

WIEDERHOLUNG

JOIN

INNER JOIN

```
SQL> SELECT * FROM PERSONAL JOIN ABTEILUNG ON PERSONAL.ABT_NR=ABTEILUNG.ABT_NR;
```

PNR	VORNAME	NACHNAME	GEH_S	ABT_N	KRA	ABT_N	NAME
123	Ada	Lovelace	it5	abt13	dek	abt13	Produktion
124	Leonhard	Euler	it2	abt11	aok	abt11	Verwaltung
126	Ren??	Descartes	it1	abt12	bak	abt12	Projektierung
127	Alan	Turing	it2	abt14	dek	abt14	Lagerung
132	Grace	Hopper	it4	abt15	dek	abt15	Verkauf
145	Marie	Curie	it4	abt15	aok	abt15	Verkauf
147	Hedy	Lamarr	it5	abt11	tkk	abt11	Verwaltung
133	Radia	Perlman	it3	abt12	tkk	abt12	Projektierung
161	Martin	Fowler	it2	abt13	dek	abt13	Produktion
163	Erich	Gamma	it1	abt14	aok	abt14	Lagerung
164	Marlyn	Meltzer	it4	abt13	bak	abt13	Produktion
177	Linus	Torvalds	it2	abt11	dek	abt11	Verwaltung

12 rows selected.

OUTER JOIN

```
SQL> SELECT * FROM PERSONAL LEFT OUTER JOIN ABTEILUNG ON PERSONAL.ABT_NR=ABTEILUNG.ABT_NR;
```

PNR	VORNAME	NACHNAME	GEH_S	ABT_N	KRA	ABT_N	NAME
177	Linus	Torvalds	it2	abt11	dek	abt11	Verwaltung
147	Hedy	Lamarr	it5	abt11	tkk	abt11	Verwaltung
124	Leonhard	Euler	it2	abt11	aok	abt11	Verwaltung
133	Radia	Perlman	it3	abt12	tkk	abt12	Projektierung
126	Ren??	Descartes	it1	abt12	bak	abt12	Projektierung
164	Marlyn	Meltzer	it4	abt13	bak	abt13	Produktion
161	Martin	Fowler	it2	abt13	dek	abt13	Produktion
123	Ada	Lovelace	it5	abt13	dek	abt13	Produktion
163	Erich	Gamma	it1	abt14	aok	abt14	Lagerung
127	Alan	Turing	it2	abt14	dek	abt14	Lagerung
145	Marie	Curie	it4	abt15	aok	abt15	Verkauf
132	Grace	Hopper	it4	abt15	dek	abt15	Verkauf
173	Richard	Helm	it1		dek		

13 rows selected.

```
SQL> SELECT * FROM PERSONAL RIGHT OUTER JOIN ABTEILUNG ON PERSONAL.ABT_NR=ABTEILUNG.ABT_NR;
```

PNR	VORNAME	NACHNAME	GEH_S	ABT_N	KRA	ABT_N	NAME
123	Ada	Lovelace	it5	abt13	dek	abt13	Produktion
124	Leonhard	Euler	it2	abt11	aok	abt11	Verwaltung
126	Ren??	Descartes	it1	abt12	bak	abt12	Projektierung
127	Alan	Turing	it2	abt14	dek	abt14	Lagerung
132	Grace	Hopper	it4	abt15	dek	abt15	Verkauf
145	Marie	Curie	it4	abt15	aok	abt15	Verkauf
147	Hedy	Lamarr	it5	abt11	tkk	abt11	Verwaltung
133	Radia	Perlman	it3	abt12	tkk	abt12	Projektierung
161	Martin	Fowler	it2	abt13	dek	abt13	Produktion
163	Erich	Gamma	it1	abt14	aok	abt14	Lagerung
164	Marlyn	Meltzer	it4	abt13	bak	abt13	Produktion
177	Linus	Torvalds	it2	abt11	dek	abt11	Verwaltung
						abt16	Forschung

