Diese Kopfleiste bitte unbedingt ausfüllen!

Familienname, Vorname (bitte durch eine Leerspalte trennen)

Bereich	Berufsnummer	IHK-Nummer	Prüflingsnummer
5 5	1 1 9 6		
Sp. 1-2	Sp. 3-6	Sp. 7-9	Sp. 10-14

IHK

Termin: Mittwoch, 23. November 2016

Abschlussprüfung Winter 2016/17

1196

1 Ganzheitliche Aufgabe I
Fachqualifikationen

Fachinformatiker
Fachinformatikerin
Anwendungsentwicklung

5 Handlungsschritte
mit Belegsatz
90 Minuten Prüfungszeit
100 Punkte

Bearbeitungshinweise

1. Der vorliegende Aufgabensatz besteht aus insgesamt 5 Handlungsschritten zu je 25 Punkten.

   In der Prüfung zu bearbeiten sind 4 Handlungsschritte, die vom Prüfungsteilnehmer frei gewählt werden können.

   Der nicht bearbeitete Handlungsschritt ist durch Streichung des Aufgabentextes im Aufgabensatz und unten mit dem Vermerk „Nicht bearbeiteter Handlungsschritt: Nr. ... „ an Stelle einer Lösungsniederschrift deutlich zu kennzeichnen. Erfolgt eine solche Kennzeichnung nicht oder nicht eindeutig, gilt der 5. Handlungsschritt als nicht bearbeitet.
2. Füllen Sie zuerst die **Kopfzeile** aus. Tragen Sie Ihren Familiennamen, Ihren Vornamen und Ihre Prüflings-Nr. in die oben stehenden Felder ein.
3. Lesen Sie bitte den **Text** der Aufgaben ganz durch, bevor Sie mit der Bearbeitung beginnen.
4. Halten Sie sich bei der Bearbeitung der Aufgaben genau an die **Vorgaben der Aufgabenstellung** zum Umfang der Lösung. Wenn z. B. vier Angaben gefordert werden und Sie sechs Angaben anführen, werden nur die ersten vier Angaben bewertet.
5. Tragen Sie die frei zu formulierenden **Antworten dieser offenen Aufgabenstellungen** in die dafür lt. Aufgabenstellung vorgesehenen Bereiche (Lösungszeilen, Formulare, Tabellen u. a.) des Arbeitsbogens ein.
6. Sofern nicht ausdrücklich ein Brief oder eine Formulierung in ganzen Sätzen gefordert werden, ist eine **stichwortartige Beantwortung** zulässig.
7. Schreiben Sie deutlich und gut lesbar. Ein nicht eindeutig zuzuordnendes oder **unleserliches Ergebnis** wird als **falsch** gewertet.
8. Zur Lösung der Rechenaufgaben darf ein nicht programmierter, netzunabhängiger **Taschenrechner** ohne Kommunikationsmöglichkeit mit Dritten verwendet werden.
9. Wenn Sie ein **gerundetes Ergebnis** eintragen und damit weiterrechnen müssen, rechnen Sie (auch im Taschenrechner) nur mit diesem gerundeten Ergebnis weiter.
10. Für **Nebenrechnungen/Hilfsaufzeichnungen** können Sie das im Aufgabensatz enthaltene Konzeptpapier verwenden. Dieses muss vor Bearbeitung der Aufgaben herausgetrennt werden. Bewertet werden jedoch nur Ihre Eintragungen im Aufgabensatz.

Nicht bearbeiteter Handlungsschritt ist Nr.

Wird vom Korrektor ausgefüllt!

Bewertung

Für die Bewertung gilt die Vorgabe der Punkte in den Lösungshinweisen. Für den abgewählten Handlungsschritt ist anstatt der Punktzahl die Buchstabenkombination „AA" in die Kästchen einzutragen.

	Punkte 1. Handlungsschritt	Punkte 2. Handlungsschritt	Punkte 3. Handlungsschritt	Punkte 4. Handlungsschritt	Punkte 5. Handlungsschritt
Spalte 1 - 14 s. o.	15 16	17 18	19 20	21 22	23 24

Gesamtpunktzahl 26 27 28

Prüfungszeit 25

Die entsprechende Ziffer (1, 2 oder 3) finden Sie in der Abfrage nach der Prüfungszeit im Anschluss an die letzte Aufgabe.

