

Omkar Mankame

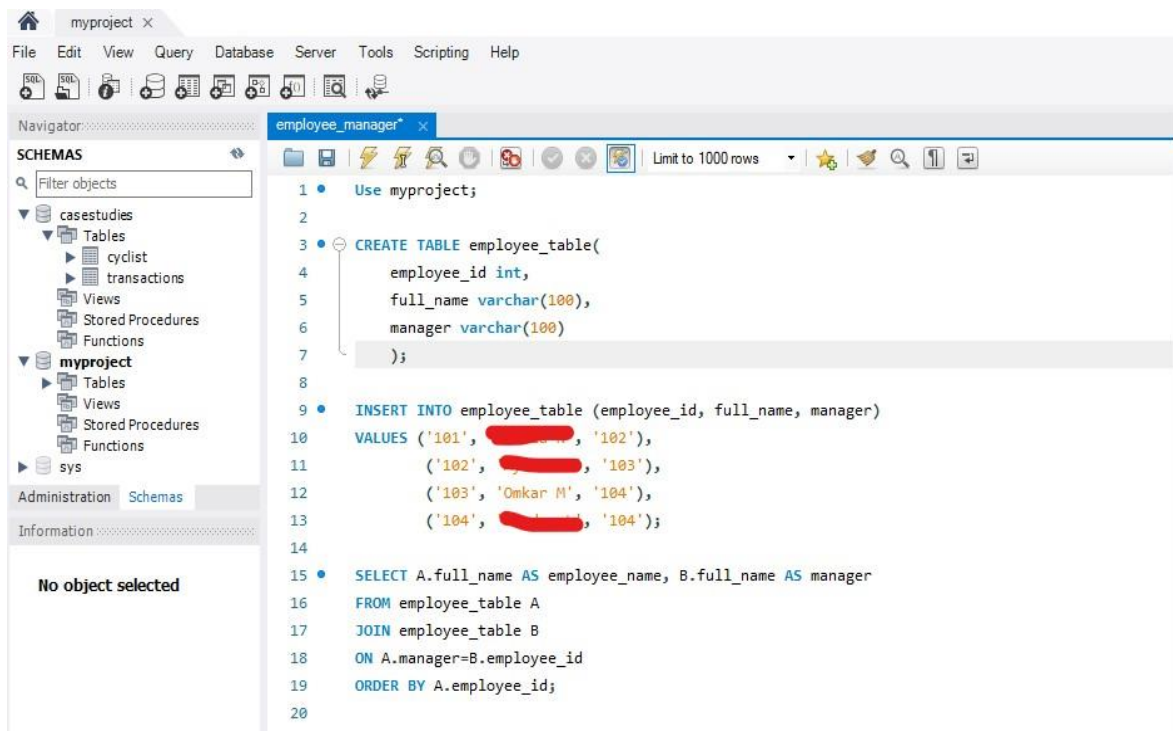
Few SQL projects

4 Sept 2024

I have done a few projects using SQL for data analysis. I have shared three complex and concise projects in this document. Others have been shared in the GitHub repository. The code is also mentioned in the GitHub codespace.

