The Relational Model (1)

EGCI321: LECTURE04 (WEEK 02)

Review: Network and Hierarchical Models

Structural information is encoded implicitly using pointers.

Consequences:

- Difficult to separate conceptual and physical schemas
- Queries must explicitly navigate the data graph $\Rightarrow$ *procedural* queries
- *Procedural* (not *semantic*) specification of integrity constraints

Relational Model

All information is organized in (*flat*) *relations.*

Features:

- Simple and clean data model
- Powerful and declarative query/update languages
- Semantic integrity constraints
- Data independence

Note: Semantic integrity constraints are used to help ensure that a database accurately reflects the real world in structure and content.

Relational Database: Definitions

Relational database: a set of relations

Relation: made up of 2 parts:

- Instance : a table, with rows and columns.

  #Rows = cardinality, #fields = degree / arity.

- 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).

Think of a relation as a set of rows or *tuples* (i.e., all rows are distinct).

tuple: computing a row of values in a relational database

Degree

Attribute

STUDENT

Relation

SID	FirstName	LastName	Tel.	Gender
001	Bob	Cat	2564896	M
002	Mary	Lamb	3648954	F
003	Tom	Green	3645875	M
004	Susan	Doll	3215694	F
005	Fay	Way	2654489	F

tuple

Relation Model: Formal Definition

Universe • a set of atomic value **D** with equality (=)

Domain • a name D with a set of values dom(D) $\subseteq$ **D**

Relation • **schema**: $R(A_1{:}D_1, A_2{:}D_2, \ldots, A_k{:}D_k)$ with

- name R
- $A_1, \ldots, A_k$ a set of distinct attribute names
- $D_1, \ldots, D_k$ a collection of (not necessarily distinct) domain names

• **instance**: a finite relation

$\mathbf{R} \subseteq \mathrm{dom}(D_1) \times \ldots \times \mathrm{dom}(D_k)$

Database • **schema**: finite set of uniquely-named relation schemas

• **instance**: a relation $\mathbf{R}_i$ for each R_i

Note: **Intention of relation**: *The associated relation schema.*

Extension of a relation: *The associated set of tuples.*

Relation Model: Properties

Note: *Relation schema have named and typed attributes*

Relational instances are finite

Properties of a relation:

1. Based on (finite) set theory
   - Attribute ordering: not strictly necessary
   - Value oriented: tuples identified by attribute values
   - Instance has *set semantics:*
     - No ordering among tuples
     - No duplicate tuples
2. All attribute values are atomic
3. Degree = # of attributes in schema
4. Cardinality = # of tuples in instance

Tables

- Holds related entities or entity set
- Also called relations
- Comprised of rows and columns

Table Characteristics

- Two-dimensional structure with rows and columns
- Rows (tuples) represent single entity
- Columns represent attributes
- Row/column intersection represents single value
- Tables must have an attribute to uniquely identify each row
- Column values all have same data format
- Each column has range of values called attribute domain
- Order of the rows and columns is immaterial to the **DBMS**

Example Tables

Table name: STUDENT

STU_NUM	STU_LNAME	STU_FNAME	STU_INIT	STU_DOB	STU_HRS	STU_CLASS
321452	Bowser	William	C	Saturday, February 12, 1972	42	So
324257	Smithson	Anne	K	Tuesday, November 15, 1977	81	Jr
324258	Brewer	Juliette		Tuesday, August 23, 1966	36	So
324269	Oblonski	Walter	H	Sunday, September 16, 1973	66	Jr
324273	Smith	John	D	Friday, December 30, 1955	102	Sr
324274	Katinga	Raphael	P	Thursday, October 21, 1976	114	Sr
324291	Robertson	Gerald	T	Wednesday, April 08, 1970	120	Sr
324299	Smith	John	B	Wednesday, November 30, 1983	15	Fr

STUDENT table, cont.

STU_GPA	STU_TRANSFER	DEPT_CODE	STU_PHONE	PROF_NUM
2.84	No	BIOL	2134	205
3.27	Yes	CIS	2256	222
2.26	Yes	ACCT	2256	228
3.09	No	CIS	2114	222
2.11	Yes	ENGL	2231	199
3.15	No	ACCT	2267	228
3.87	No	EDU	2267	311
2.92	No	ACCT	2315	230

Simple Relational Database

Table name: PRODUCT
Primary key: PROD_CODE
Foreign key: VEND_CODE

Database name: CH2_SALE_CO

PROD_CODE	PROD_DESCRIPT	PROD_PRICE	PROD_ON_HAND	VEND_CODE
001278-AB	Claw hammer	$12.95	23	232
123-21UUY	Houselite chain saw, 16-in. bar	$189.99	4	235
QER-34256	Sledge hammer, 16-lb. head	$18.63	6	231
SRE-657UG	Rat-tail file	$2.99	15	232
ZZX/3245Q	Steel tape, 12-ft. length	$6.79	8	235

