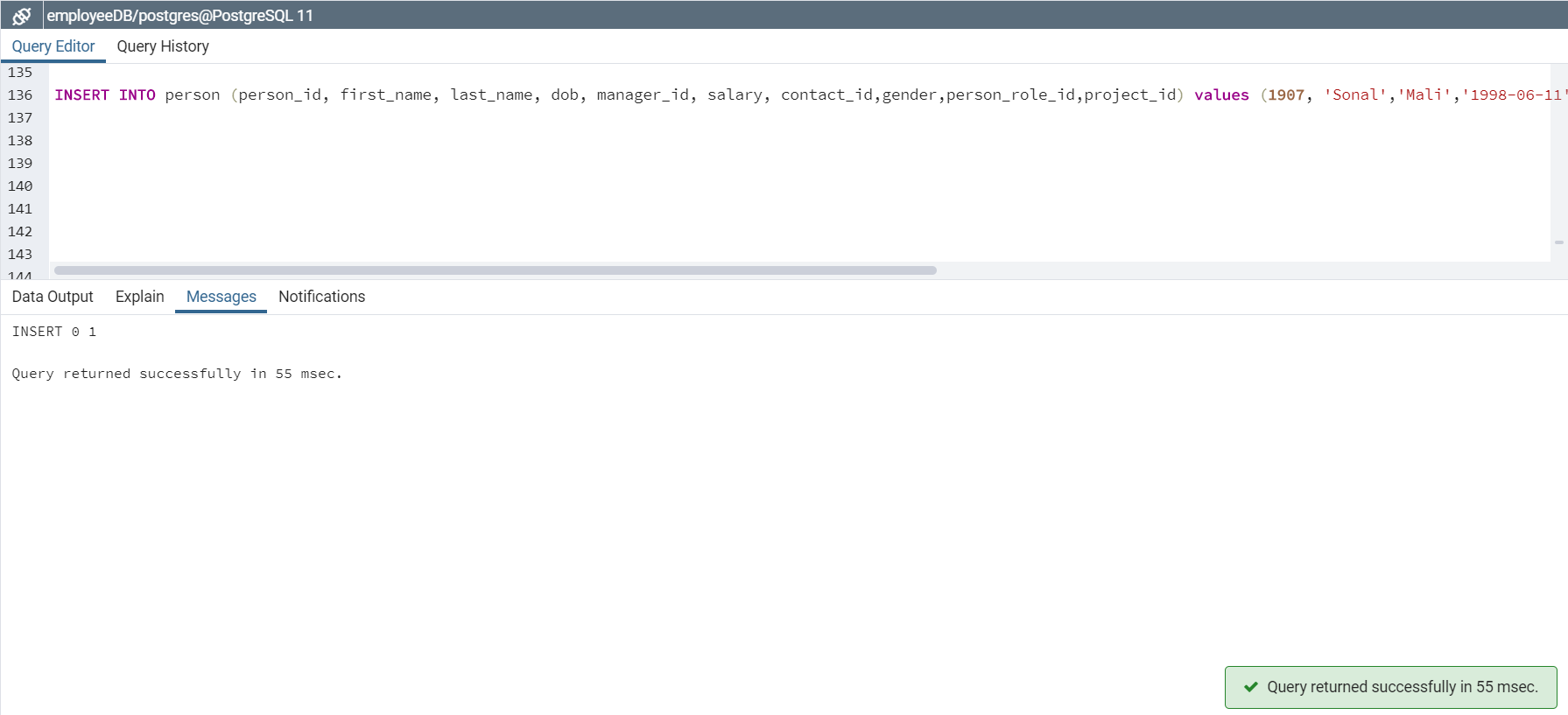
1. Some DML Commands on **person** table

