THE RELATIONAL DATA MODEL

CHAPTER 3 (6/E)
CHAPTER 5 (5/E)
LECTURE OUTLINE

- Relational Model Concepts
- Relational Database Schemas
- Update Operations
- Brief History of Database Applications (from Section 1.7)
RELATIONAL MODEL CONCEPTS

- Represent data as a collection of relations

**Table** of values

- Each row (*tuple*)
  - Represents a record of related data values
  - Facts that typically correspond to a real-world entity or relationship
- Each column (*attribute*)
  - Holds a corresponding value for each row
  - Slot for a specific interpretation for a row
- **Schema** describes table
  - Table name, attribute names and types
- **Instance** denotes the current contents of the table
  - The *relation* (or *relation state*)

---

**Figure 3.1**
The attributes and tuples of a relation STUDENT.
DOMAINS

- **Domain** $D$
  - Set of atomic values
    - $\{0,1,2,\ldots\}$
    - $\{$Jo Smith, Dana Jones, Ashley Wong, Y. K. Lee,$\ldots\}$

- **Atomic**
  - Each value indivisible

- Domain specified by **Data type** rather than by enumeration
  - Integer, String, Date, Real, etc.
  - Can be specified by format: (ddd)ddd-dddd
SCHEMAS AND ATTRIBUTES

- Relation schema
  - A relation name $R$ and a list of attributes: $A_1, A_2, \ldots, A_n$
  - Denoted by $R(A_1, A_2, \ldots, A_n)$

- Attribute $A_i$
  - Name of a role in the relation schema $R$
  - Associated with a domain $\text{dom}(A_i)$
  - Attribute names do not repeat within relation schema, but domains can repeat.

- Degree (or arity) of a relation
  - Number of attributes $n$ in its relation schema
Relation (or relation state)

- Set of $n$-tuples $r = \{t_1, t_2, ..., t_m\}$
  - Unordered, no duplicates
- Each $n$-tuple $t$
  - Ordered list of $n$ values $t = <v_1, v_2, ..., v_n>$
  - Each value $v_i$, $1 \leq i \leq n$, is an element of $\text{dom}(A_i)$
- Instance of relation schema $R(A_1, A_2, A_3, ..., A_n)$
- Finite subset of the Cartesian product of the domains defining $R$:
  - $\text{rel}(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times ... \times \text{dom}(A_n))$

Because of updates, relations are time-varying

- $\text{rel}(R)$ is relation state at a given time
- Reflects only (and all) the valid tuples that represent a particular state of the real world
RELATIONAL MODEL NOTATION

- Symbolic notation
  - Uppercase letters $Q$, $R$, $S$ denote relation names
  - Corresponding lowercase letters $q$, $r$, $s$ denote corresponding relation states
  - Uppercase letters $A$, $B$, $C$, …, $H$ denote attributes
    - Attribute $A$ can be qualified with the relation name $R$ to which it belongs using the dot notation, e.g., $R.A$
  - Lower case letters $t$, $u$, $v$ denote tuples
**ALTERNATIVE DEFN OF RELATION**

- Tuple considered as a function from attributes to values
  - \( t_j : \{A_1, A_2, A_3, \ldots, A_n\} \rightarrow \text{dom}(A_1) \cup \text{dom}(A_2) \cup \ldots \cup \text{dom}(A_n) \)
  - Use notation \( t_j[A_i] \) or \( t_j.A_i \) to refer to tuple’s value \( v_i \) from \( \text{dom}(A_i) \)
  - Similarly \( t_j[A_u, A_w, \ldots, A_z] \) and \( t_j.(A_u, A_w, \ldots, A_z) \) refer to the subtuple of values \(<v_u, v_w, \ldots, v_z>\) from \( t_j \) for attributes \( A_u, A_w, \ldots, A_z \)
- Therefore, tuple is a set of \(<\text{attribute}, \text{value}>\) pairs
  
  e.g., for *attendee* \((id, givenName, surname, company, dateOfBirth)\)
  - \( t = <10483, \text{John}, \text{Doe}, \text{IBM}, 1978-11-05> \)
  - \( t[id] = 10483 \), \( t[\text{givenName}] = \text{John} \), \( t[\text{surname}] = \text{Doe} \), \( t[\text{company}] = \text{IBM} \), \( t[\text{dateOfBirth}] = 1978-11-05 \)
  - \( t.id = 10483 \), \( t.\text{givenName} = \text{John} \), \( t.\text{surname} = \text{Doe} \), \( t.\text{company} = \text{IBM} \), \( t.\text{dateOfBirth} = 1978-11-05 \)
  - \( t = \{<\text{id}, 10483>, <\text{givenName}, \text{John}>, <\text{surname}, \text{Doe}>, <\text{company}, \text{IBM}>, <\text{dateOfBirth}, 1978-11-05> \} \)
VALUES IN TUPLES

- Each value in a tuple is atomic
  - **Flat** (as opposed to *nested*) relational model
  - Composite and multivalued attributes not allowed
  - Historically relation is said to be **in First normal form** (1NF)

- Composite attributes
  - Split into simple component attributes
  - e.g., *Waterloo, Ontario* treated as atomic or split into two attributes to store *Waterloo* separately from *Ontario*

- Multivalued attributes
  - Must be represented by separate relations
  - Recall: *Director* could be stored as attribute of FILM because only one director per film assumed, but multiple characters in a film implies that ROLE must have its own relation.
NULL VALUES

- Assume each domain is augmented with a special NULL value
  - Represent the values of attributes that may be unknown or may not apply to a tuple
- Interpretations for NULL values
  - *Nothing is known about the value*
  - *Value exists but is (currently) not available*
  - *Value undefined*
    - i.e., attribute does not apply to this tuple
- If an attribute for a tuple is mapped to NULL, cannot make any assumptions about the value for that attribute (for that tuple)
  - e.g., Ashley’s telephone number is NULL could mean
    - Ashley doesn’t have a phone
    - Ashley has a phone but we don’t know the number (perhaps withheld)
    - Ashley has a phone that has no number
    - Ashley may or may not have a phone, but regardless we don’t have a number for Ashley
MEANING OF A RELATION

- **Assertion**
  - Each tuple in the relation interpreted as a **fact**.
  - No other similar facts are of interest to the enterprise.
  - e.g., a relation for Classlist includes only registered students and all registered students are included in Classlist
    - presence in list $\iff$ registered student

- **Predicate**
  - Values in each tuple interpreted as values that satisfy predicate
  - e.g., Name of student having ID 83201556 is Lee Wong
LECTURE SUMMARY

- Characteristics differentiate relations from ordinary tables or files
- Schemas vs. instances (states)
- Formal definitions for relations and tuples
- Null values