Bazy Danych

2. DDL

Opracował: Maciej Penar

Spis treści

[1. Zanim zaczniemy 3](#_Toc508211720)

[Drzewo operatorów algebry relacji 3](#_Toc508211721)

[Ściąga sql 5](#_Toc508211722)

[2. (5 pkt) Algebra relacji – część bardziej ćwiczeniowa 8](#_Toc508211723)

[3. (7 pkt) SQL 8](#_Toc508211724)

[4. Kartkówka 9](#_Toc508211725)

# 1. Zanim zaczniemy

Zrelaksować się i przyswoić sobie teorię dot. SQL – w szczególności grup wyrażeń Data Manipulation Language (DML), Data Definition Language (DDL) oraz Data Control Language (DCL). Na chwilę odstawimy wyrażenia SELECT.

Materiały:

* SQL: <https://pl.wikipedia.org/wiki/SQL>
* Podstawowy kurs systemów baz danych, rozdział … o SQL’ach (nie mam książki przy sobie), J. Ullman, J. Widom

Oprogramowanie:

* ORACLE Database c12
  + SQL Developer

Fragmenty dokumentacji Oracle 12c:

* CREATE TABLE: [link](https://docs.oracle.com/database/121/SQLRF/statements_7002.htm)
* Indeksy: [link](https://docs.oracle.com/cd/B28359_01/server.111/b28310/indexes003.htm#ADMIN11722)
* Dziedziczenie: [link](https://docs.oracle.com/en/database/oracle/oracle-database/12.2/adobj/inheritance-in-sql-object-types.html#GUID-D6D92FB6-7BC4-4EE6-A9EC-BC69C5BA5A56)
* Typy danych: [link](https://docs.oracle.com/database/121/SQLRF/sql_elements001.htm)

# 2. Omówienie DDL

## Wstęp

Pozbiór SQL’a który nazywamy Data Definition Language (DDL) służy do modelowania bazy danych. Wyrażenia DDL na ogół nie zwracają danych, dlatego nazywa się je **poleceniami** (Commands), a nie **zapytaniami** (Queries).

Do najczęściej spotykanych wyrażeń DDL zaliczamy:

* **CREATE structure** – do tworzenia struktur w których przechowywane są dane np. CREATE TABLE
* **ALTER structure** – do zmiany istniejącej struktury np. ALTER TABLE
* **DROP structure** – do usunięcie istniejącej struktury np. DROP TABLE

Dobrą praktyką w środowiskach produkcyjnych jest wykonanie poleceń utworzenia struktur (CREATE) jednokrotnie. Ewentualne zmiany przeprowadzane są z użyciem wyrażeń ALTER.

## skrót wyrażeń

* Dotyczące tabel:
  + CREATE TABLE
  + ALTER TABLE
  + DROP TABLE
* Dotyczące indeksów:
  + CREATE INDEX
  + DROP INDEX
* Dotyczące typów:
  + CREATE TYPE
  + DROP TYPE
* Dotyczące sekwencji:
  + CREATE SEQUENCE
  + DROP SEQUENCE
* Dotyczące widoków
  + CREATE VIEW
  + ALTER VIEW
  + DROP VIEW
  + CREATE MATERIALIZED VIEW
* Ograniczenia:
  + ADD CONSTRAINT
  + DROP CONSTRAINT
* Typy ograniczeń:
  + CHECK
  + FOREIGN KEY / REFERENCES
  + UNIQUE
  + PRIMARY KEY
  + NULL / NOT NULL
  + DEFAULT

# 3. (3 pkt) Bonusowe punkty

Na któryś zajęciach laboratoryjnych chciałbym pokazać dlaczego MS Access jest **bardzo** źle zaprojektowanym narzędziem. Dodatkowo na 3-ciej liście wracamy z SQL’em, ale do tego potrzebujemy porządnej Bazy Danych – można taką postawić podążając instrukcją <https://github.com/mpenarprz/BazyDanychI4/blob/master/Laboratorium/docx/1L.Oracle%20%26%20HammerDB.docx>

Z tego też względu przyznaję bonusowe punkty za:

* (1 pkt) za pokazanie że macie zainstalowanego lokalnie Access 2016
* (2 pkt) za pokazanie że macie zainstalowanego Oracle’a 12c i posadzoną bazę TPC-H

# 3. (12 pkt) Data Definition Language

Wybrać i opisać dowolną, sensowną mini-rzeczywistość – dająca się opisać w minimum 3 encjach. Określić jakie zapytania będą najczęściej wykonywane. W oparciu opisaną mini-rzeczywistość przygotować skrypty **Up.sql, Down.sql** oraz **Upgrade.sql**.

## (8 pkt) Up.sql

W tym skrypcie powinny znaleźć się wyrażenia **CREATE** które tworzą struktury oraz związki.

1. Napisać krótki opis mikro-świata
2. Utworzyć sekwencje przez **CREATE SEQUENCE**
3. Napisać wyrażenia **CREATE TABLE / CREATE TYPE**. Upewnić się że w każdej tabeli:
   1. Istnieje klucz główny, uzupełniany wartościami z sekwencji
   2. Dobrać odpowiednie typy danych
   3. Dobrane są odpowiednie ograniczenia (CHECK / NULL/ UNIQUE/DEFAULT)
   4. Zamodelowane są poprawne związki
   5. Nie ma wiszącej tabeli (takiej które nie jest wskazywana kluczem obcym z innej tabeli oraz nie posiada klucza obcego do innej tabeli)
4. Utworzyć widok eksponujący co najmniej dwie tabele
5. Określić jakie zapytania będą najczęściej wykonywane. Napisać polecenia **CREATE INDEX** pokrywające pola w zaproponowanych zapytaniach.
6. *\* Skrypt powinien być idempotentny*

## (2 pkt) down.sql

W tym skrypcie powinny znaleźć się wyrażenia **DROP** które zniszczą wszystkie obiekty.

1. Napisać polecenie **DROP** do wszystkich obiektów ze skryptu **Up.sql**
2. *\* Skrypt powinien być idempotentny*

## (2 pkt) UPDATE.sql

W tym skrypcie powinny znaleźć się wyrażenia **ALTER** które zmodyfikują istniejącą strukturę.

1. Napisać polecenie **ALTER TABLE,** które doda kolumnę wyliczalną (np. wykona uppercase/lowercase na kolumnie tekstowej, albo wyliczy hash ze wszystkich kolumn za pomocą funkcji STANDARD\_HASH [link](https://docs.oracle.com/database/121/SQLRF/functions183.htm#SQLRF55647))
2. Napisać polecenie **ALTER TABLE** które zmodyfikuje istniejącą kolumnę tekstową i zwiększy jej długość