

Homework: Managing SQL Data

Database Setup

Create a new database:

Name the database **restaurant_reviews**

```
marutiyson@marutiyson:~/workspaces/msdocs-flask-postgresql-sample-app-azd (main)$ psql -h localhost -U app_user -d app
psql (17.6 (Debian 17.6-0+deb13u1), server 18.0 (Debian 18.0-1.pgdg13+3))
WARNING: psql major version 17, server major version 18.
Some psql features might not work.
Type "help" for help.
```

```
app=# CREATE DATABASE restaurant_reviews;
CREATE DATABASE
app=# \l
```

List of databases									
Name	Owner	Encoding	Locale	Provider	Collate	Ctype	Locale	ICU Rules	Access privileges
app	app_user	UTF8	libc		en_US.utf8	en_US.utf8			
postgres	app_user	UTF8	libc		en_US.utf8	en_US.utf8			
restaurant_reviews	app_user	UTF8	libc		en_US.utf8	en_US.utf8			=c/app_user + app_user=CTc/app_user
template0	app_user	UTF8	libc		en_US.utf8	en_US.utf8			=c/app_user + app_user=CTc/app_user
template1	app_user	UTF8	libc		en_US.utf8	en_US.utf8			=c/app_user + app_user=CTc/app_user
(5 rows)									

Connect to the database

The screenshot shows the SQLTools interface with the 'Connection Assistant' step 2/3. On the left, the 'CONNECTIONS' tree view shows a 'Container database app_user@localhost' expanded, with 'restaurant_reviews' selected. The main panel displays the 'Connection Settings' for connecting to 'restaurant_reviews'. The fields are as follows:

- Connection name***: Container database
- Connection group**: (empty)
- Connect using***: Server and Port
- Server Address***: localhost
- Port***: 5432
- Database***: restaurant_reviews
- Username***: app_user
- Use password**: Save as plaintext in settings
- Password***: (redacted)

Below these fields is a section titled 'node-pg driver specific options' containing 'SSL' (Disabled) and 'statement_timeout' (empty).

Create two tables:

restaurant table: **id, name, street_address, description.**

The screenshot shows the SSMS interface with the following details:

- Left pane (Object Explorer):** Shows the connection to "Container database app_user@localhost" and the "public" schema, which contains a "Tables" folder containing "restaurant".
- Middle pane (SQL Editor):** Displays the SQL code for creating the "restaurant" table:

```
CREATE TABLE restaurant (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    street_address VARCHAR(200) NOT NULL,
    description TEXT
);
```
- Bottom pane (SQL Console):** Shows the message "CREATE successfully executed. 11:40:36 AM".
- Bottom bar:** Includes buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN.

review table: **id, restaurant_id, user_name, rating, review_text, review_date.**

The screenshot shows the SSMS interface with the following details:

- Left pane (Object Explorer):** Shows the connection to "Container database app_user@localhost" and the "public" schema, which contains "Tables" folders for "restaurant" and "review".
- Middle pane (SQL Editor):** Displays the SQL code for creating the "review" table:

```
CREATE TABLE review (
    id SERIAL PRIMARY KEY,
    restaurant_id INTEGER NOT NULL,
    user_name VARCHAR(50) NOT NULL,
    rating INTEGER NOT NULL CHECK (rating >= 1 AND rating <= 5),
    review_text TEXT,
    review_date DATE NOT NULL,
    CONSTRAINT fk_review_restaurant
        FOREIGN KEY (restaurant_id)
            REFERENCES restaurant(id)
            ON DELETE CASCADE
);
```
- Bottom pane (SQL Console):** Shows the message "CREATE successfully executed. 11:49:40 AM".
- Bottom bar:** Includes buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN.

Inserting Data

Insert at least 3 restaurants in the restaurant table

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays a connection to 'Container database app_user@localhost:5432/r...'. Under the 'Tables' node, there is a 'restaurant' table. The 'Scripting' tab is selected, showing an 'INSERT INTO' statement:

```
1 INSERT INTO restaurant (name, street_address, descr
2 ('Teras Rumah Nenek', 'Jalan Abdulrahman No.46, Cibubur
3 ('Namaaz Dining', 'Jalan Gunawarman No.42, Selong, Kecamatan
4 ('Plataran Menteng', 'Jalan HOS. Cokroaminoto No.42
```

The status bar at the bottom indicates: 'INSERT successfully executed. 3 rows were affected. 11:58:54 AM'.

