

DOKUMENTACJA

**do rozwiązania dot. przetwarzania
danych wejść i wyjść pracowników**

Dla FIRMA KURIERSKA Sp. z o.o.

WERSJA	2.0.
DATA	26.01.2025
AUTOR	Mikołaj Kuna

Spis treści

1.	Opis ogólny	3
2.	Dane.....	3
3.	Architektura rozwiązania.....	4
3.1.	Preprocessing	4
3.2.	Batch-Processing.....	5
3.3.	Stream-Processing	6
4.	Podsumowanie	8

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'),
- strumieniowe przetwarzanie danych z czytników i wykrywanie brakujących par wejście-wyjście dla poszczególnych pracowników w ramach doby.

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, batch-processig oraz stream-processing.

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, PySpark, Amazon Kinesis i Apache Flink, aby zmaksymalizować efektywność operacji na danych oraz umożliwić skalowalność rozwiązań potencjalnie dla wszystkich centrów logistycznych Przedsiębiorstwa Kurierskiego Sp. z o.o. na terenie Polski.

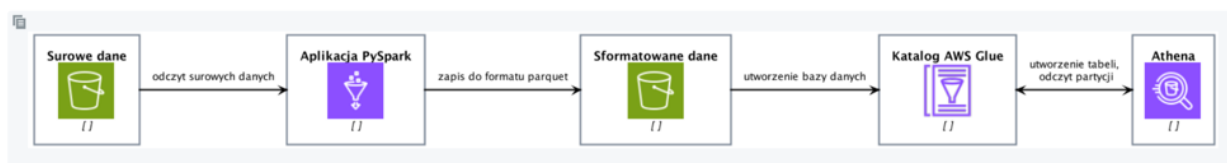


Diagram 1. Zaimplementowana architektura rozwiązania służąca do przygotowania danych

3.1. Preprocessing

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

The screenshot shows the AWS Athena console interface. At the top, there are tabs for 'Query 4', 'Query 5', and 'Query 3'. The active query is 'Query 4' with the SQL statement: `select * from wewy4;`. Below the query editor, there are buttons for 'Run again', 'Explain', 'Cancel', 'Clear', and 'Create'. To the right, there is a toggle for 'Reuse query results up to 60 minutes ago'. The 'Query results' tab is selected, showing a green status bar indicating 'Completed'. Below this, the query execution details are displayed: 'Time in queue: 98 ms', 'Run time: 5.05 sec', and 'Data scanned: 802.47 KB'. The results are shown as a table with 20,998 rows. The first row is visible, showing columns: '#', 'prac_id', 'karta_okz', 'status_id', 'nr_czytnika', and 'data_czas'. The values for the first row are: 1, 1430, 715, 1, 1, and 03-NOV-18.

#	prac_id	karta_okz	status_id	nr_czytnika	data_czas
1	1430	715	1	1	03-NOV-18

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.

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.

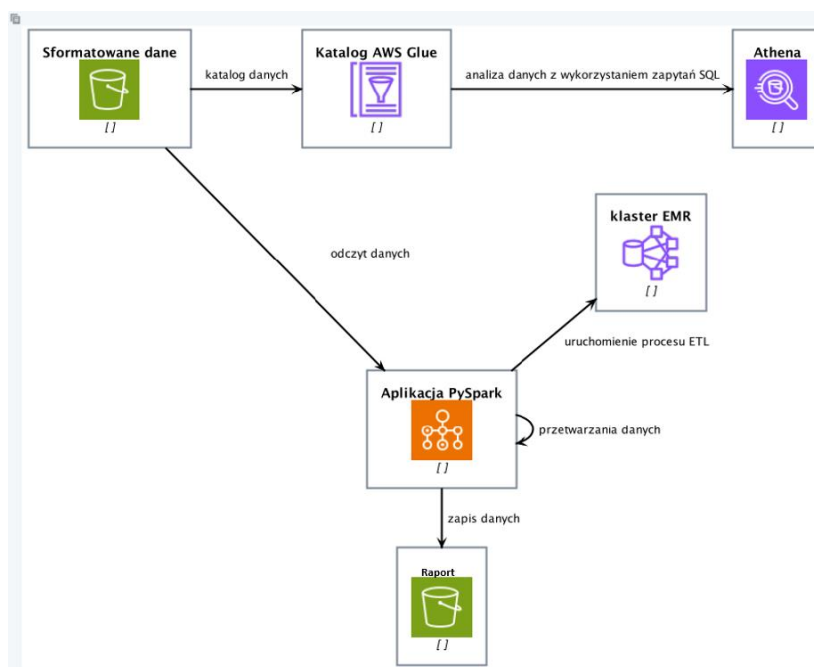


Diagram 2. Rozwiązanie z wykorzystaniem narzędzi AWS służące do generowania raportów

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):

