Vytvorenie databáz, tabuliek a integrita dát - 1

A) DB – súbory, stavy, nastavenia
B) Systémové príkazy o DB a tabuľkách
C) Vytvorenie DB
D) Vytvorenie tabuliek – kaskádovité mazanie a update

A) DB – súbory, stavy, nastavenia Systémové DB http://msdn.microsoft.com/en-us/library/ms179422.aspx

a) Súbory

- **master DB** - (metadata) info o všetkých DB a miestach ich DB súborov, užívateľské kontá, systémové konfigurácie
- **model DB** - vzor pre vytvorenie všetkých DB
- **resource DB** - obsahuje sys.objects
- **tempDB**
- **sys.databases, sys.tables** – pozri nižšie

b) Stavy a nastavenia DB

Každá DB v SQL Server obsahuje dva systémové súbory: http://msdn.microsoft.com/en-us/library/ms189237.aspx

- **dátový súbor** – primárny *.mdf* a sekundárny *.ndf* (používateľom definovaný) - tabuľky, indexy, uložené procedúry (SP), VIEW
- **log súbor** - *.ldf* informácie pre obnovu transakcií v DB

Stavy DB/súborov http://msdn.microsoft.com/en-us/library/ms190442.aspx

- ONLINE (Databáza je k dispozícii pre prístup),
- OFFLINE (napr. ukladá sa súbor na HD),
- SUSPECT (napr. po zlyhaní obnovy DB, DB môže byť poškodený),
- RESTORING (obnova súborov),
- RECOVERING (obnova DB), ...

Nastavenia(voľby, options) DB http://msdn.microsoft.com/en-us/library/ms190249.aspx

Nastavenia sú jedinečné pre každú databázu a nemajú vplyv na iné databázy.

- **Auto nastavenia** - napr. AUTO_CLOSE ON (zdroje sú oslobodené po uzavretí DB), AUTO_CREATE_STATISTICS ON (štatistiky sú automaticky vytvorené pre stĺpce použité v predikátu)
- **Cursor** (ak je nastavené na ON, všetky otvorené kurzory budú po vykonaní transakcie uzavreté.)
- **Prístupnosť DB** - OFFLINE, READ_ONLY, MULTI_USER
- **Optimalizácia korelácie dát**
- **Parametrizácia dopytov**
- **Obnova** - RECOVERY
- **SQL** - ANSI_NULLS, ...

B) Systémové príkazy o DB a tabuľkách

```
SELECT name, create_date FROM sys.databases
```

```
--  a) Vsetky DB:
USE master;
GO
SELECT * FROM sys.databases
SELECT name, create_date FROM sys.databases
--SELECT name, crdate, filename FROM sys.sysdatabases  -- stare
EXEC sp_databases -- SIZE

-- b) Vsetky tabulky:
use OsobaVztah
GO
SELECT * FROM sys.tables;

EXEC sp_tables
EXEC sp_tables @TABLE_OWNER = dbo

-- c) Vsetky stlpce:   -- !!! Ctrl + T , Ctrl + D (, Ctrl + R)
EXEC sp_columns Osoba
EXEC sp_columns @table_name = Osoba, @column_name = 'm%'
SELECT * FROM sys.columns;

-- a4) Vypis kodu SP:
EXEC sp_helptext sp_columns -- trigger, view
EXEC sp_helpdb   -- name, db_size, owner, dbid, created, status
```

C) Vytvorenie DB

Doteraz sme sa zaoberali predovšetkým s príkazom SELECT. Teraz sa pozrieme podrobnejšie na DDL, ale aj na DML príkazy.

