ACADEMIND

Databases on AWS

Storing application data

- Self-managed Databases vs Managed Database Services
- Understanding & Using RDS (incl. Aurora)
- Understanding & Using DynamoDB

ACADEMIND

So Many Options!

Different Types of Databases

e.g., SQL vs NoSQL

Different data or workload requirements favor different database types

AWS allows you to run & use all database types

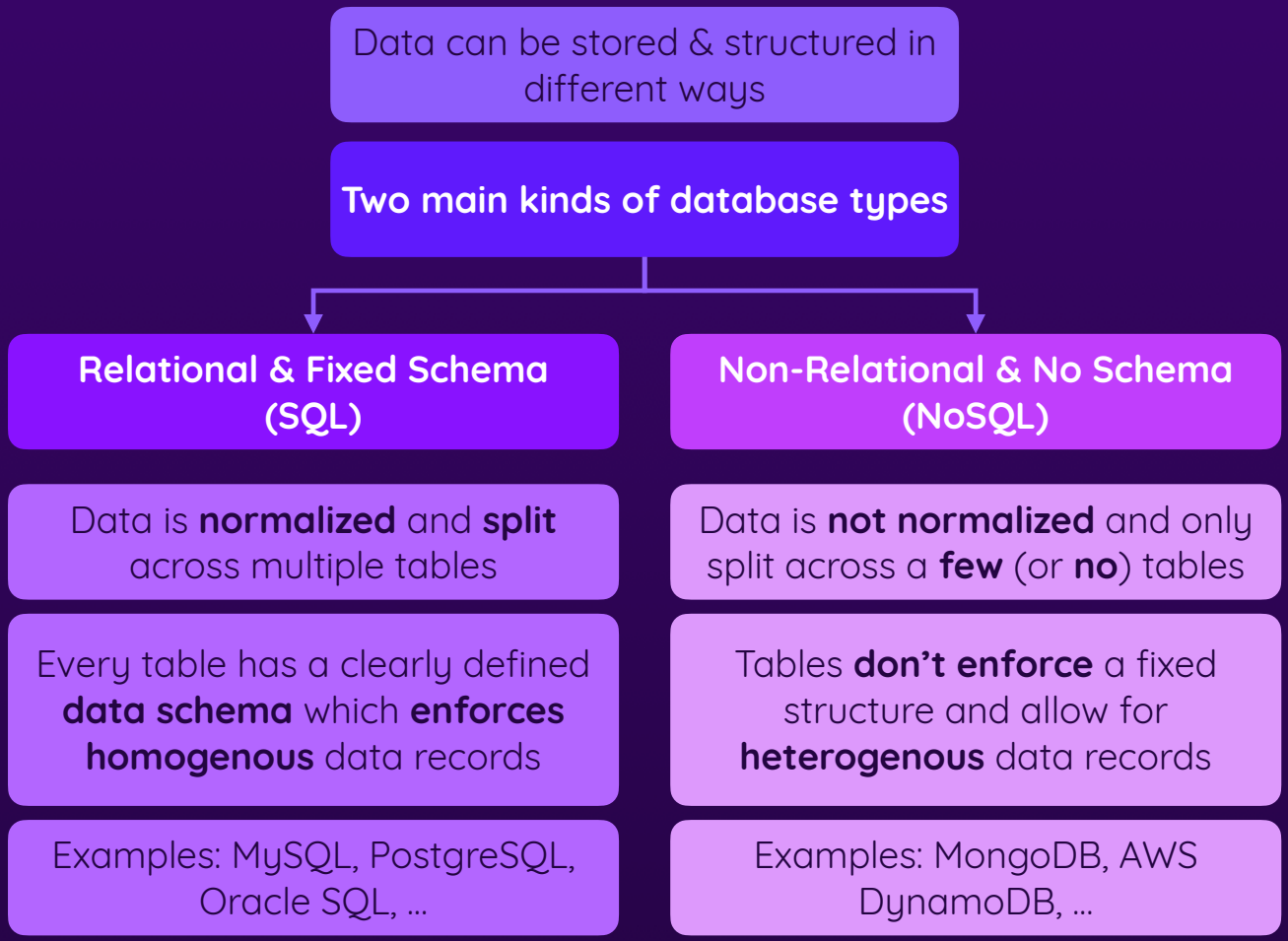
Different Hosting Options

Self-hosted vs managed

You can install + operate databases yourself (e.g., on EC2 instances) or let AWS do that

