ACOUG
All China Oracle User Group
MEMBER

题目:隐式转换引发的血案

中国社会保险
CHINA SOCIAL INSURANCE

About me

- Eric0435,个人网站 http://www.jydba.net/
- ACOUG核心会员
- Oracle Young Expert
- 系统架构师(高级工程师)
- DBAplus社群联合发起人
- 湖南省政府采购评审专家

目录

Oracle基本数据类型与比较规则

- 数字类型
- 日期类型
- 字符类型

  字符类型比较基于以下两种规则

➢二进制或语言排序规则

➢填充空格或不填充空格比较语义

Oracle数据类型转换

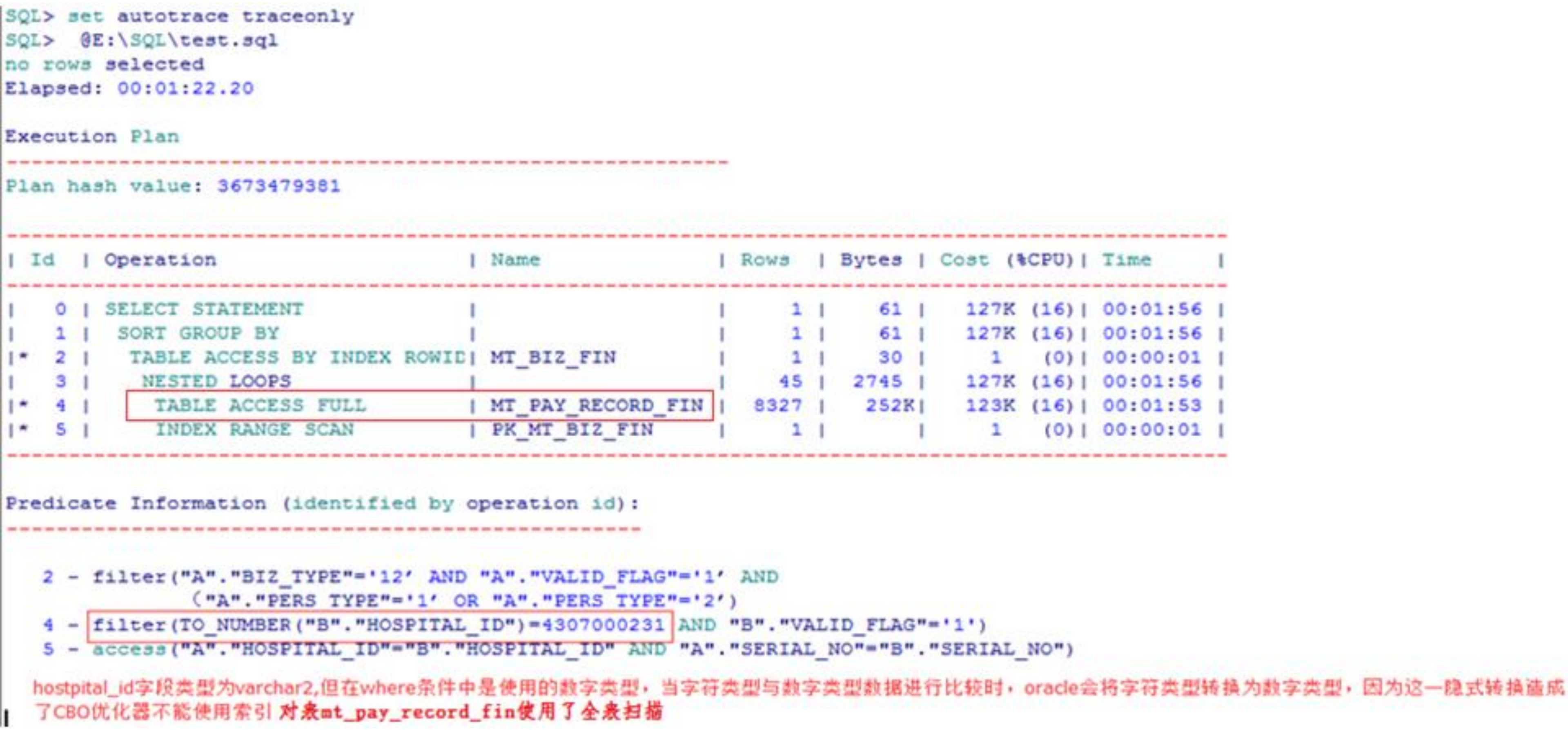
- 显式转换
  - where begin_date=to_date(‘2016-07-18’,’yyyy-mm-dd’)
- 隐式转换

Oracle数据类型隐式转换规则

- 当执行insert与update操作时，Oracle将值转换为表列所定义的数据类型。
- 当执行select from操作时，Oracle将字段数据类型转换为目标变量的数据类型。
- 当比较数字类型与字符类型数据时，Oracle通常会将字符数据转换成数字数据。
- 在字符数据或number数据与浮点数据之间进行转换时可能产生不精确的结果，因为字符类型与number类型使用十进制精度来代表数字数据，而浮点数据使用二进制精度。
- 当将clob类型数据转换为字符类型比如varchar2，或将blob类型数据转换为raw类型，如果数据被转换后比目标数据类型大，那么数据库将会返回错误。
- 当比较字符型和日期型的数据时，oracle会把字符型转换为日期型。
- 如果调用函数或过程等时，如果输入参数的数据类型与函数或者过程定义的参数数据类型不一直，则oracle会把输入参数的数据类型转换为函数或者过程定义的数据类型。
- 用连接操作符(||)时，oracle会把非字符类型的数据转换为字符类型。
- 如果字符类型的数据和非字符类型的数据作算术运算，则oracle会将字符类型的数据转换为合适的数据类型。比如CHAR/VARCHAR2 和NCHAR/NVARCHAR2之间作算术运算，则oracle会将她们都转换为number类型的数据再做比较。
- 比较CHAR/VARCHAR2 和NCHAR/NVARCHAR2时，如果两者字符集不一样，则默认的转换方式是将数据编码从数据库字符集转换为国家字符集。

Oracle隐式转换案例一

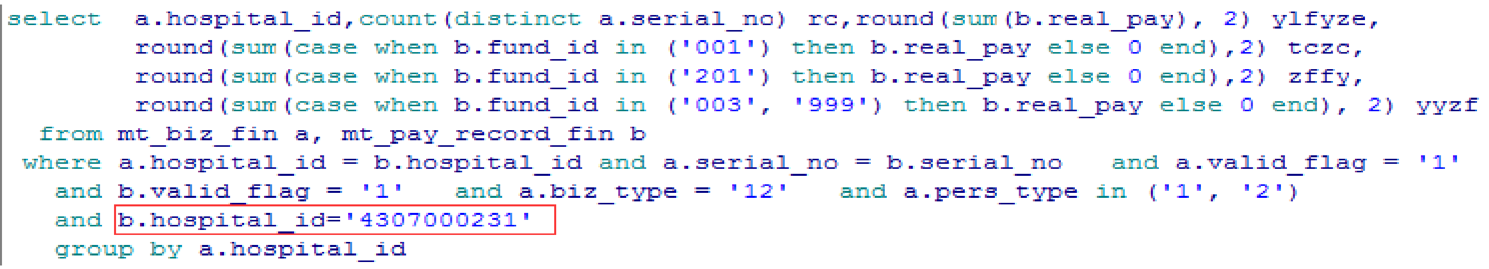
```
select   a.hospital_id,count(distinct a.serial_no) rc,round(sum(b.real_pay), 2) ylfyze,
         round(sum(case when b.fund_id in ('001') then b.real_pay else 0 end),2) tczc,
         round(sum(case when b.fund_id in ('201') then b.real_pay else 0 end),2) zffy,
         round(sum(case when b.fund_id in ('003', '999') then b.real_pay else 0 end), 2) yyzf
  from mt_biz_fin a, mt_pay_record_fin b
 where a.hospital_id = b.hospital_id and a.serial_no = b.serial_no   and a.valid_flag = '1'
   and b.valid_flag = '1'   and a.biz_type = '12'   and a.pers_type in ('1', '2')
   and b.hospital_id=4307000231
   group by a.hospital_id
```



