What are Databases?

Serial Files

- Files are a permanent storage of data.
- Characteristics:
  - Data is stored in the order in which it is entered.
  - No order to the data is maintained.
  - Useful for storing transactional data and initialisation files.

Sequential Files

- Characteristics:
  - Store data in order of a key field.
  - The order is maintained when new files are added.
  - Useful for storing master files.
- Indexed sequential files:
  - Maintain an index to allow groups of records to be accessed quickly.

Databases

- Databases are structured, persistent collections of data.
- When handling large related data sets, a database:
  - Makes processing more efficient.
  - Reduces the storage requirements.
  - Avoids redundancy.
  - Allows different users to only see relevant data.

Database Management Systems

- The data is the structured data, and is managed by an additional layer of software called the database management system (DBMS).
- It provides:
  - A manipulation language to access and change the data.
  - Integrity to ensure efficiency and structure is not compromised.
  - Additional security.
  - An interface for other programs to access and use data.
  - Program / data independence.

Database Components and Types

- Tables (entities or files) that store data in rows (records or tuples) and columns (fields or attributes).
- Queries to manipulate data, search, sort files, add, amend and delete data.
- With provision in other software for creating:
    - Forms or user interface.
    - Reports.

Flat File Database

- A database that is held in a single file.
- It is possible to define different flat files for each identity.
- But each flat file is not linked to the other.
- There is just one table and all the data is stored within it.
- Very inefficient way of storing data, because it is not easy to query.

Relational Database

- Has more than one related table.
- Data is no longer duplicated making querying much quicker.
- The process to get from a flat file to a fully relational design is known as normalisation.
- Each record must be unique.
- A special field known as a primary key is the unique identifier for the record.
- A foreign key is the field on the many side of a relationship.
- **A composite Primary Key -** a primary key which consists of more than one attribute.

Types of Relationship

- One to One (fields could be in one table)
- One to Many
- Many to Many (theoretically) but should always be broken down into two one to many relationships.

Secondary Keys and Indexing

- Primary keys are also known as an index.
- The DBMS uses this field to quickly find records.
- Data is stored physically in the order in which it is entered.
- Rather than moving the records around, a series of additional indexes are stored.
- Any field that is indexed so it can be easily stored, is known to have a secondary key.
- For faster searches, primary keys and secondary keys are usually indexed.

Benefits of Indexing

- By indexing a data structure is created and it improves the speed of data retrieval operations on a database including fast searches based on different attributes.
- The primary key is normally indexed for quick access
- The secondary key is an alternative index

Drawbacks of Indexing

- The index takes up extra space in the database
- When a data table is changed, the indexes have to be rebuilt.

Methods of Capturing Data

- It is important that data entered into a database is accurate upon entry.
- A number of methods can be employed to ensure accuracy and speed of data entry:
    - **Paper data capture form** - having a dedicated sheet with all the fields you want to collect clearly labelled.
    - **Optical Character Recognition** (OCR) - to automatically read text from the form. This is not just for forms. The post office uses OCR software to read postcodes and route mail. Cameras on roads use automatic number plate recognition software for congestion charging.
    - **Optical Mark Recognition -** is often used for multi-choice tests and lottery tickets.

Selecting Data

- A common query language called SQL or structured query language allows data to be extracted from a database.
- To a user, an interface could be presented as a series of input textboxes, options or drop down boxes enabling the data to be searched and filtered.

Managing Data

- The database management system provides a layer of obstruction for the user and programmer, hiding the underlying structure of the data storage.
- It ensures that data is kept integral, by preventing duplicate primary keys, enforcing validation rules, providing secure access, encryption ensuring program data independence.
- It manages many connections to the data source and prevents two users updating the same record at the same time.
- SQL is both a data definition language, defining the structure, and a data manipulation language.
- The data set can be changed.
- In SQL, adding tables and records, amending records and deleting records & tables.
- SQL is good enough that you do not need an interface to make the changes.

Exchanging Data

- It is common to want to be able to exchange databases and other applications.
- There are many ways to achieve this, but some include XML, which is largely replaced by JSON, a human-readable, open standard format of structured data.
- CSV (comma separated value) text files are files where each record is stored on a separate line and the fields are separated with a comma.
- These standard formats make it easy to program and import routines and data can often be output in these formats, making data exchange between systems easy.
- Another approach is not to use manual data exchange at all, but to interface two databases together so they can read and write to each other's tables, possibly through import / export routines or by a live connection.
- This is **EDI - Electronic Data Interchange.**
- **EDI -** A protocol between two systems to facilitate the exchange of data**.**

Structured Query Language

Introduction

- A common query language for all databases is the Structured Query Language (SQL).
- Developed in the 1970's, this language allows for fast and efficient retrieval, deletion and manipulation of data held in relational databases using a simple set of commands.
- It is primarily a declarative language, meaning it expresses what needs to be achieved as opposed to a procedural language, which would express the logic of how something needs to be achieved.
- SQL allows us to:
    - Query data
    - Manipulate data
    - Define data
    - Control data access

SQL Commands

- SELECT
- FROM
- WHERE
- LIKE
- AND
- OR
- DELETE
- INSERT
- DROP
- JOIN

More Stuff

- When listing field values, strings must be enclosed in quotation marks.
- It is also possible to omit the list of fields if you wish, but you will need to make sure you supply all the appropriate values for a complete record.
- Use the INSERT command to add a new record to a table.
- When listing field values, strings must be enclosed in quotation marks.
- Use a DELETE command to remove records from a table.
- When listing field values, strings must be enclosed in quotation marks.
- JOIN can be used to combine data from two or more tables by specifying a common field between them.
- The DROP TABLE command can be used to drop an existing table in a database.
- Be very careful when using this command, as it will result in the complete loss of all information stored in the table.