Project 1 - Employee_Manager Table

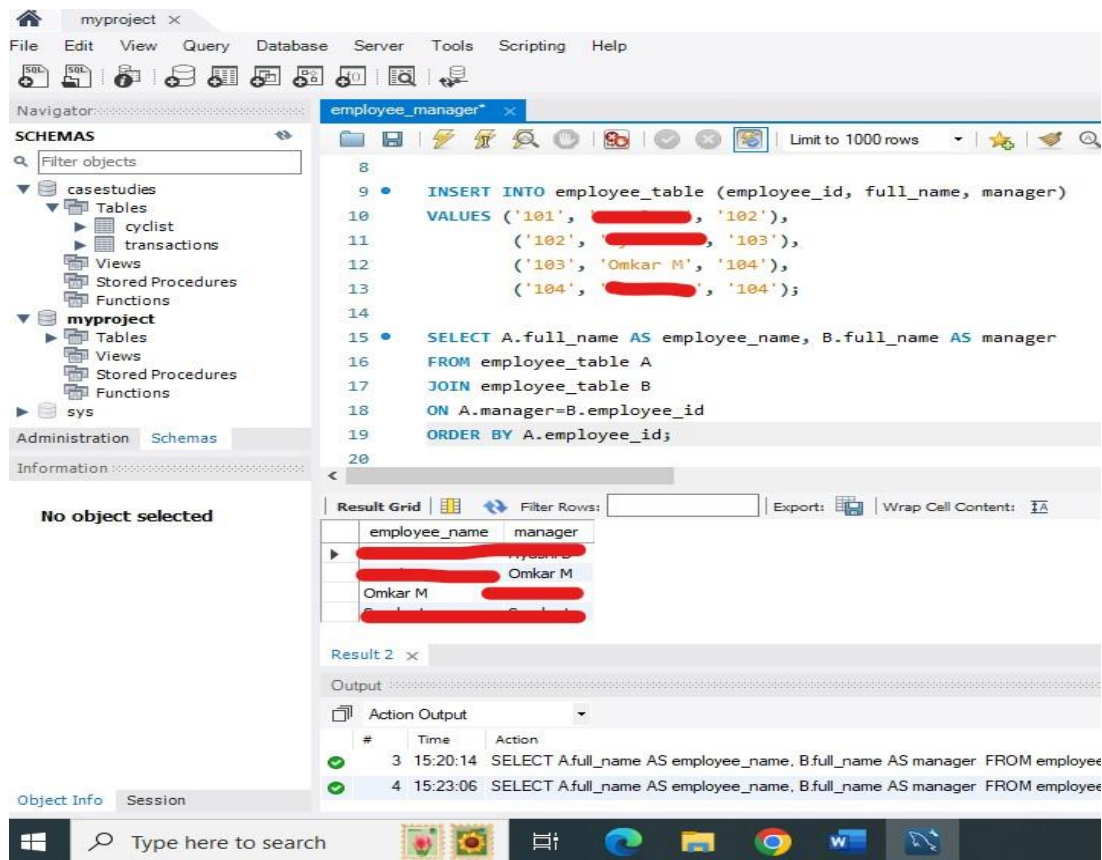
1. Created database.
2. Created a table with employees, id and manager. Inserted values.



The screenshot shows a SQL IDE interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar displays a 'SCHEMAS' tree with folders for 'casestudies', 'myproject', and 'sys'. The 'myproject' folder is expanded, showing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor window, titled 'employee_manager', contains the following SQL code:

```
1 • Use myproject;
2
3 • CREATE TABLE employee_table(
4     employee_id int,
5     full_name varchar(100),
6     manager varchar(100)
7 );
8
9 • INSERT INTO employee_table (employee_id, full_name, manager)
10 VALUES ('101', [REDACTED], '102'),
11          ('102', [REDACTED], '103'),
12          ('103', 'Omkar M', '104'),
13          ('104', [REDACTED], '104');
14
15 • SELECT A.full_name AS employee_name, B.full_name AS manager
16 FROM employee_table A
17 JOIN employee_table B
18 ON A.manager=B.employee_id
19 ORDER BY A.employee_id;
20
```

3. Joined table using self join to produce result as seen below.



SQL code (Also available in GitHub Codespace) –

Use myproject;

```
CREATE TABLE employee_table(
```

```
    employee_id int,
```

```
    full_name varchar(100),
```

```
    manager varchar(100)
```

```
);
```

```
INSERT INTO employee_table (employee_id, full_name, manager)
```

```
VALUES ('101',          , '102'),
```

