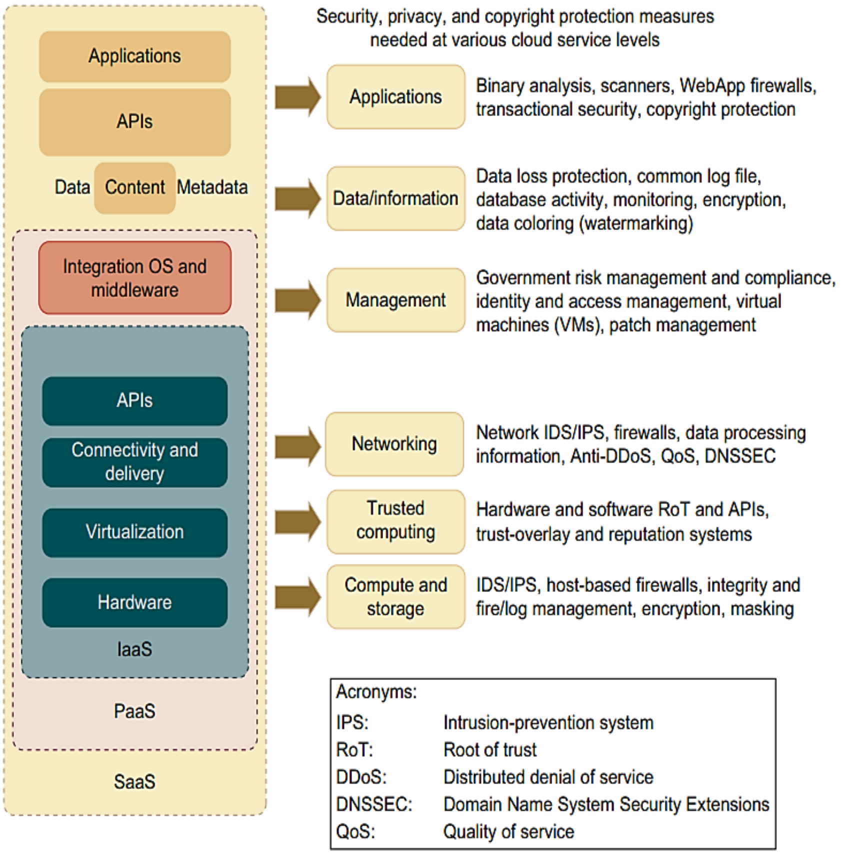
USN		CMRIT CMR Institute of Technology, Bengaluru. Accredited with A++ Grade by NAAC

Internal Assessment Test 2 - May 2025
Scheme of Evlauation

Sub:	**Cloud Computing and Security**					Sub Code:	**BIS613D**	Branch:	**ISE**
Date:	**24/05/2025**	Duration:	**90 min**	Max Marks:	**50**	Sem/Sec:	**VI / A, B & C**		**OBE**