Prüfungsort, Datum

Unterschrift

Gemeinsame Prüfungsaufgaben der Industrie- und Handelskammern. Dieser Aufgabensatz wurde von einem überregionalen Ausschuss, der entsprechend § 40 Berufsbildungsgesetz zusammengesetzt ist, beschlossen.
Die Vervielfältigung, Verbreitung und öffentliche Wiedergabe der Prüfungsaufgaben und Lösungen ist nicht gestattet. Zuwiderhandlungen werden zivil- und strafrechtlich (§§ 97 ff., 106 ff. UrhG) verfolgt. – © ZPA Nord-West 2016 – Alle Rechte vorbehalten!

Korrekturrand

Die Handlungsschritte 1 bis 5 beziehen sich auf die folgenden Ausgangssitution:

Sie sind Mitarbeiter/-in der Ansowa KG, Astadt.

Die Ansowa KG entwickelt Individualsoftware und passt Standardsoftware an.

Sie wurde von der Wings GmbH, einer kleinen Fluggesellschaft, mit verschiedenen Softwareentwicklungen beauftragt.

Sie sollen im Rahmen des Projekts vier der folgenden fünf Aufgaben erledigen:
1. UML-Aktivitätsdiagramm
2. Prozedurale Programmierung
3. Prozedurale Programmierung
4. ER-Diagramm
5. SQL

1. Handlungsschritt (25 Punkte)

Die Wartung der Flugzeuge der Wings GmbH erfolgt in vorgeschriebenen Wartungsintervallen.
- nach 600 Flugstunden einfache Wartung (A-Service) im Hangar der Wings GmbH
- nach 3.000 Flugstunden umfassende Wartung (B-Service) im Wartungshangar der Werkstatt

Eine Wartung wird wie folgt abgewickelt:
- Die Werkstatt ermittelt anhand der Flugstunden ein zur Wartung anstehendes Flugzeug.
- Die Werkstatt reserviert die Zeit, für die das Flugzeug aus dem Flugbetrieb genommen werden muss.
- Der Disponent der Wings GmbH stimmt dem Termin zu oder lehnt ihn ab.
- Wenn der Disponent den Termin abgelehnt hat, dann schlägt die Werkstatt einen neuen Termin vor.
- Wenn der der Disponent der Wings GmbH dem Termin zugestimmt hat, dann veranlasst die Werkstatt die Wartung.
- Wenn ein B-Service durchgeführt werden muss, dann veranlasst die Werkstatt zusätzlich die Überführung des Flugzeugs in den Wartungshangar.
- Stellt die Werkstatt während der Wartungsarbeiten einen Defekt fest, dann wird für den Disponenten ein Reparaturangebot erstellt und die Wartung abgebrochen.
- Nach Abschluss oder Abbruch der Wartungsarbeiten erstellt die Werkstatt eine Rechnung.
- Bei einem B-Service wird die Rückführung des Flugzeugs zum Heimatflughafen parallel zur Rechnungserstellung durchgeführt.

Erstellen Sie auf der Folgeseite das entsprechende UML-Aktivitäts-Diagramm.

Hinweis: Notation für UML-Aktivitäts-Diagramm, siehe Belegsatz, Seite 2

Werkstatt	Disponent

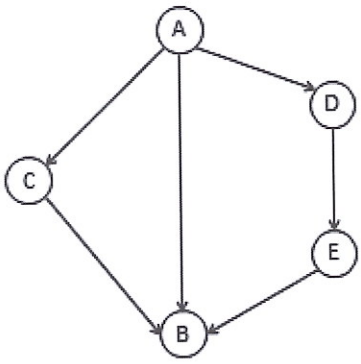
Korrekturrand

Korrekturrand

2. Handlungsschritt (25 Punkte)

Die Wings GmbH führt Frachtflüge zwischen mehreren Orten durch.

So können beispielsweise Routen von A nach B (ggf. unter Einbeziehung von Zwischenstopps) auf unterschiedliche Weise durchgeführt werden:



Die Wings GmbH benötigt nun eine Funktion, mit der die kostengünstigste Route zu vorgegebenen Startorten und Zielorten unter Berücksichtigung von freien Frachtkapazitäten ermittelt werden kann.

Die möglichen Routen zwischen Ausgangsort (A) und Zielort (B) liegen stets in einer zweidimensionalen Tabelle *Routen* vor (Inhalt beispielhaft für die obige Grafik).

