AT82.02

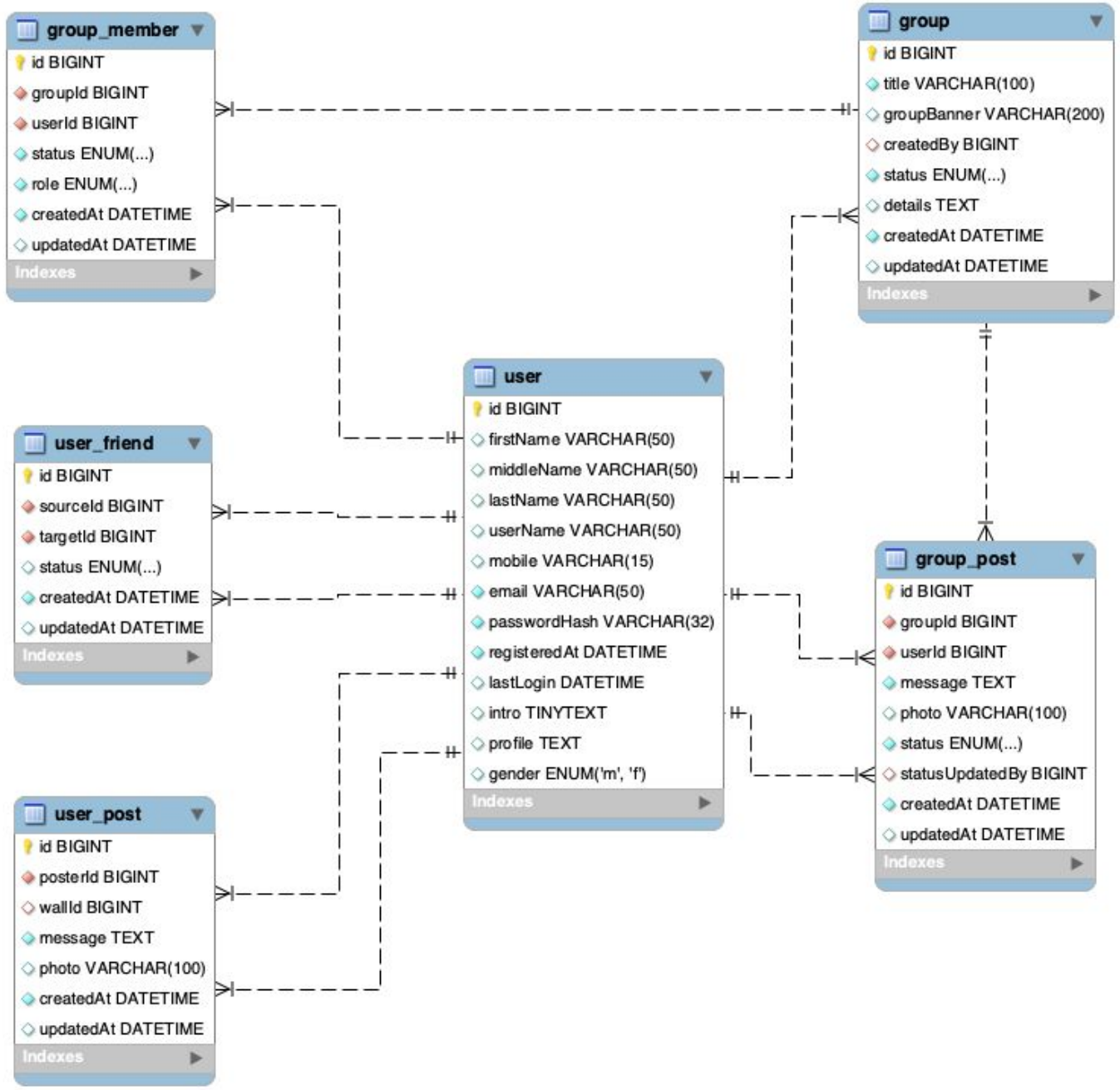
DATA MODELING AND MANAGEMENT

LAB04: SQL (PART III)

DS&AI

ASIAN INSTITUTE OF TECHNOLOGY 1959

Case Study: Social Network System



Outline

1

Set Operators in SQL

- **`UNION (ALL)`**
- **`INTERSECT`**
- **`MINUS`**

2

Joining Query

- **`[INNER] JOIN`**
- **`[LEFT | RIGHT | FULL] OUTER JOIN`**

3

Subqueries

- Subquery in **`WHERE`** Clause
- Subquery as a column
- Subquery in other operations

