**Amazon DB**

**What is the difference between OLTP and OLAP ?**

OLTP is an Online Transaction Processing system. The main focus of OLTP system is to record the current Update, Insertion and Deletion while transaction. The OLTP queries are simpler and short and hence require less time in processing, and also requires less space.

OLTP database has **normalized tables** (3NF).

OLAP is an Online Analytical Processing system. OLAP database stores historical data that has been inputted by OLTP. It allows a user to view different summaries of multi-dimensional data. Using OLAP, you can extract information from a large database and analyze it for decision making.

The transaction in OLAP are long and hence take comparatively more time for processing and requires large space. The transactions in OLAP are less frequent as compared to OLTP. Even the tables in OLAP database may not be normalized.

Database warehousing applications use OLAP.



**What is the rule of security group when we use RDS ?**

Suppose we developed an application in an EC2 instance, and trying to connect RDS instance.

Remember RDS instance is using security group. If EC2 instance and RDS instance are using different security group , we need to bound the security groups together otherwise communication cannot happen. To do that go to RDS security group and inbound tab and attach the other security group in ip address text area. Type sg to find already added security groups. This is only applicable when RDS and EC2 instance both are in same vpc of the same aws account.

**What are the rules for VPC and subnet group ?**

The vpc which RDS instance resides in should have atleast two subnets which are public and the vpc should have an internet gateway.

**A DB Instance in a VPC Accessed by an EC2 Instance Not in a VPC**

You can communicate between an Amazon RDS DB instance that is in a VPC and an EC2 instance that is not in an Amazon VPC by using *ClassicLink*. When you use Classic Link, an application on the EC2 instance can connect to the DB instance by using the endpoint for the DB instance. ClassicLink is available at no charge.

The following diagram shows this scenario.


     A DB Instance in a VPC Accessed by an EC2 Instance Not in a VPC
    

Using ClassicLink, you can connect an EC2 instance to a logically isolated database where you define the IP address range and control the access control lists (ACLs) to manage network traffic. You don't have to use public IP addresses or tunneling to communicate with the DB instance in the VPC. This arrangement provides you with higher throughput and lower latency connectivity for inter-instance communications.

**To enable ClassicLink between a DB instance in a VPC and an EC2 instance not in a VPC**

1. Sign in to the AWS Management Console and open the Amazon VPC console at <https://console.aws.amazon.com/vpc>.
2. In the navigation pane, choose **Your VPCs**.
3. Choose the VPC used by the DB instance.
4. In **Actions**, choose **Enable ClassicLink**. In the confirmation dialog box, choose **Yes, Enable**.
5. On the EC2 console, choose the EC2 instance you want to connect to the DB instance in the VPC.
6. In **Actions**, choose **ClassicLink**, and then choose **Link to VPC**.
7. On the **Link to VPC** page, choose the security group you want to use, and then choose **Link to VPC**.

**How to add encryption in RDS after creation ?**

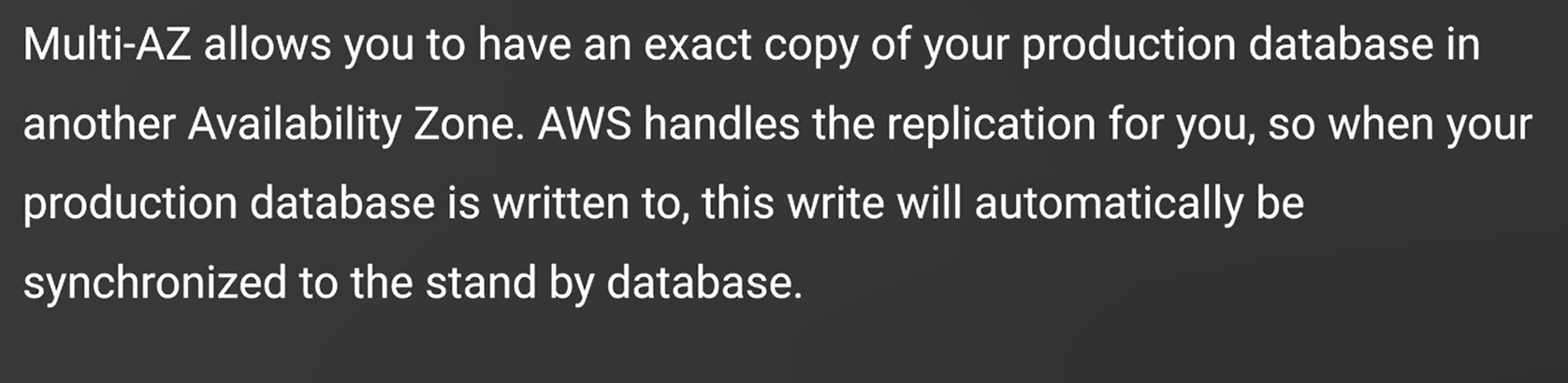
Just like EC2 create a snapshot of the RDS instance, Go to snapshot and select copy snapshot action and then you can modify encryption .

**Difference between Automatic backup and Db snapshots**

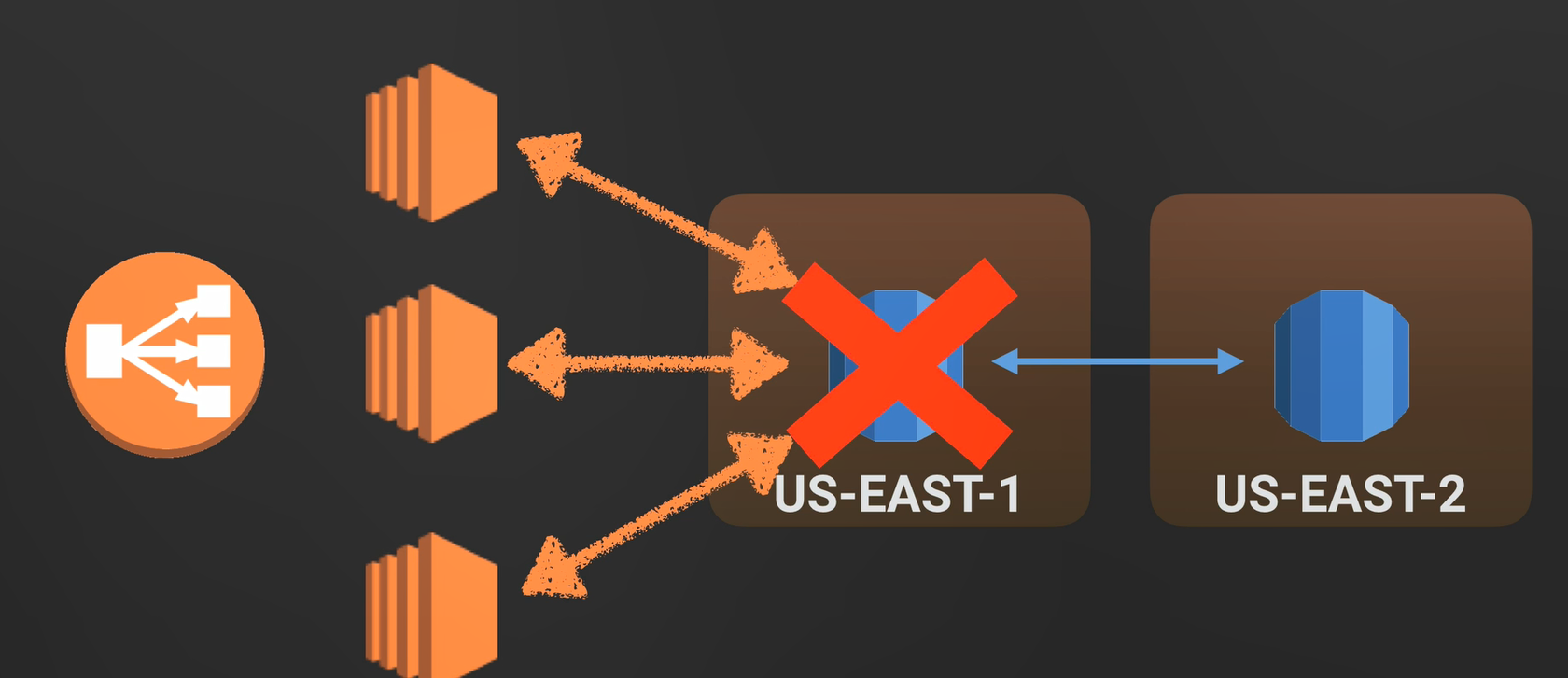
If we turn on automatic backups aws will take DB snapshots in regular interval or mentioned interval. This is a point in time backups. Automated backup will allow you to recover your database to any point in time within retention period, retention period can be between 1 to 35 days. When we do a recovery AWS will choose the recent snapshot. It will take very few seconds. Snapshots are stored in S3.

