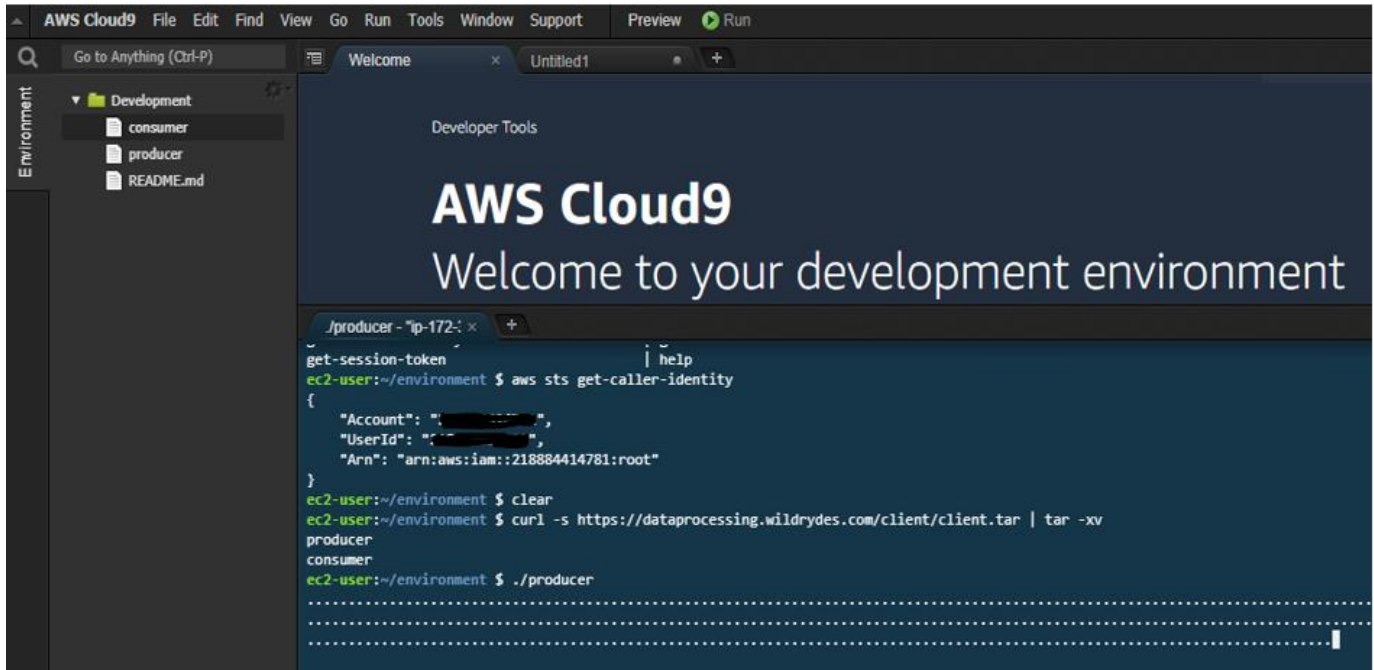
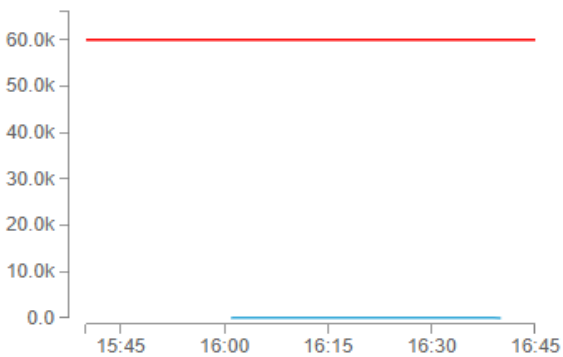


