On Track with Apache Kafka®: Building a Streaming ETL solution with Rail Data 🚂

@rmoff @confluentinc #KafkaMeetup

h/t @kennybastani / @gamussa


Departures 1
Due Destination Plat Expected
11:48 Sheffield via 11A On time
11:50 Manchester Air 16A On time
11:51 Blackpool Nth 11C On time
11:52 Newcastle 15A On time
11:53 Huddersfield via 13B On time
11:55 Selby 7A On time
11:56 Skipton 1B On time
11:58 Knottingley via 12D 12:08
11:59 Knaresborough via 5C On time
12:03 Ilkley 2B On time
12:06 Liverpool L St via 16A On time
12:06 Brighouse via 8B On time
12:08 Glasgow Cent. 9D On time
Delayed due to
signalling problems
12:09 Nottingham 17B On time
12:11 Middlesbrough 15B On time
12:11 Plymouth 11A 12:16
Departures 2
11:48
Due Destination Plat Expected
12:12 Bradford F Sq 1B 12:19
12:15 Harrogate 4B On time
12:15 London Kings X On time
Platform to be confirmed
please wait on the concourse
12:18 Manchester Air 16A On time
12:18 Morecambe 3B On time
12:18 via Halifax 10B On time
12:20 via Dewsbury 13B On time
12:21 Scarborough 15B On time
12:21 Doncaster On time
12:24 York 7A On time
12:26 Skipton 5C On time
12:28 Knottingley 12A On time
via Wakefield Westgate
12:29 via Harrogate 4B On time
12:30 via Castleford 17A On time
12:33 Ilkley 2B On time
Special Notices
Due to congestion near
the ticket gates: please
wait on the concourse
until the platform
displays for London KX
services. Thank you.
Welcome to
Leeds
Station
Arrivals
Newcastle
London Kings X
Liverpool L St
Plymouth
Manchester Air
Glasgow Cent.
London Kings X
Liverpool L St
Newcastle
Manchester Air
Blackpool Nth
Hull

https://wiki.openraildata.com/index.php?title=Train_Movements

Main page
Recent changes
Random page
Help

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information
Cite this page

Page | Discussion

Train Movements

The Train Movements feed contains data from Network Rail's TRUST system. This reports on the progress of trains al

There's a fuller description of the process available.

Because of the way TRUST works not every location will generate a report. If you want fine grain reporting then you wou
Port data. Darwin will generally capture more, better quality, and ad-hoc data as its part of the bigger customer information system - TRUST is an internal reporting tool and can be upda
be late being input. A good example is cancelled services - they may be input to Darwin via another source (Darwin Workstation, Tyrell, CIS) leading to a cancellation, the actual TRUST
later.

TRUST data may also be updated/amended 'after the fact' so a delay which is input manually from TRUST DA may overwrite a previous automatic report and that could happen severa
historically correct.

TRUST does not provide predictions, it only reports on events that have happened. TRUST reports in 30 second intervals, and rounds to whole minutes. - A train reporting 1 minute ear
in reality.

Message Topics

Train movement messages are available through ActiveMQ TOC-specific topics, as well as an 'all TOCs' topic. You should note that the list of topics you can subscribe too may not mat
new operator is added to the system it may not show up in the individual topic lists for a while, however the data is sent to the TRAIN_MVT_ALL_TOC topic by default. Once the individu
per below. If you notice a missing or incorrect operator then please email support.

Topic name	Description
TRAIN_MVT_*XX*_TOC	Train movement messages for *XX* TOC. These codes are business codes on the TOC Codes page
TRAIN_MVT_ALL_TOC	All train operating companies
TRAIN_MVT_FREIGHT	All non-passenger train operating company movements
TRAIN_MVT_GENERAL	Change of Identity messages for freight trains

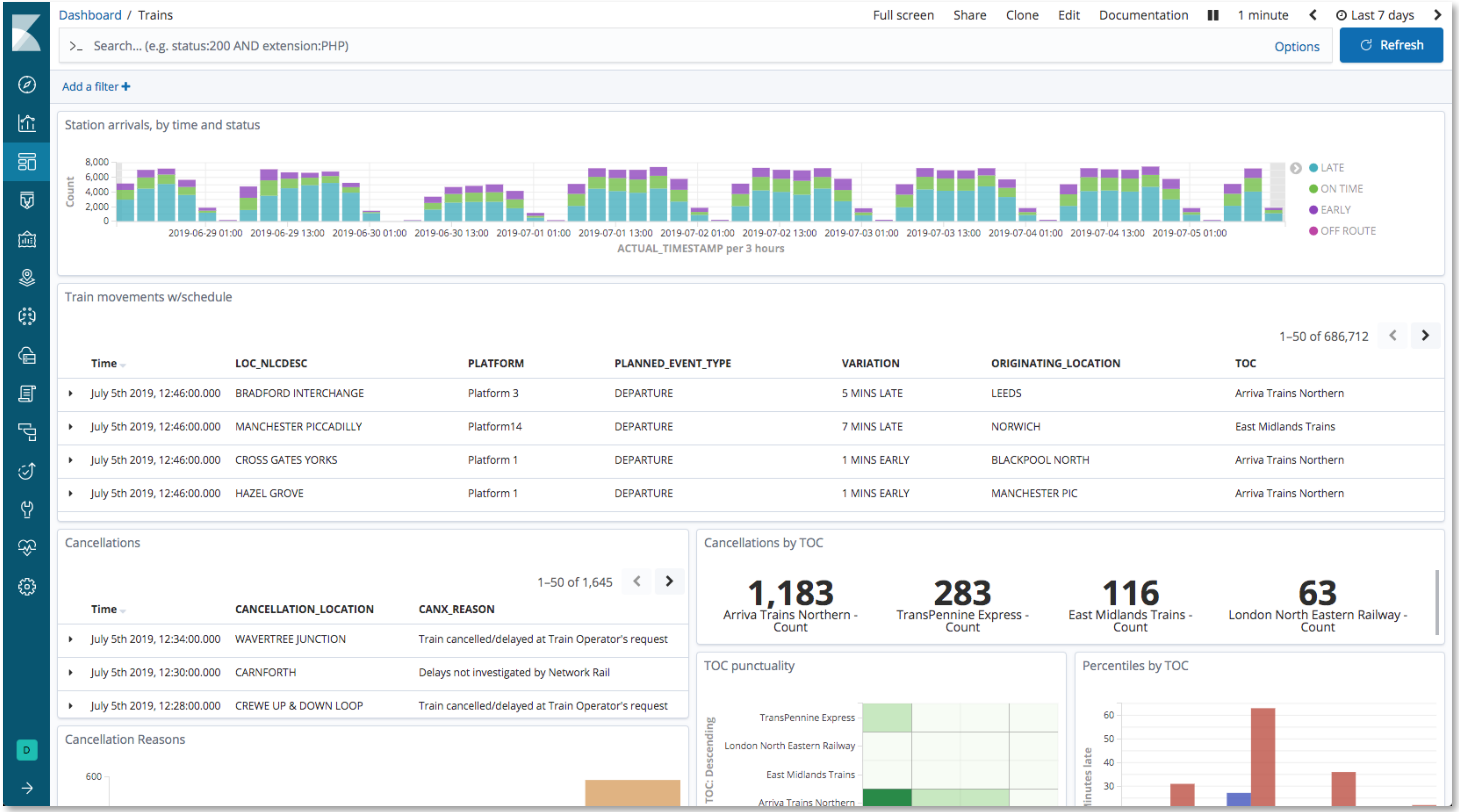
Message types

The feed contains eight message types, all in JSON format:

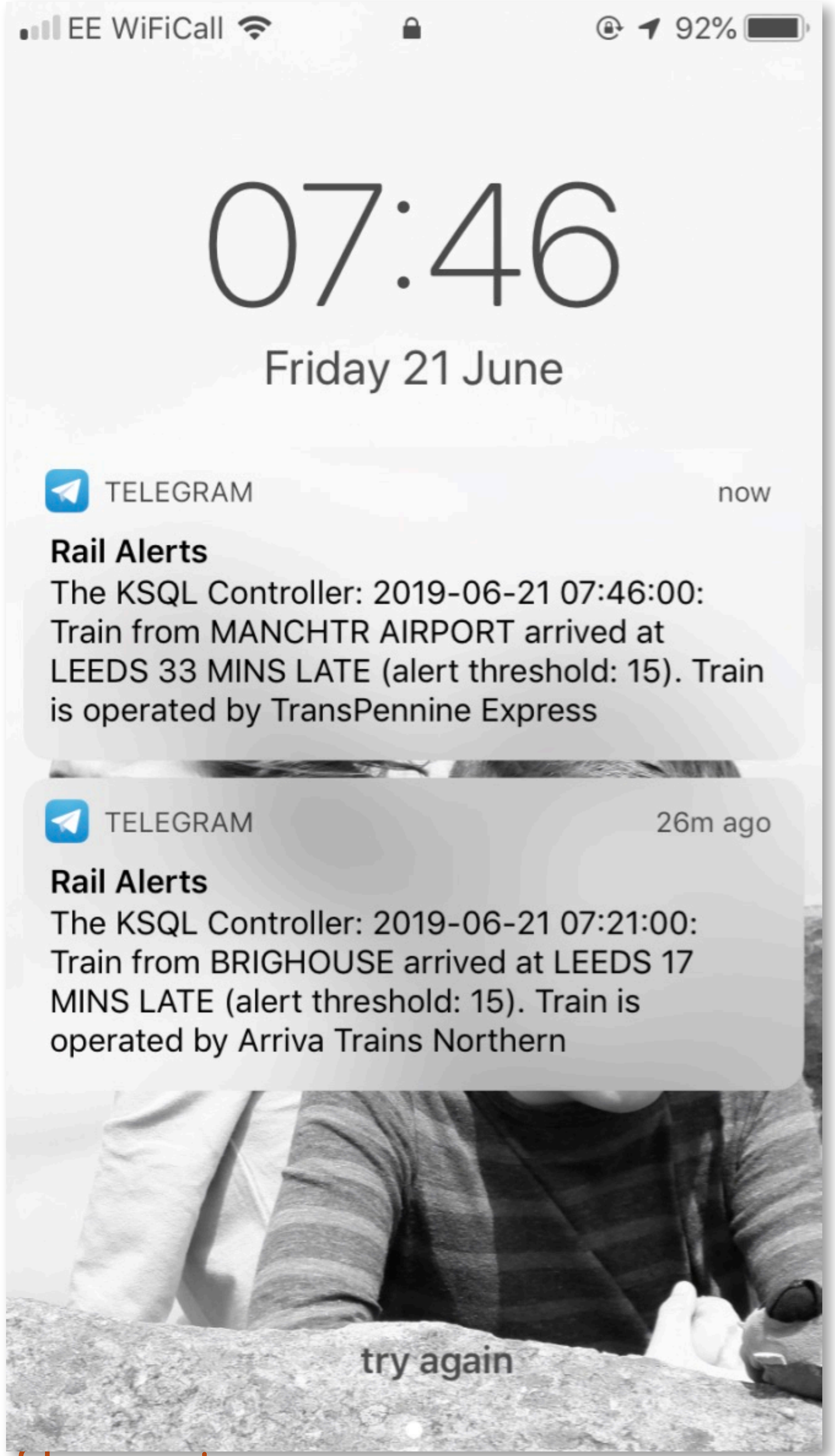
Type	Description
0001	Train Activation
0002	Train Cancellation
0003	Train Movement
0004	Unidentified Train (not used in production)

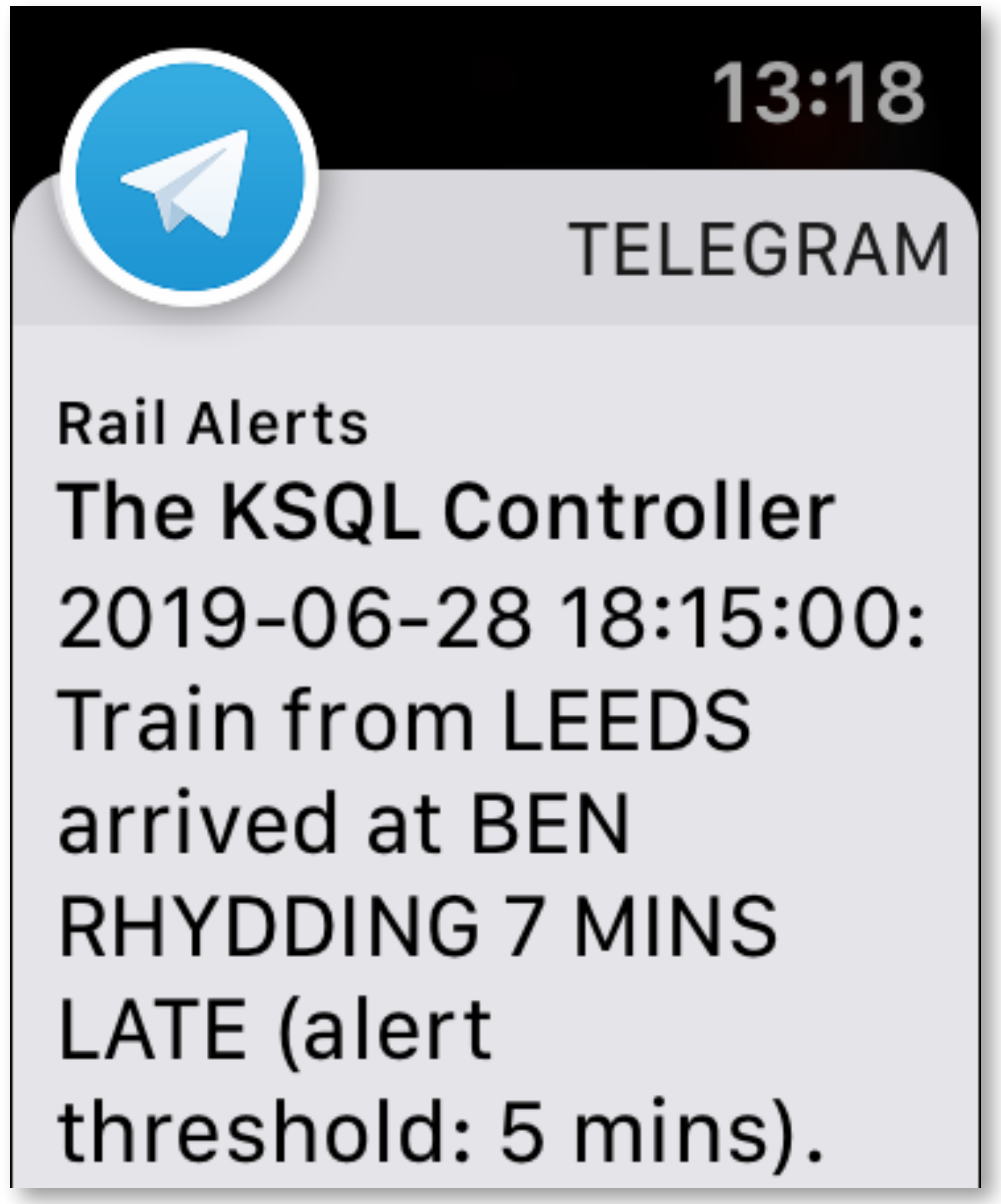
https://wiki.openraildata.com

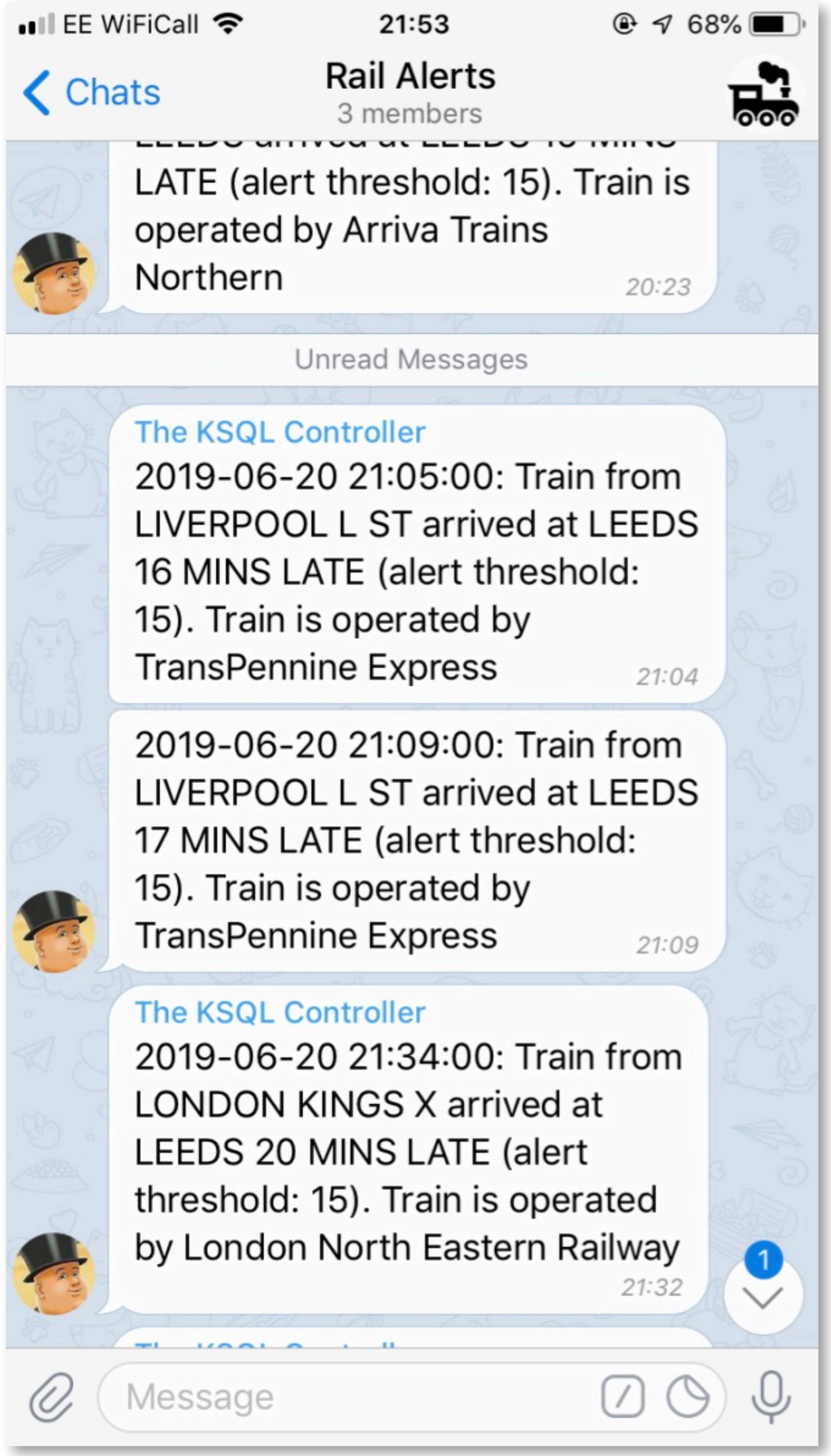
Real-time visualisation of rail data

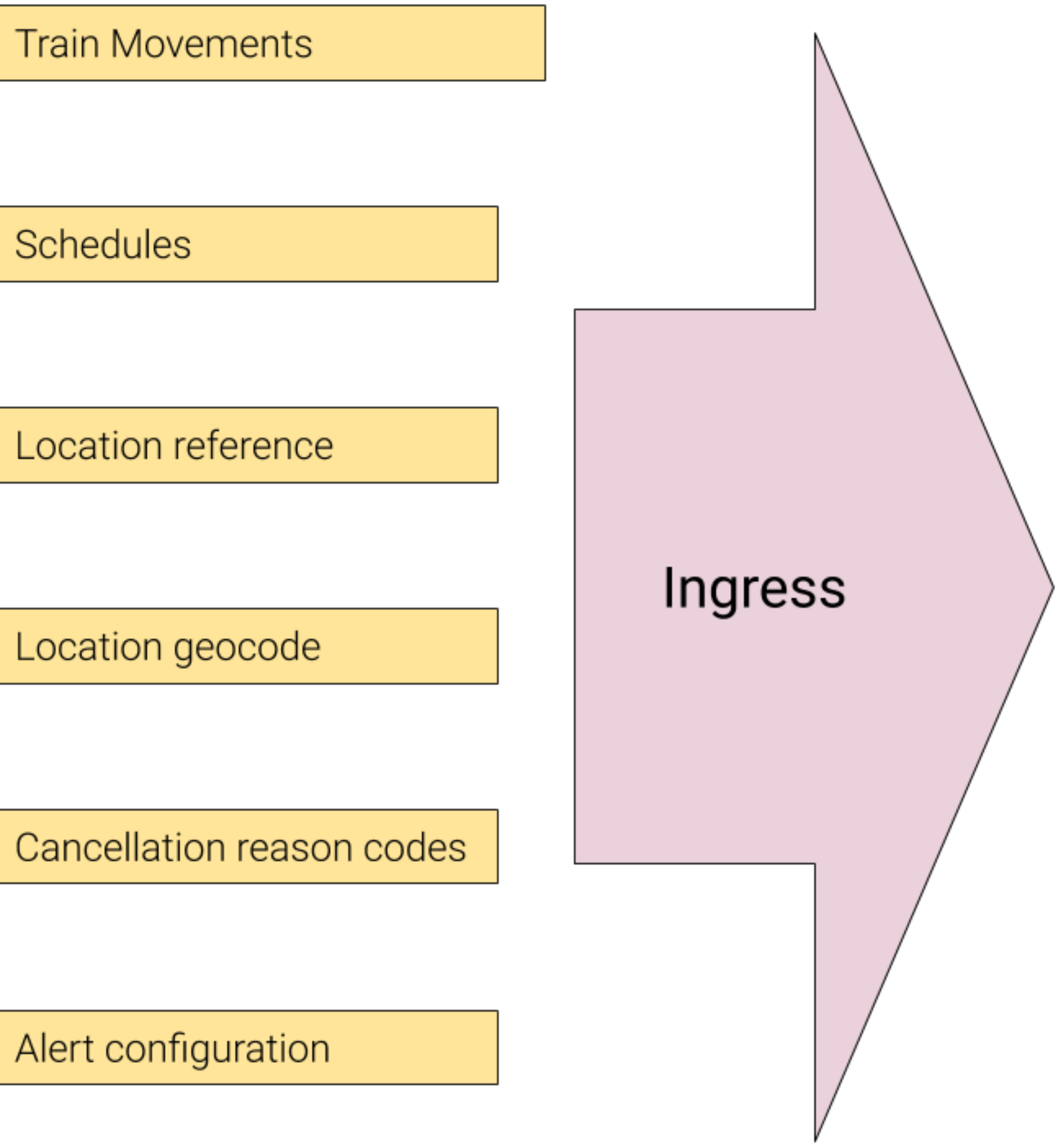


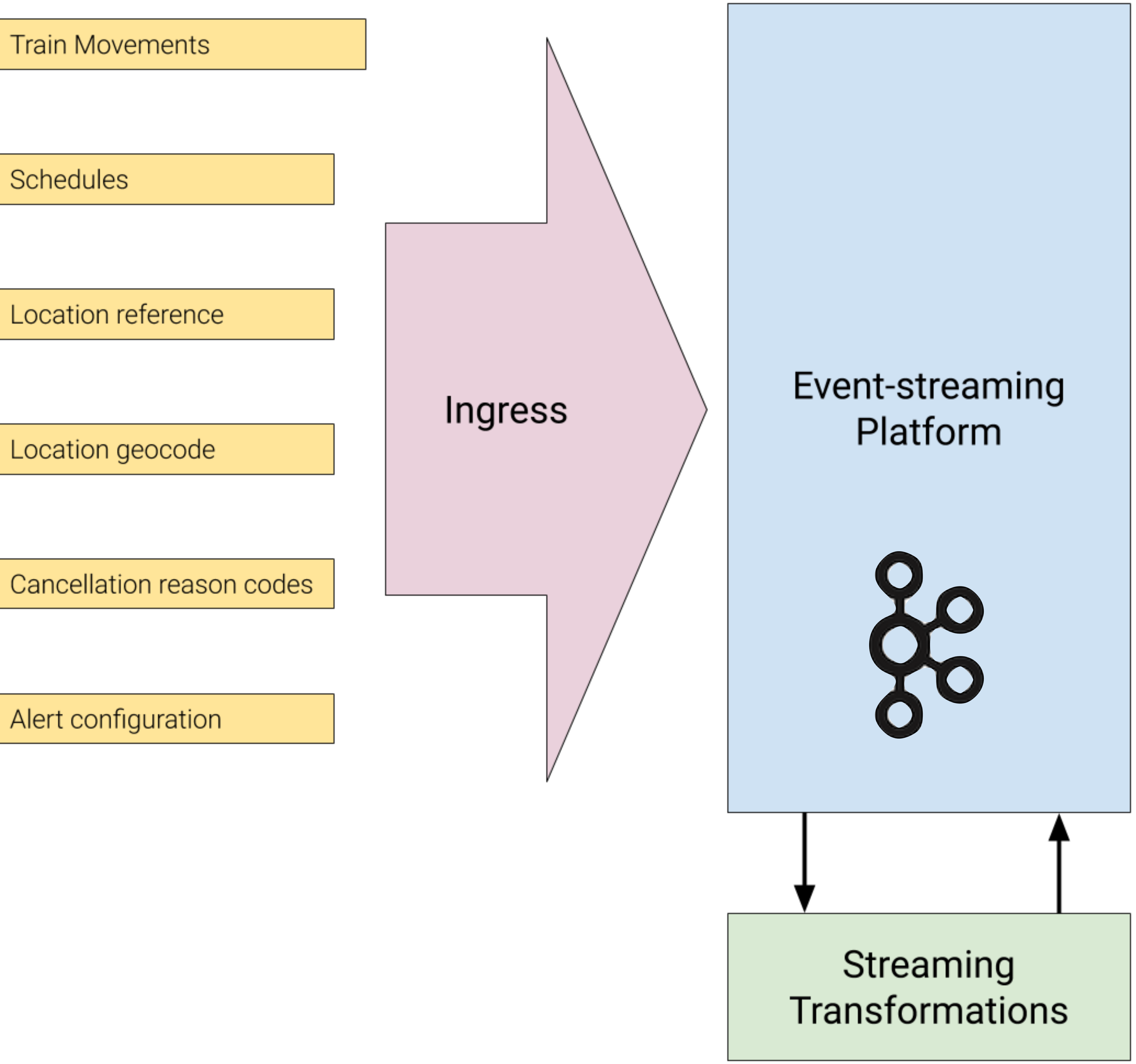
Tell me when trains are delayed at my station

