ZŁĄCZENIA w SQL

Autor: Joanna Karwowska

- Jeśli pobieramy dane z więcej niż jednej tabeli, w rzeczywistości wykonujemy tak zwane **złączenie**.

- W SQL istnieją instrukcje pozwalające na formalne wykonanie złączenia tabel – istnieje wiele typów złączeń.

I. OPERACJE POZIOME NA ZBIORACH.

1. **Rodzaje złączeń:**
   a) **WEWNĘTRZNE**
   b) **ZEWNĘTRZNE**
   c) **KRZYŻOWE**

2. Złączenie wewnętrzne – INNER JOIN

Złączenie typu INNER JOIN – to złączenie warunkowe o postaci:

SELECT *kolumna1, kolumna2, …, kolumnaN*
FROM *tabela1* **[INNER] JOIN** *tabela2*
ON *wyrażenie_warunkowe*;

W przypadku tego złączenia w wyniku pojawią się tylko te wiersze z tabeli1, które mają swoje odpowiedniki w tabeli2 (które spełniają warunki wymienione po klauzuli ON).

Słowo INNER w większości implementacji jest opcjonalne.

Jest to więc odpowiednik instrukcji:

SELECT *kolumna1, kolumna2, …, kolumnaN*
FROM *tabela1, tabela2*
WHERE *warunki*;

3. Złączenie zewnętrzne – LEFT OUTER JOIN

- Złączenie typu **LEFT OUTER JOIN** – zapytanie zwraca wszystkie wiersze z pierwszej tabeli i pasujące wiersze z drugiej tabeli.
- Konstrukcja ta ma postać:

  SELECT ***kolumna1, kolumna2, …, kolumnaN***
  FROM ***tabela1*** **LEFT [OUTER] JOIN** ***tabela2***
  ON ***wyrażenie_warunkowe***;

- Słowo OUTER w większości implementacji jest opcjonalne.

4. Złączenie zewnętrzne – RIGHT OUTER JOIN

- **Złączenie typu RIGHT OUTER JOIN** – jest przeciwieństwem LEFT OUTER JOIN.
- **Zapytanie zwraca wszystkie wiersze z drugiej tabeli i pasujące wiersze z pierwszej tabeli.**
- Ma ono postać:

  SELECT *kolumna1, kolumna2, …, kolumnaN*
  FROM *tabela1* RIGHT [OUTER] JOIN *tabela2*
  ON *wyrażenie_warunkowe*;

Złączenie typu FULL OUTER JOIN

- **Złączenie typu FULL OUTER JOIN** – jest kombinacją LEFT OUTER JOIN i RIGHT OUTER JOIN. Nie jest ono obsługiwane przez bazę MYSQL.
- Ma ogólną postać:

  SELECT *kolumna1, kolumna2, …, kolumnaN*

  FROM *tabela1* FULL [OUTER] JOIN *tabela2*

  ON *wyrażenie_warunkowe*;
- Uwzględnia ono w wynikach zapytania zarówno takie wiersze z pierwszej tabeli, które nie mają swoich odpowiedników w drugiej, jak i takie wiersze z drugiej tabeli, które nie mają swoich odpowiedników w pierwszej.

5. Złączenie krzyżowe – CROSS JOIN

Złączenie typu CROSS JOIN – to złączenie krzyżowe, którego ogólna postać jest następująca:

SELECT *kolumna1, kolumna2, …, kolumnaN*
FROM *tabela1* **CROSS JOIN** *tabela2*;

Wykonuje ono iloczyn kartezjański łączonych tabel, czyli łączy każdy wiersz (krotkę) *tabeli1* z każdym wierszem (krotką) t*abeli2*. Jest to zatem odpowiednik instrukcji:

SELECT *kolumna1, kolumna2, …, kolumnaN*
FROM *tabela1, tabela2;*

6. Złączenia i klauzula WHERE

- W złączeniach można również stosować klauzulę WHERE.
- W wyniku takiego zapytania najpierw zostanie wykonane złączenie spełniające warunki występujące po klauzuli ON.
- Następnie z otrzymanych wierszy zostaną wyeliminowane te, które nie spełniają warunków występujących po klauzuli WHERE, a serwer bazy danych zwróci ostateczny wynik.

