Database Systems

Introduction

Introduction

What is a Database?

A collection of related pieces of data:

- Representing/capturing the information about a real-world enterprise or part of an enterprise.
- Collected and maintained to serve specific data management needs of the enterprise.
- Activities of the enterprise are supported by the database and continually update the database.

An Example

University Database:

Data about students, faculty, courses, research-laboratories, course registration/enrollment etc.

Reflects the state of affairs of the academic aspects of the university.

Purpose: To keep an accurate track of the academic activities of the university.

Database Management System (DBMS)

A *general purpose* software system enabling:

- Creation of large disk-resident databases.
- Posing of data retrieval queries in a standard manner.
- Retrieval of query results efficiently.
- Concurrent use of the system by a large number of users in a consistent manner.
- Guaranteed availability of data irrespective of system failures.

OS File System Storage Based Approach

- Files of records – used for data storage
  - data redundancy – wastage of space
  - maintaining consistency becomes difficult
- Record structures – hard coded into the programs
  - structure modifications – hard to perform
- Each different data access request (a query)
  - performed by a separate program
  - difficult to anticipate all such requests
- Creating the system
  - requires a lot of effort
- Managing concurrent access and failure recovery are difficult

DBMS Approach

DBMS

- separation of data and metadata
- flexibility of changing metadata
- program-data independence

Data access language

- standardized – SQL
- ad-hoc query formulation – easy

System development

- less effort required
- concentration on logical level design is enough
- components to organize data storage

  process queries, manage concurrent access, recovery from failures, manage access control

  are all available

Data Model

Collection of conceptual tools to describe the database at a certain level of abstraction.

- *Conceptual Data Model*
  - a high level description
  - useful for requirements understanding.
- *Representational Data Model*
  - describing the logical representation of data without giving details of physical representation.
- *Physical Data Model*
  - description giving details about record formats, file structures etc.

E/R (Entity/Relationship) Model

- A conceptual level data model.
- Provides the concepts of *entities*, *relationships* and *attributes*.

The University Database Context

Entities: *student*, *faculty member*, *course*, *departments* etc.

Relationships: *enrollment* relationship between student & course,
employment relationship between faculty member, department etc.

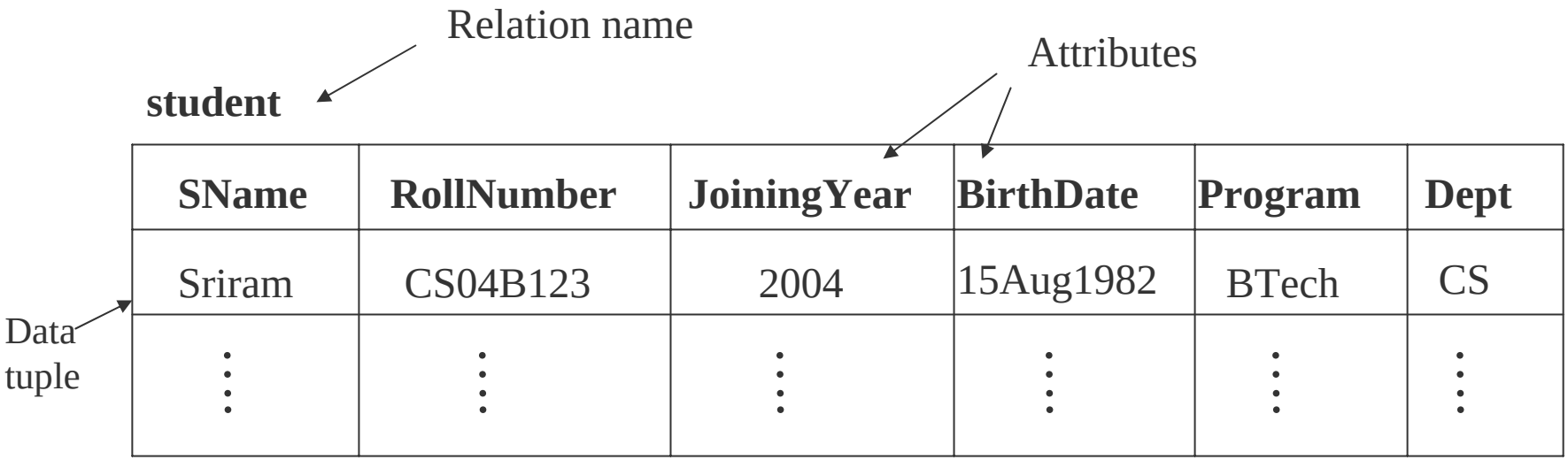
Attributes: *name*, *rollNumber*, *address* etc., of *student* entity,
name, *empNumber*, *phoneNumber* etc., of faculty entity etc.

