Case Study

Adjusting Schema to Application Needs

or

Beyond SQL Tuning – part 2

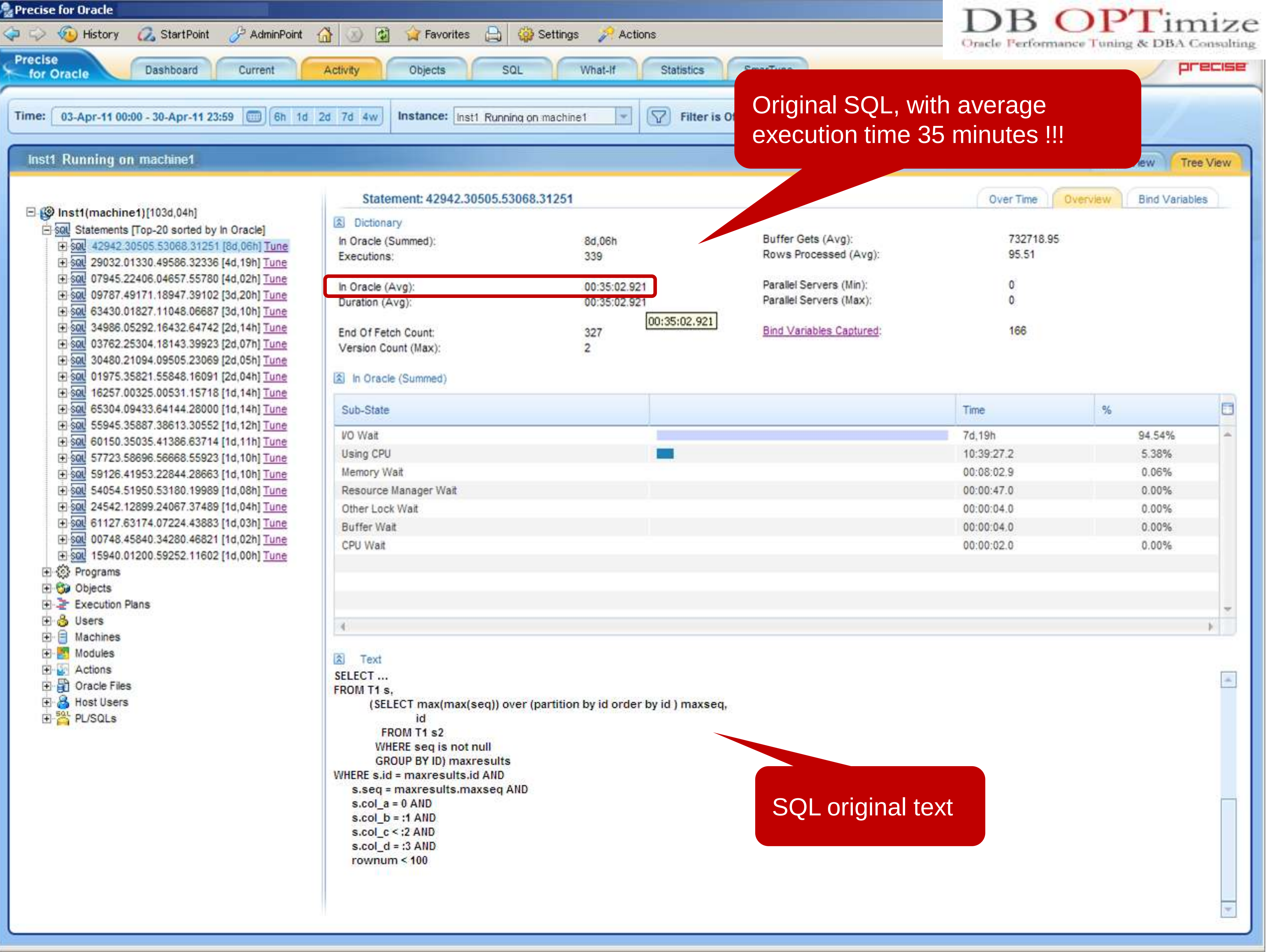
DB OPTimize
Oracle Performance Tuning & DBA Consulting

Application was constantly running a very heavy SQL hour after hour day after day. Each execution took **35 minutes** on average. Since each execution took so long, only 20-70 executions were able to run each day.

By adjusting schema to application needs and then rewriting this SQL, execution went down to **1 sec**, and SQL was able to executed 25k times a day.

Heavy overload on the instance as well as on machine and disks was stopped. SQL performance was improved dramatically, allowing huge increase in application capacity and productivity.

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Original SQL, with average execution time 35 minutes !!!

Time: 03-Apr-11 00:00 - 30-Apr-11 23:59 6h 1d 2d 7d 4w

Instance: Inst1 Running on machine1

Filter is Of

Inst1 Running on machine1

- Inst1(machine1)[103d,04h]
  - Statements [Top-20 sorted by In Oracle]
    - 42942.30505.53068.31251 [8d,06h] Tune
    - 29032.01330.49586.32336 [4d,19h] Tune
    - 07945.22406.04657.55780 [4d,02h] Tune
    - 09787.49171.18947.39102 [3d,20h] Tune
    - 63430.01827.11048.06687 [3d,10h] Tune
    - 34986.05292.16432.64742 [2d,14h] Tune
    - 03762.25304.18143.39923 [2d,07h] Tune
    - 30480.21094.09505.23069 [2d,05h] Tune
    - 01975.35821.55848.16091 [2d,04h] Tune
    - 16257.00325.00531.15718 [1d,14h] Tune
    - 65304.09433.64144.28000 [1d,14h] Tune
    - 55945.35887.38613.30552 [1d,12h] Tune
    - 60150.35035.41386.63714 [1d,11h] Tune
    - 57723.58696.56668.55923 [1d,10h] Tune
    - 59126.41953.22844.28663 [1d,10h] Tune
    - 54054.51950.53180.19989 [1d,08h] Tune
    - 24542.12899.24067.37489 [1d,04h] Tune
    - 61127.63174.07224.43883 [1d,03h] Tune
    - 00748.45840.34280.46821 [1d,02h] Tune
    - 15940.01200.59252.11602 [1d,00h] Tune
  - Programs
  - Objects
  - Execution Plans
  - Users
  - Machines
  - Modules
  - Actions
  - Oracle Files
  - Host Users
  - PL/SQLs

Statement: 42942.30505.53068.31251

Dictionary

In Oracle (Summed):	8d,06h	Buffer Gets (Avg):	732718.95
Executions:	339	Rows Processed (Avg):	95.51
In Oracle (Avg):	00:35:02.921	Parallel Servers (Min):	0
Duration (Avg):	00:35:02.921	Parallel Servers (Max):	0
End Of Fetch Count:	327	Bind Variables Captured:	166
Version Count (Max):	2		

00:35:02.921

In Oracle (Summed)