Przykład

Napisz zapytanie wykorzystujące złączenie **LEFT OUTER JOIN** do wyświetlenia danych pracowników, dla których w kolumnie *stanowisko* znajdują się wartości niemające odpowiedników w tabeli *stanowiska*.

SELECT *imie, nazwisko, pesel, nazwa, pensja*
FROM *pracownicy* LEFT OUTER JOIN *stanowiska*
ON *stanowisko=stanowiska.id_stanowiska*
WHERE nazwa IS NULL;

Podsumowanie

- Omówione wcześniej sposoby łączenia tabel w klauzuli FROM – dotyczyły łączenia zbiorów w sposób poziomy.
- Określane były typy złączeń (m.in. INNER, OUTER JOIN) oraz warunki dopasowania rekordów – wyszczególnione przed operatorem ON.
- W wyniku otrzymywaliśmy zbiór elementów, opisywany za pomocą wszystkich kolumn (atrybutów) łączonych tabel. Dlatego nazywamy to połączenie jako **poziome**.

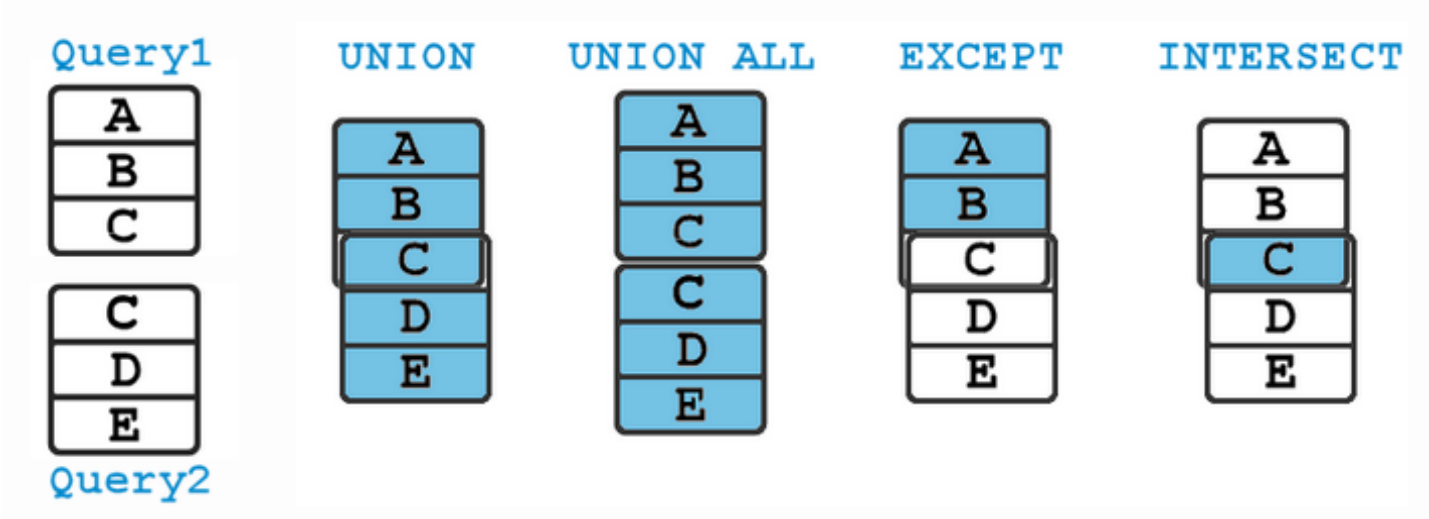
II. OPERACJE PIONOWE NA ZBIORACH.

1. Rodzaje operacji **pionowych**:
   - **UNION** – suma bez duplikatów
   - **UNION ALL** – suma z duplikatami
   - **EXCEPT** – odejmowanie zbiorów
   - **INTERSECT** – iloczyn (część wspólna)

Operacje pionowe – cd.

- Operują one zawsze, na wynikach całych kwerend (tabel wejściowych) i zwracają tabelę wynikową, będącą zbiorem identycznie określonym jak pierwsza tabela wejściowa (liczba i nazwy kolumn).
- Zawierają jednak elementy (wiersze), zgodne z arytmetyką zbiorów, określoną przez operator: UNION, UNION ALL, EXCEPT lub INTERSECT.
- W jednym zapytaniu możemy dokonywać wiele operacji na zbiorach np. łączyć (UNION) wyniki 5 kwerend (tabel wejściowych).

