Journal of Artificial Intelligence and Big Data, 2021, 1, 1350
www.scipublications.com/journal/index.php/jaibd
DOI: 10.31586/jaibd.2021.1350

Article

Optimizing Data Warehousing for Large Scale Policy Management Using Advanced ETL Frameworks

Avinash Reddy Aitha [1,*]

[1] Lead SDET, USA

*Correspondence: Avinash Reddy Aitha (avinash.reddy.aitha.research@gmail.com)

Abstract: Data warehousing is a technique for collecting, managing, and presenting data to help people analyze and use that data effectively. It involves a large database designed to support management-level staff by providing all the relevant historical data for analysis. This chapter begins with a definition of data warehousing, followed by an overview of large-scale policy management to highlight the need for data warehousing. Next, an overview of an ETL framework is presented, along with a discussion of advanced ETL techniques. The chapter concludes with an outline of performance optimization techniques for data warehousing. Data warehousing is considered a key enabler for efficient reporting and analysis, with implementation choices ranging from cost-effective desktop systems to large-scale, mission-critical data marts and warehouses containing petabytes of data. Extract, transform, and load (ETL) systems remain one of the largest cost and effort areas within data warehouse development projects, requiring significant planning and resources to build, manage, and monitor the flow of data from source systems into the data warehouse. The technology and techniques used for ETL can greatly influence the success or failure of a data warehouse. Complex business requirements for data cleansing, loading, transformation, and integration have intensified, while operational plans for real-time and near-real-time reporting add additional challenges. Parallel loading mechanisms, incremental data loading, and runtime update and insert strategies not only improve ETL performance but also optimize data warehousing performance, particularly for large-scale policy management.

Keywords: Data Warehousing, Data Collection, Data Management, Data Presentation, Historical Data, Large-Scale Policy Management, ETL Framework, Advanced ETL Techniques, Performance Optimization, Reporting, Analysis, Data Marts, Mission-Critical Systems, Petabytes, Data Cleansing, Data Transformation, Data Integration, Real-Time Reporting, Incremental Loading, Parallel Mechanisms

How to cite this paper:
Aitha, A. R. (2021). Optimizing Data Warehousing for Large Scale Policy Management Using Advanced ETL Frameworks. *Journal of Artificial Intelligence and Big Data, 1*(1), 1–19.
DOI: 10.31586/jaibd.2021.1350

Received: August 26, 2021
Revised: November 29, 2021
Accepted: December 19, 2021
Published: December 21, 2021

Copyright: © 2021 by the author. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

The objective of a data warehouse is to reliably support information tenants, often at very large scale. As an example, government Ministries of Health require support for claims processing policies that access information about tens or hundreds of millions of insured persons, with policies that encompass hundreds of millions of procedures and payments. Operating rare extreme cases on a daily basis is a challenge. New settlements request more complexity, but performance cannot be degraded in comparison to previous implementations. A large-scale system employed by a network operator processes information related to customers and the resources used for the provided services. Each day, all customers must be billed for the actual usage in accordance with the billing policy. Customers can be billed differently depending on a variety of conditions such as the type of service and used resource, the geographical location where the resource was used, the date and time of service usage, the customer category, and so forth. All data in the billing system must be kept historically; thus, customers might be billed differently depending on the time band they fell into during their service usage. This problem is complex due to the massive amounts of information being reviewed in order to compute the billing of current customers and to perform billing reprocessing of the past to update historical billing calculations for already billed customers. Policy changes, billing errors, and claims of customers or authorities are some of the reasons why previously billed customers need to receive an updated bill.

1.1. Overview of Data Warehousing Concepts

Data warehousing is a technology that collects, integrates, and manages large volumes of data from multiple data sources throughout an organization. The resulting data stores provide the information required for promoting management decision-making and control, for understanding and predicting market and environmental shifts, and for implementing all facets of organizational planning. The functions of a data warehouse characteristic of most logical implementations of the data warehousing and business intelligence problem include the following:

- Obtaining data from operational and external sources.
- Performing data transformation and cleansing operations.
- Managing the data warehouse and associated metadata.
- Providing tools for metadata and warehouse administration.
- Supporting the provisioning of historical, current, and/or projected information that is accessible to the data warehouse user environment for all facets of the organizational process. Data warehouses and data marts have evolved as powerful tools for harnessing data to optimize strategic business decisions. These repositories of information, often called key information stores, must be properly designed and populated with accurate, clean, transformed, consistent, and timely information in an appropriate format.

2. Understanding Data Warehousing

A data warehouse is a specially designed database created to support business decision processes and make the answer to organizational business history questions much easier and faster. Data warehousing provides support to a wide range of business users and enables the achievement of strategic objectives. It is essentially a way of collecting data from across the organization into one database optimized to provide answers to almost any type of question. The term data ware-house was initially introduced in the seminal articles ”The data warehouse” and The Case for a Corporate Data Warehouse. Here Andrews gives a definition of a data warehouse: A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management’s decision-making process. An efficient data warehouse architecture includes the following components: data sources, data extraction/transformation/loading/integration mechanisms, meta-data, management and control tool and access tools. In the context of policy management, especially when the volume of data is large, it is important to have a data warehouse optimized for large-scale policy management. Large-scale policy data warehousing is a contemporary policy paradigm that permits the excavation of implicit policy records contained in complex policy structures. A functional data warehouse can be built using off-the-shelf products. Data warehousing architectures can be broadly classified as centralized or decentralized, depending upon the physical location of the various components. Broadly speaking, the architecture of a centralized data warehouse can be considered to be based on two paradigms: Bottom-up and Top-down. A thought based on the two paradigms can be considered, in terms of the integration of several departmental data marts into one recognized corporate data warehouse.

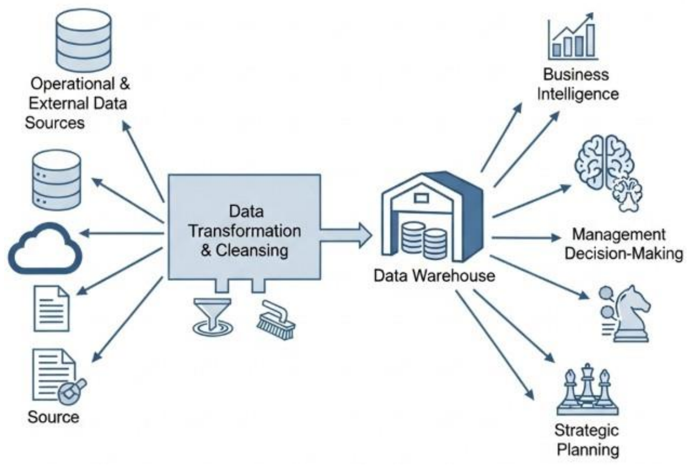
Operational & External Data Sources
Data Transformation & Cleansing
Data Warehouse
Business Intelligence
Management Decision-Making
Strategic Planning
Source

Figure 1. Data Warehousing Architecture

2.1. Definition and Purpose

Data warehousing represents a foundational element in the contemporary systems landscape. The following description sets out the basic building blocks of modern data warehousing systems for managing large-scale policies. Databases provide the principal storage for an organization's current data. A data warehouse collects the responses of distributed applications and places that information in one location. The structure is then managed to allow for creation of aggregate tables and summarized reports. Policies are expressed and managed in a simplified way; these rules help the users decide how to manage customers and accounts. The primary advantages of such systems include exposure of a large number of users to the storage and transformation of data indirectly related to their functional area. Policy management in large-scale databases, their definition, control, and progress, places high demands on data access and data analysis. Those objectives can be accomplished more efficiently when building an "ETL layer." The Extraction, Transformation, Loading (ETL) process is indeed the foundation upon which any data warehouse is built. Advanced or complex development methods are used in the design and development of the ETL framework, so that stored large-scale policy data in the data warehouse can be handled and processed smoothly and quickly. Optimal use of the ETL infrastructure results in a fulfilling demand before or during a business process.