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	**Describe six architectural design challenges of cloud.** Scheme: Explanation of each design challenge **2 marks** Solutions: Challenge 1—Service Availability and Data Lock-in Problem The management of a cloud service by a single company is often the source of single points of failure. To achieve HA, one can consider using multiple cloud providers. Even if a company has multiple data centers located in different geographic regions, it may have common software infrastructure and accounting systems. Therefore, using multiple cloud providers may provide more protection from failures. Another availability obstacle is distributed *denial of service* (DDoS) attacks. Criminals threaten to cut off the incomes of SaaS providers by making their services unavailable. Some utility computing services offer SaaS providers the opportunity to defend against DDoS attacks by using quick scale-ups. Challenge 2—Data Privacy and Security Concerns Current cloud offerings are essentially public (rather than private) networks, exposing the system to more attacks. Many obstacles can be overcome immediately with well-understood technologies such as encrypted storage, virtual LANs, and network middleboxes (e.g., firewalls, packet filters). For example, you could encrypt your data before placing it in a cloud. Many nations have laws requiring SaaS providers to keep customer data and copyrighted material within national boundaries. Traditional network attacks include buffer overflows, DoS attacks, spyware, malware, rootkits, Trojan horses, and worms. In a cloud environment, newer attacks may result from hypervisor mal- Challenge 3—Unpredictable Performance and Bottlenecks Multiple VMs can share CPUs and main memory in cloud computing, but I/O sharing is problematic. For example, to run 75 EC2 instances with the STREAM benchmark requires a mean bandwidth of 1,355 MB/second. However, for each of the 75 EC2 instances to write 1 GB files to the local disk requires a mean disk write bandwidth of only 55 MB/second. This demonstrates the problem of I/O interference between VMs. One solution is to improve I/O architectures and operating systems to efficiently virtualize interrupts and I/O channels. Challenge 4—Distributed Storage and Widespread Software Bugs The database is always growing in cloud applications. The opportunity is to create a storage system that will not only meet this growth, but also combine it with the cloud advantage of scaling arbitrarily up and down on demand. This demands the design of efficient distributed SANs. Data centers must meet programmers' expectations in terms of scalability, data durability, and HA. Data consistence checking in SAN-connected data centers is a major challenge in cloud computing. Challenge 5—Cloud Scalability, Interoperability, and Standardization The pay-as-you-go model applies to storage and network bandwidth; both are counted in terms of the number of bytes used. Computation is different depending on virtualization level. GAE automatically scales in response to load increases and decreases; users are charged by the cycles used. AWS charges by the hour for the number of VM instances used, even if the machine is idle. The opportunity here is to scale quickly up and down in response to load variation, in order to save money, but without violating SLAs. Challenge 6—Software Licensing and Reputation Sharing Many cloud computing providers originally relied on open source software because the licensing model for commercial software is not ideal for utility computing. The primary opportunity is either for open source to remain popular or simply for commercial software companies to change their licensing structure to better fit cloud computing. One can consider using both pay-for-use and bulk-use licensing schemes to widen the business coverage.	[10]	3	L2
2	**Explain the mapping of different cloud models which ensure the special security measures deployed at various cloud operating levels.**	[10]	4	L2

Scheme: Explanation of each cloud model + Diagram – **8+2 marks**
Solutions:



(a) Cloud service models (b) Security, privacy, and copyright protection measures

FIGURE 4.31

Cloud service models on the left (a) and corresponding security measures on the right (b); the IaaS is at the innermost level, PaaS is at the middle level, and SaaS is at the outermost level, including all hardware, software, datasets, and networking resources.

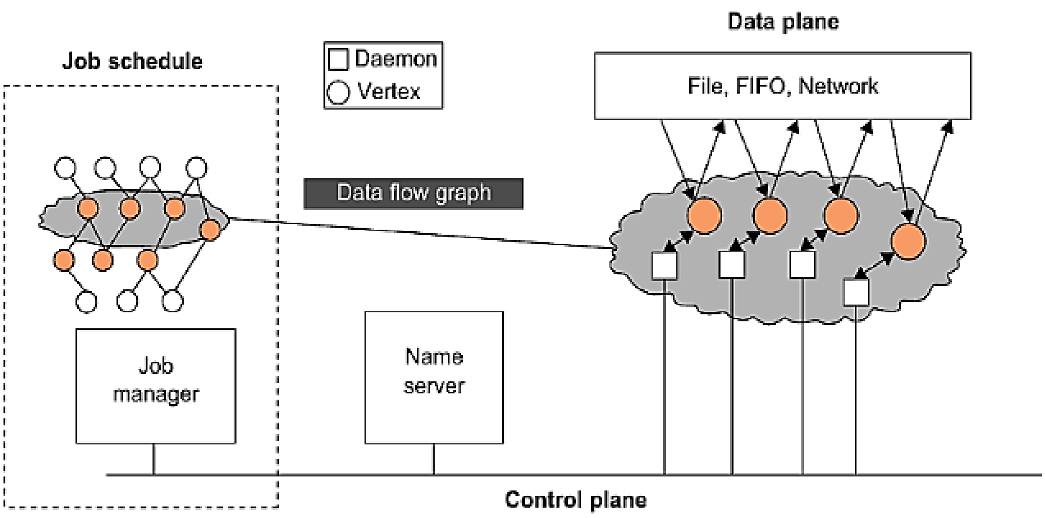
Three basic cloud security enforcements are expected. First, facility security in data centers demands on-site security year round. Biometric readers, CCTV (close-circuit TV), motion detection, and man traps are often deployed. Also, network security demands fault-tolerant external firewalls, intrusion detection systems (IDSes), and third-party vulnerability assessment. Finally, platform security demands SSL and data decryption, strict password policies, and system trust certification. Figure 4.31 shows the mapping of cloud models, where special security measures are deployed at various cloud operating levels.

