Data warehouse

Debadatta Mohanty

Housekeeping Tips

- Please mute your phone during the presentation.
- If there is too much noise, participants will be put on auto-mute.
- We shall open the table for Q&A at the end of the session.
- Please feel free to post your questions over Chat as well.
- This session will be recorded, and an email will be sent with links to the recordings after the session.
- At the end of the course, TEX will request you to provide feedback on the training.

Session Objectives

- Overview of Data Warehousing
- Data Warehouse Architectures
- How to create a data warehouse
- How to design a data warehouse
- Understand the ETL process
- What is metadata
- How to administer a data warehouse

Operational System?

- Operational systems are just what their name implies; they are the systems that help us run the day-to-day enterprise operations.

- These are the backbone systems of any enterprise, such as order entry inventory etc.

- The classic examples are airline reservations, credit-card, authorizations and ATM withdrawals etc.,

Characteristics of Operational Systems

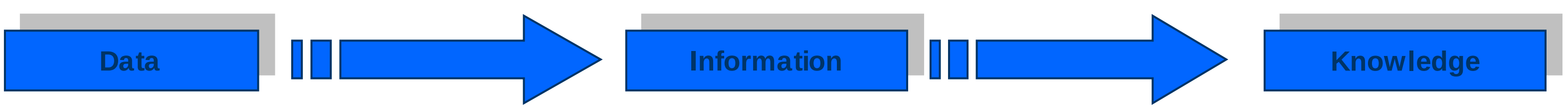
- Continuous availability
- Predefined access paths
- Transaction integrity
- Volume of transaction - High
- Data volume per query - Low
- Used by operational staff
- Supports day to day control operations
- Large number of users

Historical Look at Informational Processing

The goal of Informational Processing is to turn *data* into *information*!

Why?

Because business questions are answered using *information* and the *knowledge* of how to apply that information to a given problem.

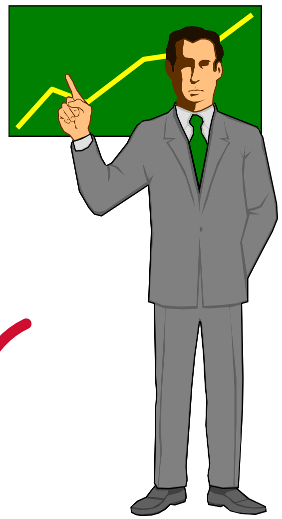


Need for a Separate informational system

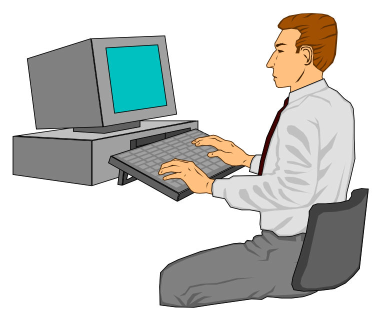
- *Data :* Informational data is distinctly different from operational data in its *structure and content.*

- *Processing :* Informational *processing* is distinctly different from operational processing in its *characteristics and use of data*

The Information Center



- **Management requires business information**
- **A request for a report is made to the Information Center**
- **Information Center works on developing the report**
- **Requirements for the report must be clarified**



HCL

The Information Center

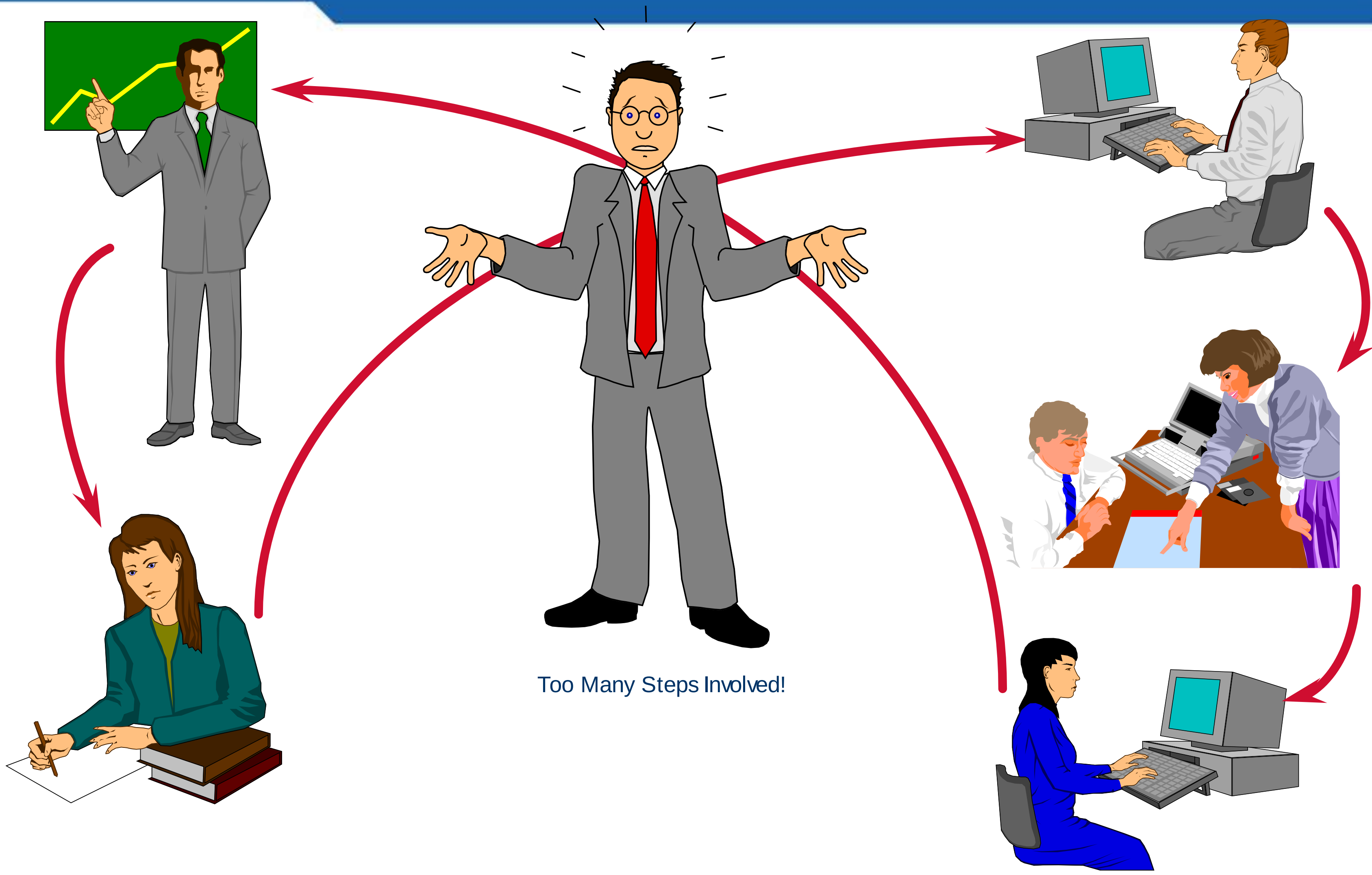
- **Report provided to analyst**
- **Analyst manipulates data for decision making**

- **Management receives information, but...**

  What took so long? and

  How do I know it's right?

The Information Center



Too Many Steps Involved!

Tactical Information

Supports day to day control operations

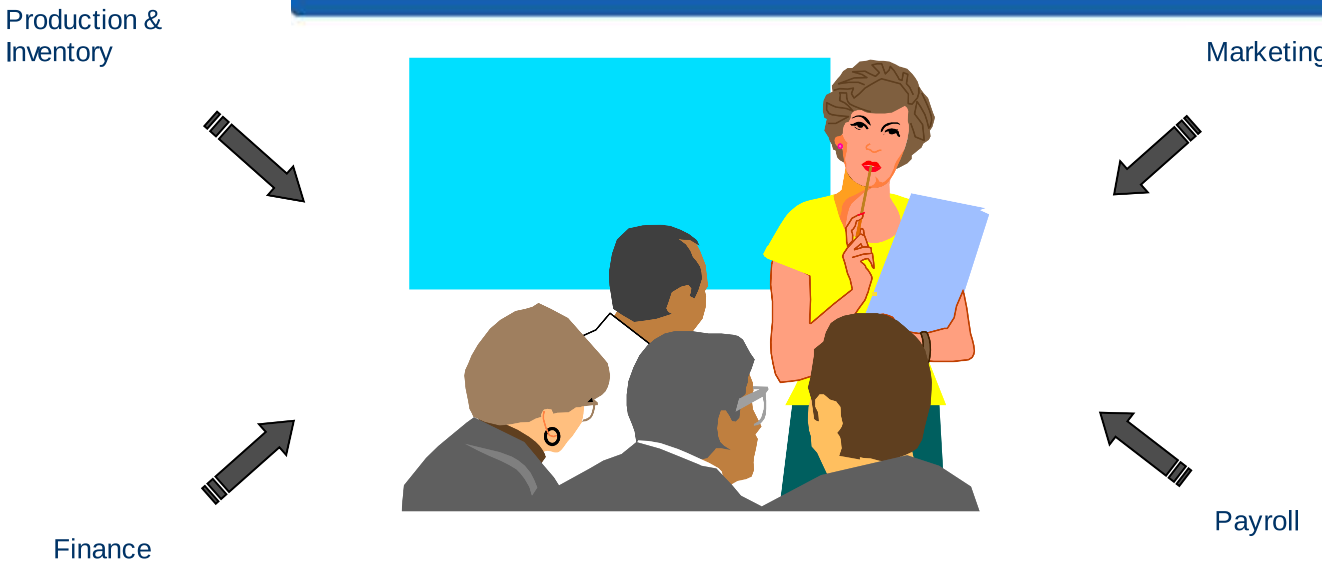
Transaction Processing

High Performance Operational Systems

Fast Response Time

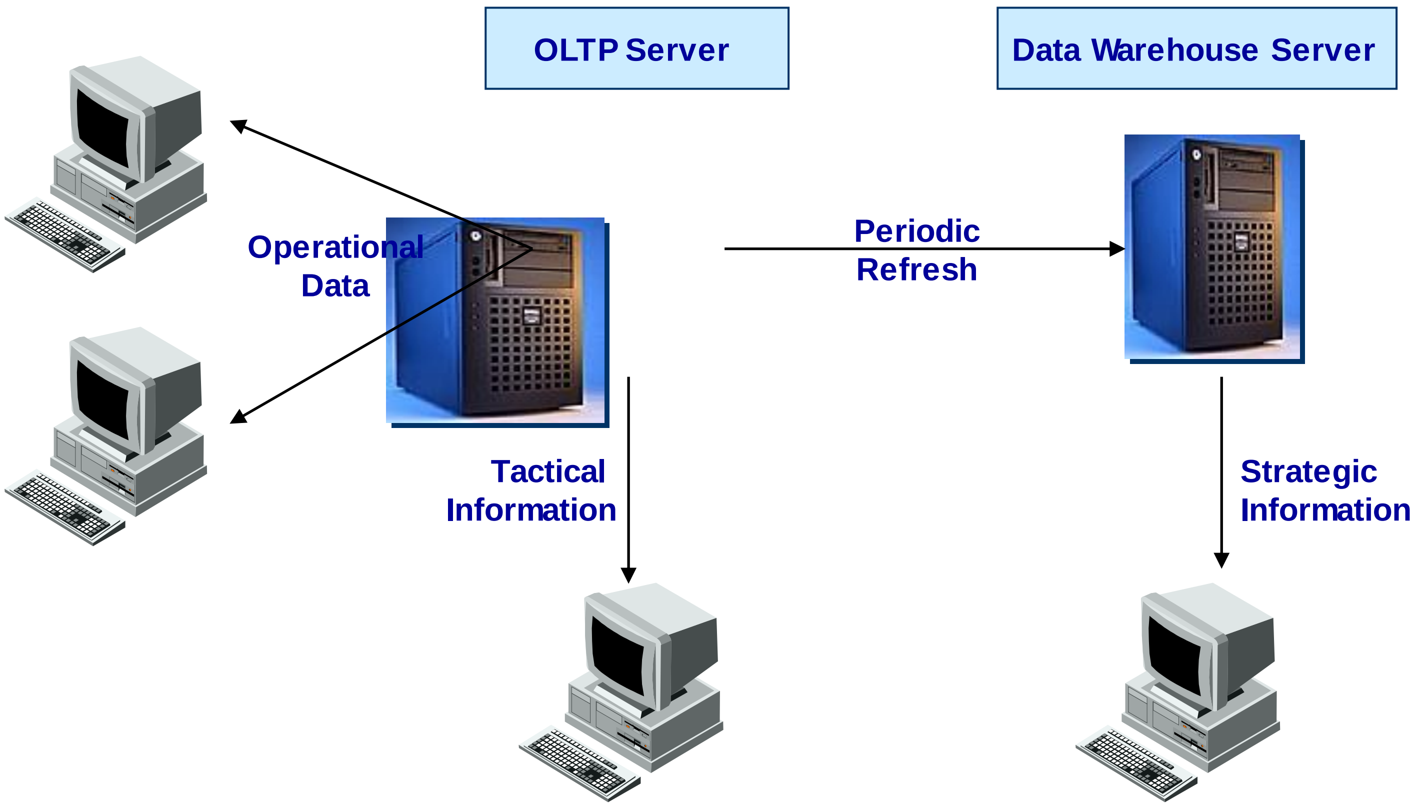
Initiates immediate action

Strategic Information



- Understand Business Issues
- Analyze Trends and Relationships
- Analyze Problems
- Discover Business Opportunities
- Plan for the Future

Need for Tactical and Strategic information



- Operational data helps the organization meet operational and tactical requirements for *<u>data</u>*.
- While the Data Warehouse data helps the organization meet *<u>strategic</u>* requirements for *<u>information</u>*

Operational Vs Analytical systems

Operational

- Primarily primitive
- Current; accurate as of now
- Constantly updated
- Minimal redundancy
- Highly detailed data
- Referential integrity
- Normalized design
- Supports day-to-day business functions

Analytical

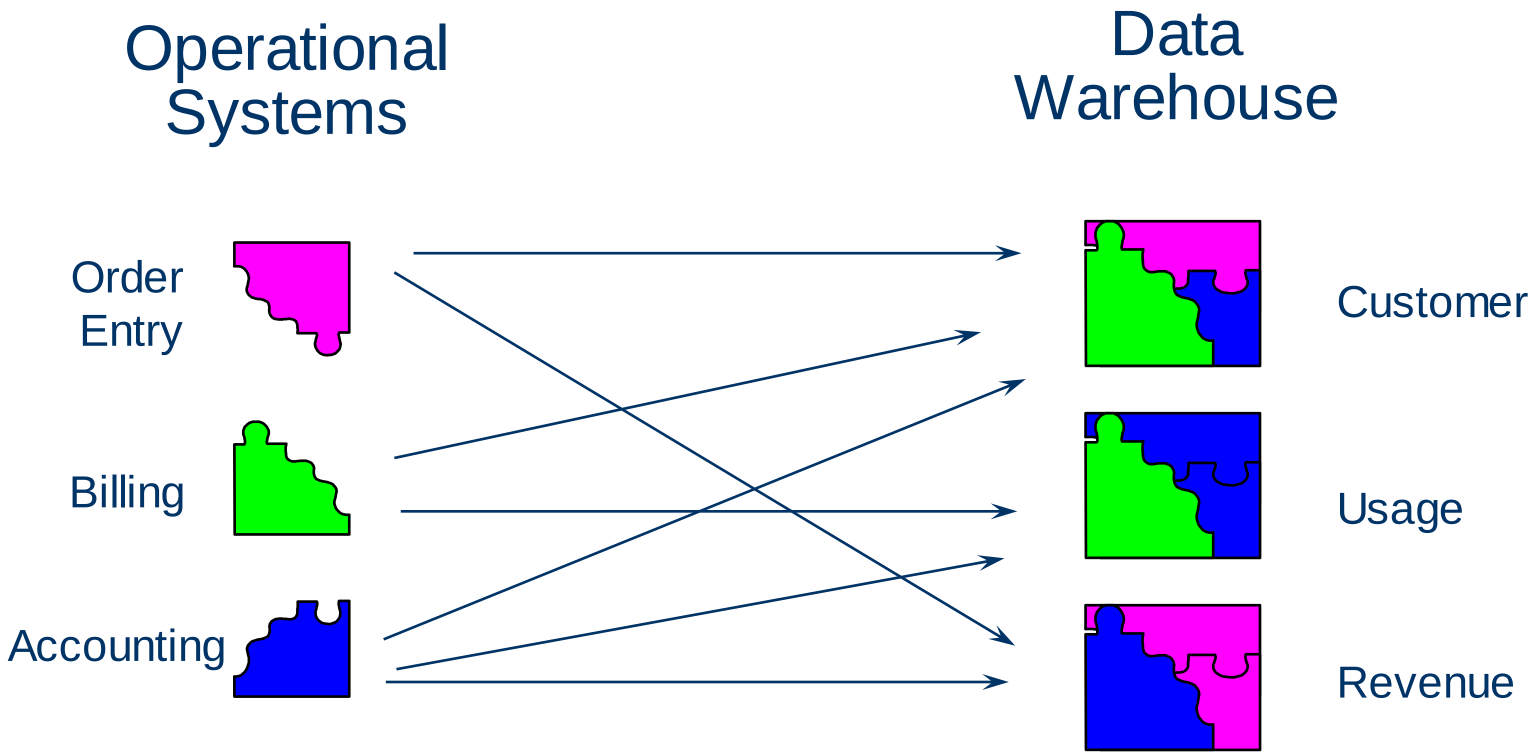
- Primarily derived
- Historical; accuracy maintained over time
- Less frequently updated
- Managed redundancy
- Summarized data
- Historical integrity
- De-normalized design
- Supports long-term informational requirements

Data Warehouse Definition

The Data Warehouse is

- Subject Oriented
- Integrated
- Time variant
- Non-volatile collection of data in support of management decision processes

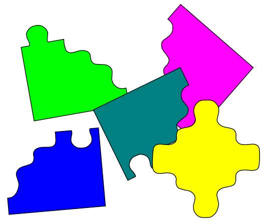
Data Warehouse- Differences from Operational Systems



- Operational data is organized by *specific processes* or tasks and is maintained by separate systems
- Warehoused data is organized by *subject area* and is populated from many operational systems

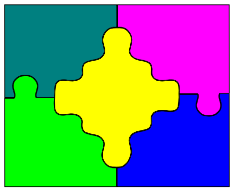
Data Warehouse- Differences from Operational Systems

Operational Systems



Application Specific

Data Warehouse



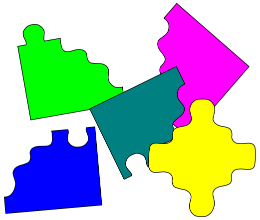
Integrated

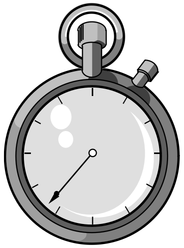
- Applications and their databases were designed and built separately
- Evolved over long periods of time

- Integrated from the start
- Designed (or "*Architected*") at one time, implemented *iteratively* over short periods of time

Data Warehouse- Differences from Operational Systems

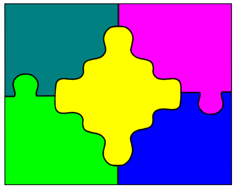
Operational Systems

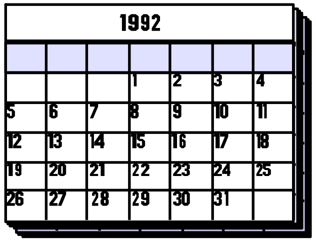




- Primarily concerned with *current* data

Data Warehouse

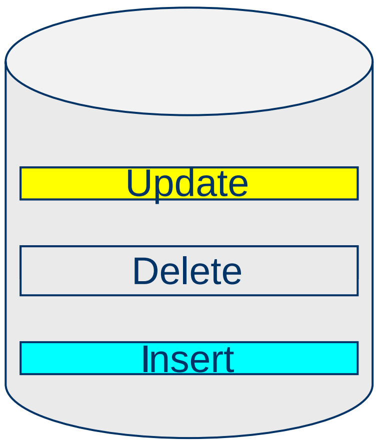




- Generally concerned with *historical* data

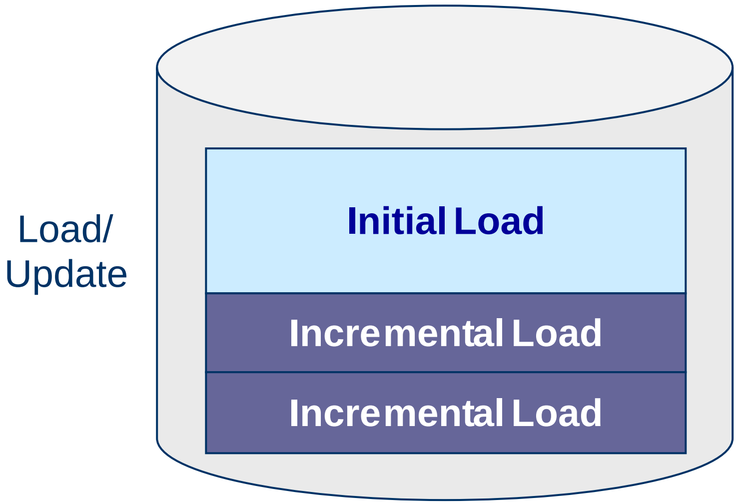
Data Warehouse- Differences from Operational Systems

Operational systems Database



Constant Change

Data warehouse



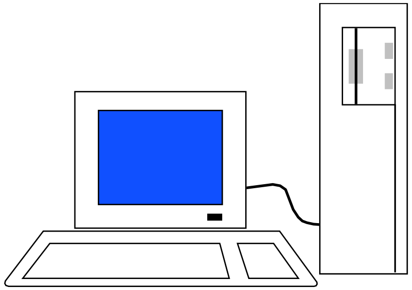
Consistent Points in Time

- Updated constantly
- Data changes according to *need*, not a fixed schedule

- Added to regularly, but loaded data is rarely *directly* changed
- Does <u>NOT</u> mean the Data warehouse is never updated or never changes!!

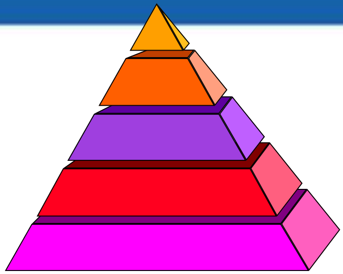
Data in a Data Warehouse

What about the data in the Datawarehouse?



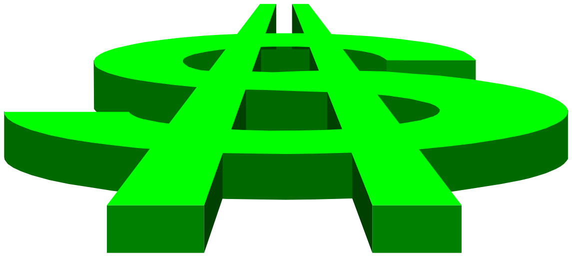
- Separate DSS data base
- Storage of data only, no data is created
- Integrated and Scrubbed data
- Historical data
- Read only (no recasting of history)
- Various levels of summarization
- Meta data
- Subject oriented
- Easily accessible

Data Warehousing Features



- Strategic enterprise level decision support
- Multi-dimensional view on the enterprise data
- Caters to the entire spectrum of management
- Descriptive, standard business terms
- High degree of scalability
- High analytical capability
- Historical data only

Datawarehouse - Business Benefits



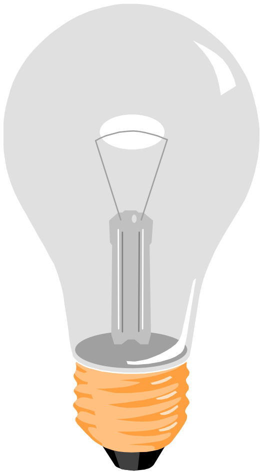
Benefits To Business

- Understand business trends
- Better forecasting decisions
- Better products to market in timely manner
- Analyze daily sales information and make quick decisions
- Solution for maintaining your company's competitive edge

Data Warehouse- Application Areas

Following are some Business Applications of a data warehouse:

- Risk management
- Financial analysis
- Marketing programs
- Profit trends
- Procurement analysis
- Inventory analysis
- Statistical analysis
- Claims analysis
- Manufacturing optimization



What is a Data mart?

- Data mart is a decentralized subset of data found either in a data warehouse or as a standalone subset designed to support the unique business unit requirements of a specific decision-support system.

- Data marts have specific business-related purposes such as measuring the impact of marketing promotions, or measuring and forecasting sales performance etc,.

