UNIT–5 : Cloud Technologies and Advancements (PYQ Solution)

AKTU · Cloud Computing (BCAM051) · Solutions by **Chirag Jain**, Ycotes

SECTION A Answers (≈70 Words Each)

Q1. What is Hadoop and why is it considered suitable for big data processing? (Jan 2026)

Apache Hadoop is an open-source framework for distributed storage (HDFS) and parallel processing (MapReduce) of very large data sets on clusters of commodity hardware. It suits big data because it is scalable (add nodes), fault tolerant (data replicated 3 times), cost effective, processes data where it is stored (data locality) and handles structured and unstructured data.

Q2. Describe cloud federation. (Jan 2025)

Cloud federation is the interconnection of multiple cloud providers' infrastructures so that they can share and exchange resources, services and workloads under agreed policies and common standards. A cloud short of capacity borrows from partners, improving scalability, availability, cost efficiency and avoidance of vendor lock-in. It needs interoperability, trust, security and SLA agreements.

Q3. Evaluate the role of federated services and applications. (Jan 2025)

Federated services and applications run across several clouds as one logical system. Their roles are to give resource elasticity beyond one provider, geographic reach and low latency, higher fault tolerance and disaster recovery, cost optimisation by choosing the cheapest provider, and no lock-in. Federated identity gives single sign-on across clouds. Challenges include interoperability, security and consistent billing.

Q4. List the four levels of Federation. (2024)

1. **Permissive (policy) level: agreements, trust and policies between providers.**
2. **Legal level: contracts, regulations, SLAs and jurisdiction.**
3. **Logical (technical) level: interoperability of APIs, protocols and data formats.**
4. **Physical (infrastructure) level: sharing of actual hardware, networks and data centres.**
   (Some texts also describe the levels as security/trust, management, logical and physical.)

Q5. Define VirtualBox. (2024)

Oracle VirtualBox is a free, open-source Type 2 (hosted) hypervisor for x86 hardware. It runs on Windows, Linux and macOS and lets users create and run several guest operating systems as virtual machines, with features such as snapshots, shared folders, USB support and virtual networking. It is widely used for testing and learning.

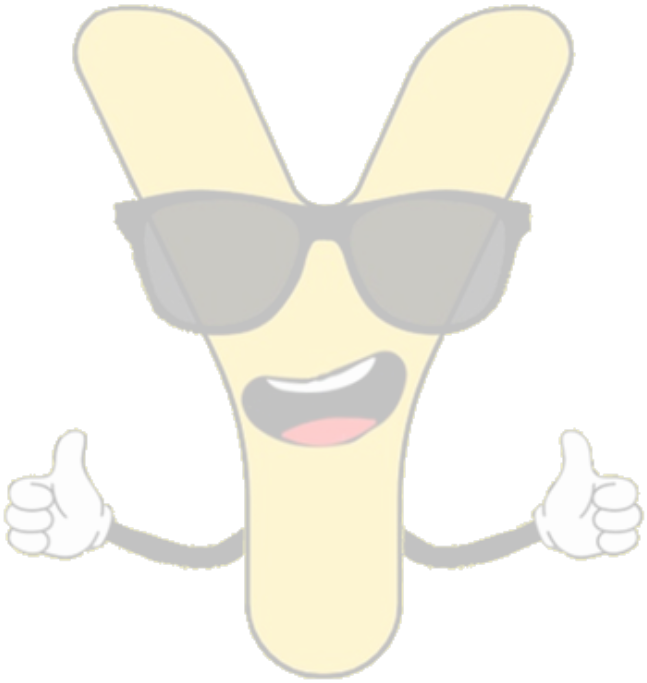
Q6. Define OpenStack and explain its significance in cloud computing. (Feb 2024)

OpenStack is an open-source cloud operating system for building and managing public and private IaaS clouds, controlling large pools of compute, storage and networking through a dashboard and APIs. Core services: Nova (compute), Swift (object storage), Neutron (network), Keystone (identity), Glance (images), Cinder (block storage). Significance: vendor neutrality, no licence cost, flexibility and a large community.

Q7. Explain Google App Engine. (Feb 2024)

Google App Engine (GAE) is a PaaS from Google for developing and hosting web applications in Google's data centres. Developers write code in Python, Java, Go, Node.js or PHP, and GAE handles automatic scaling, load balancing,

patching and monitoring. It offers services such as Datastore, Memcache, Task Queues and URL Fetch, with a free quota and pay-per-use pricing.

