Testy Funkcjonalne

Rynek Nieruchomości w Wielkiej Brytanii

Wojciech Kosiuk
Politechnika Warszawska

Szymon Matuszewski
Politechnika Warszawska

Michał Mazuryk
Politechnika Warszawska

Listopad 2023

Spis treści

1 Test struktur Apache HBase

Cel: Test polega na sprawdzeniu czy założone struktury w Apache HBase są prawidłowe i posiadają odpowiednie rodziny kolumn.

Kroki:

- Uruchomienie HBase shell
- Wywołanie komendy 'describe' dla nazw tabel
- Weryfikacja z założeniami

```
hbase(main):024:0> describe 'price_data'
Table price_data is ENABLED
price_data
COLUMN FAMILIES DESCRIPTION
{NAME => 'location', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1'
, KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => '
NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE =>
 '65536', REPLICATION_SCOPE => '0'}

{NAME => 'property_details', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSION
S => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESS
ION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOC
KSIZE => '65536', REPLICATION_SCOPE => '0'}

2 row(s)
Quota is disabled
Took 0.0271 seconds
```

Rysunek 1: Komenda 'describe' dla tabeli price_data.

```
hbase(main):023:0> describe 'property_data'
Table property_data is ENABLED
property_data
COLUMN FAMILIES DESCRIPTION
{NAME => 'dates', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', K
EEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NON
E', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '6
5536', REPLICATION_SCOPE => '0'}

{NAME => 'location', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1'
, KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => '
NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE =>
 '65536', REPLICATION_SCOPE => '0'}

{NAME => 'property_details', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSION
S => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESS
ION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOC
KSIZE => '65536', REPLICATION_SCOPE => '0'}

3 row(s)
Quota is disabled
Took 0.0741 seconds
```

Rysunek 2: Komenda 'describe' dla tabeli price_data.

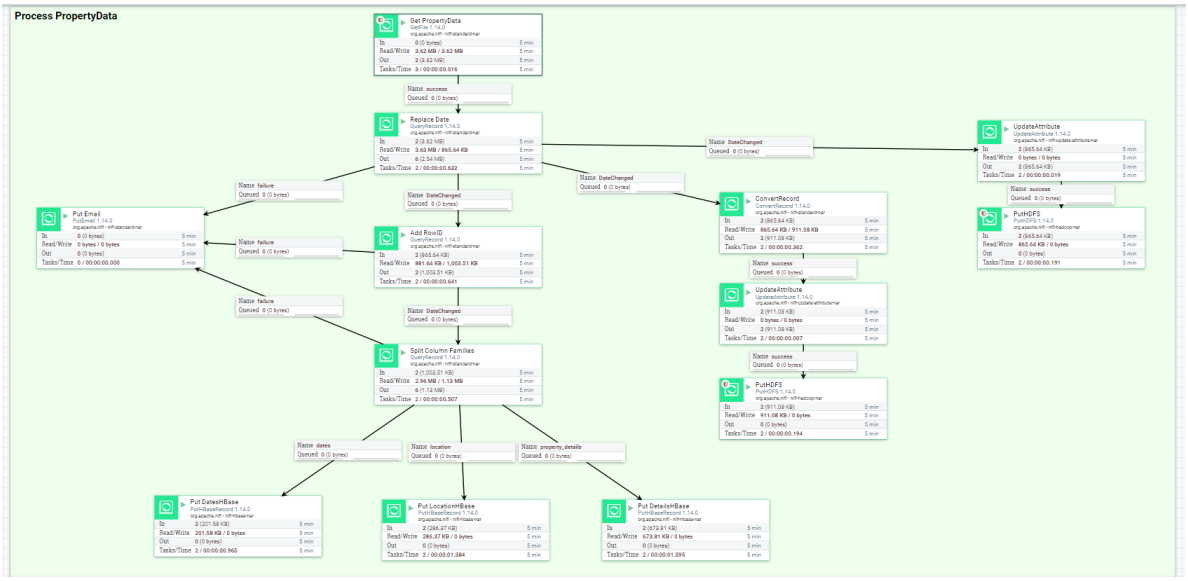
Wynik: Zgadza się z założeniami, otrzymano rodziny 'location', 'property_details' dla 'price_data' oraz 'location', 'property_details', 'dates' dla 'property_data'.

