

3. sprawozdanie z laboratorium Hurtownie Danych

Mikołaj Kubś, 272662

25 marca 2025

1 Zadanie 1 - funkcje grupujące

1.1

Przygotować zestawienie przedstawiające, ile pieniędzy wydali klienci na zamówienia na przestrzeni poszczególnych lat. Wykonaj zestawienie przy użyciu poleceń rollup, cube, grouping sets.

CUBE:

```
1 SELECT
2     CASE
3         WHEN GROUPING(Person.BusinessEntityID) = 1 THEN '1. total sum'
4         ELSE CONCAT(MIN(Person.FirstName), ' ', MIN(Person.LastName))
5     END AS FullName,
6     YEAR(OrderDate) AS OrderYear,
7     SUM(TotalDue) AS TotalSales
8 FROM Sales.SalesOrderHeader
9 JOIN Sales.Customer ON Customer.CustomerID = SalesOrderHeader.CustomerID
10 JOIN Person.Person ON Person.BusinessEntityID = Customer.PersonID
11 GROUP BY
12     CUBE(Person.BusinessEntityID, YEAR(OrderDate))
13 ORDER BY
14     FullName, OrderYear
```

	FullName	OrderYear	TotalSales
1	1. total sum	NULL	123216786.1159
2	1. total sum	2011	14155659.325
3	1. total sum	2012	37675700.312
4	1. total sum	2013	48965887.9632
5	1. total sum	2014	22419488.3157
6	A. Leonetti	NULL	3400.8402
7	A. Leonetti	2013	1814.1819
8	A. Leonetti	2014	1586.6583
9	Aaron Adams	NULL	130.3458
10	Aaron Adams	2013	130.3458
11	Aaron Alexander	NULL	77.339
12	Aaron Alexander	2014	77.339
13	Aaron Allen	NULL	3756.989
14	Aaron Allen	2012	3756.989
15	Aaron Baker	NULL	1934.8329
16	Aaron Baker	2014	1934.8329
17	Aaron Bryant	NULL	148.0258
18	Aaron Bryant	2013	148.0258
19	Aaron Butler	NULL	16.9529

Rysunek 1: Wynik kwerendy 1.1: CUBE

ROLLUP:

```

1  SELECT
2      CASE
3          WHEN GROUPING(YEAR(OrderDate)) = 1 AND
4              GROUPING(Person.BusinessEntityID) = 1
5              THEN '1. total sum'
6          WHEN GROUPING(YEAR(OrderDate)) = 1 THEN
7              CONCAT(MIN(Person.FirstName), ' ', MIN(Person.LastName), ' (total)')
8          ELSE MIN(CONCAT(Person.FirstName, ' ', Person.LastName))
9      END AS FullName,
10     YEAR(OrderDate) AS OrderYear,
11     SUM(TotalDue) AS TotalSales
12 FROM Sales.SalesOrderHeader
13 JOIN Sales.Customer ON Customer.CustomerID = SalesOrderHeader.CustomerID
14 JOIN Person.Person ON Person.BusinessEntityID = Customer.PersonID
15 GROUP BY
16     ROLLUP(Person.BusinessEntityID, YEAR(OrderDate))
17 ORDER BY
18     FullName, OrderYear

```

	FullName	OrderYear	TotalSales
1	T. total sum	NULL	123216786.1159
2	A. Leonetti	2013	1814.1819
3	A. Leonetti	2014	1586.6583
4	A. Leonetti (total)	NULL	3400.8402
5	Aaron Adams	2013	130.3458
6	Aaron Adams (total)	NULL	130.3458
7	Aaron Alexander	2014	77.339
8	Aaron Alexander (total)	NULL	77.339
9	Aaron Allen	2012	3756.989
10	Aaron Allen (total)	NULL	3756.989
11	Aaron Baker	2014	1934.8329
12	Aaron Baker (total)	NULL	1934.8329
13	Aaron Bryant	2013	148.0258
14	Aaron Bryant (total)	NULL	148.0258
15	Aaron Butler	2014	16.5529
16	Aaron Butler (total)	NULL	16.5529
17	Aaron Campbell	2014	1276.8054
18	Aaron Campbell (total)	NULL	1276.8054
19	Aaron Carter	2014	44.1779

Query executed successfully. DESKTOP-2GLE4MP (15.0 RTM) DESKTOP-2GLE4MP\jantar... AdventureWorks2014 00:00:00 45 137 rows

Rysunek 2: Wynik kwerendy 1.1: ROLLUP

GROUPING SETS:

```

1 SELECT
2     MIN(CONCAT(Person.FirstName, ' ', Person.LastName)) AS FullName,
3     YEAR(OrderDate) AS OrderYear,
4     SUM(TotalDue) AS TotalDue
5 FROM Sales.SalesOrderHeader
6 JOIN Sales.Customer ON Customer.CustomerID = SalesOrderHeader.CustomerID
7 JOIN Person.Person ON Person.BusinessEntityID = Customer.PersonID
8 GROUP BY GROUPING SETS
9     (
10        (Person.BusinessEntityID),
11        (YEAR(OrderDate), Person.BusinessEntityID)
12    )
13 ORDER BY FullName, OrderYear

