Scientific Publications

Universal Journal of Business and Management, 2022, 2, 1347
www.scipublications.com/journal/index.php/ujbm
DOI: 10.31586/ujbm.2022.1347

Review Article

Cloud Native ETL Pipelines for Real Time Claims Processing in Large Scale Insurers

Avinash Reddy Aitha *

Software Engineer, US Bank, Minneapolis, MN 55402, United States

*Correspondence: Avinash Reddy Aithai (vinash.r.a@gmail.com)

Abstract: Cloud native ETL pipelines support the extract and transform phases of real time claims processing in large scale insurers. The cloud native approach offers dramatic improvements in scalability, reliability, resiliency and agility as well as seamless integration with the diverse set of data sources, destinations and technologies characteristic of large scale insurers. The ETL process extracts data from source systems such as core transaction, fraud, customer and accounting processes, transforms the data to create a usable format for analytics and other applications, and loads the resulting tables into business intelligence or data lake systems for subsequent storage and analysis. By addressing these two phases of the overall ETL process, cloud native ETL pipelines can provide timely, reliable and consistent data to data scientists, actuaries, underwriters and other analysts. Real time processing represents a key priority within the overall claims process: faster, more accurate claim approvals reduce insurer costs, improve customer service and enhance premium pricing. As a result, a variety of claims related use cases are moving from batch to real time.

Keywords: Cloud Native, ETL, Real-Time Processing, Claims Processing

How to cite this paper:
Aitha, A. R. (2022). Cloud Native ETL Pipelines for Real Time Claims Processing in Large Scale Insurers. *Universal Journal of Business and Management*, *2*(1), 50–63.
DOI: 10.31586/ujbm.2022.1347

Received: July 16, 2022
Revised: September 23, 2022
Accepted: October 28, 2022
Published: December 24, 2022

1. Introduction

Real time processing of claims is one of the distinguishing aspects of large scale insurers. It is particularly so for those who do not write before the event business. Processing claims in real time involves assessing the impact on the profitability of the deal and on the further reinsurance contractings of the insurer. Making claims processing real time is necessary for both insurance and re-insurance companies, irrespective of whether they provide ”before the event cover” or not. Entering the claim in real time also facilitates management functions such as control of the loss ratio. The importance of cloud native extract and transform pipelines lies in the need to put data into the platform with the least potential for error and with the ability to scale on demand. Claims are routed through the platform for reassessment. Data can come from point-of-sale systems, policy administration systems, other internal systems such as customer relationships management tools, or public and commercial sources [1].

2. Ease of Use

2.1. Background and Significance

Claims Processing is a core business function within an Insurance company which plays a vital role in timely serving the end customers especially in the scenarios of Natural calamities, War etc. There are many challenges associated with Claims Processing in Insurance companies like Data Quality, Integration of data across various product lines, bidding for the data in moments of catastrophe and the ability to real-time processing of data and offering underwriting decisions or Claims decisions. The architecture of Cloud provides building blocks like Scalability, On- Demand Availability, Global Accessibility, High Availability, Reliability, Disaster Recovery, Flexibility, Elasticity, Security etc., and thus addresses these challenges. Extract, Transform and Load (ETL) is a process that combines data from multiple sources into a destination data store. Transformation is the core of the process as it involves discarding the erroneous data, combination of data from multiple sources as also Business Data Quality enhancements. This is followed by the Load phase wherein the cleansed data is made ready for consumption by the Business Applications. It is evident that there is a business need to assess the feasibility of Extract and Transform components in the Cloud for a Large Life Insurer with Global footprint in order to alleviate the existing challenges and support Real-Time claims processing [2]. Cloud native architecture enables building and running applications that exploit the advantages of the cloud computing delivery model. Its key benefits include improved scalability, elasticity, resilience, flexibility, operational agility and faster delivery to market. Extract, Transform, Loading (ETL) is a process that combines data from multiple sources into a destination data store. Streaming data platforms, a real-time data pipeline that get continuously driven with data flows enable the Extract and Transform phases in the ETL pipeline. A Cloud native ETL approach supports the real-time processing of Claims Data of an Insurer and in the process addresses the existing challenges of Data Quality, Integration of data across various product lines, bidding for data in moments of catastrophe and the need for real-time processing of data and offering underwriting decisions or Claims decision [3].

