# Introduction to Graphical databases
# LOAD CSV

## COMP3008

Lauren Ansell

# Introduction

Today's topics:

- Management of large datasets
- NoSQL
- Graphical databases
- The LOAD CSV function

Session learning outcomes - by the end of today's lecture you will be able to:

- Explain what is meant by NoSQL.
- Explain what a graphical database is.
- Importing large datasets into Neo4j

# Big Data - Recap

Big data is an ill-defined term.

There is a lack of precision in the definition of big data, which leads to ambiguity.

NoSQL is an accidental neologism. It is a term with no prescriptive definition. All you can make is an observation of common characteristics.

# Big Data Management

Any big data management strategy must include technology to support stream processing that scans, filters and selects meaningful information for capture, storage and subsequent access.

Managing big data not only subsumes many of the conventional approaches to data modelling and architecture, it entails a new cadre of technologies and processes to enable broader data accessibility and usability.

# Data Processing

Big data processing covers collecting, storing, and managing massive amounts of data that arrives from multiple sources.

Big data processing stages include data ingestion into a data lake or a stream processing engine, data cleansing and transformation, and data loading into an analytics storage optimised for querying and reporting.

**Key approaches:**

- Batch processing
- Stream processing

# Big Data Storage

Big Data Storage encompasses the Storage of diverse data types, which can originate from various sources such as social media, sensors, transactional systems, IoT devices and more.

Big Data Storage is versatile in handling data variety.

# Types Of Big Data Storage

Distributed File Systems

NoSQL Databases

Data Warehouses

Object Storage

# Pros and Cons of Using Big Data Storage

The advantages and disadvantages of Big Data Storage typically hinge on the quantity of data being handled.

| Pros | Cons |
| --- | --- |
| Data-driven Decisions | High Costs |
| Informed Choices | Resource Demands |
| Scalability | Data Security Risks |
| Efficiency | Complexity |

# Big Data Privacy and Security

Data privacy, often termed as information privacy, revolves around the proper handling, processing, storage, and usage of data.

While this data offers businesses unparalleled insights to tailor their services or products more aptly, it also comes with immense responsibility.

Several reasons underline the importance of data privacy:

- upholding trust
- legal implications
- avoiding financial repercussions
- ethical responsibility

# Emerging Data Models

**Bigtable:** Google Bigtable

**Key-value store, or key-value database**: Redis, MemcacheDB, Berkeley DB (BDB), HamsterDB...

**Document-oriented database, or document store:** MongoDB, CouchDB, OrientDB, RavenDB, Lotus Notes....

**Graph database:** Neo4j, HyperGraphDB...

# NoSQL?

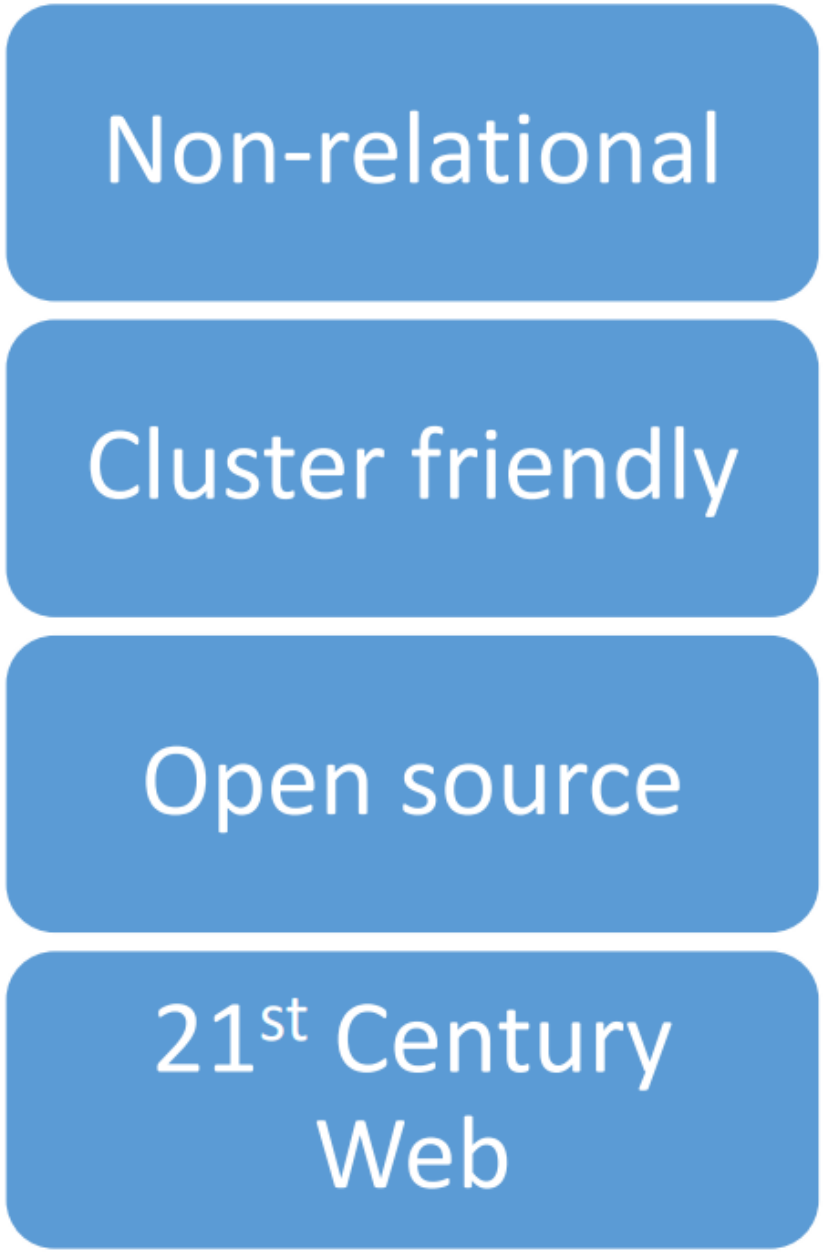
NoSQL was a Twitter hashtag (`#nosql`) chosen for a meetup organised by Johan Oskarsson in San Francisco in 2009 to discuss new databases. There were presentations delivered by Voldemort, Cassandra, Dynomite, HBase, Hypertable, CouchDB, and MongoDB.