More details will be given in the E/R Model Module.

Representational Level Data Model

Relational Model : Provides the concept of a relation.

In the context of university database:



Relation scheme: Attribute names of the relation.

Relation data/instance: set of data tuples.

More details will be given in Relational Data Model Module.

Data versus Schema or Meta-Data

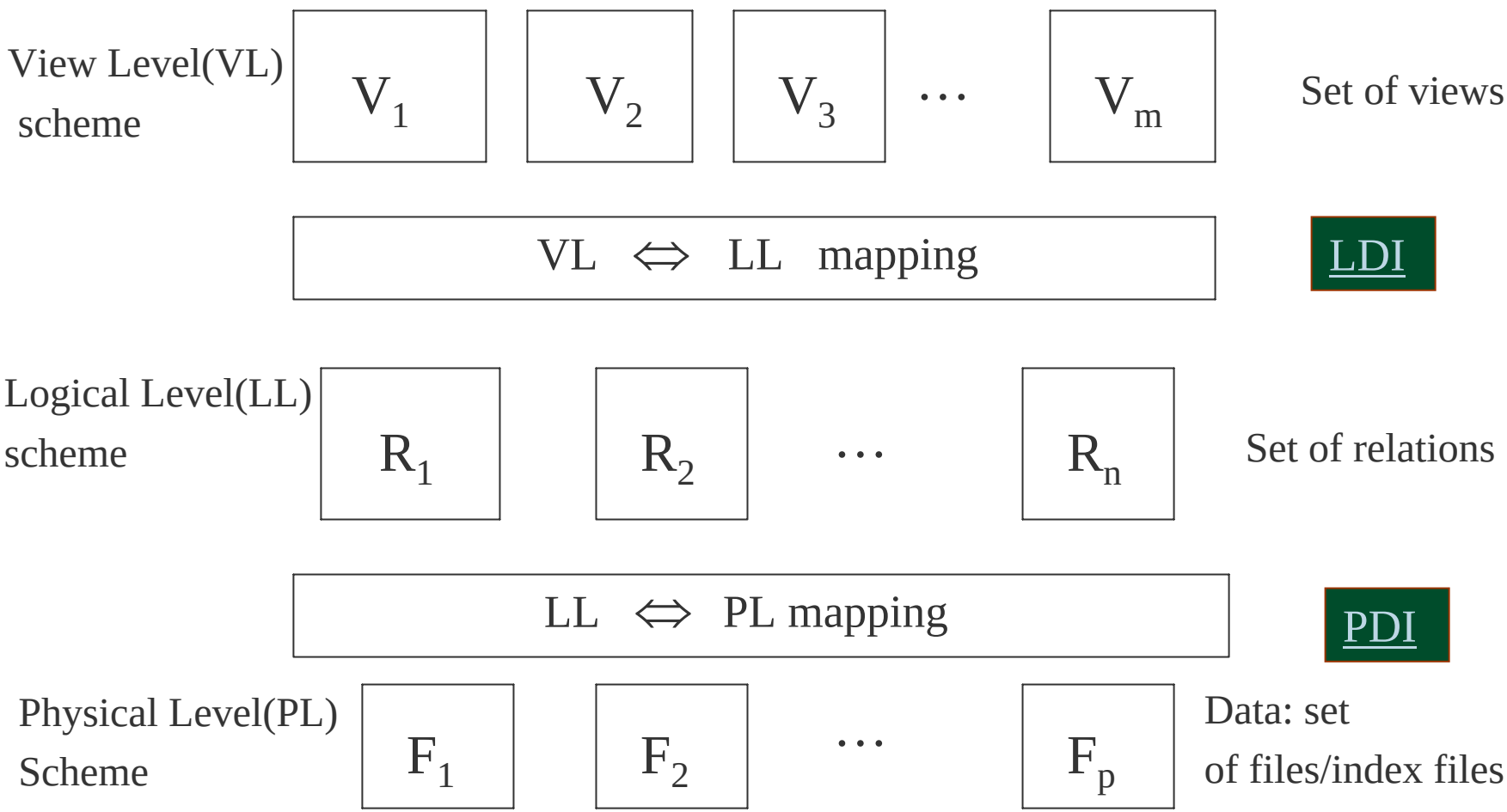
- DBMS is generic in nature
  - not tied to a single database
  - capable of managing several databases at a time
- Data and schema are stored separately.