Normalisation

First Normal Form (1NF)

1. All field names must be unique.
2. Values in fields should be from the same domain.
3. Values in fields should be atomic.
4. No two records can be identical.
5. Each table needs a primary key.

Second Normal Form (2NF)

1. The data is already in 1NF.
2. Any partial dependencies have been removed.

Third Normal Form (3NF)

1. The data is already in 2NF.
2. Any transitive dependencies have been removed.

Methods of Capturing Data

- When working with databases, we need to think about data and how we handle it in four distinct ways:
  - **Capturing:** How do we get the data into the database in the first place?
  - **Selecting:** How do we then query the and retrieve it?
  - **Managing:** How do we manage, manipulate, add, edit and delete the data?
  - **Exchanging:** How do we exchange the data with other people / systems?

Paper-based Forms

- A vast amount of data is still captured using paper-based data capture forms.
- Data input via this method is largely manual. It involves a human reading the form and typing the information into a computer-based system.
- For the process to be as fast as possible and to avoid errors, a number of tactics can be implementing when designing the data capture form.
- These all help to reduce the likelihood that the person inputting the data will make a mistake due to poor legibility of the text.

Optical Character Recognition

Referential Integrity

Multi-User Databases

- Databases can hold vast amounts of information and often need to support multiple simultaneous users.
- Large databases such as those used by the NHS or police can have millions of records and thousands of active users.
- Users can be given different access rights to a database:
    - Some will only be able to query the database and run reports.
    - Others will be able to add and modify records.
    - A select few may be allowed to delete records.
- All these different database queries result in multiple transactions taking place, often at the same time.
- It is vital that this process never causes a database to become inconsistent or corrupt.
- If transactions cause the database to become inconsistent, we can no longer guarantee its accuracy.

Data Integrity and Referential Integrity

- No matter what type of transaction is taking place, the Database Management System (DBMS) ensures that the data stored in the database remains consistent.

Data Integrity

- **“The maintenance and consistency of data in a data store. The data store must reflect the reality that it represents.”**
- The process of maintaining the consistency of the database is known as data integrity.
- Being able to guarantee the integrity of data held in a database is of vital importance.

Referential Integrity

- A key technique for ensuring data integrity in a relationship database is known as referential integrity.
- Referential integrity refers to the accuracy and consisting of data within a relationship.
- Referential Integrity helps reduce the risk of orphaned entries - entries that are not deleted when their referenced entries in other tables are removed and no longer exist - thus leading to inconsistent data.
- One way to maintain referential integrity would be to enforce a cascade delete restraint on the primary key relationship between the tables.
- Referential integrity enforces this process and helps to ensure the integrity of our data.
- In a similar way, referential integrity can be implemented to prevent us from adding a record to a table if there isn’t a matching entry for it to link to in a different table.
- Although referential integrity restraints like cascade delete can help us maintain data integrity, they must be used with caution.

Transaction Processing

- **Transaction Processing:** Any information processing that is divided into individual, indivisible operations called transaction.
- Each transaction must succeed or fail as a complete unit - it can never be only partially complete.
- All relational databases have a certain base functionality, referred to using the acronym CRUD:
  - Create
  - Read
  - Update
  - Delete
- These core functions map to the following SQL statements:
  - Create - INSERT / CREATE
  - Read - SELECT
  - Update - UPDATE
  - Delete - DELETE

ACID Rules

- To ensure data integrity, transaction processing in all database management systems (DBMS) must conform to a set of rules.
- These rules are referred to using the acronym ACID:
  - Atomicity
  - Consistency
  - Isolation
  - Durability
- These four rules describe the properties required by all database transactions.

Atomicity

- **A change to a database is either completely performed or not at all.** A half-completed change must not be saved back to the database.
- The word atomicity comes from the word atom.
- It refers to the fact that we once thought the atom was the smallest particle and the singular building block for all other matter.
- There was no concept of having half an atom - likewise, there should be no possibility of having a partially performed database transaction.

Consistency

- **Any change in the database must retain the overall state of the database.**
- A good example of this is the transferring of funds from one bank account to another - for example, when you pay for something online.
- Money debited from one account must be balanced by the money being credited in another.
- If this was not the case then, digitally speaking, money could just vanish.

Isolation

- **A transaction must not be interrupted by another transaction. The transaction must occur in isolation so other users or processes cannot access the data concerned.**
- In practical terms, a DBMS enforces isolation by implementing a system of recording locking.
- The record(s) being affected by the transaction are locked, effectively placing them in a read-only state.
- Only when the transaction is fully completed will the lock be removed.

Durability

- **Once a change has been made to a database, it must not be lost due to a system failure.**
- In real terms, durability is achieved by making sure the DBMS writes the effects of transactions immediately back to permanent secondary storage rather than simply holding those changes in any form of temporarily volatile storage such as main memory (RAM).