

“Weather Forecast Evaluator”

W205 Final Project, August 10, 2016



XT 'Emitter' Nguyen



Christopher 'Bolt' McPherson



Rama
'Programming in the Cloud'
Thamman



Chuck 'Spout' Bolin

Background

Ohio native Chris McPherson said “**Enough is enough! We need better forecasting.**”



The goal of this project is simple.

- Assign each city a score corresponding to weather forecasting accuracy.
- Place this information into the hand of the consumer.
- Investigate these questions.
 - Where (cities worldwide)
 - When (seasons)
 - Which measures(rainfall, temperature)



What is the *Weather Forecast Evaluator (WFE)*?

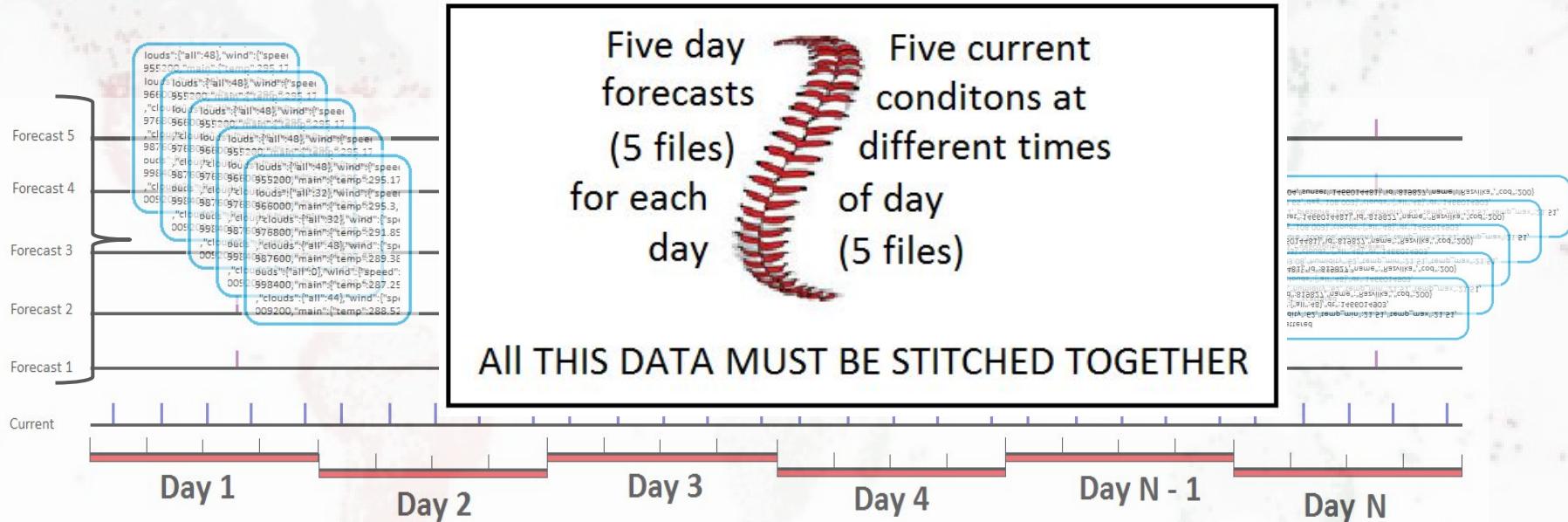
The **WFE** is a **weather data collection system** that stores and transforms data for the **purpose of calculating up to date forecasting accuracy scores** for over 70,000 cities worldwide.

- Allow for **ranking of forecasting accuracy** by city, region, and country.
- Individualized **scores** are provided **for each weather metric**
- Evaluate various models.
 - Identify **those that need better modeling**.
 - Recognize **superior predictive models**.
- Scores benefit industries and local governments worldwide in areas such as farming, construction, outdoor entertainment, travel industry, city maintenance, and more.



The *Weather Forecast Evaluator* Challenge

- The **WFE** has collected forecast and current weather data for **46 days**.



70,000+ Cities!!!

Architecture Overview

Architectural Evolution

Initial API Exploration (Week 1)

```
get_s3_bucket_files_from_list_v2.py
1 # 
2 # Purpose: Reads (files) from list from S3
3 #
4 # imports
5 import boto # pip.exe install boto
6 import urllib.request
7 from boto.s3.connection import S3Connection
8 from boto.s3.connectio
9 from boto.s3.key import Key
10 #
11 # security keys
12 access_key = 'AKMMMQHZ
13 secret_key = 'hKlmzEfe
14 #
15 # point to S3 connection
16 conn = S3Connection(ac
17 #
18 #read list file
19 myfile = open("cityfil
20 #
21 # create bucket object
22 bucketname = "weather
```

s3.list.txt - Notepad

File Edit Format View Help

06-24-2016/0/1005029-forecast-weather-00-02-38.json
06-24-2016/0/1005544-forecast-weather-00-02-38.json
06-24-2016/0/1024552-forecast-weather-00-03-04.json
06-24-2016/0/1024701-forecast-weather-00-02-50.json
06-24-2016/0/1026014-forecast-weather-00-03-04.json
06-24-2016/0/1039854-forecast-weather-00-02-38.json
06-24-2016/0/1057655-forecast-weather-00-02-38.json

