IT420: Database Management and Organization

The Relational Model
18 January 2006
Adina Crăiniceanu
www.cs.usna.edu/~adina

Database Design Process

- Requirements analysis
- Conceptual design: Entity-Relationship Model
- Logical design: transform ER model into relational schema
- Schema refinement: Normalization
- Physical tuning

Goals

- Understand:
  - The relational model
  - Relational model terminology
The Relational Model

- Introduced in 1970 by E.F. Codd (IBM)
- Turing Award in 1980
- Now the standard model for commercial DBMS products
  - Vendors: IBM, Oracle, Microsoft, Sybase, etc

Relational Database

- A relational database is a set of relations
- A relation is a two-dimensional table
- Relation schema describes the column heads for the table
  - Relation name
  - Column names
  - Column types

A Relation

```
EMPLOYEE(EmployeeNumber:integer, 
  FirstName:string, 
  LastName:string, 
  Department:string, 
  Email:string, 
  Phone:integer)
```

<table>
<thead>
<tr>
<th>EmployeeNumber</th>
<th>Firstname</th>
<th>Lastname</th>
<th>Department</th>
<th>Email</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Jerry</td>
<td>Johnson</td>
<td>Accounting</td>
<td><a href="mailto:jlj@somewhere.com">jlj@somewhere.com</a></td>
<td>206-9927</td>
</tr>
<tr>
<td>200</td>
<td>Mary</td>
<td>Anthony</td>
<td>Finance</td>
<td><a href="mailto:MA@somewhere.com">MA@somewhere.com</a></td>
<td>444-8888</td>
</tr>
<tr>
<td>300</td>
<td>Liz</td>
<td>Stephens</td>
<td>Finance</td>
<td><a href="mailto:LS@somewhere.com">LS@somewhere.com</a></td>
<td>777-0008</td>
</tr>
<tr>
<td>400</td>
<td>Tom</td>
<td>Cowper</td>
<td>Accounting</td>
<td><a href="mailto:TC@somewhere.com">TC@somewhere.com</a></td>
<td>999-9999</td>
</tr>
<tr>
<td>500</td>
<td>Jim</td>
<td>Jackson</td>
<td>Production</td>
<td><a href="mailto:JJJ@somewhere.com">JJJ@somewhere.com</a></td>
<td>444-8880</td>
</tr>
<tr>
<td>600</td>
<td>Eleanor</td>
<td>Cadick</td>
<td>Legal</td>
<td><a href="mailto:EC@somewhere.com">EC@somewhere.com</a></td>
<td>387-0000</td>
</tr>
<tr>
<td>700</td>
<td>Richard</td>
<td>Bascilene</td>
<td>Legal</td>
<td><a href="mailto:RB@somewhere.com">RB@somewhere.com</a></td>
<td>387-6660</td>
</tr>
</tbody>
</table>
Relation

- All entries in a column are of the same kind
- Each column has a unique name
- Cells of the table hold a single value
- The order of the columns is not important
- The order of the rows is not important
- No two rows may be identical

- Rows contain data about entity instances
- Columns contain data about attributes of the entity

