SECTION

05

Analytics Use Cases on Databricks

SECTION 5.1

How to Build a Marketing Analytics Solution Using Fivetran and dbt on Databricks

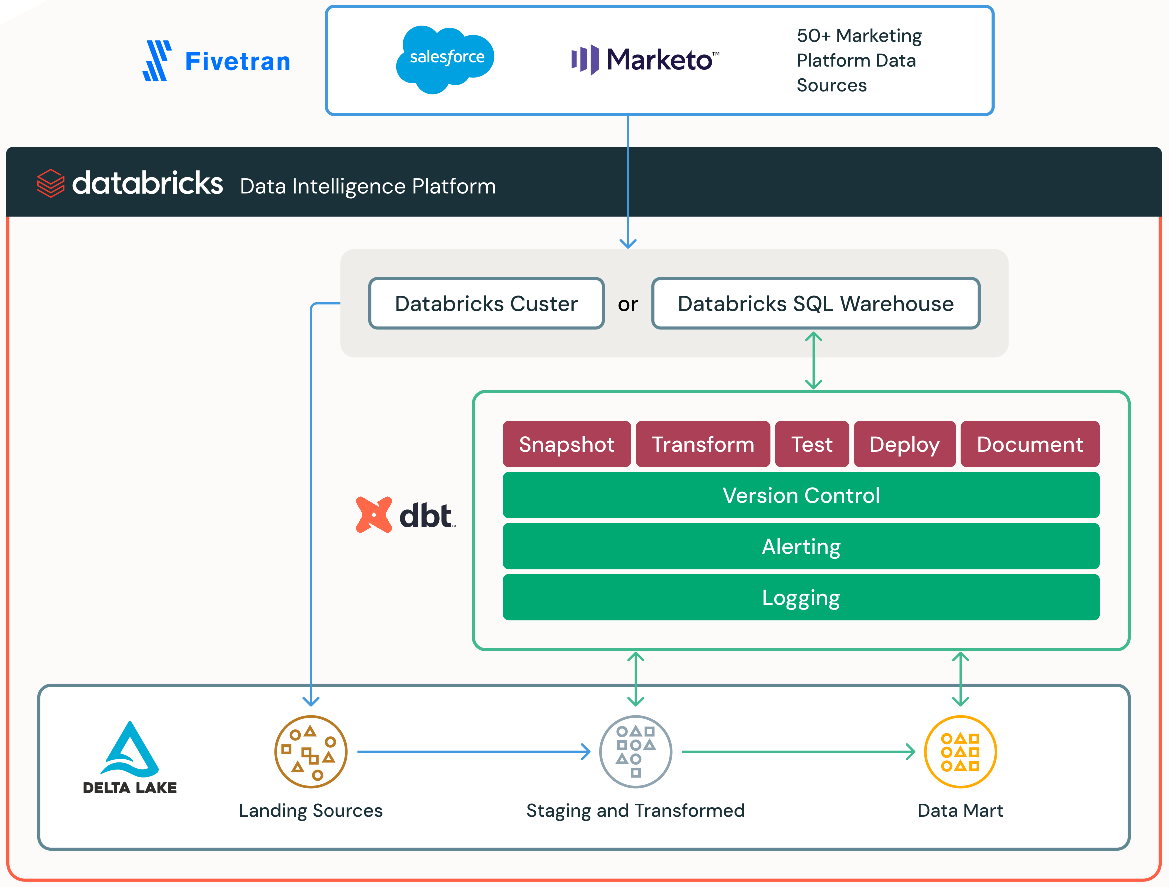
by Tahir Fayyaz, Bilal Aslam and Robert Saxby

Marketing teams use many different platforms to drive marketing and sales campaigns, which can generate a significant volume of valuable but disconnected data. Bringing all of this data together can help to drive a large return on investment, as shown by Publicis Groupe who were able to increase campaign revenue by as much as 50%.

Databricks, which unifies data warehousing and AI use cases on a single platform, is the ideal place to build a marketing analytics solution: we maintain a single source of truth, and unlock AI/ML use cases. We also leverage two Databricks partner solutions, Fivetran and dbt, to unlock a wide range of marketing analytics use cases including churn and lifetime value analysis, customer segmentation, and ad effectiveness.

Fivetran allows you to easily ingest data from 50+ marketing platforms into Delta Lake without the need for building and maintaining complex pipelines. If any of the marketing platforms' APIs change or break, Fivetran will take care of updating and fixing the integrations so your marketing data keeps flowing in.

dbt is a popular open source framework that lets lakehouse users build data pipelines using simple SQL. Everything is organized within directories, as plain text, making version control, deployment, and testability simple. Once the data is ingested into Delta Lake, we use dbt to transform, test and document the data. The transformed marketing analytics data mart built on top of the ingested data is then ready to be used to help drive new marketing campaigns and initiatives.



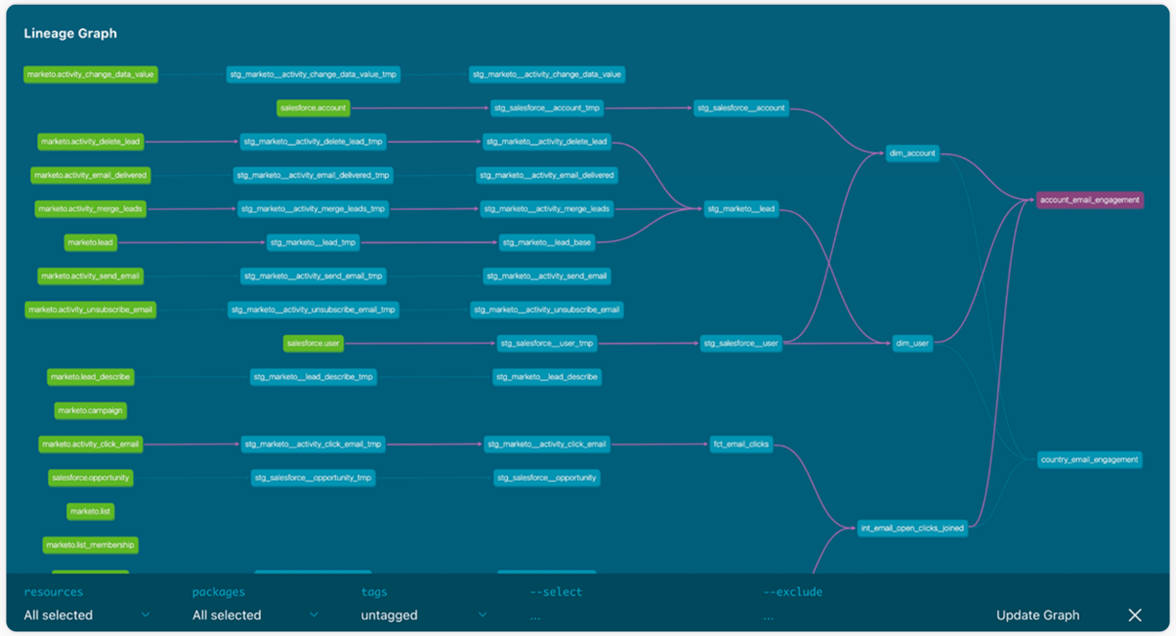
Fivetran and dbt can read and write to Delta Lake using a Databricks cluster or Databricks SQL warehouse

Both Fivetran and dbt are a part of Databricks Partner Connect, a one-stop portal to discover and securely connect data, analytics and AI tools directly within the Databricks Platform. In just a few clicks you can configure and connect these tools (and many more) directly from within your Databricks workspace.

How to build a marketing analytics solution

In this hands-on demo, we will show how to ingest Marketo and Salesforce data into Databricks using Fivetran and then use dbt to transform, test, and document your marketing analytics data model.

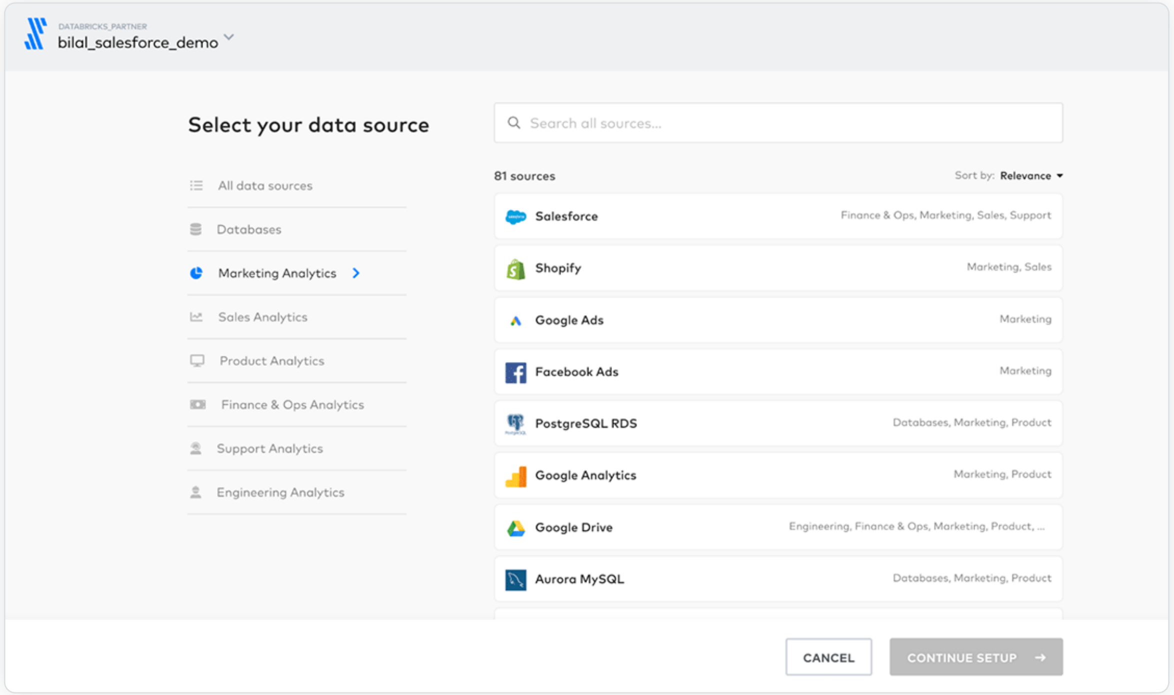
All the code for the demo is available on Github in the workflows-examples repository.


