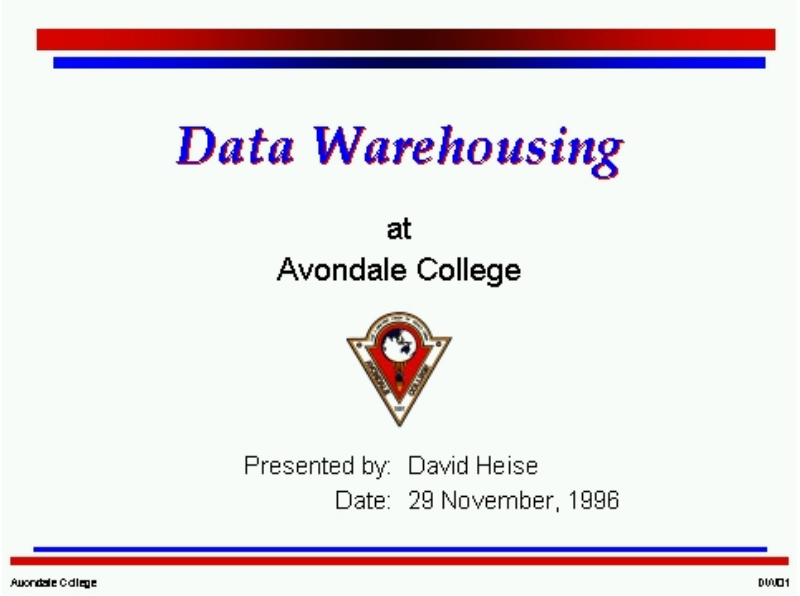
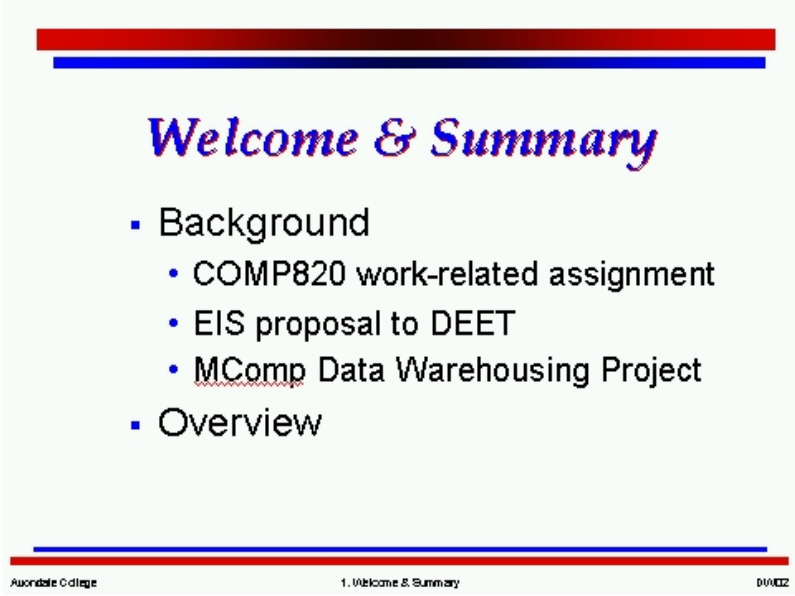


