Chapter 5

The Relational Data Model and Relational Database Constraints
Chapter Outline

- Relational Model Concepts
- Relational Model Constraints and Relational Database Schemas
- Update Operations and Dealing with Constraint Violations
The relational Model of Data is based on the concept of a Relation
A Relation is a mathematical concept based on the ideas of sets
The model was first proposed by Dr. E.F. Codd of IBM Research in 1970 in the following paper:
"A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970
The above paper caused a major revolution in the field of database management and earned Dr. Codd the coveted ACM Turing Award
Informal Definitions

- Informally, a relation looks like a table of values.
- A relation typically contains a set of rows.
- The data elements in each row represent certain facts that correspond to a real-world entity or relationship:
  - In the formal model, rows are called tuples.
- Each column has a column header that gives an indication of the meaning of the data items in that column:
  - In the formal model, the column header is called an attribute name (or just attribute).
Example of a Relation

![Diagram showing relation Student with attributes: Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa]

<table>
<thead>
<tr>
<th>Name</th>
<th>Ssn</th>
<th>Home_phone</th>
<th>Address</th>
<th>Office_phone</th>
<th>Age</th>
<th>Gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benjamin Bayer</td>
<td>305-61-2435</td>
<td>373-1616</td>
<td>2918 Bluebonnet Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.21</td>
</tr>
<tr>
<td>Chung-cha Kim</td>
<td>381-62-1245</td>
<td>375-4409</td>
<td>125 Kirby Road</td>
<td>NULL</td>
<td>18</td>
<td>2.89</td>
</tr>
<tr>
<td>Dick Davidson</td>
<td>422-11-2320</td>
<td>NULL</td>
<td>3452 Elgin Road</td>
<td>749-1253</td>
<td>25</td>
<td>3.53</td>
</tr>
<tr>
<td>Rohan Panchal</td>
<td>489-22-1100</td>
<td>376-9821</td>
<td>265 Lark Lane</td>
<td>749-6492</td>
<td>28</td>
<td>3.93</td>
</tr>
<tr>
<td>Barbara Benson</td>
<td>533-69-1238</td>
<td>839-8461</td>
<td>7384 Fontana Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.25</td>
</tr>
</tbody>
</table>

**Figure 5.1**
The attributes and tuples of a relation STUDENT.
The **Schema** (or description) of a Relation:
- Denoted by \( R(A_1, A_2, \ldots, A_n) \)
- \( R \) is the **name** of the relation
- The **attributes** of the relation are \( A_1, A_2, \ldots, A_n \)

**Example:**
CUSTOMER (Cust-id, Cust-name, Address, Phone#)
- CUSTOMER is the relation name
- Defined over the four attributes: Cust-id, Cust-name, Address, Phone#

Each attribute has a **domain** or a set of valid values.
- For example, the domain of Cust-id is 6 digit numbers.
- The domain may have a data-type and/or format
A **tuple** is an ordered set of values (enclosed in angled brackets ‘< … >’) 

Each value is derived from an appropriate **domain**.

A row in the CUSTOMER relation is a 4-tuple and would consist of four values, for example:

- `<632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">`

This is called a 4-tuple as it has 4 values

A tuple (row) in the CUSTOMER relation.

A relation is a **set** of such tuples (rows)
The relation state is a subset of the Cartesian product of the domains of its attributes
- each domain contains the set of all possible values the attribute can take.

Example: attribute Cust-name is defined over the domain of character strings of maximum length 25
- \( \text{dom(Cust-name)} \) is varchar(25)
Formal Definitions - Summary

- Formally,
  - Given $R(A_1, A_2, \ldots, A_n)$
  - $r(R) \subset \text{dom}(A_1) \times \text{dom}(A_2) \times \ldots \times \text{dom}(A_n)$
  - $R(A_1, A_2, \ldots, A_n)$ is the schema of the relation
  - $R$ is the name of the relation
  - $A_1, A_2, \ldots, A_n$ are the attributes of the relation
  - $r(R)$: a specific state (or "value" or “population”) of relation $R$ – this is a set of tuples (rows)
    - $r(R) = \{t_1, t_2, \ldots, t_n\}$ where each $t_i$ is an n-tuple
    - $t_i = <v_1, v_2, \ldots, v_n>$ where each $v_j$ element-of $\text{dom}(A_j)$
Formal Definitions - Example

- Let R(A1, A2) be a relation schema:
  - Let \( \text{dom}(A1) = \{0,1\} \)
  - Let \( \text{dom}(A2) = \{a,b,c\} \)
- Then: \( \text{dom}(A1) \times \text{dom}(A2) \) is all possible combinations:
  \( \{<0,a>, <0,b>, <0,c>, <1,a>, <1,b>, <1,c> \} \)