2 Test przetworzenia nowych kwartałów

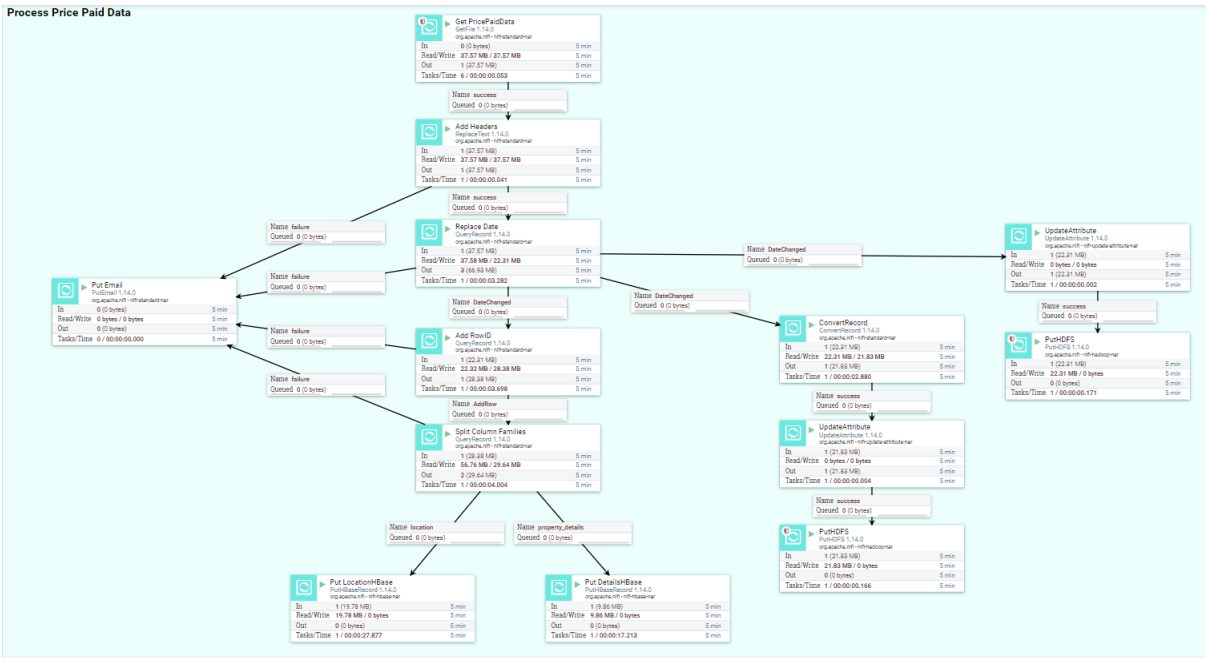
Cel: Test polega na dodaniu danych dotyczącego nowych kwartałów i zweryfikowania ich poprawnego przetworzenia przez proces.

Kroki:

- Dodanie do folderów wejściowych danych dotyczących nowych kwartałów (3 i 4 z roku 2018)
- Uruchomienie procesu w Apache NiFi
- Weryfikacja jego przejścia bez błędów
- Wybranie konkretnego wiersza dla wrzuconych nowych plików LDD i PP
- Wywołanie komendy 'get' w Apache HBase szukając odpowiedniego wiersza
- Sprawdzenie zapisu nowych plików w Apache Hadoop



Rysunek 3: Poprawnie przetworzony proces dla Property Data.

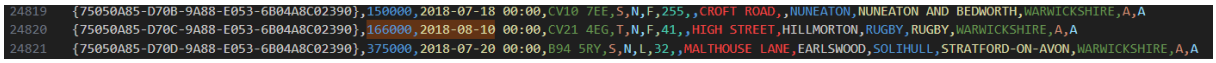


Rysunek 4: Poprawnie przetworzony proces dla Price Paid Data.

