NoSQL

Abhijit Mane
08th Oct 2015

What is NoSQL ?

- NoSQL is a non-relational database management systems.
- Its different from traditional relational database management systems in some significant ways
- It is designed for distributed data stores where very large scale of data storing needs
  - for example Google or Facebook which collects terabits of data every day for their users
- These type of data storing may not require fixed schema, avoid join operations and typically scale horizontally

Why NoSQL ?

- In today's time data is becoming easier to access and capture through third parties such as Facebook, Google+ and others.
  - Example: Personal user information, social graphs, geo location data, user-generated content and machine logging data
- To avail the above service properly, it is required to process huge amount of data.
- Its handle these huge data properly.

Real Life Examples

- Social-network graph
  - Each record: UserID1, UserID2
  - Separate records: UserID, first_name,last_name, age, gender,...
  - Task: Find all friends of friends of friends of ... friends of a given user.
- Wikipedia pages
  - Large collection of documents
  - Combination of structured and unstructured data
  - Task: Retrieve all pages regarding athletics of Summer Olympic before 1950.

RDBMS vs NoSQL

RDBMS

- Structured and organized data
- Structured query language (SQL)
- Data and its relationships are stored in separate tables.
- Data Manipulation Language, Data Definition Language
- Tight Consistency
- BASE Transaction

NoSQL

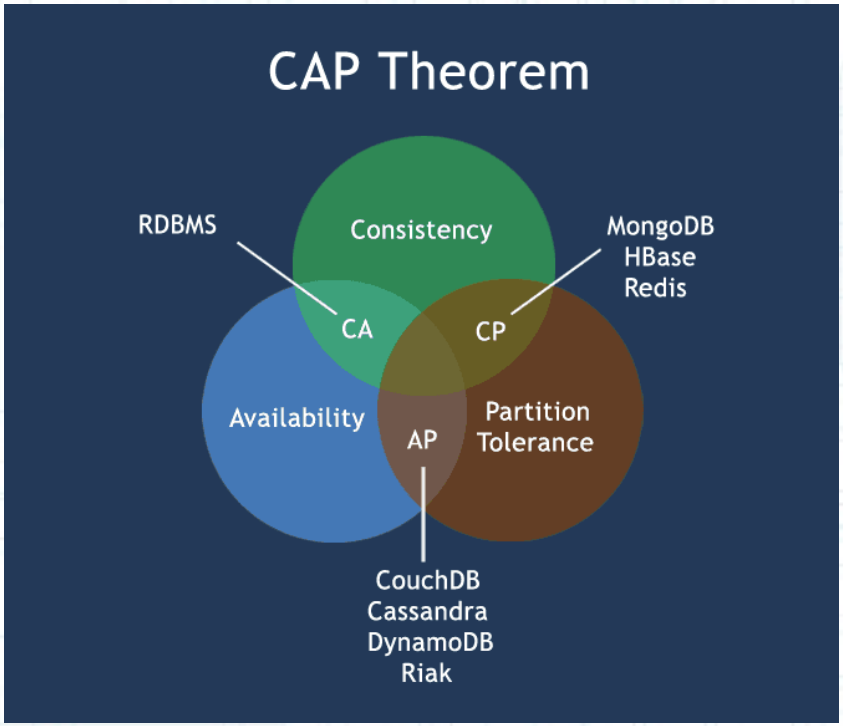
- Stands for Not Only SQL
- No declarative query language
- No predefined schema
- Key-Value pair storage, Column Store, Document Store, Graph databases
- Eventual consistency rather ACID property
- Unstructured and unpredictable data
- Prioritizes high performance, high availability and scalability

CAP Theorem (Brewer's Theorem)

- CAP theorem states that there are three basic requirements which exist in a special relation when designing applications for a distributed architecture
  - **Consistency -** This means that the data in the database remains consistent after the execution of an operation. For example after an update operation all clients see the same data.
  - **Availability -** This means that the system is always on (service guarantee availability), no downtime.
  - **Partition Tolerance -** This means that the system continues to function even the communication among the servers is unreliable, i.e. the servers may be partitioned into multiple groups that cannot communicate with one another.

CAP Theorem (Cont..)

- **All the current NoSQL database follow the different combinations of the C, A, P from the CAP theorem**
- **CA -** Single site cluster, therefore all nodes are always in contact. When a partition occurs, the system blocks.
- **CP -** Some data may not be accessible, but the rest is still consistent/accurate.
- **AP -** System is still available under partitioning, but some of the data returned may be inaccurate.



