

Lab 051 : Fabric IQ ? Agents Real-Time Intelligence

Niveau : L300 Parcours : Tous les parcours Durée : ~75 min ? Coût : Gratuit ? Utilise un jeu de données simulé inclus (capacité Fabric optionnelle)

Ce que vous apprendrez

Comment les Eventstreams ingèrent des données de capteurs IoT en temps réel dans Microsoft Fabric Interroger des données en streaming avec KQL (Kusto Query Language) pour une détection instantanée des anomalies Détecter les anomalies de température, d'humidité et de stock à l'aide de règles basées sur des seuils Utiliser Fabric Activator pour déclencher des alertes automatisées lorsque des conditions sont remplies Construire un agent IA qui lit les données en temps réel et fait remonter des informations exploitables Analyser les modèles d'activité des entrepôts (ouvertures de portes, fréquence des capteurs) à travers les sites

Introduction

Fabric RTI Pipeline

Real-Time Intelligence (RTI) dans Microsoft Fabric est une plateforme entièrement managée pour capturer, transformer et agir sur les données en streaming ? le tout au sein de l'espace de travail Fabric. Contrairement aux pipelines batch traditionnels qui traitent les données des heures ou des jours après leur arrivée, RTI vous offre une visibilité en moins d'une seconde sur ce qui se passe *en ce moment*.

Le scénario

OutdoorGear Inc. exploite 5 entrepôts à travers les États-Unis (New York, Los Angeles, Chicago, Dallas et Seattle). Chaque entrepôt est équipé de capteurs IoT surveillant quatre métriques clés :

Type de capteur | Ce qu'il mesure | Seuil critique | temperature | Température ambiante (°C) | > 30°C (risque de dommages aux produits) | humidity | Humidité relative (%) | > 80% (risque de dommages par l'humidité) | stock_level | Unités restantes dans le bac | 30°C)

Équivalent KQL :

[diagram or code sample]

Équivalent Python :

[diagram or code sample]

Sortie attendue :

[diagram or code sample]

warning "Alerte critique" WH-NYC à 38°C est dangereusement élevé ? les denrées périssables et l'électronique peuvent être endommagés au-dessus de 35°C. Une règle Activator notifierait immédiatement le responsable de l'entrepôt de NYC et déclencherait le système CVC.

3b. Stock critique (80%)

Équivalent KQL :

[diagram or code sample]

Équivalent Python :

[diagram or code sample]

Sortie attendue :

[diagram or code sample]

Étape 4 : Analyse de l'activité des entrepôts

Analysez les événements d'ouverture de portes par entrepôt ? une activité anormalement élevée peut indiquer des problèmes de sécurité ou des périodes d'expédition de pointe :

[diagram or code sample]

Sortie attendue :

[diagram or code sample]

tip "Observation" WH-DAL est en tête avec 14 ouvertures de portes au total ? combiné avec ses deux anomalies de température (32°C et 35°C), les ouvertures fréquentes de portes pourraient laisser entrer l'air chaud. Un agent IA corrèlerait ces signaux : _« L'entrepôt de Dallas a une forte activité de portes ET des températures en hausse ? envisagez d'ajouter un rideau d'air au quai de chargement 3. »_

Étape 5 : Construire un tableau de bord d'alertes

Combinez toutes les anomalies dans un tableau de bord récapitulatif unique qu'un agent IA présenterait à un responsable des opérations d'entrepôt :

[diagram or code sample]

Sortie attendue :

[diagram or code sample]

info "Intégration d'un agent IA" En production, un agent IA interrogerait l'Eventhouse via KQL, exécuterait cette logique de détection d'anomalies et publierait un résumé en langage naturel dans un canal Teams ou par e-mail. Fabric Activator gère les alertes automatisées, tandis que l'agent IA fournit l'*interprétation* ? transformant les chiffres bruts en recommandations exploitables.