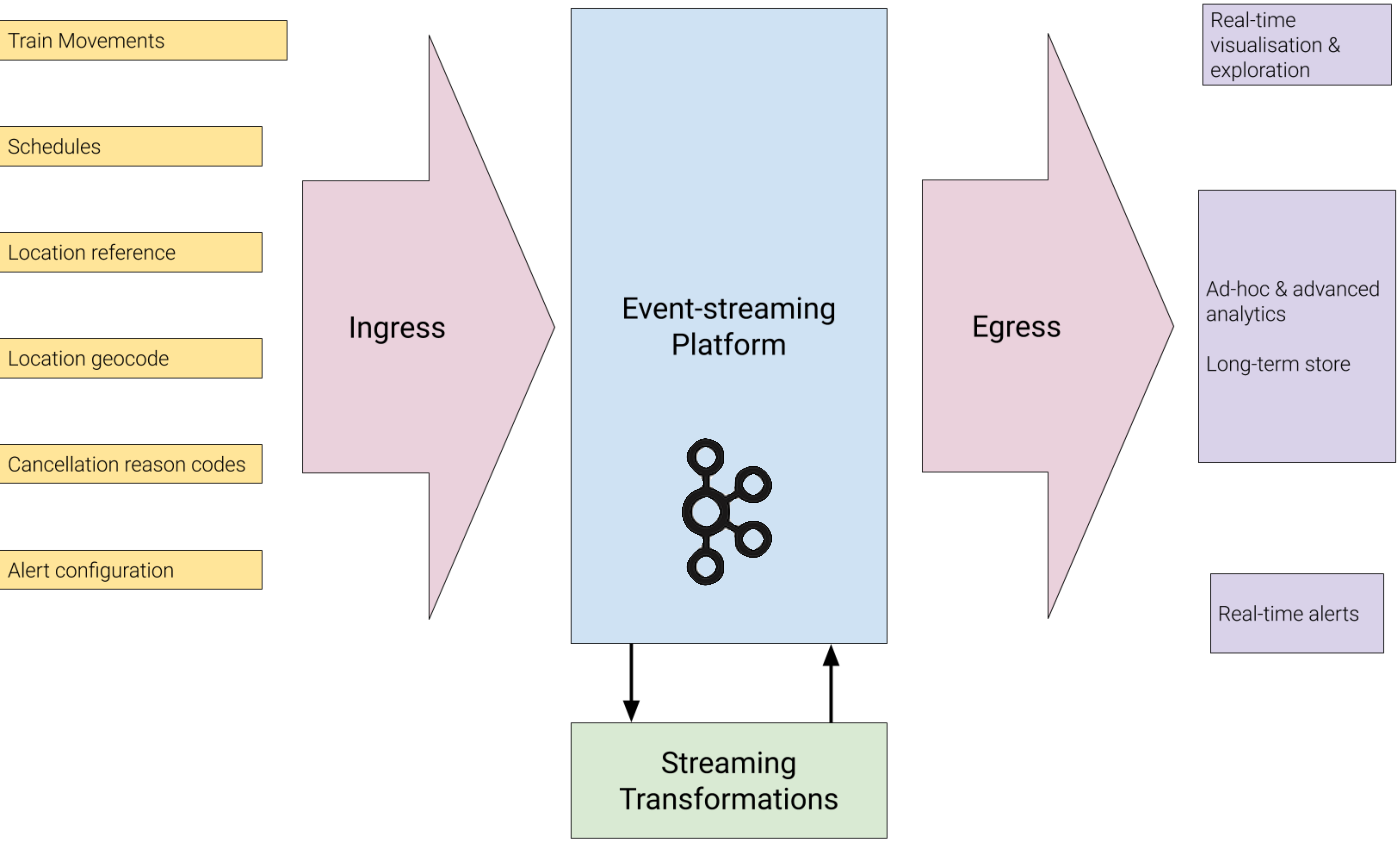


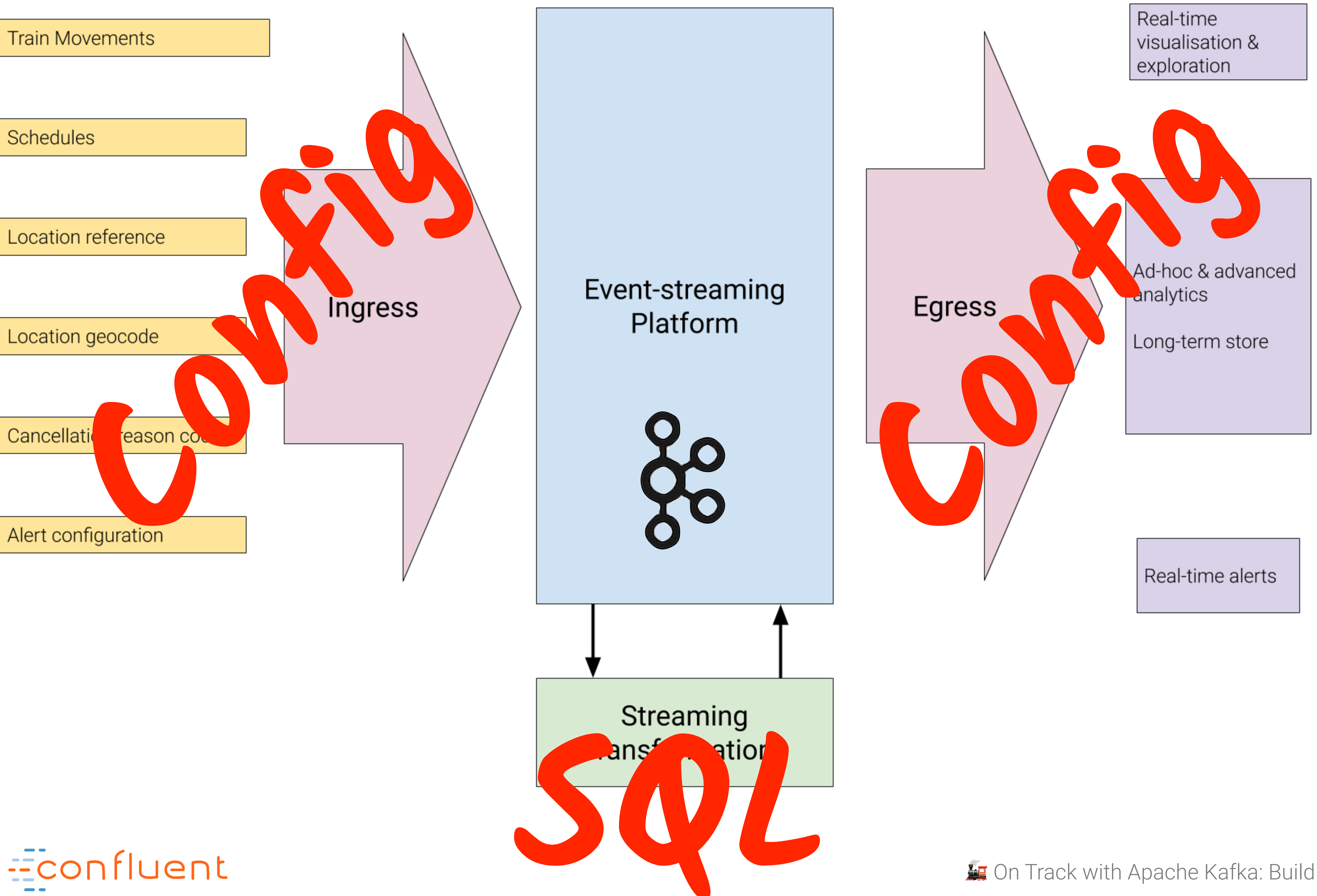




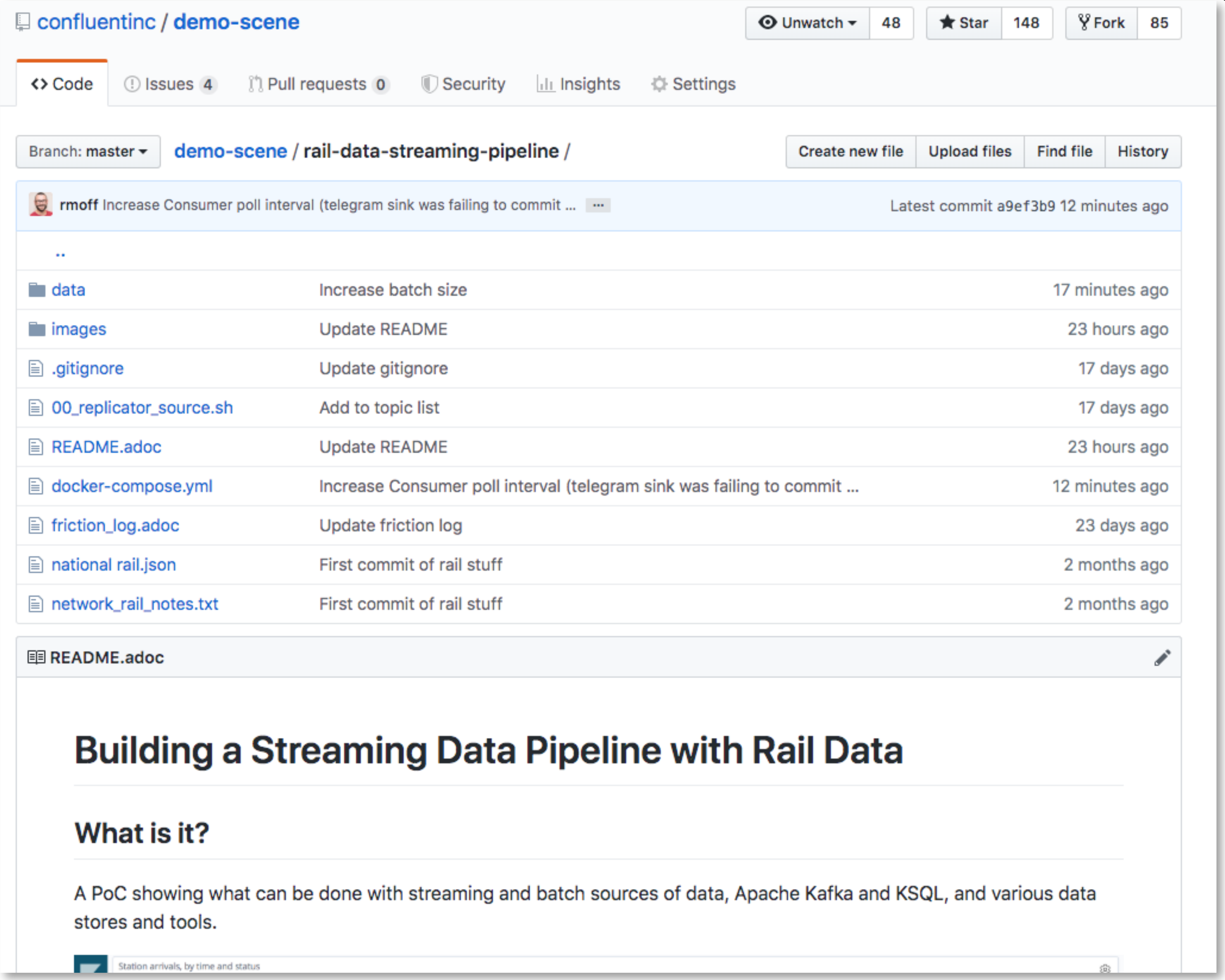

Train Movements
Schedules
Location reference
Location geocode
Cancellation reason codes
Alert configuration
Ingress


Train Movements
Schedules
Location reference
Location geocode
Cancellation reason codes
Alert configuration
Ingress
Event-streaming Platform
Streaming Transformations


Train Movements
Schedules
Location reference
Location geocode
Cancellation reason codes
Alert configuration
Ingress
Event-streaming Platform
Streaming Transformations
Egress
Real-time visualisation & exploration
Ad-hoc & advanced analytics
Long-term store
Real-time alerts


Train Movements
Schedules
Location reference
Location geocode
Alert configuration
Ingress
Config
Event-streaming Platform
Streaming
SQL
Egress
Config
Real-time visualisation & exploration
Ad-hoc & advanced analytics
Long-term store
Real-time alerts

Code!



http://rmoff.dev/kafka-trains-code-01

Getting the data in

Data Sources

Apache ACTIVEMQ

Train Movements

Continuous feed

Schedules

Updated Daily

OpenCage

JSON

Reference data

Static

Streaming Integration with Kafka Connect

kafkacat

- Kafka CLI producer & consumer
  - Also metadata inspection
- Integrates beautifully with unix pipes!

```
curl -s “https://my.api.endpoint" | \
kafkacat -b localhost:9092 -t topic -P
```

- Get it!
  - `https://github.com/edenhill/kafkacat/`
  - `apt-get install kafkacat`
  - `brew install kafkacat`
  - `https://hub.docker.com/r/confluentinc/cp-kafkacat/`

Movement data

Apache ACTIVEMQ

A train's movements
- Arrived / departed
- Where
- What time
- Variation from timetable

A train was cancelled
- Where
- When
- Why

A train was 'activated'
- Links a train to a schedule

Ingest ActiveMQ with Kafka Connect

```
"connector.class"         : "io.confluent.connect.activemq.ActiveMQSourceConnector",
"activemq.url"            : "tcp://datafeeds.networkrail.co.uk:61619",
"jms.destination.type": "topic",
"jms.destination.name": "TRAIN_MVT_EA_TOC",
"kafka.topic"             : "networkrail_TRAIN_MVT",
```

* Remember them?

```
{
    "messageID": "ID:opendata-backend.rockshore.net-44604-1564338858341-11:518:2:35:43470",
    "messageType": "text",
    "timestamp": 1565864716708,
    "deliveryMode": 2,
    "correlationID": null,
    "replyTo": null,
    "destination": {
        "destinationType": "topic",
        "name": "TRAIN_MVT_EM_TOC"
    },
    "redelivered": false,
    "type": null,
    "expiration": 1565865016708,
    "priority": 4,
    "properties": {},
    "bytes": null,
    "map": null,
    "text": "[{\"header\":{\"msg_type\":\"0003\",\"source_dev_id\":\"\",\"user_id\":\"\",\"ori
":\"TRUST\"},\"body\":{\"event_type\":\"DEPARTURE\",\"gbtt_timestamp\":\"\",\"original_loc_sta
_loc_timestamp\":\"\",\"current_train_id\":\"\",\"delay_monitoring_point\":\"false\",\"next_re
,\"correction_ind\":\"false\",\"event_source\":\"AUTOMATIC\",\"train_file_address\":null,\"pla
F15\",\"offroute_ind\":\"false\",\"variation_status\":\"ON TIME\",\"train_service_code\":\"221
nd\":\"DOWN\",\"route\":\"2\",\"planned_event_type\":\"DEPARTURE\",\"next_report_stanox\":\"27
_id\":\"\",\"original_data_source\":\"SMART\",\"msg_queue_timestamp\":\"1565864707000\",\"sour
original_loc_stanox\":\"\",\"planned_timestamp\":\"1565868150000\",\"timetable_variation\":\"
rue\",\"next_report_run_time\":\"2\",\"reporting_stanox\":\"62400\",\"actual_timestamp\":\"156
ss\":null,\"platform\":\" 5\",\"division_code\":\"28\",\"train_terminated\":\"false\",\"train_
e_code\":\"22150000\",\"toc_id\":\"28\",\"loc_stanox\":\"62400\",\"auto_expected\":\"true\",\'
_stanox\":\"62209\",\"line_ind\":\"F\"}},{\"header\":{\"msg_type\":\"0003\",\"source_dev_id\":
07000\",\"source_system_id\":\"TRUST\"},\"body\":{\"event_type\":\"DEPARTURE\",\"gbtt_timestam
ariation\":\"0\",\"original_loc_timestamp\":\"\",\"current_train_id\":\"\",\"delay_monitoring_
estamp\":\"1565868300000\",\"correction_ind\":\"false\",\"event_source\":\"AUTOMATIC\",\"trair
e\",\"train_id\":\"572N31MH15\",\"offroute_ind\":\"false\",\"variation_status\":\"ON TIME\",\'
d\":\"true\",\"direction_ind\":\"DOWN\",\"route\":\"1\",\"planned_event_type\":\"DEPARTURE\",\
ource_dev_id\":\"\",\"user_id\":\"\",\"original_data_source\":\"SMART\",\"msg_queue_timestamp\
gbtt_timestamp\":\"\",\"original_loc_stanox\":\"\",\"planned_timestamp\":\"1565868150000\",\"t
```


Envelope

Payload

Message transformation

Extract element

Convert JSON

Explode Array

```
kafkacat -b localhost:9092 -G tm_explode
  networkrail_TRAIN_MVT -u | \
 jq -c '.text|fromjson[]' | \
 kafkacat -b localhost:9092 -t
  networkrail_TRAIN_MVT_X -T -P
```

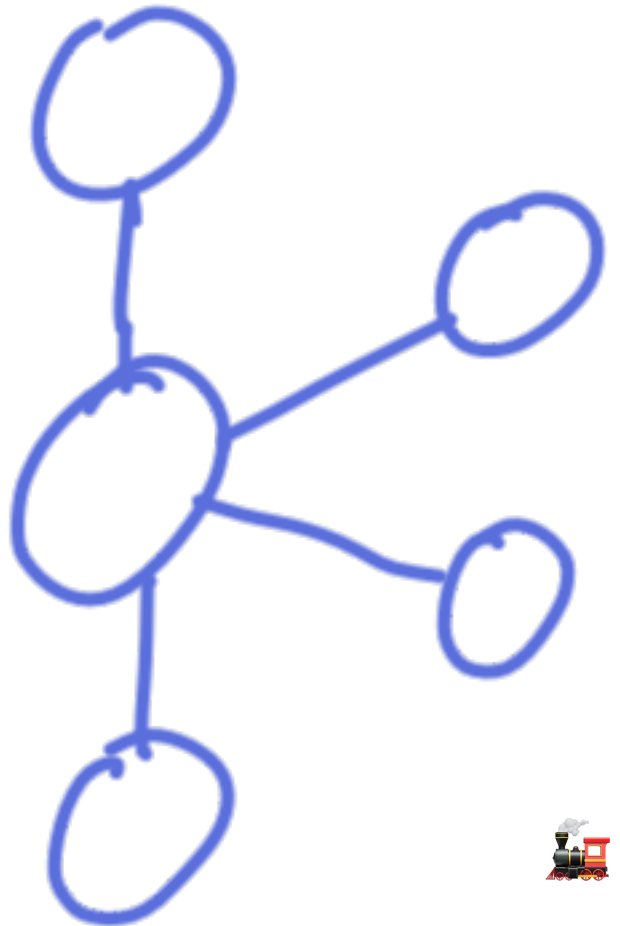

```
{
    "header": {
        "msg_type": "0003",
        "source_dev_id": "",
        "user_id": "",
        "original_data_source": "SMART",
        "msg_queue_timestamp": "1558694355000",
        "source_system_id": "TRUST"
    },
    "body": {
        "event_type": "DEPARTURE",
        "gbtt_timestamp": "",
        "original_loc_stanox": "",
        "planned_timestamp": "1558697700000",
        "timetable_variation": "4",
        "original_loc_timestamp": "",
        "current_train_id": "",
        "delay_monitoring_point": "false",
        "next_report_run_time": "4",
        "reporting_stanox": "00000",
        "actual_timestamp": "1558697940000",
        "correction_ind": "false",
        "event_source": "AUTOMATIC"
```

Schedule data

Train routes
Type of train (diesel/electric)
Classes, sleeping accomodation, reservations, etc

Load JSON from S3 into Kafka

```
curl -s -L -u "$USER:$PW" "$FEED_URL" | \
  gunzip | \
  kafkacat -b localhost -P -t CIF_FULL_DAILY
```



Reference data

OpenCage

JSON

Train company names
Location codes
Train type codes
Location geocodes

Geocoding

Type of geocoding ◉ Forward ○ Reverse

Address or place name ILKLEY

API Key YOUR-API-KEY

Geocode

Results Map Request URL JSON Response

https://api.opencagedata.com/geocode/v1/json?q=ILKLEY&key=YOUR-API-KEY&language=en&pretty=1

Results for "ILKLEY"

Results Map

Summary (see ful

1. 🇬🇧 Ilkley
53.9257254

2. 🇬🇧 Ilkley,
53.9247929

Results Map Request URL JSON Response


```
HTTP Status code: 200

HTTP Headers:
  Content-Type: application/json; charset=utf-8
  Access-control-allow-origin: *
  X-RateLimit-Limit: 15000
  X-RateLimit-Remaining: 14934
  X-RateLimit-Reset: 1568246400

Content:

{
  "documentation": "https://opencagedata.com/api",
  "licenses": [
    {
      "name": "see attribution guide",
      "url": "https://opencagedata.com/credits"
    }
  ],
  "rate": {
    "limit": 15000,
    "remaining": 14934,
    "reset": 1568246400
  },
  "results": [
    {
      "annotations": {
        "DMS": {
          "lat": "53° 55' 32.61144'' N",
          "lng": "1° 49' 27.98760'' W"
```

Static reference data

Page | Discussion

Main page
Recent changes
Random page
Help

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information
Cite this page

TOC Codes

This table lists the codes used to identify Train Operating Companie

- *Company Name* is the name of the Train Operating Company
- *Business Code* is a two-character code used within the TRUST
- *Sector Code* is the sector code used to represent the TOC in TF
- *ATOC Code* is a two-character code used to representing the T(exception of GB Railfreight services).
- *Whitelisted* means that the sector code is 'whitelisted' and TRUS published for these operators with sector code *00*.