email: dheise@andrews.edu
URL: http://andrews.edu/~dheise/DW/ACDW.html

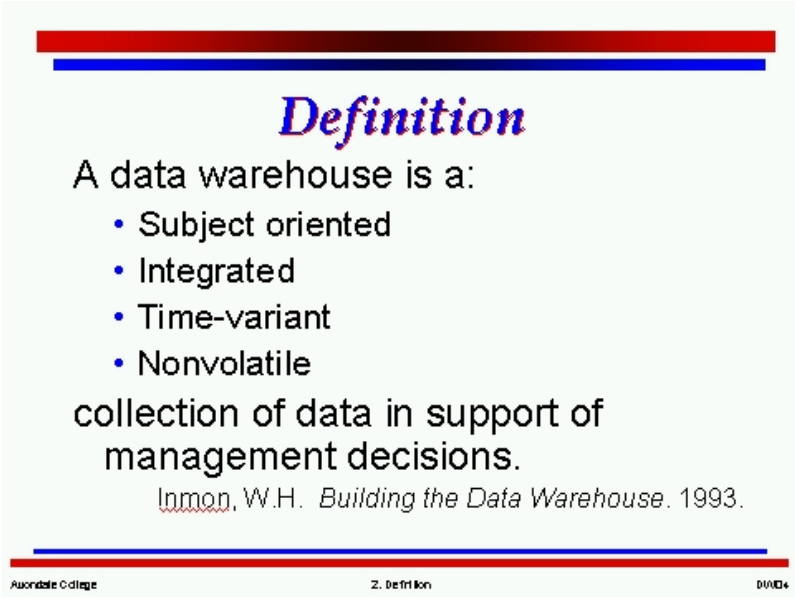


Background

* The initial ideas for this Data Warehousing project started with a work-related assignment for COMP820 (Information Systems Management and Analysis) in second semester of 1994.

* These ideas were further developed in a successful proposal requesting a grant from the National Priorities (Reserve) Fund for a "management improvement" project.

* This management improvement project received further impetus when it was accepted as the project topic for my Master of Computing.

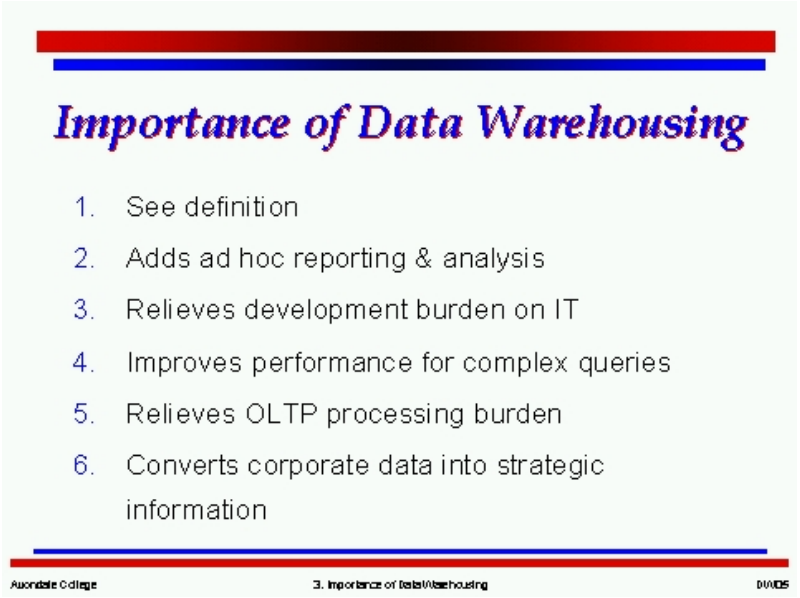


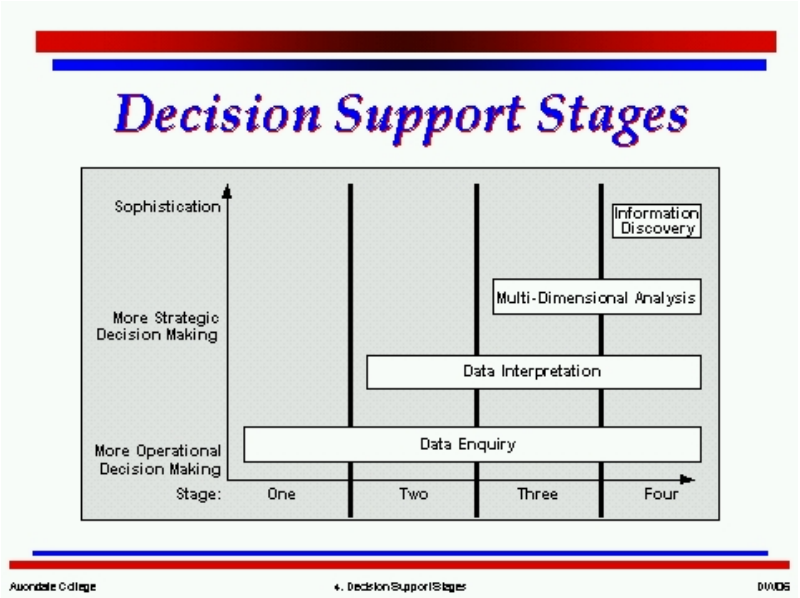
According to this definition, a data warehouse is different from an operational database in 4 important ways.

Data Warehouse	Operational Database
subject oriented	application oriented
integrated	multiple diverse sources
time-variant	real-time, current
nonvolatile	updateable

An operational database is designed primarily to support day to day operations.

A data warehouse is designed to support strategic decision making.





