Gruppe **A**

Please fill in your name and registration number (Matrikelnr.) **immediately**.

EXAM IN	**SAMPLE SOLUTION**	20.01.2023 **GROUP A**
Matrikelnr.	Last Name	First Name

Duration: 90 minutes. Provide the solutions at the designated pages; solutions on additional sheets of paper are not considered. **Good Luck!**

Task	1	2	3	4	5	6	Σ
Max. Points	12	13	10	11	12	12	70
Points							

Please, *do not* remove the staple.

Add your student ID and last name on every sheet, it simplifies entering points.

Attention!

To all questions with a multiple choice option, the following rule applies: Just checking an option gives no points; points are only granted in combination with the required justification/example/...

Notation:

In Questions 1–3, the notation as known from the lecture slides and exercises for transactions T_i is used. Recall:

- $r_i(O)$ and $w_i(O)$: Read, respectively write operation of transaction T_i on object O.
- b_i, c_i, a_i: begin (`BEGIN OF TRANSACTION`), commit (`COMMIT`) and abort (`ABORT`/`ROLLBACK`) of T_i.

The indices $_i$ can be omitted if it is clear which transaction an operation belongs to.

In addition, log records also have the same format as used throughout the lecture:

`[LSN, TA, PageID, Redo, Undo, PrevLSN]` for "normal" records,
`[LSN, TA, BOT, PrevLSN]` for `BOT` log-records, and
`[LSN, TA, COMMIT, PrevLSN]` for `COMMIT` records.

Compensation log records (CLRs) follow the format
⟨`LSN, TA, PageID, Redo, PrevLSN, UndoNextLSN`⟩ and
⟨`LSN, TA, BOT, PrevLSN`⟩

In these records, `LSN` denotes the Log-Sequence Number, `TA` the transaction, `PageID` the page that was updated, `Redo` and `Undo` the information needed for the Redo resp. Undo operations, `UndoNextLSN` the LSN of the next log record of the same transaction to be undone, and `PrevLSN` the LSN of the previous log record of the same transaction.

In case of logical logging, the changes to the previous value of the database instance are stated only using *addition* and *subtraction*, e.g. [·, ·, ·, X+=d_1, X-=d_2, ·].

Question 1: Properties of transactions (12)

Consider the schedule below, consisting of a sequence of basic operations of four transaction T_1, T_2, T_3 and T_4 on database objects A, B, C and D.

T_1	T_2	T_3	T_4
b_1	b_2	b_3	b_4
	$r_2(A)$		
	$r_2(B)$		
$r_1(A)$			
	c_2		
$w_1(A)$			
		$r_3(A)$	
			$r_4(A)$
			$w_4(B)$
			$w_4(C)$
$r_1(B)$			
$r_1(C)$			
			$r_4(C)$
			$r_4(D)$
$r_1(B)$			
			$w_4(D)$
			$w_4(E)$
		$r_3(E)$	
		$r_3(D)$	
		c_3	
			c_4
c_1			

a) Specify the transactions between which there is a read dependency. This means, for each transaction specify from which other transactions this transaction reads (if a transaction does not read from any other transaction, please cross out the corresponding field).

(4 Points)

a) Read dependency:

T_1 reads from T_4 ..... T_2 reads from -- .....

T_3 reads from T_1, T_4 ... T_4 reads from T_1 .....

b) Then determine whether the schedule avoids cascading reset or not, as well as whether it is strict or not. Provide a brief justification for each, based on the schedule.
(*Attention:* Checking an option without providing a justification gives no points!)

