BASE DE DATOS II
UNIDAD III

INTRODUCCIÓN A JOBS Y SQL LOADER

JOBS

Los JOBS son tareas programadas que se crean en el gestor de la base de datos, esta tarea programada al ejecutarse realiza el llamado al objeto que se especifique y el cual se encargará de realizar acciones en específico.

Estos JOBS se pueden ejecutar de forma repetitiva y de forma automática, en los parámetros de configuración del JOB se indica la fecha en la cual se ejecutará por primera vez el JOB y también se especifica con qué frecuencia se estará ejecutando esta tarea.

Se pueden utilizar para realizar tareas repetitivas en la base de datos y que sabemos que se deben ejecutar cada cierto tiempo.

JOBS

La sintaxis para crear el JOB es la siguiente:

```
BEGIN
        -----Esta función sirve para crear el job y configurar los parámetros de funcionamiento----
        DBMS_SCHEDULER.CREATE_JOB (
                job_name=> <AQUÍ SE DEBE INDICAR EL NOMBRE DEL JOB>,
                job_type=> <AQUÍ SE DEBE INDICAR EL TIPO DE ELEMENTO A SER LLAMADO>,
                job_action=> <AQUÍ SE DEBE INDICAR EL NOMBRE DEL ELEMENTO A LLAMAR>,
                number_of_arguments => <SE INDICA LA CANTIDAD DE PARÁMETROS QUE RECIBE EL
                ELEMENTO QUE SE ESTÁ LLAMANDO EN EL JOB_ACTION, SI EL VALOR ES MAYOR QUE 0
                ENTONCES EL VALOR ENABLED DEBE ESPECIFICARSE EN FALSE>,
                start_date=><AQUÍ SE INDICA LA FECHA EN LA CUAL SE EJECUTARÁ EL JOB POR PRIMERA VEZ>,
                repeat_interval=> <AQUÍ SE INDICA CON QUÉ FRECUENCIA SE EJECUTA EL JOB>,
                enabled=><AQUÍ  SE INDICA EL VALOR TRUE O FALSE, SE INDICA TRUE SI EL VALOR DE
                number_of_arguments ES IGUAL A CERO>
        );
END;
```

JOBS

----Esta función sirve para indicar los valores que reciben los parámetros del elemento al cual llama el JOB---

```
BEGIN
        DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE (
                job_name=><AQUÍ SE INDICA EL NOMBRE DEL JOB>,
                argument_position=> <AQUÍ SE INDICA LA POSICIÓN DEL ARGUMENTO>,
                argument_value=><AQUÍ SE INDICA EL VALOR DEL ARGUMENTO>
        );
END;
```

JOBS

---Esta función se utiliza para habilitar un JOB y que se pueda ejecutar de forma automática en el tiempo establecido en la creación del JOB

```
BEGIN
        DBMS_SCHEDULER.enable(<AQUÍ SE INDICA EL NOMBRE DEL JOB>);
END;
```

--Para borrar el job se ejecuta la siguiente instrucción

```
BEGIN
        -----Esta función sirve para borrar un JOB
        DBMS_SCHEDULER.DROP_JOB (
                job_name=> <AQUÍ SE INDICA EL NOMBRE DEL JOB>
        );
END;
```

JOBS

```
BEGIN
        -----Esta función sirve para ejecutar de forma manual un JOB
        DBMS_SCHEDULER.RUN_JOB (
                job_name=> <AQUÍ SE INDICA EL NOMBRE DEL JOB>
        );
END;

--para ver los jobs que se han ejecutado y si han tenido éxito
SELECT JOB_NAME, LOG_DATE, STATUS FROM USER_SCHEDULER_JOB_LOG;

--para ver los jobs que se han creado
SELECT * FROM USER_SCHEDULER_JOBS;
```

