CS9152 – DATABASE TECHNOLOGY

UNIT – III

EMERGING SYSTEMS

TEXT BOOK

1. Elisa Bertino, Barbara Catania, Gian Piero Zarri, "Intelligent Database Systems", Addison-Wesley, 2001.

REFERENCES

1. Carlo Zaniolo, Stefano Ceri, Christos Faloustsos, R.T.Snodgrass, V.S.Subrahmanian, "Advanced Database Systems", Morgan Kaufman, 1997.
2. N.Tamer Ozsu, Patrick Valduriez, "Principles of Distributed Database Systems", Prentice Hal International Inc. , 1999.
3. C.S.R Prabhu, "Object-Oriented Database Systems", Prentice Hall Of India, 1998.
4. Abdullah Uz Tansel Et Al, "Temporal Databases: Theory, Design And Principles",Benjamin Cummings Publishers , 1993.
5. Raghu Ramakrishnan, Johannes Gehrke, "Database Management Systems", Mcgraw Hill, Third Edition, 2004.
6. Henry F Korth, Abraham Silberschatz, S. Sudharshan, "Database System Concepts", Fourth Ediion, McGraw Hill , 2002.
7. R. Elmasri, S.B. Navathe, "Fundamentals of Database Systems", Pearson Education, 2004.

Syllabus:

UNIT III EMERGING SYSTEMS 10

Enhanced Data Models – Client/Server Model – Data Warehousing and Data Mining –Web Databases – Mobile Databases.

Table of Contents

Topic – 1: Introduction to Enhanced Data Models

Motivation:

The Enhanced-ER (EER) model includes additional concepts included in ER model and are:
Category or Union type
Specialization
Generalization
Inheritance

Enhanced-ER (EER) Model Concepts or Formal definitions for EER model

Class - It is a collection of entities
Category or Union type – It is used to represent a collection of objects that is the union of objects of different entity types.

Superclass – A set of subclasses of an entity type (super class)

subclass - A subclass S is a class whose entities must always be a subset of the entities in another class called the super class C of the super class- superclass (IS-A) relationship.

Superclass / subclass relationship or class /sublclass relationship - A relationship between the superclass and any of its subclasses.

Inheritance – A set of fields or attributes of a subclass that ***inherits*** the all the attributes of the entity as a member of the superclass.

Specialization – process of defining a set of subclasses of an entity type (super class) or process of defining a set of a subclasses of an entity type and is called superclass.

Example: The set of subclasses (SECRETARY, ENGINEER, TECN
Generalization – process of defining a generalized entity type from the given entity types.

IS-AN-INSTANCE-OF relationship (Classification & Instantiation)
IS-A-SUBCLASS-OF relationship (Specialization & Generalization)
IS-A-PART-OF / IS-A-COMPONENT-OF relationship (Aggregation & Association)

Functional Data Models (FDMs):

- Use the concept of mathematical function as their fundamental modeling construct
- Function call with arguments

• Main modeling primitives:
• Entities
• Functional relationships

• **Nested Relational Data Model:**
• Removes the restriction of 1NF
• Non-1NF or N1NF relational model
• Allows composite and multivalued attributes, thus leading to complex tuples.

Semantic Data Model (SDM):
• Uses the concepts of classes and subclasses into data modeling
• Abstraction class
• Aggregate class
• **Structural Data Model:**
• Extends the relational model with additional constraints and semantics
• Structures used:
• Relations
• Primary Relation
• Referenced Relation
• connections

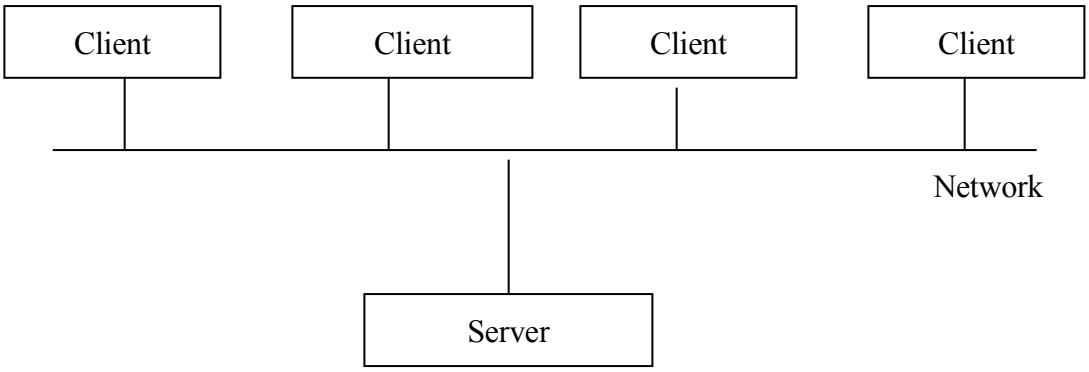
Topic – 2: Client/Server Model

Centralized Systems

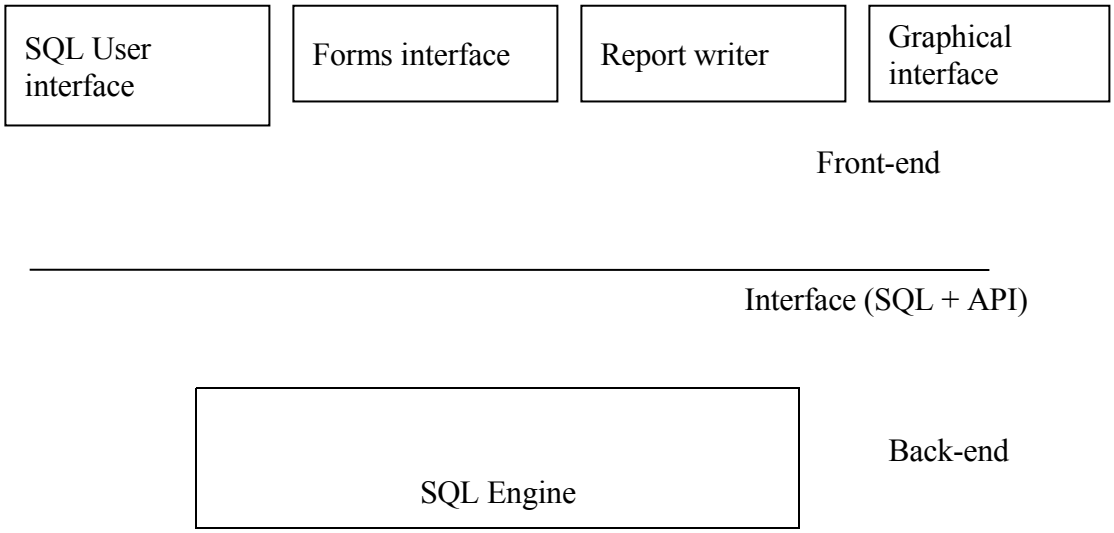
Run on a single computer system and do not interact with other computer systems.
☐ General-purpose computer system: one to a few CPUs and a number of device controllers that are connected through a common bus that provides access to shared memory.
☐ Single-user system (e.g., personal computer or workstation):
desk-top unit, single user, usually has only one CPU and one or two hard disks; the OS may support only one user.
☐ Multi-user system: more disks, more memory, multiple CPUs, and a multi-user OS. Serve a large number of users who are connected to the system vie terminals. Often called *server* systems.

Client-Server Systems

Server systems satisfy requests generated at *m* client systems, whose general structure is shown below:



Database functionality can be divided into:
☐ **Back-end**: manages access structures, query evaluation and optimization, concurrency control and recovery.
☐ **Front-end**: consists of tools such as *forms*, *report-writers*, and graphical user interface facilities.
☐ The interface between the front-end and the back-end is through SQL or through an application program interface.



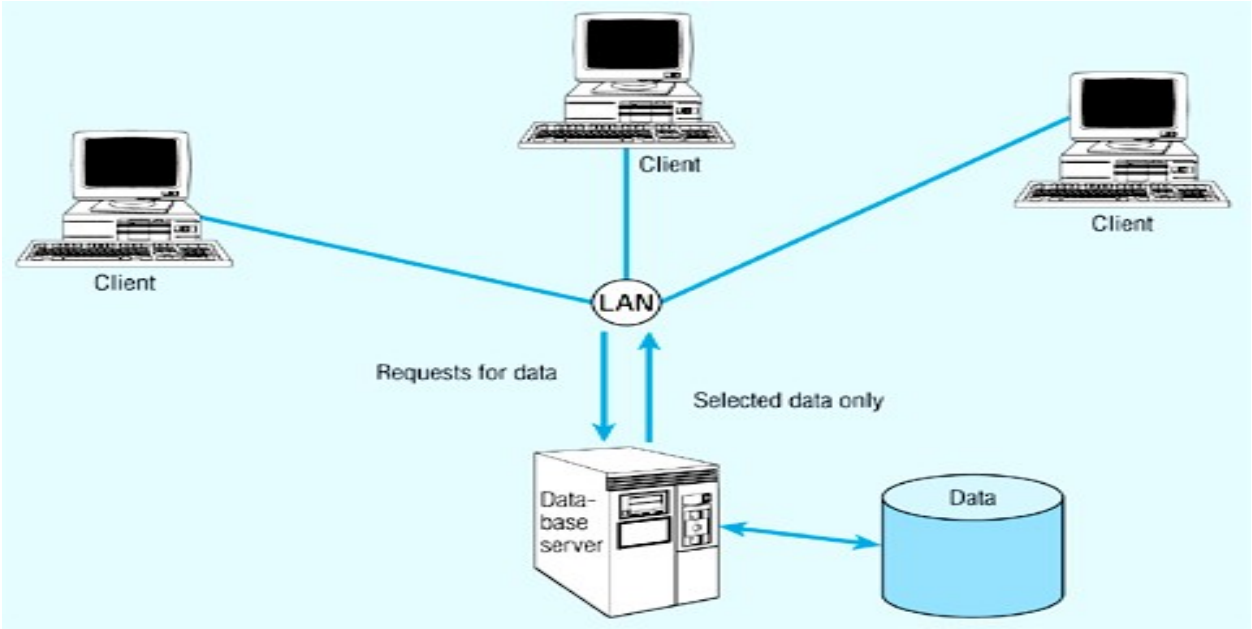
Advantages of replacing mainframes with networks ofworkstations or personal computers connected to back-end server machines:
☐ better functionality for the cost
☐ flexibility in locating resources and expanding facilities
☐ better user interfaces
☐ easier maintenance

