19/8/25

## Task 3.1 Using clauses, operators and functions on Queries

Title: Implementation of DML commands using clauses, operators and functions on queries.

* Insert table
* Select table
* Update table
* Delete table

Objective:

* To understand the different issues involved in the design and implementation of a database system.

DML:

1. Insert 2. delete 3. Update 4. delete.

Insert Into: This is used to add records

INSERT INTO < relation / table name > ( field-1, field-2, ... field-n) values (data-1, data-2, ... data-n);

Example: SQL > Insert Into member value (104, 'Shan', 'HR', 'male');

EMP

| Employee name | ID | Age | place | gender | Role. |
|---|---|---|---|---|---|
| Rajesh | 102 | 35 | chennai | M | Developer |
| Rani | 103 | 38 | Trichy | M | Developer |
| Shaan | 104 | 30 | Karur | M | HR |

Update - SET - WHERE : Used to update.

Syntax : SQL > update relation name SET field name =data, field-name 2=data, where field - name = data; WHERE field-name =data.;

Example : UPDATE Employee SET Employee name ='valan' WHERE EMP ID =104;

| Employee name | EMP-ID | Age | place | gender | Role |
|---|---|---|---|---|---|
| Rajesh | 102 | 35 | Chennai | M | Developer |
| Ravi | 103 | 38 | Trichy | M | Developer |
| valan | 104 | 30 | Karur | M | HR |

DELETE FROM : used to delete all the records

Syntax: DELETE FROM relation - name WHERE condition.

Example : DELETE FROM - Where : Used to delete a selected record from a relation.

DELETE FROM Employee where age =35

output

| Employee name | EMP-ID | Age | place | gender | Role |
|---|---|---|---|---|---|
| Ravi | 103 | 38 | Trichy | M | Developer |
| valan | 104 | 30 | Karur | M | HR |

Truncate : Delete all values and structure remaens same.

Example : TRUNCATE Employee

Queries

1. Retrieve member name end with letter 'a' and member no between 101 and 103

Query :

```
SELECT first name,
last_name, salary
FROM employees
Where first_name
Like '%.m';
```

Output

| Employee name | ID | Age | Place | gender | Role |
|---|---|---|---|---|---|
| Rani | 103 | 38 | Trichy | M | Developer |

2. List the salary where pagecount between clause and operator.

Query :

```
Selec * from book where
salary between 1000 and 2000;
```

Output

| id | Employee name | salary |
|---|---|---|
| 103 | Rane | 20,000 |

3. Find the records who has maximum number of ~~recordset~~ salary.

Output

| ~~Id~~ | Employee name | salary |
|---|---|---|
| 104 | Rajesh | 15,000 |

4. Find the records whose ~~Pdate~~ doj is greater than 2022-02-07

| ID | Employee name | doj |
|---|---|---|
| 103 | Rane | 2022-03-07 |

5. Combine the records of member id and employee relation union

Query.

```
SELECT membno FROM member
Union Select membno FROM
borrowed;
```

Output

| employee id |
|---|
| 101 |
| 102 |
| 103 |
| 104 |

6. Find the employee and their place details using group by and orderby clauses

Query.

```
SELECT Employee, places,
Count(*) no FROM book places
Group By Employee, publisher
Order by Employee.
```

Output

| Employee name | places |
|---|---|
| Raul | chennai |
| Rajesh | chennai |

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

Result: To implement the DML commands executed. successfully using clauses, operators and function in Queries completed successfully

25/8/25

# Task 3.2 Aggregate Functions

Aim:

To study and implement aggregate functions (count(), sum(), AVG(), MIN(), MAX()) on a sample student database

Procedure

1. Create a table named students
2. Insert sample records
3. Write queries using aggregate functions
4. Observe and record the output

Commands with explanation

1) Count total number of employees

```
SELECT COUNT(*) AS Total_students
FROM Employee;
```

Explanation:

* Count(*) counts how many rows (Employee) are in the table.
* As Total - Employee gives a user friendly column name.

2) Find the highest ~~marks~~ salary obtained by a student.

```
SELECT MAX(salary) AS Highest
-salary
FROM EMPLOYEE
```

Output

| Employee name | Id | salary |
|---|---|---|
| Rane | 103 | 40,000 |

3) Find the minimum salary among the employees

| Employee name | ID | salary |
|---|---|---|
| Rajesh | 104 | 15,000 |

4) Find the highest salary among the employees

| Employee name | ID | salary |
|---|---|---|
| Rane | 103 | 40,000 |

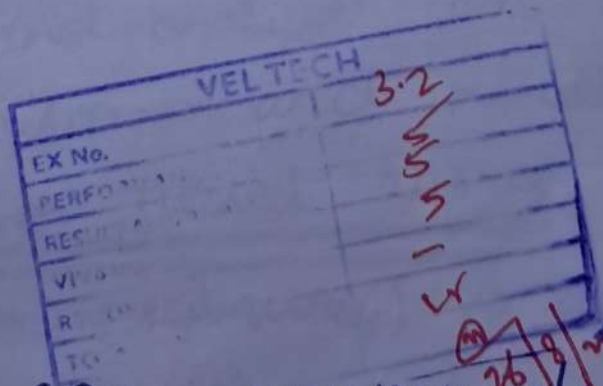
5) Find the average age of Employees

~~Employee name~~

```
SELECT AVG (Age) AS AVG - ~~Marks~~ Age
FROM employees
```

Output

| AVG-AGE |
| --- |
| 32 |




Result: Thus ~~SQL~~ commands Aggregate functions Executed successfully based on Employee is created successfully.