June 5-8, 2023 · Data Platforms · Data Engineering · Big Data

BUDAPEST DATA FORUM

TUTORIAL: Haladó SQL lekérdezések

TUTORIAL

# Haladó SQL lekérdezések

Farkas Máté (MCSA, MCSE, MCT)
linkedin.com/in/farkas-mate

June 5-8, 2023 · Data Platforms · Data Engineering · Big Data

# BUDAPEST DATA FORUM

## TUTORIAL: Haladó SQL lekérdezések

TEMATIKA:

1. rész (90 perc)
   - GROUP BY kiegészítések: ROLLUP, CUBE, GROUPPING SETS
   - Halmaz műveletek: UNION, EXCEPT, INTERSECT
   - Rekurzív CTE
   - Ideiglenes táblák és tábla változók

2. rész (90 perc)
   - PIVOT és UNPIVOT
   - SQL programozás: típusok, változók, vezérlési szerkezetek
   - Dinamikus SQL

Farkas Máté (MCSA, MCSE, MCT)
linkedin.com/in/farkas-mate

June 5-8, 2023 · Data Platforms · Data Engineering · Big Data

# BUDAPEST DATA FORUM

## TUTORIAL: Haladó SQL lekérdezések

A GYAKORLATOKHOZ

Azure Data Studio zip letöltése, kicsomagolása:
https://go.microsoft.com/fwlink/?linkid=2176806
(vagy kereső: “download azure data studio” -> zip)
(ha még nincs telepítve vagy nincs SSMS a gépen)

Azure SQL adatbázishoz kapcsolódás:
Server: mate-test-sqlserver.database.windows.net
Authentication type: SQL Login
User name: bpdata
Password: BudapestData2023
Database: TopBike

Farkas Máté (MCSA, MCSE, MCT)
linkedin.com/in/farkas-mate

June 5-8, 2023 · Data Platforms · Data Engineering · Big Data

# BUDAPEST DATA FORUM

## TUTORIAL: Haladó SQL lekérdezések

A GYAKORLATOKHOZ

Azure Data Studio zip letöltése, kicsomagolása:
https://go.microsoft.com/fwlink/?linkid=2176806
(vagy kereső: “download azure data studio” -> zip)
(ha még nincs telepítve vagy nincs SSMS a gépen)

Azure SQL adatbázishoz kapcsolódás:
Server: mate-test-sqlserver.database.windows.net
Authentication type: SQL Login
User name: bpdata
Password: BudapestData2023
Database: TopBike

Farkas Máté (MCSA, MCSE, MCT)
linkedin.com/in/farkas-mate

# Haladó SQL lekérdezések

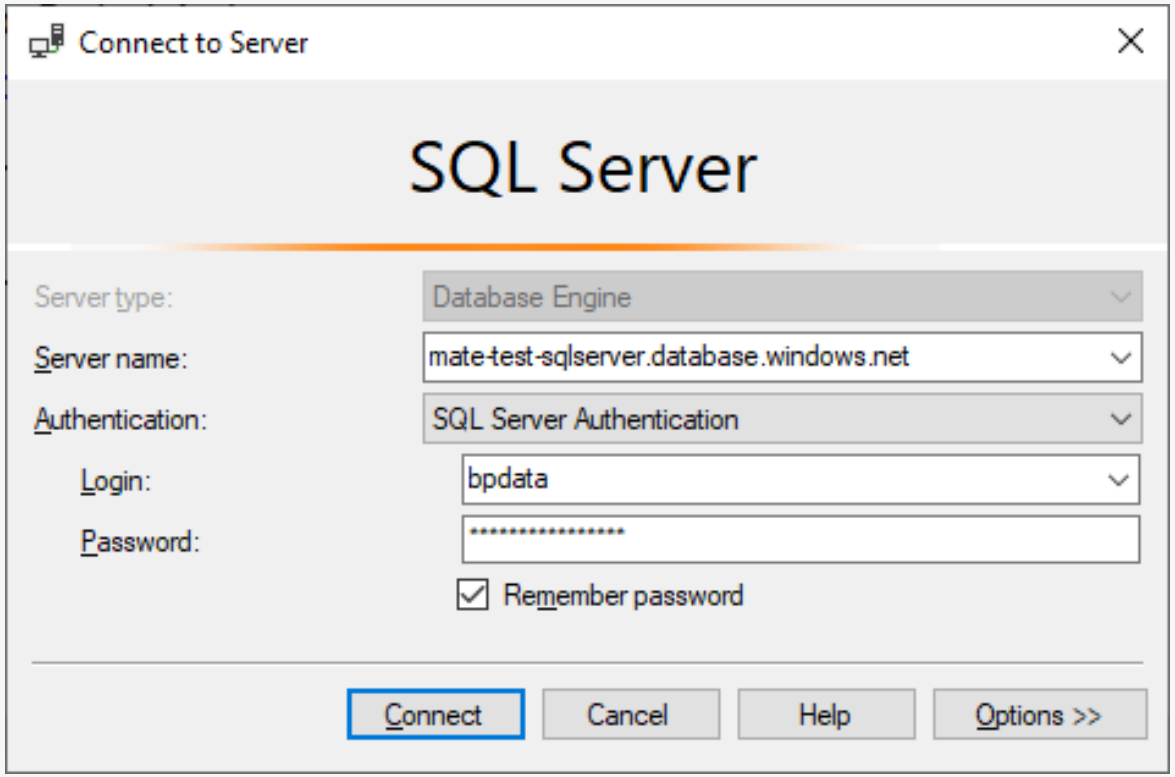
TEMATIKA:

