Georgia Institute of Technology

The Future of Data Warehousing

Pedro Henrique Ribeiro Pinto

CS 6220 - Big Data Systems Analytics
Professor: Dr. Ling Liu

December 2, 2020

Contents

Introduction **2**

A Brief History of Data Warehousing **2**
- Relational Databases . . . 2
- Potential for Data-Driven Decision Making . . . 3
- The Rise of the Data Warehouse . . . 4

Data Warehousing Architecture Overview **5**
- OLTP vs. OLAP . . . 5
- Data Warehouse Properties . . . 5
- Data Cubes . . . 6
- Specific DW Schema Architectures . . . 6
- Extract Transform Load (ETL) . . . 7

Cloud Native Data Warehousing Technologies **8**
- The Rise of Cloud Computing . . . 8
- Snowflake . . . 9

Conclusion **10**

References **12**

Introduction

The purpose of this Technology Review is to take a deep dive into a very important database architecture in the world of Big Data Systems Analytics: the ***Data Warehouse***.

Throughout this review, I will be exploring the history and traditional concepts behind data warehouses, comparing traditional architectures with modern cloud-native alternatives and providing my outlook of the future of the data warehousing industry. The sources used to complete this report are listed in the last page under the References section.

A Brief History of Data Warehousing

Relational Databases

Relational Databases were first introduced by Dr. E. F. Codd in his 1970 publication "A Relational Model of Data for Large Shared Data Banks" while working as a researcher at IBM. This foundational paper established the Relational Model, which became the basis for how computers stored, organized and retrieved data for decades to come.

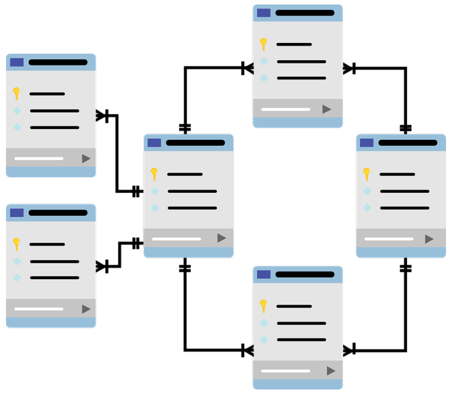


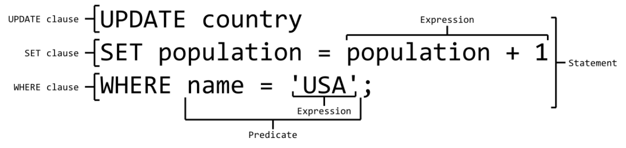
Figure 1: Artistic Representation of a Database Schema following the Relational Model

The Relational Model organizes data across tables, which were referred to as *relations* in that original paper. Here are the main concepts regarding this model:

- **Tables:** Data is saved inside Tables, where the rows represent unique records and the columns represent different attributes.

- **Tuples:** A single record of a table (a row) is referred to as a Tuple.
- **Relation Instance:** A Relation Instance is a finite set of unique tuples in a relational database.
- **Relation Schema:** A Relational Schema is the meta-data describing the entire structure of a database. It is the backbone of any relational database and establishes the relationships before any data is ever written to it.
- **Relation Key:** Each table contains a few important columns called Keys which help to uniquely identify records. These Keys are the basis of how different tables can be connected and exist in two main types:
  - **Primary Keys:** A Primary Key is the unique identifier of a row within a table. It can be composed of a single attribute or a combination of different attributes.
  - **Foreign Keys:** A Foreign Key refers to a Primary Key in a different table.
- **Relation Attribute Domain:** Every attribute (column) has a pre-defined set of values of a certain data type that are allowed.

In the years following Codd's paper, the first working Relational Database Management Systems (RDBMSs) were developed by IBM and The University of California at Berkeley, and, by the end of the decade, various database systems were released to market by companies such as Microsoft and Oracle. These systems are operated using a specific language called Structured Query Language (SQL). SQL is a declarative query language that allows users to manage databases, create schemas and manipulate data using *"English-like"* commands.



```sql
UPDATE country
SET population = population + 1
WHERE name = 'USA';
```

Figure 2: Example of a SQL query

Potential for Data-Driven Decision Making

Relational Databases quickly grew in popularity and most large business soon adopted some commercial RDBMS to store and manage their data. Prior rise of the *World Wide Web*, most entries in a relational database reflected real business operations such as a sale, inventory order, debt payment, etc. Consequently, it became usual to refer to database operations as "transactions".

