Database Management Systems

The Relational Model

Topics

- Definitions
  - Data, Database, Database Management Systems
- Importance of DBMS and Applications
- Data Modeling
- Query and Query Languages
- Concurrency Control
- Transaction
- Relations, Relation Schema and Table, Database Schema
- Nested Structures
- Incomplete Data
- Key Constraint

Data and Information

- Data is the result of a measurement, event or fact.
  - Numbers, characters, symbols, images etc., which can be processed by a computer.
  - Example: John
- Information is the data that has been processed to be meaningful to the person who receives it.
  - Information is the knowledge derived from study, experience (by the senses), or instruction.
  - Example: John is the manager

Databases

- A collection of data which
  - Models the real world entities (Student, employee, etc.)
  - Models the relationships (e.g., Lisa is taking CS 356)

- Therefore, a database includes both data and information

Example

- A University Database is a collection of:
  - Student (entity)
  - Courses (entity)
  - Instructors (entity)
  - Relations such as
    - Which students are taking a given course
    - Who is offering a given course
    - Etc.

Files and Databases

- Databases are using file systems but they extend them as follows:
  - Databases are more efficient (By using indexing, hashing and other optimization tools)
  - Concurrent access to data is safer in databases
  - Data security is better

Database Management Systems (DBMS)

- A Database Management System (DBMS) is a software designed to store and manage databases.
- DBMS
  - Provides access to data
  - Protects data from inconsistency due to multiple concurrent users
  - Provides security
  - and more…

Example Databases

- Airline Reservation System Database
  - List of flights
  - List of seats sold for each flight
  - List of passenger names
  - Concurrent access control (several sale agencies may sell a seat at the same time)
  - Fast search for a flight, passenger, connection, etc.
  - Restrict access to database (Security)

Importance of DBMS

- DBMS provides the possibility of storing and extracting information and data

  e.g. The Employee-Department relationship
- DBMS can handle data sets of very large sizes

  e.g. Millions of data items in a typical data set
- DBMS can handle the diversity of data sets

  e.g. Numbers, String, Images, Video, Audio, etc.
- Many subjects in computer science include a database (AI, Operating Systems, Multimedia, etc)

Sharing Data

- The data of the various parts often overlap

  e.g. Student affairs and Library in University database
- A database is a resource, shared by various parts

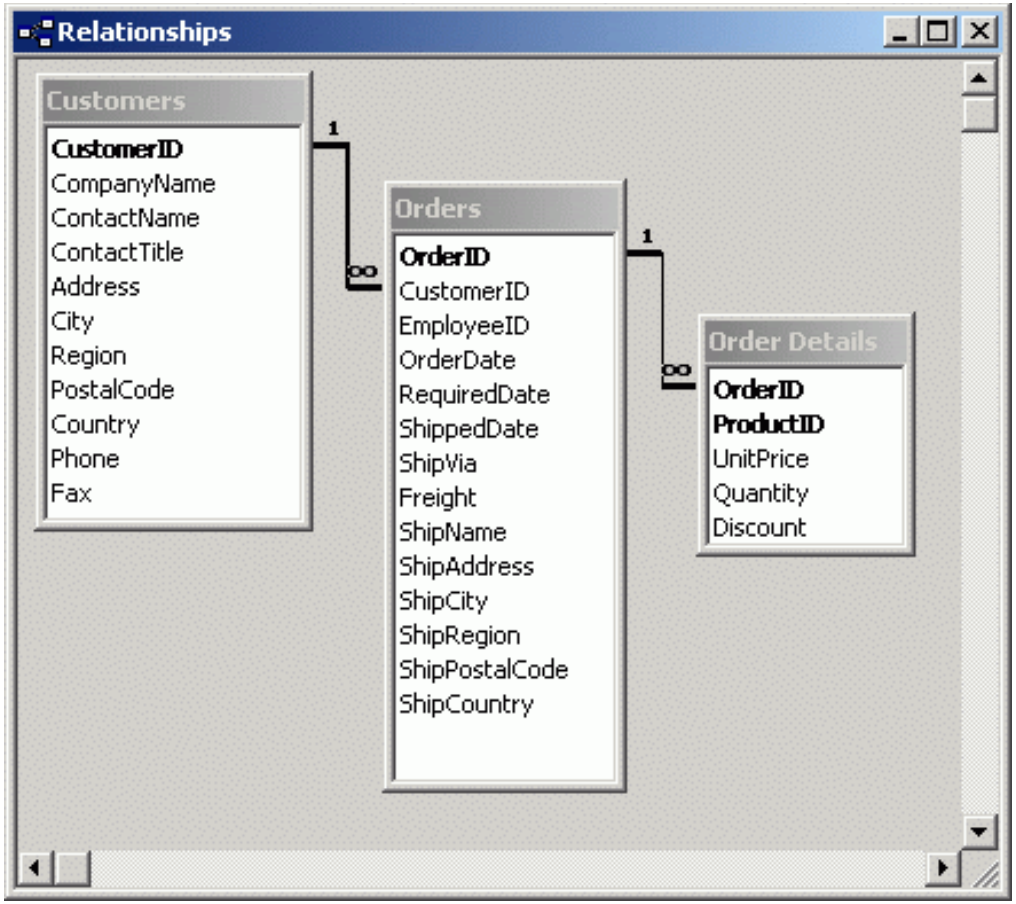
  e.g. Student address and phone number shared by Student affairs and Library
- Sharing reduces redundancy and the probability of inconsistency

