

# Wiederholung

DML – Select, Join, Union, Unterabfragen

## Oracle Online-Dokumentation – Database Administration

[http://docs.oracle.com/database/121/nav/portal\\_4.htm](http://docs.oracle.com/database/121/nav/portal_4.htm)

- [Database SQL Language Reference](#)
- [Database SQL Language Quick Reference](#)

## [Zugriff über die FH-Onlinebibliothek](#)

- **Datenbanken und SQL, Edwin Schicker** - Springer Vieweg
- **Datenbanken für Wirtschaftsinformatiker, Sönke Cordts** - Vieweg + Teubner

## O'Reilly Taschenbibliothek (9,90€)

- **SQL kurz & gut, Jonathan Gennick**  
2. Auflage November 2006 – ISBN: 978-3-89721-522-1
- **Oracle PL/SQL – kurz & gut, Steven Feuerstein, Bill Pribyl, Chip Dawes**  
4. Auflage März 2008, ISBN: 978-3-89721-538-2

## Alle Spalten (Attribute) und Zeilen (Datensätze) einer Tabelle anzeigen

```
SELECT *  
FROM tabellenname;
```

```
SELECT *  
FROM lib_author;
```

- Alle Attribute (\* = alle ) und alle Datensätze der Tabelle lib\_author anzeigen

## Ausgewählte Spalten einer Tabelle anzeigen

```
SELECT spaltenname, spaltenname...  
FROM tabellenname;
```

```
SELECT forename, surname  
FROM lib_author;
```

- Vorname und Nachname aller Datensätze in der Tabelle lib\_author anzeigen

# Filtern und Anzeigen der Daten

Nur Datensätze der Tabelle anzeigen, die bestimmte Bedingungen erfüllen

```
SELECT *  
FROM tabellenname  
WHERE condition;
```

## Nur Datensätze mit bestimmten Attributwerten anzeigen

```
SELECT *  
FROM lib_book  
WHERE title = 'Limit';
```

```
SELECT *  
FROM lib_book  
WHERE pages = 182;
```

- Alle Datensätze der Tabelle lib\_book, bei denen der Titel Limit ist (Achtung: case-sensitive)
- Alle Datensätze der Tabelle lib\_book, bei denen die Seitenzahl 182 ist

# Filtern und Anzeigen der Daten

Datensätze anzeigen, bei denen Attributwert größer/kleiner/ungleich einem Wert ist

```
SELECT *  
FROM lib_book  
WHERE pages > 300;
```

➡ Alle Bücher, die mehr als 300 Seiten haben

## Weiter Vergleiche:

größer (gleich)/kleiner (gleich): <, >, <=, >=

ungleich: != oder <>

Datensätze anzeigen, bei denen Attributwert in Liste von Werten ist

```
SELECT *  
FROM lib_book  
WHERE pages IN(376,182) ;
```

➡ Alle Datensätze, bei denen Seitenanzahl 376 oder 182 ist

## LIKE und Wildcards

Ein Wert kann mit dem LIKE-Operator auch auf ein **bestimmtes Muster** überprüft werden.

Dazu werden Wildcards (Platzhalter) verwendet

%: kein bis beliebig viele Zeichen

\_: genau ein Zeichen

```
SELECT *  
FROM lib_author  
WHERE surname LIKE 'M%';
```

- Alle Datensätze, bei denen der Nachname mit großem M anfängt:  
z.B: Mustermann, Meyer  
*Nicht: mueller, meier...*

```
SELECT *  
FROM lib_author  
WHERE surname LIKE 'M_stermann';
```

- Alle Datensätze, bei denen Nachname mit großem M anfängt, dann ein Zeichen und dann *stermann* folgt:  
z.B: Mastermann, Mistermann....  
*Nicht: Meistermann, mastermann*

## Verknüpfung von Bedingungen

Mit Hilfe der Operatoren AND und OR können Bedingungen miteinander verknüpft werden

```
SELECT *  
FROM lib_book  
WHERE title LIKE 'M%'  
      AND pages > 200;
```

- Alle Datensätze, bei denen der Titel mit großem M anfängt und die Seitenanzahl größer als 200 ist

```
SELECT *  
FROM lib_book  
WHERE pages > 100  
      OR pages < 50;
```

- Alle Datensätze, bei denen die Seitenanzahl größer 100 oder kleiner 50 ist.

# Filtern und Anzeigen der Daten

## Negation von Bedingungen

Der Operator NOT führt dazu, dass alle Datensätze angezeigt werden, bei denen die Bedingung nicht zutrifft.

```
SELECT *  
FROM lib_book  
WHERE pages NOT IN(100, 200, 300);
```

- Alle Datensätze, bei denen die Seitenzahl **NICHT** 100, 200 oder 300 ist

## Filtern von NULL-Werten

```
SELECT *  
FROM lib_book  
WHERE year IS NULL;
```

- Alle Datensätze, bei denen das Jahr NULL ist  
(hier ist auch **IS NOT NULL** möglich)

*Achtung, da NULL für einen nicht definierten Wert steht, liefern Vergleiche mit NULL niemals TRUE zurück.*

*SELECT .... WHERE year = NULL → kein Ergebnis*



# Filtern und Anzeigen der Daten

- Um Duplikate (mehrfach vorkommende Werte) zu unterdrücken, kann der **DISTINCT-Befehl** verwendet werden

```
SELECT DISTINCT *  
FROM lib_book;
```

- Unterdrückt Wiederholung identischer Datensätze

```
SELECT DISTINCT title, year  
FROM lib_book;
```