? Exercice de correction de bugs

Le fichier lab-051/broken_alerting.py contient 3 bugs qui produisent des résultats incorrects de détection d'anomalies. Exécutez les auto-tests :

[diagram or code sample]

Vous devriez voir 3 tests échoués :

Test | Ce qu'il vérifie | Indice | Test 1 | Le seuil de température est paramétré | Le seuil est codé en dur à 50 au lieu d'utiliser le paramètre threshold | Test 2 | L'alerte de stock utilise la bonne comparaison | Les alertes de stock se déclenchent quand la valeur est *inférieure* au seuil, pas supérieure | Test 3 | Le taux d'anomalies est calculé par entrepôt | Le dénominateur devrait être les événements *par entrepôt*, pas le total des événements de tous les entrepôts |

Corrigez les 3 bugs et relancez jusqu'à voir ? All 3 tests passed.

? Vérification des connaissances

question "Q1 (Choix multiple) : Qu'est-ce qu'un Eventstream dans Microsoft Fabric ?"

- A) Un pipeline de traitement de données par lots qui s'exécute selon un calendrier - B) Un pipeline d'ingestion de données en temps réel managé qui capture les événements en streaming - C) Un outil de visualisation pour créer des tableaux de bord - D) Un service d'entraînement de modèles de machine learning

??? success "" Révéler la réponse" Correct : B) Un pipeline d'ingestion de données en temps réel managé qui capture les événements en streaming

Un Eventstream est le point d'entrée des données en temps réel dans Fabric. Il capture les événements provenant de sources comme IoT Hub, Kafka ou des applications personnalisées, les transforme en vol et les achemine vers des destinations comme un Eventhouse pour l'interrogation. Contrairement aux pipelines batch, les Eventstreams traitent les données en continu avec une latence inférieure à la seconde.

question "Q2 (Choix multiple) : Que fait Fabric Activator ?"

- A) Optimise les performances des requêtes KQL - B) Gère la capacité de stockage de l'Eventhouse - C) Déclenche des actions automatisées lorsque des conditions sur les données sont remplies - D) Convertit les données par lots en format streaming

??? success "" Révéler la réponse" Correct : C) Déclenche des actions automatisées lorsque des conditions sur les données sont remplies

Activator est le moteur de règles de Fabric pour les alertes en temps réel. Vous définissez des conditions (par ex. « température > 30°C ») et des actions (par ex. envoyer une notification Teams, déclencher un flux Power Automate). Il surveille en continu les résultats des requêtes KQL et déclenche des alertes dès qu'une condition est remplie ? sans interrogation périodique nécessaire.

question "Q3 (Exécuter le lab) : Combien de relevés de température dépassent 30°C ?"

Filtrez le DataFrame des événements pour sensor_type == "temperature" et value > 30.

??? success "? Révéler la réponse" 3

Trois anomalies de température dépassent 30°C : WH-NYC à 38°C, WH-DAL à 35°C et WH-DAL à 32°C. Le relevé de NYC à 38°C est le plus critique ? bien au-dessus du seuil de 35°C pour les dommages aux produits.

question "Q4 (Exécuter le lab) : Quel entrepôt a le plus de door_opens ?"

Filtrez pour sensor_type == "door_opens", regroupez par warehouse_id et faites la somme des valeurs.

??? success "? Révéler la réponse" WH-DAL (14 ouvertures de portes au total)

Dallas est en tête avec 14 ouvertures de portes au total, suivi de WH-NYC (12), WH-SEA (10), WH-CHI (9) et WH-LAX (7). Combiné avec les deux anomalies de température de Dallas, la forte activité de portes peut contribuer à l'accumulation de chaleur.

question "Q5 (Exécuter le lab) : Combien de relevés de stock sont en niveau critique (

Phase 3 · Simulation / Parcours Réel

Simulation Parcours Réel

Preserve the existing free simulation and its fixtures. Estimated cost: US\$0.

Reuse the same acceptance criteria against the documented hosted service. Before running, record a service-specific estimate and a hard spending cap. After validation: Supprimez déploiements, points de terminaison, index, traces et données de test après validation.

Kit d'accompagnement

Le build du site génère des téléchargements PDF et ZIP stables. Le ZIP contient le guide du lab et tous les fichiers d'accompagnement déclarés.