Lineage Graph
marketo.activity_change_data_value
stg_marketo__activity_change_data_value_tmp
stg_marketo__activity_change_data_value
salesforce.account
stg_salesforce__account_tmp
stg_salesforce__account
marketo.activity_delete_lead
stg_marketo__activity_delete_lead_tmp
stg_marketo__activity_delete_lead
dim_account
marketo.activity_email_delivered
stg_marketo__activity_email_delivered_tmp
stg_marketo__activity_email_delivered
account_email_engagement
marketo.activity_merge_leads
stg_marketo__activity_merge_leads_tmp
stg_marketo__activity_merge_leads
stg_marketo__lead
marketo.lead
stg_marketo__lead_tmp
stg_marketo__lead_base
marketo.activity_send_email
stg_marketo__activity_send_email_tmp
stg_marketo__activity_send_email
marketo.activity_unsubscribe_email
stg_marketo__activity_unsubscribe_email_tmp
stg_marketo__activity_unsubscribe_email
salesforce.user
stg_salesforce__user_tmp
stg_salesforce__user
dim_user
marketo.lead_describe
stg_marketo__lead_describe_tmp
stg_marketo__lead_describe
marketo.campaign
marketo.activity_click_email
stg_marketo__activity_click_email_tmp
stg_marketo__activity_click_email
fct_email_clicks
salesforce.opportunity
stg_salesforce__opportunity_tmp
stg_salesforce__opportunity
country_email_engagement
marketo.list
int_email_open_clicks_joined
marketo.list_membership
resources
All selected
packages
All selected
tags
untagged
--select
...
--exclude
...
Update Graph

dbt lineage graph showing data sources and models

The final dbt model lineage graph will look like this. The Fivetran source tables are in green on the left and the final marketing analytics models are on the right. By selecting a model, you can see the corresponding dependencies with the different models highlighted in purple.

Data ingestion using Fivetran

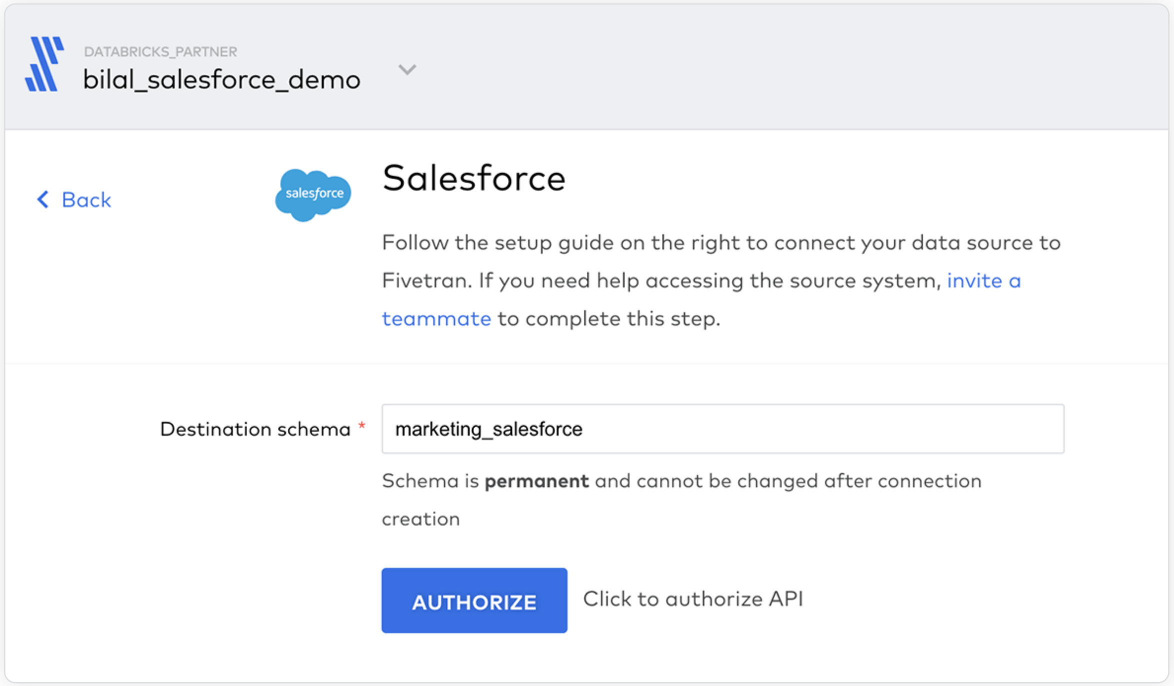


81 sources   Sort by: Relevance

Salesforce	Finance & Ops, Marketing, Sales, Support
Shopify	Marketing, Sales
Google Ads	Marketing
Facebook Ads	Marketing
PostgreSQL RDS	Databases, Marketing, Product
Google Analytics	Marketing, Product
Google Drive	Engineering, Finance & Ops, Marketing, Product, ...
Aurora MySQL	Databases, Marketing, Product

Fivetran has many marketing analytics data source connectors

Create new Salesforce and Marketo connections in Fivetran to start ingesting the marketing data into Delta Lake. When creating the connections Fivetran will also automatically create and manage a schema for each data source in Delta Lake. We will later use dbt to transform, clean and aggregate this data.



DATABRICKS_PARTNER
bilal_salesforce_demo

Back

Salesforce

Follow the setup guide on the right to connect your data source to Fivetran. If you need help accessing the source system, invite a teammate to complete this step.

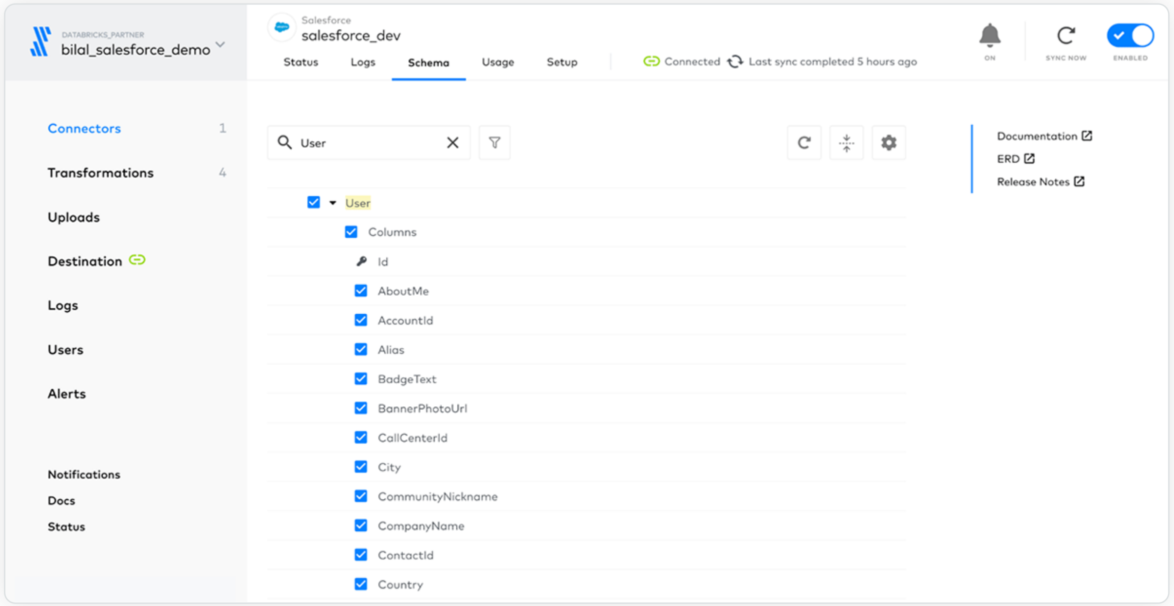
Destination schema * marketing_salesforce

Schema is **permanent** and cannot be changed after connection creation

AUTHORIZE Click to authorize API

Define a destination schema in Delta Lake for the Salesforce data source

For the demo name the schemas that will be created in Delta Lake marketing_salesforce and marketing_marketo. If the schemas do not exist Fivetran will create them as part of the initial ingestion load.



