Database Systems

Lecture 3,4,5

Chapter # 2

Database System Concepts and Architecture

eman.shahid@nu.edu.pk

Chapter Outlines

- Data Model, Schema and Instance
- Three schema architecture
- Data independence
- Classification of DBMS
- Database languages & Interfaces
- Database systems environment

What is Data Model?

Data Model

- A collection of concepts that can be used to describe the structure of a database, the set of basic operations for specifying retrievals and updates on the database and certain constraints that the database must fulfil.
- Provides the necessary means to achieve data abstraction

Database Structure and Operations

- **Structure of a database:** data types, relationships, and constraints that apply to the data.
- **User defined operations:** An example of a user-defined operation could be COMPUTE_GPA, which can be applied to a STUDENT object.
- **Generic operations:** insert, delete, modify, or retrieve any kind of object are often included in the basic data model operations.
- **Constraints:** Constraints are restrictions, must be followed so that we can get valid data.

Categories of Data Models

- **Conceptual Data Model (High-level):** Provide concepts that are close to the way many users perceive data. (Also called entity-based or object-based data models.)
  - **Concepts:** entities, attributes, and relationships.
  - **Entity:** real-world object or concept i.e. employee or a project.
  - **Attribute:** describes an entity i.e. employee's name or salary.
  - **Relationship:** association among the entities, for example, a works-on relationship between an employee and a project.

Categories of Data Models

- **Implementation or (Representational) data models:** represent data by using record structures and hence are sometimes called record-based data models.
- **Physical data models:** describe how data is stored as files in the computer by representing information such as record formats, record orderings, and access paths.
  - Access path: search structure that makes the search for particular database records efficient, such as indexing or hashing.

Database Schema

- **Database Schema:** The description of a database is called the database schema, which is specified during database design and is not expected to change frequently.
- Includes descriptions of the database structure and the constraints that should hold on the database.
- **Schema Diagram:** An ***illustrative*** display of (most aspects of) a database schema.
- **Schema Construct:** A ***component*** of the schema or an object within the schema, e.g., STUDENT, COURSE.

Schema diagram for the database

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Schema Construct: Each object in the schema i.e. STUDENT or COURSE

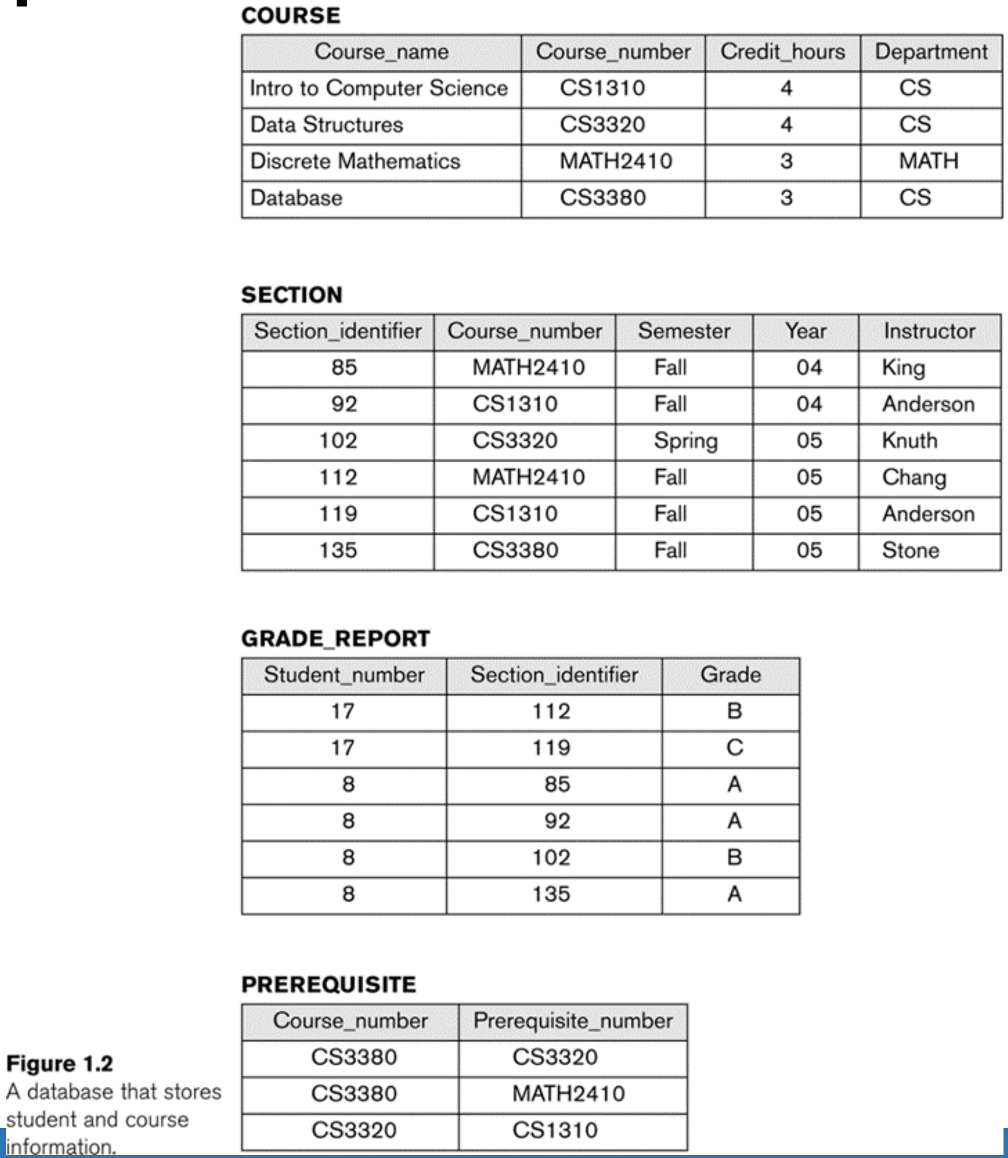
Database State

- The data in the database at a particular moment in time is called a database state or snapshot.
- It is also called the current set of occurrences or instances in the database.
- Each schema construct has its own current set of instances; for example, the STUDENT construct will contain the set of individual student entities (records) as its instances.
- The term instance is also applied to individual database components, e.g. record instance, table instance etc.