☐ Server systems can be broadly categorized into two kinds:
☐ **transaction servers** which are widely used in relational database systems, and
☐ **data servers**, used in object-oriented database systems

- Networked computing model
- Processes distributed between clients and servers
- Client – Workstation (usually a PC) that requests and uses a service
- Server – Computer (PC/mini/mainframe) that provides a service
- For DBMS, server is a database server

Database Server Architectures

- 2-tiered approach
- Client is responsible for
  - I/O processing logic
  - Some business rules logic
- Server performs all data storage and access processing ➔ DBMS is only on server
- Advantages
  - Clients do not have to be as powerful
  - Greatly reduces data traffic on the network
  - Improved data integrity since it is all processed centrally
  - Stored procedures ➔ some business rules done on server



Three-Tier Architectures

Three layers:

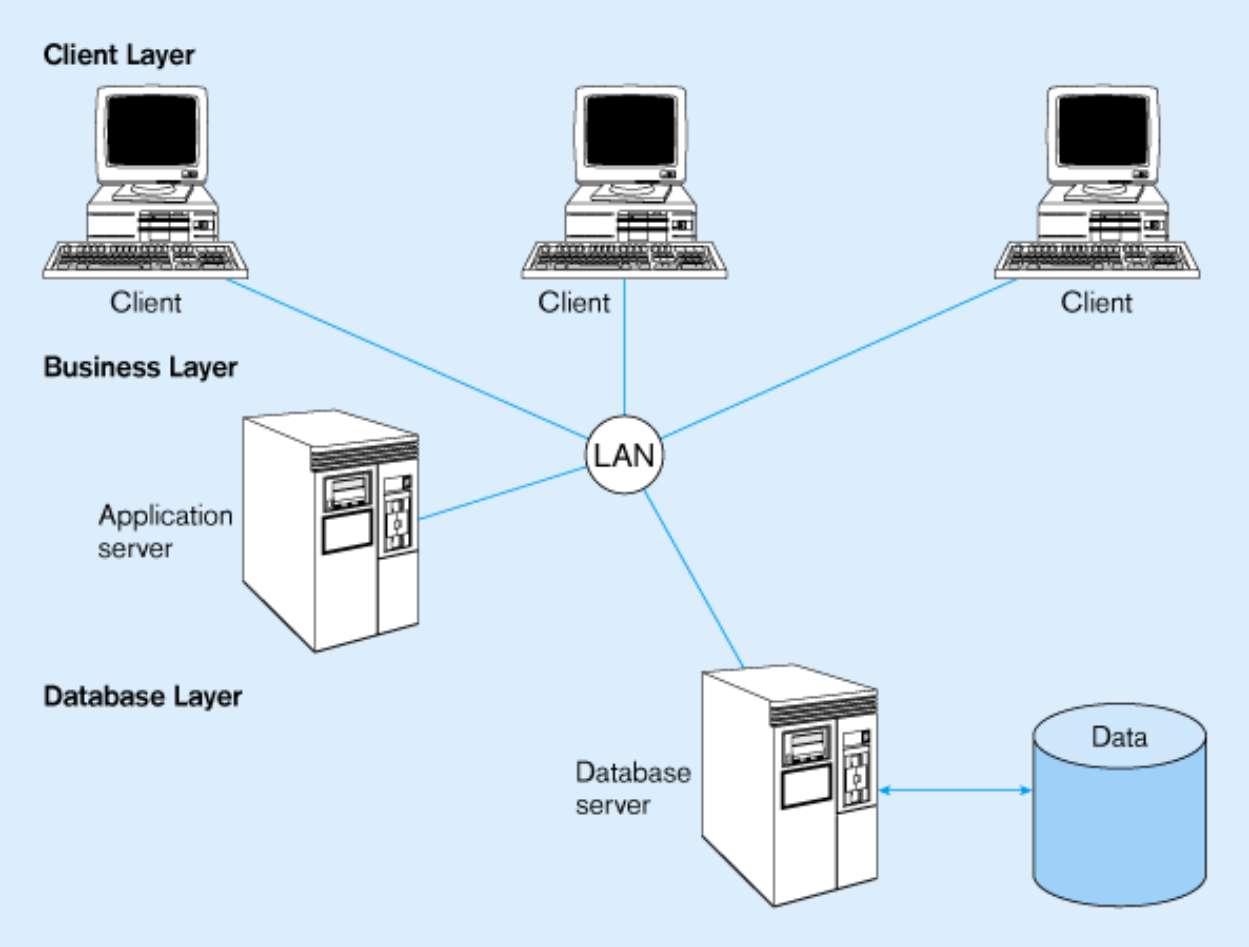
Client GUI interface *Browser*
(I/O processing)

Application server Business rules *Web Server*

Database server Data storage *DBMS*

- *Thin Client*
  - PC just for user interface and a little application processing. Limited or no data storage (sometimes no hard drive)

Three-tier architecture



Advantages of Three-Tier Architectures

- Scalability
- Technological flexibility
- Long-term cost reduction
- Better match of systems to business needs
- Improved customer service

- Competitive advantage
- Reduced risk

Challenges of Three-tier Architectures

- High short-term costs
- Tools and training
- Experience
- Incompatible standards
- Lack of compatible end-user tools

Client/Server Security

- Network environment ➔ complex security issues
- Security levels:
  - System-level password security
    - for allowing access to the system
  - Database-level password security
    - for determining access privileges to tables; read/update/insert/delete privileges
  - Secure client/server communication
    - via encryption

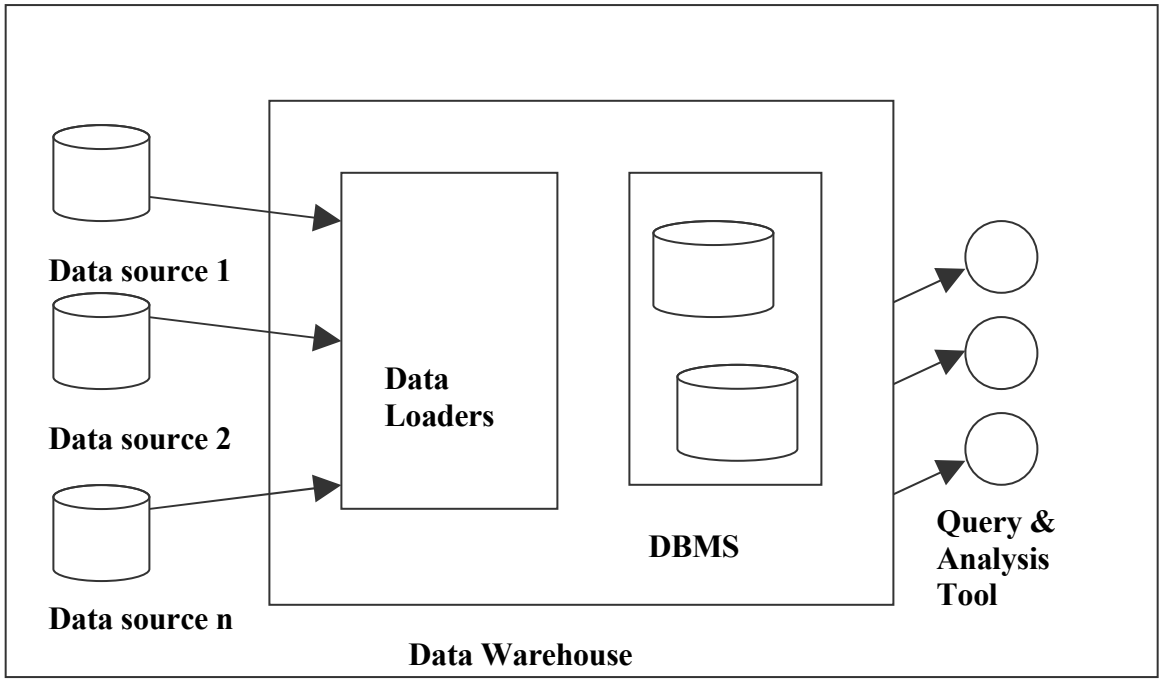
Topic – 3: Data Warehousing and Data Mining

DATA WAREHOUSING

Data Warehouse:

• Repository of information collected from multiple sources, stored under a unified schema, and which usually resides at a single site.

• *Subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management's decision making process.*



Components of Data Warehouse

- *When and how to gather data*
  - Source driven architecture: data sources transmit new information to warehouse, either continuously or periodically (e.g. at night)
  - Destination driven architecture: warehouse periodically requests new information from data sources
  - Keeping warehouse exactly synchronized with data sources (e.g. using two-phase commit) is too expensive
  - Usually OK to have slightly out-of-date data at warehouse
  - Data/updates are periodically downloaded form online transaction processing (OLTP) systems.

