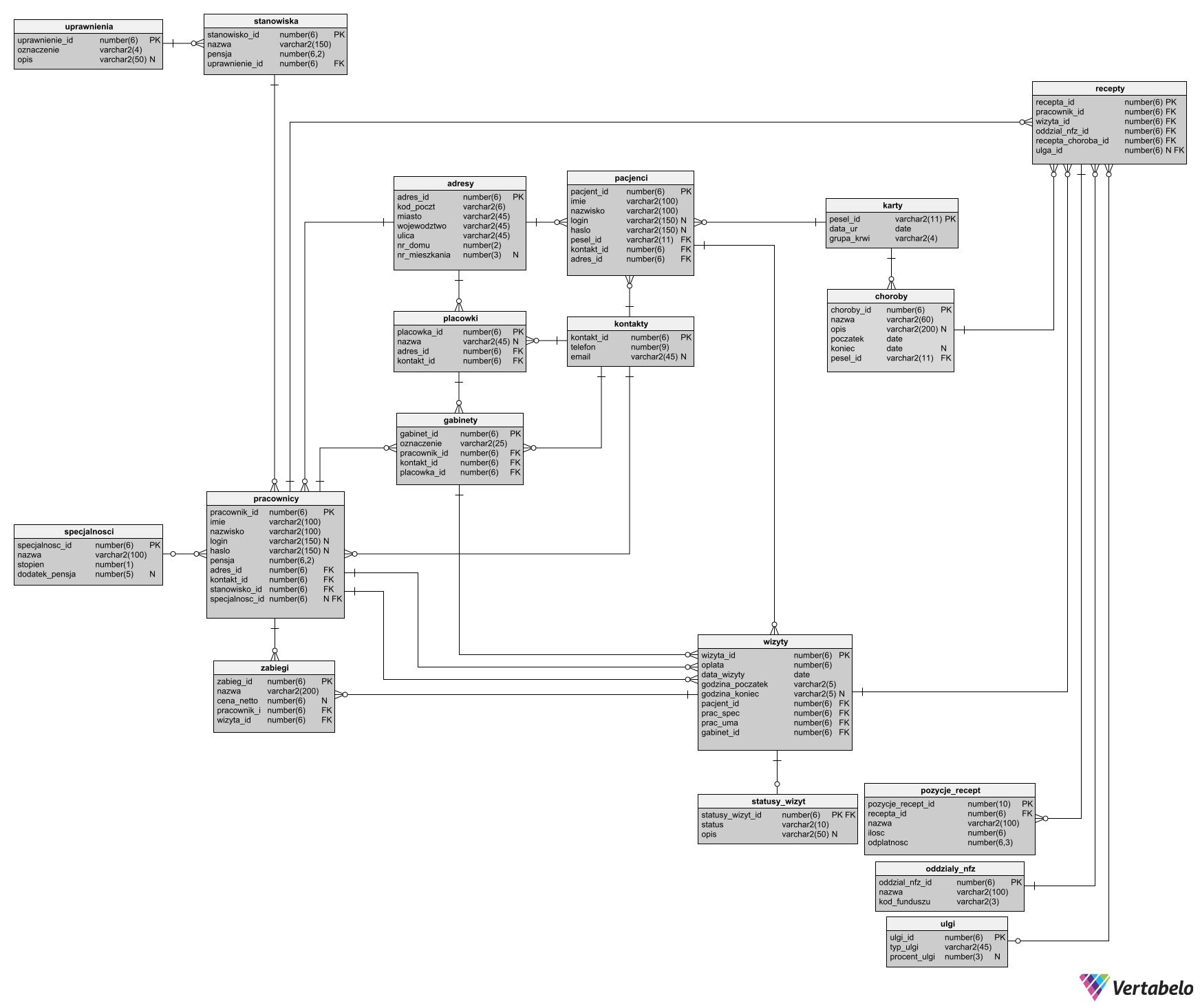
|  |  |  |  |
| --- | --- | --- | --- |
| **Politechnika Świętokrzyska w Kielcach**  **Wydział Elektrotechniki, Automatyki i Informatyki**  **Katedra Informatyki, Elektroniki i Elektrotechniki** | | | |
| Kierunek  **Informatyka** | Laboratorium nr 6.  **Bazy Danych 2** | | |
| Temat ćwiczenia   * Bazy danych NOSQL Neo4j – Zapytania złożone dla projektu | | Wykonał:  Patryk Grzywacz  Dominik Grudzień |
| Grupa dziekańska  **2ID12A** |
| Data wykonania  31.05.2021r | | Data oddania  31.05.2021r | Ocena i podpis |

1. **Schemat ERD Bazy z Projektu**

**Temat: Sieć Gabinetów Lekarskich**



* 1. **Opis tabel**
     + Adresy – Przechowuje dane dotyczące adresu fizycznego domu/budynku/placówki etc.
     + Kontakty – Przechowuje dane odnośnie możliwości kontaktowych zarówno osób jak i placówek medycznych.
     + Specjalności – Przechowuje dane o specjalnościach lekarzy.
     + Uprawnienia – Znajdują się w niej dane o uprawnieniach przypisywanych do stanowisk.
     + Stanowiska – Przechowują dane odnośnie stanowisk przydzielanych do poszczególnych pracowników.
     + Pracownicy – Przechowują dane osobiste jak i firmowe o pracownikach , a także odnośniki do adresów, kontaktów, stanowisk oraz ewentualnie do specjalności.
     + Placówki – Zawierają dane o placówkach medycznych ,adresie oraz kontakcie z nimi.
     + Gabinety – Zawierają dane o gabinetach lekarskich mieszczących się w placówkach ,do których przypisany jest odpowiedni pracownik, najczęściej jakiś lekarz.
     + Karty – Przechowują dane osobowe tj. data urodzenia czy też grupę krwi.
     + Pacjenci – Przechowują dane osobowe jak i klienckie, odnośniki do kart ,adresów oraz kontaktów.
     + Choroby – Przechowują dane odnośnie choroby/chorób przypisanych do danej Karty pacjenta
     + Ulgi – Zawierają dane ulg przeznaczonych do wykorzystania przy receptach.
     + Pozycje\_Recept – Zawierają dane materialne odnośnie leków przepisanych w ramach danej recepty.
     + Oddzialy\_NFZ – Zawierają dane specjalistyczne wymagane na każdej recepcie.
     + Statusy\_Wizyt – Zawierają dane określające status danej wizyty.
     + Wizyty – Przechowuje dane o wizytach pacjentów ,o dacie i czasie trwania, miejscu, opłacie a także kto umówił/przyjmował pacjenta.
     + Recepty – Zawiera same odnośniki do pracownika ,który ją wystawił , do wizyty podczas ,której została wystawiona a także do choroby , ulgi i oddziału nfz.
     + Zabiegi – Zawiera dane o zabiegu wykonanym w ramach danej wizyty , jego cenie oraz pracowniku, który go przeprowadzał.