13 rows selected.


```
SQL> SELECT * FROM PERSONAL FULL OUTER JOIN ABTEILUNG ON PERSONAL.ABT_NR=ABTEILUNG.ABT_NR;
```

PNR	VORNAME	NACHNAME	GEH_S	ABT_N	KRA	ABT_N	NAME
123	Ada	Lovelace	it5	abt13	dek	abt13	Produktion
124	Leonhard	Euler	it2	abt11	aok	abt11	Verwaltung
126	Ren??	Descartes	it1	abt12	bak	abt12	Projektierung
127	Alan	Turing	it2	abt14	dek	abt14	Lagerung
132	Grace	Hopper	it4	abt15	dek	abt15	Verkauf
145	Marie	Curie	it4	abt15	aok	abt15	Verkauf
147	Hedy	Lamarr	it5	abt11	tkk	abt11	Verwaltung
133	Radia	Perlman	it3	abt12	tkk	abt12	Projektierung
161	Martin	Fowler	it2	abt13	dek	abt13	Produktion
163	Erich	Gamma	it1	abt14	aok	abt14	Lagerung
164	Marlyn	Meltzer	it4	abt13	bak	abt13	Produktion
173	Richard	Helm	it1		dek		
177	Linus	Torvalds	it2	abt11	dek	abt11	Verwaltung
						abt16	Forschung

14 rows selected.

AGGREGATSFUNKTIONEN

COUNT

MIN UND MAX

SUM

AVG

UNION

SUBQUERIES

ORDER BY

FRAGEN

1. GEBEN SIE FÜR JEDEN MITARBEITER, DER MINDESTENS EINE PRÄMIE ERHALTEN HAT, DIE PERSONALNUMMER UND DIE ANZAHL DER PRÄMIEN AN.
2. GEBEN SIE DIE PNR DER MITARBEITER SORTIERT NACH GESAMTPRÄMIENHÖHE AUS.
3. GEBEN SIE ALLE MITARBEITER AUS, DIE LETZTES JAHR EINE PRÄMIE BEKOMMEN HABEN.
4. GEBEN SIE FÜR JEDEN MITARBEITER DEN NAMEN, DIE PNR UND DIE ERHALTENE PRÄMIE AN.
5. GEBEN SIE ALLE VORNAMEN ALLER MITARBEITER UND KINDER AUF EINMAL AUS, ABSTEIGEND SORTIERT.
6. GEBEN SIE ALLE IM AUGUST GEBOREN KINDER AUS.
7. GEBEN SIE ALLE VOR 2010 GEBORENEN KINDER AUS.
8. GEBEN SIE EINE LISTE ALLER KINDER SORTIERT NACH NACHNAME UND VORNAME AUS.
9. GEBEN SIE ALLE MITARBEITER AUS, DEREN VORNAMEN MIT EINEM A ANFANGEN.
10. GEBEN SIE ALLE MITARBEITER MIT EINEM JAHRESGEHALT VON MEHR ALS 40.000 EUR AUS (PRÄMIEN NICHT MIT EINGESCHLOSSEN).
11. GEBEN SIE ALLE ABTEILUNGEN AUS, DIE WENIGER ALS 2 MITARBEITER HABEN.
12. GEBEN SIE DEN NAMEN UND DEN PROZENTSATZ DES ZEITWERTS VOM NEUWERTS FÜR JEDE MASCHINE AUS.
13. GEBEN SIE DIE NAMEN DER ABTEILUNGEN AUS, DIE MITARBEITER HABEN, DIE MASCHINEN BENUTZEN.

1. GEBEN SIE FÜR JEDEN MITARBEITER, DER MINDESTENS EINE PRÄMIE ERHALTEN HAT, DIE PERSONALNUMMER UND DIE ANZAHL DER PRÄMIEN AN.