- The relation state \( r(R) \subset \text{dom}(A1) \times \text{dom}(A2) \)
- For example: \( r(R) \) could be \( \{<0,a>, <0,b>, <1,c> \} \)
  - this is one possible state (or “population” or “extension”) \( r \) of the relation \( R \), defined over \( A1 \) and \( A2 \).
  - It has three 2-tuples: \( <0,a>, <0,b>, <1,c> \)
### Definition Summary

<table>
<thead>
<tr>
<th>Informal Terms</th>
<th>Formal Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Relation</td>
</tr>
<tr>
<td>Column Header</td>
<td>Attribute</td>
</tr>
<tr>
<td>All possible Column Values</td>
<td>Domain</td>
</tr>
<tr>
<td>Row</td>
<td>Tuple</td>
</tr>
<tr>
<td>Table Definition</td>
<td>Schema of a Relation</td>
</tr>
<tr>
<td>Populated Table</td>
<td>State of the Relation</td>
</tr>
</tbody>
</table>
Example – A relation STUDENT

Table: STUDENT

<table>
<thead>
<tr>
<th>Name</th>
<th>Ssn</th>
<th>Home_phone</th>
<th>Address</th>
<th>Office_phone</th>
<th>Age</th>
<th>Gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benjamin Bayer</td>
<td>305-61-2435</td>
<td>373-1616</td>
<td>2918 Bluebonnet Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.21</td>
</tr>
<tr>
<td>Chung-cha Kim</td>
<td>381-62-1245</td>
<td>375-4409</td>
<td>125 Kirby Road</td>
<td>NULL</td>
<td>18</td>
<td>2.89</td>
</tr>
<tr>
<td>Dick Davidson</td>
<td>422-11-2320</td>
<td>NULL</td>
<td>3452 Elgin Road</td>
<td>749-1253</td>
<td>25</td>
<td>3.53</td>
</tr>
<tr>
<td>Rohan Panchal</td>
<td>489-22-1100</td>
<td>376-9821</td>
<td>265 Lark Lane</td>
<td>749-6492</td>
<td>28</td>
<td>3.93</td>
</tr>
<tr>
<td>Barbara Benson</td>
<td>533-69-1238</td>
<td>839-8461</td>
<td>7384 Fontana Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.25</td>
</tr>
</tbody>
</table>

Figure 5.1

The attributes and tuples of a relation STUDENT.
Characteristics Of Relations

- Ordering of tuples in a relation r(R):
  - The tuples are not considered to be ordered, even though they appear to be in the tabular form.

- Ordering of attributes in a relation schema R (and of values within each tuple):
  - We will consider the attributes in R(A1, A2, ..., An) and the values in t=<v1, v2, ..., vn> to be ordered.
Figure 5.2
The relation STUDENT from Figure 5.1 with a different order of tuples.

<table>
<thead>
<tr>
<th>Name</th>
<th>Ssn</th>
<th>Home_phone</th>
<th>Address</th>
<th>Office_phone</th>
<th>Age</th>
<th>Gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dick Davidson</td>
<td>422-11-2320</td>
<td>NULL</td>
<td>3452 Elgin Road</td>
<td>749-1253</td>
<td>25</td>
<td>3.53</td>
</tr>
<tr>
<td>Barbara Benson</td>
<td>533-69-1238</td>
<td>839-8461</td>
<td>7384 Fontana Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.25</td>
</tr>
<tr>
<td>Rohan Panchal</td>
<td>489-22-1100</td>
<td>376-9821</td>
<td>265 Lark Lane</td>
<td>749-6492</td>
<td>28</td>
<td>3.93</td>
</tr>
<tr>
<td>Chung-cha Kim</td>
<td>381-62-1245</td>
<td>375-4409</td>
<td>125 Kirby Road</td>
<td>NULL</td>
<td>18</td>
<td>2.89</td>
</tr>
<tr>
<td>Benjamin Bayer</td>
<td>305-61-2435</td>
<td>373-1616</td>
<td>2918 Bluebonnet Lane</td>
<td>NULL</td>
<td>19</td>
<td>3.21</td>
</tr>
</tbody>
</table>
Characteristics Of Relations

- **Values in a tuple:**
  - All values are considered atomic (indivisible).
  - Each value in a tuple must be from the domain of the attribute for that column
    - If tuple \( t = <v_1, v_2, \ldots, v_n> \) is a tuple (row) in the relation state \( r \) of \( R(A_1, A_2, \ldots, A_n) \)
    - Then each \( v_i \) must be a value from \( \text{dom}(A_i) \)
  - A special null value is used to represent values that are unknown or inapplicable to certain tuples.
Characteristics Of Relations

