

Report

Laboratory Work 10

Dmitry Ladutsko

July 28, 2022

1. Prerequisites

1.1. Passwords Index

Password Group	Login Name	Password
Operation System	root	"rootadmin"
	oracle	"oracleadmin"
Oracle System	sys	"sysadmin"
	system	"sysadmin"
Oracle Users	All DB users	"%PWD%"

1.2. Folder Paths Index

Path Group	Path Description	Path
Operation System	Oracle RDBMS – BIN	/oracle/app/oracle
	Oracle Inventory	/oracle/app/oraInventory
	Oracle Database Storage	/oracle/oradata
	Oracle Install Directory	/oracle/install
Oracle	ORACLE_BASE	/oracle/app/oracle
	ORACLE_HOME	\$ORACLE_BASE/product/11.2
FTP	ftp Incoming Folder	/ftp/incoming

2. Oracle Architecture - Parallel execution

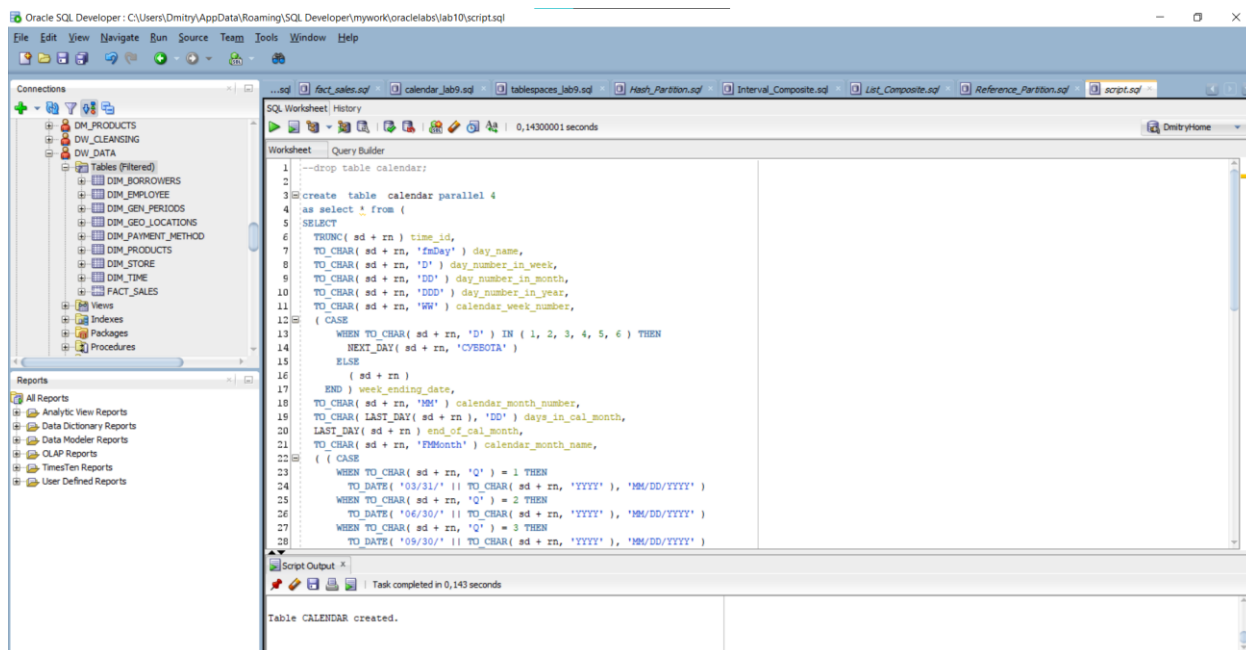
2.1. Task 01: CREATE Example of Select Parallel execution

The Main Task is to creating example of Select Parallel execution.

