Web Technologies and Applications

Winter 2001

CMPUT 499: DBMS and WWW

Dr. Osmar R. Zaïane

University of Alberta

Course Content

• Introduction	• Databases & WWW
• Internet and WWW	• SGML / XML
• Protocols	• Managing servers
• HTML and beyond	• Search Engines
• Animation & WWW	• Web Mining
• Java Script	• CORBA
• Dynamic Pages	• Security Issues
• Perl Intro.	• Selected Topics
• Java Applets	• Projects

Objectives of Lecture 10

DBMS & WWW

- Students will be able to understand the different current methods used to access databases on the Web.
- Introduce the basic database access techniques.
- Understand the benefits and trade-offs for each technique

Outline of Lecture 10

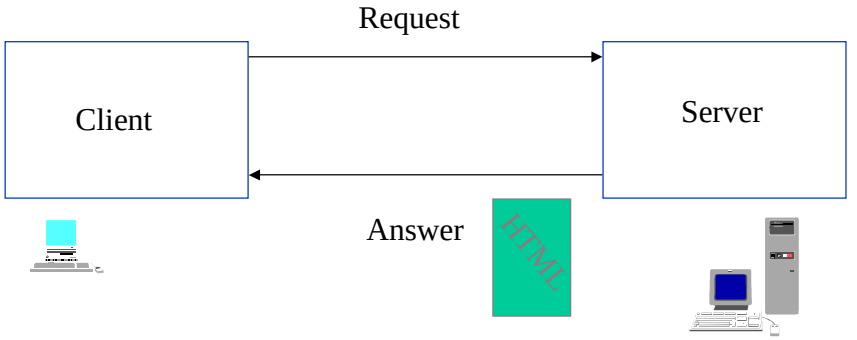
- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Introduction

Motivation

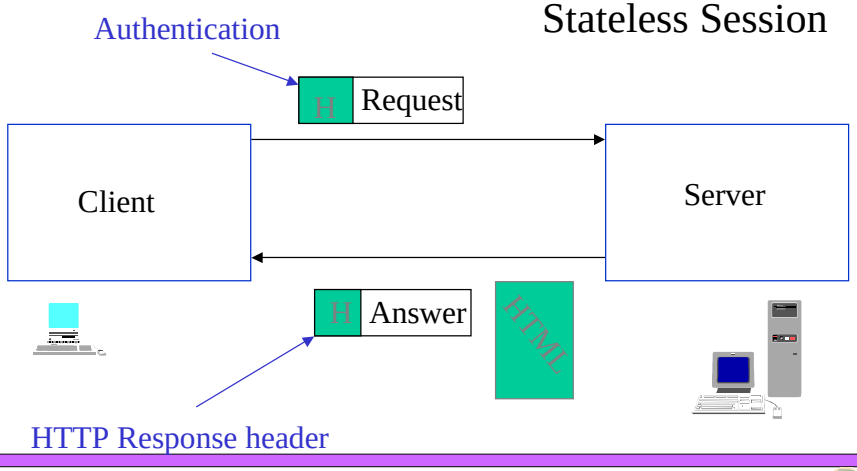
- WWW
  - user friendly
  - popular
  - accessible
  - cost effective
- Databases
  - structured and organized
  - secure/ reliable
  - most up-to-date information
  - scalable
  - high availability
  - automatic recovery
  - data integrity