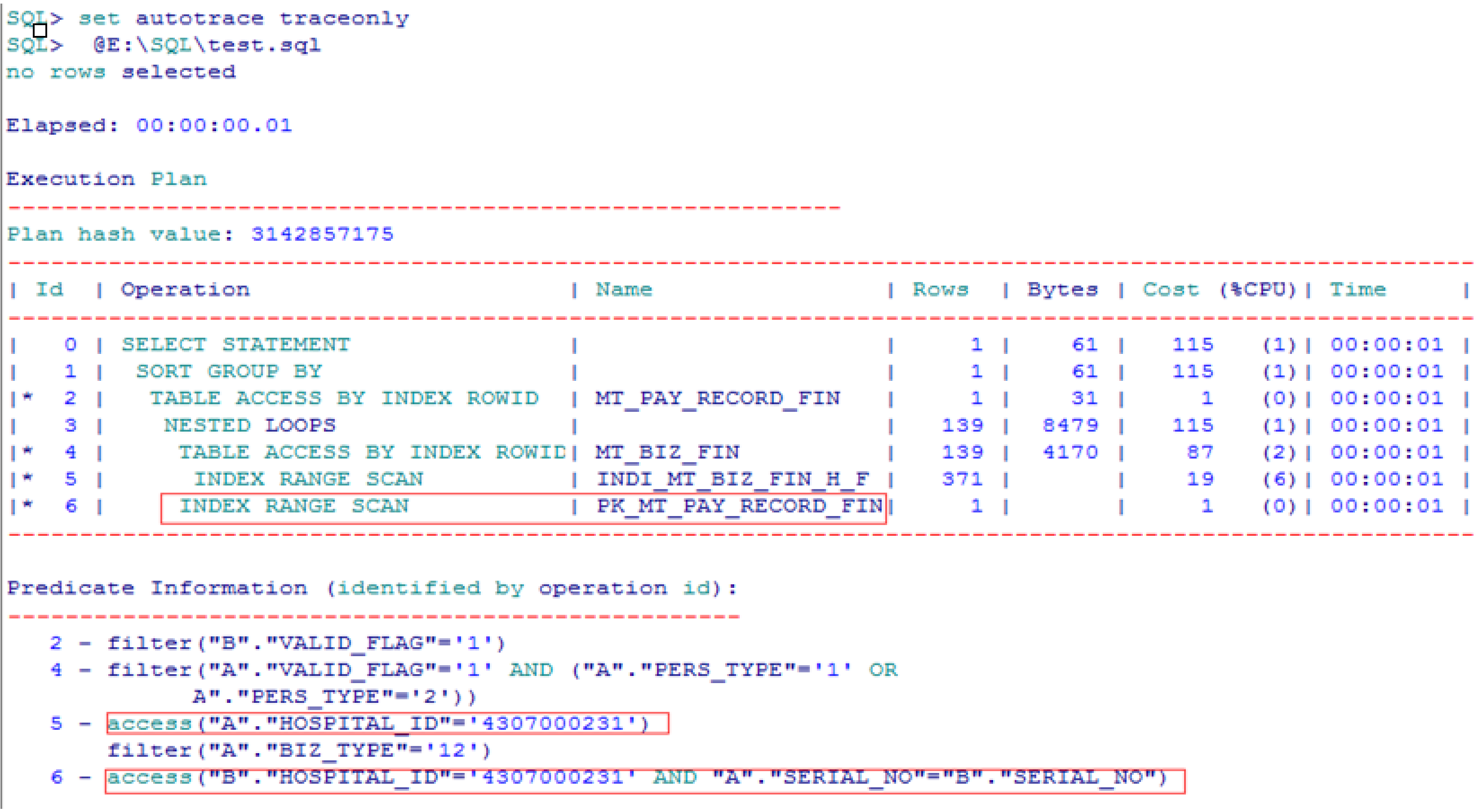
```
SQL> set autotrace traceonly
SQL>  @E:\SQL\test.sql
no rows selected
Elapsed: 00:01:22.20

Execution Plan
----------------------------------------------------------
Plan hash value: 3673479381

-------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                   |     1 |    61 |   127K  (16)| 00:01:56 |
|   1 |  SORT GROUP BY                |                   |     1 |    61 |   127K  (16)| 00:01:56 |
|*  2 |   TABLE ACCESS BY INDEX ROWID | MT_BIZ_FIN        |     1 |    30 |     1    (0)| 00:00:01 |
|   3 |    NESTED LOOPS               |                   |    45 |  2745 |   127K  (16)| 00:01:56 |
|*  4 |     TABLE ACCESS FULL         | MT_PAY_RECORD_FIN |  8327 |  252K |   123K  (16)| 00:01:53 |
|*  5 |     INDEX RANGE SCAN          | PK_MT_BIZ_FIN     |     1 |       |     1    (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter("A"."BIZ_TYPE"='12' AND "A"."VALID_FLAG"='1' AND
              ("A"."PERS_TYPE"='1' OR "A"."PERS_TYPE"='2')
   4 - filter(TO_NUMBER("B"."HOSPITAL_ID")=4307000231 AND "B"."VALID_FLAG"='1')
   5 - access("A"."HOSPITAL_ID"="B"."HOSPITAL_ID" AND "A"."SERIAL_NO"="B"."SERIAL_NO")
```

hostpital_id字段类型为varchar2,但在where条件中是使用的数字类型，当字符类型与数字类型数据进行比较时，oracle会将字符类型转换为数字类型，因为这一隐式转换造成了CBO优化器不能使用索引 **对表mt_pay_record_fin使用了全表扫描**

