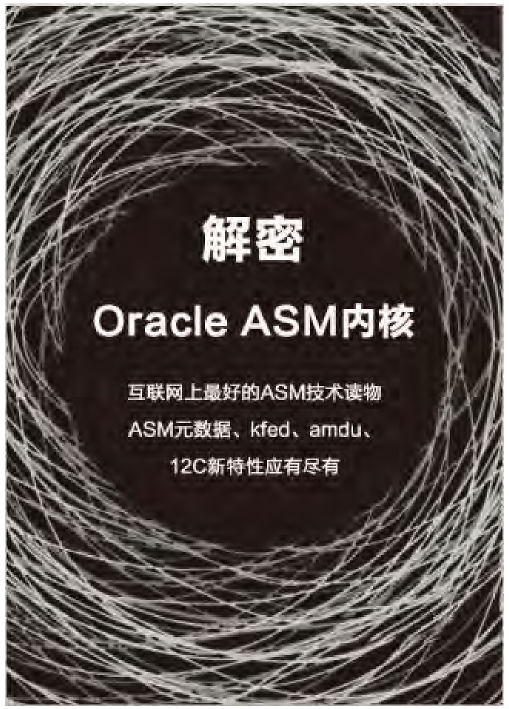
Best Practices for Getting Started with Oracle Database In-Memory 12C

Xinghua Wei
WOQUTECH
2017.10

WHO AM I

- The founder of DBGeeK user group
- Oracle ACE Associate
- Oracle Database Performance geek(10+ years)
- Troubleshooter
- Worked on WOQU Technology http://woqutech.com