Database States

- **Initial Database State:** Refers to the database when it is loaded
- **Valid State:** A state that satisfies the structure and constraints of the database.
- **Distinction**
  - The database schema changes very infrequently. The database state changes every time the database is updated, or any insertion or deletion is made.
  - Schema is also called intension, whereas state is called extension.

Example of Database State



COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

The Three-Schema Architecture/ ANSI/SPARC

- The goal of the three-schema architecture is to separate the user applications from the physical database.
- Proposed to support and visualize DBMS characteristics of:
  - Use of a catalog to store the database description (schema) so as to make it self-describing
  - **Program-data independence.**
  - Support of **multiple views** of the data.

The Three-Schema Architecture/ ANSI/SPARC

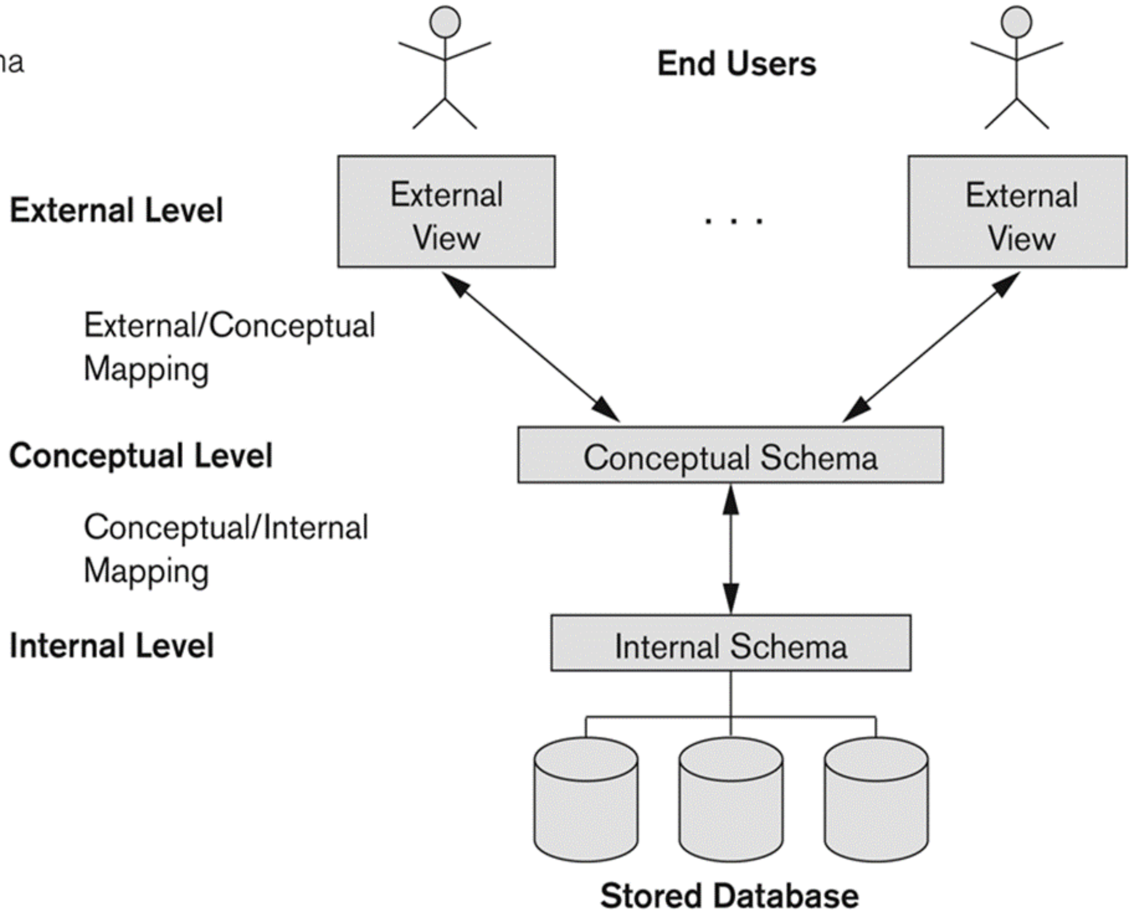


Figure 2.2
The three-schema architecture.

Objectives of Three schema Architecture

- The main objective of three level architecture is to enable multiple users to access the same data with a personalized view while storing the underlying data only once. Thus it separates the user's view from the physical structure of the database. This separation is desirable for the following reasons:
- Different users need different views of the same data.
- The approach in which a particular user needs to see the data may change over time.
- The users of the database should not worry about the physical implementation and internal workings of the database such as data compression and encryption techniques, hashing, optimization of the internal structures etc.
- All users should be able to access the same data according to their requirements.
- DBA should be able to change the conceptual structure of the database without affecting the user's
- Internal structure of the database should be unaffected by changes to physical aspects of the storage.