Equation 1: ETL Processing Time

Goal. Compute total wall-clock time to process N records through a 3-stage ETL pipeline (Extract–Transform–Load) with parallel workers per stage.

Definitions (per record): Extract service time $\tau_E s/rec$ with T_E workers → stage throughput $R_E = \tau_E p_E rec/s$.

Transform time τ_T, workers $p_T \rightarrow R_T = \tau_T p_T$

Load time τ_L, workers $p_L \rightarrow R_L = \tau_L p_L$

$$T \approx k \cdot max(p_E\tau_E, p_T\tau_T, p_L\tau_L) + min(R_E, R_T, R_L)N \tag{1}$$

2.2. Key Components

Data Warehousing helps in managing the large data which is generated by a system over a period of time for auditing purposes. Organizations need to manage thousands of systems or data sources for each of their clients. These systems can range from sending

SMSs or Emails to managing authorization rules and provisioning subscriptions. These rules and requests are created/stored in a Policy Database System which acts as a single point of entry for all the services for different operations. The generated logs provide information about the different requests executed on the PMS system. Also, these generated logs help the controllers in managing and clearing fault data for the clients. Within any organization, managing the fault data is responsible to the policy management team. As the data generated by different systems doesn't follow a centralized process of generating and storing, handling of fault data is difficult. Policy management just involves the clearing of fault data, and hence it isn't classified as skillful or complex. Yet, the data volume is increasing exponentially, and team members require mechanisms to easily locate the data in the system to handle the faults generated. As a solution, the logs generated for each of the requests executed on the PMS database can be processed, conveyed, and organized into an optimized database system for easier tracking.

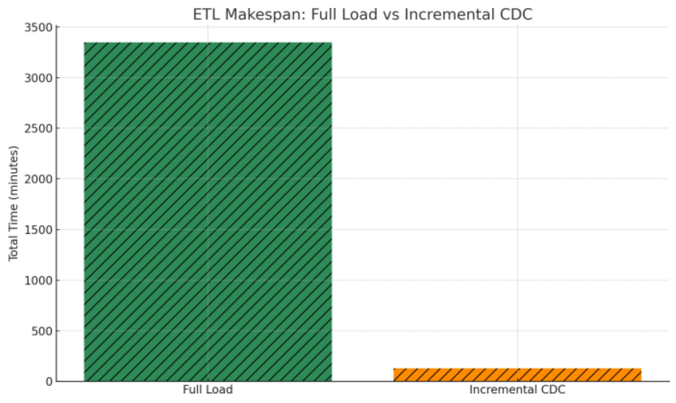

Figure 2. ETL Makespan: Full Load Vs Incremental CDC

2.3. Data Warehousing Architectures

Data warehousing is defined as a process for analyzing and managing data collected from business operations or activities. Data warehouses are typically the central point of an enterprise's decision support system, capable of managing massive amounts of service data in a detailed, highly redundant format. The main components of data warehousing encompass sources, tools for data extraction, transformation and loading (ETL), metadata, warehouse data, and query tools. The structured data flow process begins when new or altered data is detected in the operational source(s). This data is then extracted and loaded into a staging area, where any referential integrity conflicts are resolved. Prior to loading into the target data warehouse(s), the data is cleansed and consolidated. The final step involves refreshing the target data warehouse(s) with the new data. Finally, users issue queries through front-end query and analysis tools. These analytical processes cover classification, prediction, and time-series analysis. Whenever the database management system releases new logic, DML statements are generated and automatically applied to the data elements in the specified warehouse table. Frequent processing of this nature can lead to data redundancy and increased processing times in an organization. Therefore, large-scale policy management relies on an ETL framework in data warehousing for the analysis of schemes and policies.

3. Policy Management in Large-Scale Systems

Managing policies in large-scale environments is intricate, requiring interaction with diverse systems, applications, delivery platforms, and administration portals, each boasting millions of policy objects. Data warehouses (DWs) play a crucial role here, providing comprehensive analysis, decision-making, and forecasting capabilities across widespread business arenas. However, their complex functionality leads to latency issues during high-load analyses. Select indexes reduce data retrieval time, but updating them poses challenges—especially during bulk insert, update, or delete operations. A strategic approach involves dropping most indexes before loading data and rebuilding them afterward, parallelizing index builds based on system resources. Beyond indexing, successful data loading hinges on robust Extract-Transform-Load (ETL) foundations. Parallelized, incremental data loading, combined with effective data cleansing, transformation, and loading techniques, enables real-time analysis and significantly enhances DWH performance.

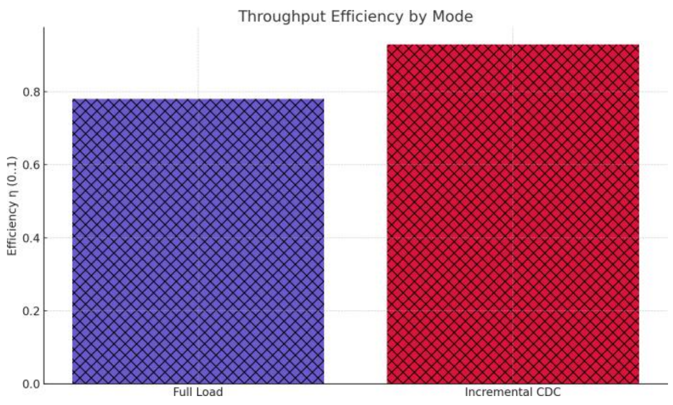

Figure 3. Throughput Efficiency by Mode

Equation 2: Data Throughput Efficiency

Goal. Compare actual achieved throughput to the theoretical ceiling.
Actual=TN, Theoretical Max=max(RE,RT,RL)

$$h = Theoretical\ Max\ Actual \in (0,1] \qquad (2)$$

3.1. Importance of Policy Management

Policy management maintains consistent application of laws and policies across organizations, public or private, to ensure proper functioning. When involved in making diverse decisions for many employees, a policy manager—be it an Insurance Company or Bank—needs a data warehouse for storing and managing large data sets. Efficient data ware-house architecture is crucial for optimized performance. Policy management might also pertain to managing an organization's internal policies on employees, such as allowances, leaves, and bonuses. Depending on the frequency of policy changes, the underlying data warehouse can become extremely large, thereby requiring optimization for performance enhancement. The 'Optimization Dataware Housing for Large Scale Policy Management Using ETL Framework' minimizes the time consumed during data processing and addresses all other performance-related issues in the data warehouse. Optimization techniques include indexing large-scale data using skyline probability and OLAP technologies at the storage stage; partitioning large data sets by valid time, maturity time, and entered date to improve query response times; optimizing join queries through star-join optimization; analyzing costs with cost-based hybrid caching and multi-query caching for grouping policy data; and addressing resource-relating issues by optimal resource allocation. All these aspects contribute to creating a fully optimized data warehouse for managing large-scale policy data.