```

	FullName	OrderYear	TotalDue
1	A. Leonetti	2014	3400.8402
2	A. Leonetti	2013	1814.1819
3	A. Leonetti	2014	1586.6583
4	Aaron Adams	2013	130.3458
5	Aaron Adams	2013	130.3458
6	Aaron Alexander	2014	77.339
7	Aaron Alexander	2014	77.339
8	Aaron Allen	2012	3756.989
9	Aaron Allen	2012	3756.989
10	Aaron Baker	2014	1934.8329
11	Aaron Baker	2014	1934.8329
12	Aaron Bryant	2013	148.0258
13	Aaron Bryant	2013	148.0258
14	Aaron Butler	2014	16.5529
15	Aaron Butler	2014	16.5529
16	Aaron Campbell	2014	1276.8054
17	Aaron Campbell	2014	1276.8054
18	Aaron Carter	2014	44.1779
19	Aaron Carter	2014	44.1779

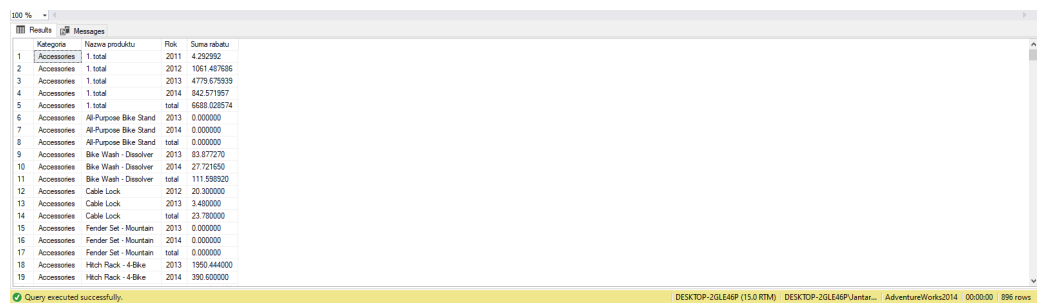
Query executed successfully. DESKTOP-2GLE4MP (15.0 RTM) DESKTOP-2GLE4MP\jantar... AdventureWorks2014 00:00:00 45 136 rows

Rysunek 3: Wynik kwerendy 1.1: GROUPING SETS

1.2

Przygotować zestawienie przedstawiające łączną kwotę zniżek z podziałem na kategorię, produkty oraz lata.

```
1 SELECT
2     ProductCategory.Name AS Kategoria,
3     ISNULL(Product.Name, '1. total') AS "Nazwa produktu",
4     ISNULL(CAST(YEAR(OrderDate) AS VARCHAR), 'total') AS Rok,
5     SUM(UnitPrice * OrderQty - LineTotal) AS "Suma rabatu"
6 FROM Sales.SalesOrderDetail
7 JOIN Sales.SalesOrderHeader ON SalesOrderHeader.SalesOrderID =
8     SalesOrderDetail.SalesOrderID
9 JOIN Production.Product ON SalesOrderDetail.ProductID = Product.ProductID
10 LEFT JOIN Production.ProductSubcategory ON
11     Product.ProductSubcategoryID = ProductSubcategory.ProductSubcategoryID
12 LEFT JOIN Production.ProductCategory ON
13     ProductSubcategory.ProductCategoryID = ProductCategory.ProductCategoryID
14 GROUP BY
15     CUBE(Product.Name, YEAR(OrderDate)),
16     ProductCategory.Name
17 ORDER BY
18     Kategoria, "Nazwa produktu", Rok
```



Kategoria	Nazwa produktu	Rok	Suma rabatu
Accessories	1. total	2011	4.252092
Accessories	1. total	2012	1061.687636
Accessories	1. total	2013	4779.679339
Accessories	1. total	2014	842.571957
Accessories	1. total	total	6688.620574
Accessories	Al-Purpose Bike Stand	2013	0.000000
Accessories	Al-Purpose Bike Stand	2014	0.000000
Accessories	Al-Purpose Bike Stand	total	0.000000
Accessories	Bike Wash - Descolver	2013	83.877270
Accessories	Bike Wash - Descolver	2014	27.721650
Accessories	Bike Wash - Descolver	total	111.598920
Accessories	Cable Lock	2012	20.300000
Accessories	Cable Lock	2013	3.490000
Accessories	Cable Lock	total	23.790000
Accessories	Fender Set - Mountain	2013	0.000000
Accessories	Fender Set - Mountain	2014	0.000000
Accessories	Fender Set - Mountain	total	0.000000
Accessories	Hitch Rack - 4-Bike	2013	1550.444000
Accessories	Hitch Rack - 4-Bike	2014	390.600000

Rysunek 4: Wynik kwerendy 1.2

2 Zadanie 2 - funkcje okienkowe

2.1

Dla kategorii 'Bikes' przygotuj zestawienie prezentujące procentowy udział kwot sprzedaży produktów tej kategorii w poszczególnych latach w stosunku

