Big Data and Machine Learning
(London, Cambridge and Munich)

The Lab Series

Tracking commercial aircraft in near real-time using a Raspberry Pi, Kafka and Vertica

Mark Whalley
Vertica Systems Engineer
12th October 2017

www.vertica.com
www.myvertica.com

MICRO FOCUS

Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

Agenda

- **Background to "The Lab Series" and the Big Data & Machine Learning Meetups**
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What's next?

Big Data & Machine Learning (London) Meetup

London, United Kingdom
Founded Dec 1, 2016

About us...

Invite friends

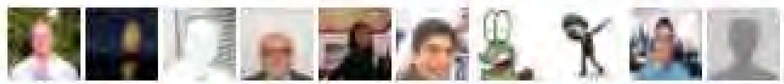
Members	1,487
Group reviews	8
Past Meetups	11
Our calendar	

Recent Meetups

3 days ago · 6:30 PM

Rate this Meetup

Big Data and Machine Learning - London - Meetup #7

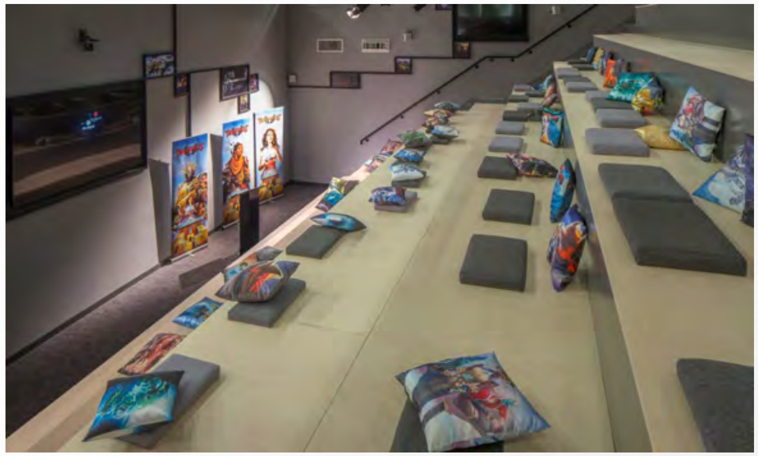


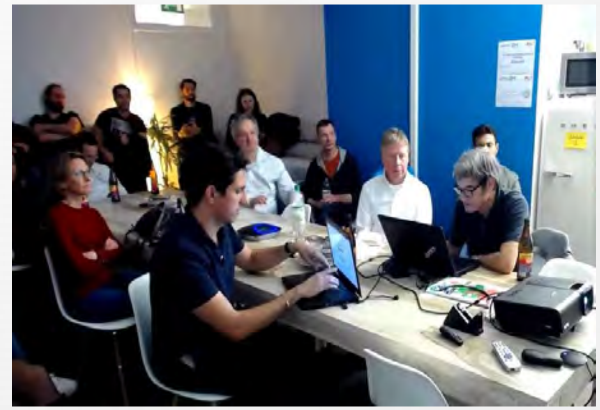
73 Members | | 3 Photos

Copy this Meetup

Meetup #7 PLEASE NOTE: Limit of 140 attendees (see below) Welcome to Meetup #7, and what we hope will be another interesting evening of presentations and lightning talks... LEARN MORE

And then there were three: London, Munich & Cambridge





CAMPUS LONDON

gini

The Lab Series - Background

- Mini projects
  - Incubate
  - Subject
  - Technology
  - Direction
  - Presentations
- Inclusive to all

Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - **Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)**
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

The Lab Series – Project #1

Automatic Dependent Surveillance - Broadcast (ADS-B)

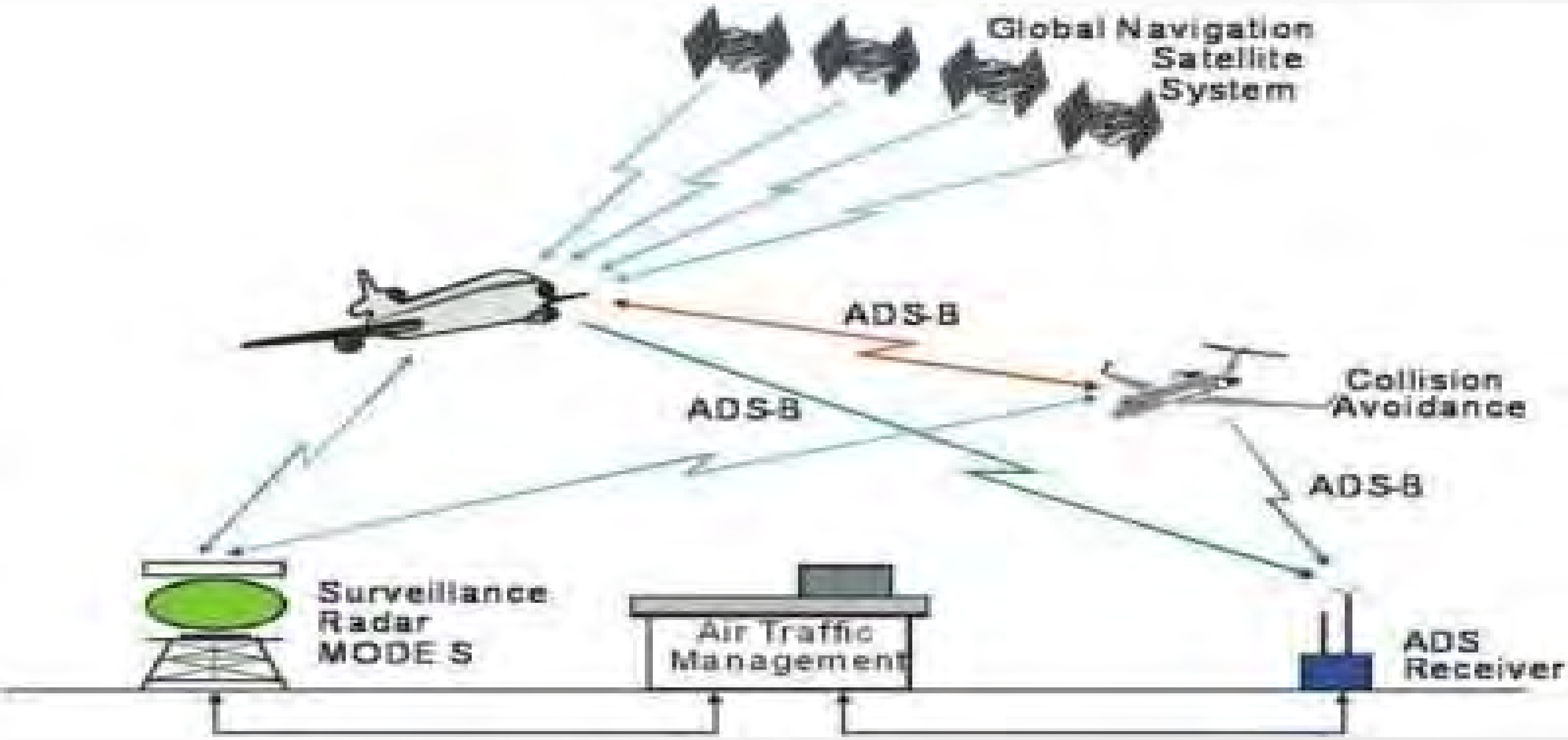
Contrary to popular belief…

This got me thinking…

https://vimeo.com/110348926

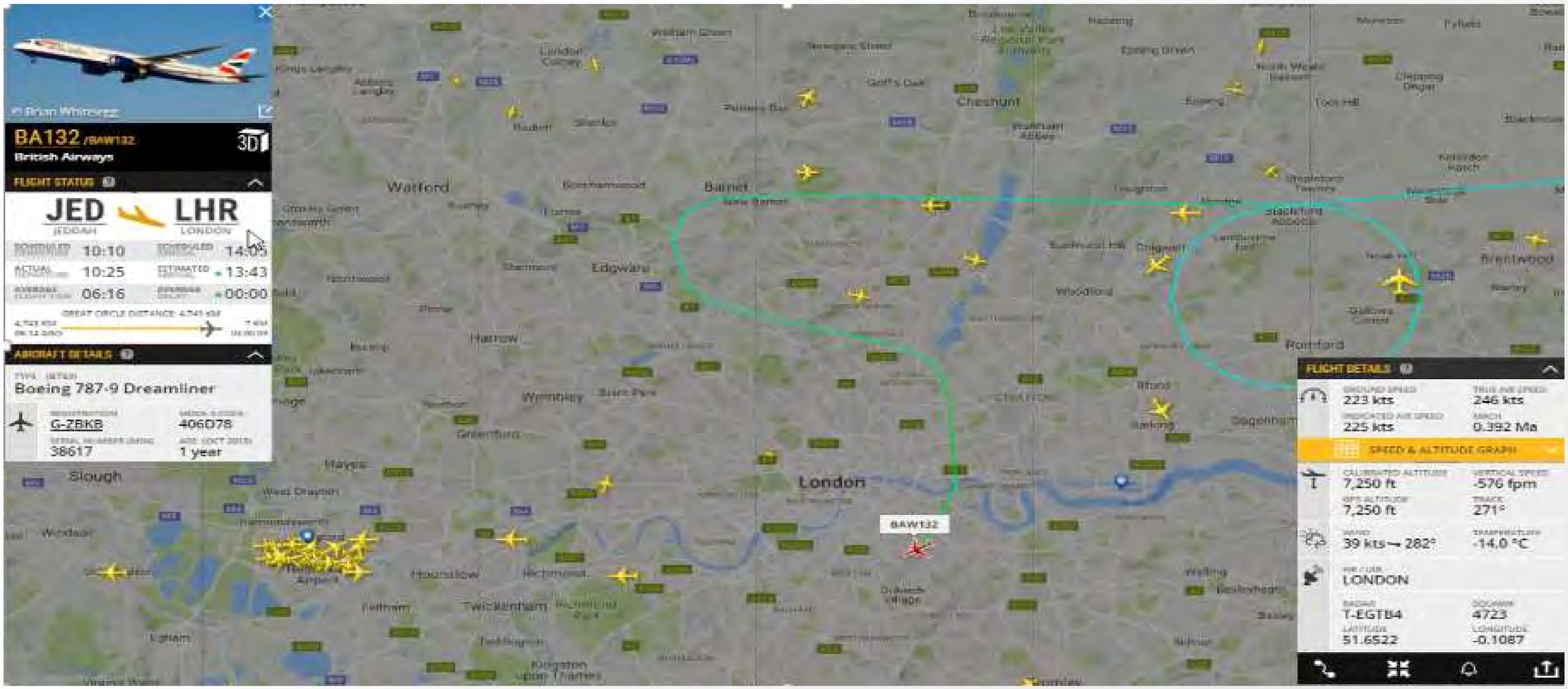
RAdio Detection and Ranging (RADAR)

Automatic Dependent Surveillance – Broadcast (ADS-B)



FlightRadar24

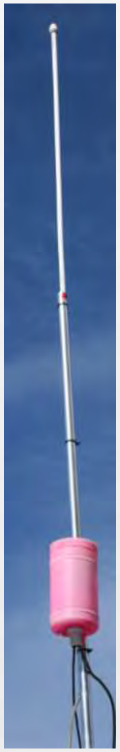
https://www.flightradar24.com/



Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - **Using a Raspberry Pi to capture and decode ADS-B signals**
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

Raspberry PI and USB RTL-SDR



DUMP1090

The “Portable” Raspberry PI

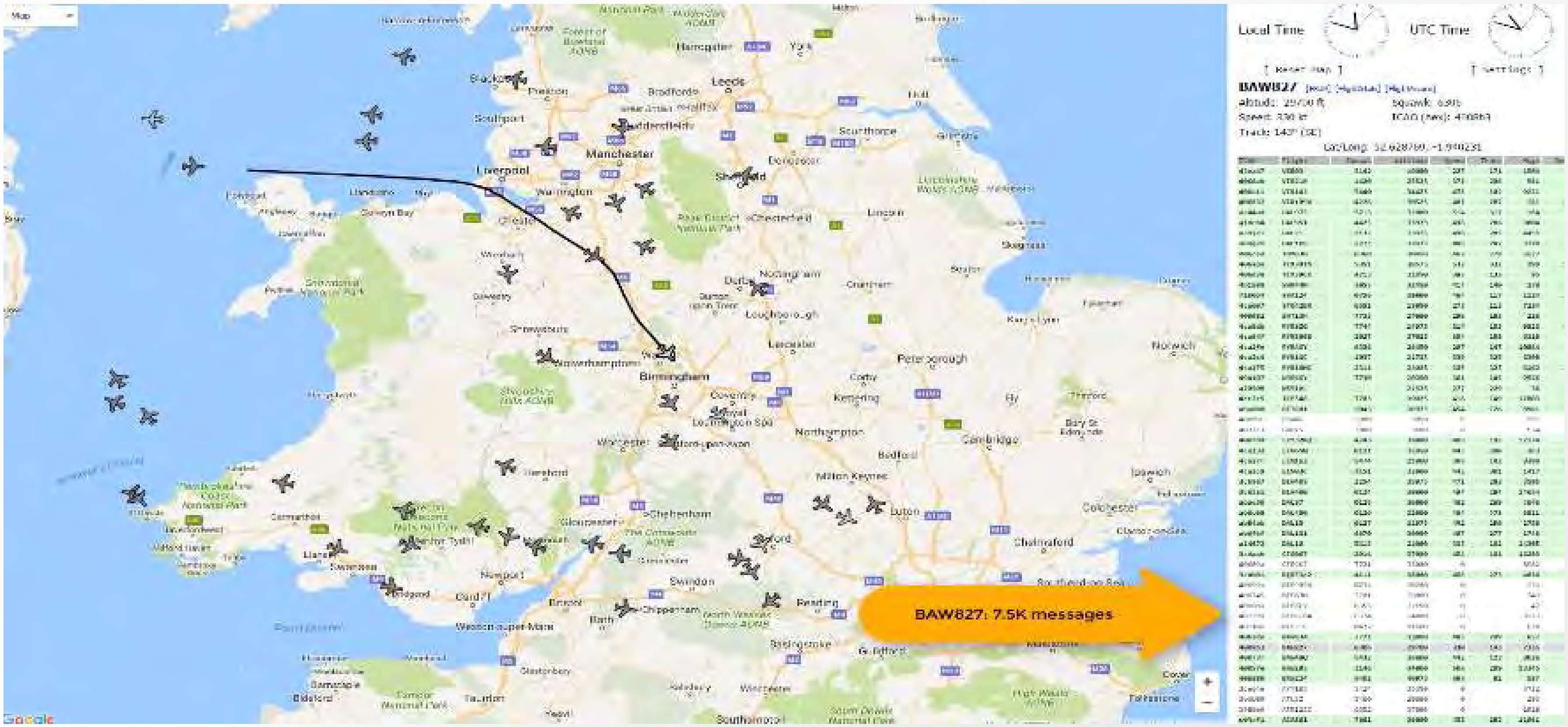
Agenda

- Background to "The Lab Series" and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - **DUMP1090 – Live tracking and streaming**
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What's next?

DUMP1090

- DUMP1090 is a Mode S decoder specifically designed for RTLSDR devices.
- Some of the main features include:
  - Robust decoding of weak messages.
  - Network support: TCP 30003 stream (MSG5...), Raw packets, HTTP.
  - Embedded HTTP server that displays the currently detected aircrafts on Google Map.
  - Single bit errors correction using the 24 bit CRC.
  - Ability to decode DF11, DF17 messages.
  - Ability to decode DF formats like DF0, DF4, DF5, DF16, DF20 and DF21 where the checksum is xored with the ICAO address by brute forcing the checksum field using recently seen ICAO addresses.

