## Povel: Insert

**Cíl :** Vložení jednoho / či více řádek do jedné tabulky

```
Insert   Into   Tabulka   |
                 Tabulka  (Sloupec1, Sloupec2,...)
          Values (Hodnota1, hodnota2,...) ;
```

```
Insert   Into   Tabulka | Tabulka(Sloupec1, Sloupec2,...)
                    Values (Hodnota1, hodnota2,...) ,          -- pro 1 radku
                           (Hodnota1, hodnota2,...) ,          -- pro 2 radku
                           (Hodnota1, hodnota2,...) ,          -- pro 3 radku
                           (Hodnota1, hodnota2,...) ;          -- pro 4 radku
```

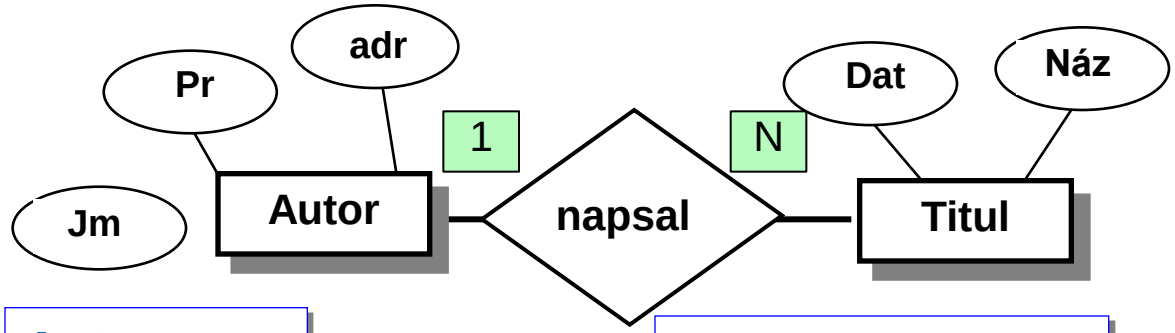
**Oracle - nelze**

Konceptuální model

Entity

Vlastnosti Entit

Vztahy mezi Entitami




Databázový model

**Autor**

PK Pr
PK Jm
Adr

**Titul**

PK FK Autor. Pr
PK FK Autor. Jm
PK Naz
Dat

DB vytvoření

```
Create Table Autor
( Pr   Varchar(20) … ,
  Jm   Varchar(20) … ,
  Adr  Varchar(50) …,

  Primary Key (Pr,Jm)
 ) ;
```

```
Create Table Titul
    ( Pr  Varchar(20) ,
      Jm   Varchar(20)
      Naz  Varchar(30) ,
      Dat   Date,
      Primary Key ( Pr, Jm, Naz ) ,
      Foreign Key ( ……. ) ) ;
```

DB realizace

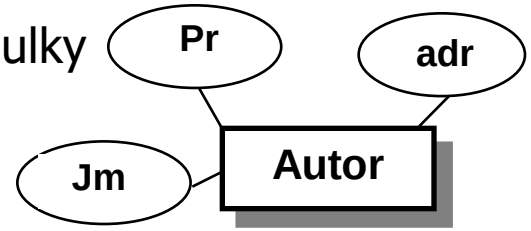
| Autor | | |
|---|---|---|
| Pr | Jm | Adr |
| | | |

| Titul | | | |
|---|---|---|---|
| Pr | Jm | Naz | Dat |
| | | | |

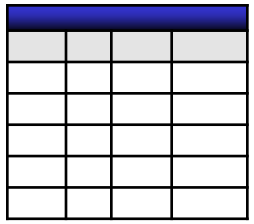
# Vlastnosti sloupců



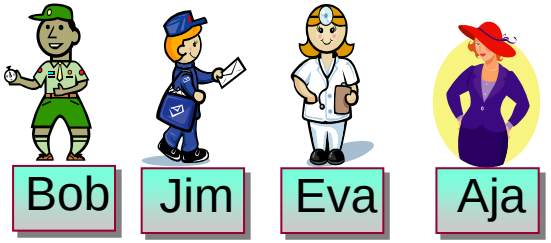
- vlastnosti entit popisují sloupce tabulky
- sloupce mají jména
- sloupce mají pořadí
- instance [výskyty entit ] představují řádky




| Autor | | |
|---|---|---|
| Pr | Jm | Adr |
| 1 | 2 | 3 |



- instance musí být rozlišitelné – identifikovatelné




| | | | |
|---|---|---|---|
| Bob | | | |
| Jim | | | |
| Eva | | | |
| Aja | | | |
| .... | | | |

- řádky nemají jména a nemají pořadí [???]

- jeden ze sloupců [nebo skupina sloupců] musí řádky identifikovat
- sloupce [kromě jména a typu dat] potřebují mít **vlastnosti**

# NULL vlastnost

- každému sloupci lze určit zda může či nesmí obsahovat hodnotu **NULL**
- pokud nezadáme sloupec může mít **NULL**y
- když tak **NULL**y musí být **zakázány**

```
Create Table T1 (a  Int                 ,
                                  [ a  Int Null ]
                    b  Int  Not Null,
                     .....  );
```

**strč do b sloupce**

```
Insert Into T1       values(8 ,6);
Insert Into T1(b)    values(4);
Insert Into T1(a)    values(1);
Insert Into T1(a,b) values(1,NULL);
```

er

er

| T1 | |
|---|---|
| a | b |
| 8 | 6 |
| Null | 4 |
| ~~1~~ | ~~Null~~ |

er

er

- **NULL** hodnoty do tabulky – explicitně či **implicitně**

# Vlastnost jednoznačnosti

- vlastnost **Unique** vyžaduje jednoznačnost sloupce ve všech řádcích tabulky
- v žádné řádce se daná hodnota nesmí opakovat
- sloupec Unique ~~je~~ identifiční **není**

```
Create Table T2 (a  Int          Unique ,
                 b  Char(4)      Unique );

Insert Into T2       values( 1, 'joe');
Insert Into T2        values(2, 'jim');
Insert Into T2(b)   values (    'bil');
Insert Into T2 (b)   values(    'hal');

Insert Into T2        values(1 , 'jim');
```