- *What schema to use*
  - Schema integration

- *Data cleansing*
  - E.g. correct mistakes in addresses
  - E.g. misspellings, zip code errors
  - Merge address lists from different sources and purge duplicates
  - Keep only one address record per household ("householding")

- *How to propagate updates*
  - Warehouse schema may be a (materialized) view of schema from data sources

- Efficient techniques for update of materialized views

- *What data to summarize*
  - Raw data may be too large to store on-line
  - Aggregate values (totals/subtotals) often suffice
  - Queries on raw data can often be transformed by query optimizer to use aggregate values

Functions:
• Data cleaning
• Data transformation
• Data integration
• Data loading &
• Periodic data refreshing
Multidimensional database structure
Physical structure:
relational data store / multidimensional data cube Data Warehousing

- Data sources often store only current data, not historical data
- Corporate decision making requires a unified view of all organizational data, including historical data
- A data warehouse is a repository (archive) of information gathered from multiple sources, stored under a unified schema, at a single site
  - Greatly simplifies querying, permits study of historical trends
  - Shifts decision support query load away from transaction processing systems

Database Vs Data Warehouse:
Operational Database
• Online transaction & query processing
• OLTP systems
• Day-to-day operations

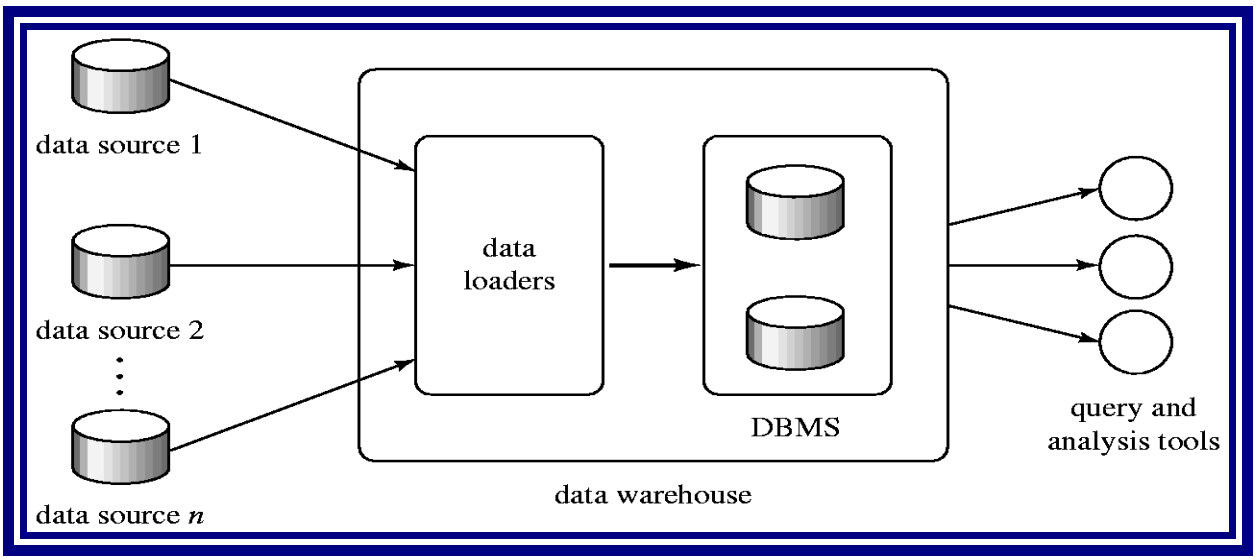
Data Warehouse
Data analysis & decision making
OLAP systems

Data Warehouse Vs Data Mart

Data Warehouse
Entire organization suited for

On-Line Analytical
Processing, or OLAP

Data Mart
Department, subset of a data
warehouse
Scope->department-wide



Steps for designing a warehouse :

•Choose a ***business process*** to model
(eg) orders, sales, shipments
•Choose the ***grain*** of the business process
(eg) individual transactions, individual snapshots etc.
•Choose the ***dimensions*** that will apply to each fact table record
(eg) time, item, customer, supplier
•Choose the ***measures*** that will populate each fact table record
(eg) numeric quantities like dollars-cold, units-sold

Design Issues

- *When and how to gather data*
  - Source driven architecture: data sources transmit new information to warehouse, either continuously or periodically (e.g. at night)

- Destination driven architecture: warehouse periodically requests new information from data sources
- Keeping warehouse exactly synchronized with data sources (e.g. using two-phase commit) is too expensive
  - Usually OK to have slightly out-of-date data at warehouse
  - Data/updates are periodically downloaded form online transaction processing (OLTP) systems.

- *What schema to use*
  - Schema integration

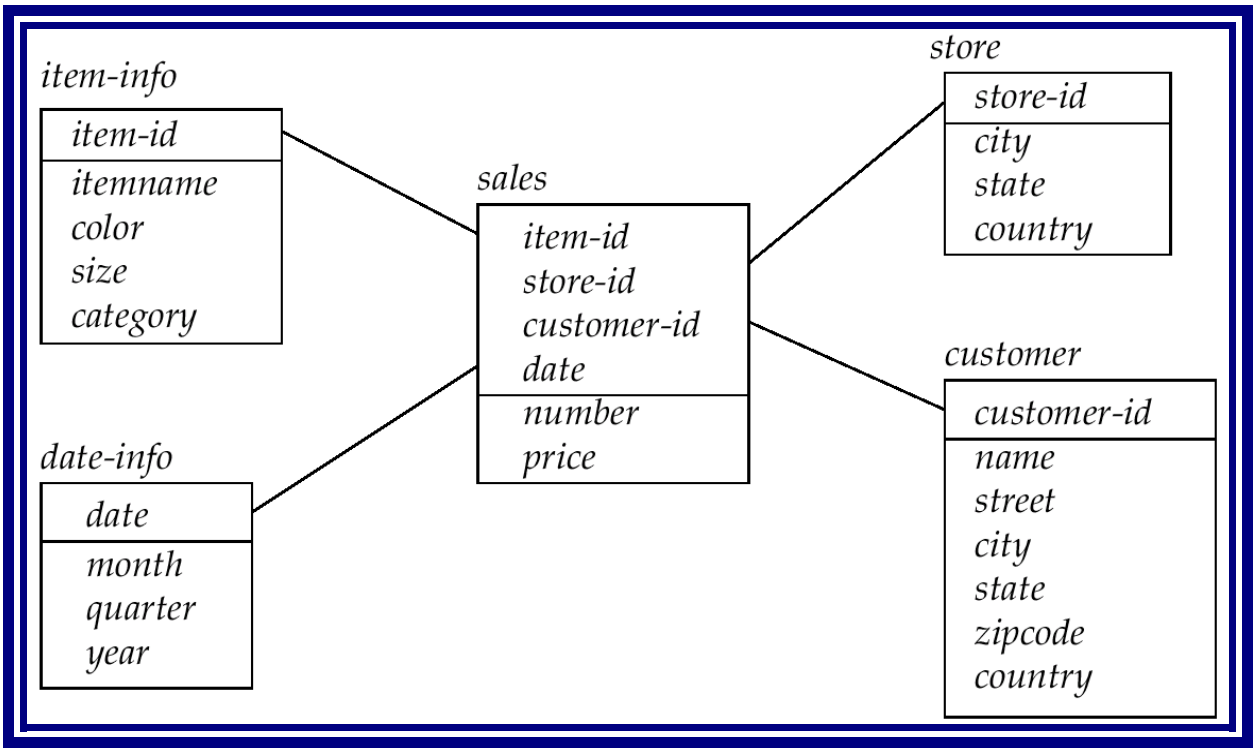
More Warehouse Design Issues

- *Data cleansing*
  - E.g. correct mistakes in addresses (misspellings, zip code errors)
  - Merge address lists from different sources and purge duplicates
- *How to propagate updates*
  - Warehouse schema may be a (materialized) view of schema from data sources
- *What data to summarize*
  - Raw data may be too large to store on-line
  - Aggregate values (totals/subtotals) often suffice
  - Queries on raw data can often be transformed by query optimizer to use aggregate values

Warehouse Schemas

- Dimension values are usually encoded using small integers and mapped to full values via dimension tables
- Resultant schema is called a **star schema**
  - More complicated schema structures
    - Snowflake schema: multiple levels of dimension tables
    - Constellation: multiple fact tables

Data Warehouse Schema



Data Mining

What Is Data Mining?

• Data mining (knowledge discovery in databases):
• Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) information or patterns from data in large databases
• Alternative names:
• Knowledge discovery(mining) in databases (KDD), knowledge extraction, knowledge mining from data, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.
• What is not data mining?
• (Deductive) query processing.
• Expert systems or small statistical programs

- Data mining is the process of semi-automatically analyzing large databases to find useful patterns
- **Prediction** based on past history
  - Predict if a credit card applicant poses a good credit risk, based on some attributes (income, job type, age, ..) and past history

- Predict if a pattern of phone calling card usage is likely to be fraudulent

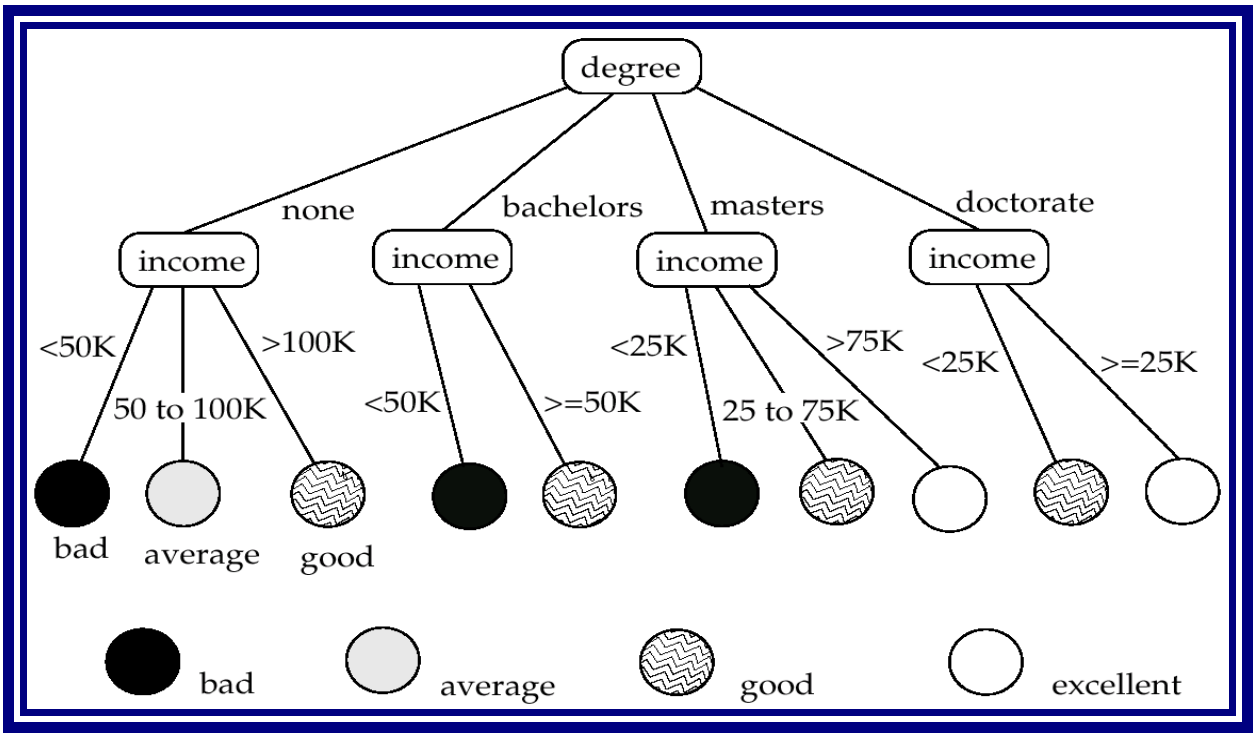
- Some examples of prediction mechanisms:
  - **Classification**
    - Given a new item whose class is unknown, predict to which class it belongs
  - **Regression** formulae
    - Given a set of mappings for an unknown function, predict the function result for a new parameter value

