MOUNT ROYAL UNIVERSITY

Department of
MATHEMATICS AND COMPUTING

COMP 4522

Database II: Advanced Databases

Contents

EXERCISE

SEVEN

DATA WAREHOUSING

Understand the need for data warehousing
Three level data warehouse architecture
Create a simple dimensional model
Trends and future
Define: data warehouse, meta-data, operational system, informational system, data mart, real-time data ware house, star schema, fact table, dimension table

What is a Data Warehouse?

A Data Warehouse (DW) is a database designed for analysis and reporting, in support of decision making, rather than transaction processing. Usually it is the basis for a company's Business Intelligence (BI) or Business Analytics system.

A data warehouse can be described as a *subject-oriented, integrated, time variant, non-updatable* collection of data in support of decision making.

Subject-oriented The data in the data warehouse is organized so that all the data elements relating to the same real-world event or object are linked together. For example, students, customers, courses, products.

Integrated The data warehouse contains data from most or all of an organization's operational systems and these data are made consistent. This is a simple statement but not so simple in practise. Still, one does what one can.

Time-variant An operational system maintains the current value. Think inventory. Operational systems sometimes contain historical data. Banner has grades from way back. But they do not have to. The data warehouse stores the historical data so that trends over time can be analyzed.

Non-volatile A transactional system is in constant flux — if anyone is using it and hopefully they are, think of the Banner registration system or Chapers.ca sales and inventory systems — the data is never static.

The data warehouse is really a read only system so it is stable. It will be refreshed daily, weekly, monthly as appropriate but while you are working on it, the data is stable. Normally a data warehouse is a monotonically increasing data set. Data gets added but never taken away.

The Need for Data Warehousing

Organizations need speed and flexibility in decision making. To make better decisions managers need access to the quality, integrated data that a data warehouse provides. There are some trends that increase this need.

@ 2020 Alan Fedoruk

No System of Record Any non-trivial organization will have multiple systems in use and no single system captures the current state of the organization. Systems are in-house and cloud-based, when organizations merge the information systems are often set up to coexist.

Multiple Sytems Not Synchronized No matter how hard you try you will not be able to keep all of an organizations systems synchronized. Different systems have different update cycles, different data formats, different coding schemes, and different data needs.

Analysis in a Balanced Way

CRM Organizations need a total picture of their relationship with customers. Different operational systems capture data from different touch points as a customer interacts with the organization but an overall picture of the customer is needed.

SRM Similarly with supplier relationships. Supply chain management is crucial to organizations.

Data Warehouse Goals

From Kimball & Ross 2013

Accessible To business people, not just developers. Must be simple and fast.

Consistent Quality assurance is top priority. Integrating systems so we need consistent labels and terms.

Adaptable DW will be around for the lifetime of the company. Cannot let changes to the business structure or operational systems render our DW obsolete. Must be able to handle new queries.

Timely No good if you need to wait a year for data to appear in the DW.

Secure An organizations data is one of its most valuable assets. It must be protected from theft and corruption.

Authoritative It must have the right data. The output of a DW is *decisions*.

Accepted The business people must accept the DW as the source for BI data otherwise, there really is no point.

Data Warehouse Architectures

Data warehousing as an idea has been around since 1988 (Devlin and Murphy). Inmon wrote an early text on the subject in 1992. Since then a variety of architectures have been proposed and adopted and evolved. Typically data warehouses use a three level architecture.

A Data Warehousing Example

In the latter half of the 1990s I worked for the Ministry of Environment, Lands and Parks (MELP) at the regional office in Nelson, BC. My role was partly database administrator and partly data administrator. MELP had implemented a data warehouse.

Operational systems were mostly located in Victoria, some very old, and some newer, and were for such things as hunting licenses, taxidermy permits, pollution permits, fishing licenses, water permits, leases (grazing, heli-skiing), trap-line data, research studies, etc. etc. Data from these systems was extracted monthly into a MELP wide data warehouse. Then the data were partitioned by region (8 of them if I remember right) and those partitions where sent to servers at each of the regional offices.

The data in the local warehouse was then integrated with GIS data to provide and analysis tool that could be used by anyone at their desk.

The data in the warehouse was essentially wide denormalized tables. Many records had geo-coordinates to allow data to be used in a GIS.

At the time it was quite innovative and successful.