HTTP Client-Server Architecture

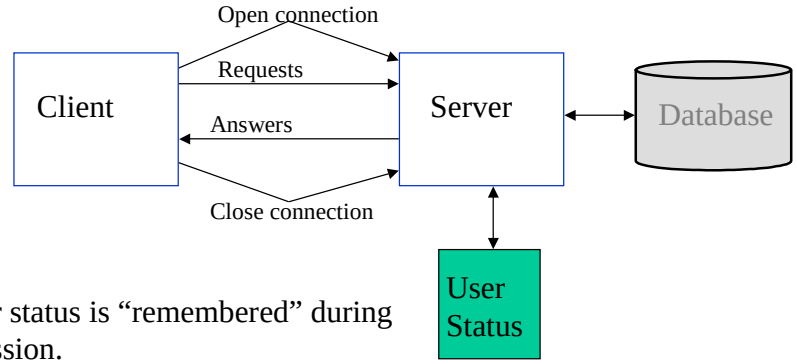


HTTP Client-Server Architecture

Stateless Session

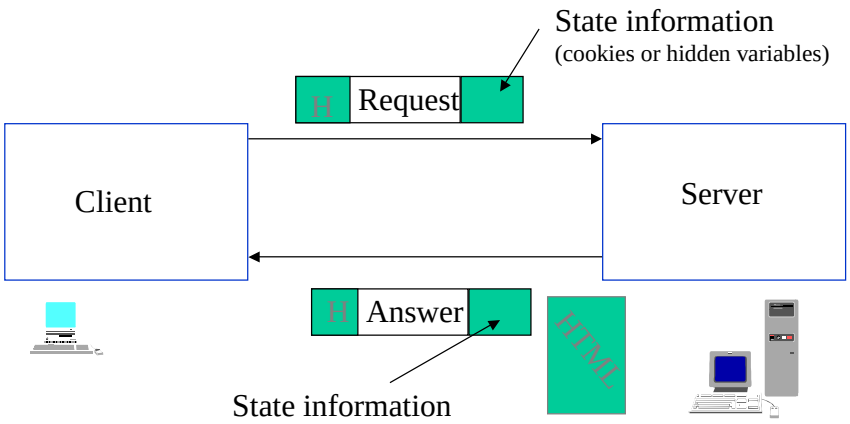


Database Client-Server Architecture



User status is “remembered” during a session.

Simulation of status in stateless session



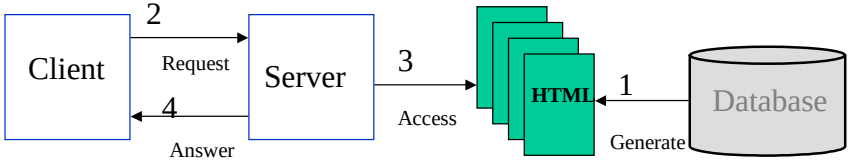
Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Off-line access to databases

- Periodically extract data from database and generate static pages based on common usage and requests
- Navigation between pages is done through static links generated in the HTML pages

Off-line access to databases



Off-line access to databases

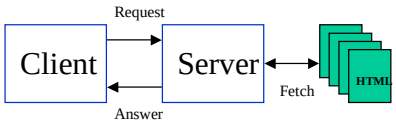
- Can be indexed by search engines
- Easy to implement
- Can be cached by client and accessed off-line

- Limited navigation
- Can not access data unless page has been generated
- Data not up-to-date

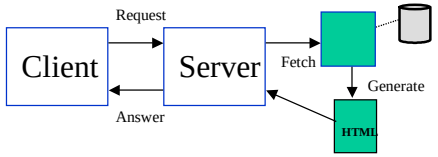
Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Static vs. Dynamic



An HTML document stored in a file is a static Web page. Unless the file is edited, its content does not change.



A dynamic Web page is generated or partially generated each time it is accessed.

Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Server Side Includes

A server side include is a simple HTML-like tag.
The Web server parses HTML files and replaces the included tags with their value or output in the HTML file.

```
<HTML> <HEAD>
          <TITLE> My Page</TITLE>
        </HEAD>
<BODY>
<H1>My Home Page</H1>
<P>
<!--# include file="top_menu"-->
<p>
</BODY></HTML>
```

<!--#INCLUDE FILE="file"-->

<!--#EXEC CMD="todo.exe"-->

<!--#ECHO VAR="DATE_LOCAL">

Server Side Includes

- Results generated on the fly
- Pages easy to maintain
- Personalized pages for each user

- All files need to be parsed
- Slow

SQL database connectivity using server side includes

- W3-msql (Hughes Technologies)
- <! msql connect www.cs.sfu.ca>
- <! msql database students>
- <! msql query "select studid, name, firstname from students" q1>
- <! msql print "Student:@q1.0 Name: @q1.2 @q1.1
">
- <! msql fetch q1>
- <! msql free q1>
- CompuServe Internet Office Webserver
- <!--#SQL SQL="select studid, firstname, name from students" format="Student: %s Name: %s %s"-->
- Connectivity to miniSQL, Sybase, Oracle, Informix and any ODBC compliant