(4 Points)

b) Properties:

Schedule avoids cascading resets: ◯ yes ⊗ no

Reason: Not all transactions that are read from .....

have their commit before the respective read operation, e.g. b_1 and b_3

Schedule is strict: ◯ yes ⊗ no

Reason: a strict history has to avoid cascading resets

.....

c) Consider the pairs of transactions given below. For each of the two pairs, determine whether the two transactions are conflict serializable or not.
If not, provide a sequence of pairs of conflict operations of the two transactions which exclude the existence of a conflict equivalent, serial schedule.
If the transactions are conflict serializable, provide a conflict equivalent, serial history (= sequence of elementary operations!) of the two transactions. (4 Points)

Transactions T_1 and T_3: conflict serializable: ⊗ yes ◯ no

Answer: b_1, $r_1(A)$, $w_1(A)$, $r_1(B)$, $r_1(C)$, $r_1(B)$, c_1, b_3, $r_3(A)$, $r_3(E)$, $r_3(D)$, c_3 .....

.....

Transactions T_2 and T_4: conflict serializable: ◯ yes ⊗ no

Answer: e.g. $(w_1(A), r_4(A))$, $(w_4(B), r_1(B))$, ...

.....

MatrikelNr: _____ **Last name:** _____

Question 2: Logging and Recovery (13)

Consider the given schedule in the table below, which contains the three Transactions T_1, T_2, and T_3. The pages contain data as given below. Assume that field A is located on page P_A, field B on page P_B, ...

$$A : 17,\ B : 4,\ C : 9,\ D : 8$$

a) Lines in the history indicate changes in a field or local variable. Provide for each line of the schedule, the value of the field/variable after the operation. (3 Points)

b) Provide changes to the log-records of the specified schedule as a list. Use the order in which records have been created. State records in the format as suggested in the beginning. Employ logical logging. (4 Points)

	T_1	T_2	T_3	(a) Var./Field	(b) Log records
1	$\texttt{BOT}_1$			.....	[#1, T_1, BOT,#0] .....
2		$\texttt{BOT}_2$		.....	[#2, T_2, BOT,#0] .....
3			$\texttt{BOT}_3$	.....	[#3, T_3, BOT,#0] .....
4		$r_2(A, a_2)$		$a_2 = 17$	.....
5		$w_2(B, a_2 + 3)$		$B = a_2 + 3 = 20$	[#4, T_2, P_B, $B+=16, B-=16$,#2]
6			$r_3(B, b_3)$	$b_3 = 20$	.....
7		$r_2(C, c_2)$		$c_2 = 9$	.....
8	$w_1(D, 2)$			$D = 2$	[#5, T_1, P_D, $D-=6, D+=6$,#1]
9		$w_2(D, c_2 + 1)$		$D = c_2 + 1 = 10$	[#6, T_2, P_D, $D+=8, D-=8$,#4]
10			$r_3(A, a_3)$	$a_3 = 17$	.....
11			$w_3(A, a_3 + 5)$	$A = a_3 + 5 = 22$	[#7, T_3, P_A, $A+=5, A-=5$,#3]
12		$r_2(B, b_2)$		$b_2 = 20$	.....
13		$w_2(D, b_2)$		$D = b_2 = 20$	[#8, T_2, P_D, $D-=10, D+=10$,#6]
14	$w_1(D, 4)$			$D = 4$	[#9, T_1, P_D, $D-=16, D+=16$,#5]
15			$w_3(A, b_3 + 5)$	$A = b_3 + 5 = 25$	[#10, T_3, P_A, $A+=3, A-=3$,#7]
16		$\texttt{commit}_2$		.....	[#11, T_2, COMMIT,#8] .....
17	$\texttt{abort}_1$			.....	⟨#12, T_1, P_D,$D+=16$,#9,#5⟩ ..
				.....	⟨#13, T_1, P_D,$D+=6$,#12,#1⟩ ..
				.....	⟨#14, T_1, BOT, #13⟩ .....

MatrikelNr: _____ **Last name:** _____

Distribution of Points Question 2

(2a)	Reads (a,b,c) correct each 0.5	+1.5
	Writes (a,b,d) correct each 0.5	+1.5
		$\sum$ 3
	Vars or Fields not distinguishable	-1
(2b)	CLR correct	+2
	LR correct	+2
		$\sum$ 4
	Format of the log record wrong	-0.5
	LastLSN wrong	-1
	UndoNextLSN wrong	-1
	Wrong values redo/undo	-0.5

c) *Consider the given log records of transactions T_1 und T_2 and the relevant part of the database content:*

P_A	*LSN: #8*
$A = 40$	

P_B	*LSN: #6*
$B = 23$	

Perform a recovery following the ARIES algorithm using these log records. Answer the sub-tasks below.

