NOVA IMS Information Management School

MGI

Master Degree Program in

Information Management

Hybrid Data Warehouses

A Cloud and On-Premises approach

Renato Alexandre Pires Severiano

Project Work

presented as partial requirement for obtaining the Master Degree Program in Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

HYBRID DATA WAREHOUSES: A CLOUD AND ON-PREMISES APPROACH

by

Renato Alexandre Pires Severiano

Project Work presented as partial requirement for obtaining the Master's degree in Information Management, with a specialization in Knowledge Management and Business Intelligence

Supervisor/Orientador(a): Miguel Castro Neto

Or Co-Supervisors/Co-Orientadores*:* Bruno Jardim

November 2023

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledge the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Renato Alexandre Pires Severiano

Sintra, 27th of November of 2023

ACKNOWLEDGEMENTS

Aos meus pais, um obrigado do tamanho do mundo, por me terem sempre incentivado a fazer mais e melhor e por acreditarem nas minhas capacidades. Devo-vos tudo o que sou hoje e tudo o que consegui atingir.

À minha irmã, um obrigado por todo o apoio que me deu na elaboração deste projeto. Pelo tempo que despendeu a ouvir-me e a reconfortar-me nos momentos difíceis. Sem ela não conseguiria chegar ao final.

Ao Diogo, agradeço por ter estado sempre lá para mim, e por me ter motivado a continuar e persistir perante as adversidades que surgiam. Pelo companheirismo e carinho incondicionais que me deu.

Aos meus amigos, um obrigado, por se interessarem pelo meu trajeto e por quererem sempre o melhor para mim.

ABSTRACT

This project investigates the crucial role of data warehouses in the decision-making processes of business analysts, focusing on the transition from traditional to cloud-based environments in the face of Big Data challenges. The spotlight is on hybrid data warehouses, bridging on-premises and cloud-based solutions, aiming to understand their architectures, motivations, technical approaches, and associated benefits and drawbacks.

In order to achieve the goal of this project, a Hybrid Data Warehouse that orchestrates on-premises and cloud systems with dimension and fact tables available in both locations was conceptualized. In this project were used Microsoft Tools, such as SQL Server, Visual Studio and the Azure Cloud.

Additionally, the project delves into reasons for organizations adopting hybrid data warehouses, including supporting hybrid data architecture, enhancing data accessibility, agility, and cost reduction, always considering the limitations of the scope, regarding the comparability to similar solutions and applicability in real-life scenarios.

Despite the limitations, while serving as a starting point, the project positions itself as a foundation for future research and development. It anticipates inspiring further exploration, optimization, and advancements in hybrid data warehouse architectures, contributing to the evolving landscape of data management in the digital era.

KEYWORDS

Business intelligence; Hybrid data warehouses; On-premises; Cloud architecture; Dimension and fact tables; ETL (Extract, Transform, Load); Replication.

Sustainable Development Goals (SGD):

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

API	Application Programming Interface
CPU	Central Processing Unit
DSR	Design Science Research
DW	Data Warehouse
ETL	Extract-Transform-Load
HDFS	Hadoop Distributed File System
OLAP	Online Analytical Processing
OLTP	Online Transactional Processing
RAM	Random Access Memory
SQL	Structured Query Language
SSIS	SQL Server Integration Services
SSMS	SQL Server Management Services

1. INTRODUCTION

1.1. BACKGROUND AND PROBLEM IDENTIFICATION

Business analysts and decision makers are notably relying on data warehouses for data analysis and reporting. A data warehouse is a centralized repository for storing and managing large amounts of data from a variety of sources, including transactional databases, log files, and social media feeds. Data warehouses are designed to support the efficient querying and analysis of data and are commonly used for tasks such as business intelligence, reporting, and analytics.

Data warehouses differ from traditional databases in several keyways. First, they are typically much larger in scale, with the ability to store and process billions of rows of data. Second, they are optimized for fast query performance, using techniques such as columnar storage and data compression to reduce the amount of disk I/O required for queries. Finally, data warehouses are designed to support a wide range of analytical workloads, including complex queries, data aggregation, and data mining.

Currently, due to the Big Data challenges, the traditional data warehouses are being transferred to cloud-based environments, with unlimited storage resources and internet based secure access from various places (Garani et al., 2019).

Hybrid data warehouses have become an increasingly popular choice for organizations looking to leverage the best of both on-premises and cloud-based data storage and processing. On the one hand, traditional on-premises data warehouses offer several benefits, such as better control over data security and compliance, lower latency, and the ability to leverage existing hardware and software investments. On the other hand, cloud-based data warehouses offer numerous advantages as well, such as greater scalability and flexibility, lower upfront costs, and the ability to leverage the latest technologies and platforms. By combining these two approaches, hybrid data warehouses allow organizations to take advantage of the benefits of both worlds and build a data management infrastructure that is tailored to their specific needs and goals.

Despite the democratization of cloud-based data warehouses, there is not much information regarding how these can be implemented in both a traditional on-premises manner and, simultaneously, on a cloud architecture.

1.2. STUDY OBJECTIVES

On the light of the challenges proposed by BI4ALL, this one specifically aims at the development of a Hybrid Data Warehouse. The main objective of this challenge and project work is to understand and define how an architecture of both on-premises and cloud systems can be orchestrated, with dimension and fact tables available on both locations. In other words, the idea is to conceptualize a Hybrid Data Warehouse.

In this project, the key targets are to provide a comprehensive overview of the current state of hybrid data warehouses, covering key topics such as their motivations and drivers, technical approaches and tools, and benefits and drawbacks. Begins by exploring the various reasons why organizations might choose to adopt a hybrid data warehouse, including the need to support a hybrid data architecture, to

improve data accessibility and agility, or to reduce costs and complexity, identifying the key challenges faced by organizations when building and operating a hybrid data warehouse, and to examine the best practices for addressing these challenges. Then delves into the technical aspects of building and maintaining hybrid data warehouses, including hybrid ETL, hybrid querying and data integration platforms. Finally, we discuss the potential benefits and drawbacks of hybrid data warehouses, including performance, cost, security, and agility, and provide guidance on how to choose the right hybrid data warehouse solution for a given organization.

Several questions are to be considered, namely "What is needed from the warehouse?", "Who will maintain the warehouse?", "What tools do we use?", etc. (Gardner, 1998). Having this in mind, I decided to follow, from the main three groups of warehousing methodologies known, the goal-driven approach (List et al., 2002). Kimball & Ross even propose a four-step approach where one starts by choosing a business process, takes the grain of the process, and identifies dimensions and facts (2013).

As a result, this project aims to achieve and provide a successful implementation of a Hybrid Data Warehouse, running both on-premises and cloud infrastructures, that follows the intended business requirements and specifications.

Ultimately, this work may result in a reliable and optimal solution for the company to apply in its infrastructures or other case scenarios. Additionally, it may extend the knowledge and theory that exists regarding implementation and orchestration of such data warehouses.

1.3. METHODOLOGY

In what concerns the investigation, Design Science Research (DSR) methodologies for Information Systems are to be applied to this project. This is a qualitative research approach in which the object of study is the design process, which undergoes in an iterative manner, providing the chance to refine the artifacts and models, and for theory development (Carstensen & Bernhard, 2019).

DSR involves multiple cycles and must be revisited when changes occur that impact the entire flow. To ensure the success of a project, it is important to consider all five phases: awareness of the problem, proposal, development, evaluation, and conclusion (Vaishnavi et al., 2019).

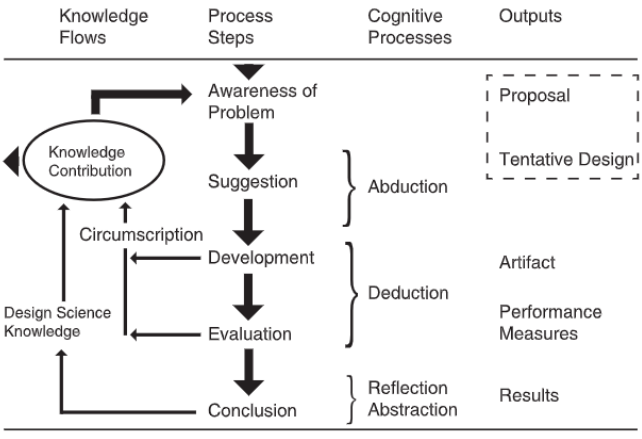

Figure 1.1- DSR Process Model (Vaishnavi et al., 2008)

2. LITERATURE REVIEW

2.1. DATA WAREHOUSES

Data warehousing is known to be a collection of decision support tools that aims at facilitating the making of better and faster decisions by the knowledge workers. There has been constant growth in this area, with the release of newer and more modern products and services, as well as increase in the adoption of these technologies in many industries (Chaudhuri & Dayal, 1997). It consists of the operations of processing incoming data via ETL operations and then storing the processed data into different storage schemas: Data Warehouse or Data Mart Storage.

Typically, data warehouses are maintained in a different infrastructure from the operational databases. This happens because a data warehouse supports OLAP, whose functional and performance requirements are rather different than those of OLTP operational databases. OLTP is commonly used to automatize day-to-day operations that are frequently done such as monetary transactions or order placement. This type of information has to be detailed and up-to-date, easily recoverable and of maximized throughput. On the other hand, OLAP databases are targeted for decision support. These constitute bigger sized databases, with historical and summarized data, of long periods of time. The main task to be performed on these is querying, whose throughput and response times of large amounts of records needs to be maximized.

A data warehouse is characterized by the four following main characteristics:

- Subject-Oriented: Data is organized around specific subjects, such as sales or customer data, to support management decision-making.
- Integrated: Data is integrated from multiple sources, such as transactional systems, to provide a consistent and comprehensive view of the organization's data.
- Non-volatile: Data is stored permanently and is not subject to change, allowing for historical analysis.
- Time-variant: Data is stored in a way that allows for analysis of data over time, such as trends and patterns.

The data warehouse holds granular corporate data, which can be used for various purposes, including anticipating and addressing future unknown requirements. (Inmon, 2005). This granularity is necessary for a data warehouse because it allows for deeper analysis and understanding of the data. It also enables the creation of more detailed reports and enables drill-down capabilities. Additionally, it allows for more accurate forecasting and modeling. However, having too much granularity can also make the data warehouse more complex and harder to manage and maintain.

Despite the fact that DW technologies have existed for a long time, some problems still exist within this matter. Robert Wrembel (2021) focused on some of those challenges, namely “handling the impact of the evolution of data source structures on an integration layer”, “optimizing executions of data processing workflows”, “cataloging available data sets and metadata management” and “assuring high quality of data in a DW”.

Regarding the diversity of warehousing architectures known, there has been for long a discussion on which of them is actually the best one. This disagreement is seemingly most predominant between

data warehousing personalities Bill Inmon and Ralph Kimball, who are at the heart of the discussion. In a study performed by Ariyachandra & Watson (2008), five architectures (independent data marts, data mart bus architecture with linked dimensional data marts, hub-and-spoke, centralized data warehouse (no dependent data marts), and federated) were investigated in order to understand the preferability and success of each one. In the end, the bus, hub-and-spoke and centralized architectures had similar results across the domains of the survey, which helped identifying why all of them have been surviving over time.

2.1.1. Traditional Data Warehouse

When it comes to the actual architecture of data warehouses, the most common ones are the single-tier, the two-tier and three-tier architectures. Nevertheless, the most widely adopted and popular is the three-tier, which creates a well-organized data flow from unstructured data to insightful knowledge.

The databank server, which builds an abstraction layer over data from various sources like transactional databanks used for front-end usage, often makes up the bottom tier of the data warehouse paradigm.

An Online Analytical Processing (OLAP) server is part of the middle tier. This level modifies the data so that it is better arranged for user analysis and diverse probing. We can also refer to it as the OLAP-focused data warehouse because it comes with an OLAP server already integrated into the design.

The client level, which comprises the tools and Application Programming Interface (API) used for advanced data analysis, querying, and reporting, is the third and highest layer. However, because it is frequently not regarded as being as crucial as the other three categories, people hardly ever include the fourth layer in the data warehouse architecture (Fatima, 2022).

