CREATE TABLE

CREATE TABLE Users (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Email VARCHAR(100) UNIQUE NOT NULL,

Password VARCHAR(255) NOT NULL);

FOREIGN KEY

CREATE TABLE Departments ( DepartmentID INT PRIMARY KEY AUTO\_INCREMENT,

DepartmentName VARCHAR(100) NOT NULL );

-- Creating the child table:

Employees with a foreign key

CREATE TABLE Employees ( EmployeeID INT PRIMARY KEY AUTO\_INCREMENT, FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

DepartmentID INT,

FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID) );

ALTER:

Add a Column:

ALTER TABLE Employees

ADD dob DATE;

Modify a Column:

ALTER TABLE Employees

MODIFY COLUMN dob DATETIME;

Rename a Column:

ALTER TABLE Employees

RENAME COLUMN dob TO birthdate;

Drop a Column:

ALTER TABLE Employees

DROP COLUMN birthdate;

Add a Constraint:

ALTER TABLE Employees

ADD CONSTRAINT unique\_email UNIQUE (email);

Drop a Constraint:

ALTER TABLE Employees

DROP CONSTRAINT unique\_email;

Rename a Table:

ALTER TABLE Employees

RENAME TO Staff;

Change Column Default Value:

ALTER TABLE Employees

ALTER COLUMN dob SET DEFAULT '2000-01-01';

Remove Column Default Value:

ALTER TABLE Employees

ALTER COLUMN dob DROP DEFAULT;

Update:

Basic Syntax:

UPDATE table\_name

SET column1 = value1, column2 = value2, ...

WHERE condition;

Update a Single Column:

UPDATE Employees

SET LastName = 'Doe'

WHERE EmployeeID = 1;

Update Multiple Columns:

UPDATE Employees

SET FirstName = 'John', LastName = 'Doe'

WHERE EmployeeID = 1;

Update All Rows:

UPDATE Employees

SET DepartmentID = 10;

Conditional Update:

UPDATE Employees

SET Salary = Salary \* 1.10

WHERE DepartmentID = (SELECT DepartmentID FROM Departments WHERE DepartmentName = 'IT');

DELETE:

Basic Syntax:

DELETE FROM table\_name

WHERE condition;

Delete a Single Row:

DELETE FROM Employees

WHERE EmployeeID = 1;

Delete Multiple Rows:

DELETE FROM Employees

WHERE DepartmentID = (SELECT DepartmentID FROM Departments WHERE DepartmentName = 'HR');

Delete All Rows:

DELETE FROM Employees;

TRUNCATE:

Basic Syntax:

TRUNCATE TABLE table\_name;

UNIQUE

Single Column Unique Constraint:

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100) UNIQUE,

DepartmentID INT,

Salary DECIMAL(10, 2)

);

Multi-Column Unique Constraint:

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

DepartmentID INT,

Salary DECIMAL(10, 2),

CONSTRAINT UniqueName UNIQUE (FirstName, LastName)

);

Add Unique Constraint to a Column:

ALTER TABLE Employees

ADD CONSTRAINT UniqueEmail UNIQUE (Email);

Add Unique Constraint to Multiple Columns:

ALTER TABLE Employees

ADD CONSTRAINT UniqueName UNIQUE (FirstName, LastName);

INSERT:

Inserting a Single Row:

INSERT INTO Employees (EmployeeID, FirstName, LastName, Email, DepartmentID, Salary)

VALUES (1, 'John', 'Doe', 'john.doe@example.com', 10, 50000);

Inserting Multiple Rows:

INSERT INTO Employees (EmployeeID, FirstName, LastName, Email, DepartmentID, Salary)

VALUES

(2, 'Jane', 'Smith', 'jane.smith@example.com', 20, 60000),

(3, 'Alice', 'Johnson', 'alice.johnson@example.com', 30, 55000);

Inserting Data into Specific Columns:

INSERT INTO Employees (FirstName, LastName, Email)

VALUES ('Bob', 'Brown', 'bob.brown@example.com');

ROLLBACK:

Basic Syntax:

ROLLBACK;

In SQL, the ROLLBACK statement is used to undo transactions that have not yet been committed to the database. This is useful for maintaining data integrity and ensuring that only valid data is stored.

Releasing a Savepoint:

RELEASE SAVEPOINT Savepoint1;

ORDER BY

Ordering by a Single Column:

SELECT \*

FROM Employees

ORDER BY Salary ASC;

Ordering by Multiple Columns:

SELECT \*

FROM Employees

ORDER BY DepartmentID ASC, Salary DESC;

Ordering by Text Columns:

SELECT \*

FROM Employees

ORDER BY LastName ASC, FirstName ASC;

Ordering by Date:

SELECT \*

FROM Employees

ORDER BY dob ASC;

Ordering with Aliases:

SELECT EmployeeID AS ID, FirstName AS Name, Salary

FROM Employees

ORDER BY Salary DESC;

GROUP BY

Simple Grouping:

SELECT DepartmentID, COUNT(\*) AS NumberOfEmployees

FROM Employees

GROUP BY DepartmentID;

Grouping with Aggregate Functions:

SELECT DepartmentID, SUM(Salary) AS TotalSalary

FROM Employees

GROUP BY DepartmentID;

Grouping by Multiple Columns:

SELECT DepartmentID, JobTitle, AVG(Salary) AS AverageSalary

FROM Employees

GROUP BY DepartmentID, JobTitle;

HAVING:

Simple Example:

SELECT DepartmentID, SUM(Salary) AS TotalSalary

FROM Employees

GROUP BY DepartmentID

HAVING SUM(Salary) > 100000;

Using Multiple Conditions:

SELECT DepartmentID, AVG(Salary) AS AverageSalary, COUNT(\*) AS NumberOfEmployees

FROM Employees

GROUP BY DepartmentID

HAVING AVG(Salary) < 60000 AND COUNT(\*) > 5;

Filtering Groups Based on Aggregate Functions:

SELECT SalesPersonID, COUNT(\*) AS NumberOfSales

FROM Sales

WHERE SaleAmount > 1000

GROUP BY SalesPersonID

HAVING COUNT(\*) > 5;

Using HAVING with GROUP BY:

SELECT DepartmentID, MAX(Salary) AS MaxSalary

FROM Employees

GROUP BY DepartmentID

HAVING MAX(Salary) > 70000;

BETWEEN

Numeric Range:

SELECT EmployeeID, FirstName, LastName, Salary

FROM Employees

WHERE Salary BETWEEN 40000 AND 60000;

Date Range:

SELECT OrderID, OrderDate, CustomerID

FROM Orders

WHERE OrderDate BETWEEN '2024-01-01' AND '2024-12-31';

Text Range:

SELECT ProductID, ProductName

FROM Products

WHERE ProductName BETWEEN 'A' AND 'M';

NOT BETWEEN:

SELECT EmployeeID, FirstName, LastName, Salary

FROM Employees

WHERE Salary NOT BETWEEN 30000 AND 50000;

JOINS

INNER JOIN:

SELECT columns

FROM table1

INNER JOIN table2 ON table1.common\_column = table2.common\_column;

LEFT JOIN (or LEFT OUTER JOIN):

SELECT columns

FROM table1

LEFT JOIN table2 ON table1.common\_column = table2.common\_column;

RIGHT JOIN (or RIGHT OUTER JOIN):

SELECT columns

FROM table1

RIGHT JOIN table2 ON table1.common\_column = table2.common\_column;

FULL JOIN (or FULL OUTER JOIN):

SELECT columns

FROM table1

FULL JOIN table2 ON table1.common\_column = table2.common\_column;

CROSS JOIN:

SELECT columns

FROM table1

CROSS JOIN table2;

SELF JOIN:

SELECT a.columns, b.columns

FROM table a

INNER JOIN table b ON a.common\_column = b.common\_column;

NETWORKS

**What is a Computer?**

A computer is an electronic device that can perform a variety of tasks by following a set of instructions, known as a program. The primary functions of a computer is for  storing, retrieving, and processing data. Computers can be used for various applications,  internet browsing, gaming.

