Korrekturrand

Die Handlungsschritte 1 bis 5 beziehen sich auf die folgende Ausgangssituation:

Sie sind Mitarbeiter/-in der BioScan GmbH, Astadt, einem Softwaredienstleister im Bereich Biometrie. Die BioScan GmbH erstellt Software zur Erfassung und Auswertung verschiedener biometrischer Daten.

Sie sollen vier der folgenden fünf Aufgaben erledigen:
1. Ein UML-Klassendiagramm erstellen
2. Eine Funktion zur Auswertung von Fingerabdrücken erstellen
3. Ein UML-Aktivitätsdiagramm erstellen
4. Ein ER-Diagramm erstellen
5. SQL-Anweisungen für eine Datenbank erstellen

1. Handlungsschritt (25 Punkte)

Die BioScan GmbH soll eine Software zur Erkennung und Speicherung von Fingerabdrücken und Handflächenabdrücken erstellen. Zur Vorbereitung der Programmierung soll ein UML-Klassendiagramm erstellt werden.

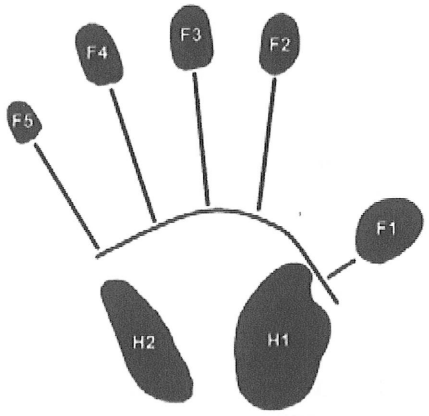
a) In einem UML-Klassendiagramm können die folgenden Beziehungen vorkommen.
Beschreiben Sie jeweils kurz

aa) Assoziation. 2 Punkte

ab) Vererbung. 2 Punkte

ac) Komposition. 2 Punkte

b) Für eine Person sollen von der linken und rechten Hand jeweils folgende Abdrücke gespeichert werden:



F1 bis F5: Abdrücke der fünf Finger
H1 und H2: Abdrücke der Handflächenbereiche

Zu jedem Abdruck sollen ein Bild und ein String gespeichert werden.

Die Zeichenkette enthält Beschreibungen derjenigen Merkmale des Abdrucks, die beim Vergleich von Fingerabdrücken verwendet werden.

Die Zeichenkette wird von der Methode *berechneZeichenkette()* anhand des Bildes berechnet.

Die Algorithmen zur Berechnung der Zeichenketten sind für Fingerabdruck und Handflächenabdruck unterschiedlich.

Es existiert bereits folgende Klasse *Abdruck*, die für das Klassendiagramm verwendet werden soll.

Abdruck
-: Bild -: String
+berechneZeichenkette()

Erstellen Sie auf der Folgeseite ein UML-Klassendiagramm, das …
- die Klassen *Person, Hand, Finger, Handflächenbereich, Abdruck, AbdruckFinger, AbdruckHandfläche* darstellt.
- die Beziehungen zwischen den Klassen mit ihren Kardinalitäten angibt.
- Geben Sie an, in welchen Klassen die Methode *berechneZeichenkette()* überschrieben werden muss. 19 Punkte

Hinweis: Notation zum UML-Klassendiagramm, siehe Seite 2 im Belegsatz

UML-Klassendiagramm

Korrekturrand

2. Handlungsschritt (25 Punkte)

Korrekturrand

Um herauszufinden, von welcher Person ein Fingerabdruck stammt, soll dieser mit Fingerabdrücken in einer Datenbank verglichen werden. Zu jedem in der Datenbank gefundenen Fingerabdruck wird ein Score ermittelt, der den Prozentsatz der Übereinstimmung angibt. Bei vollständiger Übereinstimmung beträgt der Score 100 %.
Die vorhandene Funktion *suche(abdruck)* gibt ein Array *matches* aus, das zu jedem gefunden Fingerabdruck einen Score, eine Personen-ID und eine Finger-ID enthält.