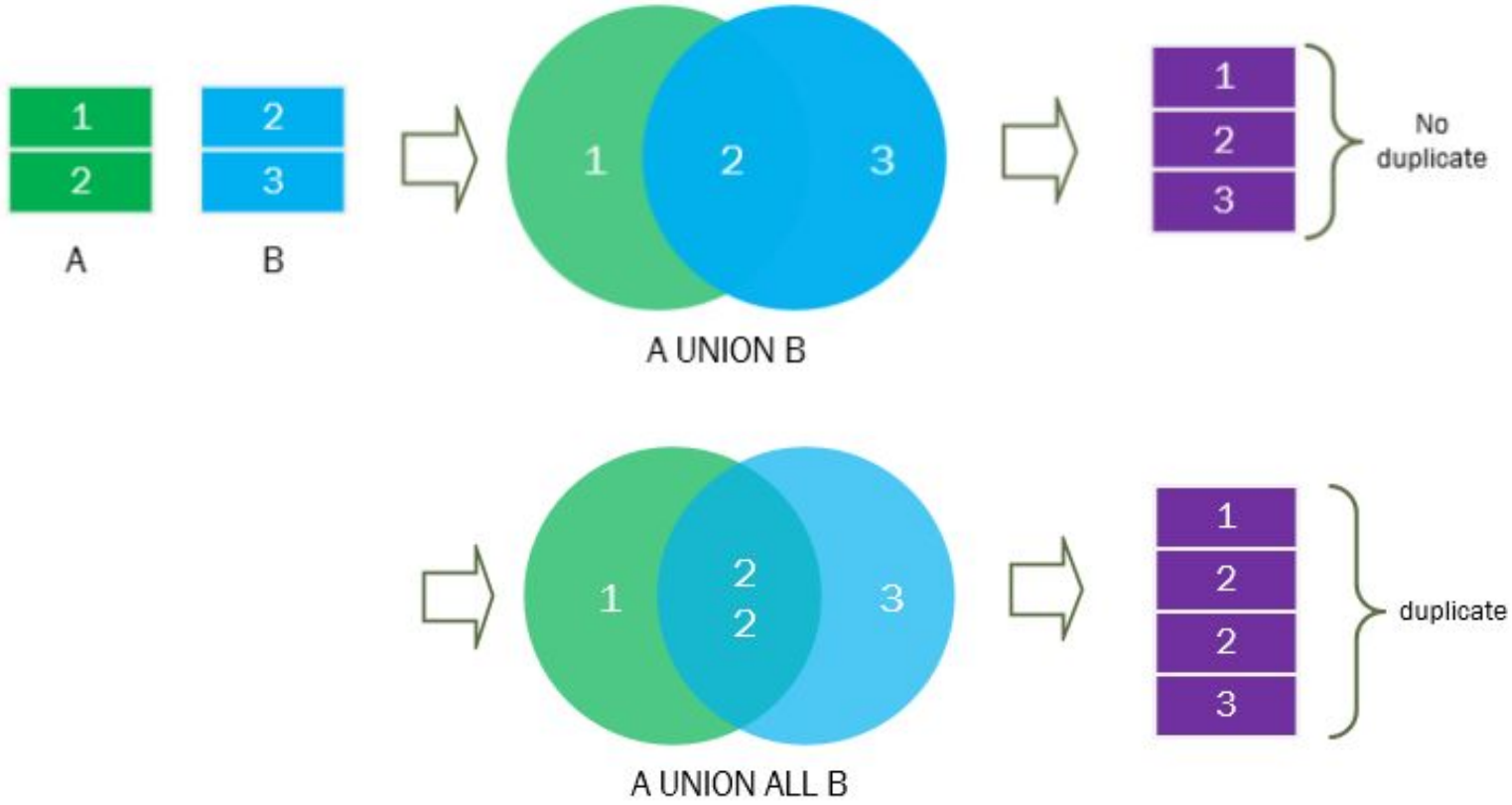
Using SET Operators

1. UNION and UNION ALL – combine result set of two or more queries into a single result set using the UNION and UNION ALL operators.
2. INTERSECT – return the intersection of two or more queries using the INTERSECT operator. **(Discarded)**
3. MINUS – subtract a result set from another result set using the MINUS operator. **(Discarded)**

Note: In MySQL, only UNION and UNION ALL are supported.

SQL UNION operator

The UNION operator combines result sets of two or more SELECT statements into a single result set.



SQL UNION Syntax

Using the UNION operator to combine result sets of two queries:

```
SELECT
    column1, column2
FROM
    table1
UNION [ALL]
SELECT
    column3, column4
FROM
    table2;
```

UNION

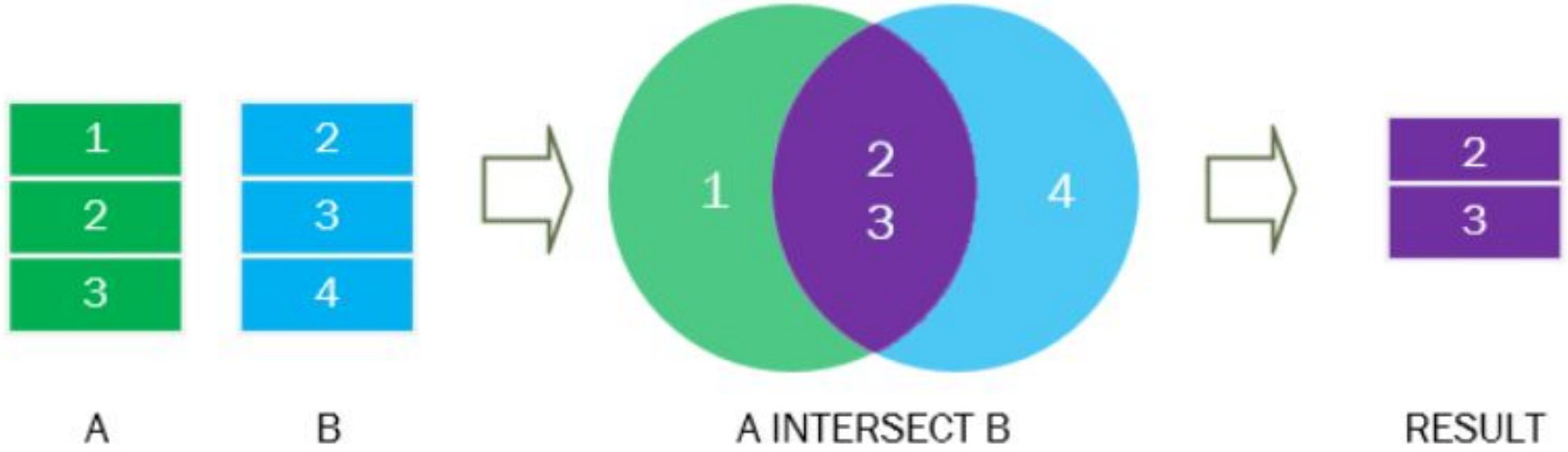
```
SELECT id, posterId,
'Wall post' AS postType, message
FROM user_post WHERE posterId=21
UNION
SELECT id, userId,
'Group post' AS postType, message
FROM group_post WHERE userId=21;
```

id	posterId	postType	message
83	21	Wall post	Quisque arcu libero, rutrum ac, lobort
517	21	Wall post	Cras mi pede, malesuada in, imperdie
806	21	Wall post	Maecenas leo odio, condimentum id, l
972	21	Wall post	Donec ut dolor. Morbi vel lectus in qu
90	21	Group post	Integer aliquet, massa id lobortis conv
207	21	Group post	Etiam pretium iaculis justo. In hac hal
224	21	Group post	Maecenas tristique, est et tempus sem
233	21	Group post	Integer ac neque. Duis bibendum. Mo

