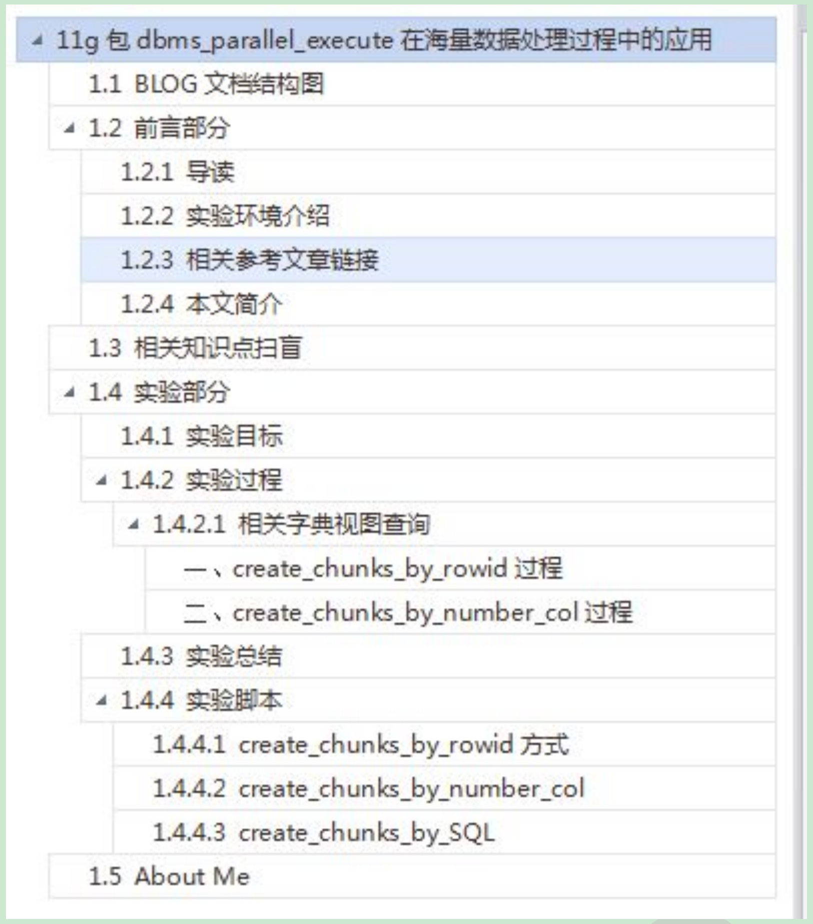
11g 包 dbms_parallel_execute 在海量数据处理过程中的应用

1.1 BLOG 文档结构图



11g 包 dbms_parallel_execute 在海量数据处理过程中的应用
1.1 BLOG 文档结构图
1.2 前言部分
1.2.1 导读
1.2.2 实验环境介绍
1.2.3 相关参考文章链接
1.2.4 本文简介
1.3 相关知识点扫盲
1.4 实验部分
1.4.1 实验目标
1.4.2 实验过程
1.4.2.1 相关字典视图查询
一、create_chunks_by_rowid 过程
二、create_chunks_by_number_col 过程
1.4.3 实验总结
1.4.4 实验脚本
1.4.4.1 create_chunks_by_rowid 方式
1.4.4.2 create_chunks_by_number_col
1.4.4.3 create_chunks_by_SQL
1.5 About Me

1.2 前言部分

1.2.1 导读

各位技术爱好者，看完本文后，你可以掌握如下的技能，也可以学到一些其它你所不知道的知识，~O(∩_∩)O~：

① 11g 包 dbms_parallel_execute 在海量数据处理过程中的应用

注意：本篇 BLOG 中代码部分需要特别关注的地方我都用黄色背景和红色字体来表示，比如下边的例子中，thread 1 的最大归档日志号为 33，thread 2 的最大归档日志号为 43

是需要特别关注的地方。

```
List of Archived Logs in backup set 11
Thrd Seq     Low SCN    Low Time            Next SCN   Next Time
---- ------- ---------- ------------------- ---------- ---------
1    32      1621589    2015-05-29 11:09:52 1625242    2015-05-29 11:15:48
1    33      1625242    2015-05-29 11:15:48 1625293    2015-05-29 11:15:58
2    42      1613951    2015-05-29 10:41:18 1625245    2015-05-29 11:15:49
2    43      1625245    2015-05-29 11:15:49 1625253    2015-05-29 11:15:53
```

本文如有错误或不完善的地方请大家多多指正，ITPUB 留言或 QQ 皆可，您的批评指正是我写作的最大动力。

1.2.2 实验环境介绍

11.2.0.1 RHEL6.5

1.2.3 相关参考文章链接

Oracle 中如何更新一张大表记录	http://blog.itpub.net/26736162/viewspace-1684095/
使用 11g dbms_parallel_execute 执行并行更新（下）	http://blog.itpub.net/26736162/viewspace-1683913/
使用 11g dbms_parallel_execute 执行并行更新（上）	http://blog.itpub.net/26736162/viewspace-1683912/

