

Moduł szkoleniowy SQL Structured Query Language

poziom podstawowy 15 godzinny

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Spis treści

1	Wprowadzenie	4
2	Cel	4
3	Opis sposobu realizacji celów	4
4	Treści kształcenia.....	5
5	Opis założonych osiągnięć ucznia	5
5.1	Korelacja z treściami podstawy programowej	5
6	Sposoby osiągania celów	6
6.1	Przykładowe tematy projektu:	6
6.2	Lista czynności w zależności od roli	7
7	Propozycje kryteriów oceny i metod sprawdzania osiągnięć ucznia	7
7.1	Opis założonych osiągnięć ucznia – przykłady wymagań na poszczególne oceny szkolne	7
7.2	Metody sprawdzania osiągnięć ucznia	8
8	Test końcowy sprawdzający wiedzę	8
9	Lekcje	10
9.1	Lekcja 1 - Bazy danych, podstawowe pojęcia (tabela, dana, wiersz, relacja, klucz podstawowy, klucz obcy, sortowanie, zapytanie, system zarządzania bazą danych), interakcje z bazą danych (Język SQL, aplikacja, formularz raport).....	10
9.1.1	Cel lekcji.....	10
9.1.2	Treść - slajdy z opisem	11
9.1.3	Ćwiczenia	27
9.1.4	Opis założonych osiągnięć ucznia	27
9.2	Lekcja 2 - Języki programowania serwerów SQL, środowisko programowania w języku SQL, wprowadzenie do języka SQL	27
9.2.1	Cel lekcji.....	27
9.2.2	Treść - slajdy z opisem	28
9.2.3	Ćwiczenia	39
9.2.4	Opis założonych osiągnięć ucznia	39
9.3	Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych.....	40
9.3.1	Cel lekcji.....	40
9.3.2	Treść - slajdy z opisem	40
9.3.3	Ćwiczenia	56
9.3.4	Opis założonych osiągnięć ucznia	56
9.4	Lekcja 4 – DML.....	56
9.4.1	Cel lekcji.....	56
9.4.2	Treść - slajdy z opisem	57
9.4.3	Ćwiczenia	66

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



9.4.4	Opis założonych osiągnięć ucznia	66
9.5	Lekcja 5 - DDL	66
9.5.1	Cel lekcji.....	66
9.5.2	Treść - slajdy z opisem	66
9.5.3	Ćwiczenia.....	80
9.5.4	Opis założonych osiągnięć ucznia	80
9.6	Lekcja 6 - DCL, uprawnienia, użytkownicy	80
9.6.1	Cel lekcji.....	80
9.6.2	Treść - slajdy z opisem	80
9.6.3	Ćwiczenia.....	96
9.6.4	Opis założonych osiągnięć ucznia	96
9.7	Lekcja 7 - Widoki i Funkcje.....	96
9.7.1	Cel lekcji.....	96
9.7.2	Treść - slajdy z opisem	96
9.7.3	Ćwiczenia.....	111
9.7.4	Opis założonych osiągnięć ucznia	111
9.8	Lekcja 8 - Procedury Triggery.....	111
9.8.1	Cel lekcji.....	111
9.8.2	Treść - slajdy z opisem	112
9.8.3	Ćwiczenia.....	125
9.8.4	Opis założonych osiągnięć ucznia	125
9.9	Lekcja 9 - Indeksy, constrainty	125
9.9.1	Cel lekcji.....	125
9.9.2	Treść - slajdy z opisem	126
9.9.3	Ćwiczenia.....	143
9.9.4	Opis założonych osiągnięć ucznia	143
9.10	Lekcja 10 - Transakcje, zatwierdzanie zmian, obsługa błędów, wprowadzenie do projektowania bazy danych.....	143
9.10.1	Cel lekcji.....	143
9.10.2	Treść - slajdy z opisem	143
9.10.3	Ćwiczenia.....	156
9.10.4	Opis założonych osiągnięć ucznia	156
9.11	Lekcja 11 – Interdyscyplinarny projekt książki teleadresowej	156
9.11.1	Cel lekcji.....	156
9.11.2	Treść - slajdy z opisem	157
9.11.3	Opis założonych osiągnięć ucznia	161

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY



1 Wprowadzenie

Język SQL ("Structure Query Language") to strukturalny język zapytań służący do budowania systemów obsługi relacyjnych baz danych (RDBMS) na dowolnej platformie sprzętowej. Obecnie jest on standardowym językiem służącym do obsługi komercyjnych baz danych.

SQL został opracowany w latach 70. w firmie IBM. Szybko stał się standardem w komunikacji z serwerami baz danych. Wiele współczesnych systemów relacyjnych baz danych używa do komunikacji z użytkownikiem SQL, dlatego potocznie mówi się, że korzystanie z relacyjnych baz danych to korzystanie z SQL-a.

Pionierską firmą, która rozpoczęła dystrybucję komercyjnych baz danych, był Oracle. Dalsze wprowadzanie SQL-a, w produktach innych firm, wiązało się nierozłącznie z wprowadzaniem modyfikacji pierwotnego języka.

Pierwotną nazwą języka miał być SEQUEL, jednakże okazało się, że nazwa ta była już zastrzeżona przez brytyjską wytwórnię lotniczą Hawker Siddeley.

Język SQL umożliwia zadawanie pytań do baz danych oraz pozwala na zmienianie struktury baz danych. W opracowaniu tym omawiamy tylko ten fragment języka który służy do zadawania pytań.

Systemy informatyczne od zawsze budowane były z myślą o przetwarzaniu elementarnych informacji, które odpowiednio zamodelowane formułowały odpowiedzi na pytania naukowców, analityków, przedsiębiorców biznesu. Bazy danych są zatem nieodłącznym elementem każdego systemu informatycznego. Żadna, nawet najbardziej użyteczna i wysokowydajna technologia, nie odniesie sukcesu, jeżeli nie będzie dostarczać informacji i wskazówek dla końcowego użytkownika. Dlatego znajomość języka manipulacji na rekordach baz danych – SQL kreuje się na pozycję obowiązkową w umiejętnościach każdej osoby, związanej z branżą IT.

W poszczególnych częściach kursu szczegółowo zostaną opisane podstawowe elementy składni języka zapytań SQL. Zapoznanie się z ich treścią pozwoli na swobodną pracę z tymże językiem w ramach dowolnego systemu zarządzania bazą danych.

2 Cel

Poznanie zagadnień z zakresu relacyjnych baz danych w przykładowym profesjonalnym środowisku relacyjnego systemu zarządzania bazą danych, jakim jest system produkcyjny Microsoft SQL Server 2008. Praktyczne zapoznanie się ze środowiskiem. Przyswojenie dedykowanej terminologii w języku polskim i angielskim. Nabycie umiejętności manipulacji zarówno bazami danych, ich strukturami, jak i przechowywanymi w nich danymi za pomocą języka SQL.

3 Opis sposobu realizacji celów

10 półtoragodzinnych lekcji składających się z przypomnienia wiedzy z poprzedniej lekcji, przedstawienia materiału wraz z przykładami, ćwiczeń praktycznych. Po zakończeniu całego cyklu - przeprowadzenie egzaminu sprawdzającego wiedzę.

Dodatkowo wprowadzono lekcję 11 zawierającą część projektu interdyscyplinarnego łączącego 4 dziedziny wiedzy z zakresu informatyki (SQL, PHP, JavaScript oraz JAVA) w zakresie zaprojektowania bazy danych oraz obiektów służących jej obsłudze.

Człowiek - najlepsza inwestycja



4 Treści kształcenia

Treść kursu została podzielona na 10 bloków tematycznych – po jednym do każdej lekcji:

1. Bazy danych, podstawowe pojęcia (tabela, dana, wiersz, relacja, klucz podstawowy, klucz obcy, sortowanie, zapytanie, system zarządzania bazą danych), interakcje z bazą danych (Język SQL, aplikacja, formularz raport)
2. Języki programowania serwerów SQL, środowisko programowania w języku SQL, wprowadzenie do języka SQL
3. Złożone zapytania, grupowanie, agregacja danych
4. DDL
5. DML
6. DCL, uprawnienia, użytkownicy
7. Widoki i Funkcje
8. Procedury i wyzwalacze (triggers)
9. Indeksy i ograniczenia (constraints)
10. Transakcje, zatwierdzanie zmian, obsługa błędów, normalizacja bazy danych
11. Projekt interdyscyplinarny – baza teleadresowa kolegów i koleżanek ze szkoły

5 Opis założonych osiągnięć ucznia

Uczeń po odbyciu kursu pozna podstawy teoretyczne związane z bazami danych, pozna techniki projektowania baz danych, nauczy się praktycznego programowania baz danych w środowisku Microsoft SQL Serwer 2008.

5.1 Korelacja z treściami podstawy programowej

Cele kształcenia	Treść kształcenia	Podstawa programowa
Uczeń poznaje podstawowe typy danych	Lekcja 5	Uczeń korzysta z wbudowanych typów danych
Uczeń tworzy procedury, funkcje, triggerzy oraz constrainty	Lekcje 8 i 9	Uczeń tworzy własne typy danych
Uczeń poznaje transakcje bazodanowe Uczeń potrafi obsługiwać błędy Uczeń prawidłowo wykonuje zapytania SQL Uczeń potrafi modyfikować dane Uczeń potrafi modyfikować struktury danych Uczeń tworzy i modyfikuje obiekty bazy danych	Lekcje: 3, 4, 5, 6, 7, 8, 9	Uczeń przestrzega zasad programowania
Uczeń tworzy widoki, procedury, funkcje, triggerzy oraz constrainty	Lekcje 8 i 9	Uczeń stosuje instrukcje, funkcje, procedury, obiekty, metody wybranych języków programowania
Uczeń tworzy widoki, procedury, funkcje, triggerzy oraz constrainty	Lekcje 8 i 9	Uczeń tworzy własne funkcje, procedury, obiekty, metody wybranych języków programowania
Uczeń potrafi zainstalować oraz poprawnie	Lekcje: 2	Uczeń wykorzystuje środowisko

Człowiek - najlepsza inwestycja

Cele kształcenia	Treść kształcenia	Podstawa programowa
skonfigurować środowisko pracy		programistyczne: edytor, kompilator, debugger
Uczeń projektuje bazy danych dla aplikacji (w tym webowych)	Lekcje: 10	Uczeń wykorzystuje języki programowania do tworzenia aplikacji internetowych realizujących zadania po stronie serwera
Uczeń poznaje mechanizmy tworzenia i modyfikacji danych	Wszystkie lekcje modułu	Uczeń pobiera dane aplikacji i przechowuje je w bazie danych
Uczeń potrafi testować zapytania Uczeń poznaje instrukcje modyfikujące obiekty bazodanowe	Lekcje: 2, 3, 4, 5, 6, 7, 8, 9	Uczeń testuje tworzoną aplikację i modyfikuje jej kod źródłowy
Uczeń potrafi komentować kod	Lekcje: 3	Uczeń dokumentuje tworzoną aplikację
Uczeń poznaje podstawy projektowania baz danych Uczeń przenosi bazy danych	Lekcje: 1,2, 10	Uczeń zamieszcza opracowane aplikacje w Internecie
Uczeń poznaje zagadnienia dotyczące bezpieczeństwa, transakcji oraz obsługi błędów	Lekcje: 10	Uczeń zabezpiecza dostęp do tworzonych aplikacji

6 Sposoby osiągnięcia celów

Po odbyciu kursu uczeń powinien wykonać projekt/zadanie, które zmotywuje go do pracy indywidualnej ze środowiskiem programistycznym i bazą danych. Zadanie utrwali zdobytą na kursie wiedzę i zmusi do wykorzystania zdobytej wiedzy teoretycznej w praktyce. Zadanie powinno być sformułowane w sposób otwarty, tak aby każdy z uczniów mógł wybrać coś dla niego interesującego. Powinna też zostać dostarczona lista możliwych tematów projektu do wyboru - dla osób, które nie będą miały pomysłu na projekt.

6.1 Przykładowe tematy projektu:

1. Projekt bazy danych placówki medycznej, z możliwością zapisów na badania (główne tabele: pacjent, lekarz, badanie, wizyta). Opcjonalnie: procedura składowana służąca do zapisów na badanie
2. Projekt bazy danych działu kadr w firmie (główne tabele: pracownik, dział, wypłaty). Opcjonalnie: procedura składowana służąca wyliczenia pensji na podstawie ilości przepracowanych godzin
3. Projekt bazy danych rezerwacji biletów w kinie (główne tabele: sala, film, seans, bilet). Opcjonalnie: procedura składowana służąca rezerwacji biletów

Człowiek - najlepsza inwestycja



6.2 Lista czynności w zależności od roli

Wykonywane czynności są takie same dla każdej z lekcji, jedynie materiał którego dotyczą jest różny. Może się zdarzyć iż w przypadku barku materiału w danej lekcji dla danej czynności, czynność nie powinna być wykonana.

Czynności nauczyciela	Czynności ucznia
Prezentowanie przygotowanego materiału.	Sporządzanie notatek z prezentowanego materiału
Zadawanie pytań kontrolnych odnośnie już pokazanego materiału	Udzielanie odpowiedzi na pytania nauczyciela
Zadawanie pytań odnośnie materiału, który dopiero będzie pokazany, jeśli możliwe jest uzyskanie odpowiedzi w drodze dedukcji	
Objaśnianie ćwiczeń lekcyjnym, pilnowanie czasu ich wykonania przez uczniów, udzielanie wskazówek	Rozwiązywanie zadań lekcyjnych
Objaśnienie zadań na koniec lekcji, udzielanie wskazówek podczas rozwiązywania, zlecenie dokończenia zadań jako pracę domową	Odrabianie zadań domowych
Sprawdzanie ćwiczeń lekcyjnych i zadań domowych: należy zwrócić uwagę na poprawne formatowanie kodu (wcięcia) poprawne użycie konstrukcji języka dokumentowanie kodu komentarzami otrzymywanie poprawnych rezultatów	
Zapisywanie uwag odnośnie materiałów szkoleniowych i sugestie ewentualnych korekt.	

7 Propozycje kryteriów oceny i metod sprawdzania osiągnięć ucznia

7.1 Opis założonych osiągnięć ucznia – przykłady wymagań na poszczególne oceny szkolne

Ocenę **dopuszczającą (2)** otrzymuje uczeń, który potrafi skonstruować proste, niezłożone zapytania SQL: SELECT, INSERT i UPDATE, zna podstawowe pojęcia związane z bazami danych i silnikami baz danych, potrafi własnymi słowami wyjaśnić podstawowe instrukcje języka.

Ocenę **dostateczną (3)** otrzymuje uczeń, który spełnia wymagania oceny dopuszczającej oraz potrafi stworzyć zapytania SQL zawierające kilka warunków. Zna podstawowe struktury danych w języku SQL. Wykonuje ćwiczenia z niewielką pomocą nauczyciela. Potrafi wskazać podstawowe zastosowania baz danych.

Ocenę **dobrą (4)** otrzymuje uczeń, który spełnia wymagania na ocenę dostateczną oraz potrafi stworzyć skrypt złożony z kilku zapytań. Potrafi łączyć tabele za pomocą instrukcji JOIN. Uczeń powinien być w stanie stworzyć własne obiekty baz danych takie jak widoki i funkcje. Na lekcjach wykonuje samodzielnie zadane ćwiczenia. Umiejętnie korzysta z mechanizmów pozwalających na dbanie o jednorodność i spójność danych (np. klucze i ograniczenia).

Ocenę **bardzo dobrą (5)** otrzymuje uczeń, który spełnia wymagania na ocenę dobrą oraz bardzo dobrze zna teorię omawianą na zajęciach, w trakcie wykonywania ćwiczeń wykazuje inicjatywę, potrafi wskazać kilka rozwiązań zadania, tworzy skrypty języka z wykorzystaniem dobrych praktyk

Człowiek - najlepsza inwestycja

programistycznych. Uczeń potrafi zaprojektować bazę danych poprawną pod względem normalizacji, relacji i ograniczeń kolumn.

Ocenę **celującą (6)** otrzymuje uczeń, którego wiedza wykracza poza omawiany zakres. Np. potrafi stworzyć skomplikowaną procedurę składowaną zawierającą kursor. Tworzy zaawansowane obiekty baz danych takie jak procedury, triggerzy, funkcje skalarne i tablicowe.

7.2 Metody sprawdzania osiągnięć ucznia

Ocena zaliczenia przedmiotu powinna być składową dwóch ocen: testu oraz projektu (zaprojektowanej bazy danych).

Kryteria oceny testu:

- 0-7 poprawnych odpowiedzi - 1
- 8-9 poprawnych odpowiedzi - 2
- 10-11 poprawnych odpowiedzi - 3
- 12-13 poprawnych odpowiedzi - 4
- 14-15 poprawnych odpowiedzi - 5

Kryteria oceny projektu:

- Zaprojektowana baza jest logicznie poprawna, występują relacje, typy danych - 3
- W zaprojektowanej bazie widać próbę normalizacji, zaznaczone są klucze podstawowe i obce, występuje logiczna konwencja nazewnictwa kolumn i tabel - 4
- Poza poprawnym projektem bazy danych uczeń napisał poprawną procedurę składowaną - 5

8 Test końcowy sprawdzający wiedzę

15 zadań na 30 minutowy test sprawdzający wiedzę. Pogrubioną czcionką zaznaczono prawidłowe odpowiedzi.

1. Argumentem funkcji MAX() może być kolumna, w której znajdują się dane typu:
 - a. Liczby naturalne
 - b. Ciągi znaków
 - c. Data
 - d. Typ money
2. Które z wymienionych funkcji agregujących są zdefiniowane w języku SQL?
 - a. AVG()
 - b. MODE()
 - c. MEDIANA()
 - d. SUMA()
3. W bazie danych znajdują się tabele:

Film (id, tytuł, rok, id_reżyser, producent)

Reżyser (id, imię, nazwisko, rok_urodzenia, narodowosc)

W wyniku wykonania zapytania:

SELECT tytuł, rok FROM Film WHERE (rok != 1984 OR rok != 2011) AND

id_reżyser = (SELECT id FROM Reżyser WHERE rok_urodzenia >= 1949 OR rok_urodzenia <= 1951)

wyświetlone zostaną:

- a. Wszystkie filmy (tytuł i rok produkcji) z tabeli Film.
- b. Wszystkie filmy (tytuł i rok produkcji), które powstały w innych latach niż 1984 i 2011, oraz których reżyserowie urodzili się pomiędzy rokiem 1949 i 1951.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



- c. Wszystkie filmy (tytuł i rok produkcji), które powstały w innych latach niż 1984 i 2011, oraz których reżyserowie urodzili się pomiędzy rokiem 1949 i 1951, uszeregowane alfabetycznie.
 - d. Żadne filmy nie zostaną wyświetlone.
4. Za pomocą polecenia JOIN możemy łączyć:
 - a. Różne tabele ze sobą
 - b. Tabelę z widokiem
 - c. Tabelę z indeksem
 - d. Tabelę z samą sobą
5. Najczęściej używanym zapytaniem w tabeli tbl jest:
 SELECT id, a, b FROM tbl WHERE a = @a AND b = @b;
 Jaki indeks powinien zostać założony na tej tabeli
 - a. CREATE INDEX tbl_idx ON a (tbl);
 - b. CREATE INDEX tbl_idx ON b (tbl);
 - c. CREATE INDEX tbl_idx ON tbl (a, b);
 - d. CREATE INDEX tbl_idx ON tbl (a);
6. Którym poleceniem można zamienić "Kowalski" na "Nowak" w kolumnie „Nazwisko” w tabeli Osoby?
 - a. MODIFY Osoby SET Nazwisko= 'Nowak' WHERE Nazwisko ='Kowalski'
 - b. UPDATE Osoby SET Nazwisko ='Kowalski' INTO Nazwisko ='Nowak'
 - c. MODIFY Osoby SET Nazwisko ='Kowalski' INTO Nazwisko ='Nowak'
 - d. UPDATE Osoby SET Nazwisko ='Nowak' WHERE Nazwisko ='Kowalski'
7. Za pomocą SQL, dodaj do tabeli Osoby wiersz, gdzie nazwiskiem będzie "Olsen"?
 - a. INSERT ('Olsen') INTO Osoby (Nazwisko)
 - b. INSERT INTO Osoby (Nazwisko) VALUES ('Olsen')
 - c. INSERT 'Olsen' INTO Osoby (Nazwisko)
 - d. INSERT INTO Osoby ('Olsen') INTO Nazwisko
8. Jakie zapytanie zwróci wszystkie rekordy z tabeli Osoby, których nazwisko zaczyna się na literę "a"?
 - a. SELECT * FROM Osoby WHERE Nazwisko='a'
 - b. SELECT * FROM Osoby WHERE Nazwisko ='%a%'
 - c. SELECT * FROM Osoby WHERE Nazwisko LIKE 'a%'
 - d. SELECT * FROM Osoby WHERE Nazwisko LIKE '%a'
9. Klauzula TOP
 - a. Musi być umieszczona na końcu zapytania.
 - b. Służy do wyboru rekordów o największej wartości we wskazanej kolumnie.
 - c. Jest niedostępna w systemie MSSQL.
 - d. Pozwala ograniczyć liczbę wyświetlanych wyników.
10. Złączenia tabel w języku SQL
 - a. Są realizowane za pomocą instrukcji MERGE
 - b. Powodują trwałe zmiany w strukturze bazy danych
 - c. Można wykonywać na dowolnej liczbie tabel (większej niż 2)
 - d. Są realizowane za pomocą operatora „;” (średnik)
11. W której postaci normalnej znajduje się poniższa tabela

Imię	Nazwisko	Adres	Pesel
Jan	Kowalski	Ul. Mała 11, 60-003 Poznań	82111112345
Joanna	Kowalska	Ul. Długa 9a, 00-103 Warszawa	78010112345

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY



- a. Tabela nie jest znormalizowana
 - b. W pierwszej postaci normalnej
 - c. W drugiej postaci normalnej
 - d. W trzeciej postaci normalnej
12. UPDATE jest częścią języka
- a. DDL
 - b. DML
 - c. DQL
 - d. Żadnego z powyższych
13. Która z poniższych definicji widoku jest prawidłowa?
- a. CREATE VIEW Widok As SELECT imie, nazwisko, pesel FROM pracownicy
 - b. CREATE VIEW Widok SELECT * FROM pracownicy
 - c. CREATE VIEW Widok FROM SELECT * FROM pracownicy
 - d. CREATE VIEW Widok As imie, nazwisko, pesel FROM pracownicy
14. Do czego służy polecenie:
- SELECT imie, nazwisko, pesel INTO pracownicy FROM klienci
- a. Zapytanie wyświetla imię, nazwisko i pesel z tabeli pracownicy
 - b. Za pytanie wyświetla imię, nazwisko i pesel z tabeli klienci
 - c. Tworzy tabelę klienci
 - d. Zapytanie używane przy tworzeniu widoków – na bazie tabeli klienci tworzy widok pracownicy
15. Które rozszerzenie języka SQL wykorzystywane jest w Microsoft SQL Server
- a. PL SQL
 - b. PLPsg SQL
 - c. T SQL
 - d. MS SQL

9 Lekcje

9.1 Lekcja 1 - Bazy danych, podstawowe pojęcia (tabela, dana, wiersz, relacja, klucz podstawowy, klucz obcy, sortowanie, zapytanie, system zarządzania bazą danych), interakcje z bazą danych (Język SQL, aplikacja, formularz raport)

9.1.1 Cel lekcji




Celem lekcji jest wyjaśnienie, czym jest baza danych. Nauczyciel zaprezentuje przykłady bazy danych w życiu codziennym (książka telefoniczna, system rezerwacji biletów lotniczych, system aukcji elektronicznych) – tak, aby uczniowie zdali sobie sprawę, że bazy danych są obecne w ich życiu. Starając się zainteresować tematem nauczyciel zapyta, w jakich bazach danych znajdują się dane uczniów (eWUŚ, PESEL, NFZ, elektroniczny dzienniczek ucznia).

W lekcji zdefiniowane zostaną podstawowe pojęcia dotyczące baz danych: baza danych, system zarządzania bazą danych, tabela, dana, wiersz, relacja, klucz podstawowy, klucz obcy, sortowanie, zapytanie. W ramach przykładów zaprezentowane zostaną proste schematy bazy danych, np. książki telefonicznej. Na ich przykładzie nauczyciel zademonstruje zdefiniowane pojęcia.






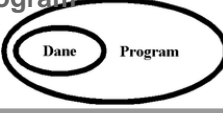



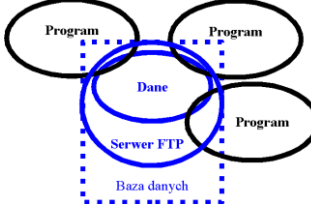

Człowiek - najlepsza inwestycja



9.1.2 Treść - slajdy z opisem

<p>Slajd 1</p>		<p>SQL jest najpopularniejszym językiem programowania baz danych. W tej lekcji wyjaśnimy sobie czym jest baza danych oraz objaśnimy podstawowe pojęcia dotyczące baz danych.</p>
<p>Slajd 2</p>		<p>Pytania do uczniów: jakie znacie przykłady baz danych? W jakich bazach znajdują się Wasze dane?</p>
<p>Slajd 3</p>		<p>W czasie kiedy komputery nie były jeszcze znane tak powszechnie jak dzisiaj, wszystkie informacje były gromadzone na papierze. Pracownicy firmy musieli ręcznie organizować dane. Zapisywać je, wyszukiwać, aktualizować itd. Dzisiaj, oprócz operacji, które muszą wykonać ręcznie, pozostałe wykonuje za pomocą funkcji, które dostarcza relacyjny system bazy danych.</p>

Człowiek - najlepsza inwestycja

<p>Slajd 4</p>	  <p>SQL Structured Query Language</p> <h3>Definicja bazy danych</h3> <ul style="list-style-type: none"> • zbiór informacji wraz z możliwością łatwego dostępu oraz ich zmiany  <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Najogólniej rzecz biorąc, „baza danych” to zbiór informacji wraz z możliwością łatwego dostępu oraz ich zmiany (tj. modyfikacją, dodawaniem nowych i usuwaniem starych) z poziomu aplikacji z niej korzystającej. Często używamy sformułowania „Używam bazy danych”, co dokładniej oznacza - „korzystam ze zbioru informacji, który łatwo odczytywać i zmieniać”.</p>
<p>Slajd 5</p>	  <p>SQL Structured Query Language</p> <h3>Elektroniczne bazy danych</h3> <h4>Pliki w programie</h4> <ul style="list-style-type: none"> • Dane dostępne tylko dla programu • Problem z wieloma operacjami jednocześnie • Brak możliwości skorzystania z danych przez inny program   <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Pytanie do uczniów: czy plik tekstowy z nazwiskami i numerem PESEL jest bazą danych?</p> <p>Plik tekstowy można uznać za bazę danych jeśli odczytanie i modyfikację zapisanych w nim informacji w konkretnym zastosowaniu uznamy za łatwe. Przykładowo niech program wyświetla krótki komunikat, który powinien mieć możliwość dowolnej zmiany. Zastosowanie wtedy zwykłego pliku z tekstem jest jak najbardziej dobrym rozwiązaniem i można go nazwać „bazą danych”.</p> <p>Rysunek na slajdzie przedstawia sytuację, gdzie program korzysta z pliku tekstowego, który jest jego bazą. Wiąże się to jednak z ograniczeniami (wymienione na slajdzie).</p>
<p>Slajd 6</p>	  <p>SQL Structured Query Language</p> <h3>Elektroniczne bazy danych</h3> <h4>Pliki dostępne dla wielu programów</h4> <ul style="list-style-type: none"> • Pliki na serwerze np. FTP • Wiele aplikacji korzystających z plików • Zarządzaniem dostępem   <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>W takim razie czy plik na serwerze (np. FTP), z którego mogą korzystać różne programy jest już „prawdziwą bazą danych”? Mogą z niego korzystać różne programy, lub praktycznie każdy, kto ma dostęp do serwera.</p> <p>W przedstawionej sytuacji nie ma mechanizmu zapewniającego zarządzania dostępem do danych.</p>

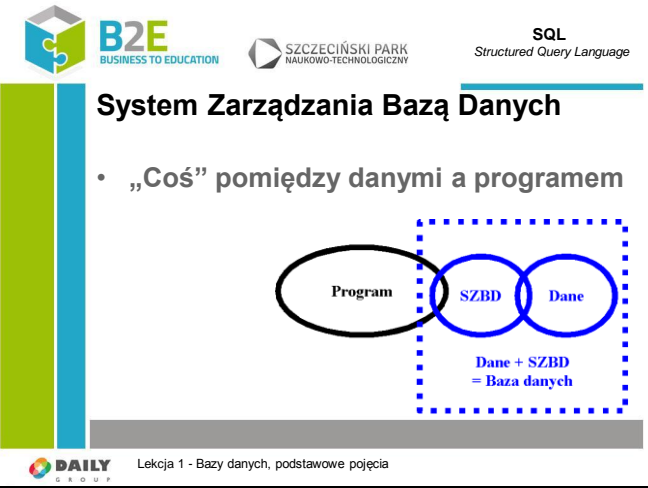
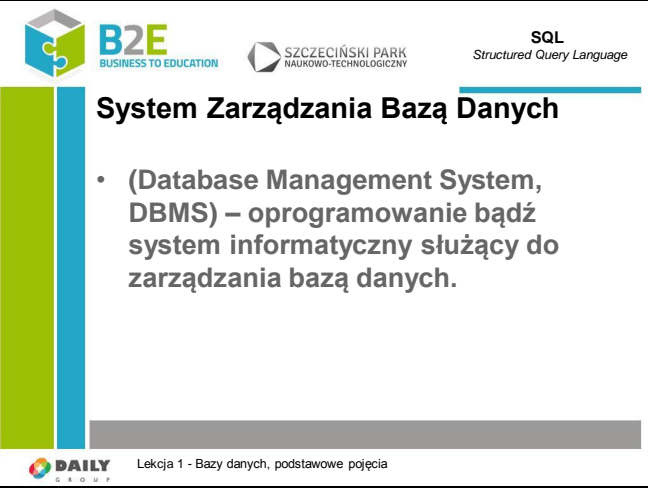
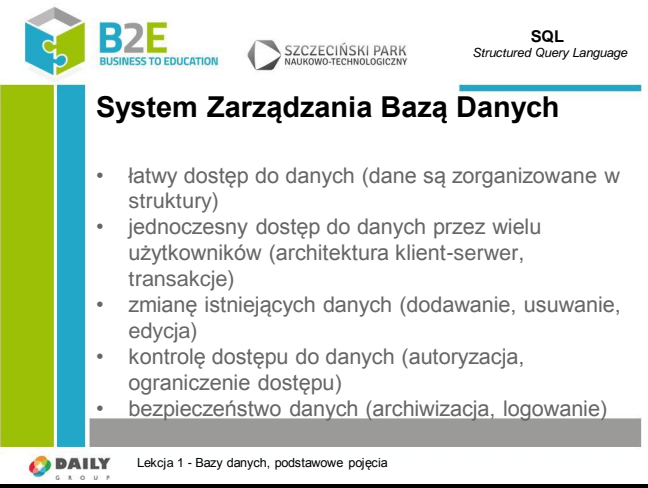
Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 7</p>	 <p>System Zarządzania Bazą Danych</p> <ul style="list-style-type: none"> „Coś” pomiędzy danymi a programem <p>Program, SZBD, Dane, Dane + SZBD = Baza danych</p> <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Brakującym elementem jest System Zarządzania Bazą Danych, który wraz z danymi stanowi BAZĘ DANYCH</p>
<p>Slajd 8</p>	 <p>System Zarządzania Bazą Danych</p> <ul style="list-style-type: none"> (Database Management System, DBMS) – oprogramowanie bądź system informatyczny służący do zarządzania bazą danych. <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Definicja SZDB</p>
<p>Slajd 9</p>	 <p>System Zarządzania Bazą Danych</p> <ul style="list-style-type: none"> łatwy dostęp do danych (dane są zorganizowane w struktury) jednoczesny dostęp do danych przez wielu użytkowników (architektura klient-serwer, transakcje) zmianę istniejących danych (dodawanie, usuwanie, edycja) kontrolę dostępu do danych (autoryzacja, ograniczenie dostępu) bezpieczeństwo danych (archiwizacja, logowanie) <p>Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Cechy, określające system zarządzania bazą danych</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
10



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Systemy Zarządzania Bazą Danych

- Przykłady:
 - Oracle
 - IBM DB2
 - Microsoft SQL Server
 - Firebird
 - MySQL
 - PostgreSQL
 - Sybase







DAILY GROUP


Lekcja 1 - Bazy danych, podstawowe pojęcia

Pytanie do uczniów: czy znacie przykłady SZBD?

Slajd
11



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Relacyjny System Bazy Danych

Działy

id_działu	nazwa	adres
1	Marketing	Warszawa, Słoneczna 3
2	Księgowość	Warszawa, Słoneczna 3
3	Administracja IT	Warszawa, Słoneczna 3a
4	Programiści	Bydgoszcz, Kwiatowa 16

Projekty

p_id	szef_projektu	nazwa	termin_oddania
it1	2	Upgrade sprzętu	10.01.2009
p1	5	Program magazynowy	1.08.2009
p2	8	Statistica	20.02.2009

Pracownicy

id	imie	nazwisko	id_działu	pensja	projekt
1	Jan	Kowalski	2	2600	
2	Adam	Nowak	3	3100	it1
3	Halina	Szańska	4	1800	
4	Anna	Ryś	3	3000	
5	Piotr	Lis	4	2000	p1
6	Paweł	Lis	4	1800	p2
7	Jan	Nowikowski	1	2200	it1
8	Adam	Kot	4	1900	p2

DAILY GROUP

Lekcja 1 - Bazy danych, podstawowe pojęcia

Relacyjny system baz danych przechowuje wszystkie dane w tabelach. Każda tabela zawiera dane na konkretny temat, np. dane o klientach, pracownikach, towarach itp. System bazy danych zarządza tymi informacjami, pozwala m.in. na szybsze ich wyszukiwanie i zorganizowanie.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
12



B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Relacyjny System Bazy Danych

- W bazach relacyjnych wiele tablic danych może współpracować ze sobą
- Wszystkie dane w bazie relacyjnej przedstawiane są w formie dwuwymiarowych tabel
- Wszystkie wartości danych oparte są na typach danych
- Możliwe jest porównywanie wartości z różnych kolumn, tabel
- Wszystkie operacje wykonywane są w oparciu o algebrę relacji
- Przez brak możliwości identyfikacji wiersza przez jego pozycję niezbędna jest jedna lub więcej kolumn niepowtarzalnych



Lekcja 1 - Bazy danych, podstawowe pojęcia

W bazach relacyjnych wiele tablic danych pozostaje w relacjach ze sobą (są między sobą powiązane). Bazy relacyjne posiadają wewnętrzne języki programowania, wykorzystujące zwykle SQL do operowania na danych, za pomocą których tworzone są zaawansowane funkcje obsługi danych. Relacyjne bazy danych (jak również przeznaczony dla nich standard SQL) oparte są na kilku prostych zasadach:

1. Wszystkie wartości danych oparte są na typach danych.
2. Wszystkie dane w bazie relacyjnej przedstawiane są w formie dwuwymiarowych tabel (w matematycznym żargonie noszących nazwę „relacji”). Każda tabela zawiera zero lub więcej wierszy (w tymże żargonie – „krotki”) i jedną lub więcej kolumn („atrybutów”). Na każdy wiersz składają się jednakowo ułożone kolumny wypełnione wartościami, które z kolei w każdym wierszu mogą być inne.
3. Po wprowadzeniu danych do bazy, możliwe jest porównywanie wartości z różnych kolumn, zazwyczaj również z różnych tabel, i scalanie wierszy, gdy pochodzące z nich wartości są zgodne. Umożliwia to wiązanie danych i wykonywanie stosunkowo złożonych operacji w granicach całej bazy danych.
4. Wszystkie operacje wykonywane są w oparciu o algebrę relacji, bez względu na położenie wiersza tabeli. Nie można więc zapytać o wiersze, gdzie (x=3) bez wiersza pierwszego, trzeciego i piątego. Wiersze w relacyjnej bazie danych przechowywane są w porządku zupełnie dowolnym – nie musi on odzwierciedlać ani kolejności ich wprowadzania, ani kolejności ich przechowywania.
5. Z braku możliwości identyfikacji wiersza przez jego pozycję pojawia się potrzeba obecności jednej lub więcej kolumn niepowtarzalnych w granicach całej tabeli, pozwalających odnaleźć konkretny wiersz. Kolumny te określa się jako „klucz podstawowy” (ang. primary key) tabeli.

Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY




Slajd
13



B2E

BUSINESS TO EDUCATION



SQL
Structured Query Language

Podstawowe pojęcia

Tabela

id	imie	nazwisko	id_dzialu	pensja	projekt
1	Jan	Kowalski	2	2600	
2	Adam	Nowak	3	3100	it1
3	Halina	Szańska	4	1800	
4	Anna	Ryś	3	3000	
5	Piotr	Lis	4	2000	p1
6	Paweł	Lis	4	1800	p2
7	Jan	Nowikowski	1	2200	it1
8	Adam	Kot	4	1900	p2



 Lekcja 1 - Bazy danych, podstawowe pojęcia


Tabela składa się z wierszy i kolumn. Wiersze w tabeli są przechowywane w porządku zależnym od implementacji bazy danych. Dla każdego wiersza każda z kolumn posiada jedno pole z wartością. Wszystkie wartości w kolumnie są tego samego typu.

Slajd
14



B2E

BUSINESS TO EDUCATION




SQL
Structured Query Language

Podstawowe pojęcia


Dana

id	imie	nazwisko	id_dzialu	pensja	projekt
1	Jan	Kowalski	2	2600	
2	Adam	Nowak	3	3100	it1
3	Halina	Szańska	4	1800	
4	Anna	Ryś	3	3000	
5	Piotr	Lis	4	2000	p1
6	Paweł	Lis	4	1800	p2
7	Jan	Nowikowski	1	2200	it1
8	Adam	Kot	4	1900	p2

 Lekcja 1 - Bazy danych, podstawowe pojęcia


Dana to pojedynczy wpis w tabeli – jak komórka w MS Excel. Często stosowanym pojęciem jest atrybut.

Slajd
15



B2E

BUSINESS TO EDUCATION




SQL
Structured Query Language

Podstawowe pojęcia

Wiersz

id	imie	nazwisko	id_dzialu	pensja	projekt
1	Jan	Kowalski	2	2600	
2	Adam	Nowak	3	3100	it1
3	Halina	Szańska	4	1800	
4	Anna	Ryś	3	3000	
5	Piotr	Lis	4	2000	p1
6	Paweł	Lis	4	1800	p2
7	Jan	Nowikowski	1	2200	it1
8	Adam	Kot	4	1900	p2

 Lekcja 1 - Bazy danych, podstawowe pojęcia

Pojedynczy wiersz tabeli nazywany jest rekordem (lub encją) i stanowi najczęściej zbiór danych o pojedynczym obiekcie (ew. grupie obiektów).

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
16

SQL
Structured Query Language

Podstawowe pojęcia

- Kolumna**

Pracownicy

id	imie	nazwisko	id_dzialu	pensja	projekt
1	Jan	Kowalski	2	2600	
2	Adam	Nowak	3	3100	it1
3	Halina	Szańska	4	1800	
4	Anna	Ryś	3	3000	
5	Piotr	Lis	4	2000	p1
6	Paweł	Lis	4	1800	p2
7	Jan	Nowikowski	1	2200	it1
8	Adam	Kot	4	1900	p2

DAILY GROUP Lekcja 1 - Bazy danych, podstawowe pojęcia

W relacyjnym modelu baz danych i podobnych, kolumny stanowią zwykle atrybuty jakiegoś obiektu (np. wielkość, grubość, tytuł, nazwisko) i stąd dane zawarte w kolumnach mają najczęściej jeden określony typ. Dodatkowo w bazach obsługiwanych przez język SQL kolumnom nadawane są nazwy, także poza etapem projektowym i nazwy te są unikatowe w obrębie jednej tabeli.

Slajd
17

SQL
Structured Query Language

Podstawowe pojęcia

- Relacja**

DAILY GROUP Lekcja 1 - Bazy danych, podstawowe pojęcia

Każda relacja (prezentowana w postaci np. tabeli) posiada unikatową nazwę, nagłówek i zawartość. Nagłówek relacji to zbiór atrybutów, gdzie atrybut jest parą nazwa_atrybutu: nazwa_typu, zawartość natomiast jest zbiorem krotek (reprezentowanych najczęściej w postaci wiersza w tabeli). W związku z tym, że nagłówek jest zbiorem atrybutów nie jest ważna ich kolejność. Atrybuty zazwyczaj

Człowiek - najlepsza inwestycja

utożsamiane są z kolumnami tabeli. Każda krotka (wiersz) wyznacza zależność pomiędzy danymi w poszczególnych komórkach (np. osoba o danym numerze PESEL posiada podane nazwisko i imię oraz adres).

Slajd
18



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Podstawowe pojęcia

• Rodzaje relacji

Zamówienia : Tabela		
ID zamówienia	ID klienta	ID pracownika
10000	FRANS	6
10001	MEREP	8
10002	FOLKO	3

W jednym zamówieniu może być wiele produktów...

Opisy zamówień : Tabela		
ID zamówienia	ID produktu	Cena jednostk
10001	25	14,00 zł
10001	40	12,80 zł
10001	59	38,50 zł
10002	25	14,00 zł

...i każdy produkt może się pojawić w wielu zamówieniach.

Produkty : Tabela		
ID produktu	Nazwa produktu	Cena jednostkowa
25	NuNuCa Nuß-Nougat Creme	14,00 zł
26	Gumbär Gummibärchen	31,23 zł



Lekcja 1 - Bazy danych, podstawowe pojęcia

1. relacja jeden-do-jednego

W relacji **jeden-do-jednego** każdy rekord w tabeli A może mieć tylko jeden dopasowany rekord z tabeli B, i tak samo każdy rekord w tabeli B może mieć tylko jeden dopasowany rekord z tabeli A. Ten typ relacji spotyka się rzadko, ponieważ większość informacji powiązanych w ten sposób byłoby zawartych w jednej tabeli. Relacji **jeden-do-jednego** można używać do podziału tabeli z wieloma polami, do odizolowania części tabeli ze względów bezpieczeństwa, albo do przechowania informacji odnoszącej się tylko do podzbioru tabeli głównej.

2. Relacja jeden-do-wielu

Relacja jeden-do-wielu jest najbardziej powszechnym typem relacji.

W relacji **jeden-do-wielu** rekord w tabeli A może mieć wiele dopasowanych do niego rekordów z tabeli B, ale rekord w tabeli B ma tylko jeden dopasowany rekord w tabeli A.

3. Relacja wiele-do-wielu

W relacji **wiele-do-wielu**, rekord w tabeli A może mieć wiele dopasowanych do niego rekordów z tabeli B i tak samo rekord w tabeli B może mieć wiele dopasowanych do niego rekordów z tabeli A. Jest to możliwe tylko przez zdefiniowanie trzeciej tabeli (nazywanej tabelą łączą), której klucz podstawowy składa się z dwóch pól z kluczy obcych z tabel A i B. Relacja **wiele-do-wielu** jest w istocie dwiema relacjami jeden-do-wielu z trzecią tabelą. Na przykład, tabele "Zamówienia" i "Produkty" są powiązane relacją **wiele-do-wielu** zdefiniowaną przez utworzenie dwóch relacji jeden-do-wielu z tabelą "Opisy zamówień".

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

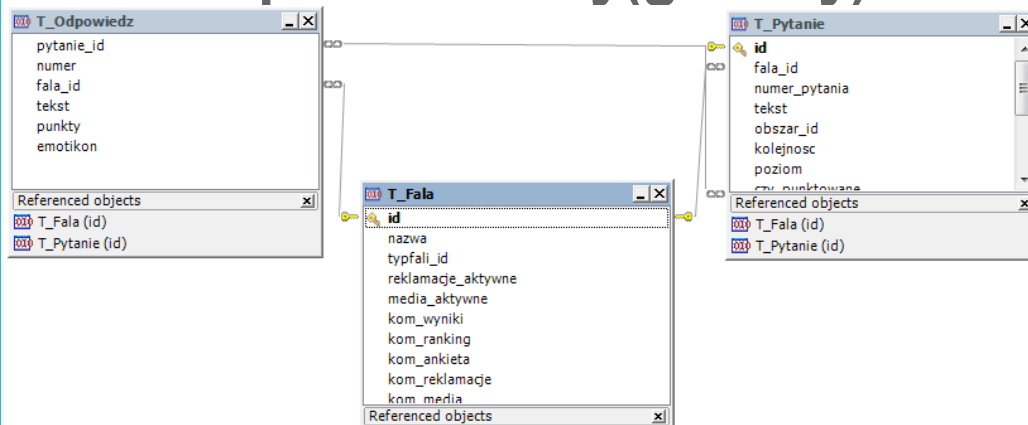
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





Podstawowe pojęcia

• Klucz podstawowy(główny)- PK

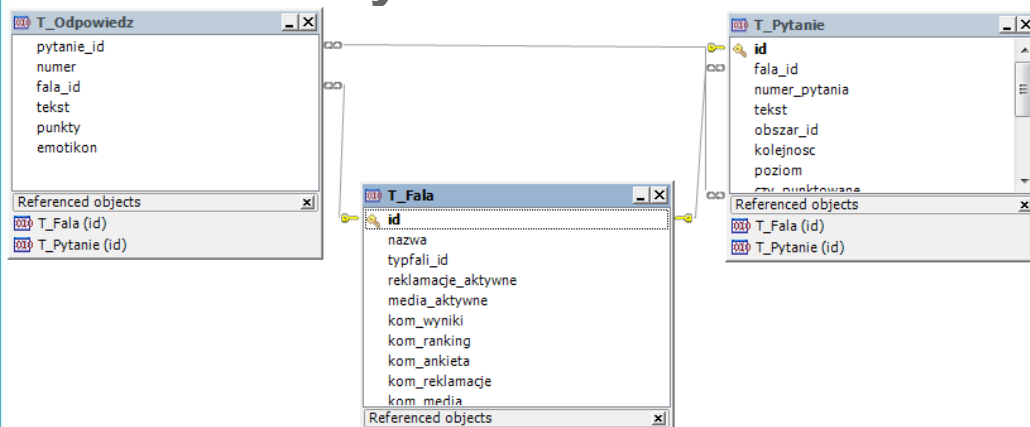


Każda tabela posiada tzw. klucz główny (primary key). Klucz ten jest unikatowym identyfikatorem w relacji i może być kombinacją kilku kolumn, często jednak obejmuje jedną kolumnę (jeden atrybut). Klucz ma za zadanie jednoznacznie identyfikować każdą krotkę (wiersz) – wartości w wyznaczonych kolumnach są jako zestaw niepowtarzalne w danej tabeli.



Podstawowe pojęcia

• Klucz obcy - FK



Innym rodzajem klucza jest tzw. klucz obcy (foreign key). Jest to zbiór atrybutów jednej tabeli (relacji) wskazujący wartości klucza kandydującego innej tabeli. Służy do wskazywania zależności pomiędzy danymi składowanymi w różnych tabelach. Klucze w modelu relacyjnym służą m.in. do sprawdzania spójności danych w bazie. Głównie dotyczy to kluczy obcych, na które nałożony jest wymóg, że w tabeli wskazywanej musi istnieć wartość klucza wskazującego.



Slajd
21



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Podstawowe pojęcia

• Sortowanie

Lp.	Nazwisko	Imię	Liczba książek
1	Pasek	Karolina	24
2	Nowak	Zbigniew	21
3	Różga	Adam	19
4	Opal	Ewa	14
5	Kowalski	Paweł	10
6	Nowak	Zuzanna	8
7	Nowak	Anna	5



Lp.	Nazwisko	Imię	Liczba książek
5	Kowalski	Paweł	10
7	Nowak	Anna	5
2	Nowak	Zbigniew	21
6	Nowak	Zuzanna	8
4	Opal	Ewa	14
1	Pasek	Karolina	24
3	Różga	Adam	19



Lekcja 1 - Bazy danych, podstawowe pojęcia

Dodatkowym elementem modelu relacyjnego jest zbiór operacji służących do przeszukiwania i manipulacji danymi. Przykładem operacji jest sortowanie danych.

Slajd
22



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Podstawowe pojęcia

• Zapytanie



Lekcja 1 - Bazy danych, podstawowe pojęcia

Najbardziej typowym zastosowaniem bazy danych jest zapytanie o wiersze spełniające konkretne kryteria, np. lista kluczowych klientów z tabeli klienci.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





Interakcje z bazą danych

• Język SQL

- Jedyny sposób interakcji z bazą danych
- Język deklaratywny
- Ustandaryzowany
 - Różni producenci stosują ten sam standard

```
select nazwisko, etat, placa
from pracownicy
where idzesp=30
and etat='kierownik'
```



Jakakolwiek interakcja aplikacji z bazą danych odbywa się za pomocą języka SQL. Jest to jedyny sposób komunikowania się aplikacji z bazą danych.

SQL jest językiem deklaratywnym (posługując się nim specyfikujemy tylko co chcemy otrzymać). Nie specyfikujemy sposobu (algorytmu) w jaki ma być zrealizowane zadanie. Przykładem polecenia SQL może być zapytanie do bazy danych poszukujące informacje o klientach banku ze Szczecina, którzy w ciągu ostatniego miesiąca wypłacili z bankomatu łącznie powyżej 8000 PLN. W tym zapytaniu specyfikujemy tylko jakie dane nas interesują. Sposób ich wyszukania jest automatycznie dobierany przez SZBD.

SQL jest językiem ustandaryzowanym. Jego standardyzacją zajmuje się specjalny międzynarodowy komitet, w skład którego wchodzi przedstawiciele największych producentów SZBD (IBM, Microsoft, Oracle). Dotychczas opracowano trzy standardy języka SQL, kolejno rozszerzające jego funkcjonalność. Standardy te to: SQL-92, SQL-99, SQL-2003.

Przykład prostego polecenia SQL będącego zapytaniem do bazy danych przedstawiono na slajdzie. Zapytanie to wyszukuje pracowników (nazwisko, etat, płaca) zatrudnionych w zespole o numerze 30 na etacie kierownika.



Slajd
 24


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

Interakcje z bazą danych

- **Aplikacje**
 - elektroniczne formularze z polami, listami,
 - elementami wyboru
 - umożliwiają wstawianie, modyfikowanie, usuwanie,
 - wyszukiwanie danych
- **raporty**
 - umożliwiają prezentowanie zawartości bazy danych (teksty, wykresy, grafika)


 Lekcja 1 - Bazy danych, podstawowe pojęcia

Język SQL jest narzędziem dostępu do bazy danych stosowanym głównie przez projektantów aplikacji, projektantów baz danych i administratorów baz danych. Standardowym sposobem korzystania z bazy danych przez użytkowników końcowych są aplikacje. Należy jednak pamiętać, że na poziomie programistycznym aplikacje również komunikują się z bazą danych za pomocą poleceń SQL.

Ze względu na funkcjonalność, wyróżnia się dwa rodzaje aplikacji, tj. formularze i raporty. Aplikację pierwszego rodzaju należy postrzegać jako elektroniczny formularz (z polami, listami, elementami wyboru) wypełniany przez użytkownika. Formularze umożliwiają pełną obsługę danych, tj. wstawianie, modyfikowanie, usuwanie i wyszukiwanie. Raporty umożliwiają wyłącznie odczytywanie danych z bazy i prezentowanie ich w różnej postaci, głównie tekstu lub wykresu.

 Slajd
 25


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

Ćwiczenie

- **Baza danych wypożyczalni filmów DVD/VHS**


 Lekcja 1 - Bazy danych, podstawowe pojęcia

Próba zaprojektowania bazy danych wypożyczalni filmów razem z uczniami, użycie pojęć zdefiniowanych w pierwszej części lekcji.

Człowiek - najlepsza inwestycja


KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

 UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY


Slajd
26

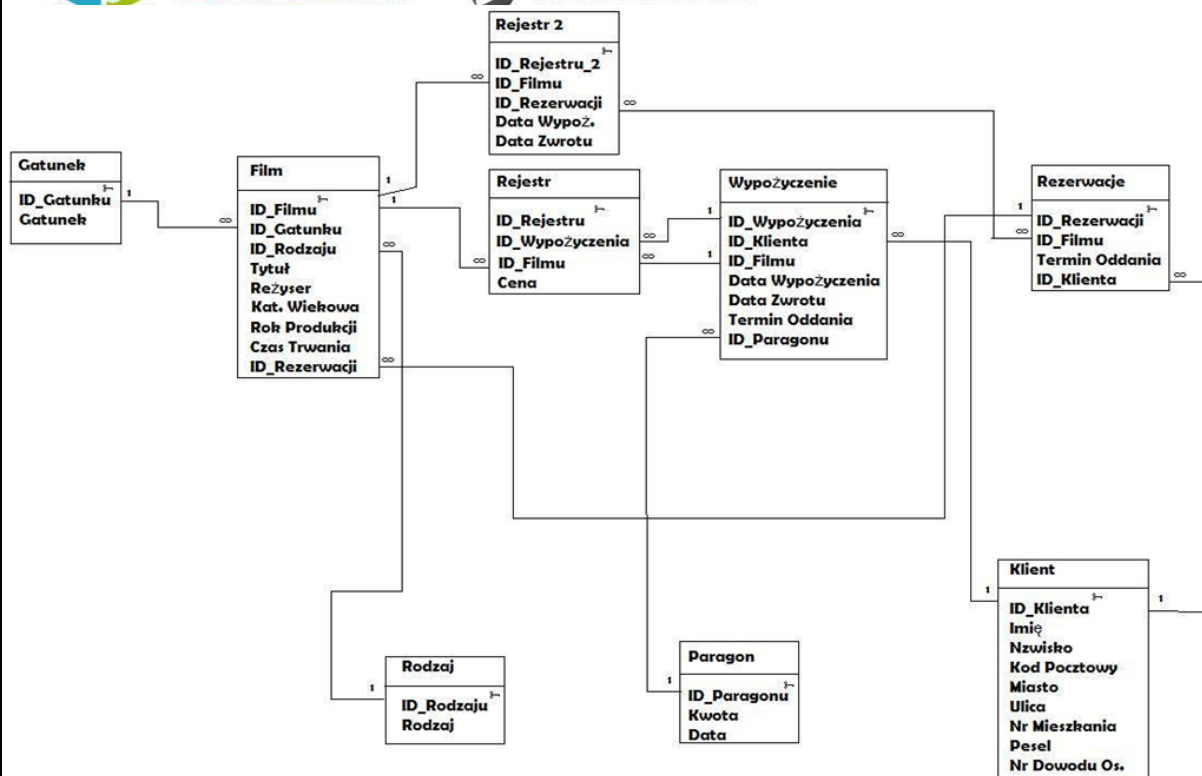


B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language



Przykład rozwiązania ćwiczenia

Slajd
27



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Ćwiczenie

- Wymień bazy danych bez których nie wyobrażasz sobie dzisiejszego świata



Lekcja 1 - Bazy danych, podstawowe pojęcia

Pytanie do uczniów o podanie najważniejszych ich zdaniem baz danych w dzisiejszej rzeczywistości. Przykłady odpowiedzi:

- Rachunki bankowe
- Mapy – nawigacje samochodowe
- Bazy urzędu skarbowego
- Bazy NFZ – np. eWUŚ
- Baza stron WWW – np. Google







Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 28</p>	  <p>SQL Structured Query Language</p> <p>Ćwiczenie</p> <ul style="list-style-type: none"> Wymień bazy danych w których znajdują się Twoje dane <p> Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Pytanie do uczniów o podanie baz danych, zawierających ich dane. Przykłady odpowiedzi:</p> <ul style="list-style-type: none"> Rachunki bankowe Mapy – nawigacje samochodowe Baza uczniów szkoły NFZ Baza Urzędu Miasta
<p>Slajd 29</p>	  <p>SQL Structured Query Language</p> <p>Ćwiczenie</p> <ul style="list-style-type: none"> Co jest najmniejszą jednostką w bazie danych? <ul style="list-style-type: none"> Tabela Kolumna Relacja Dana <p> Lekcja 1 - Bazy danych, podstawowe pojęcia</p>	<p>Odpowiedź prawidłowa: Dana</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Ćwiczenie

- Jaka Twoim zdaniem baza danych jest największa na świecie?
 - Biblioteka Kongresu
 - World Data Centre for Climate
 - YouTube
 - ChoicePoint
 - Google
 - CIA
 - China Mobile

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI



Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego



Lekcja 1 - Bazy danych, podstawowe pojęcia

Biblioteka Kongresu

To chyba akurat nikogo nie dziwi. Biblioteka Kongresu przechowuje 130 milionów dokumentów, wśród których można znaleźć autentycznie wszystko – od archiwalnych książek kucharskich aż po egzemplarze gazet z czasów kolonialnych. Gdyby wycisnąć z nich wszystkie literki, same słowa zajęłyby 20 terabajtów. BC rośnie o 10 000 egzemplarzy dziennie, a do przechowywania tego wszystkiego potrzeba półek o łącznej długości aż 530 mil. Pięć milionów dokumentów jest już dostępnych w wersji cyfrowej.

World Data Centre for Climate

Którym zarządzają Meteorologiczny Instytut Maxa Plancka oraz Niemieckie Centrum Komputerowej Analizy Klimatu. WDCC chwali się tym, że posiada bazę danych o objętości ponad 220 terabajtów, do których można dostać się przez internet oraz dodatkowo 110 terabajtów symulacji klimatycznych, a także sześć petabajtów przechowywanych na taśmach magnetycznych.

YouTube

Przy powstawaniu serwisu, przewidywano że nigdy nie będzie on wymagał więcej niż 45 terabajtów. Ze względu na to, że filmiki wrzucane przez użytkowników różnią się od siebie długością i rozmiarem, ciężko oszacować rozmiar bazy danych YT dzisiaj, ale "magiczne" 45 terabajtów zostało z pewnością już dawno przekroczone. Co godzinę na YouTube pojawia się 60 godzin wideo, oglądane są na nim cztery miliardy filmów dziennie, co daje łącznie ponad trzy miliardy godzin oglądania na miesiąc. Robi wrażenie.

ChoicePoint

Jedna z dziwniejszych firm, jakie kiedykolwiek istniały. ChoicePoint, zlokalizowana niedaleko Atlanty, zajmowała się zbieraniem danych osobowych, które potem sprzedawała tym, którzy oferowali najwyższą cenę. Udało jej się wejść w posiadanie 17 miliardów dokumentów na temat 250 milionów

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



osób – gdyby je wydrukować sięgnęłyby z Ziemi do Księżyca i z powrotem 77 razy. Dane zbierane przez ChoicePoint zawierały wszystko na temat Amerykanów – rejestry karanych, adresy, numery telefonów, wydruki z kart kredytowych i tak dalej, i tak dalej. Niestety okazało się przy okazji, że wyciekały z firmy, a do tego wielokrotnie zdarzało się, że podawane przez CP informacje okazywały się nie do końca poprawne, co potrafiło doprowadzić do tego, że niektóre osoby traciły pracy, kiedy okazywało się na przykład, że wbrew temu, co mówiły, były jednak karane.

Google

To też chyba nikogo tutaj nie dziwi. Jednak dane o dokładnych rozmiarach baz danych Google'a nie są znane, bo firma ich nie ujawnia. To co jednak wiadomo, to to, że centra danych korporacji z Mountain View pożarły w 2010 roku kilowatogodzin czyli połowę tego, co produkuje w ciągu roku słynna Zapora Hoovera – tama wysoka na ponad 220 metrów i długa na blisko 400. I że każdego dnia Google analizuje 24 petabajty danych. Żeby je wydrukować na papierze, trzeba by ściąć ponad 1,2 miliona drzew.

CIA

Tutaj małe zaskoczenie, aczkolwiek w sumie to żadne zaskoczenie – bazy danych CIA są uznawane za jedne z największych na świecie, choć nikt nie ma zielonego pojęcia, ile informacji się w nich znajduje. A wszystko to dlatego, że większość z nich jest utajnionych.

China Mobile

Tutaj też dokładnie nie wiadomo, niełatwo wyciąga się takie informacje od Chińczyków. Ale swego czasu jedną z największych baz danych miał znany operator telefonii komórkowej Sprint, który z 53 milionami użytkowników posiadał bazę danych zawierającą blisko trzy miliardy rekordów. A China Mobile oferuje swoje usługi 70% Chińczyków, mając tym samym 655 milionów klientów. Poczujcie cenną różnicę.

9.1.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdzie 25: Próba wspólnego zaprojektowania bazy danych wypożyczalni filmów DVD, wykorzystanie pojęć przedstawionych w czasie lekcji oraz dodatkowo na slajdach 26-30.

9.1.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie poznają definicję bazy danych, podstawowe pojęcia z nią związane. Będą potrafili wskazać przykłady baz danych.

9.2 Lekcja 2 - Języki programowania serwerów SQL, środowisko programowania w języku SQL, wprowadzenie do języka SQL


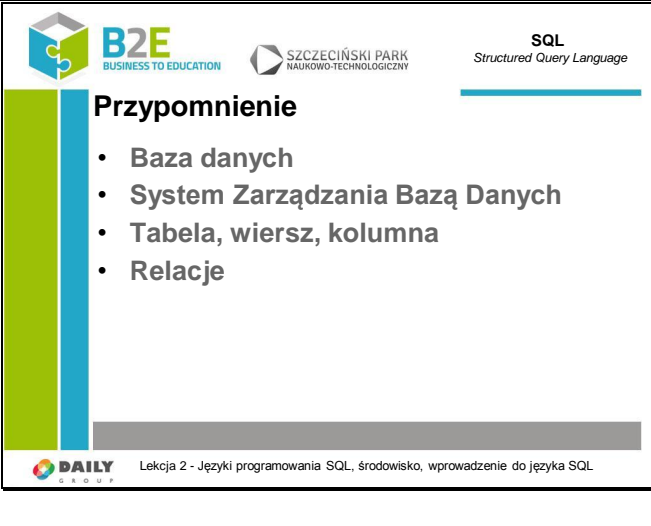
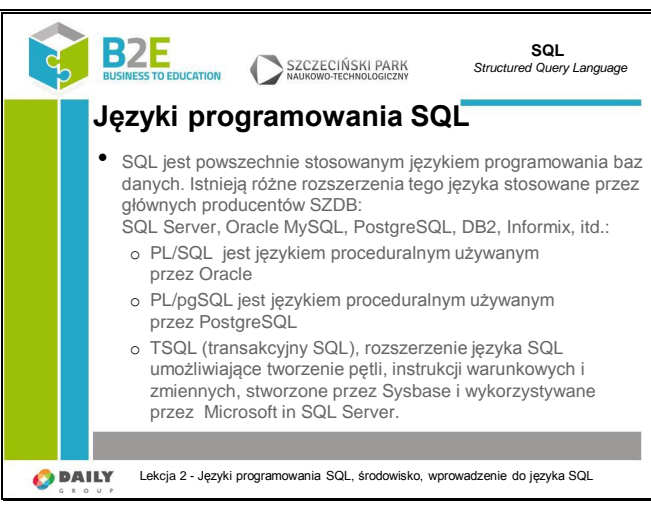
9.2.1 Cel lekcji

Celem lekcji jest poznanie różnych rozszerzeń języka SQL, stosowanych przez producentów systemów zarządzania bazą danych. Uczniowie zostaną się ze środowiskiem Microsoft SQL Management Studio Express. Zostaną poinstruowani w jaki sposób można pobrać i zainstalować środowisko.

W drugiej części lekcji uczniowie poznają podstawowe kwerendy (zapytania) SQL, a w ramach ćwiczeń – spróbują praktycznie wykorzystać wiedzę.

Człowiek - najlepsza inwestycja

9.2.2 Treść - slajdy z opisem

<p>Slajd 1</p>		<p>SQL jest najpopularniejszym językiem programowania baz danych. W tej lekcji poznamy różne rodzaje języka SQL, oraz poznamy podstawy tego języka.</p>
<p>Slajd 2</p>		
<p>Slajd 3</p>		<p>Wyjaśnienie, że pomimo wspólnego standardu istnieje wiele różnych rozszerzeń. Podstawowa składnia wszystkich rozszerzeń języka jest taka sama, jednak istnieje wiele różnic. W tym kursie zajmiemy się językiem SQL w ujęciu rozszerzenia TSQL, promowanym przez Microsoft.</p>

Człowiek - najlepsza inwestycja

Slajd
4

Środowisko programowania

- Microsoft SQL Server Management Studio
- Wersja bezpłatna – Express
- Toad for SQL Server Freeware

Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

Środowiska programowania TSQL. Microsoft oferuje wraz z bazą danych SQL Server narzędzie Microsoft SQL Server Management Studio (często używa się skrótu MSSMS). Możliwe jest pobranie wersji bezpłatnej oznaczonej przez producenta hasłem „Express”. Poza oficjalnym narzędziem dostępne są również narzędzia innych firm, np. Toad (również darmowe).

Slajd
5

MS SQL Management Studio Express

- Do pobrania ze strony Microsoft
 - <http://www.microsoft.com/en-us/download/details.aspx?id=29062>

Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

Management Studio wraz z bazą danych można pobrać ze strony Microsoftu. Adres podany jest na slajdzie. Warto zwrócić uwagę aby pobrać wersję SZDB wraz z MS SQL SMS: ENU\x64\SQLSEXPRT_x64_ENU.exe:

„Express with Tools (with LocalDB) Includes the database engine and SQL Server Management Studio Express)

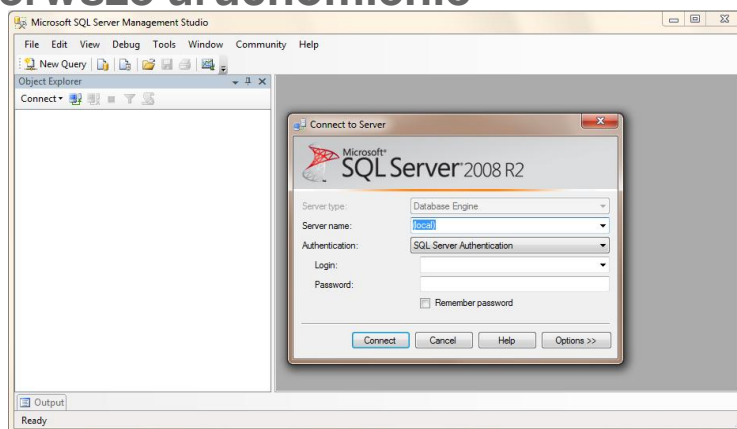
This package contains everything needed to install and configure SQL Server as a database server. Choose either LocalDB or Express depending on your needs above.”

Człowiek - najlepsza inwestycja



MS SQL Management Studio Express


- Pierwsze uruchomienie




Po instalacji oprogramowania należy się zalogować do bazy danych: wybieramy serwer lokalny, podajemy użytkownika i hasło, które zostało zdefiniowane w procesie instalacji. Domyślnym użytkownikiem administracyjnym jest sa (super administrator).



Slajd
7



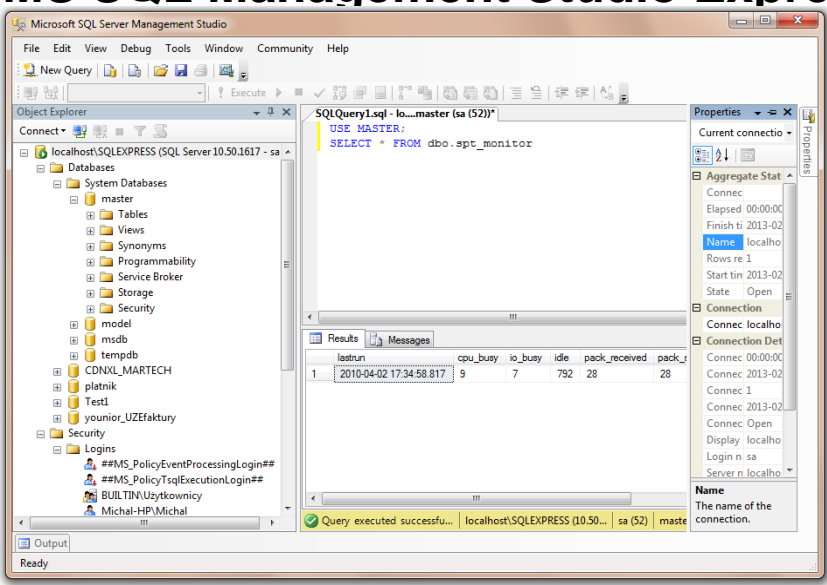
B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language


MS SQL Management Studio Express




Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

Opis elementów programu MS SQL SMS. Okno programu składa się z sekcji obiektów bazy danych po lewej stronie, sekcji roboczej (wykorzystywanej na przykład do wpisywania zapytań TSQL) na środku, poniżej – okno rezultatów zapytań TSQL oraz okna właściwości po prawej stronie.

Slajd
8



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Wprowadzenie do języka SQL

- **SQL to język dostępu do bazy danych**
- **Grupy poleceń języka:**
 - DDL - język definiowania danych (Data Definition Language)
 - DML - Język manipulowania danych (Data Manipulation Language)
 - DCL - Język sterowania danych (Data Control Language)
 - Język zapytań (Query Language)

Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

- Język definiowania danych (Data Definition Language - DDL), który umożliwia definiowanie struktury danych zawartych w bazie.
- Język manipulowania danych (Data Manipulation Language - DML), który umożliwia wypełnienie, modyfikowanie i usuwanie danych z bazy.
- Język sterowania danych (Data Control Language), który umożliwia sterowanie transakcjami tj. akceptacja lub wycofanie.
- Język zapytań (Query Language), który umożliwia pobieranie informacji z bazy za pośrednictwem określonych zapytań, warunków.

Człowiek - najlepsza inwestycja

<p>Slajd 9</p>	  <p>SQL Structured Query Language</p> <h2>Wprowadzenie do języka SQL</h2> <ul style="list-style-type: none"> Polecenie SQL może być zapisane: <ul style="list-style-type: none"> w jednym bądź wielu wierszach dużymi lub małymi literami Polecenia powinny być zakończone średnikiem <pre>SELECT * FROM pracownicy;</pre> <pre>select * from PRACOWNICY;</pre>  <p>Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>W SQL występuje dowolność wielkich i małych liter. W ramach ujednolicania rozszerzeń języków SQL, Microsoft dopuszcza stosowanie średnika na końcu polecenia (w PL/SQL jest wymagany).</p>
<p>Slajd 10</p>	  <p>SQL Structured Query Language</p> <h2>Wyświetlanie elementów (projekcja)</h2> <ul style="list-style-type: none"> Wybór wartości z tabeli, np. <pre>SELECT imie, nazwisko, pesel FROM pracownicy;</pre> <ul style="list-style-type: none"> Operatory arytmetyczne, np. +, -, *, / <pre>SELECT nazwisko, placa_zasadnicza*3, premia + 300 FROM pracownicy;</pre>  <p>Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>Przykład zapytania SELECT: pobranie imion i nazwisk wszystkich pracowników. W poleceniu tłumaczonym z j. angielskiego Wybierz imię, nazwisko, pesel z tabeli pracownicy, występują nazwy kolumn (atrybuty) oraz nazwa tabeli (pracownicy). W wyrażeniach SELECT można korzystać z operacji arytmetycznych takich jak dodawanie, odejmowanie, mnożenie i dzielenie.</p>
<p>Slajd 11</p>	  <p>SQL Structured Query Language</p> <h2>Aliasy</h2> <ul style="list-style-type: none"> Alternatywna nazwa atrybutu Można stosować słowo AS <pre>SELECT nazwisko, placa_zasadnicza*1.2 AS nowa_placa, premia + 300 AS premia_swiateczna FROM pracownicy;</pre>   <p>Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>W celu wyświetlenia kolumn o nazwach innych niż kolumny tabeli można stosować aliasy. Alias wpisujemy po nazwie kolumny lub po słowie kluczowym AS.</p>


Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

NULL


- Wartość pusta w tabeli
- Wynikiem jakiegokolwiek operacji z NULL jest NULL

```
SELECT * FROM pracownicy;
```

	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko
1	Kowalski	Jan	2300.00	250.00	82091104357	Manager
2	Nowak	Karol	2700.00	100.00	80010123987	Specjalista
3	Przepiórka	Marzena	2700.00	NULL	89121203456	Specjalista

```
SELECT nazwisko,
       placa_zasadnicza*1.2 AS nowa_placa,
       premia + 300 premia_swiateczna
FROM pracownicy;
```

	nazwisko	nowa_placa	premia_swiateczna
1	Kowalski	2760.000	550.00
2	Nowak	3240.000	400.00
3	Przepiórka	3240.000	NULL



Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

W bazach danych występuje wartość pusta NULL. Nie jest to 0, czy pusty ciąg znaków – tylko wartość nieokreślona. Wynikiem dodawania, odejmowania, mnożenia czy dzielenia wartości NULL jest również wartość NULL, co zaprezentowano na przykładzie.

