Announcement

1. Assignments submitted later than deadline will have 50% points at most.
2. We will returned to Business Intelligence

DB assignment is due two weeks later.(11/20/2012).

3. The grades for every assignment will be given in eCommons.
4. It's important to check webpage to get the latest information and assignments changes.

No Class on Thanks Giving Day, 11/22 Thursday

Final Exam

1st Choice  December 7, Friday
2nd Choice  December 10, Monday depending on Schedule Permission

Format is same as Midterm
Covering  Up to Midterm  30- %
          After Midterm  70+ %
Applications and the Organizations

**Review**

- Layered Structure Components
- System Integration

**Network Computing Organizations**

- Central/ Networked Stove piped/Integrated Structure
- Interfaces H/W, S/W
- Reusability
- Interoperability
- Scalability

**Acquisition Options**

- Make/Buy, and between
- Collaboration, Networked Standards/ diversified Quality Competition Product, Service
- Outsourcing *Offshore outsourcing*
- Planning, Testing Analysis, Deployment, Design, Maintenance Development

**Application Life cycle**

- Correctness
- Maintainability
- Dependability
- Usability
- Reliability

**Productivity**

- Efficiency
- Innovation
- Lifetime Expenses

**S/W Engineering**
Layered System requires Components/Software Qualities

- must be planned for
  - **Reusability**
    - ability to construct new software from existing pieces
  - **Interoperability**
    - ability of software (sub)systems to cooperate with others

**Scalability**

ability of a software system to grow in size while maintaining its properties and qualities
Information System Components

Figure 1–9
Understanding The Business

- New Kinds of Companies
- Companies are classified based on their main activities:
  - Production-oriented
  - Service-oriented
  - Brick-and-mortar
  - Dot-com (.com)
Acquisition Options

Make Vs. Buy and Gray Region

Purchase/ Outsourced Development/
Internal Development

*Fixed price
*Time and material Prices
Outsourcing

- Outsourcing – the practice of turning over responsibility of some to all of an organization’s information systems development and operations to an outside firm
Outsourcing

• Why Outsource?
  – Cost and quality concerns
  – Problems in IS performance
  – Supplier pressures
  – Simplifying, downsizing, and reengineering
  – Financial factors
  – Organizational culture
  – Internal irritants
Outsourcing

• **Outsourcing** has two meanings:
  – Commission development of application to other organization
  – Hire services of other company to manage services

• May not encompass development
Then, What to select?

The Nature of Decision Making

- Making effective decisions, as well as recognizing when a bad decision has been made and quickly responding to mistakes, is a key ingredient in organizational effectiveness.

- Some experts believe that decision making is the most basic and fundamental of all managerial activities.

- Decision making is most closely linked with the Planning function.
System Applications

• The process of IS management
• The system development life cycle (SDLC)
• Alternative approaches to system development
• In-house system development
• External acquisition, outsourcing, and end-user development
Application Life Cycle Considerations

Management
End-users
Operations and Administrations
Maintenance and Organization
Suppliers and Customers
What Is an IT Application?

Definition

• **Single-user System/Personal System**: An IT system used by only one person. A system that stands alone and is not interconnected with other companies or shared by other people.

• **Enterprise System**: Usually involves the same types of components as a personal system, plus server or mainframe, database, and network. It is generally a shared system.
What Is an IT Application?

Impact of IT Applications

- Improved Productivity

Greater Effectiveness

Effectiveness: The extent to which desirable results are achieved.

Increased Creativity and Innovation
The Systems Development Life Cycle (SDLC)

1: Planning
   - Does system make sense? Feasibility. Scheduling.

2: Analysis
   - How can system solve business problem? LAYPERSON language.

3: Design
   - How can system solve business problem? TECHIE language.

4: Development
   - Build the system.

5: Testing
   - Test the system. 3 times the time and resources of programming!!

6: Implementation
   - Convert from old system to new system.

7: Maintenance
   - Fix, maintain, and improve system.
Relative Costs of Fixing Software Faults

- Requirements: 1
- Specification: 2
- Planning: 3
- Design: 4
- Implementation: 10
- Integration: 30
- Maintenance: 200
Kinds of System Deployment

(a) Parallel
Old System
New System
Description: Old and new systems are used at the same time.