A Relation with Values of Varying Length

<table>
<thead>
<tr>
<th>Employee ID</th>
<th>First Name</th>
<th>Last Name</th>
<th>Department</th>
<th>Email</th>
<th>Phone</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Jerry</td>
<td>Johnson</td>
<td>Accounting</td>
<td><a href="mailto:j@somewhere.com">j@somewhere.com</a></td>
<td>212-1000</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>Mary</td>
<td>Abrahamy</td>
<td>Finance</td>
<td><a href="mailto:m@somewhere.com">m@somewhere.com</a></td>
<td>444-0000</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>Joe</td>
<td>Stephens</td>
<td>Marketing</td>
<td><a href="mailto:j@somewhere.com">j@somewhere.com</a></td>
<td>777-0000</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>Tom</td>
<td>Cawthers</td>
<td>Accounting</td>
<td><a href="mailto:t@somewhere.com">t@somewhere.com</a></td>
<td>236-9000</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>Tim</td>
<td>Jackson</td>
<td>Production</td>
<td><a href="mailto:t@somewhere.com">t@somewhere.com</a></td>
<td>666-8000</td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>Sheena</td>
<td>Calista</td>
<td>Legal</td>
<td><a href="mailto:s@somewhere.com">s@somewhere.com</a></td>
<td>777-0000</td>
<td></td>
</tr>
<tr>
<td>700</td>
<td>Richard</td>
<td>Babletone</td>
<td>Legal</td>
<td><a href="mailto:r@somewhere.com">r@somewhere.com</a></td>
<td>707-0000</td>
<td></td>
</tr>
</tbody>
</table>

This is a relation with values of varying length, as indicated by the different lengths of the values in the email column.

Tables That Are Not Relations: Multiple Entries per Cell

<table>
<thead>
<tr>
<th>Employee ID</th>
<th>First Name</th>
<th>Last Name</th>
<th>Department</th>
<th>Email</th>
<th>Phone</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Jerry</td>
<td>Johnson</td>
<td>Accounting</td>
<td><a href="mailto:j@somewhere.com">j@somewhere.com</a></td>
<td>212-1000</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>Mary</td>
<td>Abrahamy</td>
<td>Finance</td>
<td><a href="mailto:m@somewhere.com">m@somewhere.com</a></td>
<td>444-0000</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>Joe</td>
<td>Stephens</td>
<td>Marketing</td>
<td><a href="mailto:j@somewhere.com">j@somewhere.com</a></td>
<td>777-0000</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>Tom</td>
<td>Cawthers</td>
<td>Accounting</td>
<td><a href="mailto:t@somewhere.com">t@somewhere.com</a></td>
<td>236-9000</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>Tim</td>
<td>Jackson</td>
<td>Production</td>
<td><a href="mailto:t@somewhere.com">t@somewhere.com</a></td>
<td>666-8000</td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>Sheena</td>
<td>Calista</td>
<td>Legal</td>
<td><a href="mailto:s@somewhere.com">s@somewhere.com</a></td>
<td>777-0000</td>
<td></td>
</tr>
<tr>
<td>700</td>
<td>Richard</td>
<td>Babletone</td>
<td>Legal</td>
<td><a href="mailto:r@somewhere.com">r@somewhere.com</a></td>
<td>707-0000</td>
<td></td>
</tr>
</tbody>
</table>

These are tables that are not relations, as indicated by the multiple entries per cell.
Tables That Are Not Relations: Table with Required Row Order

<table>
<thead>
<tr>
<th>EmployeeNumber</th>
<th>Firstname</th>
<th>Lastname</th>
<th>Department</th>
<th>Email</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>Jan</td>
<td>Johnson</td>
<td>Accounting</td>
<td><a href="mailto:Jan@somewhere.com">Jan@somewhere.com</a></td>
<td>288-6999</td>
</tr>
<tr>
<td>600</td>
<td>Mary</td>
<td>Anderson</td>
<td>Finance</td>
<td><a href="mailto:Mary@somewhere.com">Mary@somewhere.com</a></td>
<td>888-8989</td>
</tr>
<tr>
<td>400</td>
<td>Tim</td>
<td>Smith</td>
<td>Accounting</td>
<td><a href="mailto:Tim@somewhere.com">Tim@somewhere.com</a></td>
<td>777-0988</td>
</tr>
<tr>
<td>800</td>
<td>Sue</td>
<td>Smith</td>
<td>Finance</td>
<td><a href="mailto:Sue@somewhere.com">Sue@somewhere.com</a></td>
<td>288-9999</td>
</tr>
<tr>
<td>800</td>
<td>Tom</td>
<td>Jackson</td>
<td>Production</td>
<td><a href="mailto:Tom@somewhere.com">Tom@somewhere.com</a></td>
<td>888-9999</td>
</tr>
<tr>
<td>600</td>
<td>Eleanor</td>
<td>Jackson</td>
<td>Legal</td>
<td><a href="mailto:Eleanor@somewhere.com">Eleanor@somewhere.com</a></td>
<td>300-9988</td>
</tr>
<tr>
<td>700</td>
<td>Richard</td>
<td>Williams</td>
<td>Legal</td>
<td><a href="mailto:Richard@somewhere.com">Richard@somewhere.com</a></td>
<td>767-0988</td>
</tr>
</tbody>
</table>