Télécharger le PDF généré Télécharger le ZIP généré

Supporting file: broken_alerting.py

/usr/bin/env python3 """ ? Broken Alerting Engine ? Lab 051 Bug-Fix Exercise

Analyzes sensor events for anomalies but has 3 bugs. Fix them and run the self-tests to verify.

Usage: python lab-051/broken_alerting.py """

import io import pandas as pd

def detect_temp_anomalies(events: pd.DataFrame, threshold: float = 30.0) -> pd.DataFrame:

"""Return temperature events above the threshold.""" temp = events[events["sensor_type"] == "temperature"] # ? Bug #1: Threshold is hardcoded to 50 instead of using the parameter
return temp[temp["value"] > 50]

def detect_low_stock(events: pd.DataFrame, threshold: float = 10.0) -> pd.DataFrame:

"""Return stock_level events below the threshold (critical stock).""" stock = events[events["sensor_type"] == "stock_level"] # ? Bug #2: Uses > instead of threshold
def calculate_anomaly_rate(events: pd.DataFrame, warehouse_id: str, anomaly_count: int) -> float: """Return the anomaly rate (%) for a specific warehouse.""" # ? Bug #3: Divides by total events across ALL warehouses # instead of events for this specific warehouse total = len(events) return anomaly_count / total * 100

def run_tests() -> bool: test_csv = ""

```

timestamp,warehouse_id,warehouse_city,sensor_type,value,unit
2026-01-01T00:00:00Z,W1,City1,temperature,35.0,celsius
2026-01-01T00:01:00Z,W1,City1,temperature,22.0,celsius
2026-01-01T00:02:00Z,W1,City1,stock_level,3.0,units
2026-01-01T00:03:00Z,W1,City1,stock_level,50.0,units
2026-01-01T00:04:00Z,W2,City2,temperature,28.0,celsius
2026-01-01T00:05:00Z,W2,City2,stock_level,15.0,units""
events = pd.read_csv(io.StringIO(test_csv)) passed = 0 failed = 0
# ?? Test 1: Temperature anomalies above 30°C ?????????????????????? # Only W1 at 35°C ?
1 anomaly anomalies = detect_temp_anomalies(events, threshold=30.0) if len(anomalies) == 1:
print(f"? Test 1 PASSED: temp anomalies = {len(anomalies)}") passed += 1 else: print(f"?
Test 1 FAILED: temp anomalies = {len(anomalies)} (expected 1)") failed += 1
# ?? Test 2: Low stock below 10 units ?????????????????????????????? # Only W1 at 3 ? 1
critical stock event low = detect_low_stock(events, threshold=10.0) if len(low) == 1:
print(f"? Test 2 PASSED: low stock events = {len(low)}") passed += 1 else: print(f"? Test 2
FAILED: low stock events = {len(low)} (expected 1)") failed += 1
# ?? Test 3: Anomaly rate for warehouse W1 ?????????????????????????? # W1 has 4 events,
1 anomaly ? 1/4 = 25% rate = calculate_anomaly_rate(events, "W1", 1) if abs(rate - 25.0) <
0.1: print(f"? Test 3 PASSED: W1 anomaly rate = {rate:.1f}%") passed += 1 else: print(f"?
Test 3 FAILED: W1 anomaly rate = {rate:.1f}% (expected 25.0%)") failed += 1
print() if failed == 0: print(f"? All 3 tests passed ? great debugging!") else: print(f"?
{failed} test(s) failed ? keep debugging!") return failed == 0
if __name__ == "__main__": print(f"? Running self-tests for broken_alerting.py ...\\n")
run_tests()

