Język makr: podstawy

Ćwiczenia

Język makr: podstawy został przygotowany przez Stacey Syphus, Mark Jordan, Kathy Passarella i Joannę Nawalany. Wsparcie w zakresie projektowania, edycji i produkcji materiałów zapewnił zespół *Learning Design and Development*.

Język makr: podstawy Course Notes

Kod kursu: MC1V2P, data przygotowania 04lip2022. MC1V2P_001

Spis treści

Aby dowiedzieć się więcej…

Aby uzyskać informacje o innych szkoleniach w programie prosimy o kontakt z Centrum Szkoleniowym SAS pod numerem 22 560 46 00 lub mailowo na adres CS@spl.sas.com.

Informacje dotyczące szkoleń oraz katalog szkoleń można znaleźć na stronie internetowej http://www.sas.com/poland/szkolenia

Rozdział 2 Funkcjonalność języka makr w SAS

2.1 Tworzenie i używanie makrozmiennych

Poziom 1

1. Definiowanie i wykorzystanie makrozmiennych do zastępowania tekstów

a. Otwórz **m102p01.sas z** folderu **practices**. Uruchom program, zobacz log i wyniki. Sprawdź, że tytuł to **Klienci z US w wieku od 18 do 24** i że 127 wierszy zostało odczytanych.

b. Utwórz makrozmienną o nazwie **Country** i przypisz jej wartość *US*. Zastąp wszystkie wystąpienia *US* na odwołania do makrozmiennej **Country**. Wykonaj program i sprawdź, że wyniki są takie same jak w punkcie a.

c. Zmodyfikuj wartość **Country** na *FR*. Uruchom ponownie program i sprawdź, że tytuł to **Klienci z FR w wieku od 18 do 24** i że zostało odczytanych 46 wierszy.

d. Stwórz dodatkowe makrozmienne **Age1** i **Age2**. Przypisz wartości **Age1** *25*, **Age2** *34*, a **Country** *AU*. Zastąp wszystkie wystąpienia 18 i 24 odwołaniami do makrozmiennych **Age1** i **Age2**. Uruchom program.

- Jaki jest tytuł raportu?
- Ile obserwacji zostało przeczytanych?

Poziom 2

2. Użycie odwołania do makrozmiennych i separatorów

a. Otwórz **m102p02.sas** z folderu **practices**. Uruchom program i zobacz wyniki. Sprawdź, że raport zawiera 17 wierszy.

b. Utwórz makrozmienne **Lib**, **Dsn**, **Var** i przypisz im wartości odpowiednio *mc1*, *newhires* i *Employee.* Zmodyfikuj program tak, aby każde wystąpienie *mc1*, *newhires*, i *Employee* zamienić na odwołania do odpowiednich makrozmiennych. Uruchom program.

Uwaga: Upewnij się, że 's' w słowie *Employees* pojawia się w treści tytułu.

- Jaki jest tytuł raportu?

Ile wierszy jest w raporcie?

Koniec ćwiczeń

Rozdział 3 Przechowywanie i przetwarzanie tekstu

3.1 Makrofunkcje

Poziom 1

1. Użycie funkcji %UPCASE i %SCAN

a. Otwórz **m103p01.sas** z folderu **practices**. Dodaj instrukcję %LET, żeby zmienić wartości makrozmiennej **FullName** na duże litery i przypisać wynik do makrozmiennej **FullName**. Napisz jedną instrukcję %PUT, żeby pokazać bieżącą wartość makrozmiennej **FullName** oraz jej wartość po modyfikacji, jak poniżej. Uruchom program i zobacz log, żeby sprawdzić wyniki.

Uwaga: Wykorzystaj makrofunkcję %SYSFUNC, żeby wykonać funkcję PROPCASE.

```
JAN KOWALSKI w poprawnej pisowni to Jan Kowalski.
```

b. Zmodyfikuj i uruchom program.

- Dodaj instrukcję %LET, żeby odczytać imię z makrozmiennej **FullName**, przekonwertować je na odpowiednią wielkość liter i przypisać wynik do makrozmiennej **First**.
- Dodaj kolejną instrukcję %LET, żeby odczytać nazwisko z makrozmiennej **FullName**, przekonwertować je na odpowiednią wielkość liter i przypisać wynik do makrozmiennej **Last**.
- Wyświetl w logu wartości makrozmiennych **FullName**, **First** i **Last** jak pokazano poniżej.

```
FULLNAME=JAN KOWALSKI FIRST=Jan LAST=Kowalski
```

c. Dodaj instrukcję %SYMDEL, żeby usunąć makrozmienne **FullName**, **First**, and **Last** z globalnej tablicy symboli. Wykorzystaj instrukcję %PUT, żeby wypisać w logu wartości wszystkich makrozmiennych zdefiniowanych przez użytkownika. Uruchom dwie pierwsze instrukcje i sprawdź, że makrozmienne **FullName**, **First** i **Last** nie istnieją.

Poziom 2

2. Używanie makrofunkcji do maskowania znaków

a. Otwórz **m103p02.sas** z folderu **practices** i zobacz kod. Uruchom program i zobacz wyniki. Zwróć uwagę, że stopka zawiera datę i czas, kiedy sesja SASa została uruchomiona.

b. Utwórz makrozmienną **Product** i przypisz jej wartość *R&D*. Odwołaj się do makrozmiennej **Product** w instrukcjach TITLE i WHERE, zamieniając wartość *Jacket*. Zmodyfikuj instrukcję FOOTNOTE, żeby wyświetlić bieżącą datę i czas, używając odpowiednio formatów DATE9 i TIMEAMPM9.Uruchom program i sprawdź, że tytuł to *Product Names Containing 'R&D'* i że w stopce wyświetlana jest bieżąca data i godzina.

- Ile wierszy jest w raporcie?

