Fachbereich 07 Informatik/Mathematik



Praktikum Datenbanksysteme II

Wintersemester 2018/19

Prof. Dr. Martin Staudt

Übung 2

Wimmer, Anja

IF8

Gabl, Daniel

IF6

06.12.2018

# Inhaltsverzeichnis

[Inhaltsverzeichnis](#_Inhaltsverzeichnis) II

[Aufgaben](#_Aufgaben) 1

[Aufgabe 1](#_Aufgabe_1) 1

[Aufgabe 2](#_Aufgabe_2) 1

[Aufgabe 3](#_Aufgabe_3) 1

[Aufgabe 4](#_Aufgabe_4) 4

[Aufgabe 5 (Fallstudie)](#_Aufgabe_5_(Fallstudie)) 6

[Screendumps der Tabellen](#_Screendumps_der_Tabellen) 16

[Anmerkungen](#_Anmerkungen) 18

# Aufgaben

## Aufgabe 1

-

## Aufgabe 2

Die zwei Möglichkeiten, die wir in Betracht ziehen:

1. Allmögliche Objekt-Typen definieren und gegenseitig referenzieren. Bspw. könnte die Adresse ein Typ sein, die sich aus der Straße und der Hausnummer (und ggf. PLZ und Ort) zusammensetzt. Auch könnte ein Kontotyp mit Konto-Nr., Kontostand, Art und ID der Zweigstelle ein eigenes Attribut sein. (Es ist keine Zuordnungstabelle erforderlich, da es sich bei Zweigstelle ↔ Konto um eine 1:n-Beziehung handelt.)
2. Beispielsweiße könnten wir den Konto-Typ nicht als eigene Tabelle speichern sondern als innere Tabelle beim Zweigstellen-Typs speichern, dadurch entfällt die Referenz auf diese Tabelle.

## Aufgabe 3

Wir würden folgendes Schema aufstellen: (Legende: 1. Möglichkeit, **2. Möglichkeit**, beide)

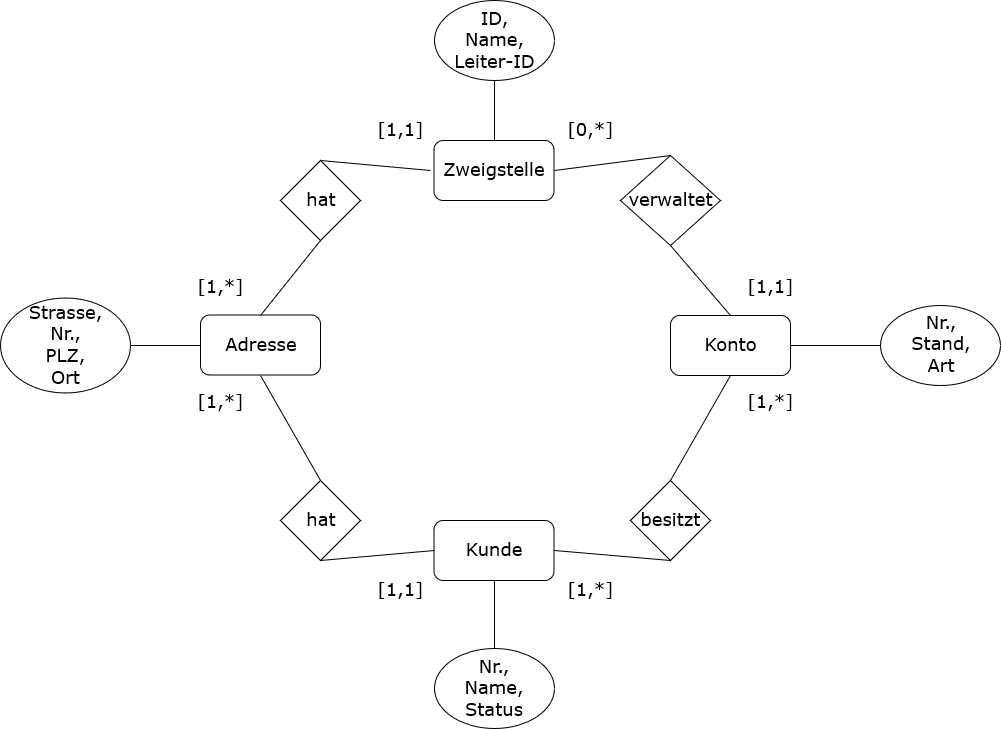
5 Typen wie folgt:

* Adress-Typ mit Straße und Hausnummer (und ggf. PLZ und Ort)
* **Kontolisten-Typ als Tabelle vom Typ Integer (Kontonummern)**
* Kunden-Typ mit Kunden-Nr., -Name, Adress-Typ, Status **und Kontolisten-Typ**
* Kontoinhaber-Typ als Tabelle vom Kunden-Typ
* **Zweigstellenkonten-Typ als Tabelle von Konto-Typen**
* Zweigstellen-Typ mit Zweigstellenname, Adress-Typ, Leiter-Id **und Zweigkonten**
* Konto-Typ mit Konto-Nr., Kontostand, Art, Kontoinhaber-Typ und Zweigstellen-Typ

