


The Relational Model

Chapter 3


1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 0



Relational Databases

- Most common data model in modern DMBS
- Many commercial systems
 - Oracle, MS SQL Server, IBM DB2, more...
- Also open source
 - MySQL, PostgreSQL, more...

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 1



Terminology Parade

- **Database:** A set of relations
- **Relation:** Made up of two parts
 - **Instance:** A table, with rows (aka tuples, records), and columns (aka fields, attributes)
 - # Rows = cardinality
 - # Columns = degree / arity
 - **Schema**
 - Relation name
 - Name and **domain** (i.e., type) for each column
 - E.g., Student (sid: integer, name: string, gpa: real)
- **Set semantics:** (classical relational model) *Every row is unique*
- **Multiset semantics:** (modern systems, SQL) *Duplicate rows allowed*

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 2

Instance of Athlete Relation

AID	Name	Country	Sport
1	Mary Lou Retton	USA	Gymnastics
2	Jackie Joyner-Kersey	USA	Track
3	Michael Phelps	USA	Swimming

What is the schema? (aid: integer, name: string, country: string, sport:string)

Cardinality & Degree? Cardinality = 3, Degree = 4

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 3

Relational Query Languages

- Supports simple, powerful *querying* of data
- Queries written *declaratively*
 - In contrast to *procedural* methods
- DBMS is responsible for efficient evaluation
 - System can optimize for efficient query execution, and still ensure that the answer does not change

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 4

SQL: Structured 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)
 - SQL-99 (major extensions, recursive queries, triggers, some OO features, ...)
 - SQL-2003 (XML, window functions, ...)

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 5

Creating Relations in SQL

- Create the Athlete relation
 - Domain constraint (type) enforced when tuples added or modified
- Create the Olympics relation
- Create the Compete relation

```
CREATE TABLE Athlete
(aid INTEGER,
name CHAR(30),
country CHAR(20),
sport CHAR(20))
```

```
CREATE TABLE Olympics
(oid INTEGER,
year INTEGER,
city CHAR(20))
```

```
CREATE TABLE Compete
(aid INTEGER,
oid INTEGER)
```

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 6

Quick Review

- This looks familiar... What are the entity sets and relationship sets?

```

graph TD
    Athlete[Athlete] --- Compete{Compete}
    Olympics[Olympics] --- Compete
    Athlete --- sport((sport))
    Athlete --- aid((aid))
    Athlete --- name((name))
    Athlete --- cntry((cntry))
    Olympics --- oid((oid))
    Olympics --- year((year))
    Olympics --- city((city))
  
```

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 7

The SQL Query Language

- Find all athletes from USA:

```
SELECT *
FROM Athlete A
WHERE A.country = 'USA'
```

AID	Name	Country	Sport
1	Mary Lou Retton	USA	Gymnastics
2	Jackie Joyner-Kersey	USA	Track
3	Michael Phelps	USA	Swimming

- Print only the names and sports:

```
SELECT A.name, A.sport
FROM Athlete A
WHERE A.country = 'USA'
```

Name	Sport
Mary Lou Retton	Gymnastics
Jackie Joyner-Kersey	Track
Michael Phelps	Swimming

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 8

Querying Multiple Relations

- What does the following query compute?

```
SELECT O.year
FROM Athletes A, Olympics O, Compete C
WHERE A.aid = C.aid AND O.oid = C.oid
AND A.name = 'Michael Phelps'
```

Find the years when Michael Phelps competed in the Olympics

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 9

Destroying & Altering Relations

```
DROP TABLE Olympics
```

Destroys the relation Olympics.
(Schema information and tuples are deleted)

```
ALTER TABLE Athlete
ADD COLUMN age: INTEGER
```

Alters Athlete schema by adding a new column
What do we put in the new field?
a null value: 'unknown' or 'inapplicable'

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 10

Adding & Deleting Tuples

- Can insert a single tuple using:

```
INSERT INTO Athlete (aid, name, country, sport)
VALUES (4, 'Johann Koss', 'Norway', 'Speedskating')
```

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

```
DELETE
FROM Athlete A
WHERE A.name = 'Smith'
```

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 11

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.
 - Catches data entry errors, too!

1/12/09

EECS 484: Database Management Systems, Kristen LeFevre

12

Key Constraints

- A *key* for a relation R is the *minimal* set of attributes A_1, \dots, A_n such that no two tuples in (*any instance of*) R can have the same values for A_1, \dots, A_n
- Not minimal? Then a *superkey*
- E.g., {ssn} is a key for Citizen relation; {ssn, name} is a superkey
- A relation can have more than one key; one designated as primary key

1/12/09

EECS 484: Database Management Systems, Kristen LeFevre

13

Primary and Candidate Keys

- Candidate keys specified using *UNIQUE*
- One of which is chosen as the *primary key*.

```
CREATE TABLE Athlete
(aid INTEGER,
 name CHAR(30) NOT NULL,
 country CHAR(20),
 sport CHAR(20),
 UNIQUE (name, country),
 PRIMARY KEY (aid))
```

In English, what restriction does candidate key impose?

For each country, every athlete has a unique name.

WARNING: If used carelessly, ICs can prevent storing instances that arise in practice!

