

MySQL: Relational Algebra

Tushar B. Kute,
<http://tusharkute.com>



Relational Algebra

- RELATIONAL ALGEBRA is a widely used procedural query language.
- It collects instances of relations as input and gives occurrences of relations as output.
- It uses various operations to perform this action. SQL Relational algebra query operations are performed recursively on a relation.
- The output of these operations is a new relation, which might be formed from one or more input relations.

Relational Algebra

- Unary Relational Operations
 - SELECT (symbol: σ)
 - PROJECT (symbol: π)
 - RENAME (symbol: ρ)
- Relational Algebra Operations From Set Theory
 - UNION (\cup)
 - INTERSECTION (\cap),
 - DIFFERENCE ($-$)
 - CARTESIAN PRODUCT (\times)
- Binary Relational Operations
 - JOIN
 - DIVISION

SELECT (σ)

- The SELECT operation is used for selecting a subset of the tuples according to a given selection condition. Sigma(σ) Symbol denotes it.
- It is used as an expression to choose tuples which meet the selection condition. Select operator selects tuples that satisfy a given predicate.

$\sigma_p(r)$

σ is the predicate

r stands for relation which is the name of the table

p is propositional logic

— Example 1: $\sigma_{\text{topic} = \text{"Database"}} (\text{Lectures})$

SELECT (σ)

- Example 2

$\sigma_{\text{topic} = \text{"Database"} \text{ and } \text{author} = \text{"Tushar"}}(\text{Lectures})$

Output – Selects tuples from Tutorials where the topic is 'Database' and 'author' is guru99.

- Example 3

$\sigma_{\text{sales} > 50000}(\text{Customers})$

Output – Selects tuples from Customers where sales is greater than 50000

Projection (π)

- The projection eliminates all attributes of the input relation but those mentioned in the projection list.
- The projection method defines a relation that contains a vertical subset of Relation.
- This helps to extract the values of specified attributes to eliminates duplicate values. (π) symbol is used to choose attributes from a relation.
- This operator helps you to keep specific columns from a relation and discards the other columns.

Projection (π) Example

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Projection (π) Example

- Here, the projection of CustomerName and status will give

$\pi_{\text{CustomerName, Status}}(\text{Customers})$

CustomerName	Status
Google	Active
Amazon	Active
Apple	Inactive
Alibaba	Active

Rename (ρ)

- Rename is a unary operation used for renaming attributes of a relation.
- $\rho(a/b)R$ will rename the attribute 'b' of relation by 'a'.

Union operation (\cup)

- UNION is symbolized by \cup symbol. It includes all tuples that are in tables A or in B. It also eliminates duplicate tuples. So, set A UNION set B would be expressed as:

The result $\leftarrow A \cup B$

- For a union operation to be valid, the following conditions must hold –
 - R and S must be the same number of attributes.
 - Attribute domains need to be compatible.
 - Duplicate tuples should be automatically removed.

Union operation (\cup)

Table A		Table B	
column 1	column 2	column 1	column 2
1	1	1	1
1	2	1	3

Table A \cup B	
column 1	column 2
1	1
1	2
1	3

Set Difference (-)

- – Symbol denotes it. The result of $A - B$, is a relation which includes all tuples that are in A but not in B.
 - The attribute name of A has to match with the attribute name in B.
 - The two-operand relations A and B should be either compatible or Union compatible.
 - It should be defined relation consisting of the tuples that are in relation A, but not in B.

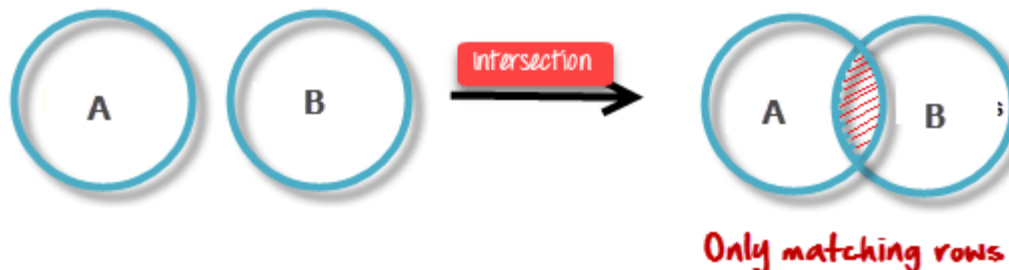
Set Difference (-)

- Example:
A-B

Table A – B	
column 1	column 2
1	2

Intersection

- An intersection is defined by the symbol \cap
 $A \cap B$
- Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.



