Department of Industrial Engineering & Management

**BMS College of Engineering Bangalore-560 019**



INSTITUTIONAL ELECTIVE

SUBJECT : DBMS(17IE71EDBMS)

SEMESTER: VII

ASSIGNMENT

SUBMITTED TO: DR.B.RAVISHANKAR

SUBMITTED BY: **SAHANA T S**

USN:**1BM16EC420**

ROLL NO: **18**

1. **THE SQL CREATE TABLE Statement:**

The CREATE TABLE statement is used to used to create a table in a database.

Tables are organized into rows and columns;and each table must have a name.

The table called EMPLOYEE\_DB is created and has 6 columns:emp\_id,emp\_name,hire\_

Year,sal,depy\_name,city.

CREATE TABLE EMPLOYEE\_DB(EMP\_ID INTEGER (10), EMP\_NAME VARCHAR (255),HIRE\_YEAR INTEGER (10),SAL INTEGER (10),DEPT\_NAME VARCHAR(255),CITY VARCHAR(255));

INSERT INTO EMPLOYEE\_DB VALUES (4550,'SAMRAT',2002,50000,'MANAGER','MUMBAI');

INSERT INTO EMPLOYEE\_DB VALUES(4551,'SAMRATH GOWDA',2001,150000,'MANAGER','MUMBAI');

INSERT INTO EMPLOYEE\_DB VALUES(4552,'HARISH',1998,20000,'ADMINISTRATOR','BANGALORE');

INSERT INTO EMPLOYEE\_DB VALUES(4553,'MANISH',1998,200000,'SECUITY ENGINEER','BANGALORE');

INSERT INTO EMPLOYEE\_DB VALUES(4554,'SAHANA',2008,50000,'GENERAL MANAGER','DELHI');

INSERT INTO EMPLOYEE\_DB VALUES(4555,'SATHVIK',2008,150000,'BACKEND ENGINEER','KASHMIR');

INSERT INTO EMPLOYEE\_DB VALUES(4556,'KARTHIK',2008,90000,'FRONTENDENGINEER','RAJASTHAN');

INSERT INTO EMPLOYEE\_DB VALUES(45516,'KUMAR',2000,9000,'CLERK','CHENNAI');

INSERT INTO EMPLOYEE\_DB VALUES(4557,'SHOBHA',2000,5000,'CLERK','CHENNAI');

INSERT INTO EMPLOYEE\_DB VALUES(4558,'SINCHANA',2000,15000,'ASSISTANT ENGINEER','HYDERABAD');

INSERT INTO EMPLOYEE\_DB VALUES(4559,'SPANDANA',2001,20000,'SOFTWAREENGINEER','KERALA');

INSERT INTO EMPLOYEE\_DB VALUES(4560,'SRUJAN',1991,50000,'SENIOR.SOFTWAREENGINEER','BANGALORE');

INSERT INTO EMPLOYEE\_DB VALUES(4561,'SANJANA',2004,5000,'ADMINISTRATOR','BANGALORE');

INSERT INTO EMPLOYEE\_DB VALUES(4562,'NAVEEN',2004,1500,'CLERK','BANGALORE');

**EMP\_ID EMP\_NAME HIRE\_YEAR SAL DEPT\_NAME CITY**

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAMRATH GOWDA 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

45516 KUMAR 2000 9000 CLERK CHENNAI

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

1. **The SQL SELECT Statement**

**The SELECT statement is used to select data from a database.**

### **SELECT Syntax:**

SELECT column1, column2, ...FROM table\_name;

Here, column1, column2, ... are the field names of the table you want to select data from.

To select all the fields available in the table, use the following syntax:

SELECT \* FROM table\_name;

**The following SQL statement selects the "EMP\_NAME" and "SAL" columns from the "EMPLOYEE\_DB" table:**

SELECT EMP\_NAME,SAL FROM EMPLOYEE\_DB;

EMP\_NAME SAL

SAMRAT 50000

SAMRATH GOWDA 150000

HARISH 20000

MANISH 200000

SAHANA 50000

SATHVIK 150000

KARTHIK 90000

KUMAR 9000

SHOBHA 5000

SINCHANA 15000

SPANDANA 20000

SRUJAN 50000

SANJANA 5000

NAVEEN 1500

# **SQL SELECT DISTINCT Statement**

The SELECT DISTINCT statement is used to return only distinct (different) values. Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

### **SELECT DISTINCT Syntax**

SELECT DISTINCT column1, column2, ...FROM table\_name;

SELECT DISTINCT CITY FROM EMPLOYEE\_DB;