The Three-Schema Architecture/ ANSI/SPARC

- **Internal level:** describes the physical storage structure of the database.
  - Internal schema: describes the complete details of data storage and access paths for the database.
- **Conceptual level:** hide the details of the physical storage structure.
  - Concentrates on describing entities, data types, relationships, user operations, and constraints.
  - A representational data model is used to describe the conceptual schema
- **The external or view level**: includes a number of external schemas or user views.
  - Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.
  - each external schema is typically implemented using a representational data model

Data Independence

- Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs.
  - Changes to constraints can be applied to the conceptual schema without affecting the external schemas or application programs.

Data Independence

- Physical data independence: capacity to change the internal schema without having to change the conceptual schema.
  - Changes to the internal schema may be needed because some physical files were reorganize i.e. by creating additional access structures—to improve the performance of retrieval or update.
  - If the same data as before remains in the database, we should not have to change the conceptual schema.

Which data independence is difficult to achieve?

Logical or Physical?

DBMS Languages

- **Data definition language (DDL):**
  - Used by the DBA and by database designers to define conceptual and internal schemas.
  - **DBMS DDL compiler:** process DDL statements in order to identify descriptions of the schema constructs and to store the schema description in the DBMS catalog.
  - **Tasks of DDL:** Create, Alter, Drop , Truncate, Rename.
- **Storage definition language (SDL):** If a clear separation is maintained between the conceptual and internal levels, the DDL is used to specify the conceptual schema only and **SDL** is used to specify the internal schema.

DBMS Languages

- **View definition language (VDL):** to specify user views and their mappings to the conceptual schema.
  - SQL is used in the role of VDL to define user or application views as results of predefined queries.
- In most DBMSs the DDL is used to define both conceptual and external schemas.

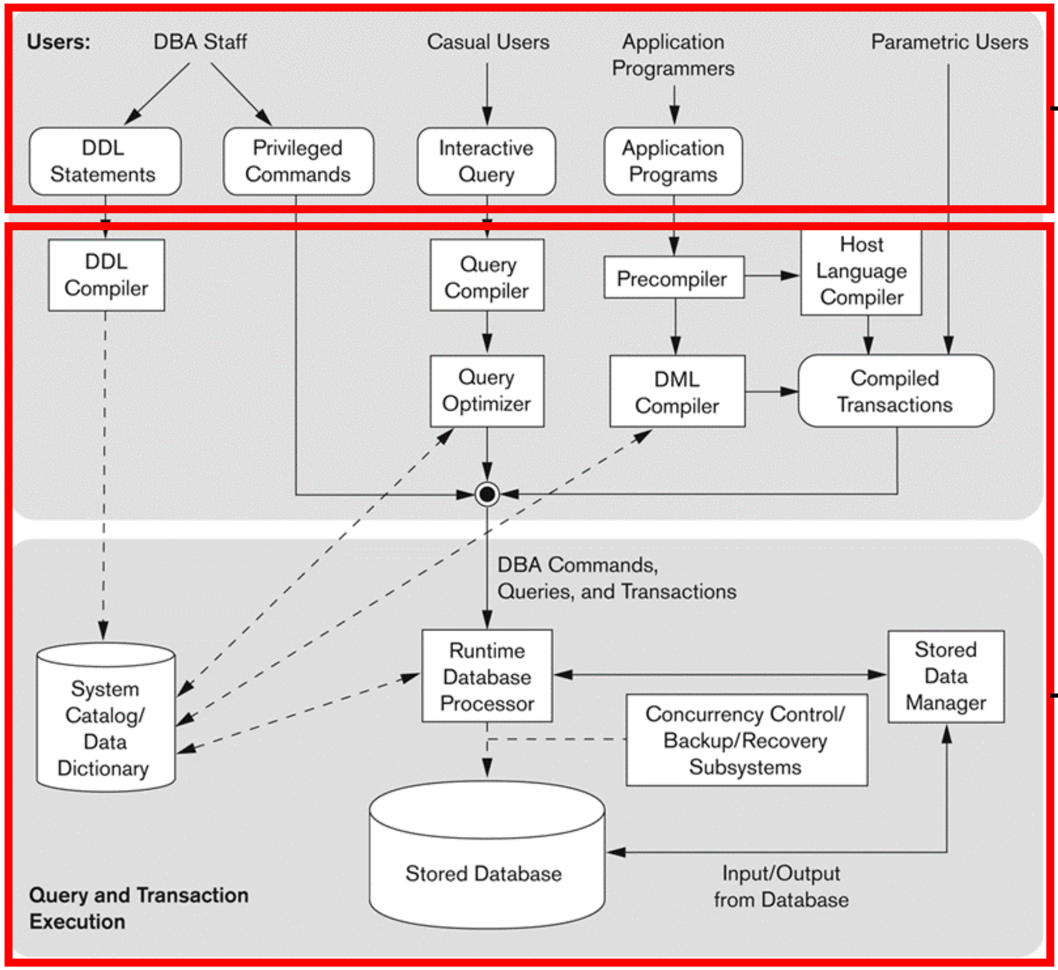
DBMS Languages

- **Data manipulation language(DML) / Data sublanguage**
    - For manipulating data, manipulations include retrieval, insertion, deletion, and modification of the data.
    - **High-level** or **Nonprocedural** DML:
        - **Set-at-a-time or set-oriented:** many records in a single DML statement.
        - SQL is a High Level DML specifies which data to retrieve not how to retrieve it. Also called Declarative Languages.
    - **Low-level** or **Procedural DML**
        - **Record-at-a-time:** use programming language constructs, such as looping, to retrieve and process each record from a set of records.
- Programming languages used to write programs and access databases using DML statements are called host language: i.e. C++, java, python etc.

