AT82.02

DATA MODELING AND MANAGEMENT

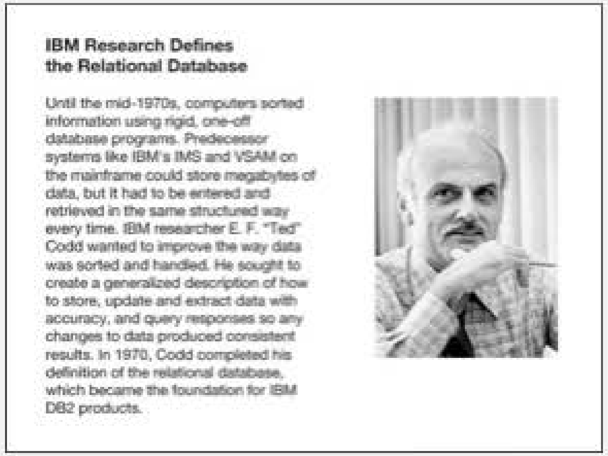
UNIT 6-1: CHOOSING THE RIGHT DATABASE TECHNOLOGY

CHUTIPORN ANUTARIYA (CHUTI AT AIT DOT AC DOT TH)

Co-funded by the
Erasmus+ Programme
of the European Union

Relational DBs are the most successful technology for the last 50 years

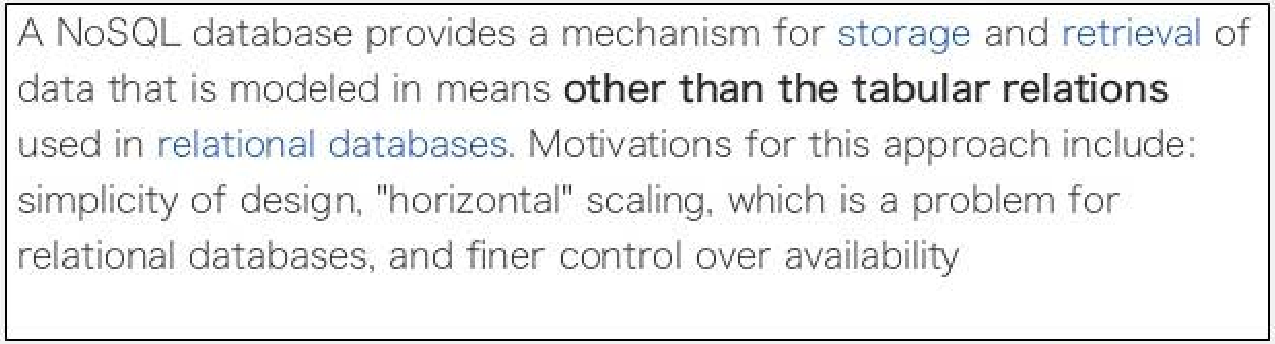
Relational



IBM Research Defines the Relational Database

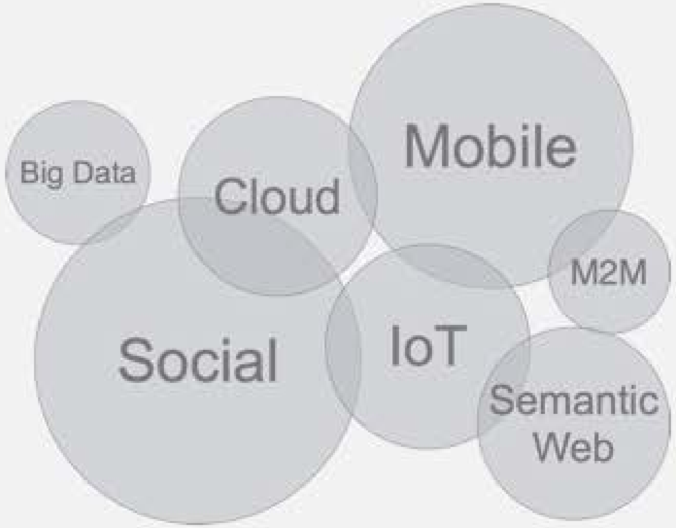
Until the mid-1970s, computers sorted information using rigid, one-off database programs. Predecessor systems like IBM's IMS and VSAM on the mainframe could store megabytes of data, but it had to be entered and retrieved in the same structured way every time. IBM researcher E. F. "Ted" Codd wanted to improve the way data was sorted and handled. He sought to create a generalized description of how to store, update and extract data with accuracy, and query responses so any changes to data produced consistent results. In 1970, Codd completed his definition of the relational database, which became the foundation for IBM DB2 products.

