Dateiname	Objekte	Beschreibung	Verwendete Technik
Personendaten_ANSI.csv		csv-Datei, die per BULK-Import-Befehl importiert wird	Muss gespeichert werden unter D:\DB\ damit der BULK Import im Skript 03_InsertData.sql später funktioniert
PlzOrt_Ansi.csv		csv-Datei, die per BULK-Import-Befehl importiert wird	Muss gespeichert werden unter D:\DB\ damit der BULK Import im Skript 03_InsertData.sql später funktioniert
00_CreateDatabase.sql		Erstellung der Datenbank	
01_CreateTables.sql	tb_Person	Erstellung der Tabelle tb_Person mit allen relevanten Personendaten mit PK. Indizierung des Nachnamens mit Duplikaten. Einschränkung auf Geschlecht: m/w/d.	
	tb_PlzOrt	Erstellung einer Referenztabelle mit PK für die Tabelle tb_Person zur Auslagerung der transitiv abhängigen Daten Plz und Ort. Einschränkung für die Länge der Plz: 5 Zeichen.	
	tb_Trainer	Erstellung der Tabelle tb_Trainer, die alle Personen beinhaltet, die einer Tätigkeit als Trainer nachgehen, mit PK. Eindeutige Indizierung der PersonID. Mehrere Default-Werte (z.B. Erfahrung in Jahren = 0).	
	tb_Teilnehmer	Erstellung der Tabelle tb_Teilnehmer, die alle Personen beinhaltet, die an Trainingsaktivitäten von ZimmerSport teilnehmen, mit PK. Eindeutige Indizierung der PersonID.	

	tb_PersonRolle	Erstellung einer Zuordnungstabelle mit PK, die den Personen weitere Rollen außer der Trainer- und Teilnehmerrolle zuordnet.	

		Eindeutige Indizierung der Kombination aus PersonID und RolleID.	
	tb_Rolle	Erstellung einer Referenztabelle mit PK für die tb_PersonRolle, die weitere Rollen außer Trainer und Teilnehmer abbildet. Eindeutige Indizierung der Spalte Beschreibung.	
	tb_TeilnehmerKurs	Erstellung einer Zuordnungstabelle mit PK, in der jedem Teilnehmer ein Kurs zugeordnet wird und weitere Informationen des Teilnehmers festgehalten werden. Eindeutige Indizierung der Kombination aus TeilnehmerID und KursID.	
	tb_Sportlichkeit	Erstellung einer Referenztabelle mit PK, in der das Sportlichkeitslevel kodiert wird. Eindeutige Indizierung der Spalte Beschreibung.	
	tb_Erfahrung	Erstellung einer Referenztabelle mit PK, in der die Erfahrung mit Functional Training kodiert wird. Eindeutige Indizierung der Spalte Beschreibung.	

	tb_Kontraindikation	Erstellung einer Referenztabelle mit PK, in der mögliche Kontraindikationen der Teilnehmer kodiert werden. Eindeutige Indizierung der Spalte Beschreibung.	
	tb_Kurs	Erstellung der Tabelle tb_Kurs mit allen relevanten Kursdaten mit PK. Einschränkungen für die Teilnehmerzahl (0 < MinTeilnehmerzahl < MaxTeilnehmerzahl) sowie das Datum (StartDatum < EndDatum).	

	tb_Kurstyp	Erstellung einer Referenztabelle mit PK, in der die Kurstypen hinterlegt sind. Eindeutige Indizierung der Spalte Beschreibung.	
	tb_Kurseinheit	Erstellung der Tabelle tb_Kurseinheit mit PK. Ein Kurs besteht aus mehreren Kurseinheiten. Fällt ein Trainer (bspw. durch Krankheit) für eine Kurseinheit aus, kann hier eine andere TrainerID hinterlegt werden. Eindeutige Indizierung der Kombination aus KursID, Datum und TrainerID (IX_KursID_Datum_TrainerID).	Eindeutige Indizierung über 3 Spalten.
	tb_Trainingsplan	Erstellung der Tabelle tb_Trainingsplan, die alle zeitlichen Daten des Kurses enthält. Default-Wert für die Dauer (DauerMin = 60). Einschränkung 1 <= Wochentag <= 7.	

