Упражнение 8. SQL: DDL и DML синтаксис

DDL – Data Definition Language
Командата за създаване на база данни има следният общ вид:
create database име_на_база;

Пример:
create database dbcourse;
use dbcourse;

Командата за създаване на таблица има следният общ вид:
create table име_на_таблица (column1 type1 [constraint], column2 type2 [constraint] ...);

Пример:
create table Employees (id bigint primary key identity(1,1), first varchar(50) NOT NULL, last varchar(50) NOT NULL, age int DEFAULT 25, city varchar(200));

Ако желаем да променим съществуваща таблица, като добавим нова колона или ограничение, то синтаксисът е следният:

Пример:
alter table Employees
add constraint age_check check(age > 18 and age < 60);

За добавяне на NULL, IDENTITY и DEFAULT, към вече съществуваща колона на таблица, може да ползвате MS SQL Server Management Studio на MS SQL Server 2005 и Modify опцията за базата данни.

Командата за изтриване на таблица е следната:
drop table име_на_таблица;

Пример:
drop table Employees;

Командата за изтриване на базата е следната:
drop database име_на_базата;

Пример:
drop database dbcourse;

DML – Data Manipulation Language
Командата за вмъкване на редове в таблицата, има следният общ вид:
insert into име_на_таблица (колона1, колона2, ...) values (стойност1, стойност2, ...)

Пример:

```
insert into Employees values('John', 'Doe', 45, 'Sofia');
insert into Employees values('Jane', 'Doe', 25, 'Varna');
insert into Employees values('Ivan', 'Ivanov', 35, 'Burgas');
insert into Employees values('Petar', 'Mirkov', 15, 'Vidin');
insert into Employees values('Vasil', 'Petkov', 35, 'Sofia');
insert into Employees values('Nikolay', 'Petkov', 45, 'Sofia');
```

Командата за обновяване на ред от таблицата е:

```
update име_на_таблица set име_на_колона = 'стойност'
[where име_на_колона = 'стойност'];
```

Пример:

```
update Employees
set first = 'Stoyan'
where last = 'Petkov';
```

Командата за изтриване на ред от таблицата е:

```
delete from име_на_таблица
[where име_на_колона = ‘стойност’]
```

Пример:

```
delete from Employees
where last = 'Petkov'
```

Командата за избиране на редове от таблицата, има следния опростен общ вид:

```
select * | [all | distinct] колона1, колона2... | израз [as псевдоним]
from име_на_таблица;
```

Задачи

Създайте базата от данни Education;

Създайте таблицата Students

- ID int // fn
- EGN varchar(10)
- Name varchar (50)
- City varchar(50)

Създайте таблицата Subjects

- ID identity(1,1)
- Name varchar (50)

Създайте таблицата Marks

- MarkID identity(1,1)
- StudentID FK Students(ID)
- SubjectID FK Subjects(ID)
- Date
- Mark

Вмъкнете в таблиците следните данни:

Students (ID, EGN, Name, City);
```
insert into Students values(43123, '8612035678', 'Maria', 'Sofia');
insert into Students values(43254, '8511045234', 'Antonia', 'Varna');
insert into Students values(43354, '8408091287', 'Milen', 'Plovdiv');
insert into Students values(30245,'8402220237', 'Sonia', 'Pleven');
insert into Students values(10523,'8507053487', 'Venelin', 'Burgas');
insert into Students values(12987,'8101091862', 'Petur', 'Sofia');
insert into Students values(10345,'8304182397', 'Ianka', 'Plovdiv');
insert into Students values(44002,'8412192453', 'Jordanka', 'Varna');
insert into Students values(42789,'8506231734', 'Gergana', 'Pleven');
insert into Students values(12003,'8510112661', 'Bojidar', 'Stara Zagora');
```

Subjects (ID, Name);
```
insert into Subjects values('Geometry');
insert into Subjects values('Database');
insert into Subjects values('Logic');
insert into Subjects values('Functional programming');
insert into Subjects values('Operation systems');
insert into Subjects values('Procedural programming');
insert into Subjects values('Artificial Intelligence');
insert into Subjects values('Computer Networks');
insert into Subjects values('Semantics of programming language');
```