Live (DUMP1090) tracking



Analysing DUMP1090 data

pi@pennardpi30: ~

```
pi@pennardpi30:~ $ nc localhost 30003
MSG,8,111,11111,40688F,111111,2017/01/14,07:38:04.672,2017/01/14,07:38:04.638,,,,,,,,,,,,0
MSG,5,111,11111,C0584F,111111,2017/01/14,07:38:04.694,2017/01/14,07:38:04.645,,38975,,,,,,,0,,0,0
MSG,3,111,11111,400691,111111,2017/01/14,07:38:04.710,2017/01/14,07:38:04.700,,38000,,,51.99165,-4.66942,,,,,,0
MSG,1,111,11111,40688F,111111,2017/01/14,07:38:04.734,2017/01/14,07:38:04.705,BAW94   ,,,,,,,,,,,0
MSG,5,111,11111,C0584F,111111,2017/01/14,07:38:04.743,2017/01/14,07:38:04.707,,38975,,,,,,,0,,0,0
MSG,3,111,11111,C0584F,111111,2017/01/14,07:38:04.751,2017/01/14,07:38:04.709,,38975,,,51.90277,-3.98553,,,,,,0
MSG,6,111,11111,C0584F,111111,2017/01/14,07:38:04.768,2017/01/14,07:38:04.764,,,,,,,,2014,0,0,0,0
MSG,7,111,11111,C0584F,111111,2017/01/14,07:38:04.819,2017/01/14,07:38:04.775,,38975,,,,,,,,,,0
MSG,7,111,11111,400691,111111,2017/01/14,07:38:04.837,2017/01/14,07:38:04.831,,38000,,,,,,,,,,0
MSG,5,111,11111,400691,111111,2017/01/14,07:38:04.887,2017/01/14,07:38:04.842,,38000,,,,,,,0,,0,0
MSG,8,111,11111,3C6503,111111,2017/01/14,07:38:04.894,2017/01/14,07:38:04.895,,,,,,,,,,,,0
MSG,5,111,11111,400691,111111,2017/01/14,07:38:04.913,2017/01/14,07:38:04.900,,38000,,,,,,,0,,0,0
MSG,8,111,11111,40688F,111111,2017/01/14,07:38:05.033,2017/01/14,07:38:05.027,,,,,,,,,,,,0
MSG,8,111,11111,40688F,111111,2017/01/14,07:38:05.046,2017/01/14,07:38:05.033,,,,,,,,,,,,0
MSG,8,111,11111,400691,111111,2017/01/14,07:38:05.070,2017/01/14,07:38:05.038,,,,,,,,,,,,0
MSG,8,111,11111,C06111,111111,2017/01/14,07:38:05.071,2017/01/14,07:38:05.039,,,,,,,,,,,,0
MSG,7,111,11111,C0584F,111111,2017/01/14,07:38:05.092,2017/01/14,07:38:05.092,,38975,,,,,,,,,,0
MSG,3,111,11111,C0584F,111111,2017/01/14,07:38:05.266,2017/01/14,07:38:05.229,,38975,,,51.90262,-3.98369,,,,,,0
MSG,8,111,11111,40688F,111111,2017/01/14,07:38:05.290,2017/01/14,07:38:05.289,,,,,,,,,,,,0
MSG,8,111,11111,C06111,111111,2017/01/14,07:38:05.319,2017/01/14,07:38:05.296,,,,,,,,,,,,0
MSG,8,111,11111,C06111,111111,2017/01/14,07:38:05.333,2017/01/14,07:38:05.300,,,,,,,,,,,,0
MSG,8,111,11111,400691,111111,2017/01/14,07:38:05.347,2017/01/14,07:38:05.303,,,,,,,,,,,,0
MSG,6,111,11111,C06111,111111,2017/01/14,07:38:05.362,2017/01/14,07:38:05.355,,,,,,,,6306,0,0,0,0
MSG,4,111,11111,40688F,111111,2017/01/14,07:38:05.364,2017/01/14,07:38:05.357,,,505,113,,,-1984,,,,,0
MSG,8,111,11111,400691,111111,2017/01/14,07:38:05.373,2017/01/14,07:38:05.359,,,,,,,,,,,,0
MSG,8,111,11111,400691,111111,2017/01/14,07:38:05.422,2017/01/14,07:38:05.420,,,,,,,,,,,,0
MSG,8,111,11111,40688F,111111,2017/01/14,07:38:05.425,2017/01/14,07:38:05.424,,,,,,,,,,,,0
MSG,8,111,11111,3C6503,111111,2017/01/14,07:38:05.428,2017/01/14,07:38:05.426,,,,,,,,,,,,0
MSG,8,111,11111,3C6503,111111,2017/01/14,07:38:05.438,2017/01/14,07:38:05.429,,,,,,,,,,,,0
MSG,8,111,11111,C0584F,111111,2017/01/14,07:38:05.443,2017/01/14,07:38:05.430,,,,,,,,,,,,0
MSG,8,111,11111,C0584F,111111,2017/01/14,07:38:05.458,2017/01/14,07:38:05.434,,,,,,,,,,,,0
```

Analysing DUMP1090 data

ID	Type		Description
MSG,1	ES Identification and Category	DF17 BDS 0,8	
MSG,2	ES Surface Position Message	DF17 BDS 0,6	Triggered by nose gear squat switch.
MSG,3	ES Airborne Position Message	DF17 BDS 0,5	
MSG,4	ES Airborne Velocity Message	DF17 BDS 0,9	
MSG,5	Surveillance Alt Message	DF4, DF20	Triggered by ground radar. Not CRC secured. MSG,5 will only be output if the aircraft has previously sent a MSG,1, 2, 3, 4 or 8 signal.
MSG,6	Surveillance ID Message	DF5, DF21	Triggered by ground radar. Not CRC secured. MSG,6 will only be output if the aircraft has previously sent a MSG,1, 2, 3, 4 or 8 signal.
MSG,7	Air To Air Message	DF16	Triggered from TCAS. MSG,7 is now included in the SBS socket output.
MSG,8	All Call Reply	DF11	Broadcast but also triggered by ground radar

Analysing DUMP1090 data

Field 1:	Message type	(MSG, STA, ID, AIR, SEL or CLK)
Field 2:	Transmission Type	MSG sub types 1 to 8. Not used by other message types.
Field 3:	Session ID	Database Session record number
Field 4:	AircraftID	Database Aircraft record number
Field 5:	HexIdent	Aircraft Mode S hexadecimal code
Field 6:	FlightID	Database Flight record number
Field 7:	Date message generated	As it says
Field 8:	Time message generated	As it says
Field 9:	Date message logged	As it says
Field 10:	Time message logged	As it says

Analysing DUMP1090 data

Field 11:	Callsign	An eight digit flight ID - can be flight number or registration (or even nothing).
Field 12:	Altitude	Mode C altitude. Height relative to 1013.2mb (Flight Level). Not height AMSL..
Field 13:	GroundSpeed	Speed over ground (not indicated airspeed)
Field 14:	Track	Track of aircraft (not heading). Derived from the velocity E/W and velocity N/S
Field 15:	Latitude	North and East positive. South and West negative.
Field 16:	Longitude	North and East positive. South and West negative.
Field 17:	VerticalRate	64ft resolution
Field 18:	Squawk	Assigned Mode A squawk code.
Field 19:	Alert (Squawk change)	Flag to indicate squawk has changed.
Field 20:	Emergency	Flag to indicate emergency code has been set
Field 21:	SPI (Ident)	Flag to indicate transponder Ident has been activated.
Field 22:	IsOnGround	Flag to indicate ground squat switch is active

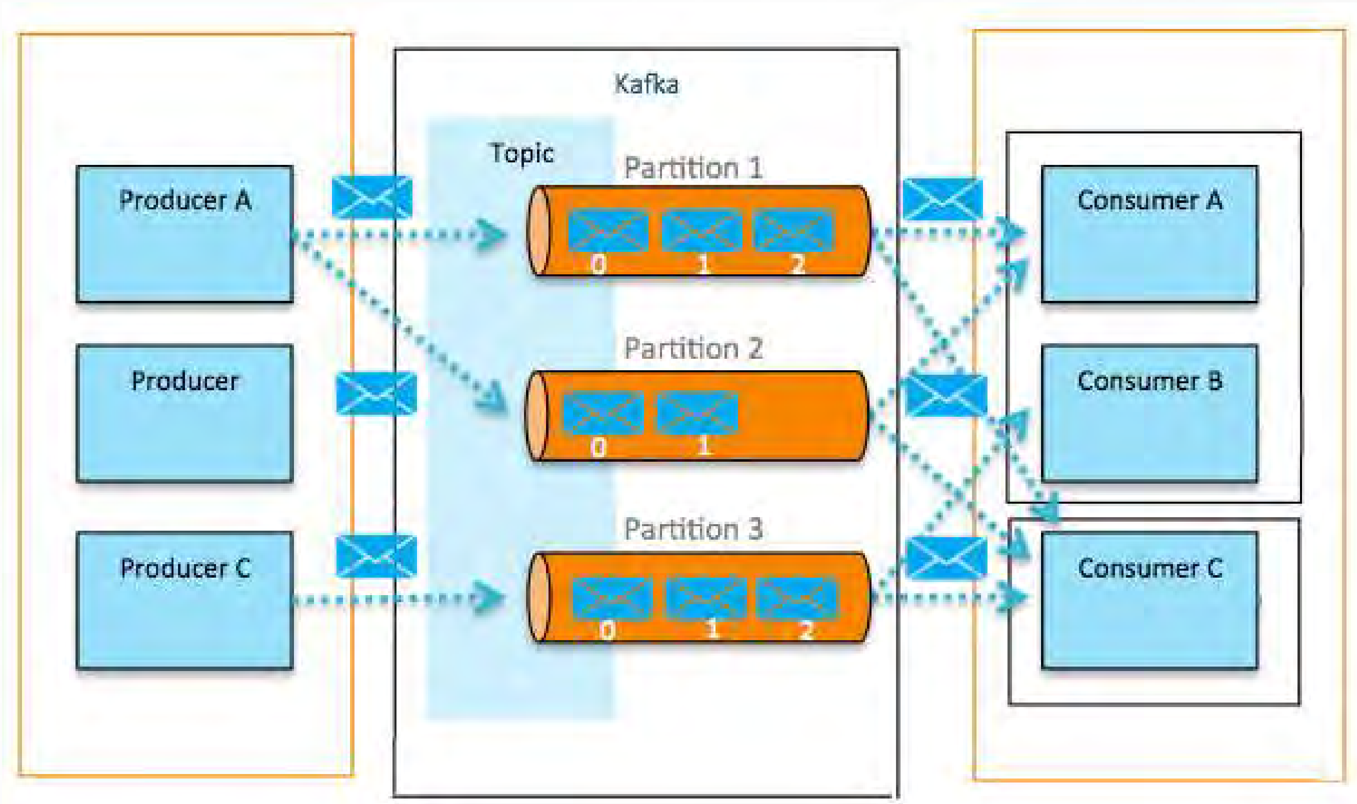
Analysing DUMP1090 data

	1	2	3	4	5	6	7	8	9	10		11	12	13	14	15	16	17	18	19	20	21	22
MSG 1	MT	TT	SID	AID	Hex	FID	DMG	TMG	DML	TML		CS											
MSG 2													Alt	GS	Trk	Lat	Lng						Gnd
MSG 3													Alt			Lat	LNG			Alrt	Emer	SPI	Gnd
MSG 4														GS	Trk			VR					
MSG 5													Alt							Alrt		SPI	Gnd
MSG 6													Alt						Sq	Alrt	Emer	SPI	Gnd
MSG 7													Alt										Gnd
MSG 8																							Gnd

Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - **Apache Kafka and Extract Transform & Load (ETL)**
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

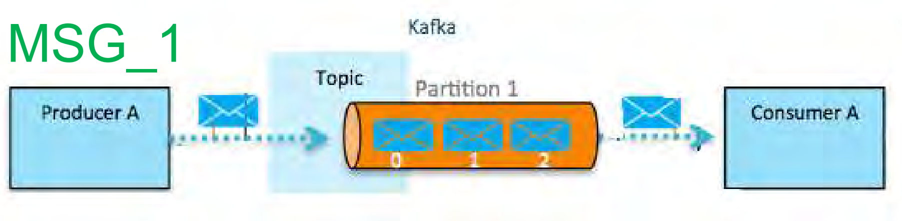
Defining KAFKA Topics



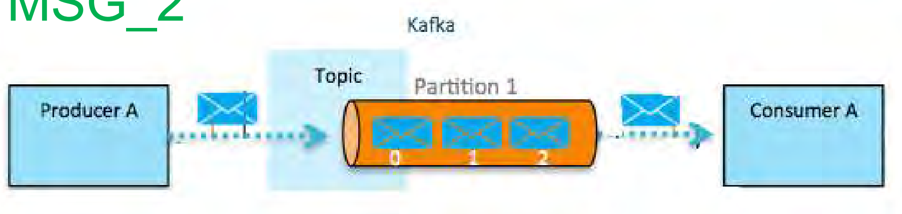
A distributed streaming platform

Defining KAFKA Topics

MSG_1



MSG_2



. . .

MSG_8

Topics:

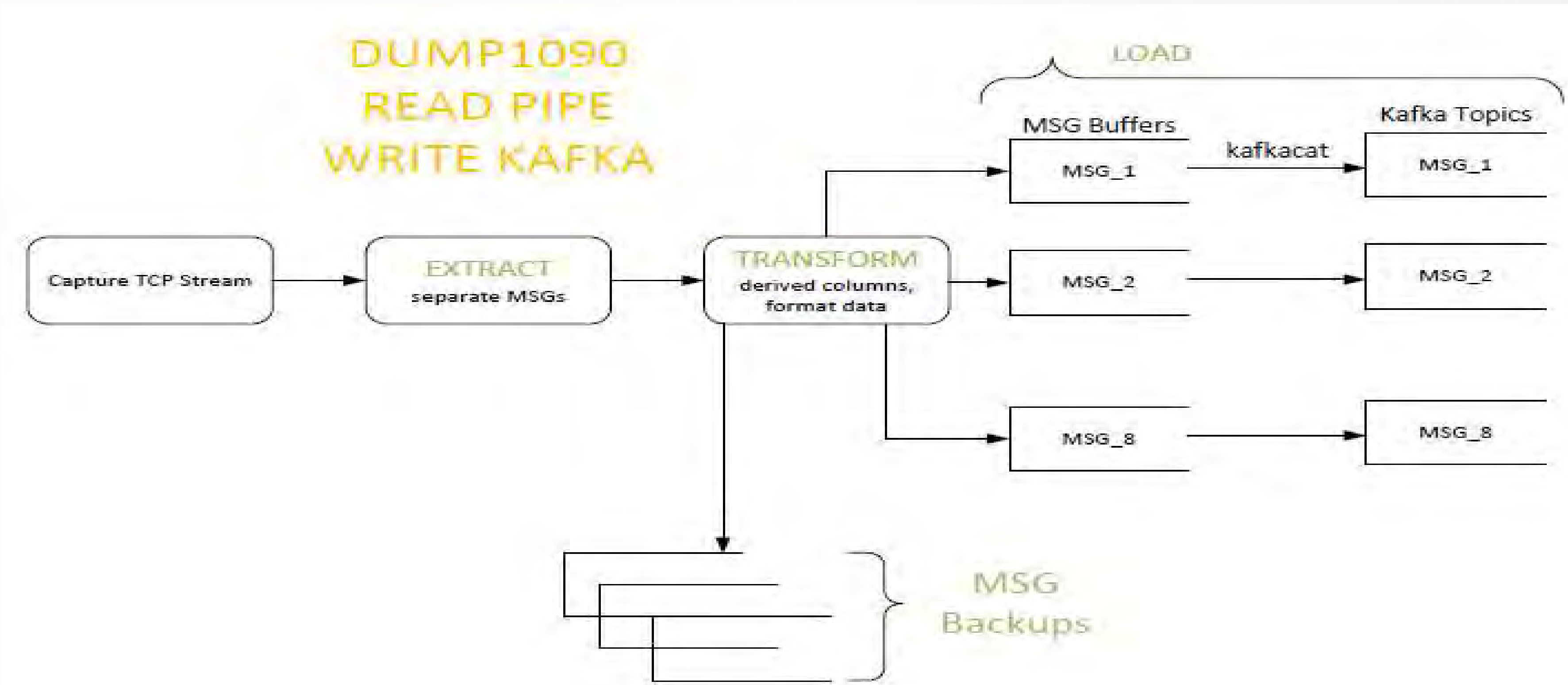
One per message type (MSG_1, MSG_2 etc)

