Einführung in die Wirtschaftsinformatik

Teil 9 – Erweiterte Funktionen

Wintersemester 2023/2024

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*

Mail August-Bebel-Str. 89 | 14482 Potsdam | Germany
Visitors Digitalvilla am Hedy-Lamarr-Platz, 14482 Potsdam
Tel +49 331 977 3322

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

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

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	Wirtschaftsingenieu
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.

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

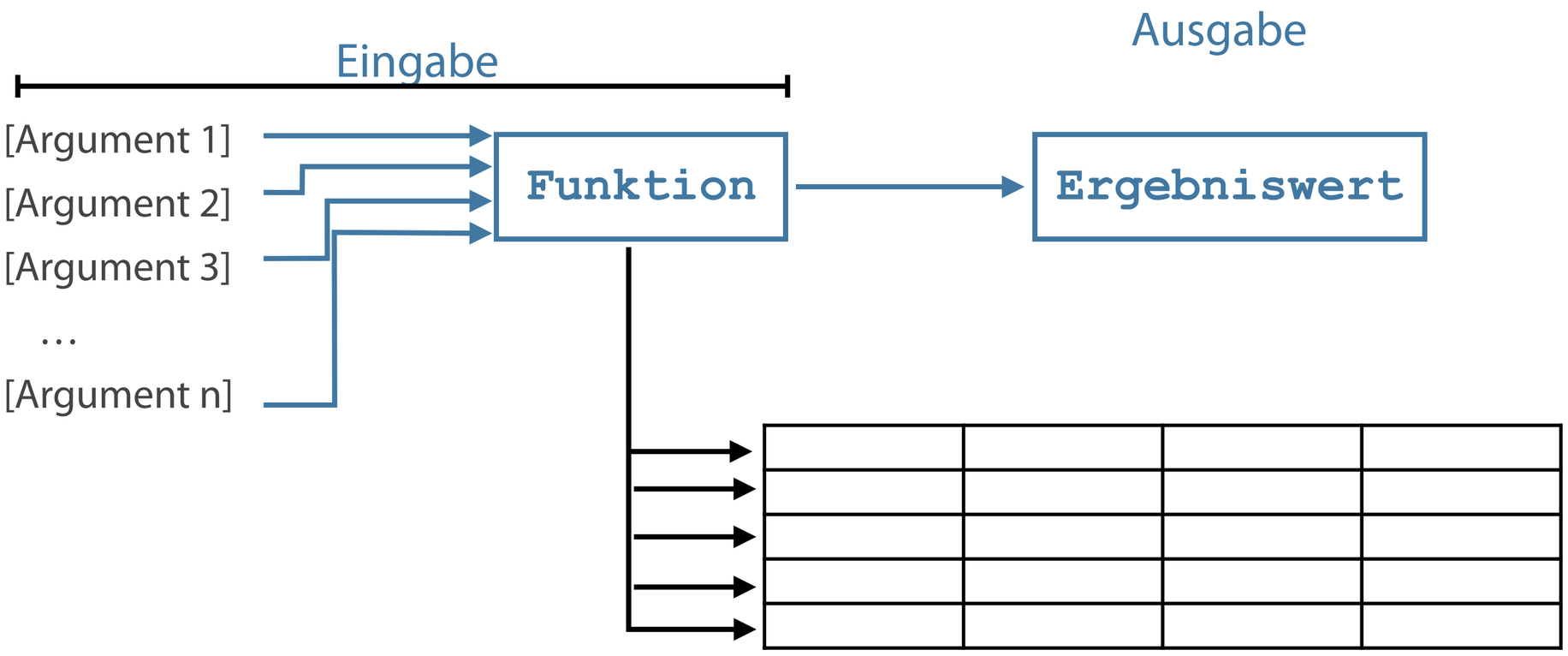
Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