Table name: VENDOR
Primary key: VEND_CODE
Foreign key: none

link

VEND_CODE	VEND_CONTACT	VEND_AREACODE	VEND_PHONE
230	Shelly K. Smithson	608	555-1234
231	James Johnson	615	123-4536
232	Annelise Crystall	608	224-2134
233	Candice Wallace	904	342-6567
234	Arthur Jones	615	123-3324
235	Henry Ortozo	615	899-3425

Example: A Bibliography Database

Database Schema:

- author (aid: int, name: string)
- wrote (author: int, publication: int)
- publication (pubid: int, title: string)
- book (pubid: int, publisher: string, year:int)
- Journal (pubid:int, volume:int, no:int, year:int)
- proceedings (pubid:int, year:int)
- article (pubid:int, crossref:int, startpage:int, endpage:int)

Note: *Relational schemas are sometimes abbreviated by omitting the attribute domains.*

Example: A Bibliography Database

Sample database instance:

Database Schema:

- author (aid: int, name: string)
- wrote (author: int, publication: int)
- publication (pubid: int, title: string)
- book (pubid: int, publisher: string, year:int)
- journal(pubid:int, volume:int, no:int, year:int)
- proceedings (pubid:int, year:int)
- article (pubid:int, crossref:int, startpage:int, endpage:int)

$$
\begin{aligned}
\texttt{author} &= \{\ (1, \text{John}), (2, \text{Sue})\ \} \\
\texttt{wrote} &= \{\ (1,1), (1,4), (2,3)\ \} \\
\texttt{publication} &= \{\ (1, \text{Mathematical Logic}), \\
&\qquad (3, \text{Trans. Databases}), \\
&\qquad (2, \text{Principles of DB Syst.}), \\
&\qquad (4, \text{Query Languages})\ \} \\
\texttt{book} &= \{\ (1, \text{AMS}, 1990)\ \} \\
\texttt{journal} &= \{\ (3, 35, 1, 1990)\ \} \\
\texttt{proceedings} &= \{\ (2, 1995)\ \} \\
\texttt{article} &= \{\ (4, 2, 30, 41)\ \}
\end{aligned}
$$

Example: A Bibliography Database

Sample database instance (tabular form):

$$
\begin{aligned}
\texttt{author} &= \{\ (1, \text{John}), (2, \text{Sue})\ \} \\
\texttt{wrote} &= \{\ (1,1), (1,4), (2,3)\ \} \\
\texttt{publication} &= \{\ (1, \text{Mathematical Logic}), \\
&\quad\ \ (3, \text{Trans. Databases}), \\
&\quad\ \ (2, \text{Principles of DB Syst.}), \\
&\quad\ \ (4, \text{Query Languages})\ \} \\
\texttt{book} &= \{\ (1, \text{AMS}, 1990)\ \} \\
\texttt{journal} &= \{\ (3, 35, 1, 1990)\ \} \\
\texttt{proceedings} &= \{\ (2, 1995)\ \} \\
\texttt{article} &= \{\ (4, 2, 30, 41)\ \}
\end{aligned}
$$

author

aid	name
1	John
2	Sue

wrote

author	publication
1	1
1	4
2	3

publication

pubid	title
1	Mathematical Logic
3	Trans. Databases
2	Principles of DB Syst.
4	Query Languages

Example Instance of Students Relation

Cardinality = 3, degree = 5, all rows distinct

Do all columns in a relation instance have to be distinct?

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Relations vs. SQL Tables

Discrepancies between Relational Model and SQL

1. Semantics of Instances
   - Relations are sets of tuples
   - Tables are multisets (bags) of tuples
2. *Unknown* values
   - SQL data model defines a particular value null (intended to mean "unknown") which has some special properties

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.
  - 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)
- SQL-99 (major extensions, current standard)

Relational Query Operators

Relational algebra determines table manipulations

Key operators

- SELECT
- PROJECT
- JOIN

Other operators

- INTERSECT
- UNION
- DIFFERENCE
- PRODUCT
- DIVIDE

The SQL Query Language

To find all 18 years old students, we can write:

```
SELECT  *
FROM ~~Students~~ S
WHERE S.age=18
```

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2

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

- SELECT S.name, S.login

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"

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

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

Enrolled

sid	cid	grade
53831	Carnatic101	C
53831	Reggae203	B
53650	Topology112	A
53666	History105	B

we get:

