GDAŃSK UNIVERSITY
OF TECHNOLOGY

Podstawy języka SQL

Marek Kulawiak, Krzysztof Goczyła

Wydział Elektroniki, Telekomunikacji i Informatyki

GDAŃSK UNIVERSITY OF TECHNOLOGY

Structured Query Language

- SQL jest językiem służący do zarządzania zawartością relacyjnych baz danych
- Składnię SQL można podzielić na kilka podzbiorów:
  - **D**ata **D**efinition **L**anguage (instrukcje CREATE, ALTER, DROP, RENAME)
  - **D**ata **M**anipulation **L**anguage (instrukcje INSERT, UPDATE, DELETE)
  - **D**ata **Q**uery **L**anguage (instrukcja SELECT)
  - **D**ata **C**ontrol **L**anguage (instrukcje GRANT, REVOKE)
- Każdy DBMS (Database Management System) realizuje standard SQL w innym stopniu, a także oferuje do niego własne rozszerzenia
  - Przykładowe DBMS: PostgreSQL, MS SQL Server, MySQL, Oracle

GDAŃSK UNIVERSITY OF TECHNOLOGY

Powszechne typy danych

- Ciągi znaków:
  - VARCHAR(max_length), CHARACTER(fixed_length)
- Liczby całkowite:
  - SMALLINT, INTEGER, BIGINT
- Liczby stałoprzecinkowe:
  - DECIMAL(precision, scale)
- Liczby zmiennoprzecinkowe:
  - REAL, FLOAT, DOUBLE PRECISION
- Data i godzina:
  - DATE, TIME, TIMESTAMP
- Dane binarne:
  - VARBINARY(max_length), BINARY(fixed_length)
- Wartości logiczne:
  - BOOLEAN

GDAŃSK UNIVERSITY OF TECHNOLOGY

Tworzenie tabel

```
-- składnia
CREATE TABLE TableName
(
    ColumnName column_type options,
    ColumnName column_type options,
    ...
);


-- przykład
CREATE TABLE SomeTable
(
    ID int PRIMARY KEY,
    SomeString varchar NOT NULL,
    ForeignData real REFERENCES AnotherTable
);
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Wstawianie danych

```sql
-- składnia 1
INSERT INTO TableName
  VALUES
    (columns, in, correct,
     order);
```

```sql
-- składnia 2
INSERT INTO TableName
  (col1, col2)
VALUES
  (col1_value, col2_value);
```

```sql
-- przykład 1
INSERT INTO SomeTable
  VALUES
    (1, 'I am data', 42),
    (2, 'Some string', 66);
```

```sql
-- przykład 2
INSERT INTO SomeTable
  (ID, SomeString)
VALUES
  (3, 'More data'),
  (4, 'Another string');
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Pobieranie danych

```sql
SELECT * FROM SomeTable; -- pobranie wszystkich kolumn

SELECT * FROM SomeTable
  WHERE ForeignData > 5;

SELECT ID, ForeignData FROM SomeTable
  WHERE SomeString = 'I am data';

SELECT ID FROM SomeTable
  WHERE ForeignData IS NOT NULL;

-- zwrócenie nowej kolumny i nadanie jej nazwy:
SELECT (ID*10) AS "ID multiplied by 10"
  FROM SomeTable;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Modyfikacja danych

```
-- zmiana nazwy stanowiska
UPDATE Jobs
  SET Position = 'CG Artist'
  WHERE Position = '3D Artist';


-- dodanie kolumny
ALTER TABLE Workers
  ADD COLUMN Title varchar;

-- usunięcie kolumny
ALTER TABLE Workers
  DROP COLUMN Title;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Usuwanie danych

```
-- usuwanie wierszy
DELETE FROM Workers
  WHERE ID = 4;


-- usunięcie całej tabeli
DROP TABLE Workers;

-- usunięcie tabeli jeśli istnieje
DROP TABLE IF EXISTS Workers;


-- usunięcie całej bazy danych
DROP DATABASE MyDB;

-- usunięcie bazy jeśli istnieje
DROP DATABASE IF EXISTS MyDB;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Gotowy przykład

```
-- usunięcie tabel
DROP TABLE IF EXISTS Workers;
DROP TABLE IF EXISTS Jobs;

-- odtworzenie struktury tabel
CREATE TABLE Jobs
(
    ID int PRIMARY KEY,
    Position varchar
);
CREATE TABLE Workers
(
    ID int PRIMARY KEY,
    FullName varchar NOT NULL,
    WorkHours real,
    Position int
        REFERENCES Jobs
);
```

```
-- wstawienie nowych wierszy
INSERT INTO Jobs
    (ID, Position)
VALUES
    (1, 'Java Programmer'),
    (2, 'Python Programmer'),
    (3, '3D Artist');

INSERT INTO Workers
    (ID, FullName, WorkHours,
     Position)
VALUES
    (1, 'Sunny Oracle', 0.5, 1),
    (2, 'John Doe', 1.0, 2),
    (3, 'Jane Doe', 1.0, 2),
    (4, 'Anon Noname', 0.5, 3);

