

**G. K. GUJAR CHARITABLE TRUST'S**

**Dr. Ashok Gujar Technical Institute's  
Dr. Daulatrao Aher College of Engineering, Karad**



**Department of Computer Science & Engineering**

**Academic Year 2023-24**

<b>Prepared by:</b>	<b>Approved by:</b>
Mrs. Archana H. Renushe	

**Course In-charge**

**HOD**

**Principal**

**INTRODUCTION ABOUT LAB**

There are 17 systems (HCL) installed in this Lab. Their configurations are as follows: Processor : , 2gb DDR 3 , SATA, DVDR/W, HDD, 18.5 TFT Monitor Key/mouse.

RAM

Hard Disk : 500gb

Monitor : HCL dual core 3.0 Ghz

**LAB CODE:**

- 1 Equipment in the lab for the use of student community. Students need to maintain a proper decorum in the computer lab. Students must use the equipment with care. Any damage is caused is punishable.
- 2 Students are required to carry their observation / programs book with completed exercises while entering the lab.
- 3 Students are supposed to occupy the machines allotted to them and are not supposed to talk or make noise in the lab. The allocation is put up on the lab notice board.
- 4 Lab can be used in free time / lunch hours by the students who need to use the systems should take prior permission from the lab in-charge.
- 5 Lab records need to be submitted on or before date of submission.
- 6 Students are not supposed to use floppy disks, pen drives and CDs.
- 7 All the students must wear ID cards, leave their foot wear out side the lab and must come with a formal dress code.
- 8 Doesn't bring bags inside the lab.

**List of Experiment:**

<b>Expt. No.</b>	<b>Title of Experiment</b>
1	Working and Implementation of Infrastructure as a service.
2	Implementation of Software as a Service.
3	Working and Implementation of Platform as a service
4	Implementation of Storage as a Service.
5	Implementation of Virtualization in Cloud Computing to Learn Virtualization Basics, Benefits of Virtualization in Cloud using Open-Source Operating
6	Installing OS on a Virtual Machine Monitor.
7	Offline/Online Migration of Virtual OS
8	Deployment and Configuration options in Amazon (AWS).
9	Deployment and Configuration options in Microsoft Azure.
10	Assignment to install and configure Google App Engine

## Experiment No: 1

**Title:** Working and Implementation of Infrastructure as a service.

**Aim:** To Study and implement Infrastructure as a Service using Amazon AWS.

### Theory:

AWS (Amazon Web Services) is a comprehensive, evolving cloud computing platform provided by Amazon that includes a mixture of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offering. AWS services can offer an organization tool such as compute power, database storage and content delivery services.

AWS launched in 2006 from the internal infrastructure that Amazon.com built to handle its online retail operations. AWS was one of the first companies to introduce a pay-as-you-go cloud computing model that scales to provide users with compute, storage or throughput as needed.

AWS offers many different tools and solutions for enterprises and software developers that can be used in data centers in up to 190 countries. Groups such as government agencies, education institutions, nonprofits and private organizations can use AWS services.

### How AWS works

AWS is separated into different services; each can be configured in different ways based on the user's needs. Users should be able to see configuration options and individual server maps for an AWS service.

More than 100 services comprise the Amazon Web Services portfolio, including those for compute, databases, infrastructure management, application development and security. These services, by category, include:

- Compute
- Storage databases
- Data management
- Migration
- Hybrid cloud
- Networking
- Development tools
- Management
- Monitoring
- Security
- Governance
- Big data management
- Analytics
- Artificial intelligence (AI)

- Mobile development
- Messages and notification

Amazon EC2 Compute:

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Features of Amazon EC2

Amazon EC2 provides the following features:

- Virtual computing environments, known as instances
- Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types
- Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop, hibernate, or terminate your instance, known as instance store volumes
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as Regions and Availability Zones
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
- Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses
- Metadata, known as tags, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS Cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

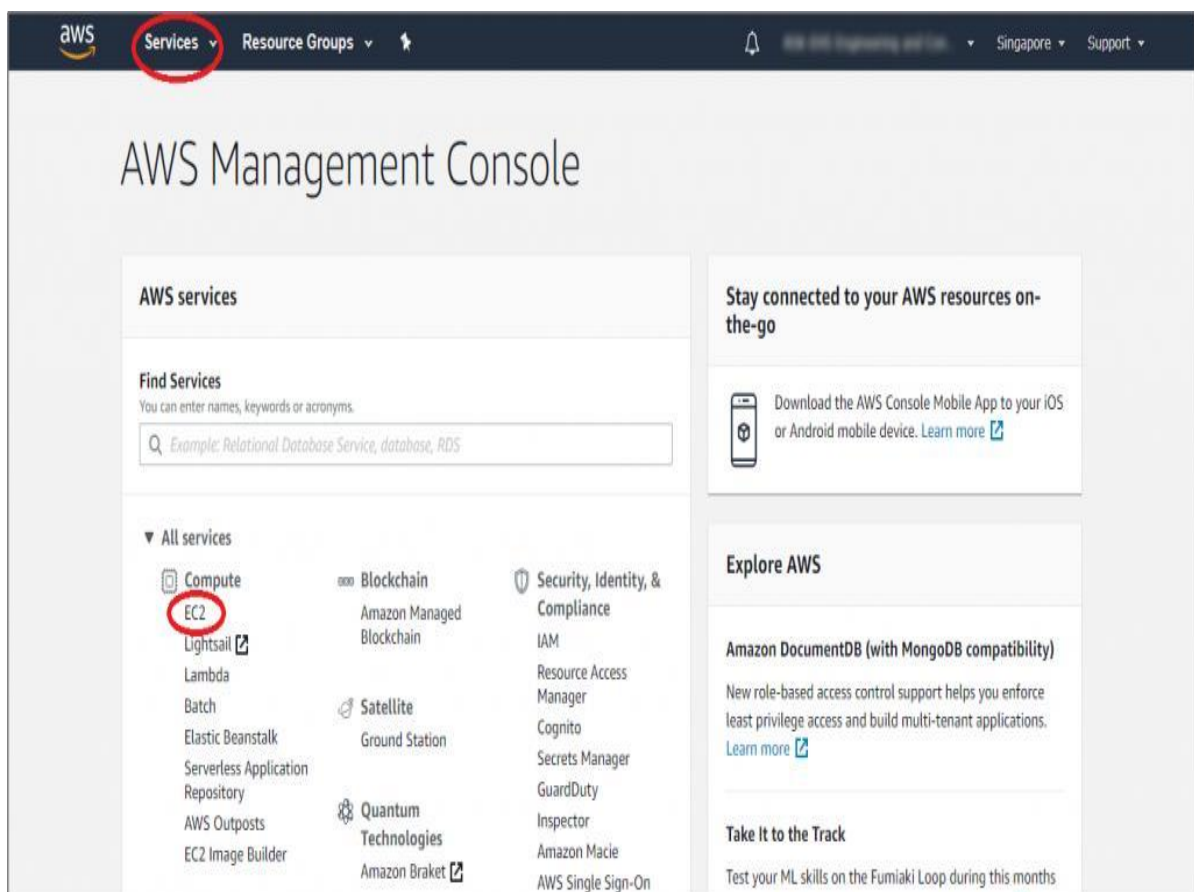
The Steps to Create and Access Platform as a service on Heroku is as follows:

STEP 1:

Login to [AWS Console](#).

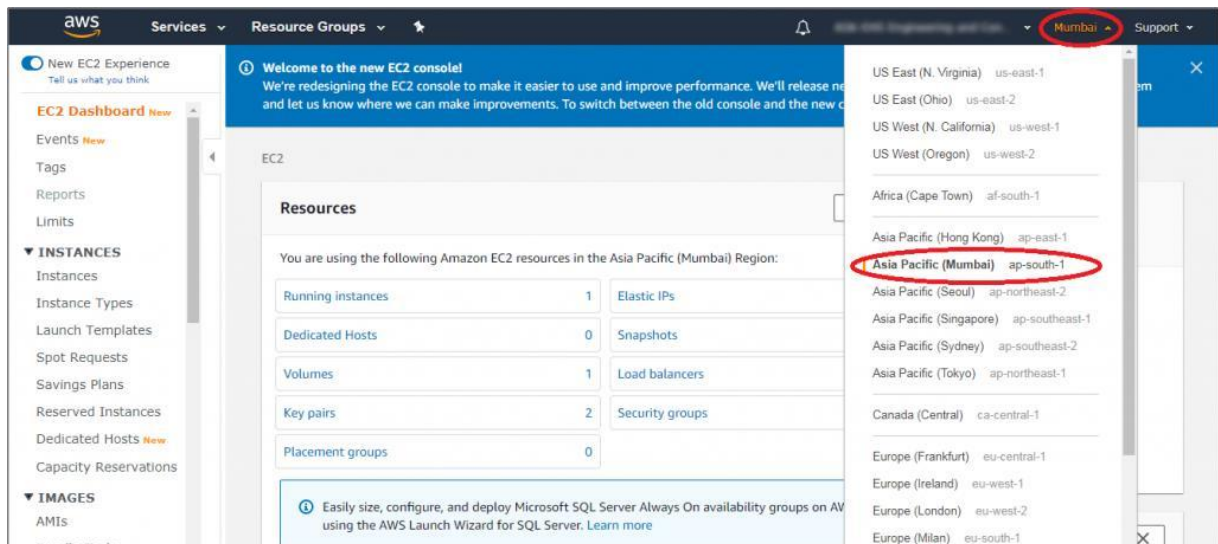
STEP 2:

After login successfully, you are on the dashboard page. Select 'Services' from the top navigation bar and click 'EC2' under the category 'Compute'.



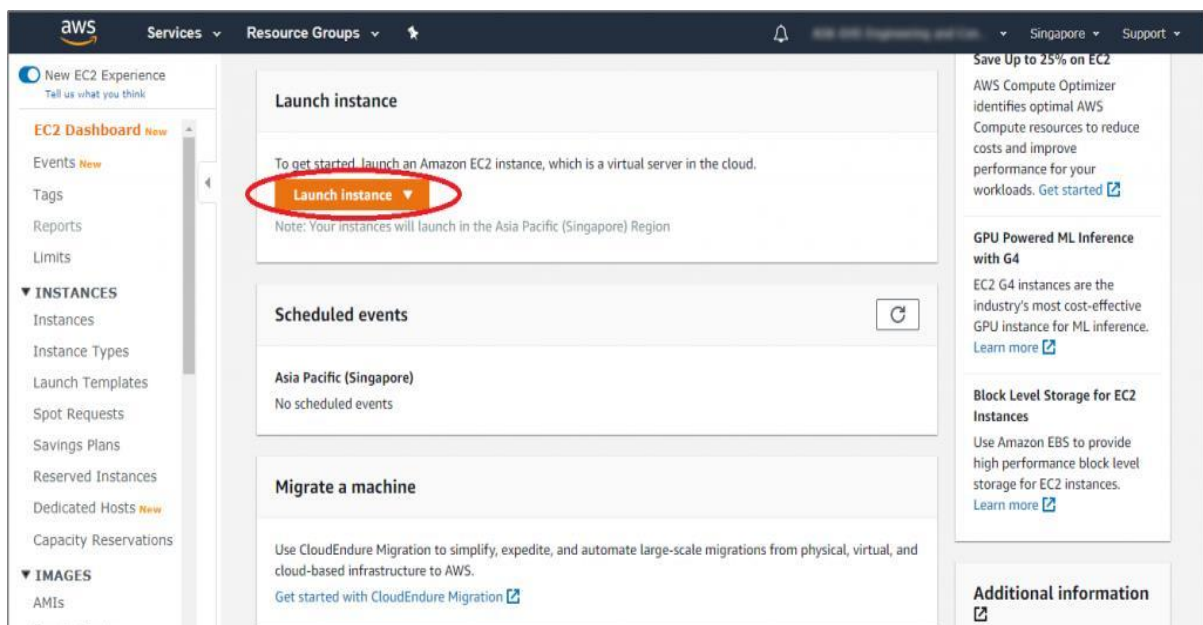
## STEP 3:

On the EC2 instance dashboard page, select the data center closest to your geographical location.



## STEP 4:

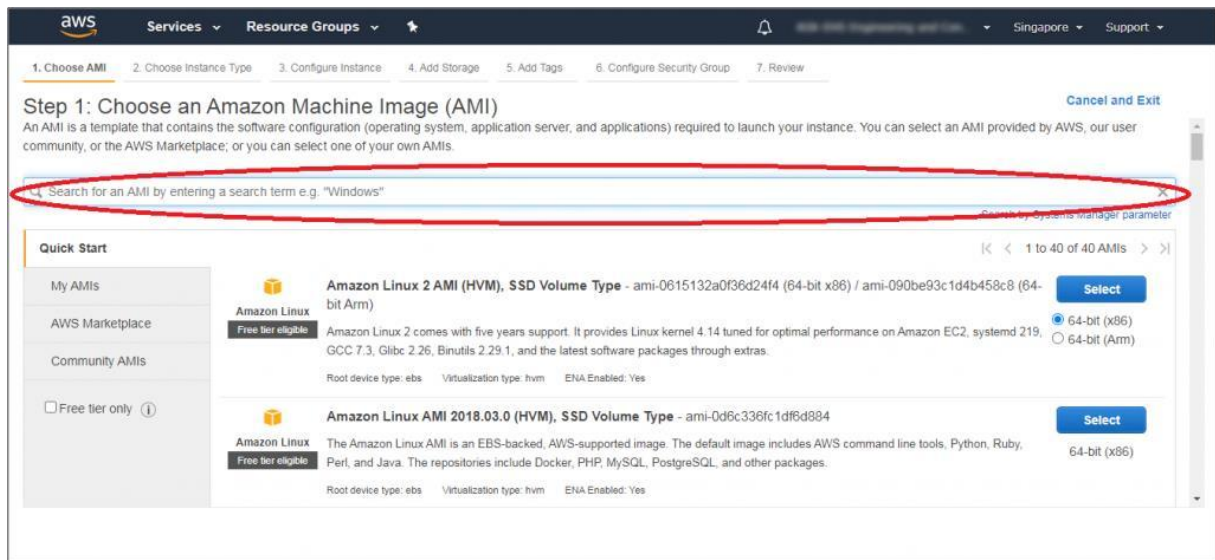
Scroll down, and in the 'Launch Instance' area, check that the selected region appears below the "Launch Instance" button. Click on the 'Launch Instance' button.





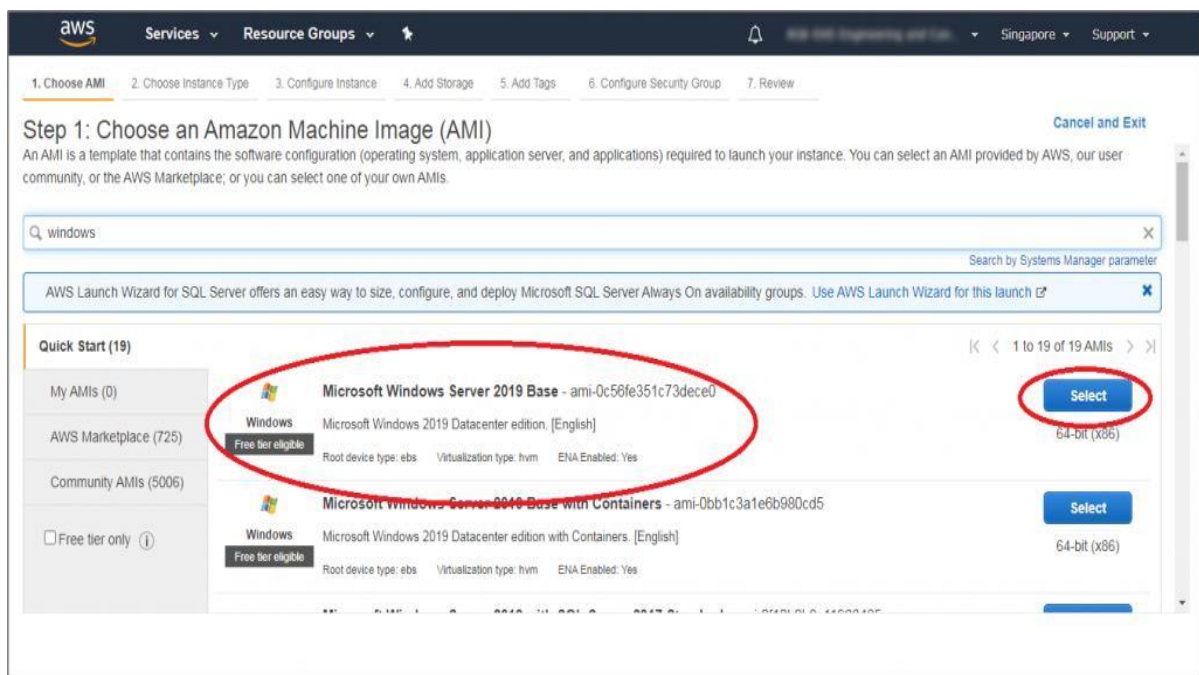
## STEP 5:

The Amazon Machine Image (AMI) page is displayed. You need to select your operating system for your EC2 instance. E.g., if you want to create Windows Server instance, on the page search, type 'windows' as the search option and press 'enter'.



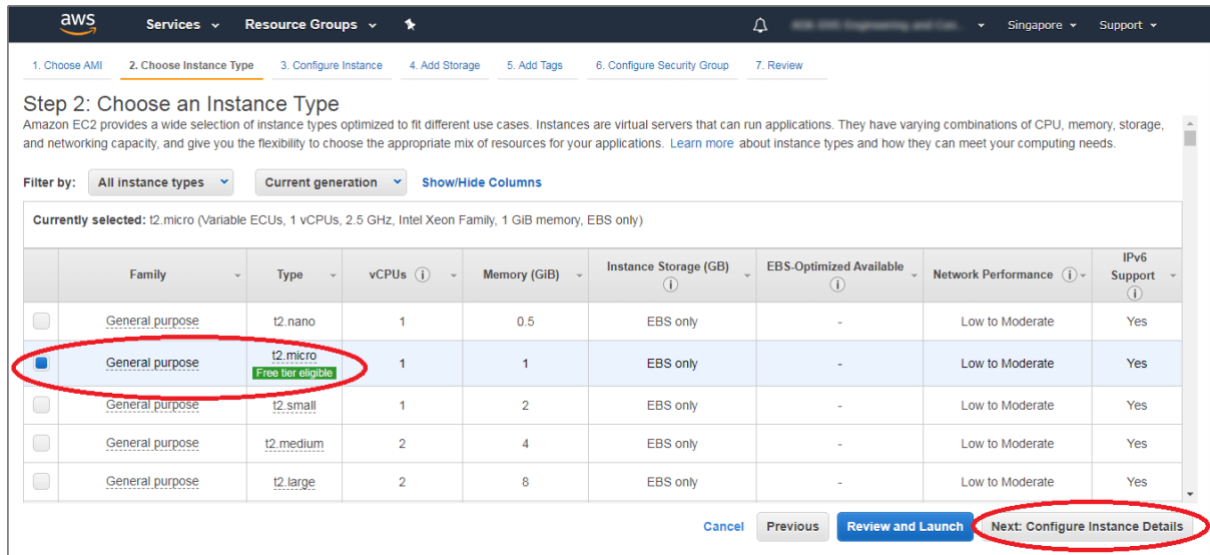
## STEP 6:

From the list of available images, click on the 'Select' button beside the needed image. In this example, we have selected the 'Microsoft Windows Server 2019 Base' image.



## STEP 7:

On the next page, select the type of the instance based on your requirement like RAM, network performance, etc. AWS selects the default micro instance and click the 'Next: Configure Instance Details' button.

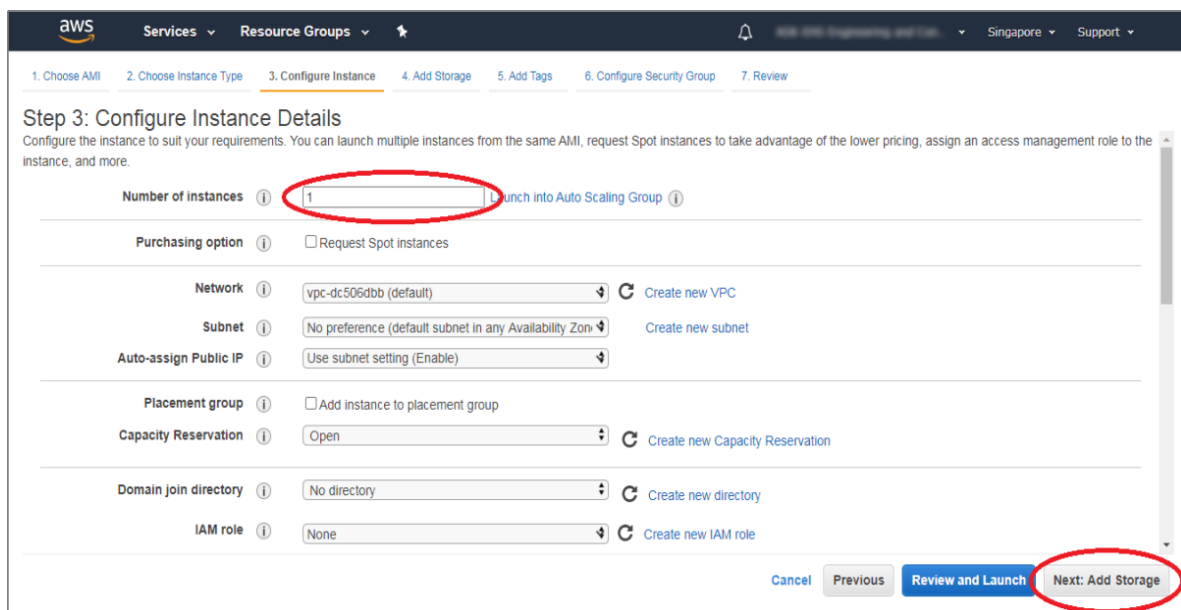


The screenshot shows the AWS console interface for Step 2: Choose an Instance Type. The page displays a table of instance types with the following columns: Family, Type, vCPUs, Memory (GiB), Instance Storage (GB), EBS-Optimized Available, Network Performance, and IPv6 Support. The 't2.micro' instance type is selected and highlighted with a red circle. The 'Next: Configure Instance Details' button is also highlighted with a red circle.

Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
General purpose	t2.micro	1	1	EBS only	-	Low to Moderate	Yes
General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes

## STEP 8:

Enter the number of instances in the 'Configure Instance Details' page. The default value is 1, and you can create multiple instances based on your need. Click on the 'Next: Add Storage' button.



The screenshot shows the AWS console interface for Step 3: Configure Instance Details. The page displays various configuration options for the instance. The 'Number of instances' input field is set to 1 and highlighted with a red circle. The 'Next: Add Storage' button is also highlighted with a red circle.

## STEP 9

On the 'Add Storage' page, you can add a storage device for storing your files. With the Free Tier, by default, you get 30GB of storage per instance. You can buy additional storage by clicking on the 'Add New Volume' button. Next, click on the 'Next: Add Tags' button.

The screenshot shows the 'Step 4: Add Storage' page in the AWS console. The page title is 'Step 4: Add Storage'. Below the title, there is a table with columns: Volume Type, Device, Snapshot, Size (GiB), Volume Type, IOPS, Throughput (MB/s), Delete on Termination, and Encryption. The table has one row for the 'Root' volume with the following values: /dev/sda1, snap-0d6c394fc4d230b9e, 30, General Purpose SSD (gp2), 100 / 3000, N/A, and Not Encrypted. A red circle highlights the 'Add New Volume' button on the left. Another red circle highlights a blue information box that states: 'Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.' A third red circle highlights the 'Next: Add Tags' button at the bottom right.

## STEP 10:

In the 'Add Tags' page, click the 'Click to add the Name tag'.

The screenshot shows the 'Step 5: Add Tags' page in the AWS console. The page title is 'Step 5: Add Tags'. Below the title, there is a table with columns: Key, Value, Instances, and Volumes. The table is empty, and a message states: 'This resource currently has no tags'. Below the message, there is a red circle highlighting the text 'click to add a Name tag'. At the bottom right, there is a 'Next: Configure Security Group' button.

## STEP 11

In the 'Key' field type 'Name' and enter the values as 'Dev Trial'. Click the 'Next: Configure Security Group' button.

The screenshot shows the AWS console interface for Step 5: Add Tags. The page title is "Step 5: Add Tags" and it includes instructions: "A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. A copy of a tag can be applied to volumes, instances or both. Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources." Below the instructions, there is a table for adding tags. The first row has "Name" in the Key field and "Dev Trial" in the Value field. The "Instances" and "Volumes" checkboxes are both checked. At the bottom right, the "Next: Configure Security Group" button is highlighted with a red circle.

Key (128 characters maximum)	Value (256 characters maximum)	Instances	Volumes
Name	Dev Trial	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Buttons: Cancel, Previous, Review and Launch, Next: Configure Security Group

## STEP 12:

On this page, you can choose the configuration to access this EC2 compute instance. By default, you choose the windows image and its automatically defined RDP connection. By default, 'Custom' is selected in the 'Source' dropdown. You can change it to 'Anywhere' to allow any IP address to access your instance. Alternatively, you can select 'My IP', which will restrict its access to your IP or keep 'Custom' which will allow you to specify the IP address range that can access your EC2 instance. Now, click the 'Review and Launch' button.

The screenshot shows the AWS console interface for Step 6: Configure Security Group. The page title is "Step 6: Configure Security Group" and it includes instructions: "A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups." Below the instructions, there is a section for "Assign a security group" with radio buttons for "Create a new security group" (selected) and "Select an existing security group". The "Security group name" field is "launch-wizard-3" and the "Description" field is "launch-wizard-3 created 2020-07-03T14:49:20.484+05:30". Below this is a table for adding rules. The first rule has "RDP" in the Type field, "TCP" in the Protocol field, "3389" in the Port Range field, "Anywhere" in the Source field, and "e.g. SSH for Admin Desktop" in the Description field. The "Source" dropdown is open, showing options: "Anywhere", "Custom", "My IP". At the bottom right, the "Review and Launch" button is highlighted with a red circle.

Type	Protocol	Port Range	Source	Description
RDP	TCP	3389	Anywhere	e.g. SSH for Admin Desktop

Buttons: Cancel, Previous, Review and Launch

## STEP 13:

On this page, review all of your selected configurations and click the ‘Launch’ button.

aws Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

**⚠ Improve your instances' security. Your security group, launch-wizard-3, is open to the world.**

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

▼ AMI Details Edit AMI

**Microsoft Windows Server 2019 Base - ami-0c56fe351c73dece0**

Free tier eligible Microsoft Windows 2019 Datacenter edition. [English]  
Root Device Type: ebs Virtualization type: hvm

If you plan to use this AMI for an application that benefits from Microsoft License Mobility, fill out the [License Mobility Form](#). [Don't show me this again](#)

▼ Instance Type Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Cancel Previous Launch

## STEP 14

A ‘Key Pair’ window is prompted. Select the ‘Create a new key pair’ option from the dropdown, and in the next text field, type a unique key pair name. Click the ‘Download Key Pair’ button. It will download the key pair file with ‘.pem’ extension on your computer. It’s crucial for generating unique passwords for the EC2 instance.

aws Services Resource Groups

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 7: Review Instance Launch

You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

▼ AMI Details Edit AMI

**Microsoft Windows Server 2019 Base - ami-0c56fe351c73dece0**

Free tier eligible Microsoft Windows 2019 Datacenter edition. [English]  
Root Device Type: ebs Virtualization type: hvm

If you plan to use this AMI for an application that benefits from Microsoft License Mobility, fill out the [License Mobility Form](#). [Don't show me this again](#)

▼ Instance Type Edit instance type

Instance Type	ECUs
t2.micro	Variable

▼ Security Groups Edit security groups

**Select an existing key pair or create a new key pair** X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. [Learn more about removing existing key pairs from a public AMI.](#)

Create a new key pair

**Key pair name**  
DevTrial

Download Key Pair

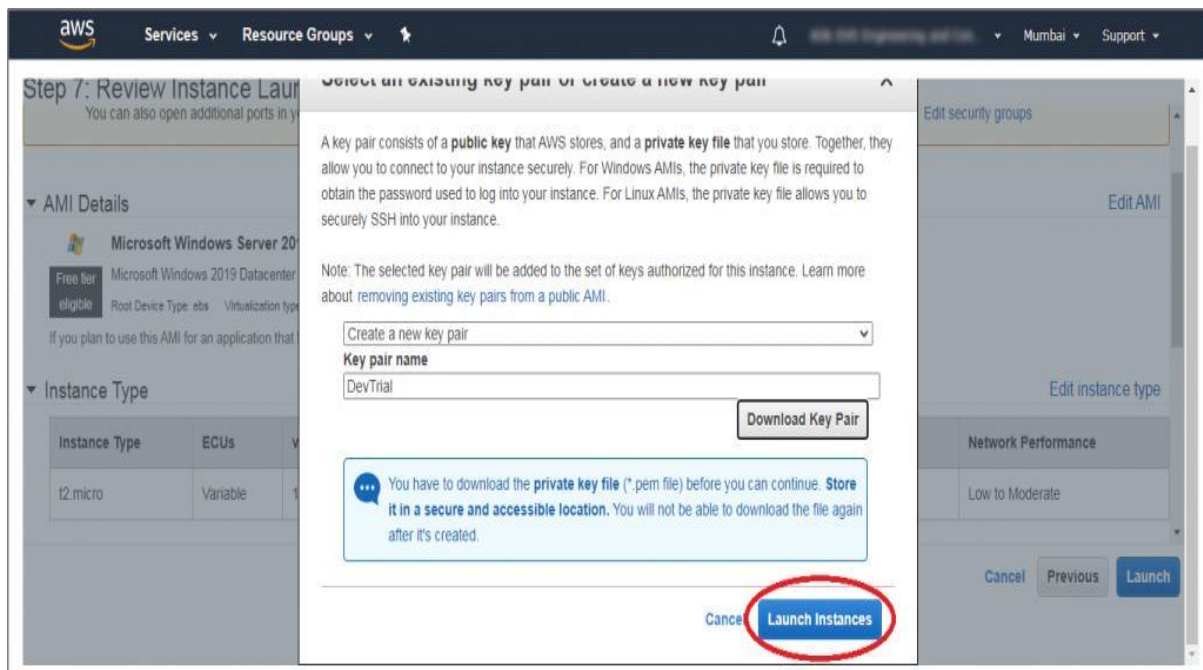
**ⓘ** You have to download the **private key file** (\*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download it again after it's created.

Cancel Launch Instances

Cancel Previous Launch

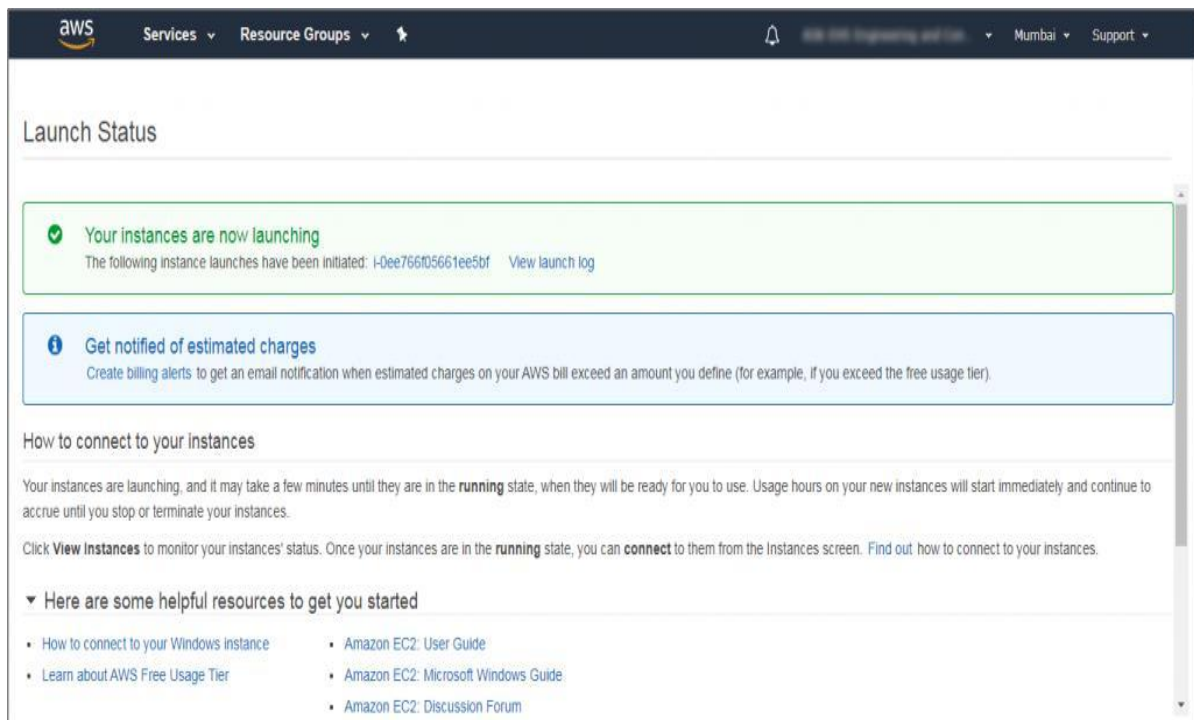
## STEP 15:

Click the 'Launch Instances' button, and it's directed to the launch status page.



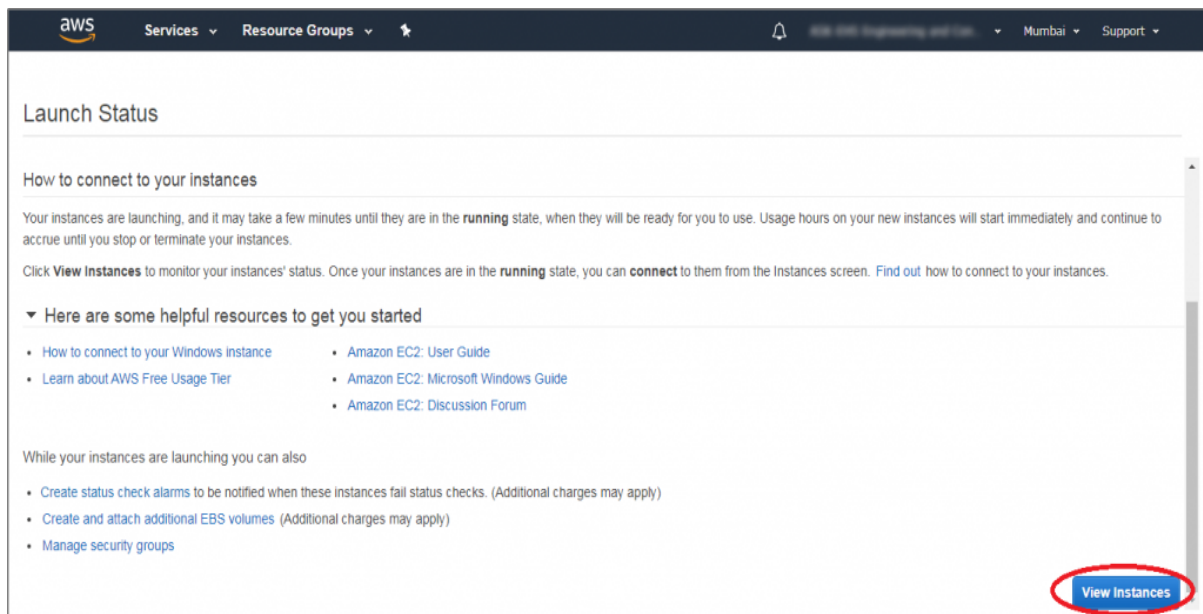
## STEP 16:

The 'Launch Status' page is displayed.



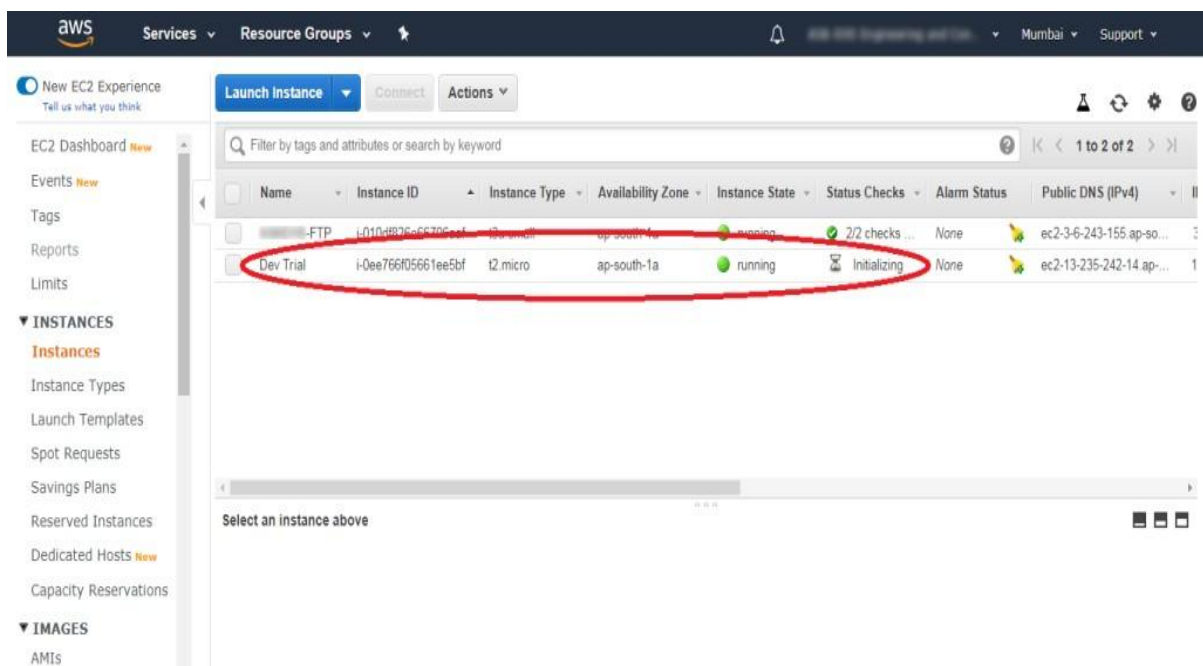
## STEP 17:

Scroll down the page and click on the 'View Instances' button.