Note: *The number of columns of both queries must be EQUAL.*

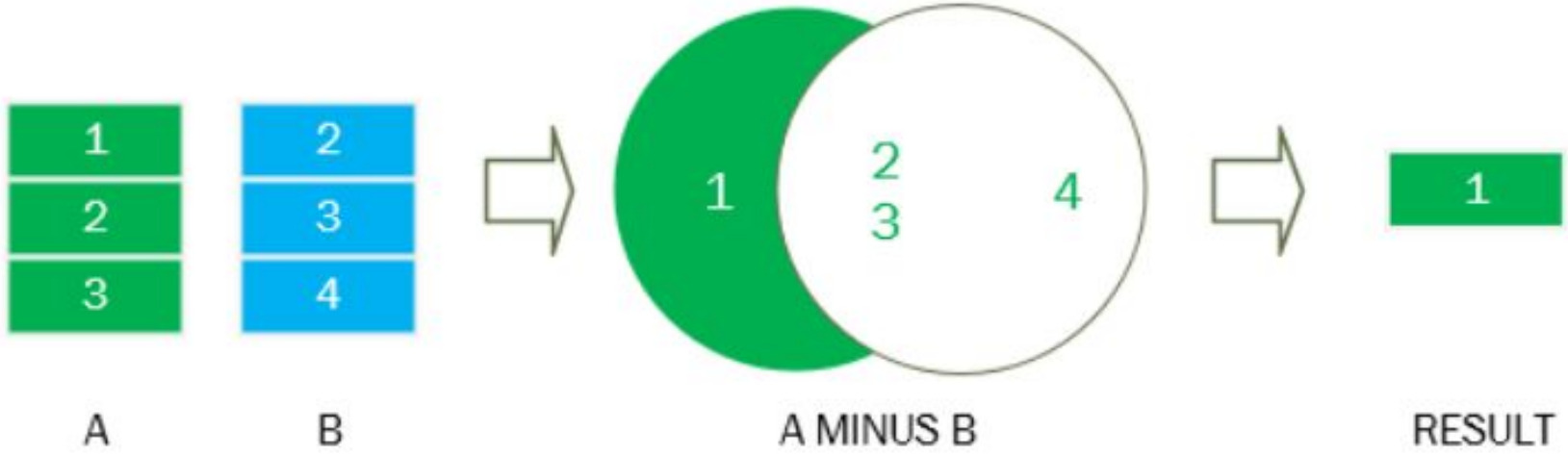
SQL INTERSECT operator

The INTERSECT operator is a set operator that returns distinct rows of two or more result sets from SELECT statements.



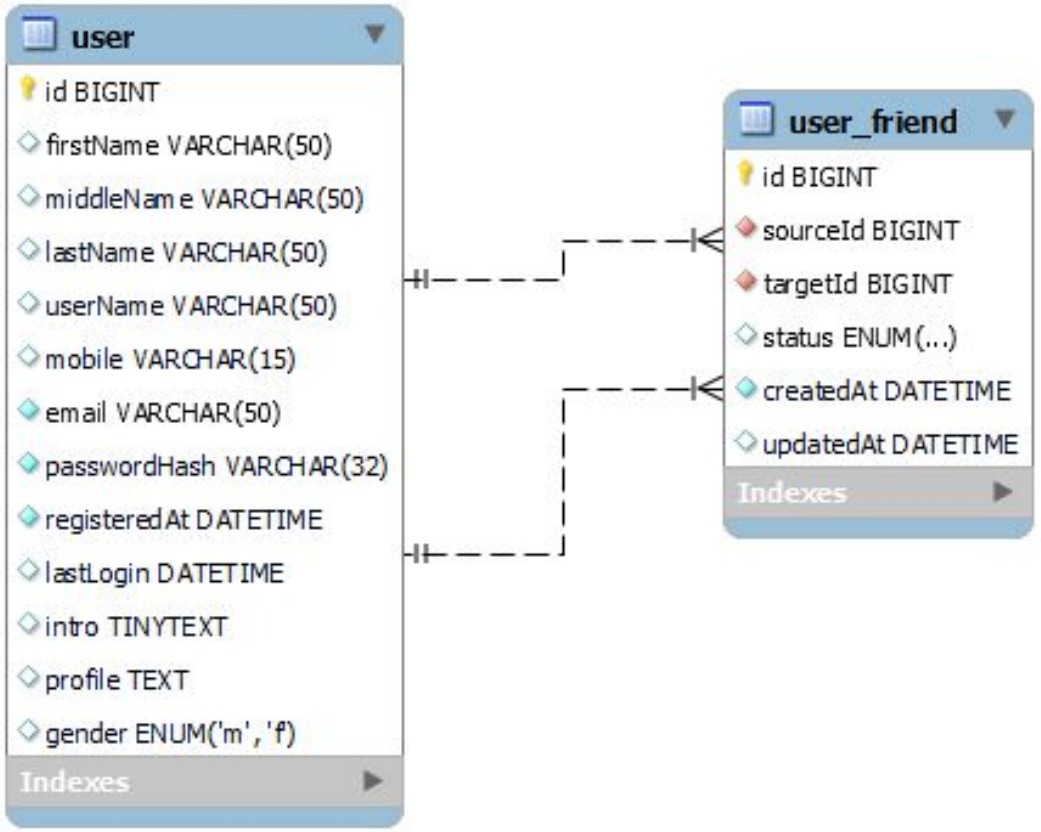
SQL MINUS operator

The MINUS operator is a set operator that allows you to subtract one result set from another result set.



Equivalent of INTERSECT and MINUS

Unfortunately, **`INTERSECT`** and **`MINUS`** operators are not available in MySQL, so we need some **`JOIN`** operations to achieve the above operators.



Joining Queries

We need to join queries when:

- to include data from multiple tables
- to link two or more related tables based on common value, especially KEYS

Consider relationships between two tables, **`user`** and **`user_friend`**

- **`user_friend`** has 2 FKs and both refer to id of a **`user`** table.

Joining Types

1. **INNER JOIN**

   Combines data from two tables that have matching values.

2. **LEFT [RIGHT] OUTER JOIN**

   Combines record from left/right table and the record that is common in both tables.

