

The Relational Data Model

Instructor: Wang-Chiew Tan

Reference:

*A First Course in Database Systems,
3rd edition, Chapter 2.1, 2.2*

What is a Data Model?

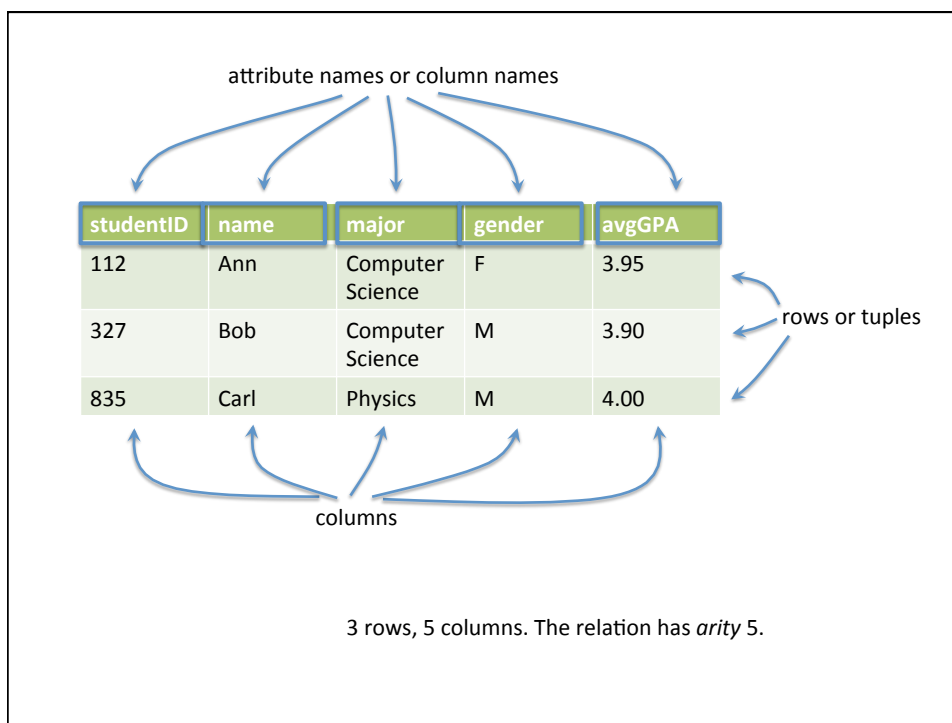
- A *data model* is a mathematical formalism that consists of two parts:
 1. A notation for describing data and mathematical objects for representing data.
 2. A set of operations for manipulating data.
- We have mentioned:
 - Network data model, hierarchical data model, relational data model, XML data model, JSON data model.

What is a relational data model?

- The relational data model (Edgar F. Codd, 1970)
 - Data is described and represented by the mathematical concept of a *relation*.
- What is a relation?
 - A table with rows and columns.

A table

studentID	name	major	gender	avgGPA
112	Ann	Computer Science	F	3.95
327	Bob	Computer Science	M	3.90
835	Carl	Physics	M	4.00



What is a relational data model? (cont'd)

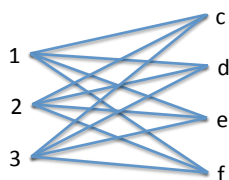
- The relational data model (Edgar F. Codd, 1970)
 - Data is described and represented by the mathematical concept of a *relation*.
- What is a relation?
 - A “table” with rows and columns.
 - A subset of a cartesian product of sets.