DBMS Languages

- SQL relational database language which represents a combination of DDL, VDL, and DML, as well as statements for constraint specification, schema evolution, and many other features.
- **DCL(Data Control Language):**
  - Deals with rights, permissions and other controls of the database system. i.e. GRANT, REVOKE
- **TCL(transaction Control Language):** deals with the transaction within the database. i.e. COMMIT, ROLLBACK, SAVEPOINT

The Database System Environment



Users of database environment and their interfaces

internal modules of the DBMS responsible for storage of data and processing of transactions

Figure 2.3
Component modules of a DBMS and their interactions.

DBMS Interfaces

- **Menu-based Interfaces for Web Clients or Browsing**
- **Apps for Mobile Devices:** allow users to access their data through a mobile phone.
  - For example, banking, reservations etc.
- **Forms-based Interfaces:** designed and programmed for naive users as interfaces to canned transactions.
- **Graphical User Interfaces:** displays a schema to the user in diagrammatic form.
  - The user then can specify a query by manipulating the diagram.
  - GUIs utilize both menus and forms.

DBMS Interfaces

- **Natural Language Interfaces:** These interfaces accept requests written in English or some other language and attempt to understand them.
  - Has its own schema similar to the database conceptual schema, as well as a dictionary of important words.
  - It refers to the words in its schema, as well as to the set of standard words in its dictionary, that are used to interpret the request.
  - If the interpretation is successful, the interface generates a high-level query corresponding to the natural language request and submits it to the DBMS for processing;
  - otherwise, a dialogue is started with the user to clarify the request.

DBMS Interfaces

- **Keyword-based Database Search:** similar to Web search engines, which accept strings of natural language words and match them with documents at specific sites (for local search engines) or Web pages on the Web at large (for engines like Google).
  - Use predefined indexes on words and use ranking functions to retrieve and present resulting documents in a decreasing degree of match.
- **Speech Input and Output.** Limited use of speech as an input query and speech as an answer to a question of a request.
  - i.e. inquiries for telephone directory, flight arrival/departure, and credit card account information.
  - The speech input is detected using a library of predefined words and used to set up the parameters that are supplied to the queries.
  - For output, a similar conversion from text or numbers into speech takes place.

DBMS Interfaces

- **Interfaces for Parametric Users:** Parametric users, such as bank tellers.
  - repetitive transactions such as account deposits or withdrawals, or balance inquiries.
- **Interfaces for the DBA:** privileged commands that can be used only by the DBA staff.
  - commands for creating accounts
  - setting system parameters
  - granting account authorization
  - changing a schema
  - reorganizing the storage structures of a database.

Database System Utilities

- Help the DBA to manage the database system.
- Common utilities:
  - **Loading:**
    - Used to load existing data file i.e. text files or sequential files.
    - Current (source) format of the data file and the desired (target) database file structure are specified to the utility, which then automatically reformats the data and stores it in the database.
  - **Backup:**
    - Creates a backup copy of the database. , usually by dumping the entire database onto tape or other mass storage medium.
    - **Incremental backups:** Only changes since the previous backup are recorded

Database System Utilities

- **Database storage reorganization:**
  - Reorganize a set of database files into different file organizations and create new access paths to improve performance.
- **Performance Monitoring:**
  - Monitors database usage and provides statistics to the DBA.
  - DBA uses the statistics in making decisions
  - i.e. whether or not to reorganize files or whether to add or drop indexes to improve performance.
- **Other utilities:** Sorting files, handling data compression, monitoring access by users, interfacing with the network, and performing other functions.

Classification Database Management Systems

- Several criteria can be used to classify DBMSs.
  1. **Data model**
     - **Relational Data Model(SQL systems):** commercial DBMSs
     - **Big Data Systems/ Key-value Storage Systems/ Nosql Systems**
       - Document-based
       - Graph-based
       - Column-based
     - **Hierarchical Data Models:** Legacy Data Models
     - **Network Data Models**: Legacy Data Models
     - **Tree-structured Data Model:** XML based

Classification Database Management Systems

2. **Number of users supported by the system:**
   - **Single-user systems:** support only one user at a time.
   - **Multiuser systems.** Multiple users at a time. Majority of databases are multiuser systems.

3. **Number of sites over which the database is distributed.**
   - **Centralized:** data is stored at a single computer site; can support multiple users.
   - **Distributed DBMS (DDBMS):** have the actual database and DBMS software distributed over many sites connected by a computer network. i.e. Big Data Systems

Classification Database Management Systems

- **Distributed DBMSs:**
  - **Homogeneous DDBMSs:** same DBMS software at all the sites
  - **Heterogeneous DDBMSs:** use different DBMS software at each site.

