Einführung in die Wirtschaftsinformatik

Teil 8 – Erweiterte Funktionen

Wintersemester 2020/2021

Lehrstuhl für Wirtschaftsinformatik
Prozesse und Systeme
Universität Potsdam

Chair of Business Informatics
Processes and Systems
University of Potsdam

Univ.-Prof. Dr.–Ing. habil. Norbert Gronau
Lehrstuhlinhaber | *Chairholder*

Karl-Marx-Str. 67 | 14482 Potsdam | Germany

Tel +49 331 977 3322
Fax +49 331 977 3406

E-Mail ngronau@lswi.de
Web lswi.de

Operatoren in zusammengesetzten Bedingungen

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Operatoren zum Vergleich von Werten

- Innerhalb der WHERE-Klausel - Vergleich für Zeichen sowie Zeichenketten, numerische und Datumswerte

```
SELECT ... FROM ...
WHERE ausdruck operator wert;
```

Operator	Bedeutung
=	Gleich
>	Größer als
>=	Größer/gleich
<	Kleiner als
<=	Kleiner/gleich
<>	Ungleich

```
SELECT name, gehalt FROM mitarbeiter
WHERE gehalt >= 2500;
```

Quelle: Greenberg 2002, S. 97

Spezielle Vergleichsoperatoren

- Operatoren mit stärkerer bzw. erweiterter Filterwirkung
- Explizite Erfassung leerer Felder über IS NULL-Operator

Operator	Bedeutung
BETWEEN … AND …	Zwischen zwei Werten (einschließlich dieser Werte)
IN (*Werteliste*)	Entspricht einem oder mehreren Werte aus einer Werteliste
LIKE	Entspricht einem zu definierenden Zeichenmuster
IS NULL	Ist ein NULL-Wert

Quelle: Greenberg 2002, S. 99

Einsatz der Operatoren BETWEEN und IN

BETWEEN – Anzeigen von Zeilen basierend auf Wertebereich

```
SELECT name, gehalt
FROM mitarbeiter
WHERE gehalt BETWEEN 2500 AND 3500;
```

NAME	GEHALT
Gast	2600
Dost	3100
…	…
Fritzsche	2910
Bode	2550

Zugehörigkeitsoperator IN – Prüfung einer Werteliste

```
SELECT name, gehalt, abt_nr
FROM mitarbeiter
WHERE abt_nr IN
('230P','240P','107R');
```

NAME	GEHALT	ABT_NR
Groß	2800	230P
Eckert	4270	230P
…	…	…
Van Der Biest	5180	107R
Nograsek	3770	107R
…	…	…
Schröder	4400	240P

Der Operator LIKE für eine bessere Suche

LIKE für Zeichenfolgenwerte

```
SELECT name, vorname, anrede
FROM mitarbeiter
WHERE name LIKE 'Sch%';
```

NAME	VORNAME	ANREDE
Scherz	Nicole	Frau
Schöneck	Sascha	Herr
…	…	…
Schulz	Carola	Frau
Schlank	Nils	Herr

Die Platzhalterzeichen % (0 bis n Zeichen) und _ (exakt 1 Zeichen) in LIKE können kombiniert werden.

Logische Operatoren

Verknüpfen mehrerer Bedingungen in einer Abfrage

Operator	Bedeutung
AND	Gibt TRUE zurück, wenn beide Komponentenbedingungen wahr sind
OR	Gibt TRUE zurück, wenn mindestens eine der beiden Komponentenbedingungen wahr ist
NOT	Gibt TRUE zurück, wenn die nachfolgende Bedingung falsch ist

Quelle: Greenberg 2002, S. 105

Operator AND

AND – Erfüllung beider Bedingungen (wahr)

```
SELECT pers_nr, name, position, gehalt
FROM mitarbeiter
WHERE gehalt >= 6500 AND position LIKE '%leiter%';
```

PERS_NR	NAME	POSITION	GEHALT
101001	Büchner	Hauptabteilungsleiter	10430
101007	Kettler	Abteilungsleiter	8080
101016	Klein	Abteilungsleiter	7960
101042	Schmiedel	Abteilungsleiterin	7210
101015	Grauer	Abteilungsleiterin	6600

AND-Wahrheitstabelle

AND		Erste Bedingung ist…		
		WAHR	FALSCH	NULL
Zweite Bedingung ist…	WAHR	WAHR	FALSCH	NULL
	FALSCH	FALSCH	FALSCH	FALSCH
	NULL	NULL	FALSCH	NULL

Quelle: Greenberg 2002, S. 106

Operator OR

OR – mindestens eine der beiden Bedingungen muss wahr sein

```
SELECT pers_nr, name, position, gehalt
FROM mitarbeiter
WHERE gehalt >= 4500 OR position LIKE '%käufer';
```

PERS_NR	NAME	POSITION	GEHALT
101001	Büchner	Hauptabteilungsleiter	10430
101003	Dost	Einkäufer	3100
…	…	…	…
101056	Hein	Einkäufer	2600
101059	Ernst	Abteilungsleiter	6500
…	…	…	…