3.2. Challenges in Large-Scale Environments

The demand for effective policy management systems continues to grow, driven by enterprises' operational intricacies and their need for information. A policy essentially a business rule enforced by an organization can range from corporate governance controls to property or insurance agreements. Policies typically span multiple categories and demonstrate variability across products, classes, and geographies. A vibrant financial services business is therefore characterized by a diverse, complex, and large policy portfolio. Managing such a portfolio presents unique challenges. Central to policy management are data analysis and the storage of voluminous records with high data quality and throughput. As enterprises expand into different countries and regions, accommodating multilingual and multicurrency support becomes imperative. Large volumes of financial data complicate precisely these requirements. At the core of these issues lie the management of the policy portfolio with its numerous attributes and the fulfillment of the risk management, accounting, and budgeting functions. Efficient reporting and forecasting for the portfolio define the performance of the portfolio management functional area.

4. ETL Frameworks Overview

Extraction-Transformation-Loading (ETL) defines a class of services responsible for the periodic movement of data between High and Low. It acts on data extracted from sources, such as Operational or File Interface, for delivery into a target location, normally a Data Mart. These tasks include copying, cleansing, transformation and summarisation. ETL must also work in conjunction with Load Services, typically implemented using SQL commands, which manage the loading of data into Data Warehouse databases. Load Services ensure that data is properly indexed and partitioned, and periodically reorganises and compresses Old data to optimise performance. Scroll down to Main Body. The terms Extraction-Transformation-Loading (ETL) and Extraction-Loading-Transformation (ELT) represent the same concept. Although the traditional form, ETL, implies that Extraction and Transformation occur prior to Loading, the processes are often carried out together within a single ETL job. The choice between ETL and ELT depends on the performance characteristics of the Data Warehouse and the distributed resources available.

4.1. Definition of ETL

ETL stands for Extract, Transfer, and Load. When mission-critical data is located in more than one source system, ETL is the process of extracting data from multiple source systems and transferring it into a data staging area. At this stage, the data can be cleansed, scrubbed, and transformed. Finally, the transformed data is loaded into the target database either as an append or as a refresh of existing data, in the case of a full load. Depending on the system design and the tools, the loading happens much later than the extraction and transfer, such as during the night. ETL supports incremental data loading [1]. Another method of off-loading the data from the source system is ELT, which stands for Extract, Load, and Transform. The extract and load functionalities are the same as the ETL method. The main difference lies in the way the data is transformed after being extracted and loaded into the target system. Very often, the transformation is done inside the Database Management System, often through a series of procedures, functions, transformations, or built-in commands. The advantage lies in the fact that data is moved as quickly as possible to the target environment: once it is in the target environment, it can be integrated with other data, for example, in a data warehouse or data mart.

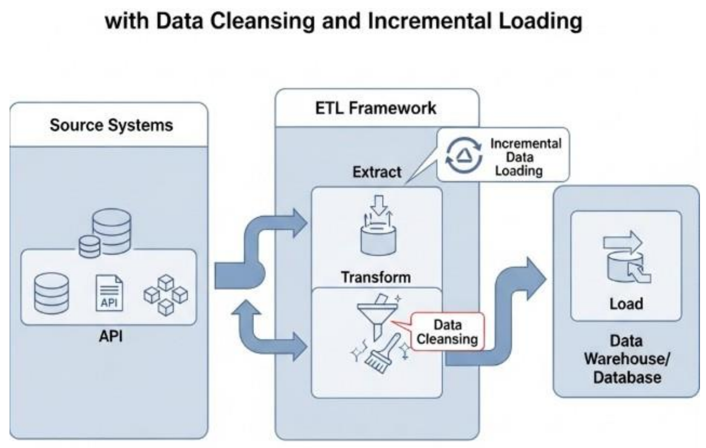

Figure 4. ETL Framework with Data Cleansing and Incremental Loading

4.2. ETL vs. ELT

Extract, Transform, and Load (ETL) is the process of extracting data, transforming it into a defined structure, and loading it into a database or warehouse. Extract, Load, and Transform (ELT) is the process of extracting data, loading it into a target system, and then performing transformation as a subsequent step. Today, most organizations handle large amounts of data. An ETL Framework is used to load this large data efficiently, as with supply chain data, product data, or large policy data. .NET applications can be used to develop such ETL Frameworks, which require planning, designing, and creating of the Application. During designing, techniques such as Incremental Data Loading are introduced. As the data volume grows, a Data Warehouse filled with it is less efficient. Introducing Data Cleansing in the ETL Framework helps maintain efficient loading and generates reports of the deleted or corrected data. Data Cleansing is a process of detecting and correcting (or removing) corrupt or inaccurate records from a database [2].

4.3. Common ETL Tools

Numerous ETL tools are available on the market, including those offered by major cloud service providers. Amazon Web Services, for example, offers Amazon Glue—an ETL tool developed on Apache Spark. Microsoft offers SQL Server Integration Services (SSIS). Google Cloud Platform offers Cloud Dataflow. Apache Nifi and Talend also provide products in this space. Each of these tools supports the bulk of standard functionality. However, services such as Amazon Glue, Microsoft SSIS, and Google Cloud Dataflow often struggle to meet performance requirements for extremely high volume data loads with tight window constraints. Continuous data loads require additional features, including real-time data streaming, change data capture, data validation, cleansing, and transformation. Data validation functions examine the incoming raw data to identify and manage duplicate or incomplete records—classified as bad data—preventing their loading into the data warehouse. Transformation tasks create new data fields derived from existing data and are typically handled by the company's business intelligence team. Key optimization aspects encompass incremental data load, data cleansing, data transformation, query optimization, and resource management [3].

5. Advanced ETL Techniques

ETL, an acronym for extract, transform, load, refers to the procedures employed to cater to the requirements of Large-Scale Data Warehousing for Policy Management. The essence of data warehousing and the intricacies of large-scale policy management have already been discussed. The discussion now extends to the practical implementation of those requirements—namely, generating a large-scale policy management data warehouse via the application of ETL Frameworks and optimizing the resultant data warehouse. Optimization in this context encompasses the application of several optimization techniques, with particular emphasis on incremental data loading. It is widely recognized that the data warehousing system is a very suitable customer for an ETL tool, which integrates data from different sources and formats, and loads it into a consistent form in the data warehouse. Incremental loading, also known as delta loading, is a useful technique in data warehousing, particularly for applications that source data from operational systems. These applications typically house enormous volumes of data. To prevent the loading process from becoming intractable, the ETL process needs to be optimized so that it loads only the newly inserted or updated records—i.e., the changes since the last load. Optimization techniques for large-scale data warehousing also extend to data cleansing, data transformation, real-time processing, and data warehouse structures [4].