3. **FULL OUTER JOIN**

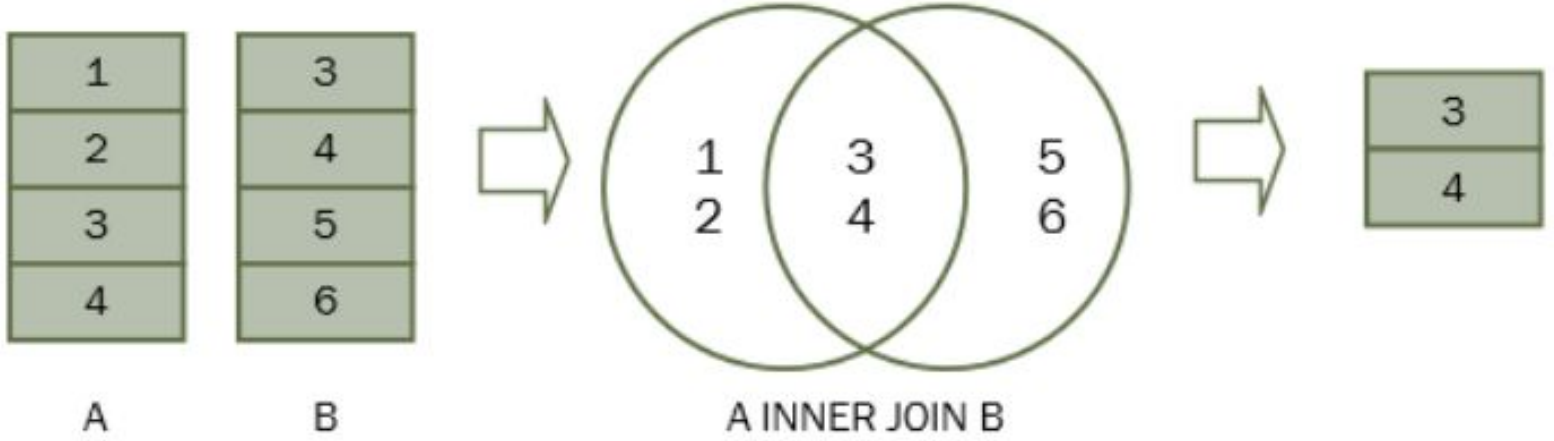
   Combines rows from both tables when there is a match in left or right table record.

4. **SELF JOIN**

   Regular join feature but the table is joined with itself.

SQL INNER JOIN

The inner join clause links two (or more) tables by a relationship between two columns.

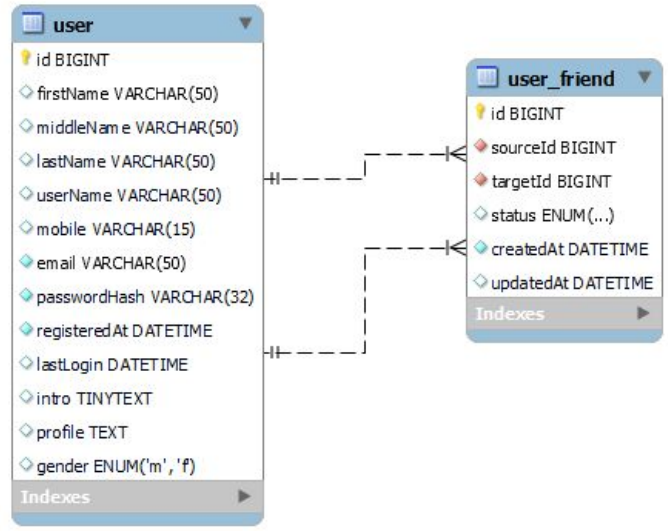


SQL INNER JOIN Syntax

Suppose the column name of the A & B tables is n, the following statement illustrates the inner join clause:

```
SELECT
  A.n
FROM A
INNER JOIN B ON B.n = A.n;
```

INNER JOIN example (1)

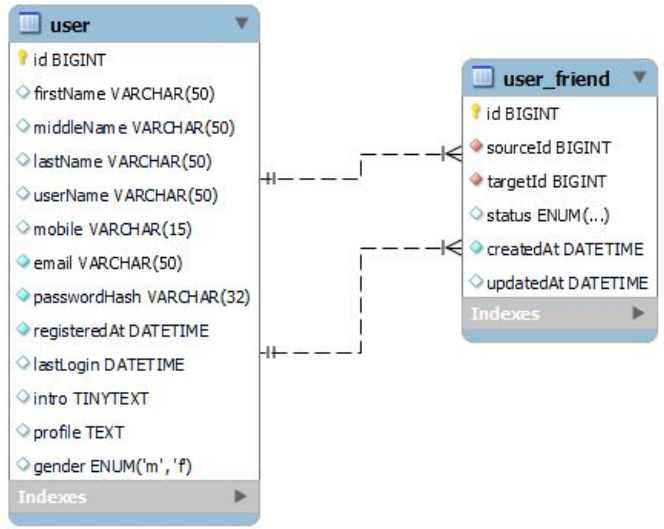


Show the target firstname and lastname in **`user_friend`** table.

```
SELECT tg.sourceId, tg.targetId,
    u.firstName AS targetFirstName, u.lastName AS targetLastName
FROM user u INNER JOIN user_friend tg
    ON u.id = tg.targetId;
```

sourceId	targetId	targetFirstName	targetLastName
1	15	Sandy	Bree
2	11	Ardelia	Madner
2	12	Tiffany	Schwandner
2	17	Reed	Rontree
2	21	Tabina	Debow
3	3	Merilee	Cornely
3	6	Silvie	Vernazza
3	10	Lacey	Bosence

INNER JOIN example (2)



