Set Operators

By Rahul Barve

Set Operators

- SQL supports few Set operations which can be performed on the table data.
- These are used to get meaningful results from data stored in the table, under different special conditions.

Set Operators

- Set operators are divided into 4 categories:
  - `UNION`
  - `UNION ALL`
  - `MINUS`
  - `INTERSECT`

Set Operators

- UNION:
  - UNION is used to combine the results of two or more SELECT statements.
  - However it will eliminate duplicate rows from its resultset.
  - In case of union, number of columns and datatype must be same in both the tables, on which UNION operation is being applied.

Set Operators

- UNION ALL:
  - It is very much similar to UNION except it also shows duplicate rows.

Set Operators

- MINUS:
  - The MINUS operation combines results of two SELECT statements and return only those in the final result, which belongs to the first set of the result.

Set Operators

- INTERSECT:
  - INTERSECT operation is used to combine two SELECT statements, but it only returns the records which are common from both SELECT statements. In case of Intersect the number of columns and data type must be same.

Joins

By Rahul Barve

Joins

- SQL JOIN clause is used to combine rows from two or more tables, based upon a common field between them.
- The common field is the FOREIGN KEY that builds an association between the 2 tables.

Joins

- Joins are divided into 4 types:
  - Inner Join
  - Left Outer Join
  - Right Outer Join
  - Full Join

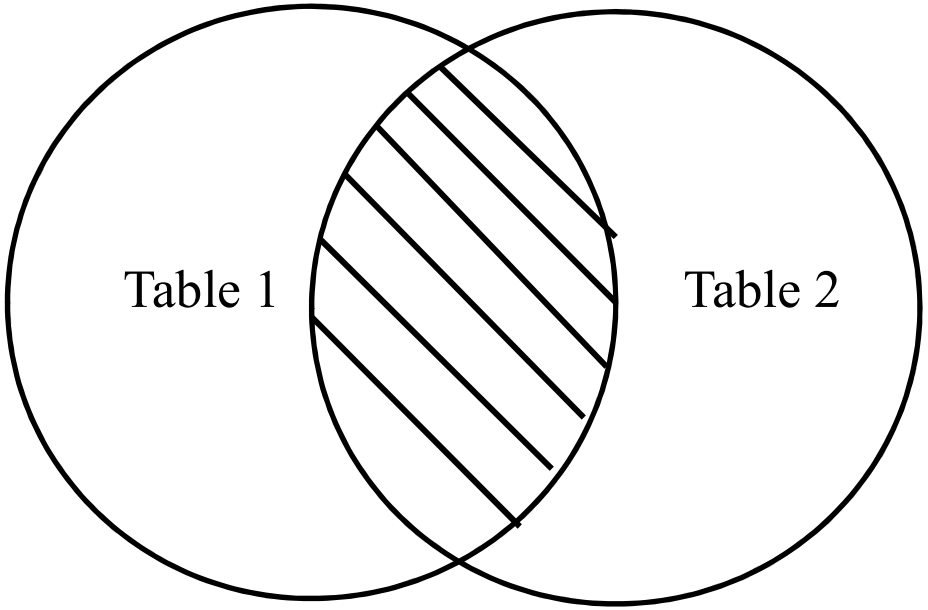
Inner Join

By Rahul Barve

Inner Join

- Returns all rows when there is at least one match in both tables.

Inner Join



Inner Join

- Syntax:

```
SELECT <column_name(s)> FROM
<table1>
INNER JOIN
                <table2>
ON <table1.column_name>
=<table2.column_name>;
```