Wyzwanie

3. Używanie makrofunkcji do wykonywania obliczeń

a. Przypisz swoją datę urodzenia w formie dzień nazwa miesiąca i rok (na przykład, 01Jan1990) na makrozmienną **Birthdate**. Użyj funkcji %SYSEVALF do wyliczenia Twojego wieku poprzez odjęcie Twojej daty urodzenia od daty bieżącej i podzieleniu przez 365.25. Wypisz wyniki w logu.

Na przykład, jeżeli Twoja data urodzenia to 1.01.1990, a dzisiaj jest 03.03.2019, wynik będzie jak poniżej.

Wskazówka: Odwołaj się do makrozmiennej **Birthdate** jak do stałej daty SAS-owej (**"&birthdate"d**).

Uwaga: Wiek zależy od bieżącej daty.

```
Mój wiek to 29.2183436002737
```

b. Wykorzystaj drugi argument funkcji %SYSEVALF, żeby zwrócić liczbę całkowitą dla wyliczanego wieku. Wypisz wyniki w logu.

Na przykład, jeżeli Twoje urodziny są 1.01.1990, a dzisiaj jest 03.04.2019, wynik będzie jak poniżej.

```
Mój wiek to 29
```

Koniec ćwiczeń

3.2 Używanie SQL do tworzenia makrozmiennych

Poziom 1

4. Generowanie makrozmiennych za pomocą PROC SQL do wykorzystania w tytule raportu

a. Otwórz **m103p04.sas** z folderu **practices**. Zobacz kod i uruchom instrukcje %LET oraz PROC SQL. Sprawdź, czy **Qty** (średnia dla **Quantity**, czyli liczby produktów) to 1.43, a **Price** (średnia dla **Total_Retail_Price**, czyli ceny) to 137.72.

b. W instrukcji TITLE2 zamień *xxx* na średnią dla **Quantity** (1.43), a *yyy* na średnią dla **Total_Retail_Price** (137.72). Uruchom instrukcję TITLE i PROC PRINT, zobacz log i wyniki.

- Ile wierszy jest w tabeli wejściowej?

c. W PROC SQL dodaj wyrażenie INTO, żeby przypisać średnią dla **Quantity** na makrozmienną **Qty**, a średnią dla **Total_Retail_Price** na makrozmienną **Price**. W instrukcji TITLE2 zamień na sztywno wpisane wartości średnich na odwołania do **Qty** i **Price**.

Uruchom cały program, zobacz log i wyniki. Sprawdź, że tytuł został poprawnie wyświetlony.

- Ile wierszy zostało wczytanych z tabeli wejściowej?
- Dlaczego średnia cena jest wyświetlona w tytule ze znakiem dolara na początku?

d. Zmodyfikuj instrukcję %LET, żeby przypisać *01Feb2019* na start i *28Feb2019* na stop. Ponownie uruchom program.

- Ile wierszy zostało wczytanych z tabeli wejściowej?
- Jakie wartości zostały podstawione za **qty** i **price** w instrukcji TITLE2?

Poziom 2

5. Generowanie makrozmiennych za pomocą PROC SQL do wykorzystania w kolejnych krokach

a. Otwórz **m103p05.sas** z folderu **practices**. Zobacz i uruchom kod w części a. Sprawdź, że średnia prędkość wiatru jest równa 79.

b. Zobacz kod i części b. Zamień każde wystąpienie *XX* średnią prędkością wiatru z kroku a. Uruchom kod z części b i zobacz wyniki.

- Ile wierszy jest w tabeli?
- Który słupek ma największą częstotliwość?

c. Zmodyfikuj PROC SQL.

- Usuń generowanie raportu przez PROC SQL.
- Zapisz wyliczoną wartość w makrozmiennej o nazwie **AvgWind** bez wiodących spacji.
- Dodaj kolejną instrukcję SELECT tak, by wybrać BasinName ze zbioru mc1.storm_basin_codes gdzie **Basin** jest równe makrozmiennej **basincode**. Zapisz wartość na makrozmienną o nazwie **BasinName**.

d. Zmodyfikuj kod.

- Zamień na sztywno wpisaną wartość *79* na odwołanie do **AvgWind**.
- Zamień wszystkie na sztywno wpisane wartości *North Atlantic* na odwołania do **BasinName**.
- Uruchom zmodyfikowany kod i sprawdź, że raport zawiera te same informacje co raport wygenerowany w części b.

e. Zmodyfikuj instrukcję %LET, żeby przypisać *2015* na **Year** i *EP* na **BasinCode**. Uruchom cały program, włącznie z instrukcją %LET i PROC SQL. Zobacz log i upewnij się, że nie ma żadnych błędów ani ostrzeżeń. Sprawdź, że tytuł raportu to *East Pacific Storms in 2015 Season Max Wind > Season Average of 92 MPH*.

- Ile wierszy jest w tabeli?
- Który słupek ma największą częstotliwość?

Koniec ćwiczeń

3.3 Wykorzystanie kodu DATA step do tworzenia makrozmiennych

Poziom 1

6. Tworzenie makrozmiennych za pomocą rutyny SYMPUTX

a. Otwórz **m103p06.sas** z folderu **practices**. Zobacz i uruchom program. Sprawdź, że tytuł raportu to *Nowi pracownicy działu: Administration* oraz że suma **Salary** wynosi $221,618.

b. Dodaj instrukcję %LET, żeby przypisać wartość *Administration* do nowej makrozmiennej **dept** i zamień każde wystąpienie *Administration* na odwołanie do **dept**. Uruchom zmodyfikowany program i sprawdź, że tytuł i raport są takie same, jak w części a.

c. Zmień wartość makrozmiennej **dept** na *Sales* i uruchom program. Sprawdź, że tytuł raportu to *Nowi pracownicy działu: Sales*.

