DATA WAREHOUSES - part2

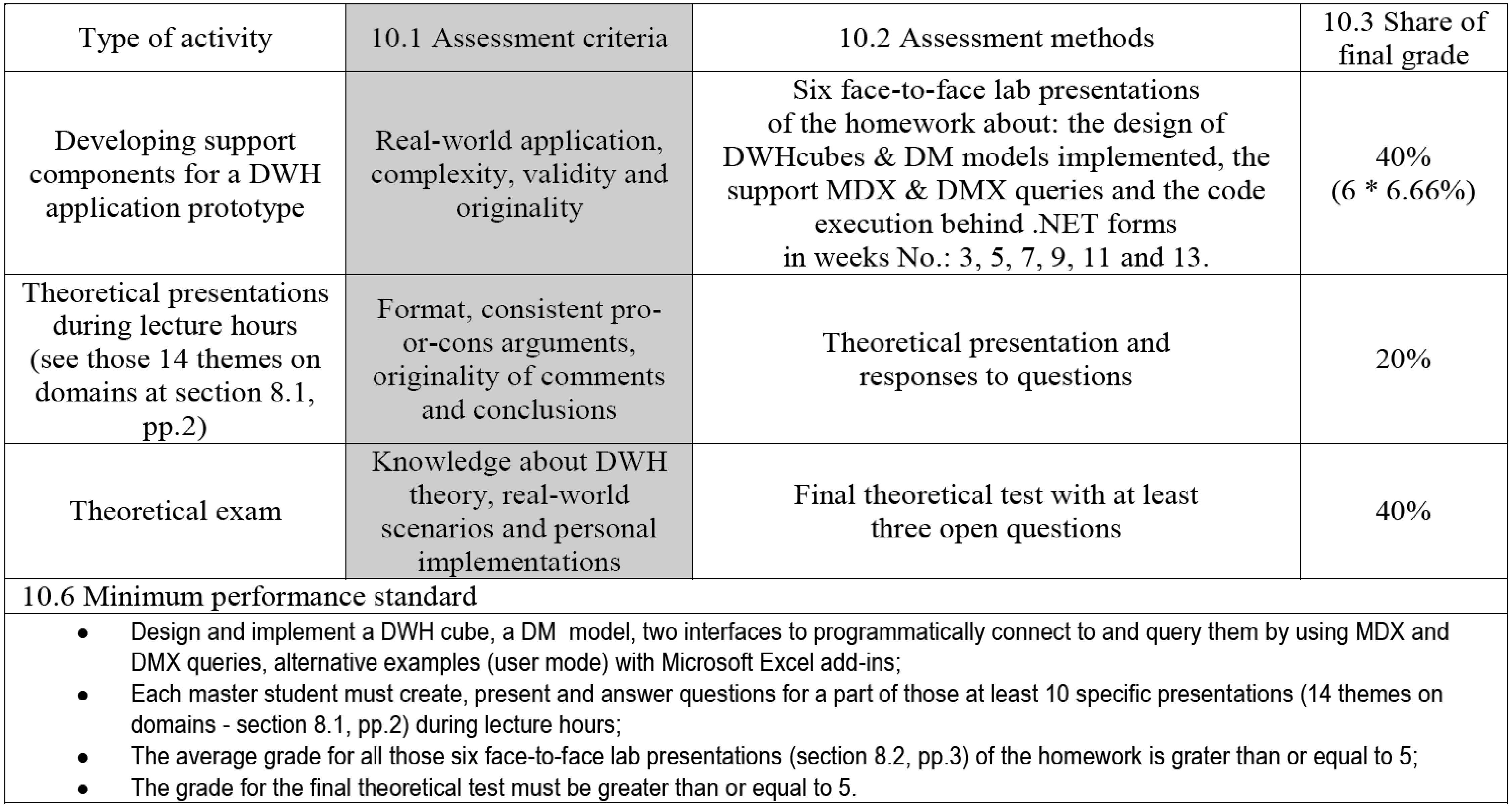
1. **Introduction to the DWH discipline**
2. **Brief history** (Inmon &Linstedt,2015) **and Data Architecture** (Kimball&Ross,2016)
3. **Dimensional Modeling Fundamentals** (Kimball&Ross,2016)
4. **Technical Architecture Considerations** (Kimball&Ross,2016)
5. **Extract Transform Load and Data Quality** (Kimball&Ross,2016)
6. **DWH Lifecycle** (Kimball&Ross,2013)
7. **Trends in the evolution of DWH:**

   Extended RDBMS Architecture (Kimball&Ross,2013)**.**

   Pushing into the Future (Reeves,2009)**.**

   DWH 1.0 vs. 2.0 (Krishnan,2013)

Assessment



Type of activity	10.1 Assessment criteria	10.2 Assessment methods	10.3 Share of final grade
Developing support components for a DWH application prototype	Real-world application, complexity, validity and originality	Six face-to-face lab presentations of the homework about: the design of DWHcubes & DM models implemented, the support MDX & DMX queries and the code execution behind .NET forms in weeks No.: 3, 5, 7, 9, 11 and 13.	40% (6 * 6.66%)
Theoretical presentations during lecture hours (see those 14 themes on domains at section 8.1, pp.2)	Format, consistent pro-or-cons arguments, originality of comments and conclusions	Theoretical presentation and responses to questions	20%
Theoretical exam	Knowledge about DWH theory, real-world scenarios and personal implementations	Final theoretical test with at least three open questions	40%

10.6 Minimum performance standard

- Design and implement a DWH cube, a DM model, two interfaces to programmatically connect to and query them by using MDX and DMX queries, alternative examples (user mode) with Microsoft Excel add-ins;
- Each master student must create, present and answer questions for a part of those at least 10 specific presentations (14 themes on domains - section 8.1, pp.2) during lecture hours;
- The average grade for all those six face-to-face lab presentations (section 8.2, pp.3) of the homework is grater than or equal to 5;
- The grade for the final theoretical test must be greater than or equal to 5.

Planning those at least 10 presentations (course hours, 20% in final grades) of master students (teams) about:

(1.) retail sales, (2.) inventory, (3.) procurement, (4.) order management, (5.) accounting, (6.) CRM, (7.) HRM,
(8.) financial services, (9.) telecommunications,
(10.) transportation, (11.) education, (12.) healthcare,
(13.) e-commerce, (14.) insurance

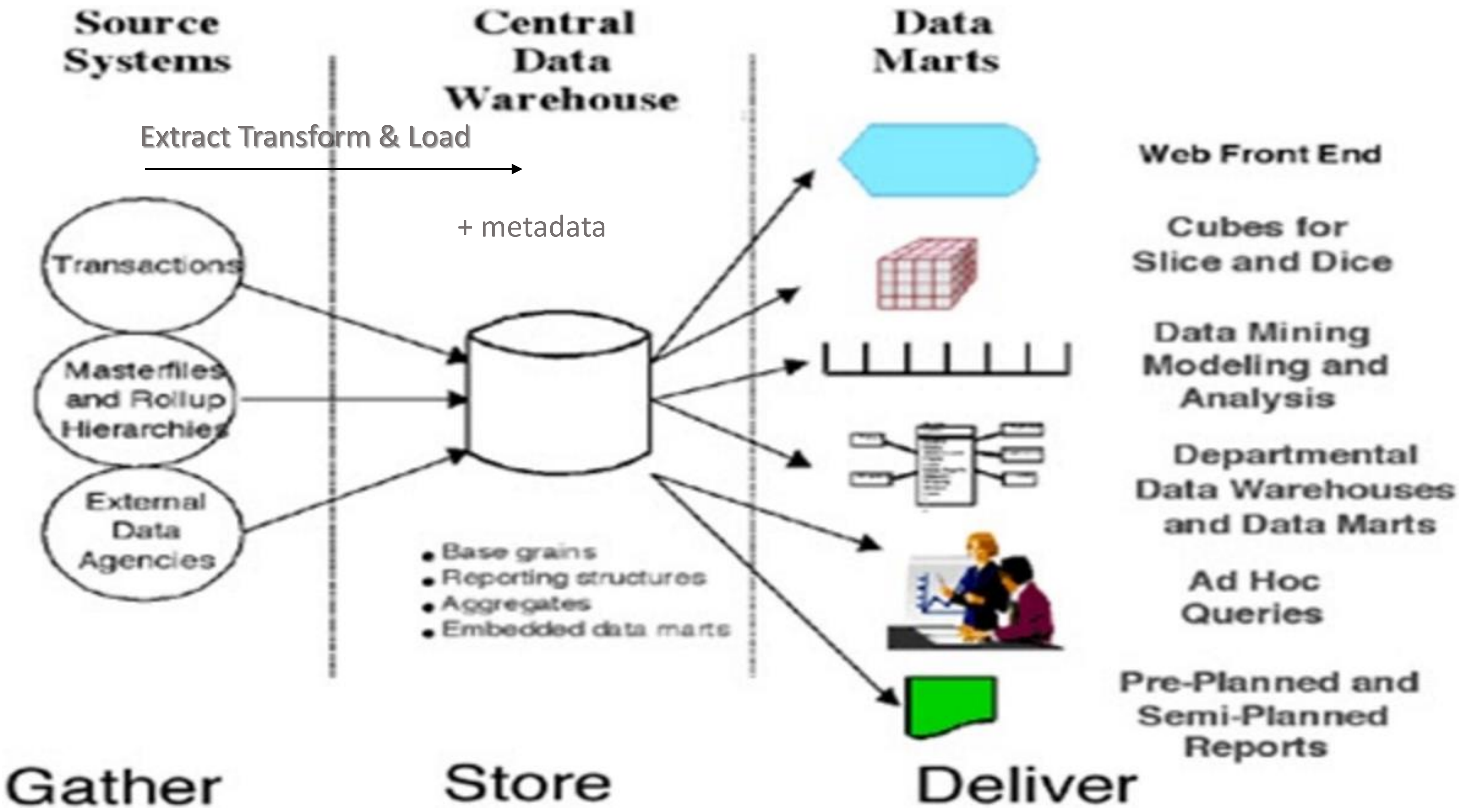
details: **Kimball&Ross,2013**.

by **e-mail**: dan.homocianu@gmail.com

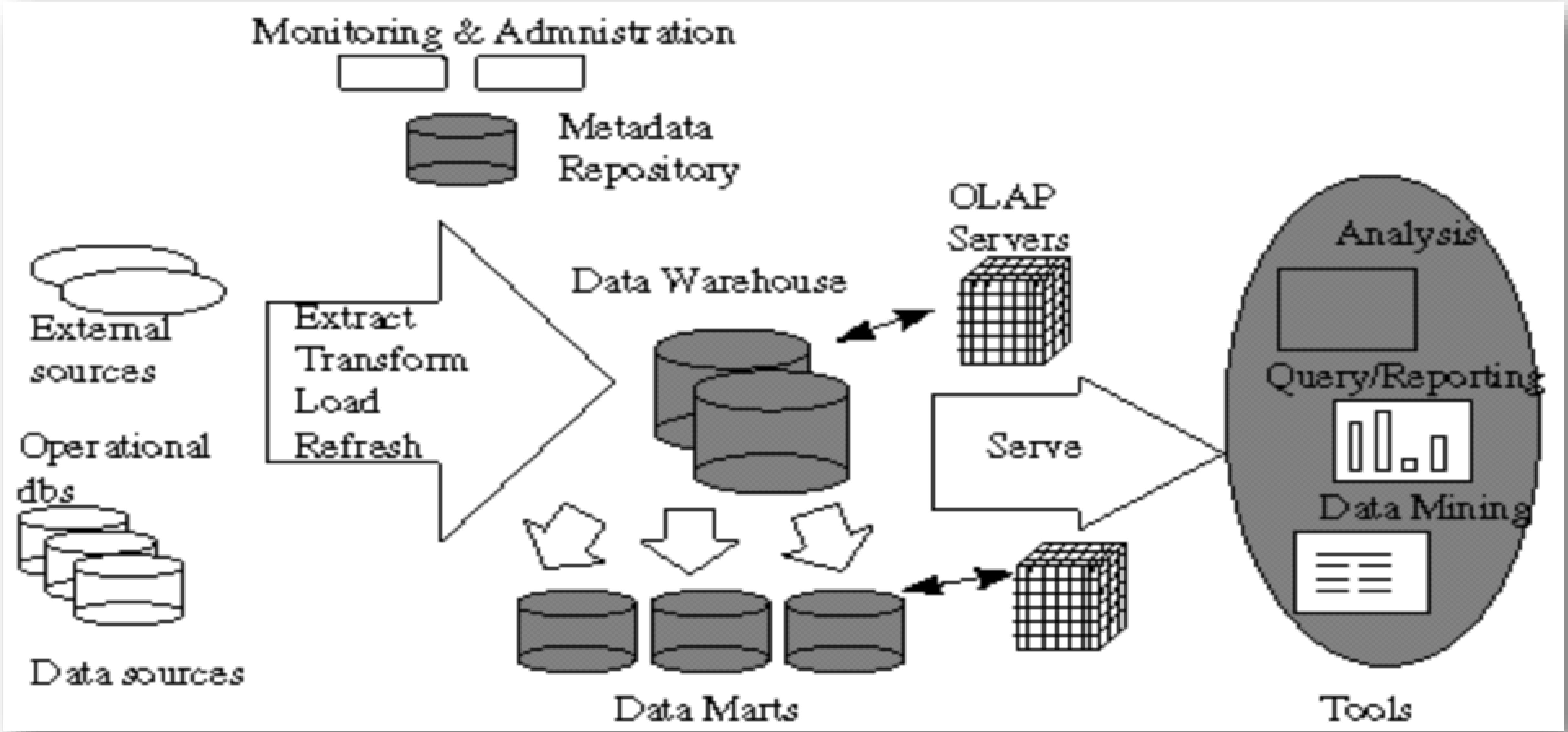
(**subject**: *DWH course pres. theme TITLE.. on DATE..*
content: *team: MEMBER1*, *MEMBER2*, *MEMBER3*)

Each master student (team member) must present a part of his team's support presentation for at least 5-10 minutes and answer questions.

DWH – overview



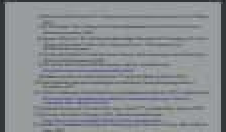
DWH – overview



http://redbook.cs.berkeley.edu/redbook3/lec28.html

29

30



architectures are described in the next sections.

1.4.1. Typical Two-Layer Architecture

Kimball has introduced an often-used, two-layer architecture [24, p114]. In this architecture, which is presented in Figure 1.3, there are only two layers that are part of the data warehouse system itself.

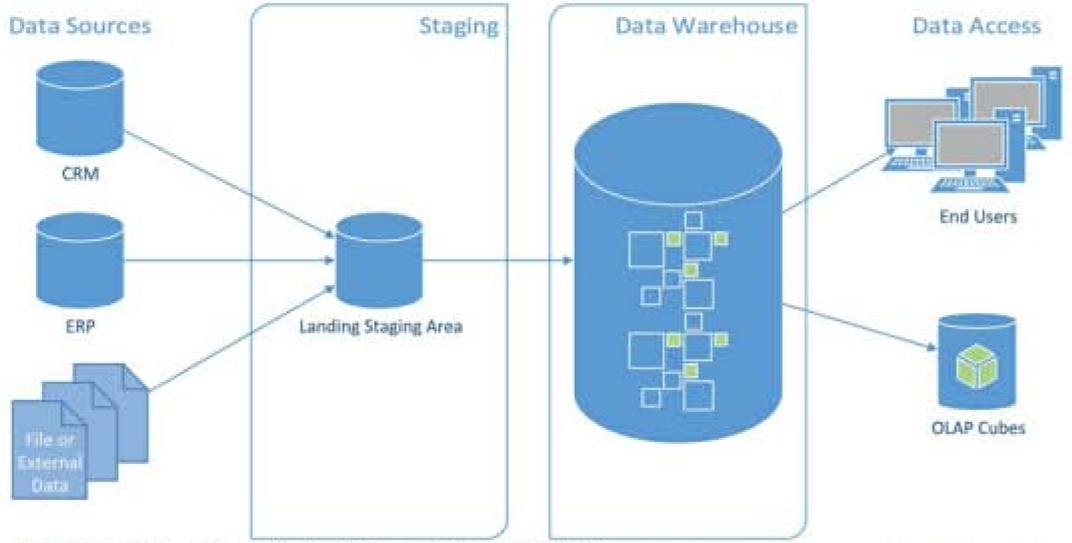


FIGURE 1.3 The Kimball Data Lifecycle [25]. Figure adapted by author from [25]. Copyright by IAS Inc. Reprinted with permission.

29

30

31

1.4.2. Typical Three-Layer Architecture

To overcome the limitations of a two-layer architecture, another commonly found architecture is based on three layers (Figure 1.4).

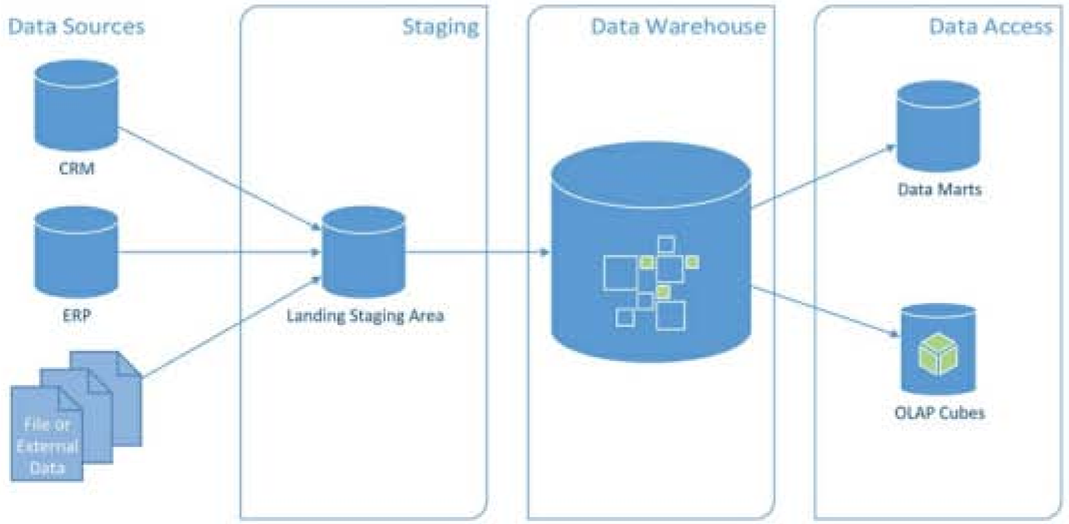
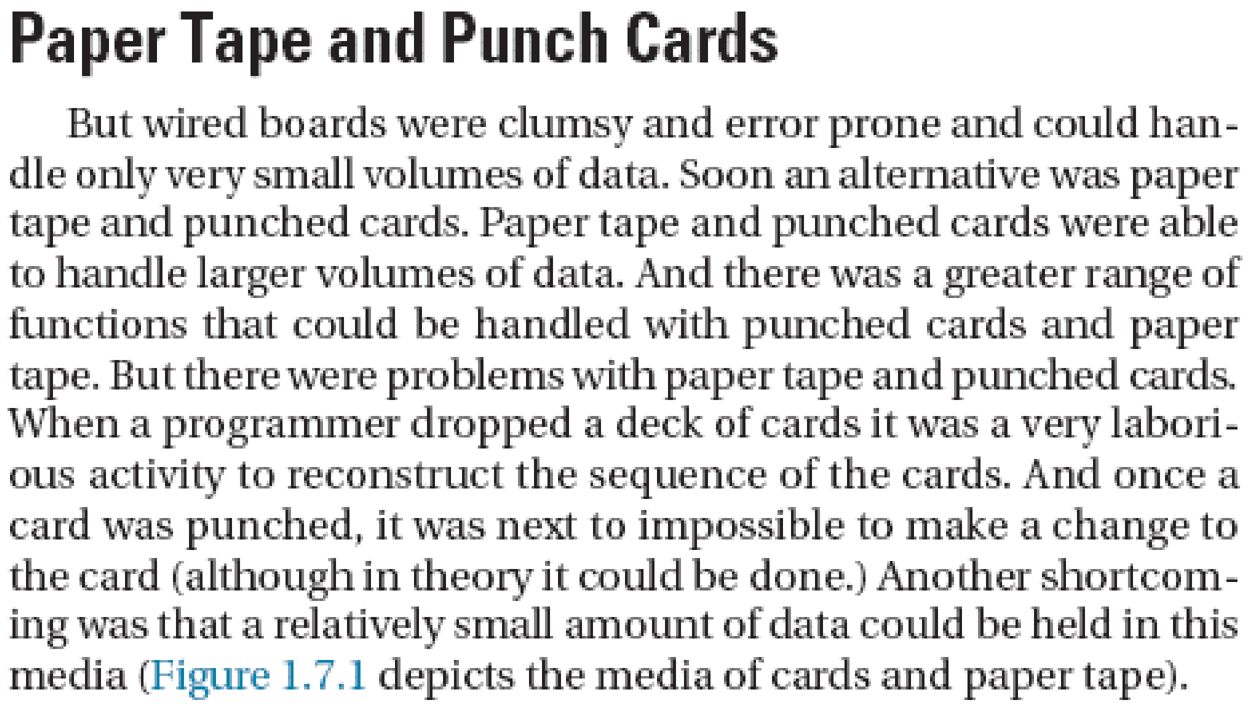


FIGURE 1.4 The Inmon Data Warehouse [25]. Figure adapted by author from [25]. Copyright by IAS Inc. Reprinted w ith permission.