Company Name	Business Code
Alliance Rail	ZB
Northern	ED
Transport for Wales	HL
c2c	HT
Caledonian Sleeper	ES
Chiltern Railway	HO

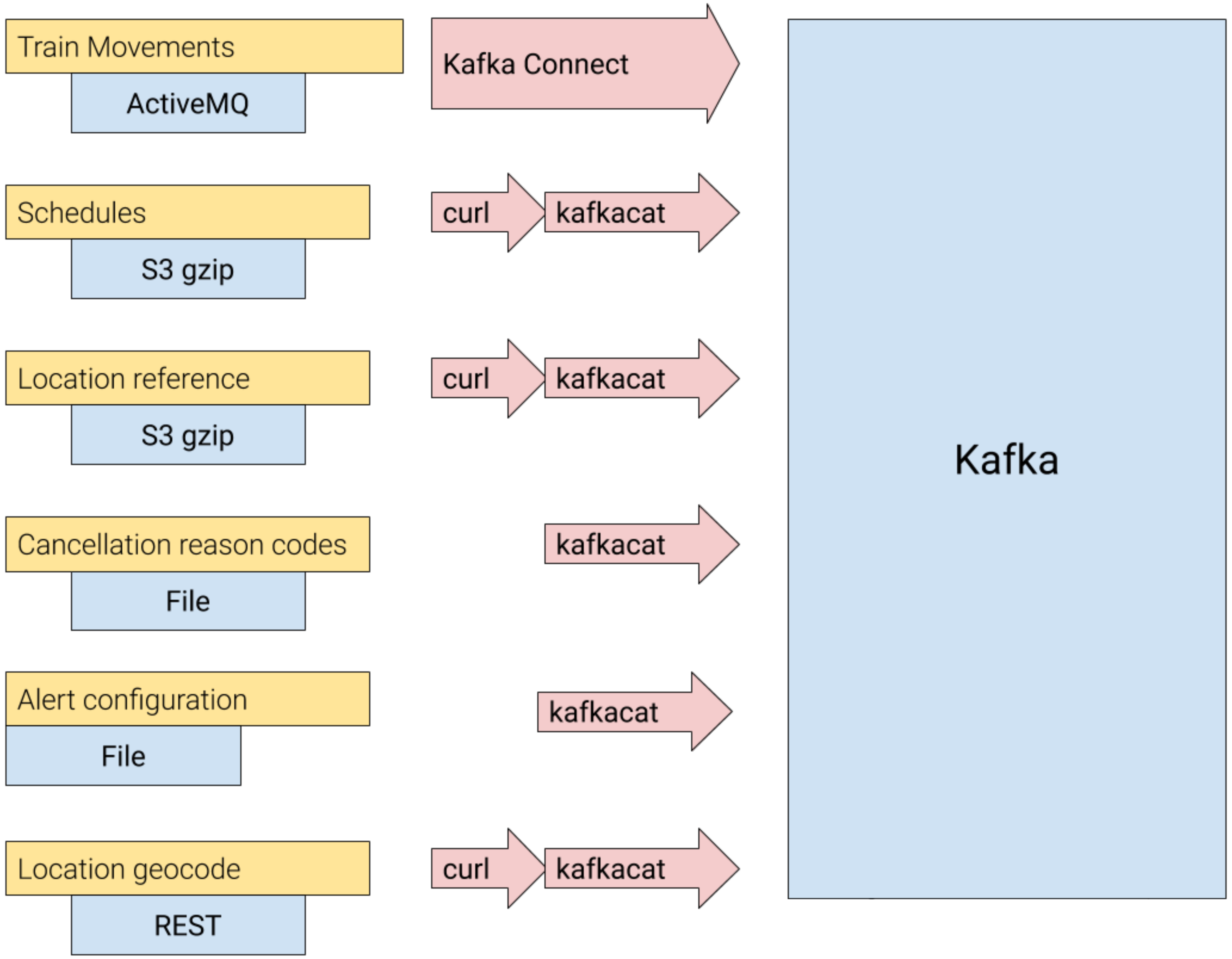
Code	Cause	Abbreviation
AA	Waiting acceptance into off Network Terminal or Yard	ACCEPTANCE
AC	Waiting train preparation or completion of TOPS list/RT3973	TRAIN PREP
AD	Off Network Terminal or Yard staff shortage including reactionary congestion caused by the shortage	WTG STAFF
AE	Congestion in off Network Terminal or Yard	CONGESTION
AG	Wagon load incident including adjusting loads or open door	LOAD INCDT
AH	Customer or off Network yard equipment breakdown/reduced capacity	YARD EQUIP
AJ	Waiting Customer's traffic including documentation	TRAFFIC
AK	Safety incidents and mishaps (e.g. derailments, fire or chemical spill) in off Network Rail network freight yard or terminal (including private sidings, and stations – where it affects FOC services)	YARDSAFETY
AX	Failure of off network infrastructure (FOC or private)	OFFNET INF
AZ	Other Freight Operating Company. Cause to be specified, including mishaps in off Network terminals or yards	YARD OTHER
DA	Non Technical Fleet Holding Code	HOLD NT FL
DB	Train Operations Holding Code	HOLD T-OPS
DC	Train Crew Causes Holding Code	HOLD T-CRW
DD	Technical Fleet Holding Code	HOLD T FL
DE	Station Causes Holding Code	HOLD STN
DF	External Causes Holding Code	HOLD EXT

```
AA:{"canx_reason_code":"AA","canx_reason":"Waiting acceptance into off Network Terminal or Yard"}
AC:{"canx_reason_code":"AC","canx_reason":"Waiting train preparation or completion of TOPS list/RT3973"}
AE:{"canx_reason_code":"AE","canx_reason":"Congestion in off Network Terminal or Yard"}
AG:{"canx_reason_code":"AG","canx_reason":"Wagon load incident including adjusting loads or open door"}
AJ:{"canx_reason_code":"AJ","canx_reason":"Waiting Customer’s traffic including documentation"}
DA:{"canx_reason_code":"DA","canx_reason":"Non Technical Fleet Holding Code"}
DB:{"canx_reason_code":"DB","canx_reason":"Train Operations Holding Code"}
DC:{"canx_reason_code":"DC","canx_reason":"Train Crew Causes Holding Code"}
DD:{"canx_reason_code":"DD","canx_reason":"Technical Fleet Holding Code"}
DE:{"canx_reason_code":"DE","canx_reason":"Station Causes Holding Code"}
DF:{"canx_reason_code":"DF","canx_reason":"External Causes Holding Code"}
DG:{"canx_reason_code":"DG","canx_reason":"Freight Terminal and or Yard Holding Code"}
DH:{"canx_reason_code":"DH","canx_reason":"Adhesion and or Autumn Holding Code"}
FA:{"canx_reason_code":"FA","canx_reason":"Dangerous goods incident"}
FC:{"canx_reason_code":"FC","canx_reason":"Freight train driver"}
```

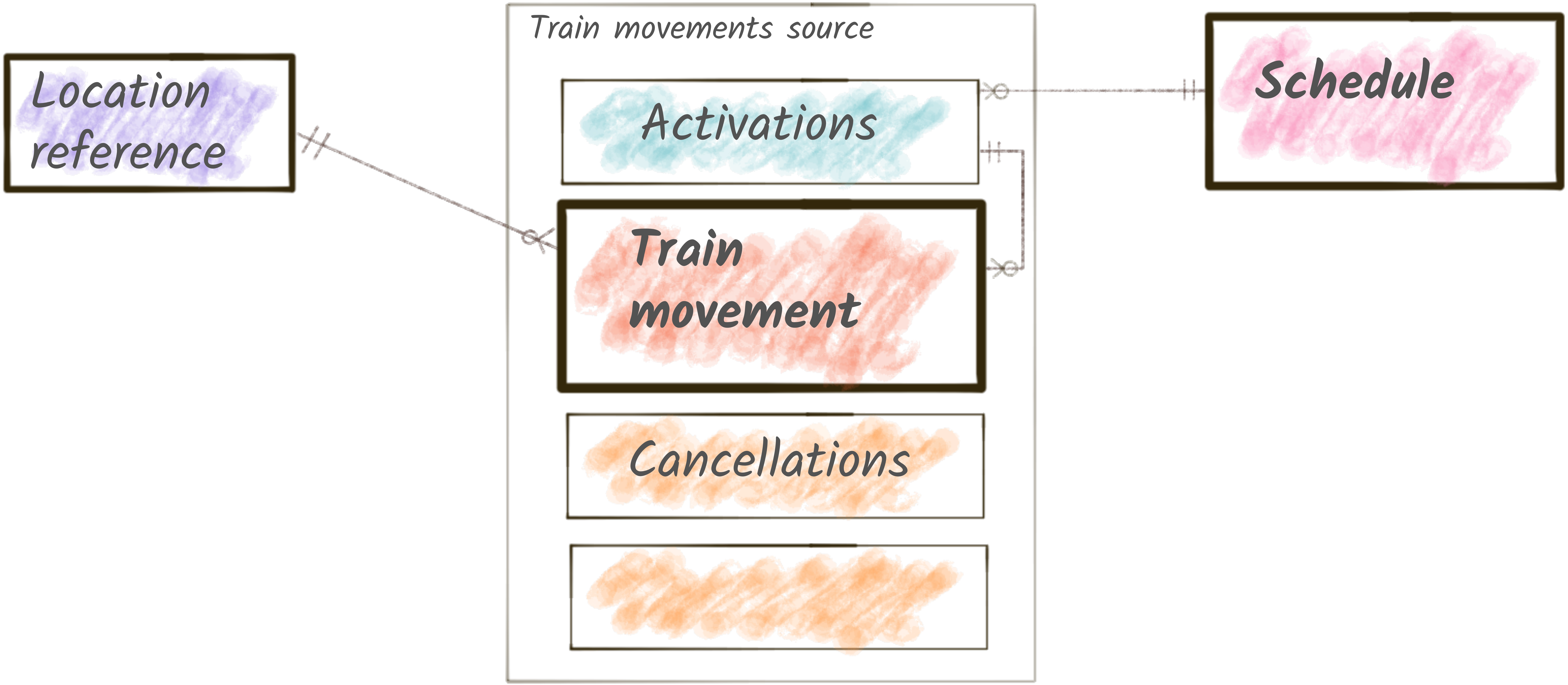

Flat file

```
kafkacat -l canx_reason_code.dat
   -b localhost:9092 -t canx_reason_code -P -K:
```

Kafka


Train Movements
ActiveMQ
Kafka Connect
Schedules
S3 gzip
curl
kafkacat
Location reference
S3 gzip
curl
kafkacat
Cancellation reason codes
File
kafkacat
Alert configuration
File
kafkacat
Location geocode
REST
curl
kafkacat
Kafka

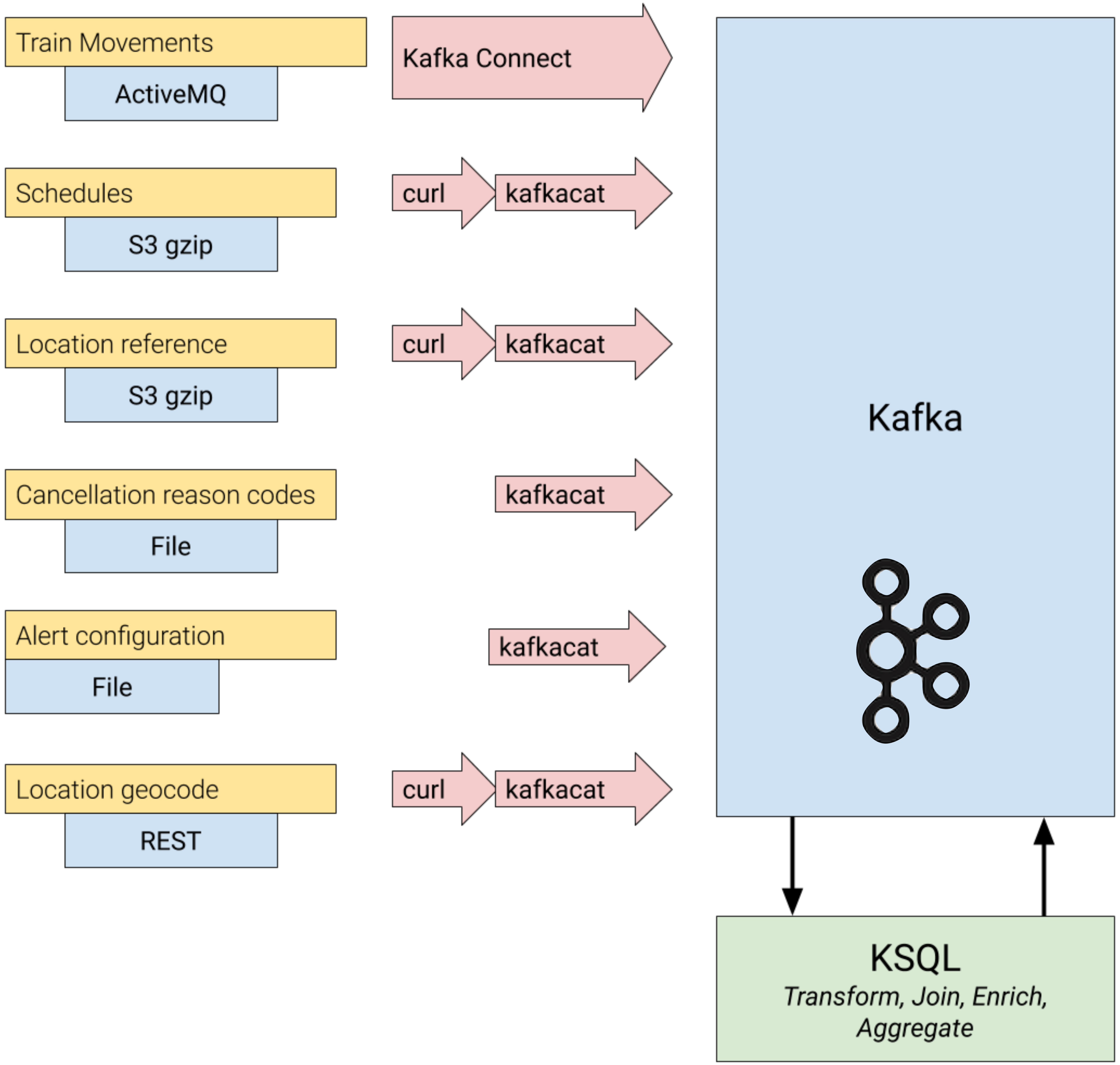
Transforming the data


Location reference
Train movements source
Activations
Train movement
Cancellations
Schedule

Streams of events

Stream Processing with KSQL

Stream Processing with KSQL


Train Movements
ActiveMQ
Kafka Connect
Schedules
S3 gzip
curl
kafkacat
Location reference
S3 gzip
curl
kafkacat
Cancellation reason codes
File
kafkacat
Alert configuration
File
kafkacat
Location geocode
REST
curl
kafkacat
Kafka
KSQL
Transform, Join, Enrich, Aggregate

Event data

ActiveMQ

Kafka Connect

NETWORKRAIL_TRAIN_MVT_X

Activation
Activation
Cancellation
Movement
Movement
Movement
Cancellation
Movement
Movement

Train Movement

{ "header": {
    "msg_type": "0003",
    "user_id": "#QHPA026",
    "original_
    "msg_queue
    "source_sy
},
"body": {
    "event_typ
    "gbtt_time
    "original_
    "planned_t
    "timetable
    "original_
    "current_t
    "delay_mor
    "next_repo
    "reporting
    "actual_ti
    "correctio
    "event_sou
    "train_fil
    "platform"
    "division_
    "train_ter
    "train_id"
    "offroute_

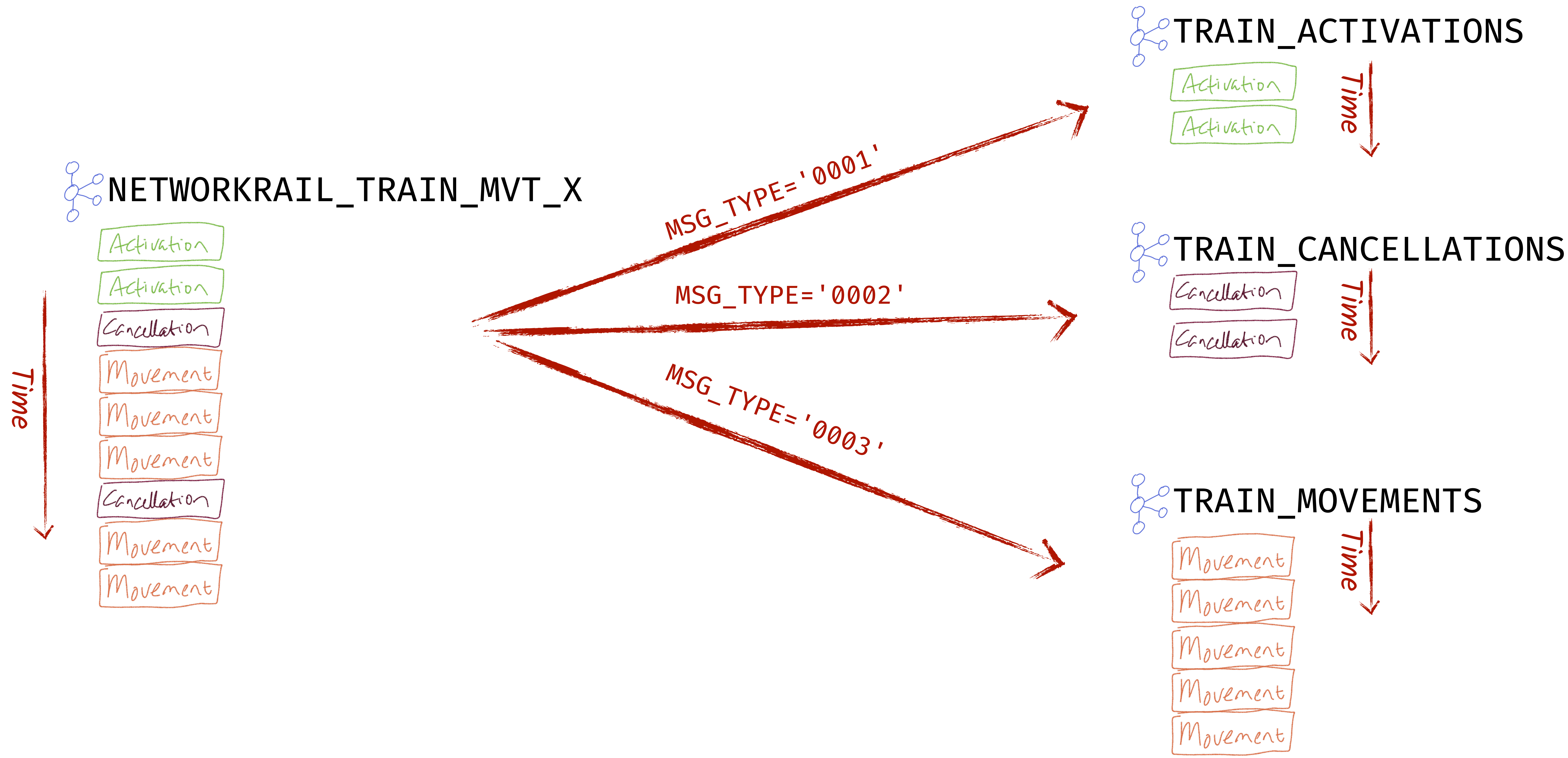
Train Cancellation

{ "header": {
    "msg_type": "0002",
    "source_dev_id": "",
    "user_id": "",
    "origina
    "msg_que
    "source_
},
"body": {
    "train_f
    "train_s
    "orig_lo
    "toc_id"
    "dep_tim
    "divisio
    "loc_sta
    "canx_ti
    "canx_re
    "train_i
    "orig_lo
    "canx_ty
}
}

Train Activation

"header": {
    "msg_type": "0001",
    "user_id": "",
    "original_data_source": "TSIA",
    "msg_queue_timestamp": "1511528234000",
    "source_system_id": "TRUST"
},
"body": {
    "schedule_source": "C",
    "train_file_address": null,
    "schedule_end_date": "2017-12-08",
    "train_id": "775F25MP24",
    "tp_origin_timestamp": "2017-11-24",
    "creation_timestamp": "1511528234000",
    "tp_origin_stanox": "",
    "origin_dep_timestamp": "1511535420000",
    "train_service_code": "25470001",
    "toc_id": "25",

Message routing with KSQL



Split a stream of data (message routing)

NETWORKRAIL_TRAIN_MVT_X

Source topic

```sql
CREATE STREAM TRAIN_MOVEMENTS_00
AS
SELECT *
  FROM NETWORKRAIL_TRAIN_MVT_X
WHERE header->msg_type = '0003';
```

Movement events

MSG_TYPE='0003'

```sql
CREATE STREAM TRAIN_CANCELLATIONS_00
AS
SELECT *
  FROM NETWORKRAIL_TRAIN_MVT_X
 WHERE header->msg_type = '0002';
```

Cancellation events

MSG_TYPE='0002'

Joining events to lookup data (stream-table joins)

TS = 10:00:01
LOC ID=42

TS=10:00:02
LOC ID=43

TRAIN_MOVEMENTS
(Stream)

LOCATION
(Table)

ID	DESCRIPTION
42	MENSTON
43	ILKLEY

```
CREATE STREAM TRAIN_MOVEMENTS_ENRICHED AS
SELECT  TM.LOC_ID       AS LOC_ID,
        TM.TS           AS TS,
        L.DESCRIPTION   AS LOC_DESC,
        […]
   FROM TRAIN_MOVEMENTS  TM
        LEFT JOIN LOCATION L
        ON TM.LOC_ID = L.ID;
```

TS = 10:00:01
LOC ID=42
LOC DESC=MENSTON

TS=10:00:02
LOC ID=43
LOC DESC=ILKLEY

TRAIN_MOVEMENTS_ENRICHED
(Stream)

