Databases

Session 2

Sandra Pico Oristrell

December 2023

laSalle
UNIVERSIDAD RAMON LLULL

Agenda

Part 1: Theory (1h:15 min)

- Introduction to Databases in AWS
- Introduction to Amazon RDS
- Introduction to Amazon DynamoDB (Key-value pair database)
- Introduction to Graph Databases and Amazon Neptune
- Resources

Break (15 min)

- **(45 min)** Use case: Building a Knowledge Graph with Amazon Neptune

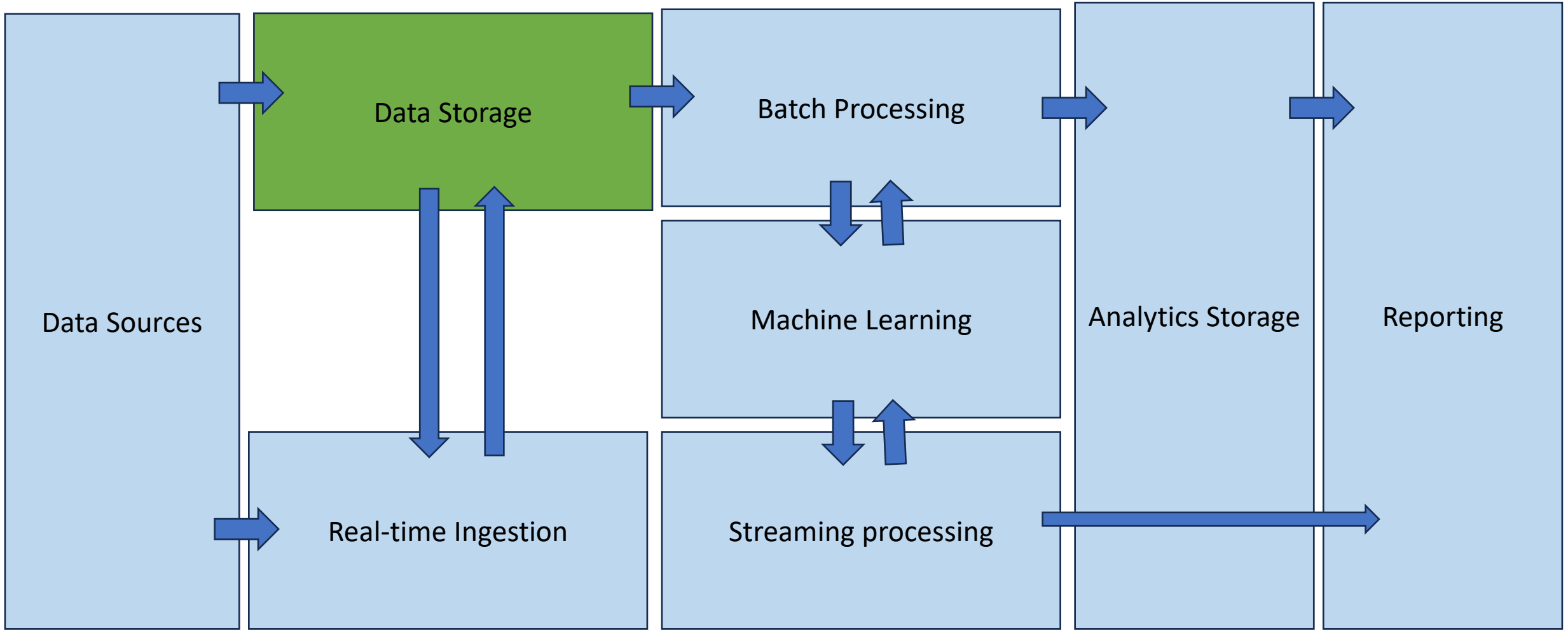
Part 2: Hands-on (45min)

- AWS Educate: Getting Started with Databases
- AWS Educate: Builder Labs (Adding Dynamic Data to your Application with DynamoDB)

Part 1: Theory

Big Data on AWS

Session 2: Databases

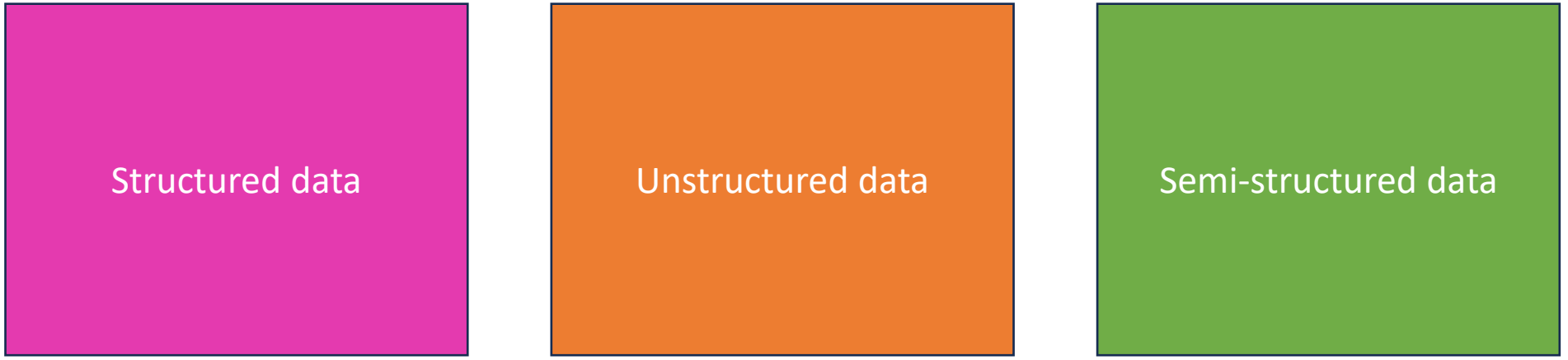


Introduction to Databases in AWS

Why databases?

A **database** is a logically organized collection of information, designed in such a way that the information within can be accessed for later use by a computer program. Databases give access to the data while keeping the integrity of the data in a secure environment.

A **data model** is the logical structure of a database and determined the rules for how information can be organized and used. The data models you choose is influenced by the structure of your data: structured, unstructured and semi structured.



Data types

Structured data

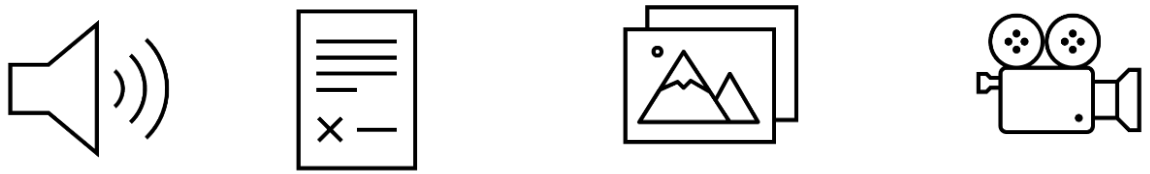
- Stored as a series of data values in related tables.
- Highly structured
- Formatted so the elements can be made addressable for more effective processing and analysis
- Data able to be used in highly complex queries.

SKU	Color	Amount	OrderDate
XYZ	Blue	200	07-2022
ABC	Orange	500	08-2022

Data types

Unstructured data

- Stored as files
- Lacks predefined structured
- Special tools to catalog and query
- Examples: images, audio, video, word processing files…