Ciekawostka: w Oracle w przeciwieństwie do MSSQL NULL traktowany jest jak pusty string.



ISNULL

- ISNULL(*kolumna, wartość_w_przypadku_NULL*)
- Możliwość zmiany wartości pustej na inną, np.

```
SELECT nazwisko,
       placa_zasadnicza*1.2 AS nowa_placa,
       ISNULL(premia,0) + 300 premia_swiateczna
FROM pracownicy;
```

	nazwisko	nowa_placa	premia_swiateczna
1	Kowalski	2760.000	550.00
2	Nowak	3240.000	400.00
3	Przepiórka	3240.000	300.00



Możliwością uniknięcia problemów z kolumnami zawierającymi wartość NULL jest polecenie ISNULL, które ma dwa parametry. Pierwszym jest nazwa kolumny, drugim wartość jaka ma być podstawiona w przypadku gdy kolumna zawiera wartość NULL.



Porządkowanie elementów




- Klauzula ORDER BY
- Słowa kluczowe DESC i ASC
- ORDER BY występuje zawsze na końcu zapytania

```
SELECT imie, nazwisko, pesel FROM pracownicy
ORDER BY nazwisko DESC, imie ASC;
```












Sortowanie wyników uzyskuje się przy wykorzystaniu klauzuli ORDER BY. Sortowanie rosnące (ascending) jest domyślne. Po nazwie kolumny można użyć słów kluczowych ASC i DESC w celu uzyskania oczekiwanej kolejności.



Slajd 15	  <div> <div>SQL Structured Query Language</div> <div>Eliminowanie duplikatów</div> <ul style="list-style-type: none"> Polecenie DISTINCT <pre>SELECT DISTINCT stanowisko FROM pracownicy;</pre>  <div>  Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL </div> </div>	<p>Słowo kluczowe DISTINCT musi występować zaraz po słowie kluczowym SELECT.</p> <p>SELECT stanowisko FROM pracownicy</p> <p>Takie zapytanie wyświetli wszystkie stanowiska obejmowane wśród pracowników. Jeżeli pojawią się dwa takie same stanowiska, tylko jedno zostanie wyświetlone. Słowo DISTINCT eliminuje wiersze, które posiadają duplikaty we wszystkich kolumnach wyspecyfikowanych w wyrażeniu SELECT. Tylko jedno słowo DISTINCT może zostać użyte w całym zapytaniu SELECT.</p>
Slajd 16	  <div> <div>SQL Structured Query Language</div> <div>Selekcja wybranych danych</div> <ul style="list-style-type: none"> Klauzula WHERE Ogólna składnia: SELECT kolumna1, kolumna2, kolumna3 FROM tabela WHERE kolumna operator wartość Operatory: =, !=, >, >=, <, <= <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE placa_zasadnicza > 2000</pre> <div>  Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL </div> </div>	<p>Używając klauzuli WHERE możemy zawęzić zwracane wartości do tylko tych które spełniają wybrane kryteria. Np. kiedy chcemy znaleźć pracowników zarabiających więcej niż 2000zł.</p>
Slajd 17	  <div> <div>SQL Structured Query Language</div> <div>Selekcja wybranych danych</div> <ul style="list-style-type: none"> Operatory BETWEEN ... AND ... <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE placa_zasadnicza BETWEEN 2000 AND 3000</pre> <ul style="list-style-type: none"> IN <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE stanowisko IN ('Specjalista', 'Manager')</pre> <div>  Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL </div> </div>	<p>Operator BETWEEN AND umożliwia wybór wartości z danego zakresu, operator IN wybór wartości zawierających się w zdefiniowanym zbiorze</p>

Człowiek - najlepsza inwestycja

<p>Slajd 18</p>	  <p>SQL Structured Query Language</p> <h3>Selekcja wybranych danych</h3> <ul style="list-style-type: none"> • LIKE <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE nazwisko LIKE 'M%'; SELECT imie, nazwisko, pesel FROM pracownicy WHERE nazwisko LIKE '%ski';</pre> • IS NULL <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE premia IS NULL;</pre> <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>Operator LIKE umożliwia wybór wartości które zawierają pewien ciąg znaków, np. zaczynają się od litery „M”, lub kończą na „ski”.</p>
<p>Slajd 19</p>	  <p>SQL Structured Query Language</p> <h3>Selekcja wybranych danych</h3> <ul style="list-style-type: none"> • Negacja operatorów SQL <ul style="list-style-type: none"> • NOT BETWEEN ... AND ... • NOT IN • NOT LIKE • IS NOT NULL <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE stanowisko NOT IN ('Specjalista', 'Manager');</pre> <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>Negowanie warunków jest możliwe za pomocą słowa kluczowego NOT. NOT występuje przed operatorem. Wyjątkiem jest operator IS NULL, gdzie negator występuje w środku: IS NOT NULL.</p>
<p>Slajd 20</p>	  <p>SQL Structured Query Language</p> <h3>Selekcja wybranych danych</h3> <ul style="list-style-type: none"> • Łączenie warunków <ul style="list-style-type: none"> • AND • OR <pre>SELECT imie, nazwisko, pesel FROM pracownicy WHERE stanowisko NOT IN ('Specjalista', 'Manager') AND placa_zasadnicza > 2000;</pre> <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>Operatory logiczne mogą być stosowane jednocześnie w tej samej klauzuli WHERE. Służą do tego operatory logiczne AND i OR. AND posiada wyższy priorytet niż OR, a zmiana priorytetu jest możliwa za pomocą nawiasów.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
21



B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Ćwiczenia

	nazwisko	imie	płaca_zasadnicza	premia	pesel	stanowisko
1	Kowalski	Jan	2300.00	250.00	82091104357	Manager
2	Nowak	Karol	2700.00	100.00	80010123987	Specjalista
3	Przepiórka	Marzena	2700.00	NULL	89121203456	Specjalista
4	Burzych	Paweł	1900.00	500.00	78032309123	Specjalista
5	Makłowicz	Marek	2000.00	NULL	54013112345	Specjalista
6	Naramowicka	Magdalena	2100.00	200.00	77121312098	Specjalista
7	Witos	Jacek	3000.00	500.00	69100967234	Manager



Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

- Ćwiczenie 1:**
Napisz zapytanie zwracające wszystkich Managerów
- Ćwiczenie 2:**
Napisz zapytanie zwracające wszystkich specjalistów, których łączna płaca jest wyższa niż 2300zł
- Ćwiczenie 3:**
Napisz zapytanie wyświetlające wszystkie kobiety wśród pracowników (końcówka imienia „a”)
- Ćwiczenie 4:**
Napisz zapytanie wyświetlające pracowników, których roczna płaca zasadnicza jest większa niż 25.000zł
- Ćwiczenie 5:**
Napisz zapytanie wyświetlające pracowników, którzy nie otrzymują premii
- Ćwiczenie 6:**
Napisz zapytanie wyświetlające pracowników, których premia jest pomiędzy 200 a 300zł
- Ćwiczenie 7:**
Napisz zapytanie wyświetlające łączny roczny dochód pracowników
- Ćwiczenie 8:**
Napisz zapytanie wyświetlające pracowników urodzonych w latach 80-tych
- Ćwiczenie 9:**
Napisz zapytanie wyświetlające specjalistów urodzonych w latach 80-tych, zarabiających więcej niż 2.700zł
- Ćwiczenie 10:**
Napisz zapytanie wyświetlające pracowników, których imię i nazwisko zaczyna się na literę M

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY



<p>Slajd 22</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz zapytanie zwracające wszystkie wiersze tabeli TABELA, których pole PREMIA jest niepuste <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>SELECT * FROM TABELA WHERE PREMIA IS NOT NULL</p>
<p>Slajd 23</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz zapytanie zwracające wszystkie wiersze tabeli TABELA, których pole PREMIA jest większe od 1000 i mniejsze od 2000 <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>SELECT * FROM TABELA WHERE PREMIA IS BETWEEN 1000 AND 2000</p>
<p>Slajd 24</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz zapytanie zwracające wszystkie wiersze tabeli TABELA, posortowane po polu PREMIA od największej do najmniejszej <p> Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL</p>	<p>SELECT * FROM TABELA ORDER BY PREMIA DESC</p>

Człowiek - najlepsza inwestycja





KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
 25



B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

Ćwiczenia

- Napisz zapytanie zwracające przygotowane premie świąteczne z tabeli TABELA, gdzie PREMIA będzie dwukrotnie wyższa i dodatkowo będzie zawierała dodatek w wysokości 30zł dla każdego pracownika. Na raporcie pokaż pole premia w postaci „Premia świąteczna”.


 Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

 SELECT (PREMIA*2)+30 AS [Premia
 świąteczna] FROM TABELA

 Slajd
 26


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

Podsumowanie

SELECT	Wybiera listę kolumn
alias	Można stosować tylko do kolumn i wyrażeń (nie do *)
*	Oznacza wszystkie kolumny
DISTINCT	Eliminuje duplikaty ze zbioru wyników
FROM t	Określa relację, z której odczytujemy dane
WHERE	Określa warunki wyboru wierszy, zawiera wartości kolumn
AND/OR	Łączy warunki w klauzuli WHERE
()	Pozwala na zmianę priorytetu operatorów
ORDER BY	Służy do określenia kryterium sortowania
ASC	Rosnący porządek sortowania
DESC	Malejący porządek sortowania

Człowiek - najlepsza inwestycja


KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI


UNIA EUROPEJSKA
 EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego


 Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

Tabela podsumowująca poznane wyrażenia

9.2.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdzie 21. Uczniowie w ramach ćwiczenia będą musieli napisać zapytania SQL z tabel przygotowanych przez szkoleniowca. Dodatkowe ćwiczenia przedstawiono na slajdach 22-25.

9.2.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie będą wiedzieli jak pobrać i zainstalować środowisko MS SMS. Dowiedzą się jak konstruować proste zapytania SQL, polegające na wyświetlaniu informacji z bazy, sortowaniu, definiowaniu podstawowych warunków zapytania.



Człowiek - najlepsza inwestycja

9.3 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych







9.3.1 Cel lekcji

Celem lekcji jest zaznajomienie uczniów z bardziej skomplikowanymi zapytaniami SQL niż w poprzedniej lekcji. Uczniowie dowiedzą się, jakie są sposoby łączenia danych z wielu tabel. W drugiej części lekcji przedstawione zostaną zagadnienia związane z funkcjami agregującymi oraz grupowaniem danych. Dzięki ćwiczeniom praktycznym, uczniowie praktycznie wykorzystają zdobytą w czasie lekcji wiedzę.

9.3.2 Treść - slajdy z opisem

Slajd 1		Na poprzedniej lekcji zajmowaliśmy się prostymi zapytaniami do bazy danych, teraz przejdziemy do bardziej złożonych zapytań, nauczymy się łączyć tabele, grupować dane. Dowiemy się jak wybierać wartości minimalne i maksymalne, nauczymy się sumować.
Slajd 2		Powtórzenie materiału z poprzedniej lekcji.

Człowiek - najlepsza inwestycja

Slajd 3	<div>   <div> SQL Structured Query Language </div> </div> <h2>Ograniczanie wyników</h2> <pre>SELECT TOP 3 imie, nazwisko, pesel FROM pracownicy</pre> <ul style="list-style-type: none"> Polecenie TOP: <ul style="list-style-type: none"> Używana zaraz po SELECT Wartość liczbowa, np. TOP 3 Wartość procentowa, np. TOP 10 PERCENT <div>  Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych </div>	<p>Ograniczanie wyników w SQL realizowane jest słowem TOP. Możliwe jest ograniczenie ilościowe i procentowe. Używanie TOP bez klauzuli ORDER BY jest mocno niezalecane z uwagi na możliwą przypadkowość wyniku!</p> <p>Ciekawostka: Od MSSQL 2012 są też bardziej praktyczne metody tj. OFFSET (ilość rekordów do pominięcia) i FETCH ilość rekordów do pobrania.</p>
Slajd 4	<div>   <div> SQL Structured Query Language </div> </div> <h2>Komentarze w SQL</h2> <ul style="list-style-type: none"> „--” – pojedynczy wiersz „/*” i „*/” – cała sekcja <pre> SELECT TOP 3 imie, --nazwisko, pesel FROM pracownicy SELECT TOP 3 /*imie, nazwisko, */ pesel FROM pracownicy </pre> <div>  Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych </div>	<p>Język SQL umożliwia stosowanie komentarzy. W celu komentowania jednego wiersza stosuje się dwa minusy. Można również zakomentować całą sekcję – zaczynamy komentarz od ukośnika i gwiazdki, kończymy komentarz gwiazdką i ukośnikiem.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY





Łączenie tabel

dbo.pracownicy						
	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko
▶	Kowalski	Jan	2300,00	250,00	82091104357	Manager
	Nowak	Karol	2700,00	100,00	80010123987	Specjalista
	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista
	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista
	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista
	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista
	Witos	Jacek	3000,00	500,00	69100967234	Manager
*	NULL	NULL	NULL	NULL	NULL	NULL

dbo.miejsca						
	nr_miejsca	ulica	numer	miasto	kod	telefon
▶	1	Mała	13	Poznań	60-002	(61) 123 09 89
	2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
	3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87



Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Na slajdzie widzimy dwie tabele – znaną z poprzednich lekcji „pracownicy” oraz „miejsca” możemy sobie wyobrazić potrzebę połączenia tych tabel, np. w celu przygotowania zestawienia – pracownicy wraz z miejscem zamieszkania.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
6

Łączenie tabel

dbo.pracownicy						
	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko
1	Jan	Jan	2300,00	250,00	82091104357	Manager
2	Karol	Karol	2700,00	100,00	80010123987	Specjalista
3	Marzena	Przepiórka	2700,00	ALL	89121203456	Specjalista
4	Paweł	Burzych	1900,00	500,00	78032309123	Specjalista
5	Marek	Makłowicz	2000,00	ALL	54013112345	Specjalista
6	Magdalena	Naramowicka	2100,00	200,00	77121312098	Specjalista
7	Jacek	Witos	3000,00	500,00	69100967234	Manager
8	ALL	ALL	ALL	ALL	ALL	ALL

dbo.miejsca					
nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87

	imie	nazwisko	pesel	ulica	numer	miasto
1	Jan	Kowalski	82091104357	Mała	13	Poznań
2	Karol	Nowak	80010123987	Mała	13	Poznań
3	Marzena	Przepiórka	89121203456	Grunwaldzka	34	Warszawa
4	Paweł	Burzych	78032309123	Wąska	4/12A	Kraków
5	Marek	Makłowicz	54013112345	Wąska	4/12A	Kraków
6	Magdalena	Naramowicka	77121312098	Mała	13	Poznań
7	Jacek	Witos	69100967234	Grunwaldzka	34	Warszawa

Często w praktyce musimy połączyć wyniki z kilku tabel. Wyobraźmy sobie zadanie – przygotować raport pracowników wraz z ich miejscem pracy.



Złączenia JOIN oraz CROSS JOIN

```
SELECT imie, nazwisko, pesel, ulica, numer, miasto
FROM pracownicy, miejsca
SELECT * FROM pracownicy CROSS JOIN miejsca
```

- Zbiór wszystkich możliwych kombinacji rekordów z obu tabel

		nazwisko	pesel	ulica	numer	miasto
1	Jan	Kowalski	82091104357	Mała	13	Poznań
2	Karol	Nowak	80010123987	Mała	13	Poznań
3	Marzena	Przepli...	89121203456	Mała	13	Poznań
4	Paweł	Burzych	78032309123	Mała	13	Poznań
5	Marek	Maklo...	54013112345	Mała	13	Poznań
6	Magdalena	Naram...	77121312098	Mała	13	Poznań
7	Jacek	Witos	69100967234	Mała	13	Poznań
8	Jan	Kowalski	82091104357	Wąska	4/1...	Kraków
9	Karol	Nowak	80010123987	Wąska	4/1...	Kraków
10	Marzena	Przepli...	89121203456	Wąska	4/1...	Kraków
11	Paweł	Burzych	78032309123	Wąska	4/1...	Kraków
12	Marek	Maklo...	54013112345	Wąska	4/1...	Kraków
13	Magdalena	Naram...	77121312098	Wąska	4/1...	Kraków
14	Jacek	Witos	69100967234	Wąska	4/1...	Kraków
15	Jan	Kowalski	82091104357	Grunwaldzka	34	Warszawa
16	Karol	Nowak	80010123987	Grunwaldzka	34	Warszawa
17	Marzena	Przepli...	89121203456	Grunwaldzka	34	Warszawa
18	Paweł	Burzych	78032309123	Grunwaldzka	34	Warszawa
19	Marek	Maklo...	54013112345	Grunwaldzka	34	Warszawa



JOIN (lub CROSS JOIN) może być stosowany w różnych wersjach

Wynikiem złączenia naturalnego jest zbiór wierszy łączonych tabel; dla tych wierszy wartości kolumn określonych jako warunek złączenia są takie same. Ponieważ w relacyjnych bazach danych informacje są podzielone pomiędzy tabele zawierające dane o obiektach jednego typu, złączenie naturalne jest najczęściej wykorzystywanym (i domyślnym) złączeniem obiektów.

W przykładowej bazie danych informacje o pracownikach i ich miejscach pracy przechowywane są w powiązanych ze sobą tabelach. Dlatego aby wyświetlić pracowników i ich miejsca pracy musimy skorzystać z obu tabel. Przy czym z reguły nie chodzi nam o uzyskanie poniższego wyniku - który wyświetla wszystkie możliwe kombinacje danych.



Złączenie INNER JOIN

```
SELECT imie, nazwisko, pesel, ulica, numer, miasto
FROM pracownicy INNER JOIN miejsca
ON pracownicy.miejsce = miejsca.nr_miejsca
```

• INNER JOIN ... ON ...

	imie	nazwisko	pesel	ulica	numer	miasto
1	Jan	Kowalski	82091104357	Mała	13	Poznań
2	Karol	Nowak	80010123987	Mała	13	Poznań
3	Marzena	Przepiórka	89121203456	Grunwaldzka	34	Warszawa
4	Paweł	Burzych	78032309123	Wąska	4/12A	Kraków
5	Marek	Makłowicz	54013112345	Wąska	4/12A	Kraków
6	Magdalena	Naramowicka	77121312098	Mała	13	Poznań
7	Jacek	Witos	69100967234	Grunwaldzka	34	Warszawa

Klauzula ON jest sposobem na podanie warunku złączenia, czyli wskazania wspólnych kolumn łączonych tabel.

Poprawnie napisana instrukcja powinna zwrócić nam tylko miejsca do których przypisany jest konkretny pracownik. Żeby to osiągnąć, musimy określić warunek złączenia, czyli poinformować serwer baz danych, co łączy zapisane w obu tabelach dane. W tym przypadku jest to identyfikator miejsca pracy — zwróć uwagę, że kolumna *miejsce* występuje w obu tabelach, czyli na podstawie tego identyfikatora jesteśmy w stanie sensownie połączyć informacje o pracownikach i miejscach.

INNER JOIN ... ON .. łączy tabele zgodnie ze zdefiniowanym warunkiem.

JOIN (z ang. łączyć) — występuje zawsze w części po FROM w zapytaniu SQL. Tłumacząc powyższą kwerendę (zapytanie):

Wyświetl imię, nazwisko, pesel, ulica, numer, miasto z tabeli pracownicy i połącz z tabelą miejsca, gdzie miejsce w tabeli pracownicy jest takie samo jak nr_miejsca w tabeli miejsca.



Złączenia zewnętrzne

- **INNER JOIN** pomija wyniki, które nie mają odpowiedników obu tabelach

```
SELECT imie, nazwisko, pesel, ulica, numer, miasto |
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejsca = miejsca.nr_miejsca
```


	imie	nazwisko	pesel	ulica	numer	miasto
1	Jan	Kowalski	82091104357	Mała	13	Poznań
2	Karol	Nowak	80010123987	Mała	13	Poznań
3	Paweł	Burzych	78032309123	Wąska	4/12A	Kraków
4	Marek	Makłowicz	54013112345	Wąska	4/12A	Kraków
5	Magdalena	Naramowicka	77121312098	Mała	13	Poznań
6	Jacek	Witos	69100967234	Grunwaldzka	34	Warszawa




Złączenie naturalne eliminuje z wyniku niepasujące (niepełniające warunku złączenia) wiersze. To dobrze, bo w innym przypadku otrzymalibyśmy zawierający mnóstwo powtórzeń i niepotrzebnych danych iloczyn kartezjański. Ale z drugiej strony ten sam warunek złączenia usunął z wyniku rekordy niemające odpowiedników w łączonej tabeli. Czyli wynik instrukcji wcale nie musi zawierać danych wszystkich naszych pracowników.



Slajd
10



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Złączenia zewnętrzne


- OUTER JOIN nie pomija wyników**

OUTER JOIN:

- LEFT OUTER JOIN – nie pomija wyników z lewej tabeli
- RIGHT OUTER JOIN – nie pomija wyników z prawej tabeli
- OUTER JOIN – nie pomija wyników

```
SELECT imie, nazwisko, pesel, ulica, numer, miasto
FROM pracownicy LEFT OUTER JOIN miejsca
ON pracownicy.nr_miejsca = miejsca.nr_miejsca
```

	imie	nazwisko	pesel	ulica	numer	miasto
1	Jan	Kowalski	82091104357	Mała	13	Poznań
2	Karol	Nowak	80010123987	Mała	13	Poznań
3	Marzena	Przepiórka	89121203456	NULL	NULL	NULL
4	Paweł	Burzych	78032309123	Wąska	4/12A	Kraków
5	Marek	Makłowicz	54013112345	Wąska	4/12A	Kraków
6	Magdalena	Naramowicka	77121312098	Mała	13	Poznań
7	Jacek	Witos	69100967234	Grunwaldzka	34	Warszawa
8	Maksymilian	Markowski	67121209878	NULL	NULL	NULL

 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Czasami chcielibyśmy uzyskać komplet danych z jednej tabeli, nawet jeżeli nie są one powiązane z danymi w innych tabelach. Umożliwia nam to złączenie zewnętrzne. Wynikiem lewo- lub prawostronnego złączenia zewnętrznego jest zbiór wierszy łączonych tabel, dla których wartości kolumn określonych jako warunek złączenia są takie same; zbiór ten uzupełniony jest pozostałymi wierszami z lewej lub prawej łączonej tabeli. Nieistniejące wartości reprezentowane są w wyniku złączenia przez wartość NULL.


OUTER JOIN:

LEFT OUTER JOIN – nie pomija wyników z lewej tabeli


RIGHT OUTER JOIN – nie pomija wyników z prawej tabeli

OUTER JOIN – nie pomija wyników z żadnej tabeli

Slajd
11



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language


Złączenia zewnętrzne

Złączenia zewnętrzne stosowane są do wyświetlania kompletnych informacji o wszystkich obiektach danego typu, nawet jeżeli nie istnieją powiązane z nimi obiekty innego typu.

 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Podsumowanie OUTER JOIN

Człowiek - najlepsza inwestycja

B2E
 BUSINESS TO EDUCATION

SQL
 Structured Query Language

Złączenie tabeli z nią samą

- Złączenie tabeli z nią samą jest jedną z technik języka SQL, odpowiadającą użyciu zmiennych w proceduralnych językach programowania.

Ważne zasady:

- Trzeba utworzyć różne aliasy dla łączonej tabeli i w ramach zapytania konsekwentnie odwoływać się do aliasów, a nie do nazwy tabeli.
- Każdy rekord, w którym wartości atrybutu złączenia będą sobie równe, zostanie dodany do wyniku złączenia, co spowoduje powstanie duplikatów rekordów.


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Złączenie tabeli z nią samą wykonywane jest w taki sam sposób, jak omawiane do tej pory złączenia różnych tabel. Chociaż serwery bazodanowe nie tworzą kopii złączonej tabeli, to wszystkie operacje przeprowadzane są tak, jakby dotyczyły dwóch identycznych tabel. Złączenie tabeli z nią samą stosujemy, kiedy chcemy wybrać rekordy z tabeli na podstawie wspólnych wartości atrybutów rekordów tej samej tabeli.




B2E
 BUSINESS TO EDUCATION

SQL
 Structured Query Language

Złączenie tabeli z nią samą

```

SELECT p.imie, p.nazwisko, p.pesel, m.imie, m.nazwisko
FROM pracownicy p INNER JOIN pracownicy m
ON p.przełożony = m.nr_pracownika
    
```

	imie	nazwisko	pesel	imie	nazwisko
1	Jan	Kowalski	82091104357	Jacek	Witos
2	Karol	Nowak	80010123987	Jan	Kowalski
3	Marzena	Przepiórka	89121203456	Jan	Kowalski
4	Paweł	Burzych	78032309123	Jacek	Witos
5	Marek	Makłowicz	54013112345	Jacek	Witos
6	Magdalena	Naramowicka	77121312098	Jan	Kowalski
7	Jacek	Witos	69100967234	Jacek	Witos
8	Maksymilian	Markowski	67121209878	Jacek	Witos


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Złączenia tabeli z samą sobą są często wykorzystywane do rekurencyjnego odczytania danych, na przykład informacji o podwładnych (podwładny osoby X może być przełożonym osoby Y, która z kolei może być przełożonym osoby Z). W testowej bazie danych nie ma zapisanych takich zależności. Przykład jest czysto szkoleniowy.

Slajd
14

SQL
Structured Query Language

Unie

- UNION
 - Umożliwia połączenie dwóch tabel
 - Nazwy kolumn muszą być takie same (można użyć alias)

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel
1	Kowalski	Jan	2300.00	250.00	82091104357
2	Nowak	Karol	2700.00	100.00	80010123987
3	Przepiórka	Marzena	2700.00	NULL	89121203456
4	Burzych	Paweł	1900.00	500.00	78032309123
5	Makłowicz	Marek	2000.00	NULL	54013112345
6	Naramowicka	Magdalena	2100.00	200.00	77121312098
7	Witos	Jacek	3000.00	500.00	69100967234
8	Markowski	Maksymilian	2000.00	NULL	67121209878

Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Unie umożliwiają łączenie tabel. Jeśli chcemy wyświetlić wszystkie osoby bez względu czy są to klienci czy pracownicy – możemy użyć właśnie polecenia UNION (warunkiem są takie same nazwy kolumn).

Ciekawostka: działania na zbiorach sortują wyniki i usuwają duplikaty. Wyjątkiem jest UNION ALL

Slajd
15

SQL
Structured Query Language

Unie










```
SELECT nazwisko , imie, pesel FROM pracownicy
UNION
SELECT nazwisko , imie, pesel FROM klienci;
```

	nazwisko	imie	pesel
1	Burzych	Paweł	78032309123
2	Kowalski	Jan	82091104357
3	Makłowicz	Marek	54013112345
4	Marcin	Kaniewski	73122990123
5	Markowski	Maksymilian	67121209878
6	Mikołaj	Wiśniewski	82040112389
7	Naramowicka	Magdalena	77121312098
8	Nowak	Karol	80010123987
9	Przepiórka	Marzena	89121203456
10	Witos	Jacek	69100967234

Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Powyższy przykład pokazuje jak w SQL połączyć dwie tabele razem. Polecenie UNION eliminuje duplikaty, a co za tym idzie sortuje wynik połączenia obu tabel. Rezultat będzie posortowany po nazwisku, a w dalszej kolejności po imieniu i numerze pesel

Człowiek - najlepsza inwestycja

<p>Slajd 16</p>	  <p>SQL Structured Query Language</p> <h3>Wyłączenie</h3> <ul style="list-style-type: none"> EXCEPT – wyświetlenie tylko kolumn z jednej tabeli, które nie znajdują się w drugiej Przykład, pokaż wszystkich pracowników, którzy nie są klientami: <pre>SELECT nazwisko , imie, pesel FROM pracownicy EXCEPT SELECT nazwisko , imie, pesel FROM klienci;</pre> <p> Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p>	<p>Jeśli chcemy wyszukać wartości tabeli A, które nie znajdują się w tabeli B – korzystamy z polecenia EXCEPT. Konstrukcja jest identyczna jak przy UNION.</p>
<p>Slajd 17</p>	  <p>SQL Structured Query Language</p> <h3>Funkcje agregujące</h3> <ul style="list-style-type: none"> COUNT - czyli zliczanie ilości zwróconych wierszy z zapytania, SUM - wynik z dodawania wartości, ze wszystkich elementów występujących w zapytaniu (każda kolumna osobno), AVG - wartość średnia (arytmetyczna), ze wszystkich wartości występujących w zapytaniu (również każda kolumna osobno), MIN - wartość najmniejsza ze wszystkich występujących w kolumnie, MAX - wartość największa ze wszystkich dostępnych w kolumnie. <p> Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p>	<p>Wprowadzenie do funkcji agregujących. Każda z nich pochodzi z j. angielskiego – w ten sposób można łatwo zapamiętać. Z wyjątkiem funkcji COUNT(*), wszystkie funkcje agregujące wykonują krok eliminacji NULL tak, że wartości NULL nie są uwzględniane w ostatecznym wyniku obliczeń.</p>
<p>Slajd 18</p>	  <p>SQL Structured Query Language</p> <h3>Grupowanie GROUP BY</h3> <ul style="list-style-type: none"> GROUP BY: <ul style="list-style-type: none"> Umożliwia grupowanie wyników względem warunku, np. <ul style="list-style-type: none"> Pokaż ilość pracowników w podziale na miejsca pracy <pre>SELECT COUNT(*), nr_miejsca FROM pracownicy GROUP BY nr_miejsca;</pre> <p> Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p>	<p>SQL jest językiem zapytań zorientowanym na przetwarzanie zbiorów, a nie pojedynczych rekordów, dlatego jego integralnym składnikiem są polecenia grupujące dane. Właśnie te polecenia, wraz z funkcjami sumującymi, stosowane są w praktyce do uzyskiwania odpowiedzi na złożone zapytania, dotyczące zawartości bazy danych. Przykład ilustruje użycie klauzuli GROUP BY.</p>

Człowiek - najlepsza inwestycja





KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
19


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY


SQL
 Structured Query Language

COUNT

- Zliczanie liczby rekordów
- Przykład liczba pracowników w poszczególnych miastach


```
SELECT miasto, COUNT(imie)
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejscas = miejsca.nr_miejscas GROUP BY miasto;
```


	miasto	(No column name)
1	Kraków	2
2	Poznań	3
3	Warszawa	1


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Funkcja COUNT liczy wystąpienia bazy danych w tym przypadku zliczamy dla każdego z miast – ilu pracowników jest zatrudnionych w konkretnej lokalizacji. W klauzuli *group by* wskazujemy po czym mają być grupowane dane.

 Slajd
20


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY


SQL
 Structured Query Language

SUM

- SUM – sumuje wartości liczbowe, np. suma pensji w poszczególnych miastach

```
SELECT miasto, SUM(placa_zasadnicza)
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejscas = miejsca.nr_miejscas GROUP BY miasto;
```

	miasto	(No column name)
1	Kraków	3900.00
2	Poznań	7100.00
3	Warszawa	3000.00


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Przykład wykorzystania funkcji SUM

 Slajd
21


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

AVG

- AVG – wylicza średnią
- Średnia łączna płaca w różnych miastach:

```
SELECT miasto, AVG(placa_zasadnicza + ISNULL(premia,0))
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejscas = miejsca.nr_miejscas GROUP BY miasto;
```

	miasto	(No column name)
1	Kraków	2200.000000
2	Poznań	2550.000000
3	Warszawa	3500.000000


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY



Slajd
22




SQL
 Structured Query Language


MIN i MAX

- MIN – to wartość minimalna, a MAX wartość maksymalna

```

SELECT miasto, MIN(placa_zasadnicza + ISNULL(premia,0)) as Minimum,
MAX(placa_zasadnicza + ISNULL(premia,0)) as Maksimum
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejscas = miejsca.nr_miejscas GROUP BY miasto;
  
```

	miasto	Minimum	Maksimum
1	Kraków	2000.00	2400.00
2	Poznań	2300.00	2800.00
3	Warszawa	3500.00	3500.00


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Slajd
23




SQL
 Structured Query Language

Grupowanie

- HAVING – umożliwia stosowanie warunku na podstawie funkcji agregujących
- Przykład – wyświetli miasta z pensją średnią większą niż 2500

```

SELECT miasto, AVG(placa_zasadnicza + ISNULL(premia,0)) as Średnia
FROM pracownicy INNER JOIN miejsca
ON pracownicy.nr_miejscas = miejsca.nr_miejscas GROUP BY miasto
having AVG(placa_zasadnicza + ISNULL(premia,0)) > 2500;
  
```

	miasto	Średnia
1	Poznań	2550.000000
2	Warszawa	3500.000000


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Klauzula HAVING służy do ograniczania zbioru wierszy, zwracanych w wyniku wykonania klauzuli GROUP BY. Między HAVING a GROUP BY występuje zależność podobna do tej, która wiąże klauzulę WHERE z poleceniem SELECT. W przeciwieństwie do klauzuli WHERE, która operuje na wierszach tabel podawanych w zapytaniu, HAVING działa na zbiorze wierszy wynikowych

Slajd
24




SQL
 Structured Query Language

Podzapytania

- Mniej wydajne niż JOIN czy UNION
- Używane gdy nie ma innej możliwości
- SELECT w SELECT:

```

SELECT nazwisko, imie, pesel FROM pracownicy
WHERE nr_miejscas = (SELECT nr_miejscas FROM miejsca WHERE miasto='Kraków');
  
```

	nr_miejscas	nr_miejscas	nr_miejscas
1	Burzych	Paweł	78032309123
2	Makłowicz	Marek	54013112345


 Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Podzapytanie jest zapytaniem umieszczonym wewnątrz innego zapytania, tzw. zapytania zewnętrznego. Podzapytania najczęściej umieszcza się w warunkach w klauzulach WHERE i HAVING zapytania zewnętrznego, niektóre SZBD dopuszczają również stosowanie podzapytań w klauzulach SELECT i FROM. Podzapytania stosuje się w ostateczności, gdy nie można zastosować połączenia typu UNION lub JOIN.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
25

SQL
Structured Query Language

Podzapytania skorelowane

- W zapytaniach zawierających podzapytania skorelowane, zapytanie wewnętrzne jest realizowane oddzielnie dla każdego wiersza z zapytania zewnętrznego.

```

SELECT nazwisko, imie, pesel,
(
    SELECT miejsce M FROM miejsca
    WHERE M.nr_miejsca = P.nr_miejsca -- faktyczna korelacja
) miejsce
FROM pracownicy P
    
```

Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

W zapytaniach zawierających podzapytania skorelowane, zapytanie wewnętrzne jest realizowane oddzielnie dla każdego wiersza z zapytania zewnętrznego.

