The Relational Model

Why Study the Relational Model?

- Most widely used model.
  - Vendors: IBM, Informix, Microsoft, Oracle, Sybase, etc.
- “Legacy systems” in older models
  - E.G., IBM’s IMS
- Recent competitor: object-oriented model
  - ObjectStore, Versant, Ontos, O2
  - A synthesis emerging: object-relational model
    - Informix Universal Server, UniSQL, Oracle, DB2

Relational Database: Definitions

- Relational database: a set of relations
- Relation: made up of 2 parts:
  - Instance: a table, with rows and columns.
  - Schema: specifies name of relation, plus name and type of each column.
    - E.G. Students(sid: string, name: string, login: string, age: integer, gpa: real)
- Can think of a relation as a set of rows or tuples (i.e., all rows are distinct).

Example Instance of Students 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>

- Cardinality = 3, degree = 5, all rows distinct
- Do all columns in a relation instance have to be distinct?

Creating Relations in SQL

- Creates the Students relation. Observe that the type (domain) of each field is specified, and enforced by the DBMS whenever tuples are added or modified.
  
  CREATE TABLE Students
  (sid: CHAR(20),
   name: CHAR(20),
   login: CHAR(10),
   age: INTEGER,
   gpa: REAL)

- As another example, the Enrolled table holds information about courses that students take.
  
  CREATE TABLE Enrolled
  (sid: CHAR(20),
   cid: CHAR(20),
   grade: CHAR(2))

Integrity Constraints (ICs)

- 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.
- If the DBMS checks ICs, stored data is more faithful to real-world meaning.
  - Avoids many data entry errors, too!
Primary Key Constraints

- A set of fields is a *superkey* for a relation if:
  - No two distinct tuples have the same values in all fields of the superkey
- A superkey is a *candidate* key if:
  - No proper subset of it is a superkey
- If there’s >1 candidate key for a relation, one of the keys is chosen (by DBA) to be the *primary* key.
- E.g., sid is a key for Students. (What about name?) The set {sid, gpa} is a superkey.

Primary and Candidate Keys in SQL

- Possibly many candidate keys (specified using UNIQUE), one of which is chosen as the primary key.
- For a given student and course, there is a single grade." vs. "Students can take only one course, and receive a single grade for that course; further, no two students in a course receive the same grade."
- Used carelessly, an IC can prevent the storage of database instances that arise in practice!

Foreign Keys, Referential Integrity

- **Foreign key**: Set of fields in one relation that is used to ‘refer’ to a tuple in another (or the same) relation. (Must correspond to primary key of the second relation.) Like a ‘logical pointer’.
- E.g. sid is a foreign key referring to Students:
  - Enrolled(sid: string, cid: string, grade: string)
  - If all foreign key constraints are enforced, referential integrity is achieved, i.e., no dangling references.
  - Can you name a data model w/o referential integrity?
- Links in HTML!

Foreign Keys in SQL

- Only students listed in the Students relation should be allowed to enroll for courses.
  - CREATE TABLE Enrolled
    - (sid CHAR(20), cid CHAR(20), grade CHAR(2), PRIMARY KEY (sid,cid), FOREIGN KEY (sid) REFERENCES Students )

Enforcing Referential Integrity

- Consider Students and Enrolled; sid in Enrolled is a foreign key that references Students.
- What should be done if an Enrolled tuple with a non-existent student id is inserted? (Reject it!)
- What should be done if a Students tuple is deleted?
  - Also delete all Enrolled tuples that refer to it.
  - Disallow deletion of a Students tuple that is referred to.
  - Set sid in Enrolled tuples that refer to it to a default sid.
  - (In SQL, also: Set sid in Enrolled tuples that refer to it to a special placeholder null, meaning ‘unknown’ or ‘inapplicable’)
- Similar if primary key of Students tuple is updated.

Referential Integrity in SQL/92

- SQL/92 supports all 4 options on deletes and updates.
  - Default is NO ACTION (delete/update is rejected)
  - CASCADE (also delete all tuples that refer to deleted tuple)
  - SET NULL / SET DEFAULT (sets foreign key value of referencing tuple)

Links in HTML!
Where do ICs Come From?

