# LO 1 - Describe the components and function of database management systems.

### Terminology

**Database**

* Self-describing collection of integrated tables (Kroenke)
* A shared collection of logically related data and its descriptions, designed to meet the needs of an organization (Connol / Begg)
* An organized collection of data, today typically in digital form. The data are typically organized to model relevant aspects of reality, in a way that supports processes requiring this information (Wikipedia)

**Metadata**

* Data about data, stored in the system catalog/data dictionary
* System catalog or data dictionary
* Self-describing and independent nature of a database provides program data independence

**DBMS – Database Management System**

* Examples of such systems include: Access, Oracle, SQLite, MySQL, SQL Server, DB2, PostgreSQL (postgres)
* Going to be used by an organization and define who can use what, and query it
* A set of programs used to define, administrate, and process database data and applications
* The software that manages and controls access to the database (Connolly / Begg)
* Translates the users data requests to the physical storage. – Queries, etc.

**RDBMS -** Relational Database Management System – This class will focus heavily on this

* Organizes the data into tables( a relation)
* DBMS that organizes the data using relations (tables)
* Other DBMS techniques include (<http://en.wikipedia.org/wiki/Database> AND <http://en.wikipedia.org/wiki/Data_model>):
  + Flat Files
  + Hierarchical
  + Network databases
  + Object-relational

**Schema** – another word for the entire structure of the DB, how the databse is structured is the schema

* The entire structure of the database including all tables, attributes, relationships, etc

**Relation** – A table that has rows and columns effectively

* The definition of a table with columns (attributes, fields) and rows (records, tuples) but not the data.

Example: Author Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **auID** | **auLName** | **auFName** | **Phone** | **Address** | **City** |
|  |  |  |  |  |  |

**Row, record, entry, tuple** – stores data about a specific item or instance, 1 person in a list of people

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **auID** | **auLName** | **auFName** | **Phone** | **Address** | **City** |
| 001 | Smith | John | 333-3333 | Box 123 | Saskatoon |
| 002 | Brown | Jane | 444-4444 | Box 321 | Saskatoon |

**Column, field, header, attribute** – describe the type of data(string of characters, numbers, etc) that can be stored in that column for all items in the table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **auID** | **auLName** | **auFName** | **Phone** | **Address** | **City** |

**Constraint** – restrictions on data in a table. – can only be between 0-100, wtc

-Rules that keep data in a DB as valid as possible.

-EX: no duplicates in a PK field

**Domain** – All the valid choices you can make for that data

* The set of allowable values for one or more attributes.
* the datatype (example: char, varchar, number, date),
* the size of the datatype (example: char(10)),
* whether the value is unique,
* whether the value exists in another table,
* whether the value has other constraints like business rules (**Check Constraint**). For example: An attribute can only be assigned a value between 0 and 9999.

**Superkey** – An attribute or set of attributes that can uniquely identify a row in a table. Difference between this and primary is that superkey can include other stuff that you don’t need. A superkey may contain additional attributes that are not necessary for uniquely identifying the row. (This could be all attributes of a row).

**Candidate Key** – Minimum set of attributes to uniquely identify a row in a table. Might have multiple attributes for primary key. You have multiple choices. A table may have several candidate keys, of which one usually is to become the PK. EX(part number, in your system, or Make number from manufacturer)

**Primary Key** – uniquely identifies a row or record in a table (A customer has a primary key as custID) –One of the candidate keys is selected to be the primary key, to uniquely identify a row, does not allow nulls.

**Foreign Key** – If you put a primary key into another table, primary key from table 1 in table 2 for linking purposes – A primary or unique key from a parent table that exists in the child table for the purpose of linking the tables together.

**Composite Key** – Any of the other keys that have 2 parts, EX first and last name is the Primary key – A key made up of more than one attribute.

**Unique Key** – Something that will be managed to be unique. A way tomake values in a coloumn unique. – An attribute or set of attributes that is unique in a table( no two records can have the same value), does allow null(null would be a unique value.

**Surrogate Key** – Often used when you don’t have a good candidate key, or when you don’t want to use them. So you come up with one, an Autonumber. Generated value that is to uniquely identify a row, not derived from application data. Usually created to act as PK, almost always a sequential number.

