Scheme of Evaluation
Internal Assessment Test 2 – June 2021

Sub:	Cloud Computing and its Application						Code:	18CS643
Date:	23/06/2021	Duration:	90mins	Max Marks:	50	Sem: VI	Branch:	ISE

Note: **Answer Any five full questions.**

Question #		Description	Marks Distribution		Max Marks
1	a)	What is Aneka Container? Explain anatomy of Aneka container in detail. Aneka Container- Services provided by Aneka 1. Fabric Services Profiling and Monitoring Resource management 2. Foundation Services Storage Management Accounting, billing, Resource pricing Resource reservation 3. Application Services Scheduling Execution	2M 1M 2M 3M 2M	10M	10M
2	a)	A company wants to set up a public cloud using Aneka. Discuss the public cloud deployment using Aneka with neat diagram. Diagram Explanation	3M 3M	6M	10M
2	b)	Explain the logical organization of Aneka cloud. Diagram Explanation	2M 2M	4M	

3	a)	What does the term NoSQL mean? Explain Google Bigtable architecture with neat diagram. NoSQL explanation Bigtable architecture diagram Explanation	1M 2.5M 2.5M	6M	10M
3	b)	Explain any two distributed file system in detail. Sector Lustre IBM General Parallel File System (GPFS) Google File System(GFS) Amazon Simple Storage Service(S3) Any two	2M*2	4M	
4	a)	What is Map Reduce Programming model? An application is generating a big data. Suggest the Map Reduce computation workflow with neat diagram to manage the big data. Write Map and Reduce functions for WordCounter Problem. Input File Deer Bear River Car Car River Deer Car Bear Output File Bear 2 Car 3 Deer 2 River 2 Map Reduce Programming Model Computation workflow diagram Explanation Map Reduce functions for WordCounter problem	1M 3M 3M 3M	10M	10M
5	a)	What is data intensive computing? Explain Amazon DynamoDB architecture with neat diagram Data Intensive Computing Amazon DynamoDB architecture diagram Explanation	1M 3M 3M	7M	10M

5	b)	Explain the characteristics of Google File System. Any three characteristics	1M*3	3M	
6	a)	Discuss the various and extensions of Map Reduce (any three). Hadoop Pig Hadoop Map Reduce Merge Twister	2M*3	6M	10M
6	b)	Compare and contrast SQL and NoSQL. Any three differences	1M*4	4M	

Scheme Of Evaluation
Internal Assessment Test 2 – June 2021

Sub:	Cloud Computing and its Application							**Code:**	18CS643
Date:	23/06/2021	Duration:	90mins	Max Marks:	50	**Sem:**	VI	**Branch:**	ISE

Note: **Answer Any full five questions**

Q.1 a) What is Aneka Container? Explain anatomy of Aneka container in detail.

- Aneka is a software platform for developing cloud computing applications.
- Aneka is a pure PaaS solution for cloud computing.
- Aneka is a cloud middleware product that can be deployed on a heterogeneous set of resources: Like: a network of computers, a multi core server, data centers, virtual cloud infrastructures, or a mixture of all
- The framework provides both middleware for managing and scaling distributed applications and an extensible set of APIs for developing them

Anatomy of the Aneka container

- o The Aneka container constitutes the building blocks of Aneka Clouds and represents the runtime machinery available to services and applications
- o The container is the unit of deployment in Aneka Clouds, and it is a lightweight software layer designed to host services and interact with the underlying operating system and hardware

The Aneka container can be classified into three major categories:

• Fabric Services
• Foundation Services
• Application Services

- o These services stack resides on top of the Platform Abstraction Layer (PAL) (Refer Diagram-5.2) it represents the interface to the underlying operating system and hardware.
- o PAL provides a uniform view of the software and hardware environment in which the container is running

Here is the functionality of each components of Aneka Framework given in diagram 5.2

PAL –Platform Abstraction Layer

- In a cloud environment each operating system has a different file system organization and stores that information differently.
- It is The Platform Abstraction Layer (PAL) that addresses this heterogeneity problem with Operating systems and provides the container with a uniform interface for accessing the relevant information, thus the rest of the container can be operated without modification on any supported platform

The PAL provides the following features:

- Uniform and platform-independent implementation interface for accessing the hosting platform
- Uniform access to extended and additional properties of the hosting platform
- Uniform and platform-independent access to remote nodes
- Uniform and platform-independent management interfaces

Also The PAL is a small layer of software that comprises a detection engine, which automatically configures the container at boot time, with the platform-specific component to access the above information and an implementation of the abstraction layer for the Windows, Linux, and Mac OS X operating systems.