DB GEEK

SH'OUG SHANGHAI ORACLE USERS GROUP 上海 ORACLE用户组

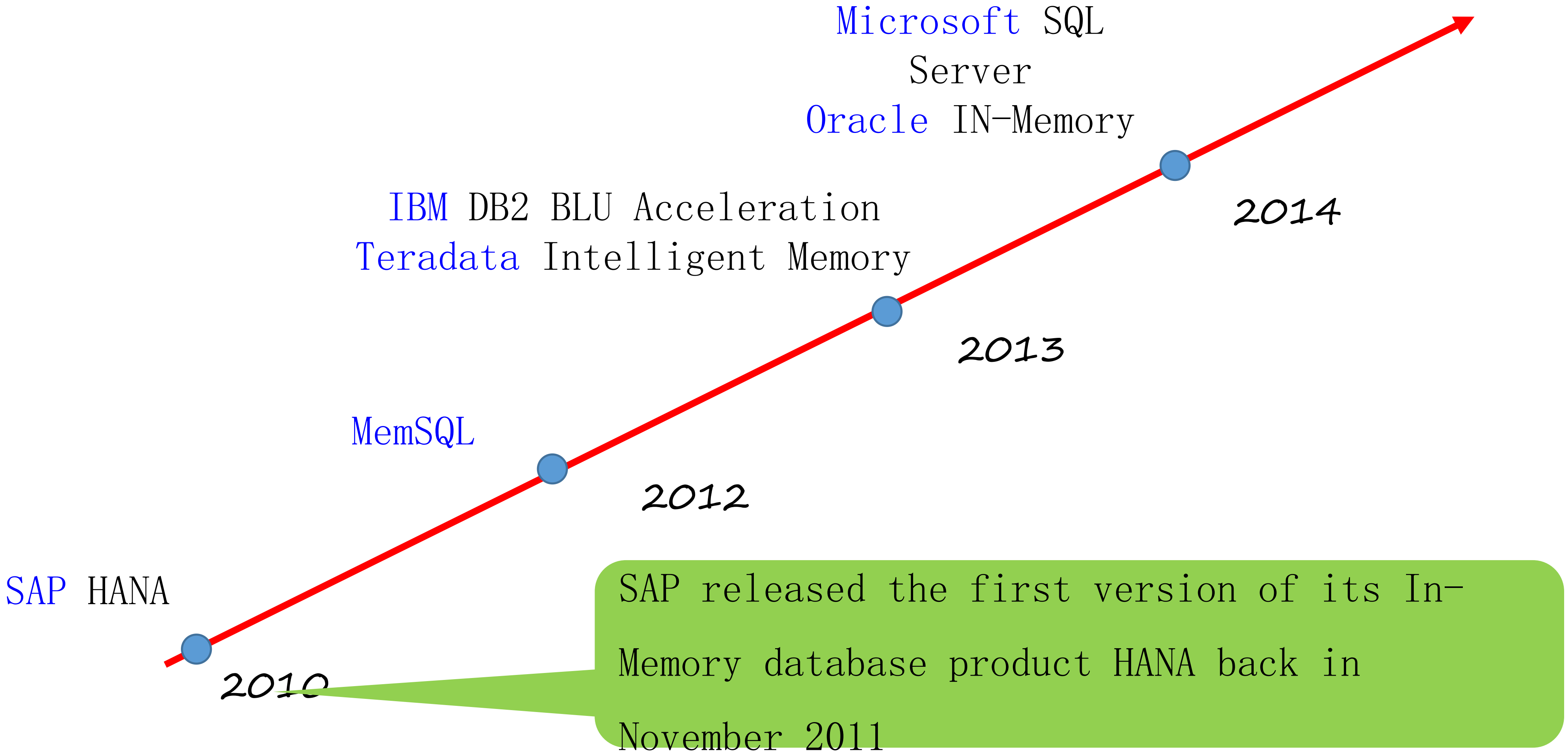
ACOUG MEMBER

AGENDA

- Memory is the future
- Why IM high performance
  - Column format
  - SIMD
  - Compression
  - Data Skipping
- When to Use Oracle Database In-Memory
- The impact of enabling IM feature on OLTP
- The Advantage of Oracle IM compares the other IM databases

In December 2013 IDC firm predicted that “Memory Optimized (“In-Memory”) Database Technology is taking over Enterprise Databases”.

Memory is the new disk



Memory is the future

- Business-driven
- Data-driven
- The maturity of the technical conditions

Memory is the future

- Business-driven
- Data-driven
- The maturity of the technical conditions

Latency Numbers Every Programmer Should Know

Latency Number Every Programmer Should Know

Latency Comparison Number							
L1 cache reference	0.5	ns					
Branch mispredict	5	ns					
L2 cache reference	7	ns					14× L1 cache
Main memory reference	**100**	**ns**					**20× L2 cache, 200× L2 cache**
Compressor 1k bytes with zippy	3,000	ns	3	us			
Send 1K bytes over 1 Gbps network	10,000	ns	10	us			
Read 4K randomly from SSD*	150,000	ns	150	us			~1GB/sec SSD
Read 1MB sequentially from memory	250,000	ns	250	us			
Round trip within same datacenter	500,000	ns	500	us			
Read 1MB sequentially from SSD*	1,000,000	ns	1,000	us	1	ms	~1GB/sec SSD, 4× memory
Disk seek	10,000,000	ns	10,000	us	10	ms	20× datacenter roundtrip
Read 1MB sequentially from disk	20,000,000	ns	20,000	us	20	ms	80× memory, 20× SSD
Send packet CA->Netherlands->CA	150,000,000	ns	150,000	us	150	ms	

Let

Memory is the future

- Business-driven
- Data-driven
- The maturity of the technical conditions

Memory is the future

- Driven by business
- Driven by the amount of data
- The

Tape is Dead, Disk is Tape, Flash is Disk, RAM Locality is King.

Jim Gray, 2006

Oracle In Memory Option

- First introduced in 12.1.0.2 release
- Accelerate data analysis, not for oltp
- The other IMDB product of oracle , timesten , for oltp
- Dual-Format: Column and Row
- Oracle optimizer is smart

Oracle In Memory Option

- The data consistency between the two formats
- The data in IM column format only resides in RAM
- In 12CR2 can sync the data in column format to disks

Oracle In Memory Option

- IBM DB2 BLU Acceleration is very similar to Oracle IMDB in the dual-format architecture
- SAP HANA dual-format architecture, but cannot be both simultaneously
- Oracle perfect ? pay some price for the data consistency between row and column format