Log Archive (task description)

$[\#1, T_1,$ BOT,		$\#0]$
$[\#2, T_1, P_A,$	$A+=20, A-=20,$	$\#1]$
$[\#3, T_2,$ BOT,		$\#0]$
$[\#4, T_1, P_A,$	$A+=4, A-=4,$	$\#2]$
$[\#5, T_1, P_B,$	$B+=4, B-=4,$	$\#4]$
$[\#6, T_1, P_B,$	$B+=8, B-=8,$	$\#5]$
$[\#7, T_2, P_A,$	$A+=10, A-=10,$	$\#6]$
$[\#8, T_2, P_A,$	$A+=10, A-=10,$	$\#7]$
$[\#9, T_2, P_B,$	$B+=2, B-=2,$	$\#8]$
$[\#10, T_2,$ COMMIT,		$\#9]$
$\langle\#11, T_1, P_B,$	$B-=8, \#6,$	$\#5\rangle$
$\langle\#12, T_1, P_B,$	$B-=4, \#11,$	$\#4\rangle$

c1) Compute the values in the im database and PageLSN after the *redo*-stage. (2 Points)

P_A	LSN: #8
$A =$ 40	

P_B	LSN: #12
$B =$ 13	

c2) State the required CLR records. (3 Points)
⟨LSN, TA, PageID, Redo, PrevLSN, UndoNextLSN⟩
⟨LSN, TA, BOT, PrevLSN⟩

⟨#13, T_1, P_A, A-=4, #12, #2 ⟩ .....

⟨#14, T_1, P_A, A-=20, #13, #1⟩ .....

⟨#15, T_1, BOT,#14⟩ .....

.....

.....

c3) Compute the values in the database and PageLSN *after* the *undo* stage. (1 Point)

P_A	LSN: #14
$A =$ 16	

P_B	LSN: #12
$B =$ 13	

Distribution of Points

(c1) $\sum$ +2 thereof LSN, Wert each +0.5:

(c3) $\sum$ +1 thereof Wert each +0.5 (no Points for LSN, since the can be read off directly)

(c2) $\sum$ +3

- 1 Point per CLR
- -1 Point per additional CLR
- -1 Wrong format
- 0 Points if undo of committed transaction

Total (c) $\sum$ +6

MatrikelNr: _____ **Last name:** _____

Question 3: Locking/Concurrency Control (10)

(3a) 2-Phase Locking (2PL)

Below, a sequence of Exclusive- and Share Locks ($X_i(O)$, $S_i(O)$), releases of locks ($relX_i(O)$, $relS_i(O)$), as well as read- and write operations is given. For different entries within the same row, an arbitrary order may by assumed.

State for each of the five transactions T_1, T_2, T_3, T_4, and T_5, whether they follow the *2-Phase Locking (2PL)* protocol. If not, describe why 2PL is violated. (5 points)

T_1	T_2	T_3	T_4	T_5
BOT	BOT	BOT	BOT	BOT
$X_1(A)$	$S_2(C)$	$X_3(E)$		
	$S_2(D)$		$X_4(G)$	
$r_1(A)$	$X_2(F)$		$w_4(G)$	$S_5(B)$
$w_1(A)$			$relX_4(G)$	$X_5(A)$
	$r_2(D)$		$S_4(B)$	$r_5(B)$
	$r_2(C)$	$w_3(E)$		
$S_1(B)$				
	$relS_2(D)$			$w_5(A)$
$w_1(B)$	$relS_2(C)$	$X_3(D)$		
	$w_2(F)$	$r_3(D)$		$X_5(C)$
$relS_1(B)$	$relX_2(F)$	$w_3(D)$		$relX_5(A)$
$relX_1(A)$	c_2			$w_5(C)$
		$X_3(A)$		
c_1		$w_3(A)$	$r_4(B)$	
		$relX_3(A)$	$relS_4(B)$	$relS_5(B)$
		$relX_3(D)$		
		$relX_3(E)$		$relX_5(C)$
		c_3	c_4	c_5

T_1: writes on B, but only SL. .....