**SQL** – Structured Query Language, sequel, and most common language for interacting with a relational database.

**Components of SQL**

* **DDL** – Data Definition Language –Commands used to specify the schema Ex: CREATE TABLE statement
* **DML** – Data Manipulation Language – Commands used to work with data, Read and update the database data (SELECT, INSERT, UPDATE, DELETE)
* **DCL** – Data Control Language – Controls permissions in the database (GRANT, REVOKE)

**4GL** – Fourth Generation Language (SQL). Non-procedural language concerned with the “What” you want to do, and not the “how”.

**Transaction –** A unit of work that either succeeds or fails in whole. Either all commands succeed, or none of the commands occur (rollback)

**Data mining** – Analyzing collected data (often stored in a data warehouse, very large) for patterns.

**Big data** – Collection of large and complex data sets to which data mining is usually applied.

**Personal Notes: Hierarchical Mode- Family Tree**

## 1.2 Discuss the historical development of DBMSs and the relational model and other database models

http://www.computerhistory.org/revolution/memory-storage/8/265/2207

**File-based systems**

-First type of digital database system

-Emulated manual filing systems

-Flat filing system

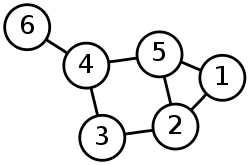
-Common tasks for file-based: to want to put in a new record, find location to put record, put new record in, and push other records “down”

**Limitations of the File-Based Approach**

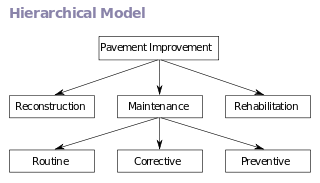
* Separation and isolation of data
  + Since the data was kept in separate files, it was difficult to get information out because significant processing was needed to derive the information.
* Duplication of data
  + Often file-based systems are built on a department basis, so that data was often duplicated between departments.
  + Duplication often results in wasted time, space, loss of data integrity.
* Data dependence
  + Physical structure and storage of the data files is defined by the application. Changing the structure requires a lot of work.
* Incompatible file formats
  + Structure of the data is dependent on the program and programming language.
* Fixed queries – prob biggest real limitation
  + The information that can be derived is dependent on the application developers writing code to develop the reports. There is no facility to do ad hoc reporting.

To solve the limitations of filebased systems the Database concept started in the 1960s.

The first generation of database systems were navigational , where applications accessed data by following pointers from one record to another. Ie. Node6.Node4.Node5.Node1



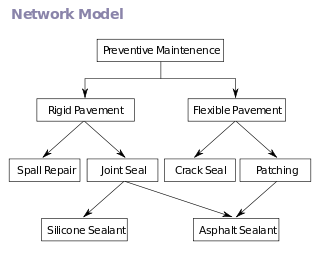
There were two main types of navigational databases, the network model (Codasyl a consortium model first proposed by Charles Bachman) and the hierarchical (IBM IMS system) model.

**The hierarchical model**

<- Root node

* Tree based
* Each record can only have 1 parent but may have multiple child nodes.
* Windows registry

**The network model**



* Graph based
* Allows each record to have multiple parents and child records

These systems were used until the 1980s where they were eventually replaced with relational databases. There was a resurgence of the hierarchical data systems in the late 1990s with the introduction of XML based systems.

**Relational Database Model – The focus of this class**

The relational database model was proposed in 1970 by Edgar Codd.(names and dates don’t matter to much, just know the what) This model departed from the navigational model.

Relational database Models:

* Has the idea of primary and foreign keys, and searching based upon content
* Specified that applications search for data by content ( not by following links)
* Primary and foreign keys are used instead of pointer to define relationships between entities (tables)
* This allows the content of the database to evolve without rewriting applications

>the use of PK and FK made searches using SQL possible.

>The relational database is a good model for searching for content, but is less well suited for modeling data (giving a visual of what the data looks like). For this purpose the entity relationship model was developed in 1976.

Order

OrderId

OrderNum

Customer

CustID

CustNum

*Advantages of RDBMS (relational database management system)*

* Control data redundancy
* Improved data integrity/consistency
* Increased productivity since the DBMS provides the low-level file handling
* Improved ability to share data
* Improved Security

*Disadvantages*

* Complexity
* Cost of software and hardware
* High impact of a failure
* Scalability (Over multiple servers) – Though many vendors have products that address this

**Object Oriented Database**

My shit > Make the “tables” of a DB to classes, from coding AKA Java. So one table of customers would be a Customer table, with an array of “orders” instead of having a 1 to many relationship in the relationship model/ERD.