**2. GEBEN SIE DIE PNR DER MITARBEITER SORTIERT NACH
GESAMTPRÄMIENHÖHE AUS.**

3. GEBEN SIE ALLE MITARBEITER AUS, DIE LETZTES JAHR EINE PRÄMIE BEKOMMEN HABEN.

**4. GEBEN SIE FÜR JEDEN MITARBEITER DEN NAMEN, DIE PNR UND
DIE ERHALTENE PRÄMIE AN.**

**5. GEBEN SIE ALLE VORNAMEN ALLER MITARBEITER UND KINDER
AUF EINMAL AUS, ABSTEIGEND SORTIERT.**

6. GEBEN SIE ALLE IM AUGUST GEBOREN KINDER AUS.

7. GEBEN SIE ALLE VOR 2010 GEBORENEN KINDER AUS.

**8. GEBEN SIE EINE LISTE ALLER KINDER SORTIERT NACH
NACHNAME UND VORNAME AUS.**

9. GEBEN SIE ALLE MITARBEITER AUS, DEREN VORNAMEN MIT EINEM A ANFANGEN.

10. GEBEN SIE ALLE MITARBEITER MIT EINEM JAHRESGEHALT VON MEHR ALS 40.000 EUR AUS (PRÄMIEN NICHT MIT EINGESCHLOSSEN).

**11. GEBEN SIE ALLE ABTEILUNGEN AUS, DIE WENIGER ALS 2
MITARBEITER HABEN.**

**12. GEBEN SIE DEN NAMEN UND DEN PROZENTSATZ DES
ZEITWERTS VOM NEUWERTS FÜR JEDE MASCHINE AUS.**

13. GEBEN SIE DIE NAMEN DER ABTEILUNGEN AUS, DIE MITARBEITER HABEN, DIE MASCHINEN BENUTZEN.

LOS GEHT'S

\o/

INSERT

```
INSERT INTO {tabellenname|viewname} [(liste von spaltennamen)]  
{VALUES (konst. ausdrück [,konst. ausdrück...])|  
SELECT-Anweisung}
```

```
INSERT INTO PERSONAL VALUES(167, ' Krause', 'Gustav', 'it3', ' d12', ' dak');
```

```
INSERT INTO PERSONAL (PNR, NAME, VORNAME) VALUES (777, 'Graf', 'Gerd');
```

```
INSERT INTO tabellenname  
[(liste von spaltennamen)] SELECT-Anweisung;
```

- **SELECT-ANWEISUNG DARF NICHT AUF DIE HINTER INSERT INTO STEHENDE TABELLE VERWEISEN**
- **ANZAHL DER ATTRIBUTNAMEN DER INSERT INTO-KOMPONENTE MUSS MIT DER ANZAHL DERSELBEN IN DER SELECT-KOMPONENTE ÜBEREINSTIMMEN**
 - **DATENTYPEN MÜSSEN EBENFALLS ÜBEREINSTIMMEN**

```
CREATE TABLE PERSONAL_NEU (  
    PNR SMALLINT PRIMARY KEY,  
    NAME VARCHAR(20) NOT NULL,  
    VORNAME VARCHAR(20)  
);
```

```
INSERT INTO PERSONAL_NEU  
SELECT PNR, NAME, VORNAME  
FROM PERSONAL  
WHERE KRANKENKASSE = "aok";
```

ALS HILFSTABELLE BENUTZEN

```
CREATE TABLE HILFE (  
    PNR SMALLINT NOT NULL,  
    NAME CHAR(20) NOT NULL,  
    VORNAME CHAR(20)  
);
```

```
INSERT INTO HILFE (PNR, NAME, VORNAME)  
SELECT PNR, NAME, VORNAME  
FROM PERSONAL;
```



```
DROP TABLE PERSONAL;
```

```
CREATE TABLE PERSONAL (  
    PNR SMALLINT NOT NULL,  
    NAME CHAR(20) NOT NULL,  
    VORNAME CHAR(20)  
);
```

```
INSERT INTO PERSONAL  
SELECT *  
FROM HILFE;
```

```
DROP TABLE HILFE;
```

