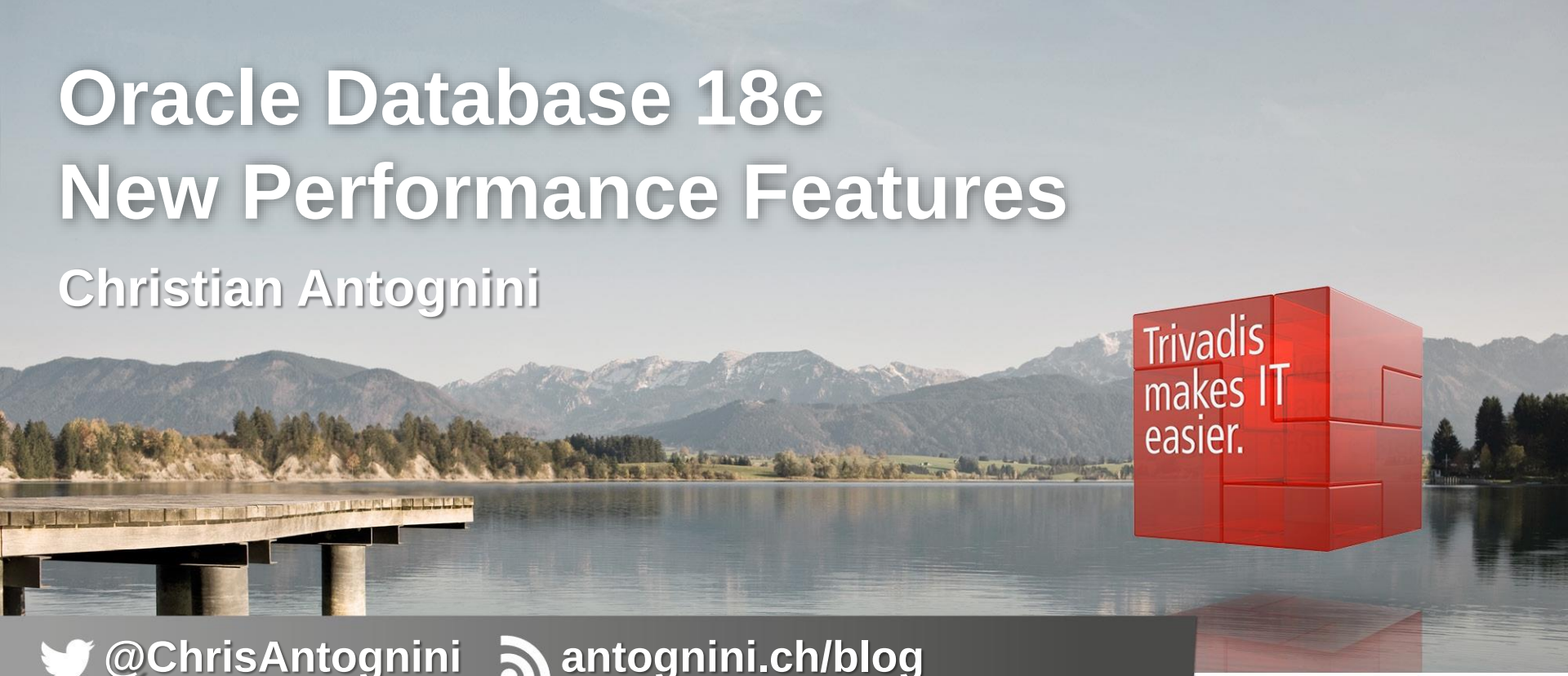
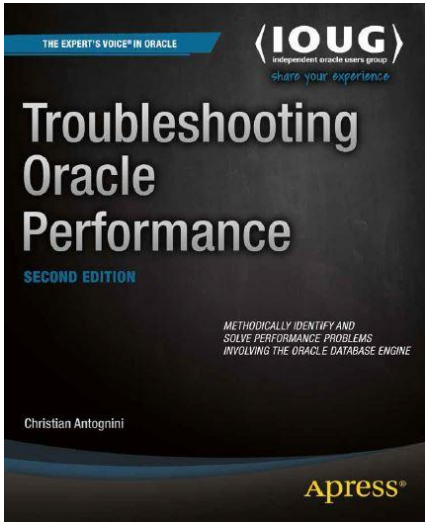
Oracle Database 18c
New Performance Features

Christian Antognini

@ChrisAntognini antognini.ch/blog



@ChrisAntognini



Senior principal consultant, trainer and partner at Trivadis

- christian.antognini@trivadis.com
- http://antognini.ch

Focus: get the most out of database engines

- Logical and physical database design
- Query optimizer
- Application performance management

Author of *Troubleshooting Oracle Performance* (Apress, 2008/14)

OakTable Network, Oracle ACE Director

trivadis
makes IT easier.