Teachers- Object oriented databases were first defined in 1985 and allows for information to be represented in the form of objects as used in object oriented programming. Most object databases offer somekind of query language buy generally make use of object query language. Since object databases store both the complex and the relationships between data directly, without having to map to rows and columns, this allows them to work well with very complex data.

OQL- Object Query Language

Potential advantages

* Works well with objects, with inheritance etc
* Objects don’t require assembly nor assembly. Don’t need to be mapped from the table to the object. Saving some coding and possibly execution time
* Less code is required when applications are object oriented (using an OQL DB and OOP application is easier)
* ORM- Object Relational Mapping- Takes ERD/Class diagram and writes code to map between DB and application.
* Easier navigation
* Better concurrency Control – a hierarchy of objects may be locked. Only specific sessions may interact with certain records. Session 1 wants record 5, only session 1 may do more than just look at record 1. A customer may only look at their orders
* Reduced paging to be efficient as possible
* Data model is based on the real world. If you have a object model it should be matching/feels more natural
* Works well for distributed architectures

Disadvantages

* Lower efficiency when data and relationships are simple
* Relational tables are simpler to understand for most people
* Access speeds may be slower. Relational systems work of performance, and getting every last bit of power out of the system
* More user tools, documentation, expertise for RDBMS
* Standards are more stable for RDBMS, less changes and the like
* Support for RDBMS is more certain and change is less likely

**XML Database**

Mine>These are DB’s that are storing XML directly, usually have a specific goal in mind: storing/transferring data. Because XML is a universal language, and because they can easily export as XML, and should be easily parse able. Really slow. About ease of communication/following universal standards

Teachers-XML DB’s allows for the data to be stored in the XML format. The data can then be queried, exported and serialized into the desired format. These db’s are usually associated with document oriented databases. Typically used for transport of information as the XML format is universally effective.

**NoSQL database**

NoSQL (Not Only SQL) is the current trend mostly being utilized by the web development world. NoSQL is used to describe the new datastores that are non-relational. There are many different ways of classifying NoSQL databases such as by data model or feature. The 4 basic categories are:

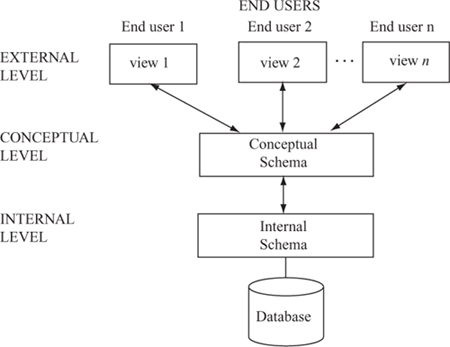
* Column – Basically an object that consists of 3 elements (unique name, value, timestamp) Example: Google BigTable – public version 2015, used on Google Cloud platforms
* Document –Documents encapsulate and encode data. Example CouchDB, uses JSON.
* Key-value – used for storing simple data and looking it up quickly. Basically a map and a dictionary that allows clients to put and request values per key. Ex: Amazon DynamoDB, Oracle NoSQL.
* Graph – A type on navigational database that employs nodes, properties and edges where the node is your entity. Ex:Amazon Neptune released 2018

## Diagram the architecture of a DBMS

When talking about DBMS architecture, we are referring to how the database is viewed by users. Abstract views of the data are provided to the users. These views hide how the data is stored so that the user does not need to worry about where data is located or how it’s stored.

The architecture is comprised of 3 different levels: external, conceptual and internal.

* External:   
  > End users have a custom screen, report, view etc, that is made for them or their task. Only shows specific info/only what they need.
  + Highest level of abstraction, also called the view level. Defines how each and end user understands the organization of the relevant data within the database. A single user can have a number of views at the external level. (Jim in HR has both the normal and HR views available to him, but don’t have any view about inventory counts). Allows for flexible security and access control.
* Conceptual:   
  > This level of abstraction is dealing with the logical structure of the entire db schema and is also known as the logical level. Describes what data is stored in the db, the relationships between the data and a complete view of the user’s requirements without any concern for the physical implementation. The “what” level.
* Internal (Physical Level):   
    
  > How the data is actually be stored on the physical drive/disk. DBA’s might set “these 2 tables in this specific drive”…etc. Might be performance options, tuning how the system will work, do not have control over the db software, just influence how it works. The “How” level
  + It is the lowest level of abstraction and deals with the physical representation of the database on the computer is also known as the physical level. It describes how the data is stored and organized on the storage medium. Concerned with the operational matters of the DBMS. It deals with the storage layout of the conceptual level, provides the supporting structures like indexes to enhance performance.



