

Rozdział 5 Podzapytania

podzapytania proste i skorelowane, podzapytania w klauzuli SELECT, klauzula WITH, operatory ANY, ALL i EXISTS, zapytania hierarchiczne



Podzapytania

- Podzapytanie jest poleceniem SELECT zagnieżdżonym w innym poleceniu SELECT. Podzapytanie może wystąpić wszędzie tam, gdzie system spodziewa się zbioru wartości, czyli w klauzulach SELECT, FROM, WHERE, HAVING.
- Ogólny format zagnieżdżania zapytań:

- Operatorem może być:

- = <> < > <= >=
- IN
- ANY, ALL

```
SELECT atrybut1, atrybut2, ...  
FROM relacja  
WHERE atrybutn operator  
      (SELECT atrybuti, atrybutj  
       FROM relacja  
       WHERE warunek);
```

UWAGA!

W podzapytaniu nie może wystąpić klauzula ORDER BY



Podzapytania wyznaczające jedną krotkę

- Wyznacz pracownika zarabiającego najmniej w instytucie

```
SELECT nazwisko, etat, placa_pod  
FROM pracownicy  
WHERE placa_pod =  
      ( SELECT MIN(placa_pod)  
        FROM pracownicy );
```

208

- Wyznacz najgorzej zarabiającego asystenta

```
SELECT nazwisko, etat, placa_pod  
FROM pracownicy  
WHERE etat = 'ASYSTENT'  
AND placa_pod =  
      ( SELECT MIN(placa_pod)  
        FROM pracownicy  
        WHERE etat='ASYSTENT' );
```

371



Podzapytania wyznaczające wiele krotek

- Wyświetl nazwiska najgorzej zarabiających pracowników w każdym zespole

```
SELECT nazwisko, etat, placa_pod  
FROM pracownicy  
WHERE (placa_pod, id_zesp) IN  
      (SELECT MIN(placa_pod), id_zesp  
       FROM pracownicy  
       GROUP BY id_zesp);
```

410	10
371	20
208	30
1350	40



Najczęściej spotykane błędy

- Lista atrybutów w klauzuli SELECT podzapytania jest niezgodna z listą atrybutów w warunku:

```
SELECT nazwisko, etat, placa_pod FROM pracownicy
WHERE id_zesp = (
    SELECT nazwisko, id_zesp FROM pracownicy
    WHERE nazwisko='SLOWINSKI' );
ORA-00913: too many values
```

- Podzapytanie zwraca więcej niż jeden wiersz a w warunku użyto operatora przewidzianego do porównywania wartości skalarnych:

```
SELECT nazwisko, etat, placa_pod FROM pracownicy
WHERE placa_pod = (
    SELECT MAX(placa_pod) FROM pracownicy
    GROUP BY id_zesp );
ORA-01427: single-row subquery returns more than one row
```

Podzapytania wyznaczające wiele krotek cd.

- Operator ANY
 - stosowany z operatorami logicznymi, warunek jest prawdziwy jeśli jest spełniony dla jakiegokolwiek wartości zwróconej przez podzapytanie.

```
SELECT nazwisko, placa_pod, etat, id_zesp FROM pracownicy
WHERE placa_pod > ANY ( SELECT DISTINCT placa_pod
    FROM pracownicy WHERE id_zesp = 30 );
```

- Operator ALL
 - stosowany z operatorami logicznymi, warunek jest prawdziwy jeśli jest spełniony dla wszystkich wartości zwróconych przez podzapytanie.

```
SELECT nazwisko, zatrudniony, etat FROM pracownicy
WHERE zatrudniony <= ALL ( SELECT DISTINCT zatrudniony
    FROM pracownicy WHERE etat = 'PROFESOR' );
```

Podzapytania w klauzuli HAVING

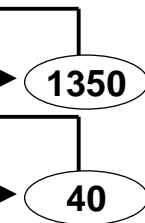
- Wyświetl te zespoły, w których średnia płaca podstawowa jest większa niż średnia płaca w całym instytucie.

```
SELECT z.nazwa, AVG(p.placa_pod) AS srednia
FROM pracownicy p, zespoły z
WHERE p.id_zesp = z.id_zesp
GROUP BY z.nazwa
HAVING AVG(p.placa_pod) >
    ( SELECT AVG(placa_pod)
    FROM pracownicy );
```