**Key Components of a Computer**

1. **CPU (Central Processing Unit)**:
   * The CPU is often referred to as the "brain" of the computer. It performs all the calculations and processes the instructions.
   * The CPU operates using a cycle known as the **fetch-decode-execute cycle**:
2. **RAM (Random Access Memory)**:
   * Responsible for temporarily storing data .
   * RAM is volatile memory, which means it requires power to maintain the stored information. When the computer is turned off, all data stored in RAM is lost.
   * RAM stores data only while the computer is in operation; when the computer is shut down, everything in RAM is erased.
3. **Hard Disk**:
   * The hard disk is the computer's long-term storage.
   * Hard Disk Drive (HDD) is a non-volatile storage device. This means it retains data even when the computer is powered off,
   * It stores  user data permanently until it is deleted or modified.
4. **SMPS (Switched Mode Power Supply)**:
   * It regulates voltage and current efficiently, providing stable power to various components in electronic devices.
   * ensures that all parts receive the correct voltage and current.
5. **Motherboard**:
   * The motherboard is the main circuit board that connects all the components of the computer. It allows communication between the CPU, RAM, storage, and other peripherals.
6. **Cabinet (Case)**:
   * The cabinet houses all the internal components of the computer, protecting them from dust, heat, and physical damage.
   * The case protects internal components from dust accumulation, which can lead to overheating and component failure.
   * Most cases are designed with vents and airflow pathways to facilitate proper cooling, helping maintain optimal temperatures.
7. **Monitor**:
   * The monitor is output device and it is the display screen that allows users to interact  with the computer.
   * **Types of Monitors**
     + **LCD (Liquid Crystal Display)**: Commonly used, offering  good image quality.
     + **LED (Light Emitting Diode)**: A type of LCD that uses LED backlighting, providing better energy efficiency and brightness.
8. **Keyboard and Mouse**:
   * These are the primary input devices for a computer. The keyboard is used for typing text and commands, while the mouse is used for pointing, clicking, and navigating the things like opening or closing the tabs ,files etc.

The first computer designed by Charles Babbage is the **Analytical Engine in**1837.

 We can use computer to store, access and transfer our data.

* + **Communication**: Facilitating communication through email, video conferencing, and social media.
  + **Education**: Supporting online learning, research, and access to educational resources.
  + **Entertainment**: Providing platforms for gaming, streaming movies .

**We are migrating from Paper to Computer age!**

**Floppy Disk**

* + - **Description**: A thin, flexible magnetic storage medium, typically 3.5 inches in size.
    - **Capacity**: Usually holds about 1.44 MB.
    - **Use**: Common in the 1980s and 1990s for storing small files and transferring data between computers.

**CD/DVD Drive**

* + - * **Capacity**: CDs can hold about 700 MB, while DVDs can hold 4.7 GB (single-layer) to 8.5 GB (dual-layer).
      * **Use**: Used for media storage, software distribution, and data .

**Pen Drive (USB Flash Drive)**

* + - * + **Description**: it is a storage devices that connect via USB ports.
        + **Capacity**: Ranges from a few GBs to several TBs.
        + **Use**: Ideal for transferringfiles

**External Hard Disk**

* + - * + **Description**: A larger  storage device that connects via USB
        + **Capacity**: Can range from hundreds of GBs to several TBs.
        + **Use**: Used for backup, large data storage, and transferring large files.

**Memory Cards**

* + - * + **Description**:  used in cameras, smartphones.
        + **Capacity**: Ranges from a few GBs to several TBs.
        + **Use**: Provides additional storage for devices

**What is Data?**

**Definition**: Data is a collection of information, which can include numbers, text, images, or any other form of information that can be processed by a computer.

**Program**: A program is a set of instructions written in a programming language that tells a computer what operations to perform. It processes data and produces results.

**Software** refers to a collection of instructions or programs that tell a computer how to perform specific tasks.

Software can be broadly categorized into two main types:

**System Software**:

**Definition**:  designed to manage and control the computer hardware and provide a platform for running application software.

**Examples**: Operating systems (e.g., Windows, macOS, Linux),

**Application Software**:

**Definition**: designed to perform specific tasks or applications for the end-users.

**exampls:**google chrome, spreadsheets ,media players (e.g., VLC Media,

**The uses of software is:**

**Data Manipulation**: It allows manipulation of data, processing inputs, and generating outputs.

**User Interface**: Application software often includes user interfaces (UI) for users to interact with the computer system.

**What is an Operating System?**

An **Operating System (OS)** is system software that acts as an intermediary between computer hardware and the user.

It provides the necessary environment for applications to run, manages hardware resources, and offers tools and interfaces for users to interact with their devices.

Key Functions of an Operating System

**Hardware Management**:

**Process Management**:

**User Interface**:

Examples of Popular Operating Systems

**Microsoft Windows**: Known for its GUI, widely used in personal and business computing.

**macOS**: Developed by Apple, known for its integration with Apple hardware and software.

**Linux**: An open-source OS  used in desktops, servers, and embedded systems.

**how a server controls and interacts with client computers:**

To  connects server computers and client computers, to communicate and exchange [data.](http://data.it/" \t "_blank)They should be  be in a NETWORK.

Network is nothing but interconnection between the devices.

there are diferent types of networks

**Local Area Network (LAN)**:

* **Definition**: A network upto certain area such as a single building or campus.
* **Use**: schools,printers

**Wide Area Network (WAN)**:

* **Definition**: A network that covers a large  area, such as cities, countries
* **example:** Internet).

**Metropolitan Area Network (MAN)**:

* **Definition**: A network that spans a city
* **Use**:  to connect multiple buildings or sites within a metropolitan area.

**Personal Area Network (PAN)**:

* **Definition**: A small network used for personal devices, typically within a few meters.
* **Use**:  smartphones,  and laptops (e.g., Bluetooth connections).

**Wireless Network**:

* **Definition**:it uses wireless technology to connect devices.
* **Use**: Wi-Fi networks

**Virtual Private Network (VPN)**:

* **Definition**: A secure network and used to create a private, encrypted connection.
* **Use**: Used for secure  access to a private network
* explain TOPOLOGIES

The client computer sends a request to the server for specific resources or services.     The server processes the request

* for example
* You compose and send an email.
* The email client sends the email to the  server.
* The  server processes and forwards the email to the recipient's email server.
* The recipient's email server stores the email until it is retrieved by the recipient’s email client.

**data transfer between Two computer**

* application>data>NIC>packet>transfer in cable>paket>NIC>data>application
* **application layer:**
* sending a doc file
* **transport layer:**
* in this layer the data is divided in small units called segments.
* **network layer:**
* it wrapping into single units called packets.the packet contains source and desination ip adresses(like from and to adresses)
* **data link layer:**
* it adds the physical adress know as MAC(like door no)
* **physical layer:**the frames(the packets are converted in frames by Network interface card(NIC) are converted into electrical or optical signal
* **reception and decoding**
* it decodes the signal using decoding techniques
* **datalink:**
* it checkswhether the MAC adresss is same or not if it is then it recieve other wise ignores.
* **Network layer**
* it checks IP **adress.**
* **transport layer:**
* it extracts the segments and reasembeled into the original data
* **finalli Application layer:**
* finally the data is delivered.

**NETWORK DEVICES:**

These are the hardware components used to connect computer,servers,printers,and other devices to a network.

* RJ45
* ETHERNET
* NIC-it is embedded in cpu.without this we cannot recieve any information from outside.
* switch
* router

**Switch**:

It connects **multiple devices** within the **same network**.

* + **Role**: The switch acts as a "smart traffic cop" for data on the network. It uses the packet's destination address to decide where to send the packet.
  + **Function**: When a packet arrives at the switch, the switch reads the packet’s source and destination MAC (Media Access Control) addresses  The MAC address is used to identify the device on the local network.
  + Switches forward data only to the port where the destination device is connected. This reduces unnecessary traffic and increases network efficiency.
  + **what is the use of connecting multiple computer to the one switch?**
  + it is interconnected so i can transfer information from any where to any where.
  + devices connected to different ports on the switch can transmit simultaneously without causing collisions

**ROUTER**

* + A router is designed to receive, analyze, and forward data packets between computer networks.
  + It connects **different networks** together.
  + there are some  Key Functions of a Router are

**Packet Forwarding:**

Routers find the best path for data packets to travel across interconnected networks.

They use routing  protocols to decide where to send each packet.

**Network Address Translation (NAT):**

* + - * Routers can perform NAT, which allows multiple devices  to share a single public IP address. This enhances security and they conserves the number of IP addresses needed.

**Firewalls:**Routers  have built-in firewalls that filter incoming and outgoing traffic based on security rules. This helps protect the network from unauthorized access and cyber threats.