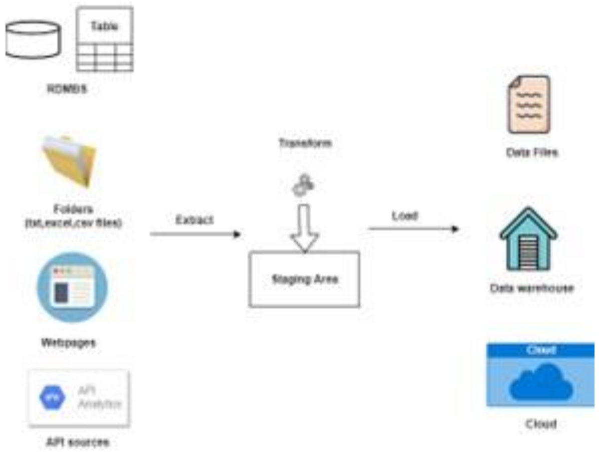

Figure 1. ETL Pipeline

Equation 01: ETL throughput

ETL throughput • = f(Data,Volume, Processing Speed, Scalability).

Formalization

Let incoming rate be λ records/s.

Let each worker process at rate μ records/s.

With N workers, coordination overhead reduces ideal linear scaling. Model efficiency by

$$\eta(N) = 1 + \kappa(N-1)1, \kappa > 0(\mathit{overhead\ factor}). \tag{1}$$

Effective capacity:

$$C(N) = N\mu\eta(N) = \frac{N\mu}{1+\kappa(N-1)} \tag{2}$$

Achieved throughput is the min of arrival and capacity:

$$T(N) = min\{\lambda, C(N)\} \tag{3}$$

What I produced

Table: workers N, C(N), T(N), ρ(N).

Line chart: f(g)C(N) and h(i)T(N) vs jN.

orkers N	capacity C(N) rec per s	throughput T(N) rec per s	utilization rho
1	500.0	500.0	1.0
2	990.09900990099	990.09900990099	1.0
3	1470.5882352941176	1470.5882352941176	1.0
4	1941.7475728155337	1941.7475728155337	1.0
5	2403.846153846154	2403.846153846154	1.0
6	2857.142857142857	2857.142857142857	1.0
7	3301.88679245283	3301.88679245283	1.0
8	3738.3177570093458	3738.3177570093458	1.0
9	4166.666666666666	4166.666666666666	1.0
10	4587.155963302752	4587.155963302752	1.0

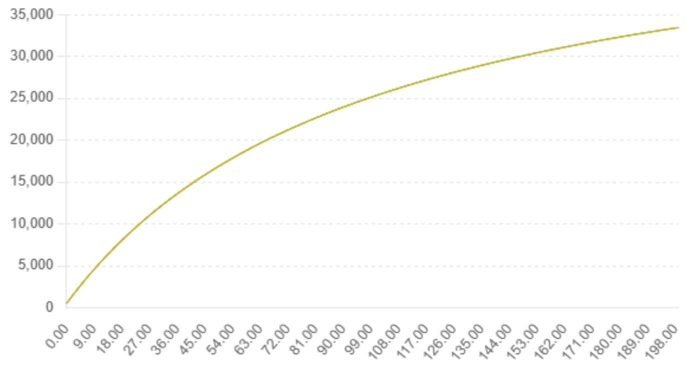

Figure 2. Throughput vs Workers

3. Understanding Cloud Native Architecture

Cloud native is an approach to building and running applications that fully exploit the advantages inherent in the cloud computing delivery model. Cloud native enables companies to create and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Technologies such as containers, service meshes, microservices, immutable infrastructure, and declarative APIs exploit robust automation, DevOps, and agile techniques. These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil. For ETL pipelines that support real-time claims processing in large-scale insurers, cloud native architecture offers horizontal scaling to accommodate growth and growth forecasts, and natively built resiliency to handle natural down times while maintaining performance and quality of service [4].

ETL stands for extract, transform, and load; but when large scale insurers perform real-time processing on claims data, only the extract and transform phases are employed. Extract or data acquisition involves capturing a continuous stream of data records from a claims application within the insurer. Transform involves cleansing and filtering, format and schema changes, and data masking and enrichment of claim records. Real-time data processing in claims handling has become increasingly important particularly where the processing of claims data from multiple lines of business can trigger analyses that considerably reduce exposure to risk or provide invaluable insight into the claim. Combining the benefits of cloud native architecture with technologies that support the real-time movement of data through the extract and transform phases has made it possible to design and implement ETL pipelines that support real-time claims processing in large-scale insurers. The challenges of data quality, data integration, and scalability frequently encountered during claims processing can likewise be overcome by adopting cloud native design principles.

3.1. Definition and Principles

Cloud native is a term describing business applications built specifically to run and take advantage of cloud computing architecture. Applications and services designed with cloud native methodologies utilize the core principles of cloud computing—scheduling, automation, disposal, orchestration, and dynamic management—to fully exploit the advantages of cloud platforms. A cloud native ETL pipeline is highly scalable, resilient to failure, flexible to change, loosely coupled, and designed for ease of upkeep. ETL (extract-transform-load) refers to the process of taking data from various sources, converting it into the desired format, and sending it to a destination system. The extract phase focuses on reading and capturing data, while the transform phase includes applying filters, cleansing data, identifying missing data, and enriching data to suit the requirements of the destination. The load phase involves writing the newly transformed data to the target location. In high-volume claims processing engines, loads are executed in real-time [5].