SQL-Funktionen

- Bearbeitung von Zeilen und Ausgabe von Ergebnissen dieser Bearbeitung



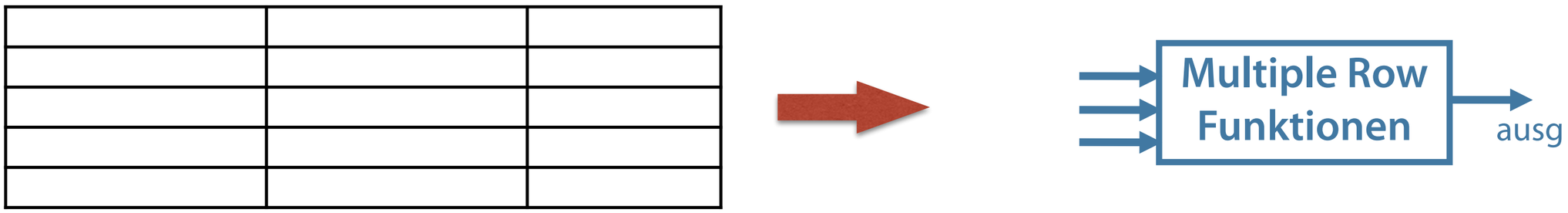
SQL-Funktionen enthalten manchmal Argumente und geben immer einen Wert zurück.

Quelle: Greenberg 2002, S. 125

Unterschied zwischen Single-Row und Multiple-Row-Funktionen

Single Row Funktion – Rückgabe – ein Ergebnis pro Zeile

Multiple Row Funktion – Rückgabe – ein Ergebnis pro Zeilengruppe



arg – Argument
ausg – Ausgabe

Multiple-Row-Funktionen werden bei verschachtelten SQL-Abfragen benötigt, um Funktionen auf eine Zeilengruppe anwenden zu können.

Quelle: Greenberg 2002, S. 126

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 I

- **CONCAT** - (concatenation – Verkettung) verknüpft Argumente (mit maximal zwei Parametern)
- **SUBSTR** - (substring - Teilkette) extrahiert eine Zeichenfolge bestimmter Länge. Extrahiert aus gegebener Zeichenfolge (Argument 1) eine Teilkette beginnend mit dem n-ten Buchstaben (Argument 2) mit der Länge von m Buchstaben (Argument 3)
- **LENGTH** - zeigt die Länge einer Zeichenfolge als numerischen Wert an
- **INSTR** - (in string) sucht die numerische Position eines angegebenen Zeichens innerhalb einer Zeichenkette

Funktion	Ausgabe
CONCAT('Betriebsteil', 'Potsdam')	BetriebsteilPotsdam
SUBSTR('BetriebsteilPotsdam',1,12)	Betriebsteil
LENGTH('BetriebsteilPotsdam')	19
INSTR('Betriebsteil', 's')	8

Quelle: Greenberg 2002, S. 133

Funktionen zum Bearbeiten von Zeichen II

- **LPAD** – setzt den in der Klammer aufgerufenen Ausdruck (Wert 1) rechtsbündig und füllt ihn mit der Zahl (Wert 2) der formulierten Zeichen (Wert 3) linksseitig (left padding) auf
- **RPAD** – setzt den in der Klammer aufgerufenen Ausdruck (Wert 1) linksbündig und füllt ihn mit der Zahl (Wert 2) der formulierten Zeichen (Wert 3) rechtsseitig (right padding) auf
- **TRIM** – (to trim – kürzen) schneidet Zeichen am Anfang oder am Ende einer Zeichenfolge ab (Wenn trim_character oder trim_source ein Zeichenliteral ist, muss es in Hochkommata gesetzt werden)

Funktion	Ausgabe
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

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

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:

Felder ohne Inhalte

Dieses Problem hatten wir in einer der vorigen VL!

```
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	
Metz	Chefsekretärin	…
…	…	-
Walker	Chief Information	-

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 +
Thyssen	1480	410V	0,13	Gehalt +
…	…	…	…	…

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

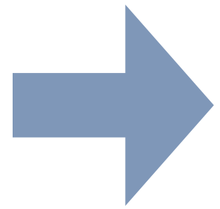
Wird zurückgegeben wenn ausdruck1 NULL ist

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

GEHALT	PROVISION
3600	
9000	0,065
2100	
1300	0,065



Jahreseinkommen
43.200
433.000
25.200
340.600

- Verschiedene Mitarbeiter beziehen ihr Jahresgehalt aus unterschiedlichen Quellen (reines Gehalt vs. Gehalt + Provision)
- Problem: Wir können keine einheitliche Spalte nutzen, um das Jahresgehalt auszurechnen - je nach Einkommensart müssen wir unterschiedliche Spalten nutzen

```
SELECT ... COALESCE(12*gehalt + 5000000*provision,
12*gehalt) "Jahreseinkommen" FROM mitarbeiter;
```

➡ Hinweis: Wir gehen davon aus, dass die Firma einen gesamten Provisionsumsatz von 5.000.000 hat, auf welchen auch die Provision berechnet wird

COALESCE() nimmt eine beliebig lange Liste von Werten an und gibt den ersten Wert ungleich NULL zurück.

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

Sortierung von Ergebnissen

Single Row-Funktionen

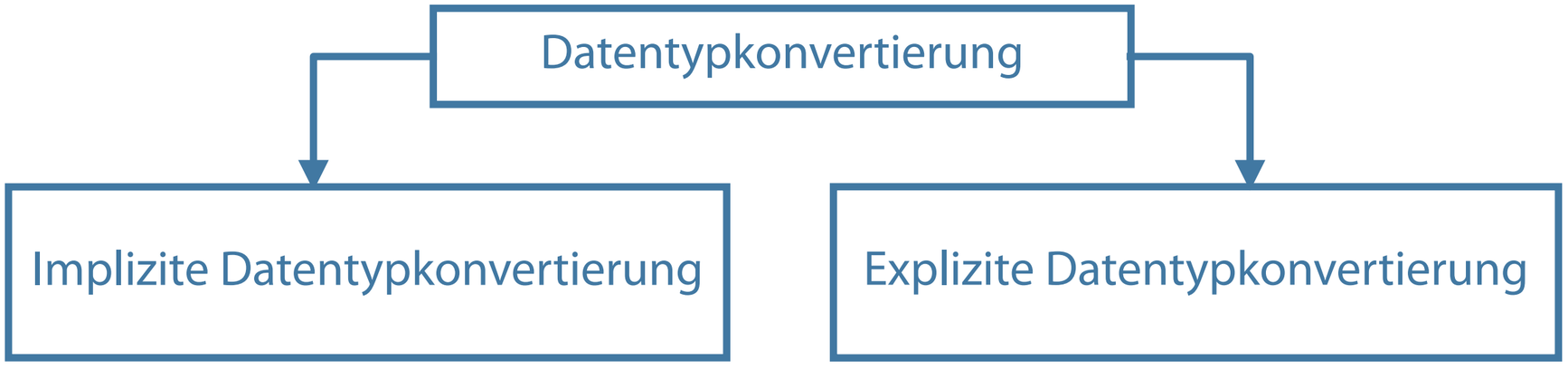
Behandlung von NULL-Werten

Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

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*1.19,'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

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

Gruppenfunktionen

Erzeugung aggregierter Informationen

- Aggregationsfunktionen bezogen auf Gruppen von Zeilen
- Rückgabe – NUR EIN Ergebnis pro Gruppe