Oracle隐式转换案例一

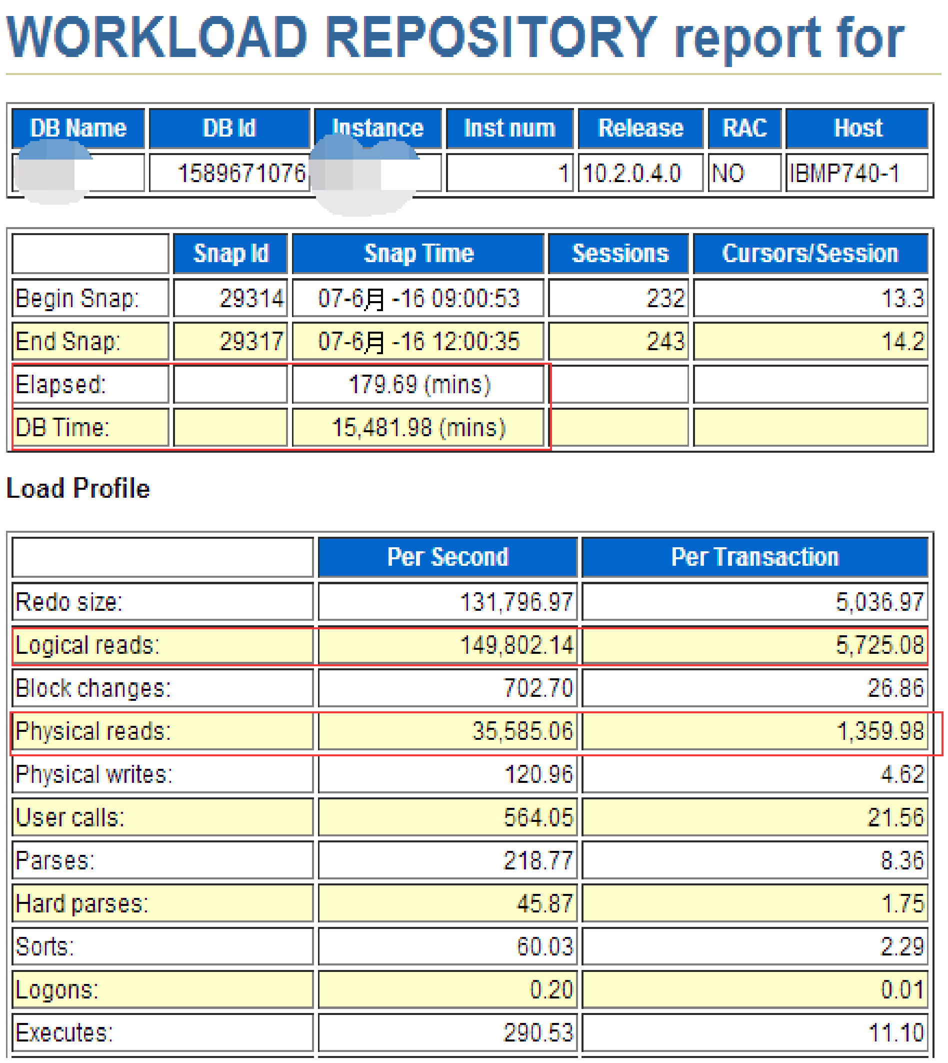


```
select   a.hospital_id,count(distinct a.serial_no) rc,round(sum(b.real_pay), 2) ylfyze,
         round(sum(case when b.fund_id in ('001') then b.real_pay else 0 end),2) tczc,
         round(sum(case when b.fund_id in ('201') then b.real_pay else 0 end),2) zffy,
         round(sum(case when b.fund_id in ('003', '999') then b.real_pay else 0 end), 2) yyzf
  from mt_biz_fin a, mt_pay_record_fin b
 where a.hospital_id = b.hospital_id and a.serial_no = b.serial_no   and a.valid_flag = '1'
   and b.valid_flag = '1'   and a.biz_type = '12'   and a.pers_type in ('1', '2')
   and b.hospital_id='4307000231'
   group by a.hospital_id
```



```
SQL> set autotrace traceonly
SQL>  @E:\SQL\test.sql
no rows selected

Elapsed: 00:00:00.01

Execution Plan
----------------------------------------------------------
Plan hash value: 3142857175

--------------------------------------------------------------------------------------------------------
| Id  | Operation                        | Name                  | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                 |                       |     1 |    61 |   115   (1)| 00:00:01 |
|   1 |  SORT GROUP BY                   |                       |     1 |    61 |   115   (1)| 00:00:01 |
|*  2 |   TABLE ACCESS BY INDEX ROWID    | MT_PAY_RECORD_FIN     |     1 |    31 |     1   (0)| 00:00:01 |
|   3 |    NESTED LOOPS                  |                       |   139 |  8479 |   115   (1)| 00:00:01 |
|*  4 |     TABLE ACCESS BY INDEX ROWID| MT_BIZ_FIN            |   139 |  4170 |    87   (2)| 00:00:01 |
|*  5 |      INDEX RANGE SCAN            | INDI_MT_BIZ_FIN_H_F |   371 |       |    19   (6)| 00:00:01 |
|*  6 |     INDEX RANGE SCAN             | PK_MT_PAY_RECORD_FIN|     1 |       |     1   (0)| 00:00:01 |
--------------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter("B"."VALID_FLAG"='1')
   4 - filter("A"."VALID_FLAG"='1' AND ("A"."PERS_TYPE"='1' OR
              A"."PERS_TYPE"='2'))
   5 - access("A"."HOSPITAL_ID"='4307000231')
       filter("A"."BIZ_TYPE"='12')
   6 - access("B"."HOSPITAL_ID"='4307000231' AND "A"."SERIAL_NO"="B"."SERIAL_NO")
```

Oracle隐式转换案例二



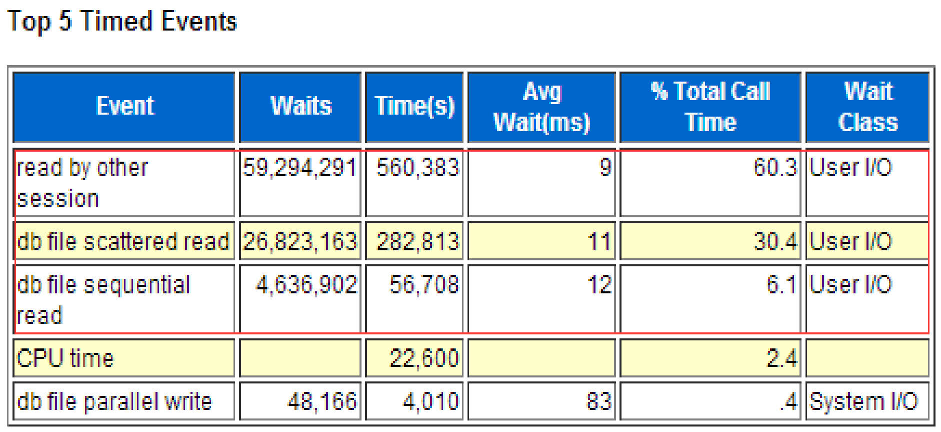
WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst num	Release	RAC	Host
[redacted]	1589671076	[redacted]	1	10.2.0.4.0	NO	IBMP740-1

	Snap Id	Snap Time	Sessions	Cursors/Session
Begin Snap:	29314	07-6月 -16 09:00:53	232	13.3
End Snap:	29317	07-6月 -16 12:00:35	243	14.2
Elapsed:		179.69 (mins)		
DB Time:		15,481.98 (mins)		

Load Profile

	Per Second	Per Transaction
Redo size:	131,796.97	5,036.97
Logical reads:	149,802.14	5,725.08
Block changes:	702.70	26.86
Physical reads:	35,585.06	1,359.98
Physical writes:	120.96	4.62
User calls:	564.05	21.56
Parses:	218.77	8.36
Hard parses:	45.87	1.75
Sorts:	60.03	2.29
Logons:	0.20	0.01
Executes:	290.53	11.10