* SELECT

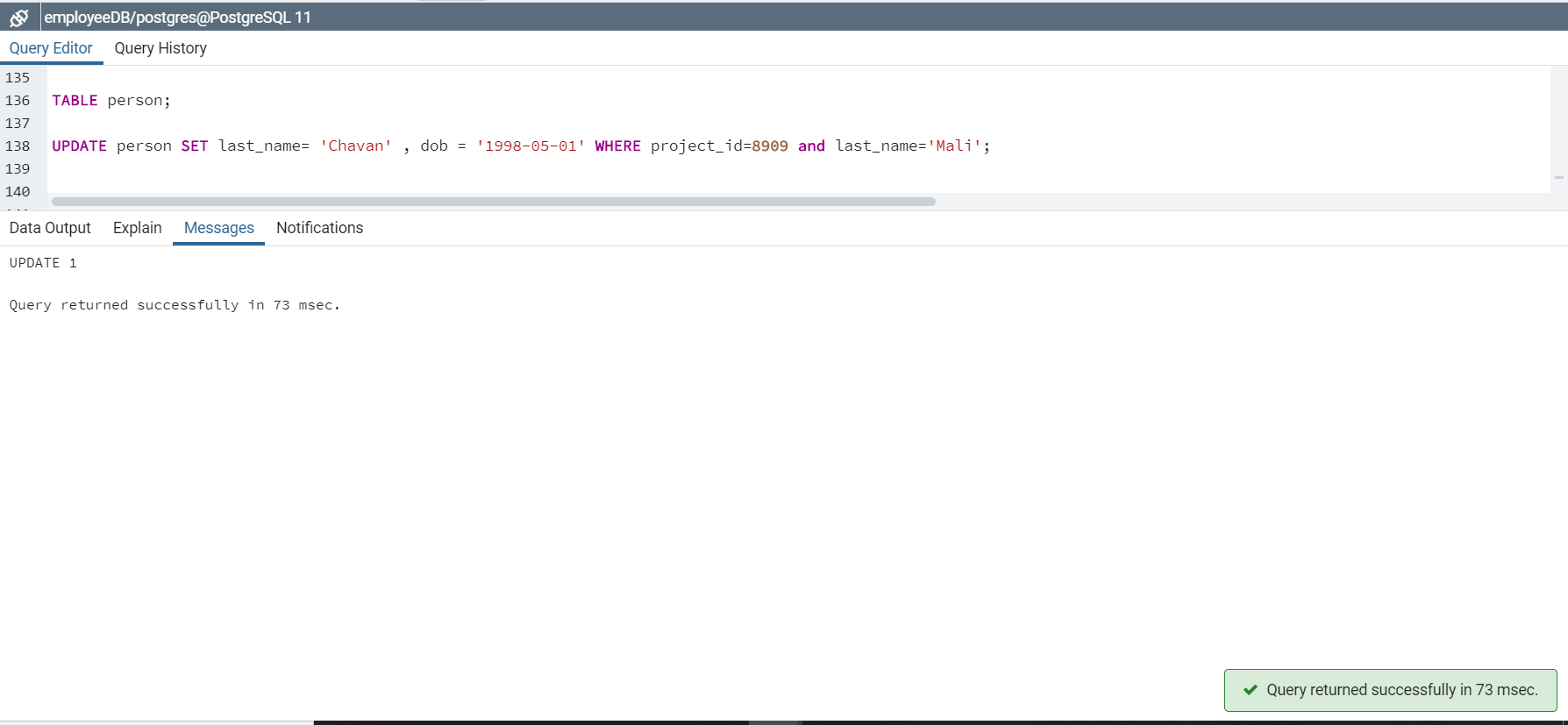




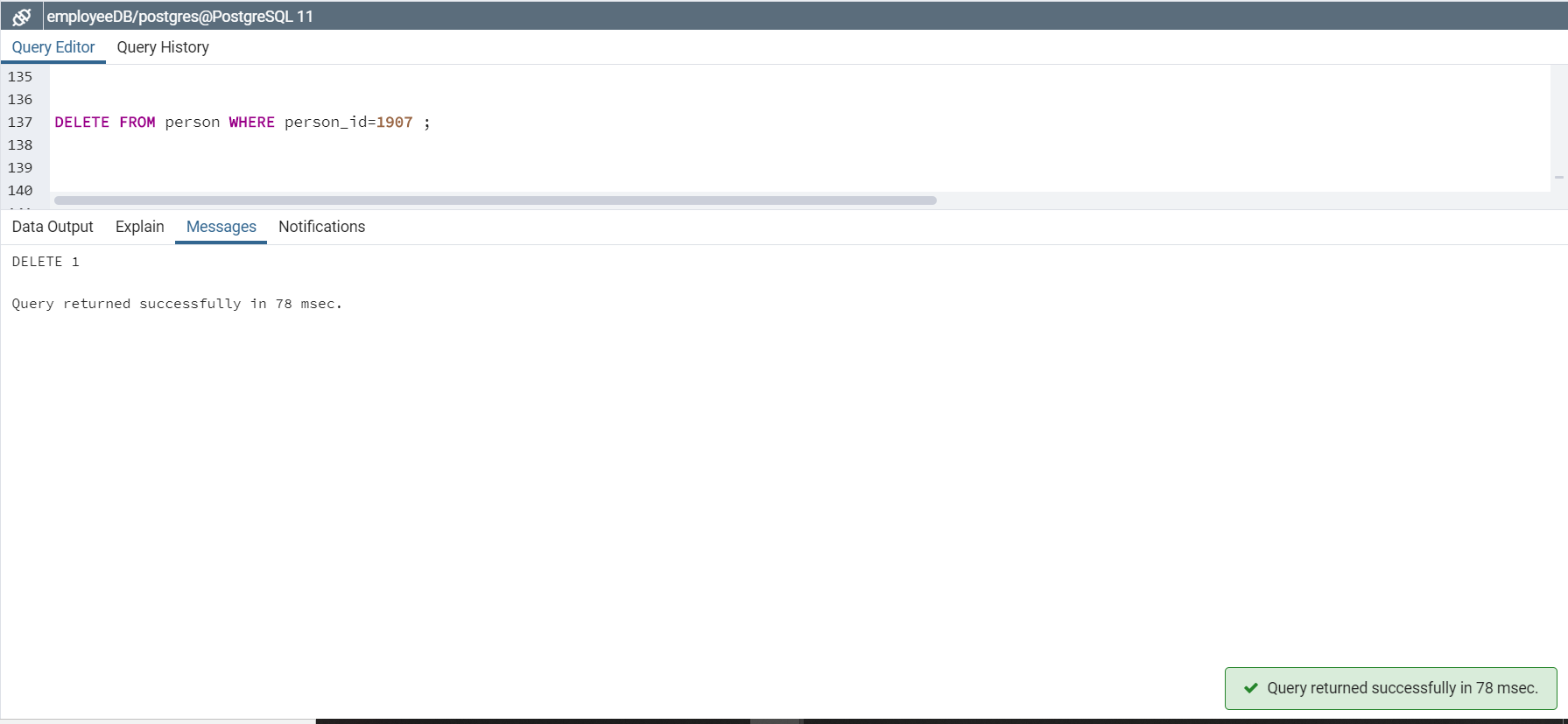
* INSERT



* UPDATE

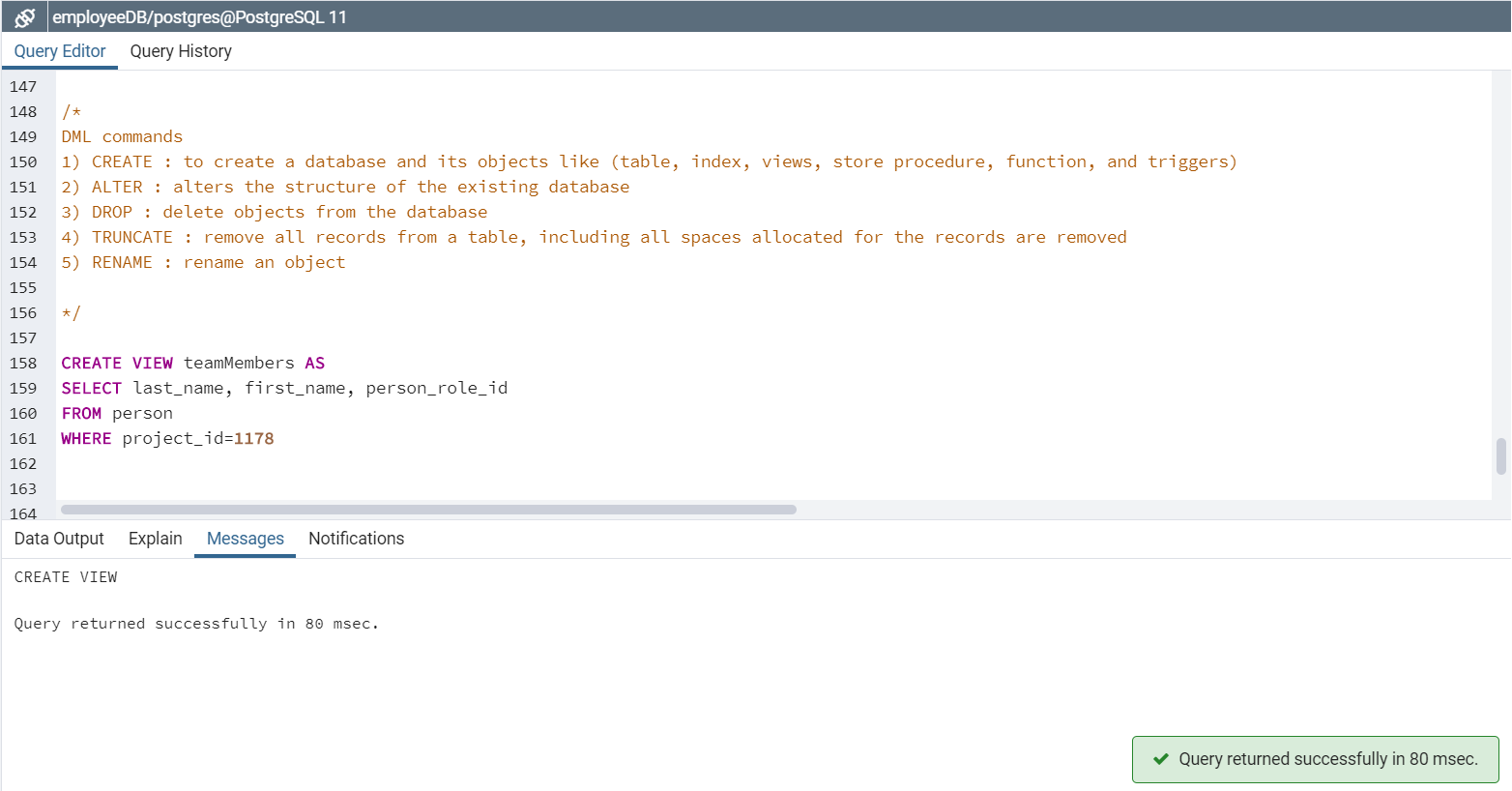


* DELETE

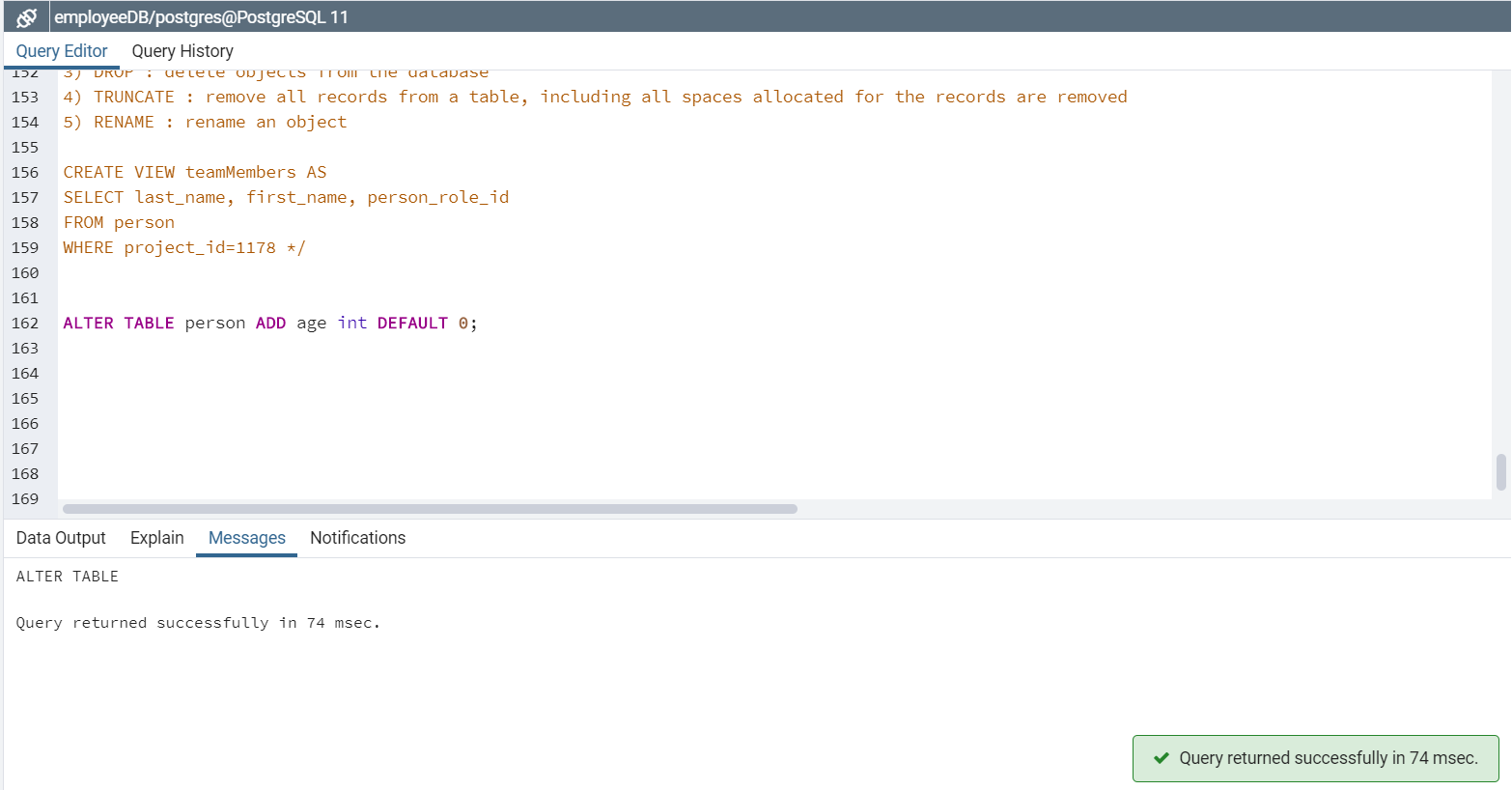


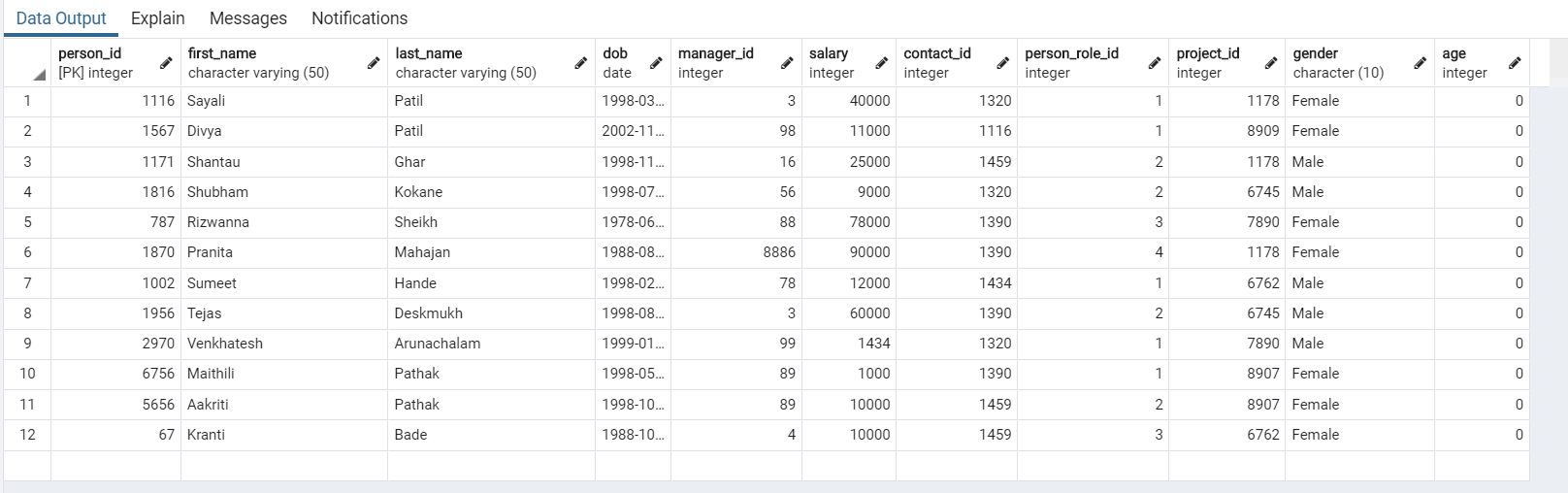
1. Some DDL commands

* CREATE

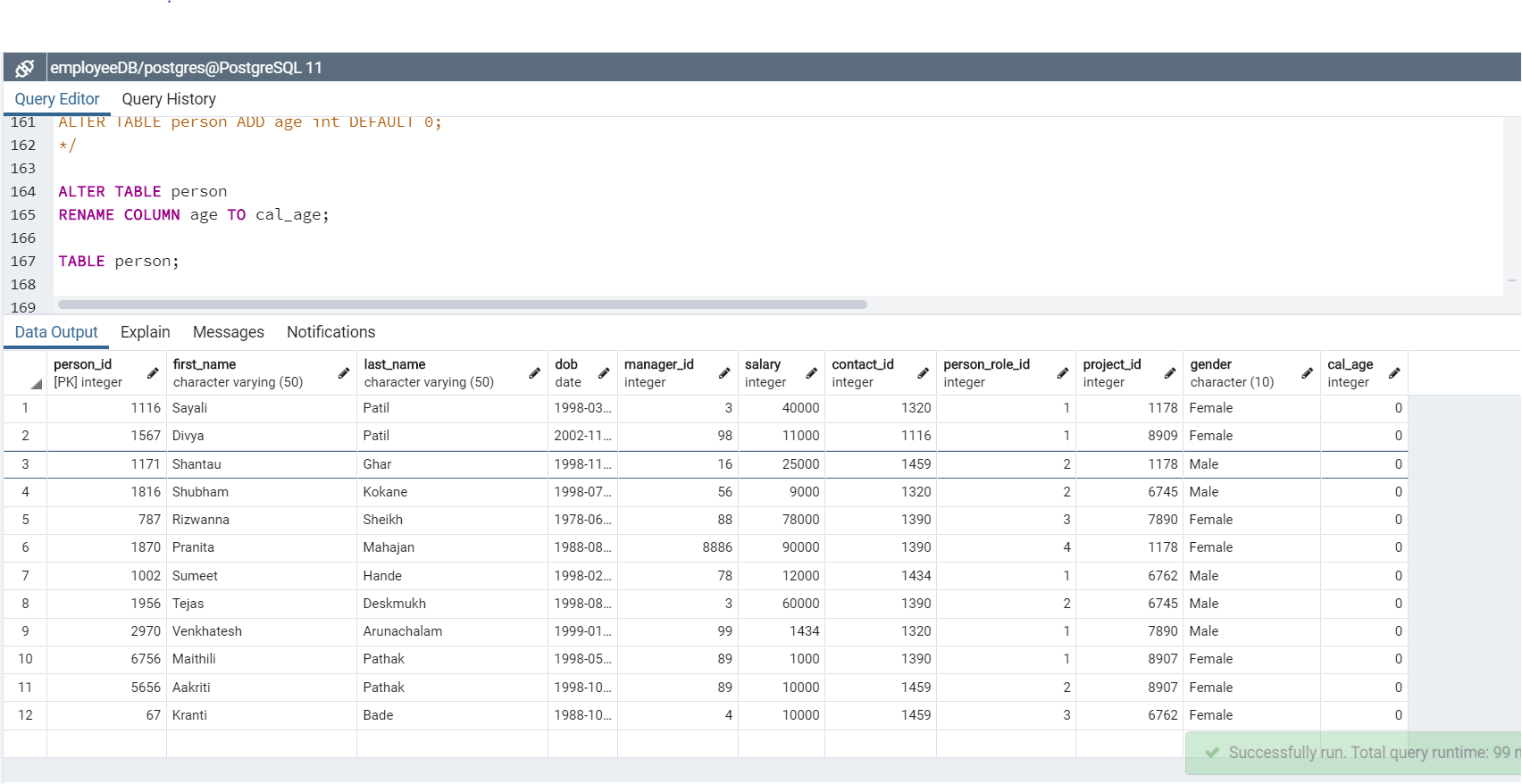


* ALTER



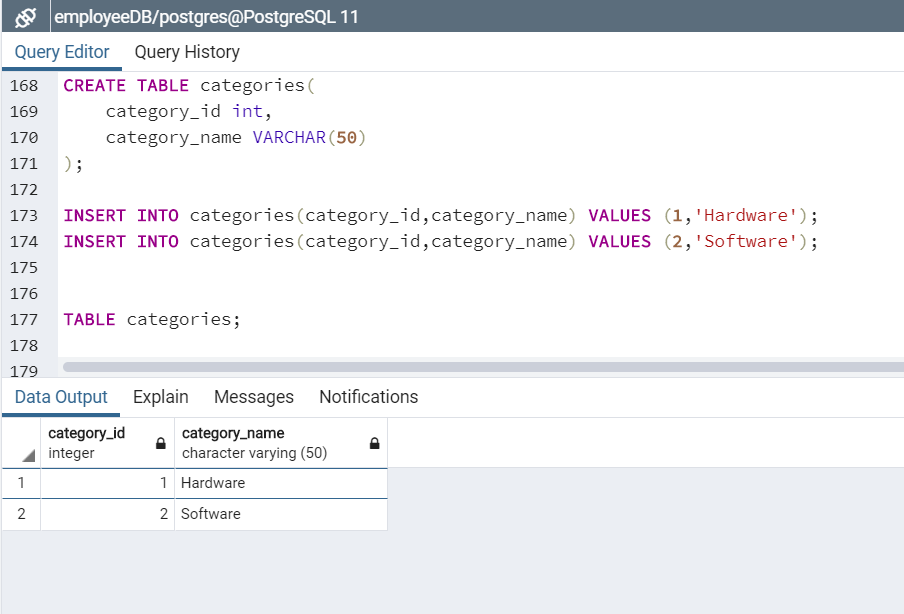


* RENAME

