Analiza Danych przestępczości oraz danych mieszkaniowych w mieście Nowy York - Dokumentacja Big Data

Adam Frej, Jan Gąska

Styczeń 2023

Spis treści

1 Wstęp/Abstrakt

Projekt skupia się na wykonaniu narzędzia do obsługi dużych wolumenów danych oraz, finalnie, dokonujących ich analizy w kontekście miasta Nowy York. Narzędzie pobiera dane z dwóch źródeł, poddaje je preprocessingowi, przechowuje je, dokonuje agregacji i ostatecznie poddaje wizualizacji. Pierwszy zestaw danych skupia się na zaobserwowanych przestępstwach na całej przestrzeni miasta, podczas gdy drugi bierze pod uwagę dane geograficzne oraz ekonomiczne mieszkaniowe dla miasta Nowy York (bardziej dokładnie są to dane dotyczące jednostek mieszkaniowych znanych w USA jako kondominia (condominium). Można je traktować jako apartamentowce zorientowane na wynajem mieszkań).
Narzędzie ma automatyzować proces poboru danych i ich analizy. System pozwala znaleźć obszary miasta z najgorszą przestępczością z podziałem na kategorie zdarzeń i sprawdza czy występuje zależność finansowa. Jego potencjalnymi korzyściami biznesowymi są:

- możliwość wykrycia zwiększonej przestępczości dla danego rejonu Nowego Yorku oraz jego typu (interwencja i prewencja). Opcja dla jednostek administracyjnych miasta.
- Dla mieszkańców poszukujących mieszkania pod wynajem w Nowym Yorku - możliwość znalezienia bezpiecznego i ekonomicznego kondominium pod wynajem.
- Dla przedsiębiorców w Nowym Yorku - możliwość relokacji zasobów lub zwiększenia prewencji występowania kradzieży i przywłaszczeń.

1.1 Odnośniki

Struktura projektu oraz wszystkie użyte w nim pliki są utrzymywane na repozytorium Git-hubowym, które można znaleźć pod adresem: github.com/adFrej/NewYork-BigData.
W strukturze repozytorium znajduje się opis poszczególnych plików i ich funkcji w projekcie oraz pliki znajdujące się na repozytorium są najbardziej aktualnymi wersjami w projekcie.

2 Dane

W projekcie posługujemy się dwoma zbiorami danych, udostępnionymi przez administrację miasta Nowego Yorku na stronie opendata.cityofnewyork.us. Dane są udostępnione na podstawie tamtejszego "Open Data Law", które ustanawia, iż dane zbierane przez różne jednostki administracyjne Zarządu Miasta Nowego Yorku muszą być w sposób darmowy ogólnodostępne na oficjalnej stronie internetowej. Zatem korzystanie z danych oraz prowadzenie analiz jest dozwolone oraz w pełni legalne. W naszej analizie korzystamy z dwóch tabel pochodzących z dwóch źródeł: danych zgłoszeń na do Nowojorskiego Departamentu Policji oraz danych zestawiających finansowe statystyki bloków z mieszkaniami własnościowymi (tzw. condominium), które zobowiązane są do zbierania statystyk.
Dane można pobrać bezpośrednio ze strony (bez logowania) w następujących formatach: CSV, TSV, RDF, XML, RSS (wykorzystujemy format CSV). Ramki danych można znaleźć pod następującymi odnośnikami:

- Dane dotyczące przestępczości
- Dane mieszkaniowe

Dokładniejszy opis danych zostanie umieszczony w odpowiednich sekcjach.

2.1 Dane dotyczące przestępczości

Dane zostały udostępnione przez Departament Policji Miasta Nowego Yorku, który zbiera, przetwarza oraz przechowuje informacje dotyczące zgłaszanych przestępstw. Dane składają się z 35 kolumn oraz posiadają 7.83 miliona rekordów, gdzie każdy rekord odpowiada faktowi dokonania zgłoszenia. Symulowane jest odświeżanie dzienne. Dane zgłoszenie zawiera informacje o:

- Miejscu zajścia (dzielnica + szerokość i długość geograficzna danego miejsca)
- Czas zajścia z dokładnością do sekundy
- Klasyfikacja zajścia (czego dotyczyło zgłoszenie, klasa spośród możliwych)
- Rasa, płeć oraz przedział wiekowy napastnika
- Rasa, płeć oraz przedział wiekowy ofiary
- Jurysdykcja policyjna miejsca zajścia

2.2 Dane mieszkaniowe

Dane zostały dostarczone przez Departament Finansów Miasta Nowego Yorku, który zbiera i dokonuje administracji danych finansowych wspólnot mieszkaniowych. Dane składają się z 61 kolumn oraz 28.5 tysięcy wierszy, gdzie każdy wiersz stanowi daną wspólnotę wchodzącą w skład jednego kondominium, zawiera jej dane finansowe oraz dane fizyczno-administracyjne (adres, pole powierzchni, rok założenia). **Uwaga**. Dane nie są atomowe i zostaną poddane obróbce. Wynika to ze względu na ponowienie (poczwórne) ciągu danych zachodzących dla danego rekordu, z tą różnicą, iż każde powtórzenie odwołuje się do podobnej jednostki mieszkalnej w zakresie danego kondominium. Na poziomie pre-processingu rozdzielone zostają 4 różne rekordy. Wedle opisu dostarczyciela danych, powielone opisy jednostek wchodzących w jeden rekord nie powielane są w następnych rekordach. Dane odświeżane są z częstotliwością roczną, a ich przedział czasowy obejmuje lata 2012 - 2021.
Ramka danych posiada dokładny opis w załączniku na stronie Dane mieszkaniowe, jednakże w skrócie opiszę jakie ważne informacje można uzyskać z danych zestawów kolumn, które zostaną zutylizowane podczas przetwarzania danych oraz ich agregacji z drugą ramką. Informacje, po zatomizowaniu, w rekordzie będą przechowywać:

- Dokładny adres danego lokalu (położenie w mieście oraz w danym bloku i przynależność do dzielnicy)
- Przychód przypadający na mieszkanie
- Przychód całościowy na stopę kwadratową powierzchni
- Całkowite pole powierzchni mieszkania
- Całkowita wartość rynkowa
- Rok budowy budynku
- Rok zgłoszenia przychodów

