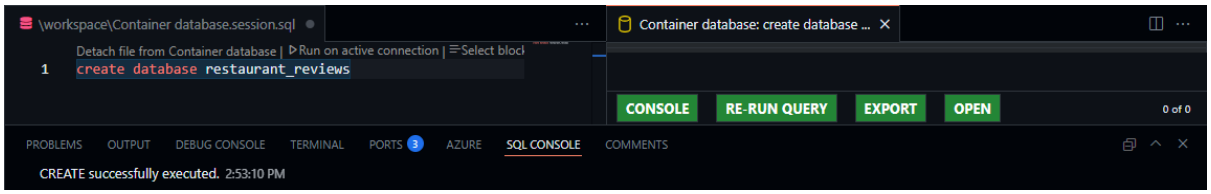
-- Database + Tables Creation

A. Database Setup

1. Create a new database:
   - Name the database `restaurant_reviews`

create database restaurant_reviews;

result:



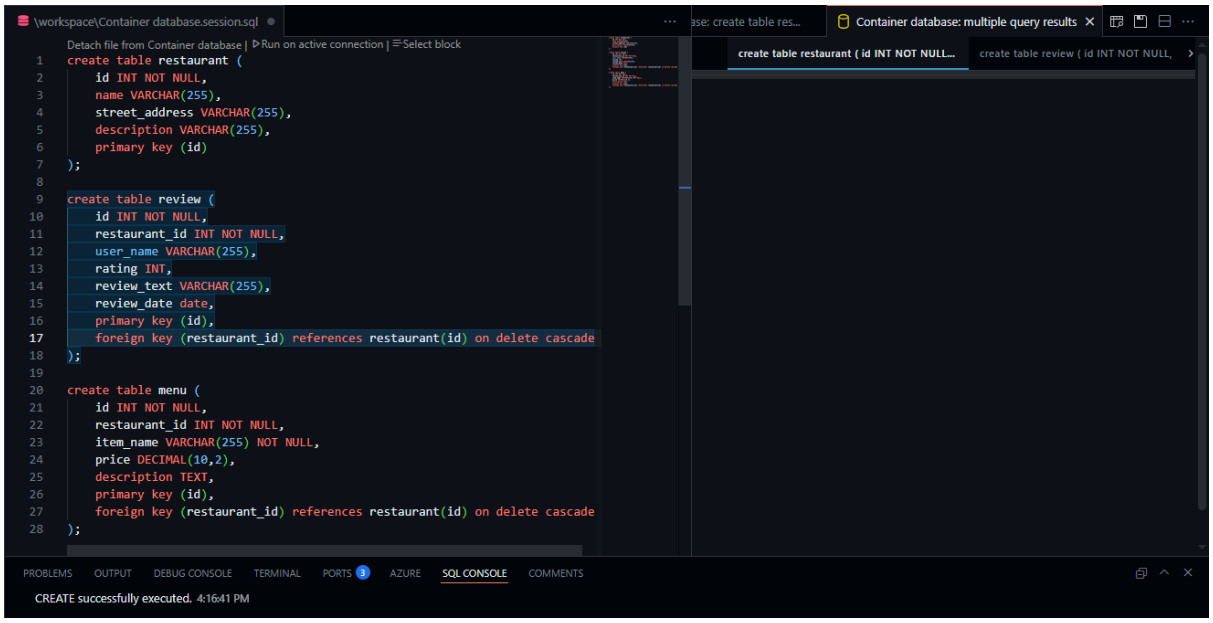
2. Create two tables:
   - restaurant table:
     - Columns: `id, name, street_address, description`.
   - review table:
     - Columns: `id, restaurant_id, user_name, rating, review_text, review_date`.
   - Ensure restaurant_id in the review table is a foreign key referencing the restaurant table.

```
create table restaurant (
   id INT NOT NULL,
   name VARCHAR(255),
   street_address VARCHAR(255),
   description VARCHAR(255),
   primary key (id)
);

create table review (
   id INT NOT NULL,
   restaurant_id INT NOT NULL,
   user_name VARCHAR(255),
   rating INT,
   review_text VARCHAR(255),
```

```
    review_date date,
    primary key (id),
    foreign key (restaurant_id) references restaurant(id) on delete cascade
);
```

Result:



-- Sample Data Insertion

B. Inserting Data

Insert sample data into both tables:

1. Insert **at least 3 restaurants** in the restaurant table.
2. Insert **at least 5 reviews** in the review table, ensuring they reference the correct restaurants via `restaurant_id`.

