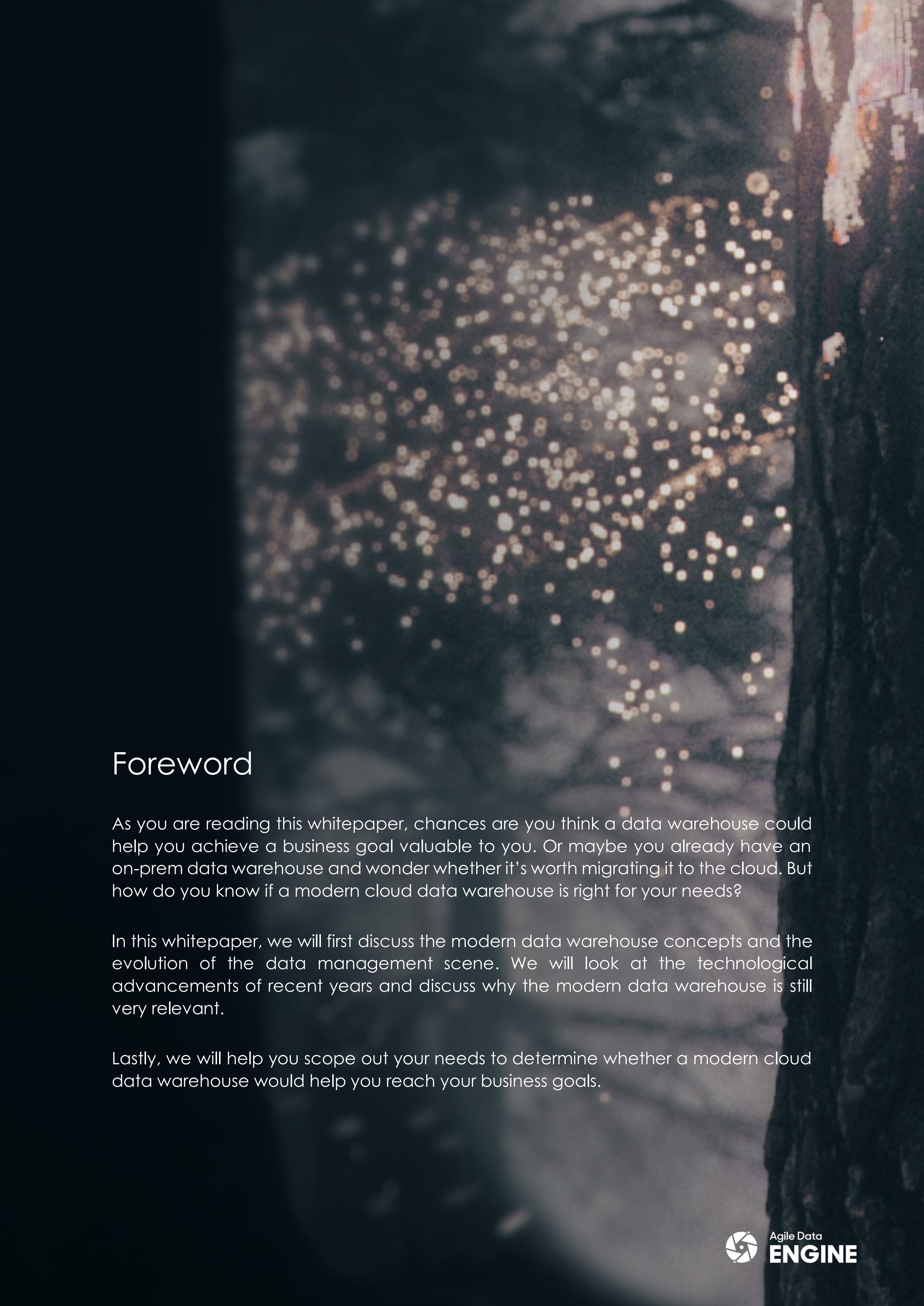
Modern data warehouse and when do you need one?