- ICs are based upon the semantics of the real-world enterprise that is being described in the database relations.
- We can check a database instance to see if an IC is violated, but we can NEVER infer that an IC is true by looking at an instance.
  - An IC is a statement about all possible instances!
  - From example, we know name is not a key, but the assertion that sid is a key is given to us.
- Key and foreign key ICs are the most common; more general ICs supported too.

Logical DB Design: ER to Relational

- Entity sets to tables.

CREATE TABLE Employees
  (ssn CHAR(11),
   name CHAR(20),
   lot INTEGER,
   PRIMARY KEY (ssn))

CREATE TABLE Works_In
  (ssn CHAR(1),
   did INTEGER,
   since DATE,
   PRIMARY KEY (ssn, did),
   FOREIGN KEY (ssn) REFERENCES Employees,
   FOREIGN KEY (did) REFERENCES Departments)

CREATE TABLE Manages
  (ssn CHAR(11),
   did INTEGER,
   since DATE,
   PRIMARY KEY (did),
   FOREIGN KEY (ssn) REFERENCES Employees,
   FOREIGN KEY (did) REFERENCES Departments,
   ON DELETE NO ACTION)

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,
   FOREIGN KEY (ssn) REFERENCES Departments,
   ON DELETE CASCADE)

CREATE TABLE Dep_Policy
  (pname CHAR(20),
   age INTEGER,
   cost REAL,
   ssn CHAR(11) NOT NULL,
   PRIMARY KEY (pname, ssn),
   FOREIGN KEY (ssn) REFERENCES Employees,
   FOREIGN KEY (ssn) REFERENCES Employees,
   ON DELETE CASCADE)
Relational Query Languages

- A major strength of the relational model: supports simple, powerful querying of data.
- Queries can be written intuitively, and the DBMS is responsible for efficient evaluation.
  - The key: precise semantics for relational queries.
  - Allows the optimizer to extensively re-order operations, and still ensure that the answer does not change.

The SQL Query Language

- Developed by IBM (system R) in the 1970s
- Need for a standard since it is used by many vendors
- Standards:
  - SQL-86
  - SQL-89 (minor revision)
  - SQL-92 (major revision, current standard)
  - SQL-99 (major extensions)

The SQL Query Language

- To find all 18 year old students, we can write:

```
SELECT *
FROM Students
WHERE age=18
```

- To find just names and logins, replace the first line:

```
SELECT name, login
FROM Students
WHERE age=18
```

Querying Multiple Relations

- What does the following query compute?

```
SELECT S.name, E.cid
FROM Students S, Enrolled E
WHERE S.sid=E.sid AND E.grade="A"
```

Querying Multiple Relations

- Given the following instance of Enrolled (is this possible if the DBMS ensures referential integrity?):

```
we get:
```
```
s.name E.cid
Smith Topology112
```

Adding and Deleting Tuples

- Can insert a single tuple using:

```
INSERT INTO Students (sid, name, login, age, gpa)
VALUES (53688, 'Smith', 'smith@ee', 18, 3.2)
```

- Can delete all tuples satisfying some condition (e.g., name = Smith):

```
DELETE FROM Students
WHERE name = 'Smith'
```

Destroying and Altering Relations

- Destroys the relation Students. The schema information and the tuples are deleted.

```
DROP TABLE Students
```

- The schema of Students is altered by adding a new field; every tuple in the current instance is extended with a null in the new field.

```
ALTER TABLE Students
ADD COLUMN firstYear: integer
```
Relational Model: Summary

- A tabular representation of data.
- Simple and intuitive, currently the most widely used.
- Integrity constraints can be specified by the DBA, based on application semantics. DBMS checks for violations.
  - Two important ICs: primary and foreign keys
  - In addition, we always have domain constraints.
- Guidelines to translate ER to relational model
- Powerful and natural query languages exist.