(b) Direct
Old System
New System
Description: Old system is discontinued on one day and the new is used on the next.

(c) Phased
Old System
New System
Description: Parts of the new system are implemented over time.

(d) Pilot (single location)
Old System
New System
Description: Entire system is used, is used in one location.
7. **Maintenance phase** – involves performing changes, corrections, additions, and upgrades to ensure the system continues to meet the business goals

>> operations, Maintenance and Upgrade

**Lifetime Expenses Concepts**

>> H/w + S/W + Maintenances
Regardless of your major, you will be involved in Steps 1, 2, 5, 6 and 7

- Technology is a huge part of business and our world in general.
- You can’t “not like computers” anymore and survive.
- Minimally, you need to understand technology and its consequences for business and life.
- That’s true even if you outsource!
What’s “outsourcing”?

• Generally that means your organization is not able/willing to complete steps 3 and 4.
• You STILL will be involved in 1, 2, 5, 6 and 7.
• Even if your hire a consultancy you can’t escape completely!!
Sourcing’s New Surge - Offshoring

• **Offshore outsourcing** – using organizations from developing countries to write code and develop systems

• According to Forrester Research, nearly half of all businesses use offshore providers, and two-thirds plan to send work overseas in the near future
Best Development Methodology
Software Engineering

The problem is *complexity*
Many sources, but *size* is a key:
- Mozilla contains 3 Million lines of code
- UNIX contains 4 million lines of code
- Windows 2000 contains $10^8$ lines of code

Second is role and combinatorics of “state”

Third is uncertainty of “inputs” and their timing
Fourth is the continuing changing “environment” and demands.

Software engineering is about managing all the sources of complexity to produce effective software.
What is the difference between software engineering and computer science?

**Computer Science**
is concerned with
- theory
- fundamentals

**Software Engineering**
is concerned with
- the practicalities of developing
- delivering useful software

*Computer science theories* are currently insufficient to act as a complete underpinning for software engineering, BUT it is a foundation for practical aspects of software engineering.
What is the difference between software engineering and system engineering?

- **Software engineering** is part of **System engineering**
- **System engineering** is concerned with all aspects of computer-based systems development including
  - hardware,
  - software and
  - process engineering
- **System engineers** are involved in
  - system specification,
  - architectural design,
  - integration and deployment
Software Engineering ≠ Software Programming

• Software programming
  – Single developer
  – “Toy” applications
  – Short lifespan
  – Single or few stakeholders
    • Architect = Developer = Manager = Tester = Customer = User
  – One-of-a-kind systems
  – Built from scratch
  – Minimal maintenance
Software Engineering ≠ Software Programming

• Software engineering
  – Teams of developers with multiple roles
  – Complex systems
  – Indefinite lifespan
  – Numerous stakeholders
    • Architect ≠ Developer ≠ Manager ≠ Tester ≠ Customer ≠ User
  – System families
  – Reuse to amortize costs
  – Maintenance accounts for 60%-80% of overall development costs
Software Engineering: From Principles to Tools
Software Qualities

• Qualities are goals in the practice of software engineering, and directly relate to many of the guiding principles.

• External vs. Internal qualities
• Product vs. Process qualities
Software Qualities

• Critical Quality Attributes
  - Correctness
  - Maintainability
  - Dependability
  - Usability
  - Reliability

• Other Attributes
  - Completeness
  - Compatibility
  - Portability
  - Internationalization
  - Understandability
  - Scalability
  - Robustness
  - Testability
  - Reusability
  - Customizability
  - Efficiency

End of Review
Topics of Business Intelligences

The problems of managing data resources in a traditional file environment

Important database design principles

The database management system

The capabilities and value of a database management system

Tools and technologies for accessing information from databases

The role of information policy, data administration, and data quality assurance in the management of a firm’s data resources
Foundation of business Intelligence

- Data Base System
  Information Management

- File Management System
  File Processing Procedure

- Data Base Systems

- Intelligence from Collection of Data
  Information Management
  Business Applications

- Data Integrity Control
  Business Data Maintenances
RR Donnelley Tries to Master Its Data

**Problem:** Explosive growth created information management challenges.

**Solutions:** Use MDM to create an enterprise-wide set of data, preventing unnecessary data duplication.

Master data management (MDM) enables companies like R.R. Donnelley to eliminate outdated, incomplete or incorrectly formatted data. Demonstrates IT’s role in successful data management.