Check the data successfully inserted

Select * from restaurant;

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays a connection to 'Container database app_user@localhost:5432/r...'. Under the 'Tables' node, there is a 'restaurant' table. The 'Scripting' tab is selected, showing a 'select * from restaurant;' query:

```
1 -- INSERT INTO restaurant (name, street_address, descr
2 -- ('Teras Rumah Nenek', 'Jalan Abdulrahman No.46, Cibubur
3 -- ('Namaaz Dining', 'Jalan Gunawarman No.42, Selong, Kecamatan
4 -- ('Plataran Menteng', 'Jalan HOS. Cokroaminoto No.42, Gondangdia...
```

The results pane on the right shows the data inserted earlier:

ID	NAME	STREET_ADDRESS
4	Teras Rumah Nenek	Jalan Abdulrahman No.46, Cibubur, Kecamatan
5	Namaaz Dining	Jalan Gunawarman No.42, Selong, Kecamatan
6	Plataran Menteng	Jalan HOS. Cokroaminoto No.42, Gondangdia...

The status bar at the bottom indicates: 'SELECT successfully executed. 11:59:57 AM'.

Insert at least 5 reviews in the review table, ensuring they reference the correct restaurants via **restaurant_id**.

```
INSERT INTO review (restaurant_id, user_name, rating, review_text, review_date)
VALUES
(4, 'Johanes', 5, 'Amazing pasta! The carbonara was perfection. Highly recommend!', '2025-10-15'),
(4, 'Sarah', 4, 'Great atmosphere and delicious food. A bit pricey but worth it.', '2025-10-18'),
(5, 'Budi', 5, 'Best sushi in town! Fresh fish and excellent presentation.', '2025-10-10'),
(5, 'Ria', 3, 'Good sushi but service was slow. Decent experience overall.', '2025-10-20'),
(6, 'Alex', 4, 'Juicy burgers with creative toppings. The fries were crispy and perfect!', '2025-10-12');
```

```
SQLTOOLS
  CONNECTIONS
    Container database app_user@localhost:5432/r...
      restaurant_reviews database
        Schemas
          public
            Tables
              restaurant
              review
              Views
              Materialized Views
              Functions

  BOOKMARKS
  QUERY HISTORY
    Container database
      INSERT INTO review (restaurant_id, user_name, rat...
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 5 AZURE SQL CO

o @marulityson → /workspaces/msdocs-flask-postgresql-sample-app-azd (main) \$

Check the data successfully inserted

Select * from review;

```
SQLTOOLS
  CONNECTIONS
    Container database app_user@localhost:5432/r...
      restaurant_reviews database
        Schemas
          public
            Tables
              restaurant
              review
              Views
              Materialized Views
              Functions

  BOOKMARKS
  QUERY HISTORY
    Container database
      select * from review 10/21/2025, 12:54:47 PM
      INSERT INTO review (restaurant_id, user_name, rat...
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 5 AZURE SQL CONSOLE

SELECT successfully executed. 12:54:47 PM

Performing CRUD Operations

Create (Insert):

1. Insert a new restaurant into the restaurant table

```
INSERT INTO restaurant (name, street_address, description)
VALUES ('Kintan Buffet', 'Gandaria City, Jalan Arteri Pondok Indah, Kebayoran Lama Utara,
Kecamatan Kebayoran Lama, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta', 'Menu
rekomendasinya ada banyak. Salah satu menu favoritnya adalah karubi garlic yang punya
tekstur lembut saat dikunyah. Soal rasa gak perlu diragukan karena dijamin enak dan sedap
di lidah.');
```

The screenshot shows a SQL tool interface with a dark theme. On the left, there's a sidebar with 'SQLTOOLS' at the top, followed by 'CONNECTIONS' which lists 'Container database app_user@localhost:5432/r...', 'restaurant_reviews database', and 'Schemas'. Under 'Schemas', there's a 'public' folder containing 'Tables', 'Views', 'Materialized Views', and 'Functions'. The 'Tables' folder is expanded, showing 'restaurant' and 'review'. Below the sidebar are sections for 'BOOKMARKS', 'QUERY HISTORY', and 'Container database'. The main area has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS 5', 'AZURE', and 'SQL CONSOLE'. The 'SQL CONSOLE' tab is active, displaying the SQL query from the code block above. Below the query, it says 'INSERT successfully executed. 1 rows were affected. 12:59:29 PM'. At the bottom right, there are buttons for 'CONSOLE', 'RE-RUN QUERY', 'EXPORT', 'OPEN', and '0 of 0'.