“NoSQL is an accidental term with no precise definition".
[Sadalage & Fowler: NoSQL Distilled, 2012]

# NoSQL


NoSQL means Not Only SQL, implying that when designing a software solution or product, there is more than one storage mechanism that could be used.

# NoSQL Features



# Why Graphical Databases?

## The Challenge of Connected Data

- Limitations of Relational Databases
- Challenges in Social Networks
- Real-world Applications

## Where Traditional Approaches Struggle

- Complex Multi-Join Queries
- Schema Rigidity Issues
- Limitations in Graph-Like Queries
- Impact on Real-Time Analytics

# What is a Graphial Database?

A type of database designed to store information by focusing on how things are connected.

Instead of organising data mainly in tables, a graphical database is built to make it very easy to:

- represents relationships between things
- follows those relationships quickly
- answer questions that depend on lots of connections

# How Graph Queries Work

“Graph traversal” as following relationships

Pattern-based querying and matching

Contrast with relational join-heavy approaches

Basic intuition for what makes graph queries fast or expressive

# When to use Graphical Databases

## When to Use a Graph Database

- When we have complex, interlinked datasets.
- When deep or frequent relationship traversals exist.
- When we have evolving structures in the data.

## When Not to Use Graphs

- When we have tabular analytics.
- Where there are heavy aggregations.
- Where simple key-value access patterns exist.

# Challenges

Scaling distributed graph systems

Complexity of graph queries

Tooling and learning curves

Data ingestion and integration issues

# Real-World Use Cases

Social Networks

Recommendation Systems

Fraud and Cybersecurity

Knowledge Graphs

# LOAD CSV

You can use LOAD CSV to import data from a CSV file into Neo4j.

| title | price | salePercentage | recentReviews | allReviews |
|---|---|---|---|---|
| Ori and the Will of the Wisps | $9.89 | -67% | Overwhelmingly Positive | Overwhelmingly Positive |
| Flashing Lights | $8.49 | -66% | Very Positive | Very Positive |
| Thronefall | $5.24 | -25% | Overwhelmingly Positive | Overwhelmingly Positive |
| DRAGON QUEST | $23.99 | -40% | Very Positive | Very Positive |

```
LOAD CSV WITH HEADERS FROM
"file:///steam_store_data_2024.csv" AS line
AS csvLine
CALL (csvLine){
CREATE (:GAMES  title:csvline.title, price:
csvline.price, salespercent:
csvline.salesPercentage,
recentreview:csvline.recentReview,
allreviews:csvline.allReviews);
```

# Importing Large Amounts of Data

If the CSV file contains a significant number of rows (approaching hundreds of thousands or millions), `TRANSACTIONS` can be used to perform a commit after several rows.

This reduces the memory overhead of the transaction state.

You can control the number of rows committed at a time.

We need to include `IN TRANSACTIONS OF 250 ROWS` at the end of the command.

# Indexes

A database index is a redundant copy of some of the data in the database for the purpose of making searches more efficient.

Indexes have a cost: additional storage space.

Thus, deciding what to index is an important and often non -trivial task.

# Creating an index

```
CREATE INDEX [index_name]
FOR (n:LabelName) ON (n.propertyName)
```

| Syntax Element | Description |
|---|---|
| <index_name> | name of the node that we are going to create |
| <LabelName> | name of a label |
| <propertyName> | name of the property |

```
CREATE INDEX FOR (t:Tweet) ON (t.tweetID)
```

# Summary

**NoSQL**

- Stands for Not only SQL
- The idea that multiple storage mechanisms can be used.

**Graphical databases**

- Focus on the relationships between the data.
- More flexible than traditional databases.

**Neo4j functions**

- Place the data we want to import in the import folder.
- LOAD CSV function to create the database.
- For large datasets, we may have to bring the data in batches.