Servers in the cloud can be physical machines or VMs. User interfaces are applied to request services. The provisioning tool carves out the systems from the cloud to satisfy the requested service. A security-aware cloud architecture demands security enforcement. Malware-based attacks such as network worms, viruses, and DDoS attacks exploit system vulnerabilities. These attacks compromise system functionality or provide intruders unauthorized access to critical information.

3 **Explain DryadLINQ in detail with a neat diagram.** [10] 5 L2

Scheme: Definition+ Explanation + Components + Diagram – **2+3+3+2 marks**
Solutions:

- **Dryad** is more flexible than MapReduce as the data flow of its applications is not dictated/predetermined and can be easily defined by users.
- To achieve such flexibility, a Dryad program or job is defined by a directed acyclic graph (DAG) where vertices are computation engines and edges are communication channels between vertices.
- Given a DAG, Dryad assigns the computational vertices to the underlying computation engines (cluster nodes) and controls the data flow through edges (communication between cluster nodes).
- Data partitioning, scheduling, mapping, synchronization, communication, and fault tolerance are major implementation details hidden by Dryad to facilitate its programming environment.



(a) Dryad control and data flow

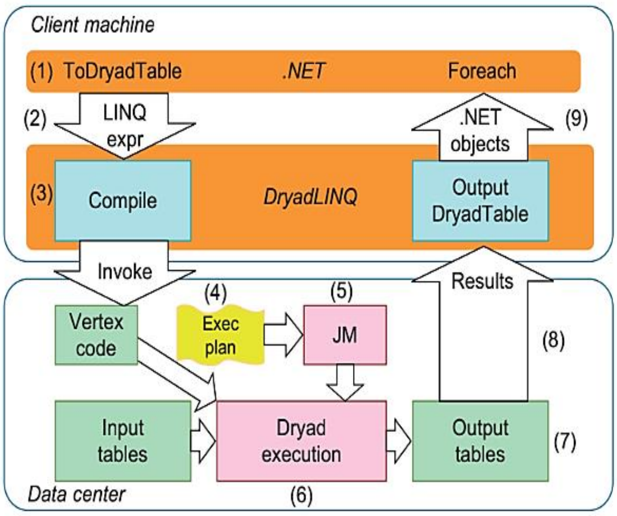


FIGURE 6.14

LINQ-expression execution in DryadLINQ.

DryadLINQ is built on top of Microsoft's Dryad execution framework (see http://research.microsoft.com/en-us/projects/DryadLINQ/). Dryad can perform acyclic task scheduling and run on large-scale servers. The goal of DryadLINQ is to make large-scale, distributed cluster computing available to ordinary programmers. Actually, DryadLINQ, as the name implies, combines two important components: the Dryad distributed execution engine and .NET Language Integrated Query (LINQ). LINQ is particularly for users familiar with a database programming model. Figure 6.14 shows the flow of execution with DryadLINQ. The execution is divided into nine steps as follows:

1. A .NET user application runs, and creates a DryadLINQ expression object. Because of LINQ's deferred evaluation, the actual execution of the expression has not occurred.
2. The application calls *ToDryadTable* triggering a data-parallel execution. The expression object is handed to *DryadLINQ*.
3. DryadLINQ compiles the LINQ expression into a distributed Dryad execution plan. The expression is decomposed into subexpressions, each to be run in a separate Dryad vertex. Code and static data for the remote Dryad vertices are generated, followed by the serialization code for the required data types.
4. DryadLINQ invokes a custom Dryad job manager which is used to manage and monitor the execution flow of the corresponding task.
5. The job manager creates the job graph using the plan created in step 3. It schedules and spawns the vertices as resources become available.