Die BioScan GmbH soll nun die Prozedur *auswertung* erstellen, die eine Fingerabdrucksuche durchführt und nur Daten der Fingerabdrücke ausgibt, deren Scores oberhalb eines bestimmten Schwellenwertes liegen.

Der Prozedur werden die folgenden drei Parameter übergeben

abdruck	Zeichenkette; Werte des Fingerabdruckbildes als Zeichenkette
schwelle	ganzzahliger Wert; Werte: 1 bis 100; gibt einen Score an, ab dem Fingerabdrücke aufgelistet werden sollen
finger	ganzzahliger Wert; 0 = Unbekannter Finger; 1 = Daumen rechts … 10 = Kleiner Finger links

Folgende Funktionen und Prozeduren sollen verwendet werden:

Funktion	**Beschreibung**
suche(abdruck)	Durchsucht die Datenbank nach Fingerabdrücken, die Übereinstimmungen (Matches) mit dem der Prozedur übergebenen Fingerabdruck aufweisen. Bei einem Match werden die Übereinstimmung in Prozent (score), die Personen-ID und die Finger-ID (1, 2 … 10) in einem Array vom Datentyp *Match* gespeichert: `Match: {score: Integer; idPerson: Integer; idFinger: Integer}.`
laenge(array)	Liefert die Länge des Arrays
loesche(array, position)	Löscht das Array-Element an der entsprechenden Position, die Array-Länge verkürzt sich dabei um 1. Das 1. Array-Element liegt an Position 0.

Zurückgegeben werden soll ein Array vom Datentyp *Match*:
- Das Array soll nur die Daten derjenigen Fingerabdrücke enthalten, deren Scores oberhalb des mit dem Übergabeparameter *schwelle* übergebenen Wertes liegen.
- Ist der Finger-Typ bekannt, von dem der Abdruck stammt (Übergabewerte *finger* = 1 bis 10), dann sollen nur Daten zu diesem Finger-Typ in das zurückzugebende Array übernommen werden, z. B. nur Daten zu rechten Zeigefingern (*idfinger* = 2), bei denen eine Übereinstimmung festgestellt wurde.
- Ist der Finger-Typ nicht bekannt, von dem der Abdruck stammt (Übergabewert *finger* = 0), dann sollen die Daten zu allen Finger-Typen (*idfinger* = 1 bis 10) in das zurückzugebende Array übernommen werden, bei denen eine Übereinstimmungen festgestellt wurde.
- Das Array soll nach Score absteigend sortiert sein. Der Sortieralgorithmus muss selbst erstellt werden.

Beispiel:

Array matches vom Typ Match, das von der Funktion *suche(abdruck)* erstellt wird:

score	idPerson	idFinger
85	93334	2
80	48774	1
98	56446	2
71	33961	10
21	73447	2
81	49982	2

Array, das von der Prozedur auswertung zurückgegeben werden soll.
Übergabewerte: *schwelle* = 80 und *finger* = 2

score	idPerson	idFinger
98	56446	2
85	93334	2
81	49982	2

Stellen Sie auf der Folgeseite den Algorithmus der Prozedur auswertung in Pseudocode oder in einem Struktogramm oder als Programmablaufplan dar.

```
auswertung(abdruck: String, schwelle: Integer, finger: Integer) : Match[]
```

Korrekturrand

3. Handlungsschritt (25 Punkte)

Korrekturrand

Die BioScan GmbH soll ein System zur Fingerabdruck-Recherche erstellen.

a) Zur Vorbereitung der Programmierung des Systems zur Recherche soll ein UML-Aktivitätsdiagramm erstellt werden.

