



SKILLS | DevOps and Cloud Computing

AWS Databases, Monitoring, Deployment, and Infrastructure as Code (IaC)



Objective

- Understand Amazon RDS and its database engines, multi-AZ deployments, and read replicas.
- Learn about Amazon DynamoDB, a NoSQL key-value database for high-performance applications.
- Configure Amazon CloudWatch for metrics, alarms, and log groups to monitor AWS resources.
- Use AWS CloudTrail for event tracking and security auditing.
- Deploy applications using AWS CodeDeploy and understand deployment groups.
- Automate AWS infrastructure using AWS CloudFormation templates and stacks.





Explaining Amazon Relational Database Service (RDS) and its managed database offerings.

Let's see

Amazon RDS (Relational Database Service) is a fully managed service by AWS that makes it easy to set up, operate, and scale relational databases in the cloud. It automates tasks like provisioning, backups, patching, and monitoring.

Managed Database Offerings:

Amazon RDS supports several database engines:

- Amazon Aurora (MySQL/PostgreSQL-compatible, high performance)
- MySQL
- PostgreSQL
- MariaDB
- Oracle
- Microsoft SQL Server

Pop Quiz

Q. What is the primary purpose of Amazon RDS?

A

To provide a managed relational database service

B

To run serverless applications

Pop Quiz

Q. What is the primary purpose of Amazon RDS?

A

To provide a managed relational database service

B

To run serverless applications



**Discussing supported
database engines
(MySQL, PostgreSQL,
MariaDB, Oracle, SQL
Server).**

Let's discuss

- **MySQL:** Popular open-source database known for reliability and ease of use.
- **PostgreSQL:** Advanced open-source database with strong support for complex queries and data integrity.
- **MariaDB:** Community-developed fork of MySQL, offering similar features with some enhancements.
- **Oracle:** Enterprise-grade database with advanced features for large-scale, mission-critical applications.
- **SQL Server:** Microsoft's relational database, widely used in enterprise environments with strong integration in Windows ecosystems.





Explaining Multi-AZ deployments for high availability and automatic failover.

Let's see

Multi-AZ deployments in Amazon RDS provide high availability by automatically replicating data to a standby instance in a different Availability Zone (AZ).

- If the primary instance fails, RDS performs an automatic failover to the standby, minimizing downtime without user intervention.
- This setup ensures data durability and continuous application availability.



Pop Quiz

Q. What is the main purpose of a Multi-AZ deployment in Amazon RDS?

A

To enable global data distribution

B

To provide high availability and
automatic failover

Pop Quiz

Q. What is the main purpose of a Multi-AZ deployment in Amazon RDS?

A

To enable global data distribution

B

To provide high availability and
automatic failover



Discussing Read Replicas for scaling read-heavy workloads.

Let's discuss

Read Replicas in Amazon RDS allow you to replicate data from a primary database to one or more read-only copies.

- They help scale read-heavy workloads by distributing read traffic, improving performance without affecting the primary database.
- Read Replicas can also be promoted to standalone databases if needed.



Pop Quiz

Q. What is the primary benefit of using Amazon RDS Read Replicas?

A

Enabling global data distribution

B

Scaling out read-heavy workloads

Pop Quiz

Q. What is the primary benefit of using Amazon RDS Read Replicas?

A

Enabling global data distribution

B

Scaling out read-heavy workloads



**Explaining DynamoDB as
a key-value and
document database for
NoSQL workloads.**

Let's see

Amazon DynamoDB is a fully managed NoSQL database that supports both key-value and document data models.

- It is designed for high performance, scalability, and low-latency access, making it ideal for web, mobile, gaming, and IoT applications.
- DynamoDB handles automatic scaling, backup, and security, with no server management required.





**Discussing Tables, Items,
and Attributes and how
data is structured.**

Let's discuss

In DynamoDB:

- A Table is a collection of data.
- An Item is a single record in a table (like a row).
- An Attribute is a piece of data within an item (like a column).

Each item is uniquely identified by a primary key, and data is stored in a flexible, schema-less format.





**Demonstrating how to
query and scan data
efficiently.**

Let's do it

Query (Efficient for Key-Based Access)

```
response = table.query(  
    KeyConditionExpression=Key('user_id').eq('123')  
)
```

Efficient because:

- Uses primary key or secondary index
- Retrieves only matching items
- Supports pagination and projections to limit data size



Let's do it

Scan (Less Efficient, Use with Caution)

```
response = table.scan(  
    FilterExpression=Attr('status').eq('active'),  
    ProjectionExpression="user_id, status"  
)
```

Less efficient because:

- Reads every item in the table
- Use FilterExpression to reduce results
- Use ProjectionExpression to return only necessary attributes
- Use pagination to handle large datasets (LastEvaluatedKey)





Explaining CloudWatch as a monitoring and observability tool.

Let's see

Amazon CloudWatch is a monitoring and observability tool that collects and tracks metrics, logs, and events from AWS resources and applications.

- It helps you visualize performance, set alarms, detect anomalies, and automate responses, enabling proactive system monitoring and troubleshooting.



Pop Quiz

Q. Which feature of CloudWatch is used to collect and store log data from applications?

A

CloudWatch Metrics

B

CloudWatch Logs

Pop Quiz

Q. Which feature of CloudWatch is used to collect and store log data from applications?

A

CloudWatch Metrics

B

CloudWatch Logs



**Demonstrating how to
track AWS service
metrics (CPU, memory,
network usage).**

Let's do it

In Amazon CloudWatch, you can track AWS service metrics like CPU, memory, and network usage using:

Built-in metrics (e.g., for EC2):

- CPUUtilization
- NetworkIn/Out
- DiskReadOps/WriteOps



Let's do it

Example (using AWS Console or CLI):

```
aws cloudwatch get-metric-statistics \
--namespace AWS/EC2 \
--metric-name CPUUtilization \
--dimensions Name=InstanceId,Value=i-1234567890abcdef0 \
--start-time 2025-05-29T00:00:00Z \
--end-time 2025-05-30T00:00:00Z \
--period 300 \
--statistics Average
```





**Discussing Alarms and
automatic actions (e.g.,
Auto Scaling trigger on
high CPU).**

Let's discuss

CloudWatch Alarms monitor metrics and trigger actions when thresholds are crossed.