Wielopoziomowe zagnieżdżanie zapytań

- Wyświetlić nazwiska i płace pracowników, zarabiających więcej niż wynosi maksymalna płaca w zespole o nazwie ALGORYTMY

```
SELECT nazwisko, placa_pod
FROM pracownicy
WHERE placa_pod >
    ( SELECT MAX (placa_pod)
    FROM pracownicy
    WHERE id_zesp =
        ( SELECT id_zesp
        FROM zespoły
        WHERE nazwa = 'ALGORYTMY' ));
```



Reguły zagnieżdżania podzapytań

- W podzapytaniu nie używamy klauzuli ORDER BY, klauzula ORDER BY może wystąpić wyłącznie jako ostatnia klauzula najbardziej zewnętrznego zapytania.
- Liczba oraz typy atrybutów występujących w klauzuli SELECT podzapytania musi być zgodna z liczbą i typem atrybutów użytych w warunku zapytania zewnętrznego.
- Podzapytania są zawsze wykonywane w kolejności od najgłębiej zagnieżdżonego do najbardziej zewnętrznego.
- Podzapytania mogą się znaleźć w dowolnym miejscu w klauzuli WHERE.

```
SELECT * FROM pracownicy
WHERE ( SELECT MIN(placa_pod) FROM pracownicy ) = placa_pod;
```

```
SELECT * FROM pracownicy
WHERE ( SELECT MAX(placa_pod) FROM pracownicy
        WHERE etat = 'PROFESOR' ) * 0.5 <= placa_pod;
```

(c) Instytut Informatyki Politechniki Poznańskiej

9

Podzapytanie skorelowane (1)

Cechy

- Podzapytanie skorelowane jest wykonywane dla każdej krotki przeglądanej przez zapytanie zewnętrzne
- Podzapytanie skorelowane operuje na wartościach atrybutów przekazanych przez zapytanie zewnętrzne
- Podzapytanie skorelowane zawsze posiada odwołanie do atrybutu zapytania zewnętrznego

```
SELECT atrybut_a, atrybut_b, ...
FROM relacja
WHERE atrybut_n >=
  ( SELECT atrybut_j
    FROM relacja
    WHERE atrybut_j = atrybut_b );
```

zapytanie nadrzędne

podzapytanie skorelowane

(c) Instytut Informatyki Politechniki Poznańskiej

10

Podzapytanie skorelowane (2)

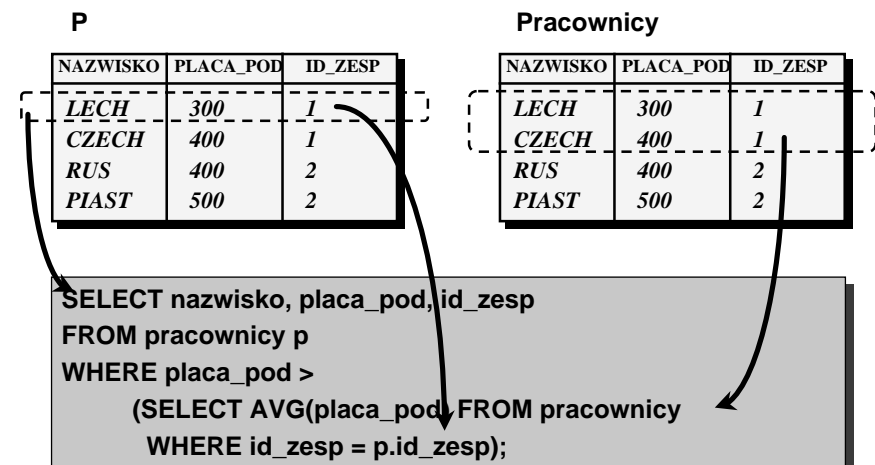
- Polecenie SELECT z podzapytaniem skorelowanym wykonywane jest następująco:
 - pobranie wiersza W_z przez zapytanie zewnętrzne
 - wykonanie zapytania wewnętrznego na podstawie wartości z wiersza W_z
 - zaakceptowanie bądź odrzucenie wiersza W_z
 - pobranie kolejnego wiersza W_{z+1} przez zapytanie zewnętrzne i powtórzenie kroków 2-4
- Przykład: Wyświetl nazwiska pracowników zarabiających powyżej średniej dla swojego zespołu.

```
SELECT p.nazwisko, p.placa_pod
FROM pracownicy p
WHERE p.placa_pod > (
  SELECT AVG(placa_pod) FROM pracownicy
  WHERE id_zesp = p.id_zesp );
```

