International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

Scalable Data Engineering Architectures for Real-Time Financial Transaction Auditing

Anumandla Mukesh

Independent Researcher, India

ABSTRACT: Even modest-sized organisations face intense regulatory pressure to provide instant assurance that high volumes of financial transactions satisfy their risk and compliance controls. Yet current approaches adopt a slow batch-cycle mentality, lacking observable data provenance, whose absence challenges trustworthiness, experiment outreach and risk-based prioritisation. This paper investigates scalable data engineering architectures that deliver audit assurance on a continuous basis in a manner that meets the needs of businesspeople and machine learning practitioners alike. Architecture-level choices, trade-offs and evaluation metrics guide a selection of the numerous available solutions in the Process and Production phases of the Data Engineering Life Cycle.

Data Observability refers to the dimension that focuses on quantifying and reporting the quality, reliability and trustworthiness of data within a platform. With increasing volumes of data being processed for insight and analysis, understanding how data quality is met and maintained is vital. Data observability enables organisations to systematically quantify these parameters, visualising baselines over time and providing alerting and reporting to allow ongoing adjustment to business-critical processes and systems. Data Provenance fills the evidential gap underneath data observability. Data provenance provides a record of the origin and history of each item of data throughout its life cycle, typically including details from how original data was captured, processing steps applied and how it was published to a risk or compliance assurance system. The absence of data provenance in a high-volume data environment reduces the trustworthiness of those systems. In risk-based approaches to data assurance this often leads auditors to perform extensive outward experiments, logging few internal checks; resourcing operations for seen risks instead of unseen assurances.

KEYWORDS: Observation, provenance, transactions, data ingestion, change data capture, streaming platforms, message brokers, data warehouse, data lakehouse, transformation pipelines, schema evolution, anomaly detection, risk scoring, data governance, access control, data lineage, audit trail.

I. INTRODUCTION

Security and compliance considerations impose stringent requirements on financial data management, necessitating transparency, provenance, and log records for detection and investigation of abnormal patterns alongside functional workloads. Traditionally, auditing is conducted by periodically making batch copies of core business data. An architecture formalism for real-time audit of database transactions is presented for web-scale systems using scalable data engineering principles. Section 1 addresses the motivation and proposed architecture goals. Section 2 considers the principles that guide the selected architecture choices. Sections 3, 4, 5, and 6 analyze the solutions for the four layers of the architecture.

Financial databases, in addition to supporting business transactions, implicitly support audit and compliance functions. Audit functions require transparency into business transaction data and logging of all corresponding changes. Audit release of production data for testing must be governed under strict rules. In critical applications, such as bank fraud detection, it may be necessary to detect altered states or transaction sequences. Risk-scoring fraud detection applications operating on the financial transaction data may also require access to the data before anomaly detection is complete. As a result, audit requirements impose constraints that strongly influence the design of the underlying database systems—and therefore the database data engineer.

1.1. Motivation and scope

The recent digitization of banking services and related data flows, combined with an increasingly hostile cyber-threat landscape, demands the implementation of strong automated controls to ensure compliance with regulatory standards. Emerging regulations related to transaction data security and control in the financial domain require the consistent, real-time auditing of all financial transaction data. Traditional financial transaction auditing architectures typically use

International Journal of Science, Research and Technology (IJSRAT)
|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

periodic batch refresh of data in dedicated data warehouses that are consumed by renewed reports or dashboards. Real-time transaction auditing, which requires instantaneous availability of risk analytics and scoring, is a naturally appealing alternative. Its architecture provides near real-time data availability for fines, anomalies, rule violations, and risk scoring. Such real-time architectures also allow for enhanced monitoring of transaction data by providing dedicated analytics that exploit data consistency and integrity.

Possibly the most threatening risk to the above analytical pipelines relates to the completeness of data ingested into the architecture. This question is particularly critical during batch refresh modes, as historical data completeness is enforced only for the duration of each of the insert-copy-refresh cycles. Other properties, such as being observably auditable, are usually overlooked in such analytics-refresh learning scenarios. Streaming architectures with continuous processing address these aspects very well, although not necessarily without sacrifice; guarantees that remain in effect in the event of node failure; care must be taken when allocating resources for testing and enforcement, as these resources directly affect the accuracy and availability of the test controls.

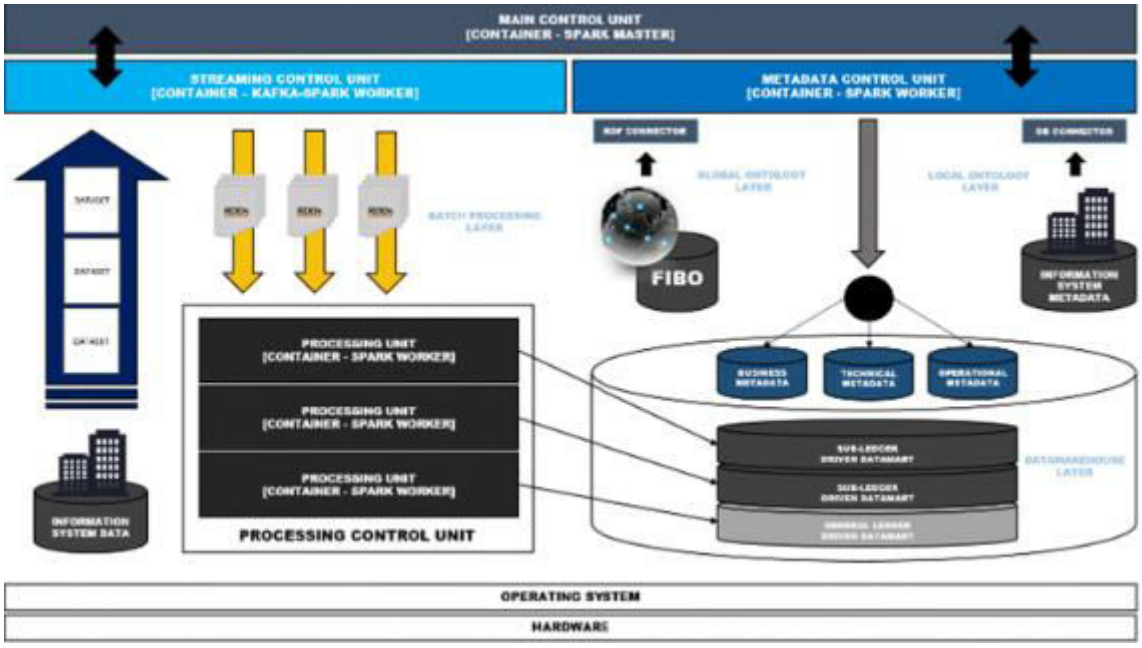


Fig 1: An adaptive and real-time based architecture for financial data integration

1.2. Stakeholders and requirements

A formal overview identifies stakeholders and requirements for scalable real-time auditing of financial transaction systems. On one hand, compliance and operational risk teams demand low-latency data and alerts for breaches of set tolerances; on the other, proximal developers clearly recognize the challenge of successfully monitoring for such breaches and propelling alerts into lower-risk ai-software for hardening.

