

## Tipps & Tricks: Januar 2005

Bereich:	PL/SQL	Erstellung:	01/2005 HA
Versionsinfo:	10.2, 11.1	Letzte Überarbeitung:	06/2010 HA

 [Als PDF Downloaden!](#)

### Neuer Job Scheduler in 10g

Mit Version 10g wurde ein neuer Job Scheduler mit deutlich erweiterten Möglichkeiten eingeführt. Die hervorstechendste Neuerung: es können externe Programme auf Betriebssystemebene aufgerufen werden.

Folgende Komponenten gehören unter Windows zum neuen Scheduler:

- das Package DBMS\_SCHEDULER.
- ein Hintergrundprozess, der sich um die Abarbeitung der Jobs kümmert (CJQ).
- ein Windows-Dienst (OracleJobScheduler<SID>), der laufen muss, wenn Jobs Programme auf BS-Ebene aufrufen; <SID> steht dabei für die SID der Datenbank
- eine Reihe neuer DD-Views

Um einen Job anlegen zu können, braucht ein User das CREATE JOB-Recht. Für die Ausführung von Betriebssystem-Jobs ist zusätzlich das Recht CREATE EXTERNAL JOB notwendig. Der Parameter JOB\_QUEUE\_PROCESSES, der in Oracle 9i auf Werte > 0 gesetzt werden musste, um die Verarbeitung von Jobs zu ermöglichen, ist unerheblich für den neuen Scheduler.

### Anlegen eines Jobs

Für jeden Job muss angegeben werden:

- Name des Jobs
- durchzuführende Aktion
- Typ

In Oracle Version 10.1 gibt es drei mögliche Typen:

- `plsqli_block`: als Aktion ist hier ein Anonymer Block möglich oder auch der Aufruf einer Stored Procedure (mit Semikolon am Ende !!!); diese Option entspricht in etwa der Funktionalität von DBMS\_JOB.
- `stored_procedure`: hier wird eine Stored Procedure aufgerufen (ohne Semikolon am Ende !!!);
- `executable`: als Aktion kann hier (prinzipiell) alles angegeben werden, was von der Kommandozeile des BS aus ausgeführt werden kann

Die Angabe des Intervalls kann auf zwei Arten erfolgen: entweder - wie bei [DBMS\\_JOB](#) - als PL/SQL-Ausdruck oder als kalendarischer Ausdruck ("calendar expression"). Im ersten unten angeführten Beispiel wurde die kalendarische Form gewählt.

Wird weder Startdatum noch Intervall angegeben, so wird der Job einmalig ausgeführt, sobald er aktiviert ist und dann gelöscht.

Wird nur ein Startdatum angegeben, so wird der Job einmalig zu diesem Zeitpunkt ausgeführt. Bei Verwendung eines kalendarischen Ausdrucks ist das Startdatum NICHT das Datum der erstmaligen Ausführung, sondern Referenzpunkt für dessen Bestimmung.

Beispiele:

Im ersten Beispiel soll die Prozedur testproc täglich um 14 Uhr ausgeführt werden:

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
    job_name => 'job1',
    job_type => 'stored_procedure',
    job_action => 'testproc',
    start_date => TRUNC(SYSDATE),
    repeat_interval => 'freq=DAILY;byhour=14',
    enabled => TRUE);
END;
```

Im zweiten Beispiel soll das Skript test.sql um 20 Uhr des gleichen Tages einmalig ausgeführt werden:

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
    job_name => 'job2',
    job_type => 'EXECUTABLE',
    job_action => 'cmd /c sqlplus scott/tiger@c:\test.sql',
    start_date => TRUNC(SYSDATE) + 20/24,
    enabled => TRUE);
END;
```

Ohne Angabe des enabled-Parameters ist ein Job standardmäßig deaktiviert. Erst nach Aktivierung (entweder durch Setzen dieses Parameters auf TRUE oder durch Aufruf der ENABLE-Prozedur) wird ein Job ausgeführt.

Ein kalendarischer Ausdruck MUSS als erste Angabe "FREQ=<interval>" enthalten, gültige Werte für <interval> sind: YEARLY, MONTHLY, WEEKLY, DAILY, HOURLY, MINUTELY oder SECONDLY.

Zusätzlich KANN er noch weitere Klauseln enthalten, die das Intervall vergrößern und/oder den genauen Zeitpunkt noch weiter eingrenzen können.

Ein Job kann mit der Prozedur DBMS\_SCHEDULER.DROP\_JOB wieder gelöscht werden.

## Programs und Schedules

Neben Jobs können auch "programs" (Programmeinheiten) und "schedules" (Ablaufpläne) erstellt werden, die dann in CREATE\_JOB anstelle von job\_action und job\_type bzw. repeat\_interval angegeben werden können.