Cartesian product

- Example:
 - $A: \{1,2,3\}$
 - $B: \{c,d,e,f\}$
 - $A \times B = \{ (1,c), (1,d), (1,e), (1,f), (2,c), (2,d), (2,e), (2,f), (3,c), (3,d), (3,e), (3,f) \}$

Cartesian product


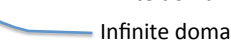
- $A: \{1,2,3\}$
- $B: \{c,d,e,f\}$
- $A \times B = \{ (1,c), (1,d), (1,e), (1,f), (2,c), (2,d), (2,e), (2,f), (3,c), (3,d), (3,e), (3,f) \}$



Tuples and Relations

- Tuple:
 - A *k-tuple* is an ordered sequence of k objects (not always distinct).
 - $(1,2)$ is a binary tuple or 2-tuple.
 - (a,b,b) is a ternary tuple or 3-tuple.
 - $(112, "Ann", "CS", "F", 3.95)$ is a 5-tuple.
- If D_1, D_2, \dots, D_k are sets of elements, then the *cartesian product* $D_1 \times D_2 \times \dots \times D_k$ is the set of all k -tuples (d_1, d_2, \dots, d_k) such that $d_i \in D_i$, where $1 \leq i \leq k$.
- Relation:
 - A *k-ary relation* is a subset of $D_1 \times D_2 \times \dots \times D_k$, where D_i , for $1 \leq i \leq k$, is a set of elements.
 - D_i is the *domain* of the i th column of the relation.

Example of a relation

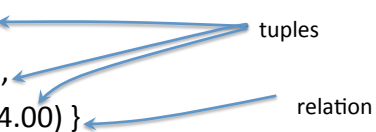
- studentID: Integer
- name: String
- major: {"CS", "Physics", "EE", "Biology", ...}
- gender: {"F", "M"} 
- avgGPA: Double 
- $r \subseteq \text{studentID} \times \text{name} \times \text{major} \times \text{gender} \times \text{avgGPA}$

Example of a relation (cont'd)

$\{ (112, \text{"Ann"}, \text{"CS"}, \text{"F"}, 3.95),$
 $(327, \text{"Bob"}, \text{"CS"}, \text{"M"}, 3.90),$
 $(835, \text{"Carl"}, \text{"Physics"}, \text{"M"}, 4.00) \}$
 \subseteq
 $\text{studentID} \times \text{name} \times \text{major} \times \text{gender} \times \text{avgGPA}$

Example of a relation (cont'd)

$\{ (112, \text{"Ann"}, \text{"CS"}, \text{"F"}, 3.95),$
 $(327, \text{"Bob"}, \text{"CS"}, \text{"M"}, 3.90),$
 $(835, \text{"Carl"}, \text{"Physics"}, \text{"M"}, 4.00) \}$



\subseteq

$\text{studentID} \times \text{name} \times \text{major} \times \text{gender} \times \text{avgGPA}$

studentID	name	major	gender	avgGPA
112	Ann	Computer Science	F	3.95
327	Bob	Computer Science	M	3.90
835	Carl	Physics	M	4.00

Can be viewed
as a table.

Practice Homework 1

- If D_1 has n_1 elements and D_2 has n_2 elements, then how many elements are there in $D_1 \times D_2$?
 - $|D_1| = n_1$, $|D_2| = n_2$, what is $|D_1 \times D_2|$?
- If D_i has n_i elements, then how many elements are there in $D_1 \times \dots \times D_k$?
- How many relations can one construct from $D_1 \times \dots \times D_k$?

Attributes and Relation Schema

- An *attribute* is the name of a column in a relation.
 - E.g., studentID, name, major, gender, avgGPA.
- A *relation schema* R is a set $\{A_1, \dots, A_k\}$ of attributes, often denoted as $R(A_1, \dots, A_k)$, where A_i is the name of the i th column of the relation.
 - The type of each attribute is atomic.
“First normal form” relation.
 - E.g., Student(studentID:int, name:str, major:str, gender:gender, avgGPA:double)
 - Or simply,
Student(studentID, name, major, gender, avgGPA)
when types are implicit.

Relation Schema and Instance of a Relation Schema

- An *instance of a relation schema* is a relation that conforms to the relation schema.
 - E.g., Student(studentID:int, name:str, major:str, gender:gender, avgGPA:double)
 - Every tuple in the relation must be a 5-tuple.
 - The i th component of each tuple in the relation must have the corresponding type.