S.name	E.cid
Smith	Topology112

Integrity Constraints

A relational schema captures only the structure of relations

Idea: *Extend relational/database schema with rules called constraints. An instance is only valid if it satisfies all schema constraints.*

Reasons to use constraints:

1. Ensure data entry/modification respects database design
   - Shift responsibility from applications to DBMS
2. Protect data from bugs in applications

Types of Integrity Constraints

Tuple-level

- Domain restrictions
- Attribute comparisons

Relation-level

- Key constraints
  - Superkey: a set of attributes for which no pair distinct tuples in the relation will ever agree on the corresponding values
  - Candidate key: a minimal superkey (a minimal set of attributes that uniquely identifies a tuple)
  - Primary key: a designated candidate key
- Functional dependencies, etc.

Key constraints: Example

employee_info:

ID	FName	LName	Parent_Name	Address	Tel.
123456	Bob	Cat	Mike Cat	Main st.	111-1234
234567	Mary	Lamb	Sam Lamb	Lakeshore St.	111-2345
345678	Susan	Cat	Tom Cat	Jane St.	111-3456

Superkey: (1) ID, (2) “FName+LName+ID”, (3) “FName+LName+Parent_Name”

Candidate key: (1) and (3)

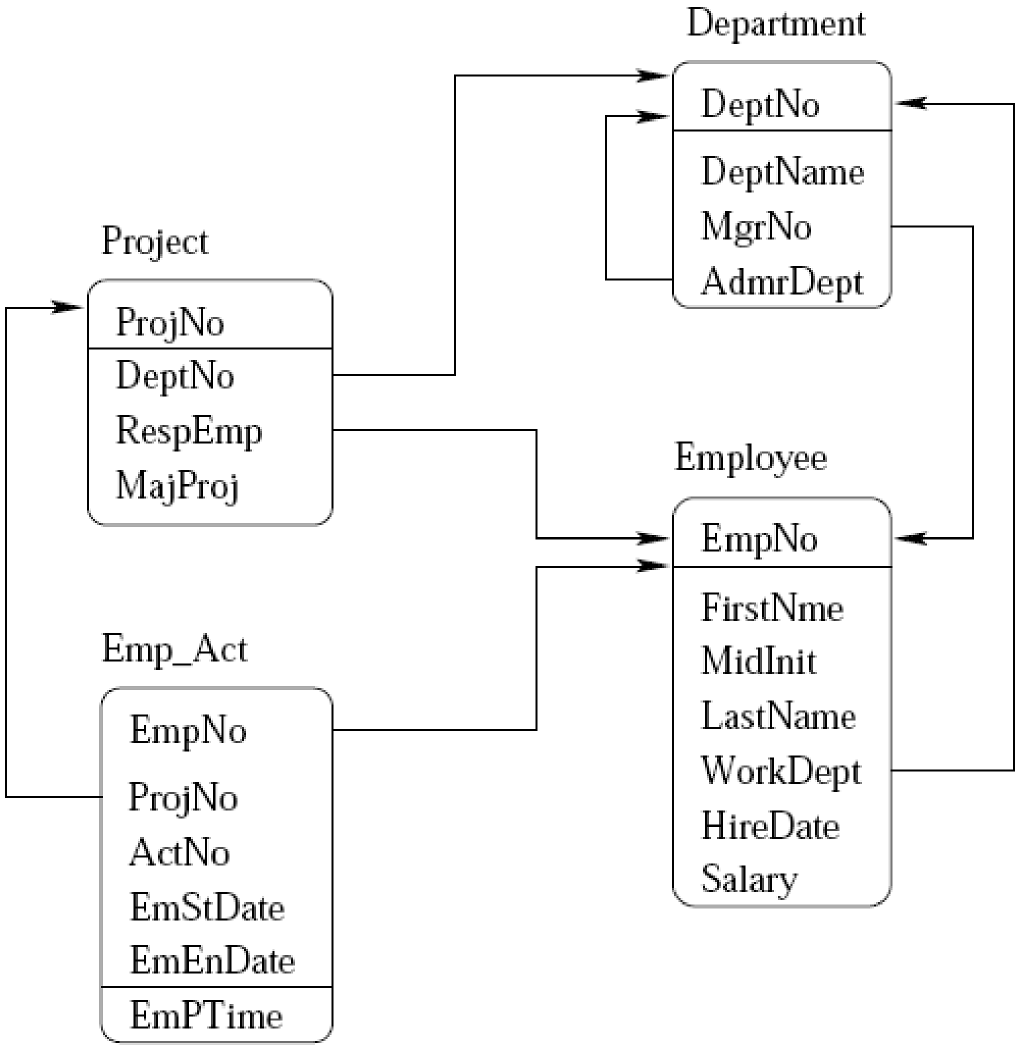
Primary key: (1)

Types of Integrity Constraints (cont.)