**THE WORKING OF ROUTER IS:**

**Receiving Packets:**When a data packet arrives at a router, it checks the packet’s destination IP address.

**Routing Decision:**The router  determine the best route for the packet. The routing table contains information about network paths and metrics that help the router choose the most efficient path.

**Forwarding Packets:**After determining the best route, the router forwards the packet to the destination device

**IP Addressing**

• An IP(Internet Protocol) address is a numerical number assigned to each device. that uses the Internet Protocol for communication. It serves as a unique identifier for devices within a network, allowing them to send and receive data across the internet.

An IP address consists of four bytes (8-bit blocks), written in decimal format  (e.g., 192.168.1.1). Each byte can range from 0 to 255.

There are

IPV4: ipv4 adresses are 32-bit number.

Ipv6: ipv6 adresses are 128 bit numbers.

**IP Addressing classes:**

Originally, IP addresses were divided into classes (A, B, C) based on their range, each class have a default subnet mask:

**Class A**: 1.0.0.0 to 126.0.0.0 with a subnet mask of 255.0.0.0

**Class B**: 128.0.0.0 to 191.255.0.0 with a subnet mask of 255.255.0.0

**Class C**: 192.0.0.0 to 223.255.255.0 with a subnet mask of 255.255.255.0

**Public IP Addresses**

Public IP addresses are assigned to devices that are directly accessible over the internet. These addresses are unique across the entire internet, ensuring that data sent from one device can be accurately delivered to another.

**Example Usage**:

Web servers, email servers, online gaming servers, and other services that need to be accessible from the internet.

class A-0.0.0.0 to 126.255.255.255

class B-128.0.0.0 to 191.255.255.255

class C -192.0.0.0 to 223.255.255.255

**PRIVATE IP ADDRESSES**

Private IP addresses are designed for use within private networks, such as homes, offices,

This means that devices using private IP addresses cannot directly to communicate with external networks unless they pass through a device that translates private IP addresses to public ones means when it passes through the device like NAT(netwwork adress translation).

Class A: 10.0.0.0 to 10.255.255.255

**class B** 172.16.0.0 to 172.31.255.255

**Class C :** 192.168.0.0 to 192.168.255.255

**SUBNETTING**

Subnetting is a technique used in networking to divide a larger network (IP address range) into smaller networks called subnet. To let you understand this I am taking an example that is

**A Large Organization's Network Design**

Imagine a large organization with multiple departments, like Sales, Marketing, IT, and HR. Each department needs its own network segment for managing computers, printers, and other devices. In that scenarios Subnetting is used to divide the larger network into smaller, manageable sub-networks (subnets).

It helps improve network performance and security .

**Benefits of Subnetting**

**Reduced network traffic**:

**Improved performance**: Smaller subnets can be managed more efficiently.

**Enhanced security**: Easier to implement security measures within smaller networks.

**Better organization**: Allows for logical division of networks based on departments, locations, etc.

**What is Port Number:**

**A port number is a numerical identifier in networking**

The port number is to identify and distinguish specific services or applications on a device. **Responsibility:** Port numbers allow a device to distinguish between different services or applications that are running on it. Each service or application listens on a unique port number.

**Example:** A web server listens on port 80 for HTTP traffic, while an email server listens on port 25 for SMTP traffic. When a device receives data, the port number in the request directs the traffic to the appropriate service.

To better understand for this concept let me take another example.

you have to met an actor who stayed at hotel at room number 105

here the destination service is the actor

hotel is the ip adress

port number is room number

**what is port forwarding:**

Port forwarding is a technique used in networking that allows external devices to access services on a private network.To understand this let metake an example that is.

**Scenario:**

Imagine you're on vacation and want to check your security cameras from your phone.

You open your camera app or web browser and connect to your home’s public IP address using the forwarded port means by using port you can connect to the your home network.

The router forwards this request to the internal camera system, that allowing you to see your live video feed just like if you were connected directly to your home network.

As I said it allows external devices means smart phone in this example to access services means security camera on a private network.

Virtual box

VirtualBox is a powerful, open-source virtualization software developed by Oracle. It allows you to create and manage virtual machines (VMs) on your computer.

**Purpose**: VirtualBox enables you to **run multiple guest operating systems** (OS) simultaneously on a single physical host machine

Installing & Configuring Web Server - PHP

* Step 1 : Install Apache in Web Server

* 1. Command : yum install httpd -y

○ Command : Set hostname as udc.example.com (using vi /etc/hostname we have to change permanently)

* Step 2 : TO Install PHP and other tools in Web Server

* 1. Commands : dnf module list

php dnf module -y enable php:8.1

dnf module -y install php:8.1/common

yum install mysql -y

yum install php-mysqli -y

* Step 3 : Check the installed PHP version and enable httpd service

○ Command : php -v

Command : systemctl enable --now httpd

* Step 4 : Create a test page in Web Server

* 1. Commands : cd /var/www/html vi php\_test.php

* Step 5 : Paste below content > Save and exit the file

<!DOCTYPE html>

<html> <body>

<h1>My first PHP page</h1>

<?php

echo "Hello World!"; ?>

</body>

</html>

* Step 6 : Now open the browser and type below

http://<Web server IP Address>/php\_test.php

Incase if its not working – we need to stop the firewall of linux

Commands :

1. setenforce 0
2. systemctl stop firewalld

* + Step 7 : Now you should see the test page
  + Step 8 : start the database server(clone copy of web server)
    1. Step 8.1 : Command : Set hostname as dbsvr.example.com (using vi

/etc/hostname we have to change permanently)

* + Step 9 : Need to stop firewall using these commands (setenforce 0, systemctl stop firewalld)
  + Step 10 : Need to remove bridged network (for security user) using nmtui
  + Step 11 : need to connect web server to database server (ssh root@192.168.30.9)

MySQL Database Installation & Configuration

* + Step 1 : Install MySQL Server in DB Server
    1. Command : dnf -y install mysql-server

* + Step 2 : Configure the character set in DB Server
    1. Command : vi /etc/my.cnf.d/charset.cnf

○ Insert below in the same file

[mysqld]

character-set-server = utf8mb4

[client]

default-character-set = utf8mb4

* + Step 2 : Enable the Mysql Service in DB Server
    1. Command : systemctl enable --now mysqld

* + Now configure the Database in DB Server
    1. Command : mysql\_secure\_installation ○ Hints for next steps (No,No,Yes,Yes,Yes)

* + After the setup, use below command to login to MySQL in DB Server
    1. Command 1 : mysql -u root -p

○ Command 2 : select user,host from mysql.user;

○ Command 3 : show databases;

* + Test Database creation commands
    1. Command 1 : create database test\_database;

○ Command 2 : create table test\_database.test\_table (id int, name varchar(50), address varchar(50), primary key (id));

○ Command 3 : insert into test\_database.test\_table(id, name, address) values("031", "CentOS", "India");

○ Command 4 : select \* from test\_database.test\_table;

○ Command 5 : drop database test\_database;

○ Command 6 : exit

# User Data Collector Application - Installation & Configuration

We need to prepare the MySQL Database for our PHP code in the DB host

* Step 1 : Login to DB server > MySQL login and Create new database called “udc”
  1. Command : mysql -u root -p

* Step 2 : Create new user called “udc” and assign permissions for the new user
  1. Command : CREATE USER 'udc'@'%' IDENTIFIED BY

'Welcome@123’;

○ Command : GRANT ALL PRIVILEGES ON udc.\* TO 'udc'@'%';

* Step 3 : Login to MySQL server from the web server host and verify the connection
  1. So first need to exit from mysql using exit;

○ Then again exit in DB server host using exit command, or else (ssh root@192.168.177.166 (udc server ip))

○ Then reconfirm the hostname (it should show as udc.example.com)

○ Command : mysql -h <DB Server IP> -u udc -p

* Step 4 : Now create a Table for our PHP application
  1. Command : Create database udc;

○ Command : Use udc;

○ Command : CREATE TABLE users ( id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255) NOT NULL, age INT NOT NULL, country VARCHAR(255) NOT NULL

);

○ Command : exit;

* Step 5 : Now create folders in DB Servers
  1. Command : mkdir /var/udc

○ Command : mkdir /var/udc/uploads

* Step 6 : Incase file is not uploading and change the access
  1. Command : chmod 777 /var/udc

○ Command : chmod 777 /var/udc/uploads

○ Command : ls -l /var

○ Command : systemctl stop firewalld.service

# Main Application Page

The Actual PHP main page code is below, please copy it and paste in “/var/www/html/main.php” file

