DBMS_SCHEDULER パッケージ

機能

プロシージャをジョブ登録して、スケジュールによる起動を行う

【時間等間隔での起動方法】

【カレンダー式での起動方法】

【期日指定スケジュールでの起動方法】

の３通りの使い方がある

DBMS_SCHEDULER で用意されているプロシージャ

プロシージャ名	機　　能　　説　　明
CREATE_JOB	新規ジョブを作成する
CREATE_SCHEDULE	新規スケジュールを作成する
CREATE_JOB	指定したジョブを削除する

使い方概要

【時間等間隔での起動方法】

CREATE_JOB プロシージャを使って、実行したいプログラム名と実行間隔を指定してスケジュールする

repeat_interval　=>　'SYSDATE+1/1440' ,　1 分間隔

※ 実行したいプログラム名 :「**実行タイプ**」 と 「**実行プログラム名**」

実行したいプログラム名の指定例

実行タイプ　job_type =>	実行プログラム名　　job_action =>
'PLSQL_BLOCK'	PL/SQL ブロック名 例）'BEGIN　PRO_1( ) ;　END ;'
'STORED_PROCEDURE'	ストアド・プロシージャ名 例）'JOB_TEST'
'EXECUTABLE'	外部ファイル名 例）'/disk1/sample1-1.sh' 'c:¥user¥test.exe'

【カレンダー式での起動方法】

1）CREATE_SCHEDULE プロシージャを使って、実行したいスケジュール（月末実行、夜間定時実行）の登録を行う

2）CREATE_JOB プロシージャを使って、実行したいプログラム名と 1）で登録したスケジュール名~~と実行間隔~~を指定して、実行ジョブとスケジュール時間を対応させる

実行間隔は、指定しない

schedule_name　=>　'1）のスケジュール名' ,

【期日指定スケジュールでの起動方法】

【カレンダー式での起動方法】と同様に、CREATE_SCHEDULE と CREATE_JOB を実施する

ただし、スケジュールの日時を変更する

start_time で、起動したい日時を指定して、end_date で、リピートされる前の日時を指定すれば、1 回限りのスケジュールとなる

例）

```
start_date          =>  to_date('2015/10/01 22:00:00','yyyy/mm/dd hh24:mi:ss'),
repeat_interval     =>  'FREQ=YEARLY;INTERVAL=99',
end_date            =>  to_date('2015/10/02 22:00:00','yyyy/mm/dd hh24:mi:ss'),
```

≪ CREATE_SCHEDULE 受渡しパラメータ ≫

repeat_interval

例） repeat_interval => 'FREQ=DAILY;INTERVAL=1' 1 日ごとに実行ジョブの実行間隔を設定

※ ジョブは次回実行時間という形で保持しているため、毎日であれば 24 時間ごとという指定になる

そのため、start_date が過去日付の場合、ジョブを有効化した時間に毎日実行することになる

例 start_date => sysdate,

repeat_interval => 'FREQ=DAILY;' 毎日実行指定

上記設定で 15 時 32 分 45 秒にジョブを有効化した場合

ジョブは毎日 15 時 32 分 45 秒に action を実行する。

FREQ

間隔(秒分時日週月年毎指定)指定間隔の初日(月曜、1 日)に実行

秒	SECONDLY
分	MINUTELY
時	HOURLY
日	DAILY
週	WEEKLY
月	MONTHLY
年	YEARLY

INTERVAL

上記設定に＋して、何回満たしたら実行するかを指定

例）repeat_interval => 'FREQ=DAILY;INTERVAL=2' 2 日ごとに実行

BYMONTH

実行月を指定します。2 月なら 2、3 文字の略語でも可能。3 月なら MAR

例）repeat_interval => 'FREQ=DAILY;BYMONTH=2' 2 月に毎日実行

毎週水曜日に実行するには、

例）repeat_interval　=>　'FREQ=WEEKLY;BYDAY=WED'）

月曜日：	MON
火曜日：	TUE
水曜日：	WED
木曜日：	THU
金曜日：	FRI
土曜日：	SAT
日曜日：	SUN

サンプル・コード

【時間等間隔での起動方法】

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
            job_name   =>   'PRO_JOB' ,
            job_type   =>   'STORED_PROCEDURE' ,
            job_action   =>   'KOZUE.TABLE_WRITE_PROCEDURE' ,
            start_date   =>   SYSDATE ,
            repeat_interval   =>   'SYSDATE+1/1440' ,
            enabled   =>   TRUE ) ;
END ;

/          ←― 無名 PL/SQL ブロックでのコード実行のための／（スラッシュ）
```

（登録スケジュールの確認）

```
SELECT   job_name ,   job_action ,   job_type ,   repeat_interval ,
            to_char( last_start_date ,   'HH24:MI:SS' ) ,
            to_char( next_run_date ,   'HH24:MI:SS' )
    FROM   user_scheduler_jobs
    WHERE   job_name   =   'PRO_JOB' ;
```

（登録スケジュールの削除）

```
EXECUTE   DBMS_SCHEDULER.DROP_JOB( 'PRO_JOB' ) ;
```

【カレンダー式での起動方法】、【期日指定スケジュールでの起動方法】

（スケジュールの登録）

```
BEGIN
  DBMS_SCHEDULER.CREATE_SCHEDULE(
                        schedule_name   =>  'WED_SCHEDULE' ,
                        start_date   =>   SYSDATE + 1/1440 ,
                        repeat_interval   =>   'FREQ=WEEKLY;BYDAY=WED' ) ;
END;

/          ←― 無名 PL/SQL ブロックでのコード実行のための／（スラッシュ）
```

（ジョブの登録）

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
                          job_name   =>   'PRO_SCHE_JOB' ,
                          schedule_name   =>  'WED_SCHEDULE' ,
                          job_type   =>   'STORED_PROCEDURE' ,
                          job_action   =>   'KOZUE.TABLE_WRITE_PROCEDURE' ,
                          enabled   =>   TRUE ) ;
END;

/          ←― 無名 PL/SQL ブロックでのコード実行のための／（スラッシュ）
```

（登録スケジュールの確認）

```
SELECT   schedule_name ,   schedule_type ,   repeat_interval ,
            to_char(start_date,'YYYY/MM/DD (DAY)')
    FROM   user_scheduler_schedules
    WHERE   schedule_name   =   'WED_SCHEDULE' ;
```

（登録ジョブの確認）

```
SELECT job_name ,   schedule_name ,   job_action ,
        to_char( start_date ,   'YYYY/MM/DD (DAY) HH24:MI:SS ' ) ,
        to_char( next_run_date ,   'YYYY/MM/DD (DAY) HH24:MI:SS' )
    FROM   user_scheduler_jobs
    WHERE   job_name   =   'PRO_SCHE_JOB' ;
```