1. rész (90 perc)
   - GROUP BY kiegészítések: ROLLUP, CUBE, GROUPPING SETS
   - Halmaz műveletek: UNION, EXCEPT, INTERSECT
   - Rekurzív CTE
   - Ideiglenes táblák és tábla változók

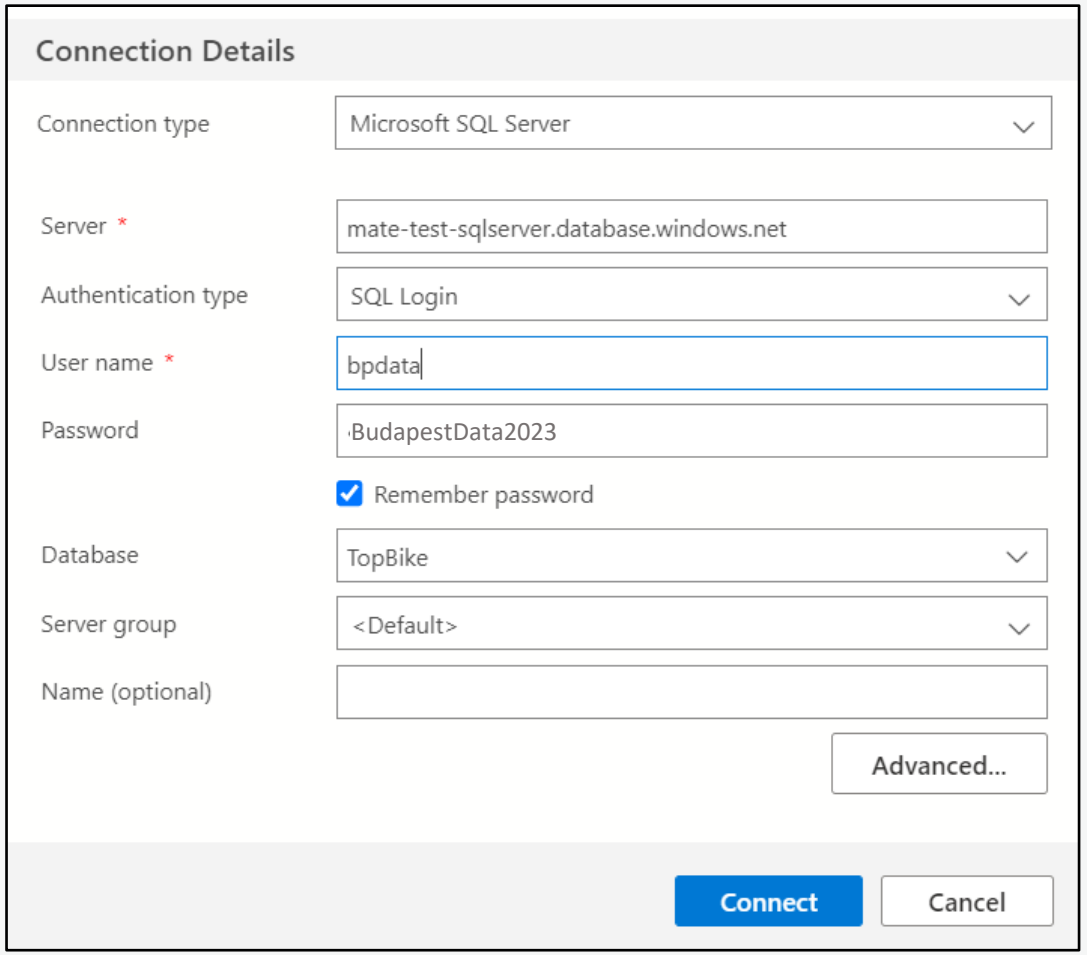
2. rész (90 perc)
   - PIVOT és UNPIVOT
   - SQL programozás: típusok, változók, vezérlési szerkezetek
   - Dinamikus SQL

# Kapcsolódás Azure SQL adatbázishoz

## SQL Server Management Studio (SSMS)




## Azure Data Studio



# GROUP BY kiegészítések: ROLLUP

Old syntax:

```
select
    Year,
    Month,
    count(*) as Orders
from (
    select
        year(OrderDate) as Year,
        month(OrderDate) as Month,
        *
    from Orders
) o
group by Year, Month
with rollup
```