DATABRICKS_PARTNER
bilal_salesforce_demo

Salesforce
salesforce_dev

Status Logs Schema Usage Setup

Connected Last sync completed 5 hours ago

ON SYNC NOW ENABLED

Connectors 1
Transformations 4
Uploads
Destination
Logs
Users
Alerts

Notifications
Docs
Status

User

✓ User
✓ Columns
Id
✓ AboutMe
✓ AccountId
✓ Alias
✓ BadgeText
✓ BannerPhotoUrl
✓ CallCenterId
✓ City
✓ CommunityNickname
✓ CompanyName
✓ ContactId
✓ Country

Documentation
ERD
Release Notes

Select which data source objects to synchronize as Delta Lake tables

You can then choose which objects to sync to Delta Lake, where each object will be saved as individual tables. Fivetran also makes it simple to manage and view what columns are being synchronized for each table:

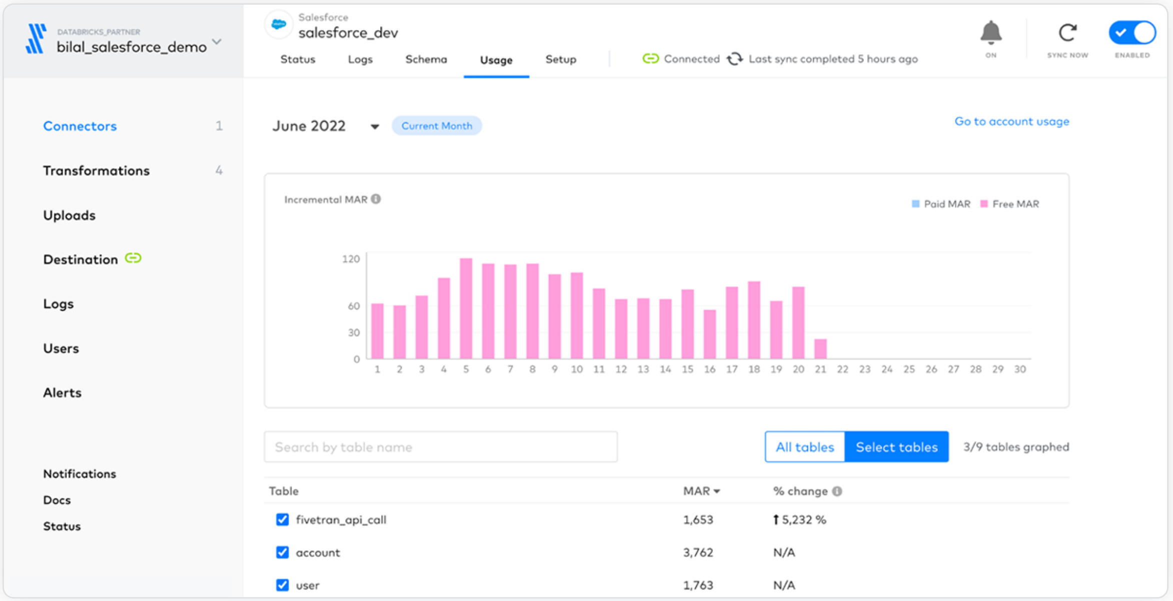


Table	MAR	% change
✓ fivetran_api_call	1,653	↑5,232 %
✓ account	3,762	N/A
✓ user	1,763	N/A

Fivetran monitoring dashboard to monitor monthly active rows synchronized

Additionally, Fivetran provides a monitoring dashboard to analyze how many monthly active rows of data are synchronized daily and monthly for each table, among other useful statistics and logs.

Data modeling using dbt

Now that all the marketing data is in Delta Lake, you can use dbt to create your data model by following these steps

Setup dbt project locally and connect to Databricks SQL

Set up your local dbt development environment in your chosen IDE by following the set-up instructions for dbt Core and dbt-databricks.

Scaffold a new dbt project and connect to a Databricks SQL Warehouse using dbt init, which will ask for following information.

```
$ dbt init
Enter a name for your project (letters, digits, underscore):
Which database would you like to use?
[1] databricks
[2] spark

Enter a number: 1
host (yourorg.databricks.com):
http_path (HTTP Path):
token (dapiXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX):
schema (default schema that dbt will build objects in):
threads (1 or more) [1]:
```

Once you have configured the profile you can test the connection using:

```
$ dbt debug
```

Install Fivetran dbt model packages for staging

The first step in using the Marketo and Salesforce data is to create the tables as sources for our model. Luckily, Fivetran has made this easy to get up and running with their pre-built Fivetran dbt model packages. For this demo, let's make use of the `marketo_source` and `salesforce_source` packages.

To install the packages just add a `packages.yml` file to the root of your dbt project and add the `marketo-source`, `salesforce-source` and the `fivetran-utils` packages:

```
packages:
  - package: dbt-labs/spark_utils
    version: 0.3.0
  - package: fivetran/marketo_source
    version: [">=0.7.0", "=0.4.0", "

To download and use the packages run
```

```
$ dbt deps
```

You should now see the Fivetran packages installed in the packages folder.

Update dbt_project.yml for Fivetran dbt models

There are a few configs in the `dbt_project.yml` file that you need to modify to make sure the Fivetran packages work correctly for Databricks.

The `dbt_project.yml` file can be found in the root folder of your dbt project.

spark_utils overriding dbt_utils macros

The Fivetran dbt models make use of macros in the `dbt_utils` package but some of these macros need to be modified to work with Databricks which is easily done using the `spark_utils` package.

It works by providing shims for certain `dbt_utils` macros which you can set using the dispatch config in the `dbt_project.yml` file and with this dbt will first search for macros in the `spark_utils` package when resolving macros from the `dbt_utils` namespace.

```
dispatch:
 - macro_namespace: dbt_utils
   search_order: ['spark_utils', 'dbt_utils']
```

Variables for the marketo_source and salesforce_source schemas

The Fivetran packages require you to define the catalog (referred to as database in dbt) and schema of where the data lands when being ingested by Fivetran.

Add these variables to the `dbt_project.yml` file with the correct catalog and schema names. The default catalog is `hive_metastore` which will be used if `_database` is left blank. The schema names will be what you defined when creating the connections in Fivetran.

```
vars:
 marketo_source:
   marketo_database: # leave blank to use the default hive_metastore catalog
   marketo_schema: marketing_marketo
 salesforce_source:
   salesforce_database: # leave blank to use the default hive_metastore catalog
   salesforce_schema: marketing_salesforce
```

Target schema for Fivetran staging models

To avoid all the staging tables that are created by the Fivetran source models being created in the default target schema it can be useful to define a separate staging schema.

In the `dbt_project.yml` file add the staging schema name and this will then be suffixed to the default schema name.

```
models:
 marketo_source:
   +schema: your_staging_name # leave blank to use the default target_schema
 salesforce_source:
   +schema: your_staging_name # leave blank to use the default target_schema
```

Based on the above, if your target schema defined in `profiles.yml` is **`mkt_analytics`**, the schema used for `marketo_source` and `salesforce_source` tables will be **`mkt_analytics_your_staging_name`**.

Disable missing tables

At this stage you can run the Fivetran model packages to test that they work correctly.

```
dbt run -select marketo_source
```

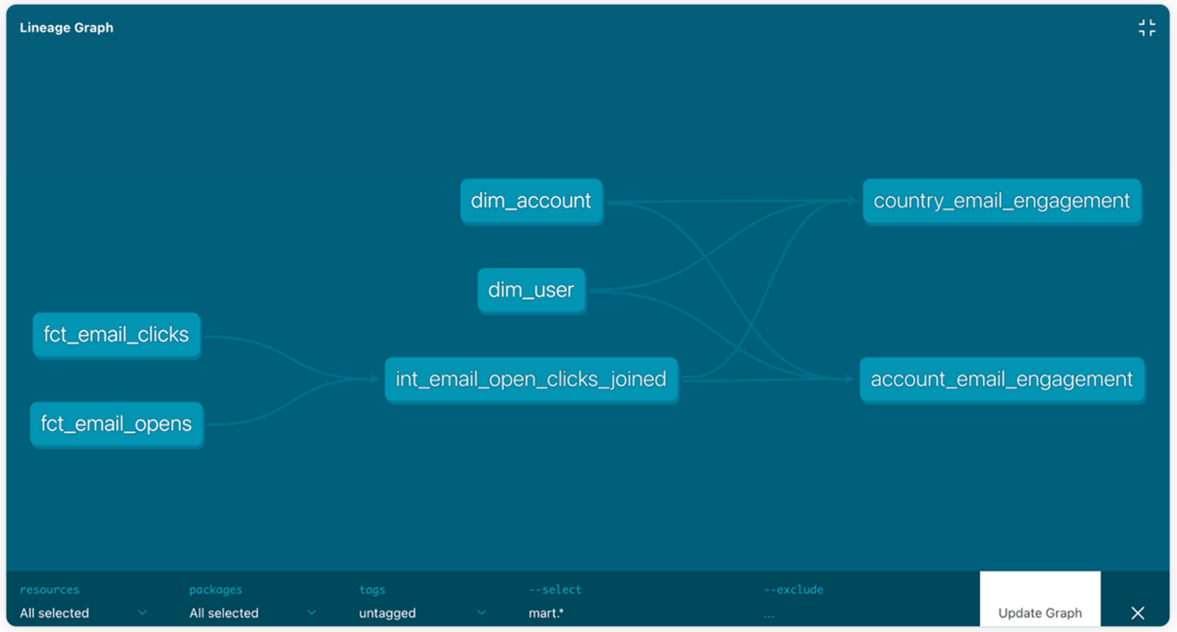
```
dbt run -select marketo_source
```

If any of the models fail due to missing tables, because you chose not to sync those tables in Fivetran, then in your source schema you can disable those models by updating the **`dbt_project.yml`** file.