DB snapshots are like EBS snapshots, we can take DB snapshots anytime. If we delete DB instance DB snapshots will not be deleted but snapshots related to automated backup are deleted automatically.

**What is Multi-AZ RDS ?**

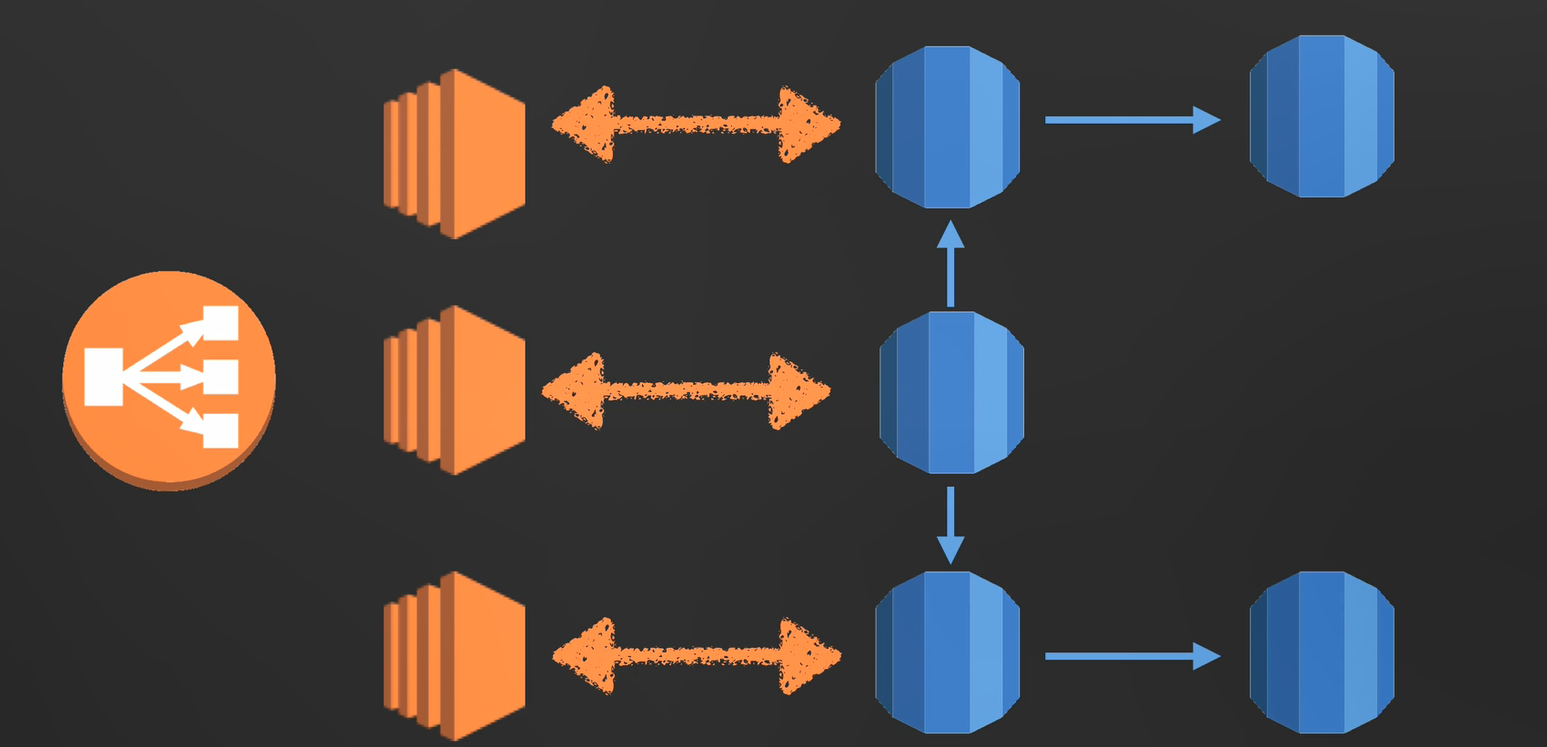


Multi availability zone is only used for disaster recovery, not for performance. Multi AZ is synchronous.

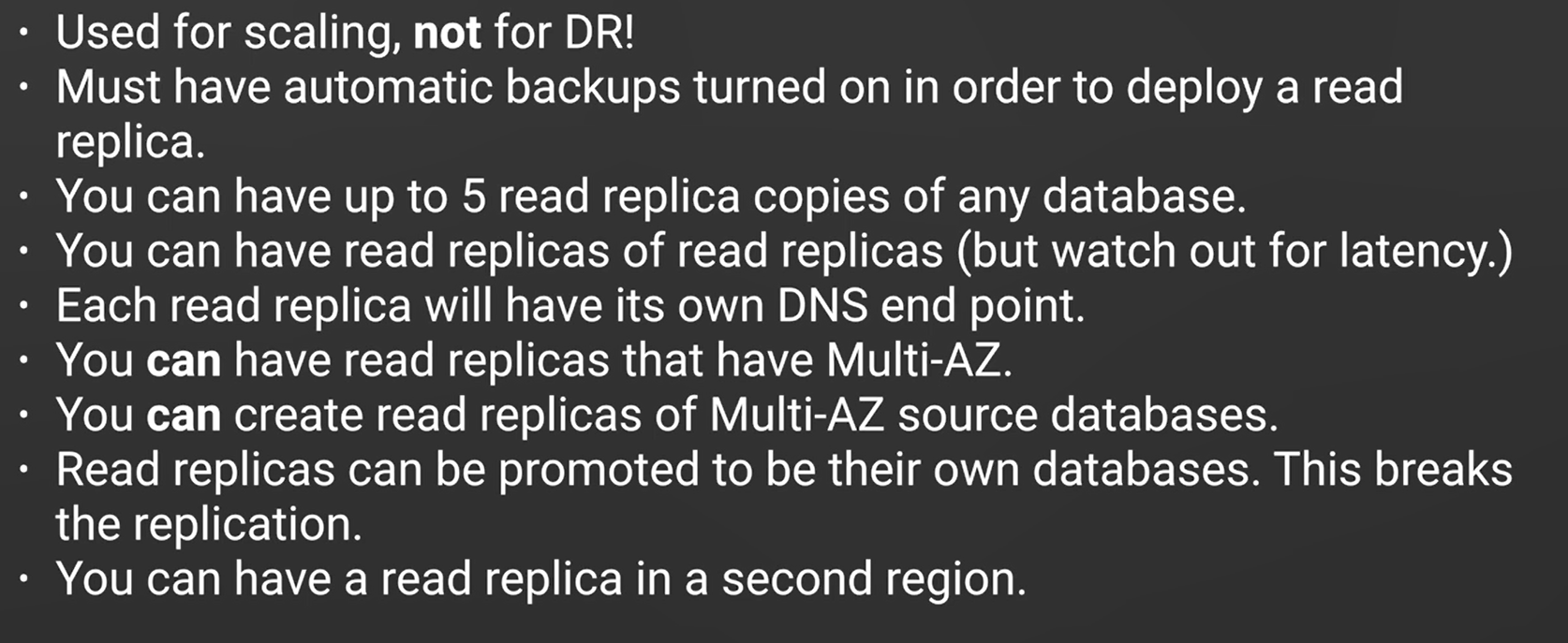


**Read replica ?**





**Properties of read replica**



**How read replica helps scaling and performance improvement ?**

When you create a Read Replica, you first specify an existing DB instance as the source. Then Amazon RDS takes a snapshot of the source instance and creates a read-only instance from the snapshot. Amazon RDS then uses the asynchronous replication method for the DB engine to update the Read Replica whenever there is a change to the source DB instance. The Read Replica operates as a DB instance that allows only read-only connections. Applications connect to a Read Replica the same way they do to any DB instance. Amazon RDS replicates all databases in the source DB instance.