02_CreateForeignKeys.sql		Alle ForeignKeys (FK) der Datenbank ZimmerSport. Benamung von ForeignKeys ist immer der FKName_PKName.	Zuerst wird mit IF EXISTS (SELECT * FROM sys.foreign_keys WHERE name= FK) geprüft, ob der FK bereits in der Datenbank existiert. Ist das so wird der FK gelöscht, damit der FK in jedem Fall neu erstellt werden kann
	FK_tb_PersonRolle_tb_Person	Erstellung eines FK von der Tabelle tb_Person zur Tabelle tb_PersonRolle	Beziehung 1:n
	FK_tb_PersonRolle_tb_Rolle	Erstellung eines FK von der Tabelle tb_PersonRolle zur Tabelle tb_Rolle	Beziehung 1:n
	FK_tb_Person_tb_PlzOrt	Erstellung eines FK von der Tabelle tb_Person zur Tabelle tb_PlzOrt	Beziehung 1:n
	FK_tb_TeilnehmerKurs_tb_Erfahrung	Erstellung eines FK von der Tabelle tb_TeilnehmerKurs zur Tabelle tb_ Erfahrung	Beziehung 1:n

	FK_tb_TeilnehmerKurs_tb_Kontraindikation	Erstellung eines FK von der Tabelle tb_TeilnehmerKurs zur Tabelle tb_ Kontraindikation	Beziehung 1:n
	FK_tb_TeilnehmerKurs_tb_Sportlichkeit	Erstellung eines FK von der Tabelle tb_TeilnehmerKurs zur Tabelle tb_ Sportlichkeit	Beziehung 1:n
	FK_tb_Kurs_tb_KursTyp	Erstellung eines FK von der Tabelle tb_Kurs zur Tabelle tb_ KursTyp	Beziehung 1:n
	FK_tb_TeilnehmerKurs_tb_Kurs	Erstellung eines FK von der Tabelle tb_TeilnehmerKurs zur Tabelle tb_ Kurs	Beziehung 1:n

	FK_tb_Person_tb_Teilnehmer	Erstellung eines FK von der Tabelle tb_Person zur Tabelle tb_ Teinehmer	Beziehung 1:1
	FK_tb_Person_tb_Trainer	Erstellung eines FK von der Tabelle tb_Person zur Tabelle tb_Trainer	Beziehung 1:1
	FK_tb_Kurseinheit_tb_Kurs	Erstellung eines FK von der Tabelle tb_Kurseinheit zur Tabelle tb_Kurs	Beziehung 1:n
	FK_tb_Kurseinheit_tb_Trainer	Erstellung eines FK von der Tabelle tb_Kurseinheit zur Tabelle tb_Trainer	Beziehung 1:n
	FK_tb_Kurs_tb_Trainer	Erstellung eines FK von der Tabelle tb_Kurs zur Tabelle tb_Trainer	Beziehung 1:n
	FK_tb_TeilnehmerKurs_tb_Teilnehmer	Erstellung eines FK von der Tabelle tb_TeinehmerKurs zur Tabelle tb_Teilnehmer	Beziehung 1:n
	FK_tb_Trainingsplan_tb_Kurs	Erstellung eines FK von der Tabelle tb_Trainingsplan zur Tabelle tb_Kurs	Beziehung 1:n
03_InsertData.sql	tb_PlzOrt	Import einer CSV Datei, die neben der Plz und dem Ort noch das zugehörige Bundesland beinhaltete.	BULK INSERT der Daten aus CSV Datei
	tb_Person	Erstellung einer Hilfstabelle (tb_Person_Hilfstabelle) für die	BULK INSERT der Daten aus CSV Datei