As database adoption grew, companies started operating multiple databases for different parts of their business (sales, HR, accounting, operations, etc), which large amounts of data. This created an incredible opportunity for companies to start using data to guide organizational decisions. However, there were a quite a few obstacles to efficiently use that data:

- **Query Speed:** Traditional RDBMs were designed to handle transactions. This requires the systems to efficiently store and return single entries of data at a time regardless of the number of requests. When trying to perform Analytics, however, the user will likely require significantly larger amounts of data to be processed at one time but not nearly as often. As traditional databases were not designed for this use case, these types of queries would take a very long time to run.
- **Application Oriented Schema:** As databases were designed to run individual applications or use cases, their schemas reflected that purpose. These schemas are generally very linear and designed with connecting individual transactions in mind instead generating aggregated data.
- **Multiple Data Sources:** Having data living in several different locations and serving different purposes makes it very challenging to integrated it for analytical purposes. This results on what are known as "Data Silos" within an organisation.
- **No Single Version of the Truth:** The main consequence of all this problems is the absence of a unified "Version of the Truth" to guide the decision-making process.

The Rise of the Data Warehouse

With the problems mentioned earlier in mind, the computer scientist Bill Inmon envisioned an alternative to traditional RDBMSs and called it a **Data Warehouse** (DW).

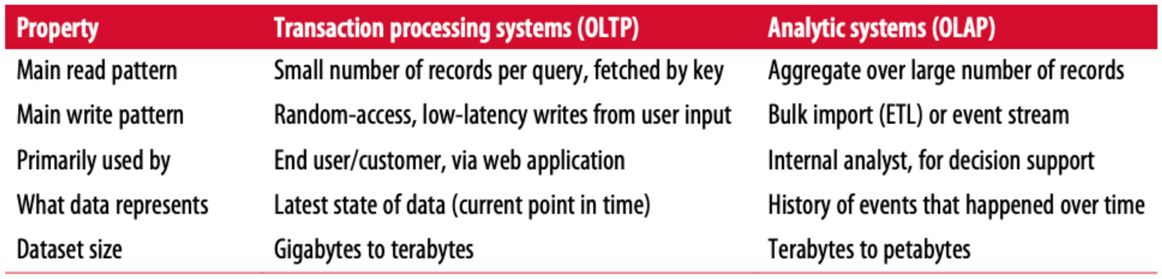
Data Warehouses are relational databases specifically designed to handle analytical operations instead of transactional ones. Everything from structural changes to query optimizers and storage engines to conceptual changes to schema conventions were implemented to adapt to the analytic use case. Quickly companies like Teradata and Oracle developed enterprise level Data Warehousing solutions that gave rise to age the of Business Intelligence.

Data Warehousing Architecture Overview

In this section of the review, we will dive deeper into the overall architecture of data warehousing solutions. These characteristics are followed by both traditional legacy systems as well as modern alternatives that will be explored later in this report.

OLTP vs. OLAP

As previously mentioned, Date Warehouses appeared as an alternative to transactional databases in order to solve analytical problems. This resulted in two distinct database paradigms: **Online Transaction Processing (OLTP)** and **Online Analytical Processing (OLAP)**. The following table, obtained from [6], summarizes the main differences between the two approaches:



Property	Transaction processing systems (OLTP)	Analytic systems (OLAP)
Main read pattern	Small number of records per query, fetched by key	Aggregate over large number of records
Main write pattern	Random-access, low-latency writes from user input	Bulk import (ETL) or event stream
Primarily used by	End user/customer, via web application	Internal analyst, for decision support
What data represents	Latest state of data (current point in time)	History of events that happened over time
Dataset size	Gigabytes to terabytes	Terabytes to petabytes

Data Warehouse Properties

Every Data Warehouse is expected to comply with the following properties:

- **Subject Oriented:** This means that a data warehouse is designed such that analysis can be performed targeting specific subject areas. Typical examples might be sales, operations, customer relationships, etc.

- **Integrated:** To solve the problem of disconnected data silos, the data warehouse guarantees the integration of company-wide data into a single place. However data integration doesn't refer exclusively to where the data resides (data locality) but also on formats, encoding and naming conventions to allow for scalability and easy information retrieval.

- **Non-Volatile:** Unlike transactional databases, Data Warehouses are not updated in real-time. DWs are, instead, updated periodically in bulk through batch processes known as ETL (Extract-Transform-Load). Consequently, the data is not influenced by momentary changes and remains non-volatile.