New syntax:

```
select
    Year,
    Month,
    count(*) as Orders
from (
    select
        year(OrderDate) as Year,
        month(OrderDate) as Month,
        *
    from Orders
) o
group by rollup (Year, Month)
```

| Year | Month | Orders |
|---|---|---|
| 2012 | 1 | 336 |
| 2012 | 2 | 219 |
| 2012 | 3 | 304 |
| 2012 | 4 | 269 |
| 2012 | 5 | 293 |
| 2012 | 6 | 390 |
| 2012 | 7 | 385 |
| 2012 | 8 | 285 |
| 2012 | 9 | 352 |
| 2012 | 10 | 321 |
| 2012 | 11 | 383 |
| 2012 | 12 | 378 |
| 2012 | NULL | 3915 |
| 2013 | 1 | 400 |
| 2013 | 2 | 325 |
| 2013 | 3 | 441 |
| 2013 | 4 | 428 |
| 2013 | 5 | 428 |
| 2013 | 6 | 719 |
| 2013 | 7 | 1740 |
| 2013 | 8 | 1789 |
| 2013 | 9 | 1791 |
| 2013 | 10 | 1968 |
| 2013 | 11 | 2103 |
| 2013 | 12 | 2050 |
| 2013 | NULL | 14182 |
| 2014 | 1 | 2141 |
| 2014 | 2 | 1756 |
| 2014 | 3 | 2399 |
| 2014 | 4 | 2115 |
| 2014 | 5 | 2411 |
| 2014 | 6 | 939 |
| 2014 | NULL | 11761 |
| NULL | NULL | 31465 |

# GROUP BY kiegészítések: CUBE

Old syntax:

```
select
    Year,
    Month,
    count(*) as Orders
from (
    select
        year(OrderDate) as Year,
        month(OrderDate) as Month,
        *
    from Orders
) o
group by Year, Month
with cube
```

New syntax:

```
select
    Year,
    Month,
    count(*) as Orders
from (
    select
        year(OrderDate) as Year,
        month(OrderDate) as Month,
        *
    from Orders
) o
group by cube (Year, Month)
```

| Year | Month | Orders |
|---|---|---|
| 2011 | 12 | 228 |
| 2012 | 12 | 378 |
| 2013 | 12 | 2050 |
| NULL | 12 | 2656 |
| NULL | NULL | 31465 |
| 2011 | NULL | 1607 |
| 2012 | NULL | 3915 |
| 2013 | NULL | 14182 |
| 2014 | NULL | 11761 |

# GROUP BY kiegészítések: GROUPING SETS

New syntax:

```
select
    Year,
    Month,
    count(*) as Orders
from (
    select
        year(OrderDate) as Year,
        month(OrderDate) as Month,
        *
    from Orders
) o
group by grouping sets ((Year, Month), ())
```

| Year | Month | Orders |
|---|---|---|
| 2013 | 11 | 2103 |
| 2013 | 12 | 2050 |
| 2014 | 1 | 2141 |
| 2014 | 2 | 1756 |
| 2014 | 3 | 2399 |
| 2014 | 4 | 2115 |
| 2014 | 5 | 2411 |
| 2014 | 6 | 939 |
| NULL | NULL | 31465 |

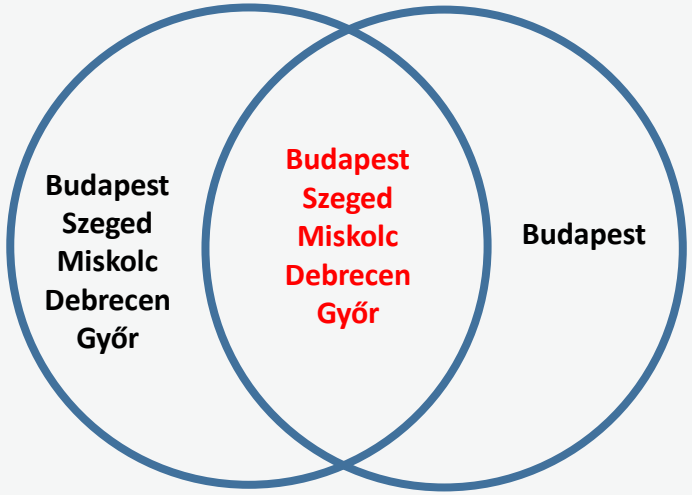
# Halmaz műveletek: UNION

```
select *
from (
VALUES(1, N'Budapest'),
      (2, N'Szeged'),
      (3, N'Miskolc'),
      (4, N'Debrecen'),
      (5, N'Győr')
) varos(ID, Nev)

union

select 1, 'Budapest'
```

**UNION**
(unió)




Az UNION egyeben egy DISTINCT is!

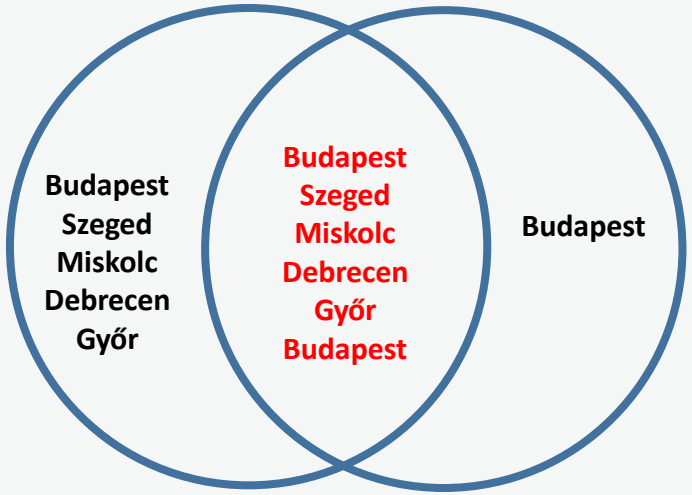
# Halmaz műveletek: UNION ALL

```
select *
from (
VALUES(1, N'Budapest'),
      (2, N'Szeged'),
      (3, N'Miskolc'),
      (4, N'Debrecen'),
      (5, N'Győr')
) varos(ID, Nev)

union all

select 1, 'Budapest'
```

