The Unit Overview

• Introduction to the theory and practice of RDBMS
  – We will take the applications emphasis covering
    • The relational data model
    • Relational algebra and calculus
    • Data modelling
    • SQL, mySQL
      – Advanced topic
        » Constraints and Triggers
        » Transactions
    • Database-backed applications with Java
    • Normalization
    • Data mining
Why Use a DBMS

• DBMS enable:
  – Data Independence
  – Efficiency
  – Data Integrity
  – Data Administration
  – Concurrency Control
  – Application Development
Relational Data Models #1

• Based on Set theory
  – Relational terminology
    • In particular, queries can be expressed with absolute precision using either
      – The relational algebra
        » A procedural language expressing how to extract information from the database.
      – The relational calculus
        » A declarative language expressing what information should be extracted from the database.

• Sets and Relations
  – Sets
    • A set is simply a collection of objects, and is usually denoted by a capital letter such as A.
  – Membership
    • whether an object belongs to the set or not.
  – Defining Properties
  – Operators
    • Union
    • Intersection
    • Cartesian Product
    • Etc.
  – Relations
    • R: \( X \times Y = \{(x,y) \mid x \text{ is elem of } X \text{ and } y \text{ is elem of } Y\} \)
    • A sets of tuple \((x,y)\).
      – Cardinality
        » one-to-one
        » One-to-many, many-to-one, many-to-many
Relational Data Models #2

• Entity Sets
  – An entity is an object or event in — is described using a set of attributes.
  – An entity set is a collection of similar objects in that every entity in the set has the same attributes.
Relational Data Models #3

• data model
  – A data model is a precise, conceptual description of the data stored in a database.
  – The relational data model is a data model where the data takes the form of relations.
    • A relation is made up of tuples.
      – We write a tuple as comma separated values within parentheses, e.g.
      – (Wayne’s World, 1992, 95, comedy, Paramount)
    • Mathematically a relation is a set of tuples.

• The relation we have been using could have the schema:
  – Movies(title, year, length, genre, company)

• Important Terms
  – relational data model
  – tuple
  – component of a tuple
  – data type of a component
  – attribute
  – relation
  – schema
  – instance relation
An abstract view

The abstract structure of a modern DBMS is a 3-level architecture as follows:

```
  External Schema 1  -->  Conceptual Schema  
                       /                  
                      /                   
  External Schema 2    /                    
                      /                   
  External Schema 3  -->  Conceptual Schema
```

Physical Schema

Storage
Entity-Relationship Modelling #2

• Database Design Process
  – six main steps in designing a database
    • Requirements Analysis
    • Conceptual Design
    • Logical Design
    • Schema Refinement
    • Physical Design
    • Application & Security Design
SQL #1

• Overview
  – Data Definition Language (DDL)
    • Creating, deleting and altering tables with CREATE TABLE, DROP TABLE and ALTER TABLE.
    • Working with views.
  – Data Manipulation Language (DML)
    • Inserting, updating and deleting rows with INSERT, UPDATE and DELETE.
    • Querying the database with SELECT combining one or more tables.
    • Summarizing data with aggregate functions and GROUP BY.
  – Advanced functions
    • Stored procedures, functions and triggers.
    • Transactions

• The process of implementing an ERD as an actual relational database.
  – converting the various entity sets and relationship sets into tables (i.e. relations) in such a way that the DBMS can check and enforce the participation and key constraints.
  – encoding relationship sets into tables taking into account
    • participation
    • key constraints.
SQL #2

• Stored Routines
  – Stored Routines
    • A stored routine is a named set of SQL statements that is stored on
      the server and which can be initiated by a single call.
    • Stored routines are further subdivided into procedures which do
      not return anything (although they can assign values to variables)
      and functions that return values to the caller.
  – Rationale for stored routines
    • A stored routine is maintained on the server which has various
      consequences both positive and negative:
      – A complex sequence of SQL statements can be prepared once by a
        professional DBA and then made available to all client programs
      – Stored routines can access confidential or sensitive tables without
        exposing them to client programs
      – Processing becomes more centralized with the server taking on a greater
        computational load
Relational Algebra

- formal language underlying the manipulation of relations.
  - Relational algebra is a procedural language that allows us to describe operations on relations in a formal and mathematically precise.
  - An expression in relational algebra describes a sequence of operations that can be applied to a relation and which produces a relation as a result.
  - The primary operations of the relational algebra are projection, selection and joins.

- Binary Operators
  - Relational algebra permits the use of the standard set operations:
    - Union (∪)
    - Intersection (∩)
    - Set Difference (−)
    - Cartesian Product (×)
JDBC: Application Layer Connectivity

• Focus on Java-based application program connects to a database using JDBC.