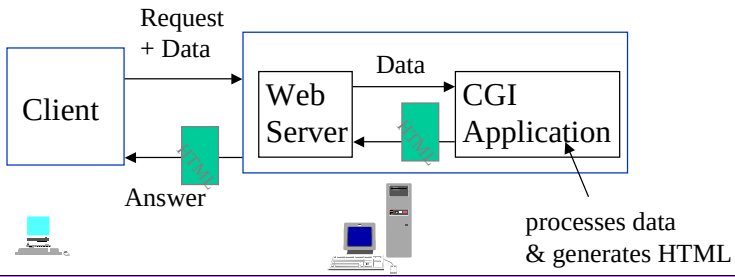
Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Common Gateway Interface

CGI is a set of specification for passing information between a client Web browser, a Web server and an application (CGI application).

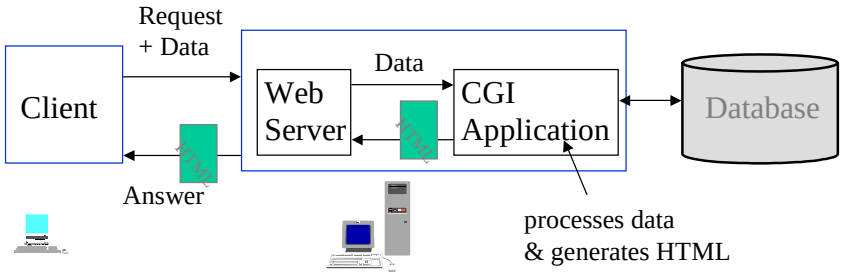
- Filling out an HTML form
- Clicking on a link in an HTML page



Common Gateway Interface

- Client sends request (GET or POST)
- Server receives request (name of CGI + data)
- Server launches CGI application and passes request to it by means of environment variables
- CGI application returns data to server (STDOUT). First line contains the MIME content-type
- Server adds standard HTTP header and returns data to client.

Common Gateway Interface

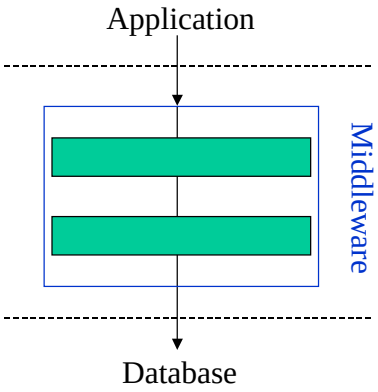


- Embed SQL in CGI application to access database
- Use hidden state information and user supplied data to build database queries
- Generate HTML based on query results

Common Gateway Interface

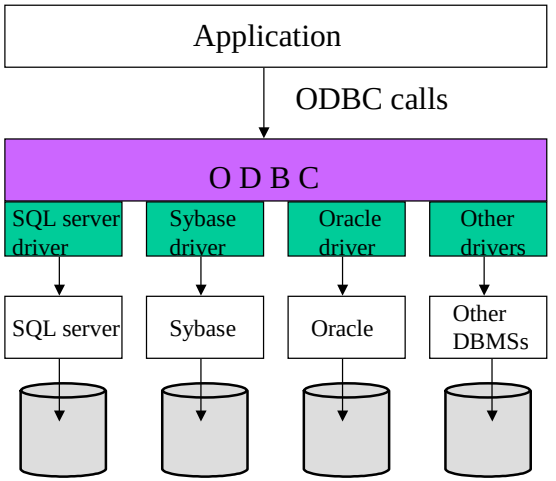
- Executed on the server
- Implemented in any programming or scripting language
- Started by server upon client request
- Generated HTML are not indexed by search engines
- New application process for each request
- Does not scale well because of the overhead of spawning new application process for each request

Multi-tier



- Modularity: specialized layers
- Scalability: replicated layers
- Flexibility: interchange layers
- Can be slow, excessive overhead
- Appropriate for standard interfaces
- Fault-tolerant