<!DOCTYPE html>

<html>

<head>

<title>User Data Collection</title>

</head>

<body>

<?php

// MySQL database configuration

$servername = "192.168.30.9";

$username = "FACEPrep";

$password ="FACEPrep@123";

$dbname = "udc";

// Create a database connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Collect user data

$name = $\_POST["name"];

$age = $\_POST["age"];

$country = $\_POST["country"];

// Insert data into the MySQL database

$sql = "INSERT INTO users (name, age, country) VALUES ('$name', $age, '$country')"; if ($conn->query($sql) === TRUE) {

echo "User data has been successfully stored in the database.<br>";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

// Handle file upload

$uploadDir = '/var/udc/uploads/';

$uploadFile = $uploadDir . basename($\_FILES['userfile']['name']); if (move\_uploaded\_file($\_FILES['userfile']['tmp\_name'], $uploadFile)) { echo "File is valid, and it has been successfully uploaded.<br>";

} else {

echo "File upload failed.<br>";

}

}

?>

<h2>Enter User Information</h2>

<form method="post" enctype="multipart/form-data">

Name: <input type="text" name="name"><br>

Age: <input type="number" name="age"><br>

Country: <input type="text" name="country"><br>

File Upload: <input type="file" name="userfile"><br>

<input type="submit" value="Submit">

</form>

<?php

// Close the database connection

$conn->close();

?>

</body>

</html>

AWS(AMAZON WEB SERVICES)

**What is Cloud Computing?**

Cloud computing is the **delivery of different services through the Internet without the need for physical infrastructure**. These services include applications like data storage, servers, databases, networking, and software.

cloud computing allows businesses and individuals to store and access data, applications, and services via remote servers hosted on the internet, rather than on local servers or personal devices.

**Benefits of Cloud Computing:**

* **Cost Savings**: Pay only for what you use.
* **Scalability**: Scale up or down based on demand.
* **Flexibility**: Access resources from anywhere with an internet connection.
* **Security**: Cloud providers invest heavily in security measures
* AWS (Amazon Web Services) is a comprehensive cloud computing platform.
* the term "comprehensive" refers to the wide-ranging or extensive set of services and tools that AWS offers.
* This means that AWS provides virtually everything like , storage, databases, networking.
* Example:
* Netflix Streaming:
* Amazon Elastic Compute Cloud (EC2), Amazon S3, Amazon CloudFront.
* Explanation: Netflix uses AWS to stream video content to millions of users globally. In this the services like EC2 provides the computing power to run their servers, S3 stores the video files, and CloudFront (a Content Delivery Network) delivers the content to users with low latency(the time taken to travel from source to destination)

**virtualization connection with cloud computing**

Virtualization and cloud computing are closely related technologies to enable scalable, flexible, and efficient IT environments.

**AWS - Architecture**

**AWS Edge Locations, Regions, and Availability Zones**

**Edge Locations**, **Regions**, and **Availability Zones** are key components of AWS's global infrastructure..

**Edge Locations**

**Definition**:

* Edge locations are **data centers that AWS uses to cache content closer to users**,
* **providing lower latency and faster access to data**.

**Real-Time Example**:

* **Amazon CloudFront**: When a **user from New York accesses a website hosted in California**, CloudFront **serves cached content from the nearest edge location in New York**, reducing load times and improving user experience.

**Regions**

**Definition**:

* A region is a **physical location around the world where AWS clusters data centers**.
* Each region consists of multiple, isolated, and physically separate Availability Zones.

**Real-Time Example**:

* **AWS Regions**: If a **company wants to deploy applications within the European customers**, it can **choose the AWS Europe (London) region**. This helps improves latency for European users.

**Availability Zones (AZs)**

**Definition**:

* Availability Zones are **isolated locations within a region, designed to be independent of failures in other zones**.

**Real-Time Example**:

* **High Availability**: **A company deploying a critical application can distribute its instances across multiple AZs** within the US East (N. Virginia) region. This ensures that if one AZ experiences an outage, the application remains available from another AZ.

**Identity and Access Management (IAM)**

🡪crucial service within AWS that **helps manage access to AWS resources securely.**

🡪It enables you to manage users, groups, and permissions.

**Users**

**Root User**

**The root user is the original account owner and is created when you first set up an AWS account. This user has full, unrestricted access to all resources and services in the AWS account.**

**IAM USER**

**An IAM user is created using AWS Identity and Access Management (IAM), and it is intended to manage access to AWS resources securely. IAM users have specific, limited permissions defined by the AWS account administrator.**

**4. Groups**

IAM **groups are collections of IAM users**. You can **use groups to assign permissions to multiple users at once**.

This simplifies management because you can attach policies to the group instead of individual users.

**Example:**

* **Group: Developers**
  + **Members:** JohnDoe, JaneSmith, AlexBrown
  + **Permissions:** Access to EC2, S3, and CodeCommit

**5. Roles**

IAM **roles are similar to users in that they define a set of permissions**. However, **instead of being associated with a specific user, a role can be assumed by any user, application**, or service that needs it.

**Example:**

**Role Scenario:**

* **Role:** LambdaExecutionRole
* **Permissions:** A Lambda function needs to read from a DynamoDB table. A role with DynamoDB read permissions is created. The Lambda function is configured to assume this role when it executes.

**6. Policies**

IAM **policies are JSON documents that define permissions**. They specify what actions are allowed or denied on which resources.

**Example:**

**Policy:** DynamoDBFullAccess

**Use Case:** A developer team needs full access to DynamoDB tables. A policy with full DynamoDB permissions is created and attached to the Developers group.

json

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"dynamodb:\*"

],

"Resource": "\*"

}

AWS – CLI Install

The **AWS CLI (Command Line Interface)** is allows users to interact with AWS services directly from the command line. It provides a unified way to manage AWS resources by executing commands in a terminal or shell.

Using username "satish123".

satish123@192.168.29.116's password:

Last login: Tue Jul 30 21:46:37 2024

[satish123@udc ~]$ cd aws-cli

-bash: cd: aws-cli: No such file or directory

[satish123@udc ~]$ su root

Password:

[root@udc satish123]# su -

[root@udc ~]# ls

anaconda-ks.cfg aws-cli

[root@udc ~]# cd aws-cli

[root@udc aws-cli]# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.z ip" -o "awscliv2.zip"

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

100 58.0M 100 58.0M 0 0 2759k 0 0:00:21 0:00:21 --:--:-- 2412kyu

[root@udc aws-cli]# yum install unzip

CentOS Stream 9 - BaseOS 32 kB/s | 6.6 kB 00:00

CentOS Stream 9 - AppStream 60 kB/s | 6.8 kB 00:00

CentOS Stream 9 - Extras packages 57 kB/s | 7.2 kB 00:00

Package unzip-6.0-56.el9.x86\_64 is already installed.

Dependencies resolved.

Nothing to do.

Complete!

[root@udc aws-cli]# unzip awscliv2.zip  
  
[root@udc aws]# aws configure

AWS Access Key ID [\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*XBFY]: AKIAQE3RORLATVQKPX6E

AWS Secret Access Key [\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*RYKZ]: pLtGbsIqvvIRPplBiKCe9bVOU4oChkYe1P +5B0wX

Default region name [None]:

Default output format [None]:

[root@udc aws]# aws sts get-caller-identity

{

"UserId": "AIDAQE3RORLAQH2PRIZUJ",

"Account": "010438478529",

"Arn": "arn:aws:iam::010438478529:user/chintu"

}  
  
then goto Iam user

S3 (simple storage service):

How many of you know what is the storage?

Yes,storage is the space which holds something.

Now tell me what are the storage devices which we use in our daily life?

Yes,pendrives,memory cards,CDs etc.

Now coming to the concept s3 ,S3 stands for simple storage service which is provided by the AWS.

In this we are going to learn:

what is s3?How it works?

Different storage classes the s3 have? **S3 have the different storage classes those are designed to meet different data access needs and cost requirements**.

Replication rules: is a **feature that automatically copies objects from one S3 bucket (the source) to another S3 bucket (the destination).The bucket from where we are coping the objects is the source bucket and the bucket to where we copied is the objects is the destination bucket.**

Versioning: versioning is used to retain multiple versions of an object, it will more helpful when you accidentally delete the file it can store in it. Means it can stores the past data also.