  e.g. Phone number changes are reflected to all departments
- Since sharing is never complete, DBMS provides support for privacy of data
- Sharing also requires that multiple accesses to data are suitably organized

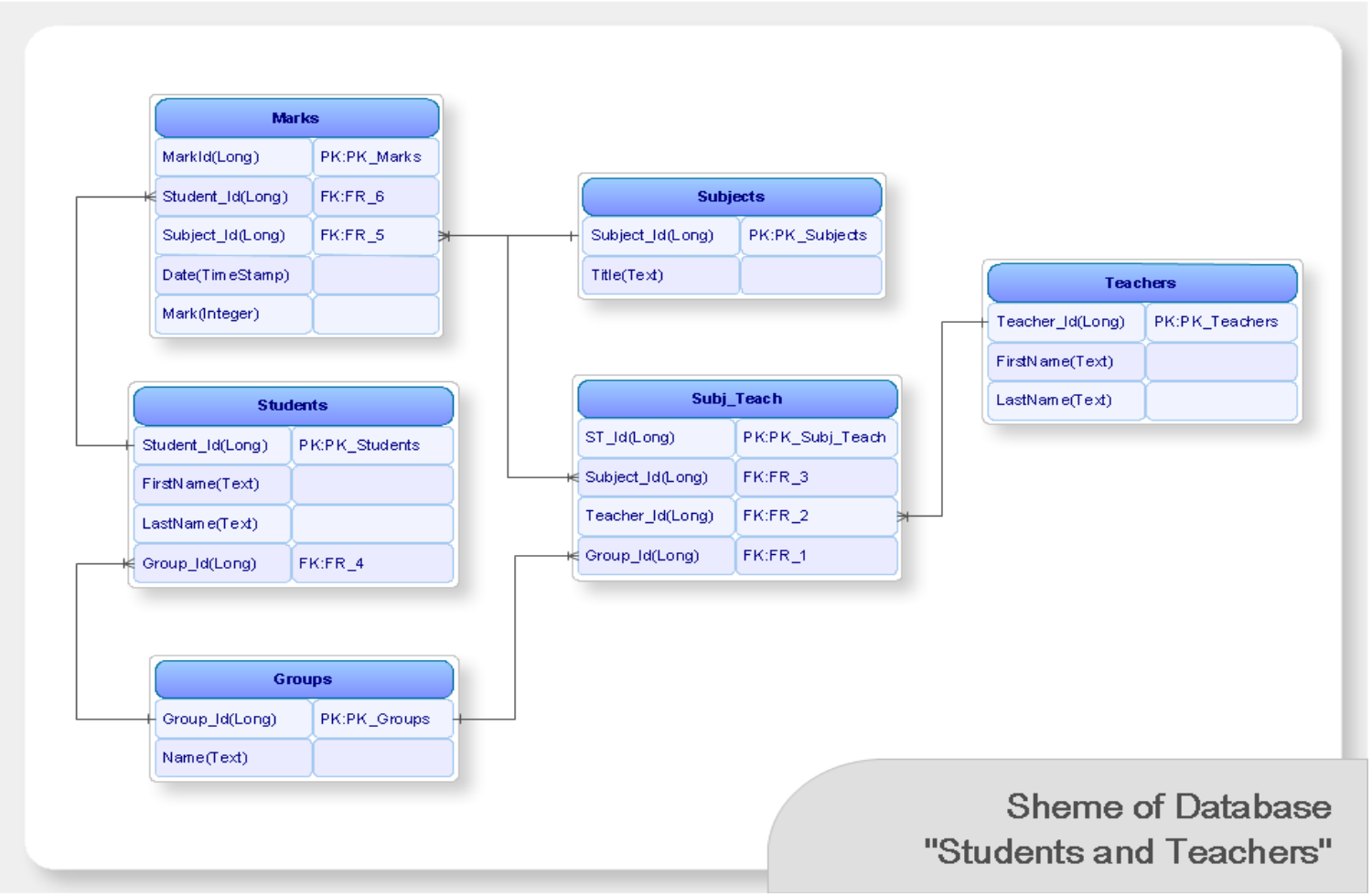
Data Models

- A data model is the method for describing data.
- The *relational* model of data is the most widely used model today.
- A *relation* is basically a table with rows and columns.

Relational Data Model Example



Example: University Database



Query

- In general, a query is a form of questioning, but we will use the term for updating data/information also.
  - e.g. Find all students taking CS 356 in spring 2020 from the University database (Retrieve Query)
  - e.g. Increase the salaries of all employees by 10% in Employee database (Update Query)

Query Languages

- Query languages are used for writing queries.