Open DataBase Connectivity



Outline of Lecture 10

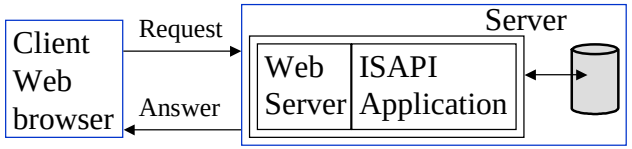
- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Microsoft Internet Information Server

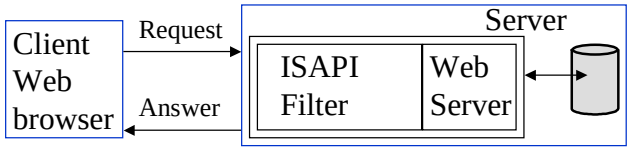
Internet Server API

- Used to create applications activated by Web users
- ISAPI used to create applications that run as DLLs on Web server
- Better performance than CGI because DLLs are loaded into memory at server run-time
- Less overhead because each request does not start a separate process
- Unstable: if ISAPI DLL application has bugs, it may crash the entire server
- Proprietary API

Internet Server API



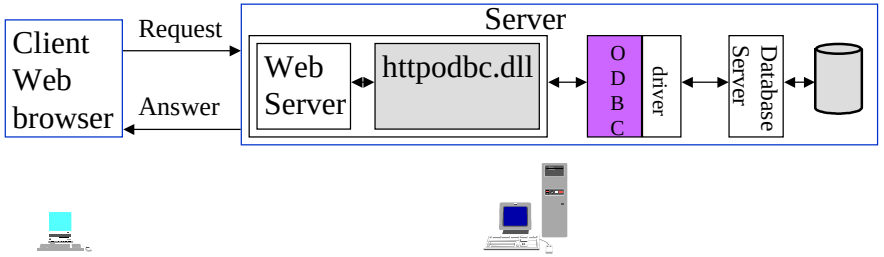
ISAPI DLL application becomes part of the Web server



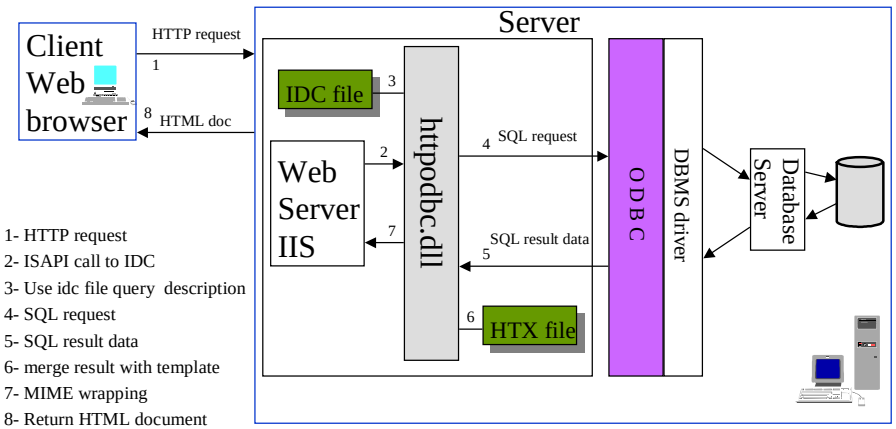
Allows pre-processing of requests and post-processing of responses

Internet Database Connector

Access to databases is accomplished through a component of the Internet information server called the Internet Database Connector (IDC).
The IDC is an ISAPI DLL (httpodbc.dll) that uses ODBC to gain access to databases.



Internet Database Connector



1- HTTP request
2- ISAPI call to IDC
3- Use idc file query description
4- SQL request
5- SQL result data
6- merge result with template
7- MIME wrapping
8- Return HTML document

Internet Database Connector

IDC merges the data being returned with the HTML extension template

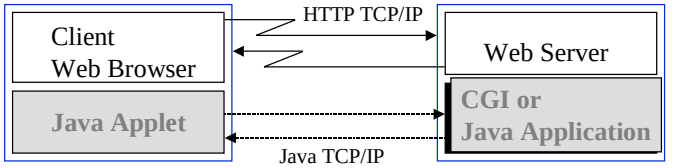
IDC file contains:
- data source name
- link to template htx file
- SQL statement