5.1. Incremental Load Strategies

There are various techniques that are useful in large policy management implementations which place a heavy demand on the underlying data warehouse. Two prominent techniques are incremental data loading and the use of an advanced ETL framework. Extensive volumes of data in the data warehouse can be managed and kept up to date efficiently by adopting of incremental update mechanisms.

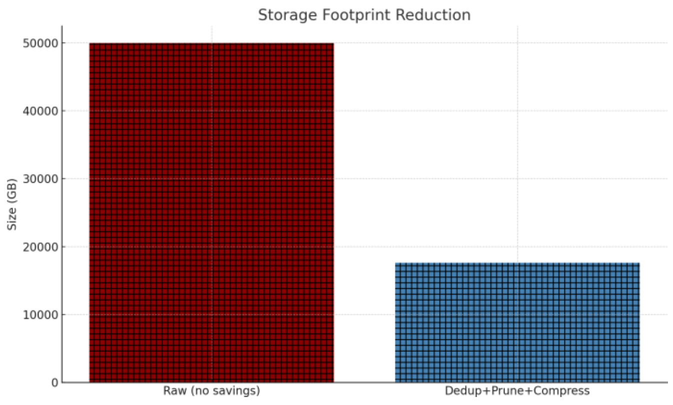

Figure 5. Storage Footprint Reduction

The objective is to only extract and load new or updated data into the warehouse. The incremental data load mechanism, also referred to as the delta load process, is a common pattern and can be readily supported by the vast majority of current ETL tools. Nevertheless, the selection of an appropriate approach must consider the specifics of the use case. Typically, incremental loading is applied to populate staging tables that feed the fact tables of a data warehouse, although the correct implementation strategy is contingent on the particular application. ”Data warehouse incremental loading and the pattern of change tracking” provides detailed insight into the design considerations and techniques associated with incremental loads. If the data warehouse schema does not include any fact tables, as is the case with policy management, an alternative approach is required. The selection of a new load strategy should be carefully weighed against the implications for metadata management. Implementation decisions also need to be informed by the specific area of policy management. ”Oracle data warehouse best practices: jointly optimising data load and query performance” explores alternative techniques for incremental loading in data warehouses which are not structured around facts and dimensions. Finally, the requirement to optimise the entire loading process, including incremental loading, may necessitate the adoption of an advanced open-source

ETL framework such as "Easy Data Transform", "Apache Airflow", "Apache NiFi", "Luigi", "Mara", "Meltano", "Metaflow", "NiPype" or "OpenLineage" [5].

Equation 3: Storage Utilization Ratio (SUR)

Goal. Model foot print after cleansing/deduplication/pruning/compression.

Let the raw size be Sraw (GB) and the fractional reductions be α_{dedup}, α_{prune}, α_{comp} (each in [0,1)).

Spost = Sraw (1 - rdedup)(1 - rprune)(1 - rcomp) If Salloc is provisioned capacity,

$$SUR = Salloc\ Spost \tag{3}$$

Table 1. Storage Utilization

Technique	Post-ETL Size (GB)	SUR (Used/Allocated)
Raw (no savings)	50000.0	0.8333333333333334
Dedup+Prune+Compress	17531.25	0.2921875

5.2. Data Cleansing and Transformation

Data cleansing and transformation represent vital stages of the ETL process, aiming to enhance data quality and semantics before loading it into the data warehouse for efficient querying and analysis. Cleansing procedures identify and resolve nulls, duplicates, inconsistencies, and incompatibilities within the data. Subsequent transformations—such as summarization, derivation, key fitting, and data splitting—adapt the cleansed data to meet schema requirements and enable fast access. Data cleansing frequently entails string manipulation, pattern matching, and, in certain scenarios, machine learning methodologies. Addressing missing values through imputation, removal, or substitution, for example, safeguards data consistency. For historical datasets generated over extended periods that can impose severe server workload, incompletely populated tables adversely affect performance. To maintain consistent current-state views, data must remain available throughout these transformations. To optimize efficiency, high volume data areas like billing, usage, policy, and geospatial sets are updated incrementally, with records changed since the last update included in the present append. Transformation functions also convert incremental changes into a structure suitable for the target [6].

5.3. Real-time Data Processing

Real time is a system of data processing, in which very small sets of data are loaded to the target area parallel with the operation of the source. It is used especially in cases where data must be detailed. It is especially used to share an account opening or closing for the bank. In the bank, an account can be opened or closed at headquarters, or branches, or call centers, or ATMs. In order to keep the opened or closed account information consistent at these locations, real-time data processing is necessary. Since data sets in real-time are very small, the requests from the source will never be reminded. The validation of the data coming from the source should be checked as carefully as possible. If the data is not validated, the data within the systems will be inconsistent. For example, a paying transaction can be done on a closed deposit account, or none transaction can be done on an opened credit account. Real-time processing means to take the requests from the source for the data update one by one and apply it to databases. A rule-based definition of data and process types should be established, since the request from the source may be "insert" or "delete" or "update." When real-time operation is executed for each request, the data of the source and the target are consistent. For example, if a deposit account information is last updated at the ATM, all of the ATMs of the bank can be used for deposit or payment transactions. But some of the ATMs cannot perform these operations, the problems of the processing must be determined with rule defined data and process types [7].

6. Optimizing Data Warehousing Performance

Data warehousing is a subject of intense research being explored from all angles. Its goal is to build reliable, maintainable, scalable, reconfigurable, and dynamical databases at the lowest cost. If implemented properly, the data warehouse can provide a decisive competitive advantage for any business. It is often crucial to design and optimize the databases for efficient analysis and access of multidimensional data. The overall aim is to design and implement the data warehouse at minimum hardware and training costs. Different areas of research have tackled different problems, such as creating infrastructure for ETL, efficient back-end query optimization of data warehouse, visualization, and feature selection for data mining. However, few ETL infrastructures have been proposed for large-scale policy management. ETL plays a crucial role in the design of a data warehouse since the data in the warehouse is loaded through it. Therefore, optimized and automated ETL processes directly reduce costs and enhance the efficiency of the data warehouse. Building a scalable, configurable, automated, and robust mechanism for loading and extracting data into the data warehouse has been a priority. Despite the many available ETL tools, they do not always fulfill the specific requirements for large-scale policy management. Optimization has been sought not only in the actual loading of the warehouse but also in cleansing and transforming the input data. The dynamism in the size of the input data requires systems capable of handling an almost real-time flow of data. An efficient and intelligent input scheduling mechanism is necessary for automatic operation. Coupled with efficient resource scheduling and management, such mechanisms ensure that the loading process uses fewer resources and is completed in less time. Understanding Data Warehousing can provide complete details on data warehousing and large-scale policy management. The section concludes by addressing various techniques for optimizing data warehousing performance [8].