1. **Napisane skrypty (Wszystkie skrypty będą w osobnych plikach .cyp wraz z sprawozdaniem , a także pliki .csv do zasilenia bazy danymi)**
   1. **Skrypty na utworzenie bazy grafowej (węzłów i relacji)**

MATCH(n:Adresy) DETACH DELETE n;

MATCH(n:Choroby) DETACH DELETE n;

MATCH(n:Gabinety) DETACH DELETE n;

MATCH(n:Karty) DETACH DELETE n;

MATCH(n:Kontakty) DETACH DELETE n;

MATCH(n:Oddzialy\_nfz) DETACH DELETE n;

MATCH(n:Pacjenci) DETACH DELETE n;

MATCH(n:Placowki) DETACH DELETE n;

MATCH(n:Pozycje\_recept)-[r]-() DETACH DELETE r;

MATCH(n:Pozycje\_recept) DETACH DELETE n;

MATCH(n:Recepty) DETACH DELETE n;

MATCH(n:Pracownicy) DETACH DELETE n;

MATCH(n:Specjalnosci) DETACH DELETE n;

MATCH(n:Stanowiska) DETACH DELETE n;

MATCH(n:Statusy\_wizyt) DETACH DELETE n;

MATCH(n:Ulgi) DETACH DELETE n;

MATCH(n:Uprawnienia) DETACH DELETE n;

MATCH(n:Wizyty) DETACH DELETE n;

MATCH(n:Zabiegi) DETACH DELETE n;

*//Rekordy w tabelach*

LOAD CSV WITH HEADERS FROM 'file:///ulgi.csv'  AS row

CREATE (:Ulgi{ulgi\_id:toInteger(row.ulgi\_id),typ\_ulgi:row.typ\_ulgi,procent\_ulgi:toInteger(row.procent\_ulgi)});

LOAD CSV WITH HEADERS FROM 'file:///oddzialy\_nfz.csv' AS row

CREATE (:Oddzialy\_nfz{oddzial\_nfz\_id:toInteger(row.oddzial\_nfz\_id),nazwa:row.nazwa,kod\_funduszu:row.kod\_funduszu});

LOAD CSV WITH HEADERS FROM 'file:///adresy.csv' AS row

CREATE (:Adresy {adres\_id:toInteger(row.adres\_id),kod\_poczt:row.kod\_poczt,miasto:row.miasto,wojewodztwo:row.wojewodztwo,ulica:row.ulica,

nr\_domu:toInteger(row.nr\_domu),nr\_mieszkania:toInteger(row.nr\_mieszkania)});

LOAD CSV WITH HEADERS FROM 'file:///kontakty.csv' AS row

CREATE (:Kontakty{kontakt\_id:toInteger(row.kontakt\_id),telefon:row.telefon,email:row.email});

LOAD CSV WITH HEADERS FROM 'file:///specjalnosci.csv' AS row

CREATE (:Specjalnosci{specjalnosc\_id:toInteger(row.specjalnosc\_id),nazwa:row.nazwa,stopien:row.stopien,dodatek\_pensja:toInteger(row.dodatek\_pensja)});

LOAD CSV WITH HEADERS FROM 'file:///uprawnienia.csv' AS row

CREATE (:Uprawnienia{uprawnienie\_id:toInteger(row.uprawnienie\_id),oznaczenie:row.oznaczenie,opis:row.opis});

LOAD CSV WITH HEADERS FROM 'file:///stanowiska.csv' AS row

CREATE (:Stanowiska{stanowisko\_id:toInteger(row.stanowisko\_id),nazwa:row.nazwa,pensja:toInteger(row.pensja),uprawnienie\_id:toInteger(row.uprawnienie\_id)});

LOAD CSV WITH HEADERS FROM 'file:///karty.csv' AS row

CREATE (:Karty{pesel\_id:row.pesel\_id,data\_ur:row.data\_ur,grupa\_krwi:row.grupa\_krwi});

LOAD CSV WITH HEADERS FROM 'file:///choroby.csv' AS row FIELDTERMINATOR '\*'

CREATE (:Choroby{choroby\_id:toInteger(row.choroby\_id),nazwa:row.nazwa,opis:row.opis,poczatek:row.poczatek,koniec:row.koniec,pesel\_id:row.pesel\_id});

LOAD CSV WITH HEADERS FROM 'file:///placowki.csv' AS row

CREATE (:Placowki{placowka\_id:toInteger(row.placowka\_id),nazwa:row.nazwa,adres\_id:toInteger(row.adres\_id),kontakt\_id:toInteger(row.kontakt\_id)});

LOAD CSV WITH HEADERS FROM 'file:///pracownicy.csv' AS row

CREATE (:Pracownicy{pracownik\_id:toInteger(row.pracownik\_id),imie:row.imie,nazwisko:row.nazwisko,login:row.login,haslo:row.haslo,pensja:toInteger(row.pensja),adres\_id:toInteger(row.adres\_id),kontakt\_id:toInteger(row.kontakt\_id),stanowisko\_id:toInteger(row.stanowisko\_id),specjalnosc\_id:toInteger(row.specjalnosc\_id)});

LOAD CSV WITH HEADERS FROM 'file:///pacjenci.csv' AS row

CREATE (:Pacjenci{pacjent\_id:toInteger(row.pacjent\_id),imie:row.imie,nazwisko:row.nazwisko,login:row.login,haslo:row.haslo,pesel\_id:row.pesel\_id,kontakt\_id:toInteger(row.kontakt\_id),adres\_id:toInteger(row.adres\_id)});

LOAD CSV WITH HEADERS FROM 'file:///gabinety.csv' AS row

CREATE (:Gabinety{gabinet\_id:toInteger(row.gabinet\_id),oznaczenie:row.oznaczenie,pracownik\_id:toInteger(row.pracownik\_id),kontakt\_id:toInteger(row.kontakt\_id),placowka\_id:toInteger(row.placowka\_id)});

