Test Cases:

DataSource

1- DatasourceTest file inside Test package contains the test case for datasources.

2- when it is run, it scans the resource/datasource directory and stores all the files inside into a list.

3- Text files stored in the resource/datasource folder contain the configuration of a datasource in json format.

4- DatasourceTest is configured to run every particular interval specified in test.properties file once it is run manually.

Following are some examples of ds file configuration:-

A- for ds with format as file and wrapper as csv field

{

        "Name": "AQI\_CSV",

        "Theme": "AQI",

        "Url": "<http://localhost:8085/eventshoplinux/ip.csv>",

        "Format": "file",

        "User\_Id": 78,

        "Syntax" : "\"timestamp\":\"DATETIME\",\"theme\":\"STRING\",\"value\":\"NUMBER\",\"loc\":{\"lon\":\"NUMBER\",\"lat\":\"NUMBER\"}",

        "Time\_Window": 300000,

        "Latitude\_Unit": 0.2,

        "Longitude\_Unit": 0.2,

        "boundingbox": "32.249974,-123.969727,42.000325,-114.257813",

        "Sync\_Time": 300000,

        "Wrapper\_Name": "csvField",

        "Wrapper\_Key\_Value": "{\"datasource\_type\":\"point\",\"spatial\_wrapper\":\"sum\",\"lat\_index\" :0,\"lon\_index\":1,\"val\_index\":2}",

        "Bag\_Of\_Words":""

}

B- for ds with format as Stream and wrapper as twitter

{

        "Name": "Iphone",

        "Theme": "Iphone",

        "Url": "[www.twitter.com](http://www.twitter.com/)",

        "Format": "stream",

        "User\_Id": 78,

        "Syntax" : "\"timestamp\":\"DATETIME\",\"theme\":\"STRING\",\"value\":\"NUMBER\",\"loc\":{\"lon\":\"NUMBER\",\"lat\":\"NUMBER\"}",

        "Time\_Window": 300000,

        "Latitude\_Unit": 0.2,

        "Longitude\_Unit": 0.2,

        "boundingbox": "32.249974,-123.969727,42.000325,-114.257813",

        "Sync\_Time": 300000,

        "Wrapper\_Name": "twitter",

        "Wrapper\_Key\_Value": "{\"datasource\_type\":\"point\",\"spatial\_wrapper\":\"count\"}",

        "Bag\_Of\_Words":"#iphone6"

}

C- for ds with format as rest endpoint and wrapper as json

{

        "Name": "Weather",

        "Theme": "OpenWeather",

        "Url": "<http://api.openweathermap.org/data/2.1/find/station>",

        "Format": "rest",

        "User\_Id": 78,

        "Syntax" : "\"timestamp\":\"DATETIME\",\"theme\":\"STRING\",\"value\":\"NUMBER\",\"loc\":{\"lon\":\"NUMBER\",\"lat\":\"NUMBER\"}",

        "Time\_Window": 300000,

        "Latitude\_Unit": 0.2,

        "Longitude\_Unit": 0.2,

        "boundingbox": "32.249974,-123.969727,42.000325,-114.257813",

        "Sync\_Time": 300000,

        "Wrapper\_Name": "json",

        "Wrapper\_Key\_Value": "{\"datasource\_type\": \"point\",\"spatial\_wrapper\": \"sum\",\"isList\": true,\"rootElement\": \"\",\"tokenizeElement\": \"list\",\"lat\_path\": \"coord.lat\",\"lon\_path\": \"coord.lon\",\"val\_path\": \"wind.speed\",\"date\_time\_path\": \"dt\",\"date\_time\_format\": \"Long\",\"queryParams\": [{\"lat\": \"38.903858\",\"lon\": \"-105.644531\",\"radius\": \"40\"}]}"

}

D- for ds with format as rest endpoint and wrapper as xml

{

        "Name": "XMLtest",

        "Theme": "XMLtest",

        "Url": "<http://uk-air.defra.gov.uk/assets/rss/current_site_levels.xml>",

        "Format": "rest",

        "User\_Id": 78,

        "Syntax" : "\"timestamp\":\"DATETIME\",\"theme\":\"STRING\",\"value\":\"NUMBER\",\"loc\":{\"lon\":\"NUMBER\",\"lat\":\"NUMBER\"}",

