

Dieses Blatt kann an der Perforation aus dem Aufgabensatz herausgetrennt werden!

SQL-Syntax

Syntax	Beschreibung
Tabelle	
CREATE TABLE Tabellenname(Spaltenname < DATENTYP >, Primärschlüssel, Fremdschlüssel)	Erzeugt eine neue leere Tabelle mit der beschriebenen Struktur
ALTER TABLE Tabellenname ADD COLUMN Spaltenname Datentyp DROP COLUMN Spaltenname Datentyp	Änderungen an einer Tabelle: Hinzufügen einer Spalte Entfernen einer Spalte
ADD FOREIGN KEY(Spaltenname) REFERENCES Tabellenname(Primärschlüsselpaltenname)	Definiert eine Spalte als Fremdschlüssel
CHARACTER	
DECIMAL	Textdatentyp
DOUBLE	Numerischer Datentyp (Festkommazahl)
INTEGER	Numerischer Datentyp (Doppelte Präzision)
DATE	Numerischer Datentyp (Ganzzahl)
PRIMARY KEY (Spaltenname)	Datum (Format DD.MM.YYYY)
FOREIGN KEY (Spaltenname) REFERENCES Tabellenname(Primärschlüsselpaltenname)	Erstellung eines Primärschlüssels Erstellung einer Fremdschlüssel-Beziehung
DROP TABLE Tabellenname	Löscht eine Tabelle
Befehle, Klauseln, Attribute	
SELECT * Spaltenname1 [, Spaltenname2, ...]	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 ... FROM ... (SELECT ... FROM ... WHERE ...) AS tbl WHERE ...	Unterabfrage (subquery), die in eine äußere Abfrage eingebettet ist. Das Ergebnis der Unterabfrage wird wie eine Tabelle – hier mit Namen "tbl" – behandelt.
SELECT DISTINCT	Eliminiert Redundanzen, die in einer Tabellen auftreten können, Werte werden jeweils nur einmal angezeigt.
JOIN / INNER JOIN	Liefert nur die Datensätze zweier Tabellen, die gleiche Datenwerte enthalten
LEFT JOIN / LEFT OUTER JOIN	Liefert von der ersten genannten (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
WHERE	Bedingung, nach der Datensätze ausgewählt werden sollen
WHERE EXISTS (subquery)	Die Bedingungen EXISTS prüft, ob die Suchbedingung einer Unterabfrage mindestens eine Zeile zurück liefert. NOT EXIST negiert die Bedingung.
WHERE NOT EXISTS (subquery)	
WHERE ... IN (subquery)	Der Wert des Datenfelds ist in der ausgewählten Menge vorhanden.
WHERE NOT... IN (subquery)	Der Wert des Datenfelds ist in der ausgewählten Menge nicht vorhanden.
GROUP BY Spaltenname1 [, Spaltenname2, ...]	Gruppierung (Aggregation) nach Inhalt des genannten Feldes
ORDER BY Spaltenname1 [, Spaltenname2, ...]	Sortierung nach Inhalt des genannten Feldes oder der genannten Felder
ASC DESC	ASC: aufsteigend; DESC: absteigend