Slajd
26

SQL
Structured Query Language

Ćwiczenia

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przełożony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87

Lekcja 2 - Języki programowania SQL, środowisko, wprowadzenie do języka SQL

Ćwiczenie 1:

Pokaż w jakich miejscowościach pracują pracownicy

Ćwiczenie 2:

Pokaż pracowników z miejscowości Warszawa

Ćwiczenie 3:

Pokaż wszystkie osoby (klienci i pracownicy)

Ćwiczenie 4:



Pokaż pracowników i ich przełożonych

Ćwiczenie 5:

Pokaż najniższą pensję całej firmie

Ćwiczenie 6:

Człowiek - najlepsza inwestycja

	<p>Pokaż średnią pensję w podziale na miejscowości</p> <p>Ćwiczenie 7:</p> <p>Pokaż miejscowość, gdzie pracuje najwięcej pracowników</p> <p>Ćwiczenie 8:</p> <p>Pokaż ilość pracowników w każdej miejscowości</p> <p>Ćwiczenie 9:</p> <p>Pokaż średnią pensję specjalistów i managerów</p> <p>Ćwiczenie 10:</p> <p>Pokaż najwyższą pensję średnią premie w podziale na stanowiska</p>	
Slajd 27	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz zapytanie zwracające 3 uczniów z tabeli TBL_UCZEN, których wartość kolumny IMIE to Anna <p>Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p>	<pre>SELECT TOP 3 * FROM TBL_UCZEN WHERE IMIE=„Anna”</pre>
Slajd 28	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz zapytanie zwracające z tabeli TBL_UCZEN średnią ocen (kolumna OCENA) wszystkich uczniów klasy 3A (kolumna KLASA) <p>Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p>	<pre>SELECT AVG(OCENA) FROM TBL_UCZEN WHERE KLASA=„3A”</pre>







Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 29</p>	<div>   </div> <div> <p>SQL Structured Query Language</p> </div> <div> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz zapytanie zwracające z tabeli TBL_UCZEN ilość uczniów klasy 3A (kolumna KLASA) </div> <div>  <p>Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p> </div>	<pre>SELECT COUNT(*) FROM TBL_UCZEN WHERE KLASA=„3A”</pre>
<p>Slajd 30</p>	<div>   </div> <div> <p>SQL Structured Query Language</p> </div> <div> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz zapytanie zwracające średnią ocen (kolumna OCENA) uczniów każdej klasy (kolumna KLASA) z tabeli TBL_UCZEN </div> <div>  <p>Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych</p> </div>	<pre>SELECT AVG(OCENA) FROM TBL_UCZEN GROUP BY KLASA=„3A”</pre>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





Podsumowanie

JOIN	Łączy dane z tabel
INNER JOIN ... ON ...	Pokazuje tylko wyniki, które są łączy wspólna wartość
LEFT/RIGHT OUTER JOIN	Nie wyklucza wartości
GROUPING	Grupuje wyniki po zadanym warunku
HAVING	Pozwala na użycie funkcji agregujących w warunku WHERE
UNION	Łączy tabele
EXCEPT	Pokazuje wartości z jednej tabeli, które nie występują w drugiej
MIN i MAX	Pozwala wyświetlić wartość maksymalną i minimalną
COUNT	Zlicza wiersze
SUM	Sumuje wartości
AVG	Liczy wartość średnią

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI



Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego



Lekcja 3 - Złożone zapytania, grupowanie, agregacja danych

Tabela podsumowująca poznane wyrażenia

9.3.3 Ćwiczenia

Ćwiczenia zostały przedstawione na slajdach 26-30. Uczniowie w ramach ćwiczenia będą musieli napisać zapytania SQL na bazie tabel przygotowanych przez szkoleniowca.

9.3.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie będą potrafili budować złożone zapytania SQL, łączyć wiele tabel ze sobą na różne sposoby, grupować dane w tabelach, wyliczać średnie, wartości minimalne i maksymalne, liczyć wiersze w tabeli. Uczniowie powinni być w stanie budować zapytania złożone z podzapytań.

9.4 Lekcja 4 – DML

9.4.1 Cel lekcji

Celem lekcji jest wyjaśnienie języka DML, szczegółowe zapoznanie z podstawowymi poleceniami tego języka: INSERT, UPDATE oraz DELETE.

Człowiek - najlepsza inwestycja












KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI


UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY




9.4.2 Treść - slajdy z opisem

<p>Slajd 1</p>	  <div>SQL Structured Query Language</div> <hr/> <h1>SQL</h1> <h2>Structured Query Language</h2> <p>Lekcja 4 DML Data Manipulation Language</p> <p><small>Człowiek - najlepsza inwestycja</small></p> <p><small>KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI</small></p> <p><small>UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY</small></p> <p><small>Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego</small></p> <p> SQL Structured Query Language Lekcja 4</p>	<p>DML (Data Manipulation Language) służy do wykonywania operacji na danych – do ich umieszczania w bazie, kasowania, przeglądania oraz dokonywania zmian. Najważniejsze polecenia z tego zbioru to: INSERT, UPDATE oraz DELETE</p>
<p>Slajd 2</p>	  <div>SQL Structured Query Language</div> <hr/> <h2>Przypomnienie</h2> <ul style="list-style-type: none"> • JOIN • INNER JOIN ... ON ... • (LEFT) (RIGHT OUTER JOIN • GROUPING • HAVING • UNION EXCEPT • MIN i MAX • COUNT • SUM i AVG <p> Lekcja 4 - DML (Data Manipulation Language)</p>	
<p>Slajd 3</p>	  <div>SQL Structured Query Language</div> <hr/> <h2>Data Manipulation Language</h2> <p>DML (Data Manipulation Language) służy do wykonywania operacji na danych – do ich umieszczania w bazie, kasowania, przeglądania oraz dokonywania zmian. Najważniejsze polecenia z tego zbioru to:</p> <ul style="list-style-type: none"> ◦ INSERT – umieszczenie danych w bazie, ◦ UPDATE – zmiana danych, ◦ DELETE – usunięcie danych z bazy. <p> Lekcja 4 - DML (Data Manipulation Language)</p>	<p>Część osób do DML zalicza także zapytania typu SELECT, jednak przeglądanie danych nie do końca można zaliczyć jako manipulacja danymi – w tej kwestii informatycy nie są zgodni.</p>

Slajd
4

**B2E**
BUSINESS TO EDUCATION


SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY


SQL
Structured Query Language

DAILY GROUP

Do wstawiania wierszy do tabeli służy polecenie INSERT. Najprostszą wersję tego polecenia przedstawiono na przykładzie. Polecenie zaczyna się od słów kluczowych INSERT INTO po którym podaje się nazwę tabeli, słowo kluczowe VALUES i w nawiasie listę wartości oddzielonych przecinkami. Wartości na tej liście odpowiadają kolejnym atrybutom relacji (w kolejności w jakiej atrybuty te zostały zdefiniowane). Wpisami na liście wartości mogą być: słowa kluczowe NULL i DEFAULT oraz konkretne wartości atrybutów i ewentualnie podzapytania zwracające jedną wartość. Jeżeli zamiast konkretnej wartości poda się NULL, wówczas nowy wiersz będzie miał pustą wartość atrybutu odpowiadającego pozycji na której wpisano NULL. Podanie DEFAULT spowoduje, że w odpowiedniej kolumnie zostanie zapisana jego wartość domyślna.

Slajd
5

**B2E**
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

DAILY GROUP

Przykład umieszczania danych w bazie. Za pomocą dwóch wierszy polecenia INSERT wstawiamy do bazy danych numer klienta, nazwisko, imię oraz PESEL klienta. Poniżej sprawdzamy zawartość bazy – upewniamy się że wartości zostały odpowiednio umieszczone w bazie danych.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
6

 **B2E**
BUSINESS TO EDUCATION

 **SZCZECIŃSKI PARK**
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

INSERT – wartości puste

- INSERT umożliwia wstawienie wiersza bez uzupełniania wszystkich wartości
- Sposób 1:

```
INSERT INTO klienci VALUES (3, 'Galaś', 'Jan', NULL);
```

- Sposób 2:

```
INSERT INTO klienci(nr_klienta, nazwisko, imie) VALUES (4, 'Zimna', 'Maria');
```

 **DAILY**
GROUP

Lekcja 4 - DML (Data Manipulation Language)

Umieszczanie pustych wartości w wierszach można zrealizować w dwojaki sposób: albo używamy wartości pustej NULL w miejscach gdzie nie znamy wartości, albo wymieniamy po nazwie tabeli wartości które zamierzamy uzupełnić. Oba sposoby dają identyczny efekt

Slajd
7

 **B2E**
BUSINESS TO EDUCATION

 **SZCZECIŃSKI PARK**
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

INSERT – wartość domyślna

- Do wstawienia wartości domyślnej służy DEFAULT
- Przykład:

```
INSERT INTO klienci VALUES (5, 'Wiśniewska', 'Joanna', DEFAULT)
```

nr_klienta	nazwisko	imie	pesel
1	Wiśniewski	Mikołaj	82040112389
2	Kaniewski	Marcin	73122990123
3	Galaś	Jan	NULL
4	Zimna	Maria	NULL
5	Wiśniewska	Joanna	NULL

 **DAILY**
GROUP

Lekcja 4 - DML (Data Manipulation Language)

Kolumny tabeli posiadają często wartość domyślną. Użycie polecenia DEFAULT w poleceniu SQL umożliwia wstawienie wartości domyślnej. W przykładzie użyto DEFAULT dla kolumny pesel, której wartością domyślną jest wartość pusta - NULL.

Slajd
8

 **B2E**
BUSINESS TO EDUCATION

 **SZCZECIŃSKI PARK**
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

INSERT z SELECT

- Wstawianie wielu wierszy z innej tabeli
- INSERT INTO tabela SELECT ... FROM tabela 2

```
INSERT INTO klienci(nazwisko, imie, pesel)
SELECT imie, nazwisko, pesel FROM pracownicy;
```

 **DAILY**
GROUP

Lekcja 4 - DML (Data Manipulation Language)

Wstawianie wielu wierszy pojedynczo jest bardzo czasochłonne. W języku SQL możemy wykorzystać zawartość innych tabel do uzupełnienia tabeli. W przykładzie użyto danych z tabeli pracownicy w celu wstawienia danych do tabeli klienci.

Człowiek - najlepsza inwestycja



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

UPDATE

- UPDATE służy do aktualizacji istniejących w tabeli wierszy
- Składnia polecenia:

UPDATE nazwa_tabeli SET kolumna = nowa_wartość WHERE kolumna2=wartość



Lekcja 4 - DML (Data Manipulation Language)



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

UPDATE – przykład 1

- Przykład:

Zaktualizuj premie managerów i ustaw na 500zł

```
UPDATE pracownicy SET premia = 500 WHERE stanowisko = 'Manager'
```

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przelozony
1	Kowalski	Jan	2300.00	500.00	82091104357	Manager	1	7
2	Nowak	Karol	2700.00	100.00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700.00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900.00	500.00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000.00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100.00	200.00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000.00	500.00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000.00	NULL	67121209878	Młodszy specjalista	NULL	7



Lekcja 4 - DML (Data Manipulation Language)

Polecenie zmienia premię na 500 dla wszystkich rekordów, gdzie stanowisko to Manager

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY




Slajd 11	<div>   <div> SQL Structured Query Language </div> </div> <h3>UPDATE – przykład 2</h3> <ul style="list-style-type: none"> Przykład: Powiększ wynagrodzenie Kowalskiego i Nowaka o 20% <pre>UPDATE pracownicy SET placa_zasadnicza = placa_zasadnicza*1.2 WHERE nazwisko = 'Kowalski' OR nazwisko = 'Nowak' UPDATE pracownicy SET placa_zasadnicza = placa_zasadnicza*1.2 WHERE nazwisko IN ('Kowalski', 'Nowak');</pre> <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	Oba polecenia wykonują taką samą operację, tj. powiększają kwotę płacy zasadniczej o 20% (mnożą płacę przez 1,2) w rekordach gdzie nazwisko to Nowak lub Kowalsk
Slajd 12	<div>   <div> SQL Structured Query Language </div> </div> <h3>UPDATE – przykład 3</h3> <ul style="list-style-type: none"> Przykład: Zmień nazwisko pracownika z Naramowicka na Naramowicka-Przybylska <pre>UPDATE pracownicy SET nazwisko = 'Naramowicka-Przybylska' WHERE nazwisko = 'Naramowicka'</pre> <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	Przykład aktualizuje pole nazwisko gdzie nazwisko to Naramowicka
Slajd 13	<div>   <div> SQL Structured Query Language </div> </div> <h3>UPDATE - ostrzeżenie</h3> <p>UWAGA!!!!</p> <ul style="list-style-type: none"> Proszę pamiętać, aby w zapytaniu UPDATE definiować odpowiednie wartości dla WHERE. W przypadku braku zdefiniowanego warunku WHERE w tabeli zostaną zmienione WSZYSTKIE wiersze! <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	Bardzo ważna uwaga: wiele początkujących osób zapomina dodać klauzulę WHERE do polecenia UPDATE, skutkuje to zastąpieniem wartości we wszystkich wierszach

Slajd
14



B2EBUSINESS TO EDUCATION




SQL
Structured Query Language

UPDATE - ostrzeżenie

UPDATE klienci SET nazwisko = 'Kowalski';

nr_klienta	nazwisko	imie	pesel
1	Kowalski	Mikołaj	82040112389
2	Kowalski	Marcin	73122990123
3	Kowalski	Jan	NULL
4	Kowalski	Maria	NULL
5	Kowalski	Joanna	NULL
6	Kowalski	Kowalski	82091104357
7	Kowalski	Nowak	80010123987
8	Kowalski	Przepiórka	89121203456
9	Kowalski	Burzych	78032309123
10	Kowalski	Makłowski	54013112345
11	Kowalski	Naramo...	77121312098
12	Kowalski	Witos	69100967234

 Lekcja 4 - DML (Data Manipulation Language)

Bardzo ważna uwaga: wiele początkujących osób zapomina dodać klauzulę WHERE do polecenia UPDATE, skutkuje to zastąpieniem wartości we wszystkich wierszach

Slajd
15



B2EBUSINESS TO EDUCATION



SQL
Structured Query Language

DELETE

- DELETE używane jest do usuwania istniejących wierszy z tabeli.
- Składnia:
DELETE FROM nazwa_tabeli WHERE nazwa_kolumny=wartosc;

 Lekcja 4 - DML (Data Manipulation Language)

Poznaliśmy sposób umieszczania danych w tabeli, ich modyfikacji – teraz czas na usuwania danych. DELETE to ostatnie z wyrażeń DML. Podobnie jak w przypadku UPDATE, należy zwrócić szczególną uwagę na warunek WHERE, tak aby nie skasować wszystkich danych z tabeli.

Slajd
16



B2EBUSINESS TO EDUCATION



SQL
Structured Query Language

DELETE - przykład

- Usuwanie z bazy pracownika o nazwisku Nowak

DELETE FROM pracownicy WHERE nazwisko = 'Nowak';

(1 row(s) affected)

 Lekcja 4 - DML (Data Manipulation Language)

Przykład usuwania rekordu – poniżej widzimy odpowiedź systemu. Widzimy, że został usunięty jeden wiersz.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
17

SQL
Structured Query Language

DELETE z podzapytaniem

- Usuwamy z bazy wszystkich pracowników z najwyższą pensją

```
DELETE FROM pracownicy WHERE placa_zasadnicza =
(SELECT MAX(placa_zasadnicza) FROM pracownicy)
```

Lekcja 4 - DML (Data Manipulation Language)

W warunku WHERE w poleceniu DELETE możemy zagnieżdżać podzapytania – podobnie jak w SELECT.

Slajd
18

SQL
Structured Query Language

Ćwiczenia

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przełożony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87

Lekcja 4 - DML (Data Manipulation Language)

Ćwiczenie 1:

Wstaw dwie miejscowości do tabeli Miejsca

Ćwiczenie 2:

Dodaj po dwóch pracowników do każdej z wstawionych w ćw. 1 lokalizacji

Ćwiczenie 3:

Podnieś wszystkim premie o 30zł, pamiętaj o osobach, które dotychczas nie miały premii

Ćwiczenie 4:

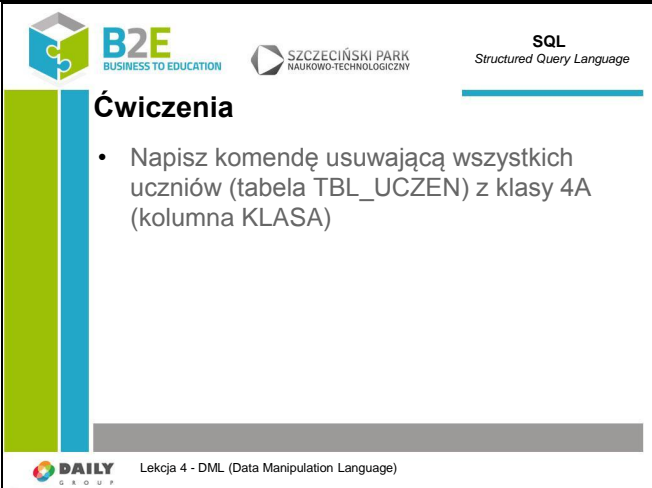
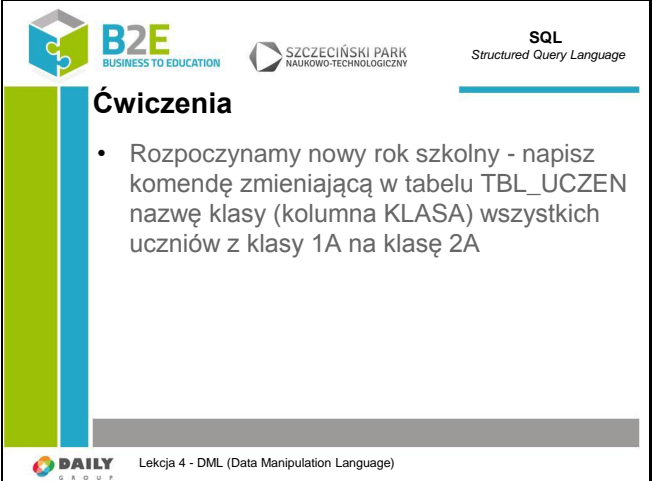
Zatrudniamy naszych klientów, wstaw do tabeli pracownicy osoby z tabeli klienci

Ćwiczenie 5:

Zmień miejsce zatrudnienia osób pracujących w Warszawie – przenosimy biuro do Szczecina

Ćwiczenie 6:

Człowiek - najlepsza inwestycja

	<p>Rezygnujemy z biura w Warszawie – usuń lokalizację z tabeli miejsca</p> <p>Ćwiczenie 7: Zwiększ pensje zasadnicze pracowników ze Szczecina o 30%</p> <p>Ćwiczenie 8: Osoby, które dotychczas nie miały przypisanej lokalizacji zostają przypisane do Szczecina – uaktualnij bazę danych</p> <p>Ćwiczenie 9: Znaleźliśmy nowych klientów – wprowadź kilka dodatkowych osób</p> <p>Ćwiczenie 10: Zaktualizuj nazwisko pani Marzeny – po ślubie zmieniła nazwisko na Nowak</p>	
Slajd 19	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz komendę usuwającą wszystkich uczniów (tabela TBL_UCZEN) z klasy 4A (kolumna KLASA) <p>Lekcja 4 - DML (Data Manipulation Language)</p>	<pre>DELETE FROM TBL_UCZEN WHERE KLASA="4A"</pre>
Slajd 20	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Rozpoczynamy nowy rok szkolny - napisz komendę zmieniającą w tabelu TBL_UCZEN nazwę klasy (kolumna KLASA) wszystkich uczniów z klasy 1A na klasę 2A <p>Lekcja 4 - DML (Data Manipulation Language)</p>	<pre>UPDATE TBL_UCZEN SET KLASA = "2A" WHERE KLASA = "1A"</pre>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 21	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz komendę dodającą Jana Kowalskiego do klasy 1A – tabela TBL_UCZEN ma kolumny: <ul style="list-style-type: none"> IMIE NAZWISKO KLASA <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	<pre>INSERT INTO TBL_UCZEN(IMIE, NAZWISKO, KLASA) VALUES („Jan”, „Kowalski”, „1A”)</pre>
Slajd 22	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz komendę wstawiającą do tabeli z poprzedniego ćwiczenia uczniów: Maria Nowak, Jakub Wiśniewski bez przypisanej klasy. <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	<pre>INSERT INTO TBL_UCZEN(IMIE, NAZWISKO, KLASA) VALUES („Maria”, „Nowak”, NULL) INSERT INTO TBL_UCZEN(IMIE, NAZWISKO, KLASA) VALUES („Jakub”, „Wiśniewski”, NULL)</pre>
Slajd 23	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz komendę aktualizującą uczennicę Marię Nowak – przyporządkującą ją do klasy 3A <div>  Lekcja 4 - DML (Data Manipulation Language) </div>	<pre>UPDATE TBL_UCZEN SET KLASA = „1A” WHERE IMIE = „Maria” AND NAZWISKO=„Nowak”</pre>

Człowiek - najlepsza inwestycja

Slajd
24

SQL
Structured Query Language

Podsumowanie

INSERT	Wstawia wiersze do tabeli
UPDATE	Aktualizuje wartość wierszy
DELETE	Usuwa wiersze z tabeli

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

DAILY
Lekcja 4 - DML (Data Manipulation Language)

Tabela podsumowująca poznane
wyrażenia

9.4.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 19-23. Uczniowie napiszą instrukcje modyfikujące dane w bazie danych.

9.4.4 Opis założonych osiągnięć ucznia

We wcześniejszych lekcjach uczniowie zapoznali się z poleceniem SELECT, potrafili już odczytywać dane z bazy. W tej lekcji nauczyli się modyfikować dane – wstawiać nowe rekordy, aktualizować i modyfikować istniejące.

9.5 Lekcja 5 - DDL

9.5.1 Cel lekcji

Celem lekcji jest zapoznanie się z językiem DDL. W tej części kursu uczniowie zapoznają się z mechanizmami tworzącymi tabele w bazie danych. Przed przystąpieniem do definiowania tabel, uczniowie zostaną zapoznani z typami danych w języku SQL.

9.5.2 Treść - slajdy z opisem

Slajd
1

SQL
Structured Query Language

SQL Structured Query Language

Lekcja 5
DDL
Data Definition Language

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI










UNIA EUROPEJSKA
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

DAILY
SQL Structured Query Language Lekcja 5

Kolejna dziedzina języka SQL – DDL – Data Definition Language, służy do definiowania obiektów bazy danych. Nauczyliśmy się wyświetlać dane z bazy (SELECT) oraz je modyfikować (DML: INSERT, UPDATE, DELETE). Przyszedł czas na tworzenie, modyfikację i usuwanie baz danych, tabel i innych obiektów.

Człowiek - najlepsza inwestycja

Slajd 2	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Przypomnienie</h2> <ul style="list-style-type: none"> • INSERT • UPDATE • DELETE <div>  Lekcja 5 - DDL (Data Definition Language) </div>	UPDATE służy do aktualizacji istniejących danych w tabeli. INSERT używane jest do dodawania nowych wierszy do tabeli. INSERT pozwala na dodawanie nowych wierszy na dwa sposoby. W jednym z nich definiujemy nazwy kolumn, zaś w drugim pomijamy nazwy kolumn wprowadzając same wartości. W przypadku drugiego sposobu, musimy wziąć pod uwagę kolejność kolumn w tabeli. DELETE używane jest do usuwania istniejących wierszy z tabeli.
Slajd 3	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Data Definition Language</h2> <p>Dzięki DDL (Data Definition Language) można operować na strukturach, w których dane są przechowywane – czyli np. dodawać, zmieniać i kasować tabele lub bazy. Najważniejsze polecenia tej grupy to:</p> <ul style="list-style-type: none"> • CREATE (np. CREATE TABLE, CREATE DATABASE, ...) – utworzenie struktury (bazy, tabeli, indeksu itp.), • DROP (np. DROP TABLE, DROP DATABASE, ...) – usunięcie struktury, • ALTER (np. ALTER TABLE ADD COLUMN ...) – zmiana struktury (dodanie kolumny do tabeli, zmiana typu danych w kolumnie tabeli). <div>  Lekcja 5 - DDL (Data Definition Language) </div>	Wyjaśnienie, że pomimo wspólnego standardu istnieje wiele różnych rozszerzeń. Podstawowa składnia wszystkich rozszerzeń języka jest taka sama, jednak istnieje wiele różnic. W tym kursie zajmiemy się językiem SQL w ujęciu rozszerzenia TSQL, promowanym przez Microsoft.
Slajd 4	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Typy danych w SQL</h2> <ul style="list-style-type: none"> • Typy numeryczne • Data i czas • Typ znakowy • Typy binarne • Typy przestrzenne <div>  Lekcja 5 - DDL (Data Definition Language) </div>	Zanim będziemy mogli stworzyć tabelę musimy poznać typy danych (w tym kolumn), które możemy użyć w języku SQL. Większość z przedstawionych typów dotyczy wszystkich rozszerzeń języka SQL. Jednak warto pamiętać, że w tym kursie skupiamy się na języku TSQL.



Typy danych w SQL

- Typy numeryczne

Typ	Opis
Bigint	8-bajtowy typ numeryczny z zakresem: -2^{63} (-9,223,372,036,854,775,808) to $2^{63}-1$ (9,223,372,036,854,775,807).
Numeric	Wielkość typu zależy od precyzji. 1-9 – 5 bajtów, 10-19 – 9 bajtów, 20-28 – 13 bajtów, 29-38 – 17 bajtów.
bit	Wartości: 0, 1 lub NULL.
smallint	2-bajtowy typ numeryczny z zakresem: -2^{15} (-32,768) do $2^{15}-1$ (32,767).
decimal	Wielkość typu zależy od precyzji. 1-9 – 5 bajtów, 10-19 – 9 bajtów, 20-28 – 13 bajtów, 29-38 – 17 bajtów.
smallmoney	4-bajtowy typ do zapisu walut z zakresem: - 214,748.3648 do 214,748.3647.

Do wartości numerycznych zalicza się typy przedstawione w tabeli.



Typy danych w SQL


- Typy numeryczne

Typ	Opis
int	4-bajtowy typ numeryczny z zakresem: -2^{31} (-2,147,483,648) do $2^{31}-1$ (2,147,483,647).
tinyint	1-bajtowy typ numeryczny z zakresem: 0 – 255.
money	8-bajtowy typ do zapisu walut z zakresem: -922,337,203,685,477.5808 do 922,337,203,685,477.5807.
float	Typ zmiennoprzecinkowy z zakresem: - 1.79E+308 to -2.23E-308, 0 and 2.23E-308 to 1.79E+308. Ilość bajtów zależy od precyzji. 1-24 – 4 bajty (7 cyfr), 25-53 – 8 bajtów (15 cyfr).
real	4-bajtowy typ zmiennoprzecinkowy z zakresem: - 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E + 38. Typ real jest 24-cyfrowym typem float.


Zanim będziemy mogli stworzyć tabelę musimy poznać typy danych (w tym kolumn), które

możemy użyć w języku SQL. Większość z przedstawionych typów dotyczy wszystkich rozszerzeń języka SQL. Jednak warto pamiętać, że w tym kursie skupiamy się na języku TSQL

Slajd
7



B2E
 BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- **Data i czas**


Typ	Opis
date	10-znakowy typ zapisu daty z precyzją: 10.0. Domyślnym formatem jest: YYYY-MM-DD, natomiast domyślna wartość to 1900-01-01
datetimeoffset	8-bajtowy typ do zapisu daty z zakresu: 1 stycznia 1 roku n.e. do 31 grudnia roku 9999, liczone wg kalendarza gregoriańskiego oraz czas 24-godzinny z dokładnością do 100 ns. Uwzględnia przesunięcie strefy czasowej.
datetime2	8-bajtowy typ do zapisu daty z zakresu: 1 stycznia 1 roku n.e. do 31 grudnia 9999 oraz czas 24-godzinny z dokładnością do 100 ns.
smalldatetime	4-bajtowy typ do zapisu daty z zakresu 1900-01-01 do 2076-06-06 z dokładnością do 1 minuty.
datetime	8-bajtowy typ do zapisu daty z zakresu 1 stycznia 1753 do 31 grudnia 9999 z dokładnością do 0.00333 sekund (zaokrągla milisekundy do .000, .003, lub .007)
time	3 – 5-bajtowy typ do zapisu czasu z dokładnością do 100 ns. Ilość bajtów zależy od skali precyzji.




Lekcja 5 - DDL (Data Definition Language)

Dane zapisywane jako data i czas powinny zostać zaprezentowane za pomocą typów przedstawionych w tabeli.

Slajd
8



B2E
 BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- **Typ znakowy**

Typ	Opis.
char	Stało-znakowy typ o wielkości 1 – 8000 bajtów.
varchar(max)	Zmiennie-znakowy typ o długości max. 2 ³⁰ – 1 bajtów.
varchar	Zmiennie-znakowy typ o długości 1 – 8000 bajtów.
nchar	Stało-znakowy typ Unicode o wielkości 1 – 4000 bajtów.
nvarchar	Zmiennie-znakowy typ Unicode o wielkości 1 – 4000 bajtów.
nvarchar(max)	Zmiennie-znakowy typ Unicode o wielkości max. 2 ³⁰ – 1 bajtów.





Lekcja 5 - DDL (Data Definition Language)

Do wartości znakowych zalicza się typy przedstawione w tabeli

Człowiek - najlepsza inwestycja

Slajd
9

**B2E**
BUSINESS TO EDUCATION


SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- Typy binarne

Typ	Opis.
binary	Przechowywany najczęściej jako stały strumień bajtów typ o wielkości 1 – 8000 bajtów.
varbinary	Przechowywany najczęściej jako zmienny strumień bajtów typ o wielkości 1 – 8000 bajtów.

DAILY GROUP

Lekcja 5 - DDL (Data Definition Language)

Do wartości binarnych zalicza typy przedstawione w tabeli

Slajd
10

**B2E**
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- Typy przestrzenne

Typ	Opis.
geography	Typ do przechowywania danych geograficznych, zaimplementowany w .NET CLR. Wykorzystywany jest głównie do zapisu pozycji GPS. Uwzględnia krzywiznę Ziemi.
geometry	Typ do przechowywania typów geometrycznych (w szczególności figur), zaimplementowany w .NET CLR.

DAILY GROUP

Lekcja 5 - DDL (Data Definition Language)

SQL Server oferuje typy do reprezentowania obiektów geometrycznych i geograficznych

Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI


UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
11



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- **Pozostałe typy**


Typ	Opis
cursor	Typ danych dla zmiennych parametrów wyjściowych procedury, które zawierają doniesienie do kursora.
hierarchid	Typ o zmiennej długości danych, służący do reprezentowania pozycji danej informacji w hierarchii.
sql_variant	Typ danych, przechowujący różne wartości typów obsługiwanych przez SQL Server. Jest odpowiednikiem typu var w języku C#.
table	Typ tabelaryczny.
uniqueidentifier	Typ umożliwiający automatyczne generowanie unikalnych liczb binarnych w bazie danych.
xml	Typ XML-owy.




Lekcja 5 - DDL (Data Definition Language)

Pozostałe typy zostały przedstawione w tabeli

Slajd
12



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Typy danych w SQL

- **Zapamiętaj, że:**
 - SQL Server 2008+ oferuje, oprócz typów standardowych, typy do zapisu danych geograficznych oraz figur geometrycznych, często wykorzystywanych w systemach GIS.
 - Typ real to tak naprawdę float(24).
 - Do zapisu daty i czasu powinno używać się typów, jakie oferuje SQL Server 2008 R2, czyli: time, datetime2, datetimeoffset.



Lekcja 5 - DDL (Data Definition Language)

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 13</p>	  <p>SQL Structured Query Language</p> <h2>CREATE DATABASE</h2> <ul style="list-style-type: none"> Służy do tworzenia bazy danych <ul style="list-style-type: none"> Składnia: <pre>CREATE DATABASE nazwa_bazy</pre> <pre>CREATE DATABASE Test;</pre> <pre>USE Test; </pre>  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>Pamiętajmy, że SZBD może zawierać wiele baz. Często używamy skrótu myślowego i nazywamy bazą danych SZBD, faktycznie system zarządzania bazą danych zawiera wiele baz danych. USE wykorzystywany jest do wyboru bazy danych, w kontekście której zamierzamy pracować.</p>
<p>Slajd 14</p>	  <p>SQL Structured Query Language</p> <h2>CREATE TABLE</h2> <ul style="list-style-type: none"> Służy do tworzenia tabel <ul style="list-style-type: none"> Składnia: <pre>CREATE TABLE nazwa_tabeli</pre> <pre>(nazwa_kolumny1 typ_danych,</pre> <pre>nazwa_kolumny2 typ_danych,</pre> <pre>nazwa_kolumny3 typ_danych,</pre> <pre>...</pre> <pre>);</pre>  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>CREATE służy do tworzenia nowych tabel, w których później możemy przechowywać różnego rodzaju dane. Tworząc nową tabelę musimy podać jej nazwę, nazwę kolumny/kolumn oraz typ danych danej kolumny. Można też podawać dodatkowe informacje, podczas tworzenia tabeli takie jak np. kodowanie znaków itd.</p>
<p>Slajd 15</p>	  <p>SQL Structured Query Language</p> <h2>CREATE TABLE</h2> <ul style="list-style-type: none"> Przykład 1 <pre>CREATE TABLE klienci1(</pre> <pre>nr_klienta int,</pre> <pre>nazwisko nvarchar(50),</pre> <pre>imie nvarchar(50),</pre> <pre>pesel nchar(11)</pre> <pre>);</pre>  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>Przykład CREATE tworzący tabelę klienci. Tworząc tabelę trzeba zwrócić uwagę na to, jaki typ danych będziemy przechowywać, aby był on najlepiej dopasowany. Jeżeli przechowujemy wyłącznie liczby, to nie ma potrzeby tworzenia kolumny typu tekstowego itd.</p>

Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY




Slajd 16	<div>   <div> SQL Structured Query Language </div> </div> <h2>CREATE TABLE</h2> <ul style="list-style-type: none"> Przykład 2 – klucz i autonumerowanie <pre>CREATE TABLE klienci2 (nr_klienta int primary key IDENTITY(1,1), nazwisko nvarchar(50), imie nvarchar(50), pesel nchar(11));</pre> <ul style="list-style-type: none"> IDENTITY(1,1) – pole autonumerowane – zaczynamy od wartości 1 i zwiększamy przy każdym insercie o 1. <div>  Lekcja 5 - DDL (Data Definition Language) </div>	<p>Najczęściej chcemy posiadać kolumnę w tabeli, która będzie przechowywała unikalne wartości automatycznie numerowane. Zastosowanie tego jest bardzo proste. Dzięki takiemu rozwiązaniu każdy wiersz w tabeli posiada unikalny "identyfikator". W tym celu możemy wykorzystać IDENTITY oraz definicję klucza głównego. Ciekawostka: Z punktu widzenia programisty, IDENTITY jako sposób na klucz główny (PK), jest straszną bolączką ORMów (mapowań relacyjno obiektowych).</p>
Slajd 17	<div>   <div> SQL Structured Query Language </div> </div> <h2>CREATE TABLE</h2> <ul style="list-style-type: none"> Przykład 3 – wartości domyśle, możliwość wartości pustej <pre>CREATE TABLE klienci3 (nr_klienta int primary key IDENTITY(1,1) NOT NULL, nazwisko nvarchar(50) NULL, imie nvarchar(50) NOT NULL DEFAULT 'Jan', pesel nchar(11) NULL);</pre> <ul style="list-style-type: none"> DEFAULT – wartość domyślna NULL/NOT NULL – czy możliwa jest wartość pusta <div>  Lekcja 5 - DDL (Data Definition Language) </div>	<p>Tworząc tabelę mamy wpływ na to czy dana kolumna będzie dopuszczała wartości puste, bądź nie. Służy do tego wyrażenie NULL lub NOT NULL po typie zmiennej. Dodatkowo możemy kolumnie nadać wartość domyślną – polecenie DEFAULT.</p>
Slajd 18	<div>   <div> SQL Structured Query Language </div> </div> <h2>DROP DATABASE</h2> <ul style="list-style-type: none"> Służy do usuwania bazy danych <pre>CREATE DATABASE Test2; DROP DATABASE Test2;</pre> <div>  Lekcja 5 - DDL (Data Definition Language) </div>	<p>DROP DATABASE usuwa całą bazę danych. Warto zwrócić uwagę, że operacja jest nieodwracalna i należy z niej korzystać ze szczególną ostrożnością.</p>

Slajd
19



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

DROP TABLE

- Służy do usuwania tabeli

```
CREATE TABLE klienci4(  
  nr_klienta int primary key IDENTITY(1,1) NOT NULL,  
  nazwisko nvarchar(50) NULL,  
  imie nvarchar(50) NOT NULL DEFAULT 'Jan',  
  pesel nchar(11) NULL  
);  
  
DROP TABLE klienci4;
```




DAILY GROUP


Lekcja 5 - DDL (Data Definition Language)

DROP TABLE usuwa tabelę. Warto zwrócić uwagę, że operacja jest nieodwracalna i należy z niej korzystać ze szczególną ostrożnością.