2. Ogólne zasady operacji na zbiorach



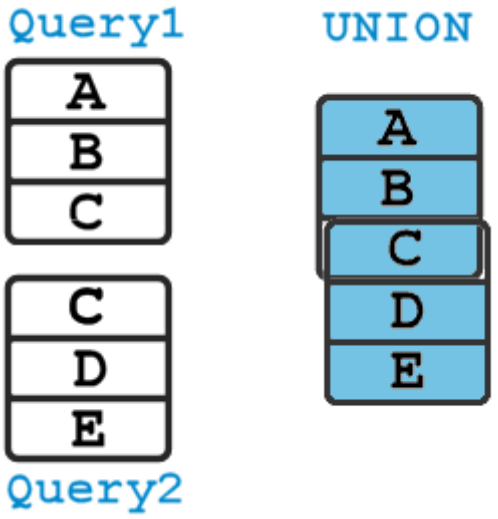
3. Warunki operacji na zbiorach

Jest kilka zasad, które muszą być spełnione:

- **Warunkiem podstawowym**, któregokolwiek ze sposobów operowania na zbiorach w sposób pionowy, **jest podobna struktura tabel wejściowych**.
- **Liczba kolumn w każdym zbiorze (kwerendzie), musi być identyczna oraz typy danych poszczególnych kolumn, muszą do siebie pasować**.
- Nazwy kolumn, nie mają znaczenia.
- W zbiorze wynikowym, atrybuty będą nazwane tak jak w pierwszej z kwerend.

4. Operator UNION

- **UNION** oznacza sumę zbiorów.
- W wyniku otrzymamy elementy znajdujące się zarówno w zbiorze pierwszym jak i drugim, ale domyślnie jest to operacja **UNION DISTINCT**, czyli z usunięciem wszystkich duplikatów.



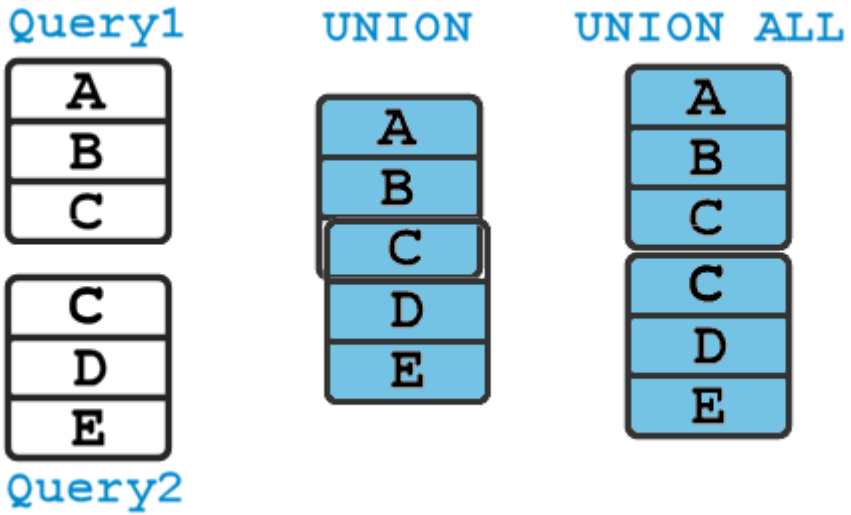
Przykład

Wykonaj zapytanie, które wyświetli dane (imię, nazwisko i PESEL) wszystkich pracowników uczelni UczelniaA i UczelniaB.