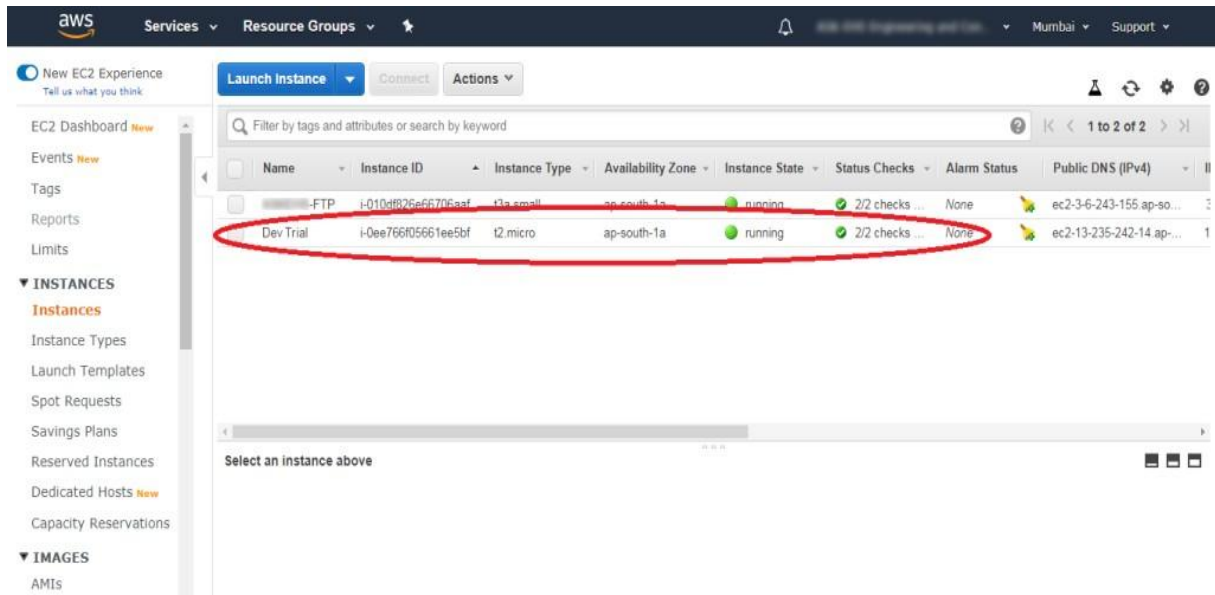
## STEP 18:

All the available instances are displayed. The following image shows that the 'Dev Trial' instance is initializing.



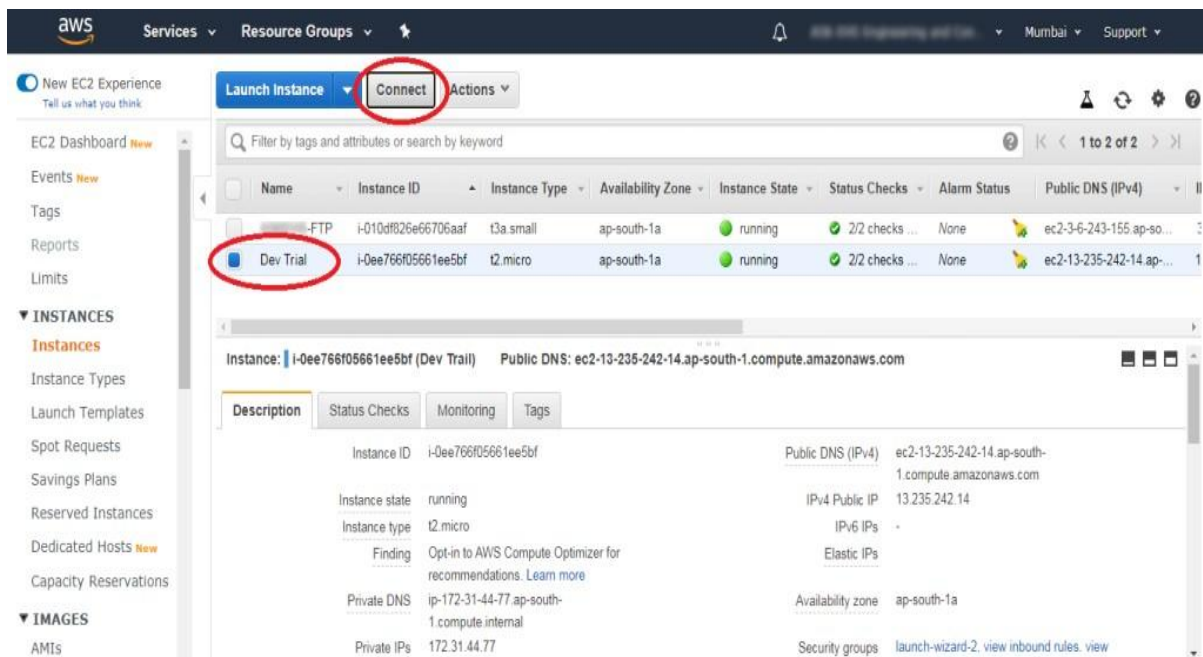
## STEP 19:

The new instance is shown running in the list of instances.



## STEP 20:

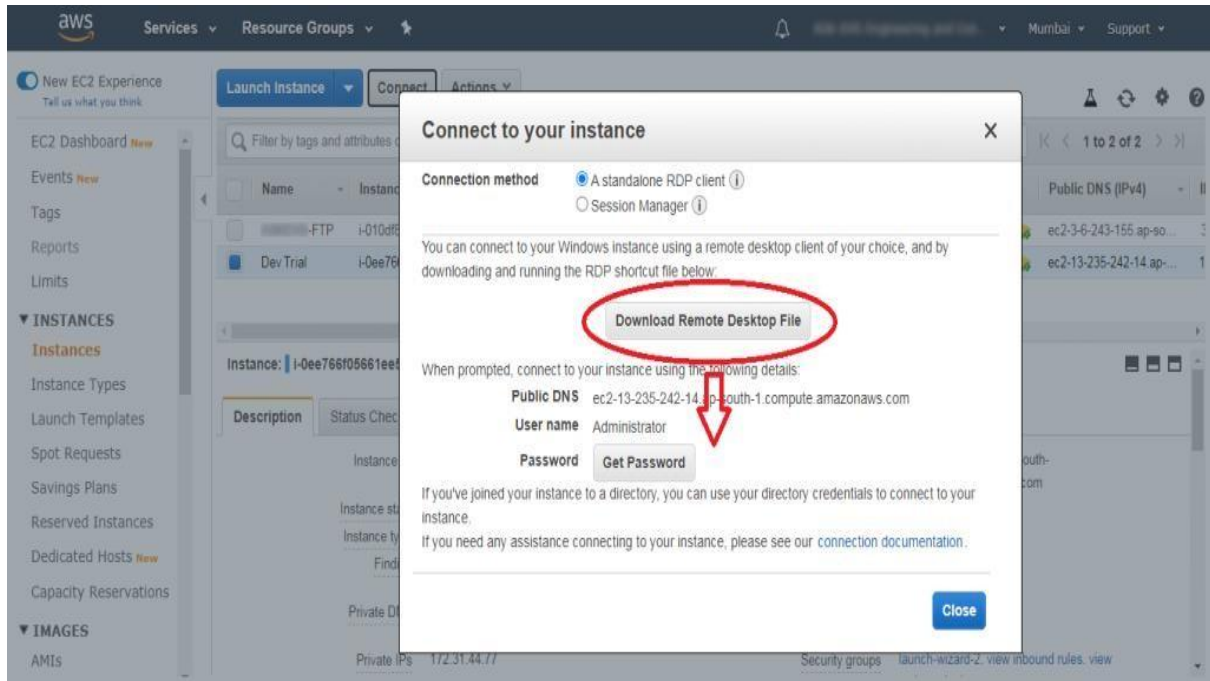
Select the newly created instance ('Dev Trial' in this case) and click on the 'Connect' button.





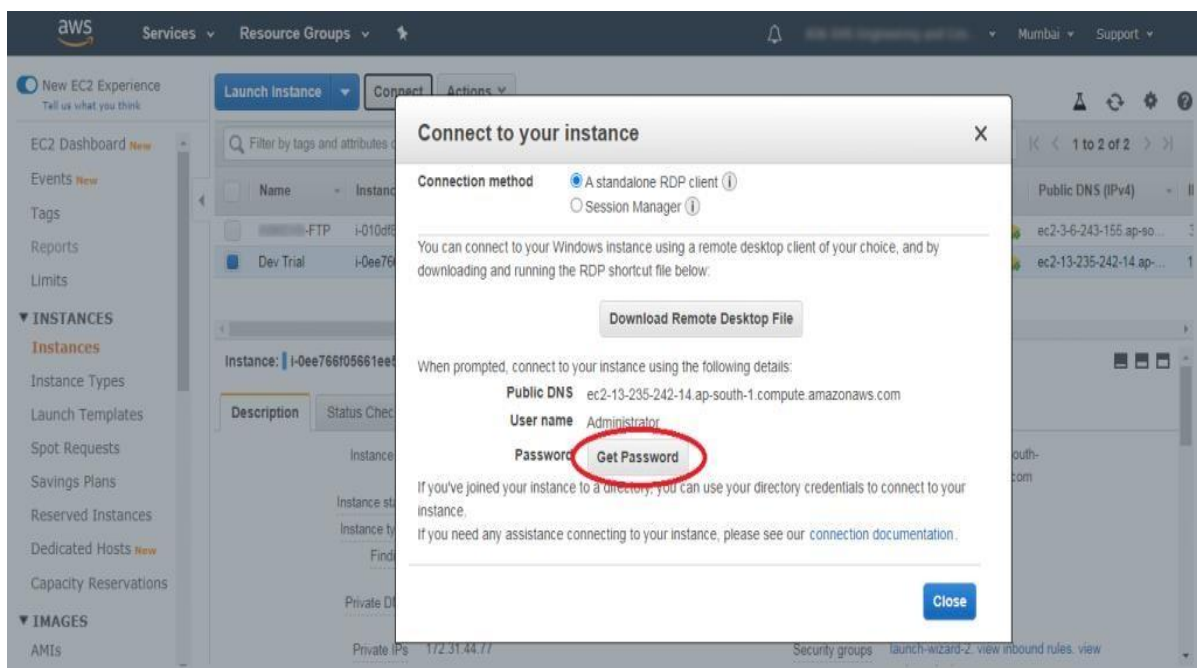
## STEP 21:

In the 'Connect to your instance' window, click the 'Download the Remote Desktop File' button to download the RDP file. Save it on your computer.



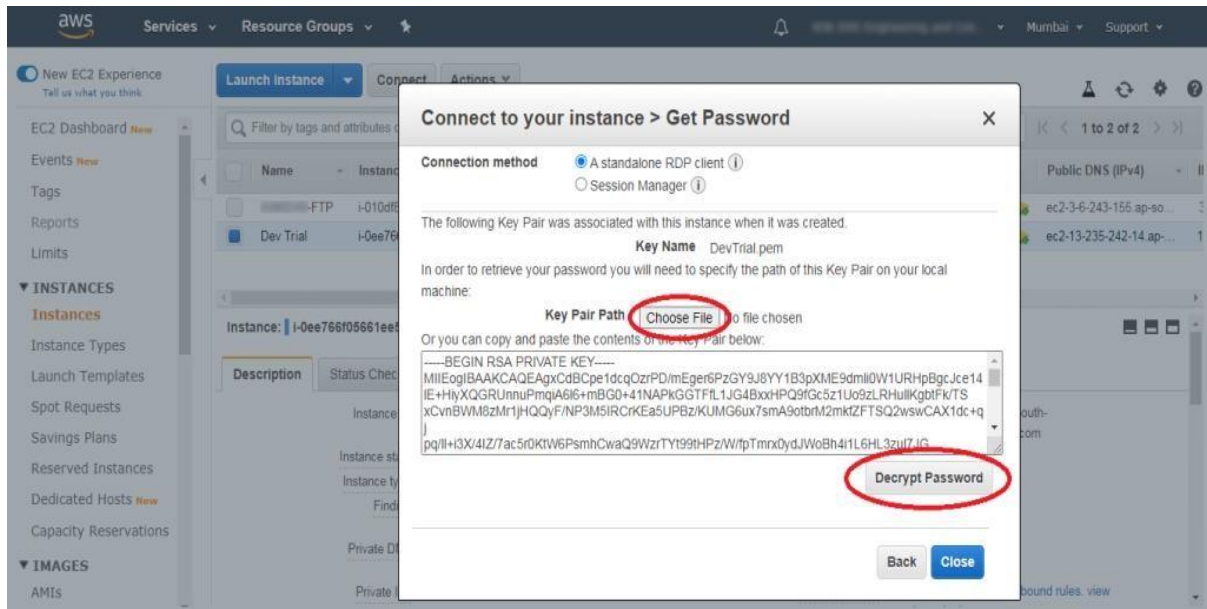
## STEP 22:

Click the 'Get Password' button.



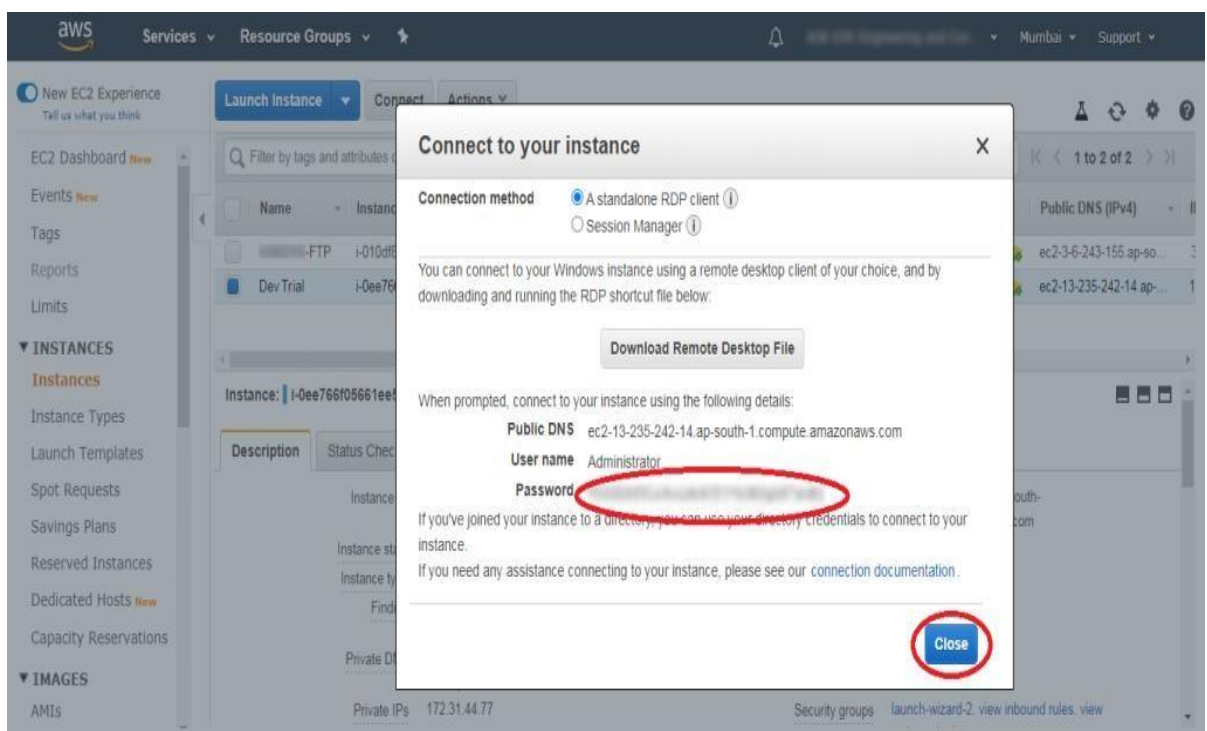
## STEP 23:

Click on the 'Choose File' button and select the already downloaded key pair file. Click on the 'Decrypt Password' button.



## STEP 24:

The decrypted password will appear. Copy the Password and paste in any word processor file. Click the 'Close' button.



STEP 25

Now, go to your downloaded RDP file. The RDP software prompts to connect with EC2 instance. Click the 'Connect' button. In the popped up 'Windows Login' window, paste the previously copied password from the AWS console and click the 'OK' button. It will start to establish connections with the AWS EC2 instance.

**Conclusion:** Thus, we Studied and implemented Infrastructure as a Service using Amazon AWS.

## Experiment No: 2

**Title:** Implementation of Software as a Service.

**Aim -** To study and implementation of Software as a Service.

### Theory –

SaaS is also known as "On-Demand Software". It is a software distribution model in which services are hosted by a cloud service provider. These services are available to end-users over the internet so, the end-users do not need to install any software on their devices to access these services.

There are the following services provided by SaaS providers -

**Business Services** - SaaS Provider provides various business services to start-up the business. The SaaS business services include ERP (Enterprise Resource Planning), CRM (Customer Relationship Management), billing, and sales.

**Document Management** - SaaS document management is a software application offered by a third party (SaaS providers) to create, manage, and track electronic documents.

Example: Slack, Samepage, Box, and Zoho Forms.

**Social Networks** - As we all know, social networking sites are used by the general public, so social networking service providers use SaaS for their convenience and handle the general public's information.

**Mail Services** - To handle the unpredictable number of users and load on e-mail services, many e-mail providers offering their services using SaaS.



## **Advantages of SaaS cloud computing layer**

### **1) SaaS is easy to buy:**

SaaS pricing is based on a monthly fee or annual fee subscription, so it allows organizations to access business functionality at a low cost, which is less than licensed applications.

Unlike traditional software, which is sold as a licensed based with an up-front cost (and often an optional ongoing support fee), SaaS providers are generally pricing the applications using a subscription fee, most commonly a monthly or annually fee.

### **2. One to Many:**

SaaS services are offered as a one-to-many model means a single instance of the application is shared by multiple users.

### **3. Less hardware required for SaaS:**

The software is hosted remotely, so organizations do not need to invest in additional hardware.

### **4. Low maintenance required for SaaS:**

Software as a service removes the need for installation, set-up, and daily maintenance for the organizations. The initial set-up cost for SaaS is typically less than the enterprise software. SaaS vendors are pricing their applications based on some usage parameters, such as a number of users using the application. So, SaaS does easy to monitor and automatic updates.

### **5. No special software or hardware versions required:**

All users will have the same version of the software and typically access it through the web browser. SaaS reduces IT support costs by outsourcing hardware and software maintenance and support to the IaaS provider.

### **6. Multidevice support:**

SaaS services can be accessed from any device such as desktops, laptops, tablets, phones, and thin clients.

### **7. API Integration:**

SaaS services easily integrate with other software or services through standard APIs.

### **8. No client-side installation:**

SaaS services are accessed directly from the service provider using the internet connection, so do not need to require any software installation.

## Disadvantages of SaaS cloud computing layer

### 1) Security:

Actually, data is stored in the cloud, so security may be an issue for some users. However, cloud computing is not more secure than in-house deployment.

### 2) Latency issue:

Since data and applications are stored in the cloud at a variable distance from the end-user, there is a possibility that there may be greater latency when interacting with the application compared to local deployment. Therefore, the SaaS model is not suitable for applications whose demand response time is in milliseconds.

### 3) Total Dependency on Internet:

Without an internet connection, most SaaS applications are not usable.

### 4) Switching between SaaS vendors is difficult:

Switching SaaS vendors involves the difficult and slow task of transferring the very large data files over the internet and then converting and importing them into another SaaS also.

## Popular SaaS Providers:



The below table shows some popular SaaS providers and services that are provided by them -

Provider	Services
Salseforce.com	On-demand CRM solutions
Microsoft Office 365	Online office suite
Google Apps	Gmail, Google Calendar, Docs, and sites
NetSuite	ERP, accounting, order management, CRM, Professionals Services Automation (PSA), and e-commerce applications.
GoToMeeting	Online meeting and video-conferencing software
Constant Contact	E-mail marketing, online survey, and event marketing
Oracle CRM	CRM applications
Workday, Inc	Human capital management, payroll, and financial management.

### Example:

Google Workspace (formerly known as Google Apps and later G Suite) is a collection of cloud computing, productivity and collaboration tools, software and products developed and marketed by Google. It was first launched in 2006 as Google Apps for Your Domain and rebranded as G Suite in 2016. Google Workspace consists of Gmail, Contacts, Calendar, Meet and Chat for communication; Currents for employee engagement; Drive for storage; and the Google Docs suite for content creation. An Admin Panel is provided for managing users and services. Depending on edition Google Workspace may also include the digital interactive whiteboard Jam board and an option to purchase such add-ons as the telephony service Voice. The education edition adds a learning platform Google Classroom and today has the name Workspace for Education.

While most of these services are individually available at no cost to consumers who use their free Google (Gmail) accounts, Google Workspace adds enterprise features such as custom email addresses at a domain (e.g., @yourcompany.com), an option for unlimited Drive storage, additional administrative tools and advanced settings, as well as 24/7 phone and email support

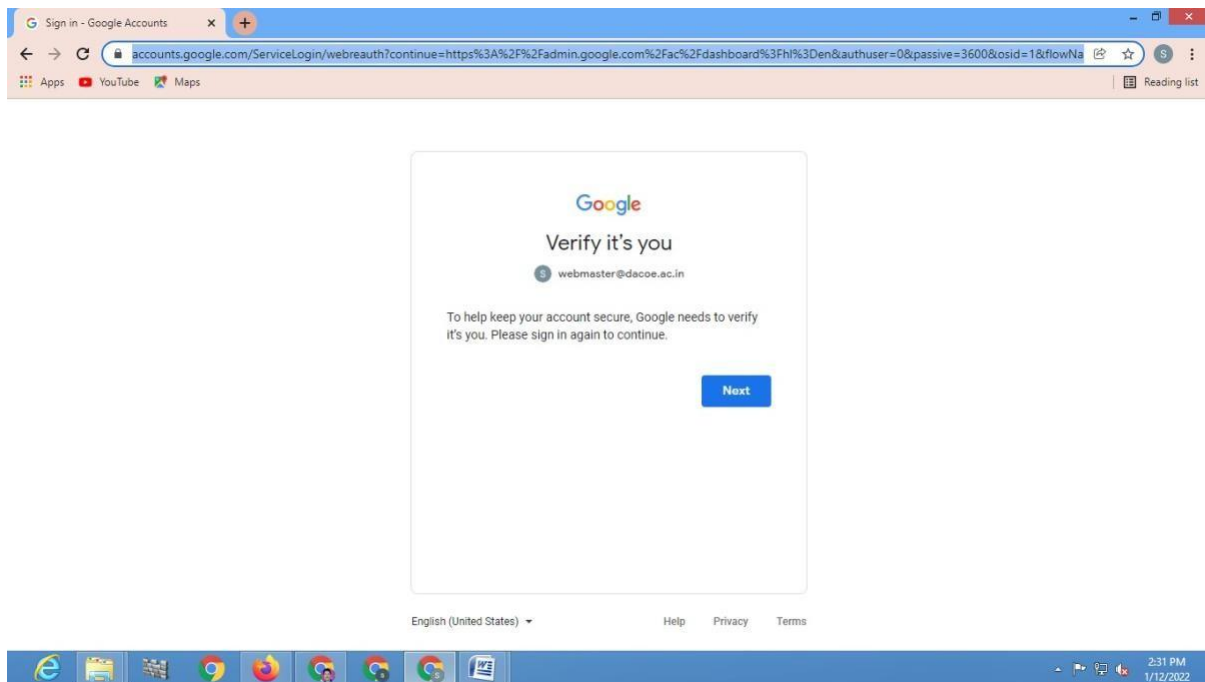
## How to start using G Suite:

1) Sign in to your G Suite administrator account and set up your first users. You can manage all of the G Suite services for your business, including mobile device management, data migration, setting password requirements, and much more. This guide walks you through the admin sign-in process.

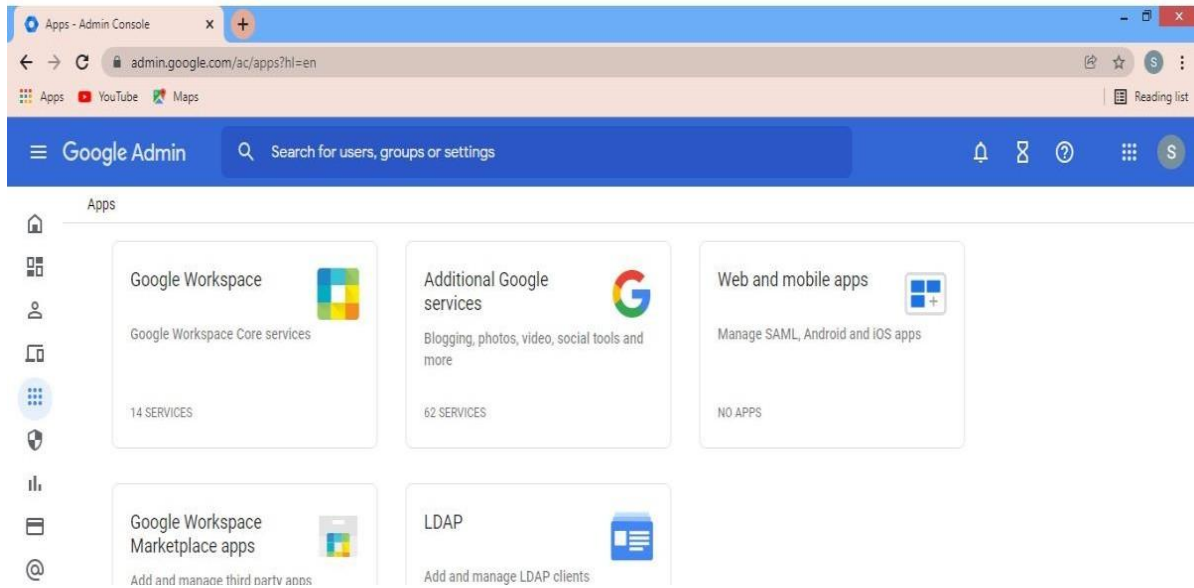
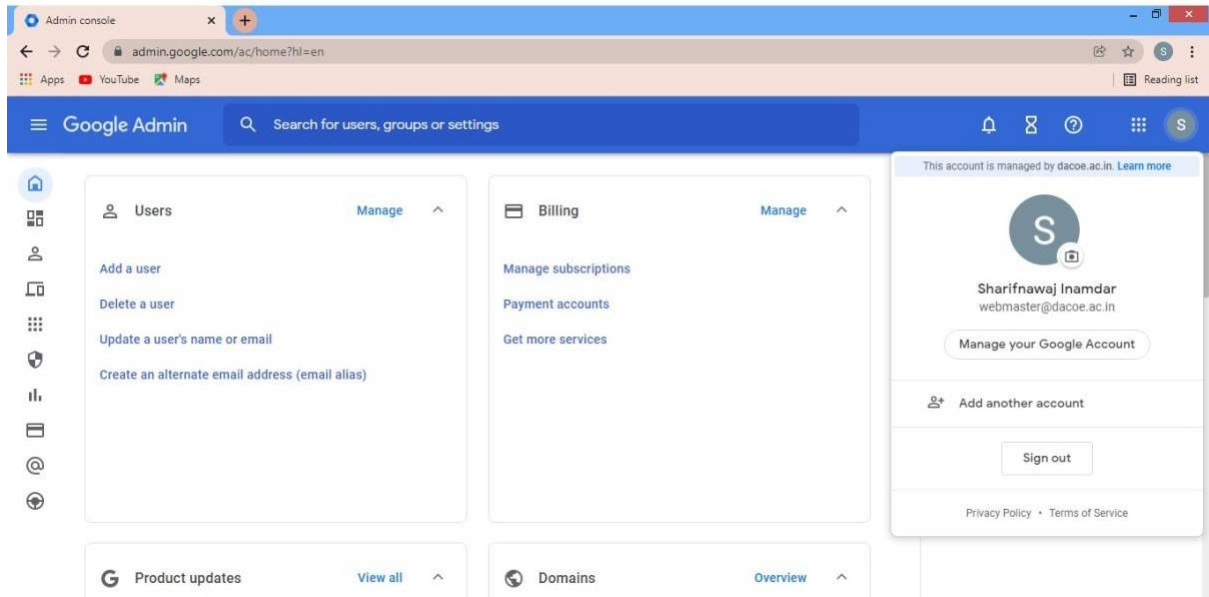
After you sign in, you will need to set up your G Suite account by

- 1.) [verifying](#) you own your domain,
- 2.) adding user accounts for your employees, and
- 3.) when all employees have been added, setting up Gmail as your email client by [updating your MX records](#). Don't worry, none of these changes will affect your website, if you have one.

## Screenshots :







**Conclusion:** Thus, we have studied and implemented Software as a Service (SaaS).

### **Experiment No: 3**

**Title:** Working and Implementation of Platform as a service.

**Aim:** To Study and implement Platform as a Service using Heroku.

Theory:

Heroku is a cloud platform as a service (PaaS) supporting several programming languages. One of the first cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Heroku is said to be a polyglot platform as it has features for a developer to build, run and scale applications in a similar manner across most languages.

Services Provided By Heroku:

#### **The Heroku Platform**

The Heroku network runs the customer's apps in virtual containers which execute on a reliable runtime environment. Heroku calls these containers "Dynos". These Dynos can run code written in Node, Ruby, PHP, Go, Scala, Python, Java, or Clojure. Heroku also provides custom buildpacks with which the developer can deploy apps in any other language. Heroku lets the developer scale the app instantly just by either increasing the number of dynos or by changing the type of dyno the app runs in.

Heroku Postgres

Heroku Postgres is the Cloud database (DBaaS) service for Heroku based on PostgreSQL. Heroku Postgres provides features like continuous protection, rollback, and high availability; also forks, followers, and dataclips.

Heroku Redis

Heroku Redis is the customized Redis from Heroku to provide a better developer experience. It is fully managed and is provided as a service by Heroku. It helps in managing instances with a CLI, associate data with Postgres to gain business insights using SQL tools, and lets customer gain performance visibility.

Heroku Teams

Heroku Teams is a team management tool which provides collaboration and controls to bring a customer's developers, processes, and tools together in order to build better software. With Heroku Teams, teams can self-organize, add, and manage members, get fine-grained control with app-level permissions and also use collaboration tools like Heroku Pipelines. It also provides delegated administration and centralized billing.

Heroku Enterprise

Heroku Enterprise provides services to large companies which help them to improve collaboration among different teams. It provides a set of features like fine-grained access controls, identity federation, and private spaces to manage their enterprise application development process, resources, and users.

## Heroku Connect

Heroku Connect lets users create Heroku apps that can easily integrate with [Salesforce](#) deployments at scale. This is done by having a seamless data synchronization between Heroku Postgres databases and Salesforce organizations.

## Heroku Elements

Heroku Elements provides users with Add-ons (tools and services for developing, extending, and operating the app), Buildpacks (which automate the build processes for the preferred languages and frameworks) and Buttons (a tool for the one-click provisioning, configuring, and deployment of third party components, libraries and patterns).

### Advantages:

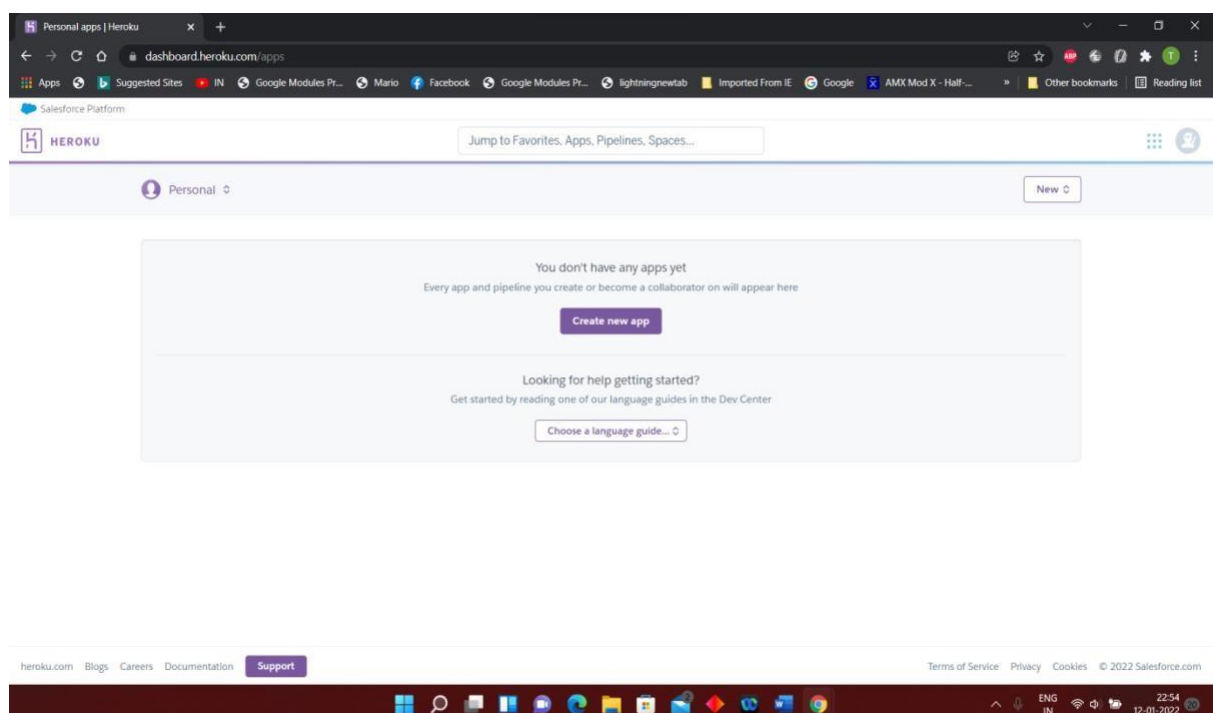
- Easy deployment
- Good for small projects
- Add-ons
- Less maintenance
- Documentation

### Disadvantage:

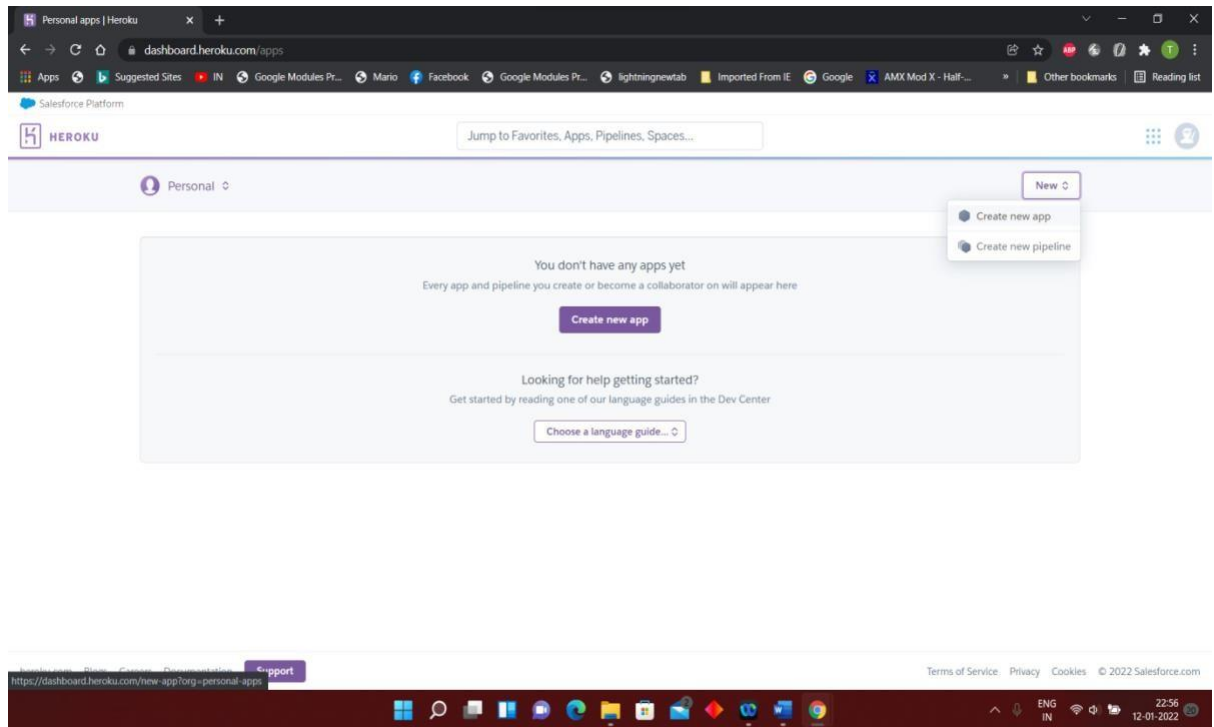
- Price when you need to scale up on your projects

### The Steps to Create and Access Platform as a service on Heroku is as follows:

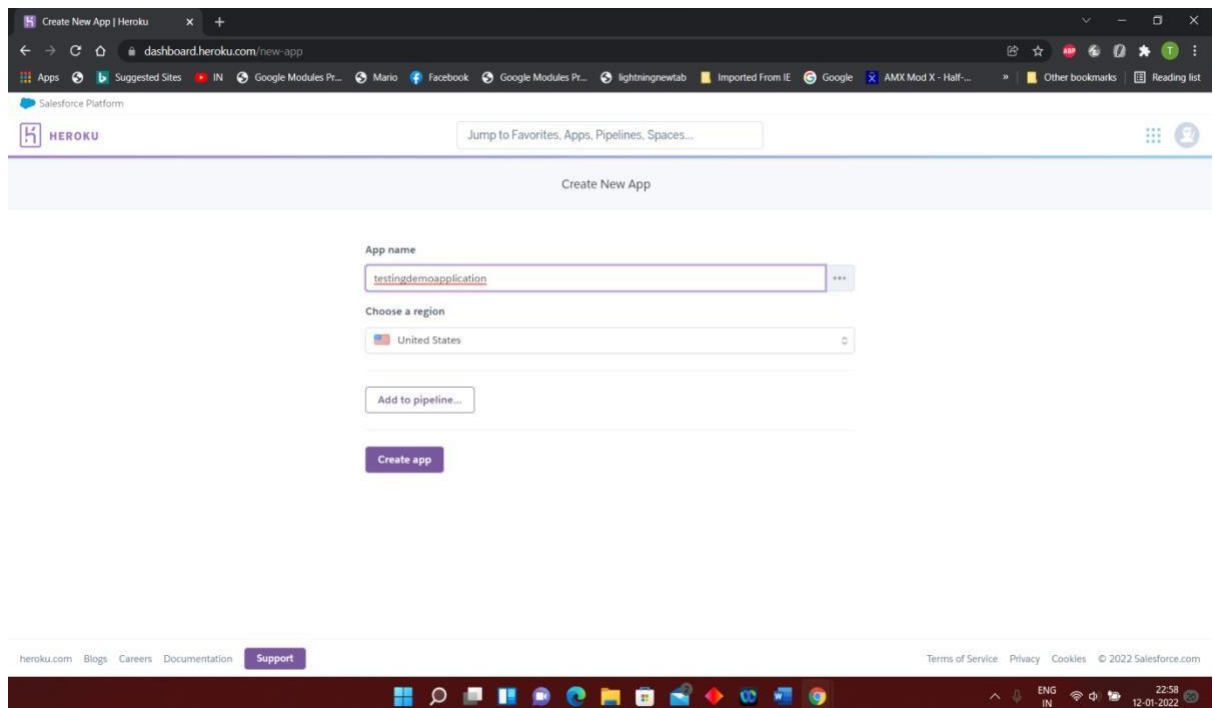
#### 1) Login to Heroku.



2) Click on “New” -> “Create new app” button at top right.



3) Give a name to your app and click “Create app”. Note that this needs to be a unique name.