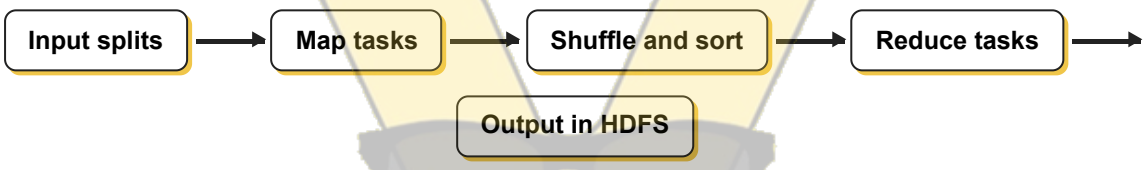


SECTION B Answers (≈250 Words Each)

Q1. Explain the MapReduce programming model and its role in the Hadoop ecosystem. (Jan 2026, Feb 2024)

MapReduce is a programming model for processing huge data sets in parallel across a cluster. The user writes two functions:

- **Map:** takes input key-value pairs and emits intermediate pairs, map(k1, v1) → list(k2, v2).
- **Reduce:** receives each intermediate key with the list of its values and emits the final result, reduce(k2, list(v2)) → list(v3).
  Between them the framework performs **shuffle and sort**, grouping all values with the same key.



MapReduce data flow

Word-count example: input "cloud is big, cloud is fast". Map emits (cloud,1), (is,1), (big,1), (cloud,1), (is,1), (fast,1). After shuffle: cloud → [1,1], is → [1,1], big → [1], fast → [1]. Reduce sums: cloud 2, is 2, big 1, fast 1.

Role in Hadoop: MapReduce is Hadoop's **processing engine**. HDFS splits files into blocks (128 MB); YARN schedules map tasks on nodes holding the blocks (**data locality**); failed tasks are re-executed automatically (fault tolerance). It lets programmers process terabytes without handling parallelisation, distribution or failures, and underlies tools such as Hive and Pig. An optional **combiner** performs local reduction to cut network traffic.

Conclusion: MapReduce makes large-scale batch processing simple, scalable and reliable.

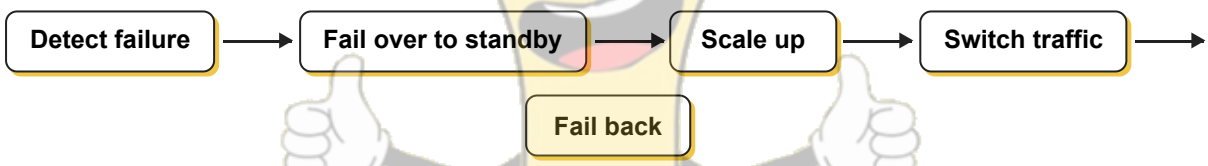
Q2. Analyze the role of cloud technologies in disaster recovery. Simulate a recovery process for a critical application and compute the downtime costs. (Jan 2025)

Role of the cloud in DR:

- **Geographic redundancy:** data and VMs replicated to another region.
- **Pay-per-use standby:** pay for a minimal "pilot light" and scale up only on disaster.
- **Automated failover and orchestration** (infrastructure as code), fast VM image restore.
- **Backup/snapshot services**, regular test drills without hardware.
- **Lower RTO/RPO** and cost compared with a second physical data centre.

Simulation (critical order-processing application). Downtime loss = ₹50,000 per hour.

- **Without cloud DR (restore from tape/backup):** RTO = 4 h → loss = 4 × 50,000 = **₹2,00,000**.
- **With cloud DR (warm standby):** detection 5 min + failover and database promotion 10 min + scale-out 10 min + DNS/traffic switch 5 min = **30 min** → loss = 0.5 × 50,000 = **₹25,000**.
- Saving per incident = 2,00,000 − 25,000 = **₹1,75,000** (87.5%).
- Standby cost ₹40,000/month = ₹4,80,000/year; if one disaster occurs in 3 years, the cloud DR investment (₹14,40,000) is not recovered by one incident, but it is justified for critical systems when risk, data loss and reputation are considered, or with a cheaper pilot-light setup.



Cloud DR process

Conclusion: cloud DR shrinks downtime sharply, and cost is weighed against downtime loss.