NoSQL pros/cons

Advantages

- High scalability
- Distributed Computing
- Lower cost
- Schema flexibility, semi-structure data
- No complicated Relationships

Disadvantages

- No standardization
- Limited query capabilities (so far)
- Eventual consistent is not intuitive to program for

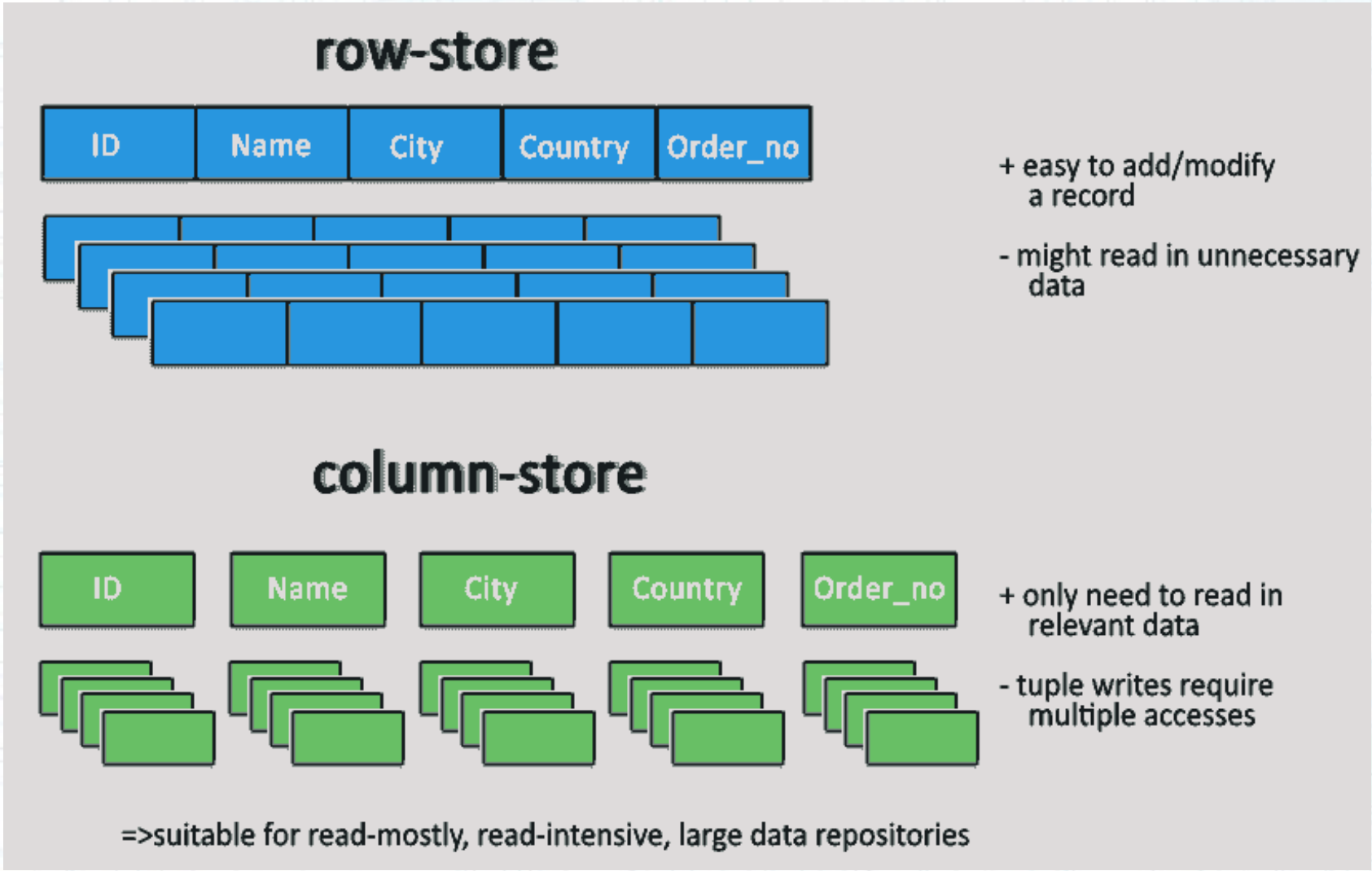
NoSQL Categories

- There are four general types (most common categories) of NoSQL databases
  - **Key-value stores**
  - **Column-oriented**
  - **Graph**
  - **Document oriented**
- **There is not a single solutions which is better than all the others, however there are some databases that are better to solve specific problems**

Key-value stores

- Key-value stores are most basic types of NoSQL databases.
- Designed to handle huge amounts of data.
- Based on Amazon’s Dynamo paper.
- Key value stores allow developer to store schema-less data.
- In the key-value storage, database stores data as hash table where each key is unique and the value can be string, JSON, BLOB (basic large object) etc.
- A key may be strings, hashes, lists, sets, sorted sets and values are stored against these keys
- Key-Value stores can be used as collections, dictionaries, associative arrays etc..