{ (112, "Ann", "CS", "F", 3.95),
(327, "Bob", "CS", "M", 3.90),
(835, "Carl", "Physics", "M", 4.00) }



{ ("112", "Ann", "CS", "F", 3.95),
("327", "Bob", "CS", "M", 3.90),
("835", "Carl", "Physics", "M", 4.00) }



A Relation Database Schema

- A *relation database schema* or, simply, a *database schema* is a set of relation schemas with disjoint relation names.
- A university database schema:
 - Student(studentID, name, major, gender, avgGPA)
 - Course(courseID, description, department)
 - Teach(profID, courseID, quarter, year)
 - Enroll(studentID, courseID, grade)
 - Professor(profID, name, department, level)

Instance of a Database Schema

- An *instance of a database schema* $\{R_1, \dots, R_k\}$ (or a *database instance* in short) is a set $\{r_1, \dots, r_k\}$ of relations such that r_i is an instance of R_i , for $1 \leq i \leq k$.

Student	studentID	name	major	gender	avgGPA
	112	Ann	Computer Science	F	3.95
	327	Bob	Computer Science	M	3.90
	835	Carl	Physics	M	4.00

Course	courseID	description	department	Teach, Enroll, Professor, ...
	CMPS101	Algo.	CS	
	BINF223	Intro. to bio.	Biology	

Tidbit of the day



- Although Edgar F. Codd wrote his seminal paper at IBM San Jose Labs in 1969, and subsequently published his paper in 1970, IBM was slow to commercialize his ideas until its commercial rivals started to commercialize Codd's ideas.
 - IBM was very much into IMS/DB, an information management system based on the hierarchical data model with extensive transaction processing capabilities.
 - Used even till today by the U.S. Federal Reserve and big banks, supplemented by RDBMSs.
- http://en.wikipedia.org/wiki/Edgar_F._Codd

Major database vendors today

Relational Database Management Systems (RDBMS) Vendors					
Total Software Revenue, Worldwide, 2010-2011 (Millions of U.S. Dollars)					
Vendor	2010	2011	Share of 2010	Share of 2011	Growth 2011
Oracle	9,990.5	11,787.0	48.2%	48.8%	18.0%
IBM	4,300.4	4,870.4	20.7%	20.2%	13.3%
Microsoft	3,641.2	4,098.9	17.6%	17.0%	12.6%
SAP/Sybase	744.4	1,101.1	3.6%	4.6%	47.9%
Teradata	754.7	882.3	3.6%	3.7%	16.9%
Other Vendors	1,315.3	1,389.7	6.3%	5.8%	5.7%
Grand Total	20,746.6	24,129.5	100.0%	100.0%	16.3%
Source: Gartner (March 2012)					

Keys



- A *key constraint* (or a *key* in short) of a relation schema R is a subset K of attributes of R such that
 1. For every instance r of R , every two distinct tuples of r must differ in their values from K .
 - Contrapositive: if two tuples agree on their values from K , then they must be the same tuple.
 2. Minimal: no proper subset of K has the above property.
- A *superkey* is a set of attributes of R that includes a key of R .
 - Fact. All keys are superkeys but some superkeys are not keys.

Examples

- Student(studentID, name, major, gender, avgGPA).
 - {studentID} is a key. It is also a superkey.
 - {studentID, name} is a superkey but not a key.
 - {studentID, name, major, gender, avgGPA} is a superkey but not a key.
- There can be multiple keys in general.
 - One key is chosen and define as the *primary key*, while the rest are *candidate keys*.
- Student(studentID, name, dob, major, gender, avgGPA)
 - {studentID}, {name, dob} are keys and also superkeys.
 - {studentID} is the primary key.
 - {name, dob} is the candidate key.
 - {name, dob, avgGPA} is a superkey.



What is a Data Model?

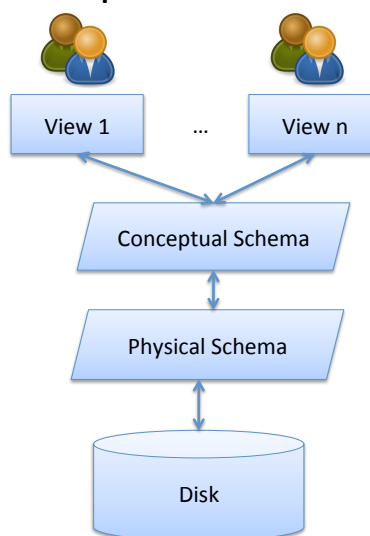
- A *data model* is a mathematical formalism that consists of two parts:
 1. A notation for describing data and mathematical objects for representing data.
 2. A set of operations for manipulating data.

