Wykład 6

SQL

Structured Query Language

7. SQL

- **TEMATYKA:**
  - Podstawowe informacje o języku SQL
  - Struktura języka SQL
  - Wyszukiwanie informacji w tabelach
  - Zakładanie tabel
  - Wpisywanie, aktualizacja, usuwanie danych z tabel

Inne operacje definiowania struktur tabel

- **Usuwanie tabeli:**

```
DROP TABLE tabela;
```

```
DROP TABLE Wykładowcy;
```

- **Zmiana struktury tabeli**
  - Dodanie kolumny:

```
ALTER TABLE tabela ADD COLUMN kolumna typ;
```

```
ALTER TABLE Wykładowcy ADD COLUMN RokUr DATE
```

  - Usuwanie kolumny:

```
ALTER TABLE tabela DROP COLUMN kolumna;
```

```
ALTER TABLE Wykładowcy DROP COLUMN RokUr
```

Inne operacje na tabelach

- **Zmiana nazwy:**

```
RENAME TABLE tabela TO nowa_tabela;
```

```
RENAME TABLE Moduły TO Przedmioty;
```

- **Wyświetlenie istniejących baz danych:**

```
SHOW TABLES;
```

- **Wyświetlenie struktury tabeli:**

```
DESCRIBE tabela;
```

```
DESCRIBE Przedmioty;
```

Inne operacje na tabelach c.d.

- **Modyfikacja typu kolumny:**

```
ALTER TABLE tabela MODIFY kolumna typ;
```

```
ALTER TABLE Wykładowcy MODIFY NazwiskoPrac VARCHAR(35);;
```

- **Zmiana nazwy kolumny:**

```
ALTER TABLE tabela CHANGE kolumna nowa_kolumna typ
```

```
ALTER TABLE Wykładowcy CHANGE NazwiskoPrac Nazwisko VARCHAR(35);;
```

- **Za pomocą ALTER TABLE możliwe jest również dodawanie i usuwanie atrybutów pól.**

Atrybuty pól tabeli

- **Przy tworzeniu lub zmianie tabeli można podać opcjonalne atrybuty pól (kolumn) tabeli:**

```
CREATE TABLE (pole typ atrybuty, ...);
```

- **Dostępne atrybuty:**
  - `NULL` – można nie podawać wartości (domyślnie)
  - `NOT NULL` – wartość musi być podana
  - `DEFAULT` *`wartość`* – gdy nie podamy wartości
  - `AUTO_INCREMENT` – automatycznie zwiększany licznik
  - `COMMENT` *`'opis'`* – komentarz
  - `PRIMARY KEY, INDEX` – indeksy główne

AUTO_INCREMENT i DEFAULT

- **AUTO_INCREMENT – nie wpisujemy danych, baza wpisuje aktualny stan licznika i zwiększa go o 1.**
- **DEFAULT – jeżeli nie wprowadzimy danych, zostanie wpisana wartość domyślna**

```
CREATE TABLE wykonawcy
       (id              INT NOT NULL AUTO_INCREMENT,
       wykonawca        VARCHAR(30),
       opis             TEXT DEFAULT ('brak opisu');
INSERT INTO wykonawcy (wykonawca) VALUES ('XYZ');
```

Wynik: (1, 'XYZ', 'brak opisu')

Indeksowanie tabel

- **Na wybrane kolumny tabeli mogą być nakładane indeksy (klucze) w celu:**
  - przyspieszenia wyszukiwania
  - zdefiniowania relacji pomiędzy tabelami (służą do tego klucze obce)

- **Tworzenie indeksów nie weszło do standardu SQL, jest jednak oferowane prze większość systemów komercyjnych**

- **Tworzenie indeksu w już istniejącej tabeli:**

```
CREATE INDEX indeks ON tabela(kolumna);
```

```
CREATE INDEX WgNazwisk ON Wykładowcy(NazwiskoPrac);

CREATE INDEX WgNazwStat ON Wykładowcy (NazwiskoPrac(10), Status);
```

Indeks wielokolumnowy, indeksowanych 10 zanków z pola NaziskoPrac

- **Usunięcie indeksu (nie usuwa danych!):**

```
DROP INDEX indeks ON tabela;
DROP INDEX WgNazwisk ON Wykładowcy;
```

Wartości niepowtarzalne

- **UNIQUE – żadne dwa rekordy w tabeli nie mogą mieć jednakowych danych w indeksowanej kolumnie. Jest to jednocześnie INDEX.**

- **Jeżeli indeksowana kolumna ma atrybut NOT NULL, dane w kolumnie muszą być unikatowe i muszą być wprowadzone.**