This architecture has been introduced by Inmon and introduces an atomic data warehouse, often a normalized operational data store (ODS) between the staging area and the dimensional model. The stage area in this architecture follows that of the two-layer architecture. The data warehouse, however, holds raw data modeled in a third-normal form. It integrates all data of the enterprise, but is still based on physical tables from the source systems. By doing so, it acts similarly to a large operational database.

Brief History of data

Inmon &Linstedt,2015



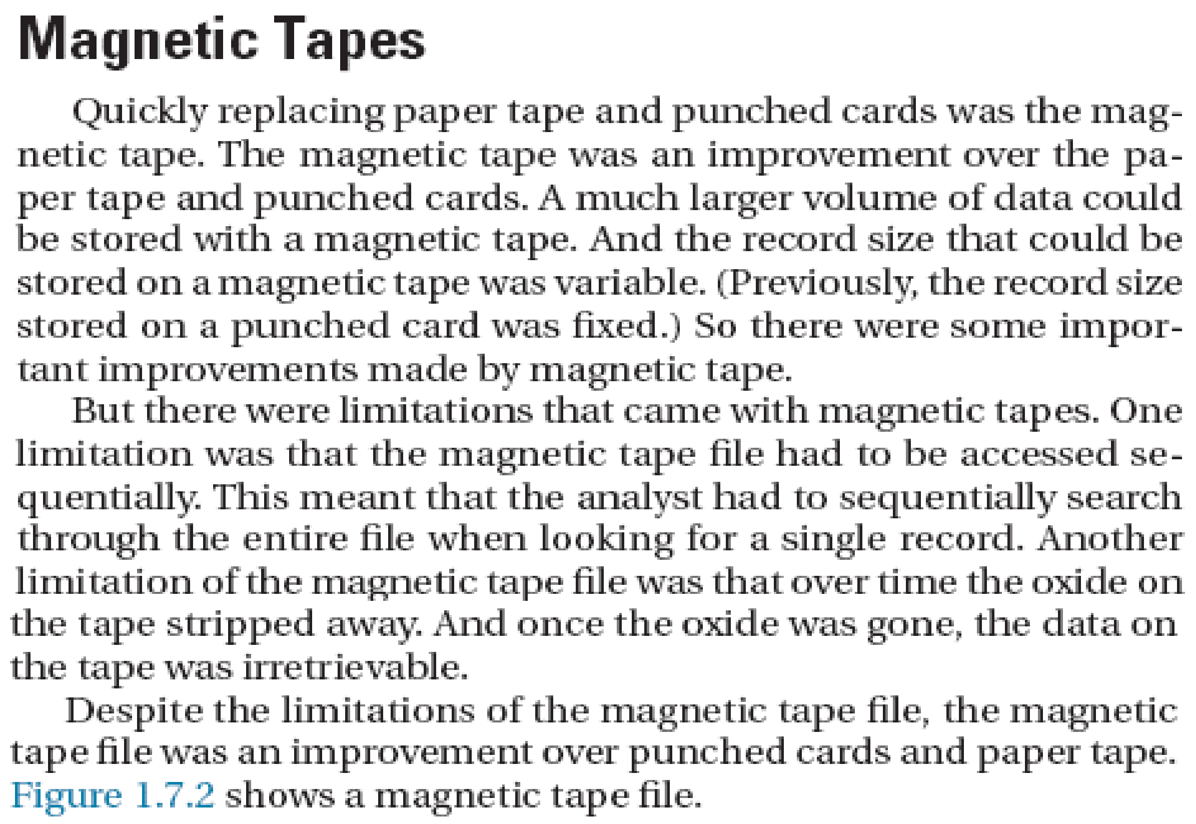
Paper Tape and Punch Cards

But wired boards were clumsy and error prone and could handle only very small volumes of data. Soon an alternative was paper tape and punched cards. Paper tape and punched cards were able to handle larger volumes of data. And there was a greater range of functions that could be handled with punched cards and paper tape. But there were problems with paper tape and punched cards. When a programmer dropped a deck of cards it was a very laborious activity to reconstruct the sequence of the cards. And once a card was punched, it was next to impossible to make a change to the card (although in theory it could be done.) Another shortcoming was that a relatively small amount of data could be held in this media (Figure 1.7.1 depicts the media of cards and paper tape).

Figure 1.7.1

Brief History of data

Inmon &Linstedt,2015



Magnetic Tapes

Quickly replacing paper tape and punched cards was the magnetic tape. The magnetic tape was an improvement over the paper tape and punched cards. A much larger volume of data could be stored with a magnetic tape. And the record size that could be stored on a magnetic tape was variable. (Previously, the record size stored on a punched card was fixed.) So there were some important improvements made by magnetic tape.

But there were limitations that came with magnetic tapes. One limitation was that the magnetic tape file had to be accessed sequentially. This meant that the analyst had to sequentially search through the entire file when looking for a single record. Another limitation of the magnetic tape file was that over time the oxide on the tape stripped away. And once the oxide was gone, the data on the tape was irretrievable.

Despite the limitations of the magnetic tape file, the magnetic tape file was an improvement over punched cards and paper tape. Figure 1.7.2 shows a magnetic tape file.

Figure 1.7.2

Brief History of data

Inmon &Linstedt,2015

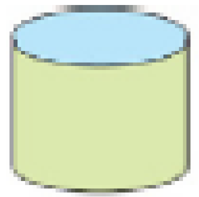


Figure 1.7.3

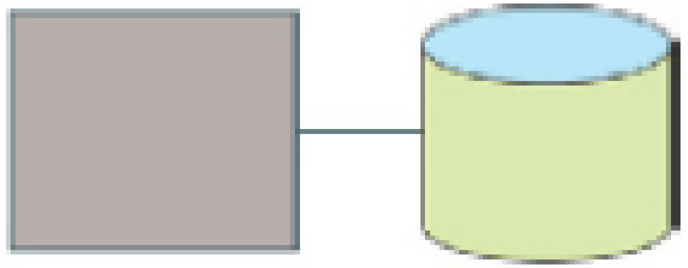


Figure 1.7.4

Disk Storage

The limitations of the magnetic tape file were such that soon there was an alternative medium. That alternative medium was called disk storage (or *direct access storage*). Direct access storage held the great advantage that data could be accessed directly. No longer was it necessary to read an entire file in order to access just one record. With disk storage it was possible to go directly to a unit of data.

Figure 1.7.3 shows disk storage. At first disk storage was expensive and there wasn't all that much capacity that was available. But the hardware vendors quickly improved on the speed, capacity, and cost of disk storage. And the improvements have continued until today.

Database Management System

Along with the advent of disk storage came the appearance of the database management system (DBMS). DBMS controlled the placement, access, update, and deletion of data on disk storage. DBMS saved the programmer from doing repetitive and complex work.

With the appearance of the DBMS came the ability to tie processors to the database (and disk). Figure 1.7.4 shows the advent of the DBMS and the close coupling of the database with the computer.

At first a simple uniprocessor architecture sufficed. In a uniprocessor architecture, there was an operating system, DBMS, and an application. The early computers managed all these components. But in short order, the capacity of the processor was stretched. It was at this point that the capacity considerations of storage switched from improvements on the storage technology to improvements on the management of the storage technology. Prior to this point in time, the great leaps forward in data had been made by improving the storage media. But after this, the great leaps forward were made architecturally, at the processor level.

Soon the uniprocessor simply ran out of capacity. The consumer could always buy a bigger faster processor, but soon the consumer was surpassing the capacity of the largest uniprocessor.

Brief History of data

Inmon &Linstedt,2015

Coupled Processors

The next major advance was the tight coupling together of multiple processors, which is shown in Figure 1.7.5. By coupling together multiple processors, the processing capacity automatically increased. The ability to couple the processors together was made possible by the sharing of memory across the different processors.

Online Transaction Processing

With the advent of greater processing power and the control of DBMS, it was now possible to create a new kind of system. The new kind of system was called the "online real-time system." The processing done by this type of system was called "online transaction processing (OLTP)."

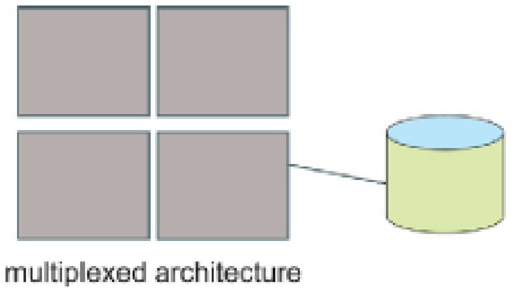


Figure 1.7.5

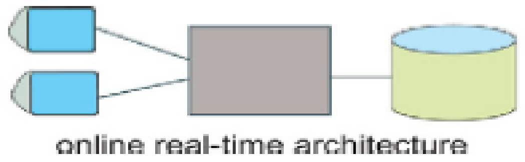


Figure 1.7.6

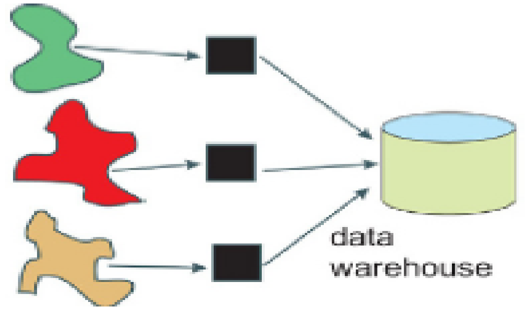


Figure 1.7.7

Figure 1.7.6 shows an online real-time system. With online real-time processing it was now possible to use the computer in a manner that had not before been possible. With the online real-time processing system, the computer could now be used interactively in a manner previously not possible. Suddenly there were airline reservation systems, bank teller systems, ATM systems, inventory management systems, car reservation systems, and many, many more systems. Once real-time online processing became a reality, the computer was used in business as never before.

And with the explosive growth in the usage of the computer there was an explosive growth in amount of data and types of data that were being created. With the flood of data came the desire to have *integrated* data. No longer was it sufficient to merely have data from an application. With the flood of data came the need to look at data in a cohesive manner.

Data Warehouse

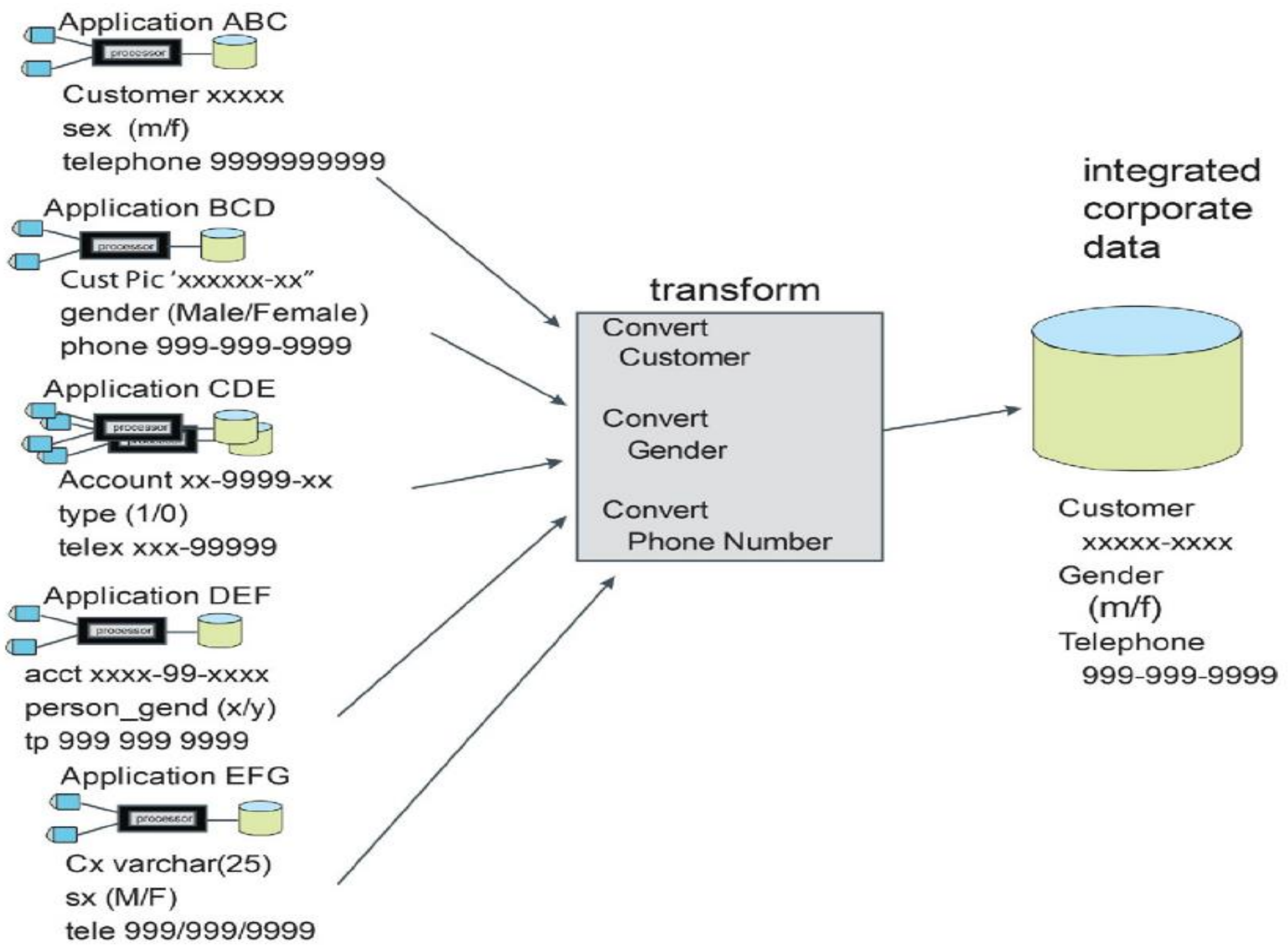
Thus was born the data warehouse, as shown in Figure 1.7.7. With the data warehouse came what was called the "single version of the truth," or the "system of record." With the single version of the truth, the organization now had a foundation of data, which it could turn to with confidence.

The volumes of data continued to explode with the advent of the data warehouse. Prior to the data warehouse there was no convenient place to store historical data. But with the data warehouse, for the first time, there was a convenient and natural place for historical data.

Brief History of data

Inmon &Linstedt,2015

"Single version of the truth!"
Do not forget the ETL tools!!!



Brief History of data

Inmon &Linstedt,2015

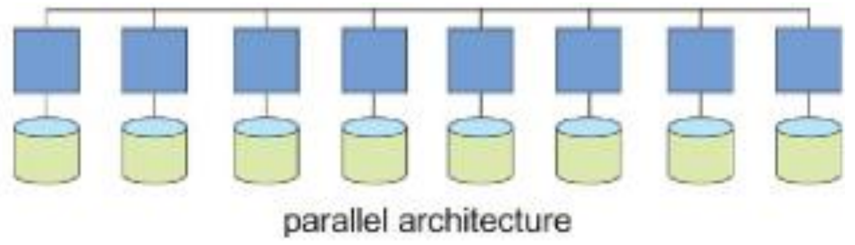


Figure 1.7.8

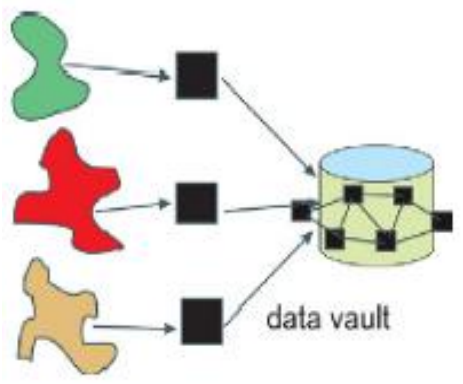


Figure 1.7.9

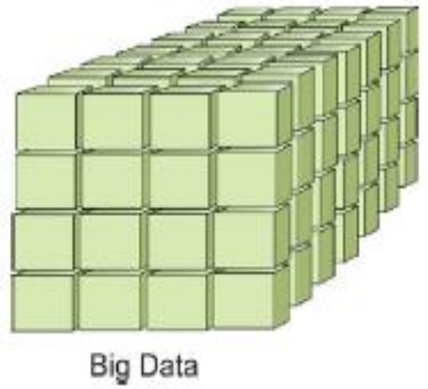


Figure 1.7.10

Parallel Data Management

It is normal and natural that with the ability to store large amounts of data, the demand for data management products and technology skyrocketed. Soon there emerged an architectural approach called the "parallel" approach to data management, which is shown in Figure 1.7.8.

With the parallel approach to data management, a huge amount of data could be accommodated. Far more data could be managed in parallel than was ever possible with nonparallel techniques. With the parallel approach, the limiting factor as to how much data could be managed was an economic limitation, not a technical limitation.

Data Vault

As data warehouses grew, it was realized that there needed to be flexibility in the design of the data warehouse and in the improvement in the integrity of data. Thus born was the "data vault," as shown in Figure 1.7.9. With the data vault, the data warehouse now enjoyed the ultimate in design and integrity.

Big Data

But volumes of data continued to increase. Soon there were systems that went beyond the capacity of even the largest parallel database. A new technology known as *Big Data* evolved in which the optimization of the data management software was on the volumes of data to be managed, not on the ability to access data in an online manner.

Figure 1.7.10 depicts the arrival of Big Data. With Big Data came the advent of the ability to capture and store an almost unlimited amount of data. The arrival of the ability to handle massive amounts of data brought with it the need for a completely new infrastructure.

The Great Divide

And with the recognition of the need for a new infrastructure came the recognition that there were two distinctly different types of Big Data as previously discussed. There is repetitive Big Data and there is nonrepetitive Big Data. And both repetitive Big Data and nonrepetitive Big Data required dramatically different infrastructure.

Brief History of DWH

Inmon &Linstedt,2015

The original concept of a database was as a single source of data for all purposes. This notion of what a database was grew from the magnetic tape file systems where there were master files.

Early Applications

Soon the magnetic tape files were replaced by disk storage as the predominant storage media. And with the advent of magnetic tapes files, where data could be accessed directly, applications grew. Figure 3.1.1 depicts the advent of the early applications where disk storage began to be used.

With disk storage came technology called database management systems (DBMSs). And in short order, online systems appeared.

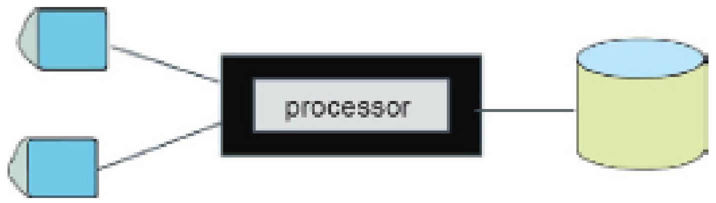


Figure 3.1.1

Online Applications

With online systems it was possible to integrate the computer into places in the business that was not previously possible. Soon there were bank teller systems, airline reservation systems, manufacturing control systems, and the like. The use of the computer in online systems heralded the great dawn of the age of computation as depicted in Figure 3.1.2.

As computer systems began to multiply, it was noticed that there were two basic components of the systems – processing and data. At the same time a problem appeared: There began to be requests to use data from a single system for other purposes. There were some basic reasons for the need to share data across systems. The requirements for the building of the systems were very narrowly defined because the requirements used to shape the systems focused almost entirely on the immediate user of the system or even the clerical help that needed to operate the system. After the system was up and running it was discovered that there were other people who needed to look at and use the data found in the system. People from accounting needed to look at the data. People from marketing needed to look at the data. People from sales and finance needed to look at the data found in the early systems. The problem was that the requirements from these communities had never been considered when the systems were being built.

