Workshop Anomaly Detection

met Azure Stream Analytics

Met behulp van deze workshop creëer je je eigen stream analytics jobs en detecteer je waarden afwijken van de verwachting. Deze afwijkingen willen wij detecteren en aangeven dat deze vaker achter elkaar optreden.

Voor deze hands-on sessie maak je gebruik van Azure resources in de resource group ‘**rg-sig-stream-analytics’** en creëer je zelf resources die je aanmaakt in je eigen resource group.

# Voorbereiding:

Maak de volgende resources aan in je eigen resource group:

* Een storage account
* Een Stream Analytics Job

Start bij het toevoegen van de input en ouput bindings van de stream analytics job. Dit kan door te navigeren naar Inputs/Ouputs of binnen Query de input/output toe te voegen doormiddel van het plusje.

A screenshot of a computer screen

Description automatically generated with medium confidence A screenshot of a computer

Description automatically generated with medium confidence

## Input 1:

Voeg een event hub toe:

* Kies voor event hub namespace: **evhns-sig-stream-analytics**
* Creëer een nieuwe consumer group (met een referentie van je naam)
* Maak gebruik van een connectionstring voor de authenticatie

Deze event hub streamt iedere 4 seconden van vier verschillende devices telemetry. Dit wordt gebruikt als input stream voor de job.

## Input 2:

Voeg een reference input toe van SQL Database:

* Kies voor de **metadata** database op **server sqlsrv-sig-stream-analytics**
* Authentiseer met de SQL credentials:
  + Username: sigadmin
  + Password:
* Voeg als snapshot query een selectie van de gehele ‘Devices’ tabel

De tabel bevat metadata van de devices. Hiermee gaan wij onze telemetry uitbreiden. De tabel ziet er als volgt uit:

A screenshot of a computer

Description automatically generated

Outputs:

Je gaat twee outputs nodig hebben, beide verwijzen naar hetzelfde storage account met een verschillende container.

## Output 1:

Voeg een blob storage als output toe:

* Kies voor je eigen storage account
* Maak een container aan met de naam ‘Amsterdam’
* Maak gebruik van een connection string als authenticatie

## Output 2:

Voeg een blob storage als output toe:

* Kies voor je eigen storage account
* Maak een container aan met de naam ‘Utrecht’
* Maak gebruik van een connection string als authenticatie

# Aan de slag

We gaan starten om de query binnen de stream analytics job langzaam op te bouwen.

De code kan op twee manieren getest worden.

* Door de job te starten en te stoppen onder ‘Overview’ in de portal. Let op: Het starten en stoppen van de job kan een paar minuten duren. Tevens kan de code niet aangepast worden als de job aanstaat.
* Binnen de query kan ook getest worden. Dit kan onder ‘Query’. Hiervoor moeten eerst de input bindings geactiveerd worden. Deze haalt sample data op, waarop vervolgens de query wordt gedraaid. Door op de output bindings te klikken, zie je het query resultaat per output.

## STAP 1:

Test of de input binding van de event hub en de output binding van het storage account goed werken. Dit kan door gebruik te maken van de test query.

## STAP 2:

Verreik de query met de locatie data uit de metadata DB.

Routeer de juiste devices naar twee verschillende containers op het storage account. Devices in Utrecht naar de ‘Utrecht’ container, en devices in Amsterdam naar de ‘Amsterdam’ container.

## STAP 3:

Controleer van ieder bericht of de data uitschieters heeft in de temperatuur. Selecteer alleen de data dat een temperatuur heeft onder de -20 en boven de 50.

TIP: Maak gebruik van een WITH clause om de dataset te kunnen refereren in de output query’s.

## STAP 4:

Constateer per device wanneer er twee fouten op een rij optreden.

TIP: Maak hiervoor gebruik van nieuwe query die het volgende HoppingWindow gebruikt:

HoppingWindow(second, 12, 4)

## STAP 5:

Naast de uitschieters in temperatuur, willen wij ook controleren of de power waarde wordt meegegeven.

5A: Selecteer alle data waarvoor geldt dat de waarde van Power NULL is

5B: Controleer of Power wordt meegegeven in het bericht.

Selecteer de data waarbij geen power element in het bericht zit. Maak hiervoor gebruik van een User Defined Function dit kan bepalen.

## STAP 6:

Vervolgens gaan wij bepalen op welk moment de anomaly getriggerd gaat worden. Hiervoor gaan wij gebruik maken van een vergelijking van de data set met zijn eigen data set uit het verleden.

## STAP 7:

De laatste stap gaan wij bepalen of de anomaly een ‘raise’ of ‘resolve’ is. Maak gebruik van de vorige query en bepaal dat een oploping van 2 naar 3 fouten een raise is, en een afloping van fouten een resolve oplevert.