Life cycle rule management

Amazon S3 (Simple Storage Service) is a highly scalable and secure object storage service.

**Real-Time Example:** A healthcare provider uses S3 to store patient records. To comply with data for protection regulations, the provider can use S3’s management features to set up bucket policies and permissions **that restrict access to sensitive information**. They can also enable versioning to keep track of changes and prevent data loss.

So with this example we are come to know ,the s3 is used to provide secure access and enables versioning to retain the multiple vesions of an object. if we are doing any changes in the file which are stored in s3 bucket those things also stored in S3.

**3.** Real-Time Example: **An online retailer uses S3 to store product images and videos**

**4**. Real-Time Example: **A media company produces large video files for streaming**. They can upload individual video files as large as 5 TB to S3

**Object Storage** means S3 stores data as objects, like files,documents etc.

**Scalability**: S3 can automatically scale means scale up or scale down to provide storage .

**Data Availability**: S3 is very high data availability, means you can always accessible when you need it.

**Security**: S3 offers various security features like encryption, access controls with AWS Identity and Access Management (IAM) to ensure your data is protected.

**Data Size and Upload**: Users can store and retrieve files ranging from 0 bytes to 5 terabytes. These Files are stored in "buckets."

**Bucket Naming**: Each bucket name must be unique across all AWS accounts and regions within a partition, ensuring that your bucket's name is globally unique.

BENEFITS:

**Unlimited Storage Space**

* **Explanation**: S3 offers virtually unlimited storage capacity, allowing you to store any amount of data without worrying about running out of space.
* **Example**: A large e-commerce company can store millions of product images, videos, and customer data in S3 without worrying about storage limitations. As the company grows and more data is generated, S3 will automatically scale to provide the increased storage needs.

**Data High Availability – 99.99%**

* **Explanation**: S3 is designed to provide high availability, it can be accessable at any time even the peak traffic occurs.

**Data Durability(**which means the risk of data loss is very low. ) **– 99.99999999999%**

* This is achieved by automatically replicating your data across multiple devices and locations.
* **Example**: A healthcare provider can store patient records and medical images in S3, Even if one copy of the data is somehow lost ,S3 has responsible to be copies are available.

**Simple Interface – Web & CLI**

* **Explanation**: S3 provides interface to managing data, through a web-based console and a command-line interface (CLI). So by this feature it can be easy to upload, retrieve, and manage data.
* **Example**: A developer can quickly upload a large dataset to S3 using the AWS CLI with a single command aws s3 followed by source path,filename,and destination path.

**Data Security**

* **Explanation**: S3 offers security features like encryption (both in transit and at rest), access controls with AWS Identity and Access Management (IAM) to manage user permissions.
* **Example**: A financial institution can store sensitive financial data in S3, encrypting it to ensure that even if unauthorized access occurs, the data remains unreadable. IAM policies can be set up to ensure that only authorized personnel can access the data.

**S3 Storage Classes:**

S3 **storage classes designed to meet different data access needs and cost requirements**.

1. **S3 Standard**

**Explanation:** S3 Standard is designed for frequently accessed data. It **offers high durability, availability, and performance**.

**Real-Time Example:** **A news website stores articles and images that are accessed frequently by readers**. Using S3 Standard ensures that the website’s content is quickly available and reliable.

1. **S3 Standard-IA (Infrequent Access)**

**Explanation:** **S3 Standard-IA is for data that is less frequently accessed but needs to be retrieved quickly when required**. It has a **lower storage cost compared to S3 Standard** but charges for retrieval.

**Real-Time Example:** An **organization archives historical financial records that are accessed only occasionally but need to be readily available when needed**. Using S3 Standard-IA helps reduce storage costs while ensuring fast access when necessary.

**3.S3 Intelligent-Tiering**

**Explanation:** S3 Intelligent-Tiering **automatically moves data between two access tiers (frequent and infrequent**) **based on changing access patterns**. This **helps optimize costs** while maintaining performance.

**Real-Time Example:** A **company stores large datasets for analysis**. **Initially, the data is accessed frequently, but over time, access becomes less frequent**. With S3 Intelligent-Tiering, the **so company’s data is automatically moved to a lower-cost tier when access decreases**,so we can saving money without manual intervention.

**4.S3 One Zone-IA**

**Explanation:** S3 One Zone-IA is **for infrequently accessed data that does not require multiple availability zones for redundancy**. It **offers lower storage costs compared** to S3 Standard-IA but with less durability.

**Real-Time Example:** A **company stores backup copies of non-critical data that doesn’t need high redundancy**. They use S3 One Zone-IA to save on storage costs while accepting that the data is stored in a single availability zone.

**5.S3 Glacier**

**Explanation:** S3 Glacier is **designed for archival storage with a low cost for long-term storage**. In this the **data that is rarely accessed and requires retrieval times** of **minutes to hours**.

**Real-Time Example:** **A research institution archives large datasets that are not frequently accessed but need to be preserved for long periods**. They use S3 Glacier to **store this data cost-effectively, accepting longer retrieval times**.

**6.S3 Glacier Deep Archive**

**Explanation:** S3 Glacier Deep Archive is the lowest-cost storage class for data that is rarely accessed and **requires retrieval times of 12 hours or more**. It is intended for long-term archival storage.

**Real-Time Example: A government agency archives legal documents and historical records that are seldom accessed but must be kept for decades**. They use S3 Glacier Deep Archive to minimize storage costs while complying with retention policies.

S3- Replication Policy

1.Amazon S3 Replication is a **feature that automatically copies objects from one S3 bucket (the source) to another S3 bucket (the destination).**

**Real-Time Example:** **A multinational company stores user-generated content in an S3 bucket** in the US (source bucket). They **set up replication to automatically copy this content to a bucket in Europe (destination bucket) to ensure data is available for users in both region**s without manual intervention.

**Real-Time Example:** A **company having two teams,marketing team and finance team each have separate AWS account**s. They set up replication from a marketing team bucket to finance team bucket to share marketing reports with the finance team.

Handson:

* Create two buckets and upload files in it
* Open the bucket which you want to replicate(source bucket)
* Go to management
* Click on create replication rule
* Replicattin name
* Status ------enable
* Source bucket-----choose a rule scope------apply to all
* Destination-----choose a bucket-------browse s3-----select destination bucket
* IAM role----chose new role------save
* Replicate existing objects?----------yes,replicate existing objects------submit

HANDS ON:

* Go to s3
* Click on create bucket
* Bucket type------general purpose
* Bucket name
* Object ownership----ACLs enabled
* Disable block all public access
* Bucket versioning
* Click on create bucket
* Open the bucket
* File upload
* Open the bucket
* Copy object url and paste it in web browser
* Its not opening ,so to make it open u can
* Go to object action
* Select make public using ACL
* Click on make public
* Copy object url and paste it in browser now its opening.

Static web hosting:

to host websites that serve static content. Static content includes HTML files, CSS, JavaScript, images, videos, and other files that do not require server-side processing.

* Go to bucket
* Open the bucket
* Go to properties scroll down and there you can find static web hosting
* Click on edit
* Static web hosting ----Enable
* Give a index document name------index.html
* Save changes.

Versioning:

versioning is used to retain multiple versions of an object, it will more helpful when you accidentally delete the file it can store in it.it can stores the past data also.

* Open the bucket
* Go to properties
* Go to bucket versioning------edit-----enable------save changes.

Life cycle rule:

AWS S3 lifecycle rules allow to automatically manage the storage of objects based on defined policies.

**Move Current Versions of Objects Between Storage Classes**

**How It Works**: You can configure a rule that moves the current version of an object from one storage class (e.g., S3 Standard) to another (e.g., S3 Standard-IA or S3 Glacier) after a specified number of days. For example, you may move objects to S3 Glacier after they haven't been accessed for 90 days.

* **Storage Classes**: Options include Standard-IA (Infrequent Access), One Zone-IA, Glacier Instant Retrieval, Glacier Flexible Retrieval, and Glacier Deep Archive.

**Move Noncurrent Versions of Objects Between Storage Classes**

**How It Works**: If versioning is enabled, noncurrent versions means older versions of objects can be automatically transitioned to lower-cost storage classes after a set number of days. For example, a noncurrent version of an object might be moved to Glacier after 30 days.

* **Application**: This rule is particularly useful when you want to keep older versions but don’t expect to access them often.

**Expire Current Versions of Objects**

**How It Works**: You can set a rule to automatically delete (expire) the current version of an object after a specified number of days. For instance, you might set a rule to expire objects that are older than one year.

