UNIT V
ADVANCED TOPICS

ACCESS CONTROL

Privacy in Oracle•

user gets a password and user name

•Privileges :
–Connect : users can read and update tables (can‘t create)
–Resource: create tables, grant privileges and control auditing
–DBA: any table in complete DB
•user owns tables they create
•they grant other users privileges:
–Select : retrieval
–Insert : new rows
–Update : existing rows
–Delete : rows
–Alter : column def.
–Index : on tables

•owner can offer GRANT to other users as well

•Users can get audits of:
–listof successful/unsuccessful attempts to access tables
–selective audit E.g. update only
–control level of detail reported

•DBA has this and logon, logoff oracle, grants/revolts privilege

•Audit is stored in the Data Dictionary.
Integrity
The integrity of a database concerns
–consistency
–correctness
–validity

–accuracy

•that it should reflect real world

•i.e... it reflects the rules of the organization it models.

•rules = integrity rules or integrity constraints
examples

•accidental corruption of the DB,

•invalid PATIENT #

•non serializable schedules

•recovery manager not restoring DB correctly after failure
Basic concepts
•trying to reflect in the DB rules which govern
organization E.g.:
INPATENT(patent#, name, DOB, address, sex, gp) LABREQ (patent#, test-type, date, reqdr)

•E.g. rules
–lab. test can‘t be requested for non
-existent PATIENT (insert to
labreq) (referential)
–every PATIENT must have unique patent number (relation)
–PATIENT numbers are integers in the range 1
-99999. (domain)
real-
world rules = integrity constraints

•Implicit Constraints -relation, domain, referential
-integral part of Relational model –Relational constraints
-define relation and attributes supported by all RDBMS
–Domain constraints
-underlying domains on which attributes

defined

–Referential constraints
-attribute from one table referencing another

•Integrity subsystem: conceptually responsible for enforcing constraints, violation + action –monitor updates to the databases, multi user system this can get expensive
•integrity constraints shared by all applications, stored in system catalogue definedby data definition

Relation constraints

•how we define relations in SQL

using CREATE we give relation name and define attributes
E.g.
CREATE TABLE INPATIENT (PATIENT #, INTEGER, NOT NULL, name,VARCHAR (20), NOT NULL,.....gpn
VARCHAR (20),PRIMARY KEY PATIENT#);

Domain constraints

•Domains are very important to the relational model

•through attributes defined over common domains that relationships between tuples belonging to different relations can be defined.

•This also ensures consistency in typing.

E.g
CREATE TABLE (PATIENT# DOMAIN (PATIENT #) not
NULL, name DOMAIN (name) not
NULL,
sex DOMAIN
(sex), PRIMARY KEY PATIENT #);
CREATE DOMAIN PATIENT# INTEGER PATIENT# > 0

PATIENT# <10000;
CREATE DOMAIN sex CHAR
(1) in (‗M‘,‗F‘);
Referential integrity
•
refers to foreign keys
•
consequences for updates and deletes
•
3 possibilities for
–RESTRICTED: disallow the update/deletion of primary keys as long as their are foreign keys referencing that primary key.
–CASCADES: update/deletion of the primary key has a cascading effect on all tuples whose foreign key references that primary key, (and they too are deleted).
–NULLIFIES: update/deletion of the primary key results in the referencing foreign keys being set to null.
FOREIGN KEY gpn REFERENCES gpname OF TABLE GPLIST
NULLS ALLOWED DELETION NULLIFIES UPDATE CASCADES;
. Explicit (semantic) constraints
•defined by
SQL but not widely implemented

•E.g.. can‘t overdraw if you have a poor credit rating
ASSERT OVERDRAFT_CONSTRAINT ON customer,
account: account.balance >0
AND account.acc# =
CUSTOMER.cust# AND
CUSTOMER.credit_rating =‗poor‘;
•Triggers
DEFINE TRIGGER
reord-constraint ON
STOCK noinstock < reordlevel
ACTION ORDER_PROC (part#);
Static and Dynamic Constraints
•State or Transaction constraints
static refer to legal DB states

•dynamic refer to legitimate transactions of the DB form one state to another ASSERT payrise
-constraint ON UPDATE OF employee:
•Security can protect the database against unauthorized users.
•Integrity can protect it against authorized users
–Domain integrity rules: maintain correctness of attribute values in relations
–Intra
-relational Integrity: correctness of relationships among atts.
in same rel.
–Referential integrity: concerned with maintaining correctness and consistency of relat
ionships between relations.
Recovery and Concurrency
•closely bound to the idea of Transaction Processing.
•important in large multi
-user centralized environment.
Authorization in SQL
•File systems identify certain access privileges on files,
E.g
.,
read, write, execute.
•In partial analogy, SQL identifies six access privileges on relations, of which the most important are:
1.SELECT = the right to query the relation.
2.INSERT = the right to insert tuples
into the relation
–may refer to
one attribute, in which case the privilege is to specify only one column of the inserted tuple.
3.DELETE = the right to delete tuples from the relation.
4.UPDATE = the right to update tuples of the relation
–may refer to
one attribute.
Granting Privileges
•You have all possible privileges to the relations you create.
•You may grant privileges to any user if you have those privileges

