MASARYKOVA UNIVERZITA
INSTITUT BIOSTATISTIKY A ANALÝZ

Databázové systémy a SQL

Lekce 6

Daniel Klimeš

Podmíněný výraz CASE

- CASE WHEN podminka THEN vysledek
  WHEN podminka2 THEN vysledek 2
  ELSE vysledek 3 END
- Až 127 WHEN,
- ELSE nepovinné,
- Vyhodnocování končí na první splněné podmínce
- Všechny výsledky musí být stejného datového typu

Příklad:

```
SELECT    vek,
          CASE WHEN vek IS NULL THEN 'neznamo'
            WHEN vek < 30 THEN 'kat < 30'
            WHEN vek < 50 THEN 'kat 30-49'
            WHEN vek < 65 THEN 'kat 50-64'
          ELSE  'kat 65 a starsi' END kategorie
FROM
(SELECT TRUNC(MONTHS_BETWEEN (SYSDATE,  date_of_birth) / 12) vek
FROM patients) /*ORACLE*/
(SELECT EXTRACT (YEAR FROM AGE(CURRENT_DATE,date_of_birth)) vek
 FROM patients) jmeno_vnoreneho /*POSTGRESQL*/
```

Ranking function – číslování řádků

RANK, DENSE_RANK, ROW_NUMBER

	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

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
```

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
```

Window a reporting function

Procentické zastoupení – standardní SQL:

```
SELECT study_id, COUNT(*),
COUNT(*) / (SELECT COUNT(*) FROM patient_study) * 100 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

```
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
```

Window a reporting function

1) Kumulativní procentické zastoupení :

```
SELECT study_id, COUNT(*),
  SUM(COUNT(*)) OVER (ORDER BY study_id
     ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
  / SUM(COUNT(*)) OVER () * 100
  kumul_procento
FROM patient_study
GROUP BY study_id
ORDER BY STUDY_ID
```

- **ROWS BETWEEN** →
  - **UNBOUNDED PRECEDING**
  - **UNBOUNDED FOLLOWING**
  - **CURRENT ROW**
  - **počet řádků PRECEDING**
  - **počet řádků FOLLOWING**

2) Klouzavý průměr:

```
AVG(COUNT(*)) OVER
(ORDER BY sloupec ROWS BETWEEN 5 PRECEDING AND CURRENT ROW)
```

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
- LEAD = hodnota z následujícího řádku

```
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')

Cvičení

1) Vytvořte sestavu: rok - měsíc, počet_nově zařazených pacientů, kumulativní_počet_pacientů
z tabulky patient_study sloupec date_of_enrollment

2) Vypište jen 10 měsíců s největším přírůstkem

3) Ve kterém kalendářním roce bylo těchto měsíců nejvíce?

Cvičení

Vypište kumulativní procentické zastoupení věku pacientek (po letech)
STUDY_ID = 169

- Nejprve spočítejte věk jednotlivých žen

```
SELECT p.patient_id, TRUNC(MONTHS_BETWEEN (sysdate, p.date_of_birth)/12)
FROM patients p, patient_study ps
WHERE p.patient_id = ps.patient_id and ps.study_id = 169
```

Cvičení

- Seskupte podle věku a přidejte kumulativní procenta

```
SELECT vek, COUNT(*),
 SUM(COUNT(*)) OVER (ORDER BY vek ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) /
 SUM(COUNT(*)) OVER () * 100 kum_proc
FROM (
SELECT p.patient_id, TRUNC(MONTHS_BETWEEN (sysdate, p.date_of_birth)/12) vek
FROM patients p, patient_study ps
WHERE p.patient_id = ps.patient_id and ps.study_id = 169
)
GROUP BY vek
```

Cvičení

Přidejte sloupec, který uvede rozdíl mezi hodnotou kumulativní četnosti aktuálního věku s předchozím řádkem

```
SELECT vek, pocet, kum_proc -  LAG(kum_proc, 1) OVER (ORDER BY vek) narust
FROM (
SELECT vek, COUNT(*) pocet,
 SUM(COUNT(*)) OVER (ORDER BY vek ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) /
 SUM(COUNT(*)) OVER () * 100 kum_proc
FROM (
SELECT p.patient_id, TRUNC(MONTHS_BETWEEN (sysdate, p.date_of_birth)/12) vek
FROM patients p, patient_study ps
WHERE p.patient_id = ps.patient_id and ps.study_id = 169
)
GROUP BY vek
)
ORDER BY vek
```

Cvičení na doma

- Najděte 5 studií s nejvyšším průměrným měsíčním přírůstkem nových formulářů

  - Nejprve připravte počty nových formulářů po měsících pro jednotlivé studie (EVENT_HEADER. DATE_COLLECTED)

```
SELECT study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm'),
COUNT(*) FROM event_header eh
GROUP BY  study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
ORDER BY study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
```

Cvičení

Přidejte sloupec, který bude obsahovat průměrný počet nových formulářů

```
SELECT study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm') mesic, COUNT(*),
AVG(COUNT(*)) OVER (PARTITION BY STUDY_ID) prumer
 FROM event_header eh
GROUP BY  study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
ORDER BY study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
```

Zapouzdřete a vytvořte sloupec s pořadím podle průměru sestupně

```
SELECT study_id, MAX(prumer), RANK() OVER (ORDER BY MAX(prumer) DESC)
FROM (
SELECT study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm') mesic, COUNT(*),
AVG(COUNT(*)) OVER (PARTITION BY STUDY_ID) prumer
 FROM event_header eh
GROUP BY  study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
ORDER BY study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm'))
GROUP BY study_id
ORDER BY max(prumer) DESC
```

Cvičení

Vyberte jen prvních 5 záznamů

```
SELECT * FROM (
SELECT study_id, MAX(prumer),
RANK() OVER (ORDER BY MAX(prumer) DESC) poradi FROM (
SELECT study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm') mesic, COUNT(*),
AVG(COUNT(*)) OVER (PARTITION BY STUDY_ID) prumer
FROM event_header eh
GROUP BY  study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm')
ORDER BY study_id, to_char(EH.DATE_COLLECTED, 'yyyy-mm'))
GROUP BY study_id
ORDER BY max(prumer) DESC)
WHERE poradi <= 5
```

- **Jaké je slabé místo uvedeného postupu?**