MYSQL

Inhaltsverzeichnis

1 Constraint

```
constraint
einzigartigkeit deklarieren
CONSTRAINT c UNIQUE (ID)
mehrfach vermeiden
CONSTRAINT c CHECK (UNIQUE ID)
nur Wert X,Y verwenden
CONSTRAINT c CHECK (value IN("X","Y"))
positiver Wert
CONSTRAINT c CHECK (value > 0)
value2 nach value1
CONSTRAINT c CHECK (value2  > value1)
maximal 5
CONSTRAINT c CHECK (value1 <= 5)
kein Wert der Vergangenh.
CONSTRAINT c CHECK (value1 = now)
wert1 = wert2
CONSTRAINT c CHECK ((SELECT value1 FROM ..)=(SELECT...))
textlnge unter 256
CONSTRAINT c CHECK (LENGTH(txt)<256)
```

2 Create

```
CREATE TABLE chara(
 CID INTEGER PRIMARY KEY,
 name VARCHAR(20) UNIQUE NOT NULL,
 ValueID INTEGER REFERENCES charaValues(VID),
 InventarID INTEGER REFERENCES inventars(IID),
 RaceID INTEGER REFERENCES races(RID)
 ...
);
CREATE TABLE charaValues(
 VID INTEGER PRIMARY KEY,
 level INTEGER
 ...
);
CREATE TABLE inventars(
 IID INTEGER PRIMARY KEY,
 ItemID INTEGER REFERENCES items(ItemID),
 item_power INTEGER
 ...
);
 CREATE TABLE races(
 RID INTEGER PRIMARY KEY,
 name VARCHAR(20) UNIQUE NOT NULL
 ...
);
CREATE TABLE shops(
 ShopID INTEGER PRIMARY KEY,
 name VARCHAR(20) UNIQUE NOT NULL,
 ItemID INTEGER REFERENCES items(ItemID)
 ...
);
CREATE TABLE items(
 ItemID INTEGER PRIMARY KEY,
 name VARCHAR(20) UNIQUE NOT NULL
 ...
);
CREATE TABLE battle(BattleID INTEGER PRIMARY KEY,
 charaID1 INTEGER,
 charaID2 INTEGER
 ...
);
CREATE TABLE battle_msg(
 MsgIG INTEGER PRIMRAY KEY,
 text VARCHAR(255)
 ...
);
CREATE TABLE buy(
 BID INTEGER PRIMARY KEY,
 ShopID INTEGER REFERENCES shops(ShopID),
 ItemID INTEGER REFERENCES items(ItemID)
 ...
);
CREATE TABLE users(
 UID INTEGER PRIMARY KEY,
 name VARCHAR(20) UNIQUE NOT NULL,
 register DATE
 ...
);
CREATE TABLE topics(
 TID INTEGER PRIMARY KEY,
```

```
 UID INTEGER REFERENCES users(UID),
 title VARCHAR(50)
 ...
);
CREATE TABLE posts(
 PID INTEGER PRIMARY KEY,
 TID INTEGER REFERENCES topics(TID),
 UID INTEGER REFERENCES users(UID),
 text VARCHAR(255)
 ...
);
CREATE TABLE asses(
AID INTEGER PRIMARY KEY,
TID INTEGER REFERENCES topics(TID),
UID INTEGER REFERENCES users(UID),
rand INTEGER,
text VARCHAR(255)
 ...
);

CREATE TABLE students(
StudiID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
NoteID INTEGER REFERENCES notes(NoteID),
Matr.Nr. INTEGER
 ...
);
CREATE TABLE subjects(
SubjectID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
TeacherID INTEGER REFERENCES teachers,
room INTEGER
 ...
);
CREATE TABLE teachers(
TeacherID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
SubjectID INTEGER REFERENCES subjects(SubjectID)
 ...
);
CREATE TABLE notes(
NoteID INTEGER PRIMARY KEY,
StudiID INTEGER REFERENCES students(StudiID),
SubjectID INTEGER REFERENCES subjects(SubjectID)
 ...
);
CREATE TABLE workbench(
WID INTEGER PRIMARY KEY,
SID INTEGER REFERENCES servers(SID),
DID INTEGER REFERENCES databases(DID),
SSID INTEGER REFERENCES sql_script(SSID),
installed DATE
 ...
);
CREATE TABLE servers(
SID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
hostname VARCHAR(20),
port INTEGER,
username VARCHAR(20) UNIQUE NOT NULL
 ...
```

```
);
CREATE TABLE databases(
DID INTEGER PRIMARY KEY,
name VARCHAR(20)
 ...
);
CREATE TABLE sql_script(
SSID INTEGER PRIMARY KEY,
title VACHAR(20),
source_code VARCHAR(255),
DID INTEGER REFERENCES databases(DID)
 ...
);
CREATE TABLE stor._Procedure(
SPID INTEGER PRIMARY KEY,
source_code VARCHAR(255),
SSID INTEGER REFERENCES sql_script(SSID)
 ...
);
CREATE TABLE users(
UID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL
 ...
);
CREATE TABLE tutorial(
TID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
UID INTEGER REFERENCES users(UID),
KID INTEGER REFERENCES kategorie(KID)
 ...
);
CREATE TABLE kategorie(
KID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
description VARCHAR(100)
 ...
);
CREATE TABLE attributs(
ATID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
KID INTEGER REFERENCES kategorie(KID),
PDF BOOL,
txt BOOL,
power_point BOOL,
swf BOOL,
mp3 BOOL
 ...
);
CREATE TABLE asses(
ASID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
TID INTEGER  REFERENCES tutorials(TID),
UID INTEGER REFERENCES users(UID)
 ...
);

CREATE TABLE bauer(
BID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
IID INTEGER REFRENCES inventar(IID),
LandID INTEGER REFERENCES land(LandID)
```

```
 ...
);
CREATE TABLE inventar(
IID INTEGER PRIMARY KEY,
GID INTEGER REFERENCES gegenstand(GID),
duration INTEGER
 ...
);
CREATE TABLE land(
LandID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL
 ...
);
CREATE TABLE rohstoff(
RID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
LandID INTEGER REFERENCES land(LandID)
 ...
);
CREATE TABLE laden(
LID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
GID INTEGER REFERENCES gegenstand(GID)
 ...
);
CREATE TABLE gegenstand(
GID INTEGER PRIMARY KEY,
name VARCHAR(20) UNIQUE NOT NULL,
price INTEGER, duration INTEGER
 ...
);
```