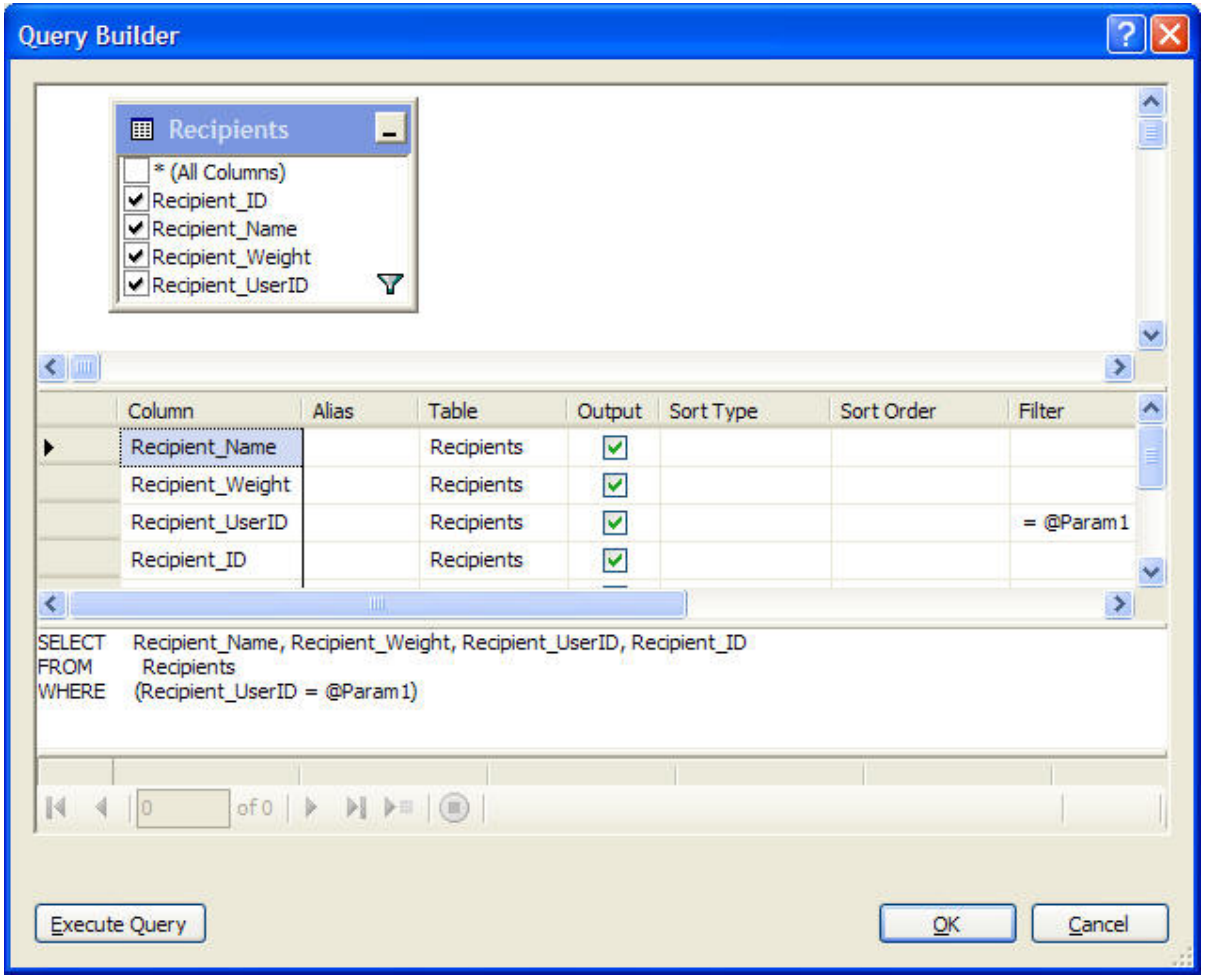
- Query languages have three types:
  - Text based languages: e.g. SQL
  - Embedded in programming languages like C or C++: e.g. embedded SQL
  - Graphical Interfaced Query Languages: e.g. Microsoft Access

Query Language Example (1)

- SQL

```
SELECT Course, Room, Floor
FROM Rooms, Courses
WHERE Code = Room
AND Floor=”Ground"
```

Query Language Example (2)



Concurrency Control

- Concurrent execution of user programs may be needed in many applications..
- Concurrency is essential for good DBMS performance.

- Problems: Inconsistency
  - e.g., check is cleared while account balance is being computed

Transactions

- Each instruction given to a database is either executed completely, or cancelled. This is called a transaction.
  - e.g. Instruction given as : "Increase the salary of all employees by 10%".
  - If the system crashes before completing the execution, we will not know whose salary has been updated. (Inconsistency)
  - If the instruction is executed as a transaction, inconsistency will not happen.

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

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 : {'John', 'Lisa', 'David'}
- Attributes:
  - Age (associated with D_1)
  - Name (associated with D_2)
- Schema:
  - StudentAge (Name, Age)
- Table

StudentAge

Name	Age
John	19
Lisa	22
David	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)

  }

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	David	1965	163	306	11
101	Mary	1970	4088	324	11
102	Daniel	1960	330	314	21
103	Emma	1950	198	307	31

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

Summary

- DBMS is used to maintain, and query large datasets
- Benefits of DBMS are:
  - Sharing data
  - Less redundancy
  - Data consistency
  - Concurrency control
  - Data security
  - Efficiency of handling data
- Disadvantage:
  - Cost
  - ?

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?