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql.functions import col, to_date, date_format, min, max
3 from pyspark.sql.window import Window
4
5 def main():
6     # Tworzymy sesję Spark
7     with SparkSession.builder.appName("MyApp").getOrCreate() as spark:
8         # Wczytujemy dane z pliku Parquet
9         df = spark.read.parquet("s3://668704778226-formatted-data")
10
11        # Preprocessing: Usuwamy wiersze, w których 'status_id' jest NULL
12        df = df.filter(df.status_id.isNotNull())
13
14        # Grupujemy po prac_id i dacie, sprawdzamy minimalny i maksymalny status_id
15        grouped = df.groupBy("prac_id", "data_czas").agg(
16            min("status_id").alias("min_status"),
17            max("status_id").alias("max_status")
18        )
19
20        # Sprawdzamy, którzy pracownicy nie mieli pełnej pary (wejście 1, wyjście 2)
21        brak_pary = grouped.filter((col("min_status") != 1) | (col("max_status") != 2))
22
23        # Wyświetlamy wynik
24        brak_pary.show()
25        brak_pary.coalesce(1).write.json("s3://668704778226-result/results.json", mode='Overwrite')
26
27 if __name__ == "__main__":
28     main()
29
```

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

3.3. Stream-Processing

To rozwiązanie zaimplementowane w Firmie Kurierskiej Sp. z o.o. służy przetwarzaniu danych z czytników, które są analizowane i przetwarzane na bieżąco, w miarę ich napływania. Zamiast przechowywania danych i późniejszego ich analizowania, stream-processing umożliwia natychmiastowe reagowanie na wydarzenia, co pozwala na szybsze podejmowanie decyzji i wykrywanie nieprawidłowości, takich jak np. brak obecności lub brak wyjścia ze zmiany pracowników w centrum logistycznym. Każdy wpis zawiera unikalny identyfikator pracownika, czas wejścia i wyjścia, a także status operacji (wejście lub wyjście).



Diagram 3. Przepływ danych z wykorzystaniem narzędzia Amazon Kinesis służące do weryfikowania odbić pracowników klienta

```

lambda_function.py x
lambda_function.py
1  import boto3
2  import json
3  import datetime
4  import logging
5
6  logger = logging.getLogger()
7  logger.setLevel("INFO")
8
9  stream_name = 'brakpary'
10 bucket = '668704778226-landing-zone'
11 key = 'REJ_WEWY_PW_DATA_TABLE4.json'
12
13 s3_client = boto3.resource('s3')
14 kinesis_client = boto3.client('kinesis', region_name='us-east-1')
15
16
17 def lambda_handler(event, context):
18     logger.info("Lambda invoked")
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

PROBLEMS OUTPUT CODE REFERENCE LOG TERMINAL Execution Results

```

'wviscie': '18-APR-26', 'event time': '2025-01-27T12:41:48.344333'}

```

Skrypt 2. Zaimplementowana funkcja będąca podstawą do dalszego procesu przy pomocy Amazon Kinesis

22-JUL-16	{"prac_id": 1431, "data_czas": "22-JUL-16", "wejście": "22-JUL..."}	January 27, 2025 at 13:41:21 GMT+1
28-JUL-16	{"prac_id": 1431, "data_czas": "28-JUL-16", "wejście": "28-JUL..."}	January 27, 2025 at 13:41:21 GMT+1
28-SEP-16	{"prac_id": 1431, "data_czas": "28-SEP-16", "wejście": "28-SEP..."}	January 27, 2025 at 13:41:21 GMT+1

Tabela 3. Wygenerowane rekordy ze strumienia 'brakpary' w ramach narzędzia Amazon Kinesis

The screenshot shows the Amazon Kinesis Data Streams console for a stream named 'brakpary'. The 'Data viewer' tab is selected, showing a list of records. The records are JSON objects containing employee ID ('prac_id'), date ('data_czas'), and entry type ('wejście'). The table also includes columns for 'Approximate arrival timestamp' and 'Sequence number'.

Partition key	Data	Approximate arrival timestamp	Sequence number
08-JAN-10	{"prac_id": 1073, "data_czas": "08-JAN-10", "wejście": "08-JA..."}	January 27, 2025 at 13:40:22 GMT+1	4965998859607119926317020424169684671220735882559225890
08-JAN-10	{"prac_id": 1112, "data_czas": "08-JAN-10", "wejście": "08-JA..."}	January 27, 2025 at 13:40:22 GMT+1	49659988596071199263170204241698055638040350511733932066
08-JAN-10	{"prac_id": 1042, "data_czas": "08-JAN-10", "wejście": "08-JA..."}	January 27, 2025 at 13:40:22 GMT+1	49659988596071199263170204241699264563859965140908638242
08-JAN-10	{"prac_id": 1022, "data_czas": "08-JAN-10", "wejście": "08-JA..."}	January 27, 2025 at 13:40:22 GMT+1	49659988596071199263170204241700473489679579770083344418
08-JAN-10	{"prac_id": 1122, "data_czas": "08-JAN-10", "wejście": "08-JA..."}	January 27, 2025 at 13:40:22 GMT+1	49659988596071199263170204241701682415499194399258050594
09-JAN-10	{"prac_id": 1012, "data_czas": "09-JAN-10", "wejście": "09-JA..."}	January 27, 2025 at 13:40:22 GMT+1	49659988596071199263170204241702891341318809028432756770

Tabela 4. Wygenerowane strumieniowo informacje, zawierające wskazania pracowników (prac_id), którzy w danym dniu nie dokonali pary odbić, pochodzący z serwisu Apache Flink

Zastosowanie narzędzia Flink przedstawione w Tabeli 4. działa na zasadzie przetwarzania strumienia danych, gdzie każde wejście i wyjście pracownika jest rejestrowane jako oddzielny rekord. System sprawdza, czy dla danego pracownika w danym dniu występuje pełna para – zarówno wejście, jak i wyjście. Jeśli brak jest któregoś z tych zdarzeń, system generuje informacje, komunikując problem z rejestracją obecności pracownika.

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.