

Przepisz i wykonaj poszczególne bloki. Sprawdź rezultaty wykonania. Część bloków działa w oparciu o bazę biblioteka (skrypty: biblioteka\_schemat.sql oraz biblioteka\_dane.sql)

```
declare @x int, @s varchar(10)
```

```
set @x=10
```

```
set @s='napis'
```

```
print @x+2
```

```
print @s
```

---

```
declare @imieP varchar(20), @nazwiskoP varchar(20)
```

```
select @imieP=imie, @nazwiskoP=nazwisko from biblioteka..pracownicy where id=7
```

```
print @imieP+' '+@nazwiskoP
```

---

```
--- z którego wiersza zostaną przypisane dane? ---
```

```
declare @imieP varchar(20), @nazwiskoP varchar(20)
```

```
select @imieP=imie, @nazwiskoP=nazwisko from biblioteka..pracownicy
```

```
print @imieP+' '+@nazwiskoP
```

---

```
---- co zostanie zwrócone? ----
```

```
---- 1.
```

```
declare @imieP varchar(20), @nazwiskoP varchar(20)
```

```
set @imieP='Teofil'
```

```
set @nazwiskoP='Szczerbaty'
```

```
select @imieP=imie, @nazwiskoP=nazwisko from biblioteka..pracownicy where id=1
```

```
print @imieP+' '+@nazwiskoP
```

```
---- 2.
```

```
declare @imieP varchar(20), @nazwiskoP varchar(20)
```

```
set @imieP='Teofil'
```

```
set @nazwiskoP='Szczerbaty'
```

```
select @imieP=imie, @nazwiskoP=nazwisko from biblioteka..pracownicy where id=20
```

```
print @imieP+' '+@nazwiskoP
```

---

```
-- WAITFOR
```

```
create table biblioteka..liczby ( licz1 int, czas datetime default getdate());
```

```
go
```

```
declare @x int
```

```
set @x=2
```

```
insert into biblioteka..liczby(licz1) values( @x );
```

```
waitfor delay '00:00:10'
```

```
insert into biblioteka..liczby(licz1) values( @x+2 );
```

```
select * from biblioteka..liczby;
```

---

```
-- IF..ELSE
```

```
if EXISTS( select * from biblioteka..wypozyczenia) print('Były wypożyczenia')
```

```
else print('Nie było żadnych wypożyczeń')
```

```

-- WHILE

declare @y int
set @y=0
while ( @y<10 )
begin
    print @y
    if ( @y=5 ) break
    set @y=@y+1
end

-- CASE

select tytuł as tytułK, cena as cenaK, [cena jest]=CASE
    when cena<20.00 then 'Niska'
    when cena between 20.00 and 40.00 then 'Przystępna'
    when cena>40 then 'Wysoka'
    else 'Nieznana'
end
from biblioteka..ksiazki

-- NULLIF
-- przydatne, kiedy trzeba pominąć jakąś wartość w funkcjach agregujących - proszę stworzyć własny
-- przykład

select
count(*) as [Liczba pracowników],
avg( nullif( zarobki, 0 ) ) as [Średnia płaca],
min( nullif( zarobki, 0 ) ) as [Płaca minimalna]
from Test..Pracownicy

-- ISNULL
-- pozwala na nadawanie wartości domyślnych tam, gdzie jest NULL - proszę stworzyć własny przykład

select title, pub_id, isnull( price, ( select MIN(price) from pubs..titles ) ) from pubs..titles

-- Komunikaty o błędzie

raiserror ( 21000, 10, 1 )
print @@error

raiserror ( 21000, 10, 1 ) with seterror
print @@error

raiserror ( 21000, 11, 1 )
print @@error

raiserror ( 'Ala ma kota', 11, 1 )
print @@error

declare @d1 datetime, @d2 datetime
set @d1='20091020'
set @d2='20091025'

select dateadd(hour, 112, @d1)
select dateadd(day, 112, @d1)

select datediff(minute, @d1, @d2)
select datediff(day, @d1, @d2)

select datename(month, @d1)
select datepart(month, @d1)

select cast(day(@d1) as varchar)+'-'+cast(month(@d1) as varchar)+'-'+cast(year(@d1) as varchar)

```

---

```
select col_length('biblioteka..pracownicy', 'imie')

select datalength(2+3.4)

select db_id('master')

select db_name(1)

select user_id()

select user_name()

select host_id()

select host_name()

use biblioteka;

select object_id('Pracownicy')

select object_name(object_id('Pracownicy'))
```

---

```
-- 1. --
if exists(select 1 from master.dbo.sysdatabases where name = 'test_cwiczenia') drop database
test_cwiczenia
go
create database test_cwiczenia
go
use test_cwiczenia
go
create table test_cwiczenia..liczby ( liczba int )
go
declare @i int
set @i=1
while @i<100
begin
    insert test_cwiczenia..liczby values( @i )
    set @i=@i+1
end
go
select * from test_cwiczenia..liczby;

-- 2. --
use test_cwiczenia
go
if exists(select 1 from sys.objects where TYPE = 'P' and name = 'proc_liczby') drop procedure
proc_liczby
go
create procedure proc_liczby @max int = 10
as
begin
    select liczba from test_cwiczenia..liczby
    where liczba<=@max
end
go
exec test_cwiczenia..proc_liczby 3
exec test_cwiczenia..proc_liczby
go
```

```
-- 3 --
use test_cwiczenia
go
if exists(select 1 from sys.objects where TYPE = 'P' and name = 'proc_statystyka') drop procedure
proc_statystyka
go
create procedure proc_statystyka @max int output, @min int output, @aux int output
as
begin
    set @max=( select max(liczba) from test_cwiczenia..liczby )
    set @min=( select min(liczba) from test_cwiczenia..liczby )
    set @aux=10
end
go
declare @max int, @min int, @aux2 int
exec test_cwiczenia..proc_statystyka @max output, @min output, @aux=@aux2 output
select @max 'Max', @min 'Min', @aux2
go
```

---

--- Proszę zmodyfikować przykłady - dostosować do własnych baz!!! -----

```
use test_cwiczenia

-- 1 --

-- drop function fn_srednia

create function fn_srednia(@rodzaj varchar(12)) returns int
begin
    return (select avg(price) from pubs..titles where type=@rodzaj)
end

select dbo.fn_srednia('business')

-- 2 --

--drop function funkcja

create function funkcja(@max int) returns table
return (select * from liczby where liczba<=@max)

select * from funkcja(3)
```