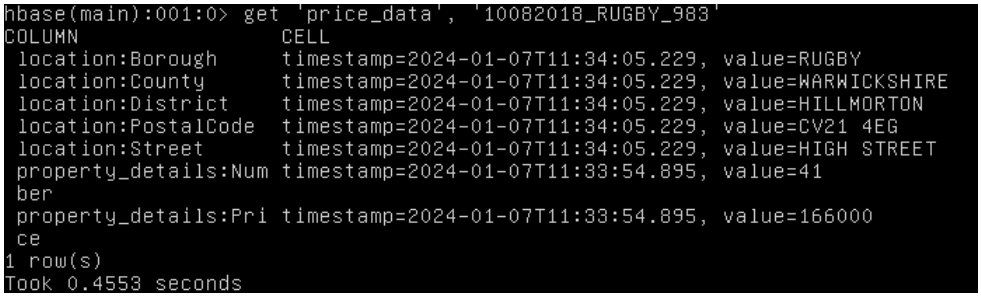
	Borough	Permission Status	No of Bedrooms	Unit Type	Residential Development Type	Primary Street Name	Post Code	Superseded date	Permission Date	Started Date	Completed date	Building height (maximum storeys)
0	Barking and Dagenham	Submitted	2.0	Flat Apartment or Maisonette	Extension to building for residential unit(s)	Becontree Avenue	RM8 3UH	NaN	13/08/2018	NaN	NaN	2.0
1	Barking and Dagenham	Started	2.0	Flat Apartment or Maisonette	New Residential Building	Rainham Road South	RM10 7XJ	NaN	09/07/2018	16/09/2019	NaN	2.0
2	Barking and Dagenham	Submitted	1.0	Flat Apartment or Maisonette	New Residential Building	Merrielands Crescent	RM9	NaN	22/08/2018	NaN	NaN	10.0
3	Barking and Dagenham	Submitted	1.0	Flat Apartment or Maisonette	New Residential Building	Merrielands Crescent	RM9	NaN	22/08/2018	NaN	NaN	10.0
4	Barking and Dagenham	Submitted	1.0	Flat Apartment or Maisonette	New Residential Building	Merrielands Crescent	RM9	NaN	22/08/2018	NaN	NaN	10.0

Rysunek 5: Fragment ramki z wybieranymi przez nas kolumnami z pliku LDD dotyczącego 3 kwartału 2018 roku. Sprawdzanym wierszem w HBase będzie ten o numerze 0.

Rysunek 6: Log z konsoli HBase w tabeli 'property_data' dotyczący wybranego wcześniej wiersza.



Rysunek 7: Fragment ramki z pliku PP dotyczącego 3 kwartału 2018 roku, sprawdzanym wierszem w HBase będzie ten o numerze 24820.



Rysunek 8: Log z konsoli HBase w tabeli 'price_data' dotyczący wybranego wcześniej wiersza.

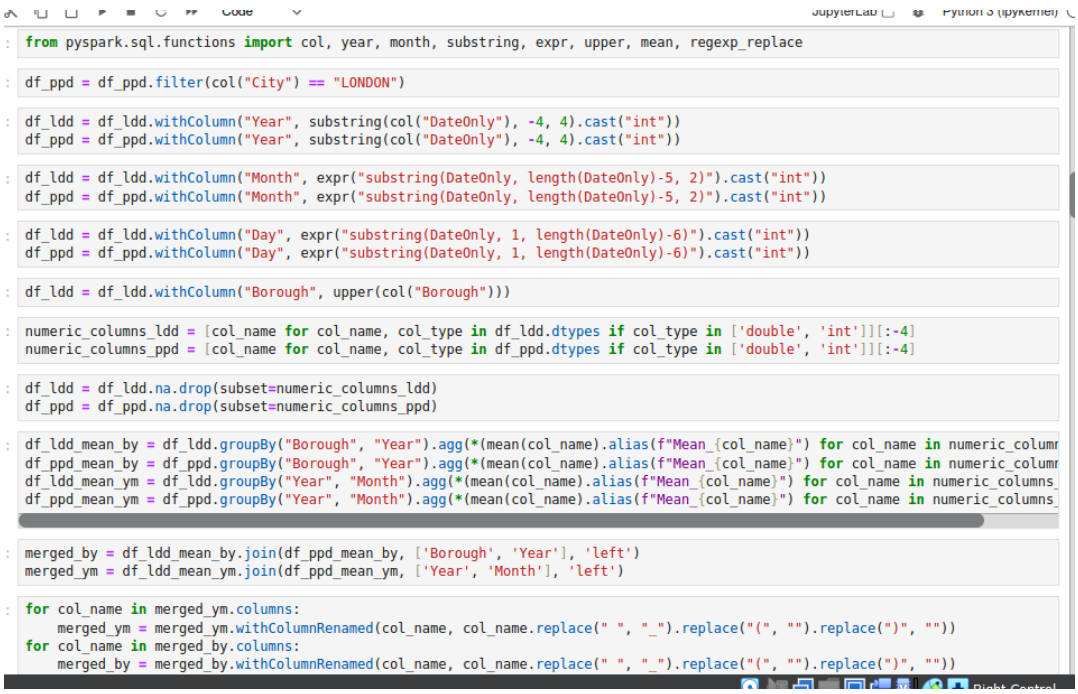
Wynik: Nowe kwartały zostały poprawnie przetworzone w NiFi. Przetworzone dane zostały dodane do HBase i HDFS. Wiersze zostały poprawnie zapisane w HBase, według struktury jaką oczekiwaliśmy.