Why the IMDB is high performance

Why high performance

- Column
- SIMD
- Compression
- Data Skipping

Why is high performance

- Column Format
- SIMD
- Compression
- Data Skipping

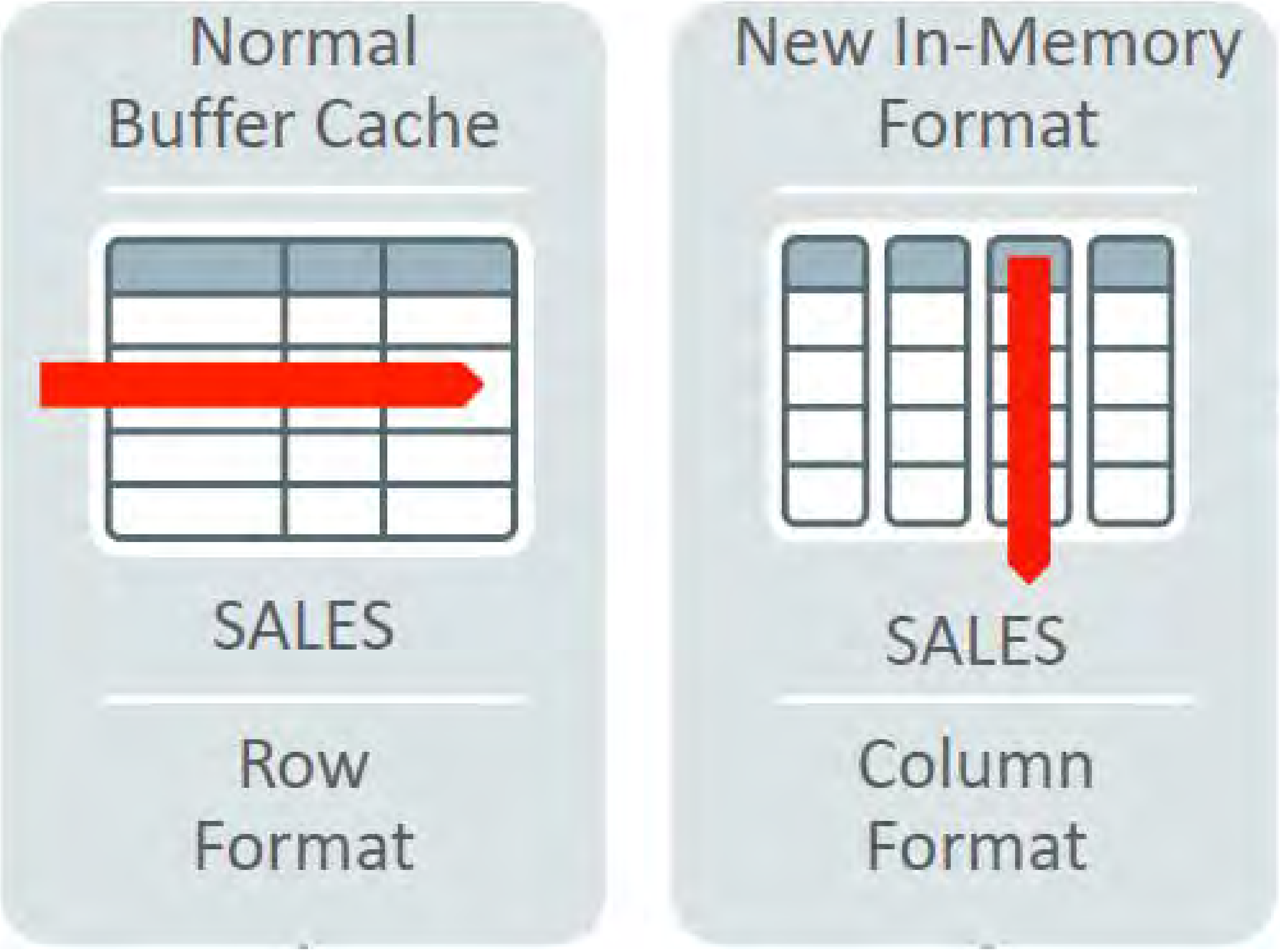
Why IM high performance

- Column Format
- SIMD
- **Compression**
- Data Skipping

Why IM high performance

- Column Format
- SIMD
- Compression
- Data

Column Format



- Column data tightly packed together
- Improve access efficiency
- Reduce memory traffic
- The cost of accessing to any column is the same