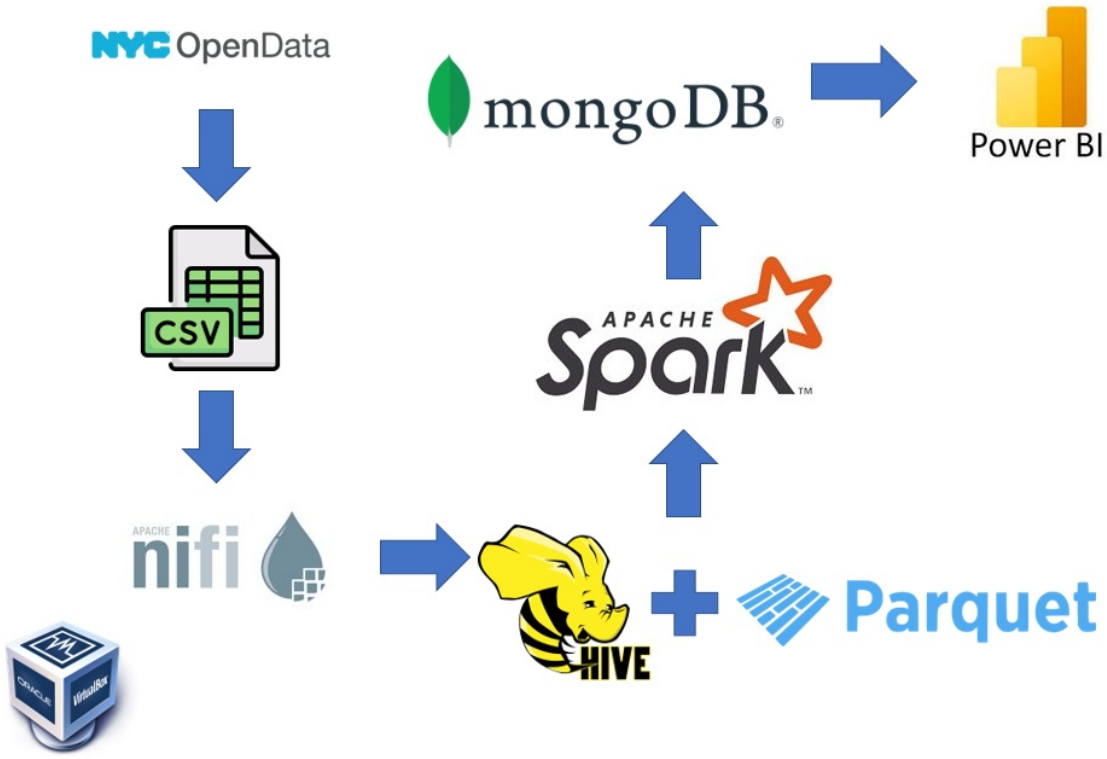
Dane pozwolą na przeprowadzenie analizy ze względu na typ danego bloku, średni przychód (odzwierciedlenie zamożności danego obszaru) oraz na geografię bloku (położenie w dzielnicy, adres).

3 Spis architektoniczny

Trzymając się założeń konspektu (za wyjątkiem warstwy analitycznej, w której wizualizacje wykonywane są w PowerBI), poszczególny etap obsługi danych został zaimplementowany w następujących narzędziach, które operowały na lokalnym środowisku:

- Składowanie - Apache Hive został użyty do przetrzymywania wszystkich niezagregowanych danych.
- Składowanie - Mongodb został użyty do składowania już obrobionych i zagregowanych danych, by następnie można było z łatwością dokonać
- Przepływ danych - Apache Nifi został użyty do preprocessingu i ładowania danych do Apache Hive.
- Processing - Apache Spark został użyty do wykonywania wielkoskalowych obliczeń oraz agregacji i ładowania ich do Mongodb.
- Analityka - PowerBI został użyty do wykonywania wizualizacji.

Flow-chart architektonicznego podejścia zaprezentowany jest na Rysunku 1.



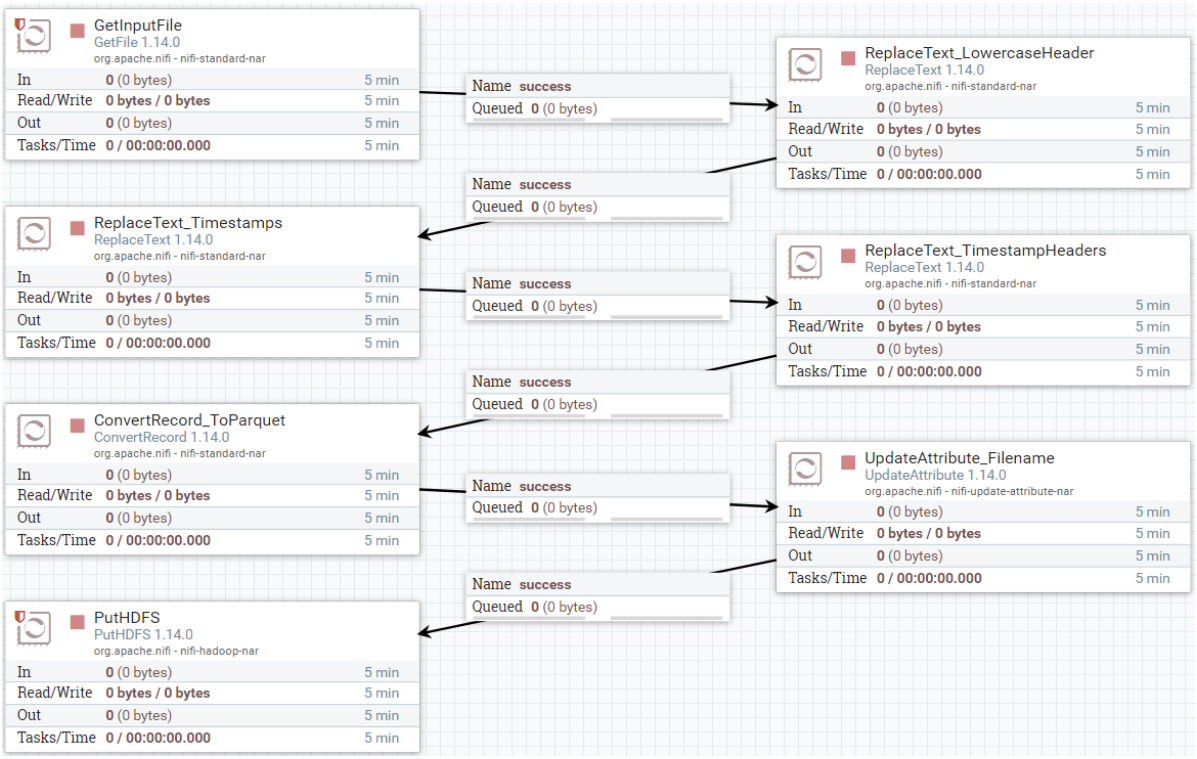
Rysunek 1: Graficzna reprezentacja stosu architektonicznego

Poniżej zamieszczamy dokładniejszy opis sposobu implementacji poszczególnych składowych projektu w poszczególnych narzędziach.