Check the data successfully inserted

Select * from restaurant;

The screenshot shows the same SQL tool interface as the previous one. The 'SQL CONSOLE' tab now contains the query 'select * from restaurant'. To the right, a results grid titled 'Container database: -- INSERT INTO r...' shows a table with three columns: 'id', 'name', and 'street_address'. The table contains seven rows of data, with row 7 being highlighted. The data is as follows:

id	name	street_address
1	Teras Rumah Nenek	Jalan Abdulrahman No.46, Cibubur
2	Namaaz Dining	Jalan Gunawarman No.42, Selong
3	Plataran Menteng	Jalan HOS. Cokroaminoto No.42, Menteng
4	Kintan Buffet	Gandaria City, Jalan Arteri Pondok Indah, Kebayoran Lama Utara, Kecamatan Kebayoran Lama, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta
5		
6		
7		

2. Insert a new review for an existing restaurant

```
INSERT INTO review (restaurant_id, user_name, rating, review_text, review_date)
VALUES (7, 'Lisa', 5, 'The best burger I have ever had! Will definitely come back!', '2025-10-21');
```

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure under 'CONTNECTIONS'. In the center, a query window titled 'Untitled-6' contains the following SQL code:

```
1 -- Run on active connection | Select block
2 INSERT INTO review (restaurant_id, user_name, rating, review_text, review_date)
3 VALUES (7, 'Lisa', 5, 'The best burger I have ever had! Will definitely come back!', '2025-10-21');
```

Below the code, the status bar indicates: 'Container database: INSERT INTO rev...'. At the bottom of the window are buttons for 'CONSOLE', 'RE-RUN QUERY', 'EXPORT', and 'OPEN'. To the right, the 'SQL CONSOLE' tab is selected, showing the message: 'INSERT successfully executed. 1 rows were affected. 1:04:49 PM'. The 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', 'AZURE', and 'SQL CONSOLE' tabs are visible at the top of the right panel.

Check the data successfully inserted

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure under 'CONTNECTIONS'. In the center, a query window titled 'Untitled-6' contains the following SQL code:

```
1 -- Run on active connection | Select block
2 -- INSERT INTO review (restaurant_id, user_name, rating, review_text, review_date)
3 -- (7, 'Lisa', 5, 'The best burger I have ever had!', '2025-10-21')
4
5 select * FROM review;
```

Below the code, the status bar indicates: 'Container database: -- INSERT INTO rev...'. At the bottom of the window are buttons for 'CONSOLE', 'RE-RUN QUERY', 'EXPORT', and 'OPEN'. To the right, the 'SQL CONSOLE' tab is selected, showing the message: 'SELECT successfully executed. 1:07:54 PM'. The 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', 'AZURE', and 'SQL CONSOLE' tabs are visible at the top of the right panel. A preview pane on the right shows the results of the SELECT query:

	id	restaurant_id	user_name	rating
6	6	3	Johanes	5
7	7	4	Sarah	4
8	8	5	Budi	5
9	9	5	Ria	3
10	10	6	Alex	4
11	11	7	Lisa	5

Read (Select):

1. Retrieve all reviews for a specific restaurant using the **restaurant_id**

```
select * FROM review  
where restaurant_id = 4;
```

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays a connection to a 'Container database' named 'app_user@localhost:5432/rest...'. Under the 'Tables' node, there is a 'review' table. The central pane contains a query editor with the following code:

```
-- INSERT INTO review (restaurant_id, user_name, rating)  
-- (7, 'Lisa', 5, 'The best burger I have ever had!')  
  
select * FROM review  
where restaurant_id = 4;
```

The right pane shows the results of the query, which returns three rows of data:

id	restaurant_id	user_name	rating
6	4	Johanes	5
7	4	Sarah	4

At the bottom of the interface, there are several buttons: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, AZURE, and SQL CONSOLE. The SQL CONSOLE tab is selected. A status message at the bottom indicates 'SELECT successfully executed. 1:24:21 PM'.