Data is arranged by **subject** rather than by application, and is more intuitive for users to navigate.

One of the initial motivations for this data warehousing project at Avondale College was to **integrate** multiple, diverse sources of data.

Data snapshots taken at times that are significant to the decision making process make it possible to analyze **trends over time**.

A data warehouse is designed to be accessible with end-user tools, and this allows ***ad hoc* reporting and analysis** by end-users.

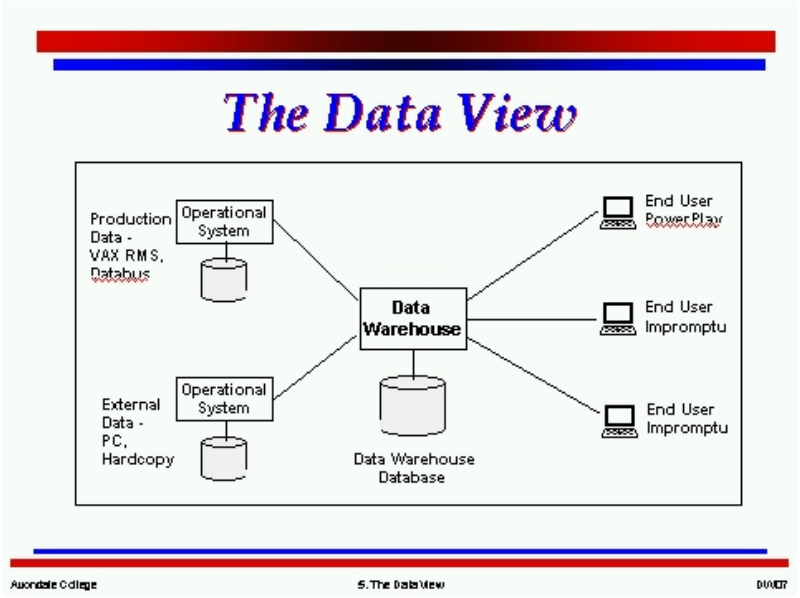
With a data warehouse and users trained in the use of appropriate desktop tools, **users can find answers to their own questions**.

Large analytical queries cause poor response time for the analysts, as well as severely impacting system **performance** for transaction processing functions.

There are **down sides** of course. Data warehouses can be extremely expensive to build and maintain. Also, the impact they can have on traditional views of data ownership and organizational structures can be quite disruptive.

Sophistication Level	Computer-based tool
data enquiry	traditional data processing applications
data interpretation	'point-and-click' reporting tools
multi-dimensional analysis	'slice and dice', 'drill-down' analytical tools
information discovery	intelligent agents

Notice that as an organization moves through the stages of decision support, and achieves higher levels of sophistication in their use of data, the lower levels are not made redundant. There will always be a place for standard operational reports. Knowledge workers will always benefit from having easy-to-use reporting tools, and so on.



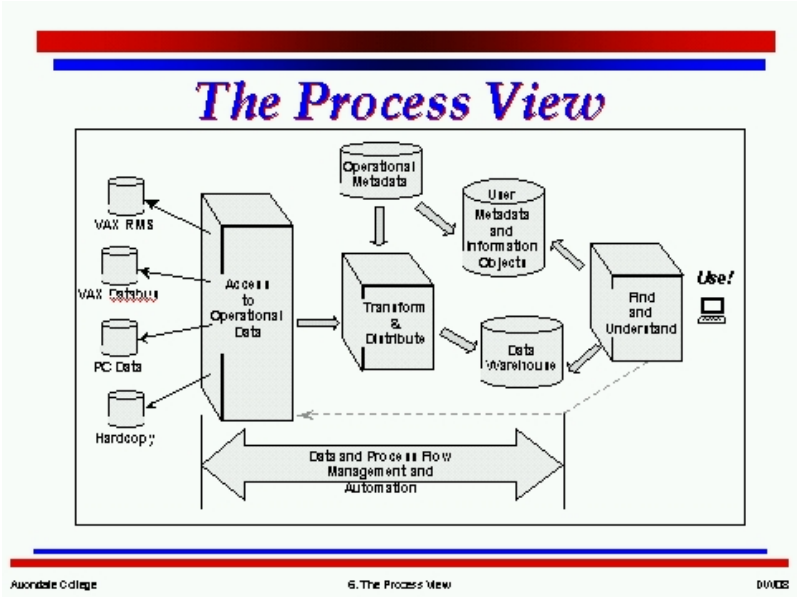
This is a simplified view of how data from the various sources is taken into the data warehouse, and is then accessible to end users for reporting and analysis