Data marts - Main Features

Main Features:

- Low cost
- Controlled locally rather than centrally, conferring power on the user group.
- Contain less information than the warehouse
- Rapid response
- Easily understood and navigated than an enterprise data warehouse.
- Within the range of divisional or departmental budgets

Advantages of Datamart over Datawarehouse

Datamart Advantages :

- Typically single subject area and fewer dimensions
- Limited feeds
- Very quick time to market (30-120 days to pilot)
- Quick impact on bottom line problems
- Focused user needs
- Limited scope
- Optimum model for DW construction
- Demonstrates ROI
- Allows prototyping

Disadvantages of Data Mart

Datamart disadvantages :

- Does not provide integrated view of business information.
- Uncontrolled proliferation of data marts results in redundancy
- More number of data marts complex to maintain
- Scalability issues for large number of users and increased data volume

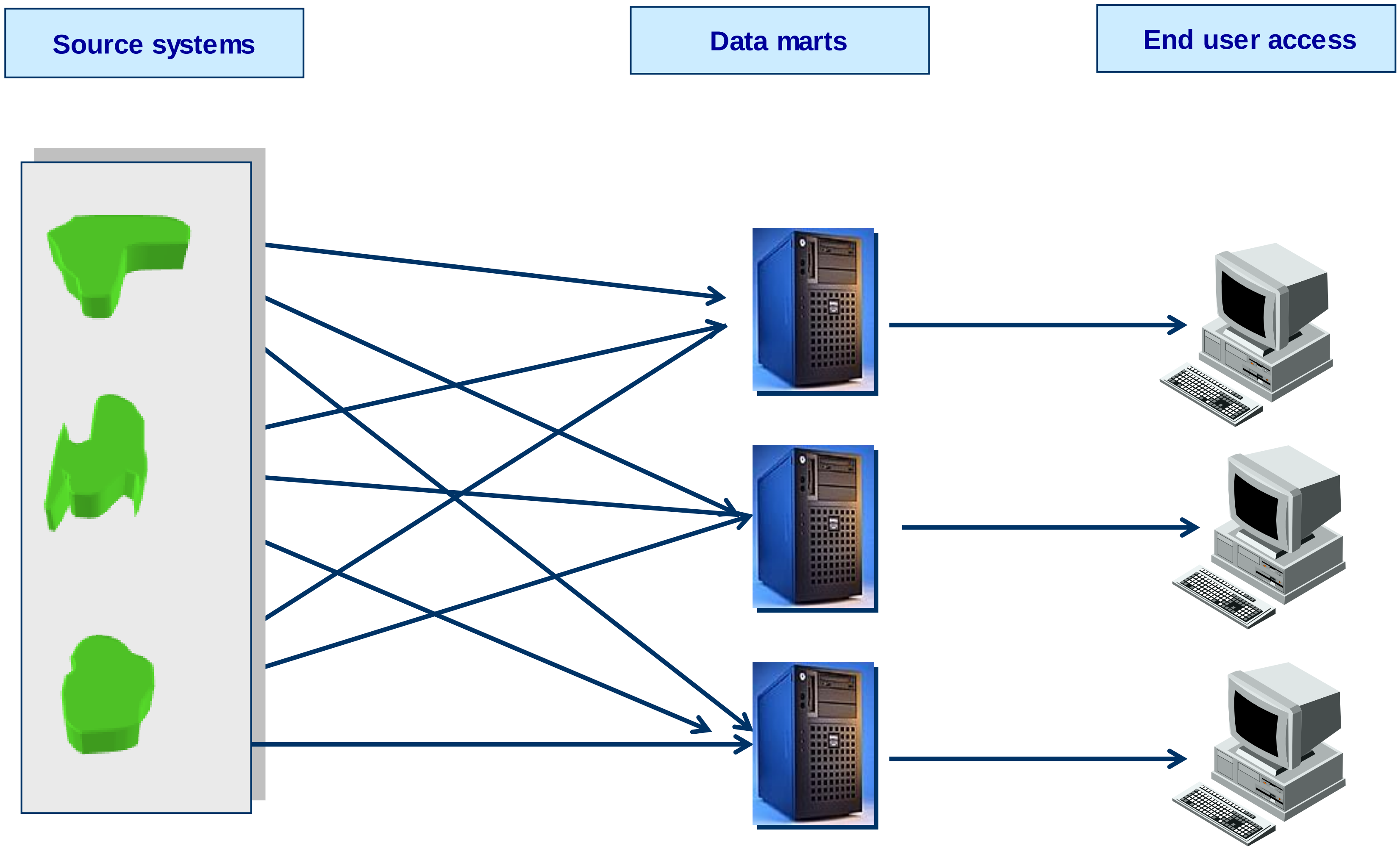
Different Approaches for Implementing Data marts

Non-architected Data marts

Question: When is a Data Warehouse not a Data Warehouse?

Answer: When it's an unarchitected collection of data marts

Non-architected Data marts



Significant and expensive duplication of effort and data.

Upsides and Downsides of Non-architected Data marts

The upsides of Non-architected Data marts are:

1. Speed

2. Low cost

The downsides of Non-architected Data marts are:

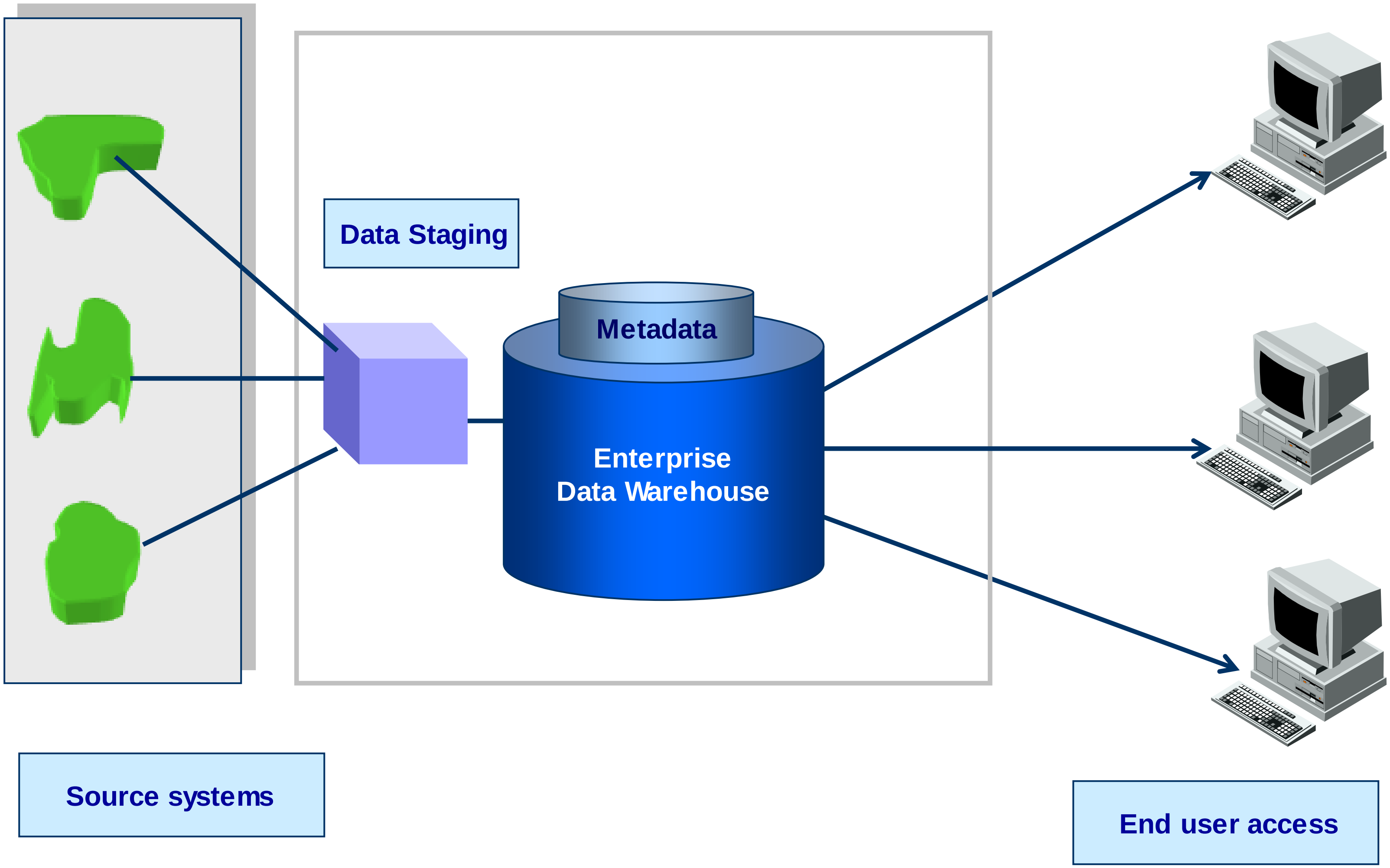
1. Multiple extraction processes

2. Multiple business rules

3. Multiple semantics

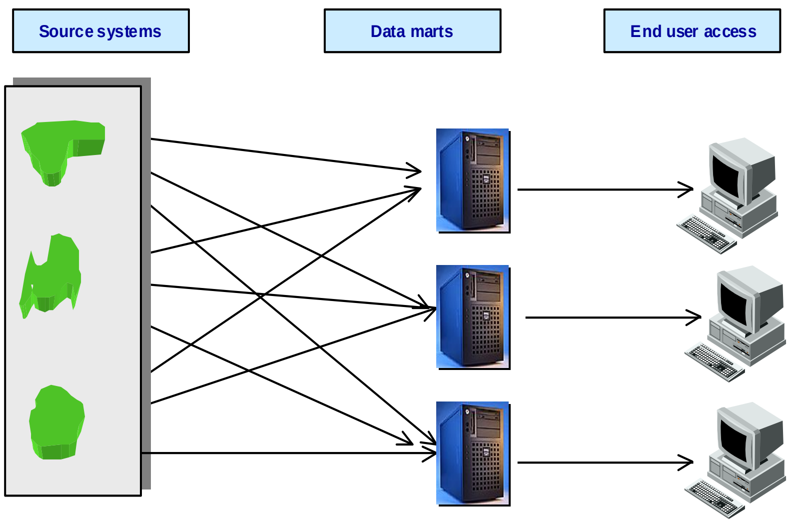
4. Extremely challenging to integrate

Architected Data Warehouse



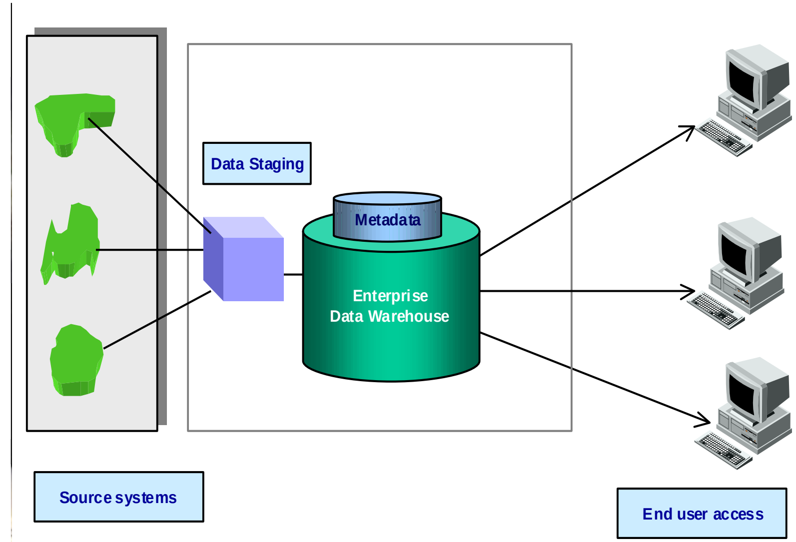
Unarchitected Data marts Vs Data warehouse

Unarchitected Data Marts



- Easy to do, Not architected
- Are the extracts, transformations, integration's & loads consistent?
- Is the redundancy managed?
- What is the impact on the sources?

Data Warehouse



- Architected
- Data and results consistent
- Redundancy is managed
- Detailed history available for drill-down
- Metadata is consistent!

ODS Definition

The ODS is defined to be a structure that is:

- Integrated
- Subject oriented
- Volatile, where update can be done
- Current valued, containing data that is a day or perhaps a month old
- Contains detailed data only.

Why We Need Operational Data Store?

Need

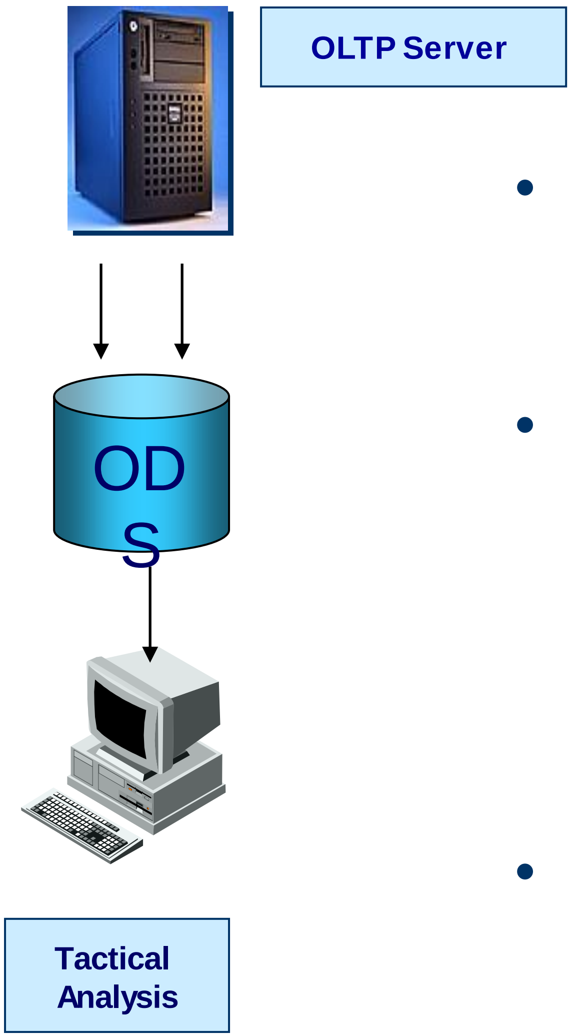
- To obtain a “system of record” that contains the **best** data that exists in a legacy environment as a source of information

Best here implies data to be

- Complete
- Up to date
- Accurate

- In conformance with the organization’s information model

Operational Data Store - Insulated from OLTP



- ODS data resolves data integration issues
- Data physically separated from production environment to insulate it from the processing demands of reporting and analysis
- Access to current data facilitated.

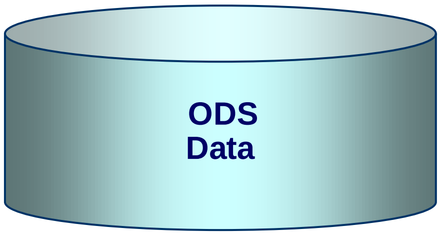
Operational Data Store - Data

- Detailed data
  - Records of Business Events
  (e.g. Orders capture)
- Data from heterogeneous sources
- Does not store summary data
- Contains current data

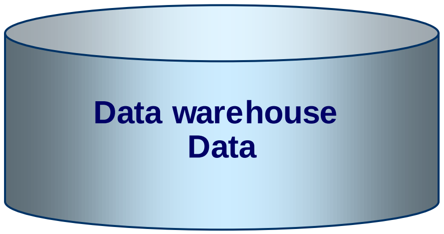
ODS- Benefits

- Integrates the data
- Synchronizes the structural differences in data
- High transaction performance
- Serves the operational and DSS environment
- Transaction level reporting on current data

Operational Data Store- Update schedule



- Update schedule - Daily or less time frequency
- Detail of Data is mostly between 30 and 90 days
- Addresses operational needs



- Weekly or greater time frequency
- Potentially infinite history
- Address strategic needs

ODS Vs Data warehouse Characteristics

Parameters	ODS	Data warehouse
Integrated and subject oriented	√	√
Updated By Transactions	√	
Stores Summarized data		√
Used for Strategic decisions		√
Used at managerial level		√
Used for tactical decisions	√	
Contains current and detailed data	√	
Lengthy historical perspective		√

OLAP

What is OLAP

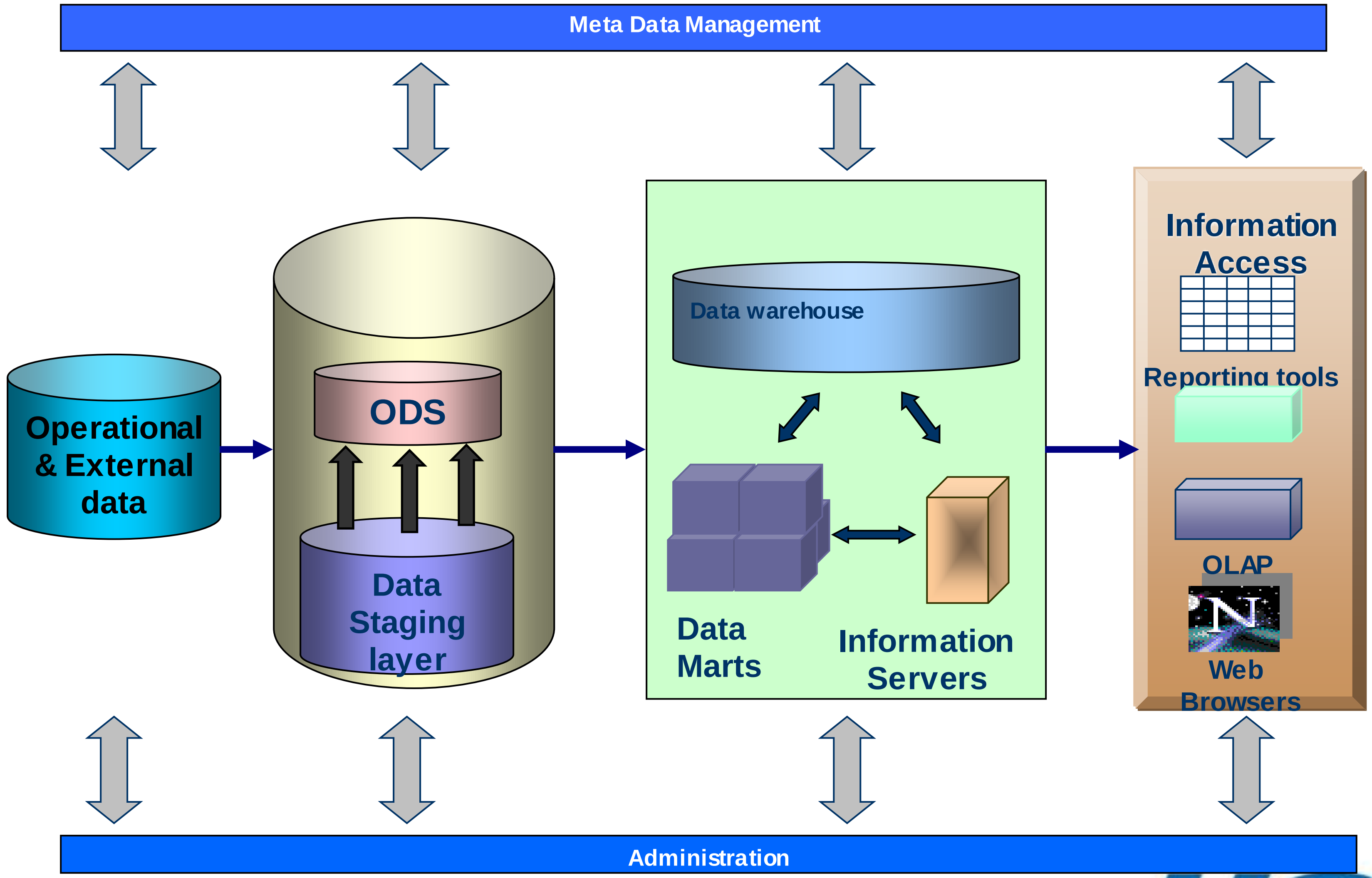
- OLAP tools are used for analyzing data
- It helps users to get an insight into the organizations data
- It helps users to carry out multi dimensional analysis on the available data
- Using OLAP techniques users will be able to view the data from different perspectives
- Helps in decision making and business planning
- Converting OLTP data into information
- Solution for maintaining your company's competitive edge

OLAP Terminology

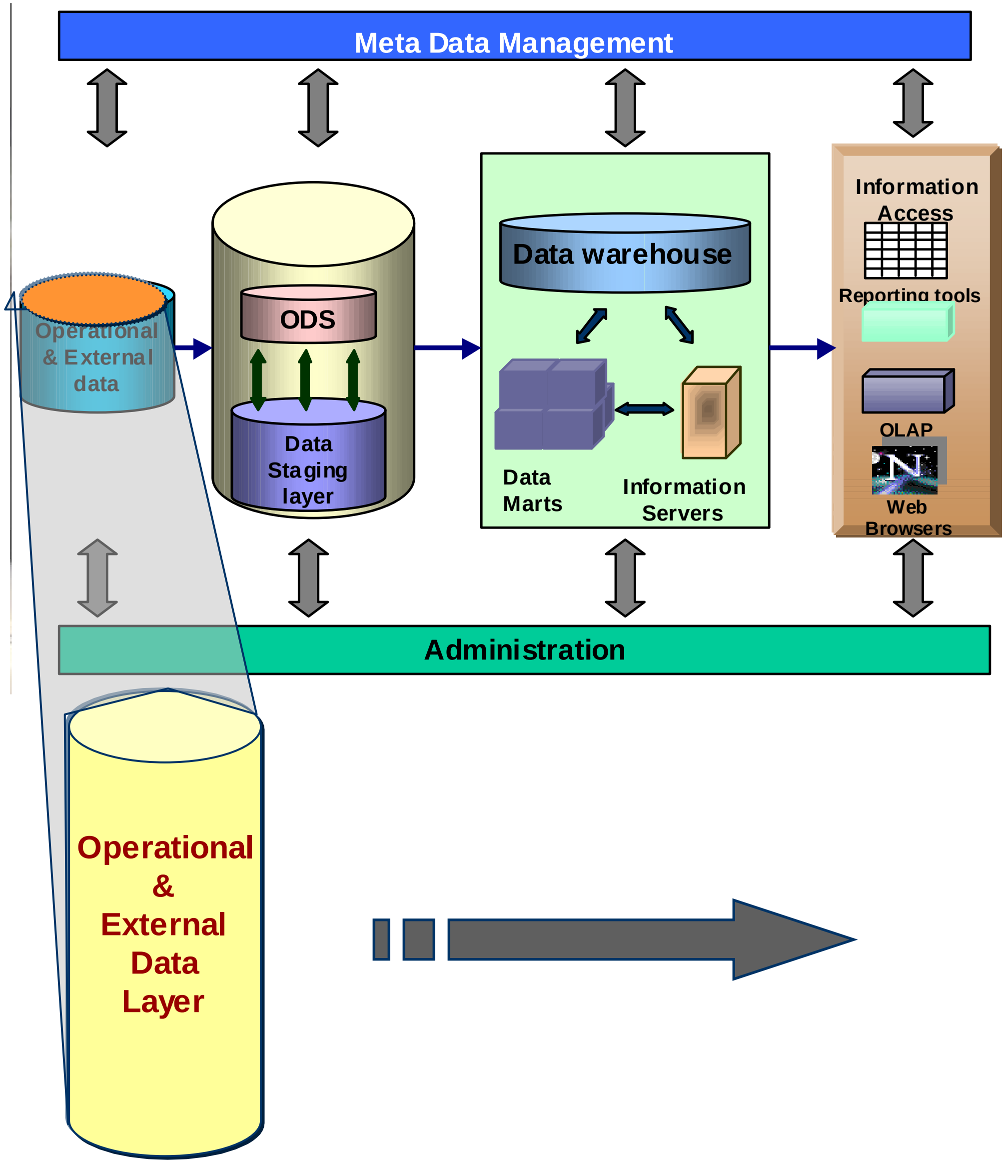
- Drill Down and Drill Up
- Slice and Dice
- Multi dimensional analysis
- What IF analysis

Data Warehouse Architecture

Basic Data Warehouse Architecture



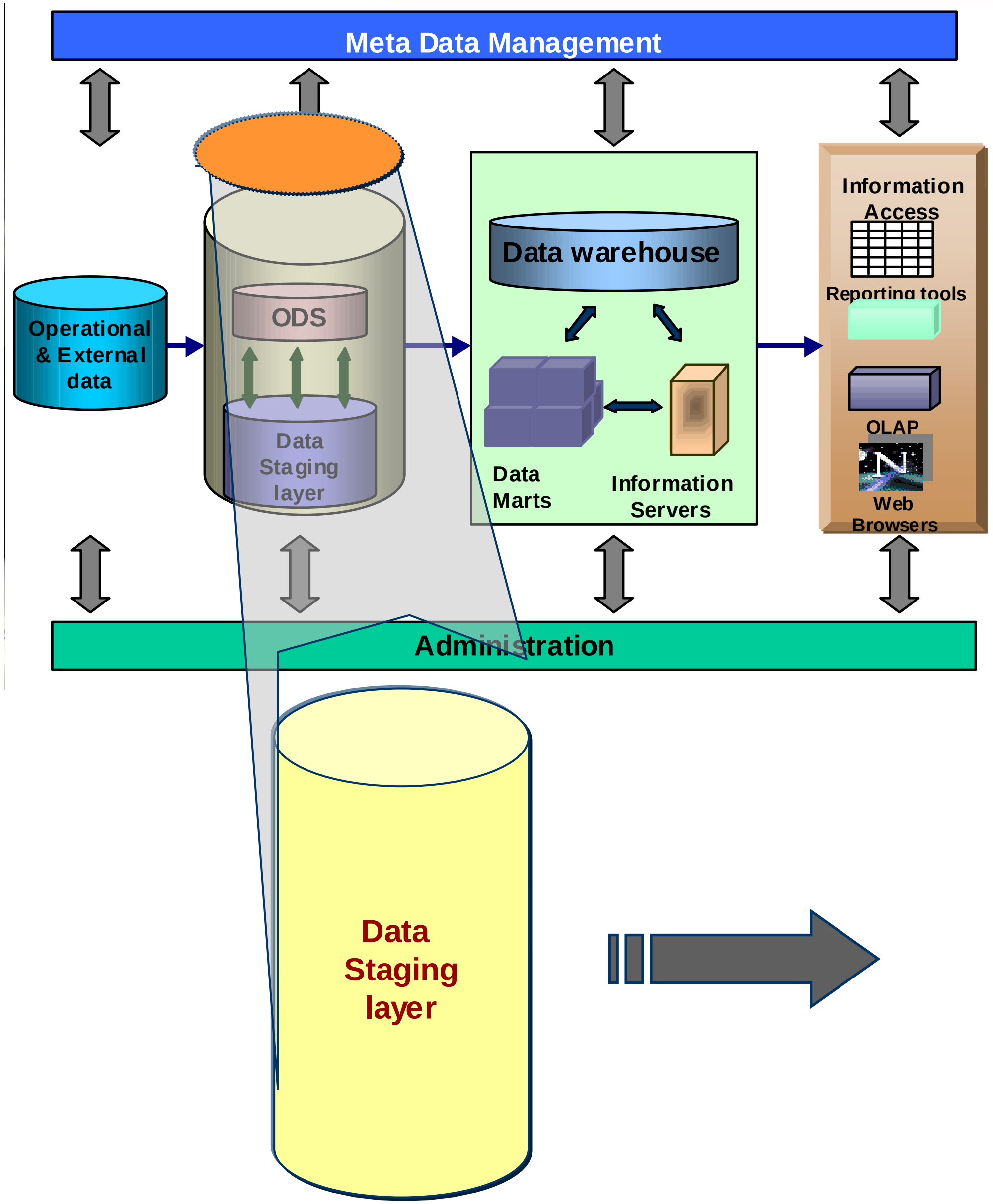
Operational & External Data layer



- The database-of-record
- Consists of system specific reference data and event data
- Source of data for the data warehouse.
- Contains detailed data
- Continually changes due to updates
- Stores data up to the last transaction.

