

Course code : **CSE2007**  
Course title : **Database Management System**  
Module : **3**  
Topic : **6**

## **SQL - JOINS**

# Objectives

This session will give the knowledge about

- SQL JOINS
- SQL Strings

# Joins

What is meant by Join?

- An SQL JOIN clause combines rows from two or more tables. It creates a set of rows in a temporary table.

How to Join two tables in SQL?

- A JOIN works on two or more tables if they have at least one common field and have a relationship between them.
- JOIN keeps the base tables (structure and data) unchanged.

Syntax:

```
SELECT col1, col2, col3... FROM table_name1, table_name2  
WHERE table_name1.col = table_name2.col;
```

# Joins

There are two types of SQL JOINS -

- EQUI JOIN
- NON EQUI JOIN

SQL EQUI JOIN :

The SQL EQUI JOIN is a simple sql join uses the equal sign(=) as the comparison operator for the condition. It has two types - SQL Outer join and SQL Inner join.

SQL NON EQUI JOIN :

The SQL NON EQUI JOIN is a join uses comparison operator other than the equal sign like >, <, >=, <= with the condition.

# Inner Joins (Equi Join)

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

```
SQL> SELECT ID, NAME, AMOUNT, DATE
      FROM CUSTOMERS
      INNER JOIN ORDERS
      ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

ID	NAME	AMOUNT	DATE
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
2	Khilan	1560	2009-11-20 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00

# Left Joins

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

```
SQL> SELECT ID, NAME, AMOUNT, DATE
FROM CUSTOMERS
LEFT JOIN ORDERS
ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

ID	NAME	AMOUNT	DATE
1	Ramesh	NULL	NULL
2	Khilan	1560	2009-11-20 00:00:00
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00
5	Hardik	NULL	NULL
6	Komal	NULL	NULL
7	Muffy	NULL	NULL

# Right Joins

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

```
SQL> SELECT ID, NAME, AMOUNT, DATE
      FROM CUSTOMERS
      RIGHT JOIN ORDERS
      ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

ID	NAME	AMOUNT	DATE
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
2	Khilan	1560	2009-11-20 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00

# Full Joins

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

```
SQL> SELECT ID, NAME, AMOUNT, DATE
      FROM CUSTOMERS
      FULL JOIN ORDERS
      ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

ID	NAME	AMOUNT	DATE
1	Ramesh	NULL	NULL
2	Khilan	1560	2009-11-20 00:00:00
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00
5	Hardik	NULL	NULL
6	Komal	NULL	NULL
7	Muffy	NULL	NULL
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
2	Khilan	1560	2009-11-20 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00



# Self Joins

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

```
SQL> SELECT a.ID, b.NAME, a.SALARY
FROM CUSTOMERS a, CUSTOMERS b
WHERE a.SALARY < b.SALARY;
```

ID	NAME	SALARY
2	Ramesh	1500.00
2	kaushik	1500.00
1	Chaitali	2000.00
2	Chaitali	1500.00
3	Chaitali	2000.00
6	Chaitali	4500.00
1	Hardik	2000.00
2	Hardik	1500.00
3	Hardik	2000.00
4	Hardik	6500.00
6	Hardik	4500.00
1	Komal	2000.00
2	Komal	1500.00
3	Komal	2000.00
1	Muffy	2000.00
2	Muffy	1500.00
3	Muffy	2000.00
4	Muffy	6500.00
5	Muffy	8500.00
6	Muffy	4500.00

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

# Cross Joins

## Customers

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

```
SQL> SELECT ID, NAME, AMOUNT, DATE
FROM CUSTOMERS, ORDERS;
```