AWS supports both options

Database Types: SQL vs NoSQL



Self-Hosted: Advantages & Disadvantages

Self-hosted Databases

Install & operate database software manually

e.g., on EC2 instances

Full control but also full responsibility

Important duties: Keep software patched, manage backups etc.

Managed Databases

Let AWS manage setup & database operations

Key services: **RDS**, **DynamoDB**

Less control but also less responsibility

You define general rules but AWS does the heavy lifting

Relational Database Service (RDS)

Managed SQL Databases

Choose Database Engine

- e.g., MySQL, PostgreSQL
- Database version
- Auto-update settings

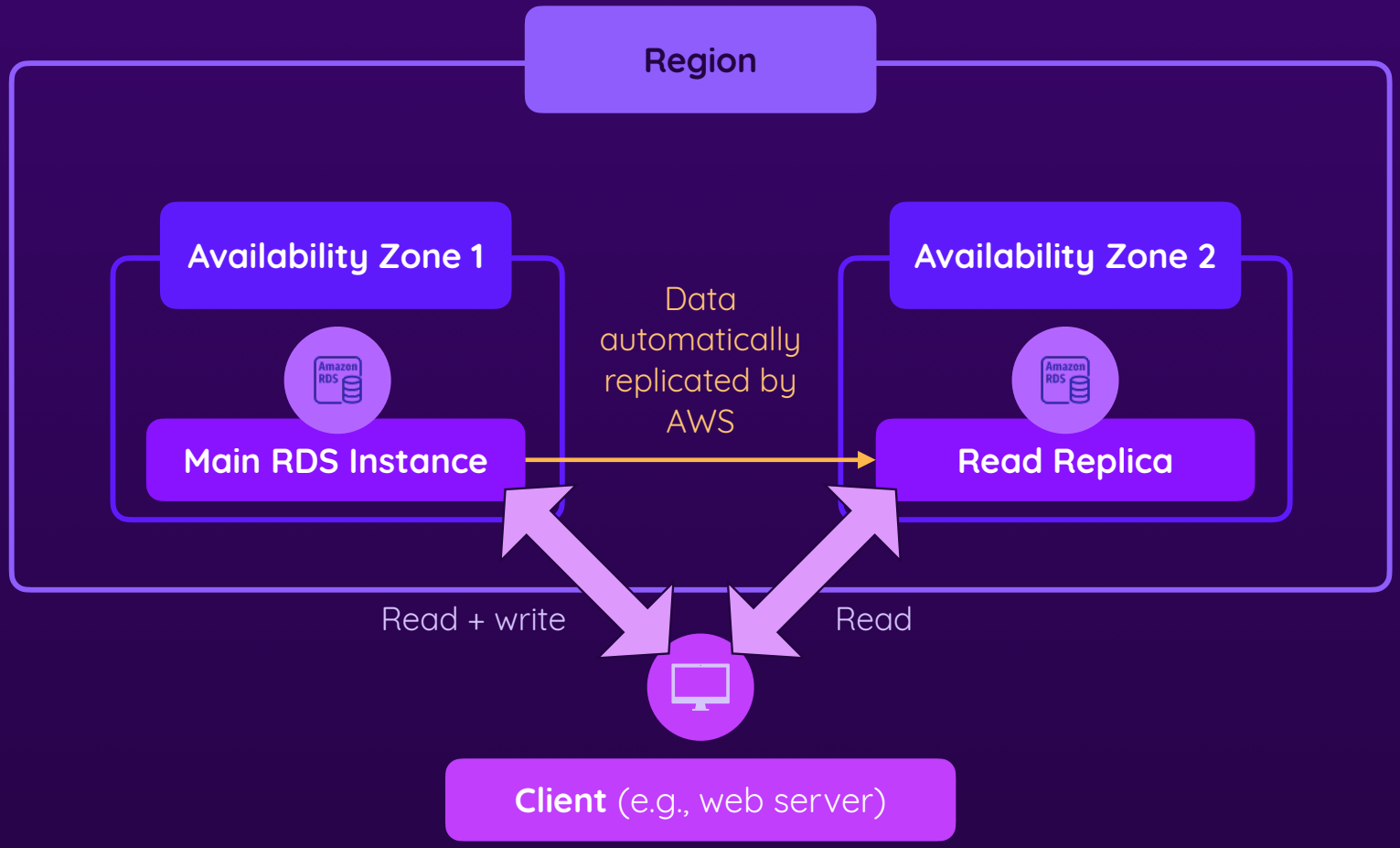
Choose Hardware & Network Configuration

- Instance class (hardware profile)
- VPC, subnet & security group

Configure Database Server

- Login credentials & port
- Replication (for high availability & performance)
- Monitoring settings
- Encryption & backup settings