?

| T2 | |
|---|---|
| a | b |
| 1 | joe |
| 2 | jim |
| Null | bil |
| Null | hal |

er

# Vlastnost jednoznačnosti



- jednoznačnost může platit pro více sloupců najednou
- pro osoby [ zaměstnance ] může platit jednoznačnost (Jmeno,Prijmeni)
- v takovém případě uvádíme jednoznačnost v samostatné řádce definice

```
Create Table Zam (Jm      Varchar( 20 ) Not Null   Unique ,
                  Pr      Varchar( 20 ) Not Null   Unique ,
                  Nar     Date          Not Null)  ,
                  unique  (Jm, Pr)
 );
```

# Vlastnost sloupců - vlastnosti tabulek - Constraints



- vlastnosti sloupců omezují možné hodnoty sloupců
- nazývají se tedy [ Integritní ] Omezení IO -- **Constraints**
- jsou vedeny jako omezení tabulek
- existuje 5 [ až 6] IO omezení

  - **Not NULL**
  - **Unique**
  - **Primary Key**
  - **Check**
  - **Foreign Key**

  - **[ Default ]**

# Primary Key



- každý sloupec NOTT NULL a UNIQUE je **identifikátorem** – Kandidátním klíčem
- každá skupina sloupců NOTT NULL a UNIQUE je **identifikátorem**
- z kandidátních sloupců je třeba vybrat jeden – **PRIMARY KEY**
- každá tabulka má mít Primární klíč - je to dobrý mrav
- primární klíč je privilegovaný – rozumné uvádět jako 1. sloupec
- **primární klíč značí** – **Not Null Unique [a privilegium]**

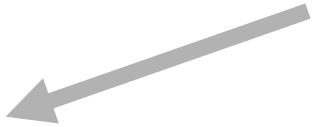
```
Create Table T5 (a        Int   Primary Key,
                 b        ….. ,
                 c        …,
                   …..
);
```

```
Create Table T5 (a        Int ,
                 b         Int,
                 c        …,
                 Primary Key (a,b)
);
```

# Primární klíč – co když nemám sloupec pro



- **umělý klíč**
  - **uměle vytvořená vlastnost objektu** - umělý sloupec
  - obsahuje **jednoznačnou identifikaci** objektu
  - hodnotou mohou být např. pořadová čísla založení řádku
  - použití je **pouze pro identifikaci**
  - **ne**ní určeno **ke zjišťování pořadí** !! [ trochu ano ]
- CisloOsoby , CisOs



| CisOs | Jmeno | Město | Adresa | Stát |
|---|---|---|---|---|
| | Alois Švehla | Benešov | Uliční 1 | CZ |
| | Jan Novák | Praha | Krásná 4 | CZ |
| | Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| | Lucie Černá | Bratislava | Ušmouněná 555 | SK |

# Primární klíč

- **umělý klíč**
  - **uměle vytvořená vlastnost objektu** - umělý sloupec
  - obsahuje **jednoznačnou identifikaci** objektu
  - hodnotou mohou být např. pořadová čísla založení řádku
  - použití je **pouze pro identifikaci**
  - **ne** **k určení pořadí** !!
- CisloOsoby , CisOs

| CisOs | Jmeno | Mesto | Adresa | Stat |
|---|---|---|---|---|
| 1 | Alois Švehla | Benešov | Uliční 1 | CZ |
| 2 | Jan Novák | Praha | Krásná 4 | CZ |
| 3 | Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| 4 | Lucie Černá | Bratislava | Ušmouněná 555 | SK |

# Primární klíč



```
Create Table Osoba   (
        CisOs       Serial   Primary Key,
        Jmeno       ….. ,
        Mesto       …..
        …..
);
```

| CisOs | Jmeno | Mesto | Adresa | Stat |
|---|---|---|---|---|
| 1 | Alois Švehla | Benešov | Uliční 1 | CZ |
| 2 | Jan Novák | Praha | Krásná 4 | CZ |
| 3 | Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| 4 | Lucie Černá | Bratislava | Ušmouněná 555 | SK |

# Primární klíč



```
Create Table Osoba   (
        CisOs       Serial   Primary Key,
        Jmeno       ….. ,
        Mesto       …..
        …..
);
```

```
Insert Into Osoba (Jmeno,Mesto,…)  Values('Alois …' ,…..);          1
Insert Into Osoba (Jmeno,Mesto,…)  Values('Jan …' ,…..);           2
Insert Into Osoba (Jmeno,Mesto,…)  Values('Petr …' ,…..);          3
Insert Into Osoba (Jmeno,Mesto,…)  Values('Vaclav' ,…..);          er   4
Insert Into Osoba (Jmeno,Mesto,…)  Values('Alois …' ,…..);   ???   5
```

| CisOs | Jmeno | Mesto | Adresa | Stat |
|---|---|---|---|---|
| 1 | Alois Švehla | Benešov | Uliční 1 | CZ |
| 2 | Jan Novák | Praha | Krásná 4 | CZ |
| 3 | Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| 4 | Lucie Černá | Bratislava | Ušmouněná 555 | SK |

# Primární klíč



- může být složený
  - tvoří jej více sloupců
- osoby jsou našimi pacienty
- identifikací je kód Pojišťovny a jejich číslo pojištěnce
  - KodPoj, CisPoj

| Jmeno | Mesto | Adresa | Stat |
|---|---|---|---|
| Alois Švehla | Benešov | Uliční 1 | CZ |
| Jan Novák | Praha | Krásná 4 | CZ |
| Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| Lucie Černá | Bratislava | Ušmouněná 555 | SK |