do łącznej kwoty sprzedaży dla tej kategorii. W zadaniu wykorzystaj funkcję okna. Wykonaj podobne zestawienia dla pozostałych kategorii.

```

1 SELECT
2     ISNULL(ProductCategory.Name, 'Total') AS "Nazwa kategorii",
3     YEAR(SalesOrderHeader.OrderDate) AS Rok,
4     SUM(SalesOrderDetail.LineTotal) AS "Suma sprzedaży",
5     CAST(ROUND(SUM(SalesOrderDetail.LineTotal) / SUM(SUM(SalesOrderDetail.LineTotal))
6         OVER (PARTITION BY ProductCategory.Name) * 100, 2) AS DECIMAL(10,2))
7     AS "Procent sprzedaży dla kategorii"
8 FROM
9     Sales.SalesOrderDetail
10    JOIN Sales.SalesOrderHeader ON
11        SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesOrderID
12    JOIN Production.Product ON
13        SalesOrderDetail.ProductID = Product.ProductID
14    JOIN Production.ProductSubcategory ON
15        Product.ProductSubcategoryID = ProductSubcategory.ProductSubcategoryID
16    JOIN Production.ProductCategory ON
17        ProductSubcategory.ProductCategoryID = ProductCategory.ProductCategoryID
18 GROUP BY
19     YEAR(SalesOrderHeader.OrderDate),
20     CUBE(ProductCategory.Name)
21 ORDER BY
22     COALESCE(ProductCategory.Name, 'Total') DESC,
23     YEAR(SalesOrderHeader.OrderDate);

```

	Nazwa kategorii	Rok	Suma sprzedaży	Procent sprzedaży dla kategorii
1	Total	2011	12641672.212954	11.51
2	Total	2012	33524261.324434	30.52
3	Total	2013	43622479.091635	39.71
4	Total	2014	20057928.810865	18.26
5	Components	2011	639173.040500	5.42
6	Components	2012	3880757.952985	32.88
7	Components	2013	5612335.340965	47.56
8	Components	2014	1669726.951980	14.15
9	Clothing	2011	36031.475346	1.70
10	Clothing	2012	555587.706540	26.20
11	Clothing	2013	1067685.698497	50.35
12	Clothing	2014	461233.644918	21.75
13	Bikes	2011	11945646.924000	12.62
14	Bikes	2012	28985915.844695	30.62
15	Bikes	2013	36265625.361212	38.32
16	Bikes	2014	17453180.574824	18.44
17	Accessories	2011	20620.773108	1.64
18	Accessories	2012	102439.820714	8.05
19	Accessories	2013	675024.650961	53.06

Rysunek 5: Wynik kwerendy 2.1

2.2

Przygotuj zestawienie dla sprzedawców z podziałem na lata i miesiące prezentujące liczbę obsłużonych przez nich zamówień w ciągu roku, w ciągu roku

narastająco oraz sumarycznie w obecnym i poprzednim miesiącu. W zadaniu wykorzystaj funkcje okna.

```
1  SELECT
2      Person.FirstName + ' ' + Person.LastName AS [Imie i nazwisko],
3      YEAR(SalesOrderHeader.OrderDate) AS [Rok],
4      MONTH(SalesOrderHeader.OrderDate) AS [Miesiac],
5      COUNT(*) AS [W miesiacu],
6      SUM(COUNT(*)) OVER
7          (PARTITION BY
8              Person.BusinessEntityID,
9              YEAR(SalesOrderHeader.OrderDate))
10     AS [W roku],
11     SUM(COUNT(*)) OVER
12         (PARTITION BY
13             Person.BusinessEntityID,
14             YEAR(SalesOrderHeader.OrderDate)
15             ORDER BY MONTH(SalesOrderHeader.OrderDate))
16     AS [W roku narastajaco],
17     COUNT(*) + LAG(COUNT(*), 1, 0) OVER
18         (PARTITION BY Person.BusinessEntityID
19             ORDER BY YEAR(SalesOrderHeader.OrderDate), MONTH(SalesOrderHeader.OrderDate))
20     AS [Obecny i poprzedni miesiac]
21 FROM
22     Sales.SalesOrderHeader
23     JOIN Sales.SalesPerson ON
24         SalesOrderHeader.SalesPersonID = SalesPerson.BusinessEntityID
25     JOIN Person.Person ON
26         SalesPerson.BusinessEntityID = Person.BusinessEntityID
27 GROUP BY
28     Person.BusinessEntityID,
29     Person.FirstName,
30     Person.LastName,
31     YEAR(SalesOrderHeader.OrderDate),
32     MONTH(SalesOrderHeader.OrderDate)
33 ORDER BY
34     [Imie i nazwisko],
35     [Rok],
36     [Miesiac];
```

	Imię i nazwisko	Rok	Miesiąc	W miesiącu	W roku	W roku następującym	Obecny i poprzedni miesiąc
1	Amy Alberts	2012	6	3	7	3	3
2	Amy Alberts	2012	9	2	7	5	5
3	Amy Alberts	2012	12	2	7	7	4
4	Amy Alberts	2013	1	1	29	1	3
5	Amy Alberts	2013	2	1	29	2	2
6	Amy Alberts	2013	3	1	29	3	2
7	Amy Alberts	2013	4	2	29	5	3
8	Amy Alberts	2013	5	1	29	6	3
9	Amy Alberts	2013	6	5	29	11	6
10	Amy Alberts	2013	7	3	29	14	8
11	Amy Alberts	2013	9	1	29	15	4
12	Amy Alberts	2013	9	4	29	19	5
13	Amy Alberts	2013	10	4	29	23	8
14	Amy Alberts	2013	11	1	29	24	5
15	Amy Alberts	2013	12	5	29	29	6
16	Amy Alberts	2014	1	1	3	1	6
17	Amy Alberts	2014	2	1	3	2	2
18	Amy Alberts	2014	3	1	3	3	2
19	David Campbell	2011	5	5	28	5	5

Rysunek 6: Wynik kwerendy 2.2

2.3

Przygotuj ranking klientów w zależności od liczby zakupionych produktów. Porównaj rozwiązania uzyskane przez funkcje rank i dense_rank.

