CHEATING YOUR APPLICATION CODE WITH ORACLE DATABASE

BE A CHEATER!

Franky Weber Faust
June 2020

Pythian

LUXOUG - COMMUNITY

FRANKY WEBER FAUST

Solutions Architect - Associate

Oracle Autonomous Database Cloud 2019 Certified Specialist

Issued by Oracle

Oracle Certified Expert, Oracle Database 12c: Performance Management and Tuning

Issued by Oracle

Oracle Exadata Database Machine and Cloud Service 2017 Certified Implementation Specialist

Issued by Oracle

Oracle Database 12c Administrator Certified Professional

Issued by Oracle

Oracle Linux 6 Certified Implementation Specialist

Issued by Oracle

Oracle Real Application Clusters 12c Certified Implementation Specialist

Issued by Oracle

Oracle Database SQL Certified Expert

Issued by Oracle

Oracle Database 11g Administrator Certified Associate

Issued by Oracle

ORACLE ACE

- Senior Oracle Database Consultant at Pythian
- Exadata Trainer at Lore Data
- 29 years old
- Writer at OTNLA and Lore Data Blog
- GUOB Board member
- Speaker at conferences around the world
  - São Paulo/Brazil (2016-2020)
  - Borovets/Bulgaria (2018)
  - Vancouver/Canada (2019)
  - Santiago/Chile (2019)
  - Wroclaw/Poland (2019)
  - Pravets/Bulgaria (2019)
- High Availability specialist
- Performance researcher
- Exadata, RAC, DataGuard, GoldenGate
- AcroYogi
- Guitar player

loredata.com.br


tatiaraujo.arq
@CORPORISACRO
@MPANETONE
@FRANKYWEBER
COSTA DA LAGOA


WASHINGTON, DISTRICT OF COLUMBIA
9°C
@TATIARAUJO.ARQ
@CORPORISACRO


INUKSUK - VANCOUVER, BC
@tatiaraujo.arq

Keep in touch

E-mail: faust@pythian.com or franky@loredata.com.br

Blog: http://loredata.com.br/blog

Facebook: https://facebook.com/08Franky.Weber

Instagram: https://www.instagram.com/frankyweber/

Twitter: https://twitter.com/frankyweber

LinkedIn: https://linkedin.com/in/frankyweber/en

Oracle ACE: https://bit.ly/2YxU6bK

450+ Technical Experts Helping Peers Globally

ORACLE ACE PROGRAM

---

bit.ly/OracleACEProgram

Nominate someone you know: **acenomination.oracle.com**

Pythian

LOVE YOUR DATA

22

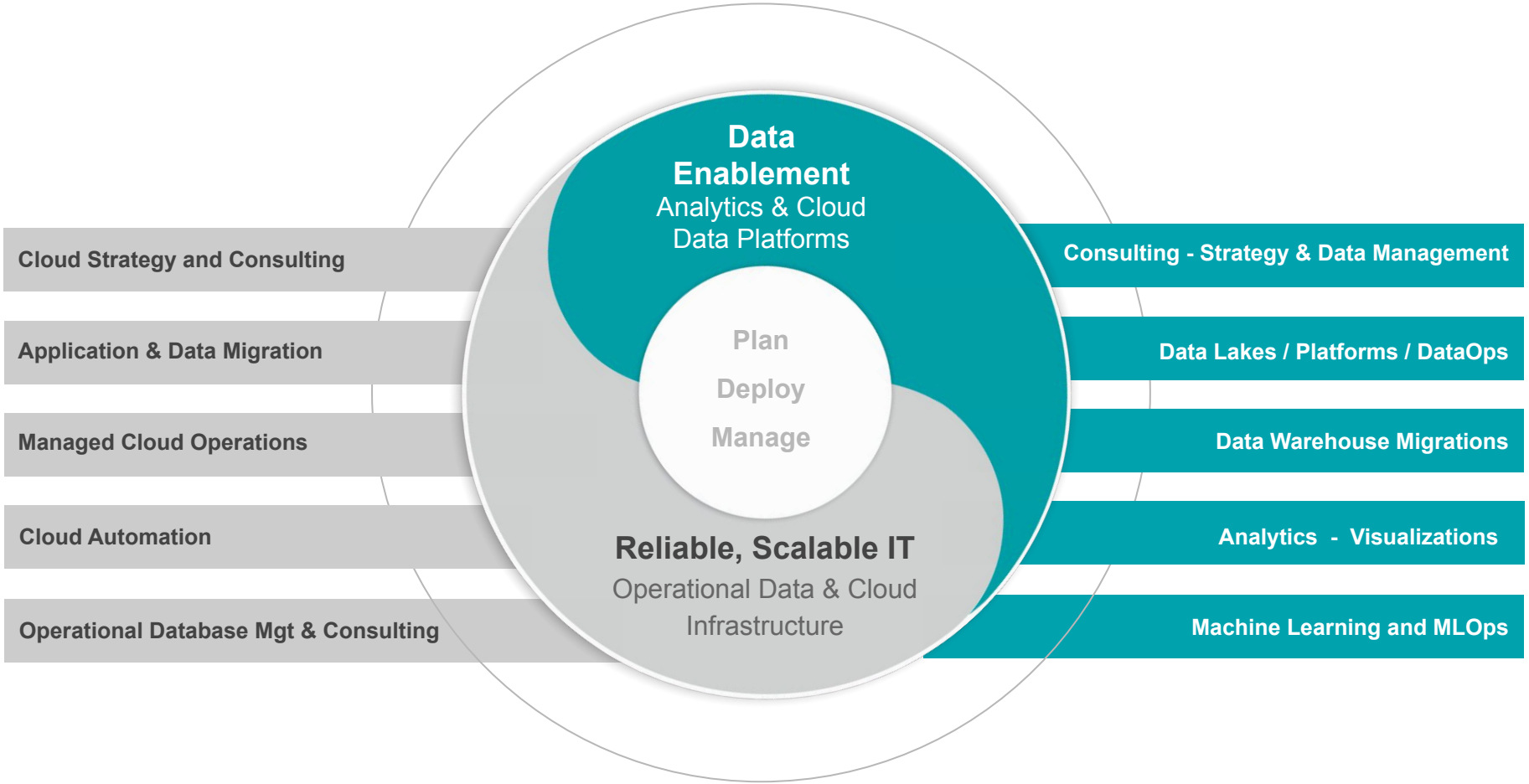
Years in Business

400+

Experts in 35 Countries

350+

Clients Globally



WHY THIS PRESENTATION?

Move to the Cloud!

AGENDA

- SQL Optimization Techniques
- Tune the application client
- *Connect / Commit / Binds

What is available to help you to cheat your code

- Cursor Sharing
- Commit wait/nowait
- Memoptimized Rowstore Fast Ingest
- Memoptimized Rowstore Fast Lookups
- Advanced Rewrite
- Materialized View Rewrite
- SQL Profiles
- SQL Patches
- Parallel Execution
- Result Cache
- Baselines (SPM)
- Advanced Compression / HCC
- Partitioning
- FBI (with nulls for rare values)
- Reverse Key Index
- Clustering Factor
- Append / NoLogging
- Text Index instead of B-Tree for like clauses
- Dynamic Sampling
- In-Memory Column Store

Optimization Techniques (what I ended up with)

- Alter the access structure
- Alter the environment
- Alter the SQL statement
- SQL Rewrite
- Hints
- Stored outlines
- SQL profiles
- SPM
- Compression

Alter the access structure

- Indexes
  - Function Based Index
  - Reverse Key
  - Bitmap
- MViews
- Partitioning

```
PLAN_TABLE_OUTPUT
SQL_ID  7w5rjb4d2sbg8, child number 1
-------------------------------------
select customer_id, cust_first_name, cust_last_name, cust_email from
customers where cust_first_name='reyes'

Plan hash value: 3298762533

-----------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                           | Name              | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers |
-----------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |                   |      1 |        |       | 5691 (100)|     44 |00:00:00.01 |      48 |
|   1 |  TABLE ACCESS BY INDEX ROWID BATCHED| CUSTOMERS         |      1 |   6093 |   273K|  5691   (1)|     44 |00:00:00.01 |      48 |
|*  2 |   INDEX RANGE SCAN                  | IDX_CUSTFIRSTNAME |      1 |   6093 |       |    18   (0)|     44 |00:00:00.01 |       4 |
-----------------------------------------------------------------------------------------------------------------------------------
```

Alter the access structure

- Indexes
  - Function Based Index
  - Reverse Key
  - Bitmap
- MViews
- Partitioning

```
PLAN_TABLE_OUTPUT
SQL_ID  9ywgw27q113cn, child number 1
-------------------------------------
select customer_id, cust_first_name, cust_last_name, cust_email from
customers where upper(cust_first_name)='REYES'

Plan hash value: 2008213504

-----------------------------------------------------------------------------------------------------------------------
| Id  | Operation         | Name      | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |
-----------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |      1 |        |       | 32423 (100)|     44 |00:00:00.89 |     119K|    119K|
|*  1 |  TABLE ACCESS FULL| CUSTOMERS |      1 |  70671 |  3174K| 32423   (1)|     44 |00:00:00.89 |     119K|    119K|
-----------------------------------------------------------------------------------------------------------------------
```

Alter the access structure

- Indexes
  - Function Based Index
  - Reverse Key
  - Bitmap
- MViews
- Partitioning

```
PLAN_TABLE_OUTPUT
SQL_ID  9ywgw27q113cn, child number 1
-------------------------------------
select customer_id, cust_first_name, cust_last_name, cust_email from
customers where upper(cust_first_name)='REYES'

Plan hash value: 1905667729

-----------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                           | Name               | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |
-----------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |                    |      1 |        |       | 26338 (100)|     44 |00:00:00.01 |      48 |      2 |
|   1 |  TABLE ACCESS BY INDEX ROWID BATCHED| CUSTOMERS          |      1 |  70671 |  4692K| 26338   (1)|     44 |00:00:00.01 |      48 |      2 |
|*  2 |   INDEX RANGE SCAN                  | FBIDX_CUSTFIRSTNAME|      1 |  28268 |       |    19   (0)|     44 |00:00:00.01 |       4 |      2 |
-----------------------------------------------------------------------------------------------------------------------------------------
```