LOAD CSV WITH HEADERS FROM 'file:///wizyty.csv' AS row

CREATE (:Wizyty{wizyta\_id:toInteger(row.wizyta\_id),oplata:toInteger(row.oplata),data\_wizyty:row.data\_wizyty,godzina\_poczatek:row.godzina\_poczatek,godzina\_koniec:row.godzina\_koniec,pacjent\_id:toInteger(row.pacjent\_id),prac\_spec:toInteger(row.prac\_spec),prac\_uma:toInteger(row.prac\_uma),gabinet\_id:toInteger(row.gabinet\_id)});

LOAD CSV WITH HEADERS FROM 'file:///recepty.csv' AS row

CREATE (:Recepty{recepta\_id:toInteger(row.recepta\_id),pracownik\_id:toInteger(row.pracownik\_id),wizyta\_id:toInteger(row.wizyta\_id),oddzial\_nfz\_id:toInteger(row.oddzial\_nfz\_id),recepta\_choroba\_id:toInteger(row.recepta\_choroba\_id),ulga\_id:toInteger(row.ulga\_id)});

LOAD CSV WITH HEADERS FROM 'file:///pozycje\_recept.csv' AS row

CREATE (:Pozycje\_recept{pozycje\_recept\_id:toInteger(row.pozycje\_recept\_id),recepta\_id:toInteger(row.recepta\_id),nazwa:row.nazwa,ilosc:toInteger(row.ilosc),odplatnosc:toFloat(row.odplatnosc)});

LOAD CSV WITH HEADERS FROM 'file:///statusy\_wizyt.csv' AS row

CREATE (:Statusy\_wizyt{statusy\_wizyt\_id:toInteger(row.statusy\_wizyt\_id),status:row.status,opis:row.opis});

LOAD CSV WITH HEADERS FROM 'file:///zabiegi.csv' AS row

CREATE (:Zabiegi{zabieg\_id:toInteger(row.zabieg\_id),nazwa:row.nazwa,cena\_netto:toInteger(row.cena\_netto),pracownik\_id:toInteger(row.pracownik\_id),wizyta\_id:toInteger(row.wizyta\_id)});

*//Relacje*

MATCH(s:Stanowiska) MATCH(u:Uprawnienia)

WHERE u.uprawnienie\_id = s.uprawnienie\_id

CREATE (s)-[:fk\_Stanowiska\_Uprawnienia]->(u);

MATCH(c:Choroby) MATCH(k:Karty)

WHERE c.pesel\_id = k.pesel\_id

CREATE (c)-[:fk\_Choroby\_Karty]->(k);

MATCH(p:Placowki)MATCH(a:Adresy)MATCH(k:Kontakty)

WHERE p.adres\_id = a.adres\_id AND p.kontakt\_id = k.kontakt\_id

CREATE (p)-[:fk\_Placowki\_Adresy]->(a),(p)-[:fk\_Placowki\_Kontakty]->(k);

MATCH(p:Pracownicy)MATCH(a:Adresy)MATCH(k:Kontakty)MATCH(st:Stanowiska)

WHERE p.adres\_id = a.adres\_id AND p.kontakt\_id = k.kontakt\_id AND p.stanowisko\_id = st.stanowisko\_id

CREATE (p)-[:fk\_Pracownicy\_Adresy]->(a),(p)-[:fk\_Pracownicy\_Kontakty]->(k),(p)-[:fk\_Pracownicy\_Stanowiska]->(st);

MATCH(p:Pracownicy)MATCH(sp:Specjalnosci)

WHERE p.specjalnosc\_id = sp.specjalnosc\_id

CREATE (p)-[:fk\_Pracownicy\_Specjalnosci]->(sp);

MATCH(p:Pacjenci)MATCH(a:Adresy)MATCH(ko:Kontakty)MATCH(ka:Karty)

WHERE p.adres\_id = a.adres\_id AND p.kontakt\_id = ko.kontakt\_id AND p.pesel\_id = ka.pesel\_id

CREATE (p)-[:fk\_Pacjenci\_Adresy]->(a),(p)-[:fk\_Pacjenci\_Kontakty]->(ko),(p)-[:fk\_Pacjenci\_Karty]->(ka);

MATCH(g:Gabinety)MATCH(p:Pracownicy)MATCH(k:Kontakty)MATCH(pl:Placowki)

WHERE g.kontakt\_id = k.kontakt\_id AND g.pracownik\_id = p.pracownik\_id AND g.placowka\_id = pl.placowka\_id

CREATE (g)-[:fk\_Gabinety\_Kontakty]->(k),(g)-[:fk\_Gabinety\_Pracownicy]->(p),(g)-[:fk\_Gabinety\_Placowki]->(pl);

MATCH(w:Wizyty)MATCH(g:Gabinety)MATCH(ps:Pracownicy)MATCH(pu:Pracownicy)MATCH(pac:Pacjenci)

WHERE w.gabinet\_id = g.gabinet\_id AND w.prac\_spec = ps.pracownik\_id AND w.prac\_uma = pu.pracownik\_id AND w.pacjent\_id = pac.pacjent\_id

CREATE (w)-[:fk\_Wizyty\_Gabinety]->(g),(w)-[:fk\_Wizyty\_Pacjenci]->(pac),(w)-[:fk\_Wizyty\_Pracownicy\_Spec]->(ps),

(w)-[:fk\_Wizyty\_Pracownicy\_Uma]->(pu);

MATCH(r:Recepty)MATCH(p:Pracownicy)MATCH(w:Wizyty)MATCH(o:Oddzialy\_nfz)MATCH(c:Choroby)

WHERE r.pracownik\_id = p.pracownik\_id AND r.wizyta\_id = w.wizyta\_id AND r.oddzial\_nfz\_id = o.oddzial\_nfz\_id AND

r.recepta\_choroba\_id = c.choroby\_id

CREATE (r)-[:fk\_Recepty\_Pracownicy]->(p),(r)-[:fk\_Recepty\_Wizyty]->(w),(r)-[:fk\_Recepty\_Oddzialy]->(o),(r)-[:fk\_Recepty\_Choroby]->(c);

MATCH(r:Recepty)MATCH(u:Ulgi)

WHERE r.ulga\_id = u.ulgi\_id

CREATE (r)-[:fk\_Recepty\_Ulgi]->(u);

MATCH(pr:Pozycje\_recept)MATCH(r:Recepty)

WHERE pr.recepta\_id = r.recepta\_id

CREATE (pr)-[:fk\_Pozycje\_Recepty]->(r);

MATCH(w:Wizyty)MATCH(s:Statusy\_wizyt)

WHERE w.wizyta\_id = s.statusy\_wizyt\_id

CREATE (s)-[:fk\_Statusy\_Wizyty]->(w);