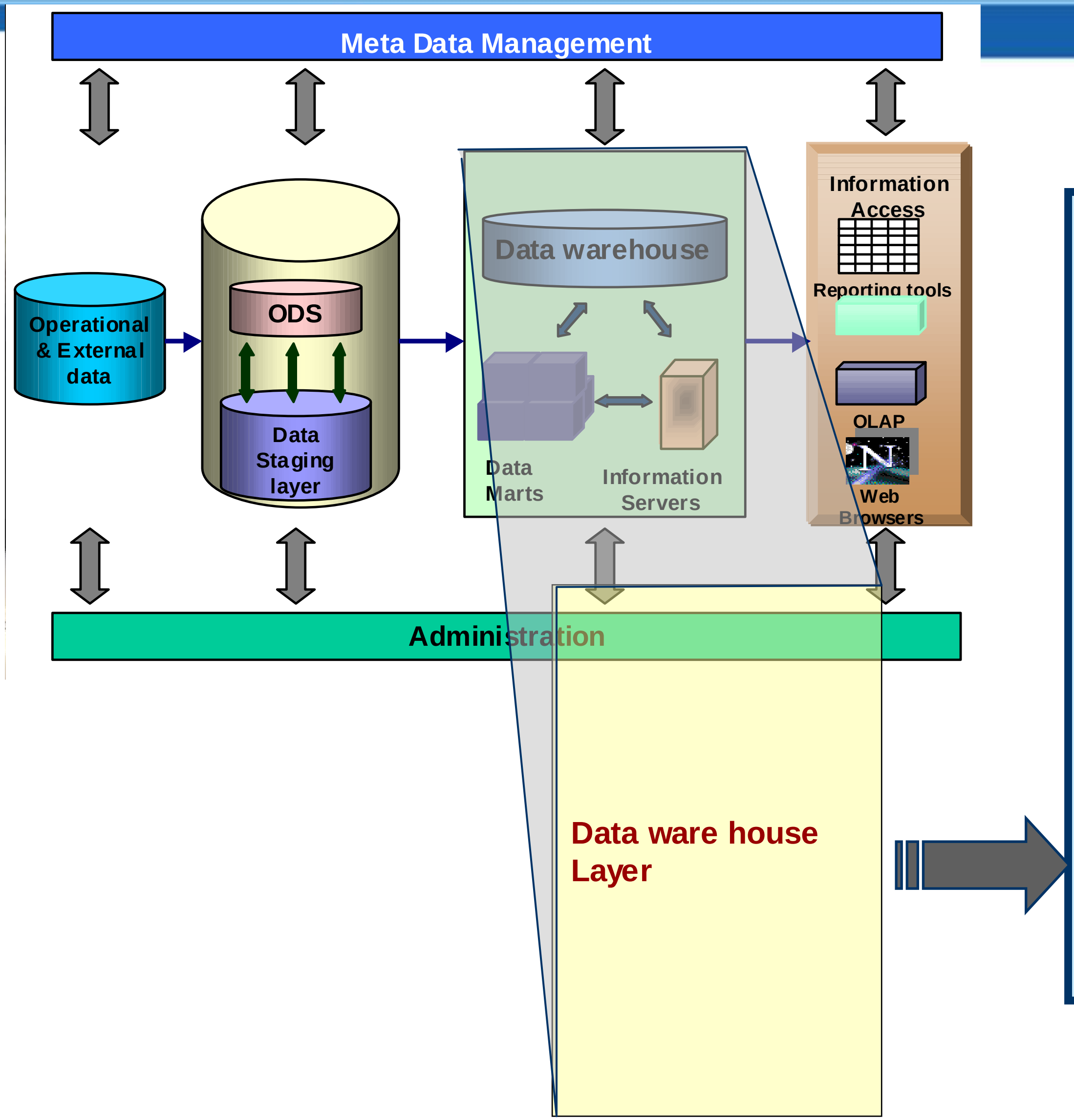
HCL

Data Staging layer



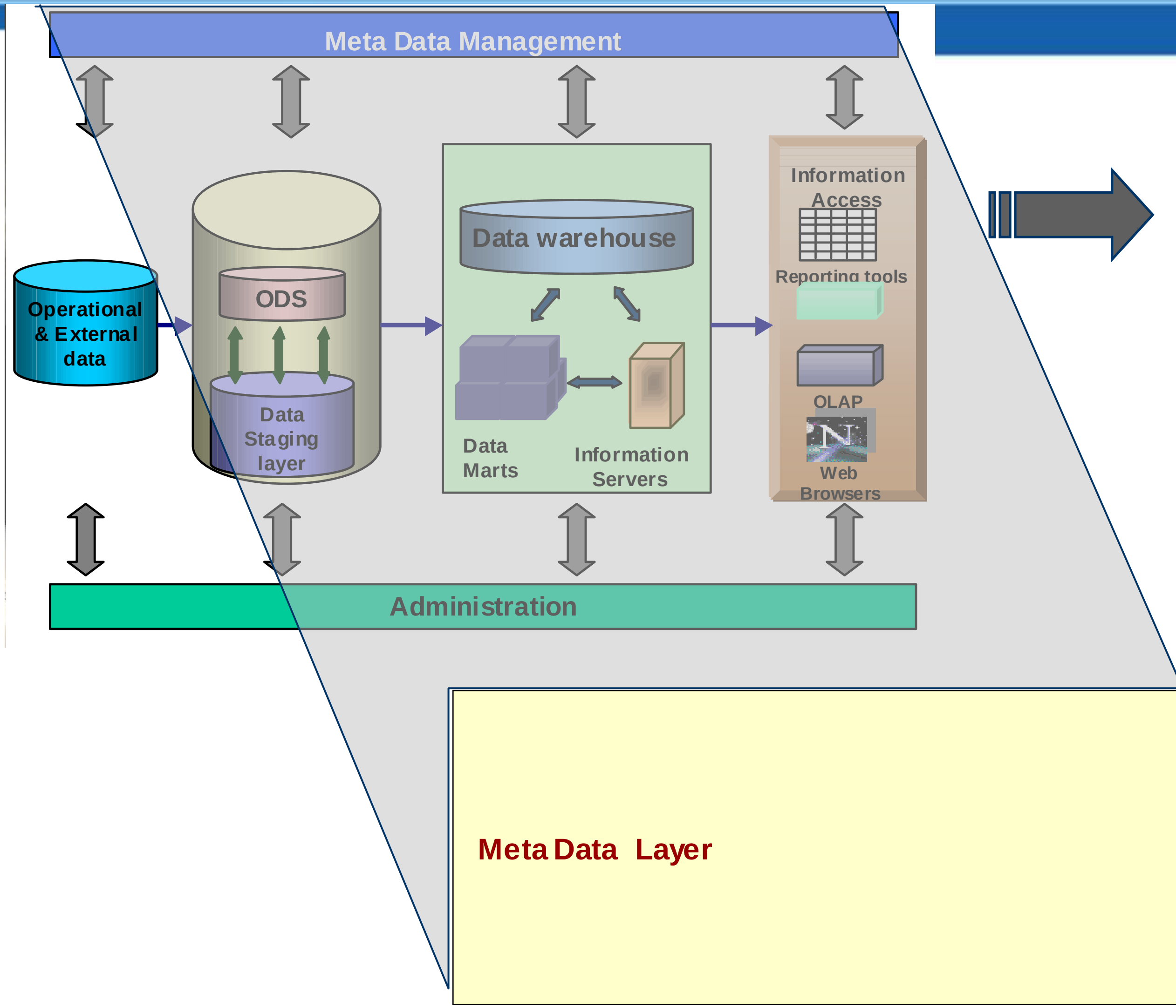
- Extracts data from operational and external databases.
- Transforms the data and loads into the data warehouse.
- This includes decoding production data and merging of records from multiple DBMS formats.

Data Warehouse layer



- Stores data used for informational analysis
- Present summarized data to the end-user for analysis
- The nature of the operational data, the end-user requirements and the business objectives of the enterprise determine the structure

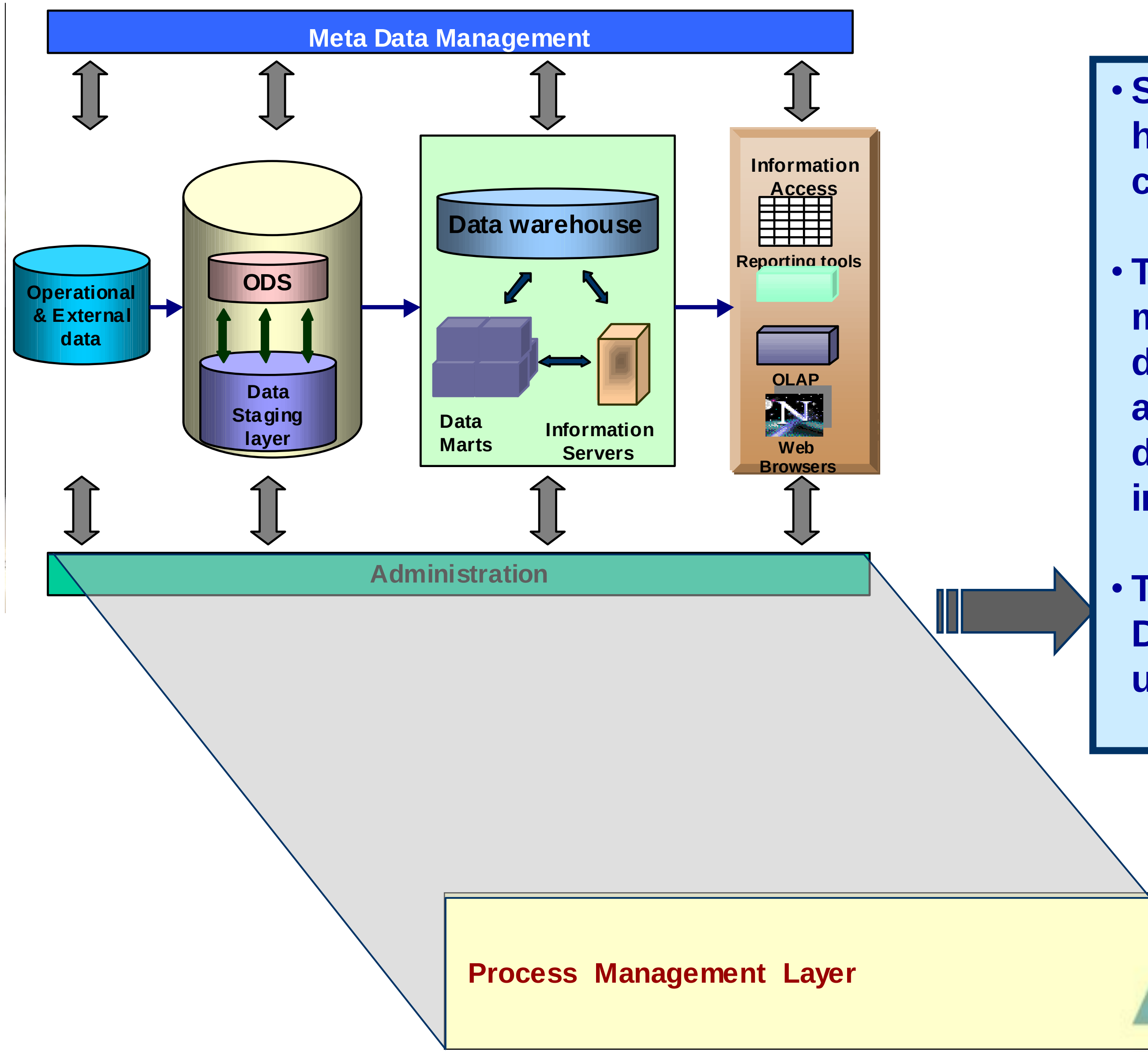
Meta Data layer



- Metadata is data about data.
- Stored in a repository.
- Contains all corporate Metadata resources: database catalogs, data dictionaries

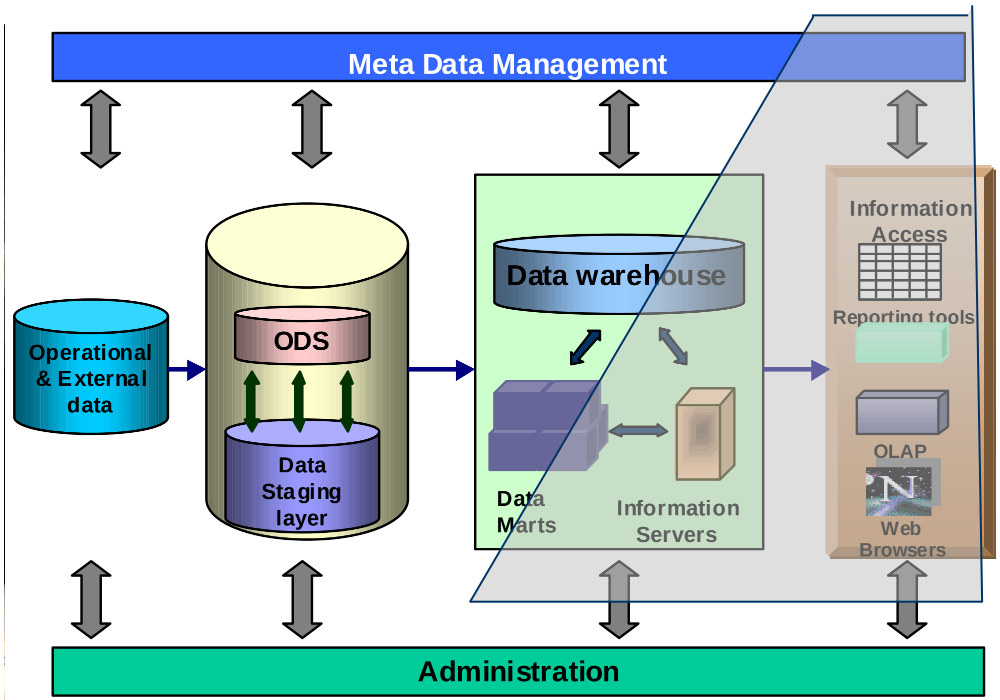
HCL

Process Management layer



- Scheduler or the high-level job control
- To build and maintain the data warehouse and data directory information
- To keep the Data warehouse up-to-date.

Information Access layer



Information Access Layer

- Interfaced with the data warehouse through an OLAP server.
- Performs analytical operations and presents data for analysis.
- End-users generates ad-hoc reports and perform multidimensional analysis using OLAP tools

Data Warehouse Architecture

The following should be considered for a successful implementation of a Data Warehousing solution:

Architecture :

- Open Data Warehousing architecture with common interfaces for product integration
- Data warehouse database server

Tools :

- Data Modeling tools
- Extraction and Transformation/propagation tools
- Analysis/end-user tools: OLAP and Reporting
- Metadata Management tools

Different Approaches for Implementing an Enterprise Data warehouse

What is an Enterprise Datawarehouse?

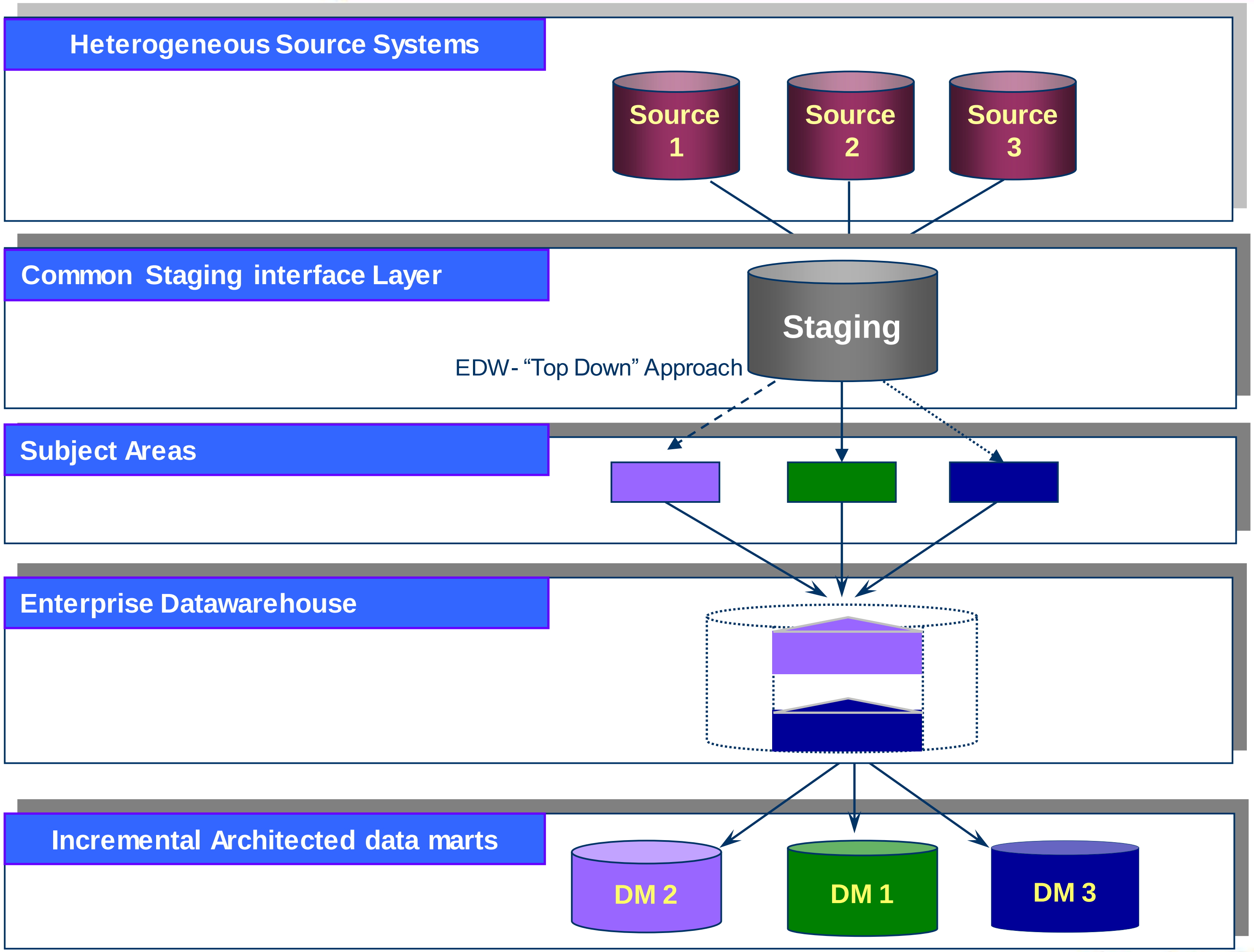
- An **Enterprise Data Warehouse** (EDW) contains

- Separate subject-oriented database.

- Supports detailed analysis of business trends over a period of time

- Used for short- and long-term business planning and decision making covering multiple business units.

EDW- “Top Down” Approach



EDW- “Top Down” Approach

- An EDW is composed of multiple subject areas, such as finance, Human resources, Marketing, Sales, Manufacturing, etc.
- In a top down scenario, the entire EDW is architected, and then a small slice of a subject area is chosen for construction
- Subsequent slices are constructed, until the entire EDW is complete

EDW- “Top Down” Approach

The upsides to a “Top Down” approach are:

1. Coordinated environment

2. Single point of control & development

The downsides to a “Top Down” approach are:

1. “Cross everything” nature of enterprise project

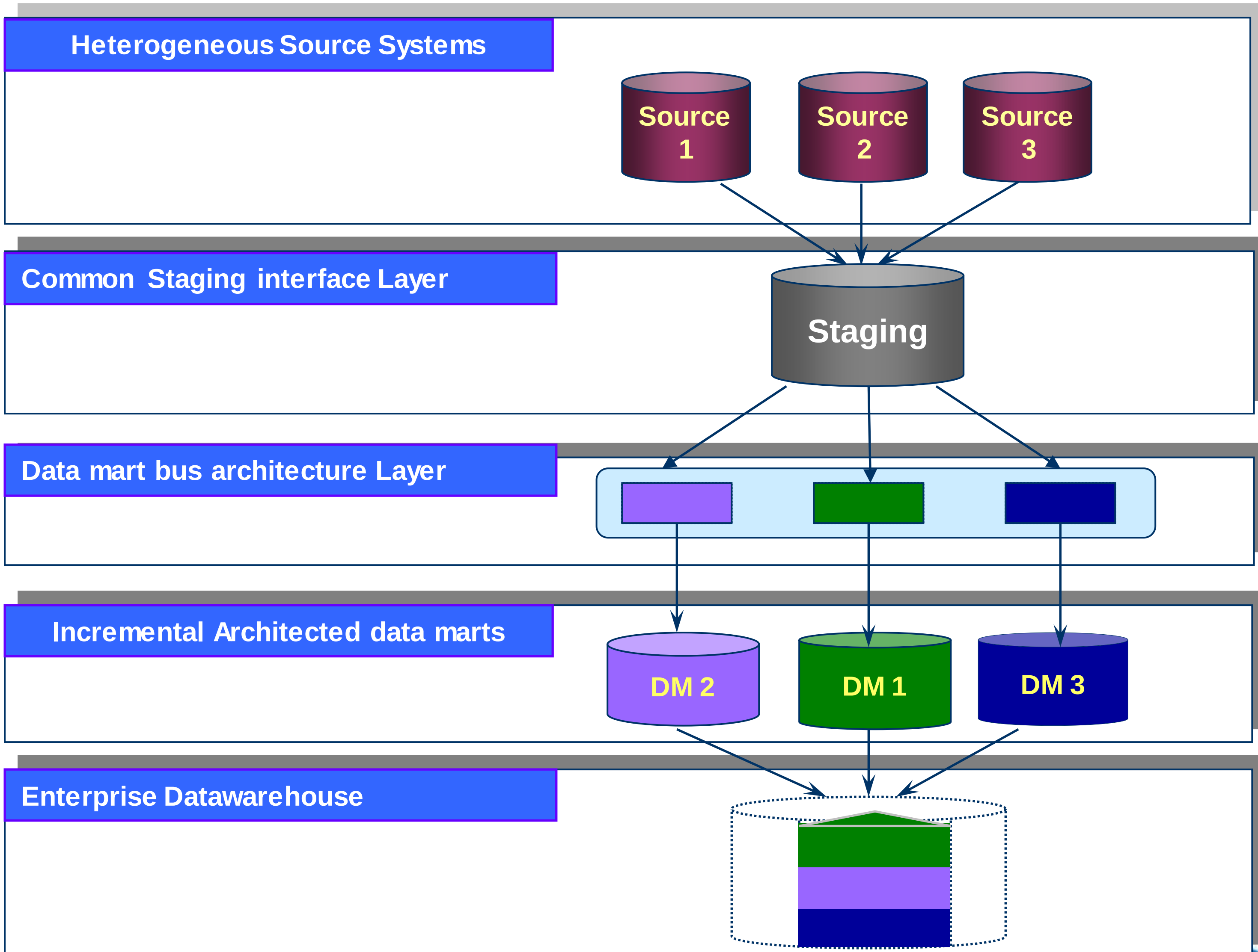
2. Analysis paralysis

3. Scope control

4. Time to market

5. Risk and exposure

EDW- “Bottom up” Approach



EDW- “Bottom up” Approach

- Initially an Enterprise Data Mart Architecture (EDMA) is developed

- Once the EDMA is complete, an initial subject area is selected for the first incremental Architected Data Mart (ADM).

- The EDMA is expanded in this area to include the full range of detail required for the design and development of the incremental ADM.