Alter the environment

- System level parameters
- Session level parameters
    - Use of logon triggers

```
20:40:15 SQL> select product_information.product_id from product_information
  2  minus
  3  select order_items.product_id from order_items;
  PRODUCT_ID
         991

Elapsed: 00:00:08.184
20:40:24 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  d60uuzp96hbb9, child number 1
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 2679511764
                                                                    ?
------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name                | Starts | A-Rows |   A-Time    | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                     |      1 |      1 |00:00:08.18 |    257K|    253K|       |       |          |
|   1 |  MINUS                |                     |      1 |      1 |00:00:08.18 |    257K|    253K|       |       |          |
|   2 |   SORT UNIQUE         |                     |      1 |   1000 |00:00:00.01 |     31 |      0 | 55296 | 55296 |49152  (0)|
|   3 |    TABLE ACCESS FULL| PRODUCT_INFORMATION |      1 |   1000 |00:00:00.01 |     31 |      0 |       |       |          |
|   4 |   SORT UNIQUE         |                     |      1 |    999 |00:00:08.18 |    257K|    253K| 52224 | 52224 |47104  (0)|
|   5 |    TABLE ACCESS FULL| ORDER_ITEMS         |      1 |    30M|00:00:01.93 |    257K|    253K|       |       |          |
------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      RBO_OUTLINE
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      FULL(@"SEL$2" "ORDER_ITEMS"@"SEL$2")
      FULL(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1")
      END_OUTLINE_DATA
  */

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

   1 - STRDEF[22]
   2 - (#keys=1) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   3 - (rowset=256) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   4 - (#keys=1) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
   5 - (rowset=256) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
20:18:39 SQL> select /*+ full(product_information) */ product_information.product_id from product_information
  2  minus
  3  select /*+ full(order_items) */ order_items.product_id from order_items;
PRODUCT_ID
       991

Elapsed: 00:00:08.121
20:18:58 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  0ztda1y1ngpxn, child number 0
-------------------------------------
select /*+ full(product_information) */ product_information.product_id
from product_information minus select /*+ full(order_items) */
order_items.product_id from order_items

Plan hash value: 2679511764

-----------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation           | Name                | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |                     |      1 |        |       |       |   214K(100)|      1 |00:00:08.12 |     257K|    253K|       |       |          |
|   1 |  MINUS              |                     |      1 |        |       |       |            |      1 |00:00:08.12 |     257K|    253K|       |       |          |
|   2 |   SORT UNIQUE       |                     |      1 |   1000 |  4000 |       |    10  (10)|   1000 |00:00:00.01 |      30 |      0 | 55296 | 55296 |49152  (0)|
|   3 |    TABLE ACCESS FULL| PRODUCT_INFORMATION |      1 |   1000 |  4000 |       |     9   (0)|   1000 |00:00:00.01 |      30 |      0 |       |       |          |
|   4 |   SORT UNIQUE       |                     |      1 |    30M|   379M|   585M|   214K  (1)|    999 |00:00:08.12 |     257K|    253K| 52224 | 52224 |47104  (0)|
|   5 |    TABLE ACCESS FULL| ORDER_ITEMS         |      1 |    30M|   379M|       | 68852   (1)|    30M|00:00:01.77 |     257K|    253K|       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      FULL(@"SEL$2" "ORDER_ITEMS"@"SEL$2")
      FULL(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1")
      END_OUTLINE_DATA
  */

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

   1 - STRDEF[22]
   2 - (#keys=1) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   3 - (rowset=256) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   4 - (#keys=1) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
   5 - (rowset=256) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
20:38:23 SQL> select product_information.product_id from product_information
  2  minus
  3  select order_items.product_id from order_items;
PRODUCT_ID
       991

Elapsed: 00:00:08.798
20:38:32 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  d60uuzp96hbb9, child number 0
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3870675128

-----------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation              | Name                  | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |                       |      1 |        |       |       |   163K(100)|      1 |00:00:08.79 |   65838 |  65452 |       |       |          |
|   1 |  MINUS                 |                       |      1 |        |       |       |            |      1 |00:00:08.79 |   65838 |  65452 |       |       |          |
|   2 |   SORT UNIQUE          |                       |      1 |   1000 |  4000 |       |     3  (34)|   1000 |00:00:00.01 |       6 |      1 | 48128 | 48128 |43008  (0)|
|   3 |    INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |       |     2   (0)|   1000 |00:00:00.01 |       6 |      1 |       |       |          |
|   4 |   SORT UNIQUE          |                       |      1 |    30M|   379M|   586M|   163K  (1)|    999 |00:00:08.79 |   65832 |  65451 | 57344 | 57344 |51200  (0)|
|   5 |    INDEX FAST FULL SCAN| ITEM_PRODUCT_IX       |      1 |    30M|   379M|       | 17342   (1)|     30M|00:00:03.04 |   65832 |  65451 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      INDEX_FFS(@"SEL$2" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      INDEX_FFS(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      END_OUTLINE_DATA
  */

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

   1 - STRDEF[22]
   2 - (#keys=1) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   3 - "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   4 - (#keys=1) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
   5 - "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
21:02:12 SQL> select product_information.product_id from product_information
  2  minus
  3  select order_items.product_id from order_items;
PRODUCT_ID
       991

Elapsed: 00:00:08.761
21:02:25 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  d60uuzp96hbb9, child number 0
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3870675128

-----------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation              | Name                  | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |                       |      1 |        |       |       |   163K(100)|      1 |00:00:08.76 |   65813 |  65434 |       |       |          |
|   1 |  MINUS                 |                       |      1 |        |       |       |            |      1 |00:00:08.76 |   65813 |  65434 |       |       |          |
|   2 |   SORT UNIQUE          |                       |      1 |   1000 |  4000 |       |     3  (34)|   1000 |00:00:00.01 |       6 |      1 | 48128 | 48128 |43008  (0)|
|   3 |    INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |       |     2   (0)|   1000 |00:00:00.01 |       6 |      1 |       |       |          |
|   4 |   SORT UNIQUE          |                       |      1 |    30M|   379M|   586M|   163K  (1)|    999 |00:00:08.76 |   65807 |  65433 | 57344 | 57344 |51200  (0)|
|   5 |    INDEX FAST FULL SCAN| ITEM_PRODUCT_IX       |      1 |    30M|   379M|       | 17342   (1)|     30M|00:00:03.05 |   65807 |  65433 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      FIRST_ROWS(1)
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      INDEX_FFS(@"SEL$2" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      INDEX_FFS(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      END_OUTLINE_DATA
  */

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

   1 - STRDEF[22]
   2 - (#keys=1) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   3 - "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   4 - (#keys=1) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
   5 - "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
21:31:17 SQL> select product_information.product_id from product_information
  2  minus
  3  select order_items.product_id from order_items;
PRODUCT_ID
       991

Elapsed: 00:00:08.474
21:31:26 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  d60uuzp96hbb9, child number 6
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3870675128

-----------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation              | Name                  | Starts | E-Rows |E-Bytes|E-Temp | Cost  | A-Rows |   A-Time    | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |                       |      1 |        |       |       |   164K|      1 |00:00:08.47 |   66281 |  65796 |       |       |          |
|   1 |  MINUS                 |                       |      1 |        |       |       |       |      1 |00:00:08.47 |   66281 |  65796 |       |       |          |
|   2 |   SORT UNIQUE          |                       |      1 |   1000 | 13000 |       |    10 |   1000 |00:00:00.01 |       6 |      1 | 48128 | 48128 |43008  (0)|
|   3 |    INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 | 13000 |       |     1 |   1000 |00:00:00.01 |       6 |      1 |       |       |          |
|   4 |   SORT UNIQUE          |                       |      1 |     30M|   378M|  1169M|   164K|    999 |00:00:08.47 |   66275 |  65795 | 57344 | 57344 |51200  (0)|
|   5 |    INDEX FAST FULL SCAN| ITEM_PRODUCT_IX       |      1 |     30M|   378M|       |  9628 |     30M|00:00:02.94 |   66275 |  65795 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('8.0.7')
      DB_VERSION('19.1.0')
      OPT_PARAM('_optimizer_aggr_groupby_elim' 'true')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      INDEX_FFS(@"SEL$2" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      INDEX_FFS(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      END_OUTLINE_DATA
  */

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

   1 - STRDEF[22]
   2 - (#keys=1) "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   3 - "PRODUCT_INFORMATION"."PRODUCT_ID"[NUMBER,22]
   4 - (#keys=1) "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
   5 - "ORDER_ITEMS"."PRODUCT_ID"[NUMBER,22]
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
PLAN_TABLE_OUTPUT
SQL_ID  1n0rp5zjghpxu, child number 2
-------------------------------------
SELECT TT.ORDER_TOTAL, TT.SALES_REP_ID, TT.ORDER_DATE,
CUSTOMERS.CUST_FIRST_NAME, CUSTOMERS.CUST_LAST_NAME FROM (  SELECT
ORDERS.ORDER_TOTAL, ORDERS.SALES_REP_ID, ORDERS.ORDER_DATE,
ORDERS.CUSTOMER_ID, RANK() OVER (   ORDER BY ORDERS.ORDER_TOTAL DESC)
SAL_RANK FROM     ORDERS WHERE ORDERS.SALES_REP_ID = 287 ) TT,
CUSTOMERS WHERE TT.SAL_RANK <= 10 AND CUSTOMERS.CUSTOMER_ID =
TT.CUSTOMER_ID

Plan hash value: 3619984409

-----------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name         | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |              |      1 |        |       |       | 72448 (100)|    555 |00:00:02.84 |    147K|    145K|       |       |          |
|   1 |  NESTED LOOPS                |              |      1 |   9882 |  1109K|       | 72448   (1)|    555 |00:00:02.84 |    147K|    145K|       |       |          |
|   2 |   NESTED LOOPS               |              |      1 |        |       |       |            |    555 |00:00:02.84 |    146K|    145K|       |       |          |
|*  3 |    VIEW                      |              |      1 |   9839 |   624K|       | 40033   (1)|    555 |00:00:02.83 |    145K|    145K|       |       |          |
|*  4 |     WINDOW SORT PUSHED RANK  |              |      1 |   9839 |   326K|   472K| 40033   (1)|   3554 |00:00:02.83 |    145K|    145K|   302K|   302K|  268K (0)|
|*  5 |      TABLE ACCESS FULL       | ORDERS       |      1 |   9839 |   326K|       | 39942   (1)|   5808 |00:00:02.81 |    145K|    145K|       |       |          |
|*  6 |    INDEX UNIQUE SCAN         | CUSTOMERS_PK |    555 |        |       |       |            |    555 |00:00:00.01 |    1146 |      0 |       |       |          |
|   7 |   TABLE ACCESS BY INDEX ROWID| CUSTOMERS    |    555 |      1 |    50 |       | 32386   (1)|    555 |00:00:00.01 |    555 |      0 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      INDEX(@"SEL$1" "CUSTOMERS"@"SEL$1" ("CUSTOMERS"."CUSTOMER_ID"))
      NLJ_BATCHING(@"SEL$1" "CUSTOMERS"@"SEL$1")
      USE_NL(@"SEL$1" "CUSTOMERS"@"SEL$1")
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      OPT_PARAM('optimizer_index_cost_adj' 500)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SEL$1")
      NO_ACCESS(@"SEL$1" "TT"@"SEL$1")
      LEADING(@"SEL$1" "TT"@"SEL$1" "CUSTOMERS"@"SEL$1")
      FULL(@"SEL$2" "ORDERS"@"SEL$2")
      END_OUTLINE_DATA
  */
```

Alter the environment

- System level parameters
- Session level parameters
  - Use of logon triggers

```
PLAN_TABLE_OUTPUT
SQL_ID  1n0rp5zjghpxu, child number 3
-------------------------------------
SELECT TT.ORDER_TOTAL, TT.SALES_REP_ID, TT.ORDER_DATE,
CUSTOMERS.CUST_FIRST_NAME, CUSTOMERS.CUST_LAST_NAME FROM (  SELECT
ORDERS.ORDER_TOTAL, ORDERS.SALES_REP_ID, ORDERS.ORDER_DATE,
ORDERS.CUSTOMER_ID, RANK() OVER (   ORDER BY ORDERS.ORDER_TOTAL DESC)
SAL_RANK FROM     ORDERS WHERE ORDERS.SALES_REP_ID = 287 ) TT,
CUSTOMERS WHERE TT.SAL_RANK <= 10 AND CUSTOMERS.CUSTOMER_ID =
TT.CUSTOMER_ID

Plan hash value: 1055577880

-----------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                              | Name             | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                       |                  |      1 |        |       |       |  5761 (100)|    555 |00:00:00.04 |    7390 |       |       |          |
|   1 |  NESTED LOOPS                          |                  |      1 |   9839 |  1104K|       |  5761   (1)|    555 |00:00:00.04 |    7390 |       |       |          |
|   2 |   NESTED LOOPS                         |                  |      1 |   9839 |  1104K|       |  5761   (1)|    555 |00:00:00.04 |    6835 |       |       |          |
|*  3 |    VIEW                                |                  |      1 |   9839 |   624K|       |  2808   (1)|    555 |00:00:00.03 |    5689 |       |       |          |
|*  4 |     WINDOW SORT PUSHED RANK            |                  |      1 |   9839 |   326K|  472K|  2808   (1)|   3554 |00:00:00.03 |    5689 |  337K|  337K|  299K (0)|
|   5 |      TABLE ACCESS BY INDEX ROWID BATCHED| ORDERS           |      1 |   9839 |   326K|       |  2717   (1)|   5808 |00:00:00.03 |    5689 |       |       |          |
|*  6 |       INDEX RANGE SCAN                 | ORD_SALES_REP_IX |      1 |   9839 |       |       |     7   (0)|   5808 |00:00:00.01 |      15 |       |       |          |
|*  7 |    INDEX UNIQUE SCAN                   | CUSTOMERS_PK     |    555 |      1 |       |       |     1   (0)|    555 |00:00:00.01 |    1146 |       |       |          |
|   8 |   TABLE ACCESS BY INDEX ROWID          | CUSTOMERS        |    555 |      1 |    50 |       |     1   (0)|    555 |00:00:00.01 |     555 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      OPT_PARAM('optimizer_index_cost_adj' 30)
      OPT_PARAM('optimizer_index_caching' 50)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SEL$1")
      NO_ACCESS(@"SEL$1" "TT"@"SEL$1")
      INDEX(@"SEL$1" "CUSTOMERS"@"SEL$1" ("CUSTOMERS"."CUSTOMER_ID"))
      LEADING(@"SEL$1" "TT"@"SEL$1" "CUSTOMERS"@"SEL$1")
      USE_NL(@"SEL$1" "CUSTOMERS"@"SEL$1")
      NLJ_BATCHING(@"SEL$1" "CUSTOMERS"@"SEL$1")
      INDEX_RS_ASC(@"SEL$2" "ORDERS"@"SEL$2" ("ORDERS"."SALES_REP_ID"))
      BATCH_TABLE_ACCESS_BY_ROWID(@"SEL$2" "ORDERS"@"SEL$2")
      END_OUTLINE_DATA
  */
```

Alter the statement

- Find all products never ordered

```
22:09:07 SQL> select product_information.product_id from product_information minus select order_items.product_id from order_items;

PRODUCT_ID
----------
       991

Elapsed: 00:00:08.590
22:11:35 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  197g6p1tq1mcc, child number 0
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3870675128

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation              | Name                    | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |                         |      1 |        |       |       |  164K(100)|      1 |00:00:08.58 |   66479 |  66429 |       |       |          |
|   1 |  MINUS                 |                         |      1 |        |       |       |           |      1 |00:00:08.58 |   66479 |  66429 |       |       |          |
|   2 |   SORT UNIQUE          |                         |      1 |   1000 |  4000 |       |     3  (34)|   1000 |00:00:00.01 |       6 |      1 | 48128 | 48128 |43008  (0)|
|   3 |    INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK  |      1 |   1000 |  4000 |       |     2   (0)|   1000 |00:00:00.01 |       6 |      1 |       |       |          |
|   4 |   SORT UNIQUE          |                         |      1 |    30M |   381M|   589M|   164K  (1)|    999 |00:00:08.58 |   66473 |  66428 | 57344 | 57344 |51200  (0)|
|   5 |    INDEX FAST FULL SCAN| ITEM_PRODUCT_IX         |      1 |    30M |   381M|       | 17342   (1)|    30M|00:00:03.34 |   66473 |  66428 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      INDEX_FFS(@"SEL$2" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      INDEX_FFS(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      END_OUTLINE_DATA
  */
```

Alter the statement

- Find all products never ordered

```
22:11:35 SQL> select product_id from product_information where product_id not in (select product_id from order_items);

PRODUCT_ID
       991


Elapsed: 00:00:00.022
22:13:11 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  831a208wap3h7, child number 0
-------------------------------------
select product_id from product_information where product_id not in
(select product_id from order_items)

Plan hash value: 3009100470

-----------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name                  | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |
-----------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                       |      1 |        |       |  2003 (100)|      1 |00:00:00.01 |    2009 |     40 |
|   1 |  NESTED LOOPS ANTI    |                       |      1 |     10 |    80 |  2003   (1)|      1 |00:00:00.01 |    2009 |     40 |
|   2 |   INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |     2   (0)|   1000 |00:00:00.01 |       6 |      1 |
|*  3 |   INDEX RANGE SCAN    | ITEM_PRODUCT_IX       |   1000 |     30M|   117M|     2   (0)|    999 |00:00:00.01 |    2003 |     39 |
-----------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$5DA710D3")
      UNNEST(@"SEL$2")
      OUTLINE(@"SEL$1")
      OUTLINE(@"SEL$2")
      INDEX_FFS(@"SEL$5DA710D3" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      INDEX(@"SEL$5DA710D3" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      LEADING(@"SEL$5DA710D3" "PRODUCT_INFORMATION"@"SEL$1" "ORDER_ITEMS"@"SEL$2")
      USE_NL(@"SEL$5DA710D3" "ORDER_ITEMS"@"SEL$2")
      END_OUTLINE_DATA
  */
```

