International Journal of Emerging Research in Engineering and Technology
Pearl Blue Research Group| Volume 6, Issue 1, pp. 50-53, 2025
ISSN: 3050-922X | https://doi.org/10.63282/3050-922X/IJERET-V6I1P107

Original Article

Building High-Performance ETL Pipelines with Incremental Data Loading

Suresh Raguraman
Technology Analyst, HCL Technologies Ltd, Bengaluru, India.

Received On: 13/02/2025 ***Revised On: 25/02/2025*** ***Accepted On: 27/02/2025*** ***Published On: 03/03/2025***

Abstract *- ETL (Extract, Transform, Load) pipelines play a critical role in modern data processing and analytics. Traditional full data loads present challenges such as performance bottlenecks, increased resource consumption, and inefficiency. Incremental data loading emerges as a robust solution, enabling optimized processing by handling only new or changed data. This piece delves into the basics of incremental data loading, benefits, major strategies, best practices, and technologies underpinning high-performance ETL pipelines.*

Keywords *- ETL, Data Integration, Load optimization, CDC, Checksum, Incremental Load*

1. Introduction

ETL pipelines enable the transfer and conversion of data between multiple systems for analytics, reporting, and decision-making. Effective data loading is vital in today's data ecosystems, where data volume keeps on increasing exponentially. The classic full data loads, in which the whole set of data is processed in each cycle, create redundancy, extra costs, delay, and risk of performance slowdown. Incremental data loading improves ETL processes by loading just the changed data, minimizing the workload and maximizing performance.

2. Understanding Incremental Data Loading

Incremental data loading is a method employed in ETL (Extract, Transform, Load) pipelines to load just new or updated data instead of reloading entire sets. Rather than carrying out complete data loads, which are costly and time-consuming, incremental loading recognizes changes from the previous data load and only updates the relevant records. The method greatly enhances efficiency, lessens processing time, and minimizes resource utilization.

2.1 Key Advantages Over Full Data Loads

- Quicker Processing – Because just new or updated records are being processed, incremental data loading lessens the processing time for extracting, transforming, and loading the data.
- Decreased Storage Usage – Entire data loads may result in redundant duplication of records. Incremental loads guarantee only relevant changes are stored, maximizing disk space use.
- Lower Compute Costs – Computing smaller subsets of data decreases CPU and memory consumption, resulting inlower cost of infrastructure, particularly in cloud-based ETL setups.

2.2 Types of Incremental Data Loads

2.2.1 Time-Based Increments

- The most straightforward method, where data is extracted based on a timestamp column (e.g., last_updated_at).
- The system retrieves only records modified after the last ETL run.
- This approach is simple but requires reliable timestamps and may miss late-arriving updates.

2.2.2 Change Data Capture (CDC)

- CDC detects, and monitors source database changes through the monitoring of transaction logs or database triggers.
- Log-based CDC reads from the transaction logs of the database directly, which is very efficient and less intrusive.
- Event-based CDC uses database triggers to record row-level changes, but it can add performance overhead.

2.2.3 Delta Comparisons

- Compares the current data with the past snapshot with hash functions or checksums to identify changes.
- Ideal for situations where timestamps are unreliable or not available.
- Expensive in terms of computation for large datasets but efficient in maintaining data integrity.

By using these incremental loading methods, organizations can maximize ETL performance, improve data freshness, and minimize processing costs. The selection of the method relies on the data source, business needs, and infrastructure limitations.

3. Designing a High-Performance ETL Pipeline with Incremental Loading

Having an effective ETL pipeline that handles large data volumes with high performance is imperative for today's data environments. Incremental loading is central to this optimization since it loads only the changed data (new, updated, or deleted) since the last extract, as opposed to reloading complete datasets. This dramatically minimizes resource usage, execution time, and storage.

Key strategies include:

3.1 Data Extraction Strategies

- Identifying Changed Data Effectively – Effective incremental extraction depends on the identification of new inserts or updated records by employing timestamps, versioning, or Change Data Capture (CDC).
- Applying CDC Mechanisms – CDC approaches, including log-based or trigger-based tracking, identify database modifications in real-time, enhancing accuracy and minimizing excess data scans.
- Using APIs and Event-Driven Architectures – Streaming APIs and event-driven platforms (e.g., Kafka, AWS Kinesis) enable the capturing of real-time changes in data, minimizing latency in processing.

3.2 Data Transformation Optimization

- New or Changed Record Processing Only – Rather than having to convert complete datasets, the ETL workflows ought to update and process merely changed records only, enhancing efficiency.
- Minimizing Data Duplication and Ensuring Consistency – Deduplication strategies (e.g., primary key checks, versioning) prevent redundant records and maintain data integrity.
- Using Parallel Processing and Distributed Computing – Leveraging multi-threading, Spark, or cloud-based parallel processing speeds up transformation tasks for large-scale data workloads.

