Vnořené dotazy a seskupování v SQL

Práce s daty, 20. dubna 2023

Vnořené dotazy

Jsou vždy v závorce a mohou se vyskytovat:

1. v sekci WHERE nadřazeného dotazu či jako parametr funkce
   - Výsledek vnořeného dotazu slouží k testování nějaké podmínky
   - `SELECT * FROM tab1 WHERE a IN (SELECT id FROM tab2);`
2. v sekci FROM nadřazeného dotazu
   - S výsledkem dotazu se pracuje jako s *odvozenou pojmenovanou* tabulkou
     1. Lze z ní vybírat data
     2. Lze je propojovat s jinými tabulkami
   - `SELECT * FROM tab1, (SELECT id FROM tab2) AS tid WHERE tab1.id=tid.id;`

Možnosti testování výsledku dotazu

- Výsledkem je jedna hodnota
  - Lze použít prakticky libovolný relační operátor (=, !=, >, <)
  - **`SELECT jmeno FROM zam WHERE plat > (SELECT AVG(plat) FROM zam);`**
- Výsledkem je jeden řádek
  - **`SELECT … WHERE (a, b) = (SELECT x, y FROM t2 WHERE id=56);`**
- Výsledkem je více řádků
  - (NOT) IN – zda se testovaná hodnota (ne)nachází mezi výsledky
  - ANY – zda testovaná podmínka platí alespoň pro jeden řádek poddotazu
  - ALL – zda testovaná podmínka platí pro všechny řádky poddotazu
  - (NOT) EXISTS – zda vnořený poddotaz (ne)vrátil alespoň jeden řádek

Příklady

Seznam jednotek je v tabulce *knihy*, přehled výpůjček v tabulce *vyp*.

- Seznam jednotek, které nejsou v přehledu výpůjček:

```
SELECT * FROM knihy WHERE id NOT IN (SELECT reckey FROM vyp);
```

- Jsou mezi jednotkami knihy z roku 2023?

```
SELECT IF(EXISTS (SELECT year FROM knihy WHERE year=2023), 'a', 'n');
```

- Počet jednotek, jejichž počet výpůjček je větší než *jakýkoliv / alespoň jeden* počet prodloužení:

```
SELECT COUNT(*) FROM knihy WHERE no_loans > ALL (SELECT renewals FROM vyp);
SELECT COUNT(*) FROM knihy WHERE no_loans > ANY (SELECT renewals FROM vyp);
```

Cvičení I

Vypište (bez duplicit) názvy titulů,

1. jejichž cena je vyšší jak průměrná (knihy s nulovou cenou nezapočítávejte)
2. které byly někdy vypůjčeny na více jak 50 dní
   - Lze řešit s pomocí IN, EXISTS i ANY
3. které byly ve všech případech vypůjčeny na více jak 50 dní
   - Toto asi bude těžší :-/

Počet dní lze zjistit pomocí `DATEDIFF(rd, ld)`

Párování tabulek pomocí `knihy.id=vyp.reckey`

Seskupování řádků

Řádky výsledky dotazu lze seskupovat pomocí klauzule **GROUP BY**

- Používá se často pro přehledy četností
- V projekční části dotazu *by měly / musí* být jen
  - ty sloupce, pomocí kterých se seskupuje, nebo
  - výsledky agregačních funkcí (COUNT, MIN, MAX, SUM, AVG apod.)
- Příklad: `SELECT call_no, count(*) FROM knihy GROUP BY call_no;`

Podmínky pro seskupené řádky zadáváme pomocí **HAVING**

- Příklad:
```
SELECT call_no, count(*) FROM knihy
    GROUP BY call_no HAVING count(*) > 5;
```

Příklady

- SELECT bortype, COUNT(*) FROM vyp GROUP BY bortype;
  - Počty čtenářů rozdělené dle jejich typu (student, zaměstnanec, ostatní)
- `SELECT CONCAT(year DIV 10,'x') as desetiletka, COUNT(*) FROM knihy GROUP BY desetiletka` **`WITH ROLLUP;`**
  - Počty jednotek rozdělené na desetiletí + řádek se součtem hodnot
- `SELECT CONCAT(year DIV 10,'x') as desetiletka, item_status, COUNT(*) FROM knihy GROUP BY desetiletka, item_status;`
  - Počty jednotek rozdělené na desetiletí a následně dle statusu
- `SELECT CONCAT(year DIV 10,'x') as item_status, desetiletka, COUNT(*) FROM knihy GROUP BY item_status, desetiletka;`
  - Počty jednotek rozdělené dle statusu a následně dle desetiletí

Okenní funkce (*window functions*)

Funkce, které ke každému řádku umožní dopočítat hodnotu závislou na jiných řádcích, přičemž řádky nemusí být na výstupu seskupeny

- *funkce*(*parametry*) OVER (PARTITION BY *sloupec* ORDER BY *sloupec*)

Možné funkce

- ROW_NUMBER – číslo řádku
- RANK – pořadí (více řádků může mít stejné pořadí)
- LAG / LEAD – přečte hodnotu z předchozího / následujícího řádku
- MEDIAN, PERCENTILE_CONT – medián, percentil
- Běžné agregační funkce jako MIN, MAX, SUM, COUNT, AVG, …

Příklady

- `SELECT ROW_NUMBER() OVER (ORDER BY title), title FROM knihy;`
  - Abecedně seřazený a očíslovaný seznam titulů
- `SELECT ROW_NUMBER() OVER (PARTITION BY title ORDER BY title) AS poradi, title FROM knihy;`
  - Abecedně seřazený seznam titulů, každý nový název se čísluje od jedničky
- `SELECT RANK() OVER (ORDER BY title), title FROM knihy;`
  - Abecedně seřazený seznam titulů, stejné názvy titulů mají stejné pořadí
- `SELECT LAG(call_no) OVER (ORDER BY call_no), call_no, LEAD(call_no) OVER (ORDER BY call_no)  FROM knihy WHERE call_no != '' GROUP BY call_no;`
  - Ke každé signatuře (*call_no*) vypíše předcházející a následující signaturu

Cvičení II

1. Vypište vynaložené náklady na koupi knih dle jednotlivých let podle roku založení záznamu (*open_date*).
2. Vypište počty jednotek rozdělené dle prefixu signatury (*call_no*) včetně celkového počtu všech jednotek na posledním řádku výpisu.
3. Vypište seznam všech titulů včetně pořadí seřazený dle počtu jednotek ve fondu (tj. tituly budou seskupeny dle názvu).