- **Jeżeli indeksowana kolumna ma atrybut NULL, dane w kolumnie muszą być unikatowe, ale mogą nie być wprowadzane (pole może pozostać puste).**

Indeks główny

- Indeks główny – PRIMARY KEY
- identyfikuje jednoznacznie każdy rekord w tabeli
- może istnieć tylko jeden w tabeli
- jest typu UNIQUE
- indeksowana kolumna otrzymuje automatycznie atrybut NOT NULL
- ma nazwę PRIMARY (nie można podać własnej)
- bierze domyślnie udział w relacjach z innymi tabelami

Tworzenie indeksu głównego

- **Tworzenie indeksu głównego podczas definiowania tabeli – w definicji kolumny:**

```
CREATE TABLE dane {
nazwisko CHAR(30) NOT NULL,
pesel                CHAR(11) PRIMARY KEY,
};
```

- **To samo w definicji tabeli (może to być ind. wielokolumn.)** CREATE TABLE dane

```
{
nazwisko CHAR(30) NOT NULL,
pesel                CHAR(11),
PRIMARY KEY (pesel)
};
```

Tworzenie indeksu głównego

- **Tworzenie indeksu głównego w już istniejącej tabeli:**
- ALTER TABLEEjDD PRIMARY KEY (pesel);

- **Usuwanie indeksu głównego w tabeli (nie usuwa danych!):**
- ALTER TABLEEzP PRIMARY KEY;

Wstawianie danych do tabeli

- **Instrukcja wstawiania składa się z następujących elementów**
  - słowa kluczowego `INSERT INTO`
  - nazwy tabeli
  - listy atrybutów ujętej w nawiasy
  - słowa kluczowego `VALUES`
  - listy wartości przypisywanych poszczególnym atrybutom

```
INSERT INTO tabela(pole1 ... PoleN) VALUES (wartość1 ... wartośćN)
```

- **Wstawianie danych z podaniem wszystkich kolumn tabeli (należy zachować prawidłową kolejność)**

```
INSERT INTO Wykładowcy
VALUES (244, 'Buczek Jan', 'P');
```

- **Wstawianie danych do wybranych kolumn tabeli**

```
INSERT INTO Wykładowcy (NrPrac, Status)
VALUES (244, 'P');
```

- **Wstawianie do tabeli danych uzyskanych w wyniku zapytania:**

```
INSERT INTO nowa (autor, dzielo)
SELECT DISTINCT wykonawca, album
FROM albumy;
```

Usuwanie rekordów

- **Instrukcja usuwania rekordów z tabeli składa się z:**
  - słowa kluczowego **DELETE FROM**
  - nazwy tabeli
  - słowa kluczowego **WHERE**
  - warunku

```
DELETE FROM tabela WHERE warunek
```

```
DELETE FROM Wykładowcy
WHERE NazwiskoPrac = 'Buczek Jan';
```

Uaktualnianie rekordów

- **Instrukcja aktualizacji składa się z następujących elementów**
  - słowa kluczowego **UPDATE**
  - nazwy tabeli
  - słowa kluczowego **SET**
  - listy wyrażeń, z których każde określa wartość pewnego atrybutu
  - słowa kluczowego **WHERE**
  - warunku

```
UPDATE tabela SET nowe_wartości WHERE warunek
```

```
UPDATE Moduły
SET NrPrac = 260
WHERE NrPrac = 244
```

Definiowanie perspektyw

- **Perspektywa w SQL jest relacją, która nie ma fizycznego odpowiednika (tabele istnieją faktycznie)**
- **Perspektywy mogą być wykorzystywane w zapytaniach, a nawet niekiedy modyfikować**
- **Deklaracja perspektywy składa się z:**
  - słowa kluczowego CREATE VIEW
  - nazwy perspektywy
  - słowa kluczowego AS
  - zapytania, które określa zakres danych perspektywy

```
CREATE VIEW perspektywa AS definicja
```

- **Perspektywa z jednej tabeli**

```
CREATE VIEW Profesorzy AS
        SELECT *
        FROM Wykładowcy
        WHERE Status = 'P';
```

Wykładowcy		
NrPrac	**NazwiskoPrac**	**Status**
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A

Definiowanie perspektyw c.d.