Database-level

- Referential integrity
  - Foreign key: Primary of one relation appearing as attributes of another relation.
  - Referential integrity: A tuple with a non-null value for a foreign key that does not match the primary key value of a tuple in the referenced relation is not allowed
- Inclusion dependencies

Example: Database Schema Showing ICs



Union

	P_CODE	P_DESCRIPT	PRICE
▶	123456	Flashlight	5.26
	123457	Lamp	25.15
	123458	Box Fan	10.99
	213345	9v battery	1.92
	254467	100W bulb	1.47
	311452	Powerdrill	34.99

UNION

	P_CODE	P_DESCRIPT	PRICE
▶	345678	Microwave	160
	345679	Dishwasher	500

yields

	P_CODE	P_DESCRIPT	PRICE
▶	123456	Flashlight	5.26
	123457	Lamp	25.15
	123458	Box Fan	10.99
	213345	9v battery	1.92
	254467	100W bulb	1.47
	311452	Powerdrill	34.99
	345678	Microwave	160
	345679	Dishwasher	500

Intersect

F_NAME
George
Jane
Elaine
Wilfred
Jorge

INTERSECT

F_NAME
Jane
William
Jorge
Dennis

yields

F_NAME
Jane
Jorge

Difference

F_NAME
George
Jane
Elaine
Wilfred
Jorge

DIFFERENCE

F_NAME
Jane
William
Jorge
Dennis

yields

F_NAME
George
Elaine
Wilfred

Product

P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26
123457	Lamp	25.15
123458	Box Fan	10.99
213345	9v battery	1.92
254467	100W bulb	1.47
311452	Powerdrill	34.99

PRODUCT

STORE	AISLE	SHELF
23	W	5
24	K	9
25	Z	6

yields

P_CODE	P_DESCRIPT	PRICE	STORE	AISLE	SHELF
123456	Flashlight	5.26	23	W	5
123456	Flashlight	5.26	24	K	9
123456	Flashlight	5.26	25	Z	6
123457	Lamp	25.15	23	W	5
123457	Lamp	25.15	24	K	9
123457	Lamp	25.15	25	Z	6
123458	Box Fan	10.99	23	W	5
123458	Box Fan	10.99	24	K	9
123458	Box Fan	10.99	25	Z	6
213345	9v battery	1.92	23	W	5
213345	9v battery	1.92	24	K	9
213345	9v battery	1.92	25	Z	6
311452	Powerdrill	34.99	23	W	5
311452	Powerdrill	34.99	24	K	9
311452	Powerdrill	34.99	25	Z	6
254467	100W bulb	1.47	23	W	5
254467	100W bulb	1.47	24	K	9
254467	100W bulb	1.47	25	Z	6

Select

Original table

P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26
123457	Lamp	25.15
123458	Box Fan	10.99
213345	9v battery	1.92
254467	100W bulb	1.47
311452	Powerdrill	34.99

SELECT ALL will yield

New table or list

P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26
123457	Lamp	25.15
123458	Box Fan	10.99
213345	9v battery	1.92
254467	100W bulb	1.47
311452	Powerdrill	34.99

SELECT only PRICE less than 2.00 will yield

P_CODE	P_DESCRIPT	PRICE
213345	9v battery	1.92
254467	100W bulb	1.47

SELECT only P_CODE=311452 will yield

P_CODE	P_DESCRIPT	PRICE
311452	Powerdrill	34.99

Project

Original table

	P_CODE	P_DESCRIPT	PRICE
▶	123456	Flashlight	5.26
	123457	Lamp	25.15
	123458	Box Fan	10.99
	213345	9v battery	1.92
	254467	100W bulb	1.47
	311452	Powerdrill	34.99

New table or list

PROJECT PRICE yields

	PRICE
▶	5.26
	25.15
	10.99
	1.92
	1.47
	34.99

PROJECT P_DESCRIPT and PRICE yields

	P_DESCRIPT	PRICE
▶	Flashlight	5.26
	Lamp	25.15
	Box Fan	10.99
	9v battery	1.92
	100W bulb	1.47
	Powerdrill	34.99

PROJECT P_CODE and PRICE yields

	P_CODE	PRICE
▶	123456	5.26
	123457	25.15
	123458	10.99
	213345	1.92
	254467	1.47
	311452	34.99

Reference

1. Ramakrishnan R, Gehrke J., Database management systems, 3rd ed., New York (NY): McGraw-Hill, 2003.