4. **The fourth criterion is cost.**
   - Additional cost for accessing additional features. i.e. data warehousing, data mining, maintenance etc.
   - Examples of free relational DBMSs: MySQL, PostgreSQL, others

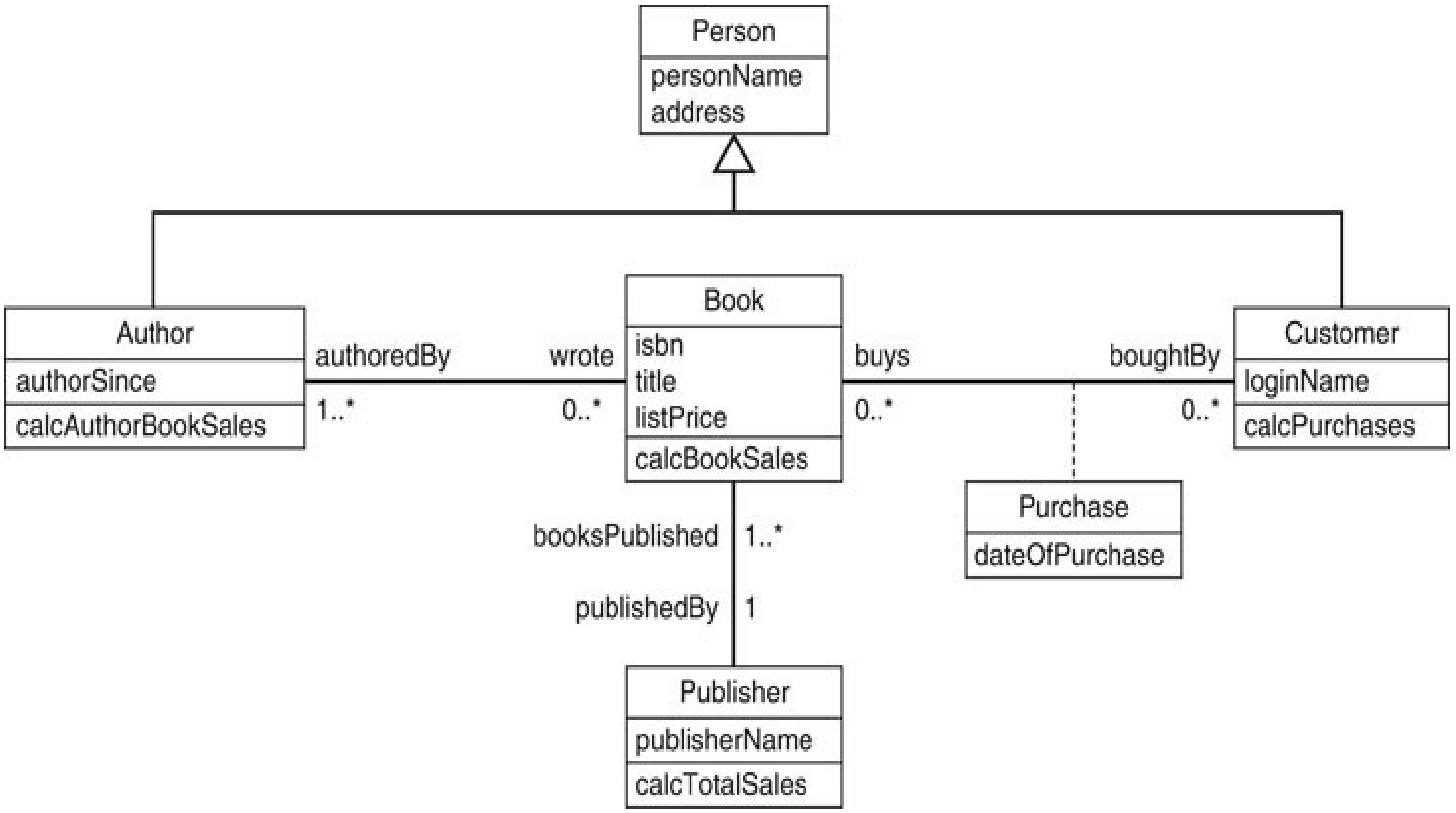
Classification Database Management Systems

- General purpose DBMSs:
- Special purpose DBMSs:
    - Use for specific application.
    - When performance is a primary consideration.
    - Also called online transaction processing (OLTP) systems.
    - i.e. Airline reservation.