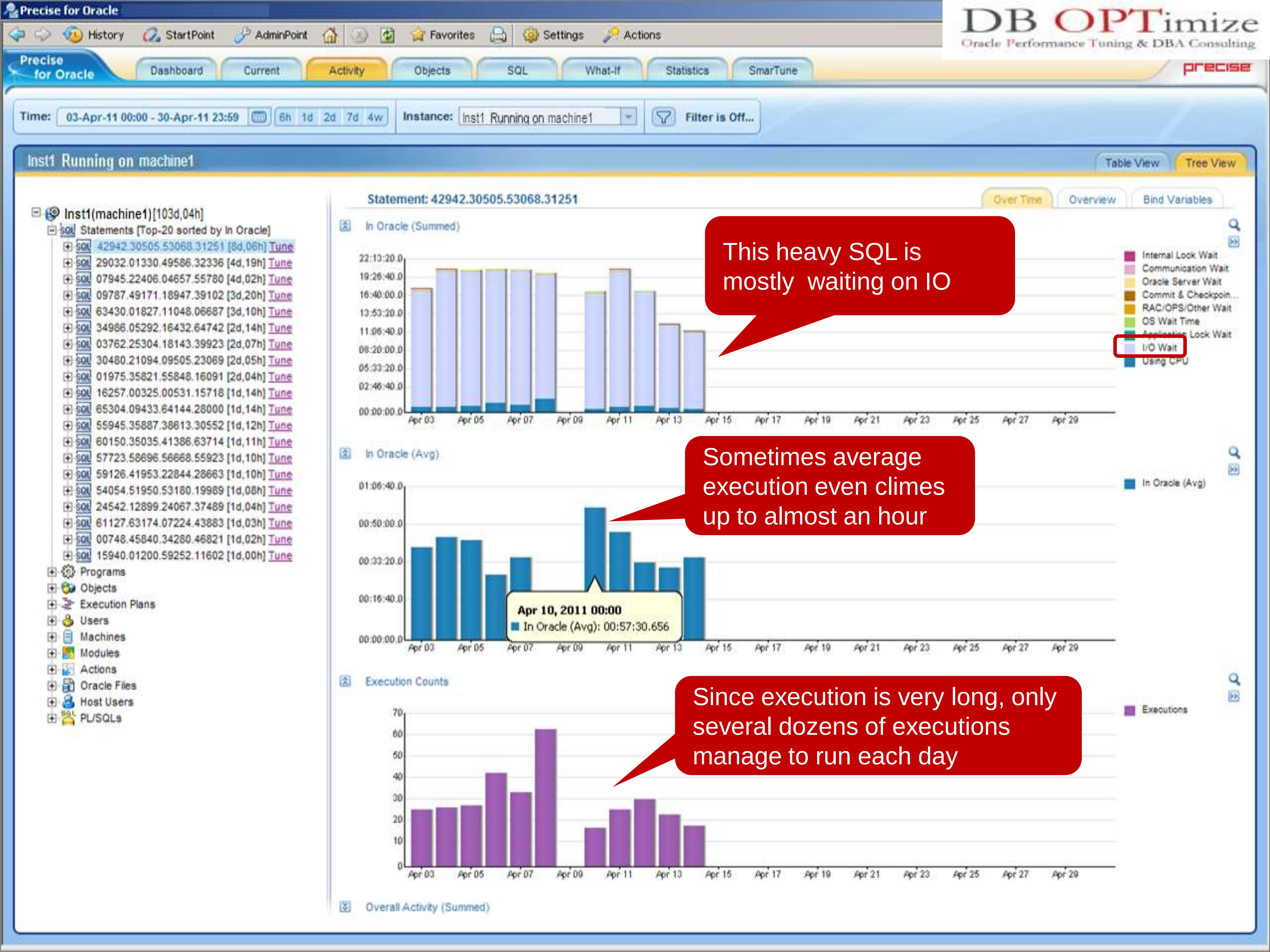
Sub-State		Time	%
I/O Wait		7d,19h	94.54%
Using CPU		10:39:27.2	5.38%
Memory Wait		00:08:02.9	0.06%
Resource Manager Wait		00:00:47.0	0.00%
Other Lock Wait		00:00:04.0	0.00%
Buffer Wait		00:00:04.0	0.00%
CPU Wait		00:00:02.0	0.00%

Text

```
SELECT ...
FROM T1 s,
        (SELECT max(max(seq)) over (partition by id order by id ) maxseq,
                    id
           FROM T1 s2
          WHERE seq is not null
          GROUP BY ID) maxresults
WHERE s.id = maxresults.id AND
     s.seq = maxresults.maxseq AND
     s.col_a = 0 AND
     s.col_b = :1 AND
     s.col_c < :2 AND
     s.col_d = :3 AND
     rownum < 100
```

SQL original text

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Inst1(machine1)[103d,04h]
- SQL Statements [Top-20 sorted by In Oracle]
  - SQL 42942.30505.53068.31251 [8d,06h] Tune
  - SQL 29032.01330.49586.32336 [4d,19h] Tune
  - SQL 07945.22406.04657.55780 [4d,02h] Tune
  - SQL 09787.49171.18947.39102 [3d,20h] Tune
  - SQL 63430.01827.11048.06687 [3d,10h] Tune
  - SQL 34986.05292.16432.64742 [2d,14h] Tune
  - SQL 03762.25304.18143.39923 [2d,07h] Tune
  - SQL 30480.21094.09505.23069 [2d,05h] Tune
  - SQL 01975.35821.55848.16091 [2d,04h] Tune
  - SQL 16257.00325.00531.15718 [1d,14h] Tune
  - SQL 65304.09433.64144.28000 [1d,14h] Tune
  - SQL 55945.35887.38613.30552 [1d,12h] Tune
  - SQL 60150.35035.41386.63714 [1d,11h] Tune
  - SQL 57723.58696.56668.55923 [1d,10h] Tune
  - SQL 59126.41953.22844.28663 [1d,10h] Tune
  - SQL 54054.51950.53180.19989 [1d,08h] Tune
  - SQL 24542.12899.24067.37489 [1d,04h] Tune
  - SQL 61127.63174.07224.43883 [1d,03h] Tune
  - SQL 00748.45840.34280.46821 [1d,02h] Tune
  - SQL 15940.01200.59252.11602 [1d,00h] Tune
- Programs
- Objects
- Execution Plans
- Users
- Machines
- Modules
- Actions
- Oracle Files
- Host Users
- PL/SQLs