# Primární klíč



- může být složený
  - tvoří jej více sloupců
- osoby jsou našimi pacienty
- identifikací je kód Pojišťovny a jejich číslo pojištěnce
  - KodPoj, CisPoj

| KodPoj | CisPoj | Jmeno | Mesto | Adresa | Stat |
|---|---|---|---|---|---|
| 120 | 2222 | Alois Švehla | Benešov | Uliční 1 | CZ |
| 130 | 3333 | Jan Novák | Praha | Krásná 4 | CZ |
| 110 | 1111 | Alois Švehla | Ostrava | Ošklivá 7 | CZ |
| 120 | 3333 | Lucie Černá | Bratislava | Ušmouněná 555 | SK |

**Primární klíč složený**

# Check – rádi bychom kladli podmínky pro správnost dat

- Sloupci [či skupině sloupců] lze zadat podmínku pro testování uložených hodnot
- podmínka se zadá **výrazem** omezení CHECK (podmínka)
- výraz musí být **skalární**

```
Create Table T5 (a        Int     Primary Key   Check  (a>0),
                 b        Int     Check  (b >0 and b < 100 ),
                 c        Int     Chcek  (c  Between 1  and 99),
                 Check    ( a+b < c ),
                 T1       Varchar(30),
                 T2       VarChar(30),
                 Check    (Length(T1) + Length (T2) < 70 )
);
```

- výraz se kontroluje při jakékoliv změně dat tabulky
- při **INSERT** , **UPDATE**
- vyjde li podmínka **FALSE**, příkaz se neprovede

- součástí výrazu **CHECK** nesmí být (skalární) **Select** -PROČ ???

# Check – rádi bychom kladli podmínky pro správnost dat

- Sloupci [či skupině sloupců] lze zadat podmínku pro testování uložených hodnot
- podmínka se zadá výrazem omezení CHECK (podmínka)
- výraz musí být **skalární**

```
Create Table T5 (a        Int     Primary Key,
                 b        Int     ,
                 c        Int     ,
                 T1       Varchar(30),
                 T2       VarChar(30),

                 Check    ( a     > 0     ),
                 Check    ( b     < 100   ),
                 Check    ( a+b < c       ),
                 Check    (Length(T1) + Length (T2) < 70 )

);
```

# Závislosti tabulek



```
Create Table Zam (
    Jm       Varchar (30)   Not Null,
    Pr       Varchar (30)  Not Null,
    StPr     VarChar(5)  NotNull
             References  Zeme  (Zkr) ,
    Primary Key   (Jm,Pr)
);

Create Table   Zeme(
     Zkr    VarChar(5)   Primary Key,
     Naz    VarChar(30) Not Null Unique
)
```

| Zam | | |
|---|---|---|
| Jm | Pr | Stpr |
| Jan | Pich | CZ |
| Piere | Macron | FR |
| Joe | Bull | GB |
| Hugo | Ruf | UG |

| Zeme | |
|---|---|
| Zkr | Naz |
| CZ | Česko |
| FR | Francie |
| GB | Anglie |

- Mezi tabulkami existuje vztah **StatníPřislušnost -> Zeme**
- Chceme mít kontrolu, zda zeme v tabulce Zam existuje i v tabulce Zeme
- Zavedeme tedy **VTAH MEZI TABULKAMI**
- **DB bude automaticky kontrolovat, zda vztah existuje**
- Když neexistuje, insert či update zápis nepřipustí

# Závislosti tabulek



```
Create Table Zam (
    Jm      Varchar (30)   Not Null,
    Pr      Varchar (30)   Not Null,
    StPr    VarChar(  5)   Not Null
            References  Zeme  (Zkr) ,
    Primary Key   (Jm,Pr)
);

Create Table   Zeme(
     Zkr    VarChar(  5)  Primary Key,
     Naz    VarChar(30) Not Null Unique
)
```

| Zam | | |
|---|---|---|
| Jm | Pr | Stpr |
| Jan | Pich | CZ |
| Piere | Macron | FR |
| Joe | Bull | GB |
| ~~Hugo~~ | ~~Ruf~~ | ~~UG~~ |

| Zeme | |
|---|---|
| Zkr | Naz |
| CZ | Česko |
| FR | Francie |
| GB | Anglie |

```
Insert Into Zam Values ('Jan',     'Pich',     'CZ');
Insert Into Zam Values ('Piere',   'Macron',   'FR');
Insert Into Zam Values ('Joe',     'Bull',     'GB');
Insert Into Zam Values ('Hugo',    'Ruf',      'UG');
```

