Podstawy baz danych

Baza danych Access - zadania

Opracował: Andrzej Nowak

Zadanie_4_uczniowie_oceny_przedmioty

Szkoła dysponuje danymi zawartymi w trzech plikach: **`uczniowie.txt`**, **`oceny.txt`**, **`przedmioty.txt`**.

- Plik **`uczniowie.txt`** zawiera następujące dane o uczniach:

  id_ucznia, nazwisko, imie, ulica, dom, id_klasy.

- Plik **`oceny.txt`** zawiera dane o ocenach:

  id_ucznia, ocena, data, id_przedmiotu.

- Plik **`przedmioty.txt`** zawiera dane o przedmiotach:

  id_przedmiotu, nazwa_przedmiotu, nazwisko_naucz, imie_naucz.

Korzystając z danych zawartych w plikach **`uczniowie.txt`**, **`oceny.txt`**, **`przedmioty.txt`** oraz z dostępnych narzędzi informatycznych wykonaj poniższe polecenia.

a. Poza rejonem szkoły leżą ulice Worcella oraz Sportowa. Podaj, ilu uczniów mieszka poza rejonem szkoły (czyli na jednej z tych dwóch ulic).
b. Wypisz wszystkie oceny ucznia Jana Augustyniaka z języka polskiego.
c. Oblicz, ile dziewcząt i ilu chłopców jest w poszczególnych klasach. Wynik przedstaw w postaci zestawienia: **id_klasy**, liczba dziewcząt, liczba chłopców. Załóż, że imiona dziewcząt (i tylko dziewcząt) kończą się na literę a.
d. Utwórz zestawienie dla klasy 2a zawierające nazwy przedmiotów i średnie ocen klasy z tych przedmiotów (średnie podaj z zaokrągleniem do dwóch miejsc po przecinku). Zestawienie posortuj nierosnąco według średnich ocen.
e. Utwórz zestawienie uporządkowane alfabetycznie według nazwisk zawierające wykaz osób z klasy 2c, które w kwietniu 2009 roku otrzymały oceny niedostateczne (imię, nazwisko, przedmiot).
f. Podaj nazwisko, imię, klasę oraz średnią ocen osoby, która osiągnęła najwyższą średnią ocen w całej szkole (jest tylko jedna taka osoba).

Rozwiązanie

Na podstawie opisu umieszczonego w zadaniu, tworzymy trzy tabele:

uczniowie					
id_ucznia	**nazwisko**	**imie**	**ulica**	**dom**	**id_klasy**
INT	VARCHAR(20)	VARCHAR(15)	VARCHAR(20)	INT	CHAR(2)

oceny			
id_ucznia	**ocena**	**data**	**id_przedmiotu**
INT	INT	DATE	INT

przedmioty			
id_przedmiotu	**nazwa_przedmiotu**	**nazwisko_naucz**	**imie_naucz**
INT	VARCHAR(15)	VARCHAR(20)	VARCHAR(15)

Tabele są powiązane relacjami:

oceny.id_ucznia → uczniowie.id_ucznia

oceny.id_przedmiotu → przedmioty.id_przedmiotu

Logujemy się do mysql i wpisujemy:

```
CREATE TABLE uczniowie
(
 id_ucznia INT PRIMARY KEY,
 nazwisko VARCHAR(20) NOT NULL,
 imie VARCHAR(15) NOT NULL,
 ulica VARCHAR(20) NOT NULL,
 dom INT,
 id_klasy CHAR(2) NOT NULL
);
CREATE TABLE oceny
(
 id_ucznia INT NOT NULL,
 ocena INT NOT NULL,
 data DATE NOT NULL,
 id_przedmiotu INT NOT NULL
);
CREATE TABLE przedmioty
(
 id_przedmiotu INT PRIMARY KEY,
 nazwa_przedmiotu VARCHAR(15) NOT NULL,
 nazwisko_naucz VARCHAR(20) NOT NULL,
 imie_naucz VARCHAR(15)
);
```

