Gruppieren mit GROUP BY

Mit dem SQL Befehl GROUP BY können Daten gruppiert werden.

Interessant sind Gruppierungen vor allem in Kombination mit Aggregatfunktionen, wie z.B. COUNT, SUM, MIN/MAX usw.

Syntax:

```
SELECT spaltenname, [...]
FROM tabellenname
[WHERE Bedingung]
GROUP BY spalte
[HAVING Bedingung]
[ORDER BY Spalte, [...] [ASC | DESC]];
```

Am Besten sieht man die Funktion von GROUP BY an Beispielen:

Beispiel 1:

Möchte man wissen in wie vielen unterschiedliche Abteilungen tatsächlich Mitarbeiter arbeiten, könnte man das z.B. so machen:

```
SELECT abteilung_id
FROM mitarbeiter
ORDER BY abteilung_id;

ABTEILUNG_ID
------------
           1
           1
           2
           2
           3
       .....
       .....
       .....
           9
          10
          10
          11
          11
          12
          12

28 rows selected.
```

So sehen wir, dass die abteilung_id zT mehrfach vorkommt. Dies könnte mit DISTINCT eliminiert werden, man kann aber auch nach abteilung_id gruppieren:

```
SELECT abteilung_id
FROM mitarbeiter
GROUP BY abteilung_id
ORDER BY abteilung_id;

ABTEILUNG_ID
------------
           1
           2
           3
       ...
       ...
          11
          12

12 rows selected.
```

Schön wäre es jetzt, wenn man sehen könnte, wie viel Mitarbeiter in jeder Abteilung arbeiten.

Wir fügen die Aggregatsfunktion „count“ hinzu:

```
SELECT abteilung_id, count(*) anzahl
FROM mitarbeiter
GROUP BY abteilung_id
ORDER BY abteilung_id;

ABTEILUNG_ID    ANZAHL
------------ ----------
           1          2
           2          2
           3          2
           4          2
           5          4
           6          2
           7          2
           8          4
           9          2
          10          2
          11          2
          12          2

12 rows selected.
```

Das ist die eigentliche Funktion von GROUP BY. In Kombination mit Aggregatsfunktionen.

Beispiel 2:

Es soll ausgegeben werden, wieviel Objekte je Objekttyp in der Datenbank sind:

```
SELECT object_type, count(*)
FROM dba_objects
GROUP BY object_type
ORDER by 1;

OBJECT_TYPE          COUNT(*)
-------------------- ----------
CLUSTER                      10
CONSUMER GROUP               25
CONTEXT                       3
DESTINATION                   2
DIRECTORY                     3
EDITION                       1
EVALUATION CONTEXT            9
FUNCTION                     85
INDEX                      1242
INDEX PARTITION             126
JOB                          11
JOB CLASS                    13
                           ...
                           ...
                           ...
RULE                          1
RULE SET                     13
SCHEDULE                      3
SCHEDULER GROUP               4
SEQUENCE                    150
SYNONYM                    3261
TABLE                      1150
TABLE PARTITION             110
TABLE SUBPARTITION           32
TRIGGER                      18
TYPE                       1296
TYPE BODY                   104
UNDEFINED                    11
VIEW                       3718
WINDOW                        9

37 rows selected.
```

Beispiel 3:

Es kann auch nach mehreren Attributen gruppiert werden:

```
SELECT object_type, owner, count(*)
FROM dba_objects
GROUP BY object_type, owner
ORDER by 1,2;

OBJECT_TYPE             OWNER                COUNT(*)
------------------- ---------------------- ----------
CLUSTER                 SYS                        10
CONSUMER GROUP          SYS                        25
CONTEXT                 SYS                         3
                     ...
INDEX                    DBSNMP                    10
INDEX                    HR                        19
INDEX                    OUTLN                      4
INDEX                    SYS                      987
INDEX                    SYSTEM                   211
INDEX                    VERSICHERUNG              11
                     ...
SEQUENCE                 DBSNMP                     2
SEQUENCE                 HR                         3
SEQUENCE                 SYS                      114
SEQUENCE                 SYSTEM                    20
SEQUENCE                 VERSICHERUNG              11
SYNONYM                  APPQOSSYS                  1
SYNONYM                  DBSNMP                     1
SYNONYM                  PUBLIC                  3245
SYNONYM                  SYS                        6
SYNONYM                  SYSTEM                     8
TABLE                     APPQOSSYS                 4
TABLE                     DBSNMP                   20
TABLE                     HALLER                    1
TABLE                     HR                        7
TABLE                     OUTLN                     3
TABLE                     SO                        2
TABLE                     SYS                     948
TABLE                     SYSTEM                  154
TABLE                     VERSICHERUNG             11
                     ...
VIEW                     DBSNMP                     7
VIEW                     SO                         1
VIEW                     SYS                     3698
VIEW                     SYSTEM                    12
WINDOW                   SYS                        9

82 rows selected.
```

