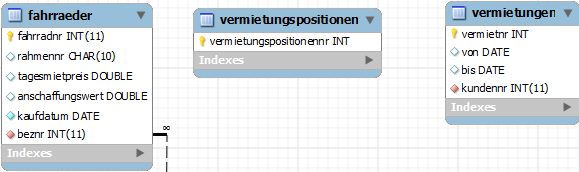
|  |  |
| --- | --- |
| Lösungsblatt | DB04\_L |

**zu 1: Entity-Relationsship-Modell**

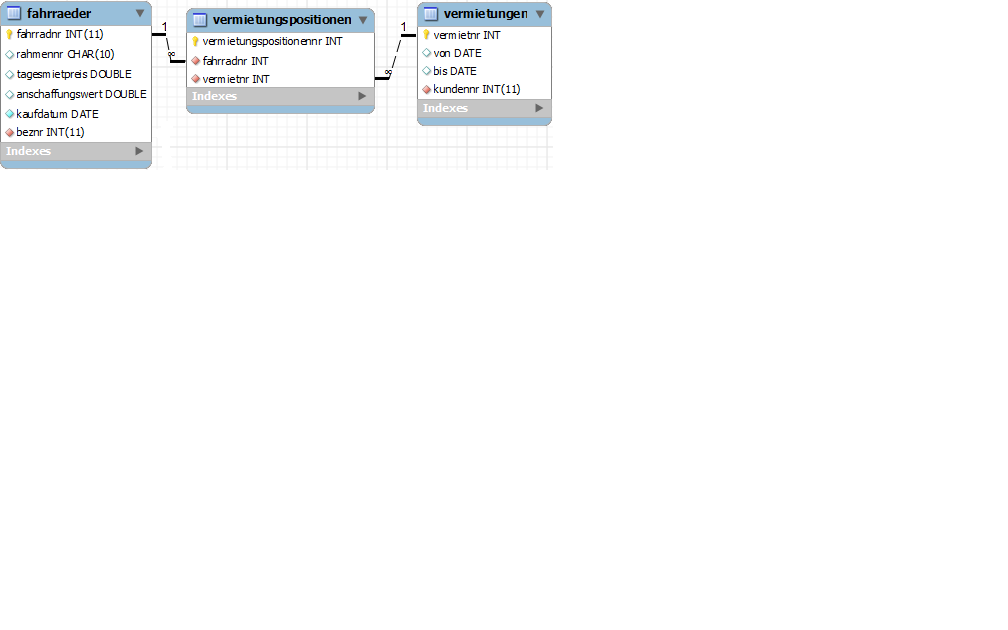


**zu 2: Datenbank anpassen**

1. Datenbankserver muss gestartet sein, dann  
   im SQL-Editor unter Object Browser die Tabelle vermietungen mit einem Rechtsklick löschen:  
   - Drop Table… und anschließend  
   - Refresh All
2. Workbench starten und das EER-Diagramm des letzten Kapitels öffnen (fahrradvermietung\_7.mwb)
3. Die Beziehung zwischen fahrraeder und vermietungen mit einem Rechtsklick löschen.  
   Der Fremdschlüssel in der Tabelle vermietungen wurde dadurch auch gelöscht.  
   Die Tabelle *vermietungspositionen* hinzufügen:



1. Die Fremdschlüssel hinzufügen und ihre Namen ändern:



1. Das EER-Diagramm unter fahrradvermietung\_8\_2013.mwb speichern !!!
2. Das EER-Diagramm wird nun per ***Forward Engineer*** in die Datenbank übertragen. In der Datenbank fahrradvermietung befinden sich nun die beiden leeren Tabellen vermietungen und vermietungspositionen.
3. Im SQL-Editor unter Object Browser die Ansicht aktualisieren (Refresh All).

**zu 3: Übernehmen Sie die vorhandenen Daten in die Datenbank**

siehe: fahrradvermietung\_2013\_8\_Daten.sql

Das SQL-Script enthält Datensätze für die beiden Tabellen vermietungen und  
 vermietungspositionen.