Alter the statement

- Find all products never ordered

```
22:15:01 SQL> select product_id from product_information where not exists (select 1 from order_items where product_information.product_id=order_items.product_id);

PRODUCT_ID
----------
       991

Elapsed: 00:00:00.019
22:15:01 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------------------------------------------------------------------
SQL_ID  dwx4yjakg6jny, child number 0
-------------------------------------
select product_id from product_information where not exists (select 1
from order_items where product_information.product_id=order_items.produc
t_id)

Plan hash value: 3009100470

-------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name                  | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers |
-------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                       |      1 |        |       | 2003 (100)|      1 |00:00:00.01 |    2009 |
|   1 |  NESTED LOOPS ANTI    |                       |      1 |     10 |    80 |  2003   (1)|      1 |00:00:00.01 |    2009 |
|   2 |   INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |     2   (0)|   1000 |00:00:00.01 |       6 |
|*  3 |   INDEX RANGE SCAN    | ITEM_PRODUCT_IX       |   1000 |    30M|   117M|     2   (0)|    999 |00:00:00.01 |    2003 |
-------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$5DA710D3")
      UNNEST(@"SEL$2")
      OUTLINE(@"SEL$1")
      OUTLINE(@"SEL$2")
      INDEX_FFS(@"SEL$5DA710D3" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      INDEX(@"SEL$5DA710D3" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      LEADING(@"SEL$5DA710D3" "PRODUCT_INFORMATION"@"SEL$1" "ORDER_ITEMS"@"SEL$2")
      USE_NL(@"SEL$5DA710D3" "ORDER_ITEMS"@"SEL$2")
      END_OUTLINE_DATA
  */
```

Alter the statement

- Find all products never ordered

```
22:17:06 SQL> SELECT pi.product_id
  2  FROM product_information pi, order_items oi
  3  WHERE pi.product_id = oi.product_id(+) AND oi.product_id IS NULL;

PRODUCT_ID
       991

Elapsed: 00:00:00.014
22:17:23 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  75q0qv07870vk, child number 0
-------------------------------------
SELECT pi.product_id FROM product_information pi, order_items oi WHERE
pi.product_id = oi.product_id(+) AND oi.product_id IS NULL

Plan hash value: 3009100470

-----------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name                  | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers |
-----------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                       |      1 |        |       | 2003 (100)|      1 |00:00:00.01 |    2010 |
|   1 |  NESTED LOOPS ANTI    |                       |      1 |     10 |    80 |  2003   (1)|      1 |00:00:00.01 |    2010 |
|   2 |   INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |     2   (0)|   1000 |00:00:00.01 |       7 |
|*  3 |   INDEX RANGE SCAN    | ITEM_PRODUCT_IX       |   1000 |     30M|   117M|     2   (0)|    999 |00:00:00.01 |    2003 |
-----------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$164C9CD5")
      OUTER_JOIN_TO_ANTI(@"SEL$1" "OI"@"SEL$1")
      OUTLINE(@"SEL$1")
      INDEX_FFS(@"SEL$164C9CD5" "PI"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      INDEX(@"SEL$164C9CD5" "OI"@"SEL$1" ("ORDER_ITEMS"."PRODUCT_ID"))
      LEADING(@"SEL$164C9CD5" "PI"@"SEL$1" "OI"@"SEL$1")
      USE_NL(@"SEL$164C9CD5" "OI"@"SEL$1")
      END_OUTLINE_DATA
  */
```

Not possible to change the SQL

- How can you make this query run faster without changing it?

```
22:09:07 SQL> select product_information.product_id from product_information minus select order_items.product_id from order_items;
PRODUCT_ID
       991


Elapsed: 00:00:08.590
22:11:35 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  197g6p1tq1mcc, child number 0
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3870675128

-------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation              | Name                   | Starts  | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem
-------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT       |                        |       1 |        |       |       |  164K(100)|       1 |00:00:08.58 |   66479 |  66429 |       |       |
|   1 |  MINUS                 |                        |       1 |        |       |       |           |       1 |00:00:08.58 |   66479 |  66429 |       |       |
|   2 |   SORT UNIQUE          |                        |       1 |   1000 |  4000 |       |     3  (34)|    1000 |00:00:00.01 |       6 |      1 | 48128 | 48128 |43008  (0)
|   3 |    INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |       1 |   1000 |  4000 |       |     2   (0)|    1000 |00:00:00.01 |       6 |      1 |       |       |
|   4 |   SORT UNIQUE          |                        |       1 |    30M|   381M|   589M|  164K   (1)|     999 |00:00:08.58 |   66473 |  66428 | 57344 | 57344 |51200  (0)
|   5 |    INDEX FAST FULL SCAN| ITEM_PRODUCT_IX        |       1 |    30M|   381M|       | 17342   (1)|     30M|00:00:03.34 |   66473 |  66428 |       |       |
-------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      OUTLINE_LEAF(@"SEL$2")
      OUTLINE_LEAF(@"SET$1")
      INDEX_FFS(@"SEL$2" "ORDER_ITEMS"@"SEL$2" ("ORDER_ITEMS"."PRODUCT_ID"))
      INDEX_FFS(@"SEL$1" "PRODUCT_INFORMATION"@"SEL$1" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      END_OUTLINE_DATA
  */
```

Not possible to change the SQL

- Same query but running a lot faster!

```
                                DATABASE_SPEED = FASTEST
22:21:49 SQL> ALTER SESSION SET QUERY_REWRITE_INTEGRITY = TRUSTED;

Session altered.

Elapsed: 00:00:00.003
22:21:49 SQL> select product_information.product_id from product_information minus select order_items.product_id from order_items;
  PRODUCT_ID
         991


Elapsed: 00:00:00.013
```

SQL Rewrite

- dbms_xplan shows the original query, but the plan is different
- Same indexes, but different approach

```
22:21:49 SQL> ALTER SESSION SET QUERY_REWRITE_INTEGRITY = TRUSTED;

Session altered.

Elapsed: 00:00:00.003
22:21:49 SQL> select product_information.product_id from product_information minus select order_items.product_id from order_items;
  PRODUCT_ID
         991


Elapsed: 00:00:00.013
22:21:56 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  197g6p1tq1mcc, child number 0
-------------------------------------
select product_information.product_id from product_information minus
select order_items.product_id from order_items

Plan hash value: 3009100470

-----------------------------------------------------------------------------------------------------------------------------
| Id  | Operation             | Name                  | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers |
-----------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                       |      1 |        |       |  2003 (100)|      1 |00:00:00.01 |    2009 |
|   1 |  NESTED LOOPS ANTI    |                       |      1 |     10 |    80 |  2003   (1)|      1 |00:00:00.01 |    2009 |
|   2 |   INDEX FAST FULL SCAN| PRODUCT_INFORMATION_PK |      1 |   1000 |  4000 |     2   (0)|   1000 |00:00:00.01 |       6 |
|*  3 |   INDEX RANGE SCAN    | ITEM_PRODUCT_IX       |   1000 |     30M|   117M|     2   (0)|    999 |00:00:00.01 |    2003 |
-----------------------------------------------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$CA1FC53F")
      UNNEST(@"SEL$64DA765C")
      OUTLINE(@"SEL$6795A312")
      MERGE(@"SEL$64DA765B" >"SET$1")
      OUTLINE(@"SEL$64DA765C")
      OUTLINE(@"SET$1")
      OUTLINE(@"SEL$64DA765B")
      INDEX_FFS(@"SEL$CA1FC53F" "PRODUCT_INFORMATION"@"SEL$64DA765B" ("PRODUCT_INFORMATION"."PRODUCT_ID"))
      INDEX(@"SEL$CA1FC53F" "ORDER_ITEMS"@"SEL$64DA765C" ("ORDER_ITEMS"."PRODUCT_ID"))
      LEADING(@"SEL$CA1FC53F" "PRODUCT_INFORMATION"@"SEL$64DA765B" "ORDER_ITEMS"@"SEL$64DA765C")
      USE_NL(@"SEL$CA1FC53F" "ORDER_ITEMS"@"SEL$64DA765C")
      END_OUTLINE_DATA
  */
```

SQL Rewrite

- SQL Magic


```
22:19:48 SQL> BEGIN
  2    SYS.DBMS_ADVANCED_REWRITE.declare_rewrite_equivalence (
  3       name             => 'rewrite_pi_oi',
  4       source_stmt      => 'select product_information.product_id from product_information minus select order_items.product_id from order_items',
  5       destination_stmt => 'select product_id from product_information where not exists (select 1 from order_items where product_information.product_id=order_items.product_id)',
  6       validate         => FALSE,
  7       rewrite_mode     => 'RECURSIVE');
  8  END;
  9  /

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.041
22:19:49 SQL> SELECT * FROM user_rewrite_equivalences;
OWNER   NAME            SOURCE_STMT                                                    DESTINATION_STMT                                               REWRITE_MODE
SOE     REWRITE_PI_OI   select product_information.product_id from product_information minus select orde   select product_id from product_information where not exists (select 1 from order   RECURSIVE
```

SQL Rewrite

```
23:04:12 SQL> ALTER SESSION SET QUERY_REWRITE_INTEGRITY = TRUSTED;

Session altered.

Elapsed: 00:00:00.001
23:04:42 SQL> ALTER SESSION SET EVENTS '10053 TRACE NAME CONTEXT FOREVER, LEVEL 1';

Session altered.

Elapsed: 00:00:00.040
23:04:52 SQL> select product_information.product_id from product_information minus select order_items.product_id from order_items;
__PRODUCT_ID
         991


Elapsed: 00:00:00.212
```

SQL Rewrite

```
*******************************************
----- Current SQL Statement for this session (sql_id=197g6p1tq1mcc) -----
select product_information.product_id from product_information minus select order_items.product_id from order_items
*******************************************


************************
Common Subexpression elimination (CSE)
************************
CSE:     CSE not performed on query block SEL$CA1FC53F (#1).
Final query after transformations:******* UNPARSED QUERY IS *******
SELECT "PRODUCT_INFORMATION"."PRODUCT_ID" "PRODUCT_ID" FROM "SOE"."ORDER_ITEMS" "ORDER_ITEMS","SOE"."PRODUCT_INFORMATION" "PRODUCT_INFORMATION" WHERE "PRODUCT_INFORMATION"."PRODUCT_ID"="ORDER_ITEMS"."PRODUCT_ID"
kkoqbc: optimizing query block SEL$CA1FC53F (#1)

        :
    call(in-use=65624, alloc=81816), compile(in-use=220592, alloc=222336), execution(in-use=6768, alloc=8088)

kkoqbc-subheap (create addr=0x7f5c14975ca8)
***************
QUERY BLOCK TEXT
***************
select product_information.product_id from product_information minus select order_items.product_id from order_items
---------------------
QUERY BLOCK SIGNATURE
---------------------
signature (optimizer): qb_name=SEL$CA1FC53F nbfros=2 flg=0
  fro(0): flg=0 objn=23731 hint_alias="PRODUCT_INFORMATION"@"SEL$64DA765B"
  fro(1): flg=0 objn=23728 hint_alias="ORDER_ITEMS"@"SEL$64DA765C"
```