T_2: correct 2PL. .....

T_3: correct 2PL. .....

T_4 locks B (SL(B)), after the lock ....

XL(D) has been released. .....

T_5 writes on A while T_1 .....

has a lock (XL(A)). .....

.....

.....

.....

.....

.....

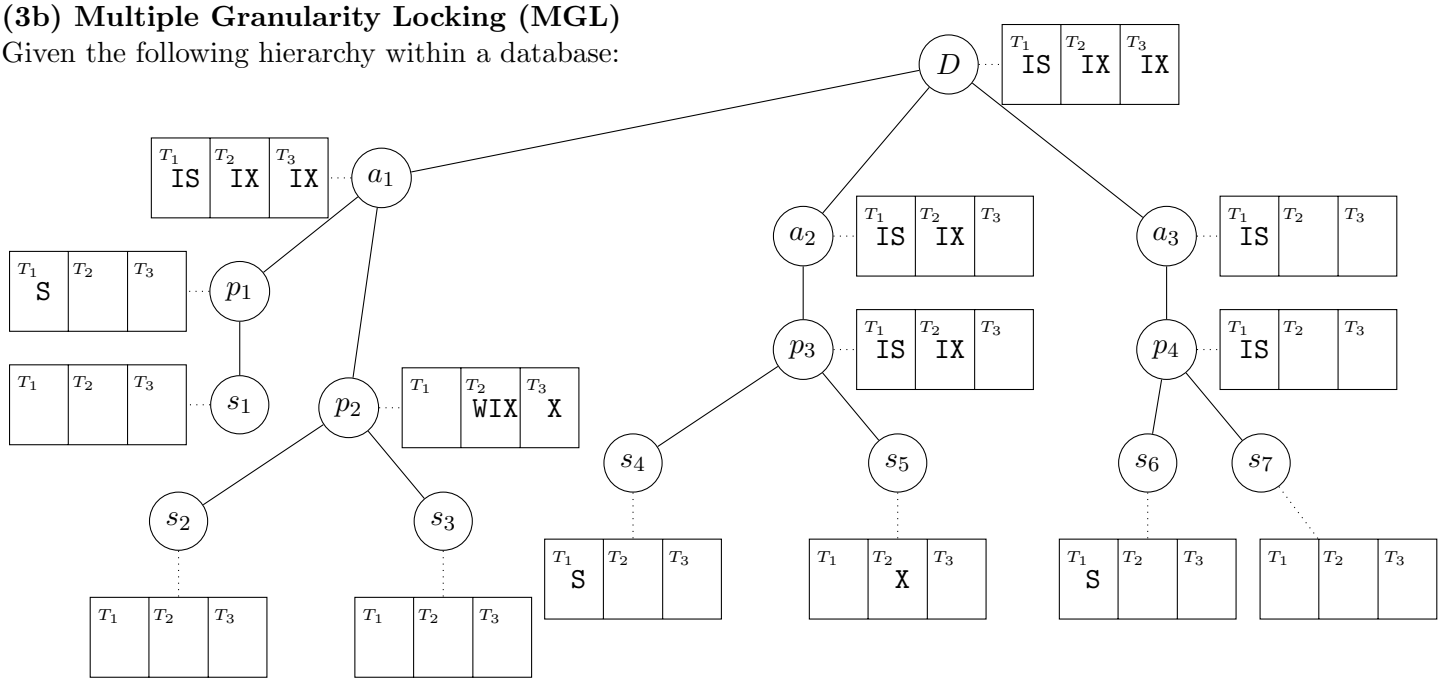
.....