1.2.4 本文简介

一个朋友 own_my 要处理批量数据，但是脚本跑的太慢了，于是网上搜到了 dbms_parallel_execute 这个包，用完后给我说这个包非常强大，于是我也学习学习，关于优化一直是我喜欢的内容，在参考了大神 realkid4 的 blog 后，我自己也做了做实验，感觉很强大，记录在此。

1.3 相关知识点扫盲

参考大神的 blog：http://blog.itpub.net/17203031/

1.4 实验部分

1.4.1 实验目标

测试 dbms_parallel_execute 包在海量数据处理过程中的应用。

1.4.2 实验过程

```
[oracle@etlhost206 ~]$ sqlplus / as sysdba

SQL*Plus: Release 11.2.0.1.0 Production on Wed Jun 3 13:40:34 2015

Copyright (c) 1982, 2009, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> CONN  LHR/lhr
Connected.
SQL> CREATE TABLE T AS SELECT * FROM DBA_OBJECTS;

Table created.

SQL> insert into t select * from t;

76369 rows created.

SQL> insert into t select * from t;

152738 rows created.

SQL> insert into t select * from t;

305476 rows created.

SQL> COMMIT;

Commit complete.
```

```
SQL> insert into t select * from t;

610952 rows created.

SQL> insert into t select * from t;

1221904 rows created.

SQL> insert into t select * from t;

2443808 rows created.

SQL> insert into t select * from t;

4887616 rows created.

SQL> COMMIT;

Commit complete.

SQL> insert into t select * from t;

9775232 rows created.

SQL> COMMIT;

Commit complete.

SQL> insert into t select * from t;

19550464 rows created.

SQL> COMMIT;

Commit complete.

SQL> select bytes/1024/1024 from dba_segments a where a.segment_name='T';

BYTES/1024/1024
---------------
           4341

SQL> SELECT COUNT(1) FROM T;

  COUNT(1)
----------
  39100928

SQL> show parameter job

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
job_queue_processes                  integer     1000
SQL> show parameter cpu

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
cpu_count                            integer     8
parallel_threads_per_cpu             integer     2
resource_manager_cpu_allocation      integer     8

SQL> set timing on
SQL> set time on;
15:50:01 SQL>
15:50:02 SQL> show parameter job
```

```
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
job_queue_processes                  integer     1000
15:50:09 SQL>  select bytes/1024/1024 from dba_segments a where a.segment_name='T';

BYTES/1024/1024
---------------
           4341

Elapsed: 00:00:00.41
15:50:31 SQL> declare
15:50:39   2    vc_task  varchar2(100);
15:50:39   3    vc_sql   varchar2(1000);
15:50:39   4    n_try    number;
15:50:39   5    n_status number;
15:50:39   6  begin
15:50:39   7    --Define the Task
15:50:39   8    vc_task := 'Task 1: By Rowid'; --Task 名称
15:50:39   9    dbms_parallel_execute.create_task(task_name => vc_task); --手工定义一个 Task 任务;
15:50:39  10
15:50:39  11    --Define the Spilt
15:50:39  12    dbms_parallel_execute.create_chunks_by_rowid(task_name   => vc_task,
15:50:39  13                                                 table_owner => 'LHR',
15:50:39  14                                                 table_name  => 'T',
15:50:39  15                                                 by_row      => true,
15:50:39  16                                                 chunk_size  => 10000); --定义 Chunk
15:50:39  17
15:50:39  18    vc_sql := 'update /*+ ROWID(dda) */ t set DATA_OBJECT_ID=object_id+1 where rowid between :start_id and :end_id';
15:50:40  19    --Run the task
15:50:40  20    dbms_parallel_execute.run_task(task_name      => vc_task,
15:50:40  21                                   sql_stmt       => vc_sql,
15:50:40  22                                   language_flag  => dbms_sql.native,
15:50:40  23                                   parallel_level => 4); --执行任务，确定并行度
15:50:40  24
15:50:40  25    --Controller
15:50:40  26    n_try    := 0;
15:50:40  27    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
15:50:40  28    while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
15:50:40  29      dbms_parallel_execute.resume_task(task_name => vc_task);
15:50:40  30      n_status := dbms_parallel_execute.task_status(task_name => vc_task);
15:50:40  31    end loop;
15:50:40  32
15:50:40  33    --Deal with Result
15:50:40  34    dbms_parallel_execute.drop_task(task_name => vc_task);
15:50:40  35  end;
15:50:40  36  /


PL/SQL procedure successfully completed.

Elapsed: 00:03:50.78
15:58:05 SQL>
15:58:06 SQL> create index idx_t_id on t(object_id) nologging parallel 4;

Index created.

Elapsed: 00:01:35.12
16:00:05 SQL> alter index idx_t_id noparallel;

Index altered.

Elapsed: 00:00:00.07
16:00:15 SQL>
16:02:51 SQL> declare
16:02:52   2    vc_task  varchar2(100);
16:02:52   3    vc_sql   varchar2(1000);
16:02:52   4    n_try    number;
```