AUFGABEN

1. FUGEN SIE ZWEI NEUE MASCHINEN EIN
2. ZWEI MITARBEITER HABEN PRÄMIEN ERHALTEN. ERGÄNZEN SIE DIESE.

UPDATE

```
UPDATE {tabellenname|viewname}  
SET spaltenname1 = {ausdruck1|NULL|(SELECT-anweisung)}  
    [,spaltenname2={ausdruck2|NULL|(SELECT-anw.)} ...]  
[WHERE suchbedingung]
```

UPDATE

NAME DER TABELLE, IN DER DIE WERTE VERANDERT WERDEN SOLLEN

SET

- ES STEHEN DIE SPALTEN, IN DENEN WERTE VERANDERT WERDEN SOLLEN
- ES STEHEN DIE WERTE, DIE DIESE SPALTEN JETZT AN STELLE DER ALTEN WERTE HABEN SOLLEN
- DIESE WERTE KÖNNEN AUCH AUS ANDEREN TABELLEN ENTNOMMEN WERDEN MITTELS SELECT

WHERE

SPEZIFIZIERT DIE ZEILE(N), IN DENEN DER SPALTENWERT VERANDERT WERDEN SOLL

VERARBEITUNG DER ANWEISUNG

- FÜR JEDES TUPEL WIRD WHERE-KOMPONENTE GEPRÜFT.
ERGEBNIS TRUE => KOPIE DES TUPELS WIRD ERZEUGT
- AUF BASIS DER WERTE IN DER KOPIE UND DER SET-KOMPONENTE WERDEN DIE NEUEN WERTE FÜR DIE ATTRIBUTE BERECHNET
- ERGEBNIS KOMMT IN DAS URSPRÜNGLICHE TUPEL
 - KOPIE DES TUPELS WIRD GELÖSCHT.

```
UPDATE PERSONAL  
SET KRANKENKASSE='TKK', GEH_STUFE='it5'  
WHERE NAME='Graf';
```

```
UPDATE GEHALT  
SET BETRAG=BETRAG * 1.07;
```

AUFGABEN

1. BENENNEN SIE DIE ABTEILUNG 'FORSCHUNG' IN 'FORSCHUNG/
BILDUNG' UM.
2. DIE NEURE BOHRMASCHINE HEISST NUN 'BOHRMASCHINE 3000'.

DELETE

```
DELETE  
[FROM] {tabellenname | viewname}  
[WHERE suchbedingung]
```

- ZEILEN EINER TABELLE WERDEN GELOSCHT, DIE DIE WHERE-BEDINGUNG ERFÜLLEN
- ALLE ZEILEN WERDEN GELÖSCHT, WENN KEINE WHERE-BEDINGUNG ANGEGEBEN IST
- UNTERABFRAGEN IN DER WHERE – KOMPONENTE DÜRFEN SICH NICHT AUF DIE TABELLE BEZIEHEN, IN DER ZEILEN GELÖSCHT WERDEN SOLLEN

```
DELETE FROM PERSONAL  
WHERE PNR = 167;
```

```
DELETE FROM PRAEMIE  
WHERE PNR = ( SELECT PNR  
              FROM PERSONAL  
              WHERE NAME = "Hahn"  
            );
```

AUFGABEN

1. FUGEN SIE EINEN NEUEN MITARBEITER UND EIN KIND FÜR DIESEN MITARBEITER EIN UND LÖSCHEN SIE BEIDE DANACH.
2. LÖSCHEN SIE ALLE PRÄMIEN UNTER 100 EUR.