- Jaka jest suma zmiennej **Salary**?

d. Zmodyfikuj DATA step tak, żeby stworzyć makrozmienną **avg**, która będzie zawierać średnią pensję wyliczoną w ostatniej iteracji.
Wskazówka: Użyj funkcji PUT i formatu DOLLAR9. przy przypisywaniu wartości do **avg**.

Dodaj instrukcję FOOTNOTE przed PROC PRINT żeby wyświetlić wartość **avg**, jak pokazano poniżej. Uruchom program i zobacz wyniki.

```
Średnia pensja: $36,749
```

- Jaki tekst jest wyświetlany w stopce?

Poziom 2

7. Wykorzystanie DATA stepu do generowania serii makrozmiennych

a. Otwórz **m103p07.sas** z folderu **practices** i obejrzyj program. Widok **sashelp.vmacro** jest dynamicznym widokiem, który zawiera nazwy i wartość makrozmiennych zdefiniowanych w bieżącej sesji SAS. Uruchom program i zobacz raport.

b. Dodaj DATA _NULL_ przed procedurę PRINT, żeby stworzyć serię makrozmiennych używając wartości ze zbioru **mc1.Storm_Ocean_Codes**. Użyj wartości zmiennej **Ocean** do zdefiniowania nazw makrozmiennych. Użyj wartości zmiennej **OceanName** jako wartości makrozmiennych.

c. Zmodyfikuj procedurę PRINT tak, żeby wyświetlić makrozmienne, których nazwy mają tylko jeden znak. Uruchom cały program i sprawdź, czy zostało stworzonych pięć makrozmiennych.

Jaka jest wartość makrozmiennej **S**?

Koniec ćwiczeń

3.4 Pośrednie odwołania do makrozmiennych

Poziom 1

8. Używanie pośrednich odwołań do makrozmiennych

a. Otwórz **m103p08.sas** z folderu **practices**. Uruchom kod i zobacz wyniki.

Produkty generujące najwyższe zyski dla zamówień typu: Type1

Wiersz	Product ID	GrossSales	Profit
1	240200200081	$852.40	$820.35
2	210201000161	$5.20	$5.00

b. Zapoznaj się z tabelą **mc1.order_type_codes**. Kolumna **Order_Type_Code** zawiera wartości *1*, *2* i *3*, a kolumna **Order_Type** identyfikuje typ zamówienia związany z danym kodem. Zamknij tabelę.

c. Na początku programu dodaj zapytanie SQL, które na podstawie zbioru **mc1.order_type_codes** stworzy serię makrozmiennych **Type1**, **Type2** i **Type3**. Przypisz wartości zmiennej **Order_Type** związane z każdym typem zamówienia. Usuń raport generowany przez PROC SQL i dodaj instrukcję %PUT, żeby wypisać w logu wartości **Type1**, **Type2** i **Type3**.

d. Uruchom procedurę SQL i instrukcję %PUT. Przejrzyj log. Sprawdź, że log zawiera *TYPE1=Retail Store TYPE2=Catalog TYPE3=Internet*.

e. Zmodyfikuj instrukcję TITLE, żeby użyć niejawnego odwołania do makrozmiennej **Type** odpowiadającej zadanej wartości makrozmiennej **code**. Uruchom cały program i zobacz wyniki. Sprawdź, że raport zawiera te same dwa wiersze, co oryginalny raport i że jego tytuł to Produkty generujące najwyższe zyski dla zamówień typu: *Retail Store*.

f. Zmodyfikuj instrukcję %LET, żeby przypisać wartość *3* do makrozmiennej **code**. Uruchom ponownie program i zobacz wyniki.

- Jaki jest tytuł raportu?

Poziom 2

9. Używanie pośrednich odwołań do makrozmiennych

a. Otwórz **m103p09.sas** z folderu **practices** i zobacz kod. Uruchom program i zobacz wyniki. Sprawdź, że raport jest zatytułowany *Klienci zamieszkujący LU*, a raport zawiera trzy wiersze.

b. Na początku programu dodaj DATA _NULL_, żeby stworzyć serię makrozmiennych ze zbioru **mc1.country_codes**. Użyj wartość kolumny **CountryCode** jako nazwę makrozmiennej, wartość kolumny **CountryName** jako wartść makrozmiennej. Uruchom DATA _NULL_, żeby stworzyć makrozmienne.

c. Zmodyfikuj instrukcję TITLE, żeby dodać nazwę kraju w oparciu o wartość makrozmiennej **code**. Sprawdź, że raport jest zatytułowany *Klienci zamieszkujący Luxembourg*.

d. Zmodyfikuj instrukcję %LET, żeby przypisać wartość *ZA* do makrozmiennej **code**. Uruchom program i zobacz wyniki.

- Ilu klientów jest z ZA i jaka jest nazwa kraju?

Koniec ćwiczeń

Rozdział 4 Praca z makroprogramami

4.1 Definiowanie i wywoływanie makra

Poziom 1

1. Definiowanie i wykorzystanie makroprogramów z parametrami

a. Otwórz plik **m104p01.sas** z folderu **practices**. Zobacz i uruchom program. Sprawdź, że generuje raport dla złotych klientów (Gold customers), który zawiera 680 wierszy.

b. Zamień kod w makroprogram **Customers**, który akceptuje jeden parametr pozycyjny o nazwie **Type**. Uruchom definicję makroprogramu i sprawdź w logu, czy został skompilowany bezbłędnie.

- Wywołaj makroprogram **%Customers** z parametrem *Gold*. Sprawdź, że tytuł raportu to *Klienci wg typu: Gold* i że raport zawiera 680 wierszy.
- Ponownie wywołaj **%Customers**, tym razem z parametrem *High Activity*. Żadne wiersze nie zostały wybrane ze względu na różną wielkość liter.

c. Zmodyfikuj makroprogram, żeby zamienić wartość parametru **Type** na duże litery, i zmień instrukcję WHERE, żeby porównanie nie brało pod uwagę wielkości liter. Uruchom definicję makroprogramu i sprawdź, ze skompilowało się bezbłędnie.