Routen für einen Flug von A nach B

A	B			direkt über die Strecke A-B
A	C	B		über die Strecken A-C und C-B
A	D	E	B	über die Strecken A-D, D-E und E-B

Der Ausgangsort steht immer in Spalte 0, der Zielort ist immer der letzte nicht leere Eintrag einer Zeile.

Folgende Funktionen stehen zur Verfügung

holeStreckeGewicht(SB:String, SE:String) : Integer	Holt für eine Strecke die freie Frachtkapazität in kg
holeStreckePreis(SB:String, SE: String) : Double	Holt für eine Strecke die Frachtkosten EUR/kg

SB = Strecken-Beginn
SE = Strecken-Ende

Erstellen Sie einen Algorithmus für eine Funktion *findeRoute(gewicht: Integer) : Integer,* die für eine Fracht mit dem Gewicht *gewicht* die kostengünstigste Route unter Berücksichtigung von freien Frachtkapazitäten ermittelt.

Die Funktion soll den Index der Tabellenzeile zurückgeben, in der die günstigste Route gefunden wurde.

Stellen Sie auf der Folgeseite den Algorithmus in Pseudocode oder einem Struktogramm oder einem PAP dar.

```
findeRoute(Gewicht: Integer) : Integer
```

Korrekturrand

Korrekturrand

3. Handlungsschritt (25 Punkte)

Die Ansowa KG soll für die Wings GmbH eine Funktion entwickeln, mit der Flüge ausgewählt werden können
- die an einem bestimmten Tag durchgeführt werden
- und die mindestens eine gewünschte Anzahl freie Plätze bieten.

Die Daten eines Fluges sind in einem *Flug*-Objekt enthalten. Die entsprechende Klasse *Flug* wurde bereits entwickelt. Für jedes Attribut der Klasse *Flug* liegen öffentliche set-/get–Methoden vor, z. B. *getFlugDatum() und getPreis()*.

Flug

- id : String
- flugDatum : Date
- abflugZeit : Date
- ankunftZeit : Date
- preis : Double
- freiePlaetze : Integer
...

Das Array *Linien_Fluege* enthält die *Flug-Objekte* aller Flüge, die auf einer bestimmten Strecke nonstop durchgeführt werden. Das gewünschte Flugdatum und die Anzahl der benötigten Plätze werden der neuen Funktion als Parameter übergeben.

Die neue Funktion
- soll aus dem Array *Linien_Fluege* die *Flug*-Objekte derjenigen Flüge auswählen, welche am gewünschten Datum durchgeführt werden und die benötigten Plätze bieten.
- soll die ausgewählten *Flug*-Objekte in einem neuen eindimensionalen Array *Auswahl_Fluege* speichern. Die *Flug*-Objekte sollen nach Preis aufsteigend sortiert sein.
- soll eine Referenz auf das Array *Auswahl_Fluege* zurückgeben.

Erstellen Sie eine entsprechende Funktion *erstelleFluege(Datum : Date, Plaetze : Integer) : Flug[ ]*.
Stellen Sie auf der Folgeseite den erforderlichen Algorithmus in Pseudocode oder einem Struktogramm oder einem PAP dar.

Fortsetzung 3. Handlungsschritt →

Fortsetzung 3. Handlungsschritt

```
erstelleFluege(Datum : Date, Plaetze : Integer) : Flug[ ]
```

Korrekturrand

4. Handlungsschritt (25 Punkte)

Korrekturrand

Die durchgeführten Flüge sollen in einem relationalen Datenbanksystem erfasst werden.

Für die Entwicklung des Datenbanksystems liegt folgende Beschreibung vor:
- Auf einem Frachtflug werden Frachten, die von einem oder mehreren Kunden stammen, transportiert.
- Die Fracht eines Kunden kann auf einen oder mehrere Flüge verteilt werden.
- Ein Flug wird mit einem Flugzeug ausgeführt.
- Ein Flugzeug wird für viele Flüge eingesetzt.
- Ein Flugzeug wird von verschiedenen Piloten geflogen.
- Piloten können auf verschiedenen Flugzeugen eingesetzt werden.
- Ein Pilot führt viele Flüge durch.
- Ein Flug wird von zwei Piloten ausgeführt.

Erstellen Sie ein entsprechendes ER-Diagramm ohne Attribute.

