CSCI-GA.2433-001
Database Systems

Lecture 3: From ER to Relational Model

Mohamed Zahran (aka Z)
mzahran@cs.nyu.edu
http://www.mzahran.com
Why Relational Model?

• Very simple yet powerful
• By far the dominant data model
• A database is a collection of one or more relations
  – Each relation is a table (rows and columns)
  – As you can see, you can’t get any simpler!!
A Relation

• Simply speaking: a table!

• Relation schema:
  – What is the relation name?
  – What are the fields (aka columns, or attributes)?
  – What are their types (domain name -> values)?

• Relation instance:
  – The contents of the table
  – Set of records (aka tuples)
A Relation

- No two rows are identical.
- Order of rows is not important.
- Order of fields is not important if mentioned by name.
- Important if names not specified and field mentioned by position.
- DB systems use combination of the above.

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>login</th>
<th>age</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>jones@cs</td>
<td>18</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>smith@eecs</td>
<td>18</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>smith@math</td>
<td>19</td>
<td>3.8</td>
</tr>
</tbody>
</table>
### A Relation

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>login</th>
<th>age</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>jones@cs</td>
<td>18</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>smith@eecs</td>
<td>18</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>smith@math</td>
<td>19</td>
<td>3.8</td>
</tr>
</tbody>
</table>

**Number of fields called** degree or arity

**Cardinality is** number of tuples

**Relation instance** = relation instance that satisfies the **domain constraints** in the relation scheme
A Relational DB

• Collection of relations with distinct relation names
• Its schema is the collection of schemas of the relations in the DB.
• An instance is the collection of relation instances
SQL
Structured Query Language

Data Definition Language
• Creating a table
• Inserting records
• Deleting records/tables

Table in SQL = Relation

Data Manipulation Language
Creating and Modifying Relations in SQL

CREATE TABLE Students ( sid CHAR(20),
                        name CHAR(30),
                        login CHAR(20),
                        age INTEGER,
                        gpa REAL)

Relation name
Field names
Domain name for each field
Creating and Modifying Relations in SQL

```
CREATE TABLE Students ( sid  CHAR(20) ,
                        name  CHAR(30) ,
                        login CHAR(20) ,
                        age   INTEGER, 
                        gpa   REAL )

INSERT INTO Students (sid, name, login, age, gpa)
VALUES (53688, 'Smith', 'smith@ee', 18, 3.2)
```
Creating and Modifying Relations in SQL

CREATE TABLE Students ( sid CHAR(20),
                        name CHAR(30),
                        login CHAR(20),
                        age INTEGER,
                        gpa REAL )

DELETE
FROM Students S
WHERE S.name = 'Smith'
Creating and Modifying Relations in SQL

CREATE TABLE Students ( sid CHAR(20),
                        name CHAR(30),
                        login CHAR(20),
                        age INTEGER,
                        gpa REAL )

UPDATE Students S
SET S.age = S.age + 1, S.gpa = S.gpa - 1
WHERE S.sid = 53688

Is applied first
A DB is as good as the data stored in it.

IC are:
- Conditions specified in the DB schema
- Restrict the data that can be stored in an instance of the DB

DBMS enforces IC by permitting only legal instances to be stored in the DB.
Key Constraints

• Certain **minimal** subset of the fields of a relation is a **unique identifier** for a tuple (of all legal instances) → **candidate key**
  – No two distinct tuples in legal instance have identical values in all fields of the key.
  – No subset of fields in a key is unique identifier

• **Superkey** = set of fields that contains a key

• **A relation may have several candidate keys**
Key Constraints

CREATE TABLE Students ( sid CHAR(20),
     name CHAR(30),
     login CHAR(20),
     age INTEGER,
     gpa REAL,
     UNIQUE (name, age),
     CONSTRAINT StudentsKey PRIMARY KEY (sid) )

It is a good practice to give name to the constraint. So that when there is violation, we know which one.

