DOKUMENTACJA

do rozwiązania dot. przetwarzania danych wejść i wyjść pracowników

Dla FIRMA KURIERSKA Sp. z o.o.

WERSJA	1.0.
DATA	26.01.2025
AUTOR	Mikołaj Kuna

Spis treści

1. Opis ogólny

Niniejszy dokument ma na celu opisanie działania rozwiązania opartego na danych dot. wejść i wyjść pracowników (kurierów) dla przedsiębiorstwa z branży logistyki. Firma Kurierska Sp. z o.o. zajmuje się dostarczaniem przesyłek na terenie dużych miast w Polsce. Klient zgłosił potrzebę:

- przetwarzania danych z czytników znajdujących się w poszczególnych centrach logistycznych na terenie Polski za pomocą rozwiązania chmurowego
- możliwości raportowania danych z wejść i wyjść pracowników w poszczególnych oddziałach (tzw. ‘odbicia’)

2. Dane

Dostarczane przez klienta dane dla zamodelowania zaproponowanego rozwiązania pochodzą z lat ubiegłych. Zostały zarejestrowane przez czytnik nr 1 w centrum logistycznym w Warszawie dla wyselekcjonowanej grupy pracowników-kurierów. Każdy rekord w pliku JSON to pojedyncze wejście lub wyjście pracownika i zawiera takie informacje jak: numer pracownika w systemie, numer karty do rozliczenia czasu pracy, datę wejścia/wyjścia, oznaczenie wejście (wartość =1) lub wyjście (wartość = 2) oraz numer czytnika. Na zbiorze dostarczonym przez klienta wykonano preprocessing oraz batch processig.

Zasadą biznesową wynikającą z regulaminu jest konieczność wejścia i wyjścia pracownika-kuriera do/z przypisanego mu centrum logistycznego w obrębie jednej doby (24h) z powodu dziennego rozliczenia pojazdów dostawczych oraz poprawnego rozliczenia czasu pracy.

Schemat danych:

PRAC_ID	Klucz pracownika
KARTA_OKZ	Numer karty rejestracji czasu pracy
DATA_CZAS	Data wejścia lub wyjścia w formacie DD-MON-YYYY
STATUS_ID	Nr wejścia oznacza odpowiednio: 1. wejście | 2. wyjście
NR_CZYTNIKA	Nr czytnika

3. Architektura rozwiązania

Do budowy rozwiązania odpowiadającego na potrzeby klienta wykorzystano kompleksowe narzędzia takie jak AWS S3, AWS Glue i PySpark, aby zmaksymalizować efektywność operacji na danych oraz umożliwić skalowalność rozwiązań potencjalnie dla wszystkich centrów logistycznych Przedsiębiorstwa Kurierskiego Sp. z o.o. na terenie Polski.

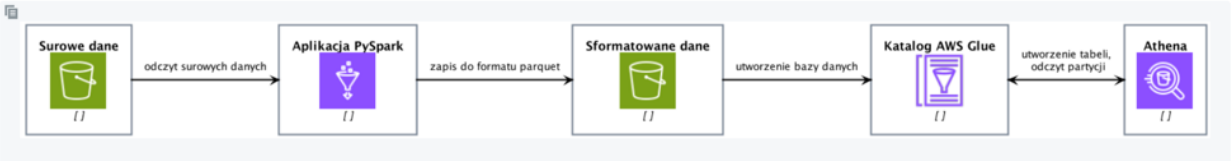

Diagram 1. Zaimplementowana architektura rozwiązania służąca do przygotowania danych

3.1. Preprocessing

Zgodnie z Diagramem 1. i przy wykorzystaniu wymienionych wyżej narzędzi dokonano preprocessingu, w wyniku czego otrzymano gotowe do dalszej obróbki dane:

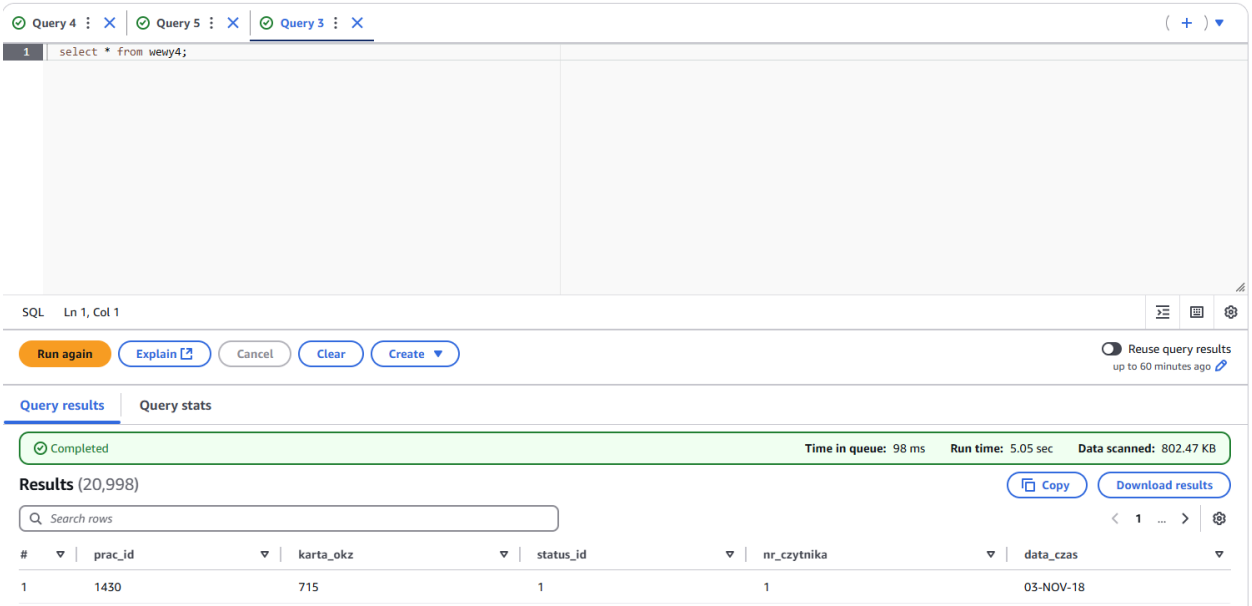

Tabela 1. Wynik zapytania SQL dla zaimplementowanych danych

Korzystając z możliwości, które daje narzędzie Amazon Athena wygenerowano statystyki dla dostarczonych przez klienta danych:

Column statistics (4) Info

Get an overview of the data profile. We estimate the approximate number of distinct values in a data set with 5% average relative error.

Find columns

Column name	Last updated...	Distinct values	Null values
karta_okz	January 26, 2025 at 1	44	0
nr_czytnika	January 26, 2025 at 1	1	0
prac_id	January 26, 2025 at 1	54	0
status_id	January 26, 2025 at 1	2	0

Tabela 2. Statystyki danych wsadowych w Amazon Athena

3.2. Batch Processing

Podjęte działania w zakresie batch processingu, czyli wykonywania zadań, w którym dane są wczytywane do pamięci a następnie kolejno wykonywane, pozwoliły na spełnienie wymagania klienta odnośnie do potrzeby generowania raportów opartego na kompletności par wejść i wyjść pracowników.