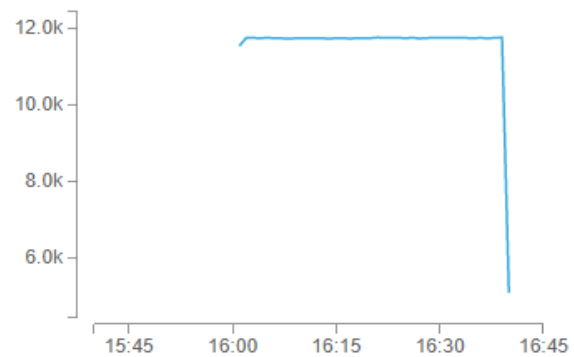
Cloud9 SDK



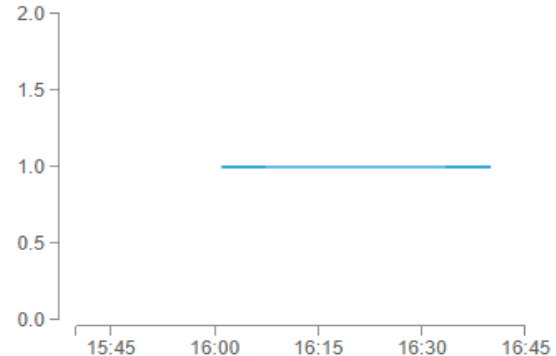
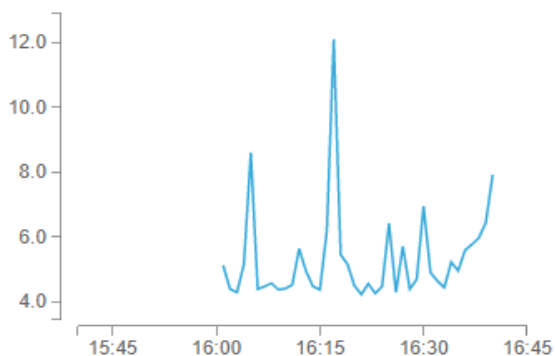
Kinesis DataStream Put Record Monitor Output



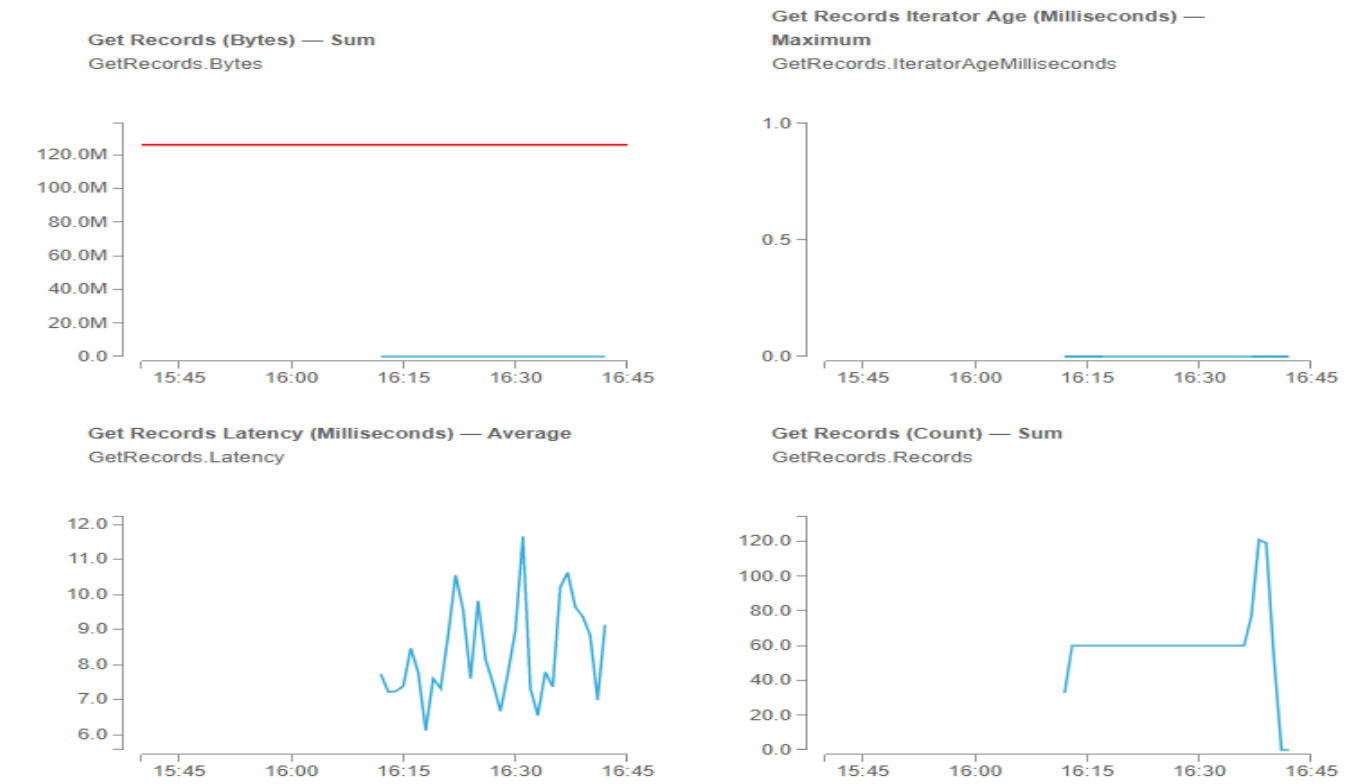
Put Record Latency (Milliseconds) — Average
PutRecord.Latency