NoSQL



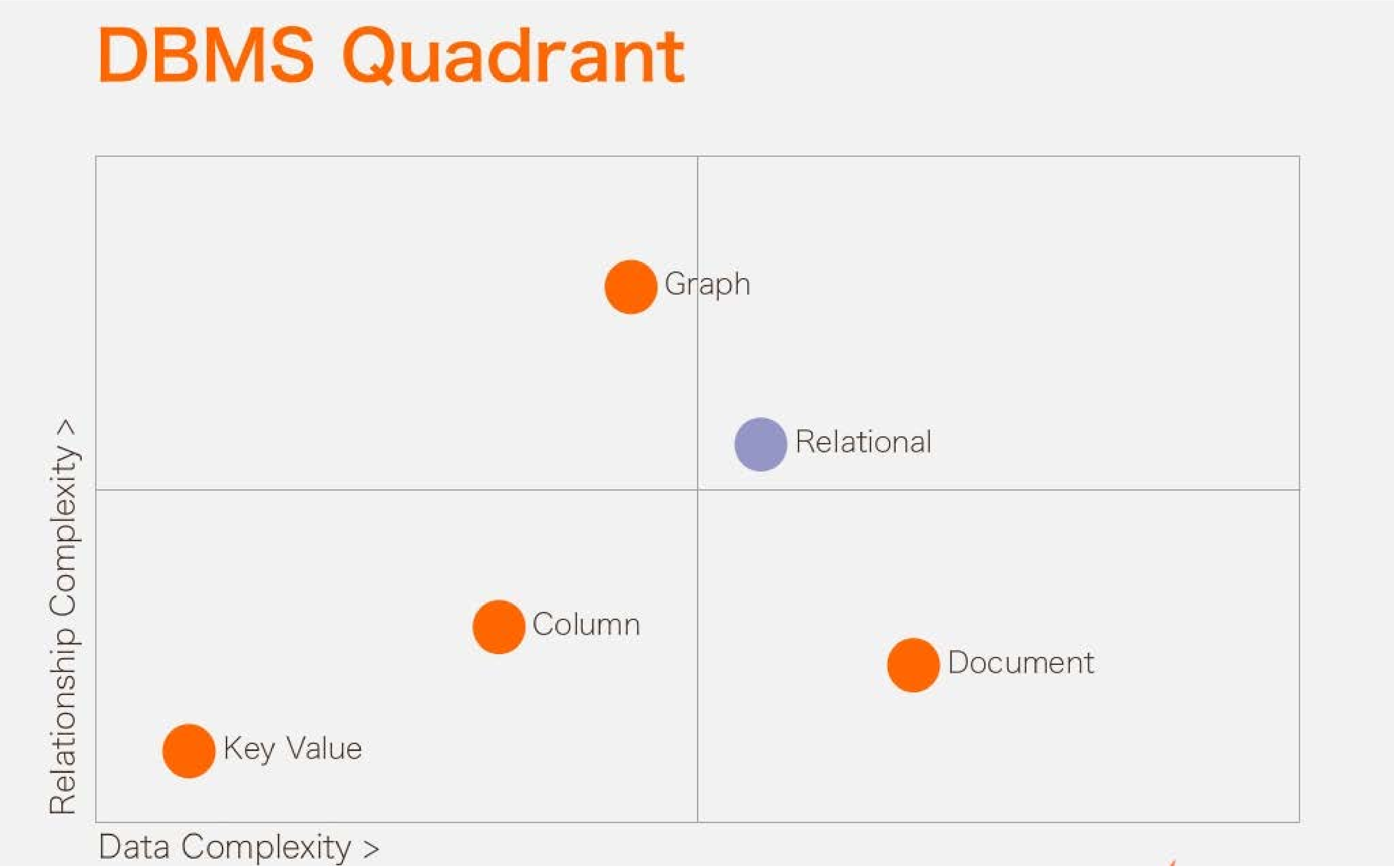
A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means **other than the tabular relations** used in relational databases. Motivations for this approach include: simplicity of design, "horizontal" scaling, which is a problem for relational databases, and finer control over availability