Agenda

1. Scalable Sequences
2. Memoptimized Rowstore
3. Parallel Execution
4. In-Memory

Scalable Sequences

Purpose

It eliminates index block contention during concurrent inserts using a sequence

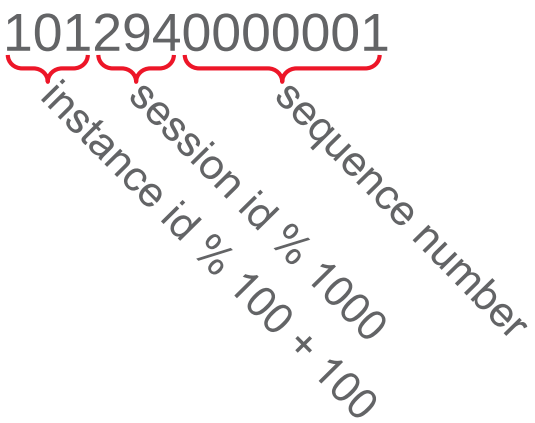
- RAC and non-RAC

Alternative approaches

- Reverse indexes
- Hash partitioned indexes

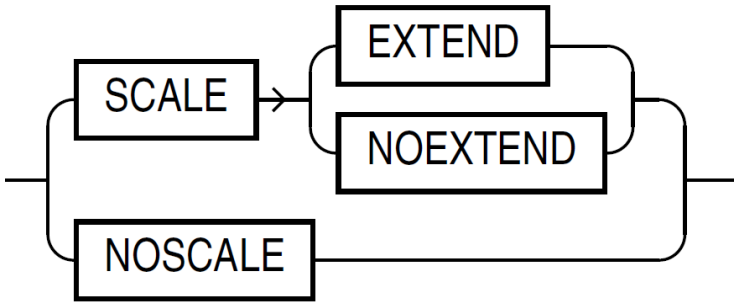
Key Concepts

A numeric prefix based on the instance/session ID is added to the number returned by the sequence



Syntax

New keywords of the CREATE/ALTER SEQUENCE statement



Source: Oracle Database SQL Language Reference guide

Memoptimized Rowstore

Purpose

It enables fast lookups of data that is queried based on PK

Key Concepts

Take advantage of a new memory structure in the SGA, the memoptimize pool, to provide fast access to specific data

The usage of the memoptimize pool has to be manually enabled at table level

Data has to be manually loaded in the memoptimize pool

Transparent for clients

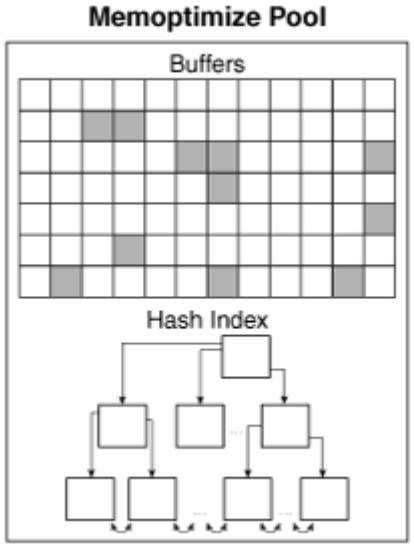
Memoptimize Pool

MEMOPTIMIZE_POOL_SIZE specifies the size of the memoptimize pool (default is 0)

- Static
- Can’t be set at the PDB level

The memoptimize pool is allocated from the SGA

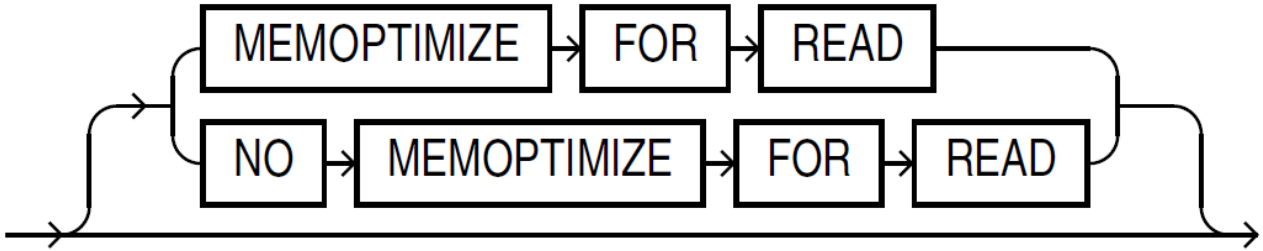
- 75% used for table blocks
- 25% used for hash index