INSERT INTO restaurant

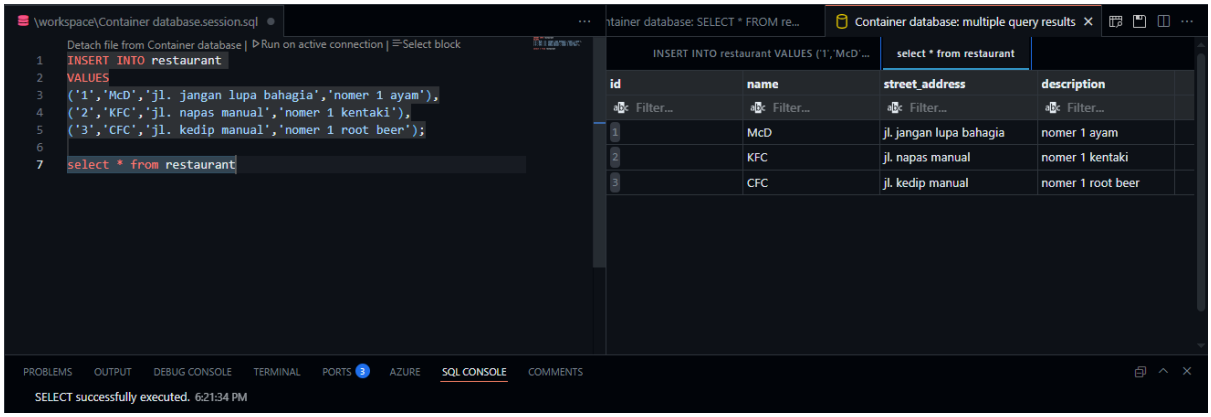
VALUES

('1','McD','jl. jangan lupa bahagia','nomer 1 ayam'),

('2','KFC','jl. napas manual','nomer 1 kentaki'),

('3','CFC','jl. kedip manual','nomer 1 root beer');

Result:



INSERT INTO review

VALUES

('1','1','ki joko pinter','4','pembohonk','01/01/2001'),

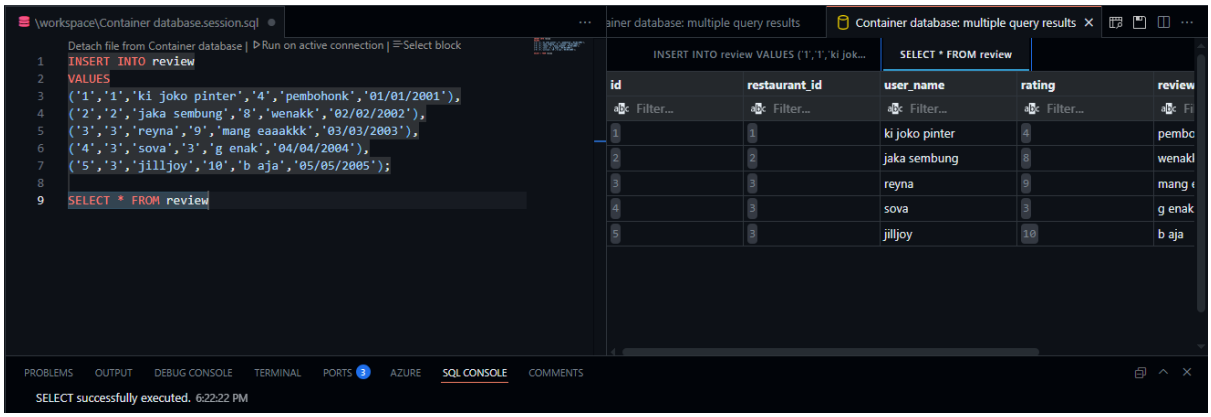
('2','2','jaka sembung','8','wenakk','02/02/2002'),

('3','3','reyna','9','mang eaaakkk','03/03/2003'),

('4','3','sova','3','g enak','04/04/2004'),

('5','3','jilljoy','10','b aja','05/05/2005');

Result:



-- Queries for CRUD

-- Create

-- Case1

C. Performing CRUD Operations

Perform the following operations on your database:

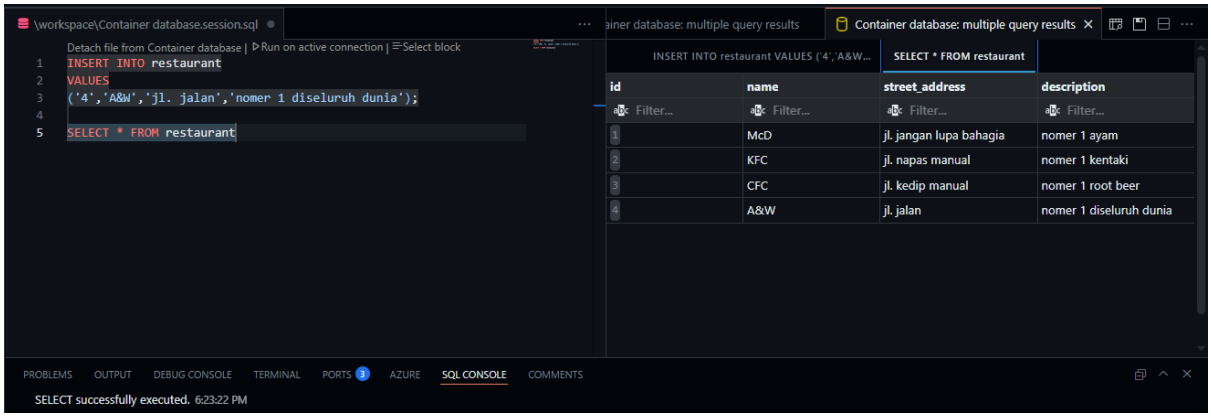
1. Create (Insert):
   - Insert a new restaurant into the restaurant table.

INSERT INTO restaurant

VALUES

('4','A&W','jl. jalan','nomer 1 diseluruh dunia');

Result:



-- Case2

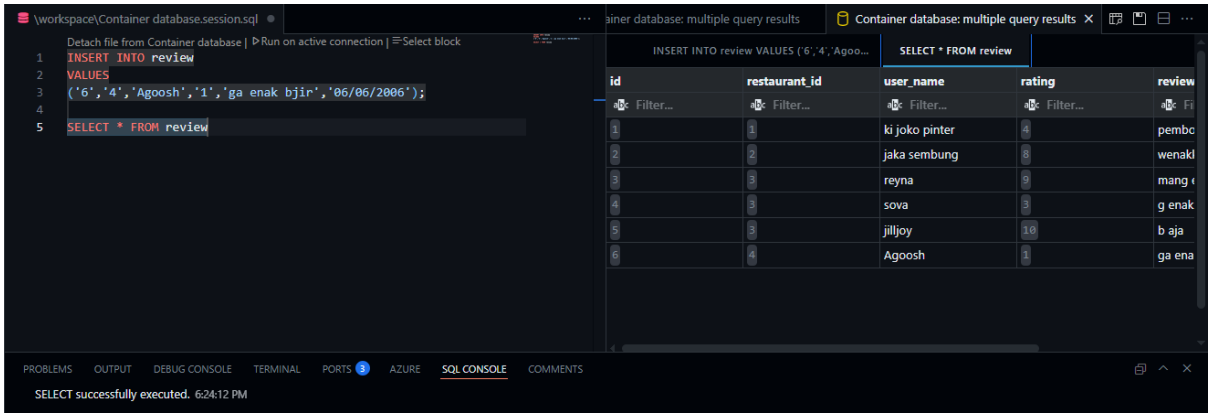
- Insert a new review for an existing restaurant.

INSERT INTO review

VALUES

('6','4','Agoosh','1','ga enak bjir','06/06/2006');

Result:



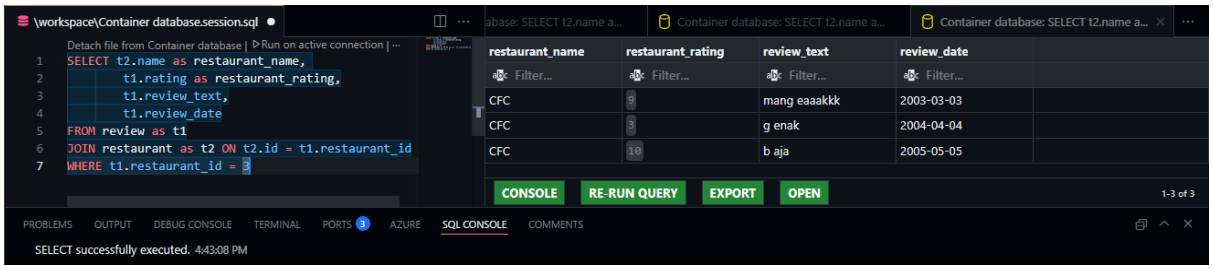
-- Read

2. Read (Select):
   - Retrieve all reviews for a specific restaurant using the `restaurant_id`.
   - Retrieve all reviews with a rating of 4 or higher.
   - Use a JOIN to display a list of restaurants along with their reviews.