"Quick and Dirty" (Week 1- 12)

```
Java - weather/src/com/weather/AWSUtil.java
File Edit Source Refactor Navigator
Package Explorer
aerosolve
Automation
> Paris 11807 [svn+ssh://50.18.170.118:11807]
qlik-automation
weather
src
com.parser
com.weather
AWSUtil.java
DbUtil.java
KafkaClient
MailUtil.java
AWSUtil.java
public class AWSUtil {
    private final static Logger LOG = LoggerFactory.getLogger(AWSUtil.class);
    private String errorMessage = null;
    private boolean status = true;
```

```
Result 1 Messages
2 select * from current_weather where city_id=4883207 and date(dt) >= '2016-07-01' and date(dt) <= '2016-07-02' limit 100;
```

1	id	city_id	city_name	dt	station_name	coord_lon	coord_lat	t	icon
698	4883207	Alisp		2016-07-01 02:08:39	stations	-87.74	41.67	299.56	10d
731	4883207	Alisp		2016-07-01 05:10:21	cmc stations	-87.74	41.67	296.07	10d
818	4883207	Alisp		2016-07-01 08:15:23	stations	-87.74	41.67	305.22	10d
897	4883207	Alisp		2016-07-01 11:05:42	stations	-87.74	41.67	296.07	10d
844	4883207	Alisp		2016-07-01 23:20:18	cmc stations	-87.74	41.67	305.22	10d

Production Ready (Week 11, 12)

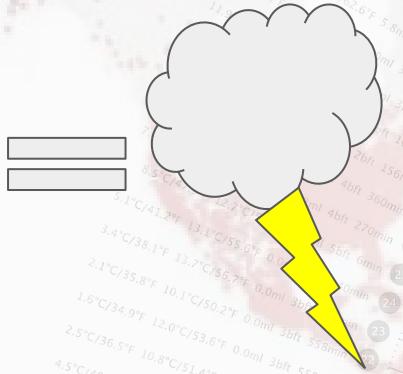
```
root@ip-172-31-29-198:~#
{
  "eve": 299.56,
  "morn": 299.56,
  "pressure": 296.07,
  "max": 305.22,
  "night": 296.07,
  "dt": 1466874000,
  "temp": {"day": 300, "m
  "buds": 88,
  "rain": 0.43},
  {"dt": 1466960800, "s
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}],
  "speed": 2.08,
  "deg": 104,
  "clouds": 11,
  "light rain", "icon": "10d"}]
```

We could not sit back and wait for the production system to be created.

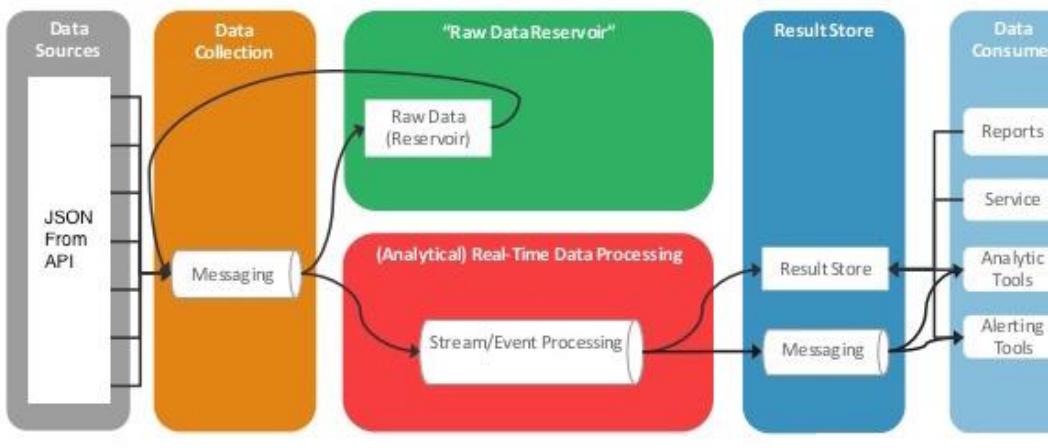
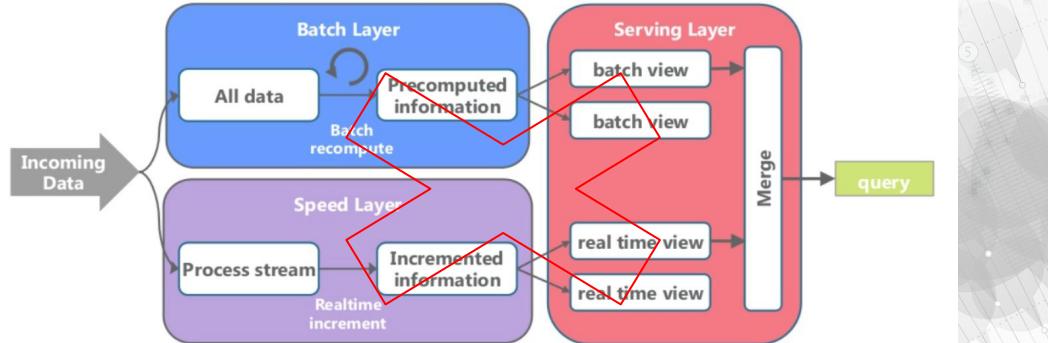
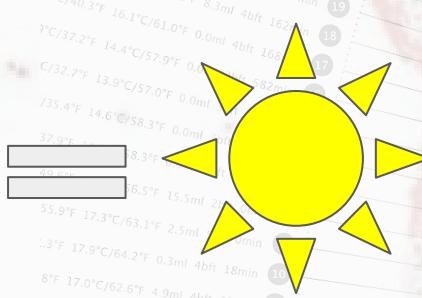
- Initial exploration of API using Python
- “Quick and Dirty” system to start archiving files.