Die Recherche im System soll wie folgt organisiert werden:

- Ein Auftraggeber schickt einen Fingerabdruck (FA) zur Identifizierung an den Operator.
- Der Operator prüft, ob die Qualität des FA in Ordnung ist.
- Ist die Qualität nicht ok, dann schickt der Operator eine entsprechende Information an den Auftraggeber und die Auftragsbearbeitung ist beendet.
- Ist die Qualität ok, dann führt der Operator eine Suche nach entsprechenden FAs durch.
- Werden keine FAs mit Übereinstimmungen gefunden, schickt der Operator eine entsprechende Info an den Auftraggeber und die Auftragsbearbeitung ist beendet.
- War die Suche erfolgreich, werden vom Operator parallel ein Report erstellt und die Auftragsdaten an den Supervisor verschickt.
- Der Supervisor protokolliert die Auftragsdaten und schickt eine Info an den Operator, dass die Daten protokolliert wurden.
- Nachdem der Report erstellt und die Info vom Supervisor verschickt wurden, versendet der Operator den Report an den Auftraggeber und die Auftragsbearbeitung ist beendet.

Stellen Sie auf der Folgeseite den geschilderten Ablauf in einem UML-Aktivitätsdiagramm dar. 20 Punkte
Hinweis: Notation zum UML-Aktivitätsdiagramm, siehe Seite 3 im Belegsatz.

b) Das Suchergebnis liegt im Array *matches* vor. Zu jedem im System gefundenen Fingerabdruck wird ein Score angegeben.

Array *matches*

score	idPerson
21	73447
85	93334
80	48774
98	56446
81	49982

Im Report soll eine Auswertung des Suchergebnisses ausgegeben werden. Dazu soll der minimale Score-Wert im Array *matches* ermittelt werden.

Beispiel:

Auswertung:
minimaler Score = 21

Stellen Sie den Algorithmus als Teil einer Prozedur in Pseudocode, in einem Struktogramm oder Programmablaufplan dar.
5 Punkte

Fortsetzung 3. Handlungsschritt →

4. Handlungsschritt (25 Punkte)

Korrekturrand

Die BioScan GmbH soll für eine Polizeikreisbehörde eine Datenbank erstellen, in der die Daten von Vorgängen erfasst werden, die bislang in folgender Excel-Tabelle gespeichert wurden. Die Namen der Beschuldigten sind geschwärzt.

Erfassung von Vorgängen

Vorgangs-ID	Delinquent-ID	Anrede	Delinquent	Geburtsdatum	Adresse	Delikt	Datum	Dokument	Bearbeiter
...									
301	5645	Herr	[redacted], [redacted]	28.02.1970	01234 AStadt Kernweg 12	Raub	02.04.2017	Personalausweis, Fahrerlaubnis	Hansen, Klaus
302	1213	Herr	[redacted], [redacted]	06.06.2000	02566 Bstadt Müller-Str. 1	Drogenmissbrauch	02.02.2014	Personalausweis	Müller, Marcel
303	7887	Herr	[redacted], [redacted]	01.07.1988	03669 Astadt Franzgasse 3	Fahrerflucht, Drogenmissbrauch	30.3.2017	Reisepass, Fahrerlaubnis	Hansen, Klaus
304	4545	Frau	[redacted], [redacted]	16.08.1991	02566 Bstadt Burgplatz 16	Drogenmissbrauch	12.4.2017	Personalausweis	Wagner, Wolfram
305	1213	Herr	[redacted], [redacted]	06.06.2000	02566 Bstadt Müller-Str. 1	Körperverletzung	08.03.2015	Personalausweis	Hansen, Klaus
...									

Erstellen Sie auf der Folgeseite für die geforderte Datenbank ein relationales Datenmodell in der dritten Normalform.

- Geben Sie den Tabellen und Attributen selbsterklärende Namen.
- Nennen Sie je Tabelle alle erforderlichen Attribute.
- Kennzeichnen Sie Primärschlüssel mit PK und Fremdschlüssel mit FK.
- Zeichnen Sie die Beziehungen mit deren Kardinalitäten ein.

Hinweis: Die Adresse des Delinquenten soll in diesem ersten Entwurf noch nicht normalisiert werden.