Decode values, and generate composite key

```
CREATE STREAM SCHEDULE_00 AS
SELECT JsonScheduleV1->CIF_train_uid + '/'
        + JsonScheduleV1->schedule_start_date + '/'
        + JsonScheduleV1->CIF_stp_indicator AS SCHEDULE_KEY,
       JsonScheduleV1->atoc_code AS atoc_code,
       CASE
           WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'D' THEN  'Diesel'
           WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'E' THEN  'Electric'
           WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'ED' THEN 'Electro-Diesel'
           WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'EMU' THEN
                                                              'Electric Multiple Unit'
           WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'HST' THEN
                                                                   'High Speed Train'
         END AS POWER_TYPE
        […]
FROM SCHEDULE_RAW;
```

Concatenate columns

Decode CIF_power_type value

Decode values, and generate composite key

```
+----------------+------------------+----------+-----------------------+-------------------+-------------------------+
| CIF_train_uid | sched_start_dt | stp_ind | SCHEDULE_KEY         | CIF_power_type | POWER_TYPE             |
+----------------+------------------+----------+-----------------------+-------------------+-------------------------+
| Y82535        | 2019-05-20     | P       | Y82535/2019-05-20/P | E              | Electric               |
| Y82537        | 2019-05-20     | P       | Y82537/2019-05-20/P | DMU            | Other                  |
| Y82542        | 2019-05-20     | P       | Y82542/2019-05-20/P | D              | Diesel                 |
| Y82535        | 2019-05-24     | P       | Y82535/2019-05-24/P | EMU            | Electric Multiple Unit |
| Y82542        | 2019-05-24     | P       | Y82542/2019-05-24/P | HST            | High Speed Train       |
```

```
SELECT JsonScheduleV1->CIF_train_uid + '/'
         + JsonScheduleV1->schedule_start_date + '/'
         + JsonScheduleV1->CIF_stp_indicator AS SCHEDULE_KEY,
```

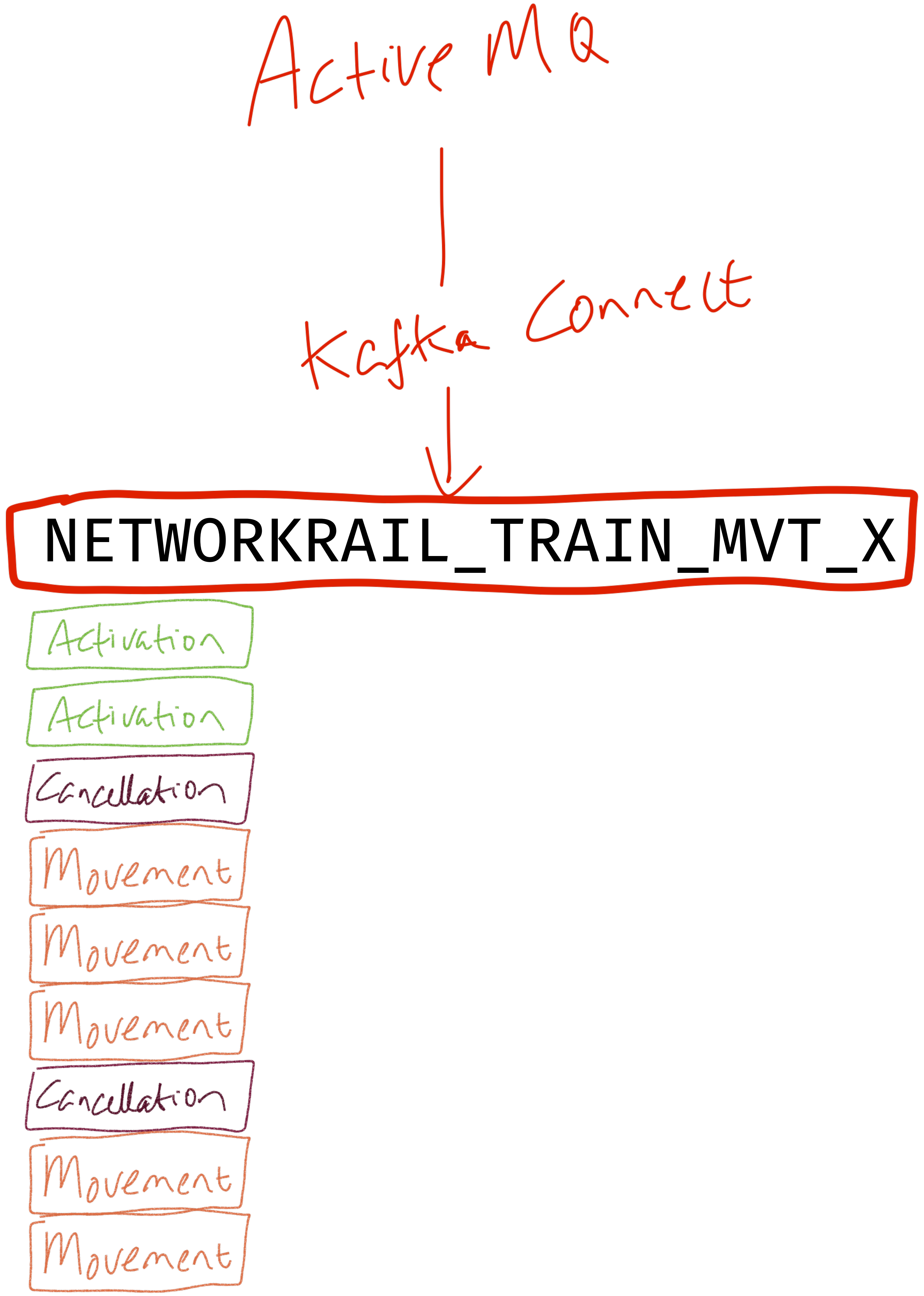
```
CASE
 WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'D'
   THEN  'Diesel'
 WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'E'
   THEN  'Electric'
 WHEN JsonScheduleV1->schedule_segment->CIF_power_type = 'ED'
   THEN 'Electro-Diesel'
END AS POWER_TYPE
```

Enrich columns

EVENT_TYPE	PLATFORM	TIMETABLE_VARIA...	VARIATION_STATUS	VARIATION
ARRIVAL		1	LATE	1 MINS LATE
DEPARTURE		10	LATE	10 MINS LATE
DEPARTURE	Platform 3	0	ON TIME	ON TIME
DEPARTURE		4	EARLY	4 MINS EARLY
DEPARTURE		1	EARLY	1 MINS EARLY
DEPARTURE	Platform 4	1	EARLY	1 MINS EARLY

```
CREATE STREAM TRAIN_MOVEMENTS_01
SELECT  TM.EVENT_TYPE AS EVENT_TYPE,
        CAST(TM.TIMETABLE_VARIATION AS INT) AS TIMETABLE_VARIATION,
        CASE WHEN LEN(TM.PLATFORM)> 0 THEN 'Platform' + TM.PLATFORM
             ELSE ''
          END AS PLATFORM,
        CASE WHEN TM.VARIATION_STATUS = 'ON TIME' THEN 'ON TIME'
             WHEN TM.VARIATION_STATUS = 'LATE' THEN TM.TIMETABLE_VARIATION +
                                                     ' MINS LATE'
             WHEN TM.VARIATION_STATUS='EARLY' THEN TM.TIMETABLE_VARIATION +
```

Schemas



Train Movement


```
{ "header": {
        "msg_type": "0003",
        "source_dev_id": "VLA5",
        "user_id": "#QHPA026",
        "original_
        "msg_queue
        "source_sy
    },
    "body": {
        "event_typ
        "gbtt_time
        "original_
        "planned_t
        "timetable
        "original_
        "current_t
        "delay_mon
        "next_repo
        "reporting
        "actual_ti
        "correctic
        "event_sou
        "train_fil
        "platform"
        "division_
        "train_ter
        "train_id"
        "offroute_
```


Train Cancellation


```
{ "header": {
        "msg_type": "0002",
        "source_dev_id": "",
        "user_id": "",
        "origina
        "msg_que
        "source_
    },
    "body": {
        "train_f
        "train_s
        "orig_lo
        "toc_id"
        "dep_tim
        "divisio
        "loc_sta
        "canx_ti
        "canx_re
        "train_i
        "orig_lo
        "canx_ty
    }
}
```


Train Activation


```
{ "header": {
        "msg_type": "0001",
        "source_dev_id": "",
        "user_id": "",
        "original_data_source": "TSIA",
        "msg_queue_timestamp": "1511528234000",
        "source_system_id": "TRUST"
    },
    "body": {
        "schedule_source": "C",
        "train_file_address": null,
        "schedule_end_date": "2017-12-08",
        "train_id": "775F25MP24",
        "tp_origin_timestamp": "2017-11-24",
        "creation_timestamp": "1511528234000",
        "tp_origin_stanox": "",
        "origin_dep_timestamp": "1511535420000",
        "train_service_code": "25470001",
        "toc_id": "25",
```

Handling JSON data & mixed schemas

```
CREATE STREAM NETWORKRAIL_TRAIN_MVT_X (
  header STRUCT< msg_type VARCHAR, […] >,
  body VARCHAR)
  WITH (KAFKA_TOPIC='networkrail_TRAIN_MVT_X',
            VALUE_FORMAT='JSON');
```

Source

Derived Stream

```
CREATE STREAM TRAIN_CANCELLATIONS_00 AS
  SELECT HEADER,
         EXTRACTJSONFIELD(body,'$.canx_timestamp')   AS canx_timestamp,
         EXTRACTJSONFIELD(body,'$.canx_reason_code') AS canx_reason_code,
         […]
   FROM networkrail_TRAIN_MVT_X
  WHERE header->msg_type = '0002';
```

Declared Schema

Serialisation & Schemas

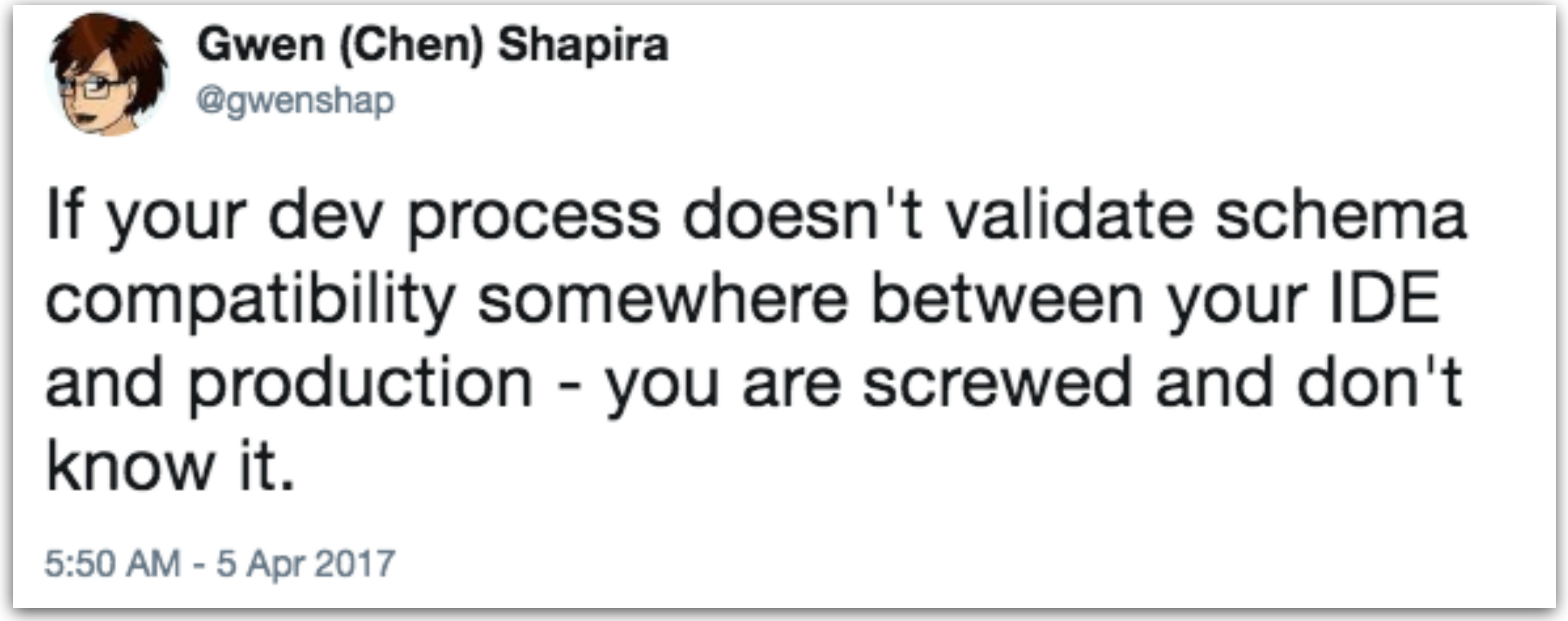
Avro

-> Confluent Schema Registry

Protobuf

JSON

CSV


Gwen (Chen) Shapira
@gwenshap

If your dev process doesn't validate schema compatibility somewhere between your IDE and production - you are screwed and don't know it.

5:50 AM - 5 Apr 2017

https://qconnewyork.com/system/files/presentation-slides/qcon_17_-_schemas_and_apis.pdf

The Confluent Schema Registry

Reserialise data

```
CREATE STREAM TRAIN_CANCELLATIONS_00
  WITH (VALUE_FORMAT='AVRO') AS
  SELECT EXTRACTJSONFIELD(body,'$.canx_timestamp')
    AS canx_timestamp,
         EXTRACTJSONFIELD(body,'$.canx_reason_code')
    AS canx_reason_code,
         […]
    FROM networkrail_TRAIN_MVT_X
  WHERE header->msg_type = '0002';
```

Flatten and re-key a stream


```
{
  "TiplocV1": {
    "transaction_type": "Create",
    "tiploc_code": "ABWD",
    "nalco": "513100",
    "stanox": "88601",
    "crs_code": "ABW",
    "description": "ABBEY WOOD",
    "tps_description": "ABBEY WOOD"
  }
}
```


Access nested element

```
CREATE STREAM TIPLOC_FLAT_KEYED
SELECT  TiplocV1→TRANSACTION_TYPE  AS TRANSACTION_TYPE ,
        TiplocV1→TIPLOC_CODE       AS TIPLOC_CODE ,
        TiplocV1→NALCO             AS NALCO ,
        TiplocV1→STANOX            AS STANOX ,
        TiplocV1→CRS_CODE          AS CRS_CODE ,
        TiplocV1→DESCRIPTION       AS DESCRIPTION ,
        TiplocV1→TPS_DESCRIPTION   AS TPS_DESCRIPTION
FROM    SCHEDULE_RAW
PARTITION BY TIPLOC_CODE;
```

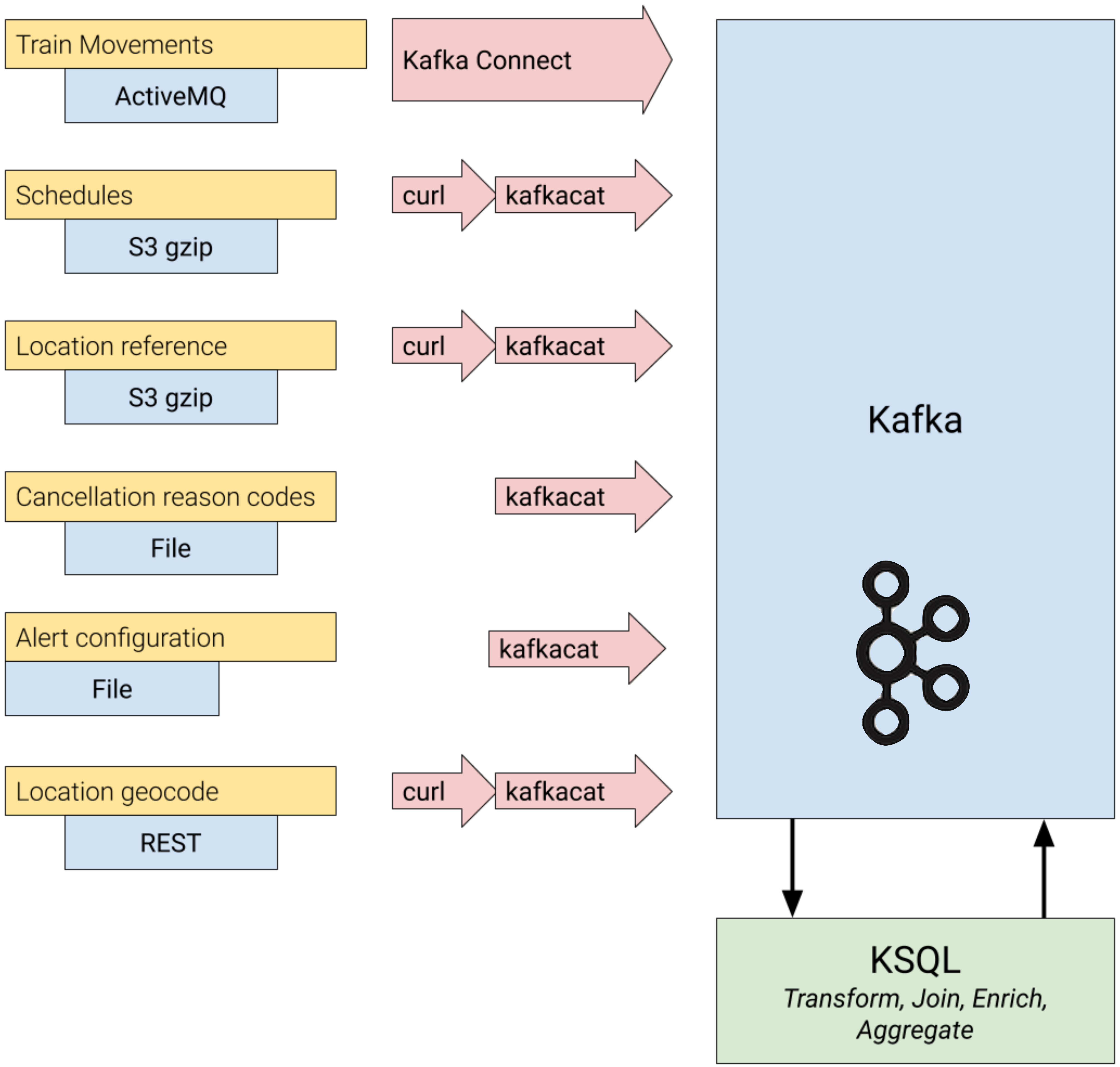
Change partitioning key

```
ksql> DESCRIBE TIPLOC_FLAT_KEYED;

Name                 : TIPLOC_FLAT_KEYED
 Field               | Type
---------------------------------------------
 TRANSACTION_TYPE    | VARCHAR(STRING)
 TIPLOC_CODE         | VARCHAR(STRING)
 NALCO               | VARCHAR(STRING)
 STANOX              | VARCHAR(STRING)
 CRS_CODE            | VARCHAR(STRING)
 DESCRIPTION         | VARCHAR(STRING)
 TPS_DESCRIPTION     | VARCHAR(STRING)
---------------------------------------------
```


ActiveMQ
networkrail_TRAIN_MVT
Cron every 5 minutes to restart any intermittent connector failures
jq to extract payload and explode array of message batch
curl
CORPUS_RAW
corpus
Reserialise to Avro
Partition by STANOX
CORPUS_BY_STANOX
CORPUS_BY_STANOX
STANOX
CORPUS_BY_STANOX
NETWORKRAIL_TRAIN_MVT_X
networkrail_TRAIN_MVT_X
Reserialise to Avro
Select Movement records
(msg_type='0003')
Flatten body/header payload
Reserialise to Avro
Select Activation records
(msg_type='0001')
Flatten body/header payload
Reserialise to Avro
Select Cancellation records
(msg_type='0002')
Flatten body/header payload
TRAIN_MOVEMENTS_00
TRAIN_MOVEMENTS_00
TRAIN_ACTIVATIONS_00
TRAIN_ACTIVATIONS_00
TRAIN_CANCELLATIONS_00
TRAIN_CANCELLATIONS_00
Join to location
Decode TOC
Enrich platform/delay fields
Set message timestamp to event time
Join to location
Join to location
Decode TOC
TRAIN_MOVEMENTS_01
TRAIN_MOVEMENTS_01
TRAIN_ACTIVATIONS_01
TRAIN_ACTIVATIONS_01
TRAIN_CANCELLATIONS_01
TRAIN_CANCELLATIONS_01
SCHEDULE_RAW
Curl
via cron @ 0700
CIF_FULL_DAILY
Reserialise to Avro
Select & flatten schedule records
Decode fields
Reserialise to Avro
Select & flatten tiploc records
Partition by TIPLOC_CODE
SCHEDULE_00
SCHEDULE_00
TIPLOC_FLAT_KEYED
TIPLOC_FLAT_KEYED
TIPLOC
TIPLOC_FLAT_KEYED
Join to location
SCHEDULE_01
SCHEDULE_01
SCHEDULE_01_T
SCHEDULE_01
CANX_REASON
canx_reason_code
file
Join to activations
TRAIN_MOVEMENTS_ACTIVATIONS_00
TRAIN_MOVEMENTS_ACTIVATIONS_00
Join cancellations to cancellation reason
TRAIN_CANCELLATIONS_02
TRAIN_CANCELLATIONS_02
Join to schedule
TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00
TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00

Using the data

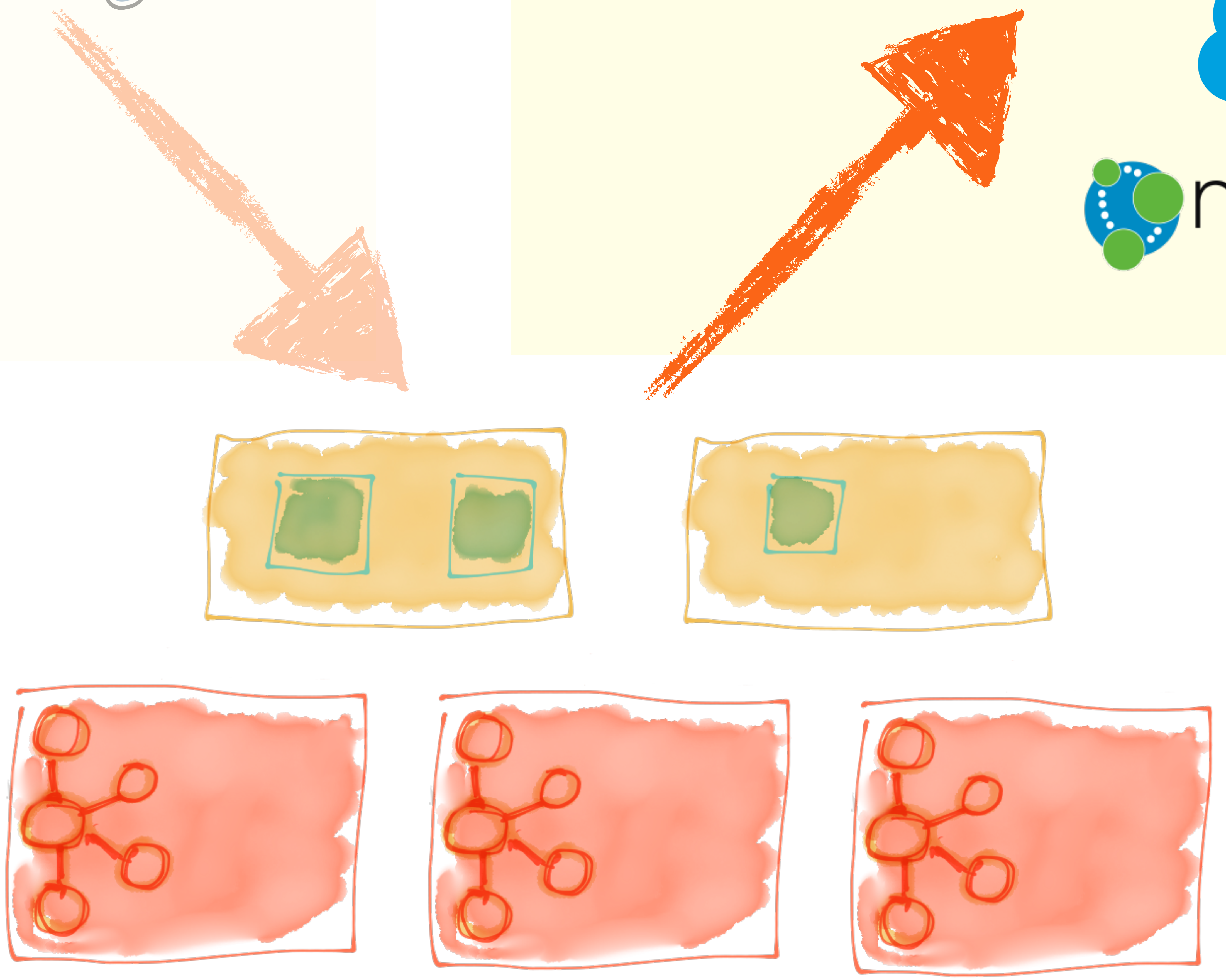

Train Movements
ActiveMQ
Kafka Connect
Schedules
S3 gzip
curl
kafkacat
Location reference
S3 gzip
curl
kafkacat
Cancellation reason codes
File
kafkacat
Alert configuration
File
kafkacat
Location geocode
REST
curl
kafkacat
Kafka
KSQL
Transform, Join, Enrich, Aggregate