3.2. Benefits of Cloud Native

Cloud Native architecture is a natural fit for ETL workloads through soft isolation and a deferred commitment to capacity and physical hardware. These characteristics translate into a set of benefits critical to process claims at scale. Soft isolation ensures a workload failure cannot cascade to other workloads running on the same hardware, enabling a granular scale-up of high-throughput, high-volume ETL workloads. Deferred commitment to capacity means an ETL workload does not consume a designated amount of CPU, RAM, or Network during idle times, resulting in a real reduction in cost. Deferred commitment to physical hardware ensures the workload is not tied to any specific physical machine or datacentre, enabling resiliency and data-loss prevention options in the post-extract stages of ETL. Aside from scalability, performance, resiliency, and economics, cloud native supports the cultural shift to DevSecOps Transform (DST) through increased transparency for enhanced security. By providing an auditable single source of truth around the runtime configuration of ETL pipelines, cloud native removes the need for manually created documents maintained outside the runtime environment. Configuration changes become very granular, can be reviewed, and systematically merged with the production environment. This aids in early detection of bugs introduced by code and configuration changes, and also along the CI/CD chain for ETL pipelines [6].

Equation 02: Cloud resiliency

$$Resiliency\ cloud = Latency\ Redundancy + Elasticity / Failover. \tag{4}$$

Normalized index (dimensionless)

Normalize latency by a target L0 (I used 100 ms): L′ = Latency / L0.

Weight inputs to reflect importance: $w_1 + w_2 + w_3 = 1$. Define the index:

$$RI = L' w_1 R + w_2 E + w_3 F \tag{5}$$

Here R=redundant copies/AZs, E=scale actions per minute, F=probability failover succeeds within SLO.

What I produced

Table: On-Prem vs Cloud (Std/Adv): inputs + Resiliency Index.
Bar chart: Resiliency Index by environment.

System	Redundancy	Elasticity	Failover prob	Latency ms	Resiliency Index
On-Prem	2	2	0.95	150	1.1233333333333333
Cloud (Std)	3	6	0.99	100	3.297
Cloud (Adv)	4	10	0.999	80	6.124624999999999

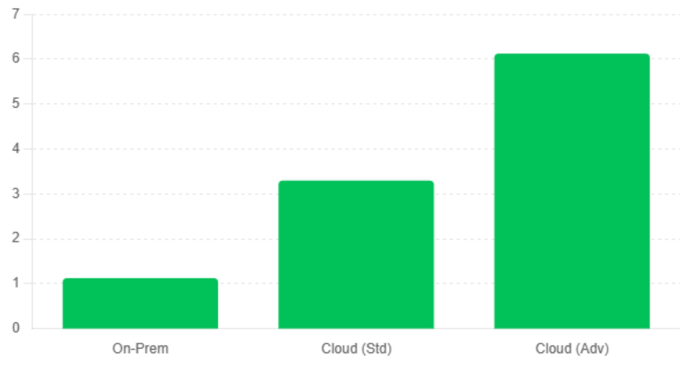

Figure 3. Resiliency (illustrative)

4. ETL Process Overview

The Extract phase of an Extract, Transform, Load (ETL) process grabs the data from the source and copies it some where else. It might move the data into a staging area entirely segregated from any data warehouse or analytics clusters. Or, it might ingest the extracted data into a messaging platform, such as Kafka. The Transform phase of an ETL process is all about structuring the data and making it usable. It involves cleansing the data, deduplicating and flattening; filtering out the values not required for analysis; converting character sets and encoded values into the correct coding to interact with the loading system; masking sensitive data such as names and credit card numbers; and validating data for correctness and integrity. Real-time data processing involves the continuous in- put and processing of streaming data with low enough latency to be able to respond within some reasonable timeframe. It is supremely well suited for cases where there is a continuous input of data and a need to see updates in real time. Many organizations deploying real-time analytics have timesensitive goals of one kind or another, such as reducing the risk-value of fraudulent transactions, delivering recommendations or offers, or predicting the need for a bundled call center personnel before a surge in calls. In other cases, near real-time response is desired to show updates on dashboards practically as they happen. Scaling such solutions to ingest not just individual events, but aggregated transaction level summaries for high- volume flows of real-time usage, can be a key requirement for broad adoption. The ever increasing adoption of smart edge devices and the emergence of next generation connectivity and 5G has led to new business possibilities for telecom providers. The ability to perform deep real-time analytics on the ever changing network and device usage can significantly reduce operating costs, as well as diminish support times for quorum clients planning a move to digital services [7].