Illustrates digital technology’s role in storing and organizing data.
Organizing Data in a Traditional File Environment

File organization concepts

**Database**: Group of related files

**File**: Group of records of same type

**Record**: Group of related fields

**Field**: Group of characters as word(s) or number

  Describes an entity (person, place, thing on which we store information)

**Attribute**: Each characteristic, or quality, describing entity

  E.g., Attributes Date or Grade belong to entity COURSE
A computer system organizes data in a hierarchy that starts with the bit, which represents either a 0 or a 1. Bits can be grouped to form a byte to represent one character, number, or symbol. Bytes can be grouped to form a field, and related fields can be grouped to form a record. Related records can be collected to form a file, and related files can be organized into a database.
Information as Processed Data

Any Production System

Parts

Storage

Assembly Line

Product

Organized Parts
(The value is in the organization)

Any Information System

Data

Database

Program (Instruction Codes)

Information

Organized Data
(a report, a form, on a screen or on paper)
Problems with the traditional file environment

Old Business Process;
Files maintained separately by different departments

Data redundancy:
Presence of duplicate data in multiple files

Data inconsistency:
Same attribute has different values

Program-data dependence:
When changes in program requires changes to data accessed by program

Lack of flexibility
Poor security
Lack of data sharing and availability
The use of a traditional approach to file processing encourages each functional area in a corporation to develop specialized applications. Each application requires a unique data file that is likely to be a subset of the master file. These subsets of the master file lead to data redundancy and inconsistency, processing inflexibility, and wasted storage resources.
Business Processes with Old Data Processing

System Inefficiencies
Longer Business Cycle
No Firm wise Information or Data Access
No Data Security
No Decision on Integrated Data and information
High Business Process Expenditure
Introduction of Data Processing System

Database – collection of persistent data from business divisions

Database Management System (DBMS) – software system that supports creation, population, and querying of a database
The Database Approach to Data Management

Database
Serves many business applications by centralizing data and controlling redundant data across division boundaries

Database management system (DBMS)
Interfaces between applications and physical data files
Separates logical and physical views of data
Solves problems of traditional file environment
  Controls redundancy
  Eliminates inconsistency
  Uncouples programs and data
  Enables organization to centrally manage data and data security
Although it is difficult to give a universally agreed definition of a database, we use the following common definition:

Definition:
A database is a collection of related, logically coherent data used by the application programs in an organization.
The American National Standards Institute/Standards Planning and Requirements Committee (ANSI/SPARC) has established a three-level architecture for a DBMS: 
internal, conceptual and external.
Database architecture
Advantages of databases

Comparing the flat-file system, we can mention several advantages for a database system.

Less redundancy

In a flat-file system there is a lot of redundancy. For example, in the flat file system for a university, the names of professors and students are stored in more than one file.

Inconsistency avoidance

If the same piece of information is stored in more than one place, then any changes in the data need to occur in all places that data is stored.
**Efficiency**

A database is usually more efficient than a flat file system, because a piece of information is stored in fewer locations.

**Data integrity**

In a database system it is easier to maintain data integrity (see Chapter 16), because a piece of data is stored in fewer locations.

**Confidentiality**

It is easier to maintain the confidentiality of the information if the storage of data is centralized in one location.
A database management system (DBMS) defines, creates and maintains a database. The DBMS also allows controlled access to data in the database. A DBMS is a combination of five components: hardware, software, data, users and procedures.
Hardware

The hardware is the physical computer system that allows access to data.

Software

The software is the actual program that allows users to access, maintain and update data. In addition, the software controls which user can access which parts of the data in the database.