Dazu noch folgende Tabellen:

* Kunden-Tabelle mit Kunden-Typ
* Zweigstellen-Adresse mit Zweigstellen-Typ
* Konto-Tabelle mit Konto-Typ **(entfällt bei der 2. Möglichkeit)**

Eine Skizzierung des Zusammenspiels der Typen und Tabellen:



SQL-Statements zum Erzeugen der Typen und Tabellen **(1. Möglichkeit)**:  
CREATE TYPE AddressType AS Object(street VARCHAR(31), houseNr VARCHAR(7), zip INT(5), place VARCHAR(31));

/

CREATE TYPE CustomerType AS Object(customerNr INT, customerName VARCHAR(63), addr AddressType, status VARCHAR(15));

/

CREATE TYPE AccountOwnerType AS TABLE OF REF CustomerType;

/

CREATE TYPE BranchOfficeType AS Object(branchOfficeName VARCHAR(63), addr AddressType, leaderId INT);

/

CREATE TYPE AccountType AS Object(accountNr INT,

balance DOUBLE PRECISION, kind VARCHAR(1),

owners AccountOwnerType, branchOffice REF BranchOfficeType);

/

CREATE TABLE Customer OF CustomerType;

CREATE TABLE BranchOffice OF BranchOfficeType;

CREATE TABLE AccountTable OF AccountType NESTED TABLE owners STORE AS lorem\_ipsum;

SQL-Statements zum Einfügen von Beispieldatensätzen in die Datenbank **(1. Möglichkeit)**:

INSERT INTO Customer VALUES(CustomerType(2345, 'H. Fach', AddressType('Münchenerstr.', '33', 60329, ' Frankfurt am Main'), 'Geschäftskunde'));

INSERT INTO Customer VALUES(CustomerType(7654, 'B. Meier', AddressType('Eschenweg', '12', 85354, 'Freising'), 'Privatkunde'));

INSERT INTO Customer VALUES(CustomerType(8764, 'J. Wiesner', AddressType('Schellingstr.', '42', 80799, 'München'), 'Geschäftskunde'));

INSERT INTO BranchOffice VALUES(BranchOfficeType('Bachdorf', AddressType('Hochstr.', '1', 81669, 'München'), 1768));

INSERT INTO BranchOffice VALUES(BranchOfficeType('Riedering', AddressType('Simseestr.', '3', 81549, 'München'), 9823));

INSERT INTO AccountTable VALUES(AccountType(120768, 234.56, 'S', AccountOwnerType((SELECT REF(c) FROM Customer c WHERE c.customerNr = 2345)), (SELECT REF(b) FROM BranchOffice b WHERE b.branchOfficeName = 'Bachdorf')));

INSERT INTO AccountTable VALUES(AccountType(678453, -456.78, 'G', AccountOwnerType((SELECT REF(c) FROM Customer c WHERE c.customerNr = 8764)), (SELECT REF(b) FROM BranchOffice b WHERE b.branchOfficeName = 'Bachdorf')));

INSERT INTO AccountTable VALUES(AccountType(348973, 12567.56, 'G', AccountOwnerType((SELECT REF(c) FROM Customer c WHERE c.customerNr = 2345), (SELECT REF(c) FROM Customer c WHERE c.customerNr = 8764)), (SELECT REF(b) FROM BranchOffice b WHERE b.branchOfficeName = 'Bachdorf')));

INSERT INTO AccountTable VALUES(AccountType(987654, 789.65, 'G', AccountOwnerType((SELECT REF(c) FROM Customer c WHERE c.customerNr = 7654)), (SELECT REF(b) FROM BranchOffice b WHERE b.branchOfficeName = 'Riedering')));

INSERT INTO AccountTable VALUES(AccountType(745363, -23.67, 'S', AccountOwnerType((SELECT REF(c) FROM Customer c WHERE c.customerNr = 8764)), (SELECT REF(b) FROM BranchOffice b WHERE b.branchOfficeName = 'Riedering')));

SQL-Statements zum Erzeugen der Typen und Tabellen **(2. Möglichkeit)**:  
CREATE TYPE AddressType AS Object(street VARCHAR(31), houseNr VARCHAR(7), zip INT(5), place VARCHAR(31));

/

CREATE TYPE AccountsT AS TABLE OF INT;

/

CREATE TYPE CustomerType AS Object(customerNr INT, customerName VARCHAR(63), addr AddressType, status VARCHAR(15), accNr AccountsT);

/

CREATE TYPE AccountType AS Object(accountNr INT,

balance DOUBLE PRECISION, kind VARCHAR(1));

/

CREATE TYPE BranchAccountsType AS TABLE OF AccountType;

/

CREATE TYPE BranchOfficeType AS Object(branchOfficeName VARCHAR(63), addr AddressType, leaderId INT, accounts BranchAccountsType);

/

CREATE TABLE Customer OF CustomerType

NESTED TABLE accNr STORE AS accNr\_useless;

CREATE TABLE BranchOffice OF BranchOfficeType

NESTED TABLE accounts STORE AS accounts\_useless;

SQL-Statements zum Einfügen von Beispieldatensätzen in die Datenbank **(2. Möglichkeit)**:

INSERT INTO Customer

VALUES(CustomerType(2345, 'H. Fach',

AddressType('Münchenerstr.', '33', 60329, 'Frankfurt am Main'), 'Geschäftskunde', AccountsT(120768, 348973)));

INSERT INTO Customer

VALUES(CustomerType(7654, 'B. Meier',

AddressType('Eschenweg', '12', 85354, 'Freising'), 'Privatkunde', AccountsT(987654)));

INSERT INTO Customer

VALUES(CustomerType(8764, 'J. Wiesner',

AddressType('Schellingstr.', '42', 80799, 'München'), 'Geschäftskunde', AccountsT(745363, 678453, 348973)));

INSERT INTO BranchOffice

VALUES(BranchOfficeType('Bachdorf', AddressType('Hochstr.', '1', 81669, 'München'), 1768, BranchAccountsType()));

INSERT INTO TABLE(SELECT accounts FROM BranchOffice WHERE branchOfficeName='Bachdorf')

VALUES(AccountType(120768, 234.56, 'S'));

INSERT INTO TABLE(SELECT accounts FROM BranchOffice WHERE branchOfficeName='Bachdorf')

VALUES(AccountType(678453, -456.78, 'G'));

INSERT INTO TABLE(SELECT accounts FROM BranchOffice WHERE branchOfficeName='Bachdorf')

VALUES(AccountType(348973, 12567.56, 'G'));

INSERT INTO BranchOffice

VALUES(BranchOfficeType('Riedering', AddressType('Simseestr.', '3', 81549, 'München'), 9823, BranchAccountsType()));

INSERT INTO TABLE(SELECT accounts FROM BranchOffice WHERE branchOfficeName='Riedering')

VALUES(AccountType(987654, 789.65, 'G'));

INSERT INTO TABLE(SELECT accounts FROM BranchOffice WHERE branchOfficeName='Riedering')

VALUES(AccountType(745363, -23.67, 'S'));

## Aufgabe 4

Bei der 1. Möglichkeit:

1. SELECT a.accountNr, a.balance, a.kind, CONCAT(CONCAT(DEREF(a.branchOffice).addr.street, ' '), DEREF(a.branchOffice).addr.houseNr) AS addr FROM AccountTable a;
2. SELECT a.accountNr, DEREF(o.COLUMN\_VALUE).customerName AS customerName, CONCAT(CONCAT(DEREF(o.COLUMN\_VALUE).addr.street, ' '), DEREF(o.COLUMN\_VALUE).addr.houseNr) as addr FROM AccountTable a, TABLE(a.owners) o;

Bei der 2. Möglichkeit:

1. SELECT a.accountNr, a.balance, a.kind,

CONCAT(CONCAT(b.addr.street, ' '), b.addr.houseNr) AS addr

FROM BranchOffice b, TABLE(b.accounts) a;

1. SELECT a.COLUMN\_VALUE,

CONCAT(CONCAT(c.addr.street, ' '),c.addr.houseNr) AS addr

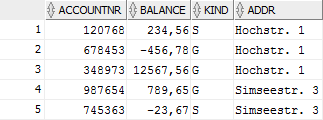
FROM Customer c, TABLE(c.accNr) a;

**Screendumps**

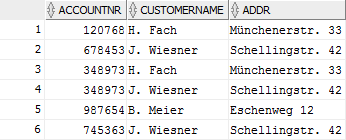
Hier Screendumps unserer Tabellen und den Ergebnissen aus Aufgabe 4.

**1. Möglichkeit:**

Screendump von Aufgabe 4a) (Kontonummer, -stand, -art und Adresse der Zweigstelle):



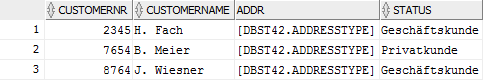
Screendump von Aufgabe 4b) (Paare von Kontonummern, Namen und Adressen der Inhaber):



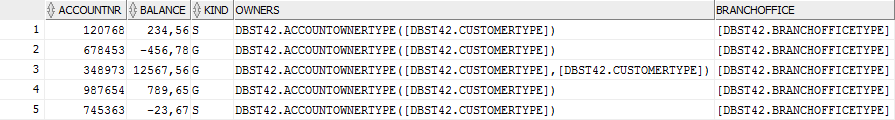
Zweigstellen-Tabelle:



Kunden-Tabelle:

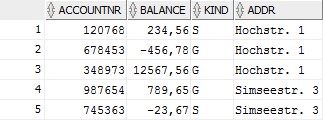


Konten-Tabelle:

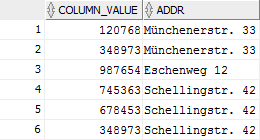


**2. Möglichkeit:**

Screendump von Aufgabe 4a)



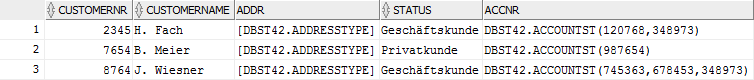
Screendump von Aufgabe 4b)



Zweigstellen-Tabelle:



Kunden-Tabelle:



## Aufgabe 5 (Fallstudie)

Zuerst haben wir für die Fallstudie die Aufgabenstellung analysiert und angefangen, Typen zu definieren um die Tabellen vollständig korrekt zu speichern.