3 1 Fall

3.1 CREATE TABLE

```
CREATE TABLE Lehrveranstaltung(
LVID  CHAR(6) PRIMARY KEY,
Fachbereich  CHAR(20)
 ...
);
CREATE TABLE Studenten(
SID CHAR(9) PRIMARY KEY,
Notenschnitt FLOAT
 ...
);
CREATE TABLE Kurswahl(
 SID CHAR(9) REFERENCES Studenten(SID),
 LVID CHAR(6)
 ...
);
CONSTRAINT LVID_FK-Check REFERENCES Lehrverstaltung(LVID));
CREATE TABLE Studi(
 id INTEGER PRIMARY KEY,
 Studienrichtung VARCHAR,
 CONSTRAINT StRichCheck
 CHECK(Studienrichtung IN ('Allgemeine', 'Wirtschafts, 'Informatik'')),
 Matrikel VARCHAR(20) UNIQUE
 ...
);
CREATE TABLE StudiList(
 ID INTEGER PRIMARY KEY,
 Vorname VARCHAR(255) DEFAULT  'unknow',
 Nachname VARCHAR(255) NOT NULL
 ...
);
CREATE TABLE StudiDresden(
 ID integer primary key,
 name vachar,
 MatrNum integer,
 Abschlussnote decimal(6,2),
 Studiengang varchar,
 constraint CHECK(IN('englisch','mathematik',''informatik'))
 ...
);
```

