

01/09/2023 (Day 4)

Azure SQL

```
CREATE TABLE CUSTOMERS (  
    C_ID INT PRIMARY KEY,  
    F_NAME VARCHAR(50),  
    L_NAME VARCHAR(50),  
    EMAIL VARCHAR(100)  
);
```

Microsoft Azure

Search resources, services, and docs (G+/)

Home > Microsoft.SqlDatabase.newDatabaseNewServer_318f9d0fe9ce4c8995c24 | Overview > juhi-database (juhi-server/juhi-database)

juhi-database (juhi-server/juhi-database) | Query editor (preview)

SQL database

Search

« Login + New Query ↑ Open query ↗ Feedback ⓘ Getting started

Query 1 ×

Run ☐ Cancel query ↓ Save query ↓ Export data as ▾ Show only Editor

```
1 CREATE TABLE CUSTOMERS (  
2     C_ID INT PRIMARY KEY,  
3     F_NAME VARCHAR(50),  
4     L_NAME VARCHAR(50),  
5     EMAIL VARCHAR(100)  
6 );
```

Results Messages

Query succeeded: Affected rows: 0

Query succeeded | 0s

```
CREATE TABLE ORDERS (  
    O_ID INT PRIMARY KEY,  
    C_ID INT,  
    O_DATE DATE,  
    TOTAL_AMT DECIMAL(10,2),  
    FOREIGN KEY (C_ID) REFERENCES CUSTOMERS (C_ID)  
);
```

Microsoft Azure Search resources, services, and docs (G+/)

Home > juhi-database (juhi-server/juhi-database)

juhi-database (juhi-server/juhi-database) | Query editor (preview)

SQL database

Search Login + New Query Open query Feedback Getting started

Overview
Activity log
Tags
Diagnose and solve problems
Query editor (preview)

Settings

- Compute + storage
- Connection strings
- Maintenance
- Properties
- Locks

Data management

- Replicas
- Sync to other databases

Integrations

Query 1 X

Run Cancel query Save query Export data as Show only Editor

```
1 CREATE TABLE ORDERS (  
2   O_ID INT PRIMARY KEY,  
3   C_ID INT,  
4   O_DATE DATE,  
5   TOTAL_AMT DECIMAL(10,2),  
6   FOREIGN KEY (C_ID) REFERENCES CUSTOMERS(C_ID)  
7 );  
8
```

Results Messages

Query succeeded: Affected rows: 0

Query succeeded | 0s

```
INSERT INTO CUSTOMERS (C_ID, F_NAME, L_NAME, EMAIL)  
VALUES  
(1, 'JOHN', 'DOE', 'JOHN.DOE@EXAMPLE.COM'),  
(2, 'JANE', 'SMITH', 'JANE.SMITH@EXAMPLE.COM')
```

Microsoft Azure Search resources, services, and docs (G+/)

Home > juhi-database (juhi-server/juhi-database)

juhi-database (juhi-server/juhi-database) | Query editor (preview)

SQL database

Search Login + New Query Open query Feedback Getting started

Overview
Activity log
Tags
Diagnose and solve problems
Query editor (preview)

Settings

- Compute + storage
- Connection strings
- Maintenance
- Properties
- Locks

Data management

- Replicas
- Sync to other databases

Integrations

Query 1 X

Run Cancel query Save query Export data as Show only Editor

```
1 INSERT INTO CUSTOMERS (C_ID, F_NAME, L_NAME, EMAIL)  
2 VALUES  
3 (1, 'JOHN', 'DOE', 'JOHN.DOE@EXAMPLE.COM'),  
4 (2, 'JANE', 'SMITH', 'JANE.SMITH@EXAMPLE.COM')
```

Results Messages

Query succeeded: Affected rows: 2

Query succeeded | 6s

```
SELECT * FROM CUSTOMERS;
```

Microsoft Azure Search resources, services, and docs (G+)

Home > juhi-database (juhi-server/juhi-database)

juhi-database (juhi-server/juhi-database) | Query editor (preview)

SQL database

Search

« Login + New Query ↑ Open query ↗ Feedback 📖 Getting started

Overview
Activity log
Tags
Diagnose and solve problems
Query editor (preview)

Settings

Compute + storage
Connection strings
Maintenance
Properties
Locks

Data management

Replicas
Sync to other databases

Integrations

Query 1 ✕

Run ☐ Cancel query ⬇ Save query ⬇ Export data as ▾ 📊 Show only Editor

```
1 SELECT * FROM CUSTOMERS;  
2
```

Results Messages

Search to filter items...