Following are the collectible data that are exposed by the PAL:

- Number of cores, frequency, and CPU usage
- Memory size and usage
- Aggregate available disk space
- Network addresses and devices attached to the node

Fabric services

- **FabricServices** define the lowest level of the software stack representing the Aneka Container.
- They provide access to the Resource-provisioning subsystem and to the Monitoring facilities implemented in Aneka.
- Resource-provisioning services are in charge of dynamically providing new nodes on demand by relying on virtualization technologies
- Monitoring services allow for hardware profiling and implement a basic monitoring infrastructure that can be used by all the services installed in the container

The two services of Fabric class are:

- Profiling and monitoring
- Resource management

Profiling and monitoring

Profiling and monitoring services are mostly exposed through following services

- Heartbeat,
- Monitoring,
- Reporting
- The Heart Beat makes the information that is collected through the PAL available
- The Monitoring and Reporting implement a generic infrastructure for monitoring the activity of any service in the Aneka Cloud;

Reporting & Monitoring Functions in detail

- The Reporting Service manages the store for monitored data and makes them accessible to other services for analysis purposes.
- On each node, an instance of the Monitoring Service acts as a gateway to the Reporting Service and forwards to it all the monitored data that has been collected on the node

  Many Built-in services use this channel to provide information, important built-in services are:

  - The Membership Catalogue service tracks the performance information of nodes.
  - The Execution Service monitors several time intervals for the execution of jobs.
  - The Scheduling Service tracks the state transitions of jobs.
  - The Storage Service monitors and obtains information about data transfer such as upload and download times, file names, and sizes.
  - The Resource Provisioning Service tracks the provisioning and life time information of virtual nodes.

  Resource management

  Aneka provides a collection of services that are in charge of managing resources. These are

  - Index Service (or Membership Catalogue)
  - Resource Provisioning Service

Resource Provisioning Service Features

- The resource provisioning infrastructure built into Aneka is mainly concentrated in the Resource Provisioning Service,
- It includes all the operations that are needed for provisioning virtual instances (Providing virtual instances as needed by users).
- The implementation of the service is based on the idea of resource pools.
- A resource pool abstracts the interaction with a specific IaaS provider by exposing a common interface so that all the pools can be managed uniformly.

Foundation services

Foundation Services are related to the logical management of the distributed system built on top of the infrastructure and provide supporting services for the execution of distributed applications.

These services cover:

- Storage management for applications
- Accounting, billing and resource pricing
- Resource reservation

Storage management

Aneka offers two different facilities for storage management:

- A centralized file storage, which is mostly used for the execution of compute- intensive applications,
- A distributed file system, which is more suitable for the execution of data-intensive applications.
- As the requirements for the two types of applications are rather different. Compute-intensive applications mostly require powerful processors and do not have high demands in terms of storage, which in many cases is used to store small files that are easily transferred from one node to another. Here, a centralized storage node is sufficient to store data
- [illegible]

Accounting, billing, and resource pricing

- Accounting Services keep track of the status of applications in the Aneka Cloud
- The information collected for accounting is primarily related to infrastructure usage and application execution
- Billing is another important feature of accounting.
- Billing is important since Aneka is a multitenant cloud programming platform in which the execution of applications can involve provisioning additional resources from commercial providers.

Resource reservation

Resource Reservation supports the execution of distributed applications and allows for reserving resources for exclusive use by specific applications.

Two types of services are used to build resource reservation:

- The Resource Reservation
- The Allocation Service
- Resource Reservation keeps track of all the reserved time slots in the Aneka Cloud and provides a unified view of the system. (provides overview of the system)
- The Allocation Service is installed on each node that features execution services and manages the database of information regarding the allocated slots on the local node.

Application services

Application Services manage the execution of applications and constitute a layer that differentiates according to the specific programming model used for developing distributed applications on top of Aneka

Two types of services are:

1. The Scheduling Service :

Scheduling Services are in charge of planning the execution of distributed applications on top of Aneka and governing the allocation of jobs composing an application to nodes. Common tasks that are performed by the scheduling component are the following:

- Job to node mapping
- Rescheduling of failed jobs
- Job status monitoring
- Application status monitoring

2. The Execution Service

Execution Services control the execution of single jobs that compose applications. They are in charge of setting up the runtime environment hosting the execution of jobs.

Some of the common operations that apply across all the range of supported models are:

- Unpacking the jobs received from the scheduler
- Retrieval of input files required for job execution
- Sandboxed execution of jobs
- Submission of output files at the end of execution

Q. 2 a) A company wants to set up a public cloud using Aneka. Discuss the public cloud deployment using Aneka with neat diagram.

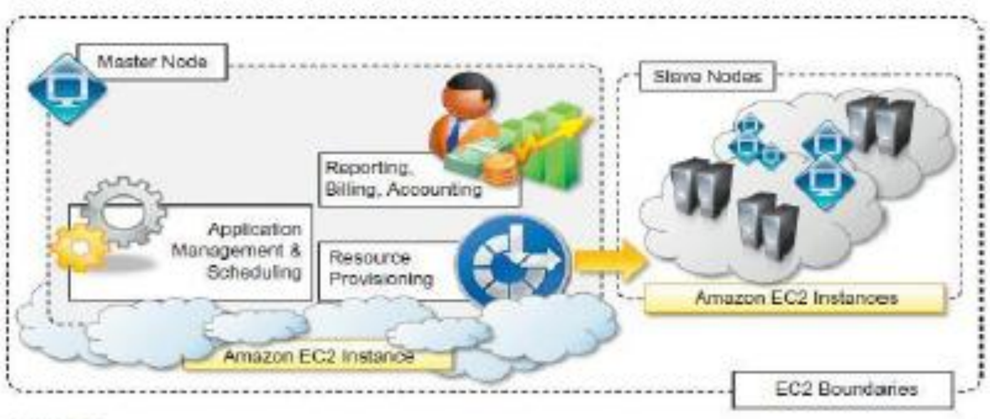


FIGURE 5.6

Public Aneka cloud deployment.