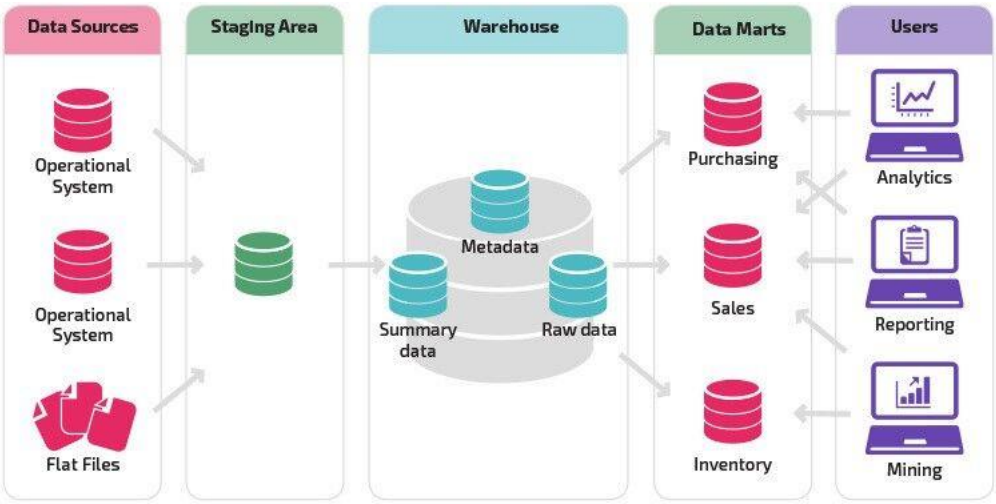

Figure 2.1- Common Data Warehouse Data Flow (Halder, 2023)

2.1.2. Cloud Data Warehouses

A particular method that offers distant processing and data storing is called cloud technology. This method addresses a number of issues that can occur when each customer has a local copy of the data set and software for local data processing and data storing. There are, however, certain drawbacks as well.

Cloud data warehouse facilities have the potential to significantly improve both the client data's accessibility and the network components required for a reliable data transfer and storage. Less than 18% of computer resources and memory are now consolidated across all local processing and storage techniques. The geographical distribution of client applications, their mobility, while simultaneously maintaining the integrity of the data, gives rise to contradictions, which is the need to increase the bandwidth capacity of the existing telecommunication component of distributed data warehouses in a context of increased requirements for their availability, as well as regarding unauthorized access and protection against damage (Shakshova et al., 2018).

The majority of cloud data warehouse architecture contains the subsequent components:

- Clusters, which are extensive sets of nodes;
- Nodes that consist of individual CPUs, RAM and memory for computing resources;
- Partitions that allocate their own disk space and memory on a node.

A cluster with two or more nodes has leader nodes and compute nodes organized within it. The client queries get executed through communication between the leader node and client apps while query execution takes place in compute nodes serving results to the former. Cloud-based data warehouses have similar three-part composition to traditional ones: data sources, storage/computing facilities, as well as consumption methods (Gupta et al., 2015).

In Amazon Redshift, for instance, numerous sources provide data that moves into a computing group comprising several compute notes along with a leading note. Each compute node consists of multiple databases in their slices. The utilization of various BI and visualization tools is applied to retrieve data that is stored in the computing cluster.

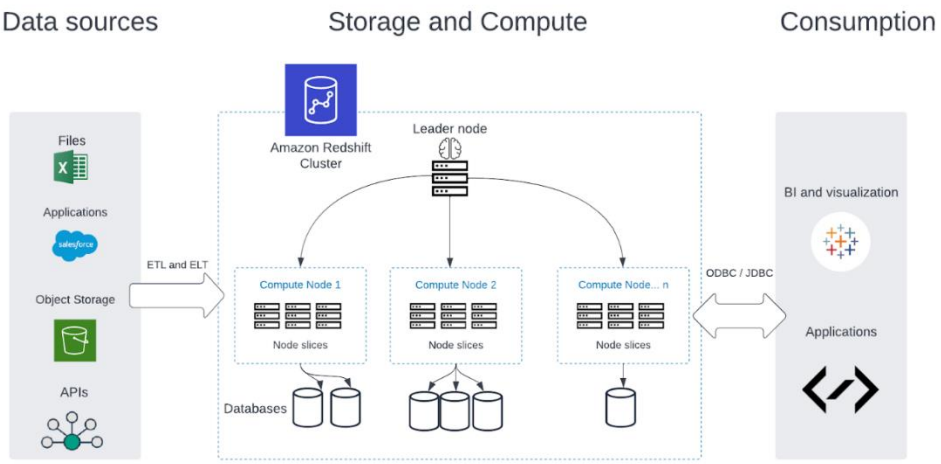

Figure 2.2- Cloud warehouse architecture exemplified with Amazon Redshift (Reno, 2023)

There are many advantages on the adoption of cloud-based data processing tools, such as the significant reducing of costs related to this task as there is no need to acquire custom powerful computing power, flexible expansion and scalability of computing resources, allocation of equipment in dedicated infrastructures and facilities and the constant remote access to these resources that almost never face downtimes (Fisher, 2018).

2.1.3. Comparison between traditional and cloud-based data warehouses

According to Yellowbrick's Key Trends in Hybrid, Multicloud, and Distributed Cloud report, 47% of enterprise IT professionals say their data warehouses are cloud-based, with another 35% using a mix of traditional and cloud data warehouses. While both these approaches to data warehousing can provide the same functionality of ingesting, storing, and serving data, their unique characteristics make each one more or less desirable in different scenarios. Moving to the cloud is the best option for your company if it wants a quick and flexible solution with fewer upfront expenditures. Smaller and medium-sized businesses that lack the staff or financial means to construct and operate internal warehouses are particularly drawn to cloud data warehouses (Reno, 2023).

In table 2.1, some characteristics of cloud-based infrastructures compared to on-premises architectures are addressed.

Table 2.1- Comparison between cloud-based infrastructures and on-premises architectures (Kim, 2009)

Aspect	**On-Premises**	**Cloud-based**
Data storage	Manages only a limited amount of data.	Manages virtually limitless data
Data structure	Best for structured data only	Easily handles unstructured data
Interoperability	Do not have an interoperable layer, making this process more difficult	Have a virtual interoperable layer that enables easy integration of data from different systems
Up-front costs	Require more equipment, costing more	Require little on-premises equipment, costing less
On-going costs	Costly on-going maintenance and upgrades	Require monthly payments to cloud providers which even out maintenance costs
Performance	Diminished performance under high data volumes and high server loads	Faster, where providers ensure 99.9% uptime
Flexibility	Less flexible	Can accommodate a variety of formats and structures

Scalability	Tedious, time-consuming, and expensive	Affordable on-demand scaling
Security	Easier to secure	More vulnerable entry points

When looking specifically for benefits of on-premises solutions, according to Fisher (2019), their main factor of preferability lay on the fact that:

- On-premises is less vulnerable to price increases,
- It is less vulnerable to data leakages or security threats (when there is a mature level of internal security),
- Offer complete control over the tech stack,
- Easier governance and regulatory compliance,
- On the long run, opting for on-premises may occur in a lesser expenditure.

2.2. BIG DATA AND HADOOP

Big Data refers to technologies treating high volume, high velocity, and high variety of datasets that require new forms of processing to enable enhanced decision making, insight discovery, and process optimization. One of the most famous and efficient big data technologies is Hadoop. This framework works to store and process data through specific proprietary algorithms and methods.

In the era of Big Data, Hadoop is an open-source framework which provides the distributed file system. It is used to process the unstructured and large amount of data using MapReduce paradigm. The characteristic of Hadoop is that it partitioned the data in different nodes of cluster. This last provides a large amount of storage for incoming data. Using Hadoop is relevant when there is unexpectedly increasing of data. Cheap computers can be used to create clusters. If one fails, Hadoop continues to run the cluster by distributing work to the other machines in the cluster, which prevents data loss and interruption of work. By dividing incoming files into units, known as "blocks," and storing each block redundantly across the pool of computers, HDFS handles storage on the cluster. In the common case, HDFS stores three complete copies of each file by copying each piece to three different servers (Bhosale & Gadekar, 2014).

The MapReduce paradigm is based on parallel and distributed algorithms for processing especially large datasets from different types (structured, unstructured and semi structured). In a conventional data warehouse, this can require doing an ETL operation on the data to create something the analyst can use. These kinds of operations are created as Java MapReduce jobs for Hadoop (Senna et al., 2014).

Higher-level programming languages like Hive and Pig make it simpler to create these programs. These jobs' results can be stored in a conventional data warehouse or written back to HDFS. There are two functions in MapReduce as follows:

- **map** – the function takes key/value pairs as input and generates an intermediate set of key/value pairs.

- **reduce** – the function which merges all the intermediate values associated with the same intermediate key.

2.3. HYBRID DATA WAREHOUSES

There is not a specifically established concept or definition for what a Hybrid Data Warehouse is, which means that the term *hybrid* can actually stand for a multitude of different aspects within the development of data warehouse systems and architectures.

For example, Aljanabi et al. (2017) proposed a hybrid model in the sense that data could be stored in three different ways to improve the performance of mining algorithms: detailed, summarized and highly summarized.

Raisinghani (2016) proposed a hybrid approach for the design and modelling of data warehouses, in what regards the use of the entity-relationship model and the dimensional model, as a method to overcome the limitations of each of these models.

For Rajdeep & Bikramjit (2010), the hybrid model was defined in what concerns the architecture that is followed, being the most common ones the *Top Down* and *Bottom Up.* Their proposal resulted in a Hybrid Architecture and Implementation that merges features from all existing models and enhances them.

Kavishwar & Pande (2021) mention hybrid DWs in terms of development and methodology approaches, being the common ones Data-driven, Goal-driven and User-driven. Their hybrid multi-dimensional data model is a combination of the data-driven methodology with the business-driven which results in the goal-driven methodology.

For Chen & Zu et al. (2015), their research on hybrid data warehouses comes closer to the purpose of this project, as they developed a hybrid data processing engine that pretends to fully integrate backend systems, whose data is distributed across several locations. This engine is focused on the optimization of the amount of data movement.

In the research of El Houari et al. (2017), the proposal on the hybrid approach boils down to the usage of both ETL and MapReduce methods in the process of building an on-demand dimensional big-data data warehouse, that enables enterprises to process data in an efficient and effective way according to their needs.

Nevertheless, in the context of this project, the main purpose of a *hybrid data warehouse* implementation is, as referred previously, to construct an architecture that enables a traditional data warehouse and a modern cloud system to work seamlessly in symbiosis with one another.

Hybrid data warehouse architectures integrate on-premises and cloud infrastructure to meet the diverse needs of organizations. By adopting both traditional and cloud-based approaches, these architectures effectively manage budget, technical requirements, and data volume. This enables businesses to reap the benefits of both components while ensuring that sensitive information is handled securely through the on-premises component (Hurwitz & Kaufman et al., 2012).

The flexibility and scalability offered by the cloud-based element are invaluable in dealing with changes in processing power or fluctuations in data volumes. Popular examples of such services include Amazon Redshift or Google BigQuery used extensively for Cloud-based Data Warehousing solutions today.

Furthermore, the hybrid data warehouse architecture enhances reliability and disaster recovery capabilities by enabling data replication across multiple locations. This architecture also enables organizations to adopt a more agile approach to data management, facilitating quick responses and adaptability to changing business requirements and market conditions (Coyne et al., 2018).

However, despite these benefits, there are also some challenges associated with hybrid data warehouse architecture. One of the main challenges associated with hybrid data warehouse architecture is the complexity involved in integrating different technologies and ensuring seamless communication between them. Another challenge associated with hybrid data warehouse architecture is the need for specialized skills and expertise to manage and maintain such a system. Managing a hybrid data warehouse demands proficiency in both storage solutions. (Mukhopadhyay, 2020).

3. METHODOLOGY

3.1. RESEARCH DESIGN

As stated initially in previous sections, this project is set to follow the Design Science Research (DSR) Methodology, with the intent of generating the proposal of a hybrid warehousing model and data architecture. Such result may be achieved by studying the current paradigm and trends of data warehouse development to achieve a final combination of both traditional and cloud warehousing systems.

