Wojskowa Akademia Techniczna

Hurtownie Danych

Sprawozdanie z projektu

Prowadzący: dr inż. Marcin Mazurek

Wykonawcy: Dominik Marchewka

Piotr Skomorowski

Grupa: I7B1S1

1. Zadanie projektowe

Wykorzystując dane o rozprzestrzenianiu się w koronowirusa zbudować hurtownię danych umożliwiającą analizę tempa rozprzestrzeniania się wirusa:

Wymiary analizy:

- Geografia – kraj (atrybuty: populacja, GDP) – kontynent,
- Czas – dzień – miesiąc – rok,
- Czas od pierwszej detekcji (numer kolejny dnia),
- Pacjent (wobec braku danych należy je wygenerować losowo lub próbki danych z Kaggle) (wiek, płeć).

Miary:

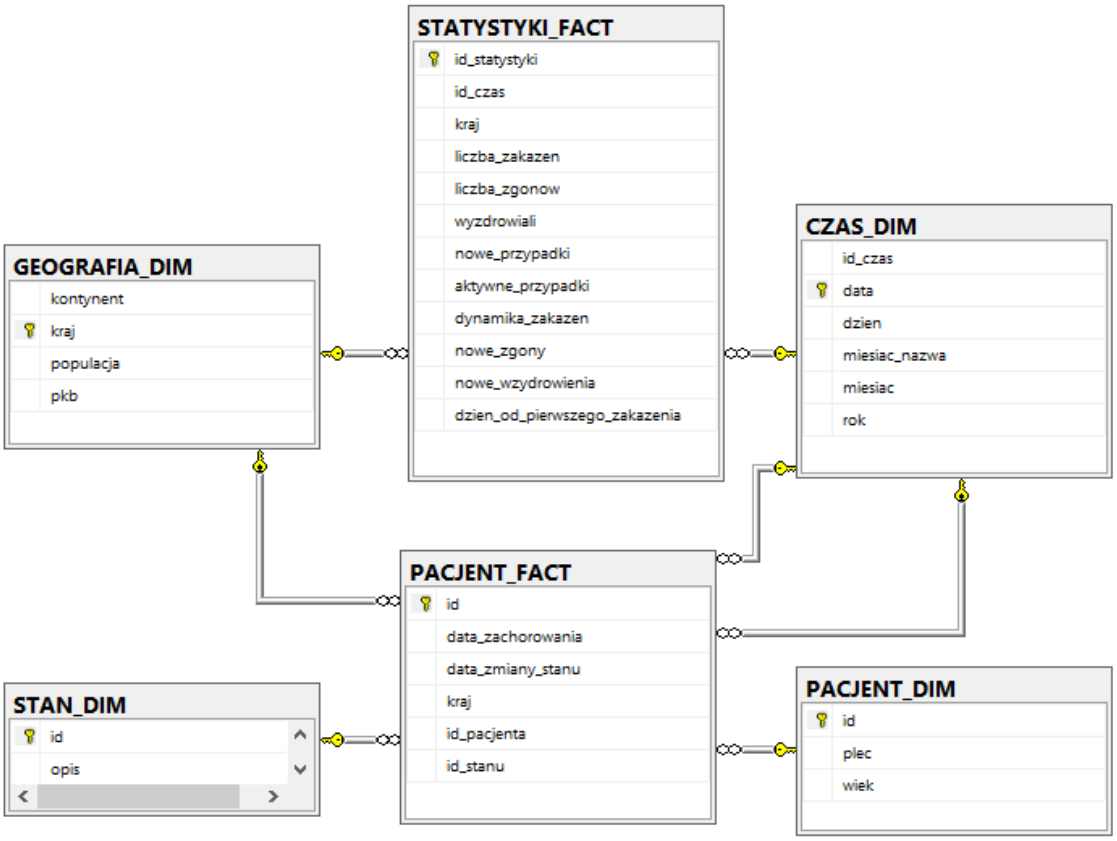
- liczba zakażeń (w okresie),
- liczba zgonów (w okresie),
- liczba pacjentów wyleczonych (w okresie),
- liczba nowych przypadków zakażeń (granulacja dzienna),
- liczba pacjentów zakażonych (stan na dzień),
- dynamika zakażeń – liczba nowych przypadków / liczba pacjentów zakażonych w dniu poprzedzającym.

2. Opis założeń biznesowych

Hurtownia danych ma umożliwiać przyrostowe dodawanie danych na temat rozprzestrzeniania się choroby COVID – 19. Zostanie ona utworzona za pomocą SQL Server 2019. Hurtownia powinna przechowywać informację na temat liczby zakażonych, zgonów, wyzdrowiałych, dynamiki zakażeń, liczny dni od zakażenia oraz informacje na temat pojedynczego pacjenta, w zależności od geografii (kraj, kontynent) oraz daty. Proces ETL powinien być uruchamiany przez administratora. Hurtownia ma umożliwiać tworzenie raportów na temat pandemii koronawirusa w programie PoewrBI.

3. Model bazy danych

Do realizacji zadani utworzono poniższą bazę danych:



Zawiera ona następujące tabele wymiarów:

- CZAS_DIM – wymiar czasu,
- GEOGRAFIA_DIM – wymiar obszarów geograficznych,
- STAN_DIM – wymiar stanów pacjenta,
- PACJENT_DIM – wymiar pacjenta,

oraz tabele faktów:

- STATYSTYKI_FACT – tabela statystyk rozprzestrzeniania się wirusa,
- PACJENT_FACT – tabela faktów pojedynczego pacjent.

4. Proces ETL

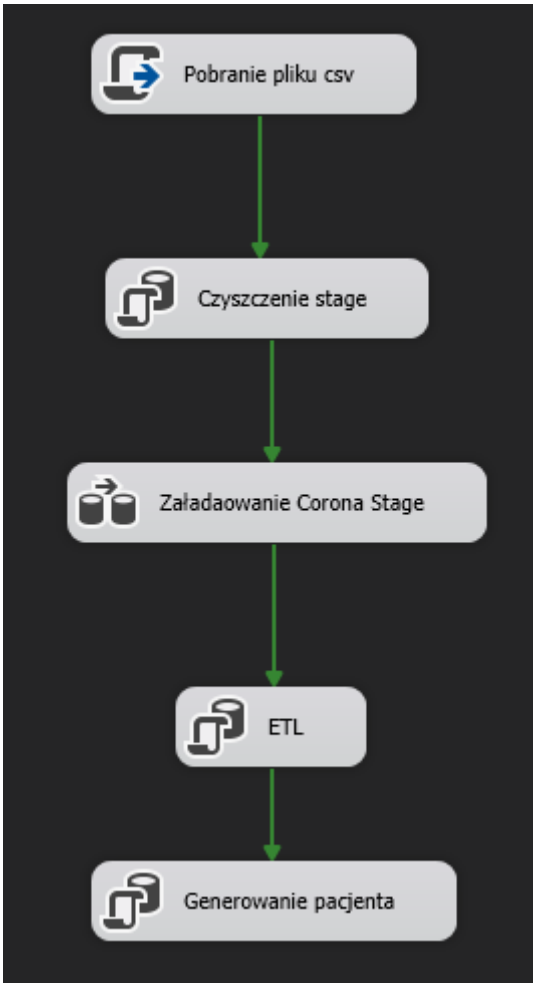
Dane w postaci pliku csv, na temat pandemii koronawirusa pochodzą ze strony:

https://raw.githubusercontent.com/datasets/covid-19/master/data/countries-aggregated.csv

Zawierają one następujące kolumny danych:

- Date - data
- Country - kraj
- Confirmed – ilość potwierdzonych przypadków danego dnia
- Recovered – ilość wyzdrowiałych osób danego dnia
- Deaths – ilość zgonów danego dnia

Implementacja procesu ETL:



Dane są automatycznie pobierane za pomocą skryptu, napisanego w języku C#. Poniżej zamieszczono jego kod:

```
#region Namespaces
using System;
using System.Data;
using Microsoft.SqlServer.Dts.Runtime;
using System.Windows.Forms;
#endregion

namespace ST_7aa9878bb087441194bc5bea0748c372
{

    [Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute]
    0 references
    public partial class ScriptMain : Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase
    {
        0 references
        public void Main()
        {


            try
            {
                bool fireAgain1 = true;
                Dts.Events.FireInformation(0, subComponent: "Download File", description: "Start downloading " + Dts.Connections["HTTP Connection Manager 2"].
                    ConnectionString, helpFile: string.Empty, helpContext: 0, ref fireAgain1);

                Object mySSISConnection1 = Dts.Connections["HTTP Connection Manager 2"].AcquireConnection(txn: null);
                HttpClientConnection myConnection1 = new HttpClientConnection(mySSISConnection1);
                myConnection1.DownloadFile(Dts.Connections["pobrane2.csv"].ConnectionString, overwriteDestination: true);


                Dts.Events.FireInformation(0, subComponent: "Download File", description: "Finished downloading " + Dts.Connections["pobrane2.csv"].ConnectionString
                    , helpFile: string.Empty, helpContext: 0, ref fireAgain1);

                Dts.TaskResult = (int)ScriptResults.Success;
            }
            catch (Exception ex)
            {
                Dts.Events.FireError(0, subComponent: "Download File", description: "Download failed: " + ex.Message, helpFile: string.Empty, helpContext: 0);

                Dts.TaskResult = (int)ScriptResults.Failure;
            }
        }

        2 references
        enum ScriptResults
        {
            Success = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Success,
            Failure = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Failure
        };
    }
}
```

Następnie czyszczona jest tablica FILE_CORONA_REC_STAGE, za pomocą skryptu w SQL:

```
use Coronavirus
delete from FILE_CORONA_REC_STAGE
```

Następnie plik csv jest importowany do pustej tabeli STAGE, przy pomocy Data Flow Task.

W następnej kolejności wykonywany jest skrypt SQL pod nazwą ETL. W pierwszej kolejności usuwana jest tabela IleOdPierwszego i w tabeli FILE_CORONA_REC_STAGE, zamieniana jest „Korea, South” na ‘South Korea’.

```
use Coronavirus;
GO
ALTER TABLE  STATYSTYKI_FACT drop column IF EXISTS IleOdPierwszego;
UPDATE       FILE_CORONA_REC_STAGE
SET                country = 'South Korea',
                    confirmed =  recovered ,
                        recovered = left(deaths, charindex(',', deaths) - 1) ,
                        deaths = RIGHT(deaths,LEN(deaths) - CHARINDEX(',',deaths))


WHERE        (country = '"Korea')
;
```

W następnej kolejności tworzone są tabele pomocnicze, gdzie konwertowane są dane oraz wyliczane potrzebne statystyki.

```
DROP TABLE IF EXISTS tmp;
DROP TABLE IF EXISTS tmp2;

create table tmp (
id_czas varchar(255),
kraj varchar(255),
liczba_zakazen varchar(255),
liczba_zgonow varchar(255),
wyzdrowiali varchar(255));
GO

insert into tmp (id_czas, kraj, liczba_zakazen, wyzdrowiali, liczba_zgonow)
select *
from FILE_CORONA_REC_STAGE;
GO

alter table tmp
alter column id_czas date;
alter table tmp
alter column liczba_zakazen bigint;
alter table tmp
alter column wyzdrowiali bigint;
alter table tmp
alter column liczba_zgonow bigint;
GO

alter table tmp
add nowe_przypadki bigint;
GO

alter table tmp
add nowe_wyzdrowienia bigint;
GO

alter table tmp
add nowe_zgony bigint;
GO

update tmp
set tmp.nowe_przypadki=tmp.liczba_zakazen
from tmp;
GO
```

```
update tmp
SET tmp.nowe_przypadki=tmp.liczba_zakazen-FILE_CORONA_REC_STAGE.confirmed
FROM FILE_CORONA_REC_STAGE
WHERE FILE_CORONA_REC_STAGE.country=tmp.kraj
AND tmp.id_czas=DATEADD(day, 1, FILE_CORONA_REC_STAGE.date);
GO

update tmp
set tmp.nowe_wyzdrowienia=tmp.wyzdrowiali
from tmp;
GO

update tmp
SET tmp.nowe_wyzdrowienia=tmp.wyzdrowiali-FILE_CORONA_REC_STAGE.recovered
FROM FILE_CORONA_REC_STAGE
WHERE FILE_CORONA_REC_STAGE.country=tmp.kraj
AND tmp.id_czas=DATEADD(day, 1, FILE_CORONA_REC_STAGE.date);
GO

update tmp
set tmp.nowe_zgony=tmp.liczba_zgonow
from tmp;
GO

update tmp
SET tmp.nowe_zgony=tmp.liczba_zgonow-FILE_CORONA_REC_STAGE.deaths
FROM FILE_CORONA_REC_STAGE
WHERE FILE_CORONA_REC_STAGE.country=tmp.kraj
AND tmp.id_czas=DATEADD(day, 1, FILE_CORONA_REC_STAGE.date);
GO

alter table tmp
add dynamika_zakazen float;
GO

update tmp
set dynamika_zakazen=0
from tmp;
GO

create table tmp2 (id_czas date,
kraj varchar(255),
liczba_zakazen bigint,
liczba_zgonow bigint,
wyzdrowiali bigint,
nowe_przypadki bigint,
dynamika_zakazen float);
insert into tmp2
select id_czas, kraj, liczba_zakazen, liczba_zgonow, wyzdrowiali, nowe_przypadki, dynamika_zakazen
from tmp;
GO

alter table tmp
alter column nowe_przypadki float;
GO

alter table tmp2
alter column nowe_przypadki float;
GO
```

```
update tmp
SET tmp.dynamika_zakazen=ISNULL(tmp.nowe_przypadki/NULLIF(tmp2.nowe_przypadki,0),tmp.nowe_przypadki)
FROM tmp2
WHERE tmp.kraj=tmp2.kraj
AND tmp.id_czas=DATEADD(day, 1, tmp2.id_czas);
GO

alter table tmp
add id varchar(255);
GO

update tmp
SET id = concat(id_czas, kraj);
GO

alter table tmp
add aktywne_przypadki bigint;
GO

update tmp
set aktywne_przypadki = (liczba_zakazen - liczba_zgonow) - wyzdrowiali;
GO

alter table tmp
add dzien_od_pierwszego_zakazenia int;
GO
```

Ładownie danych do tabeli STATYSTKI_FACT i korygowanie ujemnych danych.

```
insert into STATYSTYKI_FACT
select tmp.id, tmp.id_czas, tmp.kraj,
tmp.liczba_zakazen, tmp.liczba_zgonow, tmp.wyzdrowiali,
tmp.nowe_przypadki, tmp.aktywne_przypadki, tmp.dynamika_zakazen,
tmp.nowe_zgony, tmp.nowe_wyzdrowienia, tmp.dzien_od_pierwszego_zakazenia
from tmp
left join STATYSTYKI_FACT
on tmp.id = STATYSTYKI_FACT.id_statystyki
where STATYSTYKI_FACT.id_statystyki is null;
GO

update STATYSTYKI_FACT
SET dzien_od_pierwszego_zakazenia = 1 + DATEDIFF(DAY, b.czasPierwszegoWykrycia,STATYSTYKI_FACT.id_czas)
from (
select  kraj, min([id_czas]) as czasPierwszegoWykrycia
from  STATYSTYKI_FACT
where nowe_przypadki <> 0
Group by kraj
) as b
where STATYSTYKI_FACT.kraj = b.kraj  ;
GO

update STATYSTYKI_FACT
SET dzien_od_pierwszego_zakazenia = 0
from STATYSTYKI_FACT
where dzien_od_pierwszego_zakazenia < 0;
GO

update STATYSTYKI_FACT
set dynamika_zakazen = 0
from STATYSTYKI_FACT
where dynamika_zakazen < 0;
GO

drop table tmp2;
GO

drop table tmp;
GO
```