The risk hath o'erleapt! – William Shakespeare, Macbeth, Act III, Scene IV

Equation 1: Decompose latency by architectural stage (debuggability/observability)

Add and subtract intermediate timestamps:

$$L_{\text{e2e}}^{(i)} = (t_{\text{ingest}}^{(i)} - t_{\text{commit}}^{(i)}) + (t_{\text{lake}}^{(i)} - t_{\text{ingest}}^{(i)}) + (t_{\text{score}}^{(i)} - t_{\text{lake}}^{(i)})$$

Define:

- $L_{\text{cdc}}^{(i)} = t_{\text{ingest}}^{(i)} - t_{\text{commit}}^{(i)}$
- $L_{\text{stream}\to\text{store}}^{(i)} = t_{\text{lake}}^{(i)} - t_{\text{ingest}}^{(i)}$
- $L_{\text{compute}}^{(i)} = t_{\text{score}}^{(i)} - t_{\text{lake}}^{(i)}$

Then:

$$L_{\text{e2e}}^{(i)} = L_{\text{cdc}}^{(i)} + L_{\text{stream}\to\text{store}}^{(i)} + L_{\text{compute}}^{(i)}$$

International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

II. ARCHITECTURAL PRINCIPLES FOR REAL-TIME AUDITING

Real-time financial transaction auditing systems must provide a significant capacity for observability and transaction provenance to differentiate them from other data processing and analytics systems. The requirements for consistency, availability, and partition tolerance (CAP theorem) for such systems are also examined.

The architecture must allow stakeholders to observe all aspects of auditing transformation pipelines without significant latency. Multiple dimensions should be observable, allowing content, accuracy, and process aspects to be scrutinized, not just performance. Observability beyond basic metrics is also key, requiring the integration of statistics, ML model monitoring, and data quality tools into a common view.

The architecture should also accommodate detailed transaction provenance to provide clarity on the source, inversion point, and risk attached to any alarming transaction falling outside pre-defined thresholds. Evolutionary processes for the storage and enrichment of provenance data not only satisfy governance requirements but also enable greater contextualization of the transaction history.

Auditing real-time transactions is inherently difficult, requiring a careful balance of the CAP tradeoff. Typically, fraud detection models prioritize high availability and partition tolerance at the expense of consistency by allowing transactions to flow through the system in both directions (fraud detection and legitimate detection) to manage errors proportionately.

Equation 2: SLO check (as suggested by SLI/SLO validation in the paper)
Scalable Data Engineering Archi…
.
Define an indicator:

$$\mathbf{1}\{L_{\text{e2e}}^{(i)} \le 10 \text{ min}\} = \begin{cases} 1 & \text{if } L_{\text{e2e}}^{(i)} \le 10 \\ 0 & \text{otherwise} \end{cases}$$

Over a window W(e.g., 1 hour, 1 day), the **latency SLO compliance** is:

$$\text{SLO}_{\text{lat}}(W) = \frac{1}{|\,W\,|} \sum_{i \in W} \mathbf{1}\{\, L_{\text{e2e}}^{(i)} \le 10$$

2.1. Observability and provenance

Financial transaction auditing encompasses risk detection and fraud detection components. Real-time infra-structure for risk detection ensures that both positive and negative signals feed back into operational workflows and can be acted upon quickly. Online transaction processing (OLTP) systems typically aggregate transactions in near real time into analytical stores, allowing detection of risk signals using analytical models. Observability and provenance of data allow business analysts and developers to reason about the quality of both historical and real-time models. Audit log analysis of the infrastructure ensures the integrity of both these processes.

Without adequate observability, it is difficult to diagnose why a particular risk signal fired or understand the relevance of the historical models used to calculate such a signal. At a minimum, a Layered Data Mart approach should be preferred, in the sense that the variable definitions of each Ready-to-Ship variable become the source-of-truth definitions for that variable. Automated validation, similar to the SLI/SLO (Service-Level Indicators/Service-Level Objectives) concept in site reliability engineering, should also be employed when producing risk signals. Signals computed within the development environment should have their predictions compared against the predictions of deployed models to validate the applicability of the model in production.

International Journal of Science, Research and Technology (IJSRAT)
|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

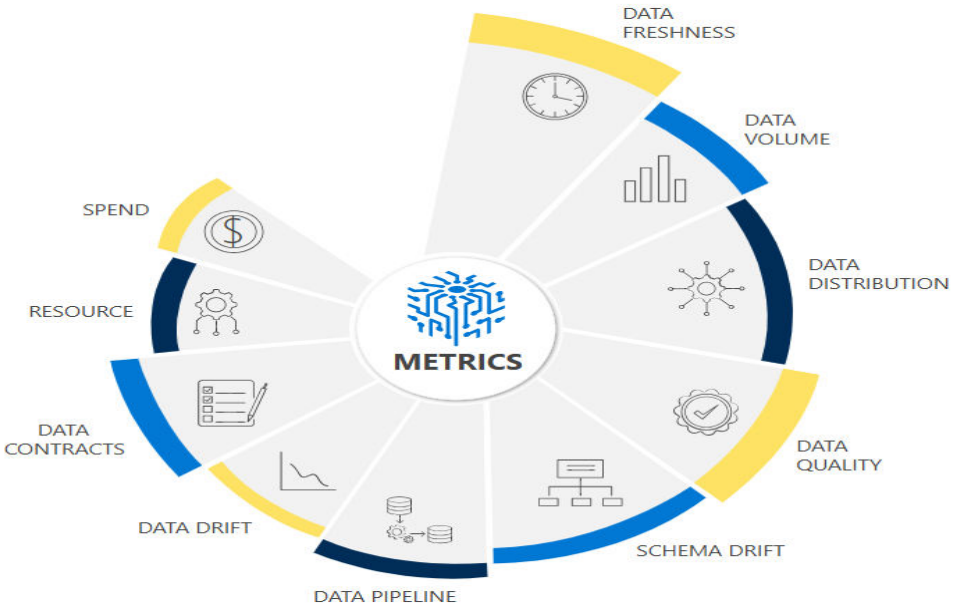


Fig 2: Data Observability Metrics

2.2. Consistency, availability, and partition tolerance considerations

Possessing conflicting requirements for high availability and wide-area distribution, retail bank transaction auditing architecture necessitates flexibility and the fine-tuning of trade-offs. With a focus on the auditing function, the architecture design embraces eventual consistency while limiting the delay for most applications. Focusing primarily on retail-banking transaction auditing, consistency, availability, and partition-tolerance considerations are shaped by the fine-tuning of trade-offs.

Banks possess specialized systems supporting customer accounts, contact databases, and fraud analysis, and lower-latency volumes of customer transactions such as those from credit cards, ATM and point-of-sale transactions, and online banking. These applications demand high availability and low query latency, forcing banks to rely on replication and geographical distribution even across a single data centre. Such conflict leads to the classification of banking functions into three categories: those where partition tolerance is paramount and therefore consistency can be sacrificed; those requiring consistency, including some, but not all, transaction-processing systems; and those needing partition tolerance without strict requirements for consistency. The auditing subsystem sits squarely in the first class, with retail transactions that must be available for fraud analysis without requiring strict consistency.

The design enables transactional auditing to achieve near real-time update-to-query latency without sacrificing integrity. Change-data-capture from the transactional databases of multiple credit-card and payment-processing partners uses reliable messaging without strict semantics. The distribution of these messages is regionally optimized while remaining available in a cross-region response. Anomaly detection, using compute-once-store-more, supports a broad range of users with various business-levels of transaction expertise and risk tolerance.

