5. Übungsblatt

Aufgabe 1 (10 Punkte) – SQL Einführung

a.) i) Wofür steht die Abkürzung „SQL“? (1)
ii) Was ist kein Bestandteil des standard(!) SQLs von den Folgenden? (1)
Schema Definitionsprache,
Befehle für Datenanfragen und -modifikation,
Befehle für Anfrage und Verarbeitung von Bilddaten,
Befehle für Anfrage und Verarbeitung von Audiodaten,
Sicherheitskonzept und Kontrolle über Zugriffsberechtigungen

b.) Gegeben ist die folgende Tabelle:

Einige_Länder

Land	**Fläche**	**Population**	**Hauptstadt**
Austria	83.871	8.372.930	Vienna
Belgium	30.528	10.827.519	Brussels
Bulgaria	110.910	7.576.751	Sofia
Cyprus	9.251	801.851	Nicosia

(Quelle: http://en.wikipedia.org/wiki/Member_State_of_the_European_Union)

Was für ein Ergebnis liefern die folgenden SQL-Anfragen? (6)

i) select sum(Population) from Einige_Länder
ii) select Land, Fläche from Einige_Länder
where Fäche in (select min(Fläche) from Einige_Länder)
iii) select Hauptstadt from Einige_Länder order by Fläche

c.) Welche der obigen Anfragen ist eine eingeschachtelte Anfrage (nested query)? (2)

Aufgabe 2 (10 Punkte) – SQL

a.) Gegeben ist die folgende Datenbank, in der man die Noten von Studierenden speichert. Sie möchten Ihre eigene Noten anfragen, allerdings haben Sie Ihre Immatrikulationsnummer vergessen. Formulieren Sie die SQL-Anfrage dazu, ohne dabei ihre Immatrikulationsnummer zu benutzen! Das Ergebnis soll so aussehen: (2)

ID_der_Lehrveranstaltung	**Name_der_Lehrveranstaltung**	**Note**
…	…	…

Die Datenbank enthält die folgenden Tabellen:

Tabelle „Studierende“

Immatrikulations-nummer	Name	G_jahr	G_monat	G_Tag	Geschlecht	Adresse
…	…	…	…	…	…	…
123456	Erika Mustermann	1987	05	22	W	31141 Hi, Marienb. Str. 22
…	…	…	…	…	…	…

(G_Jahr, G_Monat und G_Tag enthalten das Geburtsdatum.)

Tabelle „Lehrveranstaltung“

ID_der_Lehrveranstaltung	Name_der_Lehrveranstaltung	Empfohlen_In_Semester
…	…	…

Tabelle „Noten“

Immatrikulationsnummer	ID_der_Lehrveranstaltung	Note
…	…	…

b.) Gegeben ist die Datenbank der Teilaufgabe a. Formulieren Sie folgende SQL-Anfragen (6):

i) Die erste Ziffer Ihrer eigenen Immatrikulationsnummer bezeichnen wir mit *k*. Sie sollen die Anzahl der solchen studierenden finden, die in dem *k*-ten Monat geboren sind. (Zum Beispiel: wenn Ihre eigene Immatrikulationsnummer mit 1 anfängt, dann sollen sie die Anzahl der Studierenden anfragen, die in Januar geboren sind.)

ii) Fragen Sie die durchschnittliche, die beste und die schlechteste Note pro Lehrveranstaltung an! Im Ergebnis sollen die Namen der Lehrveranstaltungen und die zugehörigen Noten (durchschnittlich, beste und schlechteste) gelistet werden.

iii) „f“) Falls Sie eine Frau sind, finden Sie jene Lehrveranstaltungen, in denen die Frauen durchschnittlich bessere Noten bekommen haben, als die Männer. Das Ergebnis soll eine Liste der Namen der entsprechenden Lehrveranstaltungen sein.
„m“) Wir sagen, dass ein(e) Student(in) „glücklich“ ist, wenn sie/er zwischen 1986 und 1990 geboren ist. Falls Sie ein Mann sind, finden Sie jene Lehrveranstaltungen, in denen die „glücklichen“ Studierenden durchschnittlich bessere Noten bekommen haben, als die Anderen. Das Ergebnis soll eine Liste der Namen der entsprechenden Lehrveranstaltungen sein.