        "Time\_Window": 300000,

        "Latitude\_Unit": 0.2,

        "Longitude\_Unit": 0.2,

        "boundingbox": "32.249974,-123.969727,42.000325,-114.257813",

        "Sync\_Time": 300000,

        "Wrapper\_Name": "xml",

        "Wrapper\_Key\_Value": "{\"datasource\_type\":\"point\", \"spatial\_wrapper\":\"sum\", \"isList\":true, \"rootElement\": \"rss/channel\", \"tokenizeElement\": \"item\", \"loc\_lat\_path\" :\"/item/description/text()\", \"loc\_lat\_grok\":\": %{NUMBER:loc\_lat}&deg;%{GREEDYDATA:waste1}%{NUMBER:loc\_lon}&quot;N%{GREEDYDATA:waste2}%{WORD:status} at index %{NUMBER:index}\",\"loc\_lon\_path\":\"/item/description/text()\", \"loc\_lon\_grok\":\": %{NUMBER:loc\_lat}&deg;%{GREEDYDATA:waste1}%{NUMBER:loc\_lon}&quot;N%{GREEDYDATA:waste2}%{WORD:status} at index %{NUMBER:index}\",\"value\_path\":\"/item/description/text()\",\"value\_grok\":\": %{NUMBER:lat}&deg;%{GREEDYDATA:waste1}%{NUMBER:lon}&quot;N%{GREEDYDATA:waste2}%{WORD:status} at index %{NUMBER:value}\",\"timestamp\_path\":\"/item/pubDate/text()\", \"timestamp\_format\":\"dd MMM yyyy hh:mm:ss\", \"timestamp\_grok\":\"%{WORD:garbage}, %{GREEDYDATA:timestamp} +%{NUMBER:garbage1}\"}"

}

Query Test Case

1- QueryTest file inside Test package contains the test case for queries.

2- when it is run, it scans the resource/query directory and stores all the files inside into a list.

3- Text files stored in the resource/query folder contain the configuration of a query in json format.

4- Queries can be created as specified in the configuration text files, but will be enabled only if the datasource(s) on which the query is performed exists and is already enabled.

5- QueryTest is configured to run every particular interval specified in test.properties file once it is run manually.

Following are some examples of query file configuration:-

A- for Filter Query

{

"query\_creator\_id": 78,

"query\_name":"Test1",

"query\_esql": "[{\"qID\":1,\"patternType\":\"filter\",\"dataSources\":[\"ds54\"],\"maskMethod\":\"\",\"coords\":[32.249974,-123.969727,42.000325,-114.257813],\"placename\":\"New York City\",\"filePath\":\"\",\"valRange\":[\"0\",\"232\"],\"timeRange\":[1,1],\"normMode\":\"true\",\"normVals\":[\"0\",\"100\"],\"queryName\":\"Test1\",\"timeWindow\":300000,\"latitudeUnit\":\"0.2\",\"longitudeUnit\":\"0.2\",\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"timeType\":\"0\",\"spatial\_wrapper\":\"sum\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\"}]",

"latitude\_unit": 0.2,

"longitude\_unit": 0.2,

"boundingbox": "32.249974,-123.969727,42.000325,-114.257813"

}

B- for Grouping Query

{

"query\_creator\_id": 78,

"query\_name":"Test2",

"query\_esql": "[{\"qID\":1,\"patternType\":\"grouping\",\"dataSources\":[\"ds3\"],\"method\":\"Threshold\",\"numGroup\":\"\",\"thresholds\":[\"30\",\"70\"],\"split\":\"False\",\"doColoring\":\"True\",\"colorCodes\":[\"green\",\" yellow\",\" red\"],\"queryName\":\"Test2\",\"timeWindow\":300000,\"latitudeUnit\":\"0.2\",\"longitudeUnit\":\"0.2\",\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\",\"timeRange\":[1,1]}]",

"latitude\_unit": 0.2,

"longitude\_unit": 0.2,

"boundingbox": "32.249974,-123.969727,42.000325,-114.257813"

}

C- for Temporal Char Query

{

"query\_creator\_id": 78,

"query\_name":"Test3",

"query\_esql": "[{\"qID\":1,\"patternType\":\"tpchar\",\"dataSources\":[\"ds3\"],\"tcTimeWindow\":\"65\",\"tmplCharOperator\":\"Displacement\",\"queryName\":\"Test3\",\"timeWindow\":300000,\"latitudeUnit\":2,\"longitudeUnit\":2,\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\",\"timeRange\":[1,1]}]",

"latitude\_unit": 2,

"longitude\_unit": 2,

"boundingbox": "32.249974,-123.969727,42.000325,-114.257813"