Hierfür haben wir 30 Typen (Normale und List-Typen) angelegt, die wie folgt aussehen:

CREATE TYPE CampusT AS OBJECT (campus\_location VARCHAR(15), campus\_addr VARCHAR(127), campus\_phone VARCHAR(15), campus\_fax VARCHAR(15), campus\_head VARCHAR(31));

/

CREATE TYPE ProfessorT AS OBJECT (prof\_id INTEGER, prof\_name VARCHAR(31), prof\_contact VARCHAR(31), prof\_research VARCHAR(63), prof\_year INTEGER);

/

CREATE TYPE ProfessorListT AS TABLE OF REF ProfessorT;

/

CREATE TYPE DepartmentT AS OBJECT (dept\_id VARCHAR(3), dept\_name VARCHAR(31), dept\_head VARCHAR(31), dept\_prof ProfessorListT);

/

CREATE TYPE DepartmentListT AS TABLE OF DepartmentT;

/

CREATE TYPE SchoolT AS OBJECT (school\_id VARCHAR(3), school\_name VARCHAR(31), school\_head VARCHAR (31), school\_prof ProfessorListT);

/

CREATE TYPE SchoolListT AS TABLE OF SchoolT;

/

CREATE TYPE RCUnitT AS TABLE OF VARCHAR(127);

/

CREATE TYPE ResearchCentreT AS OBJECT (rc\_id VARCHAR(3), rc\_name VARCHAR(127), rc\_head VARCHAR(31), rc\_unit RCUnitT);

/

CREATE TYPE ResearchCentreListT AS TABLE OF ResearchCentreT;

/

-- TODO aggregation clustering technique

CREATE TYPE FacultyT AS OBJECT (fac\_id INTEGER, fac\_name VARCHAR(31), fac\_dean VARCHAR(15), dept DepartmentListT, school SchoolListT, rc ResearchCentreListT);

/

CREATE TYPE BuildingT AS OBJECT (bld\_id VARCHAR(4), bld\_name VARCHAR(31), bld\_location VARCHAR(2), bld\_level INTEGER, campus REF CampusT, fac REF FacultyT , MEMBER PROCEDURE show\_bld\_details));

/

CREATE TYPE PersonT AS OBJECT (person\_id VARCHAR(8), person\_surname VARCHAR(15), person\_forename VARCHAR(15), person\_title VARCHAR(7), person\_addr VARCHAR(127), person\_phone VARCHAR(15), person\_postcode VARCHAR(5), campus REF CampusT) NOT FINAL;

/

CREATE TYPE OfficeT AS OBJECT (office\_No VARCHAR(7), bld REF BuildingT, office\_phone VARCHAR(15));

/

CREATE TYPE ClassroomT AS OBJECT (class\_no VARCHAR(4), bld REF BuildingT, class\_capacity INTEGER);

/

CREATE TYPE LabEquipmentT AS TABLE OF VARCHAR(15);

/

CREATE TYPE LabT AS OBJECT (lab\_no VARCHAR(5), bld REF BuildingT, lab\_capacity INTEGER, lab\_equipment LabEquipmentT);

/

CREATE TYPE DegreeT AS OBJECT (deg\_id VARCHAR(4), deg\_name VARCHAR(31), deg\_length INTEGER, deg\_prereq VARCHAR(31), fac REF FacultyT);

/

CREATE TYPE ComputerskillsT AS TABLE OF VARCHAR(15);

/

CREATE TYPE OfficeskillsT AS TABLE OF VARCHAR(31);

/

CREATE TYPE TechnicianskillsT AS TABLE OF VARCHAR(15);

/

CREATE TYPE StaffT UNDER PersonT (office\_No VARCHAR(7), staff\_type VARCHAR(15)) NOT FINAL;

/

CREATE TYPE StudentT UNDER PersonT (student\_year INTEGER , MEMBER PROCEDURE insert\_student, MEMBER PROCEDURE delete\_student);

/

CREATE TYPE AdminT UNDER StaffT (admin\_title VARCHAR(31), admin\_computerskills ComputerskillsT, admin\_officeskills OfficeskillsT);

/

CREATE TYPE TechnicianT UNDER StaffT (tech\_title VARCHAR(31), tech\_skills TechnicianskillsT);

/

CREATE TYPE TutorT UNDER StaffT (tutor\_hours INTEGER, tutor\_rate DOUBLE PRECISION);

/

CREATE TYPE LecturerT UNDER StaffT (lect\_area VARCHAR(31), lect\_type VARCHAR(15)) NOT FINAL;

/

CREATE TYPE SeniorLecturerT UNDER LecturerT (senlect\_phd INTEGER, senlect\_master INTEGER, senlect\_honours INTEGER);

/

CREATE TYPE AssociateLecturerT UNDER LecturerT (asslect\_honours INTEGER, asslect\_year INTEGER);

/

CREATE TYPE SubjectT AS OBJECT (subject\_id VARCHAR(8), subject\_name VARCHAR(31), subject\_credit INTEGER, subject\_prereq VARCHAR(8), person REF PersonT);