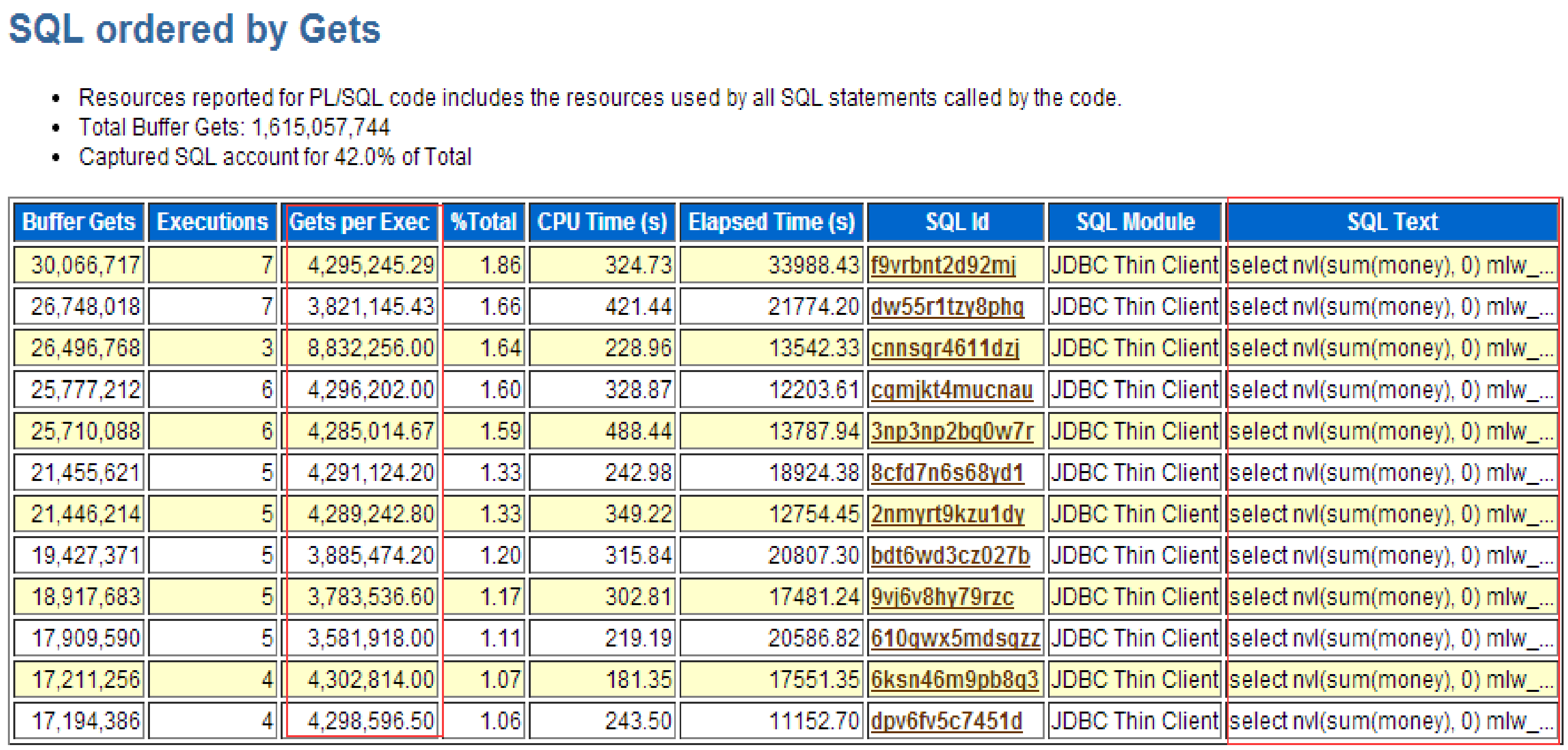


Top 5 Timed Events

Event	Waits	Time(s)	Avg Wait(ms)	% Total Call Time	Wait Class
read by other session	59,294,291	560,383	9	60.3	User I/O
db file scattered read	26,823,163	282,813	11	30.4	User I/O
db file sequential read	4,636,902	56,708	12	6.1	User I/O
CPU time		22,600		2.4	
db file parallel write	48,166	4,010	83	.4	System I/O

在采样时间为3个小时的awr报告中可以看到db time为15482分钟，CPU为64(逻辑)，那么每个CPU所消耗的时间为85分钟，小于采样时间的50%。每秒的逻辑读取为149802次，物理读为35585次，这两项确实很高，从等待事件来看，主要类型是用户I/O，read by other session占了总等待时间的60.3%，其次是db file scattered read占了百分30.4%，db file sequential read占了百分6.1%.read by other session等待的原理是多个会话并发将同一数据块从磁盘读入SGA，但ORACLE同一时间只允许一个会话从磁盘将同一数据块读入SGA，在并发情况下其它session必须等待。

Oracle隐式转换案例二



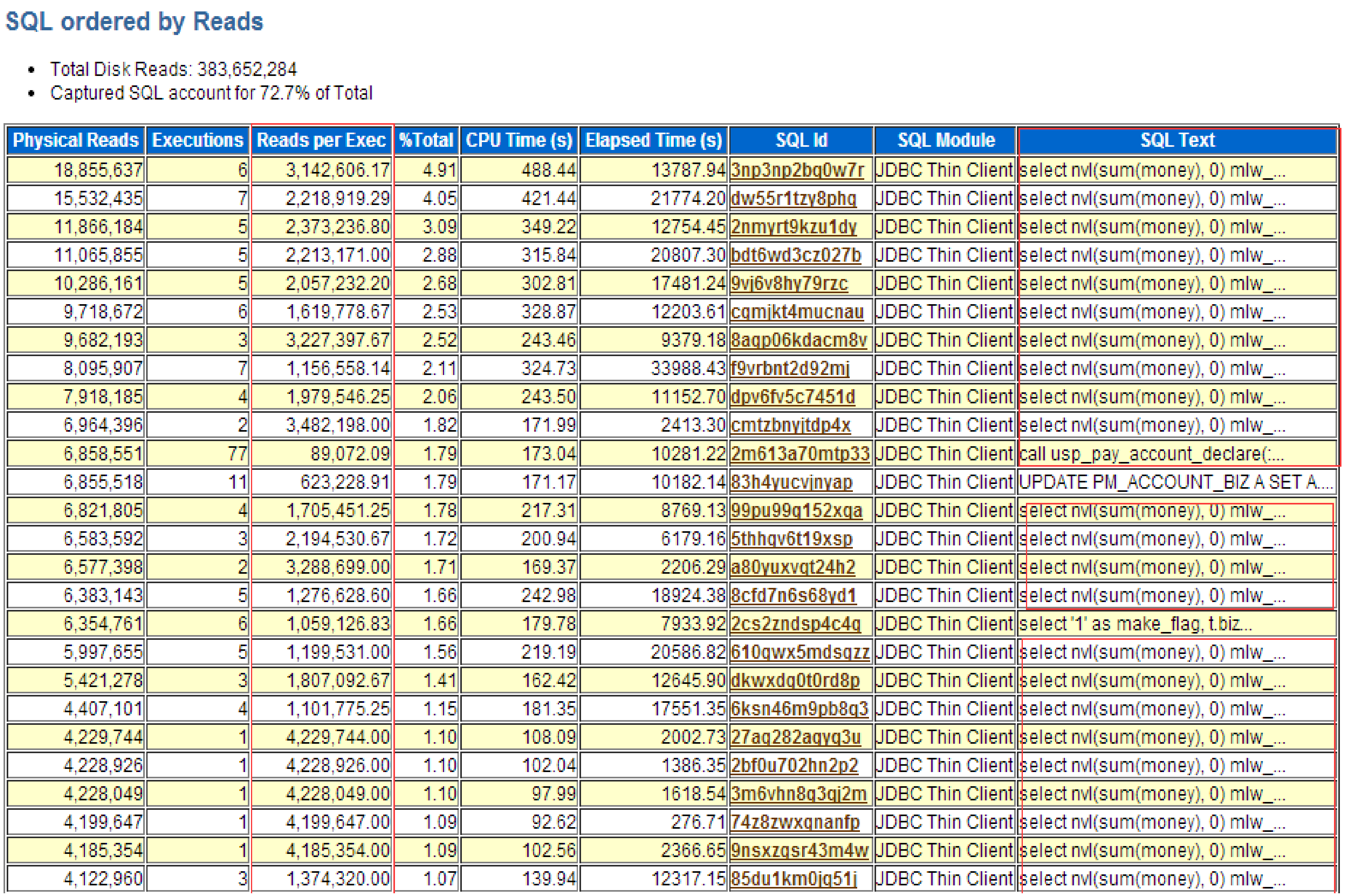
SQL ordered by Gets

- Resources reported for PL/SQL code includes the resources used by all SQL statements called by the code.
- Total Buffer Gets: 1,615,057,744
- Captured SQL account for 42.0% of Total

Buffer Gets	Executions	Gets per Exec	%Total	CPU Time (s)	Elapsed Time (s)	SQL Id	SQL Module	SQL Text
30,066,717	7	4,295,245.29	1.86	324.73	33988.43	f9vrbnt2d92mj	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
26,748,018	7	3,821,145.43	1.66	421.44	21774.20	dw55r1tzy8phq	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
26,496,768	3	8,832,256.00	1.64	228.96	13542.33	cnnsqr4611dzj	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
25,777,212	6	4,296,202.00	1.60	328.87	12203.61	cqmjkt4mucnau	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
25,710,088	6	4,285,014.67	1.59	488.44	13787.94	3np3np2bq0w7r	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
21,455,621	5	4,291,124.20	1.33	242.98	18924.38	8cfd7n6s68yd1	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
21,446,214	5	4,289,242.80	1.33	349.22	12754.45	2nmyrt9kzu1dy	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
19,427,371	5	3,885,474.20	1.20	315.84	20807.30	bdt6wd3cz027b	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
18,917,683	5	3,783,536.60	1.17	302.81	17481.24	9vj6v8hy79rzc	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
17,909,590	5	3,581,918.00	1.11	219.19	20586.82	610qwx5mdsqzz	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
17,211,256	4	4,302,814.00	1.07	181.35	17551.35	6ksn46m9pb8q3	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
17,194,386	4	4,298,596.50	1.06	243.50	11152.70	dpv6fv5c7451d	JDBC Thin Client	select nvl(sum(money), 0) mlw_...