Agile Data
ENGINE

Table of Contents



Foreword

As you are reading this whitepaper, chances are you think a data warehouse could help you achieve a business goal valuable to you. Or maybe you already have an on-prem data warehouse and wonder whether it's worth migrating it to the cloud. But how do you know if a modern cloud data warehouse is right for your needs?

In this whitepaper, we will first discuss the modern data warehouse concepts and the evolution of the data management scene. We will look at the technological advancements of recent years and discuss why the modern data warehouse is still very relevant.

Lastly, we will help you scope out your needs to determine whether a modern cloud data warehouse would help you reach your business goals.

Related terminology

The data management scene is quite messy today with all the vendor and influencer-imposed terminology. It can be difficult to separate a concept from the commercial offering names used by vendors. Before going further, let's spend a moment on the terminology used today.

In this chapter, we will briefly discuss the term data warehouse and three other concepts that are closely related to it: data mart, data lake and data lakehouse.

Data warehouse

The way we see it, the data warehouse is a concept or a system, not a technology. Still is, even if you add the word modern before it. This is in line with the original definition of a data warehouse by Bill Inmon:

“a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision-making process”

There has been a lot of discussions and disagreement about the pure definitions and best implementation practices in data warehousing. We will not go more deeply into this theoretical discussion here and agree with the fundamental definition by Bill Inmon.

We will dive deeper into the concept, technology, and potential use cases of data warehouse in the following chapters.

Data mart

If you would look at all the data warehouses implemented over the years, many of them do not completely fulfill the Inmon definition. This can be fully fine if the data warehouse creates value and solves your business problems. Perfect is the enemy of good.

Many data warehouse implementations are better described as collections of **data marts**. A data mart can be thought of as a subset of a data warehouse or an isolated solution created for a certain function, department, unit, team, or even one use case. The term is also often used to describe the set of data published for end users of a data warehouse.

Data lake

Another widespread and established concept in analytical data management today is a **data lake**. The original idea of a data lake was to store the data in its raw format, in a massive, easily accessible repository based on cheap cloud storage. It was introduced to overcome the challenges with exploded volumes of data and the variety of data (also unstructured data). It is kind of a continuum from Hadoop-based data storage and made possible with the cheap storage services introduced by public cloud vendors.

A data lake has a somewhat similar purpose to a data warehouse in storing all data in a single repository. But as a concept, it is more loosely defined than a data warehouse, as you can see for example, from this definition by Gartner:

"A data lake is a concept consisting of a collection of storage instances of various data assets. These assets are stored in a near-exact, or even exact, copy of the source format and are in addition to the originating data stores."

Data lakes have been criticized for becoming data swamps, as proper data management and governance principles have not been used in the implementation.

Data lakehouse

The technology implementations of both data lakes and data warehouses have been evolving a lot recently. At the same time, the borderlines between these two concepts have become more ambiguous in practice. One prevailing convention is that a modern data platform must have both a data lake and data warehouse capabilities and one task of a data lake is to be a universal staging area for all the data. There are also combinations of the two original concepts, but these are mostly just commercial productizations of technology vendors.

The most known combined technology concept nowadays is a **data lakehouse**. It is pretty much as it says in the name, a concept that combines elements from both the lake and the warehouse. Technical implementations of a data lakehouse typically evolved from data lake technology and adopted functionalities of a relational database platform.

The concept and technology

About ten to fifteen years ago, a data warehouse was still basically a synonym for an analytical data management platform - at least for many organizations. It was tightly coupled with relational database technology, and in practice, you could not have a proper data warehouse if you did not build it into a relational database platform using SQL. This is not the case anymore. You can implement a data warehouse concept also without an SQL-based database, as there are lots of new technology options available. But should you?

In this chapter, we discuss the relevance of the data warehouse concept and related technologies in today's environment.

2

What makes a data warehouse modern?