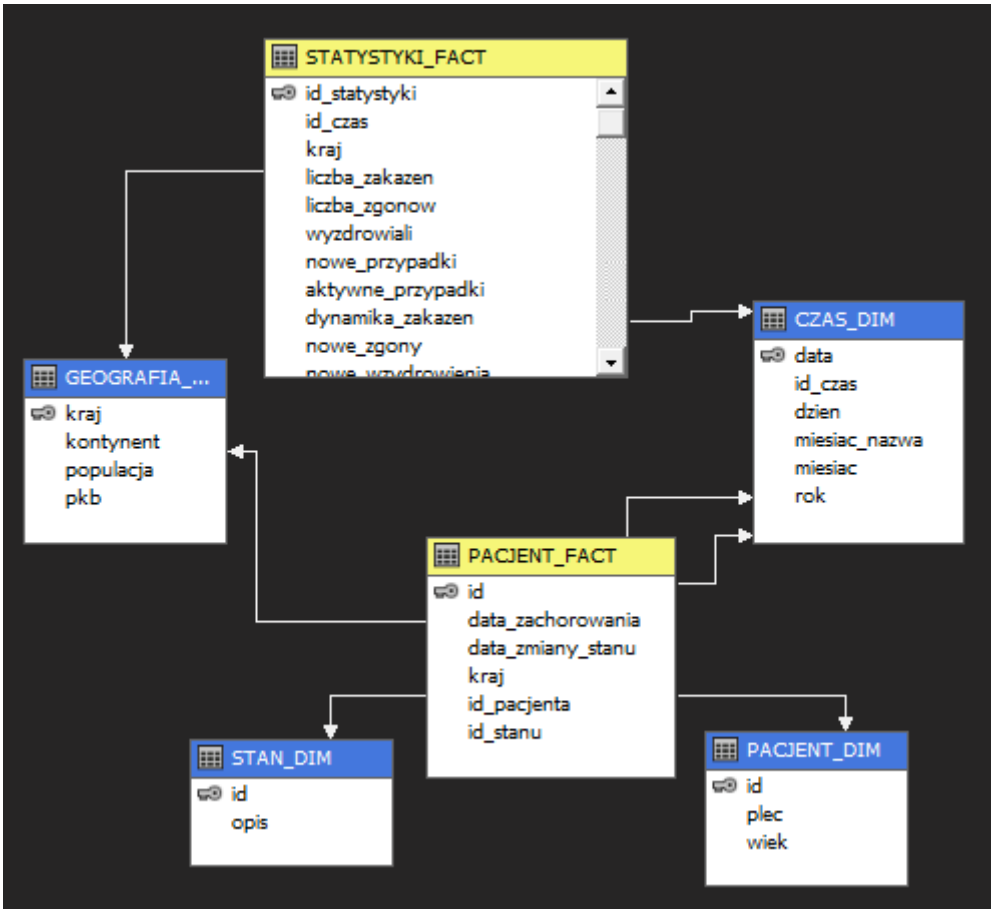
Po wykonaniu skryptu ETL, uruchamiany jest skrypt generowania pacjenta. Pacjent jest generowany w trzech pętlach (daty, kraju, ilości pacjentów). Początkowo do kolumny PACJENT_FACT generowani są nowi pacjenci, wszyscy z id_stanu = p. Następnie przy pomocy tabel pomocniczych edytowana jest tabela PACJENT_FACT poprzez zmienienie id_stanu i data_zmiany_stanu na podstawie nowych wyzdrowiałych oraz nowych zgonów.

```
use Coronavirus;
GO

begin
    declare @PatientAmount as int = (select top 1 sum(nowe_przypadki) from STATYSTYKI_FACT);
    declare @StatystykiAmount as int = (select count(id_statystyki) from STATYSTYKI_FACT);
    declare @DateFirst as varchar(255) = (select top 1 data_zachorowania from PACJENT_FACT order by data_zachorowania desc);
    print @DateFirst;
    declare @DateLast as varchar(255) = (select top 1 id_czas from STATYSTYKI_FACT order by id_czas desc);
    print @DateLast;
    --set @DateLast = '2020-01-30';
    declare @DateAmount as int = DATEDIFF(day, @DateFirst, @DateLast);
    --declare @DateAmount as int = 1 + DATEDIFF(day, @DateFirst, @DateLast);
    print @DateAmount;

    declare @AllDateAmount as int = (1 + (select top 1 count(id_czas) from STATYSTYKI_FACT group by kraj));
    print @AllDateAmount;
    --declare @DateAmount as int = ( (select top 1 count(id_czas) from STATYSTYKI_FACT group by kraj));
    declare @CountryAmount as int = (1 + (select top 1 count(kraj) from STATYSTYKI_FACT group by id_czas));;
    --declare @CounterDate as int = @AllDateAmount - @DateAmount;
    --print @CounterDate

    create table tmpKraj3 (RowKraj int, kraj varchar(255));
    insert into tmpKraj3
    select ROW_NUMBER() over (order by kraj) as RowKraj, kraj from STATYSTYKI_FACT group by kraj;

    create table tmpCzas3 (RowCzas int, czas varchar(255));
    insert into tmpCzas3
    select ROW_NUMBER() over (order by id_czas) as RowCzas, id_czas from STATYSTYKI_FACT group by id_czas;

    create table PACJENT_FACT_tmp (
    id bigint,
    data_zachorowania varchar(255),
    data_zmiany_stanu varchar(255),
    kraj varchar(255),
    id_pacjenta bigint,
    id_stanu char
    );
```

```
create table PACJENT_FACT_tmp2 (
id bigint,
data_zachorowania varchar(255),
data_zmiany_stanu varchar(255),
kraj varchar(255),
id_pacjenta bigint,
id_stanu char
);

declare @CounterDate as varchar(255) = (1+ (select top 1 RowCzas from tmpCzas3 where czas = @DateFirst));

while @CounterDate < @AllDateAmount
begin
    declare @Date as varchar(255) = (select czas from tmpCzas3 where RowCzas = @CounterDate);
    declare @CounterCountry as int = 1;
    while @CounterCountry < @CountryAmount
    begin
        --potwierdzone przypadki
        declare @Country as varchar(255) = (select kraj from tmpKraj3 where RowKraj = @CounterCountry) ;
        declare @DayPatientAmount as int = (select top 1 nowe_przypadki
        from STATYSTYKI_FACT where kraj = @Country and id_czas = @Date);
        declare @CounterPatient as int = 0;
        while @CounterPatient < @DayPatientAmount
        begin
            insert into PACJENT_DIM (plec, wiek)
            values (CAST(RAND(CHECKSUM(NEWID()))*2 as int), CAST(RAND(CHECKSUM(NEWID()))*90 as int));
            insert into PACJENT_FACT (data_zachorowania, kraj, id_pacjenta, id_stanu)
            values (@Date, @Country, (select top 1 id from PACJENT_DIM order by id desc), 'p');
            --insert into PACJENT (id_czas, kraj, plec, wiek, stan)
            --values (@Date, @Country, CAST(RAND(CHECKSUM(NEWID()))*2 as int), CAST(RAND(CHECKSUM(NEWID()))*90 as int) + 10, 'p');
            set @CounterPatient = @CounterPatient + 1;
        end

        --zgony
        set @CounterPatient = 0;
        declare @DayDeathsAmount as int = (select top 1 nowe_zgony
        from STATYSTYKI_FACT where kraj = @Country and id_czas = @Date);
        while @CounterPatient < @DayDeathsAmount
        begin
            --update PACJENT_FACT set data_zmiany_stanu = @Date, id_stanu = 'z' where id = (select top 1 min(id) from PACJENT_FACT
            insert into PACJENT_FACT_tmp (data_zmiany_stanu, kraj, id_stanu)
            values (@Date, @Country, 'z');
            set @CounterPatient = @CounterPatient + 1;
        end
```

```
        --wyzdrowiali
        set @CounterPatient = 0;
        declare @DayRecoverAmount as int = (select top 1 nowe_wzydrowienia from STATYSTYKI_FACT where kraj = @Country
        and id_czas = @Date);
        while @CounterPatient < @DayRecoverAmount
        begin
            insert into PACJENT_FACT_tmp (data_zmiany_stanu, kraj, id_stanu)
            values (@Date, @Country, 'w');
            set @CounterPatient = @CounterPatient + 1;
        end
        ----------------------------------------------------------------
        create table tmp4 (rownum bigint, id bigint, data_zachorowania varchar(255), kraj varchar(255), id_pacjenta bigint);
        insert into tmp4(rownum, id, data_zachorowania, kraj, id_pacjenta) select top
        (select count(*) from PACJENT_FACT_tmp where @Country = PACJENT_FACT_tmp.kraj and
        id_stanu <> 'p' and PACJENT_FACT_tmp.data_zmiany_stanu = @Date) ROW_NUMBER() over
        (order by data_zachorowania) as rownum, id, data_zachorowania, kraj, id_pacjenta
        from PACJENT_FACT where PACJENT_FACT.kraj = @Country and id_stanu = 'p' order by PACJENT_FACT.data_zachorowania;
        --select * from tmp4;
        create table tmp5 (rownum bigint, data_zmiany_stanu varchar(255), id_stanu char);
        insert into tmp5 (rownum, data_zmiany_stanu, id_stanu) select ROW_NUMBER() over
        (order by data_zmiany_stanu) as rownum, data_zmiany_stanu, id_stanu
        from PACJENT_FACT_tmp where @Country = PACJENT_FACT_tmp.kraj and id_stanu <> 'p'
        and PACJENT_FACT_tmp.data_zmiany_stanu = @Date order by PACJENT_FACT_tmp.data_zmiany_stanu;
        --select * from tmp5;

        create table tmp6 (id bigint, data_zachorowania varchar(255), kraj varchar(255),
        id_pacjenta bigint, data_zmiany_stanu varchar(255), id_stanu char);
        insert into tmp6 (id, data_zachorowania, kraj, id_pacjenta, data_zmiany_stanu, id_stanu)
        select tmp4.id, tmp4.data_zachorowania, tmp4.kraj, tmp4.id_pacjenta, tmp5.data_zmiany_stanu,
        tmp5.id_stanu from tmp4 inner join tmp5 on tmp4.rownum = tmp5.rownum;
        --select * from tmp6;
        --update PACJENT_FACT_tmp
        --set PACJENT_FACT_tmp.data_zmiany_stanu = tmp6.data_zmiany_stanu, PACJENT_FACT_tmp.id_stanu = tmp6.id_stanu where tm
        --insert into PACJENT_FACT_tmp2 (id, data_zachorowania, kraj, id_pacjenta, data_zmiany_stanu, id_stanu)
        --select tmp4.id, tmp4.data_zachorowania, tmp4.kraj, tmp4.id_pacjenta, tmp5.data_zmiany_stanu, tmp5.id_stanu from tmp
        update p
        set p.data_zmiany_stanu = t.data_zmiany_stanu, p.id_stanu = t.id_stanu from PACJENT_FACT as p
        inner join tmp6 as t on t.id = p.id;


        drop table tmp4;
        drop table tmp5;
        drop table tmp6;
        set @CounterCountry = @CounterCountry + 1;
    end
    set @CounterDate = @CounterDate + 1;
end


update PACJENT_DIM set plec = case when (PACJENT_DIM.plec = '1' or PACJENT_DIM.plec = 'k') then 'k' else 'm' end;

drop table tmpCzas3;
drop table tmpKraj3;
drop table PACJENT_FACT_tmp;
drop table PACJENT_FACT_tmp2;
end
```