**UNION ALL**
(teljes)



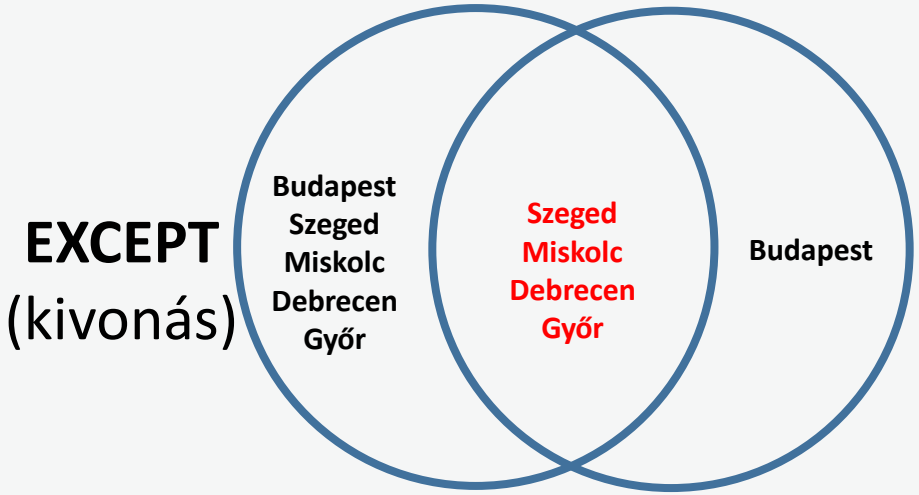

Az UNION ALL valójában összefűzés!

# Halmaz műveletek: EXCEPT

```
select *
from (
VALUES(1, N'Budapest'),
      (2, N'Szeged'),
      (3, N'Miskolc'),
      (4, N'Debrecen'),
      (5, N'Győr')
) varos(ID, Nev)

except

select 1, 'Budapest'
```




Az EXCEPT kiváltható NOT EXISTS-el

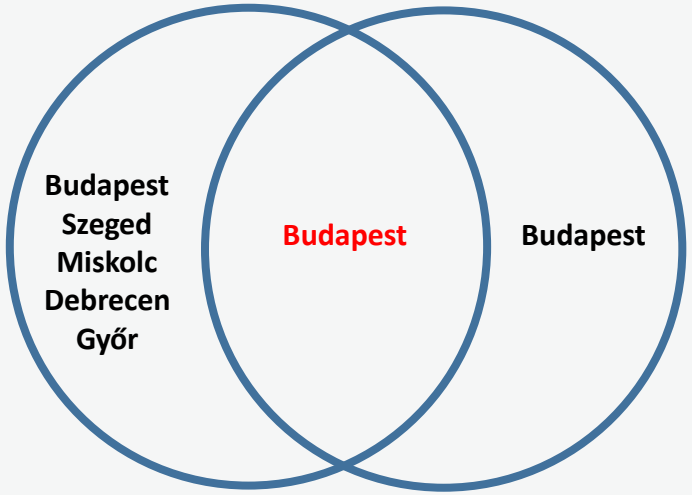
# Halmaz műveletek: INTERSECT

```
select *
from (
VALUES(1, N'Budapest'),
      (2, N'Szeged'),
      (3, N'Miskolc'),
      (4, N'Debrecen'),
      (5, N'Győr')
) varos(ID, Nev)

intersect

select 1, 'Budapest'
```

**INTERSECT**
(metszet)



# Rekurzív CTE

Önmagát lekérdező lekérdezés (hasonlóan a rekurzív függvényekhez)!
Mindíg tartalmaz egy UNION ALL –t
Kell bele egy WHERE feltétel, ami megállítja
MAXRECURSION-el lehet megnövelni a mélységet (alapból 100)
CTE:

```
with list as (
    select 1 as ID
    union all
    select ID + 1 from list
    where ID < 100
)
select ID from list
```

Listák előállításához vagy hierarchiák bejárásához.

# Ideiglenes táblák (#)

Implicit

```
drop table if exists #ord

select *
into #ord
from Orders
where SalesPersonID is not null
```

Explicit

```
drop table if exists #ord

create table #ord(
    OrderID int,
    OrderDate date,
    CustomerID int,
    SalesPersonID int,
    SubTotal decimal(20, 4)
)

insert into #ord
select
    OrderID,
    OrderDate,
    CustomerID,
    SalesPersonID,
    SubTotal
from Orders
where SalesPersonID is not null
```

# Ideiglenes táblák (#)

- CREATE TABLE-el bármi definiálható, ami egy normal táblán (megszorítások, indexek, utólagos szerkezet módosítások ALTER TABLE-el)
- SELECT INTO –val ugyanazok a megkötések mint a view-oknál és a subquery-nél (egyedi, kötelező oszlopnevek)
- Egyedi név az adatkapcsolaton belül
- Fizikailag külön adatbázisban van tárolva (tempdb)
- A kapcsolat bontásával a szerver automatikusan megszünteti vagy DROP TABLE
- Létezik lokális (#) és globális (##)
- A kód egyszerűsítésére és/vagy teljesítmény optimalizálásra is használható
- Túl sok temp tábla teljesítmény romláshoz vezethet