er

# Závislosti tabulek

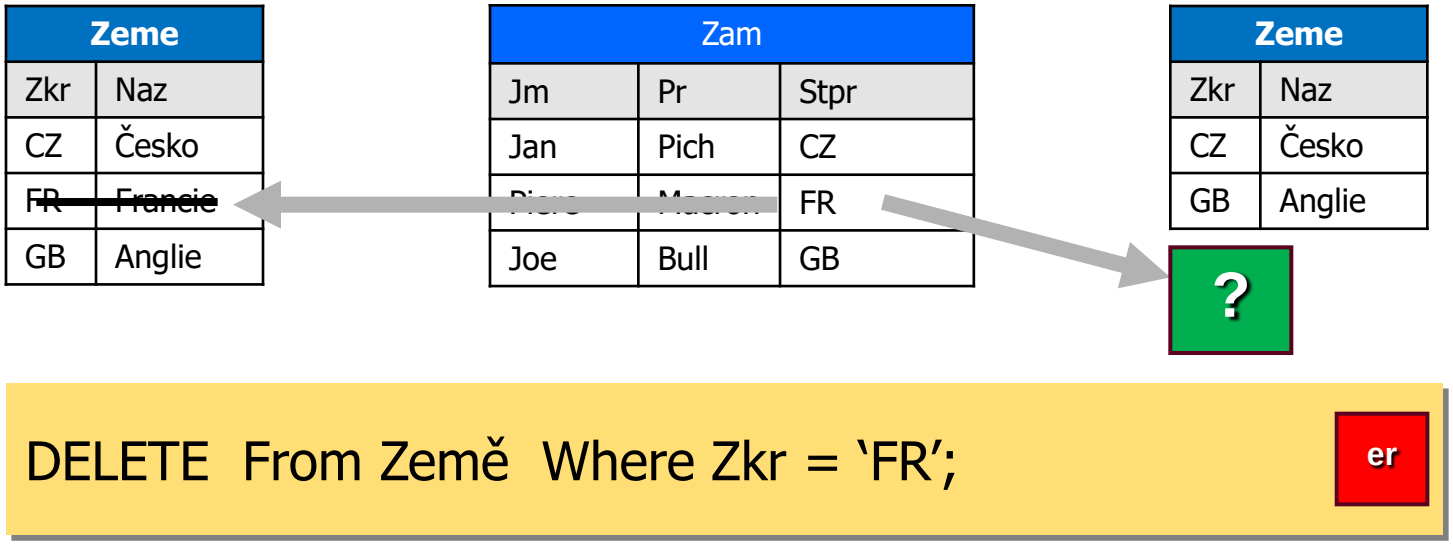


```
Create Table Zam (
    Jm      Varchar (30)   Not Null,
    Pr      Varchar (30)   Not Null,
    StPr    VarChar(5)     NotNull   References  Zeme  (Zkr) ,

    Primary Key   (Jm,Pr)
);
```

```
Create Table Zam (
    Jm      Varchar (30)   Not Null,
    Pr      Varchar (30)    Not Null,
    StPr    VarChar(5)     NotNull,

    Primary Key   (Jm,Pr),
    Foreign Key   (Stpr)  References  Zeme  (Zkr)
);
```

# Cizí klíče






DELETE From Země Where Zkr = 'FR';

er

- nelze připustit vynechání řádky
- na kterou se odvolávají nějaké **závislé řádky** z jiné tab

- to je ovšem mnohdy nepříjemné
- nedalo by se to **někdy nějak** zařídit
- aby to šlo ?

# Cizí klíče



| Zam | | |
|---|---|---|
| Jm | Pr | Stpr |
| Jan | Pich | CZ |
| Piere | Macron | FR |

| Zeme | |
|---|---|
| Zkr | Naz |
| CZ | Česko |
| GB | Anglie |

```
Create Table Zam (
    Jm        ....,
    Pr        ....,
    StPr      VarChar(5)      NotNull,

    Foreign Key    (Stpr)  References  Zeme  (Zkr)
                    [ ON DELETE       volba ]   -  řádku v Zeme
                    [ ON UPDATE       volba ]   -  řádku v Zeme
);

Volba         RESTRICT   |
              CASCADE    |
              SET NULL   |
              NO  ACTION |
              SET DEFAULT
```

# Umístění metadat v DB



- Meta data jsou uložena v databázi - ve tvaru tabulek
- [ tak kde tam jsou ??? ] ve speciálním schematu
- **information_schema**
- Výpis sloupců z tabulky Osoba schematu Evidence

```
SELECT    *
  FROM     information_schema.columns
  WHERE   table_schema = ‘Evidence'
   AND      table_name    = ‘Osoba' ;
```

# Vlastnosti sloupců a tabulek jako omezení



- data v DB mají být správná – **voňavá** - konsistentní
- vlastnosti sloupců a tabulek omezují data ve sloupcích
- jsou to pravidla, která musí být jako metadata v DB
- v DB jsou uloženy DB objekty
- [integritní] omezení **IO** jsou objekty – CONTRAINTs
- jsou uloženy v meta-tabulce **CONSTRAINTS**
- každý **DB objekt** má **typ** objektu a **jméno**
- **jméno IO omezení** generuje systém
- každé IO můžete pojmenovat sami

```
Craete Table T  (
    a    ….
    b    ….
    CONSTRAINT  Muj_FK  Foreign Key (a)
                                References  Zeme (KodZeme)
                                On Delete ….
);
```

# Select

```
SELECT    Konstanta | Sloupec | Výraz | PodDotaz | *   [položka2, položka3,....]
   From     Tabulka
[   Where   Výraz-VýběruŘádek                          ]
[Order By   SloupecA   | , SloupecB ,....
            |   3 , 1 , čísloPoložkyDotazu , ...        ]
```

- v jakém pořadí se jednotlivé klauzule provádějí ?????

## Vyhodnocení dotazu

- FROM
- více tabulek -- kartézský součin
- WHERE

- SELECT
- ORDER BY

# Příkaz Select



- obě výsledné tabulky - stejný počet sloupců
- sloupce obou dotazů - odpovídající typy
- žádná nesmí být tříděna ORDER BY

```
Select  Jmeno,.. From .. Order by Jmeno...
UNION
Select  Jmeno,..  From .. Order by Jmeno desc
UNION
Select  Jmeno,Plat,.. ... Order by Plat
```

Vyhodnocení dotazu

- FROM
- WHERE
- SELECT
- ORDER BY

opakuje se pro každý dotaz spojení

- jiný časový okamžik vyhodnocování

a zas něco jiného

# Statistiky



















# Statistiky


















# Statistiky

- o celém seskupení
- chceme vědět nějaké celkové údaje
  - kolik těch lidí je
    - kolik je žen
    - kolik můžů
  - jaký mají celkový majetek
  - jaký mají průměrný příjem
    - ...