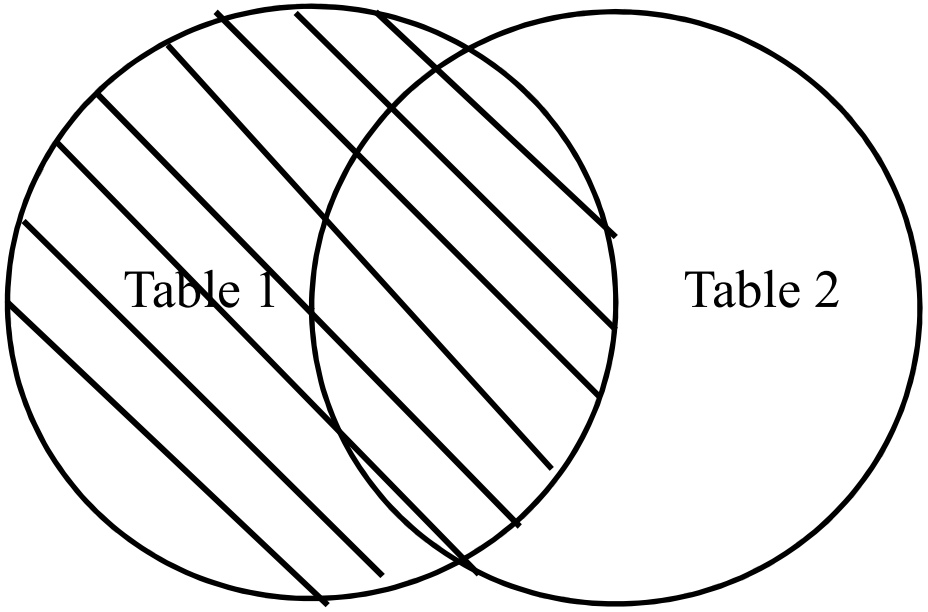
Left Outer Join

By Rahul Barve

Left Outer Join

- Returns all rows from the left table (table1), with the matching rows in the right table (table2).
- The result is NULL in the right side when there is no match.

Left Outer Join



Left Outer Join

- Syntax:

```
SELECT <column_name(s)> FROM
<table1>
LEFT JOIN
                    <table2>
ON <table1.column_name>
=<table2.column_name>;
```

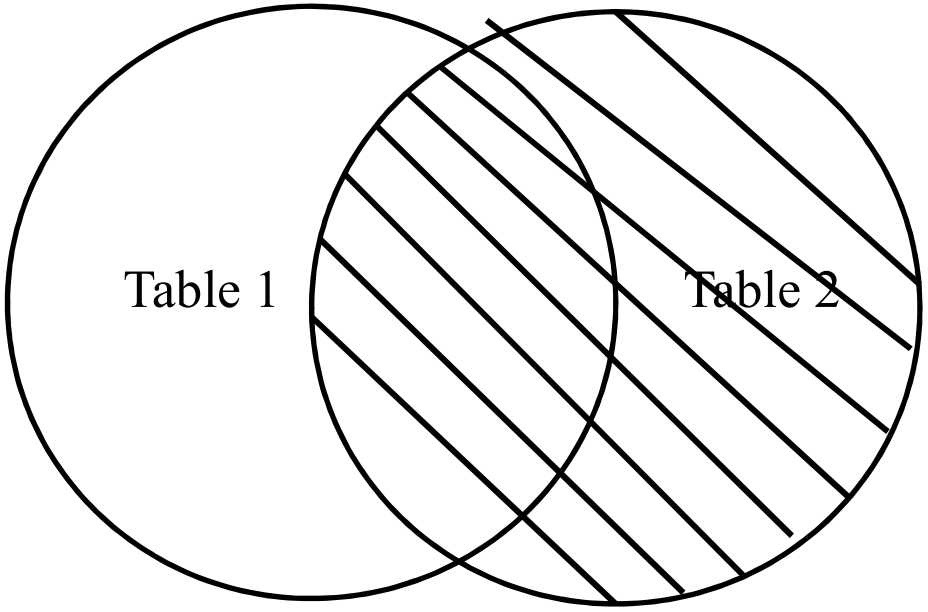
Right Outer Join

By Rahul Barve

Right Outer Join

- Returns all rows from the right table (table2), with the matching rows in the left table (table1).
- The result is NULL in the left side when there is no match.

Right Outer Join



Right Outer Join

- Syntax:

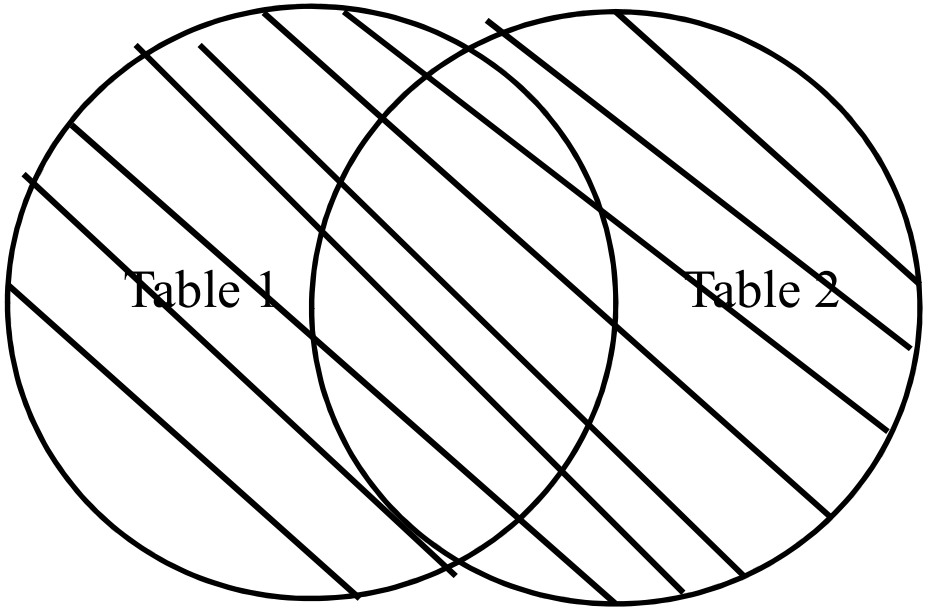
```
SELECT <column_name(s)> FROM
<table1>
RIGHT JOIN
                    <table2>
ON <table1.column_name>
=<table2.column_name>;
```

Full Join

Full Join

- Returns all rows from the left table (table1) and from the right table (table2).
- Combines the result of both LEFT and RIGHT joins.

Full Join



Full Join

- Syntax:

```
SELECT <column_name(s)> FROM
<table1>
FULL JOIN
                    <table2>
ON <table1.column_name>
=<table2.column_name>;
```

Creating Tables

By Rahul Barve

Creating Tables

- Tables can be created using `CREATE TABLE` command.
- Syntax:

```
create table <table-name> (
    <column-name> <datatype>,
    <column-name> <datatype>,
    <column-name> <datatype>
);
```

Altering Tables

By Rahul Barve

Altering Tables

- Existing table's structure can be modified using `ALTER TABLE` command.
- Tables can be altered for:
  - Modifying column's dimension
  - Adding a new column
  - Dropping an existing column
  - Adding a new constraint
  - Dropping an existing constraint

Altering Tables

- Modifying column's dimension:
  - Syntax

```
alter table <table-name>
modify <column-name>
<new-dimension>
```

- Reducing the precision is not possible unless the column is empty.

Altering Tables

- Adding a new column:
  - Syntax

```
alter table <table-name>
add <column-name>
<dimension>
```

Altering Tables

- Dropping an existing column:
  - Syntax

```
alter table <table-name>
drop column <column-name>
```

- It's not possible to drop the column if it's the only column existing in the table.

Dropping Tables

Dropping Tables

- Existing table can be dropped by using `DROP TABLE` command.
- Syntax:

```
drop table <table-name>
```

Truncating Tables

By Rahul Barve

Truncating Tables

- Existing table can be truncated by using `TRUNCATE TABLE` command.
- Deletes all the records from the table.
- Syntax:

```
truncate table <table-name>
```

Inserting Records

By Rahul Barve

Inserting Records

- Once a table is created, we can add records into the table by using `INSERT INTO` command.
- Syntax:

```
insert into <table-name> values
(value1, value2, …);
```

- Values for column of type `varchar2` must be specified using single quotation marks.

Inserting Records

- In order to insert multiple records one after the other, it's possible to use substitution parameter '&'.
- It simplifies record insertion.
- Syntax:

```
insert into <table-name> values
(&<var-name>, &<var-name>, …);
```

Inserting Records

- It's also possible to change the sequence of the columns while inserting records.
- Syntax:

```
insert into <table-name>
(<column-name1>, <column-name2>, …)
values (<value1>, <value2>, …);
```

Updating Records

By Rahul Barve

Updating Records

- Existing records can be updated by using `UPDATE` command.
- Syntax:

```
update <table-name>
set <column-name>=<value>,
<column-name>=<value>…
[WHERE <condition>]
```

Deleting Records

By Rahul Barve

Deleting Records

- Existing records can be deleted by using `DELETE` command.
- Syntax:

```
delete from <table-name>
[WHERE <condition>]
```