| **Title** | From Clicks to Deliveries: Maximizing E-commerce Performance with Real-Time Data Integration |
| --- | --- |
| **Technologies** | AWS ( Lambda, Kinesis , DynamoDB , API Gateway) , Snowflake , Python |

### 

### **Problem Statement:**

* As an e-commerce company, our success hinges on seamlessly integrating our online platform with efficient logistics management to ensure optimal customer satisfaction and operational efficiency. To achieve this synergy, we aim to leverage real-time data streams from both our website and fleet of delivery trucks.

**Online Platform Optimization**:

* We need to analyze clickstream data to understand customer preferences, enhance user experience, and optimize marketing strategies for key product categories such as mobile phones, laptops, and cameras.
* For 3 items following data is collecting in real time
  + Item ID
  + Item Name
  + Click Count

**Fleet Management and Logistics Optimization:**

* Simultaneously, we must monitor and analyze real-time telemetry data from our fleet of delivery trucks, utilizing IoT sensors installed in each vehicle. This data will enable us to optimize routes, reduce fuel consumption, proactively address maintenance issues, and ensure the safety and reliability of our delivery operations.
* For 3 truck in the fleet, the following data will be collected in near real-time(the trucks are sending the data every 1 minute):
  + **Truck ID** : should have 3 truck id’s
  + **GPS Location**: Latitude, Longitude, Altitude, Speed.
  + **Vehicle Speed**: Real-time speed of the vehicle.
  + **Engine Diagnostics**: Engine RPM, Fuel Level, Temperature, Oil Pressure, Battery Voltage.
  + **Odometer Reading**: Total distance traveled.
  + **Fuel Consumptio**n: Fuel usage over time.
  + **Vehicle Health and Maintenance**: Brake status, Tire pressure, Transmission status.
  + **Environmental Conditions:** Temperature, Humidity, Atmospheric Pressure.
* Create a API with the above parameters and use the same api to fetch the data for downstream lambda function

**Data Processors**:

* **AWS Kinesis**: Real-time data streaming and processing for both clickstream data and truck IoT sensor data.
* **AWS Lambda**: Further analysis and computation of insights for both data streams.

### **Data Destinations:**

* **Snowflake or DynamoDB**: Store the processed clickstream data and truck IoT sensor data for historical analysis, reporting, and integration with other business systems.
* For the truck data the schema should follow Type 2 SCD(Slowly changing dimensions) , we need historical data of the truck every time it send the data

Note: All data should generator randomly you can use Python script or other ways to generate data

Prototype of Truck API :

**{**

**"trucks": [**

**{**

**"truck\_id": "TRK001",**

**"gps\_location": {**

**"latitude": 34.052235,**

**"longitude": -118.243683,**

**"altitude": 89.0,**

**"speed": 65.0**

**},**

**"vehicle\_speed": 65.0,**

**"engine\_diagnostics": {**

**"engine\_rpm": 2500,**

**"fuel\_level": 75.0,**

**"temperature": 90.0,**

**"oil\_pressure": 40.0,**

**"battery\_voltage": 13.8**

**},**

**"odometer\_reading": 102345.6,**

**"fuel\_consumption": 15.5,**

**"vehicle\_health\_and\_maintenance": {**

**"brake\_status": "Good",**

**"tire\_pressure": {**

**"front\_left": 32.0,**

**"front\_right": 32.0,**

**"rear\_left": 35.0,**

**"rear\_right": 35.0**

**},**

**"transmission\_status": "Operational"**

**},**

**"environmental\_conditions": {**

**"temperature": 22.0,**

**"humidity": 50.0,**

**"atmospheric\_pressure": 1013.25**

**}**

**},**

**{**

**"truck\_id": "TRK002",**

**"gps\_location": {**

**"latitude": 36.169941,**

**"longitude": -115.139832,**

**"altitude": 610.0,**

**"speed": 55.0**

**},**

**"vehicle\_speed": 55.0,**

**"engine\_diagnostics": {**

**"engine\_rpm": 2200,**

**"fuel\_level": 60.0,**

**"temperature": 88.0,**

**"oil\_pressure": 38.0,**

**"battery\_voltage": 13.7**

**},**

**"odometer\_reading": 88342.1,**

**"fuel\_consumption": 14.2,**

**"vehicle\_health\_and\_maintenance": {**

**"brake\_status": "Needs Inspection",**

**"tire\_pressure": {**

**"front\_left": 31.0,**

**"front\_right": 31.0,**

**"rear\_left": 34.0,**

**"rear\_right": 34.0**

**},**

**"transmission\_status": "Operational"**

**},**

**"environmental\_conditions": {**

**"temperature": 30.0,**

**"humidity": 20.0,**

**"atmospheric\_pressure": 1008.00**

**}**

**},**

**{**

**"truck\_id": "TRK003",**

**"gps\_location": {**

**"latitude": 40.712776,**

**"longitude": -74.005974,**

**"altitude": 10.0,**

**"speed": 45.0**

**},**

**"vehicle\_speed": 45.0,**

**"engine\_diagnostics": {**

**"engine\_rpm": 2000,**

**"fuel\_level": 85.0,**

**"temperature": 85.0,**

**"oil\_pressure": 42.0,**

**"battery\_voltage": 14.0**

**},**

**"odometer\_reading": 120567.9,**

**"fuel\_consumption": 16.0,**

**"vehicle\_health\_and\_maintenance": {**

**"brake\_status": "Good",**

**"tire\_pressure": {**

**"front\_left": 33.0,**

**"front\_right": 33.0,**

**"rear\_left": 36.0,**

**"rear\_right": 36.0**

**},**

**"transmission\_status": "Operational"**

**},**

**"environmental\_conditions": {**

**"temperature": 18.0,**

**"humidity": 65.0,**

**"atmospheric\_pressure": 1012.00**

**}**

**}**

**]**

**}**

Schema for the Database:

* Same as API response with SCD(Slowly Changing Dimension) 2