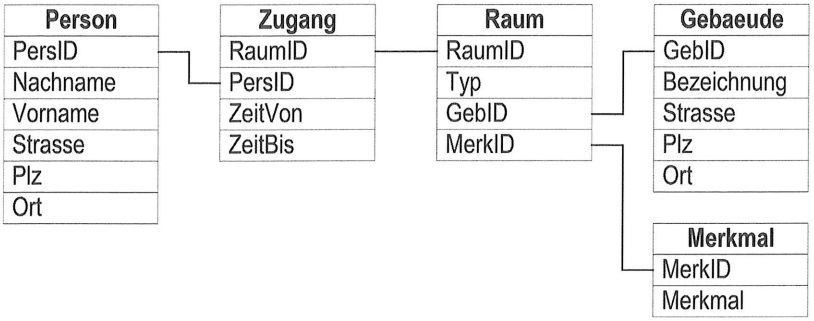
Korrekturrand

5. Handlungsschritt (25 Punkte)

Korrekturrand

Die BioScan GmbH entwickelt ein System zur Zugangskontrolle. Dazu wurde bereits folgende Datenbank entwickelt und mit Testdaten gefüllt.

Hinweis: SQL-Syntax, siehe Seiten 4 und 5 im Belegsatz



Person

PersID	Nachna-me	Vorname	Strasse	Plz	Ort
101	Müller	Max	Müllerweg 1	52335	Köln
202	Meier	Willi	Testweg 12	43333	Dortmund
404	Wester	Klaus	Hauptstr. 13	55667	Köln

Zugang

RaumID	PersID	ZeitVon	ZeitBis
1	101	08:00	10:00
1	202	10:00	14:00
2	101	14:00	18:00
5	202	08:00	18:00

Raum

RaumID	RaumTyp	GebID	MerkID
1	Besprechungsraum	2	1
2	Labor	2	2
3	Labor	1	2
4	Labor	1	2
5	Besprechungsraum	1	1
6	Labor	3	2
7	Labor	3	2
8	Labor	3	2

Gebaeude

GebID	Bezeichnung	Strasse	Plz	Ort
1	Forschung H	Heinrich-Hertz-Str. 12	50501	Köln
2	Forschung U	Heinrich-Hertz-Str. 14	50501	Köln
3	Forschung I	Heinrich-Hertz-Str. 16	50501	Köln
4	Verwaltung	Transalee 22	50555	Köln

Merkmal

MerkID	Merkmal
1	Fingerabdruck
2	Iris

Erstellen Sie die SQL-Anweisungen für folgende Ausgaben:

a) Liste aller Gebäude mit deren Räumen jeweils aufsteigend sortiert nach Gebäudebezeichnung und Raumtyp. 5 Punkte

GebID	Bezeichnung	Strasse	Plz	Ort	RaumID	RaumTyp	GebID	MerkID
1	Forschung H	Heinrich-Hertz-Str. 12	50501	Köln	5	Besprechungsraum	1	1
1	Forschung H	Heinrich-Hertz-Str. 12	50501	Köln	3	Labor	1	2
1	Forschung H	Heinrich-Hertz-Str. 12	50501	Köln	4	Labor	1	2
3	Forschung I	Heinrich-Hertz-Str. 16	50501	Köln	6	Labor	3	2
3	Forschung I	Heinrich-Hertz-Str. 16	50501	Köln	7	Labor	3	2
3	Forschung I	Heinrich-Hertz-Str. 16	50501	Köln	8	Labor	3	2
2	Forschung U	Heinrich-Hertz-Str. 14	50501	Köln	1	Besprechungsraum	2	1
2	Forschung U	Heinrich-Hertz-Str. 14	50501	Köln	2	Labor	2	2
4	Verwaltung	Transallee 22	50555	Köln	NULL	NULL	NULL	NULL

b) Liste aller Daten, die in der Tabelle Zugang gespeichert sind und die dazugehörigen Personendaten
Siehe Ergebnis: 5 Punkte