Train Movements
ActiveMQ
Schedules
S3 gzip
Location reference
S3 gzip
Cancellation reason codes
File
Alert configuration
File
Location geocode
REST
Kafka Connect
curl
kafkacat
curl
kafkacat
kafkacat
kafkacat
curl
kafkacat
Kafka
KSQL
Transform, Join, Enrich, Aggregate
Kafka Connect
Elasticsearch
Kibana
Real-time visualisation & exploration
Kafka Connect
S3 bucket
Athena
Ad-hoc & advanced analytics
Long-term store
Kafka Connect
PostgreSQL
Kafka Connect
Telegram
Real-time alerts

Streaming Integration with Kafka Connect

JSON MQTT.ORG
MySQL ORACLE
syslog CSV salesforce
http://
Microsoft SQL Server

mongoDB Java JDBC Amazon S3 snowflake
elasticsearch influxdb IBM MQ Google BigQuery
salesforce http:// ORACLE
neo4j hadoop HDFS
splunk> MQTT.ORG



Kafka Connect

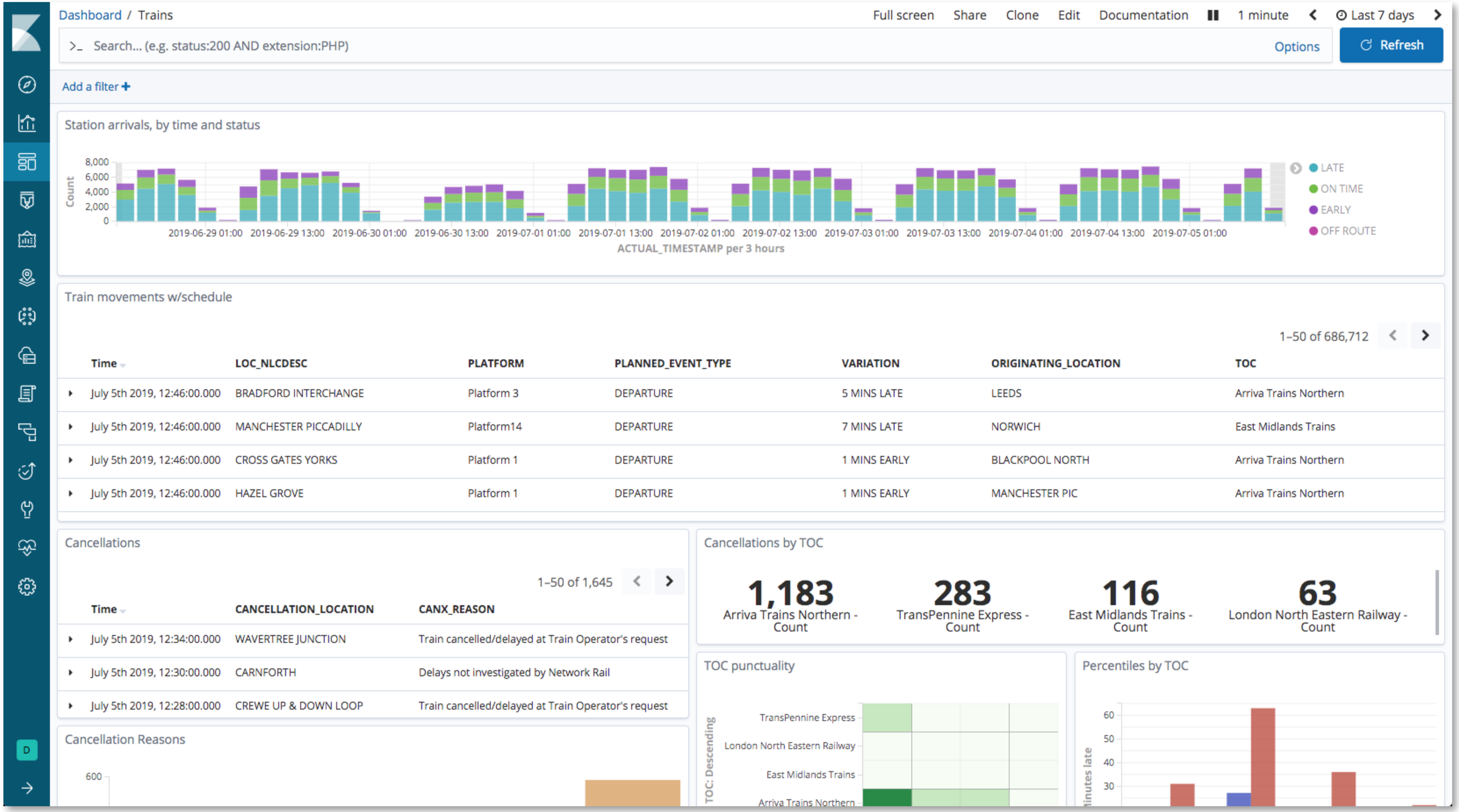
Kafka Brokers

Kafka -> Elasticsearch

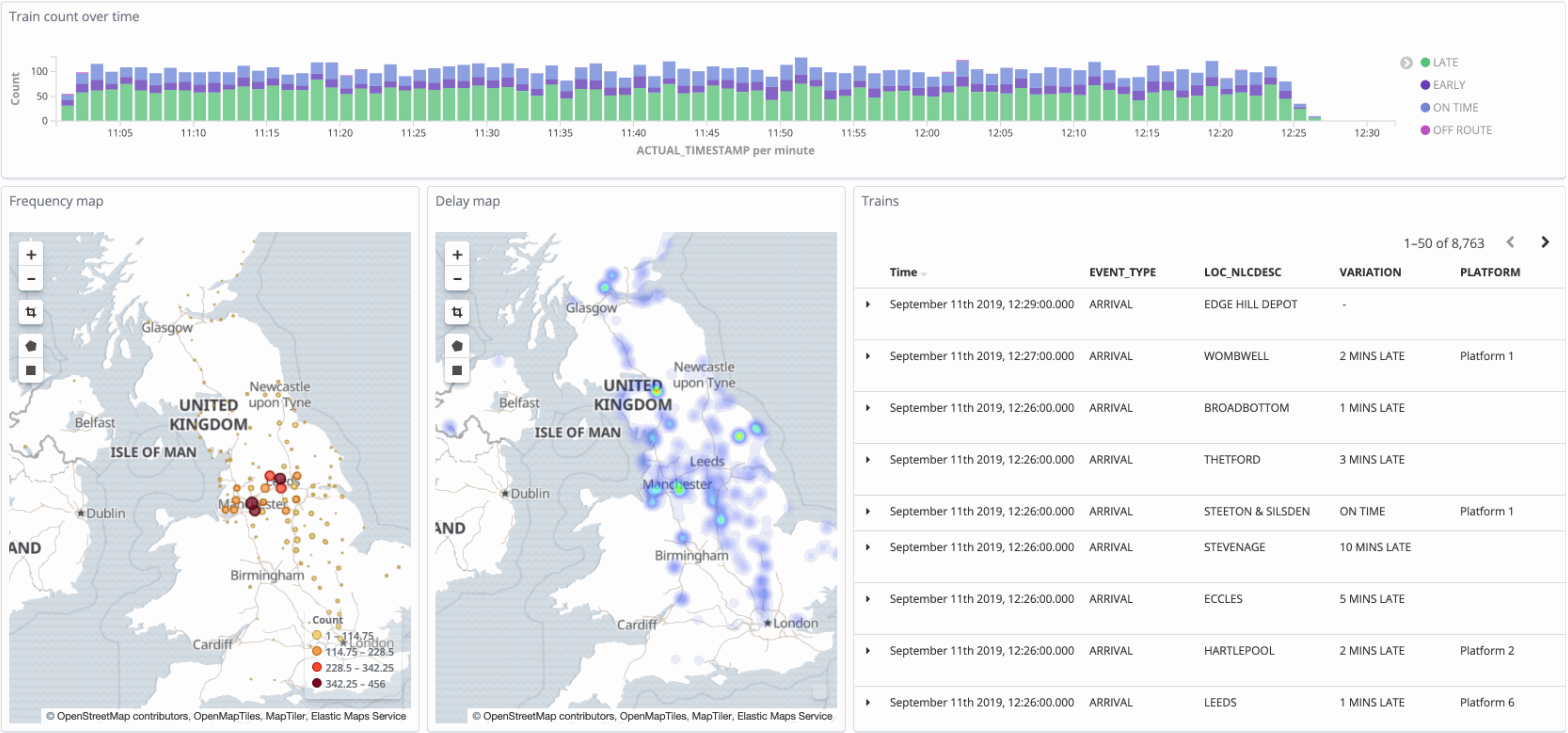
elasticsearch

```
{
    "connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",
    "topics": "TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00",
    "connection.url": "http://elasticsearch:9200",
    "type.name": "type.name=kafkaconnect",
    "key.ignore": "false",
    "schema.ignore": "true",
    "key.converter": "org.apache.kafka.connect.storage.StringConverter"
}
```

Kibana for real-time visualisation of rail data



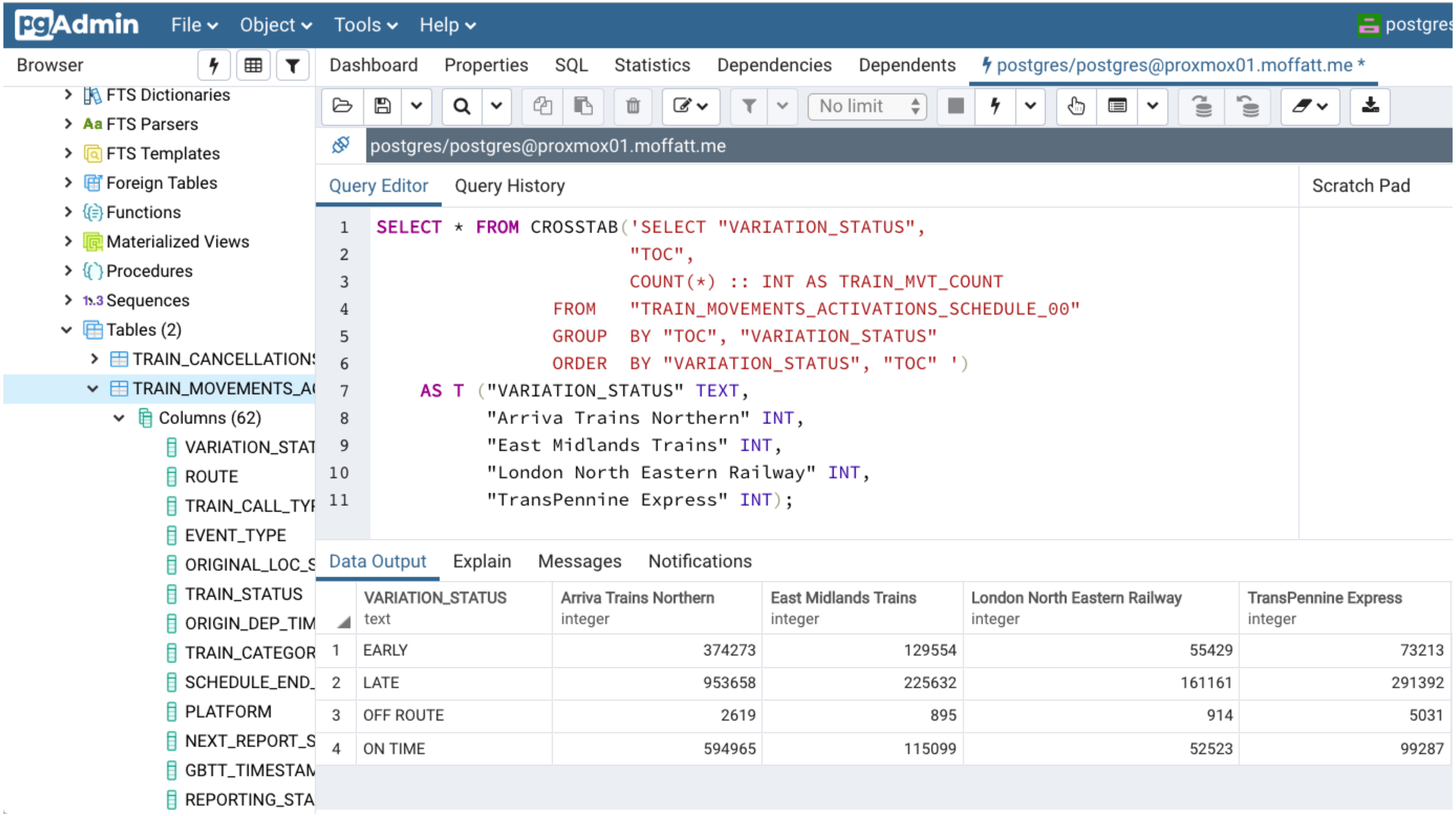
Kibana for real-time analysis of rail data



Kafka -> Postgres

```
{
  "connector.class": "io.confluent.connect.jdbc.JdbcSinkConnector",
  "key.converter": "org.apache.kafka.connect.storage.StringConverter",
  "connection.url": "jdbc:postgresql://postgres:5432/",
  "connection.user": "postgres",
  "connection.password": "postgres",
  "auto.create": true,
  "auto.evolve": true,
  "insert.mode": "upsert",
  "pk.mode": "record_key",
  "pk.fields": "MESSAGE_KEY",
  "topics": "TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00",
  "transforms": "dropArrays",
  "transforms.dropArrays.type": "org.apache.kafka.connect.transforms.ReplaceField$Value",
  "transforms.dropArrays.blacklist": "SCHEDULE_SEGMENT_LOCATION, HEADER"
}
```

Kafka -> Postgres




```sql
SELECT * FROM CROSSTAB('SELECT "VARIATION_STATUS",
                       "TOC",
                       COUNT(*) :: INT AS TRAIN_MVT_COUNT
                FROM   "TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00"
                GROUP  BY "TOC", "VARIATION_STATUS"
                ORDER  BY "VARIATION_STATUS", "TOC" ')
    AS T ("VARIATION_STATUS" TEXT,
          "Arriva Trains Northern" INT,
          "East Midlands Trains" INT,
          "London North Eastern Railway" INT,
          "TransPennine Express" INT);
```

	VARIATION_STATUS text	Arriva Trains Northern integer	East Midlands Trains integer	London North Eastern Railway integer	TransPennine Express integer
1	EARLY	374273	129554	55429	73213
2	LATE	953658	225632	161161	291392
3	OFF ROUTE	2619	895	914	5031
4	ON TIME	594965	115099	52523	99287

But do you even need a database?

TRAIN_MOVEMENTS

Movement
Movement
Movement
Movement
Movement

Time

Kafka

KSQL

```
SELECT TIMESTAMPTOSTRING(WINDOWSTART(),'yyyy-MM-dd') AS WINDOW_START_TS, VARIATION_STATUS,
        SUM(CASE WHEN TOC = 'Arriva Trains Northern'       THEN 1 ELSE 0 END) AS Arriva_CT,
        SUM(CASE WHEN TOC = 'East Midlands Trains'         THEN 1 ELSE 0 END) AS EastMidlands_CT,
        SUM(CASE WHEN TOC = 'London North Eastern Railway' THEN 1 ELSE 0 END) AS LNER_CT,
        SUM(CASE WHEN TOC = 'TransPennine Express'         THEN 1 ELSE 0 END) AS TransPennineExpress_CT
 FROM  TRAIN_MOVEMENTS WINDOW TUMBLING (SIZE 1 DAY)
GROUP BY VARIATION_STATUS;
```

Date	Variation	Arriva	East Mid	LNER	TPE
2019-07-02	OFF ROUTE	46	78	20	167
2019-07-02	ON TIME	19083	3568	1509	2916
2019-07-02	LATE	30850	7953	5420	9042
2019-07-02	EARLY	11478	3518	1349	2096
2019-07-03	OFF ROUTE	79	25	41	213
2019-07-03	ON TIME	19512	4247	1915	2936
2019-07-03	LATE	37357	8258	5342	11016
2019-07-03	EARLY	11825	4574	1888	2094

Kafka

TOC_STATS

Aggregate
Aggregate

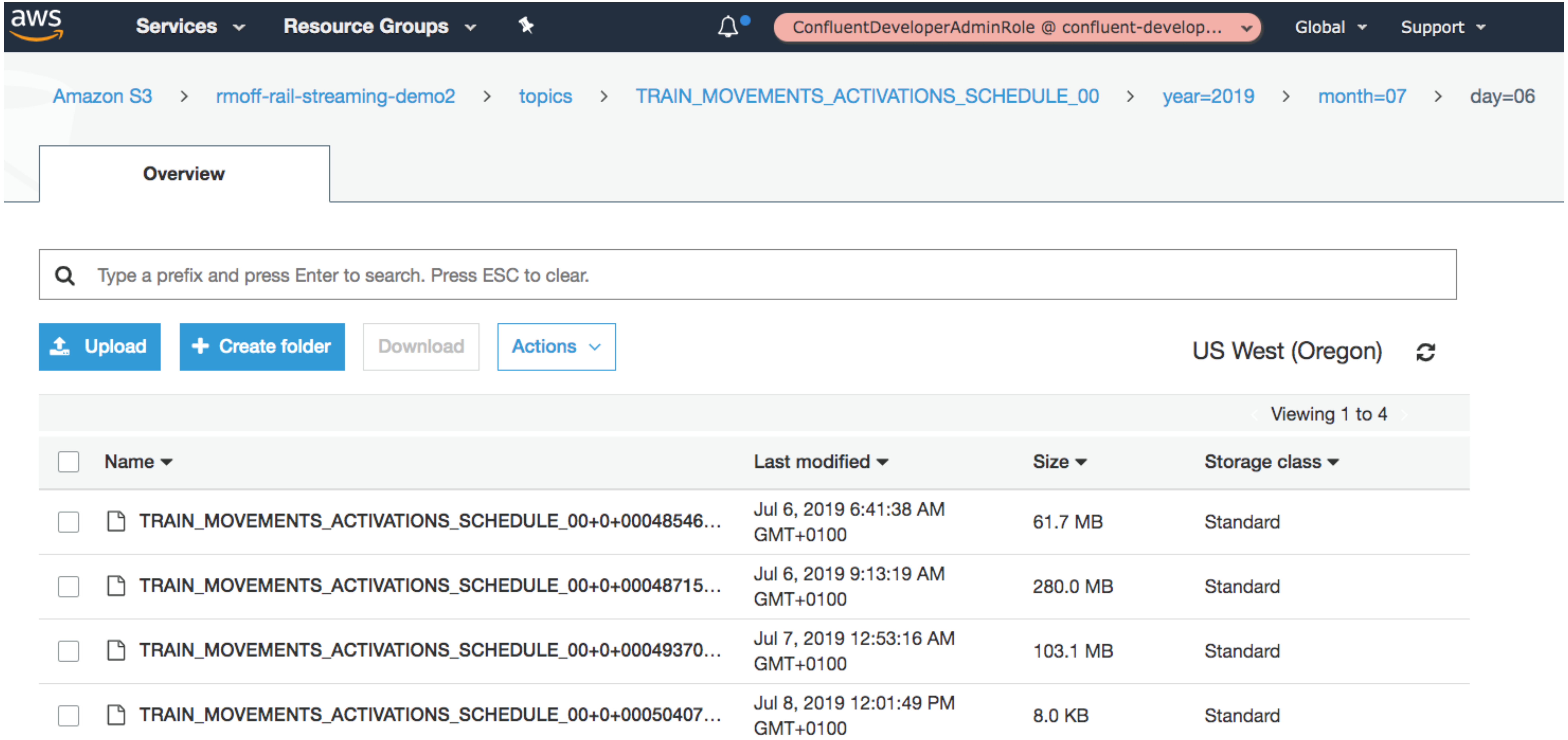
Time

Kafka -> S3

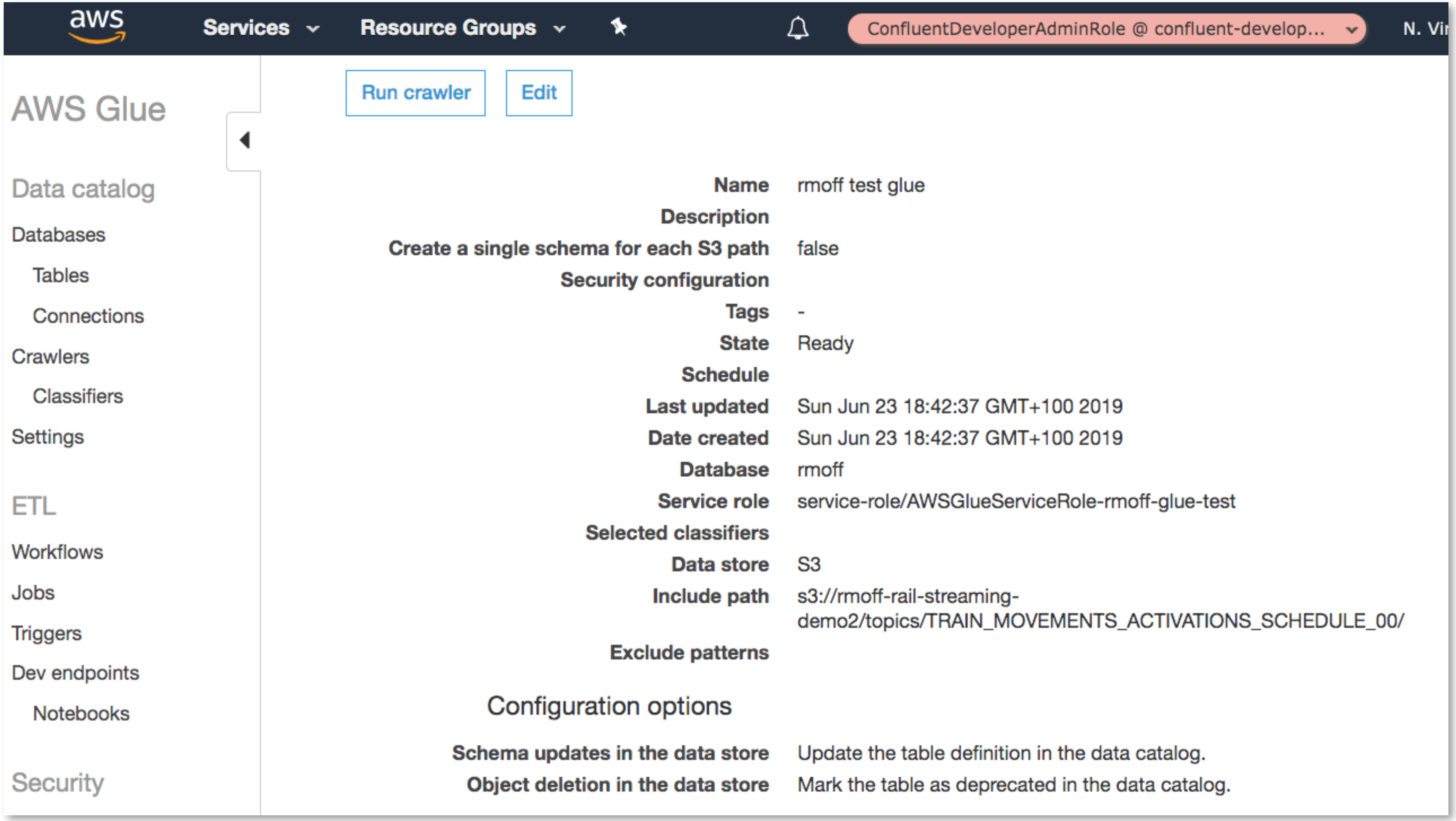
```
{
   "connector.class": "io.confluent.connect.s3.S3SinkConnector",
   "key.converter":"org.apache.kafka.connect.storage.StringConverter",
   "tasks.max": "1",
   "topics": "TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00",
   "s3.region": "us-west-2",
   "s3.bucket.name": "rmoff-rail-streaming-demo",
   "flush.size": "65536",
   "storage.class": "io.confluent.connect.s3.storage.S3Storage",
   "format.class": "io.confluent.connect.s3.format.avro.AvroFormat",
   "schema.generator.class": "io.confluent.connect.storage.hive.schema.DefaultSchemaGenerator",
   "schema.compatibility": "NONE",
   "partitioner.class": "io.confluent.connect.storage.partitioner.TimeBasedPartitioner",
   "path.format":"\'year\'=YYYY/\'month\'=MM/\'day\'=dd",
   "timestamp.extractor":"Record",
   "partition.duration.ms": 1800000,
   "locale":"en",
   "timezone": "UTC"
 }
```