		Personendaten, anschließender Import einer CSV Datei in die Hilfstabelle. Übertragung der Daten aus der Hilfstabelle in tb_Person. Übertragung der PlzOrtID in die tb_Person auf Grundlage der Plz und Ort Daten aus der Hilfstabelle. Löschen der Hilfstabelle	LEFT JOIN
	tb_Rolle tb_PersonRolle tb_Trainer tb_Kurstyp tb_Teilnehmer tb_Sportlichkeit tb_Erfahrung tb_Kontraindikation tb_Kurs tb_Kurseinheit tb_TeilnehmerKurs tb_Trainingsplan	Füllen aller Tabellen mit fiktiven Demodaten. Um zu gewährleisten, dass jedes Gruppenmitglied dieselben Daten verwendet, wird beim Einfügen vorübergehend der Autowert gestoppt.	SET Identity_Insert ON/OFF
04_CreateViews.sql		Alle erstellten Sichten	Zuerst wird mit IF EXISTS(SELECT * FROM sys.views WHERE name=VIEW) geprüft, ob die Sicht bereits in der Datenbank existiert. Ist das so wird diese gelöscht, damit sie in jedem Fall neu erstellt werden kann
	vw_TrainerTeilnehmer	Erstellung einer Sicht, die eine Übersicht aller Personen ausgibt, die sowohl Trainer als auch Teilnehmer sind	Abfrage greift auf selbst erstellte Sicht zurück
	vw_Trainer	Erstellung einer Sicht, die eine Übersicht aller Trainer erstellt	INNER JOIN

	vw_KurseMitPlatz	Erstellung einer Sicht, die prüft, ob noch Plätze in den Kursen vorhanden sind	Aggregatfunktion COUNT, INNER JOIN, HAVING, GROUP BY

	vw_AllePersonen	Liste über alle Personen inklusiver aller zugehöriger Attribute	LEFT JOIN, Alias AS für Einfügen von neuen Spaltennamen, Verwendung der IIF Bedingung, Verwendung einer selbst erstellten Skalarfunktion
	vw_AlleTeilnehmer	Auflistung belegter Kurse sowie ihrer Teilnehmer und ihrer Personendaten	RIGHT JOIN, INNER JOIN, Abfrage einer selbst erstellten Sicht
	vw_Kurse	Auflistung aller Kurse, inkl. Kursdaten und Trainerdaten	COUNT, INNER JOIN, Abfrage einer selbst erstellten Sicht
05_Functions.sql	tf_Uebersicht_TeilnehmerGeburtstag	Gibt eine Liste aller Teilnehmer und Personendaten zurück, welche in einem bestimmten Monat Geburtstag haben	Tabellenwertfunktion, 1 INPUTParameter INNER JOIN
	tf_KurseEinesTrainersUsersicht	Gibt eine Liste aller Kurse eines Trainers wieder inkl. Kursbeschreibung und Trainerdaten (für Kursinteressenten gedacht bei Kurssuche)	Tabellenwertfunktion, 1 INPUTParameter INNER JOIN
	sf_GetAge	Ermittelt das Alter anhand eines Geburtsdatums und liefert dieses zurück	Skalarwertfunktion, 1 INPUTParameter Verwendung der Systemfunktionen DATEDIFF, CURRENT_TIMESTAMP, geteilt durch 365,25 (wegen Schaltjahren) und mit FLOOR abgerundet
	tf_KurseEinesTrainersManagementsichtInputVorname Nachname	Gibt eine Liste aller Kurse eines Trainers zurück mit Angaben, die das Management interessieren anhand der Eingabe des Trainernamens	Tabellenwertfunktion, 2 INPUT-Parameter LEFT JOINs und INNER JOIN