In some cases, a Read Replica resides in a different AWS Region than its source DB instance. In these cases, Amazon RDS sets up a secure communications channel between the source DB instance and the Read Replica. Amazon RDS establishes any AWS security configurations needed to enable the secure channel, such as adding security group entries.

**If you are using a non-transactional engine such as My ISAM, you will need to perform the following steps**

To successfully set up your Read Replica. These steps are required in order to ensure that the Read Replica has a consistent copy of your data. 1. Stop all DML and DDL operations on non-transactional tables and wait for them to complete. 2. Flush and lock those tables. 3. Create the Read Replica using the Create DB instance Read Replica API. 4. Check the progress of the Replica creation using the describe DB instances API

**Use of DB parameter group**

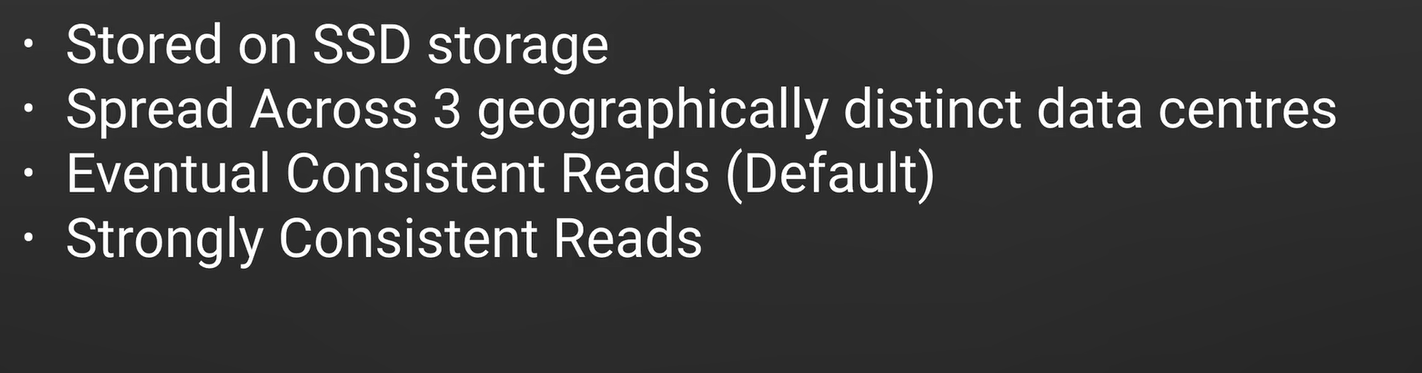
Default DB parameter group is added with a DB instance automatically when we create any DB instance. We cannot edit this default db parameter group rather we can create a new group and configure different parameter setting and attach it to a db instance.

**Will the db parameter group parameter has immediate effect ?**

All the dynamic parameters has immediate effect, but for static parameters we need to reboot the instance.

**Properties of Dynamo DB ?**

Dynamo DB is a fast and flexible NoSqL database service for all applications that need consistent, single digit milli-second latency at any scale.



**Eventually Consistent Reads**

When you read data from a DynamoDB table, the response might not reflect the results of a recently completed write operation. The response might include some stale data. If you repeat your read request after a short time, the response should return the latest data.

**Strongly Consistent Reads**

When you request a strongly consistent read, DynamoDB returns a response with the most up-to-date data, reflecting the updates from all prior write operations that were successful. A strongly consistent read might not be available if there is a network delay or outage. Strongly consistent reads are not supported on global secondary indexes.

DynamoDB uses eventually consistent reads, unless you specify otherwise. Read operations (such as GetItem, Query, and Scan) provide a ConsistentRead parameter. If you set this parameter to true, DynamoDB uses strongly consistent reads during the operation.

**What is read and write capacity ?**

When we create a dynamo db we need to select read capacity and write capacity unit. It means

If read capacity is 1 then dynamo db can read 1 unit per second. If 2 then read 2 units per second, same for the write capacity. Depending on the unit price will be calculated.

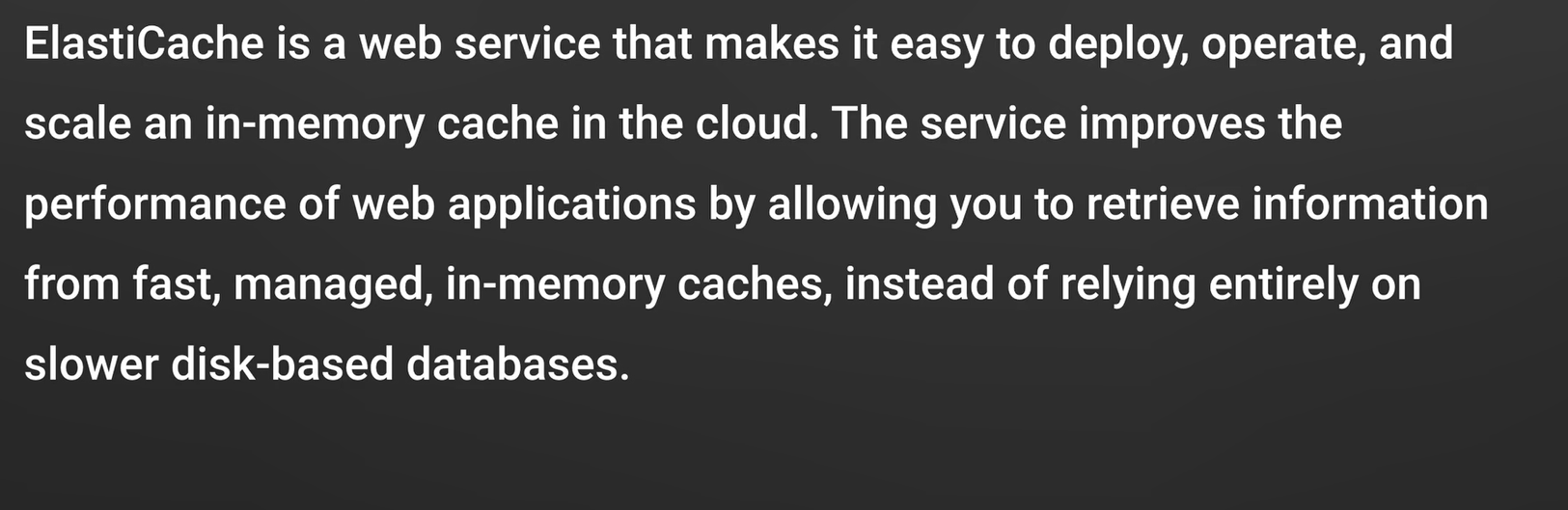
**What is one read write capacity units denote in dynamo db ?**