Structured Data	→	Unstructured Data
Small Datasets	→	Large Volume
Few Relationships	→	Connected Data
Waterfall Approach	→	Agile Approach
Scale Up	→	Scale Out
CIO	→	Developers



1970 → 2009 → What's Next?

[src: https://www.slideshare.net/lvca/polyglot-persistence-vs-multimodel-databases]



DBMS Quadrant

[src: https://www.slideshare.net/lvca/polyglot-persistence-vs-multimodel-databases]

One size fits all DB may not exist!

Polyglot Persistence

When storing data, it is best to use multiple data storage technologies, chosen based upon the way data is being used by individual applications or components of a single application.

Different kinds of data are best dealt with different data stores.

In short, it means picking the right tool for the right use case.

Polyglot Persistence example

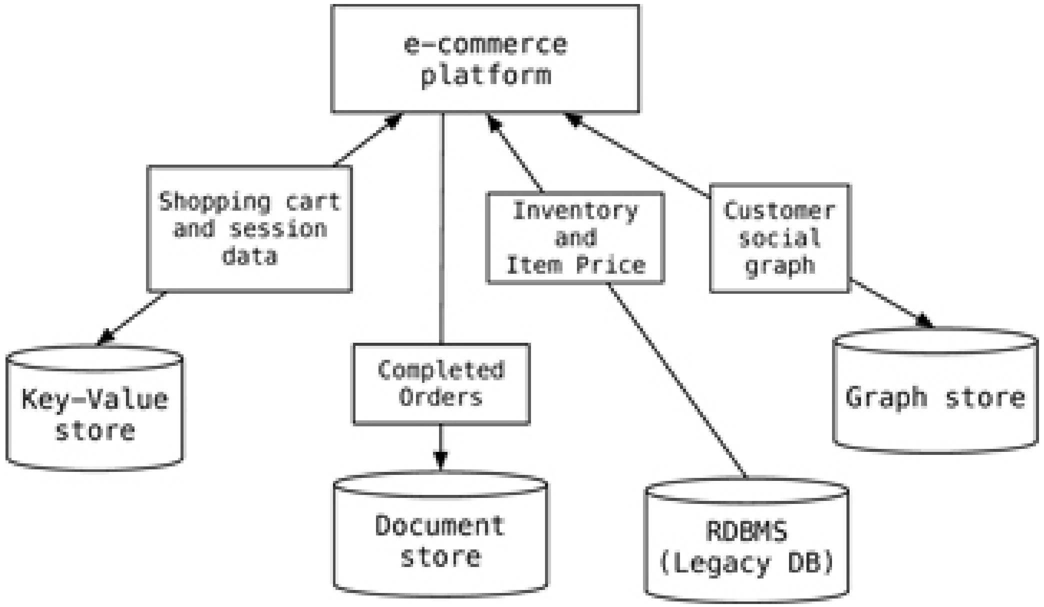
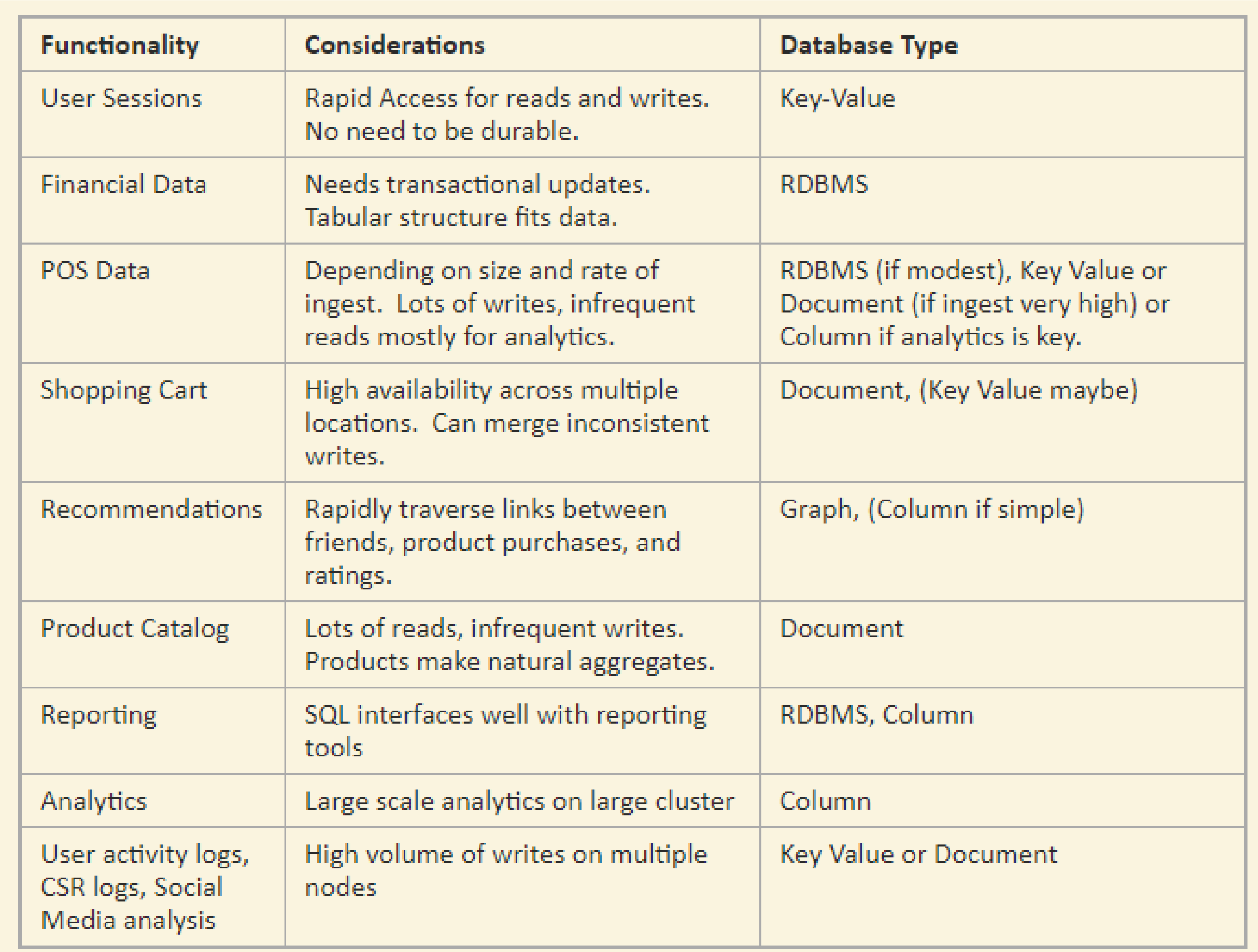


Figure 13.3. *Example implementation of polyglot persistence*

An e-commerce platform will deal with many types of data (i.e. shopping cart, inventory, completed orders, etc) using a mixture of RDBMS solutions with NoSQL solutions

[src: https://www.jamesserra.com/archive/2015/07/what-is-polyglot-persistence/]