Q3. Explain Hadoop and its history. Also illustrate the Hadoop architecture. (2024)

Hadoop is an open-source Apache framework for storing and processing big data across clusters of commodity machines.

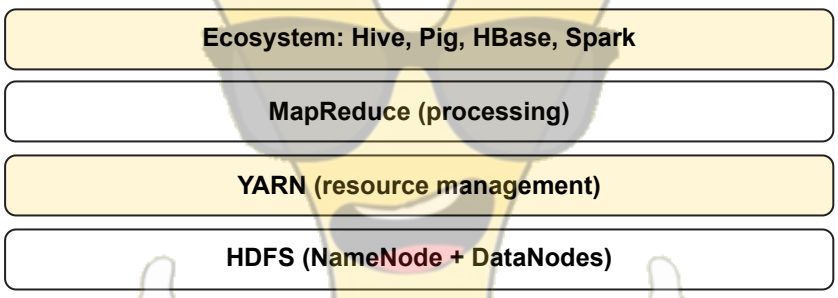
History:

- 2003-2004: Google publishes the **GFS** and **MapReduce** papers.

- 2002-2005: Doug Cutting and Mike Cafarella build the **Nutch** web crawler and implement these ideas.
- 2006: Hadoop becomes a separate project (named after Cutting's son's toy elephant); Yahoo! adopts it.
- 2008: becomes a top-level Apache project; sorts a terabyte fastest.
- 2011-2013: Hadoop 1.0 and then **2.0 with YARN**; 2017: Hadoop 3.0 (erasure coding).

Architecture:

- **HDFS (storage):** master **NameNode** stores metadata; **DataNodes** store 128 MB blocks replicated 3 times; **Secondary NameNode** checkpoints.
- **YARN (resource management): ResourceManager** and per-node **NodeManager** run containers.
- **MapReduce (processing):** JobTracker/ApplicationMaster schedules map and reduce tasks near the data.
- **Ecosystem:** Hive, Pig, HBase, Sqoop, Flume, Spark.



Hadoop architecture

Conclusion: Hadoop offers scalable, fault-tolerant storage and computation on cheap hardware.

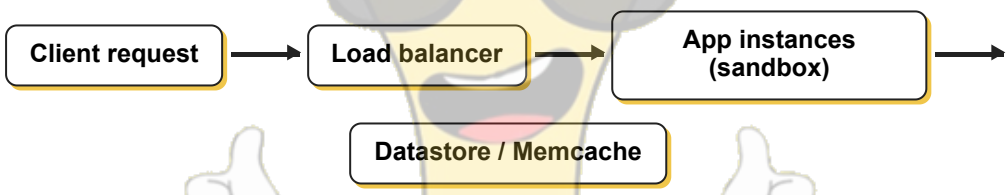
SECTION C Answers (≈500 Words Each)

Q1. Explain Google App Engine architecture and its programming environment. Simulate deployment of a web application scaling to 100,000 daily users. (Jan 2026, Jan 2025)

Google App Engine (GAE) is Google's **PaaS** for building and running web applications on Google's infrastructure with automatic scaling.

Architecture:

- **Front end / load balancer** receives HTTP requests and routes them.
- **App servers (instances)** run the application code in a **sandbox** (restricted environment, no direct file or socket access); instances are created and destroyed automatically.
- **Services:** Datastore (NoSQL, Cloud Firestore), **Memcache** (cache), **Task Queue / Cron**, URL Fetch, Mail, Blobstore/Cloud Storage and Users (authentication).
- **Admin console / SDK** for monitoring and deployment.



GAE request flow

Programming environment:

- Languages: **Python, Java, Go, Node.js, PHP, Ruby**; frameworks such as Django and Flask.
- A **local SDK/development server** for testing; deploy with `gcloud app deploy`.
- Configuration in **app.yaml** (runtime, handlers, scaling settings).
- Applications are **stateless**; state is stored in the Datastore or Memcache.
- **Standard and flexible** environments; automatic, manual or basic scaling.
- Request time limits and sandbox restrictions.

```
runtime: python39
automatic_scaling:
  min_instances: 1
  max_instances: 10
  target_cpu_utilization: 0.65
```

Deployment simulation for 100,000 daily users: each user makes 10 requests, so 1,000,000 requests/day = 1,000,000/86,400 ≈ **11.6 requests/s average**. Assume peak = 10 × average ≈ **116 requests/s**. One F2 instance handles about 20 requests/s, so peak instances = 116/20 ≈ **6**. App Engine starts with 1 instance, scales to about 6 at peak, and returns to 1-2 at night. Datastore and Memcache absorb read load; Cloud Storage serves static files. The developer only deploys the code and sets max_instances.