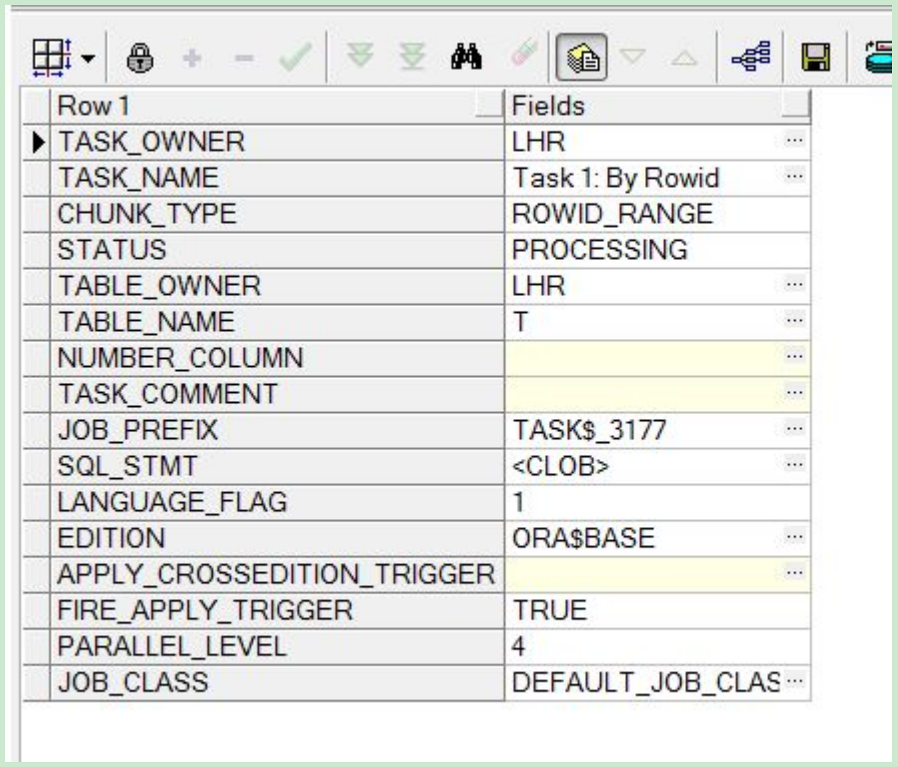
```
16:02:52   5    n_status number;
16:02:52   6  begin
16:02:52   7    --Define the Task
16:02:52   8    vc_task := 'Task 2: By Number Col';
16:02:52   9    dbms_parallel_execute.create_task(task_name => vc_task);
16:02:52  10
16:02:52  11    --Define the Spilt
16:02:52  12    dbms_parallel_execute.create_chunks_by_number_col(task_name    => vc_task,
16:02:52  13                                                      table_owner  => 'LHR',
16:02:52  14                                                      table_name   => 'T',
16:02:52  15                                                      table_column => 'OBJECT_ID',
16:02:52  16                                                      chunk_size   => 100000); --定义 chunk

16:02:53  17  16:02:53  18    vc_sql := 'update /*+ ROWID(dda) */ t set DATA_OBJECT_ID=object_id+1 where object_id between :start_id and :end_id';
16:02:53  19    --Run the task
16:02:53  20    dbms_parallel_execute.run_task(task_name      => vc_task,
16:02:53  21                                   sql_stmt       => vc_sql,
16:02:53  22                                   language_flag  => dbms_sql.native,
16:02:53  23                                   parallel_level => 4);
16:02:53  24
16:02:53  25    --Controller
16:02:53  26    n_try    := 0;
16:02:53  27    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
16:02:53  28    while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
16:02:53  29      dbms_parallel_execute.resume_task(task_name => vc_task);
16:02:53  30      n_status := dbms_parallel_execute.task_status(task_name => vc_task);
16:02:53  31    end loop;
16:02:53  32
16:02:53  33    --Deal with Result
16:02:53  34    dbms_parallel_execute.drop_task(task_name => vc_task);
16:02:53  35  end;
16:02:53  36  /
^Cdeclare
*
ERROR at line 1:
ORA-01013: user requested cancel of current operation
ORA-06512: at "SYS.DBMS_LOCK", line 201
ORA-06512: at "SYS.DBMS_PARALLEL_EXECUTE", line 44
ORA-06512: at "SYS.DBMS_PARALLEL_EXECUTE", line 390
ORA-06512: at "SYS.DBMS_PARALLEL_EXECUTE", line 417
ORA-06512: at line 20


Elapsed: 00:07:12.08

16:11:36 SQL>
16:11:36 SQL> EXEC   dbms_parallel_execute.drop_task(task_name => 'Task 2: By Number Col');

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.11
16:31:53 SQL> declare
16:32:05   2    vc_task   varchar2(100);
16:32:05   3    vc_sql    varchar2(1000);
16:32:05   4    vc_sql_mt varchar2(1000);
16:32:05   5    n_try     number;
16:32:05   6    n_status  number;
16:32:05   7  begin
16:32:05   8    --Define the Task
16:32:05   9    vc_task := 'Task 3: By SQL';
16:32:05  10    dbms_parallel_execute.create_task(task_name => vc_task);
16:32:05  11
16:32:05  12    --Define the Spilt
16:32:05  13    vc_sql_mt := 'select distinct object_id, object_id from t';
16:32:05  14    dbms_parallel_execute.create_chunks_by_SQL(task_name => vc_task,
16:32:05  15                                               sql_stmt  => vc_sql_mt,
16:32:05  16                                               by_rowid  => false);
```

```
16:32:05  17
16:32:05  18    vc_sql := 'update /*+ ROWID(dda) */t set DATA_OBJECT_ID=object_id+1 where object_id between :start_id and :end_id';
16:32:05  19    --Run the task
16:32:05  20    dbms_parallel_execute.run_task(task_name      => vc_task,
16:32:05  21                                   sql_stmt       => vc_sql,
16:32:05  22                                   language_flag  => dbms_sql.native,
16:32:05  23                                   parallel_level => 4);
16:32:05  24
16:32:05  25    --Controller
16:32:05  26    n_try    := 0;
16:32:05  27    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
16:32:05  28    while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
16:32:05  29      dbms_parallel_execute.resume_task(task_name => vc_task);
16:32:05  30      n_status := dbms_parallel_execute.task_status(task_name => vc_task);
16:32:05  31    end loop;
16:32:05  32
16:32:05  33    --Deal with Result
16:32:05  34    dbms_parallel_execute.drop_task(task_name => vc_task);
16:32:05  35  end;
16:32:05  36  /

^Cdeclare
*
ERROR at line 1:
ORA-01013: user requested cancel of current operation
ORA-06512: at "SYS.DBMS_PARALLEL_EXECUTE_INTERNAL", line 634
ORA-06512: at "SYS.DBMS_PARALLEL_EXECUTE", line 163
ORA-06512: at line 14


Elapsed: 00:01:09.08

16:33:14 SQL>  EXEC   dbms_parallel_execute.drop_task(task_name => 'Task 3: By SQL');

PL/SQL procedure successfully completed.
```