Slajd
20



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

ALTER DATABASE

- Służy do dokonywania zmian bazy danych

```
ALTER DATABASE database_name  
SET  
{  
  { <optionspec> [ ,...n ] [ WITH <termination> ] }  
}  
  
<optionspec>::=  
{  
  <auto_option>  
  | <change_tracking_option>  
  | <cursor_option>  
  | <database_mirroring_option>  
  | <date_correlation_optimization_option>  
  | <db_encryption_option>  
  | <db_state_option>  
  | <db_update_option>  
  | <db_user_access_option>  
  | <external_access_option>  
  | <parameterization_option>  
  | <recovery_option>  
  | <service_broker_option>  
  | <snapshot_option>  
  | <sql_option>  
}  
  
<auto_option> ::=
```



DAILY GROUP

Lekcja 5 - DDL (Data Definition Language)

ALTER służy głównie do modyfikacji istniejących elementów w bazie. Początkujący użytkownicy będą głównie korzystali z polecenia **ALTER TABLE**, które będzie odnosiło się do tabel. Poleceniem tym możemy modyfikować tabele między innymi dodając i usuwając kolumny, zmieniając kolumny oraz typy kolumn. W TSQL posiadamy inne typy zapytań **ALTER**, np. ALTER DATABASE (odnosi się do bazy danych), ALTER FUNCTION (odnosi się do funkcji), ALTER USER (odnosi się do użytkowników), ALTER VIEW (odnosi się do widoków), ALTER PROCEDURE (odnosi się do procedur). Na slajdzie przedstawiono możliwe opcje ALTER DATABASE. Najważniejsze z nich to zmiana kodowania, szyfrowania, opcji kursorów.










Człowiek - najlepsza inwestycja









KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 21	  <p>SQL Structured Query Language</p> <h3>ALTER TABLE</h3> <ul style="list-style-type: none"> • Służy do modyfikacji tabeli • Podstawowa składnia ALTER TABLE: ALTER TABLE nazwa_tabeli [ADD DROP ALTER ..] nazwa_kolumny ... <p> Lekcja 5 - DDL (Data Definition Language)</p>	
Slajd 22	  <p>SQL Structured Query Language</p> <h3>ALTER TABLE</h3> <ul style="list-style-type: none"> • ADD COLUMN • Składnia: ALTER TABLE nazwa_tabeli ADD COLUMN nazwa_kolumny typ_danych; <pre>ALTER TABLE klienci3 ADD miejscowosc nvarchar(150);</pre> <p> Lekcja 5 - DDL (Data Definition Language)</p>	Przykład ALTER TABLE dodający kolumnę do istniejącej tabeli.
Slajd 23	  <p>SQL Structured Query Language</p> <h3>ALTER TABLE</h3> <ul style="list-style-type: none"> • DROP COLUMN • Składnia: ALTER TABLE nazwa_tabeli DROP COLUMN nazwa_kolumny typ_danych; <pre>ALTER TABLE klienci3 DROP COLUMN miejscowosc;</pre> <ul style="list-style-type: none"> • UWAGA! <ul style="list-style-type: none"> • Wraz z usunięciem kolumny zostaną usunięte wszystkie dane zapisane w kolumnie. <p> Lekcja 5 - DDL (Data Definition Language)</p>	Przykład ALTER TABLE usuwający kolumnę z istniejącej tabeli.

<p>Slajd 24</p>	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h2>ALTER TABLE</h2> <ul style="list-style-type: none"> • ALTER COLUMN • Składnia: <p>ALTER TABLE nazwa_tabeli ALTER COLUMN nazwa_kolumny typ_danych;</p> <pre>ALTER TABLE klienci3 ALTER COLUMN miejscowosc nvarchar(200);</pre> <div>  Lekcja 5 - DDL (Data Definition Language) </div>	<p>Przykład ALTER TABLE modyfikujący kolumnę w istniejącej tabeli.</p>
<p>Slajd 25</p>	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h2>SELECT INTO</h2> <ul style="list-style-type: none"> • SELECT INTO kopiuje dane z jednej tabeli do drugiej. • SELECT INTO najczęściej jest używane do tworzenia kopii bezpieczeństwa tabel. <p>Składnia SELECT INTO:</p> <pre>SELECT * INTO nowa_tabela [IN zewnetrzna_baza] FROM stara_tabela</pre> <div>  Lekcja 5 - DDL (Data Definition Language) </div>	<p>SELECT INTO – szybka metoda kopiowania tabel.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
26

SQL
Structured Query Language

SELECT INTO

```
SELECT * INTO pracownicy_kopia FROM pracownicy
```

```
SELECT * FROM pracownicy_kopia;
```

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przelozony
1	Kowalski	Jan	3312.00	500.00	82091104357	Manager	1	7
3	Przepiórka	Marzena	2700.00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900.00	500.00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000.00	NULL	54013112345	Specjalista	2	7
6	Naramowicka-Przybylska	Magdalena	2100.00	200.00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000.00	500.00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000.00	NULL	67121209878	Młodszy specjalista	NULL	7

Lekcja 5 - DDL (Data Definition Language)

Przykład SELECT INTO kopiujący dane z tabeli pracownicy.

Slajd
27

SQL
Structured Query Language

Ćwiczenia

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przelozony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

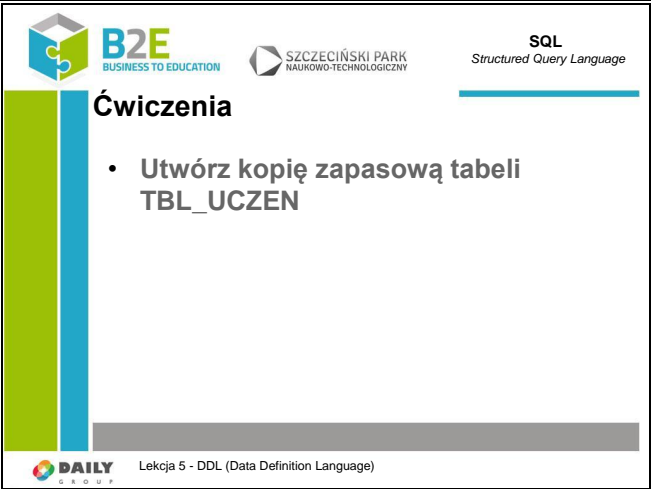
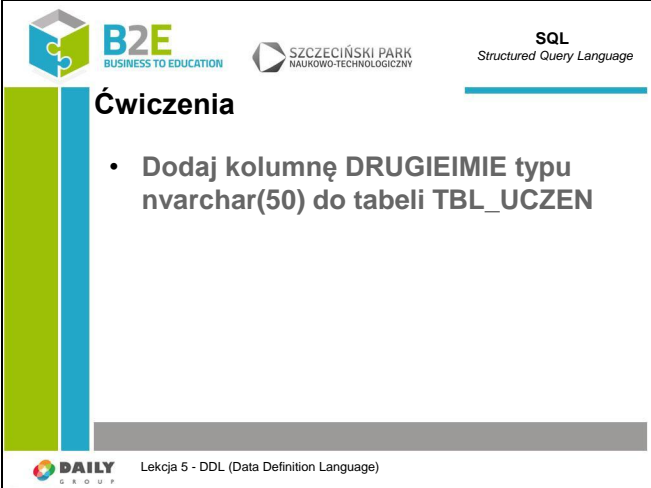
nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87

Lekcja 5 - DDL (Data Definition Language)









Ćwiczenie 1:

Utwórz nową bazę danych o nazwie Testowa_Inicjały

Człowiek - najlepsza inwestycja

	<p>Ćwiczenie 2: Utwórz nowe tabele zgodnie ze slajdem</p> <p>Ćwiczenie 3: Usuń tabelę miejsca</p> <p>Ćwiczenie 4: Zmodyfikuj tabelę klienci – dodaj kolumnę wartość zamówienia (typ decimal z dwoma miejscami po przecinku)</p> <p>Ćwiczenie 5: Zmodyfikuj kolumnę wartość zamówienia – zmień typ na money z domyślną wartością 0</p> <p>Ćwiczenie 6: Skopiuj tabelę klienci do tabeli kontrahenci</p> <p>Ćwiczenie 7: Usuń kolumnę wartość zamówienia w tabeli kontrahenci</p> <p>Ćwiczenie 8: Usuń bazę testowa</p>	
Slajd 28		<pre>SELECT * INTO TBL_UCZEN_BCKP FROM TBL_UCZEN</pre>
Slajd 29		<pre>ALTER TABLE TBL_UCZEN ADD COLUMN DRUGIEIMIE nvarchar(50);</pre>

Człowiek - najlepsza inwestycja

<p>Slajd 30</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> • Zmień długość kolumny DRUGIEIMIE na 150  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>ALTER TABLE TBL_UCZEN ALTER COLUMN DRUGIEIMIE nvarchar(150);</p>
<p>Slajd 31</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> • Utwórz bazę danych Szkoła  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>CREATE DATABASE SZKOŁA</p>
<p>Slajd 32</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> • Usuń bazę danych Szkoła  <p>Lekcja 5 - DDL (Data Definition Language)</p>	<p>DROP DATABASE SZKOŁA</p>

Człowiek - najlepsza inwestycja

Slajd
33



SQL
Structured Query Language

Podsumowanie

Najważniejsze polecenia DDL

CREATE	Tworzy nowe obiekty w bazie danych
DROP	Usuwa obiekty
ALTER	Modyfikuje obiekty
SELECT INTO	Tworzy kopię tabeli i kopiuje dane

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

DAILY Lekcja 5 - DDL (Data Definition Language)

Tabela podsumowująca poznane
wyrażenia

9.5.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 27-32. Uczniowie przyswoją wiedzę dotyczącą tworzenia i modyfikacji tabel w bazie danych.

9.5.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie będą wiedzieć jak tworzy się i modyfikuje tabele w bazie danych.

9.6 Lekcja 6 - DCL, uprawnienia, użytkownicy

9.6.1 Cel lekcji

Celem lekcji jest zapoznanie uczniów z pojęciem użytkownika i loginu w bazie danych. Uczniowie dowiedzą się jak przyznawać i odbierać uprawnienia do obiektów bazodanowych użytkownikom i grupom użytkowników.

9.6.2 Treść - slajdy z opisem

Slajd
1



SQL
Structured Query Language

SQL Structured Query Language

Lekcja 6 DCL, uprawnienia, użytkownicy

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI










UNIA EUROPEJSKA
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

DAILY SQL Structured Query Language Lekcja 5

DCL (Data Control Language) – to język wykorzystywany do nadawania uprawnień użytkownikom i grupom do obiektów baz danych. W tej lekcji przyjrzymy się uprawnieniom, użytkownikom i nauczymy się tworzyć polecenia SQL do sterowania użytkownikami i uprawnieniami.

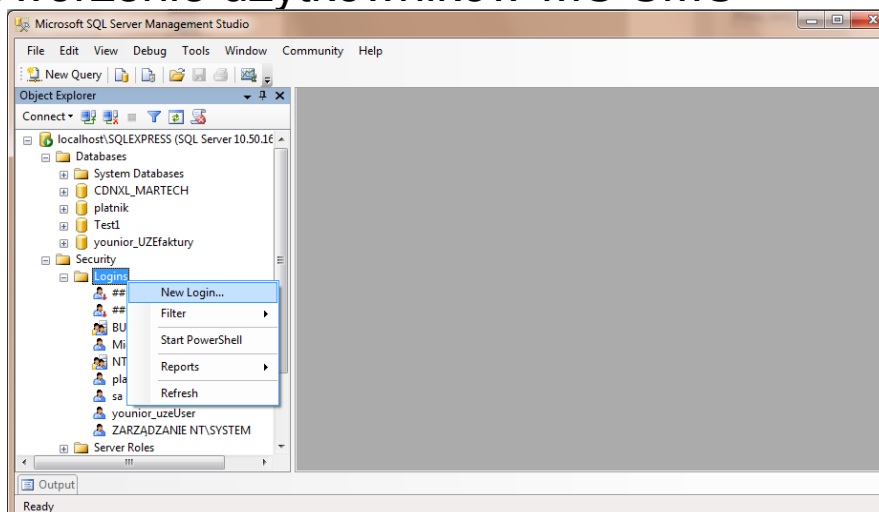
Człowiek - najlepsza inwestycja

Slajd 2	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Przypomnienie</h2> <ul style="list-style-type: none"> • CREATE • DROP • ALTER • SELECT INTO <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div>	Przypomnienie materiału z poprzedniej lekcji
Slajd 3	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Data Control Language</h2> <p>DCL (Data Control Language) ma zastosowanie do nadawania uprawnień do obiektów bazodanowych. Najważniejsze polecenia w tej grupie to:</p> <ul style="list-style-type: none"> • GRANT - służące do nadawania uprawnień do pojedynczych obiektów lub globalnie konkretnemu użytkownikowi • REVOKE – służące do odbierania wskazanych uprawnień konkretnemu użytkownikowi (np. REVOKE ALL PRIVILEGES ON EMPLOYEE FROM PIOTR - odebranie użytkownikowi wszystkich praw do tabeli EMPLOYEE). • DENY. <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div>	Definicja DCL
Slajd 4	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Użytkownicy i loginy</h2> <ul style="list-style-type: none"> • Login – jest używany do połączenia z SZBD • Użytkownik dotyczy konkretnej bazy danych • Loginy i użytkowników łączy się ze sobą za pomocą tak zwanego mapowania <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div>	Definicja loginu i użytkownika. Login to obiekt funkcjonujący w kontekście SZDB, a użytkownik w kontekście bazy danych. Na kolejnych slajdach przedstawiono tworzenie loginu i użytkownika do niego za pomocą MS SQL Management Studio oraz za pomocą TSQL

Człowiek - najlepsza inwestycja



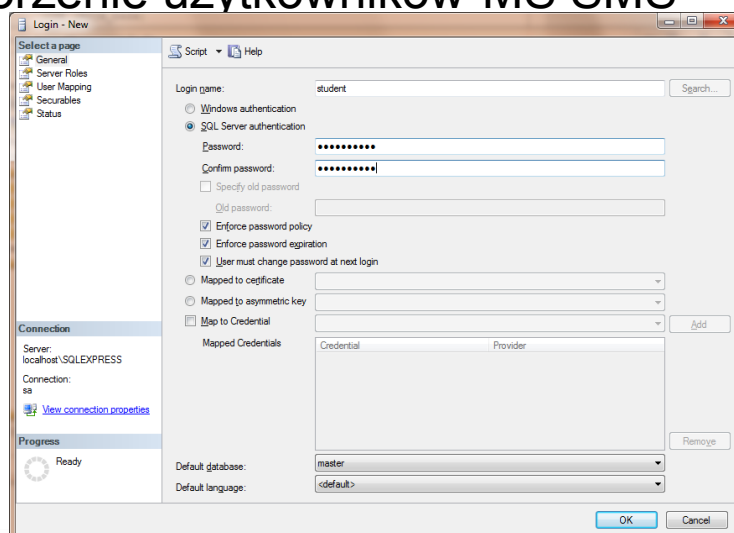
Tworzenie użytkowników MS SMS



Sekcja z loginami znajduje się w drzewie SZBD w sekcji Security -> Logins



Tworzenie użytkowników MS SMS




Po wybraniu New login pojawia nam się okno. W zakładce General wprowadzamy nazwę,


wyberamy tryb logowania. Możliwa jest autentykacja Windows lub SQL. W przypadku Windows logować się z systemu windows, w którym użytkownik jest zalogowany. Ta opcja kierowana jest głównie do użytkowników domenowych. Jeśli chcemy umożliwić logowanie się użytkownikom z różnych systemów operacyjnych wybieramy opcję SQL Server authentication.

Sekcje Default Database oraz Default language są widoczne zarówno dla loginu z autoryzacją SQL jak i Windows. W Default Database określamy domyślną bazę danych dla loginu.

Slajd
7



B2E
 BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Tworzenie użytkowników MS SMS

- W przypadku wybrania autoryzacji SQL, należy wprowadzić hasło dla loginu oraz poniżej wybrać jedną lub kilka z trzech opcji:
 - Enforce password Policy
 - Enforce password expiration
 - User must change password at next logon



Lekcja 6 - DCL, uprawnienia, użytkownicy

W przypadku wybrania autoryzacji SQL, należy wprowadzić hasło dla loginu oraz poniżej wybrać jedną lub kilka z trzech opcji:

- Enforce password Policy – hasło nie może zawierać w sobie części nazwy loginu i nie może być krótsze niż 7 znaków oraz powinno zawierać cyfry, duże i małe litery oraz znaki nie alfanumeryczne. Informacje pobierane są z polityki grupowej.
- Enforce password expiration – hasło wygasa po przekroczeniu wartości określonej w polityce grupowej.
- User must change password at next logon – wymaga zmiany hasła użytkownika przy kolejnym logowaniu.




Tworzenie użytkowników MS SMS

- Na stronie Server Roles wybieramy role serwerowe, przypisane dla tego loginu:
 - bulkadmin
 - dbcreator
 - diskadmin
 - processadmin
 - securityadmin
 - serveradmin
 - setupadmin
 - sysadmin




Na stronie **Server Roles** wybieramy role serwerowe, przypisane dla tego loginu:
bulkadmin – zezwala na operację masowego wstawiania danych (BULK INSERT),
dbcreator – zezwala na tworzenie, usuwanie, modyfikację bazy danych oraz dodawanie do niej nowych członków (CREATE DATABASE),
diskadmin – zezwala na zarządzanie plikami .mdf i .ldf (ALTER),
processadmin – zezwala na kontrolę procesów (ALTER ANY CONNECTION oraz ALTER SERVER STATE),
securityadmin – zezwala na zarządzanie loginami i uprawnieniami (ALTER ANY LOGIN),
serveradmin – zezwala na konfigurację całego serwera (ALTER SERVER STATE, ALTER SETTINGS, SHUTDOWN),
setupadmin – zezwala na zarządzanie serwerami połączonymi (ALTER ANY LINKED SERVER),
sysadmin – zezwala na pełną kontrolę nad bazami danych (CONTROL SERVER with GRANT).





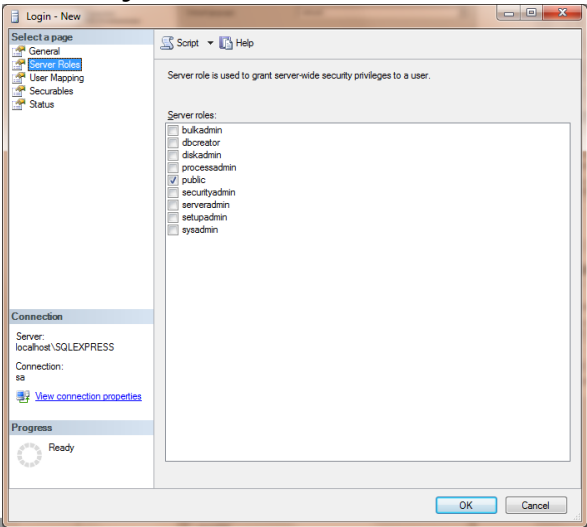
B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Tworzenie użytkowników MS SMS





Lekcja 6 - DCL, uprawnienia, użytkownicy

Na stronie **Server Roles** wybieramy role serwerowe, przypisane dla tego loginu:

- bulkadmin – zezwala na operację masowego wstawiania danych (BULK INSERT),
- dbcreator – zezwala na tworzenie, usuwanie, modyfikację bazy danych oraz dodawanie do niej nowych członków (CREATE DATABASE),
- diskadmin – zezwala na zarządzanie plikami .mdf i .ldf (ALTER),
- processadmin – zezwala na kontrolę procesów (ALTER ANY CONNECTION oraz ALTER SERVER STATE),
- securityadmin – zezwala na zarządzanie loginami i uprawnieniami (ALTER ANY LOGIN),
- serveradmin – zezwala na konfigurację całego serwera (ALTER SERVER STATE, ALTER SETTINGS, SHUTDOWN),
- setupadmin – zezwala na zarządzanie serwerami połączonymi (ALTER ANY LINKED SERVER),
- sysadmin – zezwala na pełną kontrolę nad bazami danych (CONTROL SERVER with GRANT).



Tworzenie użytkowników MS SMS

- User Mapping:
 - db_accessadmin
 - db_backupoperator
 - db_datareader
 - db_datawriter
 - db_ddladmin
 - db_denydatareader
 - db_denydatawriter
 - db_owner
 - db_securityadmin
 - public

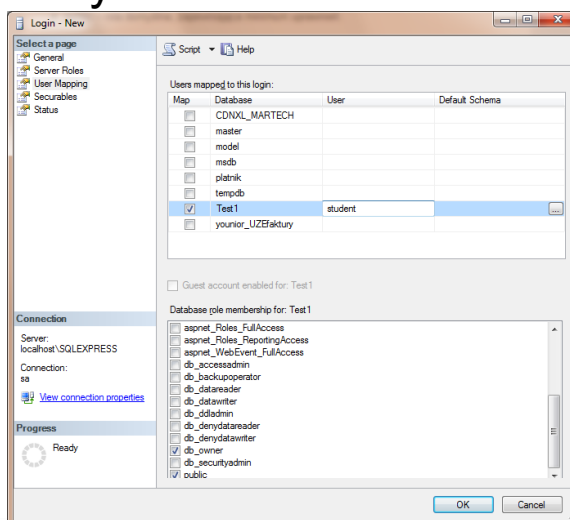


Na stronie **User Mapping** zaznaczamy, do której bazy użytkownik będzie posiadał uprawnienia (**Users mapped to this login**), a następnie w sekcji **Database role membership for** : nazwa_bazy zaznaczamy role, które chcemy nadać:

- db_accessadmin – zezwala na dodawanie i usuwanie kont,
- db_backupoperator – zezwala na wykonywanie kopii zapasowych,
- db_datareader – zezwala na odczyt baz danych,
- db_datawriter – zezwala na zapisywanie i modyfikację baz danych,
- db_ddladmin – zezwala na modyfikację i usuwanie obiektów baz danych,
- db_denydatareader – nie zezwala na odczyt baz danych,
- db_denydatawriter – nie zezwala na zapisywanie i modyfikację baz danych,
- db_owner – zezwala na pełną kontrolę nad bazą danych,
- db_securityadmin – zezwala na zarządzanie uprawnieniami oraz rolami baz danych,
- public – rola domyślna, zapewniająca minimum uprawnień.



Tworzenie użytkowników MS SMS

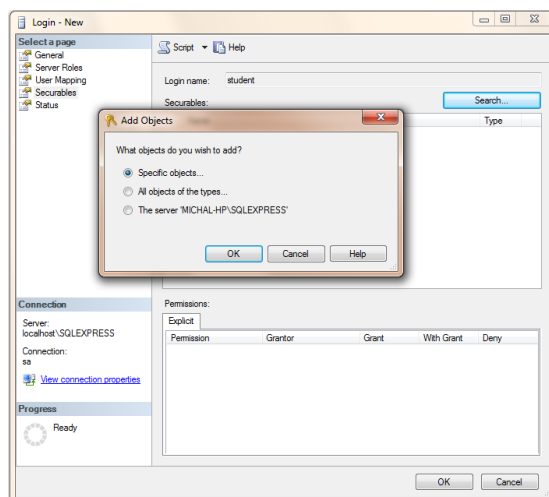


Na stronie **User Mapping** zaznaczamy, do której bazy użytkownik będzie posiadał uprawnienia (**Users mapped to this login**), a następnie w sekcji **Database role membership for** : nazwa_bazy zaznaczamy role, które chcemy nadać:

- db_accessadmin – zezwala na dodawanie i usuwanie kont,
- db_backupoperator – zezwala na wykonywanie kopii zapasowych,
- db_datareader – zezwala na odczyt baz danych,
- db_datawriter – zezwala na zapisywanie i modyfikację baz danych,
- db_ddladmin – zezwala na modyfikację i usuwanie obiektów baz danych,
- db_denydatareader – nie zezwala na odczyt baz danych,
- db_denydatawriter – nie zezwala na zapisywanie i modyfikację baz danych,
- db_owner – zezwala na pełną kontrolę nad bazą danych,
- db_securityadmin – zezwala na zarządzanie uprawnieniami oraz rolami baz danych,
- public – rola domyślna, zapewniająca minimum uprawnień.




Tworzenie użytkowników MS SMS




Strona **Securables** służy do przypisywania uprawnień do obiektów zabezpieczanych dla tego loginu.



Slajd
13



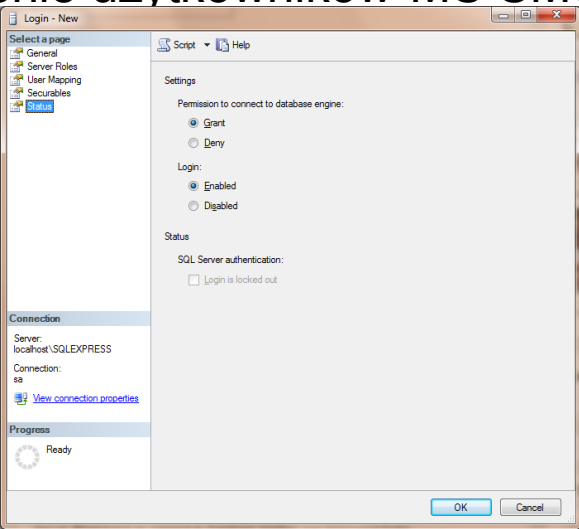
B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Tworzenie użytkowników MS SMS





Lekcja 6 - DCL, uprawnienia, użytkownicy

Strona **Status** służy do nadawania lub odejmowania uprawnień dla loginu do łączenia się z bazą danych (**Permission to connect to Database engine**) oraz blokowania konta.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





Tworzenie użytkowników TSQL

- Taki sam efekt uzyskamy za pomocą polecenia:

```
USE [master]
GO
CREATE LOGIN [student] WITH PASSWORD=N'Pa$$word' MUST_CHANGE, DEFAULT_DATABASE=[master], CHECK_EXPIRATION=ON, CHECK_POLICY=ON
GO
EXEC master..sp_addsrvrolemember @loginame = N'student', @rolename = N'sysadmin'
GO
USE [Test1]
GO
CREATE USER [student] FOR LOGIN [student]
GO
USE [Test1]
GO
EXEC sp_addrolemember N'db_datareader', N'student'
GO
USE [Test1]
GO
EXEC sp_addrolemember N'db_datawriter', N'student'
GO
USE [student]
GO
EXEC sp_addrolemember N'db_owner', N'student'
GO
```

Tworzenie nowego loginu za pomocą TSQL i nadawanie odpowiednich uprawnień.



Podstawowe zasady bezpieczeństwa

- Należy:**
 - Przyznawać użytkownikom uprawnienia tylko takie jakie potrzebują do swojej pracy
 - Monitorować przyznane uprawnienia
 - Pełne prawa do bazy powinien mieć jedynie użytkownik będący właścicielem bazy danych
- Nie należy:**
 - Przypisywać użytkownikom wszystkich uprawnień, aby rozwiązać pewien problem.
 - Pozwalać zwykłym użytkownikom na tworzenie baz danych lub obiektów w bazach

Najważniejsze zasady, o których należy pamiętać podczas przydzielania uprawnień

Slajd
16



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Prawa dostępu


- Dla każdej tabeli lub widoku (relacji) można określić prawa dostępu do danych:
 - ALL – pozwala na wszystkie operacje na danych relacji
 - SELECT – pozwala użytkownikowi na odczytywanie danych relacji
 - DELETE – pozwala na usuwanie rekordów z relacji
 - INSERT – pozwala na wstawianie nowych rekordów do tabel lub widoków
 - UPDATE – pozwala na zmianę danych w relacji
 - EXECUTE – pozwala na wykonywaniu procedur zapamiętanych




Lekcja 6 - DCL, uprawnienia, użytkownicy

Na slajdzie znajduje się zestawienie wszystkich praw dostępu, jakie można zastosować dla tabel i widoków.

Slajd
17



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Prawa dostępu

- Uwaga** – w MS SQL Server prawa SELECT i UPDATE mogą być przyznawane do wybranych kolumn.
- Domyślnie, nadanie użytkownikowi praw dostępu nie umożliwia mu przekazywanie tych praw innym użytkownikom. Dopiero dołączenie opcji WITH GRANT OPTION pozwoli mu na dalsze przekazywanie posiadanych praw.
- Prawa dostępu przechowywane są w jednej z tabel systemowych.



Lekcja 6 - DCL, uprawnienia, użytkownicy

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 18</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <hr/> <h2>GRANT</h2> <ul style="list-style-type: none"> Do przyznawania praw dostępu utworzonym wcześniej rolom bądź użytkownikom służy polecenie GRANT. Przykładowa składnia: <pre>GRANT nazwy_praw_dostępu ON nazwa_tabeli_lub_widoku TO nazwa_rol_i_lub_uzytkownika</pre> <pre>GRANT ALL ON klienci TO kierownik WITH GRANT OPTION</pre> <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div> </div>	<p>Definicja i przykład użycia polecenia GRANT</p>
<p>Slajd 19</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <hr/> <h2>REVOKE</h2> <ul style="list-style-type: none"> Polecenie REVOKE zabiera uprawnienia, które zostały wcześniej przyznane. Przykładowa składnia: <pre>REVOKE nazwy_praw_dostępu ON nazwa_tabeli_lub_widoku TO nazwa_rol_i_lub_uzytkownika</pre> <pre>REVOKE ALL ON klienci TO kierownik</pre> <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div> </div>	<p>Polecenie REVOKE – służące od odbierania praw do obiektów</p>
<p>Slajd 20</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <hr/> <h2>DENY</h2> <ul style="list-style-type: none"> Zabrania wykonywania operacji, jest silniejsze niż grant Nie można wykonywać czynności. Uprawnienie nie może zostać zmienione w wyniku członkostwa w roli. Przykładowa składnia: <pre>DENY nazwy_praw_dostępu TO nazwa_rol_i_lub_uzytkownika</pre> <pre>DENY SELECT ON klienci TO kierownik;</pre> <div>  Lekcja 6 - DCL, uprawnienia, użytkownicy </div> </div>	<p>Inaczej niż w przypadku polecenia REVOKE, polecenie DENY bezpośrednio zabiera uprawnienie polecenia. Przykładowo, jeżeli Jan jest członkiem roli bazy danych, a rola ta ma uprawnienie CREATE TABLE, Jan może również tworzyć tabele. Jednak, jeżeli Jan powinien mieć zabronione tworzenie tabel, mimo tego, że jako członek roli posiada to uprawnienie, można zabronić Janowi wykonywania tego polecenia. Dlatego, Jan nie będzie mógł uruchomić wyrażenia CREATE TABLE, pomimo tego, że normalnie rola przydzieliła mu to prawo.</p>

Człowiek - najlepsza inwestycja





KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY




Slajd
21

SQL
Structured Query Language

Przykłady

- aby przyznać użytkownikowi Jan uprawnienie do tworzenia widoku w bazie, należy uruchomić:
`GRANT CREATE VIEW TO Jan`
- aby cofnąć uprawnienie do tworzenia widoków i tabel dla użytkowników Jana i Anny należy uruchomić:
`REVOKE CREATE TABLE, CREATE VIEW FROM Anna, Jan`
- aby przyznać Joe wszystkie uprawnienia w bazie danych należy uruchomić:
`GRANT ALL TO Jan`



Lekcja 6 - DCL, uprawnienia, użytkownicy

Przegląd kilku przykładów jest najlepszym sposobem na zrozumienie działania omówionych poleceń.

Slajd
22




SQL
Structured Query Language

Ćwiczenia



Lekcja 6 - DCL, uprawnienia, użytkownicy

Ćwiczenie 1:

Utwórz nową bazę danych o nazwie Testowa_Inicjały

Ćwiczenie 2:

Utwórz użytkownika Nazwisko_studenta1 z poziomu MS SMS z domyślną bazą Testowa_Inicjały

Ćwiczenie 3:

Utwórz użytkownika Nazwisko_studenta2 z poziomu TSQL z domyślną bazą Testowa_Inicjały

Ćwiczenie 4:

Wykorzystaj procedury:

`sp_password`

`sp_defaultdb`

`sp_defaultlanguage`

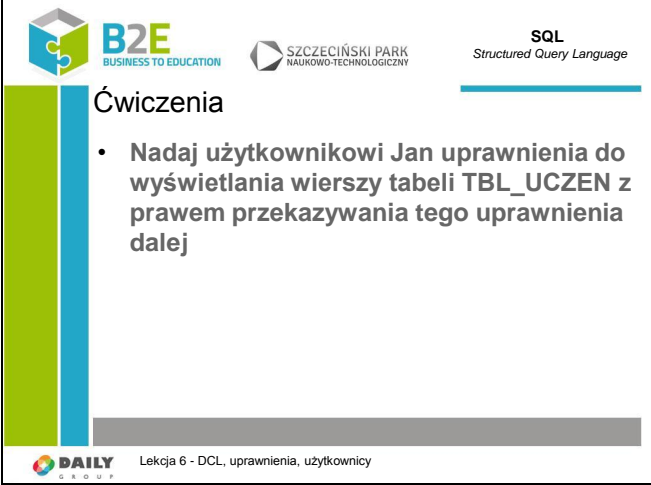
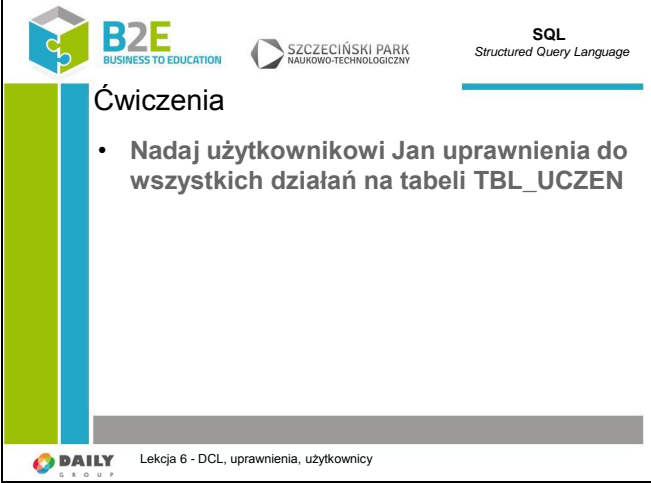
Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



	<p>sp_helplogins sp_droplogin dla użytkownika Nazwisko_studenta2</p> <p>Ćwiczenie 5: Wykorzystując procedurę <i>sp_changedbowner</i> zmień właściciela bazy danych Testowa_Inicjały na Nazwisko_studenta1</p> <p>Ćwiczenie 6: wykorzystując polecenia GRANT, REVOKE i DENY ustaw wybrane 3 uprawnienia (CREATE DATABASE, CREATE TABLE, CREATE PROCEDURE, CREATE DEFAULT, CREATE RULE, CREATE VIEW, CREATE FUNCTION, BACKUP DATABASE I BACKUP LOG) wybranym użytkownikom lub rola (w razie potrzeby utworzyć dodatkowych użytkowników lub role)</p> <p>Ćwiczenie 7: wykorzystując polecenia GRANT, REVOKE i DENY ustaw wybrane uprawnienia (SELECT, INSERT, UPDATE, DELETE, EXECUTE, REFERENCES) dla wybranych obiektów bazy Test poszczególnym użytkownikom, grupom lub rola</p> <p>Ćwiczenie 8: Zobrazuj na przykładach skutki przyznania lub zabronienia poszczególnym użytkownikom, grupom lub rola</p>	
Slajd 23	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Nadaj użytkownikowi Jan uprawnienia do wyświetlania wierszy tabeli TBL_UCZEN z prawem przekazywania tego uprawnienia dalej <p>Lekcja 6 - DCL, uprawnienia, użytkownicy</p>	GRANT SELECT ON TBL_UCZEN TO Jan WITH GRANT OPTION;
Slajd 24	 <p>Ćwiczenia</p> <ul style="list-style-type: none"> Nadaj użytkownikowi Jan uprawnienia do wszystkich działań na tabeli TBL_UCZEN <p>Lekcja 6 - DCL, uprawnienia, użytkownicy</p>	GRANT ALL ON TBL_UCZEN TO Jan;

Człowiek - najlepsza inwestycja

Slajd 25	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Odbierz użytkownikowi Jan uprawnienia do wszystkich działań na tabeli TBL_UCZEN <p> Lekcja 6 - DCL, uprawnienia, użytkownicy</p>	REVOKE ALL ON TBL_UCZEN TO Jan;
Slajd 26	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Jawnie odbierz użytkownikowi Jan uprawnienia do wszystkich działań na tabeli TBL_UCZEN, oraz wszystkim użytkownikom, którzy otrzymali od niego takie uprawnienia <p> Lekcja 6 - DCL, uprawnienia, użytkownicy</p>	DENY ALL ON TBL_UCZEN TO Jan CASCADE;
Slajd 27	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Wyświetl informacje o loginach powiązanych z użytkownikami bieżącej bazy danych oraz o kontach użytkowników i rolach istniejących w bazie. <p> Lekcja 6 - DCL, uprawnienia, użytkownicy</p>	EXEC sp_helpuser

Człowiek - najlepsza inwestycja

Slajd
28

Podsumowanie

GRANT	nadawanie uprawnień
DENY	odmawianie uprawnień
REVOKE	odbieranie uprawnień

SQL
Structured Query Language

Lekcja 6 - DCL, uprawnienia, użytkownicy

Tabela podsumowująca poznane wyrażenia

9.6.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 22-27. Uczniowie utworzą nową bazę danych, użytkowników oraz tabelę, a następnie dokonają szeregu modyfikacji uprawnień tych obiektów.