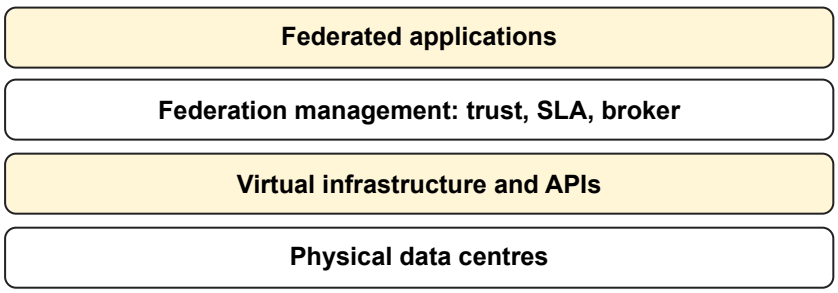
Conclusion: GAE removes server management, so scaling to 100,000 users needs only configuration.

Q2. Explain federation in cloud computing and its four levels. Define the cloud federation stack. Discuss future trends and challenges. (Jan 2026, 2024)

Cloud federation is the **voluntary interconnection of cloud providers' services and infrastructure** so that they can exchange workloads and resources under shared rules. It allows a provider to **scale beyond its own capacity, improve availability, and reduce cost and lock-in**.

Cloud federation stack (layers):

1. **Application/service layer:** federated applications and services.
2. **Federation management layer:** trust, identity federation, SLA negotiation, brokering, resource discovery and monitoring.
3. **Virtual infrastructure layer:** common virtual machine formats and interoperable APIs (OVF).
4. **Physical layer:** the members' data centres and networks.



Cloud federation stack

Four levels of federation:

- **Permissive level:** policies and trust relationships.
- **Legal level:** contracts, compliance and jurisdiction.

- **Logical level:** technical interoperability (APIs, formats, protocols).
- **Physical level:** shared infrastructure and network connectivity.

Future trends: multi-cloud and **edge/fog integration**, standard APIs and open-source platforms (OpenStack, Kubernetes), **blockchain-based trust and billing**, AI-driven placement and cost optimisation, and sovereign clouds (for example regional data clouds).

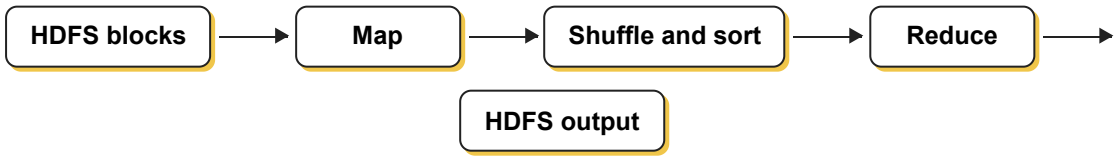
Challenges: lack of **standards and interoperability**, **security and trust** between providers, SLA and pricing negotiation, **data transfer cost and latency**, legal and data-sovereignty issues, consistent monitoring and billing, and complex management.

Conclusion: federation promises elastic, resilient clouds once standards and trust are in place.

Q3. Explain the Hadoop MapReduce framework and how Hadoop handles large-scale data processing. Calculate the processing time for a 1 TB dataset. Define Hadoop's role in distributed computing. (Jan 2025, Feb 2024)

Hadoop is an open-source framework that stores data in **HDFS** (blocks of 128 MB, replicated three times, managed by NameNode and DataNodes) and processes it using **MapReduce** under **YARN**. Large data is handled by **moving computation to data**: each map task runs on a node holding its block, giving parallelism, fault tolerance (re-run failed tasks, replicas) and linear scalability on commodity hardware. In distributed computing it gives a ready platform for storage, scheduling and processing.

MapReduce phases: input split → **map** (key-value pairs) → combiner → **shuffle and sort** → **reduce** → output to HDFS.



Hadoop MapReduce job

Processing time for 1 TB (assumed parameters):

- Data size: 1 TB = 1,048,576 MB; block size 128 MB → map tasks = 1,048,576/128 = **8,192**.