Put Record Success (Percent) — Average
PutRecord.Success



Kinesis dataStream Get Records



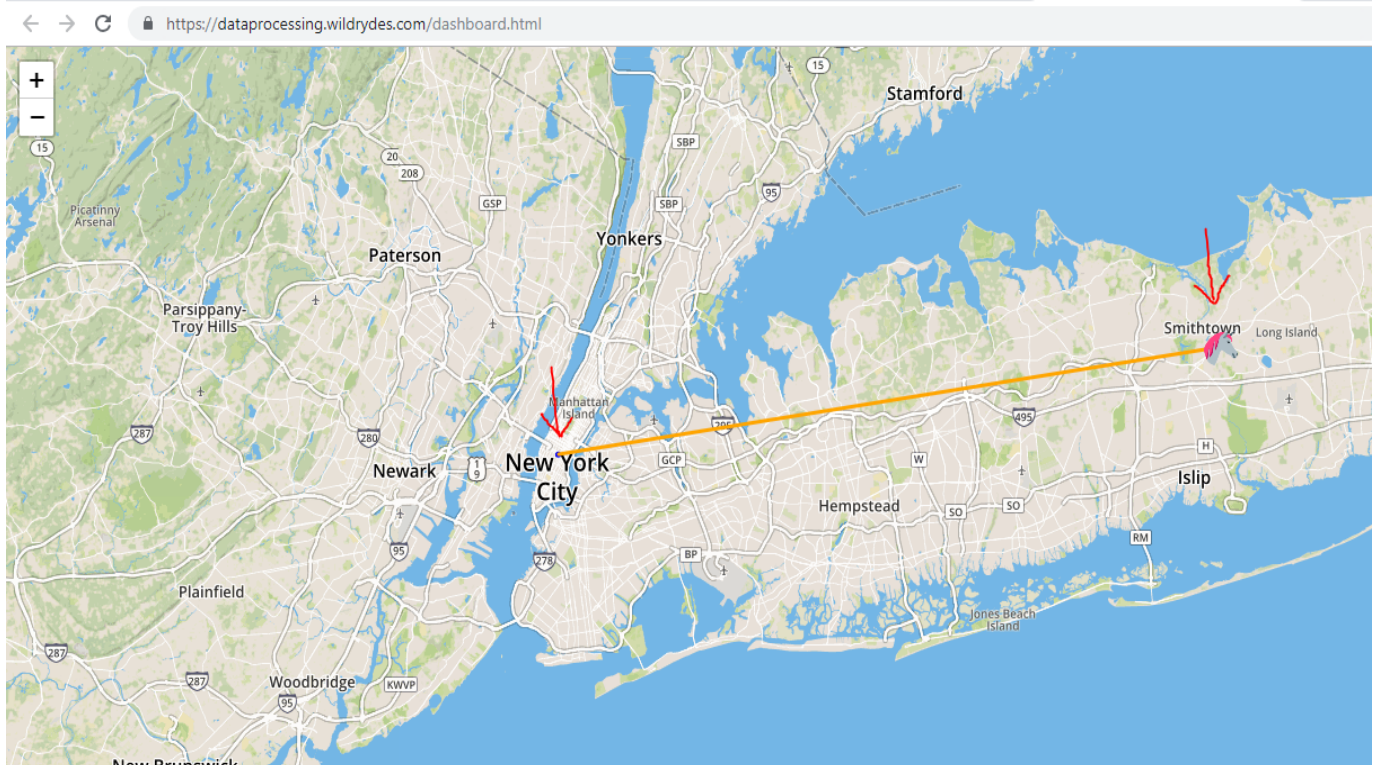
Consumer - output – displays producer data in Json

```

AWS Cloud9
Welcome to your development environment

./producer - "ip-172-..." x  ./consumer - "ip-172-..." x  +
ec2-user:~/environment $ ./consumer
{
  "Distance": 30.649337509799484,
  "HealthPoints": 178,
  "Latitude": 40.776433667493976,
  "Longitude": -73.74359344074307,
  "MagicPoints": 168,
  "Name": "Shadowfax",
  "StatusTime": "2018-10-28 23:12:27.515"
}
{
  "Distance": 29.347920735762774,
  "HealthPoints": 178,
  "Latitude": 40.77647252875948,
  "Longitude": -73.74324904794587,
  "MagicPoints": 167,
  "Name": "Shadowfax",
  "StatusTime": "2018-10-28 23:12:28.515"
}
{
  "Distance": 29.585239771966307,
  "HealthPoints": 178,
  "Latitude": 40.77651170427273,
  "Longitude": -73.74290187004755,
  "MagicPoints": 167,
  "Name": "Shadowfax",
  "StatusTime": "2018-10-28 23:12:29.515"
}
```