- Perspektywa może wybierać dane z kilku tabel

```
CREATE VIEW Obsada AS
        SELECT NazwaModułu, NazwiskoPrac
        FROM Moduły, Wykładowcy
        WHERE Moduły.NrPrac = Wykładowcy.NrPrac;
```

Moduły			
NazwaModułu	Poziom	KodKursu	NrPrac
Systemy relacyjnych baz danych	1	CSD	244
Projektowanie relacyjnych baz danych	1	CSD	244
Dedukcyjne bazy danych	4	CSD	445
Obiektowe bazy danych	4	CSD	445
Roproszone bazy danych	2	CSD	247

Wykładowcy		
NrPrac	NazwiskoPrac	Status
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A

Obsada	
NazwaModułu	NazwiskoPrac
Systemy relacyjnych baz danych	Buczek Jan
Projektowanie relacyjnych baz danych	Buczek Jan
Dedukcyjne bazy danych	Wysocki Edward
Obiektowe bazy danych	Wysocki Edward
Roproszone bazy danych	Kalita Henryk

Modyfikowanie perspektyw

- **Najczęściej perspektyw nie można modyfikować ! Gdzie zostanie umieszczony w prowadzony rekord, przecież mamy doczynienia ze strukturą logiczną ?**

- **SQL 2 określa formalnie kiedy perspektywę można modyfikować:**
  - perspektywa musi być zdefiniowana przez selekcję atrybutów tylko z jednej relacji (tabeli) R,
  - klauzula `WHERE` nie może zawierać zapytania dotyczącego relacji R
  - w klauzuli `SELECT` musi być wybranych wystarczająco dużo atrybutów. Pozostałe otrzymają wartość `NULL` lub wartości domyślne.

- **Do perspektyw modyfikowalnych można zarówno wstawiać jak i usuwać rekordy.**

Wartość NULL

- **Bardzo ważne zastosowanie `NULL` zachodzi przy wyliczaniu złączenia, gdy należy chronić dane, a krotka (rekord) z jednej relacji nie daje się połączyć z żadną krotką z innej relacji (złączenie zewnętrzne).**

- **Wartości `NULL` są stosowane w krotkach wstawianych do relacji wówczas, gdy w poleceniu nie określono wszystkich składowych wstawianej krotki**

- **Przy działaniach z wartością `NULL` należy pamiętać o dwóch regułach:**
  - Jeśli jako argument działania arytmetycznego (*, +) występuje `NULL` i pewna inna wartość to wynikiem jest wartość `NULL`
    - `NULL + ??? = NULL`
    - `NULL * ??? = NULL`

    Pułapki: 0 * NULL = NULL
    NULL – NULL = NULL
  - Przy porównywaniu `NULL` z inną wartością przy pomocy operatorów algebraicznych (=, > , <) wynik otrzymuje wartość `UNKNOWN` (trzecia wartość logiczna obok `TRUE` i `FALSE`)
    - `NULL > ??? => UNKNOWN`
    - `NULL = ??? => UNKNOWN`
    - `NULL < ??? => UNKNOWN`

Wartość logiczna UNKNOWN

- Określając reguły wyznaczania wartości logicznych w których występuje wartość UNKNOWN przyjmuje się ją jako coś pośredniego pomiędzy wartościami TRUE = 1, FALSE = 0 i przypisuje wartość ½.

x	y	x AND y	x OR y	NOT x
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	UNKNOWN	UNKNOWN	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE
UNKNOWN	TRUE	UNKNOWN	TRUE	UNKNOWN
UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN
UNKNOWN	FALSE	FALSE	UNKNOWN	UNKNOWN
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	UNKNOWN	FALSE	UNKNOWN	TRUE
FALSE	FALSE	FALSE	FALSE	TRUE

Zadanie: Czy w wyniku wykonania selekcji zostaną wybrane wszystkie rekordy

```
SELECT *
FROM Moduły
WHERE Poziom <= 3 OR Poziom > 3;
```

NIE!

Jeżeli w polu Poziom występują wartości UNKNOWN

Złączenia w SQL2
złączenie naturalne (równozłączenie)

- SQL wykonuje złączenia poprzez wskazanie wspólnych atrybutów w klauzuli `WHERE` instrukcji `SELECT`

```
SELECT NazwaModułu, KodKursu, NazwiskoPrac
FROM Moduły M, Wykładowcy W
WHERE M.NrPrac = W.NrPrac;
```

lub w SQL 2

```
SELECT NazwaModułu, KodKursu, NazwiskoPrac
FROM Wykładowcy NATURAL JOIN Moduły;
```

Możliwe bo pola klucz zewnętrzny w tabeli Moduły ma identyczną nazwę jak klucz pierwotny w tabeli Wykładowcy

gdyby klucze były różne

```
SELECT NazwaModułu, KodKursu, NazwiskoPrac
FROM Wykładowcy W NATURAL JOIN Moduły M
ON M.NrPrac = W.NrPrac;
```

- W przypadku wybrania wszystkich kolumn do relacji wyjściowej pola powtarzające się zostaną dołączone pojedynczo.
- W relacji wyjściowej znajdą się tylko rekordy powiązane.

