PLSQL 创建Oracle定时任务

用语句创建

1、创建任务执行的存储过程，如名称为testJob，向测试表中插入数据，如文章开头，此处省略。

2、创建一个 定时任务 job

declare

job number;

BEGIN

DBMS\_JOB.SUBMIT(

JOB => job, /\*自动生成JOB\_ID\*/

WHAT => 'testJob;', /\*需要执行的存储过程名称或SQL语句\*/

NEXT\_DATE => sysdate, /\*初次执行时间-立即执行\*/

INTERVAL => 'trunc(sysdate,''mi'')+1/(24\*60)' /\*每隔1分钟执行一次\*/

);

commit;

end;

3、查询定时任务相关字段信息

select \* from user\_jobs;

4 、停止定时任务

declare

begin

dbms\_job.broken(24,true,sysdate); /\*停止一个job,jobId, job的ID,里面参数true也可是false，next\_date（某一时刻停止）也可是sysdate（立刻停止）。 \*/

commit;

end;

5、启动指定定时任务

declare

begin

DBMS\_JOB.RUN(24); /\*24 job的id\*/

commit;

end;

6、删除定时任务

declare

begin

dbms\_job.remove(24); /\*删除自动执行的job,参数是 job的id\*/

commit;

end;

7、修改定时任务的间隔时间

declare

begin

dbms\_job.interval(24,interval => 'TRUNC(SYSDATE)+1'); /\*第一个参数为job的ID,第二个参数interval: 计算下一次任务执行的时间表达式\*/

commit;

end;

8、修改下一次执行时间

declare

begin

dbms\_job.next\_date(24,to\_date('2020-11-9 12:08:00','yyyy-mm-dd hh24:mi:ss')); /\*第一个参数:job的ID；第二个参数:要修改后的计算下一次执行的时间表达式\*/

commit;

end;

9、修改定时任务要执行的操作

declare

begin

dbms\_job.what(24,'testJob2();'); /\* 第一个参数:job的ID；第二个参数:要更改的新操作名称（操作名必须存在）\*/

commit;

end;

三、结

1、dba\_jobs 表中字段含义

dba\_jobs 表中字段含义：

JOB 任务的唯一标识码

LOG\_USER 提交任务的用户

PRIV\_USER 赋予任务权限的用户

SCHEMA\_USER 对用户作语法分析的用户模式

LAST\_DATE 最后一次成功执行任务的时间

LAST\_SEC 最后一次成功执行任务的时间的时分秒

THIS\_DATE 正在执行的任务的开始时间，若没有则为空

THIS\_SEC 正在执行的任务的开始时间的时分秒，若没有则为空

NEXT\_DATE 下一次执行定时任务的时间

NEXT\_SEC 下一次执行定时任务的时间的时分秒

TOTAL\_TIME 执行当前任务所需要的时间，单位：秒

BROKEN 标志参数，Y表示任务中断，以后不会再运行

INTERTAL 计算下一次执行定时任务的时间表达式

FAILURES 当前定时任务执行失败的总次数

WHAT 执行任务的PL/SQL代码块

NLS\_ENV 任务执行的NLS会话设置

MISC\_ENV 定时任务运行的其他一些参数设置

INSTANCE 标识当前任务运行是否受限，0 没有受限

2、INTERVAL 部分参数值示

INTERVAL 部分参数值示例：

每天午夜12点: 'TRUNC(SYSDATE + 1)'

每天早上8点30分: 'TRUNC(SYSDATE + 1) + （8\*60+30）/(24\*60)'

每星期二中午12点: 'NEXT\_DAY(TRUNC(SYSDATE ), ''TUESDAY'' ) + 12/24'

每个月第一天的午夜12点: 'TRUNC(LAST\_DAY(SYSDATE ) + 1)'

每个季度最后一天的晚上11点: 'TRUNC(ADD\_MONTHS(SYSDATE + 2/24, 3 ), 'Q' ) -1/24'

每星期六和日早上6点10分: 'TRUNC(LEAST(NEXT\_DAY(SYSDATE, ''SATURDAY"), NEXT\_DAY(SYSDATE, "SUNDAY"))) + （6×60+10）/（24×60）'

每月25号00：00执行: 'TRUNC(LAST\_DAY(SYSDATE ) + 25)'

－－－－－－－－－－－－－－－－－－－－－－－－－－

1:每分钟执行

Interval => TRUNC(sysdate,'mi') + 1/ (24\*60)

或

Interval => sysdate+1/1440

2:每天定时执行

例如：每天的凌晨1点执行

Interval => TRUNC(sysdate) + 1 +1/ (24)

3:每周定时执行

例如：每周一凌晨1点执行

Interval => TRUNC(next\_day(sysdate,'星期一'))+1/24

4:每月定时执行

例如：每月1日凌晨1点执行

Interval =>TRUNC(LAST\_DAY(SYSDATE))+1+1/24

5:每季度定时执行

例如每季度的第一天凌晨1点执行

Interval => TRUNC(ADD\_MONTHS(SYSDATE,3),'Q') + 1/24

6:每半年定时执行

例如：每年7月1日和1月1日凌晨1点

Interval => ADD\_MONTHS(trunc(sysdate,'yyyy'),6)+1/24

7:每年定时执行

例如：每年1月1日凌晨1点执行

Interval =>ADD\_MONTHS(trunc(sysdate,'yyyy'),12)+1/24

————————————————

版权声明：本文为CSDN博主「J\*C」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/JCmyjava/article/details/123570222