Note: Consider transactions also together (not only one at a time).

by correct answered transaction. +1

$\sum$ 5

(3b) Multiple Granularity Locking (MGL)
Given the following hierarchy within a database:



Assume that the transactions T_1, T_2, and T_3 request locks according to the following sequence:

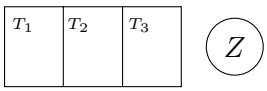
$$\mathrm{X}_3(p_2), \mathrm{X}_2(s_5), \mathrm{S}_1(p_1), \mathrm{S}_1(s_4), \mathrm{X}_2(s_2), \mathrm{X}_2(s_7), \mathrm{S}_1(s_6)$$

Furthermore, assume that the transactions follow the MGL protocol for getting the required locks while applying MGL correctly. Note that the stated sequence contains only locks, which the transactions request. In order to obtain the respective locks additional intention locks might be necessary, these need to be stated. The term $S_i(o)$ states that transaction T_i requests a shared lock for reading on object o zu erhalten and $X_i(o)$ states that T_i requests an exclusive lock for writing.

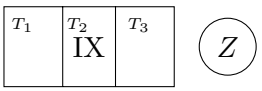
Describe the situation after working through these operations by filling into the above figure which transactions hold which locks at each node. At each node, put S, X, IS and IX into the field with the number of the transaction to indicate that the transaction holds the corresponding lock on that node.
If a transaction requests a lock but does not get it, i.e. has to wait, please put WS, WX, WIS or WIX into the corresponding field. If a transaction is blocked, then ignore all further actions of this transaction.

(5 Points)

Example:



Transaktion T_2 requires an intention exclusive lock (IX) at node Z and the state is fine. In that case, place IX in the corresponding field at the node, as given on the right side.



T_1 correct	+1
T_2 correct	+3
T_3 correct	+1
$\sum$	5
T_2 not waiting	-3
Per wrong entry	-0.5

Page intentionally left empty to simplify solving Question 4. Please, do not put solutions here.

Questions 4–6 are all based on the database schema described on this page.

Question 4: Defining a database schema using SQL and dependencies (11)

a) The following schema is given

`readers`	(name, sid, country, mostBooks: *deliveries.id*)
`deliveries`	(id, date, toName: *readers.name*, toSid: *readers.sid*)
`books`	(id: *deliveries.id*, itemNumber, pages, cost, fromAuthor: *authors.name*)
`authors`	(name, printedPages, country)

A university assistant needs to find a relational schema for a question in a data base systems test and, in the absence of new ideas, decides upon a generic schema consisting of readers, authors, deliveries and books.

A reader is uniquely identified by `name` and social security id `sid`. In addition, the `country` of the reader is also saved.

Each of the deliveries is uniquely identified by an `id`. The ID must be a sequence, starting with 10 and continuing in steps of 5. The `date` on which the delivery was sent is also recorded. The reader who received the delivery is also saved (`toName, toSid`). In addition, the `mostBooks` delivery to date is recorded for the respective reader (For the solution of this task it is not necessary to check if `mostBooks` really refers to the delivery with the most books).

Each delivery contains books. Every book is uniquely identified by the combination of the `id` of the associated delivery and an `itemNumber`. Each book also has a `cost`, a number of `pages` and an author (`fromAuthor`). `pages` must be at least 1. `cost` should be implemented as a decimal number with at most 2 decimal places.

Each author has a unique `name`. In addition, the total number of printed pages (`printedPages`) and the `country` of the author are recorded in the database. The number of printed pages must lie between 0 and 9999 for each entry in the table. (For the solution of this assignment it is not necessary to check if `printedPages` really is set to the total number of pages from all related deliveries.)