Złączenia w SQL2

złączenie naturalne (równozłączenie)

- **Polecenie ma postać**

```
Relacja1 NATURAL JOIN Relacja2
```

```
SELECT NazwaModułu, KodKursu, NazwiskoPrac
FROM Wykładowcy NATURAL JOIN Moduły
```

lub

```
SELECT NazwaModułu, KodKursu, NazwiskoPrac
FROM Wykładowcy W NATURAL JOIN Moduły M
ON M.NrPrac = W.NrPrac;
```

Moduły			
NazwaModułu	**Poziom**	**KodKursu**	**NrPrac**
Systemy relacyjnych baz danych	1	CSD	244
Projektowanie relacyjnych baz danych	1	CSD	244
Roproszone bazy danych	2	CSD	247

Wykładowcy		
NrPrac	**NazwiskoPrac**	**Status**
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A

ModWykR						
NazwaModułu	**Poziom**	**KodKursu**	**NrPrac**	**NrPrac**	**NazwiskoPrac**	**Status**
Systemy relacyjnych baz danych	1	CSD	244	244	Buczek Jan	P
Projektowanie relacyjnych baz danych	1	CSD	244	244	Buczek Jan	P
Rozproszone bazy danych	2	CSD	247	247	Wysocki Edward	SW

Złączenia w SQL2
Złączenie zewnętrzne

- **Podobne do złączenia naturalnego, różnica polega na tym, iż pozostawiane są w relacji wynikowej wiersze nie posiadające odpowiedników w obu relacjach wyjściowych**

- **Wyróżnia się złączenia zewnętrzne:**

  - Lewostronne – zachowuje nie pasujące wiersze z tabeli będącej pierwszym argumentem,

    ```
    Relacja1 LEFT OUTER JOIN Relacja 2
    ```

  - Prawostronne – zachowuje nie pasujące wiersze z tabeli będącej drugim argumentem

    ```
    Relacja1 RIGHT OUTER JOIN Relacja 2
    ```

  - Dwustronne – zachowuje nie pasujące wiersze z obu tabeli

    ```
    Relacja1 FULL OUTER JOIN Relacja 2
    ```

Przykład złączenia zewnętrznego lewostronnego w SQL2

```
SELECT *
FROM Moduły LEFT OUTER JOIN Wykładowcy
```

Moduły

NazwaModułu	Poziom	KodKursu	NrPrac
Systemy relacyjnych baz danych	1	CSD	244
Projektowanie relacyjnych baz da	1	CSD	244
Dedukcyjne bazy danych	4	CSD	445
Obiektowe bazy danych	4	CSD	445
Rozproszone bazy danych	2	CSD	247
Opracowanie baz danych	2	CSD	null
Administrowanie danymi	2	CSD	null

Wykładowcy

NrPrac	NazwiskoPrac	Status
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A
145	Zaborowski Jan	SW
447	Fusiarz Kamila	L

NazwaModułu	Poziom	KodKursu	NrPrac	NazwiskoPrac	Status
Systemy relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Projektowanie relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Dedukcyjne bazy danych	4	CSD	445	Kalita Henryk	A
Obiektowe bazy danych	4	CSD	445	Kalita Henryk	A
Rozproszone bazy danych	2	CSD	247	Wysocki Edward	SW
Opracowanie baz danych	2	CSD	null	null	null
Administrowanie danymi	2	CSD	null	null	null

Przykład złączenia zewnętrznego prawostronnego w SQL2

```
SELECT *
FROM Moduły RIGHT OUTER JOIN Wykładowcy
```

Moduły			
NazwaModułu	**Poziom**	**KodKursu**	**NrPrac**
Systemy relacyjnych baz danych	1	CSD	244
Projektowanie relacyjnych baz da	1	CSD	244
Dedukcyjne bazy danych	4	CSD	445
Obiektowe bazy danych	4	CSD	445
Rozproszone bazy danych	2	CSD	247
Opracowanie baz danych	2	CSD	null
Administrowanie danymi	2	CSD	null

Wykładowcy		
NrPrac	**NazwiskoPrac**	**Status**
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A
145	Zaborowski Jan	SW
447	Fusiarz Kamila	L