3.1 Nifi

3.1.1 Preprocessing danych dotyczących przestępczości

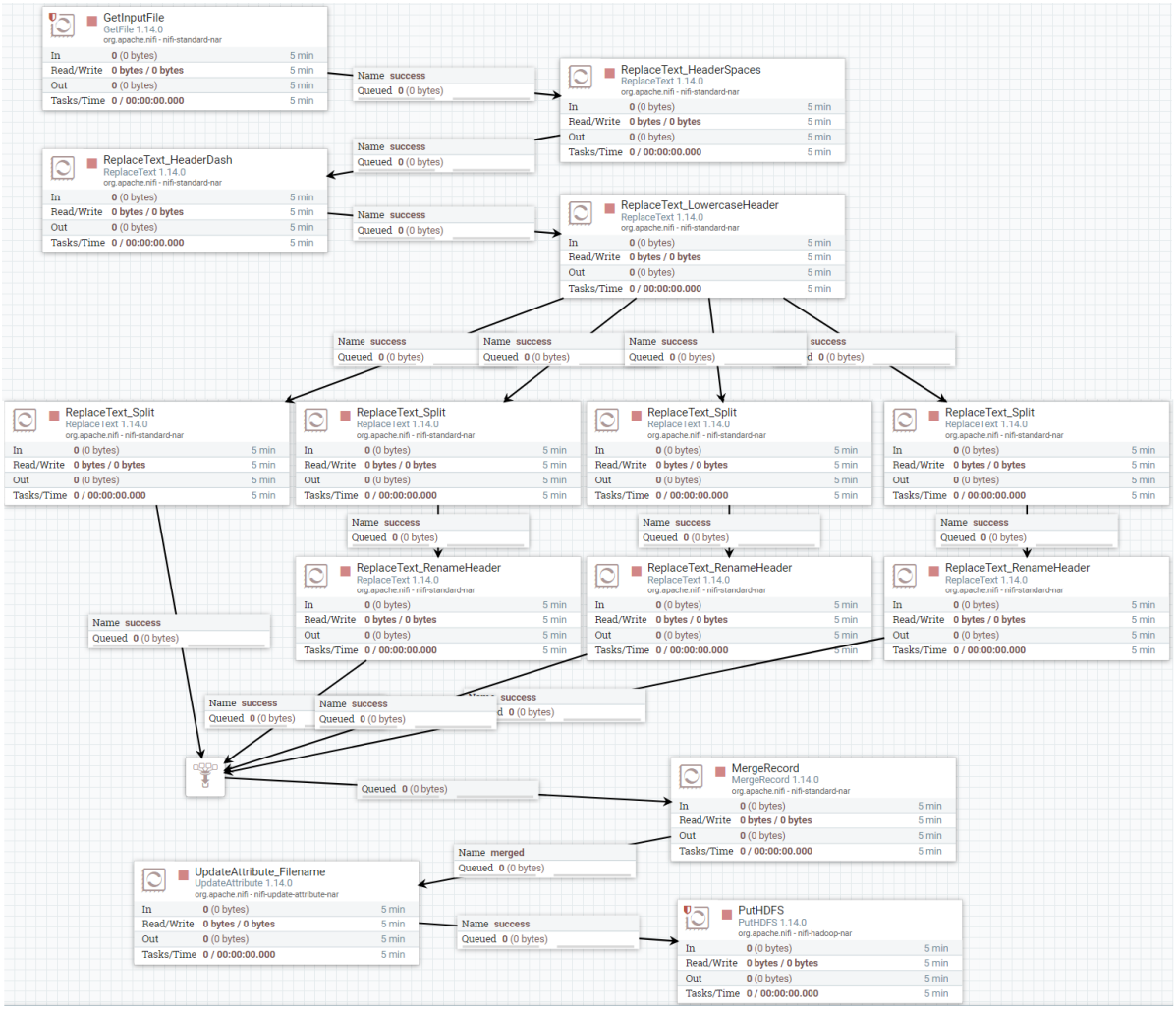
Na Rysunku 2 przedstawiony jest diagram przetwarzania danych od Nypd. Schemat zaczyna się od pobrania danych zawartych w surowych plikach CSV. Następnie nagłówek w plikach konwertowany jest do małych liter. W oryginalnych danych zdarzenia moment zajścia przestępstw podany był przy pomocy kolumn z datą i godziną. Tutaj te pola są łączone do timestampa. Na koniec pliki konwertowane zostają do formatu PARQUET przy pomocy ręcznie zdefiniowanej schemy AVRO, która jednoznacznie definiuje typy pól i pomija te niepotrzebne (np. powtórzona informacja o współrzędnych geograficznych). Finalnie pliki umieszczane są w systemie HDFS w folderze tabli HIVE.



Rysunek 2: Diagram Nifi dla Nypd

3.1.2 Preprocessing danych mieszkaniowych

Na Rysunku 3 przedstawiony jest diagram przetwarzania danych mieszkaniowych. Podobnie pobierane są pliki CSV i konwertowany nagłówek. Tym razem potrzebne też jest usunięcie białych znaków i myślników z nazw kolumn. Następnie usuwane jest powielenie obserwacji w jednym rekordzie. W tym celu dane dzielone są na 4 odrębne kopie. W każdej przy pomocy regexa wyodrębniany jest potrzebny podzbiór kolumn i ujednolicany zostaje nagłówek. Następnie dane są z powrotem łączone razem wiersz po wierszu. W ten sposób liczba kolumn zostaje zredukowana z 61 do 16, a wierszy przybywa czterokrotnie. Na koniec dane ponownie konwertowane są do plików PARQUET przy użyciu schemy AVRO i ładowane do HDFS dla tabeli HIVE.



Rysunek 3: Diagram Nifi dla mieszkań

3.2 Hive z parquet

Dla każdego źródła stworzona jest tabela external Hive wskazująca na dany folder ze składowanymi plikami PARQUET. Ich definincje pokazane są na Rysunku 4. Są one spójne ze schemami AVRO używanymi w Nifi. Do tabel trafiają dane w pełni przeprocesowane z formatami gotowymi do wczytania przez Sparka. Format PARQUET jest tu szczególnie uzasadniony, jako że dane w plikach CSV są od razu podzielone na kolumny.

```
CREATE EXTERNAL TABLE external_table_nypd
(cmplnt_num int,
cmplnt_fr_ts timestamp,
cmplnt_to_ts timestamp,
addr_pct_cd int,
rpt_dt date,
ky_cd int,
ofns_desc string,
pd_cd int,
pd_desc string,
crm_atpt_cptd_cd string,
law_cat_cd string,
boro_nm string,
loc_of_occur_desc string,
prem_typ_desc string,
juris_desc string,
jurisdiction_code int,
parks_nm string,
hadevelopt string,
housing_psa string,
x_coord_cd int,
y_coord_cd int,
susp_age_group string,
susp_race string,
susp_sex string,
transit_district double,
latitude double,
longitude double,
patrol_boro string,
station_name string,
vic_age_group string,
vic_race string,
vic_sex string)
STORED AS PARQUET
LOCATION '/projekt/external_table_nypd';
```

```
CREATE EXTERNAL TABLE external_table_condo
(condo_section string,
boro_block_lot string,
address string,
neighborhood string,
building_classification string,
total_units int,
year_built int,
gross_sqft int,
estimated_gross_income int,
gross_income_per_sqft double,
estimated_expense int,
expense_per_sqft double,
net_operating_income int,
full_market_value int,
market_value_per_sqft double,
report_year int)
STORED AS PARQUET
LOCATION '/projekt/external_table_condo';
```

Rysunek 4: Definicje tabel Hive

3.3 Apache Spark

Spark wczytuje dane z tabel Hive. Dzięki temu nie musi katalogować leżących pod spodem plików PARQUET, które są dokładane z biegiem czasu. Dane są w pełni oczyszczone, więc nie potrzebny jest już żaden preprocessing i ramki są od razu gotowe do agregacji.

