Section 4: Database Fundamentals

1. Relational database: Store data in structured tables (rows and columns).
   - Hierachical database: Arranges data in a tree structure with parent-child relationships, making it ideal for use in scenarios where there are clear hierachies
   - Network databases: uses a graph like structure to represent many to many relationships
   - Object-oriented database: Stores data as objects similar to object oriented programming and is ideal for applications with complex data models such as financial systems
   - No SQL Database: offers flexible, schema-less structures for handling large volumes of unstructured or semi-structured data
     E.g. document-oriented, key value, wide column.

2. Relational Database management system is a software system that enables users to define, create, maintain and query a relational database. In relational database management system, the data gets stored in tables, the table would contain a primary key and foreign key.

3. Primary key is a column that uniquely identifies each row. (It contains different data for example, the customer_id).

Foreign key: A column or a set of columns in one table that references the primary key of another table.

4. Normalization is the process of structuring a relational database to reduce redundacy and dependency by organising fields and tables relationships according to the rules
The importance of normalization is that it reduces data duplication which saves storage.

5. A schema is the logical structure of a database. It describes how data is organized, what tables exist, what columns they have. It helps us to arrange our data and categorize.

6. Structured data has consistent format. It is data stored in a table with rows and columns. For example, data stored in an excel file.

Semi-structured data: Has some organizational properties (tags, keys) but does not follow a rigid schema
Unstructured data: Does not have consistent format For example, images, videos, social media posts.

7. Fact table: Stores measurable quantitative data about business processes, e.g revenue.

~~Dimener~~ Dimension table: Contains descriptive attributes that gives context to the facts (e.g date, product, customer, location).

8. Data model is a formal representation of the structure of data: such as tables, the columns and the relationship between them.
It is important in database design because it provides a blue print for designing and building the database. It ensures consistency and clarity in how data is stored.

9. ~~Data wareho~~ Database is where the data gets stored. (The system used is online transaction processing) Data warehouse is a central place where we can have multiple databases. The system used is online analytical processing.
Data Lake: A large storage repository that holds raw data in its native format.

10. A data mart is a subset of a data warehouse that focused on a particular business line, department or subject area. The difference between the data ware house and data mart is that the data marts are smaller and more focused than the full data warehouse. It is easier to design and managed due to its its limited scope. ~~And~~ It is less expensive.

Section B

11. Query language is a programming or domain specific language used to make queries in databases, either to retrieve or modify data.
SQL is commonly used because it is widely supported ~~acc~~ across nearly all relational database systems. It is powerful for selecting, joining, filtering and aggregating.

12. An index is a data structure that allows the database to quickly locate and access data in a table without scanning every row. It helps with sorting and filtering by using the where ~~cat~~ clauses. etc).

13. A transaction is a sequence of one or more database operations (like insert, delete) executed as a single logical unit of work.
The ACID properties are:
- ~~A~~ Atomicity: A transaction is "all or nothing."
- Consistency: A transaction brings the database from one valid state to another, preserving all defined values
- Isolation:

- Durability: Once a transaction is commited, its changes are ~~pa~~ permanent, even if there is a system crash.

14. Database ~~engi~~ engine is the part of the database

system responsible for how data is stored, retrieved and managed on disk. The different engines optimize for different workloads e.g transactional or analytical. ~~The ethic engines are~~ Some of the engines are designed for writing fast or fast reads.

15. View: A virtual table that is based on the result set of a SQL query.
Stored procedures: A precomplied set of SQL statements that can be executed as a singe call.
Trigger: A database object that automatically executes a piece of SQL code (trigger body) in response to certain events on a table (like insert, etc).

16. ETL: Extract data from source systems
Transform it by cleaning, aggregrating using an intermediate system
Load the transformed data into the target.

ELT: -Extract data from source
- Load raw data into the ~~#~~ data lake or warehouse
- Transform the data inside the target system using its comupute ~~#~~ power, for example using SQL.

17. Batch processing: Data is collected over a period and processed in large "botches"
Stream processing: Data is processed in real time as soon as it arrives.