2 eventual consistent reads/ unit or 1 strongly consistent reads/unit

**What is dynamodb streams**

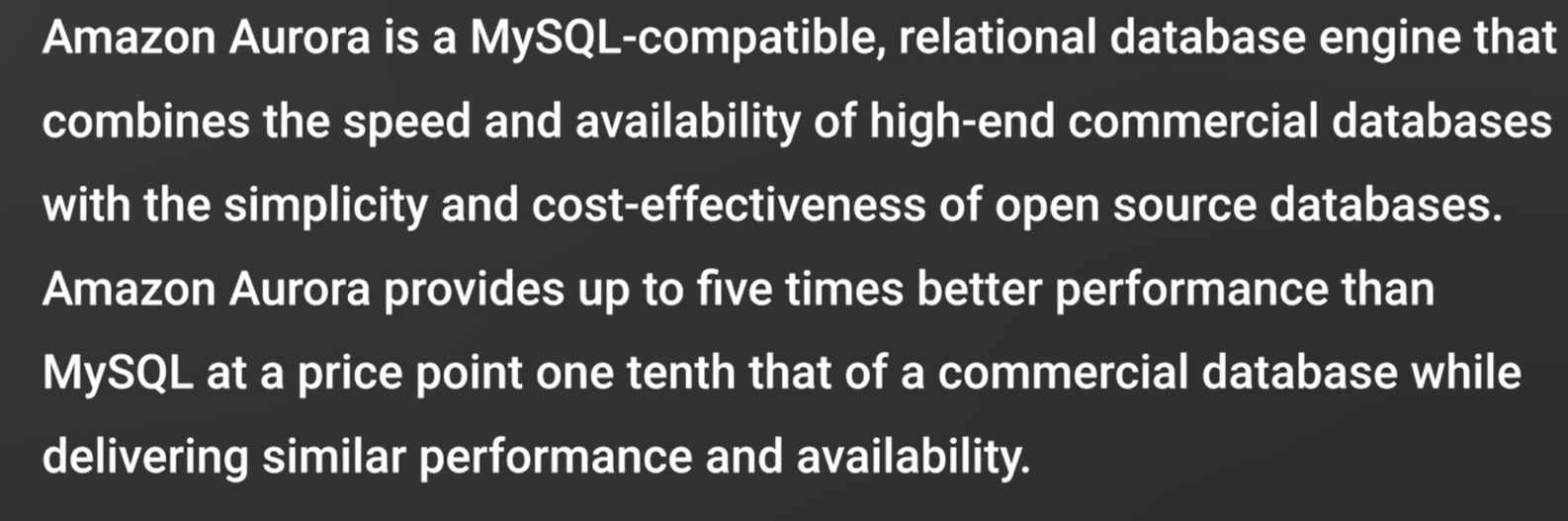
It captures data modification events in dynamodb table. Its lifetime is 24 hours. Example if we add a new customer info in dynamo db. Dunamodb stream a new email. It maintains the order as well.

**What is Elastic cache ?**

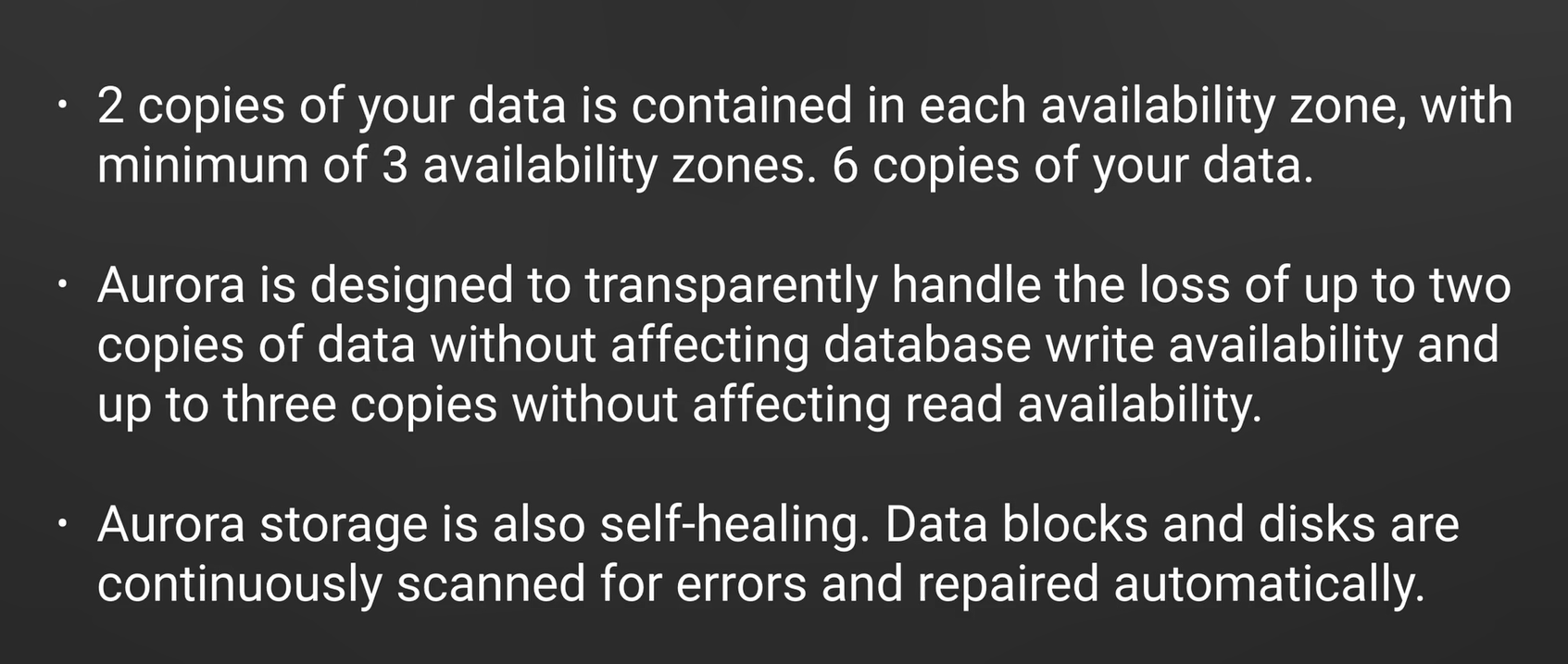


There are two types of elastic cache available. Redis and Memcached.

**What is Aurora ?**



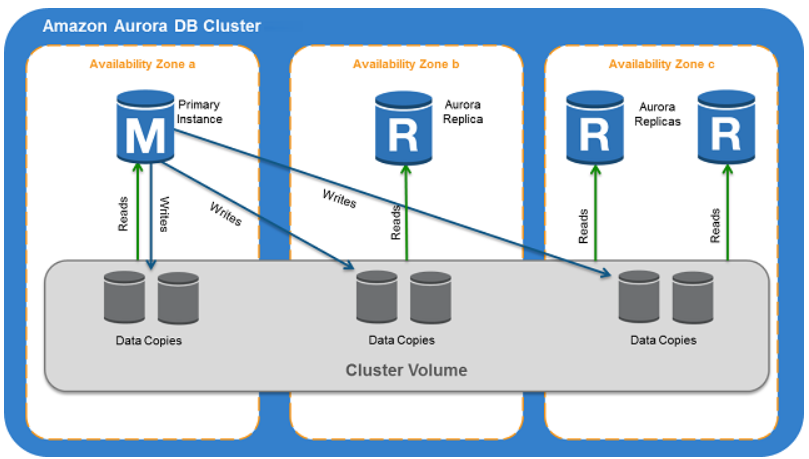
**What is aurora scaling ?**



**What is aurora server less strategy**

Aurora Server less has a pause and resume capability to automatically stop the database compute capacity after a specified period of inactivity. When paused, you are charged only for storage. It automatically resumes when new database connections are requested

**Aurora architecture**



**How disaster recovery done for other RDS systems and aurora**

For disaster recovery other RDS system store another instance in a different availability zone and incase of failure RDS will point to the backup instance. In case of aurora there are one primary instance and one or one read replicas to involve in disaster recovery. In case of failure one of the read replica which is in the different availability zone become a primary instance.

**What are the database types accepted by Aurora**

Aurora is accepted by mysql and posgresql. So we do not need to install any aurora specific driver if we have mysql and postgres sql.

**What is Redshift ?**

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. You can start with just a few hundred gigabytes of data and scale to a petabyte or more. This enables you to use your data to acquire new insights for your business and customers.

The first step to create a data warehouse is to launch a set of nodes, called an Amazon Redshift cluster. After you provision your cluster, you can upload your data set and then perform data analysis queries. Regardless of the size of the data set, Amazon Redshift offers fast query performance using the same SQL-based tools and business intelligence applications that you use today

**How encryption managed in Redshift ?**

Redshift comes with automatic encryption.

**Why Dynamo DB is better than RDS incase of downtime ?**

Dynamo db offers pushdown scaling , means we can scale our database on the fly, without any downtime. RDS is not so easy and we usually have to use a bigger instance size or add replica.