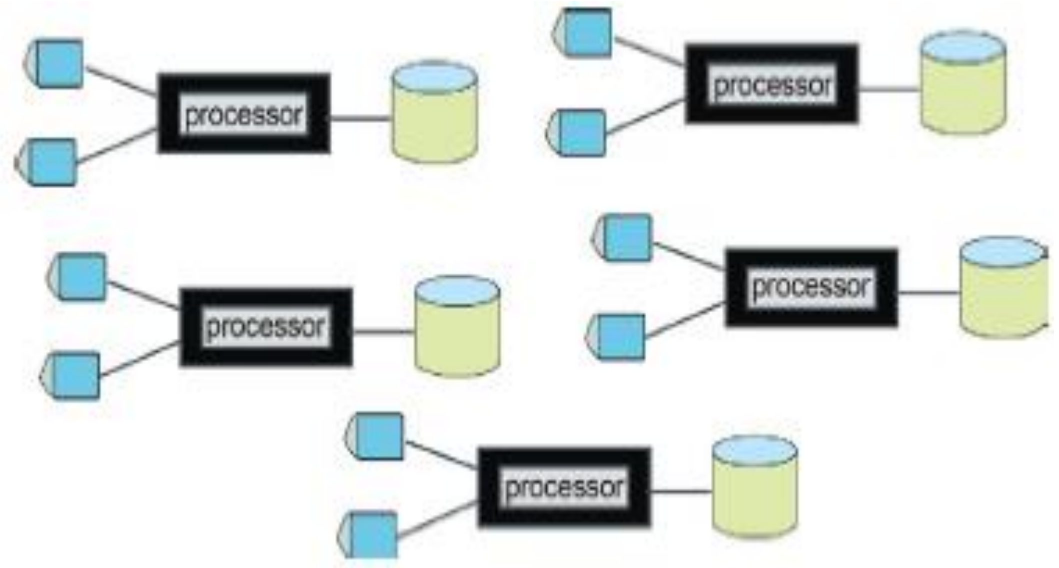


Figure 3.1.2

Brief History of DWH

Inmon &Linstedt,2015

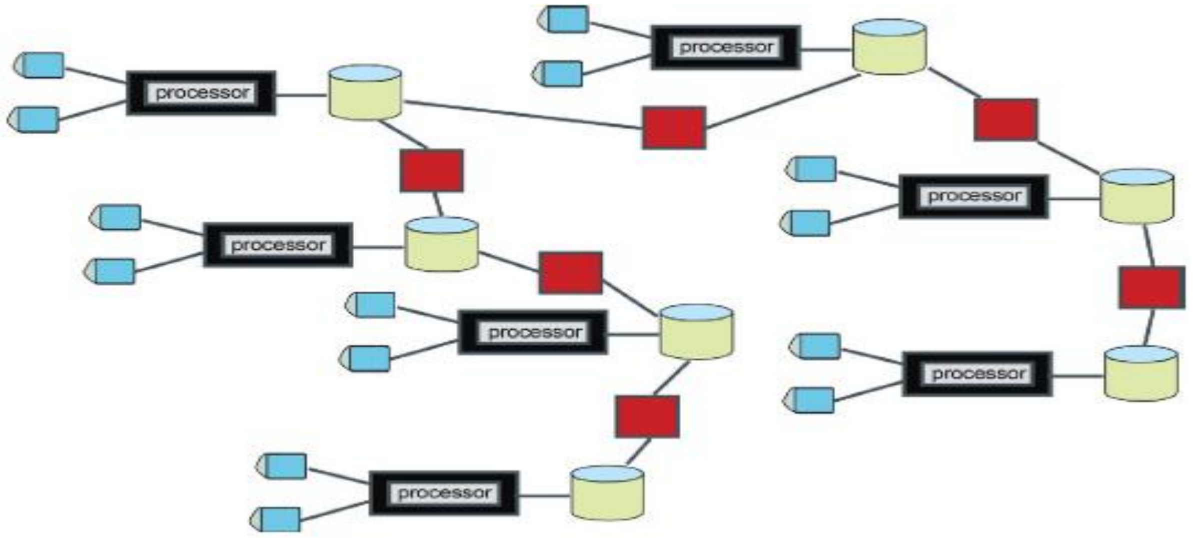


Figure 3.1.3

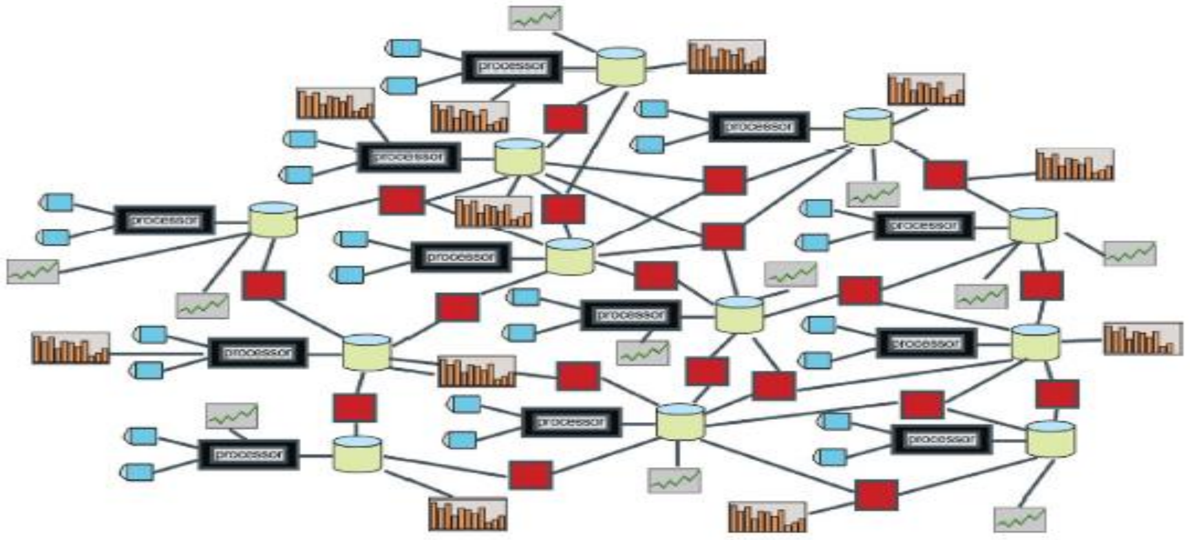


Figure 3.1.4

Extract Programs

Thus the simple "extract" program was born. The extract program was a simple program that read data that had been collected in one system and moved that data to another system. Figure 3.1.3 shows the advent of the innocent looking extract program.

For a while the extract program appeared to satisfy the needs of the organization. But soon end users discovered that they needed information that was not readily available. There was an attitude among the end user community that "the data is in there somewhere if we could just lay our hands on it."

4GL Technology

Into this fray came technology known as "fourth-generation programming language" (4GL) technology as shown in Figure 3.1.4. With 4GL technology the end user was able to take data and do his or her own processing. 4GL technology was the first step that was made to free data from the control of the information technology (IT) department and it gave the end user community the illusion of control of their own destiny.

Brief History of DWH

Inmon &Linstedt,2015

Personal Computers

Shortly after 4GL technology began to be popularized, the personal computer arrived. With the personal computer came the opportunity for even greater autonomy of the end user and even more freedom from the IT department as depicted in Figure 3.1.5.

Spreadsheets

Figure 3.1.6 shows the arrival of the spreadsheet, which gave personal computer users even more autonomy. Another tie had been broken between the end user and the IT department.

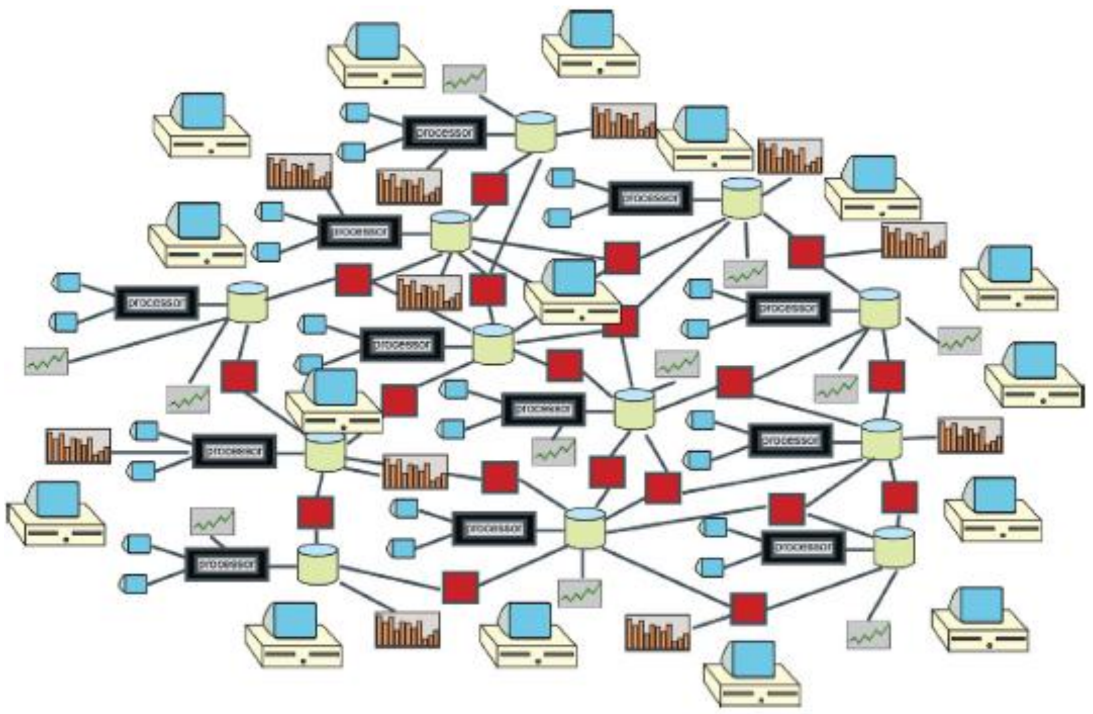


Figure 3.1.5

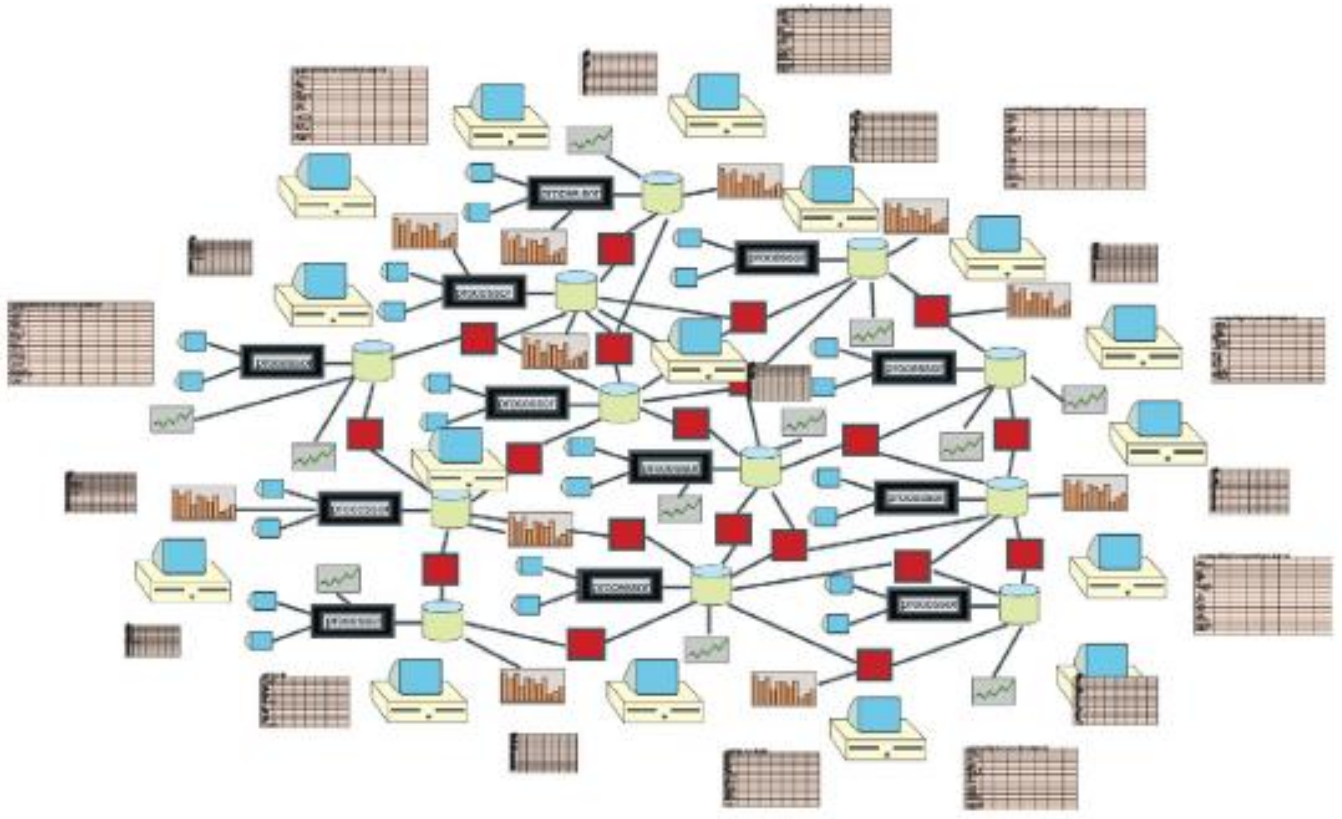
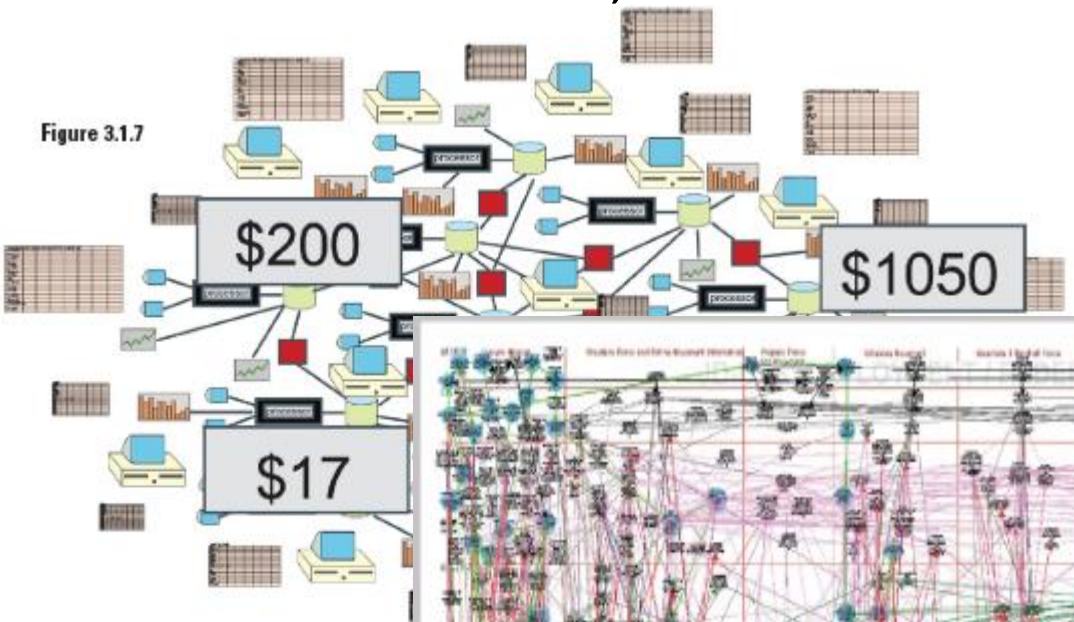


Figure 3.1.6

Brief History of DWH

Inmon &Linstedt,2015

Figure 3.1.7



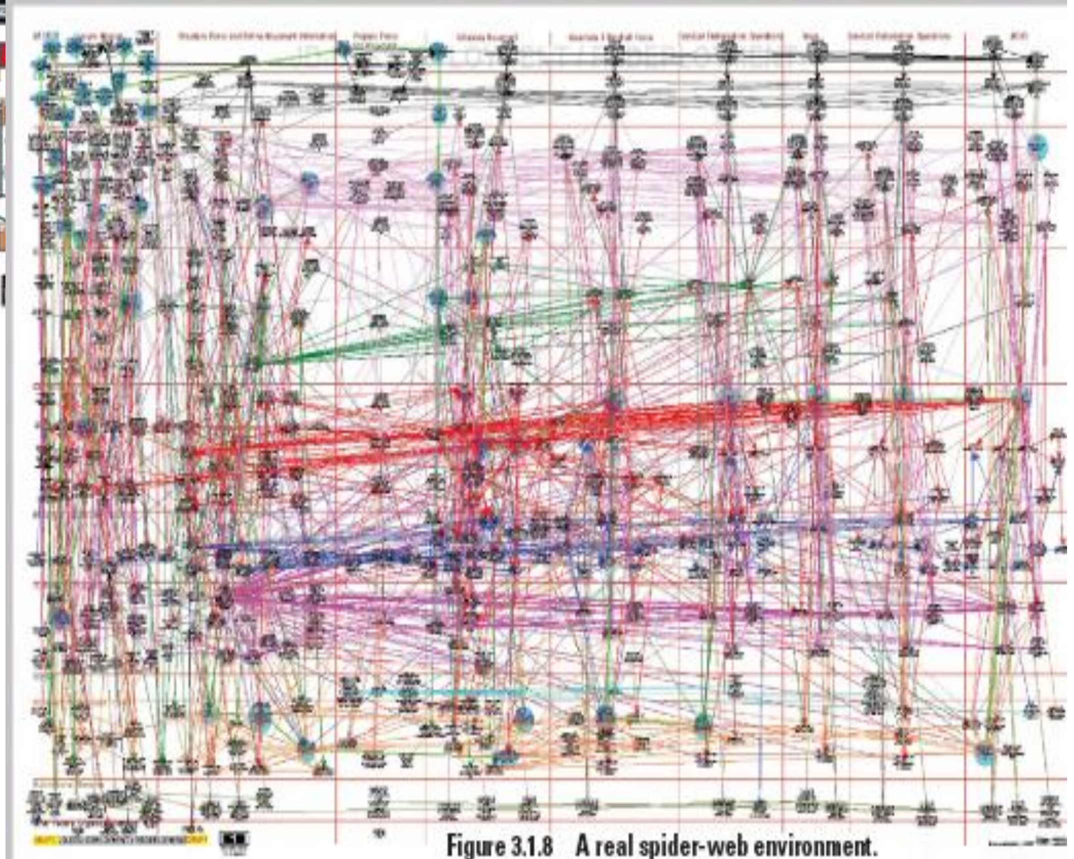
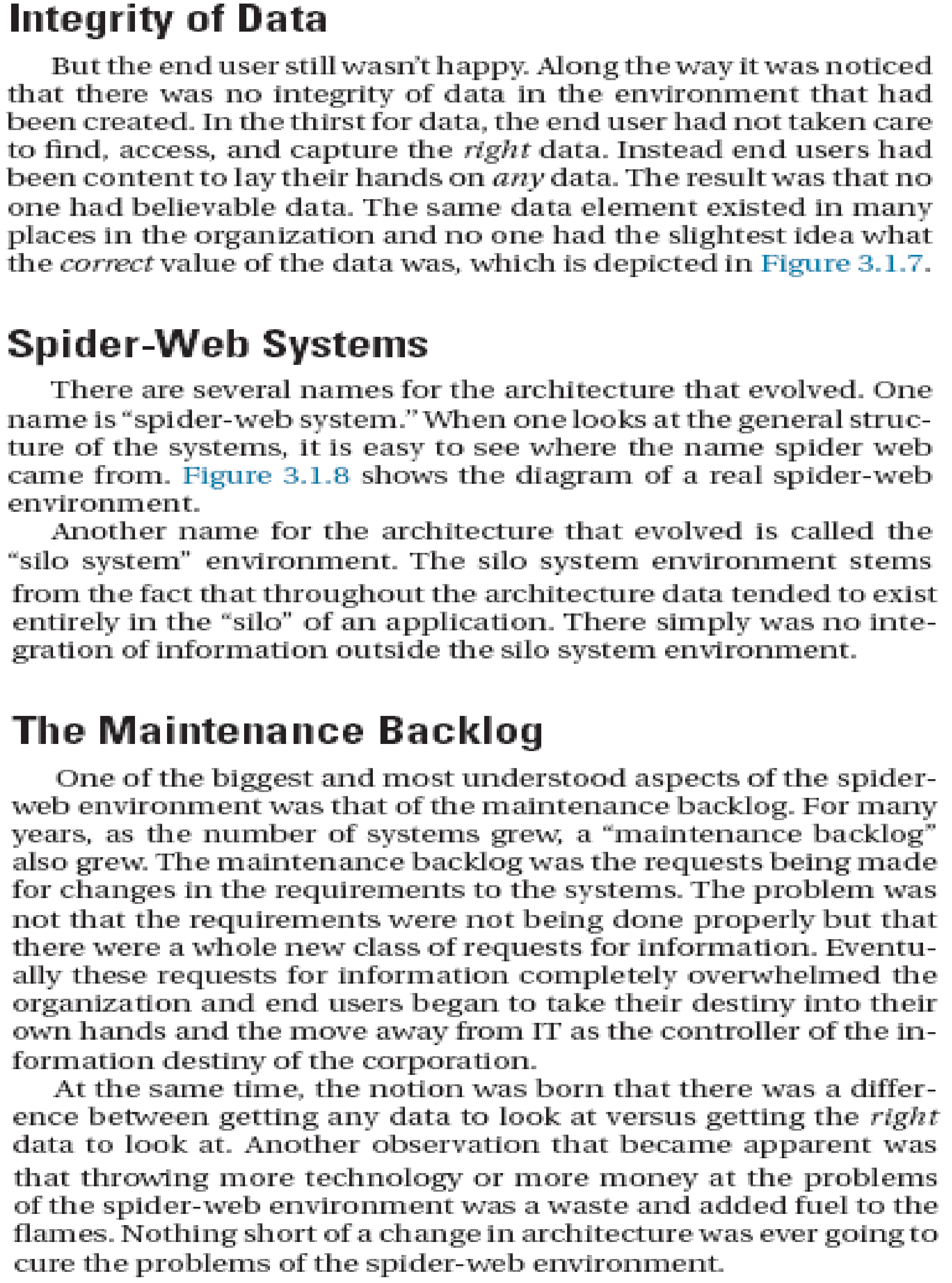


Figure 3.1.8 A real spider-web environment.