## Orders

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

ID	NAME	AMOUNT	DATE
1	Ramesh	3000	2009-10-08 00:00:00
1	Ramesh	1500	2009-10-08 00:00:00
1	Ramesh	1560	2009-11-20 00:00:00
1	Ramesh	2060	2008-05-20 00:00:00
2	Khilan	3000	2009-10-08 00:00:00
2	Khilan	1500	2009-10-08 00:00:00
2	Khilan	1560	2009-11-20 00:00:00
2	Khilan	2060	2008-05-20 00:00:00
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
3	kaushik	1560	2009-11-20 00:00:00
3	kaushik	2060	2008-05-20 00:00:00
4	Chaitali	3000	2009-10-08 00:00:00
4	Chaitali	1500	2009-10-08 00:00:00
4	Chaitali	1560	2009-11-20 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00
5	Hardik	3000	2009-10-08 00:00:00
5	Hardik	1500	2009-10-08 00:00:00
5	Hardik	1560	2009-11-20 00:00:00
5	Hardik	2060	2008-05-20 00:00:00
6	Komal	3000	2009-10-08 00:00:00
6	Komal	1500	2009-10-08 00:00:00
6	Komal	1560	2009-11-20 00:00:00
6	Komal	2060	2008-05-20 00:00:00
7	Muffy	3000	2009-10-08 00:00:00
7	Muffy	1500	2009-10-08 00:00:00
7	Muffy	1560	2009-11-20 00:00:00
7	Muffy	2060	2008-05-20 00:00:00

# Relation: Employee

```
SQL> SELECT * FROM employee;
```

EID	ENAME	AGE	DID	EXP	SALARY	HRA
106	John	23	3	2	2000	100
103	Reddy	30	3	7	1200	120
101	Naidu	28	2	5	1800	120
102	Mark	32	1	4	1100	70
104	David	25	2	1	7000	520
105	Reddy	28	2	5	5000	300
107	Test%Test	30	4	5	8000	800

## String based retrievals

Find employee names starting from 'J'

```
SQL> SELECT ename FROM employee WHERE ename LIKE 'J%';
```

```
ENAME
```

```
-----
```

```
John
```

Find employee names ends with 'y'

```
SQL> SELECT ename FROM employee WHERE ename LIKE '%y';
```

```
ENAME
```

```
-----
```

```
Reddy
```

```
Reddy
```

## String based retrievals

Find employee names having 'a' in any position

```
SQL> SELECT ename FROM employee WHERE ename LIKE '%a%';  
ENAME
```

-----

Naidu

Mark

David

Find employee names having 'i' only in 3<sup>rd</sup> position

```
SQL> SELECT ename FROM employee WHERE ename LIKE '__i%';  
ENAME
```

-----

Naidu

## String based retrievals

Find employee names having exactly 4 characters

```
SQL> SELECT ename FROM employee WHERE LENGTH(ename)=4;
```

```
ENAME
```

```
-----
```

```
John
```

```
Mark
```

Find employee names starts with 'M' ends with 'k'

```
SQL> SELECT ename FROM employee WHERE ename LIKE 'M%k';
```

```
ENAME
```

```
-----
```

```
Mark
```

## String based retrievals

Find employee names having '%' symbol

```
SQL> SELECT ename FROM employee WHERE ename LIKE '%\%%%' ESCAPE
```

```
'\';
```

```
ENAME
```

```
-----
```

```
Test%Test
```

Find employee names starts with 'J' or 'M'

```
SQL> SELECT ename FROM employee WHERE ename LIKE 'J%' OR ename
```

```
LIKE 'M%';
```

```
ENAME
```

```
-----
```

```
John
```

```
Mark
```

## String based retrievals

Display all employee names in upper case in descending order

```
SQL> SELECT UPPER(ename) FROM employee ORDER BY ename DESC;
```

```
UPPER(ENAM
```

```
-----
```

```
TEST%TEST
```

```
REDDY
```

```
REDDY
```

```
NAIDU
```

```
MARK
```

```
JOHN
```

```
DAVID
```



## String based retrievals

Display all employee names in Lower case in ascending order

```
SQL> SELECT LOWER(ename) FROM employee ORDER BY ename;
```

```
LOWER(ENAM
```

```
-----
```

```
david
```

```
john
```

```
mark
```

```
naidu
```

```
reddy
```

```
reddy
```

```
test%test
```

# String based retrievals

## Hints:

- Like joins, UNION, INTERSECT, MINUS can be applied between two table in queries.
- WHERE clause values are case sensitive
- All single row character functions can be applied on selected column

# Summary

This session will give the knowledge about

- SQL Joins
- SQL Strings