Temat: Podzapytania.

Podzapytania to mechanizm, który pozwala wykorzystać wyniki jednego zapytania w innym zapytaniu. Nazywane są również zapytaniami zagnieżdżonymi. Można je stosować w zapytaniach typu SELECT, INSERT, UPDATE, DELETE, gdzie najczęściej są wprowadzane do klauzuli WHERE bądź FROM. Możemy wyróżnić dwa główne typy podzapytań: proste i skorelowane.

Ćwiczenie P.1

Do ćwiczeń pokazujących działanie tych konstrukcji potrzebne będą tabele opisujące towary, klientów i zamówienia o następujących danych:

klienci

id	imie	nazwisko
1	Jan	Kowalski
2	Andrzej	Nowak
3	Janusz	Malinowski
4	Adam	Kowalski
5	Krzysztof	Nowicki

6

towary

id	nazwa	grupa	cena
1	Śruby	1	2.00
2	Nakrętki	1	3.00
3	Kątowniki	2	8.00
4	Płaskowniki	2	9.00
5	Gwoździe	1	1.00
6	Panele	3	15.00
7	Wkręty	1	4.00
8	Deski	3	12.00
9	Płyty	3	19.00

zamówienia

id	klient_id	towar_id	data	wartosc
1	1	2	2018-01-01	12.44
2	1	4	2018-01-01	14.88
3	1	2	2018-02-12	15.90
4	2	1	2018-01-01	22.35
5	2	1	2018-02-12	28.00
6	2	4	2018-03-01	2.59
7	3	1	2018-02-11	18.00
8	3	4	2018-01-01	12.44
9	4	1	2018-03-11	15.26
10	5	4	2018-03-02	6.25

1. Utwórz nową bazę danych o nazwie nazwisko_hurtownia. Uaktywnij ją (wybierz).
2. Zaprojektuj trzy tabele wzorując się na tabelach powyżej.
   W tabeli zamówienia są dwa klucze obce. Pamiętaj o ich zdefiniowaniu (klauzula REFERENCES).
3. Sprawdź strukturę zaprojektowanych tabel. Wypełnij tabele danymi jak wyżej. Do tabeli klienci dopisz 6-ty rekord wpisując w nim swoje imię i nazwisko.
4. Sprawdź poprawność wprowadzonych danych. Skoryguj ewentualne błędy. Skorzystaj z instrukcji: UPDATE *nazwa_tabeli* SET *kolumnaN=wartoscN* WHERE *warunek.*
5. Wyświetl zawartość trzech tabel i umieść je w edytorze tekstu pod numerem 1 listy numerowanej. W nagłówku dokumentu wpisz swoje imię i nazwisko.
6. Wykonanie następnych ćwiczeń (od P2 do P9) dokumentuj zrzutami ekranu (treść zapytania+wynik).
7. Po wykonaniu wszystkich ćwiczeń prześlij plik nauczycielowi.

Ćwiczenie P.2
Podzapytania w klauzuli FROM.

Wynikiem zapytania typu SELECT jest tablica zawierająca określone kolumny i dane. Skoro jest to tablica, to możliwe jest wykonanie na niej kolejnego zapytania typu SELECT. W ten sposób otrzymamy zapytanie złożone o ogólnej postaci:

```
SELECT kolumny_zapytania
FROM
        (SELECT kolumny_podzapytania
        FROM tablice
        WHERE warunki_podzapytania
        ) AS nazwa
WHERE warunki_zapytania
```

Argument ***nazwa*** jest to nazwa tablicy wynikowej zwróconej przez podzapytanie.
Wykonaj złączenie(relację) tabel ***klienci*** i ***zamówienia*** wiążące klientów oraz ich zamówienia i użyj tego złączenia jako podzapytania zapytania pobierającego dane zamówień o wartości większej niż 15 zł. Wyniki posortuj względem wartości zamówień w porządku rosnącym.

```
SELECT imie, nazwisko, wartosc, data
FROM
        (SELECT imie, nazwisko, wartosc, data
        FROM klienci, zamowienia
        WHERE klienci.id=zamowienia.klient_id
        ) AS klienci_zamowienia
WHERE wartosc>15 ORDER BY wartosc;
```

imie	nazwisko	wartosc	data
Adam	Kowalski	15.26	2011-03-11
Jan	Kowalski	15.90	2011-02-12
Janusz	Malinowski	18.00	2011-02-11
Andrzej	Nowak	22.35	2011-01-01
Andrzej	Nowak	28.00	2011-02-12

Ćwiczenie P.3
Podzapytania w klauzuli WHERE.

Podzapytanie proste to takie podzapytanie, które jest wykonywane raz, a jego wynik jest następnie wykorzystywany w zapytaniu głównym.