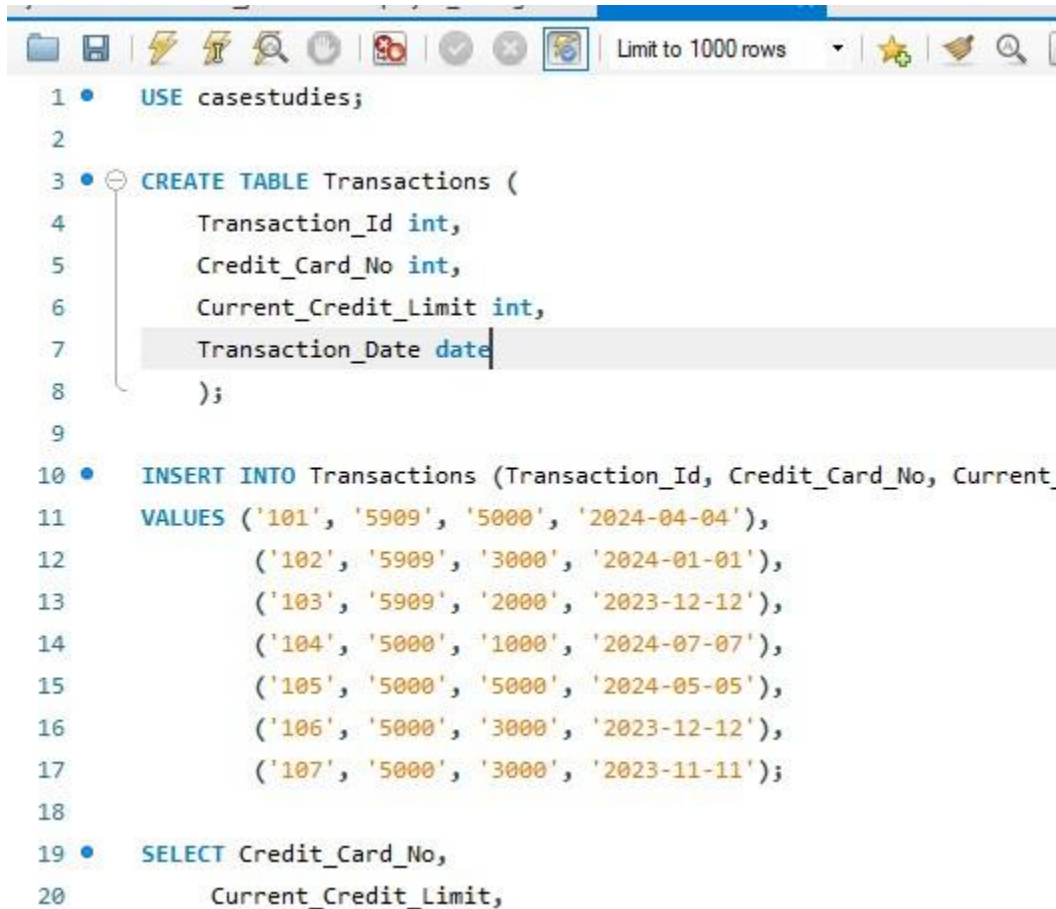
```
        ('102',          , '103'),
```

```
('103', 'Omkar M', '104'),  
( '104', ' ', '104');
```

```
SELECT A.full_name AS employee_name, B.full_name AS manager  
FROM employee_table A  
JOIN employee_table B  
ON A.manager=B.employee_id  
ORDER BY A.employee_id;
```

Project 2 – Credit Limit of Credit Card

1. Created database and Table with transaction id, credit card number, credit card limit and transaction date.
2. Inserted values in the table.



```
1 • USE casestudies;
2
3 • CREATE TABLE Transactions (
4     Transaction_Id int,
5     Credit_Card_No int,
6     Current_Credit_Limit int,
7     Transaction_Date date
8 );
9
10 • INSERT INTO Transactions (Transaction_Id, Credit_Card_No, Current_
11 VALUES ('101', '5909', '5000', '2024-04-04'),
12         ('102', '5909', '3000', '2024-01-01'),
13         ('103', '5909', '2000', '2023-12-12'),
14         ('104', '5000', '1000', '2024-07-07'),
15         ('105', '5000', '5000', '2024-05-05'),
16         ('106', '5000', '3000', '2023-12-12'),
17         ('107', '5000', '3000', '2023-11-11');
18
19 • SELECT Credit_Card_No,
20         Current_Credit_Limit,
```