2. Retrieve all reviews with a rating of 4 or higher

```
SELECT * FROM review  
WHERE rating >= 4;
```

The screenshot shows the SQL Server Management Studio interface. On the left, the Object Explorer displays a connection to a 'Container database' named 'app_user@localhost:5432/rest...'. Under the 'Tables' node, there is a 'review' table. The central pane contains a query editor with the following code:

```
-- INSERT INTO review (restaurant_id, user_name, rating)  
-- (7, 'Lisa', 5, 'The best burger I have ever had!')  
  
select * FROM review  
where restaurant_id = 4;  
  
SELECT * FROM review  
WHERE rating >= 4;
```

The right pane shows the results of the query, which returns five rows of data:

id	restau...	user_n...	rating
6	4	Johanes	5
7	4	Sarah	4
8	5	Budi	5
10	6	Alex	4
11	7	Lisa	5

At the bottom of the interface, there are several buttons: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, AZURE, and SQL CONSOLE. The SQL CONSOLE tab is selected. A status message at the bottom indicates 'SELECT successfully executed. 1:27:13 PM'.

3. Use a JOIN to display a list of restaurants along with their reviews

```
SELECT
    r.id AS restaurant_id,
    r.name AS restaurant_name,
    r.street_address,
    rev.id AS review_id,
    rev.user_name,
    rev.rating,
    rev.review_text,
    rev.review_date
FROM restaurant r
LEFT JOIN review rev ON r.id = rev.restaurant_id
ORDER BY r.id, rev.review_date DESC;
```

The screenshot shows the SQLTools interface with the following details:

- Left Panel (Connections):** Shows a tree view of database connections, schemas, tables, and views.
- Top Center (Query Editor):** Displays the SQL query with line numbers 1 through 12.
- Top Right (Results):** Shows the results of the query in a table format. The table has columns: res... (row number), restaurant... (name), street_address (address), review... (rating), user... (user name), rating (rating), and review_date (review date). The data includes:

res...	restaurant...	street_address	review...	user...	rating	review_date
1	Teras Rumah N...	Jalan Abdulrahman N...	7	Sarah	4	Great
2	Teras Rumah N...	Jalan Abdulrahman N...	6	Johanes	5	Amazing
3	Namaaz Dining	Jalan Gunawarman N...	9	Ria	3	Good
4	Namaaz Dining	Jalan Gunawarman N...	8	Budi	5	Best
5	Plataran Menteng	Jalan HOS. Cokroamini	10	Alex	4	Juicy
6	Kintan Buffet	Gandaria City, Jalan A...	11	Lisa	5	The best

- Bottom (Status Bar):** Shows "SELECT successfully executed. 1:35:55 PM".

Update:

1. Update the description of one restaurant.

Before

```
select description  
from restaurant  
where id = 4;
```

The screenshot shows the SQLTools interface. On the left, the 'CONNECTIONS' sidebar lists a connection to 'Container database app_user@localhost:5432/restaurant_reviews'. The 'Tables' section under 'Schemas' contains 'public', 'restaurant', and 'review' tables, and 'Views' which is currently selected. The main area shows a query editor with the following code:

```
1 2 3 4 5 6 7  
select description  
from restaurant  
where id = 4;
```

Below the editor, the status bar shows 'SELECT successfully executed. 1:40:54 PM'. The bottom navigation bar includes 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', 'AZURE', and 'SQL CONSOLE'.

Updating data

```
UPDATE restaurant  
SET description = 'Award-winning Italian restaurant featuring traditional recipes  
passed down through generations'  
WHERE id = 4;
```

The screenshot shows the SQLTools interface after running the update command. The 'CONNECTIONS' sidebar remains the same. The main area shows the query editor with the following code:

```
1 2 3 4 5 6 7 8 9 10 11  
-- select description  
-- from restaurant  
-- where id = 4;  
UPDATE restaurant  
SET description = 'Award-winning I...  
WHERE id = 4;
```

The status bar now shows 'UPDATE successfully executed. 1 rows were affected. 1:44:30 PM'. The bottom navigation bar includes 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', 'AZURE', and 'SQL CONSOLE'.