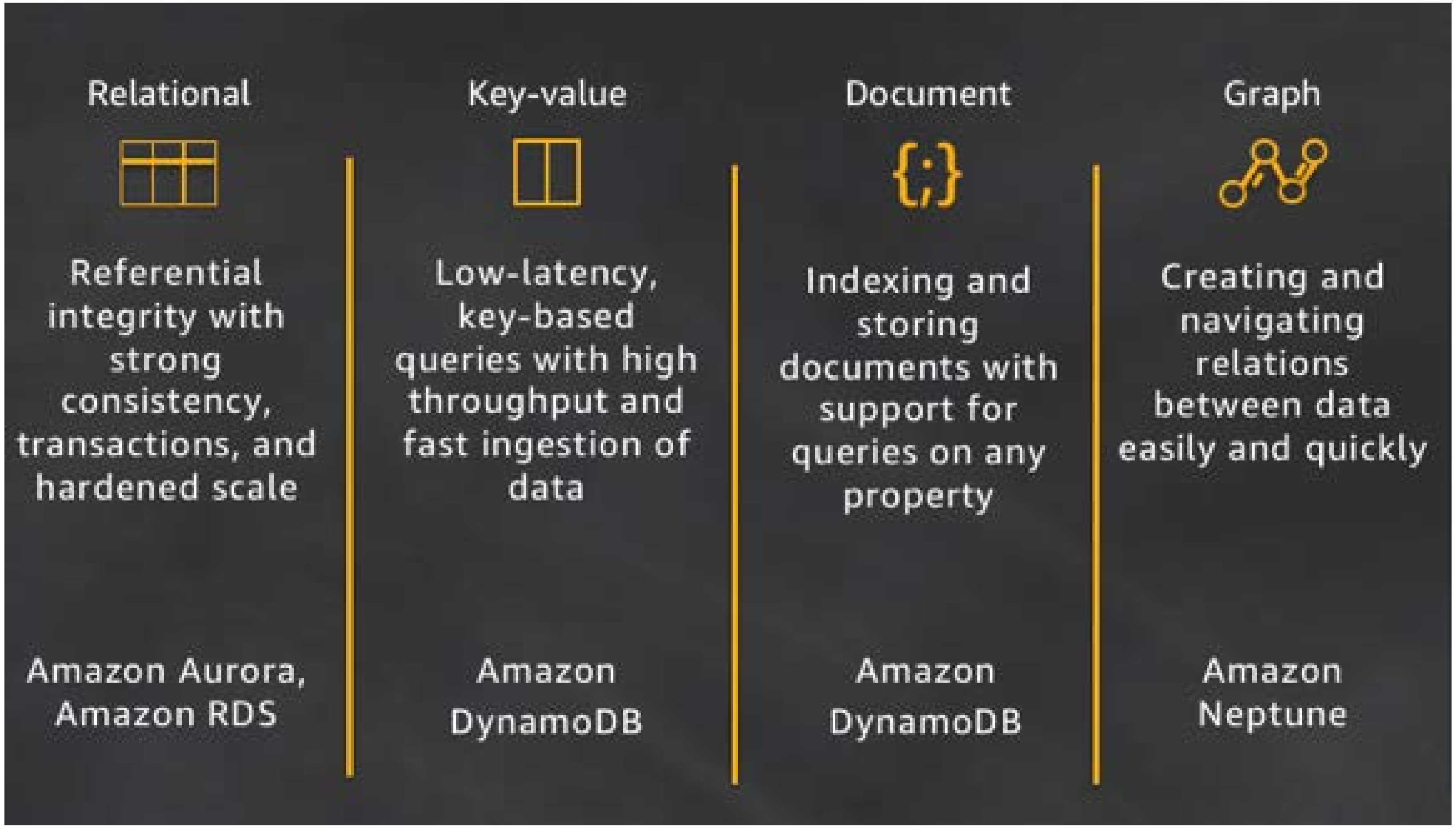
Different Data Stores are suitable with different requirements and usecases



Functionality	Considerations	Database Type
User Sessions	Rapid Access for reads and writes. No need to be durable.	Key-Value
Financial Data	Needs transactional updates. Tabular structure fits data.	RDBMS
POS Data	Depending on size and rate of ingest. Lots of writes, infrequent reads mostly for analytics.	RDBMS (if modest), Key Value or Document (if ingest very high) or Column if analytics is key.
Shopping Cart	High availability across multiple locations. Can merge inconsistent writes.	Document, (Key Value maybe)
Recommendations	Rapidly traverse links between friends, product purchases, and ratings.	Graph, (Column if simple)
Product Catalog	Lots of reads, infrequent writes. Products make natural aggregates.	Document
Reporting	SQL interfaces well with reporting tools	RDBMS, Column
Analytics	Large scale analytics on large cluster	Column
User activity logs, CSR logs, Social Media analysis	High volume of writes on multiple nodes	Key Value or Document

