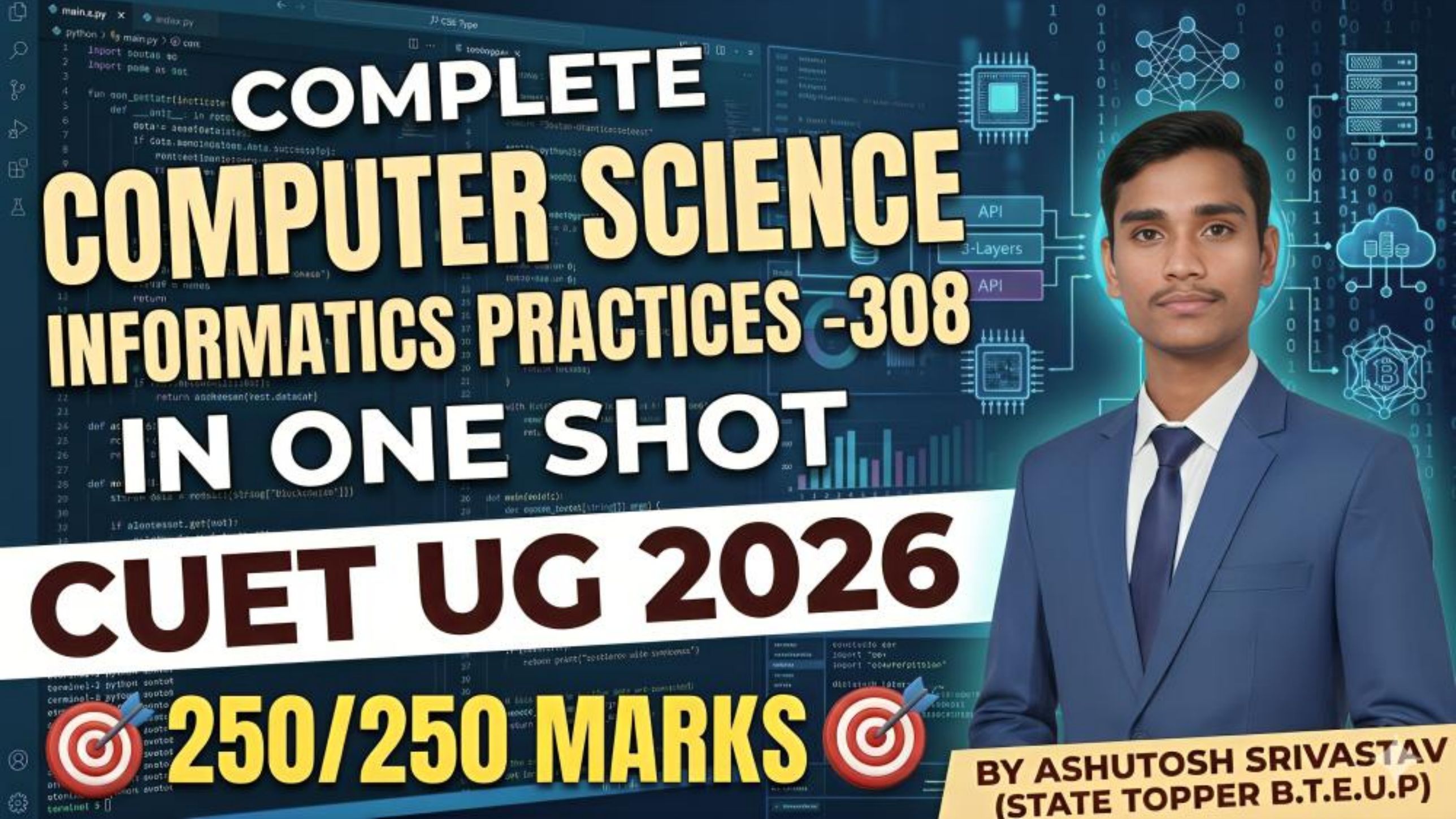
COMPLETE COMPUTER SCIENCE INFORMATICS PRACTICES -308 IN ONE SHOT

CUET UG 2026

250/250 MARKS



BY ASHUTOSH SRIVASTAV
(STATE TOPPER B.T.E.U.P)

SYLLABUS

Section A

1: Database Concepts

2: Structured Query Language – I

3: Structured Query Language – II

4: Computer Networks

Section B1: Computer Science

1: Exception and File Handling in Python

2: Stack

3: Queue

4: Searching

5: Sorting

6: Understanding Data

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

SYLLABUS

7: Database Concepts

8: Structured Query Language

9: Computer Networks

10: Data Communication

11: Security Aspects

Section B2: Informatics Practices

1: Database Query using SQL

2: Data Handling using Pandas – I

3: Data Handling using Pandas – II

4: Plotting Data using Matplotlib

5: Introduction to Computer Networks

6: Societal Impacts

7: Project Based Learning

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

Home
Shorts
Subscriptions
You
History

Sign in to like videos, comment, and subscribe.
Sign in

Explore
Shopping
Music
Movies
Show more

More from YouTube
YouTube Premium

KHUSHI FOUNDATION ACADEMY

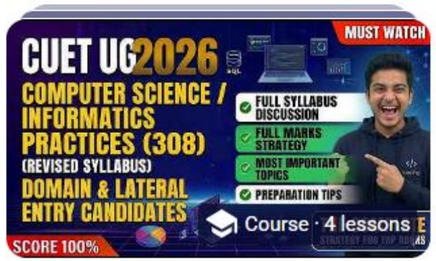
@KHUSHI.FOUNDATION.ACADEMY · 2.47K subscribers · 194 videos

यहां हम " गणित" का समाधान हिंदी और अंग्रेजी दोनों भाषाओं में आसान और स्पष्ट तरीके से प्रदान करने क ...more

t.me/khushifoundationacademy and 1 more link

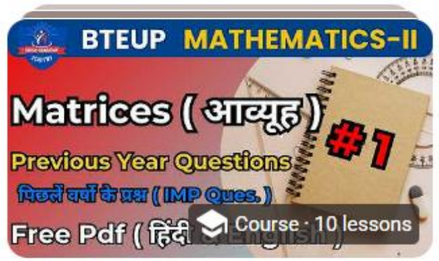
Subscribe Join

Home Videos Live Courses Playlists Posts



CUET UG 2026 Computer Science / Information Practices Full...

View full course

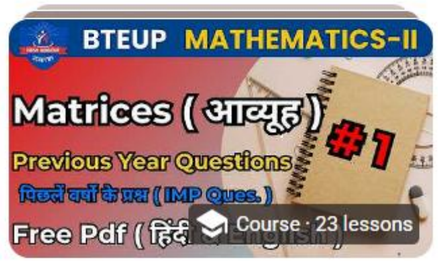


Matrix Questions for UP Polytechnic Second Semester Math for all...

View full course

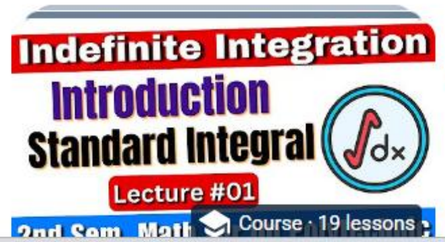
Integral Calculus 2nd Semester Math for UP Polytechnic 2025

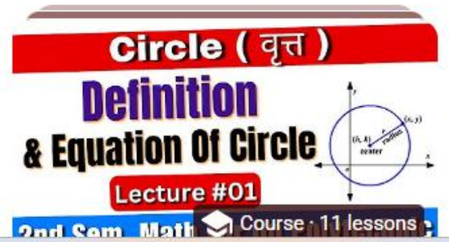
View full course



Applied Mathematics 3rd Important Questions #importantquestions...

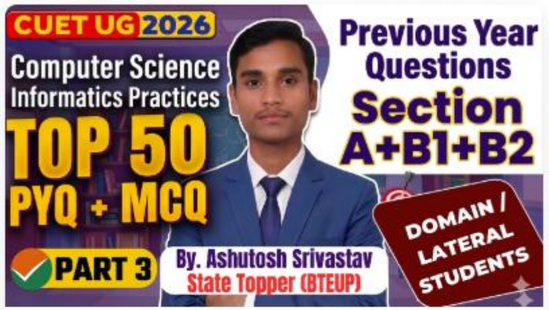
View full course



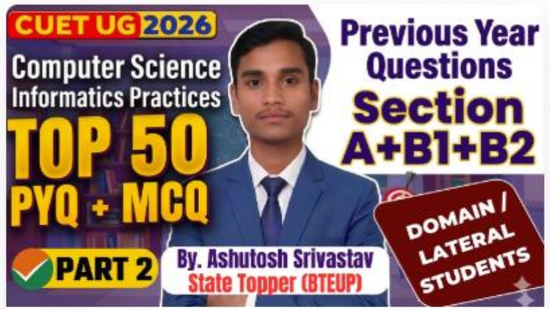


FREE NOTES

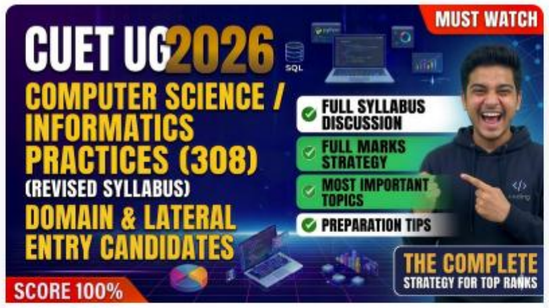
Download All Available Free Notes



Cuet Ug PYQ Part 3



CSE CUET PART 2



CUET UG CS/IP 2026 Syllabus Discussion

DATABASE CONCEPTS

What is a Database?

A **database** is an organized collection of data that can be easily accessed, managed, and updated.

Example:

Student records , Bank accounts , Library system

What is DBMS?

A **DBMS (Database Management System)** is software that allows users to **create, manage, and manipulate databases**.

Examples:

MySQL , Oracle , SQL Server

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

What is RDBMS?

An **RDBMS (Relational DBMS)** stores data in the form of **tables (relations)**.

Each table has:

Rows → **Tuples**

Columns → **Attributes**

ID	Name	Age
1	A	20

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

1. CONSTRAINTS IN RDBMS

Constraints are **rules applied on table columns** to maintain **data integrity, accuracy, and reliability**.

1.1 Primary Key Constraint

Ensures **uniqueness + NOT NULL**

Only **one primary key per table**

Can be **single column or composite**

Example:

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT
);
```

Valid Insert:

```
INSERT INTO Students VALUES (1, 'Aman', 20);
```

Invalid Insert:

```
INSERT INTO Students VALUES (1, 'Rahul', 22);
Error: Duplicate primary key
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

1.2 Foreign Key Constraint

Maintains **referential integrity**

Links one table to another

Invalid Insert:

```
INSERT INTO Employees VALUES (1, 'Ravi', 99);
```

Error: DeptID 99 does not exist

Example:

```
CREATE TABLE Departments (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(50)
);
CREATE TABLE Employees (
    EmpID INT PRIMARY KEY,
    Name VARCHAR(50),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES
Departments(DeptID)
);
```

DATABASE CONCEPTS

1.3 Unique Constraint

Ensures **all values are unique**

Allows **NULL values**

```
CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    Email VARCHAR(100) UNIQUE
);
```

1.4 Check Constraint

Applies **condition on values**

```
CREATE TABLE Employees (
    EmpID INT,
    Age INT CHECK (Age >= 18)
);
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

1.5 Default Constraint

Provides **default value**

```
CREATE TABLE Orders (
    OrderID INT,
    Status VARCHAR(20) DEFAULT 'Pending'
);
```

DATABASE CONCEPTS

2. KEYS IN RDBMS

2.1 Candidate Key

All possible attributes that can uniquely identify a row

Example:

StudentID, Email → both can be candidate keys

2.2 Primary Key

Selected candidate key

DATABASE CONCEPTS

2.3 Composite Key

Combination of multiple columns