Architecture

Lambda Architecture



Kappa Architecture



= Data in Motion

= Data at Rest

Tools

Storage

Streaming

Analytics



Application Metadata



Data Storage



Hive Metastore



Messaging System



Stream Processing



Query Engine

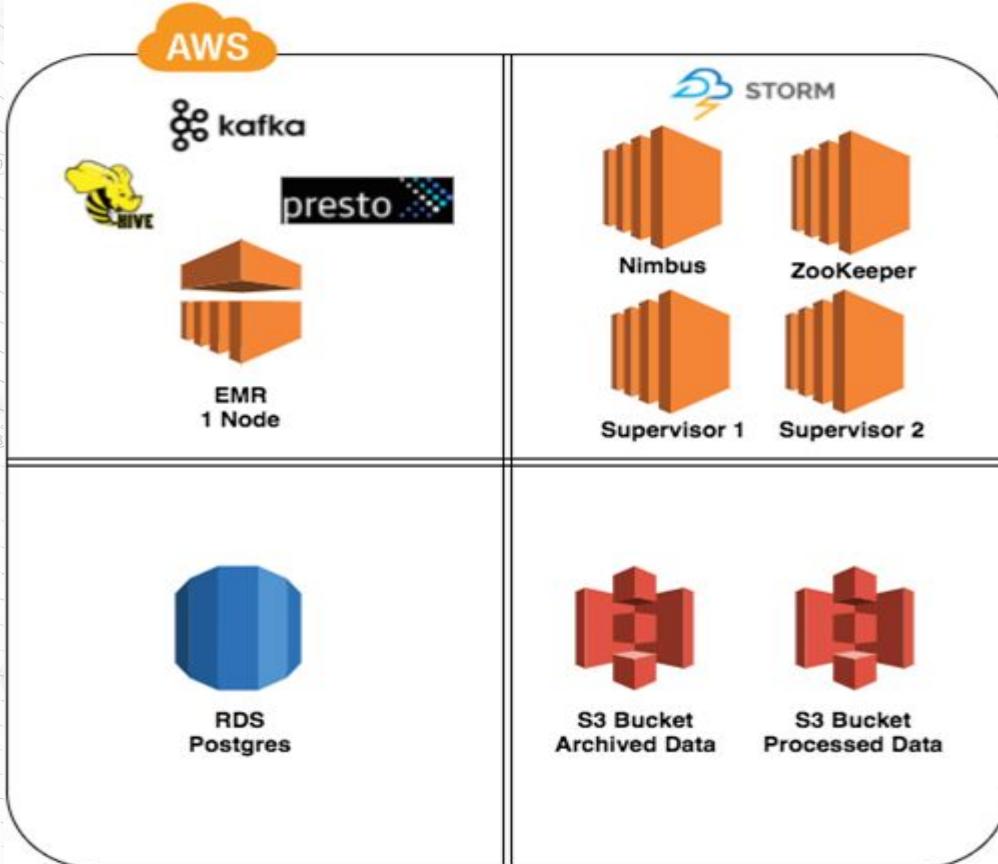


Visualizations

Infrastructure

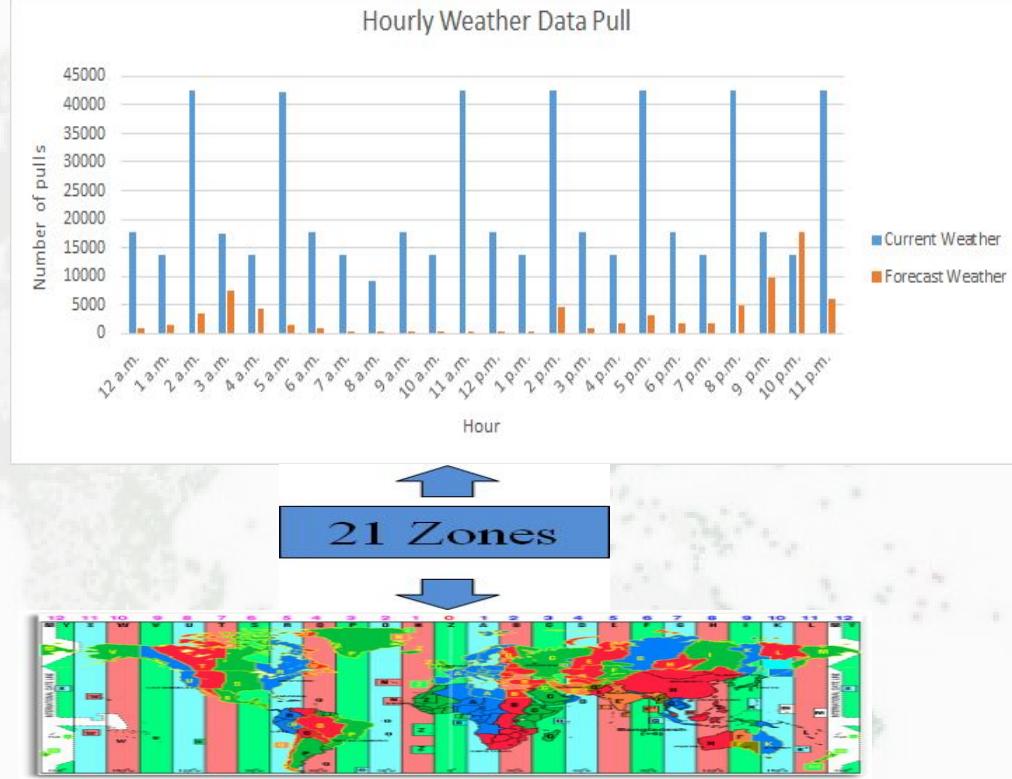
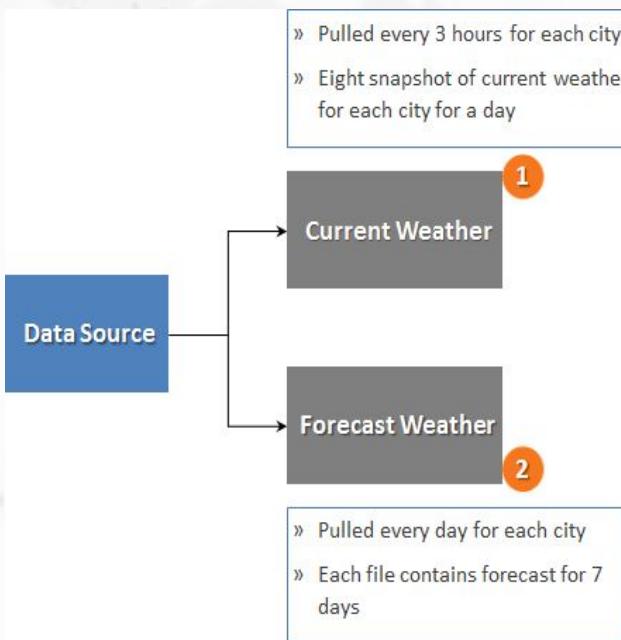


Scale Out



Implementation Details

Data Acquisition



Search ServerEmails (Ctrl+E)



Current Folder

All Unread

By Date ▾ Oldest ↓

◀ Older

w205.weather@gmail.com
W205 Project - Weather Processing Hour: 19
Kafka Upload Status: Success

6/27/2016

w205.weather@gmail.com
W205 Project - Weather Processing Hour: 20
Kafka Upload Status: Success



Kafka Upload Status: Success

S3 Upload Status: Success

Current weather processing status by Zone:

w205.weather@gmail.com
W205 Project - Weather Processing Hour: 21
Kafka Upload Status: Success

Zone: 21

of cities selected: 955

of Cities processed successfully: 955

of Error responses: 0

w205.weather@gmail.com
W205 Project - Weather Processing Hour: 22
Kafka Upload Status: Success

Zone: 20

of cities selected: 1484

of Cities processed successfully: 1484 # of Error responses: 0

Zone: 19

of cities selected: 4399

of Cities processed successfully: 4346 # of Error responses: 53

w205.weather@gmail.com
W205 Project - Weather Processing Hour: 18
Kafka Upload Status: Success



Kafka Upload Status: Success

S3 Upload Status: Success

Current weather processing status by Zone:

Zone: 21

of cities selected: 955

of Cities processed successfully: 955

of Error responses: 0

Zone: 20

of cities selected: 1484

of Cities processed successfully: 1484 # of Error responses: 0

Zone: 19

of cities selected: 4399

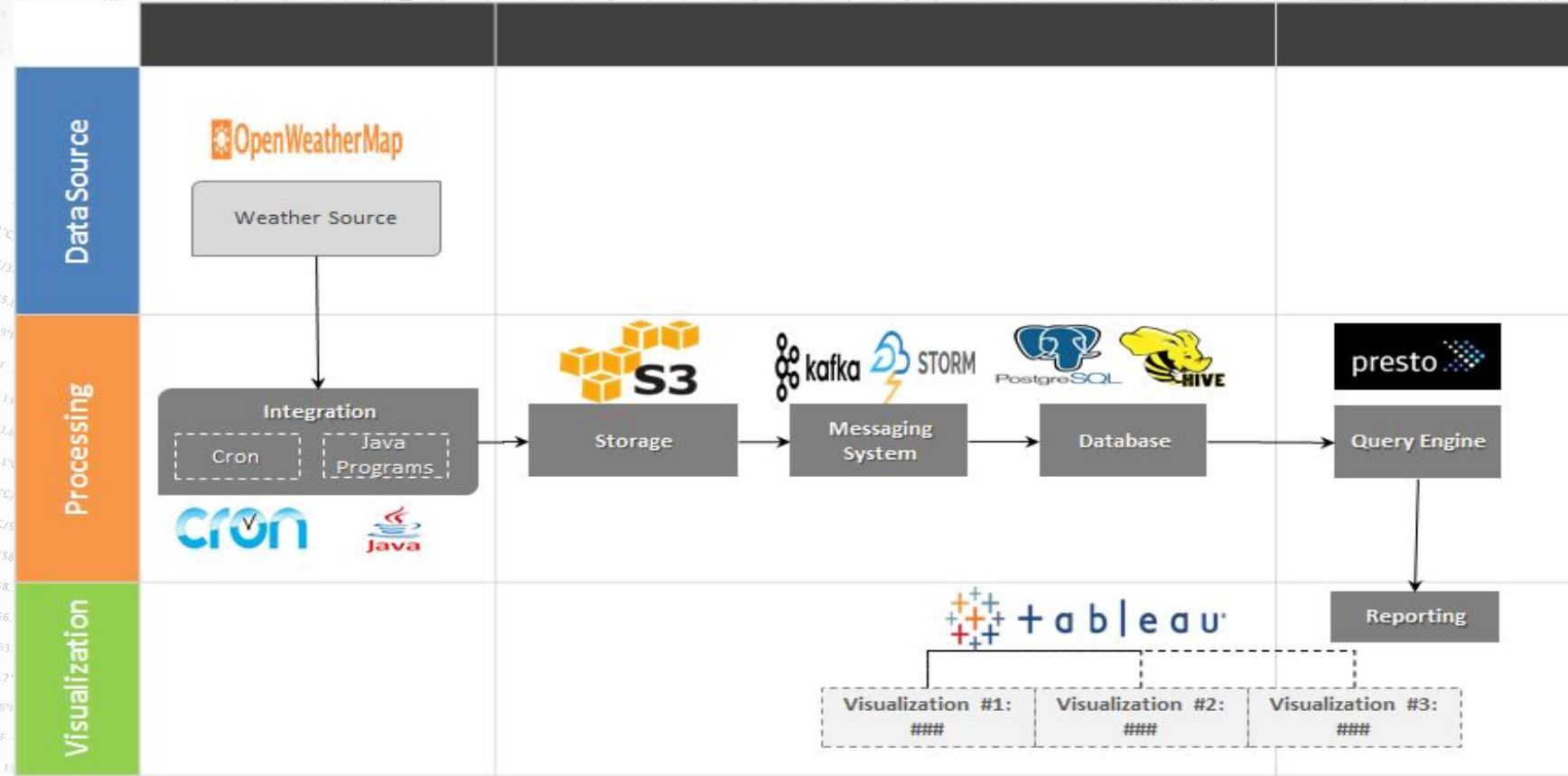
of Cities processed successfully: 4346 # of Error responses: 53