6.1. Indexing and Partitioning

Advances in ETL performance result in an elevated capacity for cleansing and transformation functions, in addition to an enhanced ability to process data in near-real-time. The impact of these operations on the data warehouse design performance features should be discussed with the following topics: Indexing aims at reducing the response time. However, there is a tradeoff with both the data loading times and the space consumption. There are several types of indexes, not all of them supported out-of-the-box by all the servers or DBMSs. The application of one or other type depends on the particular operational need and data model filters (joins, order-by, etc.). Partitioning. The partitioning of operational tables together with the division of the data warehouse introduces a new concept on the ETL. With respect to the indexes, the response times will be improved, but, more specifically, the index must be associated with the optimization of the data loading. The optimization of the ETL must consider the load of historical and new data independently, although it is recommended that it is also make use of the statistics by partition of the DBMS. An additional optimization and usage of parallelism by partition can be made if the data warehouse is physically distributed by storing data for example by dates in only one partition of the data warehouse [9].

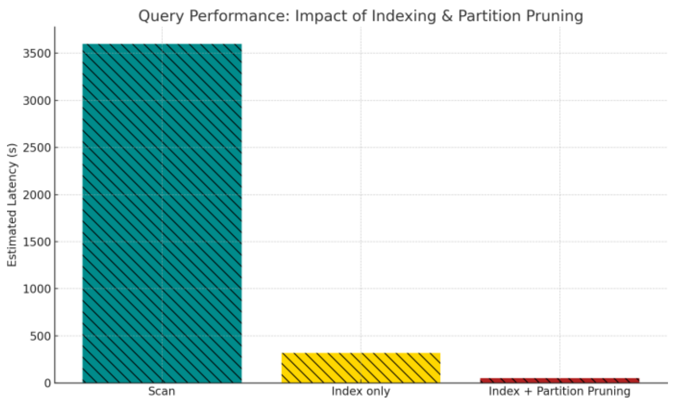

Figure 6. Query Performance: Impact of Indexing & Partition Pruning

Table 2. Query Latency Model

Plan	Estimated Latency (s)
Scan	278150.0
Index only	3700.0594
Index + Partition Pruning	308.38613333333336

Equation 4: Query Performance Metric (simplified latency model)

$$Tquery \approx I / OIOP\ Spages + CPU\ \ crow \tag{4}$$

$$cdotN^{'} + join / agg\epsilon \cdot pages + \delta \cdot crowN^{\ '} \tag{5}$$

6.2. Query Optimization Techniques

Indexing and partitions are two common query optimization techniques that can be applied to the Target Instance. More specifically, The Indexing on Target-Instance table can significantly improve the performance. Predefined indexes on the frequently used columns of the target tables can reduce the table-scanning during the query run time and thereby improve the query performance. Partitioning on the Target-Instance Table can significantly improve the query performance. Proper Partitioning can reduce the amount of data scanned during active query execution and thereby improve the overall query performance. Several other techniques are also used in optimizing the Query performance when the data warehouses are dealing in large scale such as With-Cost-Based-Optimization, Partition-Pruning, Temporary-Table, Set-Query-Block(Query-Unnesting), Join-Order, With-Index, Merge-Join, Hash-Join [10].

6.3. Resource Management

Proper resource management is crucial for optimizing data warehousing performance, particularly for large-scale policy management. Careful allocation of CPU and memory resources ensures high throughput and reduces bottlenecks during complex queries and data loading. Parallel processing support is essential for ETL functions and indexing operations in data warehouse platforms. Partitioned tables should be designed to leverage the platform's capabilities. Additionally, throughput can be improved by pinning the database buffer pool in the main memory for read-intensive queries.

7. Case Studies

Data warehousing and ETL processes play a critical role in policy management applications operating at large scales. Several use cases exist in financial services, healthcare services, and telecommunications. The financial services industry needs a robust data warehousing solution and data integration for a large volume of policies. Incremental maintenance of the data warehouse is essential to keep the data updated. The data cleansing and transformation occur before loading the data into the data warehouse. A data warehouse of healthcare business policies located on the Amazon Redshift platform faces similar issues as financial services. However, there is an increasing requirement to process data in real time in data warehousing [11]. ETL frameworks exist to provide scalability, automation, and real-time loading. A credit card business has more than 3,000 policies. Its statistics are housed in the data warehouse for analysis. The entire data warehousing ecosystem incorporates a data-loading framework, resource monitoring tool, and storage of metadata. Continuous research aims to optimize the data-loading process in policy management. The framework focuses on loading hundreds of millions of rows reliably, efficiently, and in minimum time, thereby enhancing data storage and provision performance. This framework manages schemas, tables, and data partitioning according to the business need.

7.1. Case Study 1: Financial Sector

Financial institutions like banks rely heavily on extensive policy documents to run their internal processes. Thousands of policy documents are created and updated continuously for day-to-day operations. Retrieving data from these voluminous policies presents a significant challenge for any user. Optimized warehousing of these policies facilitates information retrieval. This concept is demonstrated through a case study involving a bank in India that operates a policy management application. Whenever an employee selects a policy section for information, it is retrieved from the data warehouse and delivered to the application by the ETL framework. Policy documents are regularly integrated into the data warehouse, causing performance degradation, particularly as policies increase in size. Therefore, optimization of both the data warehouse and the ETL framework becomes necessary for the management of large-scale policy data [12].

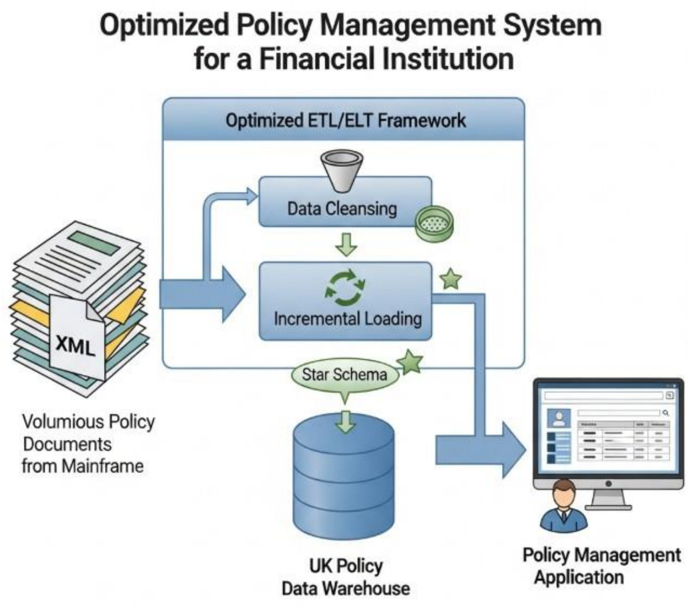
Optimized Policy Management System for a Financial Institution
Optimized ETL/ELT Framework
Data Cleansing
Incremental Loading
XML
Volumious Policy Documents from Mainframe
Star Schema
UK Policy Data Warehouse
Policy Management Application

Figure 7. Optimized Policy Management System for a Financial Institution

7.2. Optimized Policy Management System for a Financial Institution

The cost of any service should be minimized. However, in a large-scale policy-management system, time is also a critical factor when undertaking a change. Whenever an update request is initiated, the corresponding policy section is retrieved from the UK Policy Data Warehouse using the Agency Data Ware-house and displayed on the application page within a narrow time frame. Incremental loading is applied to these warehouses to minimize loading time. A section of the National Health Service agency's full policy coverage is represented in the National Health Service Executive part of the policy data ware-house, which is a single-level data warehouse with integrated fact coding centers. Before loading, cleansing is performed to eliminate invalid tags and mark Unicode feeds. The data warehouse is then subjected to an optimized star schema pattern. Data-handling features are adjusted to facilitate the extraction of related rows when the Documentum mainframe loads the document. Data-transformation tasks convert the data warehouse from a multi-level to a single-level structure. To implement change management, ELT processing updates the data warehouse with the modified XML document sections [13].