**what is EMR and what is S3DistCp ?**

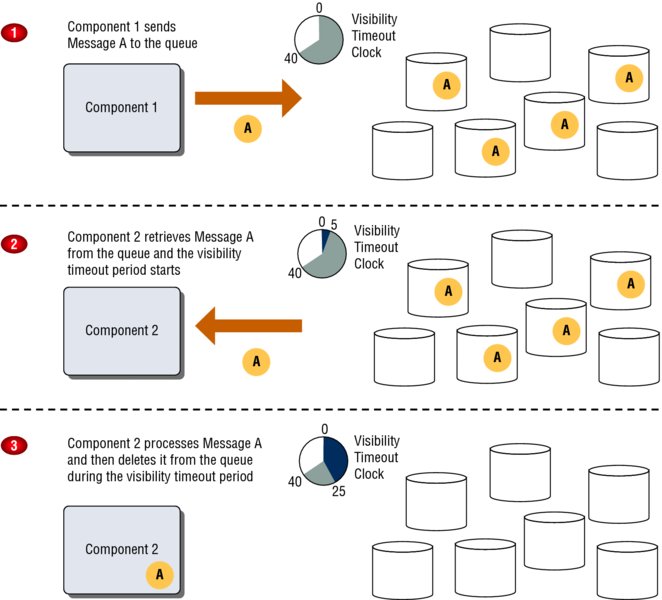
EMR is elastic map reduce which is a distributed HDFS system in aws.

Using S3DistCp, you can efficiently copy large amounts of data from Amazon S3 into HDFS where it can be processed by subsequent steps in your Amazon EMR cluster. You can also use S3DistCp to copy data between Amazon S3 buckets or from HDFS to Amazon S3. S3DistCp is more scalable and efficient for parallel copying large numbers of objects across buckets and across AWS accounts.

**What is SQS ?**

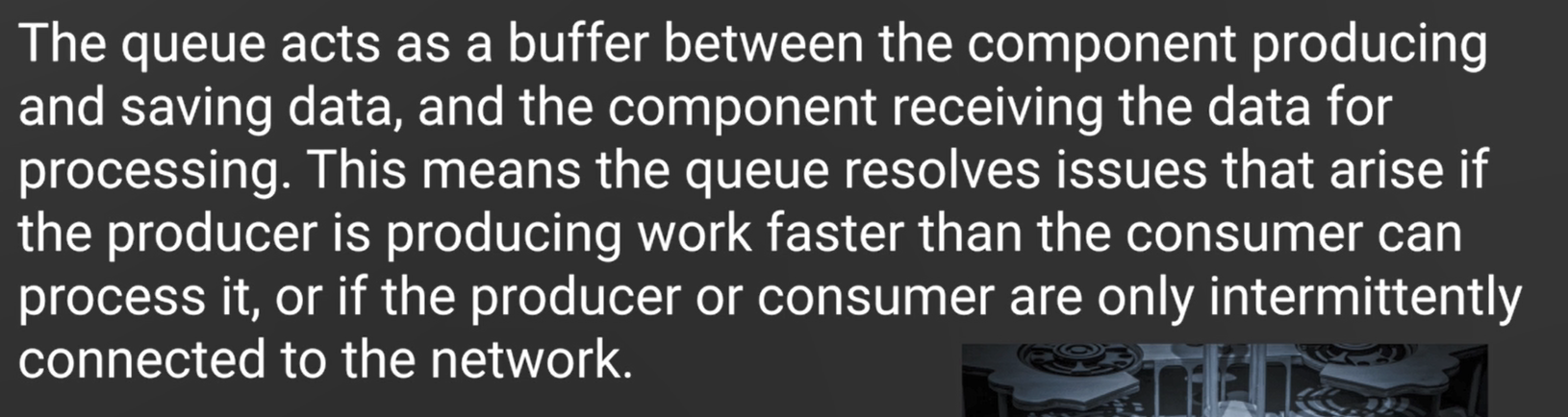
Amazon SQS is a fast, reliable, scalable, and fully managed message queuing service. Amazon SQS makes it simple and cost effective to decouple the components of a cloud application. You can use Amazon SQS to transmit any volume of data, at any level of throughput, without losing messages or requiring other services to be continuously available. Amazon SQS ensures delivery of each message at least once and supports multiple readers and writers interacting with the same queue. A single queue can be used simultaneously by many distributed application components. Amazon SQS is engineered to be highly available and to deliver messages reliably and efficiently; however, the service does not guarantee First In, First Out (FIFO) delivery of messages

**SQS message lifecycle**



1. Component 1 sends Message A to a queue, and the message is redundantly distributed across the Amazon SQS servers.
2. When Component 2 is ready to process a message, it retrieves messages from the queue, and Message A is returned. While Message A is being processed, it remains in the queue and is not returned to subsequently receive requests for the duration of the visibility timeout.
3. Component 2 deletes Message A from the queue to prevent the message from being received and processed again after the visibility timeout expires.

**When to use SQS ?**



**Different types of SQS ?**

* Standard queue : It is a default type of SQS, order are not maintained, duplicate order can be delivered, but it can guarantee that message can be delivered atleast once without any failure.
* FIFO : This is a first in first out queue, order is maintained, FIFO queue is limited to 300 transactions per second. Duplicates are not introduced into the queue.

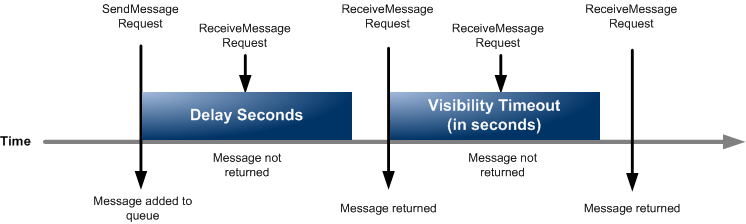
**What is SQS visibility timeouts**

Visibility timeout is a period of time during which Amazon SQS prevents other components from receiving and processing a message because another component is already processing it. By default, the message visibility timeout is set to 30 seconds, and the maximum that it can be is 12 hours.

**What is delay queue ?**

*Delay queues* allow you to postpone the delivery of new messages in a queue for a specific number of seconds.

Delay queues are similar to visibility timeouts in that both features make messages unavailable to consumers for a specific period of time. The difference is that a delay queue hides a message when it is first added to the queue, whereas a visibility timeout hides a message only after that message is retrieved from the queue.



**Queues and message identity**

Amazon SQS uses three identifiers that you need to be familiar with: queue URLs, message IDs, and receipt handles.

When creating a new queue, you must provide a queue name that is unique within the scope of all of your queues. Amazon SQS assigns each queue an identifier called a *queue URL*, which includes the queue name and other components that Amazon SQS determines. Whenever you want to perform an action on a queue, you must provide its queue URL.

Each time you receive a message from a queue, you receive a receipt handle for that message. The handle is associated with the act of receiving the message, not with the message itself. As stated previously, to delete the message or to change the message visibility, you must provide the receipt handle and not the message ID. This means you must always receive a message before you can delete it

**What is long polling in SQS ?**

When your application queries the Amazon SQS queue for messages, it calls the function ReceiveMessage. ReceiveMessage will check for the existence of a message in the queue and return immediately, either with or without a message. If your code makes periodic calls to the queue, this pattern is sufficient. If your SQS client is just a loop that repeatedly checks for new messages, however, then this pattern becomes problematic, as the constant calls to ReceiveMessage burn CPU cycles and tie up a thread.

**What is dead letter queue ?**

Amazon SQS provides support for *dead letter queues*. A dead letter queue is a queue that other (source) queues can target to send messages that for some reason could not be successfully processed. A primary benefit of using a dead letter queue is the ability to sideline and isolate the unsuccessfully processed messages. You can then analyze any messages sent to the dead letter queue to try to determine the cause of failure.

**What is SWF (Simple work flow service)**

Amazon SWF gives you full control over implementing and coordinating tasks without worrying about underlying complexities such as tracking their progress and maintaining their state.

Using Amazon SWF, you can implement distributed, asynchronous applications as *workflows*. Workflows coordinate and manage the execution of activities that can be run asynchronously across multiple computing devices and that can feature both sequential and parallel processing.

**What is workflow history ?**

The workflow history is a detailed, complete, and consistent record of every event that occurred since the workflow execution started. Workflows in aws are stateless, to maintain the status of the entire workflow in every componenets, we need to maintain workflow history.

**What is workflow domain ?**

Domains provide a way of scoping Amazon SWF resources within your AWS account. You must specify a domain for all the components of a workflow, such as the workflow type and activity types. It is possible to have more than one workflow in a domain; however, workflows in different domains cannot interact with one another.