- Wywołaj makro z wartością parametru *High Activity*. Sprawdź, że tytuł raportu to *Klienci wg typu: HIGH ACTIVITY* i że raport zawiera 461 wierszy.

d. Zmień parametr pozycyjny na parametr według klucza z domyślną wartością *inactive*.

- Wywołaj makro **%Customers** z wartością parametru *high activity*. Sprawdź, że tytuł raportu to *Klienci wg typu: HIGH ACTIVITY* i że raport zawiera 461 wierszy.
- Wywołaj makro **%Customers** bez podawania wartości dla parametru. Sprawdź, że tytuł raportu to *Klienci wg typu: INACTIVE*. Ile wierszy jest w raporcie?

Poziom 2

2. Wykorzystanie makroprogramu do wygenerowania kodu PROC MEANS

a. Otwórz plik **m104p02.sas** z katalogu **practices**. Uruchom program i zobacz wyniki. Sprawdź, że średnia wartość **Total_Retail_Price** zamówień złożonych przez klienta o identyfikatorze 51 to 314,74.

b. Zamień kod w makroprogram o nazwie **Orderstats**, akceptujący parametry podane według klucza jak poniżej. Ustaw domyślne wartości parametrów tak, aby makroprogram wywołany bez parametrów generował oryginalny raport. Zmodyfikuj kod PROC MEANS, żeby odwołać się do parametrów.

- **var** - lista zmiennych w instrukcji VAR
- **class** - lista zmiennych w instrukcji CLASS
- **stats** - wyliczane statystyki
- **decimals** wartość opcji MAXDEC=

c. Wywołaj makro **%Orderstats** bez podawania parametrów. Sprawdź, że średnia wartość **Total_Retail_Price** dla klienta o identyfikatorze 51 jest taka sama jak w części a.

d. Ponownie wywołaj makro z parametrami podanymi poniżej.

- var=CostPrice_Per_Unit
- decimals=0
- stats=mean median
- class=order_type

Jaka jest mediana dla order_type równego 3?

Wyzwanie

3. Wykorzystanie parametrów zawierających znaki specjalne

a. Otwórz plik **m104p03.sas** z folderu **practices**. Zobacz i uruchom program. Makroprogram kompiluje się bezbłędnie, ale pojawia się sześć ostrzeżeń, ponieważ parametr zawiera znak specjalny. Potrzebne są funkcje maskujące, żeby wyeliminować nierozwiązane odwołanie do makrozmiennej.

b. Zmodyfikuj wywołanie makra tak, aby ukryć znaki specjalne w czasie kompilacji i ponownie wywołaj makro. Sprawdź, że w logu wyświetlane są cztery ostrzeżenia.

c. Zabezpieczyłeś znak specjalny w czasie kompilacji, ale w czasie wykonania, wartość podstawiona za **prodline** zawiera niezamaskowany znak specjalny. Funkcje maskujące Q maskują znaki specjalne występujące w wynikach. Nazwy funkcji zaczynają się od %Q (na przykład %QSCAN).

Zmodyfikuj makroprogram tak, aby użyć funkcji Q do wyeliminowania ostrzeżeń w logu. Uruchom zmodyfikowany kod i sprawdź, że w logu nie ma żadnych błędów ani ostrzeżeń.

Koniec ćwiczeń

4.2 Zasięg makrozmiennych

Poziom 1

4. Rozumienie tablic symboli

Bez uruchamiania kodu, zidentyfikuj, w której tablicy symboli zostanie stworzona makrozmienna **Zip**. Załóż, że każdy przykład jest uruchamiany w niezależnej sesji SAS.

a.

```
%let Zip=27513;
%macro whereis;
    %let state=%sysfunc(zipnamel(&Zip));
    %put Kod pocztowy &Zip jest w &state;
%mend whereis;

%whereis
```

b.

```
%macro whereis;
    %let Zip=10312;
    %let state=%sysfunc(zipnamel(&Zip));
    %put Kod pocztowy &Zip jest w &state;
%mend whereis;

%whereis
```

c.

```
%macro whereis(Zip);
    %let state=%sysfunc(zipnamel(&Zip));
    %put Kod pocztowy &Zip jest w &state;
%mend whereis;

%whereis(91219)
```

Poziom 2

5. Kontrolowanie zakresu makrozmiennych

a. Otwórz plik **m104p05.sas** z folderu **practices**. Zauważ, że makrozmienna jest stworzona wewnątrz makra **%Scope** za pomocą makroinstrukcji %LET. Podświetl i uruchom kod w sekcji 1, zobacz log.

Uwaga: W logu globalne makrozmienne są oznaczane jako *GLOBAL* a lokalne jako *SCOPE*.

- W której tablicy symboli zostały stworzone makrozmienne?

b. Znajdź w programie sekcję 2. Zauważ, że makrozmienna jest stworzona wewnątrz makra **%Scope** za pomocą wyrażenia SQL INTO. Podświetl i uruchom kod w sekcji 2, zobacz log.

- W której tablicy symboli zostały stworzone makrozmienne?

c. Znajdź w programie sekcję 3. Zauważ, że makrozmienne są tworzone wewnątrz makra **%Scope** za pomocą rutyny CALL SYMPUTX w DATA stepie. Podświetl kod w sekcji 3 łącznie z wyrażeniem %SYMDEL. Uruchom zaznaczony kod i sprawdź log.

- W której tablicy symboli zostały stworzone makrozmienne?

d. Dodaj wyrażenie %LOCAL przed DATA stepem, żeby zdefiniować makrozmienną **x**. Podświetl i uruchom kod w sekcji 3, zobacz log.

