CTRL

Johan Strand

Google Analytics 4 + BigQuery + Dataform

Agenda

→ “Challenges” with Google Analytics 4

→ Benefits with Google Analytics 4 + BigQuery

→ Dataform to the rescue

→ The Data pipeline

Johan Strand 🇸🇪

Senior Digital Analyst @ Ctrl Digital

johan.strand@ctrldigital.com

Experiences

Resurs

APOTEK

Boozt

MEASURE CAMP MALMÖ

CTRL

“Challenges” with GA4

Challenges with reports in Google Analytics 4

GA4 logics

Limited to GA4 logics and reports

Preset attribution

Google decided on the attribution models and they are black box-ish

Data not exact

Estimates, Cardinality and HyperLogLog

Data from one source

Hard to see the entire customer journey

Data is what it is

Limited option to modify data, no for historic

Acquisition reports are shaky

How to handle multiple traffic sources in a session?

CTRL

Benefits (and challenges) with GA4 + BigQuery

Benefits of using BigQuery for GA4 reporting

Secured data retention

We own and control the data

No need for Custom Dimensions

No administration for new parameters

100% Exact numbers

No estimates

Enrich with data sources

We can bring in extra data sources to show a holistic picture

Modify historical data

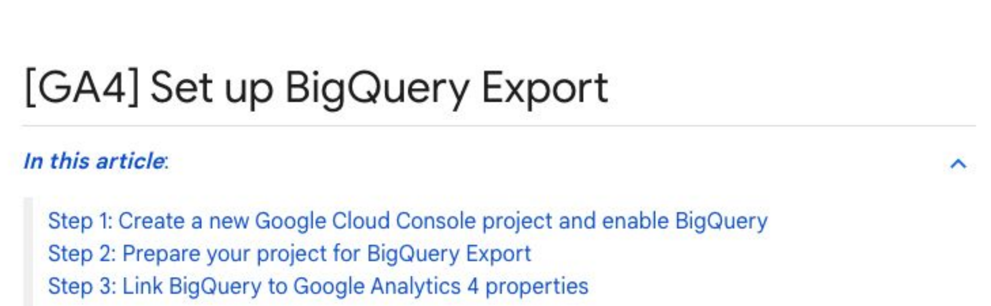
We can apply changes to historical data

Transparent attribution

We can 100% build and verify our own attribution and conversion models

You have set up the GA4 export to BigQuery…

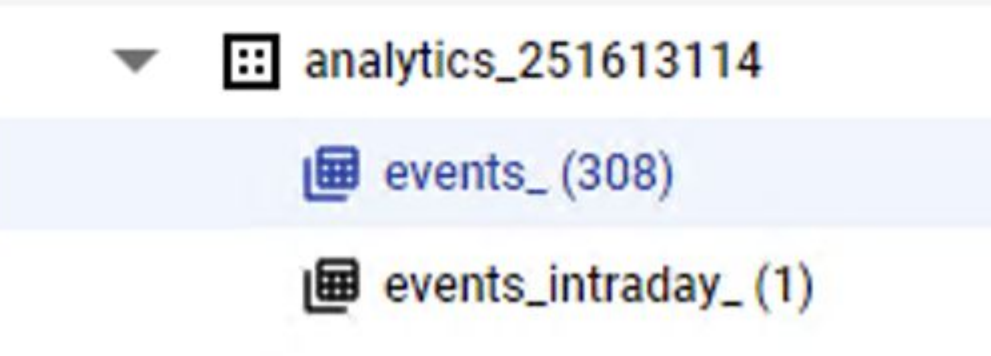
Now what?



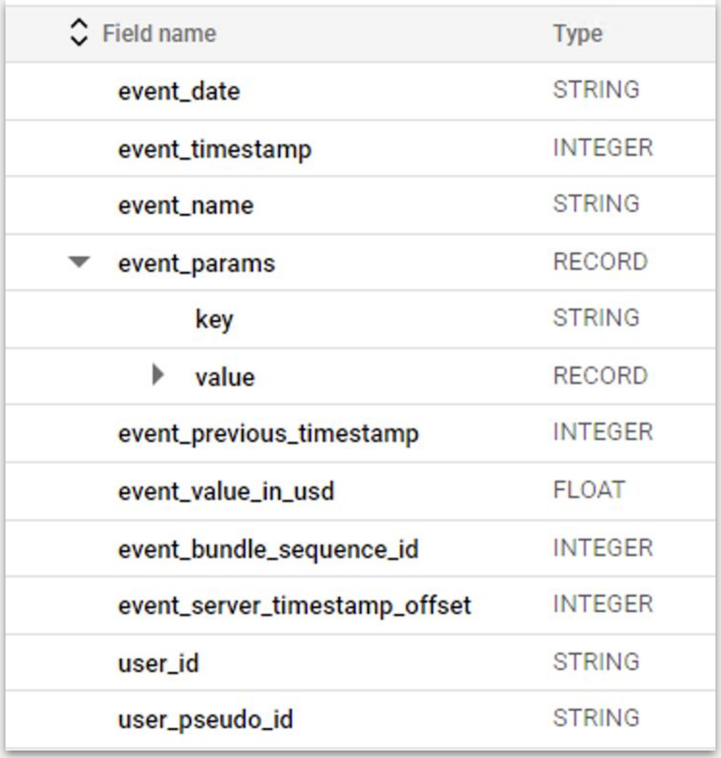
[GA4] Set up BigQuery Export

In this article:

Step 1: Create a new Google Cloud Console project and enable BigQuery
Step 2: Prepare your project for BigQuery Export
Step 3: Link BigQuery to Google Analytics 4 properties



analytics_251613114
events_ (308)
events_intraday_ (1)

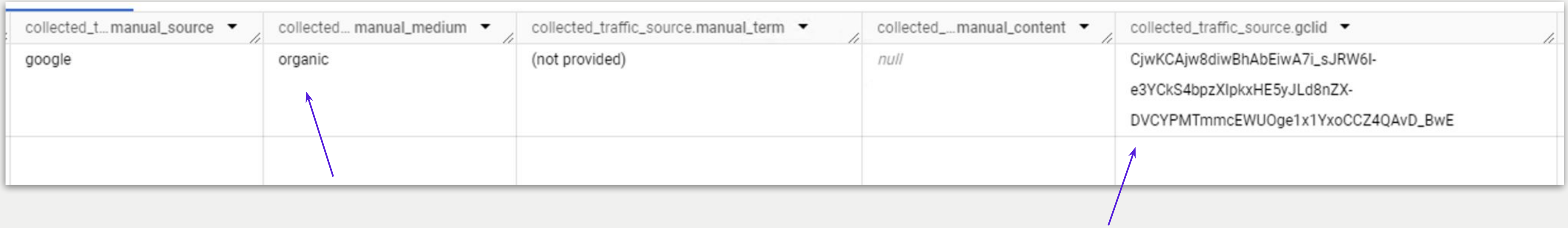


Field name	Type
event_date	STRING
event_timestamp	INTEGER
event_name	STRING
event_params	RECORD
key	STRING
value	RECORD
event_previous_timestamp	INTEGER
event_value_in_usd	FLOAT
event_bundle_sequence_id	INTEGER
event_server_timestamp_offset	INTEGER
user_id	STRING
user_pseudo_id	STRING

Challenges of using BigQuery for GA4 reporting

- Session data is incomplete
- Inaccurate traffic source data
- Nested data is complex to work with
- Already exported data can be retroactively updated
- …

We can´t even report on Google Ads..



collected_t...manual_source	collected... manual_medium	collected_traffic_source.manual_term	collected_...manual_content	collected_traffic_source.gclid
google	organic	(not provided)	*null*	CjwKCAjw8diwBhAbEiwA7i_sJRW6I-e3YCkS4bpzXIpkxHE5yJLd8nZX-DVCYPMTmmcEWUOge1x1YxoCCZ4QAvD_BwE

