

Marek Robak

Wprowadzenie do języka SQL na przykładzie baz SQLite

Przykłady najlepiej wykonywać od razu na bazie i eksperymentować z nimi.

Tworzenie tabeli

Pierwsza tabela

W relacyjnych bazach danych jedna baza może składać się z wielu tabel, przechowujących różne informacje. Zazwyczaj informacje te są ze sobą powiązane. Jeśli na przykład projektowałbym bazę danych dla systemu sprzedaży, używanego w supermarketach, zapewne musiałbym stworzyć tabele: produkty, paragony, faktury, a po dokładniejszej analizie zapewne okazałoby się, że byłoby ich jeszcze więcej. Utwórzmy zatem przykładową tabelę, zawierającą informacje o ludziach:

```
CREATE TABLE ludzie (  
  id INTEGER PRIMARY KEY,  
  imie TEXT,  
  nazwisko TEXT,  
  rok_urodzenia INT,  
  plec CHAR(1) DEFAULT 'k',  
  wzrost INT);
```

Polecenie to utworzy nową tabelę "ludzie", zbudowaną z wymienionych pól (kolumn). Po nazwie każdego pola podajemy jego typ, żeby baza danych wiedziała jak traktować te dane (np. podczas sortowania), pozwala to też zapisywać informacje w najbardziej zwięzły sposób, co ma znaczenie, gdy chcemy przechowywać w bazie miliony rekordów danych. Poszczególne systemy zarządzania bazami danych różnią się nieraz co do zastosowanych typów pól (albo co nazw typów), ale w każdym systemie możliwe jest tworzenie pól tekstowych, liczbowych i datowych.

Teraz druga tabela:

```
CREATE TABLE zwierzeta(  
  id INTEGER PRIMARY KEY,  
  imie TEXT,  
  ludzie_id INT);
```

Wprowadzanie, modyfikowanie i usuwanie danych

Wprowadzanie:

```
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES  
('Jan', 'Kowalski', 1980, 'm', 175);
```

Aktualizacja (uwaga - jeśli nie będzie wąskiego warunku, można zmodyfikować dane w dużej części tabeli lub nawet całej tabeli):

```
UPDATE ludzie SET rok_urodzenia = 1981 WHERE id = 91;
```

Usuwanie - tutaj ta sama uwaga

co w przypadku aktualizacji. Pamiętaj, że w SQL-u nie ma Ctrl+z :(

```
DELETE FROM ludzie WHERE id = 30; Teraz możemy wprowadzić trochę danych:
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Jan','Kowalski',1980,'m',175);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Janina','Malinowska',1970,'k',165);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Jacek','Plackowski',1982,'m',181);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Zenobia','Kociubowska',1960,'k',160);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Alfred','Domkowski',1989,'m',176);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Jacek','Malina',1984,'m',189);
INSERT INTO ludzie (imie, nazwisko, rok_urodzenia, plec, wzrost) VALUES
('Alfred','Tomba',1980,'m',176); Jeszcze dane zwierząt:
INSERT INTO zwierzeta (imie, ludzie_id) VALUES ('Azor',3);
INSERT INTO zwierzeta (imie, ludzie_id) VALUES ('Miauczek',5);
INSERT INTO zwierzeta (imie, ludzie_id) VALUES ('As',6);
INSERT INTO zwierzeta (imie, ludzie_id) VALUES ('Kicia',3);
```

Pobieranie danych z jednej tabeli

Pobieranie wszystkiego

```
SELECT * FROM ludzie;
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost
1	Jan	Kowalski	1980	m	175
2	Janina	Malinowska	1970	k	165
3	Jacek	Plackowski	1982	m	181
4	Zenobia	Kociubowsk	1960	k	160
5	Alfred	Domkowski	1989	m	176
6	Jacek	Malina	1984	m	189
7	Alfred	Tomba	1980	m	176

Wybrane kolumny

```
SELECT nazwisko, imie FROM ludzie;
```

```
nazwisko  imie
-----
Kowalski  Jan
Malinowska Janina
Plackowski Jacek
Kociubowsk Zenobia
Domkowski Alfred
Malina     Jacek
```

Tomba Alfred

Proste działania matematyczne

```
SELECT 3 * 7;
```

```
3 * 7
```

```
-----
```

21Dzięki zastosowaniu AS możemy nadać kolumnie wyników nazwę:

```
SELECT 3 * 7 AS wynik;
```

```
wynik
```

```
-----
```

21Możemy też wykorzystać działania matematyczne do obliczenia wieku.

```
SELECT *, 2009 - rok_urodzenia AS wiek FROM ludzie;
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost	wiek
1	Jan	Kowalski	1980	m	175	29
2	Janina	Malinowska	1970	k	165	39
3	Jacek	Plackowski	1982	m	181	27
4	Zenobia	Kociubowski	1960	k	160	49
5	Alfred	Domkowski	1989	m	176	20
6	Jacek	Malina	1984	m	189	25
7	Alfred	Tomba	1980	m	176	29

Warunek, warunek wielokrotny

Sami panowie:

```
SELECT * FROM ludzie WHERE plec = 'm';
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost
1	Jan	Kowalski	1980	m	175
3	Jacek	Plackowski	1982	m	181
5	Alfred	Domkowski	1989	m	176
6	Jacek	Malina	1984	m	189
7	Alfred	Tomba	1980	m	176

Tylko pełnoletnie panie:

```
SELECT * FROM ludzie WHERE plec = 'k' AND 2009 - rok_urodzenia >= 18;
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost
2	Janina	Malinowska	1970	k	165
4	Zenobia	Kociubowski	1960	k	160