Zone: 18

of cities selected: 7321

453

Data Flow



Data Velocity and Size

Data
Velocity

Data
Volume

Data
Variety

Daily : 630,000 pulls

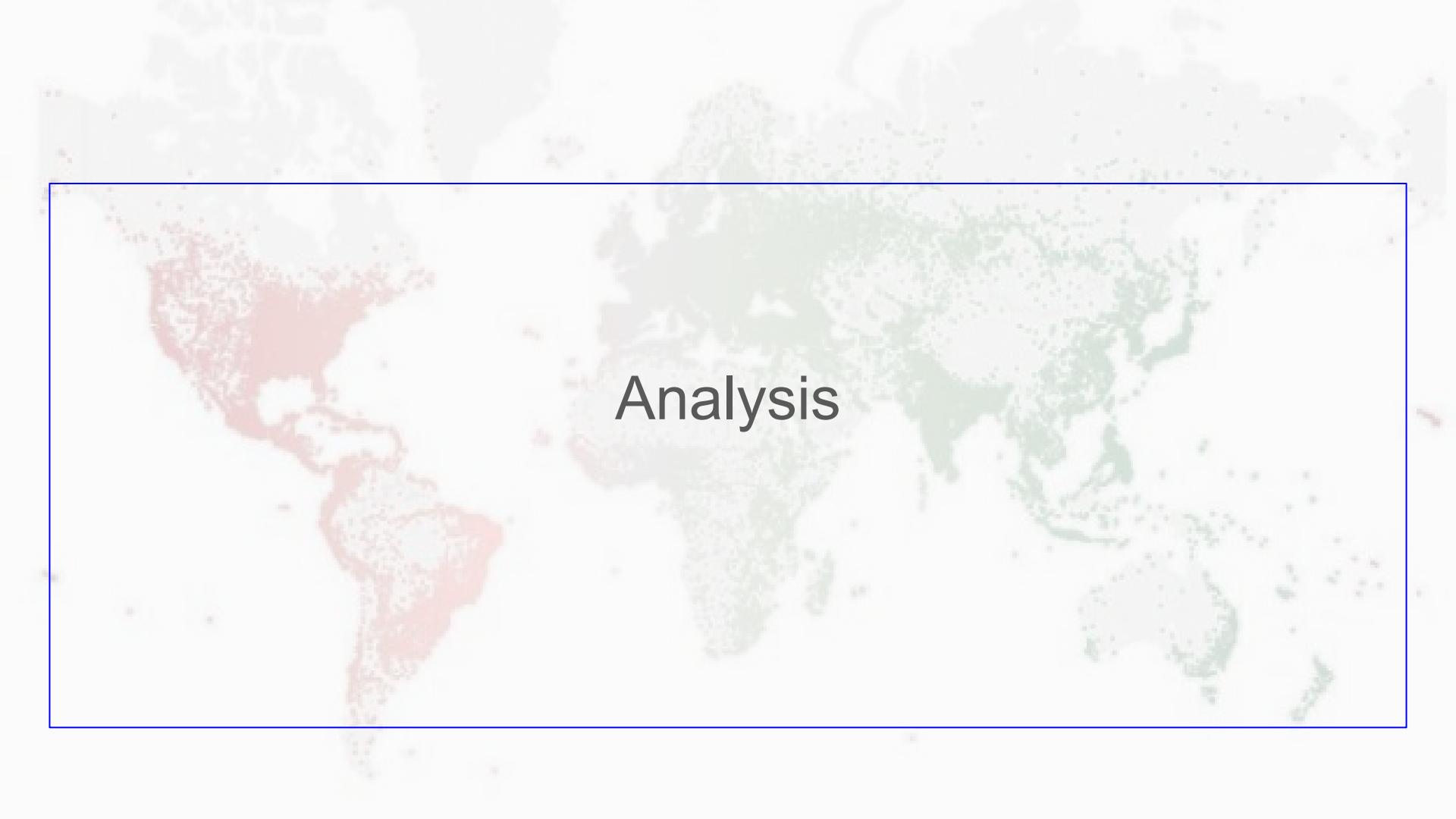
Per Minute : 440 pulls

Per Second : 6 pulls

Daily : 0.7GB

12 Weeks : ~60GB





Analysis

Analyses

- Table views:
 - city_score_temperature, city_score_weather_description, city_score_weather_name, city_score_rain, city_score_pressure, city_score_humidity, city_score_wind_speed
- How to we score accuracy for each city?
 - For 'numeric' forecasts (temperature, humidity, rain, wind speed), score is the 'variance' of the forecasts from the actual data

$$\text{Score} = \sqrt{\left(\sum(\text{deltas}^2) / N\right)}$$

- For 'string' forecasts (weather name, weather description), score is simply the % of times forecast is correct

$$\text{Score} = \frac{\text{accurate forecasts}}{\text{forecasts}} (\%)$$

Table views: city_score_rain

Postico File Edit View Navigate Connection Window Help

forecast-assistant forecast_assistant SQL Query Connected.

SQL Query

```
1 -- calculate deltas by retrieval date
2 -- delta_2_1 means diff between temp of day 2 and day 1 (COALESCE(rain_2,0) - COALESCE(rain_1,0))
3 DROP TABLE IF EXISTS city_deltas;
4 CREATE TEMPORARY TABLE city_deltas AS
5 select
6   city_id,
7   city_name,
8   country_code,
9   retrieval_date,
10  (COALESCE(rain_2,0) - COALESCE(rain_1,0)) as delta_2_1,
11  (COALESCE(rain_3,0) - COALESCE(rain_1,0)) as delta_3_1,
12  (COALESCE(rain_4,0) - COALESCE(rain_1,0)) as delta_4_1,
13  (COALESCE(rain_5,0) - COALESCE(rain_1,0)) as delta_5_1,
14  (COALESCE(rain_6,0) - COALESCE(rain_1,0)) as delta_6_1,
15  (COALESCE(rain_7,0) - COALESCE(rain_1,0)) as delta_7_1
16 from forecast_weather_flat
17 group by 1,2,3,4, rain_1, rain_2, rain_3, rain_4, rain_5, rain_6, rain_7
18 order by city_id;
19
20
21 -- calculate score for each city by this formula:
22 -- score = square root ( SUM(deltas ^ 2) / N )
23 -- we round score to 2 dp
24 DROP TABLE IF EXISTS city_score_rain;
25 CREATE TABLE city_score_rain AS
26 SELECT
27   d1.city_id AS city_id,
28   d1.city_name AS city_name,
29   d1.country_code AS country_code,
30   round(CAST((d1.deltas_sq_2_1/ d2.date_count) AS NUMERIC), 2) AS score_2,
31   round((d1.deltas_sq_3_1/ d2.date_count) AS NUMERIC), 2) AS score_3,
32   round((d1.deltas_sq_4_1/ d2.date_count) AS NUMERIC), 2) AS score_4,
33   round((d1.deltas_sq_5_1/ d2.date_count) AS NUMERIC), 2) AS score_5,
34   round((d1.deltas_sq_6_1/ d2.date_count) AS NUMERIC), 2) AS score_6,
35   round((d1.deltas_sq_7_1/ d2.date_count) AS NUMERIC), 2) AS score_7
36 FROM
37 (
38   SELECT
39     city_id,
40     city_name,
41     country_code,
42     SUM(delta_2_1 ^ 2) as deltas_sq_2_1,
43     SUM(delta_3_1 ^ 2) as deltas_sq_3_1,
44     SUM(delta_4_1 ^ 2) as deltas_sq_4_1,
45     SUM(delta_5_1 ^ 2) as deltas_sq_5_1,
46     SUM(delta_6_1 ^ 2) as deltas_sq_6_1,
```