3.3.1 Przykłady danych dostępnych dla warstwy prezentacyjnej

Poniżej zamieszczam kilka przykładów danych, które po transformacji przy użyciu Apache Sparka ładowane są do bazy mongoDB oraz w następstwie wykorzystywane są w warstwie prezentacyjnej.

vic_race	vic_age_group	vic_sex	boro_nm	law_cat_cd	count
UNKNOWN	UNKNOWN	D	MANHATTAN	MISDEMEANOR	3960
UNKNOWN	UNKNOWN	D	BROOKLYN	MISDEMEANOR	2447
UNKNOWN	UNKNOWN	E	BRONX	MISDEMEANOR	2156
UNKNOWN	UNKNOWN	E	MANHATTAN	MISDEMEANOR	2072
UNKNOWN	UNKNOWN	E	BROOKLYN	MISDEMEANOR	2067
UNKNOWN	UNKNOWN	D	QUEENS	MISDEMEANOR	1717
UNKNOWN	UNKNOWN	D	BRONX	MISDEMEANOR	1503
UNKNOWN	UNKNOWN	D	MANHATTAN	FELONY	1466
UNKNOWN	UNKNOWN	E	BROOKLYN	FELONY	1430
BLACK	25-44	F	BROOKLYN	MISDEMEANOR	1341
UNKNOWN	UNKNOWN	E	QUEENS	MISDEMEANOR	1032
UNKNOWN	UNKNOWN	D	BROOKLYN	FELONY	988
BLACK	25-44	F	BRONX	MISDEMEANOR	964
WHITE HISPANIC	25-44	F	BRONX	MISDEMEANOR	952
BLACK	25-44	M	BROOKLYN	MISDEMEANOR	890
UNKNOWN	UNKNOWN	E	MANHATTAN	FELONY	880
UNKNOWN	UNKNOWN	E	BRONX	FELONY	840
BLACK	25-44	F	BROOKLYN	VIOLATION	774
BLACK	25-44	F	BROOKLYN	FELONY	760
WHITE	25-44	M	BROOKLYN	MISDEMEANOR	724

only showing top 20 rows

Rysunek 5: Tabela po zagregowanych danych dotyczących ofiar

Dane dla tabeli 5 zostały zagregowane przy użyciu zmiennych kategorycznych :

- vic_race - rasa ofiary przestępstwa
- vic_age_group - przedział wiekowy ofiary przestępstwa
- vic_sex - płeć ofiary przestępstwa
- boro_nm - nazwa dzielnicy gdzie przestępstwo miało miejsce
- law_cat_cd - kategoria napaści definiowana przez praco stanu Nowy York

Użytą funkcją agregacyjną była funkcja zliczeń dla poszczególnych kategorii.
Drugim przykładem danych dostępnych w wersji prezentacyjnej jest poniższa tabela (pokazana w schemacie ze względu na nieprzejrzyste formatowanie się kolumn) reprezentujące zagregowane dane ze względu na geografię (adres, sąsiedztwo) danego kondominium.

```
In [35]: df_geo_condo.printSchema()

root
 |-- neighborhood: string (nullable = true)
 |-- address: string (nullable = true)
 |-- building_classification: string (nullable = true)
 |-- report_year: integer (nullable = true)
 |-- avg(net_operating_income): double (nullable = true)
 |-- avg(market_value_per_sqft): double (nullable = true)
 |-- avg(expense_per_sqft): double (nullable = true)
 |-- avg(gross_income_per_sqft): double (nullable = true)
```

Rysunek 6: Tabela po zagregowanych danych dotyczących kondominiów

Dane oryginalne zostały zagregowane przy użyciu następujących zmiennych kategorycznych:

- neighborhood - nazwa sąsiedztwa, gdzie znajduje się dane kondominium

- address - adres, na jakim znajduje się dane kondominium
- report_year - rok zgłoszenia danych fiskalnych dla danego kondominium

Zostały użyte następujące funkcje agregacyjne na następujących kolumnach:

- Średnia na net_operating_income - na ilorazie operacyjnym netto dla zagregowanych kondominiów (w dolarach)
- Średnia na market_value_per_sqft - na wartości rynkowej dla zagregowanych kondominiów na stopę kwadratową jej powierzchni (w dolarach)
- Średnia na expense_per_sqft - wydatek (dokonywany przez właściciela) na stopę kwadratową powierzchni dla zagregowanych kondominiów (w dolarach na miesiąc)
- Średnia na gross_income_per_sqft - na dochodach brutto na stopę kwadratową powierzchni dla zagregowanych kondominiów (w dolarach na miesiąc)

3.4 MongoDB

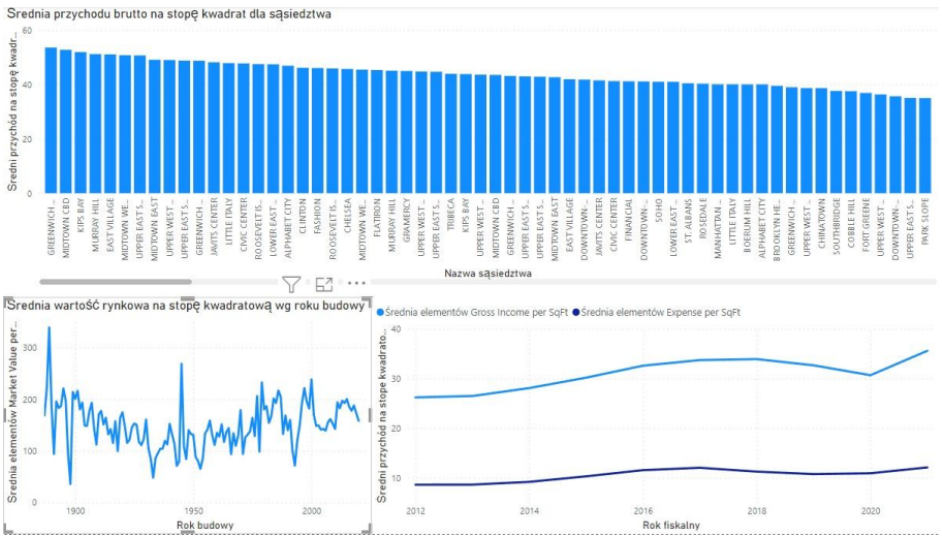
Baza Mongo używana jest do składowania gotowych agregacji w formie NoSQL. W ten sposób są one łatwo dostępne dla warstwy prezentacyjnej. Wyniki obliczeń nie są skomplikowane i ich forma może się zmienić z czasem, więc Mongo jest tu świetnym wyborem. Dodatkowo baza łatwo integruje się z PowerBI.
W ramach projektu stworzone zostały cztery kolekcje: po jednej dla samotnych agregacji zbiorów i dwie dla zespolonych danych.