- **Notation:**
  - We refer to **component values** of a tuple $t$ by:
    - $t[A_i]$ or $t.A_i$
    - This is the value $v_i$ of attribute $A_i$ for tuple $t$
  - Similarly, $t[A_u, A_v, ..., A_w]$ refers to the subtuple of $t$ containing the values of attributes $A_u, A_v, ..., A_w$, respectively in $t$
Relational Integrity Constraints

- Constraints are **conditions** that must hold on **all** valid relation states.
- There are three **main types** of constraints in the relational model:
  - **Key** constraints
  - **Entity integrity** constraints
  - **Referential integrity** constraints
- Another implicit constraint is the **domain** constraint
  - Every value in a tuple must be from the **domain of its attribute** (or it could be **null**, if allowed for that attribute)
Key Constraints

- **Superkey of R:**
  - Is a set of attributes SK of R with the following condition:
    - No two tuples in any valid relation state r(R) will have the same value for SK
    - That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK]

- **Key of R:**
  - A "minimal" superkey
  - That is, a key is a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey (does not possess the superkey uniqueness property)
Example: Consider the CAR relation schema:

CAR(State, Reg#, SerialNo, Make, Model, Year)

CAR has two keys:
- Key1 = {State, Reg#}
- Key2 = {SerialNo}

Both are also superkeys of CAR

{SerialNo, Make} is a superkey but not a key.

In general:
- Any key is a superkey (but not vice versa)
- Any set of attributes that includes a key is a superkey
- A minimal superkey is also a key
Key Constraints (continued)

- If a relation has several candidate keys, one is chosen arbitrarily to be the primary key.
  - The primary key value is used to uniquely identify each tuple in a relation.
  - The primary key attributes are underlined.
  - General rule: Choose as primary key the smallest of the candidate keys (in terms of size).

- Example: Consider the CAR relation schema:
  - CAR(State, Reg#, SerialNo, Make, Model, Year)
  - We chose SerialNo as the primary key.

- Key constraint statement:
  - Not two tuples in a relation must have the same value for the key attribute(s).
Figure 5.4
The CAR relation, with two candidate keys: License_number and Engine_serial_number.

<table>
<thead>
<tr>
<th>License_number</th>
<th>Engine_serial_number</th>
<th>Make</th>
<th>Model</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texas ABC-739</td>
<td>A69352</td>
<td>Ford</td>
<td>Mustang</td>
<td>02</td>
</tr>
<tr>
<td>Florida TVP-347</td>
<td>B43696</td>
<td>Oldsmobile</td>
<td>Cutlass</td>
<td>05</td>
</tr>
<tr>
<td>New York MPO-22</td>
<td>X83554</td>
<td>Oldsmobile</td>
<td>Delta</td>
<td>01</td>
</tr>
<tr>
<td>California 432-TFY</td>
<td>C43742</td>
<td>Mercedes</td>
<td>190-D</td>
<td>99</td>
</tr>
<tr>
<td>California RSK-629</td>
<td>Y82935</td>
<td>Toyota</td>
<td>Camry</td>
<td>04</td>
</tr>
<tr>
<td>Texas RSK-629</td>
<td>U028365</td>
<td>Jaguar</td>
<td>XJS</td>
<td>04</td>
</tr>
</tbody>
</table>
Relational Database Schema

- A set $S$ of relation schemas that belong to the same database.
- $S$ is the name of the whole database schema
- $S = \{R_1, R_2, \ldots, R_n\}$
- $R_1, R_2, \ldots, R_n$ are the names of the individual relation schemas within the database $S$
### COMPANY Database Schema

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fname</td>
<td>Minit</td>
<td>Lname</td>
<td>Ssn</td>
<td>Bdate</td>
<td>Address</td>
<td>Sex</td>
<td>Salary</td>
<td>Super_ssn</td>
<td>Dno</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| DEPARTMENT        |                         |                         |                         |                         |                         |                         |                         |                         |                         |
| Dname             | Dnumber                 | Mgr_ssn                 | Mgr_start_date          |                         |                         |                         |                         |                         |                         |
|                   |                         |                         |                         |                         |                         |                         |                         |                         |                         |

| DEPT_LOCATIONS    |                         |                         |                         |                         |                         |                         |                         |                         |                         |
| Dnumber           | Dlocation               |                         |                         |                         |                         |                         |                         |                         |                         |
|                   |                         |                         |                         |                         |                         |                         |                         |                         |                         |