# Tábla változók (@)

```
declare @ord table (
    OrderID int,
    OrderDate date,
    CustomerID int,
    SalesPersonID int,
    SubTotal decimal(20, 4)
)

insert into @ord
select
    OrderID,
    OrderDate,
    CustomerID,
    SalesPersonID,
    SubTotal
from Orders
where SalesPersonID is not null
```

# Tábla változók (@)

- Csak DECLARE –el lehet létrehozni és nem lehet megszüntetni
- Csak a CREATE TABLE szintaxisban használható elemekkel lehet létrehozni
- Nem lehet utólag a szerkezeten módosítani
- A tempdb-ben van tárolva fizikailag!
- A szerver nem számol rá statisztikát (úgy számol, hogy mindíg 1 sor van benne)
- Átadható eljárás vagy függvény paramétereként
- Definiálható egyedi típusként
- Csak lokális lehet
- Amíg a kód fut addig él, utána megszűnik
- A batch-ek futása között megszűnik (a következő batch-ben nem elérhető)

# Temp tábla vs. tábla változó

| | Ideiglenes tábla (temporary table) # | Tábla változó @ |
|---|---|---|
| **Létrehozás** | SELECT INTO vagy CREATE TABLE | DECLARE |
| **Utólagos módosítás** | ALTER TABLE | - |
| **Megszüntetés** | DROP TABLE vagy a kapcsolat bontása | - |
| **Érvényesség** | Lokális (#) vagy globális (##) | Csak lokális |
| **Élettartam** | A kapcsolat élettartamáig vagy DROP TABLE | Csak a batch futásáig |
| **Alkalmazható** | Függvényben és paraméterként nem | Használható függvényben is, paraméterként is |
| **Adatmennyiség** | Bármennyi sor és oszlop | Csak kevés sor és bármennyi oszlop |
| **Scope** | A létrehozó batch és az alatti batch-ek | Csak a létrehozó batch |
| **Fizikai tárolás** | Igen | Igen |
| **Jogosultság** | Read-only jogosultsággal is | Bármilyen |

# Kereszttáblás lekérdezés: PIVOT

```
select
    SalesPersonID,
    sum(case when year(OrderDate) = 2011 then SubTotal end) as [2011],
    sum(case when year(OrderDate) = 2012 then SubTotal end) as [2012],
    sum(case when year(OrderDate) = 2013 then SubTotal end) as [2013],
    sum(case when year(OrderDate) = 2014 then SubTotal end) as [2014],
    sum(case when year(OrderDate) = 2015 then SubTotal end) as [2015],
    sum(case when year(OrderDate) = 2016 then SubTotal end) as [2016],
    sum(SubTotal) as TotalSales
from Orders
group by SalesPersonID
```

| SalesPersonID | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | TotalSales |
|---|---|---|---|---|---|---|---|
| 284 | NULL | 441639,5961 | 1269908,9235 | 600997,1704 | NULL | NULL | 2312545,69 |
| 278 | 500091,8202 | 1283569,6294 | 1389836,8101 | 435948,9551 | NULL | NULL | 3609447,2148 |
| 281 | 967597,2899 | 2294210,5506 | 2387256,0616 | 777941,6519 | NULL | NULL | 6427005,554 |
| 275 | 875823,8318 | 3375456,8947 | 3985374,8995 | 1057247,3786 | NULL | NULL | 9293903,0046 |
| 276 | 1149715,3253 | 3834908,674 | 4111294,9056 | 1271088,5216 | NULL | NULL | 10367007,4265 |
| 287 | NULL | 116029,652 | 560091,7843 | 56637,7478 | NULL | NULL | 732759,1841 |
| 279 | 1521289,1881 | 2674436,3518 | 2188082,7813 | 787204,4289 | NULL | NULL | 7171012,7501 |

# Kereszt táblás lekérdezés: PIVOT

```
select
    SalesPersonID,
    [2011], [2012], [2013], [2014], [2015], [2016]
from (
    select SalesPersonID, year(OrderDate) Year, SubTotal
    from Orders
) o
pivot (sum(SubTotal) for Year in ([2011], [2012], [2013], [2014], [2015], [2016])) p
```

| SalesPersonID | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | TotalSales |
|---|---|---|---|---|---|---|---|
| 284 | NULL | 441639,5961 | 1269908,9235 | 600997,1704 | NULL | NULL | 2312545,69 |
| 278 | 500091,8202 | 1283569,6294 | 1389836,8101 | 435948,9551 | NULL | NULL | 3609447,2148 |
| 281 | 967597,2899 | 2294210,5506 | 2387256,0616 | 777941,6519 | NULL | NULL | 6427005,554 |
| 275 | 875823,8318 | 3375456,8947 | 3985374,8995 | 1057247,3786 | NULL | NULL | 9293903,0046 |
| 276 | 1149715,3253 | 3834908,674 | 4111294,9056 | 1271088,5216 | NULL | NULL | 10367007,4265 |
| 287 | NULL | 116029,652 | 560091,7843 | 56637,7478 | NULL | NULL | 732759,1841 |
| 279 | 1521289,1881 | 2674436,3518 | 2188082,7813 | 787204,4289 | NULL | NULL | 7171012,7501 |