Kafka -> S3

```
$ aws s3 ls s3://rmoff-rail-streaming-demo/topics/TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00/year=2019/month=07/day=07/
2019-07-07 10:05:41  200995548 TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00+0+0004963416.avro
2019-07-07 19:47:11   87631494 TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00+0+0005007151.avro
```



Infering table config from S3 files with AWS Glue



Infering table config from S3 files with AWS Glue

Schema

Showing: 1 - 66 of 66

	Column name	Data type	Partition key	Comment
1	train_id	string		
2	header	struct		
3	event_type	string		
4	gbtt_timestamp	string		
61	schedule_segment_zero…	string		
62	schedule_segment_locat…	array		
63	msg_key	string		
64	year	string	Partition (0)	
65	month	string	Partition (1)	
66	day	string	Partition (2)	

Analysis with AWS Athena

explode schedule d... | New query 2 | +

```sql
SELECT train_id,
       DATE_FORMAT(FROM_UNIXTIME(actual_timestamp/1000),
           '%Y-%m-%dT%H:%i:%sZ') AS actual_ts,
       loc_nlcdesc,
       json_extract(element_at(SCHEDULE_SEGMENT_LOCATION,-1),
                  '$.tiploc_code') AS last_station
FROM TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00
WHERE year = '2019'
        AND month='06'
        AND day = '23'
        AND originating_location='ILKLEY'
LIMIT 50;
```

Run query | Save as | Create (Run time: 6.01 seconds, Data scanned: 82.85 MB)

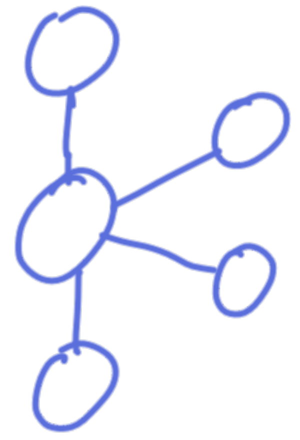
Format query | Clear

Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

Results

	train_id	actual_ts	loc_nlcdesc	last_station
1	172V31MD23	2019-06-23T08:44:00Z	GUISELEY	"LEEDS"
2	172V33ME23	2019-06-23T09:39:00Z	APPERLEY JN	"LEEDS"
3	172V33ME23	2019-06-23T09:35:00Z	GUISELEY	"LEEDS"

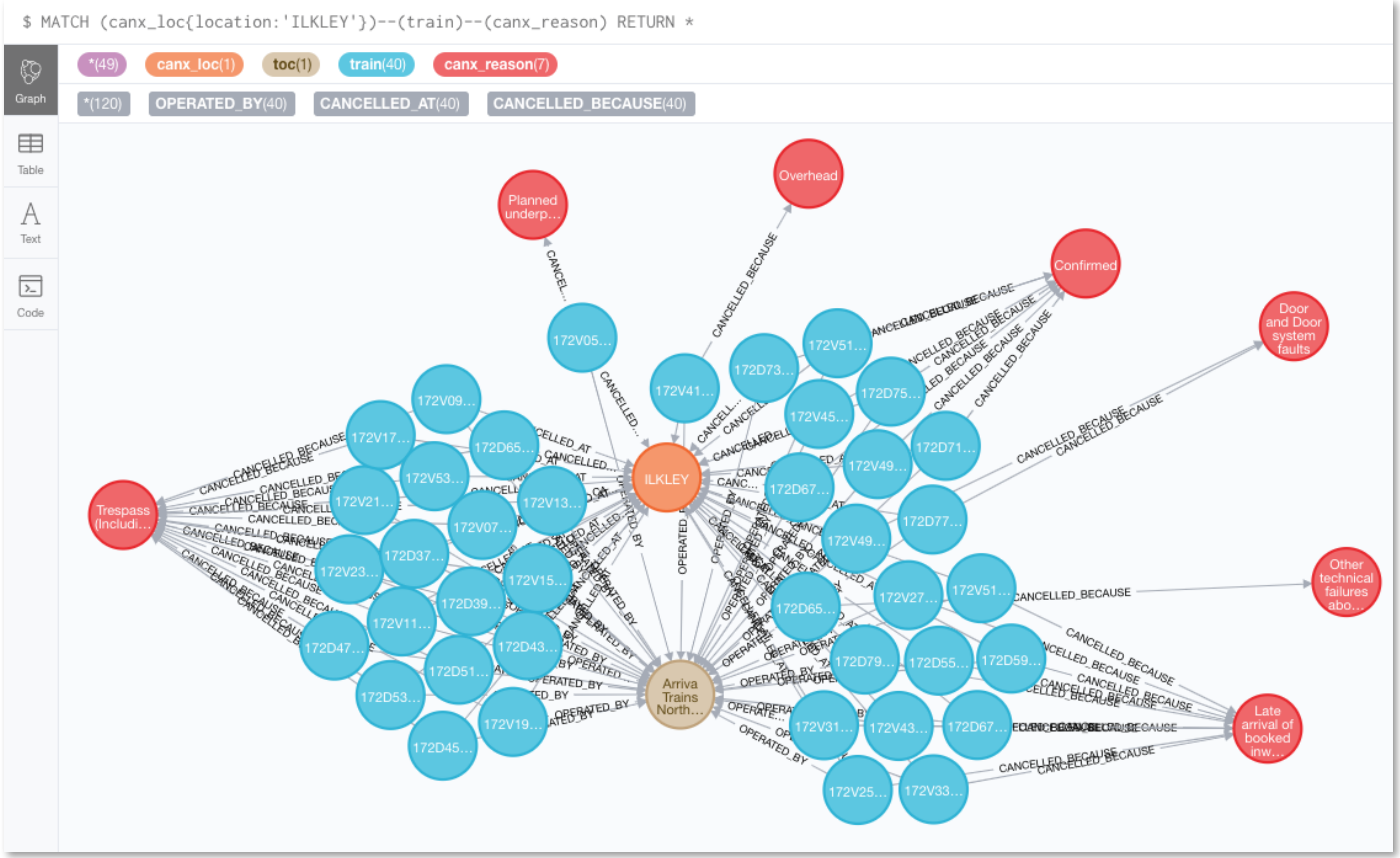
Graph analysis



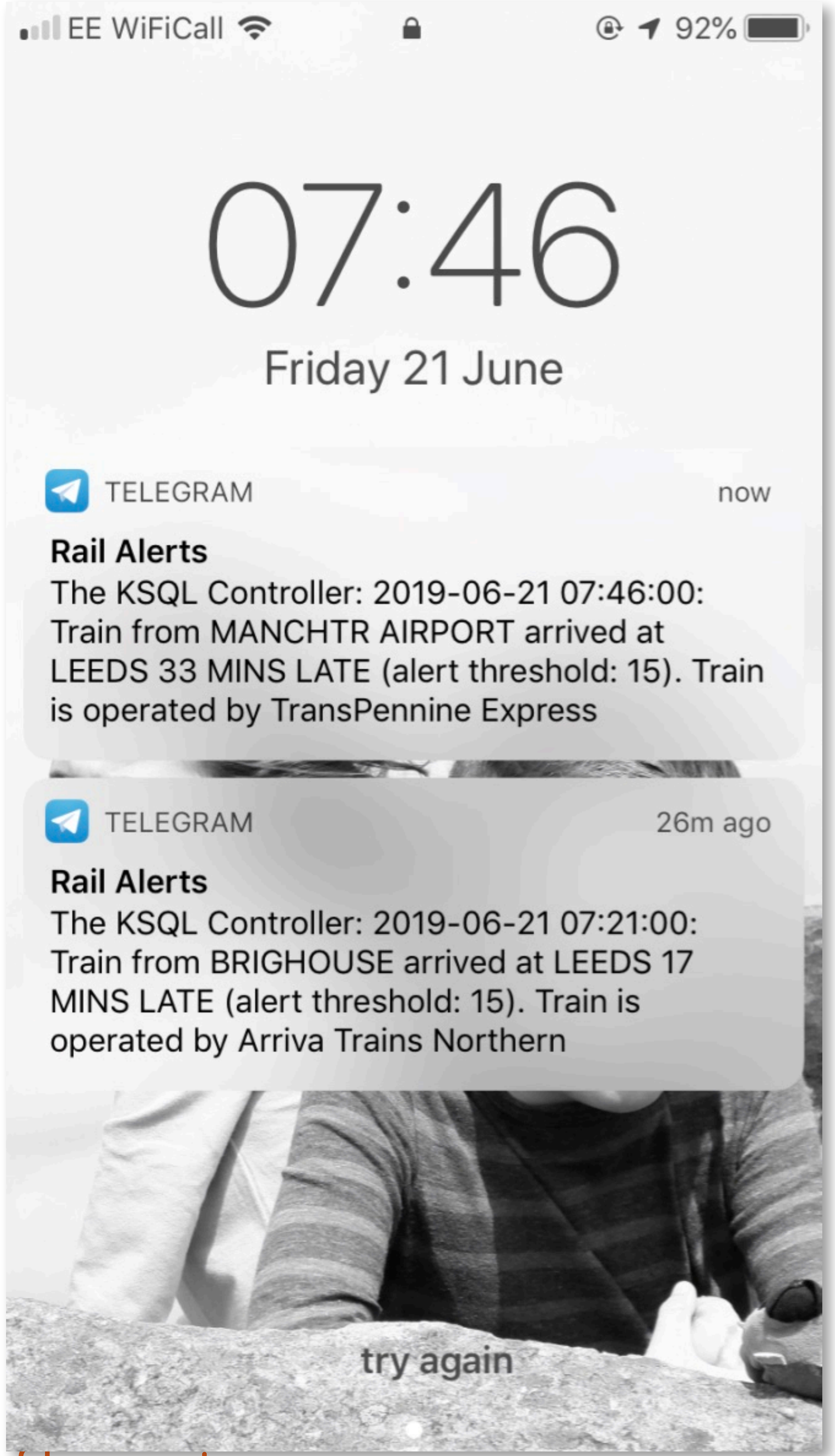
neo4j

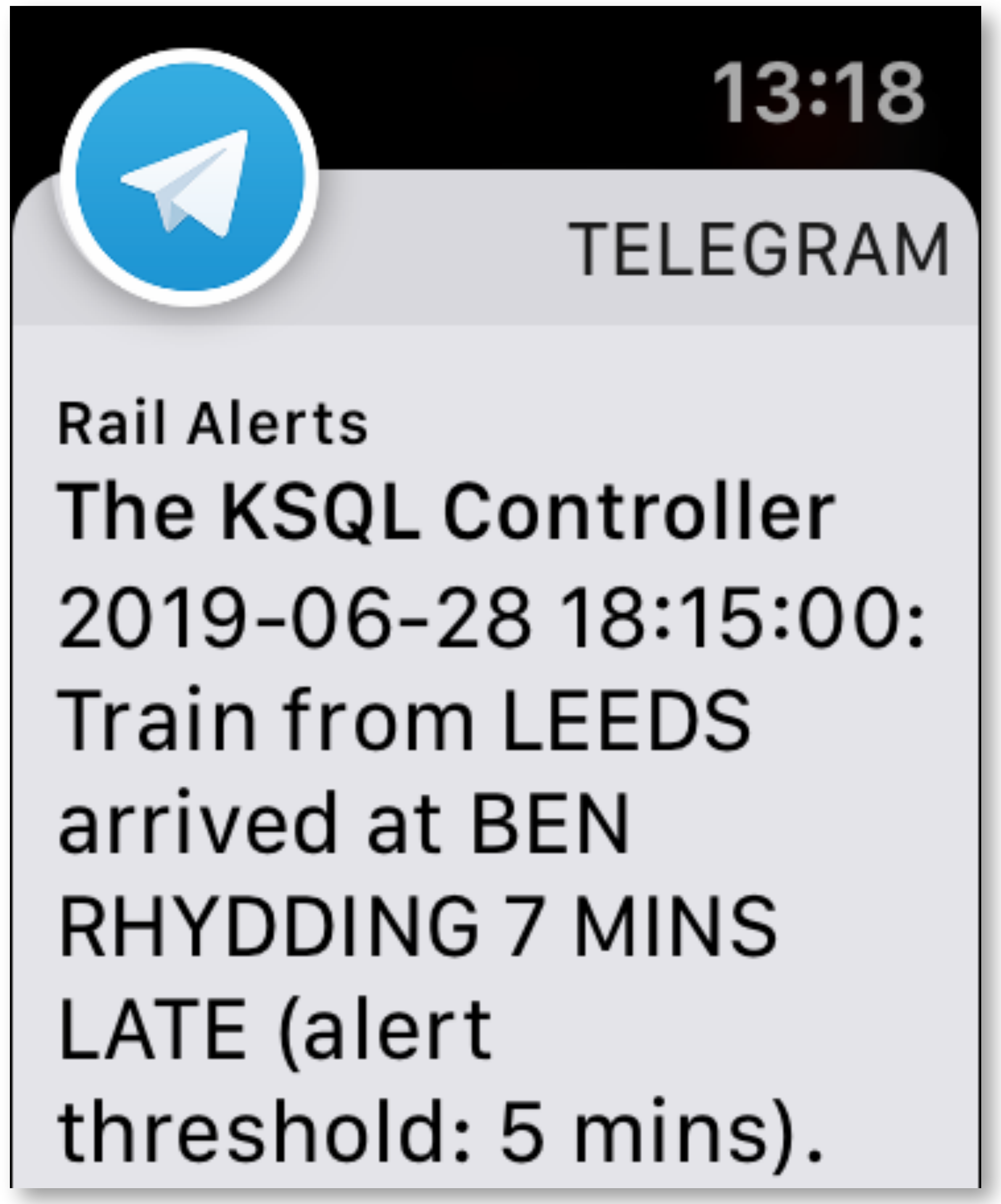
```
{
    "connector.class": "streams.kafka.connect.sink.Neo4jSinkConnector",
    "topics": "TRAIN_CANCELLATIONS_02",
    "key.converter":"org.apache.kafka.connect.storage.StringConverter",
    "errors.tolerance": "all",
    "errors.deadletterqueue.topic.name": "sink-neo4j-train-00_dlq",
    "errors.deadletterqueue.topic.replication.factor": 1,
    "errors.deadletterqueue.context.headers.enable":true,
    "neo4j.server.uri": "bolt://neo4j:7687",
    "neo4j.authentication.basic.username": "neo4j",
    "neo4j.authentication.basic.password": "connect",
    "neo4j.topic.cypher.TRAIN_CANCELLATIONS_02": "MERGE (train:train{id: event.TRAIN_ID}) MERGE
(toc:toc{toc: event.TOC}) MERGE (canx_reason:canx_reason{reason: event.CANX_REASON}) MERGE
(canx_loc:canx_loc{location: coalesce(event.CANCELLATION_LOCATION,\'<unknown>\')}) MERGE (train)-
[:OPERATED_BY]->(toc) MERGE (canx_loc)<-[:CANCELLED_AT{reason:event.CANX_REASON,
time:event.CANX_TIMESTAMP}]-(train)-[:CANCELLED_BECAUSE]->(canx_reason)"
      }
```

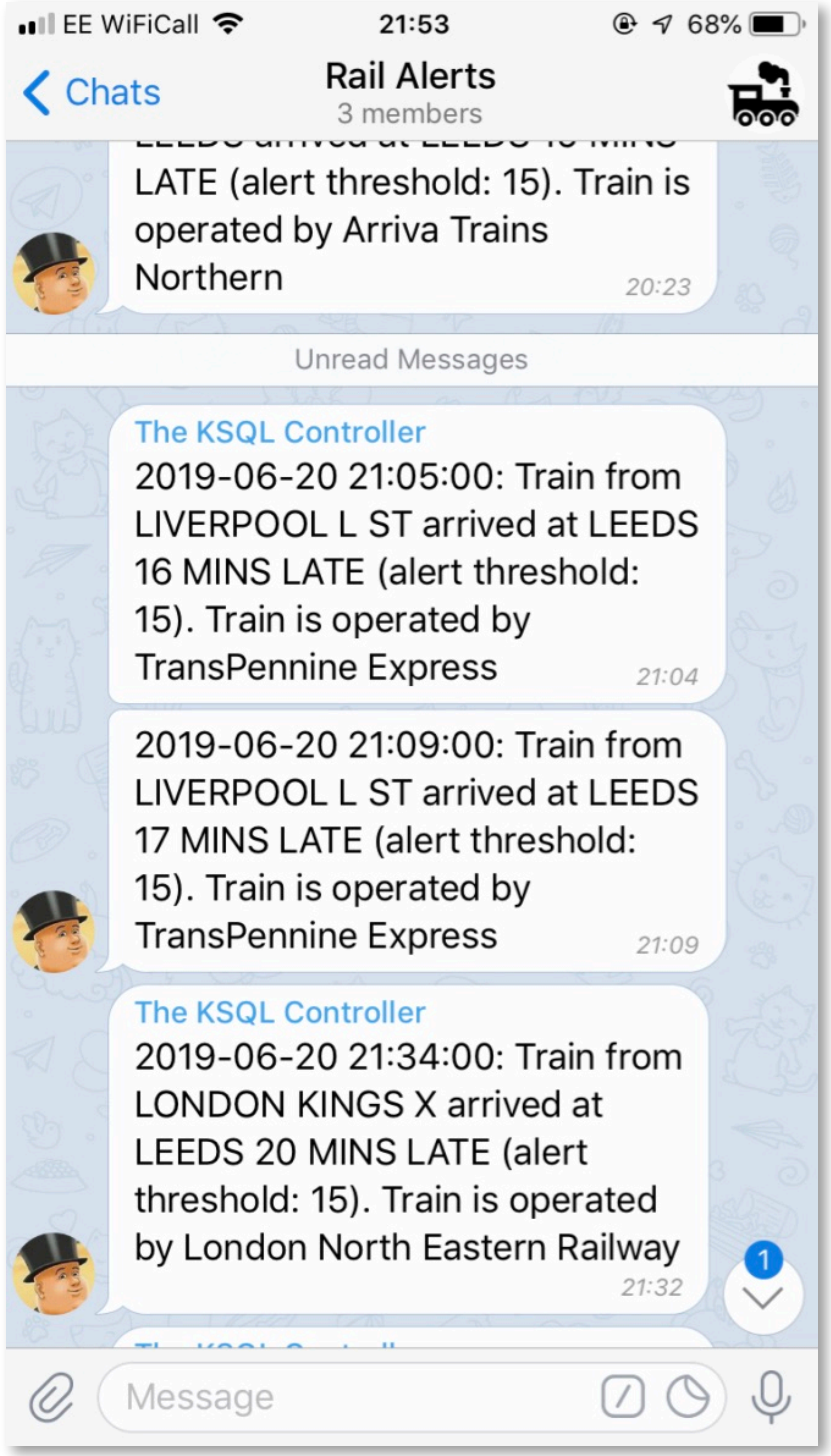
Graph relationships



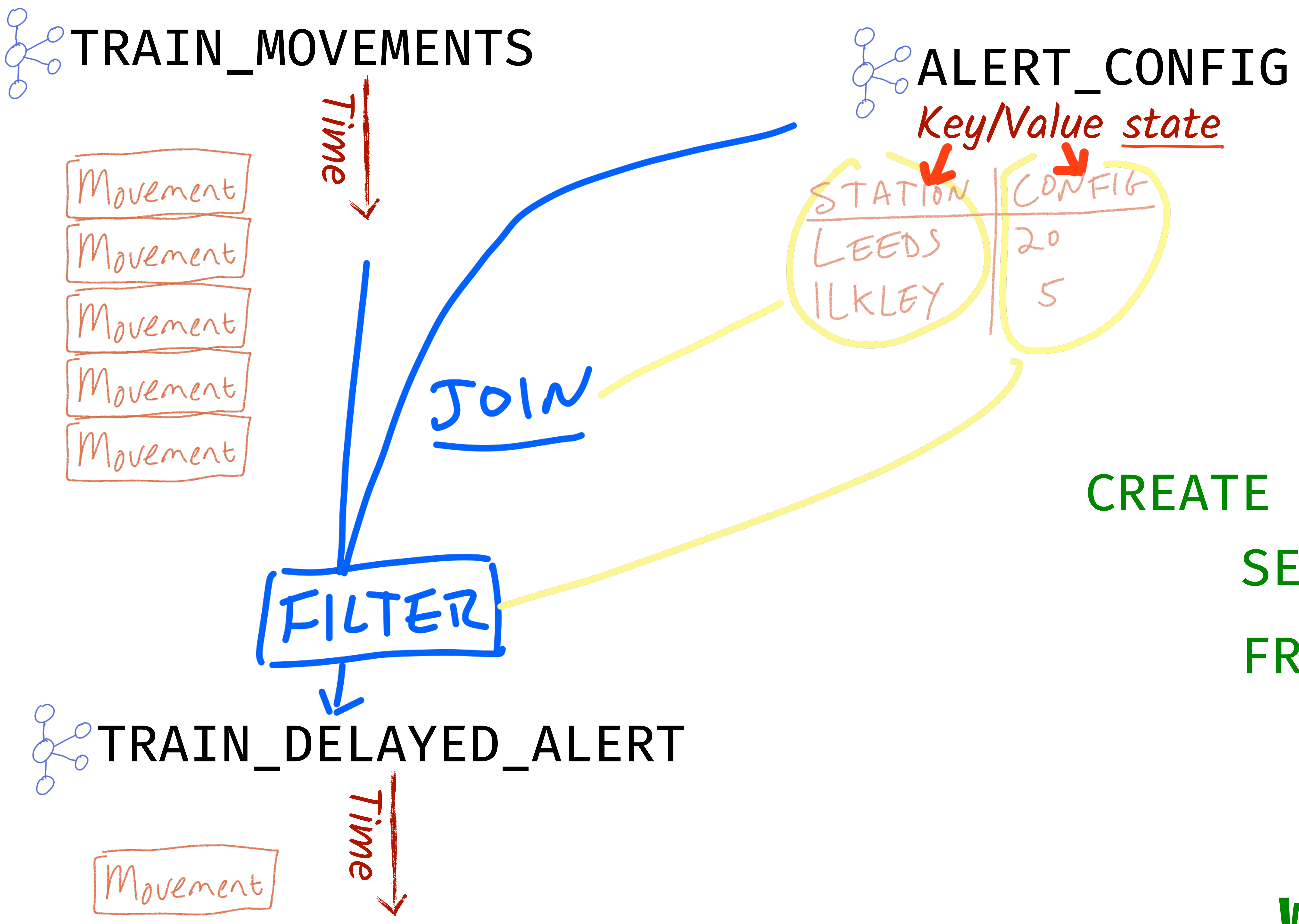
Tell me when trains are delayed at my station