5. Model kostki



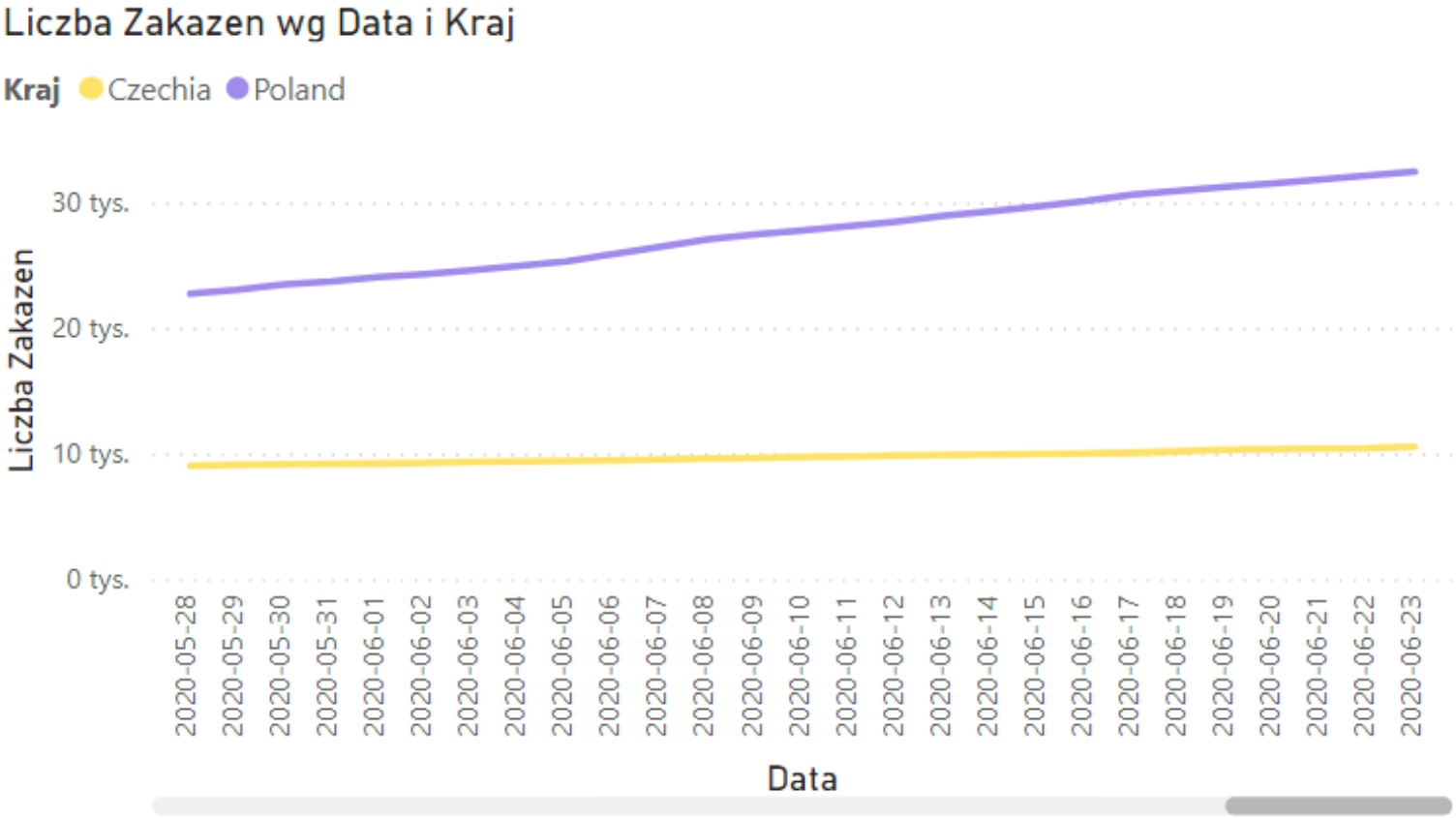
Miary:

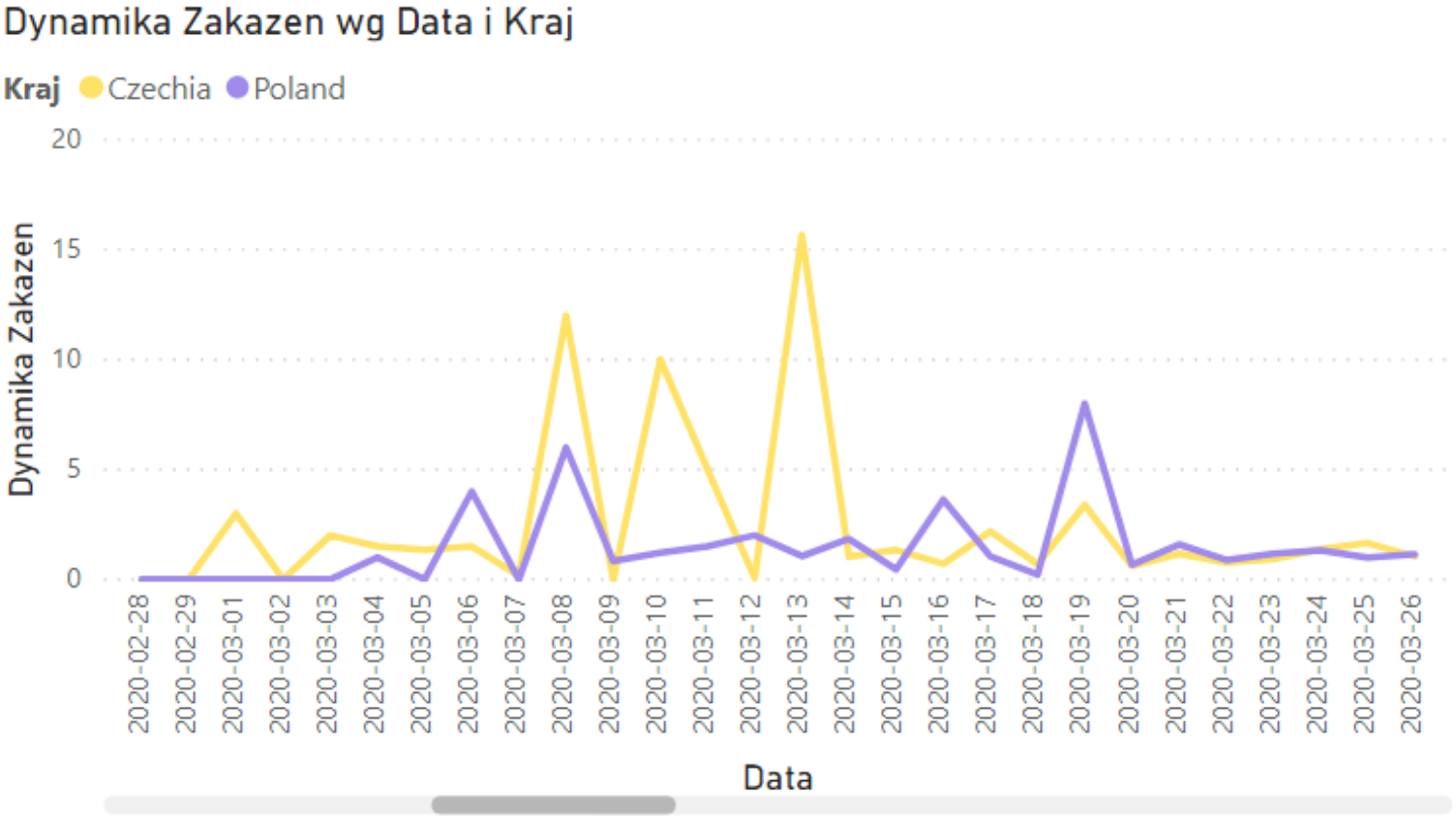
- STATYSTKI_FACT
- PACJENT_FACT

Wymiary:

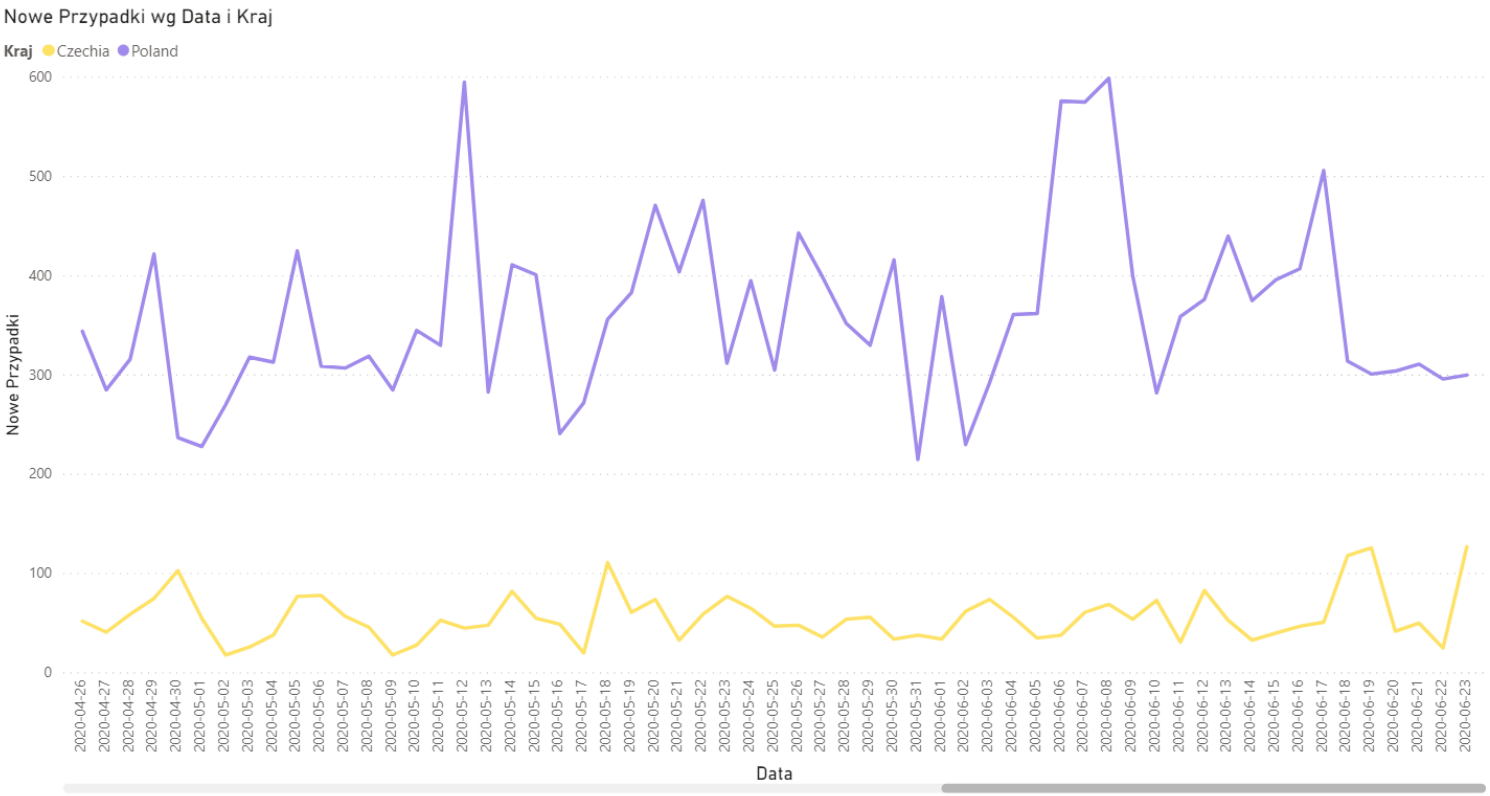
- GEOGRAFIA_DIM
- STAN_DIM
- PACJENT_DIM
- CZAS_DIM