Criteria 1 continued

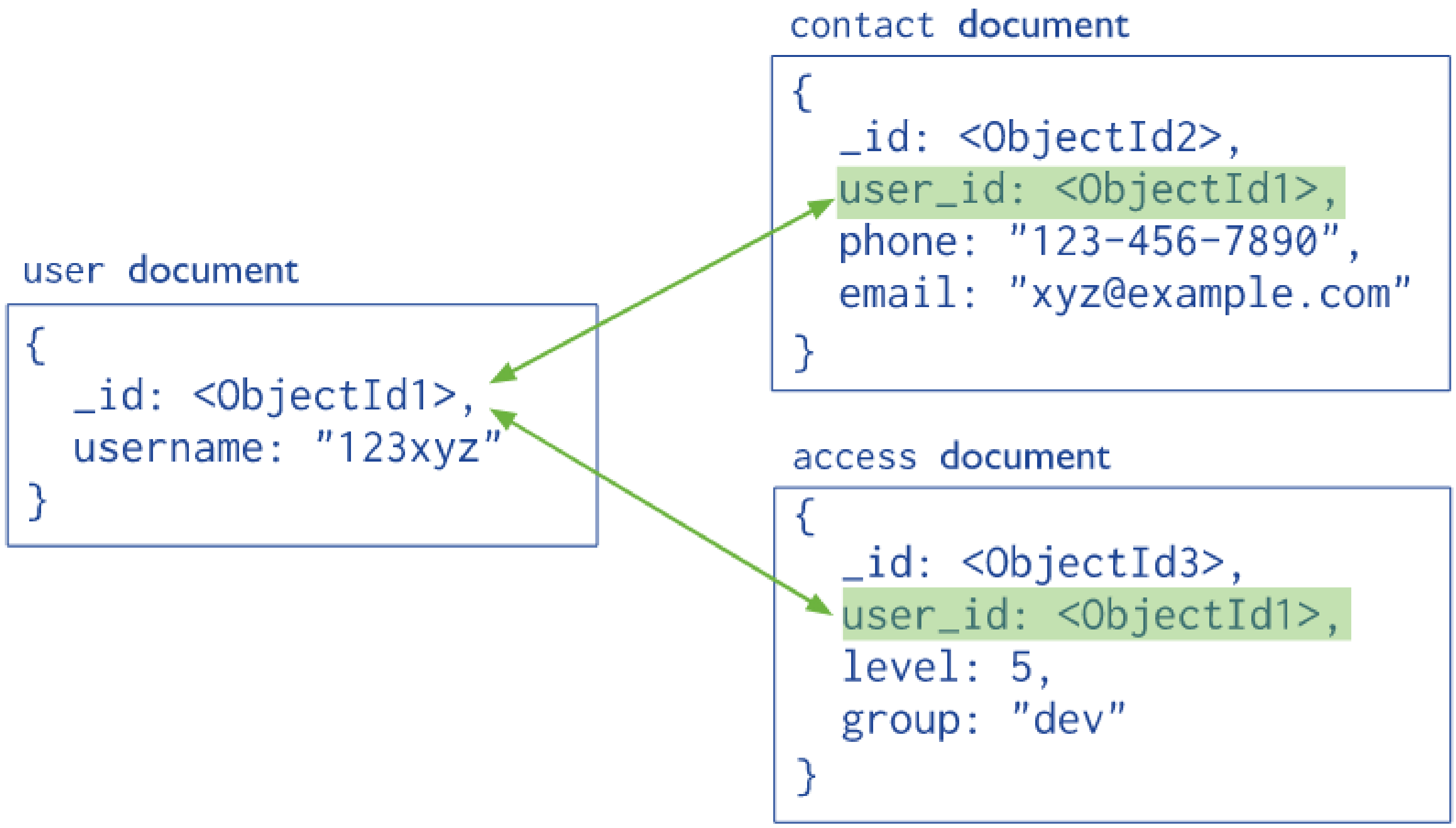
- **Relational data model**
  - Database as a collection of tables
  - Each table can be stored as a separate file
  - Uses SQL and support a limited form of user views
- **Object data model**
  - defines a database in terms of objects, their properties, and their operations.
  - Objects with the same structure and behavior belong to a class
  - Classes are organized into hierarchies (or acyclic graphs).
  - Operations of each class are specified in terms of predefined procedures called methods

Object Data Model



Criteria 1 continued

- **Big data systems:**
  - **Key-Value:**
    - Associates a unique key with each value
    - Provides very fast access to a value given its key.
    - Firebase from Google
  - **Document Data Model**
    - Stores the data as documents, which somewhat resemble complex objects.
    - Based on json (java script object notation)
    - i.e. MongoDB
  - **Graph Data Model**
    - Stores objects as graph nodes and relationships among objects as directed graph edges.
    - i.e. Neo4j
  - **Column-based Data Model**s
    - Store the columns of rows clustered on disk pages for fast access
    - Allow multiple versions of the data.
    - i.e. Hbase



contact document

```
{
  _id: <ObjectId2>,
  user_id: <ObjectId1>,
  phone: "123-456-7890",
  email: "xyz@example.com"
}
```