Fortsetzung SQL-Syntax →

Syntax	Beschreibung
Datenmanipulation	
DELETE FROM Tabellenname	Löschen von Datensätzen in der genannten Tabelle
UPDATE Tabellenname SET	Aktualisiert Daten in Feldern einer Tabelle
INSERT INTO Tabellenname[(spalte1, spalte2, ...)] 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
Berechtigungen kontrollieren	
CREATE Benutzer Rolle IDENTIFIED BY 'Passwort'	Erzeugt einen neuen Benutzer oder eine neue Rolle mit einem Passwort
GRANT Recht Rolle ON *.* Datenbank.* Datenbank.Objekt TO Benutzer Rolle [WITH GRANT OPTION]	Weist einem Benutzer oder einer Rolle ein Recht auf ein bestimmtes Datenbank-Objekt zu Weist einem Benutzer eine Rolle zu
REVOKE Rechte Rollen ON *.* Datenbank.* Datenbank.Objekt FROM Benutzer Rolle	Entzieht einem Benutzer oder einer Rolle ein Recht auf ein bestimmtes Datenbank-Objekt Entzieht einem Benutzer eine Rolle
Aggregatfunktionen	
AVG(Spaltenname)	Ermittelt das arithmetische Mittel aller Werte im angegebenen Feld
COUNT(Spaltenname *)	Ermittelt die Anzahl der Datensätze mit Nicht-NULL-Werten im angegebenen Feld oder alle Datensätze der Tabelle (dann mit Operator *)
SUM(Spaltenname Formel)	Ermittelt die Summe aller Werte im angegebenen Feld oder der Formelergebnisse
MIN(Spaltenname Formel)	Ermittelt den kleinsten aller Werte im angegebenen Feld
MAX (Spaltenname 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 Text auf Gleichheit wenn Platzhalter ("regular expressions") eingesetzt werden.
NOT	Logische Negation
OR	Logisches ODER
IS NULL	Überprüfung auf NULL
=	Test auf Gleichheit
>, >=, <, <=, <>	Test auf Ungleichheit
*	Multiplikation
/	Division
+	Addition, positives Vorzeichen
-	Subtraktion, negatives Vorzeichen

Stand 2021-09-30

4. Aufgabe (24 Punkte)

Korrekturrand

- a) Sie erhalten den Auftrag, Produktionsdaten an die Steuerung der Walzanlage zu übergeben. Die Produktionsdaten werden in einer SQL-Datenbank gespeichert. Alle Datentypen sind Ganzzahlen. Die Breite, Länge und Dicke der Wellpappe wird in der Datenbank in Millimeter gespeichert.

Die Tabelle ProductionData hat den folgenden Aufbau:

OrderID (PK)
Width
Length
Thickness
Quantity

- aa) Geben Sie den SQL-Befehl an, der die Breite, die Länge, die Dicke und die Anzahl der OrderID 736298 ausgibt. Die OrderID soll nicht in der Ergebnismenge enthalten sein. 3 Punkte

`SELECT Width, Length, Thickness, Quantity
FROM ProductionData
WHERE OrderID = 736298;`

- ab) Wie viele Produktionsaufträge für Wellpappen mit einer Dicke von 2 mm wurden bisher in der Datenbank gespeichert.

Geben Sie dazu den entsprechenden SQL-Befehl an. 4 Punkte

`SELECT COUNT(OrderID) as "Anzahl" (um Sicher zu sein)
FROM ProductionData
WHERE Thickness = 2;`

- ac) Geben Sie die Gesamtanzahl gefertigter Wellpappen aus der Datenbank an, die mit einer Dicke von 2 mm, einer Breite von 200 mm und einer Länge von 300 mm gefertigt worden sind.

Geben Sie dazu den entsprechenden SQL-Befehl an. 4 Punkte

`SELECT SUM(Quantity) as "Gesamtanzahl"
FROM ProductionData
WHERE Thickness = 2
AND Width = 200
AND Length = 300;`

- b) Die abgefragten Produktionsdaten werden über eine entsprechende API an die Steuerung der Walzanlage übergeben. Die Auftragsdaten werden im Array `result[]` mit dem Index 0 bis 3 gespeichert. Sie sollen jetzt an die Steuerung der Walzanlage durch eine von Ihnen zu erstellende Funktion übergeben werden. Gehen Sie von einem Array `result[]` aus, bei dem im Index 0 die Breite, im Index 1 die Länge, im Index 2 die Dicke und im Index 3 die Anzahl der zu produzierenden Wellpappen stehen. Erstellen Sie die Funktion „`launchTask(result[])`“.

Zur Kommunikation mit der Steuerung der Walzanlage stehen Ihnen die folgenden API-Funktionen zur Verfügung:

`setRollerDim(int, int, int)` – Übergeben wird Breite, Länge und Dicke der Wellpappe.

`rollerStart()` – Startet einen Auftrag von einem Stück. Es wird eine Wellpappe mit den gesetzten Parametern erzeugt.

Die Walzanlage verfügt über einen Notausschalter. Sie darf nur laufen, wenn der Notaus nicht ausgelöst ist.