# Kereszt táblás lekérdezés: PIVOT

```
select
    SalesPersonID,
    [2011], [2012], [2013], [2014], [2015], [2016]
from (
    select SalesPersonID, year(OrderDate) as Year, SubTotal
    from Orders
) o
pivot (sum(SubTotal) for Year in ([2011], [2012], [2013], [2014], [2015], [2016])) p
```

PIVOT 4 jellemzője:

- Mely oszlopok egyedi értékei legyenek a sor fejlécekben (row headers)
- Mely oszlop egyedi értékei képezzék az oszlop fejléceket (column header)
- Mely oszlop értékei legyenek összesítve a táblázatban
- Milyen összesítő függvénnyel

# Normalizálás: UNPIVOT

| SalesPersonID | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | TotalSales |
|---|---|---|---|---|---|---|---|
| 284 | NULL | 441639,5961 | 1269908,9235 | 600997,1704 | NULL | NULL | 2312545,69 |
| 278 | 500091,8202 | 1283569,6294 | 1389836,8101 | 435948,9551 | NULL | NULL | 3609447,2148 |
| 281 | 967597,2899 | 2294210,5506 | 2387256,0616 | 777941,6519 | NULL | NULL | 6427005,554 |
| 275 | 875823,8318 | 3375456,8947 | 3985374,8995 | 1057247,3786 | NULL | NULL | 9293903,0046 |
| 276 | 1149715,3253 | 3834908,674 | 4111294,9056 | 1271088,5216 | NULL | NULL | 10367007,4265 |
| 287 | NULL | 116029,652 | 560091,7843 | 56637,7478 | NULL | NULL | 732759,1841 |
| 279 | 1521289,1881 | 2674436,3518 | 2188082,7813 | 787204,4289 | NULL | NULL | 7171012,7501 |

```
select SalesPersonID, Year, SubTotal
from #tmp
unpivot (SubTotal for Year in ([2011], [2012], [2013], [2014], [2015], [2016])) u
```

| SalesPersonID | Year | SubTotal |
|---|---|---|
| 284 | 2012 | 441639,5961 |
| 284 | 2013 | 1269908,9235 |
| 284 | 2014 | 600997,1704 |
| 278 | 2011 | 500091,8202 |
| 278 | 2012 | 1283569,6294 |
| 278 | 2013 | 1389836,8101 |

# SQL programozási elemek

A programozáshoz szükséges elemek:

- Adattípusok
- Változók és változókkal való műveletek
- Vezérlési utasítások
  - Logikai kiértékelések
  - Kód blokk definiálás
  - Feltételes elágazások (egyirányú, többirányú)
  - Ciklusok (előtesztelő, hátultesztelő)
  - Ciklus léptetés (sorszámozás, elem számozás, léptetés) - nincs
  - Ciklus vezérlés (kilépés, újraindítás)
- Szubrutinok (függvények és eljárások)
- Batch és scope a változók számára

# Változók

- Ideiglenes adattárolásra használható programozási eszköz
- A változó neve @-al kezdődik
- Adattípus megadása kötelező
- A változó neve egyedi kell, hogy legyen a batch-en belül (kis-nagybetű független)
- A változót nem lehet megszűntetni
- A változó a futási ideéig él utána azonnal megszűnik
- Definíció:

```
DECLARE @Szam TINYNT
```

- Tömeges változó definíció:

```
DECLARE @Szam int, @Szoveg nvarchar(100), @Datum datetime
```

```
DECLARE @Szam int,
        @Szoveg nvarchar(100),
        @Datum datetime
```

# Változó értékadása

- A változó alapértelmezett értéke mindíg NULL
- Értékadás:

```
SET @Szam = 632
```

- Tömeges értékadás:

```
SELECT @Szam = 123, @Szoveg = 'ABC', @Datum = '2020-12-06'
```

- Kezdeti értékadás:

```
DECLARE @Szam INT = 123,
        @Datum DATETIME = GETDATE()
```

- Más változóból (a forrás változónak már definiálva kell legyen):

```
SET @Szam2 = @Szam1
```

- SELECT skaláris eredményéből:

```
SET @Szam = (SELECT COUNT(*) FROM Customer)
```

- SELECT eredményéből, egyszerre akár több változónak is:

```
SELECT @ID = CustomerID, @Name = FirstName + ' ' + LastName, @Email = Email
FROM Customer
```

# Adat típusok

Ugyanazok a típusok használatók, mint a táblák definicióinál:

- Egész: BIT, TINYINT, SMALLINT, INT, BIGINT
- Tört (fix): DECIMAL / NUMERIC, MONEY, SMALLMONEY
- Tört (lebegő): FLOAT / REAL
- Rövid szöveg: CHAR / VARCHAR, NCHAR / NVARCHAR
- Hosszú szöveg: VARCHAR(MAX), NVARCHAR(MAX), TEXT, NTEXT, SYSNAME
- Dátum: DATE, TIME, DATETIME, DATETIME2
- Bináris: BINARY / VARBINARY, VARBINARY(MAX), IMAGE
- Egyéb: UNIQUEIDENTIFIER, SQL_VARIANT
- Összetett: JSON, HIERARCHYID, GEOSPATIAL
- Objektum: XML, TABLE, CURSOR

```
select * from sys.types
```

# Konstansok adat típusai

Az SQL kódban használt konstans értékek:

- Számok

```
123      11.4897      -.7985
```

- Szövegek

```
'Szöveg konstans'      'Sortörés
a szövegben'
```

- Dátumok

```
'2020-01-01'      '20200101'
```

- Binárisok

```
0x660D167899
```

# Adat típus váltások (konverziók)

- A meglévő adattípusról egy másikra akarunk váltani
- A változó adattípusa nem változtatható meg.
- Az adattípus váltás nem mindíg sikeres.