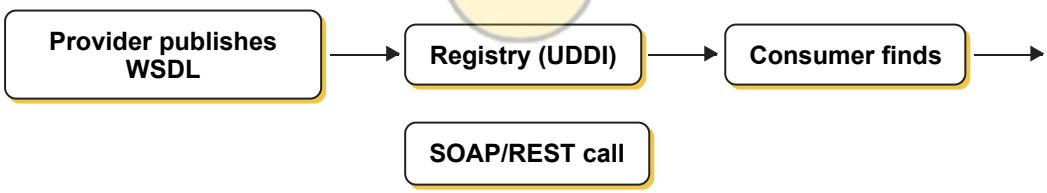
- Cluster: 100 nodes × 4 map slots = **400 parallel map slots**; each map task takes 40 s.
- Map waves = 8,192/400 = 20.48 → **21 waves** × 40 s = **840 s**.
- Shuffle of intermediate data (≈ 200 GB) over the network: assume **300 s**.
- Reduce: 100 reduce tasks in one wave, each taking **120 s**.
- **Total ≈ 840 + 300 + 120 = 1,260 s ≈ 21 minutes.**
  (Without parallelism the map phase alone would take 8,192 × 40 s = 327,680 s ≈ 91 hours; the speed-up is about 390×.)

Conclusion: Hadoop turns days of work into minutes by splitting the job over many nodes.

Q4. Explain Web services in detail and differentiate Web services and APIs. (2024)

A **web service** is a software component that exposes functionality **over a network using standard web protocols** (HTTP, XML/JSON) so that applications on different platforms can interact. Types: **SOAP** services (XML, WSDL, UDDI) and **RESTful** services.

Components (SOAP stack): SOAP (message format), **WSDL** (describes the service interface), **UDDI** (registry for discovery), and **HTTP** transport. Working: the provider publishes its WSDL in a registry; the consumer finds the service, binds and exchanges messages. Features: **platform independence, loose coupling, reusability, interoperability and standard protocols**. Used for payments, weather, maps and enterprise integration.



Web service interaction

Web service	API
Always works **over a network** (HTTP)	May be local (library, OS) or over a network
Uses SOAP, REST, XML/JSON	Any interface (functions, classes, HTTP)
Needs a network connection	Does not always need it
Machine-to-machine communication	Interface for developers to use software
Every web service is an API	Not every API is a web service
Examples: weather SOAP service	Examples: Java API, Windows API, REST API

Conclusion: a web service is a network-based kind of API.

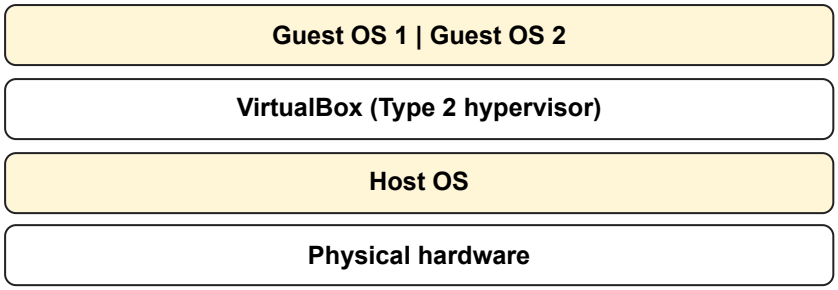
Q5. Describe VirtualBox and its use in virtualization. How does it facilitate deployment of virtual machines? (Feb 2024)

Oracle VirtualBox is a free, open-source **Type 2 hosted hypervisor** that installs on Windows, Linux or macOS and allows several **guest operating systems** to run simultaneously as VMs on one machine.

Use in virtualization: software testing, development environments, learning, running legacy applications and trying other operating systems safely.

How it facilitates VM deployment:

1. **Create VM wizard:** choose OS type, RAM, CPU cores and a virtual hard disk (VDI, VMDK).
2. **Install the guest OS** from an ISO file, then add **Guest Additions** for better graphics, clipboard and shared folders.
3. **Snapshots:** save and restore VM states.
4. **Virtual networking:** NAT, bridged, host-only and internal networks.
5. **Import/export** of appliances (OVF/OVA) to move VMs between machines.
6. **Command line (VBoxManage) and Vagrant** for automated deployment.
7. Hardware virtualization support (VT-x/AMD-V) for good performance.



VirtualBox architecture

Limitations: lower performance than bare-metal hypervisors because it relies on the host OS.

Conclusion: VirtualBox gives an easy, free way to create and manage VMs on a personal computer.