- For example, an alarm on high CPU usage can automatically trigger Auto Scaling to add more instances, helping maintain performance and availability without manual intervention.





**Showing how to organize
logs into log groups and
create dashboards.**

Let's see

- **Organize Logs:** Create Log Groups in CloudWatch to group related logs (e.g., by app or service). Logs from resources go into these groups as Log Streams.
- **Create Dashboards:** In CloudWatch, create a Dashboard, add widgets to visualize metrics (CPU, errors, etc.), and link data from log groups or metrics for real-time monitoring.





Take A 5-Minute Break!



- Stretch and relax
- Hydrate
- Clear your mind
- Be back in 5 minutes





Explaining CloudTrail's role in tracking AWS API calls and security auditing.

Let's see

AWS CloudTrail records all API calls and actions across your AWS account.

- It enables security auditing, compliance, and troubleshooting by capturing details like who did what, when, and from where, helping detect unauthorized access or changes.



Pop Quiz

Q. Where are CloudTrail logs typically stored for long-term auditing?

A

Amazon S3

B

Amazon RDS

Pop Quiz

Q. Where are CloudTrail logs typically stored for long-term auditing?

A

Amazon S3

B

Amazon RDS



**Discussing how
CloudTrail logs API
events and delivers them
to S3 for storage.**

Let's discuss

AWS CloudTrail logs API events (e.g., user actions, service calls) and delivers them as log files to an Amazon S3 bucket for secure storage.

- These logs can be used for auditing, analysis, and integrating with tools like Amazon Athena or CloudWatch Logs for deeper insights.





**Demonstrating how to
search CloudTrail logs for
security and operational
insights.**

Let's do it

To search CloudTrail logs for security and operational insights:

- Use CloudTrail Event History in the AWS Console to filter by user, event name, resource, or time range.

For advanced queries, send logs to CloudWatch Logs or S3, then:

- Use CloudWatch Logs Insights to run queries.
- Use Amazon Athena to run SQL queries on S3 logs.





Explaining AWS CodeDeploy and how it automates deployments to EC2, Lambda, and on-premises servers.

Let's see

AWS CodeDeploy is a fully managed service that automates application deployments across various compute platforms, including Amazon EC2, AWS Lambda, and on-premises servers.

How CodeDeploy Works

- 1. Deployment Package:** You define the application content and instructions in an AppSpec file (YAML or JSON).
- 2. Deployment Groups:** CodeDeploy uses these to manage target resources (EC2 instances, Lambda functions, or on-prem servers).
- 3. Deployment Types:**
 - In-place deployments (for EC2/on-prem): Code is updated on the same instance.
 - Blue/Green deployments (for EC2/Lambda): New version is deployed to new instances; traffic is shifted after testing.



**Discussing deployment
strategies (in-place,
blue/green, rolling
updates).**

Let's discuss

Deployment Strategies:

- **In-place:** Updates the existing instances with new code. Simple but may cause downtime.
- **Blue/Green:** Deploys new code to a separate environment (green), then shifts traffic from the old (blue) once tested. Minimizes risk and downtime.
- **Rolling Updates:** Gradually updates a few instances at a time. Balances availability and risk.





Introducing deployment groups and configurations.

Let's introduce

Deployment Groups in AWS CodeDeploy define which targets (EC2, Lambda, or on-prem servers) receive the deployment. They group resources by tags, names, or Auto Scaling groups.

Deployment Configurations control how the deployment proceeds, such as:

- AllAtOnce – deploy to all targets at once
- HalfAtATime – deploy to half the targets, then the rest
- OneAtATime – deploy to one target at a time





Explaining Infrastructure as Code (IaC) and why automation is essential.

Let's see

Infrastructure as Code (IaC) is the practice of managing and provisioning infrastructure using code instead of manual processes.

- It enables automation, consistency, version control, and faster deployments, reducing human error and improving scalability and reliability in cloud environments.





**Discussing
CloudFormation
templates (JSON/YAML)
and stack creation.**

Let's discuss

AWS CloudFormation uses JSON or YAML templates to define and provision AWS resources as code.

- You create a stack from a template, which automatically sets up and manages the defined resources (e.g., EC2, S3, VPC), enabling repeatable and consistent infrastructure deployment.



Pop Quiz

Q. What is the primary purpose of AWS CloudFormation?

A

To create and manage AWS resources using code

B

To monitor AWS services in real time

Pop Quiz

Q. What is the primary purpose of AWS CloudFormation?

A

To create and manage AWS resources using code

B

To monitor AWS services in real time



**Demonstrating deploying
AWS resources using
CloudFormation
templates.**

Let's do it

1. Write a template in JSON or YAML defining resources (e.g., an EC2 instance).
2. Use the AWS CLI to create a stack:

```
aws cloudformation create-stack --stack-name MyStack --template-body file://template.yaml
```

3. CloudFormation provisions the resources automatically based on the template.





Time for case study!

Important

- Complete the post-class assessment
- Complete assignments (if any)
- Practice the concepts and techniques taught in this session
- Review your lecture notes
- Note down questions and queries regarding this session and consult the teaching assistants



Thanks



!