**Permanently Delete Noncurrent Versions of Objects**

**How It Works**: This rule automatically deletes noncurrent versions of objects after a specified number of days since they became noncurrent. For example, you might choose to permanently delete noncurrent versions after 180 days.

**Delete Expired Object Delete Markers or Incomplete Multipart Uploads**

* **Expired Object Delete Markers**: When versioning is enabled, deleting an object creates a delete marker instead of actually removing the object. This rule automatically deletes these expired delete markers, which helps in reducing collection of unnecessary items that take up space and make it harder to manage in versioned buckets.
* **Incomplete Multipart Uploads**: Multipart upload is a feature in S3 that allows you to upload large objects in parts. However, sometimes uploads might not be completed, leading to unnecessary storage of incomplete parts. This rule can automatically delete parts of uploads that haven’t been completed within a specified timeframe (e.g., after 7 days).
* **Move current versions** to S3 Standard-IA after 30 days.
* **Move noncurrent versions** to Glacier after 90 days.
* **Expire current versions** after 365 days.
* **Permanently delete noncurrent versions** after 180 days.
* **Delete incomplete multipart uploads** after 7 days.

EC2 (ELASTIC COMPUTE CLOUD):

So lets start our today’s session which is most popular AWS offering service that is EC2.

* Let Say whenever you run multiple application, it is certain that you require a server ,so depending upon the application that you are running ,you might need a few servers and some times you need large once and sometimes you might need smaller ones.
* Traditionally,obtaining servers could be quite time-consuming and typically something that could take weeks or even months,and You have to do reaserach into the right kind of hardware to buy ,and may be get budjet approval and then purchase the hardware and eventually get access to your servers.The servers are all dependent as per your requirements.
* The amazon EC2 actualy provides you those virtual servers, along with secure and resizable compute capacity in the cloud. virtual in the sense that they are not physically existing but you can use them anywhere in the world without any limitation .now with amazon EC2 it makes it easier for you to obtain those virtual servers like you simply choose the instance type you want ,and the template you would like to use, and launch the quantity you need. and with amazon EC2 you can pay only for what you use.when you aare done your instance you simply terminate or stop them and you stop paying for them.
* now Let’s see why ec2 is one of the most popular aws offering service? with EC2 you can scale your instance up and down as per your requirement.and you have a freedom to choose different operating systems like windows,linux .and ec2 comes with a tight security network as easy to work with virtual private cloud(It allows you to control the network settings like IP addresses, subnets.It is like a private data center in the cloud, in this we can deploy instances and services with complete control in the networking environment.) to provide you that secure network to all the resources which you are going to utilize. EC2 allows users to launch virtual machines (EC2 instances) with different specifications. Once launched, these instances can host websites, applications, databases, or any other computing workloads.

amazon EC2 provides a range of instance types designed for different use cases.

Those are

* **On-Demand Instances:** Pay for compute capacity by the hour or second with no long-term commitments. A developer uses On-Demand Instances to test a new application.
* **Reserved Instances:** You make a one-time payment or choose a monthly payment plan for a one or three-year term to reserve instances at a significant discount.
* **Spot Instances:** Spot Instances allow you to bid for unused EC2 capacity at a lower price than On-Demand instances.
* **Dedicated instances:** Dedicated Instances run on hardware that is dedicated to a single customer. This means your instances are physically isolated from those of other customers.

So let’s quickly do a handson to see how simple it is,soooo lets get stareted with the handson . so for that ,the first thing you need to do is go to AWS management console, from there you can search for amazon ec2.

HANDSON:

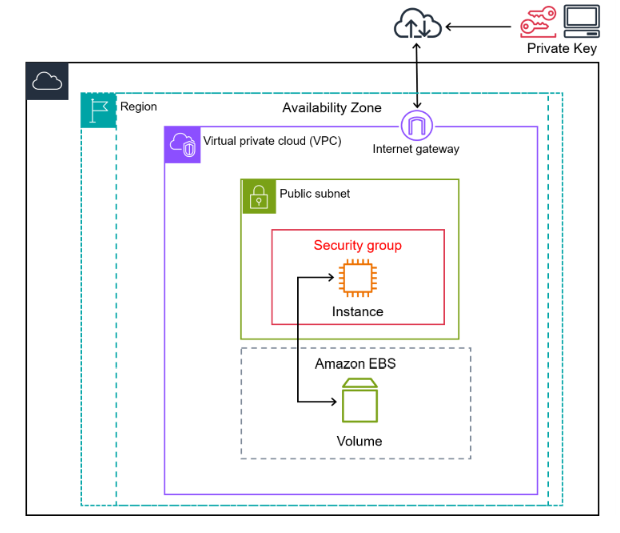
Creating linux server:

* Click on launch instance
* Name-----quick start-----linux---
* Key pair-----create-----name-----RSA-----.pem----click on create pair.( **.pem** is a widely-used format for storing public/private keys used in SSH authentication. **.ppk** is a format specific to PuTTY for storing private keys used for SSH connections. If you're using PuTTY to connect to a server, you'll need a .ppk key.
* Network settings----create security group(Security groups protect your instances by controlling the traffic.)----ssh,https,http allow them
* Click on launch instance
* Open putty gen(PuTTYgen it is also know as "PuTTY Key Generator" is a tool used to generate public and private key pairs for use with SSH (Secure Shell) and other cryptographic applications)
* Load putty gen----select danvi\_key pem file-----save private key----as danvi\_private key( We use **private keys** and **public keys** in cryptography, particularly in public key infrastructure (PKI) systems, for secure communication, encryption, and authentication. Together, these keys form a **key pair** that is essential for secure data transmission over the internet.)
* Open putty----ssh—auth—credentials---browse private key(danvi\_private key)----session—Hostname---ec2-user@public ip address of the instance----save----load---now it is opening the linux server

**Creating** windows server:

* Create instance
* Name-----quick start-----windows---
* Key pair-----create-----name-----RSA-----.pem----click on create pair.
* Network settings----create security group----ssh,https,http allow them
* Launch instance
* Connect
* RDP client connection(An **RDP (Remote Desktop Protocol) client connection** refers to a method of remotely accessing and controlling a computer or virtual machine (VM) over a network.)
* Download remote desktop file
* Get password
* Upload pem file
* Decrypt password
* password created
* open theremote desktop file
* connect
* copy the password from get password
* paste it in administrator
* click on ok
* The windows server will open

Picture explination:



Region: data is distributed globally to make it available for any regions.These regions are contains multiple data centers.

**Availability Zone (AZ**):it is a data center within a region. so if one AZ fails, the others can continue to operate.

**VPC (Virtual Private Cloud):** It allows you to control the network settings like IP addresses, subnets.

It is like a private data center in the cloud, in this we can deploy instances and services with complete control in the networking environment.

**Internet Gateway**:actually it connects VPC to the internet, it enables communication between VPC and the outside world.

The Internet Gateway allows instances in a VPC to communicate with the internet for web servers.

**Public Subnet:** These Public subnets are used for resources that need to be accessible publicly, like web servers.

**Private key:** used to securely access the instances.

**Security group:** Security groups protect your instances by controlling the traffic.

**Instance:** instances are nothing but servers we know that.

**EBS Volume (Elastic Block Store):** EBS volumes provide storage for instances.

AUTO SCALING:

What is Auto Scaling?

Before going to know about auto scaling let’s understand why auto scaling.let me take an example i.e enterprises.They were Spending a lots of money on the purchase of infrastructure if they have to set up some kind of solution So they have to purchase an infrastructure and one time cost was required So that was a burden for them in terms of a server handware, Software and then having a team of experts to manage all those infrastracture .So they used to think that no longer they require these resources.

Now after the aws Cloud Scaling that was introduced automatically the auto Scaling maintains the application performance based on the user requirements at the lowest price so what does the auto scaling does whenever there is a scalability require it manages it automatically and hence the cost optimization become possible.

Now, what is aws auto scaling, let's look into deep…. so aws auto scaling is a servece that helps users to monitor applications and automatically adjusts Capacity to maintain steadiness. they can increase the capacity they can decrease capacity also for the cost optimization.

And also predictable performance at the lowest possible cost.

How does Aws auto scaling work?

To work with autoscaling you have to configure single unified Scaling policy for application resource. By Scaling policy you can explore the application. then select the service you want to scale . and then keep track of scaling by monitoring or getting the notifications.