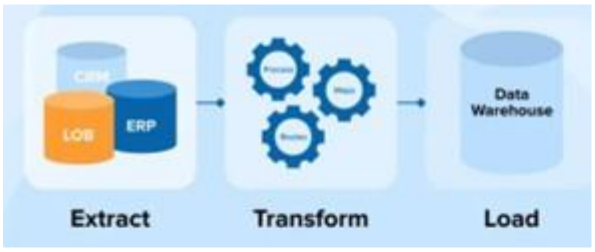

Figure 4. ETL Process (Extract Transform Load)

4.1. Extract Phase

In cloud native ETL pipelines for real time insurance claims processing, the extract phase requires scalable tools and technologies capable of working with downstream technologies, along with appropriate data acquisition and replication techniques. Claims are initiated by customers of the insurance carrier, who request benefits according to their insurance coverage. Digital methods for filing insurance claims encourage customers to upload documents directly to the insurance carrier. To complement these digital initiatives, the insurance company should also be able to ingest insurance claims data from one or more internal and external systems that may be supporting the insurance claims adjudication process. Data extracted from source systems is replicated to a cloud native streaming platform for low-latency processing. Ingested claims data is identified as source and metadata captured in a cloud native metadata management platform, where the source inherits important security parameters such as tokenization and encryption. Ingested claims data is subsequently enriched and cleansed in the transform phase to ensure alignment with scheduled payment guidelines and formats. Then the downstream Transform may have to call other External Data APIs to get the data not available in the Source Master; that request is again synchronous as an additional step in the Extract Stage of Data Enrichment / Quality Check prior to creating the topic on which the Gatekeeper Event is published.

4.2. Transform Phase

The Transform Phase of the ETL operative Sequence comprises cleansing the extracted data within the Transform component of the pipeline. In an ideal scenario of real-time-Originated-source event-driven Integration, where the downstream topic accepts claims transactions applied with the claims events, Data Quality issues are addressed at the source of these events. From a Domain perspective, that includes Insurance policy, claim, product, and merchants that detect suspicious behaviour, are the entities accountable for calls to external Fraud Management Services and Geo Information Services sequentially for a complete verification of claims before commencing payout towards the claim. Sometimes the ability to enrich the Data during the Transform event is also valuable for downstream processing needs. In the same Extract example considered for Transform, the data can be enhanced for size in case of the vendor payment address and Zip code. When the pipeline works in an on-demand capacity for Gatekeeper roles within the claims process or for Customer Business partners, the extraction of all the data elements required for processing before the Gatekeeper request is mandatory. The domain Business logic can also be applied at the Transform relates to status update Deltas and Status field Transitions for Verification that leads to resolution of Claim. These Transformers push the Gatekeeper events in their respective topics and can be consumed in near real-time for User interface display by the Gatekeeper to see every required Flag and Request for Provision Certification. If the data consumed is the full details of Claim, Policy, Provision or other related information [8].

Equation 03: Transform efficiency

$$Transform / efficiency = Total\ Data\ Extracted / Valid\ Data \times 100\%. \tag{6}$$

Derivation

Count records that pass validation/cleansing as Valid, the rest as rejects.

$$TE(\%) = 100 \cdot V / E. \tag{7}$$

What I produced

Table: Extracted vs Valid for four sources (Claims Core, OCR, Vendor, CRM) + TE
Bar chart: TE% by source.

Source	Extracted rows	Valid rows	Transform efficiency %
Claims Core	500000	485000	97.0
OCR Docs	200000	150000	75.0
External Vendor	150000	130000	86.66666666666667
CRM	120000	115000	95.83333333333334