Beneath the application of the DSR Methodology, the qualitative research methodology is going to be followed, which involves gathering and analyzing data that is non-numerical, in order to comprehend different concepts and opinions. This method is said to collect in-depth knowledge into a problem and generate new ideas for research. It is different from quantitative research, as the latter is based on the collection and analysis of numerical data (Streefkerk, 2023).

Inside the qualitative research methodology, there are several approaches, which emphasize different objectives and perspectives. In the context of this work, the Grounded Theory approach will be utilized, where researchers gather rich data about a subject and inductively develop theories. According to Charmaz (2010), Grounded theory incorporates empirical checks into the analytical process, prompting researchers to explore all possible interpretations of their findings. It has become one of the most frequently employed and highly regarded qualitative research methods across various fields of study.

Inductive research approaches aim at developing a theory, moving from specific observations to broad generalizations. This is a common procedure to follow when there is little information and literature on a topic, which is the case with the research question of this work. Nevertheless, one of the limitations of this procedure is that conclusions drawn from inductive methodologies cannot be fully proven (Bhandari, 2023).

Given all the aforementioned, the output of this project will be achieved by researching how can both on-premises and cloud warehouse systems be built and designed individually, so that a proposed mixed design is possible to generate, integrating both parts.

3.2. DATA COLLECTION

It is important to note that the proposed project may not yield tangible outputs that can be utilized in a real-world scenario. Instead, the project's focus is on developing a theoretical approach and proposal for a hybrid data warehouse system architecture. The primary objective is to advance knowledge in the field of data warehousing through the creation of a conceptual model that can be used as a foundation for future practical implementations. While the practical implications of the project may be limited, the theoretical contributions can have significant value for researchers and practitioners in the field of data warehousing.

The proposed project will not have direct input from a company or access to real-world scenarios. Despite this limitation, it is possible to list several key data that, according to Inmon (2010), would

need to be collected if the proposed hybrid data warehouse system architecture were to be implemented practically:

- Business requirements, which include understanding the business needs and goals of the organization. This data can be collected by conducting interviews or surveys with key stakeholders and decision-makers. In the context of this project, there is awareness for the need of all dimension and fact tables to be made available across both the on-premises and cloud infrastructures simultaneously,
- Data sources, by identifying the data sources that the organization uses, such as transactional databases, web logs, social media, and other sources. This information may be collected by reviewing documentation and interviewing technical staff,
- Data types and structures used in each data source. This includes data formats, schemas, and data models,
- Data volume and velocity of data generated by each data source, including the number of records, growth rate over time, access patterns across different time frames,
- Data quality in each data source, such as completeness, accuracy, consistency, and timeliness,
- Technical infrastructure, components and configurations that are used to store and manage data, such as hardware and software,
- Performance metrics such as query response time, data loading time and system availability.

3.3. DATA/DESIGN ANALYSIS

Given that the objective is to propose an architecture of hybrid data warehouse systems, there is not a need to analyze existing data. Rather, a design analysis must be developed to leverage an architecture that meets the requirements of the proposed system.

According to Kimball (2013), to create a hybrid data warehouse, one may start by defining the requirements for the system. These include identifying the types of data that will be stored, the sources of that data, its expected volume, and performance and scalability requirements.

Afterwards, it is important to determine which components are needed to support these requirements. These components include data storage systems (e.g., database systems), integration tools, and processing tools that will be needed to meet each requirement. Once these components and their associated requirements have been identified, an overall architecture for the system must be defined based on those components, as well as existing or proposed standards.

Once the architecture has been defined, it must be evaluated to ensure that it meets the requirements. This might involve simulating different scenarios, conducting performance tests, and assessing the scalability of the system. The findings must be documented in a report or functional specifications document.

3.4. HYBRID DATA WAREHOUSING AND REPLICATION

For the context of building a data warehouse where all dimension and fact tables are stored in both the cloud and on-premises locations, the data must flow to these two endpoints at a given point in time. For this, different set of techniques can be used, such as Hybrid data warehousing or Replication.

Hybrid data warehousing and Replication are related techniques and concepts, which are, however, applied to different purposes. The first is a technique that intends to combine on-premises and cloud-based data warehousing architectures, in order to create a single and integrated warehouse. With this approach, entities can take advantage of the best of both solutions, while responding to the limitations of each one. This approach provides many benefits, including scalability, elasticity, cost reduction and security.

On the other hand, replication is a technique that is used to copy data from a data warehouse to another. It is commonly used to generate backup copies of information, to increase data availability and redundancy, or to enable sharing between different applications and systems. It can be performed in batches or in real-time, accordingly with the system requirements (Coyne et al., 2018).

In a hybrid warehousing environment, as the one being discussed in this project, replication can be used to move data from the on-premises component to the cloud-based location, or vice-versa. This may aid in ensuring the consistency and availability of data, and in enabling data sharing across different parts of an organization.

3.5. TOOLS TO BE USED

In order to build the prototype for this Hybrid architecture, some tools and software must be used to enable such output. These items are essential in different parts of the ETL process, and can be replaced with equivalent ones, as this project is focusing on using Microsoft Tools.

- **SQL Server Management Services (SSMS)** is a comprehensive platform designed to manage SQL infrastructures such as SQL Server and Azure SQL Database. It provides tools for configuring, monitoring, and administering SQL Server instances and databases. SSMS allows you to deploy, monitor, and update data components used by your applications, write queries and scripts, and efficiently handle databases and data warehouses regardless of their location.
- **SQL Services Integration Services (SSIS)** is specifically designed for creating powerful data integration and transformation solutions at an enterprise level. It addresses complex business challenges such as file copying, data warehouse loading, data cleansing, and mining, as well as managing SQL Server objects and data. SSIS serves as a comprehensive toolset for handling various data-related tasks within an organization.
- **Visual Studio**, with the SSIS Extension, enables the creation of high-performance data integration and workflow solutions, including extraction, transformation, and loading (ETL) operations for data warehousing.
- **Integration Runtime** is the compute infrastructure used by Azure Data Factory and Azure Synapse pipelines. It provides data integration capabilities across different network environments, including executing data flows in a managed Azure compute environment and copying data across data stores in public or private networks. It supports connectors, format conversion, column mapping, and efficient and scalable data transfer.

- **Azure Synapse Analytics** is an enterprise analytics service that accelerates insights across data warehouses and big data systems. It combines SQL technologies used in enterprise data warehousing, Apache Spark technologies for big data processing, and Azure Data Explorer for log and time series analytics.
- **Azure SQL Pool,** formerly known as Azure SQL Data Warehouse, is a cloud-based relational database service designed for large-scale data warehousing and analytics workloads. It offers scalability and high performance for processing analytical workloads in the cloud, allowing organizations to leverage Azure's capabilities while benefiting from the flexibility and scalability of a cloud-based service.
- **Azure Data Factory** is a powerful cloud based ETL service that enables efficient and scalable data integration and transformation. It provides a user-friendly interface, comprehensive monitoring capabilities, and seamless integration with existing SSIS packages. With its serverless architecture and managed services, Azure Data Factory simplifies the management and execution of data integration workflows in the cloud.

3.6. VALIDATION

The conceptual model that outputs from this project must be validated. Once the architecture for the hybrid data warehouse system has been defined, a prototype of it can be developed. This translates to implementing a small-scaled version of the system and testing its functionality.

This prototype may then be evaluated to determine whether it meets the requirements. Testing its performance, scalability and usability enable this assessment.

Lastly, based on the results of the prototype evaluation, there may be the need to refine the architecture to improve its functionality. Previous steps in the methodology must be revisited to apply modifications to the architecture (Maxwell, 2013).

3.7. LIMITATIONS

The main limitation of the proposed project is that its validation may be restricted due to the absence of real-world use cases for this hybrid data warehouse system architecture. As a result, the proposed architecture will rely on induction from existing information, rather than actual empirical testing in real-world scenarios. This may limit the ability to establish the architecture's effectiveness and potential shortcomings under various real-world contexts.

While the proposed architecture is coherent with best practices and existing literature in the field of data warehousing, the lack of direct validation through practical application is a significant limitation to its overall reliability and generalizability. Therefore, future research may be necessary to validate the proposed architecture through empirical testing in real-world scenarios.

4. CONCEPTUAL MODEL

4.1. MACRO-VIEW OF THE DATA FLOW AND ENVIRONMENT

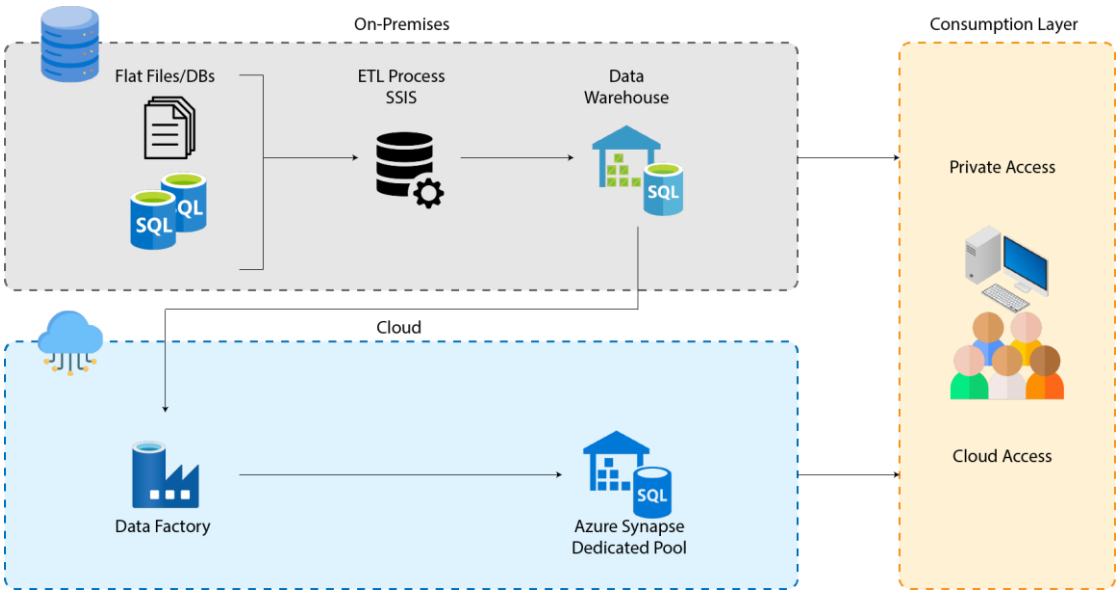

Figure 4.1- Macro View of the Hybrid environment Data Flow

Despite having different possibilities for achieving the proposed objective for this project, this diagram represents the outcome of the practical work on the topic, using mostly Microsoft Tools. When looking at this data flow from a low-depth perspective, these are the steps that constitute the processing of this hybrid environment:

1. First, the data is sourced from different sources, such as Flat files or existing OLTP Databases,
2. This data is ingested and is transformed according to the business needs and governance policy, and also to turn it adequate for analysis. This step includes a staging area that prepares the data to be inserted into the data warehouse.
3. After this process end, the data is ready to be loaded into the data warehouse.
4. Once the data warehouse is loaded with new data, the process moves to the cloud, as the Data Factory is triggered and starts importing a replica of the tables into the cloud location.
5. The Data Factory then feeds the Azure SQL Pool (formerly Azure Data Warehouse) with the imported records, keeping an identical structure as to the one found On-Premises.
6. Finally, at the consumption layer (client level), the data can be retrieved from both locations, with suitable tools for each respective endpoint, and the same results are to be expected from such retrieval.

4.2. DATA MODEL USED AS SAMPLE FOR THIS PROJECT

In order to demonstrate, in practical terms, the functioning of the proposed hybrid data warehouse model, it was necessary to have in hands a data model with several objects and a few records. It was decided to consider business processes such as Sales Order Processing and Point of Sale Transactions.

Hence, the following model was generated which features classes that could belong to a common retailer in the market containing information regarding *Orders*, *Products*, *Shippers*, and *Suppliers*. It consists of five tables, as the relationship between *orders* and *products* is many-to-many, unlike the rest of the keys which work in a one-to-many cardinality.