RANK:

```

1 SELECT
2     Customer.CustomerID AS "ID",
3     Person.FirstName AS "Imie",
4     Person.LastName AS "Nazwisko",
5     COUNT(*) AS "Liczba transakcji",
6     RANK() OVER(ORDER BY COUNT(*) DESC) AS "Ranking"
7 FROM Sales.Customer
8 RIGHT JOIN Sales.SalesOrderHeader ON
9     SalesOrderHeader.CustomerID = Customer.CustomerID
10 RIGHT JOIN Sales.SalesOrderDetail ON
11     SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesOrderID
12 JOIN Person.Person ON Person.BusinessEntityID = Customer.PersonID
13 GROUP BY Customer.CustomerID, Person.FirstName, Person.LastName
14 ORDER BY COUNT(*) DESC

```

ID	Imie	Nazwisko	Liczba transakcji	Ranking
29722	Reuben	D'Sa	530	1
29666	Richard	Lum	482	2
29614	Ryan	Calafato	451	3
29990	Yale	Li	446	4
30048	Marcia	Sultan	441	5
29712	Holly	Dickson	440	6
29705	Della	Dennett Jr	436	7
30117	Robert	Vespa	436	7
29992	Sandra	Maynard	432	9
29639	Joseph	Castelluccio	429	10
29716	Blaine	Dockter	422	11
29744	John	Evans	418	12
30103	Mandy	Vance	405	13
29996	Kate	McKulik-White	401	14
29957	Kevin	Liu	401	14
29637	Donna	Cameras	394	16
29523	John	Arthur	393	17
30107	Margaret	Vanderkamp	391	18
29734	Jauna	Ellson	387	19

Rysunek 7: Wynik kwerendy 2.3: RANK

DENSE_RANK:

```

1 SELECT
2     Customer.CustomerID AS "ID",
3     Person.FirstName AS "Imie",
4     Person.LastName AS "Nazwisko",
5     COUNT(*) AS "Liczba transakcji",
6     DENSE_RANK() OVER(ORDER BY COUNT(*) DESC) AS "Ranking"
7 FROM Sales.Customer
8 RIGHT JOIN Sales.SalesOrderHeader ON
9     SalesOrderHeader.CustomerID = Customer.CustomerID
10 RIGHT JOIN Sales.SalesOrderDetail ON
11     SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesOrderID
12 JOIN Person.Person ON Person.BusinessEntityID = Customer.PersonID
13 GROUP BY Customer.CustomerID, Person.FirstName, Person.LastName
14 ORDER BY COUNT(*) DESC

```

ID	Imie	Nazwisko	Liczba transakcji	Ranking
29722	Reuben	D'Sa	530	1
29666	Richard	Lum	482	2
29614	Ryan	Calafato	451	3
29990	Yale	Li	446	4
30048	Marcia	Sultan	441	5
29712	Holly	Dickson	440	6
29705	Della	Dennett Jr	436	7
30117	Robert	Vespa	436	7
29992	Sandra	Maynard	432	8
29639	Joseph	Castelluccio	429	9
29716	Blaine	Dockter	422	10
29744	John	Evans	418	11
30103	Mandy	Vance	405	12
29996	Kate	McKulik-White	401	13
29957	Kevin	Liu	401	13
29637	Donna	Cameras	394	14
29523	John	Arthur	393	15
30107	Margaret	Vanderkamp	391	16
29734	Jauna	Ellson	387	17

Rysunek 8: Wynik kwerendy 2.3: DENSE_RANK

2.4

Przygotuj ranking produktów w zależności od średniej liczby sprzedanych sztuk. Wyróżnij 3 (prawie równoliczne) grupy produktów: sprzedających się najlepiej, średnio i najslabiej.

```
1 WITH ProductsRanked AS
2 (
3     SELECT
4         Product.ProductID AS "ID produktu",
5         Product.Name AS "Nazwa produktu",
6         SUM(SalesOrderDetail.OrderQty) AS "Suma liczby sprzedanych sztuk",
7         AVG(CAST(SalesOrderDetail.OrderQty AS DECIMAL(10,2)))
8         AS "Średnia liczba sprzedanych sztuk",
9         NTILE(3) OVER (ORDER BY AVG(CAST(SalesOrderDetail.OrderQty AS FLOAT)) DESC)
10        AS "Numer grupy"
11    FROM Production.Product
12    JOIN Sales.SalesOrderDetail
13        ON SalesOrderDetail.ProductID = Product.ProductID
14    GROUP BY Product.ProductID, Product.Name
15 )
16 SELECT
17     ProductsRanked.[ID produktu],
18     ProductsRanked.[Nazwa produktu],
19     ProductsRanked.[Suma liczby sprzedanych sztuk],
20     ProductsRanked.[Średnia liczba sprzedanych sztuk],
21     CASE
22         WHEN "Numer grupy" = 1 THEN 'Najlepiej sprzedające się'
23         WHEN "Numer grupy" = 2 THEN 'Średnio sprzedające się'
24         WHEN "Numer grupy" = 3 THEN 'Najslabiej sprzedające się'
25     END AS SalesCategory
26 FROM ProductsRanked
27 ORDER BY ProductsRanked.[Średnia liczba sprzedanych sztuk] DESC
```