NazwaModułu	Poziom	KodKursu	NrPrac	NazwiskoPrac	Status
Systemy relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Projektowanie relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Dedukcyjne bazy danych	4	CSD	445	Kalita Henryk	A
Obiektowe bazy danych	4	CSD	445	Kalita Henryk	A
Rozproszone bazy danych	2	CSD	247	Wysocki Edward	SW
null	null	null	145	Zaborowski Jan	SW
null	null	null	447	Fusiarz Kamila	L

Przykład złączenia zewnętrznego dwustronnego w SQL2

```
SELECT *
FROM Moduły FULL OUTER JOIN Wykładowcy
```

Moduły

NazwaModułu	Poziom	KodKursu	NrPrac
Systemy relacyjnych baz danych	1	CSD	244
Projektowanie relacyjnych baz da	1	CSD	244
Dedukcyjne bazy danych	4	CSD	445
Obiektowe bazy danych	4	CSD	445
Rozproszone bazy danych	2	CSD	247
Opracowanie baz danych	2	CSD	null
Administrowanie danymi	2	CSD	null

Wykładowcy

NrPrac	NazwiskoPrac	Status
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A
145	Zaborowski Jan	SW
447	Fusiarz Kamila	L

NazwaModułu	Poziom	KodKursu	NrPrac	NazwiskoPrac	Status
Systemy relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Projektowanie relacyjnych baz danych	1	CSD	244	Buczek Jan	P
Dedukcyjne bazy danych	4	CSD	445	Kalita Henryk	A
Obiektowe bazy danych	4	CSD	445	Kalita Henryk	A
Rozproszone bazy danych	2	CSD	247	Wysocki Edward	SW
Opracowanie baz danych	2	CSD	null	null	null
Administrowanie danymi	2	CSD	null	null	null
null	null	null	145	Zaborowski Jan	SW
null	null	null	447	Fusiarz Kamila	L

Iloczyn kartezjański (złączenie krzyżowe) w SQL2

- **Operator iloczynu kartezjańskiego ma bardzo ograniczone zastosowanie.**

```
Relacja1 CROSS JOIN  Relacja1
```

```
SELECT *
FROM Moduły CROSS JOIN Wykładowcy
```

Moduły

NazwaModułu	Poziom	KodKursu	NrPrac
Systemy relacyjnych baz danych	1	CSD	244
Roproszone bazy danych	2	CSD	247

Wykładowcy

NrPrac	NazwiskoPrac	Status
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A

NazwaModułu	Poziom	KodKursu	NrPrac	NrPrac	NazwiskoPrac	Status
Systemy relacyjnych baz danych	1	CSD	244	244	Buczek Jan	P
Systemy relacyjnych baz danych	1	CSD	244	247	Wysocki Edward	SW
Systemy relacyjnych baz danych	1	CSD	244	445	Kalita Henryk	A
Roproszone bazy danych	2	CSD	247	244	Buczek Jan	P
Roproszone bazy danych	2	CSD	247	247	Wysocki Edward	SW
Roproszone bazy danych	2	CSD	247	445	Kalita Henryk	A

Suma w SQL2

- Suma daje możliwość połączenia dwóch zgodnych relacji.

```
Relacja1 UNION Relacja1
```

- W praktyce najczęściej dokonuje się sumowania wyników dwóch zapytań.

```
SELECT *
FROM Wykładowcy
UNION
SELECT *
FROM Administracja
```

Wykładowcy		
NrPrac	**NazwiskoPrac**	**Status**
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A

Administracja		
NrPrac	**NazwiskoPrac**	**Status**
1010	Pawłowicz Maria	U
247	Wysocki Edward	SW

NrPrac	NazwiskoPrac	Status
244	Buczek Jan	P
247	Wysocki Edward	SW
445	Kalita Henryk	A
1010	Pawłowicz Maria	U

Instrukcje sterowania dostępem do danych

- Każda SZBD musi zawierać mechanizm gwarantujący dostęp do baz danych tylko dla upoważnianych (przez administratora) użytkowników.
- Język SQL zawiera instrukcje `GRANT` oraz `REVOKE` dla zabezpieczenia danych.
- Mechanizm zabezpieczeń wykorzystuje identyfikatory użytkowników oraz ich prawa.
- Identyfikator rejestruje administrator bazy danych (DBA). Każdy identyfikator jest połączony z hasłem.
- Identyfikator użytkownika pozwala wyznaczyć obiekty bazy danych, z którymi może on pracować.