Source: Oracle Database Concepts guide

Enable Memoptimize at Table Level

New keywords of the CREATE/ALTER TABLE statement



Source: Oracle Database SQL Language Reference guide

At the moment it's enabled, at least a segment must exist

- No deferred storage

Populate Memoptimize Pool

Population requests are initiated through DBMS_MEMOPTIMIZE.POPULATE

Populations are carried out asynchronously by a background process

No easy way to find out how much the memoptimize pool is used

- memopt% statistics in V$SYSSTAT gives you a clue
- In Multitenant, population statistics are available in the root only

New Access Path

Lookups bypass the SQL execution layer and execute directly in the data access layer

```
-----------------------------------------------------------
| Id  | Operation                                 | Name  |
-----------------------------------------------------------
|   0 | SELECT STATEMENT                          |       |
|   1 |  TABLE ACCESS BY INDEX ROWID READ OPTIM| T4    |
|*  2 |   INDEX UNIQUE SCAN READ OPTIM            | T4_PK |
-----------------------------------------------------------
```

Technical Requirements

Heap-organized table

Primary key

Not supported

- Identity column
- Compression
- Reference partitioning

The new access path isn't used if

- WHERE clause doesn't contain only equalities on the PK columns
- Query executed from PL/SQL
- SQL trace enabled
- GATHER_PLAN_STATISTICS
- STATISTICS_LEVEL = ALL

Licensing Requirements

On-premises Oracle Database Enterprise Edition on Engineered Systems

- But not on ODA!

Oracle Database Cloud Service Enterprise Edition – Extreme Performance

Oracle Database Exadata Cloud Service

Parallel Execution

Configuration – Performance Feedback

As of 18.1 it's no longer controlled by OPTIMIZER_ADAPTIVE_STATISTICS

- Only PARALLEL_DEGREE_POLICY controls it
- Solves the 12c issue described in this blog post

Configuration – PARALLEL_MIN_DEGREE

New parameter introduced in 18.1

It controls the minimum DOP computed by auto DOP

Its default value is 1

Its maximum value is CPU_COUNT

- PARALLEL_MIN_DEGREE = CPU

Configuration – PARALLEL_SERVERS_TARGET

As of 18.1 it can be set at the PDB level

This enhancement is necessary to make parallel statement queuing configurable in a multitenant environment

Partition-wise Operations

As of 18.1 windowing functions can be executed as a parallel partition-wise operation

OPTIMIZER_FEATURES_ENABLE as well as the hints (NO_)USE_PARTITION_WISE_WIF control the feature

Partition-wise Operations – Example

```
SELECT n1, d1, avg(n2) OVER (PARTITION BY n1, d1) AS avg FROM t

-------------------------------------------------------------------------------
| Id  | Operation                | Name     |    TQ  |IN-OUT| PQ Distrib |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |          |        |      |            |
|   1 |  PX COORDINATOR          |          |        |      |            |
|   2 |   PX SEND QC (RANDOM)    | :TQ10000 |  Q1,00 | P->S | QC (RAND)  |
|   3 |    PX PARTITION LIST ALL|          |  Q1,00 | PCWC |            |
|   4 |     WINDOW SORT          |          |  Q1,00 | PCWP |            |
|   5 |      TABLE ACCESS FULL   | T        |  Q1,00 | PCWP |            |
-------------------------------------------------------------------------------
```

PARTITION BY RANGE
SUBPARTITION BY LIST

Statement Queueing Enhancements

Specify a timeout action

- Through 12.2 the statement is terminated (ORA-7454)
- As of 18.1 the action is specified by the PQ_TIMEOUT_ACTION directive (CANCEL, RUN)

In-Memory

Enabling Objects for Population

Objects eligible for population in the IMCS as well as their compression level are managed in two ways:

- Manually through the INMEMORY clause
- Automatically
  - As of 12.2 with ADO policies
  - As of 18.1 with *Automatic In-Memory*

Automatic In-Memory

In case of memory pressure (population fails), cold segments can be evicted

- ADO policies can’t be overridden
- Only segments without priority are considered

It’s controlled at the system level by INMEMORY_AUTOMATIC_LEVEL

- OFF (default)
- LOW
- MEDIUM (give priority to segments that experienced a population failure)

Expression Statistics Store (ESS)

As of 12.2 the SQL engine automatically maintains a new repository about expression evaluation

It stores information about the estimated frequency and cost of their use in queries

```
SQL> SELECT expression_text, evaluation_count, fixed_cost, created,
  2         last_modified
  3  FROM user_expression_statistics
  4  WHERE table_name = 'T';

EXPRESSION_TEXT EVALUATION_COUNT FIXED_COST CREATED   LAST_MODIFIED
--------------- ---------------- ---------- --------- -------------
ROUND("N2",0)                 1 3.4985E-06 17-AUG-17 17-AUG-17
"N5"+1                     1000 1.7493E-06 17-AUG-17 17-AUG-17
```

