II TI

Temat: Zapytania do bazy danych .

(temat do wszystkich lekcji z baz danych)

Jedną ocenę dostaliście na zdalnych lejcach na skypie.

Teraz oczekuje na wasza informację zwrotną.

Do sprawdzenia prześlij na adres : informatuser@inetria.pl do 14 kwietnia 2020r.
podając w tytule klasę, Imię i nazwisko
w dokumencie tekstowym tylko txt zapytania do bazy danych dane2.sql

do bazy danych dane2.sql wykonaj następujące zapytania:

Zapytanie 1:
wyświetl tytuł, rok ocena z tabeli filmy dla dramatu i horroru;

Zapytanie 2:
Wyświetl imię i nazwisko reżysera dla każdego z filmu (INNER JOIN)

Zapytanie 3:

Wyświetl gatunki,rezysera i tytuł z tabeli filmy dla roku 2017 i oceny 6

Zapytanie 4:

UPDATE rezyserzy SET imie="Francis Ford" WHERE id=8;

napisz co cen skrypt ealizauje

Zapytanie 5:
dodaj do tabeli filmy rekord z twoim ulubionym filmem
(do txt skopiuj zapytanie)

treści na zielono przepisujemy do zeszytu i foto również nadsyłamy w tylko dokumencie PDF na maila informatuser@interia.pl do 14 kwietnia 2020 r. w temacie klasa i nazwisko

Zapytania stanowią centralną część języka DML (and. *Data Manipulation Language*) wchodzącego w skład *SQL*, a służącego do manipulowania danymi. Niniejszy dokument zawiera podstawowe informacje na temat zapytań języka *SQL*.

Ogólna postać zapytania

Ogólna postać zapytania *SQL* wygląda następująco:

```
SELECT kolumny
FROM tabele
WHERE warunek-selekcji
GROUP BY kolumny-grupowane
HAVING warunek-dotyczący-pogrupowanych-danych
 ORDER BY kolumny-klucze-sortowania
;
```

Parametry SELECT

Po słowie kluczowym `SELECT` umieszcza się listę kolumn rozdzielonych przecinkami, która ma się znaleźć w odpowiedzi. Elementy listy mogą mieć następującą postać:

- wszystkie

  ```
  SELECT * FROM ...
  ```

- nazwy kolumn

  ```
  SELECT kolumna1, kolumna2...
  ```

- nazwy kolumn z nazwami tabel

  ```
  SELECT tabela1.kolumna1, tabela2.kolumna1...
  ```

- definicje kolumn wyliczonych

  ```
  SELECT kolumna1 + kolumna2...
  ```

- wywołania funkcji operujących na wartościach pół rekordów

  ```
  SELECT Max( kolumna1 ), Count( tabela1.kolumna2 )...
  ```

W klauzuli `SELECT` można stosować tzw. aliasy, czyli przypisywać kolumnom inne nagłówki, np.:

```
SELECT tabela1.kolumna3 AS kolumna5, kolumna1 +
tabela2.kolumna2 as SumaKolumn...
```

Po słowie kluczowym `SELECT` można podać słowo kluczowe `DISTINCT`. Spowoduje ono, że w odpowiedzi znajdą się tylko unikalne wiersze, np.:

```
SELECT DISTINCT tabela1.kolumna1, kolumna2...
```

Parametry FROM

Po słowie kluczowym `FROM` umieszcza się listę nazw tabel (rozdzieloną przecinkami), z których pochodzą kolumny, np.:

```
SELECT kolumna1, kolumna2
FROM tabela1
;
```

Lista nazw kolumn może być uzyskana z podzapytania.