After

```
select description  
from restaurant  
where id = 4;
```

The screenshot shows the SQLTools interface after executing the select query. The 'CONNECTIONS' sidebar remains the same. The main area shows the query editor with the following code:

```
1 2 3 4 5 6 7 8 9  
--- UPDATE restaurant  
--- SET description = 'Award-winning I...  
--- WHERE id = 4;  
select description  
from restaurant  
where id = 4;
```

The status bar shows 'SELECT successfully executed. 1:45:26 PM'. The bottom navigation bar includes 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', 'PORTS', 'AZURE', and 'SQL CONSOLE'.

2. Update the rating of a specific review.

Before

```
select id, user_name, rating, review_text
from review
where id = 11;
```

SOLTOOLS

CONNECTIONS

- Container database app_user@localhost:5432/rest...
- restaurant_reviews database
- Schemas
- public
- Tables
- Views
- Materialized Views
- Functions

BOOKMARKS

QUERY HISTORY

Container database

```
select id, user_name, rating, review_text from review where id = 11;
```

Container database: select * from review

id	user_name	rating	review_text
11	Lisa	5	The best burger I have ever

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE SQL CONSOLE

SELECT successfully executed. 1:55:55 PM

Updating data

UPDATE review

SET rating = 3

WHERE id = 11;

SOLTOOLS

CONNECTIONS

- Container database app_user@localhost:5432/rest...
- restaurant_reviews database
- Schemas
- public
- Tables
- Views
- Materialized Views
- Functions

BOOKMARKS

QUERY HISTORY

Container database

```
-- select id, user_name, rating, review_text from review where id = 11;
UPDATE review
SET rating = 3
WHERE id = 11;
```

Container database: select id, user_name, rating, review_text from review

id	user_name	rating	review_text
11	Lisa	3	The best burger I have ever

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE SQL CONSOLE

UPDATE successfully executed. 1 rows were affected. 1:57:46 PM

After

```
select id, user_name, rating, review_text
from review
where id = 11;
```

SOLTOOLS

CONNECTIONS

- Container database app_user@localhost:5432/rest...
- restaurant_reviews database
- Schemas
- public
- Tables
- Views
- Materialized Views
- Functions

BOOKMARKS

QUERY HISTORY

Container database

```
select id, user_name, rating, review_text from review where id = 11;
-- UPDATE review
-- SET rating = 3
-- WHERE id = 11;
```

Container database: -- select id, user_name, rating, review_text from review

id	user_name	rating	review_text
11	Lisa	3	The best burger I have ever

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE SQL CONSOLE

SELECT successfully executed. 1:58:20 PM

Delete:

1. Delete one review based on id

Before

```
select id, user_name, rating, review_text  
from review  
where id = 11;
```

id	user_name	rating	review_text
11	Lisa	5	The best burger I have ever

Delete

DELETE FROM review

```
WHERE id = 11;
```

```
-- select id, user_name, rating, review_text  
-- from review  
-- where id = 11  
  
DELETE FROM review  
WHERE id = 11;
```

After

```
select id, user_name, rating, review_text  
from review  
where id = 11;
```

id	user_name	rating	review_text
			No data

2. Delete a restaurant and ensure its associated reviews are also deleted (using cascade)

Insert new temporary data in table restaurant and review

```

SQLTOOLS          ... Initiated-10 •  INSERT INTO restaurant (name, street_address) Untitled-11 •  ...
CONNECTIONS
  Container database app_user@localhost:5432/rest...
  restaurant_reviews database
    Schemas
      public
        Tables
          restaurant
          review
          Views
          Materialized Views
          Functions

  BOOKMARKS
  QUERY HISTORY
    Container database
      INSERT INTO review (restaurant_id, user_name, rating, ...
      INSERT INTO restaurant (name, street_address, de...

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 5 AZURE SQL CONSOLE
INSERT successfully executed. 1 rows were affected. 2:10:23 PM
  