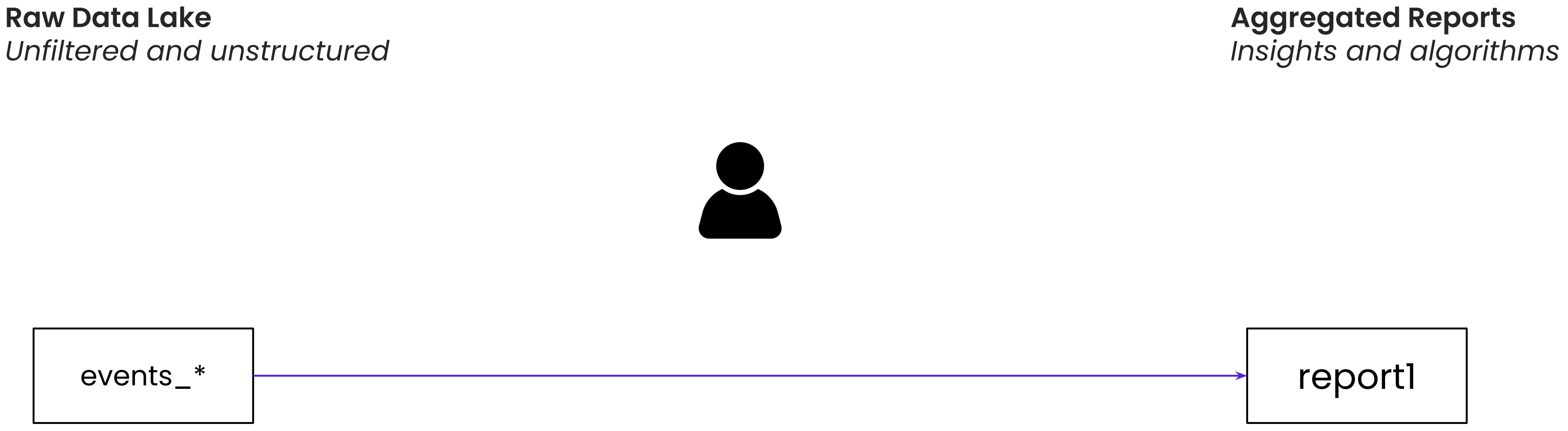
CTRL

Bonus problem

Data governance is complex

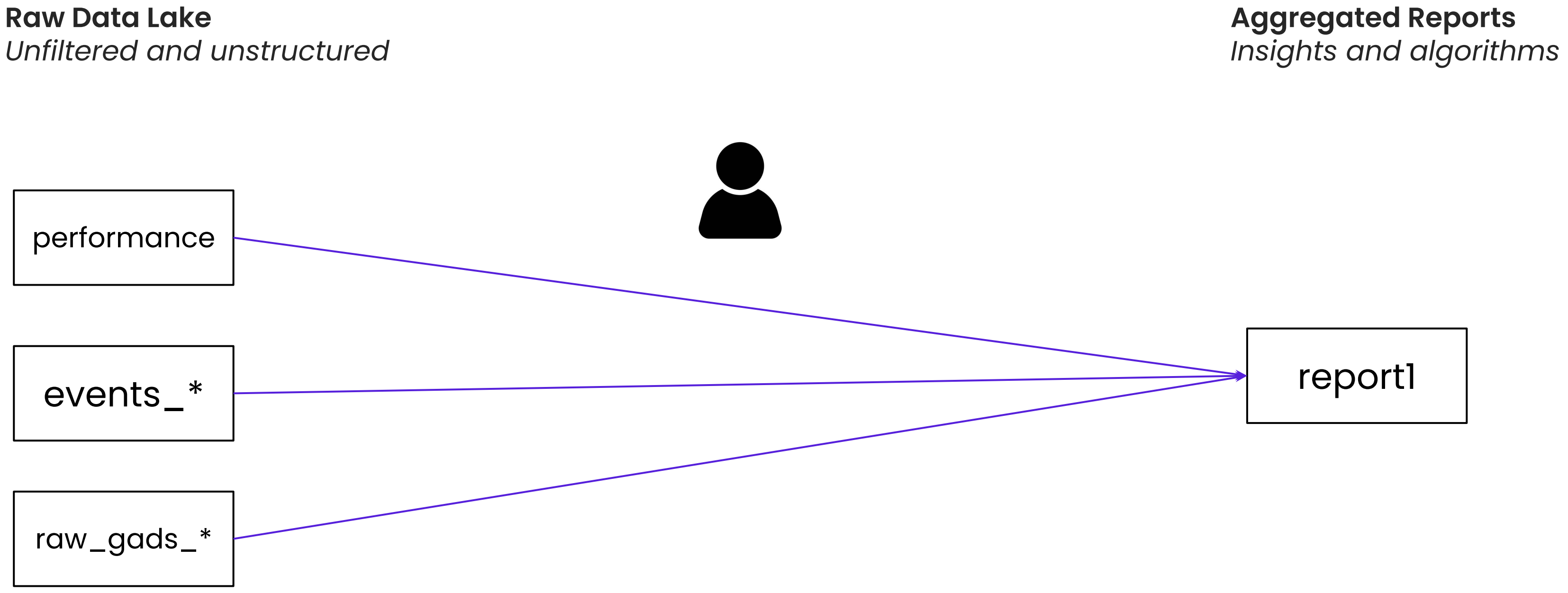
Working with SQL for reporting and aggregation

The good



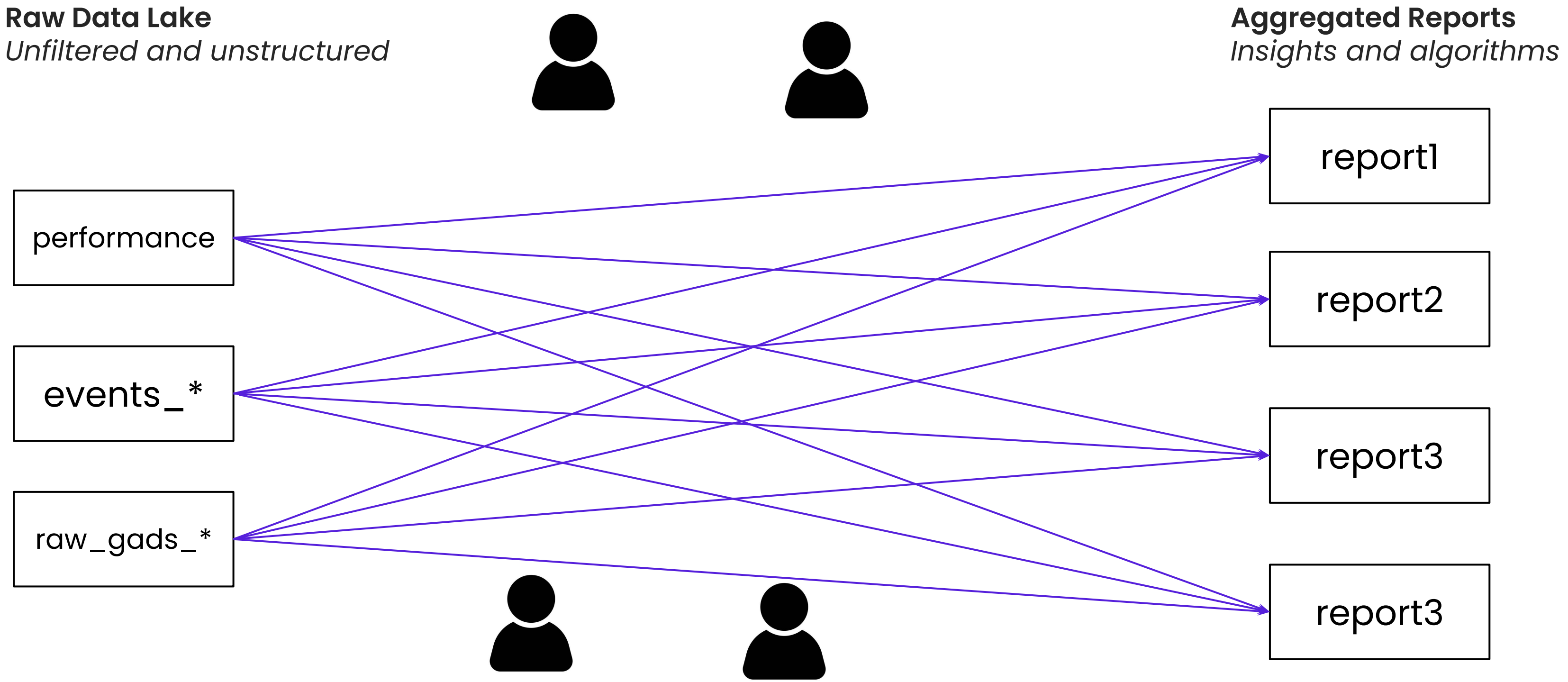
Working with SQL for reporting and aggregation

The bad



Working with SQL for reporting and aggregation

The ugly



Challenges of large SQL projects

- Multiple queries and data sources that have dependencies
- Maintaining standards and definitions
- Data quality checks
- Workspace and change management
- Lack of version control
- Documentation of data and logics

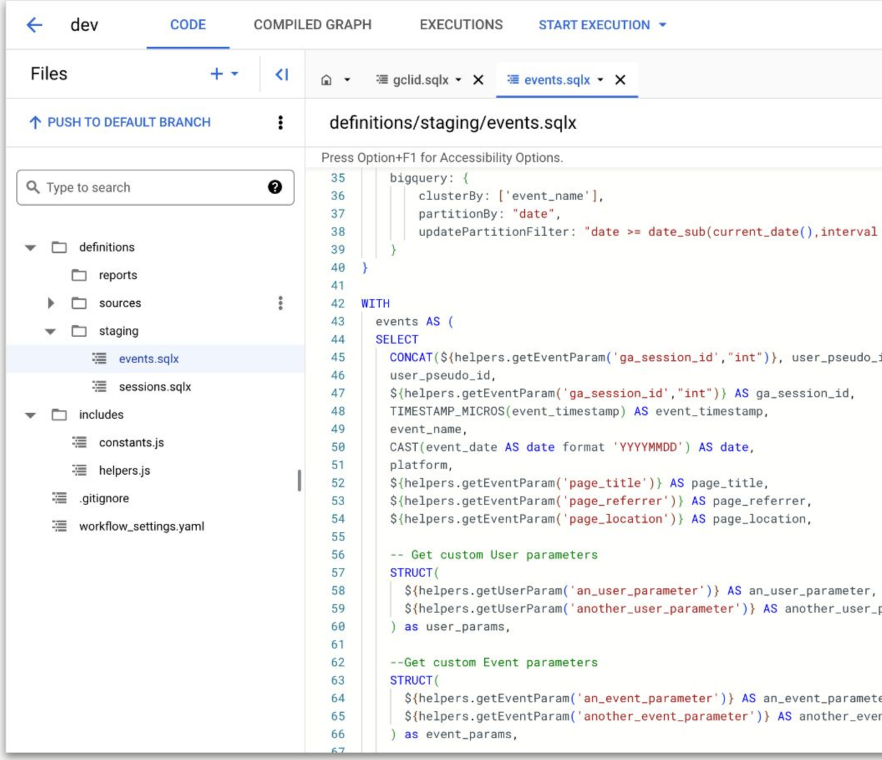
CTRL

Dataform to the rescue

What is Dataform?