3.2 INSERT INTO

```
INSERT INTO Studi VALUES(1,'info',222345);
INSERT INTO Studi VALUES(2,'info',222346);
INSERT INTO Studi VALUES(3,'info',222347);
INSERT INTO Studi VALUES(4,'kunst',222348);
INSERT INTO StudiList(ID,Nachname)VALUES(1,'smith');
INSERT INTO StudiList(ID,Nachname)VALUES(2,'johnson');
INSERT INTO StudiList(ID,Nachname)VALUES(3,'peters');
INSERT INTO StudiDresden(ID,name,MatrNum,Abschlussnote,Studiengang)
 VALUES(1,'peter',223456,2.23,'englisch');
INSERT INTO StudiDresden(ID,name,MatrNum,Abschlussnote,Studiengang)
 VALUES(2,'paul',223457,2.17,'informatik');
INSERT INTO StudiDresden(ID,name,MatrNum,Abschlussnote,Studiengang)
 VALUES(3,'otto',223458,2.07,'mathematik'');
```

3.3 SELECT FROM

```
SELECT * FROM Studi ORDER BY name;
SELECT * FROM Studi ORDER BY name DESC;
SELECT * FROM StudiList WHERE ID = 1;
SELECT * FROM Studis WHERE Studiengang IN ('Info', 'E-Technik');
SELECT MatrikelNummer FROM Studi
 WHERE AbschlussNote > 2.0 GROUP BY Studiengang;
SELECT MatrNum, name, Abschlussnote FROM StudiDresden
 WHERE Studiengang = 'Informatik' UNION
  SELECT MatrNum, name, Abschlussnote FROM StudiGoerlitz
   WHERE Studiengang = 'Informatik' ORDER BY AbschlussNote;
SELECT name FROM StudiDresden
 WHERE Studiengang = 'Informatik' UNION
SELECT name FROM StudiGoerlitz
 WHERE Studiengang = 'Informatik'
 ORDER BY AbschlussNote;
SELECT name FROM StudiDresden
 WHERE Studiengang = 'Informatik' UNION
ALL SELECT name FROM StudiZittau
 WHERE Studiengang = 'Informatik'
 ORDER BY AbschlussNote;
```

3.4 UPDATE SET

```
 UPDATE Studi AS S  SET S.Alter = S.Alter + 2 WHERE S.Name = Mueller;
```

3.5 DELETE FROM

```
DELETE FROM Studi WHERE (Start< 1998) AND NOT Studiengang = 'Informatik';
DELETE FROM Studi WHERE FirstName LIKE 'M%';
DELETE FROM Studi WHERE LastName LIKE '%son';
DELETE FROM Studi WHERE LastName LIKE '___ %';
```

4 2 Fall

4.1 CREATE TABLE

```
CREATE TABLE Person(
 id INTEGER(9) PRIMARY KEY,
 LastName VARCHAR(255),
 FirstName VARCHAR(255)
 ...
);
CREATE INDEX PersonIndex ON Person(LastName);
CREATE INDEX NameIndex ON Person(LastName, FirstName);
```

4.2 INSERT INTO

```
INSERT INTO Person(id,LastName,FirstName)
 VALUES(1,"timmy","turner");
INSERT INTO Person(id,LastName,FirstName)
 VALUES(2,"","cosmo");
INSERT INTO Person(id,LastName,FirstName)
 VALUES(3,"","wanda");
```

4.3 SELECT FROM

```
SELECT * FROM Persons;
SELECT LastName, FirstName FROM Persons;
SELECT LastName FROM Persons WHERE age >= 18;
SELECT FirstName, LastName FROM Studi
 WHERE Start = 2000
 AND Studiengang IN('Info', 'ET', 'M.Bau');
SELECT COUNT(*) FROM Studi  WHERE Start = 1999;
SELECT Abschlussnote FROM Studi
 WHERE Start = 2000 AND Studiengang = 'Informatik';
SELECT LastName AS Family, FirstName AS Name
 WHERE Family LIKE '%son' FROM Persons;
SELECT MAX(age) FROM Persons;
SELECT MIN(Preis) FROM Autos WHERE Typ = 'Cabrio';
SELECT name FROM Persons;
SELECT ALL name FROM Persons;
SELECT DISTINCT name FROM Persons;
SELECT COUNT(name) FROM Persons;
SELECT COUNT(DISTINCT name) FROM Persons;
SELECT ALL COUNT(name) FROM Persons
 WHERE name = 'Meier';
SELECT DISTINCT COUNT(name) FROM Persons;
SELECT FirstName, LastName FROM Persons
 GROUP BY LastName;
```