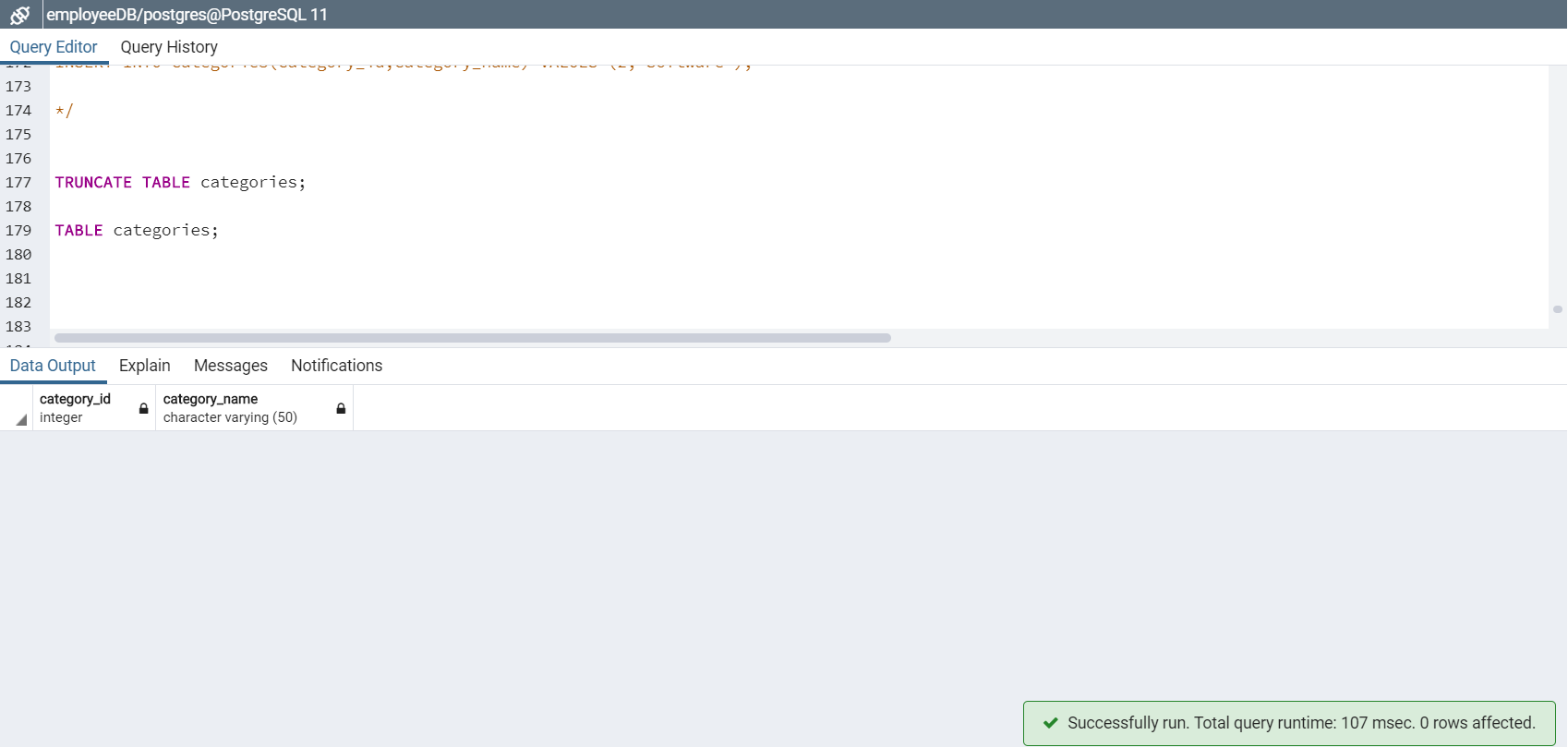


* TRUNCATE

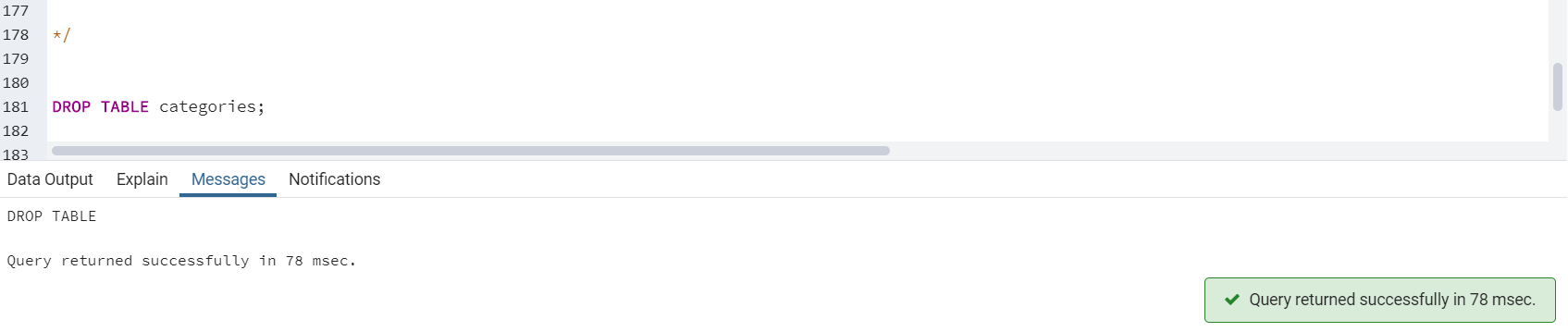
Before TRUNCATE



After TRUNCATE

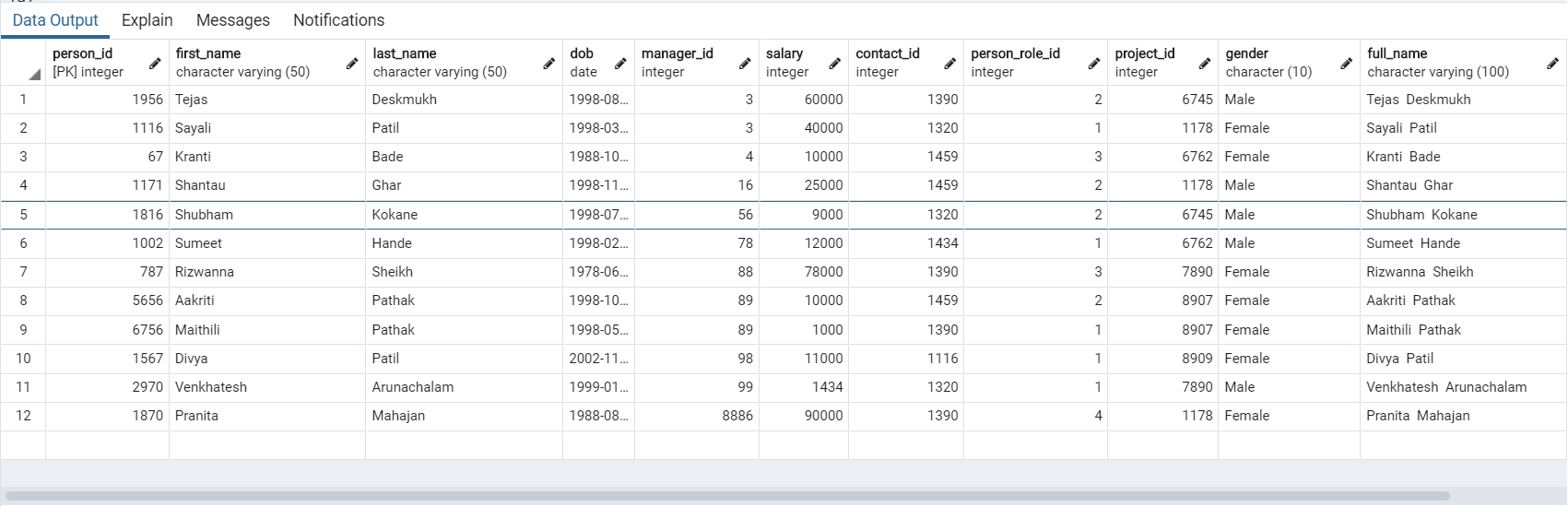


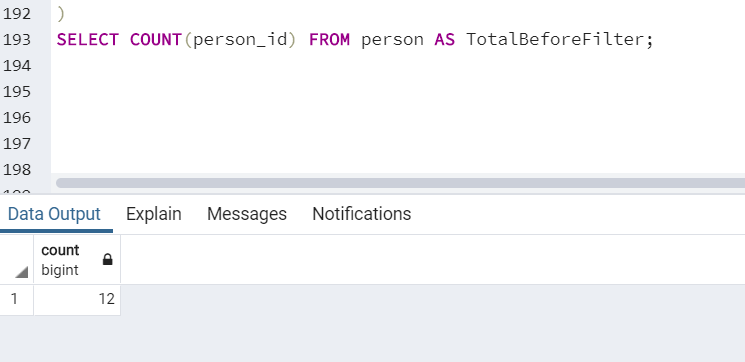
* DROP



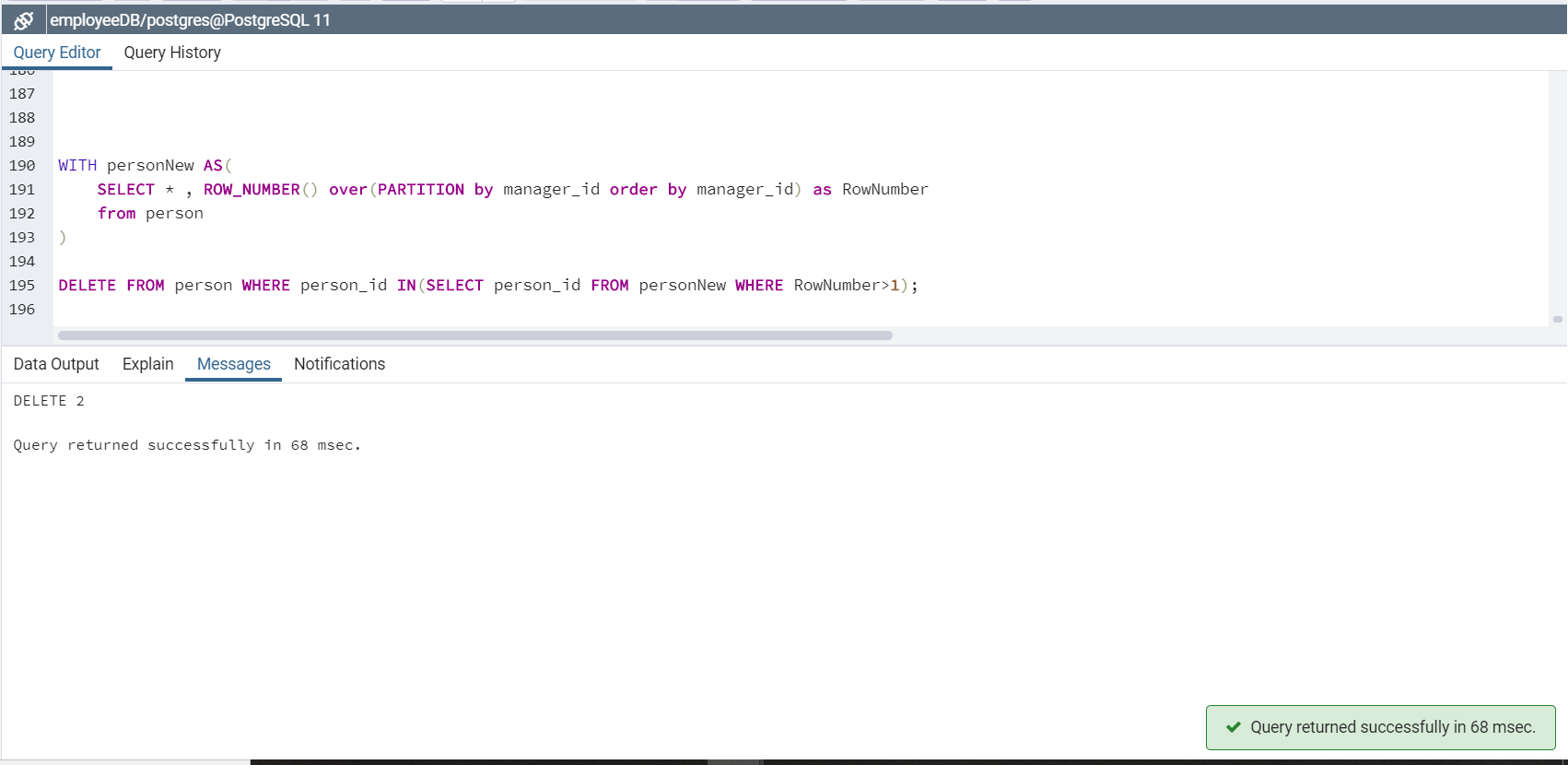
1. Filter out all duplicate manager\_id

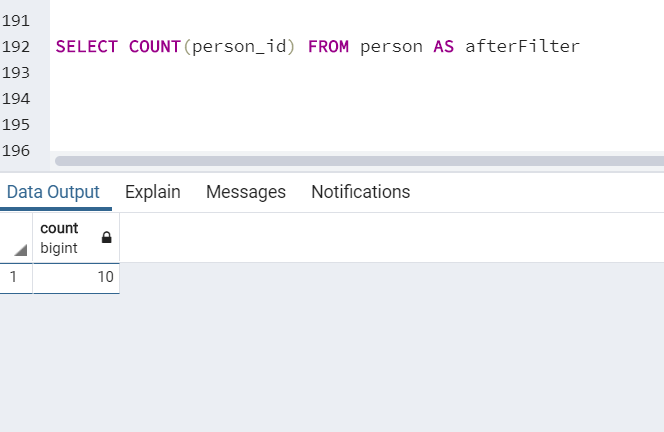
Before filtering



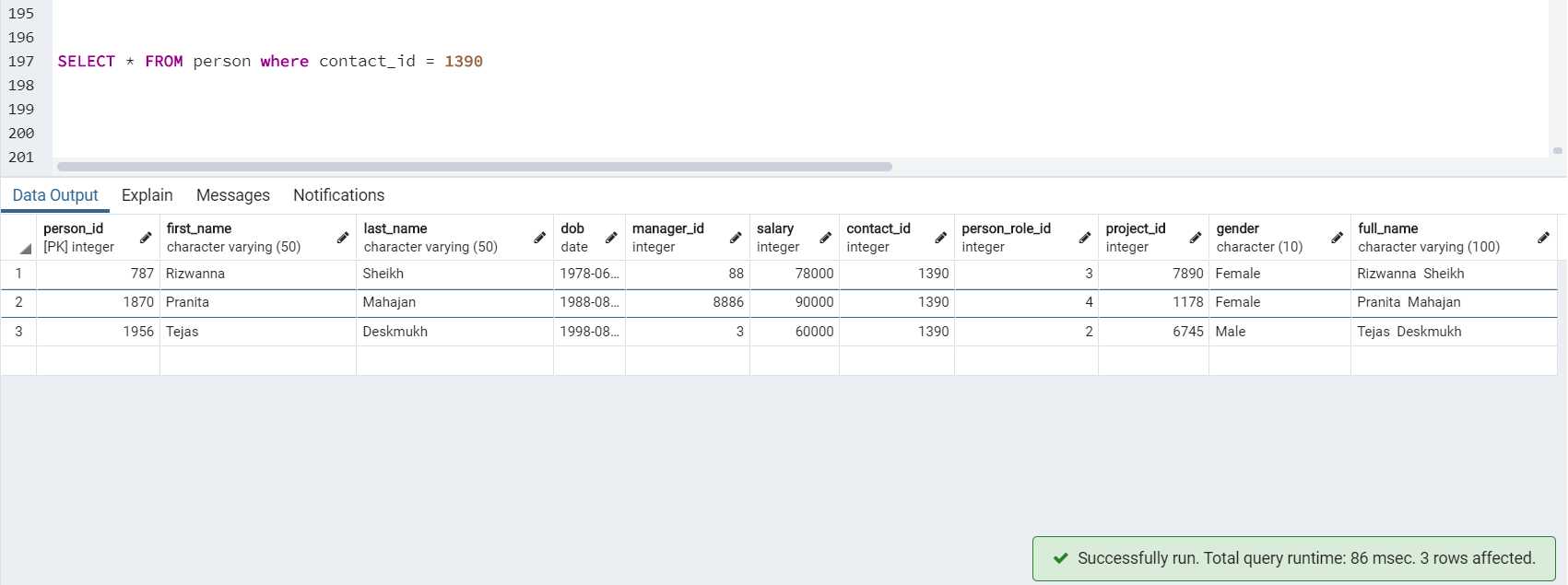


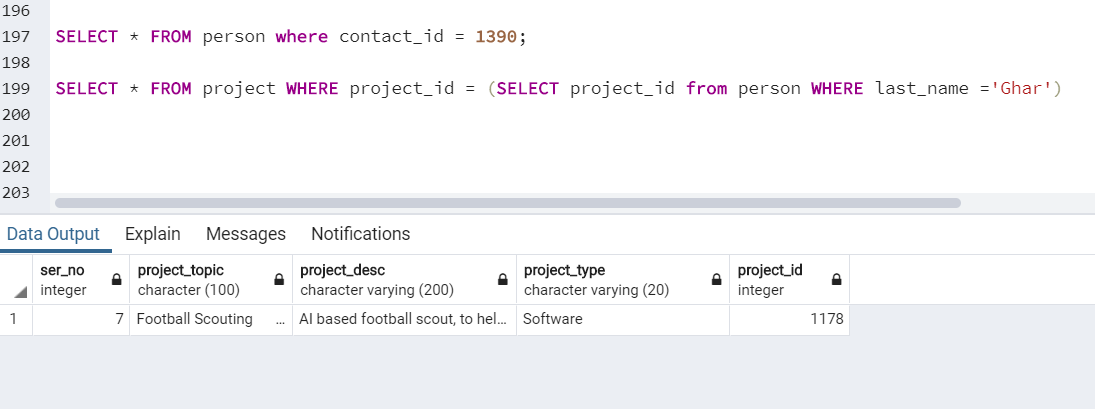