- statistiky

[SQL]

- agregace

# Agregace

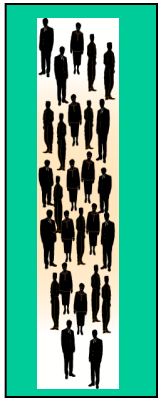
- [SQL] 5 agregačních funkcí

- počet
  - `Count (...)`
    - `Count (všeho)`
    - `Count (nějakéVlastnosti)`
- součet
  - `Sum (nějakéVlastnosti)`
- maximum
  - `Max (nějakéVlastnosti)`
- minimum
  - `Min (nějakéVlastnosti)`
- průměr
  - `Avg (nějakéVlastnosti)`

# Agregace

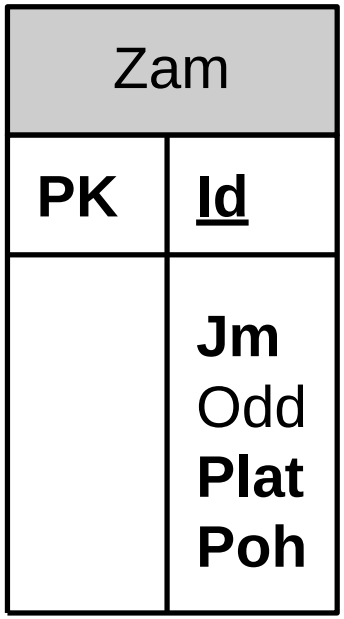
- [SQL] 5 agregačních funkcí

| | |
|---|---|
| **Count (...)** | |
| **Count (*)** | počet řádků |
| **Count (Sloupec)** | počet hodnot ve sloupci |
| **Sum (Sloupec)** | součet všech hodnot ve sloupci |
| **Max (Sloupec)** | maximální hodnota sloupce |
| **Min (Sloupec)** | minimální hodnota sloupce |
| **Avg (Sloupec)** | průměrná hodnota sloupce |



# Agregační dotazy






| Zam | | | | |
|---|---|---|---|---|
| Id | Jm | Odd | Plat | Poh |
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- kolik je zaměstnanců
- jaký je nejvyšší plat
- jaký je nejmenší plat
- kolik dělají platy celkově
- kolik je oddělení

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Count (*) From Zam → 8

Select Count (Jm) From Zam → 8

Select Min (Plat) From Zam → 10 000

Select Max (Plat) From Zam → 40 000

Select Sum (Plat) From Zam → 160 000

Select Avg (Plat) From Zam → 20 000

Select Count (Odd) From Zam → ? 7

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Count (Plat) From Zam → 8

Select Count (Distinct Plat) From Zam → 4

Select Sum (Distinct Plat) From Zam → 100 000

Select Avg (Distinct Plat) From Zam → 25 000

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

**Agregace (Distinct Sloupec)**

- zkoumá jedinečné hodnoty sloupce [bez duplicit]

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Min (Distinct Plat) — 10 000

From Zam

Select Max (Distinct Plat) — 40 000

From Zam

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

## Agregace (Distinct Sloupec)

- zkoumá jedinečné hodnoty sloupce [bez duplicit]
- u agregací Min i Max lze zadat
- nemá význam

# Agregační dotazy - agregační funkce

| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Count (Distinct Poh) From Zam

Select Count (Distinct Plat) From Zam

| | Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|---|
| 2 | 1 | Pepa | 100 | 10 000 | M |
| | 2 | Jana | 200 | 20 000 | Z |
| 4 | 3 | Adam | 100 | 30 000 | M |
| | 4 | Tom | 300 | 20 000 | M |
| | 5 | Hana | 400 | 40 000 | Z |
| | 7 | Zuzi | 300 | 20 000 | Z |
| | 6 | Bob | 100 | 10 000 | M |
| | 8 | Dan | | 10 000 | M |

**Agregace (Distinct Sloupec)**

- zkoumá jedinečné hodnoty sloupce [bez duplicit]

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Count (Poh) From Zam → 8

Select Count (**Distinct** Poh) From Zam → 2

Select Count (Plat) From Zam → 8

Select Count (**Distinct** Plat) From Zam → 4

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

**Agregace (Distinct Sloupec)**

- zkoumá jedinečné hodnoty sloupce [bez duplicit]

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Min (Jm) From Zam — Adam

Select Max (Jm) From Zam — Zuzi

Select Count (*) From Zam — 3

Where Odd=100

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

```
Select Min   (Jm) From Zam                    Adam
Select Max   (Jm) From Zam                    Zuzi
Select Count (*)  From Zam                    3
                  Where Odd=100
Select Sum(Plat),                             50 000
       Min(Plat),                             10 000
       Max(Plat)                              30 000
  From Zam       Where Odd=100
```

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

# Agregační dotazy - agregační funkce



| | |
|---|---|
| Count (*) | počet řádků výsledku |
| Count (Sloupec) | počet NOT NULL hodnot ve sloupci |
| Sum (Sloupec) | součet všech hodnot ve sloupci |
| Max (Sloupec) | maximální hodnota sloupce |
| Min (Sloupec) | minimální hodnota sloupce |
| Avg (Sloupec) | průměrná hodnota sloupce |

Select Count (*) From Zam — 8

Where Plat > 30 0000 — 1

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- leč já bych rád
- statistikoval jináč !!!