OR-Wahrheitstabelle

OR		Erste Bedingung ist…		
		WAHR	FALSCH	NULL
Zweite Bedingung ist…	WAHR	WAHR	WAHR	WAHR
	FALSCH	WAHR	FALSCH	NULL
	NULL	WAHR	NULL	NULL

Quelle: Greenberg 2002, S. 107

Operator NOT

NOT – im Sinne von "keine Übereinstimmung"

```
SELECT name, vorname, position
FROM mitarbeiter
WHERE position NOT IN
('Abteilungsleiterin','Abteilungsleiter','Hauptabteilungsleiter');
```

NAME	VORNAME	POSITION
Martens	Eugen	Dreher
Dost	Alexander	Einkäufer
…	…	…
Neumann	Michael	Systementwicklungsingenieur
Altmann	Bernd	Lagerist

NOT-Wahrheitstabelle

NOT	Erste Bedingung ist…		
	WAHR	FALSCH	NULL
Ergebnis ist	FALSCH	WAHR	NULL

Quelle: Greenberg 2002, S. 108

Prioritätsregeln der Operatoren

Auswertungsreihenfolge	Operator
1	Arithmetische Operatoren
2	Verkettungsoperator
3	Vergleichsoperatoren
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	Logischer Operator NOT
7	Logischer Operator AND
8	Logischer Operator OR

Die Auswertungsreihenfolge lässt sich durch Setzen von Klammern ändern.

Quelle: Greenberg 2002, S. 109

Zusammenfassung – Projektion und Selektion

```
SELECT name, vorname
```

Projektion

```
FROM mitarbeiter

   WHERE abt_nr <> '20'
AND position = 'Buchhalter'
```

Selektion

Durch Projektion werden die Spalten, durch Selektion die Zeilen einer Tabelle ausgewählt.

Operatoren in zusammengesetzten Bedingungen

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Sortierung mit ORDER BY

Sortierung von Ausgabezeilen

- Aufsteigende Reihenfolge (Grundeinstellung) – ASC (ascending)
- Absteigende Reihenfolge – DESC (descending)
- Klausel steht am Ende der SELECT-Anweisung

```
SELECT ausdruck
FROM tabelle
[WHERE bedingung(en)]
[ORDER BY {spalte|ausdruck} [ASC|DESC]];
```

Sortierung in auf- und absteigender Reihenfolge

```
SELECT name, vorname, position
FROM mitarbeiter
ORDER BY position ASC;
```

NAME	VORNAME	POSITION
Rösch	Konrad	Abteilungsleiter
Beyer	Maximilian	Abteilungsleiter
…	…	…
Müller-Eickhof	Petra	Wirtschaftsingenieurin
Jürgens	Maximilian	Zeichner

Aufsteigende Sortierung der Spalte POSITION

```
SELECT name, vorname, position
FROM mitarbeiter
ORDER BY position DESC;
```
Absteigende Sortierung

Die aufsteigende Sortierung ist als Standard gesetzt. Identische Werte werden willkürlich sortiert.

Sortierung nach Spalten-Aliasnamen

```
SELECT name, vorname, gehalt * 12 Jahresgehalt
FROM mitarbeiter
ORDER BY Jahresgehalt DESC;
```

NAME	VORNAME	JAHRESGEHALT
Reinhard	Bernd	672000
Johansson	Grit	420000
Klemm	Ljudmilla	384000
…	…	…
Schuster	Anika	6000
Kohl	Melanie	6000

Sortierung nach mehreren Spalten

- Bestimmung der Sortierreihenfolge durch Angabe nach ORDER BY

```
SELECT name, vorname, gehalt
FROM mitarbeiter
ORDER BY gehalt DESC, name ASC;
```

NAME	VORNAME	GEHALT
Reinhard	Bernd	56000
Johansson	Grit	35000
Klemm	Ljudmilla	32000
Krajcsir	Martin	32000
…	…	…
Schuster	Jens	700
Assmann	Niels	500
Kohl	Melanie	500
Schuster	Anika	500

Sortierung erfolgt zuerst nach Gehalt und dann nach Name

Eine Spaltensortierung ist auch nach nicht nach SELECT angegebenen Spalten möglich.