After Filtering



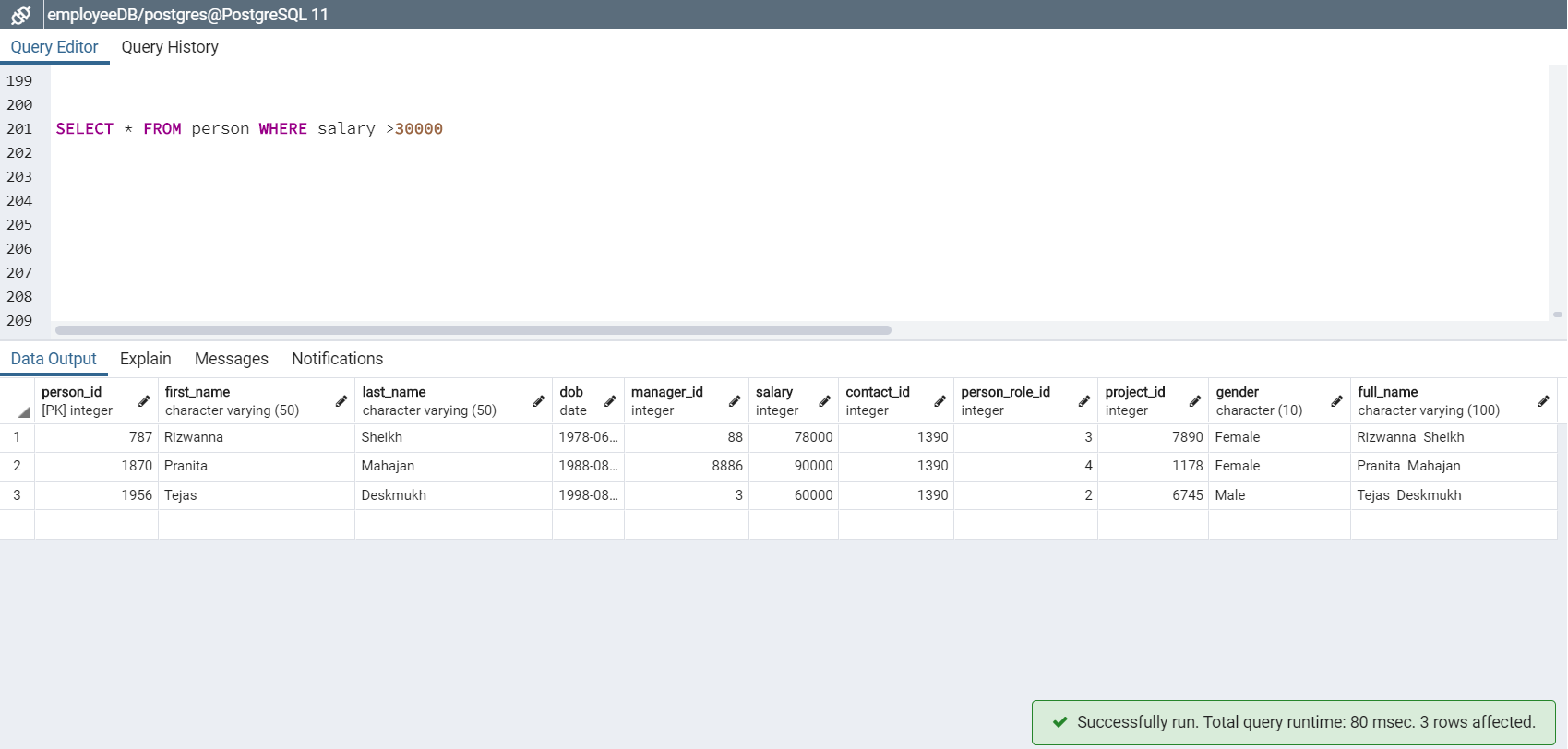


1. Queries with WHERE clause

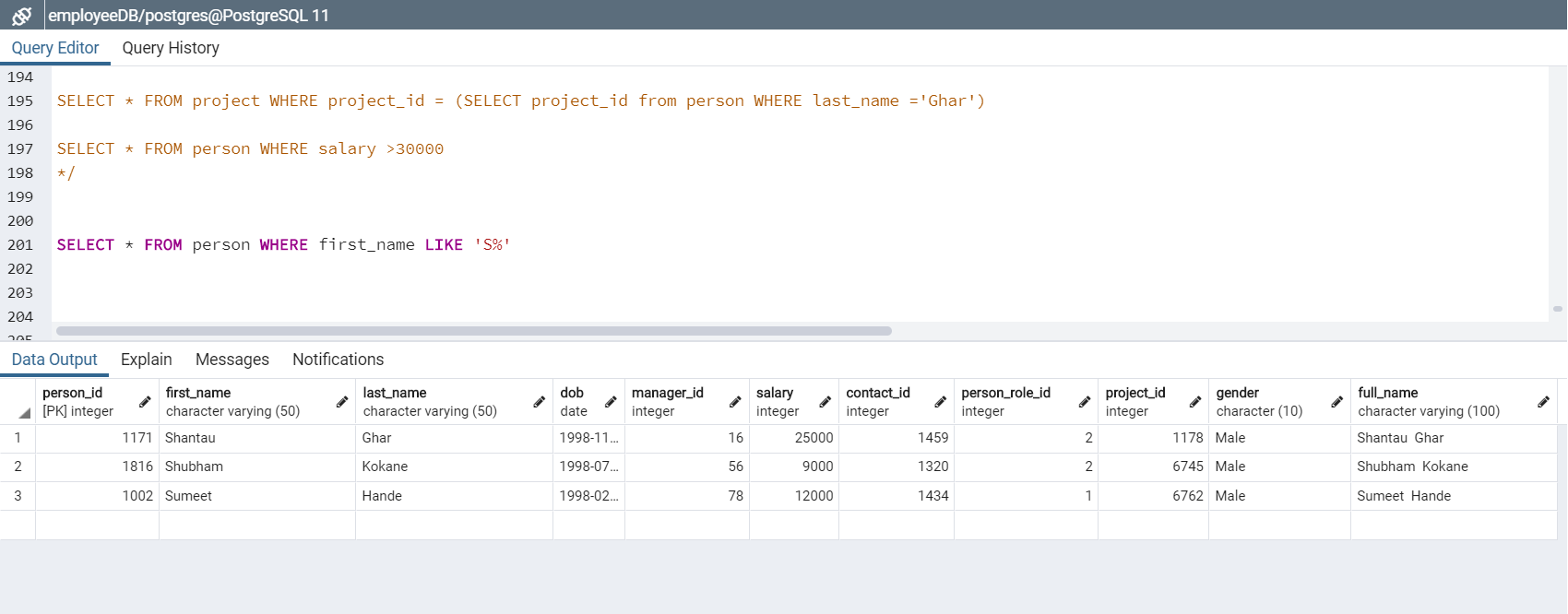




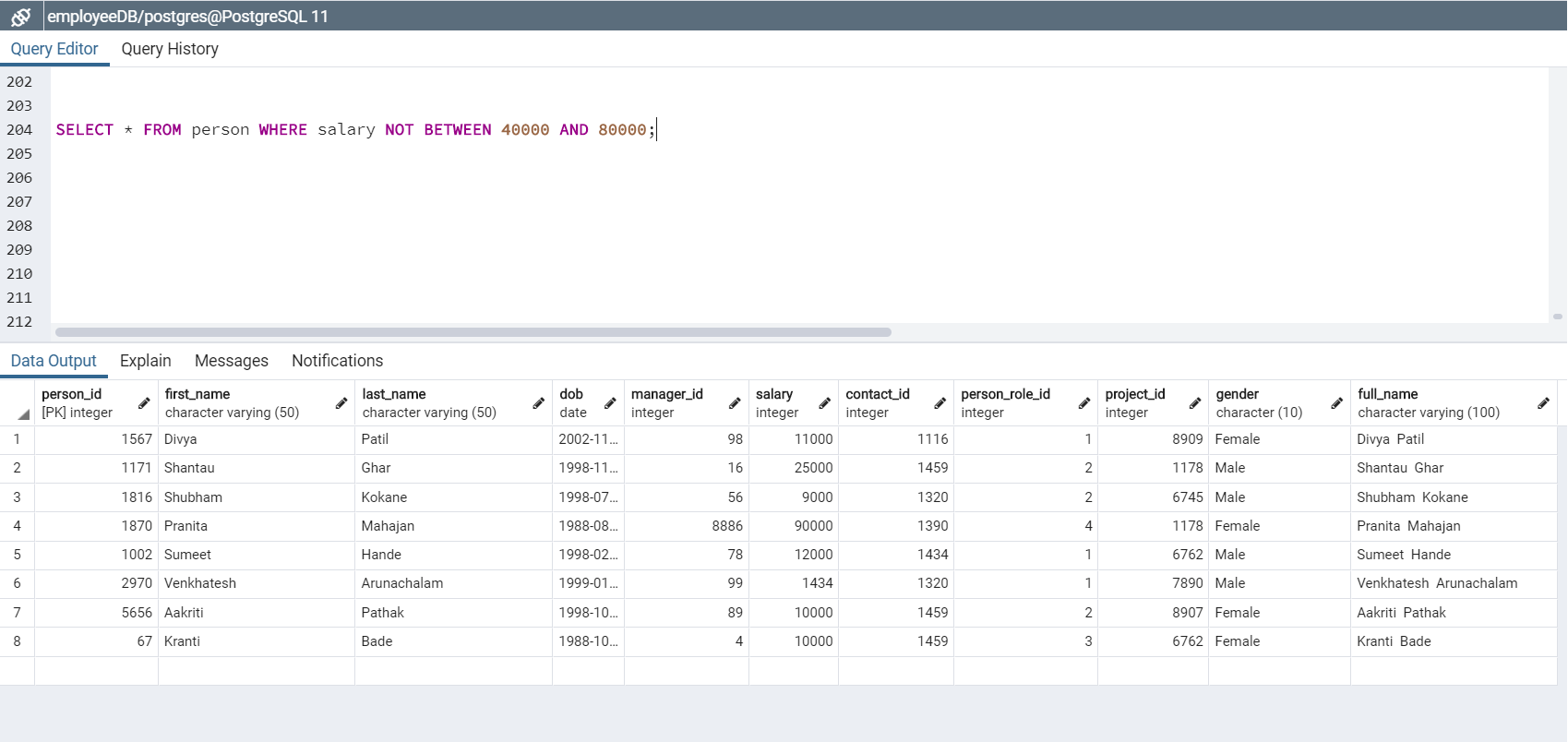
1. Comparison operator



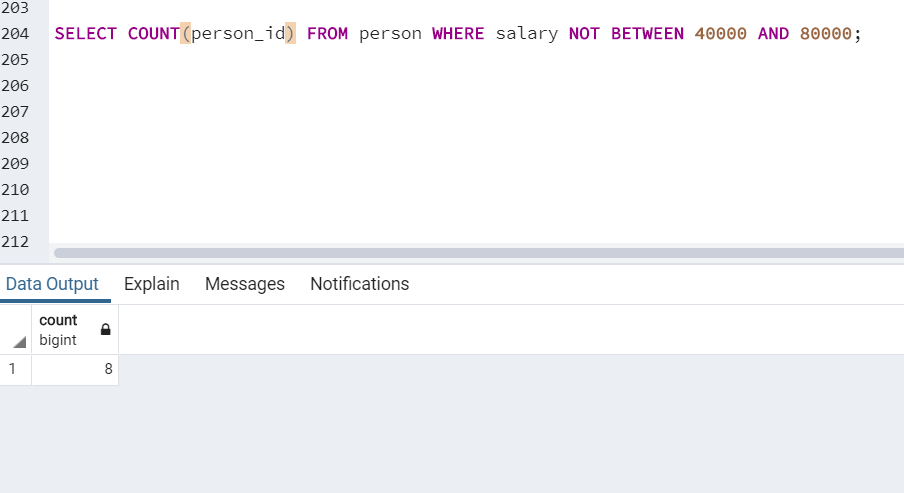
1. LIKE clause

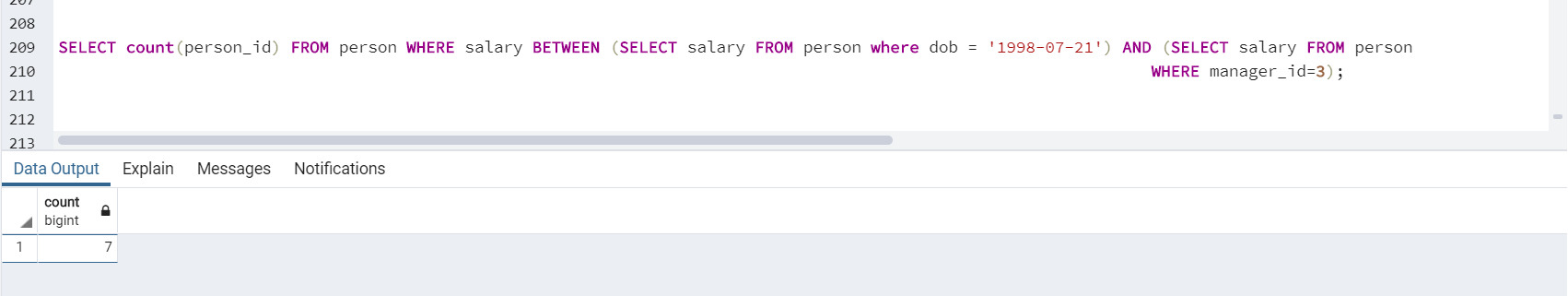


1. BETWEEN clause

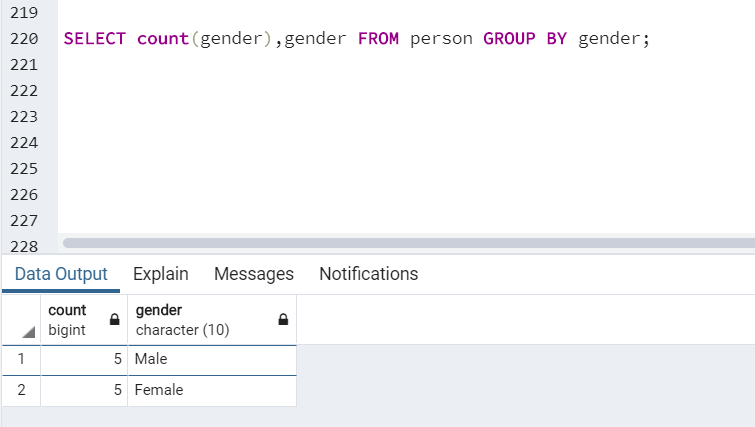


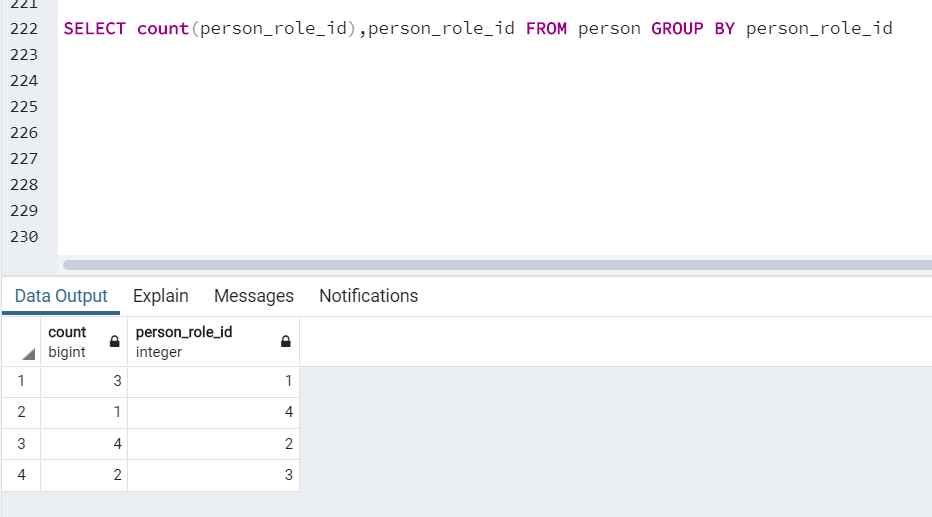