You probably have seen the expression 'modern data warehouse' being used on some occasions. The word 'modern' refers to a new kind of data platform that fulfills the analytical needs of the digital age and utilizes new technologies – a new generation of data warehouse.

A modern data warehouse is not really a distinct concept but rather describes a vague time of change. The use of the terms modern or modernization in the context of information technology is a bit silly but understandable when old established IT concepts and technologies are evolving with the changing world.

Are relational databases and SQL still relevant?

If you have been following the analytics and data management scene during the past years, you have probably noticed many claims that the concept of a data warehouse is obsolete. Similar claims have been made about relational databases and SQL as new technologies have appeared. Typically these claims have been presented by technology product (or sometimes snake oil) vendors as they have been promoting their own products and trying to gain market share from established vendors.

Indeed, a lot has happened in the data and analytics landscape during the last ten-plus years. Technological development in the data area has accelerated due to the need to handle and utilize large volumes of data. In addition, the world saw new kinds of data, analytical use cases, and workloads, like real-time analytics and machine learning. Traditional relational database platforms could not handle the volume, variety, and velocity of data and enable efficient analytics.

New and evolving technologies

The new data technology boom was initiated and led by big internet companies. They had to develop their own software and tools to handle big data and disrupt the industry incumbents with new data-driven customer experience. Many new open-source technologies and frameworks were created, leading to new commercial products and concepts; Hadoop, NoSQL databases, and Spark, just to name a few. Public cloud platforms made many of these new technologies easily available as managed services, further accelerating the technology landscape explosion.

For specific workloads, some of the new technologies are clearly better than relational databases. It is increasingly important to identify different use cases and workloads and choose the right technologies for them. But we are far from a situation where SQL and relational databases would be obsolete. In fact, they are back in the game with a vengeance.

The new cloud-native relational database platforms, and also some of the old players, have significantly improved their capability to handle large data volumes. In addition, they can better support the new data variety and velocity. And their costs are following the real usage, so you can get lots of processing power without large up-front investments. Separation of storage and processing is one of the main innovations in new cloud database services, as it enables more versatile and cost-effective scalability.

The promise of public cloud

The public cloud changed the development of analytical and data-intensive solutions and systems. It has democratized access to a variety of software services, processing power, and applications for organizations of all sizes and budgets. It also has had a big impact on data warehousing.

One of the major promises of a public cloud is the elasticity of processing and storage capacity. When you need more processing power, you can easily take it into use, and when you do not need it anymore, you can ramp it down again. In addition, you only pay for what you use, so there is no more need for considerable upfront investments. The value is, in theory, continuously in line with the cost.

The promise of “use and pay for only what you need” is very lucrative when considering data-intensive analytical applications, like data warehouses, as the data volumes are big and usage varies. However, the reality is more complex than this may sound.

Let’s think about motorcycles and cars for a moment. Both are motorized vehicles. You can use both to get from A to B. A motorcycle gives you more speed and agility than your average car but requires different skills and increased awareness to drive one safely. And the impact of possible mistakes can be very high - and sudden.

Just like the fundamentals of road traffic don’t change when you hop from a car onto a motorcycle, the fundamentals of building and managing complicated IT systems remain the same going from on-prem to cloud. Although resources have become more accessible, you still need to know what you are doing when implementing large and complicated data-intensive systems. Because of the speed and agility, getting the basics right is more important than ever when operating in a public cloud environment.

Renaissance of data modeling

Due to the explosive growth in the use of data, the demand for skilled data professionals has grown exponentially and continues to grow with similar pace. This means that a lot of new people have entered the field. People who do not have a history and experience in the discipline of data management, and some people who do not necessarily have even basic skills with databases and SQL. Of course, people learn by doing, but the speed of learning is a challenge compared to the demand. Data management is not a highly standardized discipline, which means that individual experience is a (too) big factor in implementing a good quality data solutions architecture.

