

## 実行時間のスケジュール PL/SQL プロシージャの起動時間の指定をする

DBMS\_JOB パッケージを使うと、PL/SQL プロシージャなどを時刻指定で起動させることが出来ます。

【注意】 サーバー状で、IntelligentAgent サービスが起動している必要があります。  
パッケージのプロシージャは、以下の通りです

パッケージ名 : DBMS\_JOB.

プロシージャ名	サービス提供の内容説明
SUBMIT	新規スケジュールのジョブを作成する
INTERVAL	ジョブの実行間隔を変更する (間隔指定)
NEXT_DATE	ジョブの次回の実行する日時を指定する (日時指定)
REMOVE	スケジュール内容 (ジョブ) の削除

## 構成プロシージャについて

プロシージャ名：SUBMIT

機能説明：新規スケジュールのジョブを作成する

受渡し引数

引数名	I/O	データ型	説明
job	OUT	BINARY_INTEGER	スケジュールされたジョブ番号
what	IN	VARCHAR2	実行する PL/SQL プロシージャ名
next_date	IN	DATE	ジョブを実行する日時 DEFAULT SYSDATE
interval	IN	VARCHAR2	次回実行するジョブ日時、これを基に 時間間隔が計算される DEFAULT 'NULL'

## プロシージャの呼出し方

```
/* OUT 用引数の受け取り変数の定義 */
```

```
SQL> VARIABLE JobNo NUMBER
```

```
/* パッケージの起動 */
```

```
/* OUT 用パラメータには、バインド変数としてコロン「:」を付ける */
```

```
/* プロシージャ名には、セミコロン「;」まで記述する */
```

```
SQL> EXECUTE DBMS_JOB.SUBMIT(:JobNo, 'usr_pakege.purocedure1;',  
SYSDATE, 'SYSDATE + 1') ;
```

```
/* もしくは、*/
```

```
SQL> BEGIN  
DBMS_JOB.SUBMIT( :JobNo, 'usr_pakege.purocedure1;',  
SYSDATE, 'SYSDATE + 1') ;  
END ;
```

```
SQL> /
      /* この/は、パッケージの実行指示 */
```

```
/* COMMIT の発行 */
SQL> COMMIT ;
```

```
/* ジョブ番号の出力表示 */
SQL> PRINT JobNo
```

実行例)

```
/* テスト用プロシージャの作成 */
CREATE OR REPLACE PROCEDURE TABLE_WRITE_PROCEDURE
IS
  wk_sql VARCHAR2(1024);
  o_no VARCHAR2(1024);
BEGIN
  wk_sql := 'SELECT MAX(empno) FROM emp_table';
  EXECUTE IMMEDIATE wk_sql INTO o_no;
  wk_sql := 'INSERT INTO emp_table VALUES (' || TO_CHAR(o_no + 1)
    || ', "プロシージャでの書込",';
  wk_sql := wk_sql || "' || TO_CHAR(SYSDATE,
    'YYYY/MM/DD HH24:MI:SS') || "' || ', 0)';
  EXECUTE IMMEDIATE wk_sql;
  COMMIT;
END TABLE_WRITE_PROCEDURE;
/

/* スケジュールの登録 */
/* OUT 用引数の受け取り変数の定義 */
VARIABLE JobNo NUMBER

/* パッケージの起動 */
/* OUT 用パラメータには、バインド変数としてコロン「:」を付ける */
/* プロシージャ名には、セミコロン「;」まで記述する */
EXECUTE DBMS_JOB.SUBMIT(:JobNo, 'TABLE_WRITE_PROCEDURE;',
  SYSDATE, 'SYSDATE + 1') ;

/* COMMIT の発行 */
COMMIT ;

/* ジョブ番号の出力表示 */
PRINT JobNo

/* スケジュールの変更 */
/* 時間間隔 . . . . . これは、次回起動してから反映される */
EXECUTE DBMS_JOB.INTERVAL(:JobNo, 'SYSDATE+1/1440') ;
COMMIT ;

/* 指定時間 */
EXECUTE DBMS_JOB.NEXT_DATE(:JobNo, TO_DATE('2017/10/12 15:35:00',
  'YYYY/MM/DD HH24:MI:SS')) ;
COMMIT;

/* スケジュールの削除 */
EXECUTE DBMS_JOB.REMOVE(:JobNo) ;
COMMIT ;
```

## 構成プロシージャについて

プロシージャ名：INTERVAL

機能説明：ジョブの実行間隔を変更する（間隔指定）

受渡し引数

引数名	I/O	データ型	説明
job	IN	BINARY_INTEGER	スケジュールされたジョブ番号
interval	IN	VARCHAR2	次回実行するジョブ日時以降に反映される DEFAULT

## プロシージャの呼出し方

```
/* パッケージの起動 */
```

```
/* 12 時間間隔での起動指定 */
```

```
SQL> EXECUTE DBMS_JOB.INTERVAL ( 5558, 'SYSDATE+12/24' ) ;
```

```
SQL> /
```

```
/* COMMIT の発行 */
```

```
SQL> COMMIT ;
```

### 構成プロシージャについて

プロシージャ名：NEXT\_DATE

機能説明：ジョブの次回の実行する日時を指定する（日時指定）

受渡し引数

引数名	I/O	データ型	説明
job	IN	BINARY_INTEGER	スケジュールされたジョブ番号
next_date	IN	DATE	ジョブを実行する日時 DEFAULT SYSDATE

### プロシージャの呼出し方

```
/* パッケージの起動 */
```

```
/* 12 時間間隔での起動指定 */
```

```
SQL> EXECUTE DBMS_JOB.NEXT_DATE ( 5558, 'SYSDATE+12/24' ) ;
```

```
/* COMMIT の発行 */
```

```
SQL> COMMIT ;
```

```
/* 期日指定 */
```

```
SQL> EXECUTE DBMS_JOB.NEXT_DATE ( 5558, TO_DATE('2009/02/14  
13:10:30', 'YYYY/MM/DD HH24:MI:SS') ) ;
```

```
/* COMMIT の発行 */
```

```
SQL> COMMIT ;
```

### 構成プロシージャについて

プロシージャ名：REMOVE

機能説明：スケジュール内容（ジョブ）の削除

受渡し引数：

引数名	I/O	データ型	説明
job	IN	BINARY_INTEGER	スケジュールされたジョブ番号

### プロシージャの呼出し方

```
/* パッケージの起動 */
```

```
SQL> EXECUTE DBMS_JOB.REMOVE ( 5558 ) ;
```

```
SQL> /
```

```
SQL> COMMIT ;
```