Simplify your data processing architecture

- Bought by Google in 2020
- Now integrated into BigQuery
- Helps with data orchestration
- Manage complex workflows
- Free service*



definitions/staging/events.sqlx

Press Option+F1 for Accessibility Options.

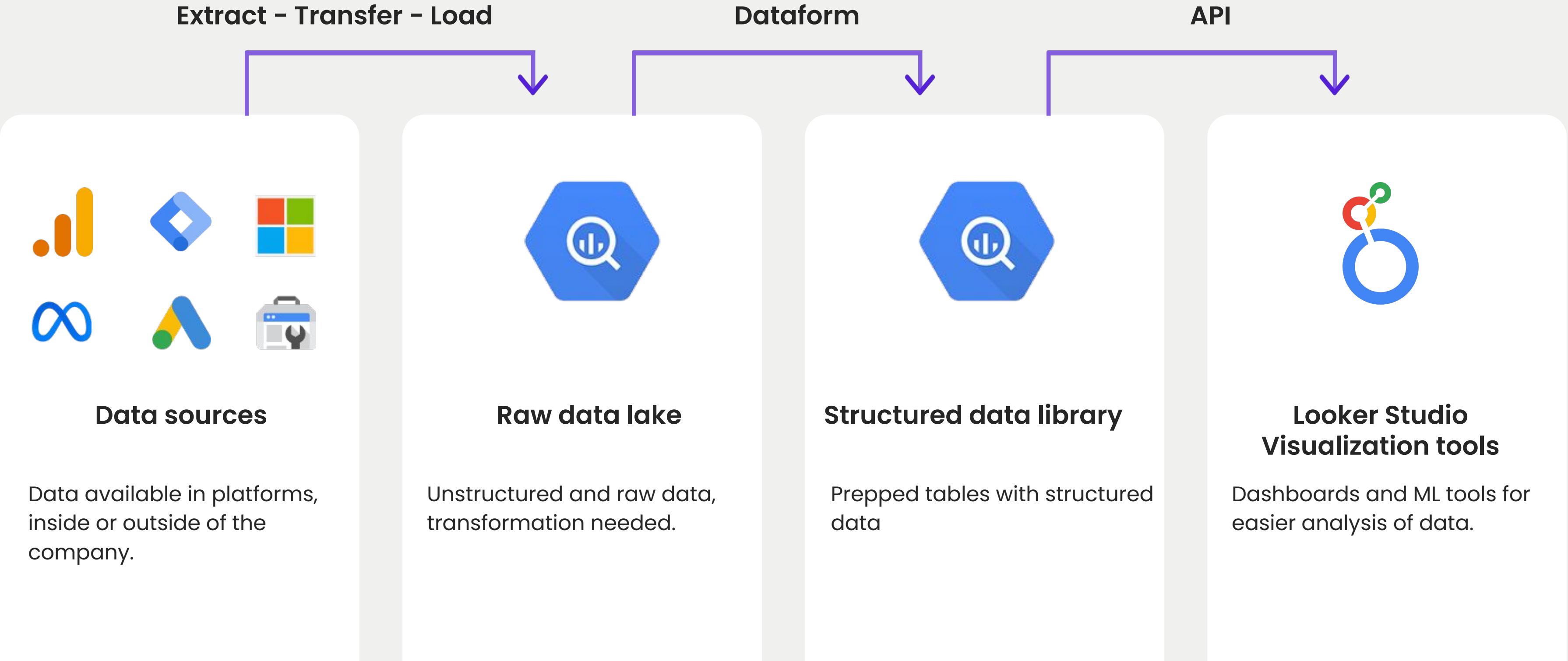
```
    bigquery: {
        clusterBy: ['event_name'],
        partitionBy: "date",
        updatePartitionFilter: "date >= date_sub(current_date(),interval
    }
}

WITH
  events AS (
  SELECT
    CONCAT(${helpers.getEventParam('ga_session_id',"int")}, user_pseudo_i
    user_pseudo_id,
    ${helpers.getEventParam('ga_session_id',"int")} AS ga_session_id,
    TIMESTAMP_MICROS(event_timestamp) AS event_timestamp,
    event_name,
    CAST(event_date AS date format 'YYYYMMDD') AS date,
    platform,
    ${helpers.getEventParam('page_title')} AS page_title,
    ${helpers.getEventParam('page_referrer')} AS page_referrer,
    ${helpers.getEventParam('page_location')} AS page_location,

    -- Get custom User parameters
    STRUCT(
      ${helpers.getUserParam('an_user_parameter')} AS an_user_parameter,
      ${helpers.getUserParam('another_user_parameter')} AS another_user_p
    ) as user_params,

    --Get custom Event parameters
    STRUCT(
      ${helpers.getEventParam('an_event_parameter')} AS an_event_paramete
      ${helpers.getEventParam('another_event_parameter')} AS another_ever
    ) as event_params,
```

Problems we need to solve for GA4 + BQ

- Events - sessions - users scope
- Dependencies between tables
- Complex code
- Session attribution
- Non-complete tables

The Data pipeline

Where does Dataform fit in?



Extract - Transfer - Load

Dataform

API

Data sources

Data available in platforms, inside or outside of the company.

Raw data lake

Unstructured and raw data, transformation needed.

Structured data library

Prepped tables with structured data

Looker Studio Visualization tools

Dashboards and ML tools for easier analysis of data.

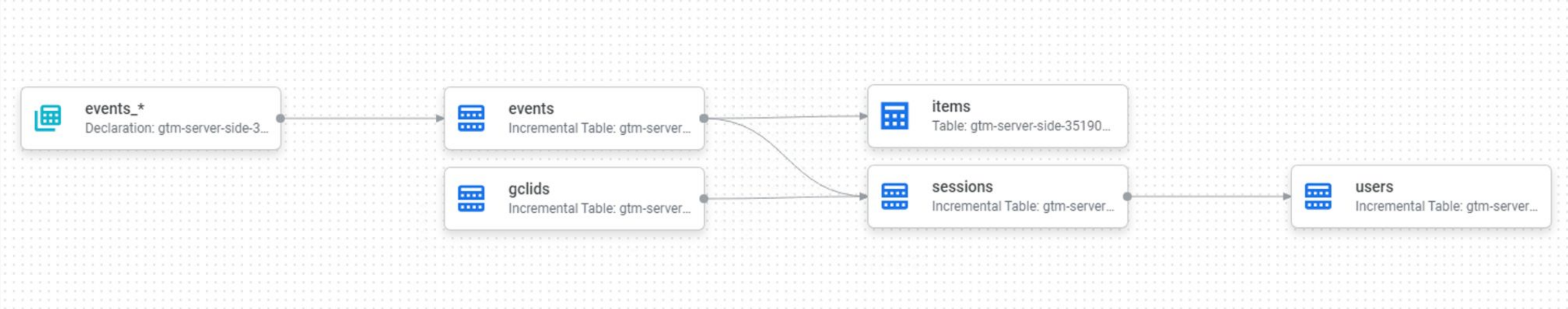
What structured tables do we need to create?

For Google Analytics 4, that is

- Raw GA4 event table
- Events table - every row is an event
  - session_id is foreign key
- Sessions table - every row is an session
  - Session_id is primary key
  - User_pseudo_id is foreign key
- Users table - every row is an user
  - User_pseudo_is primary key

Dataform handles dependencies between tables

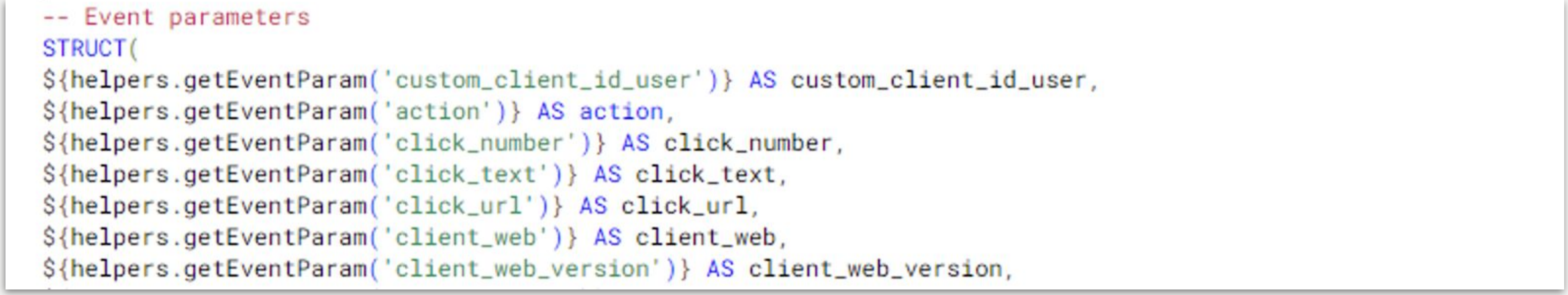
No more complex timing issues


events_*
Declaration: gtm-server-side-3...
events
Incremental Table: gtm-server...
gclids
Incremental Table: gtm-server...
items
Table: gtm-server-side-35190...
sessions
Incremental Table: gtm-server...
users
Incremental Table: gtm-server...

