Changing Data into Information

- Data are the raw facts that are stored in databases.
- Raw facts are seldom immediately useful to a decision maker.
- What the decision maker really needs is information, which is defined as data processed and presented in a meaningful form.

### Table 6.1: Simple Tabulation: Transforming Data into Information

<table>
<thead>
<tr>
<th></th>
<th>Male</th>
<th>Female</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales</td>
<td>19</td>
<td>40</td>
<td>59</td>
</tr>
<tr>
<td>Data</td>
<td>0.81</td>
<td>1.17</td>
<td>2.05</td>
</tr>
<tr>
<td>Sales</td>
<td>3.41</td>
<td>5.05</td>
<td>8.46</td>
</tr>
<tr>
<td>Total</td>
<td>5.28</td>
<td>6.15</td>
<td>11.43</td>
</tr>
</tbody>
</table>

Table 6.1: Simple Tabulation: Transforming Data into Information

The Information System

- A database is a carefully designed and constructed repository of facts and is part of a larger whole known as an information system.
- An IS provides for data collection, storage, and retrieval.
- IS also facilitates the transformation of data into information and the management of both data and information.
- Components of an information system:
  - People
  - Hardware
  - Software
  - Database(s)
  - Application programs
  - Procedures

The Systems Development Life Cycle

- The Systems Development Life Cycle (SDLC) traces the history (life cycle) of an IS.
- Database design takes place within the confines of an IS.
- Five phases of SDLC: (Figure 6.2)
  - Planning
  - Analysis
  - Detailed Systems Design
  - Implementation
  - Maintenance

Generating Information for Decision Making

Figure 6.1

- System analysis is the process that establishes the need for and the extent of an IS.
- The process of creating an IS is known as systems development.
- Applications transform data into the information.
- An application is composed of two parts: the data and the code.
- The performance of an IS depends on three factors:
  - Database design and implementation (DB development)
  - Applications design and implementation
  - Administrative procedures
The Systems Development Life Cycle

**Planning**
- The planning phase yields a general overview of the company and its objectives.
- An initial assessment of the information-flow-and-extend requirements must be made:
  - Should the existing system be continued?
  - Should the existing system be modified?
  - Should the existing system be replaced?
- A feasibility study must address the following issues if a new system is necessary:
  - Technical aspects of hardware and software requirements.
  - The system cost.

**Analysis**
- Problems defined during the planning phase are examined in greater detail:
  - What are the precise requirements of the current system’s end users?
  - Do those requirements fit into the overall information requirements?
  - The analysis phase is a thorough audit of user requirements.
  - The existing hardware and software are studied.
  - End users and system designer(s) work together to identify processes and potential problem areas.

**Detailed Systems Design**
- The designer completes the design of the system’s processes, including all technical specifications for:
  - Screen
  - Menus
  - Reports
  - Other devices
  - Conversion steps are laid out.
  - Training principles and methodologies are planned.

**Implementation**
- The hardware, the DBMS software, and application programs are installed; and the database design is implemented.
- The system enters into a cycle of coding, testing, and debugging.
- The database is created, and the system is customized.
- The database contents are loaded.
- The system is subjected to exhaustive testing.
- The final documentation is reviewed and printed.
- End users are trained.
The Systems Development Life Cycle

- **Maintenance**
  - End users’ requests for changes generate system maintenance activities.
  - Three types of system maintenance:
    - Corrective maintenance in response to systems errors.
    - Adaptive maintenance due to changes in the business environment.
    - Perfective maintenance to enhance the system.
  - Computer-assisted systems engineering (CASE) technology helps make it possible to produce better systems within a reasonable amount of time and cost.

The Database Life Cycle

- **The Database Initial Study**
  - Overall Purpose of the Initial Study:
    - Analyze the company situation.
    - Define problems and constraints.
    - Define objectives.
    - Define scope and boundaries.