3.3 Efficient Data Loading Techniques

- Upserts (Insert/Update) vs. Append-Only Approach – Upserts update an existing record when present or create a new record, whereas append-only strategies enter new data and do not touch existing entries. The selection lies in applications such as historical auditing vs. immediate updates.
- Bulk Loading Optimizations for Databases – With batch inserts, index handling, and load utility optimizations on databases (e.g., Snowflake's COPY, Redshift's COPY) performance is improved.
- Effective Handling of Schema Changes – Evolving schemas are managed using schema-on-read, versioning, and automatic schema evolution (such as BigQuery's flexible schema) to avoid ETL failures.

By adopting these practices, ETL pipelines are able to optimize efficiency, reduce resource utilization, and ensure high data accuracy.

4. Technologies and Tools for Incremental Data Loading

Various technologies and tools can be used to automate the process and make it more efficient to carry out incremental data loading within an ETL process. The tools are meant to take care of different components of data extraction, transformation, and loading, to keep resource utilization low and performance at its best. Such key technologies are:

4.1 Databases and Data Warehouses

- Snowflake, BigQuery, Redshift, Databricks, etc. – Cloud data warehouses today natively support incremental data loading with features such as partitioning, clustering, and storage optimization.
- Incremental Loading Support – These systems support Change Data Capture (CDC), MERGE operations, and time-based partitioning to handle incremental data updates efficiently.

4.2 ETL Tools and Frameworks

- Apache NiFi, Talend, Informatica, Airflow, debt, etc. – They support incremental ETL processing through CDC, workflow automation, and data pipeline orchestration.
- Incremental Processing Best Practices – Using metadata-driven ETL, dependency management, and monitoring guarantees efficient and fault-free incremental data loads.

4.3 Cloud and Serverless Solutions

- AWS Glue, Azure Data Factory, Google Dataflow – Cloud-native ETL solutions support scalable, managed data integration with incremental processing natively supported.
- Taking Advantage of Serverless ETL for Scalability and Cost Savings – Serverless architectures automatically scale with data volume, lowering operational overhead and maximizing costs.

4.4 Change Data Capture (CDC)

- CDC tools, such as Debezium, Apache Kafka, and Talend, allow the capture of changes made to the source data in real time, enabling efficient extraction of only modified records. This eliminates the need for full data scans, thus optimizing load times and reducing overhead.

By selecting the right combination of these technologies, organizations can streamline incremental data loading, improve efficiency, and maintain real-time data freshness.

5. Common Pitfalls and How to Avoid Them

- Inaccurate Change Tracking – False records about new or updated records might create incomplete or redundant records. Implementation using efficient CDC channels and time stamping will minimize chances of incorrectness.
- Slow Indexing and Query Performance Problems – Low indexing can sluggish incremental queries. Appropriate indexing, partitioning, and query optimization improve performance.
- Management of Out-of-Order and Late-Arriving Data – Data might arrive late as a result of system latency. Event timestamps in place of ingestion timestamps and reprocessing measures can avoid this problem.

6. Best Practices for Scalability and Performance

- Partitioning and Indexing Strategies – Partitioning data according to time, ID, or category ensures efficient queries and lowers processing time.
- Monitoring and Logging Incremental Loads – Enabling automated logging, monitoring, and alerting enables early detection of failures and performance bottlenecks.
- Automating Failure Recovery and Retries – Using retry logic and checkpointing guarantees failed incremental loads resume without data loss.
- Resource Scaling - Dynamically scale resources concerning the volume of data or processing needs by utilizing cloud resources (e.g., AWS Lambda, Azure Functions) or containerized environments (e.g., Kubernetes) to ensure resource allocation can address demand. Implement auto-scaling and load balancing to manage high data volumes effectively in distributed ETL environments.
- Asynchronous Processing - Utilize asynchronous processing for activities like data loading, transformation, and extraction to enable the pipeline to process several steps simultaneously and thus enhance throughput and minimize waiting times.
- Data Integrity and Consistency - Employ idempotent operations to re-execute an incremental load, not add duplicates or create inconsistencies in the data. Apply transactional integrity to ensure data updates are done atomically, preventing partial data loads or corruption.
- Incremental Loading in Batches - Load data in small, manageable batches instead of loading everything simultaneously. This minimizes the chances of system resource exhaustion and enhances reliability. Stagger batch loads to prevent high load spikes and ensure the pipeline is efficient under various workload conditions.