Confidentiality

The data in a database is stored physically on the storage devices. In a database, data is a separate entity from the software that accesses it.
Users

In a DBMS, the term users has a broad meaning. We can divide users into two categories: end users and application programs.

Procedures

The last component of a DBMS is a set of procedures or rules that should be clearly defined and followed by the users of the database.
Database management system (DBMS)

• A specific type of software for creating, storing, organizing, and accessing data from a database
• Separates the logical and physical views of the data

• Logical view: how end users view data
• Physical view: how data are actually structured and organized

• Examples of DBMS: Microsoft Access, DB2, Oracle Database, Microsoft SQL Server, MYSQL
Database vs. Database Management System (DBMS)?
A single human resources database provides many different views of data, depending on the information requirements of the user. Illustrated here are two possible views, one of interest to a benefits specialist and one of interest to a member of the company’s payroll department.
DATA MODEL OVERVIEW

• ER-Model
• Hierarchical Model
• Network Model
• Inverted Model - ADABAS
• Relational Model
• Object-Oriented Model(s)
ER-Model

• Data Structures
• Integrity Constraints
• Operations

• The ER-Model is extremely successful as a database design model
• Translation algorithms to many data models
• Commercial database design tools, e.g., ERwin
• No generally accepted query language
• No database system is based on the model
ER-Model - Integrity Constraints

- Cardinality: 1:n for E1:E2 in R
- (min, max) participation of E2 in R
- Total participation of E2 in R
- Weak entity type E2; identifying relationship type R
- Key attribute: A
- Disjoint
- Exclusion
- Partition
Hierarchical database model

In the hierarchical model, data is organized as an inverted tree. Each entity has only one parent but can have several children. At the top of the hierarchy, there is one entity, which is called the root.

An example of the hierarchical model representing a university
Hierarchical Model - Data Structures

- **record types**: flight-schedule, flight-instance, etc.
- **field types**: flight#, date, customer#, etc.
- **parent-child relationship types** *(1:n only!!)*:
  (flight-sched, flight-inst), (flight-inst, customer)  
- **one** record type is the **root**, all other record types is a **child** of **one parent** record type **only**
- substantial **duplication** of customer instances  
- **asymmetrical model** of n:m relationship types
Hierarchical Model - Data Structures - virtual records

- duplication of customer instances avoided
- still asymmetrical model of n:m relationship types
Network database model

In the network model, the entities are organized in a graph, in which some entities can be accessed through several paths (Figure 14.4).

An example of the network model representing a university
Network Model - Data Structures

**Type diagram**
**Bachman Diagram**

- `flight-schedule`
  - `flight#`
  - `reservation`
    - `flight#`  
    - `date`
    - `customer#`
  - `customer`
    - `customer#`
    - `customer name`

**Occurrence diagram**
**The Spaghetti Model**

- `F1`  
  - `R1`  
  - `R2`
  - `R3`  
  - `R4`  
  - `R5`  
  - `R6`  
  - `C1`  
  - `C4`

- **owner record types**: `flight-schedule`, `customer`
- **member record type**: `reservations`
- **DBTG-set types**: `FR`, `CR`
- **n-m relationships cannot be modeled directly**
- **recursive relationships cannot be modeled directly**
Relational Model

- Data Structures
- Integrity Constraints
- Operations

- Commercial systems include: ORACLE, DB2, SYBASE, INFORMIX, INGRES, SQL Server
- Dominates the database market on all platforms
Relational database model

In the relational model, data is organized in two-dimensional tables called relations. The tables or relations are, however, related to each other, as we will see shortly.

An example of the relational model representing a university
Object-Oriented Model(s)

- based on the object-oriented paradigm, e.g., Simula, Smalltalk, C++, Java
- area is in a state of flux

- **object-oriented model** has object-oriented repository model; adds persistence and database capabilities; (see ODMG-93, ODL, OQL)
- **object-oriented** commercial systems include GemStone, Ontos, Orion-2, Statice, Versant, O2

- **object-relational model** has relational repository model; adds object-oriented features; (see SQL3)
- **object-relational** commercial systems include Starburst, POSTGRES
Object-Oriented Paradigm

- object class
- object attributes, primitive types, values
- object interface, methods; body, implementations
- encapsulation
- visible and hidden attributes and methods
- object instance; object constructor & destructor
- object identifier, immutable
- complex objects; multimedia objects; extensible type system
- subclasses; inheritance; multiple inheritance
- references represent relationships
- transient & persistent objects
Relational DBMS;
Represent data as two-dimensional tables called relations or files.