- **The Database Life Cycle**
  - Analyze the Company Situation
    - What is the organization’s general operating environment, and what is its mission within that environment?
    - What is the organization’s structure?
  - Define Problems and Constraints
    - How does the existing system function?
    - What input does the system require?
    - What documents does the system generate?
    - How is the system output used? By Whom?
    - What are the operational relationships among business units?
    - What are the limits and constraints imposed on the system?
The Database Life Cycle

- Conceptual Design
  - Data modeling is used to create an abstract database structure that represents real-world objects.
  - The design must be software- and hardware-independent.
  - Minimal data rule: All that is needed is there, and all that is there is needed.
  - Four Steps:
    - Data analysis and requirements
    - Entity relationship modeling and normalization
    - Data model verification
    - Distributed database design

- Data analysis and requirements
  - Designer's efforts are focused on
    - Information needs.
    - Information users.
    - Information sources.
    - Information constitution.
  - Sources of information for the designer
    - Developing and gathering end user data views
    - Direct observation of the current system: existing and desired output
    - Interface with the systems design group
  - The designer must identify the company's business rules and analyze their impacts.

The Database Life Cycle

- Entity Relationship Modeling and Normalization

<table>
<thead>
<tr>
<th>Step</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify entities and write the business case.</td>
</tr>
<tr>
<td>2</td>
<td>Identify the main entities based on Step 1.</td>
</tr>
<tr>
<td>3</td>
<td>Define the relationships between the entities, based on Step 1 and 2.</td>
</tr>
<tr>
<td>4</td>
<td>Modify the entity model to the business case.</td>
</tr>
<tr>
<td>5</td>
<td>Complete the E-R diagram.</td>
</tr>
<tr>
<td>6</td>
<td>Test the model and verify the model with the data, information, and processing requirements.</td>
</tr>
<tr>
<td>7</td>
<td>Modify the E-R diagram, based on the results of Step 5.</td>
</tr>
</tbody>
</table>

Table 6.2 Developing the Conceptual Model Using E-R Diagrams

A Composite Entity

Figure 6.7
E-R Modeling Is An Iterative Process Based On Many Activities

The Database Life Cycle
- Entity Relationship Modeling and Normalization
  - Define entities, attributes, primary keys, and foreign keys.
  - Make decisions about adding new primary key attributes in order to satisfy end user and/or processing requirements.
  - Make decisions about the treatment of multivalued attributes.
  - Make decisions about adding derived attributes to satisfy processing requirements.

The Database Life Cycle
- Make decisions about the placement of foreign keys in 1:1 relationships.
- Avoid unnecessary ternary relationships.
- Draw the corresponding E-R diagram.
- Normalize the data model.
- Include all the data element definitions in the data dictionary.
- Make decisions about standard naming conventions.

The Database Life Cycle
- Data Model Verification
  - Purposes of close review of entities and attributes
    - The emergence of the attribute details may lead to a revision of the entities themselves.
    - The focus on attribute details can provide clues about the nature of the relationships as they are defined by the primary and foreign keys.
    - To satisfy processing and/or end user requirements, it might be useful to create a new primary key to replace an existing primary key.
    - Unless the entity details are precisely defined, it is difficult to evaluate the extent of the design’s normalization.
The Database Life Cycle

- Data Model Verification
  - Advantages of the Modular Approach
    - The modules can be delegated to design groups, greatly speeding up the development work.
    - The modules simplify the design work.
    - The modules can be prototyped quickly. Implementation and applications programming trouble spots can be identified more readily.
    - Even if the entire system can’t be brought on line quickly, the implementation of one or more modules will demonstrate that progress is being made and that at least part of the system is ready to begin serving the end users.

Table 6.3

<table>
<thead>
<tr>
<th>STEP</th>
<th>ACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify the E-R model's central entity.</td>
</tr>
<tr>
<td>2</td>
<td>Identify each module's transaction requirements: Internal/Update/Insert/Delete/Queries/Reports.</td>
</tr>
<tr>
<td>3</td>
<td>Identify each module's transaction requirements: External/Module interfaces.</td>
</tr>
<tr>
<td>4</td>
<td>Verify all processes against the E-R model.</td>
</tr>
<tr>
<td>5</td>
<td>Make all necessary changes suggested in Step 4.</td>
</tr>
<tr>
<td>6</td>
<td>Repeat Steps 2 through 5 for all modules.</td>
</tr>
</tbody>
</table>

The E-R Model Verification Process

Iterative E-R Model Verification Process

Figure 6.10

The Database Life Cycle

- Distributed Database Design
  - Design portion of a database may reside in different physical locations.
  - If the database process is to be distributed across the system, the designer must also develop the data distribution and allocation strategies for the database.