Data types

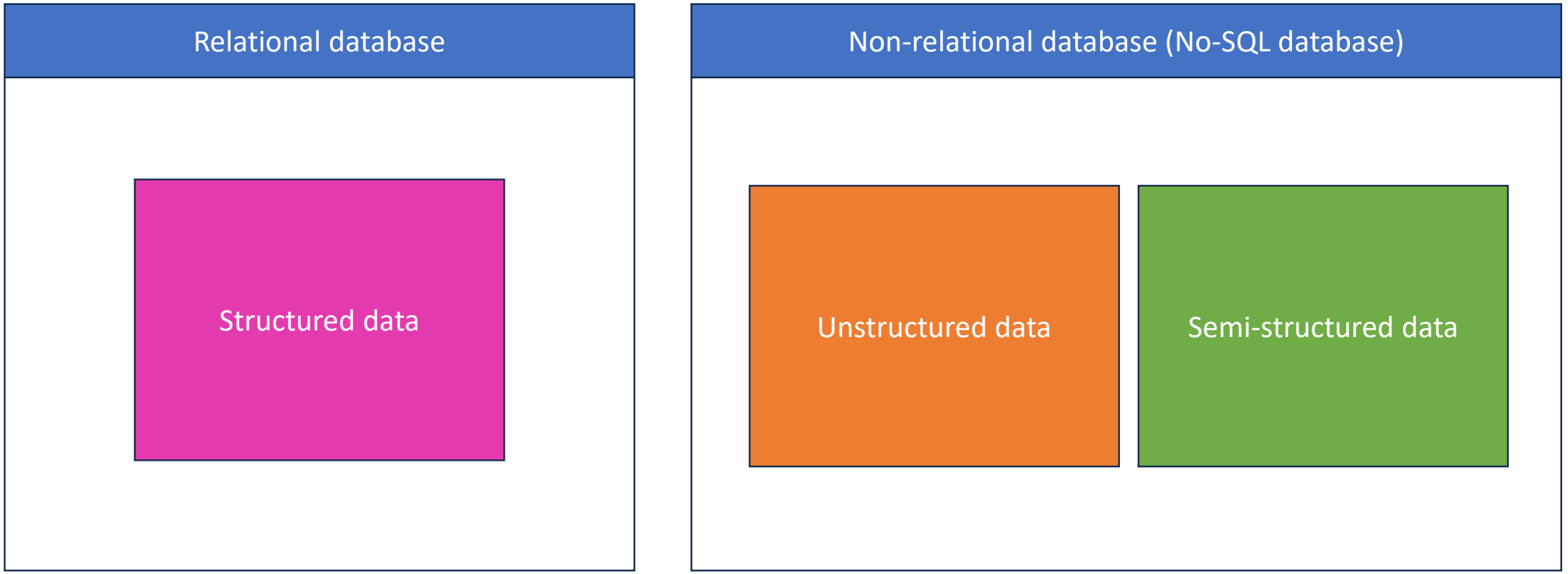
Semi-structured data

- Highly flexible because the structure is not strict and can be changed as needed within the table.

- Semi structured data can be analyzed but not with the same level of complexity that analytics on structured data can achieve.

```
{
        first_name:“Carol”,
        last_name:“Danvers”,
        order_id:“123216”,
        default_size:[6,8],
        order_total:“62.26”
},
{
        first_name: “Dave”
        last_name:“Do”,
        order_id:“599764”,
        default_size:[8,10,12],
        order_placement:“Online”
        order_total:“45.79”
}
```

Relational and non-relational database



Database terms and concepts

1 Schema

A database schema is the blueprint of the database. The schema outlines the relationships within the database and the constraints of the database.

2 IOPS

Input/output operation per second (IOPS) is the measure of performance of reads and writes to a storage location like a database.

To improve latency and read/write throughput, provision more IOPS when configuring the database. Limiting the IOPS may cause the database to hit the threshold and reduce performance. Over provisioning of IOPS will result in higher costs for the instance.

Database terms and concepts

3 ACID and BASE compliance

Transactions are groups of statements or operations that take place in a database. Issues arise when tow sessions try to change the same record or item at the same time. Transactional compliance includes methods for maintaining consistency and integrity in a database.

- **ACID** (Atomicity, consistency, isolation and durability) is used in **structured database**. One goal of an ACID compliant database is to return the most recent version of all data. It also must ensure that data entered the system always meets all assigned rules and constraints.

- **BASE** compliance is used in a **non-relational** database. BASE supports data integrity in non-relational databases.

Database terms and concepts

3 ACID and BASE compliance

ACID

Atomicity: When completing a transaction, atomicity ensures that your transactions either completely succeed or completely fail.

Consistency: Ensures that all transactions provide valid data to the database. Ensures that data updates are not made available until all replicas have been updated as well.

Isolation: Isolation ensures that one transaction cannot interfere with another current transaction.

Durability: Making sure your data changes stick.

BASE

Basically Available (BA): Allows for one instance to receive a change request and make that change available immediately. In an ACID system the change does not become available until all instances are consistent. Consistency in a BASE model is traded for availability.

Soft state: There are allowance for partial consistency across distributed instances. In an ACID system, the database is in a hard state because users cannot access data that is not fully consistent.

Eventually consistent: The data will eventually be consistent. The change will eventually be made to every copy. However, the data will be available in whatever state it is during propagation of the change.

Database terms and concepts

4 OLTP/OLAP

The structure of relational tables is great for transactional data and data that may require highly complex join operations when queried. There are two methods for organizing information: OLTP (Online Transaction Processing) or OLAP (Online analytical processing).

OLTP databases focus on recording Update, Insertion and Deletion data transactions. OLTP queries are simple and short, which require less time and space to process. Example: Bank ATM, in which you can modify your bank account using short transactions.

OLAP databases store historical data that has been input by OLTP. OLAP databases allow users to view different summaries of multidimensional data. A good example of OLAP system is a business intelligence tool which ingest business data and presents user-friendly views such as reports, dashboards and graphs.

Relational and non-relational database

Relational database

A relational database is a collection of data records organized as a set of tables with rows and columns. Each row holds information about a record, and the columns contain attributes of the data. Each row has an identifier column to uniquely distinguish the record from other records. The value of the identifier column is called a primary key. Other tables can reference a primary key. A record in another table that references a primary key is called a foreign key.

Course_ID	Title	Status
1113	Getting Started with Databases	200
1114	Getting Started with Cloud	500

Primary key

Book_ID	Book Title	Course ID
24221	Introduction to SQL	1113
24122	Introduction to IAM	1114

Ins_ID	Book Title	Course ID
8977	Marta Gonzalez	1113
8978	Joan Serra	1114

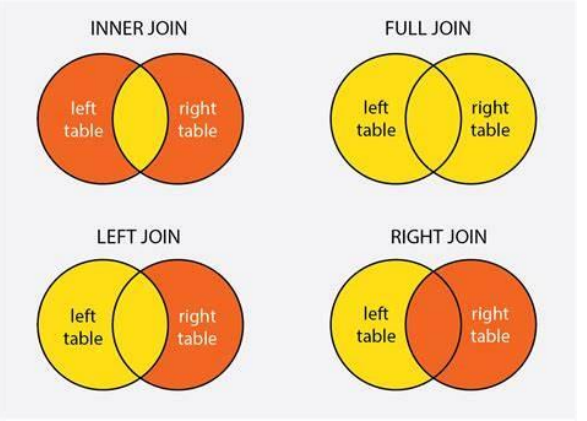
Foreign key

Relational and non-relational database

Relational database: Structured Query Language (SQL)

- **What books do I need to buy to take each course?**

```
SELECT * FROM TableA LEFT JOIN TableB ON TableA.Course_ID = TableB.Course_ID;
```



TableA

Course_ID	Title	Status
1113	Getting Started with Databases	200
1114	Getting Started with Cloud	500

Book_ID	Book Title	Course ID
24221	Introduction to SQL	1113
24122	Introduction to IAM	1114

TableB

Ins_ID	Book Title	Course ID
8977	Marta Gonzalez	1113
8978	Joan Serra	1114

TableC

laSalle
UNIVERSIDAD RAMON LLULL

Relational and non-relational database

Relational database: Structured Query Language (SQL)