9.6.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie nauczą się świadomie przydzielać prawa do obiektów bazodanowych, poznają najlepsze praktyki, a następnie w czasie ćwiczeń praktycznie wykorzystają zdobytą wiedzę.

9.7 Lekcja 7 - Widoki i Funkcje

9.7.1 Cel lekcji

Celem lekcji jest przedstawienie dwóch typów obiektów w bazie danych: widoków oraz funkcji. Uczniowie po odbyciu lekcji powinni być w stanie samodzielnie zaprogramować proste widoki i funkcje użytkownika.

9.7.2 Treść - slajdy z opisem

Slajd
1







SQL
Structured Query Language

Lekcja 7:
widoki i funkcje

SQL Structured Query Language Lekcja 7

W tej lekcji poznamy inne niż tabele obiekty bazy danych: widoki oraz funkcje. Widoki umożliwiają „zapisanie” skomplikowanych zapytań do późniejszego – szybkiego wykorzystania. Funkcje to obiekty, które już poznaliśmy. Na poprzednich lekcjach używaliśmy funkcji systemowych MIN, MAX, SUM, AVG. SQL Serwer umożliwia definicję własnych funkcji użytkownika – w tej lekcji dowiemy się w jaki sposób.

Człowiek - najlepsza inwestycja

Slajd 2	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Przypomnienie</h2> <ul style="list-style-type: none"> • Login • Użytkownik • GRANT • REVOKE • DENY <div>  Lekcja 7 – widoki i funkcje </div>	Przypomnienie materiału z poprzedniej lekcji.
Slajd 3	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h2>Widoki</h2> <ul style="list-style-type: none"> • Widok (lub perspektywa) to tabela wirtualna określona przez zapytanie SQL-owe. • Zapytanie ma własną nazwę i jest przechowywane w bazie danych. Zmiana danych w tabeli bazowej na ogół ma wpływ na dane w widoku zbudowanym na bazie tej tabeli. • Perspektywy dają duże możliwości odnośnie przetwarzania danych. <div>  Lekcja 7 – widoki i funkcje </div>	Widoki to tabele wirtualne. Wykorzystywane są w sytuacji, gdy np. mamy tabelę posiadającą 10 kolumn, a na potrzebę aplikacji chcemy prezentować tylko część z nich.

Człowiek - najlepsza inwestycja



Widoki

- Przy tworzeniu widoków możliwe jest:
 - złączenie, dowolnie wybranych tabel,
 - wykonanie operacji selekcji, projekcji i sortowania,
 - wykonanie instrukcje grupowania,
 - użycie podzapytania.



Slajd wyjaśnia jakie operacje są możliwe przy tworzeniu widoków.



Widoki

- Standardowa składnia definicji instrukcji tworzenia widoku:

```
CREATE VIEW <nazwa widoku> [ (<nazwa kol> [, <nazwa kol> ...])
```

```
AS
```

```
(SELECT <instrukcja>
```

```
[WITH [CASCADED][LOCAL] CHECK OPTION] );
```



Definicja widoku jest podobna do wszystkich definicji obiektów w bazie danych. Create, rodzaj obiektu, nazwy kolumn, słówko AS (jako) i SELECT, który chcemy zapisać w widoku.



<p>Slajd 6</p>	<div>   </div> <div>  <div> <p>Widoki</p> <ul style="list-style-type: none"> Listy kolumn używa się gdy: <ul style="list-style-type: none"> jakiegolwiek dwie kolumny mają identyczne nazwy, kolumny zawierają wartości wyliczalne, występują połączone kolumny o różnych nazwach. </div> </div> <div> <p>SQL Structured Query Language</p> </div> <div>  <p>Lekcja 7 – widoki i funkcje</p> </div>	<p>W widokach możemy zapisać SELECT * FROM table lub SELECT col1, col2, ... FROM table. Slajd wyjaśnia kiedy unikać gwiazdki.</p>
<p>Slajd 7</p>	<div>   </div> <div>  <div> <p>Widoki</p> <ul style="list-style-type: none"> Usuwanie widoków: <ul style="list-style-type: none"> Podobnie jak w przypadku tabel, do kasowania widoków stosuje się DROP: DROP VIEW <nazwa perspektywy> </div> </div> <div> <p>SQL Structured Query Language</p> </div> <div>  <p>Lekcja 7 – widoki i funkcje</p> </div>	<p>Usuwanie widoków – tak jak wszystkich obiektów w bazie danych – dokonywane jest instrukcją DROP</p>
<p>Slajd 8</p>	<div>   </div> <div>  <div> <p>Widoki - przykład</p> <pre> CREATE VIEW oaklanders AS SELECT au_fname, au_lname, title FROM authors, titles, titleauthors WHERE authors.au_id = titleauthors.au_id AND titles.title.id = titleauthors.au_id AND city = 'Oakland'; </pre> </div> </div> <div> <p>SQL Structured Query Language</p> </div> <div>  <p>Lekcja 7 – widoki i funkcje</p> </div>	<p>Przykład prezentuje widok, który łączy dwie tabele i zwraca wartości tylko do autorów pochodzących z miasta Oakland.</p>



Widoki - przykład

```
CREATE VIEW currentinfo (PUB, TYPE, INCOME, AVG_PRICE, AVG_SALES)
AS
SELECT pub_id, type, sum(price*ytd_sales), avg(price), avg(ytd_sales)
FROM titles
GROUP BY pub_id, type;
```



Przykład prezentuje widok, który dokonuje obliczeń na kolumnach tabeli titles.












Widoki - przykład

```
CREATE VIEW cities (Author, Authorcity, Pub, Pubcity)
AS
SELECT au_lname, authors.city, pub_name, publishers.city
FROM authors, publishers
WHERE authors.city = publishers.city;
```



Przykład prezentuje widok, który łączy dwie tabele i zawęża zwracane wartości tylko do autorów pochodzących z miasta Oakland.



<p>Slajd 11</p>	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Widoki modyfikowalne</h3> <ul style="list-style-type: none"> Dane w tablicach bazowych mogą być modyfikowane (wstawiane, usuwane) z poziomu widoku tylko przy spełnieniu określonych wymagań: <ul style="list-style-type: none"> zmiany muszą być określone jednoznacznie. widok oparty jest na jednej tablicy bazowej; w definicji zapytania nie występują funkcje agregujące <div>  Lekcja 7 – widoki i funkcje </div>	<p>Dane w tablicach bazowych mogą być modyfikowane (wstawiane, usuwane) z poziomu widoku tylko przy spełnieniu określonych wymagań:</p> <ul style="list-style-type: none"> zmiany muszą być określone jednoznacznie. perspektywa oparta jest na jednej tablicy bazowej; odniesienia są tylko do kolumn tej tablicy zawiera tylko jedno zapytanie (bez UNION, EXCEPT, INTERSECT), w definicji zapytania nie występują funkcje agregujące (nie ma GROUP BY, HAVING, DISTINCT).
<p>Slajd 12</p>	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Więź CHECK w widokach</h3> <ul style="list-style-type: none"> Jeśli w zapytaniu tworzącym widok jest klauzula WHERE, więź CHECK uniemożliwi takie dodanie/ zmodyfikowanie danych do widoku, które naruszałoby tę klauzulę <pre>CREATE VIEW nazwa AS SELECT treść_zapytania.select WHERE warunek [WITH [CASCADED LOCAL] CHECK OPTION];</pre> <div>  Lekcja 7 – widoki i funkcje </div>	<p>CASCADED i LOCAL mają znaczenie jeśli widok tworzony jest na podstawie innych widoków. Jeżeli wybierzemy CASCADED, klauzule WHERE “podwidoków” też są sprawdzane, nawet jeśli nie nałożono na nie jawnie więzu (CONSTRAINT – wyjaśniony w kolejnych lekcjach) CHECK.</p>
<p>Slajd 13</p>	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Definiowanie zmiennych w SQL</h3> <ul style="list-style-type: none"> Zmienne są bardzo przydatne podczas tworzenia procedur, kursorów, oraz czasem zwykłych zapytań aby zadeklarować zmienną używamy polecenia DECLARE i poprzedzamy nazwę zmiennej "@", następnie po AS deklarujemy jej typ. <div>  Lekcja 7 – widoki i funkcje </div>	<p>Typy zmiennych mogą być dokładnie takie same jak typy tabel, które omawialiśmy na poprzednich lekcjach</p>

Slajd
14



SQL
Structured Query Language

Definiowanie zmiennych w SQL

- Przykład

```
DECLARE @dana as INT
SET @dana = 1
SELECT @dana
```



Lekcja 7 – widoki i funkcje

Definicja zmiennej @dana, przypisanie do niej wartości 1, a następnie wyświetlenie wartości za pomocą znanego nam polecenia SELECT. Zmiennej możemy użyć nie tylko w selekcje ale również w warunkach.

Slajd
15



SQL
Structured Query Language

Definiowanie zmiennych w SQL

- Przykład

```
DECLARE @dana as INT
SET @dana = 3000
Select imie, nazwisko FROM pracownicy WHERE placa_zasadnicza >= @dana
```

	imie	nazwisko
1	Jan	Kowalski
2	Jacek	Witos



Lekcja 7 – widoki i funkcje

Pamiętacie naszą tabelę pracownicy? Zdefiniowanie warunku za pomocą zmiennej @dana

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 16	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Zmienne tablicowe</h3> <ul style="list-style-type: none"> Zmienne tablicowe to tabele danych przechowywane w pamięci (zapis do nich jest bardzo szybki). Zmienia deklarujemy w ten sam sposób co zwykłą zmienną DECLARE @dana następnie TABLE i w nawiasach nazwy kolumn i ich typy) <div>  Lekcja 7 – widoki i funkcje </div>	Definicja zmiennych tablicowych
Slajd 17	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Zmienne tablicowe</h3> <ul style="list-style-type: none"> Przykład <pre> DECLARE @dana TABLE (koszty int, miesiac char(10)) INSERT INTO @dana (koszty, miesiac) SELECT Koszty, miesiac FROM @dana ---kolejny selekt--- SELECT * FROM @dana </pre> <div>  Lekcja 7 – widoki i funkcje </div>	<p>Na początku deklarujemy zmienną tablicową @dana, jej kolumny i ich typy, przypadku zmiennej skalarnej nie możemy użyć SET ale możemy umieścić dane w tabeli przy użyciu INSERT INTO wyniku z poniższego zapytania. Zwróćcie szczególną uwagę na zapis ---kolejny selekt--- jest to działanie celowe, aby zademonstrować jak wprowadza się komentarze w t-sql "--" oznacza komentarz lub /* wykomentowany tekst */.</p> <p>Ostatni SELECT powoduje wyświetlenie zawartości tabeli @dana. Uwaga po zakończeniu ostatniego select wartość zmiennej jest wymazywana.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY





Zmienne systemowe

- Zmienne systemowe oznaczone są dwoma znakami "@". Często używane zmienne systemowe:

- **@@ERROR** - numer ostatniego błędu
- **@@FETCH_STATUS** - czy kursor pobrał wiersz (0 gdy pobrał)
- **@@IDENTITY** - zawiera ostatnio wygenerowaną wartość IDENTITY (przydatne, gdy chcemy użyć ID wstawionego przez INSERT wiersza)
- **@@ROWCOUNT** - zwraca liczbę wierszy, na których operowała ostatnia instrukcja SQL
- **@@VERSION** - zwraca informację o wersji SQL Serwera



Zmienne systemowe – to zdefiniowane już w SQL serwer dostępne w każdej bazie danych.



Podział funkcji

- Funkcje systemowe, np.

- GETDATE()
- CAST()
- ROUND()
- SIN()
- HOSTNAME()
- ISNULL()

```
SELECT GETDATE(), HOST_NAME()
```







Results	Messages
(No column name)	(No column name)
2013-02-08 18:23:38.903	MICHAL-HP

- Funkcje użytkownika



Funkcje dzielimy na funkcje systemowe – dostępne w każdej instalacji MSSQL oraz funkcje użytkownika, które zostały napisane i zachowane w konkretnej bazie danych.



Slajd 20	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Funkcje systemowe - przykłady</h3> <ul style="list-style-type: none"> o CONVERT(typ_danych, wyrażenie [, styl]) - dokonuje konwersji typów danych (styl jest używany przy konwersji do daty, typów walutowych itp.) o GETDATE() - zwraca aktualną datę systemową o LEFT(napis, ile_znaków) - zwraca określoną liczbę znaków napisu rozpoczynając od lewej o LEN(napis) - zwraca długość napisu o REPLACE(napis, wzorzec, napis_do_zamiany) - wyszukuje i zamienia fragment napisu o RIGHT(napis, ile_znaków) - zwraca określoną liczbę znaków napisu rozpoczynając od prawej o SUBSTRING(napis, od, do) - zwraca określoną część napisu <div>  Lekcja 7 – widoki i funkcje </div>	Przykłady najczęściej wykorzystywanych funkcji systemowych.
Slajd 21	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h3>Tworzenie funkcji użytkownika</h3> <ul style="list-style-type: none"> • Podział funkcji użytkownika: <ul style="list-style-type: none"> • Funkcje skalarne (zwracają wartość) • Funkcje tablicowe (zwracają tabelę) <div>  Lekcja 7 – widoki i funkcje </div>	Zajmijmy się teraz tworzeniem funkcji w t-sql. Możemy tworzyć funkcje użytkownika które zwracają skalarne lub tablicowe wartości. Funkcji możemy używać w widokach, w innych funkcjach, w aplikacjach oraz procedurach.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY





Funkcje skalarne

- Składnia:


```
CREATE FUNCTION function_name
( @parameter_name parameter_data_type
  [ = default ] [ READONLY ] }
[ ,...n ]
)
RETURNS return_data_type
[ AS ]
BEGIN
    function_body
    RETURN scalar_expression
END
```




Ogólna składnia tworzenia funkcji wygląda następująco. Funkcja przyjmuje parametry (jeden lub wiele) oraz zwraca wartość (funkcja skalarna zwraca jedną wartość). Podczas definicji funkcji musimy określić ilość, typ i nazwy parametrów oraz typ zwracanej wartości. Zamiast function body – wprowadza się treść funkcji.



Slajd
 23



B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Funkcje skalarne

```

CREATE FUNCTION tylko_data (@DATE smalldatetime)
RETURNS smalldatetime
AS
BEGIN
declare @return as smalldatetime

    set @return = (convert(smalldatetime, convert(nvarchar(11), left(@DATE, 11), 120), 120) )


return @return
END

SELECT dbo.tylko_data('2012-12-12 12:12:12')
```

Results
Messages

(No column name)

2012-12-12 00:00:00


Lekcja 7 – widoki i funkcje

Powyższy przykład demonstruje w jaki sposób stworzyć funkcje tylko data, czyli datę bez godziny i minuty.

 Slajd
 24



B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Funkcje tablicowe

- Funkcje o wartościach tabelarycznych (ang. table-valued function) – są nazywane również sparаметryzowanymi widokami, zwracają jako wartość tablicę rekordów, na przykład:


Lekcja 7 – widoki i funkcje

Definicja funkcji tablicowych, funkcje te często są nazywane widokami parametryzowanymi.



Funkcje tablicowe

```
create table dane (lp int identity(1,1), nazwa nvarchar(30), wartosc int default 0)
insert into dane (nazwa) values ('a'), ('b'), ('c'), ('d'), ('e')
insert into dane (nazwa, wartosc) values ('1', 1), ('2', 1), ('3', 1), ('4', 1), ('5', 1)
go

CREATE FUNCTION dbo.FunkcjaTablicowa(@wartosc bit)
RETURNS TABLE
AS
RETURN select * from (select lp, nazwa, wartosc from dane where wartosc=@wartosc ) as subk
go

select * from dbo.FunkcjaTablicowa(1)

drop function dbo.FunkcjaTablicowa
drop table dane
```



Przykład demonstruje użycie funkcji w celu otrzymania zapytania z tabeli dane z wartością jako warunkiem i parametrem funkcji.



Ćwiczenia

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przelozony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7


Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87



	<p>Ćwiczenie 1: Utwórz widok zwracający pracowników i miejsca pracy</p> <p>Ćwiczenie 2: Utwórz widok zwracający zatrudnione kobiety. Zaktualizuj pensję zasadniczą o 20% poprzez wykorzystanie UPDATE na widoku. Sprawdź czy dane zmieniły się w tabeli.</p> <p>Ćwiczenie 3: Utwórz zmienną tabelaryczną z takimi samymi kolumnami jak klienci. Wpisz kilka przykładowych wartości do zmiennej. Dodaj wartości ze zmiennej do tabeli klienci</p> <p>Ćwiczenie 4: Utwórz funkcję zwracającą średnią płacę, na podstawie lokalizacji (miejsca pracy)</p> <p>Ćwiczenie 5: Utwórz funkcję zwracającą premię na podstawie numeru pracownika</p> <p>Ćwiczenie 6: Utwórz funkcję tabelaryczną zwracającą imię, nazwisko i pesel pracownika na podstawie miejsca zatrudnienia</p> <p>Ćwiczenie 7: Utwórz funkcję zwracającą adres (w formie tabeli) na podstawie numeru pracownika</p> <p>Ćwiczenie 8: Zmień funkcję z ćwiczenia 7, tak aby zwracała wartość skalarną</p> <p>Ćwiczenie 9: Usuń utworzone widoki i funkcje</p>	
Slajd 27		SELECT GETDATE()

Człowiek - najlepsza inwestycja

<p>Slajd 28</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Wyświetl ostatni kod błędu  <p>Lekcja 7 – widoki i funkcje</p>	<p>SELECT @@ERROR</p>
<p>Slajd 29</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Zadeklaruj zmienną liczbową ILE i przypisz jej wartość 5  <p>Lekcja 7 – widoki i funkcje</p>	<p>DECLARE @ILE int SET @ILE=5</p>
<p>Slajd 30</p>	  <p>SQL Structured Query Language</p> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Utwórz widok wyświetlający wszystkie kolumny tabeli TBL_UCZEN  <p>Lekcja 7 – widoki i funkcje</p>	<p>CREATE VIEW mojwidok AS SELECT * FROM TBL_UCZEN</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
31

B2E
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Ćwiczenia

- Utwórz funkcję zwracającą liczbę uczniów w klasie. Funkcja przyjmuje nazwę klasy w parametrze

Lekcja 7 – widoki i funkcje

```
CREATE FUNCTION  
dbo.liczbaUczniow (@klasa  
nvarchar(50))  
RETURNS int  
AS  
BEGIN  
DECLARE @ile int  
        SELECT @ile = COUNT(*)  
FROM TBL_UCZEN WHERE  
KLASA=@klasa  
RETURN @ile  
END
```

Slajd
32

B2E
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Podsumowanie

Widok	tabela wirtualna określona przez zapytanie SQL-owe
Funkcje o wartościach skalarnych	zwracają jako wynik pojedynczą wartość
Funkcje o wartościach tabelarycznych	są nazywane również sparametryzowanymi widokami, zwracają jako wartość tablicę rekordów

Celownik - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA INICJATYWA OPOWIEDZI

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Lekcja 7 – widoki i funkcje

Tabela podsumowująca poznane wyrażenia

9.7.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 26-31, zadaniem uczniów jest utworzenie widoków i funkcji na podstawie zdobytej wiedzy.

9.7.4 Opis złożonych osiągnięć ucznia

Po tej lekcji uczniowie będą potrafili samodzielnie zaprogramować widoki i funkcje w bazie danych.

9.8 Lekcja 8 - Procedury Triggery

9.8.1 Cel lekcji

Celem lekcji prezentacja dwóch kolejnych typów obiektów baz danych: procedur oraz wyzwalaczy (triggerów). Szkoleniowiec zademonstruje kilka przykładów wykorzystania procedur i triggerów.

Człowiek - najlepsza inwestycja

9.8.2 Treść - slajdy z opisem

<p>Slajd 1</p>	 <p>The slide features a green and blue vertical bar on the left. At the top, it includes logos for B2E (Business to Education), Szczeciński Park Naukowo-Technologiczny, and the DAILY logo. The main title is 'SQL Structured Query Language' in a large, bold font. Below it, the subtitle is 'Lekcja 8: procedury i triggerzy'. At the bottom, there is a small text line: 'Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego'.</p>	<p>W tej lekcji poznamy kolejne dwa typy obiektów baz danych: procedury składowane i wyzwalacze(triggerzy).</p>
<p>Slajd 2</p>	 <p>The slide has a green and blue vertical bar on the left. It includes the same logos as slide 1. The title is 'Przypomnienie'. Below it, there is a bulleted list: <ul style="list-style-type: none"> • Widok • Funkcja o wartościach skalarnych • Funkcja o wartościach tabelarycznych At the bottom, it says 'Lekcja 8 – procedury i triggerzy'.</p>	<p>Podsumowanie poprzedniej lekcji.</p>
<p>Slajd 3</p>	 <p>The slide has a green and blue vertical bar on the left. It includes the same logos as slide 1. The title is 'Procedura SQL'. Below it, there is a bulleted list: <ul style="list-style-type: none"> • Procedury składowane to prekompilowane wyrażenia języka SQL przechowywane na serwerze bazodanowym. • Mogą być definiowane z parametrami wejściowymi i wyjściowymi. • Stosuje się je do wykonywania powtarzających się logicznie takich samych operacji na (bazie) danych, nie wymagających ingerencji ze strony użytkownika. At the bottom, it says 'Lekcja 8 – procedury i triggerzy'.</p>	<p>Procedura składowana, to nic innego jak specjalna struktura w bazie danych, która przypomina klasyczne programowanie. Każda z tworzonych przez nas procedur, może zawierać parametry i zwracać określone wartości, może również wykonywać pewne operacje.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

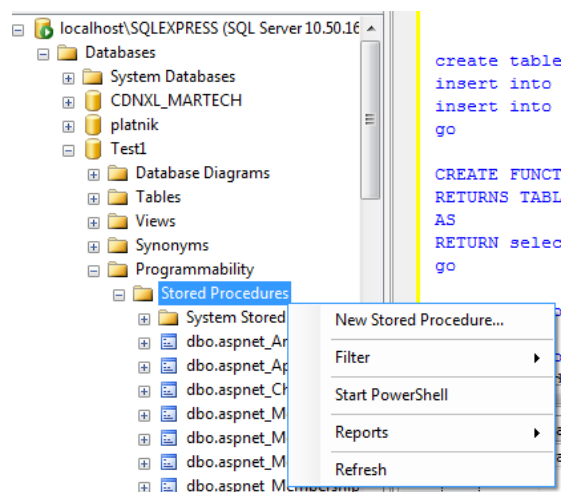
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 4	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Procedura SQL</h3> <ul style="list-style-type: none"> Składnia ogólna: <code>CREATE PROC[EDURE] nazwa_procedury lista parametrów AS instrukcje</code> Parametry podajemy zawsze ze znakiem "@", następnie typ danych i rozdzielamy parametry przecinkami. Możemy podać wartość domyślną oraz określić, że jest to parametr wyjściowy OUTPUT Definicja parametru procedury ma postać: <code>@nazwa_parametru typdanych [=wartość_domyślna] [OUTPUT]</code> ALTER PROCEDURE pozwala powtórnie wprowadzić tekst procedury. <div>  Lekcja 8 – procedury i triggerzy </div>	Procedura składowana, to nic innego jak specjalna struktura w bazie danych, która przypomina klasyczne programowanie. Każda z tworzonych przez nas procedur, może zawierać parametry i zwracać określone wartości, może również wykonywać pewne operacje.
Slajd 5	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Procedura SQL</h3> <ul style="list-style-type: none"> Zalety używania procedur składowanych: <ul style="list-style-type: none"> Różne aplikacje korzystające z tej samej bazy danych korzystają z tej samej procedury — mniejsze ryzyko popełnienia błędu. Mniejsze koszty uruchomienia i konserwacji. Z punktu widzenia wydajności <ul style="list-style-type: none"> procedura wykonywana jest przez wolniejszy język, ale na szybszym serwerze, znaczące zmniejszenie kosztu przesyłu danych. <div>  Lekcja 8 – procedury i triggerzy </div>	Zalety stosowania procedur składowanych przedstawiono na slajdzie.
Slajd 6	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Wybrane procedury systemowe</h3> <ul style="list-style-type: none"> Sp_tables - tabele bazy danych Sp_help nazwa_obiektu - informacje na temat obiektu (np. tabeli, widoku, procedury) Sp_helptext nazwa_obiektu - tekst obiektu (np. procedury) Sp_helpdb nazwa_bazy - informacje na temat bazy danych Sp_helpindex nazwa_tabeli - indeksy założone na tabeli Sp_helpconstraint nazwa_tabeli – więzy spójności na tabeli Sp_spaceused nazwa_obiektu - ilość miejsca zajętego przez obiekt <div>  Lekcja 8 – procedury i triggerzy </div>	Zalety stosowania procedur składowanych przedstawiono na slajdzie.



Tworzenie procedury w MS SMS




Zarządzanie procedurami składowanymi jest bardzo proste, bo możemy to robić przy pomocy Microsoft SQL Server Management Studio. Tam w Object Explorerze znajdują się wszystkie bazy danych, a jeśli rozwinieśmy odpowiednio drzewo, to znajdziemy również gałąź o nazwie Programmability, a w niej Stored Procedures. W tym właśnie miejscu, trzymane są wszystkie procedury przeznaczone dla konkretnej bazy danych. Idąc głębiej, możemy uzyskać dostęp do sporej kolekcji procedur systemowych. Aby utworzyć nową procedurę, wystarczy kliknąć prawym przyciskiem myszy na katalogu Stored procedures i z menu kontekstowego wybrać opcję New stored procedure.


Człowiek - najlepsza inwestycja



Slajd
8



B2E
BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Procedura SQL

```


SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE <Procedure_Name, sysname, ProcedureName>
    <@Param1, sysname, @p1> <Datatype_For_Param1, , int> = <Default_Value_For_Param1, , 0>,
    <@Param2, sysname, @p2> <Datatype_For_Param2, , int> = <Default_Value_For_Param2, , 0>
AS
BEGIN
    SET NOCOUNT ON;
    SELECT <@Param1, sysname, @p1>, <@Param2, sysname, @p2>
END
GO
```




Lekcja 8 – procedury i triggerzy

Wybranie opcji „New stored procedure” w MS SMS, spowoduje utworzenie pustego szablonu procedury, który powinien wyglądać mniej więcej tak.

Slajd
9



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY


SQL
Structured Query Language

Procedura SQL

- Najprostszy przykład

```

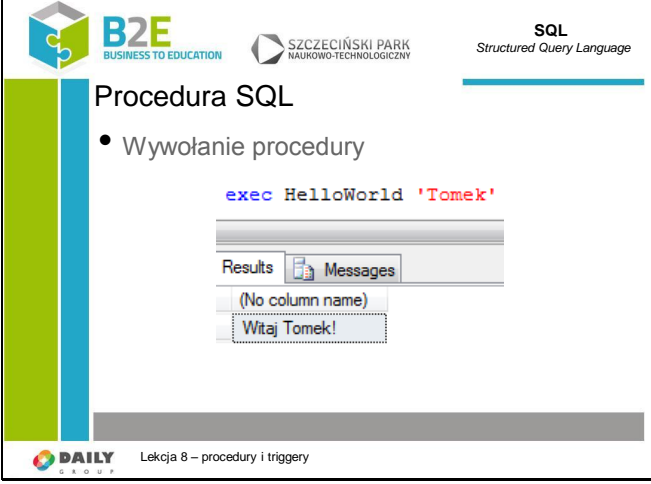
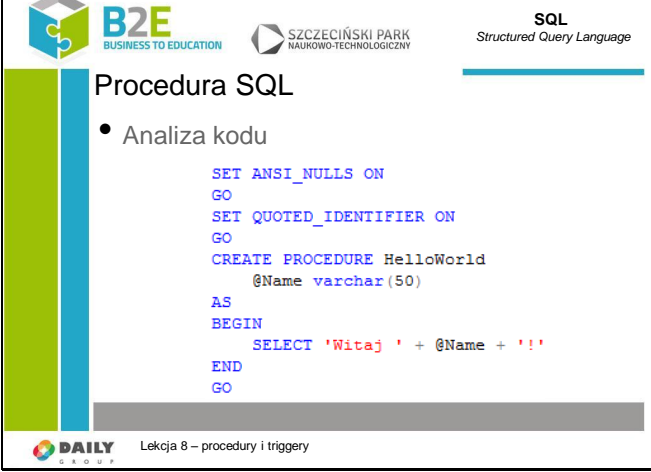
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE HelloWorld
    @Name varchar(50)
AS
BEGIN
    SELECT 'Witaj ' + @Name + '!'
END
GO
```






Lekcja 8 – procedury i triggerzy


Bardzo prosty przykład procedury składowanej. W szkielecie z poprzedniego slajdu wprowadziliśmy parametr wejściowy @Name typu varchar(50), procedura wykonuje polecenie SELECT i wyświetla Witaj + parametr + wykrzyknik.