III. DATA INGESTION AND STREAM PROCESSING

Financial transaction data is maintained by multiple heterogeneous systems, including account instruments, cards, restrictions, transfers, payment systems, and identity management. These systems do not guarantee that the data is always consistent and fail-safe, so change data capture mechanisms are usually put in place to extract modifications and replicate them into data lakes. To enhance consistency of data and accelerate its consumption, these changes can be published to a streaming platform and made available together with the information produced by transactional systems.

Change data capture captures and streams changes from the source database to a streaming platform. The changes for base tables are usually encoded using a PubSub mechanism, such as Debezium, where every change is inserted into a specific topic in the streaming platform. Change events are then consumed and streamed into the data lake. Most of the stream processing layer is implemented using a scalable Kafka Streams solution, where every change is routed to a

International Journal of Science, Research and Technology (IJSRAT)
|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

query task based on the destination table. Data is then transformed into a respective Snowflake schema and written into the data lake. The streaming elements of the architecture manage the full lifecycle of data from the source system to the data lake while enhancing data quality.

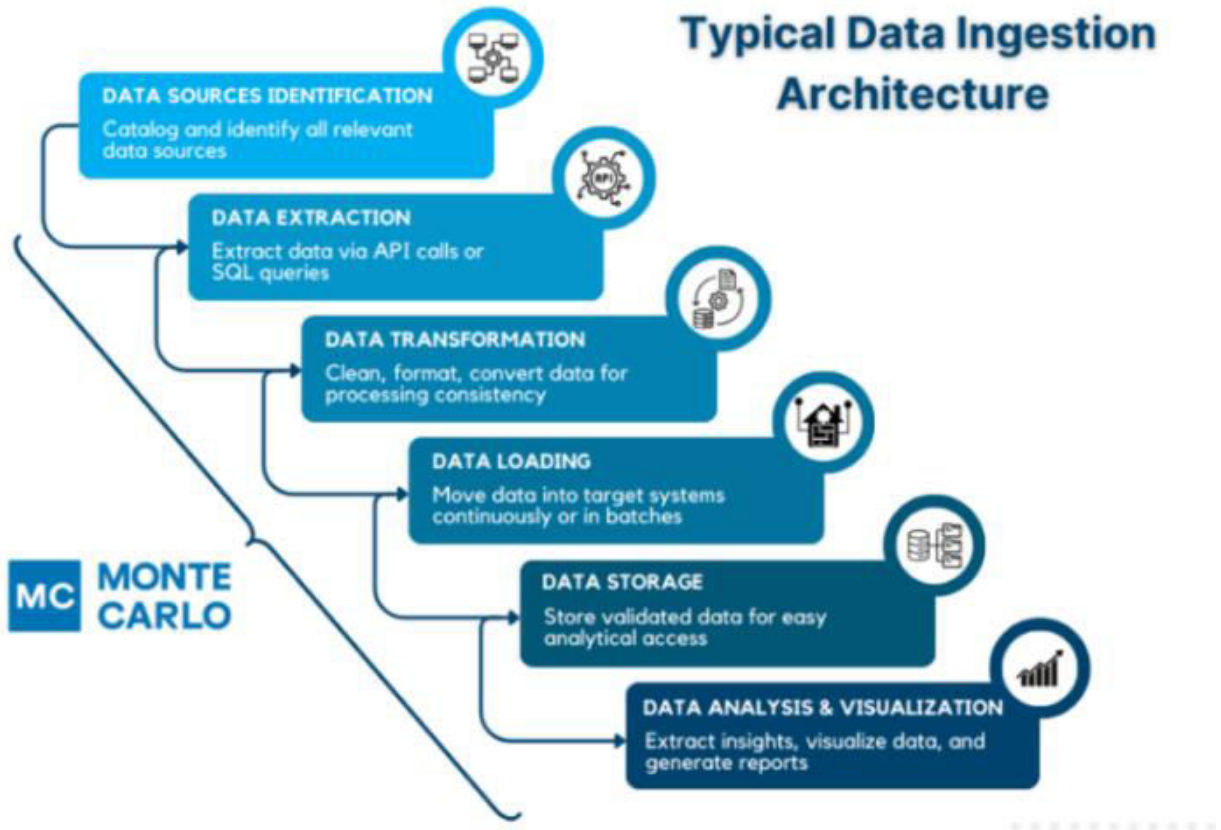


Fig 3: Data Ingestion Architecture

3.1. Source systems and change data capture

Commonly used relational and non-relational databases often serve as source systems for transaction data ingestion into an auditing architecture's processing and analytics layer. A well-supported change data capture (CDC) mechanism enables the automatic detection and exporting of modifications made to these systems. For external sources, such as external APIs, file systems, and NoSQL document stores, changes are scraped at set intervals, typically using pull architectures. In such cases, true change event semantics cannot be guaranteed, but approaches that optimize load on the source can be used. Data redundancy in the CDC products can be handled later in the architecture through de-duplication mechanisms or by leveraging the data freshness properties of the subsequent systems.

Ideally, data from all source systems is ingested in real time or with minimal latency. This requirement typically means that data moves around as a continuously growing stream of near-real-time data instead of being processed or transferred in batch mode (for instance, using daily night jobs). Data ingestion in real time or with low latency can often complete tasks at a lower cost due to reduced infrastructure and human-operating expenses. Instead of replicating the data continuously from operational systems into a reporting system, which can be complicated to maintain, establish a logical archive or audit store in combination with a replication mechanism and an additional non-DC-confined system that detects change events from the sources and applies them to the archive or audit store. The audit store contains the same information for these two systems and can provide a single record of transactions across all systems, particularly for security-sensitive transactions.

Equation 3: Batch/window completeness

In a time bucket (or logical batch) W:

- $N_{\text{produced}}(W)$: number of events the source claims were produced (from source logs / CDC offset ranges)

International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

- $N_{\text{ingested}}(W)$: number actually observed in the streaming/audit pipeline

Define completeness:

$$C(W) = \frac{N_{\text{ingested}}(W)}{N_{\text{produced}}(W)}$$

$$M(W) = N_{\text{produced}}(W) - N_{\text{ingested}}(W)$$

Pick a threshold (example: 99.9%):

$$\mathbf{1}\{C(W) \geq 0.999\}$$

3.2. Streaming platforms and message brokers

For processing streams of records, a messaging broker or streaming platform is required. Both allow client apps and enterprise architecture elements to communicate according to a publish-and-subscribe model. A messaging broker brokers the messages while a streaming platform combines the functions of a message broker with log compaction and durable data storage. The members of the most widely adopted open-source streaming ecosystem—Apache Kafka, Kafka Connect, and Kafka Streams—are implemented in production systems run by enterprises of every size. They have become a de facto standard. Other broker platforms such as Confluent or AWS MSK offer fully managed cloud-based options. In contrast, software programs such as Apache Pulsar, Redpanda or Redis Streams support only some of the Kafka capabilities. Key features of streaming platforms and message brokers include high throughput, low latency, multi-tenancy, fault tolerance, message retention capabilities, decoupled producers and consumers, message delivery guarantees, and exactly-once semantics.

For producing or consuming a data stream to trigger business processes, a message broker implementation such as Apache ActiveMQ is a good choice. Different scenarios call for specific platforms. For example, ActiveMQ, RabbitMQ, or AWS SNS are often used in conjunction with Amazon SQS for an asynchronously decoupled architecture. Apache Pulsar might be a sound choice for domain-specific content-based routing or message-driven processing. Airflow or Apache NiFi could be used for data-in-transit transfer. They would not be a good choice for triggering business processes. In other scenarios, either Kafka or AWS Kinesis can be used for decoupled components. They are often the only option if massive bandwidth is needed. In one approach, all the internal information services locate data in Kafka and access it via SQL for consumption inside or outside the enterprise.