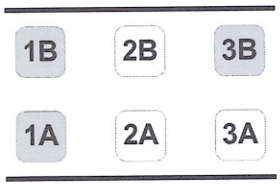
5. Handlungsschritt (25 Punkte)

Korrekturrand

Die Ansowa KG wurde von der Wings GmbH beauftragt, verschiedene SQL-Anweisungen zur Auswertung folgender Datenbank zu erstellen.

Beispiel zur Darstellung der Zusammenhänge:
Der Flug mit der ID *1* von Norddeich nach Langeoog, Abflug *09:00* Uhr wird am *02.12.2016* unter der Flug-Datum-ID *521* durchgeführt.
Für den Flug mit der Flug-Datum-ID *521* wird das Flugzeug mit der ID *14* eingesetzt, das sechs Sitzplätze hat; drei Plätze wurden von den Passagieren Karl Müller (Platz 1A), Lisa Müller (1B) und Paula Ruge (3B) gebucht.

Flug 1 am 02.12.2016 (Flug-Datum-ID 521)
Sitzplan Flugzeug ID 14 und Reservierungen



Flugzeug_Sitzplan

Sitz_ID	Flugzeug_ID	Platz
1	14	1A
2	14	1B
3	14	2A
4	14	2B
5	14	3A
6	14	3B
7	15	1A
...		

Flug

Flug_ID	von	nach	Abflug	Ankunft	Preis
1	Norddeich	Langeoog	09:00	09:20	60,00
2	Langeoog	Norddeich	10:00	10:20	40,00
3	Norddeich	Langeoog	11:00	11:20	50,00
4	Langeoog	Norddeich	12:00	12:20	40,00
5	Norddeich	Langeoog	15:00	15:20	50,00
6	Langeoog	Norddeich	16:00	16:20	40,00
7	Norddeich	Langeoog	17:00	16:20	50,00
8	Langeoog	Norddeich	18:00	18:20	50,00

Flug_Datum

Flug_Datum_ID	Flug_ID	Datum	Flugzeug_ID
...			
521	1	02.12.2016	14
522	2	02.12.2016	14
...			
528	8	02.12.2016	14
...			
693	2	15:12.2016	15
...			

Hinweis: Das Feld Datum ist vom Typ String.

Buchung

Buchung_ID	Flug_Datum_ID	Passagier_ID	Platz
...			
1265	521	877	1B
1266	521	878	1A
1267	693	877	2A
1268	693	878	2B
1269	521	1324	3B
1270	528	1324	1A
...			

Passagier

Passagier_ID	Anrede	Name	Vorname
...			
877	Frau	Müller	Lisa
878	Herr	Müller	Karl
...			
1324	Frau	Ruge	Paula
...			
1500	Herr	Zöller	Martin
...			

Korrekturrand

Erstellen Sie für folgende Auswertungen jeweils die entsprechende SQL-Anweisung.

Hinweis: SQL-Syntax, siehe Belegsatz, Seite 3 und 4

a) Für den Flug mit der ID *1* am *02.12.2016*:

Aktuelle Passagierliste nach Plätzen aufsteigend sortiert — 8 Punkte

Hinweis:
- Das Feld *Datum* ist vom Typ String.
- Auswahlkriterien sind Flug-ID *1* und das Datum *02.12.2016*.

Beispiel

Platz	Name	Vorname
1A	Müller	Lisa
1B	Müller	Karl
3B	Ruge	Paula

b) Prozentualer Anteil der Männer an der Gesamtheit aller Passagiere, die bisher mit der Fluglinie geflogen sind — 5 Punkte

Hinweis: Es wird davon ausgegangen, dass der Divisor immer > 0 ist.

Beispiel:

AnteilMaenner
50

c) Der Umsatz eines Tages der Flüge von Norddeich nach Langeoog unter der Annahme, dass bei jedem Flug vier Passagiere transportiert werden. — 5 Punkte

Beispiel:

Tagesumsatz
840,00 EUR

Fortsetzung 5. Handlungsschritt →

Fortsetzung 5. Handlungsschritt

Korrekturrand

d) Liste der freien Plätze für den Flug mit der Flug-Datum-ID 521.

Beispiel:

Platz
2A
2B
3A

Die dazu erforderliche SQL-Anweisung wurde bereits begonnen.