3. Understood data and framed results using joins.

```

18
19 • SELECT Credit_Card_No,
20         Current_Credit_Limit,
21         max(Transaction_Date)
22 FROM transactions
23 GROUP BY Credit_Card_No, Current_Credit_Limit;
24
25 • SELECT Credit_Card_No,
26         Current_Credit_Limit
27 FROM transactions
28 WHERE Transaction_Date = (SELECT max(Transaction_Date)
29                           FROM transactions);
30
31 /*
32 SELECT Credit_Card_No,
33         Current_Credit_Limit
34 FROM transactions
35 WHERE Transaction_Date = (SELECT max(Transaction_Date)
36                           FROM transactions
37                           GROUP BY Credit_Card_No)

```

4. Final result was attained using joins, group by and alias.

The screenshot shows the SQL Developer interface with a query window titled 'Credit Card Limit'. The query uses a subquery to find the maximum transaction date for each credit card and then joins it back to the transactions table to retrieve the credit card number and current credit limit.

```

64 FROM transactions
65 GROUP BY Credit_Card_No)
66 ON Credit_Card_No.home = groupedtt.home
67 AND tt.datetime = groupedtt.MaxDateTime;
68 */
69
70 • SELECT Credit_Card_No,
71         Current_Credit_Limit
72 FROM transactions AS a
73 WHERE Transaction_Date = (SELECT max(Transaction_Date)
74                           FROM transactions as b
75                           WHERE a.Credit_Card_No = b.Credit_Card_No);
76

```

The result grid at the bottom shows the following data:

Credit_Card_No	Current_Credit_Limit
5909	5000
5000	1000

Automatic toolbar to move caret position

SQL code (Also available in GitHub Codespace) –

USE casestudies;

```
CREATE TABLE Transactions (  
    Transaction_Id int,  
    Credit_Card_No int,  
    Current_Credit_Limit int,  
    Transaction_Date date  
);
```

```
INSERT INTO Transactions (Transaction_Id, Credit_Card_No, Current_Credit_Limit, Transaction_Date)  
VALUES ('101', '5909', '5000', '2024-04-04'),  
    ('102', '5909', '3000', '2024-01-01'),  
    ('103', '5909', '2000', '2023-12-12'),  
    ('104', '5000', '1000', '2024-07-07'),  
    ('105', '5000', '5000', '2024-05-05'),  
    ('106', '5000', '3000', '2023-12-12'),  
    ('107', '5000', '3000', '2023-11-11');
```

```
SELECT Credit_Card_No,  
    Current_Credit_Limit,  
    max(Transaction_Date)  
FROM transactions  
GROUP BY Credit_Card_No, Current_Credit_Limit;
```

```
SELECT Credit_Card_No,  
    Current_Credit_Limit  
FROM transactions  
WHERE Transaction_Date = (SELECT max(Transaction_Date)
```

FROM transactions);

/*

SELECT Credit_Card_No,
Current_Credit_Limit

FROM transactions

WHERE Transaction_Date = (SELECT max(Transaction_Date)

FROM transactions

GROUP BY Credit_Card_No)

GROUP BY Transaction_Date

ORDER BY Transaction_Date;

SELECT Credit_Card_No,
Current_Credit_Limit

FROM transactions AS s1

JOIN (SELECT

Credit_Card_No,
Current_Credit_Limit,

MAX(`Transaction_Date`)

FROM transactions AS dt

GROUP BY Credit_Card_No) AS s2;

ON s1.Credit_Card_No = s2.Credit_Card_No

ORDER BY `datetime`;

SELECT Credit_Card_No,
Current_Credit_Limit

FROM transactions

Transaction_Date;