c.) Sie möchten eine Tabelle erzeugen, in der jede Zeile den Namen einer/eines Studierenden und die Anzahl der von ihr/ihm besuchten Lehrveranstaltungen enthält. Nehmen wir an, dass einige Studierende an gar keine Lehrveranstaltungen teilnehmen. Sie möchten, dass auch diese Studierenden in der Ergebnistabelle vorkommen. Leider unterstütz Ihr SQL-Server weder „left join“, noch „right join“. Was können Sie machen? [Hinweise: 1) Es ist Ihnen erlaubt, den Inhalt der Datenbank sinnvollerweise zu modifizieren. 2) Geben Sie die SQL-Anfrage(n) an!]

Die Abgabe dieses Arbeitsblattes erfolgt per E-Mail an buza@ismll.de oder durch Einwurf im Briefkasten #45 bis zum 18. 05 .2010 10:00 Uhr (bei Einwurf bitte gelocht und getackert).
Viel Erfolg!

KLEINES SQL DEMO

```
CREATE TABLE LESER (ID VARCHAR(6), VORNAME VARCHAR(20), NACHNAME VARCHAR(20), GEBURTSDATUM DATE,
GESCHLECHT VARCHAR(1), NATIONALITAET VARCHAR(1));

INSERT INTO LESER VALUES ('000001', 'Krisztian', 'Buza', '11.11.11', 'M', 'H');
INSERT INTO LESER VALUES ('000002', 'Erika', 'Mustermann', '21.3.86', 'W', 'D');
INSERT INTO LESER VALUES ('000003', 'Christoph', 'Freudentaler', '15.3.98', 'M', 'A');
INSERT INTO LESER VALUES ('000004', 'Steffen', 'Rendle', '31.1.88', 'M', 'D');
INSERT INTO LESER VALUES ('000005', 'Christine', 'Preisach', '25.4.87', 'W', 'D');
INSERT INTO LESER VALUES ('000006', 'Anna', 'Musterfrau', '27.6.86', 'W', 'D');

SELECT * FROM LESER;
SELECT VORNAME, NACHNAME FROM LESER;
SELECT COUNT(*) FROM LESER;
SELECT * FROM LESER ORDER BY NACHNAME;
SELECT * FROM LESER ORDER BY NACHNAME GEBURTSDATUM;
SELECT * FROM LESER ORDER BY NACHNAME GEBURTSDATUM DESC;
SELECT count(*) FROM LESER GROUP BY GESCHLECHT;

INSERT INTO LESER VALUES ('000007', 'Annekatrin', 'Musterfrau', '27.6.86', 'W', 'A');

SELECT GESCHLECHT, count(*) FROM LESER GROUP BY GESCHLECHT;
SELECT count(*) FROM LESER WHERE GESCHLECHT = 'W' GROUP BY GESCHLECHT;
SELECT NATIONALITAET, GESCHLECHT, COUNT(*) from LESER group by NATIONALITAET, GESCHLECHT;

CREATE TABLE BUECHER (ISBN VARCHAR(9), TITEL VARCHAR(100), AUTHOR VARCHAR(20), ERSCHEINUNGSDATUM DATE,
SEITEN_ANZAZHL INT);

INSERT INTO BUECHER VALUES ('1234-5678', 'Einführung in Clustering', 'Christian Borgelt', '01.01.2008', 250);
INSERT INTO BUECHER VALUES ('1111-0000', 'C für Profis', 'Bill Gates', '01.05.2008', 250);
INSERT INTO BUECHER VALUES ('1111-0001', 'HTML Programmierung', 'Jörg S.', '31.05.2009', 250);
INSERT INTO BUECHER VALUES ('1111-0002', 'Mathematische Statistik', 'C. Freudentaler.', '01.01.2010', 500);

CREATE TABLE AUSLEIHE (BUCH_ISBN VARCHAR(9), LESER_ID VARCHAR(6))

INSERT INTO AUSLEIHE VALUES ('1111-0000','000003');
INSERT INTO AUSLEIHE VALUES ('1111-0001','000001');
INSERT INTO AUSLEIHE VALUES ('1111-0002','000001');

SELECT * from ausleihe;

SELECT VORNAME, NACHNAME, LESER_ID, BUCH_ISBN from ausleihe, LESER;

