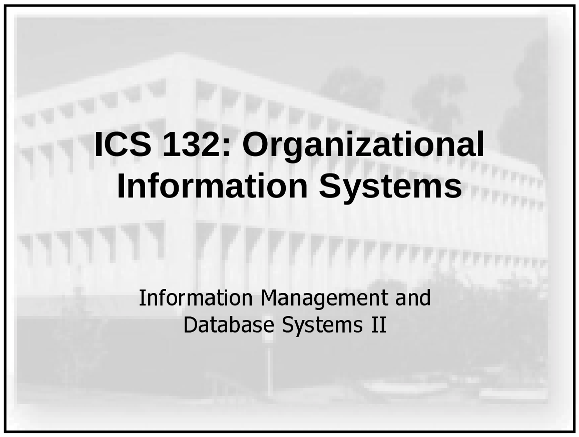

ICS 132: Organizational Information Systems

Information Management and
Database Systems II

relational databases

- schemas
  - relational databases based on formal data definitions
  - again, like specifying classes
  - schema describes table structure and storage req'ts
  - table "book":
    - author CHAR(50)
    - title CHAR(100)
    - isbn CHAR(30)
    - price DECIMAL(5,2)

exploiting structure

- all DBMS exploit common structure
  - common structure across instances
    - all books have these properties
  - common structure across databases
    - all data can be modeled in this way
      - e.g. relational data model
  - what's the point of this common structure?

SQL

- SQL is the Structured Query Language
  - originally developed for IBM's System/R in 1970s
  - now an open standard (actually, a bunch of them)
- a common interface for relational DB's
  - manipulation
    - creating tables, updating them, adding data
  - examination
    - looking data up: *queries*

SQL

- queries have three basic components
  - select
    - what aspects of the data do we want to see
  - from
    - what tables contain it
  - where
    - filtering of results
- syntax
  - **select *attribute1, attribute2,…* from *relation1, relation2, …* where *predicate***

SQL

- some basic examples
  - **select title from books**
  - **select title from books where author='dourish'**
  - **select title from books where author='dourish' and price < 35.00**
  - **select grade from students where id='12312312'**
  - **select id,name from students where grade='f'**

SQL

- queries across multiple tables
  - relational model splits data into different tables
  - queries need to integrate across multiple tables
  - selects that combine table are called *joins*
- example
  - tables: "students" (id, name), "grades" (id, score)

  -
    ```
    select name, score
    from students, grades
    where students.id = grades.id
    ```

SQL

- combining results
  - union, intersect, except
  - these are operators over *selections*
- examples

  -
    ```
    select title from books where author =
    ‘dourish’ except select title from books
    where title = ‘context-aware computing’
    ```

  -
    ```
    select id from homework1 where score > 85
    intersect select id from homework2 where
    score > 85
    ```
  - *NB:* neither of these are the easiest ways to do them…

SQL

- postprocessing (order, group)
  - need to organise results
  - order (sort), group (clustering)
- examples

  -
    ```
    select id,name,score from students
    order by score
    ```

  -
    ```
    select model, price from products where
    price < 100 order by price desc
    ```

  -
    ```
    select manufacturer from price_list
    group by manufacturer
    ```

SQL

- some processing over results
  - e.g. avg(), sum(), count(), min(), max() …
- examples

  -
    ```
    select count(*) from students where
    grade=‘a’
    ```

  -
    ```
    select avg(score) from grades
    ```

  -
    ```
    select author, avg(price) as average
    from books group by author order by
    average
    ```

SQL

- summary
  - selecting, combining, processing
- there’s more, of course…
  - subqueries
  - update and modification as well as querying

using SQL

- what SQL is not
  - not a full programming language
  - not a development environment
- sql queries normally embedded in programs
  - e.g. from java, using JDBC
  - languages differ in their degrees of integration

using SQL