Example of Key-value store DataBase : Redis, Dynamo, Riak. etc.

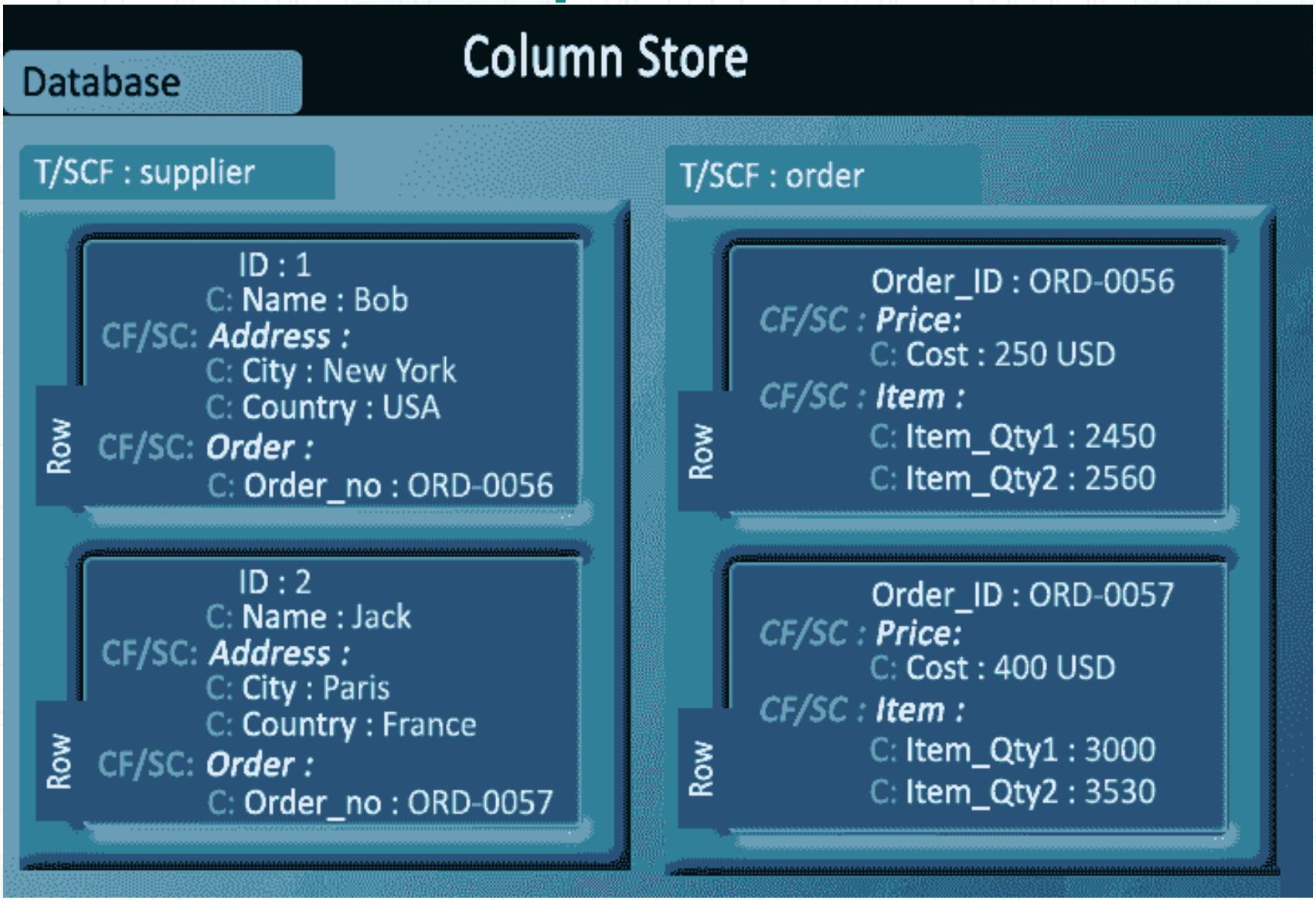


Example of Key-value store DataBase : Redis, Dynamo, Riak. etc.