Provide the necessary SQL statements to create the described schema, with all described integrity constraints. Choose appropriate types for attributes. **You may use VC in place of VARCHAR(100).**

(7 points)

```
CREATE SEQUENCE seq_id INCREMENT BY 5 MINVALUE 10 NO CYCLE;

CREATE TABLE readers(
   name         VARCHAR(100),
   sid          INTEGER,
   country              VARCHAR(100) NOT NULL,
   mostBooks    INTEGER NOT NULL,
   PRIMARY KEY (name,sid)
);

CREATE TABLE deliveries(
   id           INTEGER DEFAULT nextval('seq_id') PRIMARY KEY,
   date         DATE NOT NULL,
   toName       VARCHAR(100),
   toSid        INTEGER,
   FOREIGN KEY (toName,toSid) REFERENCES readers(name,sid)
);

CREATE TABLE authors(
   name         VARCHAR(100) PRIMARY KEY,
   printedPages INTEGER NOT NULL CHECK(printedPages BETWEEN 0 AND 9999),
   country      VARCHAR(100) NOT NULL
);

CREATE TABLE books(
   id   INTEGER REFERENCES deliveries(id),
   itemNumber   INTEGER,
   pages    INTEGER NOT NULL CHECK(pages > 0),
   cost         NUMERIC(5,2) NOT NULL,
   fromAuthor   VARCHAR(100) REFERENCES authors(name),
   PRIMARY KEY (id,itemNumber)
);

ALTER TABLE readers ADD CONSTRAINT c_mostBooks
   FOREIGN KEY (mostBooks) REFERENCES deliveries(id)
   DEFERRABLE INITIALLY DEFERRED;
```

(4a)	Create Table statement correct (each 1 point)	+1
	Create Sequence statement correct	+1
	Wrong order of statements	-1
	Foreign Key Constraint (mostBooks) correct	+2
	$\sum$ 7, min 0	
(4b)(i/ii)	One missing/incorrect value -1	-1
	Two or more missing/incorrect values -2	-2
	$\sum$ 4, min 0	

Hint: Take care of the order of your statements.

b) For this task, you have to consider different key-dependencies.

(4 points)

i) Consider the following schema:

`room` (house: *house.number*, name)
`house` (number: *room.house*, largestRoom: *room.name*)

Complete the following tables in such a way **that the resulting database instance satisfies the schema.** You must assume that the two tables represent the complete database. To get points, the tables must be filled completely.

room

house	**name**
13	Living Room
22	Bathroom
43	Kitchen
52	Bedroom
10	Garage

house

number	**largestRoom**
22	Bathroom
43	Kitchen
10	Garage
13	Living Room
52	Bedroom

Note: The solution is not unique, you need to enter only one possible solution. **Note:** There are multiple ways of solving this subtask, and only one possible solution is shown here.

ii) Consider the following schema:

`X` (id, Z: *Z.id*)
`Y` (id, X: *X.id*, Z: *Z.id*)
`Z` (id, Y: *Y.id*)

Complete the following tables in such a way **that the resulting database instance satisfies the schema.** You must assume that the three tables represent the complete database. To get points, the tables must be filled completely.

X

id	**Z**
1	3
2	2
3	1

Y

id	**X**	**Z**
1	2	1
2	3	1
3	1	2

Z

id	**Y**
1	1
2	2
3	3

Note: The solution is not unique, you need to enter only one possible solution. **Note:** There are multiple ways of solving this subtask, and only one possible solution is shown here.

Question 5: Recursive Queries (12)

You are given the following recursive query using the same database schema as was used in question 4 a) :

```
WITH RECURSIVE tmp(id, toName) AS
(
 SELECT  d.id, d.toName
 FROM    deliveries d
 WHERE   d.toName = 'Anna Schweiger' AND d.toSid = '22'
UNION
 SELECT  d.id, d.toName
 FROM    tmp t, deliveries d, books b1, books b2
 WHERE   t.id = b1.id
         AND d.id = b2.id
         AND b1.fromAuthor = b2.fromAuthor
)
SELECT  t.id, t.toName
FROM    tmp t
```

Evaluate the given query over the database instance, which can be found on Page 16. We suggest to unstaple Page 16 for easier handling.