3 Test przetwarzania Apache Spark

Cel: Aby sprawdzić poprawność przetwarzania danych w Apache Spark stworzyliśmy widoki danych, które następnie zapisaliśmy w systemie plików HDFS.

Kroki:

- Zagregowanie danych do odpowiednich tabel oraz ich złączenie.
- Sprawdzenie 20 wierszy z każdej tabeli (czy istnieją).
- Zapisanie widoku wsadowego w systemi HDFS.
- Sprawdzenie czy zapis się odbył



```
from pyspark.sql.functions import col, year, month, substring, expr, upper, mean, regexp_replace

df_ppd = df_ppd.filter(col("City") == "LONDON")

df_ldd = df_ldd.withColumn("Year", substring(col("DateOnly"), -4, 4).cast("int"))
df_ppd = df_ppd.withColumn("Year", substring(col("DateOnly"), -4, 4).cast("int"))

df_ldd = df_ldd.withColumn("Month", expr("substring(DateOnly, length(DateOnly)-5, 2)").cast("int"))
df_ppd = df_ppd.withColumn("Month", expr("substring(DateOnly, length(DateOnly)-5, 2)").cast("int"))

df_ldd = df_ldd.withColumn("Day", expr("substring(DateOnly, 1, length(DateOnly)-6)").cast("int"))
df_ppd = df_ppd.withColumn("Day", expr("substring(DateOnly, 1, length(DateOnly)-6)").cast("int"))

df_ldd = df_ldd.withColumn("Borough", upper(col("Borough")))

numeric_columns_ldd = [col_name for col_name, col_type in df_ldd.dtypes if col_type in ['double', 'int']][:-4]
numeric_columns_ppd = [col_name for col_name, col_type in df_ppd.dtypes if col_type in ['double', 'int']][:-4]

df_ldd = df_ldd.na.drop(subset=numeric_columns_ldd)
df_ppd = df_ppd.na.drop(subset=numeric_columns_ppd)

df_ldd_mean_by = df_ldd.groupBy("Borough", "Year").agg(*(mean(col_name).alias(f"Mean_{col_name}") for col_name in numeric_columr
df_ppd_mean_by = df_ppd.groupBy("Borough", "Year").agg(*(mean(col_name).alias(f"Mean_{col_name}") for col_name in numeric_columr
df_ldd_mean_ym = df_ldd.groupBy("Year", "Month").agg(*(mean(col_name).alias(f"Mean_{col_name}") for col_name in numeric_columns_
df_ppd_mean_ym = df_ppd.groupBy("Year", "Month").agg(*(mean(col_name).alias(f"Mean_{col_name}") for col_name in numeric_columns_

merged_by = df_ldd_mean_by.join(df_ppd_mean_by, ['Borough', 'Year'], 'left')
merged_ym = df_ldd_mean_ym.join(df_ppd_mean_ym, ['Year', 'Month'], 'left')

for col_name in merged_ym.columns:
    merged_ym = merged_ym.withColumnRenamed(col_name, col_name.replace(" ", "_").replace("(", "").replace(")", ""))
for col_name in merged_by.columns:
    merged_by = merged_by.withColumnRenamed(col_name, col_name.replace(" ", "_").replace("(", "").replace(")", ""))
```

Rysunek 9: Zrzut ekranu prezentujący kolejne etapy procesowania danych w Apache Spark w celu utworzenia zagregowanych tabel widokowych.

```
merged_ym.show()
```

```
[Stage 244:=================================================> (49 + 1) / 50]
+----+-----+------------------+---------------------------------+-----------------+
|Year|Month|Mean_No_of_Bedrooms|Mean_Building_height_maximum_storeys|       Mean_Price|
+----+-----+------------------+---------------------------------+-----------------+
|2015|    2| 2.2650602409638556|                26.25602409638554| 729797.3082631368|
|2017|    3|  2.264646464646465|                8.424242424242424| 953463.0840964591|
|2017|    8| 2.1725888324873095|                7.535532994923858| 864724.8906580016|
|2014|    4|  2.209621993127148|               12.958762886597938|  710839.173211237|
|2017|   10|  2.155313351498638|                7.021798365122616| 964845.0763555251|
|2018|   10|  2.175480769230769|                6.211538461538462| 898489.1211807354|
|2015|   12| 2.1358885017421603|                11.80836236933798| 928461.7741384221|
|2016|    7|  2.235135135135135|                6.094594594594595| 764402.6407185629|
|2016|   11| 2.2117903930131004|                5.762008733624454| 1028912.169087137|
|2018|    1|  2.232189973614776|                5.071240105540897|1121058.1456367925|
|2018|    3| 2.1214470284237725|                 9.21188630490956| 950243.6053100498|
|2014|   10|          2.3359375|                        7.234375|  740799.172595719|
|2016|    5| 2.1991434689507496|                9.222698072805139| 834914.1343082115|
|2014|   12|  2.300324675324675|                12.39935064935065| 774059.3787325813|
|2018|    8|  2.164009111617312|                14.98861047835991| 921170.3014925373|
|2014|    5| 2.3353293413173652|                 8.95808383233533| 715598.3746264196|
|2016|    2| 2.2880794701986753|                5.447019867549669|    730683.1640625|
|2017|    7| 2.1950549450549453|                4.862637362637362| 1083425.616948601|
|2014|    1| 2.2434210526315788|               10.355263157894736| 677154.1069452981|
|2014|    8|  2.340175953079179|               10.222873900293255| 747442.2702055759|
+----+-----+------------------+---------------------------------+-----------------+
only showing top 20 rows
```

