

25/08/25

## Task 3.1

### USING CLAUSES OPERATORS AND FUNCTIONS IN QUERIES

Aim: To implemented of DML Commands using clauses, operators and functions in queries.

#### CLAUSES

→ WHERE, order by, Group by, Having, Distinct

#### Operators

→ Equal (=)

→ Between/

→ And

→ OR

→ IN

Create table department (

Dept ID INT PRIMARY KEY,  
Dept Name VARCHAR(50) UNIQUE NOT NULL  
Location VARCHAR(50) NOT NULL);

CREATE TABLE STUDENT(

STUDENTID INT PRIMARY KEY.

NAME VARCHAR(50) NOT NULL,

AGE INT CHECK (AGE >= 18).

DEPT ID INT FOREIGN KEY REFERENCES

DEPARTMENT(DEPT ID),

CITY VARCHAR(50) DEFAULT 'UNKNOWN',

IO IN DATE DATE TIME 'DEFAULT GET DATE(1)'

INSERT INTO DEPARTMENT VALUES

(1, 'CSE', 'HYDERABAD'),

(2, 'ECE', 'MUMBAI');

(3, 'MECH', 'DELHI');

INSERT INTO STUDENT VALUES

(101, UPPER('rahul'), 20, 1, 'HYDERABAD');

INSERT INTO STUDENT VALUES

(102, 'ANJALI', 22, 2, 'MUMBAI');

INSERT INTO STUDENT VALUES

(103, 'KIRAN', 19, 1, 'PUNE');

INSERT INTO

SELECT \* FROM STUDENT

	Student ID	Name	AGE	DEPT ID	CITY	JOIN DATE
1	101	Rahul	20	1	Hyderabad	2025-08-26
2	102	Anjali	22	2	Mumbai	2025-08-26
3	103	Kiran	19	1	Pune	2025-08-26

SELECT \* FROM DEPARTMENT

	DEPT ID	DEPT NAME	LOCATION
1	1	CSE	HYD
2	2	ECE	MUMBAI
3	3	MECH	DELHI

SELECT NAME, AGE

FROM STUDENT

WHERE AGE BETWEEN 19 AND 22

	Name	AGE
1	Rahul	20
2	Anjali	22
3	Kiran	19

```
SELECT Name, DEPT ID
FROM STUDENT 1
WHERE DEPT ID IN(1,3)
ORDER DEPT ID DESC.
```

	NAME	DEPT ID
1.	MOHITH	3
2	SALAK	1
3	RAHUL	1
4	KIRAN	1

```
UPDATE STUDENT
```

```
SET AGE = AGE+1
```

```
WHERE DEPT ID=1 AND AGE < 21;
```

	STU ID	NAME	AGE	DEPT ID	CITY	JOIN DATE
1	101	Rahul	21	1	HYD	2025-08-26
2	102	Anjali	22	2	MUMBAI	2025-8-26
3	103	Kiran	20	1	PUNE	2025-8-26

select distinct city

from student 1;

	CITY
1	DELHI
2	HYDERABAD
3	MUMBAI
4	PUNE

VEL TECH - CSE	
EX NO.	2.1
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	—
TOTAL (20)	15
SIGN WITH DATE	

25/08/25

Result: The implementation of the clauses, operators & functions in the query  
(DOL and DML Commands)

2/08/2025

## Task (3-2) AGGREGATE FUNCTIONS

Aim: To study & implement aggregate functions (count(), sum(), avg(), min(), max()) on a sample database

### AGGREGATE FUNCTIONS

These are mostly used with Grouped by to group the rows

→ COUNT()

→ SUM()

→ AVG()

→ MIN()

→ MAX()

CREATE TABLE STUDENT 21

ROLLNO INT PRIMARY KEY

NAME VARCHAR(50),

AGE INT,

DEPT ID INT,

MARKS INT);

INSERT INTO STUDENT 21 VALUES

(1, 'Ajun', 20, 101, 85)

(2, 'Sreha', 21, 101, 90)

(3, 'Ravi', 19, 102, 95)

(4, 'Priya', 22, 102, 95),

(5, 'Kiran', 20, 101, 60),

(6, 'Anita', 23, 103, 80)

SELECT \* FROM STUDENT Q

	ROLL NO	NAME	AGE	DEPT ID	MARKS
1	1	Arjun	20	101	85
2	2	Snega	21	101	90
3	3	Ravi	19	102	76
4	4	Priya	22	102	95
5	5	Kiran	20	101	60
6	6	Anika	23	103	88

SELECT DEPT ID, AVG (MARKS) AS - MARKS  
FROM STUDENT Q

GROUPED BY DEPT ID;

	DEPT ID	AVG - MARKS
1	101	78
2	102	85
3	103	88

SELECT DEPT ID, MAX (MARKS) AS TOP-MARKS  
FROM STUDENT Q

GROUP BY DEPT ID;

	DEPT ID	TOP-MARKS
1	101	90
2	102	95
3	103	88

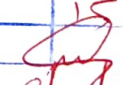


SELECT DEPT ID, MIN (MARKS) AS LEAST - MARK  
 FROM STUDENT 2.  
 GROUP BY DEPT ID.

	DEPT ID	LEAST - MARKS
1	101	60
2	102	70
3	103	88

SELECT DEPT ID, COUNT (\*) AS STU, COUNT  
 FROM STUDENT 2  
 GROUP BY DEPT ID;

	DEPT ID	STU - COUNT
1	101	3
2	102	2
3	103	1

VEL TECH - CSE	
EX NO.	3.2
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	—
TOTAL (20)	15
SIGN WITH DATE	

8/09/25

Result: Implementation of all aggregate functions has been performed Successfully on a table.