- **Descriptive Patterns**
  - **Associations**
    - Find books that are often bought by "similar" customers. If a new such customer buys one such book, suggest the others too.
  - Associations may be used as a first step in detecting **causation**
    - E.g. association between exposure to chemical X and cancer,
  - **Clusters**
    - E.g. typhoid cases were clustered in an area surrounding a contaminated well
    - Detection of clusters remains important in detecting epidemics

Classification Rules

- Classification rules help assign new objects to classes.
  - E.g., given a new automobile insurance applicant, should he or she be classified as low risk, medium risk or high risk?
- Classification rules for above example could use a variety of data, such as educational level, salary, age, etc.
  - $\forall$ person P, P.degree = masters and P.income > 75,000
    $\Rightarrow$ P.credit = excellent
  - $\forall$ person P, P.degree = bachelors and
    (P.income $\geq$ 25,000 and P.income $\leq$ 75,000)
    $\Rightarrow$P.credit = good
- Rules are not necessarily exact: there may be some misclassifications
- Classification rules can be shown compactly as a decision tree**.**

Decision Tree



Construction of Decision Trees

- Training set: a data sample in which the classification is already known.
- Greedy top down generation of decision trees.
  - Each internal node of the tree partitions the data into groups based on a partitioning attribute, and a partitioning condition for the node
  - Leaf node:
    - all (or most) of the items at the node belong to the same class, or
    - all attributes have been considered, and no further partitioning is possible.

Best Splits

- Pick best attributes and conditions on which to partition
- The purity of a set S of training instances can be measured quantitatively in several ways.
  - Notation: number of classes = k, number of instances = |S|, fraction of instances in class $i = p_i$.
- The Gini measure of purity is defined as
  - Gini (S) = 1 - $\sum$
  - When all instances are in a single class, the Gini value is 0
  - It reaches its maximum (of 1 –1 /k) if each class the same number of instances.

- Another measure of purity is the entropy measure, which is defined as
  - entropy (S) = – $\sum$

- When a set S is split into multiple sets Si, I=1, 2, …, r, we can measure the purity of the resultant set of sets as:
  - purity($S_1, S_2, ..., S_r$) = $\sum$
- The information gain due to particular split of S into S_i, i = 1, 2, …., r
  - Information-gain (S, {$S_1, S_2, …., S_r$) = purity(S) – purity ($S_1, S_2, … S_r$)
- Measure of "cost" of a split:
  Information-content (S, {$S_1, S_2, ….., S_r$})) = – $\sum$
- Information-gain ratio = Information-gain (S, {$S_1, S_2, ……, S_r$})
  - Information-content (S, {$S_1, S_2, ….., S_r$})
- The best split is the one that gives the maximum information gain ratio

Finding Best Splits

- Categorical attributes (with no meaningful order):
  - Multi-way split, one child for each value
  - Binary split: try all possible breakup of values into two sets, and pick the best
- Continuous-valued attributes (can be sorted in a meaningful order)
  - Binary split:
    - Sort values, try each as a split point
      - E.g. if values are 1, 10, 15, 25, split at $\leq 1, \leq 10, \leq 15$
    - Pick the value that gives best split
  - Multi-way split:
    - A series of binary splits on the same attribute has roughly equivalent effect

Decision-Tree Construction Algorithm

Procedure *GrowTree* (*S*)
Partition (*S*);

Procedure Partition (*S*)
if (*purity* (*S*) > δ_p or |*S*| < δ_s) **then**
return;
for each attribute *A*
evaluate splits on attribute *A*;
Use best split found (across all attributes) to partition
S into $S_1, S_2, ..., S_r$,
for *i* = 1, 2, ….., *r*
Partition (S_i);

Other Types of Classifiers

- Neural net classifiers are studied in artificial intelligence and are not covered here
- Bayesian classifiers use **Bayes theorem**, which says
  - $p(c_j \mid d) = p(d \mid c_j)\, p(c_j)$
    - $p(d)$

    where

    $p(c_j \mid d)$ = probability of instance d being in class c_j,
  - $p(d \mid c_j)$ = probability of generating instance d given class c_j,
  - $p(c_j)$ = probability of occurrence of class c_j, and

  $p(d)$ = probability of instance d occuring

Naïve Bayesian Classifiers

- Bayesian classifiers require
  - computation of $p(d \mid c_j)$
  - precomputation of $p(c_j)$
  - $p(d)$ can be ignored since it is the same for all classes
- To simplify the task, **naïve Bayesian classifiers** assume attributes have independent distributions, and thereby estimate
  - $p(d \mid c_j) = p(d_1 \mid c_j) * p(d_2 \mid c_j) * \ldots * (p(d_n \mid c_j)$
  - Each of the $p(d_i \mid c_j)$ can be estimated from a histogram on d_i values for each class c_j
    - the histogram is computed from the training instances
  - Histograms on multiple attributes are more expensive to compute and store

Regression

- Regression deals with the prediction of a value, rather than a class.
  - Given values for a set of variables, $X_1, X_2, \ldots, X_n$, we wish to predict the value of a variable Y.
- One way is to infer coefficients $a_0, a_1, a_1, \ldots, a_n$ such that

  $Y = a_0 + a_1 * X_1 + a_2 * X_2 + \ldots + a_n * X_n$
- Finding such a linear polynomial is called **linear regression**.
  - In general, the process of finding a curve that fits the data is also called **curve fitting**.
- The fit may only be approximate
  - because of noise in the data, or
  - because the relationship is not exactly a polynomial
- Regression aims to find coefficients that give the best possible fit.

Association Rules

- Retail shops are often interested in associations between different items that people buy.

- Someone who buys bread is quite likely also to buy milk
- A person who bought the book *Database System Concepts* is quite likely also to buy the book *Operating System Concepts*.

- Associations information can be used in several ways.
  - E.g. when a customer buys a particular book, an online shop may suggest associated books.
- **Association rules:**
  - *bread* $\Rightarrow$ *milk* *DB-Concepts, OS-Concepts* $\Rightarrow$ Networks
  - Left hand side: antecedent, right hand side: consequent
  - An association rule must have an associated population; the population consists of a set of instances
    - E.g. each transaction (sale) at a shop is an instance, and the set of all transactions is the population
- Rules have an associated support, as well as an associated confidence.
- **Support** is a measure of what fraction of the population satisfies both the antecedent and the consequent of the rule.
  - E.g. suppose only 0.001 percent of all purchases include milk and screwdrivers. The support for the rule is *milk* $\Rightarrow$ *screwdrivers* is low.
- **Confidence** is a measure of how often the consequent is true when the antecedent is true.
  - E.g. the rule *bread* $\Rightarrow$ *milk* has a confidence of 80 percent if 80 percent of the purchases that include bread also include milk.

Finding Association Rules

- We are generally only interested in association rules with reasonably high support (e.g. support of 2% or greater)
- Naïve algorithm
  - Consider all possible sets of relevant items.
  - For each set find its support (i.e. count how many transactions purchase all items in the set).
    - **Large itemsets**: sets with sufficiently high support
  - Use large itemsets to generate association rules.
    - From itemset A generate the rule $A - \{b\} \Rightarrow b$ for each $b \in A$.
      - Support of rule = support (A).
      - Confidence of rule = support (A) / support ($A - \{b\}$)

Finding Support

- Determine support of itemsets via a single pass on set of transactions
  - Large itemsets: sets with a high count at the end of the pass
- If memory not enough to hold all counts for all itemsets use multiple passes, considering only some itemsets in each pass.
- Optimization: Once an itemset is eliminated because its count (support) is too small none of its supersets needs to be considered.