**zu 4: SQL-Aufgaben**

4.1 SELECT kunden.kundennr, kunden.name, vermietungen.vermietnr,

vermietungspositionen.vermietungspositionennr, modelle.bezeichnung

FROM kunden, vermietungen, vermietungspositionen, fahrraeder, modelle

WHERE kunden.kundennr = vermietungen.kundennr

AND vermietungen.vermietnr = vermietungspositionen.vermietnr

AND vermietungspositionen.fahrradnr = fahrraeder.fahrradnr

AND modelle.modellnr = fahrraeder.modellnr

ORDER BY kunden.kundennr, vermietungen.vermietnr,   
 vermietungspositionen.vermietungspositionennr ASC;

4.2 SELECT vermietungspositionen.vermietnr, COUNT(\*) AS Anzahl

FROM vermietungspositionen

GROUP BY vermietungspositionen.vermietnr

ORDER BY vermietungspositionen.vermietnr ASC;

4.3 SELECT fahrraeder.fahrradnr, modelle.bezeichnung,

COUNT(vermietungspositionen.fahrradnr) AS 'Anzahl Vermietungen'

FROM fahrraeder, vermietungspositionen, modelle

WHERE fahrraeder.fahrradnr = vermietungspositionen.fahrradnr

AND modelle.modellnr = fahrraeder.modellnr

GROUP BY fahrraeder.fahrradnr, modelle.bezeichnung

ORDER BY fahrraeder.fahrradnr ASC;

**zu 5.:**

**5.1 Was ist eine Funktion:**

**In der Mathematik (z. B.):**

ist eine eindeutige Zuordnung, die jedem Element x aus der Menge D (Definitionsmenge) eindeutig ein Element y aus einer Menge W (Wertebereich) zuordnet.

Eine Funktion ordnet jedem Element einer Definitionsmenge genau ein Element der Zielmenge (Wertemenge) zu

x ∈ D und y ∈ Z

f: x 🡪 f(x) oder f mit f(x) = …

**In der Informatik (z. B.):**

Ist eine Abbildungsvorschrift, die einer Menge an Eingabedaten eine Menge an Ausgabedaten zuordnet. Eine Funktion erzeugt oder verändert Datenobjekte.

Unter einer Funktion kann man sich eine Input-Ouput-Maschine vorstellen. Sie nimmt eine Zahl oder ein Text entgegen und gibt darauf hin eine Zahl oder einen Text aus. Das macht sie nach einer genauen (eindeutigen) Vorschrift: gleiche Eingaben führen immer zu gleichen Ausgaben.

**In MySQL:**

Mit benutzerdefinierten Funktionen (UDFs) können Sie MySQL erweitern, die wie die bereits eingebauten (nativen) MySQL-Funktionen (z. B. COUNT() oder DATEDIFF()) funktionieren.

**5.2 Wie wird eine Funktion erstellt:**

Zuerst muss überlegt werden, was die Funktion tun soll:

'Ich' gebe ihr den Inhalt des Attributs ***bis*** und den Inhalt des Attributs ***von*** und sie gibt mir 0, 10 oder 20 zurück.



Im Objektbrowser in der Datenbank ***fahrradvermietung*** mit einem Rechtsklick auf Routines ***Create Routine…*** anklicken und die Programmzeilen eingeben.

**CREATE FUNCTION `rabattsatz\_berechnen` (pBis date, pVon date)**

**RETURNS double**

**BEGIN**

**if datediff(pbis,pvon) <=10 then**

**RETURN 0;**

**else**

**if datediff(pbis,pvon) <=20 then**

**RETURN 10;**

**else**

**RETURN 20;**

**end if;**

**end if;**

**END**

Erläuterung zur Kopfzeile:

Hinter dem Namen der Funktion folgt

* die Parameterliste mit dem Datentyp jedes jeweiligen Parameters,
* gefolgt vom Datentyp des Rückgabewertes.

Die Funktion wird mit einem Klick auf ***Apply*** unter dem in der Funktion angegebenen Namen gespeichert:



**5.3 Wie wird eine Funktion in einer Abfrage verwendet:**

SELECT vermietungen.vermietnr,

SUM(tagesmietpreis \* datediff(bis,von)) AS `Normalpreis`,

SUM(tagesmietpreis \* datediff(bis,von) \*

(1 - rabattsatz\_berechnen(bis,von)/100)) AS `Vermietungsgesamtpreis`,

rabattsatz\_berechnen(bis,von) AS `Rabattsatz`

FROM vermietungen, fahrraeder, vermietungspositionen,modelle

WHERE fahrraeder.fahrradnr = vermietungspositionen.fahrradnr

AND vermietungspositionen.vermietnr = vermietungen.vermietnr

AND modelle.modellnr=fahrraeder.modellnr

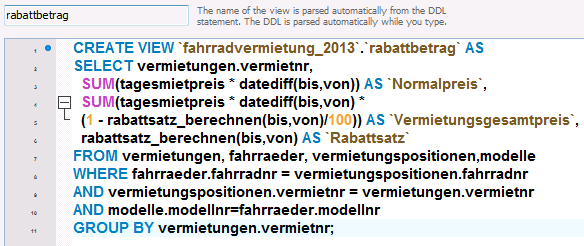
GROUP BY vermietungen.vermietnr;