3.5 PowerBI

Finalnie wszystkie wizualizacje zostały przygotowane i wyświetlone przy użyciu PowerBI, po bezpośrednim podłączeniu się narzędzia do bazy mongoDB i wczytaniu ich ze źródła.

4 Wizualizacje otrzymane przez analizę w PowerBI

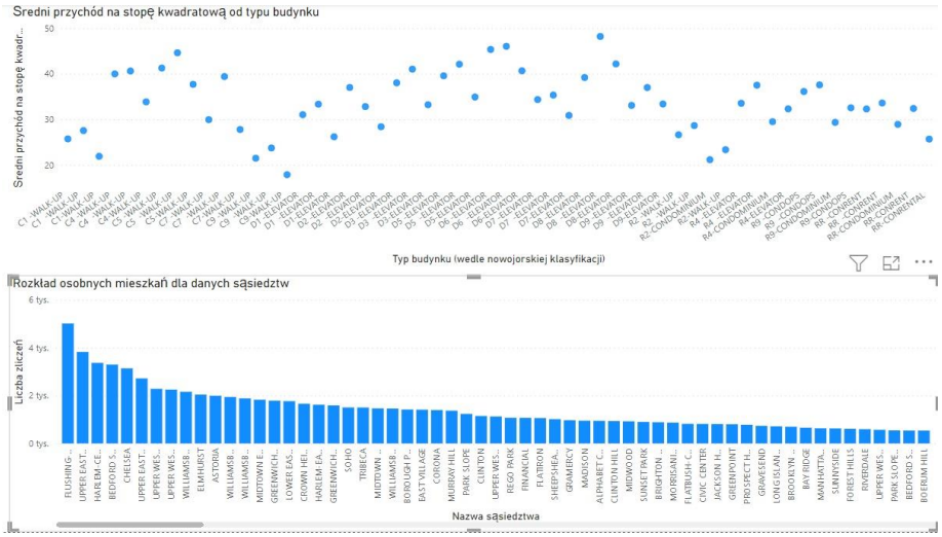
Poniżej zamieszczam kilka wizualizacji otrzymanych za pomocą narzędzia PowerBI wraz z krótkim objaśnieniem otrzymanych grafów.



Rysunek 7: Przykładowa wizualizacja ze względu na dane geograficzne i czasowe

W wizualizacji 7 występują następujące elementy:

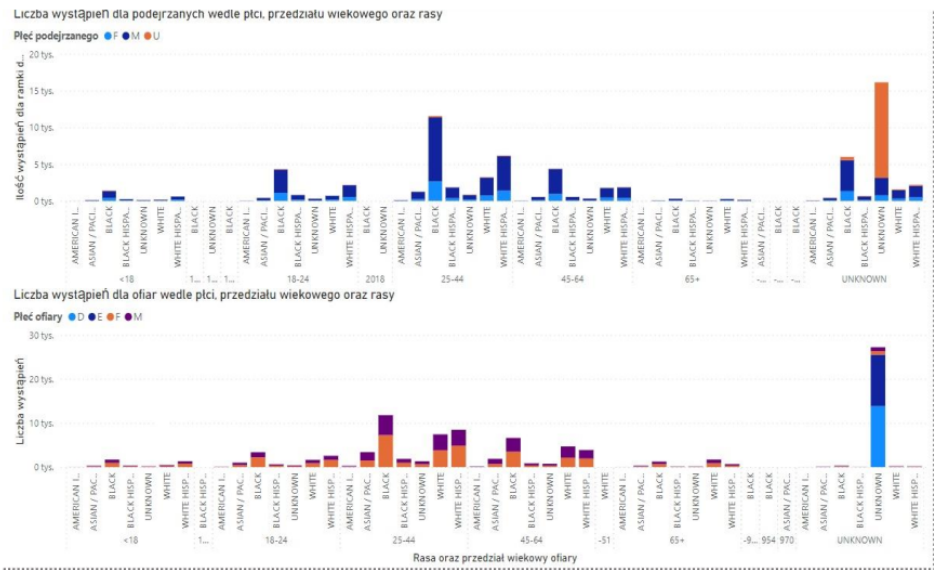
- Górny graf - Średni przychód na stopę kwadratową (w dolarach na miesiąc) dla kondominiów w zależności od sąsiedztwa.
- Lewy dolny graf - Zależność średniej wartości rynkowej na stopę kwadratową (w dolarach na stopę kwadrat) w zależności od roku budowy kondominium.
- Prawy dolny graf - Zależność średniego przychodu brutto na stopę kwadrat (jasno-niebieska linia) oraz średni wydatek na stopę kwadratową (ciemno-niebieska linia) w zależności od roku fiskalnego.



Rysunek 8: Przykładowa wizualizacja ze względu na klasyfikację budynku oraz ilości kondominiów w danych sąsiedztwach

W wizualizacji 8 występują następujące elementy:

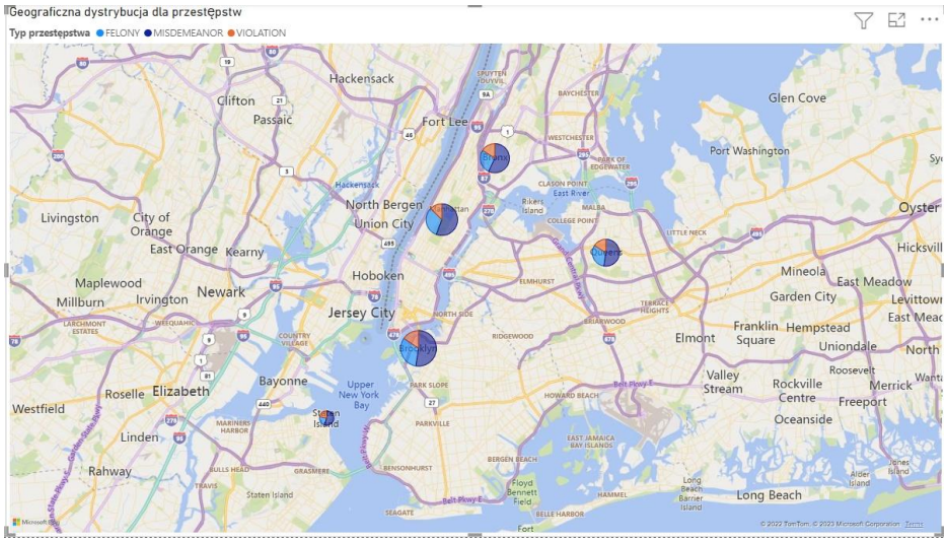
- Górny graf - Zależność średniego przychodu na stopę kwadratową dla danych kondominiów w zależności od typu kondominium (w USA zdefiniowane są różne typy kondominiów z przypisanym im kodem składającym się z litery oraz jednej cyfry, odnośnik)
- Dolny graf - Rozkład ilości osobnych mieszkań (które wchodzą w skład kondominiów) w zależności od sąsiedztwa.



