aws academy

CSCI 5902 Adv. Cloud Architecting
Fall 2023
Instructor: Dr. Lu Yang

Module 5 Adding a Database Layer (Sections 3 - 6)
Oct 16, 2023

Housekeeping and Feedback

- Start recording
- The midterm is in class on Oct 27. The contents covered are up to this Friday, Oct 20.
  - 45 multiple choice questions including 10 Azure questions.
- PIER tour tomorrow

Read replicas for performance

aws academy

Benefits and characteristics

- Enhanced performance ✓
- Increased availability ✓
- Designed for security ✓ ?
- Can be cross-AZ or cross-region CR
- Each read replica has its own DNS endpoint
- Read replicas can be promoted to be their own databases.

Supported by

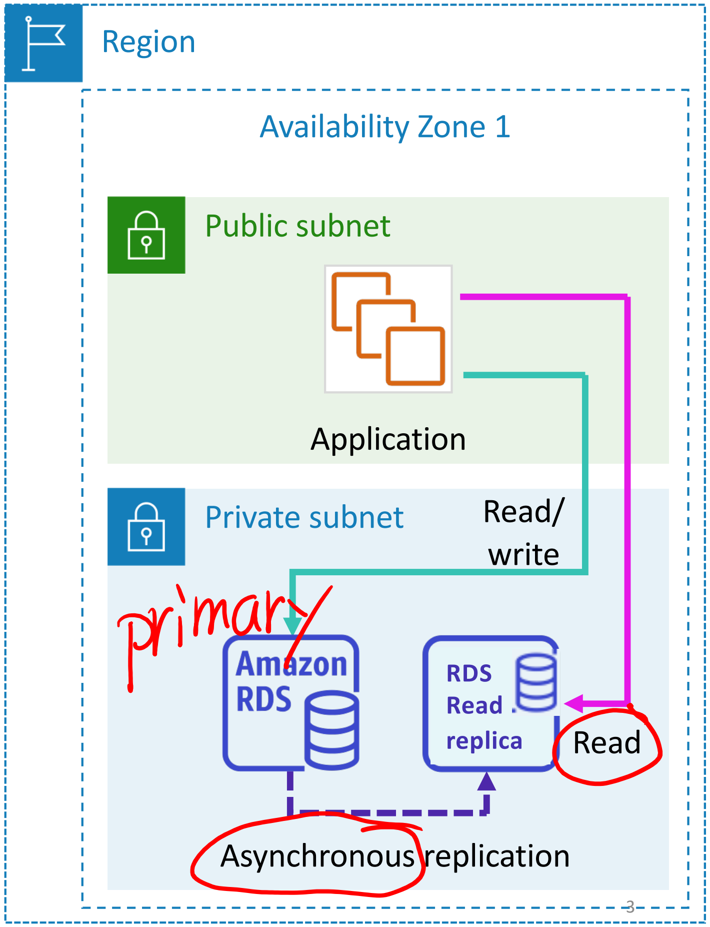
- MySQL, MariaDB, PostgreSQL, Oracle, SQL Server, Amazon Aurora

Setup

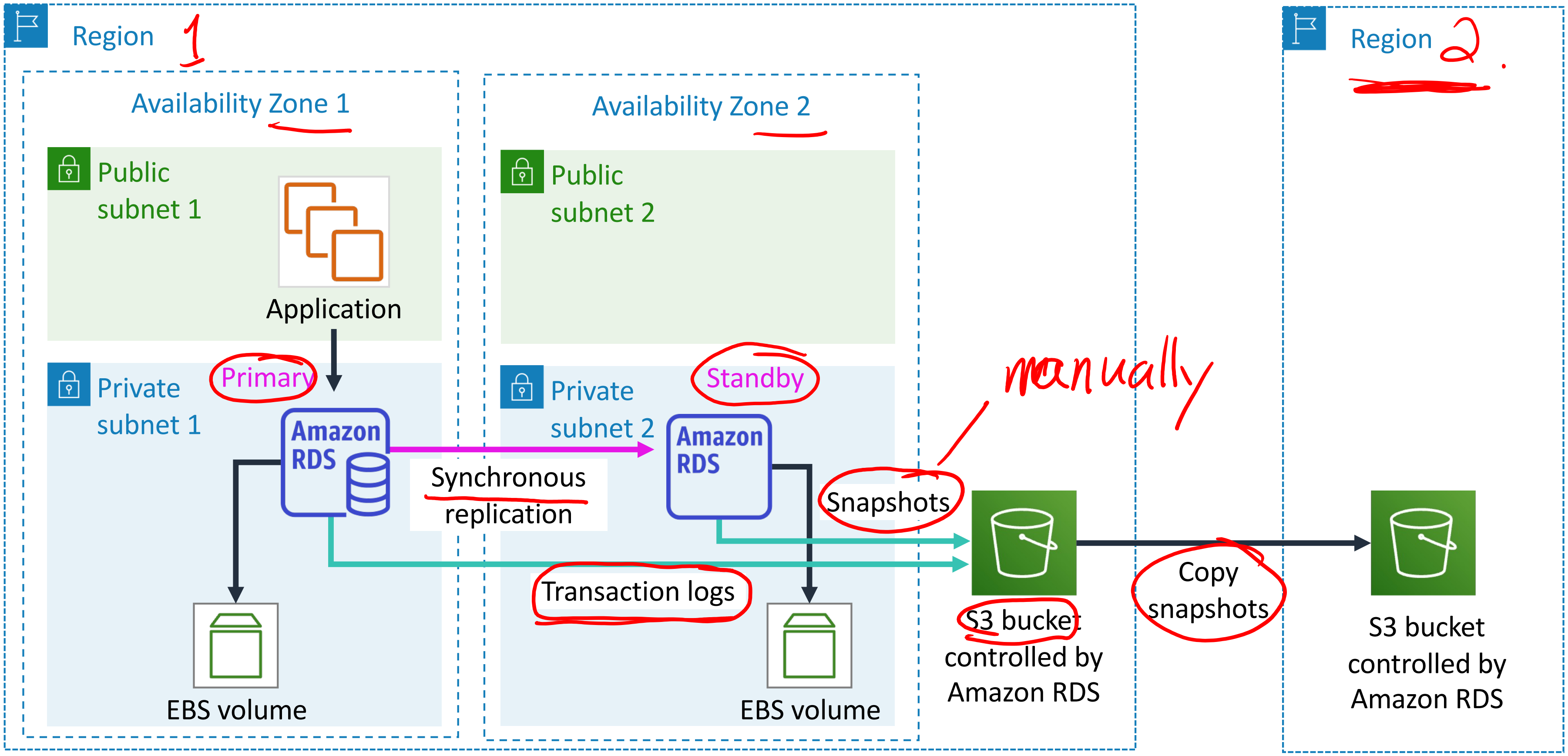
- Amazon RDS for MySQL, MariaDB and PostgreSQL allow you to add up to 15 read replicas to each DB Instance. Amazon RDS for Oracle and SQL Server allow you to add up to 5 read replicas to each DB Instance.