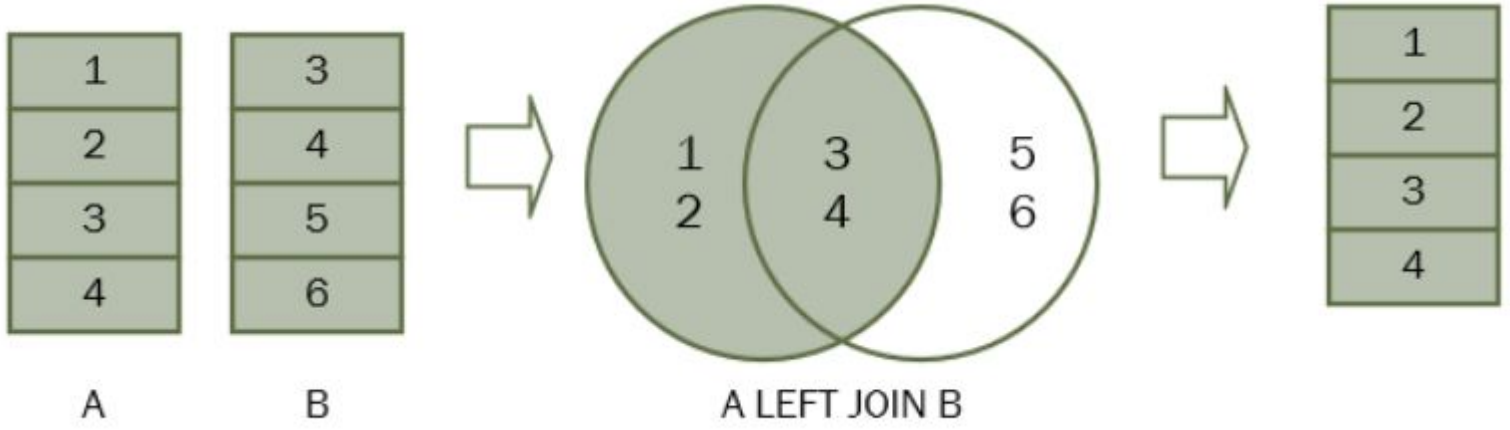
Show the firstname of both sourceId and targetId from the **`user_friend`** table.

```
SELECT fr.sourceId, sc.firstName as Source_FirstName,
fr.targetId, tg.firstName AS Target_FirstName
FROM user_friend fr INNER JOIN user sc
ON fr.sourceId = sc.id
INNER JOIN user tg
ON fr.targetId = tg.id;
```

sourceId	firstName	targetId	firstName
1	Kenn	15	Sandy
2	Jennifer	11	Ardelia
2	Jennifer	12	Tiffany
2	Jennifer	17	Reed
2	Jennifer	21	Tabina
3	Merilee	3	Merilee
3	Merilee	6	Silvie
3	Merilee	10	Lacey
3	Merilee	16	Samuel

SQL LEFT JOIN

The left join returns all rows from the left table and the row that is matching with the right table.

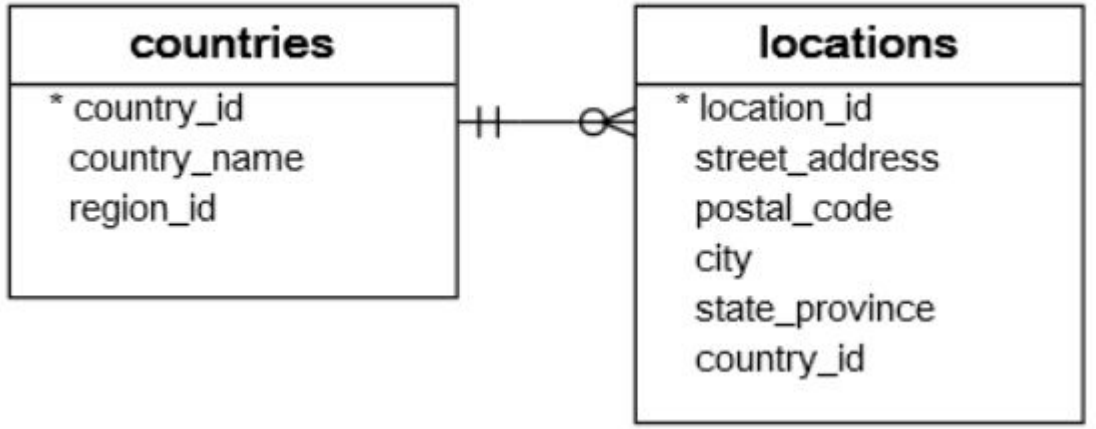


SQL LEFT JOIN Syntax

Suppose the column name of the A & B tables is n, the following statement illustrates the left join clause:

```
SELECT
        A.n
FROM
        A
LEFT JOIN B ON B.n = A.n;
```

SQL LEFT JOIN 2 tables example

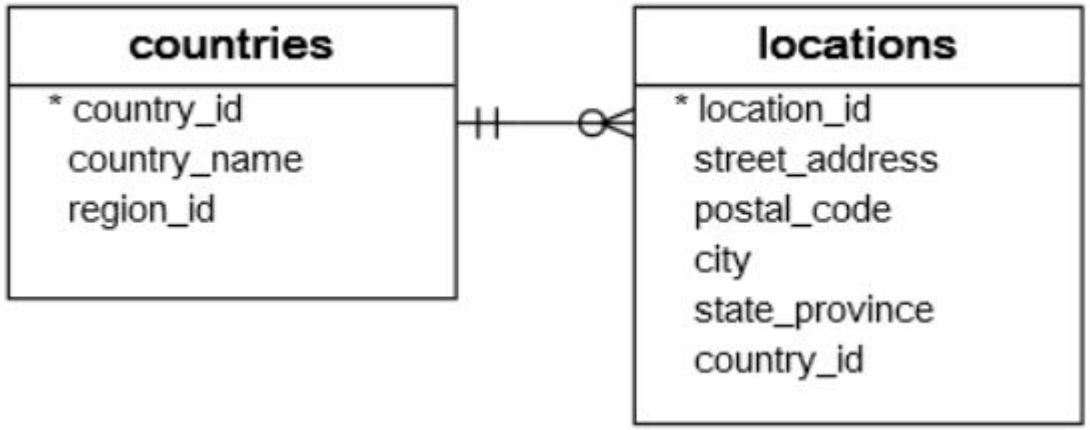



```sql
SELECT
        country_id,
        country_name
FROM
        countries
WHERE
        country_id IN ('US', 'UK', 'CN');
```

Table: countries

	country_id	country_name
	CN	China
	UK	United Kingdom
	US	United States of America

SQL LEFT JOIN 2 tables example




```
SELECT
        country_id,
        street_address,
        city
FROM
        locations
WHERE
        country_id IN ('US', 'UK', 'CN');
```