Single broker

Single partition (per topics)

APACHE kafka
A distributed streaming platform

Feeding DUMP1090 data into KAFKA topics



Kafka Topics – Receiving messages

```
pennardpi01,MSG,1,333,49132,4CA2C9,49232,2017/05/21 07:16:01.171,2017/05/21 07:16:01.171,EIN5LM
pennardpi01,MSG,1,333,49132,4CA2C9,49232,2017/05/21 07:16:01.171,2017/05/21 07:16:01.171,EIN5LM
% Reached end of topic dump1090_msg_1 [0] at offset 11071103
pennardpi01,MSG,1,333,49048,4CABAB,49148,2017/05/21 07:16:02.075,2017/05/21 07:16:02.075,RYR4DG
pennardpi01,MSG,1,333,49048,4CABAB,49148,2017/05/21 07:16:02.075,2017/05/21 07:16:02.075,RYR4DG
pennardpi01,MSG,1,333,49048,4CABAB,49148,2017/05/21 07:16:02.075,2017/05/21 07:16:02.075,RYR4DG
pennardpi01,MSG,1,333,49048,4CABAB,49148,2017/05/21 07:16:02.075,2017/05/21 07:16:02.075,RYR4DG
pennardpi01,MSG,1,333,49048,4CABAB,49148,2017/05/21 07:16:02.075,2017/05/21 07:16:02.075,RYR4DG
pennardpi01,MSG,1,333,49151,396670,49251,2017/05/21 07:16:02.163,2017/05/21 07:16:02.163,FPO725
pennardpi01,MSG,1,333,49151,396670,49251,2017/05/21 07:16:02.163,2017/05/21 07:16:02.163,FPO725
pennardpi01,MSG,1,333,49151,396670,49251,2017/05/21 07:16:02.163,2017/05/21 07:16:02.163,FPO725
pennardpi01,MSG,1,333,49151,396670,49251,2017/05/21 07:16:02.163,2017/05/21 07:16:02.163,FPO725
pennardpi01,MSG,1,333,49151,396670,49251,2017/05/21 07:16:02.163,2017/05/21 07:16:02.163,FPO725
pennardpi01,MSG,1,333,49069,4CA295,49169,2017/05/21 07:16:02.177,2017/05/21 07:16:02.177,EIN63N
pennardpi01,MSG,1,333,49069,4CA295,49169,2017/05/21 07:16:02.177,2017/05/21 07:16:02.177,EIN63N
pennardpi01,MSG,1,333,49069,4CA295,49169,2017/05/21 07:16:02.177,2017/05/21 07:16:02.177,EIN63N
pennardpi01,MSG,1,333,49069,4CA295,49169,2017/05/21 07:16:02.177,2017/05/21 07:16:02.177,EIN63N
pennardpi01,MSG,1,333,49069,4CA295,49169,2017/05/21 07:16:02.177,2017/05/21 07:16:02.177,EIN63N
pennardpi01,MSG,1,333,49186,896318,49286,2017/05/21 07:16:02.198,2017/05/21 07:16:02.198,UAE21
pennardpi01,MSG,1,333,49186,896318,49286,2017/05/21 07:16:02.198,2017/05/21 07:16:02.198,UAE21
pennardpi01,MSG,1,333,49186,896318,49286,2017/05/21 07:16:02.198,2017/05/21 07:16:02.198,UAE21
pennardpi01,MSG,1,333,49186,896318,49286,2017/05/21 07:16:02.198,2017/05/21 07:16:02.198,UAE21
pennardpi01,MSG,1,333,49186,896318,49286,2017/05/21 07:16:02.198,2017/05/21 07:16:02.198,UAE21
% Reached end of topic dump1090_msg_1 [0] at offset 11071123
pennardpi01,MSG,1,333,48914,3C4B01,49014,2017/05/21 07:16:03.070,2017/05/21 07:16:03.070,BER7417
pennardpi01,MSG,1,333,48914,3C4B01,49014,2017/05/21 07:16:03.070,2017/05/21 07:16:03.070,BER7417
pennardpi01,MSG,1,333,48914,3C4B01,49014,2017/05/21 07:16:03.070,2017/05/21 07:16:03.070,BER7417
pennardpi01,MSG,1,333,48914,3C4B01,49014,2017/05/21 07:16:03.070,2017/05/21 07:16:03.070,BER7417
pennardpi01,MSG,1,333,48914,3C4B01,49014,2017/05/21 07:16:03.070,2017/05/21 07:16:03.070,BER7417
pennardpi01,MSG,1,333,49247,4071DF,49347,2017/05/21 07:16:03.118,2017/05/21 07:16:03.118,EZY63DW
pennardpi01,MSG,1,333,49247,4071DF,49347,2017/05/21 07:16:03.118,2017/05/21 07:16:03.118,EZY63DW
pennardpi01,MSG,1,333,49247,4071DF,49347,2017/05/21 07:16:03.118,2017/05/21 07:16:03.118,EZY63DW
pennardpi01,MSG,1,333,49247,4071DF,49347,2017/05/21 07:16:03.118,2017/05/21 07:16:03.118,EZY63DW
pennardpi01,MSG,1,333,49247,4071DF,49347,2017/05/21 07:16:03.118,2017/05/21 07:16:03.118,EZY63DW
pennardpi01,MSG,1,333,49240,4008AD,49340,2017/05/21 07:16:03.633,2017/05/21 07:16:03.633,SHT73
pennardpi01,MSG,1,333,49240,4008AD,49340,2017/05/21 07:16:03.633,2017/05/21 07:16:03.633,SHT73
pennardpi01,MSG,1,333,49240,4008AD,49340,2017/05/21 07:16:03.633,2017/05/21 07:16:03.633,SHT73
pennardpi01,MSG,1,333,49240,4008AD,49340,2017/05/21 07:16:03.633,2017/05/21 07:16:03.633,SHT73
pennardpi01,MSG,1,333,49240,4008AD,49340,2017/05/21 07:16:03.633,2017/05/21 07:16:03.633,SHT73
% Reached end of topic dump1090_msg_1 [0] at offset 11071138
```

Agenda

- Background to "The Lab Series" and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - **Introduction to Vertica**
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What's next?

How the World's Leading Data-Driven Businesses Came to Rely on Vertica

Our Customers:
The World's Most Data-Driven Enterprises

"Digital Darwinism is unkind to those who wait."

- Ray Wang, Constellation Research, June 2015

VERTICA Was Born

C-Store: A Column-oriented DBMS

Mike Stonebraker[*], Daniel J. Abadi[*], Adam Batkin[+], Xuedong Chen[†], Mitch Cherniack[+], Miguel Ferreira[*], Edmond Lau[*], Amerson Lin[*], Sam Madden[*], Elizabeth O'Neil[†], Pat O'Neil[†], Alex Rasin[‡], Nga Tran[+], Stan Zdonik[‡]

[*]MIT CSAIL, Cambridge, MA
[+]Brandeis University, Waltham, MA
[†]UMass Boston, Boston, MA
[‡]Brown University, Providence, RI

Abstract

This paper presents the design of a read-optimized relational DBMS that contrasts sharply with most current systems, which are write-optimized. Among the many differences in its design are: storage of data by column rather than by row, careful coding and packing of objects into storage including main memory during query processing, storing an overlapping collection of column-oriented projections, rather than the current fare of tables and indexes, a non-traditional implementation of transactions which includes high availability and snapshot isolation for read-only transactions, and the extensive use of bitmap indexes to complement B-tree structures.

We present preliminary performance data on a subset of TPC-H and show that the system we are building, C-Store, is substantially faster than popular commercial products. Hence, the architecture looks very encouraging.

1. Introduction

Most major DBMS vendors implement record-oriented storage systems, where the attributes of a record (or tuple) are placed contiguously in storage. With this *row store* architecture, a single disk write suffices to push all of the fields of a single record out to disk. Hence, high performance writes are achieved, and we call a DBMS with a row store architecture a *write-optimized* system. These are especially effective on OLTP-style applications.

In contrast, systems oriented toward ad-hoc querying in which periodically a bulk load of new data is performed, followed by a relatively long period of ad-hoc queries. Other read-mostly applications include customer relationship management (CRM) systems, electronic library card catalogs, and other ad-hoc inquiry systems. In such environments, a *column store* architecture, in which the values for each single column (or attribute) are stored contiguously, should be more efficient. This efficiency has been demonstrated in the warehouse marketplace by products like Sybase IQ [FREN95, SYBA04], Addamark [ADDA04], and KDB [KDB04]. In this paper, we discuss the design of a column store called C-Store that includes a number of novel features relative to existing systems.

With a column store architecture, a DBMS need only read the values of columns required for processing a given query, and can avoid bringing into memory irrelevant attributes. In warehouse environments where typical queries involve aggregates performed over large numbers of data items, a column store has a sizeable performance advantage. However, there are several other major distinctions that can be drawn between an architecture that is read-optimized and one that is write-optimized.

Current relational DBMSs were designed to pad attributes to byte or word boundaries and to store values in their native data format. It was thought that it was too expensive to shift data values onto byte or word boundaries in main memory for processing. However, CPUs are getting faster at a much greater rate than disk bandwidth is increasing. Hence, it makes sense to trade CPU cycles, which are abundant, for disk bandwidth, which is not. This tradeoff appears especially profitable in a read-mostly environment.

Mike Stonebraker

VERTICA Foundation

Columnar Storage

Speeds query time by reading only necessary data

Compression

Lowers costly I/O to boost overall performance

MPP Scale-out

Provides high scalability on clusters with no name node or other single point of failure

Distributed Query

Any node can initiate the queries and use other nodes for work. No single point of failure

Projections

Combine high availability with special optimizations for query performance

An Open Architecture Integrated with Rich Ecosystem

Data Transformation

Spark

Messaging

kafka

ETL

ATTUNITY

informatica

USER
DEFINED
LOADS

User-Defined Functions

R	Java	C++	Python	SQL

Geospatial	Real-Time	Text Analytics
Event Series	VERTICA	Pattern Matching
Time Series	Machine Learning	Regression

User Defined Storage

Security

External tables to analyze in place

Parquet amazon web services S3 hadoop Apache orc

ODBC
JDBC
OLEDB

BI & Visualization

looker

Qlik

tableau

Logi ANALYTICS

Microsoft Azure · amazon web services · Google Cloud Platform · openstack · hadoop

VERTICA

MICRO FOCUS

Machine Learning at Enterprise Scale Is the **Future** of All Predictive Analytics

VERTICA

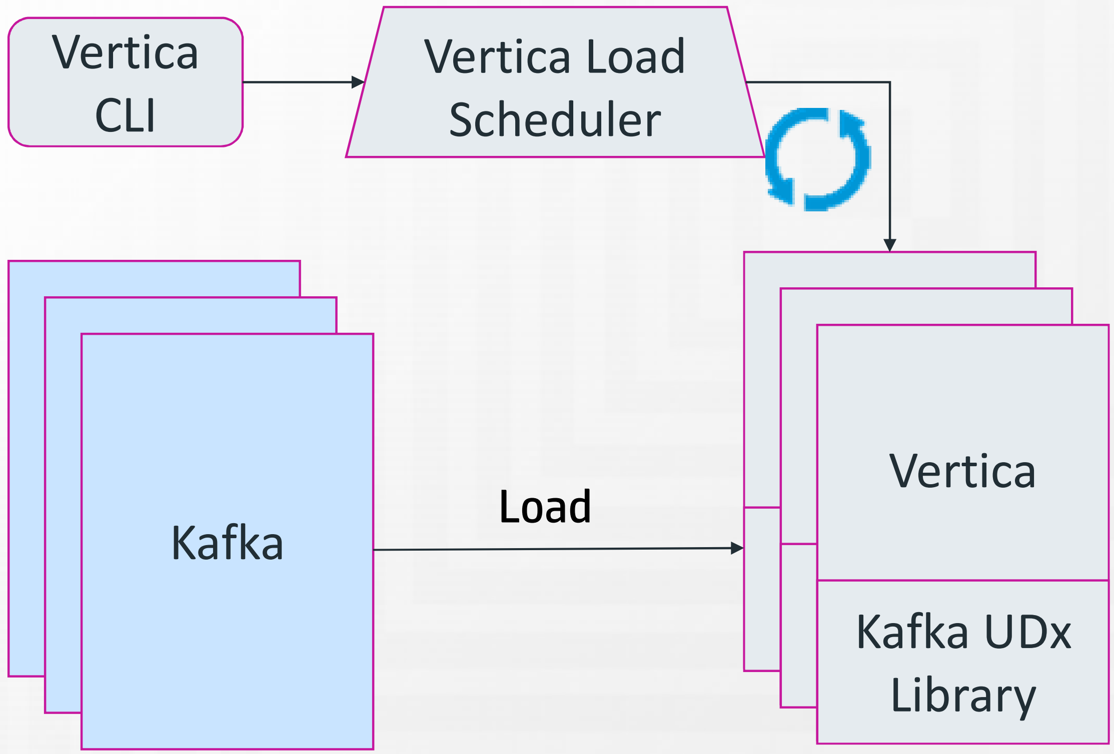
MICRO FOCUS

Agenda

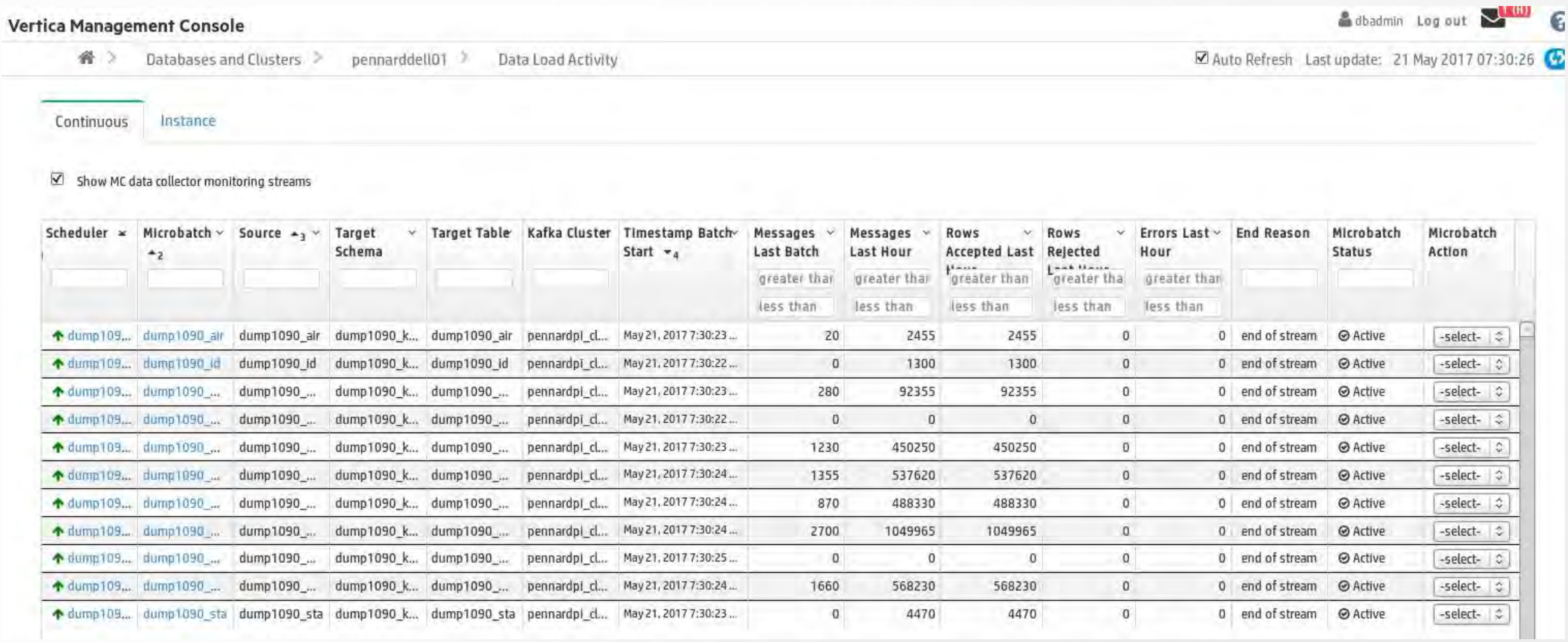
- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - **Kafka / Vertica integration and Management Console**
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