Limits

- For strict read-after-write consistency, read from the primary

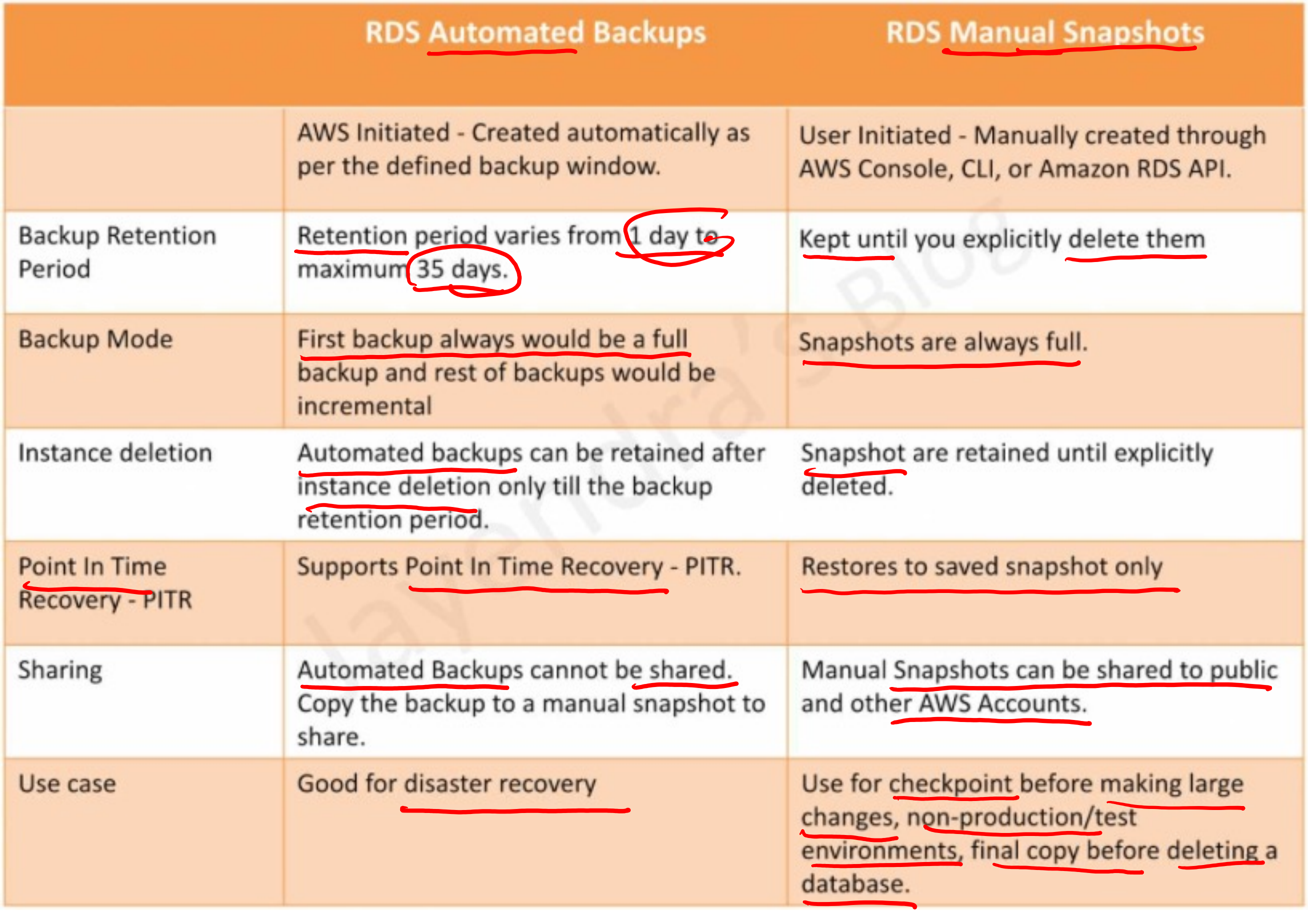


Amazon RDS backup solution (1/2)



Amazon RDS backup solution (2/2)