• Java DataBase Connectivity (JDBC) Architecture
  – A JDBC application can be viewed as consisting of four components
    • The Java application program
      – The application initiates a connection with the server, submits SQL statements and processes the results.
    • The driver manager
      – The driver manager loads JDBC drivers and passes the JDBC calls to the correct driver.
    • The drivers
      – A driver is database-specific code that translates JDBC calls into the right form for its particular database, and translates the results back into the JDBC standard.
    • back into the JDBC standard.
      – The data sources: The underlying database or databases.
JDBC Basics

• A JDBC application must perform the following steps:
  – Load the driver class for the particular database being used.
  – Open a connection to the database.
  – Create SQL queries and use the connection to execute them.
  – Process the results obtained from the database.
This lecture introduces normal forms, decomposition and normalization.

Eliminate data redundancy via decomposition:

Problems with redundancy

- Apart from unnecessary storage, redundancy leads to some more significant problems:
  - Update Anomalies
  - If one copy of a data item is updated — for example, a student changes his or her name, then the database becomes inconsistent unless every copy is updated.
  - Insertion Anomalies
  - A new data item, for example a new mark for a student, cannot be entered without adding some other, potentially unnecessary, information such as the student’s name.
  - Deletion Anomalies
  - It may not be possible to delete some data without losing other, unrelated data, as well (an example is on the next slide).

Problems with decomposition

- Lossless-join decomposition
- Lossy decomposition
- queries over the decomposed schemas may require computationally expensive joins

Informed decisions about whether to decompose or not requires

- a formal understanding about the types of redundancy and which can be resolved through decomposition — this is the theory of functional dependencies.
Normalisation #2

• Functional dependencies
  – an FD is a generalization of the concept of a key in a relation.
  – Keys
    • The obvious functional dependencies come from the keys of a relation.
    • Superkeys
      – A key is a minimal set of attributes that determines all of the remaining attributes of a relation.
Types of Normal Forms #1

• Principle
  – Avoiding certain types of redundancy problems by eliminating a certain types of FDs.
    • each type of normal form determines under what circumstances a schema can be decomposed in such a way that the resulting relations are in the desired normal form ensuring
      – Lossless and dependency preserving decompositions

• Normal form hierarchy
  – 1NF -- First normal form
    • A relation is in 1NF if and only if attribute values are atomic.
  – 2NF -- Second normal form
    • A relation is in 2NF if it is in 1NF and contains no partial dependencies.
      – A partial functional dependency exists when a non-key attribute is functionally dependent on part of, but not all of, the primary key.
Types of Normal Forms #2

- 3NF -- Third normal form
  - A relation is in 3NF if it is in 2NF and contains **no** transitive dependencies.
    - A transitive functional dependency in a relation is a functional dependency between two or more nonkey attributes.

- BCNF -- Boyce-Codd normal form
  - A relational schema is in BCNF if for every functional dependency $X \rightarrow A$ (where $X$ is a subset of the attributes and $A$ is a single attribute) either $A \rightarrow X$, or $X$ is a superkey.
    - the only FD are either the trivial ones or ones based on the keys of the relation.
    - If a relational schema is in BCNF, then there is no redundancy within the relations.
Normalisation Algorithms

- **Iterative decomposition algorithm**
  - BCNF

- **The Synthesis Algorithm**
  - Given a relation $R$ and a set of functional dependencies $X \rightarrow Y$ where $X$ and $Y$ are subsets of the attributes of $R$.
    - 1. Compute the closures of the FDs.
    - 2. Use (1) to find all the candidate keys of the relation $R$.
    - 3. Find the minimal cover of the relation $R$.
    - 4. Use (3) to produce $R_i$ decompositions in 3NF.
    - 5. If no $R_i$ is a super key of $R$, add a relation containing the key.
Exercise

Consider $R = \{A, B, C, D, E, G\}$ and the following FDs

- $AB \rightarrow C$
- $C \rightarrow A$
- $BC \rightarrow D$
- $ACD \rightarrow B$
- $D \rightarrow EG$
- $BE \rightarrow C$
- $CG \rightarrow BD$
- $CE \rightarrow AG$

- $(AB)^+ = ABCDEG$
- $(C)^+ = AC$
- $(BC)^+ = ABCDEG$
- $(ACD)^+ = ABCDEG$
- $(D)^+ = DEG$
- $(BE)^+ = ABCDEG$
- $(CG)^+ = ABCDEG$
- $(CE)^+ = ABCDEG$

- $AB$ is a key
- $BC$ is a key
- $ACD$ is a super key
- $BE$ is a key
- $CG$ is a key
- $CE$ is a key

Applying Armstrong’s Axioms gives $BD$ and $CD$ are also keys.