SELECT imie, nazwisko pesel FROM UczelniaA
UNION
SELECT imie, nazwisko pesel FROM UczelniaB;

5. Operator UNION ALL

- Drugim sposobem na **dodawanie zbiorów** jest **UNION ALL – czyli łączenie bez usuwania duplikatów.**



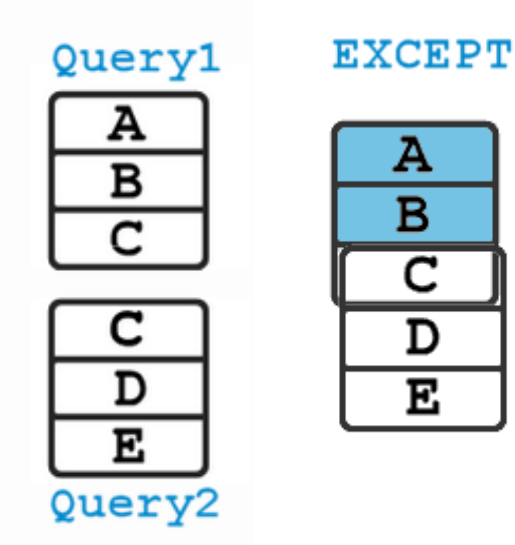
Przykład

Użyj instrukcji **UNION** wyświetlającej dane wszystkich pracowników w taki sposób, aby w wynikach znalazły się wszystkie wiersze zapytań składowych.

```
SELECT imie, nazwisko pesel FROM UczelniaA
UNION ALL
SELECT imie, nazwisko pesel FROM UczelniaB;
```

6. Operator EXCEPT

- Operator **EXCEPT** oznacza odejmowanie zbiorów.
- Ze zbioru pierwszego (czyli po lewej stronie od operatora EXCEPT), odejmowane są wszystkie elementy wspólne ze zbiorem drugim (tabeli wynikowej, kwerendy po prawej stronie).



EXCEPT – przykład

Q1

Results | Messages

	city	DuplikatNo
1	Kirkland	1
2	Redmond	1
3	Seattle	1
4	Seattle	2
5	Tacoma	1

EXCEPT

Q2

Results | Messages

	city	DuplikatNo
1	Albuquerque	1
2	Anchorage	1
3	Boise	1
4	Butte	1
5	Elgin	1
6	Eugene	1
7	Kirkland	1
8	Lander	1
9	Portland	1
10	Portland	2
11	San Francisco	1
12	Seattle	1
13	Walla Walla	1