1.4.2.1 相关字典视图查询

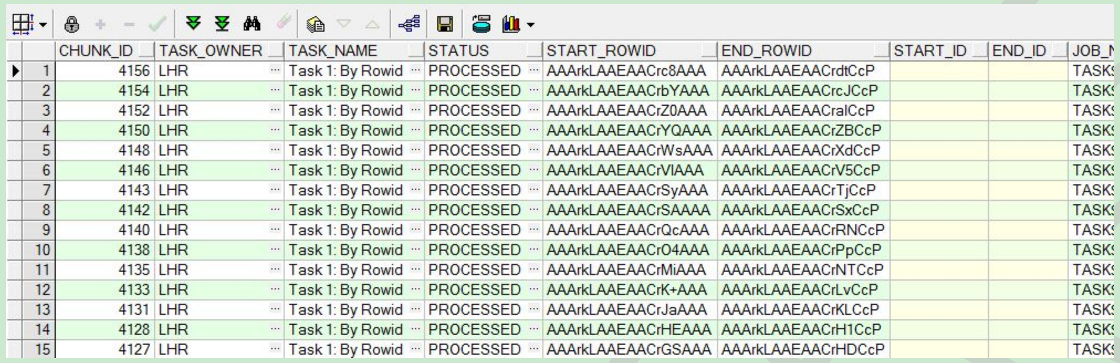
一、 create_chunks_by_rowid 过程

```
SELECT * FROM DBA_PARALLEL_EXECUTE_TASKS;
```



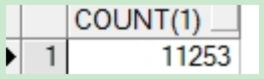
Row 1	Fields
TASK_OWNER	LHR
TASK_NAME	Task 1: By Rowid
CHUNK_TYPE	ROWID_RANGE
STATUS	PROCESSING
TABLE_OWNER	LHR
TABLE_NAME	T
NUMBER_COLUMN	
TASK_COMMENT	
JOB_PREFIX	TASK$_3177
SQL_STMT	<CLOB>
LANGUAGE_FLAG	1
EDITION	ORA$BASE
APPLY_CROSSEDITION_TRIGGER	
FIRE_APPLY_TRIGGER	TRUE
PARALLEL_LEVEL	4
JOB_CLASS	DEFAULT_JOB_CLAS

```
SELECT * FROM DBA_PARALLEL_EXECUTE_CHUNKS;
```