Statement: 42942.30505.53068.31251

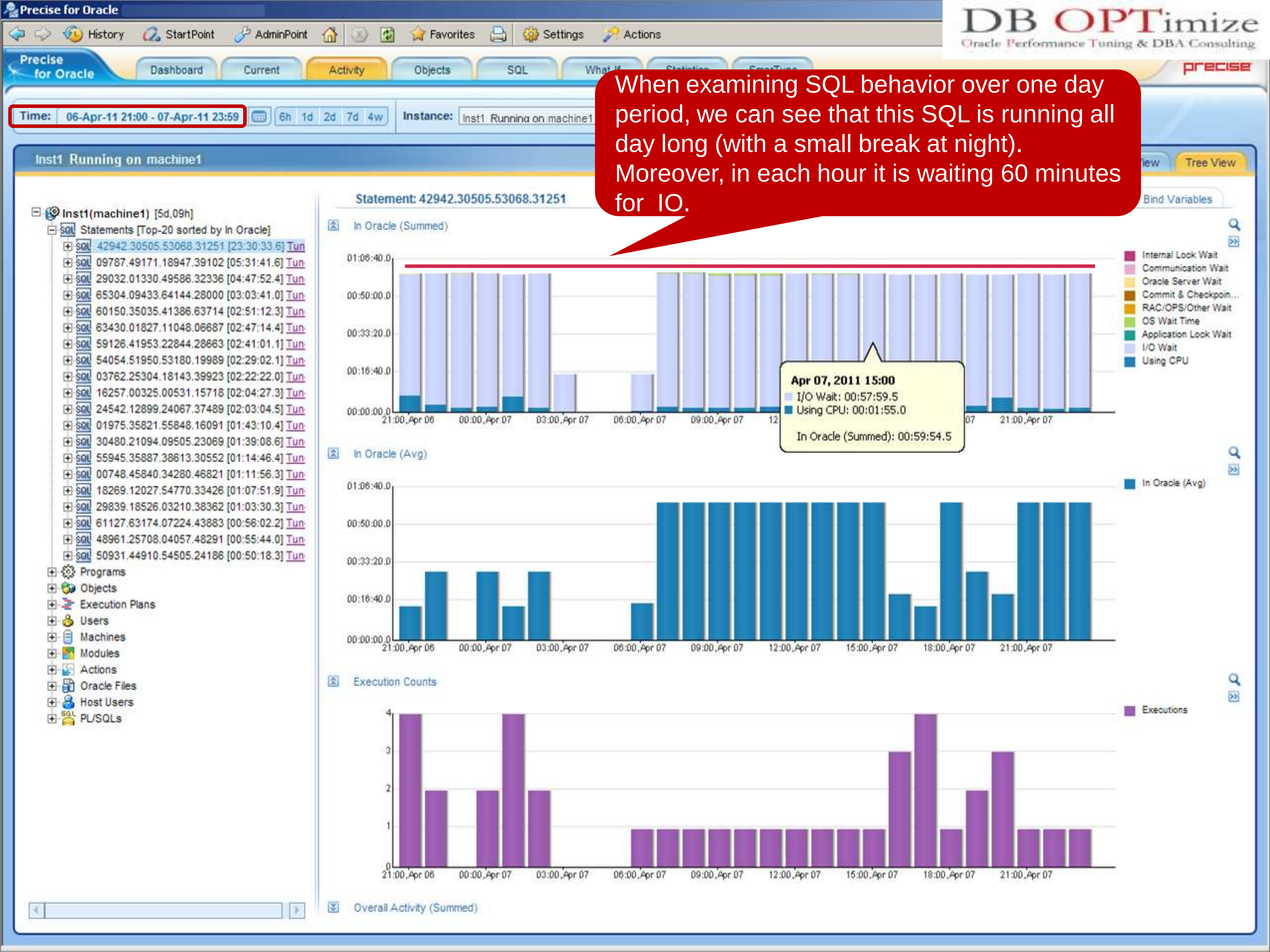
This heavy SQL is mostly waiting on IO

Sometimes average execution even climes up to almost an hour

Since execution is very long, only several dozens of executions manage to run each day

DB OPTimize
Oracle Performance Tuning & DBA Consulting

When examining SQL behavior over one day period, we can see that this SQL is running all day long (with a small break at night). Moreover, in each hour it is waiting 60 minutes for IO.