通过profile我们知道每秒的逻辑读为15w左右，从top 逻辑读SQL可以看到这些功能相同的SQL，每次执行逻辑读都在几百万

Oracle隐式转换案例二



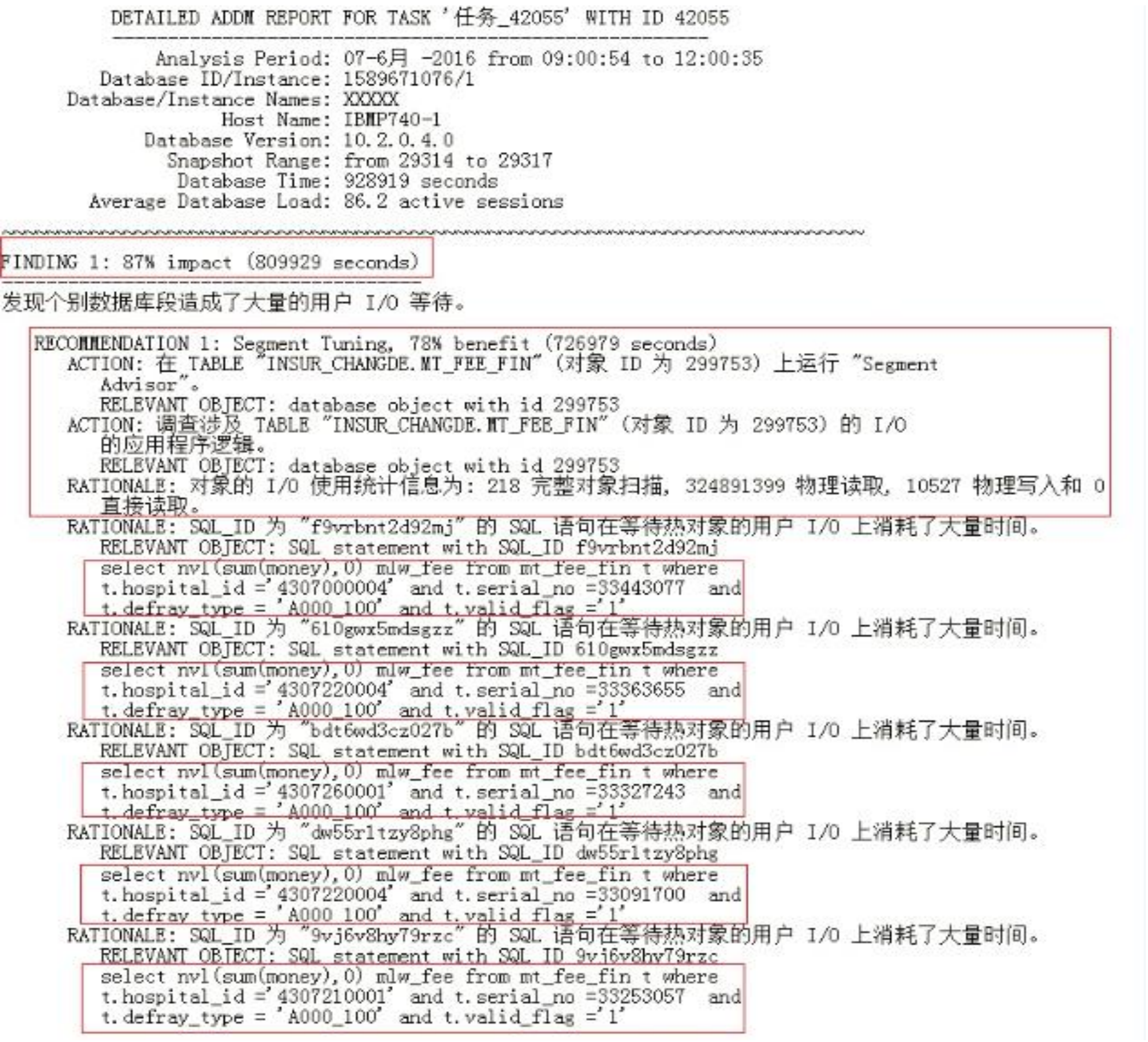
SQL ordered by Reads

- Total Disk Reads: 383,652,284
- Captured SQL account for 72.7% of Total

Physical Reads	Executions	Reads per Exec	%Total	CPU Time (s)	Elapsed Time (s)	SQL Id	SQL Module	SQL Text
18,855,637	6	3,142,606.17	4.91	488.44	13787.94	3np3np2bq0w7r	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
15,532,435	7	2,218,919.29	4.05	421.44	21774.20	dw55r1tzy8phq	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
11,866,184	5	2,373,236.80	3.09	349.22	12754.45	2nmyrt9kzu1dy	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
11,065,855	5	2,213,171.00	2.88	315.84	20807.30	bdt6wd3cz027b	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
10,286,161	5	2,057,232.20	2.68	302.81	17481.24	9vj6v8hy79rzc	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
9,718,672	6	1,619,778.67	2.53	328.87	12203.61	cqmjkt4mucnau	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
9,682,193	3	3,227,397.67	2.52	243.46	9379.18	8aqp06kdacm8v	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
8,095,907	7	1,156,558.14	2.11	324.73	33988.43	f9vrbnt2d92mj	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
7,918,185	4	1,979,546.25	2.06	243.50	11152.70	dpv6fv5c7451d	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,964,396	2	3,482,198.00	1.82	171.99	2413.30	cmtzbnyjtdp4x	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,858,551	77	89,072.09	1.79	173.04	10281.22	2m613a70mtp33	JDBC Thin Client	call usp_pay_account_declare(:...
6,855,518	11	623,228.91	1.79	171.17	10182.14	83h4yucvjnyap	JDBC Thin Client	UPDATE PM_ACCOUNT_BIZ A SET A...
6,821,805	4	1,705,451.25	1.78	217.31	8769.13	99pu99q152xqa	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,583,592	3	2,194,530.67	1.72	200.94	6179.16	5thhqv6t19xsp	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,577,398	2	3,288,699.00	1.71	169.37	2206.29	a80yuxvqt24h2	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,383,143	5	1,276,628.60	1.66	242.98	18924.38	8cfd7n6s68yd1	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
6,354,761	6	1,059,126.83	1.66	179.78	7933.92	2cs2zndsp4c4q	JDBC Thin Client	select '1' as make_flag, t.biz...
5,997,655	5	1,199,531.00	1.56	219.19	20586.82	610qwx5mdsqzz	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
5,421,278	3	1,807,092.67	1.41	162.42	12645.90	dkwxdq0t0rd8p	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,407,101	4	1,101,775.25	1.15	181.35	17551.35	6ksn46m9pb8q3	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,229,744	1	4,229,744.00	1.10	108.09	2002.73	27aq282aqyq3u	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,228,926	1	4,228,926.00	1.10	102.04	1386.35	2bf0u702hn2p2	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,228,049	1	4,228,049.00	1.10	97.99	1618.54	3m6vhn8q3qj2m	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,199,647	1	4,199,647.00	1.09	92.62	276.71	74z8zwxqnanfp	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,185,354	1	4,185,354.00	1.09	102.56	2366.65	9nsxzqsr43m4w	JDBC Thin Client	select nvl(sum(money), 0) mlw_...
4,122,960	3	1,374,320.00	1.07	139.94	12317.15	85du1km0jq51j	JDBC Thin Client	select nvl(sum(money), 0) mlw_...