Public cloud deployment mode

Public Cloud deployment mode features the installation of Aneka master and worker nodes over a completely virtualized infrastructure that is hosted on the infrastructure of one or more resource providers such as Amazon EC2 or GoGrid.

Figure 5.6 provides an overview of this scenario. The deployment is generally contained within the infrastructure boundaries of a single IaaS pro- vider. The reasons for this are to minimize the data transfer between different providers, which is generally priced at a higher cost, and to have better network performance. In this scenario it is pos- sible to deploy an Aneka Cloud composed of only one node and to completely leverage dynamic provisioning to elastically scale the infrastructure on demand. A fundamental role is played by the Resource Provisioning Service, which can be configured with different images and templates to instantiate. Other important services that have to be included in the master node are the Accounting and Reporting Services. These provide details about resource utilization by users and applications and are fundamental in a multitenant Cloud where users are billed according to their consumption of Cloud capabilities.

Q. 2 b) Explain the logical organization of Aneka cloud.

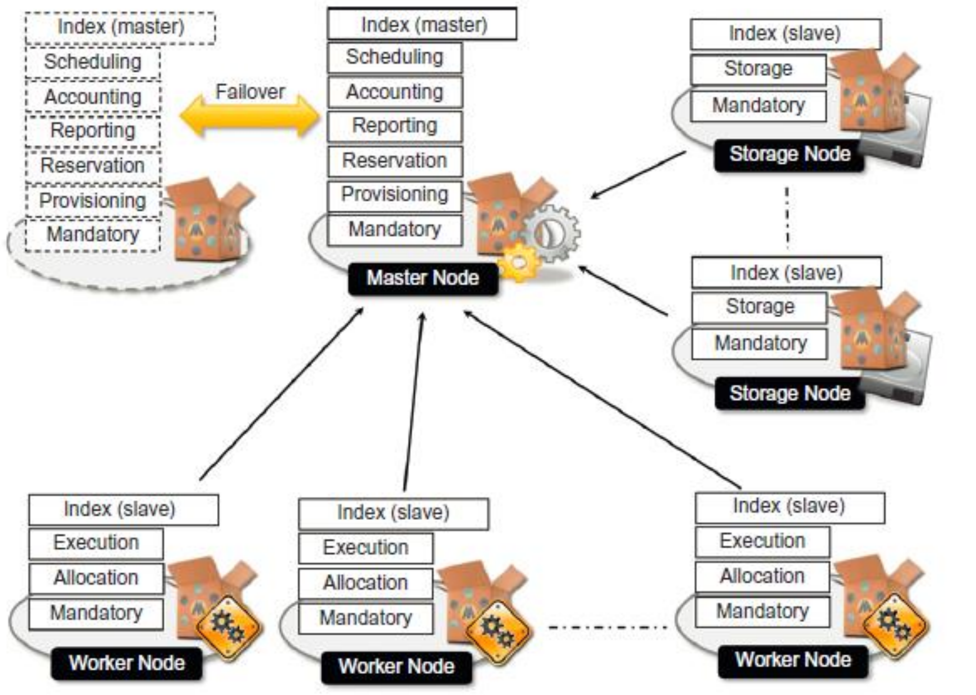


FIGURE 5.4

Logical organization of an Aneka cloud.

Logical organization

The logical organization of Aneka Clouds can be very diverse, since it strongly depends on the configuration selected for each of the container instances belonging to the Cloud.

Here is a scenario that has master-worker configuration with separate nodes for storage, the Figure 5.4. Portray

The master node comprises of following services:

- Index Service (master copy)
- Heartbeat Service
- Logging Service
- Reservation Service
- Resource Provisioning Service
- Accounting Service
- Reporting and Monitoring Service
- Scheduling Services for the supported programming models

Here Logging service and Heartbeat service and Monitoring service are considered as Mandatory services in all the block diagrams whereas other services are shown ditto.

Q. 3a) What does the term NoSQL mean? Explain Google Bigtable architecture with neat diagram.

NoSQL, which stands for "not only SQL," is an approach to database design that provides flexible schemas for the storage and retrieval of data beyond the traditional table structures found in relational databases.

c. Google Bigtable.

Bigtable provides storage support for several Google applications that expose different types of workload: from throughput-oriented batch-processing jobs to latency-sensitive serving of data to end users.

Bigtable's key design goals are wide applicability, scalability, high performance, and high availability. To achieve these goals, Bigtable organizes the data storage in tables of which the rows are distributed over the distributed file system supporting the middleware, which is the Google File System.

From a logical point of view, a table is a multidimensional sorted map indexed by a key that is represented by a string of arbitrary length. A table is organized into rows and columns; columns can be grouped in column family, which allow for specific optimization for better access control, the storage and the indexing of data.

Bigtable APIs also allow more complex operations such as single row transactions and advanced data manipulation.

Figure 8.4 gives an overview of the infrastructure that enables Bigtable.

The service is the result of a collection of processes that coexist with other processes in a cluster-based environment. Bigtable identifies two kinds of processes: master processes and tablet server processes. A tablet server is responsible for serving the requests for a given tablet that is a contiguous partition of rows of a table. Each server can manage multiple tablets (commonly from 10 to 1,000). The master server is responsible for keeping track of the status of the tablet servers and of the allocation of tablets to tablet servers. The server constantly monitors the tablet servers to check whether they are alive, and in case they are not reachable, the allocated tablets are reassigned and eventually partitioned to other servers.