For example if the email bounced and email template tables are missing from the Marketo source schema you can disable the models for those tables by adding the following under the models config:

```
models:
 marketo_source:
   +schema: your_staging_name
   tmp:
     stg_marketo__activity_email_bounced_tmp:
       +enabled: false
     stg_marketo__email_template_history_tmp:
       +enabled: false
   stg_marketo__activity_email_bounced:
     +enabled: false
   stg_marketo__email_template_history:
     +enabled: false
```

Developing the marketing analytics models



dbt lineage graph showing the star schema and aggregate tables data model

Now that the Fivetran packages have taken care of creating and testing the staging models you can begin to develop the data models for your marketing analytics use cases which will be a star schema data model along with materialized aggregate tables.

For example, for the first marketing analytics dashboard, you may want to see how engaged certain companies and sales regions are by the number of email campaigns they have opened and clicked.

To do so, you can join Salesforce and Marketo tables using the Salesforce user email, Salesforce `account_id` and Marketo `lead_id`.

The models will be structured under the mart folder in the following way.

```
marketing_analytics_demo
|-- dbt_project.yml
|-- packages.yml
|-- models
        |-- mart
                 |-- core
                 |-- intermediate
                 |-- marketing_analytics
```

You can view the code for all the models on Github in the /models/mart directory and below describes what is in each folder along with an example.

Core models

The core models are the facts and dimensions tables that will be used by all downstream models to build upon.

The dbt SQL code for the `dim_user` model:

```
with salesforce_users as (
   select
       account_id,
       email
   from {{ ref('stg_salesforce__user') }}
   where email is not null and account_id is not null
),
marketo_users as (
   select
       lead_id,
       email
   from {{ ref('stg_marketo__lead') }}
),
joined as (
   select
     lead_id,
     account_id
   from salesforce_users
     left join marketo_users
     on salesforce_users.email = marketo_users.email
)

select * from joined
```

You can also add documentation and tests for the models using a yaml file in the folder.

There are 2 simple tests in the `core.yml` file that have been added

```yaml
version: 2

models:
 - name: dim_account
   description: "The Account Dimension Table"
   columns:
     - name: account_id
       description: "Primary key"
       tests:
         - not_null
 - name: dim_user
   description: "The User Dimension Table"
   columns:
     - name: lead_id
       description: "Primary key"
       tests:
         - not_null
```

Intermediate models

Some of the final downstream models may rely on the same calculated metrics and so to avoid repeating SQL you can create intermediate models that can be reused.

The dbt SQL code for `int_email_open_clicks_joined` model:

```sql
with opens as (
        select *
        from {{ ref('fct_email_opens') }}
), clicks as (
        select *
        from {{ ref('fct_email_clicks') }}
), opens_clicks_joined as (

    select
      o.lead_id as lead_id,
      o.campaign_id as campaign_id,
      o.email_send_id as email_send_id,
      o.activity_timestamp as open_ts,
      c.activity_timestamp as click_ts
    from opens as o
      left join clicks as c
      on o.email_send_id = c.email_send_id
      and o.lead_id = c.lead_id

)

select * from opens_clicks_joined
```

Marketing Analytics models

These are the final marketing analytics models that will be used to power the dashboards and reports used by marketing and sales teams.

The dbt SQL code for `country_email_engagement` model:

```
with accounts as (
        select
        account_id,
        billing_country
        from {{ ref('dim_account') }}
), users as (
        select
        lead_id,
        account_id
        from {{ ref('dim_user') }}
), opens_clicks_joined as (

    select * from {{ ref('int_email_open_clicks_joined') }}

), joined as (

        select *
        from users as u
        left join accounts as a
        on u.account_id = a.account_id
        left join opens_clicks_joined as oc
        on u.lead_id = oc.lead_id

)

select
        billing_country as country,
        count(open_ts) as opens,
        count(click_ts) as clicks,
        count(click_ts) / count(open_ts) as click_ratio
from joined
group by country
```

Run and test dbt models

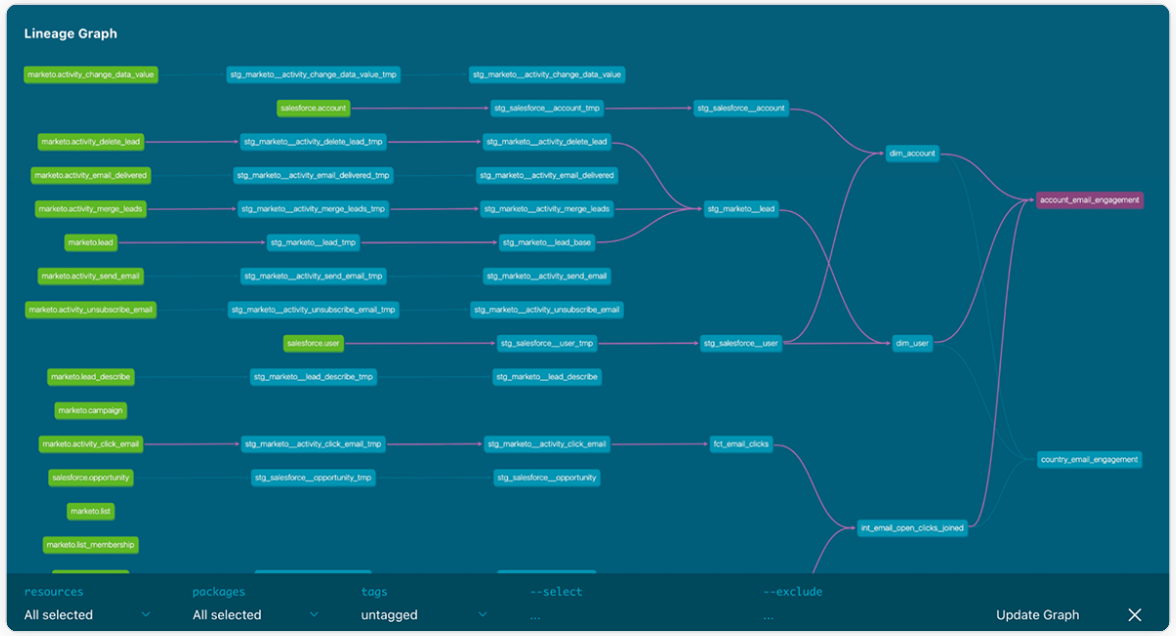
Now that your model is ready you can run all the models using

```
dbt run
```

And then run the tests using

```
dbt test
```

View the dbt docs and lineage graph


Lineage Graph
marketo.activity_change_data_value
stg_marketo__activity_change_data_value_tmp
stg_marketo__activity_change_data_value
salesforce.account
stg_salesforce__account_tmp
stg_salesforce__account
marketo.activity_delete_lead
stg_marketo__activity_delete_lead_tmp
stg_marketo__activity_delete_lead
dim_account
marketo.activity_email_delivered
stg_marketo__activity_email_delivered_tmp
stg_marketo__activity_email_delivered
account_email_engagement
marketo.activity_merge_leads
stg_marketo__activity_merge_leads_tmp
stg_marketo__activity_merge_leads
stg_marketo__lead
marketo.lead
stg_marketo__lead_tmp
stg_marketo__lead_base
marketo.activity_send_email
stg_marketo__activity_send_email_tmp
stg_marketo__activity_send_email
marketo.activity_unsubscribe_email
stg_marketo__activity_unsubscribe_email_tmp
stg_marketo__activity_unsubscribe_email
salesforce.user
stg_salesforce__user_tmp
stg_salesforce__user
dim_user
marketo.lead_describe
stg_marketo__lead_describe_tmp
stg_marketo__lead_describe
marketo.campaign
marketo.activity_click_email
stg_marketo__activity_click_email_tmp
stg_marketo__activity_click_email
fct_email_clicks
salesforce.opportunity
stg_salesforce__opportunity_tmp
stg_salesforce__opportunity
country_email_engagement
marketo.list
marketo.list_membership
int_email_open_clicks_joined
resources
All selected
packages
All selected
tags
untagged
--select
--exclude
Update Graph

dbt lineage graph for the marketing analytics model

Once your models have run successfully you can generate the docs and lineage graph using

```
$ dbt docs generate
```

