CS352 – ADVANCED DATABASE SYSTEMS

UNIT 1 – DATABASE SYSTEMS

Dr. John Conklin

Agenda

Introduction to Databases, Database Environments, and The Relational Model.

- Introduction to Databases
- Understanding Database Environments
- Exploring The Relational Model
- Summary and Q&A

INTRODUCTION TO DATABASES

Introduction to Databases

- Definition and Scope
  - Definition: A database environment configures hardware, software, data, procedures, and people designed to manage and manipulate databases.
  - Scope: Encompasses all elements and activities required to support the database, including physical storage, access methods, and administrative processes.

Introduction to Databases

- **Components of a Database Environment**

1. Hardware

1. Physical devices used to store and process data.
2. Includes servers, storage devices, networking equipment, and backup systems.

2. Software

1. Database Management System (DBMS) software to manage databases.
2. Application software to interact with the database.
3. Operating systems that support the DBMS.

3. Data

1. The actual information stored in databases.
2. Structured in tables, records, and fields.
3. Includes metadata, which is data about the data (e.g., data types, relationships).

4. Procedures

1. The instructions and rules that govern the design, operation, and use of databases.
2. Includes backup and recovery procedures, user access protocols, and maintenance routines.

5. People

1. Individuals involved in the database environment.
2. Roles include Database Administrators (DBAs), developers, end-users, and support staff.

Introduction to Databases

Types of Database Environments

1. **Centralized**
    1. All data is stored in a single location.
    2. Easier to manage and secure but can be a single point of failure.
2. **Distributed**
    1. Data is distributed across multiple locations.
    2. Improves reliability and performance but adds complexity to management.
3. **Cloud-based**
    1. Data is stored and managed in cloud services.
    2. Offers scalability, cost efficiency, and accessibility but depends on internet connectivity and third-party providers.

Comparison Table: Centralized vs. Distributed vs. Cloud-Based Database Environments

Feature	Centralized	Distributed	Cloud-Based
Definition	Data is stored in a single central location.	Data is spread across multiple locations.	Data is stored and managed in cloud services.
Management	Easier to manage and secure.	More complex management due to multiple locations.	Managed by third-party cloud providers.
Performance	Can be a bottleneck with high loads.	Generally better performance due to localization.	High scalability and performance with internet connectivity.
Reliability	Single point of failure.	Higher reliability through redundancy.	High reliability with cloud provider's infrastructure.
Scalability	Limited scalability.	Scalable but complex to implement.	Highly scalable on-demand resources.
Cost	Higher upfront costs for hardware.	Higher operational costs for maintenance.	Pay-as-you-go pricing model.
Accessibility	Limited to network availability.	Access may vary based on location.	Accessible from anywhere with internet.
Security	Easier to control security centrally.	More challenging to secure across locations.	Robust security measures by cloud providers, but dependent on them.
Maintenance	Simpler maintenance procedures.	Complex maintenance and synchronization.	Maintenance managed by cloud providers.
Backup and Recovery	Easier to manage centrally.	Requires coordinated backup strategy.	Built-in backup and recovery services.

Comparison Table

This table highlights the key features, advantages, and disadvantages of centralized, distributed, and cloud-based database environments, providing a clear comparison for easy understanding.

Introduction to Databases

Importance in Modern Applications

- Databases are the backbone of modern applications.
- Essential for storing, managing, and retrieving vast amounts of data efficiently.

Key Areas of Importance

1. **Data Management**
    1. **Structured Storage:** Efficiently organize and store data in a structured manner.
    2. **Data Integrity:** Ensure data accuracy and consistency.
    3. **Data Security:** Implement robust security measures to protect sensitive information.
2. **Performance and Scalability**
    1. **Fast Data Access:** Enable quick retrieval of data, enhancing application performance.
    2. **Scalable Solutions:** Support growing amounts of data and user requests without compromising performance.
3. **Support for Transactions**
    1. **ACID Properties:** Ensure reliable transactions with Atomicity, Consistency, Isolation, and Durability.
    2. **Concurrency Control:** Handle multiple transactions simultaneously without conflicts.