The “How”

The “What”

The lower you go the less abstraction there is

DBAs may have certain details

Developers/DBAs

End Users

Major components of DBMS architecture: (Primarily are in the database software)

* **DBMS External Interfaces** – Tool to talk to database. Means of communication with the db software, and data. Means of communication with the db to perform operations on a database and manage the DBMS. Includes APIs for programmers and GUI for DBAs.
* **Database Language Engines (or processors)** – Takes our commands, and checks validity, and interprets the command (what’s being asked, what has to be done, and in what order?). -Most operations upon databases are performed through expression in database languages. Language expressions are fed into the DBMS through interfaces and the language engine will compile or interpret the expressions to extract the intended database operations.
* **Query Optimizer** – Determine for queries “How” its going to do the command. Get info from language Engine and creates a plan to determine what is the most efficient way to do the command. – Performs query optimization: attempt to determine the most efficient way to execute the query by using the most efficient query plan.
* **Database Engine** – Manages everything else. - The core service for managing, storing, processing, and securing data.
* **Storage Engine** – Responsible for taking the actions/commands and figures out how to manipulate the data/disk/bits/ to change the data. – Handles the SQL operations for different table types, translates the operations to low level operations on the storage bits. Some references include this as part of the DB engine.
* **Transaction Engine** – A way to group commands(all happen or none of them do), you won’t end up with data that gets partially deleted or modified. - For correctness and reliability purposes most DBMS internal operations are performed encapsulated in transactions. The transaction engine tracks all the transactions and manages their execution according to transaction rules such as concurrency control.

## Discuss data warehouses and multidimensional data structures

### Data Warehouse

VIDEO- Data warehousing – An overview. Youtube andy wicks

Data warehouse (DW) – Purpose is to provide aggregate data which is in a suitable format for decision making. –Is a database used for reporting and data analysis to assist decision makers in making decisions. It is usually archived information that is taken from different enterprise application systems

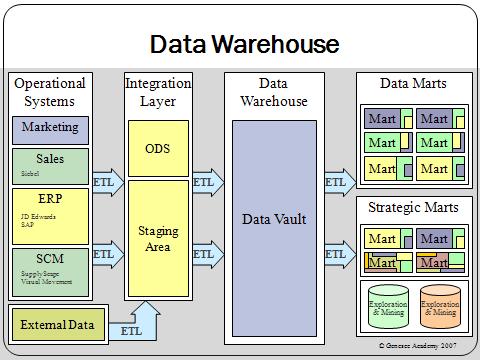
ETL-Extrantion, Transformation and Loading: Get data, Make it useful(uniform), Save it to the warehouse.

Data Marts(sub-sets of the DW)

-Don’t mess with my data -Keep it simple for the user -Small problems are easier to solve

Big Data(Second Normal form/2NF)- de normalize the data, good for query speed, but lots of repetition.

The data warehouse database tends to be:

1. Subject oriented- organized around subjects (customers, products, sales) instead of major functional areas (invoicing, stock control, product sales)
2. Integrated – integrating multiple data sources to be consistent.
3. Time variant - (usually not worth running the ETL while the business is operating, done once a day, once a week, month, etc) – data represents a series of snapshots and is only valid at some point in time or some interval. (example: the change in price of a product over time)
4. Non-volatile – data is not added in real time but instead refreshed on a regular basis. Data is added as a supplement and does NOT replace old data. [](http://upload.wikimedia.org/wikipedia/commons/4/46/Data_warehouse_overview.JPG)

Takes the data of the entire organization, makes it uniform(standardized), and then separates it again into little chunks based on user need

The **Extract, Transform and Load (ETL)** process:

* Extract data from various operational systems
* Transform the data to fit the operational needs within the integration layer
* Loads the data into the data warehouse

These pieces of information are then stored in the Marts (Data and Strategic). Marts are used to improve performance and ease of use within that specific area. They are the access layer for providing data to the users.

Data mart – A subset of the DW – Like a small data warehouse focused on a specific are of interest.

Strategic mart – A subset of the DW, DW focused on data that will be used for determining business direction and setting strategic goals.

The data is often grouped into hierarchical groups (trees) called dimensions and also into facts and aggregate facts. The combination of facts and dimensions is sometimes called a star schema.