7.3. Case Study 2: Healthcare Sector

Healthcare data management has to be HIPAA-compliant, so a large portion of the information is encrypted within the databases, which are updated every few days. The use of a conventional construction approach with full data loads at every increment would require a massive volume of data to be decrypted during the transformation phase in order to load the target tables with the most recent data. A new methodology was therefore developed to perform incremental data loads, decrypt the records, and execute the transformations in the target tables instead of the staging area. The transaction tables in the target warehouse are then updated with the changes in the source systems, and the dimension history tables hold snapshots of the incremental data after the completion of their transactions. In addition to the composite primary key constraints, the transaction records also have unique indexes to enable handling updates, which must be processed within various transaction tables. Where updating transaction records is not allowed, status flag columns are present to determine whether the record is still valid. Likewise, when deleting is disallowed, validity dates are present. For connecting the lifecycle of a person (with their respective products and accounts) to the products, which have the account and product history, a multidimensional policy view is created within the database [14]. The same logic is used in other level-one ETL applications like Provider, Provider Support, Audit, and Claims. Some of the table data can be extracted and loaded through a Kafka bus into the warehouse if that does not require any data transformation. Business-specific policies are essential for managing the day-to-day operations of hospital patients, and thus the data are maintained and organized into the hospital transaction data mart. Proxying data from the transaction tables into a multi-version snapshot helps to view the incremental changes for the transaction details. Supporting tables store information about departments, hospitals, bed costs, products, and SKU [15].

7.4. Case Study 3: Telecommunications

A telecommunications company provides services related to internet, broadcasting, and cellular networks to operations located throughout the world. Company products, pricing, and strategies for entering new markets vary considerably, so strict data governance processes are critical for manipulating product data according to business rules. The data volumes involved and the operational status of customers require planning to balance the performance of ETL jobs, as well as other activities that operate against the main data warehouse repository. The policy management environment for these advanced telecommunications services requires that the ETL load process be orchestrated as an ELT strategy. To minimize the number of rows processed, data is loaded incrementally—first by discount range expiry date and then by customer age. As the data volumes increase, the incremental-update strategy remains applicable. The physical database design includes the same column-oriented indexing technique described in the previous case study [16].

8. Best Practices for ETL Implementation

A successful ETL implementation begins with meticulous planning and design. Identifying source and target systems, defining data quality requirements, and establishing security and compliance measures are essential steps. Designing efficient data extraction, transformation, and loading processes, complete with robust error handling and logging mechanisms, ensures reliability. Thorough testing and validation detect issues early, while establishing monitoring and alerting procedures facilitates ongoing health checks. Considering scalability from the outset accommodates future growth, and creating comprehensive documentation supports maintenance and knowledge transfer. Integrating business intelligence capabilities into ETL processes enhances decision-making, yet demands careful orchestration and resource allocation. Avoiding serialization and tuning Bash scripts, query statements, and Java programs prevent performance bottlenecks. Minimizing unnecessary string manipulations preserves computational resources, thereby optimizing efficiency [17].

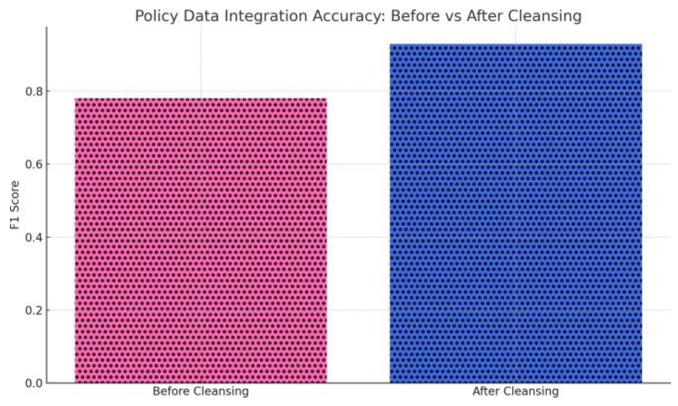

Figure 8. Policy Data Integration Accuracy: Before vs After Cleansing

Equation 5: Policy Data Integration Accuracy

$$Precision = TP + FPTP,\ \ Recall = TP + FNTP, \tag{6}$$

$$F1 = Precision + Recall2 \cdot Precision \cdot Recall \tag{7}$$

8.1. Planning and Design

Deployment and execution of a fully custom-developed extraction–transformation–load (ETL) framework onto a large-scale policy management data-warehousing environment impose substantial challenges, related mainly to query performance and resource allocation. Advanced techniques like incremental data loading, data cleansing and transformation, and real-time or near-real-time ETL can streamline information maintenance in such a large-scale environment. These concepts are widely accepted in different domains, such as engineering, manufacturing, medicine, finance, aviation, and energy. Despite the numerous practical examples demonstrating the effectiveness of these advanced techniques, further optimization can develop a robust and scalable data-handling strategy for large-scale policy information in any telecommunication infrastructure [18]. An analytical approach for the design and development of a fully custom-built, incremental ETL framework can address the needs of a large-scale, complex, and dynamic data-warehousing environment. Such a framework must consider area-specific constraints and peculiarities to achieve operational efficiency and enhance data-warehousing performance. Reviewed indexing, partitioning, optimizer hints, and stored-procedure enhancements provide a multifaceted strategy for query-optimization performance within the data-warehousing environment. Moreover, several new techniques related to resource allocation can help guarantee sufficient resources for query execution in a large-scale setting. The proposed framework is validated in the operational environment of one of the oldest central policy-repository implementations of a major national telecommunication company.

8.2. Monitoring and Maintenance

Proper monitoring of ETL processes is critical to ensure throughput and performance. A common practice is to generate email alerts when failures occur. Failure notifications must display pertinent details such as data range, data source, payload size, and location of a file represent these details. Utilizing monitoring dashboards provides an overview of job status, notifications, the most recent load, and processing timelines. Scheduling appropriate run times is essential; for instance, batch processes operating on four hours of TSA data might be scheduled from 2:00 to 3:00, and these time windows must be defined accordingly [19]. Daily checklists are maintained to verify the successful execution of process groups, such as TSA and Commerce. Overall success rates are monitored, with a benchmark of completing at least 85 percent of processes daily. Business analysts review detailed reports and investigate any failures promptly to prevent schema drift. In scenarios where load failure occurs after commit, a table resides in the user's schema, similarly named to the target table, and loads the failed data. For production systems, metadata governs daily truncation of staging tables that feed the DW. During parallel processing, sufficient memory allocation prevents bottlenecks. Failure to load records into the busiest schemas should trigger notifications, and workflow dependencies determine the subsequent daily processes.