C_ID	F_NAME	L_NAME	EMAIL
1	JOHN	DOE	JOHN.DOE@EXAMPLE.COM
2	JANE	SMITH	JANE.SMITH@EXAMPLE.COM

Query succeeded | 0s

```
INSERT INTO ORDERS (O_ID, C_ID, O_DATE, TOTAL_AMT)  
VALUES  
  (1, 1, '2023-08-01', 50.00),  
  (2, 2, '2023-08-15', 75.00)
```

Microsoft Azure Search resources, services, and docs (G+)

Home > juhi-database (juhi-server/juhi-database)

juhi-database (juhi-server/juhi-database) | Query editor (preview)

SQL database

Search

« Login + New Query ↑ Open query ↗ Feedback 📖 Getting started

Overview
Activity log
Tags
Diagnose and solve problems
Query editor (preview)

Settings

Compute + storage
Connection strings
Maintenance
Properties
Locks

Data management

Replicas
Sync to other databases

Integrations

Query 1 ✕

Run ☐ Cancel query ⬇ Save query ⬇ Export data as ▾ 📊 Show only Editor

```
1 INSERT INTO ORDERS (O_ID, C_ID, O_DATE, TOTAL_AMT)  
2 VALUES  
3   (1, 1, '2023-08-01', 50.00),  
4   (2, 2, '2023-08-15', 75.00)
```


Results Messages

Query succeeded: Affected rows: 2

Query succeeded | 0s

```
SELECT * FROM ORDERS;
```

Query 1 ✕

▶ Run ☐ Cancel query ⬇ Save query ⬇ Export data as ▾  Show only Editor

```
1 SELECT * FROM ORDERS;
```

Results Messages

🔍 Search to filter items...

O_ID	C_ID	O_DATE	TOTAL_AMT
1	1	2023-08-01T00:00:00.0000000	50.00
2	2	2023-08-15T00:00:00.0000000	75.00

INNER JOIN

```
SELECT CUSTOMERS.*, ORDERS.*
FROM CUSTOMERS
INNER JOIN ORDERS ON CUSTOMERS.C_ID = ORDERS.C_ID;
```

Query 1 ✕

▶ Run ☐ Cancel query ⬇ Save query ⬇ Export data as ▾  Show only Editor

```
1 SELECT CUSTOMERS.*, ORDERS.*
2 FROM CUSTOMERS
3 INNER JOIN ORDERS ON CUSTOMERS.C_ID = ORDERS.C_ID;
```

Results Messages

🔍 Search to filter items...

C_ID	F_NAME	L_NAME	EMAIL	O_ID	C_ID	O_DATE
1	JOHN	DOE	JOHN.DOE@EXAMP...	1	1	2023-08-01T00:00:00.0000000
2	JANE	SMITH	JANE.SMITH@EXAM...	2	2	2023-08-15T00:00:00.0000000

▶ Query succeeded | 0s

LEFT JOIN

```
SELECT CUSTOMERS.*, ORDERS.*
FROM CUSTOMERS
LEFT JOIN ORDERS ON CUSTOMERS.C_ID = ORDERS.C_ID;
```


Query 1 ✕

 Run ☐ Cancel query  Save query  Export data as  Show only Editor

```
1 SELECT CUSTOMERS.*, ORDERS.*
2 FROM CUSTOMERS
3 LEFT JOIN ORDERS ON CUSTOMERS.C_ID = ORDERS.C_ID;
```