Rysunek 9: Przykładowa wizualizacja ze względu dystrybucję napastników oraz ofiar napaści

W wizualizacji 9 występują następujące elementy:

- Górny graf - Dystrybucja wystąpień poszczególnych napaści podzielona ze względu na rasę, przedział wiekowy oraz płeć napastników.
- Dolny graf - Dystrybucja wystąpień ofiar napaści podzielona ze względu na rasę, przedział wiekowy oraz płeć napastników.

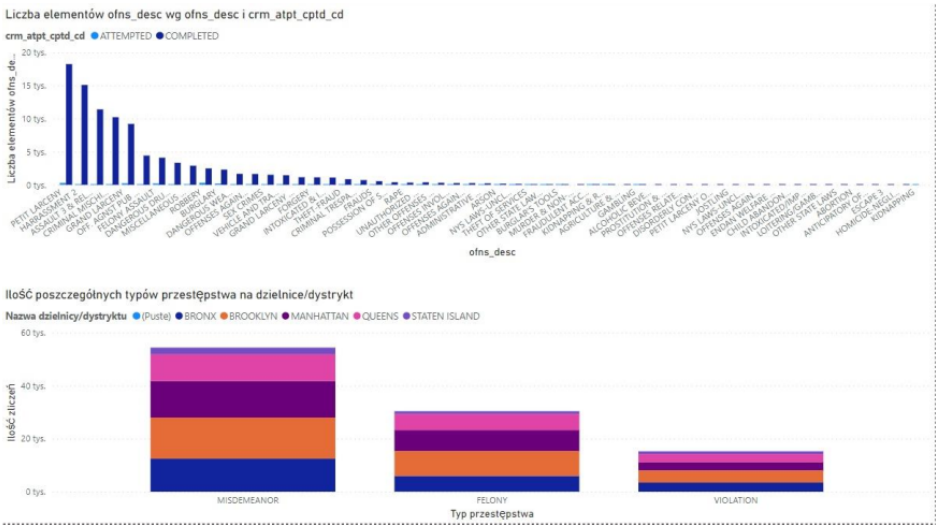


Rysunek 10: Dystrybucja przestępstw ze względu na ich klasyfikację nałożona na mapę geograficzną dzielnic Nowego Yorku

W wizualizacji 10 występują następujące elementy:

- Graf geograficzny z nałożoną dystrybucją przestępstw w Nowym Yorku z podziałem na ich typ dla danych dzielnic NYC:
  - Felony - Przestępstwo

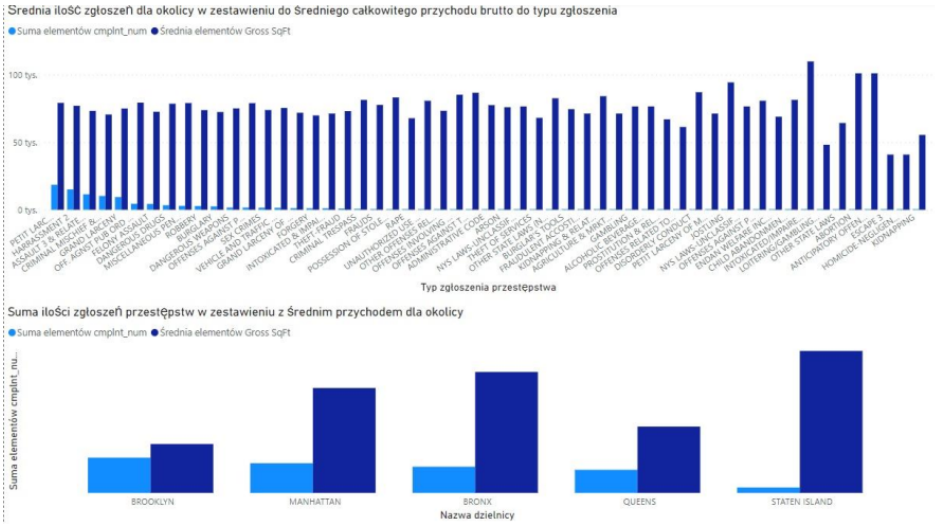
– Misdemeanor - Wykroczenie
– Violation - Naruszenie prawa



Rysunek 11: Przykładowa wizualizacja pod względem opisu danego przestępstwa (bardziej dokładna niż jego klasifikacja) oraz dystrybucja typów przestępstw na danych dzielnicach/dystryktach NYC

W wizualizacji 11 występują następujące elementy:

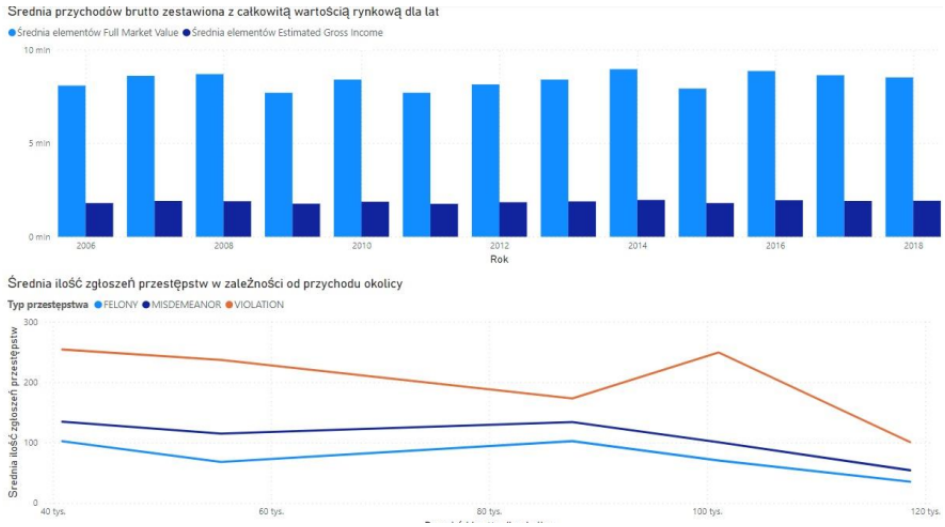
- Graf górny - Dystrybucja występowania opisu danego przestępstwa wraz z klasyfikacją jego wykonania (ATTEMPTED - Usiłowany, COMPLETED - Wykonany)
- Graf dolny - Ilość poszczególnych typów przestępstwa na dzielnice/dystrykt