Der Status des Notausschalters kann mit der Funktion `bool getEmergencyStop()` abgefragt werden, der „true“ liefert wenn der Notaus ausgelöst ist und „false“ wenn der Notaus nicht ausgelöst ist.

Ergänzen Sie das gegebene Struktogramm durch die entsprechenden Befehle zur Produktion der geforderten Anzahl von Wellpappen (siehe Index 3) in den angegebenen Maßen (siehe Index 0, 1 und 2).

7 Punkte

<code>launchTask(result[])</code>
<code>int i = 0</code>
<code>bool emergencyStop = getEmergencyStop()</code>
<code>setRollerDim(result[0], result[1], result[2])</code>
<code>while (i < result[3] and emergencyStop == false)</code>
<code>rollerStart()</code>
<code>emergencyStop = getEmergencyStop()</code>
<code>i = i + 1</code>

Fortsetzung 4. Aufgabe

Korrekturrand

- c) Für die Produktion von Wellpappen ist die vorhandene Datenbank zu erweitern. Die Firma hat sich für ein SQL-fähiges relationales Datenbanksystem entschieden, in der die nachfolgenden Bedingungen berücksichtigt werden sollen. Die Speicherung der Datenbank wird auf dem Hostrechner „Steuerungs-PC“ realisiert. In einer ersten Unterredung werden die zu speichernden Informationen definiert.

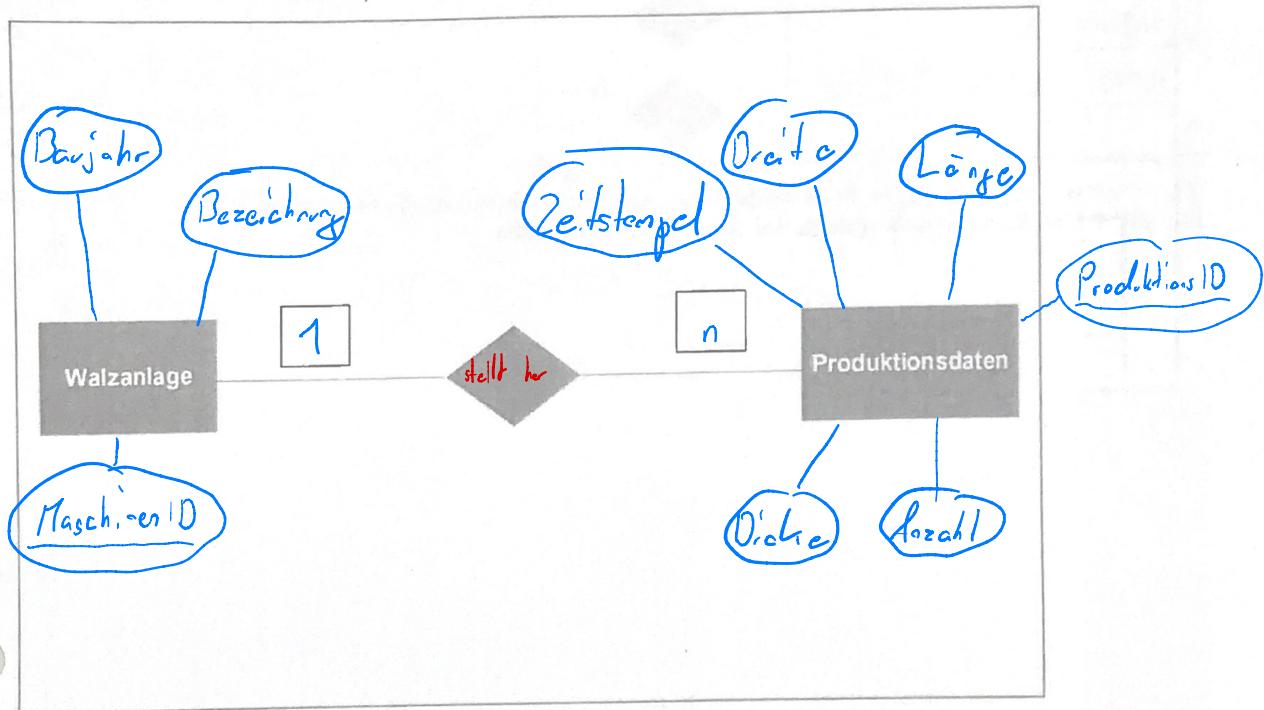
In dieser Datenbank sollen nur die Zusammenhänge zwischen den Walzanlagen, den Produktionsdaten abgebildet werden.