EDW- “Bottom up” Approach

The upsides to a “bottom up” approach are:

1. Quick ROI
2. Low risk, low political exposure learning and development environment
3. Lower level, shorter-term political will required
4. Fast delivery
5. Focused problem, focused team
6. Inherently incremental

The downsides to a “bottom up” approach are:

1. Multiple team coordination
2. Must have an EDMA to integrate incremental data marts

Data warehouse architecture

- Lot of tools and technologies
- Data warehouse system architectures.
- Top down approach
- Bottom up approach

Building a Data Warehouse

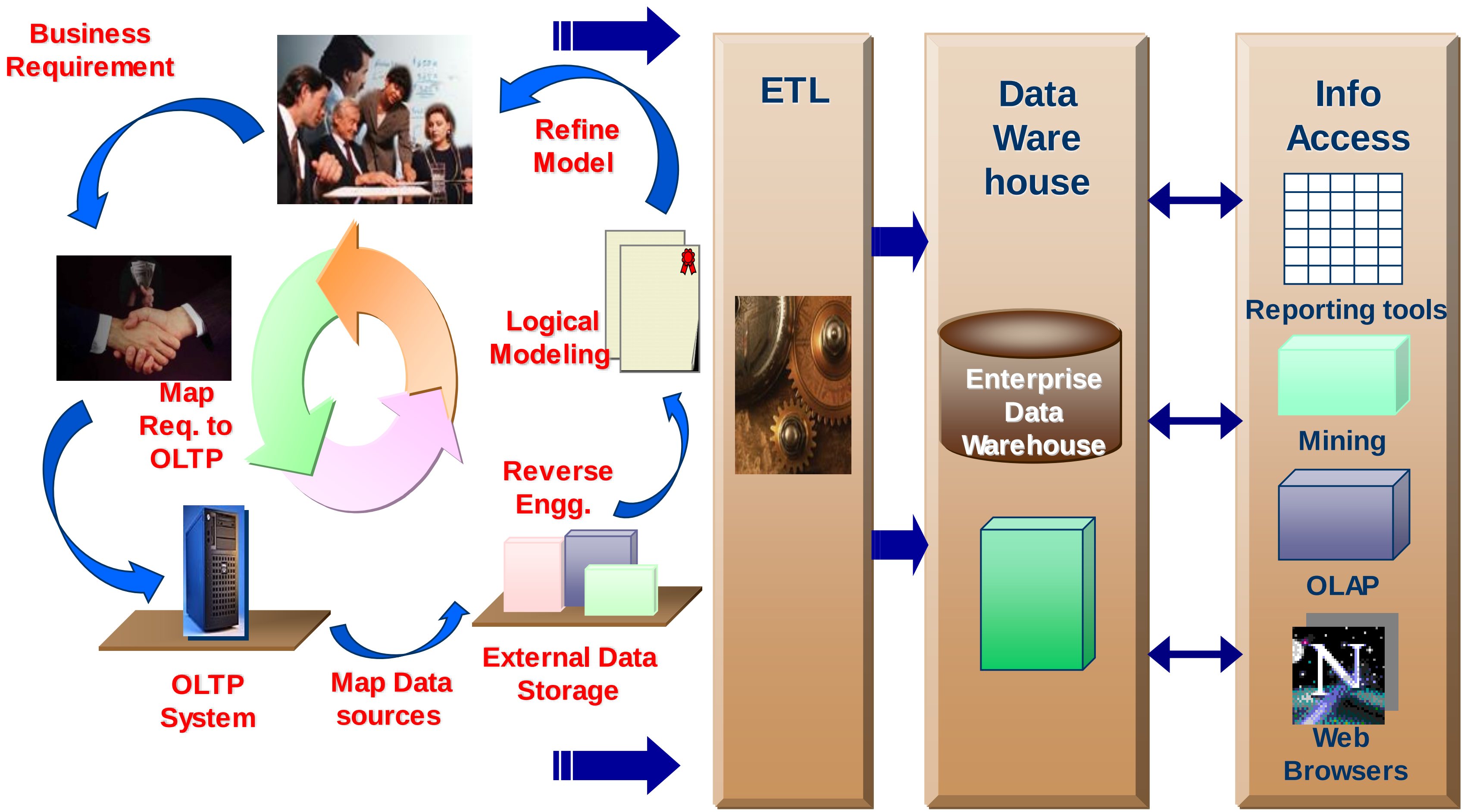
Building a Data Warehouse

The initiatives involved in building a data warehouse are

1. Identify the need and justify the cost
2. Architect the warehouse
3. Choose product and vendors
4. Create a dimensional business model
5. Create the physical model
6. Design & develop extract, transform and load systems
7. Test and refine the data warehouse

Data Warehouse design is driven by business users; Not by the IS team

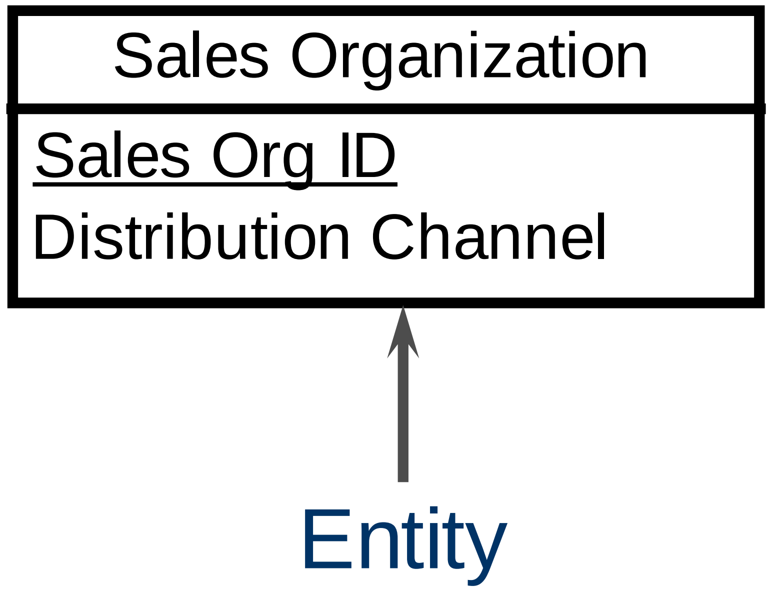
Data Warehouse Life cycle



ER Modeling

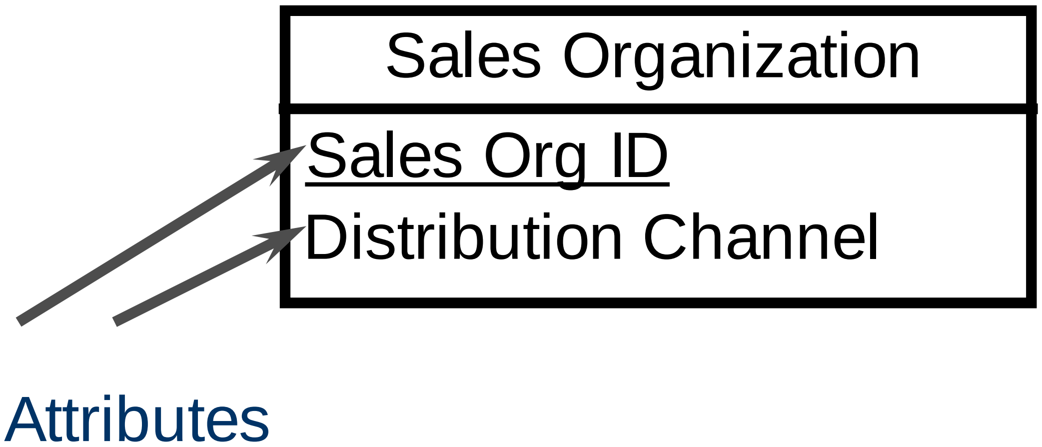
Review of Logical Modeling Terms & Symbols

- *Entities* define specific groups of information



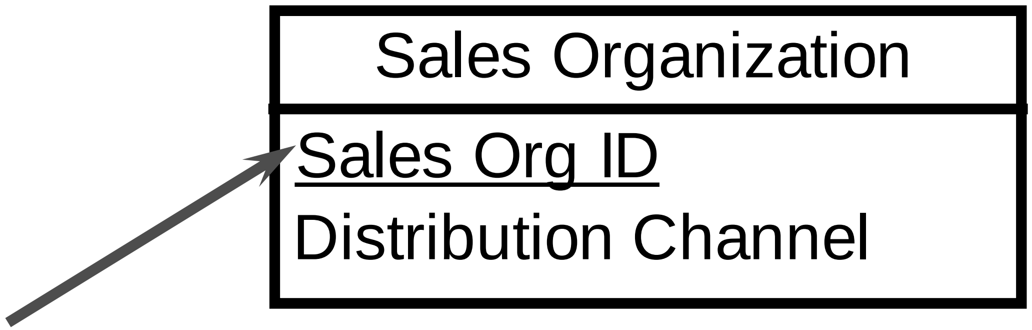
Review of Logical Modeling Terms & Symbols

- Entities are made up of *attributes*



Review of Logical Modeling Terms & Symbols

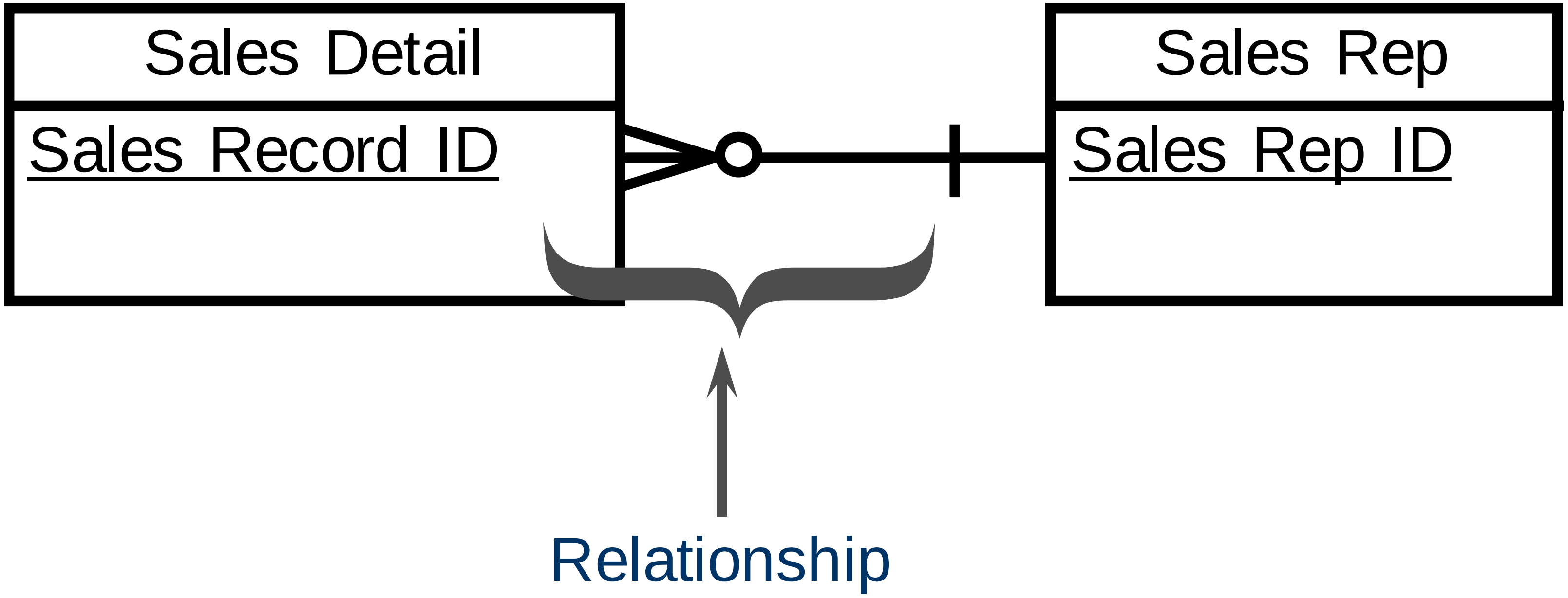
- One or more attribute uniquely *identifies* an instance of an entity



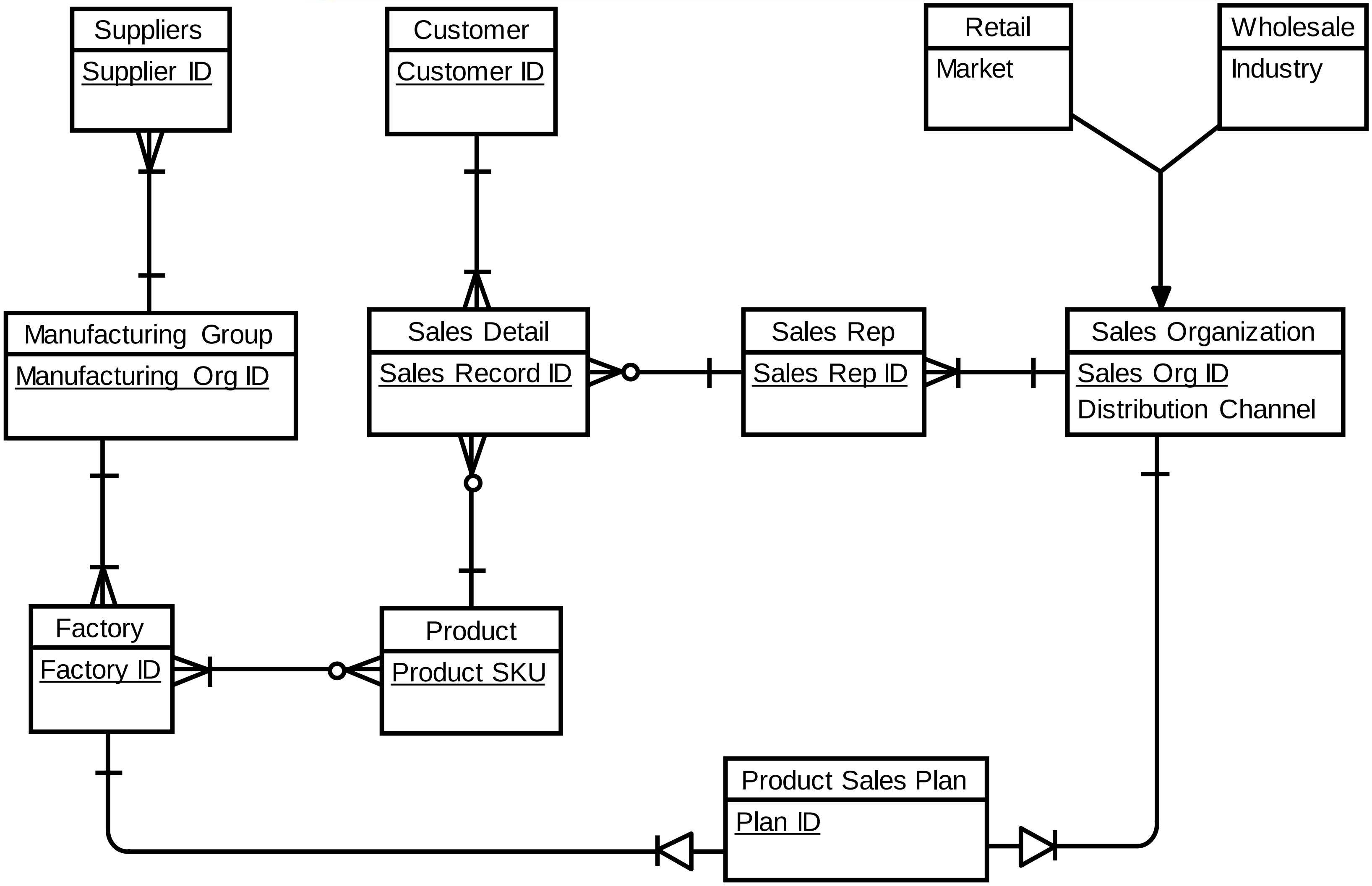
Identifier

Review of Logical Modeling Terms & Symbols

- The logical model identifies *relationships* between entities



Logical Data Model



Dimensional Modeling

Facts and Measures

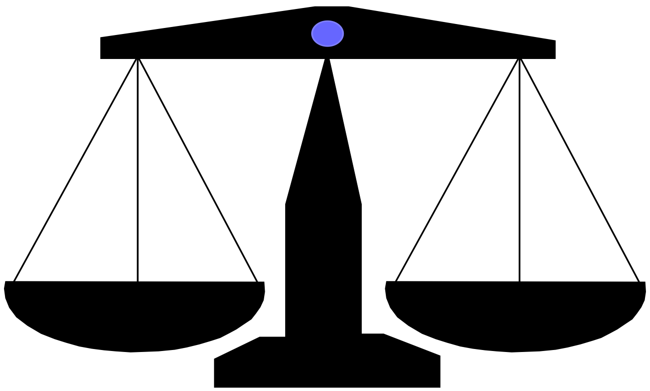
Sales Revenue

Net Profit

Gross Margin

Profitability

Cost

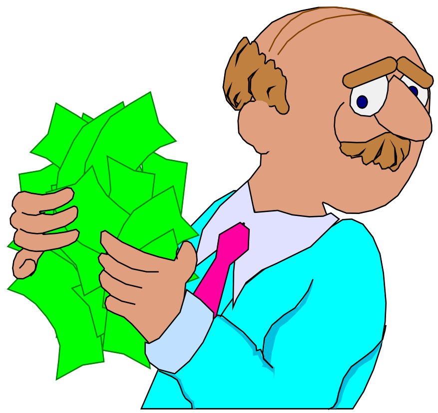


- Facts or Measures are the Key Performance Indicators of an enterprise
- Factual data about the subject area
- Numeric, summarized

HCL

Dimension

Sales Revenue
(Measure)



What was sold ?
Whom was it sold to ?
When was it sold ?
Where was it sold ?

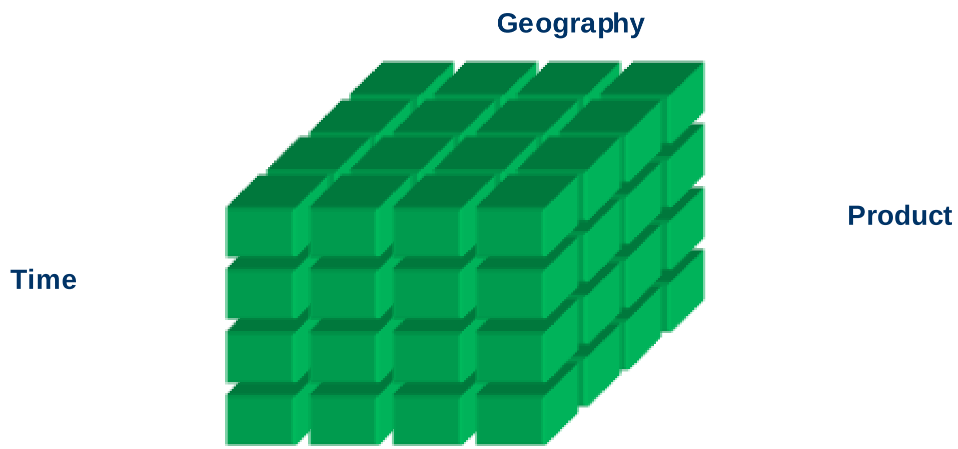
- Dimensions put measures in perspective
- What, when and where qualifiers to the measures
- Dimensions could be products, customers, time, geography etc.

Some Examples of Data warehousing Dimensions

The following Dimensions are common in all Data warehouses in various forms

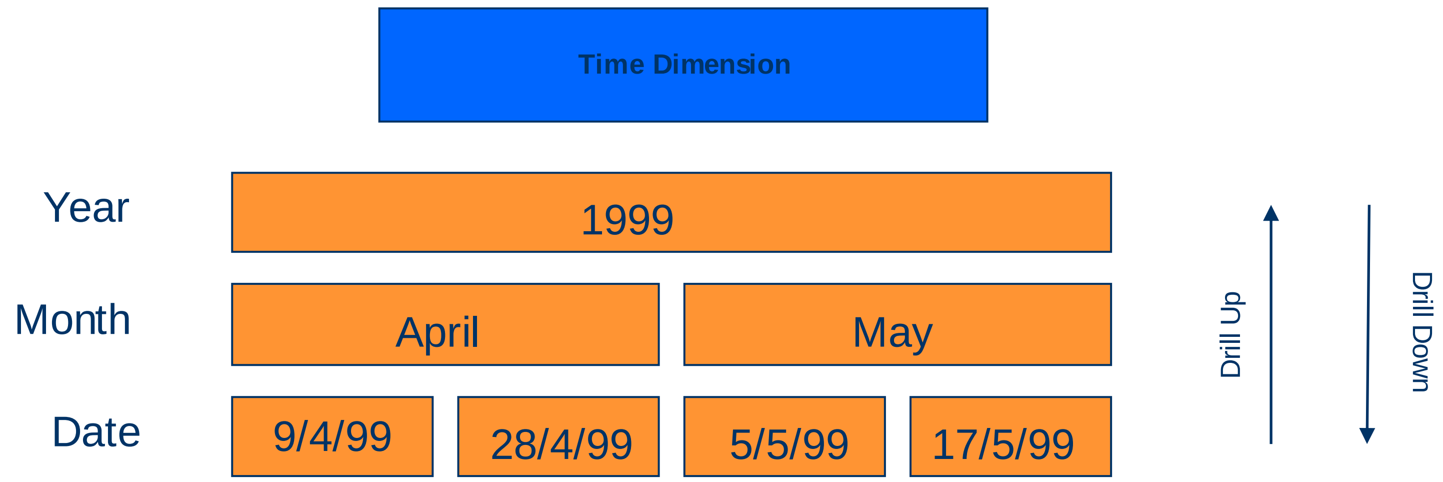
- Product Dimension
- Service Dimension
- Geographic Dimension
- Time dimension

Dimension Elements



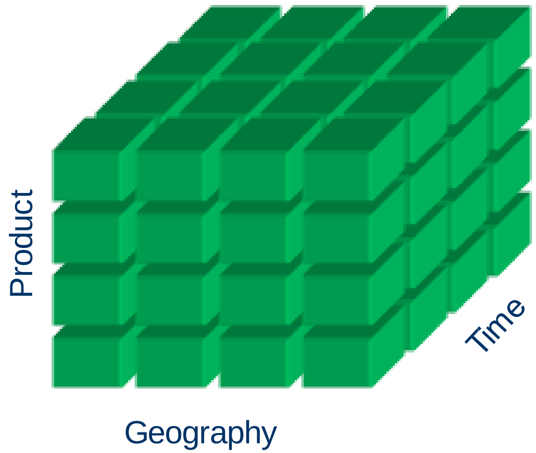
- Components of a dimension
- Represents the natural elements in the business dimension
- Directly related to the dimension
- Facilitates analysis from different perspectives of a dimension
- Often referred to as levels of a dimension

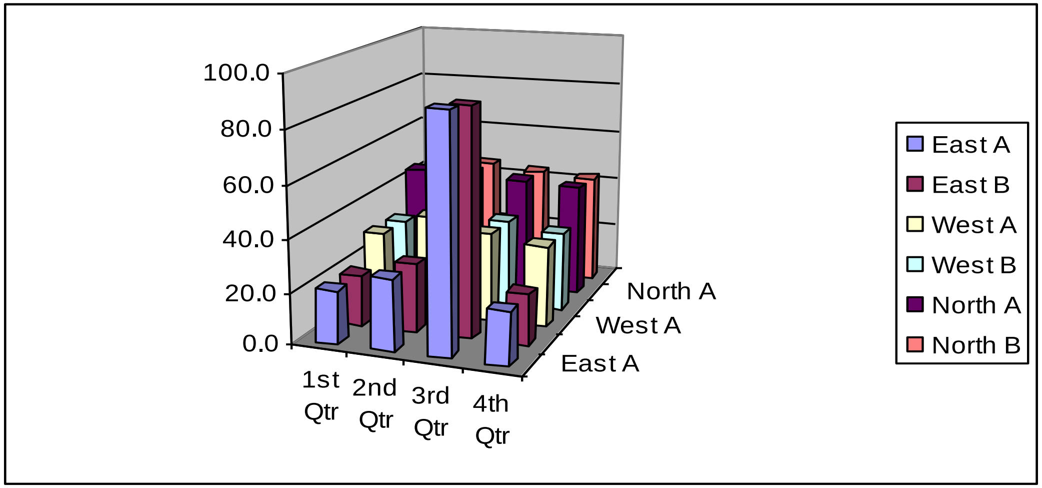
Dimension Hierarchy



- Represents the natural business hierarchy within dimension elements
- Clarifies the drill up, drill down directions
- Each element represents different levels of aggregation
- End users may need custom hierarchies

Multi-Dimensional Analysis





		1st Qtr	2nd Qtr	3rd Qtr	4th Qtr
East	A	20.4	27.4	90.0	20.4
	B	19.8	26.6	87.3	19.8
West	A	30.6	38.6	34.6	31.6
	B	29.7	37.4	33.6	30.7
North	A	45.9	46.9	45.0	43.9
	B	44.5	45.5	43.7	42.6

- Characteristic of online analytical processing (OLAP)

Drill Up & Drill Down

Current Result Set

	1st Qtr	2nd Qtr	3rd Qtr	4th Qtr
East	20.4	27.4	90	20.4
West	30.6	38.6	34.6	31.6
North	45.9	46.9	45	43.9

Up

	1999
East	158.2
West	135.4
North	181.7

Down

Jan	Feb	Mar	Apr	May
5.712	6.528	8.16	7.672	8.768
8.568	9.792	12.24	10.808	12.352
12.852	14.688	18.36	13.132	15.008

- Drill down is a process of requesting for detailed information
- Drill up is a process of summarizing the existing information

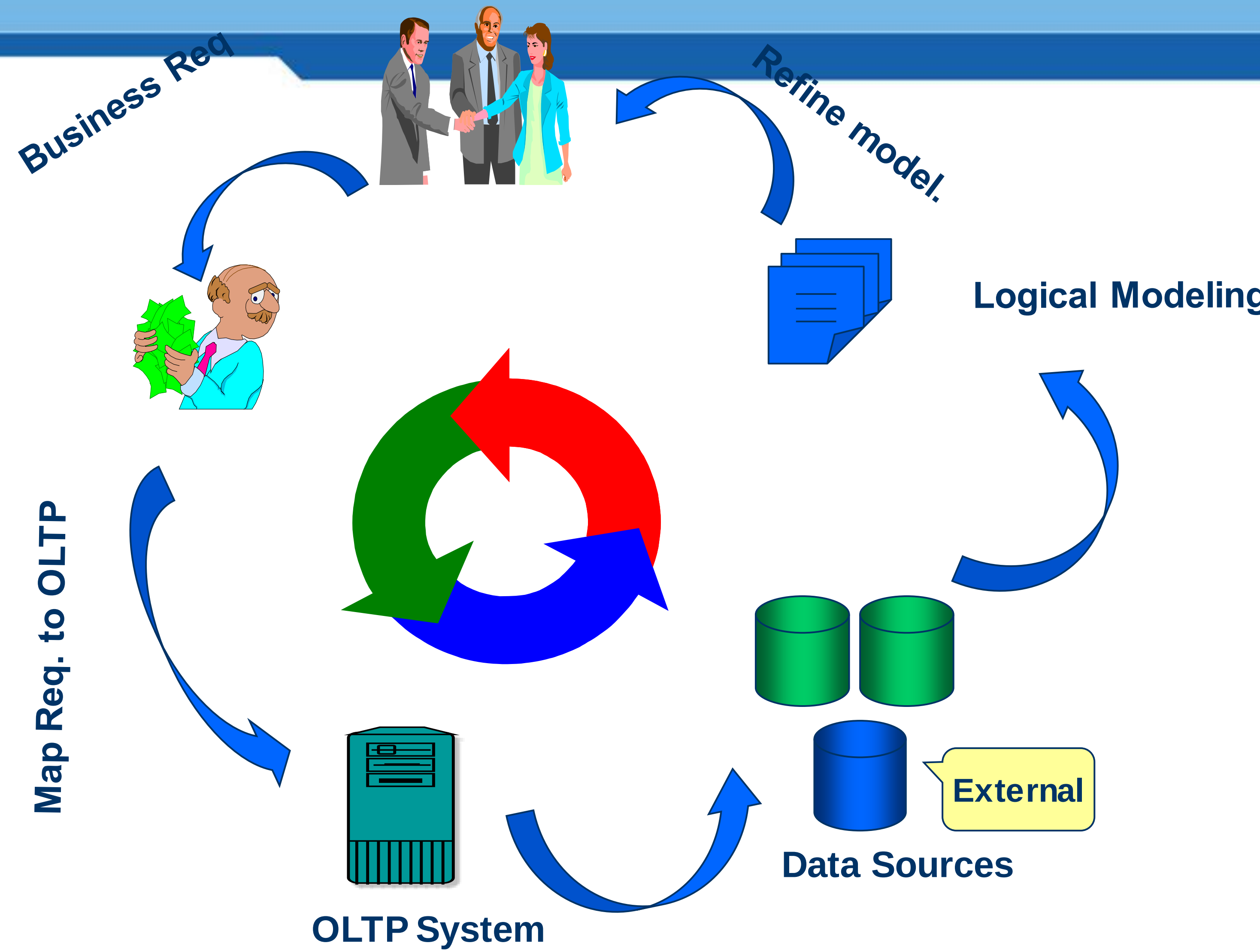
Dimensional Modeling

Subject Area	What do you want to know about?
Atomic Detail	What level of detail do you need?
Dimensions	Analyze key performance indicators
Facts	Measures
Frequency of Update	How fresh do you need it?
Depth of History	How far back do you need to know it?