MATCH(z:Zabiegi)MATCH(w:Wizyty)MATCH(p:Pracownicy)

WHERE z.wizyta\_id = w.wizyta\_id AND z.pracownik\_id = p.pracownik\_id

CREATE (z)-[:fk\_Zabiegi\_Wizyty]->(w),(z)-[:fk\_Zabiegi\_Pracownicy]->(p);

* 1. **Skrypty zapytań**

*//łączna wartość roczna za każdy zabieg przeprowadzany na pacjentach z grupą krwi A-*

MATCH(z:Zabiegi)-[:fk\_Zabiegi\_Wizyty]->(w:Wizyty)-[:fk\_Wizyty\_Pacjenci]->(p:Pacjenci)-[:fk\_Pacjenci\_Karty]->(k:Karty)

WHERE k.grupa\_krwi = "A-"

RETURN z.nazwa,date(w.data\_wizyty).year as rok,k.grupa\_krwi,

SUM(z.cena\_netto) AS SumaZaZabiegi

ORDER BY z.nazwa,rok DESC;

*//Id pacjenta,jego imie ,nazwisko , pesel oraz jego Wydatki na leki wciągu jednego roku z ulgą/bez ulgi*

MATCH(r:Recepty)-[:fk\_Recepty\_Wizyty]->(w:Wizyty)-[:fk\_Wizyty\_Pacjenci]->(p:Pacjenci)

MATCH(pz:Pozycje\_recept)-[:fk\_Pozycje\_Recepty]->(r)

OPTIONAL MATCH(r)-[:fk\_Recepty\_Ulgi]->(u:Ulgi)

RETURN p.pacjent\_id ,p.imie ,p.nazwisko,p.pesel\_id,date(w.data\_wizyty).year AS rok,

SUM(pz.odplatnosc) AS WydatkiNaLekiBezUlgi,(SUM(pz.odplatnosc) \* u.procent\_ulgi / 100) AS WydatkiNaLeki\_z\_Ulgą ORDER BY p.pacjent\_id ASC;

*//Sumaryczna kwota za wizyty w danej placówce w danym roku*

MATCH(w:Wizyty)-[:fk\_Wizyty\_Gabinety]->(g:Gabinety)-[:fk\_Gabinety\_Placowki]->(p:Placowki)-[:fk\_Placowki\_Adresy]->(a:Adresy)

call{

    WITH p,a,w

    RETURN p.placowka\_id AS placowka\_id, p.nazwa AS placowka,a.miasto AS miasto,date(w.data\_wizyty).year AS rok,SUM(w.oplata) AS Suma\_w\_DanymRoku

}

RETURN placowka\_id,placowka,miasto,rok,SUM(Suma\_w\_DanymRoku) AS Suma\_w\_DanymRoku

ORDER BY placowka\_id ASC,rok DESC;

*//Średnia opłat za wizyty w każdym roku, pacjentów pochodzących z danego miasta*

MATCH (w:Wizyty)-[:fk\_Wizyty\_Pacjenci]->(p:Pacjenci)-[:fk\_Pacjenci\_Adresy]->(a:Adresy)

RETURN date(w.data\_wizyty).year AS rok, a.miasto, avg(w.oplata) ORDER BY rok

*//Suma dochodów z zabiegów w każdym gabinecie*

MATCH (z:Zabiegi)-[:fk\_Zabiegi\_Wizyty]->(w:Wizyty)-[:fk\_Wizyty\_Gabinety]->(g:Gabinety)

RETURN w.gabinet\_id AS identyfikator\_gabinetu, sum(z.cena\_netto)

ORDER BY identyfikator\_gabinetu

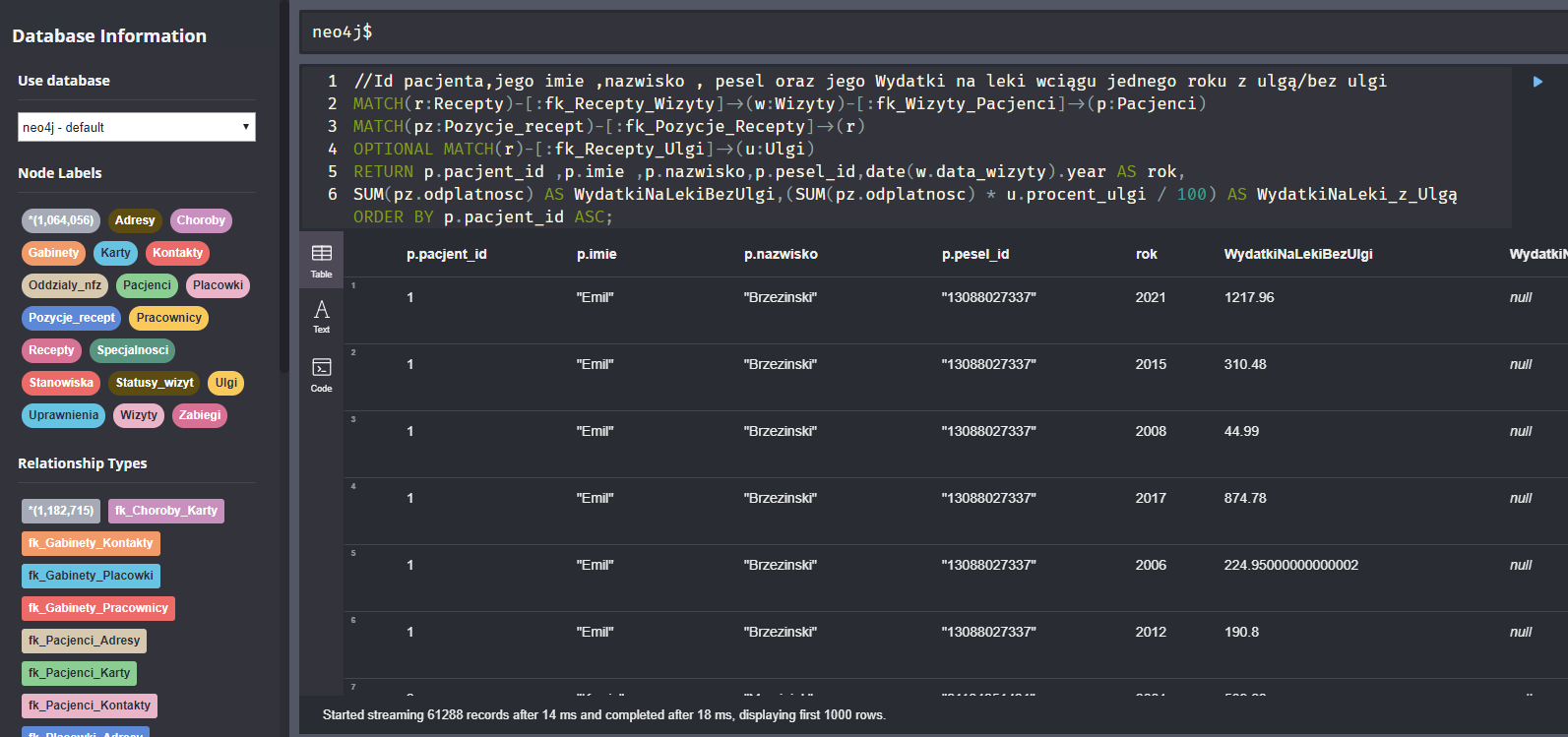
*//Suma opłat za leki z każdej recepty, w danym roku*

