Handout
Themenfeld: Datenbanken und SQL
Abschnitt: 09.01.01. Abfragen über mehrere Tabellen - Grundlagen

Autor: Thomas Krause
Stand: 14.11.2022 12:03:00

Inhalt

1 Vorbereitung und Übersicht

Vorbereitung:

Installieren Sie bei Bedarf die mitgelieferte Datenbank über folgendes Skript:

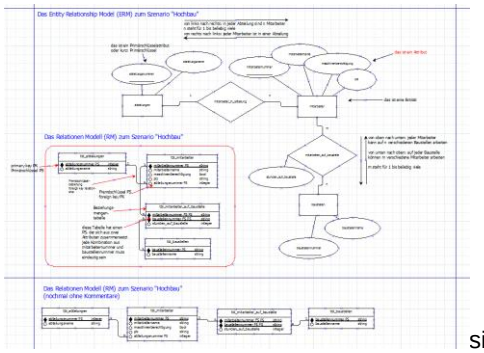
create_database_hochbau_MUSTERDATENBANK.sql

Szenario:

Der Vortrag bezieht sich auf das folgende, bereits gemeinsam bearbeitete Szenarion. Downloaden Sie bei Bedarf nochmals die Dateien bzw. installieren Sie sich die Datenbank über das mitgelieferte Skript.

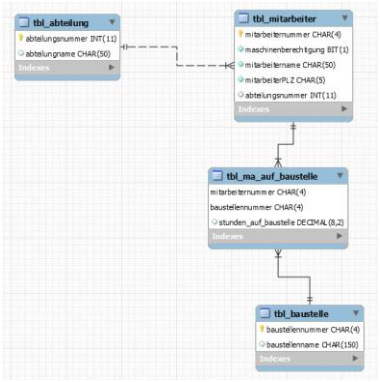
DIA:

siehe Dateien



siehe Datei:
database_hochbau_MUSTER_DB_ERM_RM.dia

Workbench:



Übersicht:

- Informationen werden in verschiedenen Tabellen gespeichert
- das Tabellen-Modell wird durch Anwendung der Normalisierungsregeln (NF1 … NF3) erstellt und enthält Tabellen mit Spalten, Primärschlüsseln und Fremdschlüsselbeziehungen
- die Normalisierung ist notwendig, um eine redundanzfreie/ redundanzarme und effiziente Speicherung zu erreichen
- die zusammengehörigen Informationen werden bei Abfragen und Verarbeitung wieder zusammengeführt, indem über mehrere Tabellen abgefragt wird

2 Abfragen über 2 Tabellen

Aufgabe/ Beispiel:

- die über Normalisierung auf mehrere Tabellen verteilten Informationen werden in einer SQL-Abfrage zusammengeführt
- im Beispiel: Liste mit Namen der Mitarbeiter und Namen der Abteilungen, in denen sie arbeiten

Ausgangstabellen:

mitarbeiternummer	maschinenberechtigung	mitarbeitername	mitarbeiterPLZ	abteilungsnummer
M009	0	Örtel	04105	9
M010	1	Stein	04838	12
M021	1	Hahn	04509	10
M024	0	Holzer	04119	9

abteilungsnummer	abteilungname
9	Hochbau
10	Haustechnik
12	Ausbau

HINWEIS: beide Tabellen sind über eine Fremdschlüsselbeziehung miteinander verbunden

Ergebnisliste:

Beispiel 1:

mitarbeiternummer	maschinenberechtigung	mitarbeitername	mitarbeiterPLZ	abteilungsnummer	abteilungsnummer	abteilungname
M009	0	Örtel	04105	9	9	Hochbau
M010	1	Stein	04838	12	12	Ausbau
M021	1	Hahn	04509	10	10	Haustechnik
M024	0	Holzer	04119	9	9	Hochbau

Beispiel 2:

mitarbeiternummer	mitarbeitername	abteilungname
M009	Örtel	Hochbau
M010	Stein	Ausbau
M021	Hahn	Haustechnik
M024	Holzer	Hochbau

Syntax:

Es existieren in SQL 2 unterschiedliche Möglichkeiten, um 2 (und mehr) Tabellen in einer Abfrage zu verbinden:

(1) mit Verwendung des Befehls JOIN bzw. INNER JOIN
(2) mit Verwendung eines logischen Ausdrucks nach dem Befehl WHERE (also ohne JOIN)