HAVING:

HAVING ist quasi die „WHERE Klausel von GROUP BY“.

Mit HAVING kann die Ausgabe eingeschränkt werden. HAVING bezieht sich auf die Aggregatsfunktion der GROUP BY Klausel.

Beispiel:

Es sollen alle Abteilungen angezeigt werden, bei welchen das totale Gehalt höher als 50‘000 ist:

```
-- Ohne Einschränkung:
SELECT department_id, sum(salary)
FROM employees
GROUP BY department_id
ORDER BY department_id;

DEPARTMENT_ID SUM(SALARY)
------------- -----------
           10        4400
           20       19000
           30       24900
           40        6500
           50      156400
           60       28800
           70       10000
           80      304500
           90       58000
          100       51600
          110       20300
                     7000

12 rows selected.

-- Eingeschränkt auf Löhne über 50‘000
SELECT department_id, sum(salary)
FROM employees
GROUP BY department_id
HAVING sum(salary) > 50000
ORDER BY department_id;

DEPARTMENT_ID SUM(SALARY)
------------- -----------
           50      156400
           80      304500
           90       58000
          100       51600

4 rows selected.
```

Verbinden mit UNION

Der SQL Befehl UNION vereinigt die Ergebnisse mehrerer Abfragen.
Bei UNION müssen alle selektierten Spalten vom gleichen Datentyp sein.
Mit UNION werden nur unterschiedliche Werte ausgegeben. Mehrfach vorkommende Rows werden, ähnlich wie bei SELECT DISTINCT, entfernt.

Syntax:

```
Select Statement 1
UNION [ALL]
Select Statement 2
UNION [ALL]
Select Statement 3;
```

bzw.

```
SELECT spaltenname, [...] FROM tabellenname
UNION [ALL]
SELECT spaltenname, [...] FROM tabellenname;
```

Der Unterschied zwischen UNION und UNION ALL ist, dass UNION Duplikate aller verbundenen Statements entfernt (wie SELECT DISTINCT) und UNION ALL keine Duplikate entfernt.

Aus diesem Grund ist UNION ALL auch schneller. Wo immer möglich sollte UNION ALL verwendet werden und nicht UNION. Das spielt bei Tabellen mit einigen 1000 Rows nicht so eine grosse Rolle, haben die Tabellen aber mehrere Millionen Einträge, macht das durchaus einen Unterschied.

Beispiel:

Es soll eine Liste aller Mitarbeiter der Schemen „VERSICHERUNG“ und „HR“ erstellt werden:

```
SELECT vorname, name FROM versicherung.mitarbeiter
UNION
SELECT first_name, last_name FROM hr.employees
ORDER BY 1,2;

VORNAME                   NAME
------------------------- ------------------------------
Adam                      Fripp
Alana                     Walsh
Alberto                   Errazuriz
Alexander                 Hunold
Alexander                 Khoo
Alexis                    Bull
Allan                     McEwen
Alyssa                    Hutton
Amit                      Banda
Angelina                  Friedrichsen
...
Christian                 Braun
Christina                 Schindler
Christopher               Olsen
Clara                     Vishney
Curtis                    Davies
Daniel                    Faviet
Daniela                   Schneider
Danielle                  Greene
David                     Austin
David                     Bernstein
David                     Lee
...
Vance                     Jones
Walter                    Meyer
Werner                    Feyerabend
William                   Gietz
William                   Smith
Winston                   Taylor
Yvonne                    Baber
Zacharias                 Carlsen
Zafer                     Aliman

135 rows selected.
```