Task Results:

Create document that will store all screenshot about Select Parallel execution

- Scripts
- Execution Plan
- Summarize table – Compare time of the same operations



Picture 1 - Creating calendar table

	TIME_ID	DAY_NAME	DAY_NUMBER_IN_WEEK	DAY_NUMBER_IN_MONTH	DAY_NUMBER_IN_YEAR	CALENDAR_WEEK_NUMBER	WEEK_ENDING_DATE	CALENDAR_MONTH_NUMBER	DAYS_IN_CAL_MONTH	END
1	01.01.03	Среда	3	01	001	01	04.01.03	01	31	31.12.02
2	02.01.03	Четверг	4	02	002	01	04.01.03	01	31	31.12.02
3	03.01.03	Пятница	5	03	003	01	04.01.03	01	31	31.12.02
4	04.01.03	Суббота	6	04	004	01	11.01.03	01	31	31.12.02
5	05.01.03	Воскресенье	7	05	005	01	05.01.03	01	31	31.12.02
6	06.01.03	Понедельник	1	06	006	01	11.01.03	01	31	31.12.02
7	07.01.03	Вторник	2	07	007	01	11.01.03	01	31	31.12.02
8	08.01.03	Среда	3	08	008	02	11.01.03	01	31	31.12.02
9	09.01.03	Четверг	4	09	009	02	11.01.03	01	31	31.12.02
10	10.01.03	Пятница	5	10	010	02	11.01.03	01	31	31.12.02
11	11.01.03	Суббота	6	11	011	02	18.01.03	01	31	31.12.02
12	12.01.03	Воскресенье	7	12	012	02	12.01.03	01	31	31.12.02
13	13.01.03	Понедельник	1	13	013	02	18.01.03	01	31	31.12.02
14	14.01.03	Вторник	2	14	014	02	18.01.03	01	31	31.12.02
15	15.01.03	Среда	3	15	015	03	18.01.03	01	31	31.12.02
16	16.01.03	Четверг	4	16	016	03	18.01.03	01	31	31.12.02
17	17.01.03	Пятница	5	17	017	03	18.01.03	01	31	31.12.02
18	18.01.03	Суббота	6	18	018	03	25.01.03	01	31	31.12.02
19	19.01.03	Воскресенье	7	19	019	03	19.01.03	01	31	31.12.02
20	20.01.03	Понедельник	1	20	020	03	25.01.03	01	31	31.12.02
21	21.01.03	Вторник	2	21	021	03	25.01.03	01	31	31.12.02
22	22.01.03	Среда	3	22	022	04	25.01.03	01	31	31.12.02

Picture 2 - Select from calendar

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DISTRIBUTION	OBJECT_NODE	OTHER_TAG
SELECT STATEMENT				200	2		
PX COORDINATOR							
PX SEND	SYS.ITQ10000	QC (RANDOM)	200		2QC (RANDOM)	:Q1000	P->S
PX BLOCK		ITERATOR	200		2	:Q1000	PCWC
TABLE ACCESS	DW_DATA.CALENDAR	FULL		200	2	:Q1000	PCWP

Picture 3 - Explain plan of select

TIME_ID	DAY_NAME	DAY_NUMBER_IN_WEEK	DAY_NUMBER_IN_MONTH	DAY_NUMBER_IN_YEAR	CALENDAR_WEEK_NUMBER	WEEK_ENDING_DATE	CALENDAR_MONTH_NUMBER	DAYS_IN_CAL_MONTH	EN
4	04.01.03 Суббота	6	04	004	01	11.01.03	01	31	31.
5	05.01.03 Воскресенье	7	05	005	01	05.01.03	01	31	31.
6	06.01.03 Понедельник	1	06	006	01	11.01.03	01	31	31.
7	07.01.03 Вторник	2	07	007	01	11.01.03	01	31	31.
8	08.01.03 Среда	3	08	008	02	11.01.03	01	31	31.
9	09.01.03 Четверг	4	09	009	02	11.01.03	01	31	31.
10	10.01.03 Пятница	5	10	010	02	11.01.03	01	31	31.
11	11.01.03 Суббота	6	11	011	02	18.01.03	01	31	31.
12	12.01.03 Воскресенье	7	12	012	02	12.01.03	01	31	31.
13	13.01.03 Понедельник	1	13	013	02	18.01.03	01	31	31.