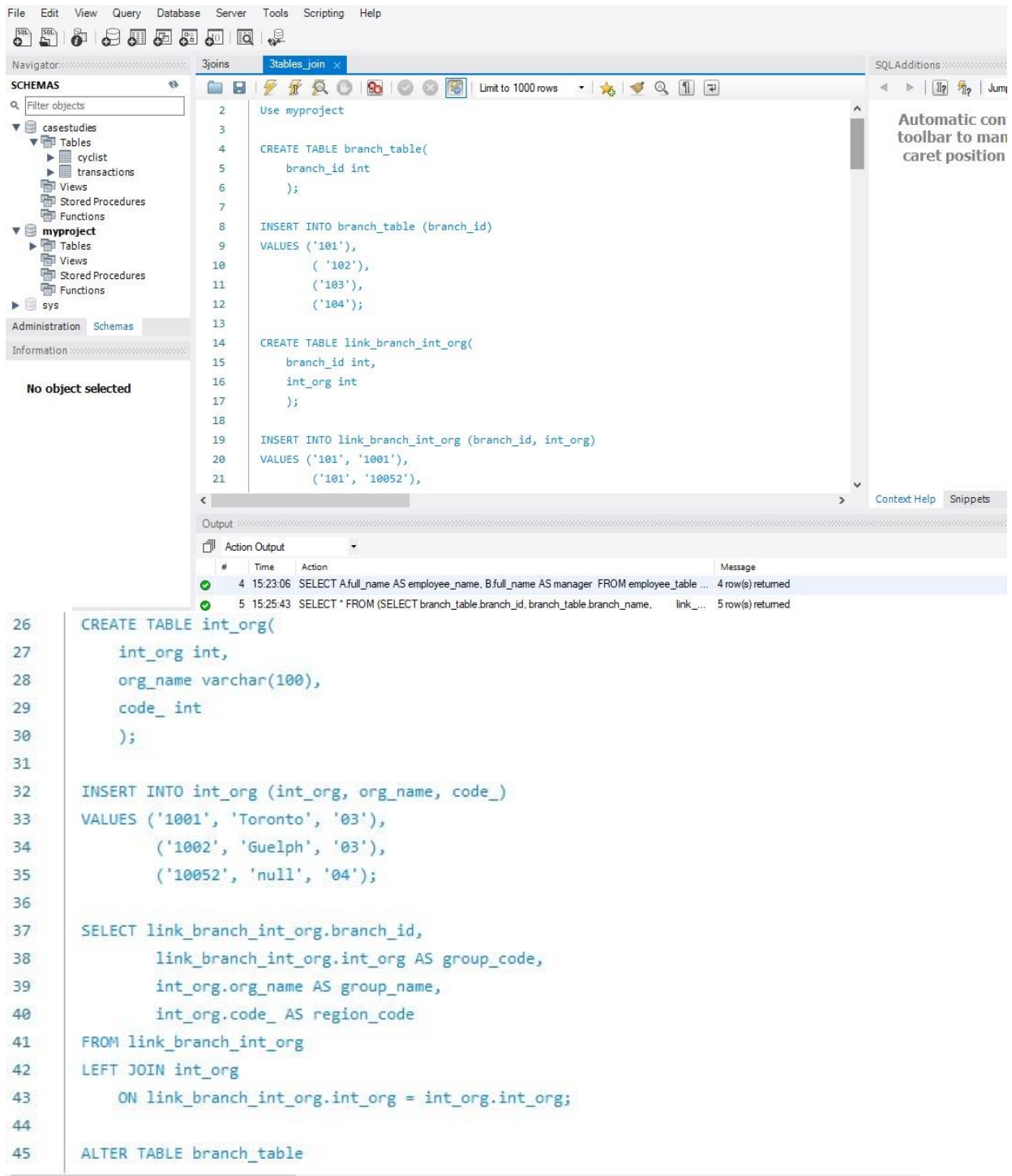
```
SELECT Credit_Card_No,  
       Current_Credit_Limit  
FROM transactions  
INNER JOIN  
    (SELECT Credit_Card_No, MAX(Transaction_Date)  
    FROM transactions  
    GROUP BY Credit_Card_No)  
ON Credit_Card_No.home = groupedtt.home  
AND tt.datetime = groupedtt.MaxDateTime;  
*/
```

```
SELECT Credit_Card_No,  
       Current_Credit_Limit  
FROM transactions AS a  
WHERE Transaction_Date = (SELECT max(Transaction_Date)  
                          FROM transactions as b  
                          WHERE a.Credit_Card_No = b.Credit_Card_No);
```

```
SELECT max(Transaction_Date)  
FROM transactions;
```