/Dazu haben wir noch folgende 13 Tabellen angelegt, um konkrete Exemplare zu speichern:

CREATE TABLE Campus OF CampusT;

CREATE TABLE Professor OF ProfessorT;

CREATE TABLE Faculty OF FacultyT

NESTED TABLE dept STORE AS department\_nm (NESTED TABLE dept\_prof STORE AS dept\_prof\_nm)

NESTED TABLE school STORE AS school\_nm(NESTED TABLE school\_prof STORE AS school\_prof\_nm)

NESTED TABLE rc STORE AS rc\_nm (NESTED TABLE rc\_unit STORE AS rc\_unit\_nm);

CREATE TABLE Building OF BuildingT;

CREATE TABLE Office OF OfficeT;

CREATE TABLE Classroom OF ClassroomT;

CREATE TABLE Lab OF LabT

NESTED TABLE lab\_equipment STORE AS lab\_equip\_nm;

CREATE TABLE DegreeTbl OF DegreeT;

CREATE TABLE Staff OF StaffT;

CREATE TABLE Student OF StudentT;

CREATE TABLE Subject OF SubjectT;

CREATE TABLE Enrolls\_in (student REF StudentT, deg REF DegreeT);

CREATE TABLE Takes (student REF StudentT, subject REF SubjectT, mark INTEGER);

Dazu haben wir noch die im Diagramm eingezeichneten Funktionen wie folgt implementiert:

CREATE OR REPLACE FUNCTION show\_bld\_details(in\_bld\_id IN VARCHAR2)

RETURN VARCHAR2

IS building\_details VARCHAR2(255);

BEGIN

SELECT 'ID: '|| building.bld\_id ||', Name: '|| building.bld\_name ||', Location: '|| building.bld\_location ||', Level: '|| building.bld\_level ||', Campus: '|| DEREF(building.campus).campus\_location ||', Faculty: '||DEREF(building.fac).fac\_name

INTO building\_details

FROM Building building

WHERE building.bld\_id = in\_bld\_id;

RETURN(building\_details);

END show\_bld\_details;

-- SELECT show\_bld\_details('BB1') AS "Building Info" FROM DUAL;

**-- below is not working correctly, declaration would be needed**

CREATE OR REPLACE FUNCTION insert\_student(in\_person\_id IN VARCHAR2, in\_year IN NUMBER)

RETURN VARCHAR2

IS success\_state VARCHAR2(5); -- boolean type not supported

BEGIN

INSERT INTO Student (

SELECT pers.\*, in\_year FROM Person pers

WHERE pers.person\_id = in\_person\_id

);

success\_state := 'TRUE';

RETURN(success\_state);

END insert\_student;

CREATE OR REPLACE FUNCTION delete\_student(in\_person\_id IN VARCHAR2)

RETURN VARCHAR2

IS success\_state VARCHAR2(5); -- boolean type not supported

BEGIN

DELETE FROM Student student WHERE student.person\_id = in\_person\_id;

success\_state := 'TRUE';

RETURN(success\_state);

END delete\_student;

Schlussendlich haben wir mittels folgenden Insert-Statements Daten in die Tabellen angelegt:

INSERT INTO Campus

VALUES (CampusT('Albury/Wodonga', 'Parkers Road Wodonga VIC 3690',

'61260583700', '620260583777', 'John Hill'));

INSERT INTO Campus

VALUES (CampusT('City', '215 Franklin St. Melb VIC 3000',

'61392855100', '610392855111', 'Michael A. O''Leary'));

INSERT INTO Campus

VALUES (CampusT('Mildura', 'Benetook Ave. Mildura VIC 3502',

'61350223757', '61350223646', 'Ron Broadhead'));

INSERT INTO Campus

VALUES (CampusT('Bundoora', '221b Baker St. London NW1',

'6195135755', '6137196482', 'Sherlock Holmes'));

INSERT INTO Faculty

VALUES (FacultyT(1, 'Health Science', 'S. Duckett',

DepartmentListT(), SchoolListT(), ResearchCentreListT()));

INSERT INTO Faculty

VALUES (FacultyT(2, 'Humanity '||'&'||' Social Sc.', 'J. A. Salmond',

DepartmentListT(), SchoolListT(), ResearchCentreListT()));

INSERT INTO Faculty

VALUES (FacultyT(3, 'Law '||'&'||' Management', 'G. C. O''Brien',

DepartmentListT(), SchoolListT(), ResearchCentreListT()));

INSERT INTO Faculty

VALUES (FacultyT(4, 'Science, Tech. '||'&'||' Eng.', 'D. Finlay',

DepartmentListT(), SchoolListT(), ResearchCentreListT()));

INSERT INTO Faculty

VALUES (FacultyT(5, 'Regional Department', 'L. Kilmartin',

DepartmentListT(), SchoolListT(), ResearchCentreListT()));

INSERT INTO Professor VALUES (ProfessorT(42, 'Nick Hoogenraad', 'hoogenraad@cu.edu', 'Advanced Software Engineering', 2007));

INSERT INTO Professor VALUES (ProfessorT(88, 'Robin Anders', 'robin.anders@cu.edu', 'Computer Architecture', 2017));