Benefits:

it give Better fault tolerance:

you can get the Servers Created and you can have a clone copy of the Servers so that you don't have to deploy the applications again and again.

2. Better cost management: Because the scalability is decided by the aws automatically based on some parameters

it was a reliable service whenever the scaling is Created or initiated you can get the notificat -ions onto your mail id's.

4. Scalability as I mentioned is always there in the auto scaling it can scale up it can Scale down and

5.it has the flexibility in terms of whenever you want to schedule it if you want to stop it if you want to keep the size of the Servers at a fixed number, you can always make the Changes on the fly. And

6.The better availability.

Different Scaling plans:

A Scaling plan helps a user to configure a set of instructions for Scaling based on software requirements.

Scaling strategy guides the service of AWS auto scaling on how to optimize resources in an application.

TYPES OF AUTO SCALING:

**Vertical Scaling**

Vertical scaling (also known as "scaling up") is used to increasing the capacity of a single machine. by adding more resources like CPU, memory, or storage to an existing server. Then the server itself becomes more powerful and can handle more load.

**Example**: Upgrading an EC2 instance from a t2.micro to an m5.large in EC2 instance(instance type)

**Advantages:**

* **Simplicity**: Easier to manage as it involves a single machine.
* **Consistency**: Maintains the same environment and configurations.

**Disadvantages:**

* **Limits**: There's a limit to how much you can scale a single machine.
* **Downtime**: (Downtime is a period during which a system is not operational or unavailable for use.)May require downtime during the upgrade process.

**Horizontal scaling**:

Horizontal scaling, also known as scaling out, adding more machines or devices to a system's infrastructure to increase its capacity and efficiently manage higher levels traffic or workload.

**Increased Capacity**: By adding more devices (servers, virtual machines, etc.), the system can handle more traffic and process more requests simultaneously.

**Redundancy**:If one device fails, others can continue to handle the load.

**Distributed Workload**: The workload can be distributed across multiple machines, it can improve performance and prevent any single device from becoming a bottleneck(bottleneck in a system refers to a point of or limitation that reduces the overall efficiency and capacity of the entire system..

**Handson**:

* Create EC2 instances
* Target group
* T.name,/index.html,next,launch target group
* Goto load balancer
* Application load balancer
* Name,1a,1b,security group-wizard 1
* Default action -give the created target group here
* Launch
* Go to autoscaling group
* Name,network-attach existing lb
* Health check-100
* Desired capacity-2
* Vcpu,memory-no min no max
* Min -2 max-3
* Target tracking
* Target value-70,instance warm -50
* Next
* Next
* Next
* Create auto scaling group
* Next see in instances automatically two instances created
* Those two should connect
* Sudo su
* Ls
* Yum install httpd -y
* Same 3 commands in another instance also
* Cd /var/www/html
* Vi index.html
* Jio movies
* In another instance Cd /var/www/html
* Vi index.html
* Zee movies
* In both instances start httpd.service
* Copy public ip of first instance and paste it into browser <http://public> ip

EFS(Elastic File System)

Elastic File System (EFS) is a scalable and fully managed file storage service that can be mounted to multiple Amazon EC2 instances. This will be more useful where multiple instances need to shared access to the same file system.

Consider an example you have a web application that is running on multiple EC2 instances. These instances need access to the same set of files By using EFS, shared access to these files.

To mount the EFS file system on multiple instances, simply repeat the mounting steps on each EC2 instance. Once mounted, all instances will have shared access to the EFS file system, and any changes made to the files will be immediately appear in all other instances.

Elastic File System uses the Network File System version 4 (NFSv4) protocol, which allows you to mount the EFS file system on multiple EC2 instances.

**Let discuss about Key Characteristics of Amazon EFS**

1. **Multiple Availability Zones:**
   * EFS is designed to be highly available across multiple Availability Zones (AZs)
   * By using multiple availability zones the data remains accessible even if an entire AvailableZones becomes unavailable.
2. **Pay-Per-Use Model:**
   * EFS uses a pay-per-use pricing model, means you have to pay money for the storage you have used.

PRACTICLES:

* Create two instances in EC2.(EFS\_1 , EFS\_2)
* Go to security group(left side)
* Basic details-name,description(same as name)
* Inbound rules---🡪type-NFS , source -wizard-1
* Click on Create security group
* Go to EFS
* Create file system
* Name---🡪customize
* File sytem type---🡪regional
* Disable automatic backups and encryption of data at rest.
* Next
* Mount targets--🡪give 3 security groups
* Next----🡪next
* Create
* Connect the instances
* (Sudo su
* Yum install httpd -y
* Service httpd start
* Yum install -y amazon-efs-utils)
* These commands should do in two instances.
* Go to file system
* Open created file system
* Click on attach
* Copy EFS mount helper sudo mount…..
* Paste it in two instances.
* ((Sudo mount ……../space/var/www/html
* Vi /var/www/html/google.html))these two commands should give in only one instance
* Ls command in both instances.
* You can see the google file in instance\_2 also.

EBS(Elastic Block Storage):

EBS is the Block-level storage where data is divided into fixed-size blocks. these storage system can access and manipulate data at the block level rather than the file level. This system is for fast access and retrieval.

As I mentioned that The key character is data divide in to blocks.Let me take an example

Imagine you have a 10MB file. Block storage will divide this file into 10 blocks of 1MB each. When you need to access part of the file, the system can retrieve the specific blocks needed without need of loading the entire file.then the retrival time and loading time is less.

**Advantages**

1. **HA – Highly available:**This means that the storage is designed to be against hardware failures.so the Data is automatically duplicated within the same Availability Zone so the data remains available.
2. **Designed for Product Environment:** let's consider an example of a healthcare application.

* Consider a health care company, The management system stores medical records in elastic block storage.
* when a patient arrives at the emergency room with a critical condition, and the medical staff needs to access the patient's records immediately.
* Then they can retraive data from medical records fastly and they can ready for doing operations.

**Dynamically Scalable**

**Means** You can increase the storage capacity as needed without any service interruption.

You have the flexibility to change the type of storage volume without

downtime means These changes can be made without shutting down the instance or impacting the performance of the system.

* **No downtime**:
* **There will be no performance issues to EC2**:
  + Adjusting storage capacity or volume type does not negatively affect the performance of the associated EC2 instances. This ensures continuous, smooth operation of applications.

Practicle:

* Create instances(in instance create key pair and download pem key) and connect.
* Sudo su
* Yum install httpd -y
* Service httpd start
* Service httpd status
* Cd /var/www/html
* Vi index.html
* Run in browser (ip address/index.html
* To modify the size -----move to instance
* Go to ebs(left side)
* Click on volume modify
* Give 30gb
* In terminal
* df -hT
* lsblk
* growpart /dev/filesystem name
* lsblk
* df -hT
* xfs\_growfs -d/
* df -hT
* now, go to volumes
* go to action
* create snapshot
* now open left side snapshot
* open snapshot
* options----🡪create image from snapshot
* Give name and create image
* Go to left side AMI
* Now change the destination
* Go to AMI
* Check in another country
* After checking status available
* Launch instance with AMI
* Give Mumbai key value pair as same launch instance.

DIFFERENCE BETWEEN SNAPSHOT AND AMI:

* In a company there was the one of the employ facing an issue with launching the vitual machines.
* So he asked his colleague a question is it possible to launch multiple virtual machines with the minimum time because it takes lots of time.he said yes.it is possible it launch multiple EC2 instances and that can be done at lesser time this can be done either snapshot or AMI.
* Snapshot :it is a kind of backup of a single EBS volume like virtual hard drive.That is attach to EC2 instance. It is used when the instance contains multiple static EBS volumes. In snapshot pay only for the storage of the modified data . snapshots are the non-bootable images means that cannot be used to launch an instance
* AMI: is use to backup of EC2 instance only. This is widely used to replace a failed EC2 instance.in AMI
* Pay only for the storage that you use.it is a bootable image on EC2 instance.

EFS(Elastic File System)

Elastic File System (EFS) is a scalable and fully managed file storage service that can be mounted to multiple Amazon EC2 instances. This will be more useful where multiple instances need to shared access to the same file system.

Consider an example you have a web application that is running on multiple EC2 instances. These instances need access to the same set of files By using EFS, shared access to these files.

To mount the EFS file system on multiple instances, simply repeat the mounting steps on each EC2 instance. Once mounted, all instances will have shared access to the EFS file system, and any changes made to the files will be immediately appear in all other instances.