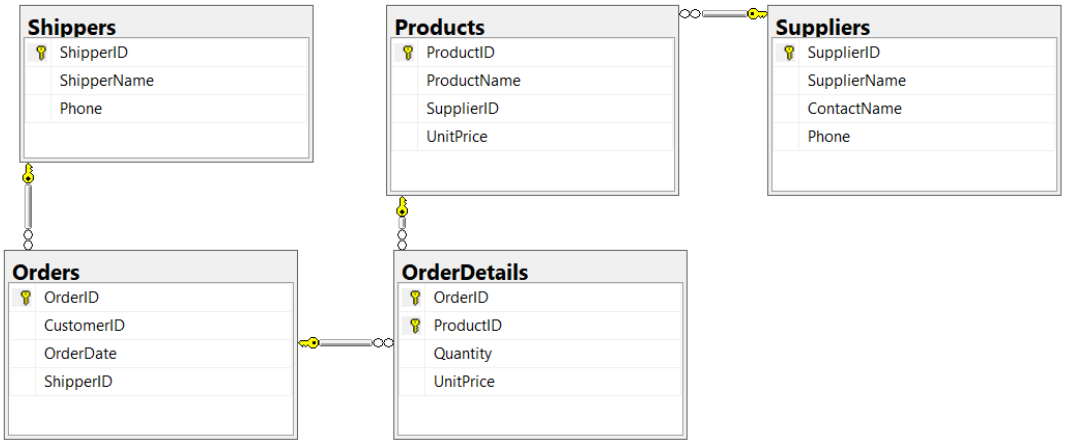

Figure 4.2- Relational Data Model generated for demonstration.

As for the test information that is used, the tables were populated with the records in figures 4.3 to 4.7:

	ShipperID	ShipperName	Phone
1	1	Speedy Express	(503) 555-9831
2	2	United Package	(503) 555-3199
3	3	Federal Shipping	(503) 555-9931

Figure 4.3- Shippers table

	ProductID	ProductName	SupplierID	UnitPrice
1	1	Chai	1	18.00
2	2	Chang	1	19.00
3	3	Aniseed Syrup	1	10.00
4	4	Chef Anton's Cajun Seasoning	2	22.00
5	5	Chef Anton's Gumbo Mix	2	21.35
6	6	Grandma's Boysenberry Spread	3	25.00
7	7	Uncle Bob's Organic Dried Pears	3	30.00
8	8	Mishi Kobe Niku	4	97.00

Figure 4.4- Products table

	SupplierID	SupplierName	ContactName	Phone
1	1	Exotic Liquids	Charlotte Cooper	(171) 555-2222
2	2	New Orleans Cajun Delights	Shelley Burke	(100) 555-4822
3	3	Grandma Kelly's Homestead	Regina Murphy	(313) 555-5735
4	4	Tokyo Traders	Yoshi Nagase	(03) 3555-5011

Figure 4.5- Suppliers table

	OrderID	ProductID	Quantity	UnitPrice
1	10248	1	12	14.00
2	10248	2	10	9.80
3	10249	3	9	18.60
4	10249	4	40	42.40
5	10250	5	10	7.70
6	10250	6	35	42.40
7	10250	7	15	16.80
8	10251	1	15	15.60
9	10251	7	20	16.80
10	10251	8	6	16.80

Figure 4.6- Orders table

	OrderID	CustomerID	OrderDate	ShipperID
1	10248	VINET	1996-07-04	3
2	10249	TOMSP	1996-07-05	1
3	10250	HANAR	1996-07-08	2
4	10251	VICTE	1996-07-08	1

Figure 4.7- OrderDetails table

4.3. Procedure for Creating the Hybrid Environment

4.3.1. Installing SQL Server Management Studio (SSMS) and SQL Server

The first step to enable this prototype was to install SSMS and SQL Server in the on-premises hardware. Both these tools allow for the creation of the correct SQL environment and development of the local database and data warehouse.

Figure 4.8- Microsoft SQL Server applications

4.3.1.1. Generating the Database

With the purpose of generating the data model for the previously designed database (Fig. 4.2), a SQL script had to be written in accordance with it. For this, a database named HYBRID_DB was instantiated, and the tables built upon it. (Annex 1)

4.3.1.2. Generating the Staging Area

With the purpose of generating the staging area model for the hybrid data warehouse, a SQL script had to be written, but with the necessary adequations to make it suitable for a Data warehouse environment. For this, a star-schema was designed, and a database named HYBRID_STG was instantiated, and the tables built upon it. (Annex 2) Additionally, two log tables were created, in order for the ETL process to register the events, such as the start and end of the process, errors that occur, among others. The result of this script was the model in figure 4.9:

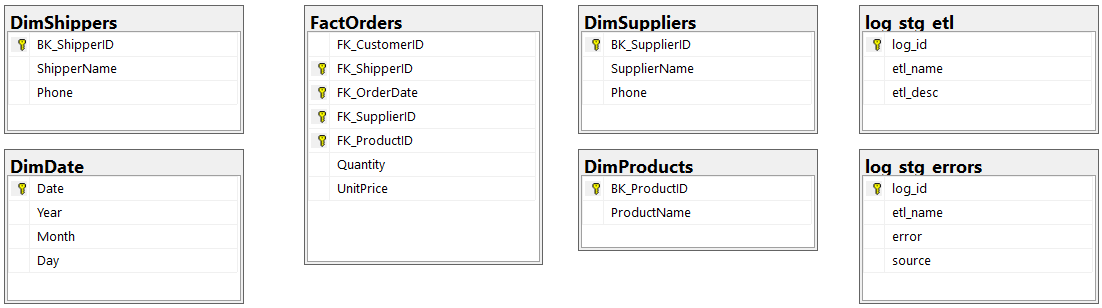

Figure 4.9- Staging Area Model

4.3.1.3. Generating the Data Warehouse

With the purpose of generating the data warehouse model for the hybrid environment, a SQL script had to be written, this time with Surrogate Keys that correspond to specific Business Keys (BKs), previously denoted as Primary Keys (PKs), that form a composite key in the fact table, defining, with the aid of the different dimensions (Product, Date, Supplier, Shipper), the different facts in the table. Additionally, the fact table also has two columns that are aggregate measures to be analyzed at the consumption layer (Quantity and UnitPrice). The final result of this script was the model in figure 4.10:

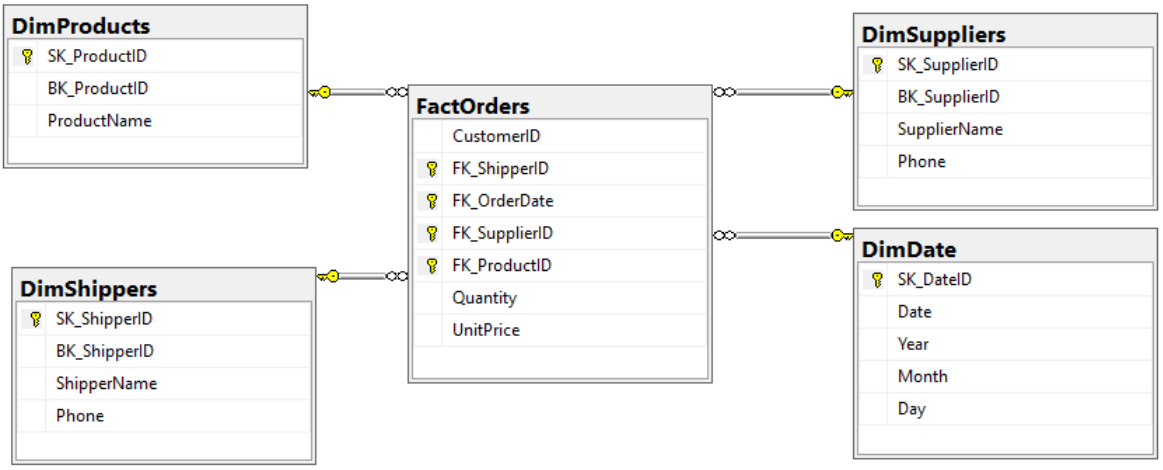

Figure 4.10- Data Warehouse Model

4.3.2. Developing the ETL Pipeline in Visual Studio

Having the database and the data warehouse correctly configured in the on-premises hardware, it is now crucial to develop the Extract-Transform-Load pipeline that will process the incoming data from the OLTP Database. For that, Visual Studio was used, alongside the Integration Services Extension.

Figure 4.11- Visual Studio Application

Launching the software, we started by creating a new integration services project. Once inside, the first step is to establish a connection between Visual Studio and SQL Server Databases. For that, using the OLE DB Driver, the three SQL components previously generated were connected.

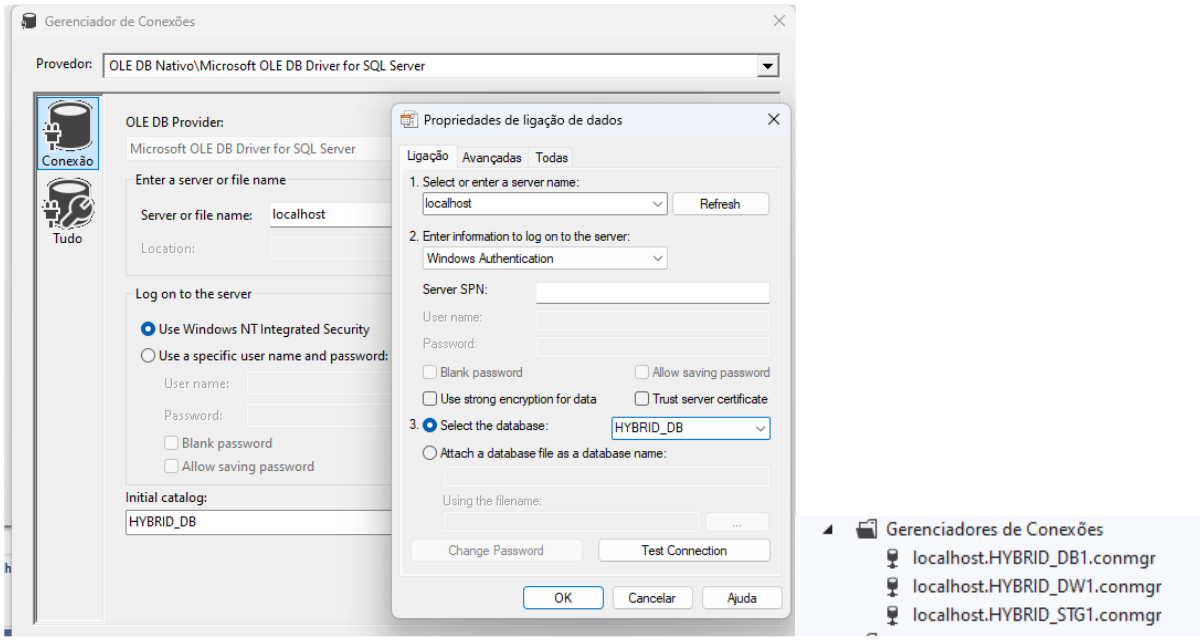

Figure 4.12- Establishing connections to SQL Server on Visual Studio

Afterwards, the SSIS packages were defined to enable the ETL process to run in parts. Firstly, the process from the source to the staging area, and then from the staging area to the data warehouse.

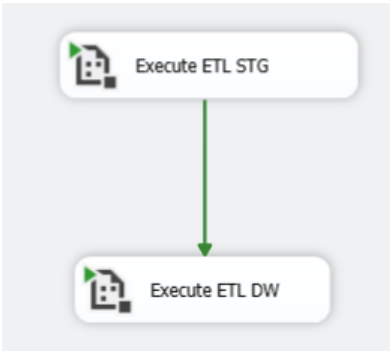

Figure 4.13- Macroview of the ETL process

Inside “Execute ETL STG”, several log tasks can be found along the process, that are used to document the several steps and help troubleshoot in case any problems are found. With the help of additional

parameters that were defined, it is possible to define if the data being imported is an “incremental load” or a “full load”, system dates for the logs, among other information. For example, the latest order date from the orders in the data warehouse is retrieved, in order to understand what records are new.

After these initial logs, the data that is in the staging area from previous ETL runs is wiped, so that afterwards the new records can be loaded into this database. In the end a count of inserted rows is registered.