INSERT INTO Professor VALUES (ProfessorT(101, 'Claude Bernard', 'cbernard@cu.edu', 'Networking', 2016));

INSERT INTO Professor VALUES (ProfessorT(420, 'Bruce Stone', 'b.stone@cu.edu', 'Software Development', 2015));

INSERT INTO Professor VALUES (ProfessorT(555, 'Chris Handley', 'handley@cu.edu', 'Compiler', 2009));

INSERT INTO Professor VALUES (ProfessorT(782, 'Sheena Reilly', 'sheenareilly@cu.edu', 'Discrete Mathematics', 2005));

INSERT INTO Professor VALUES (ProfessorT(1001, 'Alison Perry', 'Alison.Perry@cu.edu', 'Physics', 2012));

INSERT INTO Professor VALUES (ProfessorT(1337, 'Jan Branson', 'branson@cu.edu', 'Software Architecture', 2018));

INSERT INTO TABLE (SELECT dept FROM Faculty WHERE fac\_id=4)

VALUES(DepartmentT('4-1', 'Agricultural Sciences', 'Mark Sandeman',

ProfessorListT()));

INSERT INTO TABLE (SELECT dept FROM Faculty WHERE fac\_id=4)

VALUES(DepartmentT('4-2', 'Biochemistry', 'Nick Hoogenraad',

ProfessorListT()));

INSERT INTO TABLE (SELECT school FROM Faculty WHERE fac\_id=1)

VALUES(SchoolT('1-1', 'Human Biosciences', 'Chris Handley',

ProfessorListT((SELECT REF(p) FROM Professor p WHERE p.prof\_id=555))));

INSERT INTO TABLE (SELECT school FROM Faculty WHERE fac\_id=1)

VALUES(SchoolT('1-2', 'Human Comm. Sciences', 'Elizabeth Lavender',

ProfessorListT()));

INSERT INTO TABLE (SELECT rc FROM Faculty WHERE fac\_id=1)

VALUES(ResearchCentreT('1-1', 'Australian Research Centre in Sex, Health '||'&'||' Society', 'Martin Pitts', RCUnitT()));

INSERT INTO TABLE (SELECT rc FROM Faculty WHERE fac\_id=1)

VALUES(ResearchCentreT('1-2', 'Australian Institute for Primary Care', 'Hal Swerissen', RCUnitT()));

INSERT INTO TABLE(SELECT dept\_prof FROM TABLE(SELECT dept FROM Faculty WHERE fac\_id=4) dep WHERE dep.dept\_id = '4-2') (SELECT REF(p) FROM Professor p WHERE p.prof\_id=42 OR p.prof\_id=88 OR p.prof\_id=101 OR p.prof\_id=420);

INSERT INTO TABLE(SELECT school\_prof FROM TABLE(SELECT school FROM Faculty WHERE fac\_id=1) scho WHERE scho.school\_id = '1-2') (SELECT REF(p) FROM Professor p WHERE p.prof\_id=782 OR p.prof\_id=1001 OR p.prof\_id=1337);

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-1')

VALUES('SSAY Projects');

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-1')

VALUES('HIV Futures');

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-1')

VALUES('Australian Study of Health and Relationships');

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-2')

VALUES('Centre for Dev. and Innovation in Health');

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-2')

VALUES('Centre for Quality in Health '||'&'||' Community Svc.');

INSERT INTO TABLE (SELECT rc\_unit FROM TABLE (SELECT rc FROM Faculty WHERE fac\_id=1) r WHERE r.rc\_id='1-2')

VALUES('Lincoln Gerontology Centre');

INSERT INTO Building

VALUES (BuildingT('BB1', 'Beth Gleeson', 'D5', 4,

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'),

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=4)));

INSERT INTO Building

VALUES (BuildingT('BB2', 'Martin Building', 'F5', 4,

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'),

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=3)));

INSERT INTO Building

VALUES (BuildingT('BB3', 'Thomas Cherry', 'D4', 4,

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'),

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=1)));

INSERT INTO Building

VALUES (BuildingT('BB4', 'Physical Science 1', 'D5', 3,

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'),

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=4)));

INSERT INTO Office

VALUES (OfficeT('BG207',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB4'),

'94791118'));

INSERT INTO Office

VALUES (OfficeT('BG208',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB4'),

'94792393'));

INSERT INTO Classroom

VALUES (ClassroomT('TCLT',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB3'),

50));

INSERT INTO Classroom

VALUES (ClassroomT('TC01',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB3'),

30));

INSERT INTO Lab

VALUES (LabT('BG113',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB1'),

25, LabEquipmentT('25 PC', '1 Printer')));

INSERT INTO Lab

VALUES (LabT('BG114',

(SELECT REF(b) FROM Building b WHERE b.bld\_id='BB1'),

20, LabEquipmentT('21 PC')));

INSERT INTO DegreeTbl

VALUES (DegreeT('D100', 'Bachelor of Comp. Sci', 3, 'Year 12 or equivalent',

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=4)));

INSERT INTO DegreeTbl

VALUES (DegreeT('D101', 'Master of Comp. Sci', 2, 'Bachelor of Comp. Sci',

(SELECT REF(f) FROM Faculty f WHERE f.fac\_id=4)));