Requirements for a Dimensional model

- Clean, current, accurate logical models
- Physical models
- A subject area model
- Star / Snowflake schema design

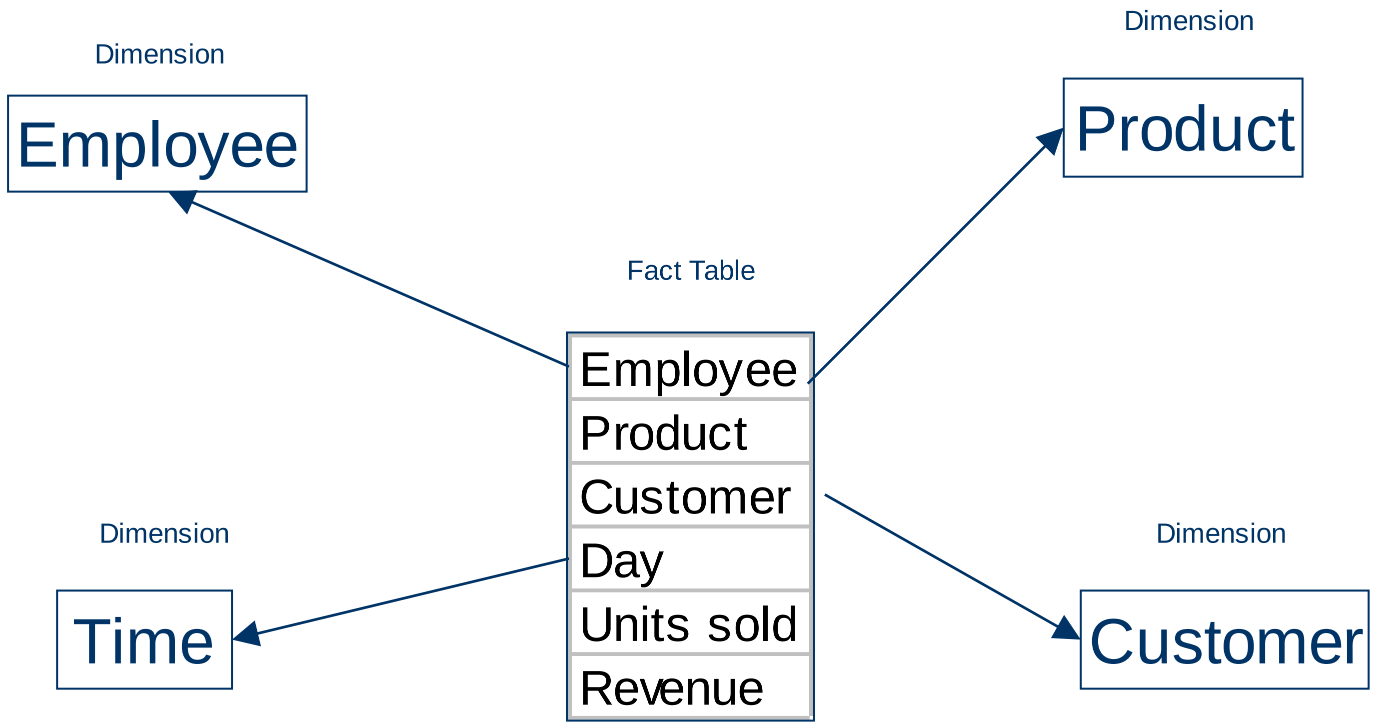
Dimensional Modeling Methodology



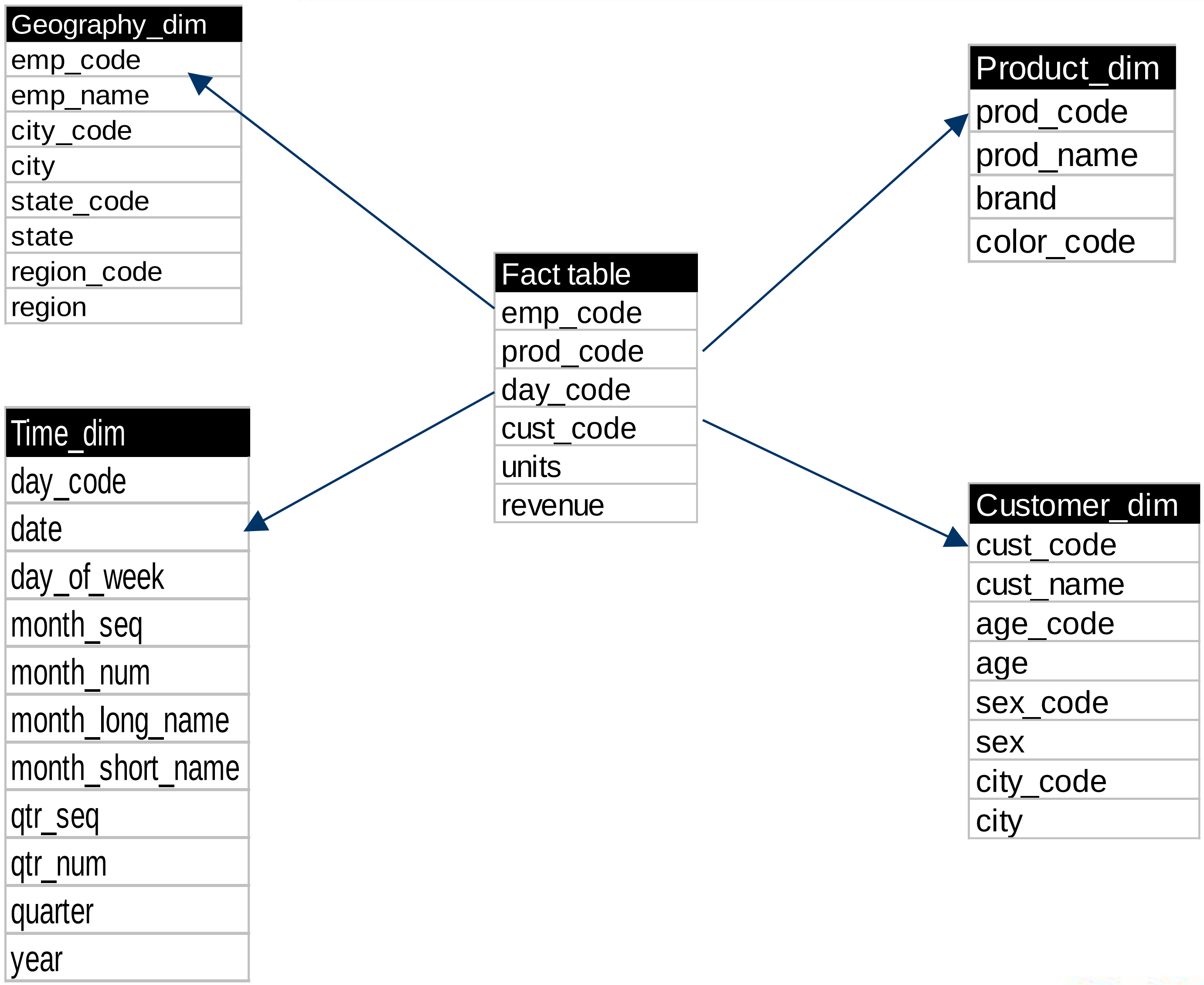
Techniques for Implementing a Dimensional model

- Star Schema
- Snow-flake Schema
- Hybrid Schema
- Optimal Snow-flake Schema

Star schema- Logical structure



Star schema: Logical view



Star schema characteristics

- A star schema is a highly denormalized, query-centric model where the basic premise is that information can be broken into two groups: facts and dimensions.

- In a star schema, facts are in a single place (the fact table) and the descriptions (or elements) that lead to those facts are in dimension tables.

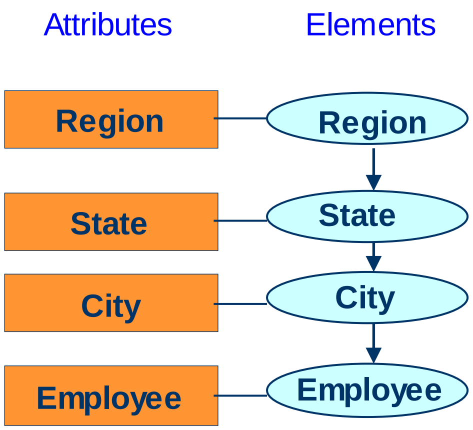
- The star schema is built for simplicity and speed. The assumption behind it is that the database is static with no updates being performed online

Star schema: Dimension Table

PK

Geography_dim

Empl_Code	empl_name	city_code	city	state_code	state	region_code	region
2341	Mike King	101	Atlantic city	NJ	New Jersey	1	New Jersey
3424	Jim McCann	106	Chicago	IL	Illinois	2	Illinois
1232	Kitty Stokes	104	Austin	PA	Pennsylvania	1	New Jersey
3554	Clem Akins	102	Medford	NJ	New Jersey	1	New Jersey
3963	Duncan Moore	101	Atlantic city	NJ	New Jersey	1	New Jersey
2924	Dawn McGuire	103	Englewood	NJ	New Jersey	1	New Jersey
2673	Joe Becker	105	Alverton	PA	Pennsylvania	1	New Jersey
3253	Geoff Bergren	107	Springfield	IL	Illinois	2	Illinois
234	Garth Boyd	106	Chicago	IL	Illinois	2	Illinois
2342	Lin Cepele	104	Austin	PA	Pennsylvania	1	New Jersey



- De-normalized structure
- Easy navigation within the dimension

Star schema: Fact Table

day_code	prod_code	cust_code	empl_code	units sold	revenue
1211	345	1231123	1232	23	7935
1211	22	1245223	3554	12	264
1211	112	1522342	3963	6	672
1212	233	1524665	2924	34	7922
1212	112	1366454	2673	76	8512
1212	22	1403453	3554	22	484

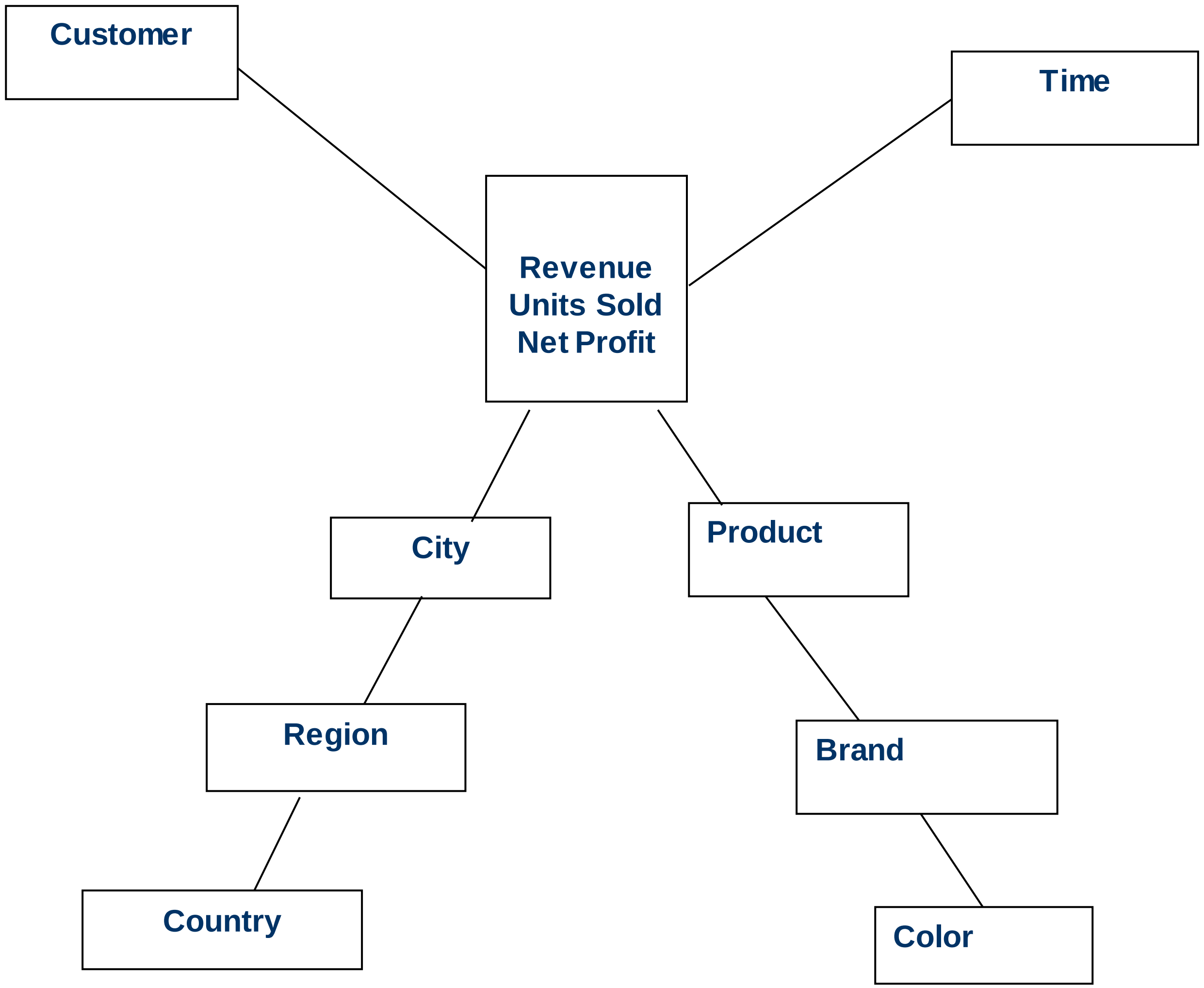
sales_fact

Dimension Keys

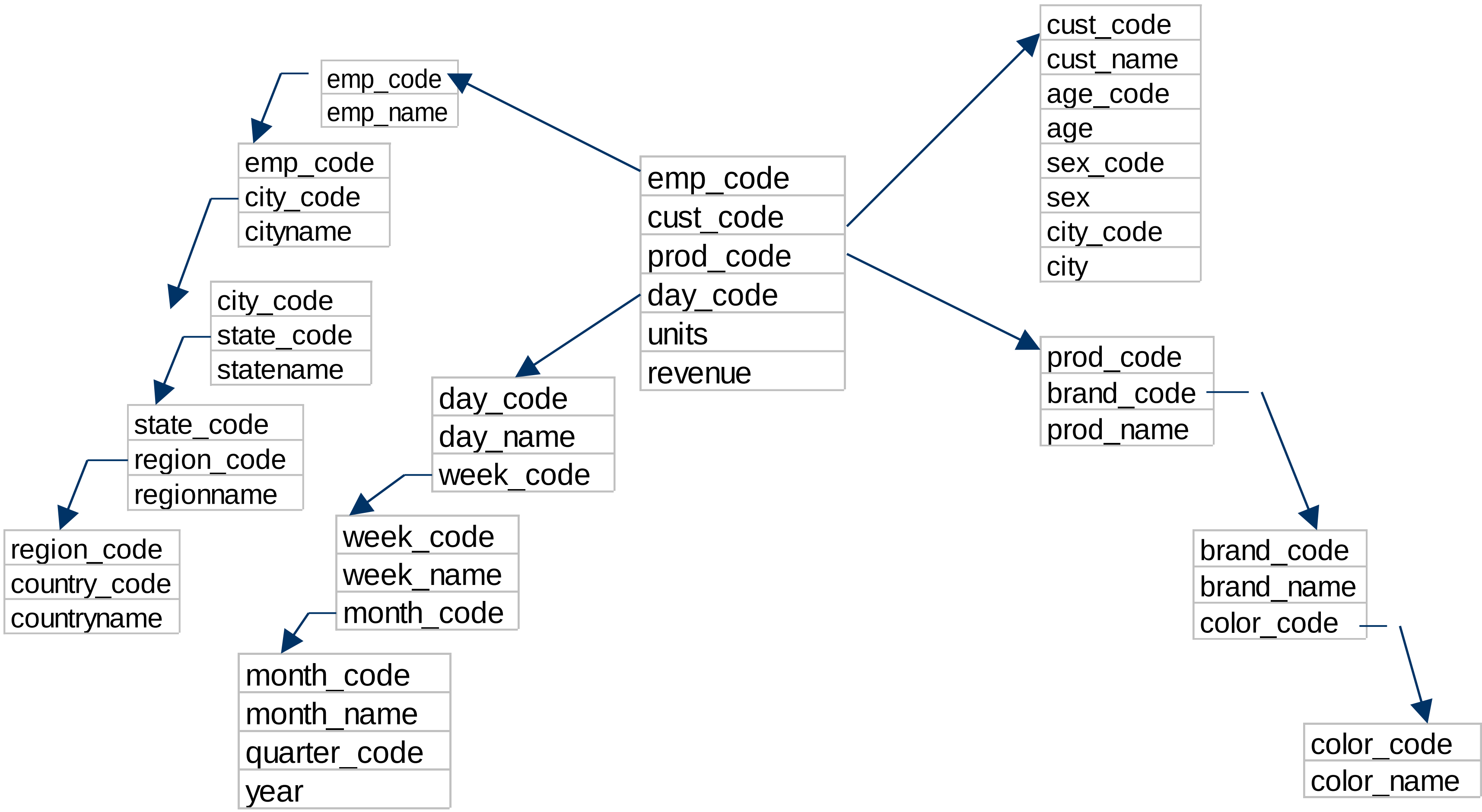
Measures

- Contains columns for measures and dimensions

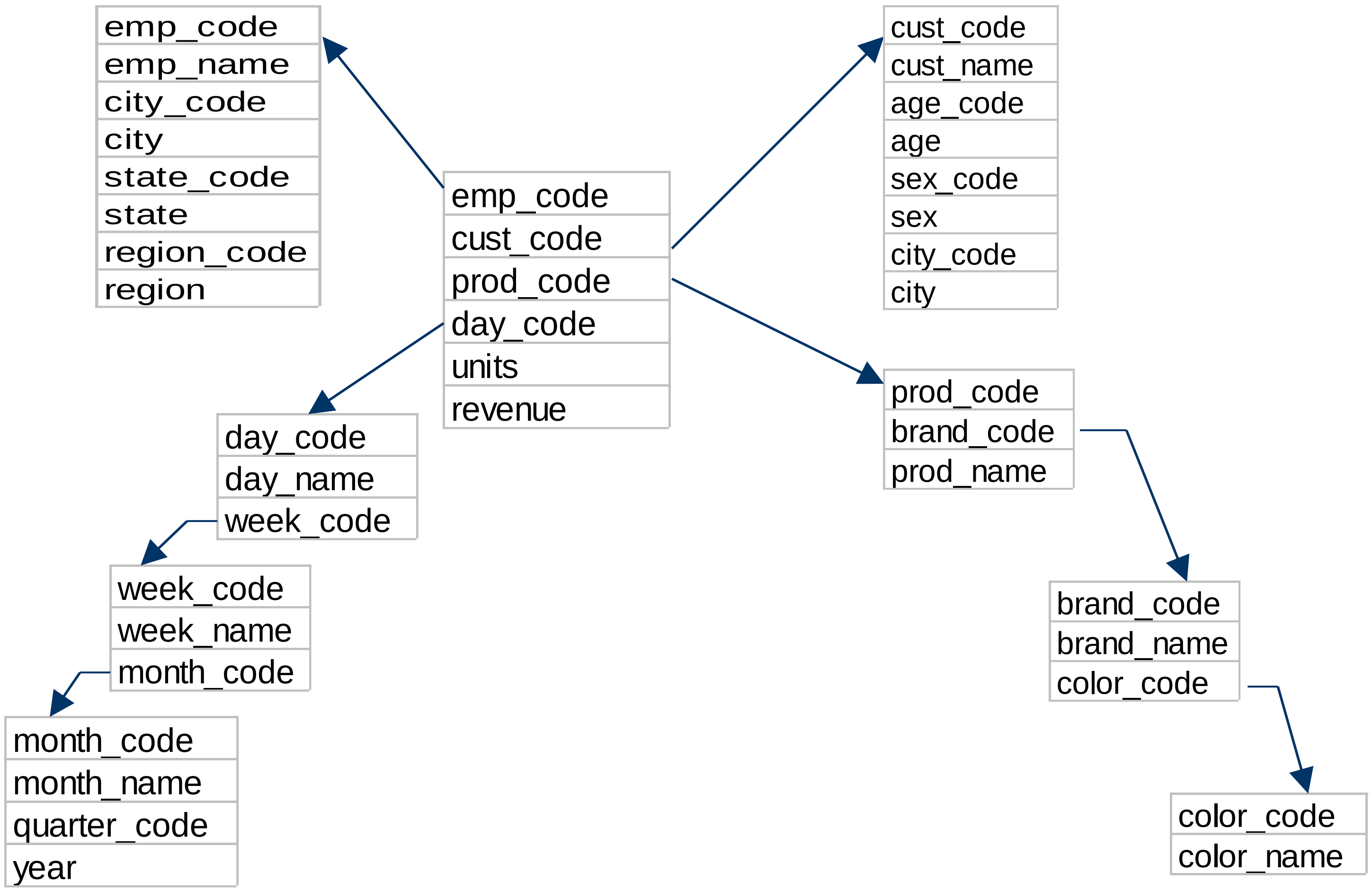
Snow-flake schema



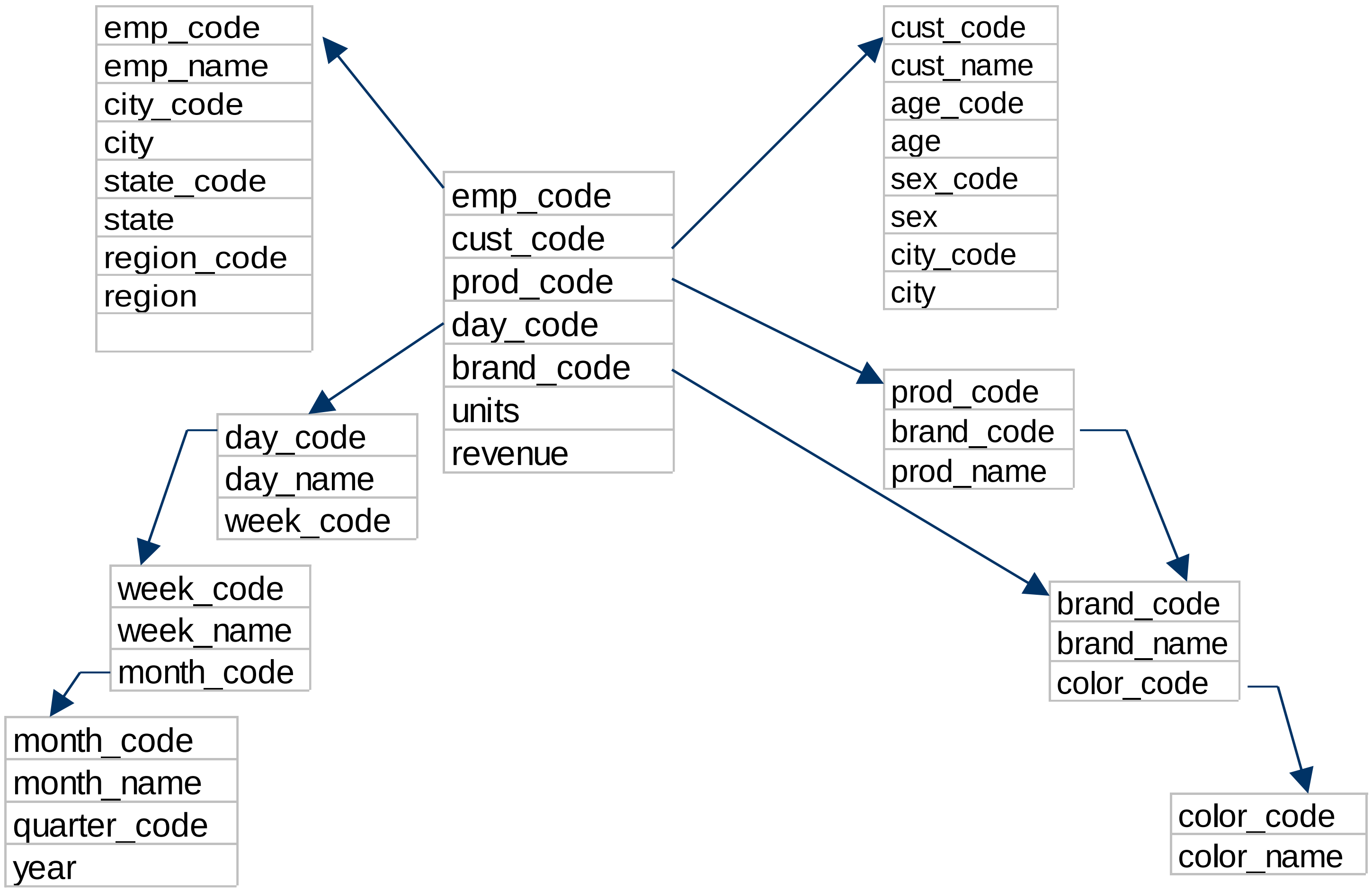
Snow-flake: Logical view



Hybrid schema: Physical view

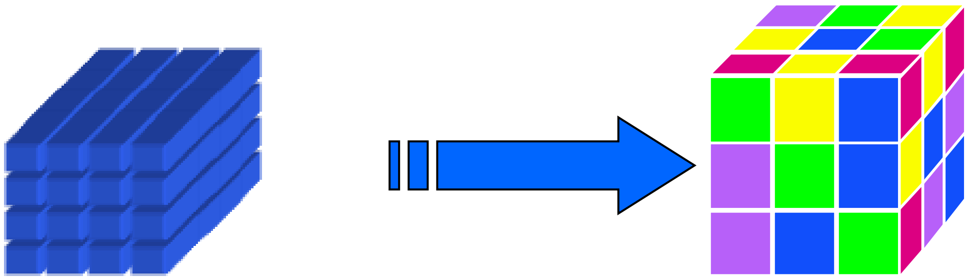


Optimal Snow-flake schema



What is a Slowly Changing Dimension?

- Although dimension tables are typically static lists, most dimension tables do change over time.
- Since these changes are smaller in magnitude compared to changes in fact tables, these dimensions are known as slowly growing or slowly changing dimensions.