CITY

MUMBAI

BANGALORE

DELHI

KASHMIR

RAJASTHAN

CHENNAI

HYDERABAD

KERALA

The following SQL statement lists the **number of different (distinct)** cities:

SELECT COUNT(DISTINCT CITY) FROM EMPLOYEE\_DB;

Number of Records: 1

COUNT(DISTINCT CITY)

8

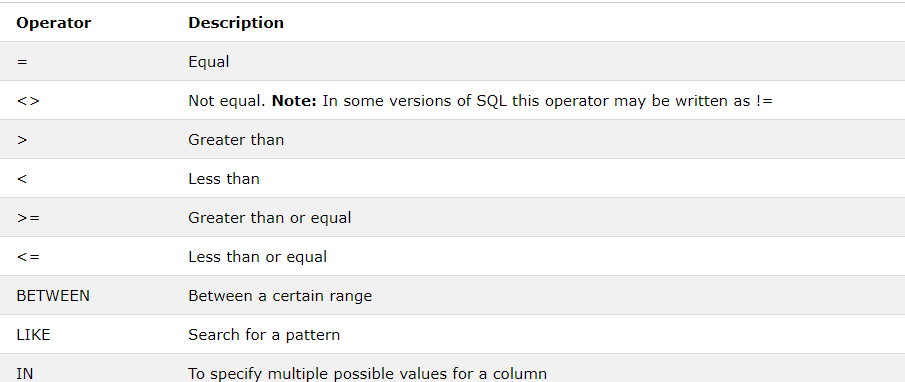
## The SQL WHERE Clause

The WHERE clause is used to filter records.The WHERE clause is used to extract only those records that fulfill a specified condition.

### **WHERE Syntax**

SELECT column1, column2, ...FROM table\_name WHERE condition;

The following operators can be used in the WHERE clause:



SQL requires **single** quotes around **text** values

SELECT \* FROM EMPLOYEE\_DB WHERE CITY='BANGALORE';

Number of Records: 5

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

However, **numeric fields** should not be enclosed in quotes:

SELECT \* FROM EMPLOYEE\_DB WHERE EMP\_ID=4553;

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

## The SQL AND, OR and NOT Operators

The WHERE clause can be combined with AND, OR, and NOT operators.

The AND and OR operators are used to filter records based on more than one condition:

* The AND operator displays a record if all the conditions separated by AND is TRUE.
* The OR operator displays a record if any of the conditions separated by OR is TRUE.
* The NOT operator displays a record if the condition(s) is NOT TRUE.

### **AND Syntax**

SELECT column1, column2, ...FROM tablenameWHERE condition1 AND condition2 AND condition3;

SELECT \* FROM EMPLOYEE\_DB WHERE CITY='BANGALORE'AND SAL=200000;

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

### **OR Syntax**

SELECT column1, column2, ...FROM tablename WHERE condition1 OR condition2 OR condition3;

SELECT \* FROM EMPLOYEE\_DB WHERE CITY='BANGALORE'OR CITY='MUMBAI';

Number of Records: 7

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAMRATH GOWDA 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

### **NOT Syntax**

SELECT column1, column2, ...FROM table\_name WHERE NOT condition;

SELECT \* FROM EMPLOYEE\_DB WHERE NOT CITY='CHENNAI';

Number of Records: 12

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

## Combining AND, OR and NOT

The following SQL statement selects all fields from "EMPLOYEE\_DB" where CITY is "BANGALORE" AND SAL must be "20000" OR "1500"

SELECT \* FROM EMPLOYEE\_DB WHERE CITY='BANGALORE' AND (SAL=20000 OR SAL=1500);Number of Records: 2

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

## The SQL ORDER BY

The ORDER BY keyword is used to sort the result-set in ascending or descending order.

The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

### **ORDER BY Syntax**

SELECT column1, column2, ...FROM table\_name ORDER BY column1, column2, ... ASC|DESC;

## ORDER BY DESC

The following SQL statement selects all customers from the "EMPLYOEE\_DB" table, sorted by the "CITY" column:

SELECT \* FROM EMPLOYEE\_DB ORDER BY CITY;

Number of Records: 14

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

45516 KUMAR 2000 9000 CLERK CHENNAI

4557 SHOBHA 2000 5000 CLERK CHENNAI

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAMRATH GOWDA 2001 150000 MANAGER MUMBAI

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

SELECT \* FROM EMPLOYEE\_DB ORDER BY SAL DESC;

emp\_id emp\_name hire\_date sal dept\_name city

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4551 SAMRATH GOWDA 2001 150000 MANAGER MUMBAI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

