# Day 7 Notes

## DATABASE & AUTHENTICATION

---

### 1. Database Basics (CRUD Operations)

CRUD stands for:

- **C**reate → Create database / table
- **R**ead → Read or display data
- **U**pdate → Modify existing data
- **D**elete → Remove data

---

### 2. SQL Commands Overview

#### 2.1 CREATE

Used to create databases or tables.

```
CREATE TABLE User (
    id INT,
    username VARCHAR(60),
    password VARCHAR(60)
);
```

---

#### 2.2 INSERT

Used to add new data into a table.

```
INSERT INTO table_name
VALUES (value1, value2, value3);
```

Example:

```
INSERT INTO User

VALUES (1, 'sriya', 'password123');
```

**Note:**

- SQL statements usually end with a **semicolon (;)**

---

**2.3 SELECT**

Used to read or display data from tables.

```
SELECT column_name FROM table_name;
```

Examples:

```
SELECT id FROM student;

SELECT * FROM User;
```

---

**2.4 UPDATE**

Used to modify existing data.

```
UPDATE table_name

SET column_name = value

WHERE condition;
```

Example:

```
UPDATE User

SET username = 'admin'

WHERE id = 1;
```

---

**2.5 DELETE**

Used to remove records from a table.

```
DELETE FROM table_name WHERE condition;
```

Example:

```
DELETE FROM User WHERE id = 1;
```

---

**3. DROP vs TRUNCATE vs RENAME**

**DROP**

- Deletes **entire table or database**
- Structure + data are removed permanently

```
DROP TABLE User;
```

---

**TRUNCATE**

- Deletes **all rows** from a table
- Table structure remains

```
TRUNCATE TABLE User;
```

---

**RENAME**

Used to rename a table.

```
RENAME TABLE User TO Users;
```

---

**4. SQL Operators**

**Comparison Operators**

- = equal
- != or <> not equal
- > greater than
- < less than

---

**Logical Operators**

- **AND**
- **OR**
- **NOT**

Example:

```
SELECT * FROM User WHERE id = 1 AND username = 'sriya';
```

---

**5. ORDER BY & GROUP BY**

**ORDER BY**

Sorts results.

```
SELECT * FROM User ORDER BY username ASC;
```

---

**GROUP BY**

Groups rows that have same values.

```
SELECT department, COUNT(*)

FROM Employee

GROUP BY department;
```

---

**6. Aggregate Functions**

Used to perform calculations on data.

- MIN() – Minimum value
- MAX() – Maximum value
- AVG() – Average
- COUNT() – Number of records

Example:

```
SELECT COUNT(*) FROM User;
```

---

### 7. SQL Patterns (LIKE operator)

Used with **LIKE** to search patterns.

| Pattern | Meaning | Example Match |
|---|---|---|
| `A%` | *Starts with A* | *Apple* |
| `%a` | *Ends with a* | *Sathiya* |
| `%a%` | *Contains a* | *Ball* |
| `A%e` | *Starts with A, ends with e* | *Apple* |

Example:

```
SELECT * FROM User WHERE username LIKE 'A%';
```

---

### 8. Joins in SQL

Used to combine data from multiple tables.

**INNER JOIN**

Returns matching records from both tables.

```
SELECT *

FROM A

INNER JOIN B ON A.id = B.id;
```

---

**LEFT JOIN**

Returns all records from left table + matching from right.

---

**RIGHT JOIN**

Returns all records from right table + matching from left.

---

**CROSS JOIN**

Returns Cartesian product.

---

## 9. Special SQL Keywords

- **IN** → Matches multiple values
- **BETWEEN** → Range
- **LIKE** → Pattern matching
- **IS NULL** → Checks NULL values

Example:

```
SELECT * FROM User WHERE id IN (1,2,3);
```

---

**10. UNION & Related Commands**

- **UNION** → Combines results, removes duplicates
- **UNION ALL** → Keeps duplicates
- **INTERSECT** → Common rows
- **EXCEPT** → Rows from first query not in second

---

**11. HAVING Clause**

Used with **GROUP BY** to filter groups.

```
SELECT department, COUNT(*)

FROM Employee

GROUP BY department

HAVING COUNT(*) > 5;
```

---

**12. JWT (JSON Web Token)**

**What is JWT?**

JWT is a **compact and secure way** to send information between client and server as a **signed token**.

Used in:

- Authentication
- Authorization
- Stateless APIs

---

**JWT Workflow**

1. User logs in (username/password)
2. Server validates credentials
3. Server generates JWT
4. Client stores token (localStorage / cookies)
5. Client sends token in request header
6. Server verifies token and allows access

---

**JWT in Python**

Install:

```
pip install pyjwt
```

Code:

```
import jwt

import datetime
```

```
SECRET_KEY = "mysecretkey"


payload = {

    "user_id": 101,

    "username": "sriya",

    "exp": datetime.datetime.utcnow() +
datetime.timedelta(minutes=30)

}


token = jwt.encode(payload, SECRET_KEY, algorithm="HS256")

print(token)
```

---

**13. OAuth (Overview)**

OAuth is an **authorization framework** that allows third-party apps to access user data **without sharing passwords**.

Examples:

- Login with Google
- Login with GitHub
- Login with Facebook

---