# Statistiky - agregace





















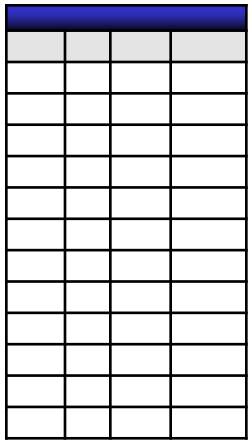
# Statistiky - agregace

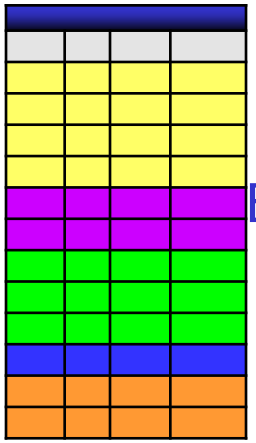
- statistiky pouze za celek ?
- celek rozdělit na skupiny
  - dle nějaké vlastnosti
  - dle nějakých vlastností
- stanovit statistiky - agregace
  - za jednotlivé skupiny
  - z hodnot [lidí] ze skupin

```
Select Count(*)  ...
Select Min  (Jm) ...
```

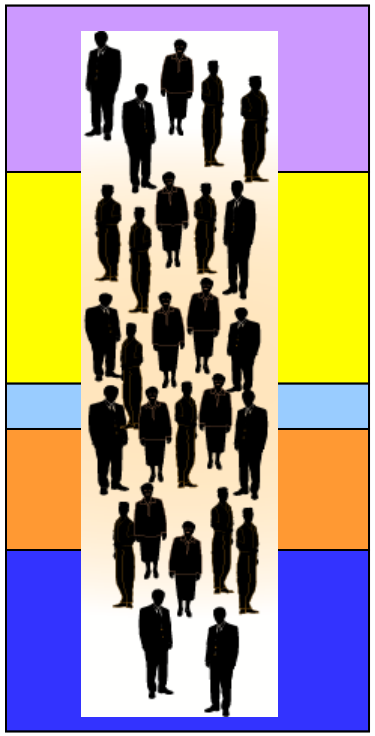
- i tabulky mají
  - schopnost
  - se sociologizovat
  - seskupovat

- jak ?



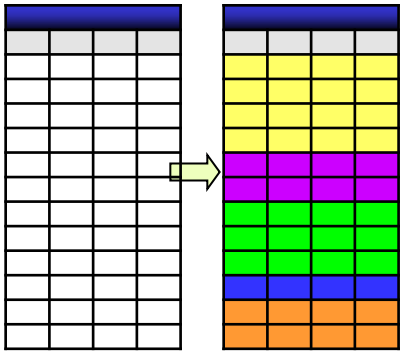


15
Adam
35
Gabi
4
8
Bob
32
Jim



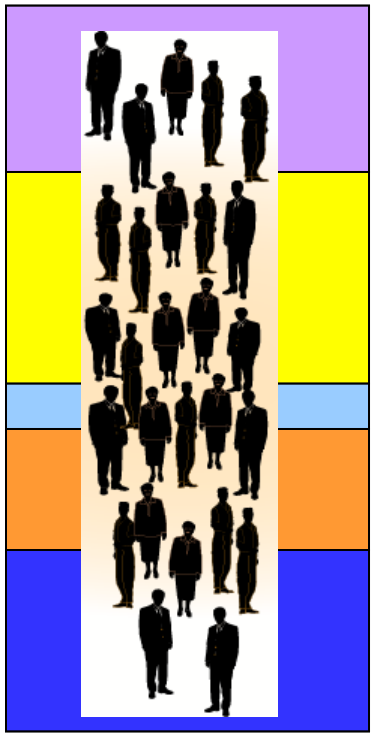
# Statistiky - agregace

- statistiky pouze za celek ?
- celek rozdělit na skupiny
  - dle nějaké vlastnosti
  - dle nějakých vlastností
- stanovit statistiky - agregace
  - za jednotlivé skupiny
  - z hodnot [lidí] ze skupin



```
Select Count(*)  ...
Select Min  (Jm) ...
```

15
Adam
35
Gabi



- dělení na skupiny **klauzulí**

4
8
Bob
32
Jim

```
Group By    Vlastnost1, Vlastnost2, ...
```

```
Group By    Sloupec [, Sloupec2, ... ]
```

# Sloupcové dotazy s klauzulí GROUP BY



```
SELECT      Odd,
            COUNT(*), MIN(Plat), MAX (Plat), SUM (Plat)
    GROUP BY Odd
```

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |


| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 8 | Dan | | 10 000 | M |
| 1 | Pepa | 100 | 10 000 | M |
| 3 | Adam | 100 | 30 000 | M |
| 6 | Bob | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 4 | Tom | 300 | 20 000 | M |
| 7 | Zuzi | 300 | 20 000 | Z |
| 5 | Hana | 400 | 40 000 | Z |

# Sloupcové dotazy s klauzulí GROUP BY



```
SELECT      Odd,
            COUNT(*), MIN(Plat), MAX (Plat), SUM (Plat)
    GROUP BY Odd
```

1. skupina →

2. skupina →

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 8 | Dan | | 10 000 | M |
| 1 | Pepa | 100 | 10 000 | M |
| 3 | Adam | 100 | 30 000 | M |
| 6 | Bob | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 4 | Tom | 300 | 20 000 | M |
| 7 | Zuzi | 300 | 20 000 | Z |
| 5 | Hana | 400 | 40 000 | Z |

Ve skupině se agreguje

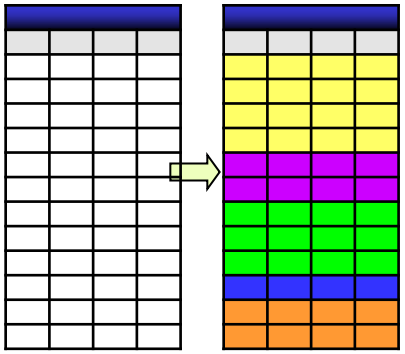
- za oddělení ----> 100
- count(*) ----> 3
- Min (Plat) ----> 10 000
- Max (Plat) ----> 30 000
- Sum (Plat) ----> 50 000

| Odd | Count | Min | Max | Sum |
|---|---|---|---|---|
| 100 | 3 | 10 000 | 30 000 | 50 000 |