- W której tablicy symboli zostały stworzone makrozmienne?

e. Usuń wyrażenie %LOCAL i zmodyfikuj CALL SYMPUTX tak, aby makrozmienna była tworzona w lokalnej tablicy symboli. Podświetl i uruchom kod w sekcji 3, zobacz log.

UWAGA**:** Wyrażenie %SYMDEL generuje ostrzeżenia ponieważ podanych makrozmiennych nie ma obecnie w globalnej tablicy symboli.

- W której tablicy symboli zostały stworzone makrozmienne?

Wyzwanie

6. Debugowanie problemów z zakresem makrozmiennych

a. Otwórz **m104p06.sas** z folderu **practices.** Zobacz i uruchom program. Dlaczego kończy się błędem?

b. Popraw program i uruchom ponownie.

Koniec ćwiczeń

4.3 Przetwarzanie warunkowe

Poziom 1

7. Wykorzystanie przetwarzania warunkowego do wyboru generowanych instrukcji

a. Otwórz **m104p07.sas** z folderu **practices**.

b. Zmodyfikuj makro **CustomerList** tak, aby sprawdzać parametr **country**. Jeżeli ma wartość pustą, dodaj następujące instrukcje do PROC PRINT:

```
title "Wszyscy klienci";
var ID Name Country Type Age_Group;
```

Jeżeli wartość nie jest pusta, dodaj następujące instrukcje do PROC PRINT:

```
title "Klienci z kraju: &country";
where Country="&country";
var ID Name Type Age_Group;
```

c. Uruchom makro i wywołaj je podając pustą wartość parametru. Sprawdź, że tytuł jest poprawny i że raport ma 1800 wierszy.

Wszyscy klienci

ID	Name	Country	Type	Age_Group
1	Albert Collet	FR	Gold high activity	61-75 years
51	Jean-Claude Nogues	FR	No member status	31-45 years
101	Christine Herfroy	FR	Club medium activity	61-75 years
151	Alex Hammersly	GB	Club medium activity	31-45 years
201	Nogarji Mcadam	US	Gold low activity	61-75 years
251	Laurent Ollivon	FR	Gold medium activity	15-30 years
301	Anneleen Vanlangenaeker	BE	Gold low activity	46-60 years

d. Wywołaj makro ponownie podając *AU* jako wartość parametru **country**. Ilu klientów jest na raporcie?

Klienci z kraju: AU

ID	Name	Type	Age_Group
2351	Kay Pachare	Gold high activity	15-30 years
2451	Meshlin Bhula	Gold high activity	15-30 years
3103	Brenda Leavey	Club medium activity	46-60 years
4704	Norlene Pinkus	No member status	46-60 years
5558	Rodger Schelleim	Gold low activity	31-45 years
6159	Pascal Filleti	Gold medium activity	61-75 years

Poziom 2

8. Używanie przetwarzania warunkowego w otwartym kodzie

a. Otwórz plik **m104p08.sas** z folderu **practices**. Uruchom program i sprawdź, że powstała tymczasowa tabela o nazwie **Profit** oraz wykres słupkowy.

b. Dodaj instrukcję LIBNAME, żeby stworzyć bibliotekę **Orion**, która wskazuje na folder **practices** w katalogach szkoleniowych. Zmodyfikuj wyrażenie CREATE TABLE tak, żeby zapisać tabelę **Profit** do biblioteki **Orion**.

c. Dodaj instrukcję %PUT, żeby wypisać w logu wartość automatycznej makrozmiennej **Syslibrc**. Jeżeli instrukcja LIBNAME wykona się poprawnie, to wartość **Syslibrc** jest równa zero. W innym przypadku wartość jest różna od zero. Uruchom instrukcje LIBNAME i %PUT.

d. Dodaj makroinstrukcję warunkową, żeby sprawdzić wartość **Syslibrc**. Jeżeli nie jest równa zero, wtedy wypisz do logu informację, że dalsza część kodu nie będzie wykonana ze względu na błąd w instrukcji LIBNAME. W przeciwnym przypadku uruchom kody PROC SQL i SGPLOT.

e. Przetestuj program z poprawną i błędną instrukcją LIBNAME. Jeżeli wykres został stworzony, jaka wartość zmiennej **Product_Category** ma największą sumę zmiennej **Profit**?

Wyzwanie

9. Wykorzystanie makroinstrukcji warunkowej do testowania wartości parametrów

a. Otwórz plik **m104p09.sas** z folderu **practices**. Zobacz kod makroprogramu **%SalesSummary**. Zauważ, że makro akceptuje jeden parametr, **ot**, który wskazuje wartość typu zamówienia (**OrderType**). Prawidłowe wartości dla **OrderType** to *1*, *2* i *3*. Uruchom program i sprawdź, że makro działa poprawnie dla parametru **ot** równego *1*, *2* lub *3*.

b. Wykorzystaj makroinstrukcje warunkowe do przetestowania wartości parametru **ot**. Jeżeli nie jest równa *1*, *2* ani *3*, to wypisz następujący komunikat błędu do logu. W przeciwnym przypadku uruchom pozostałą część makroprogramu.

```
ERROR: <wartość ot> nie jest poprawnym typem zamówienia. Poprawne typy zamówień to: 1 (Retail), 2
(Catalog) i 3 (Internet).
```

c. Wywołaj **%SalesSummary** z wartością parametru równą *4*. Potwierdź, że w logu pojawił się komunikat błędu.