Introduction to Databases

4. Analytical Capabilities

1. **Data Analysis:** Facilitate advanced data analytics and business intelligence.
2. **Decision Making:** Provide insights that drive informed decision-making.

5. Integration and Interoperability

1. **Seamless Integration:** Integrate with various applications and services.
2. **Interoperability:** Ensure compatibility with different systems and platforms.

6. User Experience

1. **Personalization:** Store user preferences and history to provide personalized experiences.
2. **Real-time Updates:** Offer real-time data updates for dynamic content and services.

7. Backup and Recovery

Data Redundancy: Implement strategies for data backup and recovery.

Disaster Recovery: Ensure data availability and continuity in case of failures.

Key Database Concepts

Real-World Examples

- **E-commerce**: Manage product catalogs, customer data, orders, and transactions.
- **Social Media**: Store user profiles, posts, messages, and interactions.
- **Healthcare**: Maintain patient records, appointments, and medical histories.
- **Finance**: Handle transactions, customer accounts, and financial analytics.

Conclusion

In a nutshell, databases are the backbone of modern applications, ensuring functionality, efficiency, and reliability. They are the driving force behind data management, transaction support, and user experience enhancement.

UNDERSTANDING DATABASE ENVIRONMENTS

Database Environments

Introduction

- Understanding fundamental database concepts is essential for effectively managing and utilizing databases.
- These concepts form how data is stored, organized, and accessed.

Key Concepts

1. **Data vs. Information**
    1. **Data:** Raw facts and figures without context (e.g., numbers, dates, strings).
    2. **Information:** Data processed and given context, making it meaningful (e.g., a sales report).
2. **Database Management System (DBMS)**
    1. **Definition:** Software that handles the creation, management, and manipulation of databases.
    2. **Functions:**
        1. Data storage and retrieval
        2. Data security and integrity
        3. Backup and recovery
        4. Data access control
3. **Key Components of a Database System**
    1. **Database:** A collection of organized data that can be easily accessed, managed, and updated.
    2. **Tables:** Structures within a database that store data in rows and columns.
    3. **Records (Rows):** Individual entries in a table, representing a single data item.
    4. **Fields (Columns):** Attributes or properties of data stored in a table (e.g., name, age, address).

Key Concepts

4. Primary Key
- **Definition:** A unique identifier for each record in a table.
- **Purpose:** Ensures each record can be uniquely identified and accessed.

5. Foreign Key
- **Definition:** A field in one table that links to the primary key of another table.
- **Purpose:** Establishes relationships between tables, enabling the creation of relational databases.

6. Indexes
- **Definition:** Structures that improve the speed of data retrieval operations.
- **Purpose:** Enhance query performance by providing quick access to rows in a table.

Key Concepts

7. Normalization

- **Definition:** Organizing data to minimize redundancy and improve data integrity.
- **Forms:** Different levels (1NF, 2NF, 3NF) that provide table structure guidelines.

8. SQL (Structured Query Language)

- **Definition:** A standardized language used to interact with relational databases.
- **Common Commands:**
  - **SELECT:** Retrieve data from a database.
  - **INSERT:** Add new data into a database.
  - **UPDATE:** Modify existing data.
  - **DELETE:** Remove data from a database.

