**Chapter 5 Summary**

* **Database Definition:**
  + An organized collection of data about related real-world items or concepts.
  + Data stored in bytes, grouped into fields and records.
* **Data Organization:**
  + Fields: Contain meaningful data items (e.g., names, addresses).
  + Records: Describe specific entities (e.g., a flight, a student).
  + Tables: Multiple records of the same type form a table in the database.
* **Field Types:**
  + Characters: Commonly used for names, addresses, and textual data.
  + Blobs: Store binary large objects like images, voice recordings.
* **Database Design Considerations:**
  + Decision-making about data organization crucial.
  + Examples: How names are stored (as a whole or separate components).
* **Transaction Processing vs. Decision-making Applications:**
  + Transaction Processing: Involves updating individual records, high transaction volumes.
  + Decision-making Applications: Analyze large volumes of records without updates, support data-driven decision-making. Different database structures for different purposes.

**Operational Databases:**

* Operational databases support business operations, store data about real-world items, and are updated during transactions (e.g., customer withdrawals, hotel check-ins).
* They serve specific purposes, like managing customer data in banks or room data in hotels.
* Relational approach, introduced by Dr. Edgar Codd, organizes data into tables with rows and columns, ensuring data integrity and minimizing redundancy.

**Relational Database Concepts:**

* Database tables resemble spreadsheet tables.
* Rows represent real-world items, columns (fields) store the same data for all rows.
* Primary keys uniquely identify rows, foreign keys connect records across different tables.
* Good database design minimizes redundancy, reducing errors and ensuring data accuracy.

**Entity-Relationship Diagramming (ERD):**

* ERDs use symbols to represent entities, attributes, relationships, and cardinalities in a database.
* Many-to-one, many-to-many, and one-to-one relationships can be modeled in ERDs.
* Attributes are listed within entity rectangles, primary keys are underlined, and foreign keys are starred.
* ERDs help in understanding and communicating database structures and designs.

**Normalization:**

* Normalization ensures clean, correct database structures by minimizing redundancy.
* Each data item is stored only once; entities are defined at the lowest possible level, and information is pushed up to higher levels.
* Normalized databases are more likely to function correctly and reduce the risk of errors.

**Distributed Databases:**

* Distributed databases store data in multiple locations and provide unified access.
* Partitioned databases can be horizontally (rows) or vertically (columns) fragmented, while replicated databases maintain copies in different places.
* Federated databases consist of multiple independent databases providing controlled access to their combined data.

**Where You Fit In:**

* Understanding operational databases and their relational structure is crucial for effective use in various job roles.
* ERDs are essential tools for both designing databases and understanding proposed designs.
* Decisions related to database distribution and design should be based on business needs and usage requirements, involving collaboration between technical and non-technical stakeholders.

**Differences in Database Usage:**

**Single vs. Multiple Record Access:** Decision-making databases involve accessing many records, not just one, making it difficult to optimize a database for both single-record and multiple-record access.

**Analysis vs. Transaction Processing:** Analysis does not change the data, and decision-making often does not require up-to-the-second data, unlike transaction processing which requires real-time data.

**Data Organization**: Transaction data is organized by application for speed, while decision-making data is organized by subject to provide an integrated view.

**Drawbacks of Relational Databases for Decision Support:**

**Long Queries and Locking:** Long queries can tie up relational databases for a considerable period, locking out transaction processing users until the query is complete to prevent errors caused by changes in the database during the query.

**Dimensional Databases:**

**Multiple Dimensions:** Transaction data can be represented in multidimensional cubes, allowing analysis along various dimensions. Dimensional databases are suitable for data warehouses, historical databases for decision-making.

**Challenges of Big Data:**

**Volume, Velocity, and Variety:** Big data is characterized by the sheer volume of data, the speed at which it is generated and needs to be analyzed, and the variety of data types, including structured, unstructured, audio, and video data.

**Hadoop and MapReduce:** Hadoop is a framework for running applications on large clusters of commodity hardware. It employs MapReduce, a computational approach where each computer analyzes its part of the data (Map) and the results are combined to produce the final output (Reduce). Hadoop allows parallel processing and can handle diverse data structures.

**Business Implications:**

**Flexible Data Analysis**: Hadoop enables businesses to analyze different types of data with varying structures simultaneously, providing valuable insights.

**Open Source Support:** Hadoop is supported by the open-source community, offering software, training, and expertise for businesses.

**Database Management Software (DBMS):**

* DBMS are system software that manage databases, providing an interface between users/applications and the underlying database.
* Databases contain metadata (data about data) which includes data element names, formats, storage locations, and access control information.
* Applications access databases through DBMS using Structured Query Language (SQL) to retrieve specific data.
* DBMS provide tools for database administrators (DBA) to create, modify, secure, and optimize databases.
* Personal DBMS (e.g., Microsoft Access, FileMaker Pro) cater to single users, making data entry and querying user-friendly through forms and simpler application development languages.

**Database Security:**

* Database security is crucial to prevent unauthorized access or modification of sensitive data, ensuring compliance with laws and privacy policies.
* Access to databases requires authorization (being allowed to use the database) and authentication (confirming the user's identity).
* Authentication methods include something you know (passwords), something you have (physical devices like dongles), something you are (biometric data), or something you do (voice or signature recognition).
* Access control methods include column access control (specific data elements), row access control (restricting access to specific rows), and type of access control (read, create, modify).
* Access permissions can be determined through role-based access, user-based access, or context-based access.
* Two security approaches are highlighted:
  + The traditional castle and moat approach (once inside the system, all resources are accessible).
  + The newer zero-trust approach (each access request is validated to ensure the user needs the specific resource for an authorized task).

**Business Implications:**

* The choice between security approaches (castle and moat vs. zero-trust) is a business decision, considering the trade-off between added security costs and benefits.
* Users and their managers must assess the balance between access benefits and security risks, making it a crucial business decision.