4.4 DELETE FROM

```
DELETE FROM Persons
 WHERE FirstName = 'Sven' AND LastName = 'Svenson';
```

5 3 Fall

5.1 CREATE TABLE

```
CREATE TABLE person(
 PID INTEGER(9) PRIMARY KEY AUTO_INCREMENT UNIQUE NOT NULL,
 name VARCHAR(255),
 firstname VARCHAR(255),
 city VARCHAR(255) DEFAULT 'unknown'
 ...
);
CREATE TABLE purchase(
 purchaseID INTEGER(9) PRIMARY KEY NOT NULL,
 buyerID INTEGER(9) REFERENCES person(PID),
 buyer VARCHAR(255) REFERENCES person(name_),
 store VARCHAR(255),
 price INTEGER(9),
 date_ VARCHAR(255),
 product INTEGER(9) REFERENCES products(ID)
 ...
);
CREATE TABLE products(
 ID INTEGER(3) PRIMARY KEY,
 name_ VARCHAR(255),
 kategorie VARCHAR(255)
 ...
);
```

5.2 INSERT INTO

```
INSERT INTO person VALUES('pan','peter','berlin');
INSERT INTO person VALUES('dieter','klaus','kln');
INSERT INTO person VALUES('hoffmann','sven','hamburg');
INSERT INTO person VALUES('hood','robin','berlin');
INSERT INTO person VALUES('jrgens','hans','berlin');
INSERT INTO person VALUES('hollnder','fliegender','kln');
INSERT INTO person VALUES('moto','marsi','berlin');
INSERT INTO person VALUES('pan2','peter','berlin');
INSERT INTO person VALUES('dieter2','klaus','kln');
INSERT INTO person VALUES('hoffmann2','sven','hamburg');
INSERT INTO person VALUES('hood2','robin','berlin');
INSERT INTO person VALUES('jrgens2','hans','berlin');
INSERT INTO person VALUES('hollnder2','fliegender','kln');
INSERT INTO person VALUES('moto2','marsi','berlin');
INSERT INTO purchase VALUES (1,5,'jrgens','berlin',999,'13.10.2012',4);
INSERT INTO purchase VALUES (2,1,'pan','berlin',999,'17.10.2012',1);
INSERT INTO purchase VALUES (3,7,'moto','berlin',999,'20.10.2012',9);
INSERT INTO purchase VALUES (4,3,'hoffmann','berlin',999,'25.10.2012',2);
INSERT INTO purchase VALUES (5,4,'hood','berlin',999,'30.10.2012',3);
INSERT INTO purchase VALUES (6,4,'hood','berlin',999,'3.11.2012',7);
INSERT INTO purchase VALUES (7,8,'pan2','berlin',999,'10.12.2012',7);
INSERT INTO purchase VALUES (8,2,'dieter','berlin',999,'10.1.2013',7);
INSERT INTO purchase VALUES (9,5,'jrgens','berlin',999,'10.1.2013',9);
INSERT INTO purchase VALUES (10,1,'pan','berlin',999,'10.1.2013',2);
INSERT INTO purchase VALUES (11,7,'moto','berlin',999,'13.2.2013',6);
INSERT INTO purchase VALUES (12,3,'hoffmann','berlin',999,'3.3.2013',2);
INSERT INTO purchase VALUES (13,4,'hood','berlin',999,'17.3.2013',1);
INSERT INTO purchase VALUES (14,4,'hood','berlin',999,'25.3.2013',1);
INSERT INTO purchase VALUES (15,8,'pan2','berlin',999,'30.3.2013',8);
INSERT INTO purchase VALUES (16,2,'dieter','berlin',999,'10.4.2013',9);
INSERT INTO products(name)SELECT DISTINCT purchase.product
```

```
FROM purchase WHERE purchase.date > 10/26/01;
INSERT INTO products(name) SELECT DISTINCT prodName
 FROM purchase WHERE prodName NOT IN (SELECT name FROM products);
INSERT INTO products VALUES (1, 'kfer', 'auto');
INSERT INTO products VALUES (2, 'dell8200', 'computer');
INSERT INTO products VALUES (3, 'lupo', 'auto');
INSERT INTO products VALUES (4, 'hp', 'drucker');
INSERT INTO products VALUES (5, 'compact', 'computer');
INSERT INTO products VALUES (6, 'belina', 'TV');
INSERT INTO products VALUES (7, 'sony', 'TV');
INSERT INTO products VALUES (8, 'VW', 'auto');
INSERT INTO products VALUES (9, 'lexmark', 'drucker');
```