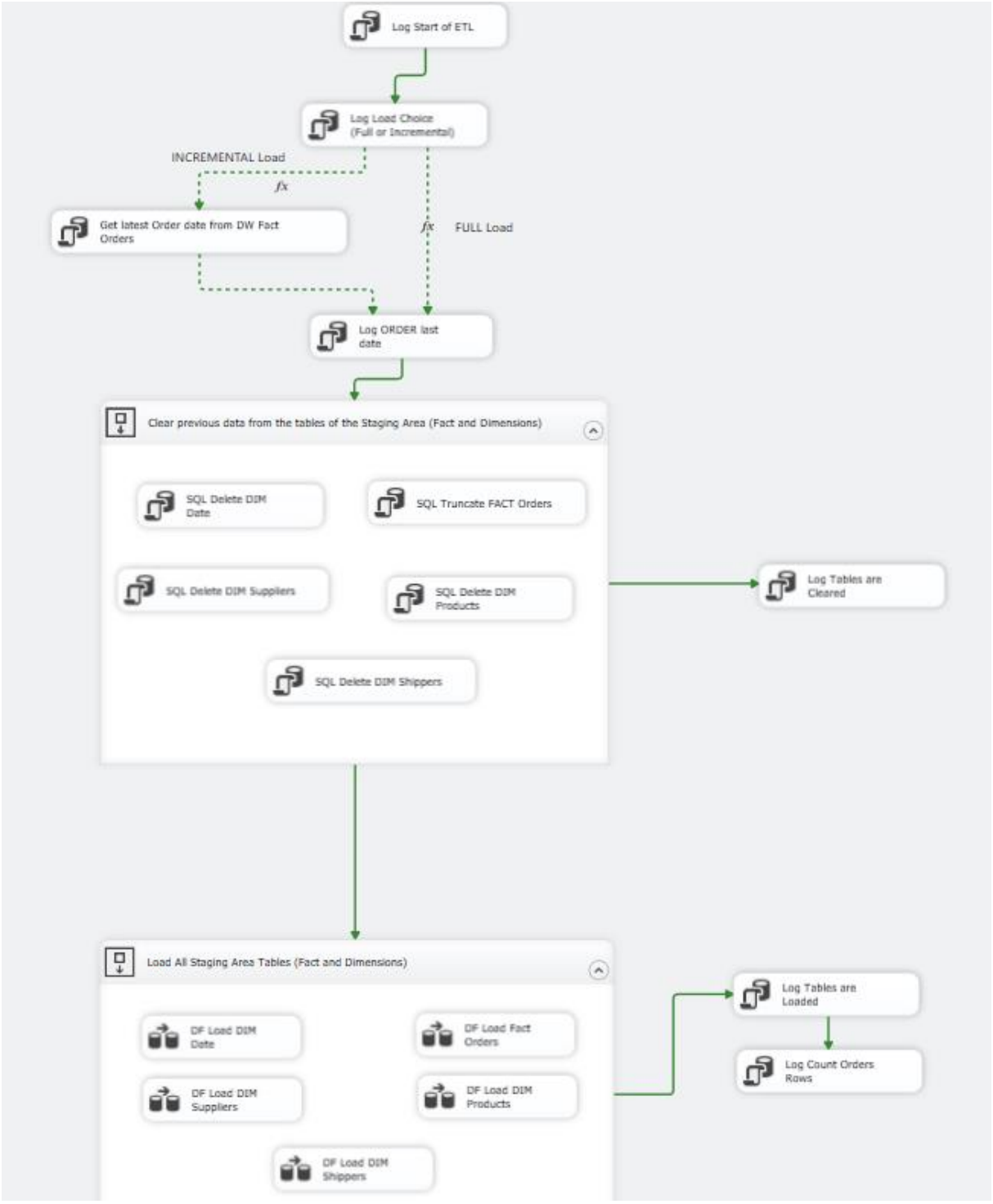

Figure 4.14- Staging Area ETL SSIS

Analyzing specifically the Fact Orders Load, and using it as example, we define OLE DB Sources based on the established connections and route the data from the database to the staging area, in this scenario, with additional steps to calculate the aggregate measures for the quantities and unit prices.

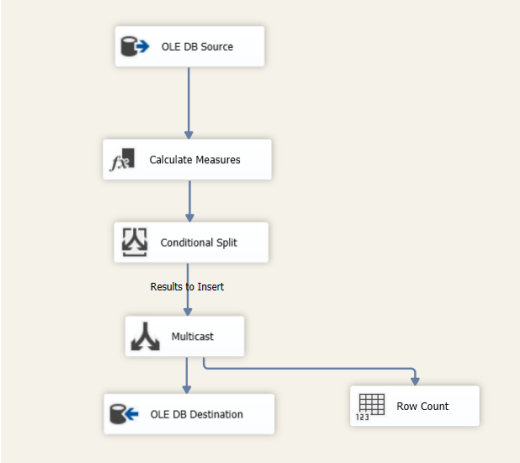

Figure 4.15- Fact Orders Staging Area Load

Moving to the “Execute ETL DW” package, there are also several log tasks scattered throughout the process that will help keep track of potential issues. Given the Foreign Key constraints that exist in the Fact table, this is the first to be erased (in case this is a full load, because otherwise the process does not even delete rows in the data warehouse), and then the remaining dimension tables are erased as well.

Once this is done, the new records that were inserted inside the staging area can now be imported into the data warehouse. The dimensions come first, and the fact table comes second, once again due to the existing constraints.

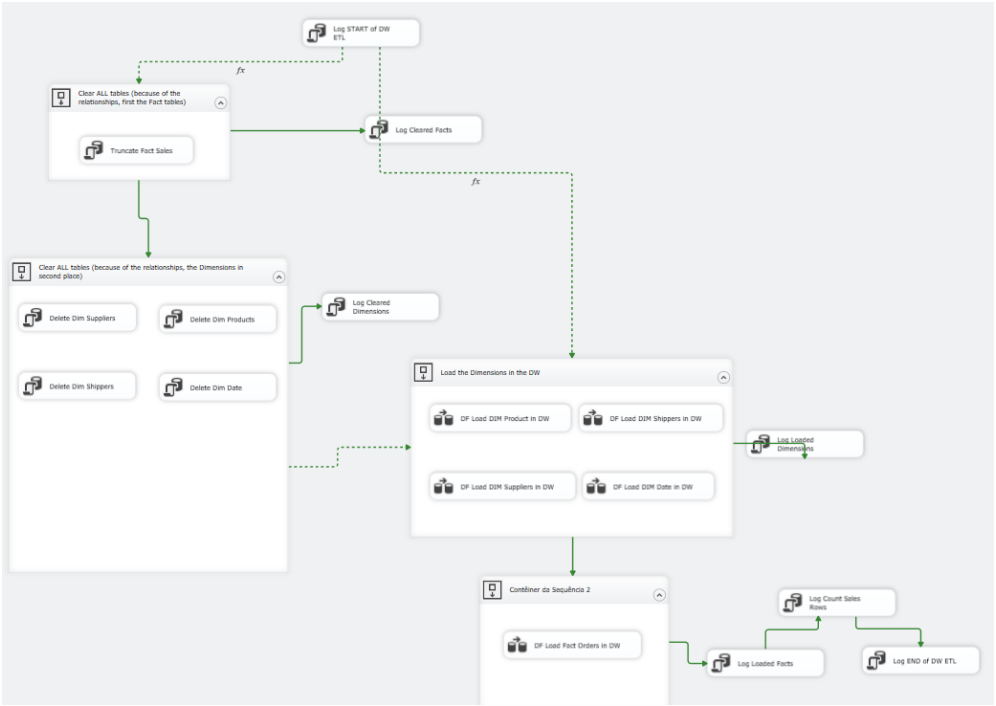

Figure 4.16- Data Warehouse ETL SSIS

Once again, looking more closely at the loading of the fact orders table, the data is routed from the staging area to the data warehouse, and since we have to adequate the Foreign Keys to be in accordance with the Surrogate Keys in the dimensions, we have to perform lookups to retrieve these SKs matching the old Foreign Keys with the Dimension's Business Keys.

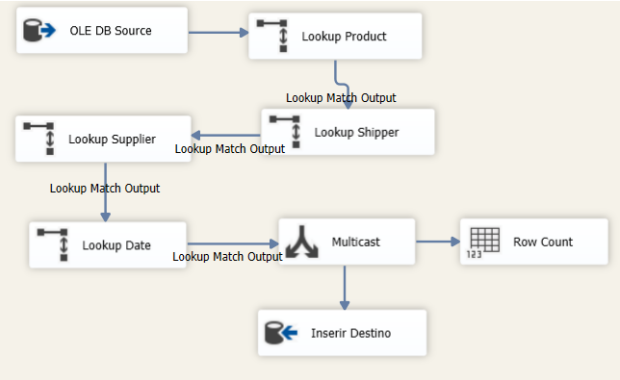

Figure 4.17- Fact Orders Data Warehouse Load

In this phase, the data is already stored and cleaned in the on-premises data warehouse, successfully, and it is possible to use this warehouse for analysis and reporting.

4.3.3. Moving Data to the Cloud

4.3.3.1. Setting up an Azure account

Since the cloud component of this project revolves around Microsoft tools, the first step on the cloud approach is to sign up for an Azure account. It is important to get a Subscription, otherwise no tools will be available for usage. Free accounts work under certain conditions and limitations.

4.3.3.2. Configuring a Resource Group and an Azure Synapse Workspace

Once inside the Azure Portal, the first step is to create a Resource Group, which includes several resources necessary for the implementation of this solution. For this we define a name for the Resource Group.

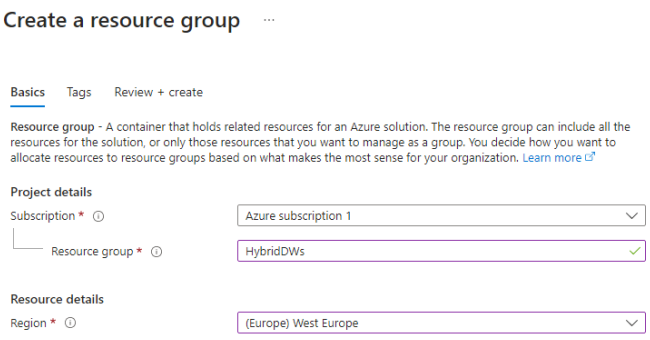

Figure 4.18- Resource group creation

Afterwards, we need to create an Azure Synapse Workspace, on which we will work on and develop our Warehouse and Pipelines. Here we need to have a resource group defined, set a name for the workspace, create a new Data Lake Storage account and a new file system name.

Figure 4.19- Azure Synapse Analytics Application

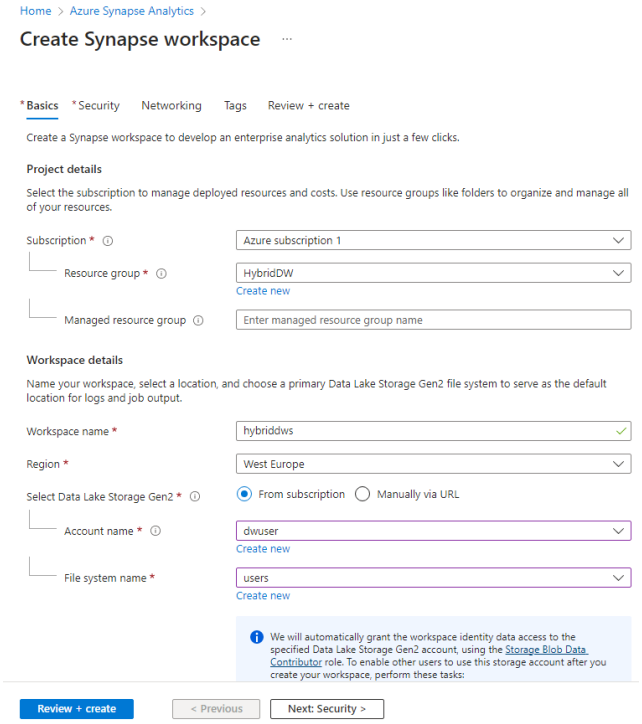

Figure 4.20- Synapse Workspace creation

4.3.3.3. Linking Services with Integration Runtime

Azure needs to communicate with the SQL Server on-premises in order to access the data warehouse that is located there. That is the main reason why Integration Runtime is needed. Opening the newly create Synapse Workspace Studio, if one goes to “Manage” and then “Integration Runtimes”, the Integration Runtime can be installed.