Ergänzen Sie dazu folgenden Entwurf. 7 Punkte

Hinweis:
- Es sollen nur die Tabellen *Flugzeug_Sitzplan* und *Buchung* verwendet werden.
- Die Bedingung NOT EXISTS ist dann erfüllt, wenn in der folgenden Unterabfrage die WHERE-Bedingungen **nicht** zutreffen.

SELECT Flugzeug_Sitzplan.Platz

FROM Flugzeug_Sitzplan

WHERE Flugzeug_Sitzplan.Flugzeug_ID = 14

AND NOT EXISTS

(SELECT *

FROM Buchung

WHERE Buchung.Flug_Datum_ID =[1] 521

AND Buchung.Platz =[1] Flugzeug_Sitzplan.Platz)

1) Operator einsetzen

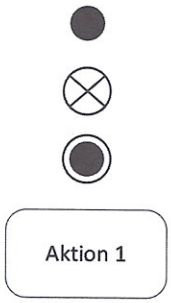
PRÜFUNGSZEIT – NICHT BESTANDTEIL DER PRÜFUNG!

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-Aktivitätsdiagramm, Notation (Auszug)

für 1. Handlungsschritt



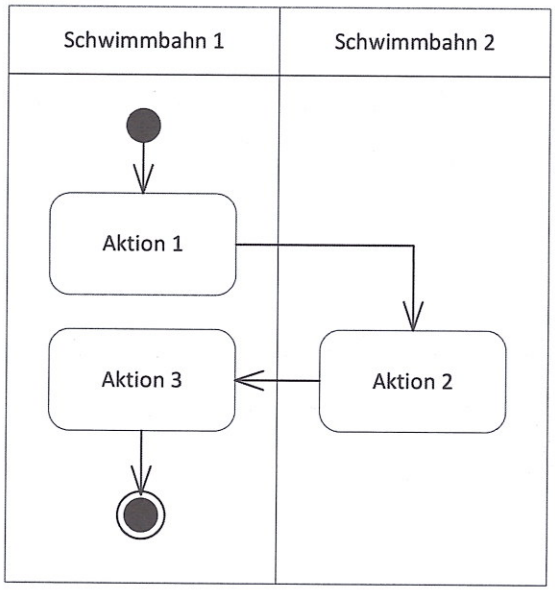
Startknoten: Markiert den Beginn eines Ablaufs

Ablaufende: Markiert das Ende eines Zweiges

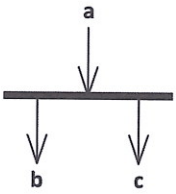
Endknoten: Markiert das Ende eines Ablaufs

Aktion: Verhalten, das eine Veränderung herbeiführt

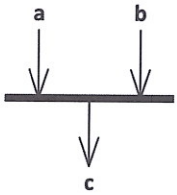
Pfeil: Beschreibt den Fluss von Aktion zu Aktion. Angabe einer Bedingung in eckiger Klammer möglich.



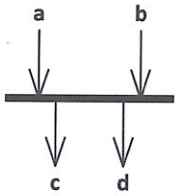
Teilung (Splitting):
Nach dem Ende einer Aktion werden mehrerer Aktionen begonnen.



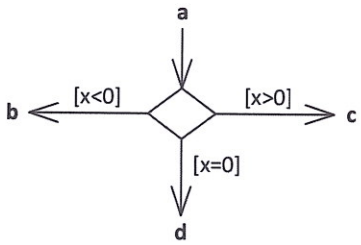
Synchronisation (Und):
Nach dem Ende mehrerer Aktionen wird eine Aktionen begonnen.



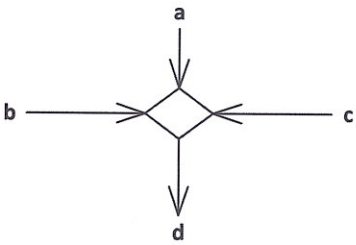
Synchronisation und Teilung:
Nach dem Ende mehrerer Aktionen wird mit mehreren Aktionen begonnen.



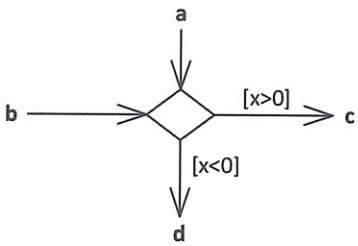
Entscheidung:
Nach der Aktion a wird entweder die Aktion b, c oder d ausgeführt.



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.



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 ode 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