6. Przykładowe raporty w PowerBI

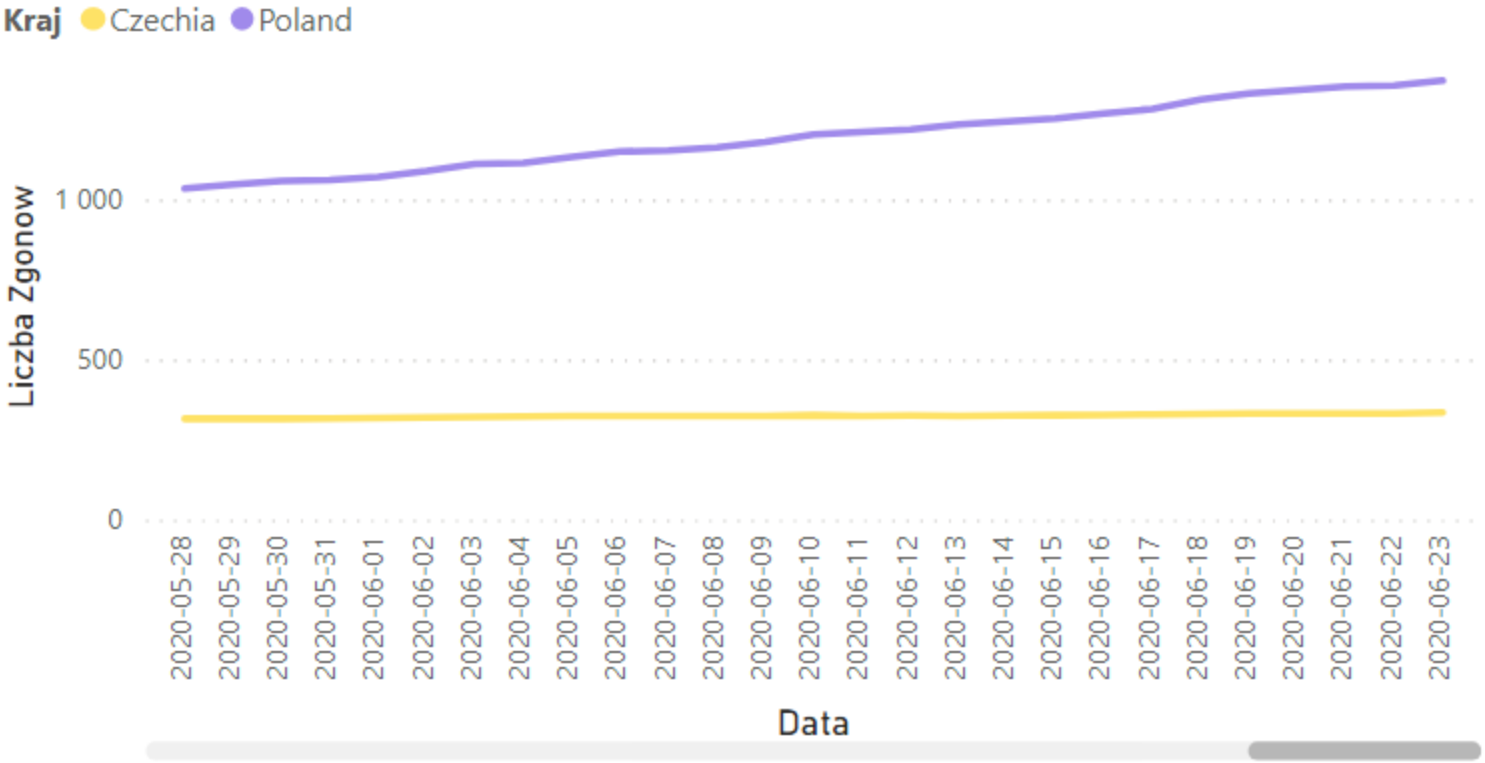




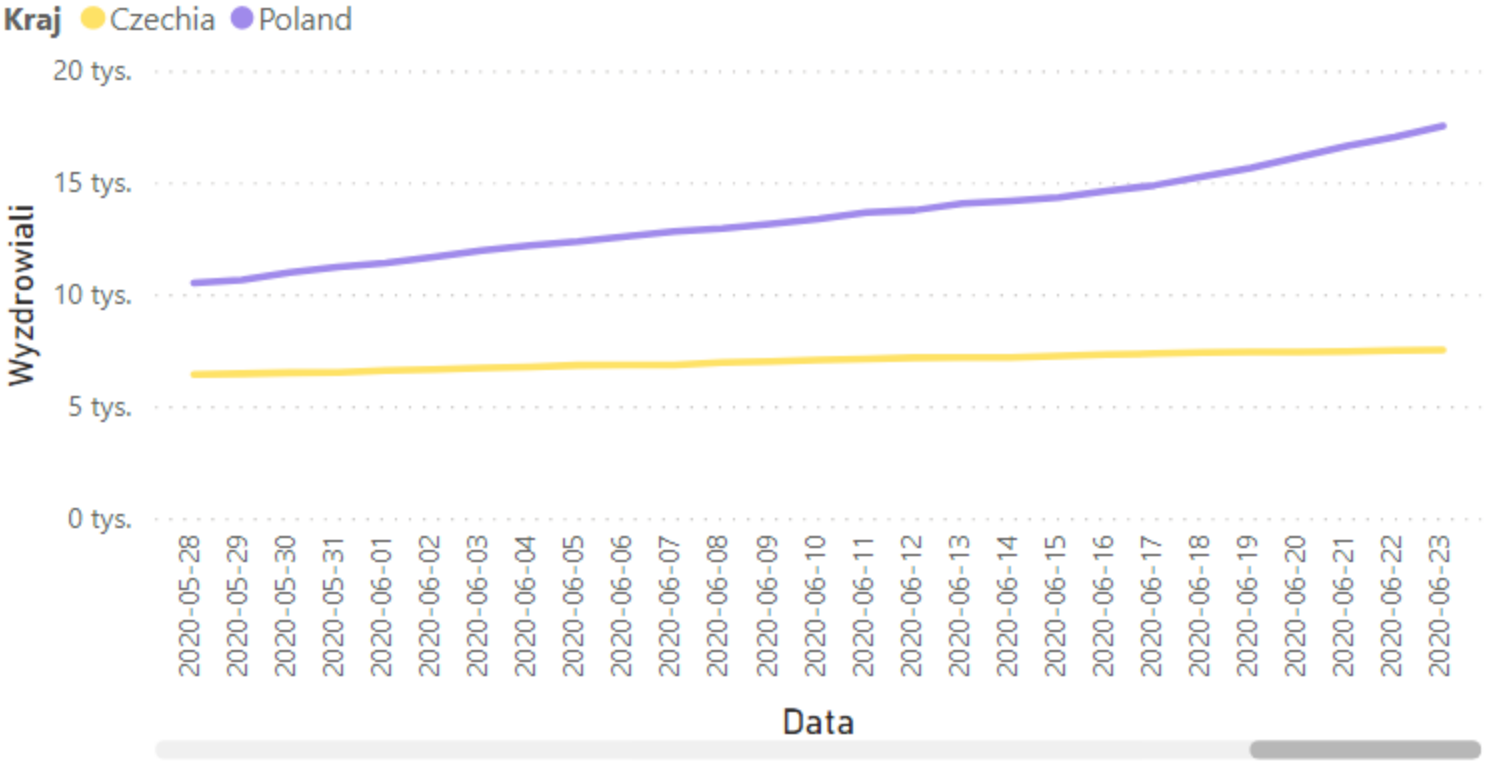
7.