HTX file contains:
- HTML document
- additional tags to format data returned

Outline of Lecture 10

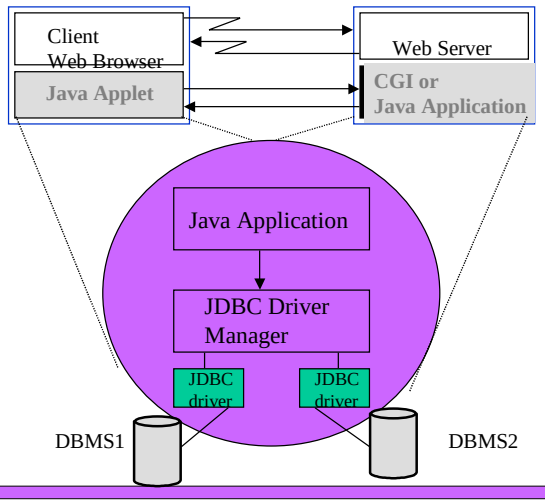
- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

Databases the Java Way

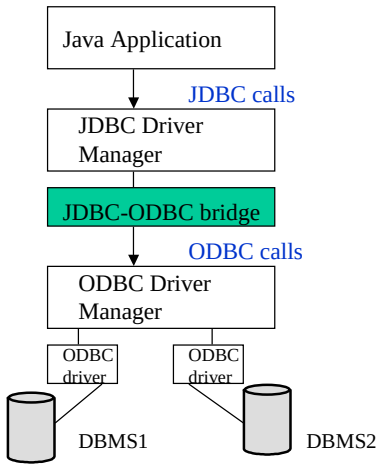


- HTTP TCP/IP connection is stateless
- Java connection can have an application session and store state information
- Java applets run on client side and can bypass Web browser-Web server connection
- Java is multi-threaded (multi-threaded socket server)

Java DataBase Connectivity



JDBC - ODBC Bridge



- A sophisticated JDBC driver
- Allows developers to use existing ODBC drivers when DBMS vendors only provide ODBC driver and no JDBC driver
- Once JDBC driver is provided, changes in Java code are minimal

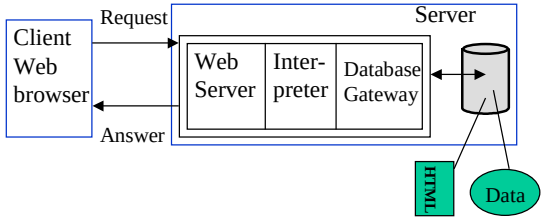
Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

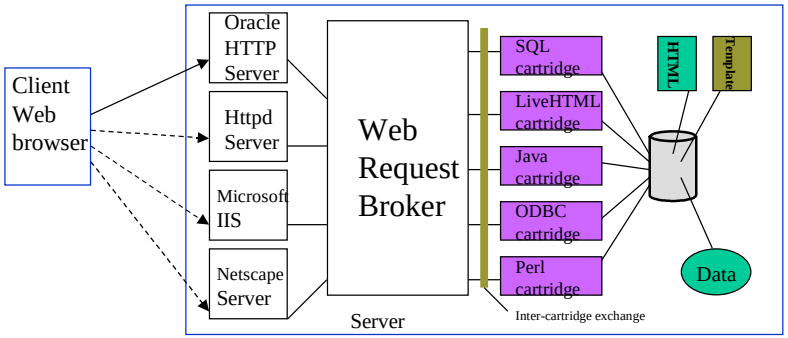
Sybase NetImpact Dynamo

- Proprietary server has built in interpreter for carrying out embedded instructions (SQL, javascript, Perl)
- in-line scripting
- web.sql
- SQL Remote replicates static and dynamic HTML documents as well as data for disconnected mobile users

```
<HTML> <HEAD>
          <TITLE> My Page</TITLE>
        </HEAD>
<BODY>
<H1>My Home Page</H1>
<P>
<SYB  type=SQL>
...
</SYB>
</BODY></HTML>
```