Join in SQL is used to combine rows from two or more tables based on a related column.

Types of joins:

1. Inner join: Returns only rows that have matching values in both tables.
E.g. select student id, name From Students A inner join enrollments as B on A student id = B student_id

2. Left (outer) join: Returns all rows from the table table and matched rows from the right table, if there are match, right side columns are null.
E.g. Select student_id, name, course name From Students as A Left join enrollments as B on A student_id = B. student_id;

3. Full outer join: Returns all the rows whether there is a match or not.
E.g. select customer_id, name, amount From customers AS A full outer join orders AS B on A. customer_id = B customer_id.

Inner Join

select student id, student name
From Students
As a inner join grades as B on A student_id = B student_id

19. Referential integrity is a property that ensures that foreign keys in a table correctly reference primary keys in another table (or the same table), and that these references are valid.
It is important in relational database because it reduces data corruption and maintains logical consistency across tables.

~~20. Redundancy affects the s~~

20. Redundancy increases storage waste as more space is used because of duplicated data.
more duplication makes schema more complex to maintain and it is difficult to enforce data integrity

Section C: Data Management & Analytics ~~Concepts~~ Concepts

21 Cloud database can scale up or down more easily (storage, compute) often automatically
Maintenance: In cloud, many tasks such as backups, patching and replication) can be automated by the provider
Access & connectivity: Cloud database are accessible over the internet.

22. Data governance is a framework of processes, ~~plans~~ policies, roles and standards that ensure data is managed properly, securely and used effectively.
Data governance is important in data management because it ensures that the data is correct, has quality. It ensures that the

data usage follows the regulatory, regal and compliance requirements. It also helps in risk management.

23. Data integrity is the accuracy, consistency and reliability of data throughout its life cycle. Firstly, it is maintained by using database constraints such as null, not null. The audit logs or history is maintained in order to track changes. The data is cleaned and lastly, there should be back ups or recovery procedures to ensure data isn't lost or corrupted.

24. Data quality refers to how well the data conforms to the requirements for its intended use. If the data quality is poor, it can lead to wrong insights, inaccurate analyses which leads to poor decision making. Having high quality data builds trust in reports and analtyics outputs.

25. A data analysts writes and runs queries using SQL to extract relevant data from databases. The data analysts cleans and transforms raw data into a usable form. The data analyst perfuims data analysis in order to understand the trends and data distributions. Builds charts, dashboards to communicate insights. Reports findings with recommendations.

26. Design and manage database schemas
Monitor database performance and tune queries, indexes and configuration
Ensure data security
Implement and manage backup and recovery strategies
Plan for capacity such as storage
Manage high availability, replication, ~~Dit~~ failover and clustering

27. Define ~~de~~ objectives & requirements: What data do you need, for what purpose?
2. Identify data sources
3. Extract data from sources safely
4. Transform data by structuring, filtering and cleaning it
5. Load. Write data into your target system either data lake or warehouse
6. Orchestrate. Use scheduling tools to automate steps
7. Track ~~pipl~~ pipeline
8. Validate and test: Ensure that the data is correct
9. Optimize. Tune for performance by indexing or batching
10. Update pipeline

28. Performance bottlenecks: There may be slow queries, inefficient transformations whereby there may be high I/O demands
Protecting ~~fr~~ against data loss, corruption and downtime.

Scalability issues: Handling growing volume of data and more users.

29. MySQL: often used in web applications, startups and small to medium businesses.
PostgreSQL: known for advanced features, robustness and extensibility. Good for complex queries
Microsoft SQL server: Enterprise-focused, integrates well with Microsoft tools, BI and analytics.
Oracle database: High-end enterprise workloads, large-scale OLTP systems, mission-critical applications and advanced features.

30. CSV: Simple, human readable text format, good for small datasets but lacks schema
JSON: Semi structured, great for nested data often used in APIs and NoSQL systems
Parquet: Columnar, binary format optimized for analytics (OLAP); supports efficient compression, predicate pushdown and fast reads.
Avro: Row based, schema based binary format, efficient for streaming, schema evolution and data interchange.