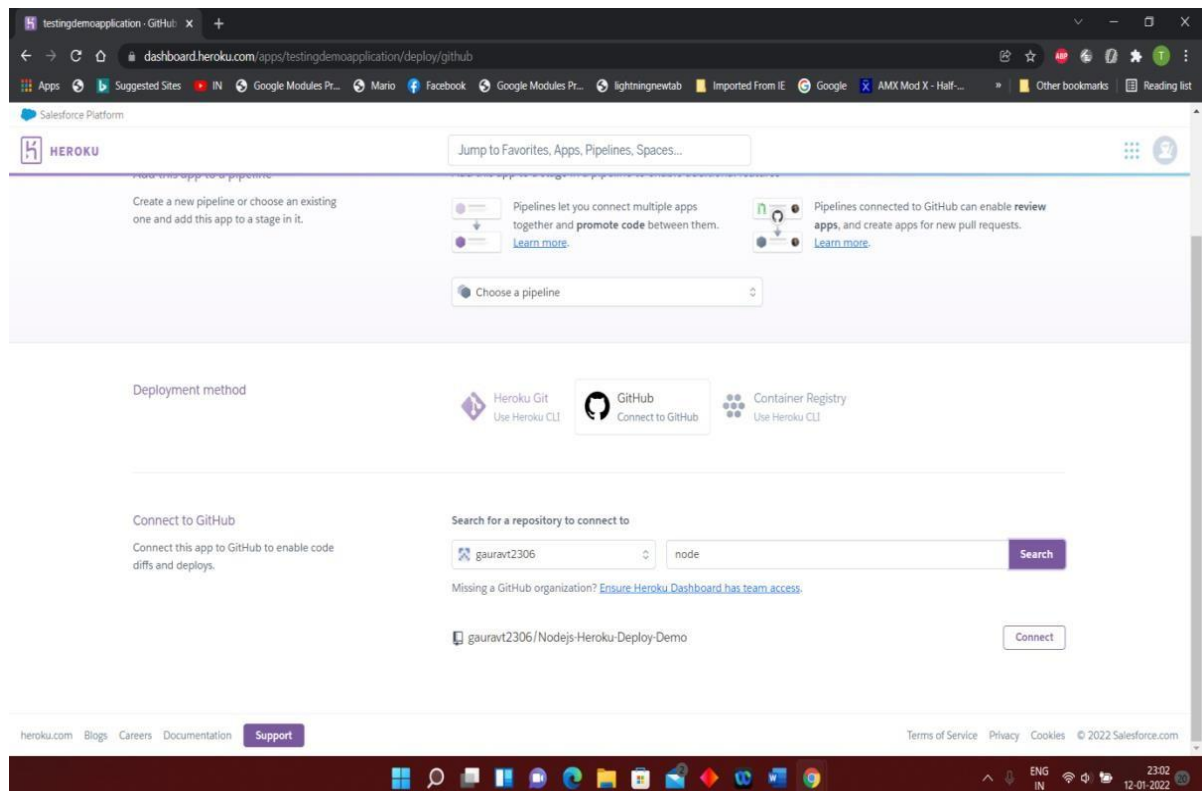


4) You will be redirected to a new page where find the “Deployment method” section and click on Github.

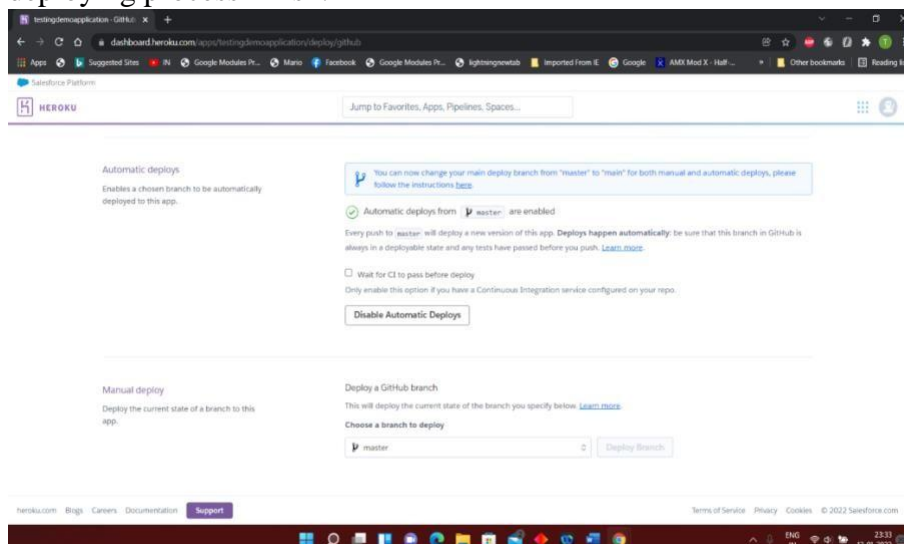
1) You will be redirected to a new page where find the “Deployment method” section and click on Github.

(You can fork this repository for testing: <https://github.com/gauravt2306/Nodejs-Heroku-Deploy-Demo>)

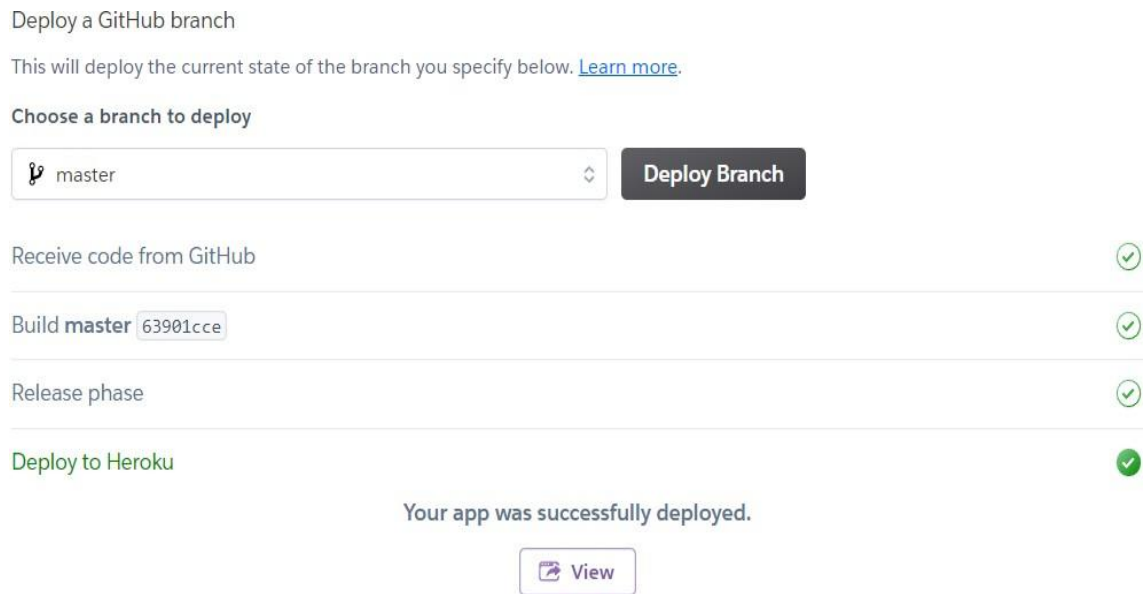
Enter the repository name -> click search -> Select correct repository -> Click connect.



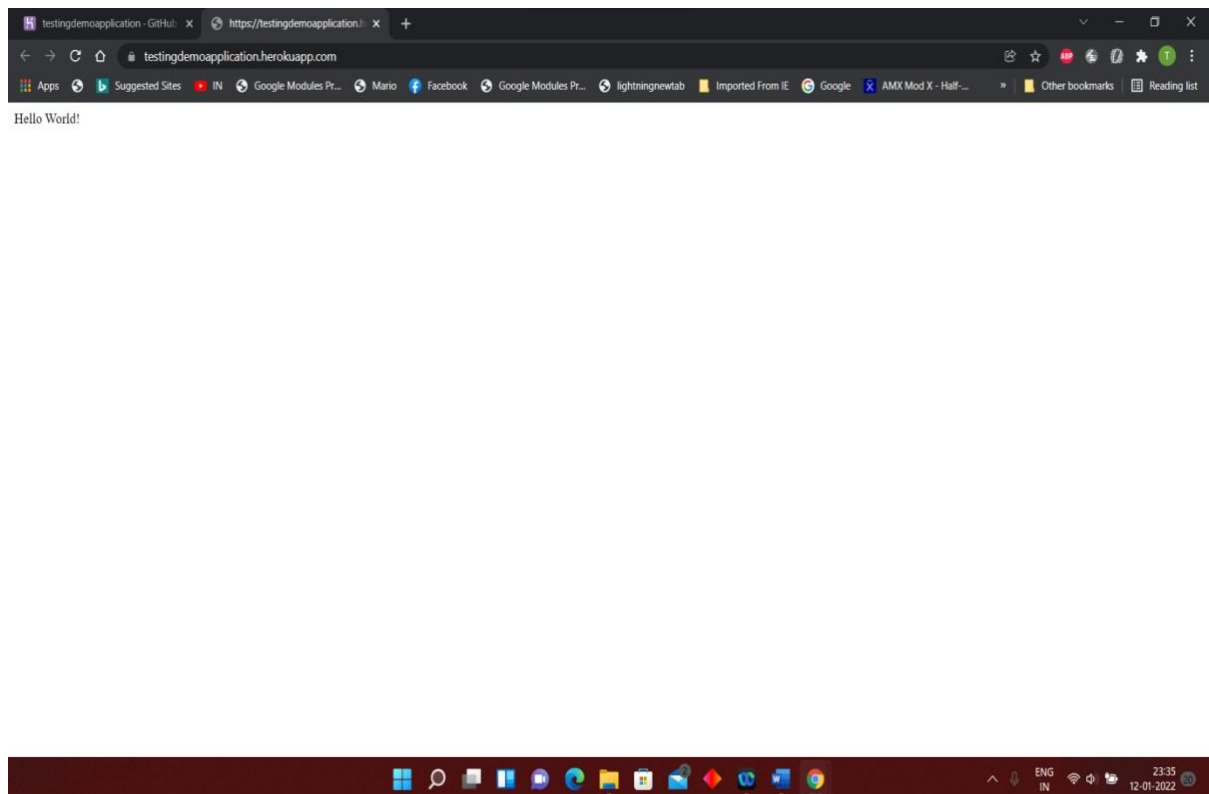
2) Click on “Enable automatic deploys” button which would mean that whenever you add any code to your GitHub repository, Heroku will update your hosted app too. And finally, click on “Deploy branch ”. Wait for a couple of minutes and let the deploying process finish.



3) After successfully deploying the website you will get a success message as shown.



4) Click on View To Open Web Application.



**Conclusion:** Thus, we have Studied and implemented Platform as a Service using Heroku

## Experiment No: 4

**Title:** Implementation of Storage as a Service.

**Aim:** To study Implementation of Storage as a Service by using AWS.

**Theory:** Amazon EC2 provides you with flexible, cost effective, and easy-to-use data storage options for your instances. Each option has a unique combination of performance and durability. These storage options can be used independently or in combination to suit your requirements.

After reading this section, you should have a good understanding about how you can use the data storage options supported by Amazon EC2 to meet your specific requirements. These storage options include the following:

- [Amazon Elastic Block Store](#)
- [Amazon EC2 instance store](#)
- [EFS](#)
- [S3](#)

The following figure shows the relationship between these storage options and your instance.

### Amazon EBS

Amazon EBS provides durable, block-level storage volumes that you can attach to a running instance. You can use Amazon EBS as a primary storage device for data that requires frequent and granular updates. For example, Amazon EBS is the recommended storage option when you run a database on an instance.

### Amazon EC2 instance store

Many instances can access storage from disks that are physically attached to the host computer. This disk storage is referred to as *instance store*. Instance store provides temporary block-level storage for instances. The data on an instance store volume persists only during the life of the associated instance; if you stop, hibernate, or terminate an instance, any data on instance store volumes is lost

### Amazon EFS file system

Amazon EFS provides scalable file storage for use with Amazon EC2. You can create an EFS file system and configure your instances to mount the file system. You can use an EFS file system as a common data source for workloads and applications running on multiple instances.

## Amazon S3

Amazon S3 provides access to reliable and inexpensive data storage infrastructure. It is designed to make web-scale computing easier by enabling you to store and retrieve any amount of data, at any time, from within Amazon EC2 or anywhere on the web

Implementation of Storage as a Service by using AWS S3:

What is AWS S3?

Amazon Simple Storage Service (S3) is storage for the internet. It is designed for large-capacity, low-cost storage provision across multiple geographical regions. Amazon S3 provides developers and IT teams with Secure, Durable and Highly Scalable object storage.

S3 is Secure because AWS provides:

- Encryption to the data that you store. It can happen in two ways:
  - Client Side Encryption
  - Server Side Encryption
- Multiple copies are maintained to enable regeneration of data in case of data corruption
- *Versioning*, wherein each edit is archived for a potential retrieval.

S3 is Durable because:

- It regularly verifies the integrity of data stored using checksums e.g. if S3 detects there is any corruption in data, it is immediately repaired with the help of replicated data.
- Even while storing or retrieving data, it checks incoming network traffic for any corrupted data packets.

S3 is Highly Scalable, since it automatically scales your storage according to your requirement and you only pay for the storage you use.

What kind and how much of data one can store in AWS S3?

You can store virtually any kind of data, in any format, in S3 and when we talk about capacity, the volume and the number of objects that we can store in S3 are unlimited.

\*An object is the fundamental entity in S3. It consists of data, key and metadata.

When we talk about data, it can be of two types-

- Data which is to be accessed frequently.
- Data which is accessed not that frequently.

Therefore, Amazon came up with 3 storage classes to provide its customers the best experience and at an affordable cost.



Let's understand the 3 storage classes with a "health-care" use case:

1. Amazon S3 Standard for frequent data access

This is suitable for performance sensitive use cases where the latency should be kept low. e.g. in a hospital, frequently accessed data will be the data of admitted patients, which should be retrieved quickly.

2. Amazon S3 Standard for infrequent data access

This is suitable for use cases where the data is long lived and less frequently accessed, i.e. for data archival but still expects high performance. e.g. in the same hospital, people who have been discharged, their records/data will not be needed on a daily basis, but if they return with any complication, their discharge summary should be retrieved quickly.

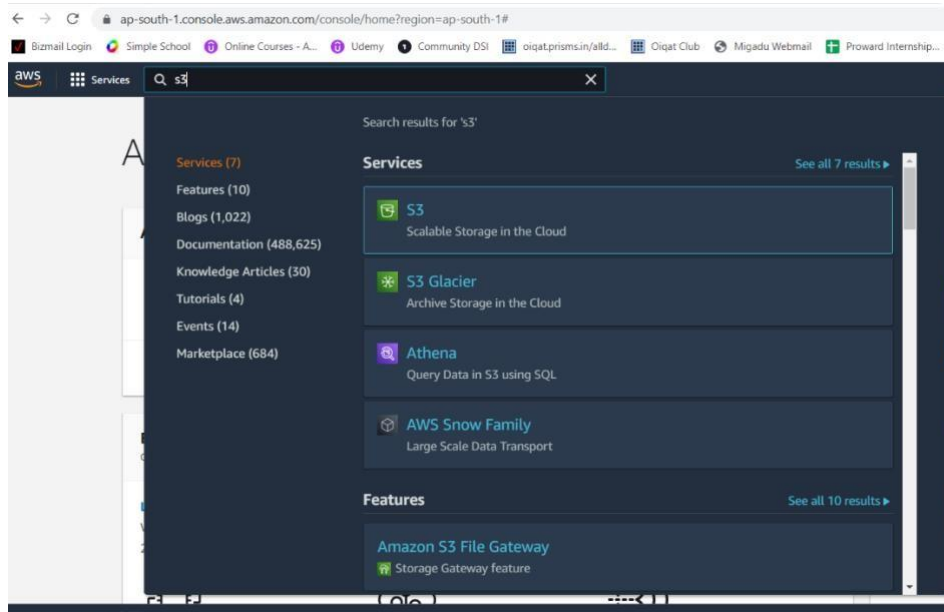
3. Amazon Glacier

Suitable for use cases where the data is to be archived and high performance is not required, it has a lower cost than the other two services. e.g. in the hospital, patients' test reports, prescriptions, MRI, X Ray, Scan docs etc. that are older than a year will not be needed in the daily run and even if it is required, lower latency is not needed.

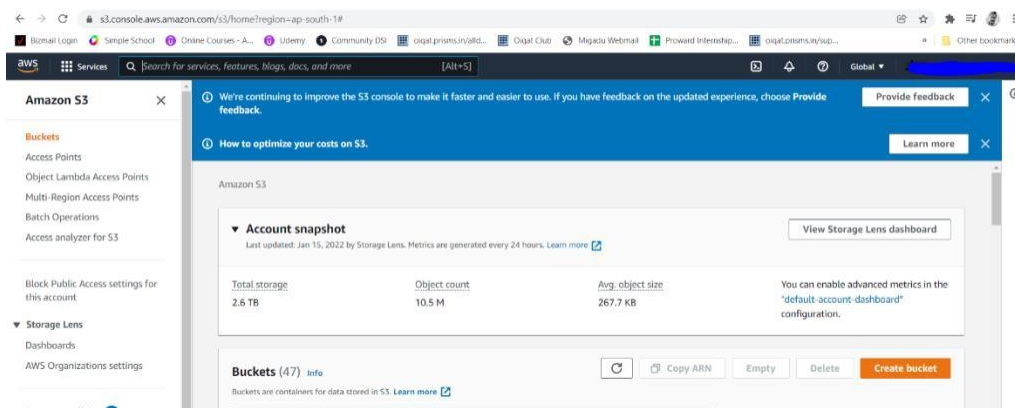
Characteristics	Standard	Standard - Infrequent Access	Glacier
Durability	99.99%	99.99%	99.99%
Availability	99.99%	99.90%	N/A
Minimum Object Size	No limit	128KB	No limit
Minimum Storage Duration	No minimum duration	30 Days	90 Days
First Byte Latency	milliseconds	milliseconds	4 hours
Retrieval Fee	No Fee	per GB retrieved	per GB retrieved

Output

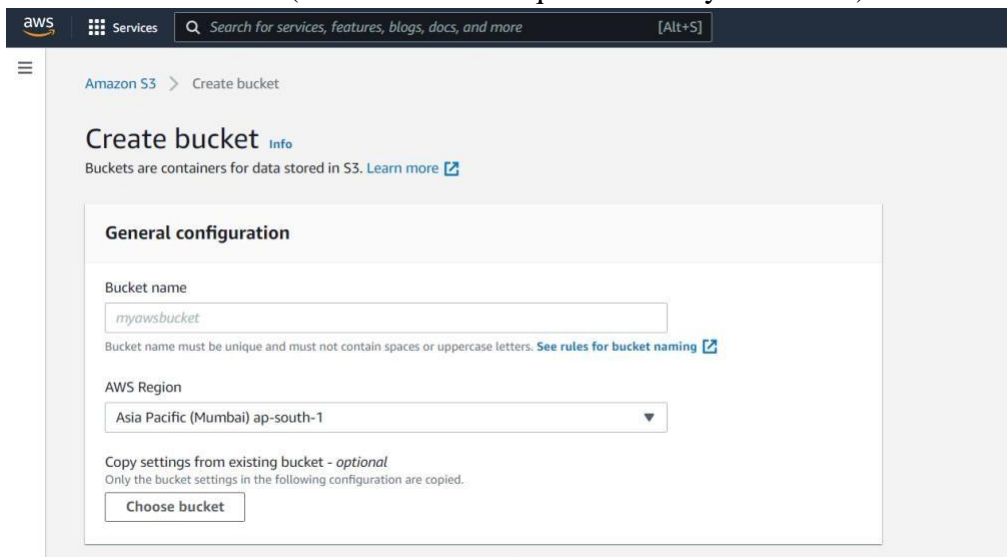
1. Login to AWS console and access S3 service



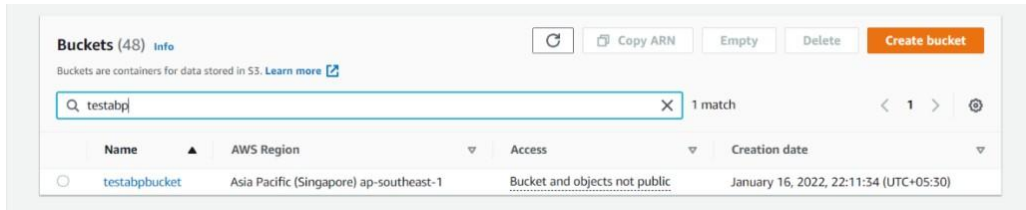
2. Create a bucket



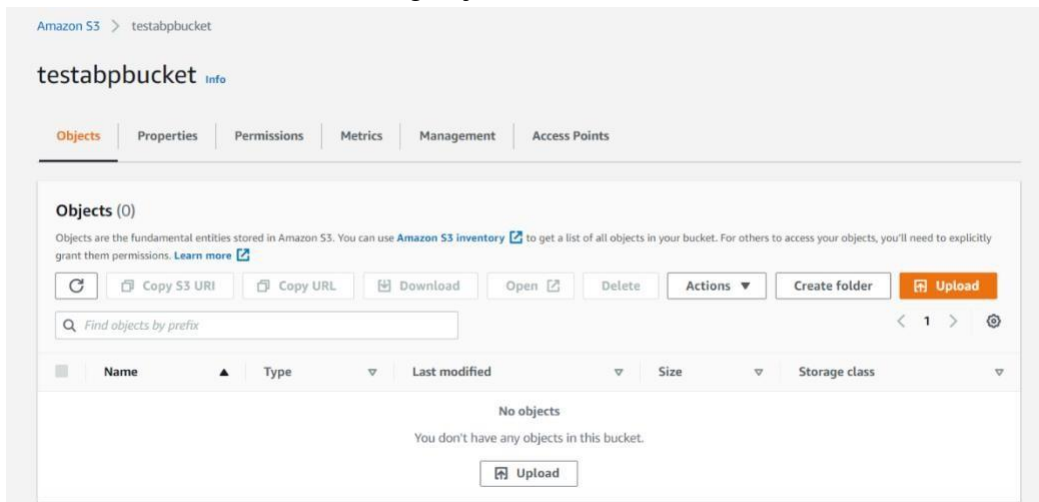
3. Add details of bucket (Note choose a unique name for your bucket)



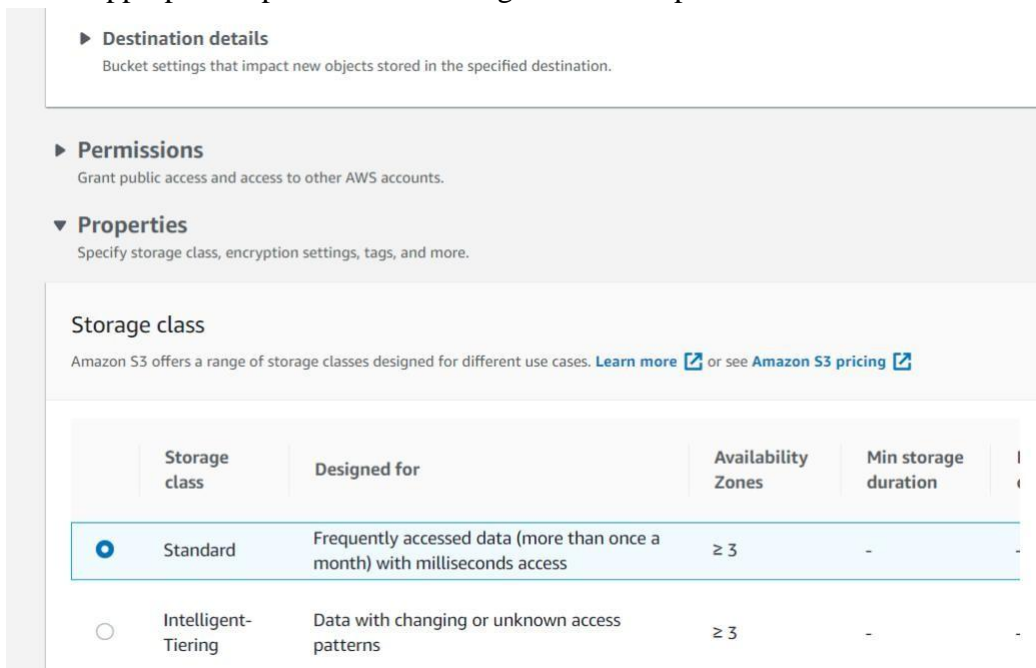
4. Search for bucket name



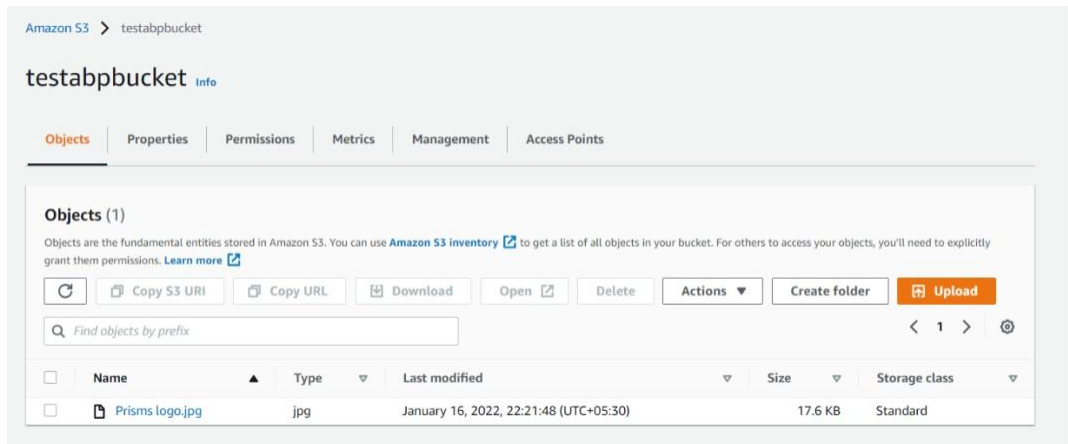
5. Click on bucket name for adding objects(items) inside buckets



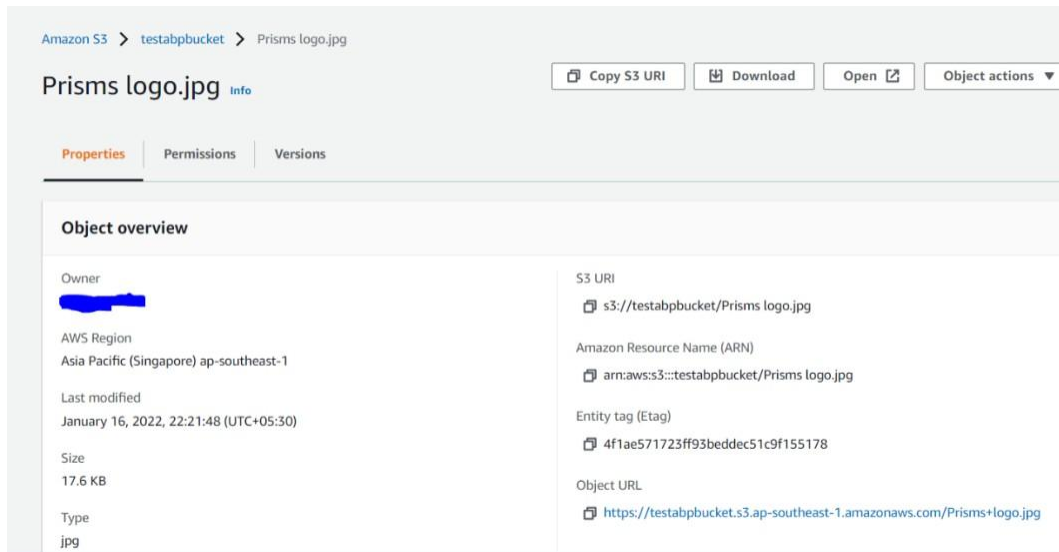
6. Select appropriate options while adding file. Same options can be selected later too.



7. Close window and go back to created bucket and open it. The object tab shows items in bucket



8. More details are shown by clicking on object name



9. Copy Object URL and paste in browser for accessing uploaded image
10. If error add necessary permissions to file.

**Conclusion:** S3 bucket is created successfully and objects are added to it. These objects can be accessed through Object URL. Also different types of S3 storage classes are observed.

## Experiment No: - 5

**Title:** Implementation of Virtualization in Cloud Computing to Learn Virtualization

Basics, Benefits of Virtualization in Cloud using Open-Source Operating System.

**Aim:** To study & Implementation of Virtualization in Cloud Computing to Learn Virtualization basics.

**Theory :**

What is Virtualization in Cloud Computing?

Virtualization is the "creation of a virtual (rather than actual) version of something, such as a server, a desktop, a storage device, an operating system or network resources". In other words, Virtualization is a technique, which allows sharing a single physical instance of a resource or an application among multiple customers and organizations. It does by assigning a logical name to a physical storage and providing a pointer to that physical resource when demanded.

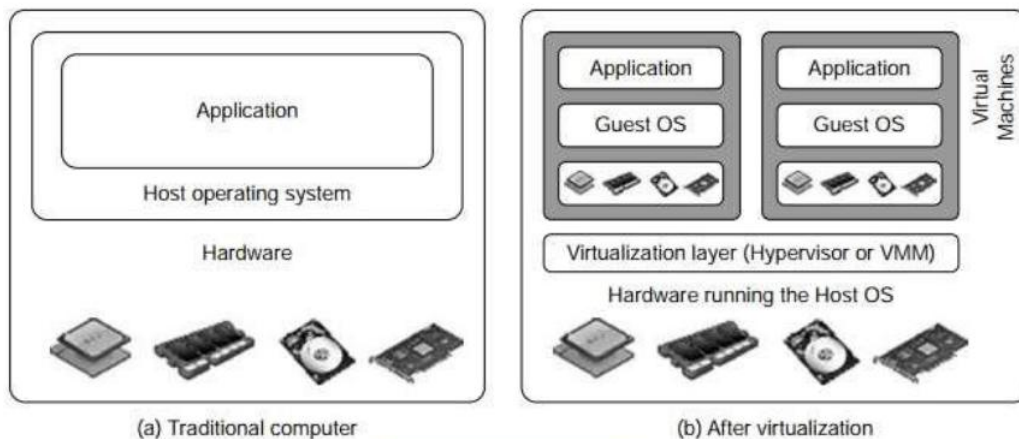


Fig: Traditional Computer Vs Virtualization

Types of Virtualization:

1. Hardware Virtualization.
2. Operating system Virtualization.
3. Server Virtualization.
4. Storage Virtualization.

**1) Hardware Virtualization:**

When the virtual machine software or virtual machine manager (VMM) is directly installed on the hardware system is known as hardware virtualization. The main job of hypervisor is to control and monitoring the processor, memory and other hardware resources. After virtualization of hardware system we can install different operating system on it and run different applications on those OS.

**Usage:** Hardware virtualization is mainly done for the server platforms, because controlling virtual machines is much easier than controlling a physical server.

## 2) Operating System Virtualization:

When the virtual machine software or virtual machine manager (VMM) is installed on the Host operating system instead of directly on the hardware system is known as operating system virtualization.

**Usage:** Operating System Virtualization is mainly used for testing the applications on different platforms of OS.

## 3) Server Virtualization:

When the virtual machine software or virtual machine manager (VMM) is directly installed on the Server system is known as server virtualization.

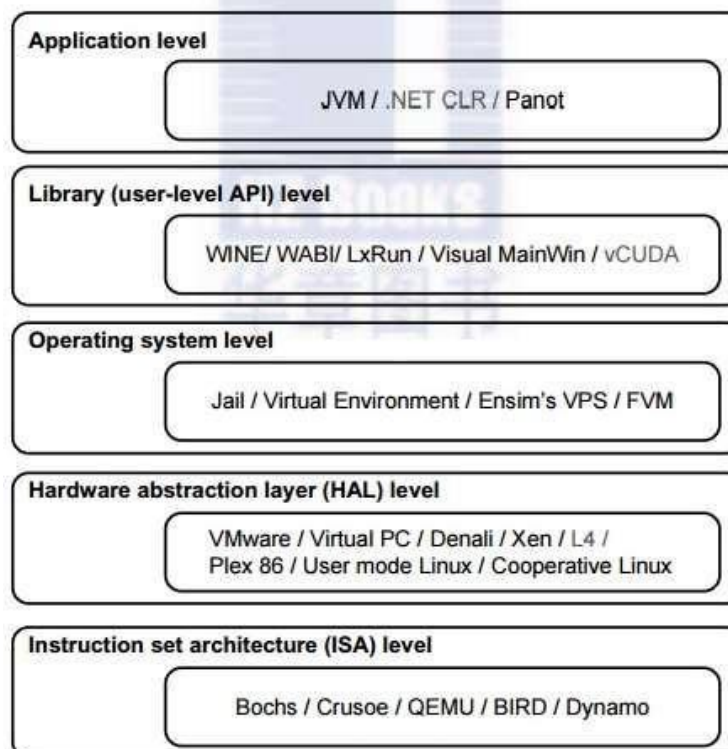
**Usage:** Server virtualization is done because a single physical server can be divided into multiple servers on the demand basis and for balancing the load.

## 4) Storage Virtualization:

Storage virtualization is the process of grouping the physical storage from multiple network storage devices so that it looks like a single storage device. Storage virtualization is also implemented by using software applications.

**Usage:** Storage virtualization is mainly done for back-up and recovery purposes.

## Levels of Virtualization:



## **1. Instruction Set Architecture Level**

At the ISA level, virtualization is performed by emulating a given ISA by the ISA of the host machine. For example, MIPS binary code can run on an x86-based host machine with the help of ISA emulation. With this approach, it is possible to run a large amount of legacy binary code writ-ten for various processors on any given new hardware host machine. Instruction set emulation leads to virtual ISAs created on any hardware machine.

## **2. Hardware Abstraction Level**

Hardware-level virtualization is performed right on top of the bare hardware. On the one hand, this approach generates a virtual hardware environment for a VM. On the other hand, the process manages the underlying hardware through virtualization. The idea is to virtualize a computer's resources, such as its processors, memory, and I/O devices. The intention is to upgrade the

hardware utilization rate by multiple users concurrently. The idea was implemented in the IBM VM/370 in the 1960s. More recently, the Xen hypervisor has been applied to virtualize x86-based machines to run Linux or other guest OS applications.

## **3. Operating System Level**

This refers to an abstraction layer between traditional OS and user applications. OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hard-ware and software in data centers. The containers behave like real servers. OS level virtualization is commonly used in creating virtual hosting environments to allocate hardware resources among a large number of mutually distrusting users.

## **4. Library Support Level**

Most applications use APIs exported by user-level libraries rather than using lengthy system calls by the OS. Since most systems provide well-documented APIs, such an interface becomes another candidate for virtualization. Virtualization with library interfaces is possible by controlling the communication link between applications and the rest of a system through API hooks. The software tool WINE has implemented this approach to support Windows applications on top of UNIX hosts. Another example is the vCUDA which allows applications executing within VMs to leverage GPU hardware acceleration.

## **5. User-Application Level**

Virtualization at the application level virtualizes an application as a VM. On a traditional OS, an application often runs as a process. Therefore, application-level virtualization is also known as process-level virtualization. The most popular approach is to deploy high level language (HLL).VMs. In this scenario, the virtualization layer sits as an application program on top of theoperating system, and the layer exports an abstraction of a VM that can run programs written andcompiled to a particular abstract machine definition. Any program written in the HLL andcompiled for this VM will be able to run on it. The Microsoft .NET CLR and Java VirtualMachine (JVM) are two good examples of this class of VM.

**Advantages of Virtualization:**

1. Resource optimization
2. Save resource and money
3. Enhance security
4. Easy disaster recovery

**Conclusion:** Thus, we have implemented Virtualization in Cloud Computing to Learn Virtualization.



## Experiment No: 6

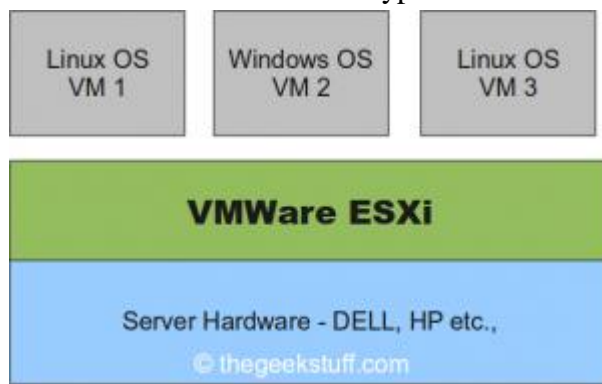
Title: Installing OS on a Virtual Machine Monitor.

### Theory:

#### VMware ESXi step-by-step Installation

VMware ESXi is free. However, the software comes with a 60 days evaluation mode. You should register on VMware website to get your free license key to come out of the evaluation mode. Once the ESXi is installed, you can either use vSphere Client or the Direct Console User Interface to administer the host.

VMware ESXi is based on hypervisor architecture that runs directly on top of hardware as shown below.



#### 1. Download ESXi server

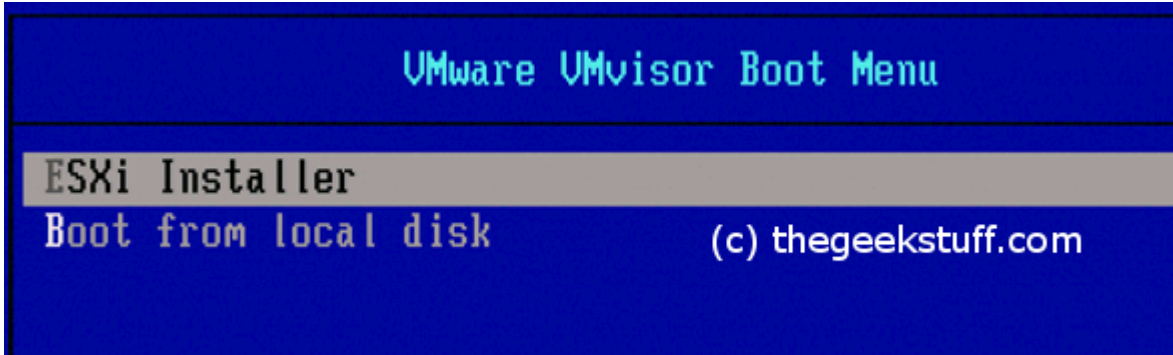
Get the software from the VMware ESXi download page.

Following are the various download options available. Select “ESXi 4.0 Update 1 Installable (CD ISO) Binary (.iso)” and burn a CD.