Intersection

- Example:

$A \cap B$

Table $A \cap B$	
column 1	column 2
1	1

Cartesian Product(X)

- Cartesian Product in DBMS is an operation used to merge columns from two relations. Generally, a cartesian product is never a meaningful operation when it performs alone.
- However, it becomes meaningful when it is followed by other operations. It is also called Cross Product or Cross Join.
- Example –

$$\sigma_{\text{column 2} = '1'} (A \times B)$$

- Output – The above example shows all rows from relation A and B whose column 2 has value 1

Cartesian Product(X)

- Example – Cartesian product

$$\sigma_{\text{column 2} = '1'} (A \times B)$$

- Output – The above example shows all rows from relation A and B whose column 2 has value 1

$$\sigma_{\text{column 2} = '1'} (A \times B)$$

column 1	column 2
1	1
1	1

Join Operations

- Join operation is essentially a cartesian product followed by a selection criterion.
- Join operation denoted by \bowtie .
- JOIN operation also allows joining variously related tuples from different relations.

Join Operations: Types

- Various forms of join operation are:
- Inner Joins:
 - Theta join
 - EQUI join
 - Natural join
- Outer join:
 - Left Outer Join
 - Right Outer Join
 - Full Outer Join

Inner Join

- In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded. Let's study various types of Inner Joins:
- Theta Join:
 - The general case of JOIN operation is called a Theta join. It is denoted by symbol θ
 - Example
$$A \bowtie_{\theta} B$$
 - Theta join can use any conditions in the selection criteria.

Inner Join

- Theta join can use any conditions in the selection criteria.
 - For example:

$A \bowtie_{A.column\ 2 > B.column\ 2} (B)$

A $\bowtie_{A.column\ 2 > B.column\ 2} (B)$	
column 1	column 2
1	2

EQUI join

- When a theta join uses only equivalence condition, it becomes a equi join.
- For example:

$A \bowtie_{A.\text{column 2} = B.\text{column 2}} (B)$

A \bowtie A.column 2 = B.column 2 (B)	
column 1	column 2
1	1

NATURAL JOIN (\bowtie)

- Natural join can only be performed if there is a common attribute (column) between the relations.
- The name and type of the attribute must be same.

NATURAL JOIN (\bowtie)

C	
Num	Square
2	4
3	9

D	
Num	Cube
2	8
3	27

NATURAL JOIN (\bowtie)

C \bowtie D

C \bowtie D		
Num	Square	Cube
2	4	8
3	9	27

OUTER JOIN

- In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.
- Left Outer Join(A B)
 - In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation, then the attributes of right relation in the join result are filled with null values.



All rows from Left Table.


Left OUTER JOIN

A	
Num	Square
2	4
3	9
4	16

B	
Num	Cube
2	8
3	18
5	75

Left OUTER JOIN

A  B

A  B		
Num	Square	Cube
2	4	8
3	9	18
4	16	–

Right Outer Join

- In the right outer join, operation allows keeping all tuple in the right relation.
- However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.



Right Outer Join

A \bowtie B

A \bowtie B		
Num	Cube	Square
2	8	4
3	18	9
5	75	–

Full Outer Join

- In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

$A \bowtie B$

$A \bowtie B$		
Num	Cube	Square
2	4	8
3	9	18
4	16	–
5	–	75

Summary

Select(σ)

The SELECT operation is used for selecting a subset of the tuples according to a given selection condition

Projection(π)

The projection eliminates all attributes of the input relation but those mentioned in the projection list.

Union Operation(\cup)

UNION is symbolized by symbol. It includes all tuples that are in tables A or in B.

Set Difference($-$)

– Symbol denotes it. The result of $A - B$, is a relation which includes all tuples that are in A but not in B.

Intersection(\cap)

Intersection defines a relation consisting of a set of all tuple that are in both A and B.

Summary

Cartesian Product(X)

Cartesian operation is helpful to merge columns from two relations.

Inner Join

Inner join, includes only those tuples that satisfy the matching criteria.

Theta Join(θ)

The general case of JOIN operation is called a Theta join. It is denoted by symbol θ .

EQUI Join

When a theta join uses only equivalence condition, it becomes a equi join.

