

Databázové systémy v biomedicíně

Lekce VI – Pokročilé konstrukce SQL

“Nadstandardní“ SQL

- Určování pořadí záznamů
- Hodnoty předchozích a následujících řádků
- Rozšířené agregace
 - Výpočet procent
 - Parciální agregace
 - Kumulativní součet
 - Klouzavý průměr

Rozšíření SQL o

... OVER (PARTITION BY sloupec ORDER BY sloupec)

Ranking functions – číslování řádků

RANK, DENSE_RANK, ROW_NUMBER

Sloupec X	RANK	DENSE_RANK	ROW_NUMBER
100	1	1	1
200	2	2	2
200	2	2	3
300	4	3	4
400	5	4	5

- **RANK() OVER ([PARTITION BY sex] ORDER BY date_of_birth DESC)**
- RANK() OVER (ORDER BY date_of_birth DESC NULLS LAST)
- **Není možné používat za WHERE a HAVING - nutné zanoření**
Can not be used after WHERE and HAVING as condition

Ranking function

Příklad:

```
SELECT patient_id, sex, date_of_birth,  
RANK( ) OVER (PARTITION BY sex ORDER BY date_of_birth DESC NULLS LAST),  
DENSE_RANK( ) OVER (PARTITION BY sex ORDER BY date_of_birth DESC NULLS LAST),  
ROW_NUMBER( ) OVER (PARTITION BY sex ORDER BY date_of_birth DESC NULLS LAST)  
FROM patients LIMIT 100
```

Využití v sekci WHERE – nutné zapouzdření

```
SELECT * FROM (  
  SELECT patient_id, sex, date_of_birth,  
  RANK( ) OVER (PARTITION BY sex ORDER BY date_of_birth DESC NULLS LAST) poradi  
  FROM patients) x  
WHERE poradi < 10
```

... Další varianta hledání nejstaršího studenta

LAG, LEAD

- LAG (value_expression [,offset] [,default]) OVER ([query_partition_clause] order_by_clause)
- LEAD (value_expression [,offset] [,default]) OVER ([query_partition_clause] order_by_clause)

- LAG = hodnota z předchozího řádku / *previous row value*
- LEAD = hodnota z následujícího řádku / *next row value*

```
SELECT study_id, TO_CHAR (date_of_enrollment, 'yyyy'), COUNT(*) letos,  
LAG(COUNT(*),1,'0') OVER(PARTITION BY study_id  
ORDER BY TO_CHAR (date_of_enrollment, 'yyyy') ) loni  
FROM patient_study  
GROUP BY study_id, TO_CHAR (date_of_enrollment, 'yyyy')  
ORDER BY study_id, TO_CHAR (date_of_enrollment, 'yyyy')
```

Pozn. POSTGRESQL 9.1: LAG(COUNT(),1, '0')*

Reportovací „window“ funkce

Agregační funkce s procentickým vyjádřením

```
SELECT COUNT(*) FROM student
```

```
SELECT studies, COUNT(*) FROM student  
GROUP BY studies
```

Procentické zastoupení – standardní SQL:

```
SELECT studies, COUNT(*) pocet , COUNT(*) * 100.0/(SELECT  
COUNT(*) FROM student) procento FROM student  
GROUP BY studies
```

Analytická funkce

```
SELECT studies, COUNT(*) pocet, COUNT(*) *100.0 /  
SUM(COUNT(*)) OVER () procento FROM student  
GROUP BY studies
```

Window and reporting function

Procentické zastoupení – standardní SQL:

```
SELECT study_id, COUNT(*),  
COUNT(*) * 100.0 / (SELECT COUNT(*) FROM patient_study) procento  
FROM patient_study  
GROUP BY study_id
```

Analytická funkce

```
SELECT study_id, COUNT(*),  
COUNT(*) / SUM(COUNT(*)) OVER () * 100 procento  
FROM patient_study  
GROUP BY study_id
```

Parciální součty

```
UPDATE student SET completiontype= 'Z'  
WHERE mod(uco,2) = 1 – Rozdělení datového souboru
```

```
SELECT completiontype, studies, count(*) pocet, COUNT(*)  
*100.0 / SUM(COUNT(*)) OVER () procento FROM student  
GROUP BY completiontype, studies  
ORDER BY completiontype
```

```
SELECT completiontype, studies, count(*) pocet, COUNT(*)  
*100.0 / SUM(COUNT(*)) OVER () procento ,  
COUNT(*) *100.0 / SUM(COUNT(*)) OVER (PARTITION BY  
completiontype) proc_podskupiny  
FROM student  
GROUP BY completiontype, studies  
ORDER BY completiontype
```


Parciální součty

Procentické zastoupení pracovišť (počtu jejich pacientů) v jednotlivých studiích

```
SELECT study_id, study_site, COUNT(*),  
COUNT(*) / SUM(COUNT(*)) OVER (PARTITION BY study_id) * 100 procento  
FROM patient_study  
GROUP BY study_id, study_site
```

Kumulativní součet

```
SELECT studies, COUNT(*) pocet FROM  
student  
GROUP BY studies
```

```
SELECT studies, COUNT(*) pocet,  
SUM(COUNT(*)) OVER (ORDER BY studies)  
FROM student  
GROUP BY studies
```

```
SELECT sex, studies, COUNT(*) pocet,  
SUM(COUNT(*)) OVER (PARTITION BY sex ORDER BY studies)  
kumulace_skupina,  
SUM(COUNT(*)) OVER (ORDER BY sex, studies) kumulace_celkem FROM  
student  
GROUP BY sex, studies  
ORDER BY sex, studies
```

Klouzavý průměr

AVG(sloupec) OVER

(ORDER BY sloupec ROWS BETWEEN x PRECEDING AND CURRENT ROW)