- **What books do I need to buy to take each course?**

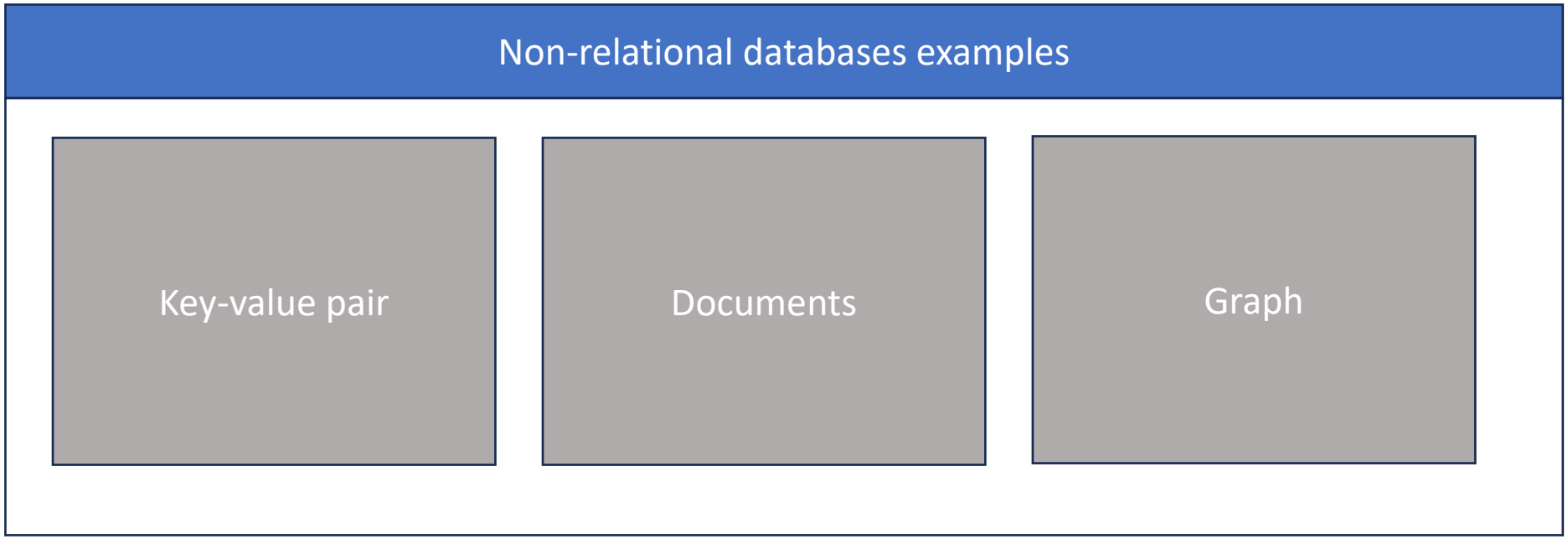
```
SELECT * FROM TableA LEFT JOIN TableB ON TableA.Course_ID = TableB.Course_ID;
```

Course_ID	Title	Status	Book_ID	Book Title
1113	Getting Started with Databases	200	24221	Introduction to SQL
1114	Getting Started with Cloud	500	24122	Introduction to IAM

Relational and non-relational database

Non-relational database

A non-relational database stores data in unstructured ways using one of many storage models, including key-value pairs, documents and graphs. Non relational schemas are dynamic – i.e., a row does not have to contain data for each column.



