Database Management Systems

The Relational Model

Topics

- Definitions
  - Relations
  - Relation Schema and Table
  - Database Schema
- Nested Structures
- Incomplete Data
- Key Constraint

Relational Model

- Introduced by E. F. Codd in 1970
- It is based on the mathematical notion of ***Relation***
- Is the most widely used model
- Many vendors such as IBM, Informix, Microsoft, Oracle, Sybase, etc.

Relational Model Definitions

- ***n-tuple***: an ***n*-tuple** is an ordered list of *n* elements.
  - e.g. <'A', 11, "Store", 100 > is a 4-tuple

- ***Cartesian Product of n Sets*** $D_1 \times D_2 \times \ldots \times D_n$

  is a set of n-tuples where elements of tuples are taken from the sets $D_1, \ldots, D_n$

Cartesian Product Example

- Set-A: {1,2,3,4,5,6}
- Set-B: {'A','B','C','D'}
- Cartesian Product of Set-A and Set-B is
  - Set-A x Set-B = { (1,'A'),(1,'B'),(1,'C'),…,

,……..,

(5,'A'),(5,'B'),(5,'C'),(5,'D'),

(6,'A'),(6,'B'),(6,'C'),(6,'D')}

Relation

- A relation is a subset of the Cartesian product of $D_1 \times D_2 \times \ldots \times D_n$
- Sets $D_1, D_2, \ldots, D_n$ are called ***domains***
- ***n*** is the ***degree*** of the relation
- The number of tuples is called the ***cardinality*** of the relation

Example Relation

- Domains:
  - Set-A: {1,2,3,4,5,6}
  - Set-B: {'A','B','C','D'}
- Relation:
  - R = { (1,'A'),(2,'B'),(3,'A'),(4,'C'),(5,'D'),(6,'A'),(6,'B') }

which is a subset of *Set-A x Set-B*

Attribute, Schema, and Table

- We associate a unique name (***Attribute***)with each domain
- ***Relation Schema*** is the name of the relation (R) with a list of attributes names $A_1,\ldots, A_n$
- Table is a set of n-tuples, with a schema so:
  - there is no ordering between n-tuples
  - the n-tuples are distinct from one another (no repetition)

Example

- Domains:
  - D_1 : {1,2,..,120}
  - D_2 : {'Ali', 'Mehmet', 'Hasan'}
- Attributes:
  - Age (associated with D_1)
  - Name (associated with D_2)
- Schema:
  - StudentAge (Name, Age)
- Table *StudentAge*

Name	Age
Ali	19
Mehmet	22
Hasan	20

Database Schema

- Database schema is a set of relation schemas with different names.

e.g. University Database Schema:

{

Student (Student Id, name, major, address),

Course (Code, name, credits),

TakesCourse(StudentID, CourseCode, Year)

}

Example

- Library Database Schema

  {

  Book(BookCode, Title, Author, Publisher, ISBN),

  User(UserID, Name, Phone, Address),

  Borrowed(BookCode, UserID, Date, Due_Date)

  }

Nested Structures

- If data is about different things, we have to put it in different structures.
- These structures are nested when they are related.
- Problem:
  - Nested structures can not be defined by a single relation without redundancy.
- Solution
  - Use several relations

Example: Nested Structure

Da Mario		
Receipt No: 1357		
Date: 5/5/92		
3	covers	3.00
2	hors d'oeuvre	5.00
3	first course	9.00
2	steak	12.00
	Total:	29.00

Da Mario		
Receipt No: 2334		
Date: 4/7/92		
2	covers	2.00
2	hors d'oeuvre	2.50
2	first course	6.00
2	bream	15.00
2	coffee	2.00
	Total:	27.50

Da Mario		
Receipt No: 3007		
Date: 4/8/92		
2	covers	3.00
2	hors d'oeuvre	6.00
3	first course	8.00
1	bream	7.50
1	salad	3.00
2	coffee	2.00
	Total:	29.50

Representing Nested Structures by a Single Relation (Data Redundancy)