DDL, DML (Data Definition & Manipulation Language), dopyt (query) – prezeranie / prehľadávanie

DB:	Table:	Údaje:
- CREATE	- CREATE	INSERT
- ALTER	- ALTER	UPDATE, MERGE
- DROP	- DROP	DELETE
		SELECT - dopyt

- **CREATE, ALTER, DROP** DB (aj Tab)
- **DELETE tab** – odstráni riadky

DROP - dáta, indexy, triggery, obmedzenia a prístupové špecifikácie

- DB OsobyProjektyVydavky

Osoby

Priezvisko	Krstne
A	a
B	b
C	c

Projekty

Nazov	Veduci
X	B
Y	C
Z	C

Vydaje

Projekt	Polozka	Cena
X	PC	30
X	DVD	2
Y	Cesta	9
Z	Tlaciaren	10
Z	PC	35

```
USE master;
GO

IF EXISTS(SELECT * from sys.databases WHERE name='OsobyProjektyVydavky')
-- ⇔
IF DB_ID ('OsobyProjektyVydavky') IS NOT NULL
                                    DROP DATABASE OsobyProjektyVydavky;
GO

CREATE DATABASE     OsobyProjektyVydavky;
GO

USE OsobyProjektyVydavky;
GO
```

D) Vytvorenie tabuliek a kaskádovité mazanie a update

Nižšie ilustrujeme niekoľko možností pri navrhovaní tabuliek. Najprv vždy uvedieme príkazy, ktoré môžu mať nežiadúci výsledok (nie nutne chybu) a potom riešenie (resp. návrhové rozhodnutie). Budeme ich označovať *k) resp. #k).

Nežiadúci výsledok
*1) – vkladáme riadok s hodnotou, ktorá sa odvoláva na neexistujúcu hodnotu
*2) – chceme odstrániť riadok s hodnotou, na ktorú sa odvoláva
*3) – nedajú sa odstrániť závislé tabuľky
*4) – chceme zmeniť hodnotu v riadku, na ktorú sa odvoláva

```
--IF OBJECT_ID('Osoby') IS NOT NULL DROP TABLE Osoby;
--GO
CREATE TABLE Osoby
(
  idOs          int  NOT NULL PRIMARY KEY,
  Priezvisko varchar(20),
  Krstne   varchar(20)
);
GO

CREATE TABLE Projekty
(
  idPr    int NOT NULL PRIMARY KEY,
  Nazov   varchar(40),
  idOs    int
);
GO

CREATE TABLE Vydaje
(
  idVy         int IDENTITY(1, 1) PRIMARY KEY, -- od, po
  idPr         int,
  Polozka varchar(30) DEFAULT 'Polozka xyz',
  Cena         money
);
GO

INSERT Osoby VALUES (1, 'A', 'a' );
INSERT Osoby VALUES (2, 'B', 'b' );
INSERT Osoby VALUES (3, 'C', 'c' );
GO

-- *1) – vkladáme riadok s hodnotou, ktorá sa odvoláva na neexistujúcu hodnotu
-- Nemal by prejst, ale tu prejde
-- INSERT Projekty VALUES (1, 'X', 5);
-- Delete Projekty --where idPr =1
Riesenie 1 - Vymazat a pisat iba spravne - kontrola mimo DBS
--       2 - az po chybe 1b)!!! - cudzi kluc FK

INSERT Projekty VALUES (1, 'X', 2);
INSERT Projekty VALUES (2, 'Y', 3);
INSERT INTO Projekty(idPr, Nazov, idOs)  VALUES (3, 'Z', 3);
GO
```

```
SET IDENTITY_INSERT Vydaje ON
INSERT Vydaje(idVy, idPr, Polozka, Cena) VALUES (1, 1, 'PC',    30);
SET IDENTITY_INSERT Vydaje OFF
INSERT Vydaje(      idPr, Polozka, Cena) VALUES (   1, 'DVD',    2);
INSERT Vydaje(      idPr,          Cena) VALUES (   2,           9);
INSERT Vydaje(      idPr, Polozka, Cena) VALUES (   3, 'Tlaciaren', 10);
INSERT Vydaje(      idPr,          Cena) VALUES (   3,          35);
GO

SELECT * FROM Osoby;
SELECT * FROM Projekty;
SELECT * FROM Vydaje;

DELETE FROM Vydaje   WHERE idVy = 5;
--  *2) – chceme odstrániť riadok s hodnotou, na ktorú sa odvoláva
--  Nemal by dovolit  ** (alebo mazat kaskadovite – navthove rozhodnutie***)
--DELETE FROM Osoby  WHERE idOs = 2;
--DELETE FROM Projekty WHERE idPr = 2;
--SELECT * FROM Osoby;
--SELECT * FROM Projekty;
--SELECT * FROM Vydaje;
```