Time: 06-Apr-11 21:00 - 07-Apr-11 23:59

Instance: Inst1 Running on machine1

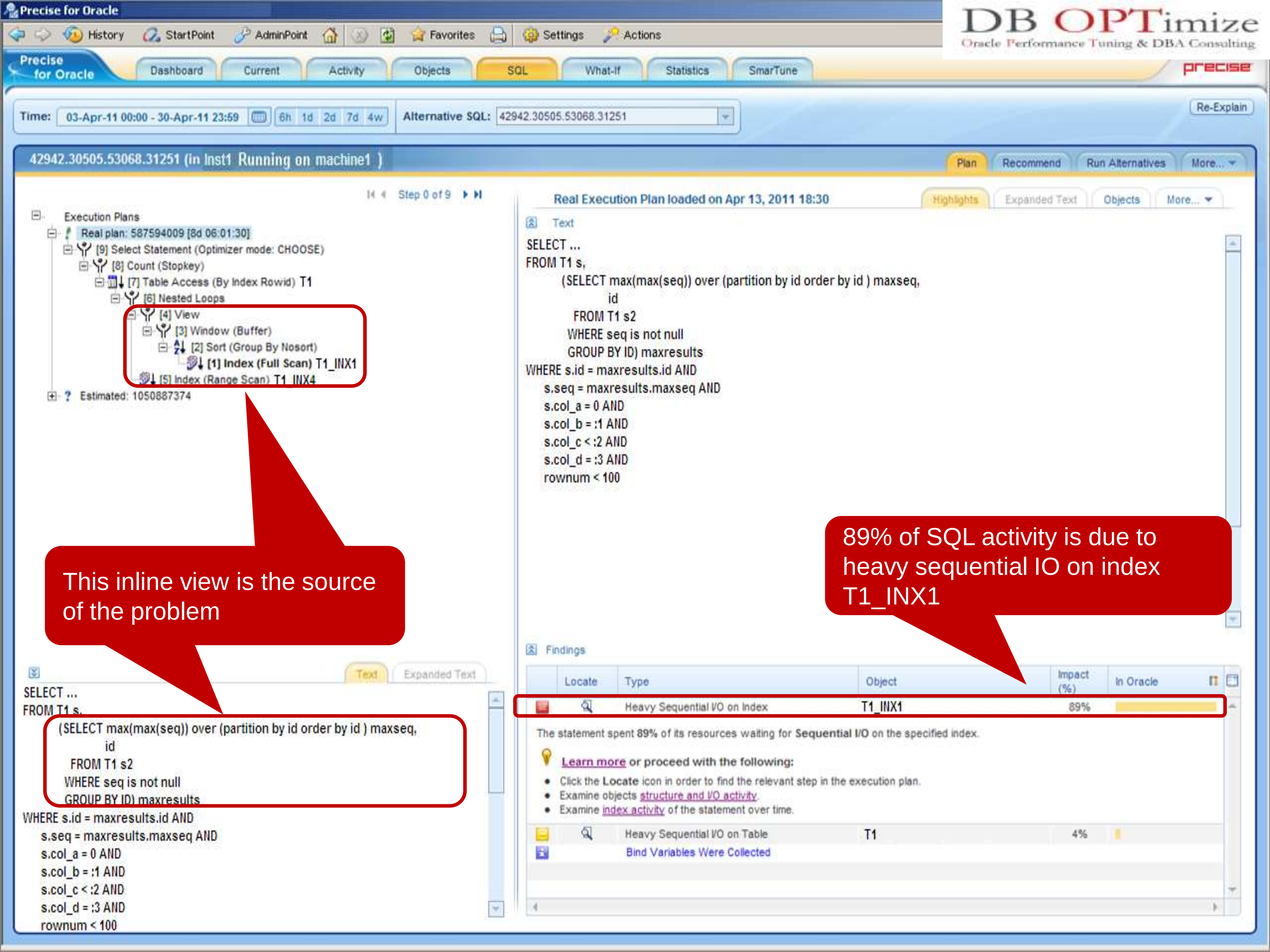
Inst1 Running on machine1

- Inst1(machine1) [5d,09h]
  - Statements [Top-20 sorted by In Oracle]
    - 42942.30505.53068.31251 [23:30:33.6] Tun
    - 09787.49171.18947.39102 [05:31:41.6] Tun
    - 29032.01330.49586.32336 [04:47:52.4] Tun
    - 65304.09433.64144.28000 [03:03:41.0] Tun
    - 60150.35035.41386.63714 [02:51:12.3] Tun
    - 63430.01827.11048.06687 [02:47:14.4] Tun
    - 59126.41953.22844.28663 [02:41:01.1] Tun
    - 54054.51950.53180.19989 [02:29:02.1] Tun
    - 03762.25304.18143.39923 [02:22:22.0] Tun
    - 16257.00325.00531.15718 [02:04:27.3] Tun
    - 24542.12899.24067.37489 [02:03:04.5] Tun
    - 01975.35821.55848.16091 [01:43:10.4] Tun
    - 30480.21094.09505.23069 [01:39:08.6] Tun
    - 55945.35887.38613.30552 [01:14:46.4] Tun
    - 00748.45840.34280.46821 [01:11:56.3] Tun
    - 18269.12027.54770.33426 [01:07:51.9] Tun
    - 29839.18526.03210.38362 [01:03:30.3] Tun
    - 61127.63174.07224.43883 [00:56:02.2] Tun
    - 48961.25708.04057.48291 [00:55:44.0] Tun
    - 50931.44910.54505.24186 [00:50:18.3] Tun
  - Programs
  - Objects
  - Execution Plans
  - Users
  - Machines
  - Modules
  - Actions
  - Oracle Files
  - Host Users
  - PL/SQLs

Statement: 42942.30505.53068.31251

Apr 07, 2011 15:00
I/O Wait: 00:57:59.5
Using CPU: 00:01:55.0
In Oracle (Summed): 00:59:54.5

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Time: 03-Apr-11 00:00 - 30-Apr-11 23:59 6h 1d 2d 7d 4w

Alternative SQL: 42942.30505.53068.31251

42942.30505.53068.31251 (in Inst1 Running on machine1)

- Execution Plans
  - Real plan: 587594009 [8d 06:01:30]
    - [9] Select Statement (Optimizer mode: CHOOSE)
      - [8] Count (Stopkey)
        - [7] Table Access (By Index Rowid) T1
          - [6] Nested Loops
            - [4] View
              - [3] Window (Buffer)
                - [2] Sort (Group By Nosort)
                  - **[1] Index (Full Scan) T1_INX1**
            - [5] Index (Range Scan) T1_INX4
  - Estimated: 1050887374

Real Execution Plan loaded on Apr 13, 2011 18:30

Text

```
SELECT ...
FROM T1 s,
       (SELECT max(max(seq)) over (partition by id order by id ) maxseq,
                   id
          FROM T1 s2
         WHERE seq is not null
         GROUP BY ID) maxresults
WHERE s.id = maxresults.id AND
    s.seq = maxresults.maxseq AND
    s.col_a = 0 AND
    s.col_b = :1 AND
    s.col_c < :2 AND
    s.col_d = :3 AND
    rownum < 100
```

Findings

	Locate	Type	Object	Impact (%)	In Oracle
🟥	🔍	Heavy Sequential I/O on Index	T1_INX1	89%	

The statement spent **89%** of its resources waiting for **Sequential I/O** on the specified index.

Learn more or proceed with the following:

- Click the **Locate** icon in order to find the relevant step in the execution plan.
- Examine objects structure and I/O activity.
- Examine index activity of the statement over time.

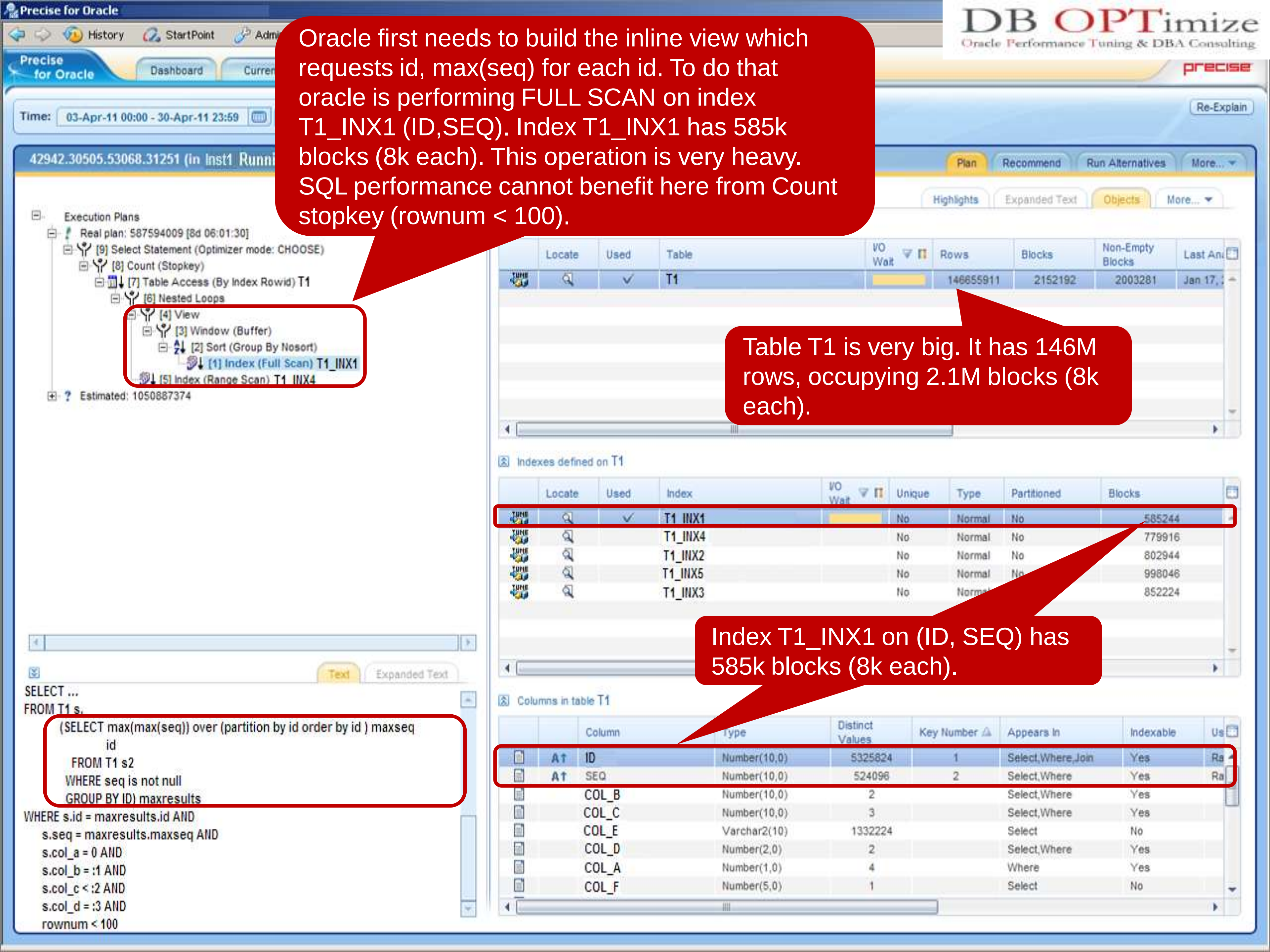
	Locate	Type	Object	Impact (%)	In Oracle
🟨	🔍	Heavy Sequential I/O on Table	T1	4%	
		Bind Variables Were Collected			

```
SELECT ...
FROM T1 s,
   (SELECT max(max(seq)) over (partition by id order by id ) maxseq,
              id
     FROM T1 s2
    WHERE seq is not null
    GROUP BY ID) maxresults
WHERE s.id = maxresults.id AND
    s.seq = maxresults.maxseq AND
    s.col_a = 0 AND
    s.col_b = :1 AND
    s.col_c < :2 AND
    s.col_d = :3 AND
    rownum < 100
```