Rysunek 12: Przykładowa wizualizacja pod względem typu przestępstwa a średniego przychodu brutto w stopach kwadratowych dla całych kondominiów w sąsiedztwie

W wizualizacji 12 występują następujące elementy:

- Graf górny - Średnia ilość zgłoszeń dla sąsiedztwa danego przestępstwa (kolor jasno-niebieski) w zestawieniu do średniego całkowitego przychodu brutto w zestawieniu (kolor ciemno-niebieski)
- Graf dolny - Suma ilości zgłoszeń przestępstw (kolor jasno-niebieski) w zestawieniu z średnim przychodem dla okolicy (kolor ciemno-niebieski)



Rysunek 13: Przykładowa wizualizacja reprezentująca dynamikę ilości zgłoszeń przestępstw od przychodu dla danej okolicy

W wizualizacji 13 występują następujące elementy:

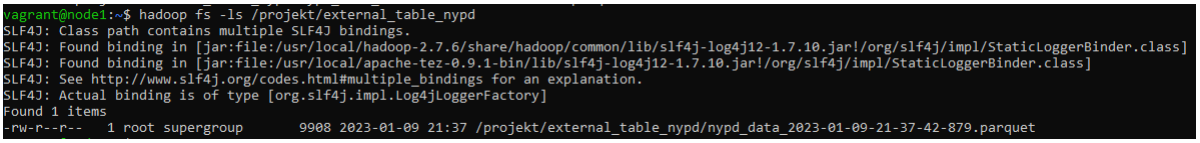
- Graf górny - Średnia przychodów brutto zestawiona z całkowitą wartością rynkową dla lat (jasno-niebieski - średnia wartość rynkowa, ciemno-niebieski - średni przychód brutto)
- Graf dolny - Średnia ilość zgłoszeń przestępstw w zależności od przychodu okolicy (w dolarach na rok) w podziale na typ zgłoszenia (jasno-niebieski - przestępstwo, ciemno-niebieski - wykroczenie, pomarańczowy - naruszenie prawa)

5 Testy Funkcjonalne

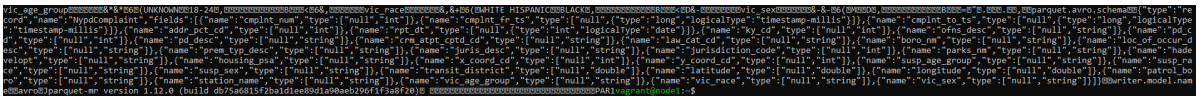
Tabela 1 przedstawia wyniki testów.

Nr	Cel	Kroki	Oczekiwany wynik	Faktyczny wynik
1	Przepływ danych w Nifi dla Nypd.	Załadowanie porcji danych CSV do Nifi i wyświetlenie zawartości folderu docelowego oraz fragmentu pliku.	Odpowiedni plik PARQUET w folderze z dobrze zdefiniowaną schemą.	Poprawne wyniki na Rysunkach: 14, 15.
2	Przepływ danych w Nifi dla mieszkań.	Załadowanie porcji danych CSV do Nifi i wyświetlenie zawartości folderu docelowego oraz fragmentu pliku.	Odpowiedni plik PARQUET w folderze z dobrze zdefiniowaną schemą.	Poprawne wyniki na Rysunkach: 16, 17.
3	Ładowanie poprawnych danych do Sparka dla Nypd.	Rozszerzenie testu 1 o wyświetlenie danych w ramkach Sparka.	Poprawne dane z typami w ramce.	Poprawne wyniki na Rysunkach: 18, 19.
4	Ładowanie poprawnych danych do Sparka dla mieszkań.	Rozszerzenie testu 2 o wyświetlenie danych w ramkach Sparka.	Poprawne dane z typami w ramce (w szczególności czterokrotne zwiększenie liczby wierszy).	Poprawne wyniki na Rysunkach: 20, 21, widoczne 40 wierszy dla 10 w testowym CSV.

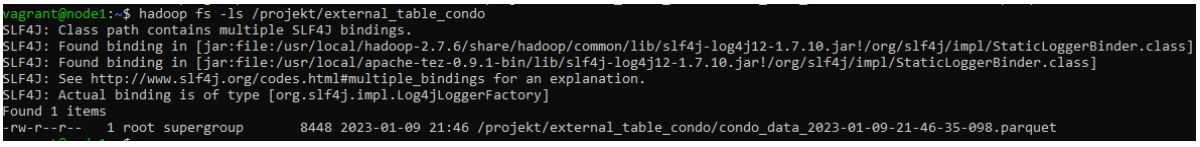
Tabela 1: Testy funkcjonalne



Rysunek 14: Plik PARQUET dla Nypd



Rysunek 15: Schema w pliku PARQUET dla Nypd



Rysunek 16: Plik PARQUET dla mieszkań

Rysunek 17: Schema w pliku PARQUET dla mieszkań

```
spark.sql("REFRESH TABLE external_table_nypd")  # refreshes table after adding files
df_nypd = spark.table("external_table_nypd")
df_nypd.show()
```

```
+----------+-------------------+-------------------+-----------+----------+-----+--------------------+-----+--------------------
-+----------------+-----------+---------+----------------+--------------------+--------------------+----------------+--------+
----------+-----------+----------+----------+--------------+--------------+--------+----------------+--------------------+-------
----------+--------------------+-----------+------------+-------------+-------+
|cmplnt_num|       cmplnt_fr_ts|       cmplnt_to_ts|addr_pct_cd|    rpt_dt|ky_cd|           ofns_desc|pd_cd|             pd_des
c|crm_atpt_cptd_cd| law_cat_cd|  boro_nm|loc_of_occur_desc|       prem_typ_desc|         juris_desc|jurisdiction_code|parks_nm|
hadevelopt|housing_psa|x_coord_cd|y_coord_cd|susp_age_group|     susp_race|susp_sex|transit_district|          latitude|
longitude|         patrol_boro|station_name|vic_age_group|      vic_race|vic_sex|
+----------+-------------------+-------------------+-----------+----------+-----+--------------------+-----+--------------------
-+----------------+-----------+---------+----------------+--------------------+--------------------+----------------+--------+
----------+-----------+----------+----------+--------------+--------------+--------+----------------+--------------------+-------
----------+--------------------+-----------+------------+-------------+-------+
| 506547392|2018-03-29 20:30:00|               null|         32|2018-03-30|  351|CRIMINAL MISCHIEF...|  254|MISCHIEF, CRIMIN
A...|       COMPLETED|MISDEMEANOR|MANHATTAN|         FRONT OF|PARKING LOT/GARAG...|   N.Y. POLICE DEPT|               0|    nu
ll|      null|       null|   1000565|    234704|          null|          null|    null|            null| 40.81087724100007|-73.
94106415099996|PATROL BORO MAN N...|       null|        25-44|         WHITE|      F|
| 629632833|2018-02-06 23:15:00|               null|         52|2018-02-07|  341|       PETIT LARCENY|  333|LARCENY,PETIT FR
O...|       COMPLETED|MISDEMEANOR|    BRONX|           INSIDE|    DEPARTMENT STORE|   N.Y. POLICE DEPT|               0|    nu
ll|      null|       null|   1009690|    257590|         45-64|         BLACK|       F|            null| 40.87367103500002|-73.
90801364899994|   PATROL BORO BRONX|       null|      UNKNOWN|       UNKNOWN|      D|
```