Why AWS databases

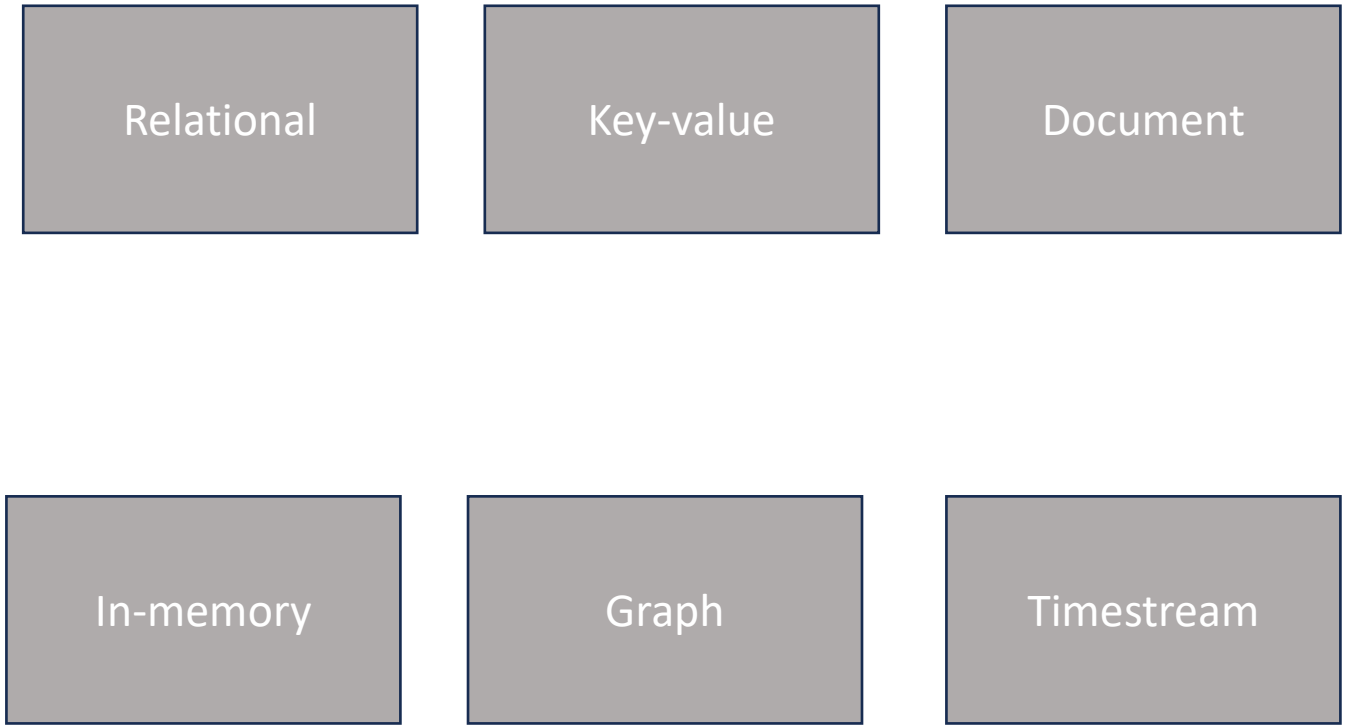
Purpose built

Performance at scale

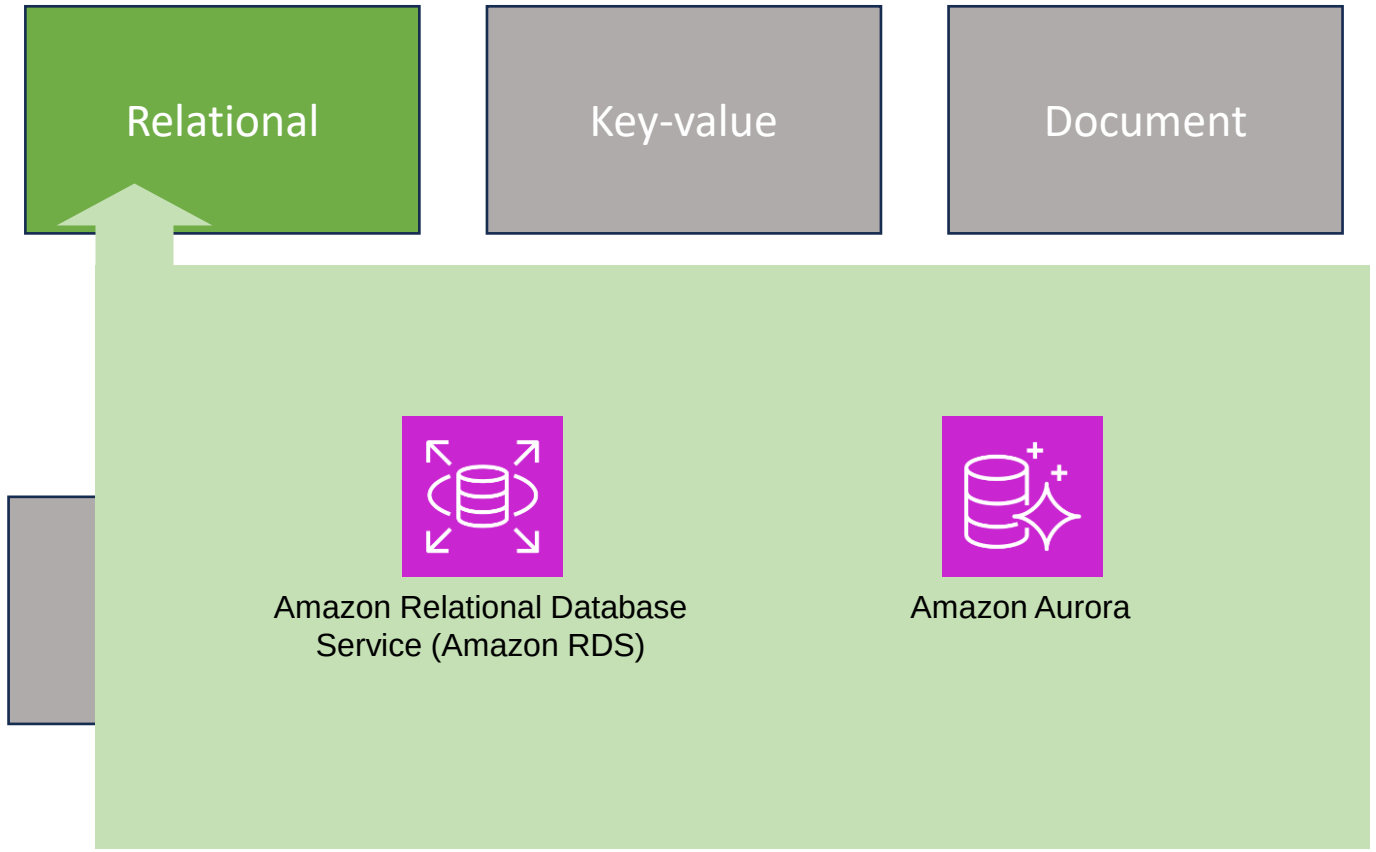
Fully managed

Secure and highly available

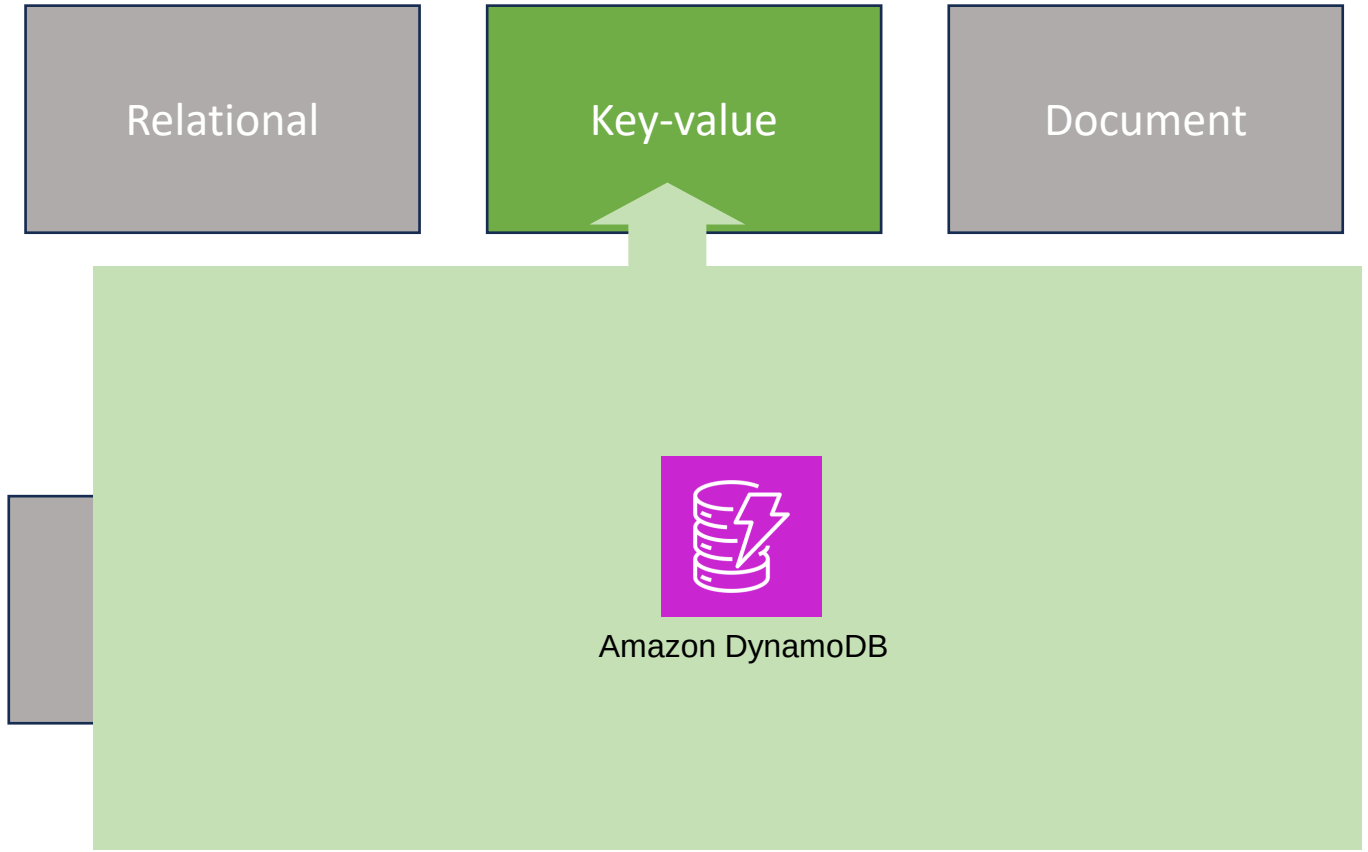
AWS Databases



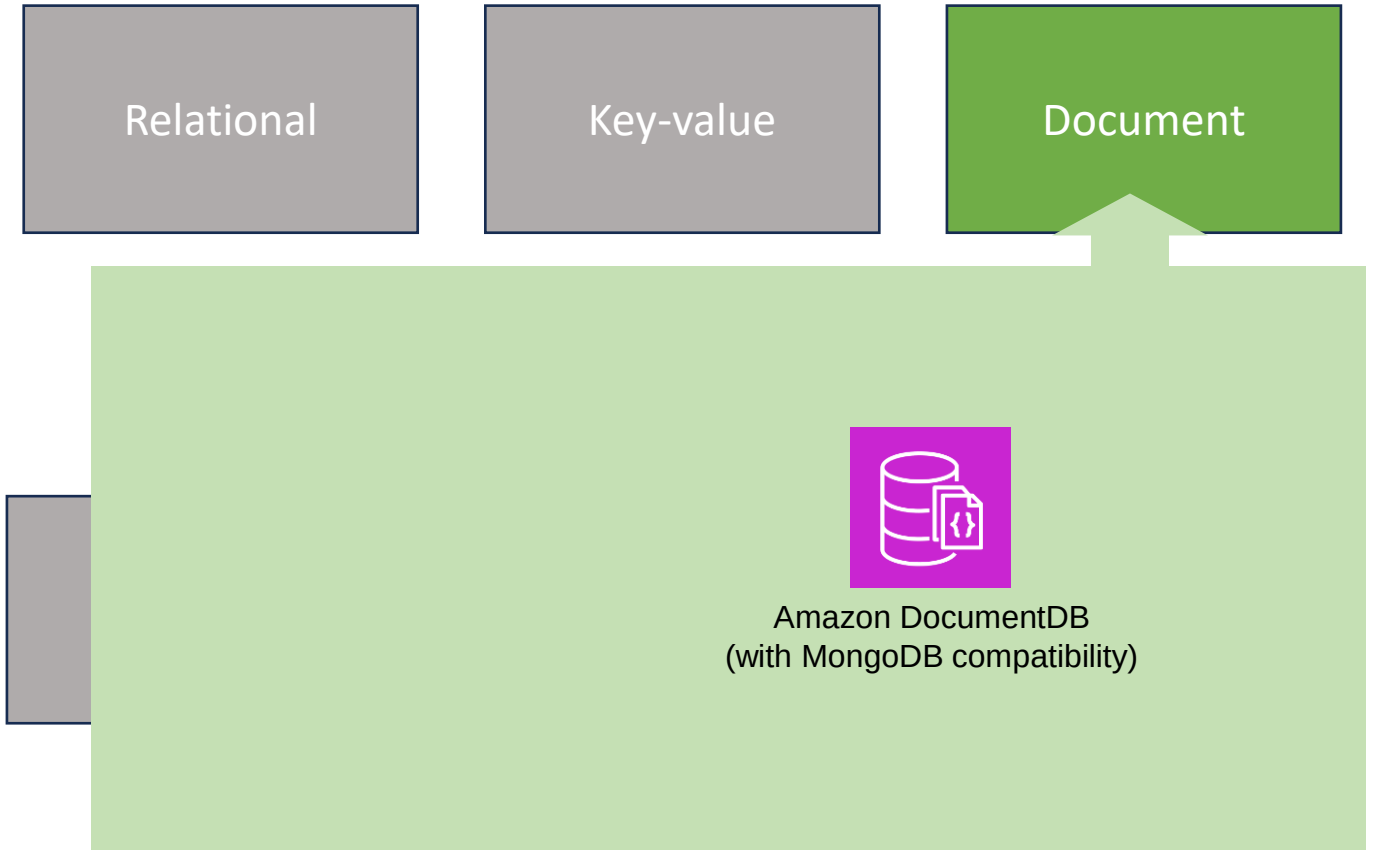
AWS Databases



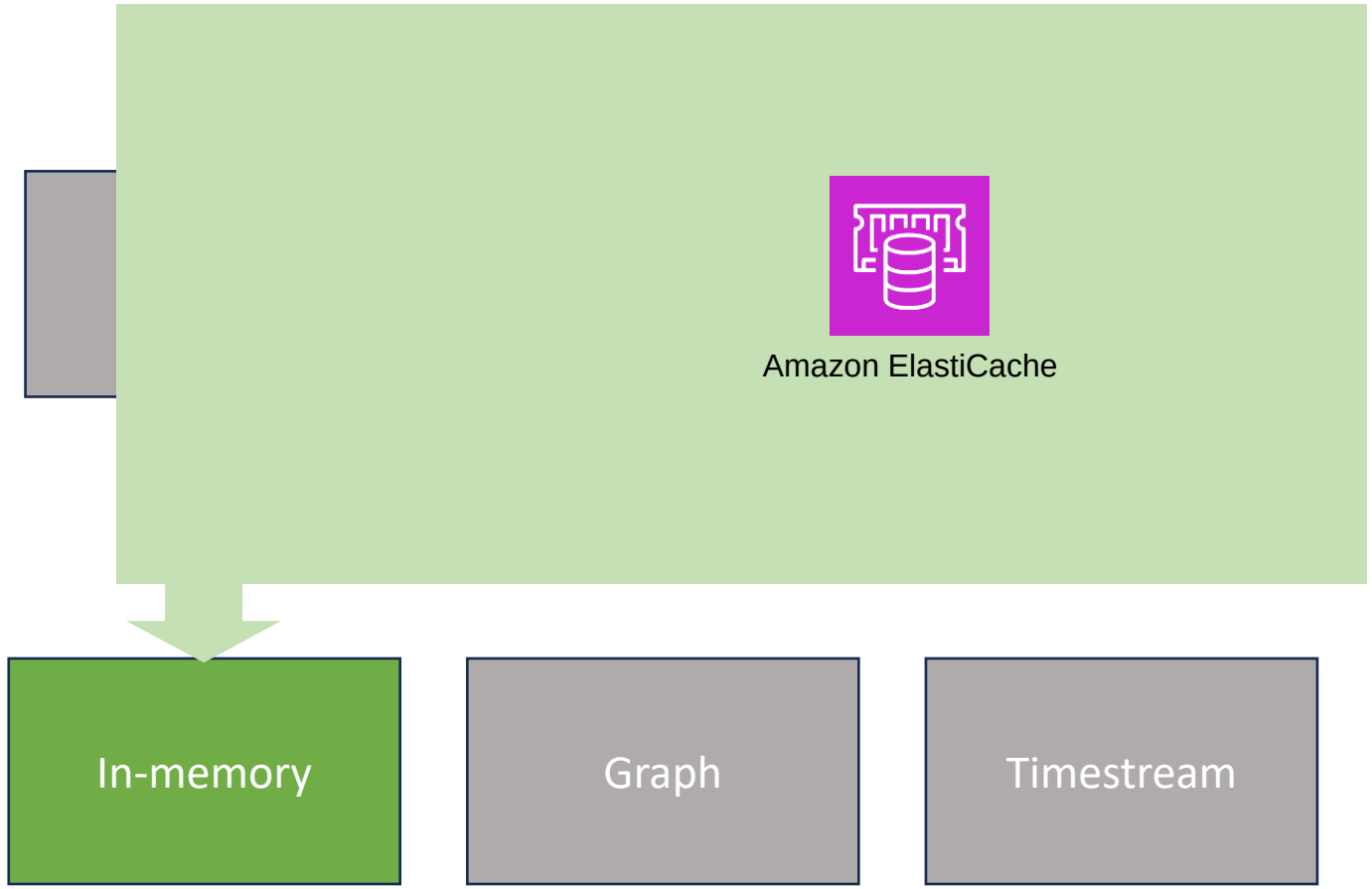
AWS Databases



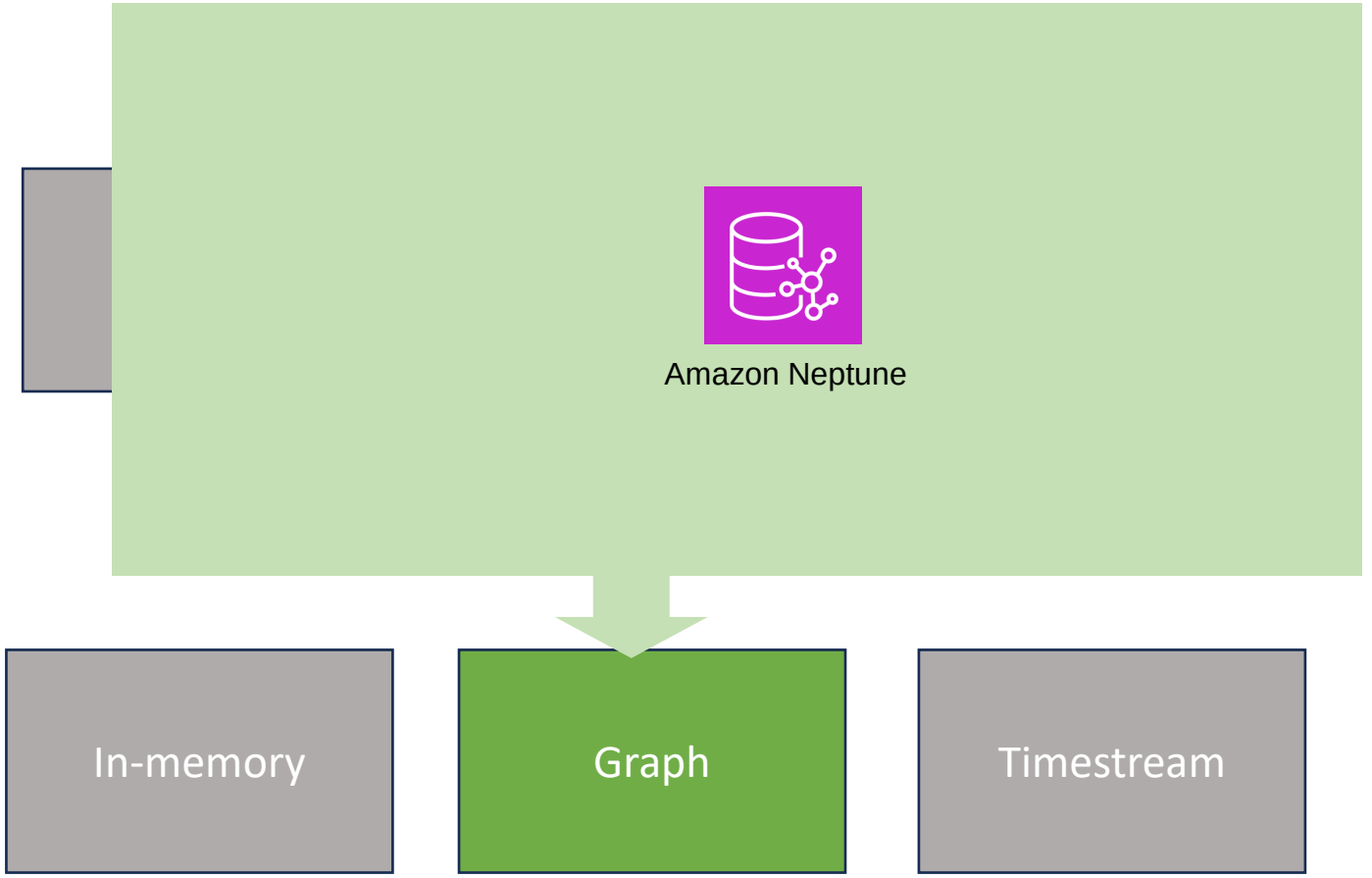
AWS Databases



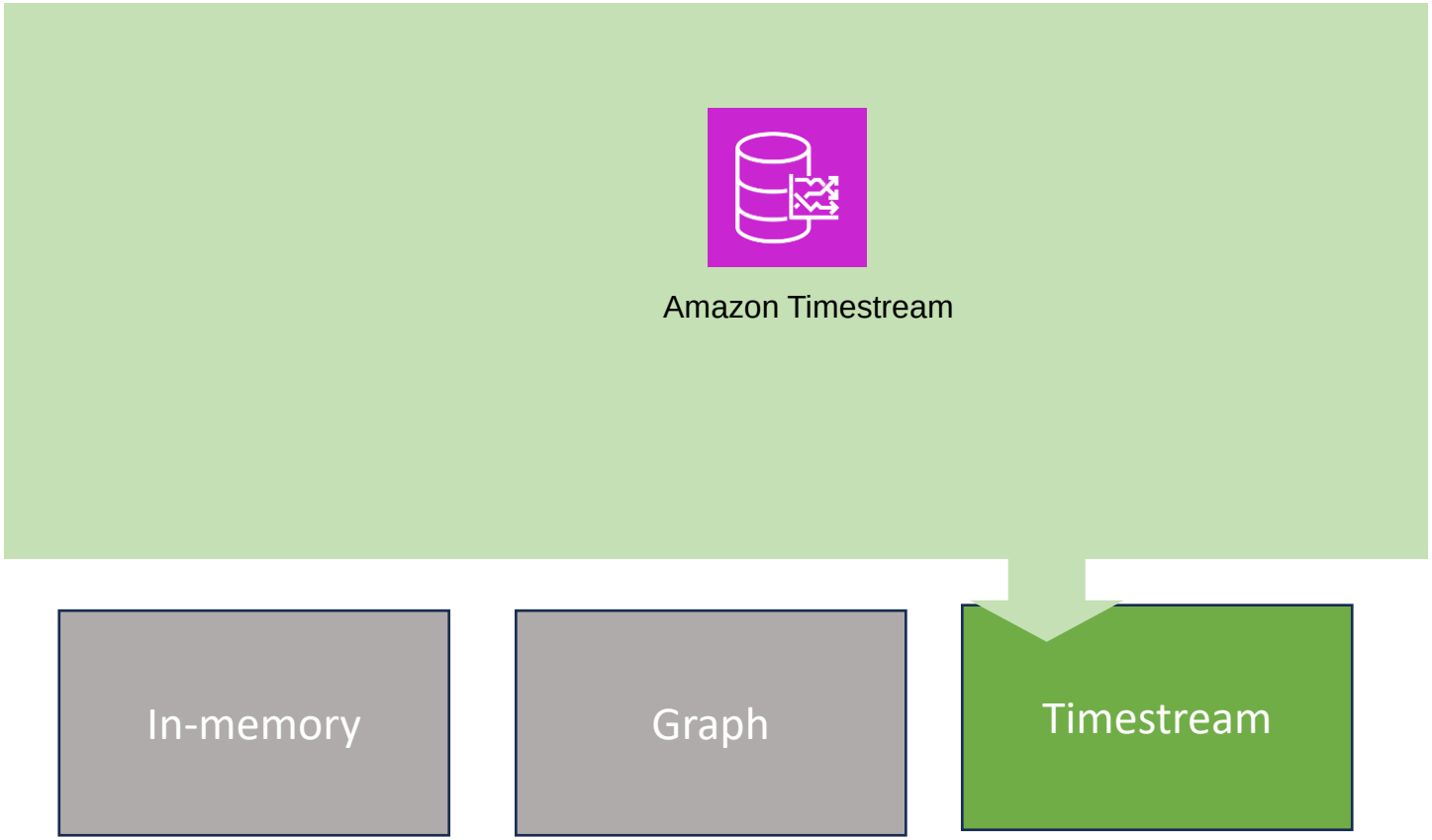
AWS Databases



AWS Databases



AWS Databases



AWS Databases

Today's session

Relational

Amazon Relational Database Service (Amazon RDS)

- Theory
- AWS Educate: Lab 1

Key-value

Amazon DynamoDB

- Theory
- AWS Educate: Lab 2