- In RDBMS context:

  Schema – table names, attribute names with their data types for each table and constraints etc.

- Database definition – setting up the skeleton structure
- Database Loading/populating – storing data

Abstraction Levels in a DBMS: Three-Schema Architecture



Three-schema Architecture(1/2)

View Level Schema

Each view describes an aspect of the database relevant to a particular group of users.

For instance, in the context of a library database:

- Books Purchase Section
- Issue/Returns Management Section
- Users Management Section

Each section views/uses a portion of the entire data.

Views can be set up for each section of users.

Three-schema Architecture(2/2)

Logical Level Schema

- Describes the logical structure of the entire database.
- No physical level details are given.

Physical Level Schema

- Describes the physical structure of data in terms of record formats, file structures, indexes etc.

Remarks

- Views are optional
  - Can be set up if the DB system is very large and if easily identifiable user-groups exist
- The logical scheme is essential
- Modern RDBMS's hide details of the physical layer

Physical Data Independence

The ability to modify physical level schema without affecting the logical or view level schema.

Performance tuning – modification at physical level creating a new index etc.

Physical Data Independence – modification is localized

- achieved by suitably modifying PL-LL mapping.
- a very important feature of modern DBMS.

Three Schema Arch

Logical Data Independence

The ability to change the logical level scheme without affecting the view level schemes or application programs

Adding a new attribute to some relation

- no need to change the programs or views that don't require to use the new attribute

Deleting an attribute

- no need to change the programs or views that use the remaining data
- view definitions in VL-LL mapping only need to be changed for views that use the deleted attribute

Three-schema Architecture

Development Process of a Database System (1/2)

Step 1. Requirements collection

- *Data model requirements*
  - various pieces of data to be stored and the interrelationships.
  - presented using a conceptual data model such as E/R model.
- *Functional requirements*
  - various operations that need to be performed as part of running the enterprise.
    - acquiring a new book, enrolling a new user, issuing a book to the user, recording the return of a book etc.

Development process of a database system (2/2)

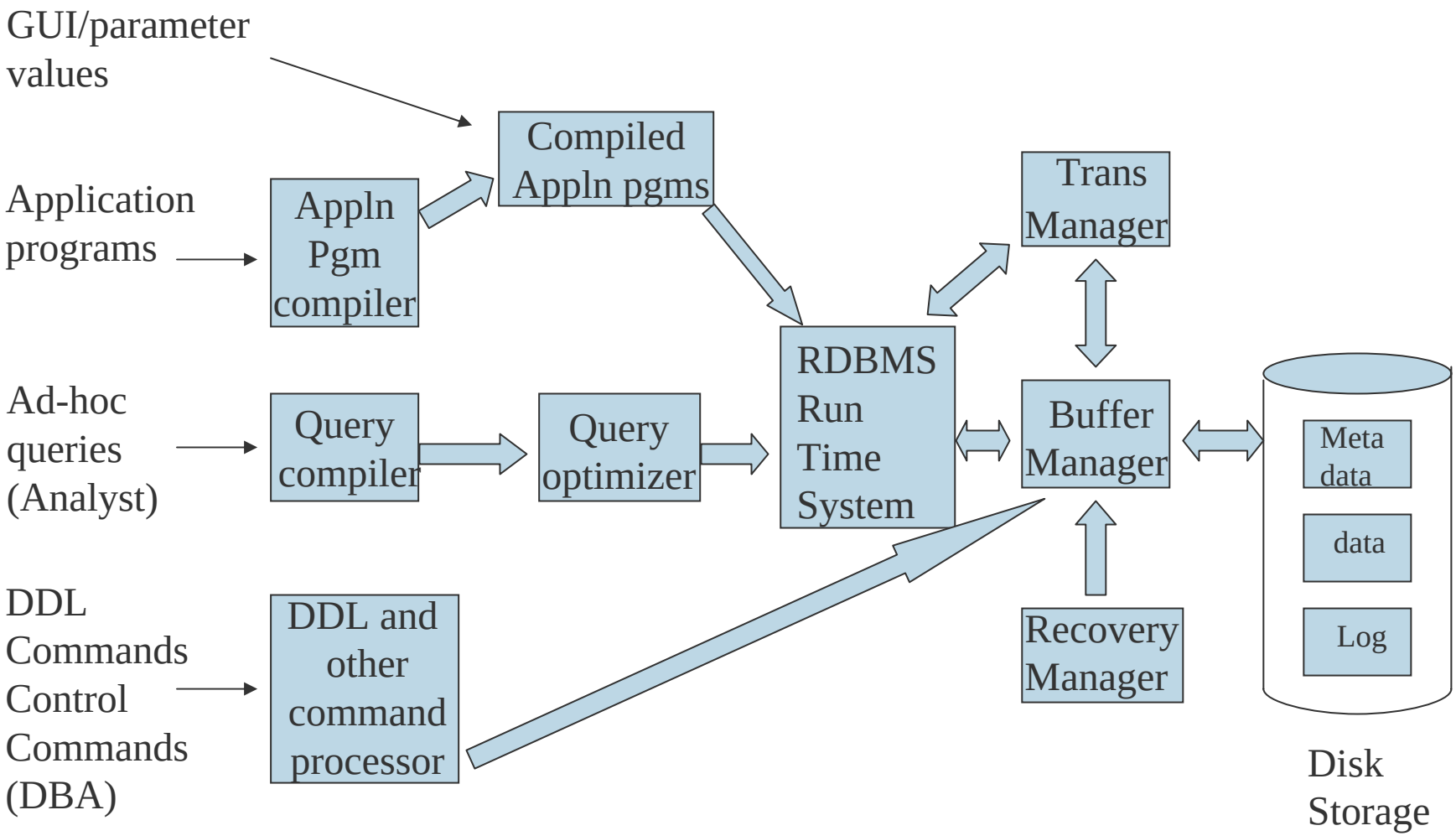
Step 2. Convert the data model into a representational level model