Koniec ćwiczeń

4.4 Przetwarzanie iteracyjne

Poziom 1

10. Przetwarzanie w pętli i odwołania pośrednie

a. Otwórz **m104p10.sas** z folderu **practices**. Zobacz i uruchom program. Sprawdź, że generuje raport procedury MEANS dla zamówień typu 1.

b. Zmodyfikuj kod definiując makro o nazwie **Orders**,które wykorzystuje pętlę %DO do wygenerowania osobnych kodów PROC MEANS dla zamówień typu 1, 2, and 3. Dodaj wywołanie makra **%Orders**. Uruchom program i sprawdź, że generuje trzy raporty PROC MEANS, po jednym dla każdego typu. Potwierdź, że średnia dla **Total_Retail_Price** dla zamówień typu 2 jest równa 162,43.

c. Zmodyfikuj program.

- Po wyrażeniu %MACRO dodaj DATA _NULL_, który stworzy serię makrozmiennych od **type1** do **type*n***, gdzie ***n*** to liczba typów zamówień ze zbioru **mc1.order**_type_codes. Dla każdej makrozmiennej połącz prefiks **type** z wartością zmiennej **order_type_code**. Przypisz odpowiednią wartość **Order_Type** do makrozmiennej.
- Stwórz makrozmienną **numTypes** i przypisz jej liczbę stworzonych makrozmiennych **type**. Wskazówka: Użyj opcji END= w instrukcji SET.
- Użyj **numTypes** jako wartości końcowej pętli.
- Użyj pośredniego odwołania, żeby dodać typ zamówienia w tytule, tak jak pokazano poniżej.

```
Typ zamówienia: Retail Store
```

d. Uruchom program. Jaki jest tytuł ostatniego raportu?

Poziom 2

11. Wykorzystanie pętli %DO %UNTIL

a. Otwórz **m104p11.sas** z folderu **practices**. Zobacz i uruchom program. Sprawdź, że maksymalna prędkość wiatru w sezonie 2015 to 213 MPH.

b. Zmodyfikuj makro **%Storms**, żeby akceptowało listę lat oddzielanych spacją. Wykorzystaj pętlę %DO %UNTIL i makrofunkcję %SCAN, żeby wygenerować raport dla każdego roku na liście. Uruchom makro i sprawdź, że skompilowało się bezbłędnie.

c. Wywołaj makro **%Storms**, żeby wygenerować raporty dla sezonów 2011 i 2014 tak, jak poniżej.

```
%storms(2011 2014)
```

- Jaka była maksymalna prędkość wiatru w każdym roku?

Wyzwanie

12. Wykorzystanie przetwarzania w pętli do wywołania jednego makra z drugiego

a. Otwórz plik **m104p12.sas** z folderu **practices**. Zobacz i uruchom program. Makro **%Print** akceptuje trzy parametry i generuje kod PROC PRINT tworzący listing zawierający wskazaną liczbę wierszy z tabeli podanej jako parametr. Wywołanie makra ma wyświetlić 10 wierszy ze zbioru **mc1.orders**. Uruchom kod z części a i zobacz wyniki.

Pierwszych 10 wierszy z tabeli: mc1.orders

Obs.	Customer_ID	Employee_ID	Street_ID	Order_ID	Product_ID	Quantity	Total_Retail_Price	CostPrice_Per_Unit	Discount	Order_Type	Order_Date	Delivery_Date
1	74503	99999999	2600100022	1230000487	240200100007	1	24.7	11.80	.	2	20089	20092
2	19278	99999999	3940106902	1230000689	230100100012	2	358.6	82.00	.	2	20089	20093
3	54339	99999999	3500100256	1230008126	240100100516	1	25.3	10.95	.	3	20090	20094
4	27990	120932	9250102718	1230008626	220200100193	2	157.2	39.40	.	1	20090	20090
5	81703	120603	4800105312	1230009131	220101400029	4	18.4	1.90	.	1	20090	20090
6	81703	120603	4800105312	1230009131	220101000002	1	17.7	8.00	.	1	20090	20090
7	29494	120554	6300100743	1230009270	240500100055	1	93.2	37.30	.	1	20090	20090
8	55541	99999999	9260105109	1230010878	240100100440	2	283.8	70.90	.	3	20090	20092
9	59593	121138	9260115071	1230010979	240800100090	2	413.8	82.80	.	1	20090	20090
10	59593	121138	9260115071	1230010979	240800200051	2	52.2	12.45	.	1	20090	20090

b. Część b zawiera początkowy kod makra **%Printlib**, które przyjmuje jako parametry nazwę biblioteki i liczbę wierszy. PROC SQL tworzy serię makrozmiennych, które zawierają nazwy tabel w podanej bibliotece.

Uzupełnij makro **%Printlib**, dodając pętlę %DO, w której będzie wywoływane makro **%Print** i będzie drukowało pierwszych **numrows** wierszy dla każdej tabeli w podanej bibliotece. Jeżeli parametr **numrows** nie jest podany, makro powinno użyć wartości domyślnej.

c. Wywołaj **%Printlib**, żeby wydrukować pierwszych 10 wierszy dla każdej tabeli z biblioteki **mc1**.

- Jaka jest nazwa tabeli w ostatnim raporcie?
- Ile wierszy jest w ostatnim raporcie?

d. Wywołaj makro **%Printlib**, żeby wyświetlić każdą tabelę w bibliotece **mc1**. Nie podawaj wartości dla **numrows**.

- Jaka jest nazwa tabeli w ostatnim raporcie?
- Ile wierszy jest wyświetlonych w każdym raporcie?