ID produktu	Nazwa produktu	Suma liczby sprzedanych sztuk	Średnia liczba sprzedanych sztuk	SaleCategory
1	863 Full-Finger Gloves, L	3378	9.280219	Najlepiej sprzedające się
2	854 Classic Vest, S	4247	6.227272	Najlepiej sprzedające się
3	862 Full-Finger Gloves, M	2205	6.144646	Najlepiej sprzedające się
4	806 ML Headset	659	5.990909	Najlepiej sprzedające się
5	709 Mountain Bike Socks, M	1107	5.888287	Najlepiej sprzedające się
6	867 Women's Mountain Shorts, S	3236	5.084119	Najlepiej sprzedające się
7	862 Women's Tights, S	2072	4.959952	Najlepiej sprzedające się
8	854 Women's Tights, L	2123	4.814058	Najlepiej sprzedające się
9	869 Women's Mountain Shorts, L	3244	4.588401	Najlepiej sprzedające się
10	856 Men's Bib Shorts, M	1616	4.577933	Najlepiej sprzedające się
11	849 Men's Sports Shorts, M	1276	4.506533	Najlepiej sprzedające się
12	884 Short-Sleeve Classic Jersey, XL	3864	4.274336	Najlepiej sprzedające się
13	844 Minipump	1130	4.232209	Najlepiej sprzedające się
14	843 Cable Lock	1087	4.180769	Najlepiej sprzedające się
15	855 Classic Vest, M	2284	4.115315	Najlepiej sprzedające się
16	875 Racing Socks, L	2473	4.094370	Najlepiej sprzedające się
17	715 Long-Sleeve Logo Jersey, L	6592	4.031804	Najlepiej sprzedające się
18	876 Hitch Rack - 4-Bike	3166	3.977386	Najlepiej sprzedające się
19	909 ML Mountain Seat/Saddle	534	3.865955	Najlepiej sprzedające się

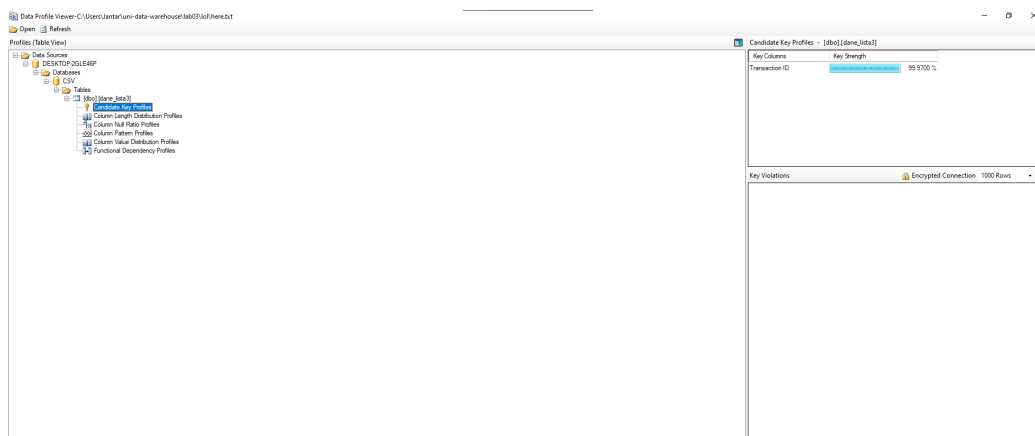
Rysunek 9: Wynik kwerendy 2.4

ID produktu	Nazwa produktu	Suma liczby sprzedanych sztuk	Średnia liczba sprzedanych sztuk	SaleCategory
83	738 LL Road Frame - Black, 52	1581	2.635000	Najlepiej sprzedające się
84	524 LL Mountain Frame - Black, 42	423	2.611111	Najlepiej sprzedające się
85	874 Racing Socks, M	1547	2.608763	Najlepiej sprzedające się
86	773 Mountain-100 Silver, 44	601	2.579399	Najlepiej sprzedające się
87	973 Road-350-W Yellow, 40	1477	2.573661	Najlepiej sprzedające się
88	754 Road-450 Red, 58	562	2.575221	Najlepiej sprzedające się
89	957 Touring-1000 Yellow, 60	1114	2.572748	Najlepiej sprzedające się
90	880 Hydration Pack - 70 oz.	2761	2.570763	Średnio sprzedające się
91	940 HL Road Pedal	676	2.560606	Średnio sprzedające się
92	969 Touring-1000 Blue, 50	1120	2.551252	Średnio sprzedające się
93	836 ML Road Frame-W - Yellow, 48	889	2.540000	Średnio sprzedające się
94	778 Mountain-100 Black, 48	616	2.534979	Średnio sprzedające się
95	764 Road-650 Red, 52	1112	2.527272	Średnio sprzedające się
96	894 LL Bottom Bracket	378	2.520000	Średnio sprzedające się
97	822 ML Road Frame-W - Yellow, 38	895	2.514044	Średnio sprzedające się
98	899 LL Touring Frame - Yellow, 44	294	2.512820	Średnio sprzedające się
99	895 LL Touring Frame - Blue, 50	316	2.507936	Średnio sprzedające się
100	877 Bike Wash - Dissolver	3319	2.501130	Średnio sprzedające się
101	939 ML Road Pedal	661	2.494239	Średnio sprzedające się