1/12/09

EECS 484: Database Management Systems, Kristen LeFevre

14

Primary and Candidate Keys

- NULLS and keys in SQL:
 - A special value `null`, denotes 'unknown' or 'inapplicable'
 - PRIMARY KEY columns can `never` be null
 - UNIQUE keys are unique only for the non-null entries

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 15

Foreign Keys, Referential Integrity

- *Foreign key*: Set of fields in one relation that is used to refer to a tuple in another relation.
- Must refer to primary key of the second relation.
 - Like a 'logical pointer'.
- E.g., *aid* in Competes relation is a foreign key referring to Athlete
 - If all foreign key constraints are enforced, *referential integrity* (no dangling references) is achieved

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 16

Foreign Keys in SQL

- Only people listed in Athletes relation should be allowed to compete


```
CREATE TABLE Compete
(aid INTEGER, oid INTEGER,
PRIMARY KEY (aid, oid),
FOREIGN KEY (aid) REFERENCES Athlete)
```
- And only in games stored in the Olympics relation...


```
CREATE TABLE Compete
(aid INTEGER, oid INTEGER,
PRIMARY KEY (aid, oid),
FOREIGN KEY (aid) REFERENCES Athlete,
FOREIGN KEY (oid) REFERENCES Olympics)
```

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 17

Enforcing ICs

- Whenever we modify database, must check for violations of ICs
- Enforcing Domain, Primary Key, Unique ICs is straightforward
 - Reject offending UPDATE / INSERT command

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 18

Enforcing Referential Integrity

- Suppose Compete references Athlete (primary key *aid*) via Foreign Key constraint
- What should we do if a Compete tuple is inserted with no corresponding Athlete *aid*?
 - Reject it!
- What if an Athlete tuple is deleted? Possible actions:
 - Disallow deletion if a Compete tuple refers to athlete
 - Delete all Compete tuples that refer to deleted athlete
 - For all Compete tuples referring to the deleted athlete,
 - set *aid* to default
 - set *aid* to null
- Similar choices in primary key of Athlete (*aid*) is updated

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 19

Referential Integrity in SQL

- SQL Supports all four options on deletes and updates
 - Default is **NO ACTION** (*action is rejected*)
 - **CASCADE** (also delete all tuples that refer to deleted tuple)
 - **SET NULL / SET DEFAULT** (sets foreign key value of referencing tuple)

```
CREATE TABLE Compete
(aid INTEGER, oid INTEGER,
PRIMARY KEY (aid, oid),
FOREIGN KEY (aid)
REFERENCES Athlete
ON DELETE CASCADE
ON UPDATE NO ACTION)
```

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 20

Where do ICs Come From?

- Based on real-world enterprise being modeled
- An IC is a statement about *all possible* instances!
- 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.
- Key and foreign key ICs are the most common
- Also table constraints and assertions
 - Next week!

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 21

Views

- **View** is used just like a relation, but we store a *definition*, rather than a set of tuples

```
CREATE VIEW Athens_Olympians (aid, name, country)
AS SELECT A.aid, A.name, A.country
FROM Athlete A, Competes C, Olympics O
WHERE A.aid = C.aid AND C.oid = O.oid
AND O.year = 2004
```

- Logical data independence
- Security
- Views can be dropped using DROP VIEW
- How to drop a table if there is a view on it?
 - DROP TABLE command has options to let user specify this

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 22

Views

What does this query compute?

```
SELECT name
FROM Athens_Olympians
WHERE country = 'USA'
```

Find the names of all athletes from country 'USA' who participated in the 2004 Olympics.

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 23

Updates on Views?

- User perspective: view is like a table
 - Fine for queries, but what about updates?
- Can be tricky to figure out how updates map back to data stored in base tables

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 24

Updates on Views - Example

Athlete			Sport		
aid	name	age	sid	name	sport
1	Alice	18	1	Alice	tennis
2	Alice	25	2	Alice	frisbee
3	Bob	22	3	Bob	skating

ActiveStudents		
name	age	sport
Alice	18	tennis
Alice	18	frisbee
Alice	25	tennis
Alice	25	frisbee
Bob	22	skating

```
CREATE VIEW ActiveStudents
AS SELECT A.name, A.age, S.sport
FROM Athlete A, Sport S
WHERE A.name = S.name
```

What if we want to delete the row (Alice, 18, tennis) from ActiveStudents ?

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 25

Updates on Views

- *Key is to guarantee that update can be mapped to precisely one tuple in one base table*
- SQL-92 Updatable Views
 - Only Select-Project views on a single base table can be updated
 - Too restrictive...
- SQL-99
 - Can update field of a view if it is obtained from exactly one base table, and primary key of that table included in view.

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 26

Relational Model: Summary

- A tabular representation of data.
- Simple and intuitive, currently the most widely used database model.
- 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.
- Views can be used for External schemas, Logical data independence

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 27

Looking Forward...

- Suggested exercises: 3.1, 3.3, 3.5, 3.7, 3.9, 3.19
- Next time: Translating ER diagrams to relational Tables
 - See Chapter 3.5

1/12/09 EECS 484: Database Management Systems, Kristen LeFevre 28