Kafka Integration with Vertica

- Vertica schedules loads to continuously **consume** from Kafka
- JSON, Avro data formats
- CLI for easy setup
- In-database monitoring



Monitoring


Vertica Management Console

dbadmin Log out

Databases and Clusters > pennarddell01 > Data Load Activity

Auto Refresh Last update: 21 May 2017 07:30:26

Continuous | Instance

Show MC data collector monitoring streams

Scheduler	Microbatch	Source	Target Schema	Target Table	Kafka Cluster	Timestamp Batch Start	Messages Last Batch	Messages Last Hour	Rows Accepted Last Hour	Rows Rejected Last Hour	Errors Last Hour	End Reason	Microbatch Status	Microbatch Action
dump109...	dump1090_air	dump1090_air	dump1090_k...	dump1090_air	pennardpi_cl...	May 21, 2017 7:30:23 ...	20	2455	2455	0	0	end of stream	Active	-select-
dump109...	dump1090_id	dump1090_id	dump1090_k...	dump1090_id	pennardpi_cl...	May 21, 2017 7:30:22 ...	0	1300	1300	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:23 ...	280	92355	92355	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:22 ...	0	0	0	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:23 ...	1230	450250	450250	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:24 ...	1355	537620	537620	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:24 ...	870	488330	488330	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:24 ...	2700	1049965	1049965	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:25 ...	0	0	0	0	0	end of stream	Active	-select-
dump109...	dump1090_...	dump1090_...	dump1090_k...	dump1090_...	pennardpi_cl...	May 21, 2017 7:30:24 ...	1660	568230	568230	0	0	end of stream	Active	-select-
dump109...	dump1090_sta	dump1090_sta	dump1090_k...	dump1090_sta	pennardpi_cl...	May 21, 2017 7:30:23 ...	0	4470	4470	0	0	end of stream	Active	-select-

Monitoring

Scheduler ▲1	Microbatch ▲2	Source ▲3	Target Schema	Target Table	Kafka Cluster	Timestamp Batch Start ▼4
dump1090Scheduler	dump1090_air	dump1090_air	dump1090_kafka	dump1090_air	pennardpi_cluster	May 21, 2017 7:32:22 AM
dump1090Scheduler	dump1090_id	dump1090_id	dump1090_kafka	dump1090_id	pennardpi_cluster	May 21, 2017 7:32:22 AM
dump1090Scheduler	dump1090_msg_1	dump1090_msg_1	dump1090_kafka	dump1090_msg_1	pennardpi_cluster	May 21, 2017 7:32:23 AM
dump1090Scheduler	dump1090_msg_2	dump1090_msg_2	dump1090_kafka	dump1090_msg_2	pennardpi_cluster	May 21, 2017 7:32:23 AM
dump1090Scheduler	dump1090_msg_3	dump1090_msg_3	dump1090_kafka	dump1090_msg_3	pennardpi_cluster	May 21, 2017 7:32:24 AM
dump1090Scheduler	dump1090_msg_4	dump1090_msg_4	dump1090_kafka	dump1090_msg_4	pennardpi_cluster	May 21, 2017 7:32:24 AM
dump1090Scheduler	dump1090_msg_5	dump1090_msg_5	dump1090_kafka	dump1090_msg_5	pennardpi_cluster	May 21, 2017 7:32:23 AM
dump1090Scheduler	dump1090_msg_6	dump1090_msg_6	dump1090_kafka	dump1090_msg_6	pennardpi_cluster	May 21, 2017 7:32:24 AM

Monitoring

- Manage

Messages Last Batch	Messages Last Hour	Rows Accepted Last Hour	Rows Rejected Last Hour	Errors Last Hour	End Reason
greater than	greater than	greater than	greater tha	greater than	
less than	less than	less than	less than	less than	
0	2405	2405	0	0	end of stream
0	1305	1305	0	0	end of stream
255	92310	92310	0	0	end of stream
0	0	0	0	0	end of stream
1365	451845	451845	0	0	end of stream
1469	538294	538294	0	0	end of stream
1040	497130	497130	0	0	end of stream
2738	1051552	1051552	0	0	end of stream

Monitoring

This microbatch is collecting from the following sources: **dump1090_msg_2 (hosts:pennardpi10:9092)**

Details

Node name	v_pennarddell01_node0001
SessionId	nnarddell01_node0001-10004:0x2fa9
TransactionId	45035996288966917
StatementId	7
Batch Number	0

Rejection

Row Number	1
Rejected Data Orig Length	113
Rejected Reason	Invalid integer format '51.60221' for column 13 (is_on_ground)
Rejected Data	pennardpi01,MSG,2,333,17056,4049C8,17156,2016/11/29 12:08:36.216,2016/11/29 12:08:36.216,0,,,51.60221,-4.06773,-1

Query 'dump1090_kafka.dump1090_msg_2_rej WHERE transaction_id = 45035996288966917 AND statement_id = 7' to get all the rejected rows.

How many? How fast?

Scheduler ▲1	Microbatch ▲2	Source ▲3	Target Schema	Target Table	Kafka Cluster	Timestamp Batch Start ▼4	Messages Last Batch	Messages Last Hour	Rows Accepted Last	Rows Rejected	Errors Last Hour	End Reason
							greater than / less than	greater than / less than	greater than / less than	greater tha / less than	greater than / less than	
dump1090Sche...	dump1090_air	dump1090_air	dump1090_kafka	dump1090_air	pennardpi_cluster	May 21, 2017 8:50:29 ...	0	189650	189650	0	0	end of stream
dump1090Sche...	dump1090_id	dump1090_id	dump1090_kafka	dump1090_id	pennardpi_cluster	May 21, 2017 8:50:28 ...	0	104121	104121	0	0	end of stream
dump1090Sche...	dump1090_msg_1	dump1090_msg_1	dump1090_kafka	dump1090_msg_1	pennardpi_cluster	May 21, 2017 8:50:29 ...	164215	5977750	5977750	0	0	deadline
dump1090Sche...	dump1090_msg_2	dump1090_msg_2	dump1090_kafka	dump1090_msg_2	pennardpi_cluster	May 21, 2017 8:50:29 ...	0	927	0	927	0	end of stream
dump1090Sche...	dump1090_msg_3	dump1090_msg_3	dump1090_kafka	dump1090_msg_3	pennardpi_cluster	May 21, 2017 8:50:30 ...	108465	5061012	5061012	0	0	deadline
dump1090Sche...	dump1090_msg_4	dump1090_msg_4	dump1090_kafka	dump1090_msg_4	pennardpi_cluster	May 21, 2017 8:50:32 ...	123785	5688580	5688580	0	0	deadline
dump1090Sche...	dump1090_msg_5	dump1090_msg_5	dump1090_kafka	dump1090_msg_5	pennardpi_cluster	May 21, 2017 8:50:33 ...	117434	5625492	5625492	0	0	deadline
dump1090Sche...	dump1090_msg_6	dump1090_msg_6	dump1090_kafka	dump1090_msg_6	pennardpi_cluster	May 21, 2017 8:50:20 ...	127092	5345799	5441082	0	0	deadline
dump1090Sche...	dump1090_msg_7	dump1090_msg_7	dump1090_kafka	dump1090_msg_7	pennardpi_cluster	May 21, 2017 8:50:29 ...	0	0	0	0	0	end of stream
dump1090Sche...	dump1090_msg_8	dump1090_msg_8	dump1090_kafka	dump1090_msg_8	pennardpi_cluster	May 21, 2017 8:50:19 ...	173224	5815234	5842579	0	0	deadline
dump1090Sche...	dump1090_sta	dump1090_sta	dump1090_kafka	dump1090_sta	pennardpi_cluster	May 21, 2017 8:50:28 ...	10	379396	379396	0	0	end of stream

How many? How fast?

Timestamp Batch Start	Messages Last Batch
May 21, 2017 8:50:29 ...	0
May 21, 2017 8:50:28 ...	0
May 21, 2017 8:50:29 ...	164215
May 21, 2017 8:50:29 ...	0
May 21, 2017 8:50:30 ...	108465
May 21, 2017 8:50:32 ...	123785
May 21, 2017 8:50:33 ...	117434
May 21, 2017 8:50:20 ...	127092
May 21, 2017 8:50:29 ...	0
May 21, 2017 8:50:19 ...	173224
May 21, 2017 8:50:28 ...	10

- Micro batch: 10 seconds
- Messages loaded into Vertica in last micro batch: 814,225
  - 4.9 million / minute
  - 293 million / hour
  - 7 billion / day

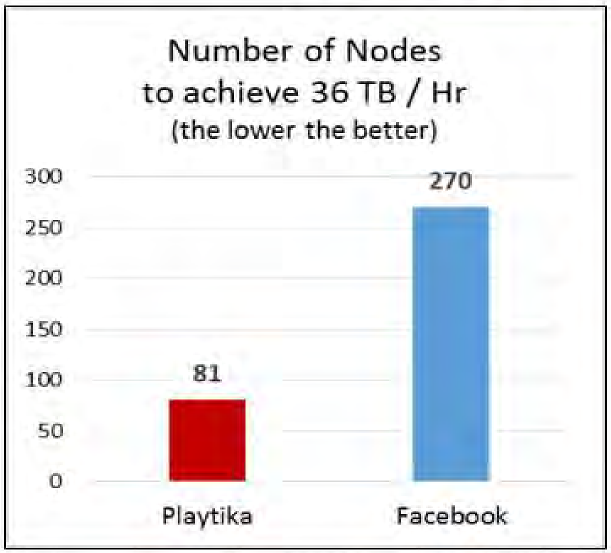
- All from:
  - 1x Raspberry PI (1 core, 0.5GB RAM) running DUMP1090
  - 1x Raspberry PI (1 core, 0.5GB RAM) running ETL, Zookeeper & Kafka
  - 1x Laptop (8 core, 16GB RAM) CentOS 7 running:
    - Vertica (single node)

How many? How fast?

Impressive Parallel COPY Performance

Loaded **2.42 Billion Rows (451 GB)**

in **7min 35sec** on an **8 Node Cluster**



Key Takeaways

- Parallel Kafka Reads to Spark RDD *(in memory)* with Parallel writes to a Vertica via tcp server – ROCKS!
- COPY 36 TB/Hour with 81 Node cluster
- No ephemeral nodes needed for ingest
- Kafka read parallelism to Spark RDD partitions
- A priori hash() in Spark RDD Partitions *(in Memory)*
- TCP Server as a Vertica User Define Copy Source
- Single COPY does not preallocate Memory across nodes

http://www.vertica.com/2014/09/17/how-vertica-met-facebooks-35-tbhour-ingest-sla/

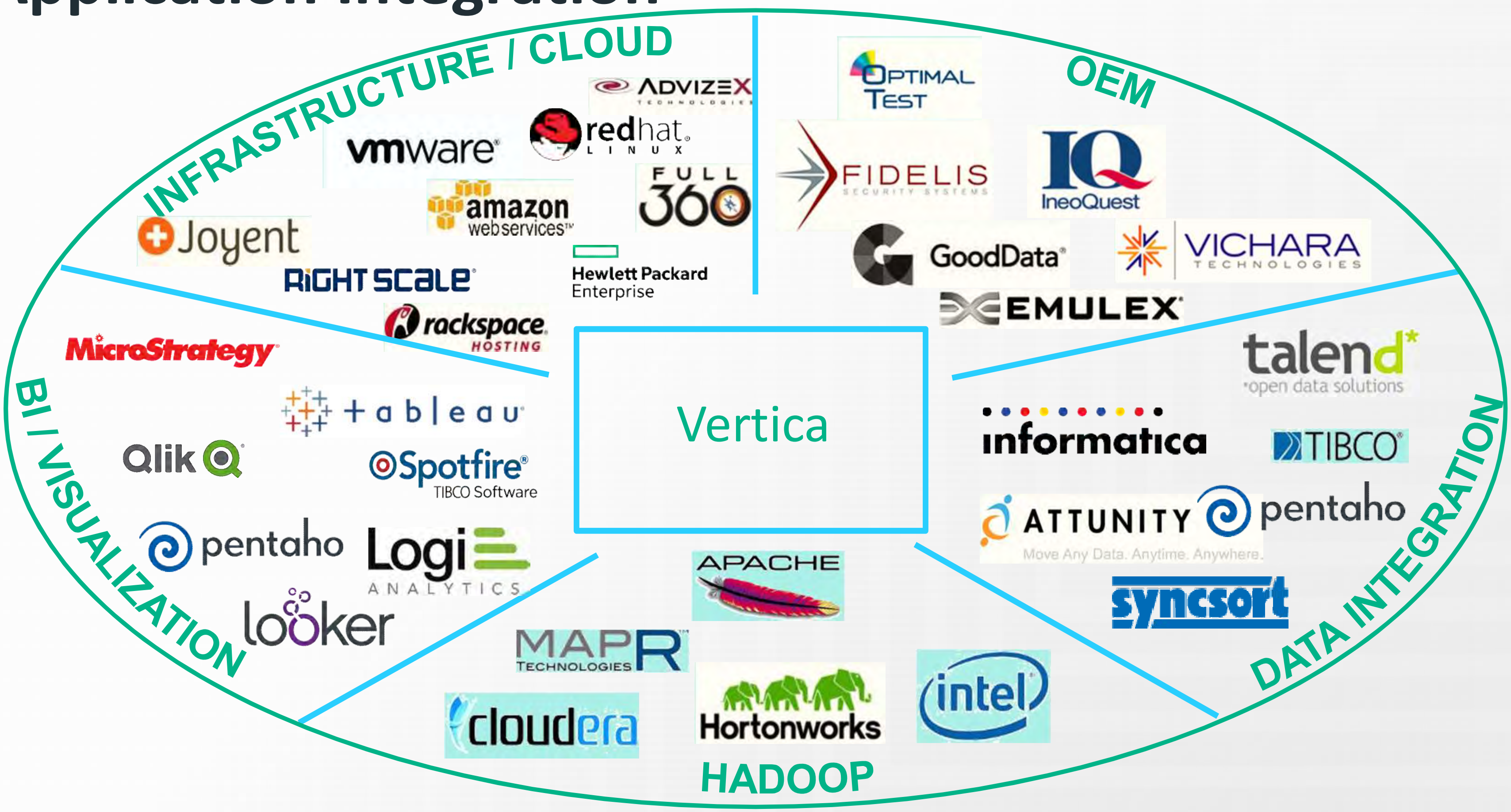
* 270 Nodes (45 Ingest Nodes + 215 Data Nodes [225 ?])