A forgotten discipline

Public cloud and other new technologies becoming more common have caused a lot of new dimensions to skillsets and resource consumption. The focus and effort within data platform and analytics development have been divided into many things that take a lot of time and resources. For example, building and configuring cloud infrastructure, scripting, coding own software tools, maintaining open source software, and managing continuous deployment tools and environments. Many of these technical tasks are not directly related to solving data and business requirements, but to enabling the development of itself, and the needed skillset is much more diverse than before. During the last 10 years, building yourself, instead of buying, has been the prevailing approach in data platform development.

With the change in focus and the entrance of new people, the data management expertise and traditions fell on the sidelines. Data management and SQL skills were not seen as important as software development and cloud infrastructure skills (because of demand in the market).

One specific area of data management, which was almost forgotten, was data modeling. In some contexts, it was seen as untrendy for many years. Some people with no data management background saw data modeling as too difficult and slow, or even unnecessary. It has been a quite common supposition that a data model is something you do just one-off for a specific purpose or need. So using a schema-on-read principle and designing application-specific data models. This application-centric approach was the reason why the data warehouse concept was originally created. To overcome the problem of data silos.

Pendulum is swinging

Schema-on-read gives lots of flexibility. But that is the other side of the coin as well. There will be lots of duplication in data models when everybody creates their own models of reality. This will result in situations where there are several truths about the same thing and users will lose confidence in the data. And obviously, the level of reuse and productivity remains poor. When new data solutions are built using duplicate data models, data quality and maintainability inevitably become a nightmare. In the end, we will have a data swamp consisting of piles of spaghetti code.

However, we've seen the tide starting to turn again. Data modeling has now again more and more champions in the industry, being vocal about its importance. Many have witnessed first-hand the problems created when data modeling is skipped totally or done badly. The data model gives structure and meaning to data. It is the interface between data, users, and developers. When implemented well, the data model aligns concepts between the developers and the business and helps to ensure correct output of the data products. It is the necessary enabling tool for integrating data across diverse applications and source systems. Data modeling is at the heart of the data warehouse concept, and also at the data lakehouse concept.

A similar pendulum swing, what has happened with data modeling, is often seen in IT industry. Once in a while, new shiny things come and take over the airspace, and old knowledge is out of fashion. And after a while, the old ideas and concepts come back again with some evolution.

The inevitable change in data warehousing

As we have been discussing earlier in this chapter, the public cloud has brought many changes to data warehousing. The majority of new data warehouses are built on top of cloud database services that are optimized for analytical workloads and use other cloud-based technologies. Many organizations are either doing or planning the migration path from their current on-premises data warehouses to the public cloud. On-premises implementations are still needed in some industries and areas, where there are for example legal requirements that prevent transferring some data to the cloud. Hybrid cloud strategy becomes more common, as companies are utilizing more than one public cloud vendor for their analytical systems. Mitigating the risk of locking yourself with one cloud vendor, is a wise strategy.

The significant and welcomed change that has happened, is that a data warehouse is not anymore serving only management decision-making and reporting needs, but a wide variety of use cases and business contexts. The use of data and analytics has become operational in almost all industries and businesses and data warehouse needs to be up for the challenge. This has created more pressure on requirements like implementation speed and data refreshment rates, which need to be in line with the fast pace of business and continuous change in the environment.

This means that the way of implementing and operating a data warehouse needed to change. Agile methodologies and thinking have become mainstream also in data development, and iterative and incremental approaches have been embraced quite well in the data industry. Agile promotes releasing value to the business more frequently and continuous learning, so solutions and systems will be more adaptable to changes. Data warehouse implementation should adopt good practices from software development and combine them with the data management best practices. Increasing team productivity over the whole lifecycle is a key principle. Automation should be used extensively, which will require standardization.

To help you to succeed in your journey, we have created a set of principles for building and operating a modern data warehouse that will dive deeper into upcoming whitepapers.