Each table contains data on entity and attributes
- **Table**: grid of columns and rows
- **Rows (tuples)**: Records for different entities
- **Fields (columns)**: Represents attribute for entity
- **Key field**: Field used to uniquely identify each record
- **Primary key**: Field in table used for key fields
- **Foreign key**: Primary key used in second table as look-up field to identify records from original table
A relational database organizes data in the form of two-dimensional tables. Illustrated here are tables for the entities SUPPLIER and PART showing how they represent each entity and its attributes. Supplier Number is a primary key for the SUPPLIER table and a foreign key for the PART table.
A relational database organizes data in the form of two-dimensional tables. Illustrated here are tables for the entities SUPPLIER and PART showing how they represent each entity and its attributes. Supplier Number is a primary key for the SUPPLIER table and a foreign key for the PART table.
Operations of a Relational DBMS

Three basic operations used to develop useful sets of data

**SELECT**: Creates subset of data of all records that meet stated criteria

**JOIN**: Combines relational tables to provide user with more information than available in individual tables

**PROJECT**: Creates subset of columns in table, creating tables with only the information specified
THE THREE BASIC OPERATIONS OF A RELATIONAL DBMS

The select, join, and project operations enable data from two different tables to be combined and only selected attributes to be displayed.
Relational Database Example

- Relational Database Management System (RDBMS)
  - Consists of a number of *tables* and single *schema* (definition of tables and attributes)
  - Students *(sid, name, login, age, gpa)*
    - *Students* identifies the table
    - *sid, name, login, age, gpa* identify attributes
    - *sid* is primary key
An Example Table

- Students (**sid**: string, **name**: string, **login**: string, **age**: integer, **gpa**: real)  

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>login</th>
<th>age</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>dave@cs</td>
<td>19</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>jones@cs</td>
<td>18</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>smith@ee</td>
<td>18</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>smith@math</td>
<td>19</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>madayan@music</td>
<td>11</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>guldu@music</td>
<td>12</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Another example: Courses

- Courses \((cid, \text{instructor}, \text{quarter}, \text{dept})\)  

<table>
<thead>
<tr>
<th>cid</th>
<th>instructor</th>
<th>quarter</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carnatic101</td>
<td>Jane</td>
<td>Fall 06</td>
<td>Music</td>
</tr>
<tr>
<td>Reggae203</td>
<td>Bob</td>
<td>Summer 06</td>
<td>Music</td>
</tr>
<tr>
<td>Topology101</td>
<td>Mary</td>
<td>Spring 06</td>
<td>Math</td>
</tr>
<tr>
<td>History105</td>
<td>Alice</td>
<td>Fall 06</td>
<td>History</td>
</tr>
</tbody>
</table>
Keys

- **Primary key** – minimal subset of fields that is unique identifier for a tuple
  - `sid` is primary key for Students
  - `cid` is primary key for Courses

- **Foreign key** – connections between tables
  - Courses (`cid`, instructor, quarter, dept)
  - Students (`sid`, name, login, age, gpa)
  - How do we express which students take each course?
Many to many relationships

- In general, need a new table
  Enrolled(cid, grade, studid)
  Studid is *foreign key* that references sid in Student table

<table>
<thead>
<tr>
<th>Enrolled</th>
<th>Foreign key</th>
</tr>
</thead>
<tbody>
<tr>
<td>cid</td>
<td>grade</td>
</tr>
<tr>
<td>Carnatic101</td>
<td>C</td>
</tr>
<tr>
<td>Reggae203</td>
<td>B</td>
</tr>
<tr>
<td>Topology112</td>
<td>A</td>
</tr>
<tr>
<td>History 105</td>
<td>B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
</tr>
<tr>
<td>50000</td>
</tr>
<tr>
<td>53666</td>
</tr>
<tr>
<td>53688</td>
</tr>
<tr>
<td>53650</td>
</tr>
<tr>
<td>53831</td>
</tr>
<tr>
<td>53832</td>
</tr>
</tbody>
</table>
Relational Algebra