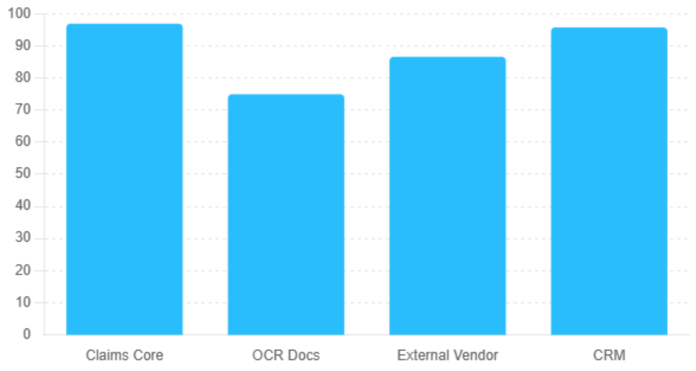

Figure 5. Transform Efficiency

5. Real-Time Data Processing

Real time data processing needs low data latency. Usually data size, the query speed, the network bandwidth, the price of storage affect the data latency. Each enterprise has different data latency requirements. Different systems store data in different ways. An efficient data pipeline minimizes the cost of data latency. The goal is to build an optimized data pipeline which can match thousands of data sources – each with different data latency. The target data is fed into such a data pipeline and it is delivered to low latency or high latency storage depending on the need of the target data. Technology is advanced and enterprises are moving towards real time data processing of their business data for better analytics and faster decision making. Apache Kafka is a good choice for real time data processing. Kafka is an open source streaming platform which can adequately replace traditional message bus technologies. Claims processing at the scale of a large insurer can be complex. Multiple heterogeneous platforms and systems, operating either synchronously or asynchronously, receive the claim request and process it. Most of these systems do not expose the claim data during or after the processing stage. Extracting the data from the journey in real time to support claims processing is a challenge and hence requires a cloud native ETL architecture and design considerations. Advanced transforms can enrich and perform complex aggregations as well as forecast trends. Transforming, enriching, and cleansing the data enables real-time KPI computation. Data quality concerns and scalability demand special care [9].

5.1. Importance in Claims Processing

Claims processing is at the core of insurer activities. A lag in claims processing tends to result in customer dissatisfaction and loss of business. The ability to process claims in real time offers a host of benefits such as reducing loss, gaining deeper insights, predicting frauds and enabling analytics on claims data in real time. The Extract-Transform-Load (ETL) pipeline plays an important role in business processes. An ETL pipeline is often implemented in the cloud in order to leverage the benefits of the cloud and is therefore called a cloud native ETL pipeline. The Extract-Transform (ET) phase of an ETL pipeline for real-time claims processing in largescale insurers is the focus here. The process of claims data in insurers is complex. Both the source and target system cater to a wide range of different systems, services and platforms. The data travels through different gateways and protocols during the extract phase and may require cleaning, validation, enrichment and augmentation in the transform phase before loading into the target system. The scalability of the ETL pipelines is pivotal during the exact phase to offer real-time claims processing support in insurers [10].

5.2. Technologies for Real-Time Processing

Real time processing presents insurtech with a fundamentally new challenge. While the execution of an underwriting request or a claim advice usually only takes a few seconds or minutes, the ingestion of real time data on a one second level or even higher can generate a very high volume of events which needs to buffered, processed, aggregated and enriched along the different lines of business. A real time event processing solution is especially relevant for property and casualty insurers who sell connected insurance products. The entire extract phase of an ETL can thus become a tool within the implementation of a real time data processing architecture. Extracting values from connected vehicles—with the additional risk information from real time sensors—can create new, additional underwriting and pricing opportunities for the motor line of business. Streaming information of car accidents taken from the connected vehicles can be equally important for real time claims processing and thereby help to be more predictive in the management of a claim.

Equation 04: Pipeline latency

$$Latency\,/\,pipeline = Bandwidth\,/\,Data\ Size + Processing\ time. \tag{8}$$

Derivation

Transfer time for payload of size S MB over B Mbps:

$$t_{net} = B8Sseconds. \tag{9}$$

Add compute time t_{proc} for parsing/transforms:

$$L(S,B) = B8S + t_{proc} \tag{10}$$

What I produced

Table: L for several S (50, 200, 800 MB) across common B tiers.
Line chart: L vs B (one curve per S).

Size MB	Bandwidth Mbps	Latency s
50	5	80.3
50	10	40.3
50	20	20.3
50	50	8.3
50	100	4.3
50	200	2.3
50	500	1.1
50	1000	0.7
200	5	320.3
200	10	160.3

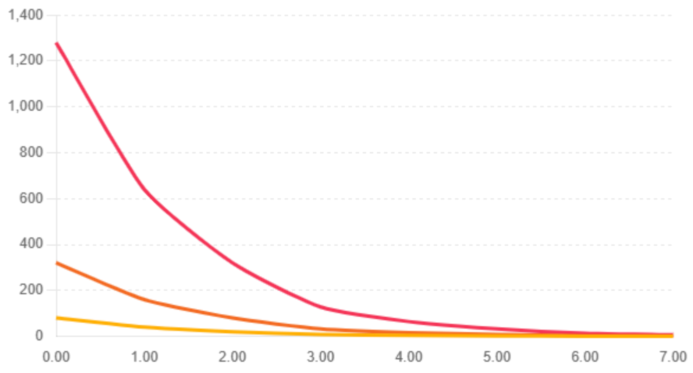

Figure 6. Latency vs Bandwidth

6. Challenges in Claims Processing

Large-scale insurers are challenged by data quality issues when dealing with claims data. Integrating. cleansing, validating. and mapping claims data from multiple external sources into the insurer's systems is a major pain area. Data may be duplicated, incomplete, erroneous, or in multiple formats, requiring careful transformation. Unreadable characters and other errors further complicate the transform stage. The inability to adequately address these compromises the accuracy of downstream metrics such as case reserves, claim liabilities, and loss ratios. The extract–transform–load approach is standard practice for all types of insurers. A cloud native ETL architecture scales live data streams, facilitating near real-time processing for claim counts, fraud detection, and overall claims analysis. The extract and transform phases are particularly critical. Real-time data processing is best performed through streaming platforms or service bus components. Designed cloud native, the extract and transform phases can rapidly ingest and cleanse data from multiple parallel sources [11].