Table: employees

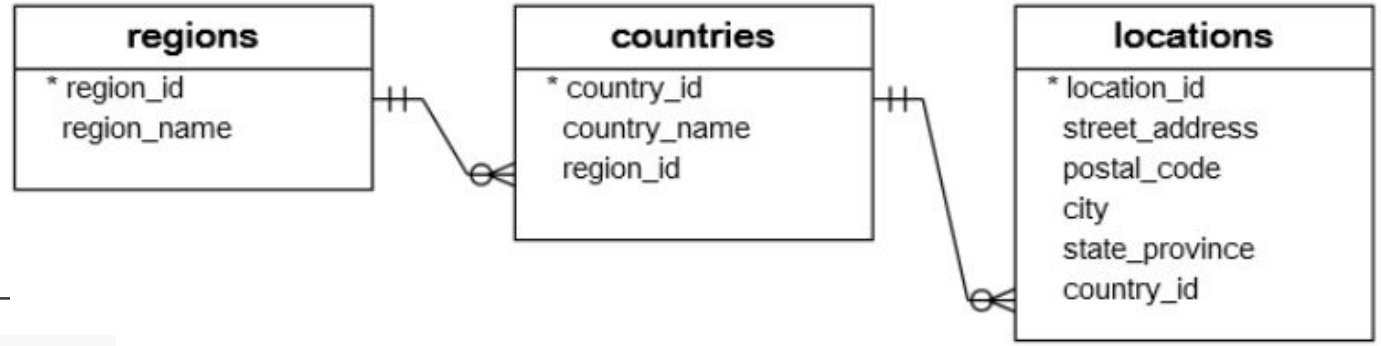
country_id	street_address	city
US	2014 Jabberwocky Rd	Southlake
US	2011 Interiors Blvd	South San Francisc
US	2004 Charade Rd	Seattle

SQL LEFT JOIN 2 tables example (2)

```
SELECT
        c.country_name,
        c.country_id,
        l.country_id,
        l.street_address,
        l.city
FROM
        countries c
LEFT JOIN locations l ON l.country_id = c.country_id
WHERE
        c.country_id IN ('US', 'UK', 'CN')
```

	country_name	country_id	country_id	street_address	city
▶	United States of America	US	US	2014 Jabberwocky Rd	Southlake
	United States of America	US	US	2011 Interiors Blvd	South San Francisco
	United States of America	US	US	2004 Charade Rd	Seattle
	United Kingdom	UK	UK	8204 Arthur St	London
	United Kingdom	UK	UK	Magdalen Centre, The Oxford Science Park	Oxford
	China	CN	NULL	NULL	NULL

SQL LEFT JOIN
3 tables example




```sql
SELECT
        r.region_name,
        c.country_name,
        l.street_address,
        l.city
FROM
        regions r
LEFT JOIN countries c ON c.region_id = r.region_id
LEFT JOIN locations l ON l.country_id = c.country_id
WHERE
        c.country_id IN ('US', 'UK', 'CN');
```

region_name	country_name	street_address	city
Americas	United States of America	2014 Jabberwocky Rd	Southlake
Americas	United States of America	2011 Interiors Blvd	South San Francisco
Americas	United States of America	2004 Charade Rd	Seattle
Europe	United Kingdom	8204 Arthur St	London
Europe	United Kingdom	Magdalen Centre, The Oxford Science Park	Oxford
Asia	China	NULL	NULL

1 2 3

EX. find the country that does not have any locations in the locations table

```
SELECT
        country_name
FROM
        countries c
LEFT JOIN locations l ON l.country_id = c.country_id
WHERE
        l.location_id IS NULL
ORDER BY
        country_name;
```

	country_name
	Argentina
	Australia
	Belgium
	Brazil
	China
	Denmark
	Egypt
	France

LEFT JOIN example (1)

Show all posts with the name who updated the post status (including posts that does not have the info of who had updated).

```
SELECT gp.groupId, gp.userId, gp.message,
    gp.status, gp.statusUpdatedBy, u.firstName
FROM group_post gp LEFT JOIN user u
    ON u.id = gp.statusUpdatedBy;
```

groupId	userId	message	status	statusUpdatedBy	firstName
81	17	Nulla tellus. In sagittis dui vel nisl. Duis ac nibh. Fus...	approved	221	Ernestine
48	107	Sed sagittis.	new	NULL	NULL
97	18	Pellentesque ultrices mattis odio. Donec vitae nisi. N...	new	NULL	NULL
43	184	Donec posuere metus vitae ipsum. Aliquam non ma...	rejected	8	Bernete
88	7	In sagittis dui vel nisl. Duis ac nibh. Fusce lacus pur...	approved	130	Lilli
27	204	Vestibulum rutrum rutrum neque. Aenean auctor gr...	approved	NULL	NULL
82	147	Nulla nisl.	rejected	54	Doralynn
99	177	Ut at dolor quis odio consequat varius. Integer ac le...	rejected	NULL	NULL

LEFT JOIN example (2)

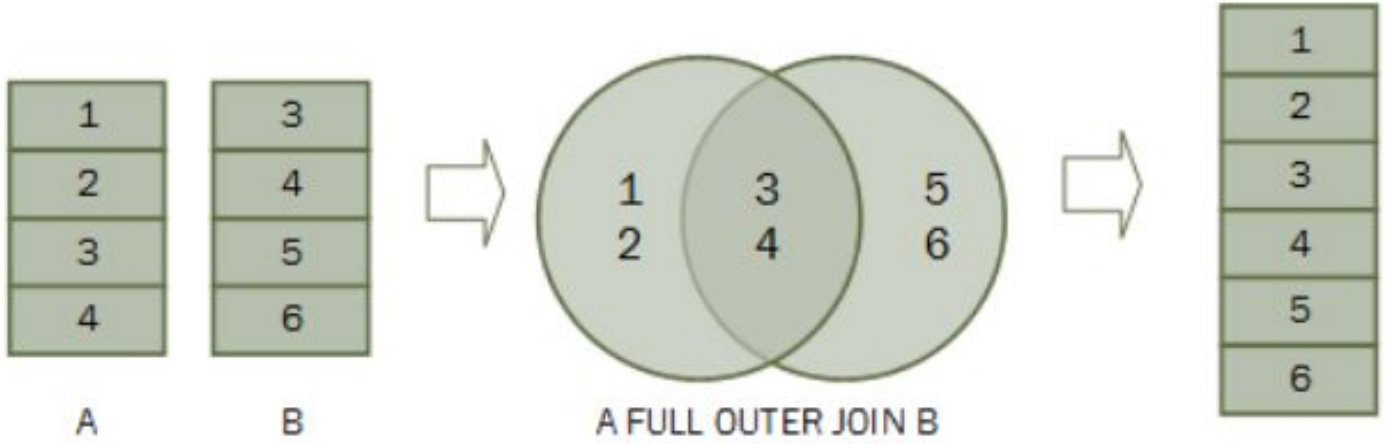
Show the id and the title of group and all posts created on each group (include some group that have no post yet).

```
SELECT gp.groupId, g.title, gp.userId, gp.message
FROM social.group g LEFT JOIN group_post gp
    ON g.id = gp.groupId;
```

id	title	userId	message
100	Flowdesk	3	Nullam molestie nibh in lectus. Pellentesque at nulla. Suspe...
100	Flowdesk	276	Pellentesque eget nunc.
100	Flowdesk	166	Donec semper sapien a libero. Nam dui.
101	AIT Market Place	NULL	NULL