PersID	Nachname	Vorname	Straße	Plz	Ort	RaumID	PersID	ZeitVon	ZeitBis
NULL	NULL	NULL	NULL	NULL	NULL	1	101	08:00	10:00
NULL	NULL	NULL	NULL	NULL	NULL	1	101	14:00	18:00
101	Müller	Max	Müllerweg 1	52335	Köln	2	202	10:00	14:00
101	Müller	Max	Müllerweg 1	52335	Köln	5	202	08:00	18:00

c) Anzahl der Räume, die bei der Zugangskontrolle das Merkmal Fingerabdruck beziehungsweise das Merkmal Iris prüfen
Siehe Ergebnis: 6 Punkte

Merkmal	AnzahlRaeume
Fingerabdruck	2
Iris	6

d) Liste der Zugangsdaten von Max Müller
Hinweis: Es ist nur der Name, nicht die PersID bekannt.
Siehe Ergebnis: 6 Punkte

Nachname	Vorname	RaumID	ZeitVon	ZeitBis
Müller	Max	1	10:00	14:00
Müller	Max	5	08:00	18:00

Korrekturrand

Fortsetzung 5. Handlungsschritt →

Fortsetzung 5. Handlungsschritt

Korrekturrand

e) Liste mit allen Personen aus dem PLZ-Gebiet 50000 bis 59999
Siehe Ergebnis:

3 Punkte

PersID	Nachname	Vorname	Straße	Plz	Ort
101	Müller	Max	Müllerweg 1	52335	Köln
404	Wester	West	Hauptstr. 13	55667	Köln

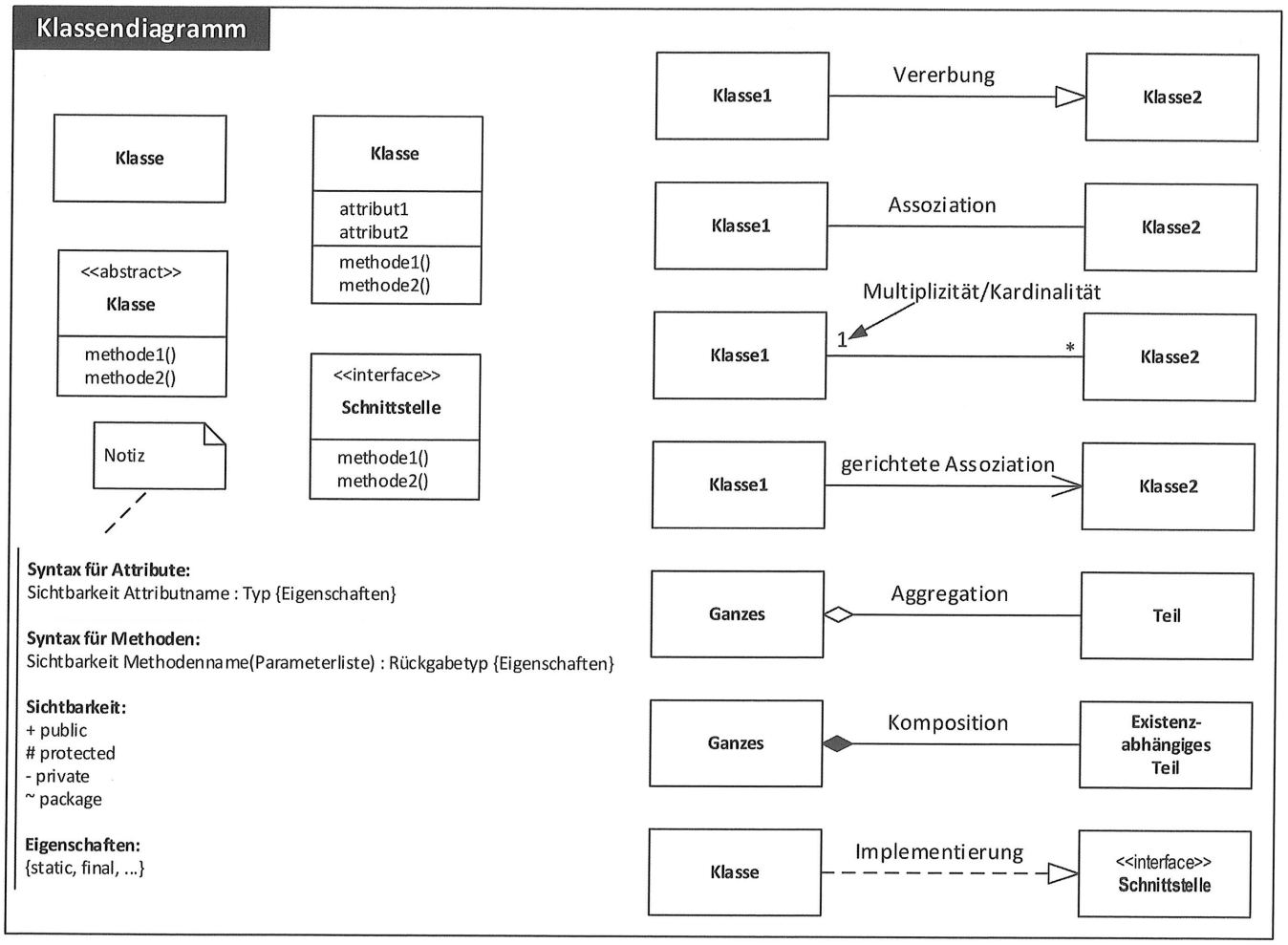
PRÜFUNGSZEIT – NICHT BESTANDTEIL DER PRÜFUNG!