Hints

- I'm smarter than the CBO

```
SQL_ID  7k0yfm3p7qcj0, child number 1
-------------------------------------
select /*+ parallel (4) index(o ORDER_PK) index(oi ITEM_ORDER_IX)
index(a ADDRESS_CUST_IX) index(c CUSTOMERS_PK) */ a.county,
sum((oi.unit_price*oi.quantity)) sales from addresses a, order_items
oi, orders o, customers c where oi.order_id=o.order_id and
o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-1 group by rollup(county) order by 1

Plan hash value: 413483043

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                                | Name             | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | OMem | 1Mem | Used-Mem | Used-Tmp|
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                         |                  |      1 |        |       |       |    11M(100)|     91 |00:02:09.48 |      |      |          |         |
|   1 |  PX COORDINATOR                          |                  |      1 |        |       |       |            |     91 |00:02:09.48 | 31744 | 31744 |28672  (0)|         |
|   2 |   PX SEND QC (ORDER)                     | :TQ10004         |      0 |     90 |  9090 |       |   11M  (1)|      0 |00:00:00.01 |      |      |          |         |
|   3 |    SORT GROUP BY                         |                  |      0 |     90 |  9090 |       |   11M  (1)|      0 |00:00:00.01 |  4096 |  4096 | 4096  (0)|         |
|   4 |     PX RECEIVE                           |                  |      0 |     90 |  9090 |       |   11M  (1)|      0 |00:00:00.01 |      |      |          |         |
|   5 |      PX SEND RANGE                       | :TQ10003         |      0 |     90 |  9090 |       |   11M  (1)|      0 |00:00:00.01 |      |      |          |         |
|   6 |       SORT GROUP BY ROLLUP               |                  |      0 |     90 |  9090 |       |   11M  (1)|      0 |00:00:00.01 |  9216 |  9216 | 8192  (0)|         |
|   7 |        NESTED LOOPS                      |                  |      0 |   492K|    47M|       |   11M  (1)|      0 |00:00:00.01 |      |      |          |         |
|   8 |         NESTED LOOPS                     |                  |      0 |   492K|    47M|       |   11M  (1)|      0 |00:00:00.01 |      |      |          |         |
|   9 |          NESTED LOOPS                    |                  |      0 |   160K|    10M|       |   10M  (1)|      0 |00:00:00.01 |      |      |          |         |
|* 10 |           HASH JOIN                      |                  |      0 |  99659 |  3017K|  3592K|   10M  (1)|      0 |00:00:00.01 |  4451K|  1549K| 3734K (1)|    5120K|
|  11 |            PX RECEIVE                    |                  |      0 |  99225 |  2422K|       |   10M  (1)|      0 |00:00:00.01 |      |      |          |         |
|  12 |             PX SEND HYBRID HASH          | :TQ10001         |      0 |  99225 |  2422K|       |   10M  (1)|      0 |00:00:00.01 |      |      |          |         |
|  13 |              STATISTICS COLLECTOR        |                  |      0 |        |       |       |            |      0 |00:00:00.01 |      |      |          |         |
|* 14 |               TABLE ACCESS BY INDEX ROWID BATCHED| ORDERS           |      0 |  99225 |  2422K|       |   10M  (1)|      0 |00:00:00.01 |      |      |          |         |
|  15 |                BUFFER SORT               |                  |      0 |        |       |       |            |      0 |00:00:00.01 |   65M|  2805K|   21M (0)|      59M|
|  16 |                 PX RECEIVE               |                  |      0 |    10M|       |       | 22311  (1)|      0 |00:00:00.01 |      |      |          |         |
|  17 |                  PX SEND HASH (BLOCK ADDRESS)| :TQ10000         |      0 |    10M|       |       | 22311  (1)|      0 |00:00:00.01 |      |      |          |         |
|  18 |                   PX SELECTOR            |                  |      0 |        |       |       |            |      0 |00:00:00.01 |      |      |          |         |
|  19 |                    INDEX FULL SCAN       | ORDER_PK         |      0 |    10M|       |       | 22311  (1)|      0 |00:00:00.01 |      |      |          |         |
|  20 |             PX RECEIVE                   |                  |      0 |  7067K|    40M|       | 15701  (1)|      0 |00:00:00.01 |      |      |          |         |
|  21 |              PX SEND HYBRID HASH         | :TQ10002         |      0 |  7067K|    40M|       | 15701  (1)|      0 |00:00:00.01 |      |      |          |         |
|  22 |               PX SELECTOR                |                  |      0 |        |       |       |            |      0 |00:00:00.01 |      |      |          |         |
|  23 |                INDEX FULL SCAN           | CUSTOMERS_PK     |      0 |  7067K|    40M|       | 15701  (1)|      0 |00:00:00.01 |      |      |          |         |
|  24 |            TABLE ACCESS BY INDEX ROWID BATCHED| ADDRESSES        |      0 |      2 |    76 |       |     4  (0)|      0 |00:00:00.01 |      |      |          |         |
|* 25 |             INDEX RANGE SCAN             | ADDRESS_CUST_IX  |      0 |     12 |       |       |     1  (0)|      0 |00:00:00.01 |      |      |          |         |
|* 26 |          INDEX RANGE SCAN                | ITEM_ORDER_IX    |      0 |      3 |       |       |     1  (0)|      0 |00:00:00.01 |      |      |          |         |
|  27 |         TABLE ACCESS BY INDEX ROWID      | ORDER_ITEMS      |      0 |      3 |    96 |       |     1  (0)|      0 |00:00:00.01 |      |      |          |         |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Hints

- I agree with the CBO

```
SQL_ID  fj9p222t0wzg9, child number 0
-------------------------------------
select a.county, sum((oi.unit_price*oi.quantity)) sales from addresses
a, order_items oi, orders o, customers c where oi.order_id=o.order_id
and o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-1 group by rollup(county) order by 1

Plan hash value: 4256304640

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                | Name         | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  | Writes |  OMem |  1Mem | Used-Mem | Used-Tmp|
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |              |      1 |        |       |       |   245K(100)|     91 |00:00:13.52 |     538K|    530K|    7777 |       |       |          |         |
|   1 |  SORT GROUP BY ROLLUP    |              |      1 |     90 |  9090 |       |   245K  (1)|     91 |00:00:13.52 |     538K|    530K|    7777 |  9216 |  9216 | 8192  (0)|         |
|*  2 |   HASH JOIN              |              |      1 |    492K|    47M|    12M|   245K  (1)|    805K|00:00:13.32 |     538K|    530K|    7777 |   27M|  5524K|    19M (1)|      22M|
|*  3 |    HASH JOIN             |              |      1 |    160K|    10M|  4192K|   109K  (2)|    267K|00:00:04.62 |     282K|    271K|    5152 |   14M|  3257K|  8868K (1)|      28M|
|*  4 |     HASH JOIN            |              |      1 |  99659 |  3017K|  3592K| 51105   (2)|    178K|00:00:03.09 |     161K|    147K|    1680 |   13M|  3557K|  7985K (1)|      14M|
|*  5 |      TABLE ACCESS FULL   | ORDERS       |      1 |  99225 |  2422K|       | 40585   (3)|    178K|00:00:01.01 |     145K|    145K|       0 |       |       |          |         |
|   6 |      INDEX FAST FULL SCAN| CUSTOMERS_PK |      1 |   7067K|    40M|       |  4275   (1)|   7067K|00:00:00.66 |   15529 |       0 |       0 |       |       |          |         |
|   7 |     TABLE ACCESS FULL    | ADDRESSES    |      1 |    10M|   382M|       | 32999   (1)|     10M|00:00:00.87 |     121K|    121K|       0 |       |       |          |         |
|   8 |    TABLE ACCESS FULL     | ORDER_ITEMS  |      1 |    30M|   939M|       | 71122   (1)|     30M|00:00:03.21 |     256K|    256K|       0 |       |       |          |         |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Hints

- I don't like FTS

```
SQL_ID  9mrtwf64p9g73, child number 0
-------------------------------------
select /*+ index(a ADDRESS_CUST_IX) */ a.county,
sum((oi.unit_price*oi.quantity)) sales from addresses a, order_items
oi, orders o, customers c where oi.order_id=o.order_id and
o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-1 group by rollup(county) order by 1

Plan hash value: 3674972460

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name            | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  | Writes |  OMem |  1Mem | Used-Mem | Used-Tmp|
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                 |      1 |        |       |       |   544K(100)|     91 |00:00:16.90 |     938K|     483K|    3015 |       |       |          |         |
|   1 |  SORT GROUP BY ROLLUP         |                 |      1 |     90 |  9090 |       |   544K  (1)|     91 |00:00:16.90 |     938K|     483K|    3015 |  9216 |  9216 | 8192  (0)|         |
|*  2 |   HASH JOIN                   |                 |      1 |    492K|    47M|    12M|   544K  (1)|    805K|00:00:16.71 |     938K|     483K|    3015 |   30M|  5211K|   23M (1)|     26M|
|   3 |    NESTED LOOPS               |                 |      1 |    160K|    10M|       |   408K  (1)|    267K|00:00:07.88 |     682K|     224K|       0 |       |       |          |         |
|   4 |     NESTED LOOPS              |                 |      1 |   1195K|    10M|       |   408K  (1)|    267K|00:00:04.96 |     414K|     165K|       0 |       |       |          |         |
|*  5 |      HASH JOIN                |                 |      1 |  99659 |  3017K|  3592K|  51105   (2)|    178K|00:00:03.65 |     161K|     145K|       0 |   13M|  3557K|   12M (0)|         |
|*  6 |       TABLE ACCESS FULL       | ORDERS          |      1 |  99225 |  2422K|       |  40585   (3)|    178K|00:00:01.17 |     145K|     145K|       0 |       |       |          |         |
|   7 |       INDEX FAST FULL SCAN    | CUSTOMERS_PK    |      1 |   7067K|    40M|       |   4275   (1)|   7067K|00:00:00.75 |   15529 |     217 |       0 |       |       |          |         |
|*  8 |      INDEX RANGE SCAN         | ADDRESS_CUST_IX |    178K|     12 |       |       |      2   (0)|    267K|00:00:01.25 |     253K|   19556 |       0 |       |       |          |         |
|   9 |     TABLE ACCESS BY INDEX ROWID| ADDRESSES      |    267K|      2 |     76 |       |     15   (0)|    267K|00:00:02.84 |     267K|   59409 |       0 |       |       |          |         |
|  10 |    TABLE ACCESS FULL          | ORDER_ITEMS     |      1 |     30M|    939M|       | 71122   (1)|     30M|00:00:03.22 |     256K|     256K|       0 |       |       |          |         |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Hints

- We can do better together

```
SQL_ID  45ndrky98hy2q, child number 1
-------------------------------------
select /*+ parallel (4) */ a.county, sum((oi.unit_price*oi.quantity))
sales from addresses a, order_items oi, orders o, customers c where
oi.order_id=o.order_id and o.customer_id=c.customer_id and
a.customer_id=c.customer_id and o.order_date>sysdate-1 group by
rollup(county) order by 1

Plan hash value: 680864913
```

Id	Operation	Name	Starts	E-Rows	E-Bytes	Cost (%CPU)	A-Rows	A-Time	Buffers	OMem	1Mem	Used-Mem	Used-Tmp
0	SELECT STATEMENT		1			41396 (100)	91	00:00:06.72	56				
1	PX COORDINATOR		1				91	00:00:06.72	56	31744	31744	28672 (0)	
2	PX SEND QC (ORDER)	:TQ10005	0	90	9090	41396 (1)	0	00:00:00.01	0				
3	SORT GROUP BY		0	90	9090	41396 (1)	0	00:00:00.01	0	18432	18432	4096 (0)	
4	PX RECEIVE		0	90	9090	41396 (1)	0	00:00:00.01	0				
5	PX SEND RANGE	:TQ10004	0	90	9090	41396 (1)	0	00:00:00.01	0				
6	SORT GROUP BY ROLLUP		0	90	9090	41396 (1)	0	00:00:00.01	0	36864	36864	8192 (0)	
* 7	HASH JOIN		0	492K	47M	41390 (1)	0	00:00:00.01	0	27M	5524K	1733K (1)	6144K
8	JOIN FILTER CREATE	:BF0000	0	160K	10M	21624 (2)	0	00:00:00.01	0				
9	PX RECEIVE		0	160K	10M	21624 (2)	0	00:00:00.01	0				
10	PX SEND HYBRID HASH	:TQ10002	0	160K	10M	21624 (2)	0	00:00:00.01	0				
11	STATISTICS COLLECTOR		0				0	00:00:00.01	0				
* 12	HASH JOIN		0	160K	10M	21624 (2)	0	00:00:00.01	0	54M	6513K	8355K (1)	14M
13	JOIN FILTER CREATE	:BF0001	0	99659	3017K	12457 (2)	0	00:00:00.01	0				
14	PX RECEIVE		0	99659	3017K	12457 (2)	0	00:00:00.01	0				
15	PX SEND BROADCAST	:TQ10001	0	99659	3017K	12457 (2)	0	00:00:00.01	0				
* 16	HASH JOIN		0	99659	3017K	12457 (2)	0	00:00:00.01	0	50M	7114K	12M (0)	
17	PX RECEIVE		0	99225	2422K	11262 (2)	0	00:00:00.01	0				
18	PX SEND BROADCAST	:TQ10000	0	99225	2422K	11262 (2)	0	00:00:00.01	0				
19	PX BLOCK ITERATOR		0	99225	2422K	11262 (2)	0	00:00:00.01	0				
* 20	TABLE ACCESS FULL	ORDERS	0	99225	2422K	11262 (2)	0	00:00:00.01	0				
21	PX BLOCK ITERATOR		0	7067K	40M	1187 (1)	0	00:00:00.01	0				
* 22	INDEX FAST FULL SCAN	CUSTOMERS_PK	0	7067K	40M	1187 (1)	0	00:00:00.01	0				
23	JOIN FILTER USE	:BF0001	0	10M	382M	9157 (1)	0	00:00:00.01	0				
24	PX BLOCK ITERATOR		0	10M	382M	9157 (1)	0	00:00:00.01	0				
* 25	TABLE ACCESS FULL	ADDRESSES	0	10M	382M	9157 (1)	0	00:00:00.01	0				
26	PX RECEIVE		0	30M	939M	19736 (1)	0	00:00:00.01	0				
27	PX SEND HYBRID HASH	:TQ10003	0	30M	939M	19736 (1)	0	00:00:00.01	0				
28	JOIN FILTER USE	:BF0000	0	30M	939M	19736 (1)	0	00:00:00.01	0				
29	PX BLOCK ITERATOR		0	30M	939M	19736 (1)	0	00:00:00.01	0				
* 30	TABLE ACCESS FULL	ORDER_ITEMS	0	30M	939M	19736 (1)	0	00:00:00.01	0				

Hints

- Forcing the index but now with parallel

```
SQL_ID  2chzj7a8y3phu, child number 0
-------------------------------------
select /*+ parallel (4) index(a ADDRESS_CUST_IX) */ a.county,
sum((oi.unit_price*oi.quantity)) sales from addresses a, order_items
oi, orders o, customers c where oi.order_id=o.order_id and
o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-1 group by rollup(county) order by 1