High Availability Thanks To Replication



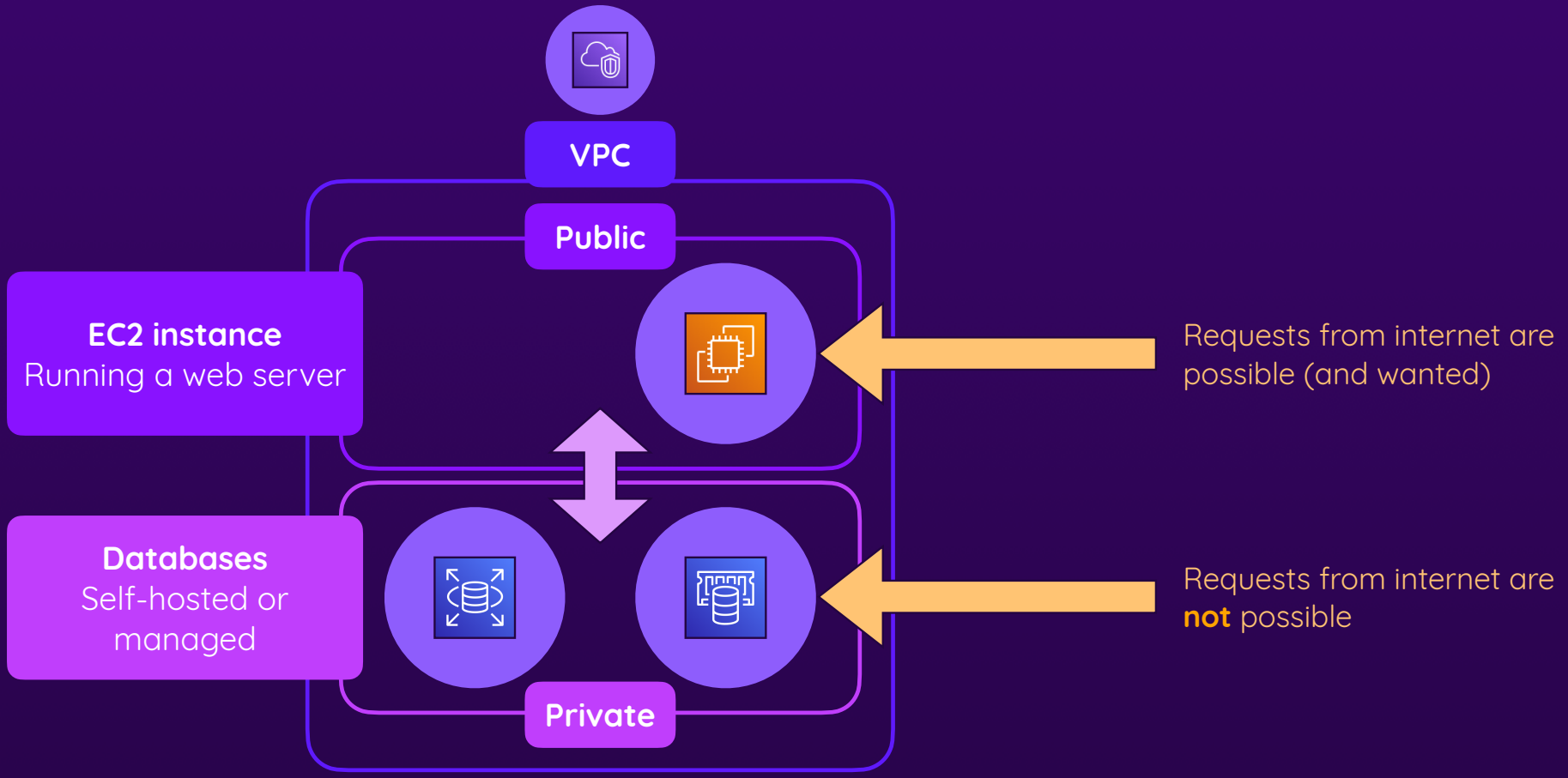
ACADEMIND

Amazon Aurora

Amazon's own SQL database engine

MySQL & PostgreSQL compatible database engine with great scalability & performance