Slowly Changing Dimension -Classification

Slowly changing dimensions are classified into three different types

- TYPE I
- TYPE II
- TYPE III

Slowly Changing Dimensions Type I

Source

Emp id	Name	Email
1001	Shane	

Target

Emp id	Name	Email
1001	Shane	Shane@xyz.com

Source

Emp id	Name	Email
1001	Shane	Shane@abc.co.in

Target

Emp id	Name	Email
1001	Shane	Shane@abc.co.in

~~Shane@xyz.com~~

Slowly Changing Dimensions Type II

Source

Emp id	Name	Email
10	Shane	Shane@xyz.com

Target

PM_PRIMARY KEY	Emp id	Name	Email	PM_VERSION_NUMBER
1000	10	Shane	Shane@xyz.com	0

Slowly Changing Dimensions Type II - Versioning

Source

Emp id	Name	Email
10	Shane	Shane@ abc.co.in

PM_PRIMARYKEY	Emp id	Name	Email	PM_VERSION_NUMBER
1000	10	Shane	Shane@ xyz.com	0
1001	10	Shane	Shane@ abc.co.in	1

Target

Slowly Changing Dimensions Type II - Versioning

Source

Emp id	Name	Email
10	Shane	Shane@ abc.com

Target

PM_PRIMARYKEY	Emp id	Name	Email	PM_VERSION_NUMBER
1000	10	Shane	Shane@ xyz.com	0
1001	10	Shane	Shane@ abc.co.in	1
1003	10	Shane	Shane@ abc.com	2

Slowly Changing Dimensions Type II - Effective Date

Emp id	Name	Email
10	Shane	Shane@xyz.com

Source

PM_PRIMARY KEY	Emp id	Name	Email	PM_BEGIN_DATE	PM_END_DATE
1000	10	Shane	Shane@xyz.com	01/01/00	

Target

Slowly Changing Dimensions Type II - Effective Date

Source

Emp id	Name	Email
10	Shane	Shane@ abc.co.in

Target

PM_PRIMARYKEY	Emp id	Name	Email	PM_BEGIN_DATE	PM_END_DATE
1000	10	Shane	Shane@ xyz.com	01/01/00	02/01/00
1001	10	Shane	Shane@ abc.co.in	03/01/00	

Slowly Changing Dimensions Type II - Effective Date

Source

Emp id	Name	Email
10	Shane	Shane@ abc.com

Target

PM_PRIMARYKEY	Emp id	Name	Email	PM_BEGIN_DATE	PM_END_DATE
1000	10	Shane	Shane@ xyz.com	01/01/00	03/01/00
1001	10	Shane	Shane@ abc.co.in	03/01/00	04/02/00
1003	10	Shane	Shane@ abc.com	05/02/00	

Slowly Changing Dimensions Type III

Emp id	Name	Email
10	Shane	Shane@xyz.com

Source

PM_PRIMARYKEY	Emp id	Name	Email	PM_Prev_Column Name	PM_EFFECT_DATE
1	10	Shane	Shane@xyz.com		01/01/00

Target

Slowly Changing Dimensions Type III

Source

Emp id	Name	Email
10	Shane	Shane@abc.co.in

PM_PRIMARYKEY	Emp id	Name	Email	PM_Prev_ColumnName	PM_EFFECT_DATE
1	10	Shane	Shane@abc.co.in	Shane@xyz.com	01/02/00

Target

Slowly Changing Dimensions Type III

Source

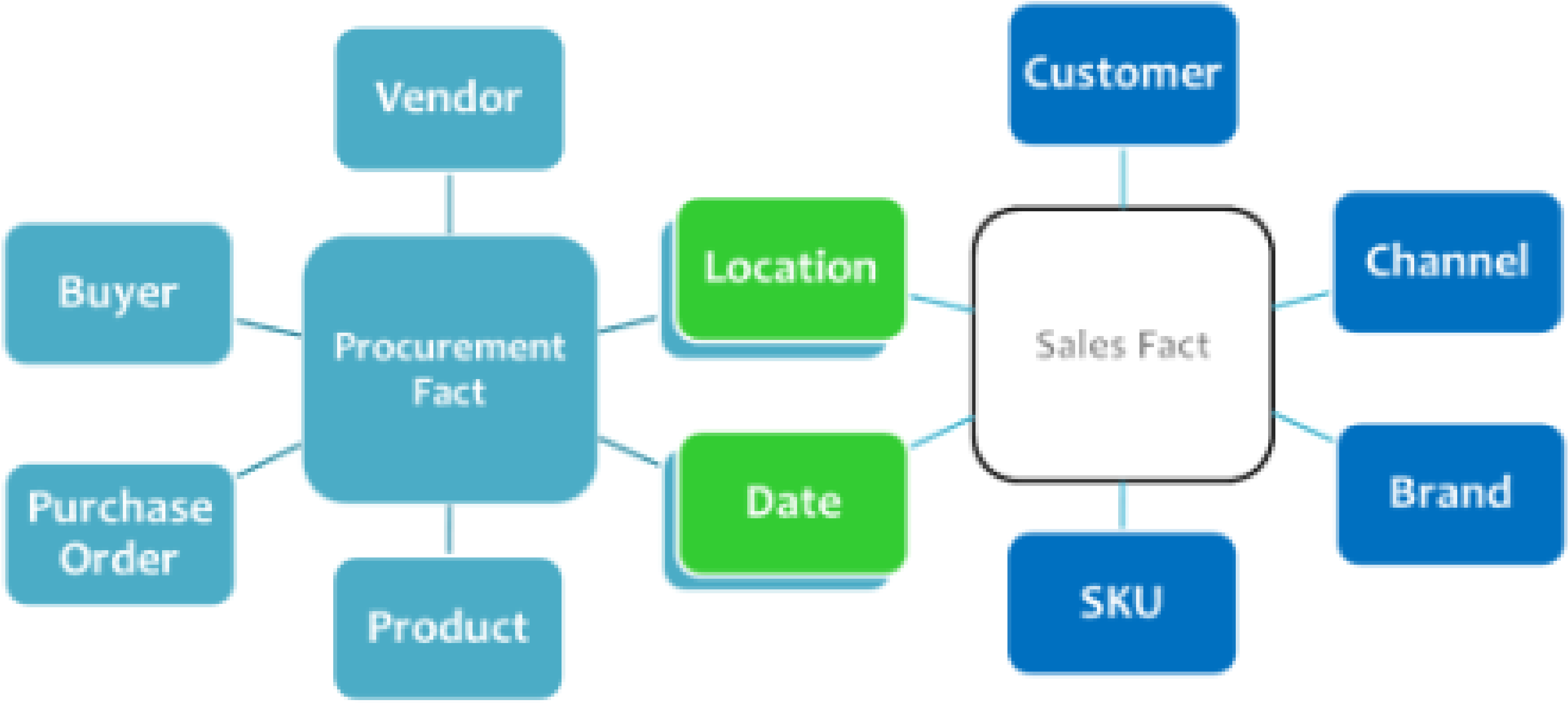
Emp id	Name	Email
10	Shane	Shane@ abc.com

PM_PRIMARYKEY	Emp id	Name	Email	PM_Prev_ColumnName	PM_EFFECT_DATE
1	10	Shane	Shane@ abc.com	Shane@ abc.co.in	01/03/00

Target

Conformed Dimensions

- Conformed dimensions are those which are consistent across Data marts. Dimension tables conform when attributes in separate dimension tables have the same column names and domain contents. Essential for integrating the Data marts into an Enterprise Data warehouse. Drill across reports can be generated by joining two different Fact tables through conformed dimension.



Degenerated Dimensions

- A degenerated dimension is when the dimension attribute is stored as part of Fact table and not in a separate dimension table. Degenerate dimensions are most common with transaction and accumulating snapshot fact tables. Product Id comes from Product dimension table. Invoice number is a standalone attribute and has no other attribute associated with it. An Invoice number is crucial since business would want to know the quantity of the product.

Invoice Number	Product Id	Quantity	Amount
11223344	12345	4	100
11223344	67892	3	200
44556677	11123	2	300
11223344	44567	1	400

- Degenerate dimensions commonly occur when the fact table's grain is a single transaction (or transaction line). Transaction control header numbers assigned by the operational business process are typically degenerate dimensions, such as order, ticket, credit card transaction, or check numbers. These degenerate dimensions are natural keys of the "parents" of the line items.
- Even though there is no corresponding dimension table of attributes, degenerate dimensions can be quite useful for grouping together related fact tables rows. For example, retail point-of-sale transaction numbers tie all the individual items purchased together into a single market basket. In health care, degenerate dimensions can group the claims items related to a single hospital stay or episode of care.

Junk Dimensions

- A **junk dimension** combines several low-cardinality flags and attributes into a single dimension table rather than modeling them as separate dimensions. Three aspects of junk dimension processing: building the initial dimension, incorporating it into the fact processing, and maintaining it over time.

1. **Build the Initial Junk Dimension**
   - If the cardinality of each attribute is relatively low, and there are only a few attributes, then the easiest way to create the dimension is to cross-join the source system lookup tables. This creates all possible combinations of attributes, even if they might never exist in the real world.
   - If the cross-join of the source tables is too big, or if you don't have source lookup tables, you will need to build your junk dimension based on the actual attribute combinations found in the source data for the fact table. The resulting junk dimension is often significantly smaller because it includes only combinations that actually occur.
   - We'll use a simple health care example to show both of these combination processes. Hospital admissions events often track several standalone attributes, including the admission type and level of care required, as illustrated below in the sample rows from the source system lookup and transaction tables

Admit_Type_Source

Admit_ Type_ID	Admit_Type_ Descr
1	Walk-in
2	Appointment
3	ER
4	Transfer

Care_Level_Source

Care_ Level_ ID	Care_Level_ Descr
1	ICU
2	Pediatric ICU
3	Medical Floor

Fact_Admissions_Source

Admit_ Type_ID	Care_ Level_ID	Admission_Count
1	1	1
2	1	1
2	2	1
5	3	1

HCL

Junk Dimensions Continues…

- The following SQL uses the cross-join technique to create all 12 combinations of rows (4×3) from these two source tables and assign unique surrogate keys.

```
SELECT ROW_NUMBER() OVER( ORDER BY Admit_Type_ID, Care_Level_ID) AS Admission_Info_Key,
Admit_Type_ID, Admit_Type_Descr, Care_Level_ID, Care_Level_Descr
FROM Admit_Type_Source
CROSS JOIN Care_Level_Source;
```

- In the second case, when the cross-join would yield too many rows, you can create the combined dimension based on actual combinations found in the transaction fact records. The following SQL uses outer joins to prevent a violation of referential integrity when a new value shows up in a fact source row that is not in the lookup table.

```
SELECT ROW_NUMBER() OVER(ORDER BY F.Admit_Type_ID) AS Admission_Info_Key,
F.Admit_Type_ID, ISNULL(Admit_Type_Descr, ‘Missing Description’) Admit_Type_Descr,
F.Care_Level_ID, ISNULL(Care_Level_Descr, ‘Missing Description’)
Care_Level_Descr — substitute NVL() for ISNULL() in Oracle
FROM Fact_Admissions_Source F
LEFT OUTER JOIN Admit_Type_Source C ON
F.Admit_Type_ID = C.Admit_Type_ID
LEFT OUTER JOIN Care_Level_Source P ON
F.Care_Level_ID = P.Care_Level_ID;
```

- Our example Fact_Admissions_Source table only has four rows which result in the following Admissions_Info junk dimension. Note the Missing Description entry in row 4.

Admission_Info_Key	Admit_Type_ID	Admit_Type_Descr	Care_Level_ID	Care_Level_Descr
1	1	Walk-In	1	ICU
2	2	Appointment	1	ICU
3	2	Appointment	2	Pediatric ICU
4	5	Missing Description	3	Medical Floor

Junk Dimensions Continues…

2. Incorporate the Junk Dimension into the Fact Row Process

- Once the junk dimension is in place, you will use it to look up the surrogate key that corresponds to the combination of attributes found in each fact table source row. Some of the ETL tools do not support a multi-column lookup join, so you may need to create a work-around. In SQL, the lookup query would be similar to the second set of code above, but it would join to the junk dimension and return the surrogate key rather than joining to the lookup tables.

3. Maintain the Junk Dimension

- You will need to check for new combinations of attributes every time you load the dimension. You could apply the second set of SQL code to the incremental fact rows and select out only the new rows to be appended to the junk dimension as shown below.

```
SELECT * FROM ( {Select statement from second SQL code listing} ) TabA
WHERE TabA.Care_Level_Descr = ‘Missing Description’
OR TabA.Admit_Type_Descr = ‘Missing Description’ ;
```

- In this example, it would select out row 4 in the junk dimension. Identifying new combinations could be done as part of the fact table surrogate key substitution process, or as a separate dimension processing step prior to the fact table process. In either case, your ETL system should raise a flag and notify the appropriate data steward if it identifies a missing entry.

Facts - Definition

- **Definition:** Fact tables are the foundation of the data warehouse. They contain the fundamental measurements of the enterprise, and they are the ultimate target of most data warehouse queries.

- **Measure Types:** Fact table can store different types of measures such as additive, non-additive, semi-additive.
  1. Additive – As its name implied, additive measures are measures which can be added to all dimensions.
  2. Non-additive – different from additive measures, non-additive measures are measures that cannot be added to all dimensions.
  3. Semi-additive – semi-additive measures are the measure that can be added to only some dimensions and not across other.

Facts – Measure Types

Additive – For a Retail domain, we might have following Fact Tables.

Date
Store
Product
Sales_Amount

The purpose of this table is to record the sales amount for each product in each store on daily basis. Sales_Amount is the fact. In this case, Sales_Amount is an additive fact, because you can sum up this fact along any of the three dimensions present in the fact table -- date, store, and product. For example, the sum of Sales_Amount for all 7 days in a week represents the total sales amount for that week.

Semi-Additive & Non-Additive – For a Banking domain, we might have following Fact Tables.

Date
Account
Current_Balance
Profit_Margin

The purpose of this table is to record the current balance for each account at the end of each day, as well as the profit margin for each account for each day. Current_Balance and Profit_Margin are the facts. Current_Balance is a **semi-additive** fact, as it makes sense to add them up for all accounts (what's the total current balance for all accounts in the bank?), but it does not make sense to add them up through time (adding up all current balances for a given account for each day of the month does not give us any useful information). Profit_Margin is a **non-additive** fact, for it does not make sense to add them up for the account level or the day level..

Building Fact

- **Build Fact:** Design Steps to build Fact

1. Stay True to the Grain: The grain is the business definition of what a single fact table record represents. The grain declaration is not a list of dimensional foreign keys that implement a primary key for the fact table. Rather, the grain is the description of the measurement event in the physical world that gives rise to a measurement. When the grocery store scanner measures the quantity and the charged price of a product being purchased, the grain is literally the beep of the scanner. That is a great grain definition!

2. Build Up from the Lowest Possible Grain: The data warehouse should always be built on fact tables expressed at the lowest possible grain. In the example, the beep of the grocery store cash register is the lowest possible grain because it cannot be divided any further. Fact tables at the lowest grain are the most expressive because they have the most complete set of possible dimensions for that business process. The beep grain fact table could have Date, Store, Product, Cashier, Manager, Customer, Promotion, Competition, Basket and even Weather if all these data sources can be marshaled when the fact records are created. Higher grain aggregated tables such as category sales by district cannot support all these dimensions and therefore are much less expressive.

Building Fact

3. Three Kinds of Fact Tables: If you stay true to the grain, then all of your fact tables can be grouped into just three types: transaction grain, periodic snapshot grain and accumulating snapshot grain.
   - The **transaction grain** corresponds to a measurement taken at a single instant. The grocery store beep is a transaction grain. The measured facts are valid only for that instant and for that event. The next measurement event could happen one millisecond later or next month or never. Thus, transaction grain fact tables are unpredictably sparse or dense. We have no guarantee that all the possible foreign keys will be represented. Transaction grain fact tables can be enormous, with the largest containing many billions of records.
   - The **periodic snapshot grain** corresponds to a predefined span of time, often a financial reporting period. The measured facts summarize activity during or at the end of the time span. The periodic snapshot is predictably dense, and applications can rely on combinations of keys always being present. Periodic snapshot fact tables can also get large. A bank with 20 million accounts and a 10-year history would have 2.4 billion records in the monthly account periodic snapshot!
   - The **accumulating snapshot** fact table corresponds to a predictable process that has a well-defined beginning and end. Order processing, claims processing, service call resolution and college admissions are typical candidates. The grain of an accumulating snapshot for order processing, for example, is usually the line item on the order. Accumulating snapshot records are revisited and overwritten as the process progresses through its steps from beginning to end. Accumulating snapshot fact tables generally are much smaller than the other two types because of this overwriting strategy.

Factless Fact Tables

A **factless** fact table is a fact table that does not have any measures.

For example, think about a record of student attendance in classes. In this case, the fact table would consist of 3 dimensions: the student dimension, the time dimension, and the class dimension. This factless fact table would look like the following:

STUDENT_ID
CLASS_ID
TIME_ID

The only measure that you can possibly attach to each combination is "1" to show the presence of that particular combination. However, adding a fact that always shows 1 is redundant because we can simply use the COUNT function in SQL to answer the same questions.

Factless fact tables offer the most flexibility in data warehouse design. For example, one can easily answer the following questions with this factless fact table:

How many students attended a particular class on a particular day?

How many classes on average does a student attend on a given day?

Surrogate Keys

- Joins between fact and dimension tables should be based on surrogate keys
- Surrogate keys should not be composed of natural keys glued together
- Users should not obtain any information by looking at these keys
- These keys should be simple integers

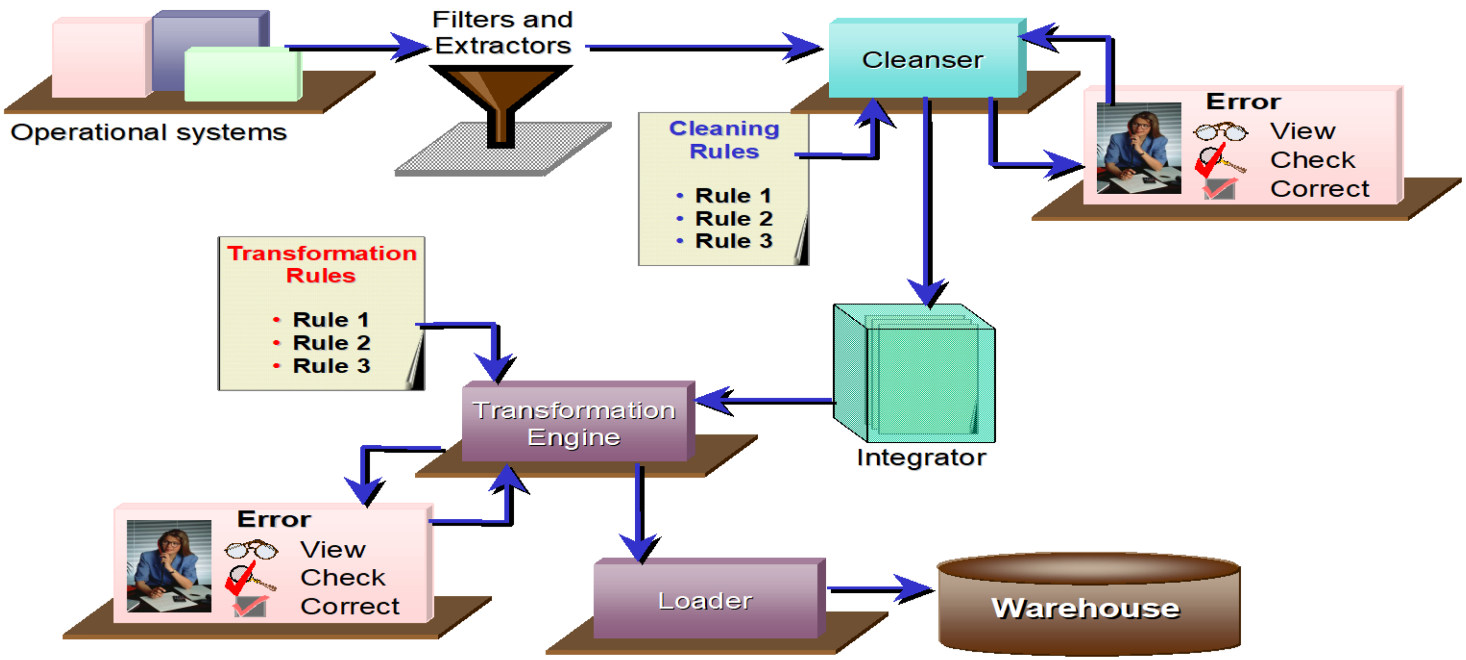
Why Existing Keys Should Not Be Used

- Keys may be reused after they have been purged even though they are used in the warehouse
- A product description or a customer description could be changed without changing the key
- Key formats may be generalized to handle some new situation
- A mistake could be made and a key could be reused

ETL- Extraction, Transformation & Loading

What is ETL?

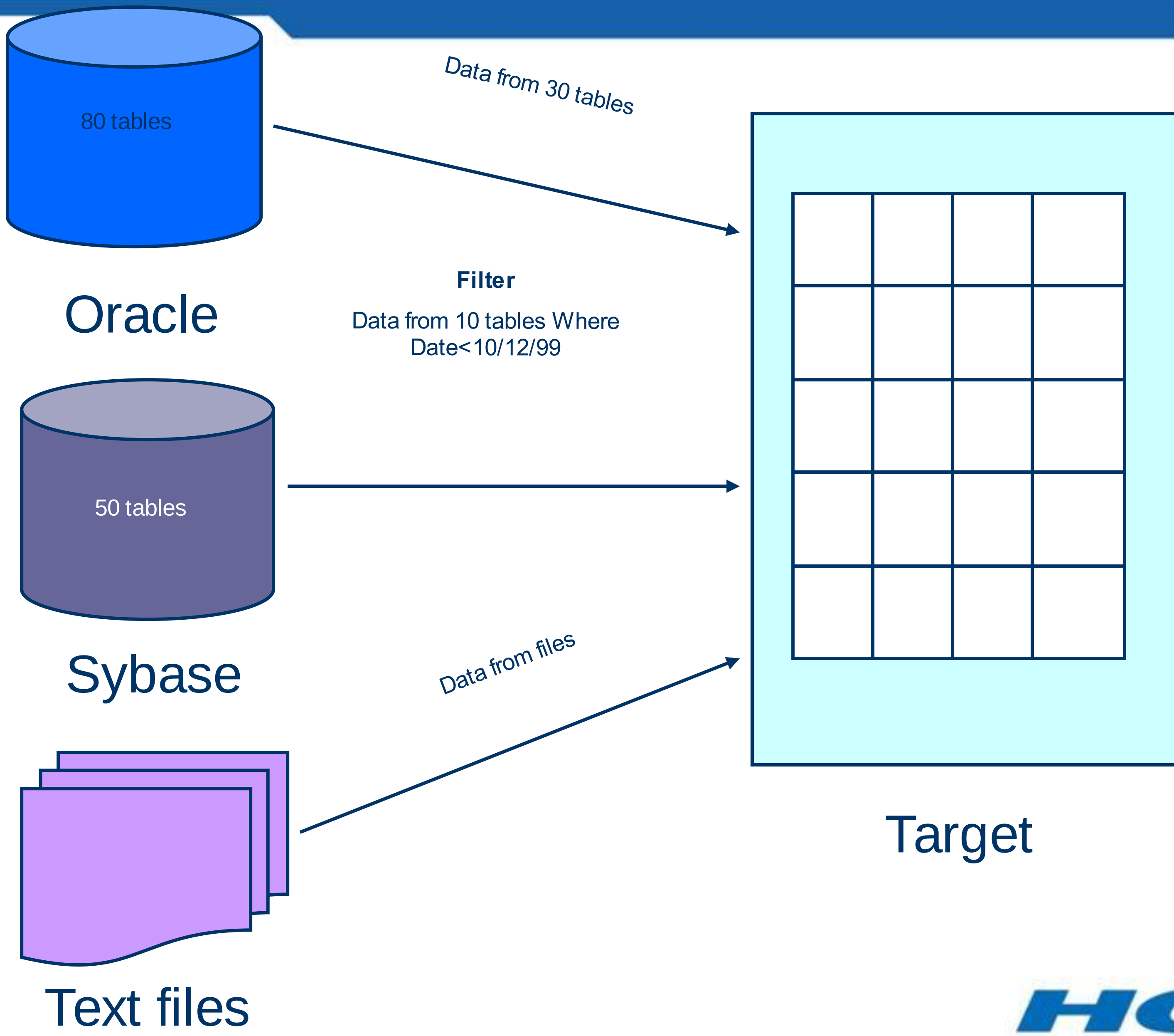
- ETL(Extraction, Transformation and Loading) is a process by which data is integrated and transformed from the operational systems into the Data warehouse environment



Operational Data - Challenges

- Data from heterogeneous sources
- Format differences
- Data Variations
  - Context
    - Across locations the same code could represent different customers
    - Across periods of time a product code could have been reused

Extraction


80 tables
Oracle
Data from 30 tables
Filter
Data from 10 tables Where Date<10/12/99
50 tables
Sybase
Data from files
Text files
Target