```
CREATE TABLE Enrollment (
    StudentID INT,
    CourseID INT,
    PRIMARY KEY (StudentID, CourseID)
);
```

2.4 Foreign Key

Links one table to another

DATABASE CONCEPTS

2.5 Super Key

Any set of attributes that uniquely identifies rows

Example:

`{StudentID}, {StudentID, Name}`

2.6 Alternate Key

Candidate keys not chosen as primary key

DATABASE CONCEPTS

3. DATABASE ARCHITECTURE (3-LEVEL ARCHITECTURE)

3.1 External Level (View Level)

What **user sees**

Different users → different views

Example:

Student sees marks only

Admin sees all data

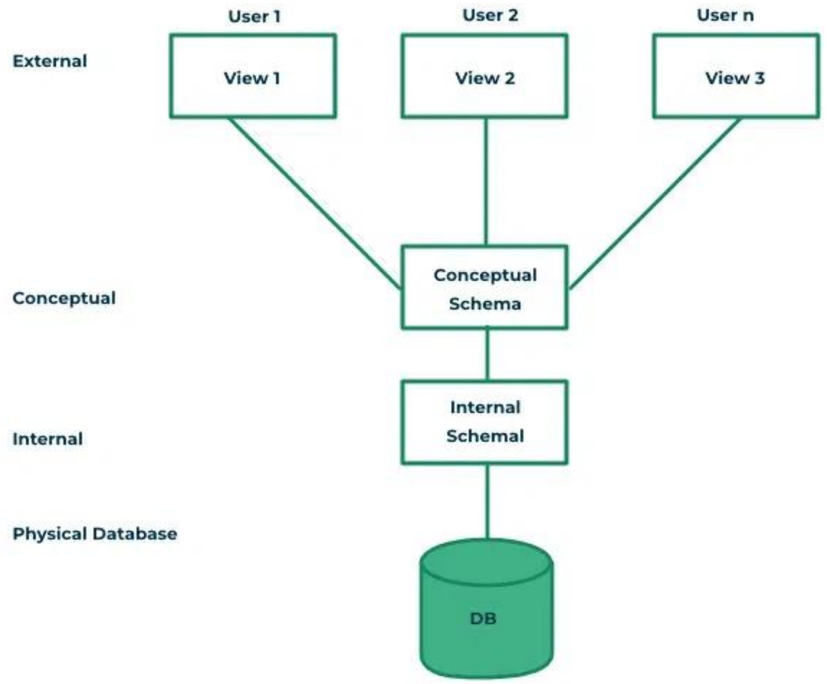
3.2 Conceptual Level (Logical Level)

Structure of database

Tables, relationships

3.3 Internal Level (Physical Level)

How data is stored in memory



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

4. RELATIONAL ALGEBRA

A **procedural query language** used in DBMS.

Defines **how to get data**

Forms base of SQL

5. RELATIONAL ALGEBRA OPERATIONS

5.1 Selection (σ Sigma)

Selects rows satisfying condition

Relational Algebra:

```
σ(Age > 30)(Student)
```

SQL:

```
SELECT * FROM Student WHERE Age > 30;
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

5.2 Projection (π)

Retrieves specific columns and removes duplicates

Relational Algebra:

```
π(Name, Address)(Student)
```

SQL:

```
SELECT DISTINCT Name, Address FROM Student;
```

5.3 Union (∪)

Combines two relations with distinct rows

```
SELECT Name FROM A
UNION
SELECT Name FROM B;
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

5.4 Set Difference (−)

Rows in one relation but not in another

```
SELECT Name FROM A
EXCEPT
SELECT Name FROM B;
```

5.5 Cartesian Product (×)

Combines every row of one table with every row of another

```
SELECT * FROM A, B;
```

DATABASE CONCEPTS

WHAT IS JOIN

A JOIN is used to **combine rows from two or more tables** based on a **related column (usually a key)**.

Why JOIN is needed:

In relational databases:

Data is stored in **multiple tables**

To avoid redundancy (normalization)

So to get complete information → we **join tables**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

Table 1: Students

ID	Name	DeptID
1	Aman	101
2	Ravi	102
3	Sita	103

Table 2: Departments

DeptID	DeptName
101	CSE
102	ECE

Now question:

→ **“Which student belongs to which department?”**

Answer requires JOIN

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

2. BASIC JOIN SYNTAX

```
SELECT columns
FROM Table1
JOIN Table2
ON Table1.common_column = Table2.common_column;
```

DATABASE CONCEPTS

3. TYPES OF JOIN (VERY IMPORTANT)

3.1 INNER JOIN (MOST IMPORTANT)

Returns only those rows where **matching values exist in both tables**

SQL:

```
SELECT Students.Name, Departments.DeptName
FROM Students
INNER JOIN Departments
ON Students.DeptID = Departments.DeptID;
```

Note :- INNER JOIN = Intersection

DATABASE CONCEPTS

3.2 LEFT JOIN (LEFT OUTER JOIN) Returns:

All rows from **left table**

Matching rows from right table

If no match → NULL

SQL:

```
SELECT Students.Name, Departments.DeptName
FROM Students
LEFT JOIN Departments
ON Students.DeptID = Departments.DeptID;
```

DATABASE CONCEPTS

3.2 LEFT JOIN (LEFT OUTER JOIN)

Returns:

All rows from **left table**

Matching rows from right table

If no match → NULL

SQL:

```
SELECT Students.Name, Departments.DeptName
FROM Students
LEFT JOIN Departments
ON Students.DeptID = Departments.DeptID;
```

DATABASE CONCEPTS

3.3 RIGHT JOIN (RIGHT OUTER JOIN)

Returns:

All rows from **right table**

Matching rows from left table

SQL:

```
SELECT Students.Name, Departments.DeptName
FROM Students
RIGHT JOIN Departments
ON Students.DeptID = Departments.DeptID;
```

DATABASE CONCEPTS

3.4 FULL OUTER JOIN

Returns:

All rows from both tables

Non-matching → NULL

SQL:

```
SELECT Students.Name, Departments.DeptName
FROM Students
FULL OUTER JOIN Departments
ON Students.DeptID = Departments.DeptID;
```

DATABASE CONCEPTS

RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

An RDBMS stores data in the form of **tables (relations)**:

Rows → represent records (tuples)

Columns → represent attributes (fields)

Table Example:

```
Student Table

ID | Name  | Age

---------------

1  | Aman  | 20

2  | Ravi  | 22
```

ID, Name, Age → Attributes

Each row → Tuple

Entire table → Relation

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

Important Insight:

Data is structured → easy querying

Relationships between tables → maintained using keys

Real-life Systems:

MySQL

Oracle

SQL Server

DATABASE CONCEPTS

2. TYPES OF USERS IN RDBMS

2.1 End Users

Who:

Normal users interacting via apps

Examples:

ATM users

Online shopping customers

Role:

Do not write SQL

Use interface (buttons, forms)

DATABASE CONCEPTS

2.2 Application Programmers

Who:

Developers who write programs

Role:

Use languages like:

- Java
- Python

Write queries inside code

Example:

```
SELECT * FROM Student WHERE ID = 1;
```

DATABASE CONCEPTS

2.3 Database Administrators (DBA)

Responsibilities:

Security (who can access what)

Backup & recovery

Performance optimization

Database design

Insight:

DBA = **Controller of database system**

DATABASE CONCEPTS

3. LIMITATIONS OF RDBMS

3.1 Scaling Problem

Difficult to scale horizontally (big data systems struggle)

3.2 Complex Relationships

Managing many tables → complex joins

3.3 Query Performance

Complex queries → slower execution

3.4 Structural Changes

Altering schema is difficult

```
ALTER TABLE Student ADD Address VARCHAR(50);
```

3.5 Not Good for Unstructured Data

Images, videos, large files → not ideal

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

4. ACID PROPERTIES (VERY IMPORTANT)

These ensure **reliable and safe transactions**

4.1 Atomicity (All or Nothing)

Idea:

Transaction either:

Fully completes

Or fully fails

Example:

Bank transfer:

Debit → Credit

If one fails → rollback

4.2 Consistency

Idea:

Database remains valid after transaction

Example:

Balance cannot become negative (if rule exists)

DATABASE CONCEPTS

4.3 Isolation

Idea:

Transactions run independently

Example:

Two users updating same account → no conflict

4.4 Durability

Idea:

Once committed → data is permanent

Even if:

System crashes

Power failure

SQL Example:

BEGIN TRANSACTION;

```
UPDATE Account SET Balance = Balance - 100 WHERE ID
= 1;
UPDATE Account SET Balance = Balance + 100 WHERE ID
= 2;

COMMIT;
```

DATABASE CONCEPTS

5. ER MODEL (ENTITY-RELATIONSHIP MODEL)

A **conceptual design tool** used before creating database

Why Needed:

Helps visualize structure

Avoids mistakes before implementation

5.1 ENTITY

Definition:

Real-world object stored in database

Examples:

Student

Employee

Product

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

5.2 RELATIONSHIP

Connection between entities

Example:

Student → enrolls → Course

5.3 SPECIALIZATION

Breaking entity into sub-entities

Example:

Employee →

Manager

Engineer

Specialization = **top-down approach**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

6. KEY TERMS (VERY IMPORTANT FOR THEORY QUESTIONS)

6.1 ATTRIBUTE

Column in a table

Example:

Name, Age, ID

6.2 TUPLE

Row in a table

(1, Aman, 20)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATABASE CONCEPTS

6.3 DOMAIN

Set of allowed values

Example:

Age → 0 to 120

Gender → Male/Female

6.4 DEGREE

Number of attributes (columns)

Example:

Table has 3 columns → Degree = 3

DATABASE CONCEPTS

6.5 CARDINALITY

Number of rows (tuples)

Example:

5 rows → Cardinality = 5

STRUCTURED QUERY LANGUAGE

1. INTRODUCTION TO SQL

SQL (Structured Query Language) is used to:

Store data

Retrieve data

Modify data

Manage databases

It works on **relational databases (tables)**.

Think SQL as a language to “talk” to database:

“Give me all students” , “Add a new record” , “Update marks”

Example:

```
SELECT * FROM Students;
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2. MYSQL VS SQL SERVER

MySQL

Open-source

Developed by Oracle

Used in web apps

SQL Server

Commercial

Developed by Microsoft

Used in enterprise systems

Insight:

Both follow SQL but differ in:

Features

Performance

Licensing

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

3. ADVANTAGES OF SQL

3.1 Data Integrity

Ensures correct and consistent data And Uses constraints (Primary Key, Foreign Key)

3.2 Efficient Data Retrieval

Fast queries even on large data

SELECT Name FROM Students WHERE Age > 20;

3.3 Data Security

Access control , Permissions

GRANT SELECT ON Students TO user1;

3.4 Versatility

Supports different data types:

Numbers , Text , Date/Time

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

4. OPERATIONS ON RELATIONS

These are based on relational algebra but implemented in SQL.

4.1 UNION

Purpose:

Combines results of two queries (removes duplicates)

```
SELECT Name FROM A
UNION
SELECT Name FROM B;
```

STRUCTURED QUERY LANGUAGE

4.2 INTERSECTION

Purpose:

Common rows in both tables

```
SELECT Name FROM A
INTERSECT
SELECT Name FROM B;
```

4.3 MINUS (EXCEPT)

Purpose:

Rows in first table but not in second

```
SELECT Name FROM A
EXCEPT
SELECT Name FROM B;
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

4.4 CARTESIAN PRODUCT

```
SELECT * FROM A, B;
```

Every row combines with every row

4.5 JOIN

Already deeply covered earlier, but key idea:

Combine related tables using common column

STRUCTURED QUERY LANGUAGE

5. SQL CLAUSES (VERY IMPORTANT)

Clauses control how query behaves.

5.1 GROUP BY

Purpose:

Groups rows based on column

Example:

```
SELECT Department, AVG(Salary)
FROM Employees
GROUP BY Department;
```

Output:

```
Dept   | Avg Salary
CSE    | 40000
ECE    | 35000
```

STRUCTURED QUERY LANGUAGE

5.2 HAVING

Purpose:

Filters grouped data (used after GROUP BY)

Example:

```
SELECT Department, AVG(Salary)
FROM Employees
GROUP BY Department
HAVING AVG(Salary) > 30000;
```

WHERE	HAVING
Filters rows	Filters groups
Before GROUP BY	After GROUP BY

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

5.3 ORDER BY

Purpose:

Sort results

Example:

```
SELECT Name, Salary
FROM Employees
ORDER BY Salary DESC;
```

Output:

Highest salary first

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

6. DATA TYPES IN SQL

6.1 INTEGER / INT

Use:

Whole numbers

Age INT

6.2 VARCHAR(n)

Use:

Variable-length text

Name VARCHAR(50)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

6.3 CHAR(n)

Use:

Fixed-length text

```
Code CHAR(5)
```

Difference:

CHAR → fixed

VARCHAR → flexible

6.4 DECIMAL(p,s)

Use:

Exact numeric values

```
Salary DECIMAL(10,2)
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

6.5 FLOAT

Use:

Approximate values

```
Marks FLOAT
```

6.6 DATE

```
DOB DATE
```

Format:

YYYY-MM-DD

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

6.7 TIME

```
Time TIME
```

Format:

HH:MM:SS

6.8 BOOLEAN

```
IsActive BOOLEAN
```

Values:

TRUE / FALSE

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

6.7 TIME

```
Time TIME
```

Format:

HH:MM:SS

6.8 BOOLEAN

```
IsActive BOOLEAN
```

Values:

TRUE / FALSE

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

WHAT ARE SQL FUNCTIONS?