7. Conclusion

Incremental data loading is an important method that enhances the effectiveness of ETL (Extract, Transform, Load) processes by loading only new or updated data in each cycle. This differs from full data loads, where all datasets are extracted, transformed, and loaded every time, irrespective of changes. By concentrating on incremental changes, organizations can decrease the time, cost, and computational resources involved in ETL processes, which is especially helpful when dealing with large-scale data environments.

Advanced techniques for incremental data loading involve using timestamps, Change Data Capture (CDC), and log-based methods to monitor data changes. These techniques enable identifying the precise data that must be updated, inserted, or deleted. Contemporary ETL software and cloud platforms support incremental loading natively, offering organizations out-of-the-box solutions to integrate these techniques into their data processes seamlessly.

The advantages of incremental data loading go beyond operational effectiveness. Optimizing the data pipeline allows companies to process data faster, leading to better decision-making and real-time analytics. Moreover, by reducing the amount of data being processed, organizations can lower their infrastructure expenses, including storage and processing capacity, since fewer resources are utilized per ETL run.

As data ecosystems expand and mature, incremental data loading will continue to be a critical element in the architecture of high-performance ETL solutions. As data volume, variety, and velocity increase, this approach guarantees that organizations can scale their data processes without compromising performance or incurring excessive costs. As companies continue to depend on data-driven insights, incremental data loading will remain a critical facilitator of scalable, efficient, and cost-effective ETL pipelines.

References

[1] Biswas, Neepa, Anamitra Sarkar, and Kartick Chandra Mondal. "Efficient incremental loading in ETL processing for real-time data integration." Innovations in Systems and Software Engineering 16, no. 1 (2020): 53-61.

[2] Dhamotharan Seenivasan, "ETL (Extract, Transform, Load) Best Practices," International Journal of Computer Trends and Technology, vol. 71, no. 1, pp. 40-44, 2023. Crossref, https://doi.org/10.14445/22312803/IJCTT-V71I1P106

[3] Rahman, Nayem. "Incremental load in a data warehousing environment." International Journal of Intelligent Information Technologies (IJIIT) 6, no. 3 (2010): 1-16.

[4] Wiener, J., and J. Naughton. Incremental loading of object databases. Stanford InfoLab, 1996.

[5] Jörg, Thomas, and Stefan Dessloch. "Formalizing ETL jobs for incremental loading of data warehouses." Datenbanksysteme in Business, Technologie und Web (BTW)–13. Fachtagung des GI-Fachbereichs" Datenbanken und Informationssysteme"(DBIS) (2009).

[6] Dhamotharan Seenivasan, "Improving the Performance of the ETL Jobs," International Journal of Computer Trends and Technology, vol. 71, no. 3, pp. 27-33, 2023. Crossref, https://doi.org/10.14445/22312803/IJCTT-V71I3P105

[7] Mekterović, Igor, and Ljiljana Brkić. "Delta view generation for incremental loading of large dimensions in a data warehouse." In 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1417-1422. IEEE, 2015.

[8] Julakanti, Sivananda Reddy, Naga Satya Kiranmayee Sattiraju, and Rajeswari Julakanti. "Incremental Load and Dedup Techniques in Hadoop Data Warehouses." NeuroQuantology 20, no. 5 (2022): 5626-5636.

[9] Hsieh, Hui-Ching, and Mao-Lun Chiang. "The incremental load balance cloud algorithm by using dynamic data deployment." Journal of Grid Computing 17 (2019): 553-575.

[10] Henke, Elisa, Yuan Peng, Ines Reinecke, Michéle Zoch, Martin Sedlmayr, and Franziska Bathelt. "An extract-transform-load process design for the Incremental Loading of German Real-World Data based on FHIR and OMOP CDM: Algorithm Development and Validation." JMIR Medical Informatics 11 (2023): e47310.

[11] Murray, Derek G., Frank McSherry, Michael Isard, Rebecca Isaacs, Paul Barham, and Martin Abadi. "Incremental, iterative data processing with timely dataflow." Communications of the ACM 59, no. 10 (2016): 75-83.

[12] Dhamotharan Seenivasan, Muthukumaran Vaithianathan, 2023. "Real-Time Adaptation: Change Data Capture in Modern Computer Architecture" ESP International Journal of Advancements in Computational Technology (ESP-IJACT) Volume 1, Issue 2: 49-61, https://www.espjournals.org/IJACT/ijact-v1i2p106

[13] Jagadish, H. V., P. P. S. Narayan, Sridhar Seshadri, S. Sudarshan, and Rama Kanneganti. "Incremental organization for data recording and warehousing." In VLDB, pp. 16-25. ResearchGate GmbH, 1997.