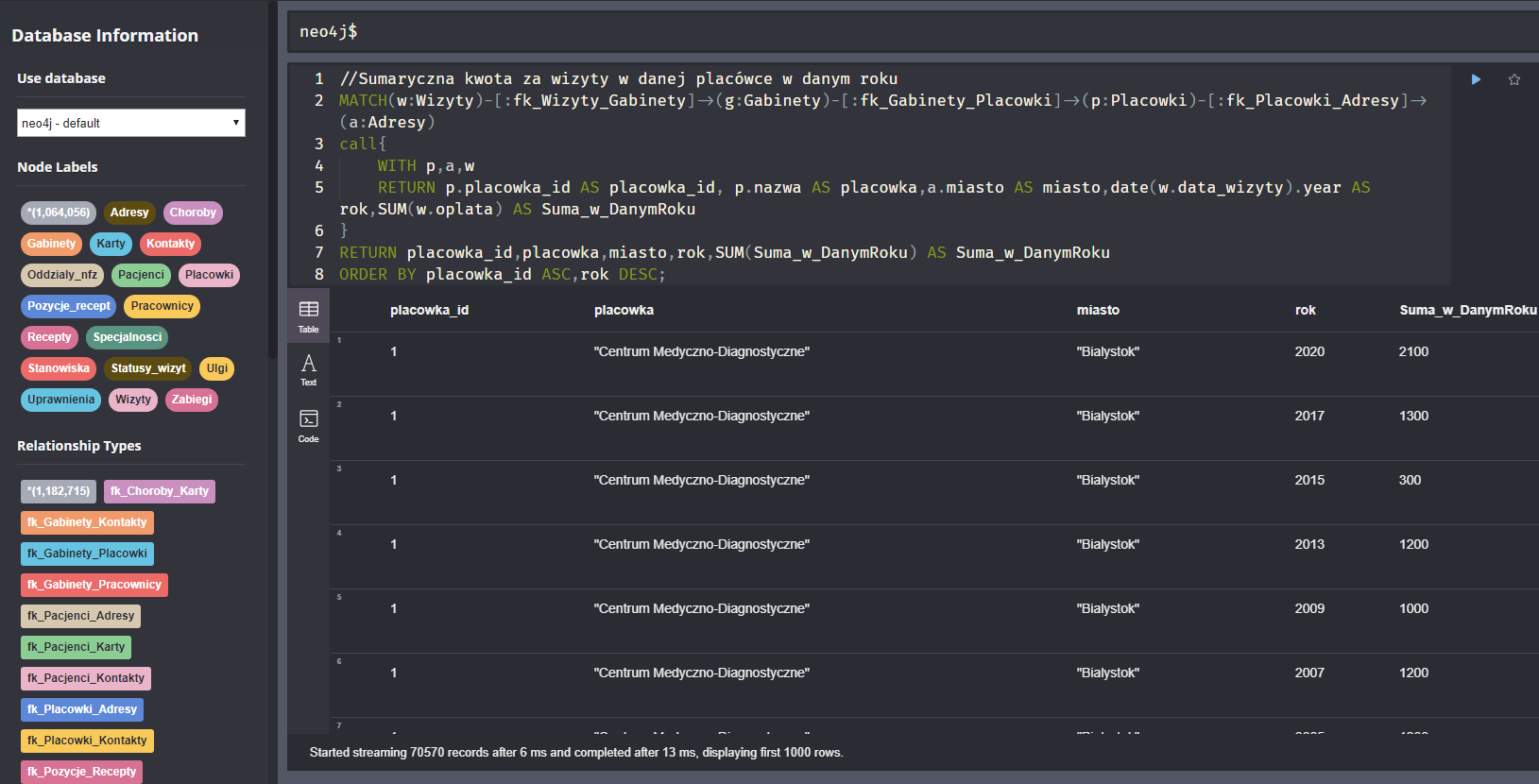
MATCH (pr:Pozycje\_recept)-[:fk\_Pozycje\_Recepty]->(r:Recepty)-[:fk\_Recepty\_Wizyty]->(w:Wizyty)