Marks (MarkID, StudentID, SubjectID, Date, Mark);
```
insert into Marks values(43123, 1,'2007-06-02',6);
insert into Marks values(43123, 2,'2008-06-02',3);
insert into Marks values(43123, 3,'2008-06-02',4);
insert into Marks values(43123, 4,'2008-02-02',5);
insert into Marks values(43123, 5,'2007-06-26',3);
insert into Marks values(43254, 3,'2007-06-02',4);
insert into Marks values(43254, 4,'2007-02-02',6);
insert into Marks values(30245, 5,'2008-07-02',5);
insert into Marks values(30245, 2,'2006-06-02',6);
insert into Marks values(42789, 4,'2008-06-02',3);
insert into Marks values(42789, 6,'2007-07-02',5);
insert into Marks values(42789, 8,'2007-02-12',6);
insert into Marks values(42789, 6,'2007-04-02',4);
insert into Marks values(10345, 2,'2007-03-02',3);
insert into Marks values(10345, 3,'2007-06-22',5);
insert into Marks values(10345, 5,'2007-06-12',6);
```

Изтрийте таблицата Students;
Изтрийте таблицата Subjects;
Изтрийте таблицата Marks;
Изтрийте базата данни Education;
Запишете скрипта за създаване и изтриване на базата данни Education;

След като се научихме на основните команди за дефиниране на схеми и манипулиране на данните в тях сега ще разгледаме частта от SQL, която включва описание на структурата на информацията в базата от данни (т.нар. описание на данните - data definition). Аспектите на SQL разгледани досега – извличане на данни и модификации на данните - често се наричат обработка на данните (data manipulation). Дефинирането на схеми от съхранени релации (таблици).

Типове данни

Всички атрибути от една релационна схема трябва да имат тип данни. Зада започнем, ще въведем някои атомарни типове данни, които се поддържат от повечето SQL системи.

Символни низове с фиксирана или плаваща дължина.
Типът CHAR(n) обозначава низ с фиксирана дължина от n символа. Това означава, че ако един атрибут е от тип CHAR(n), тогава във всеки кортеж, съответната компонентата за този атрибут, ще низ от n символа.

VARCHAR(n) обозначава низ с до най-много n символа. Компонентите на атрибути от този тип ще са низове с дължина от 0 до n символа. Обикновено низът се допълва с интервали, ако стане стойност на компонент с фиксирана дължина, която е по-голяма от неговата собствена. Например ако низът 'foo' стане компонента на атрибут от тип

CHAR(5) ще приеме стойност 'foo '. Допълнините интервали ще бъдат игнорирани, ако стойността на компонентата се сравнява с друг низ.

Битови низове с фиксирана или плаваща дължина.
Те са аналог на низовете, но стойностите им са низове от битове, а не от символи. Типът BIT(n) обозначава битов низ с дължина n, докато BIT VARYING(n) обозначава битов низ с дължина до n бита.

Типът INT или INTEGER (синоними са) обозначава целочислени стойности.

Числата с плаваща запетая могат да бъдат представени по редица начини. Можем да използваме типовете FLOAT или REAL. По-голяма точност може да бъде постигната с типа DOUBLE PRECISION (разликата между тези типове е като в езика C). SQL има и типове, които са реални числа с фиксирана десетична точка. Например DECIMAL(n,d) допуска стойности, които се състоят от n десетични цифри с десетична точка, допускаща d позиции надясно. По този начин 0123.45 е възможна стойност на типа DECIMAL(6,2). NUMERIC е синоним на DECIMAL, въпреки че в зависимост от реализацията може да съществуват известни разлики.

Дати и време могат да бъдат представени чрез типовете DATE и TIME. Стойностите им напрактика са символни низове в специален формат.

Множеството от типовете данни се различава в различните СУБД. За MS SQL Server основните типове са:

Binary	binary [(n)] - До 8000 байта двоични данни
bit	Цялото число 0 или 1
varbinary	varbinary [(*n*)] - Низ с променлива дължина (до 8000 байта), съдържащ бинарни данни
image	Низ от двоични данни с променлива дължина с максимален размер от 2^31
char	char[(*n*)] - Низ с фиксирана дължина (до 8000), съдържащ символи, които не се кодират като Unicode
nchar	nchar(n) - Низ с фиксирана дължина(до 4000), съдържащ символи, кодирани като Unicode
varchar	varchar[(n)] - Низ с променлива дължина (максимално до 8,000), съдържащ символи, които не се кодират като Unicode
nvarchar	nvarchar(n) -Низ с променлива дължина, съдържащ символи, кодирани като Unicode
text	Низ с променлива дължина (максимално до 2^31-1), съдържащ символи, които не си кодират като Unicode
ntext	Низ с променлива дължина(максимално до 2^31-1), съдържащ символи, кодирани като Unicode
decimal	Числа, включващи десетичните дроби от -10^38 до 10^38-1 decimal[(p[, s])]- Максималната точност е p= 38, 0 <= s <= p. По подразбиране s=0
numeric	numeric[(p[, s])]- Същото като decimal
float	Десетични числа с плаваща запетая между -1.79E+308 и 1.79E +308 float [(n)], където n е от 1 до 53 и определя броя битове за съхранение на мантисата. Синоним на double precision в MS SQL Server е float(53)
real	Десетични числа с плаваща запетая между -3.40E^38 и 3.40E^38. Синоним на real е float(24).
bigint	Целите числа между -2^63 (-9223372036854775808) и 2^63-1 (9223372036854775807).
int	Целите числа между -2^31 (-2,147,483,648) и 2^31 (2,147,483,647)
smallint	Целите числа от -2^15 (-32,768) до 2^15 (32,767)
tinyint	Целите числа от 0 до 255
datetime	Датите и часовете от 1/1/1753 до 12/31/9999
smalldatetime	Датите и часовете от 1/1/1900 до 6/6/2079
money	Числа, представляващи парични стойности от -2^63 до 2^63
smallmoney	Числа, представляващи парични стойности от -2147483648 до 2147483647
sysname	nvarchar(128). Потребителски дефиниран тип. Използва се за обръщение на имена на обекти от базата данни

timestamp	Уникално число в базата от данни
uniqueidentifier	Глобален идентификатор GUID, уникално за целия свят число

Дефиниране на таблици

Най-простата форма за дефиниране на релационна схема се състой от ключовите думи CREATE TABLE, последвани от името на релацията и списък от имена на атрибути и техните типове, заграден в скоби.

CREATE TABLE *име_на_таблица*
(*име_на_колона1* тип_на_данни1(размер)
[, *име_на_колона2* тип_на_данни2(размер), ...]);

Пример 1:
Релационната схема на релацията MovieStar от нашия пример, изразана чрез езика SQL изглежда по следния начин:

```
CREATE TABLE MovieStar(
name CHAR(30),
address VARCHAR(255),
gender CHAR(1),
birthdate DATETIME
);
```

Първите два атрибута, name и address, са декларирани като символни низове. По отношение на името е взето решение да се използват низове с фиксирана дължина от 30 символа, допълвайки името с интервали, ако е необходимо и отрязвайки го до 30 символа, ако е по-дълго. За разлика от името, адресът е деклариран като низ с променлива дължина до 255 символа. Не е ясно дали тези два избора са възможно най-добрите, но ги използваме за да илюстрираме двата вида низови типове данни.
Атрибутът gender (пол) приема стойности, които са отделна буква – M или F. Накрая, атрибутът birthdate естествено е от тип DATE (DATETIME в MS SQL Server).

Модифициране на релационни схеми

Можем да изтрием релацията R чрез следния SQL оператор:
DROP TABLE *R*;

Релацията R вече не е част от схемата на базата от данни и вече нямаме достъп до нейните кортежи. Много по често ни се налага да модифицираме схемата на съществуваща релация, отколкото да я изтрием. Тези операции са възможни посредством оператор, започващ с ключоватите думи ALTER TABLE и името на релацията. Имаме различни опции, най-важните от които са:
Ключовата дума ADD, последвана от име на колона и тип данни.
Ключовата дума DROP, последвана от име на колона.

Пример 2:
Можем да модифицираме релацията MovieStar, като добавим атрибут phone:

ALTER TABLE MovieStar ADD phone CHAR(16);

В резултат на изпълнението на този оператор схемата на релацията MovieStar има 5 атрибута: 4-те споменати в горния пример и атрибута phone от тип низ с фиксирана дължина от 16 бита. Кортежите на релацията ще имат компонент phone, като за съществуващите до момента на добавянето на атрибута кортежи, стойността му е неизвестна и ще има стойност NULL. Възможно е да укажем тази стойност да е някаква подразбираща се стойност различна от NULL.