Databases & VPCs

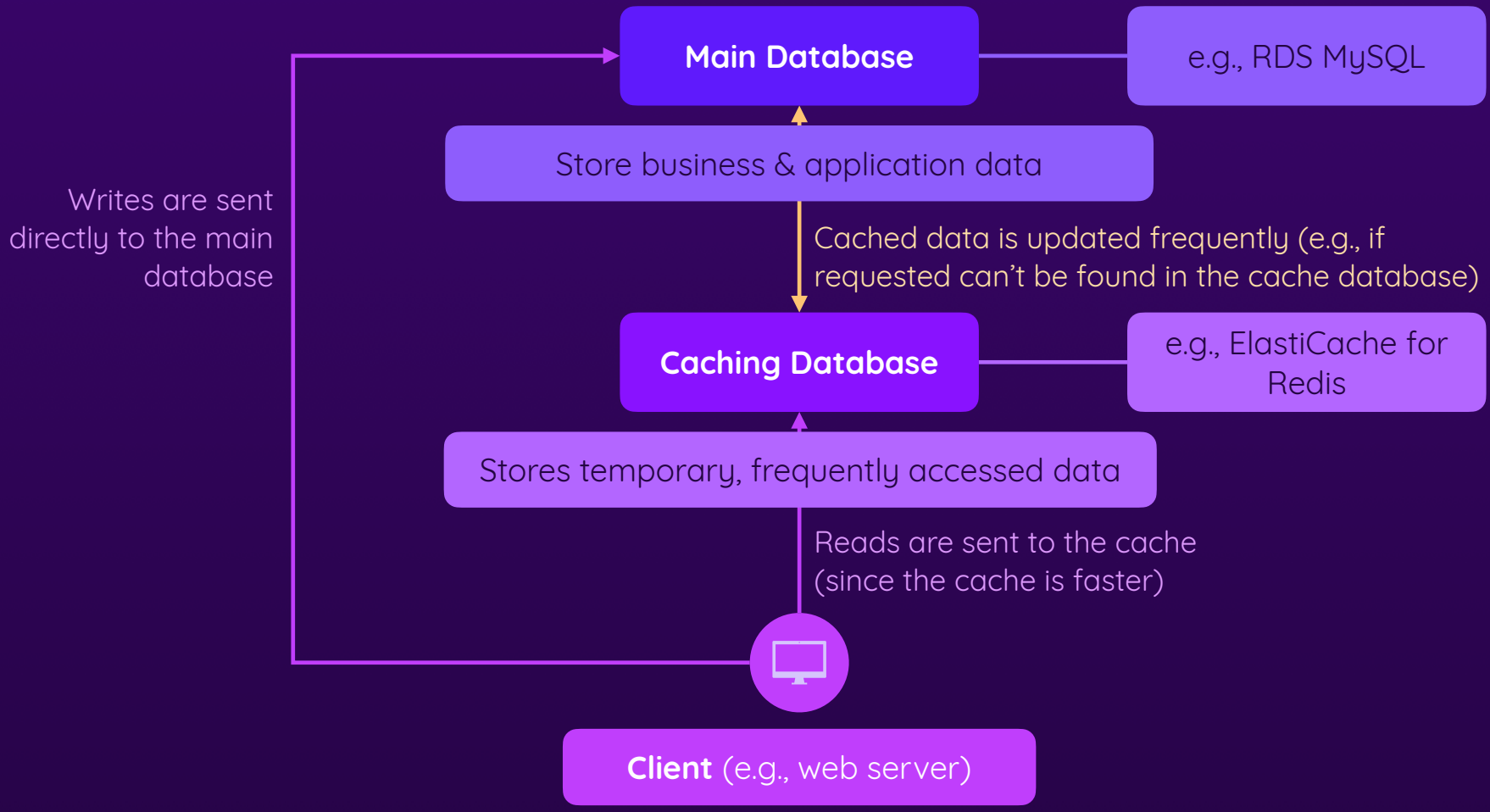


Caching Data with ElastiCache