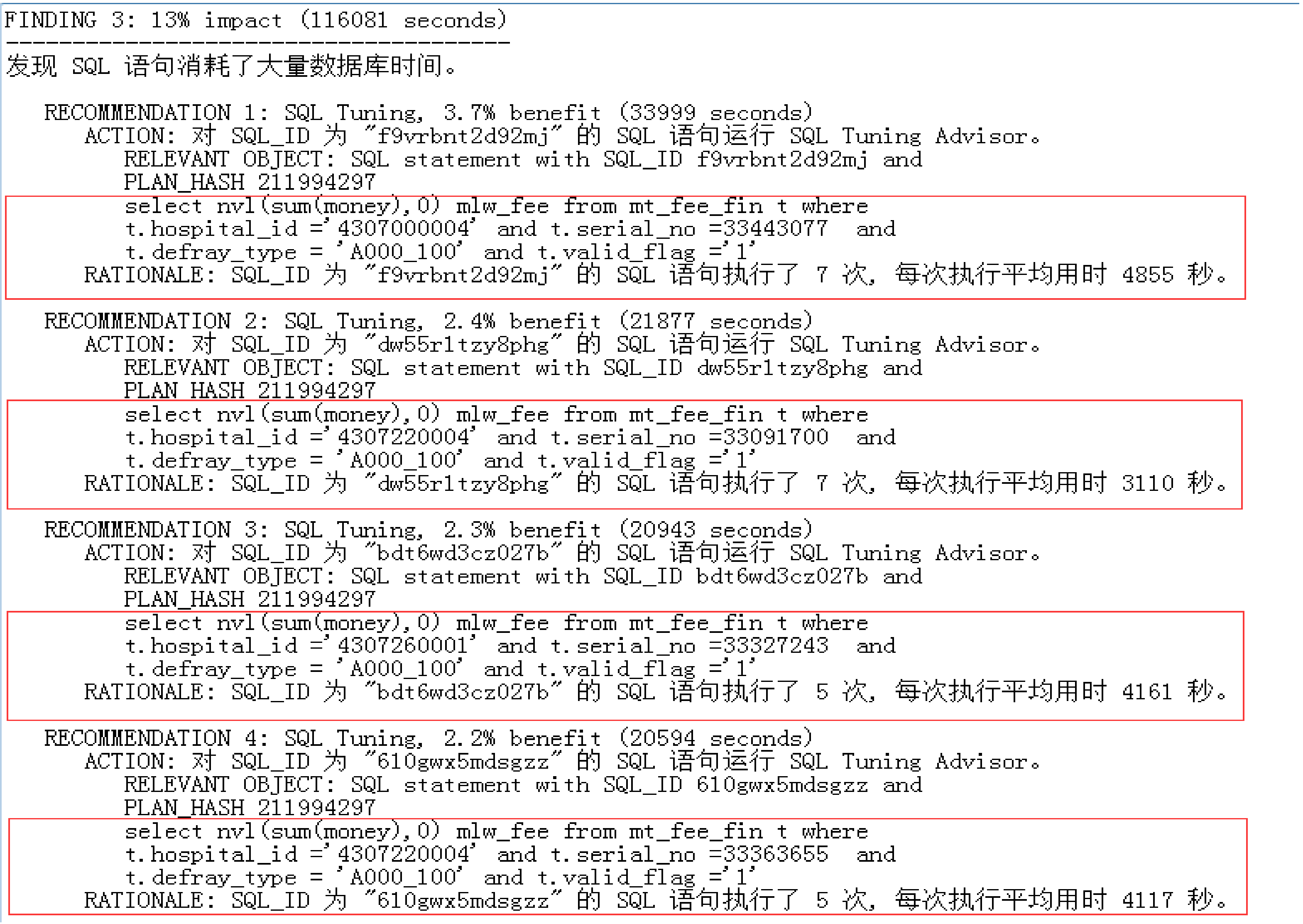
同样这些SQL单次执行的物理读也是百万级别

Oracle隐式转换案例二



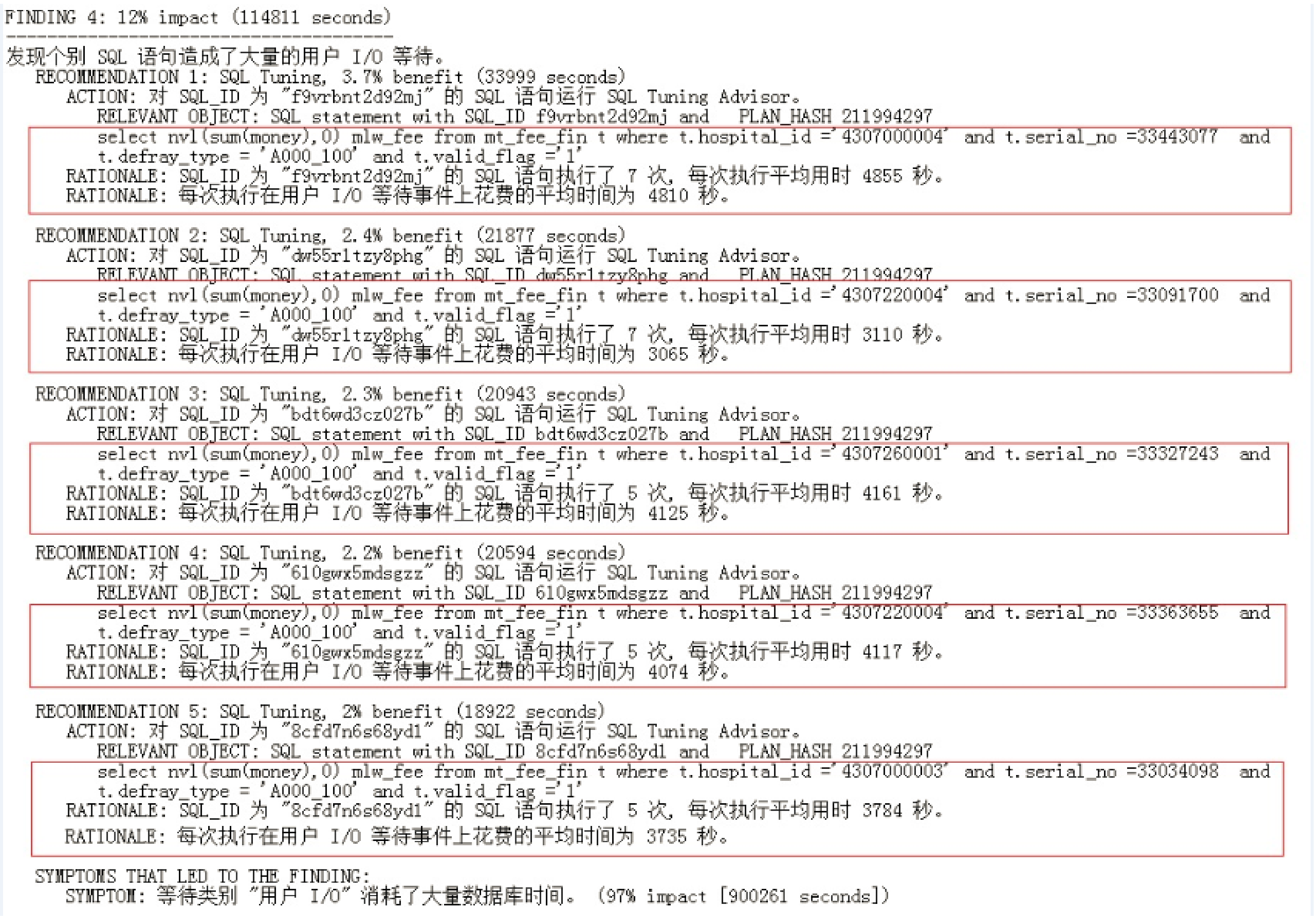
```
          DETAILED ADDM REPORT FOR TASK '任务_42055' WITH ID 42055
          ----------------------------------------------------------

              Analysis Period: 07-6月 -2016 from 09:00:54 to 12:00:35
         Database ID/Instance: 1589671076/1
      Database/Instance Names: XXXXX
                    Host Name: IBMP740-1
             Database Version: 10.2.0.4.0
               Snapshot Range: from 29314 to 29317
                Database Time: 928919 seconds
        Average Database Load: 86.2 active sessions

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

FINDING 1: 87% impact (809929 seconds)
--------------------------------------
发现个别数据库段造成了大量的用户 I/O 等待。

   RECOMMENDATION 1: Segment Tuning, 78% benefit (726979 seconds)
      ACTION: 在 TABLE "INSUR_CHANGDE.MT_FEE_FIN" (对象 ID 为 299753) 上运行 "Segment
         Advisor"。
         RELEVANT OBJECT: database object with id 299753
      ACTION: 调查涉及 TABLE "INSUR_CHANGDE.MT_FEE_FIN" (对象 ID 为 299753) 的 I/O
         的应用程序逻辑。
         RELEVANT OBJECT: database object with id 299753
      RATIONALE: 对象的 I/O 使用统计信息为: 218 完整对象扫描, 324891399 物理读取, 10527 物理写入和 0
         直接读取。
      RATIONALE: SQL_ID 为 "f9vrbnt2d92mj" 的 SQL 语句在等待热对象的用户 I/O 上消耗了大量时间。
         RELEVANT OBJECT: SQL statement with SQL_ID f9vrbnt2d92mj
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307000004' and t.serial_no =33443077  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "610gwx5mdsgzz" 的 SQL 语句在等待热对象的用户 I/O 上消耗了大量时间。
         RELEVANT OBJECT: SQL statement with SQL_ID 610gwx5mdsgzz
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307220004' and t.serial_no =33363655  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "bdt6wd3cz027b" 的 SQL 语句在等待热对象的用户 I/O 上消耗了大量时间。
         RELEVANT OBJECT: SQL statement with SQL_ID bdt6wd3cz027b
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307260001' and t.serial_no =33327243  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "dw55r1tzy8phg" 的 SQL 语句在等待热对象的用户 I/O 上消耗了大量时间。
         RELEVANT OBJECT: SQL statement with SQL_ID dw55r1tzy8phg
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307220004' and t.serial_no =33091700  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "9vj6v8hy79rzc" 的 SQL 语句在等待热对象的用户 I/O 上消耗了大量时间。
         RELEVANT OBJECT: SQL statement with SQL_ID 9vj6v8hy79rzc
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307210001' and t.serial_no =33253057  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
```