The principles for short and long term success with your data warehouse

From project to product thinking

- Do not treat implementation as a project, but a process
- Data as a product
- Data contracts
- Think both short term and long term (lifecycle)

Make it business driven

- Always a business case
- Business ownership of data and solutions
- Model the data from business perspective
- Incremental delivery
- Release fast and often

Design for productivity

- Team productivity, not individual
- Increase reusability
- Increase flow of work
- Continuous improvement
- Balance distributed and centralized approach

Design for change

- Adaptive data (model) architecture
- Resilient technology architecture
- Common practices for teams
- Iterative delivery
- Standardization and automation

Helping you choose wisely

Due to the messiness of the terminology and contradicting opinions in the field, it can be challenging to make decisions about the data platform architecture and technologies. To better understand the basics of the data warehouse concept, we will present some elementary requirement examples you can use to consider the need for a data warehouse. These are in line with the original data warehouse concept definition we presented in the first chapter.

Start with your own needs and objectives with data and analytics. Use our high-level requirements scoping approach to clarify the need for a data warehouse concept as part of your modern data platform.

3

Agile Data ENGINE

How to use the requirements scoping approach

We have created various questions to help you to scope whether your organization would benefit from a modern data warehouse. Questions are oversimplified on purpose and are meant to provide high-level viewpoints on typical analytical data requirements. Obviously, there is no universally correct answer on whether a data warehouse is something you should include in your data platform, as it depends on the unique context, in which you are. The aim of this chapter is to help you to think about it by yourself.

The base assumption here is that you have analytical needs and have data that you need to work with. What you want to do with that data, is the key deciding factor for or against a data warehouse concept. In these examples, the term 'data' refers primarily to structured data, not unstructured like pictures, videos, etc.

You can glance through the questions and find the case that best match your needs. However, we recommend that you read through all the resolutions and reasoning behind them.

Data requirements scoping

Question	Discussion
Are you collecting and storing data just in case?	When you have data that you just need to collect and dump somewhere for possible future use or for archival purposes, by default, you don't need a data warehouse. You don't need an analytical platform either! In a public cloud, the easiest and cheapest way to store data is to put it in a object storage service (for example Amazon S3, Azure Blob Storage / Data Lake, or Google Cloud Storage). This stands for both structured and unstructured data. On some occasions, a relational database service (or other database technology) might be a better option for storing structured data than cloud file storage. Besides costs involved, it depends on the further usage of the data.

Do you need to do stuff (like reporting and analytics) with the data?

When you have straightforward needs to do reporting or some simple analytics with your data, you might need a data warehouse, but not necessarily.

Check first if it’s enough to use the existing capabilities of your operative systems and applications. This might be something you can start with first, to clarify your needs. Many operative systems have quite decent functionalities available for reporting. If not, you could export the data into cloud storage (like in the previous example) or a relational database and do simple analytics from there. Typical first step is however to export data to your own computer and do some excel magic with it.

It is good to understand that if you have data in a relational database, it does not automatically equal a data warehouse. A relational database is a technology used for storing and managing data. A data warehouse is a concept for, not a technology.

Do you have lots of data and need to do complicated stuff?

When you have large volumes of data that you need to do complicated reporting or analytics on it, it is best to take the data out of your operative applications, so your interactions with the data (e.g. complicated queries) would not disturb and impose risks to the operative usage of those applications. You should also do this to achieve better performance for doing complicated data processing for analytical use.

But still, you do not necessarily need a data warehouse. Depends on what kind of use cases and analytical workloads you will do. If you have several different analytical use cases with large data volumes and you need to do those repeatedly, then implementing a data warehouse starts to be a good idea.

Do you need to transform the data to do the stuff?

We would argue that doing complicated analytics already implicitly means that you **do need to do** lots of transformations (business logic) on your raw data.

You can do transformations on demand when you are using the data (e.g. in a business intelligence software, or in other applications), but for performance and productivity reasons it is often wiser to pre-process the data and store the results.