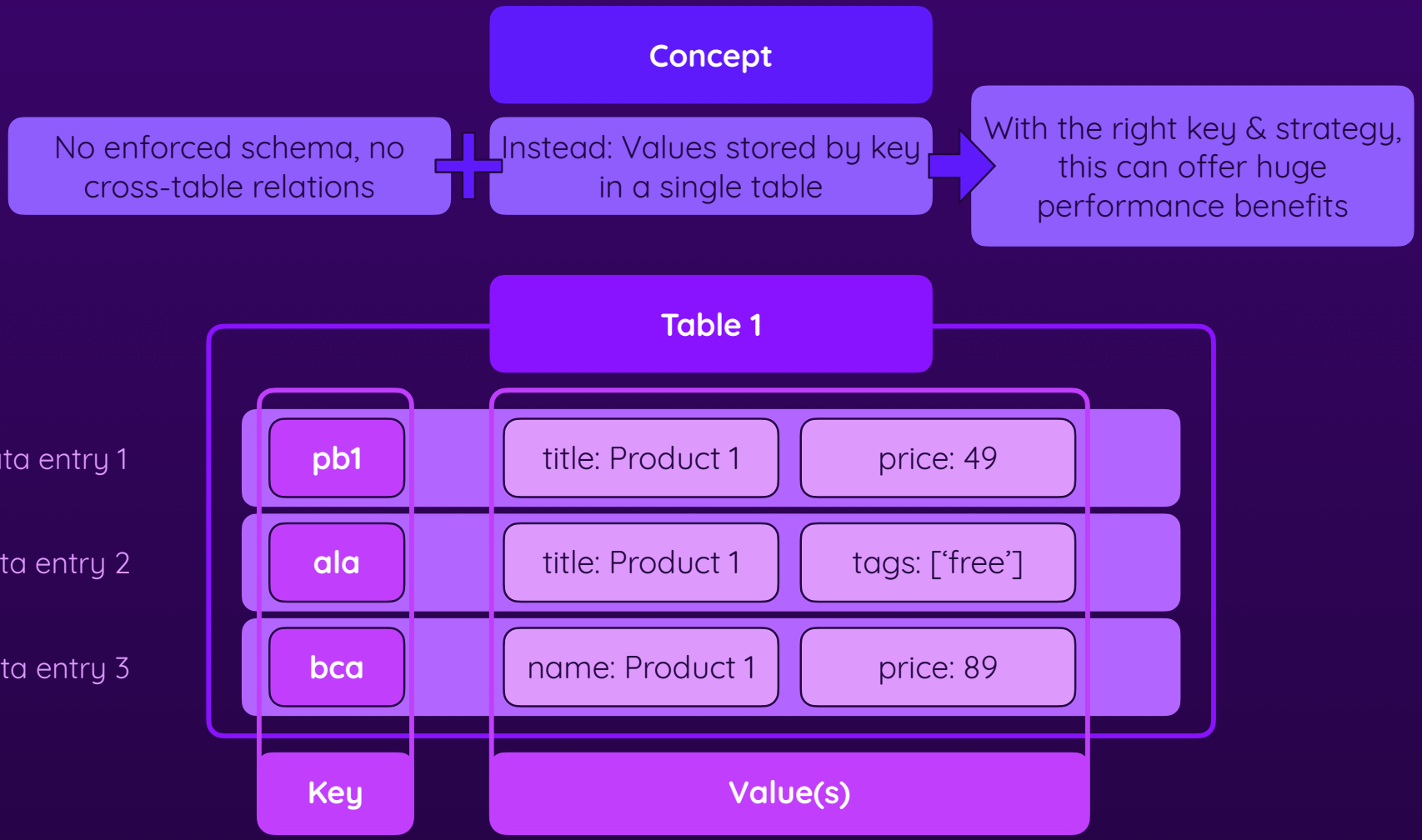
A fully managed in-memory caching database

