W2 - Data Storage

1. ETL Pipelines
2. Structure of a Database
3. Primary/Foreign Keys
4. Model Databases using Entity Relationship Diagrams
5. Normalise a Database
6. OLTP & OLAP
7. Design a Data Warehouse
8. Database, Data Warehouse & Data Lake

1. **ETL Pipelines**

Extract > Transform > Load

Extract > Load > Transform > Analytica

2. Database

Organised collection of structured information, or data, typically stored electronically in a computer system.

i.e MySQL, PostgreSQL use tables with rows and columns to organize data with relationships.

3. **Primary Key** - uniquely identifies each record in the table. Most tables should have a primary key, cannot be blank or null. The values must never change.

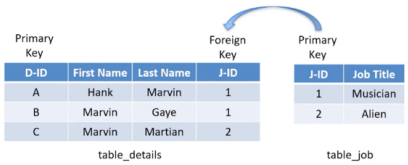
- **Compound Key** - combines more than one field to make a unique value.
- **Composite Key** - combines more than one field to make a unique value.

i.e Order No & Line No. Line No on its own is not unique.

Order No	Line No	Quantity Ordered	Product Code
10666	0001	32	0022
10666	0002	69	1001
10666	0003	53	0209
10667	0001	2	0486
10668	0001	40	1001
10668	0002	2	0022

Foreign Key - ensure that the row information in Table A corresponds to the correct row information in Table B.

- No uniqueness constraint for foreign keys
- A table can have any number of Foreign Keys
- A row cannot be deleted from a reference table if it is in use via a foreign key.



4. Types of Databases

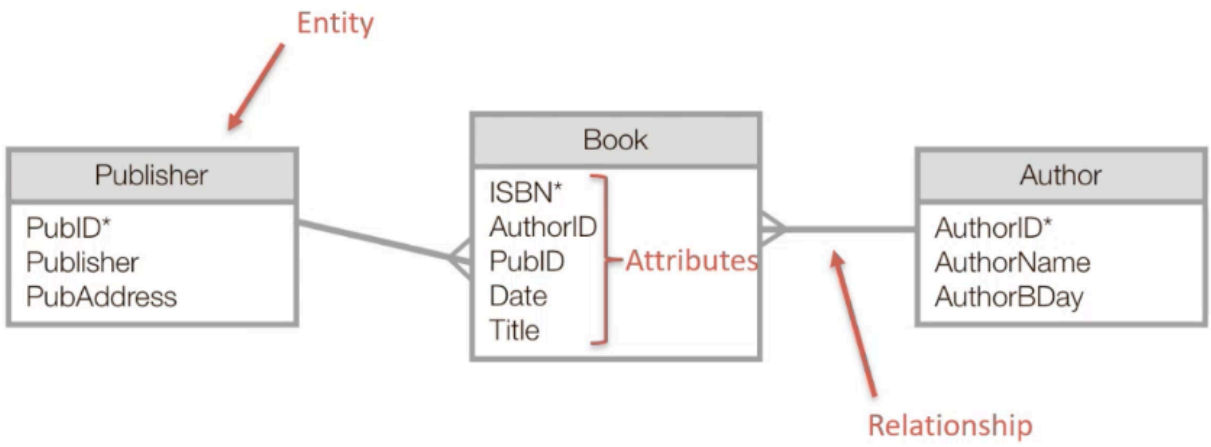
- **Flat file Database** - stores everything in 1 table. Good for small numbers of records related to a single topic.
- **Relational Database** - gives you the ability to separate masses of data into numerous tables
- **Non-Relational or Big Data** - used for Data Analytics & Business Intelligence

3. Data Model

Details of how a database will be organized. Shows how elements relate to one another.