Project 3 – Joining 3 tables

1. Created a database and 3 tables.
2. Inserted values in the table.



The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'SCHEMAS' tree with 'myproject' selected. The central pane shows a SQL script titled '3tables_join' with the following content:

```
2 Use myproject
3
4 CREATE TABLE branch_table(
5     branch_id int
6 );
7
8 INSERT INTO branch_table (branch_id)
9 VALUES ('101'),
10        ('102'),
11        ('103'),
12        ('104');
13
14 CREATE TABLE link_branch_int_org(
15     branch_id int,
16     int_org int
17 );
18
19 INSERT INTO link_branch_int_org (branch_id, int_org)
20 VALUES ('101', '1001'),
21        ('101', '10052'),
```

The right pane shows the 'Output' window with the following results:

#	Time	Action	Message
4	15:23:06	SELECT A.full_name AS employee_name, B.full_name AS manager FROM employee_table ...	4 row(s) returned
5	15:25:43	SELECT * FROM (SELECT branch_table.branch_id, branch_table.branch_name, link_...	5 row(s) returned

Below the screenshot, the full SQL script is provided:

```
26 CREATE TABLE int_org(
27     int_org int,
28     org_name varchar(100),
29     code_int
30 );
31
32 INSERT INTO int_org (int_org, org_name, code_)
33 VALUES ('1001', 'Toronto', '03'),
34        ('1002', 'Guelph', '03'),
35        ('10052', 'null', '04');
36
37 SELECT link_branch_int_org.branch_id,
38        link_branch_int_org.int_org AS group_code,
39        int_org.org_name AS group_name,
40        int_org.code_ AS region_code
41 FROM link_branch_int_org
42 LEFT JOIN int_org
43     ON link_branch_int_org.int_org = int_org.int_org;
44
45 ALTER TABLE branch_table
```

```

100
101 • SELECT *
102 FROM (SELECT link_branch_int_org.branch_id,
103             link_branch_int_org.int_org,
104             int_org.org_name,
105             int_org.code_
106         FROM link_branch_int_org
107         LEFT JOIN int_org
108             ON link_branch_int_org.int_org = int_org.int_org
109         WHERE int_org.code_ = '03') AS A
110 LEFT JOIN (SELECT link_branch_int_org.branch_id,
111             link_branch_int_org.int_org,
112             int_org.org_name,
113             int_org.code_
114         FROM link_branch_int_org
115         LEFT JOIN int_org
116             ON link_branch_int_org.int_org = int_org.int_org
117         WHERE int_org.code_ = '04') AS B
118     ON A.int_org = B.int_org;

```

joins

Limit to 1000 rows

```

123     link_branch_int_org.int_org
124 FROM branch_table
125 INNER JOIN link_branch_int_org
126     ON link_branch_int_org.branch_id = branch_table.branch_id) AS A
127 LEFT JOIN (SELECT link_branch_int_org.branch_id,
128             link_branch_int_org.int_org,
129             int_org.org_name,
130             int_org.code_
131         FROM link_branch_int_org
132         LEFT JOIN int_org
133             ON link_branch_int_org.int_org = int_org.int_org
134         WHERE int_org.code_ = '03') AS B
135     ON A.int_org = B.int_org;

```

Result Grid

Filter Rows: Export: Wrap Cell Content: ☐

	branch_id	int_org	org_name	code_	branch_id	int_org	org_name	code_
▶	101	1001	Toronto	3	NULL	NULL	NULL	NULL
	102	1002	Guelph	3	NULL	NULL	NULL	NULL

- The complex query above created a join and another join in the subquery made it possible to get the desired result.