Operatoren in zusammengesetzten Bedingungen

Sortierung von Ergebnissen

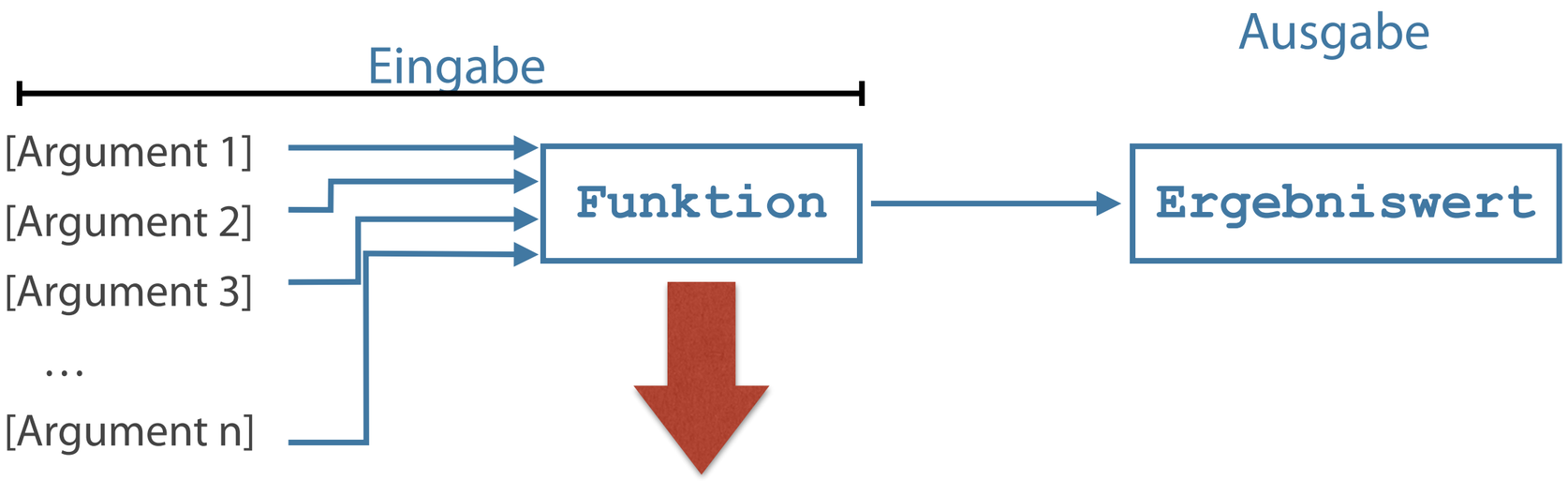
Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

SQL-Funktionen

- Bearbeitung von Zeilen und Ausgabe von Ergebnissen dieser Bearbeitung



SQL-Funktionen enthalten manchmal Argumente und geben immer einen Wert zurück.

Quelle: Greenberg 2002, S. 125

Merkmale von Single Row Funktionen

Datenelemente

- Bearbeitung jeder zurückgegebenen Zeile aus einer Hauptabfrage

Argumente und Werte

- Spalten oder Ausdrücke – als Argumente akzeptiert
- Rückgabe eines Ergebnisses als Wert je Zeile

Erläuterung der Syntax

- funktionsname – Name der Funktion
- argument1, argument2 – die von der Funktion verwendeten Argumente (Spaltenname, Ausdruck)

```
funktionsname [(argument1, argument2,...)];
```

Quelle: Greenberg 2002, S. 127

Übersicht Single Row Funktionen

Zeichenfunktionen

- Rückgabe von Zeichen- oder numerischen Werten

Konvertierungsfunktionen

- Konvertierung eines Wertes von einem Datentyp in einen anderen

Numerische Funktionen

- Rückgabe numerischer Werte

Datumsfunktionen

- Rückgabe eines Wertes vom Datentyp DATE

Übersicht der Zeichenfunktionen

Groß-/Kleinschreibung

- UPPER
- LOWER
- INITCAP

Bearbeitung von Zeichen

- CONCAT
- SUBSTR
- LENGTH
- INSTR
- TRIM
- REPLACE
- LPAD
- RPAD

Zeichenfunktionen ermöglichen vielfältige Zeichenmodifikationen und -manipulationen.

Funktionen zur Umwandlung der Groß- bzw. Kleinschreibung

Syntax

```
SELECT [LOWER|UPPER|INITCAP]   (ausdruck|'zeichenfolge')
FROM tabelle;
```

Funktion	Ergebnis
LOWER('SQL Anweisung')	sql anweisung
UPPER('SQL Anweisung')	SQL ANWEISUNG
INITCAP('SQL Anweisung')	Sql Anweisung

Quelle: Greenberg 2002, S. 131

Funktion zur Groß-/Kleinschreibung

Bei falscher Zeichensetzung (Groß-, Kleinschreibung nicht beachtet) – Ausgabe erfolglos

```
SELECT name, vorname, gehalt
FROM mitarbeiter
WHERE position = 'chief executive officer';
```

```
No rows selected
```

Lösung mit INITCAP

```
SELECT name, vorname, gehalt
FROM mitarbeiter
WHERE position = INITCAP('chief executive officer');
```

NAME	VORNAME	GEHALT
Reinhard	Bernd	56000

Funktionen zum Bearbeiten von Zeichen

Funktion	Ausgabe
CONCAT('Betriebsteil', 'Potsdam')	BetriebsteilPotsdam
SUBSTR('BetriebsteilPotsdam',1,12)	Betriebsteil
LENGTH('BetriebsteilPotsdam')	19
INSTR('Betriebsteil', 's')	8
LPAD(gehalt,10,'*')	******3500
RPAD(gehalt, 10, '*')	3500******
TRIM('P ' FROM 'Potsdam')	otsdam

Quelle: Greenberg 2002, S. 133

Bearbeiten von Zeichen für die Ausgabe

```
SELECT pers_nr, CONCAT(vorname, name) NAME, position,
       LENGTH(name) LÄNGE, INSTR(name,'o') "Enthält 'o'?"
FROM mitarbeiter
WHERE SUBSTR(position,4) = 'käufer';
```

Pers_nr	Name	Position	Länge	Enthält 'o'?
101003	AlexanderDost	Einkäufer	4	2
101025	HelmutPetersen	Einkäufer	8	0
101056	HugoHein	Einkäufer	4	0

1 4 2 3

Quelle: Greenberg 2002, S. 134

Numerische Funktionen

ROUND – Rundung eines Wertes auf eine vorgegebene Dezimalstelle

```
Beispiel: ROUND(232.667, 2) 232.67
```

TRUNC – Abschneiden eines Wertes bis zu einer bestimmten Dezimalstelle

```
Beispiel: TRUNC(232.667, 2) 232.66
```

MOD – Rückgabe des Restes einer Division

```
Beispiel: MOD(232, 56) 8
```

Rechnung: 232/56 = 4 Rest 8

ROUND rundet nach dem mathematischen Prinzip auf oder ab (Stellenwert >=5 auf, <5 ab).