Results Messages

Search to filter items...						
C_ID	F_NAME	L_NAME	EMAIL	O_ID	C_ID	O_DATE
1	JOHN	DOE	JOHN.DOE@EXAMP...	1	1	2023-01
2	JANE	SMITH	JANE.SMITH@EXAM...	2	2	2023-01

 Query succeeded | 0s

```
UNION
SELECT C_ID FROM CUSTOMERS
UNION
SELECT C_ID FROM ORDERS;
```


Query 1 ✕

 Run ☐ Cancel query  Save query  Export data as  Show only Editor

```
1 SELECT C_ID FROM CUSTOMERS
2 UNION
3 SELECT C_ID FROM ORDERS;
```

Results Messages

Search to filter items...						
C_ID						
1						
2						

 Query succeeded | 0s

>

Input

Run SQL

<

```
CREATE TABLE GALLERIES (  
  ID INT PRIMARY KEY,  
  CITY VARCHAR(50)  
);  
  
INSERT INTO GALLERIES (ID, CITY)  
VALUES  
(1, 'JAIPUR'),  
(2, 'KOLKATA'),  
(3, 'MADHUBANI')
```

Output

SQL query successfully executed. However, the result set is empty.

>

Input

Run SQL

<

```
SELECT * FROM GALLERIES;
```

Output

ID	CITY
1	JAIPUR
2	KOLKATA
3	MADHUBANI

>

Input

Run SQL

<

```
CREATE TABLE PAINTINGS (  
  ID INT PRIMARY KEY,  
  NAME VARCHAR(50),  
  G_ID INT,  
  PRICE INT  
);  
  
INSERT INTO PAINTINGS (ID, NAME, G_ID, PRICE)  
VALUES  
  (1, 'PATTERNS', 3, 5000),  
  (2, 'RINGER', 1, 4500),  
  (3, 'GIFT', 1, 3200),  
  (4, 'VIOLIN LESSON', 2, 6700),  
  (5, 'CURIOSITY', 2, 9800)
```

Output

SQL query successfully executed. However, the result set is empty.

>

Input

Run SQL

<

```
SELECT * FROM PAINTINGS
```

Output

ID	NAME	G_ID	PRICE
1	PATTERNS	3	5000
2	RINGER	1	4500
3	GIFT	1	3200
4	VIOLIN LESSON	2	6700
5	CURIOSITY	2	9800

>

Input

Run SQL

<

```
CREATE TABLE SALES_AGENTS (  
  ID INT PRIMARY KEY,  
  L_NAME VARCHAR(50),  
  F_NAME VARCHAR(50),  
  G_ID INT,  
  A_FEE INT  
);  
  
INSERT INTO SALES_AGENTS (ID, L_NAME, F_NAME, G_ID, A_FEE)  
VALUES  
(1, 'BROWN', 'DENIS', 2, 2250),  
(2, 'WHITE', 'KATE', 3, 3120),  
(3, 'BLACK', 'SARAH', 2, 1640),  
(4, 'SMITH', 'HELEN', 1, 4500),  
(5, 'STEWART', 'TOM', 3, 2130)
```

Output

SQL query successfully executed. However, the result set is empty.

>

Input

Run SQL

<

```
SELECT * FROM SALES_AGENTS
```

Output

ID	L_NAME	F_NAME	G_ID	A_FEE
1	BROWN	DENIS	2	2250
2	WHITE	KATE	3	3120
3	BLACK	SARAH	2	1640
4	SMITH	HELEN	1	4500
5	STEWART	TOM	3	2130

>

Input

Run SQL

<

```
CREATE TABLE MANAGERS (  
  ID INT PRIMARY KEY,  
  G_ID INT  
);  
  
INSERT INTO MANAGERS (ID, G_ID)  
VALUES  
(1,2),  
(2,3),  
(4,1)
```

Output

SQL query successfully executed. However, the result set is empty.

