FETCH

```
DECLARE @LastName varchar(50), @FirstName varchar(50);

DECLARE contact_cursor CURSOR FOR
SELECT LastName, FirstName FROM Person
WHERE LastName LIKE 'B%'
ORDER BY LastName, FirstName;

OPEN contact_cursor;
FETCH NEXT FROM contact_cursor
INTO @LastName, @FirstName;
WHILE @@FETCH_STATUS = 0
BEGIN
   PRINT 'Contact Name: ' + @FirstName + ' ' +  @LastName

   FETCH NEXT FROM contact_cursor
   INTO @LastName, @FirstName;
END

CLOSE contact_cursor;
DEALLOCATE contact_cursor;
GO
```

```
DECLARE @student_pk INTEGER;
DECLARE @modul_104 INTEGER;
DECLARE @modul_239 INTEGER;

--Entweder hier definieren oder direkt im while (falls es immer geändert werden muss)
SET @modul_104 = (SELECT modulPK FROM ietModul WHERE modulname = 'Modul 104 Datenbank');
SET @modul_239 = (SELECT modulPK FROM ietModul WHERE modulname = 'Modul 239 Internetserver in Betrieb nehmen');

DECLARE student_cursor CURSOR FOR
         SELECT studentPK FROM gibbStudents

OPEN student_cursor;
FETCH NEXT FROM student_cursor
         INTO @student_pk;

WHILE @@FETCH_STATUS = 0
BEGIN
         INSERT INTO visits_modul (studentFK,modulFK)
                  VALUES (@student_pk, @modul_104)

         INSERT INTO visits_modul (studentFK,modulFK)
                  VALUES (@student_pk,@modul_239)
      FETCH NEXT FROM student_cursor INTO @student_pk;
END
CLOSE student_cursor;
DEALLOCATE student_cursor;
```

```
DECLARE @ArtNr INTEGER;
DECLARE @Bild NVARCHAR(50);
DECLARE @BezeichnungDeutsch NVARCHAR(50);
DECLARE @BezeichnungFranzoesisch NVARCHAR(50);
DECLARE @BezeichnungEnglisch NVARCHAR(50);
DECLARE @PreisEuro decimal(10,2);
DECLARE @PreisCHF decimal(10,2);
DECLARE Art_cursor CURSOR FOR
         SELECT ArtNr FROM Artikel_Alt
OPEN Art_cursor;

FETCH NEXT FROM Art_cursor
         INTO @ArtNr;
WHILE @@FETCH_STATUS = 0
BEGIN

SET @Bild = (SELECT Bild FROM Artikel_Alt WHERE ArtNr = @ArtNr);
SET @BezeichnungDeutsch = (SELECT Name_D FROM Artikel_Alt WHERE ArtNr = @ArtNr);
SET @BezeichnungFranzoesisch = (SELECT Name_F FROM Artikel_Alt WHERE ArtNr = @ArtNr);
SET @BezeichnungEnglisch = (SELECT Name_E FROM Artikel_Alt WHERE ArtNr = @ArtNr);
SET @PreisEuro = (SELECT Preis FROM Artikel_Alt WHERE ArtNr = @ArtNr);
SET @PreisCHF = (SELECT Preis_1 FROM Artikel_Alt WHERE ArtNr = @ArtNr);

         INSERT INTO Artikel_Neu(ArtNr,Bild)
                  VALUES (@ArtNr, @Bild);
         INSERT INTO Preis_Neu(ArtNr,WaehrungsCode,Preis)
                  VALUES (@ArtNr,'EUR',@PreisEuro);
         INSERT INTO Preis_Neu(ArtNr,WaehrungsCode,Preis)
                  VALUES (@ArtNr,'CHF',@PreisCHF);
         INSERT INTO Bezeichnung_Neu(ArtNr,SprachCode,Bezeichnung)
                  VALUES (@ArtNr,'D',@BezeichnungDeutsch);
         INSERT INTO Bezeichnung_Neu(ArtNr,SprachCode,Bezeichnung)
                  VALUES (@ArtNr,'F',@BezeichnungFranzoesisch);
         INSERT INTO Bezeichnung_Neu(ArtNr,SprachCode,Bezeichnung)
                  VALUES (@ArtNr,'E',@BezeichnungEnglisch);
      FETCH NEXT FROM Art_cursor INTO @ArtNr;
END
CLOSE Art_cursor;
DEALLOCATE Art_cursor;

SELECT * FROM Artikel_Neu;
SELECT * FROM Preis_Neu;
SELECT * FROM Bezeichnung_Neu;
SELECT * FROM Artikel_Neu AS A
JOIN Preis_Neu AS P ON A.ArtNr=P.ArtNr
JOIN Bezeichnung_Neu AS B ON A.ArtNr=B.ArtNr;
```

(Im while kann man auch z.B. ein Update Machen etc)

CONSTRAINT

```
CREATE TABLE AUTOR (
        AUTORID bigint IDENTITY(1,1) NOT NULL,
        AdressID bigint NULL,
        Name nchar(100) NULL,
        Vorname nchar(100) NULL,
        Anrede nchar(100) NULL,
        Geburtsdatum nvarchar(50) NULL,
CONSTRAINT PK_AUTOR PRIMARY KEY CLUSTERED,
CONSTRAINT FK_AUTOR_ADRESSE FOREIGN KEY(AdressID)
REFERENCES ADRESSE (AdressID) ON UPDATE CASCADE; // Welches löscht es bei on update cascade? Dort wo on steht?
```

Referenzielle Integritätsbedingungen (RIB)