• Collection of operators for specifying queries
• Query describes step-by-step procedure for computing answer (i.e., operational)
• Each operator accepts one or two relations as input and returns a relation as output
• Relational algebra expression composed of multiple operators
Basic operators

- Selection – return rows that meet some condition
- Projection – return column values
- Union
- Cross product
- Difference
- Other operators can be defined in terms of basic operators
Example Schema (simplified)

- Courses \((cid, \text{instructor}, \text{quarter}, \text{dept})\)
- Students \((sid, \text{name}, \text{gpa})\)
- Enrolled \((cid, \text{grade}, \text{studid})\)
Selection

Select students with gpa higher than 3.3 from S1:

$$\sigma_{gpa>3.3}(S1)$$

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
</tbody>
</table>
**Projection**

Project name and gpa of all students in S1:

\[ \Pi_{\text{name}, \text{gpa}}(S1) \]

<table>
<thead>
<tr>
<th>Sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Combine Selection and Projection

- Project name and gpa of students in S1 with gpa higher than 3.3:

\[ \Pi_{\text{name}, \text{gpa}}(\sigma_{\text{gpa}>3.3}(S1)) \]

<table>
<thead>
<tr>
<th>Sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

name  | gpa  |
-------|------|
Jones  | 3.4  |
Smith  | 3.8  |
Set Operations

• Union (R U S)
  – All tuples in R or S (or both)
  – R and S must have same number of fields
  – Corresponding fields must have same domains

• Intersection (R ∩ S)
  – All tuples in both R and S

• Set difference (R – S)
  – Tuples in R and not S
Set Operations (continued)

• Cross product or Cartesian product \((R \times S)\)
  – All fields in \(R\) followed by all fields in \(S\)
  – One tuple \((r, s)\) for each pair of tuples \(r \in R, s \in S\)
### Example: Intersection

**S1**

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

**S2**

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53700</td>
<td>Tom</td>
<td>3.5</td>
</tr>
<tr>
<td>53777</td>
<td>Jerry</td>
<td>2.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

\[ S1 \cap S2 = \]

<table>
<thead>
<tr>
<th>sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Joins

- Combine information from two or more tables
- Example: students enrolled in courses:

\[ S1 \bowtie_{S1.sid=E.studid} E \]

<table>
<thead>
<tr>
<th>Sid</th>
<th>name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>Dave</td>
<td>3.3</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53688</td>
<td>Smith</td>
<td>3.2</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>cid</th>
<th>grade</th>
<th>studid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carnatic101</td>
<td>C</td>
<td>53831</td>
</tr>
<tr>
<td>Reggae203</td>
<td>B</td>
<td>53832</td>
</tr>
<tr>
<td>Topology112</td>
<td>A</td>
<td>53650</td>
</tr>
<tr>
<td>History 105</td>
<td>B</td>
<td>53666</td>
</tr>
<tr>
<td>sid</td>
<td>name</td>
<td>gpa</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
<td>-----</td>
</tr>
<tr>
<td>53666</td>
<td>Jones</td>
<td>3.4</td>
</tr>
<tr>
<td>53650</td>
<td>Smith</td>
<td>3.8</td>
</tr>
<tr>
<td>53831</td>
<td>Madayan</td>
<td>1.8</td>
</tr>
<tr>
<td>53832</td>
<td>Guldu</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Object-Oriented DBMS (OODBMS)
- Stores data and procedures as objects
- Objects can be graphics, multimedia, Java applets
- Relatively slow compared with relational DBMS for processing large numbers of transactions
- Hybrid object-relational DBMS: Provide capabilities of both OODBMS and relational DBMS

Databases in the cloud
- Typically less functionality than on-premises DBs
- Amazon Web Services, Microsoft SQL Azure
Capabilities of Database Management Systems

