1. Cartesian

- Gives all the possibe pair of tuples from 2 relations.

2. Join

- Join is combination of cartesian product and selection process based on some conditions.
- It creates a pair of tuple only, if the given join condition is satisfied.

Types of Join

1. Inner Join
2. Outer Join

2.1. Inner Join

- Inner join includes only those rows from both table for which the join condition is true.
- Rest of the rows where condition is false, those rows are not included in resulting table

Types of Inner Join

- Inner join can be divided into 3 class

    - Theta(θ) Join
    - Equi Join
    - Natural Join

2.1.1 Theta (θ) join

- Selects rows from both tables given that it satisfies (θ) condition
- This joins can use all comparision Operators
- Notation: $R1 \bowtie_\theta R2$

2.1.2. Equi Join

- When Theta joins only usages equality operator it is called EQUI join.
- In Equi Join the result is based on the matched data as per specified equality condition.
- Selects all rows from both table taht satisfy the equality condition
- keyword: **INNER JOIN**

```
SELECT column-name-list FROM
table-name1 INNER JOIN table-name2
WHERE table-name1.column-name = table-name2.column-name;
```

2.1.3. Natural Join

- A type of Inner Join.
- Does not use any comparision operator.
- Selects all rows from both tables/relations based on a column having same name and data type on both relations.
- If such column does not exits then join operation is not performed (returns empty table).
- keyword: **NATURAL JOIN**
-
    - Notation: $R1 \bowtie R2$

```
SELECT * FROM
table-name1 NATURAL JOIN table-name2;
```

2.2 Outer Join

- Outer join is opposite of Inner Join as.
- It includes all the tuples / rows from both the relation, even if the the attribute does not matches.
- Matching data are filled based on both table and remaining rows are set as **null**.
- There are 3 types of Outer Join:
    - Left Outer Join
    - Right Outer Join
    - Full Outer Join

2.2.1 Left Outer Join

- A type of outer Join
- Returs a new table after performing
    - Selects all matching rows from both side
    - Selects all remaining rows from left side
    - Remaining columns on right side are set to **null**
    - **null** values are set for the fields belonging to right side
- To specify a condition, we use the **ON** keyword with Outer Join.
- keyword: **LEFT OUTER JOIN**

```
SELECT column-name-list FROM
table-name1 LEFT OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```

2.2.2 Right Outer Join

- A type of outer Join
- Returs a new table after performing
    - Selects all matching rows from both side
    - Selects all remaining rows from right side
    - Remaining columns on left side are set to **null**
    - **null** values are set for the fields belonging to left side (table)
- To specify a condition, we use the **ON** keyword with Outer Join.
- keyword: **RIGHT OUTER JOIN**

```
SELECT column-name-list FROM
table-name1 RIGHT OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```

2.2.3 Full Outer Join

- A type of outer Join
- Returs a new table after performing
    - Selects all matching rows from both side
    - Selects all remaining rows from both side
    - Remaining columns on both side are set to **null**
- To specify a condition, we use the **ON** keyword with Outer Join.
- keyword: **FULL OUTER JOIN**

```
SELECT column-name-list FROM
table-name1 FULL OUTER JOIN table-name2
ON table-name1.column-name = table-name2.column-name;
```