- Konverzió: CONVERT / TRY_CONVERT függvénnyel

```
CONVERT(VARCHAR(20), GETDATE(), 102)
```

- Kényszerítés:
  - Implicit: a forrás típusa eltér a cél típusától

```
DECLARE @Datum DATE = GETDATE()
```

  - Explicit: CAST / TRY_CAST függvénnyel

```
PRINT 'Érték: ' + CAST(15 as varchar(10))
```

# SQL programozás: BEGIN – END

- Kód blokk definiálására használható
- Ha több SQL utasítást vagy parancsot együtt kell kezelni
- Bárhol használható (legyenek mindig párban)
- A kód blokkban kötelezően legalább 1 utasításnak lennie kell (ami nem comment)

```
BEGIN

    SELECT ...

END
```

# SQL programozás: BEGIN – END

- Kód blokk definiálására használható
- Ha több SQL utasítást vagy parancsot együtt kell kezelni
- Bárhol használható (legyenek mindig párban)
- A kód blokkban kötelezően legalább 1 utasításnak lennie kell (ami nem comment)

```
BEGIN

    SELECT ...

END
```

# SQL programozás: IF – ELSE

- IF után valamilyen logikai kifejezés kell (amely IGAZ/HAMIS-at ad vissza)
- Az IF után csak egy kifejezés állhat, de az kötelező
- Ha több utasítás van az IF után, akkor BEGIN-END blokk kell
- Nincs THEN, zárójel nem kötelező (BEGIN-END sem kötelező)

```
IF @Var = 1
    UPDATE ...
```

- Az IF folytatható ELSE ággal, de nem kötelező
- Csak egy utasítás vagy kód blokk állhat mögötte (de az kötelező)
- IF – ELSE szerkezet egymásba ágyazható (több irányú elágazásokhoz)

```
IF @Var = 1
    UPDATE ...
ELSE
    UPDATE ...
```

# SQL programozás: WHILE

- Az egyetlen ciklus szerverző utasítás T-SQL-ben
- Csak "előtesztelő" változata van
- Csak egyetlen utasítás szerepelhet utána, ha több sor kell, akkor BEGIN – END közé
- IGAZ/HAMIS kifejezést vizsgál
- Amíg a kifejezés igaz, addig a ciklus fut (végtelen ciklus probléma)
- A feltétel kifejezés minden egyes ciklus előtt újra kivizsgálásra kerül
- A WHILE utáni utasítást vagy a BEGIN – END blokkot "ciklusmagnak" hívjuk

```
WHILE (@Var = 1)
BEGIN
     ...
END
```

# SQL programozás: BREAK – CONTINUE

- Ciklus megszakítása
- A kód a END után folytatódik
- Csak ciklusban használható

```
WHILE (@Var = 1)
BEGIN
    ...
    IF @Var = 1 BREAK
END

SELECT ...
```

- Ciklus újraindítása
- A kód a WHILE –on folytatódik
- Csak ciklusban használható

```
WHILE (@Var = 1)
BEGIN
    ...
    IF @Var = 1 CONTINUE
    ...
END
```

# SQL programozás: GOTO

- A program (script vagy eljárás) egy előre definiált pontjára ugrik
- Az ugrási végpontokat előre kell definiálni ún. cimkékkel (label)
- Elavult módszer a kód vezérlésére (de még ma is használt)

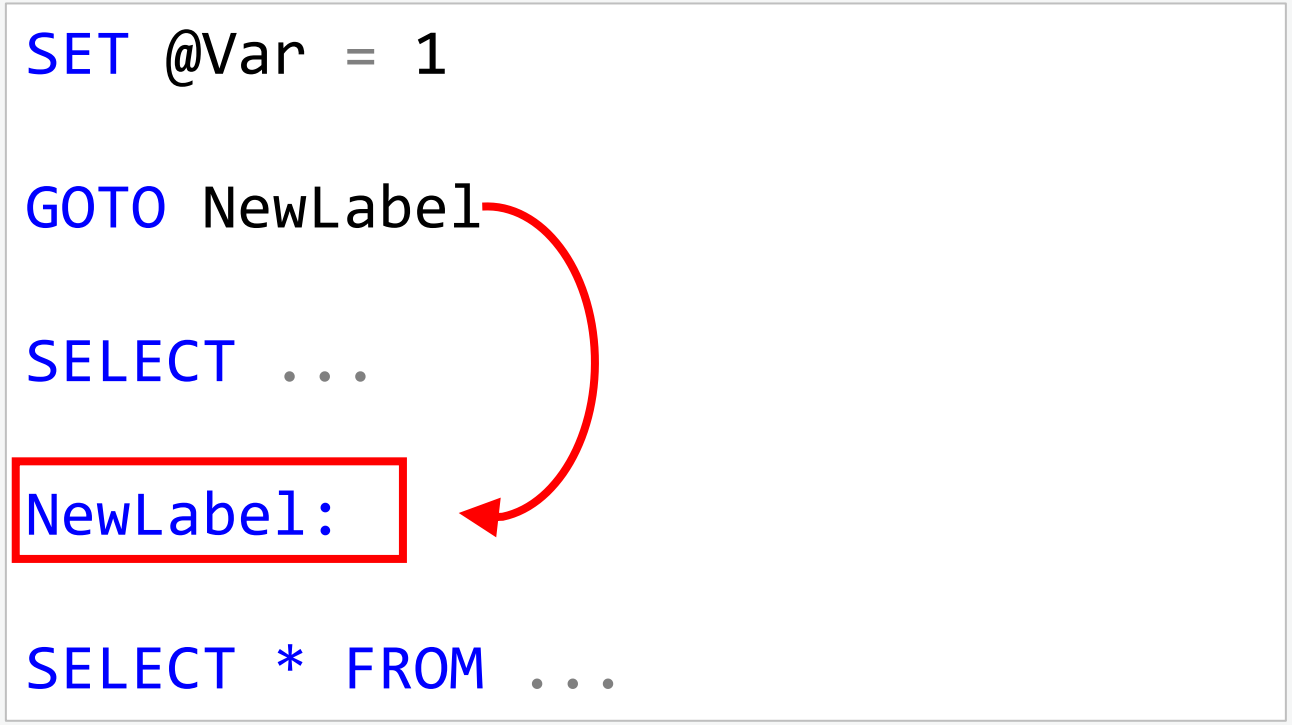
```
SET @Var = 1

GOTO NewLabel

SELECT ...

NewLabel:

SELECT * FROM ...
```