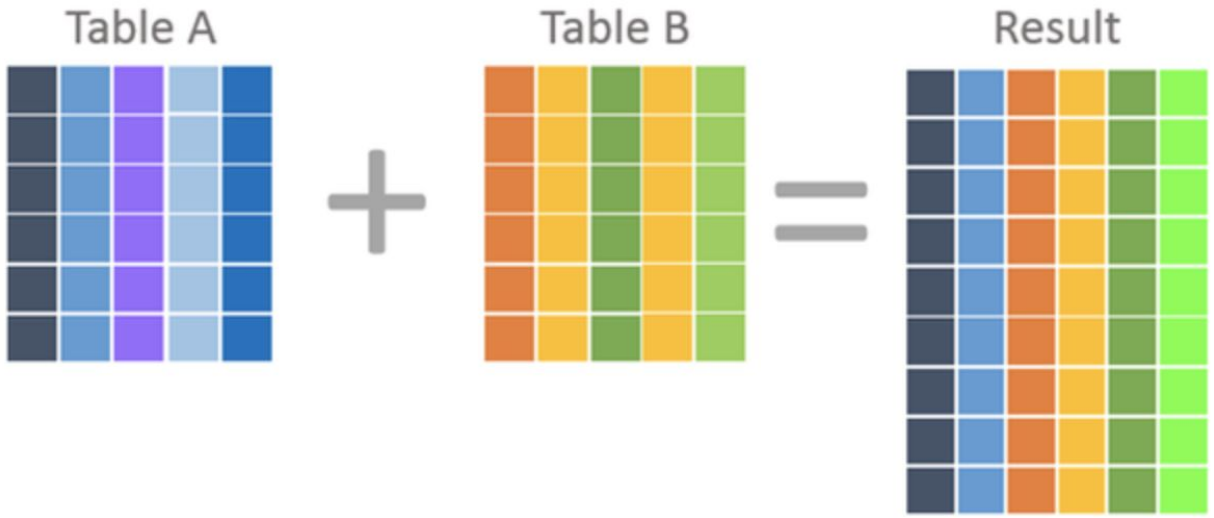
SQL FULL OUTER JOIN

The full outer join includes all rows from the joined tables **whether or not the other table has the matching row**.

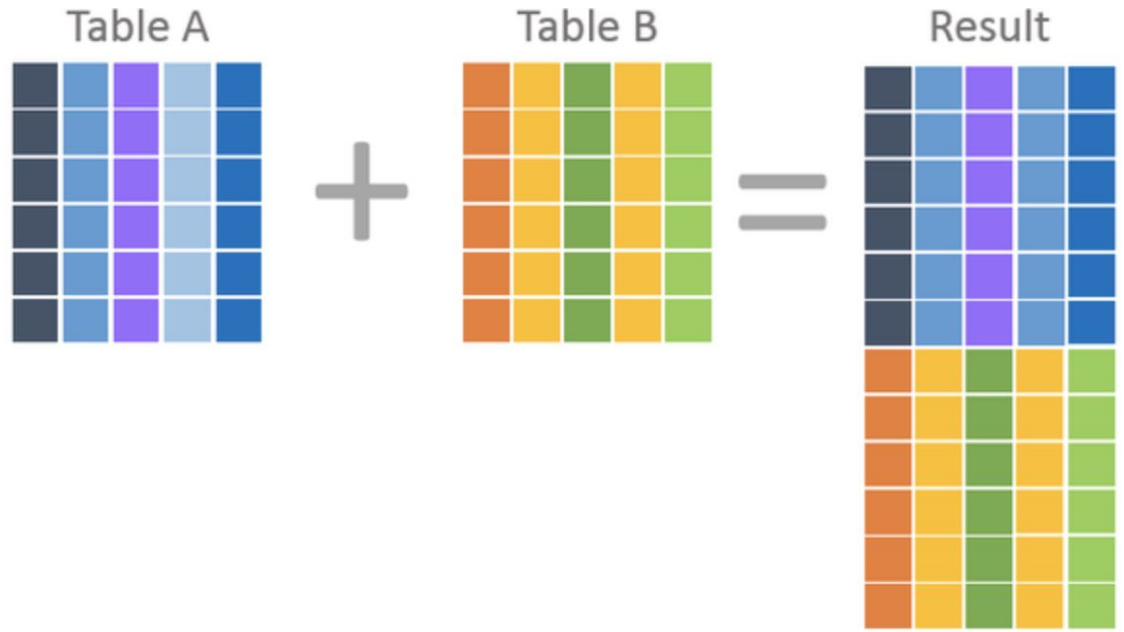


Wait….

So what is the difference between Full Outer Join and Union?



Full Outer Join



Union

SQL FULL OUTER JOIN Syntax

Suppose the column name of the A & B tables is n, the following statement illustrates the full outer join clause:

```
SELECT * FROM t1
   LEFT JOIN t2 ON t1.id = t2.id
UNION
SELECT * FROM t1
   RIGHT JOIN t2 ON t1.id = t2.id
```

FULL OUTER JOIN in MySQL

Show the full name of the user and the group that they are associated to.

```
SELECT CONCAT(u.firstName, ' ', u.lastName) AS USERS, gp.id AS GroupID

FROM user u LEFT JOIN group_member gp ON u.id = gp.userId

UNION

SELECT CONCAT(u.firstName, ' ', u.lastName) AS USERS, gp.id AS GroupID

FROM group_member gp RIGHT JOIN user u ON gp.userId = u.id;
```

USERS	GroupID
Kenn Naris	528
Kenn Naris	688
Kenn Naris	967
Kenn Naris	1006
Jennifer Leehane	49
Jennifer Leehane	285
Jennifer Leehane	506
Jennifer Leehane	963
Merilee Cornely	391
Merilee Cornely	661

In class exercise (1)

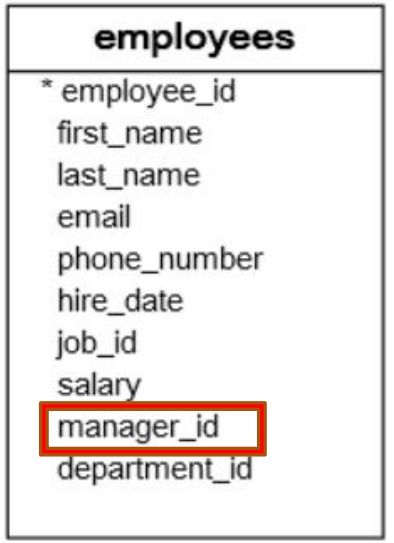
1. Show full name of users and their contact information who posted to the wall of the owner whose ID is 216.