The cost of acessing to different columns

```
SQL> desc wrh
 Name          Type
 ------------- ---------------
---
 ID1            NUMBER
 ID2            NUMBER
 ID3            NUMBER
 ID4            NUMBER
 ID5            NUMBER
 ID6            NUMBER
 ID7            NUMBER
 ID8            NUMBER
 ID9            NUMBER
 ID10          NUMBER
 ID11          NUMBER
 ID12          NUMBER
 ID13          NUMBER
 ID14          NUMBER
 ID15          NUMBER
 ID16          NUMBER
 ID17          NUMBER
 ID18          NUMBER
 ID19          NUMBER
 ID20          NUMBER
```

Cached the table in Oracle buffer cache, and populated it into IM

Count the total number of rows for column 3, 6, 9, 12, 15, 18 and 20 respectively

```
select count(ID3)  from wrh where id1>1 and id2<1000000;
select count(ID6)  from wrh where id1>1 and id2<1000000;
select count(ID9)  from wrh where id1>1 and id2<1000000;
select count(ID12) from wrh where id1>1 and id2<1000000;
select count(ID15) from wrh where id1>1 and id2<1000000;
```

Let

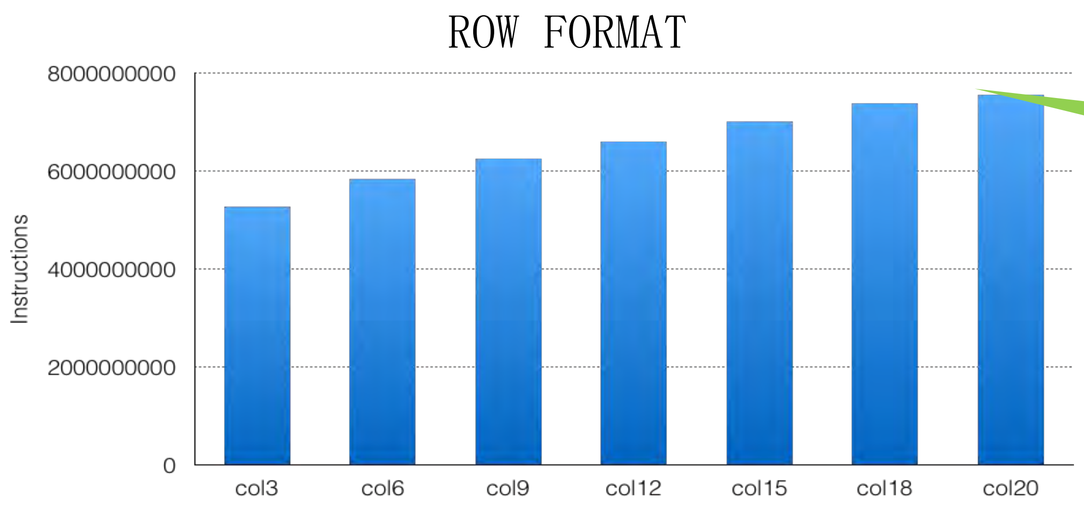
CPU Performance Counters on Linux

```
# perf stat -d -p 26031 sleep 5

 Performance counter stats for process id '26031':

         11.767587      task-clock (msec)         #    0.002 CPUs utilized
                 3      context-switches          #    0.255 K/sec
                 1      cpu-migrations            #    0.085 K/sec
               374      page-faults               #    0.032 M/sec
        14,850,049      cycles                    #    1.262 GHz                      (51.26%)
         9,410,174      stalled-cycles-frontend   #   63.37% frontend cycles idle     (55.56%)

                                                  #    0.36  stalled cycles per insn  (66.22%)
         1,861,488      branches                  #  158.188 M/sec                    (66.23%)
            30,688      branch-misses             #    1.65% of all branches          (66.24%)
         4,916,126      L1-dcache-loads           #  417.768 M/sec                    (21.91%)
         1,054,064      L1-dcache-load-misses     #   21.44% of all L1-dcache hits    (17.10%)
           299,978      LLC-loads                 #   25.492 M/sec                    (24.23%)
           240,057      LLC-load-misses           #   80.02% of all LL-cache hits     (32.41%)
```