Alternative Terminology

- Although not all tables are relations, the terms table and relation are normally used interchangeably.
- The following sets of terms are equivalent:

<table>
<thead>
<tr>
<th>Table</th>
<th>Column</th>
<th>Row</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relation</td>
<td>Attribute</td>
<td>Tuple</td>
</tr>
<tr>
<td>File</td>
<td>Field</td>
<td>Record</td>
</tr>
</tbody>
</table>

Integrity Constraints (IC)

- IC: condition that must be true for any instance of the database; e.g., domain constraints
- ICs are specified when schema is defined
- ICs are checked when relations are modified
- A legal instance of a relation is one that satisfies all specified ICs
- DBMS should not allow illegal instances
Keys

- A **key** is a combination of one or more columns that is used to identify rows in a relation.
- A **composite key** is a key that consists of two or more columns.

Keys (cont.)

- A set of columns is a **key** for a relation if:
  1. No two distinct rows can have same values in all key columns, and
  2. This is not true for any subset of the key.
- Part 2 false? A **superkey**
- E.g., EmployeeNumber is a key for EMPLOYEE (What about LastName? SSN?)
- The set {EmployeeNumber, Email} is a superkey

Candidate and Primary Keys

- A **candidate key** is a key.
- A **primary key** is a candidate key selected as the primary means of identifying rows in a relation:
  - There is one and only one primary key per relation.
  - The primary key may be a composite key.
  - The ideal primary key is short, numeric and never changes.
Surrogate Keys

- A surrogate key as an artificial column added to a relation to serve as a primary key:
  - DBMS supplied
  - Short, numeric and never changes – an ideal primary key!
  - Has artificial values that are meaningless to users
  - Remember Access

NOTE: The primary key of the relation is underlined below:

- RENTAL_PROPERTY without surrogate key:
  RENTAL_PROPERTY (Street, City, State/Province, Zip/PostalCode, Country, Rental_Rate)

- RENTAL_PROPERTY with surrogate key:
  RENTAL_PROPERTY (PropertyID, Street, City, State/Province, Zip/PostalCode, Country, Rental_Rate)

Foreign Keys

- A foreign key is the primary key of one relation that is placed in another relation to form a link between the relations
  - ‘Refer’ to a row in another relation
  - ‘Logical pointer’

  The term refers to the fact that key values are foreign to the relation in which they appear as foreign key values
Foreign Keys

NOTE: The primary keys of the relations are underlined and any foreign keys are in italics in the relations below:

DEPARTMENT (DepartmentName, BudgetCode, ManagerName)
EMPLOYEE (EmployeeNumber, EmployeeName, DepartmentName)

The Referential Integrity Constraint

- A referential integrity constraint: the values of the foreign key must exist as primary key values in the corresponding relation
- No ‘dangling references’

Foreign Key with a Referential Integrity Constraint

NOTE: The primary key of the relation is underlined and any foreign keys are in italics in the relations below:

DEPARTMENT (DepartmentName, BudgetCode, ManagerName)
EMPLOYEE (EmployeeNumber, EmployeeName, DepartmentName)

Where EMPLOYEE.DepartmentName must exist in DEPARTMENT

- Can you name a data model w/o referential integrity?