```
SELECT MAX(gehalt)
FROM mitarbeiter;
```

PERS_NR	GEHALT
101001	10430
101002	2400
101003	3100
101004	3600
101005	6590
101006	7770
101007	8080
101008	3800
101009	4100
101010	4000
...	...
101064	3400
101065	56000
...	...
101135	2910
101136	2550

Suche das höchste Gehalt in der Tabelle "MITARBEITER"

MAX(GEHALT)
56000

Typen von Gruppenfunktionen

Gruppenfunktion	**Wirkung**
AVG([Filter] spalte)	Durchschnittswert
COUNT({*|[Filter] ausdruck})	Anzahl Zeilen
MAX([Filter] ausdruck)	Höchster Wert
MIN([Filter] ausdruck)	Kleinster Wert
SUM([Filter] spalte)	Summe
Ausdruck für Filter	
ALL	Berücksichtigung aller Werte (Standard)
DISTINCT	Keine Berücksichtigung doppelter Werte

Berechnung numerischer Werte

Gruppenfunktionen mit Rückgabe numerischer Werte

- Bildung von Werten jeweils aus einer Spalte

```
SELECT AVG(gehalt), MAX(gehalt), MIN(gehalt), SUM(gehalt)
FROM mitarbeiter
WHERE anrede = 'Frau';
```

AVG(GEHALT)	MAX(GEHALT)	MIN(GEHALT)	SUM(GEHALT)
3912,310810810810810810810	35000	500	289511

```
SELECT AVG(net_preis), MAX(net_preis), MIN(net_preis)
FROM artikel;
```

AVG(NET_PREIS)	MAX(NET_PREIS)	MIN(NET_PREIS)
27273,2	190000	73

Die Funktionen AVG und SUM sind nur bei numerischen Werten zulässig.

Gruppenfunktionen und Textwerte

Gruppenfunktionen auf Zeichenketten und Datumswerten

- MIN, MAX mit allen Datentypen möglich
- Textwerte nach alphabetischer Reihenfolge

```
SELECT MIN(name), MAX(name)
FROM mitarbeiter;
```

MIN(NAME)	MAX(NAME)
Abei	von Bingen

```
SELECT MAX(geburtstag) "Geburtstag jüngster Ma.", MIN(geburtstag) "Geburtstag ältester Ma."
FROM mitarbeiter;
```

Geburtstag jüngster Ma.	Geburtstag ältester Ma.
02.10.2001	04.07.1955

Zählfunktion

- COUNT listet Anzahl der Datensätze je Gruppe auf
- COUNT(*) – Anzahl aller Zeilen in einer Tabelle
- COUNT([Filter] ausdruck) – Anzahl der mit "ausdruck" übereinstimmenden Werte
  [Filter]: DISTINCT – Keine Berücksichtigung doppelter und NULL-Werte

```
SELECT COUNT(*)
FROM mitarbeiter
WHERE anrede = 'Frau';
```

COUNT(*)
74

```
SELECT COUNT(*) FROM bestellung
WHERE SUBSTR(bestelldatum,7,4) =
'2017';
```

COUNT(*)
29

COUNT-Abfragen einer nur NULL-Werte enthaltende Spalte erzeugen bei Filterung mittels DISTINCT den Wert "0".

Verwendung von DISTINCT in der Zählfunktion

Syntax und Wirkung – COUNT(DISTINCT ausdruck)

- Rückgabe der Zahl der eindeutigen, nicht leeren Werte für "ausdruck"
- Ausdruck – [numerischer Wert, Datumswert, Zeichen(kette)]
- Frage: Wie viele Abteilungen gibt es in der Firma?

```
SELECT COUNT(DISTINCT abt_nr) Abteilungen FROM mitarbeiter;
```

ABTEILUNGEN
35

- Frage: In wieviele Länder liefert das Unternehmen (außer Deutschland)?

```
SELECT COUNT(DISTINCT land) Auslandsmärkte FROM kunde
WHERE land <> 'Deutschland';
```

AUSLANDSMÄRKTE
14

Quelle: Greenberg 2002, S. 241

Gruppenfunktion und NULL-Wert

Auswahl aller Werte ohne NULL-Werte

- Berechnung durchschnittliches Einkommen in Abhängigkeit von der Provision

```
SELECT AVG(gehalt * (1 + provision))
Durchschnittseinkommen
FROM mitarbeiter;
```

DURCHSCHNITTSEINKOMMEN
1747,5962962962962

Auswahl aller Werte inklusive von NULL-Werten – Funktion NVL

- Ersetzen leerer Werte durch Vorgabewerte
- Funktion liefert für alle Zeilen ein brauchbares Ergebnis