Here are some simple SQL query examples that demonstrate basic operations:

1. **SELECT statement**: - Retrieve all columns from a table: SELECT * FROM table_name;	2. **SELECT with WHERE clause**: - Retrieve specific columns based on a condition: SELECT column1, column2 FROM table_name WHERE condition;
3. **INSERT statement**: - Insert a new row into a table: INSERT INTO table_name (column1, column2) VALUES (value1, value2);	4. **UPDATE statement**: - Update existing records in a table: UPDATE table_name SET column1 = new_value WHERE condition;
5. **DELETE statement**: - Delete records from a table: DELETE FROM table_name WHERE condition;	6. **CREATE TABLE statement**: - Create a new table: CREATE TABLE table_name (column1 datatype, column2 datatype, ...);
7. **ALTER TABLE statement**: - Add a new column to an existing table: ALTER TABLE table_name ADD column_name datatype;	8. **DROP TABLE statement**: - Delete a table from the database: DROP TABLE table_name;

SQL Examples

These are some basic SQL queries that cover common operations like selecting data, inserting new data, updating existing data, deleting data, and managing tables in a database.

Database Environments

Introduction

- Understanding different database environments is crucial for choosing the right architecture based on application needs.
- Three primary types: Centralized, Distributed, and Cloud-based.

Centralized Database Environment

- **Definition**
  - All data is stored in a single central location.
- **Key Features**
  - Single server or a group of closely located servers.
  - Centralized control and management.
  - Easier to secure and maintain.
- **Advantages**
  - Simplified management and maintenance.
  - Easier implementation of security measures.
  - Centralized backup and disaster recovery.
- **Disadvantages**
  - Single point of failure.
  - Potential performance bottlenecks.
  - Limited scalability.
- **Use Cases**
  - Small to medium-sized businesses.
  - Applications with low to moderate data loads.
  - Systems where centralized control is crucial.

Distributed Database Environment

- **Definition**
  - Data is spread across multiple locations, connected via a network.
- **Key Features**
  - Data replication and distribution.
  - Increased reliability through redundancy.
  - Complex synchronization and consistency management.
- **Advantages**
  - Improved reliability and availability.
  - Better performance through localized data access.
  - Scalability by adding more nodes.
- **Disadvantages**
  - Complex management and maintenance.
  - Higher operational costs.
  - Security challenges across multiple locations.
- **Use Cases**
  - Large enterprises with multiple locations.
  - Applications requiring high availability and performance.
  - Systems needing distributed data processing.

Cloud-Based Database Environment

- **Definition**
  - Data is stored and managed in cloud services provided by third-party vendors.
- **Key Features**
  - Hosted on cloud infrastructure (e.g., AWS, Azure, Google Cloud).
  - On-demand resource allocation.
  - Managed services for database operations.
- **Advantages**
  - High scalability and flexibility.
  - Cost-efficient with pay-as-you-go model.
  - Access from anywhere with an internet connection.
  - Built-in backup, recovery, and security measures.
- **Disadvantages**
  - Dependence on internet connectivity.
  - Reliance on third-party providers for security and management.
  - Potential data sovereignty and compliance issues.
- **Use Cases**
  - Startups and businesses with fluctuating workloads.
  - Applications requiring global accessibility.
  - Systems that benefit from cloud service integrations.

Feature	Centralized	Distributed	Cloud-Based
Definition	Data is stored in a single central location.	Data is spread across multiple locations.	Data is stored and managed in cloud services.
Management	Easier to manage and secure.	More complex management due to multiple locations.	Managed by third-party cloud providers.
Performance	Can be a bottleneck with high loads.	Generally better performance due to localization.	High scalability and performance with internet connectivity.
Reliability	Single point of failure.	Higher reliability through redundancy.	High reliability with cloud provider's infrastructure.
Scalability	Limited scalability.	Scalable but complex to implement.	Highly scalable on-demand resources.
Cost	Higher upfront costs for hardware.	Higher operational costs for maintenance.	Pay-as-you-go pricing model.
Accessibility	Limited to network availability.	Access may vary based on location.	Accessible from anywhere with internet.
Security	Easier to control security centrally.	More challenging to secure across locations.	Robust security measures by cloud providers, but dependent on them.
Maintenance	Simpler maintenance procedures.	Complex maintenance and synchronization.	Maintenance managed by cloud providers.
Backup and Recovery	Easier to manage centrally.	Requires coordinated backup strategy.	Built-in backup and recovery services.