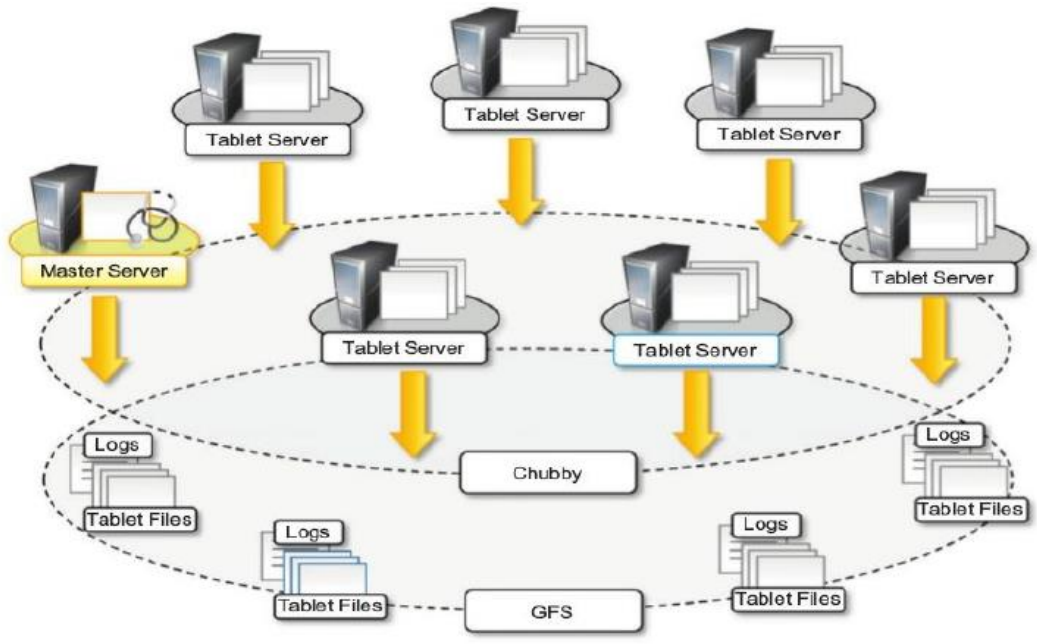


FIGURE 8.4

Bigtable architecture.

Q. 3b) Explain any two distributed file system in detail.

a. Lustre. The Lustre file system is a massively parallel distributed file system that covers the needs of a small workgroup of clusters to a large-scale computing cluster. The file system is used by several of the Top 500 supercomputing systems.

Lustre is designed to provide access to petabytes (PBs) of storage to serve thousands of clients with an I/O throughput of hundreds of gigabytes per second (GB/s). The system is composed of a metadata server that contains the metadata about the file system and a collection of object storage servers that are in charge of providing storage.

b. IBM General Parallel File System (GPFS). GPFS is the high-performance distributed file system developed by IBM that provides support for the RS/6000 supercomputer and Linux computing clusters. GPFS is a multiplatform distributed file system built over several years of academic research and provides advanced recovery mechanisms. GPFS is built on the concept of shared disks, in which a collection of disks is attached to the file system nodes by means of some switching fabric. The file system makes this infrastructure transparent to users and stripes large files over the disk array by replicating portions of the file to ensure high availability.

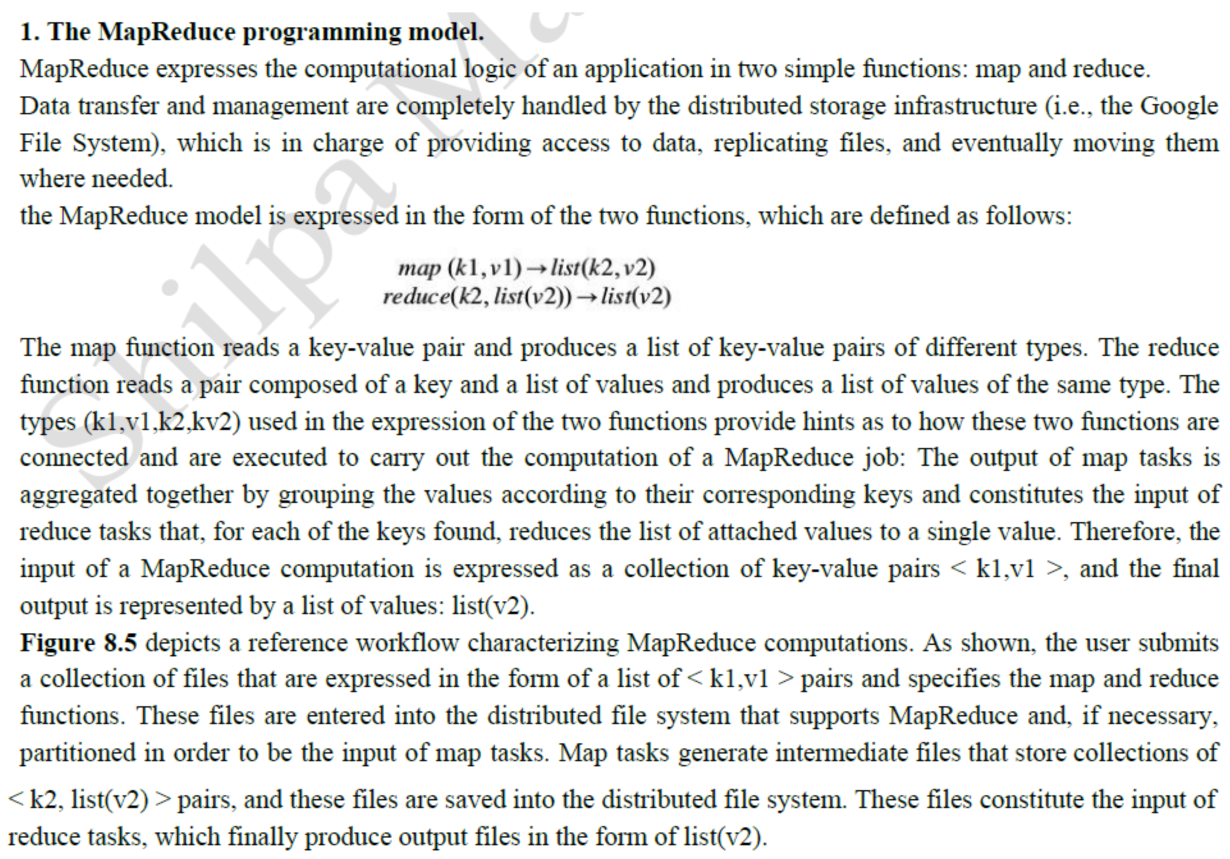
Q. 4a) What is Map Reduce Programming model? An application is generating a big data. Suggest the Ma
Reduce computation workflow with neat diagram to manage the big data. Write Map and Reduce functions f
WordCounter Problem.