4 a. **Explain the interactions among VM managers for cloud creation and management.** [05] | 3 | L2

Scheme: Explanation + Diagram – **3+2 marks**

Solutions:

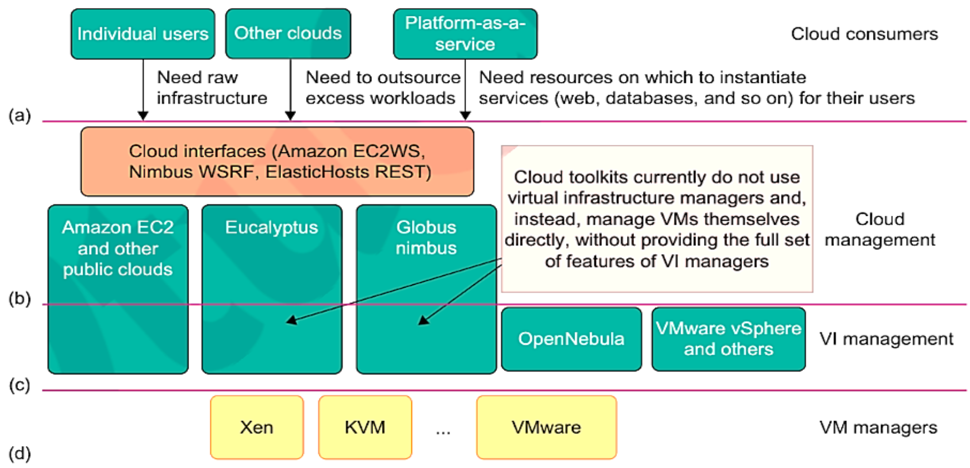


FIGURE 4.4

Cloud ecosystem for building private clouds: (a) Consumers demand a flexible platform; (b) Cloud manager provides virtualized resources over an IaaS platform; (c) VI manager allocates VMs; (d) VM managers handle VMs installed on servers.

An ecosystem was suggested by Sotomayor, et al. [39] (Figure 4.4) for building private clouds. They suggested four levels of ecosystem development in a private cloud. At the user end, consumers demand a flexible platform. At the cloud management level, the cloud manager provides virtualized resources over an IaaS platform. At the virtual infrastructure (VI) management level, the manager allocates VMs over multiple server clusters. Finally, at the VM management level, the VM managers handle VMs installed on individual host machines. An ecosystem of cloud tools attempts to span both cloud management and VI management. Integrating these two layers is complicated by the lack of open and standard interfaces between them.

An increasing number of startup companies are now basing their IT strategies on cloud resources, spending little or no capital to manage their own IT infrastructures. We desire a flexible and open architecture that enables organizations to build private/hybrid clouds. VI management is aimed at this goal. Example VI tools include oVirt (https://fedorahosted.org/ovirt/), vSphere/4

b. Illustrate the infrastructure needed to virtualized servers in data centers. [05]

Scheme: Explanation + Diagram – **3+2 marks**

Solutions:

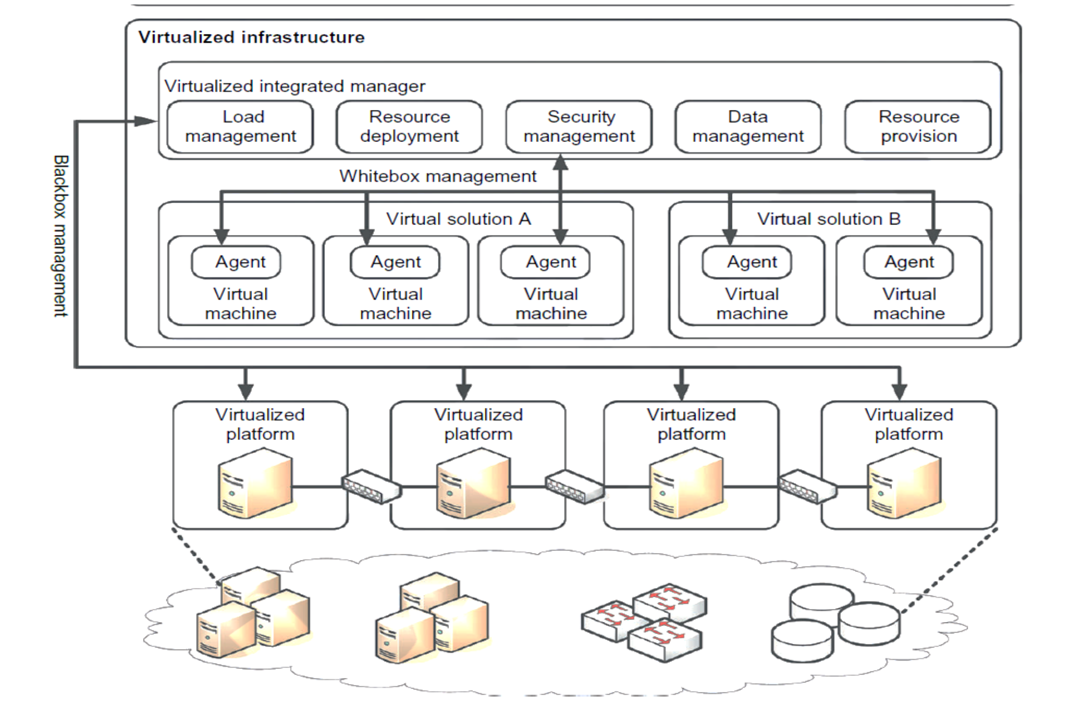


FIGURE 4.17

Virtualized servers, storage, and network for cloud platform construction.

- One very distinguishing feature of cloud computing infrastructure is the use of system virtualization and the modification to provisioning tools.
- Virtualization of servers on a shared cluster can consolidate web services. As the VMs are the containers of cloud services, the provisioning tools will first find the corresponding physical machines and deploy the VMs to those nodes before scheduling the service to run on the virtual nodes.
- In addition, in cloud computing, virtualization also means the resources and fundamental infrastructure are virtualized.
- The user will not care about the computing resources that are used for providing the services.
- Cloud users do not need to know and have no way to discover physical resources that are involved while processing a service request.
- Also, application developers do not care about some infrastructure issues such as scalability and fault tolerance (i.e., they are virtualized). Application developers focus on service logic.

5	**Describe architecture and data mutation sequences in GFS.**	[10]	5	L2

Scheme: Architecture (Explanation + Diagram) – **3+2 marks** and Mutation Sequence (Explanation + Diagram) – **3+2 marks**

Solutions:

GFS was built primarily as the fundamental storage service for Google's search engine. As the size of the web data that was crawled and saved was quite substantial, Google needed a distributed file system to redundantly store massive amounts of data on cheap and unreliable computers. None of the traditional distributed file systems can provide such functions and hold such large amounts of data. In addition, GFS was designed for Google applications, and Google applications were built for GFS. In traditional file system design, such a philosophy is not attractive, as there should be a clear interface between applications and the file system, such as a POSIX interface.

There are several assumptions concerning GFS. One is related to the characteristic of the cloud computing hardware infrastructure (i.e., the high component failure rate). As servers are composed of inexpensive commodity components, it is the norm rather than the exception that concurrent failures will occur all the time. Another concerns the file size in GFS. GFS typically will hold a large number of huge files, each 100 MB or larger, with files that are multiple GB in size quite common. Thus, Google has chosen its file data block size to be 64 MB instead of the 4 KB in typical traditional file systems. The I/O pattern in the Google application is also special. Files are typically written once, and the write operations are often the appending data blocks to the end of files. Multiple appending operations might be concurrent. There will be a lot of large streaming reads and only a little random access. As for large streaming reads, highly sustained throughput is much more important than low latency.