DROP TABLE

```
CREATE TABLE KAFFEE_KONSUM (  
    PNR INT,  
    MASCHINE_ID INT,  
    MENGE INT,  
    FOREIGN KEY (PNR) REFERENCES PERSONAL(PNR),  
    FOREIGN KEY (MASCHINE_ID) REFERENCES KAFFEE_MASCHINE(ID)  
);
```

```
CREATE TABLE KAFFEE_MASCHINE (  
    ID INT PRIMARY KEY,  
    NAME VARCHAR(20)  
);
```

```
DROP TABLE KAFFEE_MASCHINE;  
DROP TABLE KAFFEE_KONSUM;
```

CREATE SEQUENCE


```
CREATE SEQUENCE a_sample_SEQ  
  INCREMENT BY 1 START WITH 1;
```

```
SQL> CREATE SEQUENCE matrikel_seq;
```

Sequence created.

```
SQL> CREATE TABLE Student (  
    matrikelnummer INTEGER DEFAULT nextval('matrikel_seq'),  
    name VARCHAR(30)  
);  
matrikelnummer INTEGER DEFAULT nextval('matrikel_seq'),  
                                *
```

ERROR at line 2:

ORA-04044: procedure, function, package, or type is not allowed here

```
SQL> CREATE TABLE Student (  
matrikelnummer INTEGER ,  
name VARCHAR(30)  
);
```

Table created.

```
SQL> INSERT INTO Student VALUES (matrikel_seq.nextval, 'John Doe');
```

1 row created.

```
SQL> select * from Student;
```

```
MATRIKELNUMMER NAME
```

```
-----
```

```
1 John Doe
```

MYSQL

```
CREATE TABLE Student (  
    matrikelnummer SERIAL,  
    name VARCHAR(30)  
);
```

```
CREATE TABLE Student (  
    matrikelnummer INTEGER AUTO_INCREMENT,  
    name VARCHAR(30)  
);
```

POSTGRESQL

```
CREATE TABLE tablename (  
    colname SERIAL  
);
```

```
CREATE SEQUENCE tablename_colname_seq;  
CREATE TABLE tablename (  
    colname integer DEFAULT nextval('tablename_colname_seq') NOT NULL  
);
```

```
CREATE SEQUENCE personal_SEQ  
    INCREMENT BY 1 START WITH 200;
```

```
INSERT INTO PERSONAL (PNR, VORNAME, NACHNAME, GEH_STUFE, ABT_NR, KRANKENKASSE)  
    VALUES (personal_SEQ.NEXTVAL, 'Ruth', 'Teitelbaum', 'it4', NULL, 'aok');
```

AUFGABEN

1. ERSTELLEN SIE EINE SEQUENCE FÜR DIE MASCHINE-TABELLE

ALTER TABLE

```
ALTER TABLE tabellenname ADD  
    (spaltenname datentyp [NULL]  
    [,spaltenname datentyp [NULL]...])
```


- TABELLE WERDEN UM DIE GENANNTE ATTRIBUTE ERWEITERT
 - SPALTEN WERDEN MIT NULL-WERTEN VORBESETZT
- SPEZIFIKATION NOT NULL IST NICHT ERLAUBT (NUR BEI LEEREN TABELLEN).

```
ALTER TABLE PERSONAL  
  ADD V_NR number(4);
```

```
ALTER TABLE MASCHINE  
  ADD CONSTRAINT FK_MASCHINE FOREIGN KEY(PNR) REFERENCES PERSONAL (PNR);
```

```
ALTER TABLE MASCHINE  
  DROP CONSTRAINT FK_MASCHINE;
```

MODIFIZIEREN DER SPALTE EINER TABELLE

- > ANDERN DES DATENTYPS
- > ÄNDERN DER GRÖßE
- > ÄNDERN DES DEFAULT-WERTES

EINSCHRÄNKUNGEN:

- VERKLEINERUNGEN SIND NUR MÖGLICH, WENN DIE SPALTE NUR NULL-WERTE ODER DIE TABELLE KEINE ZEILEN ENTHÄLT
- EIN WECHSEL DES DATENTYPS IST NUR MÖGLICH, WENN DIE SPALTE NUR NULL-WERTE ODER DIE TABELLE KEINE ZEILEN ENTHÄLT
- EIN WECHSEL DES DEFAULT-WERTES WIRKT SICH NUR IN DER ZUKUNFT AUS

```
ALTER TABLE PERSONAL  
    MODIFY (KRANKENKASSE char(5));
```

```
ALTER TABLE MASCHINE  
    MODIFY (ANSCH_DATUM date default sysdate);
```