	tf_KurseEinesTrainersManagementsichtInputTrainerID	Gibt eine Liste aller Kurse eines Trainers zurück mit Angaben, die das Management interessieren anhand der Eingabe der TrainerID	Tabellenwertfunktion, 1 INPUT-Parameter Besonderheit: Verwendung einer bereits selbst erstellten Tabellenwertfunktion
06_Trigger.sql	tr_KurseinheitTestDatum	Tigger für die Tabelle Kurseinheit, um sicherzustellen, dass eine Kurseinheit nur stattgefunden haben kann, wenn ihr Datum nicht in der Zukunft liegt.	DLM Trigger, AFTER UPDATE
07_Prozeduren.sql	sp_AddTeilnehmerKursIFSportlich	Prozedur, welche die Anmeldung eines Teilnehmer für einen Kurs nur zulässt, wenn sein Fitnesslevel nicht unter dem durchschnittlichen Fitnesslevel der anderen bereits angemeldeten Kursteilnehmer liegt und sein Alter in der Altersgruppe der bereits angemeldeten Kursteilnehmer liegt, falls sich mehr als 6 Teilnehmer für diesen Kurs bereits angemeldet haben	3 INPUT, 2 OUTPUT Parameter, Verwendung des TRY-CATCH Statements, THROW für Fehler Generierung, Verwendung einer selbst erstellten Skalarwertfunktion und den Systemfunktionen COUNT, AVG, Stdev und ROUND, Erstellung einer temporären Tabelle mit #
08_Benutzer.sql	20200819_Reader1, 20200819_Reader2, 20200819_Writer, 20200819_Exec	Erstellung von 4 SQL-Server-Anmeldung	
	20200819_Reader1	Erstellung eines SQL-Server-Benutzers mit ausschließlich Leserechten	
	20200819_Writer	Datenbank-Benutzer mit Lese- und Schreibrechten	
	20200819_Reader2	Erstellung eines SQL-Server-Benutzers mit ausschließlich Leserechten	

	20200819_Exec	Datenbank-Benutzer mit Lese-, Schreibrechten und Ausführrechten	
09_Testskripte.sql	vw_TrainerTeilnehmer	Testskript für die Sicht vw_TrainerTeilnehmer	
	vw_Trainer	Testskript für die Sicht vw_Trainer	
	vw_KurseMitPlatz	Testskript für die Sicht vw_KurseMitPlatz	
	vw_AllePersonen	Testskript für die Sicht vw_AllePersonen	
	vw_AlleTeilnehmer	Testskript für die Sicht vw_AlleTeilnehmer	
	vw_Kurse	Testskript für die Sicht vw_Kurse	
	sf_GetAge	Testskript für die Skalarwetfunktion sf_GetAge	
	tf_Uebersicht_TeilnehmerGeburtstag	Testskript für die Tabellenwertfunktion tf_Uebersicht_TeilnehmerGeburtstag	
	tf_KurseEinesTrainersUsersicht	Testskript für die Tabellenwertfunktion tf_KurseEinesTrainersUsersicht	
	tf_KurseEinesTrainersManagementsichtInputVornameNachname	Testskript für die Tabellenwertfunktion tf_KurseEinesTrainersManagementsichtInputVornameNachname	
	tf_KurseEinesTrainersManagementsichtInputTrainerID	Testskript für die Tabellenwertfunktion tf_KurseEinesTrainersManagementsichtInputTrainerID	
	tr_KurseinheitTestDatum	Testskript für den Trigger tr_KurseinheitTestDatum	
	sp_AddTeilnehmerKursIFSportlich	Testskript für die Prozedur sp_AddTeilnehmerKursIFSportlich	

10_BackupDatabase.sql		Skript, um ein Backup (bak-Datei) für eine vorgegebene Datenbank in einem vorgegebenen Pfad mit aktuellem Zeitstempel zu erstellen. Durch den Aufruf dieses Skripts kann z.B. mittels Windows-Aufgabenplanung immer zu einem bestimmten Zeitpunkt automatisch Backups erstellt werden.	Verwendung der Funktionen GETDATE und CONVERT, um einen Zeitstempel zu generieren
10_Backup_Skript.sql		Ein Skript für gesamte Datenbank und alle Objekte über SSMS generiert	