- Jede vorhandene Kombination aus Titel und Jahr wird nur einmal angezeigt

- Daten werden oft auf mehrere Tabellen verteilt, um Redundanzen zu vermeiden
- Um diese verteilten Daten in einer SELECT-Abfrage wieder vereinigt anzuzeigen, werden die Tabellen über einen so genannten JOIN-Befehl verbunden
- Man unterscheidet unter anderem zwischen
  - **INNER JOIN**
  - **OUTER JOIN**
  - **LEFT JOIN**
  - **RIGHT JOIN**

# Tabellen-Join

- Anzeige von Büchertiteln und deren Kategorie
- Dazu werden Informationen aus den Tabellen lib\_book und lib\_category benötigt
- Verbindung zwischen den Tabellen kann über Spalte cat\_id hergestellt werden

lib\_book

BOOK_ID	TITLE	...	CAT_ID
1	Limit	...	1
2	Der Schwarm	...	1
3	PL/SQL	...	2
4	Dummy	...	NULL

Lib\_category

CAT_ID	CAT_NAME
1	Roman
2	Fachbuch
3	Kinderbuch



# INNER JOIN

```
SELECT *  
FROM lib_book  
INNER JOIN lib_category  
ON lib_book.cat_id = lib_category.cat_id;
```

**lib\_book**

BOOK_ID	TITLE	...	CAT_ID
1	Limit	...	1
2	Der Schwarm	...	1
3	PL/SQL	...	2
4	Dummy	...	NULL

**lib\_category**

CAT_ID	CAT_NAME
1	Roman
2	Fachbuch
3	Kinderbuch

**JOIN Ergebnis**

BOOK_ID	TITLE	....	CAT_ID	CAT_ID	CAT_NAME
1	Limit	...	1	1	Roman
2	Der Schwarm	...	1	1	Roman
3	PL/SQL	..-	2	2	Fachbuch

INNER JOIN liefert nur Datensätze, bei denen cat\_id links einer cat\_id rechts zugeordnet werden kann

# UNION

- Über den UNION-Befehl können Datensätze mehrerer Abfragen in einem Ergebnis zusammengefasst werden
- Dabei werden mehrfach vorhandene Datensätze entfernt (alternativ UNION ALL: ohne Unterdrückung mehrfach vorkommender Datensätze)
- Dazu müssen Datentypen und Anzahl der Spalten beider Abfragen gleich sein

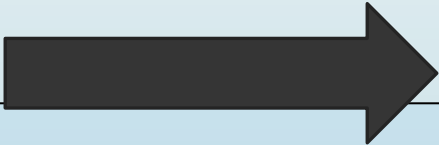
**Beispiel:** Es liegen zwei Tabellen EMPA und EMPB vor, in denen Mitarbeiterdaten für zwei Unternehmensbereiche separat erfasst werden

**EMPA**

EMPNO	ENAME	HIREDATE	SAL
7782	CLARK	03.03.1980	2300
7839	KING	01.01.1980	1500
7369	SMITH	02.02.1981	1700

**EMPB**

EMPNO	NAME	HDATE	SALARY
7499	ALLEN	15.03.1983	3500
7902	FORD	01.05.1991	2800



```
SELECT ename, sal
FROM empa
UNION
SELECT name, salary
FROM empb
```

**Ergebnis**

ENAME	SAL
CLARK	2300
KING	1500
SMITH	1700
ALLEN	3500
FORD	2800

# Unterabfragen – Rückgabe eines Wertes

In einer Abfrage kann eine weitere Abfrage geschachtelt werden → **Unterabfrage**

## Anwendungsbsp:

Ausgabe aller Bücher, die im selben Jahr erschienen sind wie Limit

### Option A

#### 1. In welchem Jahr ist Limit erschienen?

```
SELECT year  
FROM lib_book  
WHERE title = 'Limit';  
→ Year 2011
```

#### 2. Welche Bücher sind 2011 erschienen?

```
SELECT title  
FROM lib_book  
WHERE year = 2011;
```

### Option B

#### 1. Welche Bücher sind im selben Jahr wie Limit erschienen?

Im WHERE-Bereich wird **2011** durch eine Abfrage ersetzt, die das Erscheinungsjahr für Limit zurückliefert

```
SELECT title  
FROM lib_book  
WHERE year = ( SELECT year FROM lib_book  
               WHERE title = 'Limit' );
```

Unterabfrage muss wie hier **genau einen Wert** zurückliefern, dann können  
**Vergleichsoperatoren** =, <, >, <=, >= verwendet werden

# Unterabfragen – Rückgabe mehrerer Werte

Liefert die Unterabfrage eine Liste von Werten zurück, kann mit dem IN-Operator die Gleichheit mit einem Wert aus der Liste geprüft werden

## Anwendungsbsp:

Ausgabe aller Bücher, die im selben Jahr erschienen sind wie Limit oder Java 5

### Option A

1. In welchem Jahr sind Limit und der Schwarm erschienen?

```
SELECT year
FROM lib_book
WHERE title IN ('Limit', 'Java 5');
→ 2011, 2006
```

2. Welche Bücher sind 2011 oder 2006 erschienen?

```
SELECT title
FROM lib_book
WHERE year IN (2011, 2006);
```

### Option B

1. Welche Bücher sind im selben Jahr erschienen wie Limit oder Java 5?

```
SELECT title
FROM lib_book
WHERE year IN
    ( SELECT year FROM lib_book
      WHERE title IN ('Limit', 'Java 5') );
```