=

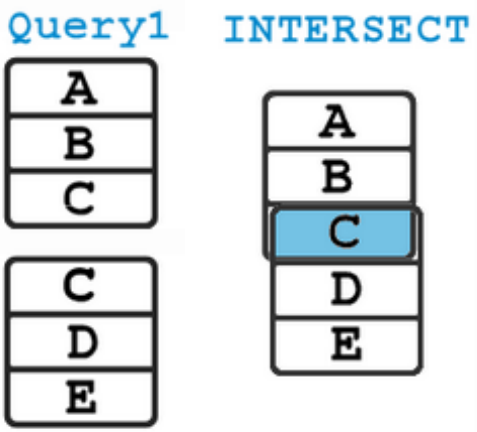
Result

Results | Messages

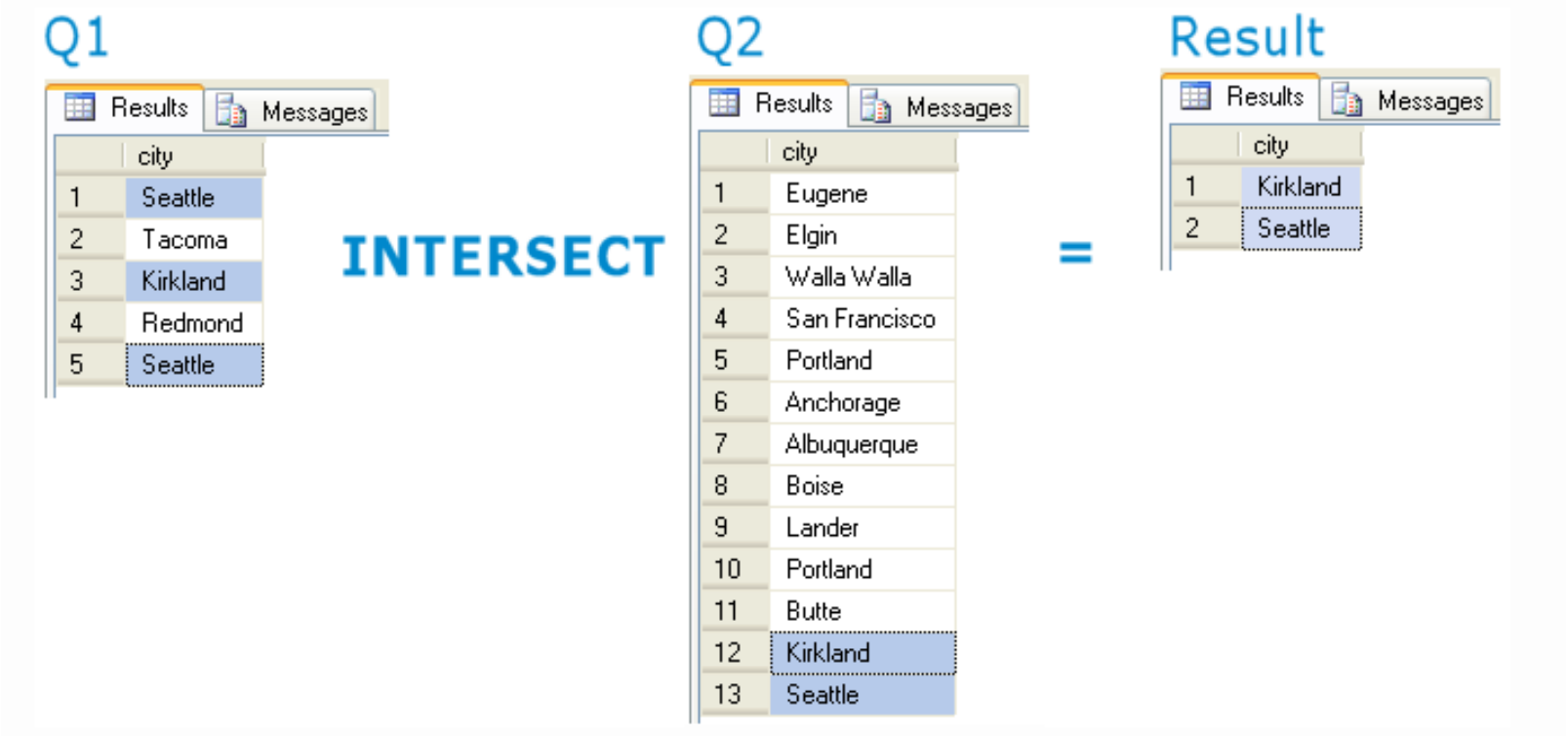
	city	DuplikatNo
1	Redmond	1
2	Seattle	2
3	Tacoma	1

7. Operator INTERSECT

- Operatora **INTERSECT** używamy do wyznaczenia części wspólnej zbiorów.
- Podobnie jak **EXCEPT**, zaimplementowany w SQL Server, został również tylko jako **INTERSECT DISTINCT**, czyli części wspólna dwóch zbiorów z usunięciem duplikatów.



INTERSECT – przykład



8. Kolejność wykonywania operacji

- Możemy operować na wielu zbiorach w ramach jednej kwerendy i stosować różne operacje.
- Obowiązuje tutaj kolejność wykonywania działań – dokładnie tak jak w matematyce.
- Poprzez stosowanie nawiasów, mamy pełną kontrolę nad logiczną kolejnością wykonywania działań.

```
Select koll, kol2, kol3 from tabela1
UNION
Select koll, kol2, kol3 from tabela2
EXCEPT
Select koll, kol2, kol3 from tabela3
INTERSECT
(
Select koll, kol2, kol3 from tabela4
UNION
Select koll, kol2, kol3 from tabela5
)
```

- Najpierw zostaną wykonane działania w nawiasach, czyli UNION ostatnich dwóch kwerend (wyciągających dane z tabeli 4 i 5).
- Następnie mnożenie zbiorów, czyli INTERSECT – część wspólna pomiędzy wynikiem wyznaczonym w kroku pierwszym a kwerendą wyciągającą dane z tabeli3.
- W końcu, jeśli nie ma już żadnych nawiasów i iloczynów, zostaną wykonane wszystkie pozostałe kroki od lewej do prawej, czyli w tym przypadku najpierw, pierwszy UNION i w końcu EXCEPT.


Dank U
Merci
mahalo
Grazie
Thank you
mauruuru
Takk
Gracias
Dziękuję
Děkuju
danke

ZA UWAGĘ!!!