user document

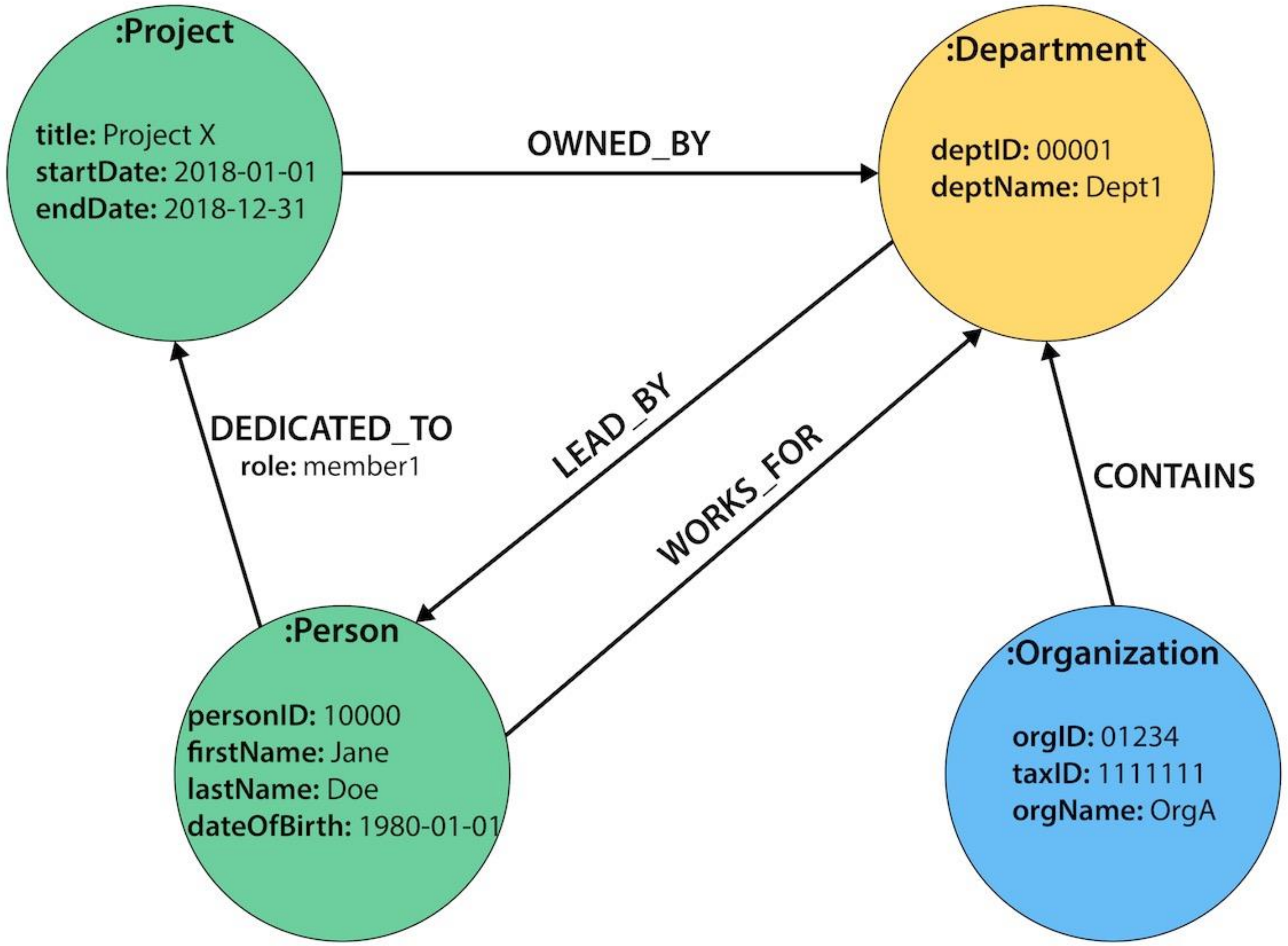
```
{
  _id: <ObjectId1>,
  username: "123xyz"
}
```

access document

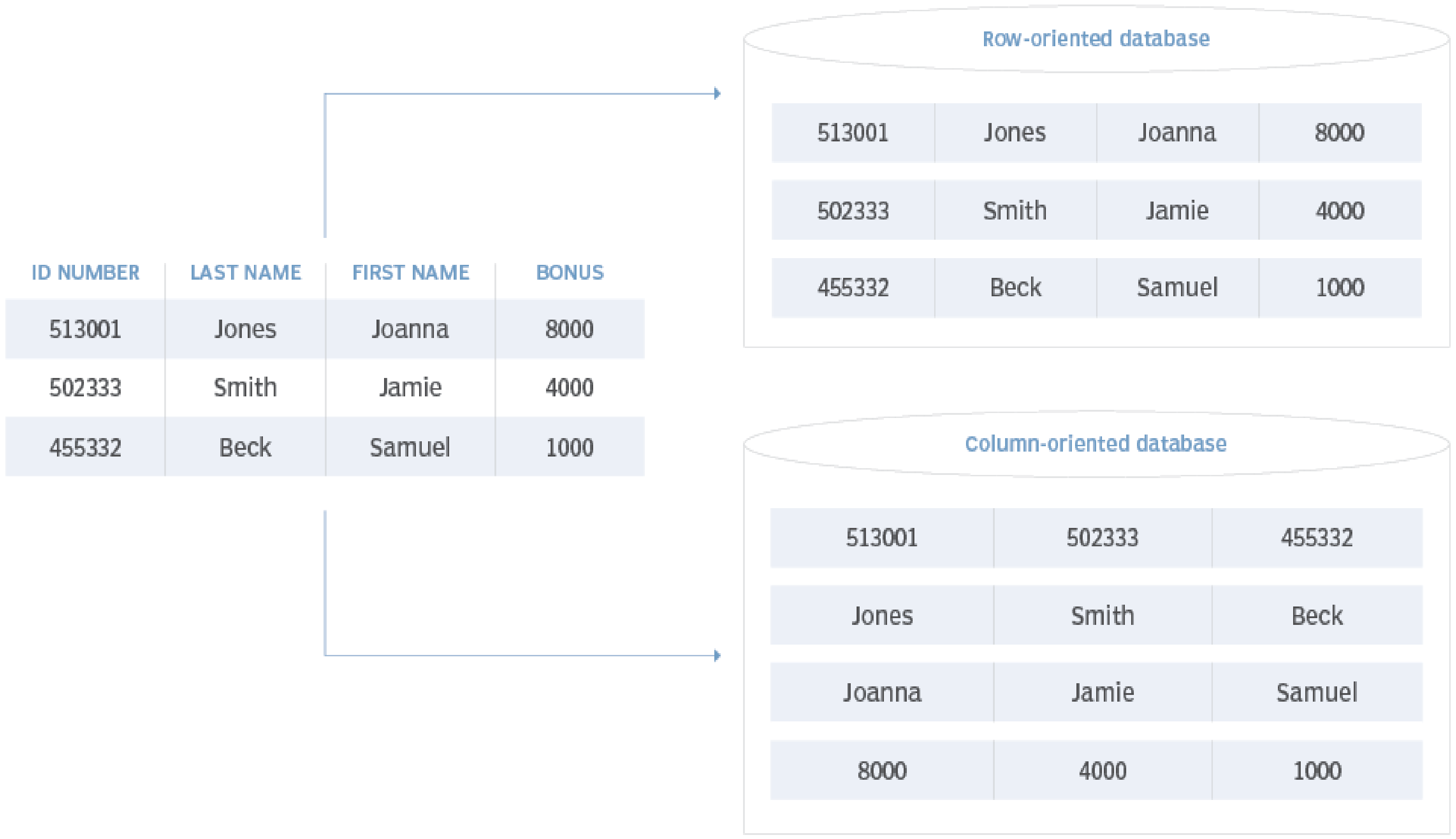
```
{
  _id: <ObjectId3>,
  user_id: <ObjectId1>,
  level: 5,
  group: "dev"
}
```

