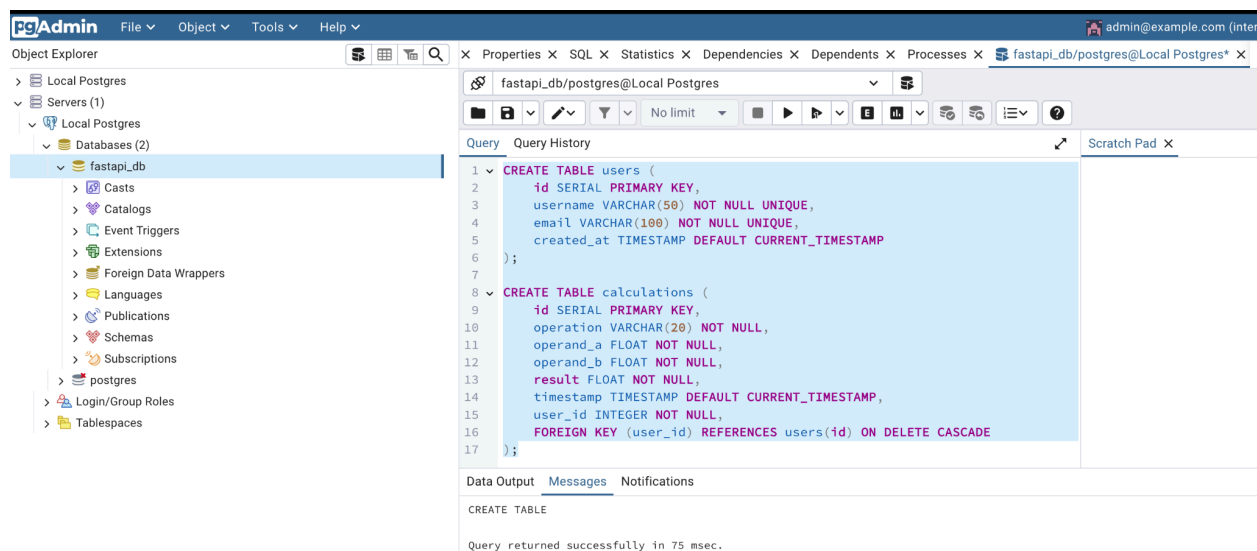


## Table Creation – users & calculations

- The first screenshot shows creation of two related tables.
- users stores unique usernames and emails with a timestamp for account creation.
- calculations records arithmetic operations performed by users, including operands, results, and timestamps.
- A foreign key (user\_id) links each calculation to a corresponding user, demonstrating a one-to-many relationship.
- Result: “Query returned successfully” confirms both tables were created without errors.

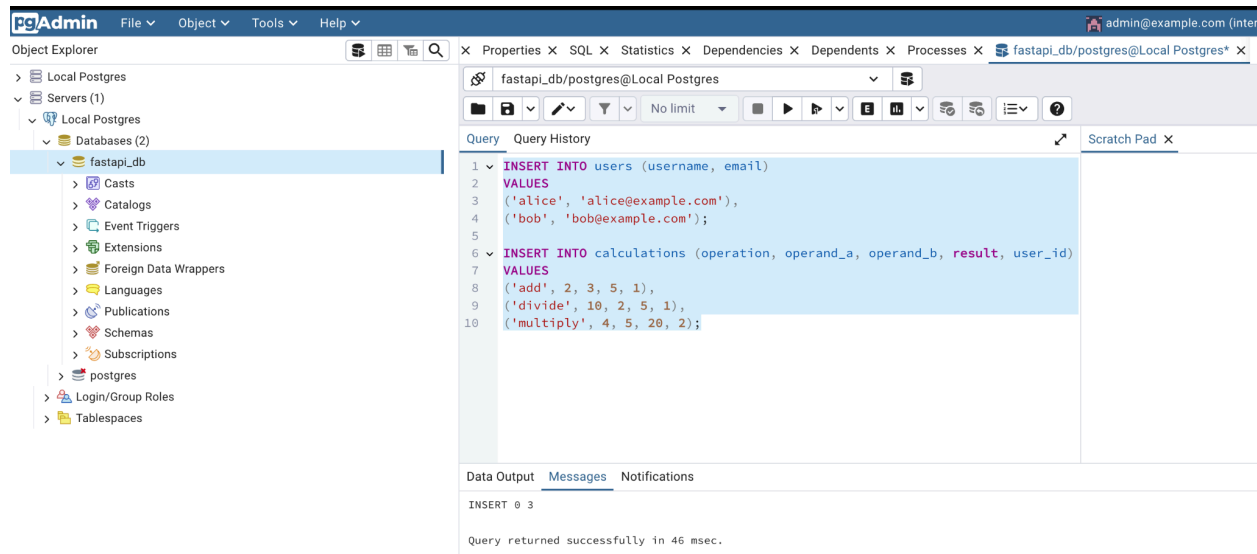
A)



## Data Insertion

- Two users (alice, bob) and three calculations were inserted into the database.
- This verifies correct use of INSERT INTO for both parent and child tables.
- The foreign key relationship ensures that each calculation references a valid user ID.
- ☒ Result: “INSERT 0 3” confirms that three rows were successfully added to calculations.