SQL functions are **predefined operations** used to:

Perform calculations

Manipulate data

Extract information

Types :

Math Functions

Text Functions

Date Functions

Aggregate Functions

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2. MATH FUNCTIONS

Used for **numeric calculations**

2.1 POWER(x, y)

Purpose:

Returns **x raised to power y**

```
SELECT POWER(2, 3);
```

Output:

8

Concept:

$2^3 = 2 \times 2 \times 2 = 8$

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2.2 ROUND(x, n)

Purpose:

Rounds number to **n decimal places**

```
SELECT ROUND(3.14159, 2);
```

Output:

3.14

Insight:

n = 0 → nearest integer

n = 2 → 2 decimal places

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2.3 MOD(x, y)

Purpose:

Returns remainder

```
SELECT MOD(10, 3);
```

Output:

1

Concept:

10 ÷ 3 → remainder = 1

STRUCTURED QUERY LANGUAGE

3. TEXT FUNCTIONS (VERY IMPORTANT)

Used to manipulate strings

3.1 UPPER() / UCASE()

Purpose:

Convert to uppercase

```
SELECT UPPER('aman');
```

Output:

AMAN

STRUCTURED QUERY LANGUAGE

3.2 LOWER() / LCASE()

```
SELECT LOWER('AMAN');
```

Output:

aman

3.3 SUBSTRING() / MID()

Purpose:

Extract part of string

```
SELECT SUBSTRING('DATABASE', 1, 4);
```

Output:

DATA

Explanation:

Start at position 1 → take 4 characters

STRUCTURED QUERY LANGUAGE

3.4 LENGTH()

```
SELECT LENGTH('SQL');
```

Output:

3

3.5 LEFT()

```
SELECT LEFT('DATABASE', 4);
```

Output:

DATA

STRUCTURED QUERY LANGUAGE

3.6 RIGHT()

```
SELECT RIGHT('DATABASE', 4);
```

Output:

BASE

3.7 INSTR()

Purpose:

Find position of substring

```
SELECT INSTR('DATABASE', 'BASE');
```

Output:

5

STRUCTURED QUERY LANGUAGE

3.8 LTRIM()

```
SELECT LTRIM('   SQL');
```

Output:

SQL

3.9 RTRIM()

```
SELECT RTRIM('SQL   ');
```

Output:

SQL

STRUCTURED QUERY LANGUAGE

3.10 TRIM()

```
SELECT TRIM('   SQL   ');
```

Output:

SQL

STRUCTURED QUERY LANGUAGE

4. DATE FUNCTIONS

4.1 NOW()

```
SELECT NOW();
```

Output:

Current date and time

4.2 DATE()

```
SELECT DATE(NOW());
```

Output:

Only date part

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

4. DATE FUNCTIONS

4.1 NOW()

```
SELECT NOW();
```

Output:

Current date and time

4.2 DATE()

```
SELECT DATE(NOW());
```

Output:

Only date part

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

4.3 MONTH()

```
SELECT MONTH('2025-04-23');
```

Output:

4

4.4 MONTHNAME()

```
SELECT MONTHNAME('2025-04-23');
```

Output:

April

4.5 YEAR()

```
SELECT YEAR('2025-04-23');
```

Output:

2025

STRUCTURED QUERY LANGUAGE

4.6 DAY()

```
SELECT DAY('2025-04-23');
```

Output:

23

4.7 DAYNAME()

```
SELECT DAYNAME('2025-04-23');
```

Output:

Wednesday

STRUCTURED QUERY LANGUAGE

5. AGGREGATE FUNCTIONS (VERY IMPORTANT)

Used on **multiple rows → give single result**

Example Table:

Salary Table

ID | Salary

---

1 | 30000

2 | 40000

3 | 50000

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

5.1 MAX()

```
SELECT MAX(Salary) FROM Employees;
```

Output:

50000

5.2 MIN()

```
SELECT MIN(Salary) FROM Employees;
```

Output:

30000

STRUCTURED QUERY LANGUAGE

5.3 AVG()

```
SELECT AVG(Salary) FROM Employees;
```

Output:

40000

5.4 SUM()

```
SELECT SUM(Salary) FROM Employees;
```

Output:

120000

STRUCTURED QUERY LANGUAGE

5.5 COUNT()

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

Output:

3

STRUCTURED QUERY LANGUAGE

1. WHAT ARE DBMS LANGUAGES?

DBMS languages are used to **interact with the database**:

Create structure

Insert/update/delete data

Retrieve data

Control access

Types:

DDL (Data Definition Language)

DML (Data Manipulation Language)

DQL (Data Query Language)

DCL (Data Control Language)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2. DDL (DATA DEFINITION LANGUAGE)

Core Idea:

Used to **define and manage database structure (schema)**

Important Concept: META DATA

Meta data = **data about data**

Example:

Table name

Column names

Data types

Constraints

Stored in **data dictionary**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2.1 CREATE

Create database or table

Create Database:

```
CREATE DATABASE College;
```

Create Table:

```
CREATE TABLE Student (
    ID INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT
);
```

Behavior:

Table structure is defined

Memory allocated

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2.2 ALTER TABLE

Purpose:

Modify existing table

Add Column:

```
ALTER TABLE Student ADD Address VARCHAR(100);
```

Add Primary Key:

```
ALTER TABLE Student ADD PRIMARY KEY (ID);
```

Modify Column:

```
ALTER TABLE Student MODIFY Age INT;
```

Insight:

ALTER changes **structure, not data**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

2.3 DROP TABLE

Purpose:

Delete entire table permanently

DROP TABLE Student;

Behavior:

Structure + data both removed

Cannot be recovered easily

STRUCTURED QUERY LANGUAGE

3. DML (DATA MANIPULATION LANGUAGE)

Core Idea:

Used to **modify data inside tables**

3.1 INSERT

Purpose:

Add new records

```
INSERT INTO Student (ID, Name, Age)
VALUES (1, 'Aman', 20);
```

Multiple Insert:

```
INSERT INTO Student VALUES
(2, 'Ravi', 22),
(3, 'Sita', 19);
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

3.2 UPDATE

Purpose:

Modify existing data

```
UPDATE Student
SET Age = 21
WHERE ID = 1;
```

Important:

Without WHERE → all rows updated

STRUCTURED QUERY LANGUAGE

3.3 DELETE

Purpose:

Remove records

```
DELETE FROM Student
WHERE ID = 2;
```

Important:

DELETE FROM Student;

→ Deletes all rows but table remains

STRUCTURED QUERY LANGUAGE

4. DQL (DATA QUERY LANGUAGE)

Core Idea:

Used to **retrieve data**

Main Command: SELECT

4.1 SELECT

```
SELECT Name FROM Student;
```

4.2 FROM

Specifies table

```
SELECT * FROM Student;
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

4.3 WHERE

Filters rows

```
SELECT * FROM Student WHERE Age > 20;
```

Combined Query:

```
SELECT Name, Age
FROM Student
WHERE Age > 20;
```

STRUCTURED QUERY LANGUAGE

5. DCL (DATA CONTROL LANGUAGE)

Core Idea:

Controls **permissions and security**

5.1 GRANT

Purpose:

Give access to users

```
GRANT SELECT ON Student TO user1;
```

Meaning:

User1 can read data from Student table

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

STRUCTURED QUERY LANGUAGE

5.2 REVOKE

Purpose:

Remove access

```
REVOKE SELECT ON Student FROM user1;
```

Meaning:

User1 can no longer access table

COMPUTER NETWORK

1. INTRODUCTION TO COMPUTER NETWORK

A computer network is a system where **multiple devices are connected** to:

Share data

Share resources

Communicate

Examples:

Internet

School computer lab

ATM network

Why Networks are Important:

Fast communication

Resource sharing (printer, internet)

Collaboration (email, cloud)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

2. EVOLUTION OF NETWORKING

Networking began in the **1960s with ARPANET**

Developed by U.S. Department of Defense

Connected research institutions

Goal → share information

Insight:

ARPANET → evolved into today's **Internet**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

3. TYPES OF COMPUTER NETWORK

Type	Full Form	Area	Speed	Cost
PAN	Personal Area Network	10 m	High	Low
LAN	Local Area Network	Building	High	Low
MAN	Metropolitan Area Network	City	Medium	Medium
WAN	Wide Area Network	Country/World	Low	High

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

3.1 PAN

Bluetooth

Mobile hotspot

3.2 LAN

School lab

Office network

3.3 MAN

City-wide cable network

3.4 WAN

Internet

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4. NETWORK TOPOLOGIES (VERY IMPORTANT)

Topology = **arrangement of network devices**

4.1 BUS TOPOLOGY

Structure:

Single main cable (backbone)

Working:

Data travels in both directions

Advantages:

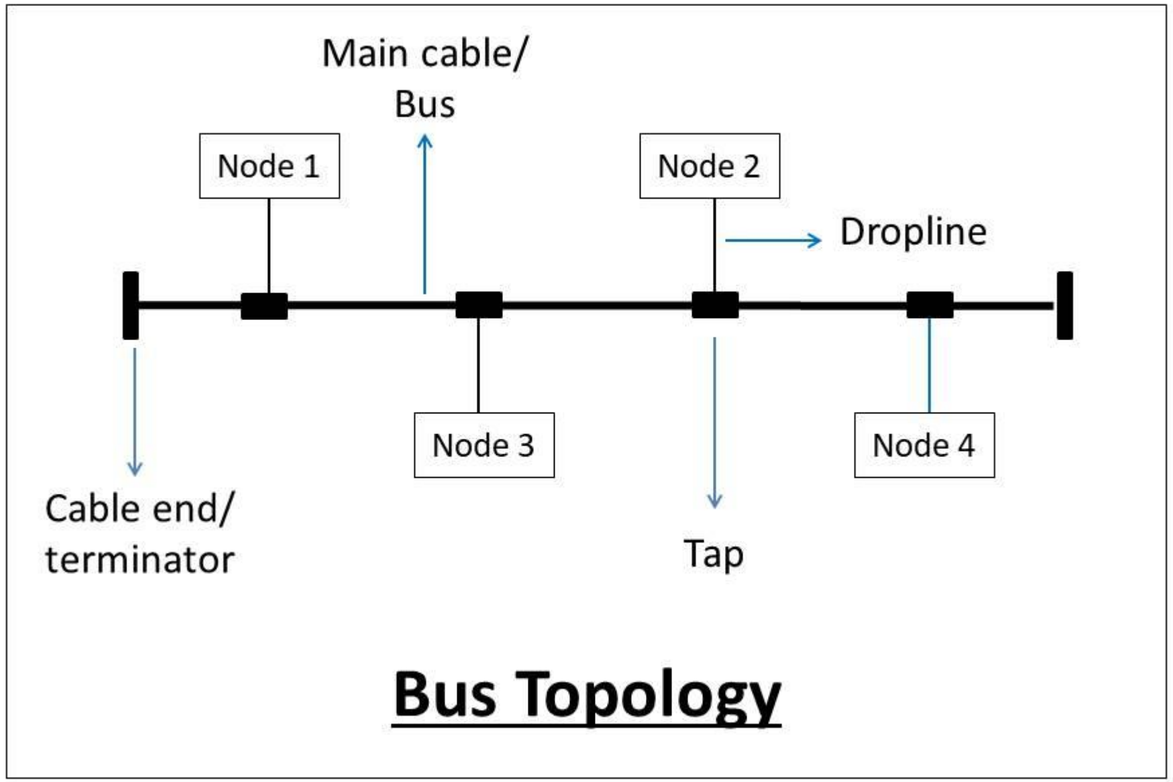
Simple

Low cost

Disadvantages:

If cable fails → entire network fails

Limited devices



COMPUTER NETWORK

4.2 STAR TOPOLOGY

Structure:

All devices connected to central hub/switch

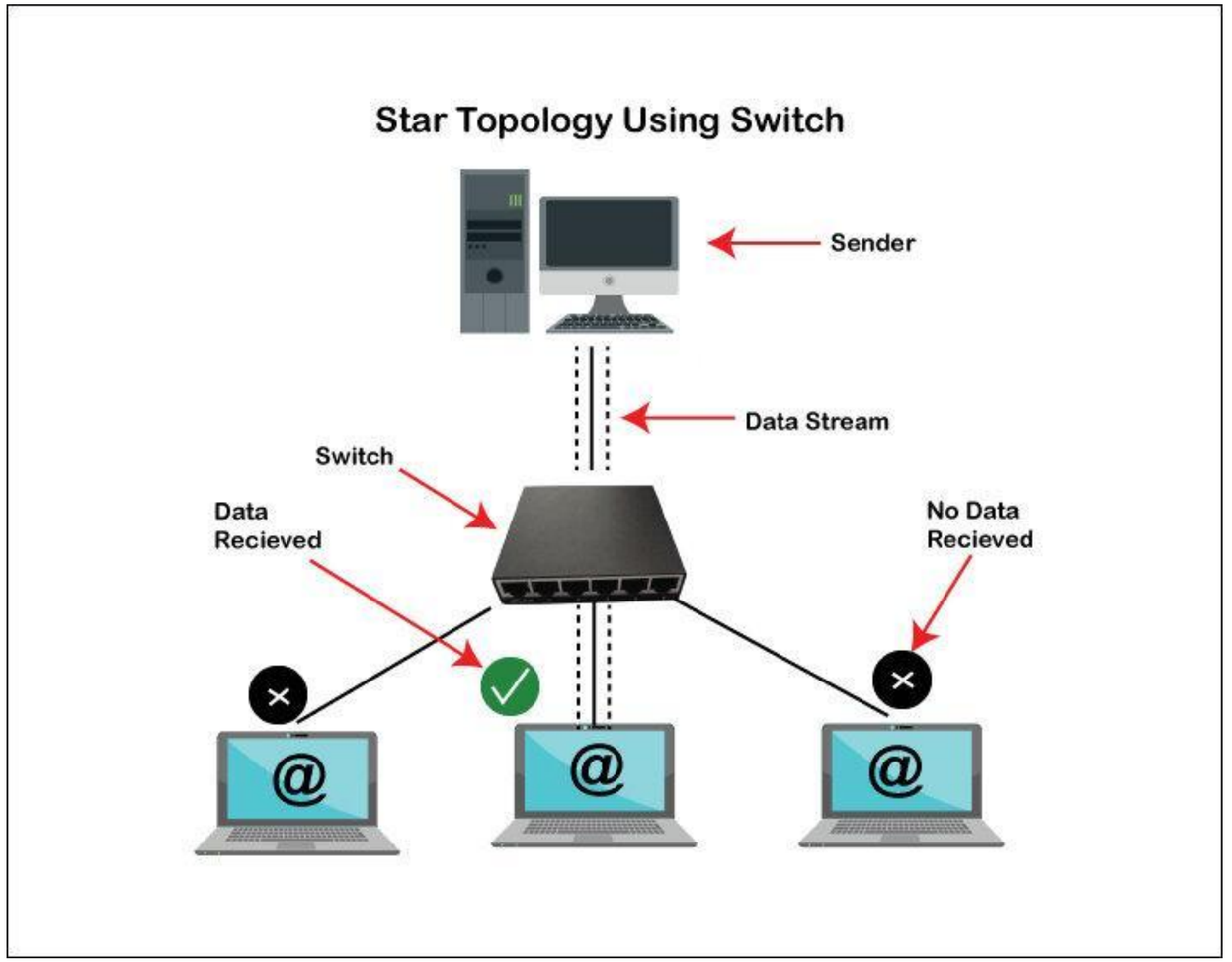
Advantages:

Easy to manage

Failure of one device doesn't affect others

Disadvantages:

Hub failure → entire network fails



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4.3 RING TOPOLOGY

Structure:

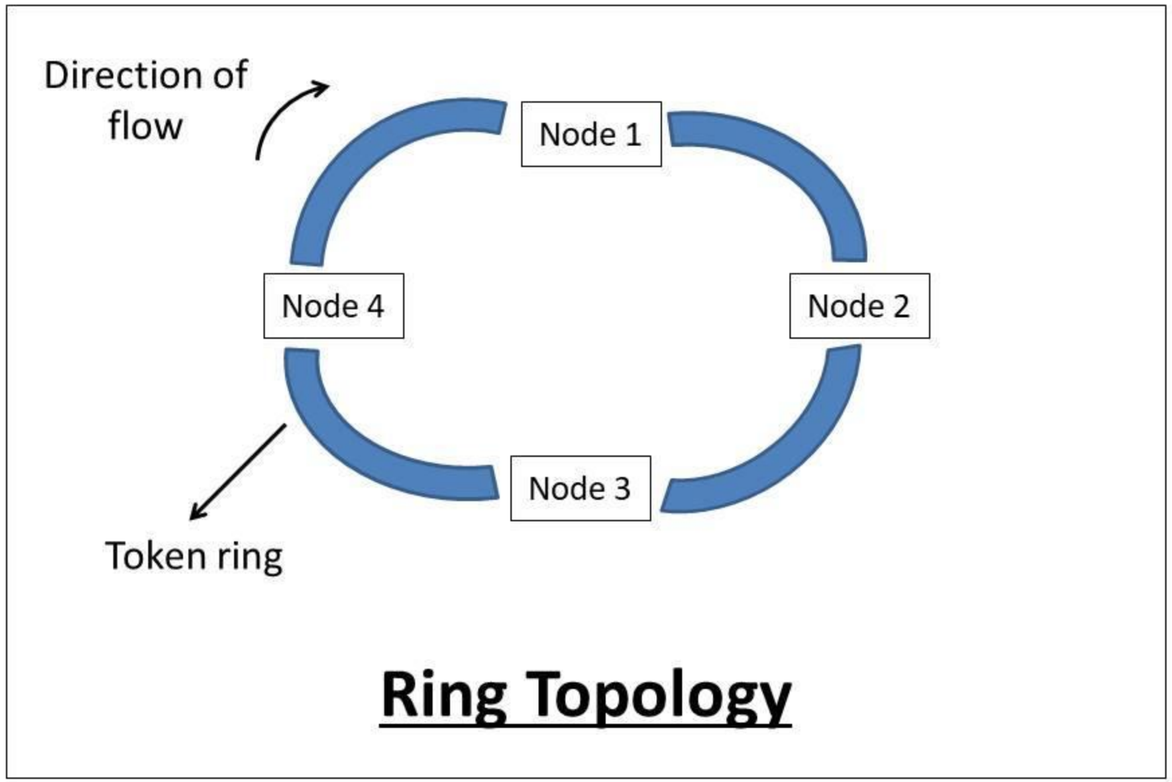
Devices form a circular loop

Working:

Data moves in one direction

Disadvantages:

One failure → entire network affected



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4.4 MESH TOPOLOGY

Structure:

Every device connected to every other device

Advantages:

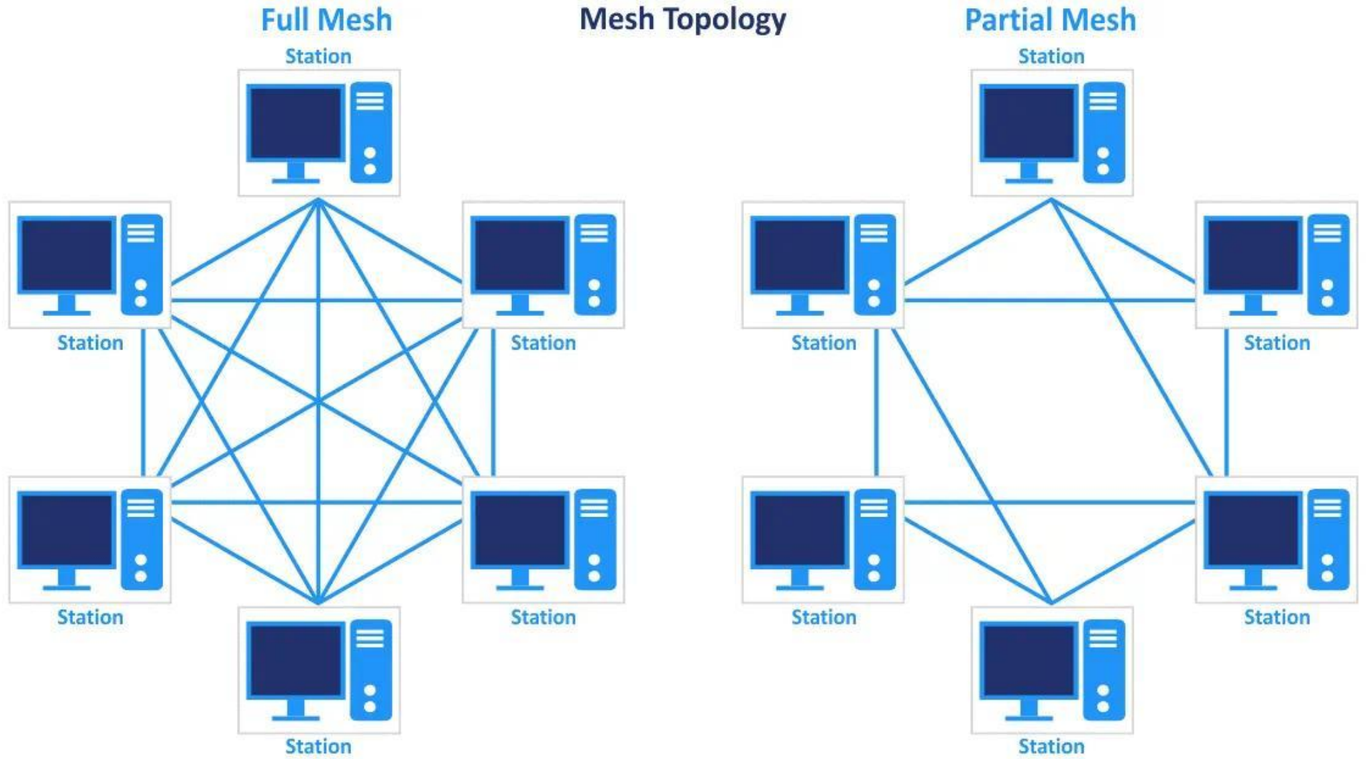
Highly reliable

No data loss

Disadvantages:

Expensive

Complex



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

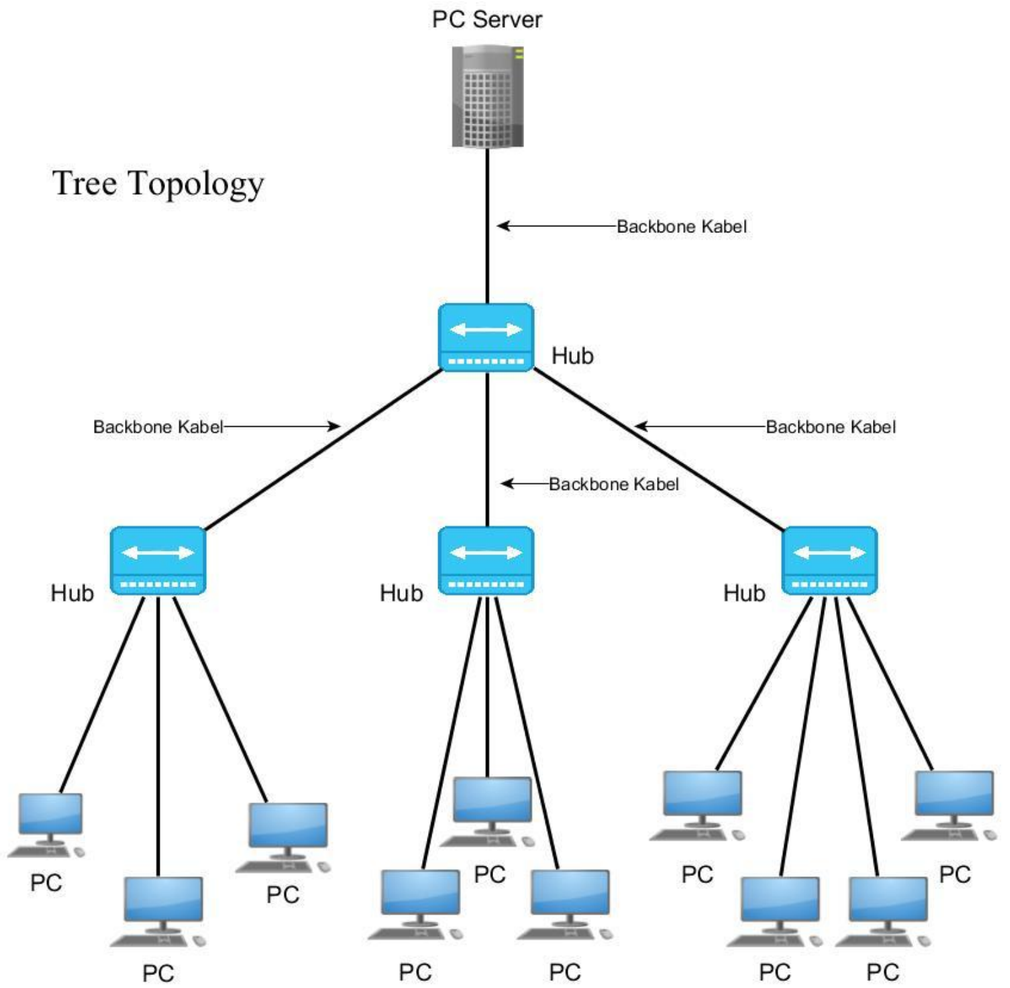
4.5 TREE TOPOLOGY

Structure:

Hierarchical (like tree)

Insight:

Combination of **star + bus topology**

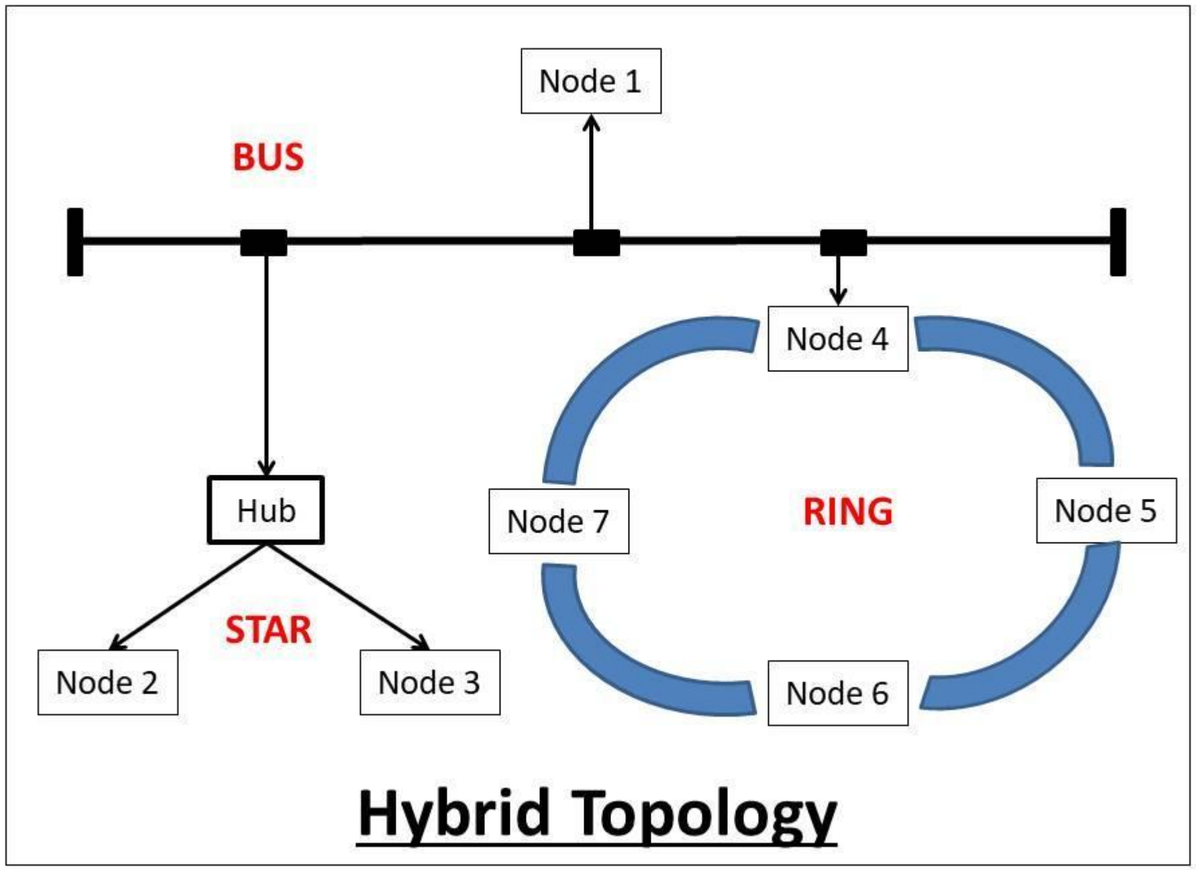


By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4.6 HYBRID TOPOLOGY

A hybrid topology is a network topology formed by combining two or more different types of network topologies into a single network.



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

5. NETWORK DEVICES (VERY IMPORTANT)

5.1 MODEM

Function:

Converts:

Digital → Analog

Analog → Digital

Use:

Connect to ISP (internet)

5.2 ETHERNET CARD (NIC)

Function:

Connects computer to LAN

Insight:

Each card has unique MAC address

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

5.3 REPEATER

Function:

Boosts weak signals

Works at:

OSI Layer 1

5.4 HUB

Function:

Connects multiple devices

Sends data to all devices

Problem:

No intelligence (broadcasts blindly)

COMPUTER NETWORK

5.5 SWITCH

Function:

Sends data to specific device

Advantage over Hub:

More efficient

Uses MAC address

5.6 ROUTER

Function:

Connects different networks

Example:

Your home WiFi router

Works at:

OSI Layer 3

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

5.7 GATEWAY

Function:

Connects different protocols/networks

Example:

Internet gateway

5.8 RJ45 CONNECTOR

Function:

Used in Ethernet cables

Features:

8 pins

Used in LAN

COMPUTER NETWORK

Device	Function	Layer
Repeater	Boost signal	Layer 1
Hub	Broadcast data	Layer 1
Switch	Smart data transfer	Layer 2
Router	Connect networks	Layer 3
Gateway	Protocol conversion	All layers

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

INTERNET

The Internet is a **global network of interconnected computers and servers** that communicate using standard protocols like **TCP/IP**.

What this really means:

Millions of devices are connected worldwide

These devices communicate using rules (protocols)

Data travels in the form of packets

Key Components Explained:

1. Devices:

Computers, mobiles, servers

These are **end systems**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

2. Routers:

Direct data from one network to another

Decide the best path

3. Communication Media:

Wired → Fiber, cables

Wireless → WiFi, satellite

Purpose of Internet:

Data exchange

Communication (email, chat)

Information access (Google, websites)

COMPUTER NETWORK

Real Working Example:

When you search something:

Your device sends request

It travels through routers

Server processes request

Response comes back

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

WORLD WIDE WEB (WWW)

WWW is **not the Internet itself** — it is a **service running on the Internet**

A collection of **interconnected web pages and resources** accessed through the Internet.

Important Concept:

Internet = Infrastructure

WWW = Content (websites)

Invented by:

Tim Berners-Lee (1990)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

Key Technologies:

2.1 HTML (HyperText Markup Language)

Used to create web pages

2.2 URL (Uniform Resource Locator)

Address of a webpage

Example:

https://www.google.com

2.3 HTTP (HyperText Transfer Protocol)

Rule for transferring web pages

COMPUTER NETWORK

Flow of Web Access:

You enter URL

Browser sends HTTP request

Server sends webpage

Browser displays it

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

3. MAC ADDRESS (PHYSICAL ADDRESS)

A MAC address is a **unique hardware identifier** assigned to a network device.

Structure:

`48 bits (12 hexadecimal digits)`

`Example:`

`00:1A:2B:3C:4D:5E`

Breakdown:

First 6 digits → Manufacturer (OUI)

Last 6 digits → Device unique number

COMPUTER NETWORK

Important Properties:

Fixed (does not change)

Assigned by manufacturer

Works at **Data Link Layer (Layer 2)**

Real Understanding:

MAC address = **Permanent identity of device**

COMPUTER NETWORK

4. IP ADDRESS (LOGICAL ADDRESS)

Definition:

An IP address is a **unique address assigned to a device on a network** for communication.

Key Difference from MAC:

MAC → Permanent

IP → Can change

Why IP is needed:

To identify device location in network

To send/receive data

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4.1 IPv4

Structure:

32-bit address

Written as 4 numbers

Example:

192.168.1.1

Range:

Each number → 0 to 255

Problem:

Limited addresses (~4.3 billion)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

4.2 IPv6

Structure:

128-bit address

Example:

2001:0db8:85a3::8a2e:0370:7334

Advantage:

Huge number of addresses

Solves IPv4 shortage

COMPUTER NETWORK

MAC vs IP ADDRESS

Feature	MAC Address	IP Address
Type	Physical	Logical
Assigned by	Manufacturer	Network
Changeable	No	Yes
Format	Hexadecimal	Decimal (IPv4)
Layer	Data Link (L2)	Network (L3)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

6. HOW INTERNET + IP + MAC WORK TOGETHER

This is the **most important conceptual understanding**

Step-by-Step Data Transfer:

You enter a website

Your device uses **IP address** to locate destination

Data travels through routers

Inside local network → MAC address is used

Data reaches correct device

Insight:

IP → “Where to go”

MAC → “Who exactly”

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

7. FINAL CONNECTION (VERY IMPORTANT)

Internet → global network

WWW → service on internet

IP → identifies device location

MAC → identifies physical device

COMPUTER NETWORK

OSI MODEL (VERY IMPORTANT CONCEPT)

Definition:

OSI (Open Systems Interconnection) Model is a **7-layer framework** used to understand how data travels in a network.

Why OSI Model Exists:

Standardizes communication

Makes troubleshooting easier

Separates network functions

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

7 LAYERS (TOP TO BOTTOM)

1. Application Layer (Layer 7)

Function:

Interface between user and network

Examples:

Browser (Chrome)

Email apps

Concept:

User interacts here

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

2. Presentation Layer (Layer 6)

Function:

Converts data format

Encryption / Decryption

Example:

Converting text → encrypted format

3. Session Layer (Layer 5)

Function:

Manages connection (session)

Example:

Login session in a website

COMPUTER NETWORK

4. Transport Layer (Layer 4)

Function:

End-to-end communication

Error control

Flow control

Protocol:

TCP / UDP

5. Network Layer (Layer 3)

Function:

Routing (path selection)

Device:

Router

COMPUTER NETWORK

6. Data Link Layer (Layer 2)

Function:

Node-to-node communication

Uses MAC address

7. Physical Layer (Layer 1)

Function:

Actual transmission of bits

Hardware (cables, signals)

FLOW:

Sender → Layer 7 → Layer 1 → Transmission → Receiver → Layer 1 → Layer 7

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

WEBSITE & WEB PAGES

WEBSITE

A website is a **collection of related web pages** under one domain.