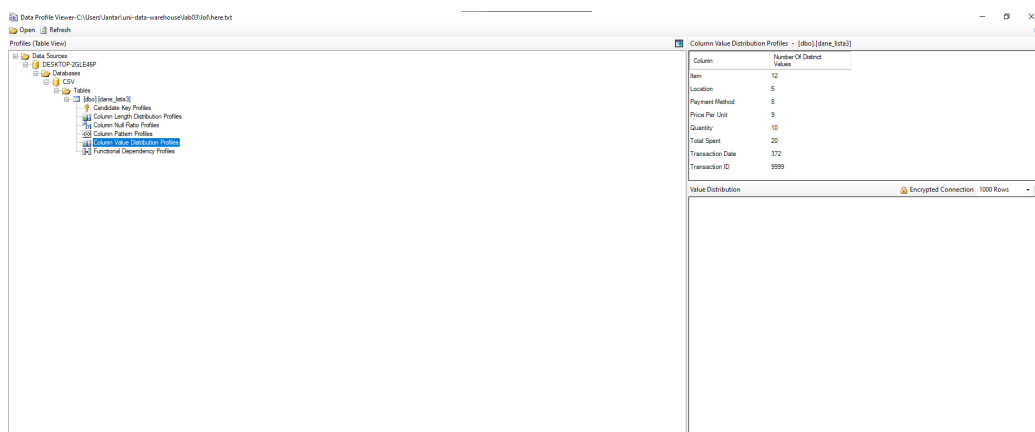
Rysunek 10: Widok przejścia do części średnio sprzedającej się

3 Zadanie 3 - profilowanie danych

Po próbach analizy w SSIS uzyskano tylko część rezultatów.



Rysunek 11: Profil kolumny kandydującej



Rysunek 12: Liczba unikatowych wartości w kolumnach

Z powodu problemów technicznych resztę analizy przeprowadzono używając *Pythona* oraz bibliotek *Pandas* i *ydata_profiling*.

```

1 import pandas as pd
2 from ydata_profiling import ProfileReport
3 from os import path
4
5 dir_path: str = path.dirname(path.realpath(__file__))
6
7 df: pd.DataFrame = pd.read_csv(path.join(dir_path, "dane_lista3.csv"))
8

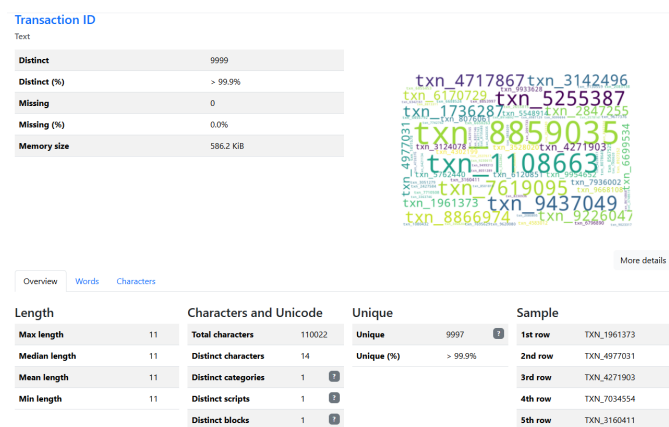
```

```

9  profile = ProfileReport(df, explorative=True)
10
11  profile.to_file(path.join(dir_path, "python_results.html"))
12
13  df['Transaction ID'] = df['Transaction ID'].astype('str')
14  df['Item'] = df['Item'].astype('str')
15  df['Quantity'] = pd.to_numeric(df['Quantity'], errors='coerce')
16  df['Price Per Unit'] = pd.to_numeric(df['Price Per Unit'], errors='coerce')
17  df['Total Spent'] = pd.to_numeric(df['Total Spent'], errors='coerce')
18  df['Payment Method'] = df['Payment Method'].astype('str')
19  df['Location'] = df['Location'].astype('str')
20  df['Transaction Date'] = pd.to_datetime(
21      df['Transaction Date'], errors='coerce')
22
23  profile = ProfileReport(df, explorative=True)
24
25  profile.to_file(path.join(dir_path, "python_results_good_data_types.html"))

```

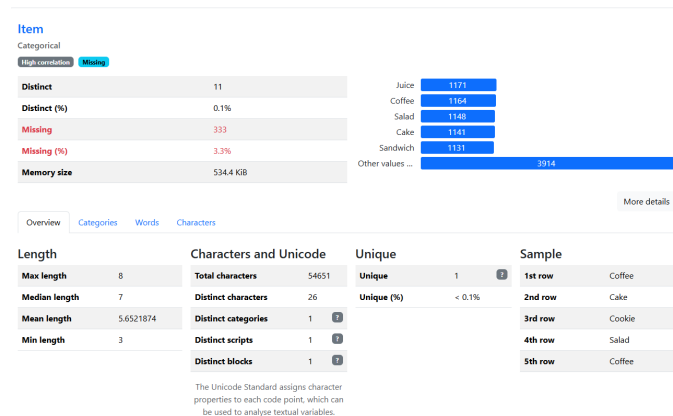
3.1 Transaction ID



Rysunek 13: Profil kolumny *Transaction ID*

Tak jak też wywnioskowano wcześniej, jest to jedyna kolumna nadająca się na klucz kandydujący. Są w niej tylko 3 duplikaty. Zawsze ma też tę samą długość - 11 znaków.