AUFGABEN

1. ANDERN SIE DIE TABELLE ABTEILUNG SO, DASS DER NEUE TITEL FUER 'FORSCHUNG/ BILDUNG' MIT 'FORSCHUNG UND ALLGEMEINE WEITERBILDUNG' EINGETRAGEN WERDEN KANN.
2. FÜGEN SIE DEN TABELLEN Kind UND Praemie EINEN KÜNSTLICHEN SCHLÜSSEL HINZU UND AKTUALISIEREN SIE DAZU DIE TABELLENEINTRÄGE.
3. PASSEN SIE ALLE VORHANDENEN TABELLEN SO AN, DASS SIE EINE SEQUENCE ZUR ERSTELLUNG IHRER PRIMÄRSCHLÜSSEL BENUTZEN.
4. FÜGEN SIE NUN FÜR JEDE TABELLE, EINEN NEUEN DATENSATZ EIN, INDEM SIE IN DEM INSERT - STATEMENT DIE SEQUENCE BENUTZEN.

DATA DICTIONARY

- METADATEN WERDEN IM DATA DICTIONARY (CATALOG, INFORMATION SCHEMA)
 - GENAUSO ABFRAGBAR WIE REGULÄRE DATEN
- SELECT-ANFRAGE KANN MAN HERAUSFINDEN, WELCHE TABELLEN IN EINER DATENBANK EXISTIEREN
- DATA DICTIONARY IST IN SQL GENORMT, UND DIE MEISTEN HERSTELLER ERGÄNZEN ES UM SINNVOLLE SICHTEN

GIBT EINE LISTE ALLER TABELLENNAMEN ZURÜCK, DIE DEM
JEWEILIGEN BENUTZER GEHÖREN

```
SELECT Table_Name  
FROM User_Tables ;
```

GIBT EINE LISTE ALLER TABELLENNAMEN ZURÜCK, DIE
IRGEND EINEM BENUTZER GEHÖREN

```
SELECT Table_Name  
FROM All_Tables ;
```

WELCHE TABELLEN SIND IM DICTIONARY? DAFUR GIBT ES WIEDER
EINE TABELLE „DICTIONARY“.

LIEFERT ALLE TABELLEN ZURUCK, AUF DIE MAN IM DATA
DICTIONARY ZUGRIFF HAT.

```
SELECT Table_Name  
FROM Dictionary ;
```

ES GIBT AUCH EINE REIHE VON TABELLEN, DIE MIT V\$ STARTEN.
DIESE TABELLEN WERDEN ZUR LAUFZEIT MIT DYNAMISCHEN
INFORMATIONEN VERSEHEN

LIEFERT INFORMATIONEN UBER AKTUELLE PROZESSE DES DBMS

```
SELECT *  
FROM V$PROCESS ;
```

TABELLENBESCHREIBUNG

describe PERSONAL;

A stylized, flat-design illustration of Rainbow Dash from the animated series My Little Pony: Friendship is Magic. She is depicted in a dynamic, flying pose, angled towards the right. Her body is a light blue color, and her wings are spread wide. Her most prominent feature is her long, flowing mane and tail, which are composed of multiple overlapping bands of color in the rainbow spectrum: red, orange, yellow, green, blue, and purple. Her eyes are large and expressive, with black outlines and pinkish-purple irises. The background is a solid, dark grey-blue color. Overlaid on the image is the German text "DAS WAR'S FÜR HEUTE" in a large, white, sans-serif font. The text is split across two lines: "DAS WAR'S FÜR" on the top line and "HEUTE" on the bottom line. The letters are bold and have a slight shadow, making them stand out against the background and the character.

DAS WAR'S FÜR
HEUTE