5.3 SELECT FROM

```
SELECT name_ FROM person WHERE city = 'kln' AND city = 'hamburg'
 UNION (SELECT PE.name_ FROM person AS PE, PURCHASE AS PU
  WHERE PU.buyerID=PE.PID AND PU.store = 'berlin');
SELECT * FROM Person WHERE (age < 25)
 AND (height>6 OR weight>190) E.g. age=20 height=NULL weight=160
SELECT product, Sum(price) AS TotalSales FROM purchase
 WHERE date > 9/1 GROUP BY product
SELECT product, Sum(price) AS TotalSales FROM purchase
 WHERE date > 9/1 GROUP BY product
SELECT product, Sum(price) AS SumSales Max(quantity) AS MaxQuantity
 FROM SELECT product, Sum(price) FROM purchase
  WHERE date>9/1  GROUP BY product HAVING Sum(quantity)>30
SELECT products.name, purchase.store FROM products
 JOIN purchase ON  products.name = purchase.prodName
SELECT products.name, purchase.store FROM products, Purchase
 WHERE products.name = purchase.prodName
SELECT products.name, purchase.store FROM products LEFT OUTER
 JOIN purchase ON products.name = purchase.prodName
```

5.4 UPDATE SET

```
UPDATE products SET price = price/2 WHERE  Product.name IN
 (SELECT product FROM purchase WHERE  Date =Oct, 25, 1999);
```

5.5 DELETE FROM

```
DELETE FROM purchase
   WHERE selle =Joe AND produc =Brooklyn Bridge
```

5.6 SELECT FROM

```
SELECT kategorie, count(kategorie) from products
 group by kategorie having count(kategorie)>1;
```

6 4 Fall

6.1 CREATE TABLE

```
CREATE TABLE Autos(
 id INTEGER(9)PRIMARY KEY,
 typ VARCHAR(255),
 price INTEGER(9)
 ...
);
CREATE TABLE Autos2(
 typ VARCHAR(255) PRIMARY KEY,
 price VARCHAR(255)
 );
CREATE TABLE Autos3(
 AID INTEGER PRIMARY KEY AUTO_INCREMENT,
 Name VARCHAR(255)
 ...
);
CREATE TABLE Autos3(
 AutoID INTEGER DEFAULT nextval('Autos_AutoID_seq'
 ...
);
CREATE TABLE Students(
 sid integer(9) primary key,
 name_ varchar(255),
 login varchar(255),
 age integer(3)
 ...
);
CREATE TABLE HasCar(
 ownerId integer(9) primary key,
 carName varchar(255)
 ...
);
CREATE SEQUENCE Autos2AutoID_seq  START WITH 10  INCREMENT BY 10;
CREATE UNIQUE INDEX Autos3_AutoID_key on Autos3(AutoID);
```

6.2 INSERT INTO

```
INSERT INTO Autos VALUES(234, 'dfe',53453);
INSERT INTO Autos2(Name, Preis) VALUES('Kaefer', 1500);
INSERT INTO Students VALUES(1,'a1','logxxx',40);
INSERT INTO Students VALUES(2,'a2','logxxx',29);
INSERT INTO Students VALUES(3,'a3','logxxx',34);
INSERT INTO Students VALUES(4,'a4','logxxx',43);
INSERT INTO HasCar VALUES(1,'vw');
INSERT INTO HasCar VALUES(3,'fiat');
```

6.3 SELECT FROM

```
SELECT carName FROM HasCar WHERE ownerId =
 (SELECT sid FROM students WHERE login = 'logxxx');
SELECT carName FROM HasCar WHERE ownerID =
 (SELECT sid FROM Students WHERE login = Mark);
SELECT carName FROM Students, HasCar WHERE Students.sid =
 HasCar.ownerID AND Students.login = Mark;
```