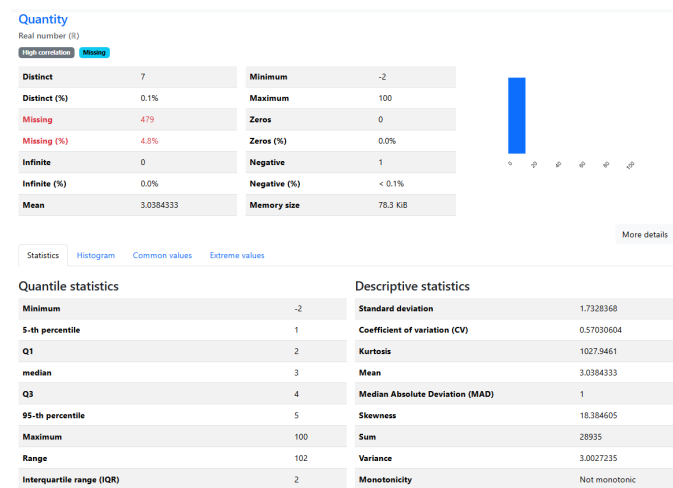
3.2 Item



Rysunek 14: Profil kolumny *Item*

Kolumna jest kategoriowa i przypisuje transakcjom kategorię kupionego towaru. W 3.3% wierszy brakuje wartości. Dodatkowo w 3.4% to *UNKNOWN*.

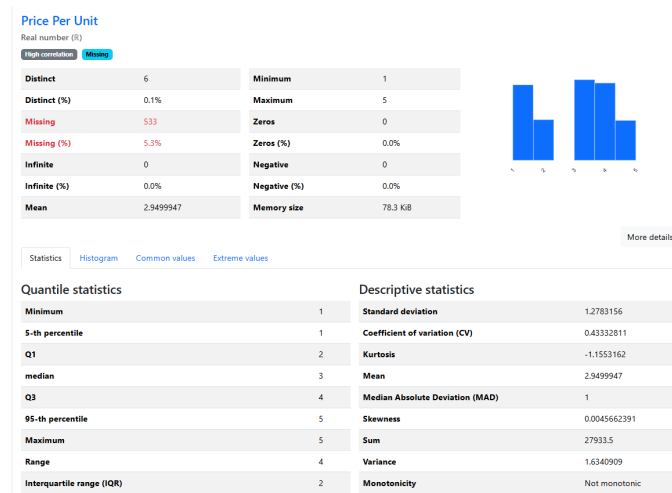
3.3 Quantity



Rysunek 15: Profil kolumny *Quantity*

W kolumnie występują anomalnie. Zdarzyła się raz wartość ujemna: -2. Zdarzyła się raz wartość 100, znacznie przekraczająca poza zakres innych (1-5), ale być może dozwolona. Brakuje danych w 4.8% wierszy. Wartości standardowe, czyli od 1 do 5 włącznie, występują z podobną częstością.

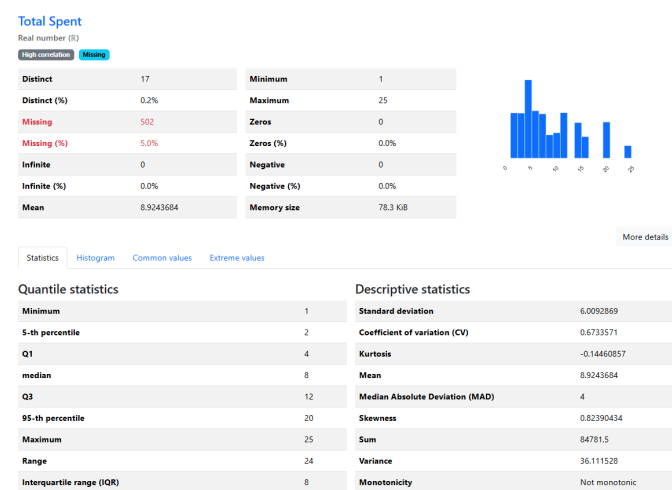
3.4 Price Per Unit



Rysunek 16: Profil kolumny *Price Per Unit*

Kolumna w większości składa się z liczb całkowitych, ale 11.3% wartości to jedyne z wartością po przecinku. Wszystkie wynoszą dokładnie 1,5. Brakuje wartości dla 5.3%. Standardowe i jedyne poza brakującymi wartości to liczby całkowite od 1-5 oraz 1,5. Najczęściej występuje wartość 3. Występowanie wartości 1,5 jest raczej jednak jak najbardziej dozwolone.

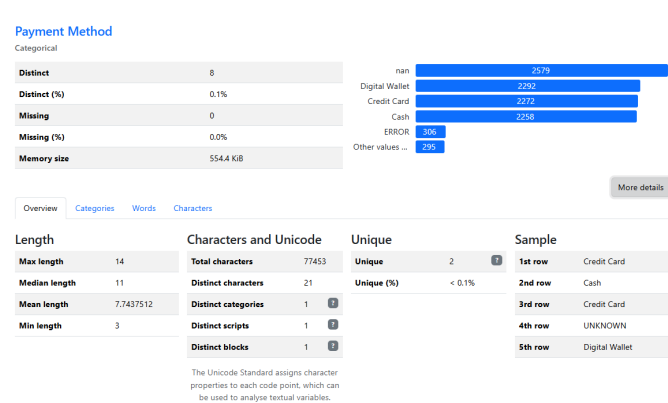
3.5 Total Spent



Rysunek 17: Profil kolumny *Total Spent*

Brakuje 5.02% wartości. Przedział to liczby od 1 do 25 (ale tylko 17 z nich jest w danych). Większość wartości to liczby całkowite, ale występują też wielokrotności 1,5, sugerując, że cena 1,5 w Price Per Unit nie jest żadną anomalią. Najczęściej występuje wartość 6. Większość wartości jest bliżej początku rozkładu niż końca.