Пример 3:
Изтриването на атрибута birthdate може да се осъществи по следния начин:

ALTER TABLE MovieStar DROP birthdate;

Забележка: В MS SQL Server е необходимо да се добави и ключовата дума COLUMN след DROP:

ALTER TABLE MovieStar DROP COLUMN birthdate;

Друга типична употреба на оператора ALTER TABLE е свързана с модификацията на съществуваща колона – можем да се промени нейния тип, размер или стойност по подразбиране. В някой СУБД за целта се използва ключовата дума MODIFY, последвана от име на колона. В MS SQL Server вместо нея се използват думите ALTER COLUMN

Пример 4:

ALTER TABLE MovieStar ALTER COLUMN name VARCHAR(40);

Стойности по подразбиране
Когато създаваме или модифицираме кортежи, понякога нямаме стойностти за всички налични компоненти. Необходимо е да използваме някаква стойност, която указва, че стойността на атрибута е неизвестена. SQL предоставя стойността NULL. Тя става стойност на онези компоненти, чиито стойности са не определени, с изключение на някои ситуации, в които стойността NULL е недопустима. Има случай, в които предпочитаме да използваме друга стойност по подразбиране за колоните с неизвестна стойност. Принципно, на всяко място, където декларираме атрибут и негов тип, можем да добавим ключовата дума DEFAULT и подходяща стойност. Тази стойност може да е NULL или някаква константа. Някой други стойности, които се предоставят от системата, като текущата дата също е възможно да са допустими.

Пример 5:
Да разгледаме Пример1. Може да искаме да използваме символа “?”, като подразбираща се стойност за атрибута gender и най-ранната възможна дата (например '1/1/1753') за неизвестен рожден ден.

```
CREATE TABLE MovieStar(
name CHAR(30),
address VARCHAR(255),
gender CHAR(1) DEFAULT '?',
birthdate DATETIME DEFAULT '1/1/1753'
);
```

Пример 6:

```
ALTER TABLE MovieStar DROP COLUMN birthdate;
ALTER TABLE MovieStar ADD birthdate DATETIME DEFAULT  getdate();
```

Пример 7:

Можем да декларираме стойност по подразбиране за атрибута phone да бъде 'unlisted', когато го добавяме към релационната схема на MovieStar :

```
ALTER TABLE MovieStar ADD phone CHAR(16) DEFAULT 'unlisted' ;
```

Изтриване на всички редове в таблицата

Освен чрез оператора DELETE, редовете в една таблица могат да бъдат изтрити и посредством оператора **TRUNCATE TABLE.** Този оператор обаче е по-бърз и използва по-малко системни ресурси от DELETE. Има ситуации обаче, в които TRUNCATE TABLE не може да бъде използван.

Създаване на база от данни чрез SQL

Минималният синтаксис на командата за създаване на база от данни е:

CREATE DATABASE *име_на_базата_от_данни*;

В зависимост от избраната СУБД командата може да има различни разширения.

За да се покаже коя база от данни се използва в Query Analyzer за MS SQL Server може да се използва команда:

USE *име_на_базата_от_данни*;

както и да се използва визуалната среда за смяна на базата.

Обобщение:

CREATE TABLE *име_на_таблица* (*име_на_колона1* тип_на_данни1(размер) [DEFAULT константен_израз] [, *име_на_колона2* тип_на_данни2(размер) [DEFAULT константен_израз] , ...]);
ALTER TABLE *име_на_таблица* **ALTER COLUMN** *име_на_колона* нов_тип_данни(размер));
ALTER TABLE *име_на_таблица* **ADD** *име_на_колона* тип_данни(размер) [DEFAULT константен_израз] [,*име_на_колона* тип_данни(размер) [DEFAULT константен_израз] , ...]);

ALTER TABLE *име_на_таблица* **DROP COLUMN** *име_на_колона* [,*име_на_колона*] , ...]);
DROP TABLE *име_на_таблица;*
TRUNCATE TABLE *име_на_таблица;*
CREATE DATABASE *име_на_базата_данни;*
DROP DATABASE *име_на_базата_данни;*

Задачи

1. Създайте база от данни FNxxxxxMyDB
2. Дефинирайте по подходящ начин следните таблици:
    a. Classes(class,type,country, numGuns,bore, displacement)
    b. Ships(name,class,launched) - задайте за колоната launched стойност по подразбиране текущата дата на вмъкване на записа
    c. Battles(name, date)
    d. Outcomes(ship,battle,result)
3. Изтрийте от таблицата Classes колоните bore и displacement.
4. Добавете към таблицата Ships колона shipyard от тип varchar(30), която указва името на корабостроителницата, в която е построен кораба.
5. Променете размера на колоната shipyard на varchar(40)
6. Изтрийте база от данни FNxxxxxMyDB.