Playtika

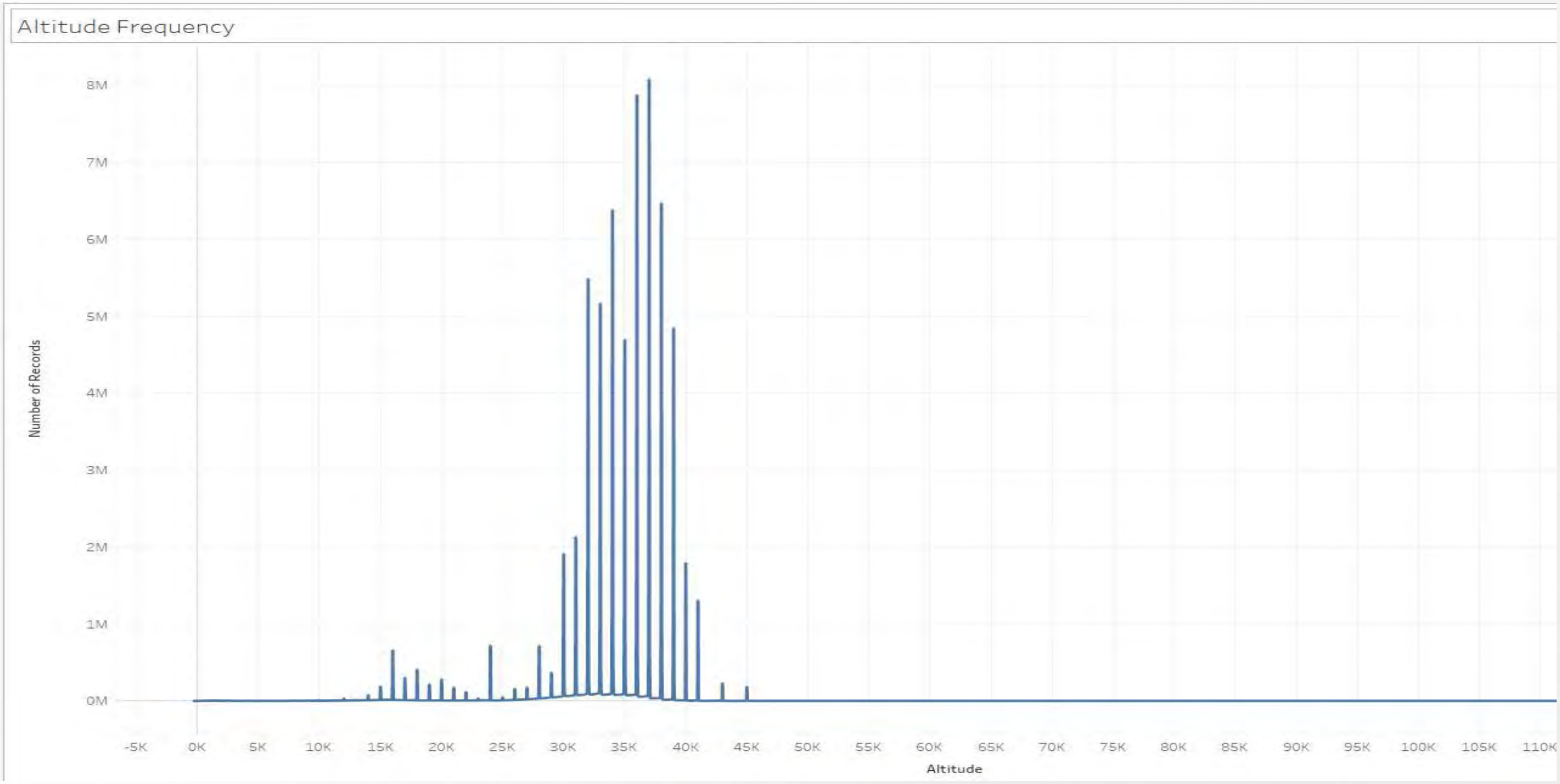
Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - **Vertica integration tools and simple visualisations**
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What’s next?

Application Integration



Altitude


Altitude Frequency
Number of Records
8M
7M
6M
5M
4M
3M
2M
1M
0M
-5K
0K
5K
10K
15K
20K
25K
30K
35K
40K
45K
50K
55K
60K
65K
70K
75K
80K
85K
90K
95K
100K
105K
110K
Altitude

Geospatial

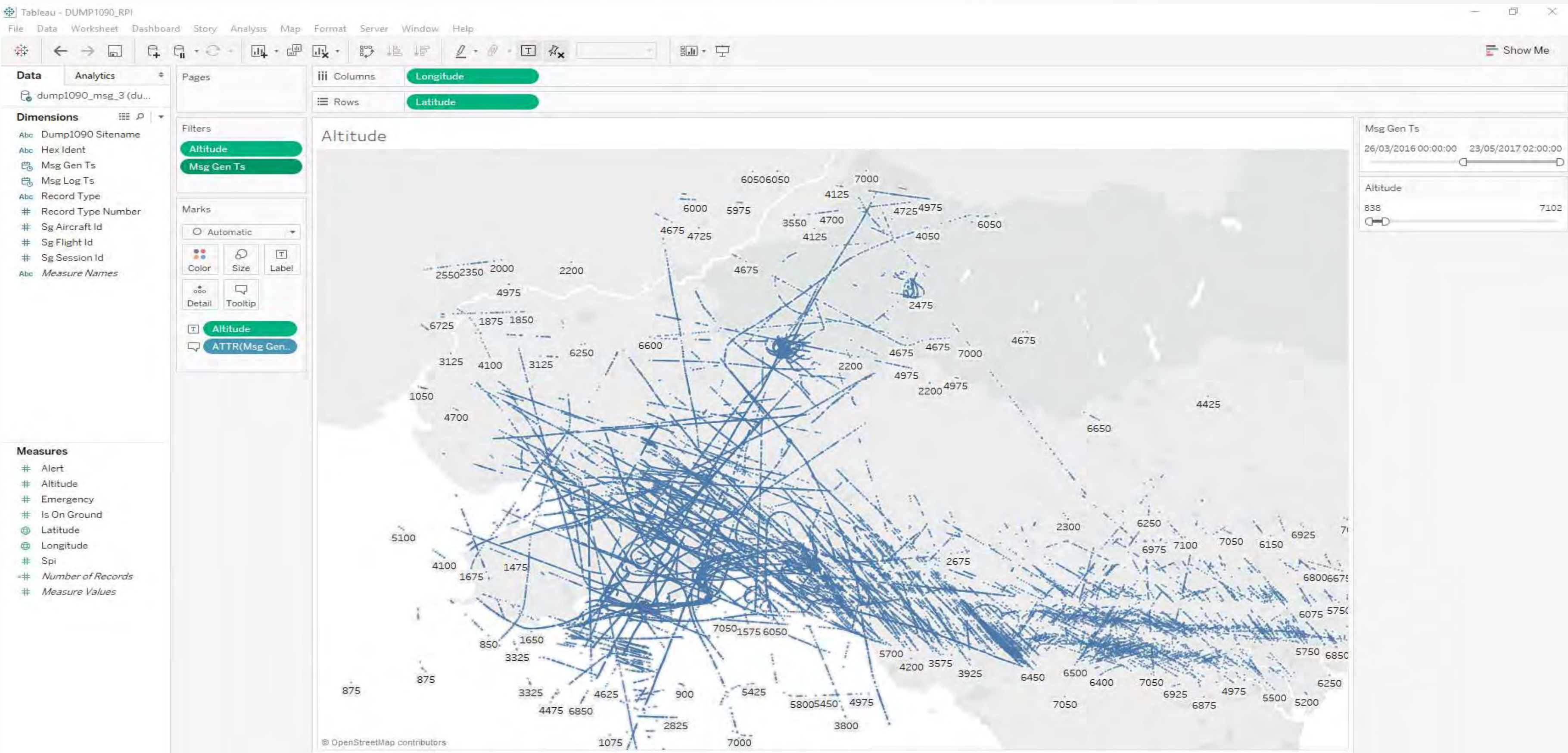
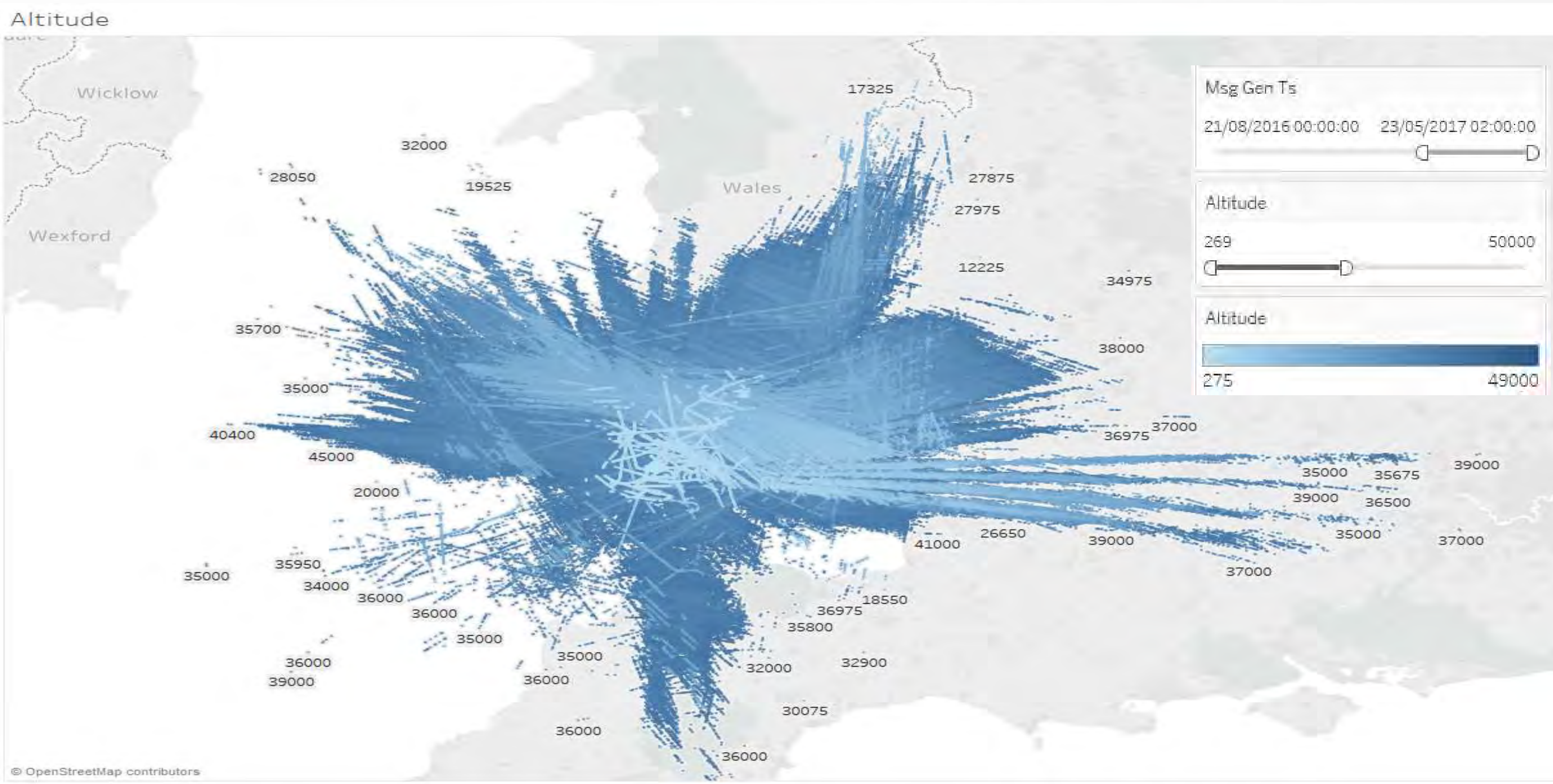
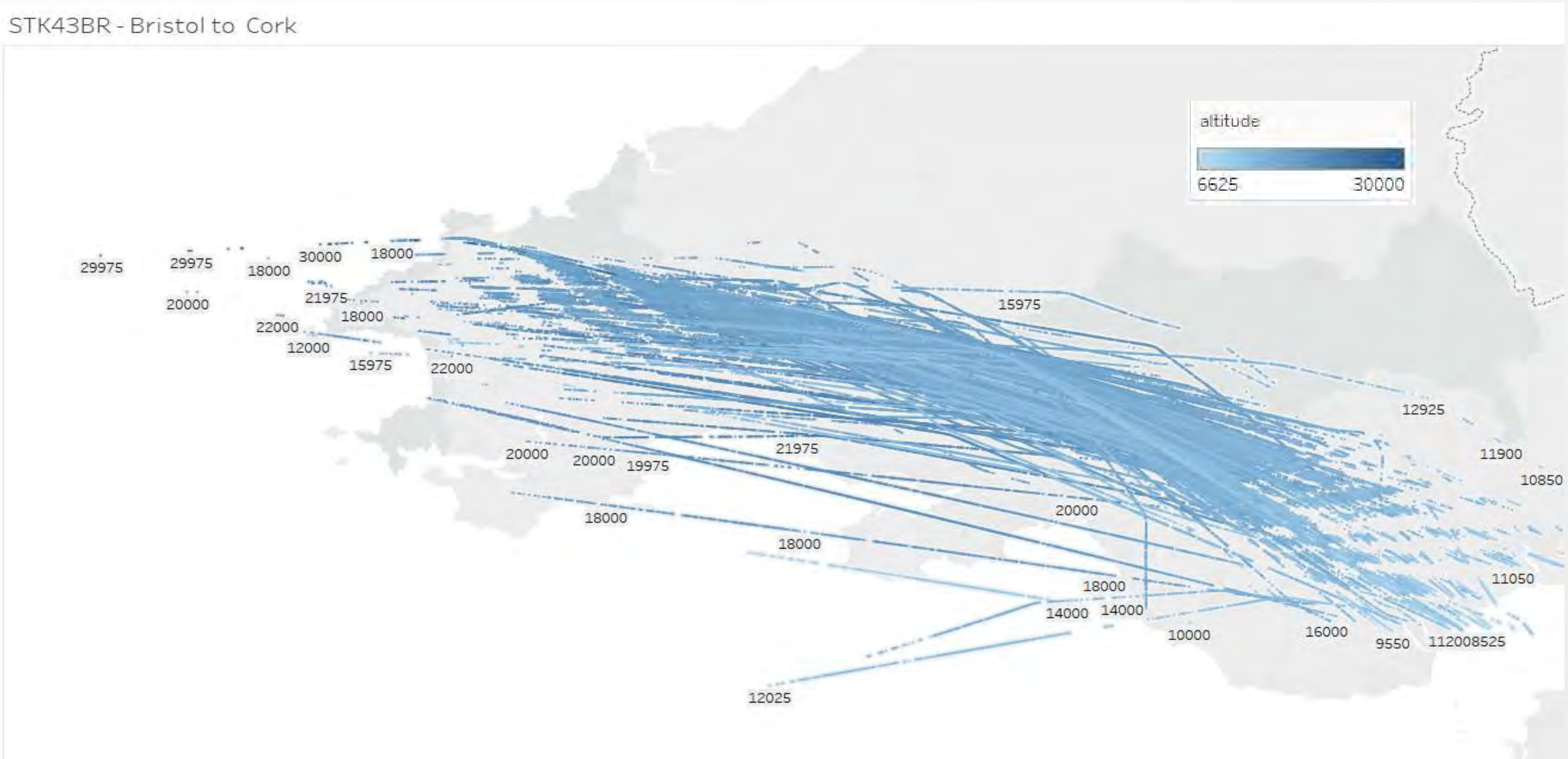