- ESXi 4.0 Update 1 Installable (CD ISO)
- Upgrade package from ESXi Server 3.5 to ESXi Server 4.0 Update 1
- Upgrade package from ESXi Server 4.0 to ESXi Server 4.0 Update 1
- VMware vSphere Client and Host Update Utility

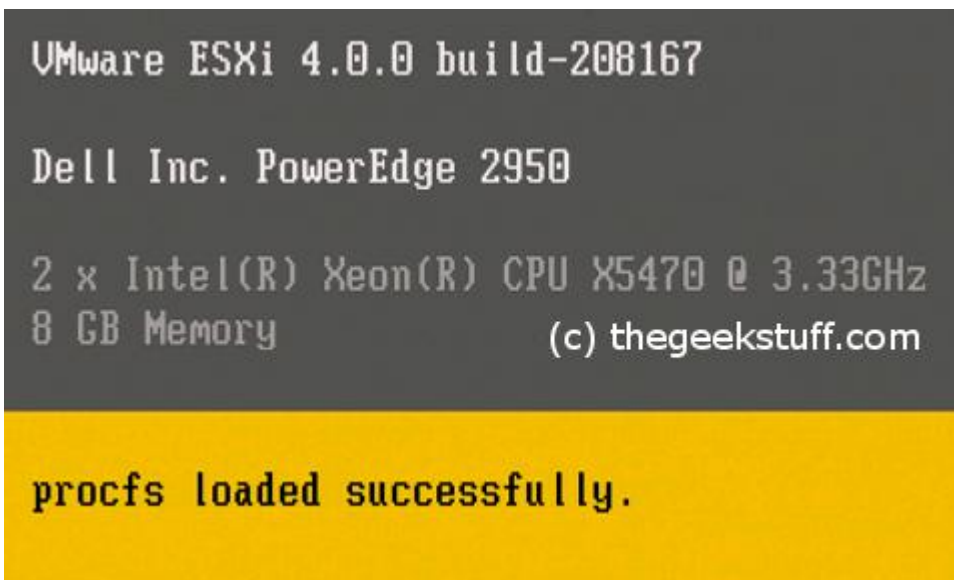
#### 2. VMware VMvisor Boot Menu

Once you insert the ESXi CD and reboot the server, it will display a boot menu with an option to launch “ESXi Installer” as shown below.



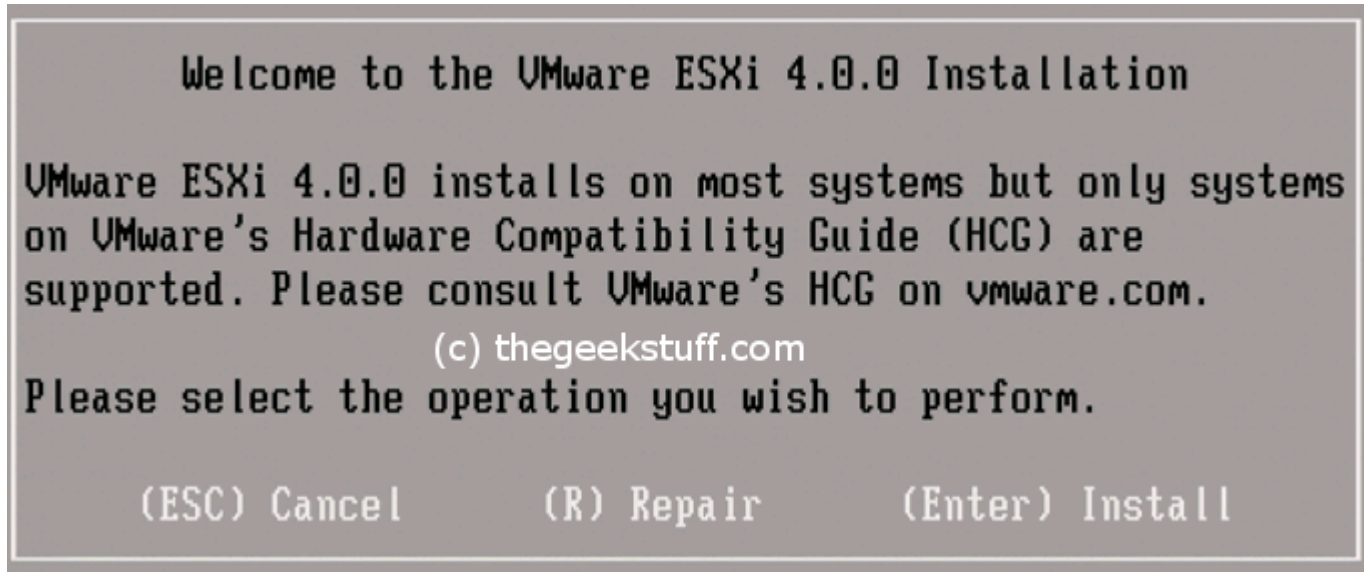
### 3. VMware ESXi Installer Loading

While the installer is loading all the necessary modules, it will display the server configuration information at the top as shown below. In this example, I was installing VMware ESXi 4.0 on a Dell PowerEdge 2950 server.



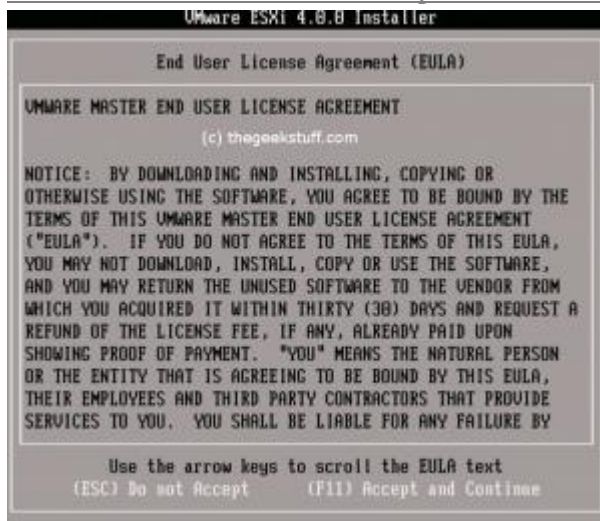
### 4. New ESXi Install

Since this is a new installation of ESXi, select "Install" in the following screen.



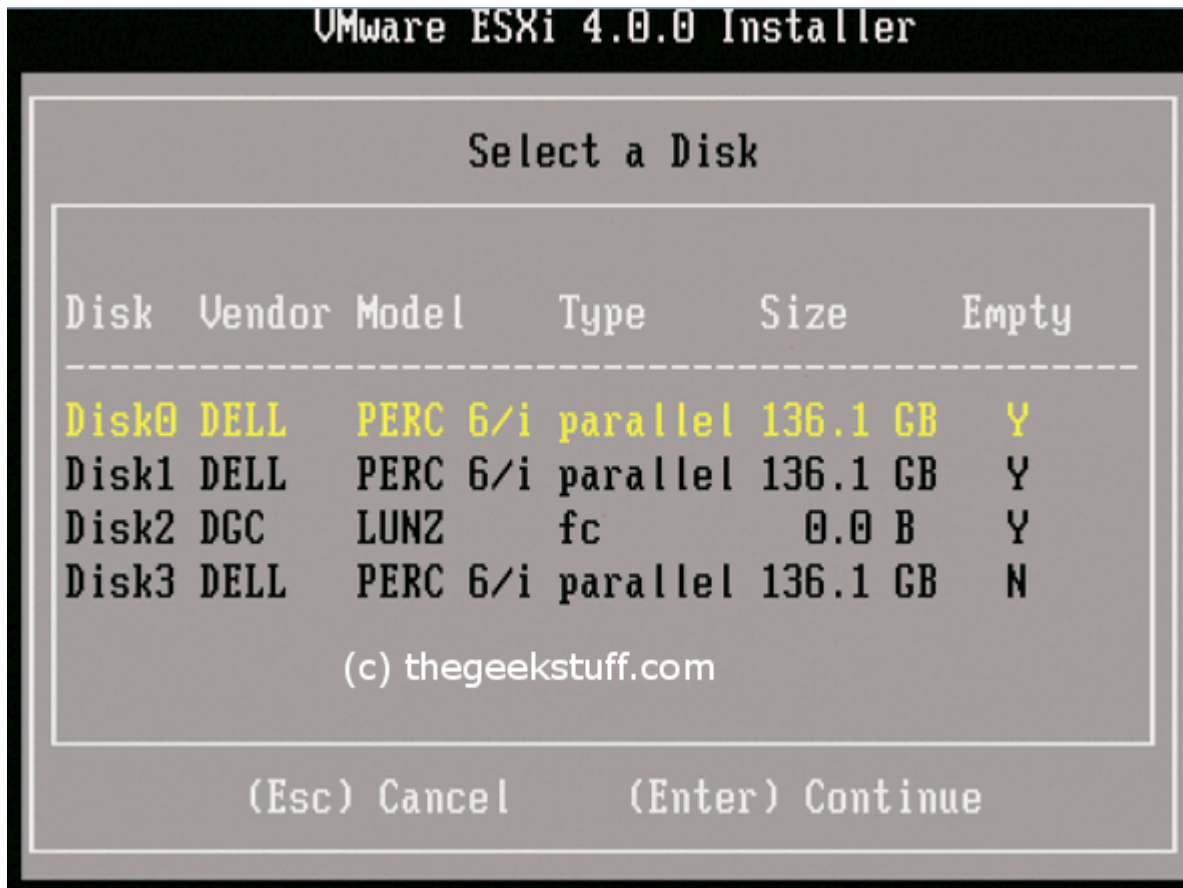
### 5. Accept VMware EULA

Read and accept the EULA by pressing F11.



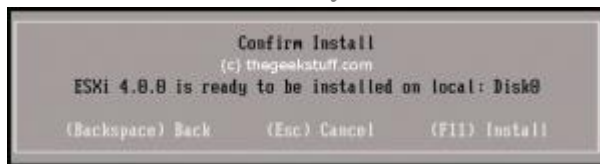
### 6. Select a Disk to Install VMware ESXi

VMware ESXi 4.0.0 Installer will display all available disk groups. Choose the Disk where you would like to install the ESXi. It is recommended to choose the Disk0.



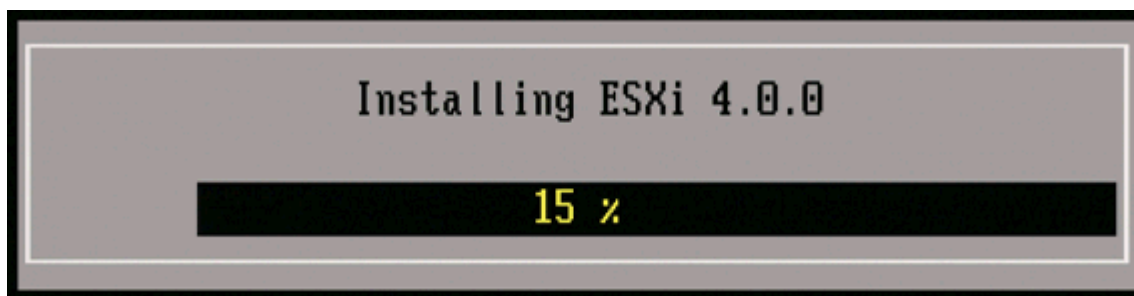
### 7. Confirm ESXi Installation

Confirm that you are ready to start the install process.



### 8. Installation in Progress

The installation process takes few minutes. While the ESXi is getting installed, it will display a progress bar as shown below.



### 9. ESXi Installation Complete

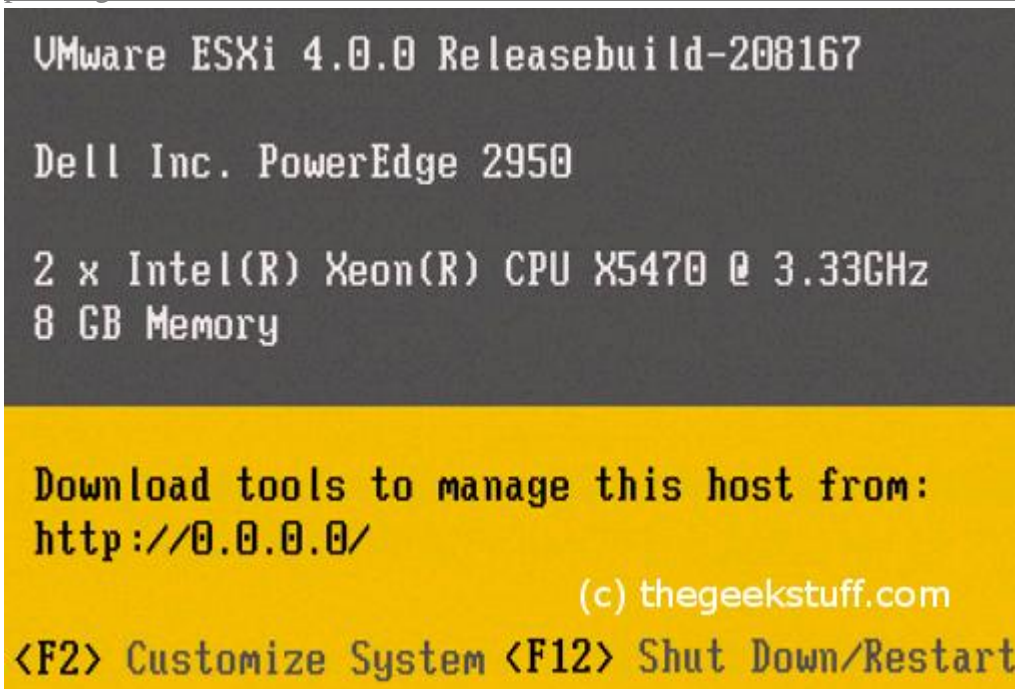
You will get the following installation completed message that will prompt you to reboot the server.



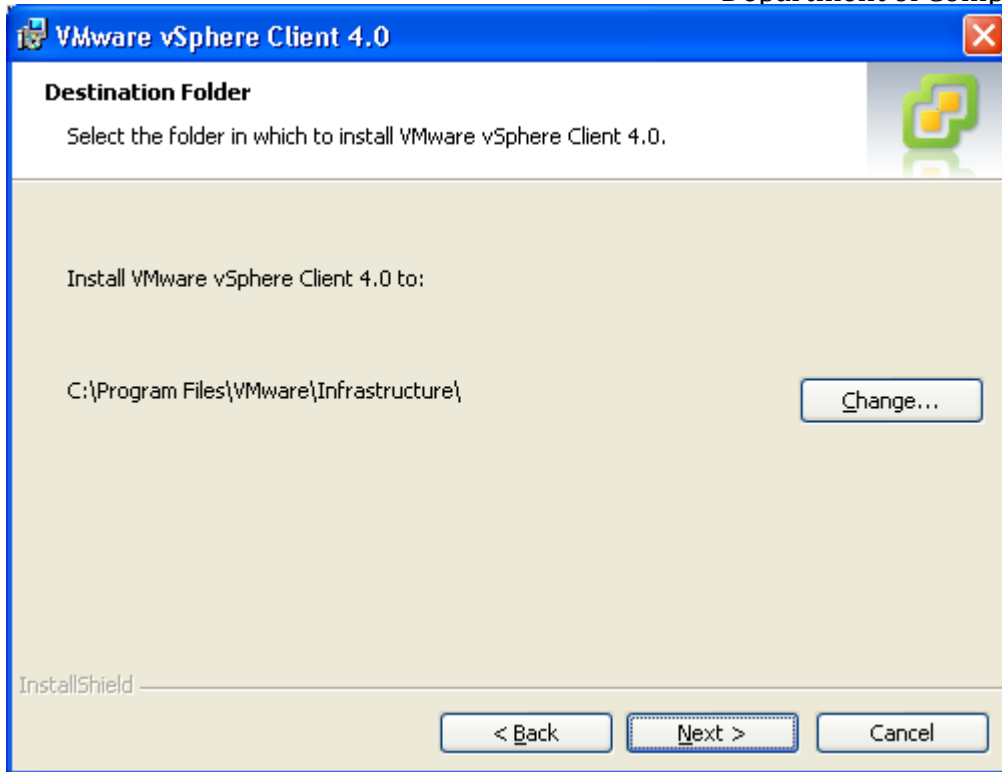
## 10. ESXi Initial Screen

After the ESXi is installed, you'll get the following screen where you can configure the system by pressing

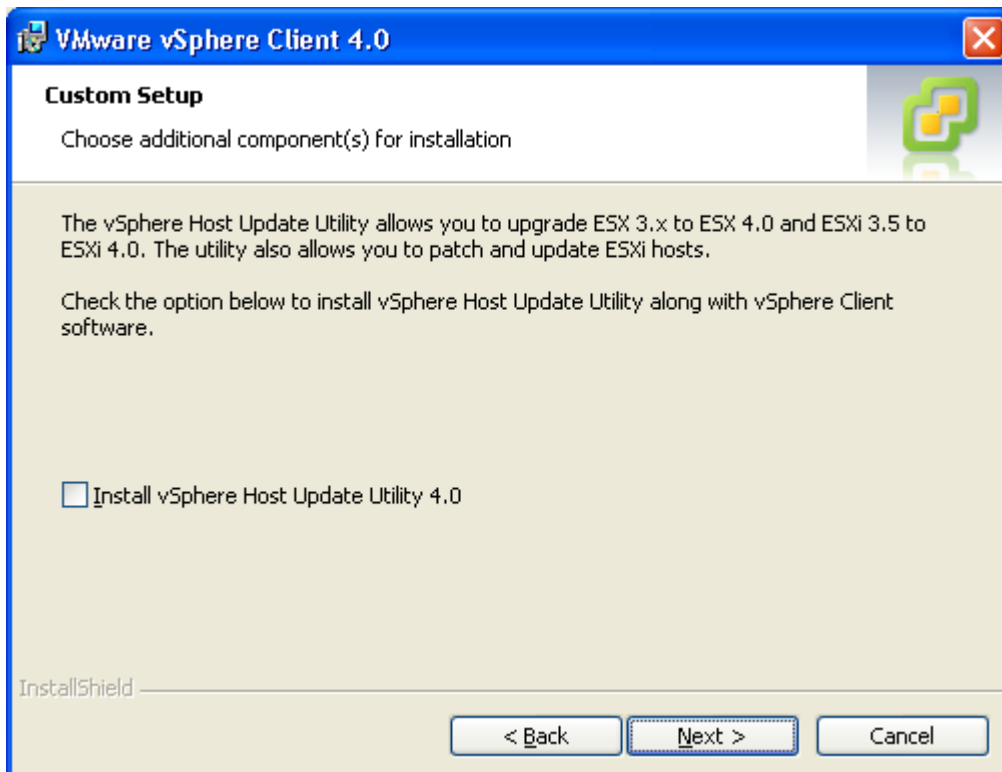
F2.



In the next article, let us review how to perform the initial ESXi configuration.



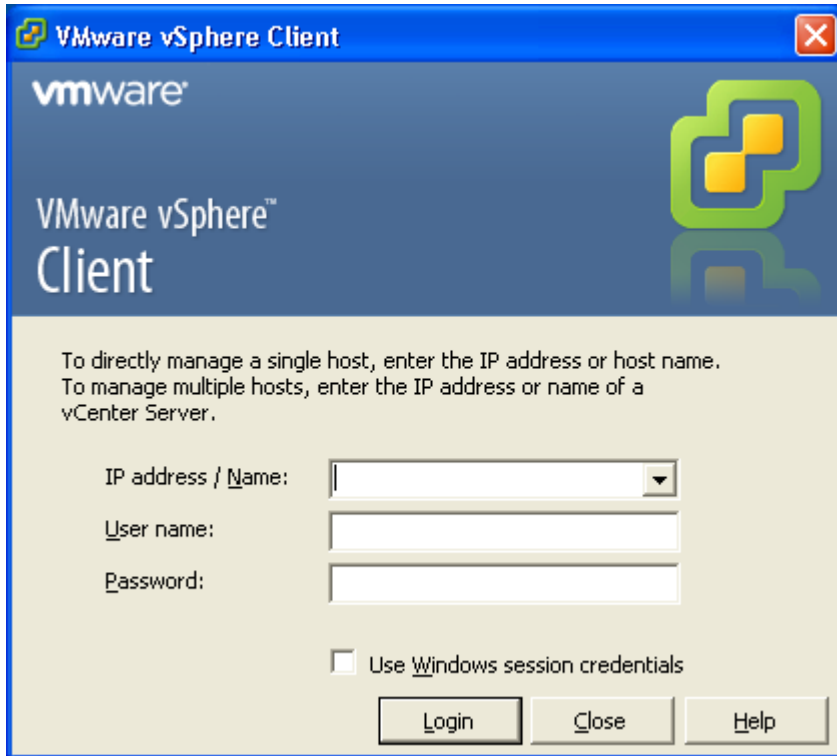
Since you are installing vSphere client to manage the newly installed ESXi 4.0 server, you don't need to select the 'Install vSphere Host Update Utility' check-box in the following screen.



### Login to vSphere Client

After the installation, launch the vSphere client utility from your Windows host and provide the following information:

- ip-address of the VMware ESXi server.
- User name: root
- Password: root password for the ESXi server.



### Ignore Security Warning

You might get a pop-up message about the security warning. Select the ‘Install this certificate and do not display any security warnings for {your-ip-address}’ check-box and click on ‘Ignore’.

VMware ESXi server is free. But, you should register at vmware website to get a license key. If you have not updated the ESXi server with the license key, you might also get the following “VMware Evaluation Notice” warning message.

### VMware vSphere Client

You are now successfully logged-in to the vSphere client as shown below. In the upcoming articles, we’ll explain how to create new VM using vSphere client and how to manage a ESXi server using this utility.

**Conclusion:** - Thus we have installed OS on Virtual Machine Monitor

## Experiment No: 7

Title: Offline/Online Migration of Virtual OS

### Theory:

#### Migrating Virtual Machines

Migration is the process of moving a virtual machine from one host or storage location to another. Copying a virtual machine creates a new virtual machine. It is not a form of migration.

In vCenter Server, you have the following migration options:

**Cold Migration** Moving a powered-off virtual machine to a new host. Optionally, you can relocate configuration and disk files to new storage locations. You can use cold migration to move virtual machines from one datacenter to another.

**Migrating a Suspended Virtual Machine** Moving a suspended virtual machine to a new host. Optionally, you can relocate configuration and disk files to new storage location. You can migrate suspended virtual machines from one datacenter to another.

**Migration with vMotion** Moving a powered-on virtual machine to a new host. Migration with vMotion allows you to move a virtual machine to a new host without any interruption in the availability of the virtual machine. You cannot use vMotion to move virtual machines from one datacenter to another. Some configurations include Metro vMotion, a feature that enables reliable migrations between hosts separated by high network round-trip latency times. Metro vMotion is automatically enabled when the appropriate license is installed. No user configuration is necessary.

**Migration with Storage vMotion** Moving the virtual disks or configuration file of a powered-on virtual machine to a new datastore. Migration with Storage vMotion allows you to move a virtual machine's storage without any interruption in the availability of the virtual machine.

Both migration of a suspended virtual machine and migration with vMotion are sometimes referred to as "hot migration", because they allow migration of a virtual machine without powering it off. Migration with vMotion is sometimes referred to as "live migration".

You can move virtual machines manually or set up a scheduled task to perform the cold migration.

Cloning a virtual machine or copying its disks and configuration file creates a new virtual machine. Cloning is not a form of migration.

#### Virtual Machine Configuration Requirements for vMotion



A number of specific virtual machine configurations can prevent migration of a virtual machine with vMotion.

The following virtual machine configurations can prevent migration with vMotion:

- You cannot use migration with vMotion to migrate virtual machines using raw disks for clustering purposes.
- You cannot use migration with vMotion to migrate a virtual machine that uses a virtual device backed by a device that is not accessible on the destination host. (For example, you cannot migrate a virtual machine with a CD drive backed by the physical CD drive on the source host.)  
Disconnect these devices before migrating the virtual machine. Virtual machines with USB passthrough devices can be migrated with vMotion as long as the devices are enabled for vMotion.
- You cannot use migration with vMotion to migrate a virtual machine that uses a virtual device backed by a device on the client computer. Disconnect these devices before migrating the virtual machine.

### **Host Configuration for vMotion**

In order to successfully use vMotion, you must first configure your hosts correctly.

Ensure that you have correctly configured your hosts in each of the following areas:

- Each host must be correctly licensed for vMotion.
- Each host must meet shared storage requirements for vMotion.
- Each host must meet the networking requirements for vMotion.

### **Important**

The ESXi firewall in ESXi 5.0 does not allow per-network filtering of vMotion traffic. Therefore, you must install rules on your external firewall to ensure that no incoming connections can be made to the vMotion socket.

### **Migrate a Powered-Off or Suspended Virtual Machine in the vSphere Web Client**

You can use the Migration wizard to migrate a powered-off virtual machine or suspended virtual machine.

### **Procedure**

- 1 Select a virtual machine.
  - In the virtual machines and templates inventory tree, select a group of virtual machines and select a virtual machine from the list on the right.

- Search for a virtual machine and select it from the search results list.
- 2 Right-click the virtual machine and select **Inventory > Migrate**.
  - 3 Select the migration type.

Option	Description
4	Select the destination resource pool for the virtual machine migration and click <b>Next</b> .
5	In the Host Name column, select the destination host or cluster for this virtual machine migration and click <b>Next</b> . Any compatibility problem appears in the Compatibility panel. Fix the problem, or select another host or cluster. Possible targets include hosts and DRS clusters with any level of automation. If a cluster has no DRS enabled, select a specific host in the cluster rather than selecting the cluster itself.
6	Select the datastore location where you want to store the virtual machine files.
7	If you chose to move the virtual machine's configuration file and virtual disks, select a disk format.
8	Review the information on the Review Selections page and click <b>Finish</b> .

- | Option | Description   |
|--------|---|
| 4      | Select the destination resource pool for the virtual machine migration and click <b>Next</b> .  |
| 5      | In the Host Name column, select the destination host or cluster for this virtual machine migration and click <b>Next</b> . Any compatibility problem appears in the Compatibility panel. Fix the problem, or select another host or cluster. Possible targets include hosts and DRS clusters with any level of automation. If a cluster has no DRS enabled, select a specific host in the cluster rather than selecting the cluster itself. |
| 6      | Select the datastore location where you want to store the virtual machine files.  |
| 7      | If you chose to move the virtual machine's configuration file and virtual disks, select a disk format.  |
| 8      | Review the information on the Review Selections page and click <b>Finish</b> .  |

vCenter Server moves the virtual machine to the new host. Event messages appear in the **Events** tab. The data displayed on the Summary tab shows the status and state throughout the migration. If errors occur during migration, the virtual machines revert to their original states and locations.

vCenter Server moves the virtual machine to the new host. Event messages appear in the **Events** tab. The data displayed on the Summary tab shows the status and state throughout the migration. If errors occur during migration, the virtual machines revert to their original states and locations.

### Migrate a Powered-On Virtual Machine with vMotion in the vSphere Web Client

You can use the Migration wizard to migrate a powered-on virtual machine from one host to another using vMotion technology. To relocate the disks of a powered-on virtual machine, migrate the virtual machine using Storage vMotion.

#### Prerequisites

Before migrating a virtual machine with vMotion, ensure that your hosts and virtual machines meet the requirements for migration with vMotion.

- [Host Configuration for vMotion](#)
- [Virtual Machine Configuration Requirements for vMotion](#)

#### Procedure

- 1 Select a virtual machine.
  - In the virtual machines and templates inventory tree, select a group of virtual machines and select a virtual machine from the list on the right.
  - Search for a virtual machine and select it from the search results list.
- 2 Right-click the virtual machine and select **Inventory > Migrate**.
- 3 Select **Change host** and click **Next**.
- 4 Select the destination resource pool for the virtual machine migration and click **Next**.
- 5 Select a destination host or cluster for the virtual machine.
  
- 6 Select the migration priority level and click **Next**.

**Reserve CPU for optimal VMotion performance**

---

**Perform with available CPU resources**

---

- 7 Review the page and click **Finish**.

**Conclusion:** - Thus we have performed Offline/Online Migration of Virtual OS.

## Experiment No: 8

Title: Deployment and Configuration options in Amazon (AWS).

Aim: To study Deployment and Configuration options in Amazon (AWS).

Theory:

Designing a deployment solution for application is a critical part of building a well-architected application on AWS. Based on the nature of your application and the underlying services (compute, storage, database, etc.) that it requires, we can use AWS services to create a flexible deployment solution that can be tailored to fit the needs of both application and our organization.

The constantly growing catalog of AWS services not only complicates the process of deciding which services will compose your application architecture, but also the process of deciding how you will create, manage, and update your application. When designing a deployment solution on AWS, you should consider how your solution will address the following capabilities:

- **Provision:** Create the raw infrastructure (Amazon EC2, Amazon Virtual Private Cloud [Amazon VPC], subnets, etc.) or managed service infrastructure (Amazon Simple Storage Service (Amazon S3), Amazon Relational Database Service [Amazon RDS], Amazon CloudFront, etc.) required for your application.
- **Configure:** Customize your infrastructure based on environment, runtime, security, availability, performance, network or other application requirements.
- **Deploy:** Install or update your application component(s) onto infrastructure resources, and manage the transition from a previous application version to a new application version.
- **Scale:** Proactively or reactively adjust the amount of resources available to your application based on a set of user-defined criteria.
- **Monitor:** Provide visibility into the resources that are launched as part of your application architecture. Track resources usage, deployment success/failure, application health, application logs, configuration drift, and more.

The task of designing a scalable, efficient, and cost-effective deployment solution should not be limited to the issue of how you will update your application version, but should also consider how you will manage supporting infrastructure throughout the complete application lifecycle. Resource provisioning, configuration management, application deployment, software updates, monitoring, access control, and other concerns are all important factors to consider when designing a deployment solution.

AWS provides a number of services that provide management capabilities for one or more aspects of your application lifecycle. Depending on your desired balance of control (i.e., manual management of resources) versus convenience (i.e., AWS management of resource

and the type of application, these services can be used on their own or combined to create a feature-rich deployment solution. This section will provide an overview of the AWS services that can be used to enable organizations to more rapidly and reliably build and deliver applications.

AWS provide following deployment services:

- AWS CloudFormation
- AWS Elastic Beanstalk
- AWS CodeDeploy
- Amazon Elastic Container Service
- Amazon Elastic Kubernetes Service
- AWS Ops-Works

AWS Elastic Beanstalk:

AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker on familiar servers such as Apache, Nginx, Passenger, and IIS. Elastic Beanstalk is a complete application management solution, and manages all infrastructure and platform tasks on your behalf.

With Elastic Beanstalk, you can quickly deploy, manage, and scale applications without the operational burden of managing infrastructure. Elastic Beanstalk reduces management complexity for web applications, making it a good choice for organizations that are new to AWS or wish to deploy a web application as quickly as possible.

When using Elastic Beanstalk as your deployment solution, simply upload your source code and Elastic Beanstalk will provision and operate all necessary infrastructure, including servers, databases, load balancers, networks, and auto scaling groups. Although these resources are created on your behalf, you retain full control of these resources, allowing developers to customize as needed.

Table 2: AWS Elastic Beanstalk Deployment Features:

Capability	Description
Provision	<p>Elastic Beanstalk will create all infrastructure components necessary to operate a web application or service that runs on one of its supported platforms. If you need additional infrastructure, this will have to be created outside of Elastic Beanstalk.</p> <p>Refer to Elastic Beanstalk Platforms for more details on the web application platforms supported by Elastic Beanstalk.</p>
Configure	<p>Elastic Beanstalk provides a wide range of options for customizing the resources in your environment.</p> <p>Refer to Configuring Elastic Beanstalk environments for more information about customizing the resources that are created by Elastic Beanstalk.</p>
Deploy	<p>Elastic Beanstalk automatically handles application deployments, and creates an environment that runs a new version of your application without impacting existing users.</p> <p>Refer to Deploying Applications to AWS Elastic Beanstalk for more details on application deployments with Elastic Beanstalk.</p>
Scale	<p>Elastic Beanstalk will automatically handle scaling of your infrastructure with managed auto scaling groups for your application instances.</p> <p>Refer to Auto Scaling Group for your Elastic Beanstalk Environment for more details about auto scaling with Elastic Beanstalk.</p>
Monitor	<p>Elastic Beanstalk offers built-in environment monitoring for applications including deployment success/failures, environment health, resource performance, and application logs.</p> <p>Refer to Monitoring an Environment for more details on full-stack monitoring with Elastic Beanstalk.</p>

Elastic Beanstalk makes it easy for web applications to be quickly deployed and managed in AWS. The following example shows a general use case for Elastic Beanstalk as it is used to deploy a simple web application.

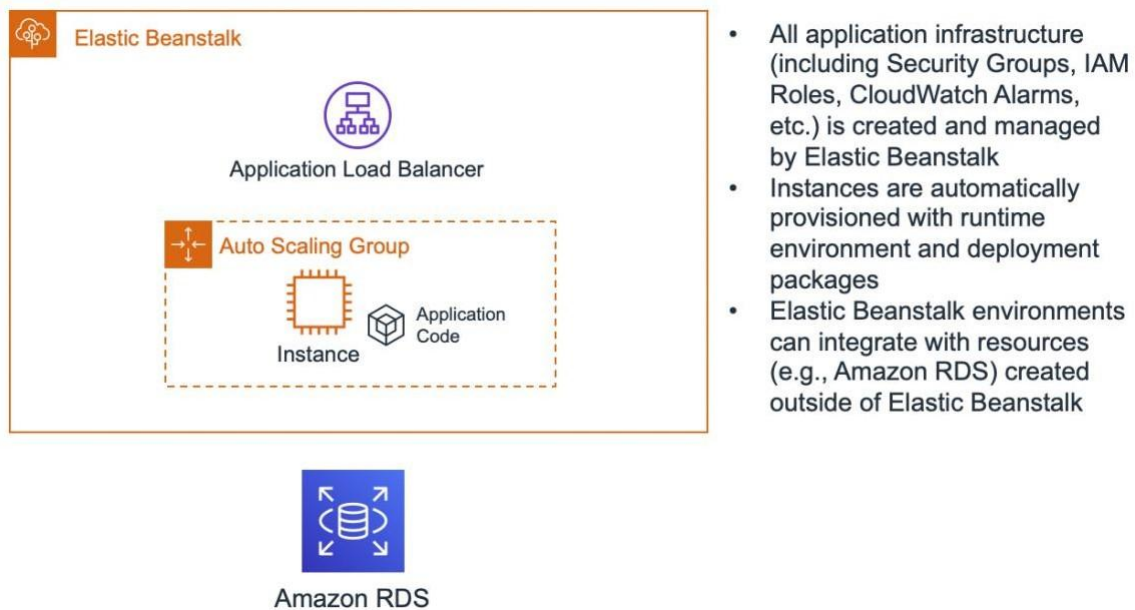


Figure: AWS Elastic Beanstalk use case

**Configuration options:**

Elastic Beanstalk defines a large number of configuration options that you can use to configure your environment's behavior and the resources that it contains. Configuration options are organized into namespaces like `aws:autoscaling:asg`, which defines options for an environment's Auto Scaling group.

The Elastic Beanstalk console and EB CLI set configuration options when you create an environment, including options that you set explicitly, and recommended values defined by the client. You can also set configuration options in saved configurations and configuration files. If the same option is set in multiple locations, the value used is determined by the order of precedence.

Configuration option settings can be composed in text format and saved prior to environment creation, applied during environment creation using any supported client, and added, modified or removed after environment creation. For a detailed breakdown of all of the available methods for working with configuration options at each of these three stages, read the following topics:

- Setting configuration options before environment creation
- Setting configuration options during environment creation
- Setting configuration options after environment creation

- **Settings applied directly to the environment** – Settings specified during a create environment or update environment operation on the Elastic Beanstalk API by any client, including the Elastic Beanstalk console, EB CLI, AWS CLI, and SDKs. The Elastic Beanstalk console and EB CLI also apply recommended values for some options that apply at this level unless overridden.
- **Saved Configurations** – Settings for any options that are not applied directly to the environment are loaded from a saved configuration, if specified.
- **Configuration Files (.ebextensions)** – Settings for any options that are not applied directly to the environment, and also not specified in a saved configuration, are loaded from configuration files in the .ebextensions folder at the root of the application source bundle.

Configuration files are executed in alphabetical order. For example, .ebextensions/01run.config is executed before .ebextensions/02do.config.

- **Default Values** – If a configuration option has a default value, it only applies when the option is not set at any of the above levels.

If the same configuration option is defined in more than one location, the setting with the highest precedence is applied. When a setting is applied from a saved configuration or settings applied directly to the environment, the setting is stored as part of the environment's configuration. These settings can be removed with the AWS CLI or with the EB CLI.

Settings in configuration files are not applied directly to the environment and cannot be removed without modifying the configuration files and deploying a new application version. If a setting applied with one of the other methods is removed, the same setting will be loaded from configuration files in the source bundle.

For example, say you set the minimum number of instances in your environment to 5 during environment creation, using either the Elastic Beanstalk console, a command line option, or a saved configuration. The source bundle for your application also includes a configuration file that sets the minimum number of instances to 2.

When you create the environment, Elastic Beanstalk sets the MinSize option in the aws:autoscaling:asg namespace to 5. If you then remove the option from the environment configuration, the value in the configuration file is loaded, and the minimum number of instances is set to 2. If you then remove the configuration file from the source bundle and redeploy, Elastic Beanstalk uses the default setting of 1.

## Recommended values

The Elastic Beanstalk Command Line Interface (EB CLI) and Elastic Beanstalk console provide recommended values for some configuration options. These values can be different from the default values and are set at the API level when your environment is created.



Recommended values allow Elastic Beanstalk to improve the default environment configuration without making backwards incompatible changes to the API.

For example, both the EB CLI and Elastic Beanstalk console set the configuration option for EC2 instance type (InstanceType in the aws:autoscaling:launchconfiguration namespace). Each client provides a different way of overriding the default setting. In the console you can choose a different instance type from a drop down menu on the **Configuration Details** page of the **Create New Environment** wizard. With the EB CLI, you can use the --instance\_type parameter for eb create.

Because the recommended values are set at the API level, they will override values for the same options that you set in configuration files or saved configurations. The following options are set:

#### Elastic Beanstalk console

- Namespace: aws:autoscaling:launchconfiguration  
Option Names: IamInstanceProfile, EC2KeyName, InstanceType
- Namespace: aws:autoscaling:updatepolicy:rollingupdate  
Option Names: RollingUpdateType and RollingUpdateEnabled
- Namespace: aws:elasticbeanstalk:application  
Option Name: Application Healthcheck URL
- Namespace: aws:elasticbeanstalk:command  
Option Name: DeploymentPolicy, BatchSize and BatchSizeType
- Namespace: aws:elasticbeanstalk:environment  
Option Name: ServiceRole
- Namespace: aws:elasticbeanstalk:healthreporting:system  
Option Name: SystemType and HealthCheckSuccessThreshold
- Namespace: aws:elasticbeanstalk:sns:topics  
Option Name: Notification Endpoint
- Namespace: aws:elasticbeanstalk:sqs  
Option Name: HttpConnections
- Namespace: aws:elb:loadbalancer  
Option Name: CrossZone
- Namespace: aws:elb:policies  
Option Names: ConnectionDrainingTimeout and ConnectionDrainingEnabled

## EB CLI

- Namespace:  
aws:autoscaling:launchconfiguration    Option  
Names: IamInstanceProfile, InstanceType
- Namespace: aws:autoscaling:updatepolicy:rollingupdate  
Option Names: RollingUpdateType and RollingUpdateEnabled
- Namespace: aws:elasticbeanstalk:command  
Option Name: BatchSize and BatchSizeType
- Namespace: aws:elasticbeanstalk:environment  
Option Name: ServiceRole
- Namespace: aws:elasticbeanstalk:healthreporting:system  
Option Name: SystemType
- Namespace: aws:elb:loadbalancer  
Option Name: CrossZone
- Namespace: aws:elb:policies  
Option Names: ConnectionDrainingEnabled

**Conclusion:** Thus, we have studied Deployment and Configuration options in Amazon(AWS).

**Experiment No: 9**

**Title:** Deployment and Configuration options in Microsoft Azure.

**Aim:** To study Deployment and Configuration options in Microsoft Azure.

**Theory:** You can use a few different technologies to deploy your Azure Functions project code to Azure. This article provides an overview of the deployment methods available to you and recommendations for the best method to use in various scenarios. It also provides an exhaustive list of and key details about the underlying deployment technologies.