- The **a priori** technique to find large itemsets:
  - Pass 1: count support of all sets with just 1 item. Eliminate those items with low support
  - Pass *i*: candidates: every set of *i* items such that all its *i-1* item subsets are large
    - Count support of all candidates
    - Stop if there are no candidates

Other Types of Associations

- Basic association rules have several limitations
- Deviations from the expected probability are more interesting
  - E.g. if many people purchase bread, and many people purchase cereal, quite a few would be expected to purchase both
  - We are interested in positive as well as negative correlations between sets of items
    - Positive correlation: co-occurrence is higher than predicted
    - Negative correlation: co-occurrence is lower than predicted
- Sequence associations / correlations
  - E.g. whenever bonds go up, stock prices go down in 2 days
- Deviations from temporal patterns
  - E.g. deviation from a steady growth
  - E.g. sales of winter wear go down in summer
    - Not surprising, part of a known pattern.
    - Look for deviation from value predicted using past patterns

Clustering

- Clustering: Intuitively, finding clusters of points in the given data such that similar points lie in the same cluster
- Can be formalized using distance metrics in several ways
  - Group points into *k* sets (for a given *k*) such that the average distance of points from the centroid of their assigned group is minimized
    - Centroid: point defined by taking average of coordinates in each dimension.
  - Another metric: minimize average distance between every pair of points in a cluster
- Has been studied extensively in statistics, but on small data sets
  - Data mining systems aim at clustering techniques that can handle very large data sets
  - E.g. the Birch clustering algorithm (more shortly)

Hierarchical Clustering

- Example from biological classification

- (the word classification here does not mean a prediction mechanism)
  - chordata

    mammalia reptilia

    leopards humans snakes crocodiles

- Other examples: Internet directory systems (e.g. Yahoo, more on this later)
- Agglomerative clustering algorithms
  - Build small clusters, then cluster small clusters into bigger clusters, and so on
- Divisive clustering algorithms
  - Start with all items in a single cluster, repeatedly refine (break) clusters into smaller ones

Clustering Algorithms

- Clustering algorithms have been designed to handle very large datasets
- E.g. the Birch algorithm
  - Main idea: use an in-memory R-tree to store points that are being clustered
  - Insert points one at a time into the R-tree, merging a new point with an existing cluster if is less than some δ distance away
  - If there are more leaf nodes than fit in memory, merge existing clusters that are close to each other
  - At the end of first pass we get a large number of clusters at the leaves of the R-tree
    - Merge clusters to reduce the number of clusters

Other Types of Mining

- Text mining: application of data mining to textual documents
  - cluster Web pages to find related pages
  - cluster pages a user has visited to organize their visit history
  - classify Web pages automatically into a Web directory
- Data visualization systems help users examine large volumes of data and detect patterns visually
  - Can visually encode large amounts of information on a single screen
  - Humans are very good a detecting visual patterns

Applications:

• Information Processing
• Analytical Processing
• Data Mining

Topic – 4: Web Databases

Introduction to WDB:

Databases on the World Wide Web(WWW)

Popularly known as "the web"- originally developed in Switzerland in early 1990 for biological scientist to share information.
Based on client-server architecture
•Web servers
•Files encoded using HTML
•Hyperlinks
•URL
•Web browsers (Internet Explorer & Netscape Navigator) use http
•Website – collection of HTML documents

Accessing Databases on the World Wide Web:

CGI (Common Gateway Interface) – middleware
User access approaches:
• Access using CGI scripts
☐ CGI - PERL or C
Drawback:
less efficiency because of grouping user's requests not possible
• Access using JDBC
☐ JDBC- a name trademarked by Sun;
☐ Java classes - Java code capable browser - JDBC drivers
ORACLE WebServer:
pictorial representation

What do WDB do?

- What are the purposes for which WBDBs are used?
- Feiler (1999) distinguishes four main purposes:
  - "***Publishing data on the Web***."
    - "Here, you use the Web as a publication tool; browsers interact with dynamic hypertext markup language [DHMTL], application servers, and database queries to present the information as requested. The data flow is one way: from the database to the user."
  - ***"Sharing data on the Web***."

- "In this scenario you use databases and the Web to share data among people; the data flow is bidirectional—some people enter data, other people look it up."

– "***E-commerce."***
- "This area includes all online commercial transactions. Although the data flow is bidirectional, it typically consists of a relatively large amount of data that flows from the database to the customer (during the shopping and evaluation steps); that is followed by a relatively small amount of data that flows from the customer to the database as the sale is consummated."

– "***Totally database-driven Web sites***."
- "You can use databases to generate Web pages and keep them up to date. In this case, the database is usually invisible to the user; it is a behind-the-scenes assistant to a Web site."

Challenges of WDB:

1. Object technology -> DOM
2. HTML functionality is too simple to support complex application requests -> XML (subset of SGML)
3. Web page content can be made more dynamic
4. Support for a large number of clients coupled with reasonable response times for queries against very large databases
5. Security

Techniques for Developing and Maintaining WBDBs

– Underlying all WBDBs is a ***relational database-management system*** (RDBMS), together with one or more ***relational databases*** (RDBs) that actually contain the data or information of interest

– A ***Webpage*** defined in HTML or Dynamic HTML (DHTML) controls the visual display that the user of the WBDB sees.

– An ***interface*** (1) receives information from the user and passes it to the RDBMS, (2) extracts information from the RDB (with the assistance of the RDBMS), and (3) provides the information to the Webpage, whose HTML or DHTML structure makes the information visible.

– ***RDBMSs used for WBDBs***

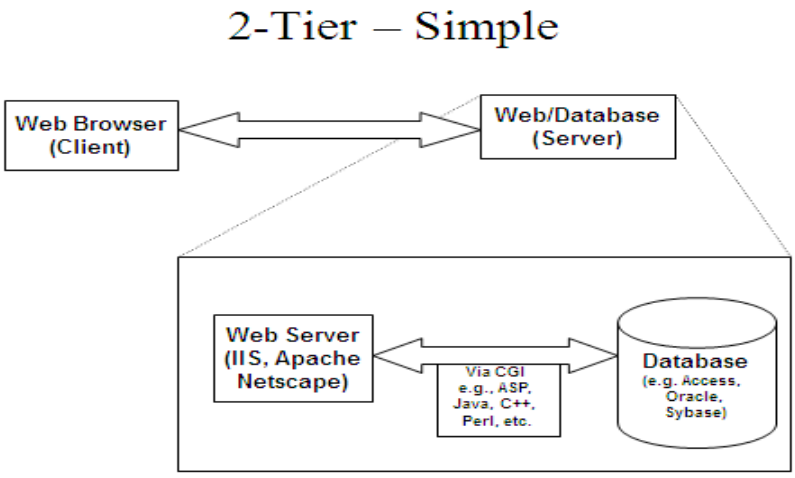
– small levels of use - Microsoft Access 97 (and later versions - for no more than a few simultaneous users).

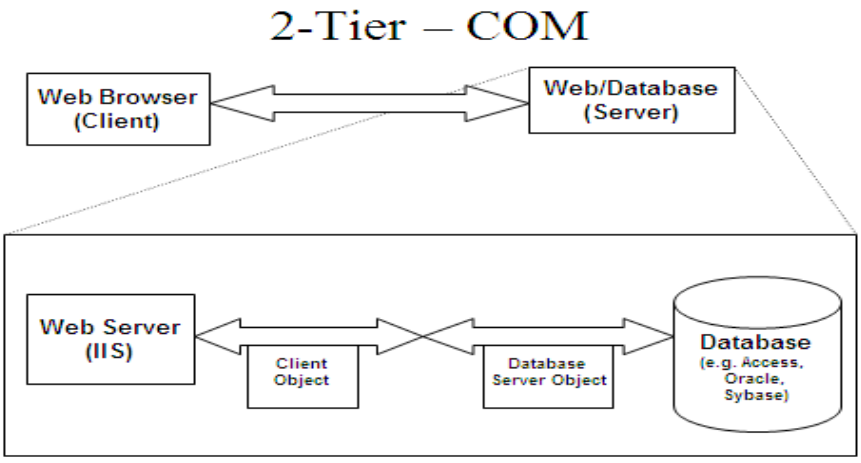
  - Large and heavily used WBDBs typically use high-level RDBMSs such as IBM DB2, Informix, Microsoft SQL Server, Oracle, and Sybase. A substantial majority of such sites use Oracle.
- The interfaces used for WBDBs fall into two broad classes
  - Interfaces intended for a specific application and written in a scripting language that conforms to Common Gateway Interface (GCI) standards

Web Architecture and Web Applications Issues

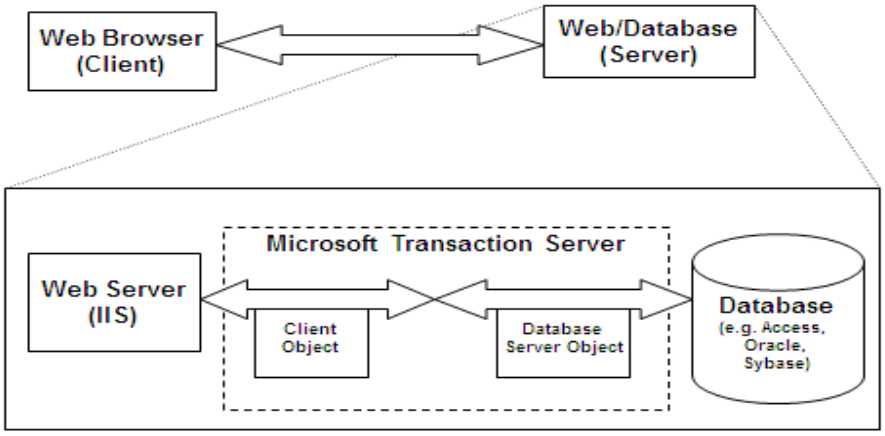
Semantic web architecture and applications are a dramatic departure from earlier database and applications generations. Semantic processing includes these earlier statistical and natural langue techniques, and enhances these with semantic processing tools.

- First, Semantic Web architecture is the automated conversion and storage of unstructured text sources in a semantic web database.
- Second, Semantic Web applications automatically extract and process the concepts and context in the database in a range of highly flexible tools.