6.1. Data Quality Issues

Poor data quality presents a well-known challenge for insurers, often causing the Transform phase in an ETL pipeline to consume a disproportionate amount of time and effort. Many insurers still rely heavily on manual key-entry of claim information combined with inconsistent data capture during the claim registration process, leading to potential inaccuracies at source. Additionally, there are frequent requirements to normalize data from a variety of disparate and often legacy systems, which can further complicate data integrity. All these factors not only increase the risk of inaccurate data being utilized for claim analysis but also make it difficult and costly to enhance processes later on [12]. Scalability is another major concern for insurers, who strive to rapidly adapt to growth, seasonality, and market volatility. The Extract and Transform phases of an ETL pipeline are particularly vulnerable to scalability constraints. If these phases cannot immediately process the wealth of new data being generated, insurers are forced to utilize stale data during claims processing decisions. This latency can greatly diminish an insurer's ability to effectively combat fraud, as analysis fundamentally depends on timely access to high-quality data.

6.2. Scalability Concerns

Real-time processing of claim data typically experiences volumes that require high scalability and hence low latency. Even a small increase in processing time can lead to missing a business SLA for real-time processing of claims. Scaling ETL pipelines is not a simple task. The ETL pipelines are typically implemented with steps using Apache Spark or any other processing framework within the same technology stack. If there is a requirement for scaling the pipelines, there is a need to ensure that the steps are stateless and independent. For instance, it is impossible to scale a step that is dependent on another step in a data processing job. For scaling transforms, one should split the logic of the transformations into independent containers so that they can be scaled independently as needed. However, this requires significant code changes. Integrating new sources or destinations also becomes difficult with such tightly coupled steps in a job pipeline. In large insurers, data resides in multiple heterogeneous data sources. It is a challenging task to continuously scale the jobs with the failure of a downstream system and also to incorporate new sources without jeopardizing the existing pipelines. Other approaches, like splitting the Extract and Transform phases into separate phases with the Transform phase processing data from a queue or topic, could be very useful since the scaling concerns of the Extract and the Transform phases can be addressed independently [13].

Equation 05: ETL scalability

$$Scalability\ ETL = Peak\ Claims\ Processed\ Claims\ \times 100\% \tag{11}$$

Derivation

For a period (e.g. day) with a peak incoming P and processed X:

$$Scalability(\%) = 100 \cdot PX. \tag{12}$$

What I produced

Table: Daily ΔP, ΔX, Scalability/Weekly
Bar chart: Scalability% per day.

Day	Peak Claims	Processed Claims	Scalability %
D1	80000	76000	95.0
D2	120000	110000	91.66666666666666
D3	90000	87000	96.66666666666667
D4	150000	140000	93.33333333333333
D5	160000	158000	98.75
D6	140000	137000	97.85714285714285
D7	100000	99000	99.0