```

Data inserted successfully

```

SQLTOOLS          ... Initiated-10 •  -- INSERT INTO restaurant (name, street_address) Untitled-11 •  ...
CONNECTIONS
  Container database app_user@localhost:5432/rest...
  restaurant_reviews database
    Schemas
      public
        Tables
          restaurant
          review
          Views
          Materialized Views
          Functions

  BOOKMARKS
  QUERY HISTORY
    Container database
      select * from restaurant LEFT JOIN review ...
      SELECT successfully executed. 2:12:56 PM

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 5 AZURE SQL CONSOLE
  
```

Deleting data

```

SQLTOOLS          ... Initiated-10 •  -- INSERT INTO restaurant (name, street_address) Untitled-11 •  ...
CONNECTIONS
  Container database app_user@localhost:5432/rest...
  restaurant_reviews database
    Schemas
      public
        Tables
          restaurant
          review
          Views
          Materialized Views
          Functions

  BOOKMARKS
  QUERY HISTORY
    Container database
      DELETE FROM restaurant WHERE name = 'Temporary Restaurant';
      DELETE successfully executed. 1 rows were affected. 2:14:03 PM

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1 AZURE SQL CONSOLE
  
```

The screenshot shows the SSMS interface with the following components:

- Left Sidebar:** Contains sections for **CONNECTIONS**, **BOOKMARKS**, and **QUERY HISTORY**. Under **CONNECTIONS**, it lists "Container database app_user@localhost:5432/rest..." and "restaurant_reviews database". Under **BOOKMARKS** and **QUERY HISTORY**, there is a single entry: "select * from restaurant LEFT JOIN review ON resta...".
- Top Bar:** Includes tabs for **PROBLEMS**, **OUTPUT**, **DEBUG CONSOLE**, **TERMINAL**, **PORTS**, **AZURE**, and **SQL CONSOLE**. The **SQL CONSOLE** tab is selected.
- Query Editor:** Displays a numbered script:

```
1 -- INSERT INTO restaurant (name, street_
2 -- ('Temporary Restaurant', '999 Test Street', 'This w
3
4
5 -- INSERT INTO review (restaurant_id, user_name, rating
6 -- ((SELECT id FROM restaurant WHERE name = 'Temporary
7
8
9 select * from restaurant LEFT JOIN review
10 ON restaurant.id = review.restaurant_id
11 WHERE restaurant.name = 'Temporary Restaurant';
12
13 -- DELETE FROM restaurant
14 -- WHERE name = 'Temporary Restaurant';
```
- Results Pane:** Shows a table titled "Container database: -- INSERT INTO r...". It has columns **id**, **name**, and **street_address**. A message "No data" is displayed.
- Bottom Buttons:** Includes **CONSOLE**, **RE-RUN QUERY**, **EXPORT**, and **OPEN**.

Additional Queries

Find the highest-rated restaurant based on the average rating of all its reviews.

```
SELECT
    r.id,
    r.name,
    r.street_address,
    AVG(rev.rating) AS average_rating,
    COUNT(rev.id) AS total_reviews
FROM restaurant r
INNER JOIN review rev ON r.id = rev.restaurant_id
GROUP BY r.id, r.name, r.street_address
ORDER BY average_rating DESC
LIMIT 1;
```

The screenshot shows a SQL development environment with the following components:

- Left Sidebar (SQLTOOLS):** Contains sections for CONNECTIONS (Container database, Schemas like public), TABLES (restaurant, review, Views, Materialized Views, Functions), and BOOKMARKS/QUERY HISTORY.
- Central Area:** A code editor window titled "Untitled-11" containing the SQL query. The code is numbered 1 to 11. Lines 1-6 define the SELECT statement with columns r.id, r.name, r.street_address, average_rating, and total_reviews. Lines 7-11 complete the query with FROM, INNER JOIN, GROUP BY, ORDER BY, and LIMIT clauses.
- Right Area:** A results grid titled "Container database: SELECT r.id...". It shows one row for "Teras Rumah..." with id 4, name "Teras Rumah...", street_address "Jalan Abd...", average_rating 4.50000000000000..., and total_reviews 2.
- Bottom Navigation:** Buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN. Status bar text includes "SELECT successfully executed. 2:18:45 PM".