Parametry WHERE

Klauzula `WHERE` służy do specyfikowania warunku, jaki muszą spełniać rekordy. Przy tworzeniu warunków można korzystać z rozmaitych konstrukcji, oto niektóre z nich:

- porównania (=, <>, <, <=, >, >=)
- operatory logiczne (`AND`, `OR`, `NOT`)
- przynależność do zbioru (`IN`), np:

  ```
  ...WHERE kolumna1 IN (war1, war2, war3)...
  ```

- przynależność do przedziału, np:

  ```
  ...WHERE kolumna1 BETWEEN dolnaWartosc AND gornaWartosc...
  ```

- zgodność z wzorcem tekstowym (`LIKE`), np:

  ```
  ...WHERE kolumna1 LIKE '%awa'...
  ...WHERE kolumna1 LIKE '_awa'...
  ```

  (znak % oznacza dowolny ciąg znaków, znak _ dowolny znak)

- sprawdzenie, czy pole ma wartość (nie)pustą (`IS (NOT) NULL`), np.:

  ```
  ...WHERE kolumna1 IS NOT NULL...
  ```

Parametry GROUP BY

Klauzula `GROUP BY` służy do grupowania (agregacji) wierszy o takiej samej wartości określonych pól. Działanie grupowania wymaga zastosowania funkcji agregujących, takich jak `MIN`, `MAX`, `AVG`, `COUNT`.

Ogólne zasady grupowania wyglądają następująco:

- kolumny zapytania dzieli się na te, według których się grupuje (zbiór G) i na pozostałe (zbiór A)
- po klauzuli `SELECT` MOGĄ wystąpić odwołania do kolumn ze zbioru G (i z reguły występują)
- po klauzuli `SELECT` odwołania do kolumn ze zbioru A MUSZĄ być poddane działaniu funkcji agregujących

Na przykład w zapytaniu:

```
SELECT imie, AVG(wiek), COUNT(*)
     FROM Tabela
     GROUP BY imie
     ;
```

Grupowanie odbywa się po kolumnie `imie`, natomiast pozostałe kolumny są poddane działaniu funkcji agregujących. Odpowiedź na to zapytanie będzie zawierała zbiór rekordów zawierających imię, średni wiek dla osoby o danym imieniu i liczbę osób posiadających dane imię, czyli mówiąc krótko pewne statystyki dotyczące elementu po którym następuje grupowanie (po imieniu).

Parametry HAVING

Klauzula HAVING służy do ograniczania liczebności grup utworzonych w wyniku agregacji. Przykładowo, nawiązując do ostatniego przykładu, zapytanie:

```
SELECT imie, AVG(wiek), COUNT(*)

    FROM Tabela
    GROUP BY imie
    HAVING AVG(wiek) >= 18
    ;
```

ograniczy rekordy opisujące statystyki imienia tylko do tych osób, które są pełnoletnie.

Po HAVING można stosować takie same konstrukcje jak po WHERE, a dodatkowo również funkcje agregujące.

Parametry ORDER BY

Po słowie kluczowym ORDER BY umieszcza się listę nazw kolumn i/lub wartości wyliczonych (rozdzieloną przecinkami), będących kluczem sortowania wraz z opcjonalnymi oznaczeniami dotyczącymi rodzaju sortowania rekordów. Oto ogólna postać:

```
...ORDER BY ([tabela.]kolumna [ASC|DESC])*...
```

Poniżej znajduje się przykład zastosowania sortowania.

```
    ...ORDER BY tabela1.kolumna1, kolumna2 ASC,
    kolumna3 DESC...
```

Powyższy przykład sortuje wyniki po kolumnie tabela1.kolumna1 rosnąco (to kierunek domyślny), następnie po kolumnie kolumna2 również rosnąco i w końcu po kolumnie kolumna3 malejąco.