INSERT INTO Student VALUES ('01234234', 'Grant', 'Felix', 'Mr', '2 Boadle Rd Bundoora VIC', '0398548753', '3083',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'), 2000);

INSERT INTO Student VALUES ('01958652', 'Doe', 'John', 'Mr', '64 Austin St Holow VIC', '0321343123', '1337',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='City'), 2000);

-- additional Student needed for Enrolls\_in / Takes Entry without existing ID

INSERT INTO Student VALUES ('10012568', 'Shields', 'Duncan', 'Mr', '13 Flame St Seaside VIC', '0195837592', '3037',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='City'), 1995);

INSERT INTO Staff VALUES (TutorT('01234234', 'Grant', 'Felix', 'Mr', '2 Boadle Rd Bundoora VIC', '0398548753', '3083',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'), 'BG265', 'Tutor', 10, 20.00));

INSERT INTO Staff VALUES (AssociateLecturerT('10008895', 'Xin', 'Harry', 'Mr', '6 Kelley St Kew VIC', '0398875542', '3088',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'), 'BG212', 'Lecturer', 'Software Engineering', 'Associate', 2, 1999));

INSERT INTO Staff VALUES (AdminT('10002935', 'Jones', 'Felicity', 'Ms', '14 Rennie St Thornbury VIC', '0398722001', '3071',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Bundoora'), 'BG210', 'Admin', 'Office Manager', ComputerskillsT(), OfficeskillsT()));

INSERT INTO Staff VALUES (TutorT('01958652', 'Doe', 'John', 'Mr', '64 Austin St Holow VIC', '0321343123', '1337',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='City'), 'BG265', 'Tutor', 30, 35.00));

INSERT INTO Staff VALUES (AdminT('10008957', 'Jane', 'Patrick', 'Mr', '23 Rainbow Rd Allumy VIC', '0236263636', '3033',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='City'), 'BG221', 'Admin', 'Receptionist', ComputerskillsT(), OfficeskillsT()));

INSERT INTO Staff VALUES (TechnicianT('10005825', 'Gibbs', 'Lewroy', 'Mr', '127 Moltres Way Jotho VIC', '0285624733', '3042',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Mildura'), 'BG231', 'Technician', 'Network Officer', TechnicianskillsT()));

INSERT INTO Staff VALUES (TechnicianT('10015826', 'Beckett', 'Kate', 'Ms', '42 Donestry St Jibisy VIC', '0263957394', '3087',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Mildura'), 'BG232', 'Technician', 'Photocopy Technician', TechnicianskillsT()));

INSERT INTO Staff VALUES (SeniorLecturerT('10000255', 'Morgan', 'Henry', 'Mr', '2 London Ave Karrigan VIC', '0395182649', '3062',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='City'), 'BG225', 'Lecturer', 'Business Information', 'Senior', 2, 5, 7));

INSERT INTO Staff VALUES (SeniorLecturerT('10000258', 'Flow', 'Max', 'Mr', '26 Hollow Tips St Precidense VIC', '0492849184', '3012',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Mildura'), 'BG226', 'Lecturer', 'Business Administration', 'Senior', NULL, 1, 5));

INSERT INTO Staff VALUES (AssociateLecturerT('10006935', 'Gunn', 'Montgomery', 'Mr', '65 Arrow Ave Catery VIC', '0492847294', '3085',

(SELECT REF(c) FROM Campus c WHERE c.campus\_location='Mildura'), 'BG225', 'Lecturer', 'Software Development', 'Associate', NULL, 2001));

INSERT INTO TABLE (SELECT TREAT(VALUE(s) AS AdminT).admin\_officeskills FROM Staff s WHERE s.person\_id = '10002935') VALUES('Managerial');

INSERT INTO TABLE (SELECT TREAT(VALUE(s) AS AdminT).admin\_computerskills FROM Staff s WHERE s.person\_id = '10008957') VALUES('MS Office');

INSERT INTO TABLE (SELECT TREAT(VALUE(s) AS AdminT).admin\_officeskills FROM Staff s WHERE s.person\_id = '10008957') VALUES('Customer Service');

INSERT INTO TABLE (SELECT TREAT(VALUE(s) AS AdminT).admin\_officeskills FROM Staff s WHERE s.person\_id = '10008957') VALUES('Phone');

INSERT INTO TABLE (SELECT TREAT(VALUE(t) AS TechnicianT).tech\_skills FROM Staff t WHERE t.person\_id = '10005825')

VALUES('UNIX');

INSERT INTO TABLE (SELECT TREAT(VALUE(t) AS TechnicianT).tech\_skills FROM Staff t WHERE t.person\_id = '10005825')

VALUES('NT');

INSERT INTO TABLE (SELECT TREAT(VALUE(t) AS TechnicianT).tech\_skills FROM Staff t WHERE t.person\_id = '10015826')

VALUES('Electrician');

INSERT INTO Subject

VALUES (SubjectT('CSE21NET', 'Networking', 10, 'CSE11IS',

(SELECT REF(s) FROM Staff s WHERE s.person\_id='10008895')));