Find the number of reviews each restaurant has received.

```
SELECT
    r.id,
    r.name,
    COUNT(rev.id) AS number_of_reviews
FROM restaurant r
LEFT JOIN review rev ON r.id = rev.restaurant_id
GROUP BY r.id, r.name
ORDER BY number_of_reviews DESC;
```

The screenshot shows the SQLTools interface with the following details:

- Left Panel (Connections):** Shows a connection to a Container database named app_user@localhost:5432/rest... and a database named restaurant_reviews.
- Center Panel (Query Editor):** Displays the SQL query to find the number of reviews per restaurant.
- Right Panel (Results):** Shows a table with the results:

id	name	number_of_reviews
4	Teras Rumah Nenek	2
5	Namaaz Dining	2
6	Plataran Menteng	1
7	Kintan Buffet	0

Below the results, there are buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN.

Display the most recent review for each restaurant.

```
SELECT DISTINCT ON (r.id)
    r.id AS restaurant_id,
    r.name AS restaurant_name,
    rev.id AS review_id,
    rev.review_date
FROM restaurant r
INNER JOIN review rev ON r.id = rev.restaurant_id
ORDER BY r.id, rev.review_date DESC;
```

The screenshot shows the SQLTools interface with the following details:

- Left Panel (Connections):** Shows a connection to a Container database named app_user@localhost:5432/rest... and a database named restaurant_reviews.
- Center Panel (Query Editor):** Displays the SQL query to find the most recent review for each restaurant.
- Right Panel (Results):** Shows a table with the results:

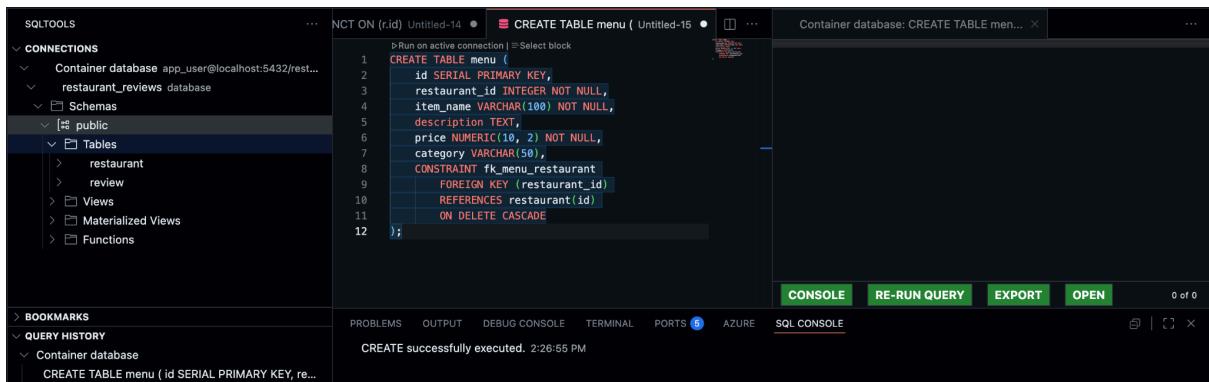
restaurant_id	restaurant_name	revie...	review_date
4	Teras Rumah Nenek	7	2025-10-18
5	Namaaz Dining	9	2025-10-20
6	Plataran Menteng	10	2025-10-12

Below the results, there are buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN.

Extra Credit (Optional)

Create a menu table

```
CREATE TABLE menu (
    id SERIAL PRIMARY KEY,
    restaurant_id INTEGER NOT NULL,
    item_name VARCHAR(100) NOT NULL,
    description TEXT,
    price NUMERIC(10, 2) NOT NULL,
    category VARCHAR(50),
    CONSTRAINT fk_menu_restaurant
        FOREIGN KEY (restaurant_id)
        REFERENCES restaurant(id)
        ON DELETE CASCADE
);
```



The screenshot shows the SQLTools interface with the following details:

- Left Sidebar (Connections):** Shows a connection to "Container database app_user@localhost:5432/rest...". Under "Schemas", the "public" schema is selected, showing tables like "restaurant", "review", "Views", "Materialized Views", and "Functions".
- Middle Panel (Query Editor):** A code editor window titled "CREATE TABLE menu (Untitled-15" contains the provided SQL code for creating the "menu" table.
- Bottom Panel (Console):** Shows the message "CREATE successfully executed. 2:26:55 PM".
- Bottom Navigation:** Buttons for "CONSOLE", "RE-RUN QUERY", "EXPORT", and "OPEN".