Oracle隐式转换案例二



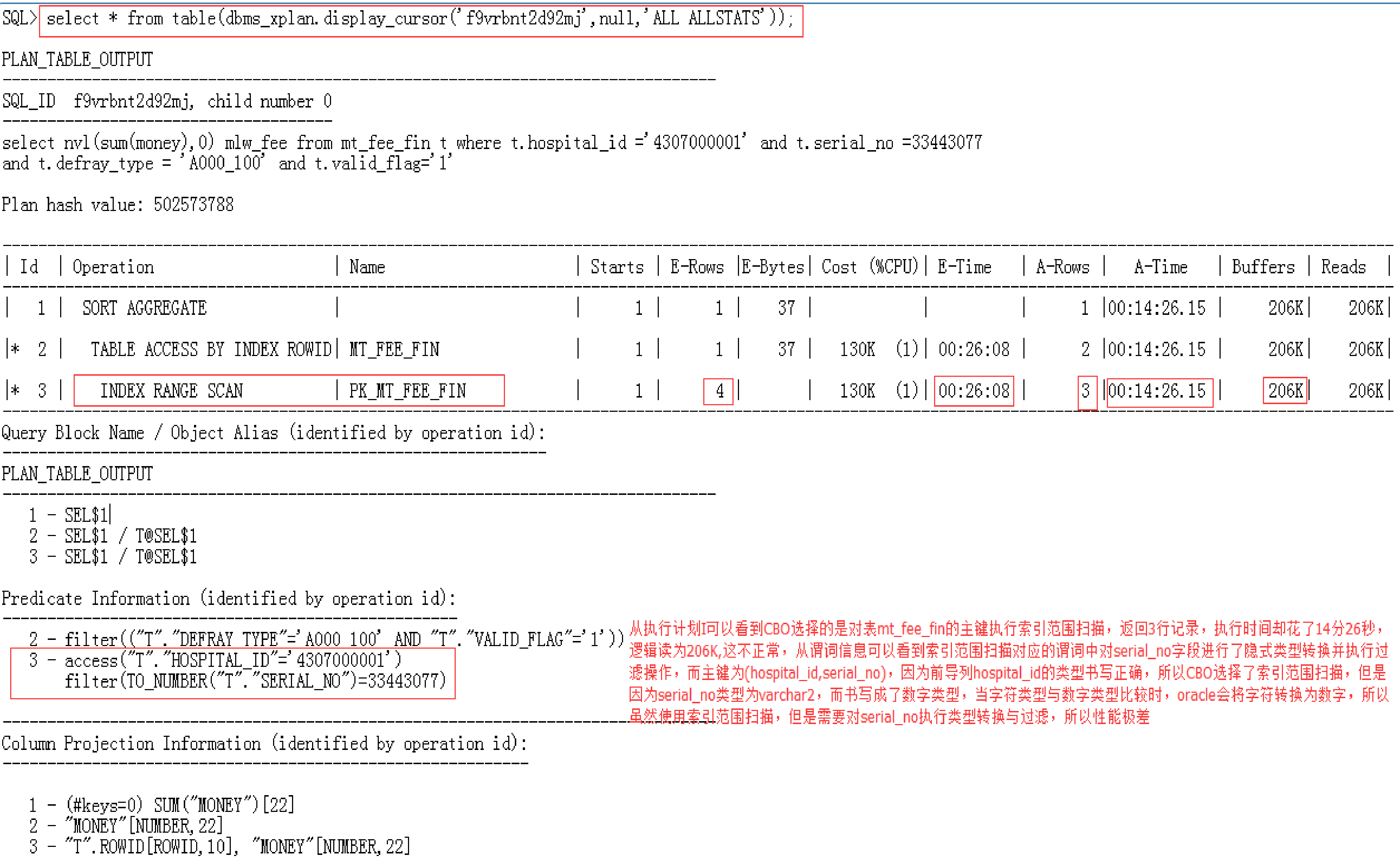
```
FINDING 3: 13% impact (116081 seconds)
--------------------------------------
发现 SQL 语句消耗了大量数据库时间。

   RECOMMENDATION 1: SQL Tuning, 3.7% benefit (33999 seconds)
      ACTION: 对 SQL_ID 为 "f9vrbnt2d92mj" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID f9vrbnt2d92mj and
         PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307000004' and t.serial_no =33443077  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "f9vrbnt2d92mj" 的 SQL 语句执行了 7 次, 每次执行平均用时 4855 秒。

   RECOMMENDATION 2: SQL Tuning, 2.4% benefit (21877 seconds)
      ACTION: 对 SQL_ID 为 "dw55r1tzy8phg" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID dw55r1tzy8phg and
         PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307220004' and t.serial_no =33091700  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "dw55r1tzy8phg" 的 SQL 语句执行了 7 次, 每次执行平均用时 3110 秒。

   RECOMMENDATION 3: SQL Tuning, 2.3% benefit (20943 seconds)
      ACTION: 对 SQL_ID 为 "bdt6wd3cz027b" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID bdt6wd3cz027b and
         PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307260001' and t.serial_no =33327243  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "bdt6wd3cz027b" 的 SQL 语句执行了 5 次, 每次执行平均用时 4161 秒。

   RECOMMENDATION 4: SQL Tuning, 2.2% benefit (20594 seconds)
      ACTION: 对 SQL_ID 为 "610gwx5mdsgzz" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID 610gwx5mdsgzz and
         PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where
         t.hospital_id ='4307220004' and t.serial_no =33363655  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "610gwx5mdsgzz" 的 SQL 语句执行了 5 次, 每次执行平均用时 4117 秒。
```