Wykonaj zapytanie proste do stwierdzenia, jaki jest identyfikator towaru, na który zastało złożone zamówienie o najniższej wartości.

```
SELECT towar_id
FROM zamowienia
WHERE wartosc=
        (SELECT MIN(wartosc) FROM zamowienia);
```

Działanie tej instrukcji jest następujące: najpierw wykonywane jest podzapytanie:

```
SELECT MIN(wartosc) FROM zamowienia;
```

a jego wynik (2.59) jest wstawiany do warunku klauzuli WHERE. Następnie wykonywane jest zapytanie zewnętrzne, które przyjmie postać:

```
SELECT towar_id
FROM zamowienia
WHERE wartosc=2.59;
```

Efekt działania:

towar_id
4

Ćwiczenie P.4
Podzapytania proste i złączenie tabel.

Wykonaj zapytanie pozwalające stwierdzić, jaka jest nazwa i identyfikator towaru, na który zostało złożone zamówienie o najniższej wartości.

```
SELECT towar_id, nazwa
FROM zamowienia, towary
WHERE wartosc=
        (SELECT MIN(wartosc) FROM zamowienia)
        AND towary.id=zamowienia.towar_id;
```

Efekt działania:

towar_id	nazwa
4	Plaskowniki

Ćwiczenie P.5
Podzapytania zwracające wiele wartości.

Napisz zapytanie zwracające szczegóły zamówień o największej wartości dla każdego z towarów.

```
SELECT zamowienia.id, imie, nazwisko, nazwa, data, wartosc
FROM zamowienia, towary, klienci
WHERE wartosc IN
        (SELECT MAX(wartosc) FROM zamowienia GROUP BY towar_id)
        AND towary.id=zamowienia.towar_id
        AND klienci.id=zamowienia.klient_id
ORDER BY wartosc;
```

Efekt działania:

id	imie	nazwisko	nazwa	data	wartosc
2	Jan	Kowalski	Plaskowniki	2011-01-01	14.88
3	Jan	Kowalski	Nakretki	2011-02-12	15.90
5	Andrzej	Nowak	Sruby	2011-02-12	28.00

Ćwiczenie P.6
Podzapytania skorelowane.

W przedstawionych dotychczas ćwiczeniach podzapytania były wykonywane tylko raz, podczas wywołania zapytania głównego, czyli najpierw było wywoływane podzapytanie, jego wynik był wstawiany do zapytania głównego, a następnie wykonywane było zapytanie główne. Zapytania skorelowane są wykonywane dla każdej wartości analizowanej przez zapytanie główne. Ich cechą charakterystyczną jest odwołanie w zapytaniu skorelowanym do kolumny tabeli występującej w zapytaniu głównym. Jeśli w obu zapytaniach występuje ta sama tabela, niezbędne jest użycie aliasu.

Napisz zapytanie pozwalające stwierdzić, które towary mają cenę wyższą niż średnia cena w grupie, do której należą.
Aby wykonać to ćwiczenie, dla każdego towaru trzeba wyliczyć średnią cenę grupy i porównać ją z ceną z kolumny cena. Nie wystarczy zatem prosta instrukcja warunkowa; trzeba będzie wykorzystać podzapytanie skorelowane. Instrukcja będzie miała postać:

```
SELECT id, nazwa, cena, grupa
FROM towary
WHERE cena >
        (SELECT AVG(cena)
        FROM towary AS towary2
        WHERE towary.grupa=towary2.grupa)
ORDER BY cena;
```

Dla każdego towaru analizowanego w zapytaniu głównym w zapytaniu skorelowanym zastanie wyliczona średnia cena grupy towarów, do której ten towar należy. Ponieważ oba zapytania bazują na tej samej tabeli, w zapytaniu skorelowanym tablica towary została przemianowana na towary2, tak, aby warunek w klauzuli WHERE miał sens (inaczej miałby on postać *towary.grupa=towary.grupa* i zapytanie nie mogłoby być poprawnie wykonane). Efekt działania tego zapytania:

id	nazwa	cena	grupa
2	Nakretki	3.00	1
7	Wkrety	4.00	1
4	Plaskowniki	9.00	2
9	Płyty	19.00	3

Ćwiczenie P.7

Podzapytania w instrukcjach aktualizujących dane.

W wykonaniu ćwiczeń ilustrujących to zagadnienie pomoże nam dodatkowa tabela, która będzie przechowywała imiona i nazwiska klientów oraz wartości złożonych przez nich zamówień. Powstanie ona dzięki instrukcji:

```
CREATE TABLE zam_tymczas
(
imie VARCHAR(20),
nazwisko VARCHAR(30)
wartosc DECIMAL(7,2)
);
```

