7dt2i6b

BAZY DANYCH II

Wprowadzenie do SQL

mgr inż. Patryk Kaczmarek

patryk.kaczmarek@ansleszno.pl

Plan laboratorium

- Iloczyn kartezjański
- Połączenie równościowe
- Połączenie nierównościowe
- Połączenie zwrotne
- Operatory zbiorowe
- Podzapytania
- Podzapytania w HAVING
- Wielopoziomowe zagnieżdżanie zapytań
- Podzapytanie skorelowane

Iloczyn kartezjański

Iloczynem kartezjańskim dwóch relacji nazywamy zbiór wszystkich możliwych kombinacji krotek z obu relacji. Rzadko bywa przydatny.

```
SELECT nazwisko, etat, nazwa FROM pracownicy, etat;
```

Połączenie równościowe

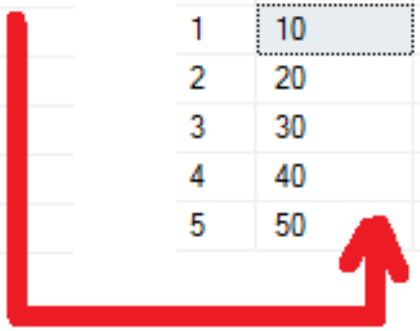
- Połączenie relacji, w którym uzyskujemy krotki spełniające warunki połączenia nazywamy połączeniem wewnętrznym (inner join) – używane w MS SQL
- Połączenie równościowe, w którym warunek połączenia dotyczy atrybutów o tej samej nazwie nazywamy (natural join) – brak w MS SQL

PRACOWNICY

	id_prac	nazwisko	ID_ZESP
1	100	Marecki	10
2	110	Janicki	40
3	120	Nowicki	30
4	130	Nowak	20
5	140	Kowalski	20

ZESPOŁY

	ID_ZESP	Nazwa
1	10	ADMINISTRACJA
2	20	SYSTEMY ROZPROSZONE
3	30	SYSTEMY EKSPERCKIE
4	40	ALGORYTMY
5	50	BADANIA OPERACYJNE



Połączenie równościowe

- **Warunkiem połączenia** nazywamy warunek porównujący ze sobą wartości z dwóch różnych relacji
- Jeżeli w łączonych relacjach istnieją takie same nazwy atrybutów, to muszą być poprzedzone nazwą relacji w celu uniknięcia pomyłki
- W klauzuli from można wprowadzić Aliasy. Jeśli alias został użyty nie można już używać nazwy relacji
- Jeżeli łączymy N relacji to trzeba podać N-1 warunków

```
SELECT p.nazwisko, z.id_zesp FROM pracownicy p,
zespoly z WHERE p.id_zesp = z.id_zesp;
```

Połączenie nierównościowe

W połączeniu tabel jako operator można stosować dowolny operator języka SQL

PRACOWNICY

	id_prac	Nazwisko	placa_pod
1	100	Marecki	4730.00
2	110	Janicki	3350.00
3	120	Nowicki	3070.00
4	130	Nowak	3960.00
5	140	Kowalski	3230.00

ETATY

	nazwa	PLACA_OD	PLACA_DO
1	ADIUNKT	2510.00	3000.00
2	ASYSTENT	1500.00	2100.00
3	DOKTORANT	800.00	1000.00
4	DYREKTOR	4280.00	5100.00
5	PROFESOR	3000.00	4000.00

```
SELECT nazwisko, nazwa, placaPOD, placaMIN, placaMAX
FROM pracownicy JOIN etaty
ON placaPOD BETWEEN placaMIN AND placaMAX;
```

Połączenia

Operatory MS SQL

- **JOIN = INNER JOIN** – wyświetli wszystkie pasujące wiersze
- **LEFT JOIN** – wyświetli wszystkie wiersze z lewej tabeli, nawet te niedopasowane
- **RIGHT JOIN** – jak wyżej tylko tabela z prawej
- **FULL JOIN** – wszystkie wiersze z lewej i prawej

```
SELECT p.nazwisko, z.id_zesp FROM pracownicy p
JOIN zespoly z  ON p.id_zesp = z.id_zesp;
```