Picture 4 - Select with hint of parallel execution

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST	DISTRIBUTION	OBJECT_NODE	OTHER_TAG
SELECT STATEMENT				200	2		
PX COORDINATOR							
PX SEND	SYS.ITQ10000	QC (RANDOM)		200	2 QC (RANDOM)	:Q1000	P->S
PX BLOCK		ITERATOR		200	2	:Q1000	PCWC
TABLE ACCESS	DW_DATA.CALENDAR	FULL		200	2	:Q1000	PCWP

Picture 5 - Explain plan of select with hint of parallel execution

SQL	Connection	TimeStamp	Type	Executed	Duration(se...)
SELECT /*+ parallel(calendar, 4) */ * FROM calendar;	DmitryHome	29.07.22 02...	SQL	2	0.048
SELECT * FROM calendar;	DmitryHome	29.07.22 02...	SQL	2	0.137

Picture 6 - Comparing selects

If we compare two variants of selects, we can see the principal difference.

2.2. Task 02: CREATE Example of Parallel DML

The Main Task is to creating example of Parallel DML

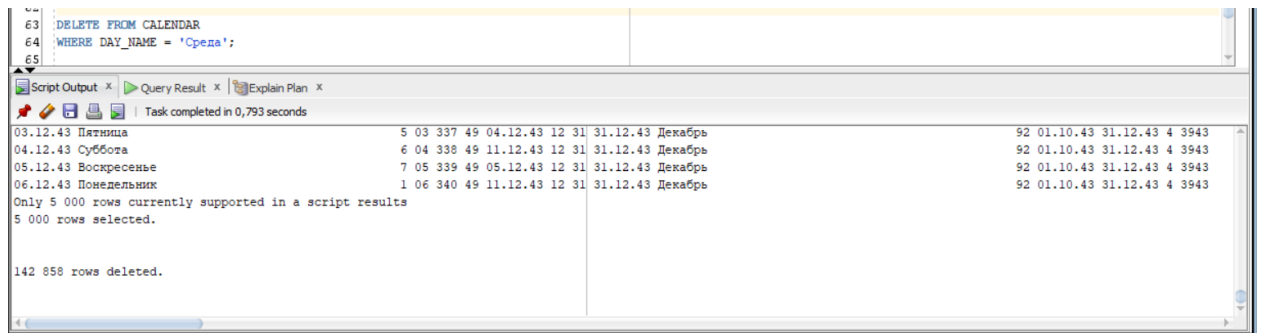
Task Results:

Create document that will store all screenshot about Parallel DML

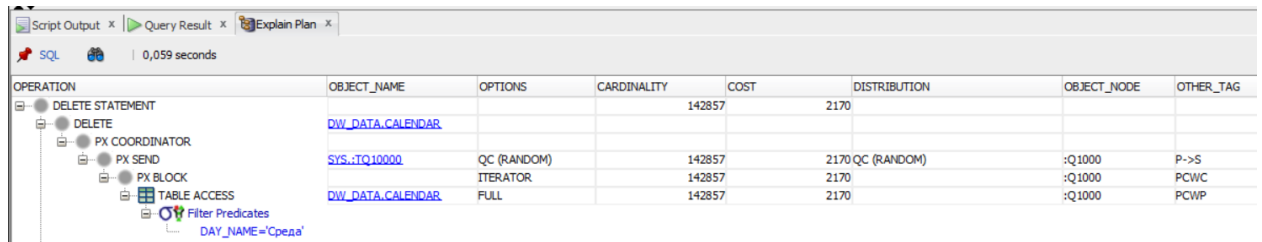
- Scripts
- Execution Plan
- Summarize table – Compare time of the same operations

Let us now delete data w\o parallelization:

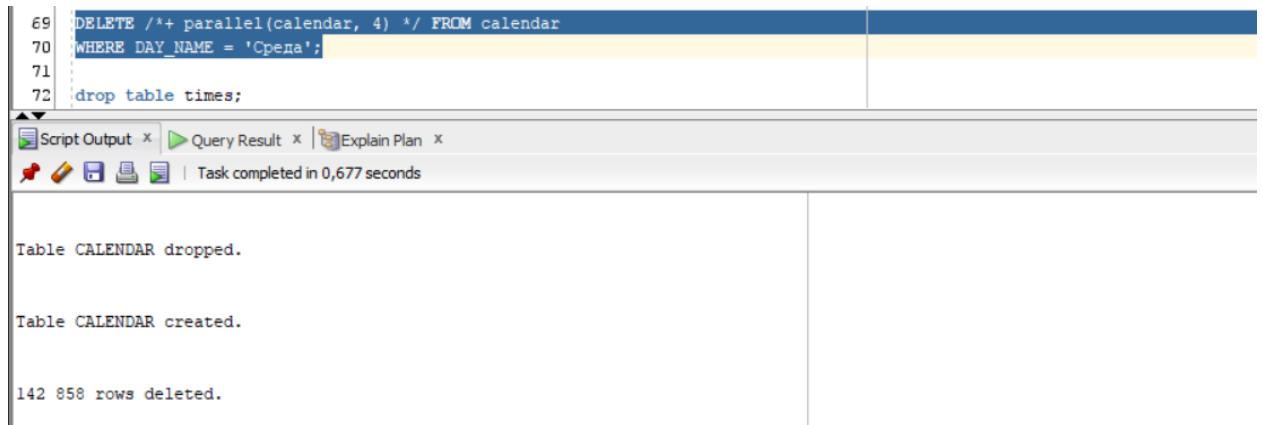
P.S. I added 1000000 rows to better see differences



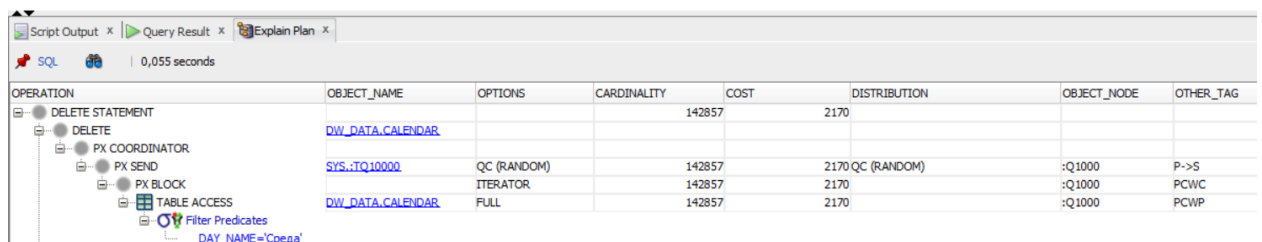
Picture 7 - Deleting Data



Picture 8 - Explain plan



Picture 9 - Deleting with parallelization



Picture 10 - Explain plan

	Simple DELETE	DELETE with PARALLEL execution
Duration	0.219	0.268

2.3. Task 03: CREATE Example of Parallel DDL

The Main Task is to creating example of Parallel DDL

Task Results:

Create document that will store all screenshot about Parallel DDL

- Scripts
- Execution Plan
- Summarize table – Compare time of the same operations