A data warehouse suitable for multi-dimensional analysis is denormalized in several ways:

1. coded data is replaced with values or meanings,
2. the complex joins are done ahead of time and stored as tables (this requires a good understanding of requirements, including anticipated queries, as well as the ability to adapt and evolve as the requirements develop)
3. significant aggregation and summarization occurs

As a result, the data warehouse we are building consists of two parts:

1. an integrated relational view of all operational data, with history - for Impromptu
2. summarized and pre-joined data to support multi-dimensional analysis - for PowerPlay

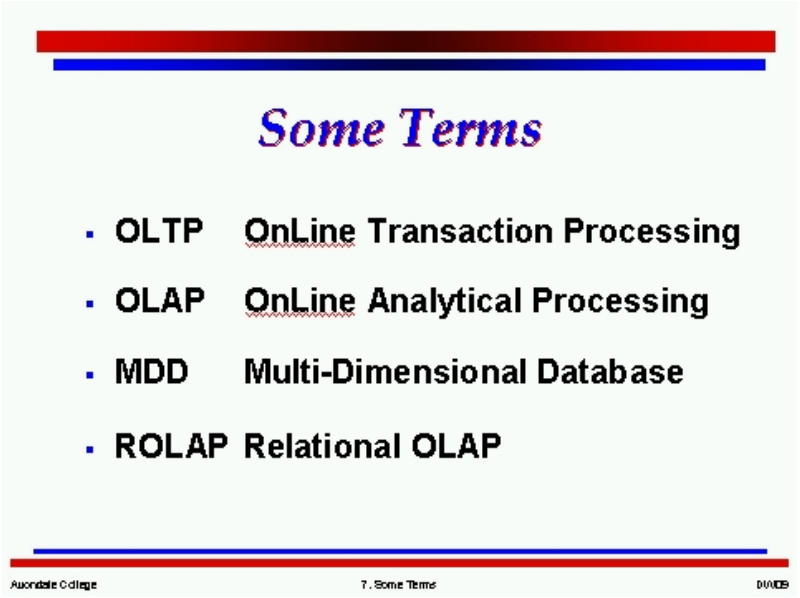


This diagram gives a more detailed view of the **processes** involved in managing and maintaining a data warehouse. The processes run from left to right, with a feedback loop from the users.

Flexibility and the ability to adapt to changing business needs are essential. Some vendors are beginning to talk about tools for automating maintenance. For this to happen, the management of the metadata needs to become more tightly integrated into the data warehousing process.

However, one of the fundamental assumptions of a data warehouse is that it is scalable. All of the advice I have seen suggests starting small with a pilot project, and then letting it grow.

The actual design process for developing a data warehouse runs from right to left in this diagram.

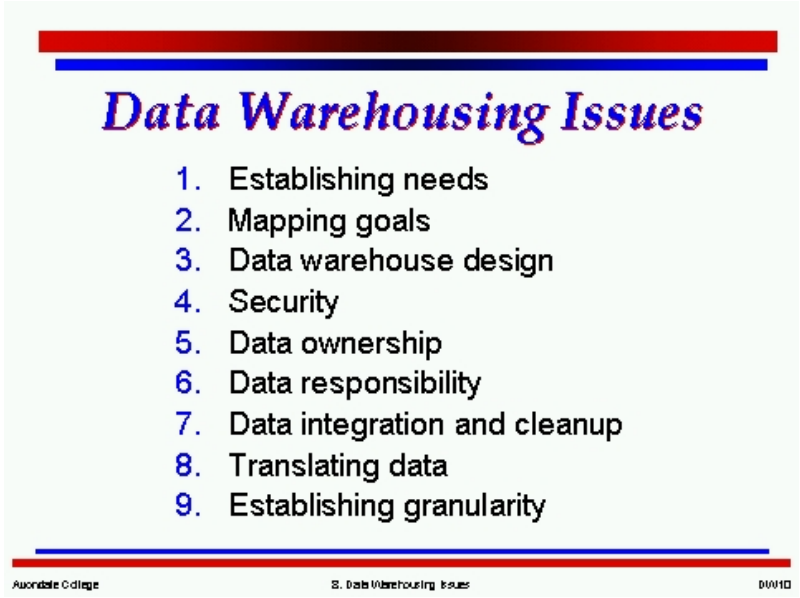


OLTP OnLine Transaction Processing
This the traditional data processing area, now dominated by relational databases, which have matured into products optimized for transaction throughput.