Insert at least 3 menu items for each restaurant.

```
INSERT INTO menu (restaurant_id, item_name, description, price, category) VALUES
(4, 'Spaghetti Carbonara', 'Classic Roman pasta with eggs, pecorino cheese, and guanciale', 16.99, 'Pasta'),
(4, 'Margherita Pizza', 'Traditional pizza with tomato sauce, mozzarella, and fresh basil', 14.99, 'Pizza'),
(4, 'Tiramisu', 'Classic Italian dessert with coffee-soaked ladyfingers and mascarpone', 8.99, 'Dessert'),

(5, 'California Roll', 'Crab, avocado, and cucumber wrapped in rice and seaweed', 12.99, 'Rolls'),
(5, 'Salmon Sashimi', 'Fresh sliced salmon served with wasabi and soy sauce', 18.99, 'Sashimi'),
(5, 'Miso Soup', 'Traditional Japanese soup with tofu, seaweed, and green onions', 4.99, 'Appetizer'),

(6, 'Classic Cheeseburger', 'Angus beef patty with cheddar, lettuce, tomato, and special sauce', 13.99, 'Burgers'),
(6, 'Sweet Potato Fries', 'Crispy sweet potato fries with garlic aioli', 6.99, 'Sides'),
(6, 'Chocolate Milkshake', 'Thick and creamy milkshake made with premium ice cream', 5.99, 'Beverages'),

(7, 'Chicken Tikka Masala', 'Tender chicken in creamy tomato-based curry sauce', 15.99, 'Main Course'),
(7, 'Garlic Naan', 'Fresh baked flatbread with garlic and butter', 3.99, 'Bread'),
(7, 'Vegetable Samosas', 'Crispy pastries filled with spiced potatoes and peas', 6.99, 'Appetizer');
```



The screenshot shows the SQLTools interface with the following details:

- SQLTOOLS** tab is selected.
- CONNECTIONS** section shows a connection to a Container database (app_user@localhost:5432).
- Schemas** section shows the public schema.
- Tables** section lists menu, restaurant, and review tables.
- BOOKMARKS** and **QUERY HISTORY** sections are present.
- CREATE TABLE menu** (Untitled-15) is shown in the center pane.
- INSERT INTO menu** (Untitled-16) is shown in the right pane.
- The right pane contains the executed SQL code, which is identical to the one provided in the text above, listing 19 menu items across 7 categories.

Write a query to display each restaurant with its menu and the average rating from its reviews.

```
SELECT
    r.id AS restaurant_id,
    r.name AS restaurant_name,
    m.item_name AS menu_item,
    m.description AS menu_description,
    m.price,
    ROUND(AVG(rev.rating), 2) AS average_rating,
    COUNT(DISTINCT rev.id) AS total_reviews
FROM restaurant r
LEFT JOIN menu m ON r.id = m.restaurant_id
LEFT JOIN review rev ON r.id = rev.restaurant_id
GROUP BY r.id, r.name, r.street_address,
r.description, m.id, m.item_name, m.description, m.price, m.category
ORDER BY r.id, m.category, m.item_name;
```

The screenshot shows the SQLTools interface with the following components:

- Left Panel (SQLTOOLS):** Shows the database structure with connections, schemas (public), tables (menu, restaurant, review, Views, Materialized Views, Functions), bookmarks, and query history.
- Middle Panel (Editor):** Displays the query code in a syntax-highlighted editor.
- Top Right Panel (Results):** Shows the results of the query execution, including columns: price, average_rating, and total_reviews. The results are as follows:

price	average_rating	total_reviews
8.99	4.50	2
16.99	4.50	2
14.99	4.50	2
4.99	4.00	2
12.99	4.00	2
18.99	4.00	2
5.99	4.00	1
13.99	4.00	1
6.99	4.00	1
6.99	NULL	0
3.99	NULL	0
15.99	NULL	0

- Bottom Right Panel (Buttons):** Includes buttons for CONSOLE, RE-RUN QUERY, EXPORT, and OPEN.