Plan hash value: 2841990537

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                          | Name           | Starts | E-Rows |E-Bytes| Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem | Used-Tmp|
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |                |      1 |        |       |   131K(100)|     91 |00:00:07.38 |      51 |      8 |       |       |          |         |
|   1 |  PX COORDINATOR                    |                |      1 |        |       |            |     91 |00:00:07.38 |      51 |      8 | 31744 | 31744 |28672  (0)|         |
|   2 |   PX SEND QC (ORDER)               | :TQ10004       |      0 |    90 |  9090 |   131K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|   3 |    SORT GROUP BY                   |                |      0 |    90 |  9090 |   131K  (1)|      0 |00:00:00.01 |       0 |      0 | 18432 | 18432 | 4096  (0)|         |
|   4 |     PX RECEIVE                     |                |      0 |    90 |  9090 |   131K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|   5 |      PX SEND RANGE                 | :TQ10003       |      0 |    90 |  9090 |   131K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|   6 |       SORT GROUP BY ROLLUP         |                |      0 |    90 |  9090 |   131K  (1)|      0 |00:00:00.01 |       0 |      0 | 36864 | 36864 | 8192  (0)|         |
|*  7 |        HASH JOIN                   |                |      0 |   492K|   47M|   131K  (1)|      0 |00:00:00.01 |       0 |      0 |   30M|  5211K| 5719K (1)|      14M|
|   8 |         JOIN FILTER CREATE         | :BF0000        |      0 |   160K|   10M|   111K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|   9 |          PX RECEIVE                |                |      0 |   160K|   10M|   111K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  10 |           PX SEND HYBRID HASH      | :TQ10001       |      0 |   160K|   10M|   111K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  11 |            STATISTICS COLLECTOR    |                |      0 |       |       |            |      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  12 |             NESTED LOOPS           |                |      0 |   160K|   10M|   111K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  13 |              NESTED LOOPS          |                |      0 |  1195K|   10M|   111K  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|* 14 |               HASH JOIN            |                |      0 | 99659 |  3017K| 12457  (2)|      0 |00:00:00.01 |       0 |      0 |   50M|  7114K|   12M (0)|         |
|  15 |                PX RECEIVE          |                |      0 | 99225 |  2422K| 11262  (2)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  16 |                 PX SEND BROADCAST  | :TQ10000       |      0 | 99225 |  2422K| 11262  (2)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  17 |                  PX BLOCK ITERATOR |                |      0 | 99225 |  2422K| 11262  (2)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|* 18 |                   TABLE ACCESS FULL| ORDERS         |      0 | 99225 |  2422K| 11262  (2)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  19 |               PX BLOCK ITERATOR    |                |      0 |  7067K|   40M|  1187  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|* 20 |                INDEX FAST FULL SCAN| CUSTOMERS_PK   |      0 |  7067K|   40M|  1187  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|* 21 |              INDEX RANGE SCAN      | ADDRESS_CUST_IX|      0 |    12 |       |     1  (0)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  22 |             TABLE ACCESS BY INDEX ROWID| ADDRESSES      |      0 |     2 |    76 |     4  (0)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  23 |          PX RECEIVE                |                |      0 |   30M|   939M| 19736  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  24 |           PX SEND HYBRID HASH      | :TQ10002       |      0 |   30M|   939M| 19736  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  25 |            JOIN FILTER USE         | :BF0000        |      0 |   30M|   939M| 19736  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|  26 |             PX BLOCK ITERATOR      |                |      0 |   30M|   939M| 19736  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
|* 27 |              TABLE ACCESS FULL     | ORDER_ITEMS    |      0 |   30M|   939M| 19736  (1)|      0 |00:00:00.01 |       0 |      0 |       |       |          |         |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

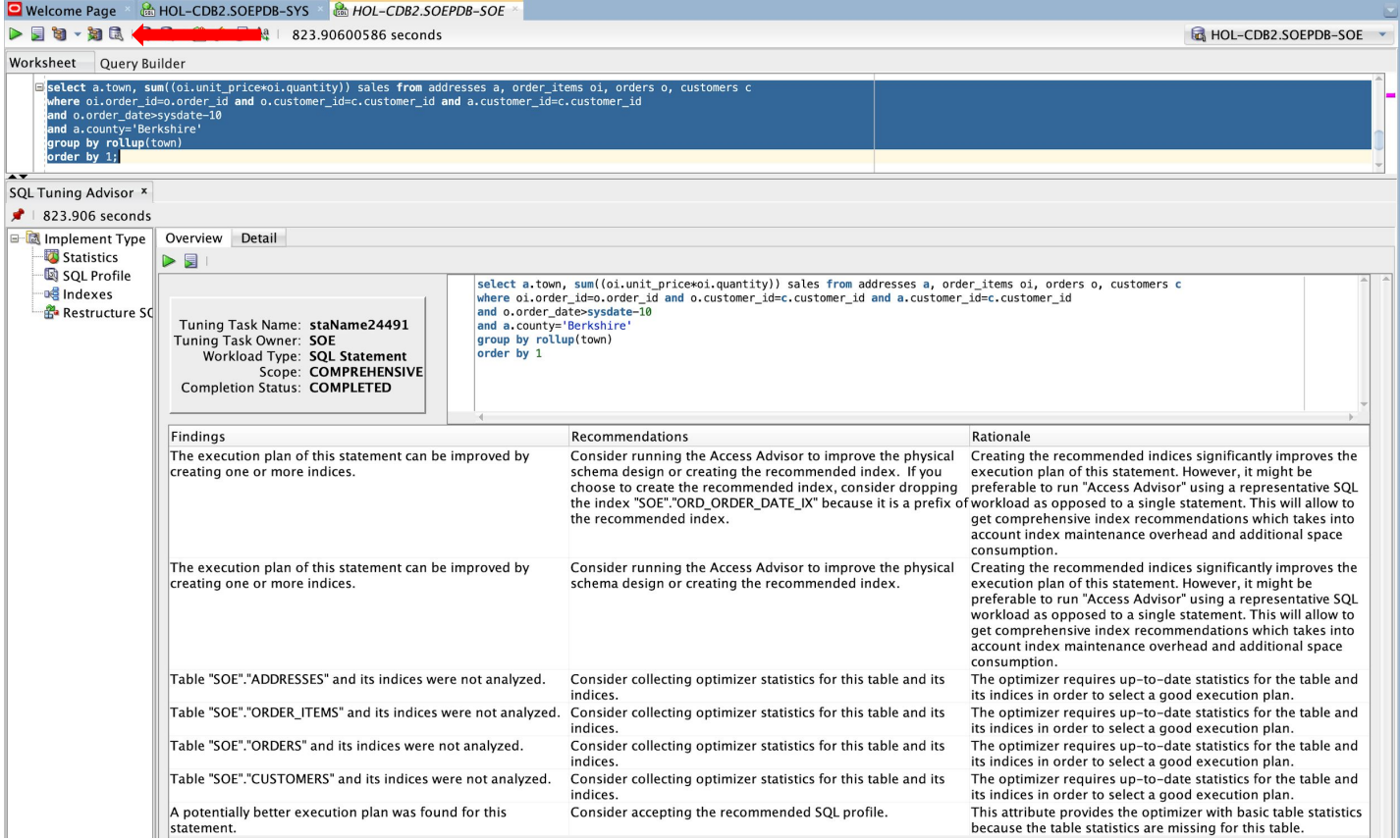
SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)

```
17:11:51 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  5r5u3upcwykp5, child number 4
-------------------------------------
select a.town, sum((oi.unit_price*oi.quantity)) sales from addresses a,
order_items oi, orders o, customers c  where oi.order_id=o.order_id and
o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-10 and a.county='Berkshire' group by rollup(town)
order by 1

Plan hash value: 2185885323

-----------------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name          | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers | Reads  |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |               |      1 |        |       |       | 87876 (100)|     94 |00:00:01.48 |     502K|    266K|       |       |          |
|   1 |  SORT GROUP BY ROLLUP         |               |      1 |  10854 |  1780K|       | 87876   (2)|     94 |00:00:01.48 |     502K|    266K|  9216 |  9216 | 8192  (0)|
|   2 |   NESTED LOOPS                |               |      1 |  10854 |  1780K|       | 87875   (2)|   9060 |00:00:01.48 |     502K|    266K|       |       |          |
|   3 |    NESTED LOOPS               |               |      1 |    195K|  1780K|       | 87875   (2)|   9060 |00:00:01.47 |     502K|    266K|       |       |          |
|*  4 |     HASH JOIN                 |               |      1 |   6100 |   768K|  9872K| 75671   (2)|   2940 |00:00:01.46 |     499K|    266K|    11M|  2905K|   15M (0)|
|   5 |      NESTED LOOPS             |               |      1 |  99073 |  8707K|       | 33019   (1)|    116K|00:00:00.75 |     353K|    121K|       |       |          |
|*  6 |       TABLE ACCESS FULL       | ADDRESSES     |      1 |  99073 |  7449K|       | 33011   (1)|    116K|00:00:00.49 |     121K|    121K|       |       |          |
|*  7 |       INDEX UNIQUE SCAN       | CUSTOMERS_PK  |    116K|      1 |    13 |       |     0   (0)|    116K|00:00:00.23 |     232K|      0 |       |       |          |
|*  8 |      TABLE ACCESS FULL        | ORDERS        |      1 |    599K|    22M|       | 40723   (3)|    178K|00:00:00.62 |     145K|    145K|       |       |          |
|*  9 |     INDEX RANGE SCAN          | ITEM_ORDER_IX |   2940 |     32 |       |       |     1   (0)|   9060 |00:00:00.01 |    2697 |     12 |       |       |          |
|  10 |    TABLE ACCESS BY INDEX ROWID| ORDER_ITEMS   |   9060 |      2 |    78 |       |     2   (0)|   9060 |00:00:00.01 |     273 |     42 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------------

Outline Data
------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      OPT_PARAM('optimizer_dynamic_sampling' 0)
      OPT_PARAM('_optimizer_use_stats_on_conventional_dml' 'false')
      OPT_PARAM('_optimizer_gather_stats_on_conventional_dml' 'false')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      FULL(@"SEL$1" "A"@"SEL$1")
      INDEX(@"SEL$1" "C"@"SEL$1" ("CUSTOMERS"."CUSTOMER_ID"))
      FULL(@"SEL$1" "O"@"SEL$1")
      INDEX(@"SEL$1" "OI"@"SEL$1" ("ORDER_ITEMS"."ORDER_ID"))
      LEADING(@"SEL$1" "A"@"SEL$1" "C"@"SEL$1" "O"@"SEL$1" "OI"@"SEL$1")
      USE_NL(@"SEL$1" "C"@"SEL$1")
      USE_HASH(@"SEL$1" "O"@"SEL$1")
      USE_NL(@"SEL$1" "OI"@"SEL$1")
      NLJ_BATCHING(@"SEL$1" "OI"@"SEL$1")
      END_OUTLINE_DATA
  */
```

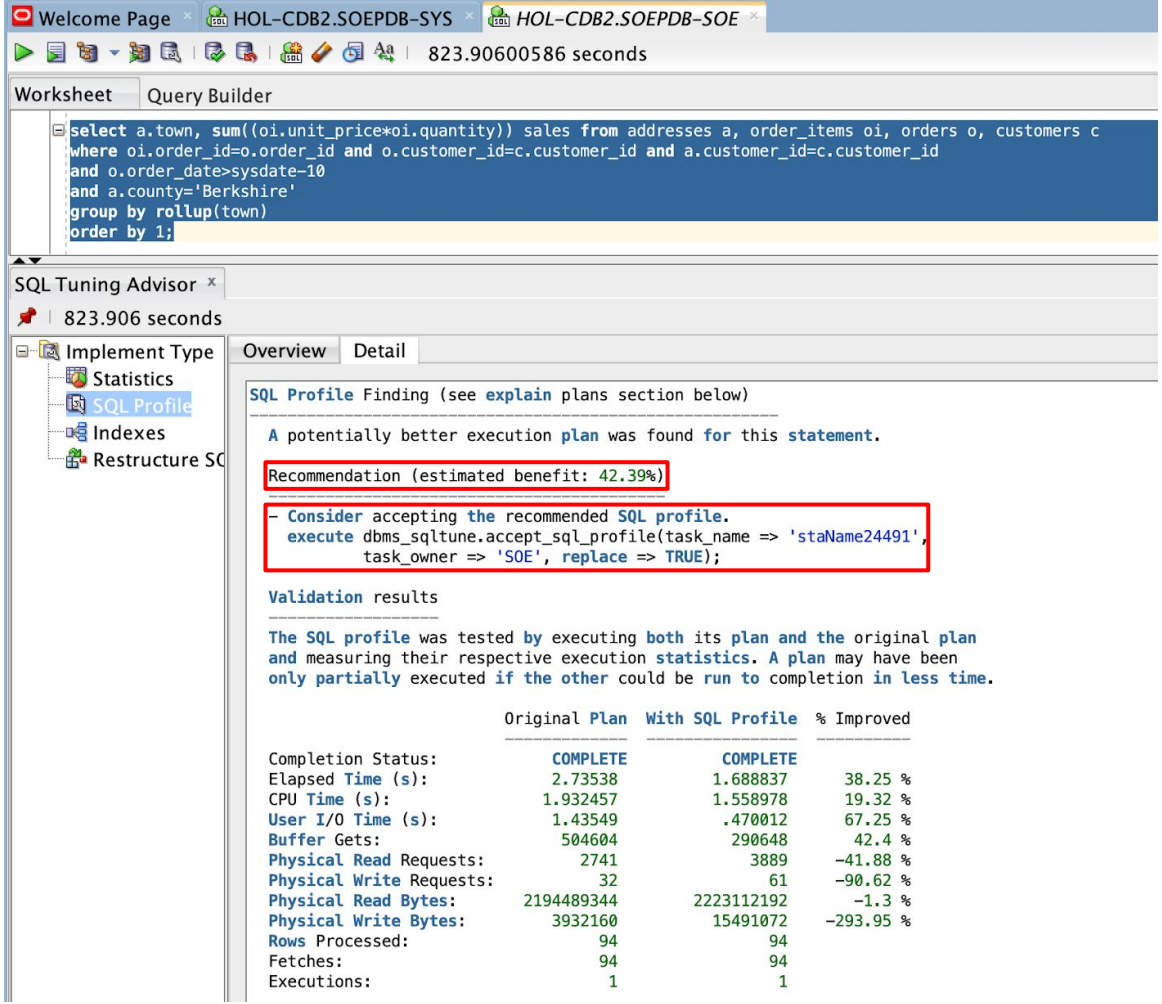
SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)



SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)



SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)

2- Original With Adjusted Cost

Plan hash value: 2185885323

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time
0	SELECT STATEMENT		502	29618		201K (1)	00:00:08
1	SORT GROUP BY ROLLUP		502	29618		201K (1)	00:00:08
2	NESTED LOOPS		9209	530K		201K (1)	00:00:08
3	NESTED LOOPS		9209	530K		201K (1)	00:00:08
* 4	HASH JOIN		3038	145K	4952K	191K (1)	00:00:08
5	NESTED LOOPS		117K	3565K		150K (1)	00:00:06
* 6	TABLE ACCESS FULL	ADDRESSES	117K	2990K		33020 (1)	00:00:02
* 7	INDEX UNIQUE SCAN	CUSTOMERS_PK	1	5		1 (0)	00:00:01
* 8	TABLE ACCESS FULL	ORDERS	182K	3204K		40597 (3)	00:00:02
* 9	INDEX RANGE SCAN	ITEM_ORDER_IX	3			2 (0)	00:00:01
10	TABLE ACCESS BY INDEX ROWID	ORDER_ITEMS	3	30		3 (0)	00:00:01