[Src: https://www.jamesserra.com/archive/2015/07/what-is-polyglot-persistence/]

Multiple AWS services



Relational	Key-value	Document	Graph
Referential integrity with strong consistency, transactions, and hardened scale	Low-latency, key-based queries with high throughput and fast ingestion of data	Indexing and storing documents with support for queries on any property	Creating and navigating relations between data easily and quickly
Amazon Aurora, Amazon RDS	Amazon DynamoDB	Amazon DynamoDB	Amazon Neptune

[src: https://www.allthingsdistributed.com/2018/06/purpose-built-databases-in-aws.html]

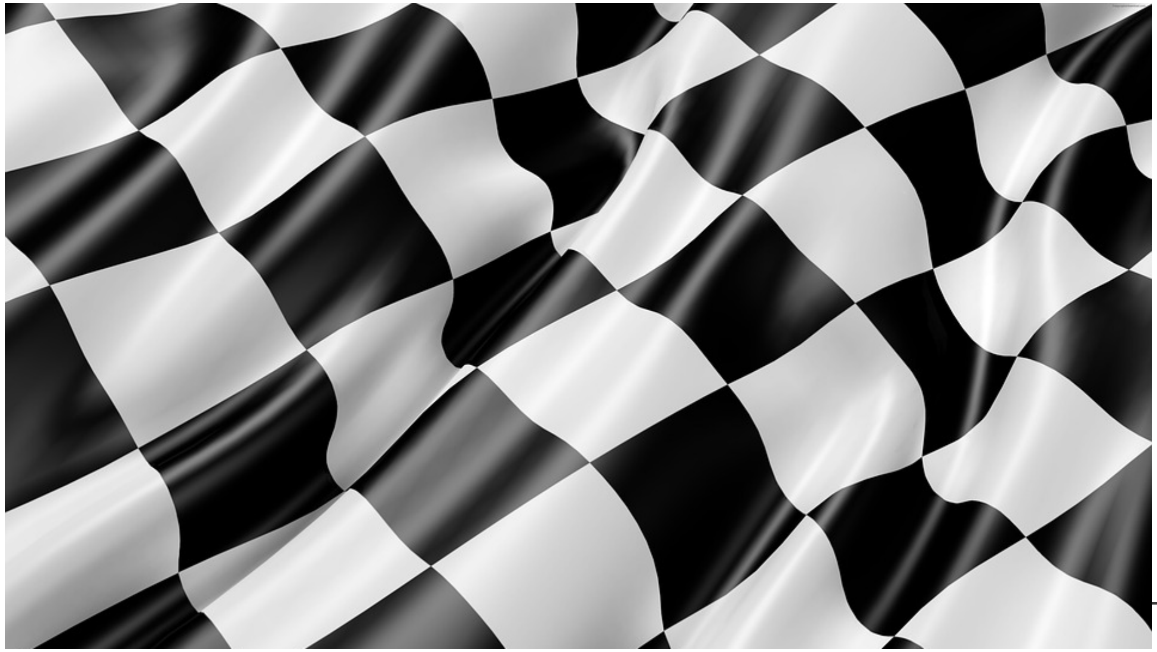
Summary

Databases are built for a purpose and matching the use case with the database will enable developers to write high-performance, scalable, and more functional applications faster.

Developers also are no longer using a single database for all use cases in an application—they are using many databases.

the REFERENCE

- P. Sadalage and M. Fowler: NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Addison-Wesley Professional, 2013
- D. Sullivan: NoSQL for Mere Mortals: Software Independent Approach, Addison-Wesley Professional, 2015



Thank you.

Let’s Summarize!