-- Case1

```
SELECT t2.name as restaurant_name,
        t1.rating as restaurant_rating,
        t1.review_text,
        t1.review_date
FROM review as t1
JOIN restaurant as t2 ON t2.id = t1.restaurant_id
WHERE t1.restaurant_id = 3
```

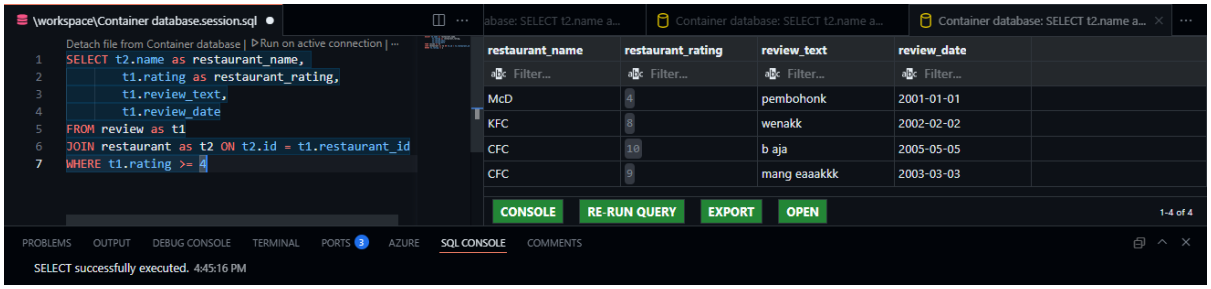
Result:



-- Case2

```
SELECT t2.name as restaurant_name,
        t1.rating as restaurant_rating,
        t1.review_text,
        t1.review_date
FROM review as t1
JOIN restaurant as t2 ON t2.id = t1.restaurant_id
WHERE t1.rating >= 4
```

Result:



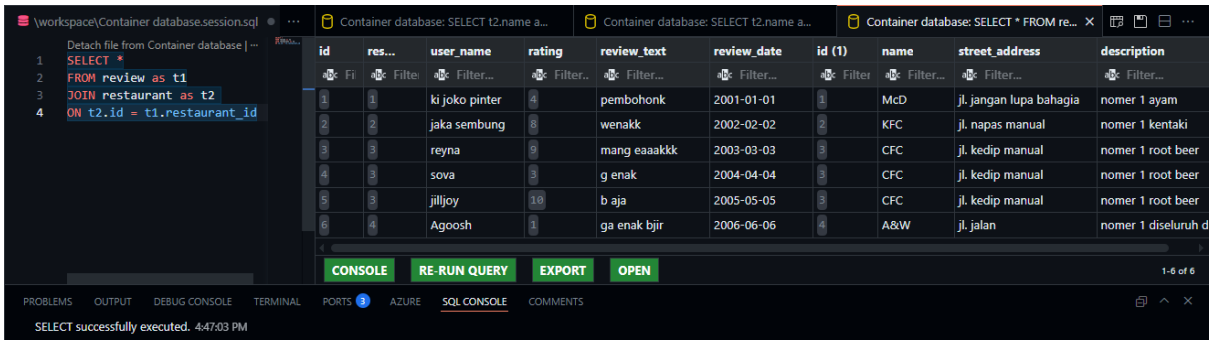
-- Case3

SELECT *

FROM review as t1

JOIN restaurant as t2 ON t2.id = t1.restaurant_id

Result:



-- Update

3. Update:
   - Update the description of one restaurant.
   - Update the rating of a specific review.

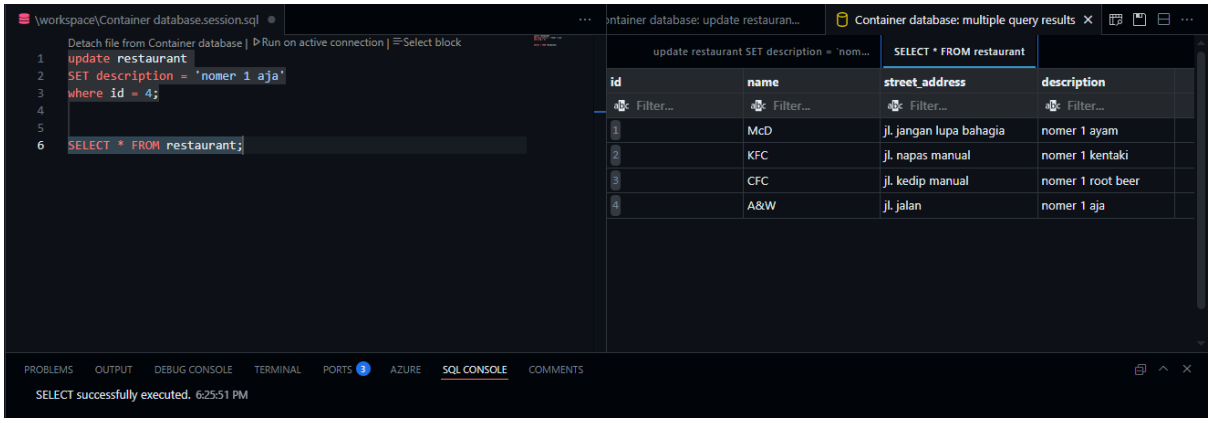
-- Case1

update restaurant

SET description = 'nomer 1 aja'

where id = 4

result:



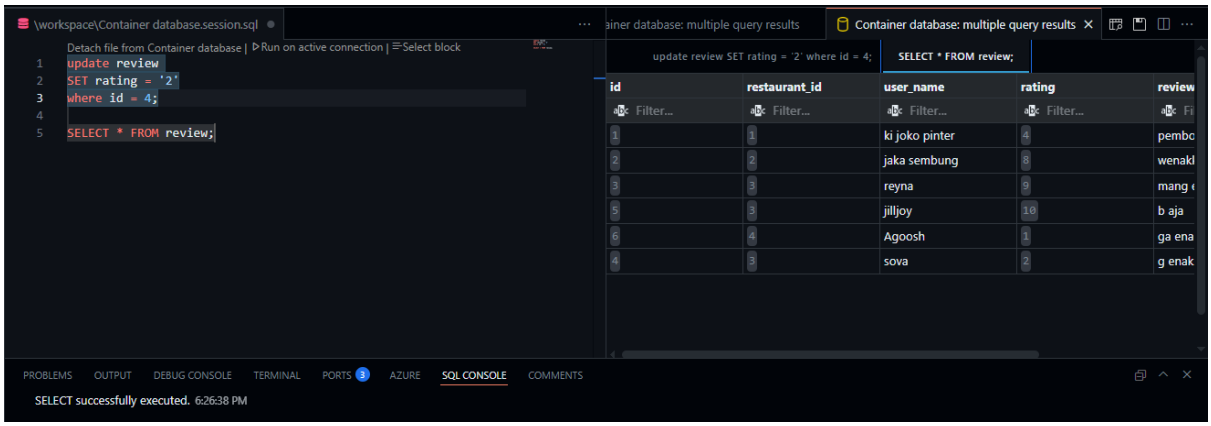
-- Case2

update review

SET rating = '2'

where id = 4

result:



-- Delete

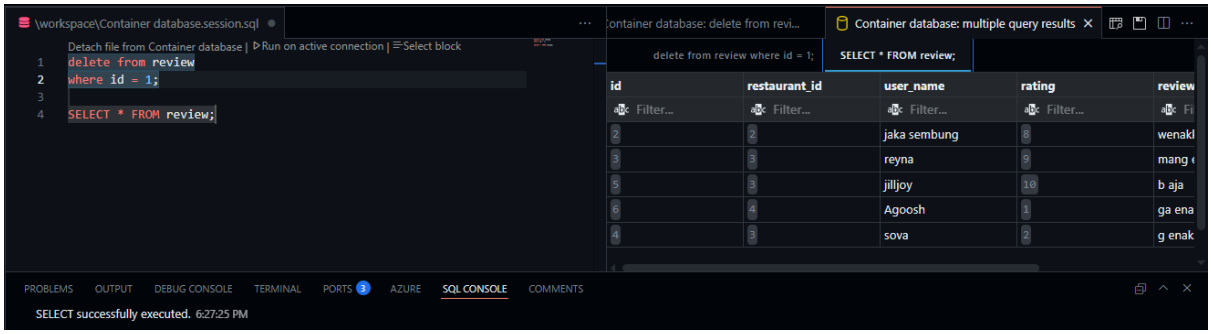
4. Delete:
   - Delete one review based on id.
   - Delete a restaurant and ensure its associated reviews are also deleted (using cascade).

-- Case1

delete from review

where id = 1

Result:

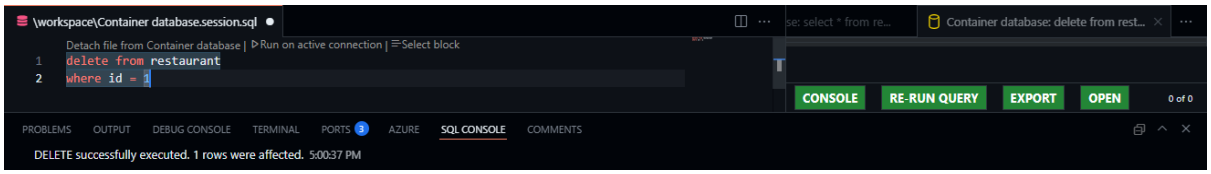


-- Case2

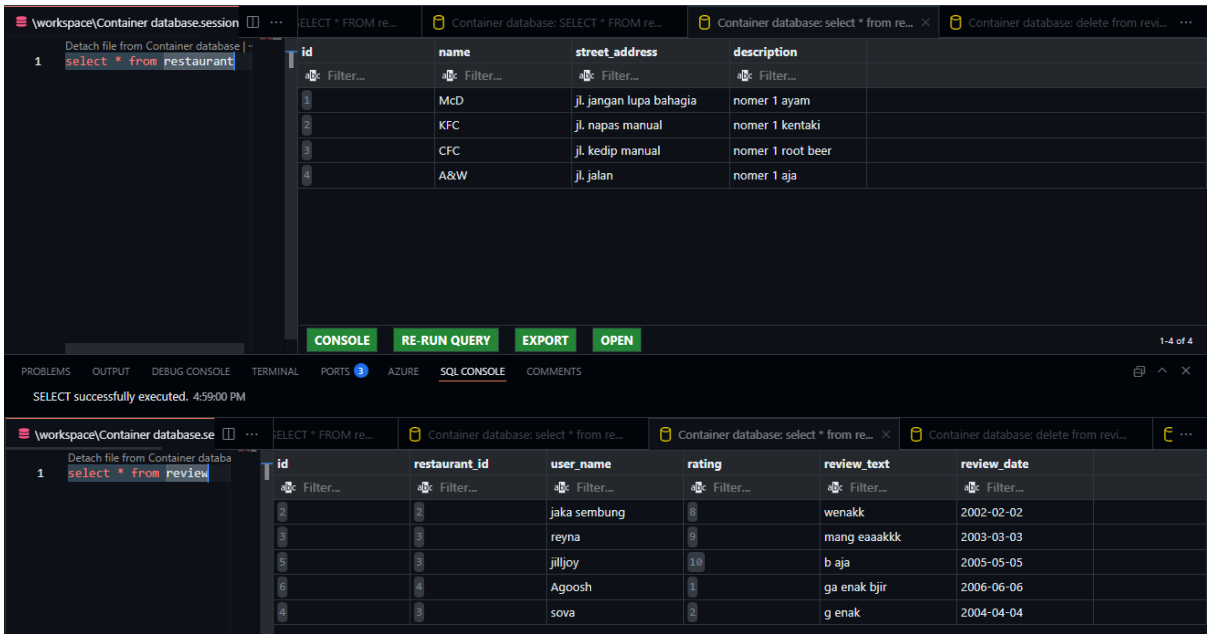
delete from restaurant

where id = 1

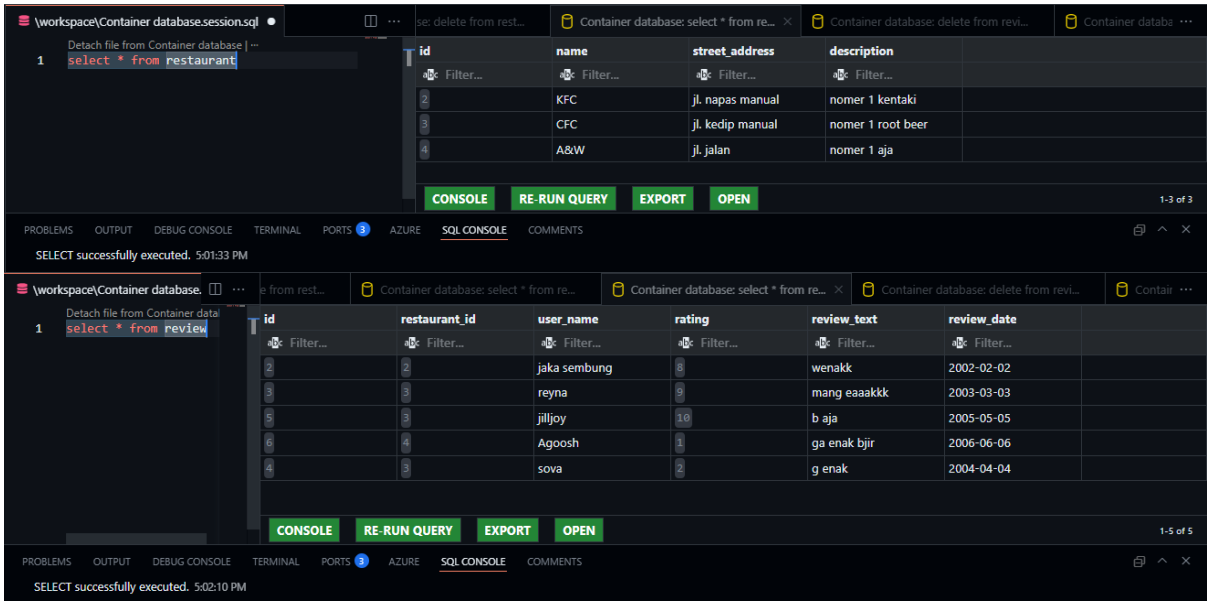
Result:



Before



After



-- Additional Queries

D. Additional Queries

1. Find the highest-rated restaurant based on the average rating of all its reviews.
2. Find the number of reviews each restaurant has received.
3. Display the most recent review for each restaurant.