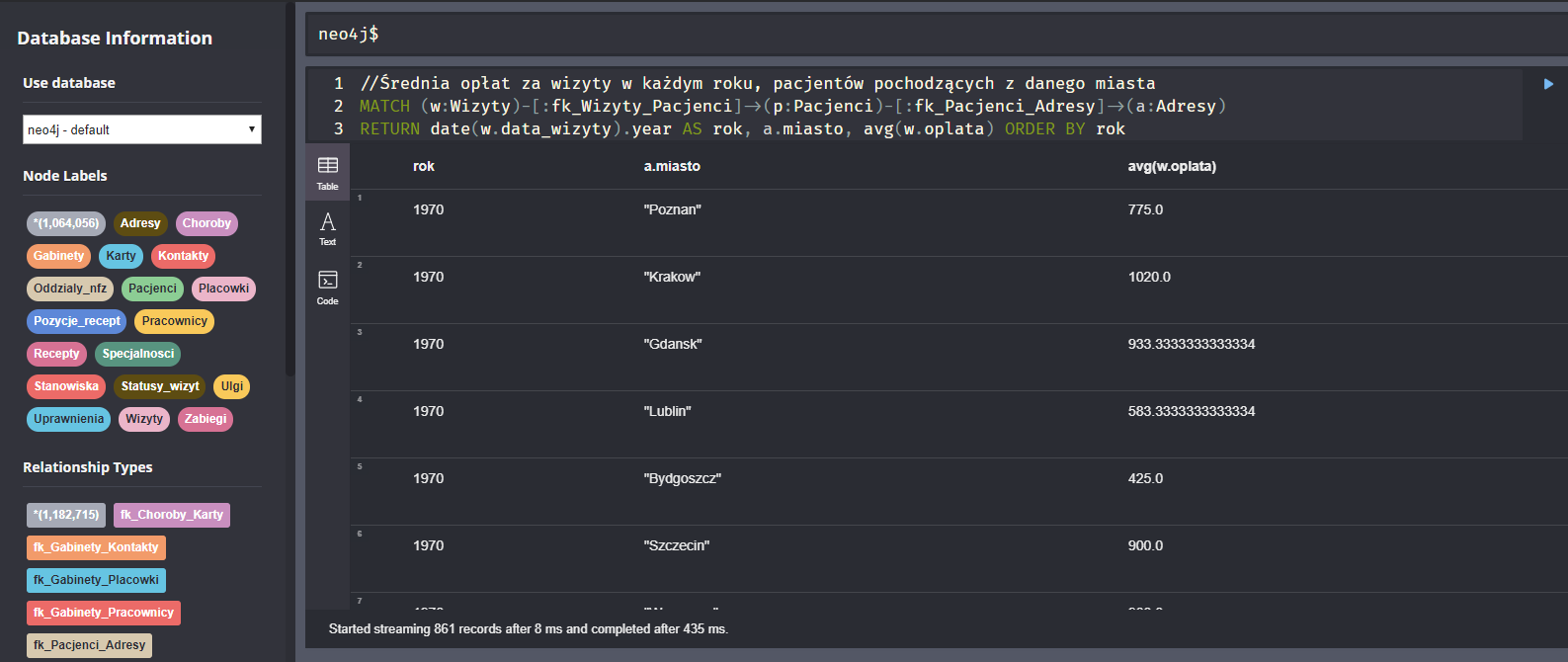
RETURN pr.recepta\_id AS id\_recepty, date(w.data\_wizyty).year AS rok, sum(pr.odplatnosc) AS suma ORDER BY rok

1. **Wyniki wykonania skryptów**

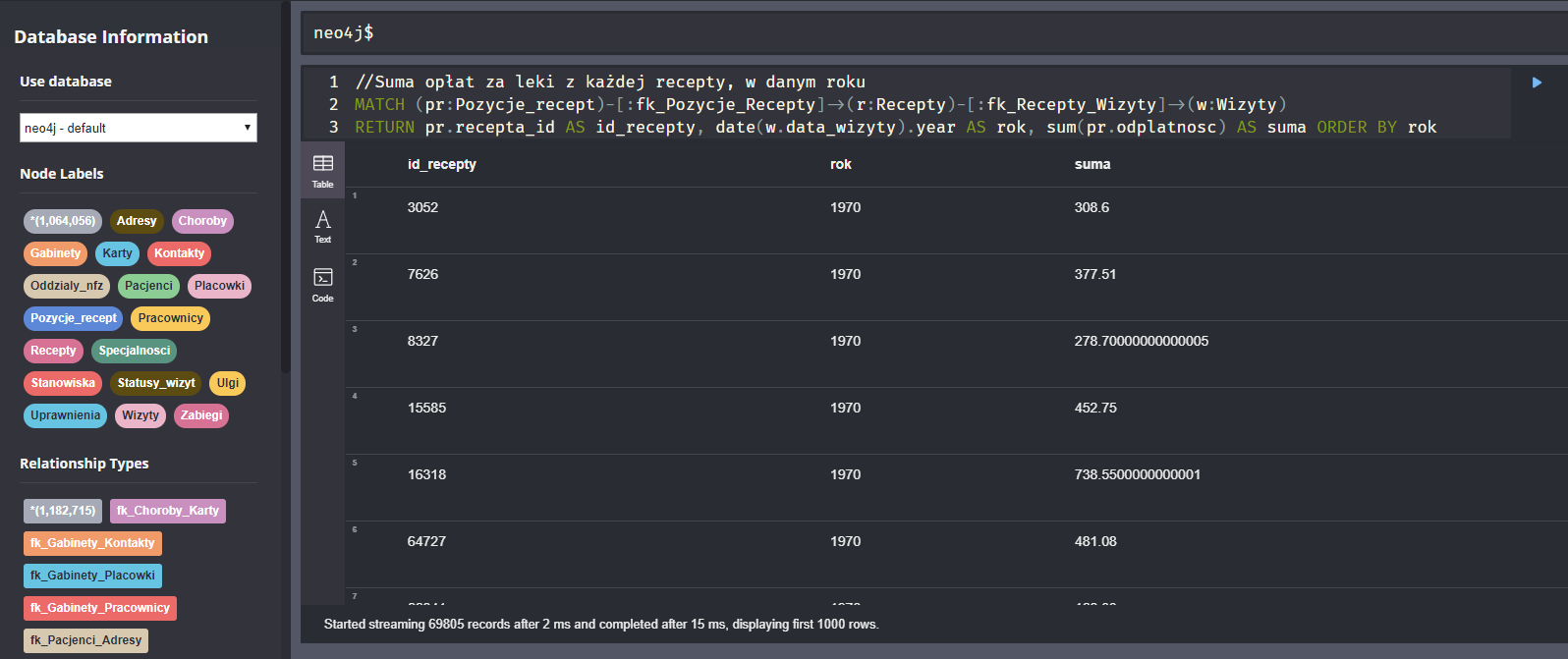
****

****

****

****

****

****

1. **Graficzne przedstawienie bazy**
   1. **Opis węzłów:**

Każdy typ węzła przedstawia określoną tabele z bazy danych. Dodatkowo węzły te noszą tę samą nazwę co dana tabela.

**Adresy –** węzeł zawierający pola z tabeli Adresy

**Choroby -** węzeł zawierający pola z tabeli Choroby

**Gabinety -** węzeł zawierający pola z tabeli Gabinety

**Karty -** węzeł zawierający pola z tabeli Karty

**Kontakty -** węzeł zawierający pola z tabeli Kontakty

**Oddzialy\_nfz -** węzeł zawierający pola z tabeli Oddzialy\_nfz

**Pacjenci -** węzeł zawierający pola z tabeli Pacjenci

**Placowki -** węzeł zawierający pola z tabeli Placowki

**Pozycje\_recept -** węzeł zawierający pola z tabeli Pozycje\_recept

**Pracownicy -** węzeł zawierający pola z tabeli Pracownicy

**Recepty -** węzeł zawierający pola z tabeli Recepty

**Specjalnosci -** węzeł zawierający pola z tabeli Specjalnosci

**Stanowiska -** węzeł zawierający pola z tabeli Stanowiska

**Statusy\_wizyt -** węzeł zawierający pola z tabeli Statusy\_wizyt

**Ulgi -** węzeł zawierający pola z tabeli Ulgi

**Uprawnienia -** węzeł zawierający pola z tabeli Uprawnienia

**Wizyty -** węzeł zawierający pola z tabeli Wizyty

**Zabiegi -** węzeł zawierający pola z tabeli Zabiegi

* 1. **Opis relacji:**

Każdy typ relacji odwzorowuje połączenie między dwoma tabelami za pomocą kluczy obcych i kluczy głównych. Klucze te zostały zawarte w polach danego węzła.