Natural Join(\bowtie)

Natural join can only be performed if there is a common attribute (column) between the relations.

Summary

Outer Join

In an outer join, along with tuples that satisfy the matching criteria.

Left Outer Join(\Join)

In the left outer join, operation allows keeping all tuple in the left relation.

Right Outer join(\Join)

In the right outer join, operation allows keeping all tuple in the right relation.

Full Outer Join(\Join)

In a full outer join, all tuples from both relations are included in the result irrespective of the matching condition.

MySQL JOINS

- MySQL JOINS are used with SELECT statement. It is used to retrieve data from multiple tables. It is performed whenever you need to fetch records from two or more tables.
- There are three types of MySQL joins:
 - MySQL INNER JOIN (or sometimes called simple join)
 - MySQL LEFT OUTER JOIN (or sometimes called LEFT JOIN)
 - MySQL RIGHT OUTER JOIN (or sometimes called RIGHT JOIN)

MySQL Inner JOIN (Simple Join)

- The MySQL INNER JOIN is used to return all rows from multiple tables where the join condition is satisfied. It is the most common type of join.
- Syntax:
 SELECT columns
 FROM table1
 INNER JOIN table2
 ON table1.column = table2.column;

MySQL Inner JOIN (Simple Join)

- Let's take an example:
- Consider two tables "officers" and "students", having the following data.

```
SELECT officers.officer_name, officers.address,  
students.course_name  
FROM officers  
INNER JOIN students  
ON officers.officer_id = students.student_id;
```

MySQL Left Outer Join

- The LEFT OUTER JOIN returns all rows from the left hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.
- Syntax:
 SELECT columns
 FROM table1
 LEFT [OUTER] JOIN table2
 ON table1.column = table2.column;

MySQL Left Outer Join

- Let's take an example:
- Consider two tables "officers" and "students", having the following data.

```
SELECT officers.officer_name, officers.address,  
       students.course_name  
FROM officers  
LEFT JOIN students  
ON officers.officer_id = students.student_id;
```

MySQL Right Outer Join

- The MySQL Right Outer Join returns all rows from the RIGHT-hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.
- Syntax:
 SELECT columns
 FROM table1
 RIGHT [OUTER] JOIN table2
 ON table1.column = table2.column;

MySQL Right Outer Join

- Let's take an example:
- Consider two tables "officers" and "students", having the following data.

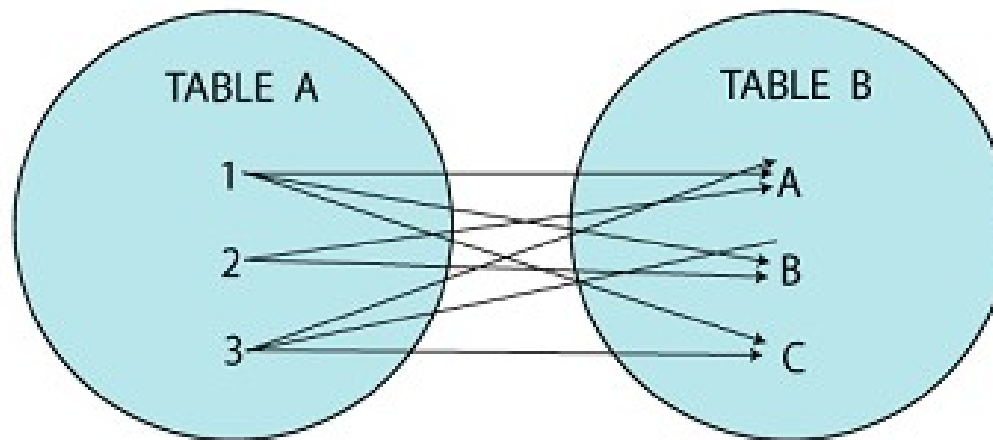
```
SELECT officers.officer_name, officers.address,  
       students.course_name, students.student_name  
FROM officers  
RIGHT JOIN students  
ON officers.officer_id = students.student_id;
```

MySQL CROSS JOIN

- MySQL CROSS JOIN is used to combine all possibilities of the two or more tables and returns the result that contains every row from all contributing tables.
- The CROSS JOIN is also known as CARTESIAN JOIN, which provides the Cartesian product of all associated tables.
- The Cartesian product can be explained as all rows present in the first table multiplied by all rows present in the second table.
- It is similar to the Inner Join, where the join condition is not available with this clause.

