# Przygotowanie danych

# Usunięcie wierszy z pustym speech\_raw

* 1. Pozostało w bazie 272 221

# Algorytm przypisania autora tekstu (SQL).

Podczas przeglądania danych zidentyfikowano błędy w przypisaniu id autorów przemówień. Konsekwencją tego jest brak możliwości łatwego przypisania autora danego wystąpienia do jego tekstu. Błędy występowały dla większości wpisów w bazie i mimo tego, że sprawiały wrażenie systematycznej zmiany (poprawna wartość wydawała się bliska z kontekście porządku leksykograficznego), nie udało się ustalić algorytmu, który mógłby przypisać poprawne wartości w sposób automatyczny.

W związku z powyższym przypisanie autora (*author\_final*) wykonano w następujących krokach (10\_ author\_final.sql):

1. Wiele wystąpień zaczyna się od schematu „Poseł Imię Nazwisko:”. Pierwsze występujące w takim kontekście imię i nazwisko przypisano jako dane autora z tekstu (kolumna *author\_by\_text*).
2. Przypisano imię i nazwisko autora w oparciu o *id\_* z bazy (kolumna *author\_by\_id*).
3. Dla przypadków gdzie dla danego id, chociaż raz *author\_by\_text* jest równe *author\_by\_id*, przypisano te wartości jako *author\_final*.
4. Dla pozostałych przypadków sprawdzono jaka wartość *author\_by\_text* pojawia się najczęściej w obrębie danego id. Jeśli wartość była niepusta, została przypisana jako *author\_final*.
5. Pozostałe przypadki to takie, gdzie najczęściej pole *author\_by\_text* było puste, tj. w tekście wystąpienia nie występowało zestawienie „Poseł Imię Nazwisko:”. Wynika to z różnej postaci plików transkrypcją – dla części z nich fragmenty pogrubione na poniższym obrazku nie znalazły się w tekście dostępnym w bazie.



1. W trakcie analizy danych niektóre id były weryfikowanych ręcznie w oryginalnych plikach pdf. Wartości dla nich wprowadzono ręcznie w kodzie.

Po zastosowaniu powyższych kroków w bazie brakowało przypisania *author\_final* dla 795 id (na 3828). Dla pojedynczych fragmentów wystąpień uzupełnienie wyniosło ok 83%.

# Czyszczenie fragmentów przemówień (Python)

Wykonano następujące etapy czyszczenia danych:

* usunięto znaki specjalne \n \r \t
* wykasowano tytuły przemówień pojawiające się na początku każdego fragmentu
* usunięty fragmenty tekstu w nawiasach (np. (Oklaski.), (Dzwonek.))
* Usunięto fragmenty identyfikujące mówcę, tj. fragment „Poseł Imię Nazwisko:”

Następnie przygotowano drugi zestaw tesktów, które oczyszczono jeszcze bardziej z fragmentów mało informacyjnych.

* Wypowiedzi innych posłów i komentarze typu "Oklaski" pojawiające się w nawiasach
* "Poseł Imię Nazwisko:"
* Panie Marszałku!
* Pani Marszałek!
* Wysoka Izbo!
* Panie Ministrze!
* Dziękuję bardzo.

Po tym etapie czyszczenia w bazie pozostało 272 217 fragmentów przemówień, z których 225 385 ma przypisanego autora. Oba zestawy tesktów z różnym poziomem oczyszczenia będą stosowane w różnych analizach.

# Łączenie fragmentów przemówień (python)

Fragmenty tych samych wypowiedzi, zgrupowane po dacie, autore i tytule został y połączone.

Po usunięciu tekstów, które po wszystkich modyfikacjach stały się puste, w bazie zostało 158 885 tekstów z przypisanym autorem.

# Wstępna analiza danych

# Generator przemówień

Na tym etapie podjęto decyzję o budowie prostego generatora przemówień opartego o n-gramy występujące w analizowanym korpusie.

Model powstał w dwóch wersjach dla 1-gramów oraz 2-gramów. Dla obu opcji w pierwszym kroku przygotowano słownik zawierający wszystkie występujące kombinacje odpowiednich n-gramów ze słowami po nich następującymi.

Generowanie przemówień polega na przypisaniu słowa początkowego (lub wybraniu losowego) a następnie w sposób losowy wybieraniu słów kolejnych na bazie słownika zbudowanego w poprzednim punkcie.

Przykładowe przemówienie o długości 40 słów.