- **Conceptual Data Models**
- **Logical Data Models**
- **Physical Data Models**

Entity Relationship Diagram



- **One-To-One** maximum one, minimum one

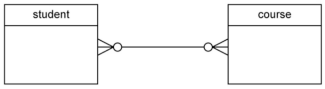
Student -> Seat

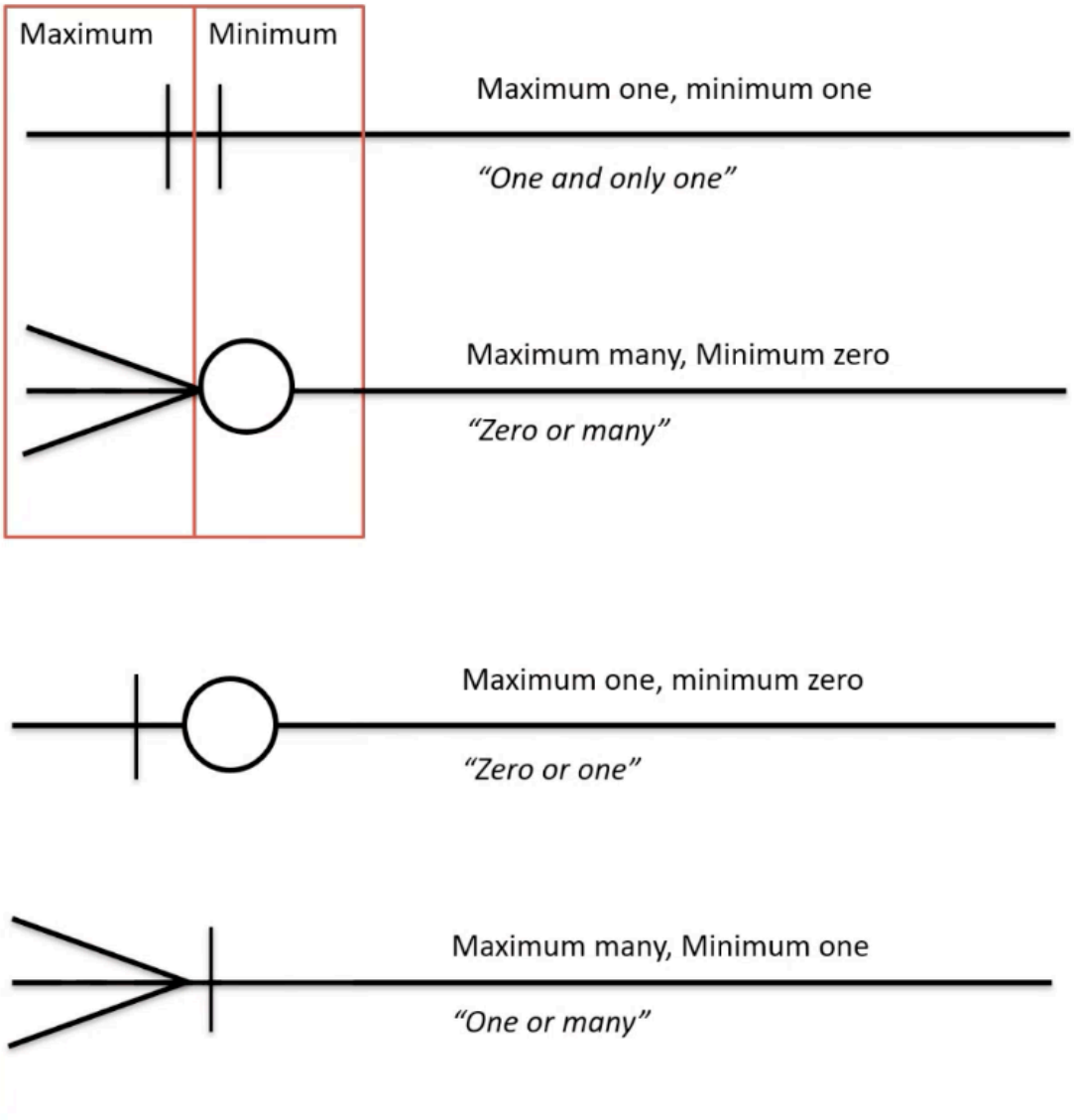
- **Zero-To-Many** maximum many, minimum zero

Lecturer -> Course (1 lecturer to 0 or many courses 1-To-Many) relationship

- **Zero-Or-One** maximum one, minimum zero
- **One-Or-Many** maximum many, minimum one

- Many-To-Many





5. Normalise a Database

1NF - First Normal Form

A database is in **First Normal Form** when the following conditions are satisfied:

- ◯ Make everything Atomic
- ◯ There should be no repeating groups

This table is NOT in 1NF:

table_product

Product ID	Colour	Price
1	red, green	10.99
2	yellow	12.00
3	green	14.99
4	yellow, blue	5.99
5	red	20.00

- Colour column contains multiple

How to make this table 1st Normal Form?