45516 KUMAR 2000 9000 CLERK CHENNAI

4557 SHOBHA 2000 5000 CLERK CHENNAI

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

The following SQL statement selects all EMPLOYEES from the "EMPLOYEE\_DB" table, sorted ascending by the "CITY" and descending by the "SAL " column:

SELECT \* FROM EMPLOYEE\_DB ORDER BY SAL DESC,CITY ASC;

Number of Records: 14

emp\_id emp\_name hire\_date sal dept\_name city

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4551 SAMRATH GOWDA 2001 150000 MANAGER MUMBAI

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

45516 KUMAR 2000 9000 CLERK CHENNAI

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4557 SHOBHA 2000 5000 CLERK CHENNAI

4562 NAVEEN 2004 1500 CLERK BANGALORE

## The SQL UPDATE Statement

The UPDATE statement is used to modify the existing records in a table.

### **UPDATE Syntax**

UPDATE table\_nameSET column1 = value1, column2 = value2, ...WHERE condition;

UPDATE EMPLOYEE\_DB SET EMP\_NAME='SAM',CITY='MUMBAI' WHERE EMP\_ID=4551;

Number of Records: 14

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

45516 KUMAR 2000 9000 CLERK CHENNAI

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

## The SQL DELETE Statement

The DELETE statement is used to delete existing records in a table.

### **DELETE Syntax**

DELETE FROM table\_name WHERE condition;

DELETE FROM EMPLOYEE\_DB WHERE EMP\_NAME='KUMAR' AND DEPT\_NAME='CLERK';

Number of Records: 13

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

## SQL LIMIT and LEN

**LIMIT Syntax:**

SELECT column\_name(s)FROM table\_nameWHERE condition LIMIT number;

SELECT \*FROM EMPLOYEE\_DB LIMIT 5;

Number of Records: 5

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

**LENGTH**

SELECT EMP\_NAME ,LENGTH(EMP\_NAME) AS LENGTHOFEMP\_NAME FROM EMPLOYEE\_DB;

Number of Records: 13

emp\_name LENGTHOFEMP\_NAME

SAMRAT 6

SAM 3

HARISH 6

MANISH 6

SAHANA 6

SATHVIK 7

KARTHIK 7

SHOBHA 6

SINCHANA 8

SPANDANA 8

SRUJAN 6

SANJANA 7

NAVEEN 6

# **SQL COUNT(), AVG() and SUM() Functions**

* The COUNT() function returns the number of rows that matches a specified criteria.
* The AVG() function returns the average value of a numeric column.
* The SUM() function returns the total sum of a numeric column.

### **COUNT() Syntax**

SELECT COUNT(column\_name)FROM table\_name WHERE condition;

## COUNT() Example

The following SQL statement finds the number of Employees:

SELECT COUNT (\*) FROM EMPLOYEE\_DB;

Number of Records: 1

COUNT (\*)

13

SELECT COUNT (DISTINCT CITY)FROM EMPLOYEE\_DB;

Number of Records: 1

COUNT (DISTINCT CITY)

8

### **AVG() Syntax**

SELECT AVG(column\_name)FROM table\_name WHERE condition;

## AVG() Example

The following SQL statement finds the average SAL of all Employees:

SELECT AVG(SAL) FROM EMPLOYEE\_DB;

Number of Records: 1

AVG(SAL)

62038.46153846154

SELECT AVG(SAL) AS SALAVERAGE FROM EMPLOYEE\_DB;

Number of Records: 1

SALAVERAGE

62038.46153846154

### **SUM() Syntax**

SELECT SUM(column\_name)FROM table\_name WHERE condition;

## SUM() Example

The following SQL statement finds the sum of the "SAL" fields in the "EMPLOYEE\_DB" table:

SELECT SUM(SAL)FROM EMPLOYEE\_DB;

Number of Records: 1

SUM(SAL)

806500

SELECT SUM(SAL) AS TOTALSAL FROM EMPLOYEE\_DB;

Number of Records: 1

TOTALSAL

806500

# **SQL MIN() and MAX() Functions**

* The MIN() function returns the smallest value of the selected column.
* The MAX() function returns the largest value of the selected column.