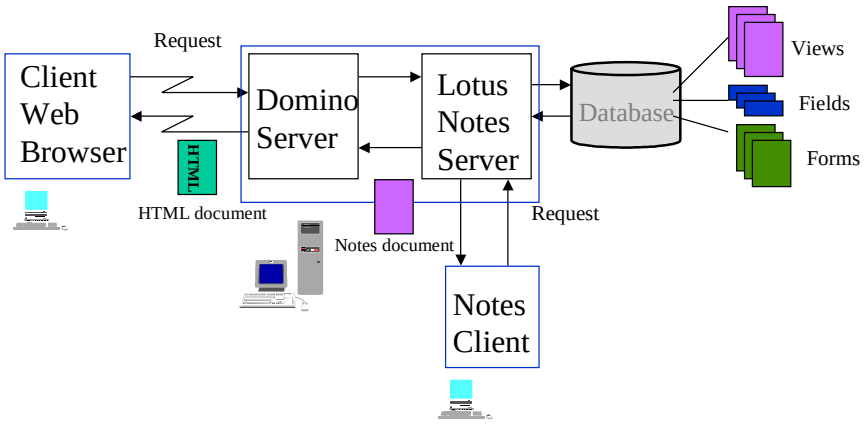


Oracle Web Application Server



- WBR dispatches and balances the load
- Open API for WRB
- Scalable
- Distributed

Lotus Notes Domino Server



Recapitulation

- Stateless HTTP client-server architecture
- Off-line access to databases becomes stale
- Dynamic Web pages can access up-to-date data
  - SQL embedded in HTML (server side includes)
  - CGI application (database gateways)
- Windows NT/IIS = idc file with SQL + htx template
- Java DBC client side connection to databases
- Sybase, Oracle and others (middleware + templates)

Outline of Lecture 10

- Introduction
- Off-line access to databases
- Static and dynamic Web pages
- SQL embedded in HTML (server side includes)
- CGI and servlet solution to database gateways
- Internet database connector: Microsoft solution
- JDBC: databases the Java way
- Solutions from database vendors
- Association Rule Mining

What Is Association Mining?

- **Association rule mining searches for relationships between items in a dataset**:
  - Finding association, correlation, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.
  - Rule form: "**Body → Head [support, confidence]**".

- **Examples:**
  - buys(x, "bread") → buys(x, "milk") [0.6%, 65%]
  - major(x, "CS") ^ takes(x, "DB") → grade(x, "A") [1%, 75%]

Basic Concepts

A transaction is a set of items: $T=\{i_a, i_b,\ldots i_t\}$

$T \subset I$, where I is the set of all possible items $\{i_1, i_2,\ldots i_n\}$

D, the task relevant data, is a set of transactions.

An association rule is of the form:
$P \rightarrow Q$, where $P \subset I$, $Q \subset I$, and $P \cap Q = \varnothing$

Basic Concepts (con't)

P➔Q holds in *D* with support s
and
P➔Q has a confidence c in the transaction set *D*.

Support(P➔Q) = Probability(P∪Q)
Confidence(P➔Q)=Probability(Q/P)

Itemsets

A set of items is referred to as itemset.

An itemset containing k items is called **k-itemset**.

An items set can also be seen as a conjunction of items (or a predicate)

Support and Confidence

- **Support** of $P = P_1 \wedge P_2 \wedge ... \wedge P_n$ in ***D***
  - σ(P/*D*) is the percentage of transactions T in *D* satisfying P. (number of T by cardinality of *D*).
- **Confidence** of a rule P $\rightarrow$ Q
  - φ(P $\rightarrow$ Q/ *D*) ratio σ((P $\wedge$ Q)/ *D*) by σ(P/ *D*)
- Thresholds:
  - *minimum support* σ'
  - *minimum confidence* φ'

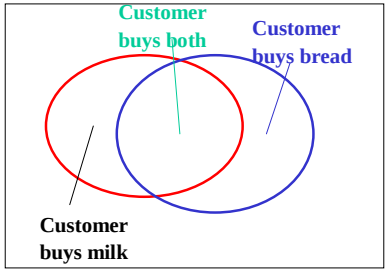
Strong Rules

- **Frequent (or large) predicate** P in set D
  - support of P larger than minimum support,
- Rule P $\rightarrow$ Q (c%) is **strong**
  - predicate (P $\wedge$ Q) is frequent (or large),
  - c is larger than minimum confidence.

How do we Mine Association Rules?