This inline view is the source of the problem

89% of SQL activity is due to heavy sequential IO on index T1_INX1

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Time: 03-Apr-11 00:00 - 30-Apr-11 23:59

42942.30505.53068.31251 (in Inst1 Runni

Oracle first needs to build the inline view which requests id, max(seq) for each id. To do that oracle is performing FULL SCAN on index T1_INX1 (ID,SEQ). Index T1_INX1 has 585k blocks (8k each). This operation is very heavy. SQL performance cannot benefit here from Count stopkey (rownum < 100).

```
Execution Plans
  Real plan: 587594009 [8d 06:01:30]
    [9] Select Statement (Optimizer mode: CHOOSE)
      [8] Count (Stopkey)
        [7] Table Access (By Index Rowid) T1
          [6] Nested Loops
            [4] View
              [3] Window (Buffer)
                [2] Sort (Group By Nosort)
                  [1] Index (Full Scan) T1_INX1
            [5] Index (Range Scan) T1_INX4
  Estimated: 1050887374
```

	Locate	Used	Table	I/O Wait	Rows	Blocks	Non-Empty Blocks	Last An
TUNE	🔍	✓	T1	▬	146655911	2152192	2003281	Jan 17,

Table T1 is very big. It has 146M rows, occupying 2.1M blocks (8k each).

Indexes defined on T1

	Locate	Used	Index	I/O Wait	Unique	Type	Partitioned	Blocks
TUNE	🔍	✓	T1_INX1	▬	No	Normal	No	585244
TUNE	🔍		T1_INX4		No	Normal	No	779916
TUNE	🔍		T1_INX2		No	Normal	No	802944
TUNE	🔍		T1_INX5		No	Normal	No	998046
TUNE	🔍		T1_INX3		No	Norma		852224

Index T1_INX1 on (ID, SEQ) has 585k blocks (8k each).

Columns in table T1

		Column	Type	Distinct Values	Key Number	Appears In	Indexable	Us
	A↑	ID	Number(10,0)	5325824	1	Select,Where,Join	Yes	Ra
	A↑	SEQ	Number(10,0)	524096	2	Select,Where	Yes	Ra
		COL_B	Number(10,0)	2		Select,Where	Yes	
		COL_C	Number(10,0)	3		Select,Where	Yes	
		COL_E	Varchar2(10)	1332224		Select	No	
		COL_D	Number(2,0)	2		Select,Where	Yes	
		COL_A	Number(1,0)	4		Where	Yes	
		COL_F	Number(5,0)	1		Select	No	

```
SELECT ...
FROM T1 s,
    (SELECT max(max(seq)) over (partition by id order by id ) maxseq
           id
      FROM T1 s2
     WHERE seq is not null
     GROUP BY ID) maxresults
WHERE s.id = maxresults.id AND
    s.seq = maxresults.maxseq AND
    s.col_a = 0 AND
    s.col_b = :1 AND
    s.col_c < :2 AND
    s.col_d = :3 AND
    rownum < 100
```

After a short time examining this SQL I knew that tuning capabilities were very limited. I had to speak with the application to fully understand the logic behind it.

It appeared that every time application was doing something on id, a new row was inserted into table T1 with id and max(seq) + 1. I have also found that application was mostly interested in max(seq) for each id, but didn’t hold this value anywhere.

It was clear to me that the way schema was designed simply didn’t match application needs.

This is when I knew that schema design must be changed. There has to be a table holding max(seq) for a given id, and this change would surly lead to performance boost.

Therefore I have created a new table T1_MAXSEQ (ID number, SEQ number), with unique index on (ID,SEQ).

Now, something has to keep this table updated and fully match values in table T1.

I found that there was a before insert trigger on T1, responsible to set the correct value of SEQ to max(seq) + 1 for any given ID.

I have update this trigger to also insert a new row into T1_MAXSEQ when a new ID is inserted to T1, and update an existing row to SEQ+1 for an existing ID.

Schema change (new table and trigger update) was minimal and transparent to the application. All they needed to do was to rewrite the SQL so that it will now join T1 with T1_MAXSEQ instead of using inline view to find max(seq) for every IDs.

New SQL should look like that:

Original Text:

```
SELECT …
        FROM T1 s,
             (SELECT max(max(seq)) over (partition by id order by id ) maxseq,
             id
        FROM T1 s2
        WHERE seq is not null
        GROUP BY id) maxresults
      WHERE s.id = maxresults.id AND
        s.seq = maxresults.maxseq AND
        s.col_a = 0 AND
        s.col_b = :1 AND
        s.col_c < :2 AND
        s.col_d = :3 AND
        rownum < 100
```

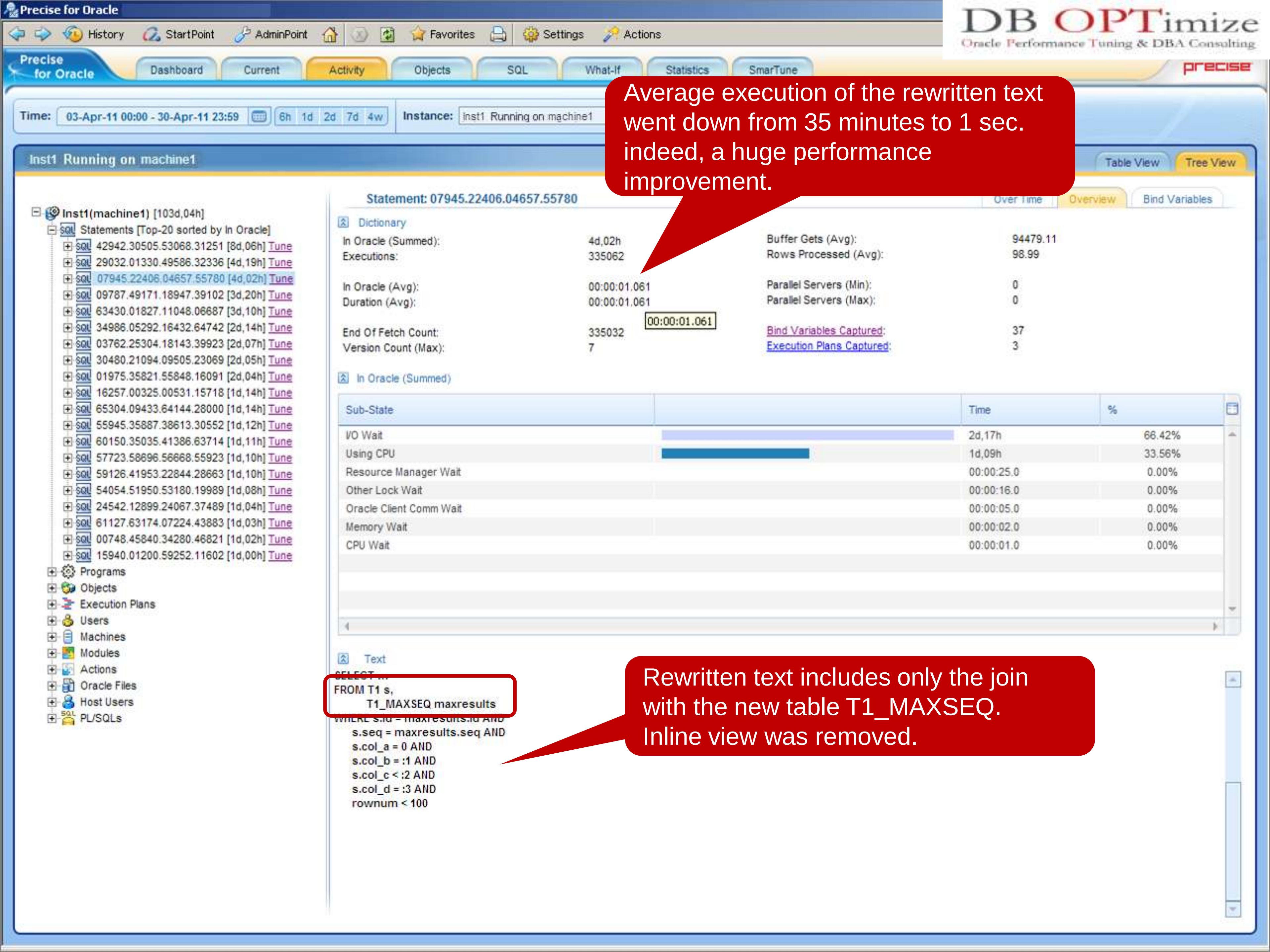
Changed Text:

Using a join with the new table T1_MAXSEQ instead of inline view with analytic function

```
SELECT …
    FROM T1 s,
         T1_MAXSEQ maxresults
    WHERE s.id = maxresults.id AND
         s.seq = maxresults.seq AND
         s.col_a = 0 AND
         s.col_b = :1 AND
         s.col_c < :2 AND
         s.col_d = :3 AND
         rownum < 100
```

Now, Let’s see how those changes have influenced on SQL performance and behavior.

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Average execution of the rewritten text went down from 35 minutes to 1 sec. indeed, a huge performance improvement.

Time: 03-Apr-11 00:00 - 30-Apr-11 23:59 6h 1d 2d 7d 4w Instance: Inst1 Running on machine1

Inst1 Running on machine1

- Inst1(machine1) [103d,04h]
  - Statements [Top-20 sorted by In Oracle]
    - 42942.30505.53068.31251 [8d,06h] Tune
    - 29032.01330.49586.32336 [4d,19h] Tune
    - 07945.22406.04657.55780 [4d,02h] Tune
    - 09787.49171.18947.39102 [3d,20h] Tune
    - 63430.01827.11048.06687 [3d,10h] Tune
    - 34986.05292.16432.64742 [2d,14h] Tune
    - 03762.25304.18143.39923 [2d,07h] Tune
    - 30480.21094.09505.23069 [2d,05h] Tune
    - 01975.35821.55848.16091 [2d,04h] Tune
    - 16257.00325.00531.15718 [1d,14h] Tune
    - 65304.09433.64144.28000 [1d,14h] Tune
    - 55945.35887.38613.30552 [1d,12h] Tune
    - 60150.35035.41386.63714 [1d,11h] Tune
    - 57723.58696.56668.55923 [1d,10h] Tune
    - 59126.41953.22844.28663 [1d,10h] Tune
    - 54054.51950.53180.19989 [1d,08h] Tune
    - 24542.12899.24067.37489 [1d,04h] Tune
    - 61127.63174.07224.43883 [1d,03h] Tune
    - 00748.45840.34280.46821 [1d,02h] Tune
    - 15940.01200.59252.11602 [1d,00h] Tune
  - Programs
  - Objects
  - Execution Plans
  - Users
  - Machines
  - Modules
  - Actions
  - Oracle Files
  - Host Users
  - PL/SQLs

Statement: 07945.22406.04657.55780

Dictionary

In Oracle (Summed): 4d,02h
Executions: 335062

In Oracle (Avg): 00:00:01.061
Duration (Avg): 00:00:01.061

00:00:01.061

End Of Fetch Count: 335032
Version Count (Max): 7

Buffer Gets (Avg): 94479.11
Rows Processed (Avg): 98.99

Parallel Servers (Min): 0
Parallel Servers (Max): 0

Bind Variables Captured: 37
Execution Plans Captured: 3

In Oracle (Summed)

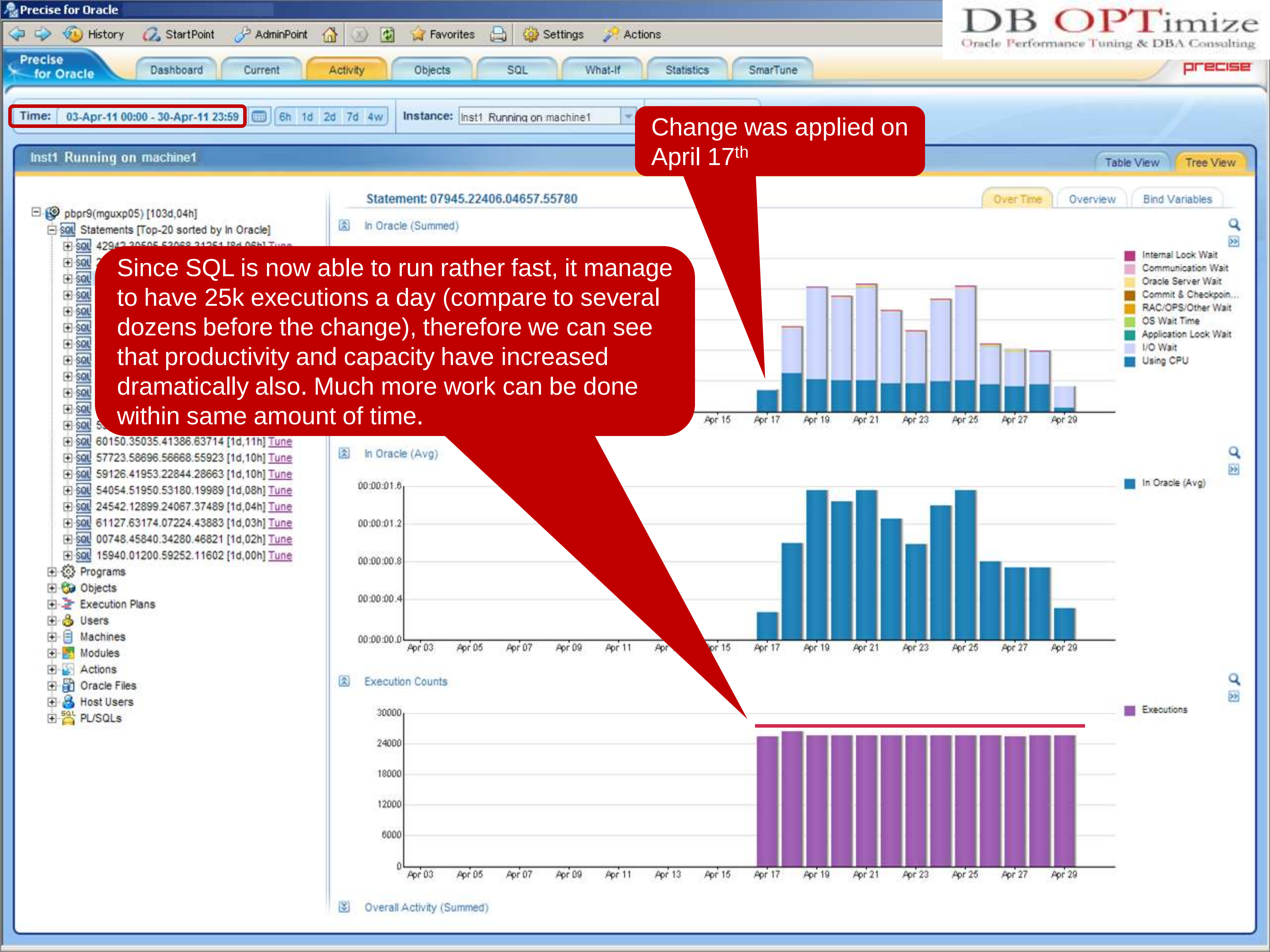
Sub-State		Time	%
I/O Wait		2d,17h	66.42%
Using CPU		1d,09h	33.56%
Resource Manager Wait		00:00:25.0	0.00%
Other Lock Wait		00:00:16.0	0.00%
Oracle Client Comm Wait		00:00:05.0	0.00%
Memory Wait		00:00:02.0	0.00%
CPU Wait		00:00:01.0	0.00%