SELECT VORNAME, NACHNAME, LESER_ID, BUCH_ISBN from ausleihe, LESER
where Ausleihe.LESER_ID=LESER.ID;

SELECT VORNAME, NACHNAME, TITEL from ausleihe, LESER, BUECHER
where Ausleihe.LESER_ID=LESER.ID AND Ausleihe.BUCH_ISBN = BUECHER.ISBN;

SELECT VORNAME, NACHNAME, count(*) from ausleihe, LESER
where Ausleihe.LESER_ID=LESER.ID GROUP BY VORNAME, NACHNAME;

CREATE VIEW AUSLEIHE_SCHOEN AS
SELECT VORNAME, NACHNAME, TITEL from ausleihe, LESER, BUECHER
where Ausleihe.LESER_ID=LESER.ID AND Ausleihe.BUCH_ISBN = BUECHER.ISBN;

CREATE VIEW AUSLEIHE_STATISTIK AS
SELECT VORNAME, NACHNAME, count(*) AS BUCH_ANZAHL from ausleihe, LESER
where Ausleihe.LESER_ID=LESER.ID
GROUP BY VORNAME, NACHNAME;
```

```
select * from Ausleihe_statistik;

select * from Ausleihe_statistik where buch_anzahl >= 2;


-- Wir nehmen an, dass es beliebig viele Exemplare von Büchern in der Bibliothek gibt
-- Bespielaufgabe: Wir suchen die Bücher (Titel!), die durch mehr Frauen als Männer ausgeliehen worden sind

INSERT INTO AUSLEIHE VALUES ('1111-0000','000005');
INSERT INTO AUSLEIHE VALUES ('1111-0001','000005');
INSERT INTO AUSLEIHE VALUES ('1111-0002','000005');
INSERT INTO AUSLEIHE VALUES ('1111-0000','000002');
INSERT INTO AUSLEIHE VALUES ('1111-0001','000002');
INSERT INTO AUSLEIHE VALUES ('1111-0002','000002');
INSERT INTO AUSLEIHE VALUES ('1234-5678','000001');
INSERT INTO AUSLEIHE VALUES ('1234-5678','000003');
INSERT INTO AUSLEIHE VALUES ('1234-5678','000004');
INSERT INTO AUSLEIHE VALUES ('1234-5678','000002');

select * from ausleihe;

select * from ausleihe, leser where ausleihe.leser_id = leser.id;

select * from ausleihe, leser, buecher
where ausleihe.leser_id = leser.id and buecher.isbn = ausleihe.buch_isbn;

select geschlecht, titel from ausleihe, leser, buecher
where ausleihe.leser_id = leser.id and buecher.isbn = ausleihe.buch_isbn;


select geschlecht, titel, count(*) from ausleihe, leser, buecher
where ausleihe.leser_id = leser.id and buecher.isbn = ausleihe.buch_isbn
group by titel, geschlecht;

create view zwischenergebnis as
select geschlecht, titel, count(*) as anzahl from ausleihe, leser, buecher
where ausleihe.leser_id = leser.id and buecher.isbn = ausleihe.buch_isbn group by titel, geschlecht;

select * from zwischenergebnis;

select * from zwischenergebnis z1, zwischenergebnis z2 where z1.titel = z2.titel;

select * from zwischenergebnis z1, zwischenergebnis z2
where z1.titel = z2.titel and z1.geschlecht ='W' and z2.geschlecht = 'M';

select * from zwischenergebnis z1, zwischenergebnis z2
where z1.titel = z2.titel and z1.geschlecht ='W' and z2.geschlecht = 'M' and z1.anzahl > z2.anzahl;

select z1.titel from zwischenergebnis z1, zwischenergebnis z2
where z1.titel = z2.titel and z1.geschlecht ='W' and z2.geschlecht = 'M' and z1.anzahl > z2.anzahl;

drop table [Name der Tabelle]
drop view [Name des Views]
```