IV. DATA STORAGE AND LAKEHOUSE SOLUTIONS

In data management, auditability often implies immutability. Once data ingestion systems have observed a state transition in a source system, subsequent alterations must be reflected in downstream services. This imperative suggests interaction patterns consistent with an append-only log, where information about past states is still accessible. Real-time auditing systems violate this fundamental principle when presenting user-facing views in a highly aggregated form: decision-support tools serving true production use cases should mirror patterns of use for source data. Keeping these data sources regularly refreshed through batch exploration systems with reasonably low latencies may be sufficient to ensure that such delta views reflect reality more accurately than user recollections. Even so, system architectures should allow for future exploration of evolving past states, allowing forensic users to follow potentially important histories.

Prominent across the many collected definitions of the term lakehouse is the use of low-cost, scale-out storage to house full fidelity source data including files and objects replicated from external systems via change data capture protocols. An architect will thus tend toward lakehouse implementations of the move and store building block pattern, with suited systems of ingestion and analytics capable of handling footprints that are highly sparse and unevenly distributed through time. A sensible low-latency view over lakehouse data requires a systems-centered rather than data-centered model: use of a database designed to optimize its internal structures for real-time transactions and scoring—and regularly refreshed by a batch pipeline serving the lakehouse at minimal overhead—will be more than fulfilling.

International Journal of Science, Research and Technology (IJSRAT)
|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

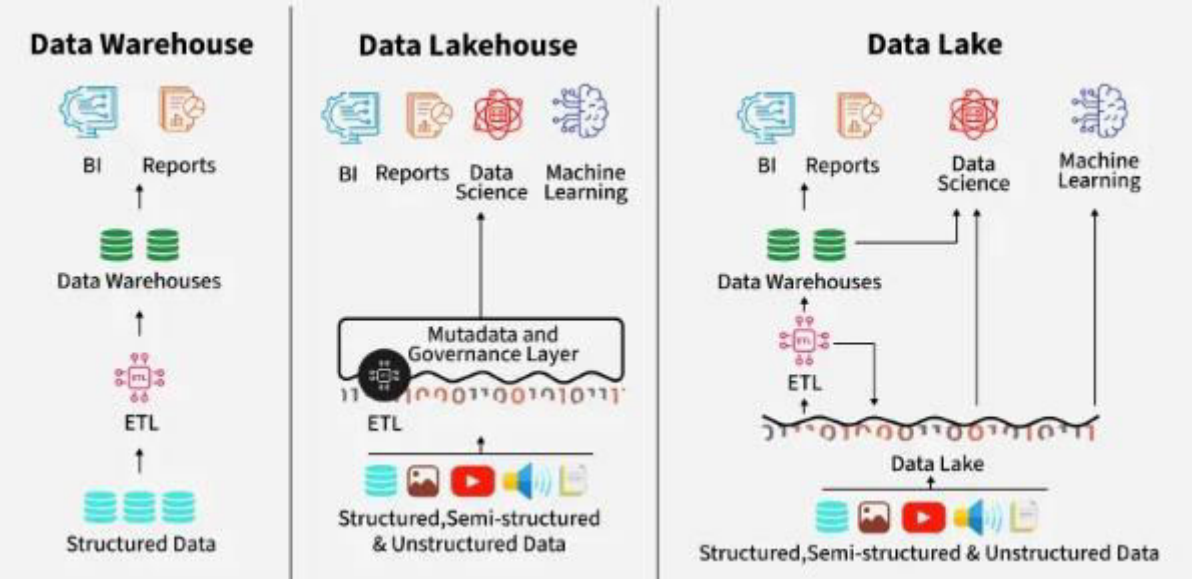


Fig 4: Data Storage and Lakehouse Solutions

4.1. Immutable immutable storage patterns

Real-time environments are subject to heavy load variations and burst transactional activity. As a consequence, using the lake-house methodology is an obvious option. Data are stored in object storage based on a file-system (primary data) with an immutable format. Query engines interact with the data in a lazy, demand-driven way by building a virtual view and executing it when queried. However, the distribution of a lake house on a file format normally induces an overhead. In addition, the data must be up to date, and therefore one also requires a near-real-time update of the views. An immediate way to minimize overhead latency that does not require complex architectures is to create a materialized view for the last second or several seconds, with the risk of being read erroneously in a busy system.

4.2. Real-time views vs. batch refresh

For near-real-time reporting use cases, the lakehouse supports periodic refreshes of real-time views populated with mutable data. During refresh, information about the modifications is written to a staging area while the main immutable zone is kept in read-only mode. The lakehouse ecosystem detects that a new set of modifications is available, triggers resolution, and publishes a new version of the real-time view populated by the refresh. Because modifications originating from the main data source systems are often written outside business hours, only transactions that occurred during that period are staged.

If the need arises to manage the cost of read queries on the real-time views, they can reside in a different data partition whenever there is demand to process the associated DataFrame. The pipeline responsible for staging the modifications includes a high-level parameter to control the partitioning of these views, moving them in or out based on elasticity considerations.

Eventually consistent atomicity guarantees are often acceptable when working with storage systems that manage data reliability and consistency in a decentralized, ephemeral, probabilistic way. During the usual course of operation, assurances of full consistency for finite sets of transactions can rarely be achieved, making it difficult to release application partitions to the rest of the system without impacting the final outcome.

Equation 4: Duplicate/replay handling (common with CDC + "reliable messaging without strict semantics")

Let:

- $N_{\text{total}}(W)$: total received events
- $N_{\text{unique}}(W)$: unique events after dedup (by event_id or (pk,lsn))

Duplicate rate:

$$D(W) = \frac{N_{\text{total}}(W) - N_{\text{unique}}(W)}{N_{\text{total}}(W)}$$

International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

V. PROCESSING AND ANALYTICS LAYER

The productionization of any analytical model emphasizes the need for a monitoring and alerting layer that surfaces models at risk. Pipelines enabling positive and negative risk scores across the financial portfolio can be a solid foundation for a risk management function. Data-driven decision makers require the latest possible information to minimize exposure and the servers providing this information should be enabled such that an SLA of <10 minutes can be achieved.