Deployment methods:

The deployment technology you use to publish code to Azure is generally determined by the way in which you publish your app. The appropriate deployment method is determined by specific needs and the point in the development cycle. For example, during development and testing you may deploy directly from your development tool, such as Visual Studio Code. When your app is in production, you are more likely to publish continuously from source control or by using an automated publishing pipeline, which includes additional validation and testing.

The following table describes the available deployment methods for your Function project.

<b>DEPLOYMENT METHODS</b>		
<b>Deployment type</b>	<b>Methods</b>	<b>Best for...</b>
Tools-based	<ul style="list-style-type: none"> <li>• <a href="#">Visual Studio Code publish</a></li> <li>• <a href="#">Visual Studio publish</a></li> <li>• <a href="#">Core Tools publish</a></li> </ul>	Deployments during development and other ad-hock deployments. Deployments are managed locally by the tooling.
App Service-managed	<ul style="list-style-type: none"> <li>• <a href="#">Deployment Center (CI/CD)</a></li> <li>• <a href="#">Container deployments</a></li> </ul>	Continuous deployment (CI/CD) from source control or from a container registry. Deployments are managed by the App Service platform (Kudu).
External pipelines	<ul style="list-style-type: none"> <li>• <a href="#">Azure Pipelines</a></li> <li>• <a href="#">GitHub actions</a></li> </ul>	Production and DevOps pipelines that include additional validation, testing, and other actions be run as part of an automated deployment. Deployments are managed by the pipeline.

## Deployment technology availability

Azure Functions supports cross-platform local development and hosting on Windows and Linux. Currently, three hosting plans are available:

- Consumption
- Premium
- Dedicated (App Service)

Each plan has different behaviors. Not all deployment technologies are available for each flavor of Azure Functions.

## Key concepts

Some key concepts are critical to understanding how deployments work in Azure Functions.

### Trigger syncing

When you change any of your triggers, the Functions infrastructure must be aware of the changes. Synchronization happens automatically for many deployment technologies.

However, in some cases, you must manually sync your triggers. When you deploy your updates by referencing an external package URL, local Git, cloud sync, or FTP, you must manually sync your triggers. You can sync triggers in one of three ways:

- Restart your function app in the Azure portal
- Send an HTTP POST request to `https://{functionappname}.azurewebsites.net/admin/host/synctriggers?code=<API_KEY>` using the master key.
- Send an HTTP POST request to `https://management.azure.com/subscriptions/<SUBSCRIPTION_ID>/resourceGroups/<RESOURCE_GROUP_NAME>/providers/Microsoft.Web/sites/<FUNCTION_APP_NAME>/syncfunctiontriggers?api-version=2016-08-01`. Replace the placeholders with your subscription ID, resource group name, and the name of your function app.

### Remote build

Azure Functions can automatically perform builds on the code it receives after zip deployments. These builds behave slightly differently depending on whether your app is running on Windows or Linux. Remote builds are not performed when an app has previously been set to run in Run from Package mode. To learn how to use remote build, navigate to zip deploy.

### Note

If you're having issues with remote build, it might be because your app was created before the feature was made available (August 1, 2019). Try creating a new function app, or running `azfunctionapp update -g <RESOURCE_GROUP_NAME> -n <APP_NAME>` to update your function app. This command might take two tries to succeed.

All function apps running on Windows have a small management app, the SCM (or Kudu) site. This site handles much of the deployment and build logic for Azure Functions.

When an app is deployed to Windows, language-specific commands, like `dotnet restore` (C#) or `npm install` (JavaScript) are run.

### **Remote build on Linux**

To enable remote build on Linux, the following application settings must be set:

- `ENABLE_ORYX_BUILD=true`
- `SCM_DO_BUILD_DURING_DEPLOYMENT=true`

By default, both Azure Functions Core Tools and the Azure Functions Extension for Visual Studio Code perform remote builds when deploying to Linux. Because of this, both tools automatically create these settings for you in Azure.

When apps are built remotely on Linux, they run from the deployment package.

### Consumption plan

Linux function apps running in the Consumption plan don't have an SCM/Kudu site, which limits the deployment options. However, function apps on Linux running in the Consumption plan do support remote builds.

### Dedicated and Premium plans

Function apps running on Linux in the Dedicated (App Service) plan and the Premium plan also have a limited SCM/Kudu site.

## Deployment technology details

The following deployment methods are available in Azure Functions.

### External package URL

You can use an external package URL to reference a remote package (.zip) file that contains your function app. The file is downloaded from the provided URL, and the app runs in Run From Package mode.

How to use it: Add `WEBSITE_RUN_FROM_PACKAGE` to your application settings. The value of this setting should be a URL (the location of the specific package file you want to run). You can add settings either in the portal or by using the Azure CLI.

If you use Azure Blob storage, use a private container with a shared access signature (SAS) to give Functions access to the package. Any time the application restarts, it fetches a copy of the content. Your reference must be valid for the lifetime of the application.

When to use it: External package URL is the only supported deployment method for Azure Functions running on Linux in the Consumption plan, if the user doesn't want a remote build to occur. When you update the package file that a function app references, you must manually sync triggers to tell Azure that your application has changed.

## Zip deploy

Use zip deploy to push a .zip file that contains your function app to Azure. Optionally, you can set your app to start running from package, or specify that a remote build occurs.

How to use it: Deploy by using your favorite client tool: Visual Studio Code, Visual Studio, or from the command line using the Azure Functions Core Tools. By default, these tools use zip deployment and run from package. Core Tools and the Visual Studio Code extension both enable remote build when deploying to Linux. To manually deploy a .zip file to your function app, follow the instructions in [Deploy from a .zip file or URL](#).

When you deploy by using zip deploy, you can set your app to run from package. To run from package, set the WEBSITE\_RUN\_FROM\_PACKAGE application setting value to 1. We recommend zip deployment. It yields faster loading times for your applications, and it's the default for VS Code, Visual Studio, and the Azure CLI.

When to use it: Zip deploy is the recommended deployment technology for Azure Functions.

## Docker container

You can deploy a Linux container image that contains your function app.

How to use it: Create a Linux function app in the Premium or Dedicated plan and specify which container image to run from. You can do this in two ways:

Create a Linux function app on an Azure App Service plan in the Azure portal. For Publish, select Docker Image, and then configure the container. Enter the location where the image is hosted.

Create a Linux function app on an App Service plan by using the Azure CLI. To learn how, see [Create a function on Linux by using a custom image](#).

To deploy to an existing app by using a custom container, in Azure Functions Core Tools, use the func deploy command.

When to use it: Use the Docker container option when you need more control over the Linux environment where your function app runs. This deployment mechanism is available only for Functions running on Linux.

## Source control

Use source control to connect your function app to a Git repository. An update to code in that repository triggers deployment. For more information, see the [Kudu Wiki](#).

**How to use it:** Use Deployment Center in the Functions area of the portal to set up publishing from source control. For more information, see [Continuous deployment for Azure Functions](#).

**When to use it:** Using source control is the best practice for teams that collaborate on their function apps. Source control is a good deployment option that enables more sophisticated deployment pipelines.

You can use local Git to push code from your local machine to Azure Functions by using Git.

**How to use it:** Follow the instructions in Local Git deployment to Azure App Service.

**When to use it:** In general, we recommend that you use a different deployment method. When you publish from local Git, you must manually sync triggers.

## Cloud sync

Use cloud sync to sync your content from Dropbox and OneDrive to Azure Functions.

**How to use it:** Follow the instructions in Sync content from a cloud folder.

**When to use it:** In general, we recommend other deployment methods. When you publish by using cloud sync, you must manually sync triggers.

## FTP

You can use FTP to directly transfer files to Azure Functions.

**How to use it:** Follow the instructions in Deploy content by using FTP/s.

**When to use it:** In general, we recommend other deployment methods. When you publish by using FTP, you must manually sync triggers.

## Portal editing

In the portal-based editor, you can directly edit the files that are in your function app (essentially deploying every time you save your changes).

**How to use it:** To be able to edit your functions in the Azure portal, you must have created your functions in the portal. To preserve a single source of truth, using any other deployment method makes your function read-only and prevents continued portal editing. To return to a state in which you can edit your files in the Azure portal, you can manually turn the edit mode back to Read/Write and remove any deployment-related application settings (like WEBSITE\_RUN\_FROM\_PACKAGE).

**When to use it:** The portal is a good way to get started with Azure Functions. For more intense development work, we recommend that you use one of the following client tools:

- Visual Studio Code
- Azure Functions Core Tools (command line)
- Visual Studio

## Deployment behaviors

When you do a deployment, all existing executions are allowed to complete or time out, after which the new code is loaded to begin processing requests.

If you need more control over this transition, you should use deployment slots.

When you deploy your function app to Azure, you can deploy to a separate deployment slot instead of directly to production. For more information on deployment slots.

## Azure App Configuration

Azure App Configuration provides a service to centrally manage application settings and feature flags. Modern programs, especially programs running in a cloud, generally have many components that are distributed in nature. Spreading configuration settings across these components can lead to hard-to-troubleshoot errors during an application deployment. Use App Configuration to store all the settings for your application and secure their accesses in one place.

## Why use App Configuration?

Cloud-based applications often run on multiple virtual machines or containers in multiple regions and use multiple external services. Creating a robust and scalable application in a distributed environment presents a significant challenge.

Various programming methodologies help developers deal with the increasing complexity of building applications. For example, the Twelve-Factor App describes many well-tested architectural patterns and best practices for use with cloud applications. One key recommendation from this guide is to separate configuration from code. An application's configuration settings should be kept external to its executable and read in from its runtime environment or an external source.

While any application can make use of App Configuration, the following examples are the types of application that benefit from the use of it:

- Microservices based on Azure Kubernetes Service, Azure Service Fabric, or other containerized apps deployed in one or more geographies
- Serverless apps, which include Azure Functions or other event-driven stateless compute apps
- Continuous deployment pipeline

## App Configuration offers the following benefits:

- A fully managed service that can be set up in minutes
- Flexible key representations and mappings
- Tagging with labels
- Point-in-time replay of settings
- Dedicated UI for feature flag management
- Comparison of two sets of configurations on custom-defined dimensions
- Enhanced security through Azure-managed identities
- Encryption of sensitive information at rest and in transit
- Native integration with popular frameworks



App Configuration complements Azure Key Vault, which is used to store application secrets. App Configuration makes it easier to implement the following scenarios:

- Centralize management and distribution of hierarchical configuration data for different environments and geographies
- Dynamically change application settings without the need to redeploy or restart an application
- Control feature availability in real-time

### Use App Configuration

The easiest way to add an App Configuration store to your application is through a client library provided by Microsoft. The following methods are available to connect with your application, depending on your chosen language and framework

---

#### USE APP CONFIGURATION

---

Programming language and framework	How to connect
.NET Core and ASP.NET Core	App Configuration provider for .NET Core
.NET Framework and ASP.NET	App Configuration builder for .NET
Java Spring	App Configuration client for Spring Cloud
Others	App Configuration REST API

**Conclusion:** Thus, we have studied Deployment and Configuration options in Microsoft Azure.

### Experiment No: 10

Title: Assignment to install and configure Google App Engine

Theory:

Before begin

1. [Create a Google Cloud Platform project](#), if you don't have one already.
2. Make sure that Python 2.7 is installed on your

system: python -V

**Note:** As of Cloud SDK version 206.0.0, the gcloud CLI has experimental support for running using a Python 3.4+ interpreter (run gcloud topic startup for exclusions and more information on configuring your Python interpreter). All other Cloud SDK tools still require a Python 2.7 interpreter.

3. **Download the archive file best suited to your operating system.** Most machines will run the 64-bit package. If you'd like to check, run uname -m to verify if you're running a 64-bit system.

Platform	Package	Size	SHA256 Checksum
Linux			
64-bit	<a href="#">google-cloud-sdk-229.0.0-linux-x86_64.tar.gz</a>	25.6 MB	b1c87fc9451598a76cf66978dd8aa06482bfced639b56cf31559dc2c7f8b7b90
(x86_64)	<a href="#">x86_64.tar.gz</a>		
Linu			
x 32-bit	<a href="#">google-cloud-sdk-229.0.0-linux-x86.tar.gz</a>	25.2 MB	ee8c45f8018d0fee92b07c32cc6d8c891241da0b88bfe289d4e58e6746c3f668
(x86)	<a href="#">x86.tar.gz</a>		

**Alternatively, to download the Linux 64-bit archive file from your command-line, run:**

```
curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-sdk-229.0.0-linux-x86_64.tar.gz
```

For the 32-bit archive file, run:

```
curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-
```

cloud-sdk- 229.0.0-linux-x86.tar.gz

4. Extract the archive to any location on your file system; preferably, your Home folder.

On Linux, you can extract the archive file by running this command:

```
tar zxvf [ARCHIVE_FILE] google-cloud-sdk
```

5. If you're having trouble getting the gcloud command to work, ensure your \$PATH is defined appropriately. Use the install script to add Cloud SDK tools to your path. You will also be able to opt-in to command-completion for your bash shell and [usage statistics collection](#) during the installation process. Run the script using this command:

```
./google-cloud-sdk/install.sh
```

Restart your terminal for the changes to take effect.

Alternatively, you can call Cloud SDK after extracting the downloaded archive by invoking its executables via the full path.

## Initialize the SDK

Use the gcloud init command to perform several common SDK setup tasks. These include authorizing the SDK tools to access Google Cloud Platform using your user account credentials and setting up the default SDK configuration.

To initialize the SDK:

1. Run the following at a command

```
prompt: gcloud init
```

**Note:** To prevent the command from launching a web browser, use `gcloud init --console-only` instead. To authorize without a web browser and non-interactively, create a service account with the appropriate scopes using the [Google Cloud Platform Console](#) and use `gcloud auth activate-service-account` with the corresponding JSON key file.

2. Accept the option to log in using your Google user account:

```
To continue, you must log in. Would you like to log in (Y/n)? Y
```

3. In your browser, log in to your Google user account when prompted and click **Allow** to grant permission to access Google Cloud Platform resources.
4. At the command prompt, select a Cloud Platform project from the list of those where

Department of Computer Science and Engineering  
you have **Owner**, **Editor** or **Viewer** permissions:

Pick cloud project to use:

[1] [my-project-1]

[2] [my-project-2]

...

Please enter your numeric choice:

If you only have one project, gcloud init selects it for you.

5. If you have the Google Compute Engine API enabled, gcloud init allows you to choose a default Compute Engine zone:

Which compute zone would you like to use as project default?

[1] [asia-east1-a]

[2] [asia-east1-b]

...

[14] Do not use default zone

Please enter your numeric

choice:

gcloud init confirms that you have complete the setup steps successfully: gcloud has now been configured!

You can use [gcloud config] to change more gcloud settings.

Your active configuration is: [default]

Run core gcloud commands

Run these gcloud commands to view information about your SDK installation:

1. To list accounts whose credentials are stored on the local

system: gcloud auth list

gcloud displays a list of credentialed accounts:

Credentialed Accounts

ACTIVE ACCOUNT

[example-user-1@gmail.com](mailto:example-user-1@gmail.com)

example-user-2@gmail.com

2. To list the properties in your active SDK configuration:

`gcloud config list`

`gcloud` displays the list of properties:

[core]

account = example-user-

1@gmail.com

disable\_usage\_reporting = False

project = example-project

3. To view information about your Cloud SDK installation and the active SDK configuration: `gcloud info`

`gcloud` displays a summary of information about your Cloud SDK installation. This includes information about your system, the installed SDK components, the active user account and current project, and the properties in the active SDK configuration.

4. To view information about `gcloud` commands and other topics from the

command line: `gcloud help`

For example, to view the help for `gcloud compute instances create`:

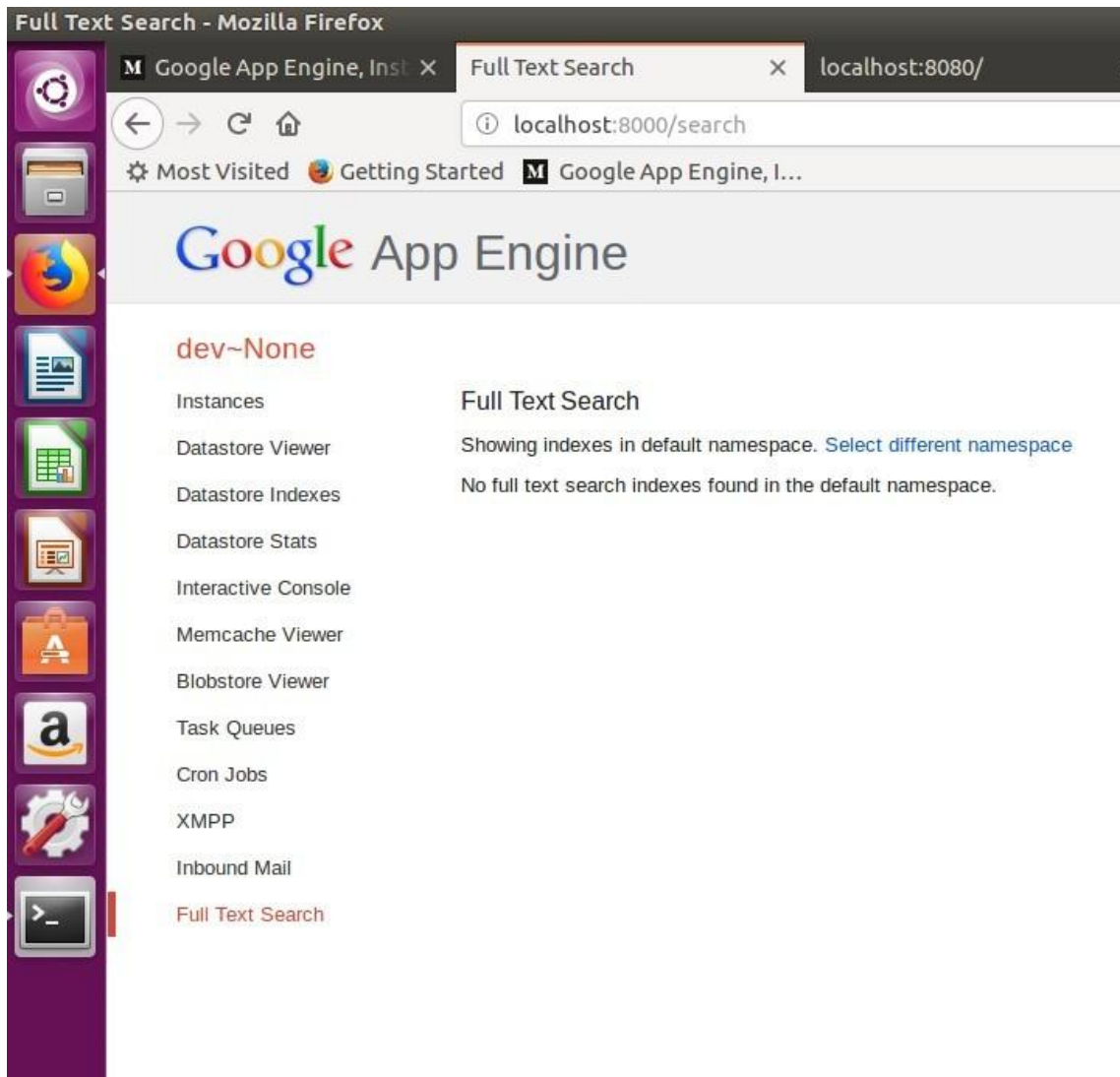
`gcloud help compute instances create`

`gcloud` displays a help topic that contains a description of the command, a list of commandflags and arguments, and examples of how to use it.

### How to Run Program:

Now as we have finished installing app engine, now it's time to create and upload an app. In this case we will be taking example of a "HELLO WORLD" app in python.

1. As we already have made sure that we have python installed in our system, It will be easier for us to clone existing code and deploy it rather than creating our own so we will use python docs- sample. Run the command "git clone <https://github.com/GoogleCloudPlatform/python-docs-samples>".
2. `cd python docs- samples/appengine/standard/hello_world`
3. `dev_appserver.py app.yaml`




**Conclusion:** Thus, we have installed and configure Google App Engine







	<p align="center"><b>G.K. Gujar Memorial Charitable Trust's, Dr. Ashok Gujar Technical Institute's Dr. Daulatrao Aher College of Engineering, Karad.</b> Vidyanagar Ext. Banawadi, Tal. Karad 415124, Dist. Satara, Maharashtra INDIA</p>
<p align="center"><b>Department of Computer Science &amp; Engineering</b></p>	

**Lab Manual  
Advanced Database Systems  
Academic Year: 2023-24**

Procedure for performing experiments.

- a) Explanation by the faculty using OHP/PPT covering the following aspects:-20 min
  - 1) Aim of the experiment.
  - 2) Software/Inputs required.
  - 3) Queries/Algorithm.
  - 4) Test Data/Tables.
- b) Writing of source program by the students 20 min.
- c) Compiling and execution of the program 60 min.
- d) Assessment 20 min.

<b>Laboratory Objectives:</b>	
Upon successful completion of this laboratory work, the student will be able to	
<b>C-I</b>	<b>Install relational database platform and familiar with basics of platform</b>
<b>C-II</b>	<b>Design and Implement schemas, concurrency control protocols and SQL Functions, Procedures, Cursors and Triggers using PL/SQL, views.</b>
<b>C-III</b>	<b>Install NoSQL database platform and familiar with basics of platform</b>
<b>C-IV</b>	<b>Implement of Advanced Data Mining Techniques</b>

**Format of First Page**

**Name :**

**Student Roll No # :**

**Experiment # :**

**Experiment Title :**

**Date of Experiment :**

**Date of Submission :**

### **Student Responsibilities**

1. In the very beginning of the laboratory work, the student has to do programming individually. For this reason, regular attendance is strictly required.
2. Every laboratory session is divided into two parts. In the first part, the instructor will be lecturing on the test objective, procedure and data collection. In the second part, the students, organized in groups (individual student), are required to do the programming. In order to perform the experiment within the assigned period, and to gain the maximum benefit from the experiment, the students must familiarize themselves with the purpose, objective, and procedure of the experiment before coming to the laboratory. Relevant lecture notes and laboratory manual should be studied carefully and thoroughly.
3. At the end of the experiment, every student should submit the written algorithm & programs & testes result for approval by the instructor.
4. It should be understood that laboratory facilities and equipment's (Hardware & Software) are provided to enhance the learning process.
5. The equipment's must be properly cared after every laboratory session. Also, students should always take precautions to avoid any possible hazards. Students must follow laboratory regulations provided at the end of this section.



**G. K. Gujar Memorial Charitable Trust's,  
Dr. Ashok Gujar Technical Institute's  
Dr. Daulatrao Aher College of Engineering, Karad.**

**Department of Computer Science & Engineering**

### **EXPERIMENT LIST**

**Department:** Computer Science and Engineering **Academic Year:** 2021-22 **Class:** Final Year

(B.Tech) **Semester:** VII **Subject:** Advanced Database Systems **Semester Duration:** 6 months

**Expt.**

**TITLE OF THE EXPERIMENT**

**No.**

**1** Installation and basic utilities of Oracle / MySQL and practicing DDL & DML commands **2**

Design and implement the Fragmentation schema & the Replication schema

**3** Implementation of 2 Phase Commit protocol for distributed databases.

**4** Demonstrate SQL Functions, Procedures, Cursors, and triggers using PL/SQL, Views. **5**

Installation of MongoDB and Apache Cassandra.

**6** Exploring MongoDB, and Apache Cassandra basics, Identify the schema design and data modeling techniques in MongoDB.

**7** Accessing MongoDB and Apache Cassandra from some of the high-level programming languages. Perform Create, Retrieve, Update and Delete or CRUD operations in MongoDB.

**8** Install CouchDB on Windows.

**9** Create and delete CouchDB database. Run CouchDB query with Mongo

**10** Case study of any Big Data system of your choice and design the distributed database architecture and analyze the probable solutions available in the market.

**11** Study of CASE concept and tools.

**12** Implementation of Oracle Synonyms and Sequence.

**13** Execute partitioning queries on parallel databases.

**14** Demonstrate all OLAP operations and cube operator in OLAP.

Case study of Oracle Database Administration and Security.

Study of database administrator's responsibilities like –

**15**

i) Installing and upgrading the database server and/or application tools.

ii) Creating user's profiles and ensuring system security by careful allocation of user permissions.

iii) Monitoring technical support for both database systems and related applications.

## **Experiment No.1**

**Title:** Installation and basic utilities of Oracle / MySQL and practicing DDL & DML commands

**Aim:** To Install & Demonstrate of DBMS like MySql, Oracle etc. and Reduce E-R model into tables  
(Practice session: Students should be allowed to choose appropriate DBMS software, install it, configure it

and start working on it. Create sample tables, execute some queries, use SQLPLUS features, use PL/SQL features like cursors on sample database. Students should be permitted to practice appropriate User interface creation tool and Report generation tool)

### **Theory:**

- Database: Collection of related data.
- Database Management System [DBMS]: A software system that enables users to define, create, maintain, and control access to the database.
- Database Application: A program that interacts with database at some point of its execution.
- Database System: A collection of application program that interacts with database along with DBMS itself.
- System Catalog/Data Dictionary/Meta-data: Its stores description of data present in database.
- Entity: Any real world distinct object.
- Attribute: It is a property of object that describes the object.
- Relationship: An association among entities.
- Entity Set: A set of all entities.
- Relationship Set: A set of relationships.
- SQL: A query language used to interact with database. It provides defining, manipulating and controlling access to data of database.
- DDL: A data definition language used to define the database. Creation, Deletion and Alter table queries provided by DDL.
- DML: A data manipulation language used to manipulate the database. It provides insert, delete, update and select queries.
- Relation: A table with columns and rows.
- Attribute: A named column in a relation.
- Domain: A set of possible values.
- Tuple: A row of relation
- Key: Any nonempty set of attributes
- Super key: Key which is uniquely identifies tuples from the database
- Candidate Key: A minimal Super Key.
- Primary Key: A candidate key chosen by database designer. It is unique and not null.
- Foreign Key: An attribute or set of attributes within one relation which is primary key of other relation.
- Relational Database: A collection of normalized relations with distinct relation names.

### **Implementation:**

**Case Study: To implement relational database of banking enterprise and Implement following DDL & DML queries.**



Introduction to SQL & basic queries.

## Basic SQL:

Each record has a unique identifier or primary key. SQL, which stands for Structured Query Language, is used to communicate with a database. Through SQL one can create and delete tables. Here are some commands:

- CREATE TABLE - creates a new database table
- ALTER TABLE - alters a database table
- DROP TABLE - deletes a database table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

SQL also has syntax to update, insert, and delete records.

- SELECT - get data from a database table
- UPDATE - change data in a database table
- DELETE - remove data from a database table
- INSERT INTO - insert new data in a database table

## 1. SELECT

The SELECT is used to query the database and retrieve selected data that match the specific criteria that you specify:

```
SELECT column1 [, column2, ...]  
FROM tablename  
WHERE condition
```

The conditional clause can include these operators

- = Equal
- > Greater than
- < Less than
- >= Greater than or equal

- <= Less than or equal
- <> Not equal to

- LIKE pattern matching operator

**SELECT \* FROM *tablename***

returns all the data from the table.

Use single quotes around text values (most database systems will also accept double quotes). Numerical values should not be enclosed in quotes.

LIKE matches a pattern. The wildcard % is used to denote 0 or more characters.

- 'A%' : matches all strings that start with *A*
- '%a' : matches all strings that end with *a*
- '%a%' : matches all strings that contain an *a*

## **2. CREATE TABLE**

The CREATE TABLE statement is used to create a new table. The format is:

```
CREATE TABLE tablename  
(column1 data type,  
column2 data type,  
column3 data type);
```

- char(size): Fixed length character string.
- varchar(size): Variable-length character string. Max size is specified in parenthesis. •
- number(size): Number value with a max number of columns specified in parenthesis •
- date: Date value
- number(size,d): A number with a maximum number of digits of "size" and a maximum number of "d" digits to the right of the decimal

## **3. INSERT VALUES**

Once a table has been created data can be inserted using INSERT INTO command.

```
INSERT INTO tablename  
(col1, ... , coln)  
VALUES (val1, ... , valn)
```

## **4. UPDATE**

To change the data values in a pre existing table, the UPDATE command can be used.

UPDATE *tablename*  
SET *colX = valX* [, *colY = valY*, ...]  
WHERE *condition*

## 5. DELETE

The DELETE command can be used to remove a record(s) from a table.

DELETE FROM *tablename*  
WHERE *condition*

To delete all the records from a table without deleting the table do

DELETE \* FROM *tablename*

## 6. DROP

To remove an entire table from the database use the DROP command.

DROP TABLE *tablename*

## 7. ORDER BY

ORDER BY clause can order column name in either ascending (ASC) or descending (DESC) order.

ORDER BY *col\_name* ASC

## 8. AND / OR

AND and OR can join two or more conditions in a WHERE clause. AND will return data when all the conditions are true. OR will return data when any one of the conditions is true.

## 9. IN

IN operator is used when you know the exact value you want to return for at least one of the columns

SELECT \* FROM *table\_name* WHERE *col\_name* IN (*val1*, *val2*, ...)

## 10. BETWEEN / AND

The BETWEEN ... AND operator selects a range of data between two values. These values can be numbers, text, or dates.

SELECT \* FROM *table\_name* WHERE *col\_name* BETWEEN *val1* AND *val2*

**Installation Guidelines:****Steps for Installation:**

Welcome to Oracle Database 10g Express Edition (Oracle Database XE)! This tutorial gets you quickly up and running using Oracle Database XE by creating a simple application. This guide covers the following topics:

- Logging in as the Database Administrator
- Unlocking the Sample User Account
- Logging in as the Sample User Account
- Creating a Simple Application
- Running Your New Application
- Using the Oracle Database XE Menus

**1. Logging in as the Database Administrator:**

The first thing you need to do is to log in as the Oracle Database XE Administrator. Follow these steps:

1. Open the Database Home Page login window:

On Windows, from the Start menu, select **Programs**(or All Programs), then Oracle Database 10g Express Edition, and then **Go To Database Home Page**.

■ On Linux, click the Application menu (on Gnome) or the K menu (on KDE), then point to Oracle Database 10g Express Edition, and then Go To Database Home Page. 2. At the Database Home Page login window, enter the following information:

- Username: Enter system for the user name.
- Password: Enter the password that was specified when Oracle Database XE was installed.

3. Click Login

The Oracle Database XE home page appears.



## 2. Unlocking the Sample User Account

To create your application, you need to log in as a database user. Oracle Database XE comes with a sample database user called HR. This user owns a number of database tables in a sample schema that can be used to create applications for a fictional Human Resources department. However, for security reasons, this user's account is locked. You need to unlock this account before you can build a sample application.

**To unlock the sample user account:**

1. Make sure you are still logged on as the database administrator, as described in the previous section.
2. Click the Administration icon, and then click Database Users.
3. Click the HR schema icon to display the user information for HR.



AGTI's, DACOE Karad Page 10

Advanced Database Systems Lab Manual

- Under Manage Database User, enter the following settings:
  - Password and Confirm Password: Enter hr for the password.
  - Account Status: Select Unlocked.
  - Roles: Ensure that both CONNECT and RESOURCE are enabled.
4. Click Alter User.

Now you are ready to create your first application.

### 3. Logging in as the Sample User Account

To log in as the sample user account:

1. Log out from the database administrator account by clicking Logout in the upper right corner of the Database Home Page.
2. In the window, click Login.
3. In the Login window, enter hr for both the user name and password.
4. Click Login.

The Database Home Page appears.

### 4. Creating a Simple Application

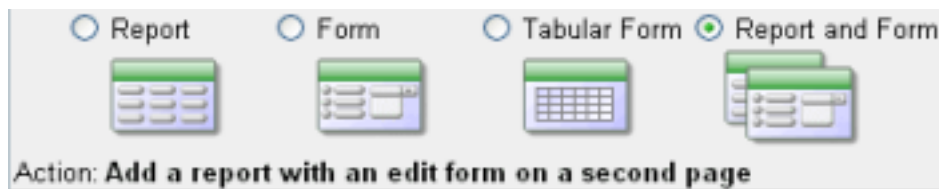
Creating an application is an easy way to view and edit your database data. You create this application based on the EMPLOYEES table, which is part of the HR schema. To create an application based on the EMPLOYEES table:

1. On the Database Home Page, click the Application Builder icon.
2. Click the Create button.

3. Under Create Application, select Create Application and click Next.
4. Under Create Application:
  - a. Name: Enter MyApp.
  - b. Accept the remaining defaults.
  - c. Click Next.

**Next, add pages to your application.**

5. Under Add Page:
  - a. For Select Page Type, select Report and Form.



Notice that Action describes the type of page you are adding.

- b. Next to the Table Name field, click the up arrow, and then select EMPLOYEES from the Search Dialog window.
- c. Click Add Page.

Two new pages display at the top of the page, under Create Application.

Create Application					Cancel	< Previous	Next >
Page	Page Name	Page Type	Source Type	Source			
1	EMPLOYEES	Report	Table	EMPLOYEES			✘
2	EMPLOYEES	Form	Table	EMPLOYEES			✘

d. Click Next

6. On the Tabs panel, accept the default (One Level of Tabs) and click Next. 7. On the Shared Components panel, accept the default (No) and click Next. This option enables you to import shared components from another application. Shared components are common elements that can display or be applied on any page within an application.

8. For Authentication Scheme, Language, and User Language Preference Derived From, accept the defaults and click Next.

9. For the theme, select Theme 2 click Next. Themes are collections of templates that you can use to define the layout and style of an entire application.

10. Confirm your selections. To return to a previous wizard page, click Previous.

To accept your selections, click Create.

After you click Create, the following message displays at the top of the page:

AGTI's, DACOE Karad Page 12

Advanced Database Systems Lab Manual

**Application created successfully.**

### **5. Running Your New Application:**

To run your application:

1. Click the Run Application icon.



In the log in page, enter hr for both the User Name and Password.

Your application appears, showing the EMPLOYEES table.

3. Explore your application.

You can query the EMPLOYEES table, if you want. To manage the application, use the Developer toolbar at the bottom on the page.

The Developer toolbar offers a quick way to edit the current page, create a new page, control, or component, view session state, or toggle debugging or edit links on and off. 4. To exit your application and return to Application Builder, click Edit Page 1 on the Developer toolbar.

5. To return to the Database Home Page, select the Home breadcrumb at the top of the page.

Congratulations! You have just created your first application using Oracle Database XE.

### **6. Using the Oracle Database XE Menus:**

You can use the Oracle Database XE menus to perform basic functions with Oracle Database XE. To see the menus, do the following:



- On Windows, from the Start menu, select Programs (or All Programs) and then Oracle Database 10g Express Edition.

- On Linux, click the Application menu (on Gnome) or the K menu (on KDE) and then point to Oracle Database 10g Express Edition.

The following menu items are available:

- Get Help: Displays the following selections:

- Go To Online Forum: Displays the online forum for discussions about Oracle Database XE.
- Read Documentation: Displays the Oracle Database XE documentation library on the Internet.
- Read Online Help: Displays the Oracle Database XE online help. This help is only available if the database is started.

AGTI's, DACOE Karad Page 13

**Advanced Database Systems Lab Manual**

- Register For Online Forum: Allows you to register for the Oracle Database XE online forum.

- Backup Database: In NOARCHIVELOG mode (the default), shuts down the database, backs it up, and then restarts it. In ARCHIVELOG mode, performs an online backup of the database. For more information on backups, refer to Oracle Database Express Edition 2 Day DBA.
- Get Started: Link to this tutorial.

- Go To Database Home Page: Displays the Oracle Database XE Home Page in your default browser. "Logging in as the Database Administrator" on page 1 explains how to log into this home page as a database administrator.

- Restore Database: Shuts down and then restores the database to the most recent backup. For more information on restoring a database, refer to Oracle Database Express Edition 2 Day DBA.

- Run SQL Command Line: Starts the SQL Command Line utility for Oracle Database XE. To connect to the database, issue the following command at the **SQL prompt that appears**:

### **Connect username/password**

where username is the user name, such as sys, system, or another account name, and password is the password that was assigned when Oracle Database XE was installed. The get help, you can enter the command help at the SQL prompt, once you have connected to the database.

- Start Database: Starts Oracle Database XE. By default, the database is started for you after installation and every time your computer is restarted. However, if you think the database is not running you can use this menu item to start it.

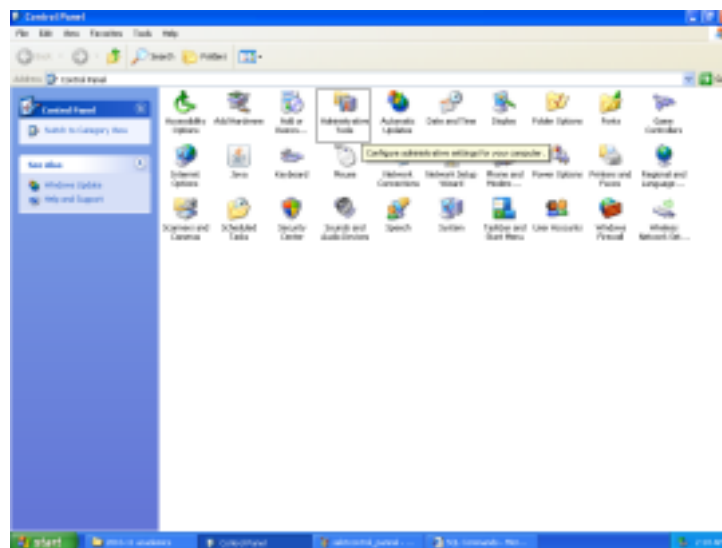
Stop Database: Stops Oracle Database XE.