Składanie wyników zapytań

Wyniki zapytań (listy wierszy) można ze sobą składać, korzystając z operatorów teoriomnogościowych sumy zbiorów (UNION), przecięcia zbiorów (INTERSECT) i różnicy zbiorów (EXCEPT). Po każdym z powyższych słów kluczowych można dodać klauzulę ALL, które wyłączy filtr powtarzających się wierszy. Zapytania można składać pod warunkiem, że typy odpowiadających kolumn są identyczne. Złożenie to wygląda następująco:

```
SELECT...
```

```
UNION ALL

SELECT...
```

Kolejność wykonywania klauzul zapytania

Zapytanie jest wykonywane w następujących krokach:

- wybór kolumn z tabel (`SELECT`, `FROM`)
- zastosowanie warunku ograniczającego (`WHERE`)
- grupowanie (`GROUPBY`)
- zastosowanie warunku ograniczającego liczebność grup (`HAVING`)
- sortowanie (`ORDER BY`)

Podzapytania

W zapytaniu w warunkach klauzul `WHERE` i `HAVING` można stosować podzapytania służąca do wyznaczenia wartości użytej w warunku, np.

```
SELECT kolumna1, kolumna2
 FROM Tabela1
 WHERE kolumna1 > (
SELECT AVG(kolumna3)
FROM Tabela2
);
```

W powyższym przykładzie wybrane są te rekordy dla których wartość pola `kolumna1` jest większa od średniej wartości wyznaczonej z innej tabeli. Należy zwrócić uwagę, że w tym przypadku wynikiem podzapytania jest pojedyncza liczba.

Wykorzystanie IN

Słowo kluczowe `IN` pozwala na zidentyfikowanie wszystkich elementów w zbiorze A, które występują lub nie występują w zbiorze B, np.:

```
 SELECT kolumna1, kolumna2
FROM Tabela
 WHERE kolumna1 NOT IN (
SELECT kolumna2
```

```
FROM Tabela2);
```

Powyższe zapytanie wybiera te rekordy, dla których wartość wybranego pola nie znajduje się w zbiorze określonym przez podzapytanie. Należy zwrócić uwagę, że typ pola `kolumna1` oraz wyników zwracanych przez podzapytanie musi być taki sam. Ponadto wynik podzapytania musi być jednokolumnowy.

Wykorzystanie ALL

Słowo kluczowe `ALL` specyfikuje warunek specyfikujący, że dana wartość musi być w określonej relacji ze wszystkimi wynikami podzapytania, np.:

```
SELECT *
FROM Tabela1
WHERE kolumna1 < ALL (
SELECT MIN(kolumna2)
FROM Tabela2
WHERE kolumna2 > 15 );
```

Powyższe zapytanie wyświetla wszystkie rekordy, dla których wartość pola `kolumna1` jest mniejsza od każdej wartości zwróconej w odpowiedzi przez podzapytanie.

Wykorzystanie ANY

Słowo kluczowe ANY specyfikuje warunek specyfikujący, że dana wartość musi być w określonej relacji z dowolnym z wyników podzapytania, np.:

```
SELECT kolumna1, AVG(kolumna2)
FROM Tabela1
GROUP BY kolumna1
HAVING AVG(kolumna2) < ANY (
SELECT MIN(kolumna2)
FROM Tabela2
WHERE kolumna2 > 15
);
```

Powyższe zapytanie wyświetla wszystkie rekordy, dla których wartość pola `kolumna1` jest mniejsza od dowolnej z wartości zwróconych w odpowiedzi przez podzapytanie.

Złączenia tabel

Złączenie tabel w zapytaniu polega na umieszczeniu w jego wyniku kolumn pochodzących z różnych tabel. Możliwe jest wyświetlenie danych z różnych tabel, które są wzajemnie powiązane

poprzez więzy relacji – w tym celu należy określić warunek łączący. Można tego dokonać na kilka sposobów np.:

```
SELECT Tabela1.kolumna1, kolumna2
FROM Tabela1, Tabela2
 WHERE Tabela1.id = Tabela2.ref_tabela1;
```

W powyższym zapytaniu warunek złączenia jest podany w klauzuli `WHERE` (porównanie wartości pól połączonych relacją).

Inne, równoznaczne formy to:

```
SELECT Tabela1.kolumna1, kolumna2
FROM Tabela1
JOIN Tabela2 ON Tabela1.id = Tabela2.ref_tabela1;
```