OLAP OnLine Analytical Processing
OLAP requires the ability to consolidate, pivot and rotate, and analyze data according to its multiple dimensions.

MDD Multi-Dimensional Database
An analysts' view of the enterprises' universe is typically multi-dimensional in nature. The multi-dimensional attributes of this data model - also known as a Hyper cube - are designed into the storage technology of the database and the desktop tool that sits on top of the database.

ROLAP Relational OLAP
ROLAP is the answer to MDD being proposed by vendors of traditional RDBMS. They argue that the multi-dimensionality of data is merely an attribute of the way the data is viewed and made available to user applications. The actual storage technology used to store the views can be treated separately.



1. Establishing needs
Refine the needs to a set of key areas to support with data or dimensions to use in data analysis.

2. Mapping goals to (measurable) performance indicators
We chose cohort retention rate and student performance.

3. Data warehouse design
Tables in a data warehouse are designated as "fact" tables and "dimension" tables. A fact table holds the information that is the subject of the analysis. A database that is designed for data warehousing will use what is known as a star schema.

4. Security
Designing a security strategy that all parties will agree to, then implementing and maintaining it, can be quite a task.

5. Data ownership
There was some initial conflict between the academic office and the alumni office over who owned the data and therefore who had the right to update it.

6. Data responsibility
The reverse problem also occurred in some instances. It became evident that we needed to identify a "data custodian" to be responsible for different

portions of the data. This was important for maintaining standards in the operational systems for data entry and update procedures.

7. Data integration and cleanup

Some fields were known by different names or had different data types in different systems, or were represented with different sets of coded values. Integrating these proved to be quite a challenge.

STUDENT_ID	PIC 9(10).
STUDENT_NUMBER	PIC 9(5).
NAME_NUMBER	PIC X(6).
(NAME_NUMBER replaces STUDENT_NUMBER plus CHECK_DIGIT)	

Semantic Differences in STUDENT ID

The most difficult cleanup problem was when duplicate records occurred for the same person, with different IDs. This happened even within the same system.

8. Translating data

We used Oracle Rdb for the data warehouse, and the PowerHouse 4GL QTP from Cognos for extracting, transforming and loading the data. Since most of the administrative software we use has been written in-house, we had already completed the integration of name and address under a single person ID. QTP has proved to be quick and easy to write and maintain, and is powerful and efficient in its operation.

```
DISPLAY 'Updating SPU_ANNUAL fields'

ACCESS *DW_DATA:RAWSPUA ALIAS RAW_SPU

DEFINE D_SPU NUM = (T_PASS / T_ATTEMPT) * 100
DEFINE D_SPU_GROUP NUM = 10 IF D_SPU EQ 100           &
                         ELSE  8 IF D_SPU GE  80      &
                         ELSE  6 IF D_SPU GE  60      &
                         ELSE  4 IF D_SPU GE  40      &
                         ELSE  2 IF D_SPU GE  20      &
                         ELSE  0

OUTPUT DW_PP_COHORT IN DW UPDATE ADD                  &
  VIA COURSE_CODE, STUDENT_ID, ANALYSIS_YEAR, SEMESTER&
  USING SUBJECT_CODE OF RAW_SPU,                      &
        NAME_NUMBER OF RAW_SPU,                       &
        YEAR_YYYY OF RAW_SPU,                         &
        SEMESTER OF RAW_SPU

  ITEM SPU_ANNUAL = D_SPU_GROUP
```

Sample QTP source code

9. Establishing granularity and a replication schedule

We are simply using batch jobs that resubmit themselves. This is working quite satisfactorily for our size of data warehouse (5-10MB), which is really only in the small data mart size category. In larger data warehouses, change detection becomes vital, so that only altered records are refreshed. In the case of QTP, the single statement

```
OUTPUT <table name> UPDATE ADD
```

automatically adds new records and updates only records that have changed.

```
$ submit daily.com -
        /restart -
        /queue=ACVSA_SLOW$BATCH -
        /after="TOMORROW+00:05:00" -
        /log=od_prog:daily.log
$ submit CI_ALL.COM /queue=acvsa$batch -
        /log=od_prog:CI_ALL.log
```

Sample JCL for scheduling: `daily.com`