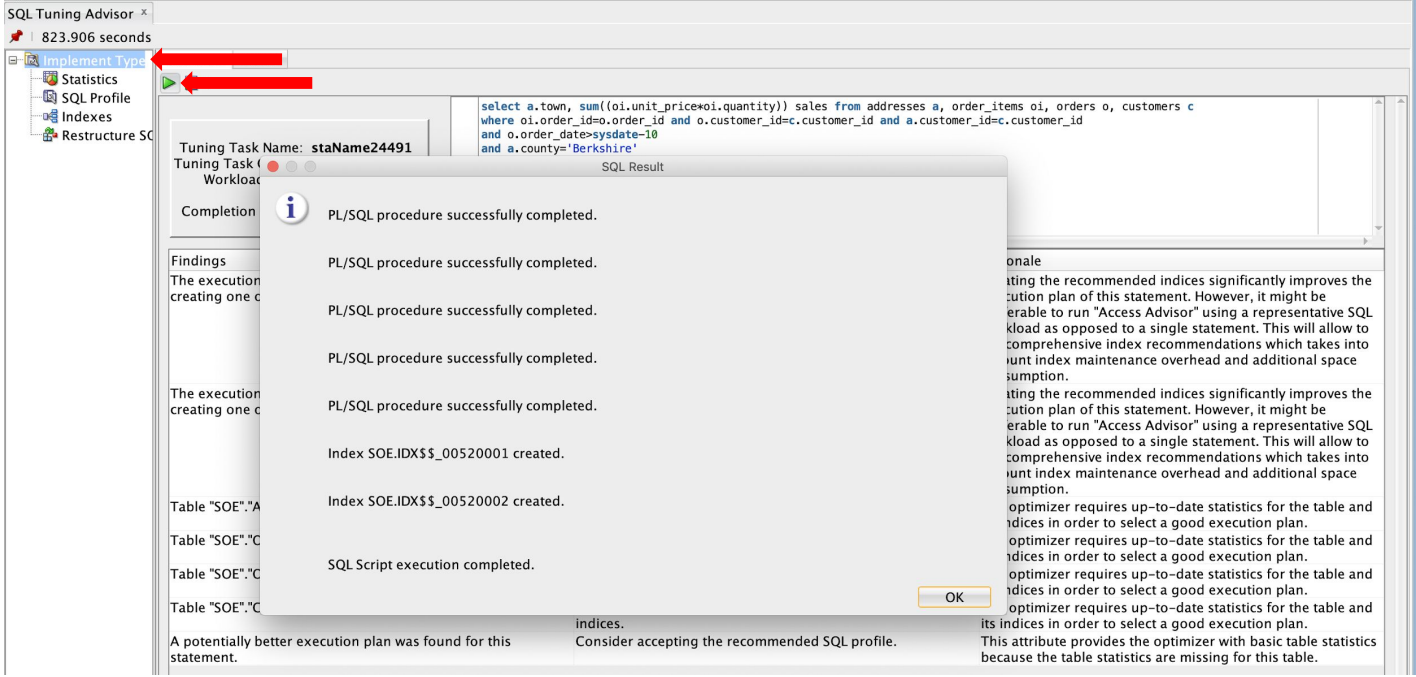
3- Using SQL Profile

Plan hash value: 849859564

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time
0	SELECT STATEMENT		502	29618		93514 (2)	00:00:04
1	SORT GROUP BY ROLLUP		502	29618		93514 (2)	00:00:04
2	NESTED LOOPS		9222	531K		93512 (2)	00:00:04
3	NESTED LOOPS		9222	531K		93512 (2)	00:00:04
* 4	HASH JOIN		3043	145K	4960K	84383 (2)	00:00:04
* 5	HASH JOIN		117K	3570K	4376K	43285 (1)	00:00:02
* 6	TABLE ACCESS FULL	ADDRESSES	117K	2990K		33020 (1)	00:00:02
7	INDEX FAST FULL SCAN	CUSTOMERS_PK	7077K	33M		4308 (1)	00:00:01
* 8	TABLE ACCESS FULL	ORDERS	182K	3204K		40597 (3)	00:00:02
* 9	INDEX RANGE SCAN	ITEM_ORDER_IX	3			2 (0)	00:00:01
10	TABLE ACCESS BY INDEX ROWID	ORDER_ITEMS	3	30		3 (0)	00:00:01

SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)



SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)

```
17:40:58 SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(format=>'ALLSTATS LAST +PEEKED_BINDS +cost +bytes +outline +projection'));
PLAN_TABLE_OUTPUT
SQL_ID  5r5u3upcwykp5, child number 2
-------------------------------------
select a.town, sum((oi.unit_price*oi.quantity)) sales from addresses a,
order_items oi, orders o, customers c  where oi.order_id=o.order_id and
o.customer_id=c.customer_id and a.customer_id=c.customer_id and
o.order_date>sysdate-10 and a.county='Berkshire' group by rollup(town)
order by 1

Plan hash value: 4215863767

-----------------------------------------------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name           | Starts | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| A-Rows |   A-Time   | Buffers |  OMem |  1Mem | Used-Mem |
-----------------------------------------------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                |      1 |        |       |       | 22212 (100)|     94 |00:00:01.70 |   20080 |       |       |          |
|   1 |  SORT GROUP BY ROLLUP         |                |      1 |    502 | 29618 |       | 22212   (1)|     94 |00:00:01.70 |   20080 |  9216 |  9216 | 8192  (0)|
|   2 |   NESTED LOOPS                |                |      1 |  10782 |   621K|       | 22211   (1)|   9060 |00:00:01.70 |   20080 |       |       |          |
|   3 |    NESTED LOOPS               |                |      1 |  10782 |   621K|       | 22211   (1)|   9060 |00:00:01.69 |   19761 |       |       |          |
|*  4 |     HASH JOIN                 |                |      1 |   3557 |   170K|  4896K| 11537   (1)|   2940 |00:00:01.69 |   16986 |   15M|  3098K|   14M (0)|
|*  5 |      INDEX RANGE SCAN         | IDX$$_00520002 |      1 |    167K|  2935K|       |    31   (0)|    178K|00:00:00.02 |    870 |       |       |          |
|*  6 |      HASH JOIN                |                |      1 |    150K|  4556K|  5584K| 10959   (1)|    116K|00:00:01.60 |   16116 |   10M|  3070K|   14M (0)|
|*  7 |       INDEX RANGE SCAN        | IDX$$_00520001 |      1 |    150K|  3816K|       |   636   (1)|    116K|00:00:00.01 |    587 |       |       |          |
|   8 |       INDEX FAST FULL SCAN    | CUSTOMERS_PK   |      1 |   7077K|    33M|       |  4308   (1)|   7067K|00:00:00.54 |   15529 |       |       |          |
|*  9 |     INDEX RANGE SCAN          | ITEM_ORDER_IX  |   2940 |      3 |       |       |     2   (0)|   9060 |00:00:00.01 |    2775 |       |       |          |
|  10 |    TABLE ACCESS BY INDEX ROWID| ORDER_ITEMS    |   9060 |      3 |    30 |       |     3   (0)|   9060 |00:00:00.01 |     319 |       |       |          |
-----------------------------------------------------------------------------------------------------------------------------------------------------------
```

SPM

- SQL Tuning Advisor (dbms_sqltune)
- SPM (SQL Plan Management)

```
Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('19.1.0')
      DB_VERSION('19.1.0')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      INDEX(@"SEL$1" "A"@"SEL$1" ("ADDRESSES"."COUNTY" "ADDRESSES"."CUSTOMER_ID" "ADDRESSES"."TOWN"))
      INDEX_FFS(@"SEL$1" "C"@"SEL$1" ("CUSTOMERS"."CUSTOMER_ID"))
      INDEX(@"SEL$1" "O"@"SEL$1" ("ORDERS"."ORDER_DATE" "ORDERS"."ORDER_ID" "ORDERS"."CUSTOMER_ID"))
      INDEX(@"SEL$1" "OI"@"SEL$1" ("ORDER_ITEMS"."ORDER_ID"))
      LEADING(@"SEL$1" "A"@"SEL$1" "C"@"SEL$1" "O"@"SEL$1" "OI"@"SEL$1")
      USE_HASH(@"SEL$1" "C"@"SEL$1")
      USE_HASH(@"SEL$1" "O"@"SEL$1")
      USE_NL(@"SEL$1" "OI"@"SEL$1")
      NLJ_BATCHING(@"SEL$1" "OI"@"SEL$1")
      SWAP_JOIN_INPUTS(@"SEL$1" "O"@"SEL$1")
      END_OUTLINE_DATA
  */

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

   4 - access("O"."CUSTOMER_ID"="C"."CUSTOMER_ID")
   5 - access("O"."ORDER_DATE">SYSDATE@!-10)
   6 - access("A"."CUSTOMER_ID"="C"."CUSTOMER_ID")
   7 - access("A"."COUNTY"='Berkshire')
   9 - access("OI"."ORDER_ID"="O"."ORDER_ID")

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

   1 - (#keys=1) "TOWN"[VARCHAR2,60], SUM("OI"."UNIT_PRICE"*"OI"."QUANTITY")[22], SYSDEF[4]
   2 - "O"."CUSTOMER_ID"[NUMBER,22], "C"."CUSTOMER_ID"[NUMBER,22], "O".ROWID[ROWID,10], "O"."ORDER_ID"[NUMBER,22], "O"."ORDER_DATE"[TIMESTAMP WITH LOCAL
       TIME ZONE,11], "A"."CUSTOMER_ID"[NUMBER,22], "TOWN"[VARCHAR2,60], "A".ROWID[ROWID,10], "A"."COUNTY"[VARCHAR2,60], "OI".ROWID[ROWID,10],
       "OI"."ORDER_ID"[NUMBER,22], "OI"."UNIT_PRICE"[NUMBER,22], "OI"."QUANTITY"[NUMBER,22]
   3 - "O"."CUSTOMER_ID"[NUMBER,22], "C"."CUSTOMER_ID"[NUMBER,22], "O".ROWID[ROWID,10], "O"."ORDER_ID"[NUMBER,22], "O"."ORDER_DATE"[TIMESTAMP WITH LOCAL
       TIME ZONE,11], "A"."CUSTOMER_ID"[NUMBER,22], "TOWN"[VARCHAR2,60], "A".ROWID[ROWID,10], "A"."COUNTY"[VARCHAR2,60], "OI".ROWID[ROWID,10],
       "OI"."ORDER_ID"[NUMBER,22]
   4 - (#keys=1) "O"."CUSTOMER_ID"[NUMBER,22], "C"."CUSTOMER_ID"[NUMBER,22], "O".ROWID[ROWID,10], "O"."ORDER_ID"[NUMBER,22], "O"."ORDER_DATE"[TIMESTAMP
       WITH LOCAL TIME ZONE,11], "A"."CUSTOMER_ID"[NUMBER,22], "TOWN"[VARCHAR2,60], "A".ROWID[ROWID,10], "A"."COUNTY"[VARCHAR2,60]
   5 - "O".ROWID[ROWID,10], "O"."ORDER_ID"[NUMBER,22], "O"."ORDER_DATE"[TIMESTAMP WITH LOCAL TIME ZONE,11], "O"."CUSTOMER_ID"[NUMBER,22]
   6 - (#keys=1) "A"."CUSTOMER_ID"[NUMBER,22], "C"."CUSTOMER_ID"[NUMBER,22], "A".ROWID[ROWID,10], "A"."COUNTY"[VARCHAR2,60], "TOWN"[VARCHAR2,60]
   7 - "A".ROWID[ROWID,10], "A"."CUSTOMER_ID"[NUMBER,22], "TOWN"[VARCHAR2,60], "A"."COUNTY"[VARCHAR2,60]
   8 - "C"."CUSTOMER_ID"[NUMBER,22]
   9 - "OI".ROWID[ROWID,10], "OI"."ORDER_ID"[NUMBER,22]
  10 - "OI".ROWID[ROWID,10], "OI"."UNIT_PRICE"[NUMBER,22], "OI"."QUANTITY"[NUMBER,22]

Note
-----
   - SQL profile SYS_SQLPROF_016e56cf2ddc0000 used for this statement
```

Tuning the Application Client

Arraysize 15

- 150 seconds
- 1,216,195 network roundtrips

```
[oracle@hol ~]$ sqlplus soe/soe@soe

SQL*Plus: Release 19.0.0.0.0 - Production on Sun Oct 13 21:25:42 2019
Version 19.3.0.0.0

Copyright (c) 1982, 2019, Oracle.  All rights reserved.

Last Successful login time: Sun Oct 13 2019 21:24:58 +02:00

Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0

SP2-0158: unknown SET option "sqlformat"

Session altered.

Elapsed: 00:00:00.00
21:25:42 SQL>
21:25:43 SQL> show arraysize
arraysize 15
21:25:49 SQL> SET AUTOTRACE TRACEONLY STATISTICS
21:25:57 SQL> select * from orders;

18242899 rows selected.

Elapsed: 00:02:30.61

Statistics
----------------------------------------------------------
         0  recursive calls
         0  db block gets
   1460359  consistent gets
    261421  physical reads
         0  redo size
1611441147  bytes sent via SQL*Net to client
  13405246  bytes received via SQL*Net from client
   1216195  SQL*Net roundtrips to/from client
         0  sorts (memory)
         0  sorts (disk)
  18242899  rows processed
```

Tuning the Application Client

Arraysize 15

- 150 seconds
- 1,216,195 network roundtrips

Arraysize 50

- 95 seconds
- 365,859 network roundtrips

```
21:28:34 SQL> set arraysize 50
21:30:09 SQL> select * from orders;

18242899 rows selected.

Elapsed: 00:01:35.08

Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
     621135  consistent gets
     261421  physical reads
          0  redo size
 1448835971  bytes sent via SQL*Net to client
    4022207  bytes received via SQL*Net from client
     364859  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
   18242899  rows processed
```