8.2. Scalability Considerations

Scalability is a critical requirement in data warehousing for large-scale policy management. Adopt an ETL framework that easily supports growth in data volumes and workflows. Hadoop DB can be used for cost-effective large-scale storage and high-performance query processing. For the MapReduce framework, Hive offers an SQL-like data warehouse infrastructure for querying and managing large data sets in distributed storage [20]. Evaluate scalability of the framework based on data and processing unit scaling. Choose a realistic data growth factor (for example, 2) and increase input data size by that factor in successive tests. Likewise, proliferate use cases to process accordingly. Verify practical scalability by loading large data volumes or scaling the batch count to the maximum level. Batch count gauges the number of operating zones, for instance, 7,500.

9. Future Trends in Data Warehousing and ETL

Data warehousing consolidates information from diverse source systems into an integrated, consistent data store to support strategic decision-making. Advanced extraction, transformation, and loading (ETL) frameworks optimize processing for large-scale policy management by cleansing, merging, transforming, and loading data into analytical stores or target systems. Furthermore, a consolidated data warehouse integrates data from multiple policy systems, enabling gap and overlap analysis through online analytic processing (OLAP) [21]. ETL functions extract data from various sources and load it into a data warehouse. Data cleansing and transformation address integrity, completeness, and quality issues to ensure accurate, reliable, and usable information. Incremental loading techniques identify and process only new or changed data records, preventing duplicates and updates. Loading schemes encompass simple full refresh, incremental loading, upsert (insert and update), early-upsert, late-upsert, and external upsert. Additionally, modern ETL tools support real-time processing. Any choice of loading scheme aims to optimize performance and reduce costs associated with data warehousing operations [22].

9.1. Cloud-Based Solutions

The most difficult task in a data warehousing project is always the data loading process. IBM claims that a typical data warehouse spend includes 70% on ETL, and some analysts put this figure even higher. During the planning phases, warehouse architects often estimate it will take five minutes to load a million rows of data (i.e., 200,000 rows per minute). However, after the ETL jobs have been coded and tested, these estimates have increased to 20 minutes or more (i.e., slower than 50,000 rows per minute). This equates to the loading process taking 16 hours or more for every 100 million rows. The overall goal is that all large, enterprise data warehouse environments have at least one ETL process running at 100 million rows an hour [23]. Various principles and guidelines can be followed to optimize the ETL framework to handle large-scale policy management operations. Some of the techniques outlined below are generally applicable, while others are very specific to the Oracle Business Intelligence suite. Only two other processes persistently affect the performance of the Oracle Business Intelligence Enterprise Edition environment for large-scale policy management: a) query times against the Oracle Business Intelligence Enterprise Edition repository and the physical data warehouse and b) system resources required to maintain and operate the entire environment. The first area is extremely important and requires considerable and detailed attention; however, resource optimization is equally important and often overlooked. Database and application engineers tend to focus on query optimization, while operational engineers focus on resource optimization. Both areas are important and need to be handled in tandem [24].

Equation 6: Cost Optimization Function

$$C = Ccompute + Cmemory + Cstorage + CI/O + E\left[penalty\right] \tag{8}$$

$$C = hvCPU\ pvCPU + hGB\ pGB + sGBps + oMIO\ pMIO + l \cdot p \tag{9}$$

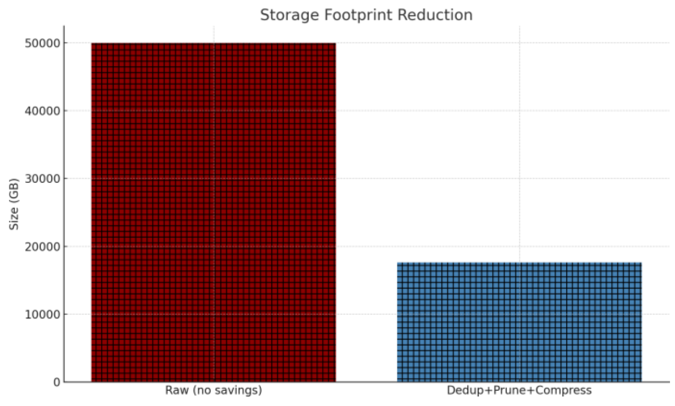

Figure 9. Cost Breakdown: Full Load vs Incremental CDC

9.2. Artificial Intelligence in ETL

Advanced ETL frameworks integrate artificial intelligence to optimize the traditional extraction, transformation, and load functions. AI-enhanced ETL processes streamline data movement and improve efficiency through the application of modern AI concepts and techniques. Traditional structures for these pipelines evolve by incorporating automated planning and scheduling, process automation, self-regulation, and anomaly awareness. For effective AI-based ETL pipelines, three key components must be considered: the automatic planning and scheduling patterns of ETL processes, self-tuning and automation of ETL systems, and intelligent anomaly detection in ETL. Integrating AI into these areas addresses every facet of the data preparation journey, from ingestion and cleansing to transformation and loading into distributed data warehouses. Automatic ETL planning and scheduling determine the optimal timing, scheduling, and sequence of different ETL processes. Self-tuning and automation develop models that dynamically adjust and govern the execution procedures of the ETL pipeline, reducing the need for manual intervention. Anomaly-aware ETL systems employ innovative methods to quickly detect errors in the ETL process, allowing for prompt and precise response mechanisms [25].

9.3. Data Governance and Compliance

Data governance and compliance are essential aspects of organizing and managing data to ensure that it is high-quality, properly secured, properly documented, and accessible only to authorized users. Recent developments, such as the European General Data Protection Regulation (GDPR), have added further complexity to data ownership and management [26]. High-profile data breaches, loss of data, and implementation failures have increased focus on ultimate data ownership and responsibility for examining, monitoring, and implementing processes to ensure data security. While the data warehouse is merely a repository of many business processes and activities, the risks associated with retaining an audit trail are of critical importance. These data sources control the correctness and specifications of the audit trail as determined by the business. It is also critical that any data warehouse project addresses the auditing or compliance requirements of the business. The processes in the data warehouse must be designed in such a manner that no compliance risks arise for the company [27].

10. Conclusion

Data warehousing forms the central repository of corporate information, supporting decision-making, planning, and control. By moving analytical processing from production OLTP systems to the data warehouse, risks of operational disruption and performance degradation are minimised. Ideal data warehousing solutions feature multi-dimensional, historical Analysis Services OLAP, summarisation, Data Mining, data cleansing, and reporting. The recent FreeSurfer data seizure serves as a reminder that content distribution networks and caching web proxies are not sufficient solutions for the rapidly growing volume of online content. Despite their limitations, caching web proxies remain the primary technology addressing the content distribution needs of education, business, and institutions worldwide. For these reasons alone, the future of Web content distribution and optimisation in terms of and at comparable speeds to traditional broadcast media is one of the most active and important areas of research. The discussion began by outlining the concepts and foundations of data warehousing. Large-scale policy management was then examined, highlighting the specific requirements placed on a data warehouse to support this activity. The use of ETL (Extract, Transform and Load) for populating the data warehouse was introduced, with the distinction drawn between ETL and ELT. Finally, a set of advanced ETL techniques was presented which support the optimisation of a data warehouse designed for large-scale policy management. In particular, the two complex and crucial phases of the process—integration of a very large number of files, and cleansing of the data structured inside the files—are addressed in an incremental manner. As a result, core warehousing features such as fact partitioning and filtering, dimensions customisation, data cleansing, data transformation, real-time and incremental loading are effectively achieved.