Document Model

Graph data model



Column Based Data Models



ID NUMBER	LAST NAME	FIRST NAME	BONUS
513001	Jones	Joanna	8000
502333	Smith	Jamie	4000
455332	Beck	Samuel	1000

Row-oriented database

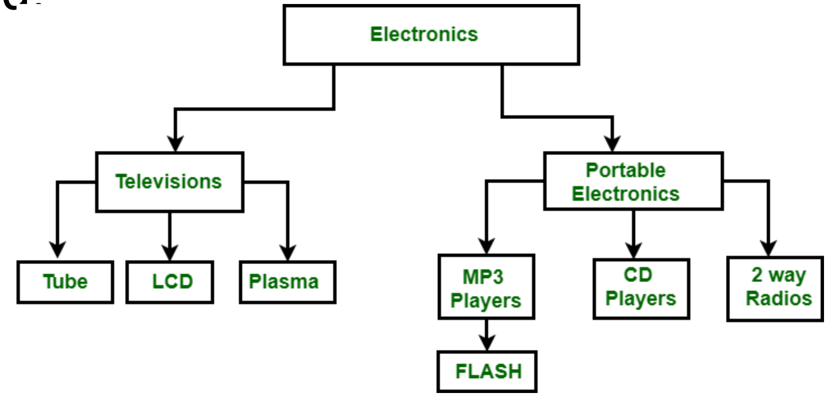
513001	Jones	Joanna	8000
502333	Smith	Jamie	4000
455332	Beck	Samuel	1000

Column-oriented database

513001	502333	455332
Jones	Smith	Beck
Joanna	Jamie	Samuel
8000	4000	1000

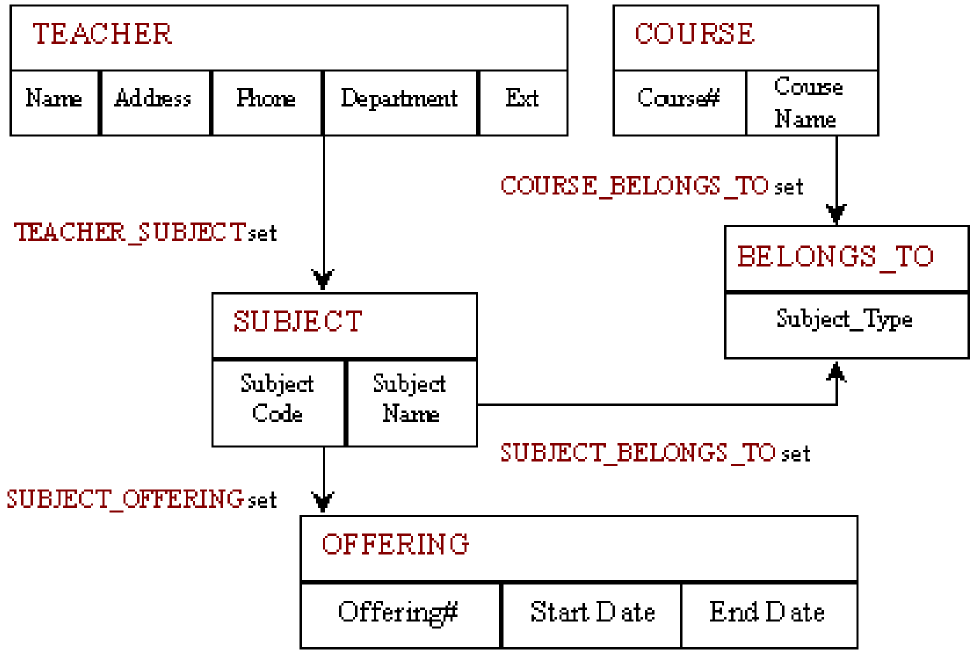
Legacy Data Models

- **Hierarchical Data Model :**
- It organizes data in the tree-like structure.
- Hierarchical model consists of the following :
  - It contains nodes which are connected by branches.
  - The topmost node is called the root node.
  - If there are multiple nodes appear at the top level, then these can be called as root segments.
  - Each node has exactly one parent.
  - One parent may have many child.



Legacy Data Models

- **Network Data Model :**
  - It is the advance version of the hierarchical data model.
  - In this child can have more than one parent.



Tools, Application Environments, and Communications Facilities

- **CASE tools:** used in the design phase of database systems.
- **Data dictionary(information repository):** storing catalog information about schemas and constraints, design decisions, usage standards, application program descriptions, and user information
- **Application development environment:**
  - PowerBuilder (Sybase) or JBuilder (Borland) including database design, GUI development, querying and updating, and application program development.
- **Communications Facilities:**
  - Provided using data communications hardware such as Internet routers, phone lines, long-haul networks, local networks, or satellite communication devices.