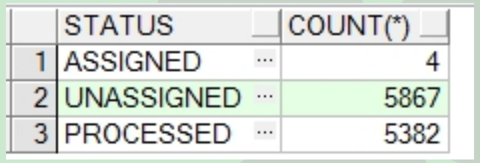
	CHUNK_ID	TASK_OWNER	TASK_NAME	STATUS	START_ROWID	END_ROWID	START_ID	END_ID	JOB_N
1	4156	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrc8AAA	AAArkLAAEAACrdtCcP			TASK
2	4154	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrbYAAA	AAArkLAAEAACrcJCcP			TASK
3	4152	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrZ0AAA	AAArkLAAEAACralCcP			TASK
4	4150	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrYQAAA	AAArkLAAEAACrZBCcP			TASK
5	4148	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrWsAAA	AAArkLAAEAACrXdCcP			TASK
6	4146	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrVIAAA	AAArkLAAEAACrV5CcP			TASK
7	4143	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrSyAAA	AAArkLAAEAACrTjCcP			TASK
8	4142	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrSAAAA	AAArkLAAEAACrSxCcP			TASK
9	4140	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrQcAAA	AAArkLAAEAACrRNCcP			TASK
10	4138	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrO4AAA	AAArkLAAEAACrPpCcP			TASK
11	4135	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrMiAAA	AAArkLAAEAACrNTCcP			TASK
12	4133	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrK+AAA	AAArkLAAEAACrLvCcP			TASK
13	4131	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrJaAAA	AAArkLAAEAACrKLCcP			TASK
14	4128	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrHEAAA	AAArkLAAEAACrH1CcP			TASK
15	4127	LHR	Task 1: By Rowid	PROCESSED	AAArkLAAEAACrGSAAA	AAArkLAAEAACrHDCcP			TASK

```
SELECT count(1) FROM DBA_PARALLEL_EXECUTE_CHUNKS;
```



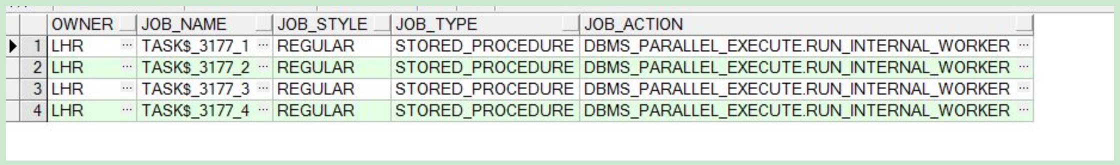
	COUNT(1)
1	11253

```
 select status, count(*) from user_parallel_execute_chunks group by status;
```



	STATUS	COUNT(*)
1	ASSIGNED	4
2	UNASSIGNED	5867
3	PROCESSED	5382

select D.owner,D.job_name,D.JOB_STYLE,D.JOB_TYPE,D.JOB_ACTION from dba_scheduler_jobs d where d.owner='LHR';



	OWNER	JOB_NAME	JOB_STYLE	JOB_TYPE	JOB_ACTION
1	LHR	TASK$_3177_1	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
2	LHR	TASK$_3177_2	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
3	LHR	TASK$_3177_3	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
4	LHR	TASK$_3177_4	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER

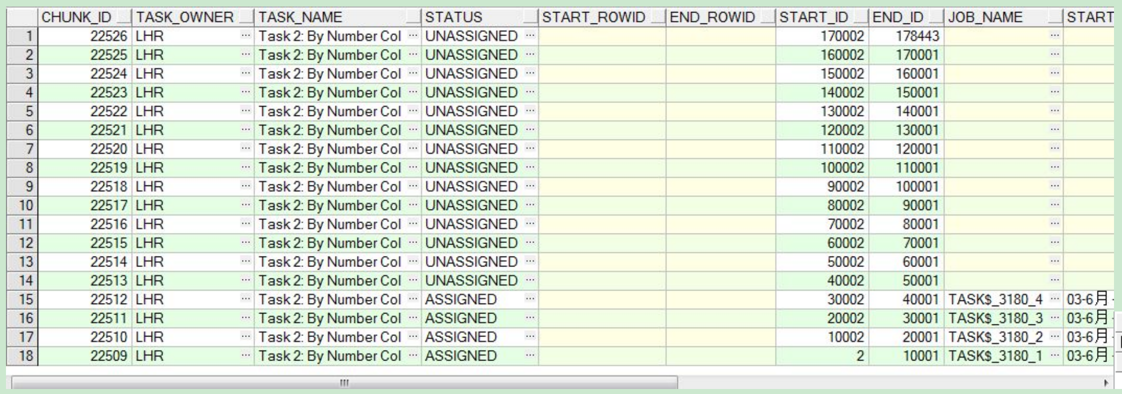
告警日志：

```
Wed Jun 03 15:53:48 2015
Archived Log entry 1202 added for thread 1 sequence 2669 ID 0x6779dfc4 dest 1:
Thread 1 advanced to log sequence 2671 (LGWR switch)
  Current log# 4 seq# 2671 mem# 0: /app/oracle/flash_recovery_area/CNYDB/onlinelog/o1_mf_4_bpxd8g7v_.log
Wed Jun 03 15:53:49 2015
Archived Log entry 1203 added for thread 1 sequence 2670 ID 0x6779dfc4 dest 1:
Wed Jun 03 15:53:57 2015
Thread 1 advanced to log sequence 2672 (LGWR switch)
  Current log# 5 seq# 2672 mem# 0: /app/oracle/flash_recovery_area/CNYDB/onlinelog/o1_mf_5_bpxdbwdz_.log
Wed Jun 03 15:53:58 2015
Archived Log entry 1204 added for thread 1 sequence 2671 ID 0x6779dfc4 dest 1:
Thread 1 advanced to log sequence 2673 (LGWR switch)
  Current log# 1 seq# 2673 mem# 0: /app/oracle/oradata/CNYDB/redo01.log
Wed Jun 03 15:54:04 2015
Archived Log entry 1205 added for thread 1 sequence 2672 ID 0x6779dfc4 dest 1:
Thread 1 advanced to log sequence 2674 (LGWR switch)
  Current log# 6 seq# 2674 mem# 0: /app/oracle/flash_recovery_area/CNYDB/onlinelog/o1_mf_6_bpxdcjx2_.log
Wed Jun 03 15:54:05 2015
Archived Log entry 1206 added for thread 1 sequence 2673 ID 0x6779dfc4 dest 1:
```