Column-oriented databases

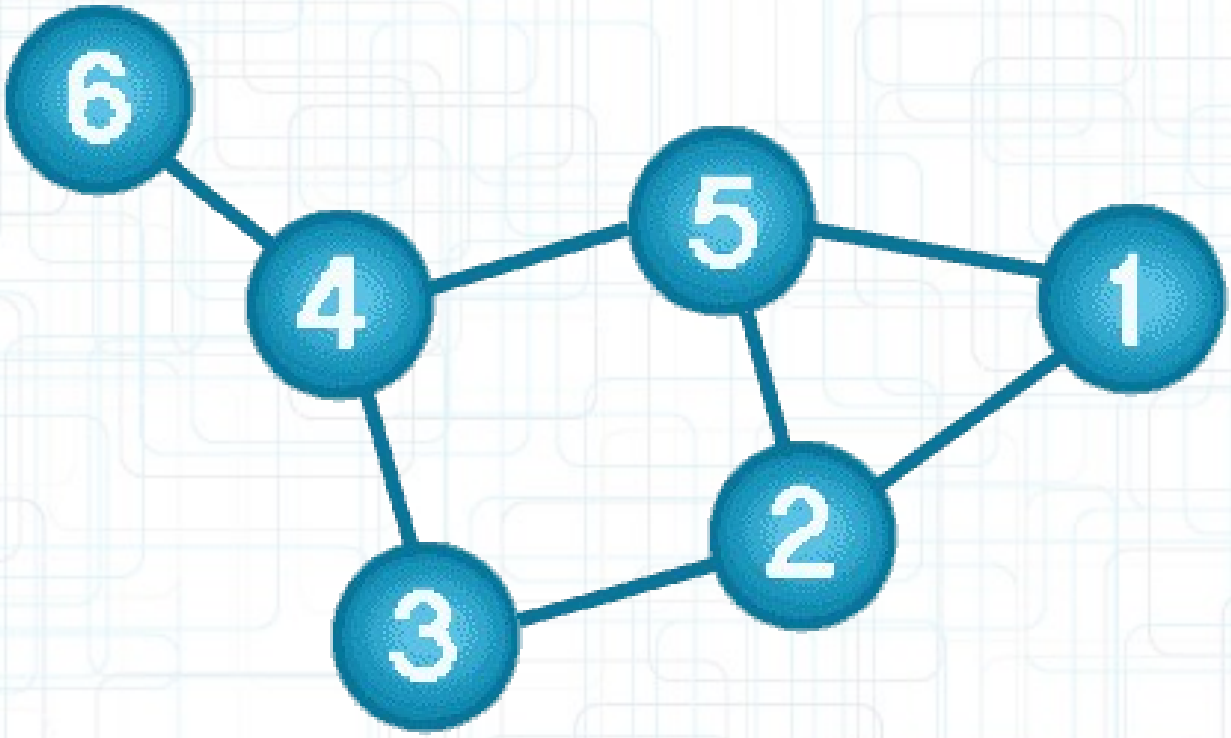
- Column-oriented databases primarily work on columns and every column is treated individually.
- Values of a single column are stored contiguously.
- Column stores data in column specific files.
- In Column stores, query processors work on columns too.
- All data within each column datafile have the same type which makes it ideal for compression.
- Column stores can improve the performance of queries as it can access specific column data.
- High performance on aggregation queries (e.g. COUNT, SUM, AVG, MIN, MAX).
- Works on data warehouses and business intelligence, customer relationship management (CRM), Library card catalogs etc.

Example of Column-oriented databases : BigTable, Cassandra, SimpleDB etc.



Graph databases

- A graph data structure consists of a finite (and possibly mutable) set of ordered pairs, called edges or arcs, of certain entities called nodes or vertices.



What is a Graph Databases?

- A graph database stores data in a graph.
- It is capable of elegantly representing any kind of data in a highly accessible way.
- A graph database is a collection of nodes and edges
- Each node represents an entity (such as a student or business) and each edge represents a connection or relationship between two nodes.
- Every node and edge is defined by a unique identifier.
- Each node knows its adjacent nodes.
- As the number of nodes increases, the cost of a local step (or hop) remains the same.
- Index for lookups.