Graph

Amazon Neptune

- Theory
- Use case: AWS Blog post

Introduction to Amazon RDS

Amazon Relational Database Service (RDS)

Introduction to Amazon RDS

Amazon Relational Database Service (Amazon RDS) is a fully managed relational database in the cloud. It supports multiple database engines to suit the needs of most relational database use cases.

There is no need to provision infrastructure or to install and maintain database software.

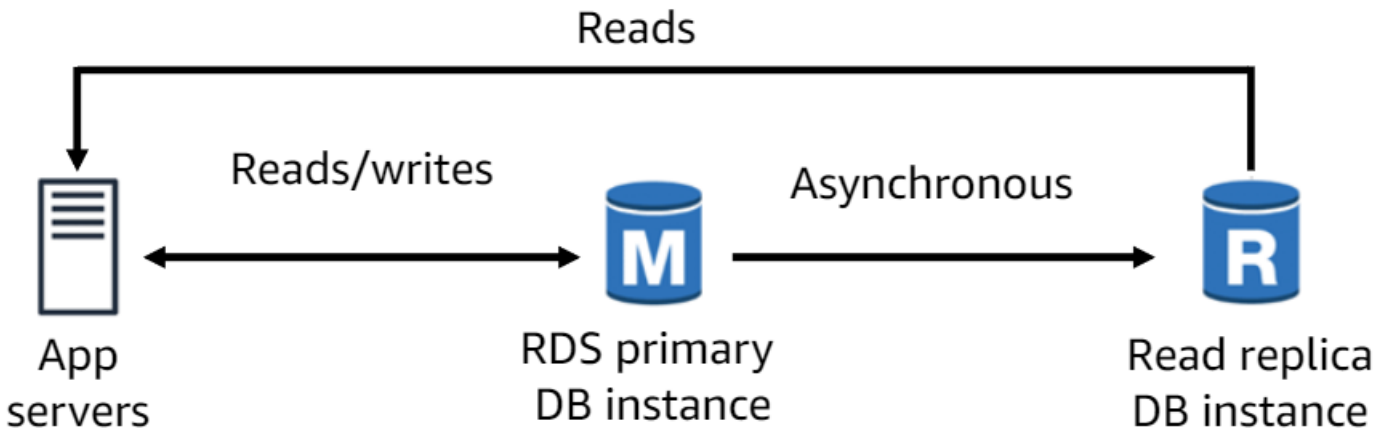
There are 6 database engines to choose from

Amazon Relational Database Service (RDS)

Use cases for Amazon RDS