Tableau - DUMP1090_RPI
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Show Me
Data
Analytics
dump1090_msg_3 (du...
Dimensions
Dump1090 Sitename
Hex Ident
Msg Gen Ts
Msg Log Ts
Record Type
Record Type Number
Sg Aircraft Id
Sg Flight Id
Sg Session Id
Measure Names
Measures
Alert
Altitude
Emergency
Is On Ground
Latitude
Longitude
Spi
Number of Records
Measure Values
Pages
Filters
Altitude
Msg Gen Ts
Marks
Automatic
Color Size Label
Detail Tooltip
Altitude
ATTR(Msg Gen..
Columns
Longitude
Rows
Latitude
Altitude
Msg Gen Ts
26/03/2016 00:00:00
23/05/2017 02:00:00
Altitude
838
7102
© OpenStreetMap contributors

High vs Low Altitude



Tracking a Single Flight (STK43BR)



Shell, SQL, HTML and Google Maps API

```
printf "%s\n" "Usage:"
printf "%s\n" "  $h_prog_name"

printf "%s\n" "   --action              {start|stop}"
printf "%s\n" "   --logfile             {DUMP1090 log file}"
printf "%s\n" "   --schema              {schema}"
printf "%s\n" "   --time_slice_number   {time slice number}"
printf "%s\n" "   --time_slice_units    {time slice units}"
printf "%s\n" "   --verbose             {Y|N}"
printf "%s\n" "   --sleep               {number of seconds to sleep between re-runs}"
```

```
 select
  date_trunc('minute',max(msg_gen_ts))            as end_time,
  date_trunc('minute',max(msg_gen_ts) - interval '$h_clv_time_slice_number $h_clv_time_slice_units')  as start_time
from
  ${h_clv_schema}.dump1090_msg_5
```

Shell, SQL, HTML and Google Maps API

```
create local temporary table if not exists dump1090_msg_1 on commit preserve rows as
select distinct
   msg_1.hex_ident            as hex_ident,
   max(msg_1.msg_gen_ts)      as msg_gen_ts,
   max(msg_1.call_sign)       as call_sign
from
   ${h_clv_schema}.dump1090_msg_1 msg_1
where
   msg_1.msg_gen_ts between '$h_clv_start_time' and '$h_clv_end_time'
group by
   hex_ident
;

create local temporary table if not exists dump1090_msg_3 on commit preserve rows as
select distinct
   msg_3.hex_ident            as hex_ident,
   max(msg_3.msg_gen_ts)      as msg_gen_ts,
   max(msg_3.altitude)        as altitude,
   max(msg_3.latitude)        as latitude,
   max(msg_3.longitude)       as longitude
from
   ${h_clv_schema}.dump1090_msg_3 msg_3
where
   msg_3.msg_gen_ts between '$h_clv_start_time' and '$h_clv_end_time'
group by
   hex_ident
;
```

Shell, SQL, HTML and Google Maps API

```
echo " <!DOCTYPE html>
<html>
  <head>
    <style type=\"text/css\">
      html, body { height: 100%; margin: 0; padding: 0; }
      #map { height: 100%; }
    </style>
    <meta http-equiv="refresh" content="10" >
  </head>
  <body>
    <div id=\"map\"></div>
    <script type=\"text/javascript\">
var map;
function initMap() {

  var myHomeLatLng = {lat: 59.331, lng: 18.031};

  map = new google.maps.Map(document.getElementById('map'), {
    center: myHomeLatLng,
    zoom: 9,
    panControl: true,
    zoomControl: true,
        zoomControlOptions: {
        style: google.maps.ZoomControlStyle.LARGE,
        position: google.maps.ControlPosition.RIGHT_CENTER
    },
```

```
var planeImage_000_045 = 'black_plane_000_045.gif';
var planeImage_045_090 = 'black_plane_045_090.gif';
var planeImage_090_135 = 'black_plane_090_135.gif';
var planeImage_135_180 = 'black_plane_135_180.gif';
var planeImage_180_225 = 'black_plane_180_225.gif';
var planeImage_225_270 = 'black_plane_225_270.gif';
var planeImage_270_315 = 'black_plane_270_315.gif';
var planeImage_315_360 = 'black_plane_315_360.gif';
```

Shell, SQL, HTML and Google Maps API

```
while read h_hex_ident h_date_msg_gen h_time_msg_gen h_altitude h_latitude h_longitude h_track h_call_sign
do

   (( h_noof_aircraft += 1 ))


   if   ((0<=h_track && h_track<=44))
   then
      h_aircraft="planeImage_000_045"
   elif ((45<=h_track && h_track<=89))
   then
      h_aircraft="planeImage_045_090"
   elif ((90<=h_track && h_track<=134))
   then
      h_aircraft="planeImage_090_135"
   elif ((135<=h_track && h_track<=179))
   then
      h_aircraft="planeImage_135_180"
   elif ((180<=h_track && h_track<=224))
```

```
echo "var plane$h_noof_aircraft = new google.maps.Marker({
  position: {lat: $h_latitude, lng: $h_longitude},
  map: map,
  title: '$h_call_sign - Alt: $h_altitude Lat: $h_latitude Long: $h_longitude Dt: $h_date_msg_gen Tm: $h_time_msg_gen HexId: $h_hex_ident',
  icon: $h_aircraft
  });" >> $h_clf_dump1090_flights_html_tmp
```

```
<script async defer
  src=\"https://maps.googleapis.com/maps/api/js?key=[redacted]&region=GB&callback=initMap\">
</script>
```

Shell, SQL, HTML and Google Maps API

Agenda

- Background to "The Lab Series" and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - **Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy**
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- What's next?

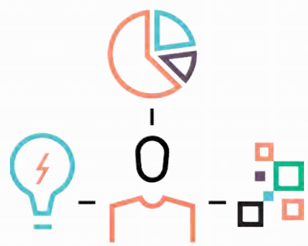
Vertica Analytical Capabilities

SQL ‘99

- Aggregate
- Analytical
- Window functions
- Date/Time functions
- String functions
- Mathematical functions

Allows for:

- Standard functionality that performs at scale



SQL Extensions

- Pattern matching
- Event series joins
- Time series
- Event-based windows

Allows for:

- Sessionization
- Conversion analysis
- Fraud detection
- Fast Aggregates (LAP)

SDKs

Analytics

- C++
- Java
- R
- Python

Connection

- ODBC/JDBC
- HIVE
- Hadoop
- Flex zone

Allows for:

- Specialized parsers
- Custom data mining
- Semi-structured data processing

In-database Analytics

- Regression
- K-means
- Statistical modeling
- Classification algorithms
- Text mining
- Geospatial

Allows for:

- Statistical modeling
- Cluster analysis
- Predictive analytics
- Geospatial analysis

Rich Set of Tools to Get Data Ready for Modeling

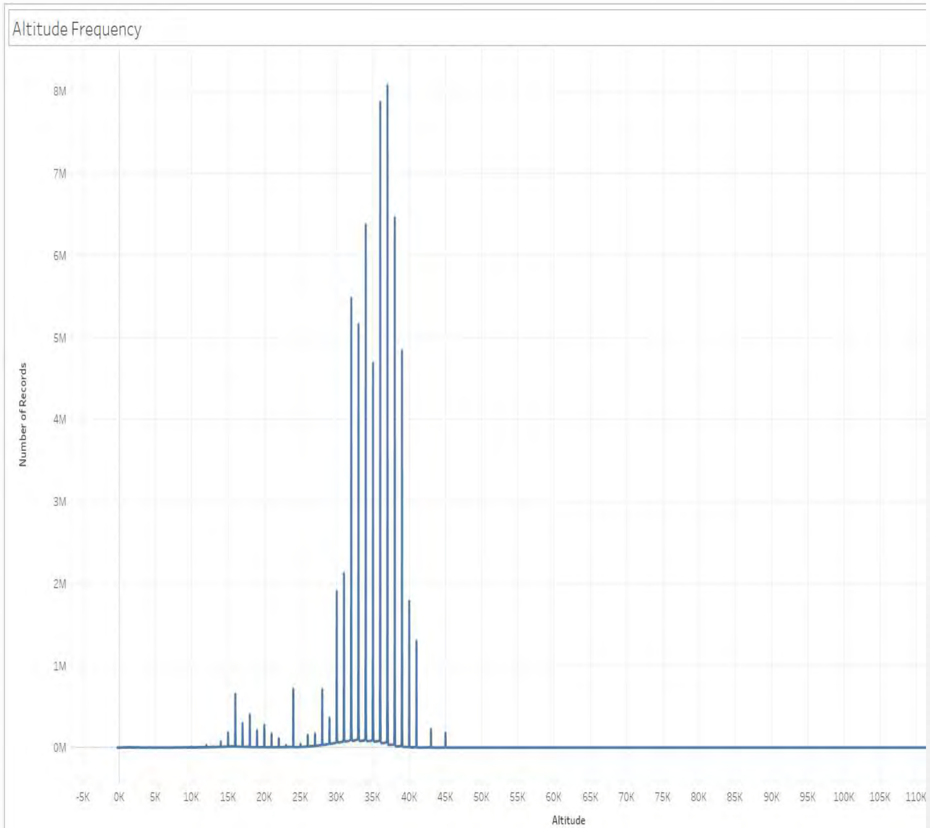
Capture & Enrich	Measure & Prepare	Model & Deploy
Copy Flex Tables External Tables Parsers: Avro, CEF, CSV, Delim, JSON, RegEX Streaming Utilities including Kafka Integration S3 & ABS ORC, Parquet, HIVE, Spark RDD & DF Shapefiles & Spatial Data	1000s of functions **Time Series Prep (GFI, Interpolation, Slicing, TSA)** **Sessionize** Pattern Matching Event Series Joins Advanced Aggregation Date & Time Algebra Window & Partition Stats & Math Data Type Handling Strings Sequences Geospatial, Joins, Conversions Balance Sampling **Outlier Detection** Normalize Missing Value Imputation	Linear Regression Logistic Regression K-Means Naïve Bayes SVM Model Evaluation & Visualization Model Management UDX Functions Text Analytics

ANSI SQL Standards – Algorithms Developed for MPP Execution – Relational Structure at PB Scale

Agenda

- Background to “The Lab Series” and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - **Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization**
- What’s next?

Example use case – Outlier Detection



Example use case – Outlier Detection

```sql
/*
 * Table counts
 */
SELECT COUNT(*)
FROM dump1090_batch.dump1090_msg_3;
```

COUNT
148,342,592

Example use case – Outlier Detection

```
/*
 * Data Exploration
 */
SELECT DISTINCT hex_ident, msg_gen_ts, altitude, latitude, longitude
FROM dump1090_batch.dump1090_msg_3
WHERE msg_gen_ts BETWEEN '27-jun-2017' AND '28-jun-2017'
ORDER BY 1, 2
LIMIT 10;
```

	hex_ident	msg_gen_ts	altitude	latitude	longitude
1	00B203	2017-06-27 18:59:31	19,000	50.66429	-0.3768
2	00B203	2017-06-27 18:59:32	19,000	50.66373	-0.37606
3	00B203	2017-06-27 18:59:38	19,150	50.65498	-0.36464
4	00B203	2017-06-27 18:59:40	19,200	50.65196	-0.36074
5	00B203	2017-06-27 18:59:43	19,275	50.64786	-0.35535
6	00B203	2017-06-27 18:59:44	19,300	50.64646	-0.35357
7	00B203	2017-06-27 18:59:45	19,350	50.64455	-0.35106
8	00B203	2017-06-27 18:59:51	19,500	50.63736	-0.3418
9	00B203	2017-06-27 18:59:52	19,525	50.63599	-0.33998
10	00B203	2017-06-27 19:00:00	19,725	50.62468	-0.32533

Example use case – Outlier Detection

```sql
/*
 * Distribution of altitude of aircraft (identified by its registration number HEX_IDENT)
 */
SELECT
   a.NOOF_FLIGHTS,
   a.MIN_ALT,
   b.P05,
   b.P10,
   b.P25,
   b.MEDIAN,
   b.P75,
   b.P90,
   b.P95,
   a.MAX_ALT,
   a.AVG_ALT
FROM
(
SELECT
   COUNT(*)          AS NOOF_FLIGHTS,
   MIN(altitude)     AS MIN_ALT,
   MAX(altitude)     AS MAX_ALT,
   AVG(altitude)     AS AVG_ALT
FROM
   (
   SELECT DISTINCT
      hex_ident,
      altitude
   FROM
      dump1090_batch.dump1090_msg_3
   ) d
) a
```

```sql
CROSS JOIN
(
SELECT DISTINCT
   PERCENTILE_CONT(0.5)  WITHIN GROUP (ORDER BY altitude) OVER() as P05,
   PERCENTILE_CONT(0.1)  WITHIN GROUP (ORDER BY altitude) OVER() as P10,
   PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY altitude) OVER() as P25,
   MEDIAN(altitude)                                       OVER() as MEDIAN,
   PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY altitude) OVER() as P75,
   PERCENTILE_CONT(0.9)  WITHIN GROUP (ORDER BY altitude) OVER() as P90,
   PERCENTILE_CONT(0.95) WITHIN GROUP (ORDER BY altitude) OVER() as P95
  FROM
(SELECT distinct hex_ident, altitude
FROM
   dump1090_batch.dump1090_msg_3
order by hex_ident, altitude ASC) c
) b ;
```

Example use case – Outlier Detection

NOOF_FLIGHTS	MIN_ALT	P05	P10	P25	MEDIAN
3,599,427	-375	28,675	15,225	21,250	28,675

P75	P90	P95	MAX_ALT	AVG_ALT
34,375	37,675	39,125	124,400	27,396.1801003326

Example use case – Outlier Detection

```
/*
 * Feature Creation - Detect Outliers
 * (using a z-score of 5.0 or higher)
 */

DROP TABLE IF EXISTS dump1090_batch.dump1090_msg_3_outliers;

SELECT DETECT_OUTLIERS('dump1090_batch.dump1090_msg_3_outliers',
                       'dump1090_batch.dump1090_msg_3',
                       'altitude',
                       'robust_zscore'
       USING PARAMETERS outlier_threshold=5.0, key_columns='hex_ident');
```

DETECT_OUTLIERS
Detected 1741503 outliers¶

Example use case – Outlier Detection

```
-- View the results

SELECT altitude, COUNT(*)
FROM dump1090_batch.dump1090_msg_3_outlier
GROUP BY 1 ORDER BY 1 ASC LIMIT 10;
```

altitude	COUNT
-375	27
-350	60
-325	109
-300	91
-275	125
-250	271
-225	115
-200	121
-175	141
-150	261