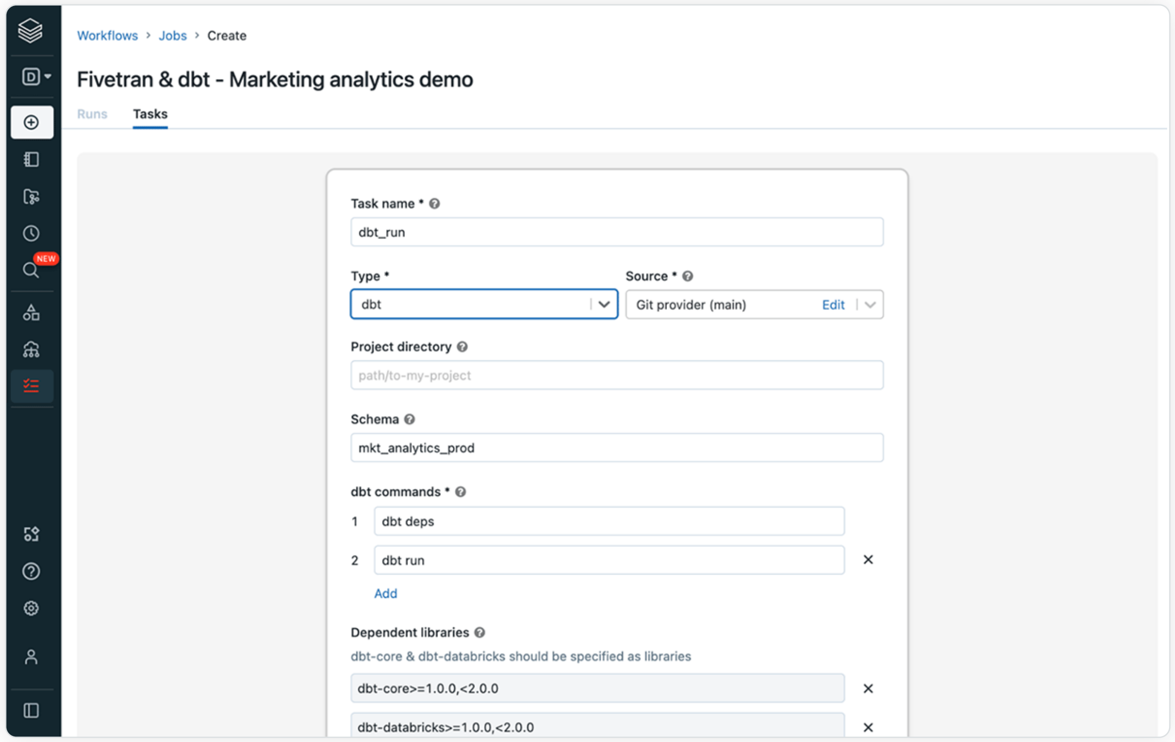
To then view them locally run

```
$ dbt docs serve
```

Deploying dbt models to production

Once you have developed and tested your dbt model locally you have multiple options for deploying into production one of which is the new dbt task type in Databricks Workflows (private preview).

Your dbt project should be managed and version controlled in a Git repository. You can create a dbt task in your Databricks Workflows job pointing to the Git repository.

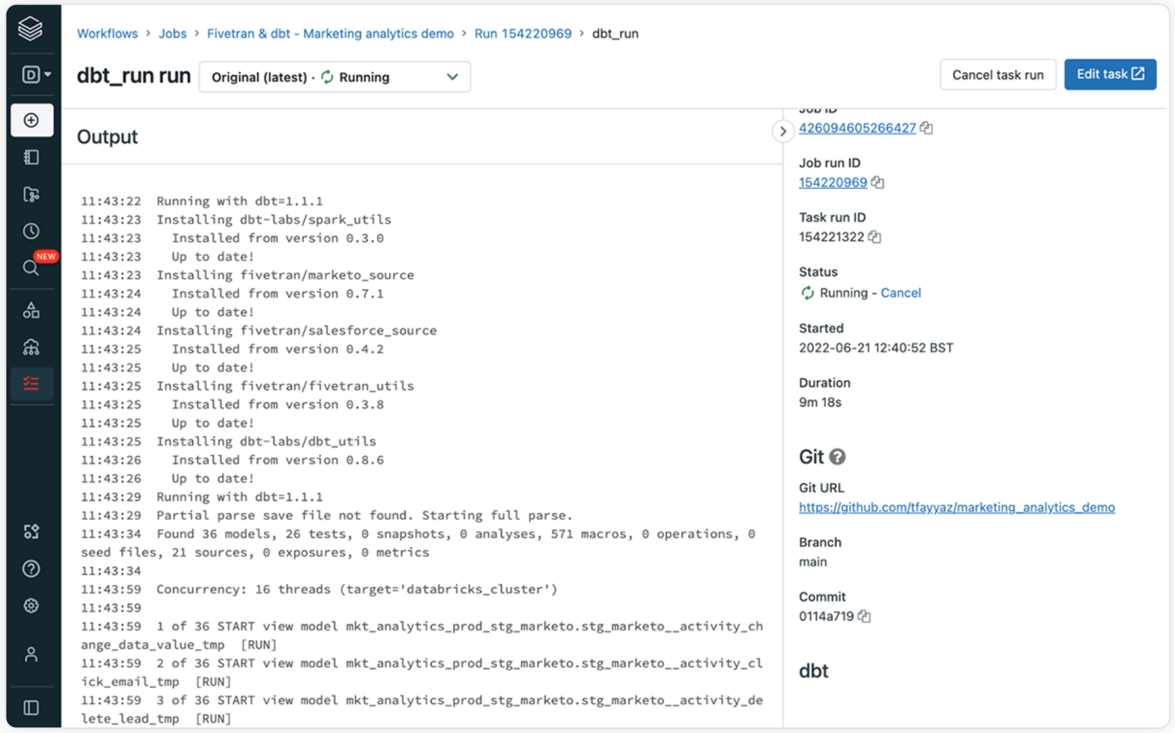


Fivetran & dbt - Marketing analytics demo

Task name * : dbt_run
Type * : dbt
Source * : Git provider (main) Edit
Project directory : path/to-my-project
Schema : mkt_analytics_prod
dbt commands * :
1 dbt deps
2 dbt run
Dependent libraries : dbt-core & dbt-databricks should be specified as libraries
dbt-core>=1.0.0,<2.0.0
dbt-databricks>=1.0.0,<2.0.0

Using a dbt task type in Databricks Workflows to orchestrate dbt

As you are using packages in your dbt project the first command should be dbt deps followed by dbt run for the first task and then dbt test for the next task.


Workflows › Jobs › Fivetran & dbt - Marketing analytics demo › Run 154220969 › dbt_run
Original (latest) · Running
Cancel task run
Edit task
NEW

dbt_run run

Output

```
11:43:22  Running with dbt=1.1.1
11:43:23  Installing dbt-labs/spark_utils
11:43:23    Installed from version 0.3.0
11:43:23    Up to date!
11:43:23  Installing fivetran/marketo_source
11:43:24    Installed from version 0.7.1
11:43:24    Up to date!
11:43:24  Installing fivetran/salesforce_source
11:43:25    Installed from version 0.4.2
11:43:25    Up to date!
11:43:25  Installing fivetran/fivetran_utils
11:43:25    Installed from version 0.3.8
11:43:25    Up to date!
11:43:25  Installing dbt-labs/dbt_utils
11:43:26    Installed from version 0.8.6
11:43:26    Up to date!
11:43:29  Running with dbt=1.1.1
11:43:29  Partial parse save file not found. Starting full parse.
11:43:34  Found 36 models, 26 tests, 0 snapshots, 0 analyses, 571 macros, 0 operations, 0
seed files, 21 sources, 0 exposures, 0 metrics
11:43:34
11:43:59  Concurrency: 16 threads (target='databricks_cluster')
11:43:59
11:43:59  1 of 36 START view model mkt_analytics_prod_stg_marketo.stg_marketo__activity_ch
ange_data_value_tmp  [RUN]
11:43:59  2 of 36 START view model mkt_analytics_prod_stg_marketo.stg_marketo__activity_cl
ick_email_tmp  [RUN]
11:43:59  3 of 36 START view model mkt_analytics_prod_stg_marketo.stg_marketo__activity_de
lete_lead_tmp  [RUN]
```

Job ID
426094605266427
Job run ID
154220969
Task run ID
154221322
Status
Running - Cancel
Started
2022-06-21 12:40:52 BST
Duration
9m 18s

Git
Git URL
https://github.com/tfayyaz/marketing_analytics_demo
Branch
main
Commit
0114a719

dbt

Viewing the dbt logs for each dbt run

For each run you can see the logs for each dbt command helping you debug and fix any issues.

Powering your marketing analytics with Fivetran and dbt

As shown here, using Fivetran and dbt alongside the Databricks Data Intelligence Platform allows you to easily build a powerful marketing analytics solution that is easy to set up, manage and flexible enough to suit any of your data modeling requirements.

To get started with building your own solution, visit the documentation for integrating Fivetran and dbt with Databricks and re-use the marketing_analytics_demo project example to quickly get started.

The dbt task type in Databricks Workflows is in private preview. To try the dbt task type, please reach out to your Databricks account executive.

SECTION 5.2

Claims Automation on Databricks

Smart claims increases efficiency by automating all aspects of claims processing from ingestion, analysis, and decision-making.

by Anindita Mahapatra and Marzi Rasooli

According to the latest reports from global consultancy EY, the future of insurance will become increasingly **data-driven**, and **analytics enabled**. The recent focus on the cloud has improved access to advanced technological infrastructure, but most organizations still need help implementing and leveraging these capabilities. It's time to shift the focus on operationalizing services to realize value.