Wie beurteilen Sie nach der Bearbeitung der Aufgaben die zur Verfügung stehende Prüfungszeit?

[1] Sie hätte kürzer sein können. [2] Sie war angemessen. [3] Sie hätte länger sein müssen.

UML-Klassendiagramm, Notation (Auszug)

für 1. Handlungsschritt, Teilaufgabe b)



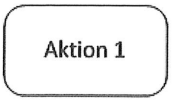
UML-Aktivitätsdiagramm, Notation (Auszug)

für 3. Handlungsschritt, Teilaufgabe a)

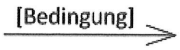
Startknoten: Markiert den Beginn eines Ablaufs

Ablaufende: Markiert das Ende eines Zweiges

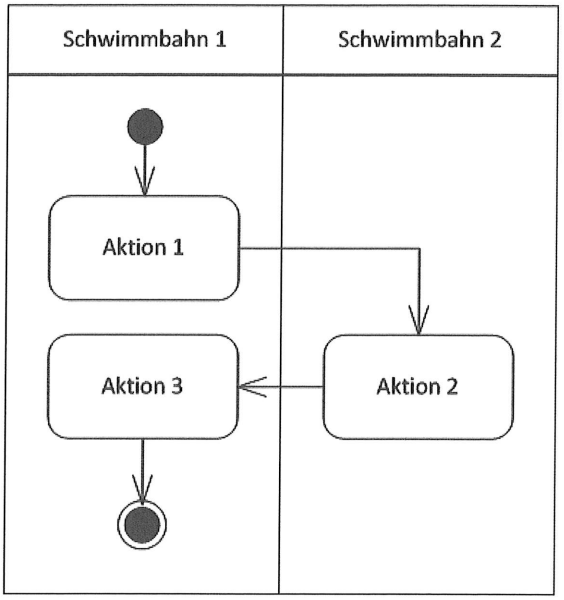
Endknoten: Markiert das Ende eines Ablaufs



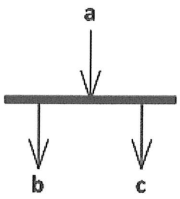
Aktion: Verhalten, das eine Veränderung herbeiführt



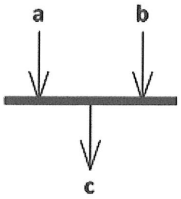
Pfeil: Beschreibt den Fluss von Aktion zu Aktion. Angabe einer Bedingung in eckiger Klammer möglich.



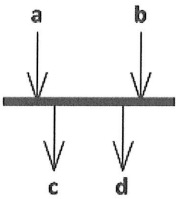
Teilung (Splitting):
Nach dem Ende einer Aktion werden mehrerer Aktionen begonnen.