Instrukcje sterowania dostępem

- **Prawa to są działania, które użytkownik może wykonywać z daną relacją bazy danych.**

- **W SQL2 określono następujące prawa:**
  - **SELECT** – prawo do wybierania danych z tabeli;
  - **INSERT** – prawo do wstawiania nowych rekordów do tabeli;
  - **UPDATE** – prawo do modyfikowania danych w tabeli;
  - **DELETE** – prawo do usuwania rekordów z tabeli;
  - **REFERENCES** – prawo do odwoływania się do danej struktury w więzach integralności.
  - **USAGE** – prawo do wykorzystywania dziedzin i elementów schematu w deklaracjach

- **Przywileje INSERT oraz UPDATE mogą dotyczyć pojedynczych kolumn w tabeli.**

- **Każdy obiekt, który jest stworzony w środowisku SQL, ma właściciela i właściciel ma wobec niego wszelkie prawa.**

Nadawanie praw – instrukcja GRANT

- **Do nadawania praw służy instrukcja GRANT**

- **Instrukcja GRANT składa się z:**
  - słowa kluczowego GRANT,
  - listy złożonej z przydzielanych praw,
  - słowa kluczowego ON
  - elementu bazy danych (tabela, perspektywa, dziedzina itp...) do którego nadajemy prawa
  - słowa kluczowego TO
  - listy użytkowników którym przydzielamy prawa
  - opcjonalnie z klauzuli WITH GRANT OPTION

```
GRANT  prawa | ALL PRIVILEGES
       ON element_bazy_danych
       TO użytkownicy | PUBLIC
       [WITH GRANT OPTION]
```

ALL PRIVILEGES – wszystkie prawa;
PUBLIC – wszyscy użytkownicy;
WITH GRANT OPTION – prawo do przekazywania praw

Nadajmy użytkownikowi o identyfikatorze Maria prawo do wstawiania i wybierania elementów w tabeli Moduły oraz wszystkie prawa z możliwością ich przekazywania w tabeli Wykładowcy

```
GRANT  SELECT, INSERT
       ON Moduły
       TO Maria
GRANT  ALL PRIVILEGES
       ON Wykładowcy
       TO Maria
       WITH GRANT OPTION
```

Odbieranie praw – instrukcja REVOKE

- **Nadane prawa można w każdej chwili odebrać**

- **Instrukcja REVOKE składa się z:**
  - słowa kluczowego REVOKE,
  - listy złożonej z odbieranych praw praw,
  - słowa kluczowego ON
  - elementu bazy danych (tabela, perspektywa, dziedzina itp...) do którego obieramy prawa
  - słowa kluczowego FROM
  - listy użytkowników którym odbieramy prawa
  - opcjonalnie z klauzul CASCADE, RESTRICT

Odbierzmy użytkownikowi o identyfikatorze Maria prawo do wstawiania elementów w tabeli Moduły oraz prawo do przekazywania praw z tabeli Wykładowcy

```
REVOKE  SELECT
        ON Moduły
        FROM Maria
REVOKE  GRANT OPTION FOR
                ALL PRIVILEGES
        ON Wykładowcy
        FROM Maria
```

```
REVOKE [GRANT OPTION FOR]  prawa | ALL PRIVILEGES
       ON element_bazy_danych
       FROM użytkownicy | PUBLIC
       [CASCADE | RESTRICT]
```

GRANT OPTION FOR	**– odebranie prawa do przekazywania praw**
ALL PRIVILEGES	**– wszystkie prawa;**
PUBLIC	**– wszyscy użytkownicy;**
CASCADE	**– odbiera prawa nadane przez osobę której prawa odbieramy**
RESTRICT	**– nie odbierać praw jeżeli powoduje to usunięcie praw innych niż jawnie wyspecyfikowane**

- **Po odebraniu praw ogólnych prawa szczegółowe obowiązują dalej**

Operacje na bazach danych

- **Tworzenie bazy danych:**
- CREATE DATABASE baza;

- **Usuwanie całej bazy:**
- DROP DATABASE baza;

- **Wyświetlenie istniejących baz danych:**
- SHOW DATABASES;

- **Przełączenie się na inną bazę danych:**
- USE baza;

Funkcje SQL

- **Język SQL udostępnia szereg funkcji umożliwiających wykonywanie operacji na danych w zapytaniach.**

- **Funkcje:**
  - matematyczne
  - tekstowe
  - daty i czasu

- **Funkcje te mogą być wykorzystywane w instrukcji SELECT, w warunku wyboru kolumn lub w warunku wyboru wierszy.**

Funkcje SQL c.d.