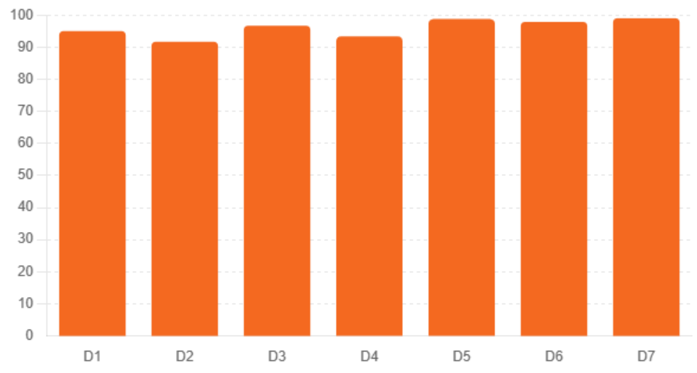

Figure 7. Weekly Scalability

7. Designing ETL Pipelines

ETL pipelines, where Extract and Transform operations within Extract, Transform, Load are implemented, exhibit the modularity and interoperability crucial for real-time bulk data processing in large enterprise insurers. The pipeline's scalable, highly reliable, and elastic design responds dynamically to fluctuating claim volumes and event frequencies across claims assessment, adjustment, and settlement. This scalability ensures adequate resource provisioning per ETL job allocation, preventing termination from over or under provisioning. The real-time capability enables generalized ETL jobs to proactively process alarm detection or prediction events during claim lifecycle phases via streaming platforms. Input data channels for the pipeline comprise company internal sources—such as lines of business platforms, policy administration, billing, and claims systems—and external sources, including law enforcement or government databases, social media, weather websites, and IoT applications like sensors, cameras, and drones. In an insurance context, the exact phase involves capturing claim records from lines of business billing and claims systems in real-time, modulated by InsurTech's business logic. During transformation, cleansing, formatting, ICD-code application, inferring additional information, and enrichment from third-party sources—such as weather, social media, or traffic data—refine the records before database upload. Proven sensitive to data quality issues, both extract and transform phases serve as choke points highlighted in non-scalability analyses [14].

7.1. Pipeline Architecture

Claims processing through a cloud native ETL pipeline can bring these important benefits: Cost avoidance for peak capacity. Capacity can be dynamically scaled to meet demand, avoiding the cost of idle capacity that's incurred when an insurer fails to meet peak demand. • Higher customer satisfaction. Real-time or near real-time processing enables insurers to deliver better customer service and respond faster to fraud. • Blazing fast claims approval turnaround. Real-time processing significantly accelerates the processing of simple claims. • Improved fraud monitoring. Real-time data processing speeds the early detection of fraud. • Improved data quality. Realtime transformation and loading can detect and resolve data quality issues early; • Greater availability and resilience. A serverless DataHub based system eliminates single points of failure; • Faster throughput and business agility. Streaming technologies increase processing concurrency and make it possible to respond rapidly to changing business requirements. The ETL process can be divided into three distinct phases: Extract, Transform, and Load. Data extraction captures and collects data from external business applications and internal databases. After data extraction, it goes through the transform phase where cleansing, filtering, formatting, matching from internal and external sources, consolidating, and aggregating happens. Real-time or near real-time data flow support is required between the extract and transform phases to enable faster data ingestion and processing.

7.2. Data Sources Integration

The Exploit phase is concerned with acquiring data. Typically encountered in claims associated with large-scale insurers, the data sources consist of a multitude of unwieldy data feeds. In the Extract phase of the ETL process, the objective is to initiate the flow of data entries within the pipeline. It is imperative that information be captured by the Travel Data Acquisition System and made available, in a real-time fashion, for subsequent analysis, triggering appropriate alerting mechanisms. Several data repositories—containing customer information, policyholder details, and insurance cover particulars—are hierarchically arranged to culminate in a consolidated data repository. This extracted data is subsequently subjected to the Transform phase, wherein it is cleansed and formatted in accordance with the specifications of the STA reporting procedure. Data quality issues, inherent in such a setup, exert a negative influence on processing speed, necessitating a re-engineering of the ETL pipeline. To address the constraints of latency in today's claims processing environment, the event-processing capabilities offered by Alpakka Kafka have been harnessed, facilitating real-time data processing.

Equation 06: ETL quality (stage yield)

$$Quality\,/\,ETL = Raw\ Data\ Cleansed\ Data\ \times 100\%. \tag{13}$$

Derivation

Track record counts by stage: Raw →→ Post-Cleansing →→ Post-Enrichment.

$$Percent\ of\ Rawat\ Stage\ i = 100 \cdot N_{raw} N_i. \tag{14}$$

What I produced

Table: Row counts per stage + % of raw retained.
Bar chart: % of raw across stages.

Stage	Rows	Quality percent of raw
Raw Intake	600000	100.0
Post-Cleansing	570000	95.0
Post-Enrichment	560000	93.33333333333333

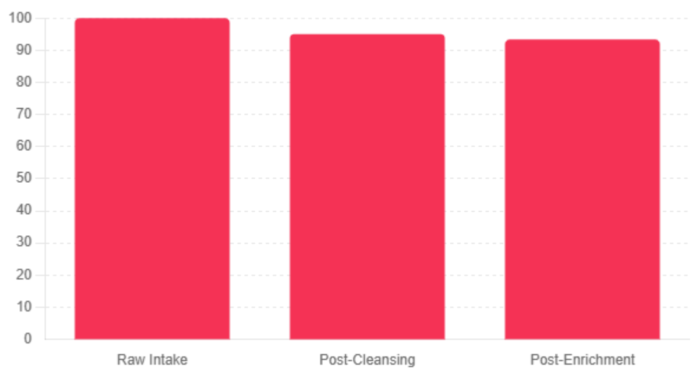

Figure 8. Data Quality Progression

8. Conclusion

Cloud native architecture enables ETL pipelines to be designed for scalability and two key phases: extract and transform. In the exact phase, data is moved from one system to another. During the transform phase, data is cleaned, reshaped, tagged, or enhanced in preparation for loading into a target system or database. Real-time data processing is increasingly valuable in data-intensive activities such as claims processing. Real-time streaming platforms like Apache Kafka can be employed to connect data streams, including events, business processes, or metadata, thereby enabling near-real-time updates to one or more target systems. Claims processing operations can encounter a range of issues—data quality; designing for multiple, diverse data sources; and scalability to meet business demands—that can be resolved by architecting an ETL pipeline around a cloud native design.