## How to create ODBC connectivity

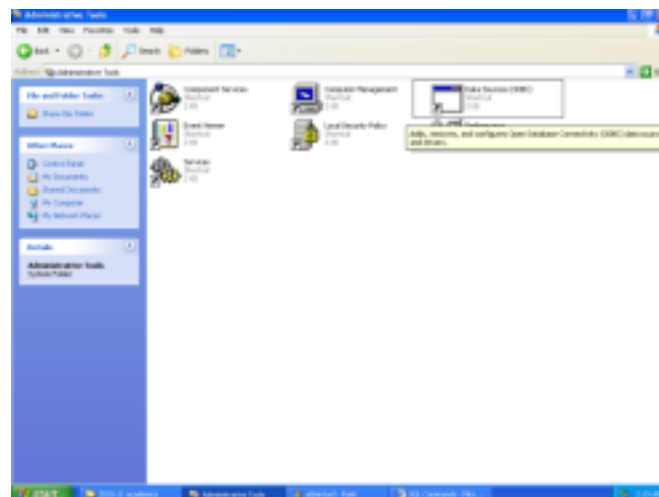
**STEP 1:** go to control panel and open Administrative tools

AGTI's, DACOE Karad Page 14

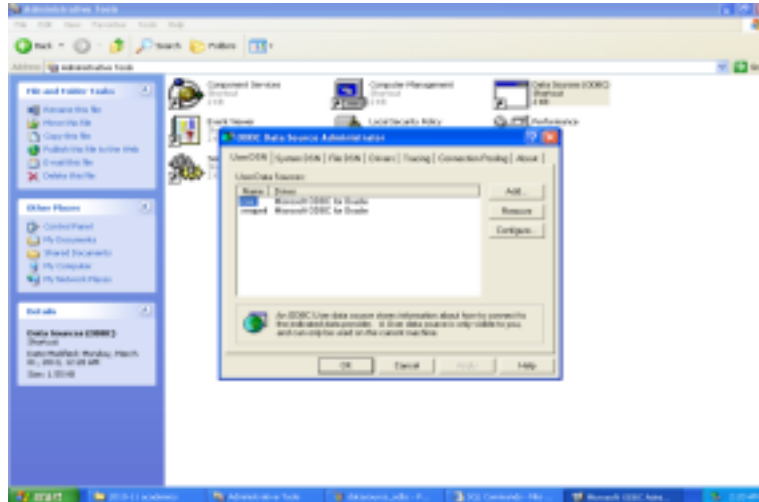
Advanced Database Systems Lab Manual



**Step 2:** select data source (ODBC) and open



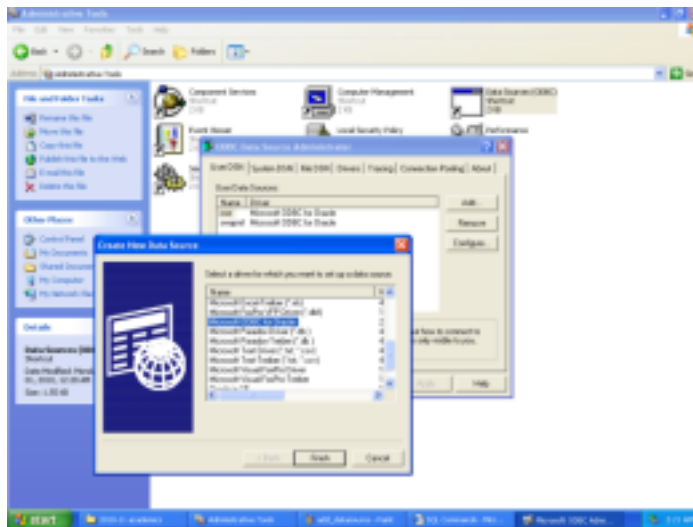
**Step 3:** Click Add...



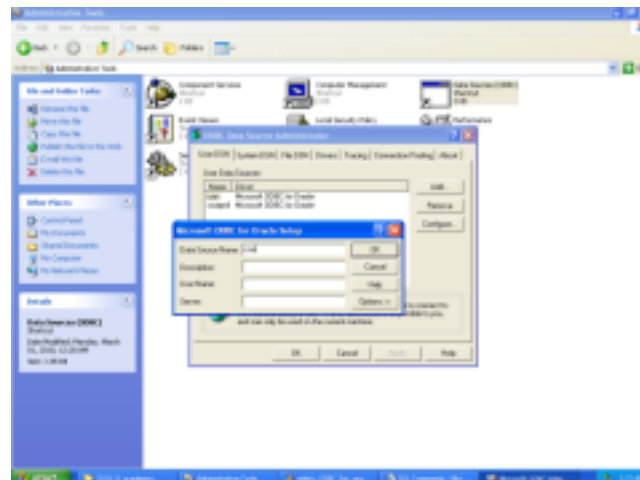
AGTI's, DACOE Karad Page 15

Advanced Database Systems Lab Manual

**Step 4:** Select Microsoft ODBC for Oracle then click Finish.



**Step 5:** Enter Data Source Name(in prevision example it is “cse”) and click Ok



After compilation it will create DSN for oracle. This DSN is used during java program for connection con = DriverManager.getConnection("jdbc:odbc:cse", "tecse", "tecse");

### How to run SQL queries in oracle 10 g.

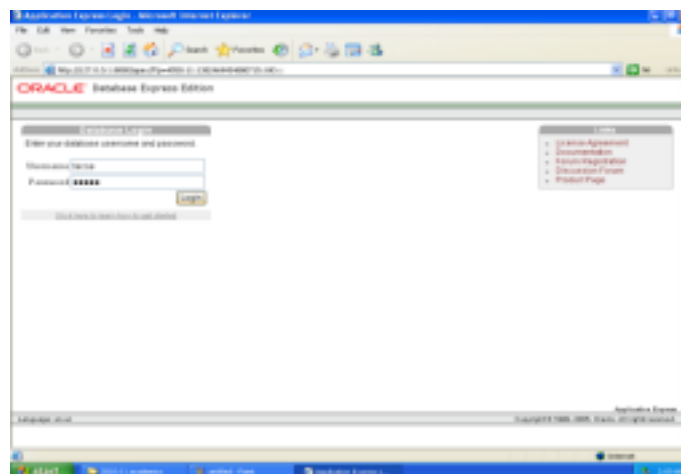
**Step1 :** Select Oracle Database 10g Express Edition - Go To Database Home Page from Start menu

AGTI's, DACOE Karad Page 16

Advanced Database Systems Lab Manual

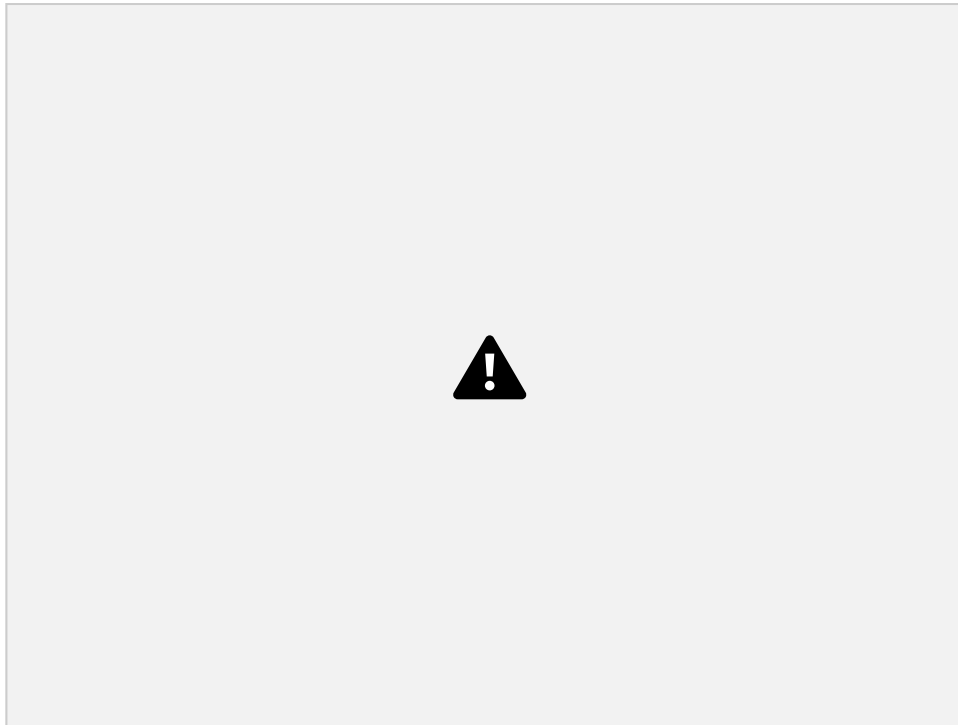


**Step 2:** login to oracle 10 g using username and password



In our example we used username: tecse and password; tecse





e.g. select \*

from classtecse

**Also we create different users and assign privileges through create user.**

### Role Name Description

CONN ECT
-------------

Enables a user to connect to the database. Grant this role to any user or application that needs database access.

AGTI's, DACOE Karad Page 18

Advanced Database Systems Lab Manual

### Role Name Description

RESOURC E
DBA

Enables a user to create certain types of schema objects in his own schema. Grant this role only to developers and to other users that must create schema objects. This role grants a subset of the *create object* system privileges. For example, it grants the **CREATE TABLE** system privilege, but does not grant the **CREATE VIEW** system privilege. It grants only the following privileges: **CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE.**

### CREATE TRIGGER, CREATE TYPE

Enables a user to perform most administrative functions, including creating users and granting privileges; creating and granting roles; creating and dropping schema objects in other users'

schemas; and more. It grants all system privileges, but does not include the privileges to start up or shut down the database. It is by default granted to user SYSTEM.

**Conclusion:** We have install Oracle Database 10g Express Edition (Oracle Database XE) successfully and implement basic utilities of SQL queries on relational database.

## **Experiment No.2**

**Title:** Design and implement the Fragmentation schema & the Replication schema

**Aim:** To design and implement the Fragmentation schema & the Replication schema on distributed database

**Theory/ Design:** The design of a distributed database introduces three new issues:

1. How to partition the database into fragments.
2. Which fragments to replicate.
3. Where to locate those fragments and replicas.

Data fragmentation and data replication deal with the first two issues, and data allocation deals with the third issue.

### **1. Data fragmentation:**

It allows you to break a single object into two or more segments, or fragments. The object might be a user's database, a system database, or a table. Each fragment can be stored at any site over a computer network.

Information about data fragmentation is stored in the distributed data catalog (DDC), from which it is accessed by the TP to process user requests.

Data fragmentation strategies, as discussed here, are based at the table level and consist of dividing a table into logical fragments. You will explore three types of data fragmentation strategies: horizontal, vertical, and mixed. (Keep in mind that a fragmented table can always be re-created from its fragmented parts by a combination of unions and joins.)

- Horizontal fragmentation refers to the division of a relation into subsets (fragments) of tuples (rows). Each fragment is stored at a different node, and each fragment has unique rows. However, the unique rows all have the same attributes (columns). In short, each fragment represents the equivalent of a SELECT statement, with the WHERE clause on a single attribute.
- Vertical fragmentation refers to the division of a relation into attribute (column) subsets. Each subset (fragment) is stored at a different node, and each fragment has unique columns—with the exception of the key column, which is common to all fragments. This is the equivalent of the PROJECT statement in SQL.
- Mixed fragmentation refers to a combination of horizontal and vertical strategies. In other words, a table may be divided into several horizontal subsets (rows), each one having a subset of the attributes (columns).

**Problem Statement:** Employee database with logical units DEPARTMENT, PROJECT, WORKS\_ON and DEPENDENT

- Before we decide on how to distribute the data, we must determine the *logical units* of the database that are to be distributed. The simplest logical units are the relations themselves; that is, each *whole* relation is to be stored at a particular site. In our example, we must decide on a site to store each of the relations EMPLOYEE,
- We may want to store the database information relating to each department at the computer site for that department. A technique called *horizontal fragmentation* can be used to partition each relation by department.
- Horizontal Fragmentation: A **horizontal fragment** of a relation is a subset of the tuples in that relation. The tuples that belong to the horizontal fragment are specified by a condition on one or more attributes of the relation. Often, only a single attribute is involved. For example, we may

define three horizontal fragments on the EMPLOYEE relation in Figure 3.6 with the following conditions: (Dno = 5), (Dno = 4), and (Dno = 1)—each fragment contains the EMPLOYEE tuples working for a particular department. Similarly, we may define three horizontal fragments for the PROJECT relation, with the conditions (Dnum = 5), (Dnum = 4), and (Dnum = 1)—each



fragment contains the PROJECT tuples controlled by a particular department. **Horizontal fragmentation** divides a relation *horizontally* by grouping rows to create subsets of tuples, where each subset has a certain logical meaning. These fragments can then be assigned to different sites in the distributed system. **Derived horizontal fragmentation** applies the partitioning of a primary relation (DEPARTMENT in our example) to other secondary relations (EMPLOYEE and PROJECT in our example), which are related to the primary via a foreign key. This way, related data between the primary and the secondary relations gets fragmented in the same way.

- Vertical Fragmentation: Each site may not need all the attributes of a relation, which would indicate the need for a different type of fragmentation. **Vertical fragmentation** divides a relation “vertically” by columns. A **vertical fragment** of a relation keeps only certain attributes of the relation. For example, we may want to fragment the EMPLOYEE relation into two vertical fragments. The first fragment includes personal information—Name, Bdate, Address, and Sex—and the second includes work-related information—Ssn, Salary, Super\_ssn, and Dno. This vertical fragmentation is not quite proper, because if the two fragments are stored separately, we cannot put the original employee tuples back together, since there is *no common attribute* between the two fragments. It is necessary to include the primary key or some candidate key attribute in *every* vertical fragment so that the full relation can be reconstructed from the fragments. Hence, we must add the Ssn attribute to the personal information fragment.
- Notice that each horizontal fragment on a relation  $R$  can be specified in the relational algebra by a  $\sigma_{C_i}(R)$  operation. A set of horizontal fragments whose conditions  $C_1, C_2, \dots, C_n$  include all the tuples in  $R$ —that is, every tuple in  $R$  satisfies  $(C_1 \text{ OR } C_2 \text{ OR } \dots \text{ OR } C_n)$ —is called a **complete horizontal fragmentation** of  $R$ . In many cases a complete horizontal fragmentation is also **disjoint**; that is, no tuple in  $R$  satisfies  $(C_i \text{ AND } C_j)$  for any  $i \neq j$ . Our two earlier examples of horizontal fragmentation for the EMPLOYEE and PROJECT relations were both complete and disjoint. To reconstruct the relation  $R$  from a *complete* horizontal fragmentation, we need to apply the UNION operation to the fragments.
- A vertical fragment on a relation  $R$  can be specified by a  $\pi_{L_i}(R)$  operation in the relational algebra. A set of vertical fragments whose projection lists  $L_1, L_2, \dots, L_n$  include all the attributes in  $R$  but share only the primary key attribute of  $R$  is called a **complete vertical fragmentation** of  $R$ . In this case the projection lists satisfy the following two conditions:
  - $L_1 \cup L_2 \cup \dots \cup L_n = \text{ATTRS}(R)$ .
  - $L_i \cap L_j = \text{PK}(R)$  for any  $i \neq j$ , where  $\text{ATTRS}(R)$  is the set of attributes of  $R$  and  $\text{PK}(R)$  is the primary key of  $R$ .
- To reconstruct the relation  $R$  from a *complete* vertical fragmentation, we apply the OUTER UNION operation to the vertical fragments (assuming no horizontal fragmentation is used). Notice that we could also apply a FULL OUTER JOIN operation and get the same result for a complete vertical fragmentation, even when some horizontal fragmentation may also have been

applied. The two vertical fragments of the EMPLOYEE relation with projection lists  $L_1 = \{\text{Ssn, Name, Bdate, Address, Sex}\}$  and  $L_2 = \{\text{Ssn, Salary, Super\_ssn, Dno}\}$  constitute a complete

vertical fragmentation of EMPLOYEE.

- Two horizontal fragments that are neither complete nor disjoint are those defined on the EMPLOYEE relation by the conditions (Salary > 50000) and (Dno = 4); they may not include all EMPLOYEE tuples, and they may include common tuples. Two vertical fragments that are not complete are those defined by the attribute lists  $L_1 = \{\text{Name, Address}\}$  and  $L_2 = \{\text{Ssn, Name, Salary}\}$ ; these lists violate both conditions of a complete vertical fragmentation.
- Mixed (Hybrid) Fragmentation. We can intermix the two types of fragmentation, yielding a **mixed fragmentation**. For example, we may combine the horizontal and vertical fragmentations of the EMPLOYEE relation given earlier into a mixed fragmentation that includes six fragments. In this case, the original relation can be reconstructed by applying UNION and OUTER UNION (or OUTER JOIN) operations in the appropriate order. In general, a **fragment** of a relation  $R$  can be specified by a SELECT-PROJECT combination of operations  $\pi_L(\sigma_C(R))$ . If  $C = \text{TRUE}$  (that is, all tuples are selected) and  $L \neq \text{ATTRS}(R)$ , we get a vertical fragment, and if  $C \neq \text{TRUE}$  and  $L = \text{ATTRS}(R)$ , we get a horizontal fragment. Finally, if  $L \neq \text{ATTRS}(R)$  and  $C \neq \text{TRUE}$ , we get a mixed fragment. Notice that a relation can itself be considered a fragment with  $C = \text{TRUE}$  and  $L = \text{ATTRS}(R)$ . In the following discussion, the term *fragment* is used to refer to a relation or to any of the preceding types of fragments.
- A **fragmentation schema** of a database is a definition of a set of fragments that includes *all* attributes and tuples in the database and satisfies the condition that the whole database can be reconstructed from the fragments by applying some sequence of OUTER UNION (or OUTER JOIN) and UNION operations. It is also sometimes useful—although not necessary—to have all the fragments be disjoint except for the repetition of primary keys among vertical (or mixed) fragments. In the latter case, all replication and distribution of fragments is clearly specified at a subsequent stage, separately from fragmentation.
- An **allocation schema** describes the allocation of fragments to sites of the DDBS; hence, it is a mapping that specifies for each fragment the site(s) at which it is stored. If a fragment is stored at more than one site, it is said to be **replicated**. We discuss data replication and allocation next.

## 2. Data Replication and Allocation:

- Replication is useful in improving the availability of data. The most extreme case is replication of the *whole database* at every site in the distributed system, thus creating a **fully replicated distributed database**. This can improve availability remarkably because the system can continue to operate as long as at least one site is up. It also improves performance of retrieval for global queries because the results of such queries can be obtained locally from any one site; hence, a retrieval query can be processed at the local site where it is submitted, if that site includes a server module. The disadvantage of full replication is that it can slow down update operations drastically, since a single logical update must be performed on every copy of the database to keep the copies consistent. This is especially true if many copies of the database exist. Full replication makes the concurrency control and recovery techniques more expensive than they would be if there was no replication, as we will see in Section 25.7.

- The other extreme from full replication involves having **no replication**—that is, each fragment is stored at exactly one site. In this case, all fragments *must be* disjoint, except for the repetition of primary keys among vertical (or mixed) fragments. This is also called **non redundant allocation**.
- Between these two extremes, we have a wide spectrum of **partial replication** of the data—that is, some fragments of the database may be replicated whereas others may not. The number of copies of each fragment can range from one up to the total number of sites in the distributed system. A special case of partial replication is occurring heavily in applications where mobile workers—such as sales forces, financial planners, and claims adjustors—carry partially replicated databases with them on laptops and PDAs and synchronize them periodically with the server database. A description of the replication of fragments is sometimes called a **replication schema**.
- Each fragment or each copy of a fragment—must be assigned to a particular site in the distributed system. This process is called **data distribution** (or **data allocation**). The choice of sites and the degree of replication depend on the performance and availability goals of the system and on the types and frequencies of transactions submitted at each site. For example, if high availability is required, transactions can be submitted at any site, and most transactions are retrieval only, a fully replicated database is a good choice. However, if certain transactions that access particular parts of the database are mostly submitted at a particular site, the corresponding set of fragments can be allocated at that site only. Data that is accessed at multiple sites can be replicated at those sites. If many updates are performed, it may be useful to limit replication. Finding an optimal or even a good solution to distributed data allocation is a complex optimization problem.

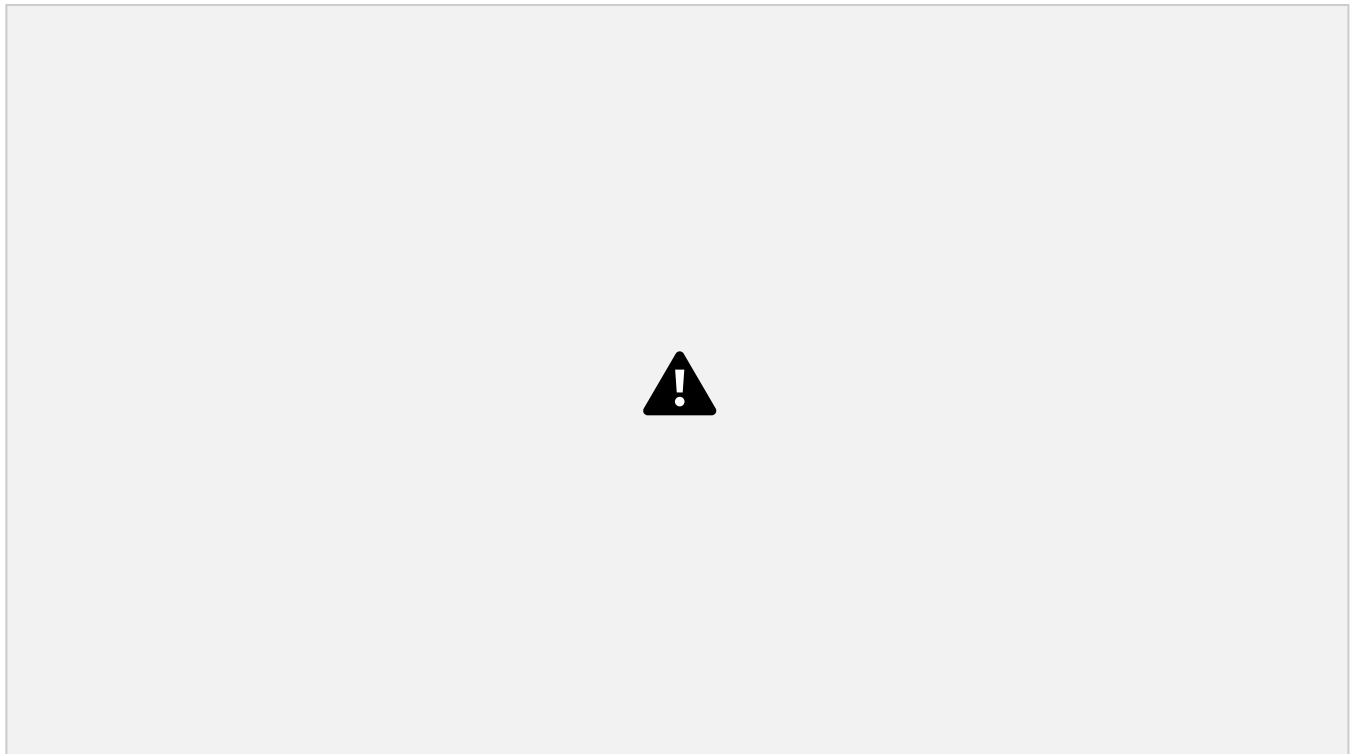
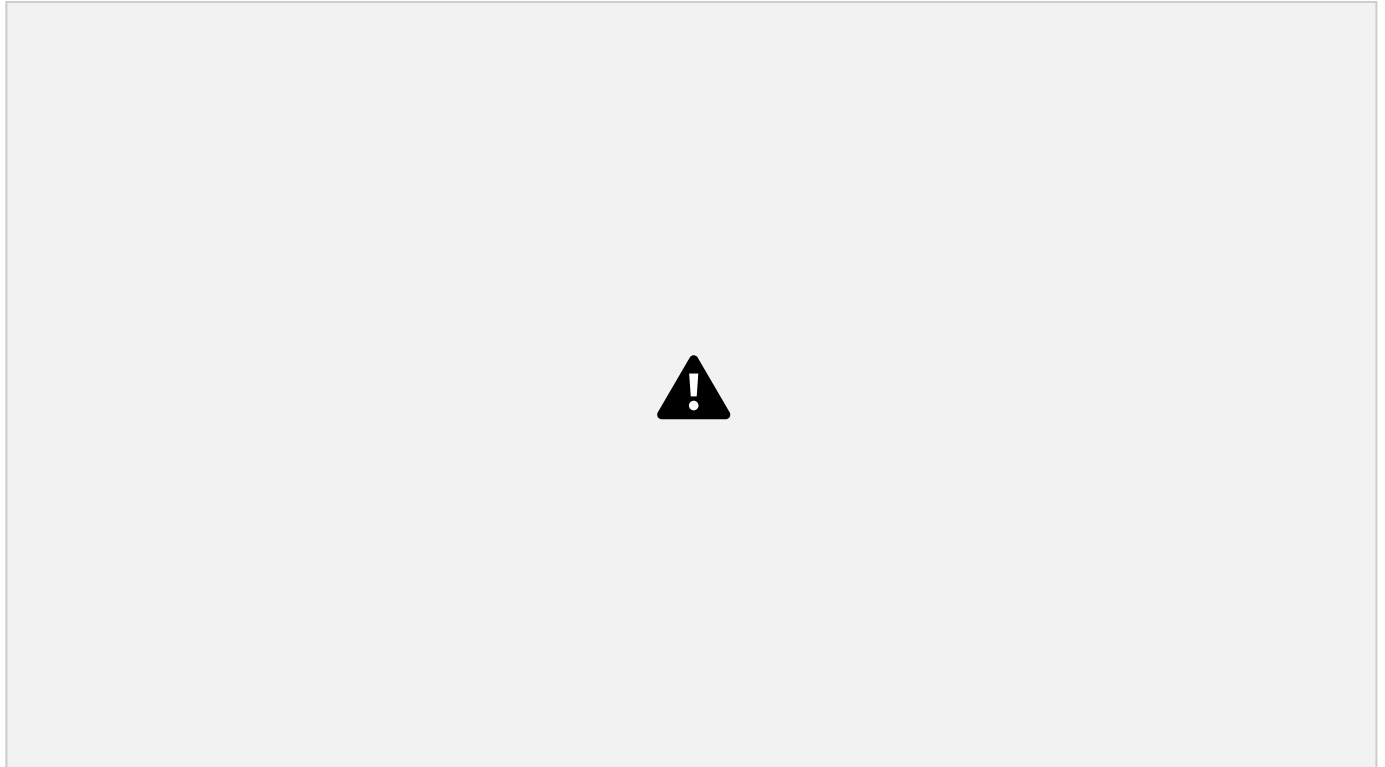
### 3. Example of Fragmentation, Allocation, and Replication

- We now consider an example of fragmenting and distributing the company data-base. Suppose that the company has three computer sites— one for each current department. Sites 2 and 3 are for departments 5 and 4, respectively. At each of these sites, we expect frequent access to the EMPLOYEE and PROJECT information for the employees *who work in that department* and the projects *controlled by that department*. Further, we assume that these sites mainly access the Name, Ssn, Salary, and Super\_ssn attributes of EMPLOYEE. Site 1 is used by company headquarters and accesses all employee and project information regularly, in addition to keeping track of DEPENDENT information for insurance purposes.
- According to these requirements, the whole database can be stored at site 1. To determine the fragments to be replicated at sites 2 and 3, first we can horizontally fragment DEPARTMENT by its key Dnumber. Then we apply derived fragmentation to the EMPLOYEE, PROJECT, and DEPT\_LOCATIONS relations based on their foreign keys for department number— called Dno, Dnum, and Dnumber, respectively, in Figure 3.5. We can vertically fragment the resulting EMPLOYEE fragments to include only the attributes {Name, Ssn, Salary, Super\_ssn, Dno}. Figure 25.8 shows the mixed fragments EMPD\_5 and EMPD\_4, which include the EMPLOYEE tuples satisfying the conditions Dno = 5 and Dno = 4, respectively. The horizontal fragments of PROJECT, DEPARTMENT, and DEPT\_LOCATIONS are similarly fragmented by department number. All these fragments—stored at sites 2 and 3—are replicated because they are also stored at headquarters—site 1.

**Implementation:**

- We must now fragment the WORKS\_ON relation and decide which fragments of WORKS\_ON to store at sites 2 and 3. We are confronted with the problem that no attribute of WORKS\_ON directly indicates the department to which each tuple belongs. In fact, each tuple in WORKS\_ON relates an employee  $e$  to a project  $P$ . We could fragment WORKS\_ON based on the department  $D$  in which  $e$  works *or* based on the department  $D$  that controls  $P$ . Fragmentation becomes easy if we have a constraint stating that  $D = D$  for all WORKS\_ON tuples—that is, if employees can work only on projects controlled by the department they work for. However, there is no such constraint in our database in Figure 3.6. For example, the WORKS\_ON tuple  $\langle 333445555, 10, 10.0 \rangle$  relates an employee who works for department 5 with a project controlled by department 4. In this case, we could fragment WORKS\_ON based on the department in which the employee works (which is expressed by the condition  $C$ ) and then fragment further based on the department that controls the projects that employee is working on, as shown in Figure 25.9.
- In Figure 25.9, the union of fragments  $G_1$ ,  $G_2$ , and  $G_3$  gives all WORKS\_ON tuples for employees who work for department 5. Similarly, the union of fragments  $G_4$ ,  $G_5$ , and  $G_6$  gives all WORKS\_ON tuples for employees who work for department 4. On the other hand, the union of fragments  $G_1$ ,  $G_4$ , and  $G_7$  gives all WORKS\_ON tuples for projects controlled by department 5. The condition for each of the fragments  $G_1$  through  $G_9$  is shown in Figure 25.9. The relations that represent M:N relationships, such as WORKS\_ON, often have several possible logical

fragmentations. In our distribution in Figure 25.8, we choose to include all fragments that can be joined to



**Conclusion:** We have designed and implemented the Fragmentation schema & the Replication schema on

### Experiment No. 3

**Title:** Implementation of 2 Phase Commit protocol for distributed databases.

**Aim:** To implement 2 Phase Commit protocol for distributed databases

#### **Theory / Design:**

During normal execution, each site maintains a log, and the actions of a sub transaction are logged at the site where it executes. A commit protocol is followed to ensure that all sub transactions of a given transaction either commit or abort uniformly. The transaction manager at the site where the transaction originated is called the coordinator for the transaction; transaction managers at sites where its sub transactions execute are called subordinates. When the user decides to commit a transaction, the commit command is sent to the coordinator for the transaction. This initiates the 2PC protocol:

1. The coordinator sends a *prepare* message to each subordinate.
2. When a subordinate receives a *prepare* message, it decides whether to abort or commit its sub transaction. It force-writes and abort or prepare log record, and *then* sends a *no* or *yes* message to the coordinator.
3. If the coordinator receives *yes* messages from all subordinates, it force-writes a commit log record and then sends a *commit* message to all subordinates. If it receives even one *no* message, or does not receive any response from some subordinate for a specified time-out interval, it force-writes an abort log record, and then sends an *abort* message to all subordinates.
4. When a subordinate receives an *abort* message, it force-writes an abort log record, sends an *ack* message to the coordinator, and aborts the sub transaction. When a subordinate receives a *commit* message, it force-writes a commit log record, sends an *ack* message to the coordinator, and commits the sub transaction.
5. After the coordinator has received *ack* messages from all subordinates, it writes an end log record for the transaction.

The name *Two-Phase Commit* reflects the fact that two rounds of messages are exchanged: First a voting phase, then a termination phase, both initiated by the coordinator. The basic principle is that any of the transaction managers involved (including the coordinator) can unilaterally abort a transaction, whereas there must be unanimity to commit a transaction.

#### **Phase 1 - Prepare:**

1. The TC writes a local **<Prepare T>** log and persists it. The TC sends a "Prepare T" message to all participants. 2. Each participant TM receives the "Prepare T" message and decides whether to commit the transaction based on its own situation:

- If a TM decides to commit the transaction, it writes a **<Ready T>** log and persists it, and then sends a "Ready T" message to the TC.
- If a TM decides not to commit the transaction, it writes an **<Abort T>** log and persists it, and sends an "Abort T" message to the TC. Then, the TM enters the transaction abortion process locally.

**Phase 2 - Commit:**

1. When the TC has received responses from all nodes or the waiting timer times out, the TC decides whether to commit or abort the transaction.

- If all participants respond with a "Ready T" message, the TC writes a `<Commit T>` log and persists it, and then sends a "Commit T" message to all participants.
- If the TC receives an "Abort T" response from at least one participant, or if any participant fails to respond within the timeout period, the TC writes a `<Abort T>` log, and then sends an "Abort T" message to all participants.

2. After the participants receive the message from the TC, they write `<Commit T>` or `<Abort T>` logs and persist them. The 2PC protocol can ensure a key point in the execution of distributed transactions: A participant can decide whether to abort the transaction at any time before it sends a "Ready T" message to the TC. Once this message is sent, the transaction enters the ready state, in which commit and abortion are completely controlled by the TC. When a participant sends a "Ready T" message, it essentially sends a formal and irreversible commitment to the TC. To ensure this commitment, the participant must persist all the necessary information before it sends a "Ready T" message. Otherwise, if the participant crashes after it sends a "Ready T" message, it may be unable to keep the preceding commitment after a restart. In Phase 2, when the coordinator has written a `<Commit T>` or `<Abort T>` log, the result of the transaction is decided, that is, it will not change anymore.



**Algorithm:-**

### At coordinator

1. start
2. Create ServerSocket object and accept all subordinates request and connect with them.
3. Send a *prepare* message to each subordinate.

AGTI's, DACOE Karad Page 27

Advanced Database Systems Lab Manual

4. Receives *yes or no* messages from all subordinates.
5. If *yes* messages from all subordinates are received, write a commit log record and then send a *commit* message to all subordinates.
6. If *no* messages from all subordinates or even one are received, write an abort log record and then send an *abort* message to all subordinates.
7. Received *ack* messages from all subordinates, write an end log record for the transaction.
8. Stop.

### At subordinate

1. start
2. Create Socket object and connect to coordinate.
3. Receives *prepare* messages from coordinate.
4. Send a *yes or no* message to coordinate.
5. Receives commit or abort message from coordinate.
6. Send a ack to coordinate.
7. Stop.

### Implementation of Program:

#### 1. Coordinate

```
import java.io.*;
import java.net.*;
class CoOrdinate
{
    public static void main(String [] args)throws Exception
    {
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        ServerSocket ss=new ServerSocket(4545);
        System.out.println("\nEnter No. of Subordinate:");
        int cnt=Integer.parseInt(br.readLine());
        Socket []s=new Socket[cnt];
        DataInputStream []dis=new DataInputStream[cnt];
        DataOutputStream []dos=new DataOutputStream[cnt];
        for(int i=0;i<cnt;i++)
        {
            s[i]=ss.accept();
```



```

        dos[i]=new DataInputStream(s[i].getInputStream());
        dos[i]=new DataOutputStream(s[i].getOutputStream());
    }
    String pr=new String("prepare");
    System.out.println("Sending prepare message");
    for(int i=0;i<cnt;i++){

```

AGTI's, DACOE Karad Page 28

Advanced Database Systems Lab Manual

```

dos[i].writeUTF(pr);
    }
    String []in=new String[cnt];
    boolean flg=true;
    for(int i=0;i<cnt;i++)
    {
        in[i]=dis[i].readUTF();
        if(in[i].equals("no"))
            flg=false;
    }
    if(flg==false)
    {
        pr="abort";
        System.out.println("Sending abort msg");
        for(int i=0;i<cnt;i++)
        {
            dos[i].writeUTF(pr);
        }
    }
    else
    {
        pr="commit";
        System.out.println("Sending commit msg");
        for(int i=0;i<cnt;i++)
        {
            dos[i].writeUTF(pr);
        }
        for(int i=0;i<cnt;i++)
        {
            System.out.println("Receiving ack");
        }
    }
}

```

```
    }  
}
```

## 2.Subordinate

```
import java.io.*;  
import java.net.*;  
class SubOrdinate  
{  
    public static void main(String [] args)throws Exception  
    {  
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));  
        Socket s=new Socket("localhost",4545);  
        DataInputStream dis =new DataInputStream(s.getInputStream());  
        DataOutputStream dos=new DataOutputStream(s.getOutputStream());  
        String out=new String();  
        String in=dis.readUTF();  
        System.out.println("Receiving "+in+" msg.");  
        System.out.println("SubOrdinate ready:'yes' or 'no:");  
        out=br.readLine();  
        System.out.println("Sending "+out);  
        dos.writeUTF(out);  
        in=dis.readUTF();  
        System.out.println("Reciving "+in+" msg");  
        if(in.equals("commit"))  
        {  
            out="ack";  
            System.out.println("Sending Ack");  
        }  
    }  
}
```

### Output:-

#### At coordinator

Enter No. of Subordinate:

4

Sending prepare message

Sending abort message

**At subordinate1**

Receiving prepare msg.

Subordinate ready: 'yes' or 'no': yes

Sending yes

Receiving abort message.

AGTI's, DACOE Karad Page 30

**Advanced Database Systems Lab Manual**

**At subordinate2**

Receiving prepare msg.

Subordinate ready: 'yes' or 'no': yes

Sending yes

Receiving abort message

**At subordinate3**

Receiving prepare msg.

Subordinate ready: 'yes' or 'no': yes

Sending yes

Receiving abort message

**At subordinate4**

Receiving prepare msg.

Subordinate ready: 'yes' or 'no': no

Sending no.

Receiving abort message

**Conclusion:** We have studied and implement 2 Phase Commit protocol for distributed databases.

## Experiment No. 4

**Title :** Demonstrate SQL Functions, Procedures, Cursors, and triggers using PL/SQL, Views.

**Aim :** To implement SQL Functions, Procedures, Cursors, and triggers using PL/SQL, Views.

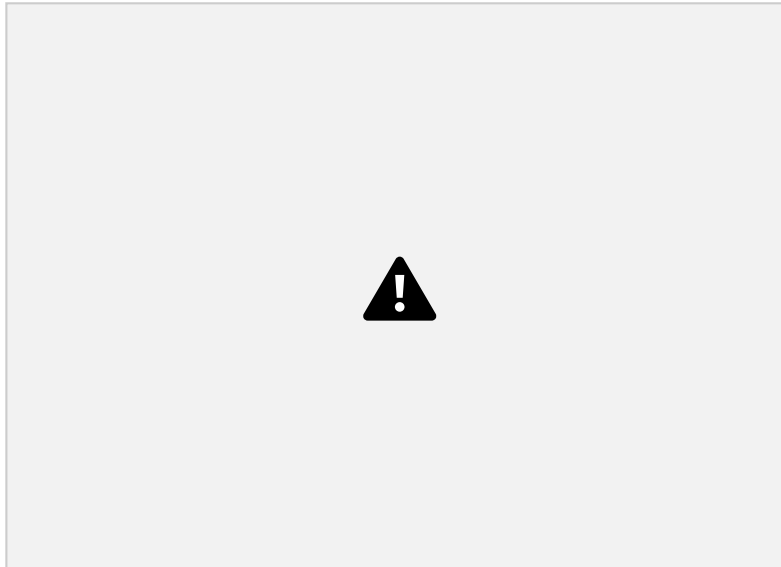
### Theory :

**PL/SQL Function:** Functions is a standalone PL/SQL subprogram. Like PL/SQL procedure, functions have a unique name by which it can be referred. These are stored as PL/SQL database objects. Below are some of the characteristics of functions.

- Functions are a standalone block that is mainly used for calculation purpose.
- Function use RETURN keyword to return the value, and the datatype of this is defined at the time of creation.
- A Function should either return a value or raise the exception, i.e. return is mandatory in functions.
- Function with no DML statements can be directly called in SELECT query whereas the function with DML operation can only be called from other PL/SQL blocks.
- It can have nested blocks, or it can be defined and nested inside the other blocks or packages. • It contains declaration part (optional), execution part, exception handling part (optional). • The values can be passed into the function or fetched from the procedure through the parameters. • These

parameters should be included in the calling statement.