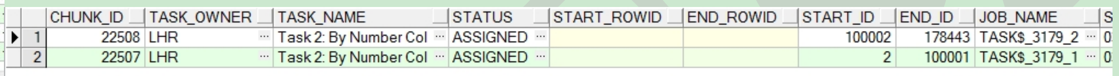
由告警日志可以看出 redo 切换非常迅速，归档来不及，所以还是需要在空闲的时候来做实验。

二、 create_chunks_by_number_col 过程

```
SELECT * FROM DBA_PARALLEL_EXECUTE_CHUNKS;
```

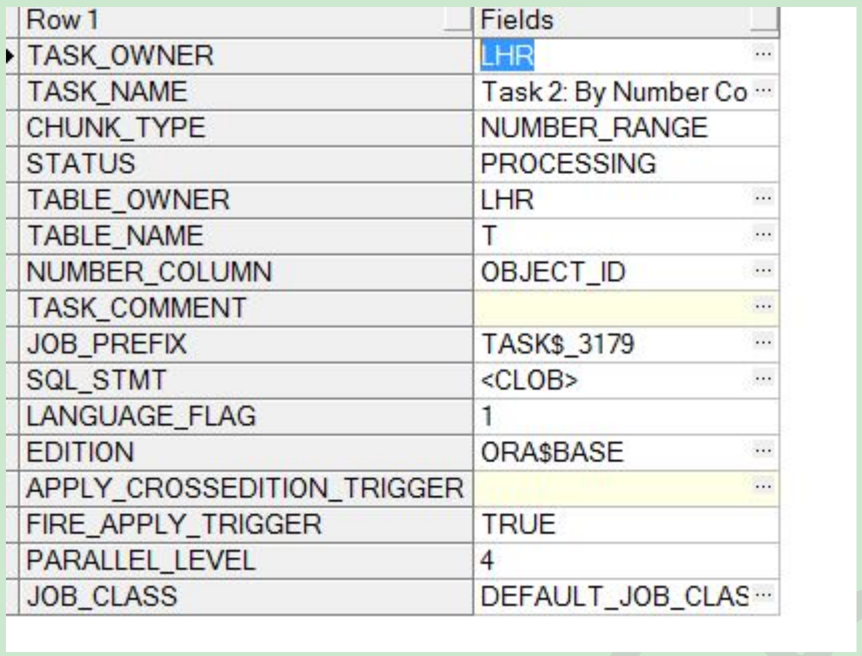


	CHUNK_ID	TASK_OWNER	TASK_NAME	STATUS	START_ROWID	END_ROWID	START_ID	END_ID	JOB_NAME	START
1	22526	LHR	Task 2: By Number Col	UNASSIGNED			170002	178443		
2	22525	LHR	Task 2: By Number Col	UNASSIGNED			160002	170001		
3	22524	LHR	Task 2: By Number Col	UNASSIGNED			150002	160001		
4	22523	LHR	Task 2: By Number Col	UNASSIGNED			140002	150001		
5	22522	LHR	Task 2: By Number Col	UNASSIGNED			130002	140001		
6	22521	LHR	Task 2: By Number Col	UNASSIGNED			120002	130001		
7	22520	LHR	Task 2: By Number Col	UNASSIGNED			110002	120001		
8	22519	LHR	Task 2: By Number Col	UNASSIGNED			100002	110001		
9	22518	LHR	Task 2: By Number Col	UNASSIGNED			90002	100001		
10	22517	LHR	Task 2: By Number Col	UNASSIGNED			80002	90001		
11	22516	LHR	Task 2: By Number Col	UNASSIGNED			70002	80001		
12	22515	LHR	Task 2: By Number Col	UNASSIGNED			60002	70001		
13	22514	LHR	Task 2: By Number Col	UNASSIGNED			50002	60001		
14	22513	LHR	Task 2: By Number Col	UNASSIGNED			40002	50001		
15	22512	LHR	Task 2: By Number Col	ASSIGNED			30002	40001	TASK$_3180_4	03-6月
16	22511	LHR	Task 2: By Number Col	ASSIGNED			20002	30001	TASK$_3180_3	03-6月
17	22510	LHR	Task 2: By Number Col	ASSIGNED			10002	20001	TASK$_3180_2	03-6月
18	22509	LHR	Task 2: By Number Col	ASSIGNED			2	10001	TASK$_3180_1	03-6月