Number	Quantity	Description	Cost	Date	Total
1357	3	Covers	3.00	5/5/92	29.00
1357	2	Hors d'oeuvre	5.00	5/5/92	29.00
1357	3	First course	9.00	5/5/92	29.00
1357	2	Steak	12.00	5/5/92	29.00
2334	2	Covers	2.00	4/7/92	27.50
2334	2	Hors d'oeuvre	2.50	4/7/92	27.50
2334	2	First course	6.00	4/7/92	27.50
2334	2	Bream	15.00	4/7/92	27.50
2334	2	Coffee	2.00	4/7/92	27.50
3007	2	Covers	3.00	4/8/92	29.50
3007	2	Hors d'oeuvre	6.00	4/8/92	29.50
3007	3	First course	8.00	4/8/92	29.50
3007	1	Bream	7.50	4/8/92	29.50
3007	1	Salad	3.00	4/8/92	29.50
3007	2	Coffee	2.00	4/8/92	29.50

Removing Redundancy by Using Multiple Relations

Receipts

Number	Date	Total
1357	5/5/92	29.00
2334	4/7/92	27.50
3007	4/8/92	29.50

Details

Number	Quantity	Description	Cost
1357	3	Covers	3.00
1357	2	Hors d'oeuvre	5.00
1357	3	First course	9.00
1357	2	Steak	12.00
2334	2	Covers	2.00
2334	2	Hors d'oeuvre	2.50
2334	2	First course	6.00
2334	2	Bream	15.00
2334	2	Coffee	2.00
3007	2	Covers	3.00
3007	2	Hors d'oeuvre	6.00
3007	3	First course	8.00
3007	1	Bream	7.50
3007	1	Salad	3.00
3007	2	Coffee	2.00

Incomplete Data

- The relational model impose a rigid structure to data:
  - information is represented by means of tuples
  - tuples have to conform to relation schemas
- In practice data may have some differences with the schema

Incomplete Data Solution

- In case of Student relation
  - Student(Student ID, Name, Major, Phone, Address)

    some student may have no telephone number.

    We have to leave the phone attribute in that tuple empty.
- An attribute left empty is said to have ***Null*** value.
- Null value is a special value (not a value of the domain)

  (e.g. Do not use "zero" as Null value for GPA attribute)

Types of Null Value

- Null value is used in three cases
- ***Unknown Value:*** there is a domain value, but it is not known (student has a birth-place but we do not know it)
- ***Not-existent Value:*** the attribute is not applicable for the tuple (e.g. in library database, periodicals do not have ISBN)
- ***No-information Value:*** we don‘t know whether a value exists or not (phone number of a student)
- DBMSs do not distinguish between the types.

Meaningless Database Examples

- Sometimes data in a tuple can be invalid

Exams

RegNum	Name	Course	Grade	Honours
6554	Rossi	B01	K	
8765	Neri	B03	C	
3456	Bruni	B04	B	honours
3456	Verdi	B03	A	honours

Courses

Code	Title
B01	Physics
B02	Calculus
B03	Chemistry

  - e.g. same RegNum for two students, invalid grade (K), invalid course code (B04), honors for a B grade,..
  - ***Constraints*** are used to avoid invalid data

Unique Identification of Tuples

- In each relation we should be able to uniquely identify tuples.

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

  - the registration number identifies students:
    - there is no pair of tuples with the same value for RegNum

Key of a Relation (Key Constraint)

- ***Key*** is a set of attributes that uniquely identifies tuples in a relation

  e.g. ***Student ID*** is the key for Student relation

  (No two students have the same ID)

  e.g. ***ISBN*** is the key for Book relation

  (No two books have the same ISBN)

Primary Key

- Keys are used for uniquely identifying tuples.
- Therefore non-null key attributes are important.
- A relation may have several keys.
- The key attribute which is not null is selected as the ***primary key***

  e.g. Student Table has the keys:

  Student ID (can be primary key)

  Name, BirthDate (BirthDate can be null)

Referential Key (Foreign Key)

- If a relation includes an attribute which is primary key in another relation, the attribute is called ***foreign key.***
- Foreign keys are used for connecting relations to each other.

  e.g.