- A PLSQL function can also return the value through OUT parameters other than using RETURN.
- Since it will always return the value, in calling statement it always accompanies with assignment operator to populate the variables.



### **PL/SQL Procedure:**

A PL/SQL procedure is a reusable unit that encapsulates specific business logic of the application. Technically speaking, a PL/SQL procedure is a named block stored as a schema object in the Oracle

AGTI's, DACOE Karad Page 32

**Advanced Database Systems Lab Manual**

Database.

The following illustrates the basic syntax of creating a procedure in PL/SQL:

**CREATE [OR REPLACE] PROCEDURE procedure\_name (parameter\_list)**

**IS**

**[declaration statements]**

**BEGIN**

**[execution statements]**

**EXCEPTION**

**[exception handler]**

**END [procedure\_name ];**

## **PL/SQL procedure header:**

A procedure begins with a header that specifies its name and an optional parameter list.

Each parameter can be in either IN, OUT, or INOUT mode. The parameter mode specifies whether a parameter can be read from or write to.

**IN :** An IN parameter is read-only. You can reference an IN parameter inside a procedure, but you cannot change its value. Oracle uses IN as the default mode. It means that if you don't specify the mode for a parameter explicitly, Oracle will use the IN mode.

**OUT:** An OUT parameter is writable. Typically, you set a returned value for the OUT parameter and return it to the calling program. Note that a procedure ignores the value that you supply for an OUT parameter.

**INOUT:** An INOUT parameter is both readable and writable. The procedure can read and modify it.

Note that OR REPLACE option allows you to overwrite the current procedure with the new code.

## **PL/SQL procedure body**

AGTI's, DACOE Karad Page 33

**Advanced Database Systems Lab Manual**

Similar to an anonymous block, the procedure body has three parts. The executable part is mandatory whereas the declarative and exception-handling parts are optional. The executable part must contain at least one executable statement.

### **1) Declarative part**

In this part, you can declare variables, constants, cursors, etc. Unlike an anonymous block, a declaration part of a procedure does not start with the DECLARE keyword.

### **2) Executable part**

This part contains one or more statements that implement specific business logic. It might contain only a NULL statement.

### **3) Exception-handling part**

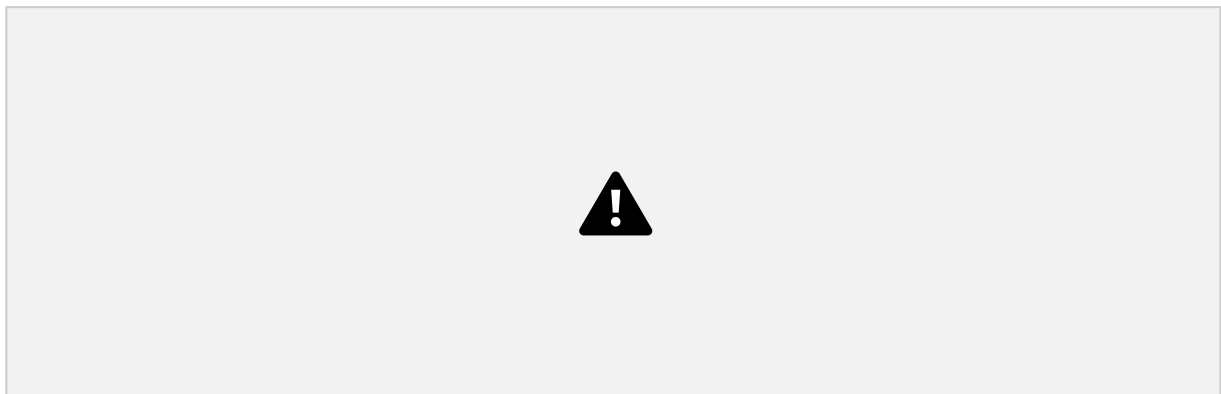
This part contains the code that handles exceptions.

- **PL/SQL Package:**

In PL/SQL, a package is a schema object that contains definitions for a group of related functionalities. A package includes variables, constants, cursors, exceptions, procedures, functions, and subprograms. It is compiled and stored in the Oracle Database.

Typically, a package has a specification and a body. A package specification is mandatory while the package body can be required or optional, depending on the package specification.

The following picture illustrates PL/SQL packages:



### **Package specification**

AGTI's, DACOE Karad Page 34

**Advanced Database Systems Lab Manual**

The package specification declares the public objects that are accessible from outside the package.

If a package specification whose public objects include cursors and subprograms, then it must have a body which defines queries for the cursors and code for the subprograms.

### **Package body**

A package body contains the implementation of the cursors or subprograms declared in the package specification. In the package body, you can declare or define private variables, cursors, etc., used only by package body itself. A package body can have an initialization part whose statements initialize variables or perform other one-time setups for the whole package. A package body can also have an exception-handling part used to handle exceptions.

### **Difference between Procedure & Function:**



### **Cursor:**

Oracle creates a memory area, known as the context area, for processing an SQL statement, which contains all the information needed for processing the statement; for example, the number of rows processed, etc.

A **cursor** is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the **active set**.

You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors –

- Implicit cursors
- Explicit cursors

### **Implicit Cursors**

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the



information in it.

Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted. For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

In PL/SQL, you can refer to the most recent implicit cursor as the **SQL cursor**, which always has attributes such as **%FOUND**, **%ISOPEN**, **%NOTFOUND**, and **%ROWCOUNT**. The SQL cursor has additional attributes, **%BULK\_ROWCOUNT** and **%BULK\_EXCEPTIONS**, designed for use with the **FORALL** statement. The following table provides the description of the most used attributes –

S.No	Attribute & Description
1	<b>%FOUND</b> Returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows or a SELECT INTO statement returned one or more rows. Otherwise, it returns FALSE.
2	<b>%NOTFOUND</b> The logical opposite of %FOUND. It returns TRUE if an INSERT, UPDATE, or DELETE statement affected no rows, or a SELECT INTO statement returned no rows. Otherwise, it returns FALSE.
3	<b>%ISOPEN</b> Always returns FALSE for implicit cursors, because Oracle closes the SQL cursor automatically after executing its associated SQL statement.

4	<b>%ROWCOUNT</b> Returns the number of rows affected by an INSERT, UPDATE, or DELETE statement, or returned by a SELECT INTO statement.
---	--

Any SQL cursor attribute will be accessed as **sql%attribute\_name** as shown below in the example.

### Example

We will be using the CUSTOMERS table we had created and used in the previous chapters.

```
Select * from customers;
```

```
+-----+-----+-----+-----+
| ID | NAME | AGE | ADDRESS | SALARY |
+-----+-----+-----+-----+
| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |
| 2 | Khilan | 25 | Delhi | 1500.00 |
| 3 | kaushik | 23 | Kota | 2000.00 |
```

```

| 4 | Chaitali | 25 | Mumbai | 6500.00 |
| 5 | Hardik | 27 | Bhopal | 8500.00 |
| 6 | Komal | 22 | MP | 4500.00 |
+-----+

```

The following program will update the table and increase the salary of each customer by 500 and use the **SQL%ROWCOUNT** attribute to determine the number of rows affected –

```

DECLARE
total_rows number(2);
BEGIN
UPDATE customers
SET salary = salary + 500;
IF sql%notfound THEN
dbms_output.put_line('no customers selected');
ELSIF sql%found THEN
total_rows := sql%rowcount;
dbms_output.put_line( total_rows || ' customers selected ');
END IF;
END;
/

```

When the above code is executed at the SQL prompt, it produces the following result –

```
6 customers selected
```

PL/SQL procedure successfully completed.

If you check the records in customers table, you will find that the rows have been updated –

```
Select * from customers;
```

```

+-----+
| ID | NAME | AGE | ADDRESS | SALARY |
+-----+
| 1 | Ramesh | 32 | Ahmedabad | 2500.00 |
| 2 | Khilan | 25 | Delhi | 2000.00 |
| 3 | kaushik | 23 | Kota | 2500.00 |
| 4 | Chaitali | 25 | Mumbai | 7000.00 |
| 5 | Hardik | 27 | Bhopal | 9000.00 |
| 6 | Komal | 22 | MP | 5000.00 |
+-----+

```

## Explicit Cursors

Explicit cursors are programmer-defined cursors for gaining more control over the **context area**. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a **SELECT** Statement which returns more than one row.

The syntax for creating an explicit cursor is –

```
CURSOR cursor_name IS select_statement;
```

Working with an explicit cursor includes the following steps –

- Declaring the cursor for initializing the memory
- Opening the cursor for allocating the memory

- Fetching the cursor for retrieving the data
- Closing the cursor to release the allocated memory

## Declaring the Cursor

Declaring the cursor defines the cursor with a name and the associated SELECT statement. For example –

```
CURSOR c_customers IS
SELECT id, name, address FROM customers;
```

## Opening the Cursor

Opening the cursor allocates the memory for the cursor and makes it ready for fetching the rows returned by the SQL statement into it. For example, we will open the above defined cursor as follows –

```
OPEN c_customers;
```

## Fetching the Cursor

Fetching the cursor involves accessing one row at a time. For example, we will fetch rows from the above-opened cursor as follows –

```
FETCH c_customers INTO c_id, c_name, c_addr;
```

## Closing the Cursor

Closing the cursor means releasing the allocated memory. For example, we will close the above-opened cursor as follows –

```
CLOSE c_customers;
```

## Example

Following is a complete example to illustrate the concepts of explicit cursors &minua;

```
DECLARE
  c_id customers.id%type;
  c_name customers.name%type;
  c_addr customers.address%type;
  CURSOR c_customers is
  SELECT id, name, address FROM customers;
BEGIN
  OPEN c_customers;
  LOOP
  FETCH c_customers into c_id, c_name, c_addr;
  EXIT WHEN c_customers%notfound;
```

```
dbms_output.put_line(c_id || ' ' || c_name || ' ' || c_addr);
END LOOP;
CLOSE c_customers;
END;
/
```

When the above code is executed at the SQL prompt, it produces the following result –

```
1 Ramesh Ahmedabad
2 Khilan Delhi
3 kaushik Kota
4 Chaitali Mumbai
5 Hardik Bhopal
6 Komal MP
```

PL/SQL procedure successfully completed.

**Trigger:** Triggers are stored programs, which are automatically executed or fired when some events occur. Triggers are, in fact, written to be executed in response to any of the following events –

- A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE) •
- A **database definition (DDL)** statement (CREATE, ALTER, or DROP).
- A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

### Benefits of Triggers:

Triggers can be written for the following purposes –

- Generating some derived column values automatically
- Enforcing referential integrity
- Event logging and storing information on table access
- Auditing
- Synchronous replication of tables
- Imposing security authorizations
- Preventing invalid transactions

### Creating Triggers

The syntax for creating a trigger is –

```
CREATE [OR REPLACE ] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
ON table_name
[REFERENCING OLD AS o NEW AS n]
[FOR EACH ROW]
WHEN (condition)
DECLARE
  Declaration-statements
BEGIN
```

```
EXCEPTION
  Exception-handling-statements
END;
```

Where,

- CREATE [OR REPLACE] TRIGGER trigger\_name – Creates or replaces an existing trigger with the *trigger\_name*.
- {BEFORE | AFTER | INSTEAD OF} – This specifies when the trigger will be executed. The INSTEAD OF clause is used for creating trigger on a view.
- {INSERT [OR] | UPDATE [OR] | DELETE} – This specifies the DML operation.
- [OF col\_name] – This specifies the column name that will be updated.
- [ON table\_name] – This specifies the name of the table associated with the trigger.
- [REFERENCING OLD AS o NEW AS n] – This allows you to refer new and old values for various DML statements, such as INSERT, UPDATE, and DELETE.
- [FOR EACH ROW] – This specifies a row-level trigger, i.e., the trigger will be executed for each row being affected. Otherwise the trigger will execute just once when the SQL statement is executed, which is called a table level trigger.
- WHEN (condition) – This provides a condition for rows for which the trigger would fire. This clause is valid only for row-level triggers.

### Example

To start with, we will be using the CUSTOMERS table we had created and used in the previous chapters –

```
Select * from customers;
```

```
+-----+-----+-----+-----+-----+
| ID | NAME | AGE | ADDRESS | SALARY |
+-----+-----+-----+-----+-----+
| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |
| 2 | Khilan | 25 | Delhi | 1500.00 |
| 3 | kaushik | 23 | Kota | 2000.00 |
| 4 | Chaitali | 25 | Mumbai | 6500.00 |
| 5 | Hardik | 27 | Bhopal | 8500.00 |
| 6 | Komal | 22 | MP | 4500.00 |
+-----+-----+-----+-----+-----+
```

The following program creates a **row-level** trigger for the customers table that would fire for INSERT or UPDATE or DELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old values and new values –

```

CREATE OR REPLACE TRIGGER display_salary_changes
BEFORE DELETE OR INSERT OR UPDATE ON customers
FOR EACH ROW
WHEN (NEW.ID > 0)
DECLARE
    sal_diff number;
BEGIN
    sal_diff := :NEW.salary - :OLD.salary;
    dbms_output.put_line('Old salary: ' || :OLD.salary);
    dbms_output.put_line('New salary: ' || :NEW.salary);
    dbms_output.put_line('Salary difference: ' || sal_diff);
END;
/

```

When the above code is executed at the SQL prompt, it produces the following result –

Trigger created.

The following points need to be considered here –

- OLD and NEW references are not available for table-level triggers, rather you can use them for record level triggers.
- If you want to query the table in the same trigger, then you should use the AFTER keyword, because triggers can query the table or change it again only after the initial changes are applied and the table is back in a consistent state.
- The above trigger has been written in such a way that it will fire before any DELETE or INSERT or UPDATE operation on the table, but you can write your trigger on a single or multiple operations, for example BEFORE DELETE, which will fire whenever a record will be deleted using the DELETE operation on the table.

### Triggering a Trigger

Let us perform some DML operations on the CUSTOMERS table. Here is one INSERT statement, which will create a new record in the table –

```

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (7, 'Kriti', 22, 'HP', 7500.00 );

```

When a record is created in the CUSTOMERS table, the above create trigger, **display\_salary\_changes** will be fired and it will display the following result –

```

Old salary:
New salary: 7500
Salary difference:

```

Because this is a new record, old salary is not available and the above result comes as null. Let us now perform one more DML operation on the CUSTOMERS table. The UPDATE statement will update an existing record in the table –

```

UPDATE customers
SET salary = salary + 500
WHERE id = 2;

```

When a record is updated in the CUSTOMERS table, the above create trigger, **display\_salary\_changes** will be fired and it will display the following result –

```
Old salary: 1500
New salary: 2000
Salary difference: 500
```

**VIEW** is a virtual table that does not physically exist. Rather, it is created by a query joining one or more tables.

## Create VIEW

### Syntax

The syntax for the CREATE VIEW Statement in Oracle/PLSQL is:

```
CREATE VIEW view_name AS
```

AGTI's, DACOE Karad Page 41

Advanced Database Systems Lab Manual

```
SELECT columns
FROM tables
[WHERE conditions];
```

### view\_name

The name of the Oracle VIEW that you wish to create.

### WHERE conditions

Optional. The conditions that must be met for the records to be included in the VIEW.

### Example

Here is an example of how to use the Oracle CREATE VIEW:

```
CREATE VIEW sup_orders AS
SELECT suppliers.supplier_id, orders.quantity, orders.price
FROM suppliers
INNER JOIN orders
ON suppliers.supplier_id = orders.supplier_id
WHERE suppliers.supplier_name = 'Microsoft';
```

This Oracle CREATE VIEW example would create a virtual table based on the result set of the SELECT statement. You can now query the Oracle VIEW as follows:

```
SELECT *
```

```
FROM sup_orders;
```

## Update VIEW

You can modify the definition of an Oracle VIEW without dropping it by using the Oracle CREATE OR REPLACE VIEW Statement.

### Syntax

The syntax for the CREATE OR REPLACE VIEW Statement in Oracle/PLSQL is:

```
CREATE OR REPLACE VIEW view_name AS
SELECT columns
FROM table
WHERE conditions;
```

**view\_name**

AGTI's, DACOE Karad Page 42

**Advanced Database Systems Lab Manual**

The name of the Oracle VIEW that you wish to create or replace.

### Example

Here is an example of how you would use the Oracle CREATE OR REPLACE VIEW Statement:

```
CREATE or REPLACE VIEW sup_orders AS
SELECT suppliers.supplier_id, orders.quantity, orders.price
FROM suppliers
INNER JOIN orders
ON suppliers.supplier_id = orders.supplier_id
WHERE suppliers.supplier_name = 'Apple';
```

This Oracle CREATE OR REPLACE VIEW example would update the definition of the Oracle VIEW called *sup\_orders* without dropping it. If the Oracle VIEW did not yet exist, the VIEW would merely be created for the first time.

## Drop VIEW

Once an Oracle VIEW has been created, you can drop it with the Oracle DROP VIEW Statement.

### Syntax

The syntax for the DROP VIEW Statement in Oracle/PLSQL is:

```
DROP VIEW view_name;
```



### **view\_name**

The name of the view that you wish to drop.

### **Example**

Here is an example of how to use the Oracle DROP VIEW Statement:

```
DROP VIEW sup_orders;
```

This Oracle DROP VIEW example would drop/delete the Oracle VIEW called *sup\_orders*.

## **IMPLEMENTATION:**

### **//FUNCTION//**

- create table college(cname char(20),cdept char(25),class char(20),student int)

Table created.

- insert into college values ('DACOE','IT','SE','60');

1 row(s) inserted.

- insert into college values ('TKIT','IT','TE','65');

1 row(s) inserted.

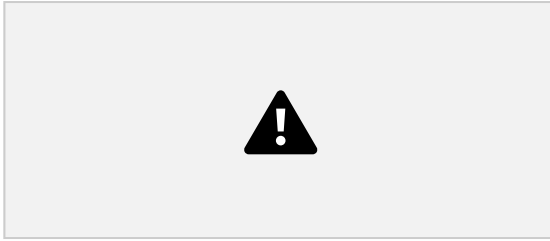
- insert into college values ('RIT','IT','BE','60');

1 row(s) inserted.

- insert into college values ('KIT','IT','FE','65');

1 row(s) inserted.

- select \*from college



4 rows returned in 0.00 seconds

```
CREATE OR REPLACE FUNCTION totals RETURN NUMBER AS
```

```
totalstudent NUMBER;  
BEGIN  
SELECT SUM(student)  
INTO totalstudent FROM college;  
RETURN totalstudent;  
END totals;
```

Function created

- select totals from college

AGTI's, DACOE Karad Page 44

Advanced Database Systems Lab Manual



4 rows returned in 0.02 seconds

```
//PROCEDURE//
```

```
//Creating staff table and inserting data
```

```
create table staff (sid int,sname varchar2(20),dname varchar2(20));
```

```
begin
```

```
insert into staff values(1,'ATS','CS');
insert into staff values(2,'AVS','CS');
insert into staff values(3,'VJY','Civil');
insert into staff values(4,'PRP','Mech');
end;
```

```
//Creating Procedure
```

```
create or replace procedure employees (name in staff.dname%type,counter out int) as
begin
select count(*) into counter from staff where dname=name;
end;
```

```
//Calling Procedure
```

```
declare
an int;
begin
```

```
AGTI's, DACOE Karad Page 45
```

**Advanced Database Systems Lab Manual**

```
employees('Civil',an);
dbms_output.put_line('Total employees '||an);
end;
```

```
//Creating Package
```

```
create or replace package pack as
procedure add_staff(id in staff.sid%type,name in staff.sname%type, dept in staff.dname%type);
end pack;
```

```
//Creating Package Body
```

```
create or replace package body pack as
```

```
procedure add_staff(id in staff.sid%type,name in staff.sname%type, dept in staff.dname%type) is
```

```
begin
```

```
insert into staff values(id,name,dept);
```

```
end add_staff;
```

```
end pack;
```

```
//Calling Procedure of Package
```

```
begin
```

```
pack.add_staff(6,'SVA','Chem');
```

```
end;
```

### **Conclusion:**

Thus we have studied and implemented SQL Functions, Procedures, Cursors, and triggers using PL/SQL, Views with an example.

## **Experiment No.5**

**Title:** Installation of MongoDB and Apache Cassandra.

**Aim:** To Install & Demonstrate MongoDB and Apache Cassandra platform on Ubuntu/Linux.

**Theory:** Related to MongoDB and Apache Cassandra and difference between these platforms.

### **Installation Steps for MongoDB:**

**Install MongoDB Community Edition on Ubuntu**

**NOTE**

## MongoDB Atlas

[MongoDB Atlas](#) is a hosted MongoDB service option in the cloud which requires no installation overhead and offers a free tier to get started.

Overview

Use this tutorial to install MongoDB 5.0 Community Edition on LTS (long-term support) releases of Ubuntu Linux using the [apt](#) package manager.



MongoDB Version

This tutorial installs MongoDB 5.0 Community Edition. To install a different version of MongoDB Community, use the version drop-down menu in the upper-left corner of this page to select the documentation for that version.

Considerations

Platform Support

### NOTE

#### EOL Notice

AGTI's, DACOE Karad Page 47

Advanced Database Systems Lab Manual

• MongoDB 5.0 Community Edition removes support for Ubuntu 16.04 on [x86\\_64](#) •

MongoDB 5.0 Community Edition removes support for Ubuntu 18.04 on [s390x](#)

MongoDB 5.0 Community Edition supports the following **64-bit** Ubuntu LTS (long-term support) releases on [x86\\_64](#) architecture:

• 20.04 LTS ("Focal")

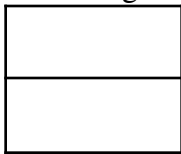
- 18.04 LTS ("Bionic")
- 16.04 LTS ("Xenial")

MongoDB only supports the 64-bit versions of these platforms.

MongoDB 5.0 Community Edition on Ubuntu also supports the [ARM64](#) architecture on select platforms.

#### Official MongoDB Packages

To install MongoDB Community on your Ubuntu system, these instructions will use the official `mongodb-org` package, which is maintained and supported by MongoDB Inc. The

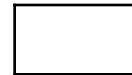
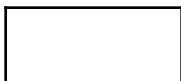


official `mongodb-org` package always contains the latest version of MongoDB, and is available from its own dedicated repo.

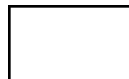
#### **IMPORTANT**



The `mongodb` package provided by Ubuntu is **not** maintained by MongoDB Inc. and conflicts with the



official `mongodb-org` package. If you have already installed the `mongodb` package on



your Ubuntu system, you **must** first uninstall the `mongodb` package before proceeding with these instructions. See [MongoDB Community Edition Packages](#) for the complete list of official packages.

#### Install MongoDB Community Edition

Follow these steps to install MongoDB Community Edition using the `apt` package manager.



## 1

AGTI's, DACOE Karad Page 48

Advanced Database Systems Lab Manual

Import the public key used by the package management system.

From a terminal, issue the following command to import the MongoDB public GPG Key from <https://www.mongodb.org/static/pgp/server-5.0.asc>:

```
wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | sudo apt-key add -
```



The operation should respond with an OK.

However, if you receive an error indicating that gnupg is not installed, you can:



1. Install gnupg and its required libraries using the following command:



```
sudo apt-get install gnupg
```

2. Once installed, retry importing the key:  

```
wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | sudo apt-key add -
```

## 2

Create a list file for MongoDB.

Create the list file `/etc/apt/sources.list.d/mongodb-org-5.0.list` for your version of Ubuntu. Click on the

appropriate tab for your version of Ubuntu. If you are unsure of what Ubuntu version the host is running, open a terminal or shell on the host and execute `lsb_release -dc`.

Ubuntu 20.04 (Focal) Ubuntu 18.04 (Bionic) Ubuntu 16.04 (Xenial)

The following instruction is for **Ubuntu 20.04 (Focal)**.

Create the `/etc/apt/sources.list.d/mongo`  
file for Ubuntu

```
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 multiverse"
```

**3**

AGTI's, DACOE Karad Page 49

Advanced Database Systems Lab Manual

Reload local package database.

Issue the following command to reload the local package database:

```
sudo apt-get update
```

**4**

Install the MongoDB packages.

You can install either the latest stable version of MongoDB or a specific version of MongoDB.



Install the latest version of MongoDB. Install a specific release of MongoDB.

To install the latest stable version, issue the following

```
sudo apt-get install -y mongodb-org
```



Optional. Although you can specify any available version of MongoDB, `apt-get` will upgrade the packages when a newer version becomes available. To prevent unintended upgrades, you can pin the package at the currently installed version:

```
echo "mongodb-org hold" | sudo dpkg --set-selections
```

```
echo "mongodb-org-database hold" | sudo dpkg --set-selections
```

```
echo "mongodb-org-server hold" | sudo dpkg --set-selections
```

```
echo "mongodb-org-shell hold" | sudo dpkg --set-selections
```

```
echo "mongodb-org-mongos hold" | sudo dpkg --set-selections
```

```
echo "mongodb-org-tools hold" | sudo dpkg --set-selections
```

For help with troubleshooting errors encountered while installing MongoDB on Ubuntu, see our [troubleshooting](#) guide.

AGTI's, DACOE Karad Page 50

Advanced Database Systems Lab Manual

Run MongoDB Community Edition

### ulimit Considerations

Most Unix-like operating systems limit the system resources that a process may use. These limits may negatively impact MongoDB operation, and should be adjusted. See [UNIX ulimit Settings](#) for the recommended settings for your platform.



**NOTE**

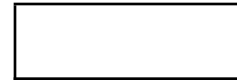


Starting in MongoDB 4.4, a startup error is generated if the ulimit value for number of open files

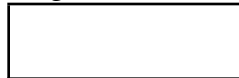


is under 64000.

### Directories



If you installed via the package manager, the data directory /var/lib/mongodb and the log directory /var/log/mongodb are created during the installation.



By default, MongoDB runs using the mongodb user account. If you change the user that runs the



MongoDB process, you **must** also modify the permission to the data and log directories to give this user access to these directories.

### Configuration File

The official MongoDB package includes a configuration file (/etc/mongod.conf). These settings (such as the data directory and log directory specifications) take effect upon startup. That is, if you change the configuration file while the MongoDB instance is running, you must restart the instance for the changes to take effect.

Procedure

Follow these steps to run MongoDB Community Edition on your system. These instructions assume that you are using the official mongodb-org package -- not the



unofficial mongodb package provided by Ubuntu -- and are using the default settings.



**Init System:** To run and manage your process, you will be using your

```
mong
od
```

operating system's built-in [init system](#). Recent versions of Linux tend to



use **systemd** (which uses the [systemctl](#) command), while older versions of Linux tend



to use **System V init** (which uses the [service](#) command).

If you are unsure which init system your platform uses, run the following command:

```
ps --no-headers -o comm 1
```

Then select the appropriate tab below based on the result:

- **systemd** - select the **systemd (systemctl)** tab below.



- **System V Init (service)** tab below.

systemd (systemctl)System V Init (service)

***Start MongoDB.***

You can start the process by issuing the following command:

```
mong  
od
```

```
sudo systemctl start mongod
```

```
mong  
od
```

If you receive an error similar to the following when starting :

```
Failed to start mongod.service: Unit mongod.service not  
found.
```

Run the following command first:

```
sudo systemctl daemon-reload
```

Then run the start command above again.

**2**

***Verify that MongoDB has started successfully.***

```
sudo systemctl status mongod
```

You can optionally ensure that MongoDB will start following a system reboot by issuing the following command:

```
sudo systemctl enable mongod
```

**3**

**Stop MongoDB.**

As needed, you can stop the process by issuing the following command:

```
mong  
od
```

sudo systemctl stop mongod

**4**

**Restart MongoDB.**

You can restart the process by issuing the following command:

```
mong  
od
```

sudo systemctl restart mongod

You can follow the state of the process for errors or important messages by watching

```
oddb/mongod.log file.
```

**5**

```
he
```

**Begin using MongoDB.**

Start a session on the same host machine as the . You can

```
mongo  
sh
```

```
mong  
od
```

```
mongo  
sh
```

```
mong  
od
```

run without any command-line options to connect to a that is running on your localhost with default port 27017.

mongosh

AGTI's, DACOE Karad Page 53

Advanced Database Systems Lab Manual

```
mongo
sh
```

For more information on connecting using , such as to connect to a instance running on a different host and/or port, see the [mongosh](#)

```
mong
od
```

[documentation](#).

To help you start using MongoDB, MongoDB provides [Getting Started Guides](#) in various driver editions. For the driver documentation, see [Start Developing with MongoDB](#).

Uninstall MongoDB Community Edition

To completely remove MongoDB from a system, you must remove the MongoDB applications themselves, the configuration files, and any directories containing data and logs. The following section guides you through the necessary steps.

### **WARNING**

This process will *completely* remove MongoDB, its configuration, and *all* databases.

This process is not reversible, so ensure that all of your configuration and data is

backed up before proceeding.

**1**

Stop MongoDB.

Stop the process by issuing the following command:

```
mong  
od
```

```
sudo service mongod stop
```

**2**

Remove Packages.

Remove any MongoDB packages that you had previously installed.

```
sudo apt-get purge mongodb-org*
```

**3**

Remove Data Directories.

Remove MongoDB databases and log files.

```
sudo rm -r /var/log/mongodb
```

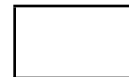
```
sudo rm -r /var/lib/mongodb
```

## Additional Information

### Localhost Binding by Default

By default, MongoDB launches with set to 127.0.0.1, which binds to the

```
bind  
Ip
```



[

localhost network interface. This means that the mongod can only accept connections from clients that are running on the same machine. Remote clients will not be able to connect to the mongod, and the mongod will not be able to initialize a replica



set unless this value is set to a valid network

interface. This value can be configured either:

- in the MongoDB configuration file with , or

bind  
Ip

- via the command-line argument

--bind  
\_ip

**Problem statement:** Install  
Apache Cassandra **Link for  
references:**

<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu>

[https://cassandra.apache.org/doc/latest/cassandra/getting\\_started/installing.html](https://cassandra.apache.org/doc/latest/cassandra/getting_started/installing.html) **Conclusion:**

We have studied how to install MongoDB & Apache Cassandra on various platform.

## Experiment No.6

**Title:** Exploring MongoDB basics, Identify the schema design and data modeling techniques in



MongoDB.

**Aim:** Implement basics of MongoDB.

**Theory:**

MongoDB is a NoSQL database. Under the *NoSQL* umbrella we put all those databases that do not use the SQL language for querying the data.

**Key characteristics of MongoDB**

MongoDB is a very JavaScript-friendly database. It exposes a JavaScript API we can use to create databases and collections of objects (called *documents*). It's *schemaless*, which means you don't need to pre-define a structure for the data before storing it.

In MongoDB you can store any object without having to worry about the particular fields that compose this object and how to store them. You tell MongoDB to store that object. Data is stored in a format similar to JSON, but enhanced to allow storing more than just basic data types.

**Installation**

Let's go ahead and install MongoDB. You could use one of the many cloud providers that offer access to a MongoDB instance, but for the sake of learning, we'll install it ourselves.

Open the terminal and run:



The instructions were not too long or complicated, assuming you know how to use the terminal and [how to install Homebrew](#).

The installation tells us this:

```
To have launchd start mongod now and restart at login:  
brew services start mongodb-community  
Or, if you don't want/need a background service you can just run:  
mongod --config /usr/local/etc/mongod.conf
```

You can choose to either launch MongoDB once and have it running forever as a background service in your computer (the thing I prefer), or you can run it just when you need it, by running the latter command.

The default configuration for MongoDB is this:

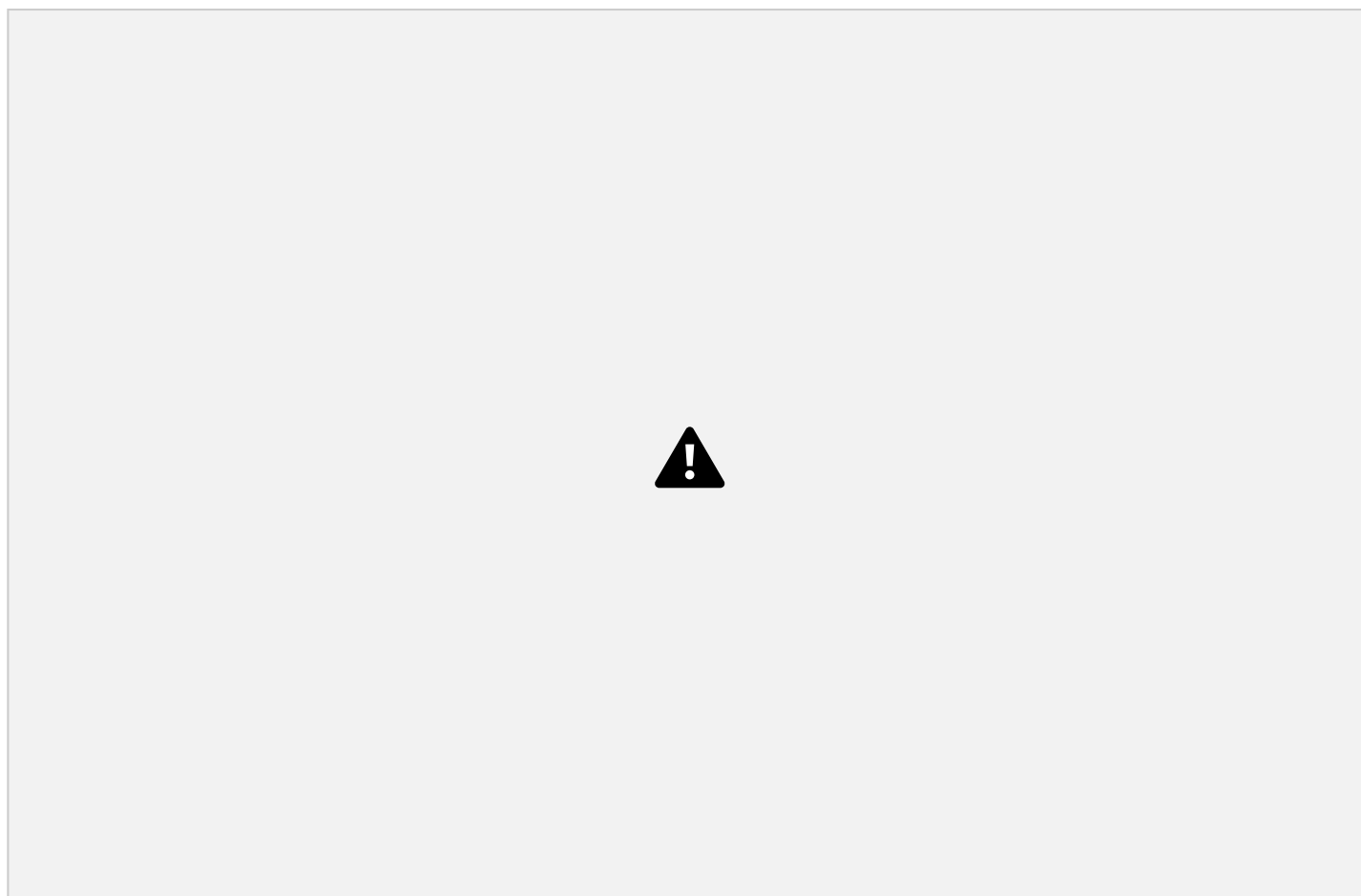
```
systemLog:  
  destination: file  
  path: /usr/local/var/log/mongodb/mongo.log  
  logAppend: true  
storage:  
  dbPath: /usr/local/var/mongodb  
net:  
  bindIp: 127.0.0.1
```

Logs are stored in `/usr/local/var/log/mongodb/mongo.log` and the database is stored in `/usr/local/var/mongodb`.

By default there is no access control, anyone can read and write to the database.

## The Mongo Shell

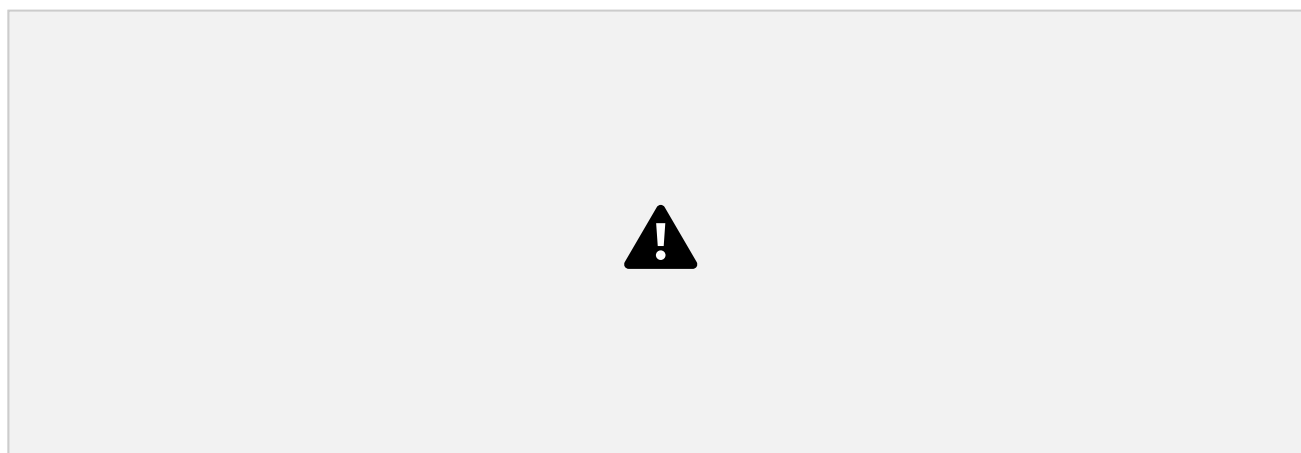
The best way to experiment with MongoDB and starting to interact with it is by running the `mongo` program, which starts the MongoDB shell.



You can now enter any command that Mongo understands.