- **Time Variant:** The data is organized in different time intervals (weekly, quarterly, annually, etc). This allows the organization to quickly analyze behaviors of data across time, such as trends and seasonalities.

Data Cubes

In order to address the Subject Orientation and Time Variant properties of a DW, data tends to be organized on what are called Data Cubes, sometimes also referred to as OLAP cubes. Data Cubes are essentially multi-dimensional arrays of data, where each dimension is a specific attribute (one of the attributes usually referring to time periods).

This architecure makes it simple to navigate different dimensions of a business problem. These cubes contain aggregated (such as sums, means or other statistics) information that has been calculated ahead of time during the data entry process into the DW. Usual dimensions for a cube could be the value in sales different products across different locations and time periods.

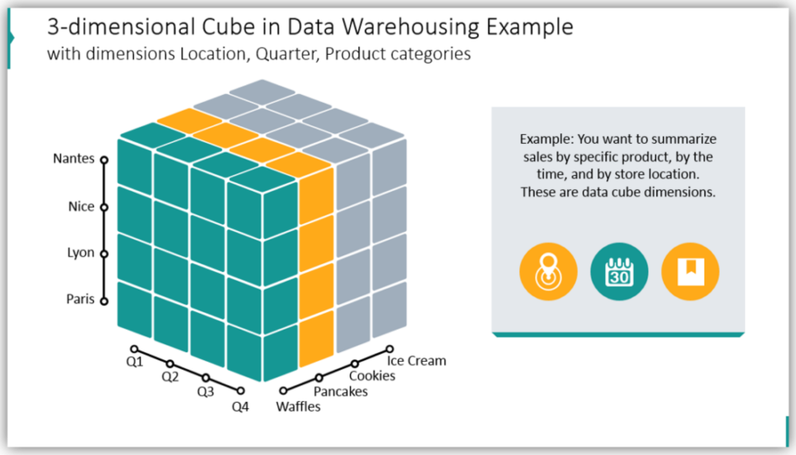

3-dimensional Cube in Data Warehousing Example
with dimensions Location, Quarter, Product categories
Nantes
Nice
Lyon
Paris
Q1
Q2
Q3
Q4
Waffles
Pancakes
Cookies
Ice Cream
Example: You want to summarize sales by specific product, by the time, and by store location. These are data cube dimensions.
30

Figure 3: Example of a Data Cube

Specific DW Schema Architectures

Unlike the traditional linear schema from OLTP databases, OLTP databases tend to be oraganized in schemas specifically designed for analytical purposes. The two main types are the Star schema, where a fact table in the middle is connected to a variety of different dimension tables. The Snowflake schema is an advancement of the star schema where dimensions contain their own hierarchies.

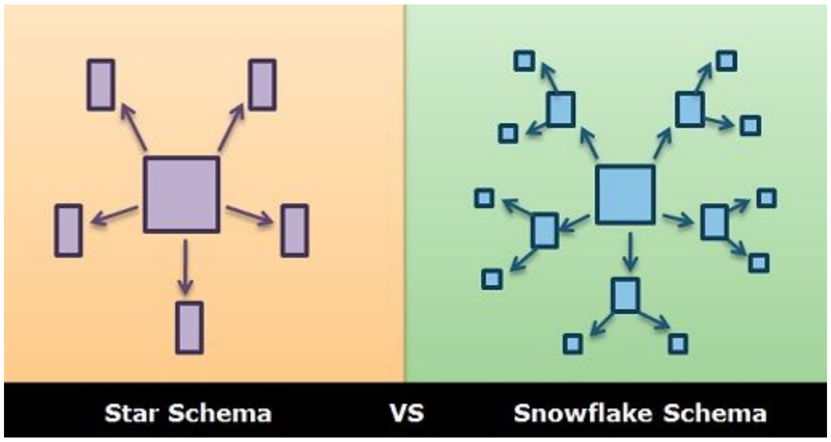


Figure 4: Comparison between the Star and Snowflake schemas

Extract Transform Load (ETL)

As previously mentioned, DWs are not updated in real-time. Instead, the data ingestion process is conducted periodically in a batch fashion, where a large number of records (possibily millions) are processed at once and written into the warehouse. In order to properly organize the ingestion process and guarantee that the data is properly cleansed and aggregated, it follows a pattern called **Extract-Transform-Load**, commonly known as ETL.