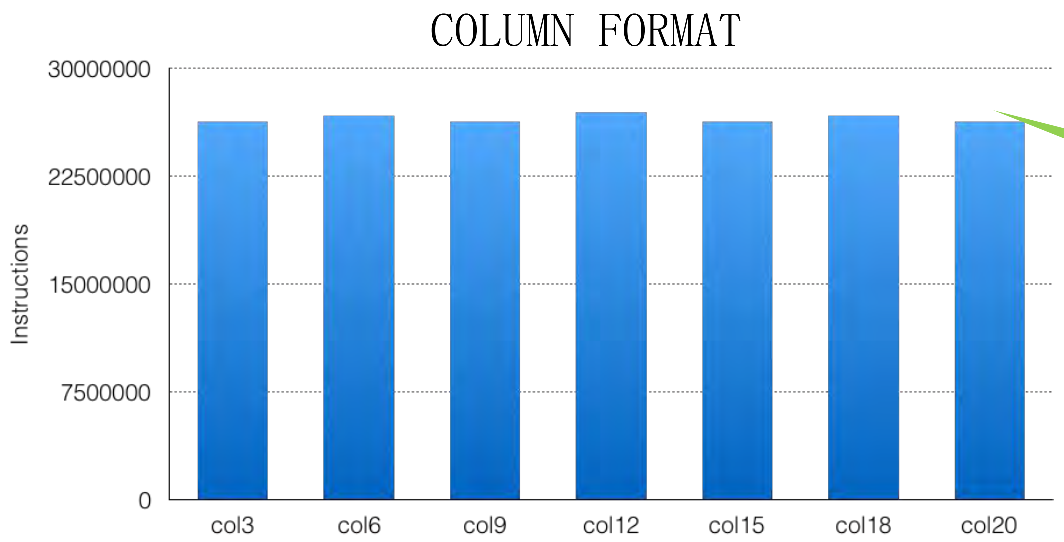
Measure what's going on inside a CPU!

Let

CPU instructions consumed



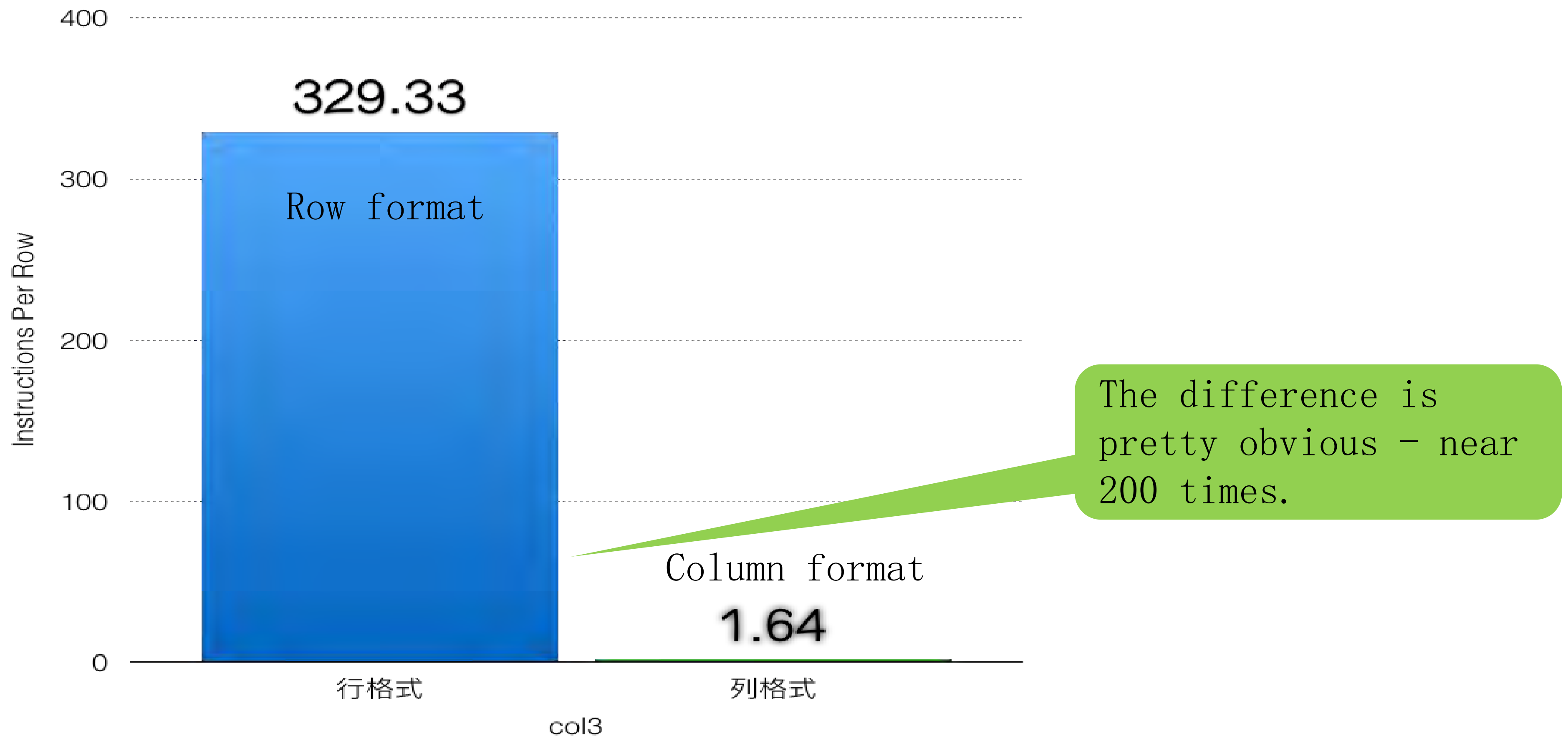
The later the column is, the more instructions it consumes