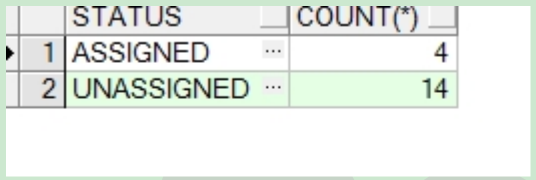
	CHUNK_ID	TASK_OWNER	TASK_NAME	STATUS	START_ROWID	END_ROWID	START_ID	END_ID	JOB_NAME	S
1	22508	LHR	Task 2: By Number Col	ASSIGNED			100002	178443	TASK$_3179_2	0
2	22507	LHR	Task 2: By Number Col	ASSIGNED			2	100001	TASK$_3179_1	0

```
SELECT * FROM DBA_PARALLEL_EXECUTE_TASKS;
```



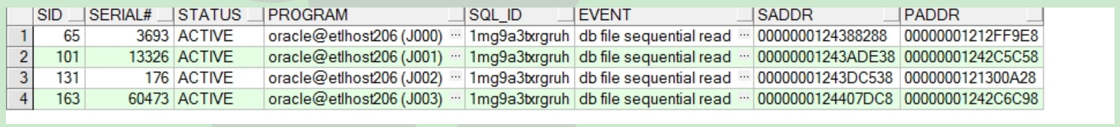
Row 1	Fields
TASK_OWNER	LHR
TASK_NAME	Task 2: By Number Co
CHUNK_TYPE	NUMBER_RANGE
STATUS	PROCESSING
TABLE_OWNER	LHR
TABLE_NAME	T
NUMBER_COLUMN	OBJECT_ID
TASK_COMMENT	
JOB_PREFIX	TASK$_3179
SQL_STMT	<CLOB>
LANGUAGE_FLAG	1
EDITION	ORA$BASE
APPLY_CROSSEDITION_TRIGGER	
FIRE_APPLY_TRIGGER	TRUE
PARALLEL_LEVEL	4
JOB_CLASS	DEFAULT_JOB_CLAS

```
select status, count(*) from dba_parallel_execute_chunks group by status;
```



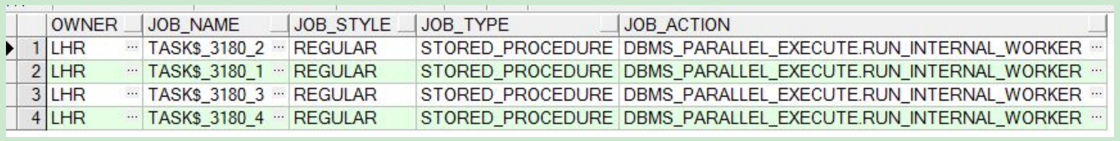
	STATUS	COUNT(*)
1	ASSIGNED	4
2	UNASSIGNED	14

```
select sid, serial#, status, PROGRAM, SQL_ID, event from v$session where action like 'TASK$%';
```



	SID	SERIAL#	STATUS	PROGRAM	SQL_ID	EVENT	SADDR	PADDR
1	65	3693	ACTIVE	oracle@etlhost206 (J000)	1mg9a3txrgruh	db file sequential read	0000000124388288	0000001212FF9E8
2	101	13326	ACTIVE	oracle@etlhost206 (J001)	1mg9a3txrgruh	db file sequential read	00000001243ADE38	0000001242C5C58
3	131	176	ACTIVE	oracle@etlhost206 (J002)	1mg9a3txrgruh	db file sequential read	00000001243DC538	000000121300A28
4	163	60473	ACTIVE	oracle@etlhost206 (J003)	1mg9a3txrgruh	db file sequential read	000000124407DC8	0000001242C6C98

select D.owner,D.job_name,D.JOB_STYLE,D.JOB_TYPE,D.JOB_ACTION from dba_scheduler_jobs d where d.owner='LHR';



	OWNER	JOB_NAME	JOB_STYLE	JOB_TYPE	JOB_ACTION
1	LHR	TASK$_3180_2	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
2	LHR	TASK$_3180_1	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
3	LHR	TASK$_3180_3	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER
4	LHR	TASK$_3180_4	REGULAR	STORED_PROCEDURE	DBMS_PARALLEL_EXECUTE.RUN_INTERNAL_WORKER