[14] Vijayalakshmi, M., and R. I. Minu. "Incremental load processing on ETL system through cloud." In 2022 International Conference for Advancement in Technology (ICONAT), pp. 1-4. IEEE, 2022.

[15] Zhang, Xufeng, Weiwei Sun, Wei Wang, Yahui Feng, and Baile Shi. "Generating incremental ETL processes automatically." In First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06), vol. 2, pp. 516-521. IEEE, 2006.

[16] Ong, Toan C., Michael G. Kahn, Bethany M. Kwan, Traci Yamashita, Elias Brandt, Patrick Hosokawa, Chris Uhrich, and Lisa M. Schilling. "Dynamic-ETL: a hybrid approach for health data extraction, transformation and loading." BMC medical informatics and decision making 17 (2017): 1-12.

[17] Behrend, Andreas, and Thomas Jörg. "Optimized incremental ETL jobs for maintaining data warehouses." In Proceedings of the Fourteenth International Database Engineering & Applications Symposium, pp. 216-224. 2010.

[18] Reddy, V. Mallikarjuna, and Sanjay K. Jena. "Active datawarehouse loading by tool based ETL procedure." (2010).

[19] Dhamotharan Seenivasan, "Critical Security Enhancements for ETL Workflows:Addressing Emerging Threats and Ensuring Data Integrity",International Journal of Innovative Research in Computer and Communication Engineering(IJIRCCE), Volume 12, Issue 3, March 2024, pp. 1301-1313, https://ijircce.com/admin/main/storage/app/pdf/XGaYnZXucV5i5bL5exiKjJYN9DX7xm7B0GTq8ivj.pdf

[20] Bathani, Ronakkumar. "Optimizing Etl Pipelines for Scalable Data Lakes in Healthcare Analytics." International Journal on Recent and Innovation Trends in Computing and Communication 9, no. 10 (2021): 17-24.

[21] Atmaja, I. Putu Medagia, Ari Saptawijaya, and Siti Aminah. "Implementation of change data capture in ETL process for data warehouse using HDFS and Apache spark." In 2017 International Workshop on Big Data and Information Security (IWBIS), pp. 49-55. IEEE, 2017.

[22] Suleykin, Alexander, and Peter Panfilov. "Metadata-driven industrial-grade ETL system." In 2020 IEEE International Conference on Big Data (Big Data), pp. 2433-2442. IEEE, 2020.

[23] Khan, Bilal, Saifullah Jan, Wahab Khan, and Muhammad Imran Chughtai. "An Overview of ETL Techniques, Tools, Processes and Evaluations in Data Warehousing." Journal on Big Data 6 (2024).

[24] Arsyad, Zulkifli. "Analisis Dynamic ETL Incremental Load untuk Data Integration Datawarehouse." INTERNAL (Information System Journal) 4, no. 2 (2021): 102-112.

[25] Dhamotharan Seenivasan, "Effective Strategies for Managing Slowly Changing Dimensions in Data Warehousing", International Journal of Emerging Technologies and Innovative Research (www.jetir.org | UGC and ISSN Approved), ISSN:2349-5162, Vol.9, Issue 4, page no. ppi492-i496, April-2022,http://www.jetir.org/papers/JETIR2204861.pdf

[26] Gour, Vishal, S. S. Sarangdevot, Govind Singh Tanwar, and Anand Sharma. "Improve performance of extract, transform and load (ETL) in data warehouse." Int. Journal on Comp. Sci. and Eng 2, no. 3 (2010): 786-789.

[27] Qu, Weiping, Vinanthi Basavaraj, Sahana Shankar, and Stefan Dessloch. "Real-time snapshot maintenance with incremental ETL pipelines in data warehouses." In Big Data Analytics and Knowledge Discovery: 17th International Conference, DaWaK 2015, Valencia, Spain, September 1-4, 2015, Proceedings 17, pp. 217-228. Springer International Publishing, 2015.

[28] Raghuveer, K., and R. Dayanand. "Towards handling incremental load for anomalies in near real time data warehouse." WSEAS Transactions on Systems and Control 15 (2020): 684-690.

[29] Mekterović, Igor, and Ljiljana Brkić. "Delta view generation for incremental loading of large dimensions in a data warehouse." In 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1417-1422. IEEE, 2015.

[30] Simitsis, Alkis, Panos Vassiliadis, and Timos Sellis. "Optimizing ETL processes in data warehouses." In 21st International Conference on Data Engineering (ICDE'05), pp. 564-575. Ieee, 2005.