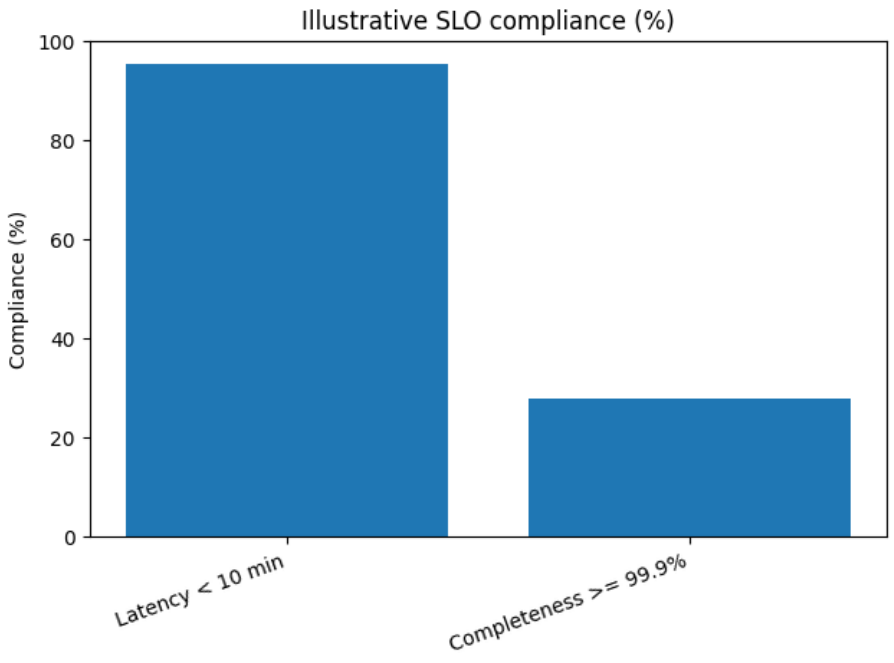
Many internal business partners want to quickly validate the accuracy of data in a new system prior to cutover, and static reports against a complete-depth table could satisfy this. Positive exposure to credit risk of trading counterparties is an area of intense scrutiny due to regulation, and revenue recognition process areas are constantly under the microscope. A product built for the data audit and anomaly detection space could be used across businesses to identify subtle problems and document their resolution.

Fast-moving financial products such as FX and CMBS harvest lot detail from booking databases. While this information originates in transactional systems, their architectures are based on the premise that assets have a quick lifecycle and that a data lake ETL enabler supporting a near real-time risk or P&L position can have batch refresh. The introduction of risk scoring and anomaly detection would push demand for these products into the months at minimum business calendar cycle time.

5.1. Transformation pipelines and schema evolution

Data transformations are often implemented in streaming pipelines that make necessary changes to the data for downstream systems as soon as possible. Monitoring capabilities are integrated into the pipelines to enable fault localization and alerting in case of issues. These monitoring capabilities generate alerts when processing latencies exceed expectations, when schema changes are detected in raw data and not handled correctly, when anomaly detection models detect data drift, or when other rules indicate an emerging problem.

Streaming pipelines easily deploy models from the analytics layer to append-model classes such as scoring and classification. Refreshing the models on a batch schedule would introduce higher latencies and risks since the conditions could change and require quick retraining. Anomaly detection models can be refreshed in a semi-automated manner by constantly evaluating incoming training data and generating a retraining alert when sufficient data for a new model is available. Data-schema evolution is supported either by using schema-on-read patterns in downstream systems or through auto-evolution configurations in the transformation layer. The pattern flows when these aspects are taught from their inception are critical to avoiding later issues.



International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

5.2. Anomaly detection and risk scoring

A variety of techniques can be employed to detect anomalies and assign risk scores. Each detection/scoring implementation should be refined based on objective model validation and an explicit business understanding of false positives/negatives and their sensitivity across different implementation areas. Risk scores assigned by models that use different underlying techniques (statistical tests, supervised machine learning, unsupervised learning, etc.) and/or take different perspectives can often be combined. For example, the scores from a model detecting outliers in refunds can be aggregated into a single score for each individual making a refund. Similarly, separate scores for different aspects of fraud detection can be computed, such as those for suggesting suspicious transactions, noting suspicious refund or transfer requests, and identifying transaction partners with a long history of being blocked for fraud.

Regardless of the detection/scoring area, features relevant to the area of concern must be determined and created. One approach is to build features primarily from the application-level business risks represented in DBT-layer tables, which have more business context than the raw source-system data structured via CDC processes in the transpiled-layer tables. Such feature sets can be built for different types of analytical approaches/models in relation to a specific risk. Once a suitable feature set exists, a variety of techniques can be used to create a risk score, which can be rolled up to various levels of hierarchy based on its scope. An alternative approach is to create features from the raw transaction content and then combine the resulting risk scores.

Equation 5: Observability metrics formalization (what "Fig 2" implies)

- **Freshness** at time t:
  - Let $t_{\max}$be latest event time successfully published.

$$F(t) = t - t_{\max}$$

- **Schema change rate**:

$$SCR(\text{day}) = \#\text{schema changes per day}$$

- **Error rate**:

$$ER(W) = \frac{N_{\text{errors}}(W)}{N_{\text{processed}}(W)}$$

VI. DATA GOVERNANCE, PRIVACY, AND SECURITY

Architectures for auditing transactions in real time reveal multiple demands for governance, privacy, and security that are addressed by various building-block technologies. These demands create a tight coupling among architectural elements and require careful consideration of provenance support, data-model choice, and privacy protection through access-control models. Auditing typically rises to sensor and operable status, demanding both the data-product models from source systems and pervasive validation for all business-sensitive data flows.

Data governance is the collection of processes and functions that ensure the availability, usability, integrity, and security of data used in an organization. In data engineering, it is often related to lineage, privacy, and access control. Privacy impacts how sensitive data are labeled, how sprinkling is employed for choice queries, and how data-access APIs are constrained. Because auditing products may easily have sensitive information, control should resort to a least-privilege model: no user or process should have more privilege than is necessary for the task at hand. Provenance enables reliable tracing of records through the system and ensures that any information that must remain secret does not leak through an apparent-vantage operation.

6.1. Access control models and least privilege

Data-generated insights are not valuable if unauthenticated users can access them. Besides controlling access to risk-model, ML checkpoints, and archival or decorators data, organizations should also put models on mission-critical data in their connected systems to ensure data governance demands are respected. A governance-by-obscurity principle offers another complementary access control model, applying an obfuscation filter directly when the data is used, only revealing the information that the user is entitled to see. PDPA requirements for sensitive customer information can be handled privately and without affecting the overall architecture.

Data exposure and update trails are essential when addressing sensitive data or information. Access to PDPA-protected data can be controlled using a governance-by-obscurity concept. By continuously tagging such sensitive attributes in

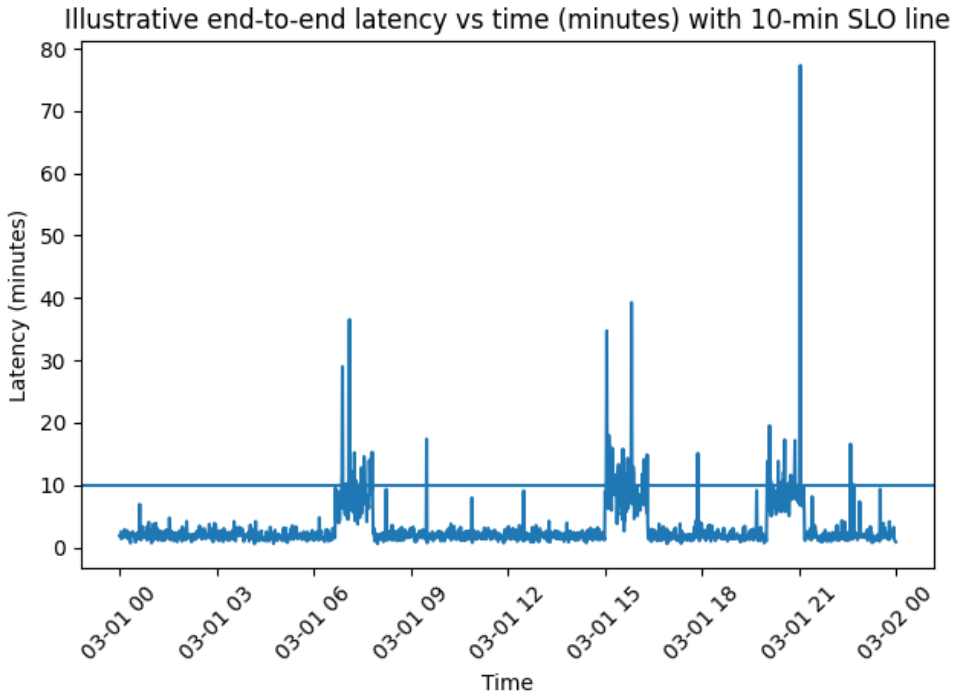
International Journal of Science, Research and Technology (IJSRAT)
|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

the transactions when they arrive, an additional access-control layer functionality for masking the sensitive data can be implemented. An additional step can check the necessary permissions of the user accessing the data. The data is stored in an encrypted state, and the decryption keys are only exposed to users who have access to the data, following the "need to know" principle for access control. Events generated from model use can also be logged as part of the audit of these sensitive data.