Example use case – Outlier Detection

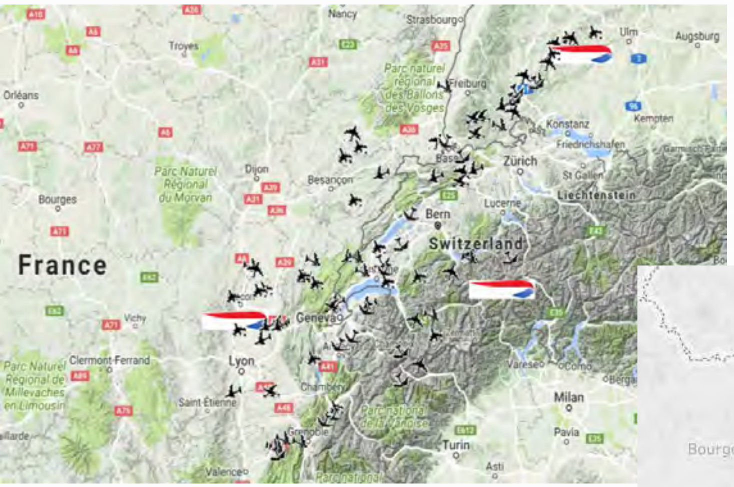
```
SELECT altitude, COUNT(*)
FROM dump1090_batch.dump1090_msg_3_outliers
GROUP BY 1 ORDER BY 1 DESC LIMIT 10;
```

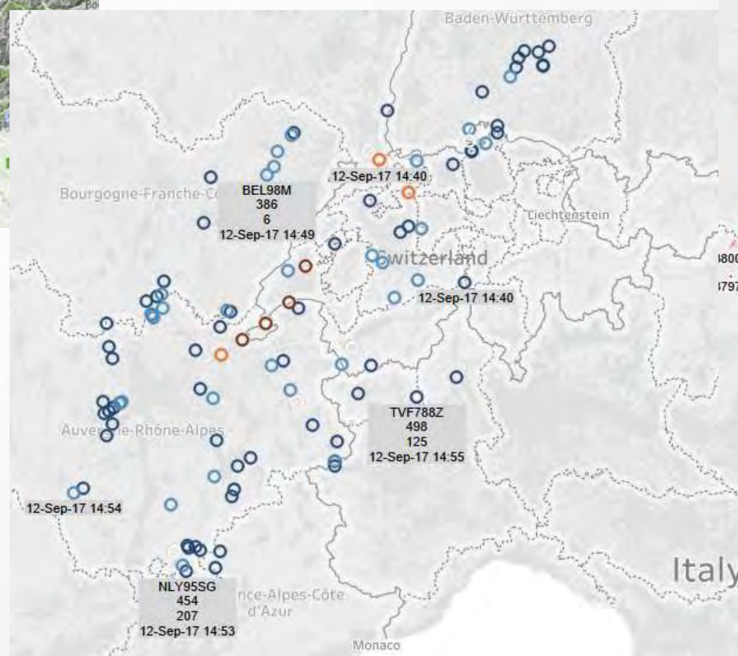
altitude	COUNT
124,400	3
124,200	2
123,100	4
122,200	2
120,400	4
120,100	1
119,100	7
118,400	4
117,100	5
116,400	8

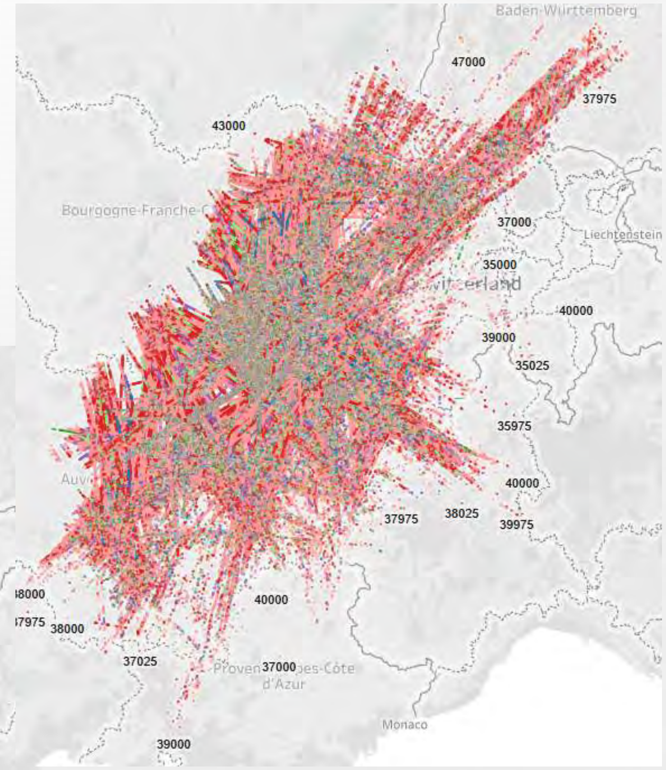
From Swansea to Switzerland

From Swansea to Switzerland

Installed in July 2017







58 Days: 1bn Messages, 12k aircraft, 200,000 KM^2 (*)

MSG Type	First TS	Latest TS	Distinct Hex Ident	No of Messages
AIR	[NULL]	[NULL]	0	0
ID	[NULL]	[NULL]	0	0
MSG_1	2017-07-17 22:01:12	2017-09-12 11:06:39	8,163	6,234,095
MSG_2	[NULL]	[NULL]	0	0
MSG_3	2017-07-17 22:01:47	2017-09-12 11:06:40	7,705	61,756,974
MSG_4	2017-07-17 22:01:47	2017-09-12 11:06:39	7,769	62,167,725
MSG_5	2017-07-17 22:01:47	2017-09-12 11:06:40	11,665	208,449,533
MSG_6	2017-07-17 22:01:47	2017-09-12 11:06:40	11,512	69,780,084
MSG_7	2017-07-17 22:01:48	2017-09-12 11:06:40	11,694	213,866,386
MSG_8	2017-07-17 22:01:48	2017-09-12 11:06:40	11,799	398,037,717
STA	[NULL]	[NULL]	0	0

Let’s start with just one aircraft

- Call Sign - G-EUPA
- Mode-S - 400801
- Aircraft type - Airbus A319-131
- Build date - September 1999 (18 years old)
- Serial Number - 1082
- Airline - BA / BAW
- Livery - Olympic Dove

How many Type-3 messages?

```sql
/*
 * Count the number of MSG_3 messages for this one aircraft
 */
SELECT
        count(*)                AS noof_messages
FROM
        dump1090_kafka.dump1090_msg_3 msg_3
WHERE
        msg_3.hex_ident         = '400801'
```

noof_messages
13,719

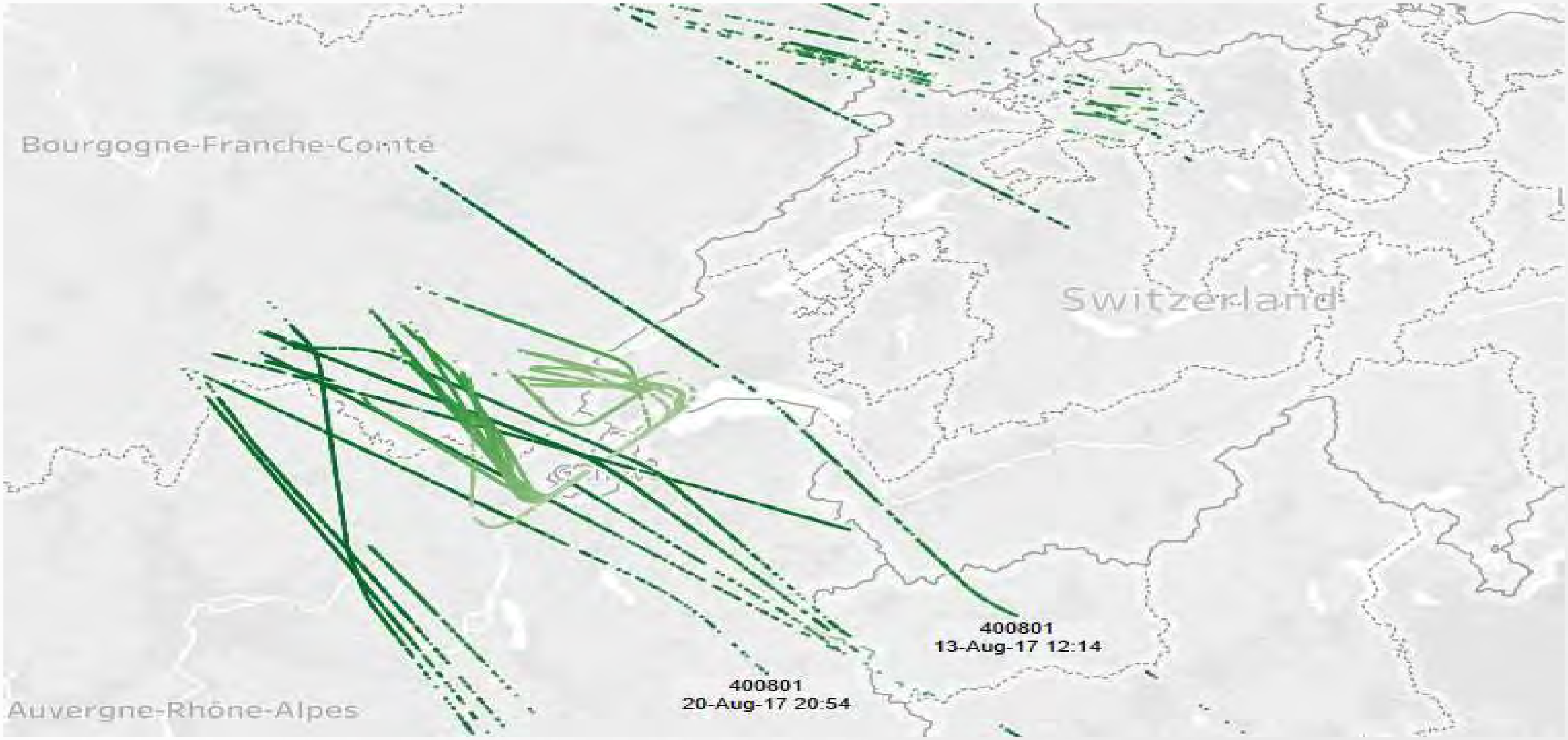
Take a closer look at these Type-3 messages

```
/*
 * Take a look at these messages in more detail
 */
SELECT
--          msg_3.hex_ident          AS msg_3_hex_ident,
            msg_3.msg_gen_ts         AS msg_3_msg_gen_ts,
            msg_3.altitude           AS msg_3_altitude,
            msg_3.latitude           AS msg_3_latitude,
            msg_3.longitude          AS msg_3_longitude
FROM
            dump1090_kafka.dump1090_msg_3 msg_3
WHERE
            msg_3.hex_ident          = '400801'
ORDER BY
            msg_3.msg_gen_ts
```

Take a closer look at these Type-3 messages

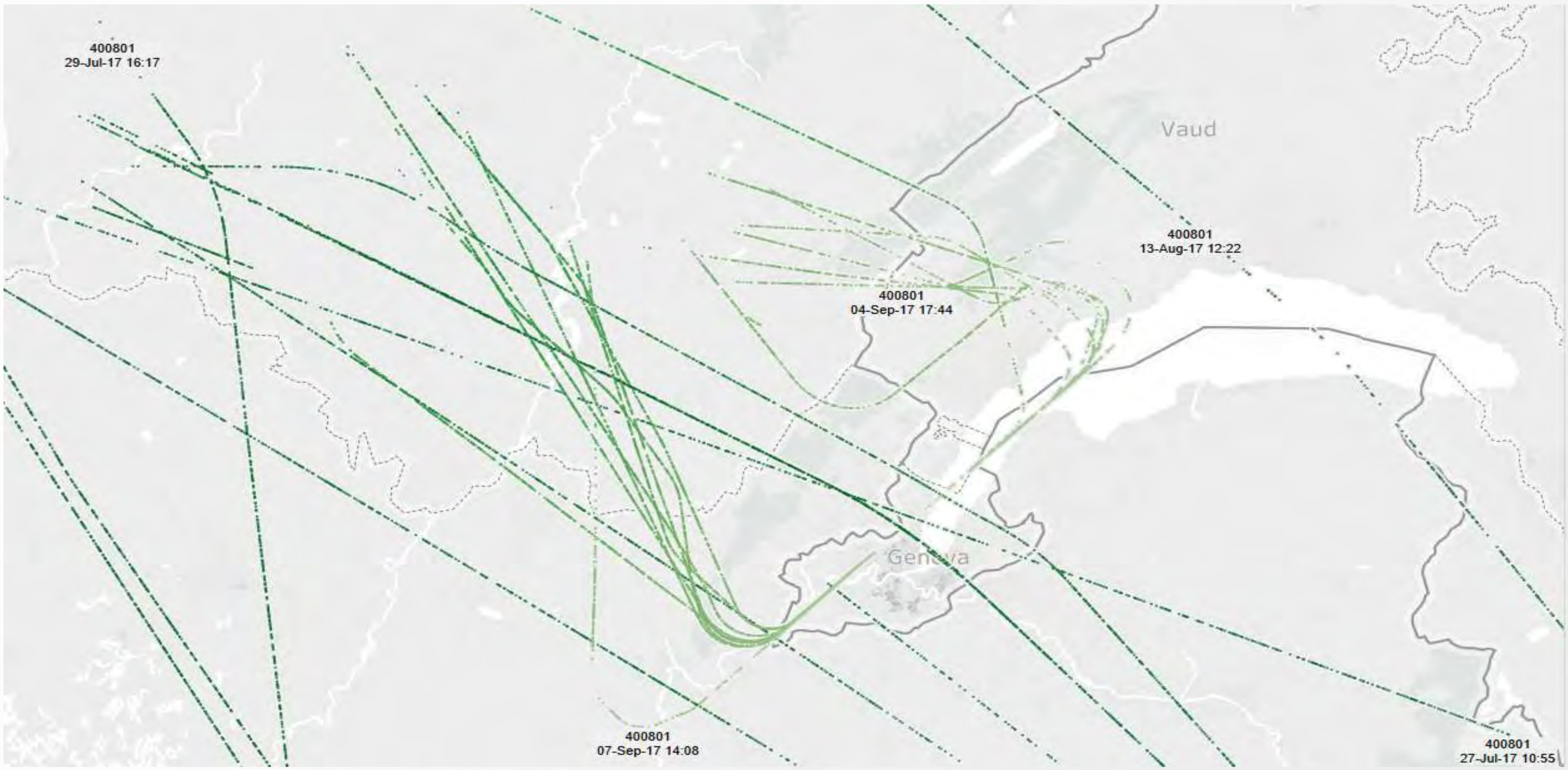
	msg_3_msg_gen_ts	msg_3_altitude	msg_3_latitude	msg_3_longitude
1	2017-07-21 13:38:00	16,100	47.44213	7.97107
2	2017-07-21 13:38:04	16,275	47.44249	7.96083
3	2017-07-21 13:38:13	16,675	47.44318	7.94133
4	2017-07-21 13:38:14	16,750	47.44331	7.93742
5	2017-07-21 13:38:16	16,850	47.44345	7.93247
6	2017-07-21 13:38:17	16,900	47.44353	7.93009
64	2017-07-21 13:47:30	34,250	47.65279	6.43016
65	2017-07-21 13:47:35	34,325	47.65557	6.41742
66	2017-07-21 13:47:38	34,400	47.65764	6.40798
67	2017-07-21 13:48:33	35,375	47.691	6.25615
68	2017-07-24 22:23:18	13,975	46.54559	5.83897
69	2017-07-24 22:23:26	13,800	46.53415	5.85055
70	2017-07-24 22:23:28	13,750	46.53186	5.8529