Zaprojektuj tabelę ***zam_tymczas***. Chcielibyśmy teraz wypełnić ja danymi pobieranymi z tabel zaprojektowanych na początku ćwiczenia 4.1. Przygotowanie osobnych instrukcji **INSERT** dla każdego wiersza byłoby z pewnością bardzo czasochłonne, a jeśli w tabelach ***klienci*** i ***zamówienia*** byłoby więcej danych – praktycznie niewykonywane. W takiej sytuacji najlepiej posłużyć się instrukcją typu **INSERT** wykorzystującą odpowiednio przygotowane podzapytanie. Schemat takiego podzapytania to:

```
INSERT INTO tablica(kolumny)
(
 SELECT kolumny_podzapytania
 FROM tablice_podzapytania
 WHERE warunki_podzapytania
);
```

Użyj podzapytania typu **SELECT** do wypełnienia tabeli **zam_tymczas** danymi poszczególnych zamówień (imię i nazwisko klienta oraz wartość zamówienia).
Aby wykonać to ćwiczenie należy użyć instrukcji:

```
INSERT INTO zam_tymczas(imie, nazwisko, wartosc)
(
 SELECT imie, nazwisko, wartosc
 FROM klienci, zamowienia
 WHERE klienci.id=zamowienia.klient_id
);
```

Po jej wykonaniu wyświetl zawartość tabeli **zam_tymczas**. Powinny znaleźć się tam następujące dane:

imie	nazwisko	wartosc
Andrzej	Nowak	2.59
Krzysztof	Nowicki	6.25
Jan	Kowalski	12.44
Janusz	Malinowski	12.44
Jan	Kowalski	14.88
Adam	Kowalski	15.26
Jan	Kowalski	15.90
Janusz	Malinowski	18.00
Andrzej	Nowak	22.35
Andrzej	Nowak	28.00

Ćwiczenie P.8

Grupowanie w podzapytaniach aktualizujących.

W podzapytaniu można użyć grupowania oraz funkcji agregujących, aby umieścić w tabeli **zam_tymczas** imiona i nazwiska klientów wraz z sumaryczną wartością złożonych przez nich zamówień.

Zaprojektuj tabelę **zam_tymczas2** (identyczną jak w poprzednim ćwiczeniu). Napisz instrukcję wypełniającą tabelę **zam_tymczas2** danymi o sumarycznej wartości zamówień dla każdego klienta. Niezbędne będzie użycie funkcji agregującej **SUM**(dla kolumny wartość) oraz klauzuli grupującej **GROUP BY**.

Właściwa instrukcja będzie miała postać:

```
INSERT INTO zam_tymczas2(imie, nazwisko, wartosc)
      (
       SELECT imie, nazwisko, SUM(wartosc) AS wartość
       FROM klienci, zamowienia
       WHERE klienci.id=zamowienia.klient_id
       GROUP BY klient_id, imie, nazwisko
      );
```

Po jej wykonaniu w tabeli **zam_tymczas2** pojawią się następujące wpisy:

imie	nazwisko	wartosc
Krzysztof	Nowicki	6.25
Adam	Kowalski	15.26
Janusz	Malinowski	30.44
Jan	Kowalski	43.22
Andrzej	Nowak	52.94

Ćwiczenie P.9

Podzapytania w instrukcji DELETE.

Chcemy teraz z pierwszej wersji tabeli **zam_tymczas** usunąć wszystkie wpisy, które w kolumnie wartość mają wartość mniejszą niż średnia arytmetyczna ze wszystkich zamówień z tabeli zamówienia.

Napisz instrukcję usuwającą z tabeli **zam_tymczas** powstałej w ćwiczeniu 4.7 takie zamówienia, których wartość jest mniejsza niż średnia wartość zamówienia w tabeli **zamowienia**.

Utwórz najpierw nową tabelę **zam_tymczas3** (taką jak w ćwiczeniu 4.7) i wypełnij ją danymi.

Podzapytanie znajdzie się w klauzuli **WHERE** instrukcji **DELETE**. Całe zapytanie będzie miało postać:

```
DELETE FROM zam_tymczas3
WHERE wartosxc <
(
 SELECT AVG(wartosc)
 FROM zamowienia
);
```

Działanie tej instrukcji jest następujące: najpierw wykonywane jest podzapytanie:

```
SELECT AVG(wartosc)
FROM zamowienia;
```

a jego wynik (**14.81**) jest wstawiany do warunku klauzuli **WHER**E. Następnie wykonywane jest instrukcja **DELETE**, która przyjmie postać:

```
DELETE FROM zam_tymczas3
WHERE wartosc < 14.81
```

W efekcie działania tej instrukcji zostaną usunięte te zamówienia, których wartość jest mniejsza niż 14,81.

```
mysql> SELECT * FROM zam_tymczas3
    -> ORDER by wartosc;
+---------+------------+---------+
| imie    | nazwisko   | wartosc |
+---------+------------+---------+
| Jan     | Kowalski   |   14.88 |
| Adam    | Kowalski   |   15.26 |
| Jan     | Kowalski   |   15.90 |
| Janusz  | Malinowski |   18.00 |
| Andrzej | Nowak      |   22.35 |
| Andrzej | Nowak      |   28.00 |
+---------+------------+---------+
```