Manual normalization

Data typically employed in a business is usually presented as below.

<table>
<thead>
<tr>
<th>Order_ID</th>
<th>Order Date</th>
<th>CustomerID</th>
<th>Customer Name</th>
<th>Customer Address</th>
<th>Product ID</th>
<th>Product Description</th>
<th>Material</th>
<th>Unit Price</th>
<th>Ordered Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234</td>
<td>04/05/2010</td>
<td>2</td>
<td>FurnCo</td>
<td>35 S St, NY</td>
<td>21</td>
<td>Sofa</td>
<td>Leather</td>
<td>2500.00</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td>TV Bench</td>
<td>Pine</td>
<td>300.00</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>17</td>
<td>Chair</td>
<td>Mahogany</td>
<td>500.00</td>
<td>6</td>
</tr>
<tr>
<td>6789</td>
<td>05/05/2010</td>
<td>9</td>
<td>CheapAs</td>
<td>35 N St, LA</td>
<td>17</td>
<td>Chair</td>
<td>Mahogany</td>
<td>500.00</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>Desk</td>
<td>Particle</td>
<td>650.00</td>
<td>1</td>
</tr>
</tbody>
</table>

However to be useful in a database the data needs to be normalized.
Databases - Constraints and Triggers

• SQL allows expression or statement that is stored in the database and executed at the appropriate “time”.
  – The time may be when an insertion, update or deletion on a relation is attempted, or equally if changes result in a boolean value being TRUE.
    • See PRIMARY/FOREIGN

• CHECK constraints - attribute
  – Constraints can be attached to an attribute.
    • The constraint is checked only when changes occur to the attribute to which the constraint is associated. It is possible for this constraint to be violated if other values involved in the constraint are changed.

• More powerful active elements
  – An assertion is a boolean-valued SQL expression that is always true.
    • The DBMS must determine if modifications affect an assertion’s truth - declarative
  – A trigger is a series of actions associated with certain events.
    • Triggers specifically identify what the DBMS needs to do – procedural
Databases - Transactions

• Definition
  – Transaction is any one execution of a user program.
  • consists of a number of statements that read and write database objects (i.e. tables, values etc).

• Transaction Properties
  – database engine must ensure that the following four properties (ACID) are maintained.
    • Atomicity
      – transactions are atomic if the system ensures that they cannot be “half-done”
    • Consistency
      – Transactions must preserve the consistency of the database.
    • Isolation
      – the user of the DB should be able to execute a transaction without regard for concurrently executing transactions.
    • Durability
      – once the user is informed of the successful completion of a transaction, then its effects on the database are persistent.
Interleaving

- There would be no problem with isolation if the DBMS were able to simply run each transaction to completion at a time before starting the next one.
- Interleaving Anomalies
  - Dirty Reads
    - occurs when one transaction reads a database value that has been altered by a transaction that has not yet committed.
  - Unrepeatable Reads
    - is essentially the dirty-read problem in reverse order
      - a value gets changed by another transaction after it has been read, rather than before.
  - Phantoms
    - is a variant of the unrepeatable read problem that occurs when one transaction performs a SELECT statement with some selection criteria, and then subsequently another transaction inserts a new row.
Schedules

• A schedule an interleaved statements is:
  – serializable if its effect on any consistent database instance is equivalent to running the transactions in some serial order.
  – recoverable if a transaction T1 that reads values changed by T2 only commits after T2 commits.

• DBMS is to ensure that the only allowed schedules are serializable and recoverable.
  – Achieved via the use of locking protocols.
    • A locking protocol is a set of rules that determine what types of lock to use in particular situations.
Strict Two-Phase Locking

- The Strict Two-Phase Locking uses two rules/policies:
  - A transaction that wishes to read an object first requests a shared lock on that object,
  - A transaction that wishes to modify an object first requests an exclusive lock on that object.
  - All locks held by a transaction are released when the transaction completes (commits or aborts).

- SQL support
Data Mining

• Is a Knowledge Discovery in Databases (KDD) method
  – the general process of trying to extract interesting or useful patterns from a (usually huge) dataset.
• We introduced data-mining via the market-basket analysis technique
  – A market basket analysis use of historical data to infer say purchasing behaviour of the customer.
• Approach
  – 1. Defining a frequent itemset with a specified minimum support
  – 2. Find association rules with a specified minimum confidence
  – 3. Finding the strongest inference we can make about a considered behaviour.