Quelle: Greenberg 2002, S. 135

Operatoren in zusammengesetzten Bedingungen

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Wiederholung Vorlesung 7: NULL-Werte in Feldern

- In der letzten VL Probleme mit NULL Werten:
- Auch beim Rechnen mit Nullwerten treten Probleme auf

```
SELECT name, vorname, akad_titel
FROM mitarbeiter;
```

NAME	VORNAME	AKAD_TITEL
Walker	John A.	
Melzer	Thomas	
Bormann	Samira	Dr.
Hofmann	Katja	
Würz	Hannah	
…		
Petersen	Helmut	
Schulz-	Paul	Dr. Ing.
Plenk	Karl	
Engel	Lothar	
Roth	Katharina	

…

Leerwerte:

Dieses Problem werden wir in einer der nächsten VL lösen

```
SELECT name, Vorname, gehalt*12*provision Jahresgehalt
FROM mitarbeiter;
```

NAME	VORNAME	JAHRESGEHALT
Christ	Adrien	2880
Mehmedovic	Ahmad	-
Winter	Tanja	-
…	…	…
Malossek	Benjamin	1886,4

NULL*gehalt*12 = NULL

Greenberg 2002, S. 64

IS NULL für den Test auf Nullwerte

Zellen ohne Werte mit IS NULL

```
SELECT name, position, abt_nr
FROM mitarbeiter
WHERE abt_nr IS NULL;
```

NAME	POSITION	ABT_NR
Riekhoff	Chefsekretärin	-
Johansson	Chief Operations Officer	
Metz	Chefsekretärin	…
…	…	-
Walker	Chief Information Officer	-

Zellen ohne Werte mit anderen Operatoren

```
SELECT name, position, abt_nr
FROM mitarbeiter
WHERE abt_nr  = '';
```

```
No data found
```

Behandlung von NULL-Werten

Angabe konkreter Werte in `ausdruck`

- NVL (ausdruck1, ausdruck2|wert)
- NVL2 (ausdruck1, ausdruck2|wert1, ausdruck3|wert2)
- COALESCE (ausdruck1, ausdruck2, ..., ausdruckn)

Vergleich und Ausgabe von `ausdruck` oder NULL-Wert

- NULLIF (ausdruck1, ausdruck2)

Diese Funktionen können für alle Datentypen eingesetzt werden.

Quelle: Greenberg 2002, S. 167

Funktion NVL

- Konvertierung von NULL-Werten in konkrete Werte bei DATE, CHARACTER, NUMBER
- Forderung – Übereinstimmung der Datentypen

```
NVL(ausdruck1, ausdruck2)
```

Datentyp NUMBER

```
Beispiel 1: NVL(gehalt,3300)
```

```
Beispiel 2: NVL(proj_kosten,0)
```

Datentyp CHAR oder VARCHAR2

```
Beispiel 3: NVL(proj_name,'Nicht verfügbar')
```

```
Beispiel 4: NVL(position,'Transportarbeiter')
```

Quelle: Greenberg 2002, S. 168f.

Funktion NVL mit numerischem Rückgabewert

```
SELECT name, gehalt, NVL(provision,0)provision,
(gehalt*12*(1+NVL(provision,0))) Jahresgehalt
FROM mitarbeiter;
```

1

2

- Berechnung Jahresgehalt aller Angestellten
- Multiplikation von Gehalt, Anzahl Monate und Provisionssatz
- Problemstellung: Provisionssatz nur für Verkäufer --> alle anderen Felder der Spalte sind leer

NAME	GEHALT	PROVISION	JAHRESGEHALT
Büchner	10430	0	125160
Martens	2400	0	28800
Dost	3100	0	37200
Fuchs	3600	0	43200
...	...	...	...
Johnson	1540	0,15	21252
Poderni	1380	0,14	18878,4
Pommer	1460	0,13	19797,6
...	...		...
Altmann	1830	1 0	2 21960

Anwendung der Funktion NVL2

Festlegung des Rückgabewerts durch Inhalt des ersten Ausdrucks

- Rückgabe eines NULL-Wertes – Ausgabe des dritten Ausdrucks von NVL2
- Rückgabe von Werten – Ausgabe des zweiten Ausdrucks von NVL2

```
NVL2(ausdruck1, ausdruck2, ausdruck3)
```

```
SELECT name, gehalt, abt_nr, NVL(provision,0),
NVL2(provision,'Gehalt + Provision','Gehalt') Einkommen
FROM mitarbeiter WHERE abt_nr IN ('410V','107R');
```

