DAY 4 Assessment

Name: Krushnakumar Patle

Email: krishnapatle128@gmail.com

Batch: Data Engineering Batch-1

Topic Join and SubQueries

```sql
INSERT INTO Student (ROLL_NO, NAME, ADDRESS, PHONE, Age) VALUES
    (1, 'HARSH', 'DELHI', '947646383', 18),
    (2, 'PRATIK', 'BIHAR', '894465483', 19),
    (3, 'RIYANKA', 'SILIGURI', '9474383332', 20),
    (4, 'DEEP', 'RAMNAGAR', '9037364632', 18),
    (5, 'SAPTARHI', 'KOLKATA', '87463283743', 19),
    (6, 'DHANRAJ', 'BARABAJAR','789046532', 20),
    (7, 'ROHIT', 'BALURGHAT', 'ALIPUR', 18),
    (8, 'NIRAJ', 'ALIPUR', '7836463722', 19);

select *from Student;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	HARSH	DELHI	947646383	18
2	PRATIK	BIHAR	894465483	19
3	RIYANKA	SILIGURI	9474383332	20
4	DEEP	RAMNAGAR	9037364632	18
5	SAPTARHI	KOLKATA	87463283743	19
6	DHANRAJ	BARABAJAR	789046532	20
7	ROHIT	BALURGHAT	ALIPUR	18
8	NIRAJ	ALIPUR	7836463722	19
NULL	NULL	NULL	NULL	NULL

DAY 4 Assessment

```
INSERT INTO StudentCourse (COURSE_ID, ROLL_NO)
VALUES
    (1, 1),
    (2, 2),
    (6, 3),
    (3, 4),
    (7, 5),
    (4, 6),
    (5, 7),
    (8, 8);
select *from StudentCourse;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

COURSE_ID	ROLL_NO
1	1
2	2
6	3
3	4
7	5
4	6
5	7
8	8
NULL	NULL

1. Inner JOIN:
   The INNER JOIN keyword selects all rows from both the tables as long as the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

```
SELECT StudentCourse.COURSE_ID, Student.NAME, Student.AGE FROM Student
INNER JOIN StudentCourse
ON Student.ROLL_NO = StudentCourse.ROLL_NO;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COURSE_ID	NAME	AGE
1	HARSH	18
2	PRATIK	19
6	RIYANKA	20
3	DEEP	18
7	SAPTARHI	19
4	DHANRAJ	20
5	ROHIT	18
8	NIRAJ	19

2. LEFT JOIN:

This join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN.

```
SELECT Student.NAME,StudentCourse.COURSE_ID
FROM Student
LEFT JOIN StudentCourse
ON StudentCourse.ROLL_NO = Student.ROLL_NO;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

NAME	COURSE_ID
HARSH	1
PRATIK	2
RIYANKA	6
DEEP	3
SAPTARHI	7
DHANRAJ	4
ROHIT	5
NIRAJ	8

3. RIGHT JOIN:

   RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.

```
SELECT Student.NAME,StudentCourse.COURSE_ID
FROM Student
RIGHT JOIN StudentCourse
ON StudentCourse.ROLL_NO = Student.ROLL_NO;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

NAME	COURSE_ID
HARSH	1
PRATIK	2
RIYANKA	6
DEEP	3
SAPTARHI	7
DHANRAJ	4
ROHIT	5
NIRAJ	8

4. FULL JOIN:
   FULL JOIN creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain NULL values.

DAY 4 Assessment

```
SELECT *
FROM Student
JOIN StudentCourse
ON StudentCourse.ROLL_NO = Student.ROLL_NO;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ROLL_NO	NAME	ADDRESS	PHONE	Age	COURSE_ID	ROLL_NO
1	HARSH	DELHI	947646383	18	1	1
2	PRATIK	BIHAR	894465483	19	2	2
3	RIYANKA	SILIGURI	9474383332	20	6	3
4	DEEP	RAMNAGAR	9037364632	18	3	4
5	SAPTARHI	KOLKATA	87463283743	19	7	5
6	DHANRAJ	BARABAJAR	789046532	20	4	6
7	ROHIT	BALURGHAT	ALIPUR	18	5	7
8	NIRAJ	ALIPUR	7836463722	19	8	8

5. NATURAL JOIN:

   A NATURAL JOIN is a type of join in SQL that automatically matches columns with the same name in both tables. It's based on the idea that if two tables have columns with the same name, they are likely to be related. The NATURAL JOIN clause eliminates the need to explicitly specify the join condition.

```
SELECT *
FROM Student
NATURAL JOIN StudentCourse;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ROLL_NO	NAME	ADDRESS	PHONE	Age	COURSE_ID
1	HARSH	DELHI	947646383	18	1
2	PRATIK	BIHAR	894465483	19	2
3	RIYANKA	SILIGURI	9474383332	20	6
4	DEEP	RAMNAGAR	9037364632	18	3
5	SAPTARHI	KOLKATA	87463283743	19	7
6	DHANRAJ	BARABAJAR	789046532	20	4
7	ROHIT	BALURGHAT	ALIPUR	18	5
8	NIRAJ	ALIPUR	7836463722	19	8

6. EQUI JOIN:
   EQUI JOIN creates a JOIN for equality or matching column(s) values of the relative tables. EQUI JOIN also create JOIN by using JOIN with ON and then providing the names of the columns with their relative tables to check equality using equal sign (=).

DAY 4 Assessment

```sql
-- Using WHERE clause
SELECT student.name, student.id, record.class, record.city
FROM student, record
WHERE student.city = record.city;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	id	class	city
Gouri	6	3	Delhi
Megha	4	3	Delhi
Hina	3	3	Delhi
Gouri	6	2	Delhi
Megha	4	2	Delhi
Hina	3	2	Delhi
Gouri	6	2	Delhi
Megha	4	2	Delhi
Hina	3	2	Delhi

```sql
-- Using JOIN ON clause
SELECT student.name, student.id, record.class, record.city
FROM student
JOIN record ON student.city = record.city;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	id	class	city
Gouri	6	3	Delhi
Megha	4	3	Delhi
Hina	3	3	Delhi
Gouri	6	2	Delhi
Megha	4	2	Delhi
Hina	3	2	Delhi
Gouri	6	2	Delhi
Megha	4	2	Delhi
Hina	3	2	Delhi

7. NON EQUI JOIN :
   NON EQUI JOIN performs a JOIN using comparison operator other than equal(=) sign like >, <, >=, <= with conditions.

DAY 4 Assessment

```
SELECT Student.name, Record.id, Record.city
FROM Student, Record
WHERE Student.id < Record.id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Co

name	id	city
Hina	9	Delhi
Hina	10	Delhi
Hina	12	Delhi
Megha	9	Delhi
Megha	10	Delhi
Megha	12	Delhi
Gouri	9	Delhi
Gouri	10	Delhi
Gouri	12	Delhi