Transformation

Source

Emp id	Last Name	First Name
10001	Jones	Indiana
10002	Holmes	Sherlock

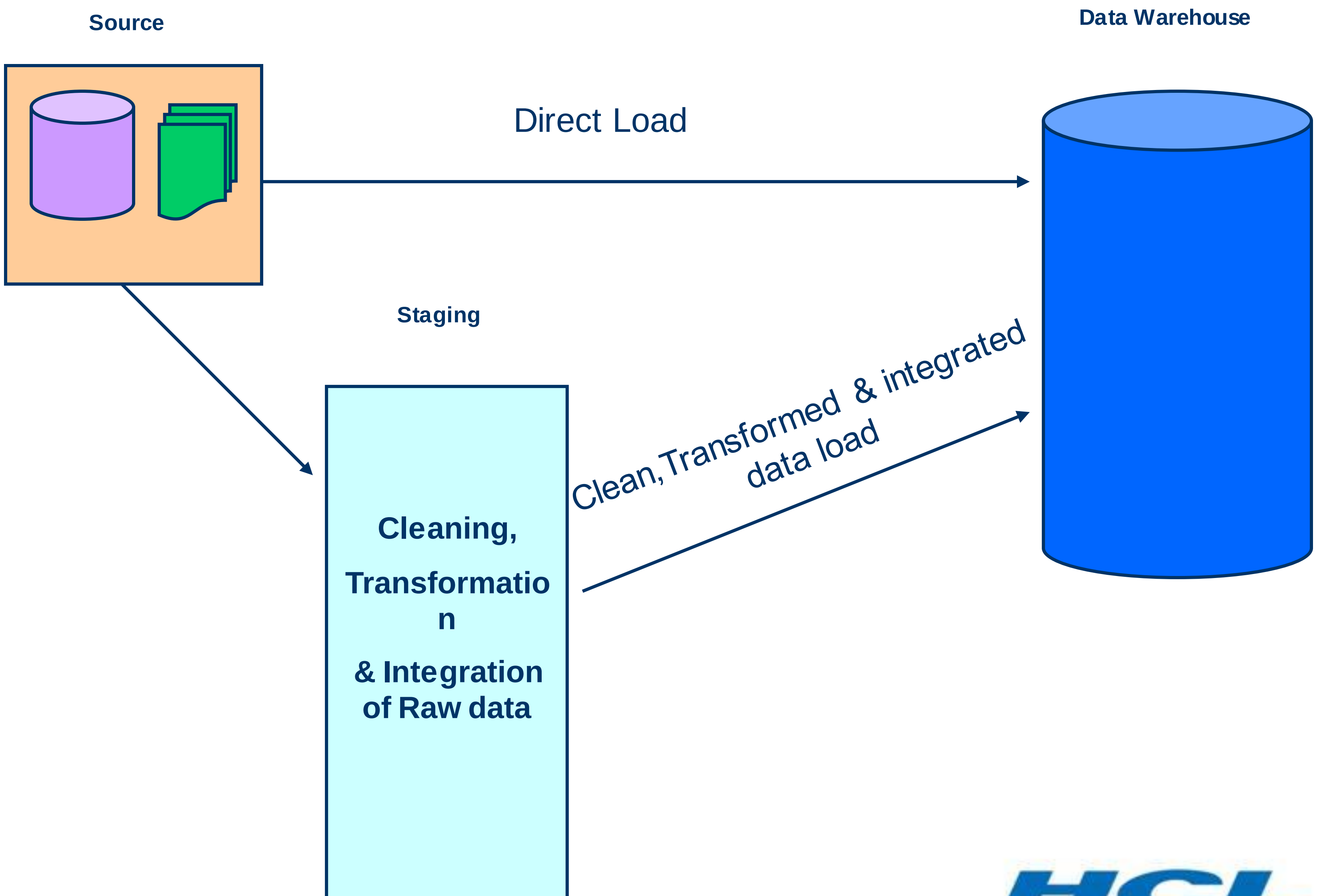
Name = Concat(First Name, Last Name)

Indiana Jones

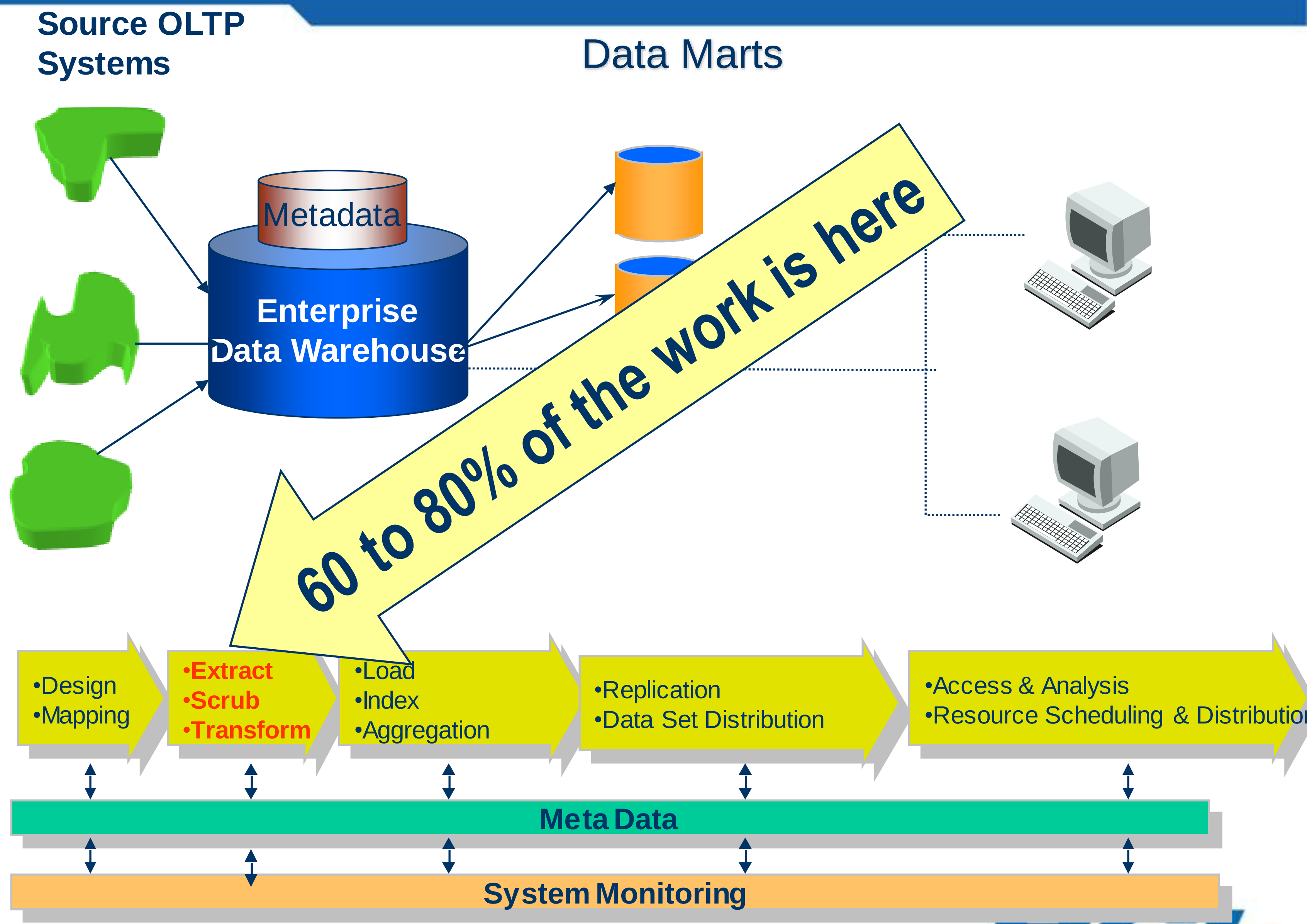
Sherlock Homes

Staging

Loading


Source
Data Warehouse
Direct Load
Staging
Clean,Transformed & integrated data load
Cleaning,
Transformation
& Integration of Raw data

Volume of ETL in a Data warehouse

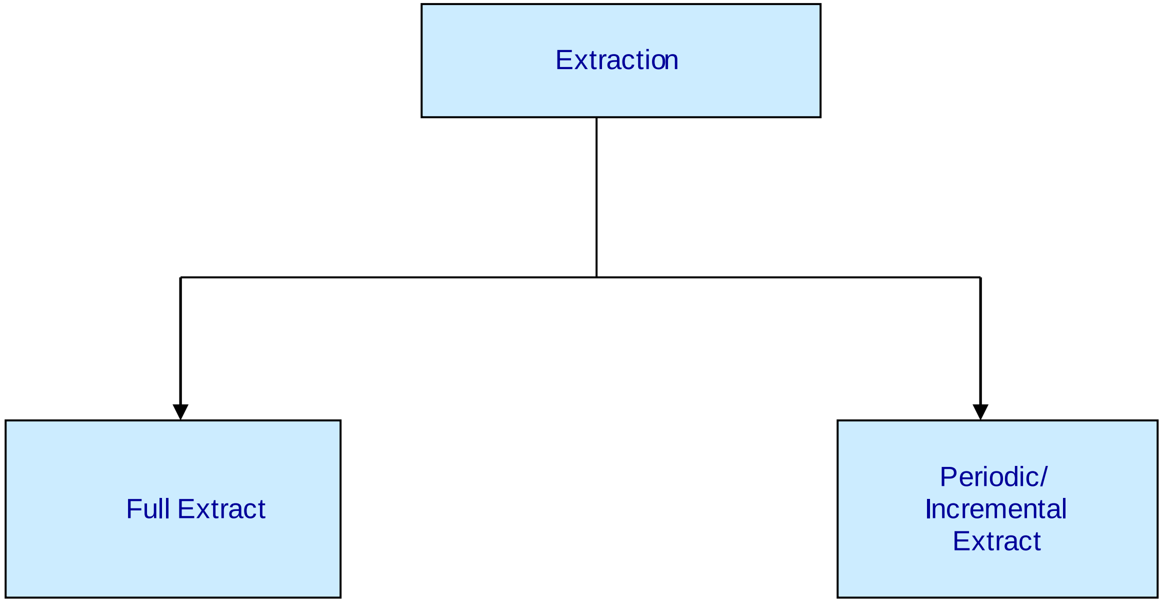


Factors Influencing ETL Architecture

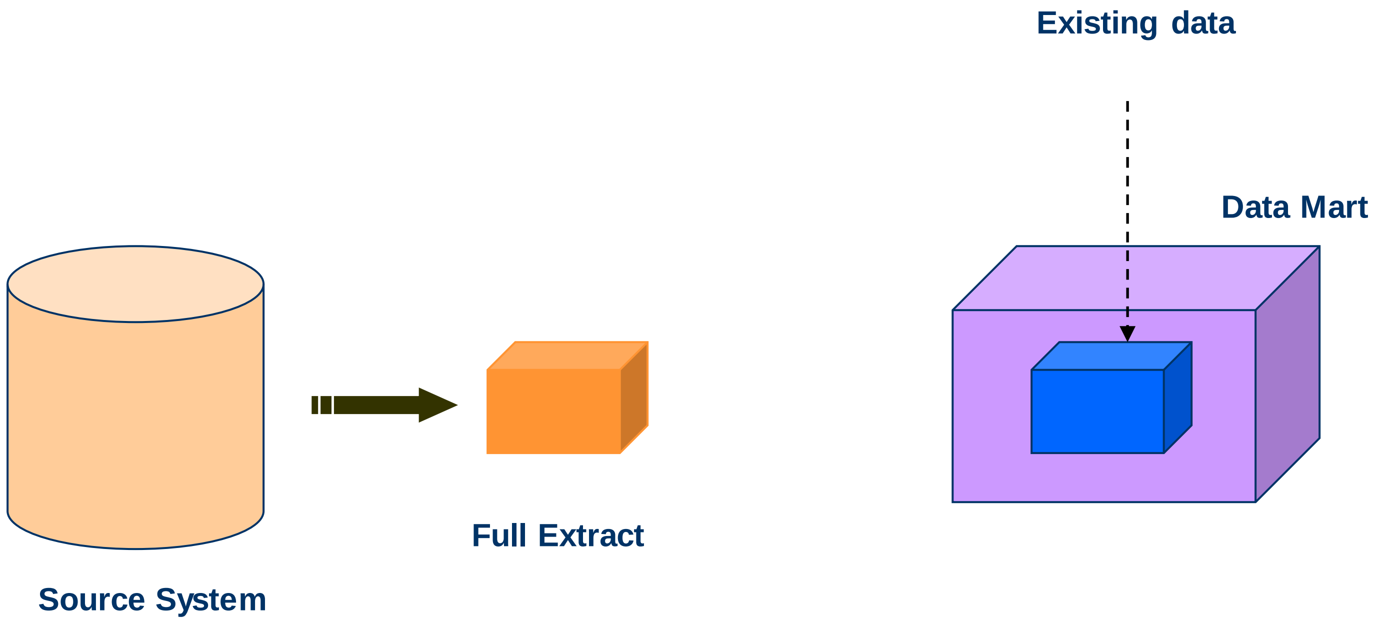
- Volume at each warehouse component.
- The time window available for extraction.
- The extraction type (Full,Periodic etc.)
- Complexity of the processes at each stage.

Extraction Types

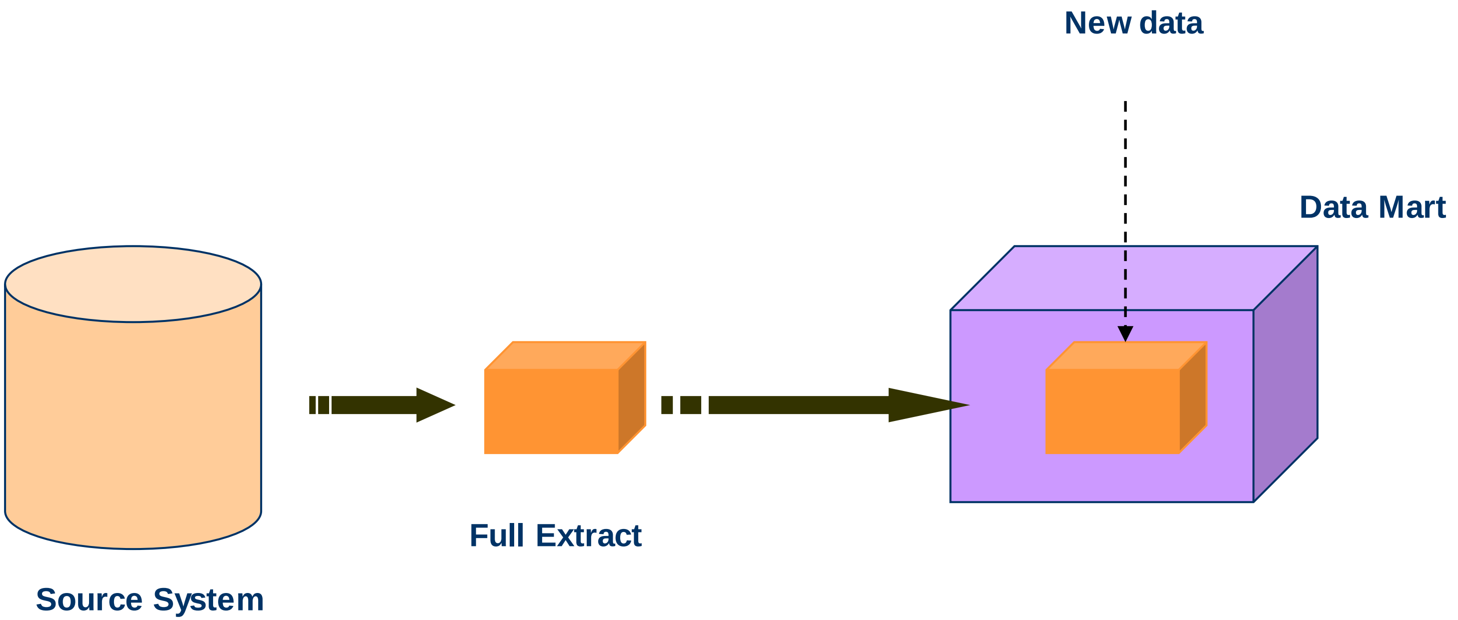
Extraction Types



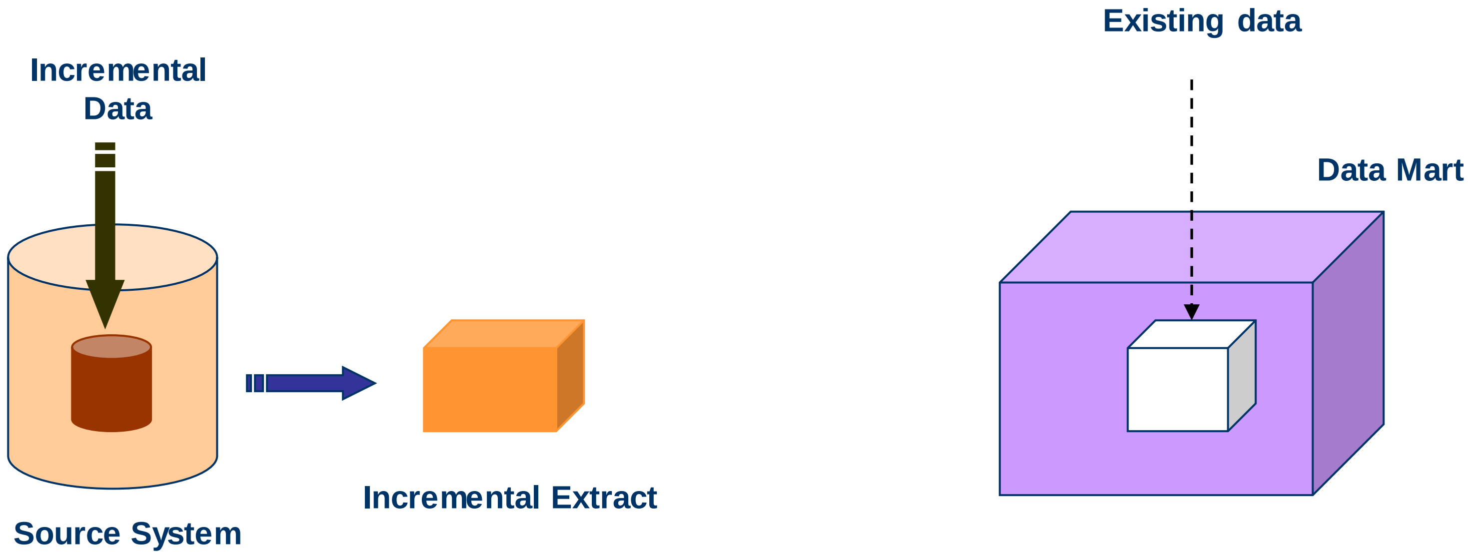
Full Extract



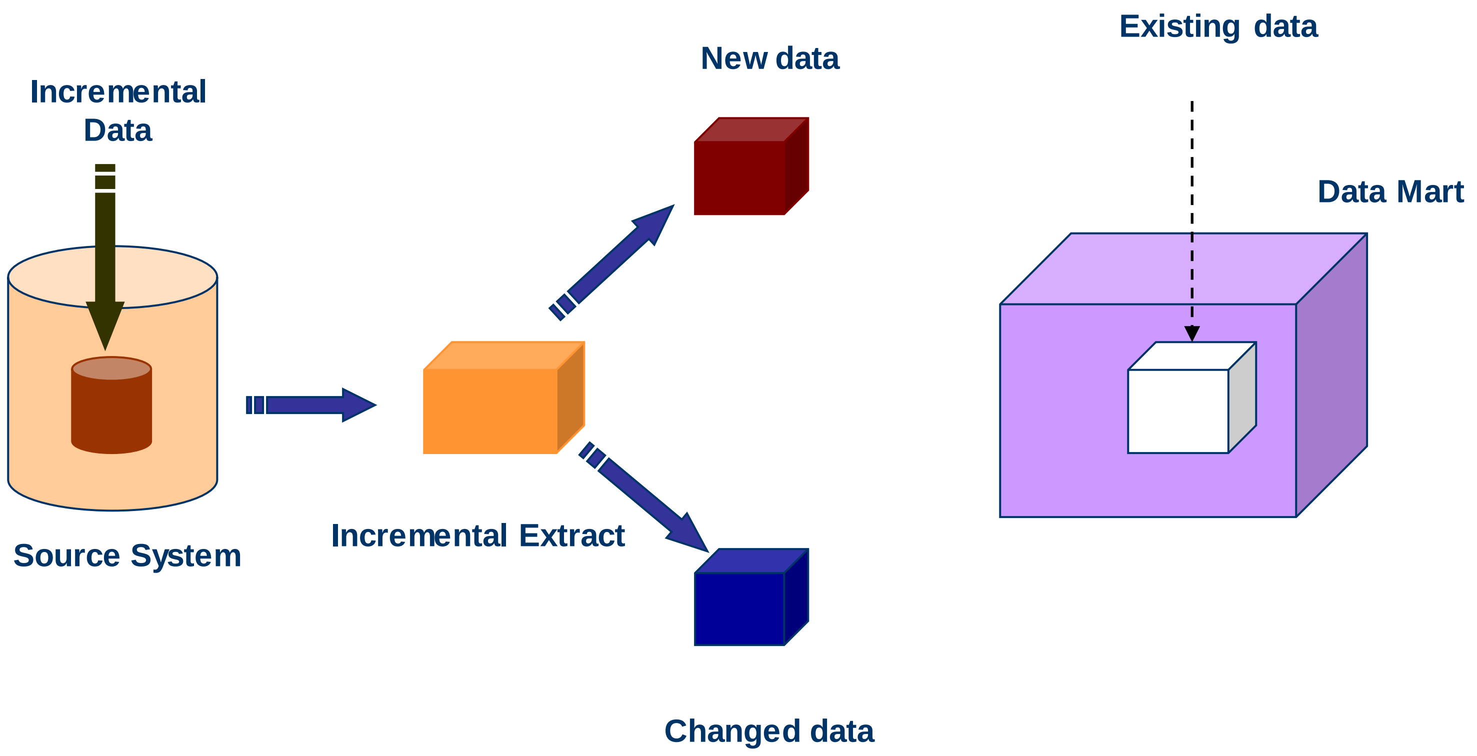
Full Extract



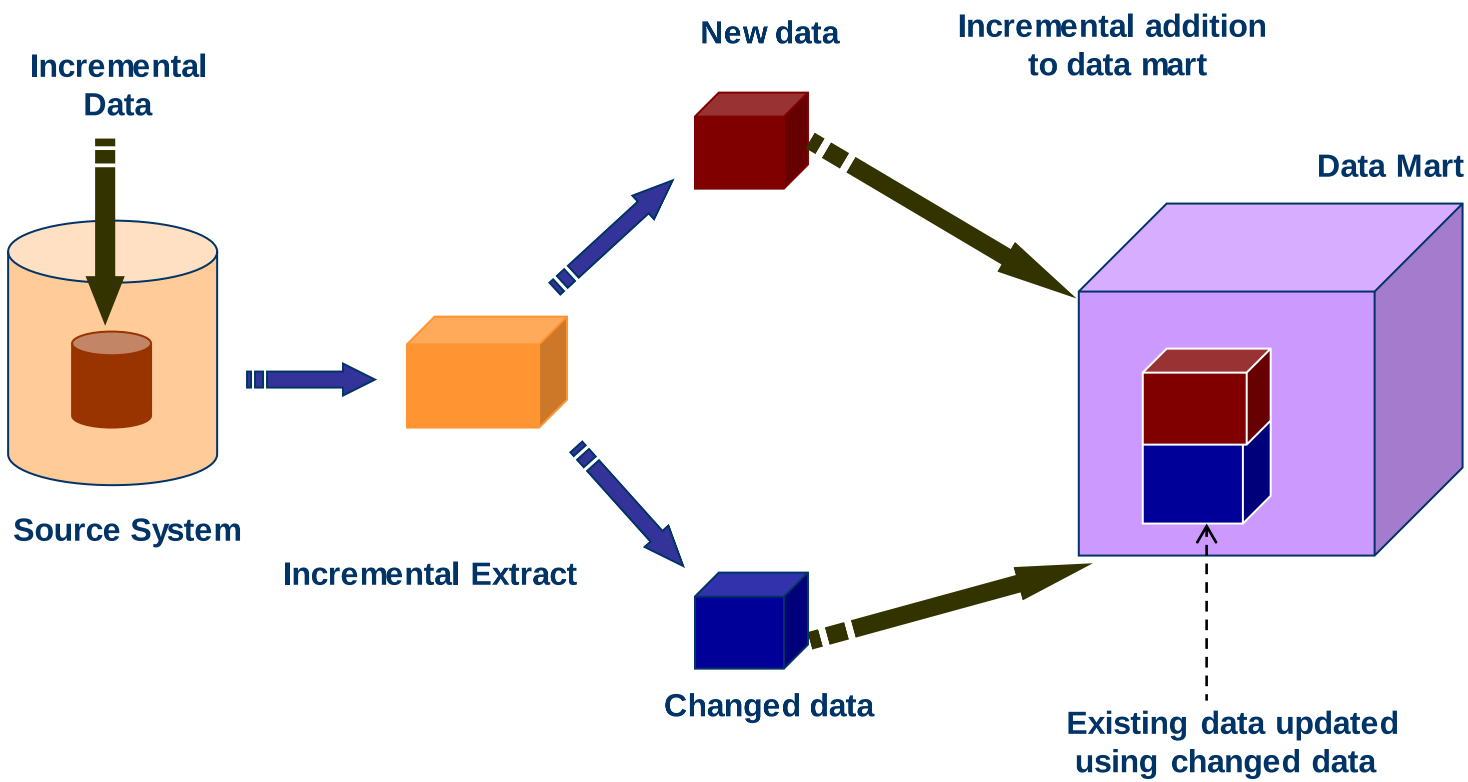
Incremental Extract



Incremental Extract



Incremental Extract



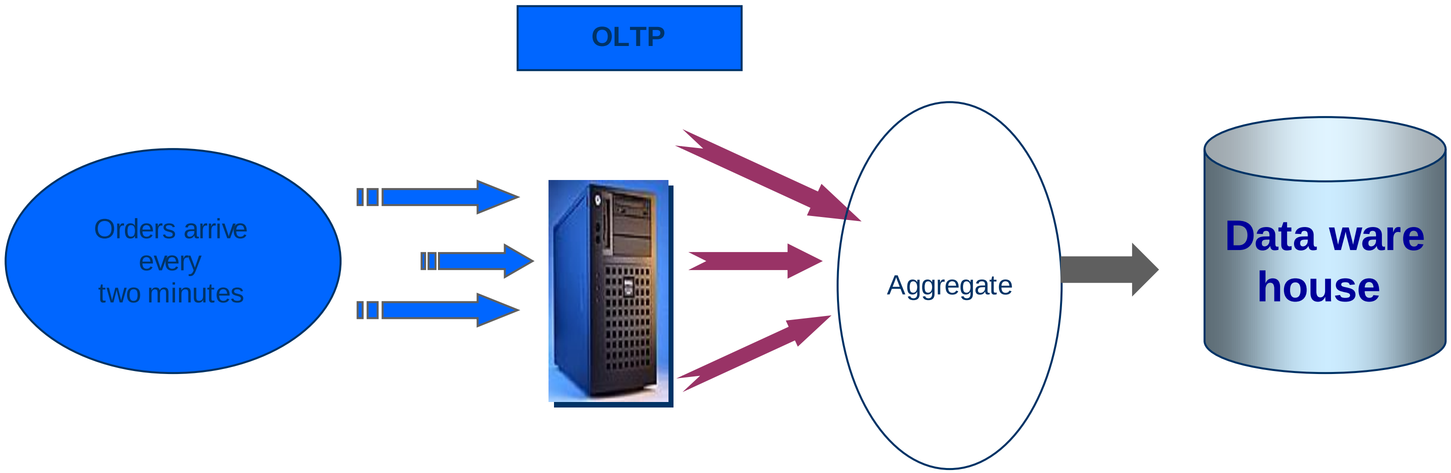
Transformation

Data Transformation

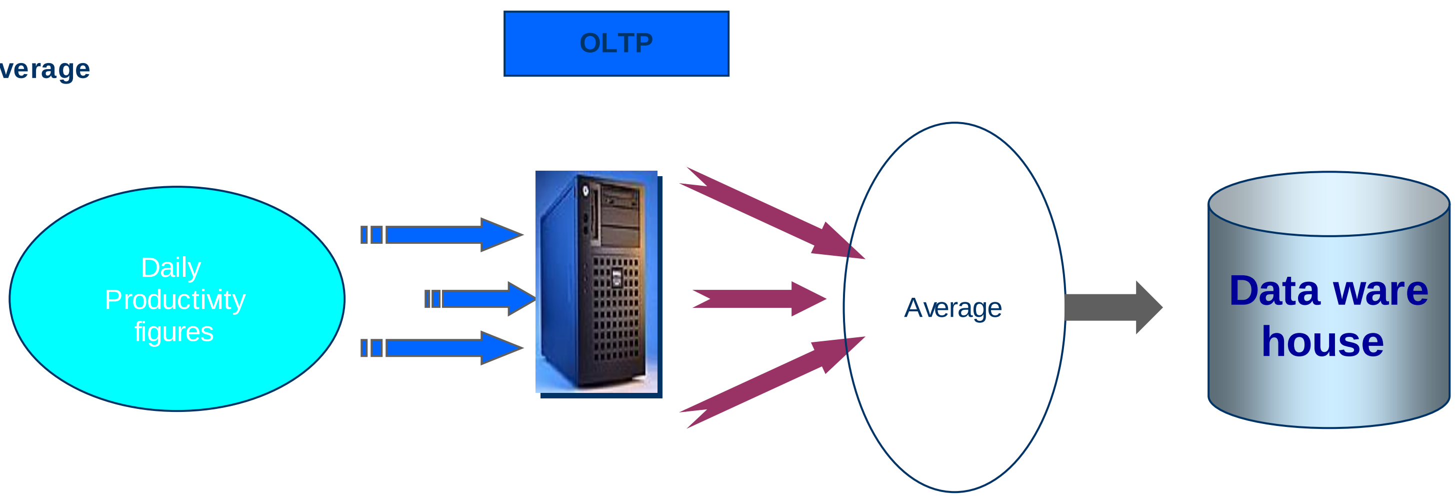
- Conversions
  - Data type (e.g. Char to Date)
  - Bring data to common units (Currency,Measuring Units)
- Classifications
  - Changing continuous values to discrete ranges (e.g. Temperatures to Temperature Ranges)
- Splitting of fields
- Merging of fields
- Aggregations (e.g. Sum, Avg., Count)
- Derivations (Percentages, Ratios, Indicators)