- **Extract:** This stage refers to reading the data from its original sources. These sources are usually various OLTP databases running essential parts of the business. Usually there is also a middle step where the data gets temporarily saved into an Operational Data Store (ODS) before being saved into the DW.
- **Transform:** The transform phase is where you convert the raw data into the format and conventions required by the DW. This can involve data cleaning, normalization, aggregation, data matching, etc.
- **Load:** Finally, the data is loaded into the Data Warehouse. After this stage, the date could once again be loaded into smaller data warehouses known as Data Marts. Data Marts contain aggregated data referring to specific parts of the business for even faster querying.

The following images illustrates a typical ETL pipeline in a production environment:

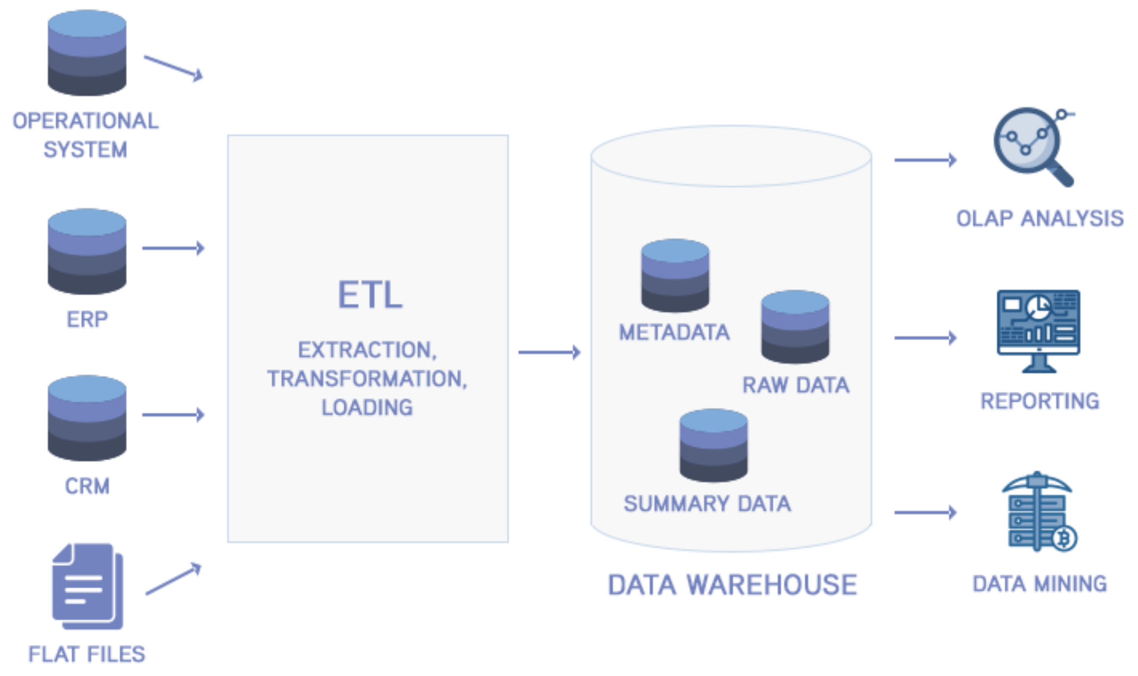


Figure 5: Example of an ETL pipeline

Cloud Native Data Warehousing Technologies

The Rise of Cloud Computing

One the most important innovations in computing over the last twenty years was the advancement of cloud computing. More specifically, when technology giants like Amazon, Microsoft and Google used their resources to build their massive computing clusters and datacenters the entire cloud industry shifted to IaaS (Infrastructure as a service) model.

This means that companies no longer need to build their entire IT infrastructure but can instead rent from one of the several public cloud providers. As a result, this significantly reduced the cost of both computing and storage capabilities in the cloud. Now, there are several new Data Warehousing technologies built specifically for the Cloud such as Amazon Redshift, Google BigQuery and Snowflake. In this report I will be exploring the Snowflake DW Soltuion.