SQL code (Also available in GitHub Codespace) –

/*

Use myproject

```
CREATE TABLE branch_table(
```

```
    branch_id int
```

```
);
```

```
INSERT INTO branch_table (branch_id)
```

```
VALUES ('101'),
```

```
        ( '102'),
```

```
        ('103'),
```

```
        ('104');
```

```
CREATE TABLE link_branch_int_org(
```

```
    branch_id int,
```

```
    int_org int
```

```
);
```

```
INSERT INTO link_branch_int_org (branch_id, int_org)
```

```
VALUES ('101', '1001'),
```

```
        ('101', '10052'),
```

```
        ('102', '1002'),
```

```
        ('103', '10052'),
```

```
        ('104', null);
```

```
CREATE TABLE int_org(
```

```
    int_org int,
```

```
    org_name varchar(100),
```

```
code_int
```

```
);
```

```
INSERT INTO int_org (int_org, org_name, code_)
```

```
VALUES ('1001', 'Toronto', '03'),
```

```
        ('1002', 'Guelph', '03'),
```

```
        ('10052', 'null', '04');
```

```
SELECT link_branch_int_org.branch_id,
```

```
        link_branch_int_org.int_org AS group_code,
```

```
        int_org.org_name AS group_name,
```

```
        int_org.code_ AS region_code
```

```
FROM link_branch_int_org
```

```
LEFT JOIN int_org
```

```
    ON link_branch_int_org.int_org = int_org.int_org;
```

```
ALTER TABLE branch_table
```

```
ADD branch_name varchar(100);
```

```
ALTER TABLE branch_table
```

```
ADD PRIMARY KEY (branch_id);
```

```
UPDATE branch_table
```

```
SET branch_name ='Toronto'
```

```
WHERE branch_id = '101';
```

```
UPDATE branch_table
```

```
SET branch_name ='Guelph'
```

```
WHERE branch_id = '102';
```

```
UPDATE branch_table
SET branch_name ='Mississauga'
WHERE branch_id = '103';
```

```
UPDATE branch_table
SET branch_name ='Oakville'
WHERE branch_id = '104';
```

```
SELECT branch_table.branch_id,
       branch_table.branch_name,
       link_branch_int_org.int_org
FROM branch_table
INNER JOIN link_branch_int_org
       ON link_branch_int_org.branch_id = branch_table.branch_id;
```

```
SELECT link_branch_int_org.branch_id,
       link_branch_int_org.int_org,
       int_org.org_name,
       int_org.code_
FROM link_branch_int_org
LEFT JOIN int_org
       ON link_branch_int_org.int_org = int_org.int_org;
```

```
SELECT link_branch_int_org.branch_id,
       link_branch_int_org.int_org,
       int_org.org_name,
       int_org.code_
FROM link_branch_int_org
```

```
LEFT JOIN int_org
    ON link_branch_int_org.int_org = int_org.int_org
WHERE int_org.code_ = '03';
```

```
SELECT link_branch_int_org.branch_id,
       link_branch_int_org.int_org,
       int_org.org_name,
       int_org.code_
FROM link_branch_int_org
LEFT JOIN int_org
    ON link_branch_int_org.int_org = int_org.int_org
WHERE int_org.code_ = '04';
*/
```

```
SELECT *
FROM (SELECT link_branch_int_org.branch_id,
            link_branch_int_org.int_org,
            int_org.org_name,
            int_org.code_
FROM link_branch_int_org
LEFT JOIN int_org
    ON link_branch_int_org.int_org = int_org.int_org
WHERE int_org.code_ = '03') AS A
LEFT JOIN (SELECT link_branch_int_org.branch_id,
                link_branch_int_org.int_org,
                int_org.org_name,
                int_org.code_
FROM link_branch_int_org
LEFT JOIN int_org
```

```

        ON link_branch_int_org.int_org = int_org.int_org
WHERE int_org.code_ = '04') AS B

        ON A.int_org = B.int_org;

SELECT *
FROM (SELECT branch_table.branch_id,
             branch_table.branch_name,
             link_branch_int_org.int_org
FROM branch_table
INNER JOIN link_branch_int_org
        ON link_branch_int_org.branch_id = branch_table.branch_id) AS A
LEFT JOIN (SELECT link_branch_int_org.branch_id,
                 link_branch_int_org.int_org,
                 int_org.org_name,
                 int_org.code_
FROM link_branch_int_org
LEFT JOIN int_org
        ON link_branch_int_org.int_org = int_org.int_org
WHERE int_org.code_ = '03') AS B

        ON A.int_org = B.int_org;

```