aws academy

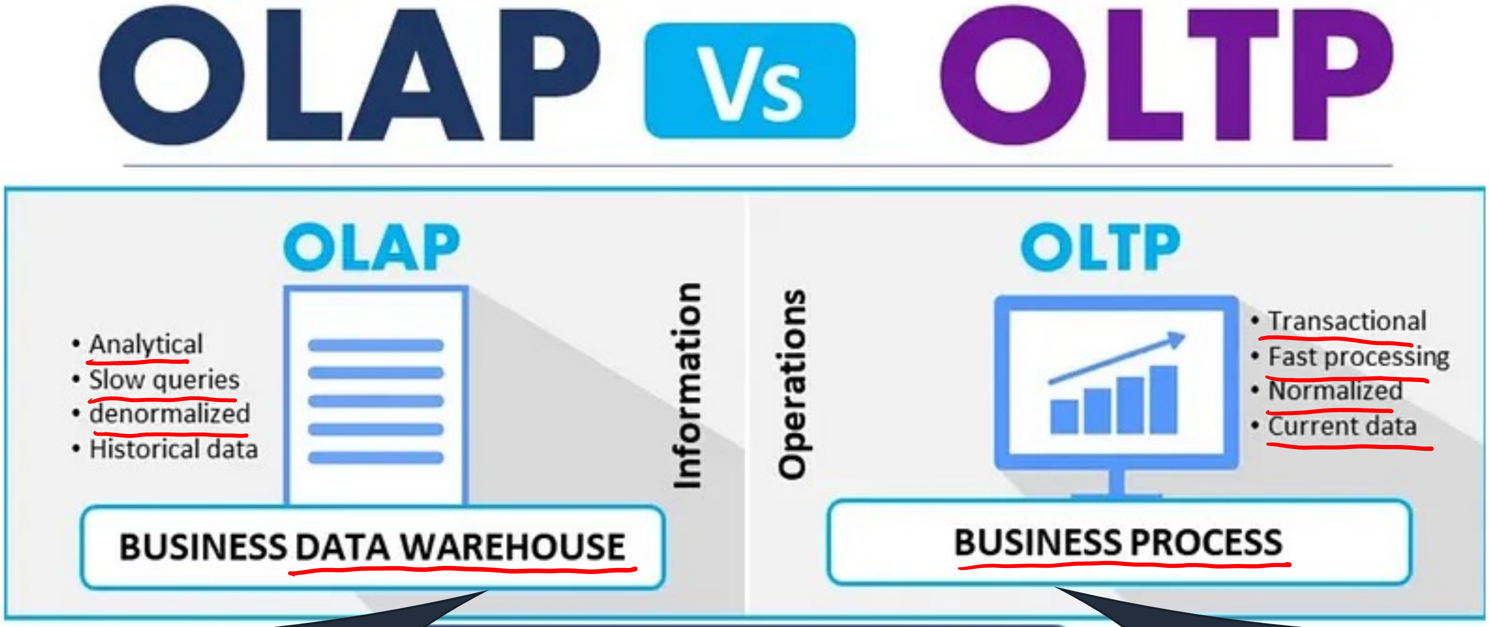


	RDS Automated Backups	RDS Manual Snapshots
	AWS Initiated - Created automatically as per the defined backup window.	User Initiated - Manually created through AWS Console, CLI, or Amazon RDS API.
Backup Retention Period	Retention period varies from 1 day to maximum 35 days.	Kept until you explicitly delete them
Backup Mode	First backup always would be a full backup and rest of backups would be incremental	Snapshots are always full.
Instance deletion	Automated backups can be retained after instance deletion only till the backup retention period.	Snapshot are retained until explicitly deleted.
Point In Time Recovery - PITR	Supports Point In Time Recovery - PITR.	Restores to saved snapshot only
Sharing	Automated Backups cannot be shared. Copy the backup to a manual snapshot to share.	Manual Snapshots can be shared to public and other AWS Accounts.
Use case	Good for disaster recovery	Use for checkpoint before making large changes, non-production/test environments, final copy before deleting a database.

https://jayendrapatil.com/aws-rds-db-snapshot-backup-restore/#:~:text=a%20final%20DB%20snapshot%20can,deleted%20and%20cannot%20be%20recovered

Demonstration: Amazon RDS Automated Backup and Read Replicas

OLAP vs. OLTP



s://ashutosh-bitmesra.medium.com/oltp-and-olap-what-are-the-difference

Data analysis using large amounts of data, as well as complex queries that take a long time to complete.

Data processing and completing large numbers of small transactions in real time.

Amazon Aurora

aws academy

Amazon Aurora

Amazon Aurora is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine.

- Used for online transactional processing (**OLTP**)
- Provides up to 5x the throughput of MySQL*
- Provides up to 3x the throughput of PostgreSQL*
- Requires little change to your existing application
- Start with 10GB and scale up to 128 TB.
- 2 copies of your data are contained in each AZ, with a minimum of 3 AZs. So 6 copies of your data.
- Transparently handle the loss of up to 2 copies of data without affecting database write availability and up to 3 copies without affecting read availability.
- Aurora storage is self-healing. Data blocks and disks are continuously scanned for errors and repaired automatically.
- It costs 20% more than RDS, but more efficient.
- It can load/offload data directly from/to S3

Amazon Redshift

Amazon Redshift

Amazon Redshift is a data warehousing service.

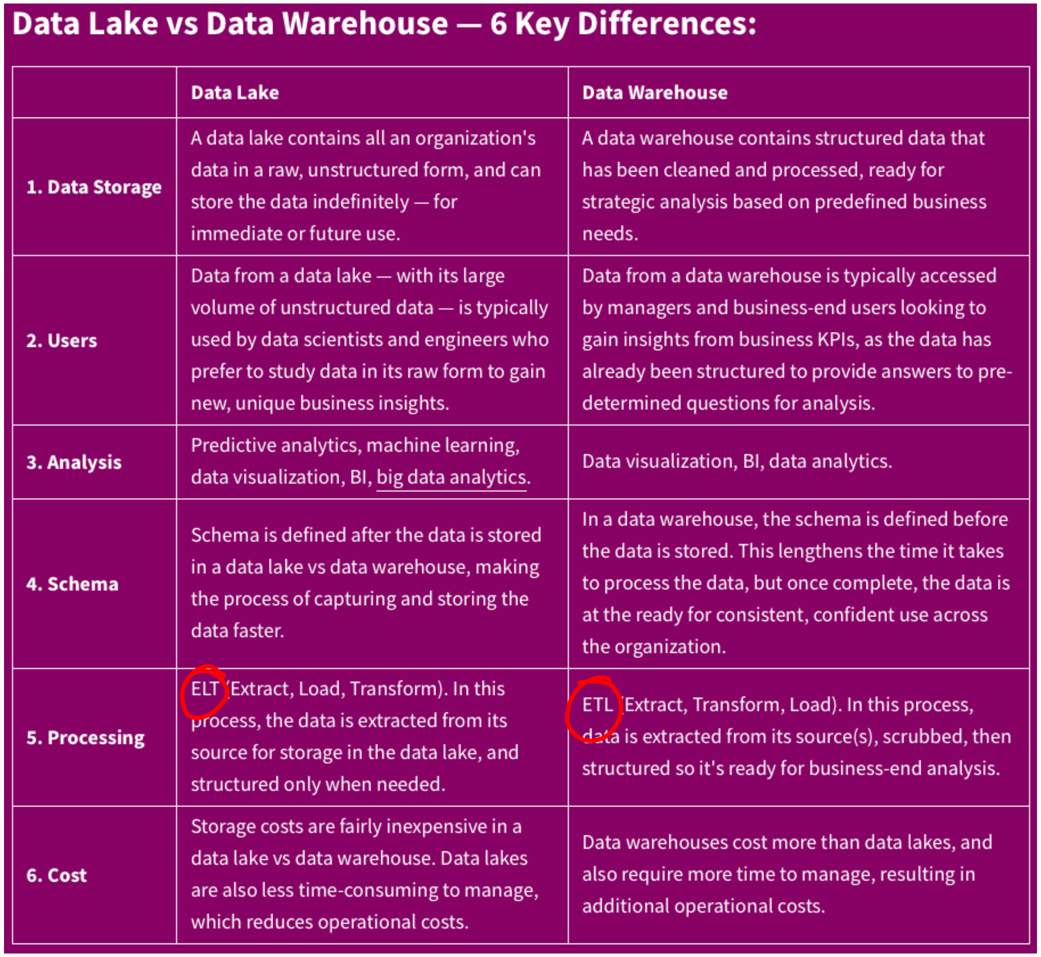
- Is used for online analytics processing (OLAP)
- Stores very large datasets
  - Store highly structured, frequently accessed data in Amazon Redshift
  - Can also store exabytes of structured, semistructured, and unstructured data in Amazon S3