Capture of IM Expressions

As of 12.2 In-Memory leverages the ESS to find out which are the hottest expressions during a capture interval

Source: Oracle Database In-Memory guide

Dynamic Capture Window for In-Memory Expressions

18.1 introduces a new capture interval

- CUMULATIVE: no time limit (12.2)
- CURRENT: 24h (12.2)
- WINDOW: user-defined interval

Dynamic Capture Window for In-Memory Expressions Example

Start the user-defined capture window

```
dbms_inmemory_admin.ime_open_capture_window()
```

Wait as long as necessary…

Stop the window and capture expressions

```
dbms_inmemory_admin.ime_close_capture_window()
```

```
dbms_inmemory_admin.ime_capture_expressions('WINDOW')
```

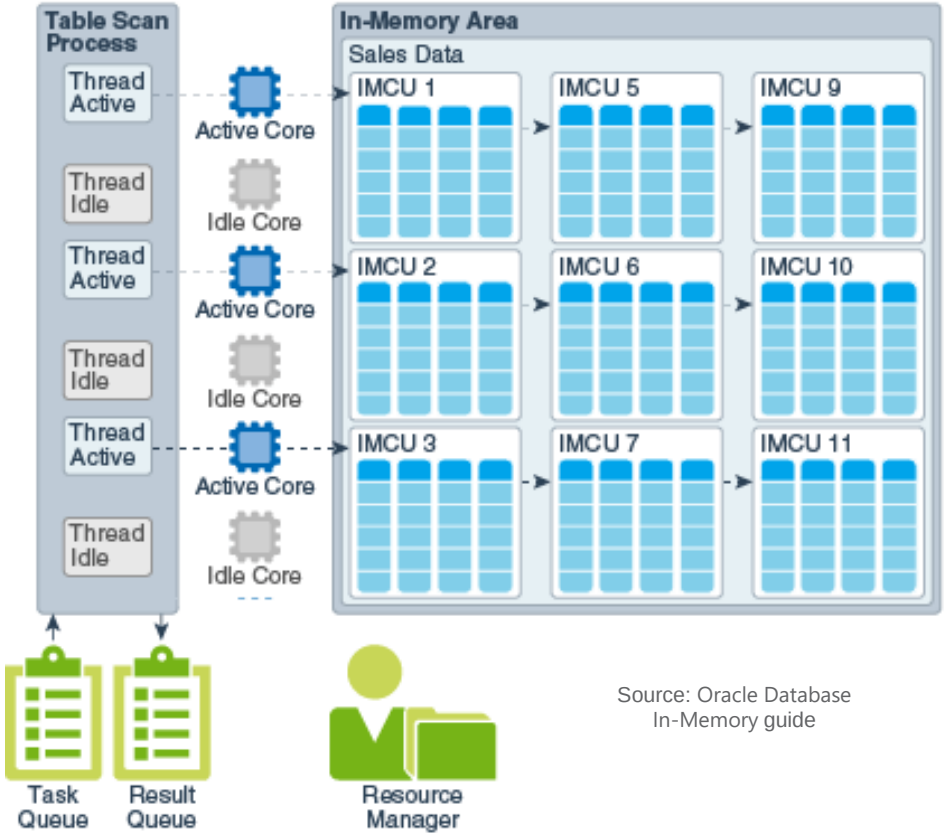
Dynamic Scans

Multithreaded IM scans (independent from PX)

Resource Manager must be enabled

Transparent for the client

No configuration needed



Source: Oracle Database In-Memory guide

Optimized Arithmetic

It uses an optimized encoding for NUMBER enabling fast calculations using SIMD instructions

Dual storage because not all row sources support it

Only for QUERY LOW compressed segments

Controlled at the system level by INMEMORY_OPTIMIZED_ARITHMETIC (disabled by default)

Support for External Tables

As of 18.1 external tables are eligible for population in the IMCS

Restrictions:

- Only manual population
- Only ORACLE_LOADER and ORACLE_DATAPUMP
- QUERY_REWRITE_INTEGRITY = STALE_TOLERATED

Not supported:

- Parallel execution
- Partitioning
- Join groups
- IM expressions
- Optimized arithmetic

Summary

- Few new features
- Scalable sequences: simple improvement that can solve real problem
- Memoptimized rowstore: interesting concept, but irrelevant for many because of licensing requirements
- PX and IMCS are getting better and better

Questions and Answers

Christian Antognini
Senior Principal Consultant

christian.antognini@trivadis.com

@ChrisAntognini

trivadis
makes IT easier.