# Zásady skupinového dotazu - omezení

Seskupující sloupce

sloupce v klauzuli GROUP BY



- musí být sloupce tabulky

  nelze seskupovat dle výrazu

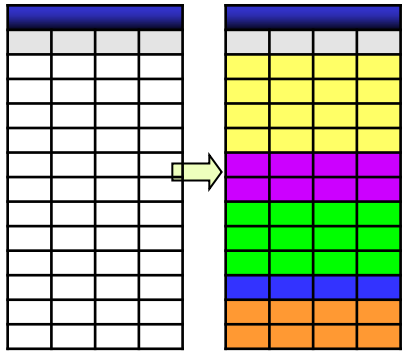
  (vypočteného)

```
SELECT      ...,
            COUNT(*), MIN(...), MAX (...), SUM (...)
  GROUP BY  Plat + 1000
```

error

proč asi ?

# Zásady skupinového dotazu - omezení



SELECT položka - dotaz **může** žádat pouze

- něco, čeho je ve skupině pouze jedno
- tj
- konstantu – pro celou skupinu mi dej 77
- seskupující sloupec (z GROUP BY)- **označení skupiny**
- agregační funkci
- výraz - kombinaci předchozího

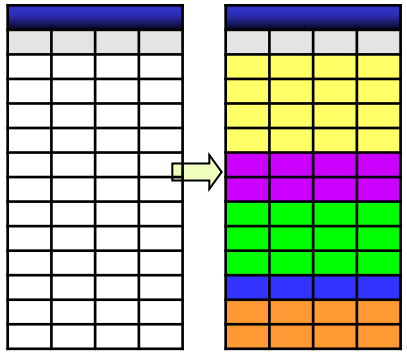
SELECT **nesmí** požadovat

- něco, čeho je ve skupině více
- co by se z toho asi mělo vybrat **?????**

# Zásady skupinového dotazu - omezení

SELECT položka - dotaz **může** žádat pouze

- konstantu
- seskupující sloupec (z GROUP BY)-označení skupiny
- agregační funkci
- výraz - kombinaci předchozího



SELECT **nesmí** požadovat

- sloupec - není v Group By
- [neseskupující sloupec]
- výraz s takovým sloupcem

```
SELECT      Odd, JmenoZamestnance,
            COUNT(*), MIN(...), MAX (...), SUM (...)
....
   GROUP  BY  Odd
```

error

proč asi ?

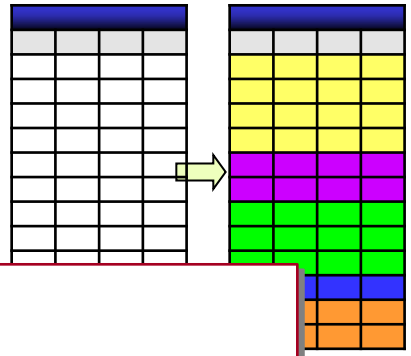
# Zásady skupinového dotazu - omezení

SELECT položka - dotaz **může** žádat pouze

- konstantu
- seskupující sloupec (z GROUP BY)-označení skupiny
- agregační fu
- výraz - komb

SELECT **nesmí**

- sloupec - ne
- [nes
- výr



Jmen v jednom oddělení je mnoho.

Nevím které jedno se má vybrat.

Proto **odmínu požadavek**

```
SELECT     Odd, JmenoZamestnance,
           COUNT(*), MIN(...), MAX (...), SUM (...)
....
   GROUP  BY  Odd
```

error

proč asi ?

# Sloupcové dotazy s klauzulí WHERE



```
SELECT odd, COUNT(*), MIN(Plat), MAX (Plat), SUM (Plat)
   WHERE Poh="M"
   GROUP BY odd
```

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 8 | Dan | | 10 000 | M |
| 1 | Pepa | 100 | 10 000 | M |
| 3 | Adam | 100 | 30 000 | M |
| 6 | Bob | 100 | 10 000 | M |
| 4 | Tom | 300 | 20 000 | M |

## Seskupování pouze na vybraných řádcích

- ženy se sekupování (agregačních výpočtů) nezůčastní

# Hodnoty NULL a agregační funkce

**agregační funkce zcela ignorují hodnoty Null**

| T... | | | |
|---|---|---|---|
| Jmeno | Odd | Plat | ... |
| Pepa | 100 | 10 000 | ... |
| Franta | 100 | Null | ... |
| Hana | 200 | 20 000 | ... |
| Jim | 300 | Null | ... |
| Tedy | 400 | Null | ... |
| Bobo | 400 | 30 000 | ... |

10 000 + Null --> Null

```
SELECT COUNT(Plat), Min(Plat), Max (Plat), Sum (Plat)
   From  T...
```

| | | | |
|---|---|---|---|
| Count | Min | Max | Sum |
| 3 | 10 000 | 30 000 | 60 000 |

**proč asi ?**

# Hodnoty NULL v seskupujících sloupcích



- [porovnání] **Null = Null** [-->Null]
- **pro seskupování GROUP BY**
  - ↗ **výjímečně považovány za shodné !!!**

| T... | | | |
|---|---|---|---|
| Jmeno | Odd | **Plat** | ... |
| Pepa | 100 | **10 000** | ... |
| Franta | 100 | **Null** | ... |
| Hana | 200 | **20 000** | ... |
| Jim | 300 | **Null** | ... |
| Tedy | 400 | **Null** | ... |
| Bobo | 400 | **30 000** | ... |

Null = Null --> Null

```
SELECT   Plat, COUNT(*), ....
  From    T...
Group By  Plat
```

| | | | |
|---|---|---|---|
| Plat | Count | | |
| Null | 3 | | |
| 10 000 | 1 | | |
| ... | ... | | |

