

Databases Illuminated

Chapter 2

Database Planning and Database Architecture

Data as a Resource

- Resource: an asset that has value and incurs cost
- Resources include capital equipment, financial assets, personnel **and data**
- **Database** is a resource because
 - Operational data has value
 - Database incurs cost
 - Professionally managed by DBA

Characteristics of Data

- Data vs. information
 - Data: raw facts
 - Example: printout of tables as they are stored
 - Information: processed data, useful for decision-making
 - Example: formatted report using database

Four Levels of Data

1. Real world
 - **Enterprise** in its environment
 - **Mini-world**, or **Universe of Discourse** – part of the world that is represented in the database
2. Conceptual Model
 - Entities, entity sets, attributes, relationships
3. Logical model of database
 - **Metadata**, data about data
 - Record types, data item types, data aggregates
 - Stored in data dictionary
4. Data occurrences
 - Database itself
 - Data instances
 - files

Data Sublanguages

- Every DBMS uses a data sublanguage, which has two parts
 - **Data definition language (DDL)** - used to define the database
 - **Data manipulation language (DML)** - is used to process the database
 - Data sublanguage may be embedded in a general programming language (such as Java, C, C++, C#, COBOL, and so on) which is called the **host language**

Staged Database Design

- **Systems analysis** approach to design assumes every system has a **lifecycle**, and will eventually be replaced
- **Staged database design** centers on first developing a **conceptual model** that evolves and survives

Characteristics of a Conceptual Database Model

- Faithfully mirrors the operations of the organization
- Flexible enough to allow changes as new information needs arise
- Supports many different user views
- Independent of physical implementation
- Does not depend on the model used by a particular database management system

Stages in Database Design

- Analyze user environment
- Develop conceptual data model
- Choose a DBMS
- Develop logical model, by mapping conceptual model to DBMS
- Develop physical model
- Evaluate physical model
- Perform tuning, if indicated
- Implement physical model

See **Figure 2.3** – note loops

Design Tools

- **CASE** (Computer-Aided Software Engineering) tools
 - **Upper case**: used for collecting and designing data, designing logical model, designing applications
 - **Lower case**: used for implementing the database, including prototyping, data conversion, generating application code, generating reports, testing
- Data dictionary
- Project management software

Data Dictionary

- Contains **metadata**
- Can be **integrated** (part of DBMS) or **free-standing**
- Useful for
 - Collecting information about data in central location
 - Securing agreement on meanings of items
 - Communicating with users
 - Identifying inconsistencies – synonyms and homonyms
 - Keeping track of changes to DB structure
 - Determining impact of changes to DB structure
 - Identifying sources of/responsibility for items
 - Recording external/logical/physical models & mappings
 - Recording access control information
 - Providing audit information

Project Management Software

- Tools to help plan and manage projects, especially those with many people
- Include several types of charts and graphs
 - GANTT chart- See **Figure 2.12**
 - PERT chart
- User specifies
 - Scope and objectives
 - Major tasks and phases
 - Task dependencies
 - Resources, including personnel
 - Timelines
- Software can
 - Generate calendars
 - Produce graphs with different views of project
 - Provide means of communication for staff

Database Administrator Skills

- DBA must be
 - Technically competent
 - Good manager
 - Have excellent interpersonal and communication skills
- Has primary responsibility for planning, designing, developing and managing the operating database
- Database designer may do conceptual and logical design; DBA does physical design, implementation, develops, manages system

Planning and Design Stage

- Preliminary planning
- Identifying user requirements
- Developing and maintaining the data dictionary
- Designing the conceptual model
- Choosing a DBMS
- Developing the logical model
- Developing the physical model

Development Phase

- Creating and loading the database
- Developing user views
- Writing and maintaining documentation
- Developing and enforcing data standards
- Developing and enforcing application program standards
- Developing operating procedures
- Doing user training

Database Management Phase

- Monitoring performance
- Tuning and reorganizing
- Keeping current on database improvements

Three-level Database Architecture

- CODASYL DBTG and ANSI/X3/SPARC reports proposed viewing database architecture at 3 levels of abstraction – **external, logical, internal** - each with a written description called a **schema**
- Rationale for separation of external and internal levels
 - Different users need different views of same data
 - Users data needs may change over time
 - Hides complexity of database storage structures
 - Can change logical structure without affecting all users
 - Can change data and file structures without affecting overall logical structure or users' views
 - Database structure unaffected by changes to the physical aspects of storage
- See **Figure 2.5**

External Level

- Consists of many user models or **views**
- Has external records - records seen by users
- May include calculated or virtual data
- Described in external schemas (sub-schemas)
- Used to create user interface

Logical Level

- Entire information structure of database
- “community view” as seen by DBA
- Collection of logical records
- All entities, attributes, relationships represented
- Includes all record types, data item types, relationships, constraints, semantic information, security and integrity information
- Relatively constant over time
- Described in logical schema
- Used to create logical record interface

Internal Level

- Physical implementation level
- Includes data structures, file organizations used by DBMS
- Depends on what DBMS is used
- Described in internal schema
- Used to create stored record interface with operating system
- Operating system creates physical files and physical record interface, below DB

Data Independence

- Logical data independence
 - Immunity of external models to changes in the logical model
 - Occurs at user interface level
- Physical data independence
 - Immunity of logical model to changes in internal model
 - Occurs at logical interface level

Data Models

- Collection of tools for describing structure of database
- Often includes a type of diagram and specialized vocabulary
- Description of the data, relationships in data, constraints on data, some data meanings
- Most permanent part in database architecture
- corresponds to conceptual level or logical level
- **Intension** or scheme of the database
- May change with schema evolution

Entity-Relationship Model

- A semantic model, captures meanings
- Conceptual level model
- Proposed by P.P. Chen in 1970s
- **Entities** are real-world objects about which we collect data
- **Attributes** describe the entities
- **Relationships** are associations among entities
- **Entity set** – set of entities of the same type
- **Relationship set** – set of relationships of same type
- Relationships sets may have descriptive attributes
- Represented by E-R diagrams

Relational Model

- Record-based model
- Logical-level model
- Proposed by E.F. Codd
- Based on mathematical relations
- Uses **relations**, represented as tables
- Columns of tables represent attributes
- Tables represent relationships as well as entities
- Successor to earlier record-based models—**network** and **hierarchical**

Object-oriented Model

- Similar to E-R but includes **encapsulation, inheritance**
- Objects have both **state** and **behavior**
- State is defined by **attributes**
- Behavior is defined by **methods** (functions or procedures)
- Designer defines **classes** with attributes, methods, and relationships
- Class constructor method creates object instances
- Each object has a unique object ID
- Classes related by class hierarchies
- Database objects have **persistence**
- Both conceptual-level and logical-level model

Object-relational model

- Adds new complex datatypes to relational model
- Adds objects with attributes and methods
- Adds inheritance
- SQL extended to handle objects in SQL:1999

Semi-structured Model

- Collection of nodes, each with data, and with different schemas
- Each node contains a description of its own contents
- Can be used for integrating existing databases
- **XML tags** added to documents to describe structure
- XML tags identify elements, sub-elements, attributes in documents
- **XML DTD** (Document Type Definition) or **XML Schema** used to define structure