### **MAX() Syntax**

SELECT MAX(column\_name)FROM table\_name WHERE condition;

SELECT MAX (SAL) FROM EMPLOYEE\_DB;

Number of Records: 1

MAX (SAL)

200000

SELECT MAX(SAL)AS HIGHESTSAL FROM EMPLOYEE\_DB;

Number of Records: 1

HIGHESTSAL

200000

### **MIN() Syntax**

SELECT MIN(column\_name)FROM table\_name WHERE condition;

SELECT MIN(SAL) FROM EMPLOYEE\_DB;

Number of Records: 1

MIN(SAL)

1500

SELECT MIN(SAL)AS SMALLESTORDERSAL FROM EMPLOYEE\_DB;

Number of Records: 1

SMALLESTORDERSAL

1500

# **SQL LIKE Operator**

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards used in conjunction with the LIKE operator:

* % - The percent sign represents zero, one, or multiple characters
* \_ - The underscore represents a single character
* The percent sign and the underscore can also be used in combinations!

### **LIKE Syntax**

SELECT column1, column2, ...FROM table\_name WHERE columnN LIKE pattern;

* The following SQL statement selects all customers with a city starting with "b"

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE 'B%';

Number of Records: 5

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

* The following SQL statement selects all customers with a city ending with "i":

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE '%I';

Number of Records: 4

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4557 SHOBHA 2000 5000 CLERK CHENNAI

* The following SQL statement selects all employees with a city that have "bai" in any position:

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE '%BAI%';

Number of Records: 2

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

* The following SQL statement selects all empoyees with a city that does NOT have “bai” in between:

SELECT \*FROM EMPLOYEE\_DB WHERE CITY NOT LIKE '%BAI%';

Number of Records: 11

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

* The following SQL statement selects all employees with a city that does NOT end with "i":

SELECT \*FROM EMPLOYEE\_DB WHERE CITY NOT LIKE '%I';

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

4562 NAVEEN 2004 1500 CLERK BANGALORE

* The following SQL statement selects all customers with a city starting with "ch"

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE 'CH%';

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4557 SHOBHA 2000 5000 CLERK CHENNAI

* The following SQL statement selects all employess with a City starting with any character, followed by "umbai":

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE '\_UMBAI%';

Number of Records: 2

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

* The following SQL statement selects all customers with a City starting with "c", followed by any character, followed by "e", followed by any character, followed by "ai":

SELECT \*FROM EMPLOYEE\_DB WHERE CITY LIKE 'C\_E\_\_\_AI%';

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4557 SHOBHA 2000 5000 CLERK CHENNAI

## The SQL BETWEEN Operator

The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

The BETWEEN operator is inclusive: begin and end values are included.

### **BETWEEN Syntax**

SELECT column\_name(s)FROM table\_name WHERE column\_name BETWEEN value1 AND value2;

* The following SQL statement selects all employes with a sal BETWEEN 1500 and 2000:

SELECT \*FROM EMPLOYEE\_DB WHERE SAL BETWEEN 1500 AND 2000;

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4562 NAVEEN 2004 1500 CLERK BANGALORE

* To display the employees outside the range of the previous example, use NOT BETWEEN:

SELECT \*FROM EMPLOYEE\_DB WHERE SAL NOT BETWEEN 1500 AND 2000;

Number of Records: 12

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

* The following SQL statement selects all employes with a sal BETWEEN 1500 and 2000. In addition do not show products with a emp\_ID 4550,4551:

SELECT \*FROM EMPLOYEE\_DB WHERE (SAL BETWEEN 1500 AND 2000) AND NOT EMP\_ID IN(4550,4551);

Number of Records: 1

emp\_id emp\_name hire\_date sal dept\_name city

4562 NAVEEN 2004 1500 CLERK BANGALORE

* The following SQL statement selects all employees BETWEEN s' and 't':

SELECT \*FROM EMPLOYEE\_DB WHERE EMP\_NAME BETWEEN 'S' AND 'T';

Number of Records: 9

emp\_id emp\_name hire\_date sal dept\_name city

4550 SAMRAT 2002 50000 MANAGER MUMBAI

4551 SAM 2001 150000 MANAGER MUMBAI

4554 SAHANA 2008 50000 GENERAL MANAGER DELHI

4555 SATHVIK 2008 150000 BACKEND ENGINEER KASHMIR

4557 SHOBHA 2000 5000 CLERK CHENNAI

4558 SINCHANA 2000 15000 ASSISTANT ENGINEER HYDERABAD

4559 SPANDANA 2001 20000 SOFTWAREENGINEER KERALA

4560 SRUJAN 1991 50000 SENIOR.SOFTWAREENGINEER BANGALORE

4561 SANJANA 2004 5000 ADMINISTRATOR BANGALORE

* The following SQL statement selects all employees NOT BETWEEN s' and 't'

SELECT \*FROM EMPLOYEE\_DB WHERE EMP\_NAME NOT BETWEEN 'S' AND 'T';