Event-driven Alerting - logic

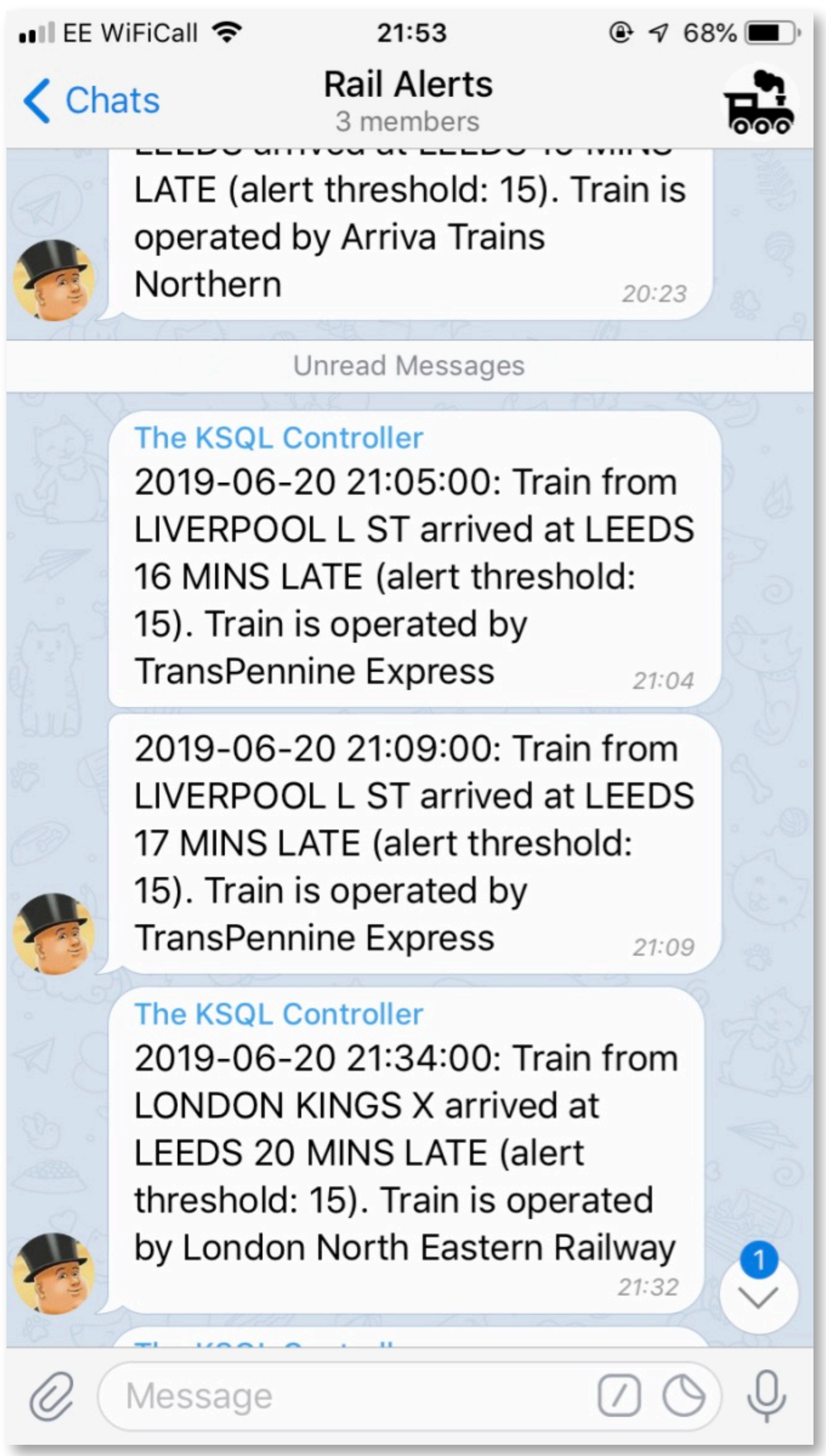



```sql
CREATE STREAM TRAINS_DELAYED_ALERT AS
     SELECT […]
     FROM TRAIN_MOVEMENTS T
          INNER JOIN ALERT_CONFIG A
          ON T.LOC_NLCDESC = A.STATION
      WHERE TIMETABLE_VARIATION
              > A.ALERT_OVER_MINS;
```

Sending messages to Telegram

```
curl -X POST -H 'Content-Type: application/json'
     -d '{"chat_id": "-364377679", "text": "This is a test from curl"}'
     https://api.telegram.org/botxxxxx/sendMessage
```

```
{
  "connector.class": "io.confluent.connect.http.HttpSinkConnector",
  "request.method": "post",
  "http.api.url": "https://api.telegram.org/[…]/sendMessage",
  "headers": "Content-Type: application/json",
  "topics": "TRAINS_DELAYED_ALERT_TG",
  "tasks.max": "1",
  "batch.prefix": "{\"chat_id\":\"-364377679\",\"parse_mode\": \"markdown\",",
  "batch.suffix": "}",
  "batch.max.size": "1",
  "regex.patterns":".*\\\{MESSAGE=(.*)\\\}.*",
  "regex.replacements": "\"text\":\"$1\"",
  "regex.separator": "~",
  "key.converter": "org.apache.kafka.connect.storage.StringConverter"
}
```



Event-driven Alerting - config

Compacted topic

```
$ kafkacat -b localhost:9092 -t alert_config -P -K: <<EOF
LEEDS:{"STATION":"LEEDS","ALERT_OVER_MINS":"45"}
EOF
```

```
CREATE TABLE ALERT_CONFIG (STATION VARCHAR,
                           ALERT_OVER_MINS INT)
        WITH (KAFKA_TOPIC='alert_config',
              VALUE_FORMAT='JSON',
              KEY='STATION');
```

```
ksql> SELECT STATION, ALERT_OVER_MINS
FROM ALERT_CONFIG WHERE STATION='LEEDS';

LEEDS | 45
```

```
CREATE STREAM ALERT_CONFIG_STREAM (STATION VARCHAR,
                                   ALERT_OVER_MINS INT)
        WITH (KAFKA_TOPIC='alert_config',
              VALUE_FORMAT='JSON');
```

```
ksql> SELECT STATION, ALERT_OVER_MINS
FROM ALERT_CONFIG_STREAM WHERE STATION='LEEDS';

LEEDS | 45
```

```
$ kafkacat -b localhost:9092 -t alert_config -P -K: <<EOF
LEEDS:{"STATION":"LEEDS","ALERT_OVER_MINS":"20"}
EOF
```

```
ksql> SELECT STATION, ALERT_OVER_MINS
FROM ALERT_CONFIG WHERE STATION='LEEDS';

LEEDS | 20
```

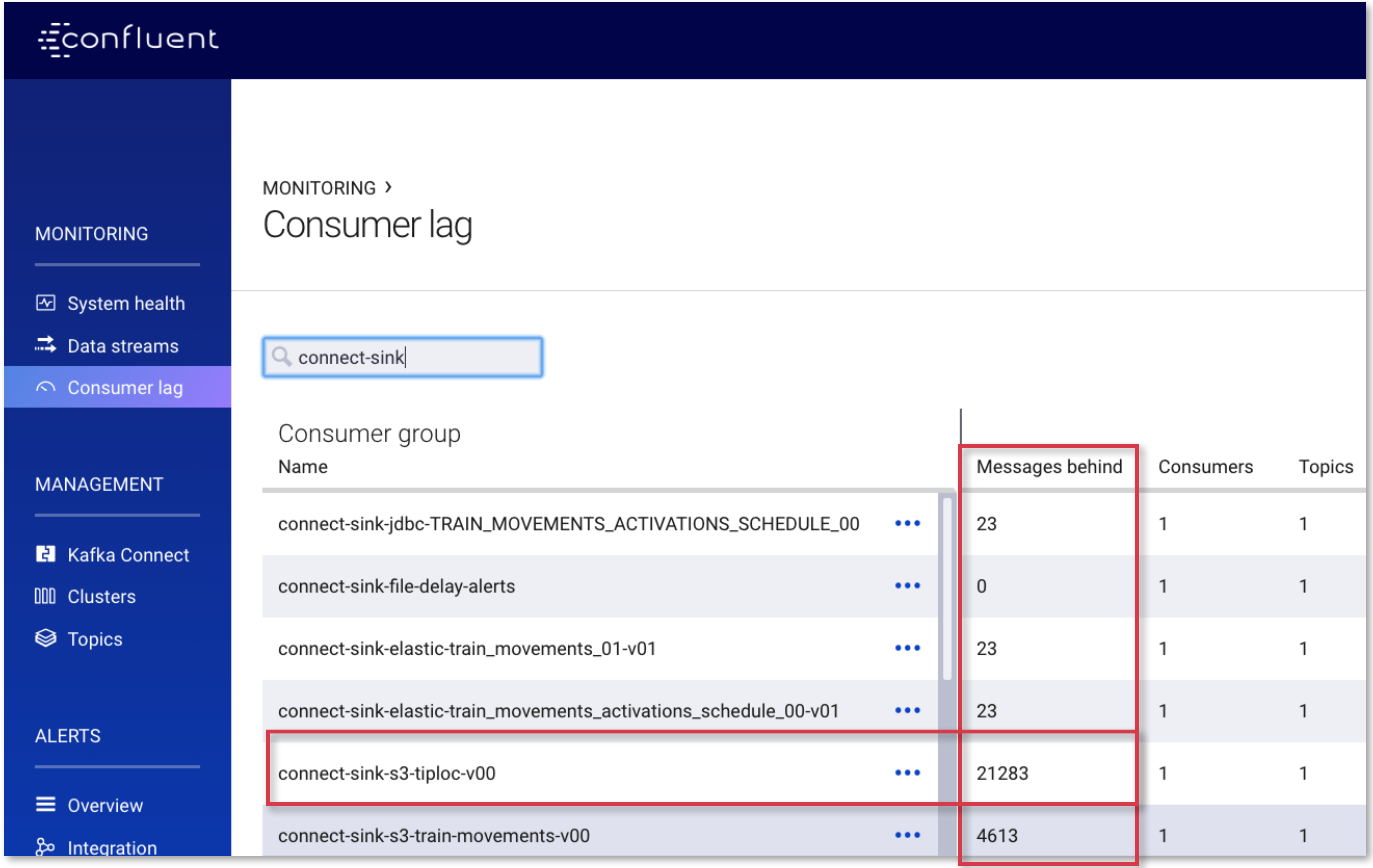
```
ksql> SELECT STATION, ALERT_OVER_MINS
FROM ALERT_CONFIG_STREAM WHERE STATION='LEEDS';

LEEDS | 45 *
LEEDS | 20
```

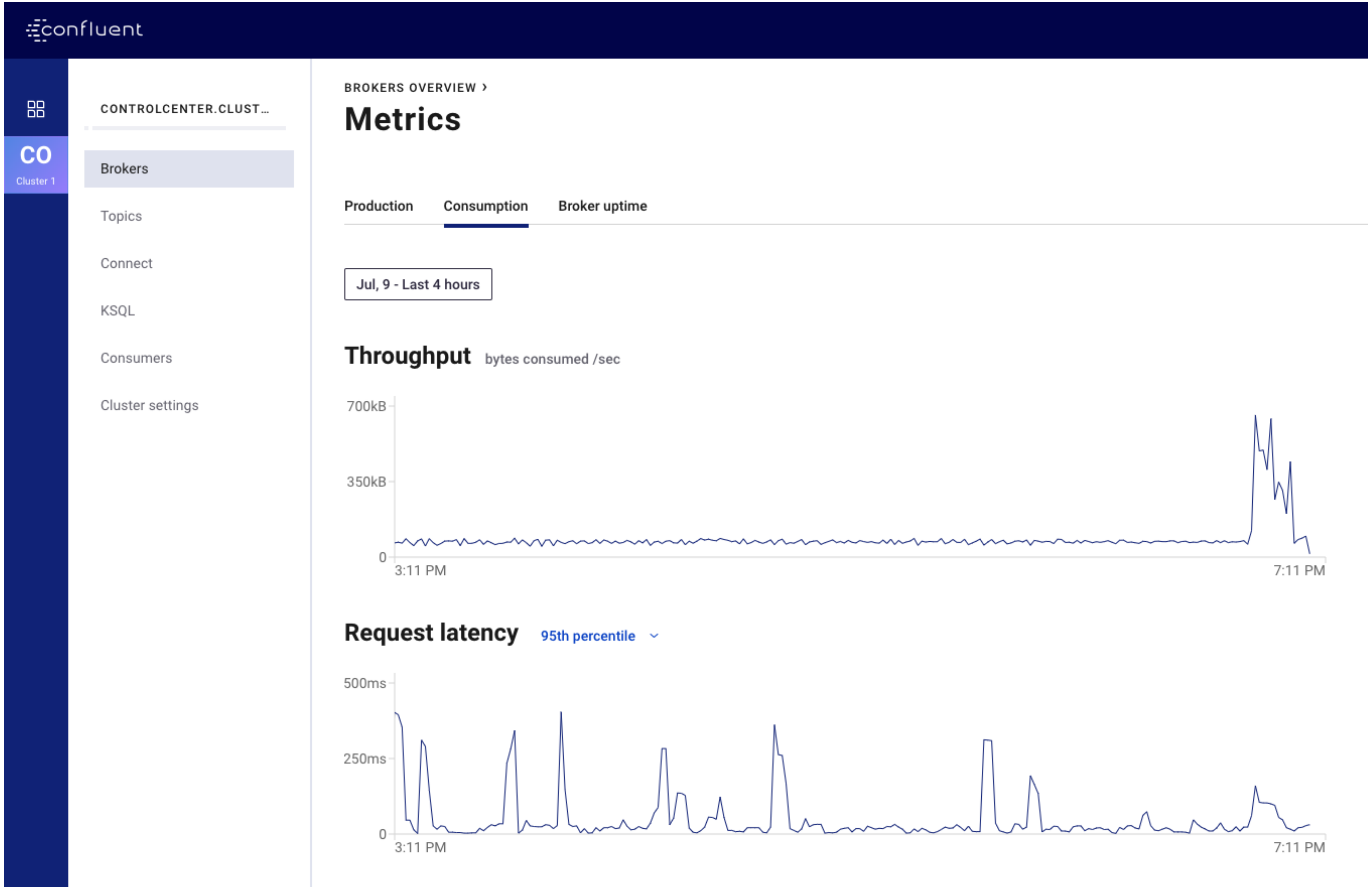
* until topic compaction runs

Running it & monitoring & maintenance

Consumer lag



System health & Throughput



Restart failing connectors

```
rmoff@proxmox01 > crontab -l
*/5 * * * * /home/rmoff/restart_failed_connector_tasks.sh
```

```
rmoff@proxmox01 > cat /home/rmoff/restart_failed_connector_tasks.sh
#!/usr/bin/env bash
# @rmoff / June 6, 2019

# Restart any connector tasks that are FAILED
curl -s "http://localhost:8083/connectors" | \
  jq '.[]' | \
  xargs -I{connector_name} curl -s "http://localhost:8083/connectors/"{connector_name}"/status" | \
  jq -c -M '[select(.tasks[].state=="FAILED") | .name,"§±§",.tasks[].id]' | \
  grep -v "\[\]"| \
  sed -e 's/^\[\"//g'| sed -e 's/\",\"§±§\",/\/tasks\//g'|sed -e 's/\]$//g'| \
  xargs -I{connector_and_task} curl -X POST "http://localhost:8083/connectors/"{connector_and_task}"/
restart"
```

Is everything running?

http://rmoff.dev/kafka-trains-code-01

```
Every 2.0s: ./list_indices_stats.sh

Connectors
----------
sink-elastic-train_cancellations_02-v01                   |  RUNNING  |  RUNNING
sink-elastic-train_movements_01-v01                       |  RUNNING  |  RUNNING
sink-elastic-train_movements_activations_schedule_00-v01  |  RUNNING  |  RUNNING
sink-jdbc-TRAIN_MOVEMENTS_ACTIVATIONS_SCHEDULE_00         |  RUNNING  |  RUNNING
sink-s3-tiploc-v00                                        |  RUNNING  |  RUNNING
sink-s3-train-movements-v00                               |  RUNNING  |  RUNNING

Indices and doc count
---------------------
train_movements_activations_schedule_00 2710181
train_movements_01                      2712204
train_cancellations_02                     9383

Indices and timestamp of latest doc
-----------------------------------
2019-06-27 09:00:25          train_movements_activations_schedule_00
2019-06-27 09:00:25          train_movements_01
2019-06-27 08:53:55          train_cancellations_02
```

check_latest_timestamp.sh

Just read one message

Start at latest message

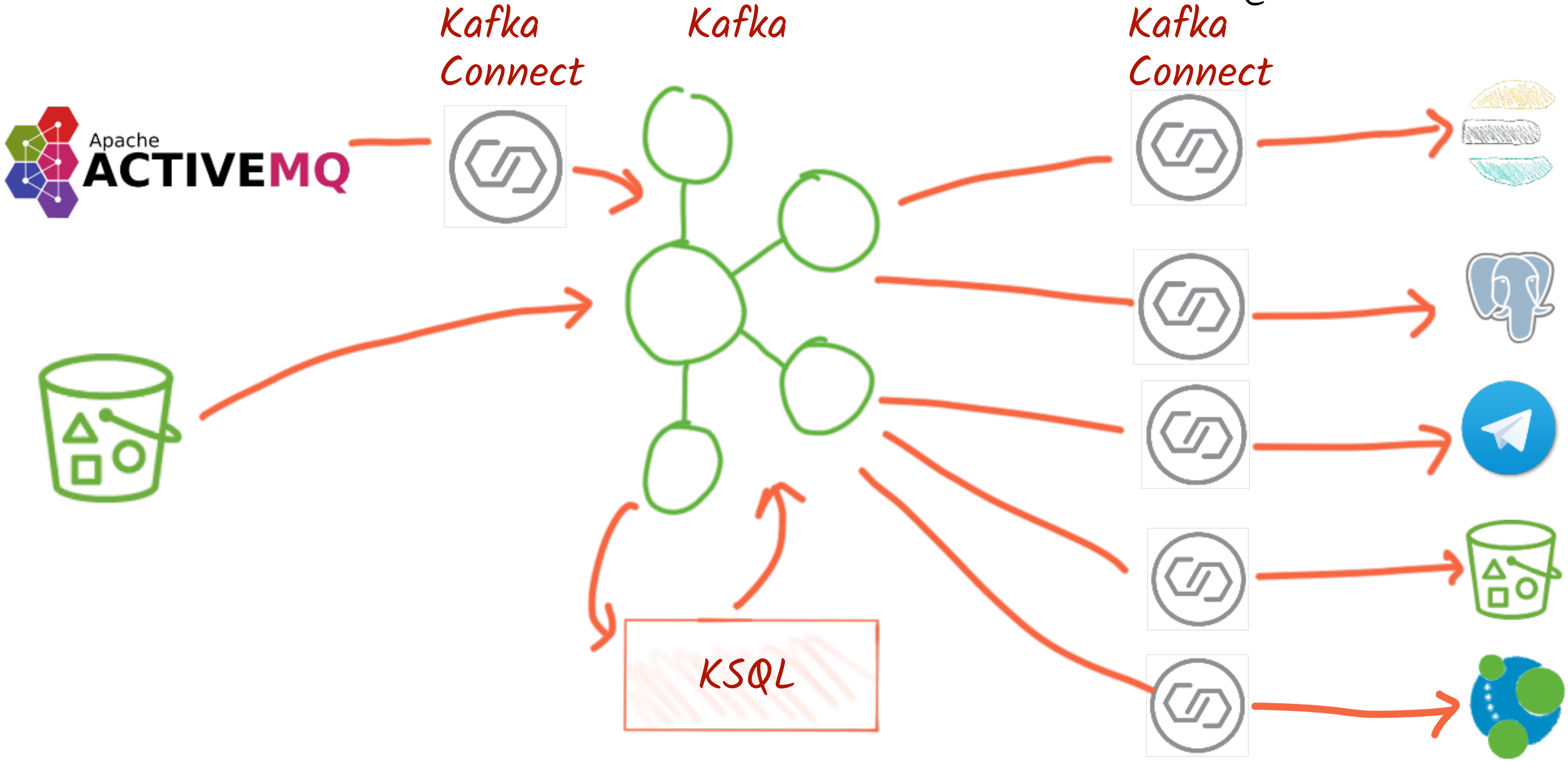
```
kafkacat -b localhost:9092 -t networkrail_TRAIN_MVT_X -o-1 -c1 -C | \
jq '.header.msg_queue_timestamp' | \
sed -e 's/"//g' | \
sed -e 's/000$//g' | \
xargs -Ifoo date --date=@foo
```