id	name
10	Anna Schweiger
30	Hubert Stoll
40	Elisabeth Stark
35	Theodor Riegler

Punkteverteilung Aufgabe 5

'Anna Schweiger' richtig (nicht rekursiv)	+2
Andere Zeile richtig (rekursiv)	+3
Pro komplett falsche Zeile	-2
Leichter Fehler (z.B. eine falsche id)	-1
Schwerer Fehler (z.B. statt id toSid verwendet)	-2
Duplikate (Ausführung ca. wie UNION ALL)	-2
Behauptete Lösung ist leer da Endlosschleife	-2
$\sum$ 11, min 0	
Richtiges Schema	+1

Question 6: PL/SQL Trigger (12)

Consider the database instance given on **Page 16** (last sheet of the exam). We suggest to unstaple that page.

Each of the sub-tasks contains an SQL statement, which has to be evaluated on the **given instance**. For each task assume that the database was reset and only the content of the last page is relevant, meaning, an insert statement from sub-task (a) has no effect on sub-task (b) etc. Moreover, each function and trigger is only relevant for the corresponding sub-task.

Provide the result of the `SELECT`-Statements. If an error occurs, provide an explanation.

You are free to use arbitrary shorthand notations in your answers (as long as they can be uniquely identified).

a) (4 points)

```
CREATE OR REPLACE FUNCTION fInsertBooksTwo() RETURNS TRIGGER AS $$
BEGIN

        UPDATE authors SET printedPages = printedPages + NEW.pages
        WHERE NEW.fromAuthor = name;

        RETURN NEW;

END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER fInsertBooksTwo AFTER INSERT ON books
       FOR EACH ROW EXECUTE PROCEDURE fInsertBooksTwo();
```

```
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (40, 3, 100, 50.00, 'Wells');
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (35, 2, 50, 48.99, 'Swift');

SELECT * FROM authors WHERE country = 'United Kingdom';
```

```
 name | printedpages | country
----------+--------------+----------------
Christie | 1360 | United Kingdom
Tolkien | 550 | United Kingdom
Wells | 100 | United Kingdom
Swift | 375 | United Kingdom
```

b) (4 points)

```
CREATE OR REPLACE FUNCTION fInsertBooks() RETURNS TRIGGER AS $$
DECLARE
        curr_del INTEGER;
        curr_sid INTEGER;
        curr_name VARCHAR(255);
        largest_del INTEGER;
BEGIN

        SELECT COUNT(itemNumber), d.toSid, d.toName
               INTO curr_del, curr_sid, curr_name
        FROM deliveries d, books b
        WHERE NEW.id = d.id AND d.id = b.id
        GROUP BY d.toSid, d.toName;


        SELECT COUNT(itemNumber) INTO largest_del
        FROM readers r, deliveries d, books b
        WHERE curr_name = r.name AND curr_sid = r.sid AND
              r.mostBooks = d.id AND d.id = b.id;

        IF curr_del > largest_del THEN
                UPDATE readers SET mostBooks = NEW.id
                WHERE curr_name = name AND curr_sid = sid;
        END IF;

        RETURN NEW;

END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER fInsertBooks AFTER INSERT ON books
       FOR EACH ROW EXECUTE PROCEDURE fInsertBooks();
```

```
INSERT INTO deliveries(id, date, toName, toSid) VALUES (50, '24.11.2022', 'Anna Schweiger', 22);
INSERT INTO deliveries(id, date, toName, toSid) VALUES (55, '24.11.2022', 'Elisabeth Stark', 25);

INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (50, 1, 475, 50.00, 'Goethe');
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (50, 2, 325, 30.00, 'Twain');
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (50, 3, 250, 40.00, 'Jelinek');
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (55, 1, 322, 45.50, 'Twain');

SELECT name, mostBooks FROM readers;
```

```
 name | mostbooks
------------------+-----------
Thomas Mueller | 25
Johanne Mueller | 20
Theodor Riegler | 35
Hubert Stoll | 30
Elisabeth Stark | 40
Lieschen Mueller | 15
Anna Schweiger | 50
```