}

D- for Aggregation Query

{

"query\_creator\_id": 78,

"query\_name":"Test4",

"query\_esql": "[{\"qID\":1,\"patternType\":\"filter\",\"dataSources\":[\"ds3\"],\"maskMethod\":\"\",\"coords\":[33.694638 , -118.410645,34.150454 , -117.861328],\"placename\":\"los angeles\",\"filePath\":\"\",\"valRange\":[\"0\",\"232\"],\"timeRange\":[1,1],\"normMode\":\"true\",\"normVals\":[\"0\",\"100\"],\"queryName\":\"Test4\",\"timeWindow\":300000,\"latitudeUnit\":\"0.2\",\"longitudeUnit\":\"0.2\",\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\"},{\"qID\":2,\"patternType\":\"filter\",\"dataSources\":[\"ds4\"],\"maskMethod\":\"\",\"coords\":[33.694638 , -118.410645,34.150454 , -117.861328],\"placename\":\"los angeles\",\"filePath\":\"\",\"valRange\":[\"-33\",\"3020\"],\"timeRange\":[1,1],\"normMode\":\"true\",\"normVals\":[\"0\",\"100\"],\"queryName\":\"Test4\",\"timeWindow\":300000,\"latitudeUnit\":\"0.2\",\"longitudeUnit\":\"0.2\",\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\"},{\"qID\":3,\"patternType\":\"aggregation\",\"dataSources\":[\"Q1\",\"Q2\"],\"values\":[],\"scalarFirst\":\"false\",\"aggOperator\":\"AggAvg\",\"valueNorm\":\"true\",\"normedRange\":[\"0\",\"100\"],\"queryName\":\"Test4\",\"timeWindow\":300000,\"latitudeUnit\":\"0.2\",\"longitudeUnit\":\"0.2\",\"queryStatus\":\"S\",\"qryCreatorId\":\"78\",\"boundingBox\":\"32.249974,-123.969727,42.000325,-114.257813\",\"timeRange\":[1,1]}]",

"latitude\_unit": 0.2,

"longitude\_unit": 0.2,

"boundingbox": "32.249974,-123.969727,42.000325,-114.257813"

}

Alert

1- AlertTest file inside Test package contains the test case for alerts.

2- when it is run, it scans the resource/alert directory and stores all the files inside into a list.

3- Text files stored in the resource/alert folder contain the configuration of an alert in json format.

4- Alerts can be created as specified in the configuration text files, but will be enabled only if the datasource(s) on which the alert is performed exists.

5- AlertTest is configured to run every particular interval specified in test.properties file once it is run manually.

Following are some examples of query file configuration:-

A- For Alert with solution

{

    "alertName" : "Alert1",

    "alertType" : 1,

    "theme" : "test",

    "alertSource" : "ds3",

    "safeSource" : "ds4",

    "alertMessage" : "PM2.5 content high in {$coordinate} in {$geoAddress} because of high traffic.",

    "alertMin" : 80,

    "alertMax" : 100,

    "safeMin" : 20,

    "safeMax" : 60,

    "alertStatus" : 0,

    "user\_id" : 78,

    "email" : "[abhisek@abc.com](mailto:abhisek@abc.com)",

    "resultEndpoint" : "<http://localhost:8085/eventshoplinux/rest/alertwebservice/resultAlert/>",

    "lat" : "55.00000",

    "lng" : "66.00000",

    "radius" : "6.00000"

}

B- For Alert without solution

{

    "alertName" : "Alert2",

    "alertType" : 2,

    "theme" : "test",

    "alertSource" : "ds3",

    "safeSource" : "ds4",

    "alertMessage" : "PM2.5 content high in {$coordinate} in {$geoAddress} because of high traffic.",

    "alertMin" : 80,

    "alertMax" : 100,

    "safeMin" : 20,

    "safeMax" : 60,

    "alertStatus" : 0,

    "user\_id" : 78,

    "email" : "abhisek@abc.com",

    "resultEndpoint" : "http://localhost:8085/eventshoplinux/rest/alertwebservice/resultAlert/",

    "lat" : "55.00000",

    "lng" : "66.00000",

    "radius" : "6.00000"

}

Rule Test Case:

1- RuleTest file inside Test package contains the test case for Rules. when it is run, it scans the resource/rule directory and stores all the files inside into a list.

3- Text files stored in the resource/rule folder contain the configuration of a rule in json format.

4- Rule Test is configured to run every particular interval specified in test.properties file once it is run manually.

Following are some examples of rule file configuration:-

Rule with “>” greater than operator.

{

"source": "ds30",

"rules": [

{

"dataField": "value",

"ruleOperator": ">",

"ruleParameters": "50"

}

],

"extractFields": "location.lat,location.lon,value,"

}

Rule with Geo Coordinates.

{

"source": "ds30",

"rules": [

{

"dataField": "location",

"ruleOperator": "coordinates",

"ruleParameters": "7.45,69.25,32.6,92.0"

}

],

"extractFields": "location.lat,location.lon,value,"

}

Rule with Equals operator. Which is used to compare strings.

{

"source": "ds30",

"rules": [

{

"dataField": "theme",

"ruleOperator": "equals",

"ruleParameters": "Krumbs\_SB\_Test"

}

],

"extractFields": "location.lat,location.lon,value,"

}