Key word: big

It can support up to 16 PB.

Data Warehouse vs. Data Lake (optional)

aws academy



Data Lake vs Data Warehouse — 6 Key Differences:

	Data Lake	Data Warehouse
1. Data Storage	A data lake contains all an organization's data in a raw, unstructured form, and can store the data indefinitely — for immediate or future use.	A data warehouse contains structured data that has been cleaned and processed, ready for strategic analysis based on predefined business needs.
2. Users	Data from a data lake — with its large volume of unstructured data — is typically used by data scientists and engineers who prefer to study data in its raw form to gain new, unique business insights.	Data from a data warehouse is typically accessed by managers and business-end users looking to gain insights from business KPIs, as the data has already been structured to provide answers to pre-determined questions for analysis.
3. Analysis	Predictive analytics, machine learning, data visualization, BI, big data analytics.	Data visualization, BI, data analytics.
4. Schema	Schema is defined after the data is stored in a data lake vs data warehouse, making the process of capturing and storing the data faster.	In a data warehouse, the schema is defined before the data is stored. This lengthens the time it takes to process the data, but once complete, the data is at the ready for consistent, confident use across the organization.
5. Processing	ELT (Extract, Load, Transform). In this process, the data is extracted from its source for storage in the data lake, and structured only when needed.	ETL (Extract, Transform, Load). In this process, data is extracted from its source(s), scrubbed, then structured so it's ready for business-end analysis.
6. Cost	Storage costs are fairly inexpensive in a data lake vs data warehouse. Data lakes are also less time-consuming to manage, which reduces operational costs.	Data warehouses cost more than data lakes, and also require more time to manage, resulting in additional operational costs.

Reference:
https://www.qlik.com/us/data-lake/data-lake-vs-data-warehouse

Section 3 key takeaways

- Managed AWS database services handle administration tasks so you can focus on your applications
- Amazon RDS supports Microsoft SQL Server, Oracle, MySQL, PostgreSQL, Aurora, and MariaDB
- Amazon RDS Multi-AZ deployments provide high availability with automatic failover
- Amazon RDS for Oracle and SQL Server allow you add up to 5 read replicas per primary database
- Amazon Aurora is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine
- Amazon Redshift is a relational database offering for data warehousing

Module 5: Adding a Database Layer

Section 4: Amazon DynamoDB

Amazon DynamoDB

aws academy

A fully managed non-relational key-value and document database service.

Amazon DynamoDB

Performance at any scale

Extreme horizontal scaling capability

~~24/7~~

Serverless

Event-driven programming (serverless computing)

Enterprise-ready

Encryption, access controls, backups

Amazon DynamoDB characteristics

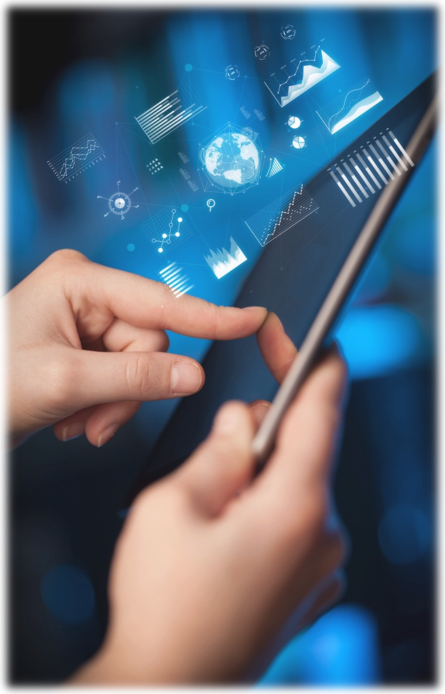
Amazon DynamoDB

Works well for applications that:

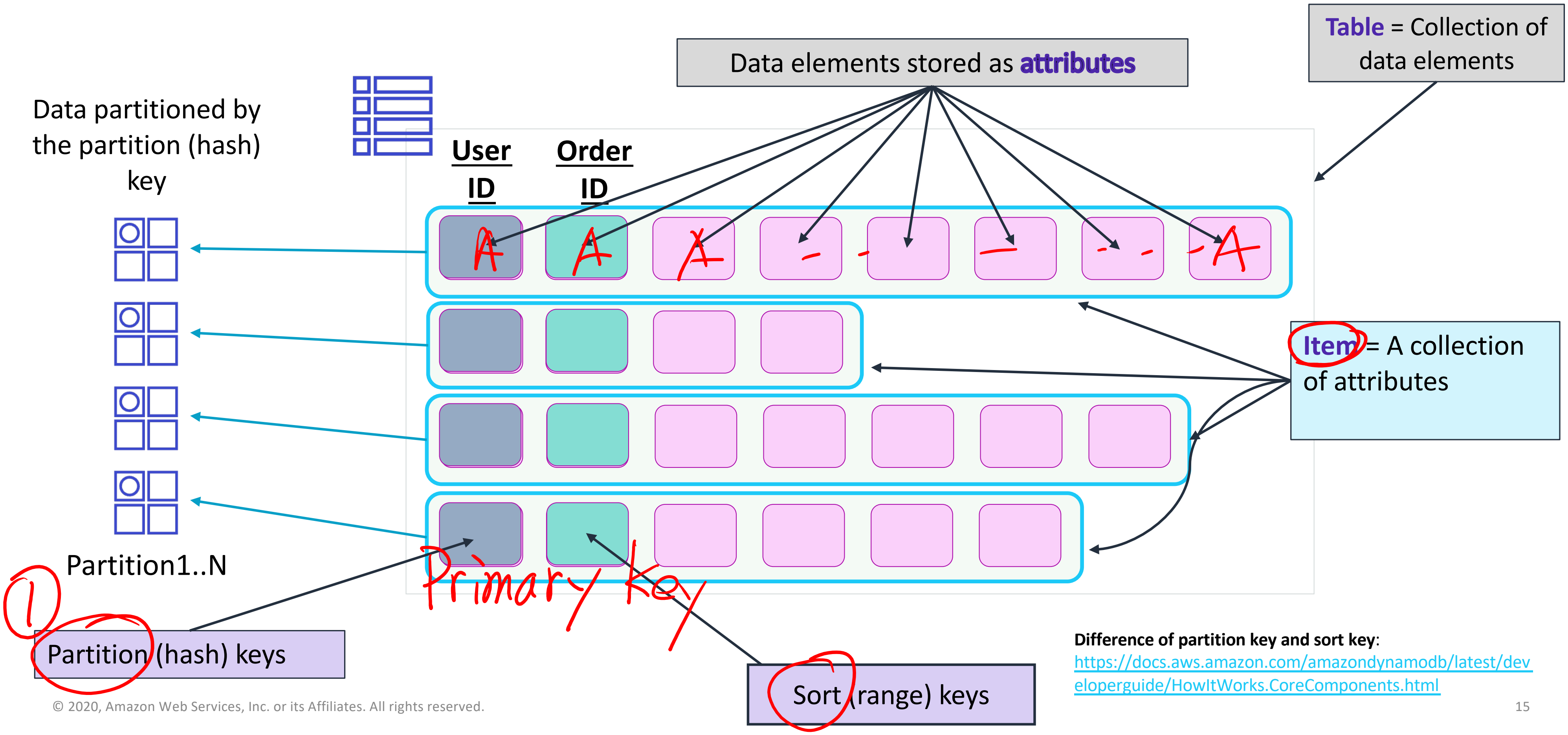
- Have high-volume data (high-TB range)
- Must scale quickly
- Don't need complex joins
- Require ultra-high throughput and low latency

Key features

- NoSQL tables
- Items can have differing attributes
- In-memory caching
- Support for peaks of more than 20 million requests per second



Amazon DynamoDB data model



Data elements stored as **attributes**

Table = Collection of data elements

Data partitioned by the partition (hash) key

User ID

Order ID

Item = A collection of attributes

Partition1..N

Partition (hash) keys

Sort (range) keys

Difference of partition key and sort key:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.CoreComponents.html

Amazon DynamoDB ACID Transactions – Who, What, Why?

- Who are using it?

  Disney, Salesforce, Snap Inc, Dropbox, Capital One, and more, use DynamoDB for various use cases

- What are DynamoDB **ACID transactions**?
  - For multiple, all-or-nothing (ACID) operations within and across one or more tables
  - DynamoDB provides ACID transactions across one or more tables within a single AWS account and region
  - No extra cost to enable transactions for DynamoDB tables

- **Atomic**: A quality that ensures each transaction is **fully committed** or **not at all**. It treats each transaction as a single unit of change to the database and cannot be partially completed. The transaction is **either committed or rolled back**.
- **Consistent**: A quality that ensures the database is in a **consistent** and **valid** state after a transaction is committed. It prevents the database from being corrupted or causing data integrity issues.
- **Isolated**: It guarantees that transactions are **not dependent** on each other. A transaction is executed in such a way that it does not affect other transactions.
- **Durable**: It guarantees that a transaction is committed even if the system fails and there is **no breakdown** in the event of **failures** (such as power loss).