6.4 DELETE FROM

```
DELETE FROM Autos WHERE(Verfuegbar = 0 AND Preis > 1000);
```

7 5 Fall

7.1 CREATE TABLE

```
CREATE TABLE greetings(
 i INTEGER GENERATED ALWAYS AS IDENTITY,
 ch CHAR(50)
 ...
);
```

7.2 INSERT INTO

```
INSERT INTO greetings VALUES (DEFAULT, 'hello');
INSERT INTO greetings(ch) VALUES ('bonjour');
```

7.3 SELECT FROM

```
SELECT SUM(Betrag) FROM Bestellungen
 WHERE BestellMonat
 BETWEEN '2009-01-01' AND '2009-03-31';
SELECT SUM(Menge) FROM LagerTabelle
 WHERE ArtikelKategorie = 'Monitor';
SELECT MAX(age) FROM Persons;
SELECT MIN(Preis) FROM Autos WHERE Typ = 'Cabrio';
SELECT name FROM Persons;
SELECT ALL name FROM Persons;
SELECT DISTINCT name FROM Persons;
SELECT COUNT(name) FROM Persons;
SELECT COUNT(DISTINCT name) FROM Persons;
SELECT ALL COUNT(name) FROM Persons
 WHERE name = 'Meier';
SELECT DISTINCT COUNT(name) FROM Persons;
SELECT * FROM Students ORDER BY name;
SELECT * FROM Students ORDER BY name DESC;
SELECT FirstName, LastName FROM Persons
 GROUP BY LastName;
SELECT MatrikelNummer FROM Studi
 WHERE AbschlussNote > 2.0
  GROUP BY Studiengang;
```

7.4 UPDATE SET

```
UPDATE Bilanz AS B SET B.Guthaben =
 B.Guthaben  (SELECT SUM(Ueberweisung)
 F ROM Bestellungen AS Best WHERE B.Name = Best.Besteller
);
```

8 6 Fall

8.1 CREATE TABLE

```
CREATE TABLE Author(
 ID integer primary key,
 name varchar,
 login varchar
 ...
);
CREATE TABLE Wrote(
 ID integer primary key,
 url varchar,
 login varchar
 ...
);
CREATE TABLE Mention(
 ID integer primary key,
 url varchar
 ...
);
```

8.2 INSERT INTO

```
INSERT INTO VALUES Author(1,'author1','login');
INSERT INTO VALUES Wrote(1,'url1','login');
INSERT INTO VALUES Mention(1,'url1');
```

8.3 SELECT FROM

```
SELECT DISTINCT Author.name FROM Author
 WHERE count (SELECT Wrote.url
 FROM Wrote WHERE Author.login=Wrote.login) > 10;
SELECT Author.name FROM Author, Wrote
 WHERE Author.login=Wrote.login;
 GROUP BY Author.name HAVING count(wrote.url) > 10
SELECT Author.name FROM Author, Wrote, Mentions
 WHERE Author.login=Wrote.login AND Wrote.url=Mentions.url
 GROUP BY Author.name
 HAVING count(distinct Mentions.word) > 10000;
```

9 Select++

```
SELECT SUM(Betrag) FROM Bestellungen WHERE BestellMonat
 BETWEEN '2009-01-01' AND '2009-03-31';
SELECT SUM(Menge) FROM LagerTabelle WHERE ArtikelKategorie = 'Monitor';
SELECT MAX(age) FROM Persons;
SELECT MIN(Preis) FROM Autos WHERE Typ = 'Cabrio';
SELECT name FROM Persons;
SELECT ALL name FROM Persons;
SELECT DISTINCT name FROM Persons;
SELECT COUNT(name) FROM Persons;
SELECT COUNT(DISTINCT name) FROM Persons;
SELECT ALL COUNT(name) FROM Persons WHERE name = 'Meier';
SELECT DISTINCT COUNT(name) FROM Persons;
SELECT * FROM Students ORDER BY name;
SELECT * FROM Students ORDER BY name DESC;
SELECT FirstName, LastName FROM Persons GROUP BY LastName;
SELECT MatrikelNummer FROM Studi WHERE AbschlussNote > 2.0 GROUP BY Studiengang;
```