c) (4 points)

```
CREATE OR REPLACE FUNCTION fCheapPocketBooks() RETURNS TRIGGER AS $$
BEGIN

        IF NEW.pages < 40 AND NEW.cost > 20 THEN
                INSERT INTO books VALUES (NEW.id, NEW.itemNumber,
                                          NEW.pages, NEW.cost / 2, NEW.fromAuthor);
                RETURN NULL;
        ELSE
                RETURN NEW;
        END IF;

END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER fCheapPocketBooks BEFORE INSERT ON books
       FOR EACH ROW EXECUTE PROCEDURE fCheapPocketBooks();
```

```
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (40, 3, 10, 60.00, 'Tolkien');
INSERT INTO books(id, itemNumber, pages, cost, fromAuthor) VALUES (15, 4, 30, 26.00, 'Wells');

SELECT fromAuthor, itemNumber, cost FROM books WHERE itemNumber >= 3;
```

```
fromauthor | itemNumber | cost
-----------+------------+------
Jelinek | 3 | 50.99
Tolkien | 3 | 15.00
Wells | 4 | 13.00
```

Punkteverteilung Aufgabe 6

(6a)

for correct entry (Wells | 100 | UK) - calls trigger once	+1.5
for correct entry (Swift | 375 | UK) - calls trigger once	+1.5
for correct entry (Christie | 1360 | UK) - does not call the trigger	+0.5
for correct entry (Tolkien | 550 | UK) - does not call the trigger	+0.5
	$\sum$ 4

(6b)

for correct entry (Elisabeth Stark | 40) - calls trigger once	+1.5
for correct entry (Anna Schweiger | 50) - calls trigger once	+1.5
for correct remaining entries - do not call the trigger	+1
	$\sum$ 4
for false schema	-0.5

(6c)

for correct entry (Tolkien | 3 | 15.00) - calls trigger two times	+2
for correct entry (Wells | 4 | 13.00) - calls trigger once	+1.5
for correct remaining entries - do not call the trigger	+0.5
	$\sum$ 4
If solely (Tolkien | 3 | 30.00) was added	-1
If (Tolkien | 3 | 60)and (Tolkien | 3 | 30) were added	-1

Overall: 70 points

Have a successful exam.

Instance for Task 5 and Task 6:

You may separate this page form the exam and keep this page.

Please do not write on this page! Solutions written on this sheet will not be graded!

readers

name	sid	country	mostBooks
Thomas Mueller	21	Austria	25
Johanne Mueller	23	United States	20
Anna Schweiger	22	Germany	10
Theodor Riegler	24	Germany	35
Hubert Stoll	26	Brazil	30
Elisabeth Stark	25	Denmark	40
Lieschen Mueller	27	Japan	15

books

id	itemNumber	pages	cost	fromAuthor
10	1	475	50.00	Goethe
10	2	305	60.50	Christie
15	1	255	75.00	Twain
15	2	660	40.50	Jelinek
15	3	220	50.99	Jelinek
30	1	430	60.00	Christie
35	1	95	44.50	Swift
20	1	115	50.00	Tolkien
25	1	435	60.95	Tolkien
40	1	230	70.00	Swift
40	2	625	75.90	Christie

deliveries

id	date	toName	toSid
10	24.11.2022	Anna Schweiger	22
15	15.11.2022	Lieschen Mueller	27
20	27.11.2022	Johanne Mueller	23
25	12.10.2020	Thomas Mueller	21
30	14.09.2021	Hubert Stoll	26
35	01.12.2022	Theodor Riegler	24
40	12.12.2020	Elisabeth Stark	25

authors

name	printedPages	country
Swift	325	United Kingdom
Goethe	475	Germany
Christie	1360	United Kingdom
Twain	255	United States
Wells	0	United Kingdom
Jelinek	880	Austria
Tolkien	550	United Kingdom