Visualising the flight tracks of aircraft "400801"



MICRO FOCUS

Visualising the flight tracks of aircraft “400801”



Vertica's time series Gap Filling & Interpolation

```
SELECT
   gfi_msg_3_msg_gen_ts  AS msg_3_msg_gen_ts,
   gfi_altitude_int      AS altitude,
   gfi_longitude         AS longitude,
   gfi_latitude          AS latitude
FROM
   (
   SELECT
      gfi_msg_3_msg_gen_ts                                AS gfi_msg_3_msg_gen_ts,
      TS_FIRST_VALUE(msg_3.altitude,  'LINEAR')::INT AS gfi_altitude_int,
      TS_FIRST_VALUE(msg_3.latitude,  'LINEAR')      AS gfi_latitude,
      TS_FIRST_VALUE(msg_3.longitude, 'LINEAR')      AS gfi_longitude
   FROM
      dump1090_kafka.dump1090_msg_3 msg_3
   WHERE
      msg_3.hex_ident = '400801'
   TIMESERIES
      gfi_msg_3_msg_gen_ts AS '1 seconds'
   OVER
      (
      PARTITION BY
         msg_3.hex_ident
      ORDER BY
         msg_3.msg_gen_ts::TIMESTAMP(0)
      )
   ) a
```

The result of applying time series GFI

	msg_3_msg_gen_ts	altitude	longitude	latitude
1	2017-07-21 13:38:01	16,100	7.97107	47.44213
2	2017-07-21 13:38:02	16,144	7.96851	47.44222
3	2017-07-21 13:38:03	16,188	7.96595	47.44231
4	2017-07-21 13:38:04	16,231	7.96339	47.4424
5	2017-07-21 13:38:05	16,275	7.96083	47.44249
6	2017-07-21 13:38:06	16,325		
7	2017-07-21 13:38:07	16,375		
8	2017-07-21 13:38:08	16,425		
631	2017-07-21 13:48:31	35,339		
632	2017-07-21 13:48:32	35,357		
633	2017-07-21 13:48:33	35,375		
634	2017-07-21 13:48:34	35,375	[illegible]	[illegible]
635	2017-07-21 13:48:35	35,375	6.2561471237	47.690992103
636	2017-07-21 13:48:36	35,375	6.2561456856	47.6909881544
637	2017-07-21 13:48:37	35,375	6.2561442475	47.6909842059

65	2017-07-21 13:47:35	34,325
66	2017-07-21 13:47:38	34,400
67	2017-07-21 13:48:33	35,375
68	2017-07-24 22:23:18	13,975
69	2017-07-24 22:23:26	13,800

Introducing Sessions

```
SELECT
   msg_3.hex_ident        AS msg_3_hex_ident,
   msg_3.msg_gen_ts       AS msg_3_msg_gen_ts,
   msg_3.altitude         AS msg_3_altitude,
   msg_3.latitude         AS msg_3_latitude,
   msg_3.longitude        AS msg_3_longitude,
   CONDITIONAL_TRUE_EVENT
      (
      msg_3.msg_gen_ts - LAG(msg_3.msg_gen_ts) > '1 hour'
      )
      OVER
           (
           PARTITION BY
              msg_3.hex_ident
           ORDER BY
              msg_3.msg_gen_ts
           ) AS msg_3_flight
FROM
   dump1090_kafka.dump1090_msg_3 msg_3
WHERE
   msg_3.hex_ident = '400801'
) msg_3
```

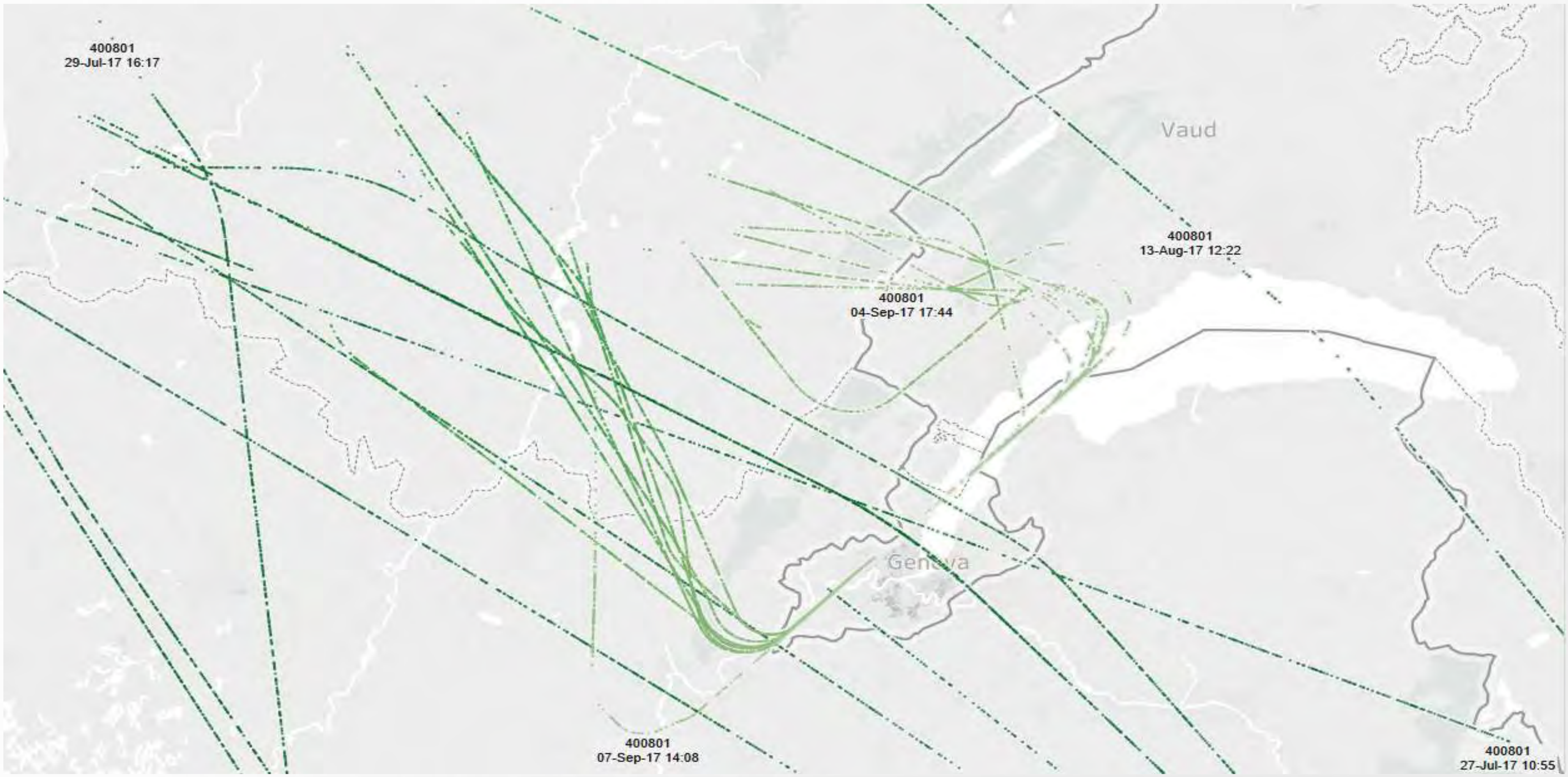
Introducing Sessions

```
SELECT
   gfi_msg_3_msg_gen_ts   AS msg_3_msg_gen_ts,
   gfi_altitude_int       AS altitude,
   gfi_longitude          AS longitude,
   gfi_latitude           AS latitude,
   msg_3_flight           AS flight
FROM
   (
   SELECT
      msg_3_hex_ident                                        AS msg_3_hex_ident,
      gfi_msg_3_msg_gen_ts                                   AS gfi_msg_3_msg_gen_ts,
      TS_FIRST_VALUE(msg_3_altitude,  'LINEAR')::INT         AS gfi_altitude_int,
      TS_FIRST_VALUE(msg_3_latitude,  'LINEAR')              AS gfi_latitude,
      TS_FIRST_VALUE(msg_3_longitude, 'LINEAR')              AS gfi_longitude,
      msg_3_flight                                           AS msg_3_flight
   FROM
      (
      SELECT
         msg_3.hex_ident       AS msg_3_hex_ident,
         msg_3.msg_gen_ts      AS msg_3_msg_gen_ts,
         msg_3.altitude        AS msg_3_altitude,
         msg_3.latitude        AS msg_3_latitude,
         msg_3.longitude       AS msg_3_longitude,
         CONDITIONAL_TRUE_EVENT
            (
            msg_3.msg_gen_ts - LAG(msg_3.msg_gen_ts) > '1 hour'
            )
            OVER
               (
               PARTITION BY
                  msg_3.hex_ident
               ORDER BY
                  msg_3.msg_gen_ts
               ) AS msg_3_flight
      FROM
         dump1090_kafka.dump1090_msg_3 msg_3
      WHERE
         msg_3.hex_ident = '400801'
      ) msg_3
   TIMESERIES
      gfi_msg_3_msg_gen_ts AS '1 seconds'
   OVER
      (
      PARTITION BY
         msg_3_hex_ident,
         msg_3_flight
      ORDER BY
         msg_3_msg_gen_ts::TIMESTAMP(0)
      )
   ) a
ORDER BY
   gfi_msg_3_msg_gen_ts
```

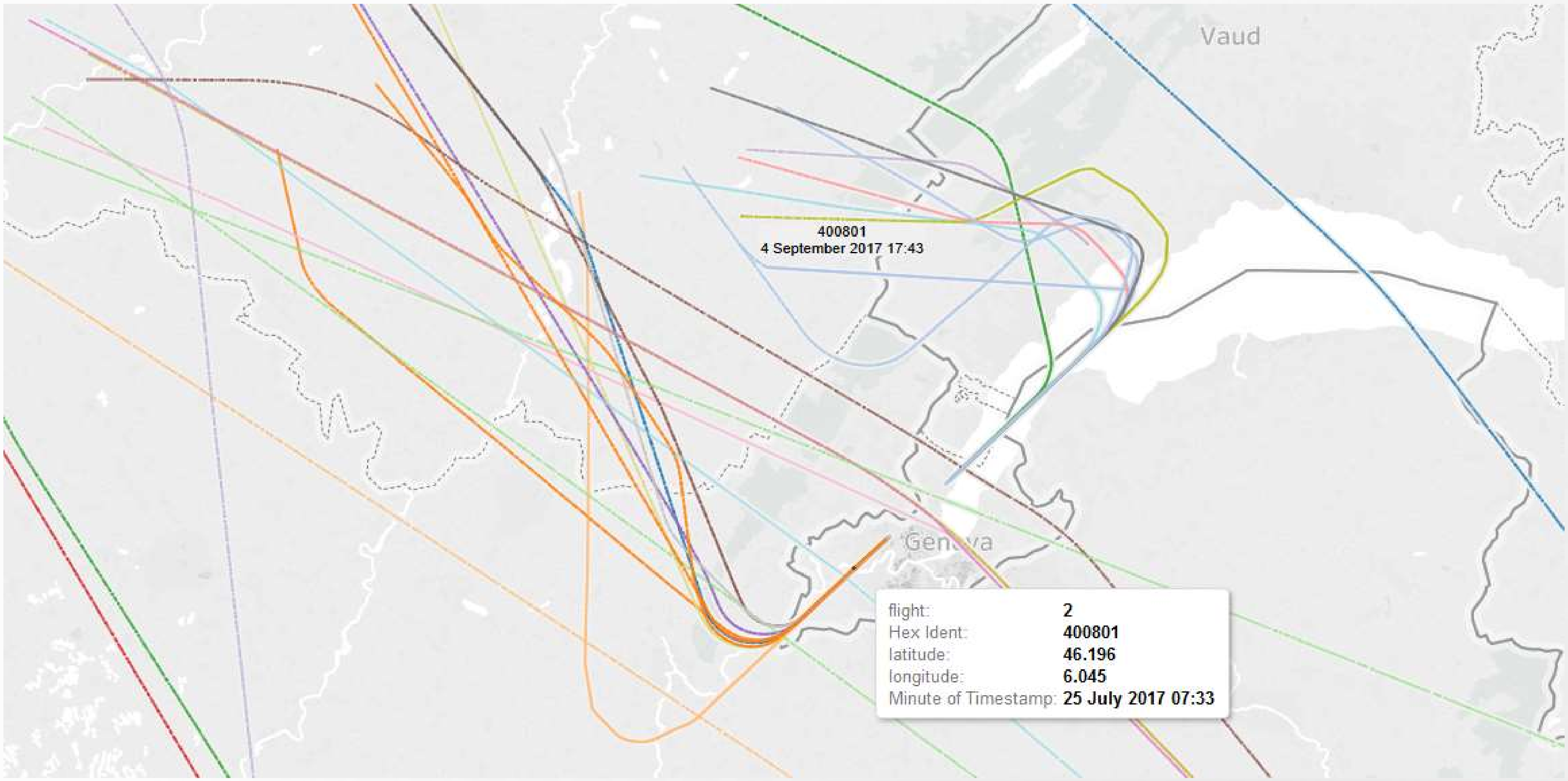
Result of sessionising our GFI data

	msg_3_msg_gen_ts	altitude	longitude	latitude	flight
1	2017-07-21 13:38:01	16,100	7.97107	47.44213	0
2	2017-07-21 13:38:02	16,144	7.96851	47.44222	0
3	2017-07-21 13:38:03	16,188	7.96595	47.44231	0
4	2017-07-21 13:38:04	16,231	7.96339	47.4424	0
5	2017-07-21 13:38:05	16,275	7.96083	47.44249	0
6	2017-07-21 13:38:06	16,325	7.9583925	47.44257625	0
631	2017-07-21 13:48:31	35,339	6.2617733333	47.6897644444	0
632	2017-07-21 13:48:32	35,357	6.2589616667	47.6903822222	0
633	2017-07-21 13:48:33	35,375	6.25615	47.691	0
634	2017-07-24 22:23:19	13,975	5.83897	46.54559	1
635	2017-07-24 22:23:20	13,953	5.8404175	46.54416	1
636	2017-07-24 22:23:21	13,931	5.841865	46.54273	1
27318	2017-09-07 15:24:47	26,225	5.55071	46.81439	44
27319	2017-09-07 15:24:48	26,250	5.54877	46.81508	44

What our flights tracks looked like…



… and what they look like with GFI and sessions



Agenda

- Background to "The Lab Series" and the Big Data & Machine Learning Meetups
- Covered so far on Project #1:
  - Introduction to Automatic Dependent Surveillance – Broadcast (ADS-B)
  - Using a Raspberry Pi to capture and decode ADS-B signals
  - DUMP1090 – Live tracking and streaming
  - Apache Kafka and Extract Transform & Load (ETL)
  - Introduction to Vertica
  - Kafka / Vertica integration and Management Console
  - Vertica integration tools and simple visualisations
  - Vertica data modelling tools – Capture & Enrich / Measure & Prepare / Model & Deploy
  - Measure and Prepare: Outlier detection, gap filling & interpolation and sessionization
- **What's next?**

What's Next?

The next Big Data & Machine Learning Meetups

- Big Data and Machine Learning (London) – Meetup #8

  Date, time and venue TBC

- Big Data and Machine Learning (Munich) – Meetup #2

  Thursday 23rd November 2017 @ CGI Munich

- Big Data and Machine Learning (Cambridge) – Meetup #2 & #3

  Wednesday 10th January 2018 @ Jagex, Cambridge

  Wednesday 4th April 2018 @ Jagex, Cambridge

Thank you

Mark.Whalley@microfocus.com

www.vertica.com

www.myvertica.com