4.11 Big Data

Big Data

Big Data is a term used to describe **datasets that are too large, complex or rapidly generated to be handled effectively using traditional data-processing methods.**

Big Data can be described in terms of the **three Vs.**

Volume

The volume of data is too large to store or process on a single server, so it is distributed across multiple servers.

Velocity

Data is generated and needs to be processed extremely rapidly.

Variety

The data comes in many forms and lacks a specific structure.

Data can be structured, semi-structured or unstructured, including text, images, video and other formats.

This variety can make the data more difficult to analyse

Examples of Big Data

Social media data

- Platforms like Facebook, TikTok and Instagram generate massive amounts of data every second.
    - Posts, likes, comments
    - Images and videos
    - User behaviour patterns

E-Commerce

- Online stores like Amazon and Shopify analyse:
    - Millions of customer purchases per day
    - Browsing behaviour
    - Real-time stock levels
    - Recommendations

Healthcare and medical records

- Hospitals collect huge datasets on patients.
    - MRI/CT scan images
    - Patient electronic health records
    - Data from wearable devices tracking heart rate, sleep, etc

Financial Trading Systems

- Stock exchanges and trading algorithms process:
    - Millions of transactions per second
    - Market trends and price movements

Functional Programming

Functional programming can be used to process Big Data across multiple servers. Different processors can work simultaneously on separate parts of the dataset, with the results combined afterwards.

A programming paradigm called **functional programming** is useful for overcoming this issue. It allows many processors to work simultaneously on parts of the dataset without affecting the other parts. The results can then be combined.

Features of functional programming that make it **useful for distributing over multiple servers** are:

- **Immutable data structures**
  - data structures cannot be modified after they have been created.
- **Statelessness**
  - Functions do not have side effects. The output of a function depends solely on the input.
- **Higher-order functions**
  - Functions that take functions as arguments or return a function as a result. These are easily parallelised for processing results.
- **Code does not determine the order of execution**
  - The order in which functions execute is not fixed by the program code, allowing functions to be executed in parallel.

Graph Schema

Traditional relational databases may not be suitable for processing some Big Data because of its volume and variety.

The **fact-based model** allows the capture of a single piece of information called a **fact**.

Examples of a fact captured could be:

- Coordinates = 42.423565, -72234645
- Manufacturer = Sony
- Number of items sold = 87

Graph schema can be used to explore the relationships between connected entities in a dataset.

- **Entities** are represented by nodes, which have **properties** containing facts about them.
- **Edges** represent relationships between nodes.