6.2. Data lineage and audit trails

Access control mechanisms that enforce the principle of least privilege are critical in any enterprise architecture that uses an external platform for data storage and computing. Storing personally identifiable information (PII) without appropriate safeguards can lead to a loss of competitive advantage or regulatory violations; revealing proprietary algorithm details could render these unique propositions useless. Although lakehouses facilitate dynamic change-data capture and minimize the operational overhead of maintaining multiple data formats, unintended data leaks are possible.

Provenance for all batch and stream-processing operations must be guaranteed, ensuring that analysts working with prepared datasets are aware of any modifications that have been applied. Observability features and built-in monitoring can help, but organizations must also apply fine-grained access-management policies for each table, column, and row, even for data packaged for presentation. Audit trails must span external cloud resources, data governance and observability tools, and other platforms that are integrated with the ecosystem, enabling analysts and operations teams to backtrack and detect the reason for any anomaly or unexpected event with minimal effort.



VII. CONCLUSION

Emerging technologies can enhance these systems. Real-time virtualized views over transactional data can now be created with data-lake technology for forward-looking detection. Sensor data connected to financial transactions allow monitoring of countries and customers for anti-money laundering purposes. Geolocation for credit card transactions implies risk scores enriched with machine-learning models for scam detection. Ideas initially developed for online game management systems can be reused for fraud detection and real-time communication with customers. Graph databases enable the output of end-user-dedicated graphs to assess customer relationships with instant notification on wise or fraudulent situations. And the growing relevance of open-source tools such as Airflow enables the implementation of light-weight-but-reliable engines for control and obsolescence alerts.

International Journal of Science, Research and Technology (IJSRAT)

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

An agile development approach allows rapid prototyping and incremental improvement of the data engineering layers sustaining these solutions. Requests for new features or changes are accommodated and delivered in weeks, rather than months or years. These environments can adapt and incorporate changes from the business and regulatory worlds at the speed of a click.

While real-time video and audio monitoring solutions use well-crafted convolutional neural nets, risk-scoring services usually rely more on assembling well-designed features, perhaps by shallow models or rule engines. Because analyst-led custom features may affect risk sensitive processes, updates to existing risk-scoring models should be carefully considered to reflect an appropriate trade-off of risk and recall; recall is often irretrievably.

SLI	Definition	Why it matters
Ingestion completeness	C = ingested / produced	Detect missing CDC/messages
End-to-end latency	L = t_available - t_commit	Meet <10 min SLA for risk scoring
Freshness	F = now - t_max_event	Staleness of analytical view
Schema change rate	SCR = #schema_changes / day	Uncontrolled evolution alarms
Duplicate rate	DR = duplicates / total	CDC replays, idempotency issues

Table : Example observability metrics table

7.1. Emerging Trends

Among emerging trends in data engineering is the increasing use of so-called ”lakehouse” storage solutions, which bring together the advantages of data lake and warehouse systems while avoiding some of their weaknesses. A lakehouse allows raw and processed ingested data to be stored and queried from the same place, merging different optimization strategies and unique functionality in a single solution. Jiang et al. propose lakehouse architecture elements and interoperable systems to simplify operationalizing end-to-end machine learning workflows.

Many industrial applications have the ability to leverage batch processes without modifying user-facing services. A common practice is to refresh an analytical view once a day with the previous day’s data or to add another night’s worth of data to a view built from one of the previous days. As part of a batch process, an analytical view might avoid the cost of maintaining the current accurate view and instead only refresh it with belated but accurate changes. Nonetheless, moving from bulk processing to real-time updates in a risk-scoring solution does incur consideration. Audio, video, and photo services seek to detect and respond to anomalies after the occurrences, classification, embedding, searching, recommendation, and enhancement have been processed. The key challenge in such designs is overcoming inappropriate timing. For example, developers often want to access appropriate products after the user has checked out.

REFERENCES

[1] Agustí, M., & Orta-Pérez, M. Continuous assurance and the monitoring of performance obligations in real-time streaming platforms. Journal of Accounting and Finance Research.
[2] Aitha, A. R. (2022). Cloud Native ETL Pipelines for Real Time Claims Processing in Large Scale Insurers. Available at SSRN 5532601.
[3] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749.
[4] Dwaraka Nath Kummari. (2022). Fiscal Policy Simulation Using AI And Big Data: Improving Government Financial Planning. Kurdish Studies, 10(2), 934–945. https://doi.org/10.53555/ks.v10i2.3855
[5] Arasu, A., & Kaushik, R. (2014). Data cleansing: A context dependent approach. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 135–146.
[6] Rongali, S. K. (2020). Predictive Modeling and Machine Learning Frameworks for Early Disease Detection in Healthcare Data Systems. Current Research in Public Health, 1(1), 1-15.
[7] Armbrust, M., Das, T., Davidson, A., Ghodsi, A., Or, A., Rosen, J., Stoica, I., Wendell, P., Xin, R., & Zaharia, M. (2021). Delta Lake: High-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment, 13(12), 3411–3424.