In this online transaction processing application (OLTP), Amazon RDS is used with a read replica which is asynchronously updated. Amazon RDS read replica make it easy to elastically scale out for read-heavy database workloads beyond the capacity constraints of a single database instance.

You can create one or more replicas of a given source database instance and serve high-volume application read traffic from multiple copies of your data, thereby increasing aggregate read throughput.



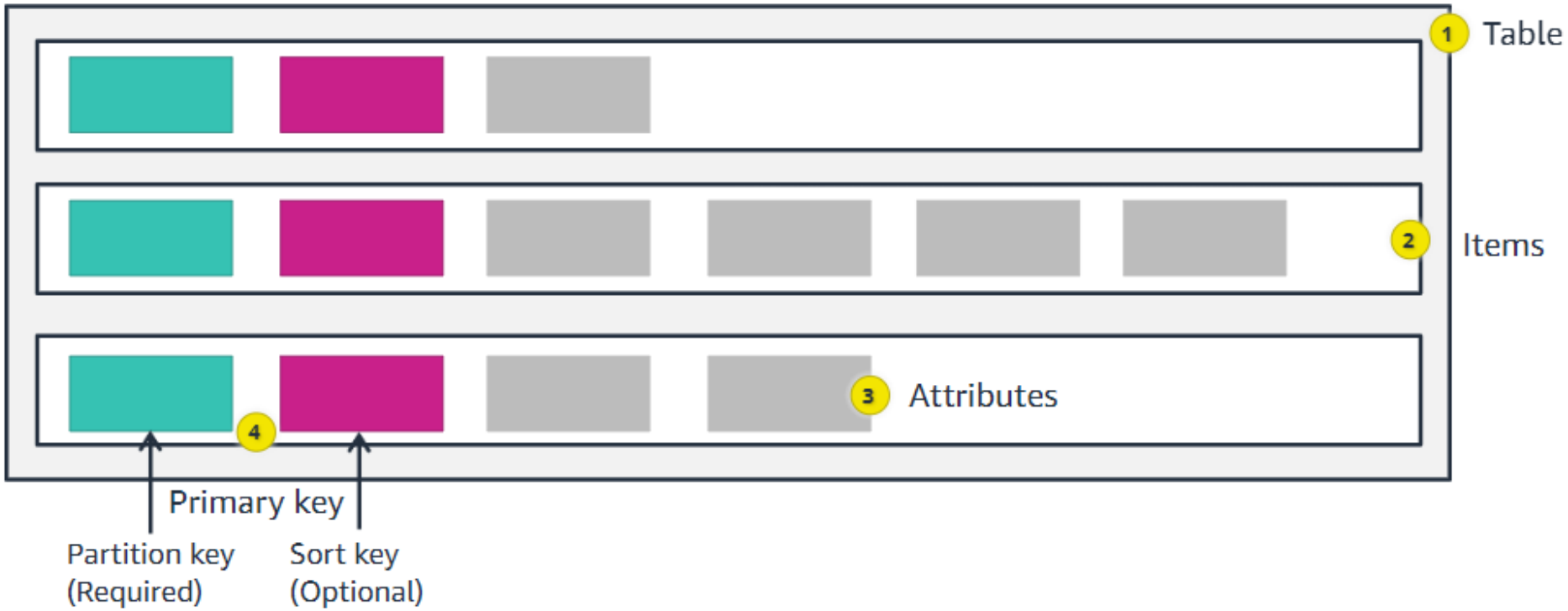
Introduction to Amazon DynamoDB

Amazon DynamoDB

Amazon DynamoDB is a fast, non-relational database service that makes cost-effective to store data. It is a fully managed, multi-region, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications.