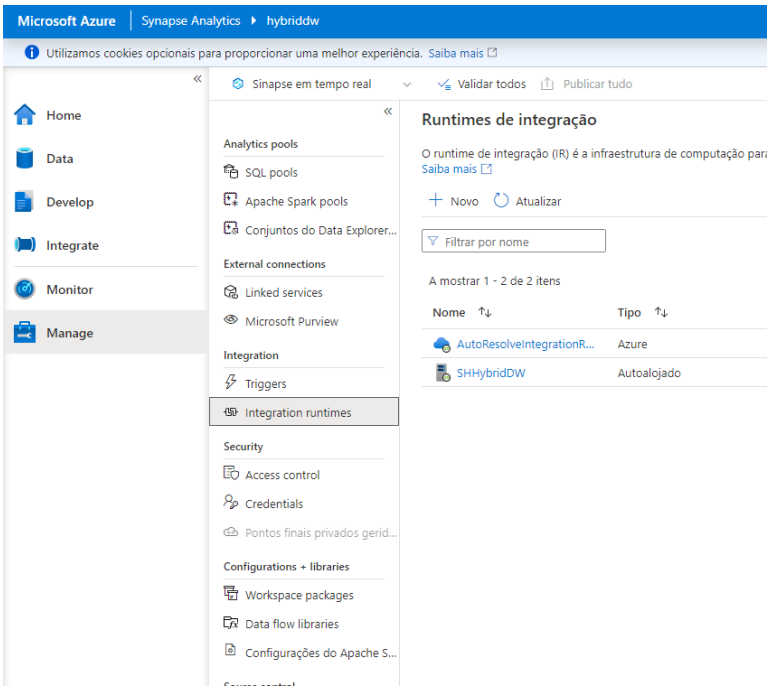

Figure 4.21- Integration runtimes section on Synapse Studio

Once on the Integration runtime screen, a new instance needs to be added, more specifically of Self-Hosted type, like seen on the image below. When the instance is created, we can quickly or manually configure it on the on-premises system. Option 1 downloads an installer that automatically sets up the authentication keys for us and establishes the connection to our local server.

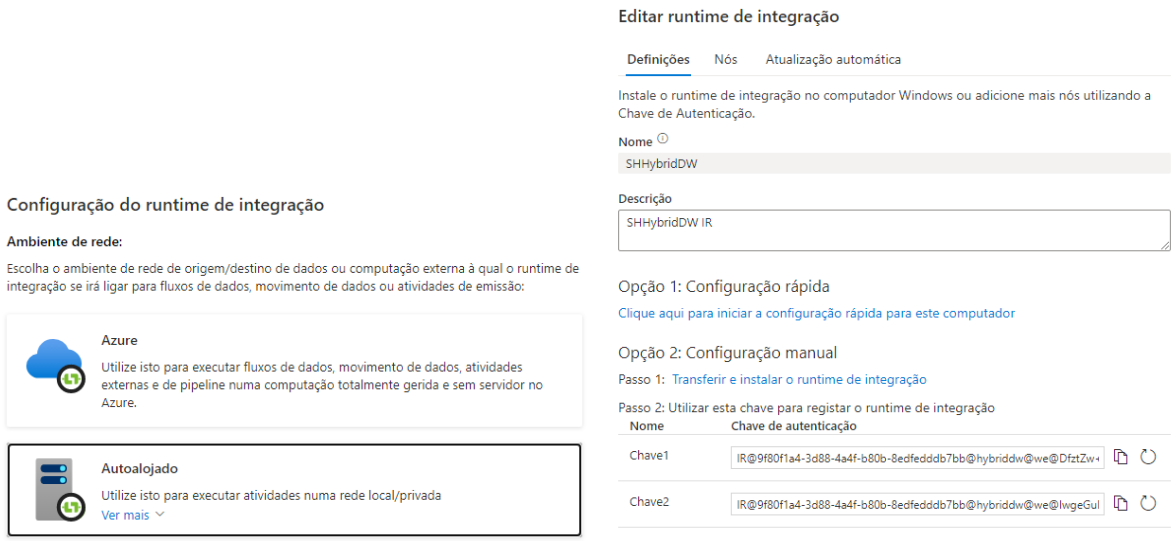

Figure 4.22- Configuring a self-hosted integration runtime

The final result of this configuration is this configuration manager, as seen below, that indicates that the local on-premises node is connected to the cloud resource, with the names and settings defined for that occasion.

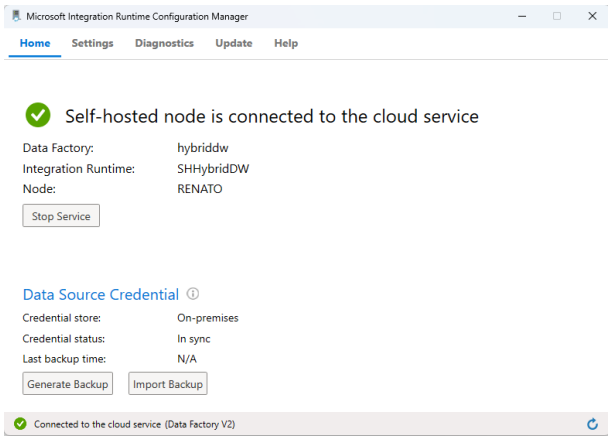

Figure 4.23- Integration Runtime running on the on-premises system

4.3.3.4. Developing a Pipeline in Azure Data Factory

With these previous connections established and the correct configuration, it is time to proceed to the development of a data flow pipeline to import and replicate the data warehouse to the cloud, similar to the one that was built in Visual Studio.

Moving to the “Integrate” section of Synapse Studio, a new pipeline needs to be instantiated on the menu. Afterwards, different “Copy Tasks” need to be defined according to the different number of tables that exist in our data warehouse. These tasks must be connected between them.

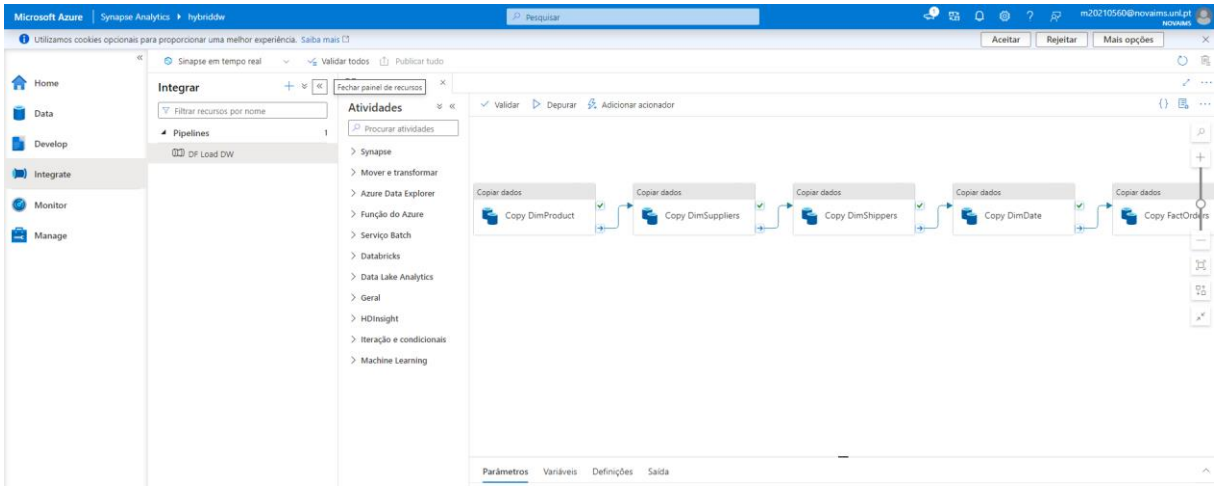

Figure 4.24- Pipeline definition on Azure Synapse

Inside each Task, several parameters must be defined, such as the Source of the data and the Destination of it. For this, we need to define a SQL Server connection to each table on the Data warehouse based on the integration runtime that was defined previously. For the destination on each task, similarly, we need to define a new table to be hosted on the SQL Pool (DW).

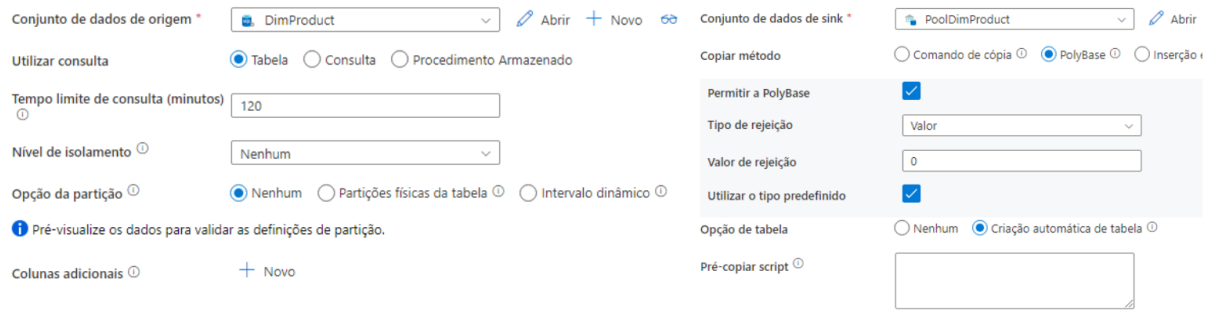

Figure 4.25- Source and Destination of Copy Task in Pipeline

After the debugging and deployment of this pipeline, the data must be successfully retrieved from the data warehouse and inserted into the cloud location and become coherent between both on-premises and cloud systems, and accessible on the client level from any of these warehouses.

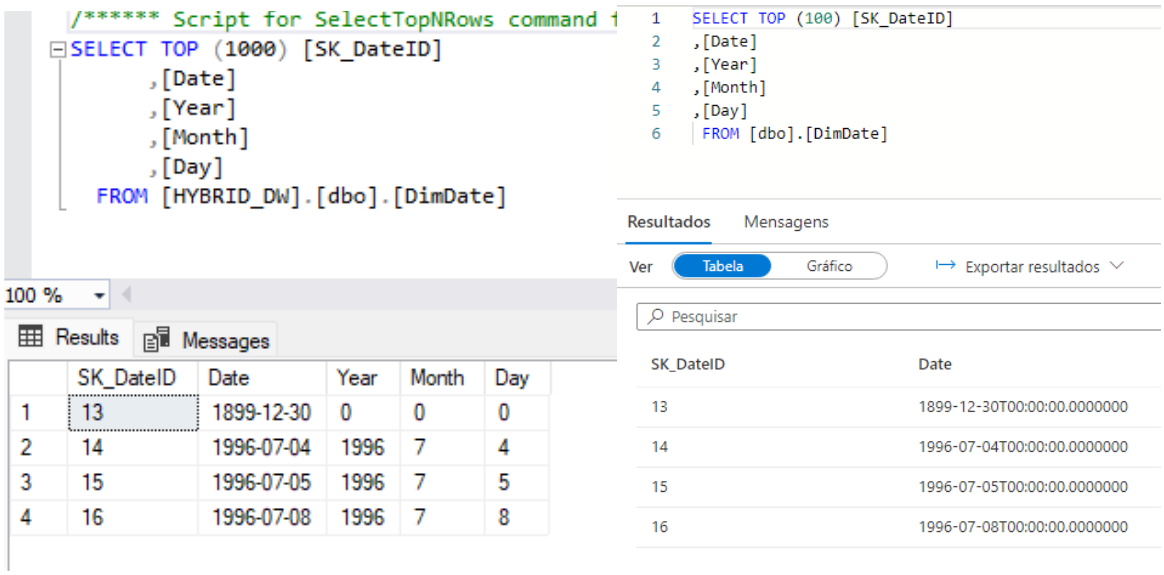

Figure 4.26- Dim Date across SQL Server and Azure SQL Pool

5. RESULTS AND DISCUSSION

After having successfully completed the implementation of the proposed architecture of a hybrid data warehouse, we can now move forward to discussing how this solution behaves in comparison to other existing solutions that are currently used for this matter. It is important to note that Azure Cloud offers several different bundles of performance for their cloud systems, and the ones used for the demonstration in this project were always the cheapest available for trial purposes.