Integrity of Data

But the end user still wasn't happy. Along the way it was noticed that there was no integrity of data in the environment that had been created. In the thirst for data, the end user had not taken care to find, access, and capture the *right* data. Instead end users had been content to lay their hands on *any* data. The result was that no one had believable data. The same data element existed in many places in the organization and no one had the slightest idea what the *correct* value of the data was, which is depicted in Figure 3.1.7.

Spider-Web Systems

There are several names for the architecture that evolved. One name is "spider-web system." When one looks at the general structure of the systems, it is easy to see where the name spider web came from. Figure 3.1.8 shows the diagram of a real spider-web environment.

Another name for the architecture that evolved is called the "silo system" environment. The silo system environment stems from the fact that throughout the architecture data tended to exist entirely in the "silo" of an application. There simply was no integration of information outside the silo system environment.

The Maintenance Backlog

One of the biggest and most understood aspects of the spider-web environment was that of the maintenance backlog. For many years, as the number of systems grew, a "maintenance backlog" also grew. The maintenance backlog was the requests being made for changes in the requirements to the systems. The problem was not that the requirements were not being done properly but that there were a whole new class of requests for information. Eventually these requests for information completely overwhelmed the organization and end users began to take their destiny into their own hands and the move away from IT as the controller of the information destiny of the corporation.

At the same time, the notion was born that there was a difference between getting any data to look at versus getting the *right* data to look at. Another observation that became apparent was that throwing more technology or more money at the problems of the spider-web environment was a waste and added fuel to the flames. Nothing short of a change in architecture was ever going to cure the problems of the spider-web environment.

Brief History of DWH

Inmon &Linstedt,2015

The Data Warehouse

It was into this environment that the notion that there should be a data warehouse was born. The idea of a data warehouse suggested that there should be a different types of databases. There should be one type of database for operational systems and another type of database for decision support systems (DSSs).

The notion that there should be two types of databases was heresy to the database theoreticians of the day. The database theoreticians still clung to the theory that there should be one database for all purposes. The advent and popularity of data warehouses shook the academic database theoreticians to their core.

A database was a subject-oriented, integrated, nonvolatile, and time-variant collection of data in support of management's decisions. This notion of a data warehouse has stood the definition of time and has been accepted as the definition of what a data warehouse is.

Another term for the data warehouse was the "single version of the truth." The data warehouse formed the basis for *believable* corporate data. Data across the corporation, not application data, was represented by the data warehouse.

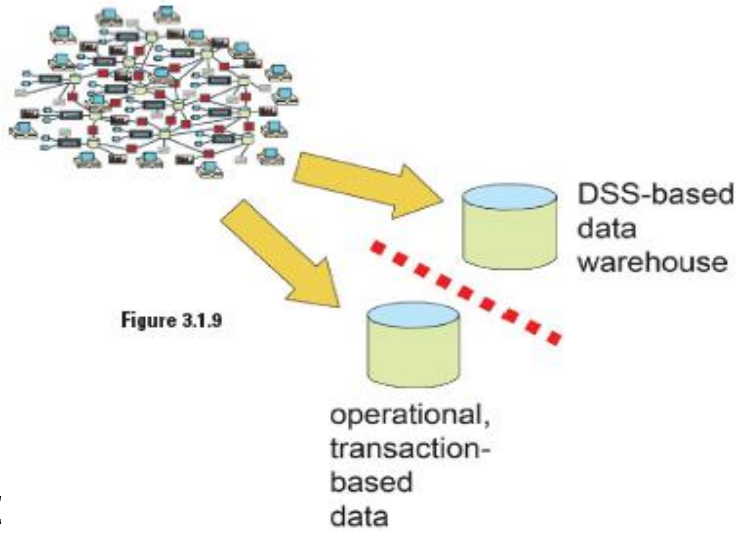


Figure 3.1.9

To an Architected Environment

Figure 3.1.9 shows the movement away from spider-web systems to an architected environment where there are operational databases and data warehouses.

As interesting and as important as the architectural evolution away from spider-web systems to data warehousing was, it was merely the start of an evolution.

To the CIF

Soon it was discovered that there was a lot more to the data warehouse than just a place to store and retrieve data. Indeed a whole infrastructure was required. There was ETL software (extract/transform/load) technology and operational systems. There were data marts, whose structure was centered on dimensional technology, as espoused by Ralph Kimball. There was the operational data store (ODS) that was an essential part of the architecture.

Soon the data warehouse evolved into the corporate information factory (CIF). For many organizations, CIF became the definition of their information architecture. But the evolution of data warehousing did not stop there. After a period of time it was recognized that there was an architecture beyond CIF. The next step in the evolution of information architecture centered on the data warehouse was DW 2.0 architecture.

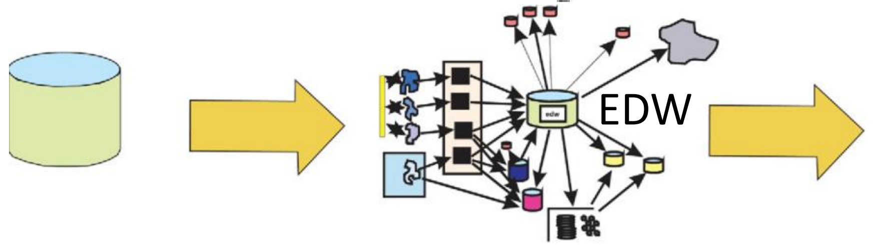
Brief History of DWH

Inmon &Linstedt,2015

DW 2.0

With DW 2.0, several important aspects of the data warehouse environment were recognized. One of these was the life cycle of data within the data warehouse environment. It was simply true that over time data began to live out its own life cycle once entered into a data warehouse.

Another important point of DW 2.0 was that unstructured data was an important and essential aspect of the world of data warehousing. Prior to DW 2.0 the only data found in data warehousing was operational structured information. But with DW 2.0, there was the recognition that unstructured data also belonged in the data warehouse.

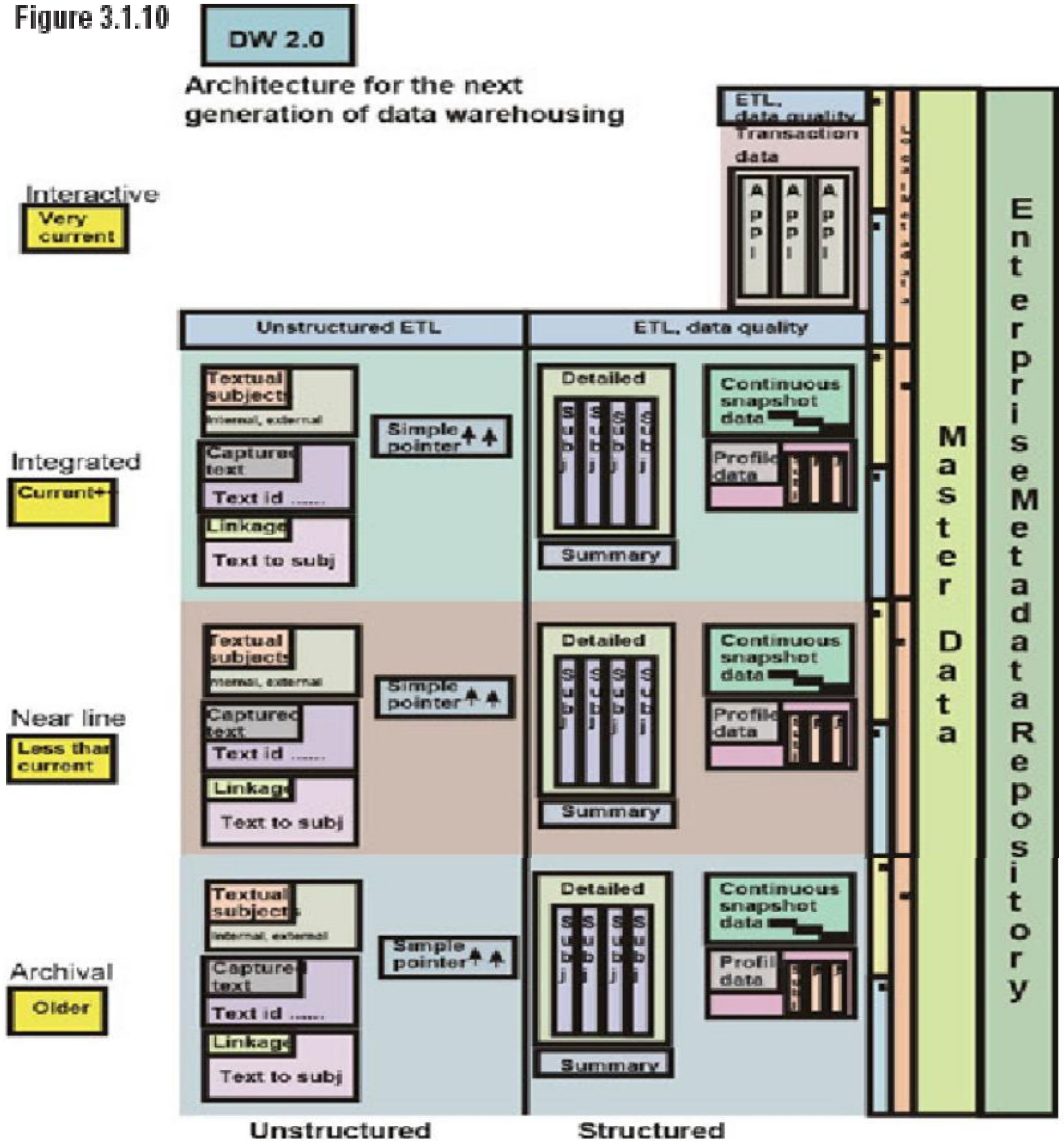


Yet another revelation of DW 2.0 was the recognition that metadata was an integral part of the infrastructure. DW 2.0 recognized that enterprise metadata was as important as local metadata. And finally, at about the time that DW 2.0 was being discussed, it was recognized that the design of data warehouses was greatly enhanced by the advances made by the Data Vault.

But perhaps the greatest advance made by DW 2.0 was the recognition that another form of bulk storage was necessary. DW 2.0 speaks to near line storage. In fact near line storage was a predecessor to Big Data.

Figure 3.1.10 shows the evolution of information architecture surrounding data warehouse.

Figure 3.1.10



DWH data architecture

Kimball &Ross,2016

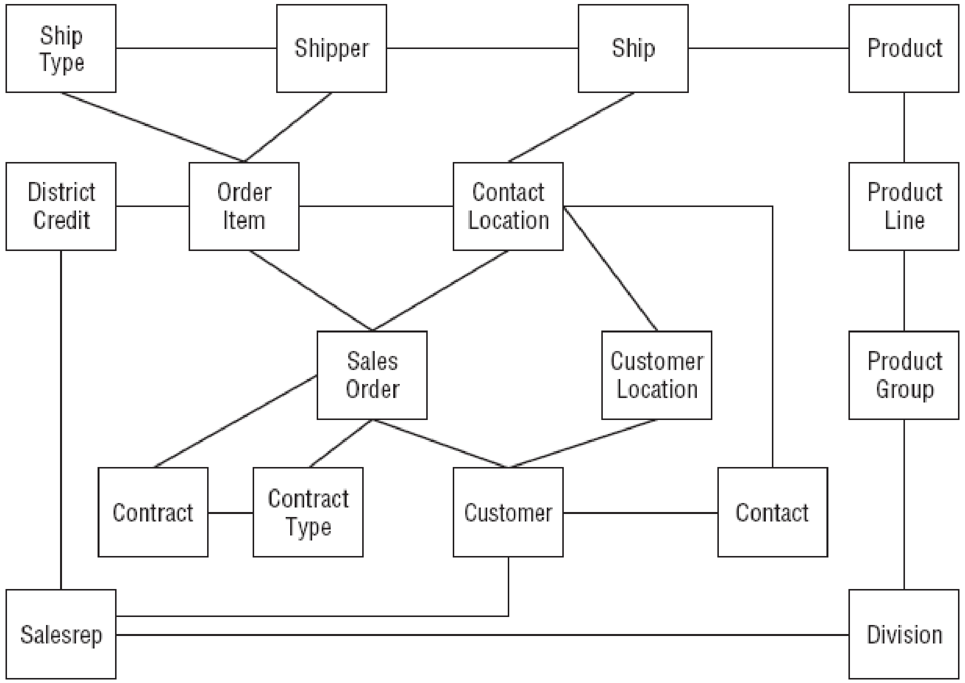


Figure 5-3: A high level 3NF model.

The 3NF modeling technique is a discipline used to illuminate the microscopic relationships among data elements. The highest art form of 3NF modeling is to remove all redundancy in the data. This is immensely beneficial to transaction processing because transactions are made very simple and deterministic. The transaction of updating a customer's address may devolve to a single record lookup in a customer address master table. This lookup is controlled by a customer address key, which defines uniqueness of the customer address record and allows an indexed lookup that is extremely fast. It is safe to say that the success of transaction processing in relational databases is mostly due to the discipline of 3NF modeling.

However, in our zeal to make transaction processing efficient, we have lost sight of our original, most important goal. We have created databases that cannot be queried! Even our simple orders example creates a database of dozens of normalized tables linked together by a bewildering spider web of joins, as illustrated in Figure 5-3.

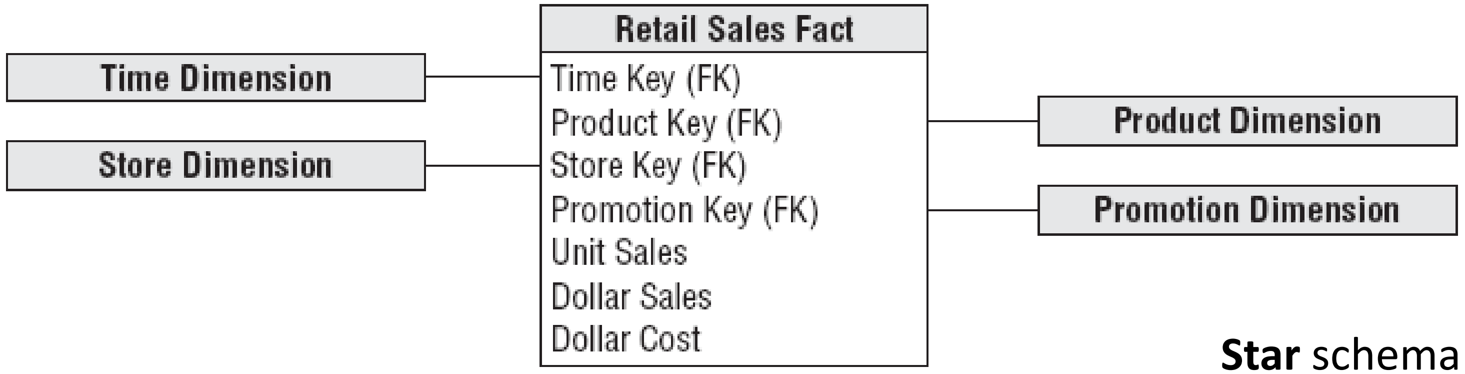
This situation is not just an annoyance, it is a showstopper:

- Business users cannot understand, navigate, or remember a 3NF model. There is no graphical user interface (GUI) that takes a general 3NF model and makes it usable by users.
- Software cannot usefully query a general 3NF model. Cost-based optimizers that attempt to do this are notorious for making the wrong choices, with disastrous consequences for performance.
- Use of the 3NF modeling technique defeats the basic allure of data warehousing, namely intuitive and high performance retrieval of data.

DWH data architecture

Kimball &Ross,2016

Normalized modeling is a powerful technique for designing transaction processing systems in relational environments. The normalization of physical data structures has greatly contributed to the phenomenal success of getting large amounts of data into relational databases. However, models normalized to third normal form do not contribute to the users' ability to query the data. I recommend a different technique, called dimensional modeling (DM), to structure data for querying. Now that you have succeeded in getting the data into your operational databases, it is time to get the data out.



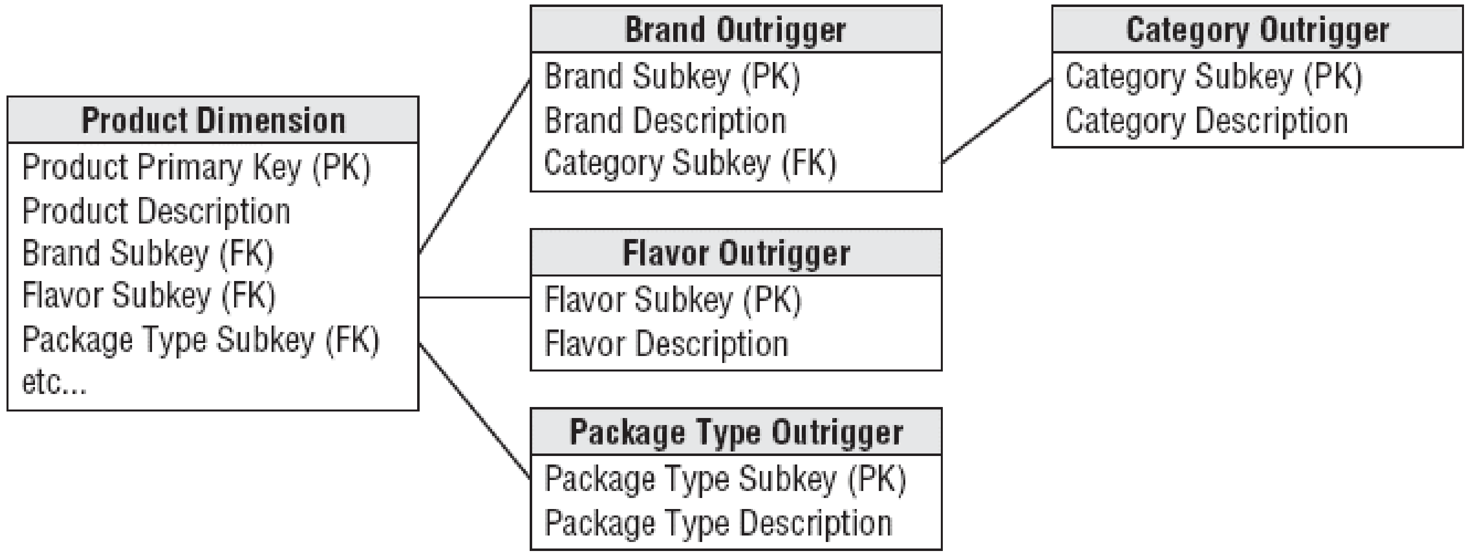
Star schema

Figure 5-1: A dimensional model for grocery store sales.