Output – Dashboard display stream data results on User Interface



Kinesis Analytics application - Real Time data analytics output

Amazon Kinesis
Dashboard
Data Streams
Data Firehose
Data Analytics
Video Streams
External resources
What's new

Real-time analytics

[Save and run SQL](#) [Add SQL from templates](#) [Download SQL](#) [SQL reference guide](#) [Kinesis data generator tool](#)

```
8 *  
9 * Get started by clicking "Add SQL from templates" or pull up the  
10 * documentation and start writing your own custom queries.  
11 */  
12  
13 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (  
14   "Name"          VARCHAR(16),  
15   "StatusTime"    TIMESTAMP,  
16   "Distance"      SMALLINT,  
17   "MinMagicPoints" SMALLINT,  
18   "MaxMagicPoints" SMALLINT,  
19   "MinHealthPoints" SMALLINT,  
20   "MaxHealthPoints" SMALLINT  
21 )
```

Source data **Real-time analytics** **Destination** Application status: RUNNING

Streaming data
☒ SOURCE_SQL_STREAM_001

Reference data (optional) ⓘ
[Connect reference data](#)

The streaming data below is a sample from Kinesis data stream [wildrydes](#)

Actions ▼

ROWTIME	Distance	HealthPoints	Latitude	Longitude
TIMESTAMP	DOUBLE	INTEGER	DOUBLE	DOUBLE
2018-10-29 00:36:53.24	29.26101129557579	157	40.66181590373277	-74.103995327

DataStream Output

Source data

Real-time analytics

Destination

Application status: RUNNING

Streaming data

SOURCE_SQL_STREAM_001

Reference data (optional)

Connect reference data

The streaming data below is a sample from Kinesis data stream [wildrydes](#)

Actions

Filter by column name

ROWTIME TIMESTAMP	Distance DOUBLE	HealthPoints INTEGER	Latitude DOUBLE	Longitude DOUBLE
2018-10-29 00:36:53.24	29.26101129557579	157	40.66181590373277	-74.103995327
2018-10-29 00:36:54.21	29.924868941016946	157	40.661628115893826	-74.104250581
2018-10-29 00:36:55.232	30.2075052610205	156	40.66143855442427	-74.104508244
2018-10-29 00:36:56.239	29.702447864350425	156	40.66125216234662	-74.104761599
2018-10-29 00:36:57.245	30.14273729795912	156	40.66106300731612	-74.105018709
2018-10-29 00:36:58.253	30.748713454495714	156	40.66087004959721	-74.105280986
2018-10-29 00:36:59.219	29.65834731264184	156	40.66068393426419	-74.105533963
2018-10-29 00:37:00.245	30.55963076766224	155	40.66049216309781	-74.105794626
2018-10-29 00:37:01.252	30.71508479828426	155	40.660299416409146	-74.106056615
2018-10-29 00:37:02.218	29.30530029190015	155	40.66011551655574	-74.106306578
2018-10-29 00:37:03.244	29.864936514907853	154	40.65992810481137	-74.106561314

After analytics – summery results

Source data

Real-time analytics

Destination

Application status: RUNNING

In-application streams:

DESTINATION_SQL_STREAM

error_stream

Pause results

New results are added every 2-10 seconds. The results below are sampled.