B)



## Data Retrieval (SELECT Queries & JOIN)

- Individual queries retrieved data from users and calculations tables.
- The JOIN query combines both tables using user\_id, clearly showing which user performed each operation.
- The output displays rows for alice and bob with their respective operations (add, divide, multiply).
- ☒ Result: Confirms relational integrity and demonstrates successful data linking between tables.

C)

The screenshot shows the PgAdmin interface with the 'fastapi\_db/postgres@Local Postgres' connection selected. The 'Query' tab is active, displaying the following SQL code:

```

1  -- Retrieve all users
2  SELECT * FROM users;
3
4  -- Retrieve all calculations
5  SELECT * FROM calculations;
6
7  -- Join users and calculations
8  SELECT u.username, c.operation, c.operand_a, c.operand_b, c.result
9  FROM calculations c
10 JOIN users u ON c.user_id = u.id;

```

The 'Data Output' tab shows the results of the query as a table with 5 columns: username, operation, operand\_a, operand\_b, and result. The table contains 3 rows of data.

	username character varying (50)	operation character varying (20)	operand_a double precision	operand_b double precision	result double precision
1	alice	add	2	3	5
2	alice	divide	10	2	5
3	bob	multiply	4	5	20

## Updating Records

- The UPDATE statement modifies an existing record in calculations, setting result = 6 for the record with id = 1.
- This demonstrates how to modify specific values while preserving other columns.
- Result: "UPDATE 1" confirms one record was successfully updated.

D)

The screenshot shows the PgAdmin interface with the 'fastapi\_db/postgres@Local Postgres' connection selected. The 'Query' tab is active, displaying the following SQL code:

```

1  UPDATE calculations
2  SET result = 6
3  WHERE id = 1; -- or whichever row you want to update

```

The 'Messages' tab is active, showing the output of the query:

```

UPDATE 1

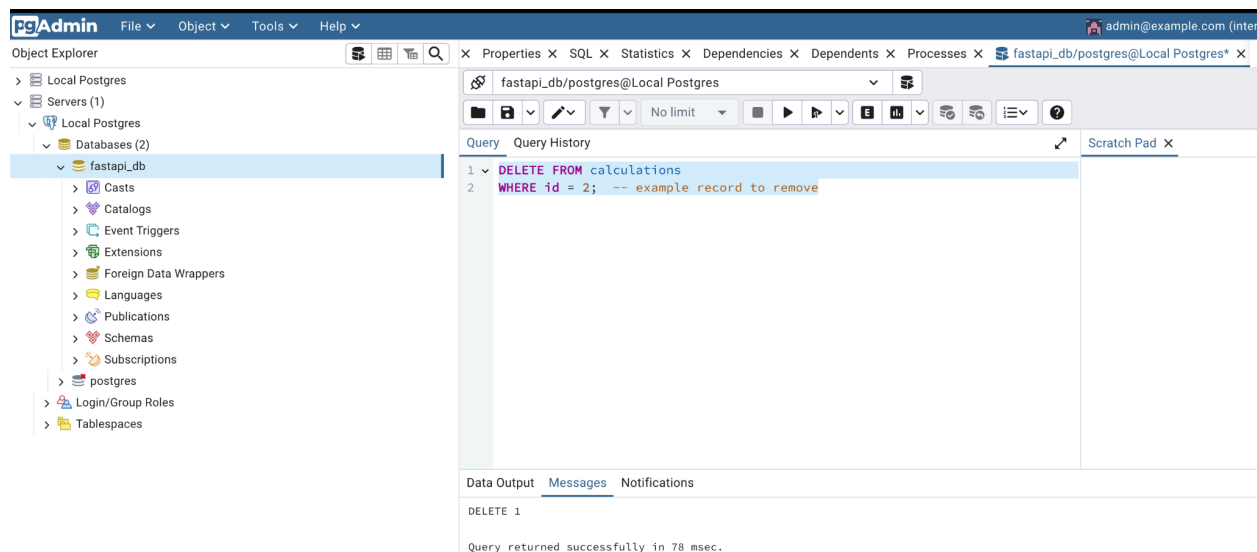
Query returned successfully in 58 msec.

```

## Deleting Records

- The DELETE statement removes the calculation with id = 2.
- This verifies cascade and referential behavior—deleting a calculation doesn't affect the parent user record.
- *Result:* “DELETE 1” confirms the record was successfully removed.

## E)



## Overall Outcome

- All Docker containers (FastAPI, PostgreSQL, pgAdmin) are running correctly.
- The database operations confirm understanding of CRUD principles and relational modeling (one-to-many relationships using foreign keys).
- Queries executed successfully with expected results, fulfilling the assignment requirements.