Input File

Deer Bear River
Car Car River
Deer Car Bear

Output File

Bear 2
Car 3
Deer 2
River 2



1. The MapReduce programming model.

MapReduce expresses the computational logic of an application in two simple functions: map and reduce.
Data transfer and management are completely handled by the distributed storage infrastructure (i.e., the Google File System), which is in charge of providing access to data, replicating files, and eventually moving them where needed.
the MapReduce model is expressed in the form of the two functions, which are defined as follows:

$$map\,(k1, v1) \rightarrow list(k2, v2)$$
$$reduce(k2, list(v2)) \rightarrow list(v2)$$

The map function reads a key-value pair and produces a list of key-value pairs of different types. The reduce function reads a pair composed of a key and a list of values and produces a list of values of the same type. The types (k1,v1,k2,kv2) used in the expression of the two functions provide hints as to how these two functions are connected and are executed to carry out the computation of a MapReduce job: The output of map tasks is aggregated together by grouping the values according to their corresponding keys and constitutes the input of reduce tasks that, for each of the keys found, reduces the list of attached values to a single value. Therefore, the input of a MapReduce computation is expressed as a collection of key-value pairs < k1,v1 >, and the final output is represented by a list of values: list(v2).

Figure 8.5 depicts a reference workflow characterizing MapReduce computations. As shown, the user submits a collection of files that are expressed in the form of a list of < k1,v1 > pairs and specifies the map and reduce functions. These files are entered into the distributed file system that supports MapReduce and, if necessary, partitioned in order to be the input of map tasks. Map tasks generate intermediate files that store collections of < k2, list(v2) > pairs, and these files are saved into the distributed file system. These files constitute the input of reduce tasks, which finally produce output files in the form of list(v2).