Comparison of Database Environments

These slides provide a detailed overview of the types of database environments, highlighting their key features, advantages, disadvantages, and typical use cases.

EXPLORING THE RELATIONAL MODEL

The Relational Model

Definition

- The relational model is a method of structuring data using relations, which are commonly known as tables.
- Proposed by E.F. Codd in 1970, it revolutionized database management.

Key Concepts

1.Tables (Relations)

- Fundamental structure in a relational database.
- Consists of rows (records) and columns (attributes).
- Each table represents a specific entity (e.g., customers, orders).

2.Rows (Tuples)

- Individual records in a table.
- Each row represents a single instance of the entity.
- Example: A row in a "Customers" table represents one customer.

3.Columns (Attributes)

- Define the properties of the entity.
- Each column holds a specific type of data (e.g., name, address, phone number).
- Example: Columns in a "Customers" table might include CustomerID, Name, and Email.

The Relational Model

4.Keys

- **Primary Key:** Unique identifier for each row in a table.
  - Ensures each record is uniquely identifiable.
  - Example: CustomerID in a "Customers" table.
- **Foreign Key:** A field in one table that links to the primary key of another table.
  - Establishes relationships between tables.
  - Example: CustomerID in an "Orders" table linking to the "Customers" table.

5.Relationships

- Define how tables relate to each other.
- Types of relationships:
  - One-to-One
  - One-to-Many
  - Many-to-Many

The Relational Model

Example: Simple Relational Database

- Customers Table:

CustomerID	Name	Email
1	John Doe	john@example.com
2	Jane Smith	jane@example.com

- Orders Table:

OrderID	OrderDate	CustomerID
101	2023-01-15	1
102	2023-01-16	2

The Relational Model

- **Advantages of the Relational Model**
  - **Simplicity:** Easy to understand and use.
  - **Flexibility:** Allows complex queries and data manipulation.
  - **Data Integrity:** Enforces data accuracy and consistency through constraints.
  - **Normalization:** Reduces data redundancy and improves data integrity.

SUMMARY AND Q&A

Recap of Key Points

Key Points

1. Definition and Importance

1. Databases are structured collections of data that support storage, retrieval, and management.
2. Crucial for modern applications across various domains like e-commerce, social media, healthcare, and finance.

2. Types of Databases

1. **Centralized:** Single location, easy to manage but has a single point of failure.
2. **Distributed:** Data spread across multiple locations, improves reliability and performance, but complex to manage.
3. **Cloud-Based:** Managed in the cloud, highly scalable and flexible, dependent on internet and third-party providers.

3. Key Database Concepts

1. **Data vs. Information:** Raw data versus processed information.
2. **DBMS:** Software that manages databases, ensuring data integrity, security, and efficient access.
3. **Tables, Records, and Fields:** Fundamental structures for organizing data.
4. **Primary and Foreign Keys:** Unique identifiers and relationships between tables.
5. **Indexes, Normalization, and SQL:** Tools and techniques for optimizing data management and access.

4. Relational Model

1. **Tables (Relations):** Core structure in relational databases.
2. **Rows and Columns:** Represent individual records and attributes.
3. **Keys and Relationships:** Ensure data uniqueness and define connections between tables.
4. **Advantages:** Simplicity, flexibility, data integrity, and reduced redundancy.

Recap: Key Points for Database Environments

Key Points

1. **Types of Database Environments**
    1. **Centralized:** All data stored in a single central location.
    2. **Distributed:** Data spread across multiple locations.
    3. **Cloud-Based:** Data stored and managed in cloud services.
2. **Centralized Database Environment**
    1. **Features:** Single server, centralized control, and management.
    2. **Advantages:** Easier to manage, secure, and maintain; centralized backup and recovery.
    3. **Disadvantages:** Single point of failure, potential performance bottlenecks, limited scalability.
    4. **Use Cases:** Small to medium-sized businesses, applications with low to moderate data loads.
3. **Distributed Database Environment**
    1. **Features:** Data replication and distribution across multiple locations.
    2. **Advantages:** Improved reliability and availability, better performance, enhanced scalability.
    3. **Disadvantages:** Complex management, higher operational costs, security challenges.
    4. **Use Cases:** Large enterprises, applications requiring high availability, distributed data processing.

