Abschlussprojekt

Dozent: Lev A. Brodski
Modul: SQL
Stand: August 2023

Aufgabe:

Eine relationale Datenbank entwerfen, implementieren und programmieren

- dabei **Redundanzen** und **Inkonsistenzen** vermeiden
- DB soll **dynamische Daten** verwalten (nicht nur statische Tabellen)
- **Business-Logik implementieren** (s. unten - Prozeduren und Trigger)
- *für die Note „gut“* - mit 7 bis 9 Tabellen
- *für die Note „sehr gut“* - mit mind. 10 Tabellen
- **Primärindizes** und gegebenenfalls **Sekundärindizes** (NONCLUSTERED INDEX) erstellen
- **Beziehungen** zwischen Tabellen erstellen
- **Mindestens eine m:n Beziehung**
- **Standartwerte und Einschränkungen** erstellen
- **Diagramm** erstellen
- **CREATE - Skripte** speichern:
  - CREATE DATABASE
  - **Je ein Skript:**
    - CREATE TABLE
    - CREATE NONCLUSTERED INDEX
    - ADD CONSTRAINT FOREIGN KEY
    - ADD CONSTRAINT CHECK

Daten

- Genug Datensätze um Business-Logik prüfen zu können, aber nicht zu viel

Sichten (VIEW)

- ein SELECT - Sicht über mehreren Tabellen erstellen, die Tabellen mit INNER JOIN verknüpfen
- ein SELECT - Sicht über mehreren Tabellen mit GROUP BY, SUM oder COUNT und HAVING erstellen
- *für die Note „sehr gut“* - Sicht über mehreren Tabellen erstellen, die Tabellen mit LEFT/RIGHT/FULL OUTER JOIN oder SELF JOIN verknüpfen
- **alle Skripte speichern**

Gespeicherte Funktionen

- eine **Skalarwertfunktion** erstellen
- eine **Tabellenwertfunktion** erstellen
- **Testskripte** erstellen
- **alle Skripte speichern**

Gespeicherte Prozeduren

- eine **gespeicherte Prozedur** mit Parameter erstellen
- die Prozedur soll **Business-Logik** implementieren
- die Prozedur soll **Eingangsparameter prüfen** und **Daten** in Tabellen **ändern**
- *für die Note „sehr gut“*: in der Prozedur eigene gespeicherte Funktionen verwenden
- *für die Note „sehr gut“*: in der Prozedur OUTPUT-Parameter für die Fehlermeldungen etc. verwenden
- Skripte für Ausführen erstellen
- **alle Skripte speichern**

Trigger

- ein Trigger (INSERT, UPDATE oder DELETE)
- *für die Note „sehr gut“*: Trigger soll Business-Prozesse automatisieren
- Testskript erstellen
- **alle Skripte speichern**

Anmeldungen, Benutzer, Rechte (nur wenn die Zeit reicht)

- DB-Benutzer mit Lese-, Schreib- und EXEC-Rechte anlegen

BACKUP

- DB sichern, .bak-Datei speichern
- Ein Skript für gesamte DB und alle DB-Objekte vom SSMS generieren lassen:
  - alle Tabellen, Sichten, Prozeduren etc.
  - alle Logins, User, Rollen
  - alle Daten
  - **OHNE DROP!**
- Die zwei DB-Datei, .mdf und .ldf speichern (nicht obligatorisch, nur wenn nicht mehr zu tun ist:)

für die Note „sehr gut“:

- **CURSOR-Skript** erstellen (wenn Ihre Business-Logik einen braucht)
- oder **Daten importieren** (z.B. aus EXCEL)
- oder Access/Excel/Power BI/*** verbinden und **Front-End** entwickeln (1 Formular und 1 Bericht)
- oder **1:1 Beziehung**
- oder **1 Zirkelbezug + Views**
- oder **mehr** Funktionen, Prozeduren und/oder Trigger schreiben

Projektdokumentation:

- Eine kurze Projektbeschreibung (s. Muster) als .pdf
- Datenbankdiagramm hinzufügen

Für alle Aufgaben gilt:

- Als Muster für die Skripte können die Übung-Skripte genutzt werden
- Benutzen Sie hierbei eigene Namen (D/EN, in camelCase oder PascalCase) für alle Ihre Funktionen, Variablen, Feldnamen etc.
- Alle Skripte / Funktionen sollten ausreichend kommentiert sein (lieber auf Deutsch, aber Englisch ist auch OK)

Sollten Sie noch Zeit haben, erweitern Sie ihr Projekt nach Ihrem Geschmack und Kenntnisstand (mehr Tabellen, Funktionen, Prozeduren etc.).