One of the projects I focused on during that time was creating a meta-data repository. It is all well and good to have all this data, but the meta-data is vital to being able to use it effectively. At one point I worked with the Conservation Officer service to analyze hunting licenses and taxidermy permits and records to catch poachers. Someone wanting a taxidermist to mount an animal must see the valid hunting permit before doing the work. If the animal was poached, there would not be a valid permit. Poachers were reusing permits and through the DW the COs were able to catch them.

The levels in the DW architecture are usually

1. Source data systems. Operational Data.

@ 2020 Alan Fedoruk

2. Data staging area (operational data store). Reconciled Data.
3. Data and meta-data storage area – the DW. Derived Data.

Data from the source data systems are *Extracted* into the staging area. There the data is cleaned, combined, reconciled, derived, summarized and in a word, *transformed*. Data from there is then *loaded* into the data warehouse.

This cycle of extract, transform and load, or ETL is extremely important. If it is not done well the data warehouse will not be very useful. Since you are using the DW for decision making, having poor data from a poorly created T process will be bad.

Differences between Operational Systems and Data Warehouse Systems

Operational systems are the ones that users interact with to make changes (transactions) to the underlying databases. They are optimized for consistency and transaction throughput.

A data warehouse system is used for decisions making, analysis and prediction. It is optimized for efficient querying and integration of data.

- A transactional system is usually *normalized* to prevent update/insert/delete anomalies, that is to maintain a high level of consistency. This is important. Since the DW is not used for updates we can relax things and have un-normalized data for ease of use and performance.
- We know that having a fully normalized database will be consistent, but it can be slower and more awkward to work with due to the large number of joins required. (E.g. PERSON, ADDRESS, EMAIL, PHONE, etc.) all required to be accessed to produce a simple list. In the data warehouse you are more concerned with efficient access and access in different ways so data can be denormalized and sometimes there are duplicated data!
- Calculated or summarized data is avoided in a transactional system because it gets out of date and requires compute power to recalculate. Since the data warehouse is static we can include calculated fields such as aggregate or summary information.
- Typically a transactional system will only access a few rows of data at a time. Data warehouse queries access many rows at once. e.g. Average grade in 1501 from first offering until now.

Variations

There are a number of variations on the basic DW structure.

The frequency of the ETL is very important. Data can be extracted and loaded yearly, by semester, monthly, daily, etc. It can even be near continuous. Some data have a natural frequency, such as semesters or academic years for a university. Companies may want quarterly or monthly data. Some data warehouses are essentially real-time data warehouse with continuous feeds of data from operational systems.

Sometimes one operational unit may not need access to the entire DW. This was the example from MELP above. In the Nelson regional office, there was little need for data from Smithers. In this case the DW may be broken up into smaller *data marts*. These may be physical data marts (like the MELP example) or logical data marts, that are essentially views of the main DW.

Data marts can be constructed on a location, a function, an entity. They may contain summary rather than detail information and have less historical data.

Metadata

Metadata are technical and business data that describe other data. Each of the layers must have accurate, detailed and complete meta-data. It is vital that users of the DW can access metadata to ensure that the reports, analysis and predictions they are doing are an appropriate thing to do with the DW.

Data Warehouse Schemas: The Dimensional Model

Just as there is more than one way to skin a cat, there is more than one way to create a data warehouse. We will look at one way: Dimensional Modeling. This technique was first popularized by Kimball and has been widely adopted. Implementing a dimensional model in a relational database results in a *fact table* with a number of *dimension tables* radiating out from it. A person with imagination can see this as a star some and sometimes we talk about *star schemas*.

@ 2020 Alan Fedoruk

Fact Tables

At the centre of the star is a subject based *Fact Table*.

A fact table is a table of measurements of a particular business process. A grade in a course, a sale, etc. The fact table contains two types of columns: a measurement(s) and foreign keys to the dimension tables. For our example the measurement could be the grade that a particular student got in a course, or some aggregation: the average/median grade in a course, the number of a particular grade for example.

It is important to determine what 'grain' the fact table will have. That is, what level of detail will it contain? Grades for every student? Average grades for a section or course? Usually the finer the grain the better (more flexible, more scalable). Remember that with a fine grain you can always aggregate data in your queries, but with a coarse grain, you cannot de-aggregate!