Koniec ćwiczeń

Rozdział 5 Tworzenie makroaplikacji

5.1 Generowanie kodu zależnego od danych

Poziom 1

1. Generowanie raportów zależnych od danych

a. Otwórz **m105p01.sas** z folderu **practices**. Zobacz i uruchom program. Sprawdź, że został wygenerowanych raport dotyczący klientów **Gold high activity**.

b. Dodaj instrukcję %MACRO, żeby stworzyć makro **GroupList** z dwoma parametrami: **Tab** do podania tabeli wejściowej i **Col** do podania wybranej kolumny. Makro będzie uogólniać program początkowy tak, żeby raport procedury PRINT był generowany dla każdej unikalnej wartości podanej kolumny.

c. Użyj procedury SQL, żeby select the distinct uppercase values of the Col parameter. Load the values into a numbered series of macro variables starting with Val1. Read the column from the table specified by the Tab parameter.

d. Dodaj makropętlę %DO, żeby powtórzyć instrukcję TITLE oraz procedurę PRINT dla każdej stworzonej makrozmiennej **Val*n***.

Uwaga: Wykorzystaj makrozmienną **Sqlobs**, tworzoną automatycznie przez procedurę SQL, aby określić wartość końcową pętli.

e. Użyj pośredniego odwołania do makrozmiennej, aby zastąpić *GOLD HIGH ACTIVITY* odwołaniem do **Val1** w pierwszej iteracji pętli, **Val2** w drugiej iteracji itd.

f. Zakończ makropętlę instrukcją %END i definicję makroprogramu instrukcją %MEND.

g. Wywołaj makro **%GroupList**, podających **mc1.customers** i **Type** jako wartości parametrów. Sprawdź, że osobne kody procedury PRINT zostały wykonane dla każdej unikalnej wartości **type**.

h. Wywołaj makro **%GroupList** z wartościami parametrów **sashelp.cars** i **DriveTrain**.

- Jaka jest wartość **DriveTrain** w pierwszym wygenerowanym raporcie?

Poziom 2

2. Eksportowanie danych do osobnych arkuszy w Excelu

a. Otwórz **m105p02.sas** z folderu **practices**. Zobacz i uruchom program. Otwórz plik Excel, żeby sprawdzić, że zawiera huragany z akwenu NA.

- SAS Studio: W zakładce Pliki i foldery panelu nawigacyjnego przejdź do folderu szkoleniowego, wybierz NAStorms.xlsx file, i kliknij  (**Pobierz**). Po obejrzeniu pliku zamknij Excel.
- SAS Enterprise Guide: W oknie Serwery, rozwiń **Lokalne** ⇨ **Pliki** i przejdź do folderu szkoleniowego. Kliknij dwa razy **NAStorms.xlsx**. Po obejrzeniu pliku zamknij Excel.

b. Stwórz makroprogram o nazwie **BasinXLS**, który wypisuje huragany dla każdej unikalnej wartości zmiennej **Basin** do osobnych arkuszy pliku Excel **BasinStorms.xlsx**. Uwzględnij następującą logikę w makroprogramie:

- Użyj automatycznej makrozmiennej **syslibrc**, żeby sprawdzić, czy instrukcja LIBNAME zakończyła się pomyślnie (**SYSLIBRC**=0). Jeżeli były błędy (**SYSLIBRC**≠0), napisz w logu własny komunikat o błędzie i zakończ działanie makra.
- Stwórz serię makrozmiennych dla każdej wartości zmiennych **Basin** i **BasinName** w tabeli **mc1.storm_basin_codes**. Usuń wszystkie spacje z wartości **BasinName** przed przypisaniem ich do makrozmiennych.
- Użyj makropętli %DO, żeby powtórzyć DATA step dla każdej wartości zmiennej **Basin**.

c. Wywołaj makro **%BasinXLS** i otwórz plik **BasinStorms.xlsx**. Sprawdź, że plik zawiera sześć arkuszy.

- Jaka jest nazwa pierwszego huraganu na arkuszu **SouthIndian_Storms**?

Wyzwanie

3. Analiza wszystkich kolumn z tabeli

a. Otwórz **m105p03.sas** z folderu **practices**. Zobacz i uruchom program. Zapytanie SQL generuje raport w oparciu o **dictionary.columns**, która zawiera listę wszystkich kolumn z wybranej tabeli i ich wszystkich atrybutów. Procedura SGPLOT tworzy histogram, żeby przedstawić rozkład wartości kolumny numerycznej **MSRP**. Procedura SGPIE tworzy wykres pierścieniowy, żeby pokazać rozkład kolumny znakowej.

b. Stwórz makroprogram **ColumnReport** z jednym parametrem **Tab**. Zastosuj następującą logikę w makrze, żeby stworzyć histogram dla wszystkich kolumn numerycznych w wybranej tabeli i wykres pierścieniowy dla wszystkich zmiennych znakowych:

- Zmień wartość parametru **Tab** na duże litery.
- Po pierwszym zapytaniu SQL dodaj drugą procedurę SQL, żeby stworzyć serię numerowanych makrozmiennych, które przechowują wartości kolumn **Name** i **Type**.
- Wykorzystaj makropętlę %DO, żeby wykonać albo SGPLOT albo SGPIE dla każdej kolumny w tabeli. Jeżeli typ kolumny to *num*, wtedy uruchom procedurę SGPLOT tworzącą histogram. Jeżeli typ kolumny to *char*, wtedy uruchom procedurę SGPIE tworzącą wykres pierścieniowy.
- Dla każdego wykresu zdefiniuj tytuł: **Rozkład zmiennej <*nazwa_kolumny*>**.

c. Wywołaj makro **%ColumnReport** z parametrem **mc1.storm_final**. Sprawdź, że właściwe wykresy zostały wygenerowane dla każdej kolumny.

Uwaga: Wartości kolumn znakowych zawierających wiele unikalnych wartości, np. **Name**, zostały pogrupowane w grupę o nazwie *Inne*.

Uwaga: Ostrzeżenie dotyczące czcionki można zignorować.

d. Wywołaj makro ponownie, podając **mc1.customers** jako wartość parametru.

- Ile członków Orion Club Gold jest na wykresie **Rozkład zmiennej Group**?