Sortowanie kolumn, sortowanie odwrotne, sortowanie wielu kolumn

```
SELECT imie, nazwisko, rok_urodzenia FROM ludzie ORDER BY rok_urodzenia;
```

imie	nazwisko	rok_urodzenia
Zenobia	Kociubowska	1960
Janina	Malinowska	1970
Jan	Kowalski	1980
Alfred	Tomba	1980
Jacek	Plackowski	1982
Jacek	Malina	1984
Alfred	Domkowski	1989

```
SELECT imie, nazwisko, rok_urodzenia FROM ludzie ORDER BY rok_urodzenia DESC;
```

imie	nazwisko	rok_urodzenia
Alfred	Domkowski	1989
Jacek	Malina	1984
Jacek	Plackowski	1982
Jan	Kowalski	1980
Alfred	Tomba	1980
Janina	Malinowska	1970
Zenobia	Kociubowsk	1960

```
SELECT nazwisko, imie, rok_urodzenia FROM ludzie WHERE rok_urodzenia > 1980  
ORDER BY nazwisko DESC;
```

nazwisko	imie	rok_urodzenia
Plackowski	Jacek	1982
Malina	Jacek	1984
Domkowski	Alfred	1989

Agregowanie danych, liczba, średnia, min, max

```
SELECT COUNT(*) FROM ludzie;
```

```
COUNT(*)
```

```
-----  
7
```

```
SELECT MAX(rok_urodzenia) FROM ludzie;
```

```
MAX(rok_urodzenia)
```

```
-----  
1989
```

```
SELECT AVG(2009 - rok_urodzenia) AS sredni_wiek FROM ludzie;
```

```
sredni_wiek
```

```
-----  
31.1428571428571
```

Agregowanie z grupowaniem

```
SELECT plec, COUNT(*) AS ile FROM ludzie GROUP BY plec;
```

plec	ile
k	2
m	5

Wartość unikalna

```
SELECT DISTINCT imie FROM ludzie;
```

imie
Alfred
Jacek
Jan
Janina
Zenobia

Warunek po grupowaniu

```
SELECT imie, COUNT(*) AS ile FROM ludzie GROUP BY imie HAVING COUNT(*) > 1;
```

imie	ile
Alfred	2
Jacek	2

Tworzy statystyki występowania imion, ale tylko tych, które wystąpiły więcej niż 1 raz w tabeli ludzie.

Złączenia wielu kolumn

Proste złączenie

```
SELECT * FROM zwierzeta  
LEFT JOIN ludzie ON zwierzeta.ludzie_id = ludzie.id;
```

id	imie	ludzie_id	id	imie	nazwisko	rok_urodzenia	plec
----	------	-----------	----	------	----------	---------------	------

wzrost

1	Azor	3	3	Jacek	Plackowski	1982	m	181
2	Miauczek	5	5	Alfred	Domkowski	1989	m	
176								
3	As	6	6	Jacek	Malina	1984	m	189
4	Kicia	3	3	Jacek	Plackowski	1982	m	181

Różnice między złączeniem lewym, wewnętrznym i prawym

LEFT JOIN bierze wszystkie dane z pierwszej (lewej) tabeli, jeśli nie udało się do nich dopasować żadnego rekordu z drugiej tabeli, wstawia wartości NULL.

INNER JOIN = JOIN bierze tylko takie pary lewej i prawej kolumny, które udało się dopasować

RIGHT JOIN - w niektórych systemach bazowych w ogóle nie występuje, działa jak LEFT JOIN tylko odwrotnie łączy tabele (do prawej dopasowuje lewą, zamiast do lewej prawą). Ten sam efekt uzyskuje się zmieniając kolejność łączenia kolumn w zapytaniu

Złączenie z podzapytaniem

```
SELECT * FROM ludzie
LEFT JOIN (SELECT ludzie_id, COUNT(*) AS ile FROM zwierzeta GROUP BY ludzie_id) as
zwierzeta_statystyki
ON ludzie.id = zwierzeta_statystyki.ludzie_id;
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost	ludzie_id	ile
1	Jan	Kowalski	1980	m	175		
2	Janina	Malinowska	1970	k	165		
3	Jacek	Plackowski	1982	m	181	3	2
4	Zenobia	Kociubowsk	1960	k	160		
5	Alfred	Domkowski	1989	m	176	5	1
6	Jacek	Malina	1984	m	189	6	1
7	Alfred	Tomba	1980	m	176		

W tym przypadku jeśli osoba nie ma zwierzęcia, baza zwraca NULL. Jeśli chcielibyśmy w tym przypadku otrzymać 0, możemy zastosować funkcję COALESCE(), która zwraca pierwszy parametr inny niż NULL.

```
SELECT ludzie.*, COALESCE(zwierzeta_statystyki.ile, 0) AS ile FROM ludzie
LEFT JOIN (SELECT ludzie_id, COUNT(*) AS ile FROM zwierzeta GROUP BY ludzie_id) as
zwierzeta_statystyki
ON ludzie.id = zwierzeta_statystyki.ludzie_id;
```

id	imie	nazwisko	rok_urodzenia	plec	wzrost	ile
----	------	----------	---------------	------	--------	-----

1	Jan	Kowalski	1980	m	175	0
2	Janina	Malinowska	1970	k	165	0
3	Jacek	Plackowski	1982	m	181	2
4	Zenobia	Kociubowsk	1960	k	160	0
5	Alfred	Domkowski	1989	m	176	1
6	Jacek	Malina	1984	m	189	1
7	Alfred	Tomba	1980	m	176	0