Amazon DynamoDB Components

- A **table** is a collection of **items**, and each item is a collection of **attributes**.
- DynamoDB tables do no have fixed schemas, and each item might have a different number of attributes.



Amazon DynamoDB: Primary Key

Amazon DynamoDB uses **primary keys** to uniquely identify each item in a table and secondary indexes to provide more querying flexibility. Therefore, the primary key uniquely identifies each item in the table so that no two items can have the same key.

Primary key	Attributes		
Actor			
Tom Hanks	Role	Year	Genre
	Chuck Noland	2000	Drama
Tim Allen	Role	Year	Genre
	Buzz Lightyear	1995	Children's

Amazon DynamoDB: Composite keys

Composite primary key (partition and sort key): A composite primary key consists of two attributes. The first attribute is the *partition key*, and the second attribute is the *sort key.*

The composite of the two keys must be unique across the entire table. The composite key must be unique.

Primary key		Attributes		
Actor **(partition)**	Movie **(sort)**			
Tom Hanks	Cast Away	Role	Year	Genre
		Chuck Noland	2000	Drama
Tim Allen	Toy Story	Role	Year	Genre
		Buzz Lightyear	1995	Children's
Tom Hanks	Toy Story	Role	Year	Genre
		Woody	1995	Children's

Amazon DynamoDB consistency models

When reading data from DynamoDB, users can specify whether they want the read to be eventually consistent or strongly consistent.

Eventually consistent reads (the default) – The eventual consistency option maximizes your read throughput. However, an eventually consistent read might not reflect the results of a recently completed write. All copies of the data usually reach consistency within a second. Repeating the read after a short time should return the updated data.

Strongly consistent reads: In addition to eventual consistency DynamoDB also give you the flexibility and control to request a strong consistent read. A strongly consistent read returns a result that reflects all writes that received a successful response before the read.

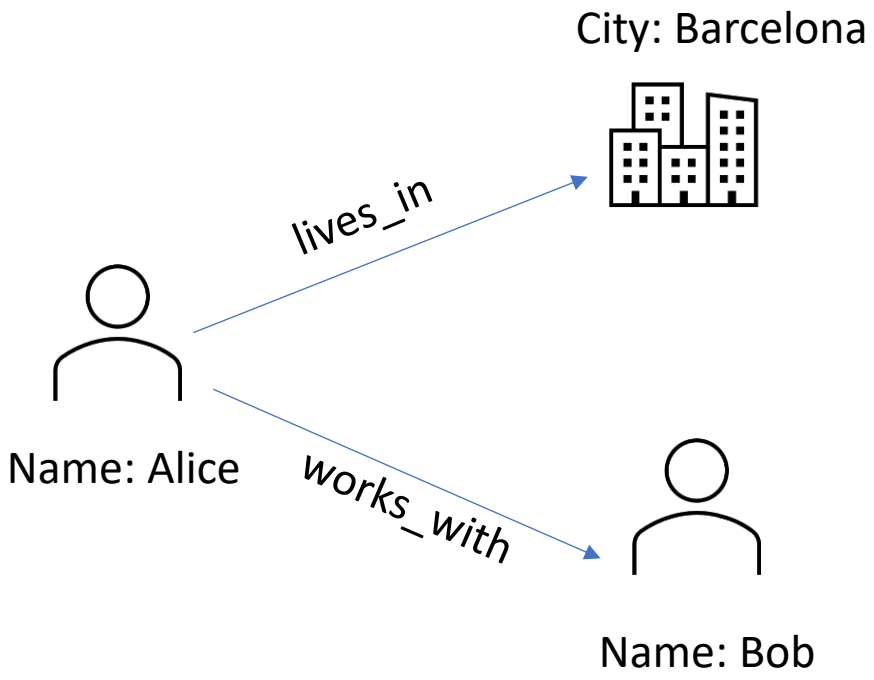
Amazon DynamoDB: Read and Write capacity modes

Provisioned mode	On-demand mode
• You have predictable application traffic. • You run applications whose traffic is consistent or ramps gradually. • You can forecast capacity requirements to control cost.	• You create new tables with unknown workloads. • You have unpredictable application traffic. • You prefer the ease of paying for only what you use.

Introduction to Graph Databases and Amazon Neptune

laSalle
UNIVERSIDAD RAMON LLULL

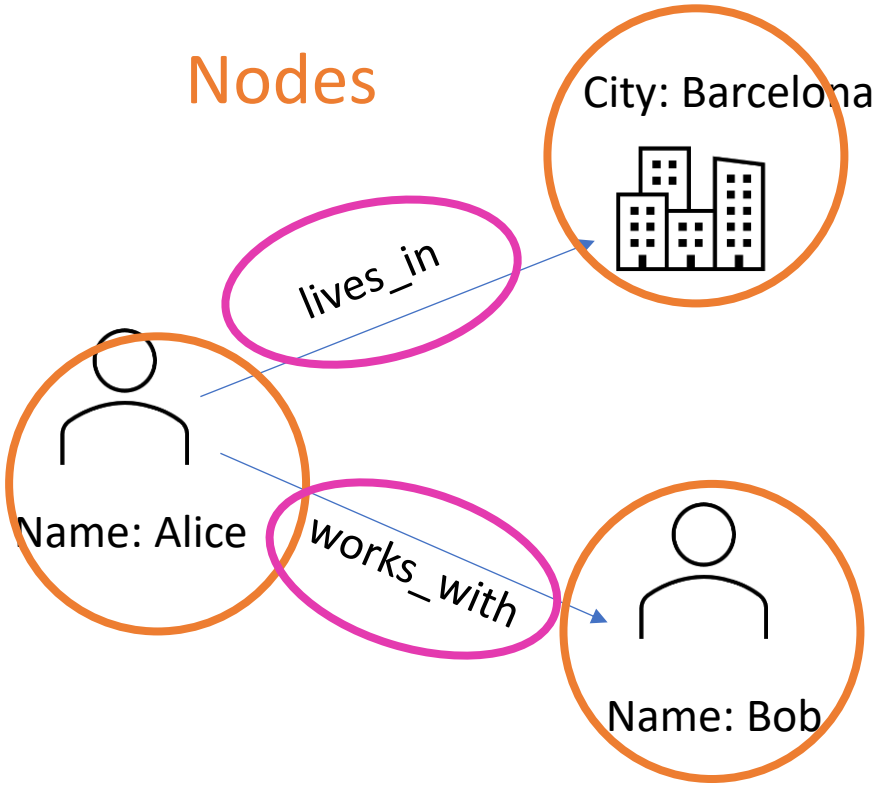
Graph Databases with Amazon Neptune



Graphs model data based on **relationships**.

Graphs explore relationships and patterns in **connected data.**

Graph Databases with Amazon Neptune



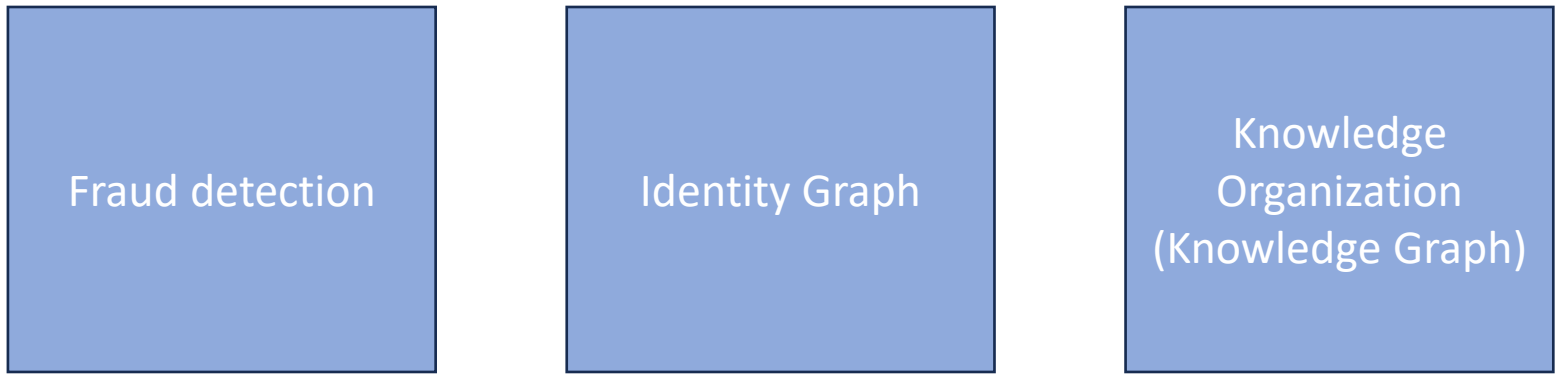
Graphs model data based on **relationships**.