1. COUNT clause



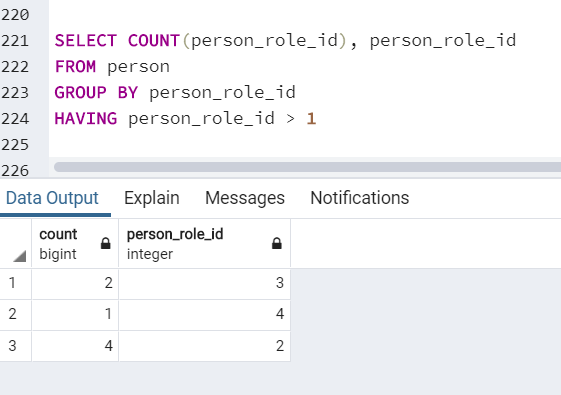


1. Using Group By clause on person table



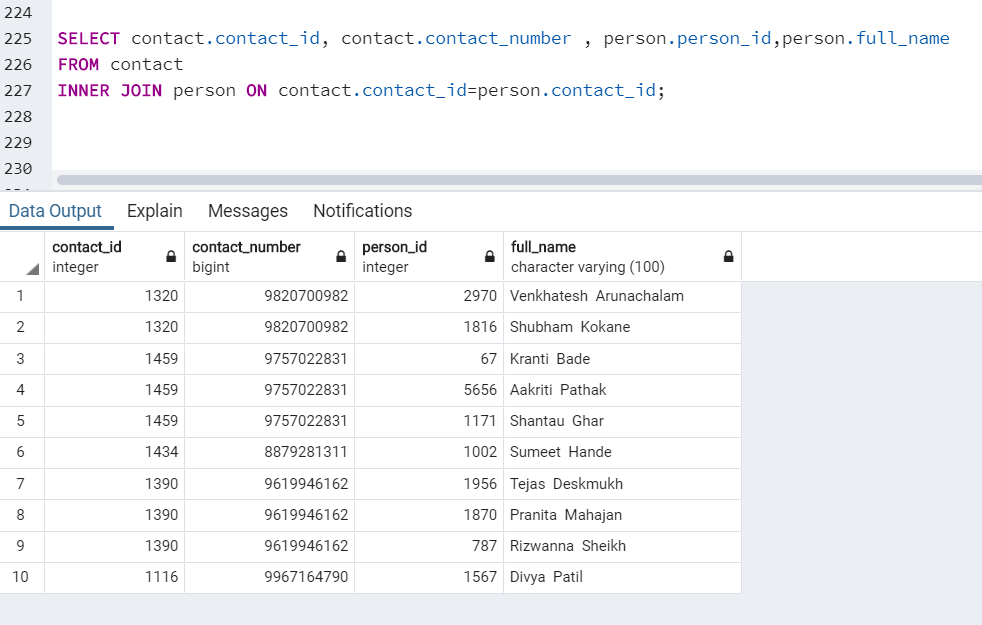


1. Using Having Clause

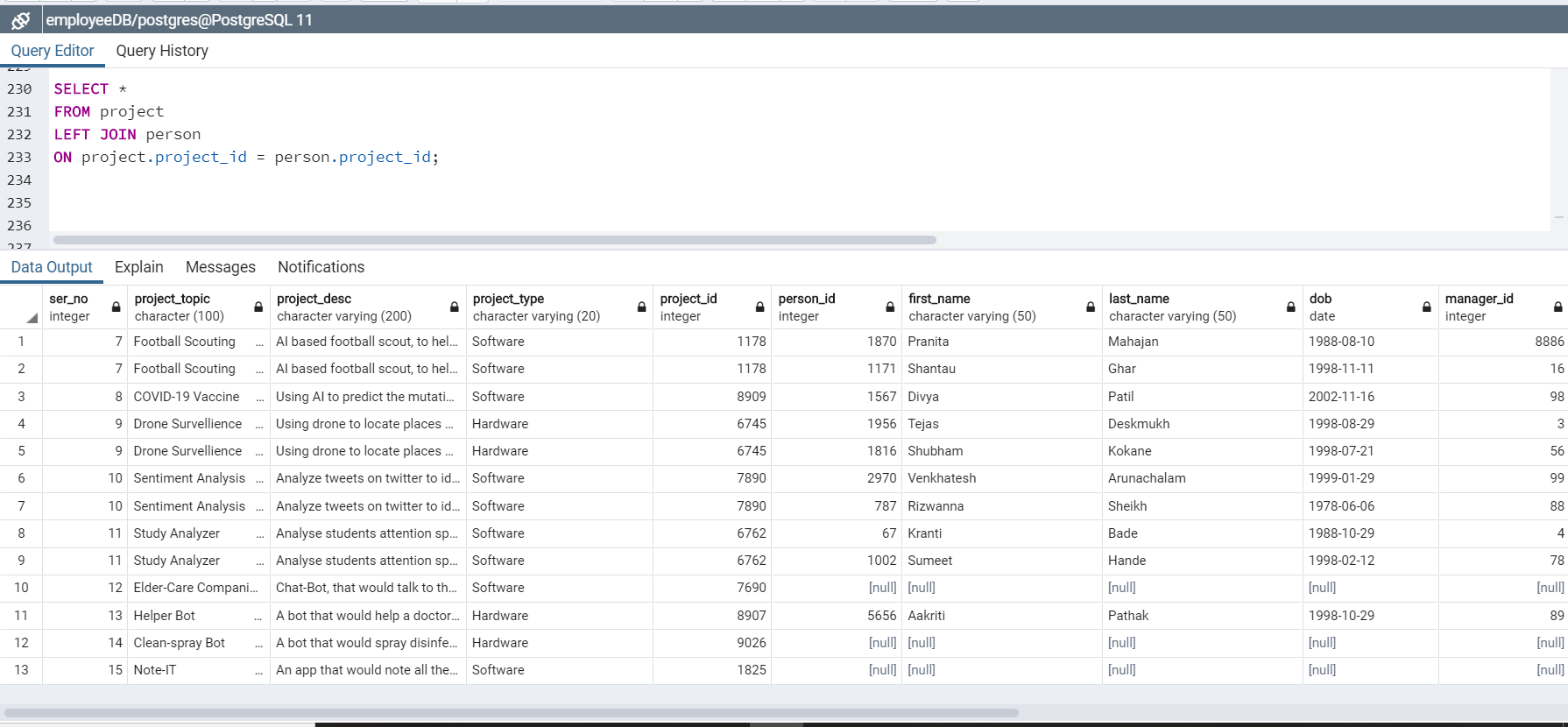


1. JOIN Clause

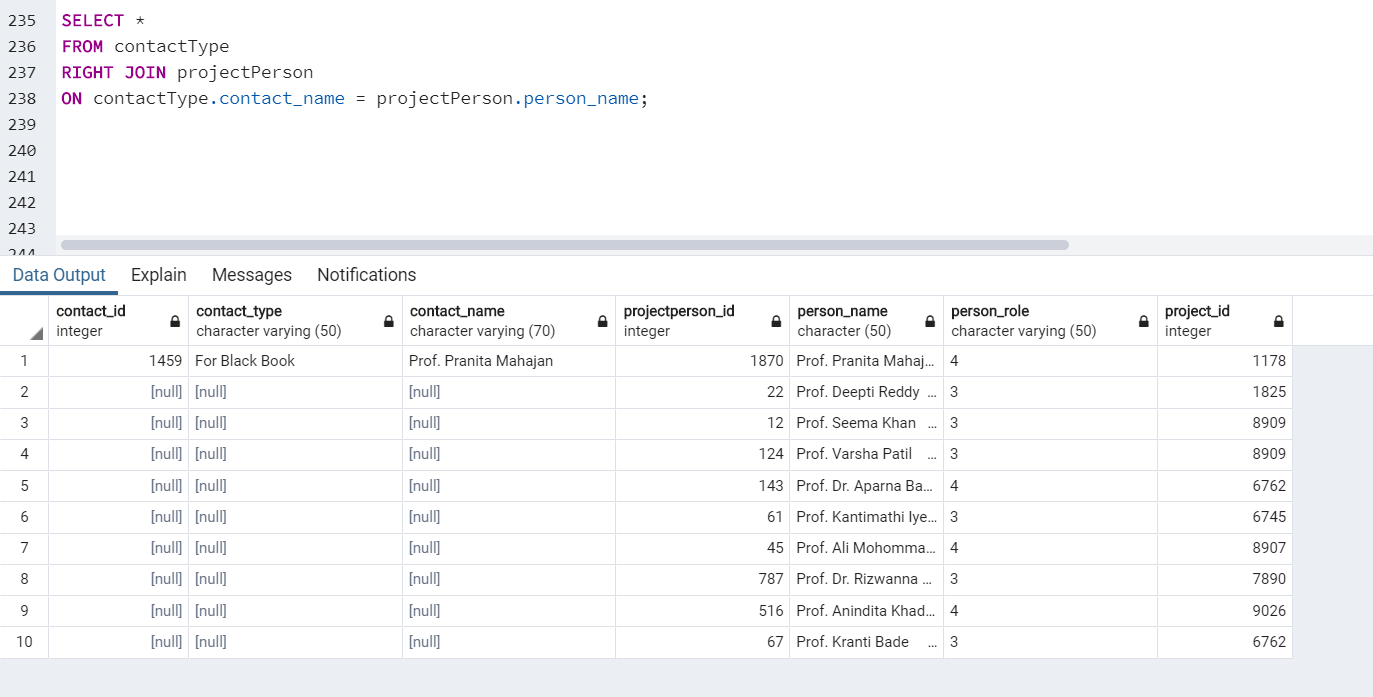
* INNER JOIN: Returns records that have matching values in both tables.



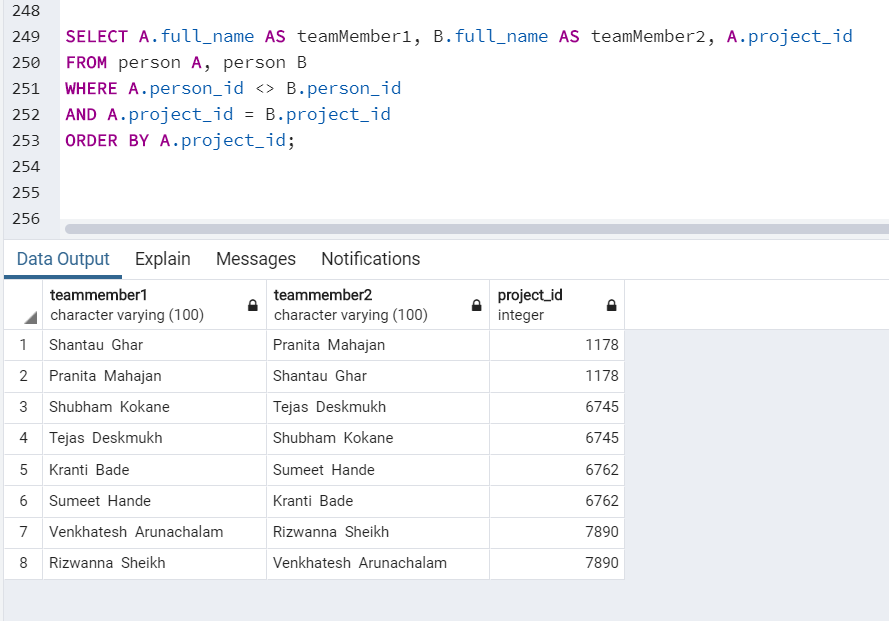