```
SELECT AVG(gehalt * (1 + NVL(provision,0)))
Durchschnittseinkommen
FROM mitarbeiter;
```

DURCHSCHNITTSEINKOMMEN
4373,1105

Quelle: Greenberg 2002, S. 243

Gruppenfunktion GROUP BY

- Zulässige Datentypen – CHAR, VARCHAR2, NUMBER, DATE
- GROUP BY – Zusammenfassung der Werte innerhalb einer Spalte
- ORDER BY – Sortierung in der Spalte

```
SELECT [spalte,] [DISTINCT|ALL] gruppenfunktion(spalte), ...
FROM tabelle
  [WHERE bedingung]
  [GROUP BY spalte]
```

GROUP BY fasst Datensätze zu Gruppen zusammen.

Quelle: Greenberg 2002, S. 237

Bildung von Zeilengruppen – Dezidierte Selektion

Gruppierung der Ergebnisse mittels GROUP BY

- Selektion der Ergebnisse einer Gruppenfunktion nach verschiedenen Attributen
- Frage: Wie hoch sind die Personalausgaben in den einzelnen Abteilungen?

```
SELECT abt_nr, SUM(gehalt) FROM mitarbeiter
GROUP BY abt_nr;
```

ABT_NR	SUM (GEHALT)
-	235180
260M	11940
310V	16470
210V	20024
210E	17660
…	…

Bei der Gruppierung mittels GROUP BY bilden die NULL-Werte standardmäßig eine eigene Gruppe.

Quelle: Greenberg 2002, S. 245

Gruppierung nach mehreren Spalten

Suche in der Tabelle "MITARBEITER" das Durchschnittsgehalt der einzelnen Berufe innerhalb jeder Abteilung

```
SELECT position, abt_nr, AVG(gehalt) "Mittl. Gehalt" FROM mitarbeiter
GROUP BY abt_nr, position;
```

POSITION	ABT_NR	GEHALT
Vertriebsgruppenleiterin	310V	4490
Vertriebsbeauftragter	110V	1310
Vertriebsbeauftragter	210V	1380
Vertriebsgruppenleiterin	210V	4515
Vertriebsbeauftragter	110V	1310
Geschäftsleiter	10VL	7330
...	...	...
Qualitätsmanager	106Q	4080
...	...	...
Sachbearbeiterin	510L	2700
Sachbearbeiter	510L	2400
Sachbearbeiterin	520G	2500
Sachbearbeiterin	520G	2630
Sachbearbeiter	530A	2400
...	...	...

***Beispiel**:*
Abteilung 520G
2 Sachbearbeiterinnen
Durchschnittsgehalt = 2565 Euro

POSITION	ABT_NR	Mittl. Gehalt
...	...	...
Leiter interne Revision	107R	5180
Sekretärin	330E	2500
Sekretärin	340R	2500
Entwicklungsingenieu	340R	4580
Elektriker	630E	3100
Vertriebsbeauftragter	110V	1352
...	...	...
Sachbearbeiterin	520G	2565
...	...	...

Quelle: Greenberg 2002, S. 245

Kombination von GROUP BY und ORDER BY- Klausel

Gruppierung über mehrere Spalten

- Abfrage des Durchschnitts aller Gehälter innerhalb jeder einzelnen Abteilung bezogen auf die Berufe

```
SELECT abt_nr, position, AVG(gehalt) Durchschnittsgehalt
FROM mitarbeiter
GROUP BY abt_nr, position
ORDER BY abt_nr;
```

ABT_NR	POSITION	DURCHSCHNITTSGEHALT
100V	Assistent der Konzernleitung	3780
100V	Assistentin der Konzernleitung	3400
100V	Investor Relations	3630
100V	Leiterin Vorstandsstab	6800
100V	Public Relations	3610
105C	Abteilungsleiter	8900
105C	Controller	3400
105C	Sekretärin	2450
...	...	...

Verschachteln von Gruppenfunktionen

```
SELECT MAX(AVG(gehalt))"Maximales Durchschnittsgehalt"
FROM mitarbeiter
GROUP BY abt_nr;
```

Maximales Durchschnittsgehalt
19598,33333…..