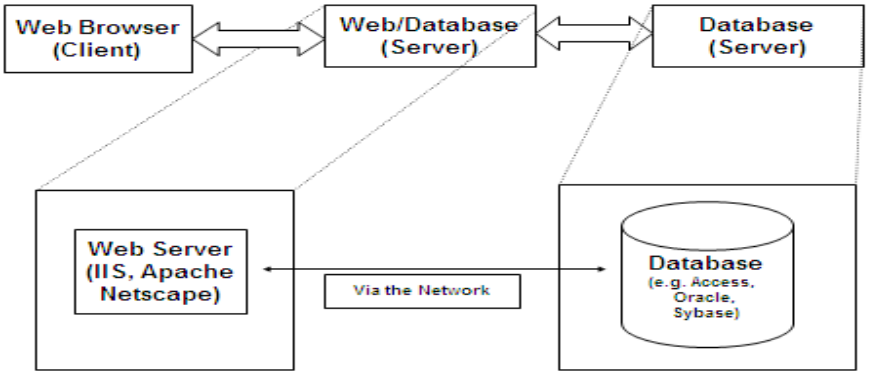



2-Tier – COM
Web Browser
(Client)
Web/Database
(Server)
Web Server
(IIS)
Client
Object
Database
Server Object
Database
(e.g. Access,
Oracle,
Sybase)

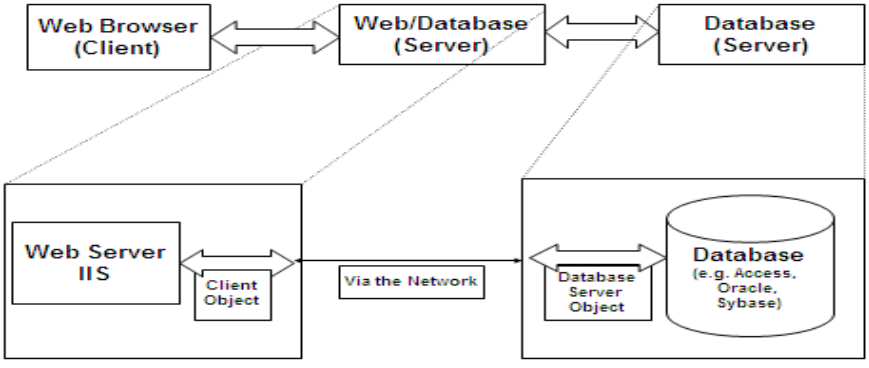
2-Tier -Transactional COM

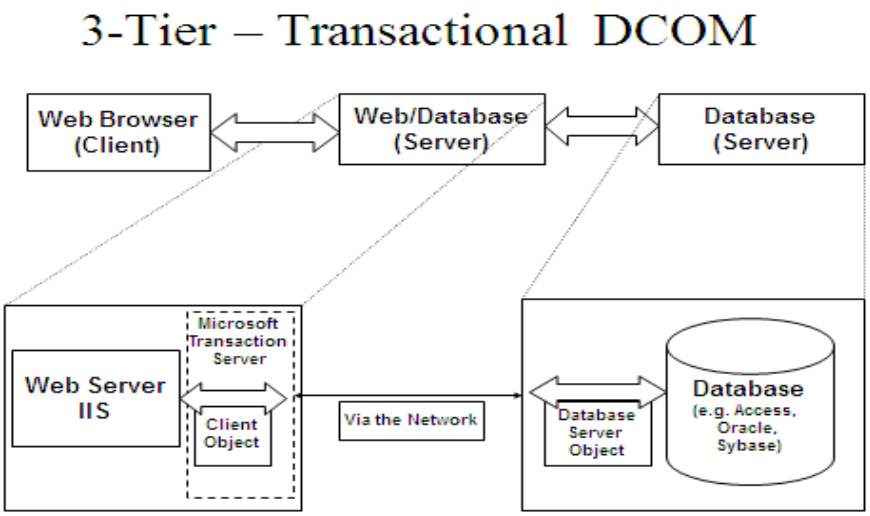


3-Tier – Simple



3-Tier – Distributed COM (DCOM)





a. Architecture; not only Application

First, the Semantic web is a complete database architecture, not only an application program.

- Semantic web architecture combines a two-step process. First, a Semantic Web database is created from unstructured text documents. And, then Semantic Web applications run on the Semantic Web database; not the original source documents.

- The Semantic Web architecture is created by first converting text files to XML and then analyzing these with a semantic processor.
- This process understands the meaning of the words and grammar of the sentence, and also the semantic relationships of the context. These meanings and relationships are then stored in a Semantic web database.

- Semantic Web applications directly access the logical relationships in the Semantic Web database. Semantic web applications can efficiently and accurately search, retrieve, summarize, analyze and report discrete concepts or entire documents from huge databases.

b. Structured and Unstructured Data

Second, Semantic Web architecture and applications handle both structured and unstructured data. Structured data is stored in relational databases with static classification systems, and also in discrete documents.

- These databases and documents can be processed and converted to Semantic Web databases, and then processed with unstrctured data.

- Much of the data we read, produce and share is now unstructured; emails, reports, presentations, media content, web pages. And, these documents are stored in many different formats; text, email files, Microsoft word processor, spreadsheet, presentation files, Lotus Notes, Adobe.pdf, and HTML.
- It is difficult, expensive, slow and inaccurate to attempt to classify and store these in a structured database. All of these sources can be automatically converted to a common Semantic Web database, and integrated into one common information source.

c. Dynamic and Automatic; not Static and Manual

Third, Semantic Web database architecture is dynamic and automated.

- Each new document which is analyzed, extracted and stored in the Semantic Web expands the logical relationships in all earlier documents. These expanding logical relationships increase the understanding of content and context in each document, and the entire database.
- The Semantic Web conversion process is automated. No human action is required for maintaining a taxonomy, meta data tagging or classification. The semantic database is constantly updated and more accurate.

- Semantic Web architecture is different from relational database systems.
- Relational databases are manual and static because these are based on a manual process for maintaining a taxonomy, meta data tagging and document classification in static file structures.
- Documents are manually captured, read, tagged, classified and stored in a relational database only once, and not updated.

More important, the increase in new documents and information in a relational database does not make the database more "intelligent" about the concepts, relationships or documents.

d. From Machine Readable to Machine Understandable

Fourth, Semantic Web architecture and applications support both human and machine intelligence systems.

- Humans can use Semantic Web applications on a manual basis, and improve the efficiency of search, summary, analysis and reporting tasks.
- Machines can also use Semantic Web applications to perform tasks that humans cannot do; because of the cost, speed, accuracy, complexity and scale of the tasks.

e. Synthetic vs Artificial Intelligence:

- Semantic Web technology is NOT "Artificial Intelligence".
- AI was a mythical marketing goal to create "thinking" machines.
- The Semantic Web supports a much more limited and realistic goal. This is "Synthetic Intelligence". The concepts and relationships stored in the Semantic Web database are "synthesized", or brought together and integrated, to automatically create a new summary, analysis, report, email, alert; or launch another machine application.
- The goal of Synthetic Intelligence information systems is bringing together all information sources and user knowledge, and synthesizing these in global networks.

Topic – 5: Mobile Databases

Mobile computing → Data communication & processing

☐ Wireless technology – establishes communication with other uses & manages their work while they are mobile.
(eg) traffic police, weather reporting services, financial market reporting,

information brokering applications**.**
Problems:
Data management, transaction management, database recovery

- The main advantage of using a mobile database in your application is offline access to data—in other words, the ability to read and update data without a network connection. This helps avoid problems such as dropped connections, low bandwidth, and high latency that are typical on wireless networks today.

Types of data in Mobile Applications:

☐ **Mobile applications**

Vertical applications Users access data within a specific cell
Horizontal applications Users access data distributed throughout the system

☐ **Data (eg) e-mail**

Private data Single user owns & manages the data
Shared data Accessed both in read & write mode by a group of users. (eg) inventory
Public data Anyone can read data; only one source updates it.
(eg) stock prices, weather bulletins

What is a Mobile Database System (MDS)?

A system with the following structural and functional properties

- Distributed system with *mobile connectivity*
- Full database system capability
- Complete spatial mobility
- Built on PCS/GSM platform
- Wireless and wired communication capability

What is a mobile connectivity?
A mode in which a client or a server can establish communication with each other whenever needed. *Intermittent connectivity* is a special case of mobile connectivity.

What is intermittent connectivity?
A node in which only the client can establish communication whenever needed with the server but the server cannot do so.

Mobile Database Systems (MDS)

- **Architecture**
- **Data categorization**
- **Data management**
- **Transaction management**
- **Recovery**

MDS Applications

- Insurance companies
- Emergencies services (Police, medical, etc.)
- Traffic control
- Taxi dispatch
- E-commerce

MDS Limitations

- ❖ Limited wireless bandwidth
- ❖ Wireless communication speed
- ❖ Limited energy source (battery power)
- ❖ Less secured
- ❖ Vulnerable to physical activities
- ❖ Hard to make theft proof

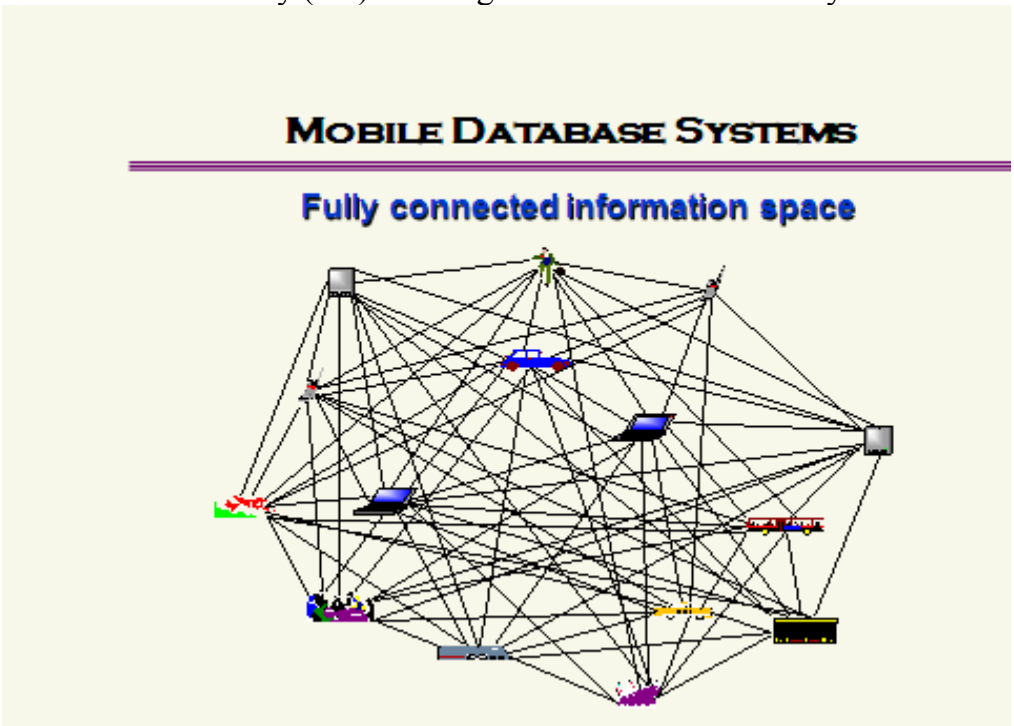
MDS capabilities

- ❖ Can physically move around without affecting data availability
- ❖ Can reach to the place data is stored
- ❖ Can process special types of data efficiently
- ❖ Not subjected to connection restrictions
- ❖ Very high reachability
- ❖ Highly portable