1.4.3 实验总结

由实验可以看出，采用 dbms_parallel_execute.create_chunks_by_rowid 方法，4 千万的数据量大约 4G 大小的表更新完大约 4 分钟，这个速度还是可以的，另外 2 种方式更新下来速度太慢就没有测试了，具体可以参考这里：http://blog.itpub.net/26736162/viewspace-1683912/，http://blog.itpub.net/26736162/viewspace-1683913/。

1.4.4 实验脚本

1.4.4.1 create_chunks_by_rowid 方式

```
declare
  vc_task  varchar2(100);
  vc_sql   varchar2(1000);
  n_try    number;
  n_status number;
begin
  --Define the Task

  vc_task := 'Task 1: By Rowid'; --Task名称

  dbms_parallel_execute.create_task(task_name => vc_task); --手工定义一个Task任务；

  --Define the Spilt
  dbms_parallel_execute.create_chunks_by_rowid(task_name   => vc_task,
                                               table_owner => 'LHR',
                                                      table_name  => 'T',
                                                      by_row      => true,
                                                      chunk_size  => 10000); --定义Chunk

  vc_sql := 'update /*+ ROWID(dda) */ t set DATA_OBJECT_ID=object_id+1 where rowid between :start_id and :end_id';
  --Run the task
  dbms_parallel_execute.run_task(task_name      => vc_task,
                                 sql_stmt       => vc_sql,
                                 language_flag  => dbms_sql.native,
                                 parallel_level => 4); --执行任务，确定并行度

  --Controller
  n_try    := 0;
  n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
    dbms_parallel_execute.resume_task(task_name => vc_task);
    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  end loop;

  --Deal with Result
  dbms_parallel_execute.drop_task(task_name => vc_task);
end;
/
```

1.4.4.2 create_chunks_by_number_col

```
declare
  vc_task  varchar2(100);
  vc_sql   varchar2(1000);
  n_try    number;
  n_status number;
begin
  --Define the Task
  vc_task := 'Task 2: By Number Col';
  dbms_parallel_execute.create_task(task_name => vc_task);

  --Define the Spilt
  dbms_parallel_execute.create_chunks_by_number_col(task_name    => vc_task,
                                                    table_owner  => 'LHR',
                                                    table_name   => 'T',
                                                    table_column => 'OBJECT_ID',
                                                    chunk_size   => 10000); --定义chunk
  vc_sql := 'update /*+ ROWID(dda) */ t set DATA_OBJECT_ID=object_id+1 where object_id between :start_id and :end_id';
  --Run the task
  dbms_parallel_execute.run_task(task_name     => vc_task,
                                 sql_stmt      => vc_sql,
                                 language_flag  => dbms_sql.native,
                                 parallel_level => 4);

  --Controller
  n_try    := 0;
  n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
    dbms_parallel_execute.resume_task(task_name => vc_task);
    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  end loop;

  --Deal with Result
  dbms_parallel_execute.drop_task(task_name => vc_task);
end;
/
```

1.4.4.3 create_chunks_by_SQL

```
declare
  vc_task   varchar2(100);
  vc_sql    varchar2(1000);
  vc_sql_mt varchar2(1000);
  n_try     number;
  n_status  number;
begin
  --Define the Task
  vc_task := 'Task 3: By SQL';
  dbms_parallel_execute.create_task(task_name => vc_task);

  --Define the Spilt
  vc_sql_mt := 'select distinct object_id, object_id from t';
  dbms_parallel_execute.create_chunks_by_SQL(task_name => vc_task,
                                             sql_stmt  => vc_sql_mt,
                                             by_rowid  => false);

  vc_sql  := 'update /*+ ROWID(dda) */t set DATA_OBJECT_ID=object_id+1 where object_id between :start_id and :end_id';
  --Run the task
  dbms_parallel_execute.run_task(task_name     => vc_task,
                                 sql_stmt      => vc_sql,
                                 language_flag  => dbms_sql.native,
                                 parallel_level => 4);

  --Controller
  n_try    := 0;
  n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  while (n_try < 2 and n_status != dbms_parallel_execute.FINISHED) loop
    dbms_parallel_execute.resume_task(task_name => vc_task);
    n_status := dbms_parallel_execute.task_status(task_name => vc_task);
  end loop;

  --Deal with Result
  dbms_parallel_execute.drop_task(task_name => vc_task);
end;
/
```

1.5 About Me

....

本文作者：小麦苗，只专注于数据库的技术，更注重技术的运用

ITPUB BLOG：http://blog.itpub.net/26736162

本文地址：

本文pdf版：http://yunpan.cn/QCwUAI9bn7g7w　提取码：af2d

QQ：642808185 若加 QQ 请注明你所正在读的文章标题

创作时间地点：2015-06-03 10:00~ 2015-06-03 18:00 于外汇交易中心

....