Example:

amazon.com

google.com

WEB PAGE

A single document within a website

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

TYPES OF WEB PAGES

1. Static Web Page

Features: - Fixed content , Same for all users

Example:- Basic HTML page

2. Dynamic Web Page

Features:

Content changes , Depends on user interaction

Technologies:

JavaScript

PHP

Example: Instagram feed , YouTube homepage

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

WEB BROWSERS

Software used to **access and display web pages**

Examples:

Chrome

Firefox

Safari

Features:

Tabs

Bookmarks

Privacy mode

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

BROWSER SETTINGS & TOOLS

1. Add-ons / Plugins

Purpose:

Extend browser functionality

Example:

Ad blocker

2. Browser Settings

Used for:

Privacy

Security

Homepage

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

3. Cookies

Definition:

Small data stored in browser

Purpose:

Remember login

Personalization

Important:

Cookies help in:

Sessions

Tracking

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

WEB SERVERS

Definition:

A web server is a system that **stores and delivers web pages**

Function:

Receives request

Sends response

Example Flow:

Browser sends request

Server processes

Server sends webpage

Types:

Apache

NGINX

IIS

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

WEB HOSTING

Definition:

Service that stores websites on servers

Purpose:

Makes website accessible online

COMPUTER NETWORK

WEB HOSTING

Definition:

Service that stores websites on servers

Purpose:

Makes website accessible online

COMPUTER NETWORK

WEB SERVICES (VERY IMPORTANT)

5.1 DNS (Domain Name System)

Function:

Converts domain name → IP address

Example:

google.com → 142.250.183.14

Insight:

Human-friendly → Machine-readable

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

EMAIL (Electronic Mail)

Function:

Send and receive messages

Protocols:

SMTP → Sending

POP3 → Receiving

IMAP → Receiving (advanced)

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

COMPUTER NETWORK

5.3 CHAT

Function:

Real-time communication

Examples:

WhatsApp

Telegram

COMPUTER NETWORK

5.4 VoIP (Voice over Internet Protocol)

Function:

Voice/video calls over internet

Examples:

Zoom

Skype

Google Meet

Advantage:

Cheap communication

Global connectivity

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

1. What is an Exception?

An **exception** is an event that occurs during the execution of a program that **disrupts the normal flow of instructions**.

Simple meaning:

An error that happens **while the program is running (runtime)**.

Example:

```
print(10 / 0)
```

Output:

```
ZeroDivisionError
```

Here, the program stops because Python cannot divide by zero.

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

2. Types of Errors in Python

(1) Syntax Errors

Occur when Python rules (syntax) are not followed

Detected before execution

Example:

```
if x > 5   # Missing colon
    print("Hello")
```

(2) Runtime Errors (Exceptions)

Occur during execution

These are called **exceptions**

Example:

```
x = 5 / 0
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

(3) Logical Errors

Program runs successfully but gives wrong output

Hardest to detect

Example:

```
print(2 * 2)   # If you expected 5 → logic error
```

EXCEPTION HANDLING

4. Common Built-in Exceptions

Exception	Description
ValueError	Wrong value type (e.g., string instead of integer)
TypeError	Operation on wrong data type
ZeroDivisionError	Division by zero
IndexError	Index out of range in list/tuple
NameError	Variable not defined
ImportError	Module cannot be imported
IOError	File-related error
EOFError	No input provided
IndentationError	Incorrect indentation
OverflowError	Number too large

EXCEPTION HANDLING

5. try – except Statement

Syntax:

```
try:
    # risky code
except:
    # error handling code
```

How it Works:

Python executes code inside try

If error occurs → jumps to except

If no error → except is skipped

Example:

```
try:
    x = int(input("Enter number: "))
    print(10 / x)
except:
    print("An error occurred")
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

6. Handling Specific Exceptions

You can handle different errors separately:

```
try:
    x = int(input("Enter number: "))
    print(10 / x)
except ValueError:
    print("Invalid input")
except ZeroDivisionError:
    print("Cannot divide by zero")
```

EXCEPTION HANDLING

7. try – except – else

Syntax:

```
try:
    # risky code
except:
    # error handling
else:
    # runs if no error
```

Example:

```
try:
    x = int(input("Enter number: "))
except:
    print("Error")
else:
    print("You entered:", x)
```

Explanation:

else executes **only when no exception occurs**

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

8. try – finally

Syntax:

```
try:
    # risky code
finally:
    # always runs
```

Example:

```
try:
    print("Running code")
finally:
    print("Always executed")
```

Explanation:

finally block executes **no matter what happens**

Used for cleanup tasks

(closing files, releasing resources)

EXCEPTION HANDLING

9. raise Statement

Used to **manually generate an exception**

Example:

```
age = -1
if age < 0:
    raise ValueError("Age cannot be negative")
```

Why use raise?

To enforce rules in program

To stop execution if condition is wrong

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

EXCEPTION HANDLING

10. assert Statement

Purpose:

Used for **debugging and testing conditions**

Syntax:

assert condition, "Error message"

Example:

```
x = 5
assert x > 0, "x must be positive"
```

Working:

If condition is True → nothing happens

If False → AssertionError

FILE HANDLING

1. What is a File?

Definition:

A **file** is a named location on storage used to **store data permanently**.

2. Why File Handling?

Store data permanently

Read saved data later

Handle large data efficiently

3. Opening a File

Syntax:

```
file = open("filename", "mode")
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

FILE HANDLING

4. File Modes

Mode	Meaning
'r'	Read only
'w'	Write (overwrites file)
'a'	Append data
'r+'	Read + Write
'rb'	Read binary
'wb'	Write binary

Important:

- 'w' deletes old content
- 'a' keeps old content

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

FILE HANDLING

5. Closing a File

Syntax:

```
file.close()
```

Why important?

Frees memory

Saves data properly

Best Method (with statement):

```
with open("file.txt", "r") as f:
    data = f.read()
```

File automatically closes

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

FILE HANDLING

6. Reading from File

read()

Reads full content

```
f.read()
```

readline()

Reads one line

```
f.readline()
```

readlines()

Returns list of lines

```
f.readlines()
```

FILE HANDLING

7. Writing to File

write()

```
f.write("Hello")
```

writelines()

```
f.writelines(["Line1\n", "Line2\n"])
```

8. File Pointer Operations

tell()

Returns current position

```
f.tell()
```

seek()

Moves pointer to specific position

```
f.seek(0)
```

9. Binary Files

Store data in binary format

Used for images, objects, etc.

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

PICKLE

1. What is Pickling?

Pickling = Converting Python objects → Binary format

Example:

```
import pickle
data = {"name": "Ashu", "age": 20}
with open("file.dat", "wb") as f:
    pickle.dump(data, f)
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

PICKLE

2. What is Unpickling?

Unpickling = Converting binary data → Python objects

Example:

```
import pickle
with open("file.dat", "rb") as f:
    data = pickle.load(f)
print(data)
```

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

1. DATA COMMUNICATION

Definition:

Data communication is the **process of exchanging information between two or more devices** through a transmission medium.

Real Understanding:

Whenever you:

Send a message on WhatsApp

Open a website

Download a file

→ Data is being transmitted from one device to another

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

2. COMPONENTS OF DATA COMMUNICATION

2.1 Sender

Definition:

Device that sends data

Examples:

Computer

Mobile

DATA COMMUNICATION

2.2 Receiver

Device that receives data

Examples:

Laptop

Printer

2.3 Message

Actual data being sent

Types:

Text

Image

Audio

Video

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

2.4 Transmission Medium

Definition:

Path through which data travels

Types:

Wired → Cable, fiber

Wireless → WiFi, Bluetooth

Full Flow:

Sender → Medium → Receiver

DATA COMMUNICATION

3. TYPES OF DATA COMMUNICATION

3.1 SIMPLEX

Definition:

One-way communication only

Example:

Keyboard → Computer

DATA COMMUNICATION

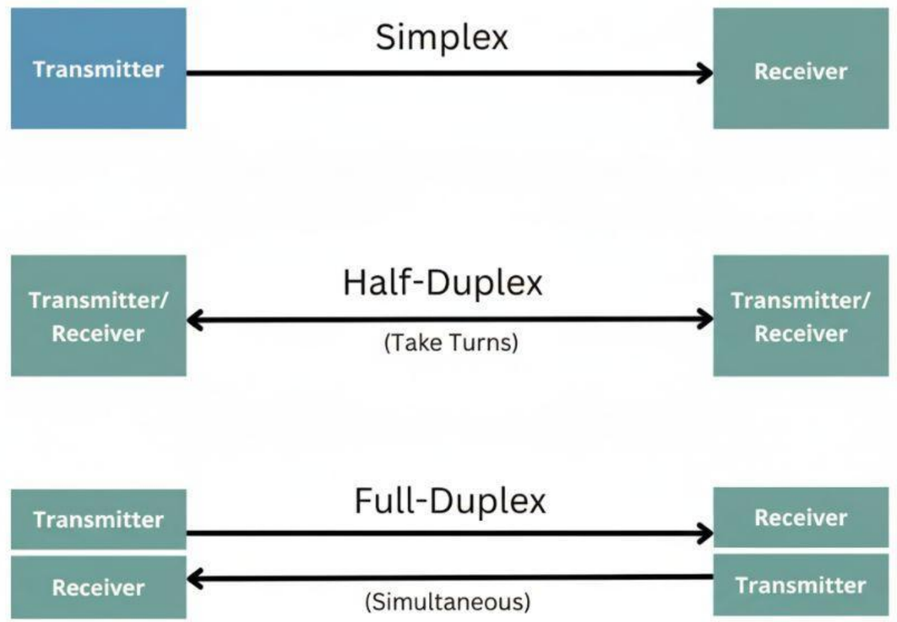
3.2 HALF-DUPLEX

Definition:

Two-way but not at the same time

Example:

Walkie-talkie



3.3 FULL-DUPLEX

Definition:

Two-way communication simultaneously

Example:

Phone call

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

NETWORK PROTOCOLS

Definition:

Protocols are **rules that govern communication between devices**

Why Needed:

Ensure proper data transfer

Avoid errors

Maintain structure

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

TYPES OF PROTOCOLS

HTTP

Purpose:

Transfer web pages

FTP

Purpose:

Transfer files

IP

Purpose:

Addressing + routing

PPP

Purpose:

Direct communication between two devices

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

CHANNEL, BANDWIDTH, DATA TRANSFER RATE

5.1 CHANNEL

Definition:

Path through which data is transmitted

Types:

Wired

Wireless

5.2 BANDWIDTH

Definition:

Amount of data a channel can carry

Units:

Hz, KHz, MHz

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

Insight:

Higher bandwidth → more data → faster communication

5.3 DATA TRANSFER RATE

Definition:

Actual speed of data transmission

Units:

bps

Kbps

Mbps

Gbps

Conversion:

1 Kbps = 1024 bps , 1 Mbps = 1024 Kbps

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

SWITCHING TECHNIQUES

Definition:

Methods used to transfer data from source to destination

6.1 CIRCUIT SWITCHING

Definition:

A dedicated path is established before communication

Example:

Traditional telephone

Features:

Fixed path

Reliable

Slow setup

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

6.2 PACKET SWITCHING

Definition:

Data is divided into packets and sent independently

Example:

Internet

Features:

Fast

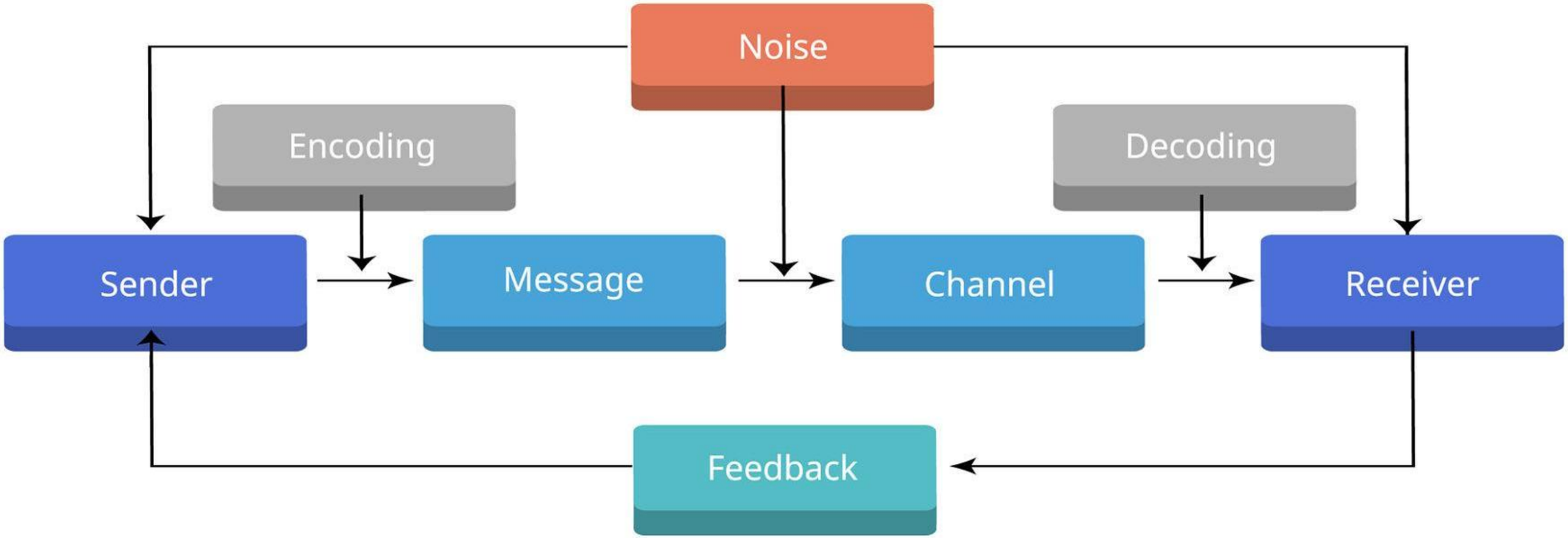
Efficient

No fixed path

Circuit Switching	Packet Switching
Fixed path	Dynamic path
Continuous transmission	Data in packets
Used in telephones	Used in Internet

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

COMMUNICATION MEDIA

Definition:

Communication media is the **path through which data travels** from sender to receiver.

Types:

Guided (Wired) Media

Unguided (Wireless) Media

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

2. GUIDED (WIRED) MEDIA

These use **physical cables**.

2.1 TWISTED PAIR CABLE

Structure:

Two insulated copper wires twisted together

Why twisting?

Reduces noise and interference

Uses:

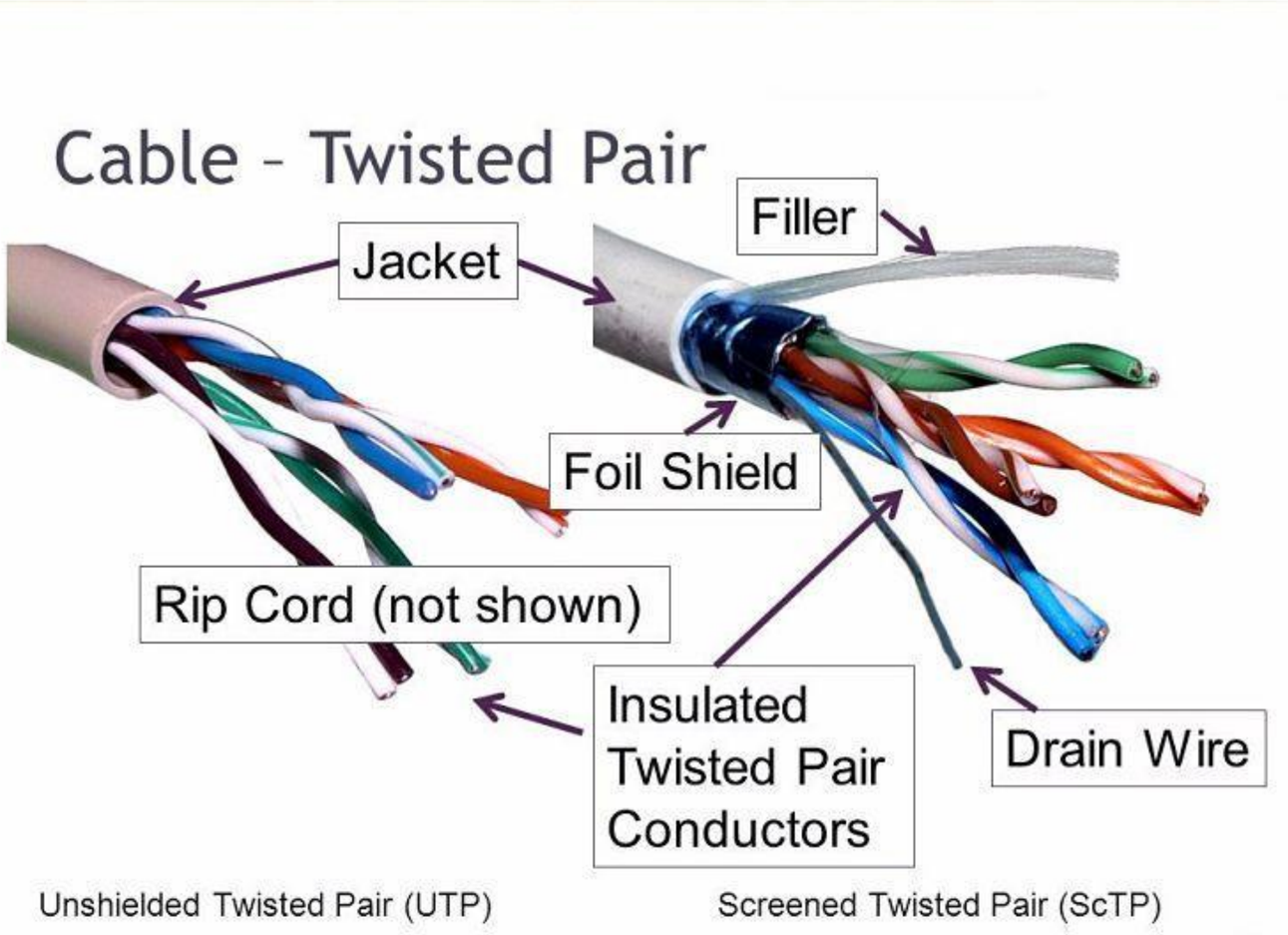
Telephone lines , LAN networks

Types:

UTP (Unshielded) , STP (Shielded)

Advantages: :-Cheap , Easy to install

Disadvantages: - Low bandwidth , Short distance



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

2.2 COAXIAL CABLE

Structure:

Central conductor

Insulation

Metallic shield

Uses:

Cable TV

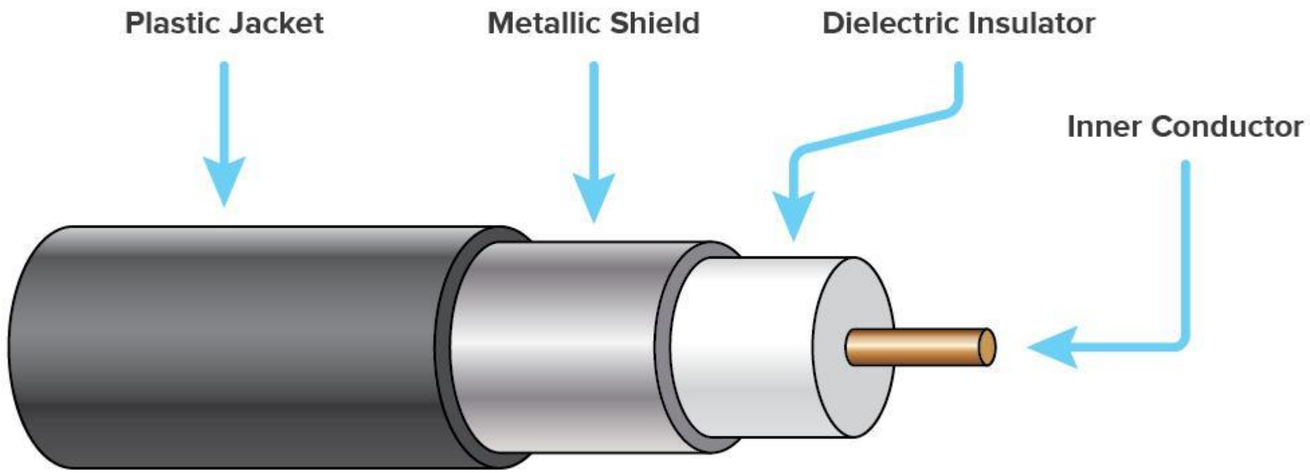
Internet

Advantages:

Better protection than twisted pair

Disadvantages:

More expensive



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

2.3 OPTICAL FIBRE

Structure:

Glass/plastic fiber

Working:

Data transmitted using **light signals**

Advantages:

Very high speed

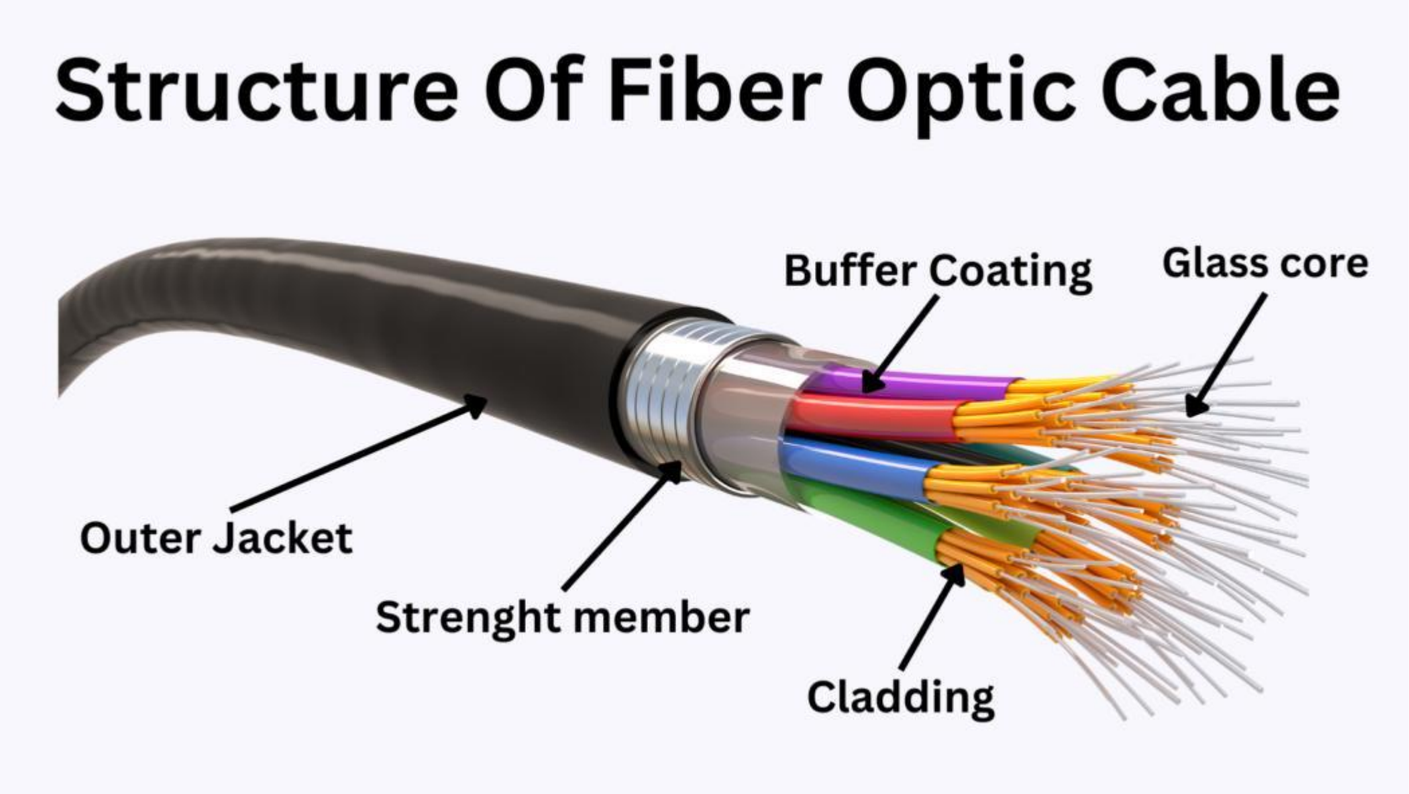
Long distance

No interference

Disadvantages:

Expensive

Difficult to install



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

3. UNGUIDED (WIRELESS) MEDIA

No physical wires → signals travel through air

3.1 RADIO WAVES

Frequency:

3 KHz – 1 GHz

Features:

Omni-directional

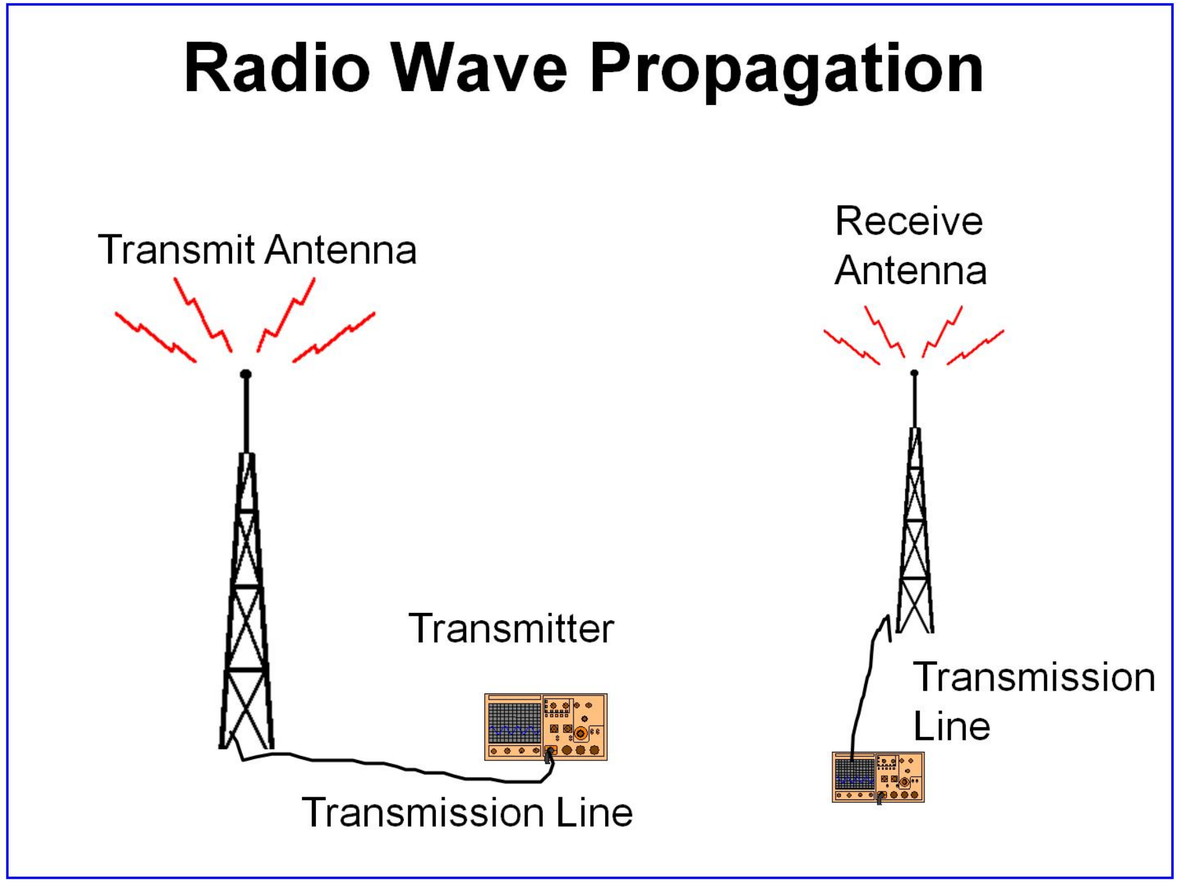
Travel long distances

Uses:

Radio

TV

Wireless communication



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

3.2 MICRO WAVES

Frequency:

1 GHz – 300 GHz

Features:

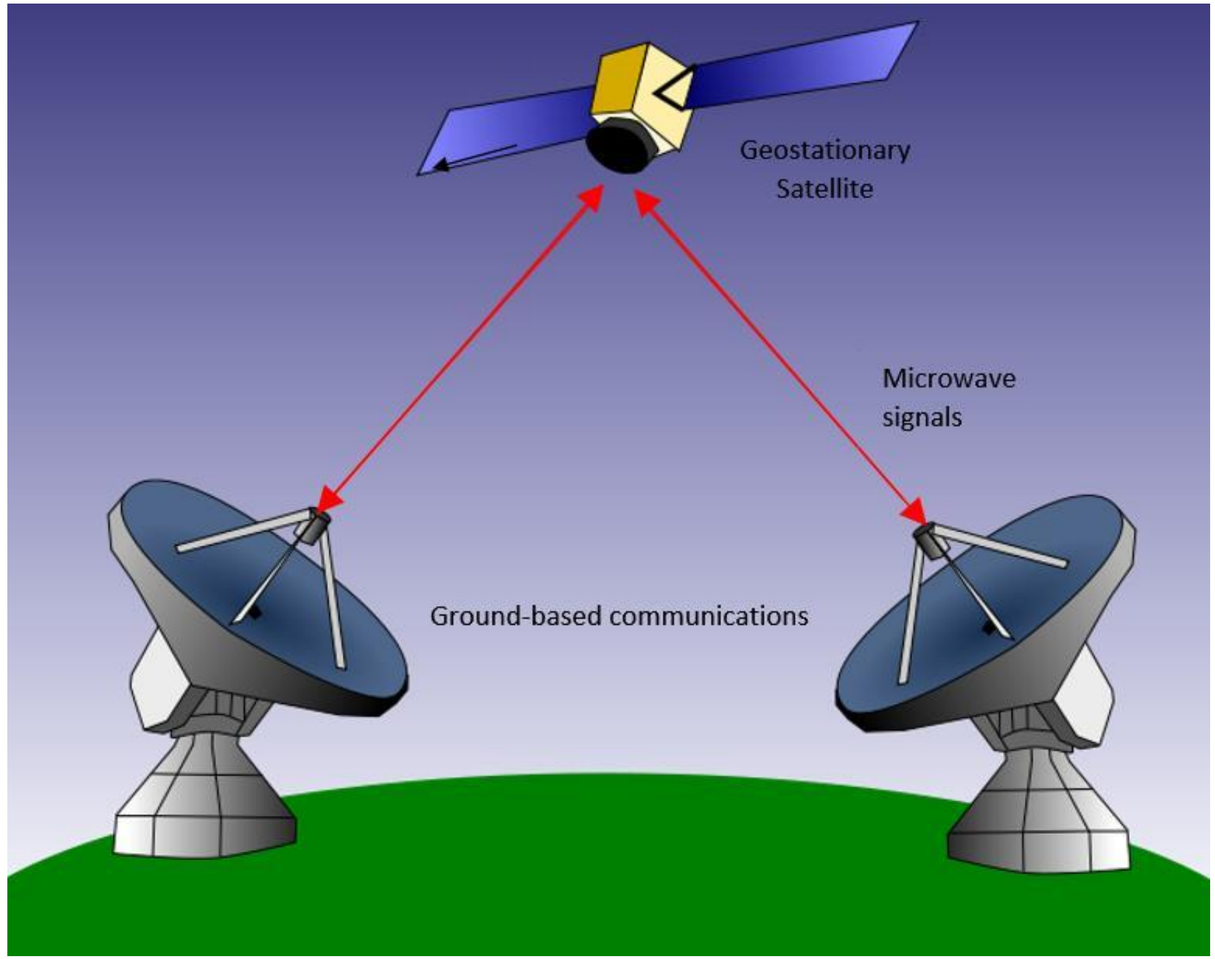
Uni-directional

Requires line-of-sight

Uses:

Satellite communication

Radar



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

3.3 INFRARED WAVES

Features:

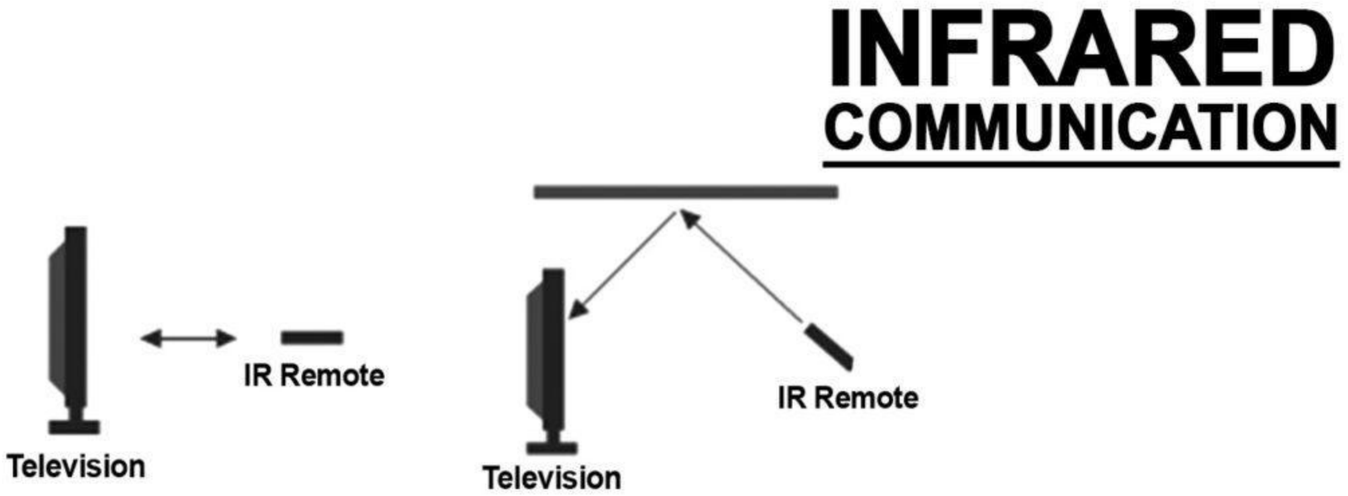
Short range

Cannot pass through walls

Uses:

Remote controls

Short-range communication



By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

WIRELESS TECHNOLOGIES

4.1 BLUETOOTH

Definition:

Short-range wireless communication

Range:

~10 meters

Uses:

Headphones

Keyboard

File sharing

Concept:

Forms a **PAN (Personal Area Network)**

DATA COMMUNICATION

4.2 WLAN (Wi-Fi)

Definition:

Wireless Local Area Network

Features:

No cables

Uses access point

Example:

Home WiFi

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

4.2 WLAN (Wi-Fi)

Definition:

Wireless Local Area Network

Features:

No cables

Uses access point

Example:

Home WiFi

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

MOBILE TECHNOLOGIES (VERY IMPORTANT)

1G (1982)

Features:

Analog signals

Voice only

2G (1991)

Features:

Digital signals

Better voice quality

3G (2001)

Features:

Internet + voice

4G

Features:

High-speed internet

Streaming, video calls

5G

Features:

Very high speed

Low latency

Future technologies

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

WIRED vs WIRELESS

Feature	Wired	Wireless
Medium	Cable	Air
Speed	High	Variable
Cost	Moderate	Low
Mobility	Low	High

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)

DATA COMMUNICATION

OPTICAL vs TWISTED PAIR

Feature	Optical Fiber	Twisted Pair
Speed	Very high	Low
Distance	Long	Short
Cost	High	Low
Feature	Optical Fiber	Twisted Pair

By :- Ashutosh Srivastav (State Topper B.T.E.U.P)