Mobile Computing Architecture:

☐ Fixed Hosts (FS) → Interconnected through a highspeed wired network
Base Stations (BS) → Interconnected through a highspeed wired network

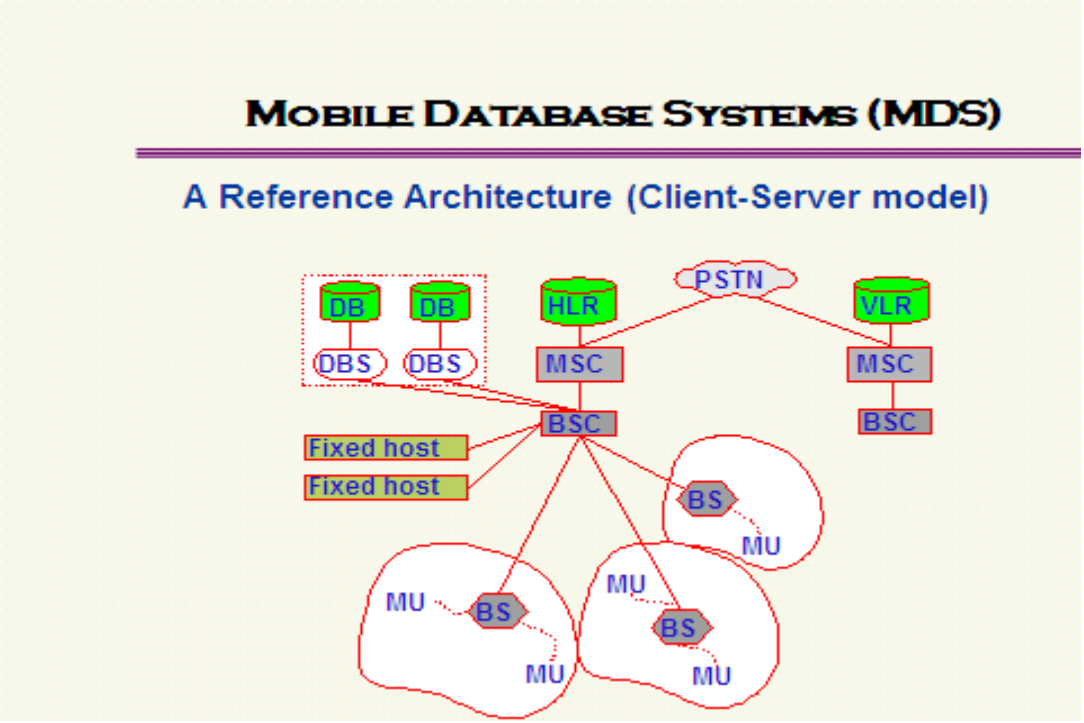
☐ Equipped with wireless interfaces
☐ Clinet-server paradigm
☐ Mobile Units (MU), Base stations communicate through wireless channels
☐ Uplink channel, downlink channel
☐ Geographic mobility domain
☐ Residence latency (RL) – average duration of a user's stay in the cell



Fully connected information space

- Each node of the information space has some communication capability.
- Some node can process information.
- Some node can communicate through voice channel.
- Some node can do both

Can be created and maintained by integrating legacy database systems, and wired and wireless systems (PCS, Cellular system, and GSM)



MDS Design

Objective
To build a truly ubiquitous information processing system by overcoming the inherent limitations of wireless architecture.

MDS Issues:

Data Management
Data Caching
Data Broadcast(Broadcast disk)
Data Classification

Transaction Management

Query Processing
Concurrency control
Database recovery

MDS Data Management Issues
Data Management Issues:
Distributed data management issues can be applied to mobile databases with the additional considerations.
- ☐ Data distribution and replication
- ☐ Transaction modes
- ☐ Query Processing
- ☐ Recovery & Fault tolerance
- ☐ Mobile database design

Intermittently Synchronized DataBase Environment (ISDBE)
Intermittently Synchronized DataBases (ISDBs)

How to improve data availability to user queries using limited bandwidth?
Possible schemes
- ❖ Semantic data caching: The cache contents is decided by the results of earlier transactions or by *semantic data set.*
- ❖ Data Broadcast on wireless channels

How to improve data availability to user queries using limited bandwidth?
Semantic caching
- ❖ Client maintains a semantic description of the data in its cache instead of maintaining a list of pages or tuples.
- ❖ The server processes simple predicates on the database and the results are cached at the client.

Data Broadcast (Broadcast disk)
A set of most frequently accessed data is made available by continuously broadcasting it on some fixed radio frequency. Mobile Units can tune to this frequency and download the desired data from the broadcast to their local cache.
A broadcast (file on the air) is similar to a disk file but located on the air.

Data Broadcast (Broadcast disk)
The contents of the broadcast reflects the data demands of mobile units.
This can be achieved through data access history, which can be fed to the data broadcasting system.
For efficient access the broadcast file use index or some other method.

How MDS looks at the database data?

Data classification

- Location Dependent Data (LDD)
- Location Independent Data (LID)

Location Dependent Data (LDD)

- The class of data whose value is functionally dependent on location. Thus, the value of the location determines the correct value of the data.
- Location Data value
- Examples: City tax, City area, etc.

Location Independent Data (LID)

The class of data whose value is functionally independent of location. Thus, the value of the location does not determine the value of the data.

Example: Person name, account number, etc.

The person name remains the same irrespective of place the person is residing at the time of enquiry.

Location Dependent Data (LDD)

Example: Hotel Taj has many branches in India. However, the room rent of this hotel will depend upon the place it is located. Any change in the room rate of one branch would not affect any other branch.

Schema: It remains the same only multiple correct values exists in the database.

Location Dependent Data (LDD)

LDD must be processed under the location constraints. Thus, the tax data of Pune can be processed correctly only under Pune's finance rule.

Needs *location binding* or *location mapping* function.

Location Dependent Data (LDD)

Location binding or *location mapping* can be achieved through database schema or through a location mapping table.

MDS Data Management Issues

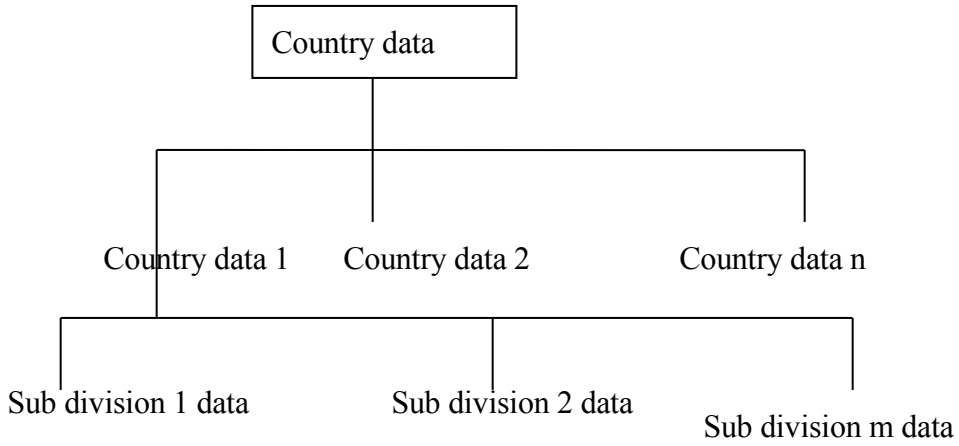
Location Dependent Data (LDD) Distribution

MDS could be a federated or a multidatabase system. The database distribution (replication, partition, etc.) must take into consideration LDD.

One approach is to represent a city in terms of a number of mobile cells, which is referred to as "Data region". Thus, Pune can be represented in terms of *N* cells and the LDD of Pune can be replicated at these individual cells.

Concept of Hierarchy in LDD

In a data region the entire LDD of the location can be in a hierarchical fashion.



MDS Query processing

Query types

- Location dependent query
- Location aware query
- Location independent query

Location dependent query

A query whose result depends on the geographical location of the origin of the query.

