Cigniti

How to Define a CDC Architecture for Real-time Insights

A Tech Brief

Introduction

As data volumes continue to grow and business users demand incessant access to insights, real-time extraction of underlying data is no longer feasible. To support near real-time analytics applications, solutions must be deployed to identify and replicate a change log. A proper data ingestion strategy is critical to the success of any data lake. This tech brief will discuss how Kafka messaging engine best suits data ingestion from frequently refreshed data sources.

Change Data Capture in the real-time streaming environment can be stitched together through a data pipeline-based solution leveraging different tools that exist today.

Understanding Real-time Data Capture

Most transactions (operations) in the current world are real-time, resulting in constant CRUD operations (**C**reate, **R**ead, **U**pdate, and **D**elete) on transactional systems. However, typical data warehouse implementations have been D-1, where reports and dashboards always reflect KPIs as of yesterday. This was mainly due to:

- The defined ETL processes &
- The pipelines were created to work in batch mode.

In the present times, reports and dashboards should be as real-time as possible. While 'Change Data Capture' as a framework is fairly standardized and is enabled through different tools, the focus was not really on real-time data capture. **CDC, as a design pattern, allows the process to capture these types of changes and provides efficient integration with the rest of the enterprise systems.** The approach of capturing and processing only the changed data leads to all-around efficiency improvement, in terms of computing, performance, storage, and costs of ownership.

Flexibility with Options

We can achieve the CDC approach in multiple ways:

- By developing 'Database Triggers' at the source application database to extract, based on the change that happened.
- By implementing a comparison logic in the application layer to identify the changed data and create an alternate, yet continuous, pipeline to capture changed data.
- By implementing a dedicated CDC tool such as GoldenGate, Syncsort, etc.
- By implementing CDC platforms such as Confluent, Striim, Attunity, etc.
- Leveraging the CDC capture mechanism provided by the databases, such as SQLServer.
- Extracting changed data from the database transaction log files.

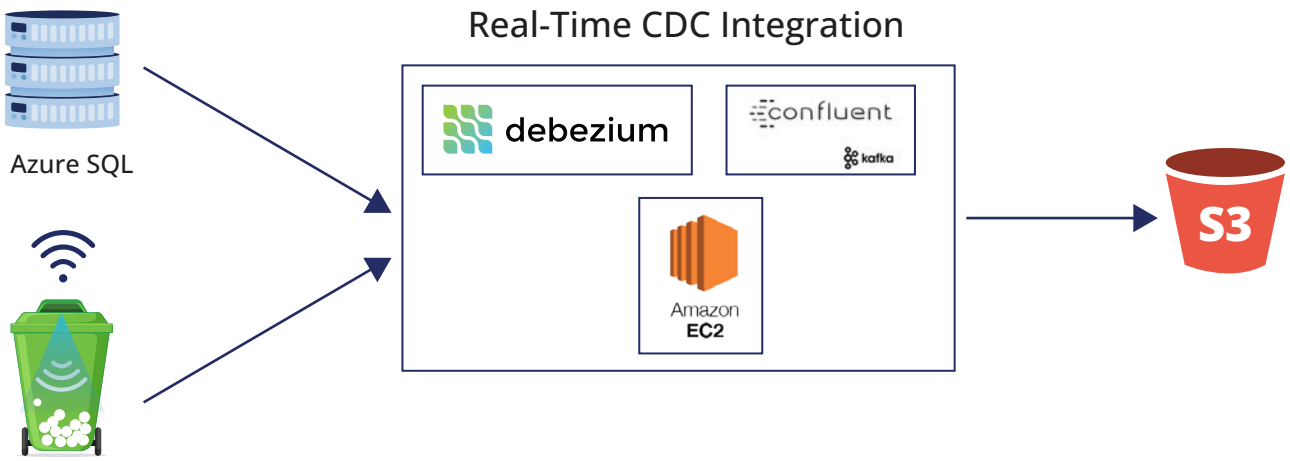
There are both advantages and disadvantages to each of these approaches. A few of them are listed below:

- The Database Triggers Approach is a time-tested traditional approach, however with the caveat that it impacts the operating system performance.
- Application Logic/Triggers is an approach that can only work if the record passes through the application layer. For example, a deleted record may not pass through the application layer and hence, gets missed in this approach.
- CDC tools are typically expensive and may not be in the affordability range for all projects.
- CDC tables feature is provided by only a few database vendors.
- The Database Log Files-based Approach is technically ideal, as it does not affect the OLTP system and can get to all types of changes that happened in the database. However, log-based CDC drivers for all the database systems are not available.

Our POV

Presented below is our analysis and findings in terms of implementing CDC-based real-time data integration using Kafka as the messaging engine.