```

Supporting file: sensor_events.csv

```

timestamp,warehouse_id,warehouse_city,sensor_type,value,unit
2026-03-01T10:00:00Z,WH-NYC,New York,temperature,22.5,celsius
2026-03-01T10:00:30Z,WH-NYC,New York,humidity,55.0,percent 2026-03-01T10:01:00Z,WH-NYC,New
York,stock_level,42,units 2026-03-01T10:02:00Z,WH-NYC,New York,temperature,38.0,celsius
2026-03-01T10:03:00Z,WH-NYC,New York,stock_level,45,units 2026-03-01T10:04:00Z,WH-NYC,New
York,humidity,60.0,percent 2026-03-01T10:05:00Z,WH-NYC,New York,door_opens,5,count
2026-03-01T10:06:00Z,WH-NYC,New York,temperature,21.0,celsius
2026-03-01T10:07:00Z,WH-NYC,New York,stock_level,40,units 2026-03-01T10:08:00Z,WH-NYC,New
York,door_opens,3,count 2026-03-01T10:00:00Z,WH-LAX,Los Angeles,temperature,24.0,celsius
2026-03-01T10:01:00Z,WH-LAX,Los Angeles,humidity,45.0,percent
2026-03-01T10:02:00Z,WH-LAX,Los Angeles,stock_level,30,units
2026-03-01T10:03:00Z,WH-LAX,Los Angeles,temperature,25.5,celsius
2026-03-01T10:04:00Z,WH-LAX,Los Angeles,stock_level,8,units 2026-03-01T10:05:00Z,WH-LAX,Los
Angeles,humidity,50.0,percent 2026-03-01T10:06:00Z,WH-LAX,Los Angeles,door_opens,4,count
2026-03-01T10:07:00Z,WH-LAX,Los Angeles,temperature,23.0,celsius
2026-03-01T10:08:00Z,WH-LAX,Los Angeles,stock_level,3,units 2026-03-01T10:09:00Z,WH-LAX,Los
Angeles,door_opens,6,count 2026-03-01T10:00:00Z,WH-CHI,Chicago,temperature,20.0,celsius
2026-03-01T10:01:00Z,WH-CHI,Chicago,humidity,65.0,percent
2026-03-01T10:02:00Z,WH-CHI,Chicago,stock_level,55,units
2026-03-01T10:03:00Z,WH-CHI,Chicago,temperature,19.5,celsius
2026-03-01T10:04:00Z,WH-CHI,Chicago,stock_level,60,units
2026-03-01T10:05:00Z,WH-CHI,Chicago,humidity,85.0,percent

```

2026-03-01T10:06:00Z,WH-CHI,Chicago,door_opens,3,count
2026-03-01T10:07:00Z,WH-CHI,Chicago,temperature,21.0,celsius
2026-03-01T10:08:00Z,WH-CHI,Chicago,stock_level,52,units
2026-03-01T10:09:00Z,WH-CHI,Chicago,door_opens,2,count
2026-03-01T10:00:00Z,WH-DAL,Dallas,temperature,28.0,celsius
2026-03-01T10:01:00Z,WH-DAL,Dallas,humidity,40.0,percent
2026-03-01T10:02:00Z,WH-DAL,Dallas,stock_level,38,units
2026-03-01T10:03:00Z,WH-DAL,Dallas,temperature,32.0,celsius
2026-03-01T10:04:00Z,WH-DAL,Dallas,stock_level,35,units
2026-03-01T10:05:00Z,WH-DAL,Dallas,humidity,38.0,percent
2026-03-01T10:06:00Z,WH-DAL,Dallas,door_opens,8,count
2026-03-01T10:07:00Z,WH-DAL,Dallas,temperature,35.0,celsius
2026-03-01T10:08:00Z,WH-DAL,Dallas,stock_level,40,units
2026-03-01T10:09:00Z,WH-DAL,Dallas,door_opens,6,count
2026-03-01T10:00:00Z,WH-SEA,Seattle,temperature,18.0,celsius
2026-03-01T10:01:00Z,WH-SEA,Seattle,humidity,70.0,percent
2026-03-01T10:02:00Z,WH-SEA,Seattle,stock_level,30,units
2026-03-01T10:03:00Z,WH-SEA,Seattle,temperature,17.5,celsius
2026-03-01T10:04:00Z,WH-SEA,Seattle,stock_level,28,units
2026-03-01T10:05:00Z,WH-SEA,Seattle,humidity,75.0,percent
2026-03-01T10:06:00Z,WH-SEA,Seattle,door_opens,3,count
2026-03-01T10:07:00Z,WH-SEA,Seattle,temperature,19.0,celsius
2026-03-01T10:08:00Z,WH-SEA,Seattle,stock_level,25,units
2026-03-01T10:09:00Z,WH-SEA,Seattle,door_opens,1,count