* LEFT INNER JOIN: Returns all records from the left table, and the matched records from the right table



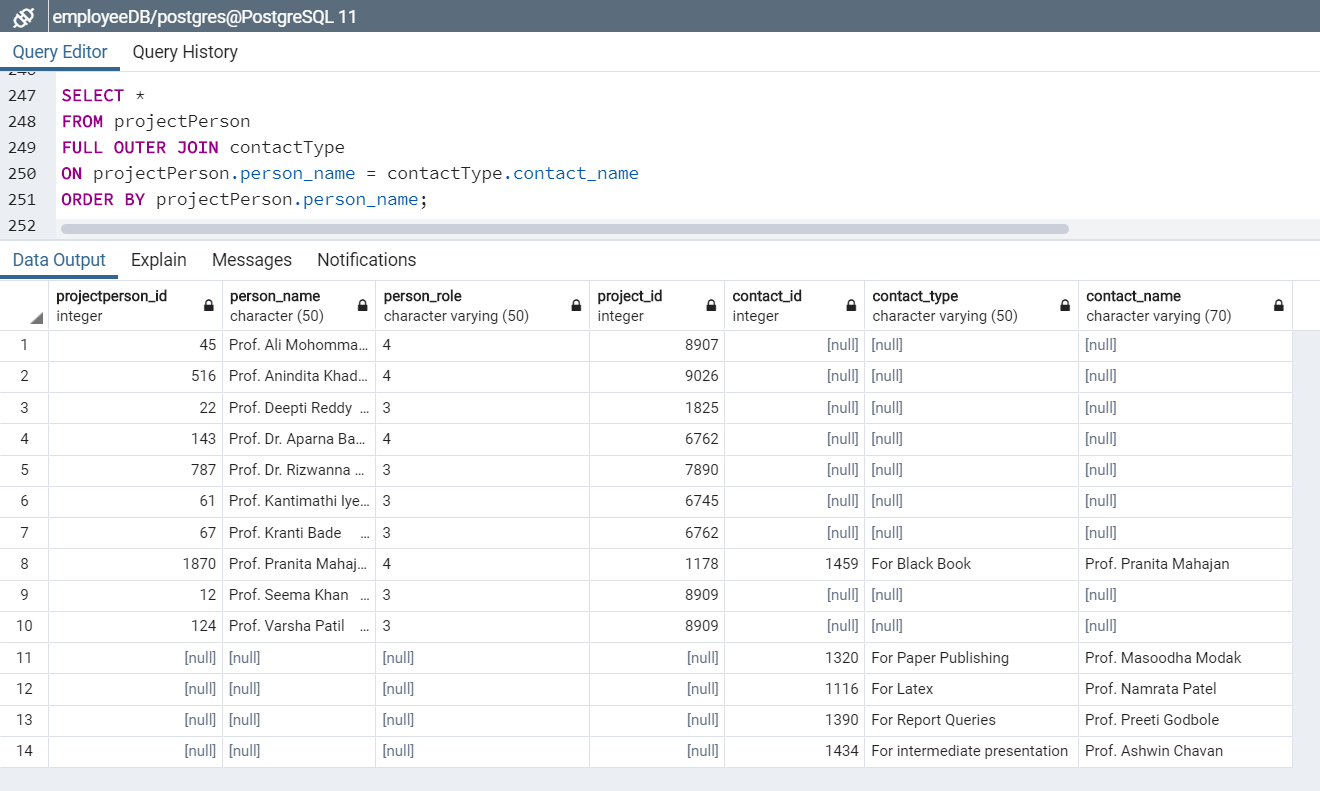
* RIGHT INNER JOIN: Returns all records from the right table, and the matched records from the left table.



* SELF JOIN: A self-JOIN is a regular join, but the table is joined with itself.

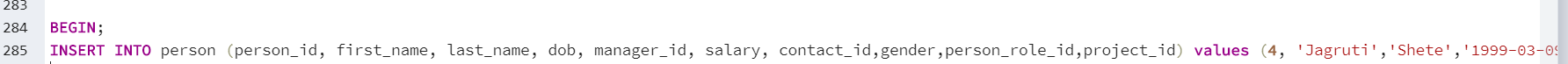


* FULL OUTER JOIN: Returns all records when there is a match in either left or right table