DWH data architecture

Kimball &Ross,2016

Snowflake schema



Star schema

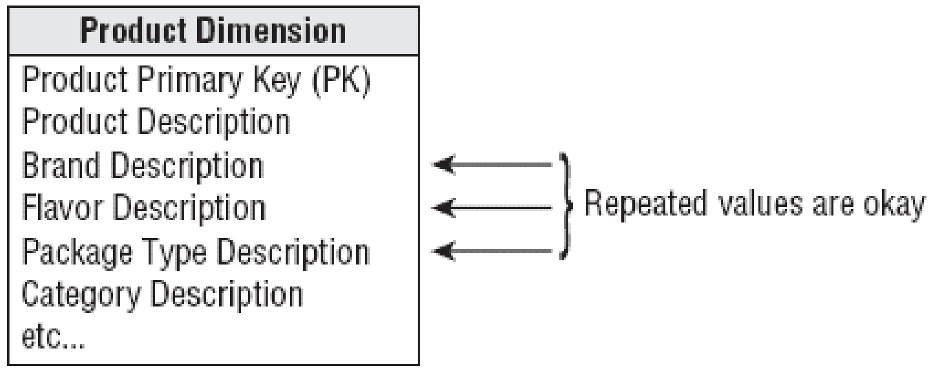
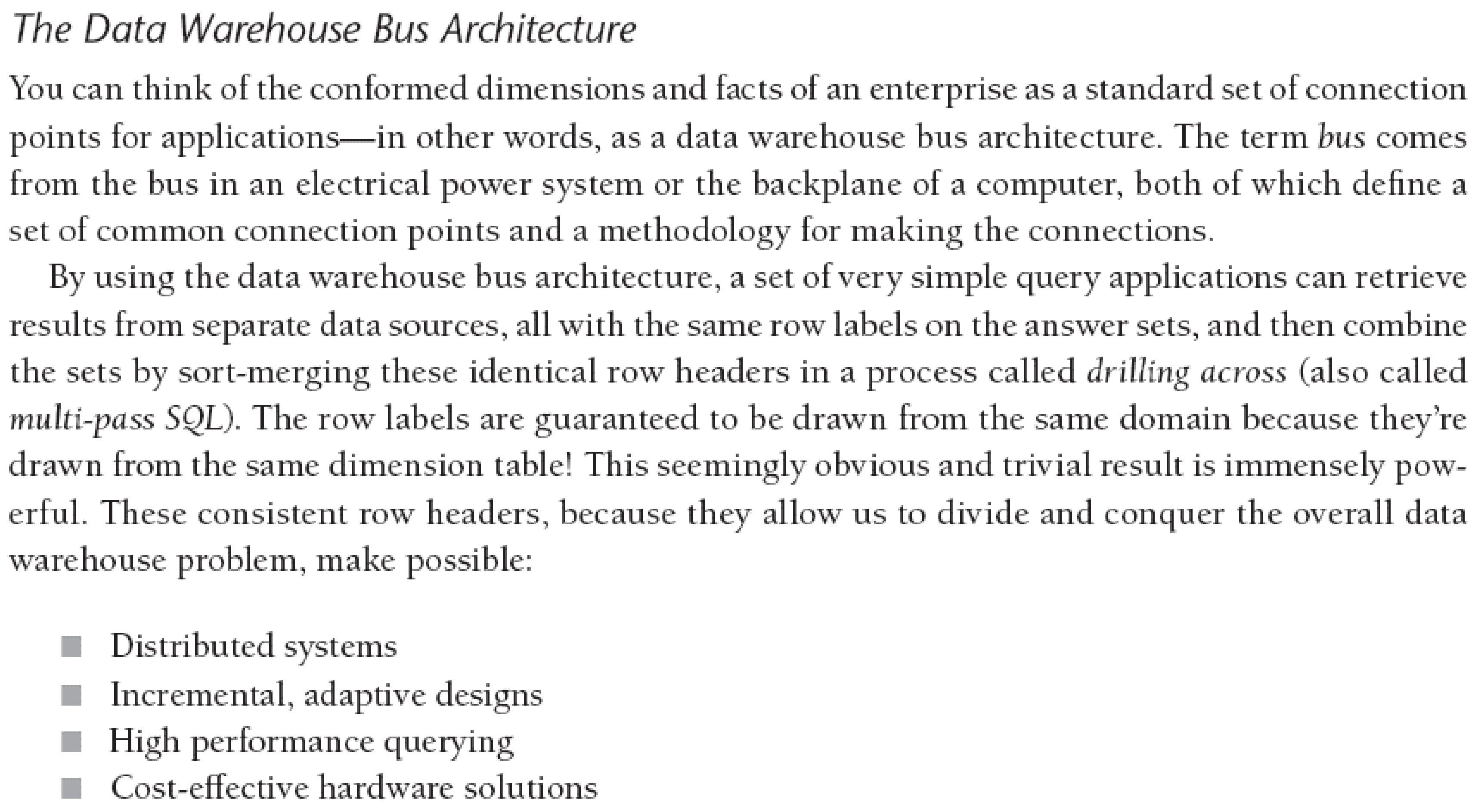


Figure 5-2: Two representations of a product dimension—the top snowflaked version is normalized to 3NF, and the bottom is denormalized into a flat dimension.

DWH data architecture

Kimball &Ross,2016



The Data Warehouse Bus Architecture

You can think of the conformed dimensions and facts of an enterprise as a standard set of connection points for applications—in other words, as a data warehouse bus architecture. The term *bus* comes from the bus in an electrical power system or the backplane of a computer, both of which define a set of common connection points and a methodology for making the connections.

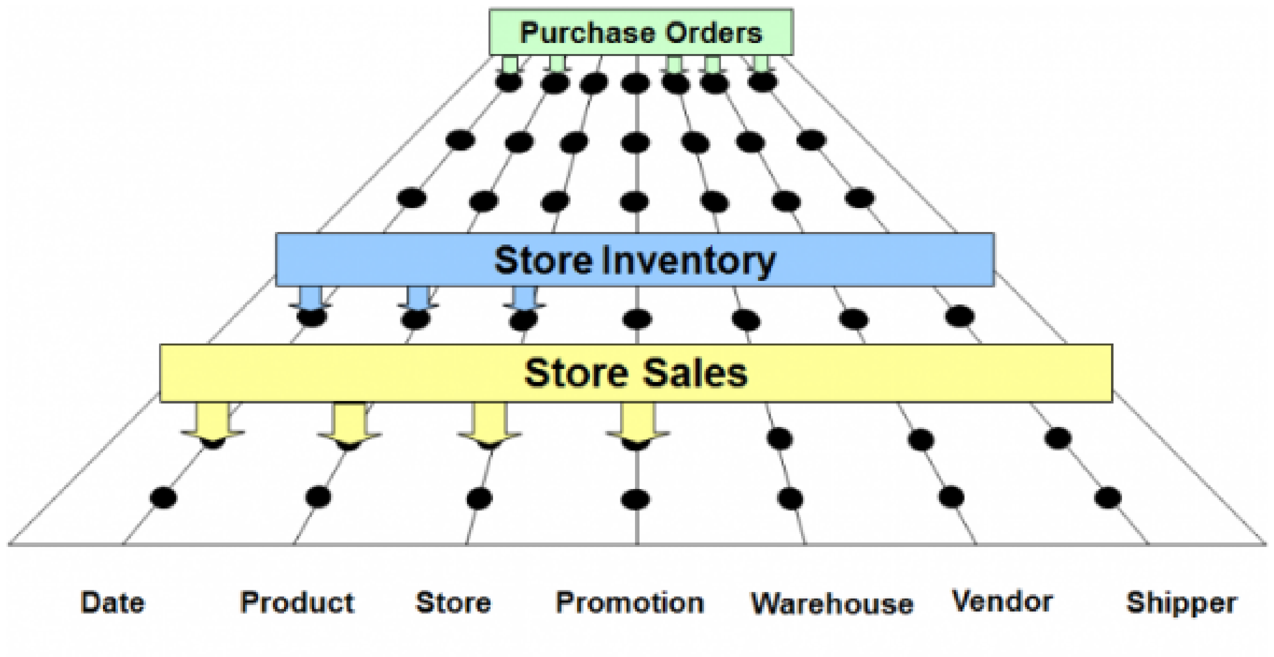
By using the data warehouse bus architecture, a set of very simple query applications can retrieve results from separate data sources, all with the same row labels on the answer sets, and then combine the sets by sort-merging these identical row headers in a process called *drilling across* (also called *multi-pass SQL*). The row labels are guaranteed to be drawn from the same domain because they're drawn from the same dimension table! This seemingly obvious and trivial result is immensely powerful. These consistent row headers, because they allow us to divide and conquer the overall data warehouse problem, make possible:

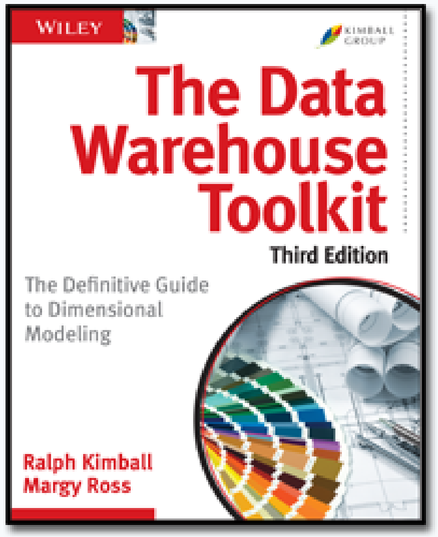
- Distributed systems
- Incremental, adaptive designs
- High performance querying
- Cost-effective hardware solutions

DWH data architecture

Kimball &Ross,2016

The Kimball Group's Enterprise Data Warehouse Bus Architecture is a key element of our approach. Introduced in the 1990s, the technology- and database-independent bus architecture allows for incremental data warehouse and business intelligence (DW/BI) development. It decomposes the DW/BI planning process into manageable pieces by focusing on the organization's core business processes, along with the associated conformed dimensions.




WILEY
The Data Warehouse Toolkit
Third Edition
The Definitive Guide to Dimensional Modeling
Ralph Kimball
Margy Ross

The Data Warehouse Toolkit, 3rd Edition

(Wiley, 2013)

Conformed dimensions are common, standardized, master dimensions that are managed once in the extract, transformation, and load (ETL) system and then reused by multiple fact tables. Conformed dimensions deliver consistent descriptive attributes across dimensional models. They support the ability to drill across and integrate data from multiple business processes. Finally, reusing conformed dimensions shortens the time-to-market by eliminating redundant design and development efforts.

DWH data architecture

Kimball &Ross,2016

Kimball Bus Architecture

Staging begins with coordinated extracts from the operational source systems. Some staging "kitchen" activities are centralized, such as maintenance and storage of common reference data, while others may be distributed.

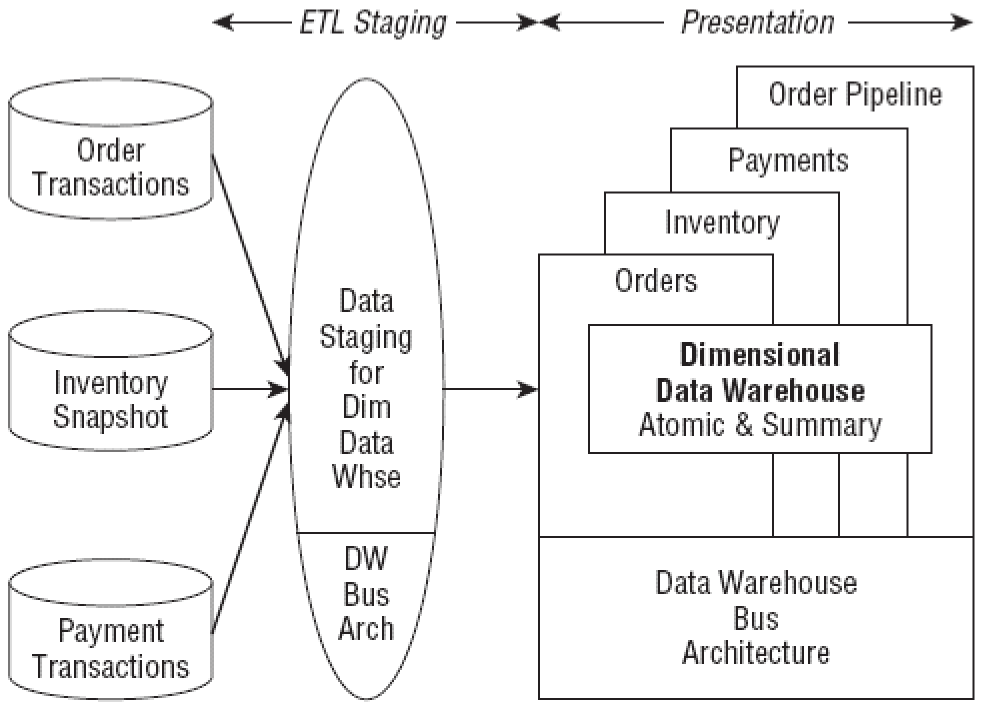


Figure 5-9: Dimensional data warehouse.

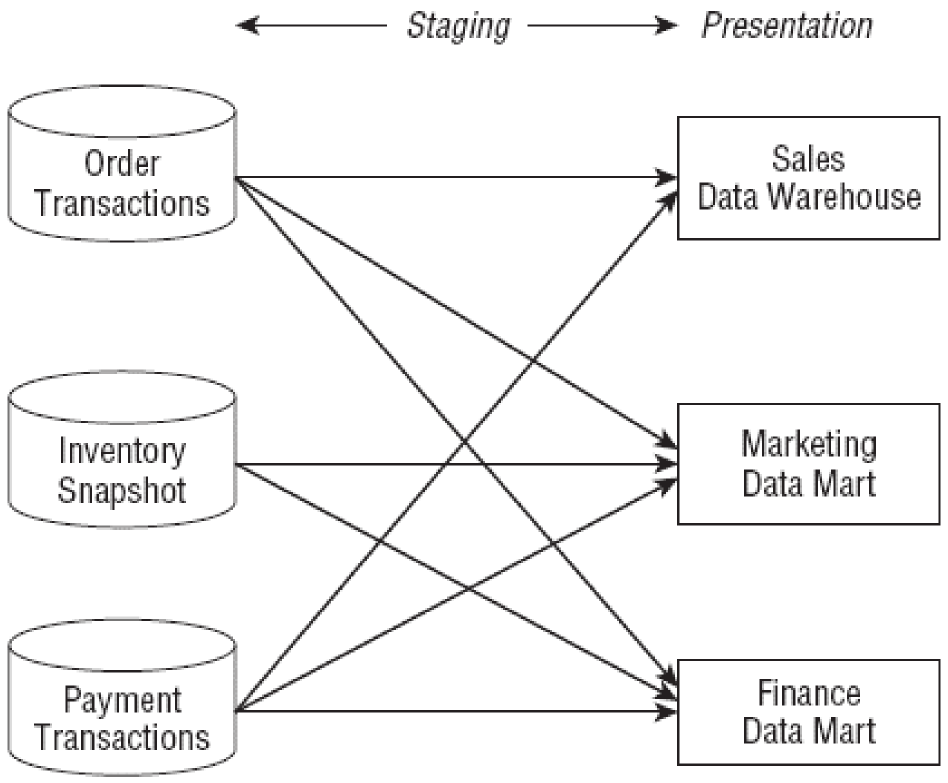


Figure 5-8: Independent data marts/warehouses.

DWH data architecture

Kimball &Ross,2016

Corporate Information Factory

Figure 5-10 illustrates the Corporate Information Factory (CIF) approach, once known as the EDW approach. Like the Kimball approach, there are coordinated extracts from the source systems. From there, a third normal form (3NF) relational database containing atomic data is loaded. This normalized data warehouse is used to populate additional presentation data repositories, including special purpose warehouses for exploration and data mining, as well as data marts.

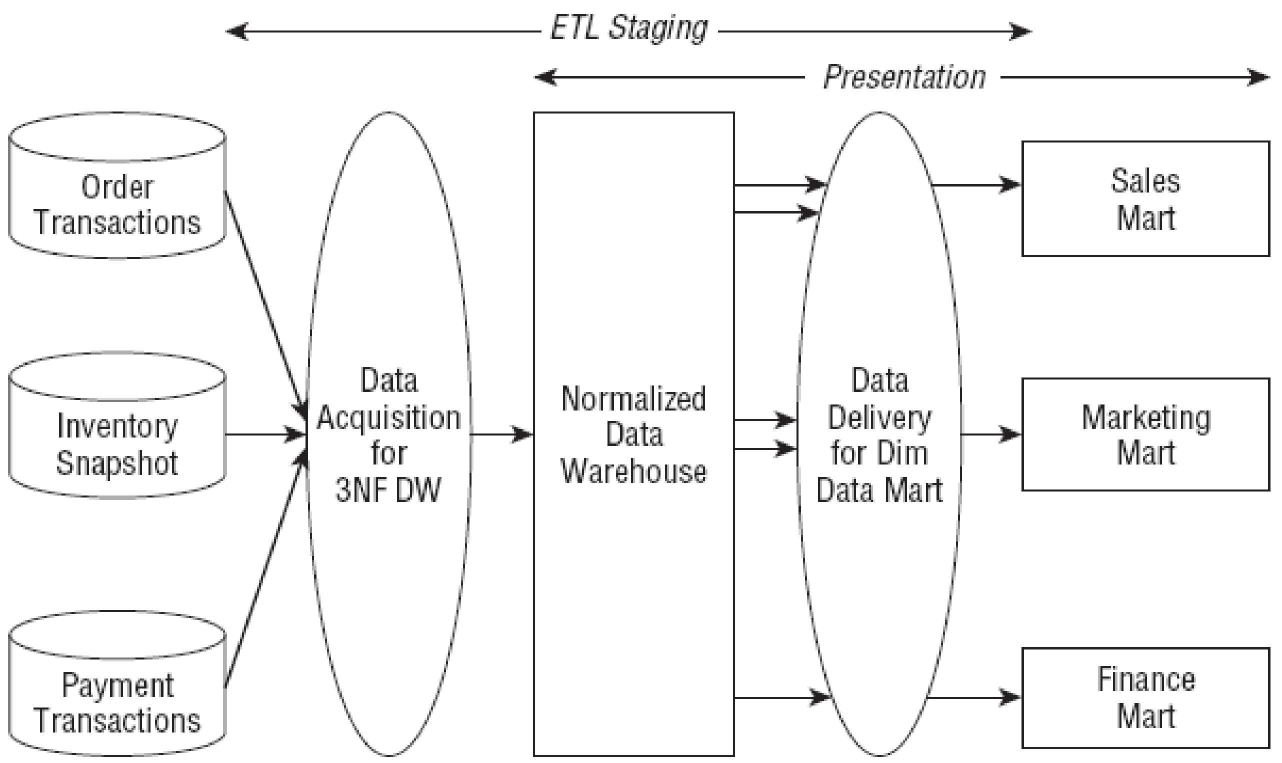


Figure 5-10: Normalized data warehouse with departmental data marts.

DWH data architecture

Kimball &Ross,2016

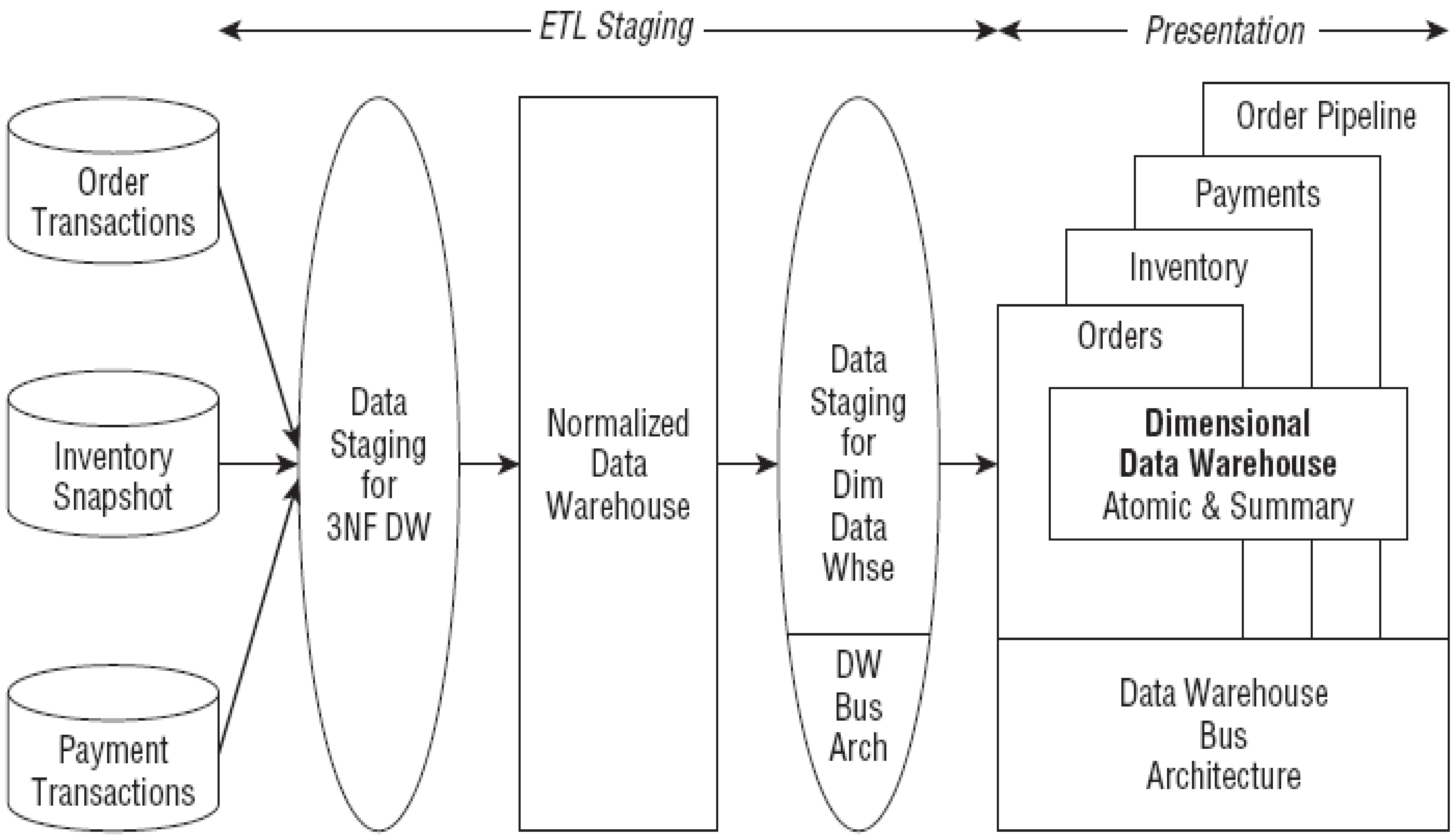


Figure 5-11: Hybrid of normalized data warehouse and dimensional data warehouse.