**Fk\_Stanowiska\_Uprawnienia –** połączenie klucza obcego węzła typu Stanowiska, z kluczem głównym węzła typu Uprawnienia

**Fk\_Choroby\_Karty –** połączenie klucza obcego węzła typu Choroby, z kluczem głównym węzła typu Karty

**Fk\_Placowki\_Adresy -** połączenie klucza obcego węzła typu Placowki, z kluczem głównym węzła typu Adresy

**Fk\_Placowki\_Kontakty -** połączenie klucza obcego węzła typu Placowki, z kluczem głównym węzła typu Kontakty

**Fk\_Pracownicy\_Adresy -** połączenie klucza obcego węzła typu Pracownicy, z kluczem głównym węzła typu Adresy

**Fk\_Pracownicy\_Kontakty -** połączenie klucza obcego węzła typu Pracownicy, z kluczem głównym węzła typu Kontakty

**Fk\_Pracownicy\_Stanowiska -** połączenie klucza obcego węzła typu Pracownicy, z kluczem głównym węzła typu Stanowiska

**Fk\_Pracownicy\_Specjalnosci -** połączenie klucza obcego węzła typu Pracownicy, z kluczem głównym węzła typu Specjalnosci

**Fk\_Pacjenci\_Adresy -** połączenie klucza obcego węzła typu Pacjenci, z kluczem głównym węzła typu Adresy

**Fk\_Pacjneci\_Kontakty -** połączenie klucza obcego węzła typu Pacjenci, z kluczem głównym węzła typu Kontakty

**Fk\_Pacjenci\_Karty -** połączenie klucza obcego węzła typu Pacjenci, z kluczem głównym węzła typu Karty

**Fk\_Gabinety\_Kontakty –** połączenie klucza obcego węzła typu Gabinety, z kluczem głównym węzła typu Kontakty

**Fk\_Gabinety\_Pracownicy -** połączenie klucza obcego węzła typu Gabinety, z kluczem głównym węzła typu Pracownicy

**Fk\_Gabinety\_Placowki -** połączenie klucza obcego węzła typu Gabinety, z kluczem głównym węzła typu Placowki

**Fk\_Wizyty\_Gabinety -** połączenie klucza obcego węzła typu Wizyty, z kluczem głównym węzła typu Gabinety

**Fk\_Wizyty\_Pacjneci -** połączenie klucza obcego węzła typu Wizyty, z kluczem głównym węzła typu Pacjneci

**Fk\_Wizyty\_Pracownicy\_Spec -** połączenie klucza obcego węzła typu Wizyty, z kluczem głównym węzła typu Pracownicy

**Fk\_Wizyty\_Pracownicy\_Uma -** połączenie klucza obcego węzła typu Wizyty, z kluczem głównym węzła typu Pracownicy

**Fk\_Recepty\_Pracownicy -** połączenie klucza obcego węzła typu Recepty, z kluczem głównym węzła typu Pracownicy

**Fk\_Recepty\_Wizyty -** połączenie klucza obcego węzła typu Recepty, z kluczem głównym węzła typu Wizyty

**Fk\_Recepty\_Oddzialy -** połączenie klucza obcego węzła typu Recepty, z kluczem głównym węzła typu Oddzialy\_nfz

**Fk\_Recepty\_Choroby -** połączenie klucza obcego węzła typu Recepty, z kluczem głównym węzła typu Choroby

**Fk\_Recepty\_Ulgi -** połączenie klucza obcego węzła typu Recepty, z kluczem głównym węzła typu Ulgi

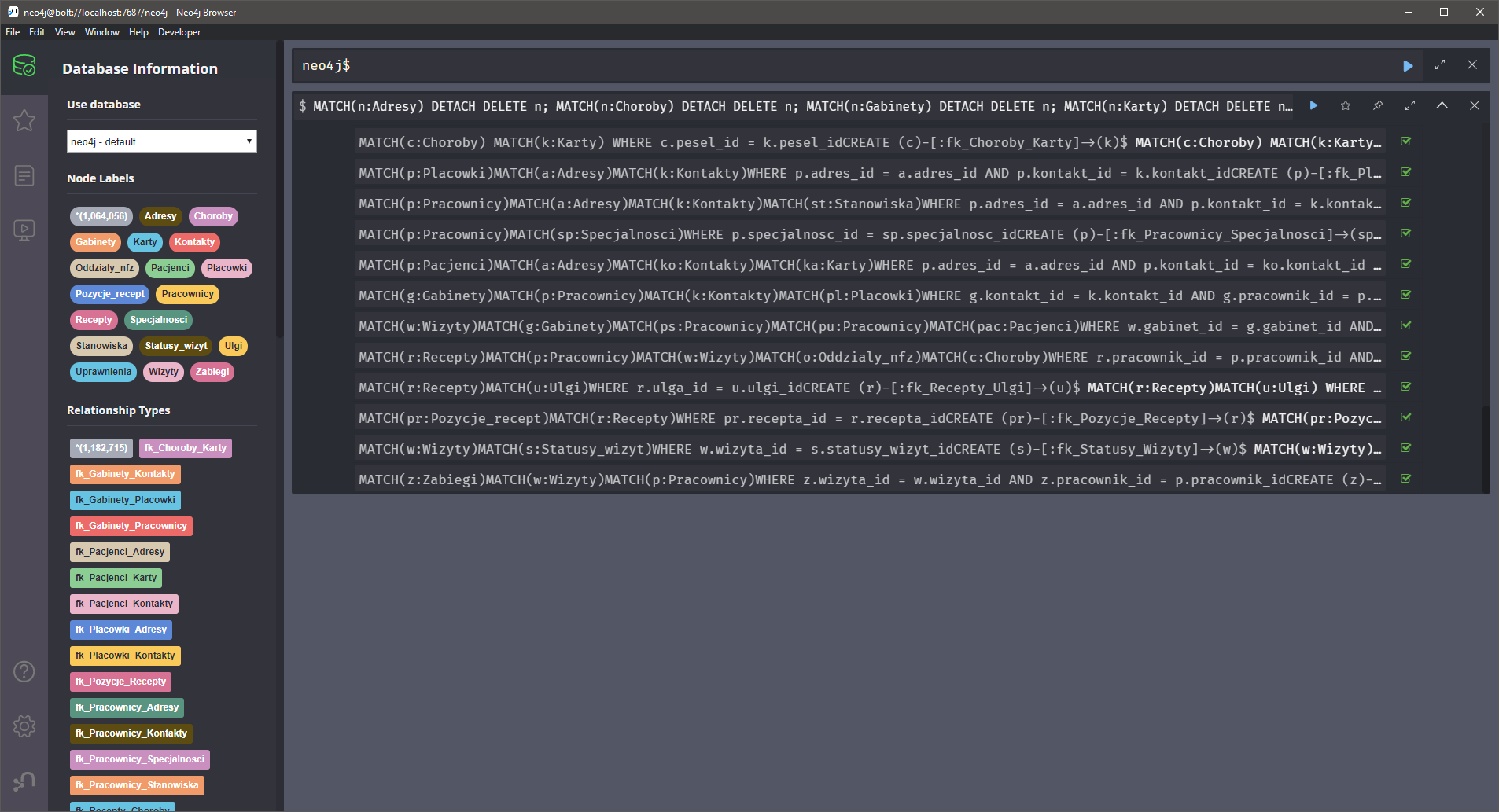
**Fk\_Pozycje\_Recepty -** połączenie klucza obcego węzła typu Pozycje\_recept, z kluczem głównym węzła typu Recepty