Graphs explore relationships and patterns in **connected data.**

laSalle
UNIVERSIDAD RAMON LLULL

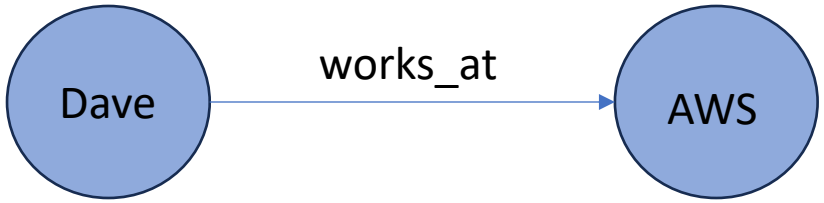
Graph use cases

- We need to get better at detecting fraud.
- My customers want better or more personalized recommendations.
- We need to connect our siloed data sources.
- We have multiple websites/applications, and we need to link customer identities in these systems.
- Our machine learning algorithms need improvement.



Query Languages

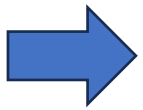
Graph query languages are optimized to use connections to move through a network.




```
g.V().has('person','Dave')
```

Person	Age
Dave	30

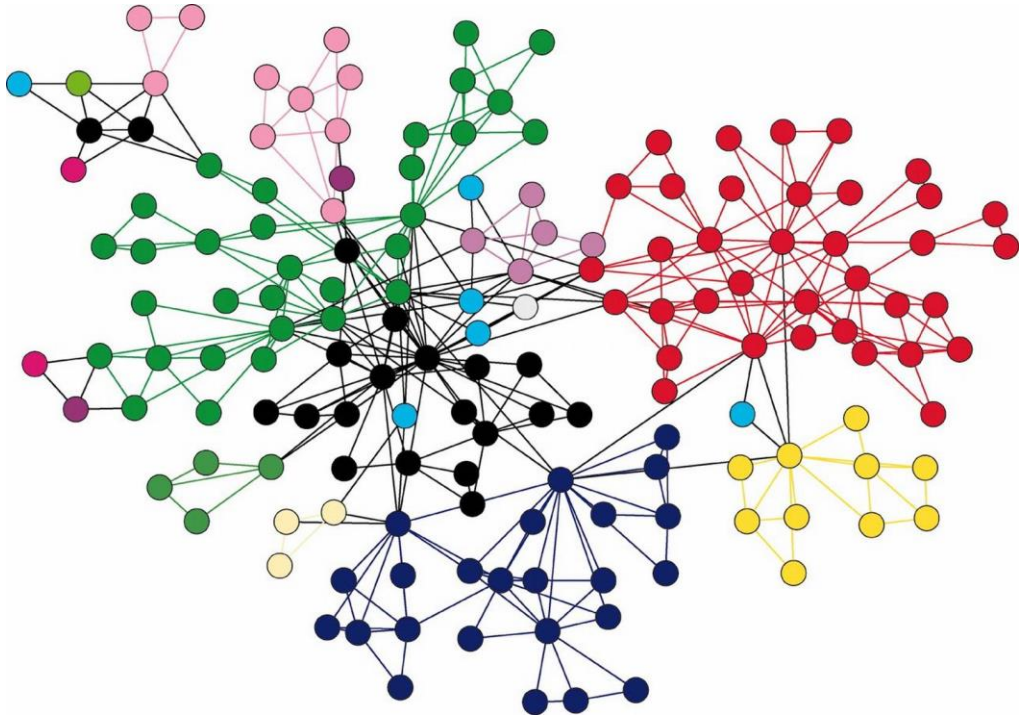
Company	StartDate
AWS	30/05/2021



Person	Company
Dave	AWS

Schema flexibility

The schema flexibility makes adding new data sources easy.



Why use Amazon Neptune?

Traditional databases are not optimized for connections

Self-managed solutions become complex

Evolution at Scale

Why use Amazon Neptune?

Fully managed, purpose-built graph database in the cloud.

- Optimized to store and map billions of relationships.
- Enables real-time navigation of connections with millisecond query response time.
- Supports open standard query languages openCypher, Gremlin and SPARQL.

Amazon Neptune

Cost effective	No hardware management	Instant provisioning	Scaling	Security and Compliance

Leading graph models and languages

Property graph	Resource Description Framework (RDF)
openCypher (SQL-inspired) Apache TinkerPop Gremlin $G = (V, E)$	RDF SPARQL
• Edges representing real world entities and edge representing connections. • Ability to associate properties to the nodes and to the edges themselves.	• RDF represents data as a set of triples. • Each triple has subject, predicate and object.

Amazon Neptune Architecture

Application Layer

- Fraud detection
- Identity Graph
- Knowledge Organization (Knowledge Graph)

Compute Layer

- 1 writer and up to 15 read replicas

Shared Storage Layer

- 6 copies of data across 3AZs
- Data up to 128TiB
- Automated backup and restore
- Database fast clone

Service Features


Bulk load from S3


Neptune Streams


Status Endpoint


Query Profile and Explain


Read replica auto scaling

Service Integrations


AWS Backup


Neptune ML


Amazon OpenSearch

Use case

Building a Knowledge Graph

Amazon Neptune

[Building a knowledge graph in Amazon Neptune using Amazon Comprehend Events | AWS Database Blog]

[Announcing the launch of Amazon Comprehend Events | AWS Machine Learning Blog]

Reading time	30 min
Discussion time	15 min

Resources

Resources

The content of this section was based on the following resources:

- AWS Educate: Getting Started with Databases
- Amazon Neptune Dive Deep Session

However, given the number of Databases that exist in AWS, if you would like to learn more about them, the following resources are helpful for further learning.

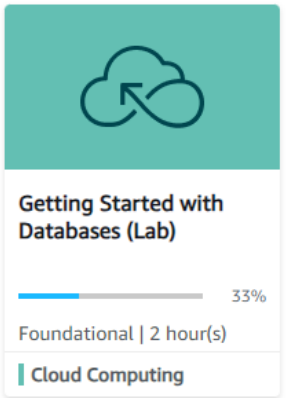
- AWS Databases Workshops

Part 2: Hands-on

Hands-on Labs

AWS Educate Labs

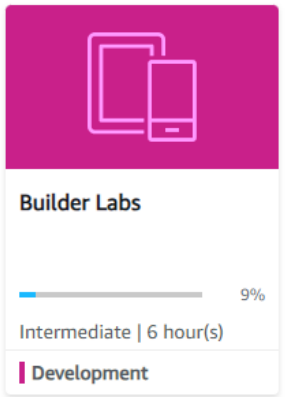
1	Getting Started with Databases
Link	Getting Started with Databases (instructure.com)
Lab duration	~ 1hour
Description	Traditionally, creating a database can be a complex process that requires either a database administrator or a systems administrator. In the cloud, you can reduce the number of steps that you take during this process by using Amazon Relational Database Service (Amazon RDS). In this lab, you learn how to use Amazon RDS to provision a MySQL database and perform a few basic administrative actions.



laSalle
UNIVERSIDAD RAMON LLULL

Hands-on Labs

AWS Educate Labs

2	Builder Labs: Adding Dynamic Data to Your Application with DynamoDB
Link	Builder Labs - DynamoDB lab
Lab duration	~ 45 min
Description	In this lab, you will learn how to create an Amazon DynamoDB table and use it to support and existing JavaScript application. 

laSalle
UNIVERSIDAD RAMON LLULL

Thank you

laSalle
UNIVERSIDAD RAMON LLULL