Recap: Key Points for Database Environments

4. Cloud-Based Database Environment

1. **Features:** Hosted on cloud infrastructure, on-demand resource allocation, managed services.
2. **Advantages:** High scalability, cost-efficiency, global accessibility, built-in backup and security.
3. **Disadvantages:** Dependence on internet connectivity, reliance on third-party providers, potential data sovereignty issues.
4. **Use Cases:** Startups, businesses with fluctuating workloads, applications requiring global access.

5. Comparison Table Highlights

1. **Management:** Easier in centralized, complex in distributed, managed by providers in cloud-based.
2. **Performance:** Potential bottlenecks in centralized, better in distributed, high scalability in cloud-based.
3. **Reliability:** Single point of failure in centralized, higher in distributed, very high in cloud-based.
4. **Scalability:** Limited in centralized, scalable but complex in distributed, highly scalable in cloud-based.
5. **Cost:** Higher upfront in centralized, higher operational in distributed, pay-as-you-go in cloud-based.
6. **Accessibility:** Limited in centralized, varies in distributed, anywhere with internet in cloud-based.
7. **Security:** Easier in centralized, challenging in distributed, robust but dependent on providers in cloud-based.
8. **Maintenance:** Simpler in centralized, complex in distributed, managed by providers in cloud-based.
9. **Backup and Recovery:** Easier in centralized, requires strategy in distributed, built-in services in cloud-based.

Recap of The Relational Model

- **Concept**:
  - Introduced by E.F. Codd in 1970.
  - Organizes data into tables (relations).
- **Key Components**:
  - **Tables (Relations)**: Represents entities or relationships.
  - **Rows (Tuples)**: Individual records in tables.
  - **Columns (Attributes)**: Fields that define characteristics of data.
- **Primary Keys**:
  - Unique identifier for each row.
  - Ensures each row is distinct.
- **Relationships**:
  - **Foreign Keys**: Links tables together.
  - **Normalization**: Reduces redundancy and dependency.
- **Operations**:
  - **Select**: Retrieves specific rows.
  - **Project**: Retrieves specific columns.
  - **Join**: Combines related tables.

Recap of The Relational Model

- **Advantages**:
  - Simplicity and structural independence.
  - Support for data integrity and consistency.
- **Applications**:
  - Widely used in relational databases (e.g., MySQL, PostgreSQL).
  - Foundation for SQL (Structured Query Language).
- **Challenges**:
  - Performance issues with complex queries.
  - Not suitable for all data types (e.g., hierarchical data).

Recap of the Importance of Understanding These Concepts

- **Foundation of Database Design**:
  - **Structure**: Learn how data is organized into tables, rows, and columns.
  - **Normalization**: Ensure efficient storage and minimize redundancy.
  - **Integrity**: Maintain data accuracy through primary and foreign keys.
- **Query Optimization**:
  - **Efficiency**: Understand how to write optimized queries using SQL.
  - **Joins**: Master techniques for combining data from multiple tables.
- **Data Consistency and Integrity**:
  - **Constraints**: Implement rules to enforce data integrity (e.g., uniqueness, referential integrity).
  - **Transactions**: Manage atomicity, consistency, isolation, and durability (ACID properties).
- **Scalability and Performance**:
  - **Indexing**: Utilize indexes for faster data retrieval.
  - **Normalization**: Reduce redundant data to improve performance.