Measurements are usually numeric and additive, though it is possible to have non-numeric and non-additive measurements.

Fact tables normally have a large number of rows.

The primary key for a fact table is the combination of all the foreign keys to the dimension tables.

Dimensions

You create a dimension table for each dimension of the data you are interested in.

While fact tables typically have a small number of columns and a large number of rows dimension tables can have a large number of columns and not that many rows. Dimension tables are not typically normalized. Some people do normalize them creating a 'snowflake', but there is no real reason to do so, except clinging to the idea that all data muse be in 3NF.

The dimensions are how you select, group and aggregate data in your queries.

The Cube

If you have 2 dimensions, for example a fact table like: COURSE(Course_ID, Term_ID, Avg, Med) With dimension tables Course and Term. You have

created a square. One side is the course side/dimension the other is the term side/dimension. The cell (the facts) at the intersection of the COMP1501 and 201203 is the Avg and Med for that course in that term.

If we add another dimension, Instructor, we have created a cube.

In general a fact table with n dimension tables creates an n-dimensional data cube.

Querying the DW then becomes a process of slicing and dicing the cube.

Galaxies

A galaxy of stars. It is possible to have multiple fact tables. Usually these would rely on some or all of the same dimension tables.

Indexes

Almost every query you perform against your data warehouse will involve a join between a fact table and one or more dimension tables. Fact tables can have millions of rows. You need to be concerned about query performance.

Bitmap Indexes are commonly used in star schema DWs. Bitmaps are very fast and take less space, for large tables, then do typical B-trees indexes. Oracle does some query optimizations when you use bitmaps in a star schema.

Bitmaps are a bit string as long as the number of rows, with a 1 set if that row has that value. E.g. Province codes. Term codes, Instructors, Programs.

Bitmaps are really code for selecting on multiple values (dimensions) which is something you do a lot of in a DW!

Also good for counting. Just add up the 1s.

Bitmaps work well for domains with a small number of values, not so much for domains with a large number of values.

EXAMPLE

Imagine a student information system that can perform the following operations:

@ 2020 Alan Fedoruk

- Admit a new student to a program.
- Create a new course in the calendar of courses. Courses have names, descriptions, number of credits and zero or more prerequisite courses.
- Maintain a list of the faculty employed at the institution.
- Create a schedule of classes for a term. There are three terms per year. Not all courses are offered in any given term. Each section is given a unique CRN. A section is taught by one faculty member.
- Students can register in sections of courses. For each section a percentage grade is recorded.

Logical Model:

- Students (ID, Name, Program)
- Course (ID, Name, Description)
- PreReg(ID, PID)
- Section (CRN, Term, CourseID, Instructor)
- Faculty(ID, Name, Degree)
- Registration(SID, CRN, Grade)

DIMENSIONAL MODELING

Four step process:

1. Select the business process. These are the things that the organization does. Grade a student, sell a product, complete a project. You repeat this for all the processes you are interested in.
2. Declare the grain. Usually start with atomic grain and only make it coarser in special circumstances.
3. Identify the dimensions. This is what will determine how the DW can be used.
4. Identify the facts. Be sure to understand which of your facts are additive and which are not.

Data Lakes and Swamps

A well structured and maintained data warehouse can be a boon for many organizations. But, creating and maintaining a well structured and up to date data warehouse is hard.

- They are complex
- The ETL process is expensive to keep up to date and running properly
- Data analytics requirements change rapidly and an enterprise data warehouse can be slow to respond
- The initial and ongoing costs make them effective mainly for large organizations
- The structure can become rigid and difficult to modify, lock in can occur

Over time, with readily available methods to efficiently store and retrieve large amounts of data (Hadoop for example), pressure to provide analytic results quickly, and NoSQL and other semi or unstructured databases, other techniques have emerged to compete with the more traditional data warehouse.

One of the main ones is the data lake. In a data lake, data is stored in its natural format. A data lake can contain structured relational table, semi-structured data like JSON, CSV, XML etc. documents, raw data files, binary data and objects (blobs), and so on. The idea is that this type of structure will not have the high overhead and rigidity of the data warehouse and will provide responsive data analytics to the organization.

Data lakes are not without their problems. A huge repository of unstructured data can very easily become a data graveyard or data swamp.

Data warehouses and data lakes will both be around for a while and organizations can find the right balance between the two.