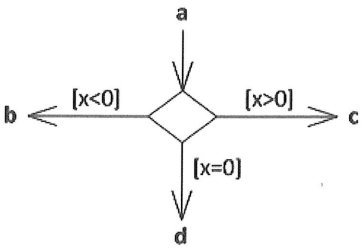
Synchronisation (Und):
Nach dem Ende mehrerer Aktionen wird eine Aktionen begonnen.



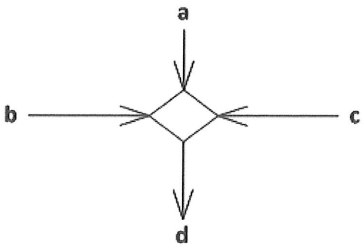
Synchronisation und Teilung:
Nach dem Ende mehrerer Aktionen wird mit mehreren Aktionen begonnen.



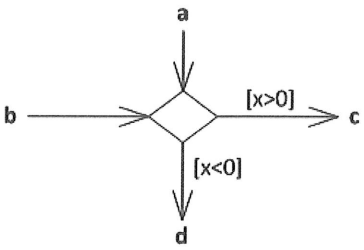
Entscheidung:
Nach der Aktion a wird entweder die Aktion b, c oder d ausgeführt.



Zusammenführung (Oder):
Nach der Aktion a, b oder c wird die Aktion d ausgeführt.



Zusammenführung und Entscheidung
Nach der Aktion a oder b wird entweder die Aktion c oder d ausgeführt.



Stand 2016-02-04

SQL-Syntax (Auszug)

für 5. Handlungsschritt

Syntax	Beschreibung
Tabelle	
CREATE TABLE Tabellenname(Feldname < DATENTYP >, Primärschlüssel, Fremdschlüssel)	Erzeugt eine neue leere Tabelle mit der beschriebenen Struktur
ALTER TABLE Tabellenname **ADD COLUMN** Spaltenname Datentyp **DROP COLUMN** Spaltenname	Änderungen in einer Tabelle: Hinzufügen einer Spalte Entfernen einer Spalte
CHARACTER	Textdatentyp
DECIMAL	Numerischer Datentyp (Festkommazahl)
DOUBLE	Numerischer Datentyp (Doppelte Präzision)
INTEGER	Numerischer Datentyp (Ganzzahl)
DATE	Datum (Format DD.MM.YYYY)
PRIMARY KEY (Feldname)	Erstellung eines Primärschlüssels
FOREIGN KEY (Feldname) **REFERENCES**	Erstellung von Fremdschlüssel-Beziehungen
DROP TABLE Tabellenname	Löscht eine Tabelle
Befehle, Klauseln, Attribute	
SELECT * | Feldname1 [, Feldname2, ...]	Wählt die Spalten einer oder mehrerer Tabellen, deren Inhalte in die Liste aufgenommen werden sollen; alle Spalten (*) oder die namentlich aufgeführten
FROM	Name der Tabelle oder Namen der Tabellen, aus denen die Daten der Ausgabe stammen sollen
SELECT ... (**SELECT** ... **FROM** ... **WHERE** ...) **AS** xyz **FROM** ... **WHERE** ...	Unterabfrage, die in eine äußere SELECT-Anweisung geschachtelt ist. Das Ergebnis der Unterabfrage wird im Spaltenausdruck (z. B. hier: *xyz)* ausgegeben.
INNER JOIN	Liefert nur die Datensätze zweier Tabellen, die gleiche Datenwerte enthalten
LEFT JOIN / Left OUTER JOIN	Liefert von der erstgenannten (linken) Tabelle alle Datensätze und von der zweiten Tabelle jene, deren Datenwerte mit denen der ersten Tabelle übereinstimmen
RIGHT JOIN / RIGHT OUTER JOIN	Liefert von der zweiten (rechten) Tabelle alle Datensätze und von der ersten Tabelle jene, deren Datenwerte mit denen der zweiten Tabelle übereinstimmen
FULL JOIN	Liefert aus beiden Tabellen jeweils alle Datensätze
WHERE	Bedingung, nach der Datensätze ausgewählt werden sollen
WHERE EXISTS (subquery) **WHERE NOT EXISTS** (subquery)	Die Bedingungen EXISTS prüft, ob die Suchbedingung einer Unterabfrage mindestens eine Zeile zurückliefert. NOT EXIST negiert die Bedingung.
GROUP BY Feldname1 [,Feldname2, ...]	Gruppierung (Aggregation) nach Inhalt des genannten Feldes
ORDER BY Feldname1 [,Feldname2, ...] **ASC | DESC**	Sortierung nach Inhalt des genannten Feldes oder der genannten Felder ASC: aufsteigend; DESC: absteigend
Datenmanipulation	
DELETE FROM Tabellenname	Löschen von Datensätzen in der genannten Tabelle
UPDATE Tabellenname **SET**	Aktualisiert Daten in Feldern einer Tabelle
INSERT INTO Tabellenname **VALUES** (Wert für Spalte 1 [, Wert für Spalte 2, ...]) oder **SELECT** ... **FROM** ... **WHERE**	Fügt Datensätze in die genannte Tabelle, die entweder mit festen Werten belegt oder Ergebnis eines SELECT-Befehls sind