- **CONTEXT:** Extract the changed data from the source system (Azure SQL) in real-time and process / transform it as a stream, using spark streaming, and store it in a star schema modelled RedShift database. The target state architecture for CDC integration is as follows:



Debezium Drivers:

JDBC Drivers for source connection and Debezium CDC driver are used in our implementation. Separate configuration files have been created to cater to inserts and updates at the source system

Confluent Kafka:

Confluent is a fully managed Apache Kafka service providing a Real-Time Stream Processing Platform for enterprises. Kafka is a highly scalable, fault-tolerant distributed system that provides a powerful event streaming platform through publish/subscribe messaging service. Data producers publish messages in the form of Kafka topics, and data consumers subscribe to these topics to receive data.

Property Files:

Two property files are created. One for the source property file (for Azure SQL database) and the second for the sink property file (for S3 bucket). The source property file consists of credentials information for source name, password, database name, username, and topic names, along with JDBC Source connectors class information.

Sink properties file contains details of S3 as below:

- S3 bucket name
- S3 location
- Default flush size as 3 records

Kafka Modes:

Kafka can be implemented in the following modes:

Incrementing: Uses a strictly incrementing column on each table to detect only new rows. Note that this will not detect modifications or deletions of existing rows.

Bulk: Performs a bulk load of the entire table each time it is polled

Timestamp: Uses a timestamp (or timestamp-like) column to detect new and modified rows. This assumes the column is updated with each write, and that values are monotonically incrementing, but not necessarily unique.

Timestamp + Incrementing: Uses two columns, a timestamp column that detects new and modified rows and a strictly incrementing column that provides a globally unique ID for updates so each row can be assigned a unique stream offset.

Query: Uses a custom query for polling the data from the source system

We implemented the Timestamp + Incrementing mode of Kafka as our system requires capturing both incremental (new records) as well as updated records.

- **KEY CONCEPTS AND TECHNOLOGIES:**
  - Live Data Extraction and CDC using Confluent Kafka
  - Data Transformation and Processing using PySpark Structured Streaming
  - Amazon Redshift Data warehouse design, setup and loading
  - Big Data Analytics on Amazon RedShift using Power BI

Data Pipeline:

Kafka reads data from the Azure SQL database and writes in Kafka's topic. The data pipeline consists of reading data from the Kafka topic and writing the raw data in the S3 bucket. Significant transformations, integrations, and aggregations are implemented in Spark streaming jobs and are written in S3, and then subsequently into the RedShift database.

PySpark:

park is a distributed processing framework for handling big data applications. It uses In-memory Caching and Optimized Query Execution for querying against large-scale data in a fast and efficient manner. PySpark is the Python API for Spark. While CDC is captured by Kafka, Spark is used to implement transformations and loading into the RedShift. A custom module is developed in Spark to handle the varying datatypes and column names between source systems and RedShift. The following configurations are done in the custom module:

Kafka	S3	RedShift
Host Name	S3 Path	Database Name
Multiple Topic Prefix	Access keys	Master User
		Password
		JDBC URL

Read stream: To read the data continuously from the Kafka topic.

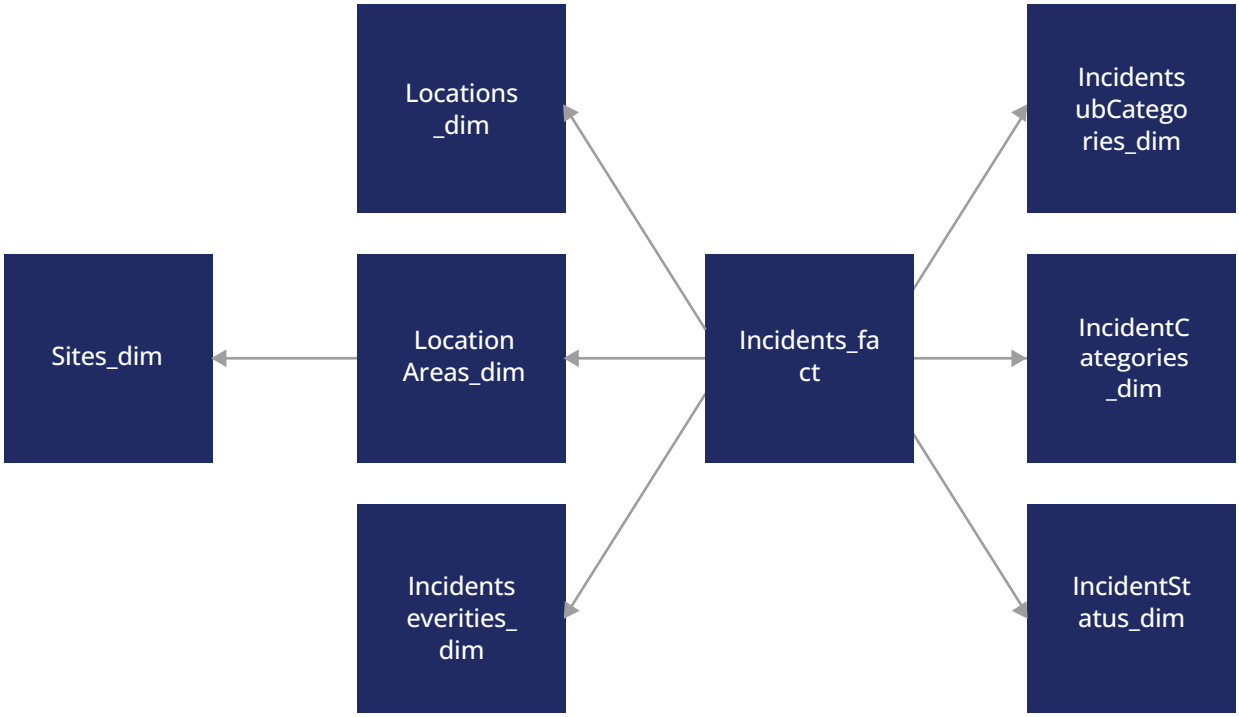
Starting Offset: "Earliest" offset is set to not lose data in case of failures.

