Relational Model

- Proposed by Edgar F. Codd in early 1970's
- Data is stored in tables (a.k.a. relations)
- All major database systems these day are relational

<table>
<thead>
<tr>
<th>student_id</th>
<th>first_name</th>
<th>last_name</th>
<th>birthday</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000001</td>
<td>John</td>
<td>Doe</td>
<td>1970-1-1</td>
</tr>
<tr>
<td>2000002</td>
<td>Jane</td>
<td>Doe</td>
<td>1971-1-1</td>
</tr>
<tr>
<td>2000003</td>
<td>Tom</td>
<td>Smith</td>
<td>1962-2-2</td>
</tr>
</tbody>
</table>

Table (Relation)

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>John</td>
</tr>
<tr>
<td>1001</td>
<td>Jane</td>
</tr>
</tbody>
</table>

Records (Rows) (Tuples)

Students

Table and Database Schema

- Table schema
  - Name of the table, and the names and types of the attributes
  - E.g. students(id:integer, name:string)
  - or just students(id, name)

- Database schema
  - Schemas of all the tables in the database

About Relational Model

- Attributes must be of simple type
- No order among attributes
- No order among records

From ER to Relations ...
... From ER to Relations

Students (id, name, address)
Departments (name)
Classes (code, name, quarter, section, department_name)
Enrollment (code, quarter, section, student_id)

Basic Rules of ER to Relational Conversion ...

A entity set is converted to a table
- Entity set name → table name
- Entity set attributes → table columns
- Entity set key → table key

A many-to-many relationship is also converted to a table, including
- Its own attributes (if any)
- Key attributes from the associated entity sets → table key

... Basic Rules of ER to Relational Conversion

A many-to-one relationship is converted to a foreign key column on the “many” side referencing the “one” side

About Keys: Primary Key

Entity set key: an attribute or a combination of several attributes that uniquely identifies an entity
Primary key in relational model: UNIQUE + NOT NULL

About Keys: Foreign Key

Foreign key in relational model
- A link (or association) between two tables – a foreign key column is like an object reference in a Java class
- A data integrity constraint
- There is NO foreign key in ER model, because the association is already expressed as a relationship

Another Common Problem in ER Design
More Conversion Examples

Basic ER to Relational Conversion Steps

- Step 1: convert entity sets to tables
- Step 2: convert relationships
  - Many-to-many → table
  - Many-to-one → foreign key column
- Step 3: rename tables and columns when necessary

Special Cases of Conversion

- One-to-One relationship
- Multiway relationship
- Weak entity set
- Subclass

Converting One-to-One Relationship ...

... Converting One-to-One Relationship

- Which one of the following is better??

  Faculty( id, name, chair_of_department )
  Departments( id, name )
  or
  Faculty( id, name )
  Departments( id, name, department_chair )

Converting Multiway Relationship

- Should this relationship be treated as many-to-many or many-to-one??
Converting Weak Entity Set ...

- The table for a weak entity set includes its complete key as well as its own non-key attributes.
- A supporting relationship is redundant and yields no relation.

... Converting Weak Entity Set

- Converting Subclass ...

- Object-oriented approach
  - One table per concrete class
  - Each entity belongs to exact one table
- ER approach
  - One table per subclass
  - Each entity may appear in multiple tables
- NULL approach
  - One table per class hierarchy

Object-Oriented Approach

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>John</td>
</tr>
</tbody>
</table>

Users

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>cin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>Jane</td>
<td>212345678</td>
</tr>
</tbody>
</table>

Students

ER Approach

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>John</td>
</tr>
<tr>
<td>1001</td>
<td>Jane</td>
</tr>
</tbody>
</table>

Users

<table>
<thead>
<tr>
<th>user_id</th>
<th>cin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>212345678</td>
</tr>
</tbody>
</table>
NULL Approach ...

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>cin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>John</td>
<td>NULL</td>
</tr>
<tr>
<td>1001</td>
<td>Jane</td>
<td>212345678</td>
</tr>
</tbody>
</table>

Users

... NULL Approach

<table>
<thead>
<tr>
<th>id</th>
<th>user_type</th>
<th>name</th>
<th>cin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>staff</td>
<td>John</td>
<td>NULL</td>
</tr>
<tr>
<td>1001</td>
<td>student</td>
<td>Jane</td>
<td>212345677</td>
</tr>
</tbody>
</table>

Users

Comparison of Subclass Conversion Approaches

- Constraints and data integrity
- Query performance

Q1: list all students
Q2: list all non-student users
Q3: list all users

Example 1: Restaurant

<table>
<thead>
<tr>
<th>#42</th>
<th>Some Restaurant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date: Jul 06, 2008 02:57PM</td>
<td></td>
</tr>
<tr>
<td>Server: John</td>
<td></td>
</tr>
<tr>
<td># of Guest: 2</td>
<td></td>
</tr>
<tr>
<td>Table: 42</td>
<td></td>
</tr>
<tr>
<td>1 Boiled Pork Wanton</td>
<td>4.95</td>
</tr>
<tr>
<td>1 Dumpling w/ Crabmeat</td>
<td>8.00</td>
</tr>
<tr>
<td>1 Beef Noodle Soup</td>
<td>6.80</td>
</tr>
<tr>
<td>Subtotal: 19.75</td>
<td></td>
</tr>
<tr>
<td>GST: 0.99</td>
<td></td>
</tr>
<tr>
<td>Total: 20.74</td>
<td></td>
</tr>
<tr>
<td>Open Time: Jul 06, 2008 02:57PM</td>
<td></td>
</tr>
<tr>
<td>Printed by: Cashier</td>
<td></td>
</tr>
</tbody>
</table>

Example 2: Folders and Files

```
C:\
|-- WINNT
|   |-- Document and Settings
|   |   |-- Program Files
|   |     |-- vsun
|   |       |-- \database material
|   |           |-- \web material
|           |-- file1
|           |   |-- file2
|           |       |-- file3
|           |           |-- file4
```

Example 3: Price Changes

What if we want to model price changes?

$ Price of a product X

time