The Relational Database Model

ISM 318  
Spring 2000
Dr. Hamid Nemati

Objectives

- Understand the relational model’s logical structure
- Understand components of a relational database and their characteristics and relationships
- Understand basic concepts of table design
- Understand characteristics of good and bad tables

A Logical View of Data

- Relational database model’s structural and data independence enables us to view data logically rather than physically.
- The logical view allows a simpler file concept of data storage.
- The use of logically independent tables is easier to understand.
- Logical simplicity yields simpler and more effective database design methodologies.
A Logical View of Data

- Entities and Attributes
  - An entity is simply a person, place, event, or thing for which we intend to collect data.
  - Examples:
    - University – Students, Faculty Members, Courses
    - Airlines – Pilots, Aircraft, Routes, Suppliers
  - Each entity has certain characteristics known as attributes.
    - Examples:
      - Student – Student Number, Name, GPA, Date of Enrollment, Home Address, Phone Number, Major
      - Aircraft – Aircraft Number, Data of Last Maintenance, Total Hours Flown, Hours Flown since Last Maintenance

A Logical View of Data

- Entities and Attributes
  - A grouping of related entities becomes an entity set.
  - Examples:
    - The STUDENT entity set contains all student entities.
    - The FACULTY entity set contains all faculty entities.
    - The AIRCRAFT entity set contains all aircraft entities.

Tables and Their Characteristics

- contains a group of related entities -- i.e. an entity set.
- A table is also called a relation.
A Logical View of Data

- **Characteristics of a Relational Table**
  - A table is perceived as a two-dimensional structure composed of rows and columns.
  - Each table row (tuple) represents a single entity within the entity set.
  - Each column represents an attribute and each column has a distinct name.
  - Each row/column intersection represents a single data value.
  - Each table must have a primary key that uniquely identifies each row.

- **All values in a column must conform to the same data format.**
- **Each column has a specific range of values known as the attribute domain.**
- **Each row carries information describing one entity occurrence.**
- **The order of the rows and columns is immaterial to the DBMS.**

- **Keys**
  - **Controlled redundancy** (shared common field) makes the relational database work.
  - The primary key of one table appears again as the link (foreign key) in another table.
  - If the foreign key contains either matching values or nulls, the table(s) that make use of such a foreign key are said to exhibit referential integrity.
A Logical View of Data
(The Relational Scheme)

- A vendor supplies one or more products

---

Keys

A key relationships. It is an attribute that determines the values of other attributes within the entity.
The key's role is based on a concept known as determination functional dependence.
The attribute B is functionally dependent determines B.

- key

- A multi-attribute key is known as a .

---

Relational Database Keys

- Superkey: An attribute (or combination of attributes) that uniquely identifies each entity in a table.
- Candidate Key: A minimal superkey. A superkey that does not contain a subset of attributes that is itself a superkey.
- Primary Key: A candidate key selected to uniquely identify all other attribute values in any given row. Cannot contain null entities.
- Secondary Key: An attribute (or combination of attributes) used strictly for data retrieval purposes.
- Foreign Key: An attribute (or combination of attributes) in one table whose values must either match the primary key in another table or be null.
Integrity Rules Revisited

**ENTITY INTEGRITY**
- **Requirement**: All entities are unique and no null entries in a primary key.
- **Purpose**: Guarantees that each entity will have a unique identity.

**REFERENTIAL INTEGRITY**
- **Requirement**: Foreign key must have either a null entry or an entry that matches the primary key value in a table to which it is related.
- **Purpose**: Makes it possible for an attribute NOT to have a corresponding value, but it will be impossible to have an invalid entity. The enforcement of the referential integrity rule makes it impossible to delete a row in one table whose primary key has mandatory matching foreign key values in another table.
- **Example**: A customer might not (yet) have an assigned sales representative (number), but it will be impossible to have an invalid sales representative (number).

---

Relational Database Operators

- The degree of relational completeness can be defined by the extent to which relational algebra is supported.
- Relational algebra defines the theoretical way of manipulating table contents using the eight relational functions: SELECT, PROJECT, JOIN, INTERSECT, UNION, DIFFERENCE, PRODUCT, and DIVIDE.

