Outline

- Introduction
- Background
  - Distributed DBMS Architecture
  - Distributed Database Design (Briefly)
  - Distributed Query Processing (Briefly)
  - Distributed Transaction Management (Extensive)
- Building Distributed Database Systems (RAID)
- Mobile Database Systems
- Privacy, Trust, and Authentication
- Peer to Peer Systems
Useful References

- Textbook *Principles of Distributed Database Systems*,
  Chapter 1.4-1.7.1
Distributed Database - User View
Distributed DBMS - Reality
Potentially Improved Performance

- Proximity of data to its points of use
  - Requires some support for fragmentation and replication

- Parallelism in execution
  - Inter-query parallelism
  - Intra-query parallelism
System Expansion

- Issue is database scaling
- Peer to Peer systems
- Communication overhead
Distributed DBMS Issues

- Distributed Database Design
  - how to distribute the database
  - replicated & non-replicated database distribution
  - a related problem in directory management

- Query Processing
  - convert user transactions to data manipulation instructions
  - optimization problem
  - \( \text{min}\{\text{cost} = \text{data transmission} + \text{local processing}\} \)
  - general formulation is NP-hard
Distributed DBMS Issues

- **Concurrency Control**
  - Synchronization of concurrent accesses
  - Consistency and isolation of transactions' effects
  - Deadlock management

- **Reliability**
  - How to make the system resilient to failures
  - Atomicity and durability

- **Privacy/Security**
  - Keep database access private
  - Protect against malicious activities

- **Trusted Collaborations (Emerging requirements)**
  - Evaluate trust among users and database sites
  - Enforce policies for privacy
  - Enforce integrity
Relationship Between Issues
Related Issues

- **Operating System Support**
  - operating system with proper support for database operations
  - dichotomy between general purpose processing requirements and database processing requirements

- **Open Systems and Interoperability**
  - Distributed Multidatabase Systems
  - More probable scenario
  - Parallel issues

- **Network Behavior**
Outline

- Introduction
- Background
- Distributed DBMS Architecture
  - Introduction to Database Concepts
  - Architecture, Schema, Views
  - Alternatives in Distributed Database Systems
  - Datalogical Architecture
  - Implementation Alternatives
  - Component Architecture
- Distributed Database Design (Briefly)
- Distributed Query Processing (Briefly)
- Distributed Transaction Management (Extensive)
- Building Distributed Database Systems (RAID)
- Mobile Database Systems
- Privacy, Trust, and Authentication
- Peer to Peer Systems
Architecture of a Database System

- Background materials of database architecture
- Defines the structure of the system
  - components identified
  - functions of each component defined
  - interrelationships and interactions between components defined
ANSI/SPARC Architecture

- Users
  - External Schema
    - External view
  - Conceptual Schema
    - Conceptual view
  - Internal Schema
    - Internal view
Standardization

Reference Model

- A conceptual framework whose purpose is to divide standardization work into manageable pieces and to show at a general level how these pieces are related to one another.

Approaches

- Component-based
  - Components of the system are defined together with the interrelationships between components.
  - Good for design and implementation of the system.
- Function-based
  - Classes of users are identified together with the functionality that the system will provide for each class.
  - The objectives of the system are clearly identified. But how do you achieve these objectives?
- Data-based
  - Identify the different types of describing data and specify the functional units that will realize and/or use data according to these views.
Conceptual Schema Definition

RELATION EMP [  
  KEY = {ENO}  
  ATTRIBUTES = {  
    ENO : CHARACTER(9)  
    ENAME : CHARACTER(15)  
    TITLE : CHARACTER(10)  
  }  
]  

RELATION PAY [  
  KEY = {TITLE}  
  ATTRIBUTES = {  
    TITLE : CHARACTER(10)  
    SAL : NUMERIC(6)  
  }  
]
## Conceptual Schema Definition

**RELATION PROJ** [

**KEY** = \{PNO\}

**ATTRIBUTES** = {

  PNO : CHARACTER(7)
  PNAME : CHARACTER(20)
  BUDGET : NUMERIC(7)

}

]

**RELATION ASG** [

**KEY** = \{ENO,PNO\}

**ATTRIBUTES** = {

  ENO : CHARACTER(9)
  PNO : CHARACTER(7)
  RESP : CHARACTER(10)
  DUR : NUMERIC(3)

}

]
Internal Schema Definition

RELA$$TION$$ EMP [  
  KEY = {ENO}  
  ATTRIBUTES = {  
    ENO : CHARACTER(9)  
    ENAME : CHARACTER(15)  
    TITLE : CHARACTER(10)  
  }  
]  

↓

INTERNAL_REL EMPL [  
  INDEX ON E# CALL EMINX  
  FIELD = {  
    HEADER : BYTE(1)  
    E# : BYTE(9)  
    ENAME : BYTE(15)  
    TITLE : BYTE(10)  
  }  
]
Create a BUDGET view from the PROJ relation

\[
\text{CREATE VIEW BUDGET(PNAME, BUD) AS SELECT PNAME, BUDGET FROM PROJ}
\]
Create a Payroll view from relations EMP and TITLE_SALARY

```
CREATE VIEW PAYROLL (ENO, ENAME, SAL)
AS SELECT EMP.ENO, EMP.ENAME, PAY.SAL
FROM EMP, PAY
WHERE EMP.TITLE = PAY.TITLE
```