Rysunek 10: Zrzut ekranu ze sprawdzeniem czy tabela 1 poprawnie się utworzyła.

```
merged_by.show()
```

```
[Stage 245:=========================================>              (3 + 1) / 4]
+--------------------+----+------------------+---------------------------------+-----------------+
|             Borough|Year|Mean_No_of_Bedrooms|Mean_Building_height_maximum_storeys|        Mean_Price|
+--------------------+----+------------------+---------------------------------+-----------------+
|             ENFIELD|2014|   2.27027027027027|                14.243243243243244|     377910.940625|
|          HILLINGDON|2017|  2.314814814814815|                 7.1111111111111111|          495000.0|
|              MERTON|2017|             2.375|                 4.086538461538462| 836466.2555066079|
|         WESTMINSTER|2018|  2.207692307692308|                 8.138461538461538|              null|
|              LAMBETH|2015|               2.0|                         31.44375| 604497.8282076395|
|            HOUNSLOW|2016| 1.9426229508196722|                 7.475409836065574| 973610.5623762376|
|             BROMLEY|2018| 2.2142857142857144|                3.2857142857142856|417977.14483627205|
|      CITY OF LONDON|2016|               2.2|                             11.0| 5596348.251028807|
|       TOWER HAMLETS|2017| 1.9627118644067796|                             18.8| 830312.8342342342|
|              BEXLEY|2015|              2.25|                 3.054054054054054| 271562.4835164835|
|          WANDSWORTH|2015| 2.2460526315789475|                 18.21184210526316|  702075.705095057|
|               BRENT|2017| 1.9587155963302751|                13.252293577981652| 725372.9486143187|
|BARKING AND DAGENHAM|2015| 2.3855421686746987|                 9.602409638554217|          383500.0|
|              BARNET|2016| 2.2564102564102564|                 7.256410256410256| 638135.2124738189|
|BARKING AND DAGENHAM|2016| 1.8833333333333333|                 7.416666666666667|         1430000.0|
|             CROYDON|2017| 2.1325167037861914|                 4.271714922048997|389188.81743119267|
|              MERTON|2014|             2.525|                             3.13|  694368.585303186|
|              BEXLEY|2017| 2.1288659793814433|                2.6804123711340204| 307851.7458563536|
|              BEXLEY|2014|  2.240740740740741|                3.2777777777777777|239084.14646464647|
|             CROYDON|2015|  2.153679653679654|                 4.571428571428571|333223.20958083833|
+--------------------+----+------------------+---------------------------------+-----------------+
only showing top 20 rows
```

Rysunek 11: Zrzut ekranu ze sprawdzeniem czy tabela 2 poprawnie się utworzyła.

Writing batch views

```
merged_ym.write.parquet('hdfs://localhost:8020/user/matuszewskis/project/PricePaidData/merged_year_month.parquet')
merged_by.write.parquet('hdfs://localhost:8020/user/matuszewskis/project/PricePaidData/merged_borough_year.parquet')
```

Rysunek 12: Zrzut ekranu z zapisem do pliku wsadowego wywołany aby sprawdzić istnienie zapisu na HDFS.

```
Found 42 items
drwxr-xr-x   - vagrant supergroup          0 2024-01-07 13:02 /user/matuszewskis/project/PricePaidData/merged_bo
rough_year.parquet
drwxr-xr-x   - vagrant supergroup          0 2024-01-07 13:02 /user/matuszewskis/project/PricePaidData/merged_ye
ar_month.parquet
```

Rysunek 13: Zrzut ekranu z logiem faktycznego zapisu widoków wsadowych na HDFS.

Wynik: Poprawnie dodano nowe pliki.