Number of Records: 4

emp\_id emp\_name hire\_date sal dept\_name city

4552 HARISH 1998 20000 ADMINISTRATOR BANGALORE

4553 MANISH 1998 200000 SECUITY ENGINEER BANGALORE

4556 KARTHIK 2008 90000 FRONTENDENGINEER RAJASTHAN

4562 NAVEEN 2004 1500 CLERK BANGALORE

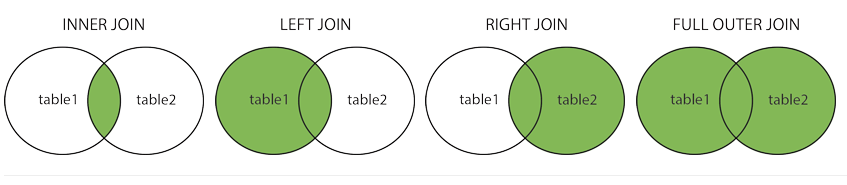
# **SQL Joins**

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

**Different Types of SQL JOINs**

Here are the different types of the JOINs in SQL:

* **(INNER) JOIN**: Returns records that have matching values in both tables
* **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
* **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
* **FULL (OUTER) JOIN**: Return all records when there is a match in either left or right table



CREATE TABLE EMP\_PROJ(EMP\_ID INTEGER(10),PROJ\_NAME VARCHAR(255),PROJ\_LOCATION VARCHAR(255));

INSERT INTO EMP\_PROJ VALUES(4552,'AUTOMATION','UK');

INSERT INTO EMP\_PROJ VALUES(4553,'IOT','AUSTRALIA');

INSERT INTO EMP\_PROJ VALUES(4556,'PYTHON WORKSHOP','INDIA');

INSERT INTO EMP\_PROJ VALUES(4557,'NETWORK SECURITY','INDIA');

Number of Records: 4

EMP\_ID PROJ\_NAME PROJ\_LOCATION

4552 AUTOMATION UK

4553 IOT AUSTRALIA

4556 PYTHON WORKSHOP INDIA

4557 NETWORK SECURITY INDIA

SELECT EMPLOYEE\_DB.EMP\_ID,EMPLOYEE\_DB.EMP\_NAME,EMPLOYEE\_DB.SAL,EMP\_PROJ.PROJ\_NAME,EMP\_PROJ.PROJ\_LOCATION FROM EMPLOYEE\_DB JOIN EMP\_PROJ ON EMPLOYEE\_DB.EMP\_ID=EMP\_PROJ.EMP\_ID;

Number of Records: 4

emp\_id emp\_name sal PROJ\_NAME PROJ\_LOCATION

4552 HARISH 20000 AUTOMATION UK

4553 MANISH 200000 IOT AUSTRALIA

4556 KARTHIK 90000 PYTHON WORKSHOP INDIA

4557 SHOBHA 5000 NETWORK SECURITY INDIA

1. **UPPERCASE() FUNCTION**

The UPPER() functiom converts the value of a field to uppercase:

SELECT UPPER(EMP\_NAME) FROM EMPLOYEE\_DB;

Number of Records: 13

UPPER(EMP\_NAME)

SAMRAT

SAM

HARISH

MANISH

SAHANA

SATHVIK

KARTHIK

SELECT UPPER(EMP\_NAME)AS EMPLOY\_NAME,CITY FROM EMPLOYEE\_DB;

Number of Records: 13

EMPLOY\_NAME city

SAMRAT MUMBAI

SAM MUMBAI

HARISH BANGALORE

MANISH BANGALORE

SAHANA DELHI

SATHVIK KASHMIR

KARTHIK RAJASTHAN

SHOBHA CHENNAI

SINCHANA HYDERABAD

SPANDANA KERALA

SRUJAN BANGALORE

SANJANA BANGALORE

NAVEEN BANGALORE

1. **LOWERCASE() FUNCTION**

The LOWER() functiom converts the value of a field to lowercase:

SELECT LOWER(EMP\_NAME) FROM EMPLOYEE\_DB;

Number of Records: 13

LOWER(EMP\_NAME)

samrat

sam

harish

manish

sahana

sathvik

karthik

shobha

sinchana

spandana

srujan

sanjana

naveen

SELECT LOWER(EMP\_NAME)AS EMPLOY\_NAME,CITY FROM EMPLOYEE\_DB;

Number of Records: 13

EMPLOY\_NAME city

samrat MUMBAI

sam MUMBAI

harish BANGALORE

manish BANGALORE

sahana DELHI

sathvik KASHMIR

karthik RAJASTHAN

shobha CHENNAI

sinchana HYDERABAD

spandana KERALA

srujan BANGALORE

sanjana BANGALORE

naveen BANGALORE