-- Case1

select t1.name as restaurant_name, avg(t2.rating) as restaurant_rating

from restaurant as t1

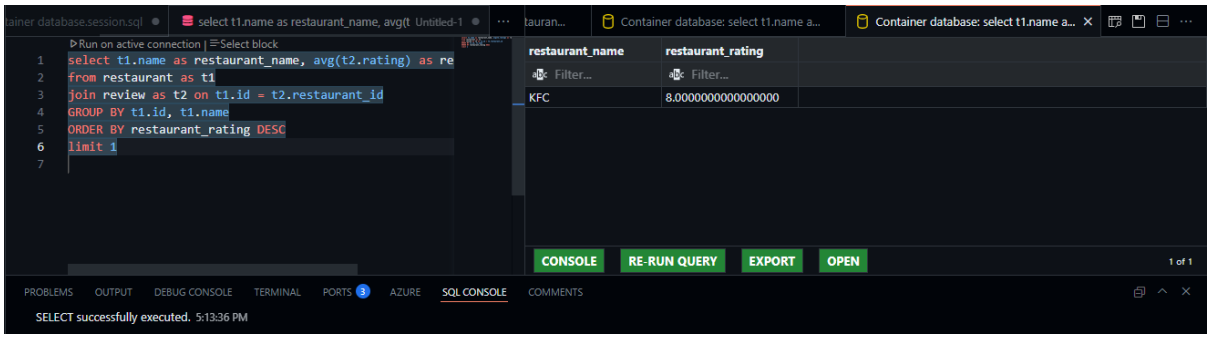
join review as t2 on t1.id = t2.restaurant_id

GROUP BY t1.id, t1.name

ORDER BY restaurant_rating DESC

limit 1

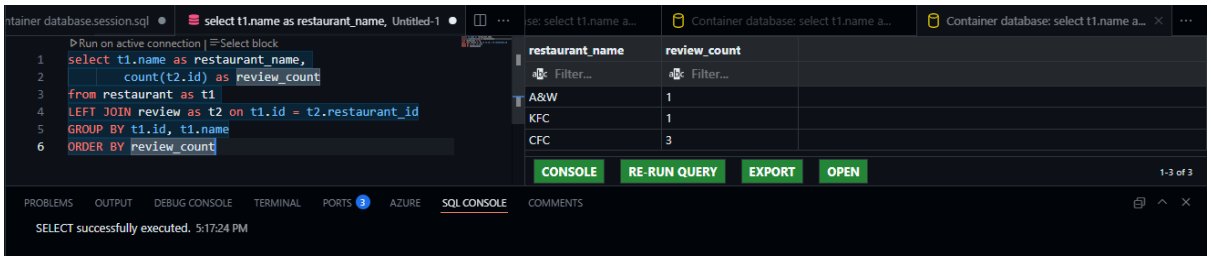
Result:



-- Case2

```
select t1.name as restaurant_name,
        count(t2.id) as review_count
from restaurant as t1
LEFT JOIN review as t2 on t1.id = t2.restaurant_id
GROUP BY t1.id, t1.name
ORDER BY review_count
```

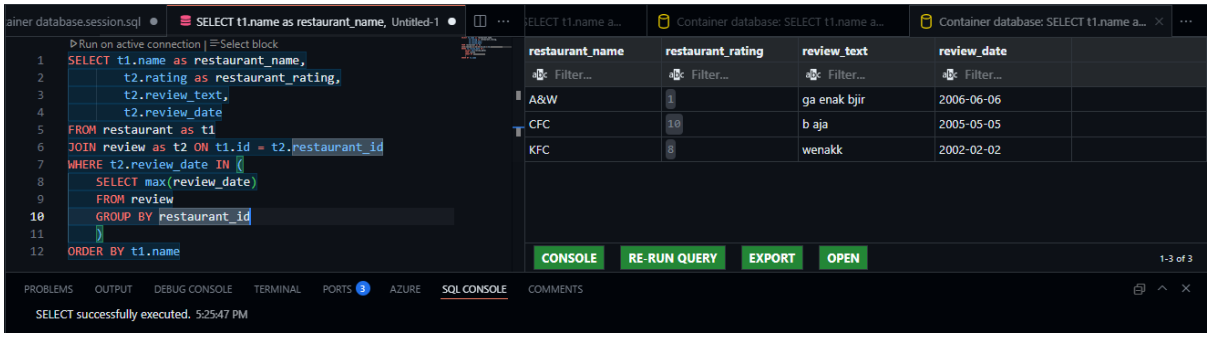
Result:



-- Case3

```
SELECT t1.name as restaurant_name,
        t2.rating as restaurant_rating,
        t2.review_text,
        t2.review_date
FROM restaurant as t1
JOIN review as t2 ON t1.id = t2.restaurant_id
WHERE t2.review_date IN (
    SELECT max(review_date)
    FROM review
    GROUP BY restaurant_id
    )
ORDER BY t1.name
```

Result:



-- Extra Credits

Extra Credit (Optional)

1. Create a menu table, similar to the one used in our class session, and insert at least 3 menu items for each restaurant.
2. Write a query to display each restaurant with its menu and the average rating from its reviews.