- **Przykład stosowania funkcji w instrukcji SELECT**
- **Funkcja UPPER() zamienia litery na wielkie.**

- **Użycie w warunku wyboru kolumn – zamienia litery na wielkie w zwracanych danych:**
- ```
  SELECT UPPER(wykonawca) FROM albumy;
  ```

- **Użycie w warunku wyboru rekordu – zawartość pola po konwersji musi odpowiadać warunkowi:**
- ```
  SELECT * FROM albumy
  WHERE UPPER(wykonawca)='U2';
  ```

Funkcje matematyczne

- ABS(x) – wartość bezwzględna
- SIGN(x) – znak liczby (-1, 0, 1)
- MOD(m,n) – reszta z dzielenia M/N
- FLOOR(x) – zaokrąglenie w dół
- CEIL(x) – zaokrąglenie w górę
- ROUND(x) – zaokrąglenie do najbliższej l. całkowitej
- m DIV n – część całkowita z dzielenia m/n
- EXP(x) – e^x
- LN(x), LOG2(x), LOG10(x), LOG(b,x) – logarytmy
- POWER(x,y) = x^ySQRT(x) – pierwiastek kwadratowy
- PI() – wartość π
- SIN(x), COS(x), TAN(x), COT(x) – funkcje trygonometr.
- ASIN(x), ACOS(x), ATAN(x) – odwrotne funkcje tryg.
- CRC32('wyr') – kod CRC wyrażenia *wyr*
- RAND() – liczba losowa od 0 do 1
- LEAST(x,y,...) – najmniejsza wartość z listy
- GREATEST(x,y,...) – największa wartość z listy
- DEGREES(x), RADIANS(x) – konwersja stopnie/radiany
- TRUNCATE(x,d) – skrócenie *x* do *d* miejsc po przecinku

Funkcje tekstowe (1)

- ASCII(x) – kod ASCII znaku
- ORD(x) – suma na podstawie kodów ASCII
- CONV(x,m,n) – konwersja między systemami liczbowymi
- BIN(x), OCT(x), HEX(x) – konwersja między systemami
- CHAR(x) – ciąg złożony ze znaków o podanych kodach
- CONCAT(s1,s2,...) – łączy podane napisy w jeden
- CONCAT_WS(sep,s1,s2,...) – łączy napisy separatorem
- LENGTH(s) – długość napisu
- LOCATE(s1,s2,p) – pozycja napisu *s1* w *s2* (szuk. od p)
- INSTR(s1,s2) – pozycja napisu *s2* w *s1*
- LPAD(s1,n,s2) – poprzedza *s1* ciągiem *s2* do długości *n*
- RPAD(s1,n,s2) – dopisuje do *s1* ciąg *s2* do długości *n*
- LEFT(s,n) – *n* pierwszych znaków z napisu *s*
- RIGHT(s,n) – *n* ostatnich znaków z napisu *s*
- SUBSTRING(s,m,n) – *n* znaków z napisu *s* od poz. *m*
- SUBSTRING_INDEX(s,sep,n) – część napisu *s* przed *n*-tym wystąpieniem separatora *sep*

Funkcje tekstowe cd

- LTRIM(s) – usuwa początkowe spacje
- RTRIM(s) – usuwa końcowe spacje
- TRIM(s) – usuwa początkowe i końcowe spacje
- SPACE(n) – napis złożony z *n* spacji
- REPLACE(s1,s2,s3) – zamień *s2* na *s3* w napisie *s1*
- REPEAT(s,n) – napis z *n* powtórzeń *s*
- REVERSE(s) – odwraca napis *s*
- INSERT(s1,m,n,s2) – wstawia *n* znaków *s2* do *s1* na poz. *m*
- ELT(n,s1,s2,...) – zwraca *n*-ty napis ze zbioru
- FIELD(s,s1,s2,...) – zwraca indeks napisu *s* w zbiorze
- LOWER(s) – zmienia litery na małe
- UPPER(s) – zmienia litery na wielkie
- LOAD_FILE(plik) – odczytuje zawartość pliku
- QUOTE(s) – poprzedza znaki specjalne znakiem '\'
- STRCMP(s1,s2) – porównanie dwóch napisów

Funkcje daty i czasu (1)

