PostgreSQL – Aggregierte Abfragen 1

Stephan Karrer

Wichtige Gruppenfunktionen

Funktion	Beschreibung
AVG	Durchschnittswert, ignoriert NULL Werte
COUNT	Anzahl der ausgewählten Zeilen
MAX	Gibt den höchsten Wert einer Auswahl wieder
MIN	Gibt den niedrigsten Wert einer Auswahl wieder
STDDEV	Standardabweichung einer Gruppe von Werten, wobei NULL – Werte ignoriert werden
SUM	Summenwerte einer Gruppe von Werten, wobei NULL – Werte ignoriert werden
VARIANCE	Abweichung einer Gruppe von Werten, wobei NULL – Werte ignoriert werden

- Es gibt selbstverständlich viele weitere, siehe https://www.postgresql.org/docs/18/functions-aggregate.html

Einfache Verwendung von Gruppenfunktionen

```
SELECT COUNT(*) "Anzahl Zeilen" FROM employees;

SELECT AVG( COALESCE(salary, 0)) FROM employees;

SELECT MAX(salary) FROM employees
       WHERE job_id = 'IT_PROG' ;

SELECT count(DISTINCT department_id) FROM employees;
```

- Die Menge der zu aggregierenden Werte, wird durch den Query mit der Filterung durch die WHERE-Klausel definiert.
- Es kann auch DISTINCT innerhalb der Aggregation verwendet werden.
- Sofern die Menge der Werte leer ist, liefern alle (außer COUNT) einen NULL-Value zurück !

Spezielle Möglichkeiten

```
SELECT count(DISTINCT department_id)
       FILTER (WHERE department_id > 40) FROM employees;

SELECT string_agg(last_name, ', ') FROM employees;

SELECT string_agg(last_name, ', ' ORDER BY last_name)
FROM employees;
```

- Die Filterung der zu aggregierenden Werte kann auch mittels der FILTER-Klausel erfolgen.
- Bei einigen Aggregatsfunktionen, insbesondere bei Zeichenketten-Funktionen, kann die Sortierung wichtig sein
  (dies gilt auch für die entsprechenden JSON- und XML-Funktionen)

Verwendung von Gruppenfunktionen

```
SELECT [ALL|DISTINCT] Auswahlliste
        FROM Quelle
        [WHERE Where-Klausel]
        [GROUP BY (Group-by-Attribut)
                [HAVING Having-Klausel]]
        [ORDER BY (Sortierungsattribut)[ASC|DESC]]
```

- WHERE:
  schränkt die Ausgangsmenge ein.
- GROUP BY:
  zerlegt die Ausgangsmenge in Gruppen. Je Gruppe wird aggregiert.
- HAVING:
  nachträgliche Filterung der Ergebnisse.
- SELECT:
  in der Auswahlliste können nur die Gruppierungsattribute und Aggregate verwendet werden !!

Verwendung von Gruppenfunktionen: GROUP BY und HAVING

```
SELECT department_id, COUNT(DISTINCT job_id)
       FROM employees
       GROUP BY department_id;

SELECT job_id, SUM(salary)
       FROM employees
       WHERE department_id <> 100
       GROUP BY job_id
       ORDER BY job_id DESC;

SELECT job_id, SUM(salary)
       FROM employees
       WHERE department_id <> 100
       GROUP BY job_id
       HAVING SUM(salary) > 20000
       ORDER BY job_id DESC;
```

Nach mehreren Spalten gruppieren

```
SELECT department_id  dept_id,
       job_id,
       SUM(salary)
       FROM employees
       GROUP BY department_id, job_id;
```

- Es wird anhand der tiefsten Gruppenbildung aggregiert.

Gruppierung nach berechneten Werten

```
SELECT length(last_name) as "Length", count(*) as "Count"
  FROM employees
  GROUP BY LENGTH(last_name)
  ORDER BY "Length";
```

- Generell kann nach Ausdrücken und damit nach berechneten Werten aggregiert werden.
- Der Ausdruck darf aber keine Aggregats- oder Analytische Funktion enthalten (keine Schachtelung von Aggregatsfunktionen).