INSERT INTO Subject

VALUES (SubjectT('CSE42ADB', 'Advanced Database', 15, 'CSE21DB',

(SELECT REF(s) FROM Staff s WHERE s.person\_id='10006935')));

INSERT INTO Enrolls\_in VALUES ((SELECT REF(s) FROM Student s WHERE s.person\_id='01234234'), (SELECT REF(d) FROM DegreeTbl d WHERE d.deg\_id='D101'));

INSERT INTO Enrolls\_in VALUES ((SELECT REF(s) FROM Student s WHERE s.person\_id='10012568'), (SELECT REF(d) FROM DegreeTbl d WHERE d.deg\_id='D101'));

INSERT INTO Takes VALUES ((SELECT REF(st) FROM Student st WHERE st.person\_id='01234234'), (SELECT REF(su) FROM Subject su WHERE su.subject\_id='CSE42ADB'), 70);

INSERT INTO Takes VALUES ((SELECT REF(st) FROM Student st WHERE st.person\_id='10012568'), (SELECT REF(su) FROM Subject su WHERE su.subject\_id='CSE42ADB'), 80);

Bitte verzeihen Sie das fehlende Syntax-Highlighting, wir machen dies immer von Hand.

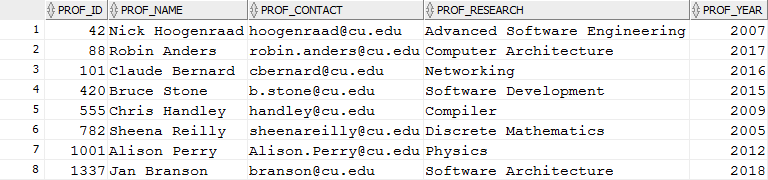
### Screendumps der Tabellen

Campus-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Professor-Table:



Faculty-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Building-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Office-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Classroom-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Lab-Table:

Ein Bild, das Screenshot enthält.

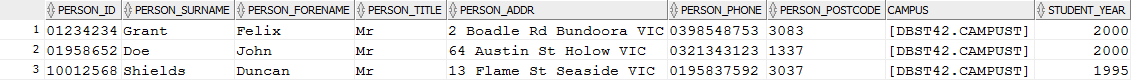
Automatisch generierte Beschreibung

Degree-Table:

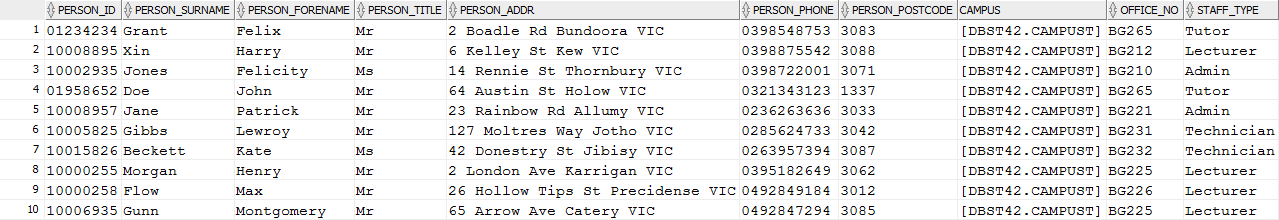
Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Student-Table:



Staff-Table:



Subject-Table:



Enrolls-In-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Takes-Table:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Um bspw. alle Tutors mit deren Attributen zu erhalten, können wir folgendes Script benutzen:

SELECT t.\*, TREAT(VALUE(t) AS TutorT).tutor\_hours tutor\_hours, TREAT(VALUE(t) AS TutorT).tutor\_rate tutor\_rate FROM Staff t WHERE TREAT(VALUE(t) AS TutorT) IS NOT NULL;

Das Ergebnis sieht dann wie folgt aus:





# Anmerkungen

Uns ist bei der Planung von Typen und Tabellen aufgefallen, dass einige Tabellenattribute unnötig waren, da man über Joins an dieses Attribut gelangen könnte. So hatte bspw. der Staff-Type in der Fallstudien-Beschreibung noch ein Attribut für das Gebäude (Building), dieses haben wir nicht übernommen, da man mit dem Office\_Nr-Attribut über einen Join zur Office-Tabelle zu dem Gebäude gelangen kann. So haben wir unnötige Redundanzen / Speicherplatz-Verschwendung vermieden.

Auf der anderen Seite haben wir in vielen Tabellen wie bspw. Building, Office, Classroom, Lab uvm. mit Referenzen statt einfachen IDs benutzt, dies hat zwei Vorteile:

* Wir können ohne Join sofort auf die Daten zugreifen (Dereferenzierung)
* Wir können sicherstellen das in bspw. der Takes-Tabelle nur gültige Studenten sind

Ich hoffe, dass diese Änderungen in Ihrem Sinn sind und unsere Gedankengänge für Sie sinnvoll bzw. nachvollziehbar erscheinen.

Falls Sie sich die SQL-Statements genauer ansehen oder zum selbst Testen kopieren möchten, finden Sie diese und alle anderen Dateien / Informationen auf [GitHub.com/AnyaW/DBS2](https://github.com/AnyaW/DBS2).