- DATE(s) – pobiera datę z wyrażenia *s*
- TIME(s) - pobiera czas z wyrażenia *s*
- TIMESTAMP(s) – pobiera datę i czas z wyrażenia *s*
- DAYOFWEEK(data) – podaje dzień tygodnia
- DAYOFMONTH(data) – podaje dzień miesiąca
- DAYOFYEAR(data) – podaje dzień w roku
- MONTH(data) – podaje numer miesiąca
- DAYNAME(data) – podaje nazwę dnia
- MONTHNAME(data) – podaje nazwę miesiąca
- WEEK(data) – podaje numer tygodnia (od 0)
- WEEKOFYEAR(data) – podaje numer tygodnia (od 1)
- YEAR(data) – podaje rok
- YEARWEEK(data) – podaje rok i numer tygodnia
- HOUR(data) – podaje godzinę
- MINUTE(data) – podaje minutę
- SECOND(data) – podaje sekundę
- MICROSECOND(data) – podaje ułamki sekundy
- PERIOD_ADD(p,m) – dodaje *m* miesięcy do daty *p*
- PERIOD_DIFF(p1,p2) – różnica dwóch dat

Funkcje daty i czasu (2)

- DATE_ADD(data, INTERVAL wyr typ)
  – dodaje do daty podany czas
- DATE_SUB(data, INTERVAL wyr typ) – odejmowanie daty
- ADDDATE(data,n) – dodaje *n* dni do daty
- SUBDATE(data,n) – odejmuje *n* dni od daty
- ADDTIME(s1,s2) – dodawanie czasu
- SUBTIME(s1,s2) – odejmowanie czasu
- EXTRACT(typ FROM data) – pobranie części daty
- TO_DAYS(data) – zamienia datę na numer dnia
- FROM_DAYS(n) – zamienia numer dnia na datę
- DATE_FORMAT(data,format) – formatowanie daty
- TIME_FORMAT(czas,format) – formatowanie czasu
- MAKEDATE(rok,dzień) – podaje datę
- MAKETIME(godz,min,sek) – podaje czas
- CURDATE() – bieżąca data
- CURTIME() – bieżący czas
- NOW() – bieżąca data i czas
- UNIX_TIMESTAMP() – bieżąca data i czas w formacie UNIX
- SEC_TO_TIME(sek) – konwersja sekund na czas
- TIME_TO_SEC(czas) – konwersja czasu na sekundy

Przykład operacji na datach

- **Dodawanie i odejmowanie**
- DATE_ADD('1997-12-31 23:59:59', INTERVAL 1 DAY);
- DATE_ADD('1997-12-31 23:59:59',
  INTERVAL '1:1' MINUTE_SECOND);
- DATE_SUB('1998-01-02', INTERVAL 31 DAY);

- **Pobranie części daty**
- EXTRACT(YEAR FROM "1999-07-02");

- **Formatowanie daty**
- DATE_FORMAT('1997-10-04 22:23:00', '%W %M %Y');

Funkcje konwersji

- Konwersje typów danych
- CAST(wyr AS typ) – zmiana *wyr* na *typ*
- CONVERT(wyr,typ) – jw.
- CONVERT(wyr USING kod) – zmiana strony kodowej

- Typy: BINARY, CHAR, DATE, DATETIME, SIGNED, TIME, UNSIGNED

Transakcje

- **Domyślnie wszystkie instrukcje są wykonywane od razu po ich wprowadzeniu – zmiana danych w bazie.**
- **W pewnych sytuacjach nie chcemy aby wykonywane operacje modyfikowały fizyczny zbiór danych.**
- **Tryb transakcji – wprowadzane operacje zostaną wykonane dopiero po podaniu odpowiedniej komendy.**

Transakcje c.d.

- **START TRANSACTION – rozpoczęcie transakcji**
- **Kolejne operacje są zapamiętywane, ale nie są wykonywane.**
- **COMMIT – wykonanie operacji z całej transakcji**
- **ROLLBACK – cofnięcie do początku transakcji**

- **Niektóre komendy automatycznie wykonują COMMIT, np. CREATE INDEX, DROP INDEX, DROP TABLE, DROP DATABASE, ALTER TABLE, RENAME TABLE, TRUNCATE**
- **Możliwe jest ustawienie w trakcie transakcji punktów zapisu za pomocą komendy**
- SAVEPOINT *nazwa*

- **Wykonanie komendy ROLLBACK TO SAVEPOINT *nazwa***
- **powoduje cofnięcie do punktu zapisu o podanej nazwie**

- COMMIT nadal wykonuje całą transakcję.

Blokowanie tabel

- **W pewnych sytuacjach potrzebne jest czasowe zablokowanie tabeli, aby inny użytkownik nie zmodyfikował danych.**
- **Blokowanie:**
- LOCK TABLES *tabela1 typ*, *tabela2 typ*, ... ;
- **Typ blokady:**
  - READ – blokada do odczytu
  - WRITE – blokada do zapisu

- **Odblokowanie tabel:**
- UNLOCK TABLES;