Elastic File System uses the Network File System version 4 (NFSv4) protocol, which allows you to mount the EFS file system on multiple EC2 instances.

**Let discuss about Key Characteristics of Amazon EFS**

1. **Multiple Availability Zones:**
   * EFS is designed to be highly available across multiple Availability Zones (AZs)
   * By using multiple availability zones the data remains accessible even if an entire AvailableZones becomes unavailable.
2. **Pay-Per-Use Model:**
   * EFS uses a pay-per-use pricing model, means you have to pay money for the storage you have used.

PRACTICLES:

* Create two instances in EC2.(EFS\_1 , EFS\_2)
* Go to security group(left side)
* Basic details-name,description(same as name)
* Inbound rules---🡪type-NFS , source -wizard-1
* Click on Create security group
* Go to EFS
* Create file system
* Name---🡪customize
* File sytem type---🡪regional
* Disable automatic backups and encryption of data at rest.
* Next
* Mount targets--🡪give 3 security groups
* Next----🡪next
* Create
* Connect the instances
* (Sudo su
* Yum install httpd -y
* Service httpd start
* Yum install -y amazon-efs-utils)
* These commands should do in two instances.
* Go to file system
* Open created file system
* Click on attach
* Copy EFS mount helper sudo mount…..
* Paste it in two instances.
* ((Sudo mount ……../space/var/www/html
* Vi /var/www/html/google.html))these two commands should give in only one instance
* Ls command in both instances.
* You can see the google file in instance\_2 also.

ELB(Elastic Load Balancing):

What is elastic load balancing

Elastic Load Balancing (ELB) is automatically distributes network traffic across multiple targets, such as Amazon EC2 instances.

Real Life Example:

If you take an example like"FLIPKART," the number of visitors to the "FLIPKART" varies at different times of the day. during special sales or promotional events the number of visitors will be more.which experiences traffic means loading the page throughout the day, To ensure the website remains responsive and available even during peak traffic, in that situation the Elastic Load Balancing will more helpful.

We can do the same for multiple AZ’S also:

Means, By deploying "flipkart" across multiple AZs using Elastic Load Balancing, it ensure that the website can handle the traffic, it will be more available to the visitor.

NOW,

How the load balancing will work?

**Let me take an example that is flipkart**

* + Users make requests to the website (e.g., [www.flipkart.com](http://www.flipkart.com)).
  + We know that [www.flipkart](http://www.flipkart) is the DNS name.So,it will under go the resolution to the ip address in load balancer.
  + The load balancer receives incoming traffic on specified ports like HTTP on port 80 or HTTPS on port 443
  + Here the **Listeners** play key role. Listeners are nothing but They listen for requests from ports and direct the traffic to the appropriate target groups based on rules.
  + Here the **rules** arenothing but they used to determine how incoming traffic should be routed (shared) to different target groups based on the conditionslike Path-based Condition(based on the URL path in the request) or Host-based Condition(based on the hostname (domain) in the request)
  + A **target group** a set of targets like instances ,that are used to route incoming traffic, based on defined rules.
  + The load balancer performs **health checks** on targets means EC2 instances whether they are healthy or not and are they can handle the requests.
  + Only healthy targets(instances)can receive the traffic and
  + The load balancer distributes incoming requests to the healthy targets using a selection algorithm (e.g., round robin, least connections).To ensures there are enough resources to handle the incoming traffic the load balancer establishes the scaling groups.These scaling groups ensures that usage of enough resources to handle the incoming traffic.
  + Targets process the requests and give responses to the load balancer.
  + The load balancer then sends the responses back to the clients.
  + This is work of load balancer.

AWS have several types of load balancers, each designed to handle different types of traffic.

CLASSIC LB

APPLICATION LB

GATEWAY LB

NETWORK LB

CLASSIC LB:

**Classic Load Balancer (CLB)** is AWS's original load balancer it is designed to handle HTTP/HTTPS traffic.

Layer 7 (Application Layer) Support**s** to the classic load balancer to Handle HTTP and HTTPS traffic, allowing for basic routing based on URL paths and hostnames.

How it works?

**Request Reception**

**Rule Evaluation**

**Health Checks**

**Traffic Routing**

**Request Forwarding**

**Response Handling**

These things are handled by the classic load balancer.

APPLICATION LOAD BALANCER:

**Application Load Balancer (ALB)** is operates at Layer 7 (Application Layer) of the OSI model, it is useful for complex routing and content-based traffic management. ALB provides advanced features for managing HTTP and HTTPS traffic.

How it works?

**Request Reception**

**Rule Evaluation**

**Health Checks**

**Traffic Routing**

**Request Forwarding**

**Response Handling**

The main difference between ALB and CLB is

Routing Capabilities:

* ALB supports
  + **Content-Based Routing:** Routes requests based on URL paths, hostnames, and HTTP headers.
  + **Path-Based Routing:** Directs traffic based on the URL path (e.g., /images vs. /api).
  + **Host-Based Routing:** Routes requests based on the hostname (e.g., api.example.com vs. [www.example.com](http://www.example.com)).

CLB supports the basic routing only

* + **Basic Routing:** Routes traffic based on the IP address and port. For HTTP/HTTPS traffic, it does not offer advanced routing capabilities.
  + It does not support routing based on URL paths or hostnames.
  + **Health Checks:**
  + **ALB:** Provides advanced health checks that can be customized to monitor specific paths and response codes.
  + **CLB:** Provides basic health checks with simple configuration options. Health checks are typically done on HTTP status codes for HTTP/HTTPS traffic.
  + **Target Types**
  + **ALB:** can manages the traffic in EC2 instances, containers (ECS), IP addresses, and Lambda functions.
  + **CLB:** Primarily manage traffic to EC2 instances. It does not support containers .

**PRACTICLE:**

Create 1st instances

* Create instance
* Name of the instance
* Type of instance-linux
* Key-pair(existed)
* Networking settings---edit-🡪subnet---🡪south-1a
* Security group firewall--🡪existing security group---🡪Launch wizard-1

Create 2nd instance

* Create instance
* Name of the instance
* Type of instance-linux
* Networking settings---edit-🡪subnet---🡪south-1b
* Security group firewall--🡪existing security group---🡪Launch wizard-1

connect them run these commands in two instances

* Sudo su
* Yum install httpd -y
* Service httpd start
* Vi /var/www/html/index.html
* It opens editor “hi this is swarupa”
* Esc+:wq!+enter

classic load balancer:

Go to Classic load balancer(left side)

Click on create

Give name for load balancer

Scheme---🡪infernet- facing

Networking--🡪south-la,south-1b

Security group--🡪wizard 1

Listeners

Health check

Add instances---add instance--🡪 instance\_1 and instance\_2---🡪confirm

Create load balancer

Click on the created load balancer

Copy DNS and paste it in browser(refresh it will give two pages).

You can see the output in window+R also

Curl DNS name again

Curl DNS name

Note:delete classic load balancer

APPLICATION LOAD BALANCER:

In instance\_1 connection In instance\_2

Cd/var/www/html cd/var/www/html

Mkdir shop1 mkdir shop2

Cd shop1 cd shop2

Vi shop1.html vi shop2.html

NEXT create target groups:

* Create target group
* Basic configuration---🡪instances
* Target group name-shop1
* Protocol-http
* Ip address-ipv4
* http1
* health check path--🡪/shop1/shop1.html
* next
* Register targets---select instance\_1
* Click on include as pending below
* create target group
* go to load balancer(left side)
* create load balancer
* application load balancer Click on create
* Give name for load balancer
* Scheme---🡪infernet- facing
* Networking--🡪south-la,south-1b
* Security group--🡪wizard 1
* Listener and routing----🡪port-80,default action-shop1
* Create load balancer
* Open the load balancer
* Go to listener and rules----🡪open http-80
* Go to listener rules----🡪add rule
* Name--🡪shop1
* Next
* Add condition ---🡪select path--🡪/shop1\*---🡪confirm
* Next
* Action--🡪target group--🡪shop1-🡪next-🡪priority---1
* Next
* open http-80
* Go to listener rules----🡪add rule
* Name--🡪shop2
* Next
* Add condition ---🡪select path--🡪/shop1\*---🡪confirm
* Next
* Action--🡪target group--🡪shop2-🡪next-🡪priority---2
* next
* Create
* Copy application load balancer DNS name and paste it in browser.it opens.