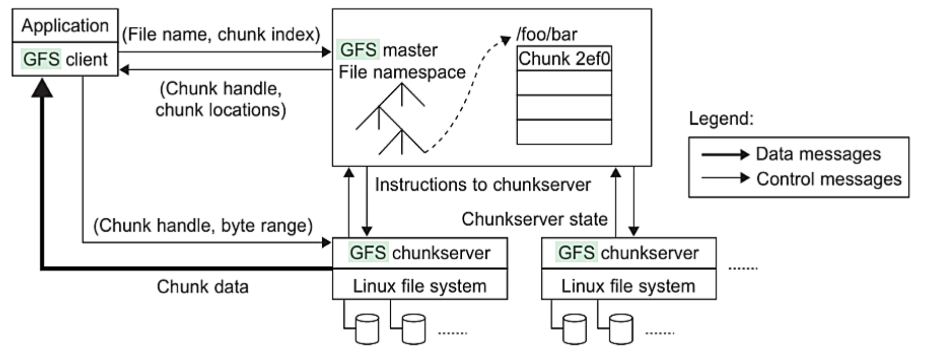
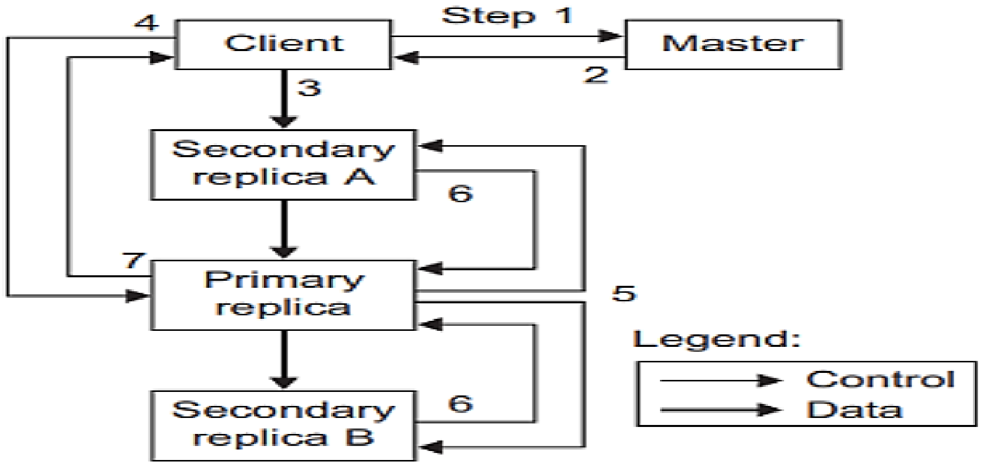


FIGURE 6.18

Architecture of Google File System (GFS).