5.1. POSITIVE POINTS OF THIS HYBRID APPROACH OVER A TRADITIONAL OR CLOUD ONLY

Effectively, adopting the Replication approach of building a hybrid data warehouse, as the one exemplified in the conceptual model of the project, can consist of several different opportunities and advantages for their users. It makes up for a strategic balance between on-premises and cloud environments, giving businesses the advantages of both while attending to particular needs in terms of cost, performance, scalability, and governance.

For starters, considering data redundancy and availability, the replication approach will ensure that data is redundant and is backed up with a copy of it in both on-premises and cloud environments. This redundancy enhances both availability and resilience. If, for instance, one of the environments experiences moments of downtime, the other one will still provide access to critical data, mitigating the risk of data loss and interruptions. It also supports robust business continuity and disaster recovery strategies, ensuring minimal disruption in operations.

Regarding performance, replication allows localized access to data, which may increase performance by decreasing latency. The users are able to query and analyze data from the location that are geographically closer to them, in the case of companies operating abroad, resulting in quicker response times. Companies with high headcounts may also benefit from decentralizing their data.

In terms of scalability and flexibility, the replication approach enables these qualities, since as data grows, additional resources can be allocated to on-premises and cloud locations independently, allowing for efficient scaling according to specific needs. The adaptability provided by this possibility is important for handling varying workloads and changes in data volume.

This approach also facilitates data governance and compliance by permitting that companies maintain control over sensitive data inside on-premises infrastructures, while using the scalability of the cloud for non-sensitive data. This enables easier addressing of regulatory requirements and residency constraints.

Optimizing costs is also possible, as companies may prefer to replicate only essential data to the cloud, instead of migrating the entire data warehouse. This enables the organizations to reduce costs by utilizing the cloud resources for certain workloads and retaining the on-premises structures for other purposes, such as sensitive data or legacy systems.

The replication approach is also viable for companies that are planning on migrating completely to the cloud in the future, as this process allows for incremental migration. Transitioning gradually to the cloud enables impact reduction on operations, reduces risks, and provides time for testing and optimization.

5.2. NEGATIVE POINTS OF THIS APPROACH

On the other hand, designing and implementing such a solution may constitute challenges and disadvantages in some respects and it is essential for organizations considering this approach to carefully weigh the negative sides against the benefits and to develop strategies to address them. Hybrid data warehouse architecture implementation and maintenance require careful planning, strong governance procedures, and continual monitoring.

Regarding the complexity of this approach, implementing and managing data replication between on-premises and cloud environments may prove to be a rather intricate process. It requires coordination, thorough planning and ongoing maintenance to ensure the consistency across the replicated data.

Data consistency is also another important factor to consider, as maintaining it between replicated data can be challenging, specifically in environment with high volume of data flowing. Delays or errors may result in inconsistencies, impacting the reliability of analytics and decision making.

In monetary terms, replicating data to both environments can lead to increased storage costs. Storing duplicate data across several locations may contribute to higher storage expenses, particularly in scenarios where the volume of data is substantial.

Regarding security, the process of replicating data introduces additional security concerns. It is crucial to ensure that the data is secured during transitioning and storing. Managing access controls, encryption and compliance with protection regulations becomes more difficult in a hybrid environment.

Considering bandwidth, the procedure of replicating data over networks, namely for large data volumes, may occur in latency constraints. Resulting possibly in delays in data syncing and impacting the performance of applications that rely on real-time data.

In terms of vendors, depending on the technologies adopted by the company and the cloud services that are chosen, there can be risks of vendor lock-in, as proprietary replication solutions and depending on specific providers could limit flexibility to make changes in the future.

While data governance may become simplified in a hybrid environment, in some cases it can become a challenge, as managing compliance in these scenarios involves complexities related to regulatory requirements and ensuring that replicated data is in accordance with data regulations and standards becomes a concern.

The initial setup costs for this environment can also be significant, as this includes investing in technology, training and infrastructures to support the process. It is important to consider, additionally, that integration with legacy systems may pose limitations or require additional modifications to allow the integration of the systems to be seamless and cohesive.

5.3. HOW DOES IT COMPARE TO EXISTING SOLUTIONS? HOW CAN IT IMPROVE IN THE FUTURE?

After having discussed the advantages and disadvantages of the Hybrid Data Warehouse approach alone, it is also important to compare it side-by-side with the traditional and cloud-only approach.

It is imperative to stress that SQL Server and Azure Cloud, two Microsoft products, were used in the implementation of the hybrid solution this project presented. The project's complexity and the requirement to utilize a cohesive ecosystem for smooth interoperability and integration guided this decision. It is important to recognize, nonetheless, that the choice of Microsoft tools does not rule out the potential of using other toolkits and combinations that could result in different price points, different performance levels, and extra capabilities. The choice to adopt a Microsoft-centric approach for this project was made under particular restrictions, and it is important to consider these factors when interpreting the results.

It is crucial to understand that different toolkits and configurations may be used in real-world circumstances, and the constraints of this project preclude a thorough evaluation of the true costs and effectiveness of the selected solution in more expansive, dynamic operational contexts. As a result, businesses thinking about implementing similar hybrid data warehouses are advised to carry out in-depth analyses based on their particular needs, taking into account a range of tools and technologies to make well-informed decisions in line with their particular objectives and limitations.

The table below offers a succinct comparison across important characteristics, making it possible to quickly assess the advantages and disadvantages of each strategy.

Table 5.1- Comparison between hybrid data warehouses and other solutions.

Dimension	Hybrid with Replication	On-Premises	Cloud-Only
Architecture and Deployment	Combines on-premises control with cloud scalability. Allows for data redundancy and localized access.	Complete control over data security and compliance.	Greater scalability and flexibility. Potential concerns about data security.
Scalability and Flexibility	Scales by leveraging cloud resources while maintaining on-premises control.	Limited scalability compared to cloud solutions.	Highly scalable to accommodate varying workloads. Scaling costs may increase with usage.
Data Consistency and Integration	Offers data redundancy for improved availability.	Centralized control over data consistency.	Optimized for cloud-based data integration. Potential challenges in integrating with on-premises data sources.
Cost Considerations	Potential cost savings through a phased transition.	Lower long-term operational costs.	Lower upfront costs with pay-as-you-go models. Long-term costs may increase with high usage.
Security and Compliance	Leverages on-premises security measures.	Complete control over on-premises security measures.	Benefits from cloud provider security measures. Concerns about data sovereignty and control.
Conclusion	Strikes a balance between on-premises control and cloud scalability. Requires careful management of replication complexities.	Offers complete control over data security but may lack the scalability and flexibility of cloud solutions.	Highly scalable but may raise concerns about data security and long-term costs.

Despite the developed approach have not been tested in the real world, it still constitutes a viable implementation in this context, especially for small businesses starting the process of moving their data from on-premises to the cloud. Because the organization's data infrastructure is gradually evolving, the design places a high priority on sustaining data in both locations. Although it might not be the perfect example of a hybrid warehouse, it is a great place to start since it makes the first steps of the transformation possible and already significantly improves data accessibility.

Therefore, future research may be necessary to validate the proposed architecture through empirical testing in real-world scenarios, allowing the purposed solution to become even better and to ensure that it adjusts to more specific use-cases. As a result, in future iterations the following aspects could be thought of:

- Performance optimizations as to discover solutions that improve the performance of the hybrid data warehouse, especially in scenarios of large volumes of data. This could be done by fine-tuning queries, improving the pipelines and algorithms used, or investigating other mechanisms to increase the efficiency.
- Conducting a detailed cost analysis to comprehend how this hybrid solution impacts organizations financially. Understanding what other alternatives exist, such as improving resource allocation, using reserved instances or other pricing plans provided by cloud providers.
- Analyzing the inclusion of modern tools and technologies to scale the hybrid data warehouse. Could include the integration of sophisticated analytics tools, machine learning algorithms, or data processing frameworks to improve the capabilities of the system.
- Improving the security measures of this hybrid solution, by analyzing modern encryption techniques, implementing access restrictions, and ensuring that the newest security practices are being employed.
- Automating data synchronization processes among on-premises and cloud environments. Investigate procedures that demand less manual intervention while ensuring near-real-time consistency across both locations.
- Examining other cloud providers to understand how the solution operates in different settings. This allows for the comparison of features, pricing and performance, translating in useful information for decision-making in enterprises.

6. CONCLUSIONS

This project has successfully addressed the primary objective of delivering a viable method for implementing a hybrid data warehouse architecture. While leveraging Microsoft tools such as SQL Server and Azure Cloud, the hybrid solution provided here is a first step for small businesses to transfer their data from on-premises to the cloud while keeping a dual presence.

The benefits of using a hybrid data warehouse include its adaptability to the changing demands of a small business, enabling progressive migration, and improving data accessibility. The comparison with similar designs provides useful insights into the strengths and potential areas for improvement, establishing the framework for informed decision-making.

The exploration of future works in this project reflects a commitment to ongoing refinement and optimization. Opportunities for performance improvements, cost analyses, real-world testing, and tool integration pave the way for a dynamic and evolving data management infrastructure. This work establishes a clear trajectory for continuous improvement by acknowledging the limitations of the current solution and positioning it as a starting point.

However, it is critical to recognize the current scope's limitations. The lack of ability to assess the solution's true costs and performance in a real-world scenario highlights the need for future empirical validation. Furthermore, the comparison with other solutions is based on theoretical considerations, whereas real-world testing is required for a more thorough evaluation.

In essence, this project not only provides a practical methodology for implementing hybrid data warehouses, but it also lays the groundwork for future research and development. This work is expected to inspire further research, optimization, and advancements in hybrid data warehouse architectures, contributing to the ever-changing landscape of data management in the digital era.

BIBLIOGRAPHICAL REFERENCES

Carstensen, A.-K., & Bernhard, J. (2019). Design science research – a powerful tool for improving methods in engineering education research. European Journal of Engineering Education, 44(1–2), 85–102. https://doi.org/10.1080/03043797.2018.1498459

Garani, G., Chernov, A., Savvas, I., & Butakova, M. (2019). A Data Warehouse Approach for Business Intelligence. 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 70–75. https://doi.org/10.1109/WETICE.2019.00022

Gardner, S. R. (1998). Building the data warehouse. Communications of the ACM, 41(9), 52–60. https://doi.org/10.1145/285070.285080

List, B., Bruckner, R. M., Machaczek, K., & Schiefer, J. (2002). A Comparison of Data Warehouse Development Methodologies Case Study of the Process Warehouse. Em A. Hameurlain, R. Cicchetti, & R. Traunmüller (Eds.), Database and Expert Systems Applications (Vol. 2453, pp. 203–215). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-46146-9_21

Kimball, R., & Ross, M. (2013). The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling. Wiley.

Ariyachandra, T., & Watson, H. J. (2008). Technical opinion Which data warehouse architecture is best? Communications of the ACM, 51(10), 146–147. https://doi.org/10.1145/1400181.1400213

Bhosale, H. S., & Gadekar, D. P. (2014). A Review Paper on Big Data and Hadoop. International Journal of Scientific and Research Publications, 4(10).

Chen, Y., Xu, C., Rao, W., Min, H., & Su, G. (2015). Octopus: Hybrid Big Data Integration Engine. 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom). https://doi.org/10.1109/cloudcom.2015.111

Fatima, N. (2022, December 12). Data Warehouse Architecture: Types, Components, & Concepts. Astera. https://www.astera.com/type/blog/data-warehouse-architecture/

Houari, M. E., Rhanoui, M., & Asri, B. E. (2017). Hybrid big data warehouse for on-demand decision needs. 2017 International Conference on Electrical and Information Technologies (ICEIT). https://doi.org/10.1109/eitech.2017.8255261

Moscoso-Zea, O., Castro, J., Paredes-Gualtor, J., & Lujan-Mora, S. (2019). A Hybrid Infrastructure of Enterprise Architecture and Business Intelligence & Analytics for Knowledge Management in Education. IEEE Access, 7, 38778–38788. https://doi.org/10.1109/access.2019.2906343

Shakhovska, N., Boyko, N., & Pukach, P. (2018). The Information Model of Cloud Data Warehouses. Advances in Intelligent Systems and Computing, 182–191. https://doi.org/10.1007/978-3-030-01069-0_13

Chaudhuri, S., & Dayal, U. (1997). An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26(1), 65–74. https://doi.org/10.1145/248603.248616

Inmon, W. H. (2005). Building The Data Warehouse, 4Th Edition (4th ed.). Wiley.