Aktion	Beschreibung
ON DELETE/UPDATE CASCADE	Ein DELETE/UPDATE in der Primärtabelle führt auch zu einem Löschen/ändern der entsprechenden Sätze in der Fremdschlüsseltabelle.
ON DELETE/UPDATE RESTRICT	Ein DELETE/UPDATE in der Primärtabelle kann nur ausgeführt werden, wenn in keiner Fremdschlüsseltabelle mehr ein Satz mit dem entsprechenden Wert existiert. Dies ist das Defaultverhalten.
ON DELETE/UPDATE SET NULL	Die Spalten des Fremdschlüssels in der Fremdschlüsseltabelle werden beim Löschen/ändern in der Primärtabelle auf NULL gesetzt.
ON DELETE/UPDATE SET DEFAULT	Die Spalten des Fremdschlüssels in der Fremdschlüsseltabelle werden beim Löschen/ändern in der Primärtabelle auf ihren Defaultwert gesetzt. Dazu muss für jede Spalte des Fremdschlüssels ein Defaultwert definiert worden sein. Ist kein Defaultwert definiert, so werden die betreffenden Spalten des Fremdschlüssels auf NULL gesetzt.

Diese funktionieren nur bei Datenbanken mit „Logging“ wegen dem ausführen eines Rollbacks, welches beim Scheitern einer DELETE, INSERT oder UPDATE Operation...

DCL (GRANT SELECT, UPDATE ON BUCH TO VMuser, REVOKE SELECT, INSERT ON BUCH FROM VMuser, GRANT SELECT, UPDATE ON BUCH TO VMRole)

Begriffe:	BERECHTIGUNGEN: `SELECT, UPDATE, INSERT, DELETE, ALL, (ALTER TABLE, INDEX, REFERENCES)` BENUTZERNAME: `PUBLIC, (SELBSTERSTELLTE)`

GRANT	`1)GRANT BERECHTIGUNGEN ON TABELLENNAME TO BENUTZERNAME`
	`2)GRANT UPDATE (ATTRIBUTE(,)),INSERT (ATTRIBUTE(,))`
	`ON TABELLENNAME to BENUTZERNAME`
	`GRANT BERECHTIGUNGEN ON TABELLENNAME TO BENUTZERNAME WITH GRANT OPTION`
	Der Beunutzer darf auch anderen Berechtigungen auf diese Tabelle geben

REVOKE	`1)REVOKE BERECHTIGUNGEN ON TABELLENNAME FROM BENUTZERNAME` `2)REVOKE BERECHTIGUNGEN ON TABELLENNAME FROM BENUTZERNAME`

ROLE	`CREATE ROLE ROLLENNAME`
	`GRANT BERECHTIGUNGEN ON TABELLENNAME TO ROLLENNAME`
	`GRANT ROLLENNAME TO BENUTZERNAME, BENUTZERNAME`
	`DROP ROLE ROLLENNAME`

ALLGEMEIN

Befehl	Beschreibung

`BACKUP DATABASE DBName`
`TO DISK = 'path'`
`RESTORE DATABASE DBName`
`FROM DISK = 'path'`
	Backup erstellen
	
	Backup wiederherstellen

`-- Beziehungen BUCH --> BUCH_STICHWORT`
`ALTER TABLE BUCH_STICHWORT ADD CONSTRAINT FK_BUCH_STICHWORT_BUCH`
`FOREIGN KEY(BuchID)`
`REFERENCES BUCH (BuchID)`
	Foreign Key nachträglich einfügen

`ALTER TABLE BUCH_STICHWORT DROP CONSTRAINT` `FK_BUCH_STICHWORT_BUCH`	Foreign Key löschen / Entsprechenden Constraint Name brauchen
`ALTER TABLE BUCH_STICHWORT ADD COLUMN STICHWORT NVARCHAR(50);`	Spalte hinzufügen
`ALTER TABLE BUCH_STICHWORT DROP COLUMN STICHWORT;`	

`CREATE INDEX "INDEX_NAME" ON "TABELLEN_NAME" (SPALTEN_NAME)`
`CREATE INDEX IDX_ADRESSE ON ADRESSE (Strasse, PLZ, Land)`
`DROP INDEX INDEX_NAME ON TABELLEN_NAME`
	Aufbau Index
	Beispiel Index
	Löschen Index

SELECT [AGGREGATION()] Attributname1 [AS „Alias“], Attributname2,… FROM TABELLENNAME [WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert] [AND, OR, NOT] [ORDER BY Attributname1 [DESC, ASC]] [GROUP BY Attributname1] [HAVING COUNT(*) > 2] [LIMIT N];	[Aggregation wie MIN/MAX/DISTINCT],[Ein Alias setzen] Eine Einschränkung setzen z.B. IS NULL, NOT IN [DESC=Absteigend ZYX ASC=Aufsteigend ABC] [Gruppieren] [Wenn eine Aggregatsfunktion gebraucht wird] [Eine Limite setzen]
INSERT INTO TABELLENNAME (Attributname1, Attributname2,…) VALUES (‘Datenwert1‘,‘Datenwert2‘,…) ,(), (),…;	Eine Zeile hinzufügen
UPDATE TABELLENNAME SET Attributname1 = Datenwert1 AttributnameN = DatenwertN WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert [AND, OR, NOT]	Einen Datenwert aktualisieren
DELETE FROM TABELLENNAME WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert [AND, OR, NOT]	Eine Zeile löschen
TRUNCATE TABLE TABELLENNAME;	Löschen ohne Rollback
SELECT AttributnameN FROM TabelleL [INNER / LEFT / RIGHT JOIN / FULL OUTER] TabelleR ON AttrL=AttrR;	Eine weitere Tabelle hinzufügen
SELECT * FROM Table1 UNION SELECT * FROM Table2	Resultate vereinen