SQL SELF JOIN

We join a table to itself to evaluate the rows with other rows in the same table. To perform the self-join, we use either an inner join or left join clause.

```
SELECT
        column1,
        column2,
        column3,
        ...
FROM
        table1 A
INNER JOIN table1 B ON B.column1 = A.column2;
```

SQL SELF JOIN example




```
SELECT
    e.first_name || ' ' || e.last_name AS employee,
    m.first_name || ' ' || m.last_name AS manager
FROM
    employees e
      INNER JOIN
    employees m ON m.employee_id = e.manager_id
ORDER BY manager;
```

	employee	manager
▶	Bruce Ernst	Alexander Hunold
	David Austin	Alexander Hunold
	Valli Pataballa	Alexander Hunold
	Diana Lorentz	Alexander Hunold
	Alexander Khoo	Den Raphaely
	Shelli Baida	Den Raphaely
	Sigal Tobias	Den Raphaely
	Guy Himuro	Den Raphaely
	Karen Colmenares	Den Raphaely

SQL SELF JOIN example (2)

employees

- * employee_id
- first_name
- last_name
- email
- phone_number
- hire_date
- job_id
- salary
- manager_id
- department_id

```
SELECT
    e.first_name || ' ' || e.last_name AS employee,
    m.first_name || ' ' || m.last_name AS manager
FROM
    employees e
        LEFT JOIN
    employees m ON m.employee_id = e.manager_id
ORDER BY manager;
```

employee	manager
Steven King	NULL
Bruce Ernst	Alexander Hunold
David Austin	Alexander Hunold
Valli Pataballa	Alexander Hunold
Diana Lorentz	Alexander Hunold
Alexander Khoo	Den Raphaely
Shelli Baida	Den Raphaely
Sigal Tobias	Den Raphaely
Guy Himuro	Den Raphaely

Subquery

Many complex queries use sub querying technique to obtain the desired data to support complicated operations or data requirements. Some techniques to achieve subqueries are:

1. Subquery in **`WHERE`** Clause
2. Subquery as a column

Subquery in WHERE clause (1)

- Using a subquery to return some value to be the criteria of the query

Example 1: Show firstname, lastname and email columns of friend requests sent by the user with id - 99.

```
SELECT u.firstname, u.lastname, u.email
FROM social.user u
WHERE u.id IN (
    SELECT targetId FROM user_friend WHERE sourceId = 99
);
```

firstname	lastname	email
Ardelia	Madner	amadnera@adobe.com
Erwin	Kitchen	ekitchenh@cdbaby.com

Subquery in WHERE clause (2)

Example 2: Find users who have posted in the group id 15 but are not members of the group, showing the fullname (concatenating firstName, middleName and lastName), their mobiles and their emails.

```
SELECT u.Id, CONCAT(u.firstName,   ' ', COALESCE(u.middleName, ' '), u.lastName)
    AS PosterName, u.mobile, u.email
FROM user u
WHERE u.Id IN (
                SELECT userId FROM group_post WHERE groupId = 15 and userId
                NOT IN (
                    SELECT userId FROM group_member WHERE groupId = 15
                )
);
```

Id	PosterName	mobile	email
28	Lucho CordsenMacVean	6435876803	lmacveanr@ustream.tv
30	Marena Metzing	7355823618	mmetzingt@blogs.com
48	Ferdinande Pitkaithly	3763050049	fpitkaithly1b@1688.com
139	Juliann Roset	1851729505	jroset3u@census.gov
155	Florina Maciaszek	5538910719	fmaciaszek4a@hibu.com
223	Rebekkah HandasydeNolton	6998526114	rnolton66@mediafire.com
263	Arda Fillan	3428715432	afillan7a@mayoclinic.com

Subquery as a column

In the expression list of SELECT part, we can specify values from a subquery to a column data.

Example: List the number of users in the group separated by gender as each column.
(**`id | title | maleMember | femaleMember | unspecifiedMember`**)

```
SELECT g.id, g.title,
(SELECT COUNT(gm.id) FROM social.group_member gm
        INNER JOIN social.user u ON gm.userId = u.id
    WHERE gm.groupId = g.id AND u.gender = 'm') AS maleMember,
(SELECT COUNT(gm.id) FROM social.group_member gm
        INNER JOIN social.user u ON gm.userId = u.id
    WHERE gm.groupId = g.id AND u.gender = 'f') AS femaleMember,
(SELECT COUNT(gm.id) FROM social.group_member gm
        INNER JOIN social.user u ON gm.userId = u.id
    WHERE gm.groupId = g.id AND u.gender IS NULL) AS unspecifiedMember
FROM social.group g;
```

id	title	maleMember	femaleMember	unspecifiedMember
1	Hatity	3	4	5
2	Stim	4	3	5
3	Gembucket	4	3	3
4	Cardify	2	3	7
5	Treeflex	2	1	5
6	Subin	3	1	3
7	Lotstring	1	6	5
8	Overhold	5	2	3
9	Stim	3	4	3

Subquery in other operations

This example shows how to delete rows which satisfy the criteria specified in the subquery.

Example: Delete all post in the group id=20 created by users who are not the group members.

```
DELETE FROM group_post p
WHERE p.userId NOT IN (
    SELECT userId FROM group_member gm WHERE gm.groupId = p.groupId
) AND p.groupId = 20;
```

In-class exercise (2)

1. Using subquery, show users who never post in public including their full name, username and contact informations.

References

- SQL Tutorial: https://www.sqltutorial.org/
- Sample Datasets: https://www.sqltutorial.org/sql-sample-database/


Thank you.