|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

[8] Inala, R. Advancing Group Insurance Solutions Through Ai-Enhanced Technology Architectures And Big Data Insights.
[9] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58.
[10] Chava, K., Chakilam, C., & Recharla, M. (2021). Machine Learning Models for Early Disease Detection: A Big Data Approach to Personalized Healthcare. International Journal of Engineering and Computer Science, 10(12), 25709–25730. https://doi.org/10.18535/ijecs.v10i12.4678
[11] Babcock, J., Chaudhuri, S., & Das, G. (2004). Dynamic sample selection for approximate query processing. Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, 539–550.
[12] Sriram, H. K. (2022). Advancements in Credit Score Analytics using Deep Learning and Predictive Modeling Techniques. Available at SSRN 5255128.
[13] Davuluri, P. N. (2020). Improving Data Quality and Lineage in Regulated Financial Data Platforms. Finance and Economics, 1(1), 1-14.
[14] Muthusamy, S., Kannan, S., Lee, M., Sanjairaj, V., Lu, W. F., Fuh, J. Y., ... & Cao, T. (2021). Cover Image, Volume 118, Number 8, August 2021. Biotechnology and Bioengineering, 118(8), i-i.
[15] Maguluri, K. K., Pandugula, C., Kalisetty, S., & Mallesham, G. (2022). Advancing Pain Medicine with AI and Neural Networks: Predictive Analytics and Personalized Treatment Plans for Chronic and Acute Pain Managements. Journal of Artificial Intelligence and Big Data, 2(1), 112-126.
[16] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2021). Privacy-Preserving Gen AI in Multi-Tenant Cloud Environments. Sateesh kumar and Raghunath, Vedaprada and Jyothi, Vinaya Kumar and Kudithipudi, Karthik, Privacy-Preserving Gen AI in Multi-Tenant Cloud Environments (January 20, 2021).
[17] Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile Networks and Applications, 19(2), 171–209.
[18] Sriram, H. K., ADUSUPALLI, B., & Malempati, M. (2021). Revolutionizing Risk Assessment and Financial Ecosystems with Smart Automation, Secure Digital Solutions, and Advanced Analytical Frameworks.
[19] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794.
[20] Gadi, A. L. The Role of Digital Twins in Automotive R&D for Rapid Prototyping and System Integration.
[21] Das, T., Zhu, A., Li, S., Narayanamurthy, S., & Bhat, P. (2013). Distributed and fault-tolerant streaming computation in Spark. Proceedings of the ACM Symposium on Cloud Computing, 1–12.
[22] Siva Hemanth Kolla. (2022). Knowledge Retrieval Systems for Enterprise Service Environments. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 495–506. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/8037
[23] Goutham Kumar Sheelam, "Semiconductor Innovation for Edge AI: Enabling Ultra-Low Latency in Next-Gen Wireless Networks," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2022.111258
[24] Paleti, S. (2022). Financial Innovation through AI and Data Engineering: Rethinking Risk and Compliance in the Banking Industry. Available at SSRN 5250726.
[25] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., & Vogels, W. (2007). Dynamo: Amazon's highly available key-value store. Proceedings of the 21st ACM Symposium on Operating Systems Principles, 205–220.
[26] Kalisetty, S., Vankayalapati, R. K., Reddy, L., Sondinti, K., & Valiki, S. (2022). AI-Native Cloud Platforms: Redefining Scalability and Flexibility in Artificial Intelligence Workflows. Linguistic and Philosophical Investigations, 21(1), 1-15.
[27] Dwork, C. (2008). Differential privacy: A survey of results. Proceedings of the 5th International Conference on Theory and Applications of Models of Computation, 1–19.
[28] Varri, D. B. S. (2021). Cloud-Native Security Architecture for Hybrid Healthcare Infrastructure. Available at SSRN 5785982.
[29] Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering, 19(1), 1–16.
[30] Kothapalli Sondinti, L. R., & Syed, S. (2022). The Impact of Instant Credit Card Issuance and Personalized Financial Solutions on Enhancing Customer Experience in the Digital Banking Era. Universal Journal of Finance and Economics, 1(1), 1223. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1223
[31] Fader, P. S., Hardie, B. G. S., & Lee, K. L. (2005). "Counting your customers" the easy way: An alternative to the Pareto/NBD model. Marketing Science, 24(2), 275–284.
[32] Inala, R. (2022). Engineering Data Products for Investment Analytics: The Role of Product Master Data and Scalable Big Data Solutions. International Journal of Scientific Research and Modern Technology, 155-171.

|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

[33] Davuluri, P. N. (2020). Improving Data Quality and Lineage in Regulated Financial Data Platforms. Finance and Economics, 1(1), 1-14.
[34] Meda, R. Enabling Sustainable Manufacturing Through AI-Optimized Supply Chains.
[35] Ghemawat, S., Gobioff, H., & Leung, S. T. (2003). The Google file system. Proceedings of the 19th ACM Symposium on Operating Systems Principles, 29–43.
[36] Varri, D. B. S. (2022). A Framework for Cloud-Integrated Database Hardening in Hybrid AWS-Azure Environments: Security Posture Automation Through Wiz-Driven Insights. International Journal of Scientific Research and Modern Technology, 1(12), 216-226.
[37] Yandamuri, U. S. (2021). A Comparative Study of Traditional Reporting Systems versus Real-Time Analytics Dashboards in Enterprise Operations. Universal Journal of Business and Management, 1(1), 1–13. Retrieved from https://www.scipublications.com/journal/index.php/ujbm/article/view/1357
[38] Gottimukkala, V. R. R. (2022). Licensing Innovation in the Financial Messaging Ecosystem: Business Models and Global Compliance Impact. International Journal of Scientific Research and Modern Technology, 1(12), 177-186.
[39] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). Springer.
[40] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2022). AI-Driven Cybersecurity: Enhancing Cloud Security with Machine Learning and AI Agents. Sateesh kumar and Raghunath, Vedaprada and Jyothi, Vinaya Kumar and Kudithipudi, Karthik, AI-Driven Cybersecurity: Enhancing Cloud Security with Machine Learning and AI Agents (February 07, 2022).
[41] Hellerstein, J. M., Haas, P. J., & Wang, H. J. (1997). Online aggregation. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, 171–182.
[42] Garapati, R. S. (2022). Web-Centric Cloud Framework for Real-Time Monitoring and Risk Prediction in Clinical Trials Using Machine Learning. Current Research in Public Health, 2, 1346.
[43] Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. Proceedings of the 2008 IEEE International Conference on Data Mining, 263–272.
[44] Amistapuram, K. (2022). Fraud Detection and Risk Modeling in Insurance: Early Adoption of Machine Learning in Claims Processing. Available at SSRN 5741982.
[45] Davuluri, P. S. L. N. (2021). Event-Driven Compliance Systems: Modernizing Financial Crime Detection Without Machine Intelligence. Journal of International Crisis and Risk Communication Research , 339–354. https://doi.org/10.63278/jicrcr.vi.3636
[46] Meda, R. (2022). Integrating Edge AI in Smart Factories: A Case Study from the Paint Manufacturing Industry. International Journal of Science and Research (IJSR), 1473-1489.
[47] Jagadish, H. V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R., & Shahabi, C. (2014). Big data and its technical challenges. Communications of the ACM, 57(7), 86–94.
[48] Segireddy, A. R. (2020). Cloud Migration Strategies for High-Volume Financial Messaging Systems.
[49] Khatri, V., & Brown, C. V. (2010). Designing data governance. Communications of the ACM, 53(1), 148–152.
[50] Amistapuram, K. (2021). Digital Transformation in Insurance: Migrating Enterprise Policy Systems to .NET Core. Universal Journal of Computer Sciences and Communications, 1(1), 1–17.
[51] Dwaraka Nath Kummari,. (2022). Machine Learning Approaches to Real-Time Quality Control in Automotive Assembly Lines. Mathematical Statistician and Engineering Applications, 71(4), 16801–16820. Retrieved from https://philstat.org/index.php/MSEA/article/view/2972
[52] Nagabhyru, K. C. (2022). Bridging Traditional ETL Pipelines with AI Enhanced Data Workflows: Foundations of Intelligent Automation in Data Engineering. Available at SSRN 5505199.
[53] Lahiri, M., & Venkatasubramanian, S. (2013). Robust record linkage. Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 101–112.
[54] Rongali, S. K. (2021). Cloud-Native API-Led Integration Using MuleSoft and .NET for Scalable Healthcare Interoperability. Available at SSRN 5814563.
[55] Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). Mining of massive datasets (2nd ed.). Cambridge University Press.
[56] Rongali, S. K. (2022). AI-Driven Automation in Healthcare Claims and EHR Processing Using MuleSoft and Machine Learning Pipelines. Available at SSRN 5763022.
[57] Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, 7(1), 76–80.
[58] Meda, R. (2021). Digital Infrastructure for Predictive Inventory Management in Retail Using Machine Learning. International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI, 10.
[59] Lin, J., Kolcz, A., & Szymanski, B. K. (2012). Large-scale machine learning at Twitter. Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, 793–804.