  Instructor (ID, Name, BirthDate, Phone, Office, DeptCode)

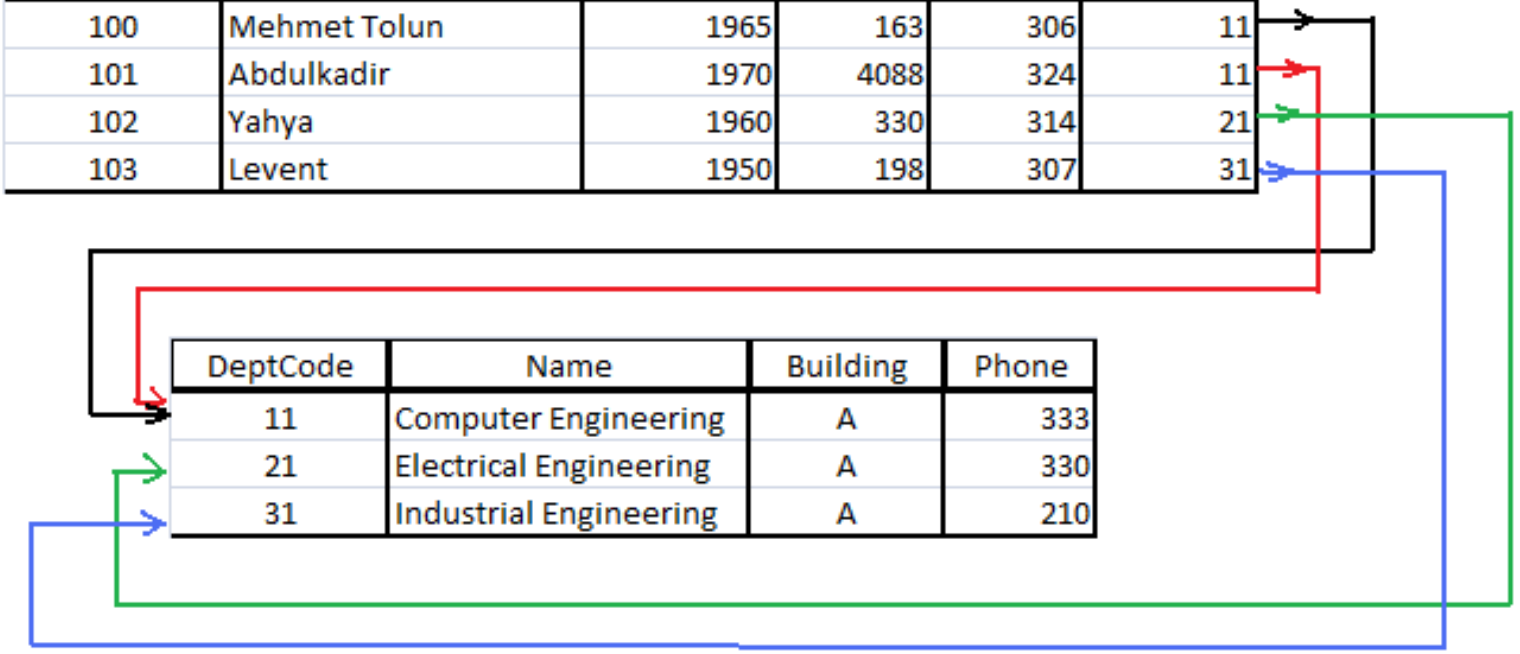
  Department(DeptCode, Name, Building, Phone)

  DeptCode in Instructor relation is a foreign key

Example

ID	Name	BirthDate	Phone	Office	DeptCode
100	Mehmet Tolun	1965	163	306	11
101	Abdulkadir	1970	4088	324	11
102	Yahya	1960	330	314	21
103	Levent	1950	198	307	31

DeptCode	Name	Building	Phone
11	Computer Engineering	A	333
21	Electrical Engineering	A	330
31	Industrial Engineering	A	210



Summary

- A database is a set of relations
- Each relation is a set of n-tuples
- Each relation has a schema which defines its structure
- An element of an n-tuple can have null value
- Primary key is used to identify tuples in a relation
- Foreign keys are used to connect relations

Questions?