DWH data architecture

Inmon &Linstedt,2015

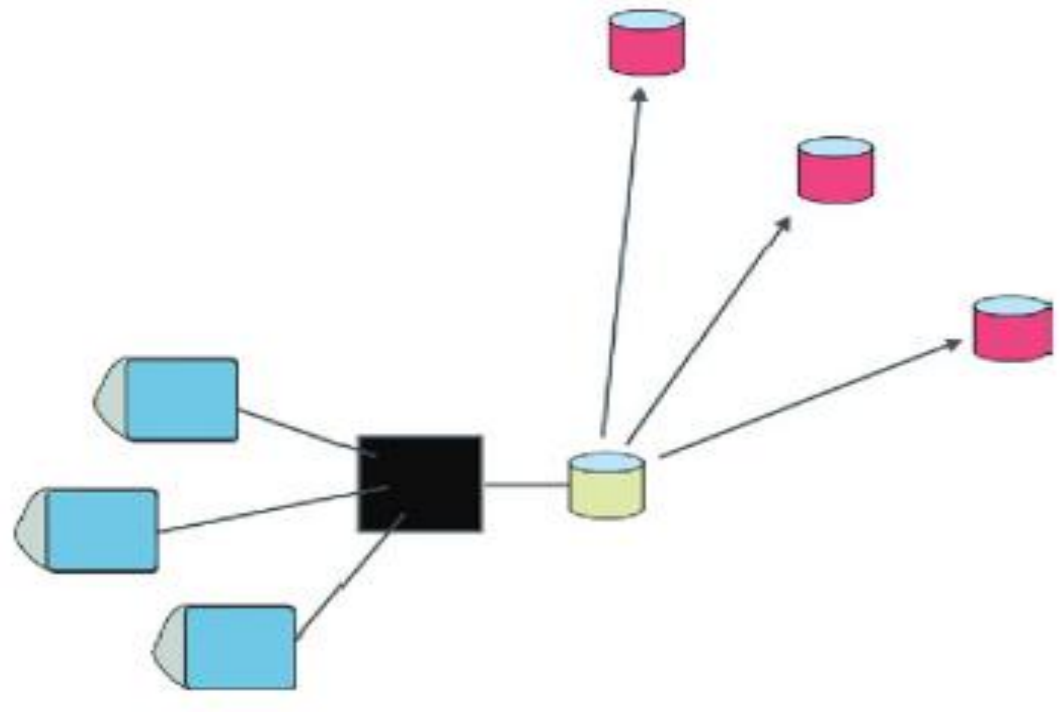
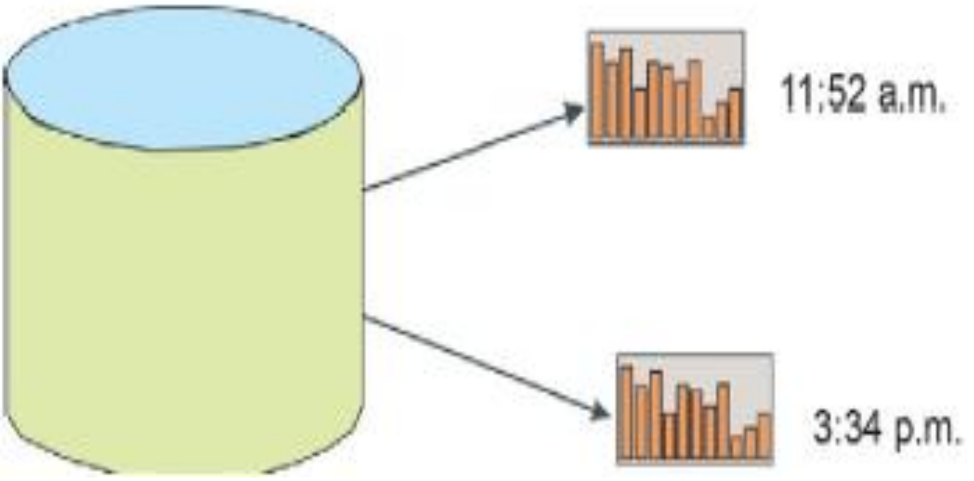


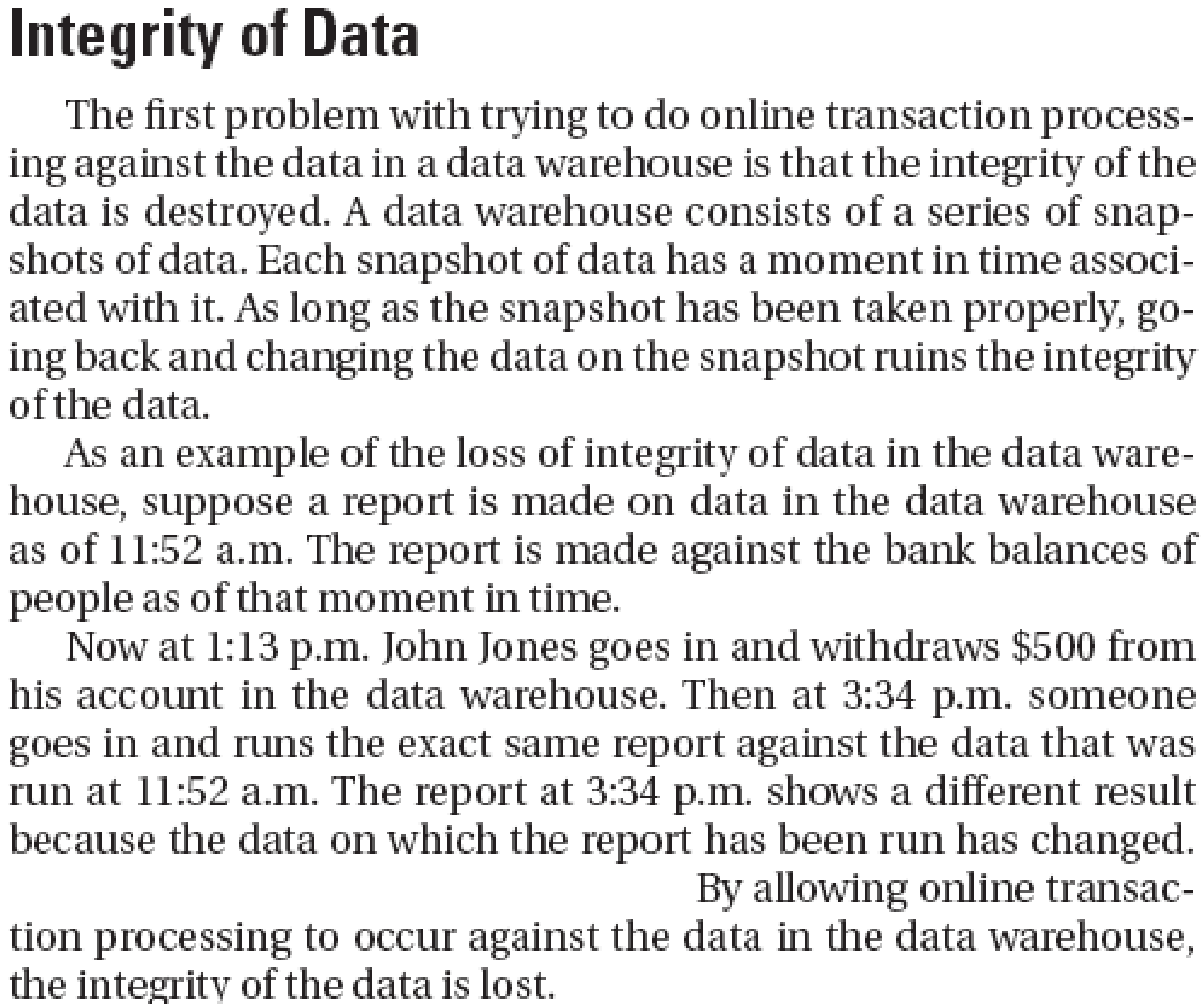
Figure 3.6.2

Figure 3.6.2 shows that online transaction processing is being done on a data warehouse. So what is wrong with this picture? It turns out that there is plenty that is wrong.

DWH data architecture

Inmon &Linstedt,2015





Integrity of Data

The first problem with trying to do online transaction processing against the data in a data warehouse is that the integrity of the data is destroyed. A data warehouse consists of a series of snapshots of data. Each snapshot of data has a moment in time associated with it. As long as the snapshot has been taken properly, going back and changing the data on the snapshot ruins the integrity of the data.

As an example of the loss of integrity of data in the data warehouse, suppose a report is made on data in the data warehouse as of 11:52 a.m. The report is made against the bank balances of people as of that moment in time.

Now at 1:13 p.m. John Jones goes in and withdraws $500 from his account in the data warehouse. Then at 3:34 p.m. someone goes in and runs the exact same report against the data that was run at 11:52 a.m. The report at 3:34 p.m. shows a different result because the data on which the report has been run has changed.

By allowing online transaction processing to occur against the data in the data warehouse, the integrity of the data is lost.

DWH data architecture

Inmon &Linstedt,2015

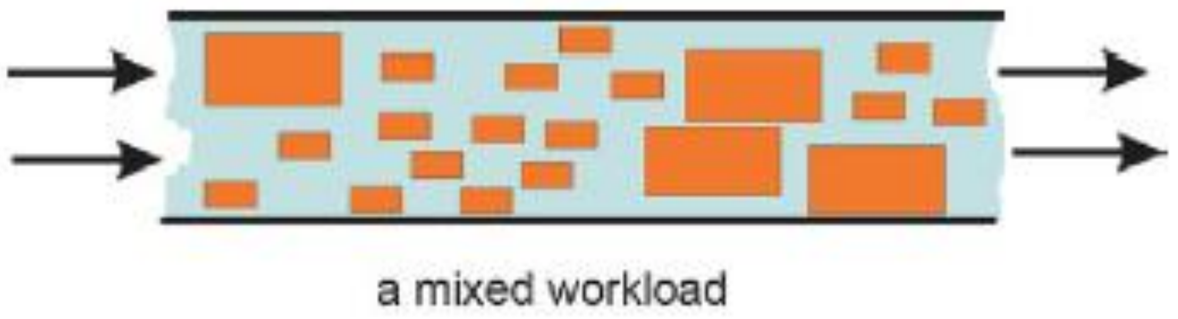


Figure 3.6.4

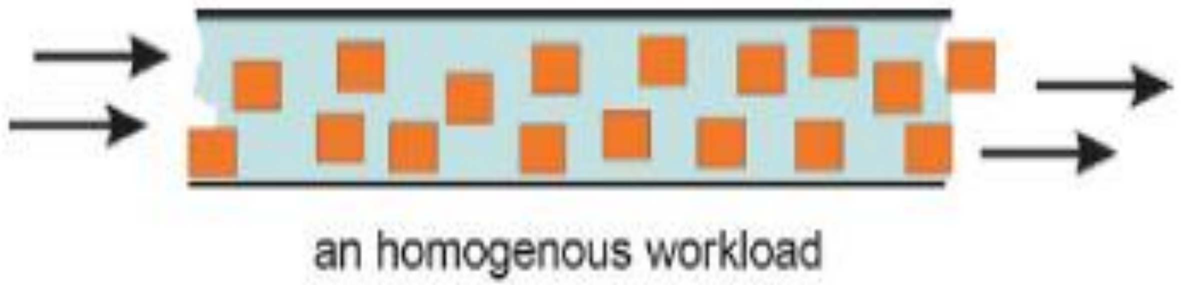


Figure 3.6.5

The Data Warehouse Workload

As important as integrity of data is, it is not the only reason why online transaction processing should not be done against the data in a data warehouse. Another reason is the workload that runs against the data warehouse.

Consider the workload that is suggested by Figure 3.6.4, which shows that there is a mixed workload running against the data warehouse. Some queries are short and fast and other queries are large and bulky.

Now consider the workload suggested in Figure 3.6.5. In Figure 3.6.5 the workload is fast and homogenous.

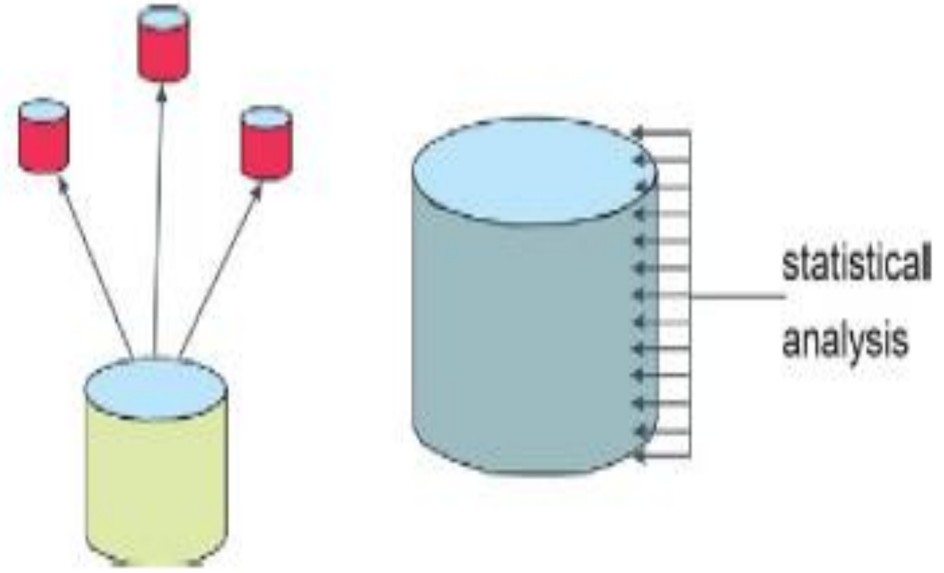
The homogenous workload is typical for an online high-performance system. In order to get the good consistent response time that is a hallmark of an online high-performance system, the work load must be fast and homogenous.

Stated differently, the types of queries done against a data warehouse prevent the data warehouse from being used by high-performance transaction processing. This then is a second good reason why data warehousing should not be used for high-performance transaction processing.

DWH data architecture

Inmon &Linstedt,2015

Figure 3.6.8 shows the data warehouse and the exploration warehouse in a complementary juxtaposition to each other.



The Exploration Warehouse

So what does an organization do if it needs to run regular statistical processing (as some organizations need to do)? In the eventuality that there is a need to run regular statistical processing against data, there is a need to build a specialized structure called an "*exploration warehouse.*"

An exploration warehouse has many similarities to a data warehouse. But there are some distinct differences. Some of the differences between a data warehouse and an exploration warehouse are:

- The data warehouse is a persistent structure, whereas the exploration warehouse is built on a project or as-needed basis.
- The data warehouse is built to accommodate business intelligence (BI) software, whereas the exploration warehouse is built to accommodate statistical analysis software.
- The data warehouse contains data that is highly normalized, whereas the exploration warehouse often contains data that is edited (sometimes called "*convenience fields*"), in anticipation of the statistical analysis that will be done.
- The data warehouse contains data from the legacy environment, whereas the exploration warehouse contains data from the legacy environment and external. In truth, the data warehouse normally does not contain much (if any) external data, whereas the exploration warehouse contains a lot of external data.

DWH data architecture

Inmon &Linstedt,2015

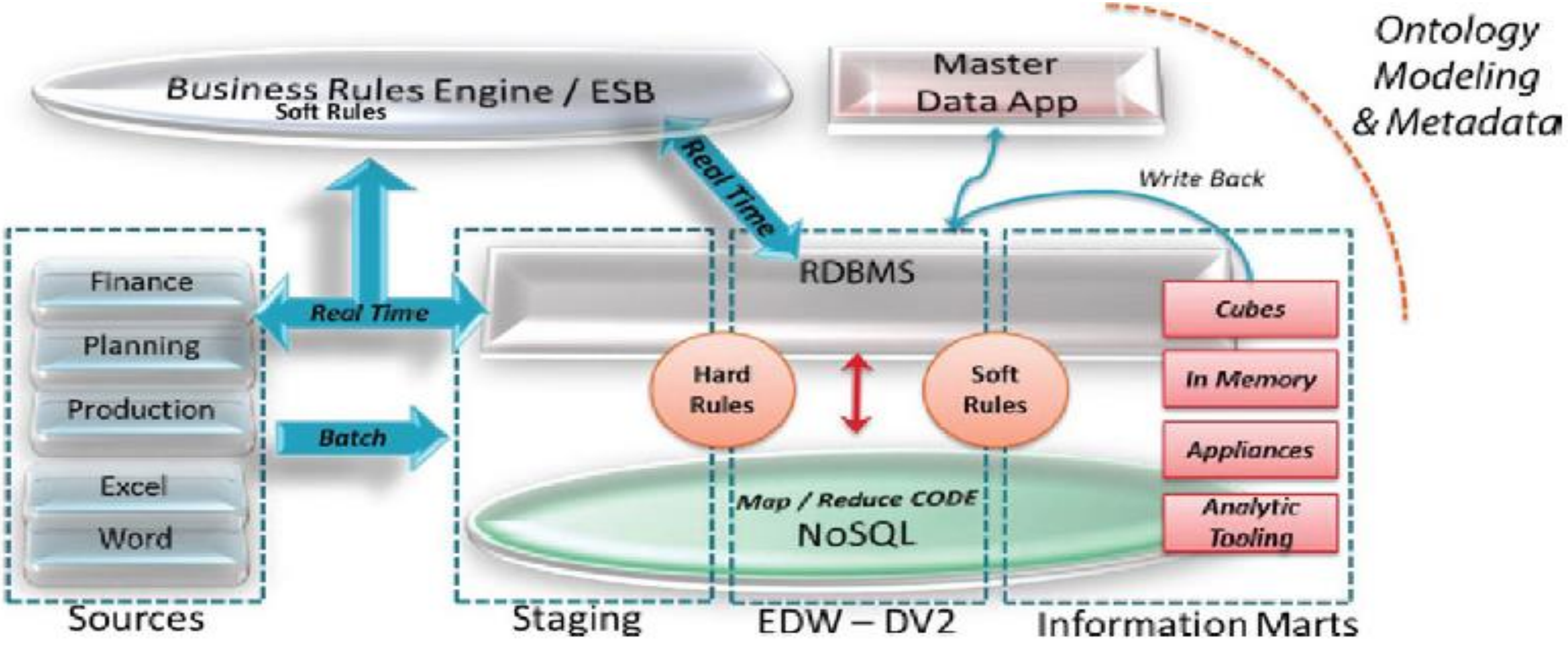


Figure 4.3.1 Data Vault 2.0 Architecture Overview

DWH data architecture

Inmon &Linstedt,2015

in-memory analytics Leveraging advances in memory to provide faster and deeper analytics by querying a system's random-access memory (RAM) instead of disks; in-memory analytics' architectural options include in-memory analytics in the BI tools, as part of the database or on the BI appliance platform.