**Data definition capability:** Specifies structure of database content, used to create tables and define characteristics of fields

**Data dictionary:** Automated or manual file storing definitions of data elements and their characteristics

**Data manipulation language:** Used to add, change, delete, retrieve data from database

**Structured Query Language (SQL)**

Microsoft Access user tools for generation SQL

Many DBMS have report generation capabilities for creating polished reports (Crystal Reports)
Microsoft Access has a rudimentary data dictionary capability that displays information about the size, format, and other characteristics of each field in a database. Displayed here is the information maintained in the SUPPLIER table. The small key icon to the left of SupplierNumber indicates that it is a key field.
EXAMPLE OF AN SQL QUERY

```
SELECT PART.Part_Number, PART.Part_Name, SUPPLIER.Supplier_Number, SUPPLIER.Supplier_Name
FROM PART, SUPPLIER
WHERE PART.Supplier_Number = SUPPLIER.Supplier_Number AND Part_Number = 137 OR Part_Number = 150;
```

This graphic shows an example SQL statement that would be used to retrieve data from a database. In this case, the SQL statement is retrieving records from the PART table illustrated on Slide 14 (Figure 6-5) whose Part Number is either 137 or 150.

Ask students to relate what each phrase of this statement is doing. (For example, the statement says to take the following columns: Part_Number, Part_Name, Supplier Number, Supplier Name, from the two tables Part and Supplier, when the following two conditions are true …)
Illustrated here is how the query in Figure 6-7 would be constructed using Microsoft Access query building tools. It shows the tables, fields, and selection criteria used for the query.
Designing Databases
  Conceptual (logical) design: Abstract model from business perspective
  Physical design: How database is arranged on direct-access storage devices

Design process identifies
  Relationships among data elements, redundant database elements
  Most efficient way to group data elements to meet business requirements, needs of application programs

Normalization
  Streamlining complex groupings of data to minimize redundant data elements and awkward many-to-many relationships

This slide describes activities involved in designing a database. To create an efficient database, you must know what the relationships are among the various data elements, the types of data that will be stored, and how the organization will need to manage the data. Note that the conceptual database design is concerned with how the data elements will be grouped, what data in what tables will make the most efficient organizations.
An unnormalized relation contains repeating groups. For example, there can be many parts and suppliers for each order. There is only a one-to-one correspondence between Order_Number and Order_Date.
An unnormalized relation contains repeating groups. For example, there can be many parts and suppliers for each order. There is only a one-to-one correspondence between Order_Number and Order_Date.
Entity-relationship diagram

Used by database designers to document the data model
Illustrates relationships between entities

A Simple Entity-Relationship Diagram

SUPPLIER — provides — PART

is supplied by
This diagram shows an example of an entity relationship diagram.

It shows that one ORDER can contain many LINE_ITEMs. (A PART can be ordered many times and appear many times as a line item in a single order.)

Each LINE ITEM can contain only one PART.

Each PART can have only one SUPPLIER, but many PARTs can be provided by the same SUPPLIER.
Distributing databases:
Storing database in more than one place

Partitioned: Separate locations store different parts of database

Replicated: Central database duplicated in entirety at different locations
There are alternative ways of distributing a database. The central database can be partitioned (a) so that each remote processor has the necessary data to serve its own local needs. The central database also can be replicated (b) at all remote locations.
Using Databases to Improve Business Performance and Decision Making

Very large databases and systems require special capabilities, tools
To analyze large quantities of data
To access data from multiple systems

*Three key techniques*
1. Data warehousing
2. Data mining
3. Tools for accessing internal databases through the Web
Data warehouse:
Stores current and historical data from many core operational transaction systems
Consolidates and standardizes information for use across enterprise, but data cannot be altered
Data warehouse system will provide query, analysis, and reporting tools

Data marts:
Subset of data warehouse
Summarized or highly focused portion of firm’s data for use by specific population of users
Typically focuses on single subject or line of business
The data warehouse extracts current and historical data from multiple operational systems inside the organization. These data are combined with data from external sources and reorganized into a central database designed for management reporting and analysis. The information directory provides users with information about the data available in the warehouse.
Business Intelligence (BI):
Tools for consolidating, analyzing, and providing access to vast amounts of data to help users make better business decisions