Dataform SQLX helps with complex code

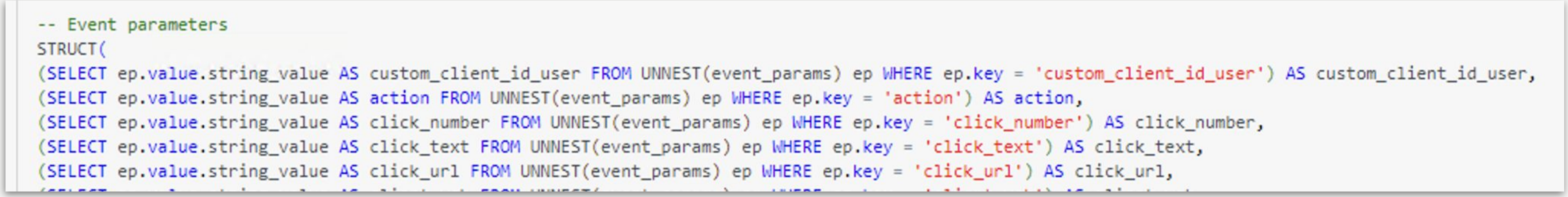
SQL + JavaScript = SQLX!

This code…



```
-- Event parameters
STRUCT(
${helpers.getEventParam('custom_client_id_user')} AS custom_client_id_user,
${helpers.getEventParam('action')} AS action,
${helpers.getEventParam('click_number')} AS click_number,
${helpers.getEventParam('click_text')} AS click_text,
${helpers.getEventParam('click_url')} AS click_url,
${helpers.getEventParam('client_web')} AS client_web,
${helpers.getEventParam('client_web_version')} AS client_web_version,
```

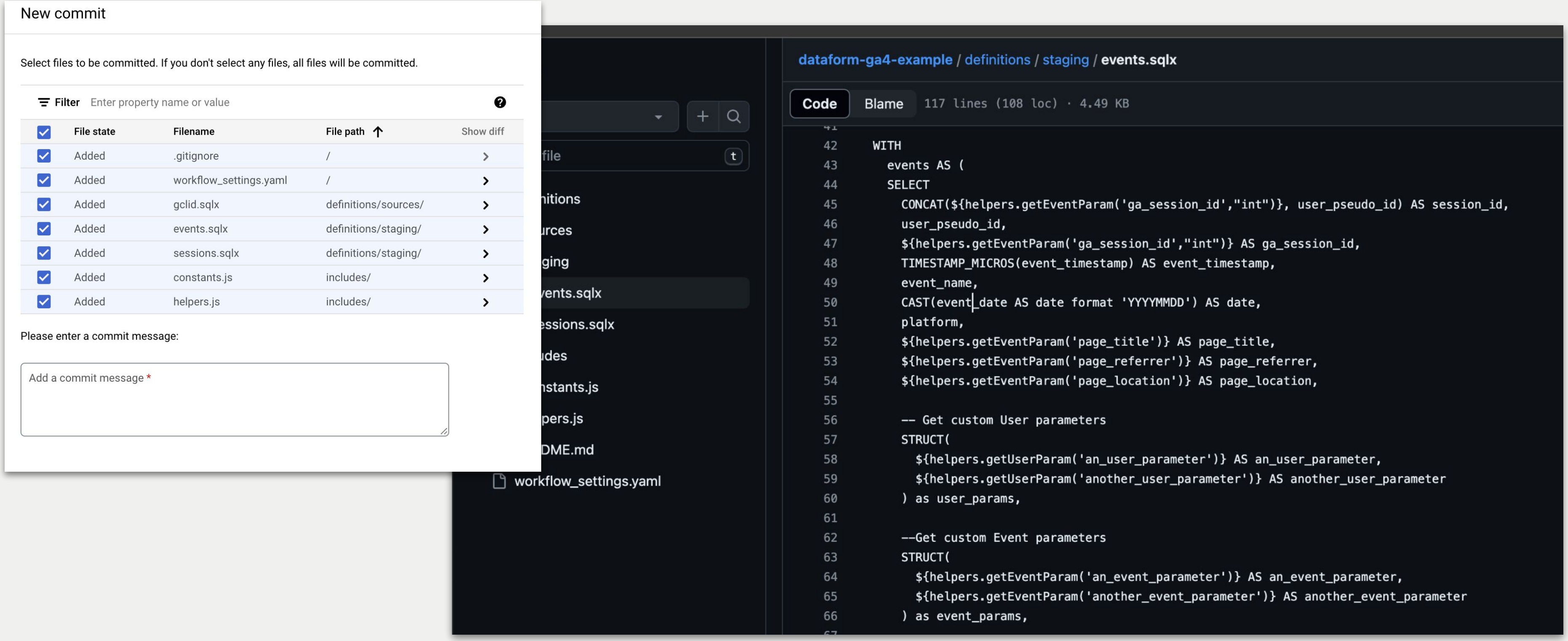
… will compile into this SQL



```
-- Event parameters
STRUCT(
(SELECT ep.value.string_value AS custom_client_id_user FROM UNNEST(event_params) ep WHERE ep.key = 'custom_client_id_user') AS custom_client_id_user,
(SELECT ep.value.string_value AS action FROM UNNEST(event_params) ep WHERE ep.key = 'action') AS action,
(SELECT ep.value.string_value AS click_number FROM UNNEST(event_params) ep WHERE ep.key = 'click_number') AS click_number,
(SELECT ep.value.string_value AS click_text FROM UNNEST(event_params) ep WHERE ep.key = 'click_text') AS click_text,
(SELECT ep.value.string_value AS click_url FROM UNNEST(event_params) ep WHERE ep.key = 'click_url') AS click_url,
```

Dataform have version control!

Use GitHub to handle versions and dev branches



New commit

Select files to be committed. If you don't select any files, all files will be committed.

Filter Enter property name or value

☑	File state	Filename	File path ↑	Show diff
☑	Added	.gitignore	/	>
☑	Added	workflow_settings.yaml	/	>
☑	Added	gclid.sqlx	definitions/sources/	>
☑	Added	events.sqlx	definitions/staging/	>
☑	Added	sessions.sqlx	definitions/staging/	>
☑	Added	constants.js	includes/	>
☑	Added	helpers.js	includes/	>

Please enter a commit message:

Add a commit message *

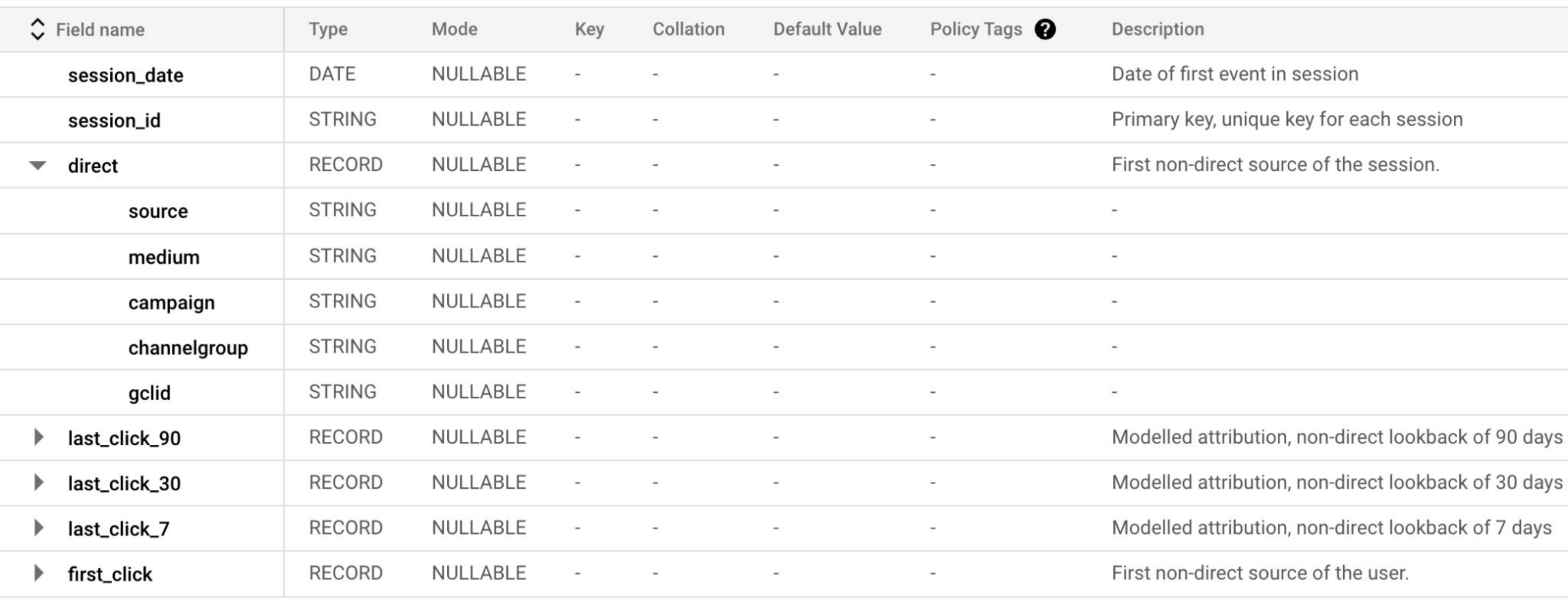
dataform-ga4-example / definitions / staging / events.sqlx