Cubes
In Memory
Appliances
Analytic Tooling

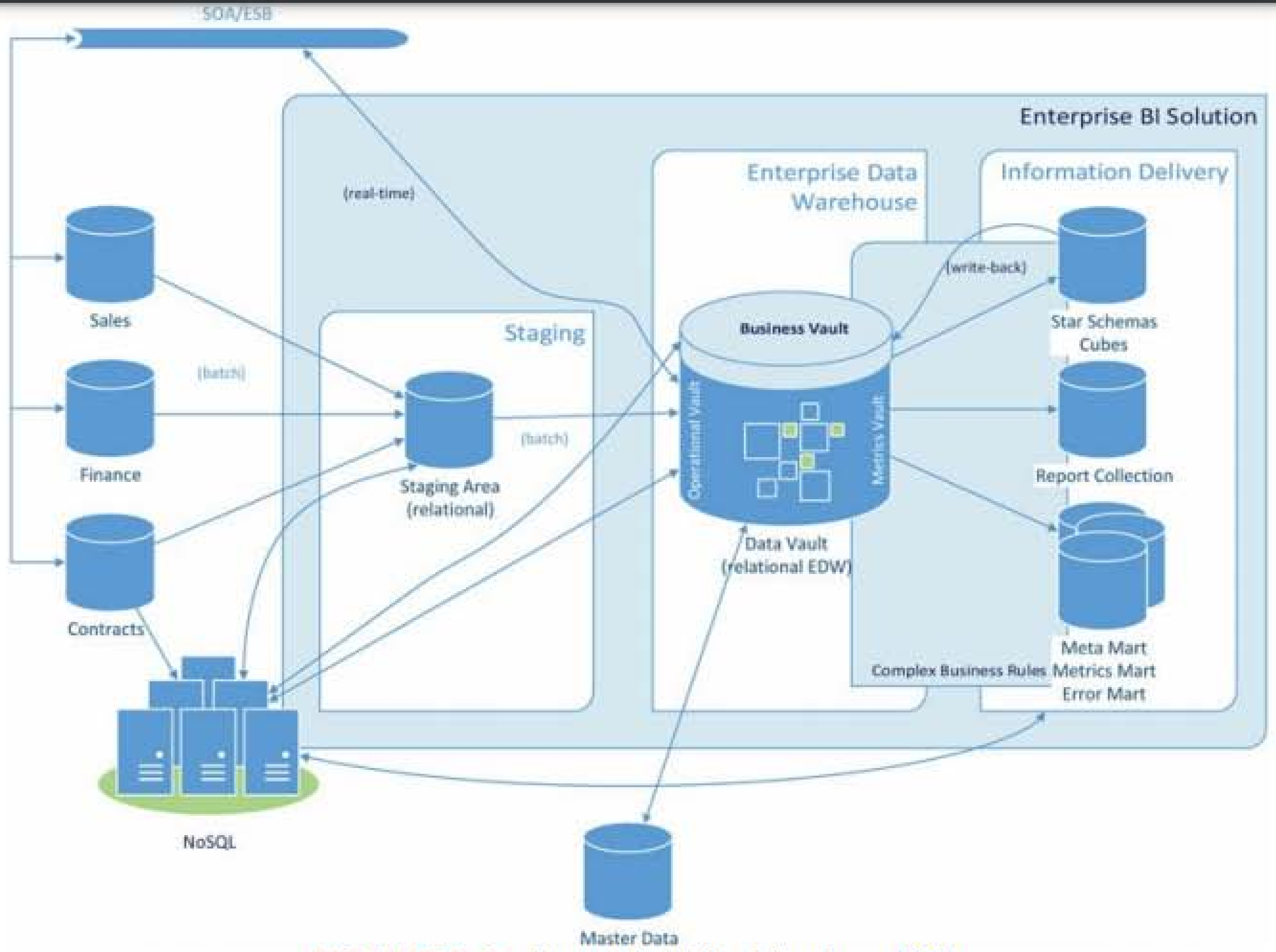
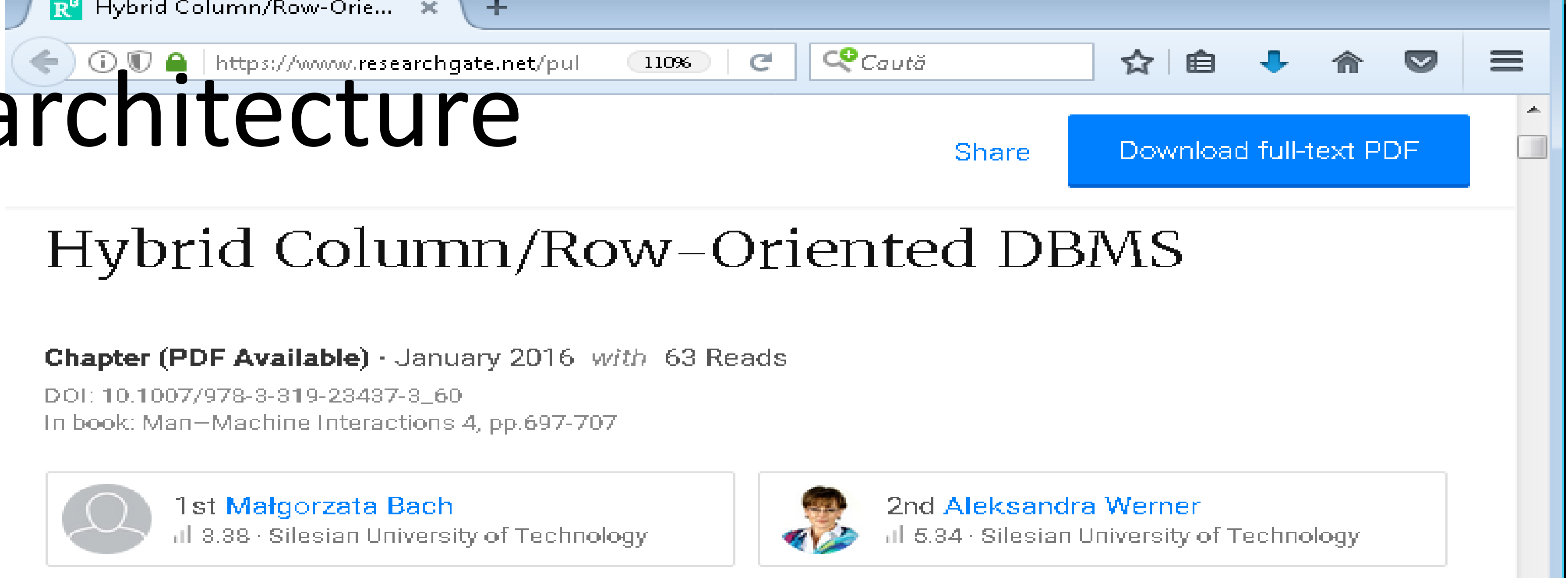


FIGURE 2.2 Data Vault Architecture [19].

The Data Vault 2.0 architecture is based on three layers: the staging area, which collects the raw data from the source systems; the enterprise data warehouse layer, modeled as a Data Vault 2.0 model; and the information delivery layer, with information marts as star schemas and other structures. The architecture supports both batch loading of source systems and real-time loading from the enterprise service bus (ESB) or any other service-oriented architecture (SOA). But it is also possible to integrate unstructured NoSQL database systems into this architecture. Due to the platform independence of Data Vault 2.0, NoSQL can be used for every data warehouse layer, including the stage area, the enterprise data warehouse layer, and information delivery.

DWH data architecture



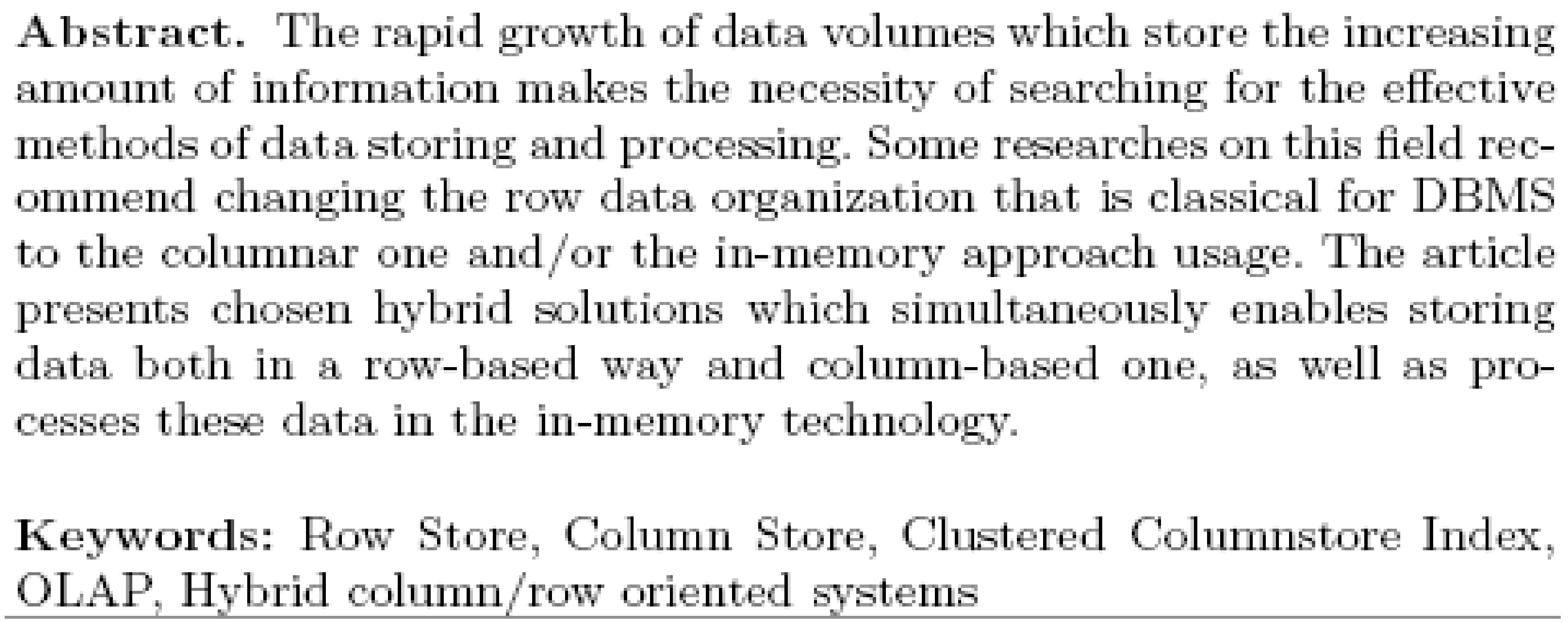
Hybrid Column/Row–Oriented DBMS

Chapter (PDF Available) · January 2016 *with* 63 Reads

DOI: 10.1007/978-3-319-23437-3_60
In book: Man–Machine Interactions 4, pp.697-707

1st Małgorzata Bach
3.38 · Silesian University of Technology

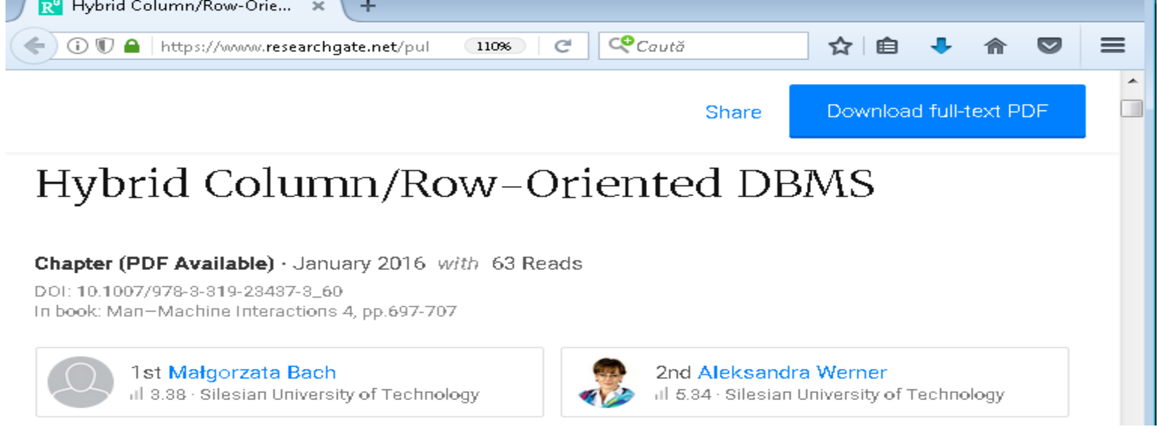
2nd Aleksandra Werner
5.34 · Silesian University of Technology



Abstract. The rapid growth of data volumes which store the increasing amount of information makes the necessity of searching for the effective methods of data storing and processing. Some researches on this field recommend changing the row data organization that is classical for DBMS to the columnar one and/or the in-memory approach usage. The article presents chosen hybrid solutions which simultaneously enables storing data both in a row-based way and column-based one, as well as processes these data in the in-memory technology.

Keywords: Row Store, Column Store, Clustered Columnstore Index, OLAP, Hybrid column/row oriented systems

DWH data architecture



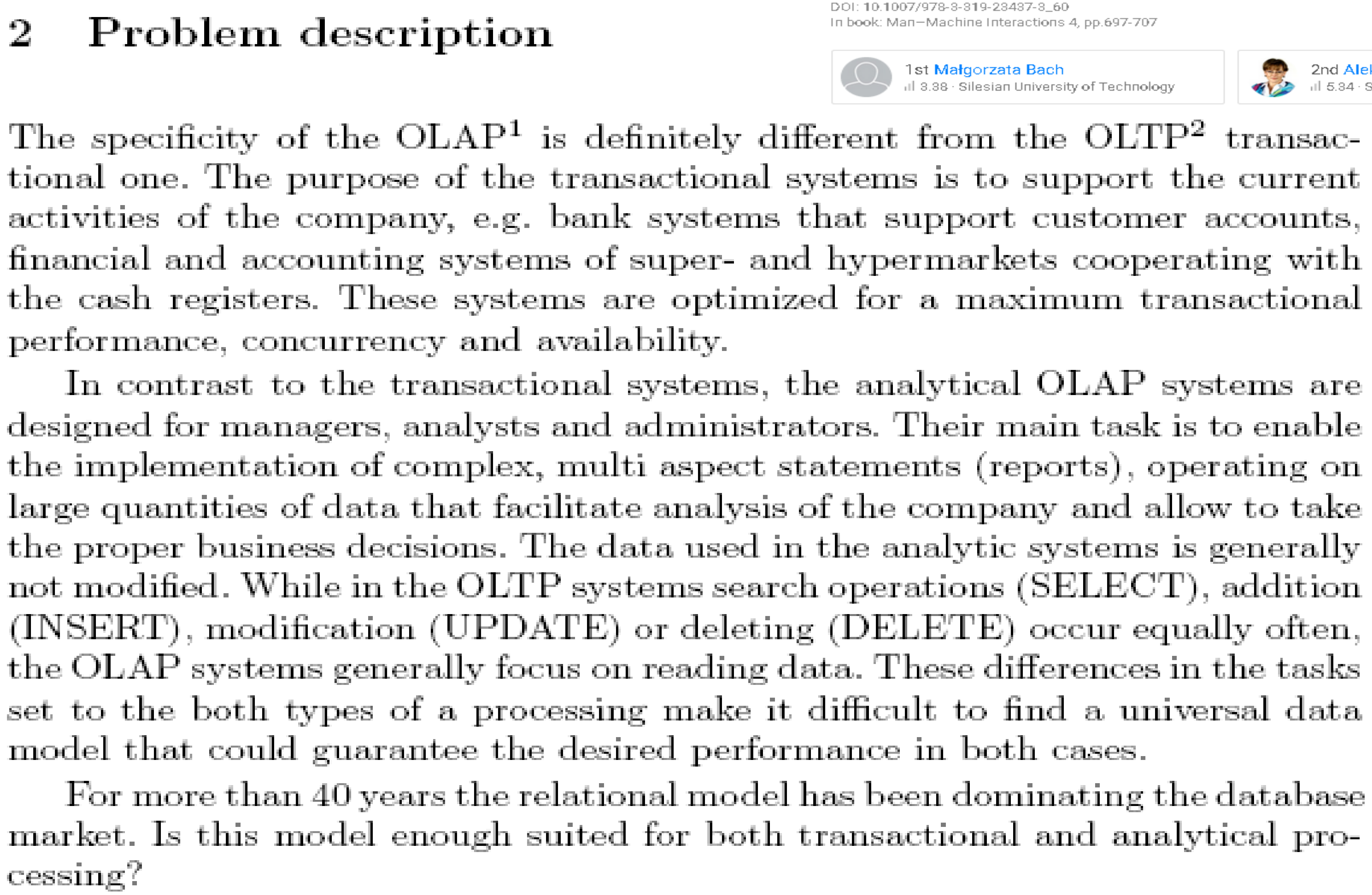
Hybrid Column/Row–Oriented DBMS

Chapter (PDF Available) · January 2016 *with* 63 Reads

DOI: 10.1007/978-3-319-23437-3_60
In book: Man–Machine Interactions 4, pp.697-707

1st Małgorzata Bach
3.38 · Silesian University of Technology

2nd Aleksandra Werner
5.34 · Silesian University of Technology



2 Problem description

The specificity of the OLAP[1] is definitely different from the OLTP[2] transactional one. The purpose of the transactional systems is to support the current activities of the company, e.g. bank systems that support customer accounts, financial and accounting systems of super- and hypermarkets cooperating with the cash registers. These systems are optimized for a maximum transactional performance, concurrency and availability.

In contrast to the transactional systems, the analytical OLAP systems are designed for managers, analysts and administrators. Their main task is to enable the implementation of complex, multi aspect statements (reports), operating on large quantities of data that facilitate analysis of the company and allow to take the proper business decisions. The data used in the analytic systems is generally not modified. While in the OLTP systems search operations (SELECT), addition (INSERT), modification (UPDATE) or deleting (DELETE) occur equally often, the OLAP systems generally focus on reading data. These differences in the tasks set to the both types of a processing make it difficult to find a universal data model that could guarantee the desired performance in both cases.

For more than 40 years the relational model has been dominating the database market. Is this model enough suited for both transactional and analytical processing?

DWH data architecture

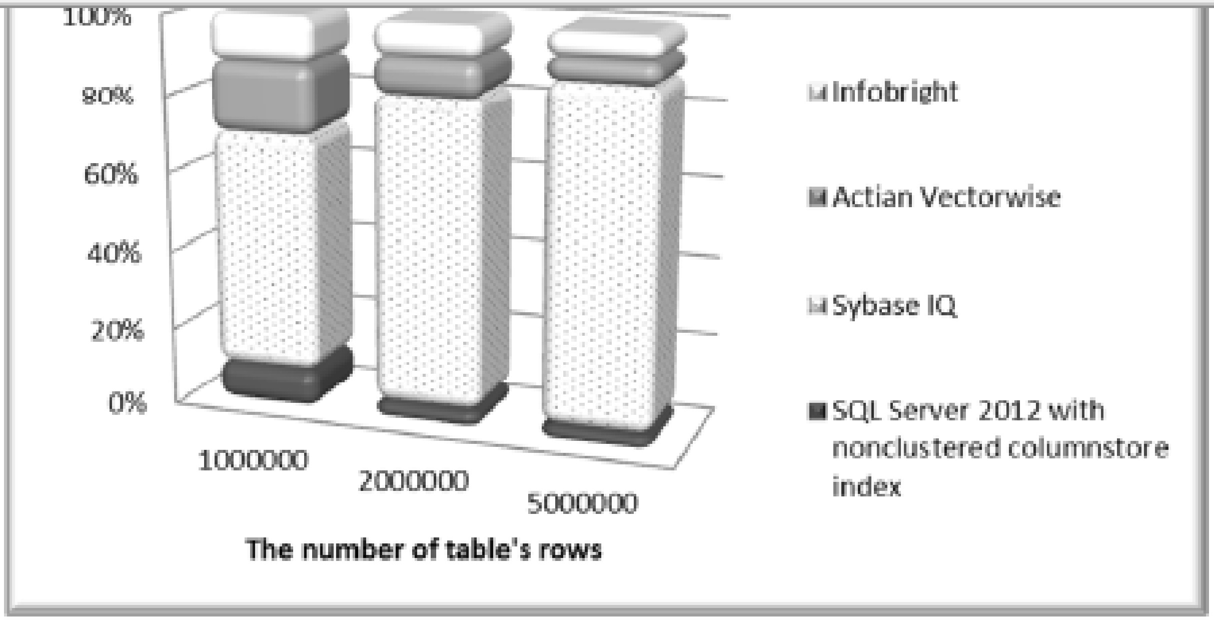
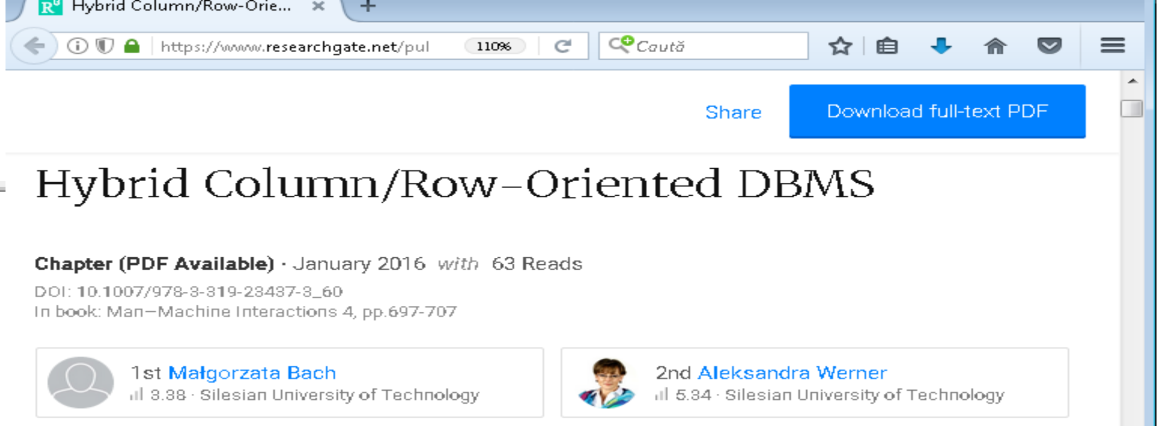


Fig. 1. The cumulative execution time of sample queries for Actian Vectorwise, Infobright, Sybase IQ and MS SQL Server 2012