The combination is unique. Not each field alone
Foreign Key Constraints

- The foreign key must match the primary key.
- Linking information from one relation to another
- If one is modified, the other must be checked and then may be updated.
- Foreign key need not be the primary key of its relation.
- Foreign key could refer to the same relation.
Foreign Key Constraints

- The foreign key must match the primary key.
- Linking information from one relation to another
- If one is modified, the other must be checked and then may be updated.
- Foreign key need not be the primary key of its relation.
- Foreign key could refer to the same relation.
CREATE TABLE Enrolled ( studid CHAR(20),
cid CHAR(20),
grade CHAR(10),
PRIMARY KEY (studid, cid),
FOREIGN KEY (studid) REFERENCES Students)
NULL Value

• Used in a field of a tuple to mean: unknown or not applicable
• Null values are not allowed in primary key.
Enforcing Integrity Constraints

Question: Do you think INSERT, DELETE, and UPDATE can cause IC violation?

```
CREATE TABLE Enrolled ( studid CHAR(20),
                       cid    CHAR(20),
                       grade  CHAR(10),
                       PRIMARY KEY (studid, dd),
                       FOREIGN KEY (studid) REFERENCES Students
                       ON DELETE CASCADE
                       ON UPDATE NO ACTION)
```

Actions relative to `referenced` relation

Default
Transactions

• A program that runs against a DB
  – Can contain several statements that access the DB

• When are constraints checked?
  – By default at the end of every SQL statement
  – Sometimes this is inflexible and check must be deferred
Transactions

CREATE TABLE Students (sid CHAR(20),
name CHAR(30),
login CHAR(20),
age INTEGER,
honors CHAR(10) NOT NULL,
gpa REAL)
PRIMARY KEY (sid),
FOREIGN KEY (honors) REFERENCES Courses (cid))

CREATE TABLE Courses (cid CHAR(10),
cname CHAR(10),
credits INTEGER,
grader CHAR(20) NOT NULL,
PRIMARY KEY (dd)
FOREIGN KEY (grader) REFERENCES Students (sid))

How are going to insert the very first course and the very first student?

A constraint in deferred mode is checked at commit time.
Querying Relational Data

```
SELECT  *  
FROM     Students S  
WHERE    S.age < 18

SELECT  S.name, S.login  
FROM     Students S  
WHERE    S.age < 18

SELECT  S.name, E.cid  
FROM     Students S, Enrolled E  
WHERE    S.sid = E.studid AND E.grade = 'A'
```
ER Model → Relational Model

- Entity set → table
- Relationship set (without constraints) → table
- Translating relationship set with key constraints
- Translating relationship set with participation constraints
- Translating weak entity sets
- Translating class hierarchies
- Translating aggregation
CREATE TABLE Employees
(ssn CHAR(11),
name CHAR(20),
lot INTEGER,
PRIMARY KEY (ssn))
Relationships Without Constraints → Tables

• Relationship set is mapped to a relation in the relational model

• Attributes include
  – Primary key of each participating entity sets, as foreign keys
  – Descriptive attributes of the relationship set
CREATE TABLE Works_In2 (ssn CHAR(11),
did INTEGER,
address CHAR(20),
since DATE,
PRIMARY KEY (ssn, did, address),
FOREIGN KEY (ssn) REFERENCES Employees,
FOREIGN KEY (address) REFERENCES Locations,
FOREIGN KEY (did) REFERENCES Departments)
CREATE TABLE Reports_To (  
  supervisor...ssn  CHAR(11),  
  subordinate...ssn  CHAR(11),  
  PRIMARY KEY (supervisor...ssn, subordinate...ssn),  
  FOREIGN KEY (supervisor...ssn) REFERENCES Employees(ssn),  
  FOREIGN KEY (subordinate...ssn) REFERENCES Employees(ssn) )
Translating Relationship With Key Constraints

CREATE TABLE Manages (ssn CHAR(11),
                      did INTEGER,
                      since DATE,
                      PRIMARY KEY (did),
                      FOREIGN KEY (ssn) REFERENCES Employees,
                      FOREIGN KEY (did) REFERENCES Departments)

CREATE TABLE DepLMgr (did INTEGER,
                       dname CHAR(20),
                       budget REAL,
                       ssn CHAR(11),
                       since DATE,
                       PRIMARY KEY (did),
                       FOREIGN KEY (ssn) REFERENCES Employees)