—with grant option.‖
u You have this option to your own relations.
Example
1.Here, Sally can query Sells and can change prices, but cannot pass on this power:
GRANT SELECT ON Sells,
UPDATE(price) ON
Sells TO sally
DISTRIBUTED DATABASES VS CONVENTIONAL DATABASES

Data Warehouse:

Large
organizations have complex internal organizations, and have data stored at
different locations, on different operational (transaction processing) systems, under
different schemas
¬Data sources often store only current data, not historical data
¬Corporate dec
ision 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 When and how to gather data

•Source driven architecture: data sources transmit new information to warehouse, either
continuously or pe
riodically (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.Typically warehouse data is multidimensional, with very large fact tables
•Examples of dimensions: item
-id, date/time of sale, store where sale was made, customer identifier
•Examples of measures: number of items sold, price of itemsDimension values are usually encoded using small integers and

mapped to full values via dimension tablesResultant schema is called a star schemaMore complicated schema structures
•Snowflake schema: multiple levels of dimension tables
•Constellation: multiple fact tables

Data Mining

¬Broadly speaking,
data mining is the process of semi
-automatically analyzing large
databases to find useful patterns.
¬Like knowledge discovery in artificial intelligence data mining discovers statistical rules
and patterns
¬Differs from machine learning in that it deals wit
h large volumes of data stored primarily
on disk.
¬Some types of knowledge discovered from a database can be represented by a set of
rules. e.g.,: —Young women with annual incomes greater than $50,000 are most likely to
buy sports cars‖
.¬
Other types of knowledge represented by equations, or by prediction functions
.
¬Some manual intervention is usually required
•Pre-processing of data, choice of which type of pattern to find, postprocessing to
find novel patterns
Applications of Data Mining
¬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 customer is likely to switch brand loyalty
•Predict if a customer is likely to respond to —junk mail‖

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

¬Some examples of prediction mechanisms:

¬Classification

•Given a training set consisting of items belonging to different classes, and a new item whose class is unknown, predict which class it belongs to.¬Regressionformulae

•Given a set of parameter
-value to function
-result mappings for an unknown
function, predict the function
-result for a new parameter
-value

¬Descriptive Patterns
Associations

•Find books that are often
bought by the same customers. If a new customer buys
one such book, suggest that he buys the others too.

•Other similar applications: camera accessories, clothes, etc.
Associations may also be used as a first step in detecting
Causation

•E.g. association be
tween exposure to chemical X and cancer, or new medicine
and cardiac problems 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 a set of 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 knowledg

e, such as educational level of applicant, salary of applicant, age of applicant, etc.
person P, P.degree = masters and P.income > 75,000
⇒ P.credit = excellent
person P, P.degree = bachelors and(P.income≥25,000 and P.income ≤75,000)
⇒P.credit = good
¬Rules are not necessarily exact: there may be some misclassifications

¬Classification rules

can be compactly shown as a decision tree.

Decision Tree
•Training set
: a data sample in which the grouping for each tuple is already known.
•Consider credit risk example: Suppose degree
is chosen to partition the data at the root.

¬Since degree has a small
number of possible values, one child is created for each value.
•At each child node of the root, further classification is done if required. Here, partitions are defined by income
.¬Since income is a continuous attribute, some number of intervals are chosen,and one child created for each interval.
•Different classification algorithms use different ways of choosing which attribute to partition on at each node, and what the intervals, if any, are.
•In general
¬Different branches of the tree could grow to different levels.
¬Different nodes at the same level may use different partitioning attributes.
•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 conditionfor the node¬More on choosing partioning attribute/condition shortly¬Algorithm is greedy: the choice is made once and not revisited as more of the tree is constructedThe data at a node is not partitioned further if either ¬All (or most) of the itemsat the node belong to the same class, or ¬All attributes have been considered, and no further partitioning is possible. Such a node is a leaf node. Otherwise the data at theode is partitioned further by picking an attribute for partitioning data at the node.Decision-Tree Construction Algorithm