Taking into account the average result of all made tests, the SQL Server 2012 with the column index turned out to be the best[4] (Fig. 1). The study paid attention to certain limitations which entailed the usage of these types of indexes. Among them the fact that the nonclustered columnstore index was an additional structure that in a significant way could increase the size of the database, was emphasized. The research in this area was presented in [2]. Besides



Hybrid Column/Row–Oriented DBMS

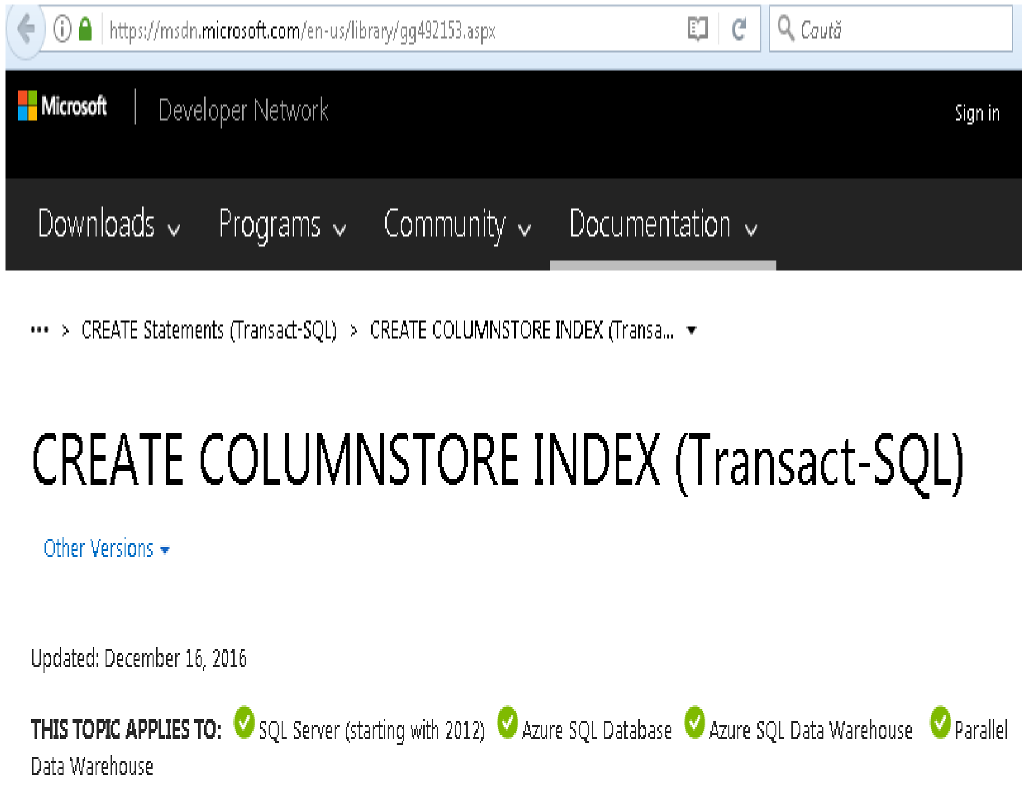
Chapter (PDF Available) · January 2016 *with* 63 Reads

DOI: 10.1007/978-3-319-23437-3_60
In book: Man–Machine Interactions 4, pp.697-707

1st Małgorzata Bach
3.38 · Silesian University of Technology

2nd Aleksandra Werner
5.34 · Silesian University of Technology

https://msdn.microsoft.com/en-us/library/gg492153.aspx


https://msdn.microsoft.com/en-us/library/gg492153.aspx
Caută
Microsoft | Developer Network
Sign in
Downloads
Programs
Community
Documentation
... > CREATE Statements (Transact-SQL) > CREATE COLUMNSTORE INDEX (Transa...
Other Versions

CREATE COLUMNSTORE INDEX (Transact-SQL)

Updated: December 16, 2016

THIS TOPIC APPLIES TO: ✓ SQL Server (starting with 2012) ✓ Azure SQL Database ✓ Azure SQL Data Warehouse ✓ Parallel Data Warehouse

DWH data architecture

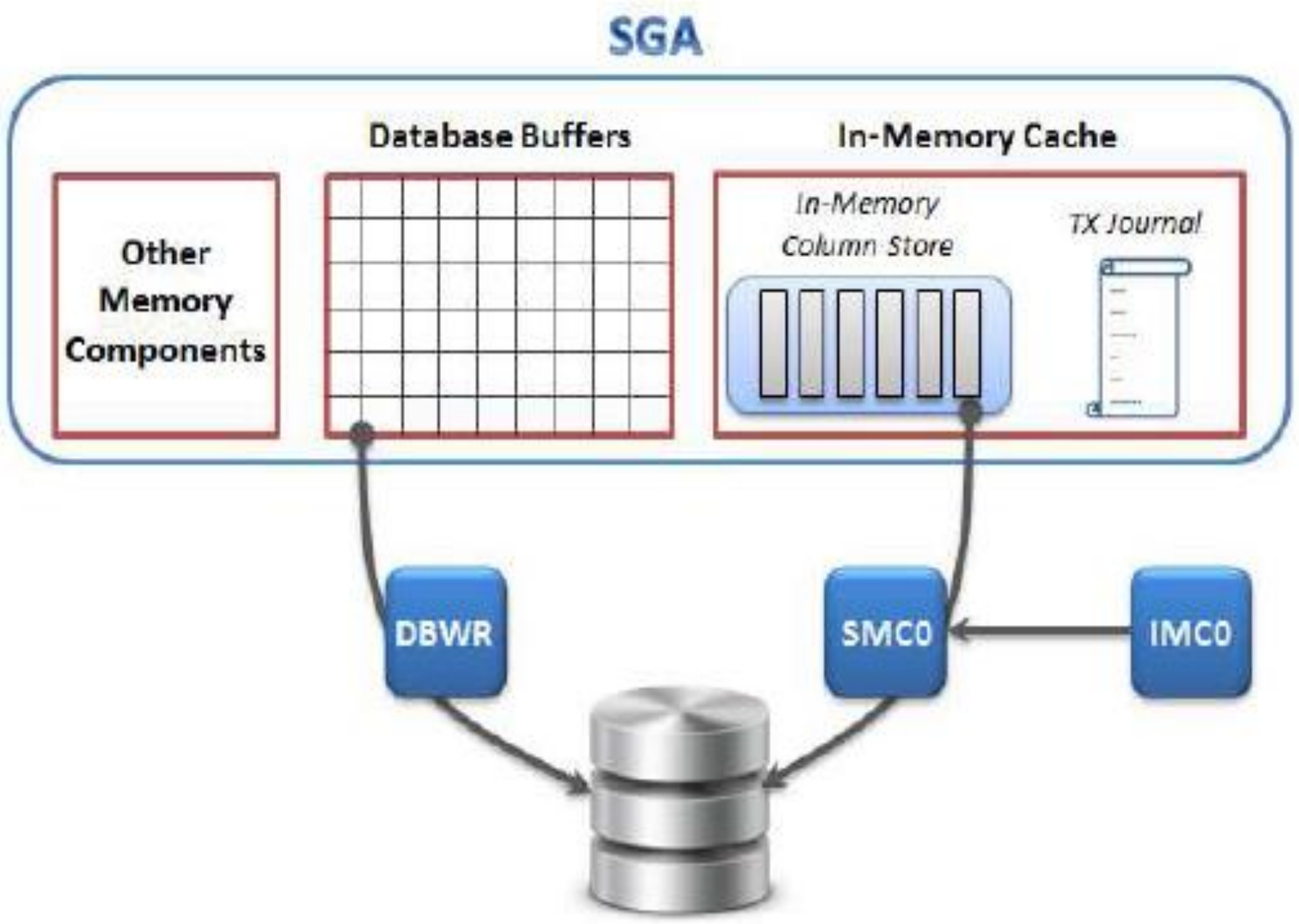
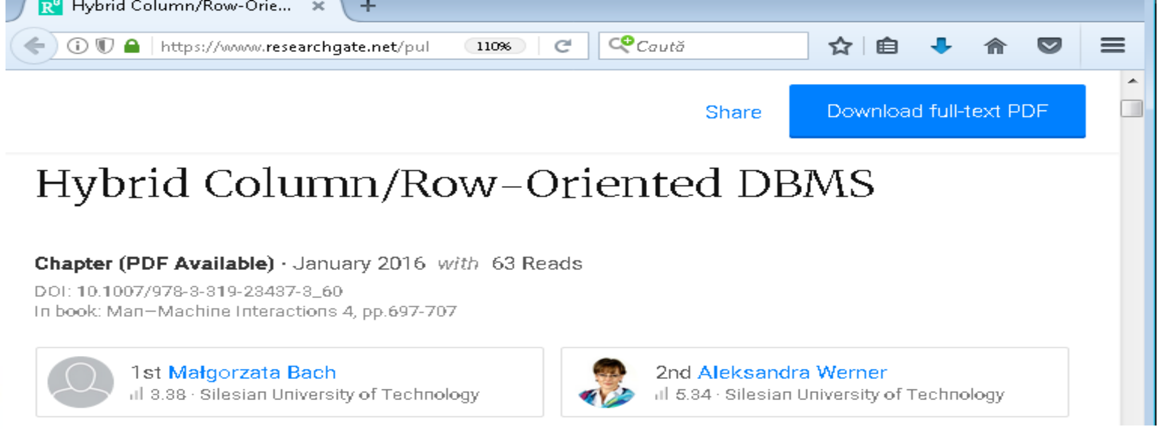


Fig. 2. The In-Memory Structures in the Oracle Instance



Hybrid Column/Row–Oriented DBMS

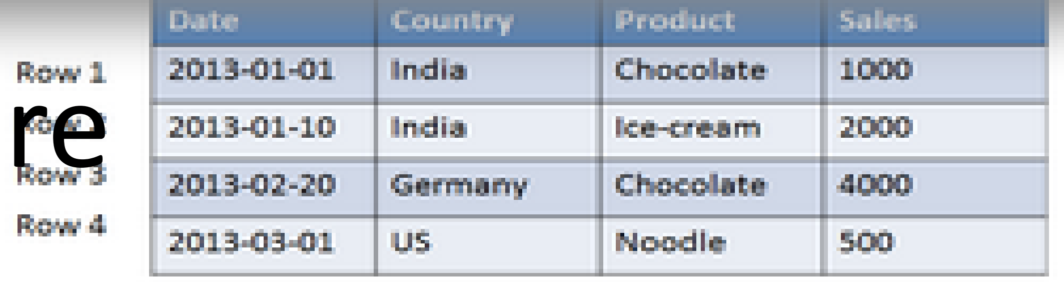
Chapter (PDF Available) · January 2016 *with* 63 Reads

DOI: 10.1007/978-3-319-23437-3_60
In book: Man–Machine Interactions 4, pp.697-707

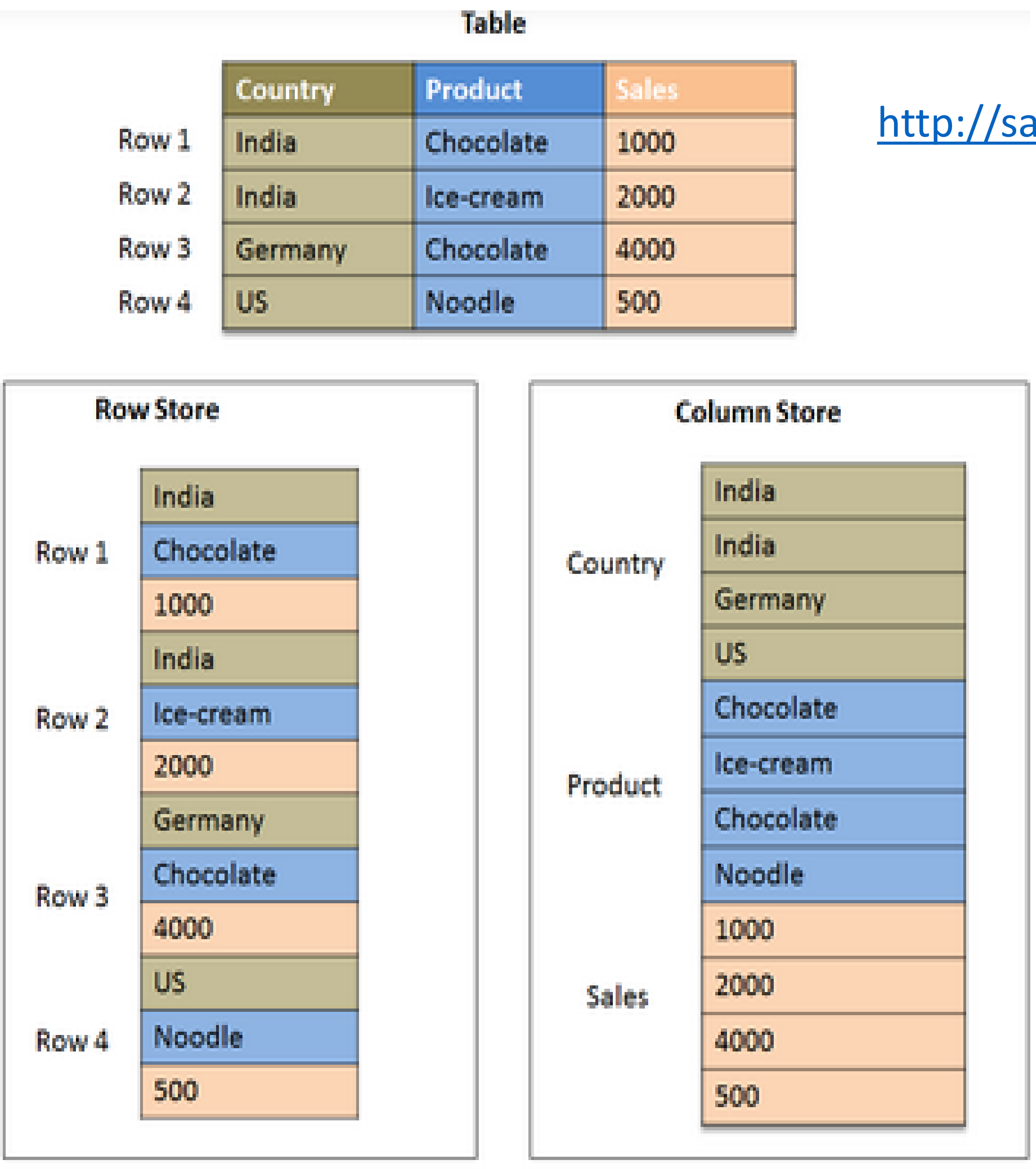
1st Małgorzata Bach
3.38 · Silesian University of Technology

2nd Aleksandra Werner
5.34 · Silesian University of Technology

DWH data architecture



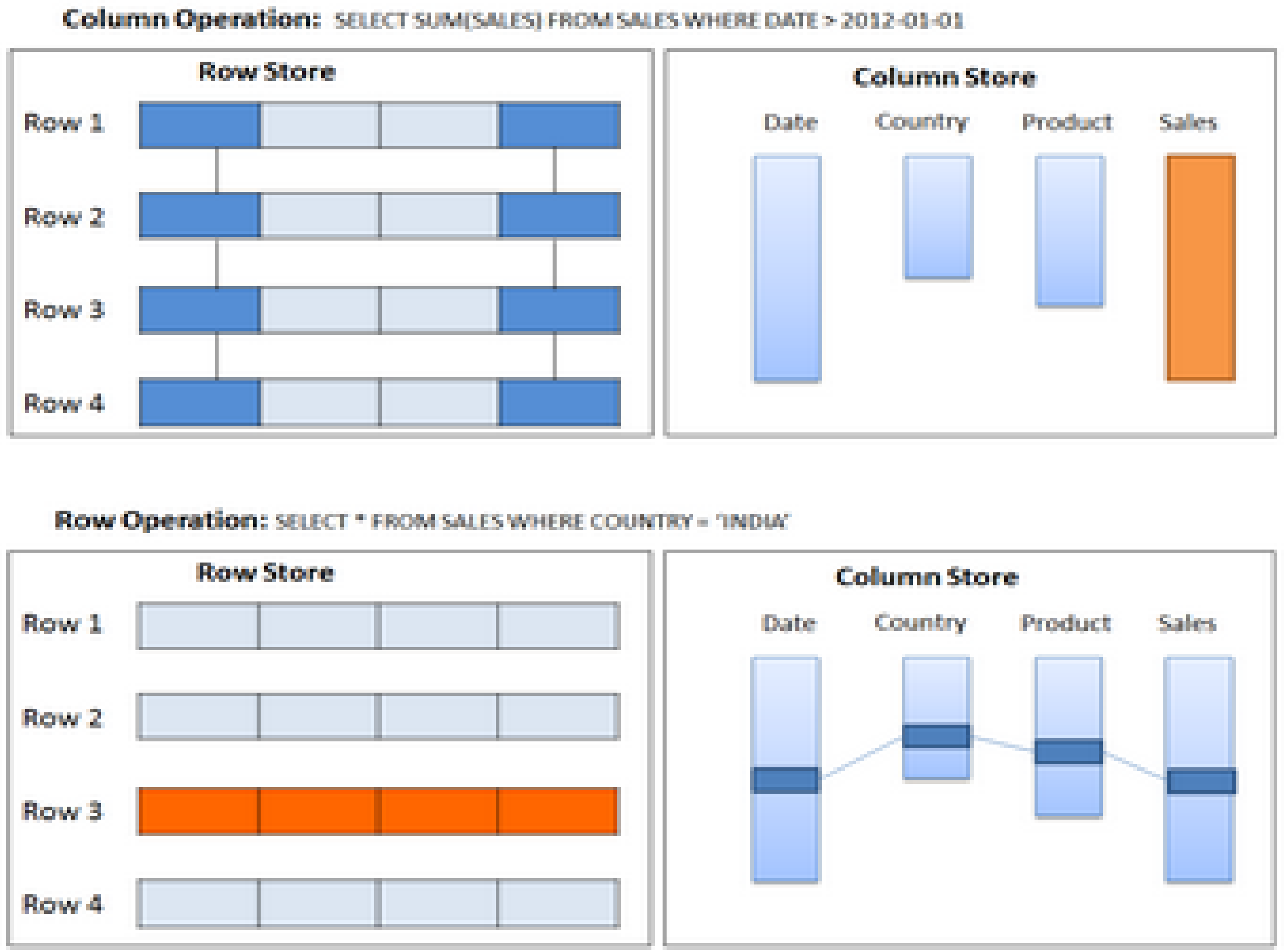
	Date	Country	Product	Sales
Row 1	2013-01-01	India	Chocolate	1000
Row 2	2013-01-10	India	Ice-cream	2000
Row 3	2013-02-20	Germany	Chocolate	4000
Row 4	2013-03-01	US	Noodle	500



Table

	Country	Product	Sales
Row 1	India	Chocolate	1000
Row 2	India	Ice-cream	2000
Row 3	Germany	Chocolate	4000
Row 4	US	Noodle	500

http://saphanatutorial.com/column-data-storage-and-row-data-storage-sap-hana/



Column Operation: SELECT SUM(SALES) FROM SALES WHERE DATE > 2012-01-01

Row Operation: SELECT * FROM SALES WHERE COUNTRY = 'INDIA'

Bibliography

-Airinei, D., Depozite de date, Polirom, Iaşi, 2002
portal.feaa.uaic.ro/Master/sia/Pages/default.aspx
-Airinei, D., Dospinescu, O., Huiban, A., Aplicatii practice cu sisteme OLAP si Depozite de date, Editura Sedcom Libris, Iaşi, 2008
-Homocianu, D., Sistemele de asistare a deciziilor in contextual societatii cunoasterii, Editura UAIC, Iasi, 2009 (ssrn.com/abstract=2384380)
-Inmon,W.H.,Linstedt, D., Data Architecture: A primer for the data scientist, MK, MA, 2015 (tinyurl.com/h7dcq66)
-Kimball, R., Ross, M., The Data Warehouse Toolkit Third Edition. The Definitive Guide to Dimensional Modeling, John Wiley & Sons, New York, 2013 (tinyurl.com/jogo5uy)
-Kimball, R., Ross, M., The Kimball Group Reader. Relentlessly Practical Tools for Data Warehousing and Business Intelligence – Remastered Collection, Second Edition, Wiley, New York, 2016 (tinyurl.com/johox4v)
-Krishnan, K, Data Warehousing in The Age of Big Data, Morgan Kaufmann (MK), MA, 2013 (tinyurl.com/zrjo75j)
-Reeves, L.L., A Manager's Guide to Data Warehousing, Wiley, New York, 2009 (tinyurl.com/htsslk8)
-Sarka, D., et. al., Implementing a Data Warehouse with Microsoft SQL Server 2012. Training Kit, O'Reilly Media, Sebastopol, 2012 (tinyurl.com/jk6lclk)
-Sheldon, B., et. al., Professional Visual Basic 2012 and .NET 4.5 Programming, John Wiley & Sons, Indianapolis, 2013 (tinyurl.com/hrb6j9u)