### **Create a database**

When you start, Mongo creates a database called `test`. Run `db` in the shell to tell you the name of the active database

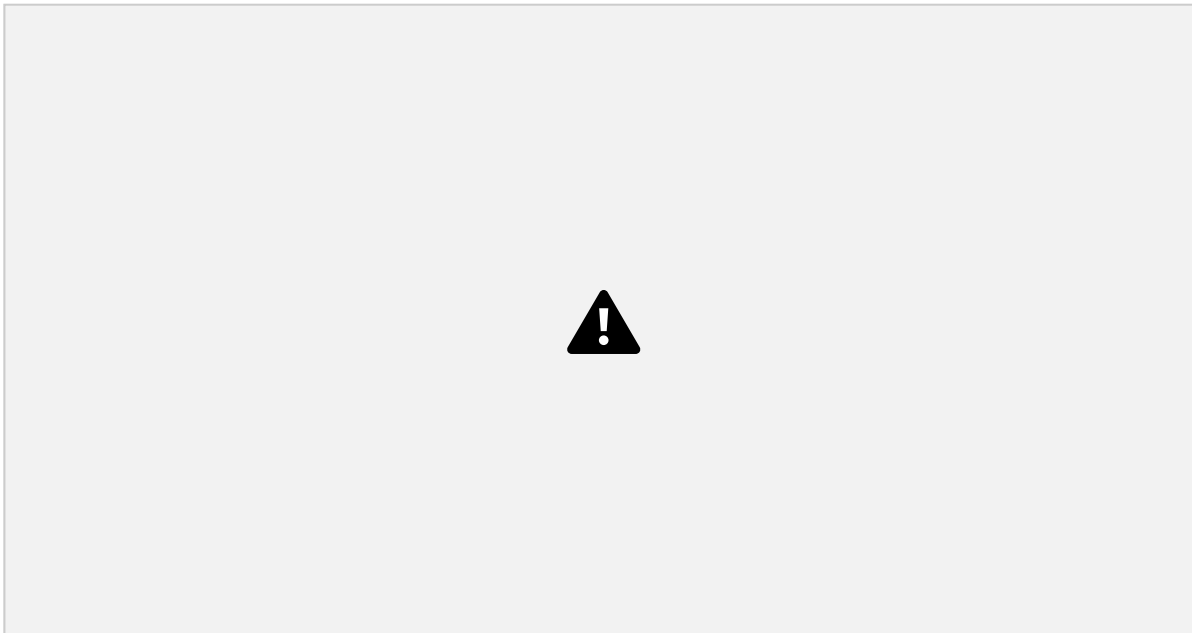


To change the database, just write `use newname` and the `newname` database will be instantly created and

the shell switches to using that.



Use `show databases` to list the available databases:



AGTI's,

DACOE Karad Page 59

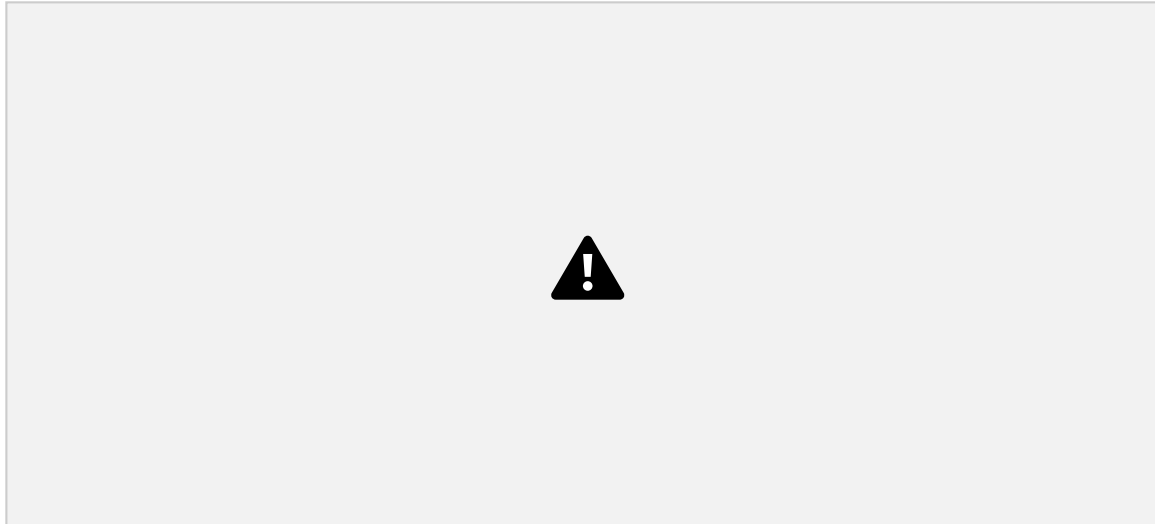
**Advanced Database Systems Lab Manual**

As you can see, the `something` database is not listed, just because there is no collection yet in it. Let's create one.

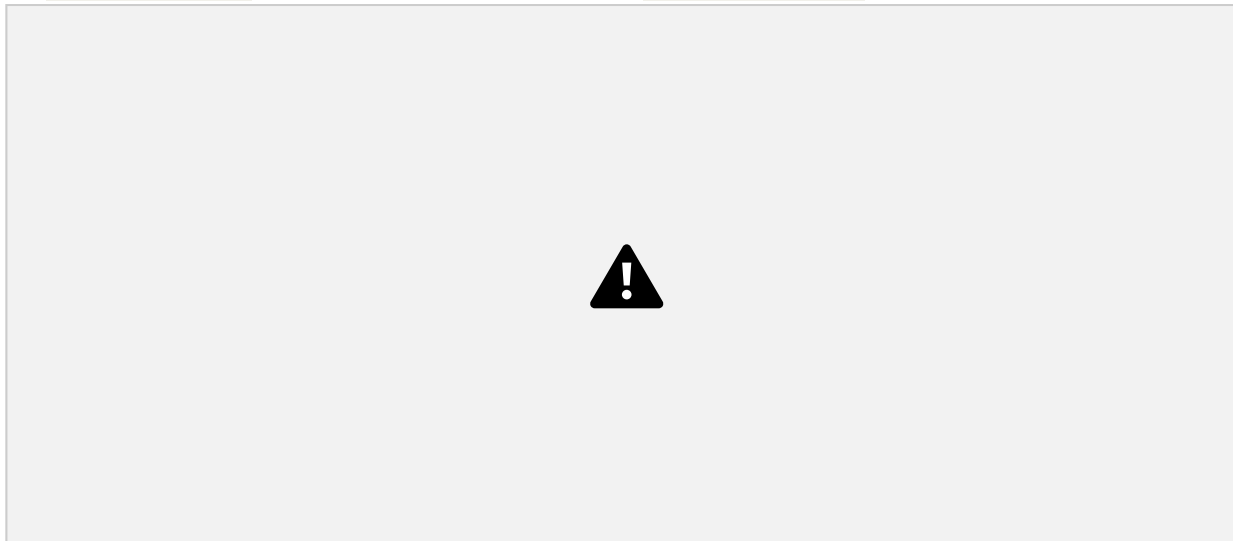
## **Collections**

In MongoDB, a **collection** is the equivalent of a SQL database table.

You create a collection on the current database by using the `db.createCollection()` command. The first argument is the database name, and you can pass an options object as a second parameter.



Once you do so, `show databases` will list the new database, and `show collections` will list the collection.



AGTI's,

You can also create a new collection by using it as a property of the `db` object, and calling `insert()` to add an object to the collection:



### Listing objects in a collection

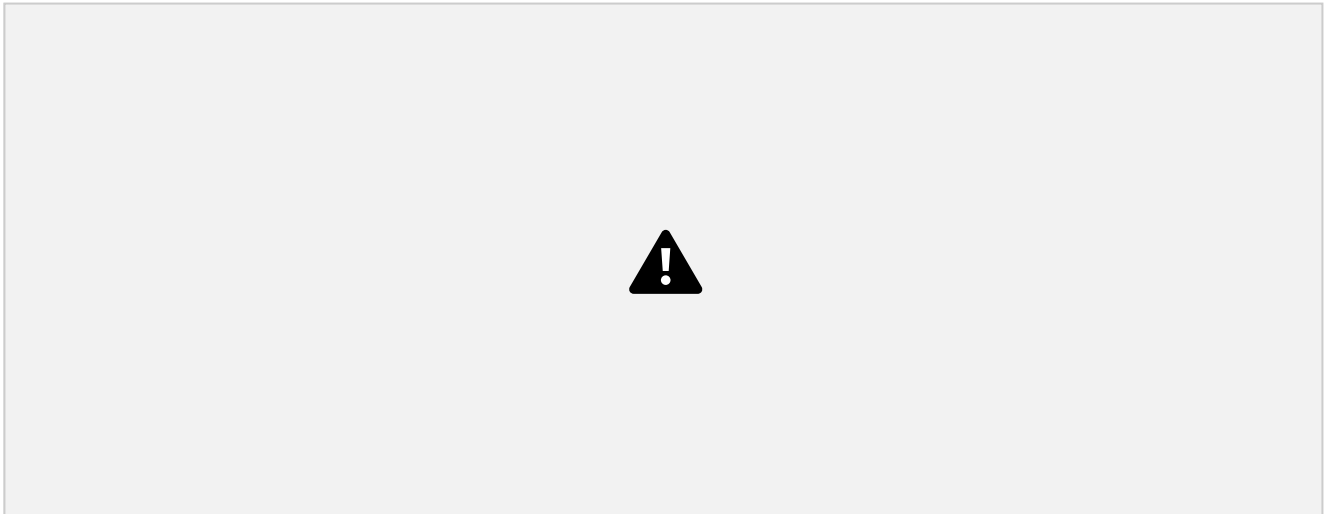
To show the objects added to a collection, use the `find()` method:



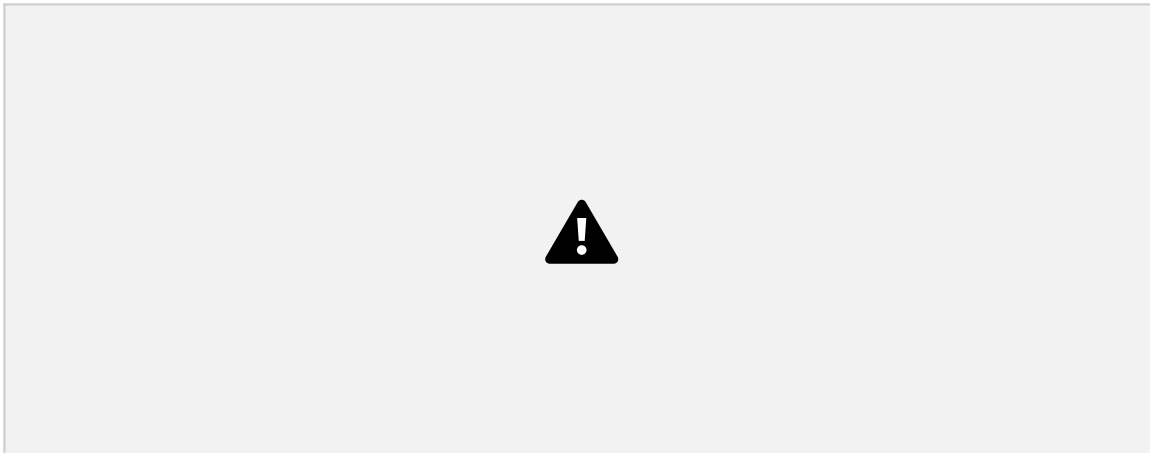
As you can see there is an additional `_id` property for the record we added. That's automatically generated for us by MongoDB.

Now, add more dogs:

Calling `db.dogs.find()` will give us all the entries, while we can pass a parameter to filter and retrieve a specific entry, for example with `db.dogs.find({name: 'Roger'})`:



The `find()` method returns a cursor you need to iterate on. There is another method which is handy when you know you'll only get one record, which is `findOne()`, and it's used in the same way. If multiple records match a query, it will just return the first one.



AGTI's, DACOE

**Updating records:** To update a record you can use the `update()` method on a collection:



### Removing records

You can remove a record calling the `remove()` method on a collection, passing an object to help identify it:



To remove all the entries from a collection, pass an empty object:

**Problem Statement:** Similarly implement basics of Apache Cassedra

**Conclusion:** We have studied and implement basics of MongoDB & Apache Cassedra.

## Experiment No.7

**Title:** Installation of CouchDB and Run CouchDB query with Mongo



**Aim:** To Install & Demonstrate CouchDB and creating database in it.

**Theory:**

**What is CouchDB?**

CouchDB is an open source database developed by Apache software foundation. The focus is on the ease of use, embracing the web. It is a NoSQL document store database.

It uses JSON, to store data (documents), java script as its query language to transform the documents, http protocol for api to access the documents, query the indices with the web browser. It is a multi master application released in 2005 and it became an apache project in 2008.

**Why CouchDB?**

- CouchDB have an HTTP-based REST API, which helps to communicate with the database easily. And the simple structure of HTTP resources and methods (GET, PUT, DELETE) are easy to understand and use.
- As we store data in the flexible document-based structure, there is no need to worry about the structure of the data.
- Users are provided with powerful data mapping, which allows querying, combining, and filtering the information.
- CouchDB provides easy-to-use replication, using which you can copy, share, and synchronize the data between databases and machines.

**Data Model:**

- Database is the outermost data structure/container in CouchDB.
- Each database is a collection of independent documents.
- Each document maintains its own data and self-contained schema.
- Document metadata contains revision information, which makes it possible to merge the differences occurred while the databases were disconnected.
- CouchDB implements multi version concurrency control, to avoid the need to lock the database field during writes.

**Features of CouchDB:** Reduce the Content Document Storage

CouchDB is a document storage NoSQL database. It provides the facility of storing documents with unique names, and it also provides an API called RESTful HTTP API for reading and updating (add, edit, delete) database documents.

In CouchDB, documents are the primary unit of data and they also include metadata. Document fields are uniquely named and contain values of varying types (text, number, Boolean, lists, etc.), and there is no set limit to text size or element count.

Document updates (add, edit, delete) follow Atomicity, i.e., they will be saved completely or not saved at

all. The database will not have any partially saved or edited documents.

#### Json Document Structure

```
{  
  "field" : "value",  
  "field" : "value",  
  "field" : "value",  
}
```

#### **ACID Properties:**

CouchDB contains ACID properties as one of its features.

**Consistency** – When the data in CouchDB was once committed, then this data will not be modified or overwritten. Thus, CouchDB ensures that the database file will always be in a consistent state.

A multi-Version Concurrency Control (MVCC) model is used by CouchDB reads, because of which the client will see a consistent snapshot of the database from the beginning to the end of the read operation.

Whenever a documents is updated, CouchDB flushes the data into the disk, and the updated database header is written in two consecutive and identical chunks to make up the first 4k of the file, and then synchronously flushed to disk. Partial updates during the flush will be discarded.

If the failure occurred while committing the header, a surviving copy of the previous identical headers will remain, ensuring coherency of all previously committed data. Except the header area, consistency checks or fix-ups after a crash or a power failure are never necessary.

#### **Compaction:**

Whenever the space in the database file got wasted above certain extent, all the active data will be copied (cloned) to a new file. When the copying process is entirely done, then the old file will be discarded. All this is done by compaction process. The database remains online during the compaction and all updates and reads are allowed to complete successfully.

#### **Views:**

Data in CouchDB is stored in semi-structured documents that are flexible with individual implicit structures, but it is a simple document model for data storage and sharing. If we want see our data in many different ways, we need a way to filter, organize and report on data that hasn't been decomposed into tables.

To solve this problem, CouchDB provides a view model. Views are the method of aggregating and reporting on the documents in a database, and are built on-demand to aggregate, join and report on database documents. Because views are built dynamically and don't affect the underlying document, you can have as many different view representations of the same data as you like.

#### **History**

- CouchDB was written in Erlang programming language.
- It was started by Damien Katz in 2005.
- CouchDB became an Apache project in 2008.

**Official Online Resources** — <http://couchdb.apache.org> and [www.couchbase.com](http://www.couchbase.com). Most of the authors are part of Couchbase, Inc.

**History** — Work started in 2005 and it was incubated into Apache in 2008.

**Technologies and Language** — Implemented in Erlang with some C and a JavaScript execution environment.

**Access Methods** — Upholds REST above every other mechanism. Use standard web tools and clients to access the database, the same way as you access web resources.

**Open-Source License** — Apache License version 2. **Who Uses It** — Apple, BBC, Canonical, Cern, and more at [http://wiki.apache.org/couchdb/CouchDB\\_in\\_the\\_wild](http://wiki.apache.org/couchdb/CouchDB_in_the_wild).

**Implementation:** Create database in CouchDB and perform basic operations on it.

**Conclusion:** We have studied and implement the database in CouchDB.

## Experiment No.8

**Title:** Study of big data systems

**Aim:** To study and deal with big data systems.

### Theory:

#### What is Big Data?

Big data means really a big data, it is a collection of large datasets that cannot be processed using traditional computing techniques. Big data is not merely a data, rather it has become a complete subject, which involves various tools, techniques and frameworks.

#### What Comes Under Big Data?

Big data involves the data produced by different devices and applications. Given below are some of the fields that come under the umbrella of Big Data.

**Black Box Data:** It is a component of helicopter, airplanes, and jets, etc. It captures voices of the flight crew, recordings of microphones and earphones, and the performance information of the aircraft. **Social**

**Media Data:** Social media such as Facebook and Twitter hold information and the views posted by millions of people across the globe.

**Stock Exchange Data:** The stock exchange data holds information about the 'buy' and 'sell' decisions made on a share of different companies made by the customers.

**Power Grid Data:** The power grid data holds information consumed by a particular node with respect to a base station.

**Transport Data:** Transport data includes model, capacity, distance and availability of a vehicle. **Search**

**Engine Data:** Search engines retrieve lots of data from different databases. Thus Big Data includes huge volume, high velocity, and extensible variety of data. The data in it will be of three types.

**Structured data:** Relational data.

**Semi Structured data:** XML data, Json data.

**Unstructured data:** Word, PDF, Text, Media Logs.

**Benefits of Big Data** Big data is really critical to our life and its emerging as one of the most important technologies in modern world. Follow are just few benefits which are very much known to all of us:

Using the information kept in the social network like Facebook, the marketing agencies are learning about the response for their campaigns, promotions, and other advertising mediums.

Using the information in the social media like preferences and product perception of their consumers, product companies and retail organizations are planning their production.

Using the data regarding the previous medical history of patients, hospitals are providing better and quick service.

#### Big Data Technologies:

Big data technologies are important in providing more accurate analysis, which may lead to more concrete decision-making resulting in greater operational efficiencies, cost reductions, and reduced risks for the business. To harness the power of big data, you would require an infrastructure that can manage and process huge volumes of structured and unstructured data in real-time and can protect data privacy and security. There are various technologies in the market from different vendors including Amazon, IBM, Microsoft, etc., to handle big data. While looking into the technologies that handle big data, we examine the following two classes of technology:

**Operational Big Data:**

It includes systems like MongoDB that provide operational capabilities for real-time, interactive workloads where data is primarily captured and stored. NoSQL Big Data systems are designed to take advantage of new cloud computing architectures that have emerged over the past decade to allow massive computations to be run inexpensively and efficiently. This makes operational big data workloads much easier to manage, cheaper, and faster to implement. Some NoSQL systems can provide insights into patterns and trends based on real-time data with minimal coding and without the need for data scientists and additional infrastructure.

**Analytical Big Data:**

This includes systems like Massively Parallel Processing MPP database systems and MapReduce that provide analytical capabilities for retrospective and complex analysis that may touch most or all of the data. MapReduce provides a new method of analyzing data that is complementary to the capabilities provided by SQL, and a system based on MapReduce that can be scaled up from single servers to thousands of high and low end machines. These two classes of technology are complementary and frequently deployed together.

**Question:** Explain Traditional Data Mining Life Cycle Vs Big Data life cycle.

**Implementation:** Data collection from any social site using any big data tool.

**Conclusion:** We have studied big data and understand how to deal with big data system.

## Experiment No.9

**Title:** Implementation of parallel joins, sorting and aggregates

**Aim:** To implement parallel joins, sorting and aggregates

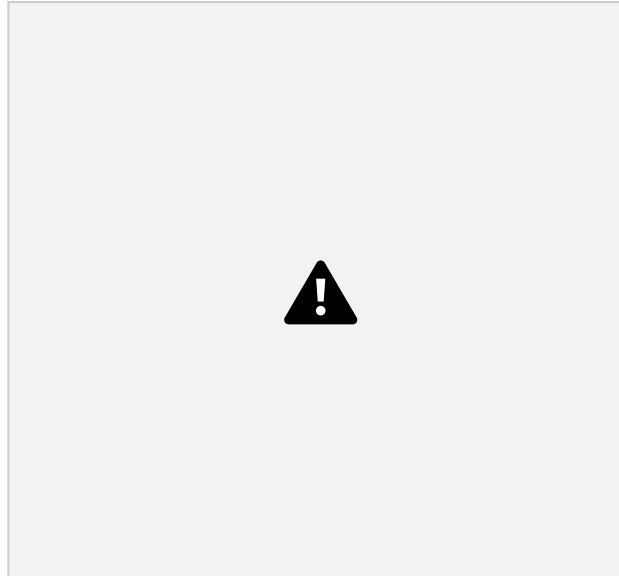
**Theory:** Parallel hash join makes use of the merge and split operators and they are parallelized.

Suppose,  $A$  and  $B$ , two relation to be joined on the *age* attribute. They are initially distributed across several disks in some way.

The basic idea for joining  $A$  and  $B$  in parallel is to decompose the join into a collection of  $k$  smaller joins. We can decompose the join by partitioning both  $A$  and  $B$  into a collection of  $k$  logical buckets or partitions. By using the same partitioning function for both  $A$  and  $B$ , we ensure that the union of the  $k$  smaller joins computes the join of  $A$  and  $B$ .

The partitioning step can itself be done in parallel at these processors. At each processor, all local tuples are retrieved and hashed into one of  $k$  partitions, with the same hash function used at all sites. Alternatively, we can partition  $A$  and  $B$  by dividing the range of the join attribute *age* into  $k$  disjoint sub ranges and placing  $A$  and  $B$  tuples into partitions according to the sub range to which their *age* values belong. Having decided on a partitioning strategy, assign each partition to a processor and carry out a local join, using any join algorithm at each processor. The number of partitions  $k$  is chosen to be equal to the number of processors  $n$  that are available for carrying out the join, and during partitioning, each processor sends tuples in the  $i$ th partition to processor  $i$ .

After partitioning, each processor joins the *A* and *B* tuples assigned to it. Each join process executes sequential join code, and receives input *A* and *B* tuples from several processors; a merge operator merges all incoming *A* tuples, and another merge operator merges all incoming *B* tuples. If range partitioning is used, the algorithm outlined above leads to a parallel version of a sort merge join, with the advantage that the output is available in sorted order. If hash partitioning is used, we obtain a parallel version of a hash join.



**Data Flow Diagram for Joining the Tables**

**Algorithm:-**

1. Start
2. Insert the data into tables.
3. Create the buckets using hash.
4. Carry out join of appropriate buckets and display join.
5. Stop.

**Implementation:**

**Program in Java:-**

```
Import java.io.*;
import java.util.*;
class Stu
{
    int rno;
    String name;
    Stu(int n,String nm)
    {
        rno=n;
        name=new String(nm);
    }
}
```

```

    void display()
    {
        System.out.println("Name:"+name+"\nRno:"+rno)
    ; }
}

```

```

class Marks
{
    int rn;
    int mark;
    Marks(int n,int mrk)
    {
        rn=n;
        mark=mrk;
    }
    void display()
    {
        System.out.println("Mark:"+mark+"\nRno:"+rn);
    }
}

```

```

class HashJoin
{

    public static void main(String []args)throws Exception
    {

```

AGTI's, DACOE Karad Page 70

**Advanced Database Systems Lab Manual**

```

        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        LinkedList ls0=new LinkedList();
        LinkedList ls1=new LinkedList();
        LinkedList ls2=new LinkedList();
        LinkedList ls4=new LinkedList();
        while(true)
        {
            System.out.print("1.Enter Student data\n2..Enter Student Marks\n3.Display
Bucktes\n4.Display join\n5.Exit\n\tOption:");
            int opt=Integer.parseInt(br.readLine());
            switch(opt)
            {
                case 1:

                    System.out.print("\nInput rno:-");
                    int r=Integer.parseInt(br.readLine());
                    System.out.print("\nInput Name:-");
                    String nm=br.readLine();
                    Stu s=new Stu(r,nm);

```



```

        int h=r%2; //hash
        if(h==0)
            ls0.add(s);
        else
            ls1.add(s);
        break;
case 2:
    System.out.print("\nInput rno:-");
    int rn=Integer.parseInt(br.readLine());
    System.out.print("\nInput Mark:-");
    int mr=Integer.parseInt(br.readLine());
    Marks m=new Marks(rn,mr);
int hv=rn%2; //hash value
    if(hv==0)
        ls2.add(m);
    else
        ls4.add(m);
    break;
case 3:
    System.out.println("Bucket A0:");
    for(int i=0;i<ls0.size();i++)
    {
        Stu ss1=(Stu)ls0.get(i);
        System.out.println(ss1.rno+" "+ss1.name);
    }
    System.out.println("Bucket A1:");
    for(int i=0;i<ls1.size();i++)
    {

```

```

        Stu ss2=(Stu)ls1.get(i);
        System.out.println(ss2.rno+" "+ss2.name);
    }
    System.out.println("Bucket B0:");
    for(int i=0;i<ls2.size();i++)
    {
        Marks ms1=(Marks)ls2.get(i);
        System.out.println(ms1.rn+" "+ms1.mark);
    }
    System.out.println("Bucket B1:");
    for(int i=0;i<ls4.size();i++)
    {
        Marks ms2=(Marks)ls4.get(i);
        System.out.println(ms2.rn+" "+ms2.mark);
    }
    break;
case 4:

```

```

System.out.println("Join of A0 & B0");
System.out.println("rno name mrk");
for(int i=0;i<ls0.size();i++)
{
    Stu s1=(Stu)ls0.get(i);
    for(int j=0;j<ls2.size();j++)
    {
        Marks mm=(Marks)ls2.get(j);
        if(s1.rno==mm.rn)
            System.out.println(s1.rno+"
+s1.name+" "+mm.mark);
    }
}
System.out.println("Join of A1 & B1");
System.out.println("rno name mrk");
for(int k=0;k<ls1.size();k++)
{
    Stu s2=(Stu)ls1.get(k);
    for(int l=0;l<ls4.size();l++)
    {
        Marks m2=(Marks)ls4.get(l);
        if(s2.rno==m2.rn)
            System.out.println(s2.rno+" "+s2.name+"
+m2.mark);
    }
}
case 5:
    System.exit(0);
}
}}}

```

**Output:-**

1.Enter Student data  
2..Enter Student  
Marks 3.Display  
Bucktes 4.Display join  
5.Exit  
Option:1

Input rno:-1  
Input Name:-mukesh  
1.Enter Student data  
2..Enter Student  
Marks 3.Display  
Bucktes 4.Display join  
5.Exit

Option:1  
Input rno:-2  
Input Name:-mahesh  
1.Enter Student data  
2..Enter Student  
Marks 3.Display  
Bucktes 4.Display join  
5.Exit

Option:2  
Input rno:-1  
Input Mark:-99

1.Enter Student data  
2..Enter Student  
Marks 3.Display  
Bucktes 4.Display join  
5.Exit

Option:2  
Input rno:-3  
Input Mark:-88  
1.Enter Student data  
2..Enter Student  
Marks 3.Display  
Bucktes 4.Display join  
5.Exit

Option:3  
Bucket A0:  
2 mahesh  
Bucket A1:  
1 mukesh]  
Bucket B0:  
Bucket B1:

AGTI's, DACOE Karad Page 73

**Advanced Database Systems Lab Manual**

1 99  
3 88

1.Enter Student data  
2..Enter Student Marks  
3.Display Bucktes  
4.Display join  
5.Exit

Option:4

Join of A0 & B0  
rno name mrk  
Join of A1 & B1

rno name mrk  
1 mukesh 99

## **IMPLEMENTATION in SQL:**

### **//Table Creation**

1. create table student (grn int,rn int,name varchar2(250),pname varchar2(250),mobile int,pmobile int,dob varchar2(20),eid varchar2(150),elective varchar2(150),address varchar2(1250),pincode int,category varchar2(50));
2. create table result(rn int primary key, ut1 int,ut2 int,ut3 int,attendance int);

### **// Data Loading**

1. Click:- Loading Data using utilities>Data Load/Unload>Load>Load Spreadsheet

Data> 2. Load To:

Click Existing Table

Load From:

Copy and Paste

3. Type Table name i.e. student, result
4. Copy the contents from excel sheet
5. Paste the contents in Data Block
6. Click on load data

AGTI's, DACOE Karad Page 74

**Advanced Database Systems Lab Manual**

### **//Normal Join Query**

select \* from student natural join result // 0.12 Seconds

### **//Parallel Join Query**

SELECT /\*+parallel (10) \*/ \* FROM student natural join result // 0.03 Seconds

### **CONCLUSION:**

We implement parallel joins, sorting and aggregates and observed that parallel execution of join operation gives faster output than normal execution of join operation.

## **Experiment No.10**

**Title:** Demonstrate all OLAP operations and cube operator in OLAP.

**Aim:** To implement all OLAP operations and cube operator in OLAP.

**Theory:**

- **Multidimensional Data Model**

Multidimensional data model stores data in the form of data cube. Mostly, data warehousing supports two or three-dimensional cubes. A data cube allows data to be viewed in multiple dimensions. Dimensions are entities with respect to which an organization wants to keep records. For example in store sales record, dimensions allow the store to keep track of things like monthly sales of items and the branches and locations. A multidimensional database helps to provide data-related answers to complex business queries quickly and accurately. Data warehouses and Online Analytical Processing (OLAP) tools are based on a multidimensional data model. OLAP in data warehousing enables users to view data from different angles and dimensions.

• **Online Analytical Processing:**

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information.

**OLAP operations in multidimensional data.**

- Roll-up
- Drill-down
- Slice and dice
- Pivot (rotate)

**Roll-up:**

Roll-up performs aggregation on a data cube in any of the following ways –

- By climbing up a concept hierarchy for a dimension
- By dimension reduction

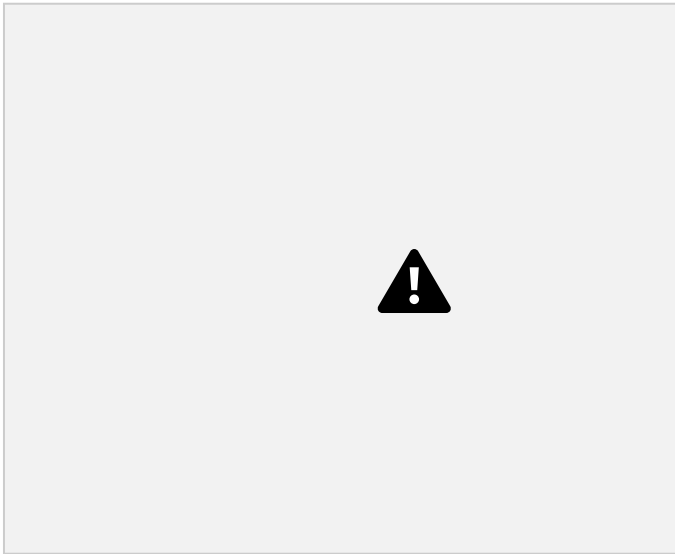
The following diagram illustrates how roll-up works.

- Roll-up is performed by climbing up a concept hierarchy for the dimension location. •

Initially the concept hierarchy was "street < city < province < country".

- On rolling up, the data is aggregated by ascending the location hierarchy from the level of city to the level of country.
- The data is grouped into cities rather than countries.
- When roll-up is performed, one or more dimensions from the data cube are removed.

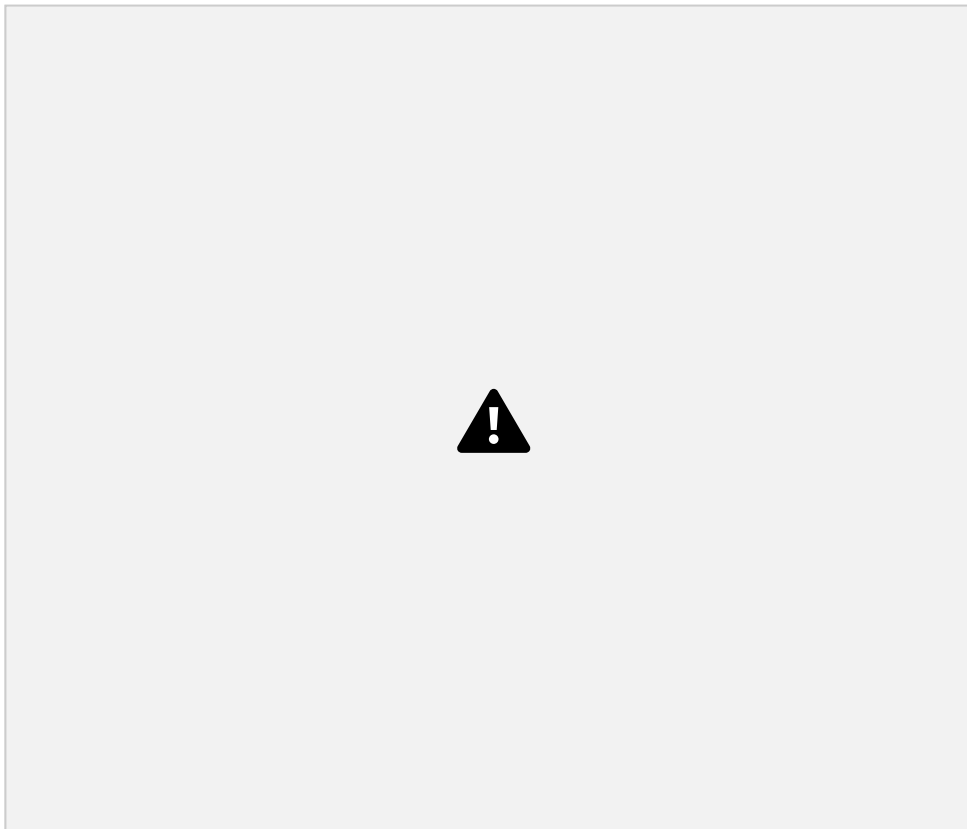
**Drill-down:**



Drill-down is the reverse operation of roll-up. It is performed by either of the following ways –

- By stepping down a concept hierarchy for a dimension
- By introducing a new dimension.

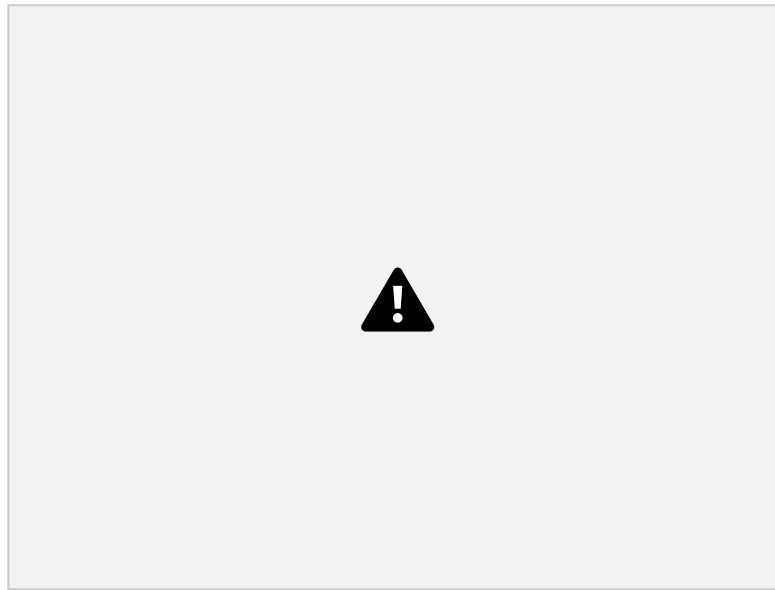
The following diagram illustrates how drill-down works –



- Drill-down is performed by stepping down a concept hierarchy for the dimension time. • Initially the concept hierarchy was "day < month < quarter < year."
- On drilling down, the time dimension is descended from the level of quarter to the level of month. • When drill-down is performed, one or more dimensions from the data cube are added. • It navigates the data from less detailed data to highly detailed data.

**Slice:**

The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Consider the following diagram that shows how slice works.

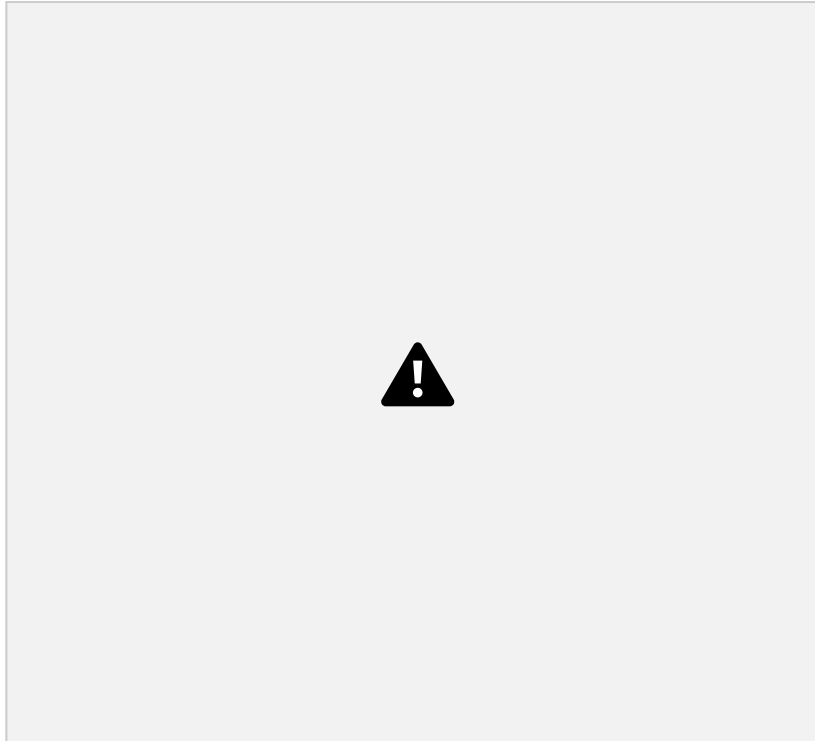


- Here Slice is performed for the dimension "time" using the criterion time = "Q1". • It will form a new sub-cube by selecting one or more dimensions.

**Dice:**

Dice selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram that shows the dice operation.





The dice operation on the cube based on the following selection criteria involves three dimensions.

- (location = "Toronto" or "Vancouver")
- (time = "Q1" or "Q2")
- (item = " Mobile" or "Modem")

### **IMPLEMENTATION:**

#### **//Creating Clothes table**

```
create table clothes (item varchar2(40),color varchar2(20),size1 varchar2(10),num int);
```

#### **//Inserting Data**

```
insert into clothes values('shirt','white','small',20);
```

```
insert into clothes values('shirt','blue','medium',10);
```

```
insert into clothes values('pant','black','large',25);
```

```
insert into clothes values('dress','white','medium',30);
```

### //Rollup Query

```
insert into clothes values('pant','blue','large',20);
```

Advanced Database Systems Lab Manual

```
insert into clothes values('dress','pink','small',15);
```

```
select item,color,size1,sum(num) from clothes group by rollup(item,color,size1);
```

### //Cube Query

```
select item,color,size1,sum(num) from clothes group by cube(item,color,size1);
```

### CONCLUSION:

CUBE generates a result set that shows aggregates for all combinations of values in the selected columns. ROLLUP generates a result set that shows aggregates for a hierarchy of values in the selected columns.

<b>Laboratory Outcomes (LOs)</b> Student will be able to	
1	Install using guidelines and operate platform confidently
2	Apply SQL Function, Procedure, Trigger, cursor and views for managing the complex SQL scripts
3	Recognize conceptual design techniques and create database schema for a given application scenario.
4	Select appropriate database and construct solution to real world problems of storing large data like Big data system
5	Apply various types of skills needed by database administrators