Rysunek 18: Ramka Nypd w Sparku

```
df_nypd.printSchema()
```

```
root
 |-- cmplnt_num: integer (nullable = true)
 |-- cmplnt_fr_ts: timestamp (nullable = true)
 |-- cmplnt_to_ts: timestamp (nullable = true)
 |-- addr_pct_cd: integer (nullable = true)
 |-- rpt_dt: date (nullable = true)
 |-- ky_cd: integer (nullable = true)
 |-- ofns_desc: string (nullable = true)
 |-- pd_cd: integer (nullable = true)
 |-- pd_desc: string (nullable = true)
 |-- crm_atpt_cptd_cd: string (nullable = true)
 |-- law_cat_cd: string (nullable = true)
 |-- boro_nm: string (nullable = true)
 |-- loc_of_occur_desc: string (nullable = true)
 |-- prem_typ_desc: string (nullable = true)
 |-- juris_desc: string (nullable = true)
 |-- jurisdiction_code: integer (nullable = true)
 |-- parks_nm: string (nullable = true)
 |-- hadevelopt: string (nullable = true)
 |-- housing_psa: string (nullable = true)
 |-- x_coord_cd: integer (nullable = true)
 |-- y_coord_cd: integer (nullable = true)
 |-- susp_age_group: string (nullable = true)
 |-- susp_race: string (nullable = true)
 |-- susp_sex: string (nullable = true)
 |-- transit_district: double (nullable = true)
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
 |-- patrol_boro: string (nullable = true)
 |-- station_name: string (nullable = true)
 |-- vic_age_group: string (nullable = true)
 |-- vic_race: string (nullable = true)
 |-- vic_sex: string (nullable = true)
```

Rysunek 19: Typy danych w ramce Nypd w Sparku

```
spark.sql("REFRESH TABLE external_table_condo")  # refreshes table after adding files
df_condo = spark.table("external_table_condo")
df_condo.show()
```

```
+-------------+--------------+--------------------+--------------------+----------------------+-----------+----------+--------
--+----------------------+---------------------+-----------------+----------------+--------------------+-----------------+-----
----------------+-----------+
|condo_section|boro_block_lot|             address|        neighborhood|building_classification|total_units|year_built|gross_sq
ft|estimated_gross_income|gross_income_per_sqft|estimated_expense|expense_per_sqft|net_operating_income|full_market_value|marke
t_value_per_sqft|report_year|
+-------------+--------------+--------------------+--------------------+----------------------+-----------+----------+--------
--+----------------------+---------------------+-----------------+----------------+--------------------+-----------------+-----
----------------+-----------+
|      0003-R1|  1-00576-7501|   60 WEST 13 STREET|GREENWICH VILLAGE...|          R4  -ELEVATOR|         70|      1966|     820
17|               4452703|                54.29|           1729739|           21.09|             2722964|         22115002|
269.64|       2019|
|      0007-R2|  1-01271-7501|       1360 6 AVENUE|        MIDTOWN WEST|          R4  -ELEVATOR|        183|      1963|    1417
38|               7113830|                50.19|           2361355|           16.66|             4752475|         38596999|
272.31|       2019|
|      0009-R1|  1-00894-7501|      77 PARK AVENUE|         MURRAY HILL|          R4  -ELEVATOR|        109|      1924|    1585
71|               7329152|                46.22|           2854278|            18.0|             4474874|         36343010|
229.19|       2019|
```

Rysunek 20: Ramka mieszkań w Sparku

```
df_condo.printSchema()
```

```
root
 |-- condo_section: string (nullable = true)
 |-- boro_block_lot: string (nullable = true)
 |-- address: string (nullable = true)
 |-- neighborhood: string (nullable = true)
 |-- building_classification: string (nullable = true)
 |-- total_units: integer (nullable = true)
 |-- year_built: integer (nullable = true)
 |-- gross_sqft: integer (nullable = true)
 |-- estimated_gross_income: integer (nullable = true)
 |-- gross_income_per_sqft: double (nullable = true)
 |-- estimated_expense: integer (nullable = true)
 |-- expense_per_sqft: double (nullable = true)
 |-- net_operating_income: integer (nullable = true)
 |-- full_market_value: integer (nullable = true)
 |-- market_value_per_sqft: double (nullable = true)
 |-- report_year: integer (nullable = true)
```

```
df_condo.count()
```

```
40
```

Rysunek 21: Typy danych w ramce mieszkań w Sparku i liczba wierszy

6 Podsumowanie finalnej wersji rozwiązania

Reasumując, udało nam się stworzyć system składający się z pięciu modułów komunikujących się ze sobą, które od pozyskania surowych dwóch zbiorów danych ze źródła dokonują ich transformacji, załadowania, składowania, analizy wsadowej, składowania agregacji oraz wizualizacji. System pozwala na wykonanie wygodnej analizy przekształconych już danych i wykonuje większość z tych funkcjonalności automatycznie (przez brak API ze stroną z danymi źródłowymi należy importować dane do folderu wejściowego dla surowych danych), sprawnie oraz, przede wszystkim, szybko dla rozważanych wolumenów danych. Na podstawie utworzonych przykładowych analiz można zaobserwować pewne trendy (bądź ich brak) i stanowią one podsumowanie wykonanych analiz.

7 Podział pracy w projekcie

Zadanie	Adam	Jan
Zaprojektowanie idei projektu	✓	✓
Utrzymanie projektu na GitHub	✓	✓
Dokumentacja Projektu	✓	✓
Pozyskanie danych	✓	✓
Automatyzacja przepływu danych (Nifi)	✓	
Składowanie danych Hive	✓	
Składowanie agregacji NoSql		✓
Wsadowa analiza danych		✓
Wizualizacje	✓	✓
Testy funkcjonalne	✓	✓
Prezentacja	✓	✓
Raport	✓	✓

Tabela 2: Podział obowiązków w projekcie