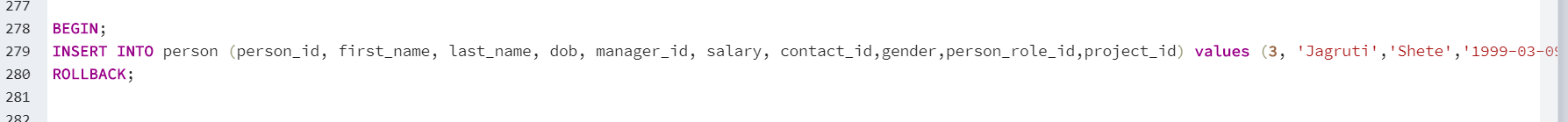


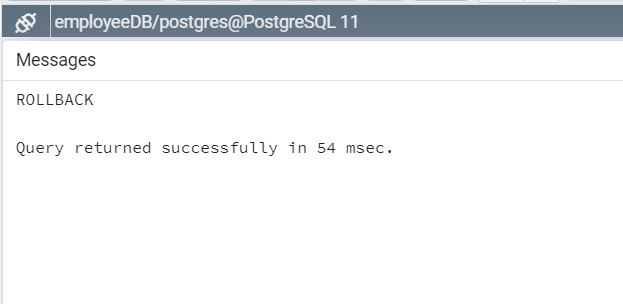
1. Initiating a transaction

Transactions can be started using BEGIN TRANSACTION or simply BEGIN command. Such transactions usually persist until the next COMMIT or ROLLBACK command is encountered. But a transaction will also ROLLBACK if the database is closed or if an error occurs.