Procedure Grow.
Tree(S)Partition(S);
Procedure
Partition (S)
116
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
Sinto
S_1
, S_2
, ..., S
$_r$
,
for
i
= 1, 2, ...,

r
Partition(
S
i
);
Other Types of Classifiers
Further types of classifiers
¬
Neural net classifiers
¬
Bayesian classifiers
Neural net
classifiers use the training data to train artificial neural nets
●
Widely studied in AI, won‘t cover here
Bayesian classifiers use
Bayes theorem
, which says
where
p(c_jd) = probability of instance
d being in class
c_j,
p(d | 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
doccuring
Naive Bayesian Classifiers
Bayesian classifiers require
¬computation of
p(d| 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

$p(dc_j) = p(d_1|+c_j) *$

$p(d_2c_j) * ....* (p(d_nc_j)$

Each of the

$p(d_i|c)$ 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

$_2$, ..., X_n

, we wish to predict the value of a

variable Y.

One way is to infer coefficients a

$_0$, a_1, a_1, ..., a_n

such that

$Y= a_0+ a_1* X_1+a_2* X_2 + ... + a_n * X_n$

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 RulesRetail shops are often interested in associations between different items that people buy. Someone who buys bread is quite likely also to buy milkA person who bought the book Database System Conceptsis 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
⇒milk DB
-Concepts, OS
-Concepts
⇒Networks
Left hand side: antecedent, right hand side: consequentAn 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 nfidence.
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 ⇒screwdrivers is low.We usually want rules with a reasonably high support Rules with low support are usually not very useful Confidence is a measure of how often the consequent is true when the antecedent is true.

E.g. the rule bread
⇒milk has a confide
nce of 80 percent if 80 percent of the purchases that
include bread also include milk.
Usually want rules with reasonably large confidence.
Finding Association Rule We are generally only interested in association rules with reasonably high support (e.g. support of 2% or greater)
Naïve algorithm
1.Consider all possible sets of relevant items.
2.For each set find its support (i.e. count how many transactions purchase all items
in the set).
HLarge itemsets: sets with sufficiently high support
3.Use large itemsets to generate association rules.
H From itemset
A generate the rule

A -
{b}
$\Rightarrow$ b
for each
b $\in$ A.
4 Support of rule = support (A)
.4 Confidence of rule = support (
A) / support (A
- {
b})
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 (prob1 * prob2) 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 tem poral 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

E.g. Group points into k sets (for a given k
) such that the average distance of points from
the centroidof 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
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) cluste rs into smaller ones.

OBJECT ORIENTED DATABASE

Basically, an OODBMS is an object database that provides DBMS capabilities to objects that have been created using an object -oriented programming language (OOPL). The basic principle is to add persistence to objects and to make objects persistent.
Consequently application programmers who use OODBMSs typically write programs in a native OOPL such as Java, C++ or Smalltalk, and the language has some kind of Persistent class, Database class, Database Interface, orDatabase API that provides DBMS functionality as, effectively, an extension of the OOPL.Object -oriented DBMSs, however, gomuch beyond simply adding persistence to any one object -oriented programming language. This is because, historically, many object-oriented DBMSs were built to serve the market for computer -aided design/computer -aided manufacturing (CAD/CAM) applications in which features like fast navigational access,versions, and long transactions are extremely important.

Object-oriented DBMSs, therefore, support advanced object-oriented database applications with features like support for persistent objects from more than one programming language, distribution of data, advanced transaction models, versions, schema evolution, and dynamic generation of new types.Object data modelingAn object consists of three parts: structure (attribute, and relationship to other objects like aggregation, and association), behavior (a set of operations) and characteristic of types (generalization/serialization). An object is similar to an entity in ER model; therefore we begin with an example to demonstrate the structure and relationship.
Attributes are like the fields in a relational model.
However in the Book example we have, for attributes publishedBy and writtenBy, complex types Publisher and Author, which are also objects. Attributes with complex objects, in RDNS, are usually other tables linked by keys to the employee table.Relationships: publish and writtenBy ressociations with I: N and 1:1 relationship; composed of is an aggregation (a Book is composed of chapters). The 1: N relationship is usually realized as attributes through complex types and at the behavioral level. For example
,Generalization/Serialization is the is a relationship, which is supported in OODB through class hierarchy. An ArtBook is a Book, therefore the ArtBook class is a subclass of Book class. A subclass inherits all the attribute and methodof its superclass.
Message: means by which objects communicate, and it is a request from one object to another to execute one of its
methods. For example:Publisher_object.insert (||Rose||, 123...) i.e. request to execute the insert method on a Publisher object)
Method: defines the behavior of an object. Methods can
be used to change state by modifying
its attribute values
to query the value of selected attributes The method that responds to the message example is the method insert defied in the Publisher class.