Atomic

Consistent

Isolated

Durable

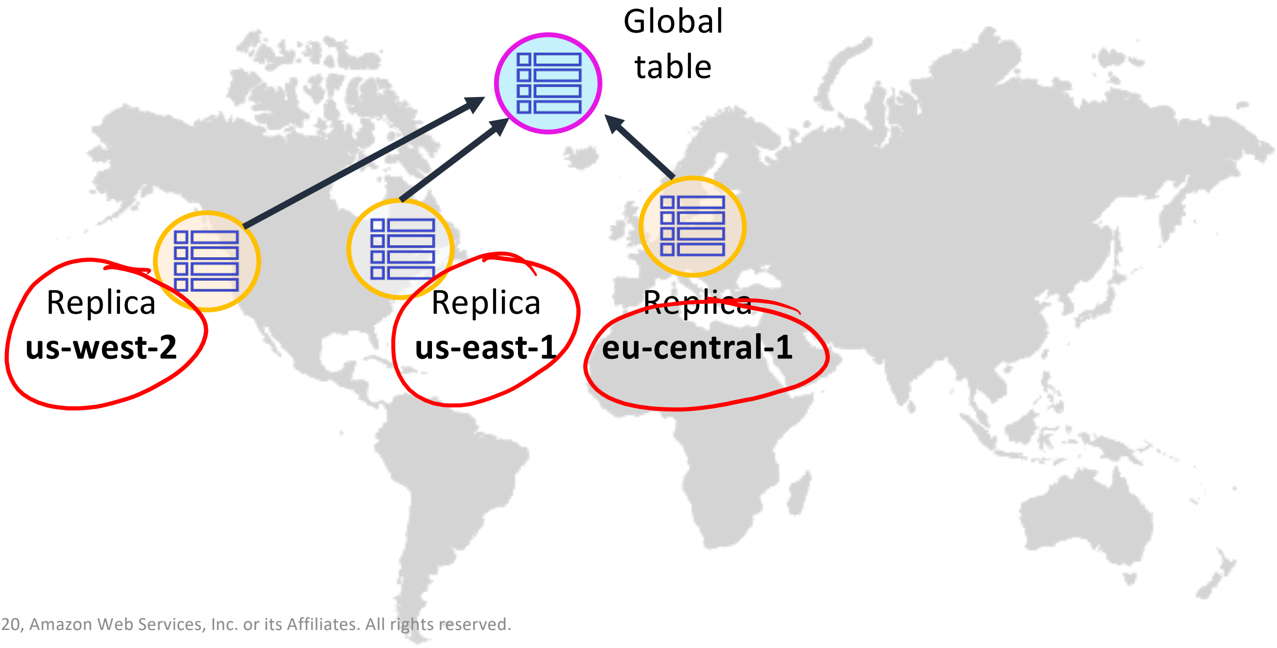
Reference: https://dev.to/hirendhaduk_/dynamodb-acid-transactions-what-why-how-mcf

Amazon DynamoDB ACID Transactions – Who, What, Why?

- Why use DynamoDB ACID transactions?
  - They extend the scale, performance, and enterprise benefits of DynamoDB to a broader range of mission-critical workloads
  - You can support more sophisticated workloads and business logic that require **updating**, **adding**, and **deleting** multiple items as a single, all-or-nothing operation.
    - Processing financial transactions
    - Managing and fulfilling orders
    - Building multiplayer game engines
    - Coordinating actions across distributed services and components

Amazon DynamoDB global tables

- Managed multi-master, multi-region replication
- Based on DynamoDB streams
- Multi-region redundancy for disaster recovery or high availability
- Replication latency under 1 second



Fast local read/write performance for global applications.

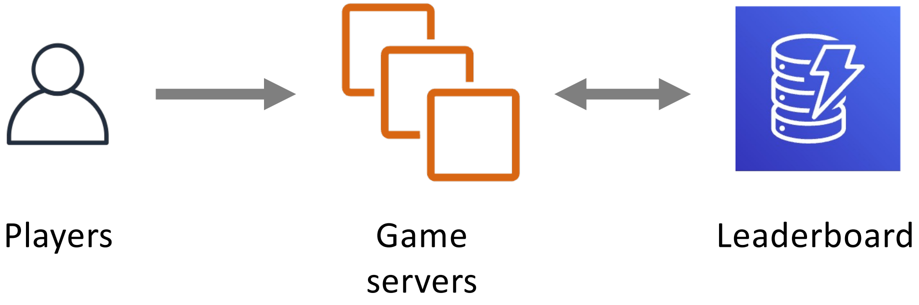
Demonstration: Global tables for inter-Regional replication

https://www.youtube.com/watch?v=Fa8Vf4Y7J_A

(4:25-end)