- **Input**
  - A database of transactions
  - Each transaction is a list of items (Ex. purchased by a customer in a visit)
- Find all rules that associate the presence of one set of items with that of another set of items.
  - Example: *98% of people who purchase tires and auto accessories also get automotive services done*
  - There are no restrictions on the number of items in the head or body of the rule.

Rule Measures: Support and Confidence



Find all the rules X & Y ➔ Z with minimum confidence and support
- support, ***s***, probability that a transaction contains {X, Y, Z}
- confidence, ***c***, conditional probability that a transaction having {X, Y} also contains Z.

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Let minimum support 50%, and minimum confidence 50%, we have
- A ➔ C (50%, 66.6%)
- C ➔ A (50%, 100%)

Mining Association Rules

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Min. support 50%
Min. confidence 50%

Frequent Itemset	Support
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

For rule A ➔ C:

support = support({A, C}) = 50%

confidence = support({A, C})/support({A}) = 66.6%

The Apriori principle:

Any subset of a frequent itemset must be frequent.

Mining Frequent Itemsets: the Key Step

① Find the *frequent itemsets*: the sets of items that have minimum support
- A subset of a frequent itemset must also be a frequent itemset, i.e., if {AB} is a frequent itemset, both {A} and {B} should be frequent itemsets
- Iteratively find frequent itemsets with cardinality from 1 to k (k-itemsets)

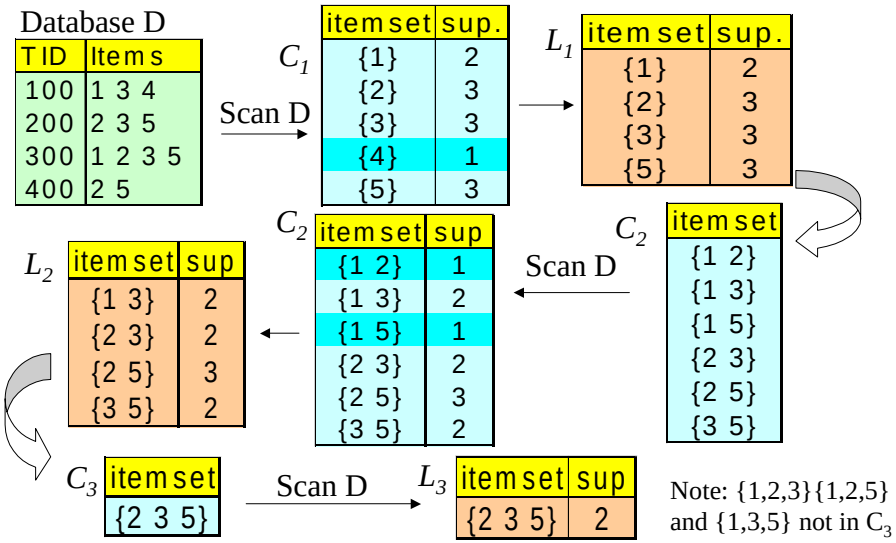
② Use the frequent itemsets to generate association rules.

The Apriori Algorithm

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) **do begin**
  C_{k+1} = candidates generated from L_k;
  for each transaction t in database **do**
    increment the count of all candidates in C_{k+1} that are contained in t
  L_{k+1} = candidates in C_{k+1} with min_support
  end
return $\cup_k L_k$;

The Apriori Algorithm -- Example



Generating Association Rules from Frequent Itemsets

- Only strong association rules are generated.
- Frequent itemsets satisfy minimum support threshold.
- Strong AR satisfy minimum confidence threshold.

- Confidence(A➔B) = Prob(B/A) = $\frac{\text{Support}(A \cup B)}{\text{Support}(A)}$

For each frequent itemset, **f**, generate all non-empty subsets of **f**.
For every non-empty subset **s** of **f** **do**
  output rule **s➔(f-s)** if support(**f**)/support(**s**) ≥ min_confidence
end

Recommender with Association Rules

- There exist recommender systems using statistical correlations, neural networks etc.
- Association rule based recommenders need to be trained. ➔ training set ➔ updated often
- Based on transactions user$_i$bought <i_1, i_2,…>
- If User$_x$ buys i_a and <i_a, i_b> is frequent itemset and user x never bought i_b then suggest i_b