8.1. Emerging Trends

Cloud native ETL pipelines offer significant benefits for real-time claims processing in insurance. These include scalability, availability and resilience, automation and orchestration, quick and easy setup of new environments, greater co-location and decentralization of teams, hybrid and multi-cloud capabilities, reduced dependency on shared data center resources, able to penetrate new markets through edge computing, and process optimization through artificial intelligence and data analytics. ETL is a process that involves the extraction of data from a data source, transforming the data—such as data cleansing, data harmonization, or data enrichment—and loading it into the end application. Extract and transform are the focus when designing a large scale, real-time, cloud native ETL pipeline. When large insurers are involved, there are typically many applications or systems of records that need to be integrated into the claims process, such as core policy administration, claims management, document management, OCR, business rules, data hub, business intelligence, or visualization platforms. Real-time data processing has a specific set of variables and requirements. Streaming and streaming platforms provide a set of patterns and primitives to help enable real-time data processing, such as data ingestion, data transformation, data analytics, data curation, data quality, and event-driven application integration. The challenge with claims processing is that data quality suffers from the same problems as batch-oriented processes; dirty data—data that is incomplete, inconsistent, or inaccurate—slows down the claims business process. Furthermore, real-time claims processing needs cloud native horizontal scalability, availability, and resilience [15].

References

[1] Hannousse, A., & Yahiouche, S. (2020). Securing microservices and microservice architectures: A systematic mapping study. *arXiv preprint arXiv:2003.07262*.

[2] Haase, C., Ro¨seler, T., & Seidel, M. (2022). *METL: a modern ETL pipeline with a dynamic mapping matrix*. arXiv. https://arxiv.org/abs/2203.10289

[3] Lahari Pandiri. (2022). Risk Assessment In Homeowners And Renters Insurance Using Explainable AI Models. Migration Letters, 19(S2), 1945–1967. Retrieved from https://migrationletters.com/index.php/ml/article/view/11939

[4] Botlagunta Preethish Nandan. (2021). Enhancing Chip Performance Through Predictive Analytics and Automated Design Verification. Journal of International Crisis and Risk Communication Research, 265–285. https://doi.org/10.63278/jicrcr.vi.3040

[5] Wang, J., Li, T., Song, H., Yang, X., Zhou, W., Li, F., & Chen, X. (2023). PolarDB-IMCI: A cloud-native HTAP database system at Alibaba. *arXiv preprint arXiv:2305.08468*.

[6] Krishna Reddy Koppolu, H., Recharla, M., & Chakilam, C. (2022). Revolutionizing Patient Care with AI and Cloud Computing: A Framework for Scalable and Predictive Healthcare Solutions. International Journal of Science and Research (IJSR), 1457–1472. https://doi.org/10.21275/ms2212142204

[7] Jindal, A., Quiane-Ruiz, J.-A., & Madden, S. (2017). INGESTBASE: A declarative data ingestion system. *Proceedings of the VLDB Endowment*, 10(12), 1933-1944.

[8] 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

[9] Garc´ıa-Lo´pez, P., Arjona, A., Sampe, J., Slominski, A., & Villard, L. (2020). Triggerflow: Trigger-based orchestration of serverless workflows. *arXiv preprint arXiv:2006.08654*.
[10] Kummari, D. N. (2022). Fiscal Policy Simulation Using AI And Big Data: Improving Government Financial Planning. Kurdish Studies. https://doi.org/10.53555/ks.v10i2.3855
[11] Burugulla, J. K. R., & Inala, R. (2022). The Future of Payments: A Comprehensive Review of AI, ML, and Cloud Technologies in Finance. Kurdish Studies. https://doi.org/10.53555/ks.v10i2.3870
[12] 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
[13] Sheelam, G. K. (2022). Reconfigurable Semiconductor Architectures For AI-Enhanced Wireless Communication Networks. Kurdish Studies. https://doi.org/10.53555/ks.v10i2.3867
[14] Levcovitz, A., Terra, R., & Valente, M. T. (2016). Towards a technique for extracting microservices from monolithic enterprise systems. *arXiv preprint arXiv:1605.03175*.
[15] Neupane,R. L., Bonnah, E., Bhusal, B., Neupane, K., Hoque, K. A., & Calyam, P. (2024). Formal Verification for Blockchain-based Insurance Claims Processing. *arXiv preprint arXiv:2402.13169*.