Stays the same when accessing different columns

Let

CPU instructions consumed by each row



SIMD Vector Processing

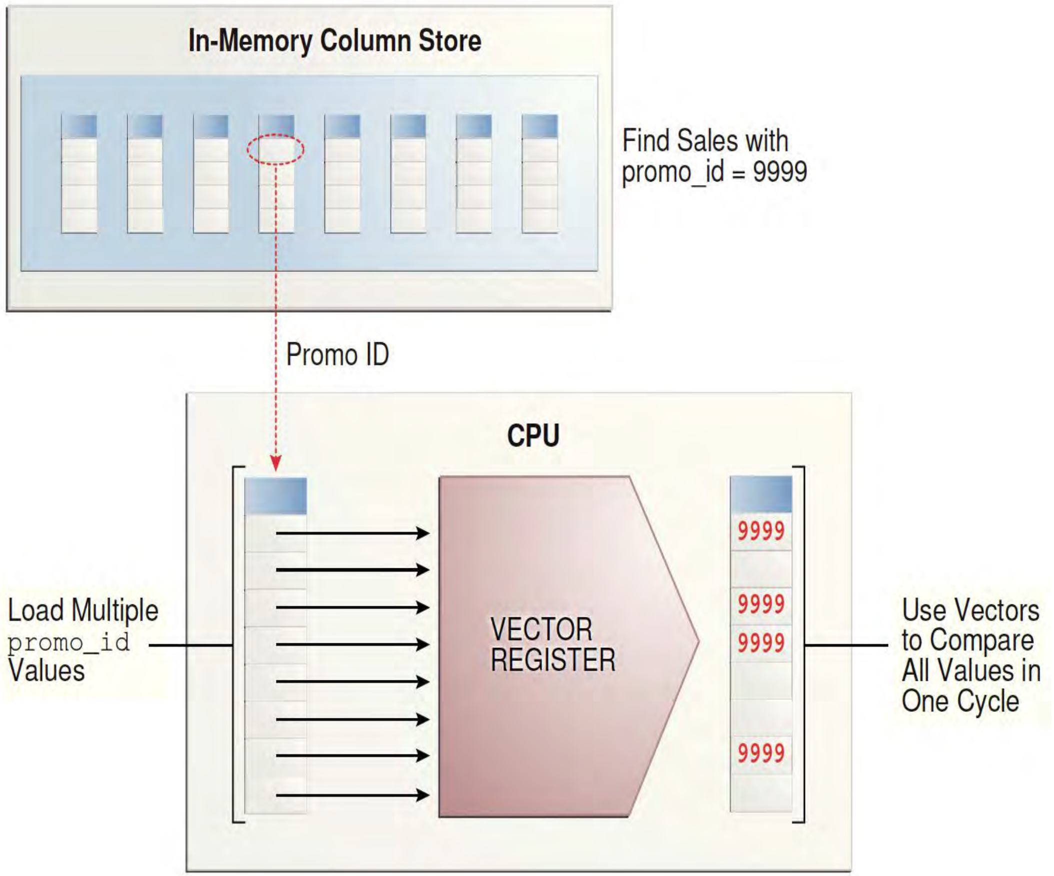
- SIMD, Single instruction, multiple data
- Get the CPU to simultaneously process multiple values in a vector
- Modern Intel CPUs Have 16-32 SIMD registers
- Applies only to column format
- The columnar data is packed tightly together,