The screenshot shows a SQL Worksheet with a query to create a table named 'calendar'. The query uses a SELECT statement with various date and time functions to populate the table. The execution results show that the table was created successfully, with a task completion time of 23,827 seconds. The results also show recursive CPU usage, session logical reads, user I/O wait time, and user calls.

```
--drop table calendar;
create table calendar
as select * from (
SELECT
TRUNC( sd + rn ) time_id,
TO_CHAR( sd + rn, 'fmDay' ) day_name,
TO_CHAR( sd + rn, 'D' ) day_number_in_week,
TO_CHAR( sd + rn, 'DD' ) day_number_in_month,
TO_CHAR( sd + rn, 'DDD' ) day_number_in_year,
TO_CHAR( sd + rn, 'WW' ) calendar_week_number,
( CASE
WHEN TO_CHAR( sd + rn, 'D' ) IN ( 1, 2, 3, 4, 5, 6 ) THEN
NEXT_DAY( sd + rn, 'CYBEBOTA' )
ELSE
( sd + rn )
END ) week_ending_date,
TO_CHAR( sd + rn, 'MM' ) calendar_month_number,
TO_CHAR( LAST_DAY( sd + rn ), 'DD' ) days_in_cal_month,
LAST_DAY( sd + rn ) end_of_cal_month,
TO_CHAR( sd + rn, 'FMMonth' ) calendar_month_name
)
```

Task completed in 23,827 seconds

21 recursive cpu usage
149971 session logical reads
4 user I/O wait time
20 user calls

Table CALENDAR dropped.

Table CALENDAR created.

Picture 11 - Creating table w/o parallelization

The screenshot shows the execution plan for a query. The plan includes a SELECT STATEMENT, a VIEW, a COUNT, a CONNECT BY, a Filter Predicates, and a FAST DUAL. The execution plan shows that the query is executed in a single step, with a cardinality of 1 and a cost of 3.

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	3
VIEW	DW_DATA.null		1	3
COUNT				
CONNECT BY		WITHOUT FILTERING		
Filter Predicates				
FAST DUAL			1	3

Picture 12 - Explain plan

SQL Worksheet | History

23,56200027 seconds

Worksheet | Query Builder

```

1  --drop table calendar;
2
3  create table calendar
4  as select /*+parallel(4)*/ * from (
5  SELECT
6      TRUNC( sd + rn ) time_id,
7      TO_CHAR( sd + rn, 'fmDay' ) day_name,
8      TO_CHAR( sd + rn, 'D' ) day_number_in_week,
9      TO_CHAR( sd + rn, 'DD' ) day_number_in_month,
10     TO_CHAR( sd + rn, 'DDD' ) day_number_in_year,
11     TO_CHAR( sd + rn, 'WW' ) calendar_week_number,
12     ( CASE
13         WHEN TO_CHAR( sd + rn, 'D' ) IN ( 1, 2, 3, 4, 5, 6 ) THEN
14             NEXT_DAY( sd + rn, 'CYBOTA' )
15         ELSE
16             ( sd + rn )
17         END ) week_ending_date,
18     TO_CHAR( sd + rn, 'MM' ) calendar_month_number,
19     TO_CHAR( LAST_DAY( sd + rn ), 'DD' ) days_in_cal_month,
20     LAST_DAY( sd + rn ) end_of_cal_month,
21     TO_CHAR( sd + rn, 'FMMonth' ) calendar_month_name

```

Script Output x | Query Result x | Explain Plan x

Task completed in 23,562 seconds

Session altered.

Table CALENDAR dropped.

Table CALENDAR created.

Picture 13 - Creating table with hint of parallelization

Script Output x | Query Result x | Explain Plan x

0,059 seconds

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
CREATE TABLE STATEMENT				1 2
LOAD AS SELECT	DW_DATA.CALENDAR			
OPTIMIZER STATISTICS GATHERING				1 2
VIEW	DW_DATA.null			1 2
COUNT				
CONNECT BY				
Filter Predicates				
LEVEL <= 1000000				
FAST DUAL				1 2
		WITHOUT FILTERING		

Picture 14 - Explain plan

	Simple CREATE	CREATE with PARALLEL execution
Duration	23.826	23.562

Laboratory work summary:

Despite we use 100000 rows this amount is not that big to show such principal difference, so, the parallelization does not improve the results. To benefit from parallel query execution, we need to work with tremendous amount of data.

All diagrams and scripts are stored in GitHub (link in README file in Labs folder)