E.g., Harrah’s Entertainment analyzes customers to develop gambling profiles and identify most profitable customers

Principle tools include:

- Software for database query and reporting
- Online analytical processing (OLAP)
- Data mining
Online analytical processing (OLAP)

- Supports multidimensional data analysis
  - Viewing data using multiple dimensions
  - Each aspect of information (product, pricing, cost, region, time period) is different dimension

  E.g., how many washers sold in the East in June compared with other regions?

OLAP enables rapid, online answers to ad hoc queries
The view that is showing is product versus region. If you rotate the cube 90 degrees, the face that will show is product versus actual and projected sales. If you rotate the cube 90 degrees again, you will see region versus actual and projected sales. Other views are possible.
Using Databases to Improve Business Performance and Decision Making

Data mining:

More discovery driven than OLAP

Finds hidden patterns, relationships in large databases and infers rules to predict future behavior

E.g., Finding patterns in customer data for one-to-one marketing campaigns or to identify profitable customers.

Types of information obtainable from data mining

Associations
Sequences
Classification
Clustering
Forecasting
Predictive analysis

Uses data mining techniques, historical data, and assumptions about future conditions to predict outcomes of events

E.g., Probability a customer will respond to an offer
WHAT CAN BUSINESSES LEARN FROM TEXT MINING?

Text mining
Extracts key elements from large unstructured data sets
(e.g., stored e-mails)

What challenges does the increase in unstructured data present for businesses?

How does text-mining improve decision-making?

What kinds of companies are most likely to benefit from text mining software?

In what ways could text mining potentially lead to the erosion of personal information privacy?
Web mining
Discovery and analysis of useful patterns and information from WWW

E.g., to understand customer behavior, evaluate effectiveness of Web site, etc.

Web content mining
Knowledge extracted from content of Web pages

Web structure mining
E.g., links to and from Web page

Web usage mining
User interaction data recorded by Web server
Databases and the Web

Many companies use Web to make some internal databases available to customers or partners.

Typical configuration includes:
- Web server
- Application server/middleware/CGI scripts
- Database server (hosting DBM)

Advantages of using Web for database access:
- Ease of use of browser software
- Web interface requires few or no changes to database
- Inexpensive to add Web interface to system
Firms use the Web to make information from their internal databases available to customers and partners
• Middleware and other software make this possible
• Database servers
• CGI
• Web interfaces provide familiarity to users and savings over redesigning and rebuilding legacy systems
Users access an organization’s internal database through the Web using their desktop PCs and Web browser software.
Managing Data Resources

Establishing an information policy
  Firm’s rules, procedures, roles for sharing, managing, standardizing data

Data administration:
  Firm function responsible for specific policies and procedures to manage data

Data governance:
  Policies and processes for managing availability, usability, integrity, and security of enterprise data, especially as it relates to government regulations

Database administration:
  Defining, organizing, implementing, maintaining database; performed by database design and management group
Ensuring data quality
More than 25% of critical data in Fortune 1000 company databases are inaccurate or incomplete
Most data quality problems stem from faulty input

Before new database in place, need to:
Identify and correct faulty data
Establish better routines for editing data once database in operation
Data quality audit:
Structured survey of the accuracy and level of completeness of the data in an information system
Survey samples from data files, or
Survey end users for perceptions of quality

Data cleansing
Software to detect and correct data that are incorrect, incomplete, improperly formatted, or redundant
Enforces consistency among different sets of data from separate information systems
Assess the business impact of credit bureaus’ data quality problems for the credit bureaus, for lenders, for individuals.

Are any ethical issues raised by credit bureaus’ data quality problems?

Analyze the people, organization, and technology factors responsible for credit bureaus’ data quality problems.

What can be done to solve these problems?
Describe how a relational database organizes data and compare its approach to an object oriented database.

- Identify and describe the principles of a database management system.

- Evaluate tools and technologies for providing information from databases to improve business performance and decision making.