Recap of the Importance of Understanding These Concepts

- **Data Modeling and Analysis**:
  - **Design**: Create effective data models for specific business requirements.
  - **Analysis**: Extract meaningful insights through complex queries and reporting.
- **Compatibility and Interoperability**:
  - **Standardization**: Ensure compatibility with SQL standards.
  - **Integration**: Facilitate seamless data exchange between systems.
- **Career Advancement**:
  - **Skills**: Develop fundamental skills highly valued in database administration and development roles.
  - **Problem-solving**: Solve real-world data management challenges effectively.
- **Future Trends**:
  - **Big Data**: Lay the groundwork for handling large volumes of data.
  - **Cloud Computing**: Understand implications for database management in cloud environments.

INDIVIDUAL PROJECT

Individual Project

Description

The current database that your company is using keeps track of all of the customers and suppliers as well as products offered for sale, but the current environment has all of these data spread out across multiple different databases, and it is difficult to see a complete picture of the environment. You have been tasked to consolidate the environment into a single database environment with the end goal of adding a data warehouse for the company.

Your manager is excited with the project description, is anxious to have a new database built for the company, and more excited that at the end of the project he will also have a data warehouse to help make strategic decisions. To start, your manager would like to understand the benefits of following a formal design methodology, especially with respect to database design. Describe how and why the company can benefit from spending time planning at the beginning of the project, instead of just jumping in and developing the applications to meet the perceived needs. To ensure that you give the best response, you choose to describe the 3-level ANSI-SPARC architecture and want to make sure you discuss how the use of this will promote data independence to save time in the long run.

Individual Project

- Your paper should contain the following information:
  - Describe the 3-level architecture.
  - Describe data independence.
  - Talk about the differences between a DA and a DBA.
  - Discuss the pros and cons of having a separate DA and DBA or the need for 1 person doing it all.

You should also take this opportunity to create the template for your entire project and create a Word document that you will add to each remaining assignment.

The document should follow this format:

Advanced Database Systems Project document shell: Use MS Word

Title Page

- Course number and name
- Project name
- Student name
- Date

Individual Project

Table of Contents

- Use autogenerated TOC
- Separate page
- Maximum of 3 levels deep
- Update fields of TOC so it is up-to-date before submitting project

Individual Project

- Section headings (create each heading on a new page with TBD as the content, except for the sections listed under "New content" in each week's assignment)
  - Project Outline
  - The Database Models, Languages, and Architecture
  - Database System Development Life Cycle
  - Database Management Systems
  - Advanced SQL
  - Web and Data Warehousing and Mining in the Business World

Individual Project

The Database Models, Languages, and Architecture (Week 1: IP)

- A description of the 3-level ANSI architecture model
- A description of data independence
- The difference in responsibility between:
  - Data administrator
  - Database administrator

Add the discussion about the 3-layer ANSI architecture, data independence, and the different administrators to the section titled "The Database Models, Languages, and Architecture."

Contact Information

Email:	Jconklin@coloradotech.edu
Phone:	602.796.5972
Website:	http://drjconklin.com
Office Hours:	Wednesdays: 6:00 PM – 7:00 PM (CST)
	Saturdays: 11:00 AM – 12:00 PM (CST)
Live Chats:	Thursdays: 7:00 PM – 8:00 PM (CST)

References

Frisendal, T. (2023). *History of Data Modeling.* *http://graphdatamodeling.com/GraphDataModeling/History.html*

interviewbit.com. (2023). Components of DBMS (Database Management System). https://www.interviewbit.com/blog/components-of-dbms/

Kelly, D. (2022). A brief history of databases: From relational, to NoSQL, to distributed SQL. https://www.cockroachlabs.com/blog/history-of-databases-distributed-sql/

Modise, M. (n.d.). Chapter 1. Introduction to the Module. In. https://www.cs.uct.ac.za/mit_notes/database/htmls/chp01.html#

tudip.com. (2022). *9 Common Database Management Challenges and How to Fix them.* *https://tudip.com/blog-post/9-common-database-management-challenges-and-how-to-fix-them/*