MySQL CROSS JOIN

CROSS JOIN



MySQL CROSS JOIN

- The CROSS JOIN keyword is always used with the SELECT statement and must be written after the FROM clause.
- The following syntax fetches all records from both joining tables:

```
SELECT column-lists  
FROM table1  
CROSS JOIN table2;
```

- In the above syntax, the column-lists is the name of the column or field that you want to return and table1 and table2 is the table name from which you fetch the records.

MySQL EquiJoin

- The process is called joining when we combine two or more tables based on some common columns and a join condition.
- An equijoin is an operation that combines multiple tables based on equality or matching column values in the associated tables.
- We can use the equal sign (=) comparison operator to refer to equality in the WHERE clause.
- This joining operation returns the same result when we use the JOIN keyword with the ON clause and then specifying the column names and their associated tables.

MySQL EquiJoin

```
SELECT column_name (s)
FROM table_name1, table_name2, ..., table_nameN
WHERE table_name1.column_name =
table_name2.column_name;
```

OR

```
SELECT (column_list | *)
FROM table_name1
JOIN table_name2
ON table_name1.column_name = table_name2.column_name;
```

MySQL EquiJoin

```
mysql> select * from customer;
```

id	customer_name	account	email
1	Stephen	1030	stephen@javatpoint.com
2	Jenifer	2035	jenifer@javatpoint.com
3	Mathew	5564	mathew@javatpoint.com
4	Smith	4534	smith@javatpoint.com
5	david	7648	david@javatpoint.com

```
5 rows in set (0.00 sec)
```

```
mysql> select * from balance;
```

id	account_num	balance
1	1030	50000.00
2	2035	230000.00
3	5564	125000.00
4	4534	80000.00
5	7648	45000.00

```
5 rows in set (0.00 sec)
```

MySQL EquiJoin

```
mysql> SELECT cust. customer_name, bal.balance  
FROM customer AS cust, balance AS bal  
WHERE cust.account = bal.account_num;
```

```
mysql> SELECT cust. customer_name, bal.balance  
-> FROM customer AS cust, balance AS bal  
-> WHERE cust.account = bal.account_num;
```

customer_name	balance
Stephen	50000.00
Jenifer	230000.00
Mathew	125000.00
Smith	80000.00
david	45000.00

```
5 rows in set (0.00 sec)
```


Union

- MySQL Union clause allows us to combine two or more relations using multiple SELECT queries into a single result set. By default, it has a feature to remove the duplicate rows from the result set.
- Union clause in MySQL must follow the rules given below:
 - The order and number of the columns must be the same in all tables.
 - The data type must be compatible with the corresponding positions of each select query.
 - The column name in the SELECT queries should be in the same order.

Union

```
SELECT column_name(s) FROM table_name1  
UNION  
SELECT column_name(s) FROM table_name2;
```

Example:

```
SELECT stud_name, subject FROM student1  
UNION  
SELECT stud_name, subject FROM student2;
```

MySQL Copy/Clone/Duplicate Table

- MySQL copy or clone table is a feature that allows us to create a duplicate table of an existing table, including the table structure, indexes, constraints, default values, etc.
- Copying data of an existing table into a new table is very useful in a situation like backing up data in table failure.
- It is also advantageous when we need to test or perform something without affecting the original table, for example, replicating the production data for testing.

MySQL Copy/Clone/Duplicate Table

- We can copy an existing table to a new table using the CREATE TABLE and SELECT statement, as shown below:

```
CREATE TABLE new_table_name  
SELECT column1, column2, column3  
FROM existing_table_name;
```

MySQL Copy/Clone/Duplicate Table

```
CREATE TABLE IF NOT EXISTS new_table_name  
SELECT column1, column2, column3  
FROM existing_table_name  
WHERE condition;
```

MySQL Copy/Clone/Duplicate Table

- `CREATE TABLE IF NOT EXISTS new_table_name LIKE existing_table_name;`
- `INSERT new_table_name SELECT * FROM existing_table_name;`

Thank you

This presentation is created using LibreOffice Impress 7.4.1.2, can be used freely as per GNU General Public License



@mitu_skillologies



@mITuSkillologies



@mitu_group



@mitu-skillologies



@MITUSkillologies

kaggle

@mituskillologies

Web Resources

<https://mitu.co.in>

<http://tusharkute.com>



@mituskillologies

contact@mitu.co.in
tushar@tusharkute.com