**Fk\_Statusy\_Wizyty -** połączenie klucza obcego węzła typu Statusy\_wizyt, z kluczem głównym węzła typu Wizyty

**Fk\_Zabiegi\_Wizyty -** połączenie klucza obcego węzła typu Zabiegi, z kluczem głównym węzła typu Wizyty

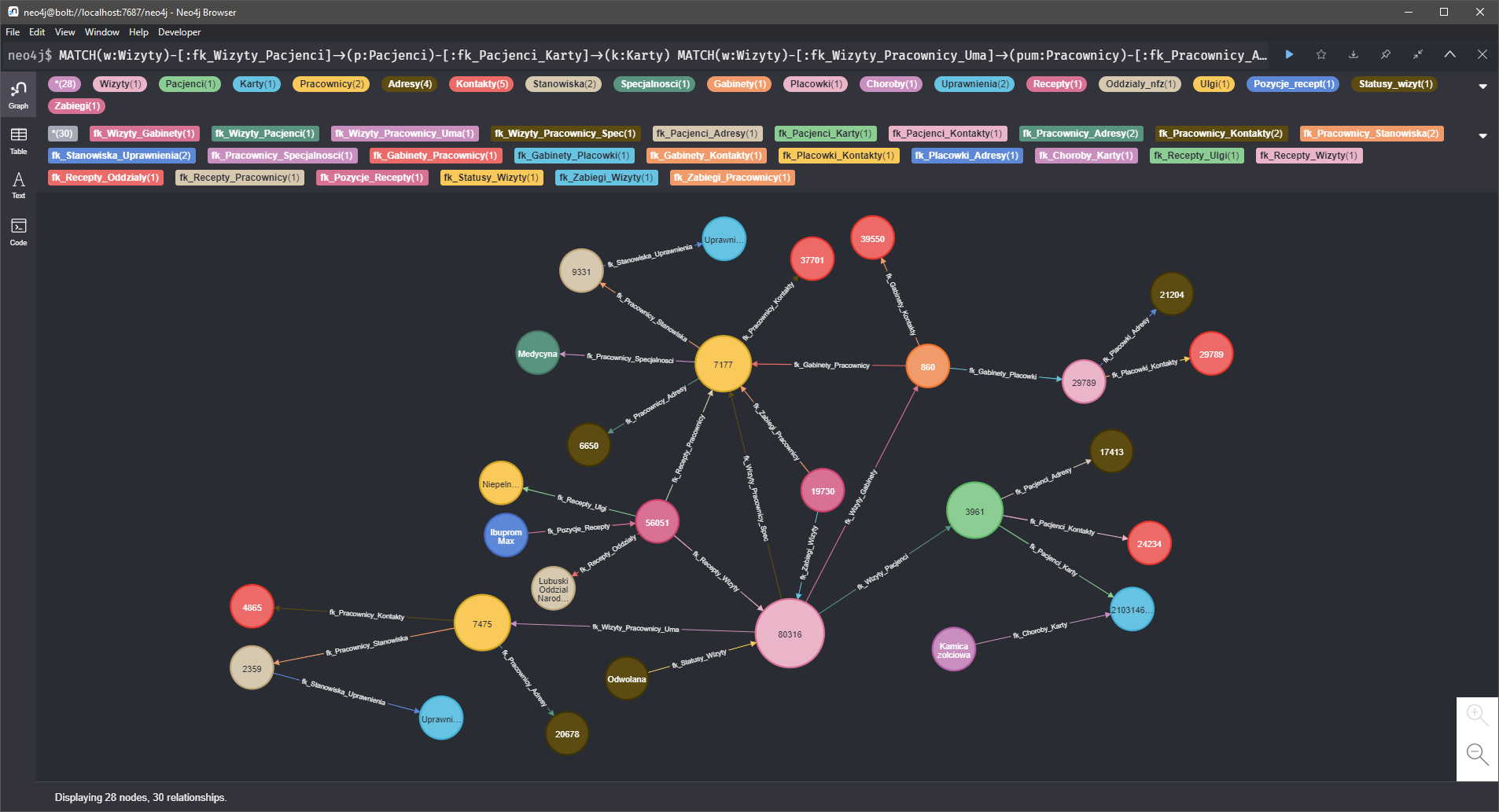
**Fk\_Zabiegi\_Pracownicy -** połączenie klucza obcego węzła typu Zabiegi, z kluczem głównym węzła typu Pracownicy

* 1. **Sposób tworzenia bazy**
     1. Przeniesienie plików .csv do pod folderu import dla projektu w neo4j
     2. Uruchomienie skryptów create.cyp



Po wykonaniu skryptów z create.cyp mamy bazę grafową z ponad milionem węzłów i relacji.

* 1. **Przykładowe powiązanie danych**



Podgląd 1 węzła wizyty wraz z resztą węzłów z całej bazy skorelowanych relacjami z nim.