☒ Scroll to bottom when new results arrive.

Filter by column name

ROWTIME	Name	StatusTime	Distance	MinMagicPoints	MaxMa
2018-10-29 00:37:00.0	Shadowfax	2018-10-29 00:37:00.0	602	136	140
2018-10-29 00:38:00.0	Shadowfax	2018-10-29 00:38:00.0	1805	140	152

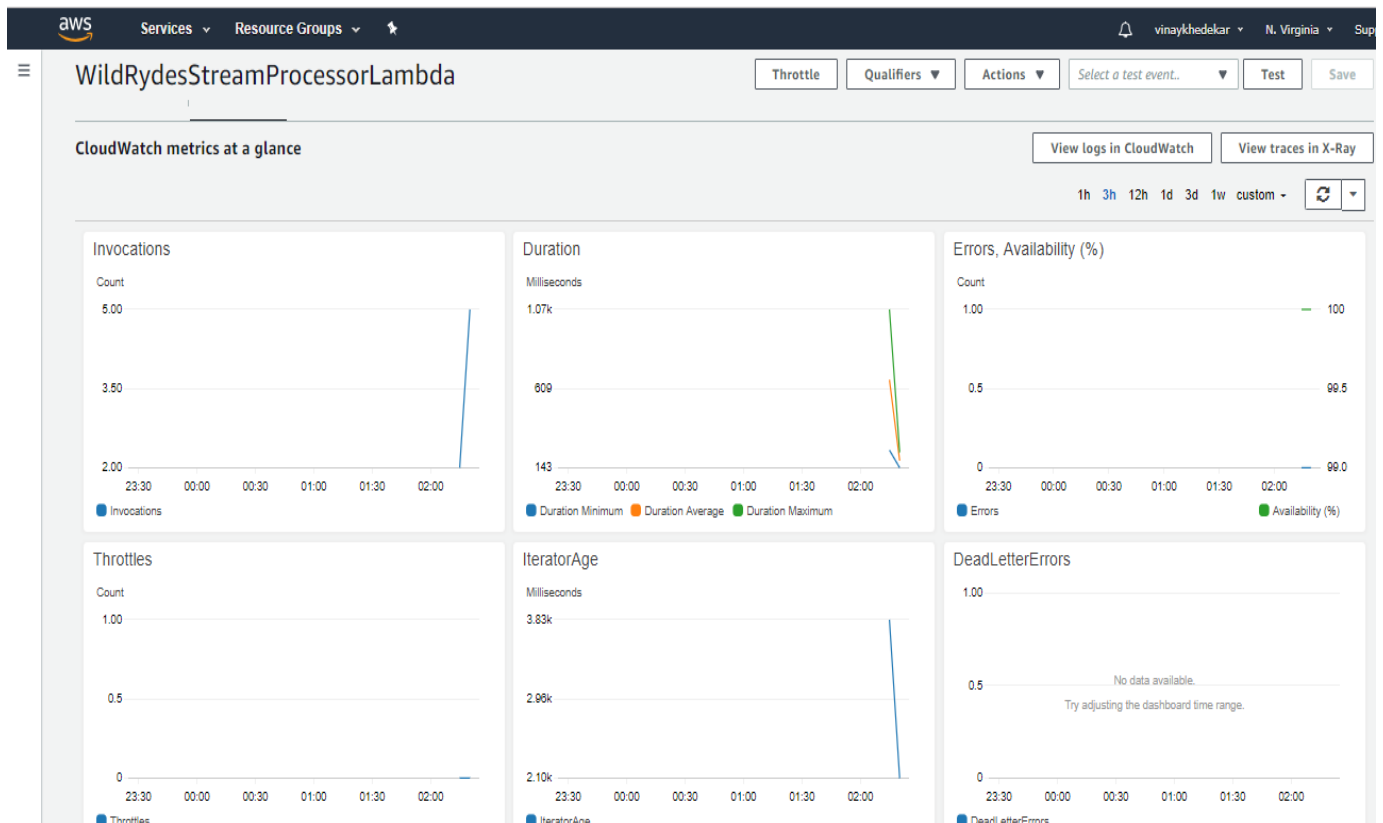
Close

Store Stream data into DynamoDB

The screenshot shows the AWS Management Console interface for a DynamoDB table named `UnicornSensorData`. The left sidebar contains navigation links for DynamoDB, Dashboard, Tables, Backups, Reserved capacity, Preferences, DAX, and various subnets and parameter groups. The main content area displays the table's details, including a search bar, a 'Create item' button, and a table of data. The table has columns for Name, StatusTime, Distance, MaxHealthPoints, MaxMagicPoints, MinHealthPoints, and MinMagicPoints. The data shows a series of sensor readings for a user named 'vinay' over time.

Name	StatusTime	Distance	MaxHealthPoints	MaxMagicPoints	MinHealthPoints	MinMagicPoints
vinay	2018-10-29 02:18:00.000	1312	155	153	150	148
vinay	2018-10-29 02:19:00.000	1804	163	160	152	153
vinay	2018-10-29 02:20:00.000	1800	161	164	154	157
vinay	2018-10-29 02:21:00.000	1797	161	163	154	155
vinay	2018-10-29 02:22:00.000	1797	161	163	153	153
vinay	2018-10-29 02:23:00.000	1802	166	164	161	154
vinay	2018-10-29 02:24:00.000	1805	169	171	162	155

Lambda monitoring logs – triggered when new data stream produced



Amazon Athena – In place query on raw stream data

The screenshot shows the Amazon Athena Query Editor interface. On the left, the 'Database' dropdown is set to 'sampledb'. Below it, a list of tables is shown: 'elb_logs' and 'wildrydes'. The 'Views' section is empty. The main query editor area contains a SQL statement to create an external table named 'wildrydes' with columns: Name (string), StatusTime (timestamp), Latitude (float), Longitude (float), Distance (float), HealthPoints (int), and MagicPoints (int). The table is located in an S3 bucket at 's3://wildrydes-data-bucket/'. The query is executed successfully, as indicated by the 'Run query' button and the 'Query successful.' message in the results section.