Streefkerk, R. (2023, January 3). Inductive vs. Deductive Research Approach | Steps & Examples. Scribbr. https://www.scribbr.com/methodology/inductive-deductive-reasoning/

Bhandari, P. (2023, January 30). What Is Qualitative Research? | Methods & Examples. Scribbr. https://www.scribbr.com/methodology/qualitative-research/

Reno, G. (2023, February 14). Data Warehouse Architecture: Traditional vs. Cloud. FirstEigen. https://firsteigen.com/blog/data-warehouse-architecture/

Kim, W. H. (2009). Cloud Computing: Today and Tomorrow. The Journal of Object Technology, 8(1), 65. https://doi.org/10.5381/jot.2009.8.1.c4

Fisher, C. (2018). Cloud versus On-Premise Computing. American Journal of Industrial and Business Management, 08(09), 1991–2006. https://doi.org/10.4236/ajibm.2018.89133

Bryant, A., & Charmaz, K. (2010). The SAGE Handbook of Grounded Theory: Paperback Edition. SAGE.

Maxwell, J. A. (2013). Qualitative Research Design: An Interactive Approach: An Interactive Approach. SAGE.

Inmon, W., Strauss, D., & Neushloss, G. (2010). DW 2.0: The Architecture for the Next Generation of Data Warehousing. Elsevier.

Coyne, L., Dain, J., Forestier, E., Guaitani, P., Haas, R., Maestas, C. D., Maille, A., Pearson, T., Sherman, B., Vollmar, C., & Redbooks, I. (2018). IBM Private, Public, and Hybrid Cloud Storage Solutions. IBM Redbooks.

Hurwitz, J. S., Kaufman, M., Halper, F., & Kirsch, D. (2012). Hybrid Cloud For Dummies. John Wiley & Sons.

Senna, C. R., Russi, L. G. C., & Madeira, E. R. M. (2014). An Architecture for Orchestrating Hadoop Applications in Hybrid Cloud. IEEE/ACM International Symposium Cluster, Cloud and Grid Computing. https://doi.org/10.1109/ccgrid.2014.46

Mukhopadhyay, S. (2020). Best of both worlds – The case for hybrid cloud. https://www.linkedin.com/pulse/best-both-worlds-case-hybrid-cloud-soumendu-mukhopadhyay?trk=pulse-article_more-articles_related-content-card

Gupta, A., Agarwal, D., Tan, D. S., Kulesza, J., Rahul, P., Stefani, S., & Srinivasan, V. (2015). Amazon Redshift and the Case for Simpler Data Warehouses. International Conference on Management of Data. https://doi.org/10.1145/2723372.2742795

Alhieng. (n.d.). Hybrid ETL with Azure Data Factory - Azure Architecture Center. Microsoft Learn. https://learn.microsoft.com/en-us/azure/architecture/example-scenario/data/hybrid-etl-with-adf

Chugugrace. (2023, March 3). SQL Server Integration Services - SQL Server Integration Services (SSIS). Microsoft Learn. https://learn.microsoft.com/en-us/sql/integration-services/sql-server-integration-services?view=sql-server-ver16

Erinstellato-Ms. (2023, May 24). Download SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS). Microsoft Learn. https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16

Kexugit. (2016, August 18). Migrating data to Azure SQL Data Warehouse in practice. Microsoft Learn. https://learn.microsoft.com/en-us/archive/blogs/sqlcat/migrating-data-to-azure-sql-data-warehouse-in-practice

Pimorano. (2023, February 22). Quickstart: Create and query a dedicated SQL pool (formerly SQL DW) (Azure portal) - Azure Synapse Analytics. Microsoft Learn.

Chowdhury, R., & Pal, B. (2010). Proposed hybrid data warehouse architecture based on data model. International Journal of Computer Science and Communication, 1(2), 211-213.

Nilimesh Halder, P. (2023, May 21). The Ultimate Guide to data warehousing in 2023: Concepts, techniques, and emerging trends. Medium. https://medium.com/@HalderNilimesh/the-ultimate-guide-to-data-warehousing-in-2023-concepts-techniques-and-emerging-trends-7843cb422f7

Reno, G. (2023, February 14). Data Warehouse Architecture: Traditional vs. Cloud. FirstEigen. https://firsteigen.com/blog/data-warehouse-architecture/

ANNEXES

Annex 1 – Database Script

```
CREATE DATABASE HYBRID_DB
GO

USE HYBRID_DB;
GO

-- Create or alter the Suppliers table
CREATE TABLE Suppliers (
    SupplierID INT PRIMARY KEY,
    SupplierName VARCHAR(40),
    ContactName VARCHAR(30),
    Phone VARCHAR(15)
);

-- Insert data into the Suppliers table
INSERT INTO Suppliers (SupplierID, SupplierName, ContactName, Phone)
VALUES
    (1, 'Exotic Liquids', 'Charlotte Cooper', '(171) 555-2222'),
    (2, 'New Orleans Cajun Delights', 'Shelley Burke', '(100) 555-4822'),
    (3, 'Grandma Kelly''s Homestead', 'Regina Murphy', '(313) 555-5735'),
    (4, 'Tokyo Traders', 'Yoshi Nagase', '(03) 3555-5011');

-- Create or alter the Shippers table
CREATE TABLE Shippers (
    ShipperID INT PRIMARY KEY,
    ShipperName VARCHAR(40),
    Phone VARCHAR(15)
);

-- Insert data into the Shippers table
INSERT INTO Shippers (ShipperID, ShipperName, Phone)
VALUES
    (1, 'Speedy Express', '(503) 555-9831'),
    (2, 'United Package', '(503) 555-3199'),
    (3, 'Federal Shipping', '(503) 555-9931');

-- Create or alter the Products table
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(40),
    SupplierID INT,
    UnitPrice DECIMAL(10,2),
    FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)
);

-- Insert data into the Products table
INSERT INTO Products (ProductID, ProductName, SupplierID, UnitPrice)
VALUES
    (1, 'Chai', 1, 18.00),
    (2, 'Chang', 1, 19.00),
    (3, 'Aniseed Syrup', 1, 10.00),
    (4, 'Chef Anton''s Cajun Seasoning', 2, 22.00),
    (5, 'Chef Anton''s Gumbo Mix', 2, 21.35),
    (6, 'Grandma''s Boysenberry Spread', 3, 25.00),
    (7, 'Uncle Bob''s Organic Dried Pears', 3, 30.00),
    (8, 'Mishi Kobe Niku', 4, 97.00);
```

```sql
-- Create or alter the Orders table
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID VARCHAR(5),
    OrderDate DATE,
    ShipperID INT,
    FOREIGN KEY (ShipperID) REFERENCES Shippers(ShipperID)
);

-- Insert data into the Orders table
INSERT INTO Orders (OrderID, CustomerID, OrderDate, ShipperID)
VALUES
    (10248, 'VINET', '1996-07-04', 3),
    (10249, 'TOMSP', '1996-07-05', 1),
    (10250, 'HANAR', '1996-07-08', 2),
    (10251, 'VICTE', '1996-07-08', 1);

       -- Create or alter the OrderDetails table
CREATE TABLE OrderDetails (
    OrderID INT,
    ProductID INT,
    Quantity INT,
    UnitPrice DECIMAL(10,2),
    PRIMARY KEY (OrderID, ProductID),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

-- Insert data into the OrderDetails table
INSERT INTO OrderDetails (OrderID, ProductID, Quantity, UnitPrice)
VALUES
    (10248, 1, 12, 14.00),
    (10248, 2, 10, 9.80),
    (10249, 3, 9, 18.60),
    (10249, 4, 40, 42.40),
    (10250, 5, 10, 7.70),
    (10250, 6, 35, 42.40),
    (10250, 7, 15, 16.80),
    (10251, 8, 6, 16.80),
    (10251, 1, 15, 15.60),
    (10251, 7, 20, 16.80);

GO
```

Annex 2 – Staging Area Script

```sql
CREATE DATABASE HYBRID_STG
GO

USE HYBRID_STG;
GO



CREATE TABLE DimProducts(
  BK_ProductID INT PRIMARY KEY,
  ProductName varchar(50)
);

CREATE TABLE DimDate(
  Date  DATE NOT NULL PRIMARY KEY ,
  Year  SMALLINT NOT NULL,
```

```
  Month TINYINT  NOT NULL,
  Day   TINYINT  NOT NULL
);

CREATE TABLE DimSuppliers(
  BK_SupplierID INT PRIMARY KEY,
  SupplierName varchar(40),
  Phone VARCHAR(15)
);

CREATE TABLE DimShippers(
  BK_ShipperID INT PRIMARY KEY,
  ShipperName varchar(40),
  Phone VARCHAR(15)
);

DROP TABLE FactOrders

CREATE TABLE FactOrders(
  FK_CustomerID VARCHAR(5),
  FK_ShipperID INT,
  FK_OrderDate DATE,
  FK_SupplierID INT,
  FK_ProductID INT,
  Quantity INT,
  UnitPrice decimal(9,2)
 CONSTRAINT [PK_stg_FACT_Orders] PRIMARY KEY CLUSTERED
(
       FK_ShipperID ASC,
       FK_OrderDate ASC,
       FK_SupplierID ASC,
       FK_ProductID ASC
));


/* ***********************************************************************
 * The Log table for ETL errors.
 *
 * We create a table in the SA that will enable us
 * to save errors from each individual ETL run into
 * Staging Area, so we can investigate at later time.
 *
 * ***********************************************************************/
CREATE TABLE log_stg_errors (
    log_id INT identity(1, 1) PRIMARY KEY,
       etl_name NVARCHAR(50) NULL,
       error NVARCHAR(max) NULL,
       source NVARCHAR(100) NULL
       );


/* ***********************************************************************
 * The Log table for each ETL action.
 *
 * We create a table in the SA that will enable us
 * to save information from each individual ETL
 * run into Staging Area, for historical records.
 *
 * ***********************************************************************/
CREATE TABLE log_stg_etl (
       log_id INT identity(1, 1) PRIMARY KEY,
       etl_name NVARCHAR(50) NULL,
```

```
	etl_desc NVARCHAR(50) NULL
	);
```

Annex 3 – Data Warehouse Script

```
CREATE DATABASE HYBRID_DW
GO

USE HYBRID_DW;
GO



CREATE TABLE DimProducts(
  SK_ProductID INT PRIMARY KEY identity,
  BK_ProductID INT,
  ProductName varchar(50)
);

CREATE TABLE DimDate(
  SK_DateID INT PRIMARY KEY identity,
  Date  DATE NOT NULL  ,
  Year  SMALLINT NOT NULL,
  Month TINYINT  NOT NULL,
  Day   TINYINT  NOT NULL
);

CREATE TABLE DimSuppliers(
  SK_SupplierID INT PRIMARY KEY identity,
  BK_SupplierID INT,
  SupplierName varchar(40),
  Phone VARCHAR(15)
);

CREATE TABLE DimShippers(
  SK_ShipperID INT PRIMARY KEY identity,
  BK_ShipperID INT,
  ShipperName varchar(40),
  Phone VARCHAR(15)
)
DROP TABLE FactOrders
CREATE TABLE FactOrders(
  CustomerID varchar(5),
  FK_ShipperID INT FOREIGN KEY REFERENCES DimShippers(SK_ShipperID),
  FK_OrderDate INT FOREIGN KEY REFERENCES DimDate(SK_DateID),
  FK_SupplierID INT FOREIGN KEY REFERENCES DimSuppliers(SK_SupplierID),
  FK_ProductID INT FOREIGN KEY REFERENCES DimProducts(SK_ProductID),
  Quantity INT,
  UnitPrice decimal(9,2)
  CONSTRAINT [PK_FACT_Orders] PRIMARY KEY CLUSTERED
(
	FK_ShipperID ASC,
	FK_OrderDate ASC,
	FK_SupplierID ASC,
	FK_ProductID ASC
));
```