Człowiek - najlepsza inwestycja

<p>Slajd 10</p>		<p>Wywołanie procedury z poprzedniego slajdu. Procedurę wywołujemy zawsze za pomocą polecenia EXEC (z ang. execute).</p>
<p>Slajd 11</p>		<p>Przeanalizujemy teraz poszczególne polecenia w naszej procedurze: W linii 1 oraz 3, określamy ustawienia Servera SQL. Pierwszy zapis mówi o zasadach porównywania wartości <i>NULL</i>, natomiast drugi zapis oznacza sposób używania cudzysłowów. Dopiero w linii 5, zaczynam właściwy kod procedury. Aby stworzyć procedurę, wykorzystujemy standardowe polecenie <i>CREATE</i>. Następnie, mówimy jaki obiekt chcemy utworzyć i jaką nazwę ma przyjąć. Kolejnym krokiem, będzie określenie parametrów. Choć są one opcjonalne, jednak mogą okazać się bardzo przydatne. Np. w tym przypadku określamy imię użytkownika. Każdy parametr musi zaczynać się od znaku @. Musimy podać także jego typ oraz ewentualnie długość. Parametry (jeśli jest ich więcej niż jeden), rozdzielamy przecinkiem. Po utworzeniu nagłówka procedury, w linii 7, stosujemy słowo kluczowe <i>AS</i>, które pozwala nam na wprowadzenie właściwego ciała procedury. Treść procedury, umieszczamy pomiędzy kolejnymi słowami kluczowymi - <i>BEGIN</i> oraz <i>END</i>.</p>

Człowiek - najlepsza inwestycja

	<p>Kluczowym poleceniem naszej procedury, jest zawarte w linii 10 „ SELECT 'Witaj ' + @Name + '!', która wykonuje <i>SELECT</i> z naszym imieniem. Procedura ta nie odwołuje się do żadnej tabeli z bazy danych, jedynie wyświetla powitanie na podstawie podanego parametru.</p>
Slajd 12	<div data-bbox="263 470 1396 1299">   <div data-bbox="1069 515 1340 582"> SQL <i>Structured Query Language</i> </div> <h2 data-bbox="414 616 1340 683">Procedury mogą uzyskać dostęp do tabel</h2> <pre data-bbox="414 716 1372 918"> CREATE PROCEDURE WyzerujPremie @numer_pracownika int AS BEGIN UPDATE pracownicy SET premia=0 WHERE nr_pracownika = @numer_pracownika END GO </pre> <div data-bbox="702 985 1037 1164"> <pre>exec WyzerujPremie 1 </pre> <div data-bbox="702 1064 1037 1120"> Messages </div> <p>(1 row(s) affected)</p> </div> <div data-bbox="303 1243 462 1299">  DAILY G R O U P </div> <div data-bbox="494 1243 798 1288"> Lekcja 8 – procedury i triggerzy </div> </div> <p>Procedury składowane w SQL mają dostęp do tabel w bazie danych. Na powyższym przykładzie zademonstrowano procedurę składowaną ustawiającą premię pracownika na 0. Jeśli operację zerowania premii przeprowadzamy często – warto stworzyć procedurę, gdzie podając numer pracownika kasujemy premię. Dzięki temu nie musimy każdorazowo używać odpowiedniego polecenia UPDATE.</p> <p>Poniżej przykład wywołania procedury wraz z odpowiedzią serwera – jeden wiersz został zaktualizowany.</p>

<p>Slajd 13</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <div>Podstawowe instrukcje</div> <ul style="list-style-type: none"> • RETURN w procedurze <pre>CREATE PROCEDURE Ilu_pracownikow AS BEGIN DECLARE @ile INT; SELECT @ile = COUNT(*) FROM pracownicy; RETURN @ile; END;</pre> <div>DAILY GROUP Lekcja 8 – procedury i triggerzy</div> </div>	<p>Procedura zliczy wszystkie wiersze z tabeli pracownicy i zwróci wartość poprzez RETURN. Stosuje się ją zwykle do przekazania stanu obliczeń wywołania procedury, np. czy i jaki wystąpił błąd.</p>
<p>Slajd 14</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <div>Podstawowe instrukcje</div> <ul style="list-style-type: none"> • PRINT <pre>DECLARE @Imie VARCHAR(9), @Nazwisko VARCHAR(20) SELECT @nazwisko = Nazwisko, @imie=imie FROM pracownicy WHERE nr_pracownika=1 PRINT 'Imię='+@imie+', Nazwisko='+@Nazwisko</pre> <div>Messages Imię=Jan, Nazwisko=Kowalski</div> <div>DAILY GROUP Lekcja 8 – procedury i triggerzy</div> </div>	<p>Do debuggowania (śledzenia poszczególnych kroków fragmentu kodu), najczęściej używa się instrukcji PRINT, która w konsoli umożliwia wyświetlenie (wydruk) zmiennej lub ciągu znaków.</p>
<p>Slajd 15</p>	<div>   </div> <div> <div>SQL Structured Query Language</div> <div>Podstawowe instrukcje</div> <ul style="list-style-type: none"> • Instrukcja warunkowa <p>IF warunek instrukcja lub blok [ELSE instrukcja lub blok]</p> <pre>IF @pensja > 0 BEGIN INSERT INTO pracownicy (nazwisko, placa_zasadnicza, nr_miejsc) VALUES (@nazwisko, @pensja, @miejsc); PRINT 'Wstawiono pracownika'; END ELSE PRINT 'Niepoprawna pensja';</pre> <div>DAILY GROUP Lekcja 8 – procedury i triggerzy</div> </div>	<p>Przykład warunku IF-ELSE. Jeśli pensja jest większa niż zero – wykonywany jest blok instrukcji zawarty pomiędzy BEGIN oraz END, w przeciwnym razie (ELSE) wykonywany jest PRINT.</p>

Człowiek - najlepsza inwestycja









KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 16	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Podstawowe instrukcje</h3> <ul style="list-style-type: none"> Instrukcja iteracji <p>WHILE warunek instrukcja lub blok</p> <pre> WHILE (SELECT AVG(placa_zasadnicza) FROM pracownicy) < 2000 BEGIN UPDATE pracownicy SET placa_zasadnicza = placa_zasadnicza * 1.2; IF (SELECT MAX(placa_zasadnicza) FROM pracownicy) > 4000 BREAK; ELSE CONTINUE; END; SELECT * FROM pracownicy; </pre> <div>  Lekcja 8 – procedury i triggerzy </div>	Przykład warunku iteracji - WHILE. Istnieje możliwość wyjścia z pętli używając instrukcji BREAK. Instrukcja CONTINUE powoduje, że reszta instrukcji w pętli jest ignorowana i następuje wykonanie kolejnej iteracji.
Slajd 17	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>SELECT w procedurze</h3> <ul style="list-style-type: none"> Procedura oprócz parametrów wejściowych i wyjściowych może wykonywać jedno lub kilka zapytań SELECT <pre> CREATE PROC PokazTabele AS BEGIN SELECT * FROM pracownicy; SELECT * FROM klienci; END; GO EXEC PokazTabele; </pre> <div>  Lekcja 8 – procedury i triggerzy </div>	Przykład procedury wywołującej dwa zapytania typu SELECT. Rezultatem działania tej procedury będzie wyświetlenie zawartości obu tabel: klienci i pracownicy.
Slajd 18	<div>   </div> <div> SQL <i>Structured Query Language</i> </div> <hr/> <h3>Triggerzy (wyzwalacze)</h3> <ul style="list-style-type: none"> Predefiniowane operacje na danych wyzwalane zdarzeniami. Definiowane przez użytkownika i uruchamiane automatycznie. Inne nazwy: wyzwalacze, procedury wyzwalane. <div>  Lekcja 8 – procedury i triggerzy </div>	Definicja triggerów

<p>Slajd 19</p>	<div>   </div> <div> <div>SQL</div> <div>Structured Query Language</div> </div> <h3>Triggery (wyzwalacze)</h3> <ul style="list-style-type: none"> • Triggery uruchamiane są zdarzeniami: <ul style="list-style-type: none"> ◦ INSERT ◦ UPDATE ◦ DELETE • Triggery mogą być uruchamiane: <ul style="list-style-type: none"> ◦ INSTEAD OF ◦ AFTER • Triggery mogą być uruchamiane: <ul style="list-style-type: none"> ◦ FOR EACH ROW ◦ FOR EACH STATEMENT <div>  Lekcja 8 – procedury i triggery </div>	<p>Wyzwalacze uruchamiane są zdarzeniami: INSERT, UPDATE, DELETE.</p> <p>Wyzwalacze mogą być uruchamiane: INSTEAD OF, AFTER, czyli odpowiednio zamiast lub po modyfikacji danych.</p> <p>Wyzwalacze mogą być uruchamiane: FOR EACH ROW, FOR EACH STATEMENT, czyli dla każdego wiersza, którego dotyczy działanie lub raz dla całej instrukcji.</p>
<p>Slajd 20</p>	<div>   </div> <div> <div>SQL</div> <div>Structured Query Language</div> </div> <h3>Triggery (wyzwalacze)</h3> <ul style="list-style-type: none"> • Składnia: <pre>CREATE TRIGGER nazwa_wyzwalacza ON tabela FOR [INSERT UPDATE DELETE] AS instrukcje</pre> • Po słowie FOR (równoważnie AFTER) możemy napisać jedną, dwie lub wszystkie trzy nazwy (oddzielone przecinkami). • ALTER TRIGGER pozwala powtórnie wprowadzić zmodyfikowany tekst wyzwalacza. <div>  Lekcja 8 – procedury i triggery </div>	<p>Składnia wyzwalacza zawiera nazwę triggera, tabelę, na której trigger będzie działał oraz warunek wystąpienia:</p> <p>INSERT – trigger zostanie uruchomiony w czasie wstawiania danych do tabeli</p> <p>UPDATE – trigger zostanie uruchomiony w czasie aktualizacji danych w tabeli</p> <p>DELETE – trigger zostanie uruchomiony w czasie usuwania danych z tabeli</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
21




SQL
Structured Query Language

Triggery – przykład

- Uniemożliwienie usuwania danych


```
CREATE TRIGGER wyzwl ON pracownicy
FOR DELETE AS
BEGIN PRINT 'Nie możesz usuwać danych z tej tabeli';
SELECT * FROM pracownicy;
ROLLBACK;
END;
```
- Próba usunięcia


```
DELETE FROM pracownicy;
```

Results Messages

Nie możesz usuwać danych z tej tabeli

(0 row(s) affected)



Msg 3609, Level 16, State 1, Line 1
The transaction ended in the trigger. The batch has been aborted.



Lekcja 8 – procedury i triggery

Przykład demonstrujący utworzenie triggera, który zabezpiecza nas przed omyłkowym usunięciem danych z tabeli pracownicy. Po zdefiniowaniu triggera, próbujemy usunąć dane.

Slajd
22


SQL
Structured Query Language

Usuwanie triggerów i procedur

- Usuwanie triggerów


```
DROP TRIGGER nazwa_triggera
```
- Usuwanie procedur składowanych


```
DROP PROCEDURE nazwa_procedury
DROP PROC nazwa_procedury
```



Lekcja 8 – procedury i triggery

Usuwanie obiektów typu trigger i procedura składowana. Proszę zwrócić uwagę, że SQL (T-SQL) wyrażenia PROC i PROCEDURE są równoważne.


Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





B2E
 BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Ćwiczenia

Pracownicy


nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przełożony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87



Lekcja 8 – procedury i triggerzy

Ćwiczenie 1:

Trigger uniemożliwiający usuwanie danych z tabeli miejsca.

Ćwiczenie 2:

Utwórz trigger, który w utworzy kopię danych kasowanych z tabeli klienci w tabeli klienci_kopia z identyczną strukturą jak klienci i dodatkową kolumną data usunięcia.

Ćwiczenie 3:

Utwórz trigger, który w czasie wstawiania nowych osób do tabeli pracownicy, automatycznie doda ich do tabeli klienci.

Ćwiczenie 4:

Napisz procedurę składowaną, która wylicza planowany budżet na pensje i premie na najbliższe 12 miesięcy.

Ćwiczenie 5:

Napisz procedurę składowaną, do wykorzystania przez aplikację drukującą koperty, która zwraca w tabeli imię, nazwisko i adres zatrudnienia pracownika w jednej kolumnie.

Człowiek - najlepsza inwestycja









KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 24</p>	<div>   </div> <div> SQL Structured Query Language </div> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Wywołaj procedurę OBLICZ <div>  Lekcja 8 – procedury i triggerzy </div>	<p>EXEC OBLICZ</p>
<p>Slajd 25</p>	<div>   </div> <div> SQL Structured Query Language </div> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Napisz procedurę, która zwraca liczbę z parametru podniesioną do kwadratu <div>  Lekcja 8 – procedury i triggerzy </div>	<p>CREATE PROCEDURE potega @liczba int AS BEGIN SELECT @liczba * @liczba END</p>
<p>Slajd 26</p>	<div>   </div> <div> SQL Structured Query Language </div> <hr/> <p>Ćwiczenia</p> <ul style="list-style-type: none"> Usuń procedurę z poprzedniego ćwiczenia <div>  Lekcja 8 – procedury i triggerzy </div>	<p>DROP PROCEDURE potega</p>

<p>Slajd 27</p>	<div>  B2E <small>BUSINESS TO EDUCATION</small> </div> <div>  <small>SZCZECIŃSKI PARK NAUKOWO-TECHNOLOGICZNY</small> </div> <div> SQL <small>Structured Query Language</small> </div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz trigger , który po usunięciu wiersza z tabeli TBL_UCZEN wyświetli usuniętą zawartość <div>  <small>DAILY</small> </div> <div> <small>Lekcja 8 – procedury i trigger</small> </div>	<pre>CREATE TRIGGER ttbl_uczen ON tbl_uczen AFTER DELETE AS BEGIN SELECT * FROM DELETED END</pre>
<p>Slajd 28</p>	<div>  B2E <small>BUSINESS TO EDUCATION</small> </div> <div>  <small>SZCZECIŃSKI PARK NAUKOWO-TECHNOLOGICZNY</small> </div> <div> SQL <small>Structured Query Language</small> </div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> Napisz trigger, który po dodaniu wiersza w tabeli TBL_UCZEN napisze na ekranie konsoli „Gratulacje dodałeś wiersz” <div>  <small>DAILY</small> </div> <div> <small>Lekcja 8 – procedury i trigger</small> </div>	<pre>CREATE TRIGGER ttbl_uczen1 ON tbl_uczen AFTER INSERT AS BEGIN PRINT „Gratulacje dodałeś wiersz” END</pre>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
29



SQL
Structured Query Language

Podsumowanie

Procedura składowana	Prekompilowane wyrażenia języka SQL przechowywane na serwerze bazodanowym
Wyzwalacz	Predefiniowane operacje na danych wyzwalane zdarzeniami
Warunek IF	Instrukcja warunkowa wykorzystywana w funkcjach, procedurach i triggerach
Pętla WHILE	Instrukcja iteracji wykorzystywana w funkcjach, procedurach i triggerach



Człowiek - najlepsza inwestycja



Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego



Lekcja 8 – procedury i triggerzy

Tabela podsumowująca poznane wyrażenia

9.8.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 23-28. Uczniowie wykorzystają zdobytą wiedzę przy tworzeniu procedur i triggerów.

9.8.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie będą potrafili samodzielnie implementować procedury i triggerzy w języku TSQL.

9.9 Lekcja 9 - Indeksy, constrainty

9.9.1 Cel lekcji

Celem lekcji jest poznanie struktury optymalizującej bazę danych jaką jest indeks. W drugiej części lekcji uczniowie poznają ograniczenia bazy – constraint, służące zachowaniu integralności danych.

Człowiek - najlepsza inwestycja



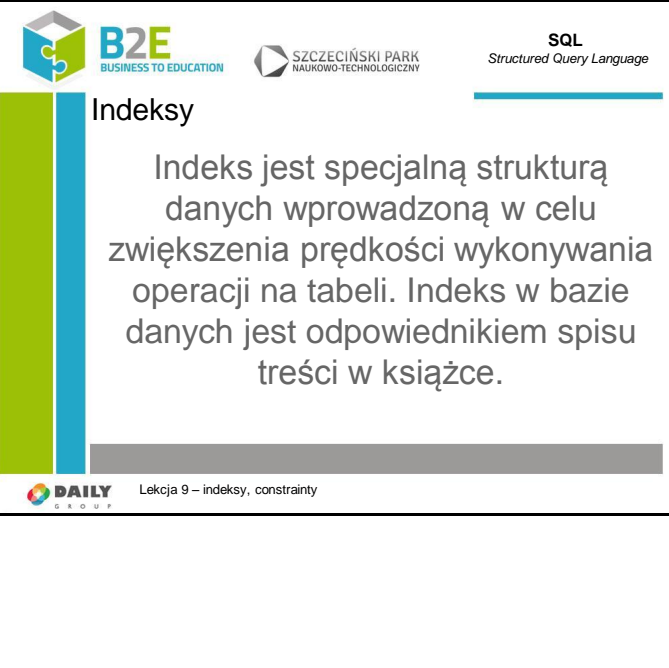


KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



9.9.2 Treść - slajdy z opisem

<p>Slajd 1</p>	 <p>The slide features a green and blue vertical bar on the left. At the top, there are logos for B2E (Business to Education), Szczeciński Park Naukowo-Technologiczny, and SQL Structured Query Language. The main title 'SQL Structured Query Language' is prominently displayed in the center. Below it, the subtitle 'Lekcja 9: indeksy, constrainty' is shown. At the bottom, there are logos for 'KAPITAŁ LUDZKI' (National Strategy for Human Capital) and 'UNIA EUROPEJSKA FUNDUSZ SPOŁECZNY' (European Union Social Fund), along with a small text line: 'Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego'.</p>	<p>W trakcie tej lekcji poznamy strukturę optymalizującą bazę danych jaką jest indeks. W drugiej części lekcji poznamy ograniczenie bazy – constraint, służące zachowaniu integralności danych.</p>
<p>Slajd 2</p>	 <p>The slide has a green and blue vertical bar on the left. It features the same top logos as slide 1. The title 'Przypomnienie' is centered. Below it, a bulleted list contains: 'Procedura składowana', 'Wyzwalacz', 'Warunek IF', and 'Pętla WHILE'. The bottom section includes the 'KAPITAŁ LUDZKI' and 'UNIA EUROPEJSKA' logos, with the text 'Lekcja 9 – indeksy, constrainty'.</p>	<p>Przypomnienie pojęć z poprzednich zajęć</p>
<p>Slajd 3</p>	 <p>The slide features a green and blue vertical bar on the left. It includes the same top logos. The title 'Indeksy' is centered. Below it, a paragraph explains: 'Indeks jest specjalną strukturą danych wprowadzoną w celu zwiększenia prędkości wykonywania operacji na tabeli. Indeks w bazie danych jest odpowiednikiem spisu treści w książce.' The bottom section contains the 'KAPITAŁ LUDZKI' and 'UNIA EUROPEJSKA' logos, with the text 'Lekcja 9 – indeksy, constrainty'.</p>	<p>Po co kartkować całą książkę dla znalezienia jednej interesującej nas informacji, jeśli możemy zajrzeć do spisu treści i na jego podstawie odnaleźć stronę, na której znajduje się to, czego szukamy. Zaoszczędzimy w ten sposób cenny czas choćby dlatego, że spis treści jest zwykle zorganizowany w sposób alfabetyczny, co znacznie upraszcza wyszukanie frazy, która nas interesuje. Indeks w bazie danych wykorzystuje się przy zapytaniach typu DQL (SELECT), które mają na celu wyszukiwanie odpowiednich wartości w bazie danych. Podczas realizowania zapytania optymalizator (Serwer</p>







Człowiek - najlepsza inwestycja



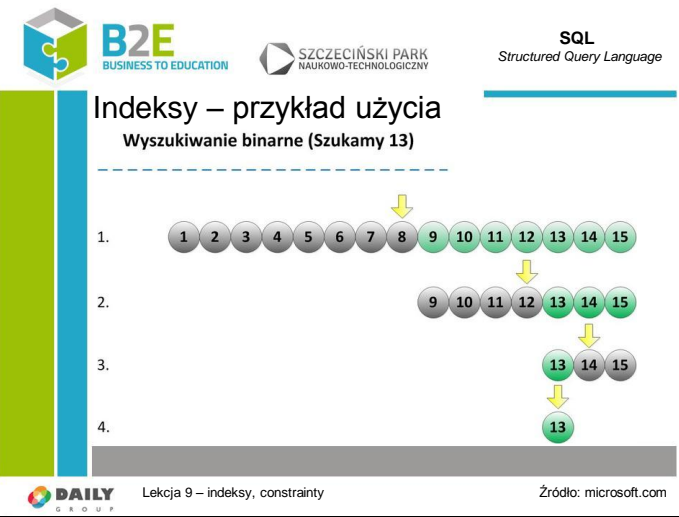
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



		<p>SQL) najpierw przeszukuje indeks, który jest uporządkowany, a następnie na podstawie indeksu odczytuje odpowiednie rekordy. Indeks posiada strukturę logiczną i fizyczną niezależną od tabeli, do jakiej się odwołuje. Posiada również własną przestrzeń dyskową oraz jest automatycznie utrzymywany przez system zarządzania bazą danych.</p>
Slajd 4	 B2E BUSINESS TO EDUCATION  SZCZECIŃSKI PARK NAUKOWO-TECHNOLOGICZNY <div> SQL <i>Structured Query Language</i> </div> <h2>Indeksy</h2> <ul style="list-style-type: none"> • Tworzenie indeksów <pre>CREATE INDEX nazwa_indeksu ON tabela (nazwa_indeksu);</pre> <ul style="list-style-type: none"> • lub <pre>ALTER TABLE nazwa_tabeli ADD INDEX (nazwa_indeksu);</pre> <div>  DAILY GROUP </div> Lekcja 9 – indeksy, constrainty	
Slajd 5	 B2E BUSINESS TO EDUCATION  SZCZECIŃSKI PARK NAUKOWO-TECHNOLOGICZNY <div> SQL <i>Structured Query Language</i> </div> <h2>Indeksy – do czego służą?</h2> <ul style="list-style-type: none"> • Zwiększają wydajność • Zapisują gdzie znajduje się dana wartość • Umożliwiają wyszukiwanie w tabelach bez potrzeby sprawdzania zawartości od początku do końca <div>  DAILY GROUP </div> Lekcja 9 – indeksy, constrainty	<p>Bez indeksów SQL serwer oferuje nam dostęp do danych – są one dostępne. Jednak przy większych zbiorach (niż nasze tabele klienci, miejsca, pracownicy), musimy zwrócić uwagę na wydajność bazy danych. W tym momencie powinniśmy zastanowić się nad indeksami, gdyż dzięki nim szybciej odnajdujemy dane w tabeli. Podczas wykonywania polecenia typu DQL (SELECT) serwer bazy danych musi wykonać bardzo dużo operacji – wybieranie danych, sortowanie wyników itd. SQL Server może pracować z bardzo dużą ilością rekordów, a często z dziesiątkami milionów rekordów w tabeli, dlatego niezwykle ważną rolę odgrywa w nim optymalizacja wszelkiego rodzaju</p>

Człowiek - najlepsza inwestycja

		<p>wyszukiwania oraz sortowania. Rekordy ułożone są w tabeli w takiej kolejności, w jakiej zostały dodane*, co oznacza, że jeśli spróbujemy wyszukać w bazie danych np. sklepu ceny danego produktu, to SQL Server musi za każdym razem na nowo przetrząsać od początku do końca wszystko, co tam się znajduje. Kiedy produktów będzie parę tysięcy, może to potrwać dosyć długo. W takiej sytuacji pomocą służą nam indeksy. Indeks nałożony na pole jest niczym innym jak kopią jego zawartości, tyle że posortowaną i odpowiednio ułożoną.</p> <p>*Taka sytuacja występuje domyślnie, w przypadku gdy PK jest indeksem klastrowanym. Można to zmienić wskazując inny indeks klastrowany niż PK.</p>
<p>Slajd 6</p>	 <p>Indeksy – przykład użycia Wyszukiwanie binarne (Szukamy 13)</p> <p>1. 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 2. 9 10 11 12 13 14 15 3. 13 14 15 4. 13</p> <p><small>DAILY Lekcja 9 – indeksy, constrainty Źródło: microsoft.com</small></p>	<p>Mamy tysiąc wartości posortowanych rosnąco, a nas interesują rekordy z zakresu od 600 do 700. Jeśli rekordy są posortowane rosnąco, SQL Server może zacząć od środkowego rekordu tego zbioru i sprawdzić, czy znaleziona wartość jest mniejsza, czy większa od podanego zakresu. Naturalnie, jeśli wartość środkowego elementu jest mniejsza od podanego zakresu, przeszukuje tylko późniejsze rekordy (również przez wyszukiwanie połówkowe w nowym zbiorze, stanowiącym połowę zbioru poprzedniego), jeśli wartość tego elementu jest większa od podanego zakresu, SQL Server szuka jedynie w rekordach wcześniejszych. Jeśli z kolei poszukiwana wartość mieści się w podanym zakresie, porusza się w obu kierunkach, dopóki nie wypadnie poza założony zakres. W sytuacji, gdy nie byłoby</p>





Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



		<p>posortowanej tablicy, musiałby najpierw albo ją posortować, albo przejrzeć wszystkie wartości w celu wyodrębnienia tych, które go interesują. Slajd przedstawia przykład binarnego wyszukiwania elementu numer 13 w zbiorze uporządkowanym 15-elementowym.</p> <p>Dzięki posortowaniu rekordów SQL Server na wstępie odrzucił całe mnóstwo rekordów, które nie pasują do naszego zapytania. Dodatkowo mamy zrealizowaną automatycznie funkcję sortowania, bez uruchamiania jakiegokolwiek funkcji sortującej.</p> <p>Źródło: microsoft.com</p>
Slajd 7	   <p>SQL Structured Query Language</p> <h2>Indeksy – kiedy stosować</h2> <ul style="list-style-type: none"> • Indeks optymalizuje zapytania = stosować wszędzie? • Nie: <ul style="list-style-type: none"> • Indeks zajmuje miejsce w bazie danych • Indeks optymalizuje odczyt, pogarsza czas zapisu • Stosujemy indeksy na polach, które najczęściej wykorzystujemy w warunku WHERE • EXPLAIN – ułatwia podejmowanie decyzji gdzie stosować indeks <p> Lekcja 9 – indeksy, constrainty</p>	<p>Niestety, nie możemy stosować indeksów wszędzie gdzie tylko możliwe. To dlatego, że ceną za zwiększenia wydajności, jakie oferuje nam indeks, jest zwiększenie rozmiarów bazy, gdyż indeks również potrzebuje trochę miejsca. Przestrzeń dyskowa jest z kolei kluczowym aspektem podczas projektowania bazy danych. A zatem już na początku została obalona postawiona teza o popieraniu rozrzutności podczas indeksowania danych. Indeksy zazwyczaj</p>

Człowiek - najlepsza inwestycja

	<p>zajmują tyle samo miejsca co dane i jest to sytuacja zupełnie naturalna. Łatwiej jest zaakceptować wydatki na przestrzeń dyskową niż akceptować czas oczekiwania na zapytanie zwiększony z sekund do kilku godzin.</p> <p>Należy się zastanowić, na których polach indeks będzie nam potrzebny, a na których jest zwyczajnie zbędny. W tym momencie pomocne będzie środowisko SQL Server, które udzieli nam diagnostycznej informacji o wykonywaniu zapytania, na podstawie której przekonamy się, czy wykorzystał on nasze indeksy. Wystarczy zapytanie SELECT poprzedzić słowem EXPLAIN, a system nam podpowie.</p> <p>O wyborze indeksu decyduje optymalizator kwerend określający, które (ewentualnie) indeksy będą najbardziej użyteczne.</p> <p>Warto również w tym punkcie określić, czym jest selektywność indeksu. Jest to nic innego, jak parametr określający, czy indeks na określonych kolumnach może być przydatny. Wartość tego parametru wyliczamy ze wzoru:</p> $S = U/W$ <p>gdzie:</p> <ul style="list-style-type: none"> • S – selektywność; • U – liczba unikalnych wartości dla kolumny; • W – liczba wszystkich wierszy w kolumnie. <p>Jeśli wartość selektywności indeksu wynosi mniej niż 85%, SQL Server raczej nie będzie z niego korzystał.</p> <p>Indeksy, z wyjątkiem grupujących, w pewnych sytuacjach mogą wydłużać czas operacji wstawiania i modyfikowania danych.</p> <p>Indeks grupujący powinien być tworzony dla kolumny, według której użytkownicy często sortują dane odczytywane z tabeli lub dla kolumn przechowujących wartości, na podstawie których zwracane są zbiory danych.</p>
--	--

Człowiek - najlepsza inwestycja

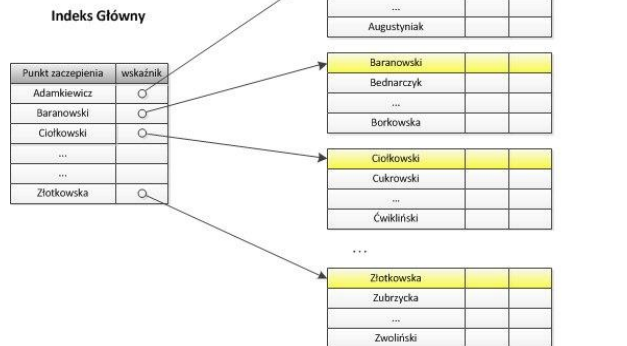


Slajd
8

Indeksy – podział

- Ze wzgl. na liczbę wskazań indeksu :

- Indeks gęsty
- Indeks rzadki, przykład:



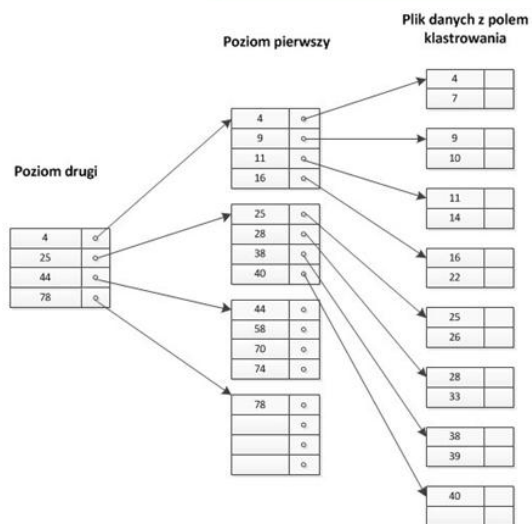
Z punktu widzenia liczby wskazań indeksu do pliku danych rozróżnia się:

- Indeks gęsty (dense) – zawiera wpis dla każdej wartości klucza wyszukiwania, czyli dla każdego rekordu.
- Indeks rzadki (sparse) – posiada wpis jedynie dla niektórych wartości wyszukiwania (np. bloków).



Indeksy – podział

- Ze względu na liczbę poziomów
 - Indeksy jednopoziomowe
 - Indeksy wielopoziomowe



Na slajdzie przykład indeksu wielopoziomowego.

Indeksy wielopoziomowe – dla pierwszego poziomu tworzymy indeks podstawowy i nazywamy go indeksem drugiego poziomu. Analogicznie dla poziomu drugiego, gdzie tworzy się indeks poziomu trzeciego.



Slajd
10

B2E
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Indeks główny

- Primary key (PK)

Indeks Główny

Punkt zaczepienia	wskaźnik
Adamkiewicz	○
Baranowski	○
Ciołkowski	○
...	
Złotkowska	○

Plik danych z kluczem głównym

Rekord kotwiczący

Adamkiewicz		
Aleksandrowicz		
...		
Augustyniak		

Baranowski		
Bednarczyk		
...		
Borkowska		

Ciołkowski		
Cukrowski		
...		
Ćwikliński		

...

Złotkowska		
Zubrzycka		
...		
Zwoliński		

Lekcja 9 – indeksy, constrainty

Indeks główny (Primary index) – zwany także podstawowym, jest założony na kluczu podstawowym pliku uporządkowanego i zawiera jeden klucz dla każdego bloku dyskowego. Pierwszy z rekordów danego bloku nazywamy rekordem zaczepienia lub rekordem kotwiczącym. Należy on do grupy indeksów rzadkich. PK jest indeksem klastrowanym (domyślnie). Wskazanie innego indeksu klastrowanego (można jeden) powoduje zmianę uporządkowania struktury pliku zgodnie ze wskazanym indeksem klastrowanym.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY





Indeks zgrupowany

- Clustered index

Plik indeksu

Wartość pola klastrowania	wskaźnik
1	○
2	○
3	○
...	
6	○

Plik danych z polem klastrowania

1		
1		
2		
2		

2		
3		
3		
3		

3		
4		
4		
4		

5		
5		
...		
6		



Indeks zgrupowany (Clustered index) – jest założony na atrybucie niebędącym kluczem podstawowym pliku uporządkowanego (nieunikatowym) porządkującym pliku uporządkowanego. Indeks zawiera jeden klucz dla każdej wartości atrybutu. Posiada dwa pola – pierwsze ma ten sam typ co pole klastrowania, drugie – wskaźnik. Indeks zawiera wpis do każdej odrębnej wartości klastrowania oraz wskaźnik na pierwszy blok, do którego ona należy. Należy do grupy indeksów rzadkich.



B2E
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Indeks niezgrupowany

- Nunclustered index

Plik indeksu

Wartość pola indeksu	wskaźnik
1	○
2	○
3	○
4	○
5	○
6	○

Plik danych z polem indeksowania

7		
32		
3		
16		

23		
15		
11		
6		

64		
2		
12		
31		

5		
18		
4		
8		

Lekcja 9 – indeksy, constrainty

Indeks niezgrupowany (Nunclustered index) – jest zakładany na pole, które ma unikatowe wartości w każdym rekordzie lub które nie jest polem klucza i posiada powtarzające się wartości. Plik indeksu niezgrupowanego jest uporządkowany i posiada dwa pola – jedno jest tego typu co wybrane pole niebędące polem uporządkowania pliku (pole indeksujące), drugie – wskaźnikiem na blok lub rekord. Dla jednego pliku może być wiele indeksów drugorzędnych. Należy do grupy indeksów zagęszczonych. Wskaźniki we wpisach indeksu są wskaźnikami na bloki.

Jest on zakładany na atrybucie indeksowym pliku danych, który nie jest atrybutem porządkującym tego pliku. Każdy rekord pliku danych posiada swój odpowiednik w rekordzie indeksu. Stąd indeks wtórny jest indeksem gęstym. Rekord indeksu wtórnego składa się z dwóch pól – wartości pola indeksowego i wskaźnika albo do rekordu albo do bloku danych zawierającego ten rekord.

Slajd
13



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Indeksy – przykłady

- Tworzenie indeksów przy tworzeniu tabeli:

```
CREATE TABLE nazwa_tabeli (  
    kolumna1 INT,  
    kolumna2 INT,  
    UNIQUE indeks_unikalny (kolumna1),  
    INDEX indeks_zwykly (kolumna2)|  
)
```



Lekcja 9 – indeksy, constrainty

Slajd
14



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Indeksy – przykłady

- Tworzenie indeksów w istniejącej tabeli:

```
ALTER TABLE nazwa_tabeli  
ADD UNIQUE indeks_unikalny (kolumna1),  
ADD INDEX indeks_zwykly (kolumna2)
```



Lekcja 9 – indeksy, constrainty

Slajd
15



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Indeksy – przykłady

- Możemy też utworzyć sam indeks za pomocą CREATE INDEX

```
CREATE UNIQUE INDEX indeks_unikalny ON nazwa_tabeli (kolumna1)  
CREATE INDEX indeks_zwykly ON nazwa_tabeli (kolumna2)
```



Lekcja 9 – indeksy, constrainty

Człowiek - najlepsza inwestycja





KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
 16


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language


Indeksy – przykłady

- Indeks na kilku polach

```

CREATE INDEX indeks_imie ON osoby (imie)
CREATE INDEX indeks_nazw ON osoby (nazwisko)

CREATE INDEX indeks_imie_nazw ON osoby (imie, nazwisko)
    
```


 Lekcja 9 – indeksy, constrainty

Dwa powyższe przykłady mogą wydawać się podobne, jednak występuje pomiędzy nimi zasadnicza różnica. Rozważmy, jak zostanie zinterpretowany przez system SQL Server poniższy przykład:

```
SELECT * FROM osoby WHERE imie='Jan' AND nazwisko='Kowalski'
```

Jeśli tabela posiada dwa różne indeksy, każdy na pojedynczej kolumnie, baza danych wykona to zapytanie w następujących krokach:

- wyszuka wszystkie rekordy, gdzie występuje imie = Jan;
- wyszuka wszystkie rekordy, gdzie występuje nazwisko = Kowalski;
- obliczy część wspólną zbiorów rekordów z pierwszego i drugiego kroku, i zwróci ją jako wynik zapytania.

W drugim przypadku, tzn. kiedy indeks jest nałożony na kolumnach (imię, nazwisko), baza danych może wyszukać potrzebne dane w jednym kroku. Sprawdzi równocześnie wartości w polach „imię” i „nazwisko”.

 Slajd
 17


B2E
 BUSINESS TO EDUCATION


 SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
 Structured Query Language

Klauzula CONSTRAINT

- CONSTRAINT = więź, ograniczenie
- Ograniczenie podobne do indeksu, chociaż może również służyć do ustanowienia relacji z inną tabelą.
- Za pomocą klauzuli CONSTRAINT w instrukcjach ALTER TABLE i CREATE TABLE można tworzyć i usuwać ograniczenia.
- Istnieją dwa typy klauzul CONSTRAINT:
 - ograniczenie dla jednego pola,
 - ograniczenie dla większej liczby pól.


 Lekcja 9 – indeksy, constrainty

Człowiek - najlepsza inwestycja


KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

 UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY


<p>Slajd 18</p>	  <p>SQL Structured Query Language</p> <h2>Podstawowe CONSTRAINT</h2> <ul style="list-style-type: none"> • NOT NULL, • CHECK, • UNIQUE, • PRIMARY KEY, • FOREIGN KEY <p> Lekcja 9 – indeksy, constrainty</p>	<p>W trakcie wcześniejszych lekcji omówiliśmy już NULL/NOT NULL oraz PRIMARY KEY, na kolejnych slajdach przyjrzymy się pozostałym.</p>
<p>Slajd 19</p>	  <p>SQL Structured Query Language</p> <h2>CHECK</h2> <ul style="list-style-type: none"> • Umożliwia sprawdzenie (walidację) wartości przed zapisem do tabeli w bazie danych, np. <pre>ALTER TABLE dbo.pracownicy WITH CHECK ADD CONSTRAINT CK_placa CHECK ((placa_zasadnicza > 0));</pre> <pre>INSERT INTO dbo.pracownicy (id_pracownika, placa_zasadnicza) VALUES (10, 0);</pre> <p><small>Msg 517, Level 16, State 5, Line 1 The INSERT statement conflicted with the CHECK constraint "CK_placa". The conflict occurred in database "Text1", table "dbo.pracownicy", column "placa_zasadnicza". The statement has been terminated.</small></p> <p> Lekcja 9 – indeksy, constrainty</p>	<p>Wykonanie operacji z przykładu uniemożliwi wstawianie nowych wierszy do tabeli pracownicy, lub modyfikację istniejących zmieniających płacę zasadniczą na wartość mniejszą lub równą 0. Uwaga – w warunku można użyć także funkcję użytkownika lub funkcję systemową!</p>
<p>Slajd 20</p>	  <p>SQL Structured Query Language</p> <h2>UNIQUE</h2> <ul style="list-style-type: none"> • Umożliwia sprawdzenie (walidację) wartości przed zapisem do tabeli w bazie danych pod kątem niepowtarzalności <pre>ALTER TABLE pracownicy ADD UNIQUE (pesel)</pre> <p> Lekcja 9 – indeksy, constrainty</p>	<p>Wykonanie operacji z przykładu uniemożliwi wstawianie nowych wierszy do tabeli pracownicy, lub modyfikację istniejących zmieniających płacę zasadniczą na wartość mniejszą lub równą 0.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
21



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

FOREGIN KEY

- Klucz obcy - dodatkowa kolumna lub zbiór kolumn w danej tabeli z wartościami, stanowiącymi klucz główny w innej tabeli

dbo.miejsca

Columns

- nr_miejsca (PK, FK, int, not null)
- ulica (nvarchar(150), null)
- numer (nvarchar(50), null)
- miasto (nvarchar(150), null)
- kod (nvarchar(50), null)
- telefon (nvarchar(50), null)

dbo.pracownicy

Columns

- nr_pracownika (PK, int, not null)
- nazwisko (nvarchar(50), null)
- imie (nvarchar(50), null)
- placa_zasadnicza (decimal(18,2), null)
- premia (decimal(18,2), null)
- pesel (nvarchar(11), null)
- stanowisko (nvarchar(50), null)
- nr_miejsca (FK, int, null)
- przełożony (int, null)


Keys

- PK_pracownicy
- FK_pracownicy_miejsca


DAILY GROUP Lekcja 9 – indeksy, constrainty

Przykład tabeli pracownicy, gdzie nr_miejsca jest kluczem obcym do kolumny nr_miejsca w tabeli miejsca.