117 lines (108 loc) · 4.49 KB

```
WITH
 events AS (
 SELECT
  CONCAT(${helpers.getEventParam('ga_session_id',"int")}, user_pseudo_id) AS session_id,
  user_pseudo_id,
  ${helpers.getEventParam('ga_session_id',"int")} AS ga_session_id,
  TIMESTAMP_MICROS(event_timestamp) AS event_timestamp,
  event_name,
  CAST(event_date AS date format 'YYYYMMDD') AS date,
  platform,
  ${helpers.getEventParam('page_title')} AS page_title,
  ${helpers.getEventParam('page_referrer')} AS page_referrer,
  ${helpers.getEventParam('page_location')} AS page_location,

  -- Get custom User parameters
  STRUCT(
   ${helpers.getUserParam('an_user_parameter')} AS an_user_parameter,
   ${helpers.getUserParam('another_user_parameter')} AS another_user_parameter
  ) as user_params,

  --Get custom Event parameters
  STRUCT(
   ${helpers.getEventParam('an_event_parameter')} AS an_event_parameter,
   ${helpers.getEventParam('another_event_parameter')} AS another_event_parameter
  ) as event_params,
```

Attributions models easy to handle

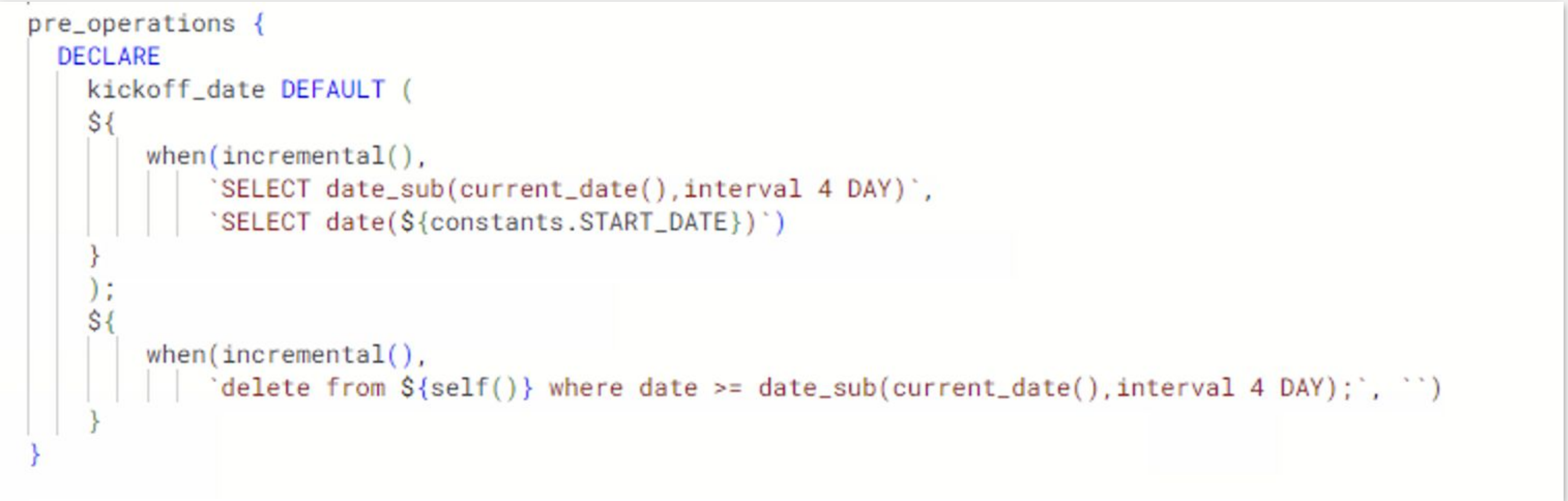
Bring them back to life with ease



Field name	Type	Mode	Key	Collation	Default Value	Policy Tags ?	Description
session_date	DATE	NULLABLE	-	-	-	-	Date of first event in session
session_id	STRING	NULLABLE	-	-	-	-	Primary key, unique key for each session
direct	RECORD	NULLABLE	-	-	-	-	First non-direct source of the session.
source	STRING	NULLABLE	-	-	-	-	-
medium	STRING	NULLABLE	-	-	-	-	-
campaign	STRING	NULLABLE	-	-	-	-	-
channelgroup	STRING	NULLABLE	-	-	-	-	-
gclid	STRING	NULLABLE	-	-	-	-	-
last_click_90	RECORD	NULLABLE	-	-	-	-	Modelled attribution, non-direct lookback of 90 days
last_click_30	RECORD	NULLABLE	-	-	-	-	Modelled attribution, non-direct lookback of 30 days
last_click_7	RECORD	NULLABLE	-	-	-	-	Modelled attribution, non-direct lookback of 7 days
first_click	RECORD	NULLABLE	-	-	-	-	First non-direct source of the user.

Handling retroactive backfill of raw table

For increments, have an rolling 4 day update - delete, then update



```
pre_operations {
  DECLARE
    kickoff_date DEFAULT (
    ${
        when(incremental(),
            `SELECT date_sub(current_date(),interval 4 DAY)`,
            `SELECT date(${constants.START_DATE})`)
    }
    );
    ${
        when(incremental(),
            `delete from ${self()} where date >= date_sub(current_date(),interval 4 DAY);`, ``)
    }
}
```

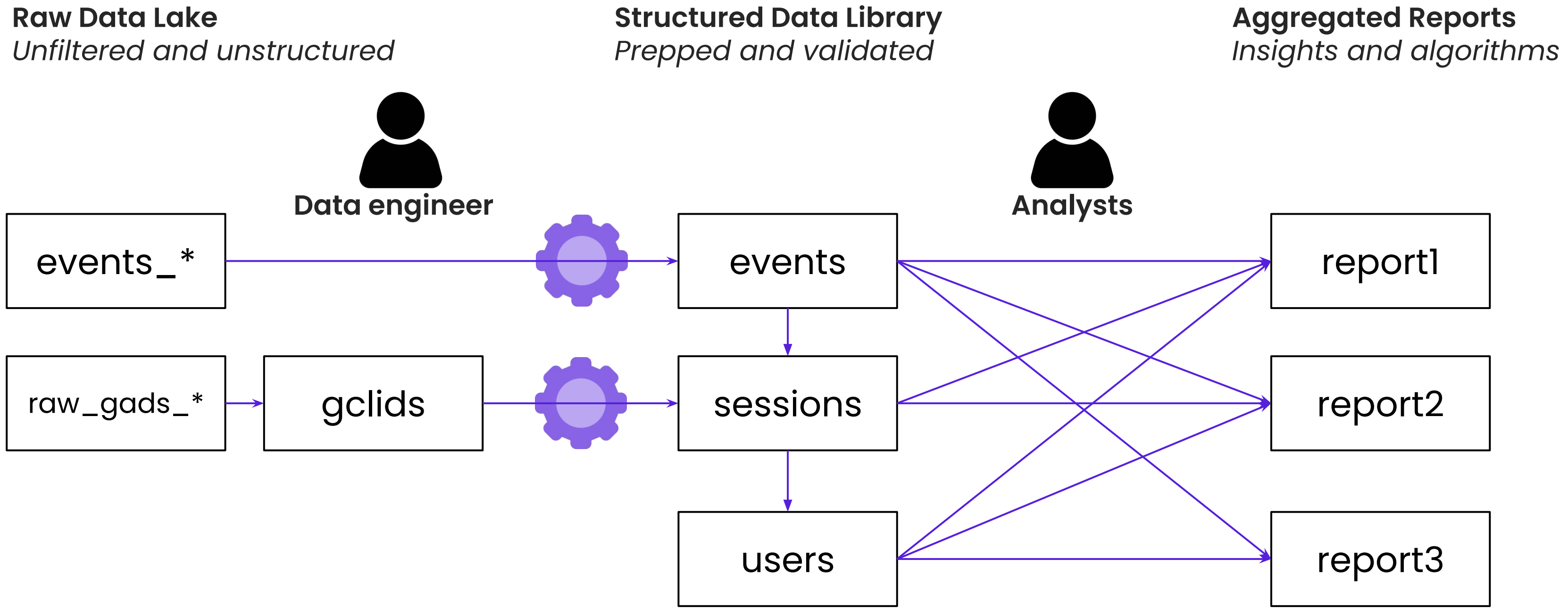
CTRL

The Data pipeline

Basic setup

An example for a GA4 data library

⚙ = Logics, KPIs, Attribution



Remember the large ecommerce site in the beginning?
(85 queries, yes them)