**What is workflow actors ?**

Amazon SWF consists of a number of different types of programmatic features known as *actors*. Actors can be workflow starters, deciders, or activity workers. These actors communicate with Amazon SWF through its API. You can develop actors in any programming language.

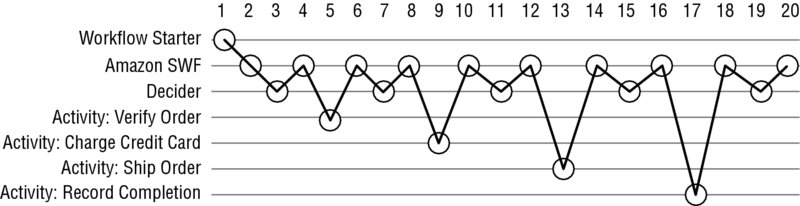
* **Starter actor :** A workflow starter is any application that can initiate workflow executions. For example, one workflow starter could be an e-commerce website where a customer places an order. Another workflow starter could be a mobile application where a customer orders takeout food or requests a taxi.
* **Decider actor :** Activities within a workflow can run sequentially, in parallel, synchronously, or asynchronously. The logic that coordinates the tasks in a workflow is called the *decider*
* **Activity workers :** An *activity worker* is a single computer process (or thread) that performs the activity tasks in your workflow. Different types of activity workers process tasks of different activity types, and multiple activity workers can process the same type of task.

**What is tasks ?**

Amazon SWF provides activity workers and deciders with work assignments, given as one of three types of tasks: activity tasks, AWS Lambda tasks, and decision tasks.

**Lifecycle of a Workflow Execution**

From the start of a workflow execution to its completion, Amazon SWF interacts with actors by assigning them appropriate tasks: either activity tasks or decision tasks.

****

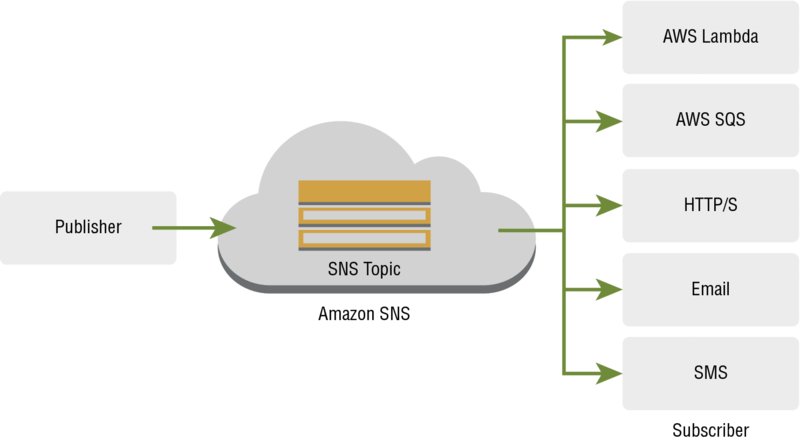
**The following 20 steps describe the workflow detailed in figure**

1. A workflow starter calls an Amazon SWF action to start the workflow execution for an order, providing order information.
2. Amazon SWF receives the start workflow execution request and then schedules the first decision task.
3. The decider receives the task from Amazon SWF, reviews the history, and applies the coordination logic to determine that no previous activities occurred. It then makes a decision to schedule the Verify Order activity with the information the activity worker needs to process the task and returns the decision to Amazon SWF.
4. Amazon SWF receives the decision, schedules the Verify Order activity task, and waits for the activity task to complete or time out.
5. An activity worker that can perform the Verify Order activity receives the task, performs it, and returns the results to Amazon SWF.
6. Amazon SWF receives the results of the Verify Order activity, adds them to the workflow history, and schedules a decision task.
7. The decider receives the task from Amazon SWF, reviews the history, applies the coordination logic, makes a decision to schedule a Charge Credit Card activity task with information the activity worker needs to process the task, and returns the decision to Amazon SWF.
8. Amazon SWF receives the decision, schedules the Charge Credit Card activity task, and waits for it to complete or time out.
9. An activity worker activity receives the Charge Credit Card task, performs it, and returns the results to Amazon SWF.
10. Amazon SWF receives the results of the Charge Credit Card activity task, adds them to the workflow history, and schedules a decision task.
11. The decider receives the task from Amazon SWF, reviews the history, applies the coordination logic, makes a decision to schedule a Ship Order activity task with the information the activity worker needs to perform the task, and returns the decision to Amazon SWF.
12. Amazon SWF receives the decision, schedules a Ship Order activity task, and waits for it to complete or time out.
13. An activity worker that can perform the Ship Order activity receives the task, performs it, and returns the results to Amazon SWF.
14. Amazon SWF receives the results of the Ship Order activity task, adds them to the workflow history, and schedules a decision task.
15. The decider receives the task from Amazon SWF, reviews the history, applies the coordination logic, makes a decision to schedule a Record Completion activity task with the information the activity worker needs, performs the task, and returns the decision to Amazon SWF.
16. Amazon SWF receives the decision, schedules a Record Completion activity task, and waits for it to complete or time out.
17. An activity worker Record Completion receives the task, performs it, and returns the results to Amazon SWF.
18. Amazon SWF receives the results of the Record Completion activity task, adds them to the workflow history, and schedules a decision task.
19. The decider receives the task from Amazon SWF, reviews the history, applies the coordination logic, makes a decision to close the workflow execution, and returns the decision along with any results to Amazon SWF.
20. Amazon SWF closes the workflow execution and archives the history for future reference.

**What is aws SNS(Simple notification service) ?**

Amazon SNS follows the publish-subscribe (pub-sub) messaging paradigm, with notifications being delivered to clients using a push mechanism that eliminates the need to check periodically (or poll) for new information and updates.

When you send a message to a topic, it is automatically forwarded to each subscriber of that topic using the communication method configured for that subscriber.



**What is fanout scenario in SNS ?**

A fanout scenario is when an Amazon SNS message is sent to a topic and then replicated and pushed to multiple Amazon SQS queues, HTTP endpoints, or email addresses This allows for parallel asynchronous processing



**What are the different services SNS can perform ?**

* Application and system alert, which is already being used in the queue like alarm system.
* Push email and text messaging
* Mobile push notification

**Different protocols used for SNS ?**

You can use the following protocols with Amazon SNS: HTTP, HTTPS, SMS, email, email-JSON, Amazon SQS, and AWS Lambda.

**Can we create a FIFO queue in SNS ?**

Unlike SQS we can create a FIFO or ordered message queue in SNS.

**What is ARN(Amazon Resource name) ?**

When we create a topic in SNS a unique ARN name is specified immediately.

**What is SES?**

Amazon Simple Email Service (Amazon **SES**) is a cloud-based email sending service designed to help digital marketers and application developers send marketing, notification, and transactional emails

**Relation of SES and SNS ?**

Amazon **SES** Notifications through Amazon **SNS**. You can configure Amazon **SES** to notify an Amazon **SNS** topic when you receive bounces or complaints, or when emails are delivered. Amazon **SNS** notifications are in JavaScript Object Notation (JSON) format, which enables you to process them programmatically.

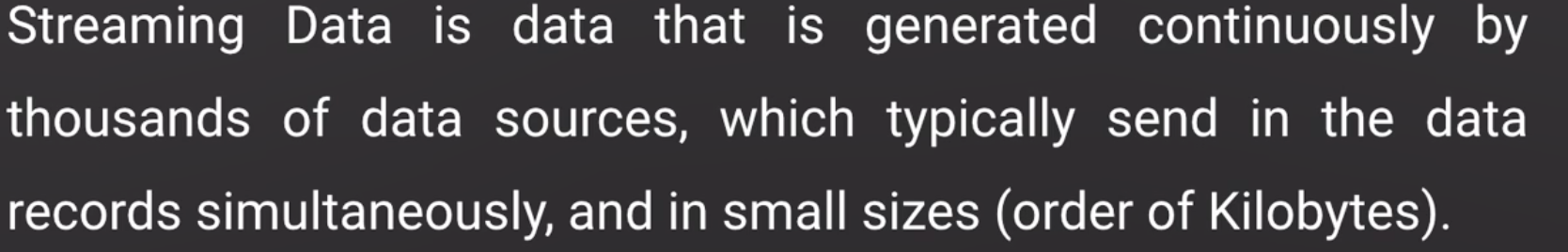
**What is SPF(Sender policy framework ) and DKIM ?**

SPF is an email validation framework. It is designed to combat email spoofing. SPF is automatically enabled with SES. If you are using your own domain then additionally add SPF in DNS server.