Koniec ćwiczeń

5.2 Walidacja parametrów i dokumentowanie makr

Poziom 1

4. Sprawdzanie poprawności parametru

a. Otwórz **m105p04.sas** z folderu **practices**. Zobacz i uruchom program.

b. Zmodyfikuj makro tak, aby zamienić wartość parametru **Type** na duże litery.

c. Dodaj instrukcję %IF-%THEN/%ELSE, żeby sprawdzić wartość parametru **Type**. Makro powinno generować instrukcję TITLE i kod procedury FREQ tylko, jeżeli wartość parametru **Type** jest równa *LOW*, *MEDIUM* lub *HIGH*. Jeżeli wartość parametru **Type** nie jest prawidłowa, makro powinno wypisywać następujący komunikat w logu (gdzie *xxxx* jest wartością parametru **Type**):

```
ERROR: Błędna wartość TYPE: xxxx
ERROR: Poprawne wartości TYPE to HIGH, MEDIUM, LOW
```

d. Skompiluj makro i wywołaj je używając prawidłowych i nieprawidłowych wartości parametru.

e. Zmodyfikuj makro, żeby najpierw sprawdzić, czy **Type** jest pusty. Jeżeli tak, to nie generuj instrukcji TITLE ani kodu procedury FREQ. Zamiast tego, makro powinno wypisywać w logu komunikat:

```
ERROR: Brakująca wartość TYPE
ERROR: Poprawne wartości TYPE to HIGH, MEDIUM, LOW
```

f. Skompiluj makro i wywołaj je w pustym parametrem, poprawnymi wartościami podanymi małymi i dużymi literami oraz nieprawidłową wartością.

- Ilu klientów *low-activity* pochodzi z Włoch (*IT*)?

Poziom 2

5. Sprawdzanie poprawności parametrów w oparciu o dane

a. Otwórz **m105p05.sas** z folderu **practices**. Zobacz i uruchom program.

b. Użyj procedury SQL do wygenerowania makrozmiennych **MinSeason** i **MaxSeason**, które przechowują minimalną i maksymalną wartość kolumny **Season** column w tabeli **mc1.storm_final**. Dodaj warunek sprawdzający, czy wartość parametru **Season** jest pomiędzy **MinSeason** i **MaxSeason**. Jeżeli wartość parametru **Season** jest poza poprawnym zakresem, wypisz do logu komunikat:

```
ERROR: Poprawne lata przyjmują wartości z zakresu <minSeason> i <maxSeason>
```

c. Jeżeli wartość parametru jest poprawna, użyj procedury SQL, żeby stworzyć listę poprawnych kodów basenów oddzielanych spacjami, na podstawie zmiennej **Basin** z tabeli **mc1.storm_basin_codes**. Zapisz listę na makrozmiennej.

d. Jeżeli wartość parametru **Basin** jest na liście poprawnych wartości, wykonaj instrukcję TITLE i procedurę SGPLOT.

Uwaga: Upewnij się, że opcja MINOPERATOR została podana w instrukcji %MACRO, żeby można było wykorzystać operator IN.

e. Jeżeli wartość parametru **Basin** nie jest poprawna, wypisz w logu komunikat:

```
ERROR: AA to niepoprawny kod oceanu. Poprawne kody to NA WP EP SP NI SI.
```

f. Przetestuj makro z poprawnymi i niepoprawnymi wartościami **Basin** i **Season**.

- Ile burz tropikalnych było w akwenie EP w sezonie 2010?

Wyzwanie

6. Sprawdzanie poprawności parametrów, których wartości wymagają funkcji maskujących

Makrofunkcja %SUPERQ jest pomocną makrofunkcją, która odczytuje wartość makrozmiennej z tablicy symboli i natychmiast maskuje znaki specjalne, dzięki czemu makroprocesor nie próbuje rozwiązać niczego, co może się pojawić w wartości makrozmiennej. Jeżeli wartość makrozmiennej zawiera znaki %, &lub inne symbole specjalne, %SUPERQ jest wygodną alternatywą do odwołania ***&mac-var***.

```
%SUPERQ(mac-var)
```

Uwaga: Nie dodawaj znaku **&**przed nazwą makrozmiennej.

a. Otwórz **m105p06.sas** z folderu **practices**. Zobacz i uruchom program.

b. Zmodyfikuj makroprogram tak, aby w przypadku, gdy parametr **Ctry** jest pusty, instrukcja TITLE i kod procedury SGPLOT dotyczył klientów ze wszystkich krajów.

Uwaga: Kody krajów *IN* (Indie) i *LT* (Litwa) są możliwymi wartościami parametru, które mogą być interpretowane jako operatory IN (na liście) lub LT (mniejsze niż). W instrukcji %IF użyj funkcji %SUPERQ, żeby wymusić traktowanie wartości parametru **Ctry** jako tekstu.

c. Jeżeli **Ctry** nie jest pusty, sprawdź, że jego wartość to jedna z unikalnych wartości zmienne **Country** z tabeli **mc1.customers**. Jeżeli **Ctry** jest poprawnym kodem kraju, wygeneruj wykres dla klientów z tego kraju.

Uwaga: Użyj funkcji %SUPERQ, żeby zapewnić, że wartość parametru **Ctry** i lista kodów krajów są traktowane jako tekst.

d. Jeżeli wartość parametru **Ctry** jest niepoprawna, wypisz komunikat do logu i podaj listę poprawnych kodów:

```
ERROR: ZZ nie jest poprawną wartością kodu kraju
             Poprawne wartości kodu krajów to <lista kodów krajów>
```

e. Przetestuj makro z wartościami parametrów: pusta, *AU*, *in*, *zz*.

Jaki jest ostatni poprawny kod kraju wyświetlony w komunikacie w logu, gdy wartość parametru to *zz*?

Koniec ćwiczeń