Połączenie zwrotne

W tym połączeniu użycie aliasów jest obowiązkowe.


```
SELECT p.nazwisko, s.nazwisko FROM pracownicy p JOIN pracownicy s ON p.id_szefa = s.id_prac;
```

Operatory zbiorowe

- **UNION** – pozwala na połączenie dwóch tabel poprzez ich „sklejenie” . Wymaga by dwa zapytania miały tę samą liczbę kolumn oraz typy danych. (SUMA ZBIORÓW)

```
SELECT nazwisko FROM pracownicy UNION
SELECT nazwa FROM zespoly;
```

- **INTERSECT** – część wspólna zbiorów, eliminuje duplikaty
- **EXCEPT** – różnica zbiorów, eliminuje duplikaty

Operatory zbiorowe często nazywamy operatorami złączeń pionowych, ponieważ łączą kolumny zbiorów wynikowych

Operatory zbiorowe – reguły stosowania

- W łączonych operatorami zbiorowymi klauzulach SELECT musi wystąpić ta sama liczba atrybutów.
- Typy odpowiednich atrybutów różnych klauzul SELECT muszą być zgodne.
- W wyniku zapytania pojawiają się nazwy atrybutów wyłącznie z pierwszej klauzuli SELECT.
- Klauzula ORDER BY może być użyta tylko jako ostatnia klauzula zapytania.
- Polecenia SELECT są wykonywane w kolejności ich wystąpienia (od góry do dołu), nawiasy umożliwiają zmianę kolejności).

Podzapytanie

Podzapytanie to polecenie SELECT zagnieżdżone w innym poleceniu SELECT.

W podzapytaniu nie może wystąpić klauzula ORDER BY.

Operatory:

= <> != <= >=

Ogólny model podzapytania

```
SELECT atrybut1, atrybut2 FROM tabela WHERE atrybut1
OPERATOR (SELECT atrybut2 FROM tabela WHERE warunek)
```

Podzapytanie - przykład

Wyświetl pracownika zarabiającego najmniej:

```
SELECT nazwisko FROM pracownicy WHERE placapod  = (SELECT
MIN(placapod) FROM  pracownicy);
```

Podzapytanie HAVING- przykład

Wyświetl zespoły, których średnia płaca jest większa niż średnia na całej uczelni:

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

Wielopoziomowe zapytania - przykład

Wyświetl nazwiska i płace zarabiających więcej niż maksymalna płaca w zespole Algorytmy:

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

Kwantyfikatory ALL i SOME(ANY)

W odniesieniu do podzapytania zwracającego listę wartości możemy użyć:

- **ALL –** realizuje operację „od każdej wartości na liście”
- **SOME(ANY)** – „od jakiejkolwiek wartości na liście”

Kwantyfikatory ALL i SOME(ANY)

Przykład:

Znajdź pracownika, który zarabia więcej niż najlepiej zarabiający w zespole 20

```
SELECT nazwisko, placa_pod FROM pracownicy
WHERE placa_pod > ALL
  ( SELECT placa_pod FROM pracownicy WHERE id_zesp = 20);
```

ALL i SOME(ANY) dlaczego wykorzystujemy

Przykład:

Znajdź dział gdzie są najwyższe średnie zarobki

```
SELECT NAZWA FROM ZESPOLY z INNER JOINPRACOWNICY p
ON z.ID_ZESP = p.ID_ZESP GROUP BY NAZWA
HAVING AVG(PLACA_POD) = (SELECT MAX(AVG(PLACA_POD)) FROM
PRACOWNICY GROUP BY ID_ZESP);
```

Cannot perform an aggregate function on an expression containing an aggregate or a subquery.

```
SELECT NAZWA FROM ZESPOLY z INNER JOINPRACOWNICY p
ON z.ID_ZESP = p.ID_ZESP GROUP BY NAZWA
HAVING AVG(PLACA_POD) >= ALL (SELECT AVG(PLACA_POD) FROM PRACOWNICY
GROUP BY ID_ZESP);
```

Podzapytanie skorelowane

- 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 atrybuta, atrybutb, ...
FROM relacja
WHERE atrybutn >=
  ( SELECT atrybutj FROM relacja WHERE atrybutj = atrybutb );
```