Hinweis:

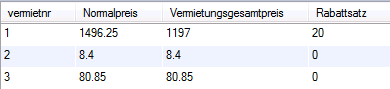
**Evtl. muss die Mietdauer erhöht werden!!!**

**5.4 Wie kann die Abfrage allen Sachbearbeitern zur Verfügung gestellt werden:**

Es muss ein view erstellt werden.



Ergebnis des view:



**6.1 Die Function**

CREATE DEFINER=`root`@`localhost` FUNCTION `linafa\_berechnen`(pAnschaffungswert double, pAnschaffungsjahr integer) RETURNS double

BEGIN

DECLARE maxnutzungsjahre integer;

DECLARE linafa double;

Declare aktnutzungsjahr integer;

set maxnutzungsjahre=5;

set aktnutzungsjahr=year(now())-pAnschaffungsjahr+1 ;

if aktnutzungsjahr > 0 Then

If aktnutzungsjahr < maxnutzungsjahre Then

set linafa = pAnschaffungswert / maxnutzungsjahre;

End If;

If aktnutzungsjahr= maxnutzungsjahre Then

set linafa = (pAnschaffungswert / maxnutzungsjahre) -1;

End If;

If aktnutzungsjahr > maxnutzungsjahre Then

set linafa = 0;

End If;

Else

set linafa = 0;

End If;

RETURN linafa;

END

**6.2 Die Abfrage**

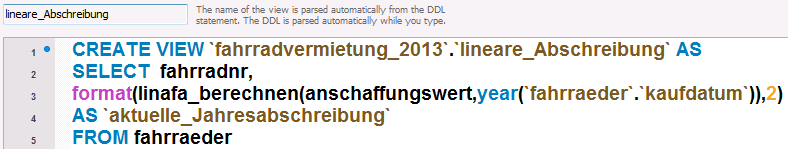
SELECT fahrradnr,

format(linafa\_berechnen(anschaffungswert,year(`fahrraeder`.`kaufdatum`)),2)

AS `aktuelle\_Jahresabschreibung`

FROM fahrraeder

**6.3 Der View**

****

**Sonderaufgabe:**

Code zu den Ereignisprozeduren der Buttons:

Option Compare Database

Private Sub Form\_Current()

tfAfA = 0

tfSumme = 0

End Sub

Private Sub btnAfA\_Click()

Dim mNutzungsjahr As Integer

mNutzungsjahr = Year(Date) + 1 - Year(kaufdatum)

tfAfA = linafa(anschaffungswert, mNutzungsjahr, 5)

End Sub

Private Sub btnSumme\_Click()

Dim mNutzungsjahr As Integer

mNutzungsjahr = Year(Date) + 1 - Year(kaufdatum)

tfSumme = summelinafa(anschaffungswert, mNutzungsjahr, 5)

End Sub

Function linafa(pAnschaffungswert As Double, pNutzungsjahr As Integer, pMaxjahre As Integer) As Double

Dim abschreibungsbetrag As Double

If pNutzungsjahr > 0 Then

If pNutzungsjahr < pMaxjahre Then

abschreibungsbetrag = pAnschaffungswert / pMaxjahre

End If

If pNutzungsjahr = pMaxjahre Then

abschreibungsbetrag = pAnschaffungswert / pMaxjahre - 1

End If

If pNutzungsjahr > pMaxjahre Then

abschreibungsbetrag = 0

End If

Else

abschreibungsbetrag = 0

End If

linafa = abschreibungsbetrag

End Function

Function summelinafa(pAnschaffungswert As Double, pNutzungsjahr As Integer, pMaxjahre As Integer) As Double

Dim summe As Double

Dim i As Integer

summme = 0

If pNutzungsjahr > 0 Then

For i = 1 To pNutzungsjahr

summe = summe + linafa(pAnschaffungswert, i, pMaxjahre)

Next i

End If

summelinafa = summe

End Function