Data Models, Architecture, and Languages

CS157A
Chris Pollett
Aug. 29, 2005.
Outline

• Data Models, Schema, and Instances
• Three Schema Architecture
• Database Languages and Interfaces
Data Models

• A **data model** is a collections of concepts used to describe the structure of a database. That is, the basic data types, relationships, and constraints that hold on the data.

• A data model usual also comes with a set of **basic operations** that can be applied to the data. Like storage, retrieval, update.

• Additionally, it might say things about **dynamic aspects** of the database and about what **user-defined operations** possible.
Types of Data Models

- **Conceptual Data Models** - a high level model close to how the user will perceive the data. (Entity Relationship model. Might have notion of entity, attribute, and relationship)

- **Physical Data Models** - a low-level model which provides details about how the data is actually stored.

- **Representational Data Models** - mid-level models which expose the end user to some of the internals. (Relational Data Model, Network Data Model, and Hierarchal Data Model).

- **Object Data Models** - an OO model.
Schemas, Instances, and DB State

- In any data model need to distinguish between the description of the database and the database itself.
- The former is called the **database schema** and the latter is called the **database instance**.
- Sometimes say: instance is the **extension** of the schema; the schema is the **intention** of instance.
- As the database changes with time the **database instance** changes, but the schema usually stays the same. (Can change but usually more slowly, called **schema evolution**.)
- The current instance is called the **database state**.
Data associated with this schema would be the instance. Before first started with some first **population** of data, the database is in its **initial state**. This state evolves to the **current state**.
Three-Schema Architecture

- **External Level**
  - External View
  - External/conceptual mapping

- **Conceptual Level**
  - Conceptual/in-ternal mapping
  - Conceptual Schema

- **Internal Level**
  - Internal Schema
  - Stored Database

**End Users**

Diagram shows the relationships between the levels and the mapping processes.
Data Independence

• One motivation for the three schema architecture is in terms of data independence.
• Data independence is the capacity to change one level without having to change the next higher level. This split into:
  – Logical data independence - can change the conceptual schema without needing to change existing external views (just need to change the mapping)
  – Physical data independence - can change the way the data is stored without having to change the conceptual schema (just need to change the mapping)
After designing the database, the next step is to implement it according to the three schema architecture.

Usually, one has a data definition language (DDL) is used to specify the three levels.

Sometimes this language though is split into three languages: the storage definition language (SDL), the DDL, and the view definition language (VDL).

To add and modify data in the database a data manipulation language (DML) is used. There are high (set at a time) and low level (record at a time) variants of DMLs. One can also have a query language which is used to retrieve data from the database.
DBMS Interfaces

• Menu and Web-Based Interfaces
• Forms-Based Interfaces
• Graphical User Interfaces
• Natural Language Interfaces
• Interfaces for Parametric Users
• Interfaces for DBAs.