Ersetzt bei einer Provision NULL den NVL Value durch 0

Wird ausgegeben wenn Provision NULL

Wird ausgegeben wenn Provision nicht NULL

NAME	GEHALT	ABT_NR	NVL(PROVISION,0)	EINKOMMEN
Probst	4510	410V	0	Gehalt
Peplinski	3050	410V	0	Gehalt
Petrova	2100	410V	0	Gehalt
De Ridder	5890	410V	0	Gehalt
Schöneck	1380	410V	0,13	Gehalt + Provision
Thyssen	1480	410V	0,13	Gehalt + Provision
…	…	…	…	…

Funktion NULLIF

Vergleich von zwei Ausdrücke

- Bei Gleichheit – Ausgabe NULL-Wert
- Bei Ungleichheit – Ausgabe ausdruck1

```
NULLIF(ausdruck1, ausdruck2)
```

Beispiel

```
SELECT name, NULLIF(position,'Abteilungsleiter') "Nicht
leitende Angestellte" FROM mitarbeiter;
```

Wenn position = 'Abteilungsleiter' wird Nullwert zurückgegeben

NAME	Nicht leitende Angestellte
...	...
Dost	Einkäufer
Fuchs	Sekretärin
Rösch	-
Beyer	-
Kettler	-
Schneider	Schleifer
...	...

NULL-Werte

Beyer's Position = 'Abteilungsleiter' —> Rückgabewert ist NULL

Funktion COALESCE

- Dient der Vermeidung von NULL-Werten
- Liefert aus Parameterliste den Wert eines Parameters zurück, der nicht NULL ist
- Wenn erster Ausdruck kein NULL-Wert – Rückgabe dieses Ausdrucks
- Rückgabe von ausdruck2,...,n dann, wenn vorhergehender Ausdruck NULL-Wert enthält

```
COALESCE(ausdruck1, ausdruck2, ... ausdruckn)
```

Liefert aus Parameterliste den Wert eines Parameters zurück, der nicht NULL ist

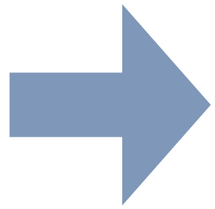
Wird zurückgegeben wenn ausdruck1 NULL ist

Wird zurückgegeben wenn Ausdrücke 1 bis n-1 alle NULL sind

Vorteil der Funktion COALESCE gegenüber der Funktion NVL ist die Angabe von mehr als zwei alternativen Werten.

Anwendung der Funktion COALESCE

LOHN	GEHALT	UMSATZ	PROVISION
2800			
	3100		
	3600		
		650000	0,065
3450			
		740000	0,065



Jahreseinkommen
33600
37200
43200
42250
41400
48100

- Alternative Anzeige in der Spalte "Jahreseinkommen"
- Je Zeile nur Auswahl des ersten Ausdrucks ungleich NULL
- Problem: Berechnung Jahreseinkommen bei mehr als einer Einkommensart (z. B. Gehalt + Umsatz) nicht möglich

```
SELECT ... COALESCE(12*lohn, 12*gehalt, umsatz*provision)
"Jahreseinkommen" FROM mitarbeiter;
```

➡ Hinweis: Das hier gezeigte Beispiel dient nur der theoretischen Betrachtung

Funktionen verschachteln

- Beliebige Verschachtelungstiefe der Single Row-Funktionen
- Auswertung der Funktionen erfolgt von innen nach außen

```
Fkt2(Fkt1(spalte,argument1),argument2)
```

Schritt 1 = Ergebnis 1

Schritt 2 = Ergebnis 2

```
SELECT name, NVL(TO_CHAR(leiter), 'Ohne Vorgesetzten')Vorgesetzter
FROM mitarbeiter WHERE leiter IS NULL;
```

NAME	VORGESETZTER
Kellner	Ohne Vorgesetzten
Reinhard	Ohne Vorgesetzten

Verschachtelte Funktionen werden grundsätzlich durch runde Klammern getrennt.

Quelle: Greenberg 2002, S. 165

Operatoren in zusammengesetzten Bedingungen

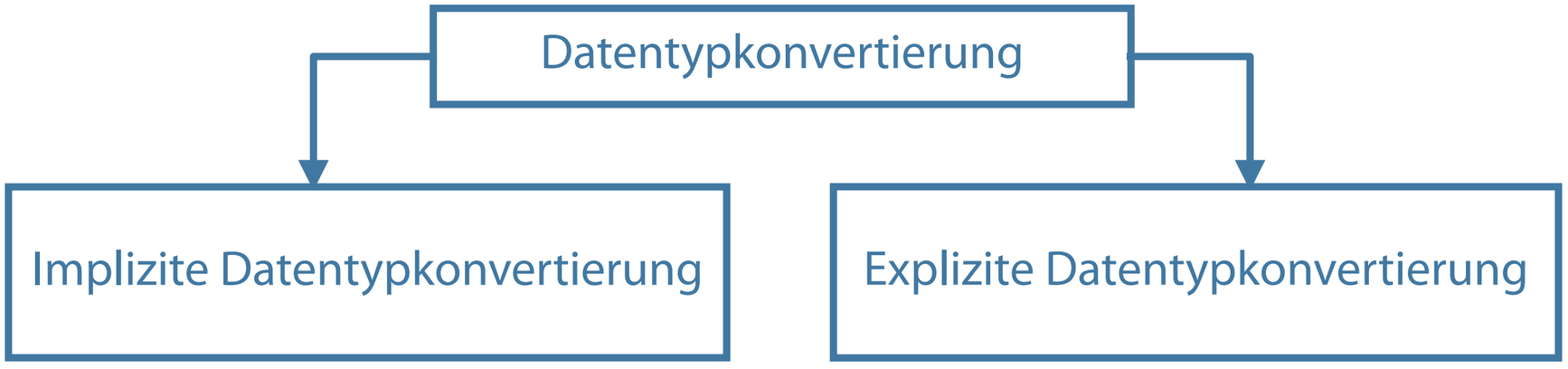
Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Übersicht der Konvertierungsfunktionen