Fortsetzung ->

SQL-Syntax (Auszug) – Fortsetzung

Syntax	Beschreibung
Aggregatfunktionen	
AVG(Feldname)	Ermittelt das arithmetische Mittel aller Werte im angegebenen Feld
COUNT(Feldname | *)	Ermittelt die Anzahl der Datensätze mit Nicht-NULL-Werten im angegebenen Feld oder alle Datensätze der Tabelle (dann mit Operator *)
SUM(Feldname | Formel)	Ermittelt die Summe aller Werte im angegebenen Feld oder der Formelergebnisse
MIN(Feldname | Formel)	Ermittelt den kleinsten aller Werte im angegebenen Feld
MAX (Feldname | Formel)	Ermittelt den größten aller Werte im angegebenen Feld
Funktionen	
LEFT(Zeichenkette, Anzahlzeichen**)**	Liefert *Anzahlzeichen* der Zeichenkette von links.
RIGHT(Zeichenkette, Anzahlzeichen**)**	Liefert *Anzahlzeichen* der Zeichenkette von rechts.
CURRENT	Liefert das aktuelle Datum mit der aktuellen Uhrzeit
CONVERT(time,[DatumZeit**])**	Liefert die Uhrzeit aus einer DatumZeit-Angabe
DATE(Wert**)**	Wandelt einen Wert in ein Datum um
DAY(Datum**)**	Liefert den Tag des Monats aus dem angegebenen Datum
MONTH(Datum**)**	Liefert den Monat aus dem angegebenen Datum
TODAY	Liefert das aktuelle Datum
WEEKDAY(Datum**)**	Liefert den Tag der Woche aus dem angegebenen Datum
YEAR(Datum**)**	Liefert das Jahr aus dem angegebenen Datum
DATEADD(Datumsteil, Intervall, Datum)	Fügt einem Datum ein Intervall (ausgedrückt in den unter Datumsteil angegebenen Einheiten) hinzu
DATEDIFF(Datumsteil, Anfangsdatum, Enddatum) Datumsteile: **DAY, MONTH, YEAR**	Liefert Enddatum-Startdatum (ausgedrückt in den unter Datumsteil angegebenen Einheiten)
Operatoren	
AND	Logisches UND
LIKE	Überprüfung von Textattributen auf Gleichheit, Verwendung von Platzhaltern möglich.
NOT	Logische Negation
OR	Logisches ODER
=	Test auf Gleichheit
>, >=, <, <=, < >	Test auf Ungleichheit
*****	Multiplikation
/	Division
+	Addition, positives Vorzeichen
-	Subtraktion, negatives Vorzeichen

Stand 2016-06-17