• **ROWS BETWEEN**



- **UNBOUNDED PRECEDING**
- **UNBOUNDED FOLLOWING**
- **CURRENT ROW**
- **počet řádků PRECEDING**
- **počet řádků FOLLOWING**

```
CREATE TABLE pocet_pacientu as
SELECT TO_CHAR(date_of_enrollment, 'yyyy-mm') mesic, COUNT(*)
pocet FROM patient_study
WHERE date_of_enrollment >= '2004-01-01'
GROUP BY TO_CHAR(date_of_enrollment, 'yyyy-mm')
ORDER BY TO_CHAR(date_of_enrollment, 'yyyy-mm')
```

Klouzavý průměr

```
SELECT * FROM pocet_pacientu  
ORDER BY mesic
```

```
SELECT AVG(pocet) FROM  
pocet_pacientu
```

```
SELECT mesic, pocet,  
ROUND(AVG(pocet) OVER (ORDER BY mesic ROWS  
BETWEEN 3 PRECEDING AND CURRENT ROW),1)  
klouzavy_prumer  
FROM pocet_pacientu
```

Cvičení/task

Spočítejte v tabulce pocet_pacientu
Compute on table pocet_pacientu

- Kumulativní počet pacientů
- Procento měsíčního počtu
 - k celkovému počtu
 - k maximálnímu počtu
 - k průměrnému počtu
 - k ročnímu průměru
 - K předchozímu měsíci
- Klouzavý průměr za 2 uplynulé měsíce

Cvičení

```
SELECT mesic, pocet  
FROM pocet_pacientu  
ORDER BY mesic
```

Cvičení

```
SELECT mesic, pocet,  
SUM(pocet) OVER (ORDER BY mesic) kumulativni_pocet,  
SUM(pocet) OVER () suma,  
MAX(pocet) OVER () maximum,  
AVG(pocet) OVER () prumer,  
AVG(pocet) OVER (PARTITION BY SUBSTR(mesic,1,4))  
ročni_prumer,  
LAG(pocet,1,'0') OVER (ORDER BY mesic) predchozi,  
ROUND(AVG(pocet) OVER (ORDER BY mesic ROWS BETWEEN 2  
PRECEDING AND 1 PRECEDING),1) klouzavy_prumer  
FROM pocet_pacientu  
ORDER BY mesic
```

Cvičení

- Zanoření a dopočet procent

```
SELECT mesic, pocet,  
pocet * 100/ suma suma_proc,  
pocet * 100/ maximum max_proc,  
pocet * 100 / predchozi predchozi_proc  
FROM (  
SELECT mesic, pocet,  
SUM(pocet) OVER (ORDER BY mesic) kumulativni_pocet,  
SUM(pocet) OVER () suma,  
MAX(pocet) OVER () maximum,  
AVG(pocet) OVER () prumer,  
AVG(pocet) OVER (PARTITION BY SUBSTR(mesic,1,4)) rocni_prumer,  
LAG(pocet,1,'0') OVER (ORDER BY mesic) predchozi,  
ROUND(AVG(pocet) OVER (ORDER BY mesic ROWS BETWEEN 2 PRECEDING  
AND 1 PRECEDING),1) klouzavy_prumer  
FROM pocet_pacientu  
) a  
ORDER BY mesic
```


Cvičení

- **Ošetření dělení nulou**

```
SELECT mesic, pocet,  
pocet * 100/ suma suma_proc,  
pocet * 100/ maximum max_proc,  
CASE WHEN predchozi > 0 THEN pocet * 100 / predchozi ELSE 0 END  
predchozi_proc  
FROM (  
....  
) a
```

Cvičení

- Zobrazte kumulativní procentické zastoupení pacientů podle věku
 - Věk, počet pacientů, kumulativní procento

Cvičení

- Zobrazte kumulativní procentické zastoupení pacientů podle věku
 - Věk, počet pacientů, kumulativní procento

```
SELECT EXTRACT (YEAR FROM AGE(date_of_birth))  
FROM patients limit 100
```

```
SELECT vek, COUNT(*) FROM (  
    SELECT EXTRACT (YEAR FROM AGE(date_of_birth)) vek  
    FROM patients) a  
WHERE vek > 0 and vek < 100  
GROUP BY vek  
ORDER BY vek
```

Cvičení

```
SELECT vek, pocet, kum_pocet * 100 / pocet_celkem kum_procento
FROM (
  SELECT vek, COUNT(*) pocet, SUM(COUNT(*)) OVER (ORDER BY VEK)
    kum_pocet, SUM(COUNT(*)) OVER () pocet_celkem
  FROM (
    SELECT EXTRACT (YEAR FROM AGE(date_of_birth)) vek
    FROM patients) a
  WHERE vek > 0 and vek < 100
  GROUP BY vek
  ORDER BY vek
) b
```

Another window functions

- FIRST_VALUE (sloupec)
- LAST_VALUE (sloupec)
- NTH_VALUE (sloupec, poradi)

Srovnání s únorovou hodnotou daného roku

```
SELECT mesic, pocet,  
       NTH_VALUE(pocet,2) OVER (PARTITION BY  
SUBSTR(mesic,1,4)) unor_rok  
FROM pocet_pacientu
```

Other aggregate functions

<https://www.postgresql.org/docs/15/functions-aggregate.html>

Medián

```
SELECT  
percentile_cont(0.5) WITHIN GROUP (ORDER BY a),  
percentile_disc(0.5) WITHIN GROUP (ORDER BY a)  
FROM (  
    SELECT a FROM generate_series(1,5) a  
) x
```

```
SELECT a, percent_rank() OVER (ORDER BY a)  
--percentile_cont (0.2) WITHIN group (ORDER BY a)  
FROM generate_series (10,20) a
```