Let's look at their case

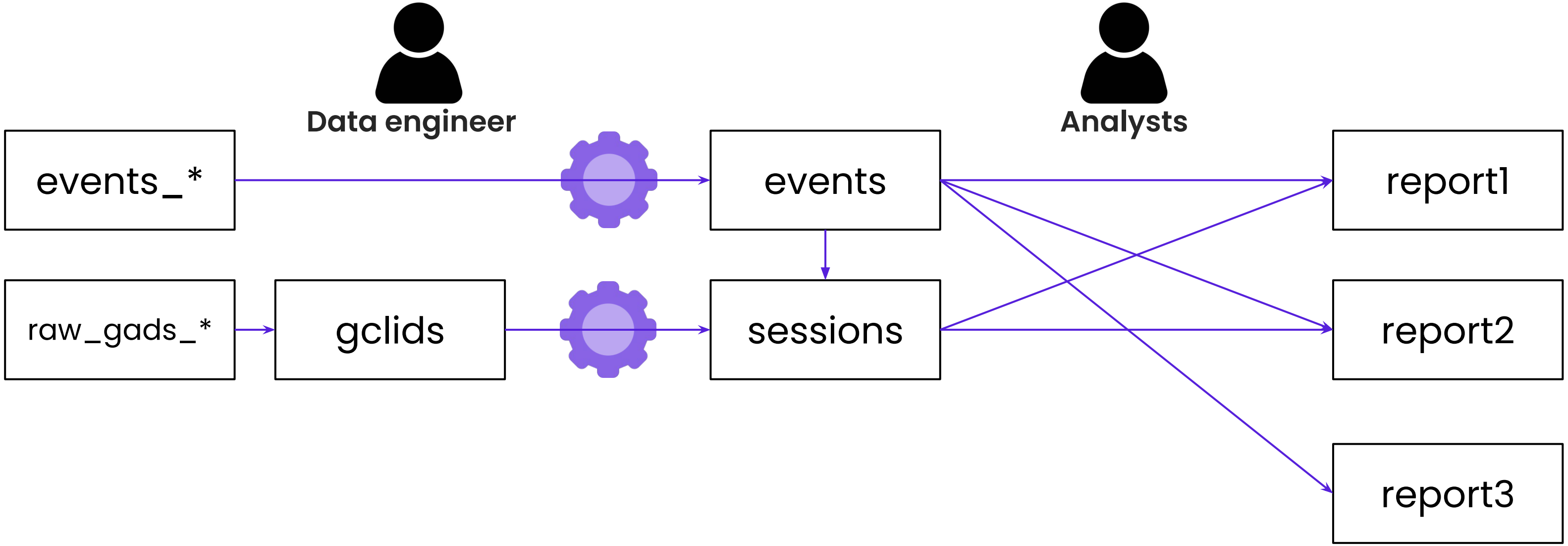
The setup we made for a large ecommerce site

⚙ = Logics, KPIs, Attribution

Raw Data Lake
Unfiltered and unstructured

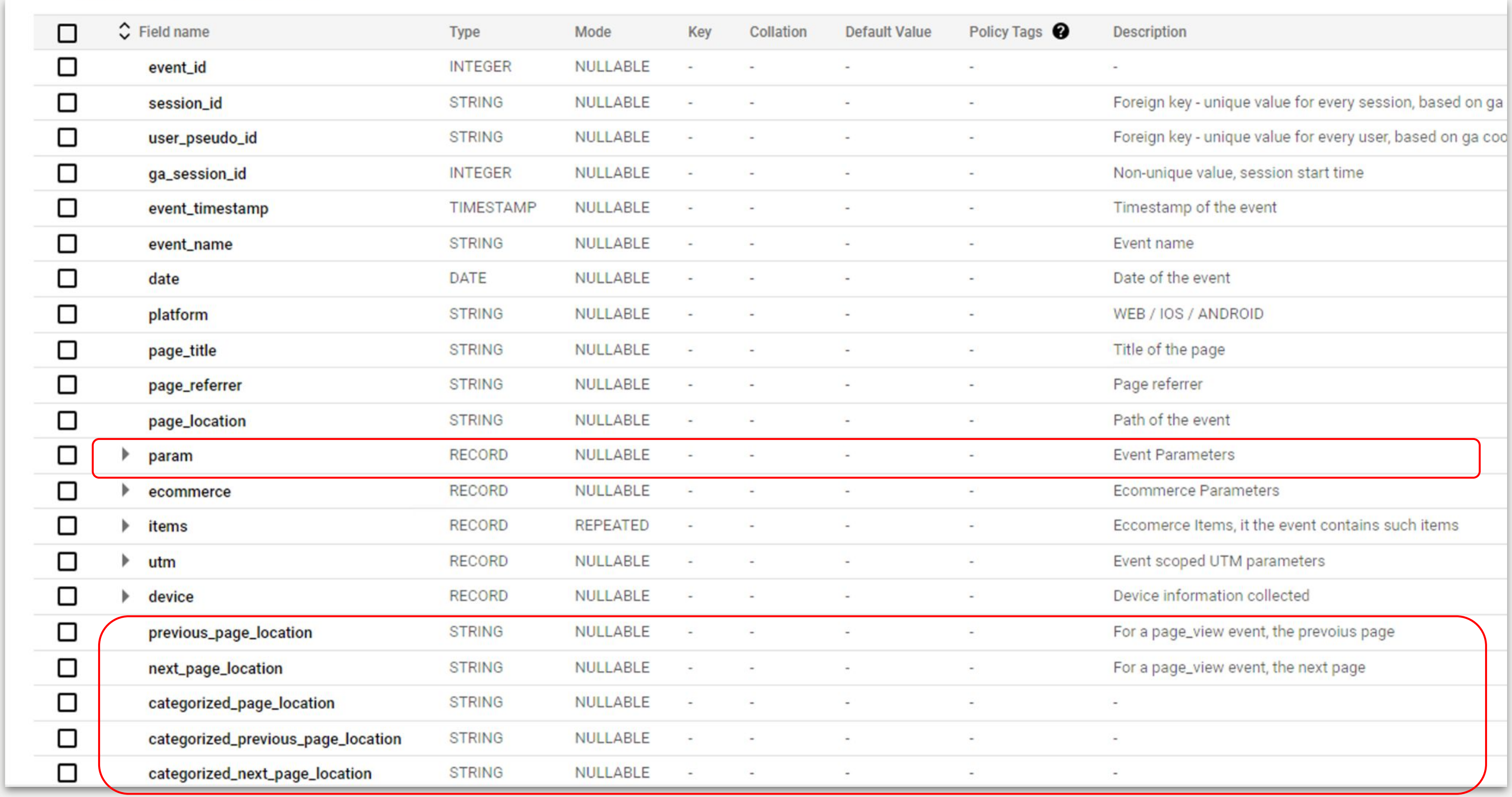
Structured Data Library
Prepped and validated

Aggregated Reports
Insights and algorithms



The events table

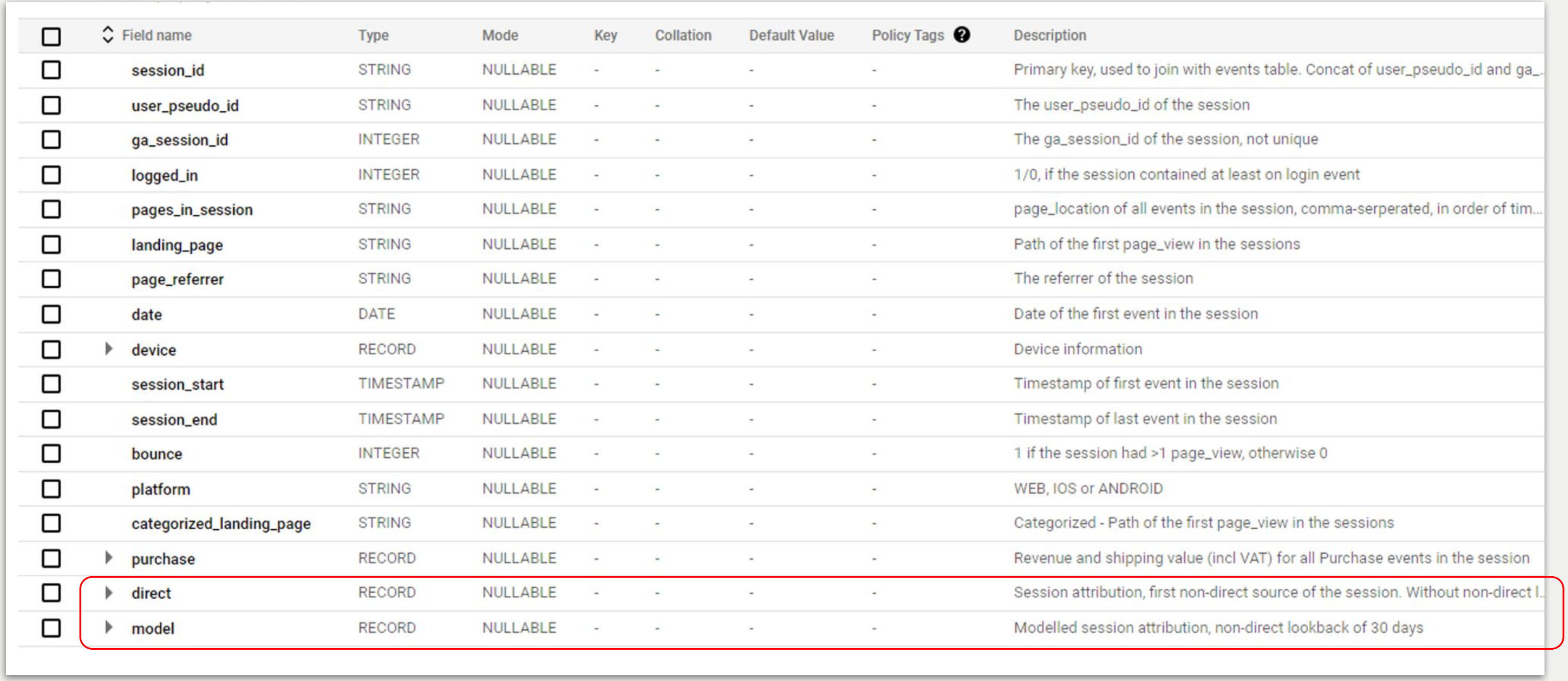
Event parameters is available in an un-nested record, page_locations prepared for analysis