Amazon DynamoDB use case 1

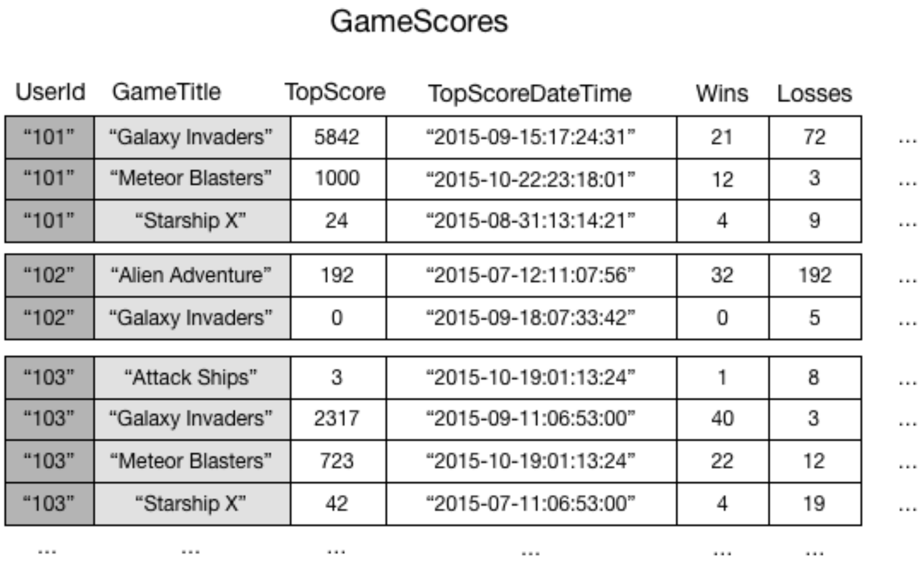
Leaderboards and Scoring



Players

Game servers

Leaderboard

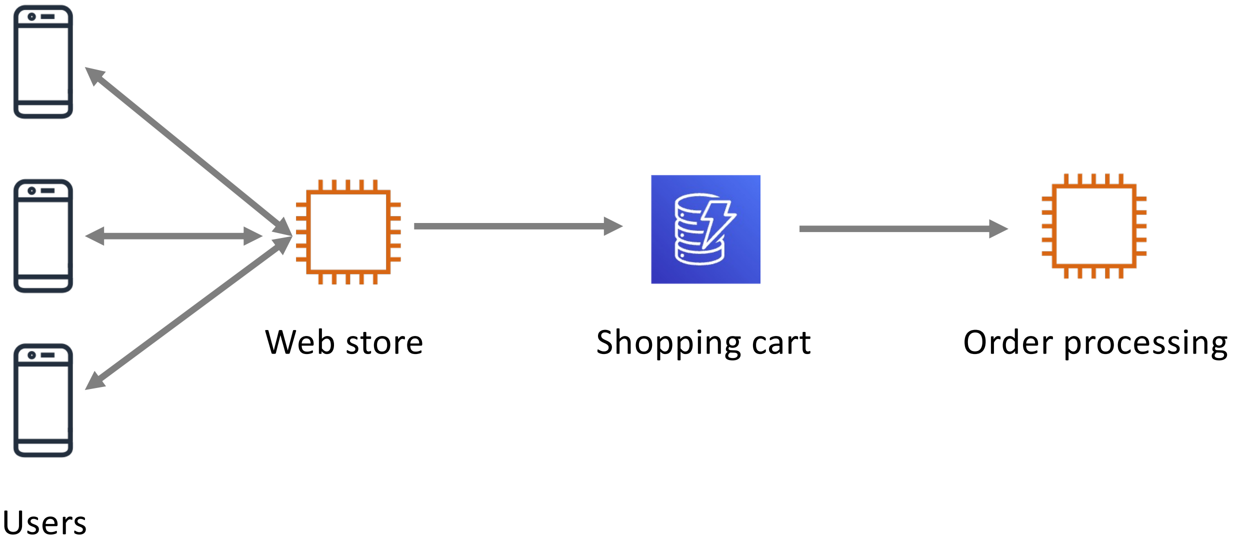


GameScores

UserId	GameTitle	TopScore	TopScoreDateTime	Wins	Losses	
"101"	"Galaxy Invaders"	5842	"2015-09-15:17:24:31"	21	72	...
"101"	"Meteor Blasters"	1000	"2015-10-22:23:18:01"	12	3	...
"101"	"Starship X"	24	"2015-08-31:13:14:21"	4	9	...
"102"	"Alien Adventure"	192	"2015-07-12:11:07:56"	32	192	...
"102"	"Galaxy Invaders"	0	"2015-09-18:07:33:42"	0	5	...
"103"	"Attack Ships"	3	"2015-10-19:01:13:24"	1	8	...
"103"	"Galaxy Invaders"	2317	"2015-09-11:06:53:00"	40	3	...
"103"	"Meteor Blasters"	723	"2015-10-19:01:13:24"	22	12	...
"103"	"Starship X"	42	"2015-07-11:06:53:00"	4	19	...
...	...	...	...	...	...	

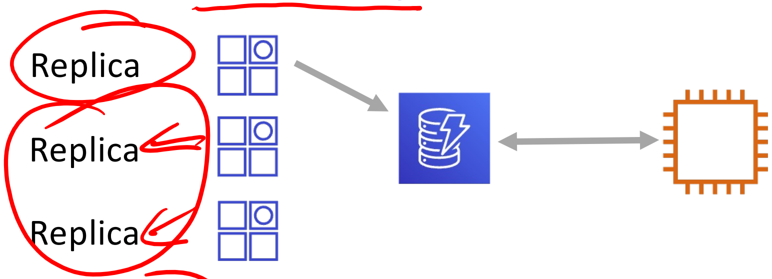
Amazon DynamoDB use case 2

Temporary Data (Online Cart)



Amazon DynamoDB consistency options

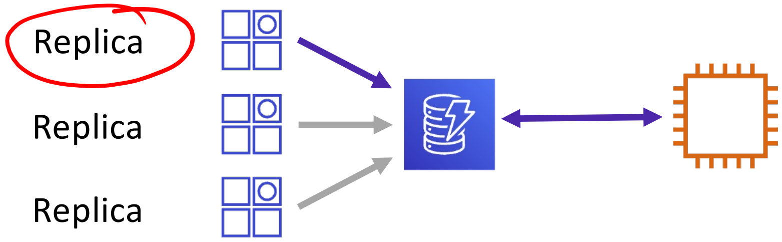
Eventually consistent



The **default** setting. All copies of data usually reach consistency within 1 second.

Pros	Cons
Low latency	Return stale data
Prioritizes availability	Debugging process hard

Strongly consistent



This feature is **optional**. Used for applications that require all reads to return a result that reflects all writes before the read.

Pros	Cons
Return the most updated data	Higher latency
System more accurate	Not supported on global secondary indexes
	Have to specify in the query
	2x expensive than eventually

Section 4 key takeaways

- Amazon DynamoDB is a fully managed non-relational key-value and document NoSQL database service.
- DynamoDB is serverless, provides extreme horizontal scaling and low latency.
- DynamoDB global tables ensure that data is replicated to multiple Regions.
- DynamoDB provides eventual consistency by default (in general, it is fully consistent for reads 1 second after the write). Strong consistency is also an option.

Module 5: Adding a Database Layer

Section 5: Database security controls

Securing Amazon RDS databases

aws academy

Recommendations

- Run the RDS instance in a virtual private cloud (VPC)
  - Provides service isolation and IP firewall protection
- Use AWS Identity and Access Management (IAM) policies for authentication and access
  - Permissions determine who is allowed to manage Amazon RDS resources
- Use security groups to control what IP addresses or EC2 instances can connect to your databases
  - By default, network access is disabled
- Use Secure Sockets Layer (SSL) for encryption in transit
- Use Amazon RDS encryption on DB instances and snapshots to secure data at rest
- Use the security features of your DB engine to control who can log in to the databases on a DB instance
- Configure event notifications to alert you when important Amazon RDS events occur

SNS

Amazon RDS

Securing Amazon DynamoDB

aws academy

Recommendations

- Use **IAM roles** to authenticate access
- Use **IAM policies**
  - To define fine-grain access permissions to use DynamoDB APIs
  - Define access at the table, item, or attribute level
  - Follow the **principle of granting least privilege**
- Configure **VPC endpoints**
  - Prevents connection traffic from traversing the open internet
  - VPC endpoint policies allow you to control and limit API access to a DynamoDB table
- Consider **client-side encryption**
  - Encrypt data as close as possible to its origin

Amazon DynamoDB

Security provided by default

- **Encryption at rest** of all user data stored in tables, indexes, streams, and backups
- **Encryption in transit –** All communications to and from DynamoDB and other AWS resources use HTTPS

Demonstration:

Securing your Amazon DynamoDB and RDS databases

https://www.youtube.com/watch?v=J9PtBYodD24

Module 5: Adding a Database Layer

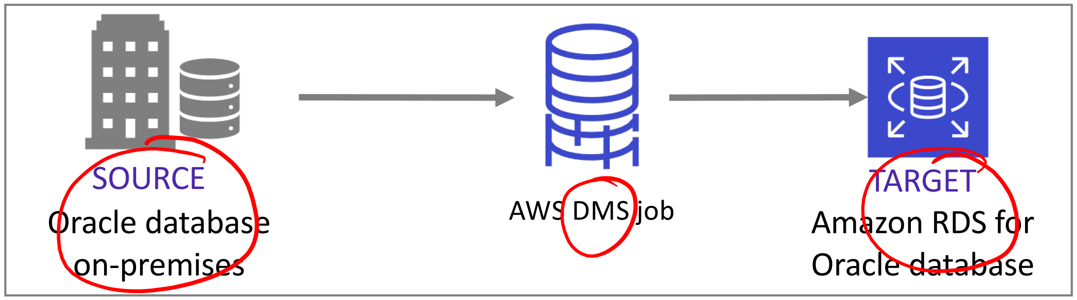
Section 6: Migrating data into AWS databases

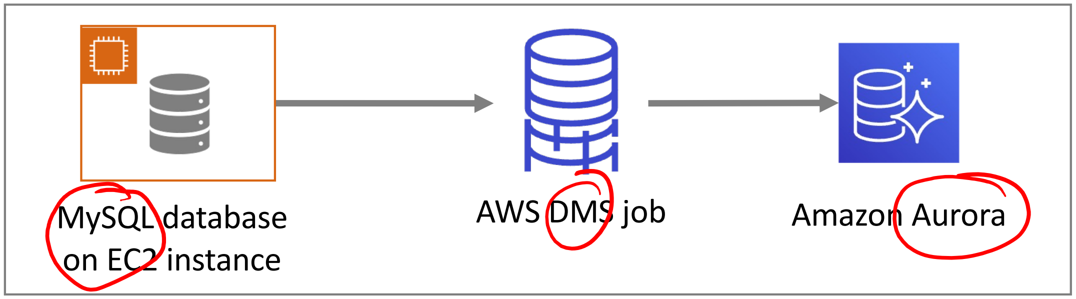
AWS Database Migration Service

AWS Database Migration Service (AWS DMS)

- Use to migrate to and from most commercial and open source databases
- Migrate between databases on Amazon EC2, Amazon RDS, Amazon S3, and on-premises

Example homogenous migration





Example heterogeneous migration

AWS DMS key features

- Perform one-time migrations
- Or, accomplish continuous data replication
  - Example: Configure continuous data replication of an on-premises database to an RDS instance
- AWS Schema Conversion Tool (AWS SCT) supports changing the database engine between source and target
- Typical migration major steps:
  1. Create a target database
  2. Migrate the database schema SCT
  3. Set up the data replication process
  4. Initiate the data transfer, and confirm completion
  5. Switch production to the new database (for one-time migrations)

DMS {

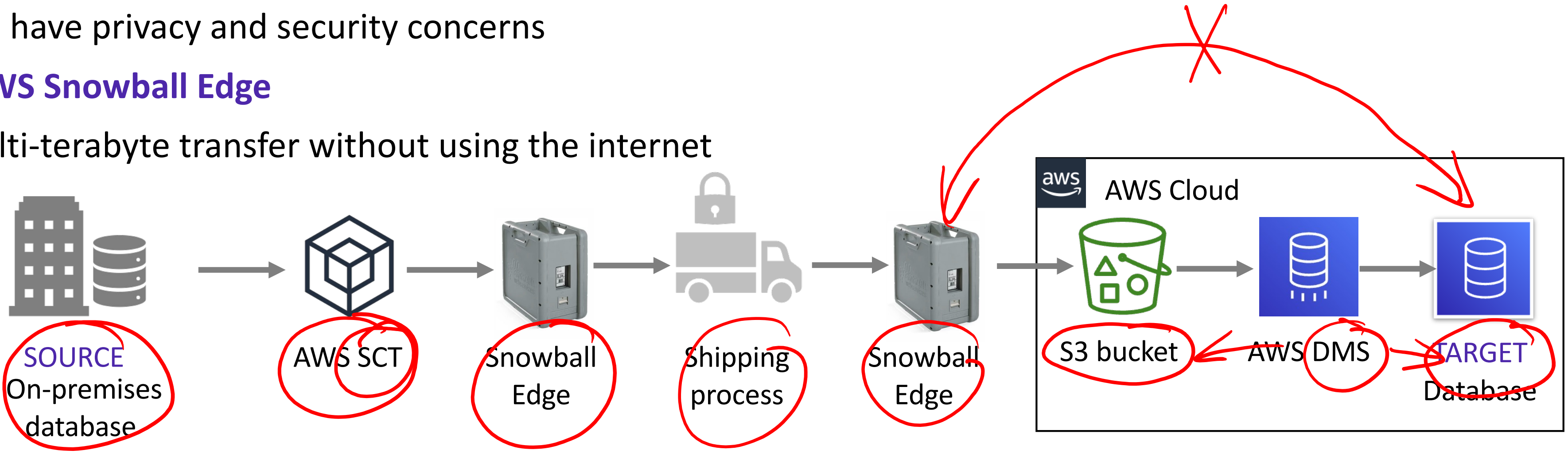
Using AWS Snowball Edge with AWS DMS

When migrating data is not practical:

- Database is too large
- Connection is too slow
- You have privacy and security concerns

Use **AWS Snowball Edge**

- Multi-terabyte transfer without using the internet



Optional reading:
AWS Schema Conversion Tool: https://aws.amazon.com/dms/schema-conversion-tool/

Module 5: Adding a Database Layer

Module wrap-up

Module summary

In summary, in this module, you learned how to:

- Compare database types
- Differentiate between managed versus unmanaged services
- Explain when to use Amazon Relational Database Service (Amazon RDS)
- Explain when to use Amazon DynamoDB
- Describe available database security controls
- Describe how to migrate data into Amazon Web Services (AWS) databases
- Deploy a database server