Relational Model Vs Graph Model

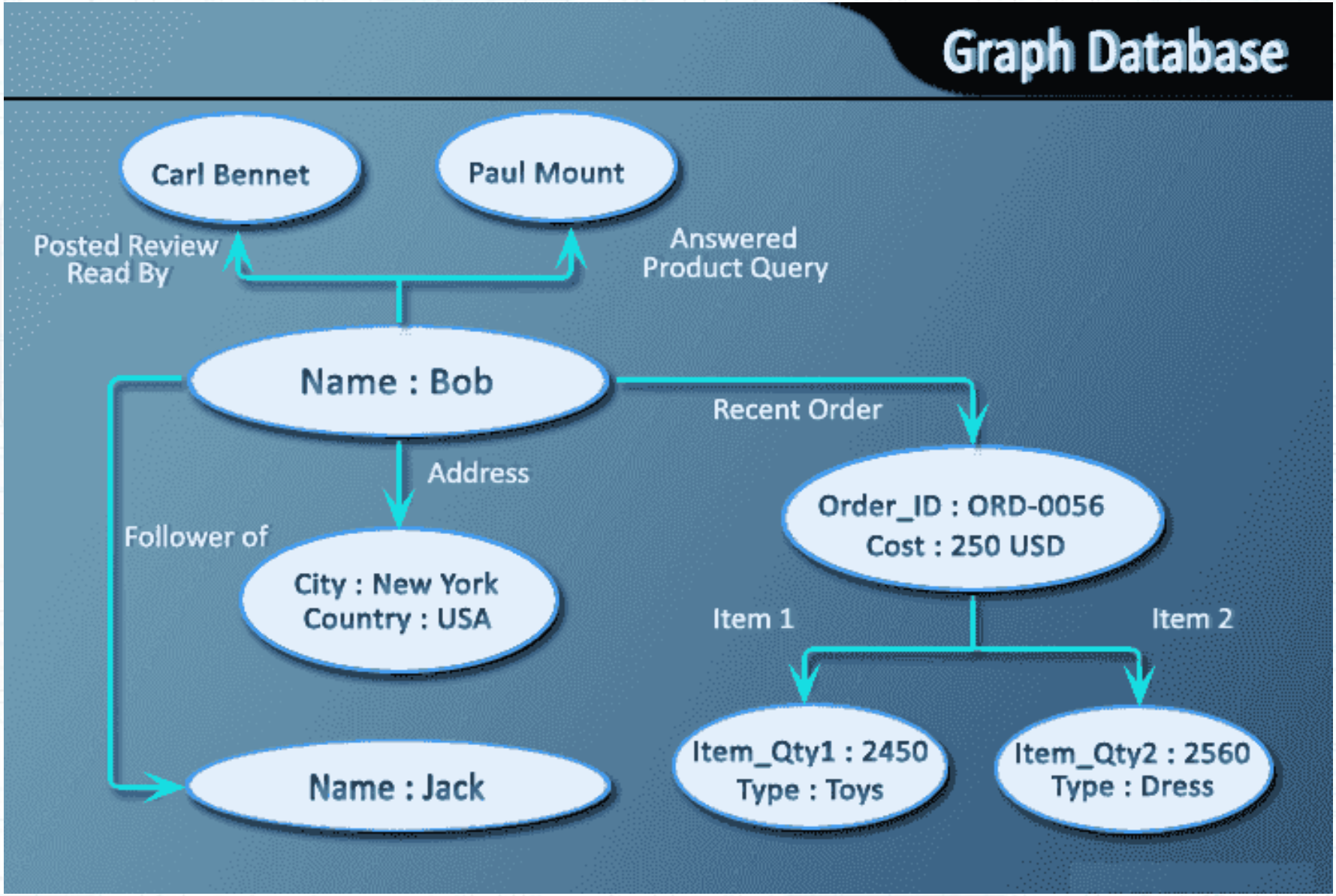
Relational mode

- Tables
- Rows
- Columns
- Joins

Graph model

- Vertices and Edges set
- Vertices
- Key/value pairs
- Edges

Example of Graph databases : OrientDB, Neo4J, Titan.etc.



Document Oriented databases

- A collection of documents
- Data in this model is stored inside documents.
- A document is a key value collection where the key allows access to its value.
- Documents are not typically forced to have a schema and therefore are flexible and easy to change.
- Documents are stored into collections in order to group different kinds of data.
- Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.

Relational Model Vs Document Model

Relational mode

- Tables
- Rows
- Columns
- Joins

Document model

- Collections
- Documents
- Key/value pairs
- not available

Example of Document Oriented databases : MongoDB, CouchDB etc.