□	Field name	Type	Mode	Key	Collation	Default Value	Policy Tags ?	Description
□	event_id	INTEGER	NULLABLE	-	-	-	-	-
□	session_id	STRING	NULLABLE	-	-	-	-	Foreign key - unique value for every session, based on ga
□	user_pseudo_id	STRING	NULLABLE	-	-	-	-	Foreign key - unique value for every user, based on ga coo
□	ga_session_id	INTEGER	NULLABLE	-	-	-	-	Non-unique value, session start time
□	event_timestamp	TIMESTAMP	NULLABLE	-	-	-	-	Timestamp of the event
□	event_name	STRING	NULLABLE	-	-	-	-	Event name
□	date	DATE	NULLABLE	-	-	-	-	Date of the event
□	platform	STRING	NULLABLE	-	-	-	-	WEB / IOS / ANDROID
□	page_title	STRING	NULLABLE	-	-	-	-	Title of the page
□	page_referrer	STRING	NULLABLE	-	-	-	-	Page referrer
□	page_location	STRING	NULLABLE	-	-	-	-	Path of the event
□	▸ param	RECORD	NULLABLE	-	-	-	-	Event Parameters
□	▸ ecommerce	RECORD	NULLABLE	-	-	-	-	Ecommerce Parameters
□	▸ items	RECORD	REPEATED	-	-	-	-	Eccomerce Items, it the event contains such items
□	▸ utm	RECORD	NULLABLE	-	-	-	-	Event scoped UTM parameters
□	▸ device	RECORD	NULLABLE	-	-	-	-	Device information collected
□	previous_page_location	STRING	NULLABLE	-	-	-	-	For a page_view event, the prevoius page
□	next_page_location	STRING	NULLABLE	-	-	-	-	For a page_view event, the next page
□	categorized_page_location	STRING	NULLABLE	-	-	-	-	-
□	categorized_previous_page_location	STRING	NULLABLE	-	-	-	-	-
□	categorized_next_page_location	STRING	NULLABLE	-	-	-	-	-

The sessions table

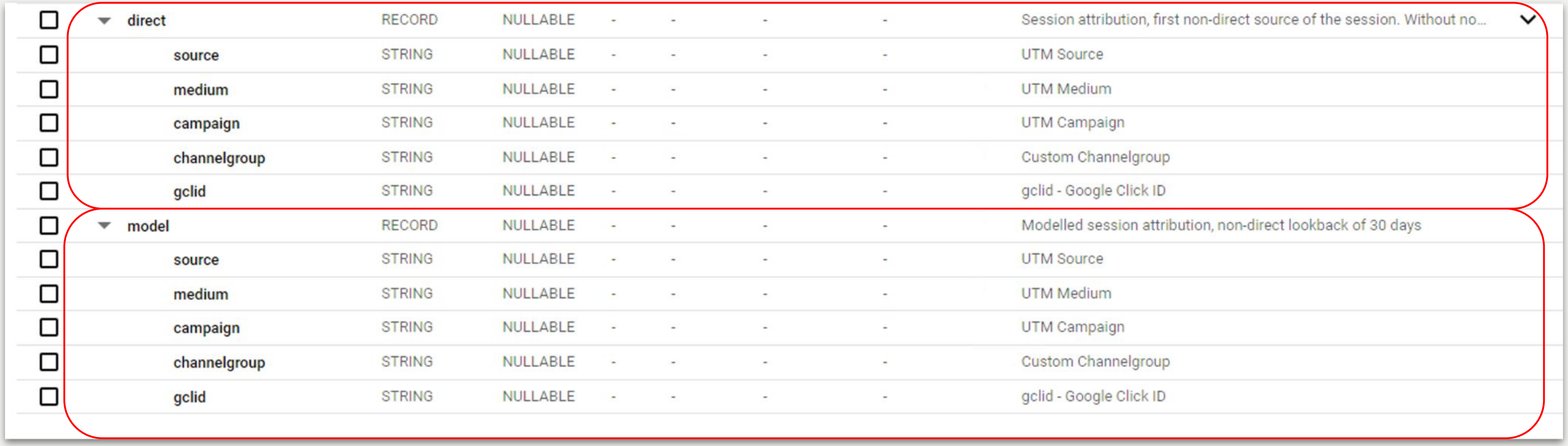
Sessionized data and session_id acts as key to event table



□	Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
□	session_id	STRING	NULLABLE	-	-	-	-	Primary key, used to join with events table. Concat of user_pseudo_id and ga_...
□	user_pseudo_id	STRING	NULLABLE	-	-	-	-	The user_pseudo_id of the session
□	ga_session_id	INTEGER	NULLABLE	-	-	-	-	The ga_session_id of the session, not unique
□	logged_in	INTEGER	NULLABLE	-	-	-	-	1/0, if the session contained at least on login event
□	pages_in_session	STRING	NULLABLE	-	-	-	-	page_location of all events in the session, comma-serperated, in order of tim...
□	landing_page	STRING	NULLABLE	-	-	-	-	Path of the first page_view in the sessions
□	page_referrer	STRING	NULLABLE	-	-	-	-	The referrer of the session
□	date	DATE	NULLABLE	-	-	-	-	Date of the first event in the session
□	▸ device	RECORD	NULLABLE	-	-	-	-	Device information
□	session_start	TIMESTAMP	NULLABLE	-	-	-	-	Timestamp of first event in the session
□	session_end	TIMESTAMP	NULLABLE	-	-	-	-	Timestamp of last event in the session
□	bounce	INTEGER	NULLABLE	-	-	-	-	1 if the session had >1 page_view, otherwise 0
□	platform	STRING	NULLABLE	-	-	-	-	WEB, IOS or ANDROID
□	categorized_landing_page	STRING	NULLABLE	-	-	-	-	Categorized - Path of the first page_view in the sessions
□	▸ purchase	RECORD	NULLABLE	-	-	-	-	Revenue and shipping value (incl VAT) for all Purchase events in the session
□	▸ direct	RECORD	NULLABLE	-	-	-	-	Session attribution, first non-direct source of the session. Without non-direct l...
□	▸ model	RECORD	NULLABLE	-	-	-	-	Modelled session attribution, non-direct lookback of 30 days

The sessions table

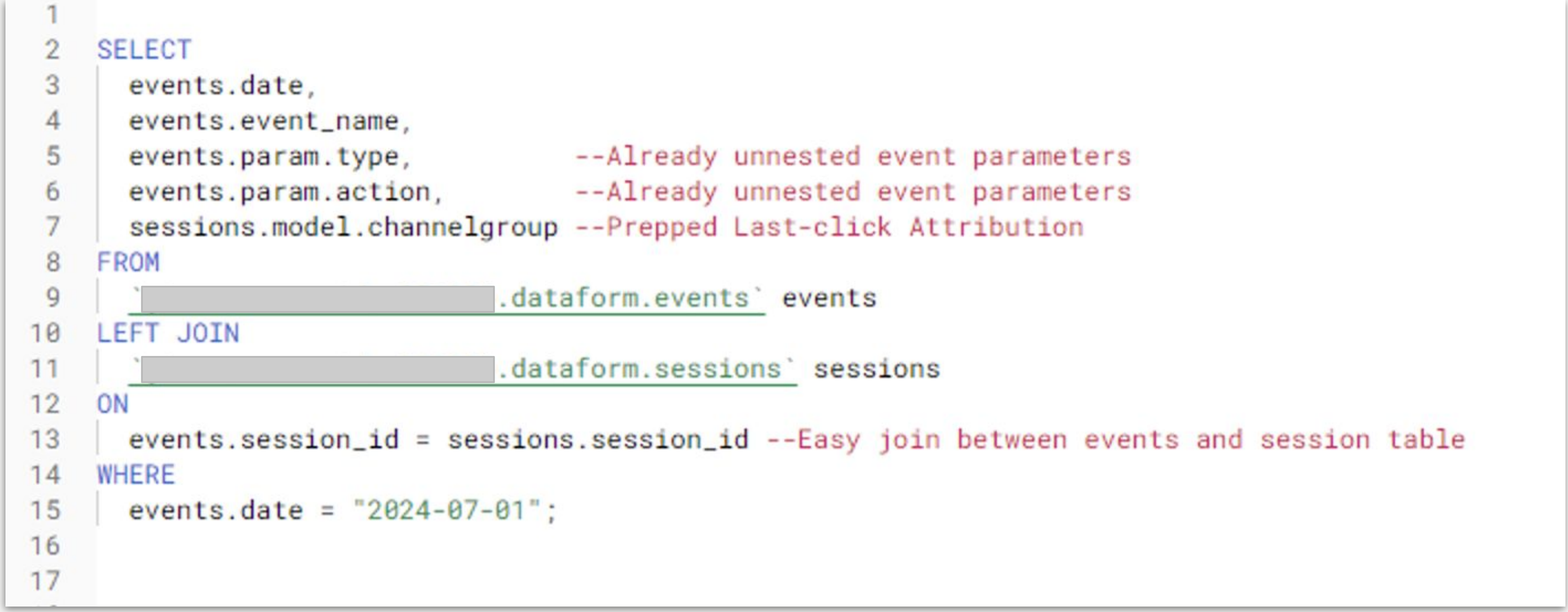
Attribution models and purchase logic is available on session level