- Wenn Daten aus einem Objekt A zu einem anderen Objekt B verschoben oder mit diesem verglichen oder kombiniert werden, müssen möglicherweise Daten des Datentyps des Objektes A zum Datentyp des Objektes B konvertiert werden
- Durchführung bestimmter Operationen nur bei Typengleichheit
- Implizite Datentypkonvertierungen durch Server
- Explizite Datentypkonvertierungen durch Benutzer mit Hilfe der Konvertierungsfunktionen

Explizite Datentypkonvertierungen bewirken zuverlässigere SQL-Anweisungen.

Quelle: Greenberg 2002, S. 147

Implizite Datentypkonvertierung

- Datenkonvertierung bei dem Einlesen von beispielsweise externen Datenquellen (z.B. CSV Dateien die keine Informationen zum Datentyp enthalten)
- Konvertierung erfolgt durch Oracle-Server automatisch
- Konvertierung der Datentypen beim Zuweisen

Ursprungsdatentyp	Zieldatentyp
VARCHAR2 oder CHAR	NUMBER
VARCHAR2 oder CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

- Konvertierung der Datentypen beim Auswerten von Ausdrücken

Ursprungsdatentyp	Zieldatentyp
VARCHAR2 oder CHAR	NUMBER
VARCHAR2 oder CHAR	DATE

Konvertierungen von CHAR in NUMBER sind nur erfolgreich, wenn die Zeichenfolge eine gültige Zahl darstellt.

Quelle: Greenberg 2002, S. 148f.

Explizite Datentypkonvertierung

Funktionen für Umwandlung eines Wertes von einem Datentyp in einen anderen

- Numerischer Wert/Datumswert → Zeichenkette

```
TO_CHAR (number|date,[format])
```

```
SELECT artikel_nr, TO_CHAR(net_preis*1.15,'L099,999.00') FROM
artikel
```

- Zeichenkette → Numerischer Wert

```
TO_NUMBER (char,[format])
```

➡ wird in der Regel nur benötigt, wenn numerische Werte als Textdaten im System gespeichert wurden und zur Berechnung herangezogen werden

- Zeichenkette → Datumswert

```
TO_DATE (char,[format])
```

```
SELECT TO_DATE('22 Oktober 2018','DD.MM.YYYY') FROM dual;
```

Funktion TO_CHAR mit Zahlenwerten

Übersetzung eines Wertes vom Datentyp NUMBER in VARCHAR2

```
TO_CHAR (number,[format_model])
```

Formatelemente (format_model)

	Beschreibung	Beispiel	Ausgabe
9	Anzahl anzuzeigender Ziffern	999999	1234
0	Erzwingt die Anzeige führender Nullen	‘099999‘	‘001234‘
$	Setzt ein führendes Dollarzeichen	‘$999999‘	‘$1234‘
L	Verwendet lokales Währungssymbol	L999999	‘€1234‘
.	Druckt einen Dezimalpunkt	999999.99	1234.00
,	Druckt ein Tausendertrennzeichen	999,999	1,234