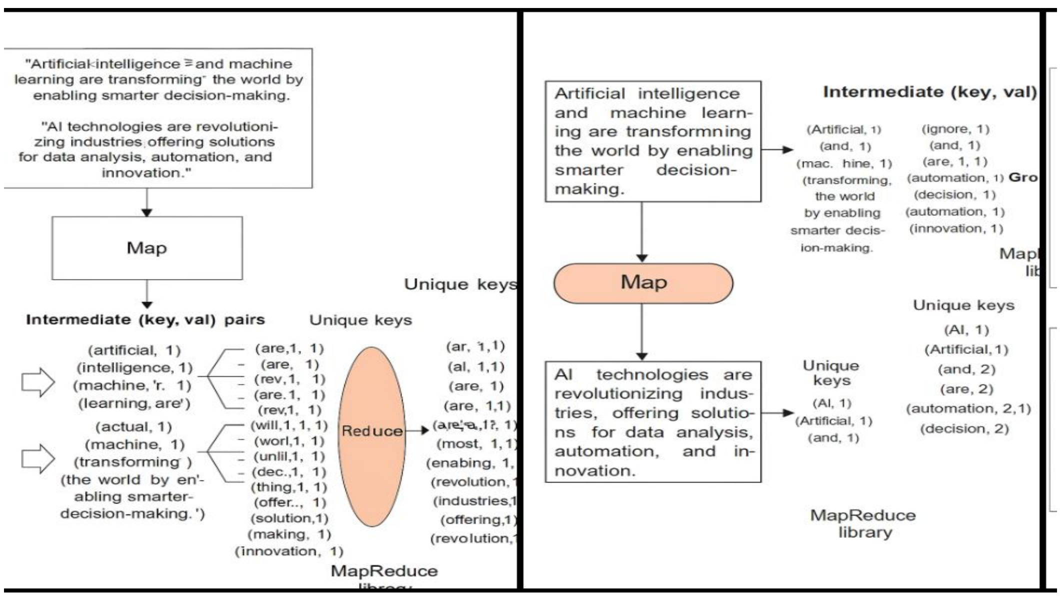
Data Mutation Sequence

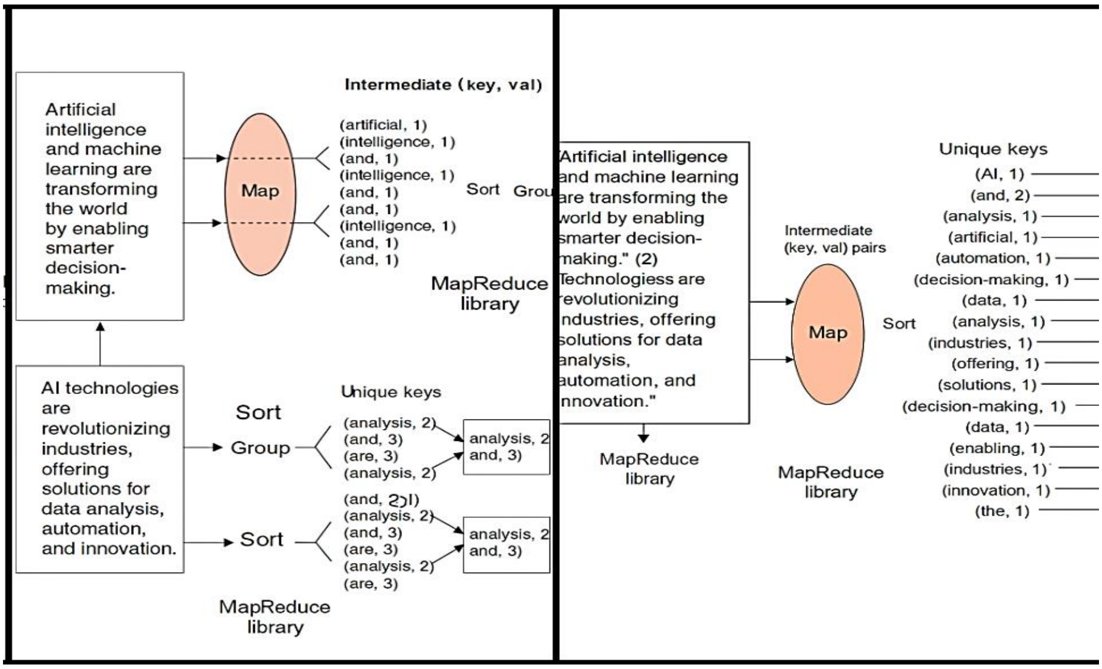


6	**Apply stepwise data flow Map Reduce model for the given word-count problem:** **(1) "Artificial intelligence and machine learning are transforming the world by enabling smarter decision-making."** **(2) "AI technologies are revolutionizing industries, offering solutions for data analysis, automation, and innovation."**	[10]	5	L3

Scheme: Computation of Map,Reduce,Sort,Group,Unique keys – 2 marks each

Solutions:





Faculty Signature CCI Signature HOD Signature