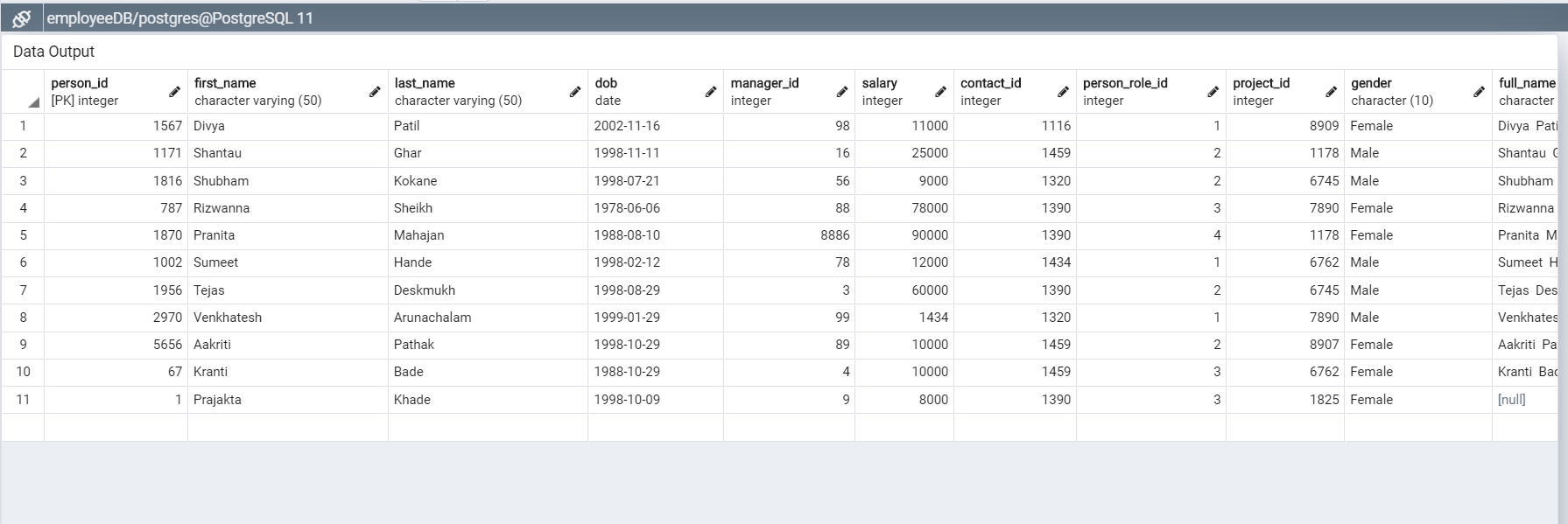


1. ROLLBACK the previous transaction





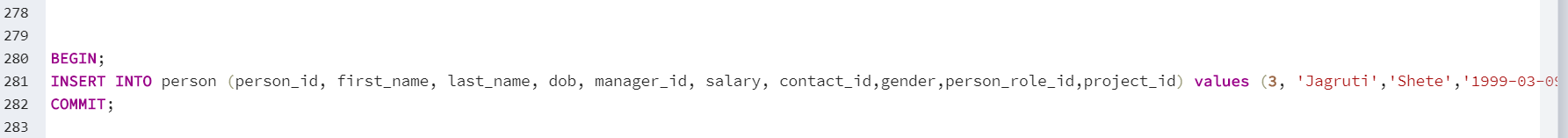
No change after ROLLBACK



The ROLLBACK command is the transactional command used to undo transactions that have not already been saved to the database.

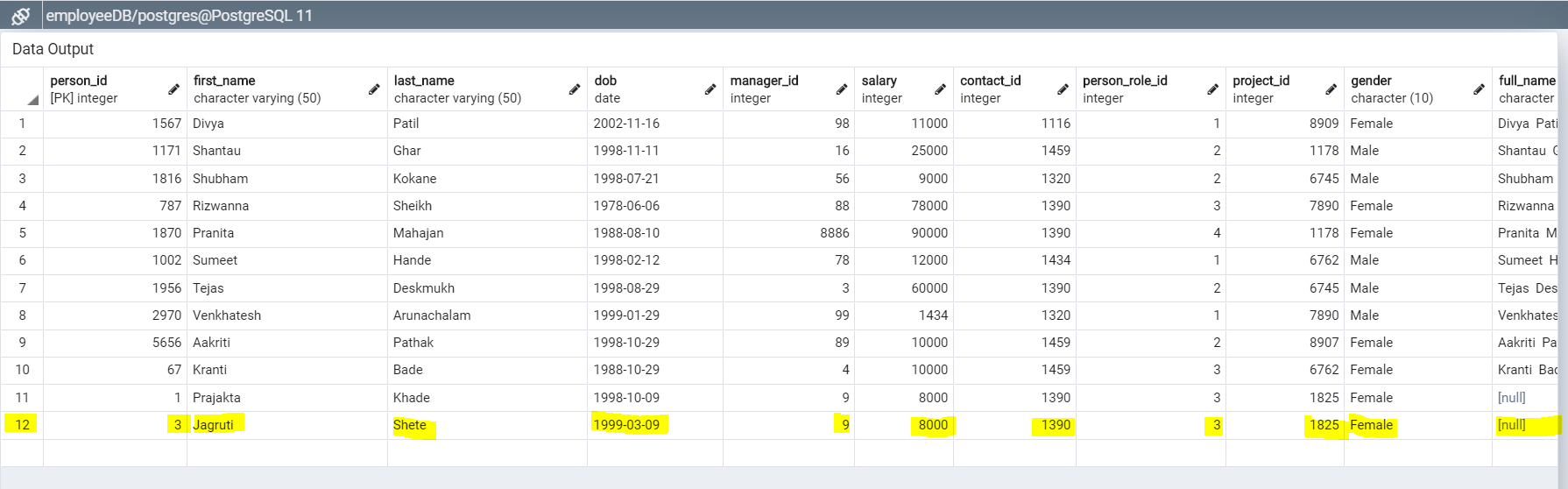
The ROLLBACK command can only be used to undo transactions since the last COMMIT or ROLLBACK command was issued.

1. COMMIT the transaction





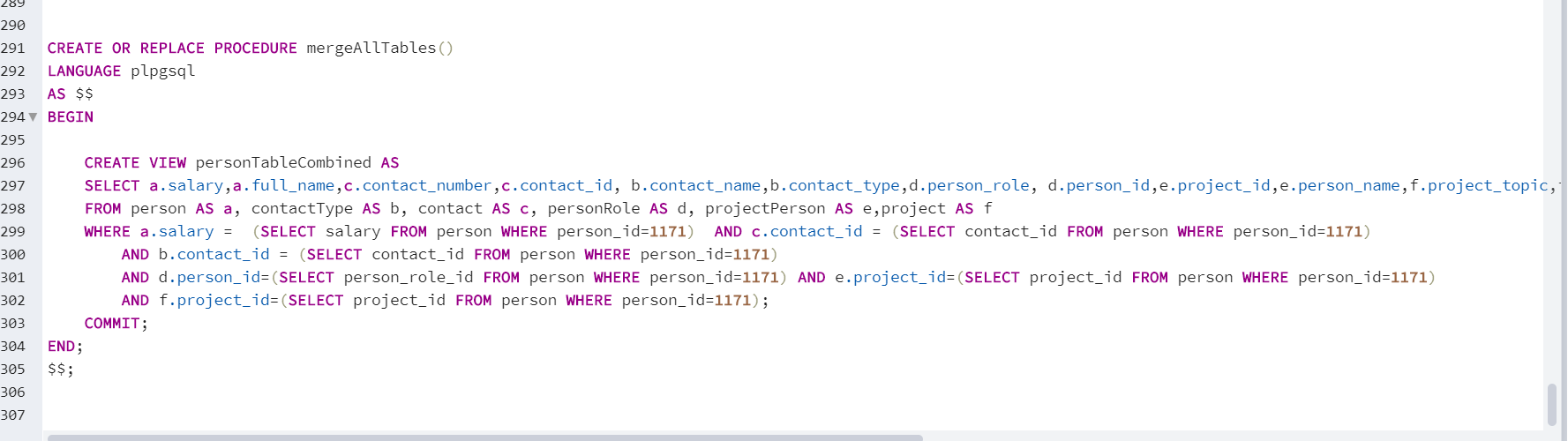
Commit adds the record

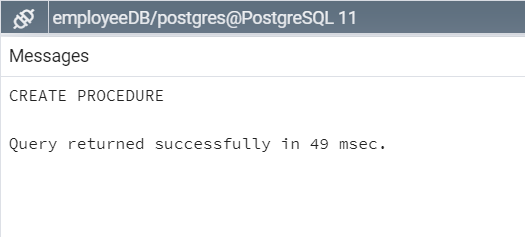


The COMMIT command is the transactional command used to save changes invoked by a transaction to the database.

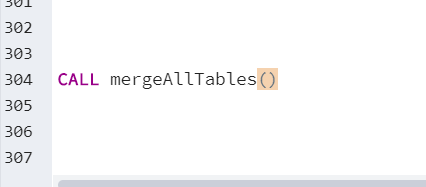
The COMMIT command saves all transactions to the database since the last COMMIT or ROLLBACK command.

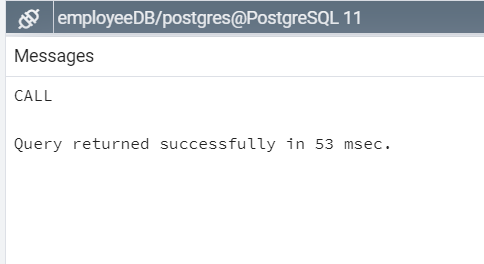
1. Create PROCEDURE





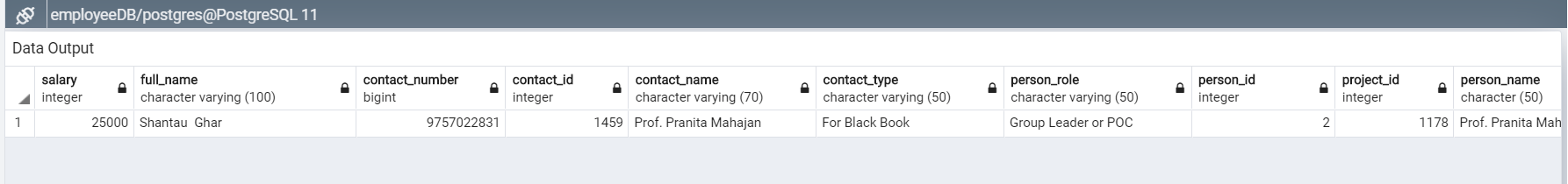
1. Call PROCEDURE





1. Result





1. DELETE is a Data Manipulation Language command, DML command and is used to remove tuples/records from a relation/table. Whereas DROP is a Data Definition Language, DDL command and is used to remove named elements of schema like relations/table, constraints or entire schema.