Osoby

Priezvisko	Krstne
A	a
C	c

Projekty

Nazov	Veduci
Z	C

Vydaje

Projekt	Polozka	Cena
Z	Tlaciaren	10

	id	Priezvisko	Krstne
1	1	A	a
2	3	C	c

	id	Naz...	idVed
1	3	Z	3

	id	idProj	Polozka	Cena
1	4	3	Tlaciaren	10,00

Výsledkom nie sú horné tri dobré tabuľky ale dolné tri zlé:

	idOs	Priezvisko	Krstne
1	1	A	a
2	3	C	c

	idPr	Naz...	idOs
1	1	X	2
2	3	Z	3

	idVy	idPr	Polozka	Cena
1	1	1	PC	30,00
2	2	1	DVD	2,00
3	3	2	Polozka xyz	9,00
4	4	3	Tlaciaren	10,00

```
-- #1), #2) Riesenie FK – vlozit minimalne pred INSERTovanim do Projektov
ALTER TABLE Projekty ADD CONSTRAINT fk_idOs FOREIGN KEY (idOs)
REFERENCES Osoby(idOs)
ALTER TABLE Vydaje   ADD CONSTRAINT fk_idPr FOREIGN KEY (idPr)
REFERENCES Projekty(idPr)
-- <=> in tab Vydaje:
  idPr int FOREIGN KEY (idPr) REFERENCES Projekty(idPr),

... From 1a) INSERT ...

-- *3) – nedajú sa odstrániť závislé tabuľky – kvoli FK
--IF OBJECT_ID('Osoby')    IS NOT NULL DROP TABLE Osoby;
--IF OBJECT_ID('Projekty') IS NOT NULL DROP TABLE Projekty;
--IF OBJECT_ID('Vydaje')   IS NOT NULL DROP TABLE Vydaje;
-- #3) Riesenie - mazat (drop) v spravnom poradi (najprv odstranime tabulku, na ktoru sa neodvolava,
teda Vydaje, ...):
...
DELETE FROM Vydaje   WHERE idVy = 5;
---- 2b) OK - nedovoli:
 --DELETE FROM Osoby      WHERE idOs = 2;
--DELETE FROM Projekty WHERE idPr = 2;
--  ALE ak chcem mazat kaskadovite ***?
--  Riesenie *** kaskadovite mazanie: ON DELETE CASCADE
ALTER TABLE Projekty ADD CONSTRAINT fk_idOs FOREIGN KEY (idOs)
REFERENCES Osoby(idOs)
                         ON DELETE CASCADE
ALTER TABLE Vydaje  ADD CONSTRAINT fk_idPr FOREIGN KEY (idPr)
REFERENCES Projekty(idPr)
                         ON DELETE CASCADE

-- *4) – chceme zmeniť hodnotu riadku, na ktorú sa odvoláva
-- Nedovoli, ale my chceme
UPDATE Osoby SET idOs = -1
     WHERE idOs = 3;
GO
-- #4) Riesenie: ON UPDATE CASCADE
ALTER TABLE Projekty ADD CONSTRAINT fk_idOs FOREIGN KEY (idOs)
REFERENCES Osoby(idOs)
                         ON DELETE CASCADE   ON UPDATE CASCADE;
ALTER TABLE Vydaje  ADD CONSTRAINT fk_idPr FOREIGN KEY (idPr) REFERENCES
Projekty(idPr)
                         ON DELETE CASCADE   ON UPDATE CASCADE;
DELETE FROM Vydaje   WHERE idVy = 5;
DELETE FROM Osoby    WHERE idOs = 2;
DELETE FROM Projekty WHERE idPr = 2;
```