Choose Database Engine	Choose Hardware & Network Configuration	Configure Database Server
Redis or Memcached	Node type (hardware profile)	Encryption & backup settings
Optionally enable cluster mode (for scaling)	VPC, subnet & security group	Maintenance settings
Choose engine version		Monitoring settings

What Is "Caching"?



SQL Alternative: NoSQL Key-Value Stores



Understanding DynamoDB

A fully managed NoSQL key-value database

Create Tables

- Set name & key(s)
- Set expected read / write capacities (or on-demand)
- Choose encryption settings

Manage Data

- Write & read via AWS API / CLI / SDK
- Configure backups

More DynamoDB Features

Streams

- Time-ordered series of database item changes
- Subscribe to process item changes

Global Tables

- Fully managed multi-region database
- High availability thanks to automatic replication
- Great performance thanks to global reach

DAX

- Managed in-memory cache for DynamoDB
- Accelerates database queries

Other Database Services

MemoryDB

Persistent in-memory storage

DocumentDB

Document (nested data structure) database

Keyspaces

Wide column database (flexible column formats)

Neptune

Graph database (complex data relations)

Timestream

Time series database

Quantum Ledger Database

Immutable log of data changes

AWS Backup

Centralized Backup Management

Create Plan

- Use template or create custom
- Set frequency, retention period etc.
- Define destination & timeframe

Manage Resources & Backups

- Assign resources (e.g., RDS cluster) to backup plan
- Access & restore backups if necessary

Summary

Different Database Services

- Self-hosted (on EC2) vs managed services
- SQL (RDS, Aurora) vs NoSQL (DynamoDB, DocumentDB, ...)
- Different database for different workloads / data requirements

RDS, Aurora & ElastiCache

- Managed relational database services
- Configure database cluster hardware, network & behavior
- Leverage built-in scaling & availability (replication) features
- Access databases via HTTP endpoints / SQL queries

DynamoDB & More

- DynamoDB: Managed high-performance key-value database
- Define partition keys & read / write capacity (or on-demand)
- Access DynamoDB data via AWS API / SDKs
- Other databases for specific use-cases