Domains key identified mail (DKIM) is an email protection system which protects against unauthorized tampering of messages during transit. SES automatically add DKIM, to configure your own domain create your own DKIM.

Use DMARC to protect your email notification with both SPF and DKIM.

**What is streaming data?**



**Example of streaming data ?**

Stock prices

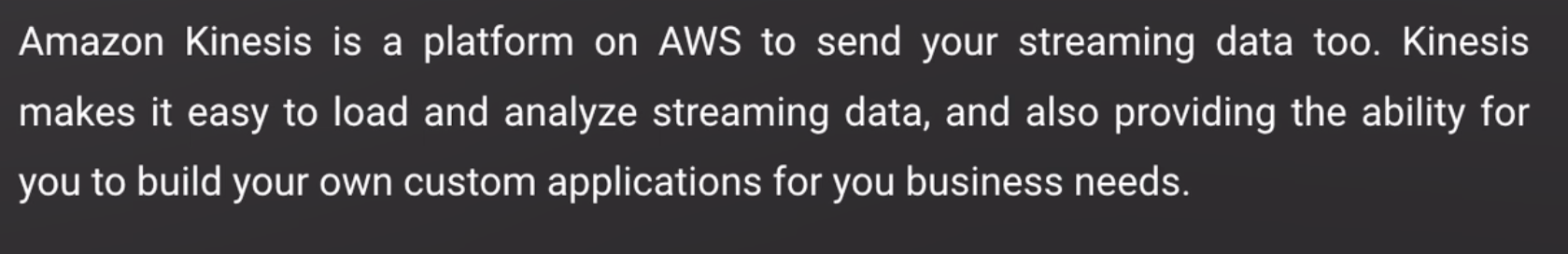
Game data

Social network data

Geospatial data

iot sensor data

**What is kinesis ?**

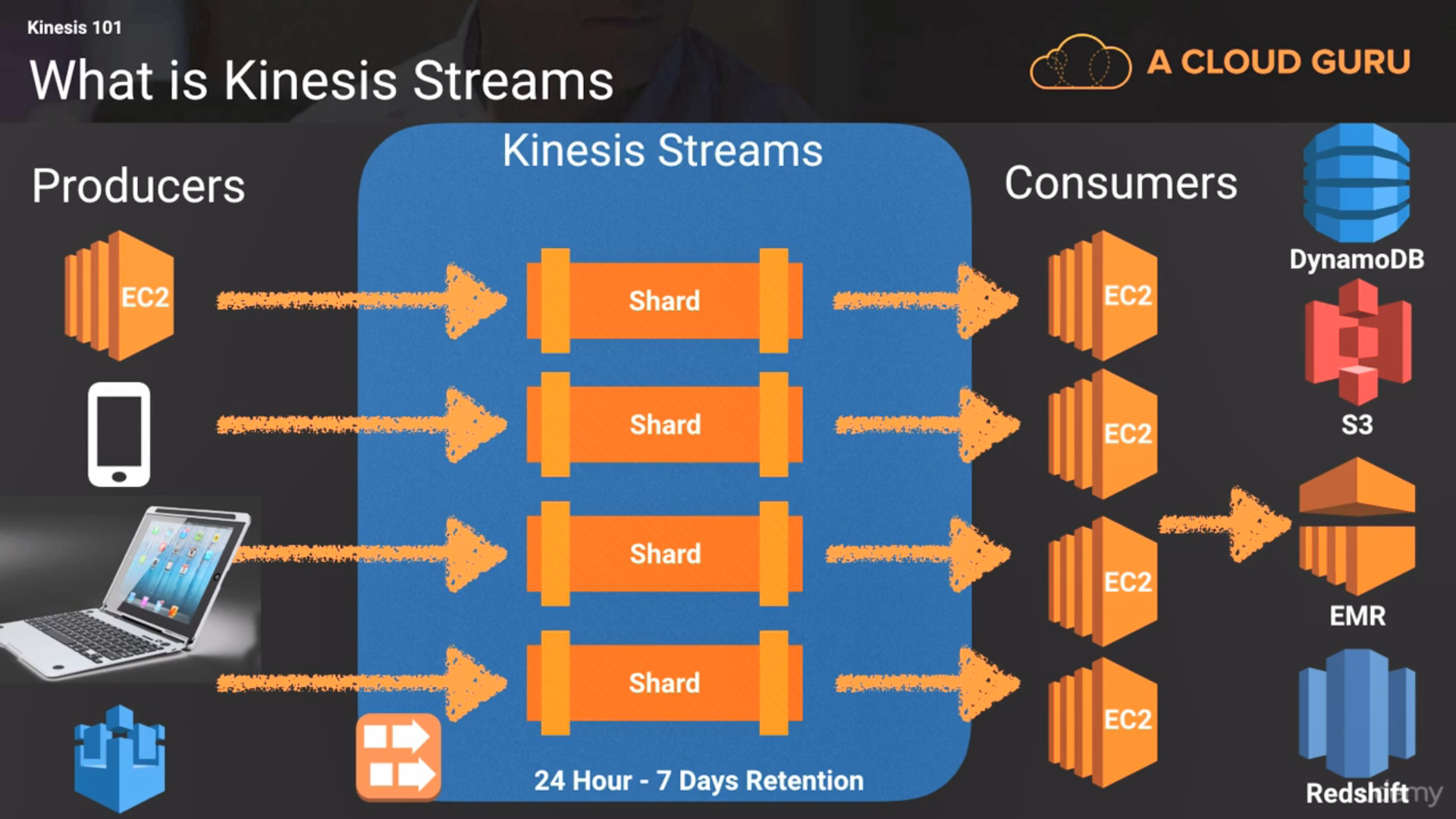


**What are the different kind of kinesis system ?**

* Kinesis stream
* Kinesis firehose
* Kinesis analytics

**What is Kinesis stream ?**

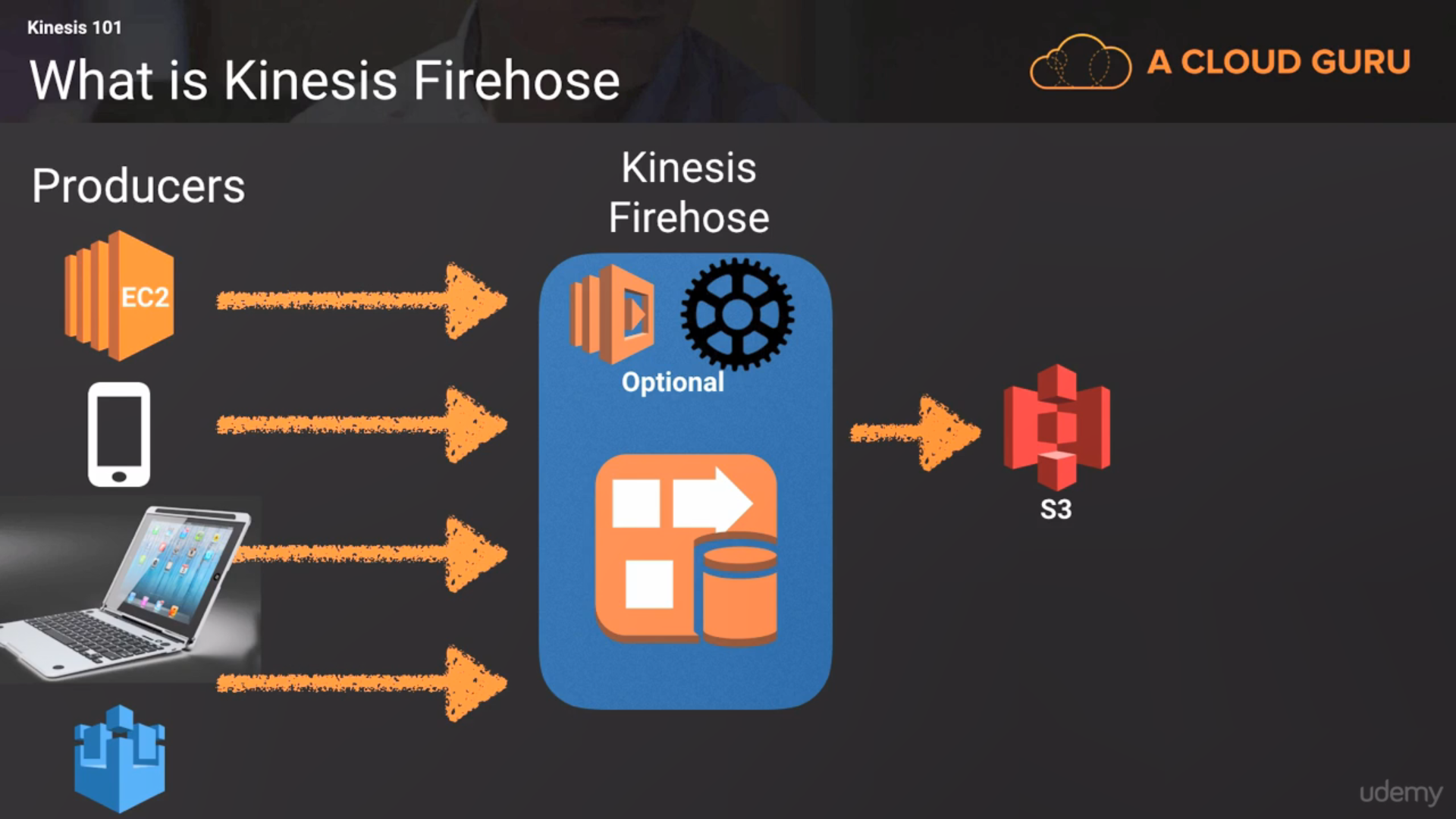
You can use Amazon Kinesis Data Streams to collect and process large [streams](https://aws.amazon.com/streaming-data/) of data records in real time. You can create data-processing applications, known as Kinesis Data Streams applications. A typical Kinesis Data Streams application reads data from a data stream as data records. These applications can use the Kinesis Client Library, and they can run on Amazon EC2 instances.



**Different components of kinesis stream ?**

* data record : unit of data stored in streams consists of sequence number which is increasing with time. partition key and data blob.
* shard : Streams devided into shards, one shard provides 1 MB/s write and 2 MB/s read and upto 1000 PUTs operation. number\_of\_shards = max(incoming\_write\_bandwidth\_in\_KiB/1024, outgoing\_read\_bandwidth\_in\_KiB/2048)
* partition key : Its is used to route the data record to different shards which is declared by the producer. If producer does not define any partition key kinesis automatically define.

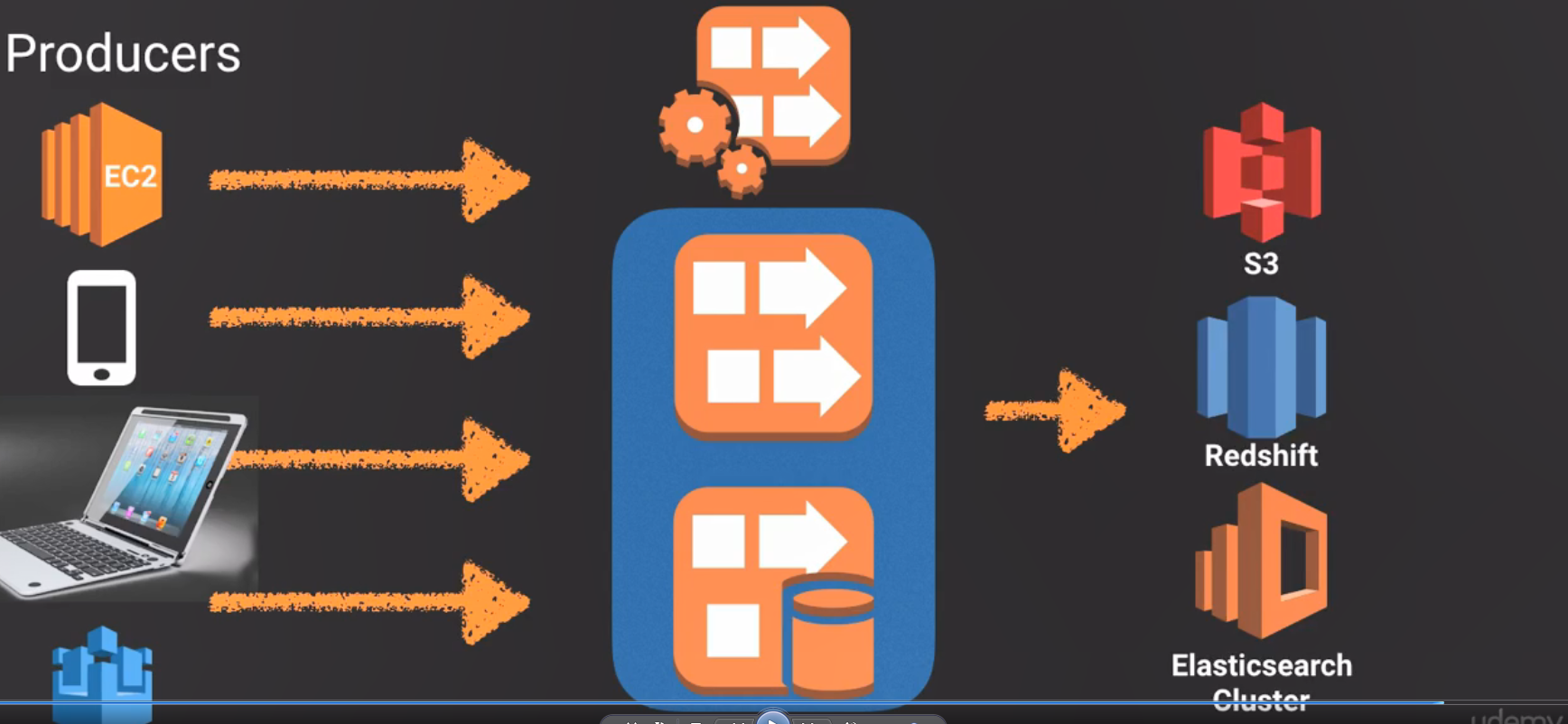