For any analytics solution there is an issue that is of utmost importance and that is the metadata. Metadata is of course the data about the data. Often organizations will implement some sort of a data dictionary that keeps information about origin, meaning, ownership, quality and quantity, date ranges etc. This is very much akin to a library catalogue.

@ 2020 Alan Fedoruk

Keeping this type of data store up to date is a thankless task, but is it vital to ensuring that your analytics output is reliable.

@ 2020 Alan Fedoruk

EXERCISE

EIGHT

DATA MINING

What is data mining
Basic procedures

DATA MINING IS A PROCESS of discovering and extracting patterns in large data sets. To put it in its proper context, let's step back a bit.

Artificial intelligence (AI) is the computing discipline that creates and investigates algorithms that can perform tasks or solve problems intelligently. *Machine Learning* algorithms are a sub set of AI algorithms that learn from data and create models that can be used to make predictions and decisions without being explicitly programmed. There are different learning algorithms many of which are based in statistical and mathematical methods.

Data mining uses machine learning, descriptive statistics, like means, medians, and distributions, predictive statistics, such as regressions, data management, database management and data processing. The goal is to discover patterns or knowledge from the data to enable you to make better business decisions.

Data mining is a bit of a misnomer. If you have ever panned for gold, *gold mining*, you are sifting through gravel to find gold (gold is much heavier than the gravel so when the gravel is washed, gold ends up at the bottom and can be collected. If we were using the same logic for data mining we would call gold mining gravel mining. We are not mining for data (gravel), we are mining for knowledge or patterns (gold). So really it should be knowledge mining, but sometimes terms get into common usage and we are stuck with them.

Data Science is emerging as a name for the whole field. In fact MRU is preparing a proposal to offer a BSc in Data Science. Students in that program would study programming, databases, mathematics, statistics, data wrangling, machine learning, data visualization and very importantly, ethics. In 2012, an article in the *Harvard Business Review* published an article, "Data Scientist: The Sexiest Job of the 21st Century". Demamnd for data scientists is large and growing.

Another way to think of data mining is as applied data science.

THE DATA MINING PROCESS

In this section I will outline, in general, how data mining works, the inputs, outputs and processing.

@ 2020 Alan Fedoruk

Inputs

What are the inputs to a data mining project?

You need a problem that you are trying to solve. Maybe you would like to identify spam emails, or you want to be able to classify potential customers as good credit risks or poor, or you want to identify plant specimens based on certain measurements, or you want to model the behaviour of a natural phemomena, such as a geyser, or you want to determine if their are patterns in the items people buy from your store, or if the words in Twitter tweets have some relatonships.

The next thing you need is some relevent data. When it comes to data, usually more is better! While you don't necessarily need huge data sets, if more data is available you should try to get your hands on it.

Data and Data Wrangling

As you know, data can take many forms, but for many data mining projects the data is typically tabular. The data can be stored in a database, usually relational but could be a NoSQL database, a collection of structured text files like **.csv** or JSON. Typically data will be structured something like this:

ID	$Attr_1$	$Attr_2$	...	$Attr_n$
1	AAA	BBB	...	999
2	CBS	CDE	...	21
...				
m	ZZZ	YY	...	42

Each row represents an instance of whatever we are working with. There is typically an ID and then some number of attributes that describe the instance. One or more of the attributes may be a classification, like the species of a flower, a credit risk, whether or not a passenger on the Titanic survived.

The collection of items is sometimes referred to as the population, P.

Data that you have availble is not necessarily in nice tabular format and may need to be drawn from several sources, sometimes from disparate databases. Using data from different sources, even within the same organization can introduce inconsistencies such as different coding schemes, different standards or domains for the dame data.

You will need to work with the raw data and to 'wrangle' it into a form

that your data mining program can use. This can involve SQL queries. NoSQL queries, web searches and downloads.

Data Exploration

You must understand your data.

You must understand your data.

To understand the data, you need to explore it. If you are working in SQL you can use SQL queries and if you are working in Pandas you can use Pandas queries and graphs.

You will at least want to find out

- how many items (rows) you have;
- what attributes are present;
- the domain of each attribute, especially if it is numberic or text, if it is categorical, or ordinal, real or integer;
- basic descriptive statistics for numeric attributes;
- distributions for categorical, ordinal and numeric attributes;
- relationships between attributes.