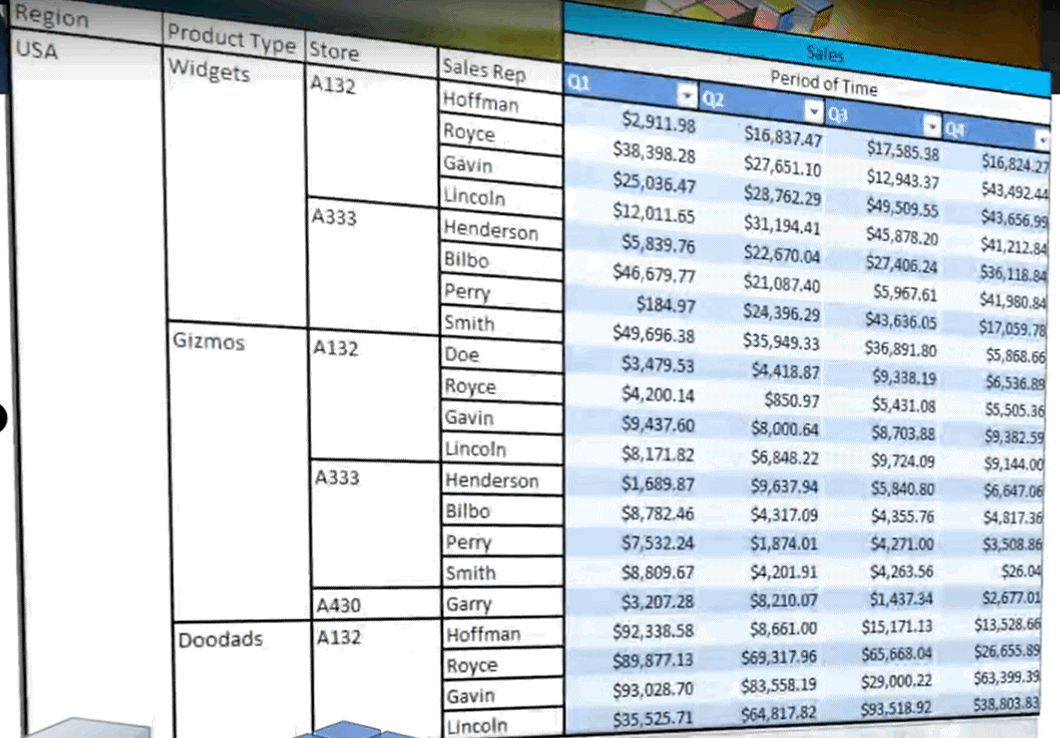
There are two main approaches to storing data in a data warehouse, the dimensional approach (Star Schema) and the normalized approach.

In a dimensional approach, transaction data are partitioned into "facts", which are generally numeric transaction data, and "dimensions", which are the reference information that gives context to the facts.

Example: A sales transaction can be broken up into

VIDEO:What is OLAP?

Equivalent of a crosstab table or pivot table



^Dimensions ^Facts

1. facts: (number we want to total, the “buckets” of a crosstab table, the values)
   1. the number of products ordered
   2. the price paid for the products
2. and dimensions: (column headings, etc)
   1. order date
   2. customer name
   3. product number
   4. order ship location
   5. salesperson

The main disadvantages of the dimensional approach are:

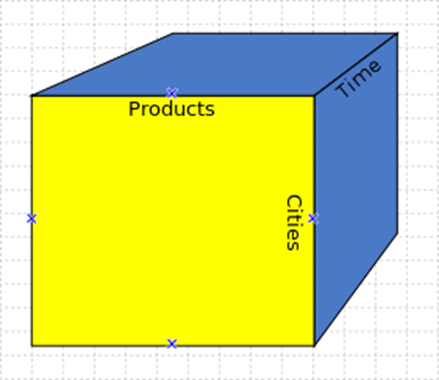
1. In order to maintain the integrity of facts and dimensions, loading the data warehouse for different operational systems is complicated

2. It is difficult to modify the warehouse structure if the organization changes the way in which it does business.

It should be noted that both normalized – and dimensional models can be represented in entity-relationship diagrams as both contain joined relational tables. The difference between the two models is the degree of normalization.

### Multidimensional Data Structures

In Relational databases, tables are only able to provide a 2 dimensional view of data. The multidimensional structure is similar to the relational model. The dimensions of the cube-like model have data relating to elements in each cell. This structure gives a spreadsheet-like view of data. This structure is easy to maintain because records are stored as fundamental attributes—in the same way they are viewed—and the structure is easy to understand. Its high performance has made it the most popular database structure when it comes to enabling online analytical processing (OLAP).