**Kinesis Firehose**



The destination can be S3 , Elastic search, Redshift

No concepts of consumers and shard here. It can be analyzed in the firehose and report stored into S3 or Redshift directly.

**Kinesis analytics**



Kinesis analytics seating on top of Kinesis stream and firehose , analyze the data stream and store the result in S3 , redshift or elastic search. We can make sql query on stored data of kinesis firehose or shard.

**When to use which service ?**

**SNS**

**My two favourite things about SNS:** it scales brilliantly and it is filterable.

**My worry about SNS:** that there’s a real chance to lose data if your alerts and design aren’t robust enough.

**My “thing to keep in mind” about SNS:** the costs. These can give you a nasty shock if you aren’t on the ball with your projections.

**SQS**

**My favourite thing about SQS:** it scales, massively, with no effort while keeping all your data nice and safe.

**My worry about SQS:** that it doesn’t work with multiple consumers. This can lead to a more complex architecture. With experience that’s not a problem, but as a novice, it can be intimidating.

**My thing to keep in mind about SQS:** because it scales so awesomely when you use Lambda to consume it, it can smash your [concurrency limits](https://docs.aws.amazon.com/lambda/latest/dg/limits.html)! Be sure to ring-fence the Lambda concurrency for the consumer.

**Kinesis**

**My favourite thing about Kinesis** is that it keeps a record of your messages for up to seven days. Brilliant if you want to “replay” data.

**My worry about Kinesis: t**hat scaling is up to you and can be awkward and slow (compared to SNS and SQS).

**My “thing to keep in mind” about Kinesis:** that although multiple consumers are theoretically supported, they can be mischevious to implement.