- Split the data into 2 tables

2NF - Second Normal Form

A database is in Second Normal Form when the following conditions are satisfied:

- ◯ It is in **1st Normal Form**
- ◯ All non-key attributes are fully functional dependent on the Primary Key

Here a **Composite Key** is used. 'Product ID' and 'Store' combine to make the Primary Key. In this case 'Location' only depends on Store which is only part of the primary key.

table_purchase_detail

Product ID	Store	Location
1	1	London
1	3	Tokyo
2	1	London
3	2	New York
4	3	Tokyo

This does not satisfy **2NF** as it's only dependent on the store and not the product ID as well.

How to make this 2NF?

Separate out the tables, remove the partial function of dependency.

table_purchase

Product ID	Store
1	1
1	3
2	1
3	2
4	3

table_store

Store	Location
1	London
2	New York
3	Tokyo

3NF - Third Normal Form

A database is in Third Normal Form when the following conditions are satisfied:

- ◯ It is in **2NF**
- ◯ There is no transitive functional dependency (when a non-key column is functionally dependent on another non-key column, which is functionally dependent on the primary key)

In this example, Book ID determines Genre ID, which in turn determines Genre Type - meaning they're functionally dependent on each other

table_book_detail

Book ID	Genre ID	Genre Type	Price
1	1	Fiction	09.99
2	2	Travel	14.99
3	1	Fiction	24.99

How should the data look like:

table_book

Book ID	Genre ID	Price
1	1	09.99
2	2	14.99
3	1	24.99

table_genre

Genre ID	Genre Type
1	Fiction
2	Travel

6. Relational Data Processing - OLTP & OLAP - There are 2 main ways of managing business data

OLTP - Online Transaction Processing & OLAP

Management of transactional data - database transactions including

Any Change to database

- Payments received
- Orders taken
- Services delivered
- Product moving through inventory

Requires high degree of normalization

OLAP - Online Analytic Processing

Designed to extract business intelligence from OLTP

Adds layer of abstraction and aggregation

- Semantic Data Model describing meaning of data elements
- Data integrated from multiple sources and aggregated together across multiple dimensions.

7. Data Warehouse

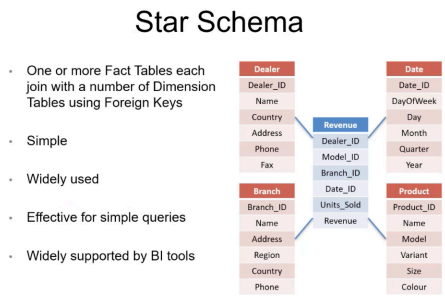
Designed for data analysis

- Aggregates data from a variety of data stores

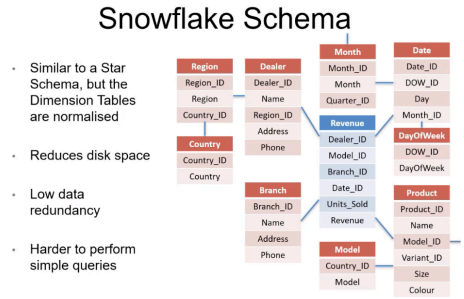
- Supports analysis on the composite data
- Only stores data that answer concrete business questions

Design - Dimensional Modelling

STAR Schema



SNOWFLAKE Schema - dimension tables are normalised



- Create a Star Schema to support queries to show Blockbuster rentals. The queries will pull in the following fields:
- Time
- Date
- Customer Name
- Customer Age
- Customer Gender
- Employee (who served them)
- Product Name
- Product Type (e.g. DVD, BluRay, PS4 etc)
- Length of Loan
- Return Date
- Amount Paid

8. Data Lakes

- Stores structured and unstructured data in original form.
- Processed when analysis is needed, not when stored.
- Low cost storage vs Warehouse
- Highly scalable
- Highly agile

Differences between

1. **Database** - contains structured data regarding tables with rows & columns
2. **Data Warehouse** (Snowflake, Google Big Query) - centralized repository used for storing, collected from multiple sources (sales, marketing, finance) - cleaned, structured, and ready for analysis and report creation.
3. Data Lake - stores structured and unstructured data in original form at large scale.