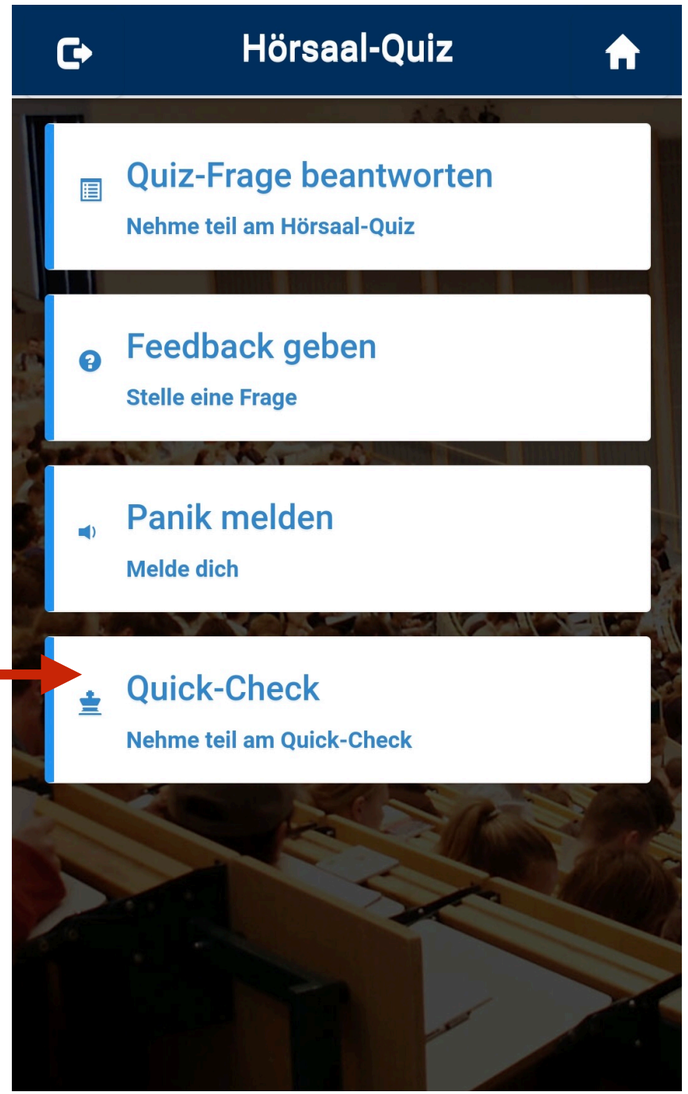
```
SELECT pers_nr, name,
TO_CHAR(gehalt,'L99,999') Gehalt
FROM mitarbeiter WHERE name = 'Genz';
```

PERS_NR	NAME	GEHALT
101014	Genz	€6,800.00

Quelle: Greenberg 2002, S. 159

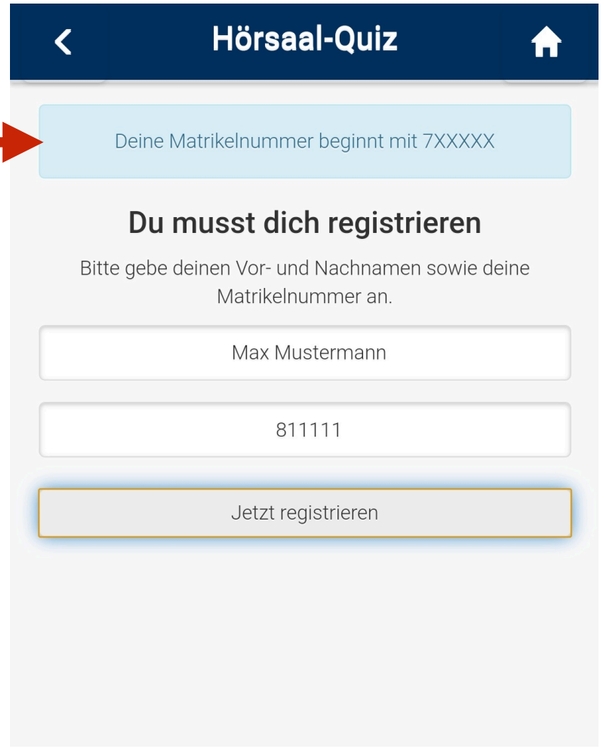
Wiederholung Quick-Checks

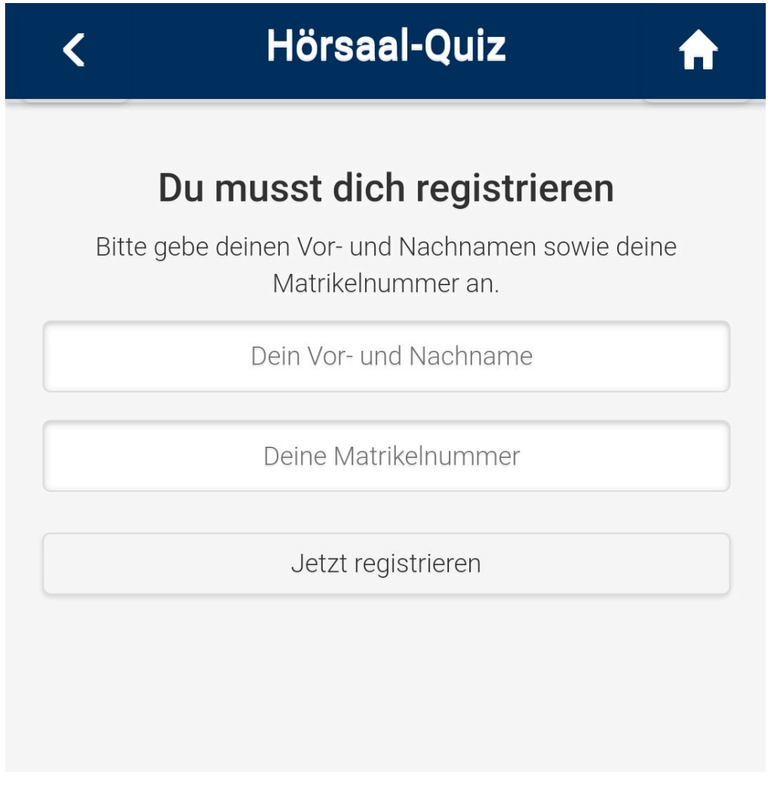
- Zur Teilnahme am Quick-Check rufen Sie bitte unsere Lehrstuhl-App unter quiz.lswi.de auf
- Veranstaltungskürzel: ewinf
- Quick Checks bringen Bonuspunkte, welche in der Klausur angerechnet werden können
- Insgesamt 9 Bonuspunkte durch Quickchecks zu erreichen