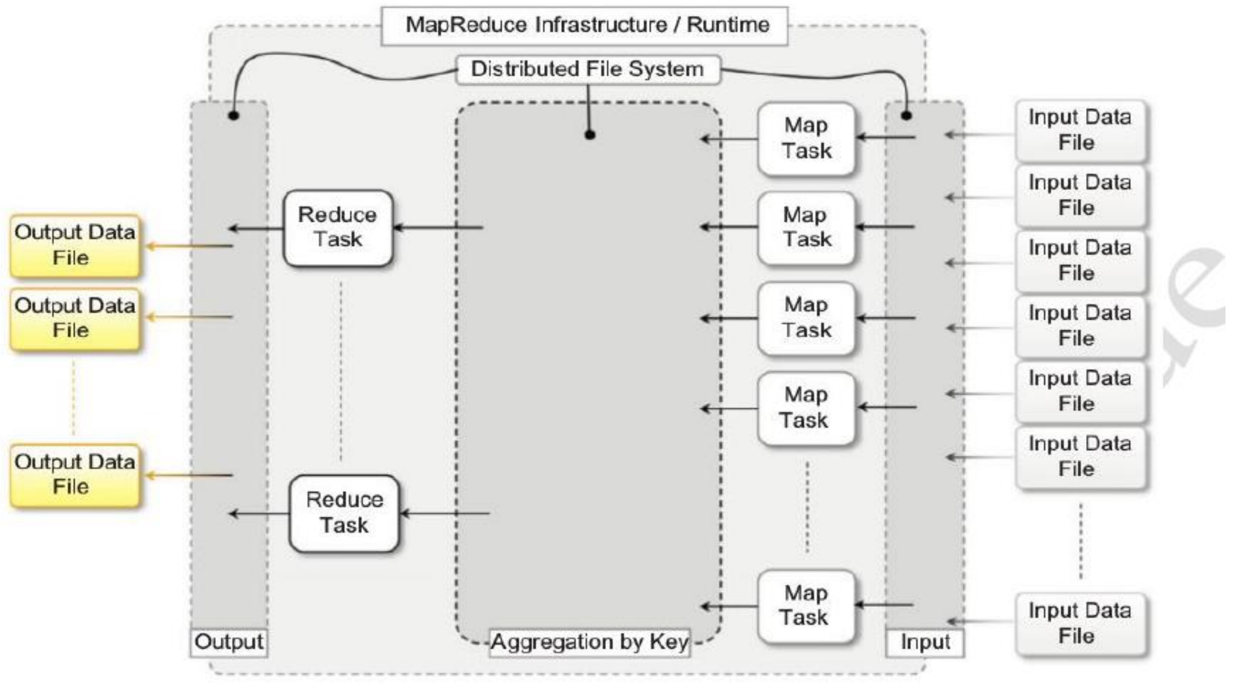
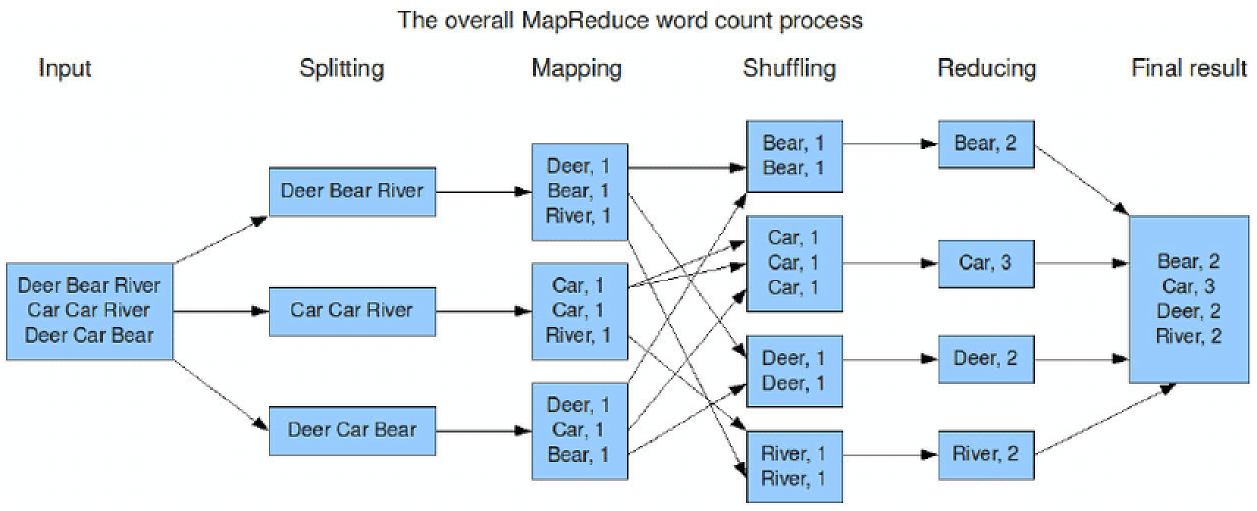


FIGURE 8.5

MapReduce computation workflow.

The computation model expressed by MapReduce is very straightforward and allows greater productivity for people who have to code the algorithms for processing huge quantities of data.



MapReduce consists of 2 steps:

- **Map Function –** It takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (Key-Value pair).

Example – (Map function in Word Count)

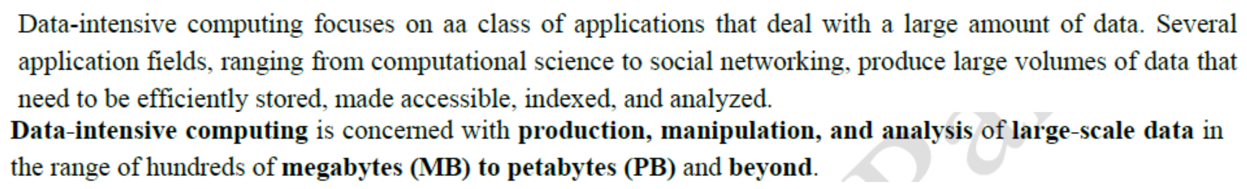
Input	Set of data	Deer, Bear, River, Car, Car, River, Deer, Car, Bear
Output	Convert into another set of data (Key,Value)	(Deer,1), (Bear,1), (River,1), (Car,1), (Car,1), (River,1), (Deer,1), (Car,1), (Bear,1)

Reduce Function – Takes the output from Map as an input and combines those data tuples into a smaller set of tuples.

Input **(output of Map function)**	Set of Tuples	(Deer,1), (Bear,1), (River,1), (Car,1), (Car,1), (River,1), (Deer,1), (Car,1), (Bear,1)
Output	Converts into smaller set of tuples	(Deer,2), (Bear,2), (River,2) (Car,3)

Example – (Reduce function in Word Count)

Q. 5a) What is data intensive computing? Explain Amazon DynamoDB architecture with neat diagram.



Data-intensive computing focuses on aa class of applications that deal with a large amount of data. Several application fields, ranging from computational science to social networking, produce large volumes of data that need to be efficiently stored, made accessible, indexed, and analyzed.

Data-intensive computing is concerned with **production, manipulation, and analysis** of **large-scale data** in the range of hundreds of **megabytes (MB) to petabytes (PB)** and **beyond**.

b. Amazon Dynamo.

The main goal of Dynamo is to provide an incrementally scalable and highly available storage system. This goal helps in achieving reliability at a massive scale, where thousands of servers and network components build an infrastructure serving 10 million requests per day. Dynamo provides a simplified interface based on get/put semantics, where objects are stored and retrieved with a unique identifier (key).