MÖGLICHKEIT 1 mit Befehl INNER JOIN:

```
SELECT *
FROM <name_tab1> [INNER] JOIN <name_tab2> ON [<name_tab1>.]<feld_tab1> = [<name_tab2>.]<feld_tab2>


SELECT
        <feld>,
        <feld>,
        [<name_tab1>.]<feld>
        [<name_tab2>.]<feld>
FROM <name_tab1> [INNER] JOIN <name_tab2> ON [<name_tab1>.]<feld_tab1> = [<name_tab2>.]<feld_tab2>
```

Ausgangstabellen: siehe oben

Ergebnisliste:

Beispiel 1:

mitarbeiternummer	maschinenberechtigung	mitarbeitername	mitarbeiterPLZ	abteilungsnummer	abteilungsnummer	abteilungname
M009	0	Örtel	04105	9	9	Hochbau
M010	1	Stein	04838	12	12	Ausbau
M021	1	Hahn	04509	10	10	Haustechnik
M024	0	Holzer	04119	9	9	Hochbau

```
select
       *
from tbl_mitarbeiter join tbl_abteilung
on tbl_mitarbeiter.abteilungsnummer = tbl_abteilung.abteilungsnummer;
```

Beispiel 2:

mitarbeiternummer	mitarbeitername	abteilungname
M009	Örtel	Hochbau
M010	Stein	Ausbau
M021	Hahn	Haustechnik
M024	Holzer	Hochbau

```
select
       mitarbeiternummer,
       mitarbeitername,
       abteilungname
from tbl_mitarbeiter join tbl_abteilung
on tbl_mitarbeiter.abteilungsnummer = tbl_abteilung.abteilungsnummer;
```

BEACHTE:

- bei vielen DBMS kann das Schlüsselwort INNER entfallen. Wenn nur JOIN verwendet wird, wird das automatisch als INNER JOIN interpretiert

MÖGLICHKEIT 2 mit Befehl WHERE:

```
SELECT *
FROM <name_tab1>, <name_tab2> WHERE [<name_tab1>.]<feld_tab1> = [<name_tab2>.]<feld_tab2>


SELECT
        <feld>,
        <feld>,
        [<name_tab1>.]<feld>
        [<name_tab2>.]<feld>
FROM <name_tab1>, <name_tab2> WHERE [<name_tab1>.]<feld_tab1> = [<name_tab2>.]<feld_tab2>
```

Ausgangstabellen: siehe oben

Ergebnisliste:

Beispiel 1:

mitarbeiternummer	maschinenberechtigung	mitarbeitername	mitarbeiterPLZ	abteilungsnummer	abteilungsnummer	abteilungname
M009	0	Örtel	04105	9	9	Hochbau
M010	1	Stein	04838	12	12	Ausbau
M021	1	Hahn	04509	10	10	Haustechnik
M024	0	Holzer	04119	9	9	Hochbau

```
select
       *
from tbl_mitarbeiter, tbl_abteilung
where tbl_mitarbeiter.abteilungsnummer = tbl_abteilung.abteilungsnummer;
```

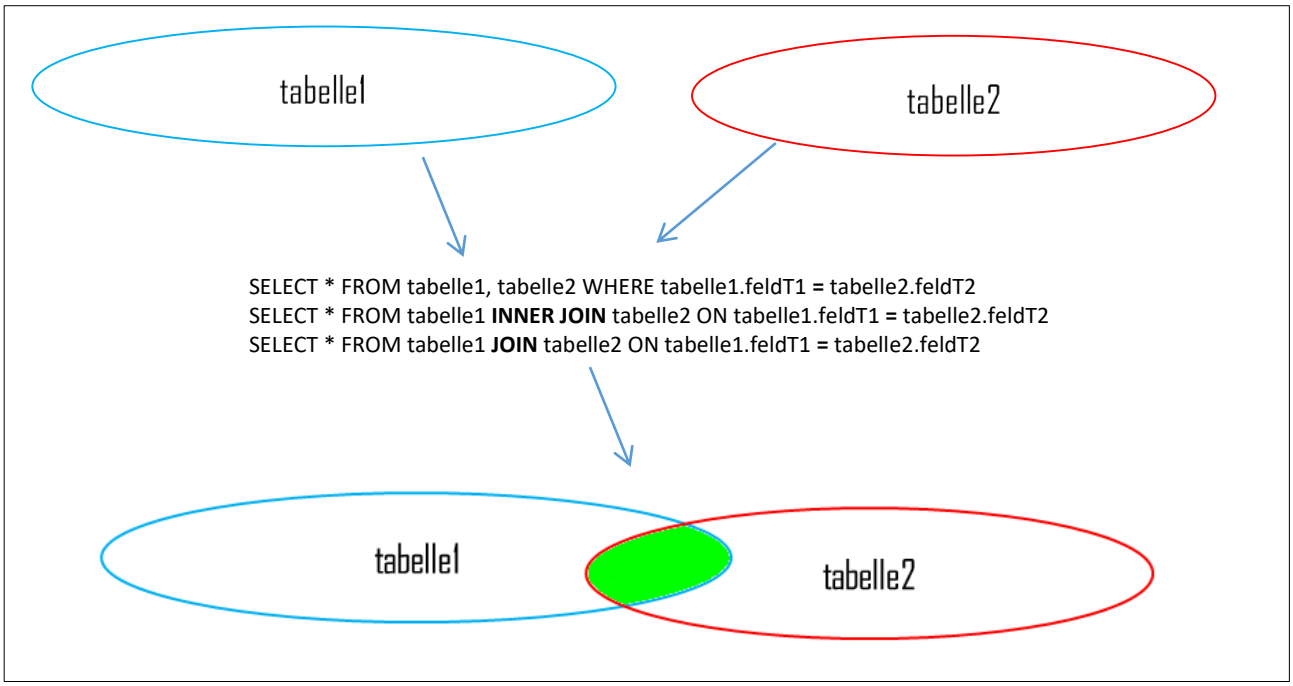
Beispiel 2:

mitarbeiternummer	mitarbeitername	abteilungname
M009	Örtel	Hochbau
M010	Stein	Ausbau
M021	Hahn	Haustechnik
M024	Holzer	Hochbau

```
select
       mitarbeiternummer,
       mitarbeitername,
       abteilungname
from tbl_mitarbeiter, tbl_abteilung
where tbl_mitarbeiter.abteilungsnummer = tbl_abteilung.abteilungsnummer;
```

BEACHTE:

- Befehl JOIN bzw. INNER JOIN entfällt → Tabellen werden nur mit Komma getrennt aufgeführt
- Befehl ON wird durch WHERE ersetzt



weitere Erläuterungen:

- wird auch Equi-Join genannt (wegen des = Zeichens)
- es werden die Datensätze angezeigt, bei denen die beiden angegebenen Felder feldT1 und feldT2 in beiden Tabellen die gleichen Werte enthalten
- INNER JOIN → es werden NUR die Datensätze geliefert, bei den die Verknüpfungsbedingung übereinstimmt, ein TRUE liefert
  (die Verknüpfungsbedingung ist das =)
- heißt auch „Schnittmenge“ oder auch „Intersection“
- wird i.d.R. dort angewendet, wo die Felder feldT1 und feldT2 über eine Fremdschlüsselbeziehung verbunden sind (das muss aber nicht so sein)
- besondere Form des INNER JOIN ist der NATURAL JOIN
  → es werden automatisch die beiden Felder der beiden Tabellen verknüpft, die den gleichen Namen haben und
  → im Ergebnis werden identische Spalten nur einmal aufgeführt
  → nicht in jedem DBMS verfügbar
- es können auch verwendet werden: WHERE, ORDER BY, GROUP BY
- die verknüpften Felder müssen den gleichen Datentyp haben (eventuell anpassen über Befehl CAST)

praktische Anwendungen:

Beispiel 1:

Liste mit Nummer sowie Namen der Baustellen und geleisteten Stunden (einzelne Positionen)

Ausgangstabellen:

mitarbeiternummer	baustellennummer	stunden_auf_baustelle
M009	B253	37.00
M010	B021	12.00
M010	B112	23.00
M021	B056	21.00
M021	B112	24.00
M021	B253	34.00
M024	B056	8.00
M024	B253	24.00

```
select * from tbl_ma_auf_baustelle;
```

baustellennummer	baustellenname
B021	MIDL
B056	Brutto
B112	Kaufstadt
B253	GaleriaX

```
select * from tbl_baustelle;
```

Ergebnisliste:

baustellennummer	baustellenname	stunden_auf_baustelle
B253	GaleriaX	37.00
B021	MIDL	12.00
B112	Kaufstadt	23.00
B056	Brutto	21.00
B112	Kaufstadt	24.00
B253	GaleriaX	34.00
B056	Brutto	8.00
B253	GaleriaX	24.00

Möglichkeit 1:

```
select
	tbl_baustelle.baustellennummer,
	baustellenname,
	stunden_auf_baustelle
from tbl_ma_auf_baustelle join tbl_baustelle
on tbl_ma_auf_baustelle.baustellennummer = tbl_baustelle.baustellennummer;
```

Möglichkeit 2:

```
select
      tbl_baustelle.baustellennummer,
      baustellenname,
      stunden_auf_baustelle
from tbl_ma_auf_baustelle, tbl_baustelle
where tbl_ma_auf_baustelle.baustellennummer =
tbl_baustelle.baustellennummer;
```