Approach 1

Approach 2

Incorporating the relationship into relation “Departments”
Translating Relationship Sets With Participation Constraints

CREATE TABLE Dept_Mgr ( 
  did INTEGER,
  dname CHAR(20),
  budget REAL,
  ssn CHAR(11) NOT NULL,
  since DATE,
  PRIMARY KEY (did),
  FOREIGN KEY (ssn) REFERENCES Employees
  ON DELETE NO ACTION)
Translating Weak Entity Sets

CREATE TABLE Dep_Policy (pname CHAR(20),
                         age INTEGER,
                         cost REAL,
                         ssn CHAR(11),
                         PRIMARY KEY (pname, ssn),
                         FOREIGN KEY (ssn) REFERENCES Employees
                         ON DELETE CASCADE)
Approach 1 (General and always applicable):
• 3 relations: Employees, Hourly_Emps and Contract_Emps.
• Every employee is recorded in Employees.
• For Hourly emps, extra info recorded in Hourly_Emps (same for Contract_Emps) (hourly_wages, hours_worked, ssn);
• Must delete Hourly_Emps tuple if referenced Employees tuple is deleted.
• Queries involving all employees is easy.
• Those involving just Hourly_Emps require a join to get some attributes.
Translating Class Hierarchies

**Approach 1:**
- Create only two relations: **Hourly_Emps** and **Contract_Emps**.
- They contain their own attributes as well as **Employees**.
- Cannot always be applied if, say, an employee is neither hourly or contract.
Views

• Definition: A view is a table whose rows are not explicitly stored in the DB
• Views are computed, on demand, from a view definition.
• Can provide logical data independence and security

CREATE VIEW B-Students (name, sid, course) AS SELECT S.sname, S.sid, E.cid FROM Students S, Enrolled E WHERE S.sid = E.studid AND E.grade = 'B'
What Will You do?

"I would like my customers to be able to browse my catalog of books and place orders over the Internet. Currently, I take orders over the phone. I have mostly corporate customers who call me and give me the ISBN number of a book and a quantity; they often pay by credit card. I then prepare a shipment that contains the books they ordered. If I don't have enough copies in stock, I order additional copies and delay the shipment until the new copies arrive; I want to ship a customer's entire order together. My catalog includes all the books I sell. For each book, the catalog contains its ISBN number, title, author, purchase price, sales price, and the year the book was published. Most of my customers are regulars, and I have records with their names and addresses. New customers have to call me first and establish an account before they can use my website. On my new website, customers should first identify themselves by their unique customer identification number. Then they should be able to browse my catalog and to place orders online."
What Will You do?

"I would like my customers to be able to browse my catalog of books and place orders over the Internet. Currently, I take orders over the phone. I have mostly corporate customers who call me and give me the ISBN number of a book and a quantity; they often pay by credit card. I then prepare a shipment that contains the books they ordered. If I don't have enough copies in stock, I order additional copies and delay the shipment until the new copies arrive; I want to ship a customer's entire order together. My catalog includes all the books I sell. For each book, the catalog contains its ISBN number, title, author, purchase price, sales price, and the year the book was published. Most of my customers are regulars, and I have records with their names and addresses. New customers have to call me first and establish an account before they can use my website. On my new website, customers should first identify themselves by their unique customer identification number. Then they should be able to browse my catalog and to place orders online."
CREATE TABLE Books ( isbn CHAR(10), 
title CHAR(80),
author CHAR(80),
qty_in_stock INTEGER,
price REAL,
year_published INTEGER,
PRIMARY KEY (isbn))

CREATE TABLE Orders ( isbn CHAR(10),
cid INTEGER,
carenum CHAR(16),
qty INTEGER,
order_date DATE,
ship_date DATE,
PRIMARY KEY (isbn,cid),
FOREIGN KEY (isbn) REFERENCES Books,
FOREIGN KEY (cid) REFERENCES Customers)

CREATE TABLE Customers ( cid INTEGER,
cname CHAR(80),
address CHAR(200),
PRIMARY KEY (cid)
Conclusions

• ER model needs to be translated to a more formal model
  – semantic data model → data model
• The most widely used data model: relational model
• ER model → Relational model is approximate