Pliki **uczniowie.txt**, **oceny.txt**, **przedmioty.txt** kopiujemy do katalogu /tmp i wczytujemy do tabel:

```
LOAD DATA INFILE '/tmp/uczniowie.txt' INTO TABLE uczniowie FIELDS TERMINATED BY ';';
LOAD DATA INFILE '/tmp/oceny.txt' INTO TABLE oceny FIELDS TERMINATED BY ';';
LOAD DATA INFILE '/tmp/przedmioty.txt' INTO TABLE przedmioty FIELDS TERMINATED BY ';';
```

a) liczba uczniów mieszkających na ulicy Worcella i Sportowa

Po prostu zliczamy rekordy z tabeli uczniowie, w których pole ulica jest równe Worcella lub Sportowa:

```
SELECT COUNT(*) AS 'Liczba uczniów spoza rejonu'
FROM uczniowie
WHERE ulica='Worcella' OR ulica='Sportowa';
```

b) oceny ucznia Jana Augustyniaka z języka polskiego

Tworzymy zapytanie z trzech tabel:

```
SELECT GROUP_CONCAT(ocena) AS 'Oceny z polskiego Jana Augustyniaka'
FROM oceny,uczniowie,przedmioty
WHERE oceny.id_przedmiotu=przedmioty.id_przedmiotu AND nazwa_przedmiotu='polski' AND
        oceny.id_ucznia=uczniowie.id_ucznia AND nazwisko='Augustyniak' AND imie='Jan';
```

c) liczba chłopców i dziewcząt w poszczególnych klasach

Tutaj wykorzystujemy funkcję agregującą SUM(), funkcję warunkową IF() oraz grupowanie rekordów wg id_klasy.

Funkcja SUM() zlicza dla każdego rekordu danej klasy to, co jest umieszczone w jej nawiasach. Umieścimy tam funkcję warunkową IF(). Posiada ona trzy argumenty:

IF(warunek,w1,w2). Funkcja wyznacza wartość logiczną warunku. Jeśli otrzyma prawdę, to zwraca jako wynik w1. Inaczej zwraca jako wynik w2.

Tworzymy następujące zapytanie:

```
SELECT
   id_klasy,
   SUM(IF(imie LIKE '%a',1,0)) AS 'Liczba dziewcząt',
   SUM(IF(imie NOT LIKE '%a',1,0)) AS 'Liczba chłopców'
FROM uczniowie
GROUP BY id_klasy;
```

d) zestawienie średnich ocen klasy 2a

Tutaj wykorzystujemy funkcję agregującą AVG(), która oblicza średnią arytmetyczną. Zaokrąglenie do dwóch miejsc po przecinku uzyskamy za pomocą funkcji ROUND(x,2). Rekordy muszą być zgrupowane wg przedmiotów. Wyniki będą sortowane malejąco wg średnich ocen. W zapytaniu musimy połączyć wszystkie trzy tabele, ponieważ informację o ocenie i klasie uzyskujemy poprzez tabelę uczniowie.

```
SELECT nazwa_przedmiotu AS Przedmiot, ROUND(AVG(ocena),2) AS 'Zrednia w 2a'
FROM oceny, uczniowie, przedmioty
WHERE oceny.id_ucznia=uczniowie.id_ucznia AND
          oceny.id_przedmiotu=przedmioty.id_przedmiotu AND
          id_klasy='2a'
GROUP BY nazwa_przedmiotu
ORDER BY AVG(ocena) DESC;
```

e) osoby z 2c, które w kwietniu 2009 otrzymały ocenę ndst

Ponieważ w wynikach zapytania musimy umieścić ucznia oraz przedmiot, w zapytaniu należy użyć wszystkich trzech tabel. Miesiąc odczytamy z pola data za pomocą funkcji MONTH(d), która zwraca numer miesiąca (1 = styczeń, 2 = luty,...). Rok odczytamy za pomocą YEAR(d).

```
SELECT imie,nazwisko,nazwa_przedmiotu
FROM oceny, uczniowie, przedmioty
WHERE oceny.id_ucznia=uczniowie.id_ucznia AND
          oceny.id_przedmiotu=przedmioty.id_przedmiotu AND
           ocena=1 AND
           MONTH(data)=4 AND YEAR(data)=2009 AND id_klasy='2c'
ORDER BY nazwisko;
```

f) nazwisko, imię, klasa oraz średnia ocen osoby, która osiągnęła najwyższą średnią ocen w całej szkole

Rozwiązanie polega na wyświetleniu listy uczniów posortowaną wg rosnących średnich i odczytanie ostatniego rekordu:

```
SELECT nazwisko,imie,id_klasy,ROUND(AVG(ocena),2) AS 'Zrednia'
FROM oceny,uczniowie
WHERE oceny.id_ucznia=uczniowie.id_ucznia
GROUP BY oceny.id_ucznia
ORDER BY AVG(ocena);
```