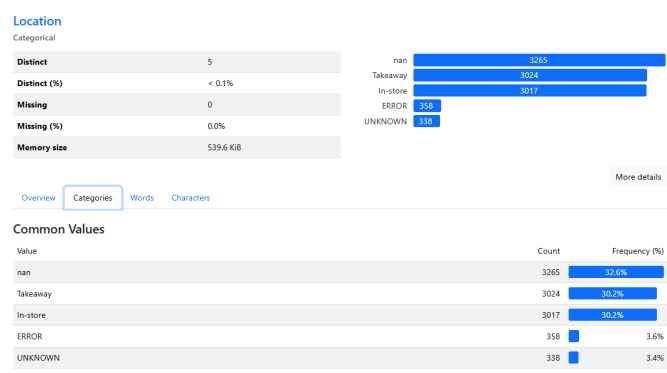
3.6 Payment Method



Rysunek 18: Profil kolumny *Payment Method*

Niepoprawne to 31,78% całości - połączenie *nan*, *ERROR* i *UNKNOWN*. Oczywiście przy odczytywaniu karty mogą zdarzać się błędy, ale wtedy transakcje raczej powinny być odrzucane i nie znajdować się w bazie danych. Dodatkowo 2 razy wystąpiły literówki - Digital Walle i CreditCard zamiast Digital Wallet i Credit Card (występujących znacznie częściej). Poprawne kategorie to Digital Wallet, Credit Card i Cash.

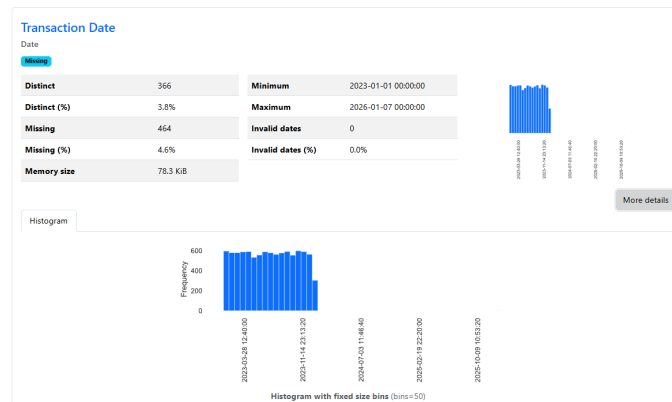
3.7 Location



Rysunek 19: Profil kolumny *Location*

Podobnie jak poprzednio, wielu danych brakuje. Nan, ERROR i UNKNOWN występują z częstością 39,61%. Poza tym, 2 poprawne kategorie to Takeaway lub In-store.

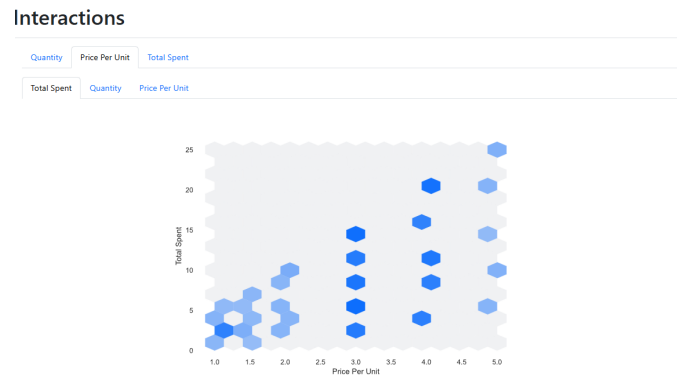
3.8 Transaction Date



Rysunek 20: Profil kolumny *Transaction Date*

Przez jedną literówkę w danych zakres wydaje się być od 2023-01-01 do 2026-01-07. Rzeczywiście jednak dane są od 2023-01-01 do 2023-12-31, a wystąpiła literówka w 2026 i rok powinien zostać zmieniony na 2023. Poza tym rozkład jest dość równomierny, jedynie w grudniu jest wyraźny spadek. Brakuje 4,6% wartości.

3.9 Przykład interakcji



Rysunek 21: Przykład interakcji między Price Per Unit a Total Spent

Jak widać, występuje pozytywna korelacja między Price Per Unit a Total Spent.

4 Wnioski

4.1 Wnioski z zadań 1

Klienci bardzo różnią się między sobą. Niektórzy wydają tylko w jednym roku niskie sumy, inni wydają na przestrzeni lat duże pieniądze. Ostatecznie sprowadza się to do wysokich przychodów firmy, rosnących między 2011-2013. Jest szansa na utrzymanie tego trendu w 2014 roku.

Wiele produktów nigdy nie miało rabatu. Najwięcej rabatów udzielono w 2013 roku, ale ta wartość i tak jest niska, biorąc pod uwagę przychody firmy.
TODO TUTAJ

4.2 Wnioski z zadań 2