Convert from epoch to human-readable timestamp format

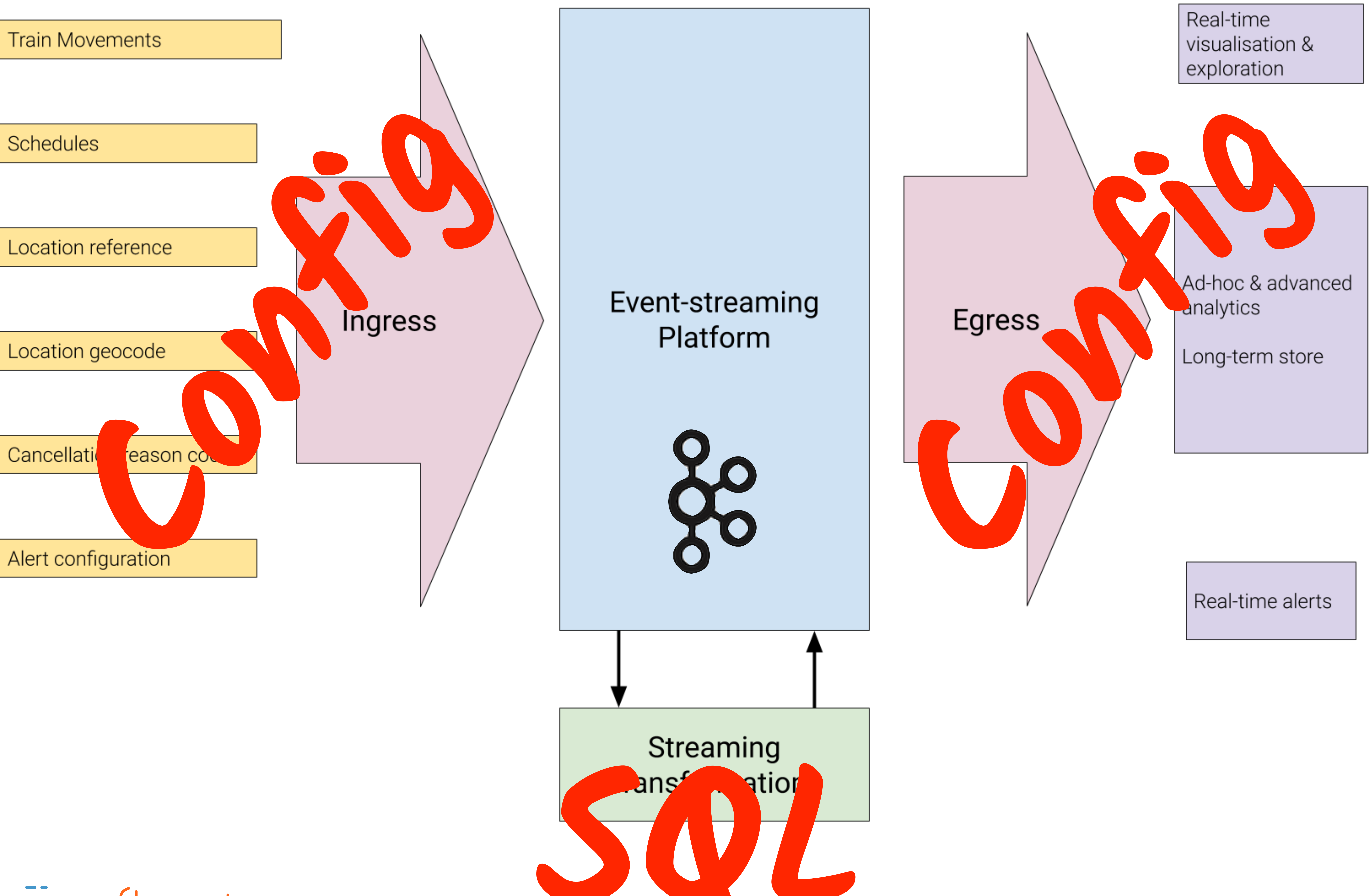
```
$ ./check_latest_timestamp.sh
Fri Jul  5 16:28:09 BST 2019
```

$ cat TODO.txt

- Automate ingest & monitoring
  - currently cron, replace with Apache Airflow?
- Ad-hoc visual analysis
  - Superset? Google Data Studio? AWS Quicksight?
- Handle schedule nested array to be able to show for each train movement its planned route
- Proper partitioning of topics (current partitions=1 🙄)



✅ Stream raw data ✅ Stream enriched data ✅ Historic data & data replay



Train Movements
Schedules
Location reference
Location geocode
Alert configuration
Ingress
Config
Event-streaming Platform
Egress
Config
Real-time visualisation & exploration
Ad-hoc & advanced analytics
Long-term store
Real-time alerts
Streaming
SQL

NEXT STOP

SAN FRANCISCO

SEPT 30-OCT 1

CONFLUENT COMMUNITY DISCOUNT CODE

KS19Meetup

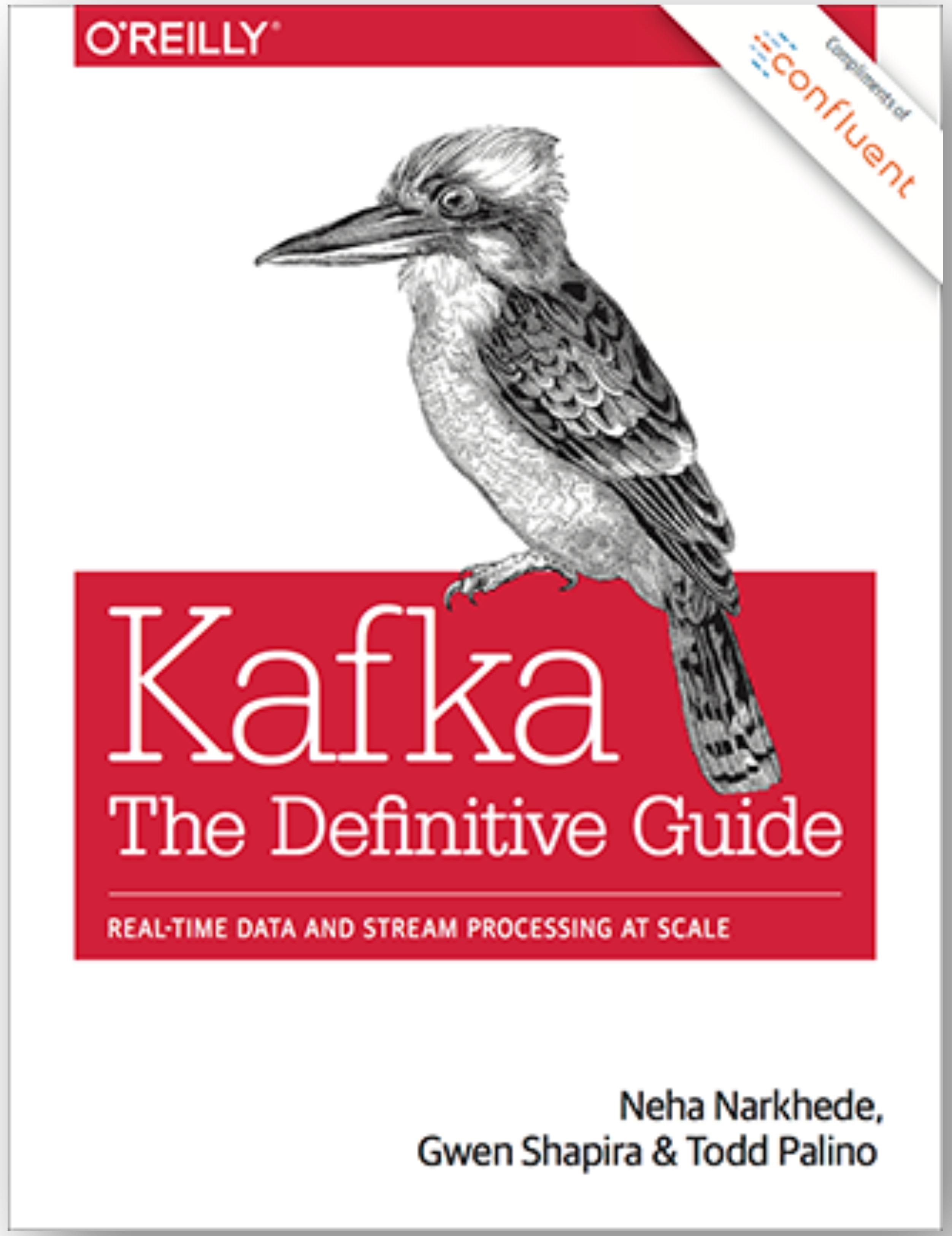
25% OFF*

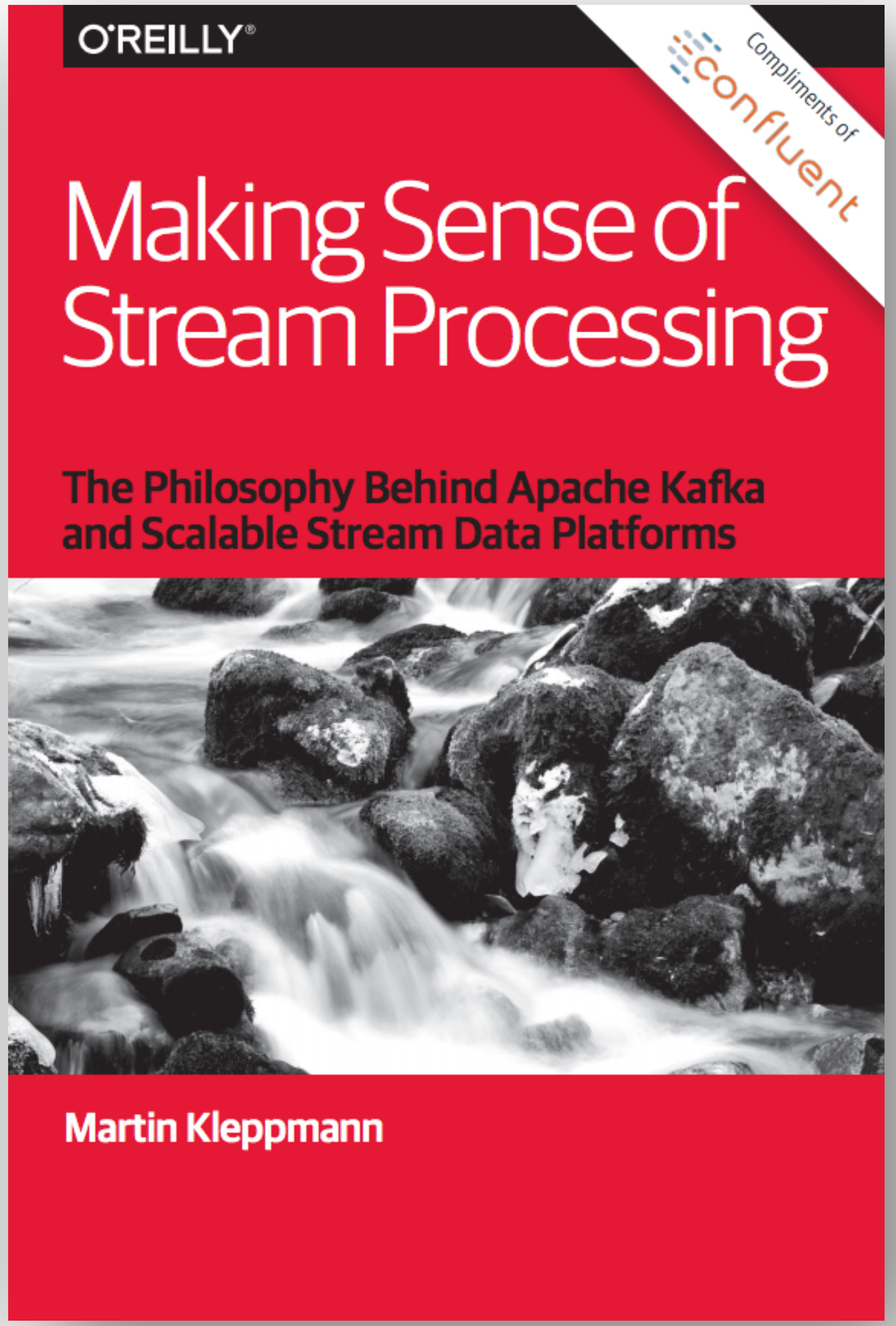
kafka summit

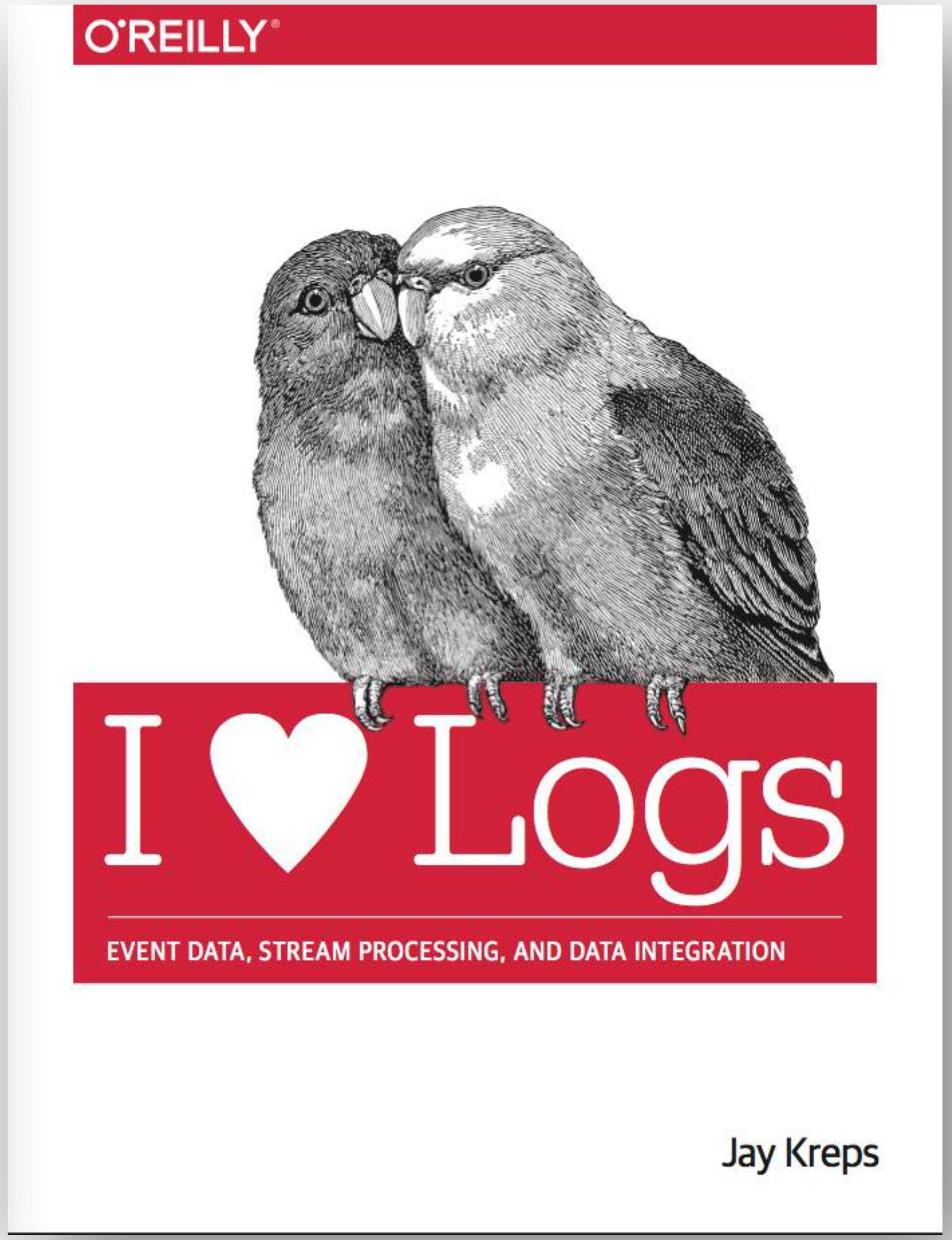
*Standard Priced Conference pass

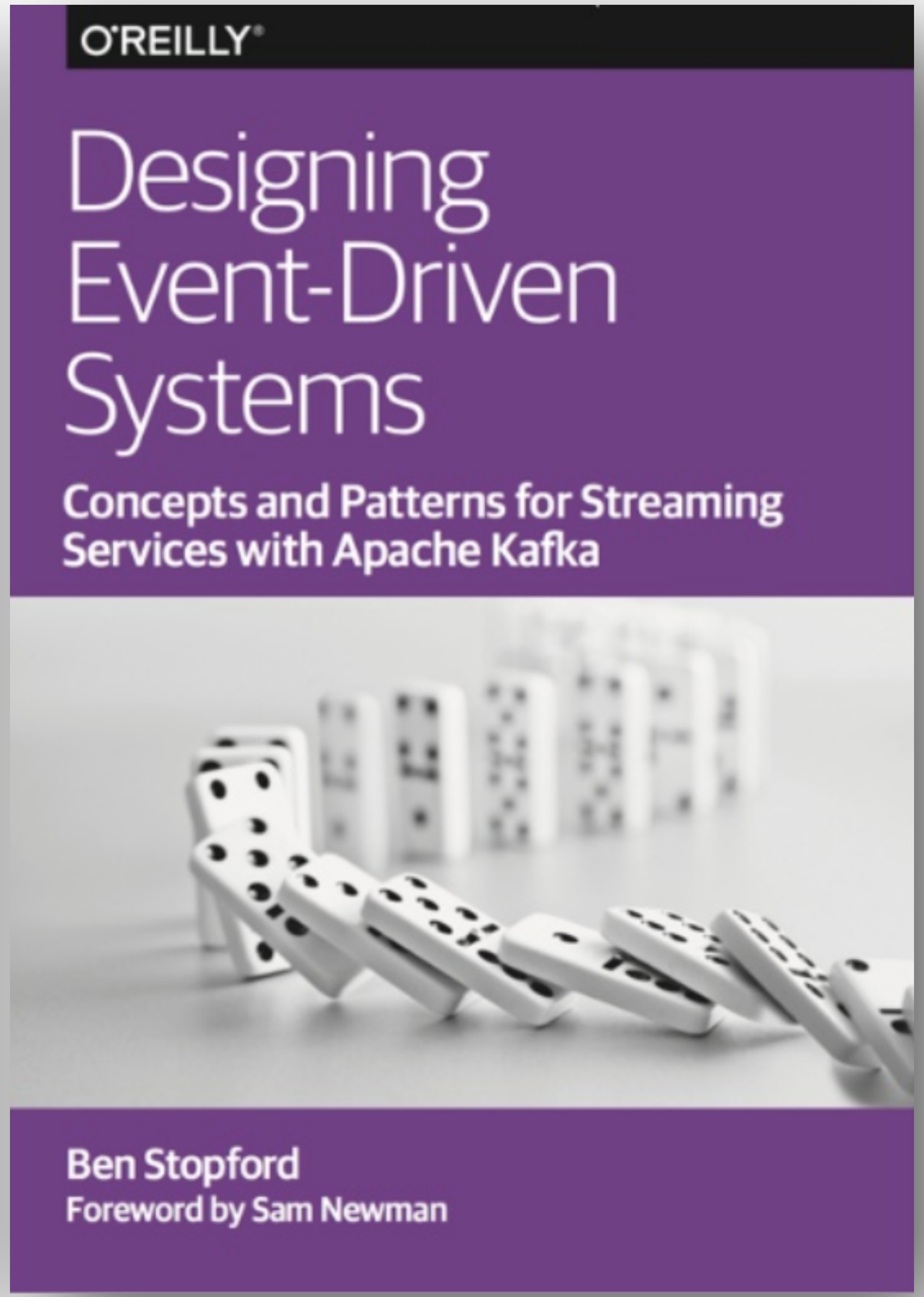
Free Books!

http://cnfl.io/book-bundle

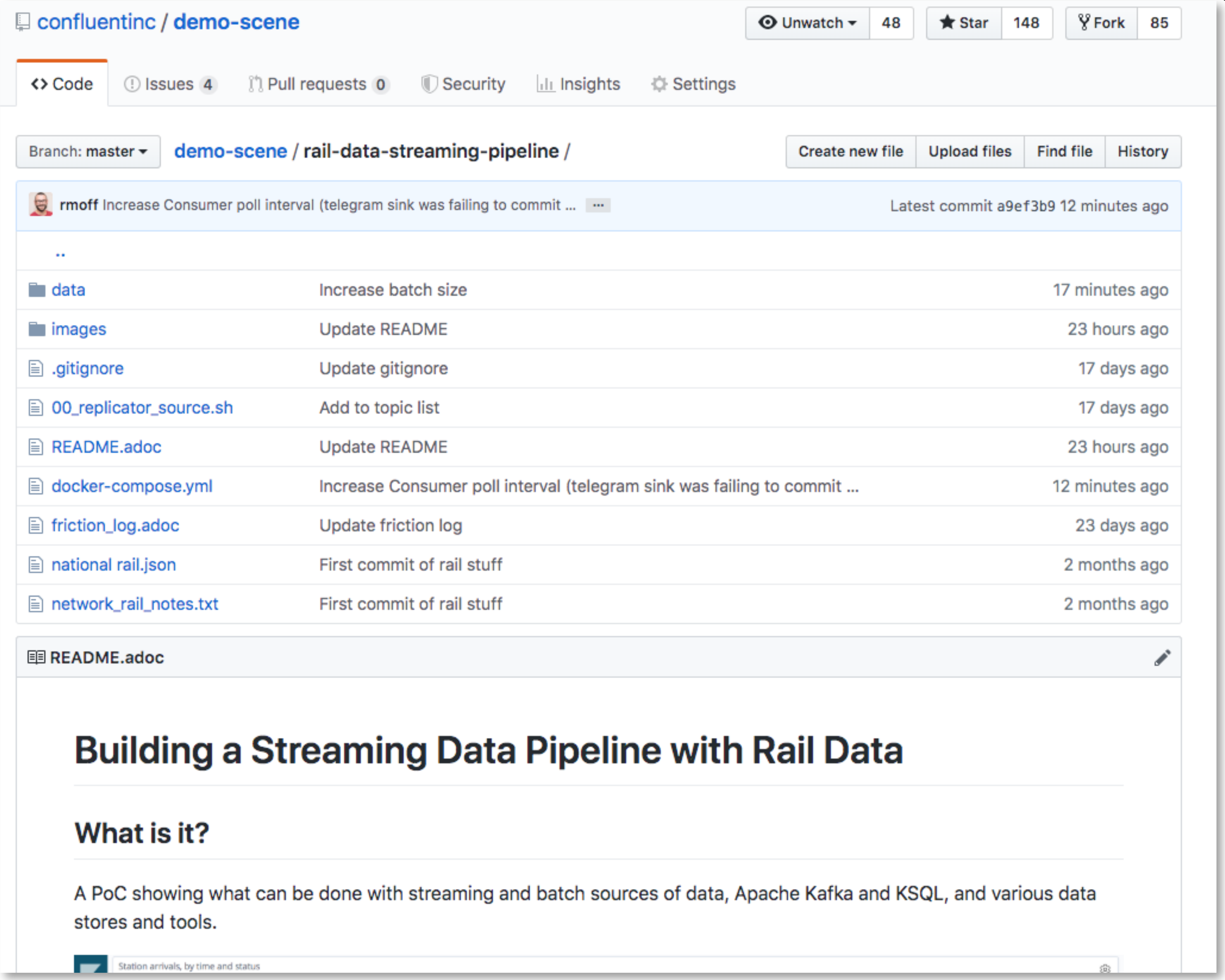








Code!



http://rmoff.dev/kafka-trains-code-01

@rmoff #KafkaMeetup

#EOF

💬 Join the Confluent Community Slack group at http://cnfl.io/slack

https://talks.rmoff.net