Text

```
SELECT ...
FROM T1 s,
        T1_MAXSEQ maxresults
WHERE s.id = maxresults.id AND
    s.seq = maxresults.seq AND
    s.col_a = 0 AND
    s.col_b = :1 AND
    s.col_c < :2 AND
    s.col_d = :3 AND
    rownum < 100
```

Rewritten text includes only the join with the new table T1_MAXSEQ. Inline view was removed.

DB OPTimize
Oracle Performance Tuning & DBA Consulting



Time: 03-Apr-11 00:00 - 30-Apr-11 23:59

Instance: Inst1 Running on machine1

Inst1 Running on machine1

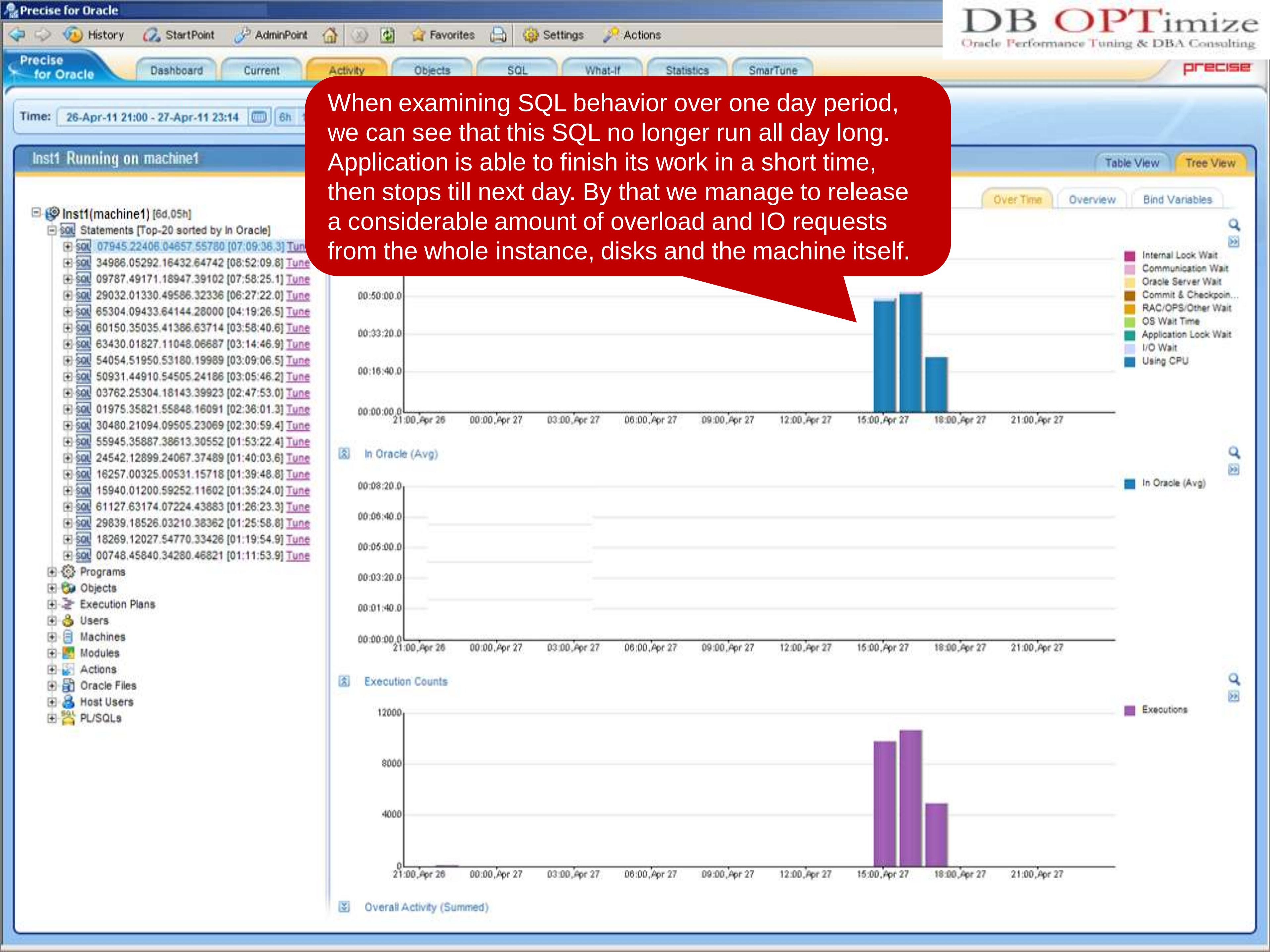
Statement: 07945.22406.04657.55780

pbpr9(mguxp05) [103d,04h]
Statements [Top-20 sorted by In Oracle]
42942.30505.53068.31251 [8d,06h] Tune
60150.35035.41386.63714 [1d,11h] Tune
57723.58696.56668.55923 [1d,10h] Tune
59126.41953.22844.28663 [1d,10h] Tune
54054.51950.53180.19989 [1d,08h] Tune
24542.12899.24067.37489 [1d,04h] Tune
61127.63174.07224.43883 [1d,03h] Tune
00748.45840.34280.46821 [1d,02h] Tune
15940.01200.59252.11602 [1d,00h] Tune
Programs
Objects
Execution Plans
Users
Machines
Modules
Actions
Oracle Files
Host Users
PL/SQLs

Change was applied on April 17th

Since SQL is now able to run rather fast, it manage to have 25k executions a day (compare to several dozens before the change), therefore we can see that productivity and capacity have increased dramatically also. Much more work can be done within same amount of time.