The architecture of the Dynamo system, shown in **Figure 8.3**, is composed of a collection of storage peers organized in a ring that shares the key space for a given application. The key space is partitioned among the storage peers, and the keys are replicated across the ring, avoiding adjacent peers. Each peer is configured with access to a local storage facility where original objects and replicas are stored.

Each node provides facilities for distributing the updates among the rings and to detect failures and unreachable nodes.

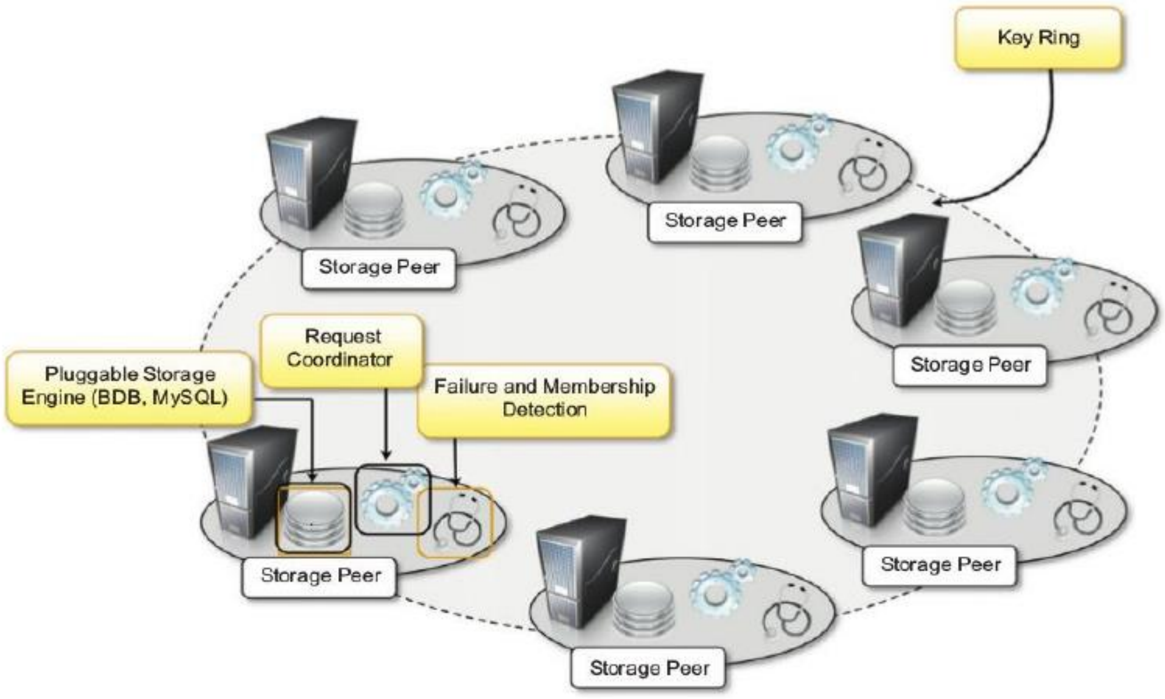


FIGURE 8.3

Amazon Dynamo architecture.

Q. 5 b) Explain the characteristics of Google File System

1. The system is built on top of commodity hardware that often fails.
2. The system stores a modest number of large files; multi-GB files are common and should be treated efficiently, and small files must be supported, but there is no need to optimize for that.
3. The workloads primarily consist of two kinds of reads: large streaming reads and small random reads.
4. The workloads also have many large, sequential writes that append data to files.
5. High-sustained bandwidth is more important than low latency.

Q. 6 a) Discuss the various and extensions of Map Reduce (any three).

A. Hadoop.

Apache Hadoop is a collection of software projects for reliable and scalable distributed computing. The initiative consists of mostly two projects: Hadoop Distributed File System (HDFS) and Hadoop MapReduce. The former is an implementation of the Google File System; the latter provides the same features and abstractions as Google MapReduce.

B. Pig.

Pig is a platform that allows the analysis of large datasets. Developed as an Apache project, Pig consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The Pig infrastructure's layer consists of a compiler for a high-level language that produces a sequence of MapReduce jobs that can be run on top of distributed infrastructures.

C. Hive.

Hive is another Apache initiative that provides a data warehouse infrastructure on top of Hadoop MapReduce. It provides tools for easy data summarization, ad hoc queries, and analysis of large datasets stored in Hadoop MapReduce files.

Hive's major advantages reside in the ability to scale out, since it is based on the Hadoop framework, and in the ability to provide a data warehouse infrastructure in environments where there is already a Hadoop system running.