# Hodnoty NULL v seskupujících sloupcích



- [porovnání] **Null = Null** [-->Null]
- **pro seskupování GROUP BY**
  - ↗ **výjímečně považovány za shodné !!!**

| T... | | | |
|---|---|---|---|
| Jmeno | Odd | **Plat** | ... |
| Pepa | 100 | **10 000** | ... |
| Franta | 100 | **Null** | ... |
| Hana | 200 | **20 000** | ... |
| Jim | 300 | **Null** | ... |
| Tedy | 400 | **Null** | ... |
| Bobo | 400 | **30 000** | ... |

Null = Null --> Null

```
SELECT   Plat, COUNT(*), ....
  From    T...
Group By  Plat
```

| | | | |
|---|---|---|---|
| Plat | Count | | |
| Null | 3 | | |
| 10 000 | 1 | | |
| ... | ... | | |

**proč asi ?**

# Výběr výsledných skupin



| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- zadáme seskupovací dotaz
- všechny vzniklé skupiny
- nás nezajímají
- zajímají nás pouze údaje
- o odděleních
  - mají alespoň dva zaměstnance

```
SELECT      Odd, Count(*), Min(Plat)
  From      Zam
GROUP BY    Odd

 Where      Count(*) > 1
```

error

proč?

# Výběr výsledných skupin



| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- zadáme seskupovací dotaz
- všechny vzniklé skupiny
- nás nezajímají
- zajímají nás pouze údaje
- o odděleních
  - mají alespoň dva zaměstnance

```
SELECT      Odd, Count(*), Min(Plat)
  From      Zam
GROUP BY    Odd

 Where      Count(*) > 1
```

```
 Having     Count(*) > 1
```

# Výběr výsledných skupin



| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- zadáme seskupovací dotaz
- všechny vzniklé skupiny
- nás nezajímají
- zajímají nás pouze údaje
- o odděleních
  - mají alespoň dva zaměstnance

```
SELECT      Odd, Count(*), Min(Plat)
  From      Zam
GROUP BY    Odd

 Having     Count(*) > 1
```

| Odd | Count | Min |
|---|---|---|
| 100 | 3 | 10 000 |
| 300 | 2 | 20 000 |

# Výběr výsledných skupin



| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- dotaz lze zadat současně s klauzulemi
  - **Where**
  - **Having**
- výběr do seskupování
- výběr sestavených skupin do výsledku

```
SELECT    Odd, Count(*), Min(Plat)
  From    Zam
 Where    Poh = "M"
GROUP BY  Odd
 Having   Count(*) > 1
```

| Odd | Count | Min |
|---|---|---|
| 100 | 3 | 10 000 |

- ženy se nedostaly vůbec do seskupování
- vyřazeny skupiny s 1 zaměstnancem [mužským]

# Vyhodnocení povelu Select [bez Group By]



```
SELECT      Jmeno, Odd, Plat+2000
  From      Zam
 Where      Poh = "M“
  Order by  Jmeno
```

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- **From** vyber všechy řádky a sloupce
- z tabulky From – ZAM
- **Where** pokud je klauzule WHERE
- vyber pouze odpovídající řádky
- **Select** vyber a vypočti výsledné sloupce [z klauzule Select ]
- **Order By** pokud zadána klauzule Order By
- přetřiď řádky dle zadaného
- výsledek zobraz [zašli klientovi]

# Vyhodnocení seskupeného povelu Select [Group By]



```
SELECT    Odd, Max(Plat), Min(Plat)
  From    Zam
  Where   Poh = "M“
 Group By Odd
 HAVING   Odd < 400
  Order  by  Odd
```

| Id | Jmeno | Odd | Plat | Poh |
|---|---|---|---|---|
| 1 | Pepa | 100 | 10 000 | M |
| 2 | Jana | 200 | 20 000 | Z |
| 3 | Adam | 100 | 30 000 | M |
| 4 | Tom | 300 | 20 000 | M |
| 5 | Hana | 400 | 40 000 | Z |
| 7 | Zuzi | 300 | 20 000 | Z |
| 6 | Bob | 100 | 10 000 | M |
| 8 | Dan | | 10 000 | M |

- **From** vše - řádky a sloupce z From
- **Where** pokud je, vyber pouze odpovídající řádky
- **Group By** setřiď dle zadaných skupin – vytvoř skupiny
- v jednotlivých skupinách proveď agregační funkce
- **Select** připrav 1 řádek pro každou skupinu – zadané sloupce
- **Order By** pokud zadána klauzule - přetřiď řádky dle zadaného
- výsledek zobraz [zašli klientovi]

# ANSI/ISO SQL standard



| | | | |
|---|---|---|---|
| 1986 | SQL86 | | ANSI |
| | SQL87 | | ISO |
| 1989 | SQL89 | SQL1 | Základní verze normy. Nedůležité. |
| 1992 | SQL92 | SQL2 | Moderní koncepce relační DB. |
| 1999 | SQL99 | SQL3 | OO, rekurze, reg-výrazy, spouště, výrazy nsk |
| 2003 | SQL2003 | (4) | xml, funkce oken, sekvence, gener.sloupce |
| 2006 | SQL2006 | (5) | xml-sql, XQuery |
| 2008 | SQL2008 | (6) | spouště INSTEAD OF, truncate, order by .... |
| 2011 | SQL2011 | (7) | 14 částí standardu |

# ISO/IEC 9075:2011

- sedmá verze standardu ISO - ANSI
- 14 částí

1. SQL Framework
2. SQL Foundation
3. Call Level Interface
4. Persistent Stored Modules
5. Embeded SQL
6. SQL Transaction
7. SQL Temporal
8. Objects/Extended Objects
9. Management of External Data
10. Object Language Binding
11. Schemata
12. Replication
13. Routines+Types Java PL
14. SQL/XML


Thanks