Data Independence

- The relational data model provides a logical view of the data, and hides the physical representation of data.
 - Data is represented, conceptually, as tables and may not correspond to how it is stored on disk.
 - Operators manipulate data as tables. Users focus formulate queries based on *what* is needed, without knowing *how* the data is stored.
 - Optimizer generates a plan on *how* to compute the answers.
- Advantages:
 - Applications are shielded from low-level details.
 - Physical database design and optimizer can evolve without affecting applications.

Two levels of data independence

- *Logical data independence*: Protects views from changes in logical (conceptual) structure of data.
- *Physical data independence*: Protects conceptual schema from changes in physical structure of data.

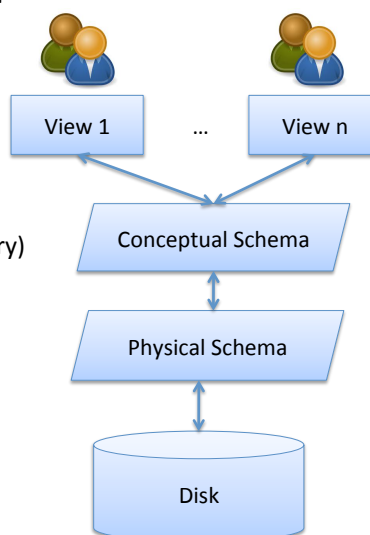


Example

View 1 defines
Faculty(name, department)
based on Professor relation
schema.

Professor(pid, name, department, salary)

Professor relation may be
stored as an ordered file.



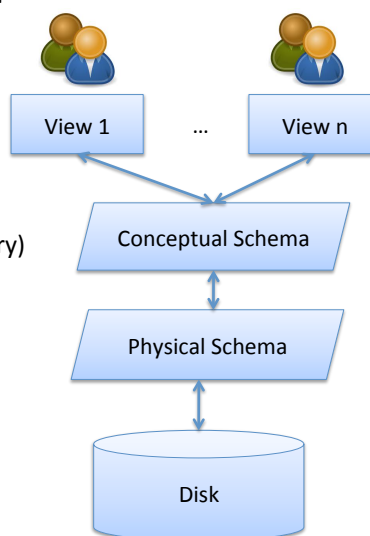
Example

View 1 defines
Faculty(name, department)
based on Professor relation
schema.

Professor(pid, name, department, salary)

Professor relation may be stored
as an unordered file with a B+
tree index on the pid.

Physical data independence:
Protects conceptual schema
from changes in physical
structure of data.



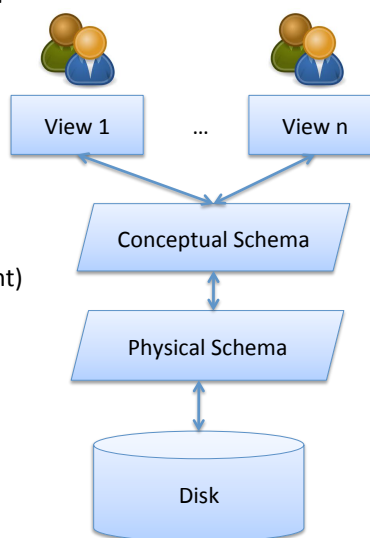
Example

View 1 defines
Faculty(name, department)
based on Professor-public
relation schema. Users
unaffected.

Professor-private(pid, salary)
Professor-public(pid, name, department)

Professor relation may be stored
as an unordered file with a B+
tree index on the pid.

Logical data independence:
Protects views from changes
in logical (conceptual)
structure of data.



Summary

- A data model.
- A relation schema.
 - Attributes or column names
 - Tuple or row
 - Columns
 - Arity
- A relation.
- A relational database schema.
- A database (or an instance of a database schema).
- Logical and physical data independence.