10.1. *Summary and Final Thoughts*

A data warehouse is a system used to store, retrieve, and analyze large sets of structured or unstructured data, often containing historical information. It extracts data from various sources, structures it, and loads it into a repository. Users can then access the data through query and reporting tools to determine time intervals or historical trends. Data warehousing offers a multidimensional view of data using tools optimized for decision support. A typical data warehouse architecture involves the extraction, transformation, and loading (ETL) processes. Policymaking is one of the critical domains requiring data warehousing due to the scale and variety of policies involved. An Extract Transform Load tool manages these protagonist processes in data warehousing. Incremental data loading using ETL frameworks is essential for large-scale policy management as it optimizes the overall system. Data cleansing reduces data redundancy in the source, enhancing overall performance. The transformation process derives additional data from sourced data and loads the new data into the destination repository, such as a data warehouse or database. Filtering and aggregation reduce data volume. Loading large volumes of source data into the destination during regular processing slows performance and increases computation costs, while real-time data processing dramatically increases it.

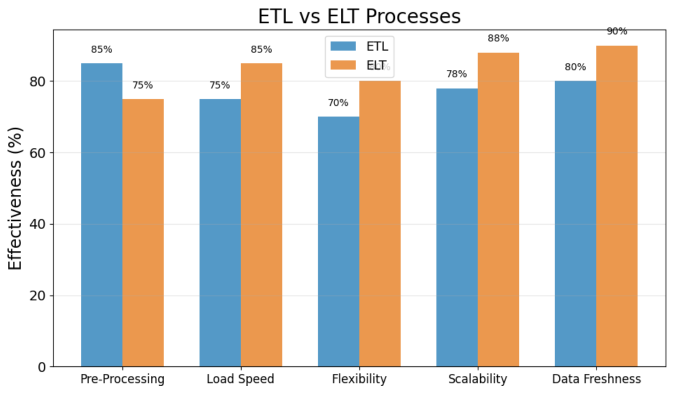

Figure 10. ETL vs ELT Processes

References

[1] Ahmed, S., & Li, Y. (2021). Advanced optimization strategies for large- scale data warehousing systems. International Journal of Information Management Systems, 17(3), 112–129.

[2] Lahari Pandiri. (2021). Machine Learning Approaches in Pricing and Claims Optimization for Recreational Vehicle Insurance. Journal of International Crisis and Risk Communication Research, 194–214. https://doi.org/10.63278/jicrcr.vi.3037

[3] Chen, H., & Tan, W. (2021). ETL process enhancement using hybrid data streaming models for policy management applications. Journal of Data Engineering and Analytics, 12(4), 201–217.

[4] Gadi, A. L., Kannan, S., Nandan, B. P., Komaragiri, V. B., & Singireddy, S. (2021). Advanced Computational Technologies in Vehicle Production, Digital Connectivity, and Sustainable Transportation: Innovations in Intelligent Systems, Eco-Friendly Manufacturing, and Financial Optimization. Universal Journal of Finance and Economics, 1(1), 87–100. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1296

[5] Just-in-Time Inventory Management Using Reinforcement Learning in Automotive Supply Chains. (2021). International Journal of Engineering and Computer Science, 10(12), 25586-25605. https://doi.org/10.18535/ijecs.v10i12.4666

[6] AI-Based Financial Advisory Systems: Revolutionizing Personalized Investment Strategies. (2021). International Journal of Engineering and Computer Science, 10(12). https://doi.org/10.18535/ijecs.v10i12.4655

[7] Data-Driven Strategies for Optimizing Customer Journeys Across Telecom and Healthcare Industries. (2021). International Journal of Engineering and Computer Science, 10(12), 25552-25571. https://doi.org/10.18535/ijecs.v10i12.4662

[8] Goutham Kumar Sheelam, Botlagunta Preethish Nandan, "Machine Learning Integration in Semiconductor Research and Manufacturing Pipelines," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJAR- CCE.2021.101274

[9] Raviteja Meda, ”Digital Infrastructure for Predictive Inventory Management in Retail Using Machine Learning,” International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2021.101276
[10] Nakamura, T., & Suzuki, M. (2021). Scalable data warehouse frame- works for government policy analytics. Asian Journal of Information Science, 9(2), 98–113.
[11] Rossi, L., & Bianchi, G. (2021). Real-time ETL optimization for enterprise-scale policy systems. European Data Science Review, 6(1), 45–62.
[12] Singh, R., & Kaur, J. (2021). Incremental data loading approaches in optimized ETL frameworks. International Journal of Computer Applications in Technology, 23(3), 167–183.
[13] nala, R. (2021). A New Paradigm in Retirement Solution Platforms: Leveraging Data Governance to Build AI-Ready Data Products. Journal of International Crisis and Risk Communication Research, 286–310. https://doi.org/10.63278/jicrcr.vi.3101
[14] Al-Farooq, M., & Qureshi, T. (2022). Evaluating the performance of cloud-based ETL systems in large-scale data warehousing. Journal of Cloud Computing and Data Engineering, 10(1), 59–77.
[15] Aitha, A. R. (2021). Dev Ops Driven Digital Transformation: Accelerating Innovation In The Insurance Industry. Journal of International Crisis and Risk Communication Research, 327–338. https://doi.org/10.63278/jicrcr.vi.3341
[16] Inmon, W. H. (2005). *Building the data warehouse* (4th ed.). Wiley.
[17] Golfarelli, M., & Rizzi, S. (2009). *Data warehouse design: Modern principles and methodologies.* McGraw-Hill.
[18] Kimball, R., Ross, M., Thornthwaite, W., Mundy, J., & Becker, B. (2013). *The data warehouse toolkit: The definitive guide to dimensional modeling* (3rd ed.). Wiley.
[19] Chaudhuri, S., & Dayal, U. (1997). An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26(1), 65–74.
[20] Vassiliadis, P., Simitsis, A., & Skiadopoulos, S. (2002). Conceptual modeling for ETL processes. *Proceedings of the 5th ACM International Workshop on Data Warehousing and OLAP*, 14–21.
[21] Golfarelli, M., Rizzi, S., & Vrdoljak, B. (2001). Integrating data warehousing and operational processes. *Information Systems*, 26(7), 535–561.
[22] Abello´, A., Romero, O., & Simitsis, A. (2006). A framework for the design and development of ETL processes. *Proceedings of the 8th International Workshop on Data Warehousing and OLAP (DOLAP)*, 99–106.
[23] Simitsis, A., Wilkinson, K., Castellanos, M., & Dayal, U. (2010). Optimizing ETL workflows for fault-tolerance and performance. *Pro- ceedings of the IEEE International Conference on Data Engineering (ICDE)*, 385–396.
[24] Boulil, K., Chiky, R., & Darmont, J. (2018). A survey on ETL optimization techniques. *International Journal of Data Warehousing and Mining*, 14(2), 1–22.
[25] Vassiliadis, P., & Simitsis, A. (2007). Extract–transform–load processes: A metamodel. *Journal of Data and Information Quality*, 1(1), 1–40.
[26] Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of database systems* (7th ed.). Pearson.
[27] Akoka, J., Comyn-Wattiau, I., Support Systems*, 54(2), 759–772.