Tuning the Application Client

Arraysize 15

- 150 seconds
- 1,216,195 network roundtrips

Arraysize 50

- 95 seconds
- 365,859 network roundtrips

Arraysize 500

- 62 seconds
- 36,487 network roundtrips

```
21:31:51 SQL> set arraysize 500
21:33:48 SQL> select * from orders;

18242899 rows selected.

Elapsed: 00:01:02.26

Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
     297393  consistent gets
     261421  physical reads
          0  redo size
 1386116919  bytes sent via SQL*Net to client
     402380  bytes received via SQL*Net from client
      36487  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
   18242899  rows processed
```

Tuning the Application Client

Arraysize 15

- 150 seconds
- 1,216,195 network roundtrips

Arraysize 50

- 95 seconds
- 365,859 network roundtrips

Arraysize 500

- 62 seconds
- 36,487 network roundtrips

Arraysize 5000

- 57 seconds
- 3,650 network roundtrips

```
21:34:54 SQL> set arraysize 5000
21:37:02 SQL> select * from orders;

18242899 rows selected.

Elapsed: 00:00:57.36

Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
     265022  consistent gets
     261421  physical reads
          0  redo size
 1379845052  bytes sent via SQL*Net to client
      40952  bytes received via SQL*Net from client
       3650  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
   18242899  rows processed
```

Tuning the Application Client

JDBC tuning:

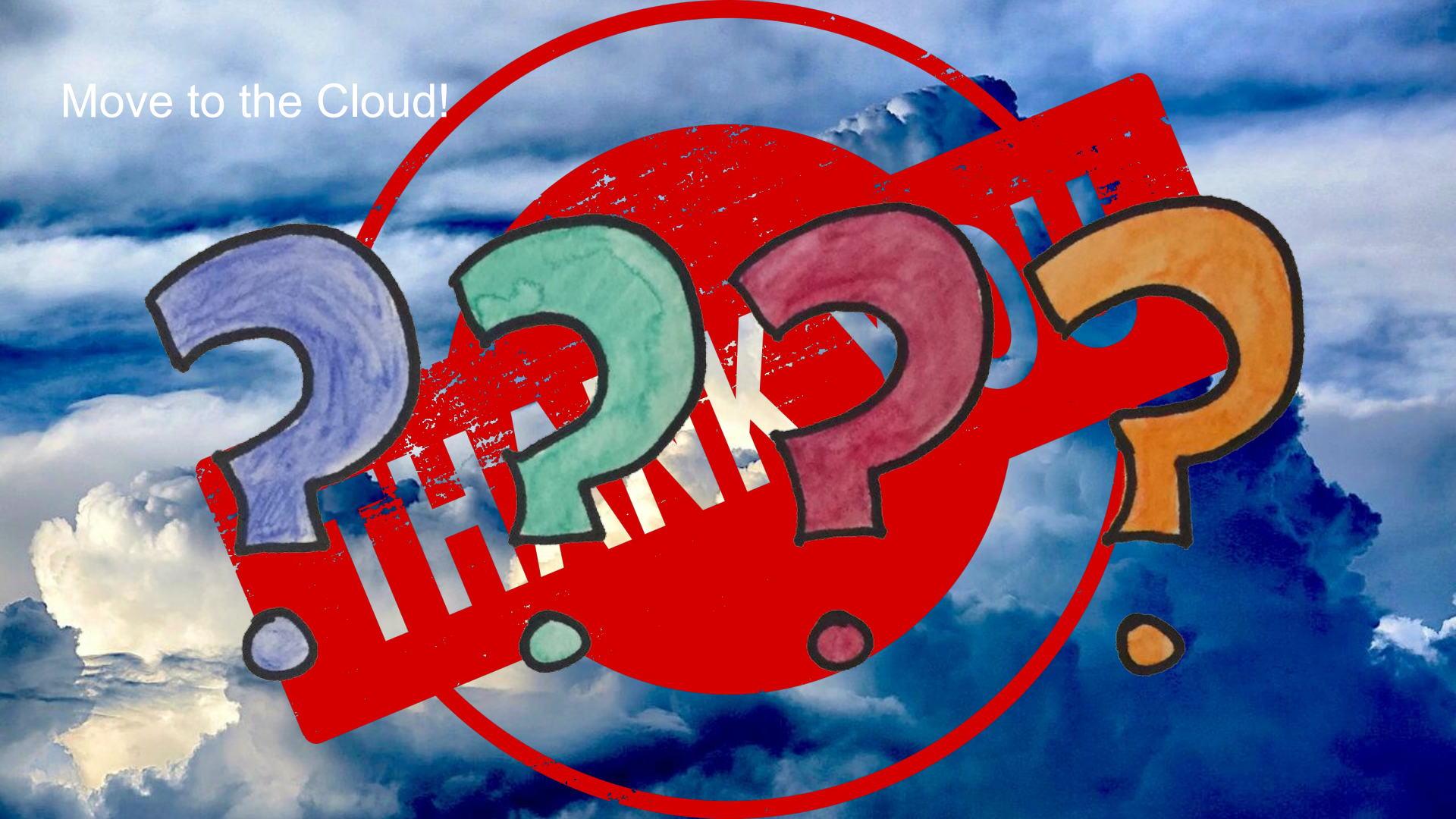
- At connection level you can use:
  - `oracle.jdbc.OracleConnection.setDefaultRowPrefetch()`
  - the property "`defaultRowPrefetch`" when you get a connection with = `oracle.jdbc.DriverManager.getConnection()`
- At statement level you can use:
  - `java.sql.Statement.setFetchSize()`

Client/Server tuning:

- SEND_BUF_SIZE: OS send buffer size
  - sqlnet.ora / tnsnames.ora
- RECV_BUF_SIZE: OS receive buffer size
  - sqlnet.ora / tnsnames.ora
- DEFAULT_SDU_SIZE: larger SDU more throughput, less syscalls/CPU, consumes more memory
  - sqlnet.ora / tnsnames.ora

Conclusion

- Work together with the CBO
- Keep your stats up-to-date
- USE THE DATABASE FEATURES


Move to the Cloud!
?THINK?

Connect / Parse / Commit

Bad code!

```
my $oracle_password = 'test';

### LOOP para teste de carga.
for (my $numero = 1; $numero < $quantidade; $numero++)
        {
        my $oracle_dbh = DBI->connect("dbi:Oracle:host=$oracle_hostname;service_name=$oracle_database", $oracle_username, $oracle_password, {RaiseError => 1, AutoCommit => 1})
        my $oracle_sql = "INSERT INTO T314 (C1) VALUES ($numero)";
        my $oracle_sth = $oracle_dbh->prepare($oracle_sql) or die $DBI::errstr;
        $oracle_sth->execute() or die $DBI::errstr;
        $oracle_dbh->disconnect;
        }

exit;
```

Good code!

```
my $oracle_password = 'test';
my $oracle_dbh = DBI->connect("dbi:Oracle:host=$oracle_hostname;service_name=$oracle_database", $oracle_username, $oracle_password, {RaiseError => 1, AutoCommit => 0});

### LOOP para teste de carga.
my $oracle_sql = "INSERT INTO T314 (C1) VALUES (?)";
my $oracle_sth = $oracle_dbh->prepare($oracle_sql) or die $DBI::errstr;
for (my $numero = 1; $numero < $quantidade; $numero++)
        {
        $oracle_sth->execute($numero) or die $DBI::errstr;
        }

$oracle_dbh->disconnect;
exit;
```

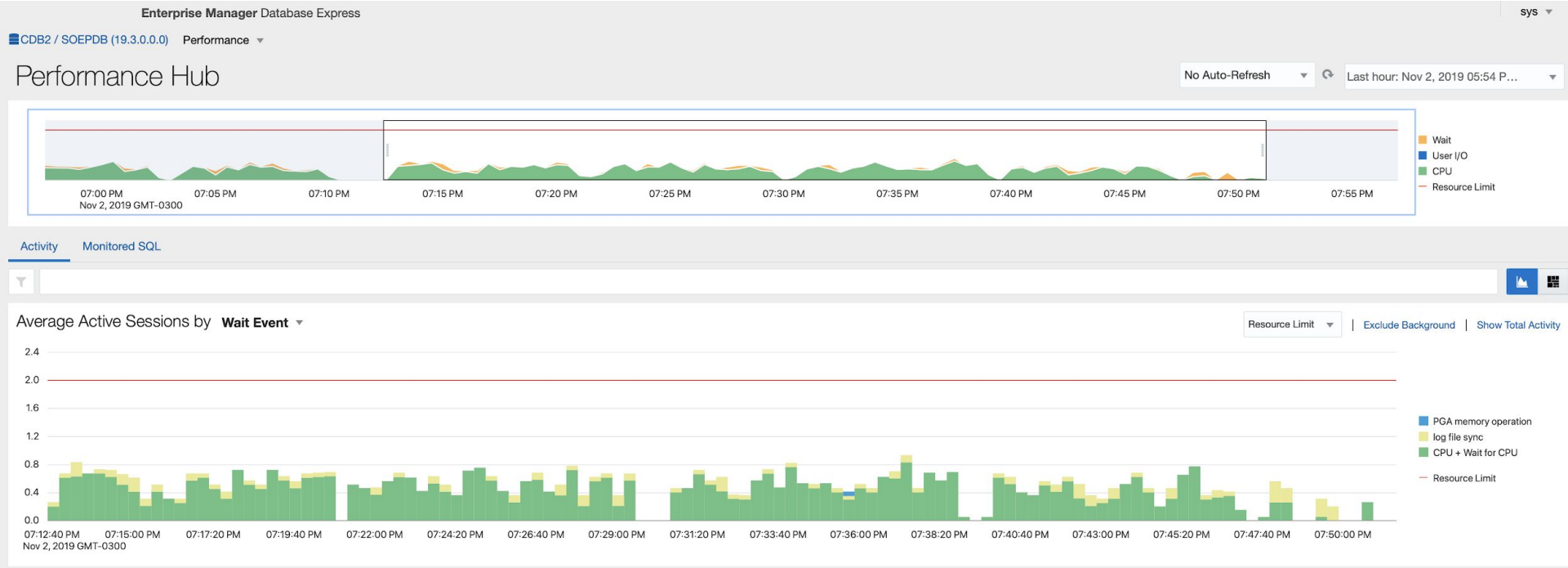
Connect / Parse / Commit

```
18:42:07 SQL> show parameter commit
NAME                 TYPE    VALUE
-------------------- ------- -----
commit_logging       string
commit_point_strength integer 1
commit_wait          string
commit_write         string
```

```
18:42:17 SQL> show parameter cursor_sharing
NAME           TYPE   VALUE
-------------- ------ -----
cursor_sharing string EXACT
18:42:21 SQL> show parameter memoptimize
NAME                 TYPE        VALUE
-------------------- ----------- -----
memoptimize_pool_size big integer 0
```

```
[oracle@hol lab]$ date
Sat Nov  2 19:12:30 -03 2019
[oracle@hol lab]$ sh ConnectCommitBind.sh
/home/oracle/lab/ConnectBAD_CommitBAD_BindsBAD.pl 10000

real    8m3.699s
user    0m53.682s
sys     0m8.226s

/home/oracle/lab/ConnectBAD_CommitBAD_BindsGOOD.pl 10000

real    7m41.909s
user    0m54.193s
sys     0m8.609s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsBAD.pl 10000

real    8m16.127s
user    0m54.828s
sys     0m8.702s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsGOOD.pl 10000

real    7m11.294s
user    0m54.144s
sys     0m8.967s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsBAD.pl 10000

real    0m23.081s
user    0m1.414s
sys     0m0.356s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD.pl 10000

real    0m14.084s
user    0m1.405s
sys     0m0.306s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD.pl 10000

real    0m9.098s
user    0m1.130s
sys     0m0.447s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD_ONE.pl 10000

real    0m2.089s
user    0m0.398s
sys     0m0.254s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD.pl 10000

real    0m2.125s
user    0m0.550s
sys     0m0.233s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD_PERFECT.pl 10000

real    0m2.082s
user    0m0.231s
sys     0m0.146s

[oracle@hol lab]$
[oracle@hol lab]$ date
Sat Nov  2 19:57:48 -03 2019
```

Connect / Parse / Commit



Connect / Parse / Commit

```
20:21:21 SQL> alter  system set cursor_sharing=force;

System altered.

[oracle@hol lab]$ date
Sat Nov  2 20:21:35 -03 2019
[oracle@hol lab]$ sh ConnectCommitBind.sh
/home/oracle/lab/ConnectBAD_CommitBAD_BindsBAD.pl 10000

real    7m13.629s
user    0m53.806s
sys     0m8.294s

/home/oracle/lab/ConnectBAD_CommitBAD_BindsGOOD.pl 10000

real    7m15.735s
user    0m53.674s
sys     0m8.536s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsBAD.pl 10000

real    6m56.346s
user    0m54.030s
sys     0m8.632s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsGOOD.pl 10000

real    7m9.046s
user    0m53.464s
sys     0m9.043s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsBAD.pl 10000

real    0m17.092s
user    0m1.182s
sys     0m0.387s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD.pl 10000

real    0m17.102s
user    0m1.504s
sys     0m0.253s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD.pl 10000

real    0m2.104s
user    0m0.480s
sys     0m0.239s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD_ONE.pl 10000

real    0m2.077s
user    0m0.482s
sys     0m0.147s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD.pl 10000

real    0m3.118s
user    0m0.616s
sys     0m0.366s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD_PERFECT.pl 10000

real    0m1.405s
user    0m0.178s
sys     0m0.169s
```

8m03s -> 7m13s

8m16s -> 6m56s

23s -> 17s

09s -> 02s

02s -> 02s

Connect / Parse / Commit

cursor_sharing=force 

R -> MOS Doc IDs:

- **462520.1**
- **1680306.1**
- **2492866.1**
- **14087914.8**

Connect / Parse / Commit

```
22:10:54 SQL>  alter system set commit_wait=nowait;

System altered.

Elapsed: 00:00:00.017
22:11:06 SQL> alter system set commit_logging=batch;

System altered.

[oracle@hol lab]$ date
Sat Nov  2 22:47:58 -03 2019
[oracle@hol lab]$ sh ConnectCommitBind.sh
/home/oracle/lab/ConnectBAD_CommitBAD_BindsBAD.pl 10000

real    8m2.779s
user    0m53.728s
sys     0m8.392s

/home/oracle/lab/ConnectBAD_CommitBAD_BindsGOOD.pl 10000

real    7m30.270s
user    0m53.964s
sys     0m8.898s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsBAD.pl 10000

real    8m33.667s
user    0m53.932s
sys     0m8.899s

/home/oracle/lab/ConnectBAD_CommitGOOD_BindsGOOD.pl 10000

real    6m59.584s
user    0m53.422s
sys     0m9.176s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsBAD.pl 10000

real    0m11.107s
user    0m1.127s
sys     0m0.602s

/home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD.pl 10000

real    0m3.136s
user    0m0.376s
sys     0m0.514s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD.pl 10000

real    0m9.086s
user    0m1.165s
sys     0m0.385s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsBAD_ONE.pl 10000

real    0m2.094s
user    0m0.423s
sys     0m0.202s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD.pl 10000

real    0m2.076s
user    0m0.482s
sys     0m0.274s

/home/oracle/lab/ConnectGOOD_CommitGOOD_BindsGOOD_PERFECT.pl 10000

real    0m1.074s
user    0m0.156s
sys     0m0.137s
```

8m03s -> 8m02s

7m41s -> 7m30s

23s -> 11s

14s -> 03s

Memoptimized rowstore

```
08:13:05 SQL>  alter system set commit_wait=wait container=all;

System altered.

Elapsed: 00:00:00.029
08:13:17 SQL> alter system set commit_logging=immediate container=all;

System altered.


08:18:16 SQL> r
  1* select segment_name, bytes/1024/1024 mb from user_segments where segment_name='T314';
SEGMENT_NAME     MB
T314              9


Elapsed: 00:00:00.005
08:18:18 SQL> drop table T314 purge;

Table dropped.

Elapsed: 00:00:00.271
08:19:17 SQL> CREATE TABLE T314 (C1 NUMBER) SEGMENT CREATION IMMEDIATE MEMOPTIMIZE FOR WRITE;

CREATE TABLE T314 (C1 NUMBER) SEGMENT CREATION IMMEDIATE MEMOPTIMIZE FOR WRITE
 *
ERROR at line 1:
ORA-12754: Feature 'Memoptimized Rowstore' is disabled due to missing capability 'Runtime Environment'.
```

```
08:26:10 SQL> select a.ksppinm name,
  2  b.ksppstvl value,
  3  b.ksppstdf deflt,
  4  decode
  5  (a.ksppity, 1,
  6  'boolean', 2,
  7  'string', 3,
  8  'number', 4,
  9  'file', a.ksppity) type,
 10  a.ksppdesc description
 11  from
 12  sys.x$ksppi a,
 13  sys.x$ksppcv b
 14  where
 15  a.indx = b.indx
 16  --and a.ksppinm like '\_%' escape '\'
 17  and a.ksppinm like '\_exadata%' escape '\'
 18  order by name;
NAME                 VALUE   DEFLT   TYPE      DESCRIPTION
_exadata_feature_on  FALSE   TRUE    boolean   Exadata Feature On
```

Memoptimized rowstore

```
08:29:21 SQL> alter system set "_exadata_feature_on"=true;

alter system set "_exadata_feature_on"=true
                 *
ERROR at line 1:
ORA-02095: specified initialization parameter cannot be modified

Elapsed: 00:00:00.002
08:29:26 SQL> alter system set "_exadata_feature_on"=true scope=spfile;

System altered.

Elapsed: 00:00:00.009
08:29:37 SQL> startup force;

Total System Global Area   1577055360 bytes
Fixed Size                    9135232 bytes
Variable Size               469762048 bytes
Database Buffers           1090519040 bytes
Redo Buffers                  7639040 bytes
Database mounted.
Database opened.
08:29:57 SQL> alter session set container=soepdb;

Session altered.

Elapsed: 00:00:00.005
08:30:09 SQL> CREATE TABLE T314 (C1 NUMBER) SEGMENT CREATION IMMEDIATE MEMOPTIMIZE FOR WRITE;

Table created.

Elapsed: 00:00:00.039
```

```
08:30:11 SQL> alter system set memoptimize_pool_size=10M;

alter system set memoptimize_pool_size=10M
 *
ERROR at line 1:
ORA-65040: operation not allowed from within a pluggable database

Elapsed: 00:00:00.002
08:30:39 SQL> alter session set container=cdb$root;

Session altered.

Elapsed: 00:00:00.001
08:30:55 SQL> alter system set memoptimize_pool_size=10M;

alter system set memoptimize_pool_size=10M
 *
ERROR at line 1:
ORA-02097: parameter cannot be modified because specified value is invalid
ORA-02095: specified initialization parameter cannot be modified

Elapsed: 00:00:00.002
08:30:58 SQL> alter system set memoptimize_pool_size=10M scope=spfile;

System altered.

Elapsed: 00:00:00.010
08:31:09 SQL> startup force;

Total System Global Area   1627387104 bytes
Fixed Size                    9135328 bytes
Variable Size               452984832 bytes
Database Buffers           1157627904 bytes
Redo Buffers                  7639040 bytes
Database mounted.
Database opened.
08:31:31 SQL>
```

Memoptimized rowstore

```
08:36:07 SQL> show parameter memop
NAME                 TYPE        VALUE
-------------------- ----------- -----
memoptimize_pool_size big integer 64M
08:36:12 SQL> alter session set container=soepdb;

Session altered.

Elapsed: 00:00:00.003
08:44:29 SQL> alter table T314 memoptimize for read;

alter table T314 memoptimize for read
 *
ERROR at line 1:
ORA-62142: MEMOPTIMIZE FOR READ feature requires NOT DEFERRABLE PRIMARY KEY constraint on the table

Elapsed: 00:00:00.030
08:44:45 SQL> alter table T314 add constraint pk01 primary key (c1);

Table altered.

Elapsed: 00:00:00.049
08:48:00 SQL> alter table T314 memoptimize for read;

Table altered.

Elapsed: 00:00:00.013
```

Memoptimized rowstore

The Memoptimized Rowstore provides the following functionality:

- Fast ingest

  Fast ingest optimizes the processing of high-frequency, single-row data inserts into a database. Fast ingest uses the large pool for buffering the inserts before writing them to disk, so as to improve data insert performance.

- Fast lookup

  Fast lookup enables fast retrieval of data from a database for high-frequency queries. Fast lookup uses a separate memory area in the SGA called the *memoptimize pool* for buffering the data queried from tables, so as to improve query performance.

Note: For using fast lookup, you must allocate appropriate memory size to the memoptimize pool using the `MEMOPTIMIZE_POOL_SIZE` initialization parameter.

Memoptimized rowstore

Limitations for using fast ingest

Tables with the following characteristics cannot use fast ingest:

- Tables with:
  - disk compression
  - in-memory compression
  - column default vales
  - encryption
  - functional indexes
  - domain indexes
  - bitmap indexes
  - bitmap join indexes
  - ref types
  - varray types
  - OID$ types
  - sub-partition stats
  - unused columns
  - virtual columns
  - LOBs
  - triggers
  - binary columns
  - foreign keys
  - row archival
  - invisible columns
- Temporary tables
- Nested tables
- Index organized tables
- External tables
- Materialized views with on-demand refresh

The following are some additional considerations for fast ingest:

- Because fast ingest buffers data in the large pool, there is a possibility of data loss in the event of a system failure. To avoid data loss, a client must keep a local copy of the data after performing inserts, so that it can replay the inserts in the event of a system failure before the data is written to disk. A client can use the `DBMS_MEMOPTIMIZE` package subprograms to track the durability of the inserts. After inserts are written to disk, a client can destroy its local copy of the inserted data.
- Queries do not read data from the large pool, hence data inserted using fast ingest cannot be queried until it is written to disk.
- Parent-child transactions must be synchronized to avoid errors. For example, foreign key inserts and updates of rows inserted into the large pool can return errors, if the parent data is not yet written to disk.
- Index operations are supported by fast ingest similar to the regular inserts. However, for fast ingest, database performs index operations while writing data to disk, and not while writing data into the large pool.

Note: A table can be configured for using both fast ingest and fast lookup.

```
09:00:52 SQL> alter table T314 move tablespace SOE_DATA;

alter table T314 move tablespace SOE_DATA
            *
ERROR at line 1:
ORA-62180: MEMOPTIMIZE FOR WRITE unsupported DDL.
```

```
09:10:29 SQL> alter table T314 no memoptimize for write;

Table altered.

Elapsed: 00:00:00.006
09:10:36 SQL> alter table T314 move tablespace SOE_DATA;

Table altered.

Elapsed: 00:00:00.019
09:10:41 SQL> alter table T314 memoptimize for write;

Table altered.

Elapsed: 00:00:00.006
```

Memoptimized rowstore

```
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl ConnectGOOD_CommitBAD_BindsGOOD.pl 10000

real    0m18.126s
user    0m1.460s
sys     0m0.279s
```

14s -> 03s -> 18s

```
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl ConnectGOOD_CommitBAD_BindsGOOD.pl 10000
DBD::Oracle::st execute failed: ORA-00001: unique constraint (TEST.PK) violated (DBD ERROR: OCIStmtExecute) [for Statement "INSERT INTO T314 (C1) VALUES (?)"
DBD::Oracle::st execute failed: ORA-00001: unique constraint (TEST.PK) violated (DBD ERROR: OCIStmtExecute) [for Statement "INSERT INTO T314 (C1) VALUES (?)"

real    0m1.104s
user    0m0.037s
sys     0m0.012s
```

```
[oracle@hol lab]$ grep memoptimize_write /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl
       my $oracle_sql = "INSERT /*+ memoptimize_write */ INTO T314 (C1) VALUES (?)";
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl 10000

real    0m2.128s
user    0m0.551s
sys     0m0.190s
```

14s -> 03s -> 18s -> 02s

```
[oracle@hol lab]$ date
Sun Nov  3 09:59:04 -02 2019
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl 10000

real    0m2.074s
user    0m0.549s
sys     0m0.156s
```

No PK error

Memoptimized rowstore

```
10:01:38 SQL> select count(*) from t314;
  COUNT(*)
      9999
```

```
Elapsed: 00:00:00.003
10:02:49 SQL> exec dbms_memoptimize_admin.writes_flush();
PL/SQL procedure successfully completed.

Elapsed: 00:00:00.002

10:04:10 SQL> select count(*) from t314;
  COUNT(*)
      9999

10:05:05 SQL> truncate table t314;

Table truncated.

Elapsed: 00:00:00.025

10:05:27 SQL> select count(*) from t314;
  COUNT(*)
         0
[oracle@hol lab]$ date
Sun Nov  3 10:05:34 -02 2019
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl 10000

real    0m2.080s
user    0m0.555s
sys     0m0.239s
```

```
10:05:57 SQL> select count(*) from t314;
  COUNT(*)
      9999
```

```
[oracle@hol lab]$ date
Sun Nov  3 10:08:37 -02 2019
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl 10000

real    0m2.088s
user    0m0.580s
sys     0m0.193s
```

```
10:08:47 SQL> select count(*) from t314;
  COUNT(*)
      9999

10:08:48 SQL> BEGIN dbms_memoptimize_admin.writes_flush(); END;
  2* /
PL/SQL procedure successfully completed.

10:08:53 SQL> select count(*) from t314;
  COUNT(*)
      9999
```

```
[oracle@hol lab]$ date
Sun Nov  3 10:17:58 -02 2019
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl ConnectGOOD_CommitBAD_BindsGOOD.pl 10000
DBD::Oracle::st execute failed: ORA-00001: unique constraint (TEST.PK) violated (DBD ERROR: OCIStmtExecute) [for Statement "INSE
DBD::Oracle::st execute failed: ORA-00001: unique constraint (TEST.PK) violated (DBD ERROR: OCIStmtExecute) [for Statement "INSE
```

Memoptimized rowstore

```
[oracle@hol lab]$ date
Sun Nov  3 10:29:11 -02 2019
[oracle@hol lab]$ time /u01/app/oracle/product/19/perl/bin/perl /home/oracle/lab/ConnectGOOD_CommitBAD_BindsGOOD_MemoptWrite.pl 20000

real    0m4.097s
user    0m1.037s
sys     0m0.464s

10:29:22 SQL> select count(*) from t314;
  COUNT(*)
     19999
```

Memoptimized rowstore

```
11:02:10 SQL> alter table T314 drop constraint pk;

alter table T314 drop constraint pk
 *
ERROR at line 1:
ORA-62142: MEMOPTIMIZE FOR READ feature requires NOT DEFERRABLE PRIMARY KEY constraint on the table

Elapsed: 00:00:00.004
11:02:18 SQL> alter table T314 NO MEMOPTIMIZE FOR READ;

Table altered.

Elapsed: 00:00:00.029
11:02:34 SQL> alter table T314 drop constraint pk;

Table altered.

Elapsed: 00:00:00.181
11:02:39 SQL> truncate table T314;
```

```
11:03:20 SQL> select min(c1) , max(c1) from t314;

  MIN(C1)   MAX(C1)
_________ _________
        1      9999

11:02:50 SQL> select count(*) from t314;

  COUNT(*)
__________
         0


Elapsed: 00:00:00.005

11:02:56 SQL> select count(*) from t314;

  COUNT(*)
__________
      9999


Elapsed: 00:00:00.003

11:03:06 SQL> select count(*) from t314;

  COUNT(*)
__________
     39996
```

Move to the Cloud!


Pythian
LOVE
YOUR
DATA