Nowe Przypadki wg Data i Kraj
Kraj
Czechia
Poland
Nowe Przypadki
600
500
400
300
200
100
0
2020-04-26
2020-04-27
2020-04-28
2020-04-29
2020-04-30
2020-05-01
2020-05-02
2020-05-03
2020-05-04
2020-05-05
2020-05-06
2020-05-07
2020-05-08
2020-05-09
2020-05-10
2020-05-11
2020-05-12
2020-05-13
2020-05-14
2020-05-15
2020-05-16
2020-05-17
2020-05-18
2020-05-19
2020-05-20
2020-05-21
2020-05-22
2020-05-23
2020-05-24
2020-05-25
2020-05-26
2020-05-27
2020-05-28
2020-05-29
2020-05-30
2020-05-31
2020-06-01
2020-06-02
2020-06-03
2020-06-04
2020-06-05
2020-06-06
2020-06-07
2020-06-08
2020-06-09
2020-06-10
2020-06-11
2020-06-12
2020-06-13
2020-06-14
2020-06-15
2020-06-16
2020-06-17
2020-06-18
2020-06-19
2020-06-20
2020-06-21
2020-06-22
2020-06-23
Data

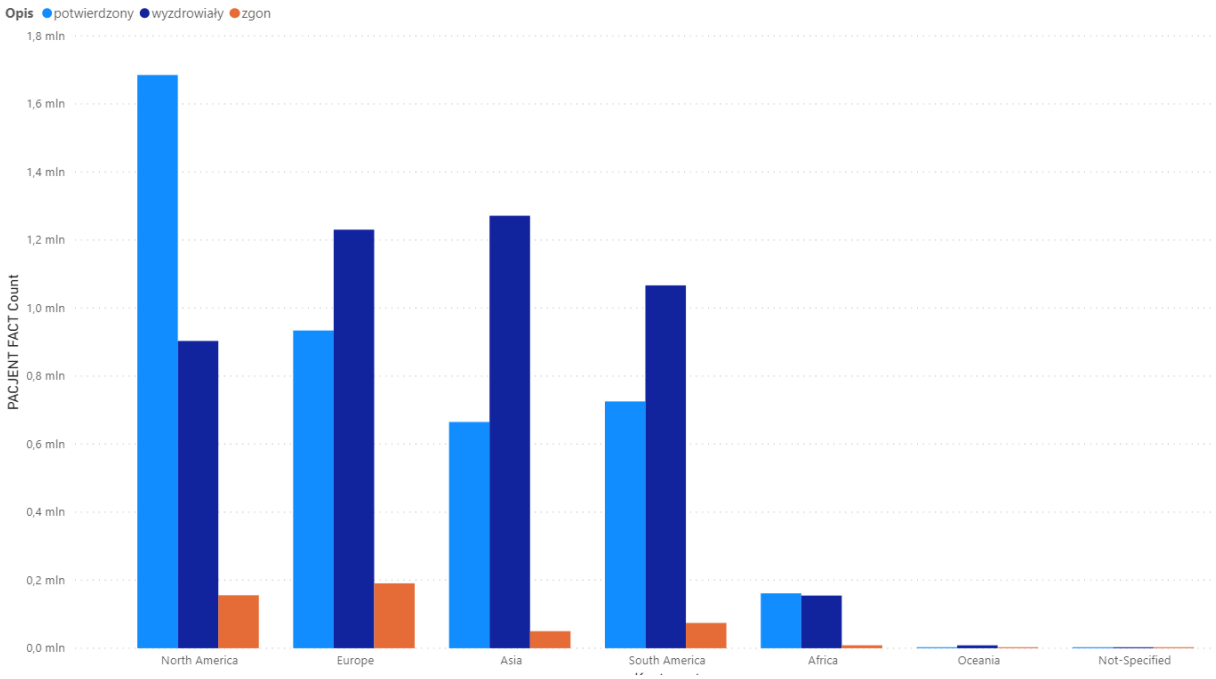
Liczba Zgonow wg Data i Kraj



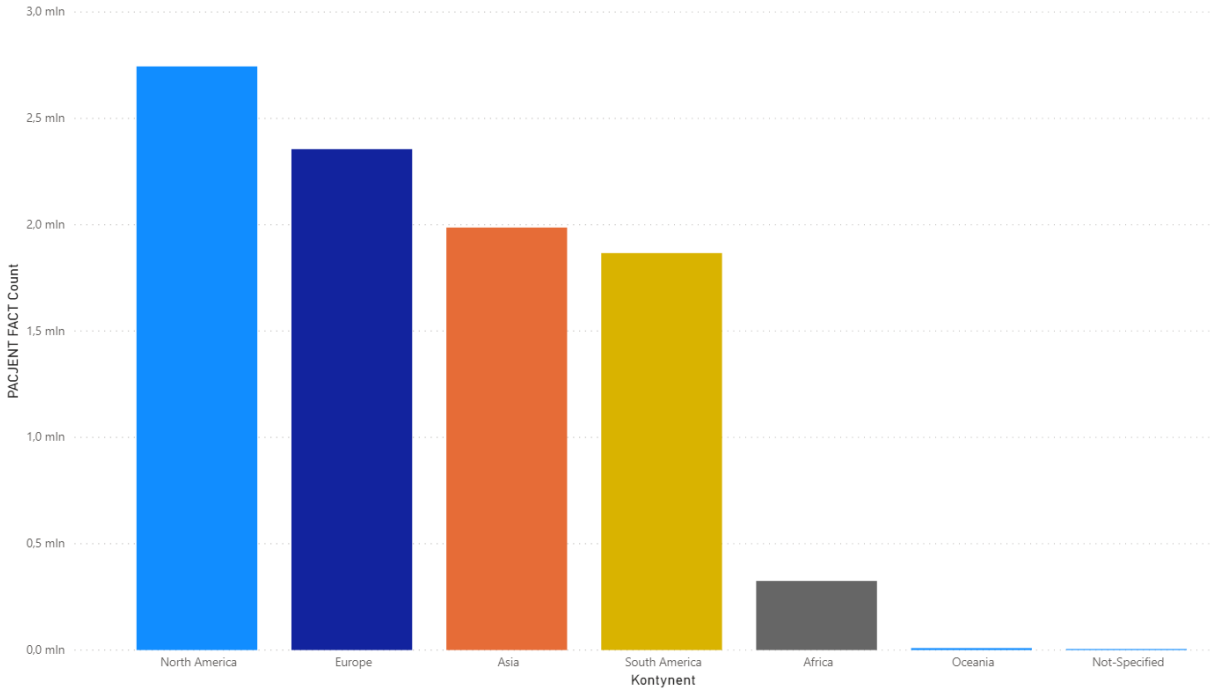
Wyzdrowiali wg Data i Kraj

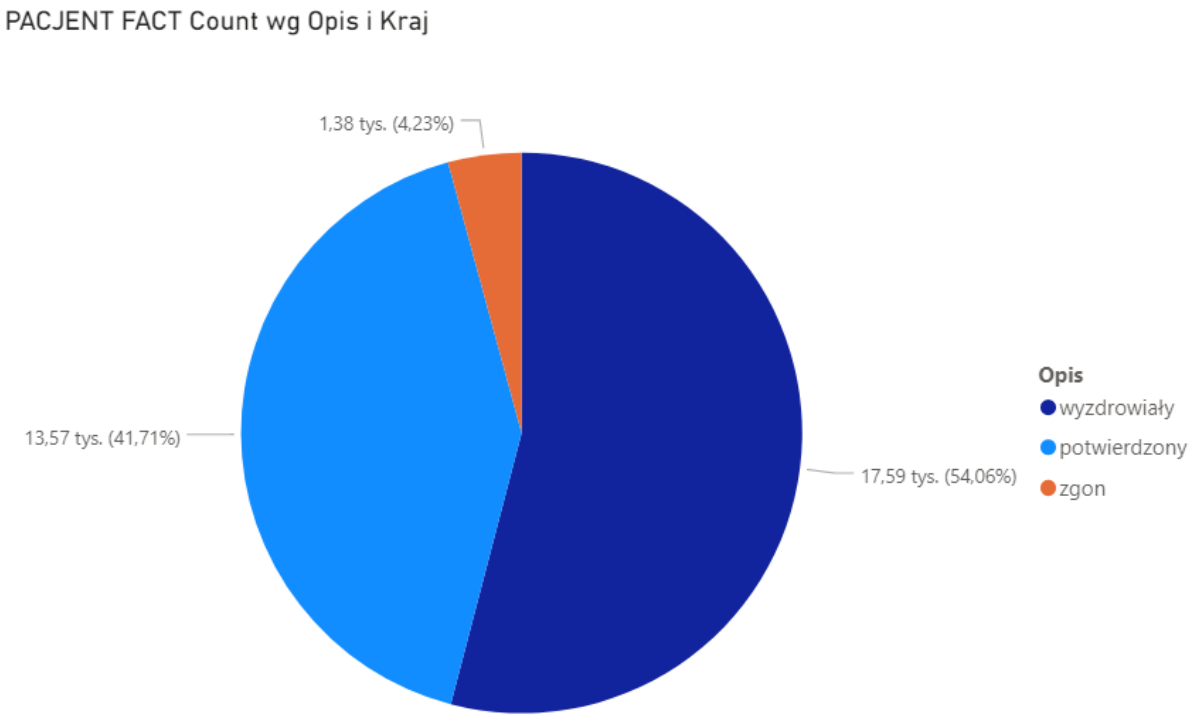


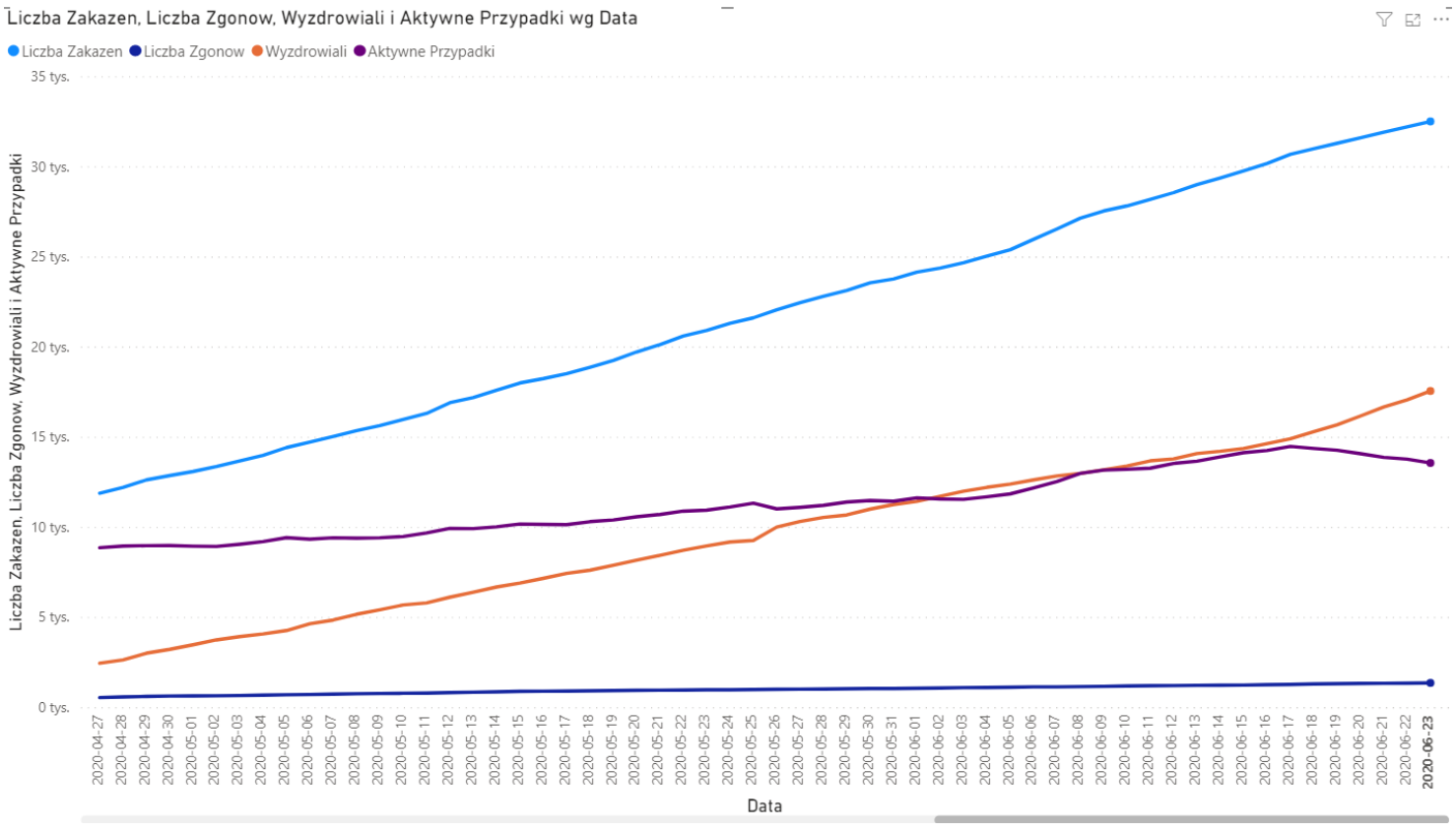
Ilość pacjentów wg kontynentu (z podziałem na ich stan)



PACJENT FACT Count wg Kontynent