(c) Instytut Informatyki Politechniki Poznańskiej

11

Podzapytanie skorelowane (3)



(c) Instytut Informatyki Politechniki Poznańskiej

12

Operator EXISTS

Operatory EXISTS, NOT EXISTS

- Operator zwraca wartość TRUE jeżeli podzapytanie zwraca jakąkolwiek wartość. Podzapytanie nie musi zwracać wartości z bazy danych, równie dobrze może zwracać dowolny literał.

```
SELECT id_prac, nazwisko, etat, id_zesp
FROM pracownicy p
WHERE EXISTS ( SELECT id_prac FROM pracownicy
               WHERE id_szefa = p.id_prac );
```

```
SELECT nazwisko, etat, id_zesp
FROM pracownicy p
WHERE NOT EXISTS ( SELECT 1 FROM zespoly
                  WHERE id_zesp = p.id_zesp );
```

Podzapytania w klauzuli SELECT

Zapytanie które zwraca dokładnie jedną wartość jest poprawnym wyrażeniem i może być wykorzystane wszędzie tam, gdzie SQL oczekuje na wyrażenie, np. w klauzuli SELECT

```
SELECT nazwa,
       ( SELECT MAX(placa_pod)
         FROM pracownicy
         WHERE id_zesp = z.id_zesp ) AS max_placa
FROM zespoly z;
```

```
SELECT p.nazwisko,
       ( SELECT nazwisko
         FROM pracownicy
         WHERE id_prac = p.id_szefa ) AS szef
FROM pracownicy p
ORDER BY nazwisko;
```

Podzapytania w klauzuli FROM

Wynik podzapytania może być wykorzystany jako wejściowy zbiór danych dla innego zapytania. Podzapytanie może się znaleźć w klauzuli FROM, zamiast nazwy tabeli.

```
SELECT nazwisko, nazwa, pozycja
FROM
  ( SELECT p.nazwisko, z.nazwa,
    p.etat AS pozycja
    FROM pracownicy p NATURAL JOIN zespoly z
    WHERE p.placa_pod > 800
    ORDER BY p.nazwisko );
```

Klauzula WITH

Poszczególnym podzapytaniom można nadawać nazwy w celu uproszczenia składni zapytania. Służy do tego klauzula WITH

```
WITH prac_zesp AS
  ( SELECT nazwa, nazwisko, etat, placa_pod
    FROM pracownicy JOIN zespoly USING (id_zesp) )
SELECT * FROM prac_zesp
WHERE placa_pod > 1200;
```

```
WITH profesorowie AS
  ( SELECT * FROM pracownicy WHERE etat = 'PROFESOR' ),
asystenci AS
  ( SELECT * FROM pracownicy WHERE etat = 'ASYSTENT' )
SELECT * FROM profesorowie
WHERE EXISTS
  ( SELECT * FROM asystenci
    WHERE id_szefa = profesorowie.id_prac );
```

Zapytania hierarchiczne

- Zapytania hierarchiczne pozwalają na rekurencję w relacjach zawierających dane hierarchiczne
- Pseudokolumna LEVEL reprezentuje poziom rekurencji w drzewie hierarchii
- Operator PRIOR służy do odwoływania się do rodzica danego węzła
- Klauzula START WITH definiuje korzeń drzewa

```
SELECT id_prac, id_szefa, nazwisko, LEVEL  
FROM pracownicy  
CONNECT BY PRIOR id_prac = id_szefa  
START WITH nazwisko = 'WEGLARZ';
```

```
SELECT id_prac, id_szefa, nazwisko, LEVEL  
FROM pracownicy  
CONNECT BY PRIOR id_prac = id_szefa  
START WITH etat = 'PROFESOR';
```