In der Produktionshalle sind mehrere Walzanlagen vorhanden. Diese jeweiligen Walzanlagen können Wellpappen mit unterschiedlichen Dicken (z. B. kleiner 4 mm, 4-8 mm, 8-12 mm) herstellen. In der Datenbank soll gespeichert werden, welche Walzanlage für welche Dicken (Spezifikation) verwendet werden kann. Außerdem soll das Baujahr, die Bezeichnung und eine eindeutige Maschinennummer gespeichert werden.

Für jede Walzanlage sollen die entsprechenden Produktionsdaten (Breite, Länge, Dicke und Anzahl) mit dem jeweiligen Zeitstempel abgespeichert werden.

Vervollständigen Sie das vorgegebene Entity-Relationship-Modell (kurz: ERM) für diese Datenbank mit allen erforderlichen Attributen und Kardinalitäten. 6 Punkte

Hinweis: Die eventuell benötigten Fremdschlüsse müssen nicht in diesem Entwurf eingetragen werden. Die Kardinalität zwischen den beiden Tabellen soll auf die entsprechenden Beziehungslinien eingetragen werden.



Hinweise: Tabelle (Chen-Notation)

Korrekturrand

Bezeichnung	Darstellung
Entity-Typ	
Attribut	
Primärschlüssel	
Beziehung (Relation, Relationship, Assoziation)	

PK bezeichnet ein Primärschlüsselattribut, **FK** ein Fremdschlüsselattribut, Primärschlüsselattribute werden unterstrichen, Fremdschlüsselattribute werden durch ein nachgestelltes Hash-Zeichen (#) kenntlich gemacht.

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.

Fortsetzung 4. Aufgabe

Korrekturrand

- d) Sie planen, eine eigene Lösung für eine automatisierte Konfiguration der Standardarbeitsplätze zu programmieren. Aus einer Datenbank werden alle zu konfigurierenden PCs ausgelesen. Danach wird für jeden PC aus der Datenbank die zu installierende Software abgefragt und auf dem PC installiert.

Es gibt die folgenden Variablen:

PCNr Ganzzahl – Laufvariable
SoftwareNr Ganzzahl – Laufvariable

Es gibt die folgenden Felder (Array)

PCListe[] Stringliste mit den Namen der PC
SoftwareListe[] Stringliste mit den Namen der Software

Es stehen Ihnen die folgenden Funktionen zur Verfügung:

getPC() – liefert eine Liste von PC-Namen aus der Datenbank
getSoftware(String) – liefert zu dem angefragten PC eine Liste der zu installierenden Software

installSoftware(String, String) Installiert die im ersten String angegebene Software auf dem im zweiten String übergebenen PC

Tragen Sie die Anweisungen folgerichtig in das nebenstehende Struktogramm ein.

9 Punkte

1. installSoftware(SoftwareListe [SoftwareNr], PCListe[PCNr])
2. Solange SoftwareNr < Anzahl der Elemente in SoftwareListe []
3. PCListe[] = getPC()
4. PCNr = PCNr + 1
5. PCNr = 0
6. SoftwareListe[] = getSoftware(PCListe[PCNr])
7. SoftwareNr = 0
8. SoftwareNr = SoftwareNr + 1
9. Solange PCNr < Anzahl der Elemente in PCListe[]

PCListe[3] = getPC()

PCNr = 0

Solang PCNr < Anzahl der Elemente in PCListe[]

SoftwareListe[3] = getSoftware(PCListe[PCNr])

SoftwareNr = 0

Solang SoftwareNr < Anzahl der Elemente in SoftwareListe[]

install Software (SoftwareListe[SoftwareNr],
PCListe[PCNr])

SoftwareNr = SoftwareNr + 1

PCNr = PCNr + 1