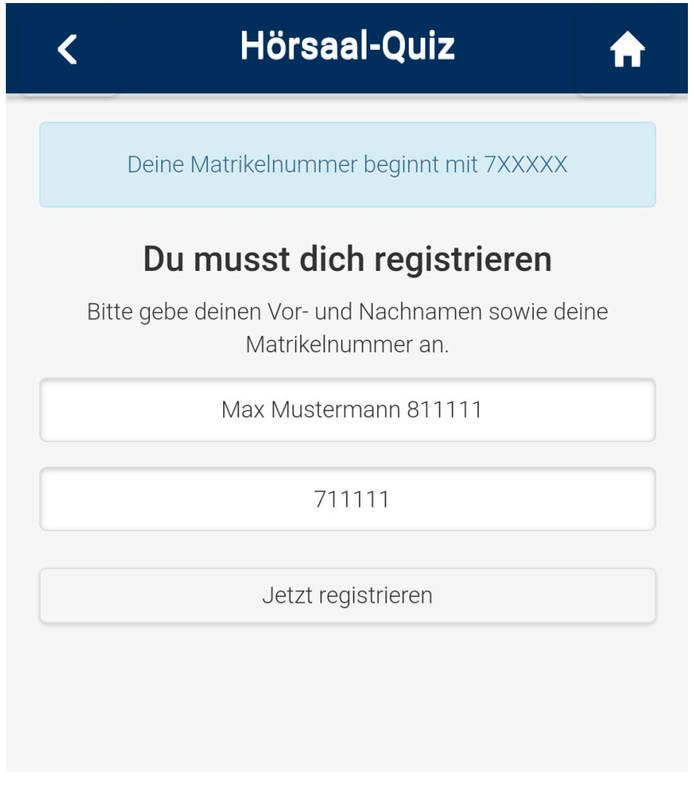


Wiederholung Quick-Checks

- Zur Anmeldung zum Quickcheck Name und Matr. Nummer angeben
- Achtung: Matrikelnummern mit einer 8 vorne werden momentan nicht akzeptiert
- Schreiben Sie zur Lösung ihre Matr. Nummer hinter Ihren Namen und ersetzen Sie die 8 vorne in ihrer Matr. Nummer durch eine 7 wie im Beispiel zu sehen







Lösung:

Kontrollfragen

- Welche Aufgabe erfüllt die WHERE-Klausel?
- Welche Bedeutung kommt der Einschränkung der Ausgabe redundanter Daten zu?
- Mit Hilfe welcher Operatoren kann eine Verknüpfung mehrerer Bedingungen erfolgen?
- Wie kann eine Liste nach einer bestimmten Spalte sortiert werden?
- Welches wesentliche Merkmal zeichnet Single Row-Funktionen aus?

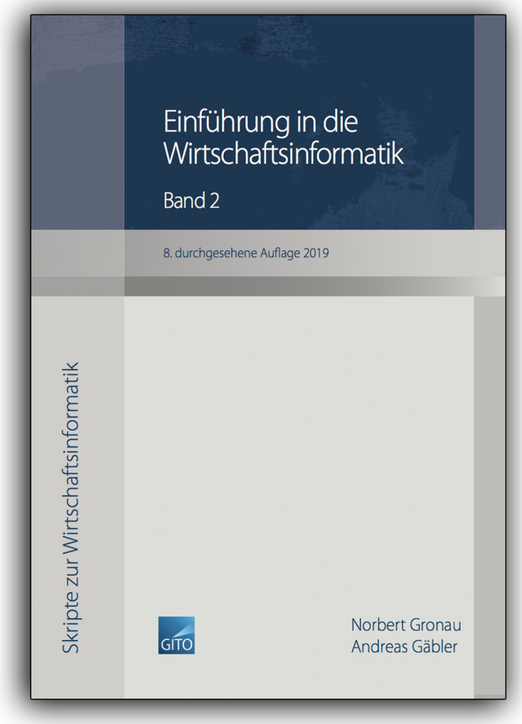
Literatur

Vossen, G.: Datenbankmodelle, Datenbanksprachen und Datenbankmanagementsysteme. - 4. Aufl. - Oldenbourg Verlag München 2000

Elmazri, R./Navathe, S. B.: Grundlagen von Datenbanksystemen; 3. Auflage, 2002, Addison-Wesley

Greenberg. N./Nathan, P.: Professioneller Einstieg in Oracle9i SQL - Band 1; 2002, Oracle

Zum Nachlesen



Kontakt

Univ.-Prof. Dr.-Ing. Norbert Gronau

Universität Potsdam
Karl-Marx-Str. 67 | 14482 Potsdam
Germany

Tel. +49 331 977 3322
E-Mail ngronau@lswi.de

Gronau, N., Gäbler, A.:
Einführung in die Wirtschaftsinformatik, Band 2
8. überarbeitete Auflage

GITO Verlag Berlin 2019. ISBN 978-3-95545-285-8