-- wypisanie zawartości
SELECT * FROM Workers;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Dostępne operatory

- Arytmetyczne:
  - + - * /
  - ^ ** (potęgowanie)
- Porównania:
  - < <= = >= > <>
- Logiczne:
  - AND OR NOT
- Specjalne:
  - IN LIKE BETWEEN MATCHES
- Pozostałe:
  - IS NULL IS NOT NULL

GDAŃSK UNIVERSITY OF TECHNOLOGY

Zastosowanie operatorów

```
/* Wyszukaj tych pracowników, których imię zaczyna się
   od litery J, K, L, M, N, O, P, Q lub R (ale nie S):
SELECT * FROM Workers
  WHERE FullName BETWEEN 'J' AND 'S';


-- Wyszukaj wszystkich programistów w firmie:
SELECT * FROM Workers WHERE Position IN
  (SELECT ID FROM Jobs
    WHERE Position LIKE '%Programmer%');
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Funkcje agregujące

- COUNT() - zwraca liczbę wierszy
- AVG() - oblicza średnią arytmetyczną
- SUM() - sumuje wartości kolumn
- MIN() - znajduje wartość minimalną w kolumnie
- MAX() - znajduje wartość maksymalną w kolumnie

Funkcje agregujące (przykłady)

```
-- Ile mamy stanowisk pracy?
SELECT COUNT(*) FROM Jobs;

-- Ilu mamy programistów Pythona?
SELECT COUNT(*) FROM Workers
  WHERE Position = 2;

-- Ilu mamy pracowników na poszczególnych posadach?
SELECT J.Position, COUNT(*)
  FROM Jobs J, Workers W
  WHERE W.Position = J.ID
  GROUP BY J.Position;

/* Jaka jest suma godzin pracy wszystkich pracowników
   w ciągu tygodnia? */
SELECT (SUM(WorkHours) * 80) FROM Workers;

-- Jaki jest średni wymiar czasu pracy w firmie?
SELECT AVG(WorkHours), ' etatu' FROM Workers;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Sortowanie wyników

```
-- lista pracowników w kolejności rosnącej:
SELECT * FROM Workers ORDER BY FullName ASC;
-- lub:
SELECT * FROM Workers ORDER BY FullName;


-- lista pracowników w kolejności malejącej
SELECT * FROM Workers ORDER BY FullName DESC;


/* lista pracowników w kolejności rosnącej,
   pogrupowanych wg ich stanowiska pracy: */
SELECT * FROM Workers
  ORDER BY Position, FullName;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Łączenie tabel

```
/* Zwrócenie wartości występujących w pierwszej
   lub w drugiej tabeli: */
SELECT Column FROM TableOne
  UNION
  SELECT Column FROM TableTwo;


/* Zwrócenie wartości występujących w pierwszej
   tabeli z pominięciem wartości z drugiej tabeli: */
SELECT Column FROM TableOne
  EXCEPT
  SELECT Column FROM TableTwo;


/* Zwrócenie wartości wspólnych dla obu tabel: */
SELECT Column FROM TableOne
  INTERSECT
  SELECT Column FROM TableTwo;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia

- Złączenia wewnętrzne: w wyniku zwracane są wiersze z obu tabel; wiersze w jednej tabeli, które nie mają swoich odpowiedników w drugiej tabeli, są odrzucane
- Złączenia zewnętrzne: zwracane są wiersze z obu tabel (**lewej** i **prawej**), a w przypadku braku odpowiedników dodawane są także elementy puste
  - **Lewe złączenie zewnętrzne** (LEFT OUTER JOIN): zwracane są wszystkie wiersze z *lewej* tabeli; z prawej tabeli zwracane są wiersze odpowiadające tabeli lewej lub wartości puste
  - **Prawe złączenie zewnętrzne** (RIGHT OUTER JOIN): zwracane są wszystkie wiersze z *prawej* tabeli; z lewej tabeli zwracane są wiersze odpowiadające tabeli prawej lub wartości puste
  - **Pełne złączenie zewnętrzne** (FULL OUTER JOIN): zwracane są wszystkie wiersze z *obu* tabel (wynik operacji UNION)

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia wewnętrzne

```
-- zapis explicit
SELECT *
  FROM Tab1 INNER JOIN Tab2
  ON C11 = C21

-- zapis implicit
SELECT *
  FROM Tab1, Tab2
  WHERE C11 = C21
```

Tab1

C11	C12
1	2
2	4
5	7
6	9

Tab2

C21	C22
1	6
2	2
2	5
3	9

Wynik:

C11	C12	C21	C22
1	2	1	6
2	4	2	2
2	4	2	5

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia wewnętrzne

```
-- zapis explicit
SELECT *
  FROM Tab1 INNER JOIN Tab2
  ON C11 < C21

-- zapis implicit
SELECT *
  FROM Tab1, Tab2
  WHERE C11 < C21
```

Tab1

C11	C12
1	2
2	4
5	7
6	9

Tab2

C21	C22
1	6
2	2
2	5
3	9

Wynik:

C11	C12	C21	C22
1	2	2	2
1	2	2	5
1	2	3	9
2	4	3	9

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia wewnętrzne

```
/* Złączenie tabeli z samą
sobą (tzw. samozłączenie) */


-- zapis explicit
SELECT *
  FROM Tab AS T1 INNER JOIN
  Tab AS T2
    ON T1.C11 = T2.C13


-- zapis implicit
SELECT *
  FROM Tab T1, Tab T2
  WHERE T1.C11 = T2.C13
```

Tab

C11	C12	C13
1	2	1
2	4	2
5	7	2
6	9	3

Wynik

C11	C12	C13	C11	C12	C13
1	2	1	1	2	1
2	4	2	2	4	2
2	4	2	5	7	2

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia zewnętrzne

```
SELECT *
  FROM Tab1 LEFT JOIN Tab2
  ON C11 = C21
```

Tab1

C11	C12
1	2
2	4
5	7
6	9

Tab2

C21	C22
1	6
2	2
2	5
3	9

Wynik:

C11	C12	C21	C22
1	2	1	6
2	4	2	2
2	4	2	5
5	7	null	null
6	9	null	null

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia zewnętrzne

```
SELECT *
  FROM Tab1 RIGHT JOIN Tab2
  ON C11 = C21
```

Tab1

C11	C12
1	2
2	4
5	7
6	9

Tab2

C21	C22
1	6
2	2
2	5
3	9

Wynik:

C11	C12	C21	C22
1	2	1	6
2	4	2	2
2	4	2	5
null	null	3	9

GDAŃSK UNIVERSITY OF TECHNOLOGY

Złączenia zewnętrzne

```
SELECT *
  FROM Tab1 FULL JOIN Tab2
  ON C11 = C21
```

Tab1

C11	C12
1	2
2	4
5	7
6	9

Tab2

C21	C22
1	6
2	2
2	5
3	9

Wynik:

C11	C12	C21	C22
1	2	1	6
2	4	2	2
2	4	2	5
5	7	null	null
6	9	null	null
null	null	3	9

GDAŃSK UNIVERSITY OF TECHNOLOGY

Widoki

- Widoki są wirtualnymi tabelami, pozwalającymi na wypisywanie wybranych wierszy i kolumn pochodzących z rzeczywistych tabel w bazie danych
- Widoki nie zawierają własnych danych, a ich zawartość jest materializowana w razie potrzeby
- Widok może być używany w celu wstawiania nowych danych i ich aktualizacji, chyba że w jego definicji zawarto takie czynniki ograniczające jak:
  - wykorzystanie fraz GROUP BY, ORDER BY, DISTINCT czy funkcji agregujących w ramach instrukcji SELECT
  - pobieranie danych z więcej niż jednej tabeli
  - wykorzystanie podzapytań
- Jeśli widok jest aktualizowalny, wówczas przydatne może być wykorzystanie klauzuli *WITH CHECK OPTION*, która uniemożliwia dodanie do bazy takich danych, które nie spełniają wymogów narzuconych przez frazę WHERE w definicji widoku

GDAŃSK UNIVERSITY OF TECHNOLOGY

Widoki

```
/* Stworzenie widoku pozwalającego na dodanie do bazy
   informacji o pracownikach zatrudnionych
   na przynajmniej pół etatu. */

CREATE OR REPLACE VIEW FullTimeWorkers
  AS SELECT ID, FullName, WorkHours, Position
    FROM Workers WHERE WorkHours > 0.49
  WITH CHECK OPTION;

INSERT INTO FullTimeWorkers
  VALUES(10, 'New Worker', 1.0, 2);

SELECT * FROM FullTimeWorkers;
```

GDAŃSK UNIVERSITY OF TECHNOLOGY

Kursory

- Kursor to tymczasowa struktura pozwalająca na sekwencyjne pobieranie rekordów z bazy danych
  - w danym momencie wskazuje na pojedynczy wiersz z wynikowego zbioru danych
  - pozwala na przypisanie pobranych wartości zmiennym programowym
- Fazy użycia kursora:
  1. Deklaracja (DECLARE), pozwalająca na skojarzenie kursora z instrukcją SELECT
  2. Otwarcie (OPEN), polegające na wybraniu odpowiednich wierszy i ustawienie wstępnej pozycji
  3. Pobranie danych (FETCH) z aktualnego wiersza do zmiennych programowych oraz przejście do kolejnego wiersza
  4. Zamknięcie (CLOSE) kursora

GDAŃSK UNIVERSITY OF TECHNOLOGY

Indeksy

- Indeks jest strukturą pozwalającą na szybsze przeszukiwanie danych pod kątem konkretnej informacji
  - tworzona jest pomocnicza tabela przechowująca adresy poszczególnych wierszy, posortowanych wg wartości wskazanej kolumny
  - przydatny w sytuacji, gdy tabela jest często przetwarzana sekwencyjnie zgodnie z wartościami wybranej kolumny
  - wadą indeksacji jest zwiększenie czasu modyfikowania zindeksowanej tabeli (konieczność uaktualniania indeksu podczas dodawania, usuwania i aktualizowania wierszy) oraz zwiększenie rozmiaru bazy danych