JOBS

Ejemplo de un JOB que se ejecuta diariamente y que hace el llamado a un procedimiento almacenado

```
--Primero creamos el procedimiento almacenado
CREATE OR REPLACE PROCEDURE SP_NOMB_BRAND_MAYUS
IS
BEGIN
        UPDATE BRANDS SET BRAND_NAME=UPPER(BRAND_NAME);
        COMMIT;
END;
```

JOBS

Ejemplo de un JOB que se ejecuta diariamente y que hace el llamado a un procedimiento almacenado

```
--luego creamos el JOB
BEGIN
        DBMS_SCHEDULER.CREATE_JOB(
                JOB_NAME=>'JOB_MAYUS_BRAND_NAME',
                JOB_TYPE=>'STORED_PROCEDURE',
                JOB_ACTION=>'SP_NOMB_BRAND_MAYUS',
                NUMBER_OF_ARGUMENTS=>0,
                START_DATE=>TO_TIMESTAMP('2020-03-31 09:45:00','YYYY-MM-DD HH24:MI:SS'),
                REPEAT_INTERVAL=>'FREQ=DAILY',
                ENABLED=>TRUE
        );
END;
```

SQL LOADER

Es una herramienta que proporciona ORACLE para poder cargar datos de forma masiva y cuyo origen de datos son archivos.

Para realizar esta tarea necesitamos realizar tres pasos:

1) Crear un archivo que tendrá los datos que se desean cargar
2) Un archivo de control para saber cómo se deben interpretar los datos en el archivo de datos, además de dónde y cómo deseamos cargarlos
3) Ejecutar el comando necesario para que la herramienta SQL LOADER se encargue de cargar los datos en la tabla que indiquemos

SQL LOADER

Tipos de datos soportados por SQL LOADER

- CHAR=> PARA CAMPOS CHAR/VARCHAR2
- DATE=> PARA CAMPOS DATE
- INTEGER EXTERNAL, DECIMAL EXTERNAL=> PARA CAMPOS NUMÉRICOS
- SMALLINT=> PARA CAMPOS NUMÉRICOS
- FLOAT=> PARA CAMPOS NUMÉRICOS
- DOUBLE=> PARA CAMPOS NUMÉRICOS

SQL LOADER

Ejemplo para utilizar la herramienta SQL LOADER

1) Primero creamos el archivo de datos, este archivo se llamará carga_datos.csv

IDDEPARTAMENTO,NOMBREDEPARTAMENTO,IDPAIS

5,Atlántida,1

6,Colón,1

7,Comayagua,1

8,Copán,1

9,Choluteca,1

10,El Paraíso,1

11,Gracias a Dios,1

SQL LOADER

Ejemplo para utilizar la herramienta SQL LOADER

1) Primero creamos el archivo de datos, este archivo se llamará carga_datos.csv

IDDEPARTAMENTO,NOMBREDEPARTAMENTO,IDPAIS

5,Atlántida,1

6,Colón,1

7,Comayagua,1

8,Copán,1

9,Choluteca,1

10,El Paraíso,1

11,Gracias a Dios,1

SQL LOADER

Ejemplo para utilizar la herramienta SQL LOADER

2) Luego creamos el archivo de control, este archivo se llamará control.ctl

```
OPTIONS (SKIP='1')
LOAD DATA
APPEND INTO TABLE TBLDEPARTAMENTOS
WHEN (IDPAIS='1’)
(           IDDEPARTAMENTO INTEGER EXTERNAL TERMINATED BY ",",
            NOMBREDEPARTAMENTO CHAR TERMINATED BY ",",
            IDPAIS INTEGER EXTERNAL TERMINATED BY ","
)
```

SQL LOADER

Ejemplo para utilizar la herramienta SQL LOADER

3) Por último, ejecutamos el comando necesario para hacer uso de la herramienta

sqlldr <nombre de usuario del gestor de base de datos>/<contraseña del usuario>@<nombre de la instancia de ORACLE> control='<ruta y nombre del archivo de control>' data='<ruta y nombre del archivo de datos a cargar>'

sqlldr user1/123@XE control='C:\control.ctl' data='C:\carga_datos.csv'