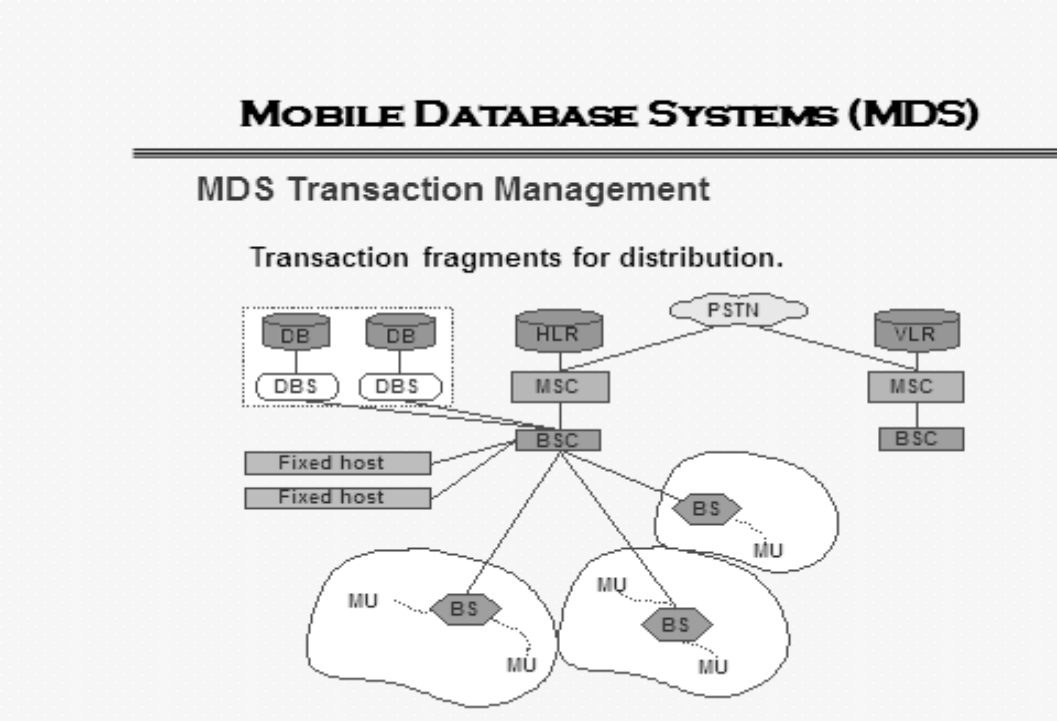
Example

What is the distance of Pune railway station from here?

The result of this query is correct only for "here".

Location dependent query
Situation: Person traveling in the car desires to know his progress and continuously asks the same question. However, every time the answer is different but correct.
Requirements: Continuous monitoring of the longitude and latitude of the origin of the query. GPS can do this.

MDS Transaction Management
Transaction properties: ACID (Atomicity, Consistency, Isolation, and Durability). Too rigid for MDS. Flexibility can be introduced using workflow concept. Thus, a part of the transaction can be executed and committed independent to its other parts.



Transaction fragments for distributed execution
Execution scenario: User issues transactions from his/her MU and the final results comes back to the same MU. The user transaction may not be completely executed at the MU so it is fragmented and distributed among database servers for execution. This creates a Distributed mobile execution.

MOBILE DATABASE SYSTEMS (MDS)

MDS Transaction Management

A mobile transaction (MT) can be defined as

T_i is a triple $<F, L, FLM>$; where

$F = \{e_1, e_2, \ldots, e_n\}$ is a set of execution fragments,

$L = \{l_1, l_2, \ldots, l_n\}$ is a set of locations, and

$FLM = \{flm_1, flm_2, \ldots, flm_n\}$ is a set of fragment location mapping where $\forall j, flm_i(e_j) = l_i$

MOBILE DATABASE SYSTEMS (MDS)

MDS Transaction Management

An execution fragment e_{ij} is a partial order $e_{ij} = \{\sigma_i, \leq_i\}$ where

- $\sigma_i = OS_j \cup \{N_j\}$ where $OS_j = \cup_k O_{jk}$, $O_{jk} \in \{read, write\}$, and $N_j \{Abort_L, Commit_L\}$.
- For any O_{jk} and O_{jl} where $O_{jk} = R(x)$ and $O_{jl} = W(x)$ for data object x, then either $O_{jk} \leq_j O_{jl}$ or $O_{jl} \leq_j O_{jk}$.

Mobile Transaction Models

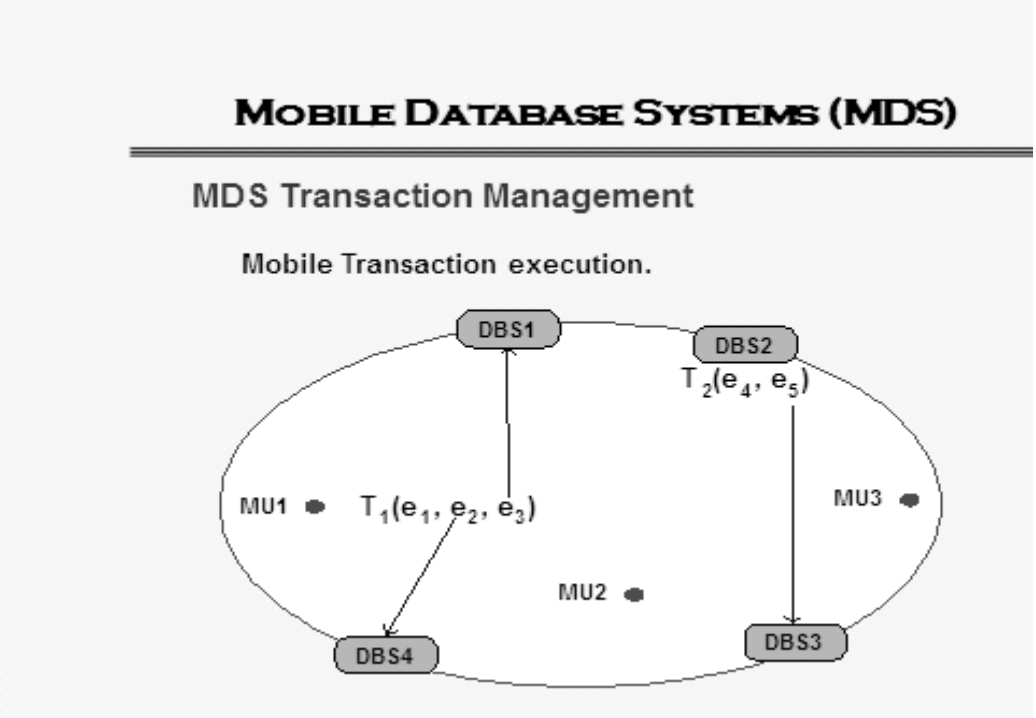
Kangaroo Transaction: It is requested at a MU but processed at DBMS on the fixed network. The management of the transaction moves with MU. Each transaction is divided into subtransactions. Two types of processing modes are allowed, one ensuring overall atomicity by requiring compensating transactions at the subtransaction level.

Reporting and Co-Transactions: The parent transaction (workflow) is represented in terms of reporting and co-transactions which can execute anywhere. A reporting transaction can share its partial results with the parent transaction anytime and can

commit independently. A co-transaction is a special class of reporting transaction, which can be forced to wait by other transaction.

Clustering: A mobile transaction is decomposed into a set of weak and strict transactions. The decomposition is done based on the consistency requirement. The read and write operations are also classified as weak and strict

Semantics Based: The model assumes a mobile transaction to be a long lived task and splits large and complex objects into smaller manageable fragments. These fragments are put together again by the merge operation at the server. If the fragments can be recombined in any order then the objects are termed *reorderable* objects.



Serialization of concurrent execution

- Two-phase locking based (commonly used)
- Timestamping
- Optimistic

Reasons these methods may not work satisfactorily

- Wired and wireless message overhead.
- Hard to efficiently support disconnected operations.
- Hard to manage locking and unlocking operations.

Serialization of concurrent execution.

New schemes based on timeout, multiversion, etc., may work. A scheme, which uses minimum number of messages, especially wireless messages is required

Database update to maintain global consistency

Database update problem arises when mobile units are also allowed to modify the database. To maintain global consistency an efficient database update scheme is necessary.

Transaction commit.

In MDS a transaction may be fragmented and may run at more than one nodes (MU and DBSs). An efficient commit protocol is necessary. 2-phase commit (2PC) or 3-phase commit (3PC) is no good because of their generous messaging requirement. A scheme which uses very few messages, especially wireless, is desirable

Transaction commit.

One possible scheme is "timeout" based protocol.

Concept: MU and DBSs guarantee to complete the execution of their fragments of a mobile transaction within their predefined timeouts. Thus, during processing no communication is required. At the end of timeout, each node commit their fragment independently.

Protocol: TCOT-Transaction Commit On Timeout

Requirements
Coordinator: Coordinates transaction commit
Home MU: Mobile Transaction (MT) originates here
Commit set: Nodes that process MT (MU + DBSs)
Timeout: Time period for executing a fragment

Protocol: TCOT-Transaction Commit On Timeout

- MT arrives at Home MU.
- MU extract its fragment, estimates timeout, and send rest of MT to the coordinator.
- Coordinator further fragments the MT and distributes them to members of commit set.
- MU processes and commits its fragment and sends the updates to the coordinator for DBS.
- DBSs process their fragments and inform the coordinator.
- Coordinators commits or aborts MT.

Transaction and database recovery

Complex for the following reasons

- Some of the processing nodes are mobile
- Less resilient to physical use/abuse
- Limited wireless channels
- Limited power supply
- Disconnected processing capability

Desirable recovery features

- Independent recovery capability
- Efficient logging and checkpointing facility
- Log duplication facility

- Independent recovery capability reduces communication overhead. Thus, MUs can recover without any help from DBS
- Efficient logging and checkpointing facility conserve battery power
- Log duplication facility improves reliability of recovery scheme

Possible approaches

- Partial recovery capability
- Use of mobile agent technology

Possible MU logging approaches

- Logging at the processing node (e.g., MU)
- Logging at a centralized location (e.g., at a designated DBS)
- Logging at the place of registration (e.g., BS)
- Saving log on Zip drive or floppies.

Sample Questions

Topic – 1:

Topic – 2:

Topic – 3:

Topic – 4:

1. Explain databases on the World Wide Web. (8M)

Topic – 5:

1. Highlight the features of Mobile Databases. (8M)

University Questions

1. Explain the architecture of a data warehouse with a neat diagram. (8M)
2. What are the various issues to be considered while building a data warehouse? Explain. (8M)
3. Discuss about the following data mining techniques:
    a) Association rules
    b) Classification

********** End of Unit – III **********