```
Class.forName(JDBC_CLASS);
Connection conn = DriverManager.getConnection(DB_URL, "ics132", “password");
Statement statement = conn.createStatement();
ResultSet rs = statement.executeQuery(“select title,author from books”);
ResultSetMetaData md = rs.getMetaData();

out.println("<TABLE BORDER=2>");
out.println("<TR>");
for (int i = 1; i < md.getColumnCount() + 1; i++) {
  out.println("<TD><B>" + md.getColumnName(i).trim() + "</B></TD>");
}
out.println("<TR>");
while (rs.next()) {
  out.println("<TR>");
  for (int i = 1; i < md.getColumnCount() + 1; i++) {
    out.println("<TD>" + rs.getString(i) + "</TD>");
  }
  out.println("</TR>");
}
out.println("</TABLE>");
```

queries and definitions

- must consider queries & definitions together
  - form of the database determines query complexity
  - reducing joins
- constraints on data definitions
  - looking at queries reveals patterns of definition
    - e.g. for multiway relations
- database normalization
  - a set of procedures for structuring relations
  - normal forms

first normal form

- no repeating groups
  - essentially, normalise the record length

Title	Price	Author1	Author2	Author3
Where the Action Is	$30.00	Dourish		
Analyzing Social Settings	$31.95	Lofland	Lofland	
Compilers	$72.00	Aho	Sethi	Ullman

first normal form

- no repeating groups
  - essentially, normalise the record length

Title	Price	Author
Where the Action Is	$30.00	Dourish
Analyzing Social Settings	$31.95	Lofland
Compilers	$72.00	Aho
Compilers	$72.00	Sethi
Compilers	$72.00	Ullman

second normal form

- no non-key attributes depend on *part* of the key
  - essentially, make key as small as it can be

Author	Title	Price	Email
Dourish	Where the Action Is	$30.00	jpd@ics.uci.edu
Baldi	Bioinformatics	$49.95	baldi@ics.uci.edu

second normal form

- no non-key attributes depend on *part* of the key
  - essentially, make key as small as it can be

Author	Email
Dourish	jpd@ics.uci.edu
Baldi	baldi@ics.uci.edu

Author	Title	Price
Dourish	Where the Action Is	$30.00
Baldi	Informatics	$49.95

third normal form

- no attributes depend on other *non*-key attributes
  - every relation should be about just one thing

Author	Title	Price	Purchaser	Date
Dourish	Where the Action Is	$30.00	Maria	12/21/00
Dourish	Where the Action Is	$30.00	Joe	1/1/01
Baldi	Bioinformatics	$49.95	Lisa	1/2/01

third normal form

- no attributes depend on other *non*-key attributes
  - every relation should be about just one thing

Title	Purchaser	Date
Where the Action Is	Maria	12/21/00
Where the Action Is	Joe	1/1/01
Bioinformatics	Lisa	1/2/01

Author	Title	Price
Dourish	Where the Action Is	$30.00
Baldi	Informatics	$49.95

phases

- definition
- querying
- execution?

the transaction model

- normalisation spreads data across multiple tables
  - single action requires many updates
    - a new customer placing a new order?
  - may be executing many operations concurrently
  - consistency is king
    - across time and "space"
- "transactions" group operations into logical units
  - all-or-nothing execution semantics
  - "rollback"

the ACID properties

- Atomicity
- Consistency
- Isolation
- Durability

the ACID properties

- Atomicity
  - all-or-nothing semantics for transactions
- Consistency
- Isolation
- Durability

the ACID properties

- Atomicity
  - all-or-nothing semantics for transactions
- Consistency
  - goes from one consistent state to another
- Isolation
- Durability

the ACID properties

- Atomicity
  - all-or-nothing semantics for transactions
- Consistency
  - goes from one consistent state to another
- Isolation
  - one transaction doesn’t see another’s intermediate products
- Durability

the ACID properties

- Atomicity
  - all-or-nothing semantics for transactions
- Consistency
  - goes from one consistent state to another
- Isolation
  - one transaction doesn’t see another’s intermediate products
- Durability
  - transaction’s changes are persistent

assignment

- two questions
  - one on queries
  - one on normalization
- database running on drzaius.ics.uci.edu
  - let me ***and*** TAs know quickly in case of problems
- assignment is due in Monday’s lecture

summary

- key points:
  - information processing is about making the world tractable
    - amenable to summarisation, modeling & prediction
  - DBMS provides a framework for data management
    - regularised for efficiency, consistency & maintenance
  - relational databases
    - organise information according to relations & tables
    - sql provides uniform access

what’s coming up

- Friday
  - discussion section
- Monday
  - performance and competition
  - read Alter chapter 6
- next Wednesday is the mid-term
  - I’ll set office hours next week to discuss problems or questions
    - as usual, I’m also available any other time you can find me free…