# SQL programozás: RETURN

- A script vagy eljárás azonnali megszakítására használható
- Leginkább tesztelés során használjuk
- Tárolt eljárásokban visszatérési értéket is megadhatunk

```
WHILE (@Var = 1)
BEGIN
    ...
    IF @Var = 1 RETURN
END

SELECT ...
```

# A dinamikus SQL

- Akkor használjuk, ha
  - a lekérdezés írásakor még nem ismert a lekérdezés eredményének végső formája vagy
  - a forrás táblák és oszlopok neve vagy
  - a felhasznált join-ok típusa

- Az SQL lekérdezés a futtatás indításakor még nem ismert
- Az lekérdezés futás közben van összeállítva és letárolva egy változóba
- Az összeállításhoz bármely programvezérlő utasítás és változók is használhatók
- A dinamikus sql lekérdezésnek lehetnek bemeneti paraméterei
- Futtatás EXEC paranccsal (vagy sp_executesql -el)

- Dinamikus SQL-t csak végső esetben használjuk, ha nincs más megoldás!

# A dinamikus SQL

Alap szintaxis

```
declare @SQL nvarchar(max)

set @SQL = 'SELECT OrderID FROM Orders'

exec(@SQL)
```

Külső sql (statikus)

Belső sql (dinamikus)

TIPP: Kódolási konvenció, hogy a “külső” vagy statikus sql-t kisbetűkkel írjuk, de a “belső” vagy dinamikus sql-t pedig NAGYBETŰKKEL. Így jobban elkülönülnek egymástól.

# Dinamikus SQL példa (dinamikus PIVOT)

- A lekérdezés keresse meg az összes eladást és azok vásárlóit és jelenítsen meg belőle egy PIVOT lekérdezést
- A PIVOT paraméterei:
  - Sor fejlécek: vásárló országa (a kétbetűs kód elegendő), növekvőbe rendezett
  - Oszlop fejlécek: a termék színe (csak ahol van, balról jobbra növekvő sorrend)
  - Összesítő függvény: összeg
  - Összesített adat: LineTotal

Az alap lekérdezés

```
select c.Country, p.Color, od.LineTotal
from Orders o
join OrderDetails od on od.OrderID = o.OrderID
join Product p on p.ProductID = od.ProductID
join Customer c on c.CustomerID = o.CustomerID
where p.Color is not null
and c.Country is not null
```

Az eredmény:

| Country | Black | Red | Silver | Yellow |
|---|---|---|---|---|
| DE | 143256,4315 | 321991,9495 | 65814,0432 | 19008,3125 |
| GB | 218550,4638 | 255484,1746 | 78985,41 | 29012,6875 |
| AU | 681809,9519 | 1094629,6269 | 311327,4168 | 37016,1875 |
| CA | 116862,4537 | 432767,3407 | 28842,7464 | 9003,9375 |
| FR | 181310,1031 | 279417,3688 | 40685,5428 | 21009,1875 |
| US | 519155,5457 | 1219877,4304 | 226756,26 | 58025,375 |

# Dinamikus SQL példa (dinamikus PIVOT)

```
declare @RowHeaders nvarchar(max) = 'Country'
declare @ColumnHeader nvarchar(max) = 'Color'
declare @Column nvarchar(100) = 'LineTotal'
declare @Function nvarchar(100) = 'sum'
declare @Query nvarchar(max) = '
select c.Country, p.Color, od.LineTotal
from Orders o
join OrderDetails od on od.OrderID = o.OrderID
join Product p on p.ProductID = od.ProductID
join Customer c on c.CustomerID = o.CustomerID
where p.Color is not null
and c.Country is not null
'
```

TUTORIAL

# Haladó SQL lekérdezések

# Kérdések / Válaszok

Farkas Máté (MCSA, MCSE, MCT)
linkedin.com/in/farkas-mate