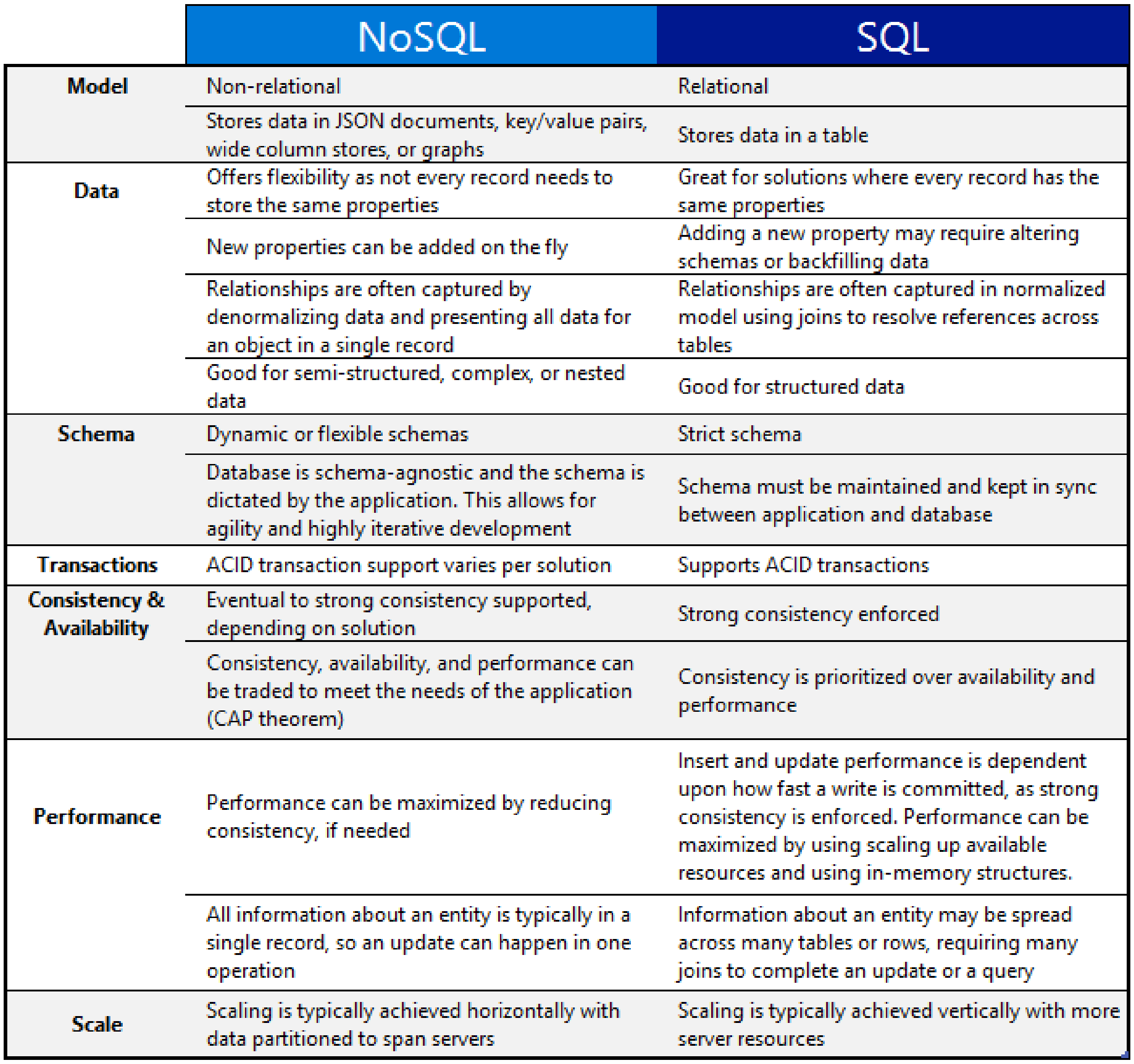
E. Twister.

Twister is an extension of the MapReduce model that allows the creation of iterative executions of MapReduce jobs. With respect to the normal MapReduce pipeline, the model proposed by Twister proposes the following extensions:

```
1. Configure Map
2. Configure Reduce
3. While Condition Holds True Do
    a. Run MapReduce
    b. Apply Combine Operation to Result
    c. Update Condition
4. Close
```

Twister provides additional features such as the ability for map and reduce tasks to refer to static and in-memory data; the introduction of an additional phase called combine, run at the end of the MapReduce job, that aggregates the output together.

Q. 6 b) Compare and contrast SQL and NoSQL.



	NoSQL	SQL
Model	Non-relational	Relational
	Stores data in JSON documents, key/value pairs, wide column stores, or graphs	Stores data in a table
Data	Offers flexibility as not every record needs to store the same properties	Great for solutions where every record has the same properties
	New properties can be added on the fly	Adding a new property may require altering schemas or backfilling data
	Relationships are often captured by denormalizing data and presenting all data for an object in a single record	Relationships are often captured in normalized model using joins to resolve references across tables
	Good for semi-structured, complex, or nested data	Good for structured data
Schema	Dynamic or flexible schemas	Strict schema
	Database is schema-agnostic and the schema is dictated by the application. This allows for agility and highly iterative development	Schema must be maintained and kept in sync between application and database
Transactions	ACID transaction support varies per solution	Supports ACID transactions
Consistency & Availability	Eventual to strong consistency supported, depending on solution	Strong consistency enforced
	Consistency, availability, and performance can be traded to meet the needs of the application (CAP theorem)	Consistency is prioritized over availability and performance
Performance	Performance can be maximized by reducing consistency, if needed	Insert and update performance is dependent upon how fast a write is committed, as strong consistency is enforced. Performance can be maximized by using scaling up available resources and using in-memory structures.
	All information about an entity is typically in a single record, so an update can happen in one operation	Information about an entity may be spread across many tables or rows, requiring many joins to complete an update or a query
Scale	Scaling is typically achieved horizontally with data partitioned to span servers	Scaling is typically achieved vertically with more server resources