```
SELECT MAX(AVG(gehalt)) "Maximales Durchschnittsgehalt"
FROM mitarbeiter
WHERE abt_nr IS NOT NULL
GROUP BY abt_nr;
```

Maximales Durchschnittsgehalt
4900

Eine Auswertung verschachtelter Funktionen erfolgt immer von innen nach außen.

Quelle: Greenberg 2002, S. 257

Sortierung von Ergebnissen

Single Row-Funktionen

Behandlung von NULL-Werten

Konvertierungsfunktionen

Aggregierung von Daten

Filterung von Gruppenergebnissen

Syntax der HAVING-Klausel

Formulierung von Bedingungen für Gruppen – Abarbeitungsreihenfolge

1. Prüfung der WHERE-Bedingung für jeden einzelnen Datensatz
2. Gefilterte Datensätze werden gruppiert
3. Anzeige der Gruppendatensätze entsprechend der HAVING-Klausel

```
SELECT spalte, gruppenfunktion
FROM tabelle
[WHERE bedingung]
[GROUP BY group_by_ausdruck]
[HAVING gruppenbedingung]
[ORDER BY spalte];
```

Mögliche Funktionen innerhalb der Bedingung:
MIN, MAX, SUM, COUNT, AVG

HAVING wirkt ausschließlich bei Einschränkungen nach Gruppenfunktionen.

Quelle: Elmasri 2002, S. 302

Einschränkungen mit Hilfe der HAVING-Klausel

Funktion mit numerischem Argument

- Abfrage des Durchschnitts aller Gehälter innerhalb jeder einzelnen Abteilung, das zwischen 4000 und 5000 Euro liegt.

```
SELECT abt_nr, AVG(gehalt) Durchschnittsgehalt
FROM mitarbeiter
GROUP BY abt_nr
HAVING AVG(gehalt) BETWEEN 4000 AND 5000;
```

ABT_NR	DURCHSCHNITTSGEHALT
420F	4900
620W	4296
250A	4262,5
320M	4473,333333333333333333333
310T	4607,5
610A	4322,5
...	...

Aggregation und Filterung

GEHALT	ABT_NR
4490	310V
1310	110V
1380	210V
4515	210V
1310	110V
7330	10VL
4810	410V
1580	110V
1400	210V
2104	10VL
...	...
6300	630E
3400	100V
3100	240P
4400	240P
2740	260F
...	...

1 Suche in der Tabelle "MITARBEITER" das minimale Gehalt innerhalb jeder Abteilung.

1 Aggregation (GROUP BY)

GEHALT	ABT_NR
1310	310V
1360	210V
3000	-
2400	260M
1310	110V
2570	210E
2500	520G
1380	410V
2060	260D
...	...

2 Gib nur die aus, deren Wert unter 2800 Euro liegt.

Ergebnisse durch Selektion gruppierter Werte ausgeblendet

3 Filterung (HAVING)

GEHALT	ABT_NR
1310	310V
1360	210V
2400	260M
1310	110V
2570	210E
2500	520G
...	...

Die HAVING-Klausel filtert die anzuzeigenden Tabellenzeilen in der Gruppierung.

Kontrollfragen

- Wie kann die Datenausgabe eingeschränkt werden?
- Welche Aufgabe erfüllt die WHERE-Klausel?
- 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?
- Welche Aufgabe erfüllen Konvertierungsfunktionen?
- Was sind Aggregatfunktionen?
- Können Gruppenfunktionen auf beliebige Datentypen angewandt werden?
- Wie lässt sich eine Gruppierung nach mehreren Spalten realisieren?

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