>

Input

Run SQL

<

```
SELECT * FROM MANAGERS;
```

Output

ID	G_ID
1	2
2	3
4	1

SINGLE ROW SUBQUERY

>

Input

Run SQL

<

```
SELECT NAME, PRICE,
(SELECT AVG(PRICE) FROM PAINTINGS) FROM PAINTINGS;
```

Output

NAME	PRICE	(SELECT AVG(PRICE) FROM PAINTINGS)
PATTERNS	5000	5840
RINGER	4500	5840
GIFT	3200	5840
VIOLIN LESSON	6700	5840
CURIOSITY	9800	5840

>

Input

Run SQL

<

```
SELECT AVG(A_FEE)
FROM SALES_AGENTS;
```

Output

AVG(A_FEE)
2728

>

Input

Run SQL

<

```
SELECT AVG(A_FEE)
FROM SALES_AGENTS
WHERE ID
NOT IN
(SELECT ID FROM MANAGERS);
```

Output

AVG(A_FEE)
1885

MULTIPLE ROW SUBQUERY

Subqueries that return multiple rows as output to their parent query .

>

Input

Run SQL

<

```
SELECT *
FROM SALES_AGENTS
WHERE ID
NOT IN (SELECT ID FROM MANAGERS)
```

Output

ID	L_NAME	F_NAME	G_ID	A_FEE
3	BLACK	SARAH	2	1640
5	STEWART	TOM	3	2130

MULTI-COLUMN SUBQUERY

>

Input

⌵

🌙

⋮

Run SQL

<

```

SELECT ID, NAME, PRICE
FROM PAINTINGS
WHERE (NAME, PRICE)
IN (SELECT NAME, MIN(PRICE) FROM PAINTINGS)
  
```

Output

ID	NAME	PRICE
3	GIFT	3200

CORRELATED SUBQUERY

Subqueries that return multiple columns as output depending on the information obtained from the parent query.

>

Input

⌵

🌙

⋮

Run SQL

<

```

SELECT CITY,
(SELECT COUNT(*)
FROM PAINTINGS P
WHERE G.ID=P.G_ID)
COUNT_PAINTING
FROM GALLERIES G;
  
```

Output

CITY	COUNT_PAINTING
JAIPUR	2
KOLKATA	2
MADHUBANI	1

Find out the information of those sales agents whose agency fees is greater than or equal to the agency fees of their gallery

>

Input

⌂

🌙

⋮

Run SQL

<

```

SELECT F_NAME, L_NAME, A_FEE
FROM SALES_AGENTS SA1
WHERE SA1.A_FEE >= (SELECT AVG(A_FEE)
                    FROM SALES_AGENTS SA2
                    WHERE SA2.G_ID=SA1.G_ID);

```

Output

F_NAME	L_NAME	A_FEE
DENIS	BROWN	2250
KATE	WHITE	3120
HELEN	SMITH	4500

Nested Subquery

Subqueries that are inside another sub-query are called nested subquery.

>

Input

⌂

🌙

⋮

Run SQL

<

```

SELECT NAME AS PAINTING, PRICE,
(SELECT AVG(PRICE)
FROM PAINTINGS
WHERE PRICE IN
(SELECT PRICE
FROM PAINTINGS
WHERE PRICE>=5000)
) AS AVG_PRICE
FROM PAINTINGS;

```

Output

PAINTING	PRICE	AVG_PRICE
PATTERNS	5000	7166.666666666667
RINGER	4500	7166.666666666667
GIFT	3200	7166.666666666667
VIOLIN LESSON	6700	7166.666666666667
CURIOSITY	9800	7166.666666666667

VIEW

Input

```
CREATE VIEW VW_PAINTINGPRICE AS
SELECT NAME AS PAINTING, PRICE,
(SELECT AVG(PRICE)
FROM PAINTINGS
WHERE PRICE IN
(SELECT PRICE
FROM PAINTINGS
WHERE PRICE>=5000)
) AS AVG_PRICE
FROM PAINTINGS;

SELECT * FROM VW_PAINTINGPRICE;
```

Run SQL

Output

PAINTING	PRICE	AVG_PRICE
PATTERNS	5000	7166.666666666667
RINGER	4500	7166.666666666667
GIFT	3200	7166.666666666667
VIOLIN LESSON	6700	7166.666666666667
CURIOSITY	9800	7166.666666666667

INDEXES
CLUSTERED INDEXES