Der Vorteil dieser Option liegt in der Wiederverwendbarkeit: unterschiedliche Jobs können das gleiche program oder den gleichen Schedule referenzieren.

Bei Schedules sind nur kalendarische Ausdrücke zur Angabe des Intervalls zulässig.

Will man auf programs aus einem fremden Schema heraus zugreifen, so braucht man dafür das entsprechende EXECUTE-Recht. Schedules dagegen sind frei zugänglich.

Jobs, programs und schedules sind in 10g eigenständige Datenbank-Objekte, die auch in der view DBA\_OBJECTS aufgeführt werden. Ihr Name muss dementsprechend innerhalb eines Schemas eindeutig sein.

Beispiele:

```
BEGIN
  DBMS_SCHEDULER.CREATE_PROGRAM(
```

```
        program_name => 'program1',  
        program_type => 'stored_procedure',  
        program_action => 'testproc',  
        enabled => TRUE);
```

```
END;
```

```
BEGIN
```

```
    DBMS_SCHEDULER.CREATE_JOB(  
        job_name => 'job2',  
        program_name => 'program1',  
        start_date => SYSTIMESTAMP,  
        repeat_interval => 'TRUNC(SYSDATE) + 1 + 1/24',  
        enabled => TRUE);
```

```
END;
```

```
BEGIN
```

```
    DBMS_SCHEDULER.CREATE_SCHEDULE(  
        schedule_name => 'schedule1',  
        repeat_interval => 'freq=monthly; byday=SAT; byhour=3');
```

```
END;
```

```
BEGIN
```

```
    DBMS_SCHEDULER.CREATE_JOB(  
        job_name => 'job3',  
        program_name => 'program1',  
        schedule_name => 'schedule1',  
        enabled => TRUE);
```

```
END;
```

Das Intervall in schedule1 bedeutet: jeden Samstag um 3 Uhr. Da keine Minuten- und Sekundenangabe erfolgt, werden diese Angaben aus start\_date ermittelt.

## Data Dictionary Views

Mit dem neuen Scheduler ist eine ganze Reihe neuer DD-Views dazugekommen. Insbesondere werden einzelne Ausführungen jetzt ebenfalls im Data Dictionary dokumentiert, egal, ob sie erfolgreich waren oder nicht. Bei Misserfolg wird zusätzlich der Grund mitprotokolliert. Standardmäßig werden diese Informationen 30 Tage lang aufbewahrt.

Die wichtigsten DD-Views sind:

- DBA\_SCHEDULER\_JOBS
- DBA\_SCHEDULER\_RUNNING\_JOBS
- DBA\_SCHEDULER\_PROGRAMS
- DBA\_SCHEDULER\_SCHEDULES
- DBA\_SCHEDULER\_JOB\_RUN\_DETAILS
- DBA\_SCHEDULER\_JOB\_LOG

## Weiterführende Konzepte

Dieser Tipp zeigt nur die grundlegenden Möglichkeiten des neuen Schedulers auf, die mit dem CREATE JOB-Recht zur Verfügung stehen. Für alle weiterführenden Konzepte ist das MANAGE\_SCHEDULER-Recht nötig, das in den Rollen DBA und SCHEDULER\_ADMIN enthalten ist. Dazu gehören:

- die generelle Verwaltung des Schedulers (z. B. muss das generelle Attribut DEFAULT\_TIMEZONE auf die passende Zeitzone gesetzt werden!)
- das Anlegen von "Windows"; das sind Zeitfenster, die (statt eines genauen Zeitpunkts) zur Ausführung zur Verfügung stehen
- das Anlegen von "Window Groups"
- das Anlegen von "Job Classes"

Weitere Einzelheiten finden Sie in der Oracle-Dokumentation.

#### **Anmerkungen:**

Das Arbeiten mit DBMS\_JOB ist auch unter 10g und 11g weiterhin möglich. Jobs, die mit dem alten bzw. dem neuen Scheduler angelegt wurden, werden getrennt voneinander verwaltet, auch wenn die gleichen Hintergrundprozesse für die Ausführung zuständig sind. Zur Ausführung von Jobs, die über DBMS\_JOB angelegt wurden, muss auch weiterhin job\_queue\_processes auf einen Wert > 0 gesetzt werden.

Ein Unterschied zu DBMS\_JOB besteht darin, dass die DBMS\_SCHEDULER-Aufrufe letztlich Datenbank-Objekte erzeugen bzw. löschen und daher ein implizites Commit absetzen. Werden in PL/SQL-Routinen Jobs generiert oder gelöscht, sollte daher darauf geachtet werden, dass diese Aufrufe nicht das Transaktionskonzept der Applikation aushebeln. Gegebenenfalls ist es hier sinnvoller, bei DBMS\_JOB zu bleiben.