-- Case1

```
create table menu (
   id INT NOT NULL,
   restaurant_id INT NOT NULL,
   item_name VARCHAR(255) NOT NULL,
   price DECIMAL(10,2),
   description TEXT,
   primary key (id),
   foreign key (restaurant_id) references restaurant(id) on delete cascade
);

INSERT INTO menu
VALUES
('1','2','babi kecap','50000','babi dikecapin'),
('2','2','babi panggang','45000','babi dipanggang'),
('3','2','babi goreng','30000','babi digoreng garing'),
```

('4','3','babi sambal matah','55000','babi dikasi sambel matah'),

('5','3','babi guling','75000','babi diguling'),

('6','3','sate babi','60000','babi disate'),

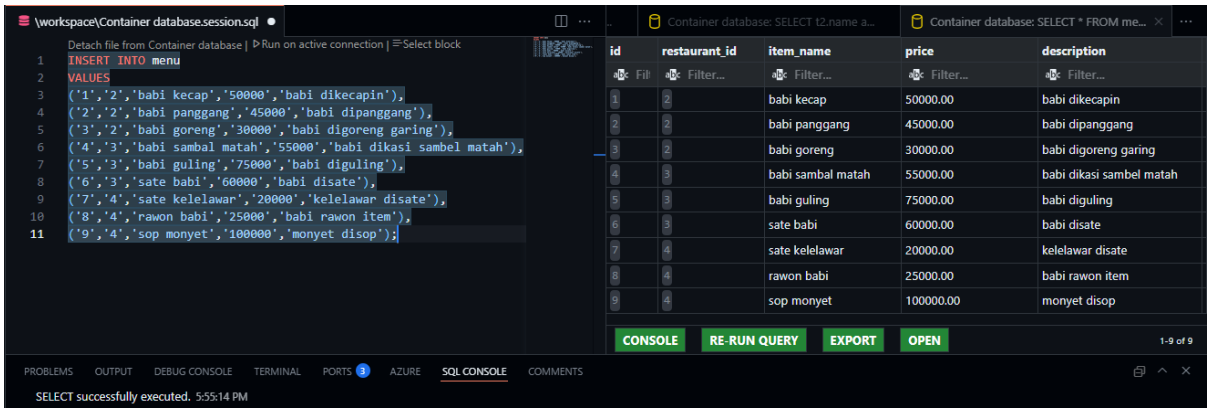
('7','4','sate kelelawar','20000','kelelawar disate'),

('8','4','rawon babi','25000','babi rawon item'),

('9','4','sop monyet','100000','monyet disop');

SELECT * FROM menu

Result:



-- Case2

SELECT t2.name as restaurant_name,

t3.item_name as restaurant_menu,

AVG(t1.rating) as restaurant_rating

FROM review as t1

JOIN restaurant as t2 ON t1.restaurant_id = t2.id

JOIN menu as t3 ON t2.id = t3.restaurant_id

GROUP BY t2.id, t2.name, t3.item_name

ORDER BY restaurant_rating DESC

Result:

\workspace\Container database.session.sql

Detach file from Container database | Run on active connection | Select block

```sql
SELECT t2.name as restaurant_name,
       t3.item_name as restaurant_menu,
       AVG(t1.rating) as restaurant_rating
FROM review as t1
JOIN restaurant as t2 ON t1.restaurant_id = t2.id
JOIN menu as t3 ON t2.id = t3.restaurant_id
GROUP BY t2.id, t2.name, t3.item_name
ORDER BY restaurant_rating DESC
```

Container database: SELECT t2.name a...

Container database: SELECT t2.name a...

restaurant_name	restaurant_menu	restaurant_rating
KFC	babi kecap	8.0000000000000000
KFC	babi goreng	8.0000000000000000
KFC	babi panggang	8.0000000000000000
CFC	babi sambal matah	7.0000000000000000
CFC	babi guling	7.0000000000000000
CFC	sate babi	7.0000000000000000
A&W	sop monyet	1.0000000000000000...
A&W	rawon babi	1.0000000000000000...
A&W	sate kelelawar	1.0000000000000000...

CONSOLE RE-RUN QUERY EXPORT OPEN 1-9 of 9

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 3 AZURE SQL CONSOLE COMMENTS

SELECT successfully executed. 5:42:44 PM