Write Stream: To write data continuously to RedShift.

Temp Directory: Path to S3 data storage is given for continuous writing to RedShift.

Checkpoint Location: HDFS path is given where the offset of the data is stored in a way that duplication does not happen.

Redshift Data Modelling: Amazon Redshift is the fastest cloud-based data warehouse service to run high-performance queries on petabytes of data. The Redshift database is modeled following the star schema approach.



Conclusion

Change Data Capture, in the real-time streaming environment, can be stitched together through a data pipeline-based solution leveraging different tools that exist today. The implementation, that we have done, powers the dashboards for a Risk Management Platform that is used by leading malls and airports across Australia and New Zealand.

Success Stories

Senior IT leaders share how our services helped them win in the platform age.

CTO Speak

Data pipeline buildout, Software development, Salesforce development, AWS System admin/DevOps, BI/Dashboard. The execution has been very good.

- Satyadeep "Bobby" Patnaik, CTO

Lafayette Square

CEO Speak

They understood that product development was iterative and they patiently worked through our requirements even as they rapidly evolved.

- Dr. Ganesh Naidoo, CEO

medmate

CTO Speak

Proven technical ability in both web and mobile development; strong project/product management expertise; the ability to become part of the extended BA365 team.

- Graeme Dollar, CTO

BUSINESS ACCELERATOR 365

COO Speak

I have worked with hundreds of service providers and consultants, RoundSqr (Part of Cigniti) is absolutely one of the best. The people are highly skilled, very hard-working, and have a "can do" attitude.

- Mark Mortimer, COO

Adelphoi

Analyst Recognitions

NelsonHall recognized Cigniti as a "LEADER" i n its 2022 NEAT evaluation for Overall **Quality** Engineering, Continuous Testing, Application **Security Testing, and AI &** Cognitive

Cigniti is mentioned as a "Pure Play Testing Vendor" in Gartner's Market Guide for Application Testing Services, 2022

Cigniti is mentioned as "API Testing Vendor" in Gartner's Hype Cycle for Managed IT Services and APIs, 2022

Recognized as "LEADER" in ISG IPL for Next-Gen ADM Services under Continuous Testing Specialists category for the US region for 2021 and 2022

Recognized as "RISING STAR" in UK Region in ISG IPL for Next-Gen ADM Services 2022

Cigniti is recognized as a **Strong Performer** in the Forrester Wave: **Continuous Automation and Testing** Services, Q3 2021.

One of the top 3 leaders in Agile Testing and DevOps in the Forrester Wave: Continuous Automation and Testing Services, Q3 2017.

Provides Cigniti with **"Best in Class"** rating for Buyer satisfaction.

About Cigniti

Cigniti Technologies Limited (NSE: CIGNITITEC; BSE: 534758) is the World's Leading AI & IP-led Digital Assurance and Digital Engineering Services Company providing software quality engineering, software testing, automation, and consulting services. 4100+ Cignitians worldwide help Fortune 500 & Global 2000 enterprises across 24 countries accelerate their digital transformation journey across various stages of digital adoption and help them achieve market leadership by providing transformation services leveraging IP & Platform-led innovation with expertise across multiple verticals and domains.

Our global customers have benefitted with measurable outcomes, millions of dollars of savings, significant ROI, and delightful, frictionless experiences utilizing our flagship digital assurance and full cycle software quality engineering services. Our AI-led digital engineering services cover Data engineering services, softwareplatform, and digital product engineering, AI/ML engineering services, intelligent automation, big data analytics, and Blockchain development.

We are headquartered in Hyderabad, India, with global offices spread across the USA, Canada, UK, UAE, Australia, South Africa, Czech Republic, and Singapore.

To learn more, visit www.cigniti.com

Website LinkedIn YouTube Blog Facebook Twitter