DB OPTimize
Oracle Performance Tuning & DBA Consulting



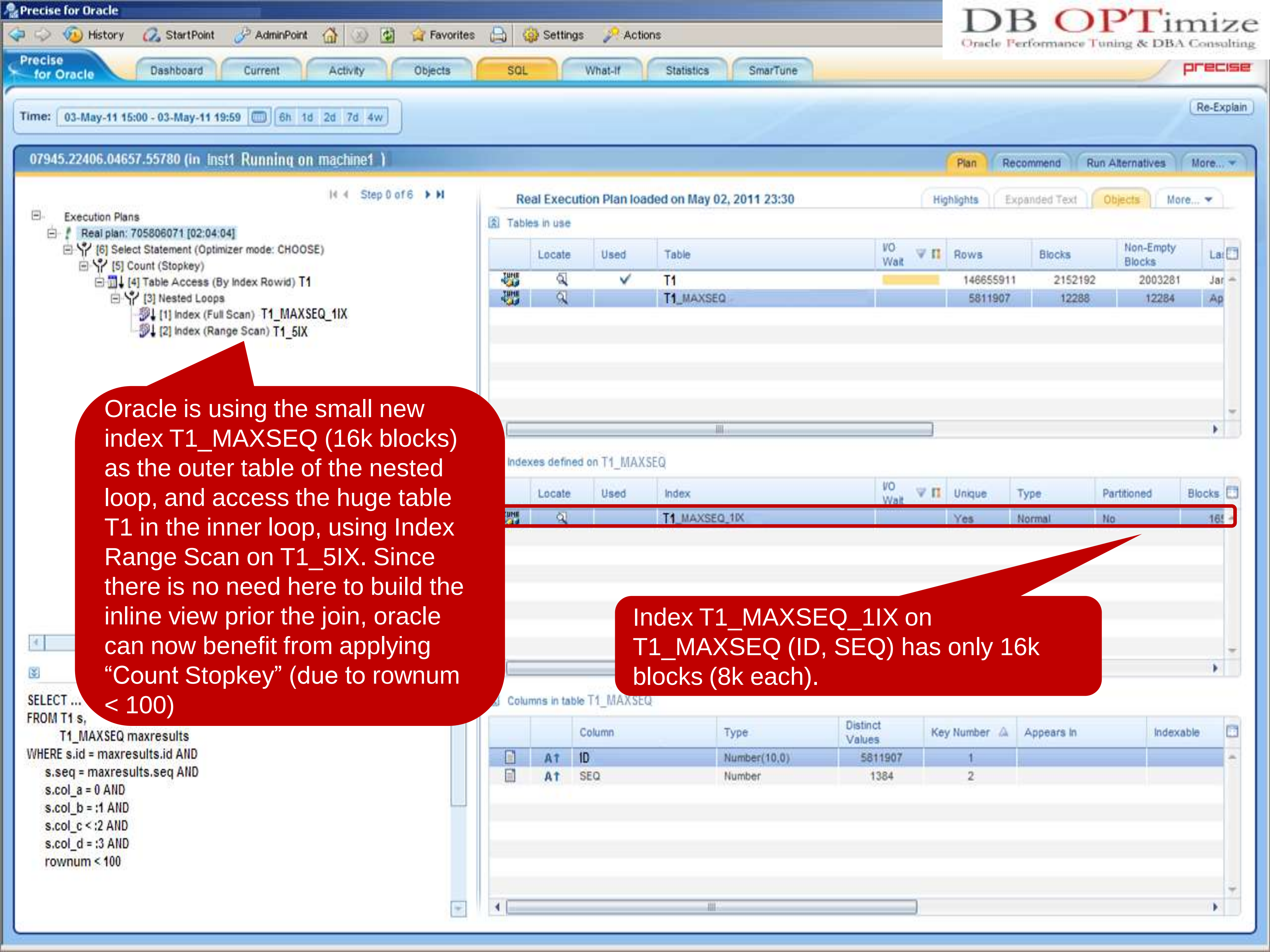
Time: 26-Apr-11 21:00 - 27-Apr-11 23:14 6h

Inst1 Running on machine1

- Inst1(machine1) [6d,05h]
  - Statements [Top-20 sorted by In Oracle]
    - 07945.22406.04657.55780 [07:09:36.3] Tun
    - 34986.05292.16432.64742 [08:52:09.8] Tune
    - 09787.49171.18947.39102 [07:58:25.1] Tune
    - 29032.01330.49586.32336 [06:27:22.0] Tune
    - 65304.09433.64144.28000 [04:19:26.5] Tune
    - 60150.35035.41386.63714 [03:58:40.6] Tune
    - 63430.01827.11048.06687 [03:14:46.9] Tune
    - 54054.51950.53180.19989 [03:09:06.5] Tune
    - 50931.44910.54505.24186 [03:05:46.2] Tune
    - 03762.25304.18143.39923 [02:47:53.0] Tune
    - 01975.35821.55848.16091 [02:36:01.3] Tune
    - 30480.21094.09505.23069 [02:30:59.4] Tune
    - 55945.35887.38613.30552 [01:53:22.4] Tune
    - 24542.12899.24067.37489 [01:40:03.6] Tune
    - 16257.00325.00531.15718 [01:39:48.8] Tune
    - 15940.01200.59252.11602 [01:35:24.0] Tune
    - 61127.63174.07224.43883 [01:26:23.3] Tune
    - 29839.18526.03210.38362 [01:25:58.8] Tune
    - 18269.12027.54770.33426 [01:19:54.9] Tune
    - 00748.45840.34280.46821 [01:11:53.9] Tune
  - Programs
  - Objects
  - Execution Plans
  - Users
  - Machines
  - Modules
  - Actions
  - Oracle Files
  - Host Users
  - PL/SQLs

When examining SQL behavior over one day period, we can see that this SQL no longer run all day long. Application is able to finish its work in a short time, then stops till next day. By that we manage to release a considerable amount of overload and IO requests from the whole instance, disks and the machine itself.

DB OPTimize
Oracle Performance Tuning & DBA Consulting



07945.22406.04657.55780 (in Inst1 Running on machine1)

- Execution Plans
  - Real plan: 705806071 [02:04:04]
    - [6] Select Statement (Optimizer mode: CHOOSE)
      - [5] Count (Stopkey)
        - [4] Table Access (By Index Rowid) T1
          - [3] Nested Loops
            - [1] Index (Full Scan) T1_MAXSEQ_1IX
            - [2] Index (Range Scan) T1_5IX

```
SELECT ...
FROM T1 s,
     T1_MAXSEQ maxresults
WHERE s.id = maxresults.id AND
   s.seq = maxresults.seq AND
   s.col_a = 0 AND
   s.col_b = :1 AND
   s.col_c < :2 AND
   s.col_d = :3 AND
   rownum < 100
```

Real Execution Plan loaded on May 02, 2011 23:30

Tables in use

	Locate	Used	Table	I/O Wait	Rows	Blocks	Non-Empty Blocks	La
		✓	T1		146655911	2152192	2003281	Jar
			T1_MAXSEQ		5811907	12288	12284	Ap

Indexes defined on T1_MAXSEQ

	Locate	Used	Index	I/O Wait	Unique	Type	Partitioned	Blocks
			T1_MAXSEQ_1IX		Yes	Normal	No	16!

Columns in table T1_MAXSEQ

		Column	Type	Distinct Values	Key Number	Appears In	Indexable
	A↑	ID	Number(10,0)	5811907	1		
	A↑	SEQ	Number	1384	2		

Oracle is using the small new index T1_MAXSEQ (16k blocks) as the outer table of the nested loop, and access the huge table T1 in the inner loop, using Index Range Scan on T1_5IX. Since there is no need here to build the inline view prior the join, oracle can now benefit from applying “Count Stopkey” (due to rownum < 100)

Index T1_MAXSEQ_1IX on T1_MAXSEQ (ID, SEQ) has only 16k blocks (8k each).

DB OPTimize

Oracle Performance Tuning & DBA Consulting

www.dboptimize.co.il

merav@dboptimize.co.il

Blog: meravkedem.blogspot.com