|Bimonthly | Peer-Reviewed | Scholarly Journal|
|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|
DOI: 10.15662/IJSRAT.2022.0506004

[60] Sheelam, G. K. Power-Efficient Semiconductors for AI at the Edge: Enabling Scalable Intelligence in Wireless Systems. International Journal of Innovative Research in Electrical, Elec-tronics, Instrumentation and Control Engineering (IJIREEICE), DOI, 10.
[61] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. H. (2011). Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute.
[62] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Rongali, S. K., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2021). Legal and Ethical Considerations for Hosting GenAI on the Cloud. International Journal of AI, BigData, Computational and Management Studies, 2(2), 28-34.
[63] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. Proceedings of the International Conference on Learning Representations, 1–12.
[64] Ramesh Inala. (2022). Cross-Domain MDM Integration Using AI-Driven Data Governance: A Case Study In Financial Technology Architecture. Migration Letters, 19(2), 280–304. Retrieved from https://migrationletters.com/index.php/ml/article/view/11982
[65] Montoya, D. Y., Neto, A. M., & da Silva, A. S. (2016). A survey of entity resolution in big data. Journal of Big Data, 3(1), 1–22.
[66] Aitha, A. R. (2021). Optimizing Data Warehousing for Large Scale Policy Management Using Advanced ETL Frameworks.
[67] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster computing with working sets. Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, 1–7.
[68] Varri, D. B. S. (2022). AI-Driven Risk Assessment and Compliance Automation in Multi-Cloud Environments. Available at SSRN 5774924.
[69] Lehner, O. M., et al. (2022). Governance of the human-machine hand-off: Oversight committees for data pipelines and model updates. Journal of Applied Accounting Research.
[70] Segireddy, A. R. (2021). Containerization and Microservices in Payment Systems: A Study of Kubernetes and Docker in Financial Applications. Universal Journal of Business and Management, 1(1), 1–17.
[71] Zhai, C., & Massung, S. (2016). Text data management and analysis: A practical introduction to information retrieval and text mining. ACM & Morgan Claypool.
[72] Davuluri, P. N. (2020). Event-Driven Architectures for Real-Time Regulatory Monitoring in Global Banking.
[73] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135–146.
[74] Keerthi Amistapuram , "Energy-Efficient System Design for High-Volume Insurance Applications in Cloud-Native Environments," International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering (IJIREEICE), DOI 10.17148/IJIREEICE.2020.81209
[75] Goutham Kumar Sheelam. (2022). Reconfigurable Semiconductor Architectures For AI-Enhanced Wireless Communication Networks. Kurdish Studies, 10(2), 1027–1040. https://doi.org/10.53555/ks.v10i2.3867
[76] Duppati, G., et al. Liquidity risk and AI-themed exchange-traded funds (ETFs): Challenges for modern finance researchers. Financial Markets and Portfolio Management.
[77] Gottimukkala, V. R. R. (2021). Digital Signal Processing Challenges in Financial Messaging Systems: Case Studies in High-Volume SWIFT Flows.
[78] Bhasin, H., & Bhatia, P. (2020). Clickstream data mining for web analytics and customer behavior modeling: A review. ACM Computing Surveys, 53(6), 1–34.
[79] Kolla, S. H. (2021). Rule-Based Automation for IT Service Management Workflows. Online Journal of Engineering Sciences, 1(1), 1–14. Retrieved from https://www.scipublications.com/journal/index.php/ojes/article/view/1360
[80] Uday Surendra Yandamuri. (2022). Cloud-Based Data Integration Architectures for Scalable Enterprise Analytics. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 472–483. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/8005
[81] Abedjan, Z., Golab, L., & Naumann, F. (2016). Profiling relational data: A survey. The VLDB Journal, 24(4), 557–581.
[82] Yandamuri, U. S. (2022). Big Data Pipelines for Cross-Domain Decision Support: A Cloud-Centric Approach. International Journal of Scientific Research and Modern Technology, 1(12), 227–237. https://doi.org/10.38124/ijsrmt.v1i12.1111
[83] Dwaraka Nath Kummari. (2022). AI-Driven Audit Frameworks For Enhancing Compliance In Modern Manufacturing Systems. Migration Letters, 19(S8), 2150–2177. Retrieved from https://migrationletters.com/index.php/ml/article/view/11912
[84] Davuluri, P. N. Event-Driven Compliance Systems: Modernizing Financial Crime Detection Without Machine Intelligence.

|Bimonthly | Peer-Reviewed | Scholarly Journal|

|ISSN 2454-3853|www.ijsrat.com|editor@ijsrat.com|Vol.5, Issue 6, November-December 2022|

DOI: 10.15662/IJSRAT.2022.0506004

[85] Charoenwong, B., Kowaleski, Z. T., Kwan, A., & Sutherland, A. G. RegTech: Technology-driven compliance and its effects on profitability, operations, and market structure. Journal of Financial Economics, 154, 103792.
[86] Avinash Reddy Aitha. (2022). Deep Neural Networks for Property Risk Prediction Leveraging Aerial and Satellite Imaging. International Journal of Communication Networks and Information Security (IJCNIS), 14(3), 1308–1318. Retrieved from https://www.ijcnis.org/index.php/ijcnis/article/view/8609
[87] Akter, S., Hu, M., & Wang, Y. (2022). Deep-learning image classifiers and anomaly-detection engines in structured accounting evidence. International Journal of Information Management.
[89] Garapati, R. S. (2022). AI-Augmented Virtual Health Assistant: A Web-Based Solution for Personalized Medication Management and Patient Engagement. Available at SSRN 5639650.
[90] Gottimukkala, V. R. R. (2020). Energy-Efficient Design Patterns for Large-Scale Banking Applications Deployed on AWS Cloud. power, 9(12).
[91] Ahmad, M. A., Eckert, C., & Teredesai, A. (2018). Interpretable machine learning in healthcare. Proceedings of the ACM Conference on Health, Informatics, and Data Science, 1–10.
[92] Kalisetty, S., & Ganti, V. K. A. T. (2019). Transforming the Retail Landscape: Srinivas's Vision for Integrating Advanced Technologies in Supply Chain Efficiency and Customer Experience. Online Journal of Materials Science, 1, 1254.
[93] Aljabre, A. (2019). Cloud computing security in healthcare. Journal of King Saud University – Computer and Information Sciences, 31(1), 10–18.
[94] Kolla, S. K. (2021). Architectural Frameworks for Large-Scale Electronic Health Record Data Platforms. Current Research in Public Health, 1(1), 1–19. Retrieved from https://www.scipublications.com/journal/index.php/crph/article/view/1372
[94] Akanfe, O. A. (2022). Advancing digital financial inclusion: Data privacy, regulatory compliance, and cross-country cultural values in digital payment systems use (Doctoral dissertation, The University of Texas at San Antonio).
[95] Avinash Reddy Segireddy. (2022). Terraform and Ansible in Building Resilient Cloud-Native Payment Architectures. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 444–455. Retrieved from https://www.ijisae.org/index.php/IJISAE/article/view/7905