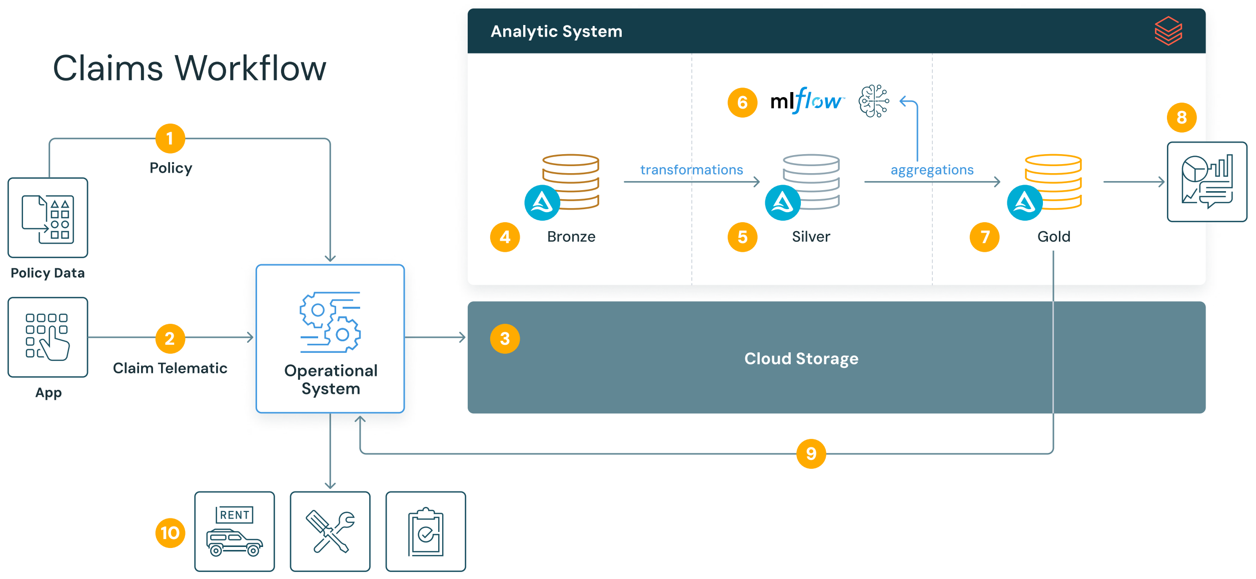
In today's economic circumstances, insurance companies face an ever-increasing number of challenges. Insurers are being forced to leverage data to their advantage and **innovate** at an accelerated pace. For personal P&C insurers, this means an increased focus on **personalization** and customer retention. Brand loyalty is at an all-time low, with customers continuously shopping for more competitive rates and better overall experiences, which increases the risk of churn. An increase in fraudulent claims further erodes profit margins. Insurers need to find additional ways to reduce costs and better manage risks.

Automating and optimizing the claims-handling process is one area that can significantly reduce costs through time saved and lesser reliance on human capital. Furthermore, effectively leveraging insights from data and advanced analytics can substantially reduce the overall exposure to risk.

The motivation behind the "Smart Claims" solution accelerator is simple — improve the claims handling process to enable faster settlement, lower processing costs, and deliver quicker insights about potentially fraudulent claims, with the lakehouse. Implementing the lakehouse paradigm simplifies the current architectural landscape and sets the scene for future expansion across the organization. The accompanying assets can be found here.

Reference Architecture and Workflow

A typical claims workflow involves some level of orchestration between operational systems such as Guidewire and analytical systems like Databricks. The diagram below shows an example of such a workflow for an automotive insurer.



Smart Claims Reference Architecture and Workflow

Automating and optimizing the claims handling process requires a deep understanding of customer interaction with operational systems and the various sources of information available for analysis.

In this example, we assume that customers primarily interact through a mobile application, and from there, they can submit claims and monitor the status of existing cases. This touch point offers vital information about customer behavior. Another important source of information is IoT devices installed in customer vehicles. Telematics data can be streamed to the operational and analytical systems, providing valuable insights into customer-driving behavior and patterns. Other external data sources may include weather and road conditions data that supplement the traditional data categories such as vehicle characteristics (make, model, year), driver characteristics and exposure/coverage (limits, deductibles)

Access to additional data sources can become increasingly important, especially in the absence of data from traditional sources such as credit bureaus. Credit scores from bureaus usually form the basis for risk modeling, assessing the exposure for drivers, which ultimately impacts their premiums. On the other hand, data from mobile applications and IoT devices provide a more personalized view of customer behavior, which could be used to create a more accurate indicator of the risk associated with a given party. This alternative, **behavioral-based** approach to risk modeling and pricing is essential for delivering a hyper-personalized customer experience.

The lakehouse architecture powered by the Databricks Data Intelligence Platform is the only platform that combines all the required features and services to support a future-proof claims-handling process. From streaming to machine learning and reporting, Databricks offers the best platform to build an end-to-end solution for the insurance industry of tomorrow.

The following steps capture the overall flow:

- Policy data is ingested.
- Telematics data is continuously ingested from IoT sensors. A claimant submits claims data via a mobile app.
- All the operational data is ingested into cloud storage.
- This is incrementally loaded as 'Raw data' into Delta Bronze tables
- The data is wrangled and refined via various data transformations
- Data is scored using the trained model
- The predictions are loaded to a Gold table
- The Claims Dashboard is refreshed for visualization
- The resulting insights are fed back to the operational system. This provides the feedback loop of pulling data from Guidewire and passing 'Next Best Action' back to Guidewire in real time to know which claims should be prioritized.
- The Claims Decisioning workflows use these generated insights to route the case appropriately. (Eg. approve repair expenses, rental reimbursement, or alert authorities)

How the lakehouse paradigm aids Smart Claims

The lakehouse architecture enables all data personas (data engineers, data scientists, analytic engineers, and BI analysts) to work collaboratively on a single platform. Supporting all big-data workloads and paradigms (e.g., batch processing, streaming, DataOps, ML, MLOps, and BI) in a single, collaborative platform greatly simplifies the overall architecture, improves stability, and reduces cost significantly.

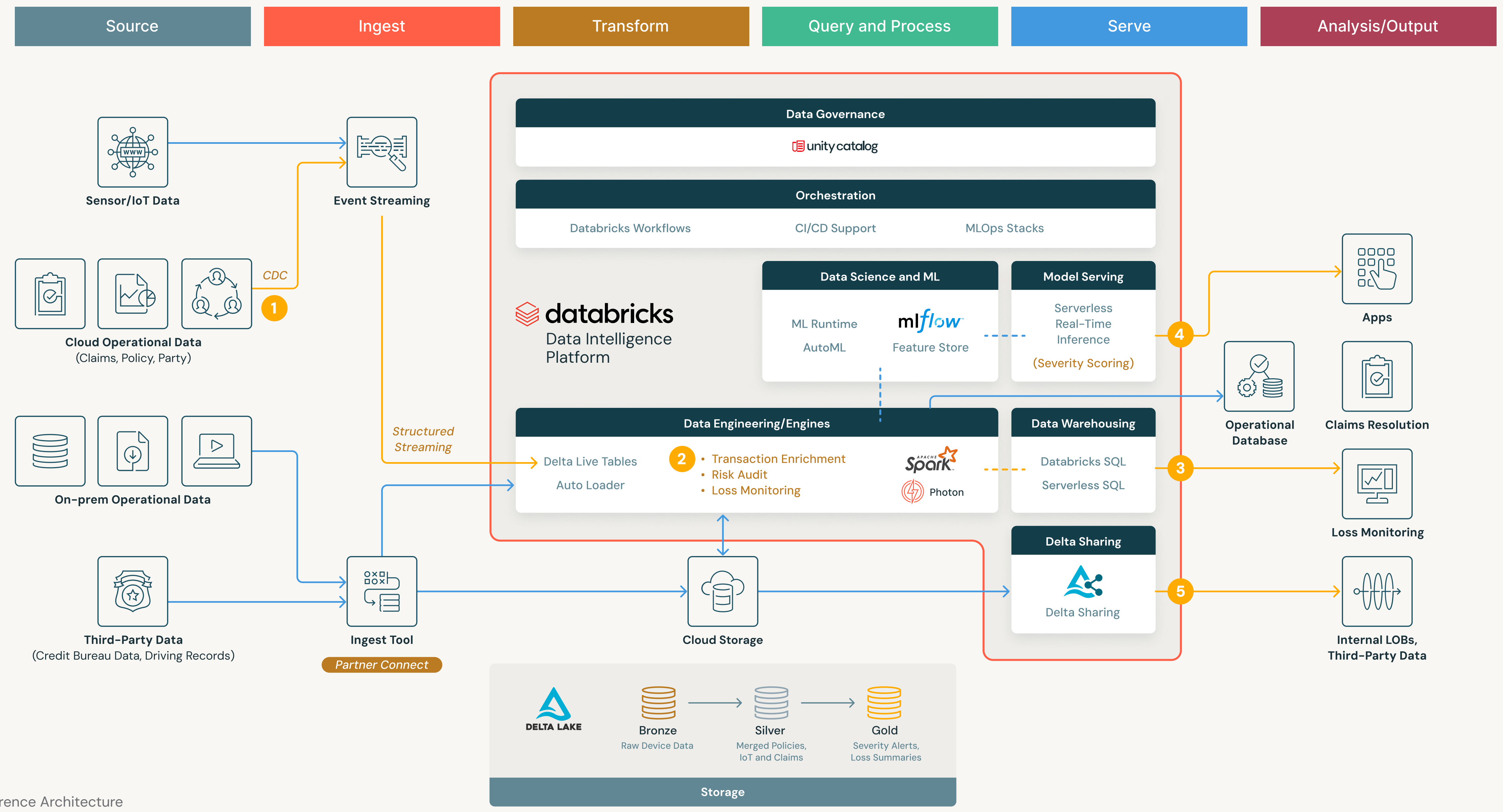
Databricks Delta Live Tables (DLT) pipelines offer a simple, declarative framework to develop and implement workloads quickly. It also provides native support for data quality management with granular constraints to guarantee the integrity of outputs.

ML and AI workloads can easily be created and managed with MLflow for reproducibility and auditability. MLFlow simplifies the entire model lifecycle, from experimenting through model deployment, serving, and archiving. ML can be run on all types of data including unstructured data beyond text (images, audio, video, etc). In this solution, we will use computer vision capabilities to assess damage to the vehicle.