cities cities2 city_score city_score_humidity city_score_pressure city_score_rain city_score_temperature city_score_weather_descri... city_score_weather_name city_score_wind_speed city_zones current_weather forecast_weather forecast_weather_flat pg_temp_4 pg_catalog information_schema

cast-assistant forecast_assistant city_score_rain Connected.



city_id	city_name	country_code	score_2	score_3	score_4	score_5	score_6	score_7
2974942	Serezin-du-Rhone	FR	6.74	12.21	9.27	9.31	13.81	11.07
289	Vakrah	QA	0	0	0.18	0	0	0
3616253	Siuna	NI	20.21	19.53	12.89	17.83	15.76	16.42
5030856	Hoyt Lakes	US	8.13	10.57	8.34	9.44	12.75	12.24
2094027	Kieta	PG	28.7	28.1	28.75	17.36	14.38	27.97
5354943	Half Moon Bay	US	0	0	0.05	0	0.32	0.42
511287	Perevoz	RU	13.09	5.15	3.46	4.51	4.87	5.92
481608	Troitsk	RU	3.42	6.05	4.56	3.28	4.77	5.49
2181742	Taupo	NZ	17.14	12.26	9.17	20.21	17.44	6.92
1847968	Zushi	JP	18.32	9.86	27.99	6.59	4.14	6.17
2511287	San Miguel	ES	0.06	0	0	0	0.33	0.09
1683852	Talisayan	PH	5.38	9.98	9.62	11.47	20.82	9.63
2609906	Baltiysk	RU	13.1	15.32	5.13	5.61	4.92	7.86
5866583	Kodiak	US	3	3.16	2.17	7.39	4.43	2.9
3652567	Quevedo	EC	3.42	3.47	2.02	9.56	3.34	5.54
2169237	Deagon	AU	6.11	3.51	3.29	12.36	14.96	18.51
609906	Fort-Shevchenko	KZ	2.04	2.99	2.99	6.78	4.09	4.13
711635	Borodyanka	UA	3.82	5.06	3.09	6.16	5.58	9.4
4031742	Egvekinot	RU	5.71	5.05	3.88	1.31	4.24	5.68
5364782	Lakeside	US	0.13	0.09	0.09	0.22	0.12	0.15
2073140	Darlington	AU	8.32	9.44	10.19	8.55	9.29	7.16
3726540	Dessalines	HT	7.26	8.18	6.6	4.98	6.49	8.91
742987	Kizilirmak	TR	2.44	2.64	2.51	3.17	2.8	6.44

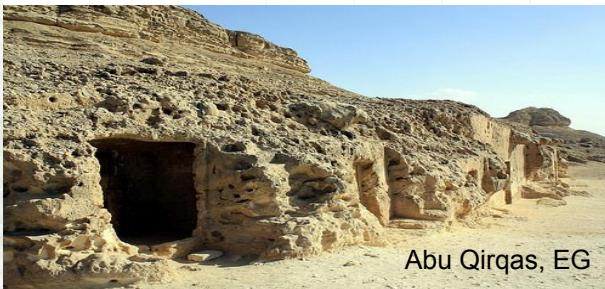
forecast_assistant city_score_temperature Connected.

city_id	city_name	country_code	score_2	score_3	score_4	score_5	score_6	score_7
2094027	Kieta	PG	0.89	1.19	1.16	1.34	1.03	1.33
4031742	Egvekinot	RU	1.82	2.03	3.05	2.51	4.03	4.24
2181742	Taupo	NZ	1	4.86	4.1	5.93	6.49	6.37
711635	Borodyanka	UA	4.18	4.79	5.57	6.56	7.45	7.33
4883207	Alsip	US	3.39	4.82	5.11	5.27	5.03	4.52
742987	Kizilirmak	TR	4.02	4.69	5.25	5.53	6.04	5.65
1847968	Zushi	JP	2.65	2.84	2.74	2.83	2.59	2.3
3726540	Dessalines	HT	5.85	6.66	8.76	10.32	10.97	10.81
5030856	Hoyt Lakes	US	4.59	4.48	4.28	5.47	4.89	6.29