**UNION** combines all rows from two tables. The two tables must be union compatible.
Relational Database Operators

- **INTERSECT** produces a listing that contains only the rows that appear in both tables. The two tables must be union compatible.

- **DIFFERENCE** yields all rows in one table that are not found in the other table; i.e., it subtracts one table from the other. The tables must be union compatible.

- **PRODUCT** produces a list of all possible pairs of rows from two tables.
The Relational Database Model

Relational Database Operators

- **SELECT** yields values for all attributes found in a table. It yields a horizontal subset of a table.

    ![Select Example](image)

- **PROJECT** produces a list of all values for selected attributes. It yields a vertical subset of a table.

    ![Project Example](image)

- **JOIN** allows us to combine information from two or more tables. JOIN is the real power behind the relational database, allowing the use of independent tables linked by common attributes.

    ![Join Example](image)
Relational Database Operators

- **Natural JOIN** links tables by selecting only the rows with common values in their common attribute(s). It is the result of a three-stage process.
- **A PRODUCT** is performed on two tables.
  - **A SELECT** is performed to yield only the rows for which the common attribute values match.
  - **A PROJECT** is performed to yield a single copy of each attribute, thereby eliminating duplicate column.

---

**DIVIDE** requires the use of one single-column table and one two-column table.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Table 2</th>
<th>Table 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>LOC</td>
<td>CODE</td>
</tr>
<tr>
<td>A</td>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

---

**Relational Database Software Classification**

- Not all relational DBMSs are equal. Some are more relational than others.
- The degree to which DBMS conforms to the fully relational model affects database design and implementation effort.
- It is important that a relational DBMS should at least enforce the integrity rules.
The Data Dictionary and the System Catalog

- **Data dictionary** contains metadata to provide detailed accounting of all tables found within the database.
- **System catalog** is a very detailed system data dictionary. It describes all objects within the database.
  - System catalog is a system-created database whose tables store the database characteristics and contents.
  - System catalog tables can be queried just like any other tables.
  - System catalog automatically produces database documentation.

Relationships within the Relational Database

- **Entity Relationship (E-R) Model** provides a simplified picture of the relationship among entities.
- **E-R Diagram (ERD)** is used to map the E-R model:
  - Rectangles are used to represent entities.
  - Diamonds are used to represent the relationship(s) between the entities.
  - The number 1 is used to represent the “1” side of the relationship.
  - The letter M is used to represent the “many” sides of the relationship.

The Relationship between PAINTER and PAINTING

- A Painter paints many painting, but a painting is painted by only one painter.
The Relational Database Model

Relationships within the Relational Database

- A Course has multiple classes, or sections

![Diagram showing the relationship between COURSE and CLASS]

The Relationship Between COURSE and CLASS

Relationships within the Relational Database

- A student can enroll on a number of classes which may have other students as well.

![Diagram showing the M:N relationship between STUDENT and CLASS]

The M:N Relationship Between STUDENT and CLASS

Changing the M:N Relationship to Two 1:M Relationships

![Diagram showing the revised E-R diagram with two sets of 1:M relationships]

Revise E-R Diagram: Two Sets of 1:M relationships

![Diagram showing the new relationships between STUDENT, ENROLL, and CLASS]

Information Systems Operations Management
The Relational Database Model

Relationships within the Relational Database

The Relational Database Model

The Relational Schema

The Relational Schema for student enrollment Entity Relationship Diagram

Data Redundancy Revisited

- Database designers must reconcile three often contradictory requirements: design elegance, processing speed, and information requirements.
- Sometimes controlled data redundancies are required to serve crucial information purposes as shown in the following INVOICE example:
Data Redundancy Revisited

- In this example, the redundancy is crucial to the system’s success. Copying the product price from the PRODUCT table to the LINE table means that it is possible to maintain the historical accuracy of the transactions.

Indexes

- An index is composed of an index key and a set of pointers.

Basic Data Management

- Data Entry
  
  INSERT INTO <table name> VALUES (attribute 1 value, attribute 2 value, ... etc.);
  
  Examples:
  
  INSERT INTO VENDOR
  
  INSERT INTO PRODUCT
  VALUES('11 QER/31`, ‘Power painter, 15 psi., 3-nozzle’, ‘12/2/96’, 8.5, 109.99, 0.00, 25595);
The Relational Database Model