SQL-Statements in der Praxis – weniger ist mehr

Ernst Leber, MT AG

Das ist wohl jedem schon einmal passiert: Man schaut mal eben in der Datenbank nach. „Select * from …“ ist schnell eingegeben und fertig ist das Ergebnis! „Ergebnis“ heißt in diesem Fall, die Daten rauschen über den Bildschirm. Aber was wollte man eigentlich wissen? Streng genommen weiß man doch nur, dass in der Tabelle Daten vorhanden sind. Erst mit der zweiten oder dritten Variante des SQL-Statements kommt man dem eigentlichen Ziel näher. Der Artikel zeigt, wie sich mit einigen relativ einfachen Select-Statements konkrete Aussagen zu den Daten in der Datenbank machen lassen.

Ob man als DBA nach der User-Verteilung sucht oder als Applikations-Entwickler nach der Datenverteilung in seinen Tabellen: Durch die gezielte Nutzung von „GROUP BY“ und Aggregat-Funktionen wie „sum()“ oder „count()“ lässt sich mit einfachen Statements die gewünschte Information schnell aus dem Datenbestand ermitteln. Je mehr man diese Statements nutzt, desto schneller wird man feststellen, dass diese SQL-Statements einem bestimmten Schema folgen.

Basierend auf diesen Aggregat-Funktionen lassen sich Grundgerüste für SQL-Abfragen entwickeln, die leicht zu merken und einfach anzupassen sind. Zur Verdeutlichung ein Beispiel, basierend auf einer der Aussagen, die DBAs sehr oft hören: „Die Datenbank ist aber langsam heute.“ Schauen wir doch mal nach, was in der Datenbank los ist. „select * from v$session” könnte doch jetzt schnell eingegeben werden, aber genau das sollte ja vermieden werden. Wie wäre es mit diesem Ansatz in *Listing 1*?

Die Ausgabe dieses Statements kommt aus einer Spiel-Datenbank, in der nichts los ist. Dasselbe Statement in einer anderen Datenbank ergibt ein anderes Ergebnis *(siehe Listing 2)*.

Das Beispiel zeigt, dass in der Datenbank die Applikations-User „APPL1“ und „APPL2“ mit etlichen Sessions und „APPL2“ mit zehn Active Sessions vertreten sind. Hier wäre ein Ansatz, zu prüfen, ob einer dieser User oder eventuell die Summe der User das Problem ausmachen. Eine andere Variante, um schnell zu prüfen, was in der Datenbank los ist, ist die Nutzung von ASH v$active_session_history *(siehe Listing 3)*.

Hier ist zu sehen, dass gerade ein Backup auf einer Datenbank läuft, deren Datenfiles im ASM liegen. Dies muss nicht die Ursache für die Probleme sein, die der Anwender mit der Meldung „Die Datenbank ist aber langsam heute!“ meinte. Der wesentliche Punkt ist, dass man durch einfaches Zählen und Gruppieren der Events Rückschlüsse darauf ziehen kann, was in der Datenbank gerade passiert. Diese Information haben wir mit einfachen Mitteln erhalten.

Die beiden Statements aus *Listing 1 und 3* folgen jedoch einem bestimmten Schema. In *Listing 4* ist das Prinzip dieses Grundgerüsts übersichtlich dargestellt. Aufbauend auf diesem Schema sind sehr schnell einige grundlegende SQL-Statements erstellt.

```
select inst_id, username, status, count(*)
     from gv$session
  group by inst_id, username,status
order by 1,2
;

INST_ID  USERNAME  STATUS    COUNT(*)
-------  --------  --------  -------
  1         SYS    ACTIVE        1
  1                ACTIVE       35
```

Listing 1: Sessions zählen

```
INST_ID  USERNAME  STATUS    COUNT(*)
-------  --------  --------  -------
  1      DBSNMP    INACTIVE      1
  1      APPL1     ACTIVE       40
  1      APPL1     INACTIVE      2
  1      APPL2     INACTIVE     140
  1      APPL2     ACTIVE       10
  1      SYS       ACTIVE        1
  1                ACTIVE       35
```

Listing 2: Sessions in einer aktiven Datenbank

```
select decode(session_state,'ON CPU','ON CPU', event), count(*)
  from v$active_session_history
  group by decode(session_state,'ON CPU','ON CPU', event)
  order by 1;

EVENT                                  COUNT(*)
---------------------------------------   --------
ASM file metadata operation                 10
Backup: MML commit backup piece             41
Backup: MML create a backup piece            1
Backup: MML initialization                   1
Backup: MML query backup piece               1
Backup: MML write backup piece           21632
ON CPU                                    4448
SQL>
```

Listing 3: Events zählen

```
select <spalten liste> <Aggregat Funktion>
  from <tabelle>
 group by <spalten liste>
 order by .....
```

Listing 4: Select-Grundgerüst

```
select owner,tablespace_name, segment_type,
      round(sum(Bytes) / power(1024,2)) MB
  from dba_segments
 group by owner,tablespace_name, segment_type
 order by 1, 2, 3;
```

Listing 5: Objektverteilung nach User, Typ und Größe

```
select tablespace_name,
      round(sum(Bytes) / power(1024,2)) MB
  from dba_segments
 group by tablespace_name
 order by 1;
```

Listing 6: Leere Tablespaces

```
select owner, ´Table` Type,
      trunc(last_analyzed) last_analyzed,
      STALE_STATS stale, count(*) cnt
  from dba_tab_statistics
 group by owner, trunc(last_analyzed), STALE_STATS
union
select owner, ´Index` Type,
      trunc(last_analyzed) last_analyzed,
      STALE_STATS stale, count(*) cnt
  from dba_ind_statistics
 group by owner, trunc(last_analyzed), STALE_STATS
 order by 1, 3, 5;
```

Listing 7: Statistiken anzeigen

Dieses Grundgerüst ist natürlich der SQL-Syntax geschuldet, dem Autor ist aber immer wieder aufgefallen, dass gerade das mächtige „GROUP BY“ mit Aggregat-Funktionen bei Standard-Abfragen gar nicht oder höchst selten genutzt wird. Auf Nachfrage bekam er zu hören: „Das ist zu kompliziert, die Syntax kann ich mir nicht merken.“ Nun, basierend auf dem Grundgerüst einige Beispiele dafür, wie mithilfe des „select ... from ... group by ...“ Daten analysiert und überschaubar angezeigt werden können. *Listing 5* zeigt, wie sich die Objekte in der Datenbank auf die Tablespaces und User verteilen und was für Objekt-Typen das sind. Gibt es Tablespaces in der Datenbank, die leer sind *(siehe Listing 6)*, beziehungsweise wann wurden die Statistiken in der Datenbank erstellt *(siehe Listing 7)*?

Fazit

Falls man es nicht schon nutzt, sollte man die Kombination aus Aggregat-Funktionen und „GROUP BY“ in SQL-Statements ausprobieren. Man wird erstaunt sein, welche Möglichkeiten sich mit diesem simplen SQL-Konstrukt bieten und wie schnell man Antworten auf seine Fragen erhält.

Ernst Leber
ernst.leber@mt-ag.com
https://eleoracle.wordpress.com