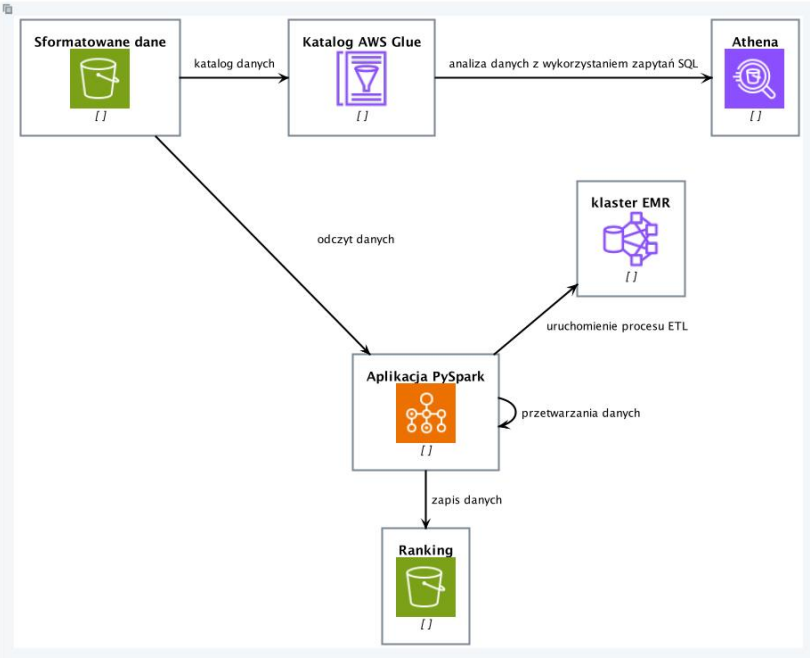

Diagram 2. Rozwiązanie z wykorzystaniem narzędzi AWS służące do generowania raportów

Celem procesu ETL jest wygenerowanie zestawienia pracowników, którzy w danym dniu nie dokonali obowiązkowego odbicia wejścia i wyjścia (brak pary 1-2 dla kolumny status_id):

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, to_date, date_format, min, max
from pyspark.sql.window import Window

def main():
    # Tworzymy sesję Spark
    with SparkSession.builder.appName("MyApp").getOrCreate() as spark:
        # Wczytujemy dane z pliku Parquet
        df = spark.read.parquet("s3://668704778226-formatted-data")

        # Preprocessing: Usuwamy wiersze, w których 'status_id' jest NULL
        df = df.filter(df.status_id.isNotNull())

       # Grupujemy po prac_id i dacie, sprawdzamy minimalny i maksymalny status_id
        grouped = df.groupBy("prac_id", "data_czas").agg(
            min("status_id").alias("min_status"),
            max("status_id").alias("max_status")
        )

        # Sprawdzamy, którzy pracownicy nie mieli pełnej pary (wejście 1, wyjście 2)
        brak_pary = grouped.filter((col("min_status") != 1) | (col("max_status") != 2))

        # Wyświetlamy wynik
        brak_pary.show()
        brak_pary.coalesce(1).write.json("s3://668704778226-result/results.json", mode='Overwrite')

if __name__ == "__main__":
    main()
```

Skrypt 1. Zaimplementowany kod będący podstawą do dalszego procesu przy pomocy Cloud9

Wygenerowane dane w formie pliku json stanowią załącznik do niniejszej dokumentacji. Zidentyfikowano 1026 przypadków braku par wejścia-wyjścia w obrębie doby rozliczeniowej:

part-00000-1634e454
-7187-442a-b122-03b

4. Podsumowanie

Implementacja naszego rozwiązania na platformie AWS pozwala na łatwą integrację z innymi usługami chmurowymi, a także zapewnia bezpieczeństwo i elastyczność przechowywania danych. Dzięki użyciu S3 do przechowywania danych wejściowych i wyników analizy, a także wykorzystaniu EC2 lub innych instancji obliczeniowych, rozwiązanie jest zarówno wydajne, jak i oszczędne pod względem zasobów.

Jesteśmy przekonani, że to rozwiązanie może znacząco usprawnić procesy logistyczne i kontrolne w centrach dystrybucyjnych Firmy Kurierskiej Sp. z o.o. Z niecierpliwością czekamy na dalszą współpracę z klientem przy implementacji tego rozwiązania w centrach logistycznych na terenie całego kraju, aby zapewnić efektywność i transparentność operacji na szeroką skalę.