Structural Transformations

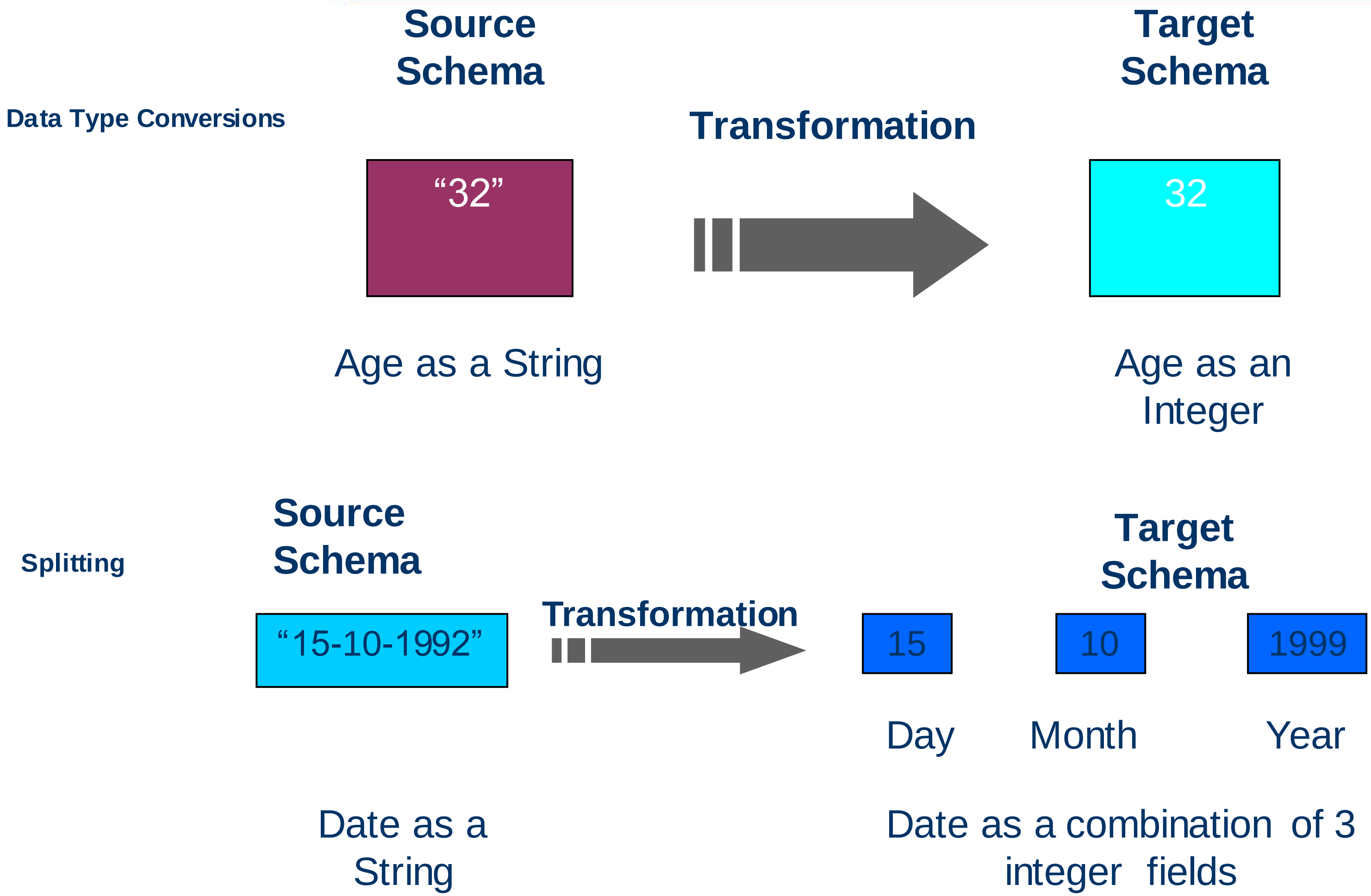
- Additive



Average



Format transformation



Simple Conversions

Source Schema | **Target Schema**

Rs. 10000 — **Multiply by 1/43** → $232.56

Revenue in Rupees → Revenue in Dollars

1000 lbs. — **Multiply by 0.4536** → 453.56

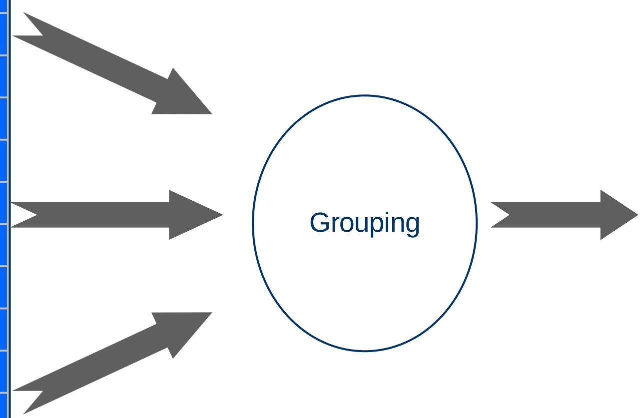
Production in Pounds → Production in Kilograms

Source Schema | **Target Schema**

- Transformations using Simple Conversions

Classification

Name	Age
John Black	27
Richard Wayne	53
Jennifer Goldman	45
Helmut Koch	37
Anna Ludwig	32
Shito Maketha	28
Tracy Withman	39
Ada Zhesky	25
David Rosenberg	33
Pankaj Sharma	29
Zhu Ling	44
George Kurtz	27
Rita Hartman	34



Age Group	Frequency
20-25	1
26-30	4
31-35	3
36-40	2
41-45	2
46-50	1
51-55	1
56-60	0

Data Consistency Transformations

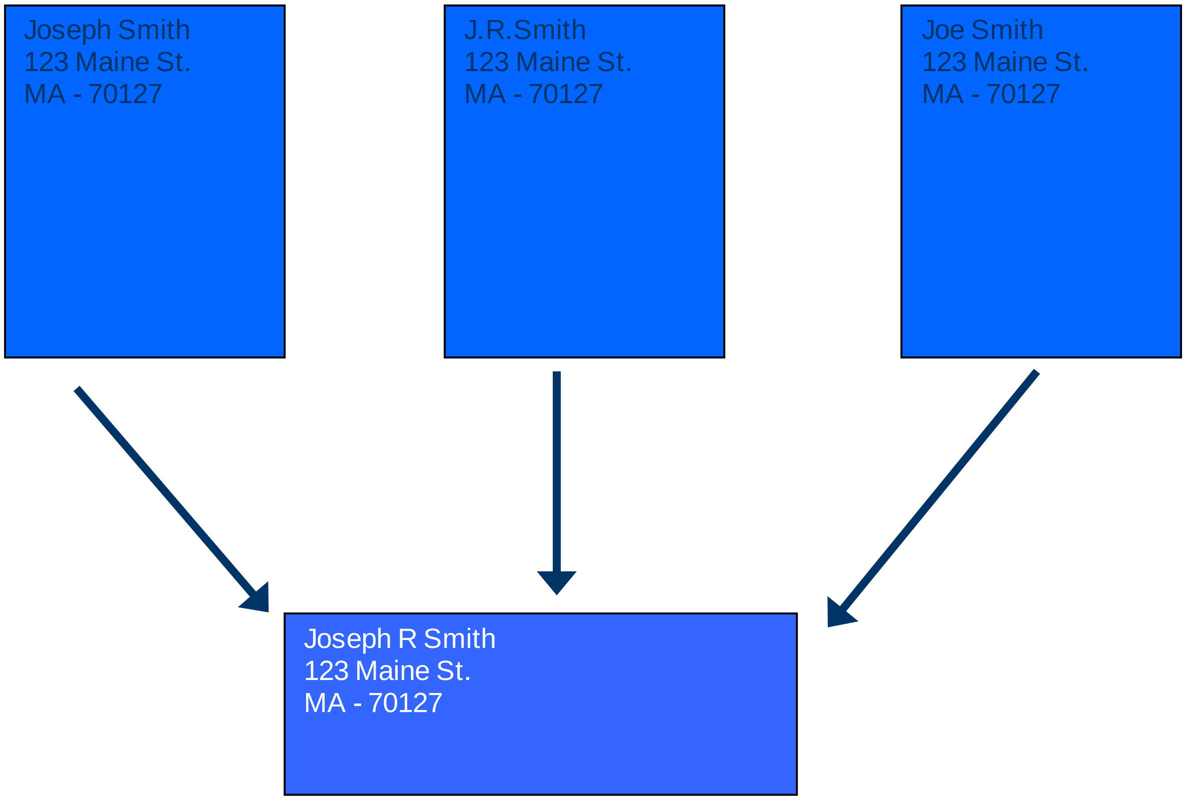
Source 1
Gender
Male – M
Female – F

Source 2
Gender
Male – Male
Female – Female

Source 1
Gender
Male – 1
Female – 2

Target
Gender
Male – M
Female – F

Reconciliation of Duplicated data

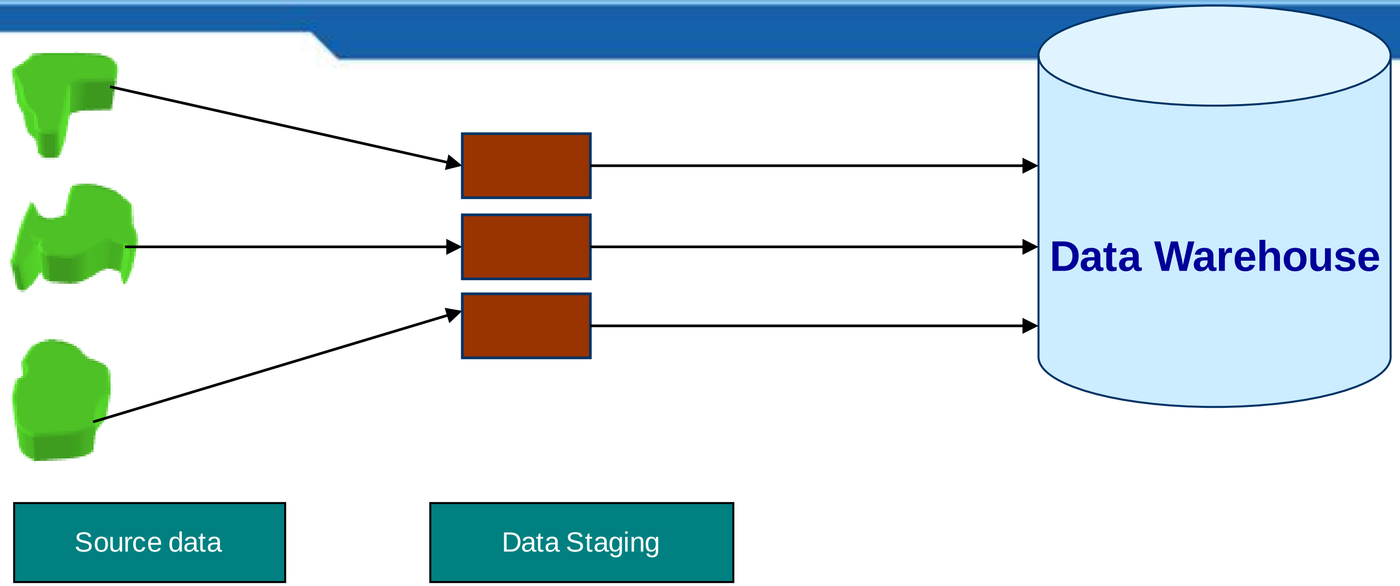


Loading

Types of Data warehouse Loading

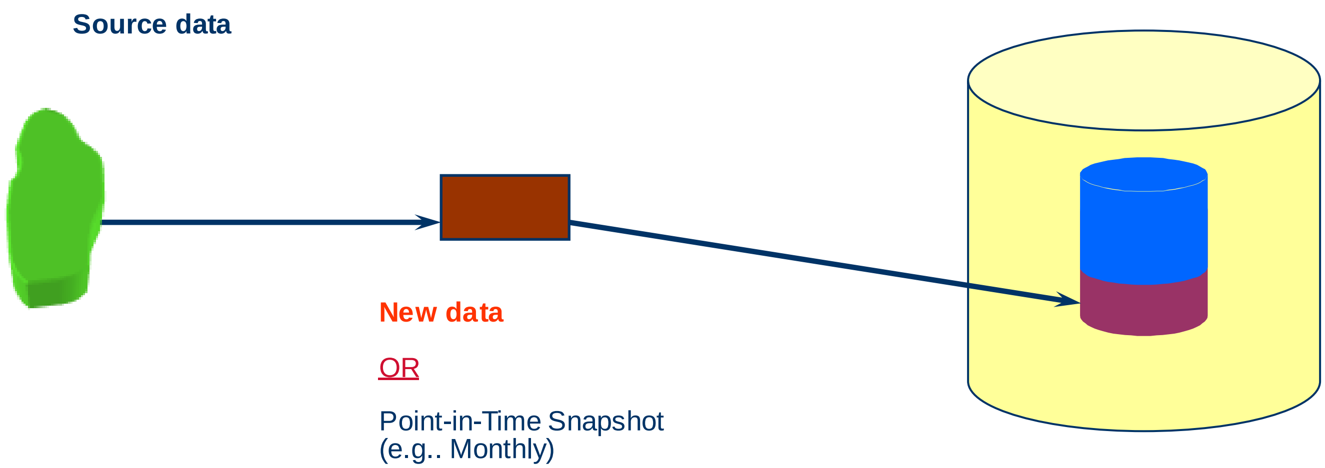
- Target update types
  - Insert
  - Update

Types of Data Warehouse Updates

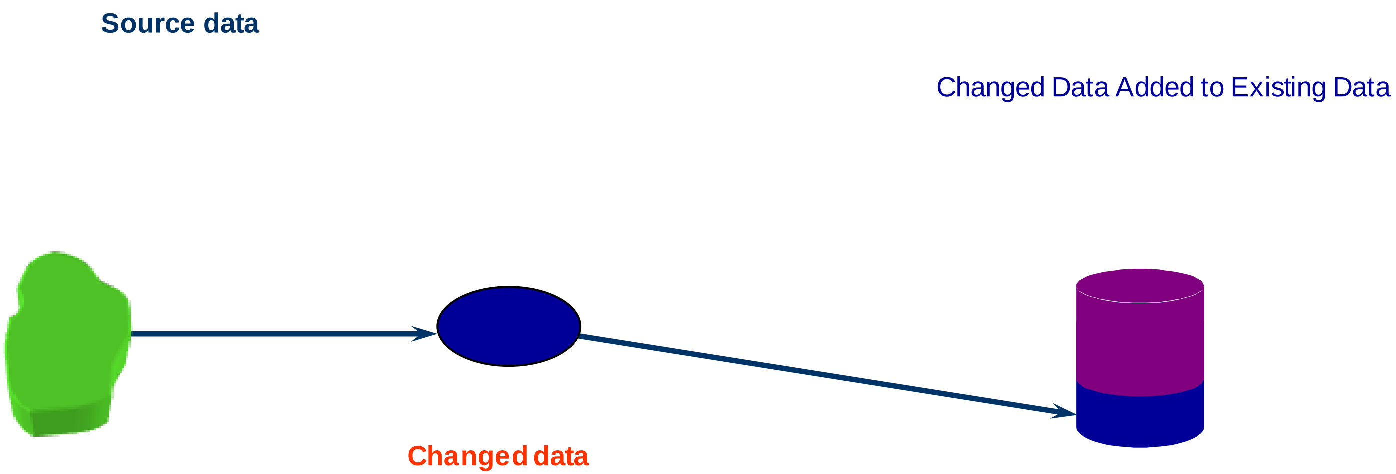


- Insert
- Full Replace
- Selective Replace
- Update
- Update plus Retain History

New Data and Point-In-Time Data Insert



Changed Data Insert



Change of Dimension values

When the value of dimension in a data warehouse changes,

then

- History of change needs to be maintained.
- Changed data alone needs to be identified
- Changed data should be easier to access.
- Reconstruction of the dimension table any point in time should be easier

ETL - Approach in a nutshell

1) Identify the Operational systems based on data islands in the

   target
2) Map source-target dependencies.
3) Define cleaning and transformation rules
4) Validate source-target mapping
5) Consolidate Meta data for ETL
6) Draw the ETL architecture
7) Build the cleaning, transformation and auditing routines using either a tool or customized programs

Meta Data in a Datawarehouse

What is Metadata?

- Data about data and the processes
- Metadata is stored in a data dictionary and repository.
- Insulates the data warehouse from changes in the schema of operational systems.
- It serves to identify the contents and location of data in the data warehouse

Why Do You Need Meta Data?

- Share resources
  - Users
  - Tools
- Document system

- Without meta data
  - Not Sustainable
  - Not able to fully utilize resource

The Role of Meta Data in the Data Warehouse

Meta Data enables data to become information, because with it you

- <u>Know</u> what data you have

  <u>and</u>

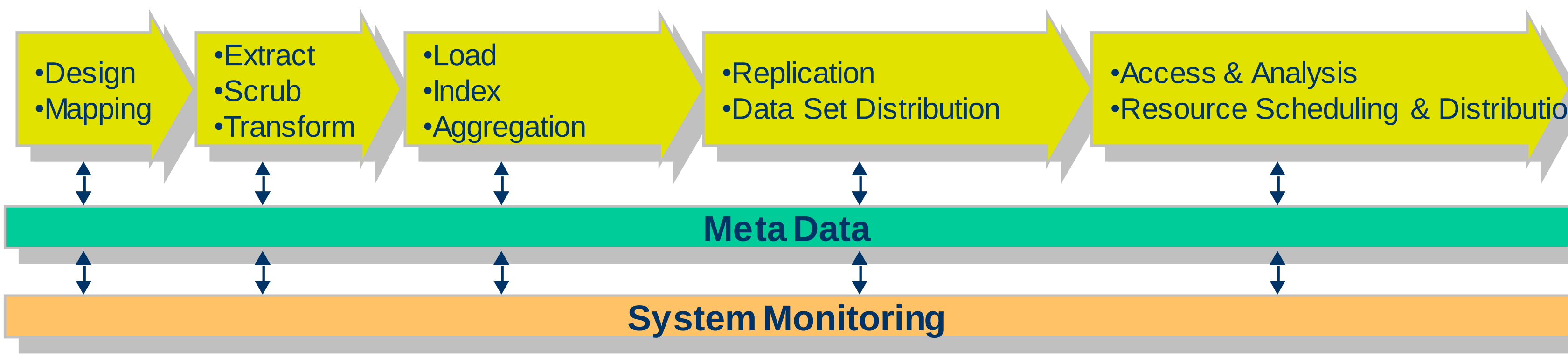
- You can <u>trust</u> it!

Meta Data Answers….

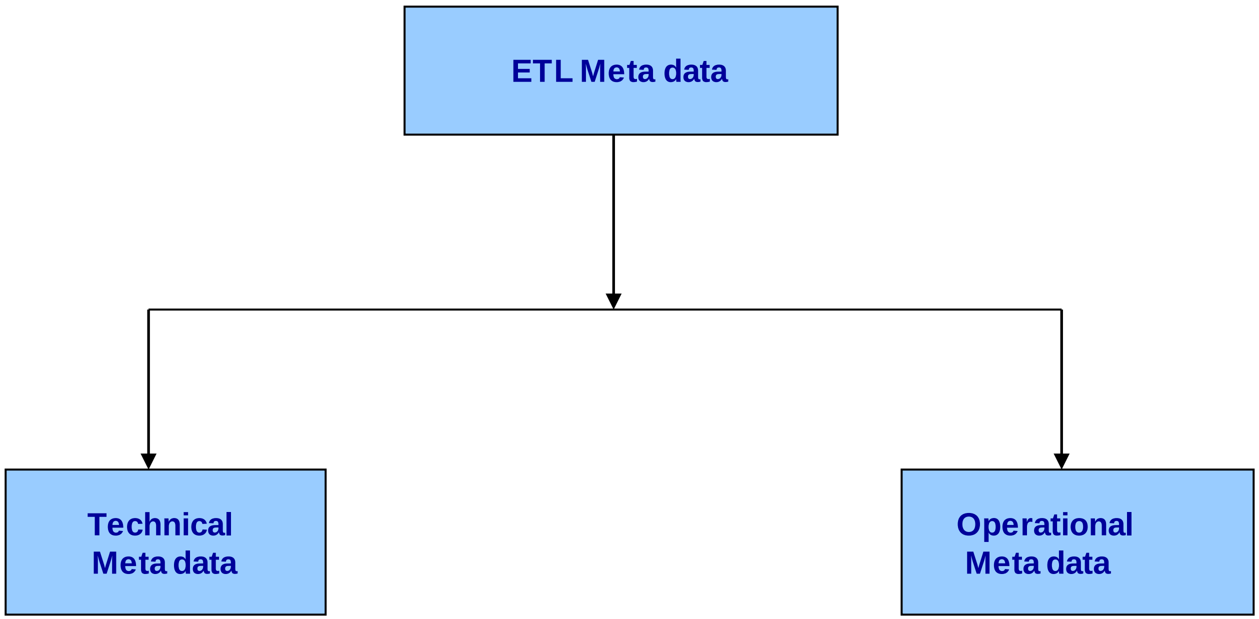
* How have business definitions and terms changed over time?
* How do product lines vary across organizations?
* What business assumptions have been made?
* How do I find the data I need?
* What is the original source of the data?
* How was this summarization created?
* What queries are available to access the data

Meta Data Process

- Integrated with entire process and data flow
  - Populated from beginning to end
  - Begin population at design phase of project
  - Dedicated resources throughout
    - Build
    - Maintain



Types of ETL Meta Data



.

Classification of ETL Meta Data

- Data Warehouse Meta data

  This Meta data stores descriptive information about the physical implementation details of data warehouse.

- Source Meta data

  This Meta data stores information about the source data and the mapping of source data to data warehouse data

ETL Meta Data

- Transformations & Integrations.

  This Meta data describes comprehensive information about the Transformation and loading.

- Processing Information

  This Meta data stores information about the activities involved in the processing of data such as scheduling and archives etc

- End User Information

  This Meta data records information about the user profile and security.

ETL - Planning for the Movement

The following may be helpful for planning the movement

- Develop a ETL plan
- Specifications
- Implementation

Data Warehouse Administration

Data Warehouse Administrative Tasks

- Build and maintain the data warehouse
- Maintaining the meta data
- To keep the data warehouse up to date
- Tuning the data warehouse
- General administrative tasks

Dormant Data

- The data that is hardly used in a data warehouse is called dormant data

- The faster data warehouses grows the more data becomes dormant. Over a period of time the amount of dormant data in a data warehouse increases

Origins of Dormant Data

- Storing history data that is not required
- Storing columns that are never used
- Storing detail level data when only summary level data is used
- Creating summary data that is never used

Strategy For Removing Dormant Data

The strategy for removing dormant data might include:

- `Removing data after a period of time say after two years
- Removing summary data that has not been accessed in the past six months
- Removing columns that have never or only very infrequently been accessed
- Storing data for high profile users even though that data has not been accessed
- Storing data for selected accounts even though that data has not been accessed

Tuning a Data Warehouse

Some of the techniques that can be used for tuning a data warehouse are:

- Handling dormant data
- Storing pre summarized data based on data pattern usage
- Creating indexes for data that is frequently used
- Merging tables that have common and regular access

Feedback

If you have any suggestion regarding the presentation or training, want to suggest inclusion of any topic etc., please mail to debadatta.mohanty@hcl.com