Finally, Databricks SQL provides a fast and efficient engine to query curated and aggregated data. These insights can then be packaged and served through interactive Dashboards within minutes.

Unity Catalog provides a multi-cloud, centralized governance solution for all data and AI assets including files, tables, machine learning models and dashboards with built-in search, discovery, automated workload lineage.

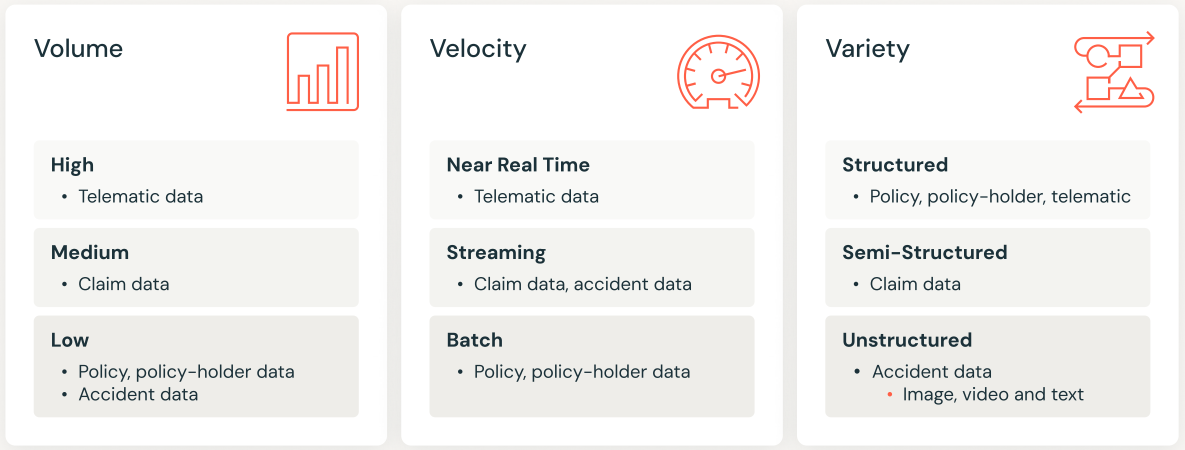
The diagram below shows a reference architecture for the lakehouse in the context of typical insurance use cases:



Insurance Reference Architecture

Data Ingestion using DLT and Multitask Workflows

Automating the claims-handling process starts with optimizing the ingestion and data engineering workflow. The figure below offers a summary of the typical data sources encountered including structured, semi-structured and unstructured. Some sources are slower-moving, while others update more rapidly. Additionally, some sources might be additive, requiring appending, while others offer incremental updates and must be treated as slow-changing dimensions.



Volume	Velocity	Variety
High • Telematic data	**Near Real Time** • Telematic data	**Structured** • Policy, policy-holder, telematic
Medium • Claim data	**Streaming** • Claim data, accident data	**Semi-Structured** • Claim data
Low • Policy, policy-holder data • Accident data	**Batch** • Policy, policy-holder data	**Unstructured** • Accident data • Image, video and text

Sample Datasets used in Claims Processing

DLT can simplify and operationalize the data processing pipeline. The framework offers support for Auto Loader to facilitate ingestion from streaming sources, efficient auto-scaling to handle sudden changes in data volumes, and resiliency via a restart of task failure.

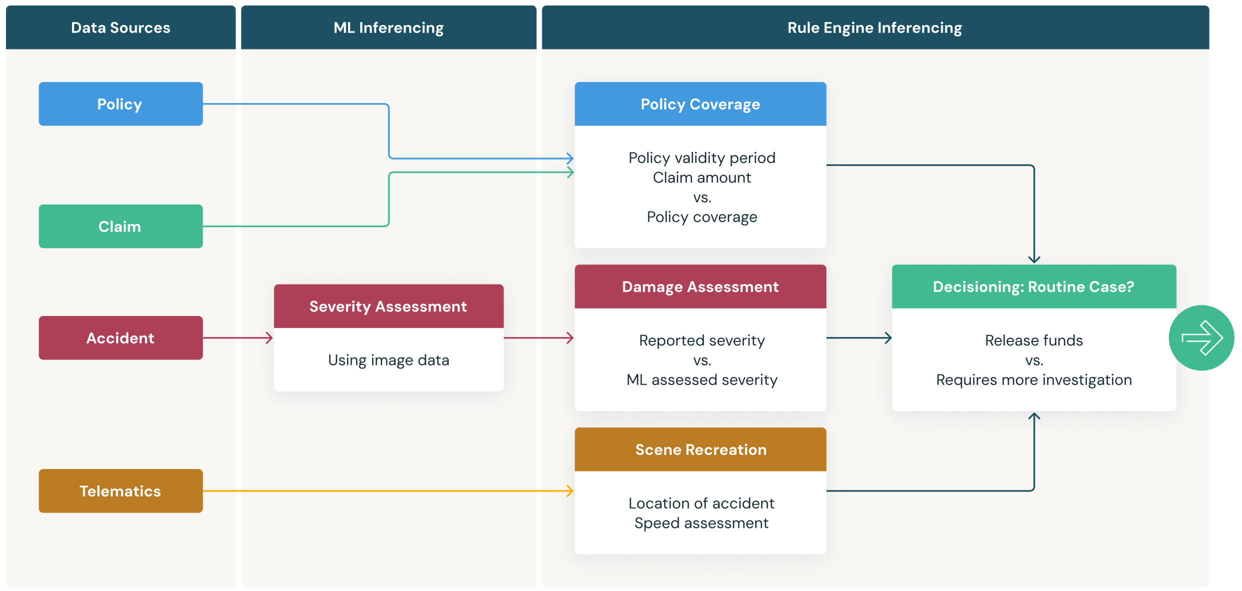
Databricks Workflows can accommodate multiple tasks and workloads (e.g., notebooks, DLT, ML, SQL). Workflows support repair-and-run and compute sharing across tasks – enabling robust, scalable, cost-effective workloads. Additionally, Workflow can easily be automated through schedules or programmatic invoking via REST APIs.

Insight Generation using ML and Dynamic Rules Engine

Leveraging ML is essential to uncovering previously unknown patterns, highlighting new insights, and flagging suspicious activity. However, combining ML and traditional rules-based approaches can be even more powerful.

Within the claims-handling process, ML can be used for several use cases. One example would be using computer vision and ML for assessing and scoring images submitted with vehicle insurance claims. Models can be trained to focus on the validity and severity of damages. Here, MLFlow can be crucial in simplifying the model training and serving process with its end-to-end MLOps capabilities. MLFlow offers a serverless model serving through REST APIs. Trained models can be operationalized and put into production with the click of a button.

On the other hand, rules engines offer flexible ways of defining known operational characteristics and statistical checks, which can be automated and applied without requiring human interaction. Flags are raised whenever data does not comply with preset expectations and are sent for human review and investigation. Incorporating such an approach with ML-based workflows offers additional oversight and significantly reduces the time claims investigators require to dissect and review flagged cases.

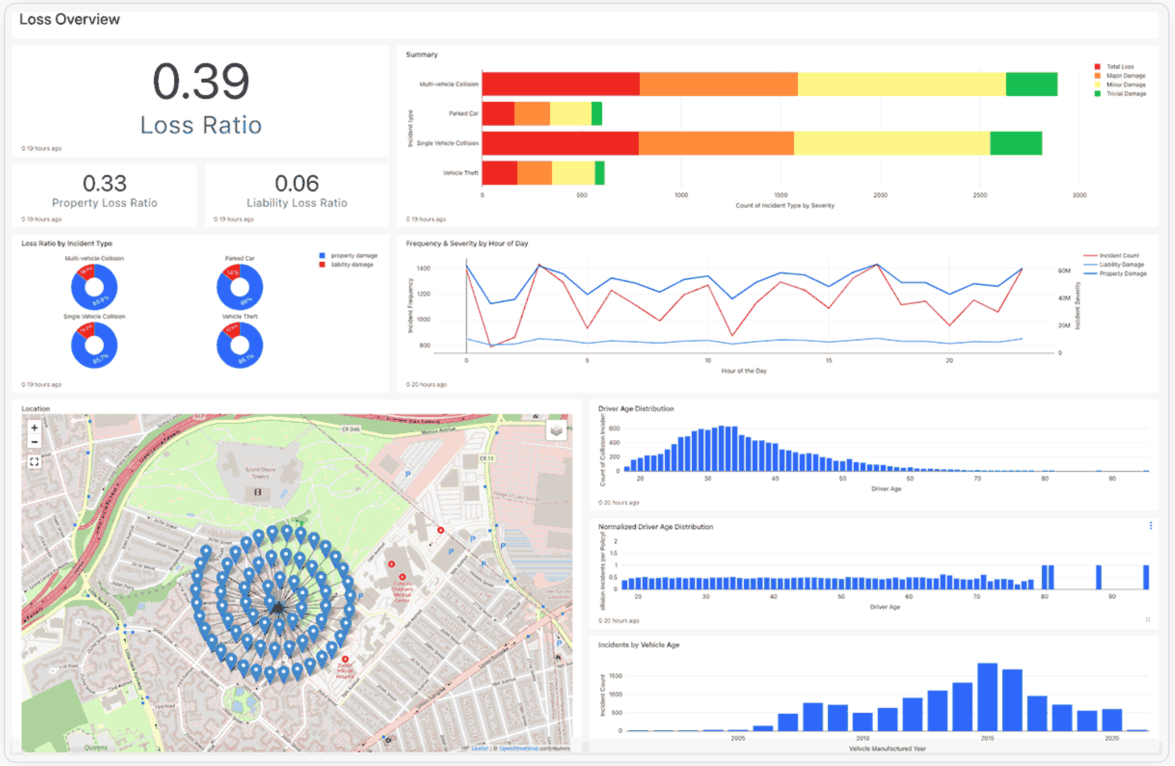


ML and Rule Engine Inferencing

Insight visualization using Dashboards

In this example, we created two dashboards to capture critical business insights. The dashboards include the following:

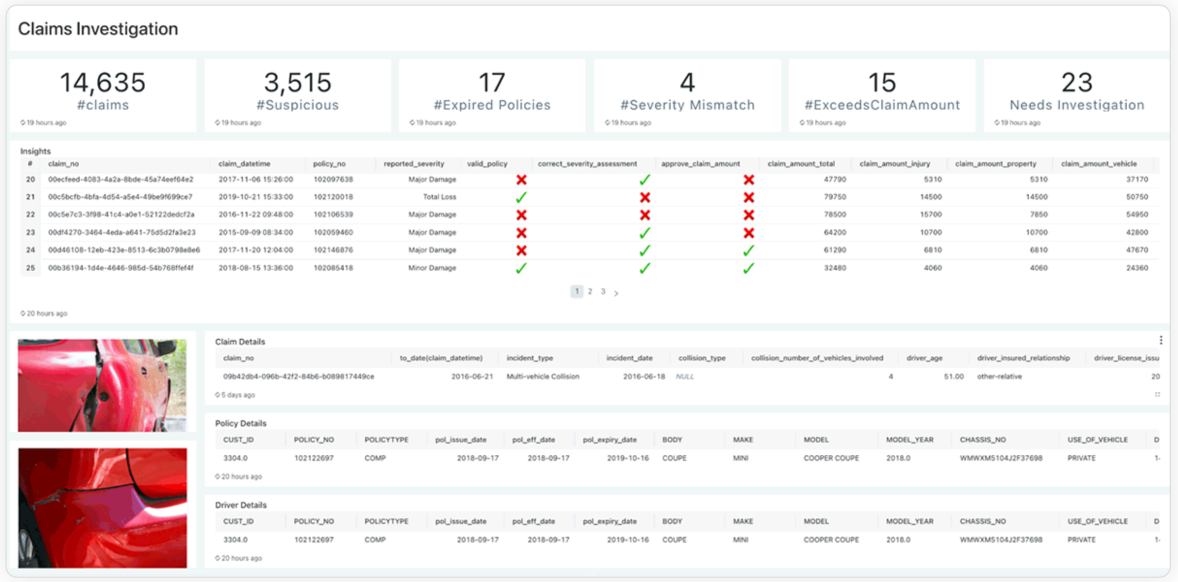
- A **Loss Summary** dashboard for a high-level view of the overall business operations; and
- A **Claims Investigation** dashboard with a granular view of claims details to understand the specifics of a given case.



Analyzing recent trends can further aid in reviewing similar cases such as :

- Loss Ratio is computed by insurance claims paid plus adjustment expenses divided by total earned premiums. E.g. typical average Loss Ratio (all coverages combined, Bodily Injury, and Physical Damage) for personal auto should be around 65%
- Summary visualization captures count of incident type by damage severity
- Trend lines over various features/dimensions
- Geographic distribution of policies

The Claims Investigation dashboard facilitates faster investigation by providing all relevant information around a claim allowing the human investigator to drill down to a specific claim to see details such as images of the damaged vehicle, Claim, Policy and Driver details, Telematic data draws the path taken by the vehicle, Reported data is contrasted with assessed data insights.



Claims Investigation

14,635	3,515	17	4	15	23
#claims	#Suspicious	#Expired Policies	#Severity Mismatch	#ExceedsClaimAmount	Needs Investigation
19 hours ago	19 hours ago	19 hours ago	19 hours ago	19 hours ago	19 hours ago

Insights

#	claim_no	claim_datetime	policy_no	reported_severity	valid_policy	correct_severity_assessment	approve_claim_amount	claim_amount_total	claim_amount_injury	claim_amount_property	claim_amount_vehicle
20	00ecfeed-4083-4a2a-8bde-45a74eef64e2	2017-11-06 15:26:00	102097638	Major Damage	✗	✓	✗	47790	5310	5310	37170
21	00c5bcfb-4bfa-4d54-a5e4-49be9f699ce7	2019-10-21 15:33:00	102120018	Total Loss	✓	✗	✗	79750	14500	14500	50750
22	00c5e7c3-3f98-41c4-a0e1-52122dedcf2a	2016-11-22 09:48:00	102106539	Major Damage	✗	✗	✗	78500	15700	7850	54950
23	00df4270-3464-4eda-a641-75d5d2fa3e23	2015-09-09 08:34:00	102059460	Major Damage	✗	✓	✗	64200	10700	10700	42800
24	00d46108-12eb-423e-8513-6c3b0798e8e6	2017-11-20 12:04:00	102146876	Major Damage	✗	✓	✓	61290	6810	6810	47670
25	00b36194-1d4e-4646-985d-54b768ffef4f	2018-08-15 13:36:00	102085418	Minor Damage	✓	✓	✓	32480	4060	4060	24360

1 2 3 >

20 hours ago

Claim Details

claim_no	to_date(claim_datetime)	incident_type	incident_date	collision_type	collision_number_of_vehicles_involved	driver_age	driver_insured_relationship	driver_license_issu
09b42db4-096b-42f2-84b6-b089817449ce	2016-06-21	Multi-vehicle Collision	2016-06-18	NULL	4	51.00	other-relative	20

5 days ago

Policy Details

CUST_ID	POLICY_NO	POLICYTYPE	pol_issue_date	pol_eff_date	pol_expiry_date	BODY	MAKE	MODEL	MODEL_YEAR	CHASSIS_NO	USE_OF_VEHICLE	D
3304.0	102122697	COMP	2018-09-17	2018-09-17	2019-10-16	COUPE	MINI	COOPER COUPE	2018.0	WMWXM5104J2F37698	PRIVATE	1-

20 hours ago

Driver Details

CUST_ID	POLICY_NO	POLICYTYPE	pol_issue_date	pol_eff_date	pol_expiry_date	BODY	MAKE	MODEL	MODEL_YEAR	CHASSIS_NO	USE_OF_VEHICLE	D
3304.0	102122697	COMP	2018-09-17	2018-09-17	2019-10-16	COUPE	MINI	COOPER COUPE	2018.0	WMWXM5104J2F37698	PRIVATE	1-

20 hours ago

Claim Investigation Dashboard

Provides recent claims that are auto-scored in the pipeline using ML inferencing and rule engine

- A green tick is used to denote auto-assessment matches claims description
- A red cross indicates a mismatch that warrants further manual investigation

Summary

Innovation and personalization are essential for insurance firms to differentiate themselves from the competition. Databricks provides a data intelligence platform for insurers to enable and accelerate innovation with an open, secure, extensible architecture that easily integrates with third-party tools and services. This Solution Accelerator demonstrates how the paradigm can be applied to claims handling. Further, the Databricks ecosystem offers a range of capabilities to enable data teams and business stakeholders to collaborate and generate and share insights that support business decisions and drive tangible value to the bottom line.

The technical assets, including pipeline configurations, models, and sample data used in this example, can be accessed here or directly on Git.

SECTION 5.3

Design Patterns for Batch Processing in Financial Services

Laying the foundation for automating workflows

by Eon Retief

Introduction

Financial services institutions (FSIs) around the world are facing unprecedented challenges ranging from market volatility and political uncertainty to changing legislation and regulations. Businesses are forced to accelerate digital transformation programs; automating critical processes to reduce operating costs and improve response times. However, with data typically scattered across multiple systems, accessing the information required to execute on these initiatives tends to be easier said than done.

Architecting an ecosystem of services able to support the plethora of data-driven use cases in this digitally transformed business can, however, seem to be an impossible task. This blog will focus on one crucial aspect of the modern data stack: batch processing. A seemingly outdated paradigm, we'll see why batch processing remains a vital and highly viable component of the data architecture. And we'll see how Databricks can help FSIs navigate some of the crucial challenges faced when building infrastructure to support these scheduled or periodic workflows.

Why batch ingestion matters

Over the last two decades, the global shift towards an instant society has forced organizations to rethink the operating and engagement model. A digital-first strategy is no longer optional but vital for survival. Customer needs and demands are changing and evolving faster than ever. This desire for instant gratification is driving an increased focus on building capabilities that support real-time processing and decisioning. One might ask whether batch processing is still relevant in this new dynamic world.

While real-time systems and streaming services can help FSIs remain agile in addressing the volatile market conditions at the edge, they do not typically meet the requirements of back-office functions. Most business decisions are not reactive but rather, require considered, strategic reasoning. By definition, this approach requires a systematic review of aggregate data collected over a period of time. Batch processing in this context still provides the most efficient and cost-effective method for processing large, aggregate volumes of data. Additionally, batch processing can be done offline, reducing operating costs and providing greater control over the end-to-end process.

The world of finance is changing, but across the board incumbents and startups continue to rely heavily on batch processing to power core business functions. Whether for reporting and risk management or anomaly detection and surveillance, FSIs require batch processing to reduce human error, increase the speed of delivery, and reduce operating costs.