The Database Life Cycle

Database Software Selection

- Common factors affecting the decision:
  - Cost – Purchase, maintenance, operational, license, installation, training, and conversion costs.
  - DBMS features and tools.
  - Underlying model.
  - Portability – Platforms, systems, and languages.
  - DBMS hardware requirements.
The Database Life Cycle

Logical Design
- Logical design translates the conceptual design into the internal model for a selected DBMS.
- It includes mapping of all objects in the model to the specific constructs used by the selected database software.
- For a relational DBMS, the logical design includes the design of tables, indexes, views, transactions, access authorities, and so on.

Physical Design
- Physical design is the process of selecting the data storage and data access characteristics of the database. It affects not only the location of the data in the storage device(s) but also the performance.
- The storage characteristics are a function of:
  - The types of devices supported by the hardware.
  - The type of data access methods supported by the system.
  - The DBMS.
- Physical design is particularly important in the older hierarchical and network models.
- Relational databases are more insulated from physical layer details than hierarchical and network models.
The Database Life Cycle

- Implementation and Loading
  - Create the database storage group.
  - Create the database within the storage group.
  - Assign the rights to use the database to a database administrator.
  - Create the table space(s) within the database.
  - Create the table(s) within the table space(s).
  - Assign access rights to the table spaces and the tables within specified table spaces.
  - Load the data.

Physical Organization of a DB2 Database Environment

Figure 6.12

Physical Design Issues

- Performance
- Security
  - Physical security
  - Password security
  - Access rights
  - Audit trails
  - Data encryption
  - Diskless workstations
- Backup and Recovery
- Integrity
- Company standards
- Concurrency controls

The Need for Concurrency Control

Table 6.5

<table>
<thead>
<tr>
<th>Condition</th>
<th>Value</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read 100</td>
<td>480</td>
<td>7s</td>
</tr>
<tr>
<td>Read 500</td>
<td>560</td>
<td>73s</td>
</tr>
<tr>
<td>Write 100 – 10</td>
<td>365</td>
<td>73s</td>
</tr>
<tr>
<td>Write 500 – 200</td>
<td>200</td>
<td>74s</td>
</tr>
<tr>
<td>Value stored in database</td>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>
The Database Life Cycle

- **Operation**
  - Once the database has passed the evaluation stage, it is considered to be operational.
  - The beginning of the operational phase invariably starts the process of system evolution.

The Database Life Cycle

- **Maintenance and Evolution**
  - Preventive maintenance
  - Corrective maintenance
  - Adaptive maintenance
  - Assignment and maintenance of access permissions
  - Generation of database access statistics
  - Periodic security audits based on the system-generated statistics
  - Periodic system-usage summaries for internal billing or budgeting purposes.

A Special Note about Database Design Strategies

- **Two Classical Approaches to Database Design:**
  - **Top-down design** starts by identifying the data sets, and then defines the data elements for each of these sets.
  - **Bottom-up design** first identifies the data elements (items), and then groups them together in data sets.

Centralized vs Decentralized Design

- **Centralized design**
  - It is productive when the data component is composed of a relatively small number of objects and procedures.

Centralized vs Decentralized Design

- **Decentralized design**
  - It may be used when the data component of the system has a considerable number of entities and complex relations on which very complex operations are performed. (Figure 6.16)
  - Aggregation problems must be addressed: (Figure 6.17)
    - Synonyms and homonyms.
    - Entity and entity subtypes.
    - Conflicting object definitions.

Top-Down Versus Bottom-Up Design Sequencing

- **Figure 6.14**

Centralized vs Decentralized Design

- **Figure 6.15**

Centralized vs Decentralized Design

- **Figure 6.16**

Centralized vs Decentralized Design

- **Figure 6.17**