| PROJECT           |                         |                         |                         |                         |                         |                         |                         |                         |                         |
| Pname             | Pnumber                 | Plocation               | Dnum                    |                         |                         |                         |                         |                         |                         |
|                   |                         |                         |                         |                         |                         |                         |                         |                         |                         |

| WORKS_ON          |                         |                         |                         |                         |                         |                         |                         |                         |                         |
| Essn              | Pno                     | Hours                   |                         |                         |                         |                         |                         |                         |                         |
|                   |                         |                         |                         |                         |                         |                         |                         |                         |                         |

| DEPENDENT         |                         |                         |                         |                         |                         |                         |                         |                         |                         |
| Essn              | Dependent_name          | Sex                     | Bdate                   | Relationship            |                         |                         |                         |                         |                         |
|                   |                         |                         |                         |                         |                         |                         |                         |                         |                         |

---

Figure 5.5
Schema diagram for the COMPANY relational database schema.
Entity Integrity

- The *primary key attributes* PK of each relation schema R in S cannot have null values in any tuple of r(R).
  - This is because primary key values are used to *identify* the individual tuples.
  - t[PK] ≠ null for any tuple t in r(R)
  - If PK has several attributes, null is not allowed in any of these attributes
- Note: Other attributes of R may be constrained to disallow null values, even though they are not members of the primary key.
Referential Integrity

- A constraint used to specify a **relationship** among tuples in two relations:
  - The **referencing relation** and the **referenced relation**.
- Tuples in the **referencing relation** R1 have attributes FK (called **foreign key** attributes) that reference the primary key attributes PK of the **referenced relation** R2.
  - A tuple t1 in R1 is said to **reference** a tuple t2 in R2 if t1[FK] = t2[PK].
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R1.FK to R2.
Referential Integrity (or foreign key) Constraint

- Statement of the constraint
  - The value in the foreign key column (or columns) FK of the referencing relation R1 can be either:
    - (1) a value of an existing primary key value of a corresponding primary key PK in the referenced relation R2, or
    - (2) a null (if it is not part of its own primary key)
Displaying a relational database schema and its constraints

- Each relation schema can be displayed as a row of attribute names
- The name of the relation is written above the attribute names
- The primary key attribute (or attributes) will be underlined
- A foreign key (referential integrity) constraints is displayed as a directed arc (arrow) from the foreign key attributes to the referenced table
  - Can also point to the primary key of the referenced relation for clarity
Figure 5.7
Referential integrity constraints displayed on the COMPANY relational database schema.
Other Types of Constraints

- Semantic Integrity Constraints:
  - based on application semantics and cannot be expressed by the model per se
  - Example: “the max. no. of hours per employee for all projects he or she works on is 56 hrs per week”

- SQL-99 allows triggers and **ASSERTIONS** to express for some of these
Populated database state

- Each *relation* will have many tuples in its current relation state
- The *relational database state* is the union of all the individual relation states
- Whenever the database is changed, a new state arises
- Basic operations for changing the database:
  - INSERT a new tuple in a relation
  - DELETE an existing tuple from a relation
  - MODIFY an attribute of an existing tuple
Figure 5.6
One possible database state for the COMPANY relational database schema.
Update Operations on Relations

- Integrity constraints should not be violated by the update operations: INSERT, DELETE, MODIFY
- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (RESTRICT or REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine
The INSERT operation

- INSERT may violate any of the constraints:
  - Domain constraint:
    - if one of the attribute values provided for the new tuple is not of the specified attribute domain
  - Key constraint:
    - if the value of a key attribute in the new tuple already exists in another tuple in the relation
  - Referential integrity:
    - if a foreign key value in the new tuple references a primary key value that does not exist in the referenced relation
  - Entity integrity:
    - if the primary key value is null in the new tuple
The DELETE operation

- DELETE may violate only referential integrity:
  - If the primary key value of the tuple being deleted is referenced from other tuples in the database
    - Can be remedied by several actions: RESTRICT, CASCADE, SET NULL (see Chapter 8 for more details)
      - RESTRICT option: reject the deletion
      - CASCADE option: delete the tuples that reference the tuple that is being deleted
      - SET NULL option: set the foreign keys of the referencing tuples to NULL
  - One of the above options must be specified during database design for each foreign key constraint
The UPDATE operation

- UPDATE (or MODIFY) may violate domain constraint and NOT NULL constraint on an attribute being modified.
- Any of the other constraints may also be violated, depending on the attribute being updated:
  - Updating the primary key (PK):
    - Similar to a DELETE followed by an INSERT
    - Need to specify similar options to DELETE
  - Updating a foreign key (FK):
    - May violate referential integrity
  - Updating an ordinary attribute (neither PK nor FK):
    - Can only violate domain constraints