Slajd
22



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

FOREGIN KEY - tworzenie

```

CREATE TABLE Nazwa_tabeli
(
    kolumna1 TYP,
    kolumnaN TYP,
    CONSTRAINT nazwa_klucza_obcego
    FOREIGN KEY (nazwa_kolumny_z_tworzonej_tabeli)
    REFERENCES tabela_referencyjna (kolumna_tabeli_referencyjnej)
)
    
```

DAILY GROUP Lekcja 9 – indeksy, constrainty

T-SQL pozwala na oznaczenie danej kolumny, bądź zbioru kolumn, jako klucza obcego. Jedną z metod jest zdefiniowanie tzw. funkcji CONSTRAINT podczas tworzenia tabeli. Składnia przykładowego zapytania została przedstawiona na slajdzie. Jeżeli zapytanie zostanie wykonane bezbłędnie, w SQL Server Management Studio, w lokalizacji Object Explorer->Baza Danych->Nazwa Tabeli->Keys, utworzony klucz obcy powinien funkcjonować wyłącznie z podaną nazwą.

FOREIGN KEY

- Najważniejsze informacje
 - Klucz obcy stanowi kolumnę, bądź zbiór kolumn, będących kluczem głównym w innej tabeli.
 - Łączenie tabel może odbywać się poprzez zapytanie z wykorzystaniem INNER/OUTER JOIN.
 - Klucze obce tworzy się w T-SQL, dodając odpowiednie wartości w klauzuli CONSTRAINT FOREIGN KEY.

Ćwiczenia

Pracownicy

nr_pracownika	nazwisko	imie	placa_zasadnicza	premia	pesel	stanowisko	nr_miejsca	przełożony
1	Kowalski	Jan	2300,00	250,00	82091104357	Manager	1	7
2	Nowak	Karol	2700,00	100,00	80010123987	Specjalista	1	1
3	Przepiórka	Marzena	2700,00	NULL	89121203456	Specjalista	NULL	1
4	Burzych	Paweł	1900,00	500,00	78032309123	Specjalista	2	7
5	Makłowicz	Marek	2000,00	NULL	54013112345	Specjalista	2	7
6	Naramowicka	Magdalena	2100,00	200,00	77121312098	Specjalista	1	1
7	Witos	Jacek	3000,00	500,00	69100967234	Manager	3	7
8	Markowski	Maksymilian	2000,00	NULL	67121209878	Młodszy specjalista	NULL	7

Klienci

nr_klienta	nazwisko	imie	pesel
1	Mikołaj	Wiśniewski	82040112389
2	Marcin	Kaniewski	73122990123

Miejsca

nr_miejsca	ulica	numer	miasto	kod	telefon
1	Mała	13	Poznań	60-002	(61) 123 09 89
2	Wąska	4/12A	Kraków	30-012	(12) 234 23 23
3	Grunwaldzka	34	Warszawa	00-123	(22) 876 62 87

Ćwiczenie 1:

Utwórz klucze główne w trzech tabelach

Ćwiczenie 2:

Utwórz klucz obcy w tabeli pracownicy – kolumna nr_miejsca

Ćwiczenie 3:



Dodaj unikalność pola pesel










Ćwiczenie 4:

Wprowadź mechanizm zabezpieczający w tabeli pracownicy przed dodaniem wiersza z premią mniejszą niż 0

Ćwiczenie 5:


Jak zoptymalizować zapytania do tabeli klientów zakładając, że zazwyczaj wyszukiwani są po

	<p>nazwisku? Zaproponuj skrypt wprowadzający zmianę.</p> <p>Ćwiczenie 6:</p> <p>Jakie indeksy należy założyć na tabeli pracownicy, najczęstsze zapytania to wyszukiwaniu po parze kolumn imię i nazwisko oraz po numerze pesel</p> <p>Ćwiczenie 7:</p> <p>Napisz skrypt definiujący tabelę pracownicy, gdzie pola imię, nazwisko, pesel i stanowisko nie mogą być puste, nr_pracownika jest PK, nr_miejsca FK, pesel musi być unikalny, a płaca większa niż 1600zł</p> <p>Ćwiczenie 8:</p> <p>Napisz funkcję sprawdzającą czy wartość nie jest większa od płacy minimalnej (1600zł) podłącz funkcję jako CHECK CONSTRAINT do kolumny płaca zasadnicza w tabeli pracownicy</p>	
Slajd 25		<pre>ALTER TABLE TBL_UCZEN ADD PRIMARY KEY (id_ucznia)</pre>
Slajd 26		<pre>CREATE INDEX i_tbl_uczen ON TBL_UCZEN</pre>

Slajd 27	  <div>SQL Structured Query Language</div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> • Utwórz unikalność pola PESEL w tabeli TBL_UCZEN <div>  Lekcja 9 – indeksy, constrainty </div>	ALTER TABLE TBL_UCZEN ADD UNIQUE (PESEL)
Slajd 28	  <div>SQL Structured Query Language</div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> • Dodaj sprawdzanie (walidację) pola IMIE w tabeli TBL_UCZEN. Pole IMIE nie może być krótsze niż 3 znaki <div>  Lekcja 9 – indeksy, constrainty </div>	ALTER TABLE TBL_UCZEN WITH CHECK ADD CONSTRAINT CK_IMIE CHECK(LEN(IMIE)>=3)
Slajd 29	  <div>SQL Structured Query Language</div> <hr/> <h3>Ćwiczenia</h3> <ul style="list-style-type: none"> • Utwórz unikalny indeks na polu ID_UCZNIA w tabeli TBL_UCZEN <div>  Lekcja 9 – indeksy, constrainty </div>	CREATE UNIQUE INDEX indeks_unikalny ON TBL_UCZEN (ID_UCZNIA)


Człowiek - najlepsza inwestycja

Slajd
30



B2E

BUSINESS TO EDUCATION



SZCZECIŃSKI PARK


NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Podsumowanie

Indeks	Struktura w bazie danych optymalizująca wydajność
Constraint	Ograniczenie, służące zachowaniu spójności danych




KAPITAŁ LUDZKI

INWESTYCJA W NAJLEPSZĄ PRACUJĄCĄ SIŁĘ


Człowiek - najlepsza inwestycja

UNIA EUROPEJSKA

Europejski Fundusz Społeczny



Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego



DAILY

GROUP

Lekcja 9 – indeksy, constrainty

Tabela podsumowująca poznane wyrażenia

9.9.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 24-29. Zadaniem uczniów będzie praktyczna implementacja indeksów oraz constraintów.

9.9.4 Opis założonych osiągnięć ucznia






Po tej lekcji uczniowie będą mogli samodzielnie implementować tabele, z odpowiednimi ograniczeniami służącymi zachowaniu integralności bazy danych. Dodatkowo po zapoznaniu się z mechanizmem indeksów będą mogli podjąć próbę optymalizacji baz danych.

9.10 Lekcja 10 - Transakcje, zatwierdzanie zmian, obsługa błędów, wprowadzenie do projektowania bazy danych

9.10.1 Cel lekcji

Celem lekcji jest wyjaśnienie transakcyjności bazy danych, wyjaśnienie istoty obsługi błędów oraz wprowadzenie do projektowania baz danych.

9.10.2 Treść - slajdy z opisem

Slajd 1	  <div style="text-align: right;"> SQL <i>Structured Query Language</i> </div> <hr/> <h1 style="text-align: center;">SQL</h1> <h2 style="text-align: center;">Structured Query Language</h2> <p style="text-align: center;">Lekcja 10: Transakcje, zatwierdzanie zmian, obsługa błędów, wprowadzenie do projektowania bazy danych</p> <div style="text-align: center; margin-top: 20px;">  <small>Człowiek - najlepsza inwestycja</small>  </div> <p style="text-align: center; font-size: small;">Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego</p> <div style="display: flex; justify-content: space-between; align-items: center; margin-top: 10px;">  SQL Structured Query Language Lekcja 10 </div>	Ostatnia lekcja w kursie SQL dotyczyć będzie transakcyjności baz danych. Omówimy istotę obsługi błędów. Oraz zajmiemy się zagadnieniami związanymi z normalizacją baz danych – jako podstawy do projektowania baz.
------------	---	--

Człowiek - najlepsza inwestycja

Slajd
2

**B2E**
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Przypomnienie

- INDEKS
- CONSTRAINT

DAILY GROUP

Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Podsumowanie zagadnień z poprzedniej lekcji

Slajd
3

**B2E**
BUSINESS TO EDUCATION

SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Transakcje

- BEGIN TRAN - rozpoczęcie transakcji
- COMMIT TRAN - zatwierdzenie transakcji
- ROLLBACK TRAN - wycofanie transakcji

Domyślnie ustawiona jest opcja IMPLICIT_TRANSACTIONS na OFF. Przy takim ustawieniu, jeśli nie zastosujemy BEGIN TRAN, system traktuje każdą instrukcję DML jako osobną transakcję i zatwierdza ją. Aby to wyłączyć, należy użyć (wtedy BEGIN TRAN przestaje być konieczne):
SET IMPLICIT_TRANSACTIONS ON

DAILY GROUP

Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Podstawowe komendy związane z transakcjami.

Człowiek - najlepsza inwestycja





KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd
 4


SQL
Structured Query Language

Transakcje

- Przykład



```

BEGIN TRANSACTION UsuwaniePracownika
    WITH MARK N'Usuwanie pracownika';
GO
USE Test1;
GO
DELETE FROM pracownicy WHERE nr_pracownika = 13;
GO
COMMIT TRANSACTION UsuwaniePracownika;
GO
    
```


 Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

WITH MARK – oznacza transakcje odpowiednim opisem (można go później odnaleźć w logu transakcji).

 Slajd
 5

SQL
Structured Query Language


Obsługa błędów

```

create table dane (id int, produkt nvarchar(30))







BEGIN TRANSACTION;-- początek transakcji
BEGIN TRY
    INSERT INTO dane (id,produkt) SELECT 1,'masi0'
    SELECT 2/0
END TRY

BEGIN CATCH
    SELECT
        ERROR_NUMBER() AS ErrorNumber
        ,ERROR_SEVERITY() AS ErrorSeverity
        ,ERROR_STATE() AS ErrorState
        ,ERROR_PROCEDURE() AS ErrorProcedure
        ,ERROR_LINE() AS ErrorLine
        ,ERROR_MESSAGE() AS ErrorMessage;
    IF @@TRANCOUNT > 0
        ROLLBACK TRANSACTION;-- wycofywanie zmian
END CATCH;
IF @@TRANCOUNT > 0
    COMMIT TRANSACTION;-- akceptacja zmian
SELECT * FROM dane
    
```


 Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Tak jak w innych językach programowania, tak w SQLu powinniśmy zadbać o obsługę błędów – to znaczy jeśli coś pójdzie nie tak w trakcie wykonywania kodu, powinniśmy „obsłużyć” problem i odpowiednio zareagować. Na slajdzie przedstawiono przykład prawidłowej obsługi błędów w SQL.

Przykład przedstawia celowe spowodowanie błędu: dzielenie przez 0, jego obsługa polega na odczytaniu z systemu wszystkich możliwych danych i zaprezentowaniu użytkownikowi.

<p>Slajd 6</p>	  <p>SQL Structured Query Language</p> <h3>Obsługa błędów</h3> <ul style="list-style-type: none"> • Kiedy stosować? <ul style="list-style-type: none"> o W transakcjach - po wykryciu błędu wycofujemy transakcję, o W procedurach składowanych - wykrywamy błędy powstałe głównie w wyniku niepoprawnych wartości parametrów przez użytkownika, o W triggerach - podobnie jak w transakcjach, po napotkaniu błędu wycofywana jest transakcja wywołująca trigger, o W blokach kodu SQL - wszelkie rozbudowane bloki kodu wymagają wykrywania błędów  <p>Lekcja 8 – procedury i triggerzy</p>	<p>Kiedy obsługa błędów jest niezbędna.</p>
<p>Slajd 7</p>	  <p>SQL Structured Query Language</p> <h3>Obsługa błędów</h3> <ul style="list-style-type: none"> • Jak wykrywać błędy? <ul style="list-style-type: none"> o IF...ELSE o @@ERROR o TRY...CATCH  <p>Lekcja 8 – procedury i triggerzy</p>	<p>Jak można wykrywać błędy? Metod na to jest wiele. Po pierwsze można zastosować instrukcje warunkowe, np. IF...ELSE, do sprawdzania wartości zmiennych jeszcze przed wystąpieniem błędu. Druga metoda to wykorzystanie istniejących w SZBD funkcji wykrywających błędy. W systemie Microsoft SQL Server taką funkcją jest @@ERROR, która zwraca numer błędu napotkanego w ostatnio napotkanego błędu w bieżącej sesji. Aktualnie istnieją także inne - lepsze metody wykrywania i obsługi błędów. Chodzi o strukturalną obsługę wyjątków. Jako wyjątek rozumiemy błąd, który wymaga obsługi, jedną z takich metod jest TRY...CATCH (TRY - próbuj, CATCH - przechwyc i obsłuż)</p>










Człowiek - najlepsza inwestycja









KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



Slajd 8	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h2>TRY CATCH</h2> <ul style="list-style-type: none"> Składnia: <pre>BEGIN TRY { sql_statement statement_block } END TRY BEGIN CATCH [{ sql_statement statement_block }] END CATCH [;]</pre> W bloku try – kod stanowiący potencjalne ryzyko W bloku catch – obsługa ryzyka <div>  Lekcja 8 – procedury i triggerzy </div>	<p>Wyjaśnienie try – catch.</p> <p>Konstrukcja stosowana jest w większości obecnych języków programowania i jest jednym ze sposobów obsługi wyjątków. Jeśli w bloku TRY nastąpi błąd – nastąpi przejście do bloku CATCH. Jeśli nie będzie błędu – blok CATCH nigdy nie zostanie wywołany.</p>
Slajd 9	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h2>Normalizacja</h2> <p>Normalizacja bazy danych jest to proces mający na celu eliminację powtarzających się danych w relacyjnej bazie danych.</p> <div>  Lekcja 10 – Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych </div>	<p>Główna idea polega na trzymaniu danych w jednym miejscu, a w razie potrzeby linkowania do danych. Taki sposób tworzenia bazy danych zwiększa bezpieczeństwo danych i zmniejsza ryzyko powstania niespójności (w szczególności problemów anomalii).</p> <p>Istnieją sposoby ustalenia czy dany schemat bazy danych jest "znormalizowany", a jeżeli jest to jak bardzo. Jednym ze sposobów jest przyrównanie danej bazy do schematów zwanych postaciami normalnymi (ang. normal forms lub NF).</p>
Slajd 10	<div>   <div> SQL <i>Structured Query Language</i> </div> </div> <h2>1NF - Pierwsza postać normalna</h2> <p>Jej jedynym warunkiem jest aby każda składowa w każdej krotce była elementarna (nie dawała podzielić się na mniejsze wartości). Ważną cechą relacji utworzonych zgodnie z modelem relacyjnym jest to, że zawsze są znormalizowane - spełniają 1NF.</p> <div>  Lekcja 10 – Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych </div>	<p>Przykład:</p> <p>Czy pole adres jest polem elementarnym czy nie?</p> <p>Jeśli wiemy, że w czasie operowania na bazie zawsze będziemy potrzebowali całego adresu, to pole adres możemy uznać za elementarne. Jeśli jednak dopuszczamy możliwość, że będziemy potrzebowali tylko samej miejscowości, to wtedy pole adres nie jest już elementarne i nie spełnia 1NF. Należy więc rozbić pole adres, np. na pola: ulica, miejscowość, kod pocztowy (czyli na pola elementarne).</p>

Człowiek - najlepsza inwestycja

<p>Slajd 11</p>	<div>   <div>SQL Structured Query Language</div> </div> <h2>2NF - Druga postać normalna</h2> <ul style="list-style-type: none"> • Utwórz oddzielne tabele dla zestawów wartości, odnoszących się do wielu rekordów. • Ustal powiązania tabel za pomocą klucza obcego. <div>  Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych </div>	<p>Rekordy nie powinny zależeć od niczego innego tylko od klucza podstawowego tabeli (w razie potrzeby może to być klucz złożony). Rozważmy na przykład adres klienta w systemie księgowym. Obecność adresu konieczna jest w tabeli Klienci, ale również w tabelach Zamówienia, Wysyłka, Faktury, Należności i Inkaso. Zamiast przechowywać adres w postaci wpisu w każdej tabeli, przechowuje się go w jednym miejscu: albo w tabeli Klienci, albo w oddzielnej tabeli Adresy.</p>
<p>Slajd 12</p>	<div>   <div>SQL Structured Query Language</div> </div> <h2>3NF - Trzecia postać normalna</h2> <ul style="list-style-type: none"> • Wyeliminuj pola, które nie zależą od klucza. <div>  Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych </div>	<p>Wartości rekordu, które nie są częścią jego klucza, nie należą do tabeli. Zazwyczaj, jeśli zawartość grupy pól odnosi się do więcej niż jednego rekordu tabeli, należy rozważyć umieszczenie tych pól w oddzielnej tabeli.</p> <p>Na przykład w tabeli Rekrutacja pracowników może znajdować się nazwa i adres uczelni, którą ukończył kandydat. Do korespondencji seryjnej potrzebna jest jednak kompletna lista uczelni. Jeśli informacje o uczelniach przechowywane są w tabeli Kandydaci, nie ma możliwości wyświetlenia listy uczelni bez aktualnych kandydatów. Utwórz oddzielną tabelę Uczelnie i połącz ją z tabelą Kandydaci za pomocą klucza z kodem uczelni.</p> <p>WYJĄTEK: Stosowanie reguł trzeciej postaci normalnej, chociaż teoretycznie wskazane, nie zawsze jest praktyczne. Chcąc wyeliminować wszystkie możliwe wewnętrzne zależności pomiędzy polami tabeli Klienci, należałoby utworzyć oddzielne tabele dla miast, kodów pocztowych, przedstawicieli handlowych, klas</p>

Człowiek - najlepsza inwestycja




KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



klienta i innych czynników, które mogą być zdublikowane w wielu rekordach. Normalizacja oznacza teoretycznie poprawę wydajności. Jednak wiele mniejszych tabel może spowodować spadek wydajności lub brak możliwości otwarcia pliku i przekroczenie pojemności pamięci.

Slajd
13



B2E
 BUSINESS TO EDUCATION




SZCZECIŃSKI PARK
 NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Ćwiczenie normalizacja

- Wyeliminuj pola, które nie zależą od klucza. (uzyskaj 1NF)

NrStudenta	Doradca	Pok-Dor	Klasa1	Klasa2	Klasa3
1022	Nowak	412	101-07	143-01	159-02
4123	Kowalski	216	201-01	211-02	214-01



Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Spróbujmy na innej tabeli: jak powinna wyglądać ta tabela w 1NF?

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY





Ćwiczenie normalizacja

- Pierwsza postać normalna: brak powtarzających się grup

NrStudenta	Doradca	Pok-Dor	NrKlasy
1022	Nowak	412	101-07
1022	Nowak	412	143-01
1022	Nowak	412	159-02
4123	Kowalski	216	201-01
4123	Kowalski	216	211-02
4123	Kowalski	216	214-01



Rozwiązaniem problemu jest relacja jeden-do-wielu, w której nie należy strony jeden i strony wielu umieszczać w tej samej tabeli. Zamiast tego, należy utworzyć inną tabelę w pierwszej postaci normalnej, eliminując powtarzające się grupy (NrKlasy), tak jak to przedstawiono na slajdzie.



Ćwiczenie normalizacja

- Druga postać normalna: eliminowanie powtarzających się danych

NrStudenta	Doradca	Pok-Dor
1022	Nowak	412
4123	Kowalski	216

NrStudenta	NrKlasy
1022	101-07
1022	143-01
1022	159-02
4123	201-01
4123	211-02
4123	214-01

W poprzedniej tabeli dla każdego pola NrStudenta istnieje wiele wartości w polach NrKlasy. Pole NrKlasy nie jest czynnościowo zależne od pola NrStudenta (klucz podstawowy), dlatego ta relacja nie znajduje się w drugiej postaci normalnej. Drugą postać normalną przedstawiono na podstawie następujących dwóch tabel: Pierwsza postać normalna: studenci i rejestracja.



Ćwiczenie normalizacja

- Trzecia postać normalna: eliminowanie danych, które nie zależą od klucza

Studenci:		Wydział:		
NrStudenta	Doradca	Nazwa	Pokój	Wydział
1022	Nowak	Nowak	412	42
4123	Kowalski	Kowalski	216	42



W ostatnim przykładzie pole Pok-Dor (numer pokoju doradcy) jest czynnościowo zależne od atrybutu Doradca. Rozwiązaniem jest przeniesienie tego atrybutu z tabeli Studenci do tabeli Wydział, tak jak to przedstawiono na slajdzie.



Normalizacja plusy i minusy

Cel normalizacji:

- uniknięcie redundancji (tj. powtarzania się pól z identycznymi wartościami w różnych tabelach);
- wyeliminowanie niewygodnych relacji wieloznacznych;
- uniknięcie anomalii przy aktualizacji: modyfikacji, wstawianiu i usuwaniu;
- uniknięcie niespójności.

Koszt:

- Mnożenie liczby tabel – wydłużenie czasu dostępu do danych.

Poszukiwanie kompromisu (dwa współzawodniczące cele):

- Zapobieganie anomalii oraz zapewnienie rozsądnego czasu dostępu do danych.



Normalizacja ma swoje wady i zalety. W praktyce podczas projektowania baz danych najczęściej szukamy kompromisu.



Slajd
 18

SQL
Structured Query Language

Ćwiczenia

Wprowadź obsługę błędów w procedurze (TRY-CATCH i RAISERROR):

```
CREATE PROCEDURE podzieli1
(
    @dzielna float,
    @dzielnik float
)
AS
BEGIN
    SELECT @dzielna / @dzielnik
END
```



Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Np.


```
CREATE PROCEDURE podzieli1
(
    @dzielna float,
    @dzielnik float
)
AS
IF @dzielnik = 0
    RAISERROR('Dzielenie przez ZERO!',14,1)
ELSE
    SELECT @dzielna / @dzielnik
END
```

Człowiek - najlepsza inwestycja


KAPITAŁ LUDZKI
 NARODOWA STRATEGIA SPÓJNOŚCI

 UNIA EUROPEJSKA
 EUROPEJSKI
 FUNDUSZ SPOŁECZNY


Slajd
19



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Ćwiczenia


Przeprowadź normalizację tabeli:

nazwa	imie prowadzacego	nazwisko prowadzacego	adres prowadzacego
język angielski dla dzieci	Katarzyna	Nowak	ul. Zgierska 19 m. 6 Łódź
judo	Tomasz	Woźniak	ul. Motylowa 5 Łódź
rytmika	Zofia	Leśniak	ul. Złota 14/17 Zgierz
język francuski	Katarzyna	Nowak	ul. Zgierska 19 m. 6 Łódź
karate	Janusz	Kot	Smardzew 8
gimnastyka korekcyjna	Tomasz	Woźniak	ul. Motylowa 5 Łódź
...			

DAILY GROUP

Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Slajd
20



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Ćwiczenia

Wprowadź obsługę błędów w poniższej procedurze (w przypadku błędu zwróć -1:


```
CREATE PROCEDURE dbo.p_dziel_przez_0
AS
BEGIN
DECLARE @wynik int;
SELECT @wynik = 1/0;
RETURN @wynik;
END
GO
```

DAILY GROUP

Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

```
CREATE PROCEDURE
dbo.p_dziel_przez_0
AS
BEGIN
BEGIN TRY
DECLARE @wynik int;
SELECT @wynik = 1/0;
RETURN @wynik;
END TRY
BEGIN CATCH
RETURN -1;
END CATCH
END
GO
```

Slajd
21



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL

Structured Query Language

Ćwiczenia

Zmodyfikuj procedurę z poprzedniego ćwiczenia, tak aby obok -1 zwróciła kod błędu

DAILY GROUP

Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

```
CREATE PROCEDURE
dbo.p_dziel_przez_0
AS
BEGIN
BEGIN TRY
DECLARE @wynik int;
SELECT @wynik = 1/0;
RETURN @wynik;
END TRY
BEGIN CATCH
SELECT -1, ERROR_NUMBER() AS
ErrorNumber
RETURN -1;
END CATCH
END
GO
```

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



<p>Slajd 22</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <p>Wyświetl ostatni kod błędu</p> <p><small>DAILY GROUP Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych</small></p>	<p>SELECT @@ERROR</p>
<p>Slajd 23</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <p>Obuduj poniższe zapytanie rozpoczęciem i zakończeniem transakcji:</p> <p>SELECT Imie, Nazwisko, Klasa FROM TBL_UCZEN</p> <p><small>DAILY GROUP Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych</small></p>	<p>BEGIN TRANSACTION MojaTransakcja GO SELECT Imie, Nazwisko, Klasa FROM TBL_UCZEN GO COMMIT TRANSACTION MojaTransakcja GO</p>
<p>Slajd 24</p>	  <p>SQL Structured Query Language</p> <h3>Ćwiczenia</h3> <p>Wymień i scharakteryzuj postaci normalne</p> <p><small>DAILY GROUP Lekcja 10 –Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych</small></p>	

Człowiek - najlepsza inwestycja



Podsumowanie

TRANSACTION	Zestaw operacji w SQL, zamin następuje faktyczne zatwierdzenie
COMMIT	Służy do zatwierdzania transakcji
ROLLBACK	Służy co odrzucenia transakcji
TRY ... CATCH ...	Zabezpieczenie kodu
Normalizacja	Eliminacja powtarzających się danych w relacyjnej bazie danych
1NF	Pierwsza postać normalna
2NF	Druga postać normalna
3NF	Trzecia postać normalna

Człowiek - najlepsza inwestycja



Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego



Lekcja 10 – Transakcje, obsługa błędów, wprowadzenie do projektowania bazy danych

Tabela podsumowująca poznane wyrażenia

9.10.3 Ćwiczenia

Ćwiczenie zostało przedstawione na slajdach 18 - 24, będzie polegało na wprowadzeniu obsługi błędów w procedurze składowanej oraz na normalizacji przykładowej tabeli.

9.10.4 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie będą mogli podjąć próbę zaprojektowania bazy danych. Bloki kodu przez nich tworzone będą zawierały obsługę błędów, będą stosować transakcyjne podejście w programowaniu baz danych.

9.11 Lekcja 11 – Interdyscyplinarny projekt książki teleadresowej

9.11.1 Cel lekcji

Celem lekcji jest zaprojektowanie bazy danych do użycia w na stronie WWW oraz w aplikacji mobilnej z funkcjonalnością książki teleadresowej kolegów i koleżanek ze szkoły lub klasy. W tej lekcji uczniowie dowiedzą się jak praktycznie wykorzystać wiedzę zdobytą w tym module na potrzeby pierwszego projektu informatycznego składającego się z elementów wykorzystujących cztery języki programowania: SQL, PHP, JavaScript i JAVA.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



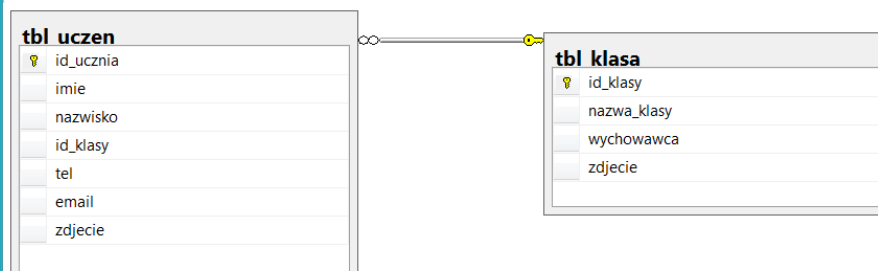
9.11.2 Treść - slajdy z opisem

<p>Slajd 1</p>	 <p>SQL <i>Structured Query Language</i></p> <p>SQL Structured Query Language</p> <p>Lekcja 11: Interdyscyplinarny projekt: Książka teleadresowa kolegów/koleżanek ze szkoły</p> <p><small>Kapitał Ludzki - najlepsza inwestycja</small></p> <p><small>Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego</small></p> <p>DAILY GROUP SQL Structured Query Language Lekcja 11</p>	<p>Proces powstawania książki teleadresowej kolegów/koleżanek ze szkoły. W module SQL pokażemy jak zaprojektować bazę danych, na PHP i JavaScript, jak stworzyć stronę WWW do dodawania kolegów i koleżanek z klasy do takiej książki, a w module JAVA nauczymy się, jak zaprogramować prostą aplikację na telefony z systemem operacyjnym Android do przeszukiwania danych o uczniach z różnych klas.</p>
<p>Slajd 2</p>	 <p>SQL <i>Structured Query Language</i></p> <p>Założenia</p> <ul style="list-style-type: none"> • Lista klas w szkole • Lista uczniów w klasie • Szczegółowe informacje na temat ucznia: <ul style="list-style-type: none"> • Imię • Nazwisko • Adres e-mail • Numer telefonu • Zdjęcie <p>DAILY GROUP Lekcja 11 – Interdyscyplinarny projekt</p>	<p>Założenia naszej aplikacji – chcemy aby aplikacja umożliwiała prezentację listy klas w naszej szkole. Tak abyśmy mogli swobodnie przeglądać uczniów w każdej z klas. Po wybraniu szukanego ucznia chcielibyśmy zaprezentować użytkownikom naszej aplikacji (mobilnej lub webowej) podstawowe informacje dotyczące ucznia: imię, nazwisko, adres e-mail, numer telefonu oraz zdjęcie.</p>
<p>Slajd 3</p>	 <p>SQL <i>Structured Query Language</i></p> <p>Propozycja bazy danych</p> <ul style="list-style-type: none"> • Tabela tbl_uczen • Tabela tbl_klasa <p>DAILY GROUP Lekcja 11 – Interdyscyplinarny projekt</p>	<p>W tym celu potrzebne nam będą dwie tabele, które zgodnie z najlepszymi praktykami oznaczmy prefixem „tbl_”: tbl_uczen oraz tbl_klasa.</p>

Człowiek - najlepsza inwestycja



Propozycja bazy danych



Tworzymy relacyjną bazę danych. Nawet przy dwóch tabelach taką relację należy utworzyć. Kluczem głównym tabeli `tbl_uczen` będzie `id_ucznia`, kluczem głównym tabeli `tbl_klasa` – `id_klasy`. Kluczem obcym tabeli `tbl_uczen` będzie `id_klasy`.



Slajd
5



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Bazy danych - skrypt

```

CREATE DATABASE SPNT

USE [SPNT]

CREATE TABLE [dbo].[tbl_klasa](
    [id_klasa] [int] IDENTITY(1,1) NOT NULL,
    [nazwa_klasa] [nvarchar](50) NOT NULL,
    [wychowawca] [nvarchar](150) NOT NULL,
    [zdjecie] [nchar](250) NULL,
    CONSTRAINT [PK_tbl_klasa] PRIMARY KEY CLUSTERED
    (
        [id_klasa] ASC
    )
    ON [PRIMARY]

CREATE TABLE [dbo].[tbl_uczen](
    [id_ucznia] [int] IDENTITY(1,1) NOT NULL,
    [imie] [nvarchar](150) NOT NULL,
    [nazwisko] [nvarchar](150) NOT NULL,
    [id_klasa] [int] NOT NULL,
    [tel] [nchar](10) NULL,
    [email] [nvarchar](250) NULL,
    [zdjecie] [nvarchar](250) NULL,
    CONSTRAINT [PK_tbl_uczen] PRIMARY KEY CLUSTERED
    (
        [id_ucznia] ASC
    )
    ON [PRIMARY]

ALTER TABLE [dbo].[tbl_uczen] WITH CHECK ADD CONSTRAINT [FK_tbl_uczen_tbl_klasa] FOREIGN KEY([id_klasa])
REFERENCES [dbo].[tbl_klasa] ([id_klasa])
GO

ALTER TABLE [dbo].[tbl_uczen] CHECK CONSTRAINT [FK_tbl_uczen_tbl_klasa]
    
```

Przygotujmy skrypt tworzący bazę danych oraz obie tabele, wraz z kluczami i relacją.

Slajd
6



B2E
BUSINESS TO EDUCATION



SZCZECIŃSKI PARK
NAUKOWO-TECHNOLOGICZNY

SQL
Structured Query Language

Widok

- Widok prezentujący uczniów wszystkich klas

```

CREATE VIEW [dbo].[v_uczniowie]
AS
SELECT dbo.tbl_uczen.id_ucznia, dbo.tbl_uczen.imie, dbo.tbl_uczen.nazwisko, dbo.tbl_klasa.nazwa_klasa,
    dbo.tbl_klasa.wychowawca, dbo.tbl_uczen.tel, dbo.tbl_uczen.email, dbo.tbl_uczen.zdjecie
FROM    dbo.tbl_klasa INNER JOIN dbo.tbl_uczen ON dbo.tbl_klasa.id_klasa = dbo.tbl_uczen.id_klasa
    
```

Lekcja 11 – Interdyscyplinarny projekt

Na potrzeby czytelnej dla użytkownika prezentacji listy uczniów z przyporządkowaniem do klasy – utwórzmy właściwy widok.

<p>Slajd 7</p>	<div>   </div> <div> <div>SQL</div> <div>Structured Query Language</div> </div> <h3>Procedura</h3> <ul style="list-style-type: none"> Procedura zapisująca nowego ucznia <pre> CREATE PROCEDURE dodaj_ucznia @imie nvarchar(150), @nazwisko nvarchar(150), @tel nchar(10), @email nvarchar(150), @id_klasy int AS BEGIN INSERT INTO [dbo].[tbl_uczen] (@imie, @nazwisko, @tel, @email, @id_klasy) VALUES (@imie, @nazwisko, @tel, @email, @id_klasy) END GO </pre> <div>  Lekcja 11 – Interdyscyplinarny projekt </div>	<p>Dobrze jest nie wykonywać operacji zapisu z kodu PHP przez bezpośrednie napisanie kwerendy INSERT – utwórzmy więc procedurę do zapisu nowego ucznia.</p>
<p>Slajd 8</p>	<div>   </div> <div> <div>SQL</div> <div>Structured Query Language</div> </div> <h3>Funkcja</h3> <ul style="list-style-type: none"> Funkcja zwracająca ilość uczniów w klasie <pre> CREATE FUNCTION dbo.f_ile_w_klasie (@id_klasy int) RETURNS int AS BEGIN DECLARE @ile int SELECT @ile = COUNT(*) FROM tbl_uczen WHERE id_klasy = @id_klasy RETURN @ile END GO </pre> <div>  Lekcja 11 – Interdyscyplinarny projekt </div>	<p>Być może jako programistom przyda nam się funkcja zwracająca ilość uczniów w klasie. Utwórzmy taką funkcję.</p>
<p>Slajd 9</p>	<div>   </div> <div> <div>SQL</div> <div>Structured Query Language</div> </div> <h3>Podsumowanie</h3> <ul style="list-style-type: none"> Gotowa prosta baza danych do gromadzenia informacji na temat uczniów i klas Widok wyświetlający w przyjemny dla użytkownika sposób informacje o uczniach Procedura do zapisu uczniów Funkcja do obliczania ilości uczniów <div>   </div> <div>  Lekcja 11 – Interdyscyplinarny projekt </div>	<p>Podsumowanie utworzonych obiektów bazodanowych.</p>

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY



9.11.3 Opis założonych osiągnięć ucznia

Po tej lekcji uczniowie przygotowują dla siebie projekt bazy danych do wykorzystania w ich pierwszym projekcie informatycznym – książki teleadresowej kolegów i koleżanek z klasy i ze szkoły dostępnej poprzez WWW oraz aplikację mobilną.

Człowiek - najlepsza inwestycja



KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