PACJENT FACT Count wg Opis i Kraj
1,38 tys. (4,23%)
13,57 tys. (41,71%)
17,59 tys. (54,06%)
Opis
wyzdrowiały
potwierdzony
zgon


Liczba Zakazen, Liczba Zgonow, Wyzdrowiali i Aktywne Przypadki wg Data
Liczba Zakazen
Liczba Zgonow
Wyzdrowiali
Aktywne Przypadki
35 tys.
30 tys.
25 tys.
20 tys.
15 tys.
10 tys.
5 tys.
0 tys.
Liczba Zakazen, Liczba Zgonow, Wyzdrowiali i Aktywne Przypadki
2020-04-27
2020-04-28
2020-04-29
2020-04-30
2020-05-01
2020-05-02
2020-05-03
2020-05-04
2020-05-05
2020-05-06
2020-05-07
2020-05-08
2020-05-09
2020-05-10
2020-05-11
2020-05-12
2020-05-13
2020-05-14
2020-05-15
2020-05-16
2020-05-17
2020-05-18
2020-05-19
2020-05-20
2020-05-21
2020-05-22
2020-05-23
2020-05-24
2020-05-25
2020-05-26
2020-05-27
2020-05-28
2020-05-29
2020-05-30
2020-05-31
2020-06-01
2020-06-02
2020-06-03
2020-06-04
2020-06-05
2020-06-06
2020-06-07
2020-06-08
2020-06-09
2020-06-10
2020-06-11
2020-06-12
2020-06-13
2020-06-14
2020-06-15
2020-06-16
2020-06-17
2020-06-18
2020-06-19
2020-06-20
2020-06-21
2020-06-22
2020-06-23
Data

Liczba Zakazen wg Kraj