Oracle隐式转换案例二



```
FINDING 4: 12% impact (114811 seconds)
---------------------------------------
发现个别 SQL 语句造成了大量的用户 I/O 等待。
   RECOMMENDATION 1: SQL Tuning, 3.7% benefit (33999 seconds)
      ACTION: 对 SQL_ID 为 "f9vrbnt2d92mj" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID f9vrbnt2d92mj and   PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307000004' and t.serial_no =33443077  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "f9vrbnt2d92mj" 的 SQL 语句执行了 7 次, 每次执行平均用时 4855 秒。
      RATIONALE: 每次执行在用户 I/O 等待事件上花费的平均时间为 4810 秒。

   RECOMMENDATION 2: SQL Tuning, 2.4% benefit (21877 seconds)
      ACTION: 对 SQL_ID 为 "dw55r1tzy8phg" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID dw55r1tzy8phg and   PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307220004' and t.serial_no =33091700  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "dw55r1tzy8phg" 的 SQL 语句执行了 7 次, 每次执行平均用时 3110 秒。
      RATIONALE: 每次执行在用户 I/O 等待事件上花费的平均时间为 3065 秒。

   RECOMMENDATION 3: SQL Tuning, 2.3% benefit (20943 seconds)
      ACTION: 对 SQL_ID 为 "bdt6wd3cz027b" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID bdt6wd3cz027b and   PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307260001' and t.serial_no =33327243  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "bdt6wd3cz027b" 的 SQL 语句执行了 5 次, 每次执行平均用时 4161 秒。
      RATIONALE: 每次执行在用户 I/O 等待事件上花费的平均时间为 4125 秒。

   RECOMMENDATION 4: SQL Tuning, 2.2% benefit (20594 seconds)
      ACTION: 对 SQL_ID 为 "610gwx5mdsgzz" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID 610gwx5mdsgzz and   PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307220004' and t.serial_no =33363655  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "610gwx5mdsgzz" 的 SQL 语句执行了 5 次, 每次执行平均用时 4117 秒。
      RATIONALE: 每次执行在用户 I/O 等待事件上花费的平均时间为 4074 秒。

   RECOMMENDATION 5: SQL Tuning, 2% benefit (18922 seconds)
      ACTION: 对 SQL_ID 为 "8cfd7n6s68yd1" 的 SQL 语句运行 SQL Tuning Advisor。
         RELEVANT OBJECT: SQL statement with SQL_ID 8cfd7n6s68yd1 and   PLAN_HASH 211994297
         select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307000003' and t.serial_no =33034098  and
         t.defray_type = 'A000_100' and t.valid_flag ='1'
      RATIONALE: SQL_ID 为 "8cfd7n6s68yd1" 的 SQL 语句执行了 5 次, 每次执行平均用时 3784 秒。
      RATIONALE: 每次执行在用户 I/O 等待事件上花费的平均时间为 3735 秒。

   SYMPTOMS THAT LED TO THE FINDING:
      SYMPTOM: 等待类别 "用户 I/O" 消耗了大量数据库时间。 (97% impact [900261 seconds])
```

Oracle隐式转换案例二



```
SQL> select * from table(dbms_xplan.display_cursor('f9vrbnt2d92mj',null,'ALL ALLSTATS'));

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
SQL_ID  f9vrbnt2d92mj, child number 0
-------------------------------------
select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307000001' and t.serial_no =33443077
and t.defray_type = 'A000_100' and t.valid_flag='1'

Plan hash value: 502573788

------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name           | Starts | E-Rows |E-Bytes| Cost (%CPU)| E-Time   | A-Rows |   A-Time    | Buffers | Reads  |
------------------------------------------------------------------------------------------------------------------------------------------------------------
|   1 |  SORT AGGREGATE              |                |      1 |      1 |    37 |            |          |      1 |00:14:26.15 |    206K|   206K|

|*  2 |   TABLE ACCESS BY INDEX ROWID| MT_FEE_FIN     |      1 |      1 |    37 |   130K  (1)| 00:26:08 |      2 |00:14:26.15 |    206K|   206K|

|*  3 |    INDEX RANGE SCAN          | PK_MT_FEE_FIN  |      1 |      4 |       |   130K  (1)| 00:26:08 |      3 |00:14:26.15 |    206K|   206K|
------------------------------------------------------------------------------------------------------------------------------------------------------------
Query Block Name / Object Alias (identified by operation id):
-------------------------------------------------------------

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

   1 - SEL$1
   2 - SEL$1 / T@SEL$1
   3 - SEL$1 / T@SEL$1

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter(("T"."DEFRAY_TYPE"='A000_100' AND "T"."VALID_FLAG"='1'))
   3 - access("T"."HOSPITAL_ID"='4307000001')
       filter(TO_NUMBER("T"."SERIAL_NO")=33443077)

------------------------------------------------------------------------------------------------------------------------------------------------------------
Column Projection Information (identified by operation id):
-----------------------------------------------------------

   1 - (#keys=0) SUM("MONEY")[22]
   2 - "MONEY"[NUMBER,22]
   3 - "T".ROWID[ROWID,10], "MONEY"[NUMBER,22]
```

从执行计划I可以看到CBO选择的是对表mt_fee_fin的主键执行索引范围扫描，返回3行记录，执行时间却花了14分26秒，逻辑读为206K,这不正常，从谓词信息可以看到索引范围扫描对应的谓词中对serial_no字段进行了隐式类型转换并执行过滤操作，而主键为(hospital_id,serial_no)，因为前导列hospital_id的类型书写正确，所以CBO选择了索引范围扫描，但是因为serial_no类型为varchar2，而书写成了数字类型，当字符类型与数字类型比较时，oracle会将字符转换为数字，所以虽然使用索引范围扫描，但是需要对serial_no执行类型转换与过滤，所以性能极差

Oracle隐式转换案例二

```
SQL> select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307000001' and t.serial_no ='=33443077' and t.defray_type = 'A000_100' and
t.valid_flag='1';

  MLW_FEE
----------
    63.18

SQL> select * from table(dbms_xplan.display_cursor(null,null,'ALL ALLSTATS'));

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
SQL_ID  adthdmabrmzyz, child number 0
-------------------------------------
select nvl(sum(money),0) mlw_fee from mt_fee_fin t where t.hospital_id ='4307000001' and t.serial_no ='33443077' and t.defray_type ='A000_100'
and t.valid_flag='1'

Plan hash value: 4172899360

------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                      | Name          | Starts | E-Rows |E-Bytes| Cost (%CPU)| E-Time     | A-Rows |   A-Time    | Buffers | Reads  |
------------------------------------------------------------------------------------------------------------------------------------------------------
|   1 |  SORT AGGREGATE                |               |      1 |      1 |    37 |            |            |      1 |00:00:00.01  |       5 |      2 |
|*  2 |   TABLE ACCESS BY INDEX ROWID  | MT_FEE_FIN    |      1 |      1 |    37 |     1   (0)| 00:00:01   |      2 |00:00:00.01  |       5 |      2 |
|*  3 |    INDEX RANGE SCAN            | PK_MT_FEE_FIN |      1 |      4 |       |     1   (0)| 00:00:01   |      3 |00:00:00.01  |       4 |      2 |
------------------------------------------------------------------------------------------------------------------------------------------------------
Query Block Name / Object Alias (identified by operation id):
-------------------------------------------------------------

   1 - SEL$1
   2 - SEL$1 / T@SEL$1
   3 - SEL$1 / T@SEL$1

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter(("T"."DEFRAY_TYPE"='A000_100' AND "T"."VALID_FLAG"='1'))
   3 - access("T"."HOSPITAL_ID"='4307000001' AND "T"."SERIAL_NO"='33565503')
```

当hospital_id,serial_no谓词类型书写正确后，CBO选择对表mt_fee_fin的主键执行索引范围扫描，返回3行记录，实际执行时间为0.01秒，逻辑读为4,物理读为2.从谓词信息部分中可以看到执行计划ID=3的谓词信息没有了之前的类型转换与过滤，正是没有了类型转换与过滤使用得索引范围扫描的效率提高了几个数量级

Oracle数据类型隐式转换的缺点

- 当使用显式数据类型转换时SQL语句更容易理解
- 隐式数据类型转换对性能有负作用，当把字段的数据类型转换为常量的数据类型，而不是把常量的数据类型转换为字段的数据类型时负作用更为明显。最常见的就是使用不了索引。
- 隐式转换依赖于它所处的上下文环境并且不是在每种情况下都会以相同方式工作。例如，依赖于nls_date_format参数，将日期时间类型的值隐式转换为varchar2类型值时可能返回一个不是你所期待的年份。
- 隐式转换算法在不同软件版本和Oracle产品中可能会发生改变。显式转换的行为更可控。

ACOUG
All China Oracle User Group
中国 Oracle 用户组

Q&A