- typically relational data model.
- choose an RDBMS system and create the database.

Step 3. Convert the functional requirements into application programs

- programs in a high-level language that use embedded SQL to interact with the database and carry out the required tasks.

Architecture of an RDBMS system



Architecture Details (1/3)

Disk Storage:

Meta-data – schema

- table definition, view definitions, mappings

Data – relation instances, index structures

statistics about data

Log – record of database update operations

essential for failure recovery

DDL and other command processor:

Commands for relation scheme creation

Constraints setting

Commands for handling authorization and data access control

Architecture Details (2/3)

Query compiler

Compiles
- SQL adhoc queries
- update / delete commands

Query optimizers

Selects a near optimal plan for executing a query

- relation properties and index structures are utilized

Application Program Compiler

Preprocess to separate embedded SQL commands

Use host language compiler to compile rest of the program

Integrate the compiled program with the libraries for SQL commands supplied by RDBMS

Architecture Details (3/3)

RDBMS Run Time System:
- Executes Compiled queries, Compiled application programs
- Interacts with Transaction Manager, Buffer Manager

Transaction Manager:
- Keeps track of start, end of each transaction
- Enforces concurrency control protocols

Buffer Manager:
- Manages disk space
- Implements paging mechanism

Recovery Manager:
- Takes control as restart after a failure
- Brings the system to a consistent state before it can be resumed

Roles for people in an Info System Management (1/2)

Naive users / Data entry operators

- Use the GUI provided by an application program
- Feed-in the data and invoke an operation
  - e.g., person at the train reservation counter,
    person at library issue / return counter
- No deep knowledge of the IS required

Application Programmers

- Embed SQL in a high-level language and develop programs to handle functional requirements of an IS
- Should thoroughly understand the logical schema or relevant views
- Meticulous testing of programs - necessary

Roles for people in an Info System management (2/2)

Sophisticated user / data analyst:

- Uses SQL to generate answers for complex queries

DBA (Database Administrator)

- Designing the logical scheme
- Creating the structure of the entire database
- Monitor usage and create necessary index structures to speed up query execution
- Grant / Revoke data access permissions to other users etc.

Text Books

- Ramez Elmasri and Shamkant B Navathe, *Fundamentals of Database Systems*, 3rd Edition, Addison Wesley, 2000.
- Raghu Ramakrishnan and Johannes Gehrke, *Database Management Systems*, 3rd Edition, McGraw Hill, 2003.
- A Silberschatz, H F Korth and S Sudarshan, *Database System Concepts*, 5th Edition, 2006.
- H Garcia-Molina, J D Ullman, and Jennifer Widom, Database Systems-The Complete Book, Pearson Education, 2004.