```
1 CREATE EXTERNAL TABLE IF NOT EXISTS wildrydes (  
2     Name string,  
3     StatusTime timestamp,  
4     Latitude float,  
5     Longitude float,  
6     Distance float,  
7     HealthPoints int,  
8     MagicPoints int  
9 )  
10 ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'  
11 LOCATION 's3://wildrydes-data-bucket/';
```

Run query Save as Create (Run time: 0.62 seconds, Data scanned: 0KB)

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

Results

Query successful.

Kinesis data firehose - Query operation on data stored in s3 bucket

The screenshot shows the Amazon Athena Query Editor interface. On the left, the 'Database' dropdown is set to 'sampledb'. Below it, a list of tables is shown: 'elb_logs' and 'wildrydes'. The 'Views' section is empty. The main query editor area contains a SQL statement to select all data from the 'wildrydes' table. The query is executed successfully, as indicated by the 'Run query' button and the 'Query successful.' message in the results section. The results are displayed as a table with 7 rows and 7 columns: name, statustime, latitude, longitude, distance, healthpoints, and magicpoints.

```
1 SELECT * FROM wildrydes;
```

Run query Save as Create (Run time: 1.68 seconds, Data scanned: 21.6KB) Format query

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

Results

	name	statustime	latitude	longitude	distance	healthpoints	magicpoints
1	vinay	2018-10-29 02:48:28.427	40.749428	-73.985214	29.11478	150	151
2	vinay	2018-10-29 02:48:29.427	40.749706	-73.985245	30.858177	149	151
3	vinay	2018-10-29 02:48:30.427	40.74997	-73.985275	29.198677	149	151
4	vinay	2018-10-29 02:48:31.427	40.750237	-73.985306	29.64253	149	152
5	vinay	2018-10-29 02:48:32.427	40.750515	-73.985344	30.872377	148	151
6	vinay	2018-10-29 02:48:33.427	40.750793	-73.985374	30.81427	148	150
7	vinay	2018-10-29 02:48:34.427	40.75107	-73.985405	30.355764	147	149