Graphs and other visualizataions are very helpful here. Python provides quick and easy ways to plot data.

The Titanic example shows the basics of data exploration for that data set. Each data set is different and you need to spend time with the data to really understand what is there.

As you are exploring you will find some issues that you need to deal with.

Nulls

Then once you have a table close to what you need, you need to clean the data. One of the most common issues is null values. Some data mining algorithms can handle null values but others cannot. You have some choices when trying to clean up null values.

- Discard the rows with null values. This is an easy fix, but if there are a lot of rows with null values you risk deleting valuable data.

@ 2020 Alan Fedoruk

- Replace the null values with a marker value. For example, if you have a numeric column with null values you could replace the nulls with -1, -999 or 0. This is good, but it could mess up your decriptive statistics.

- Null values can be replaced with some other value, for example, null values could be replaced with the mean of the non-null values, or the minimum, or maximum or some other appropriate value.

Consistency

Another issue that you might need to do is to make the data consistent. For example, you may have an attribute that is province codes, some of which may be upper case and some lower case. You will need to convert all of the codes to either upper or lower case to be consistent. This is a simple example and there are others that may involve a lot more computation. You may have data from two different systems, one that uses 'M', 'T', 'W' etc. to represent days of the week and another that uses 'MON', 'TUE', 'WED' etc. These will need to be brought into agreement.

Errors

You may discover that there are errors in the data. These can be corrected in ways that are similar to dealing with nulls. You can also attempt to correct the data, if that is possible.

Similar to errors are outliers. You may find data items that are wildly different than the majority. Judgement is needed for these. It may be that they truly are errors and should be deleted, but it is hard to know for sure if they are not valid data.

Wrangling

This process of obtaining, examining and exploring the data is collectively known as Data Wrangling. In the data warehousing world, this is referred to as ETL or Extract, Transform, Load. Data is extracted from systems, it is transformed and loaded into the data warehouse. This is a similar process though you the data may be going into a data mining program rather than a data warehouse. Sometimes the data you are using comes from a data warehouse.

The wrangling or transformation part of a project can consume a lot of time and resources. It is difficult, sometimes tedious work that requires patience and attention to detail. But it is also vital that is it done and done well. If you don't understand the data and have confidence in it, there can be no confidence in the result. Time spent in wrangling will more than pay for itself when you start the data mining.

Training and Testing Data

For many data mining applcations you will need to create some subsets of the data.

Training Set The training set is the data that you will use to train the algoritm and create the model. For some applications, like classifications, the training set will include a target attribute that is the classification of the item. For example, in the iris data set, the species is the target.

Testing Set The testing set is used to test the model. It would normally include the target, if that is appropriate.

Target Set This is the data that you will use the model on.

There are many different ways to divide the data into training and testing sets. For example, the training set could be a random sample from the entire data set that is 50% of the total population of items and the testing set would be the other 50%. Many strategies for doing this are possible, including multiple training sets, and multiple testing sets.

The Data Mining Process

The process is a bit different depending on the particular method being used but there are some common processes.

If you have your data, cleaned and broken up into training and test sets the process goes like this:

1. Adjust the data mining algorithms parameters.
2. Run the data mining algorithm and create a model.
3. Use to model against the test set to evaluate the model.
4. If the performance of the model is acceptable, stop, if not, go back to 1.

@ 2020 Alan Fedoruk

TYPES OF MINING ALGORITHMS

In this class we examine three representative data mining algorithms. There are many others, but these, or variants of them are freqeuently used and encountered.

- Classification. This invovles building a model that will allow items to be classified by their attributes. There are different ways to classify, but we look at decision trees. A decision tree consists of interior nodes that are predicates on the attributes that determine which child node to proceed to. Leaf nodes are the classification of the item.

  Classification is a form of supervised learning becuase you provide the classifications before hand.

  Classification has wide applicability.

- Associations. This looks at sets of items, the common metaphor is a shopping basket and tries to find rules that are likely to occur together. The most common algorithm is Apriori and its variations.

- Clustering. Clustering is related to classification in that it attempts to put items into categories, but in this case the categories are not determined ahead of time. The algorithm uses the data to find the categories.

  Clustering is a form of unsupervised learning and has applicability to a wide array of situations.

@ 2020 Alan Fedoruk