Transforming is not just about implementing the business logic of one particular use case. It also includes doing data model transformations, cleaning the data, joining it with other data, and doing aggregations and statistical calculations. And for that, a data warehouse would help you do it consistently and at scale.

Does the stuff involve comparing data over time or capturing history?

If you need to compare data between different time periods or need to capture the change history of the data, the need for a data warehouse becomes quite strong. If you need to be able to do time variant reporting and analytics continuously in an efficient manner, a data warehouse is a good solution. Especially if you are aiming at enabling self-service analytics for business users (more on that later).

You could of course just take snapshots of your data and store them in cloud storage or a data lake and implement use case specific solutions based on them. But if you try to do that at scale, you can find yourself in an operative nightmare.

A data warehouse contains a dimension of time so that it can be used to study changes and trends. Typically, you cannot trust operative applications to enable this, as they need to be optimized for transaction processing, not analytics.

Do you need data from several separate systems to do your stuff?

If you need to integrate data from several separate systems and do analytics and reporting on the integrated data, you are at the very core of data warehouse's purpose.

A data warehouse is typically designed to be subject-oriented, so it can answer business questions for a certain subject area, like sales, regardless of which system the original data has been created in. This requires an integration of data that are created in different systems and have system-specific data models. These diverse data models need to be transformed into the same consistent data model. This often includes resolving semantical problems, master data issues, and inconsistencies between operative systems. This integration is often the hardest part of building and maintaining the data warehouse.

Do you need to enable self-service for the data and/or stuff?

One task of a data warehouse is to publish the data to end users in a format they can easily use. These end users are not necessarily technical experts and have lots of other duties, so they might not have too much time to dig for the data they would need.

Database structure (data model) of transactional and operational systems can be quite complicated and cryptic, as they are not designed for analysis usage, but for being efficient in inputting the data to the system.

The most common method for publishing data in a data warehouse is to use so-called star schema modeling (dimensional modeling). It is well known and understood methos, so there are a lot of skilled experts available in the market. It is a rather intuitive structure even for non-technical people to use and you can get your users going relatively easily and with some basic training. Of course, besides training the basics about data model, you need to always make sure users understand the data content.

Do you need to understand the audit trail of the data?

When you have a complicated analytical data solution with lots of transformation steps and business logic, you will inevitably have situations when you need to check from where the data is coming and understand why the data is what it is. A well-designed and implemented data warehouse is typically good for this. When you have an audit trail and lineage of data, it adds to the trustworthiness of the system from a business user perspective.

Many domains also have strong external requirements related to audit trail. We claim that a data warehouse implemented with SQL in a relational database is by default easier to understand than a system implemented with Python or other programming languages. At least it is easier to make the data pipeline more transparent. But of course, it depends on how the system is built.

Conclusions

A modern data and analytics platform needs to consist of different capabilities, that can each serve different business needs, use cases, and workload requirements. Therefore also different technologies, tools, and concepts are needed. You need to understand the differences and trade-offs with them to be able to make architecture and technology decisions. There is no silver bullet technology or concept that solves all the data problems and requirements.

The data warehouse concept is still one of the critical capabilities in a modern data platform and the concept is still very much alive. The biggest change is that data warehouses are not, and should not be, anymore serving only management decision-making and reporting needs. The use of data and analytics has become operational in almost all businesses. A modern data warehouse must be able to serve also other than just reporting or traditional business intelligence use cases.

Another evolutionary change is in the way of implementing and operating a data warehouse. It should adopt practices from software development and combine them with the data management best practices. And it needs to utilize automation extensively. We will address these topics in other separate whitepapers.

The first questions to answer are: what business outcomes you are driving, and what kind of use cases and workloads you will have. When you have a clear understanding of this, you can proceed to think about what kind of technical data capabilities you need, and whether a data warehouse is one of the required capabilities.

Agile Data ENGINE

Agile Data
ENGINE