forecast_assistant city_score_weather_description

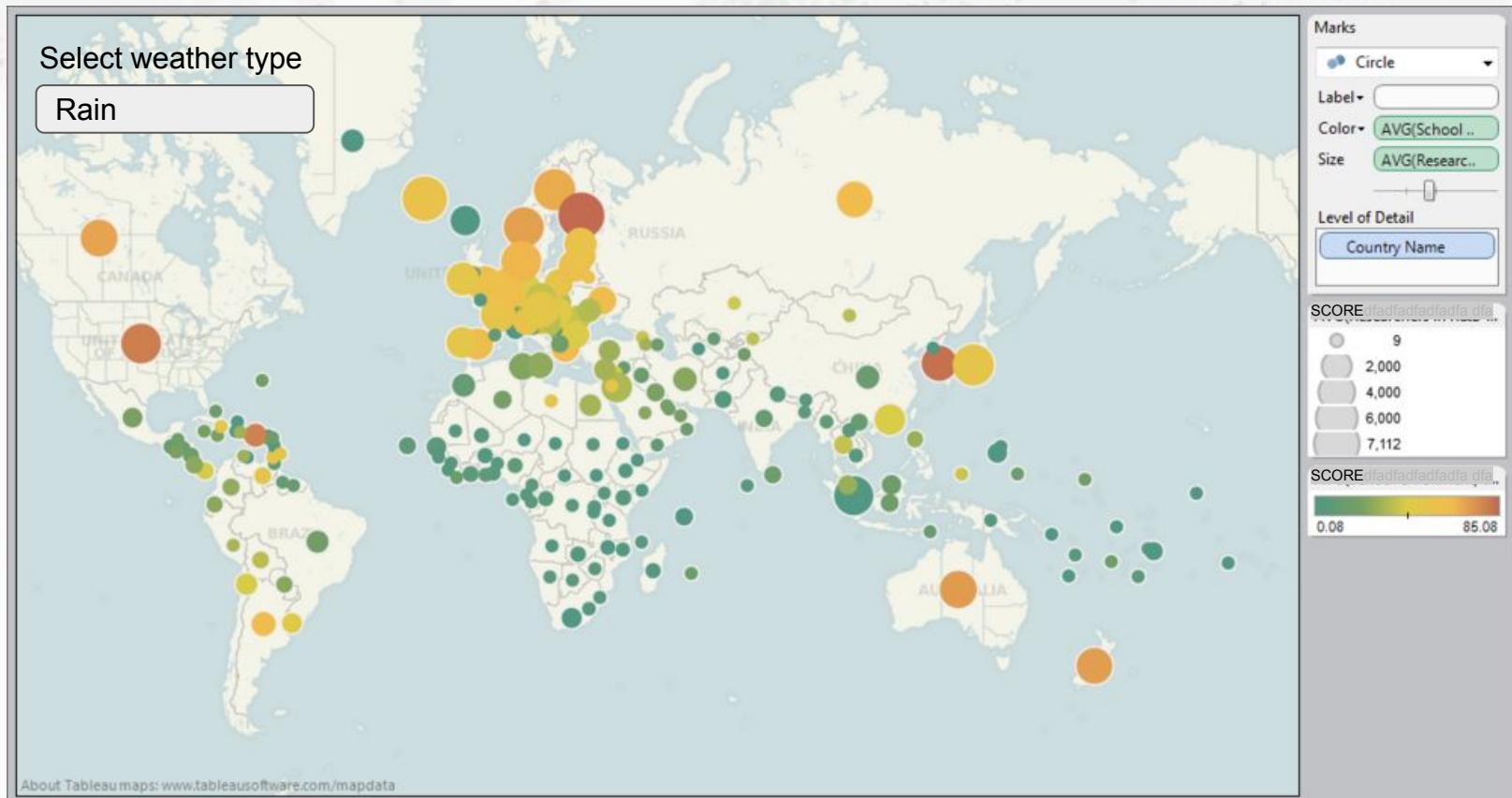
city_id	score_2	score_3	score_4	score_5	score_6	score_7
289915	0.94	1	0.94	0.88	0.88	0.94
362277	1	1	1	1	0.94	0.94



2073140	0.29	0.12	0.18	0.41	0.24	0.24
2094027	0.18	0.18	0.18	0.18	0.06	0.18
2169237	0.4	0.4	0.4	0.33	0.33	0.2
2181742	0.2	0.2	0.27	0.27	0.07	0.2
2511287	0.55	0.45	0.45	0.36	0.45	0.64
2609906	0.07	0.36	0.14	0.21	0.36	0.21



Map the world with weather scores



Project Conclusion

- Quality of weather forecast varies based on the location and the type of forecast
- We demonstrated that this is valuable information to be able to attach a numeric value to how much we should trust a particular weather forecast
- Hopefully, this would bring value to tourists, construction workers, farmers, campers and anyone who watches the weather to plan their daily activities

Postmortem

What went well?

- Team meetings every week since May.
- Setup Slack.com channel to improve team communications.
- Project played to team members individual strengths.

What could have gone better?

- More time to experiment with technology and to develop greater competencies.
- Longer semester to tweak production system and to play with analysis.

Future Work

- Collect forecast data at 3-hour intervals so that we could have more accurate and granular weather evaluation
- Include data for a variety of sources to be able to have more comprehensive comparison of the quality of different forecasters
- Collect data for a longer period of time to be able to evaluate accuracies at different times of year
- Build:
 - REST API
 - Web Page
 - Javascript library



Q&A

Q &A

Q&A