#### OnLine Analytical Processing

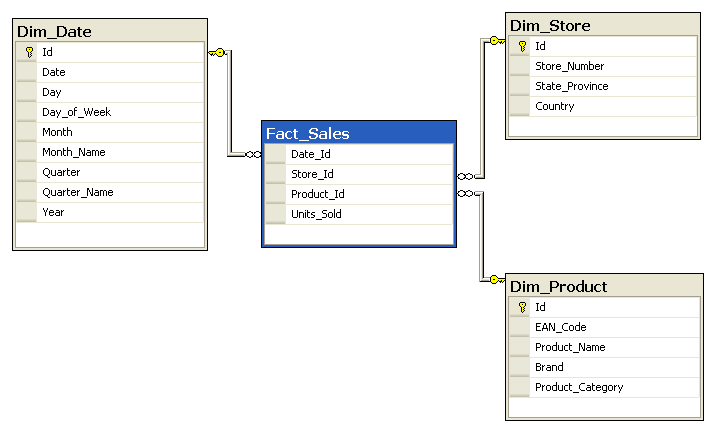
A cube can be thought of as a generalization of a two-dimensional spreadsheet. For example a company might wish to summarize financial data by product, by time-period, by city to compare actual and budget expenses. Product, time, city and scenario (actual and budget) are the data's dimensions.

Cube is a shortcut for multidimensional dataset, given that data can have an arbitrary number of dimensions. The term hypercube is sometimes used, especially for data with more than three dimensions.

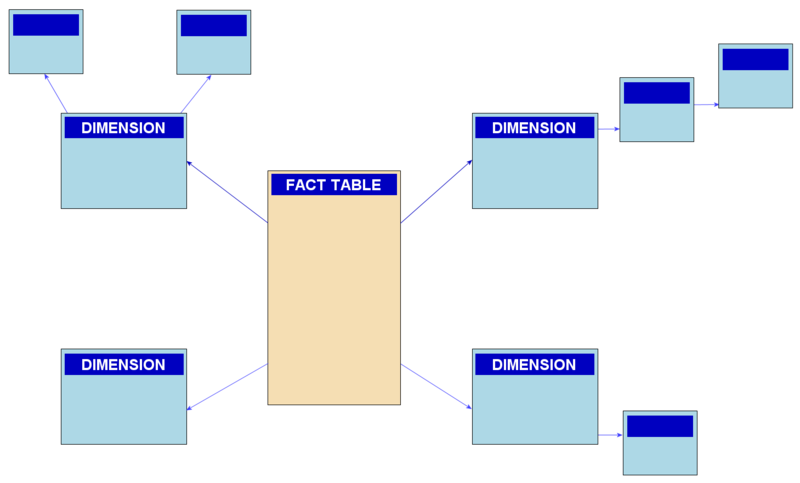
Each cell of the cube holds a number that represents some measure of the business, such as sales, profits, expenses, budget and forecast.

OLAP data is typically stored in a star schema or snowflake schema in a relational data warehouse or in a special-purpose data management system. Measures are derived from the records in the fact table and dimensions are derived from the dimension tables.

**Star Schema**



**Snowflake Schema**

[](http://upload.wikimedia.org/wikipedia/commons/b/b2/Snowflake-schema.png)

## Problems that data warehousing attempts to solve

1. Slow data/Reports
   1. Generation of reports should be quicker in a “Big Data” 2NF table from a DW.
   2. Some Data might be aggregated or pre-calculated before the report is run
2. Mission Level Information
   1. Business /Mission level users require a lot of aggregation of data(summation, averages, grouping) which are very processor intensive
   2. DW can simplify this data and aggregate it, or the schema can be made simpler resulting in faster queries
3. Scalable Access
   1. Simplify access to data so IT doesn’t have to customize every report
   2. Users have functionality to create their own reports and dashboards
4. Multiple version of the truth
   1. Reports from various departments get different results
   2. DW standardizes and integrates the data so apples can be compared with apples.

**Review for LO1:**

Work on review questions document.

Optional Resources and Extra Information:

Data mining – analyzing collected data (often stored in a data warehouse) for patterns.

Data mining video (Viewer discretion advised - this video uses language that some may find offensive. It also shows a dramatization of a pregnant woman’s water breaking). <https://www.youtube.com/watch?v=f2Kji24833Y>

Big data – collection of large and complex data sets to which data mining is usually applied.

Ted talk 22 minutes on big data: <https://www.youtube.com/watch?v=Zr02fMBfuRA>