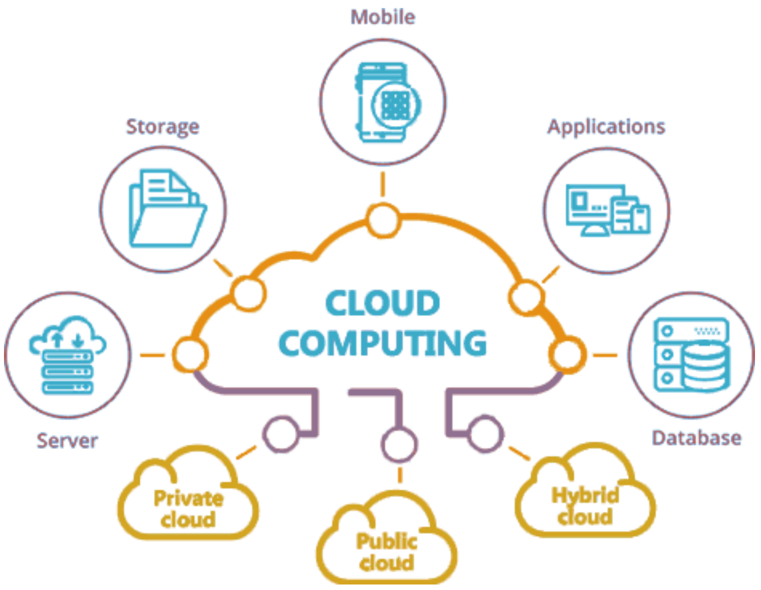


Figure 6: Cloud Computing

Snowflake

snowflake

Founded in 2012, Snowflake Inc. developed an entirely new Data Warehousing solution built specifically to take advantage of the elasticity of public cloud infrastructures.

The main difference between Snowflake and traditional legacy DWs like Teradata, is that it completely separates the computing and data storage layers inside its implementation. By leveraging the auto-scalability of public clouds, these two layers are free to adapt in size based on the current needs of the warehouse. This allows the computing power to automatically grow when bulk analytics are being perform and reduce when in standby to not incur in IaaS bill costs. It also allows for real-time analytics, which is very hard in traditional systems.

Another great advantage of Snowflake is that, in order to create copies of the data for testing purposes, the system doesn't need to fully copy the acutal data. Simply by duplicating the meta-data, you've essentially created a second warehouse without adding any data storage overhead.

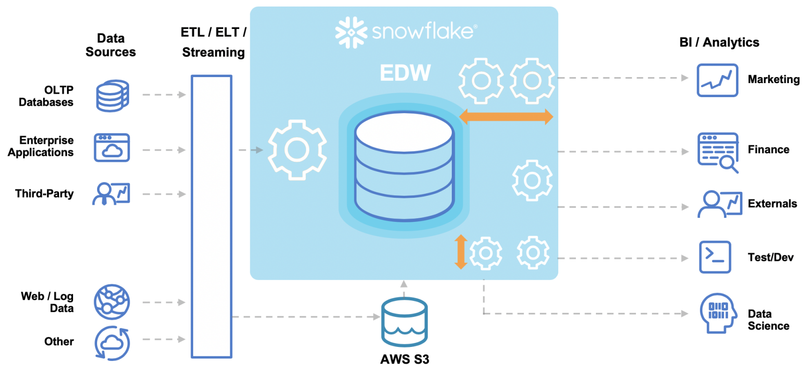


Figure 7: Snowflake Architecture

Conclusion

As can be seen, Data Warehousing technology has been developed for decades and has played a pivotal role in the rise of business analytics. My outcome of the future of the industry is that technologies specifically built for the cloud will dominate the market while traditional legacy systems try to catch up.

References

[1] S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. *ACM SIGMOD Record*, 26(1):65–74, 1997.

[2] C. Coronel and S. Morris. *Database systems: design, implementation, and management*, page 539–591. Cengage Learning, 2017.

[3] J. Hillam. What is a data warehouse? https://www.intricity.com/whitepapers/what-is-a-data-warehouse, Sep 2020. INTRICITY.

[4] J. Hillam. What is snowflake? https://www.intricity.com/whitepapers/what-is-snowflake, Oct 2020. INTRICITY.

[5] J. Hillam. Why a 'snowflake killer' is so hard. https://www.intricity.com/whitepapers/why-a-snowflake-killer-is-so-hard, Oct 2020. INTRICITY.

[6] M. Kleppmann. *Designing data-intensive applications the big ideas behind reliable, scalable, and maintainable systems*, page 69–103. O'Reilly, 2019.

[7] D. L. Liu. Data warehousing: Data summarization technology. Recorded lecture and lecture slides from the CS 6220 - Big Data Systems & Analytics course, Fall 2020.

[8] L. Liu. Data warehousing: Data summarization technology.

[9] D. Wells. The future of data warehousing integrating: with data lakes, cloud and self-service, Sep 2018. Eckerson Group.