☐	▾ direct	RECORD	NULLABLE	-	-	-	-	Session attribution, first non-direct source of the session. Without no...
☐	source	STRING	NULLABLE	-	-	-	-	UTM Source
☐	medium	STRING	NULLABLE	-	-	-	-	UTM Medium
☐	campaign	STRING	NULLABLE	-	-	-	-	UTM Campaign
☐	channelgroup	STRING	NULLABLE	-	-	-	-	Custom Channelgroup
☐	gclid	STRING	NULLABLE	-	-	-	-	gclid - Google Click ID
☐	▾ model	RECORD	NULLABLE	-	-	-	-	Modelled session attribution, non-direct lookback of 30 days
☐	source	STRING	NULLABLE	-	-	-	-	UTM Source
☐	medium	STRING	NULLABLE	-	-	-	-	UTM Medium
☐	campaign	STRING	NULLABLE	-	-	-	-	UTM Campaign
☐	channelgroup	STRING	NULLABLE	-	-	-	-	Custom Channelgroup
☐	gclid	STRING	NULLABLE	-	-	-	-	gclid - Google Click ID

Example query for the analyst

Easy to access prepperad data where advanced logic have already been applied



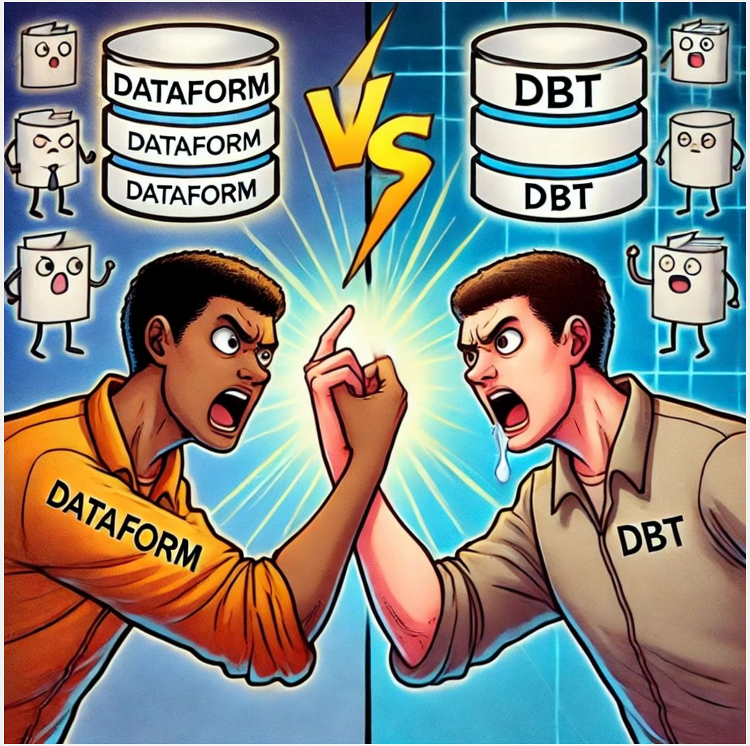
```sql

SELECT
  events.date,
  events.event_name,
  events.param.type,          --Already unnested event parameters
  events.param.action,        --Already unnested event parameters
  sessions.model.channelgroup --Prepped Last-click Attribution
FROM
  `[redacted].dataform.events` events
LEFT JOIN
  `[redacted].dataform.sessions` sessions
ON
  events.session_id = sessions.session_id --Easy join between events and session table
WHERE
  events.date = "2024-07-01";


```

CTRL

What about dbt?

Dataform and dbt aim to solve the same problems

Slight differences, outcome is the same



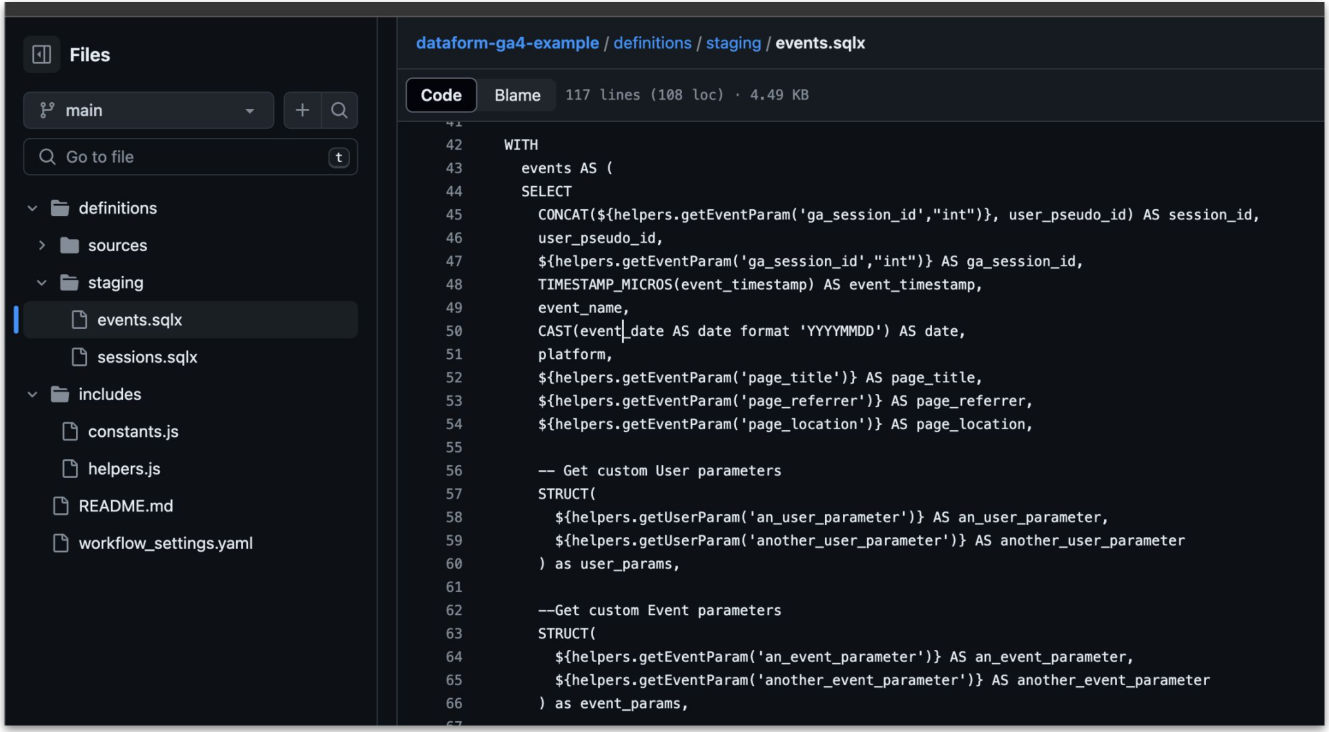
Summary

- Dataform help us orchestrate our SQL
- It doesn´t replace BigQuery
- No reason not to use, only upsides
- A bit of time investment to get started, then huge ROI in time

“The amount of time we'll save on this gives me goosebumps.”

My example repository for GA4 and Dataform

https://github.com/ctrl-digital/dataform-ga4-example



```
WITH
  events AS (
  SELECT
    CONCAT(${helpers.getEventParam('ga_session_id',"int")}, user_pseudo_id) AS session_id,
    user_pseudo_id,
    ${helpers.getEventParam('ga_session_id',"int")} AS ga_session_id,
    TIMESTAMP_MICROS(event_timestamp) AS event_timestamp,
    event_name,
    CAST(event_date AS date format 'YYYYMMDD') AS date,
    platform,
    ${helpers.getEventParam('page_title')} AS page_title,
    ${helpers.getEventParam('page_referrer')} AS page_referrer,
    ${helpers.getEventParam('page_location')} AS page_location,

    -- Get custom User parameters
    STRUCT(
      ${helpers.getUserParam('an_user_parameter')} AS an_user_parameter,
      ${helpers.getUserParam('another_user_parameter')} AS another_user_parameter
    ) as user_params,

    --Get custom Event parameters
    STRUCT(
      ${helpers.getEventParam('an_event_parameter')} AS an_event_parameter,
      ${helpers.getEventParam('another_event_parameter')} AS another_event_parameter
    ) as event_params,
```

CTRL

johan.strand@ctrldigital.com

Thanks! Questions?

Let´s connect on Linkedin