take full advantage of the CPU features such as SIMD, superscalar, the friendly data structure is the key point.

The evolution of SIMD on Intel CPU

Instruction set	MMX	SSE	SSE2/SSE3/ SSSE3/SSE4	AVX/AVX2	AVX3 or AVX512
Register Size	64 Bits	128 bits	128 bits	256 Bits	512 bits
# Registers	8	8	16	16	32
Register Name	MM0 to MM7	XMM0 to XMM7	XMM0 to XMM15	YMM0 to YMM15	ZMM0 to ZMM31
Processors	Pentium II	Pentium III	Pentium IV to Nehalem	Sandy Bridge - Haswell	Skylake
Other		Only four 32 bits single precision floating point numbers	Usage expansion (two 64 bits double precision, four 32 bits integers and up to sixteen 8 bits bytes)	Three operand instructions (non destructive) : A+B=C rather than A=A+B Alignements requirements relaxed	

SIMD Vector Processing



The CPU evaluates the data as follows:

1. Loads the first 8 values from the promo_id column into the SIMD register, and then compares them with the value 9999 in a single instruction.
2. Discards the entries.
3. Loads another 8 values into the SIMD register, and then continues in this way until it has evaluated all entries.

Which SIMD extension does your CPU support?

```
$ grep "^model name" /proc/cpuinfo | sort | uniq
model name      : Intel(R) Xeon(R) CPU E5-4627 v2 @ 3.30GHz

$ grep ^flags /proc/cpuinfo | egrep "avx|sse " | sed 's/ /\n/g' | egrep "avx|sse " | sort | uniq
avx
sse
sse2
sse4_1
sse4_2
ssse3
```

In my environment, support AVX and SSEx extensions, does not support AVX2, AVX512 extensions.

Which extension is Oracle actually using?


```
$ pmap 8527 | grep libshpk
00007feeeb310000   2484K r-x--  /u01/app/oracle/product/12.2.0/dbhome_1/lib/libshpkavx12.so
00007feeeb57d000   2044K -----  /u01/app/oracle/product/12.2.0/dbhome_1/lib/libshpkavx12.so
00007feeeb77c000    132K rw---  /u01/app/oracle/product/12.2.0/dbhome_1/lib/libshpkavx12.so
```

In my environment the AVX has been used by oracle.

Which extension is Oracle actually using?


```
$ find
/u01/app/oracle/product/12.2.0/dbhome_1/rdbms/admin/libshpkavx12.def
/u01/app/oracle/product/12.2.0/dbhome_1/rdbms/admin/libshpkavx212.def
/u01/app/oracle/product/12.2.0/dbhome_1/lib/libmkl_avx512.so
/u01/app/oracle/product/12.2.0/dbhome_1/lib/libmkl_avx512_mic.so
/u01/app/oracle/product/12.2.0/dbhome_1/lib/libmkl_vml_avx512.so
/u01/app/oracle/product/12.2.0/dbhome_1/lib/libshpkavx212.so
/u01/app/oracle/product/12.2.0/dbhome_1/lib/libmkl_vml_avx512_mic.so
```

- 12CR1, does not support AVX2, AVX512 extensions
- 12CR2, supports AVX2,but I am not sure about AVX512, through Oracle lib directory already exists AVX512 lib

Compression

There are two important benefits :

- Less memory traffic
- Decompression on - the - fly (probably) benefits from CPU L2/L3 cache

The general purpose of compression is to save space, But for IM it ‘s just a side effect

Less memory traffic

- CPU is faster
- RAM access is the bottleneck of modern computers
- Want to wait less? Do it less!

Decompression on - the - fly

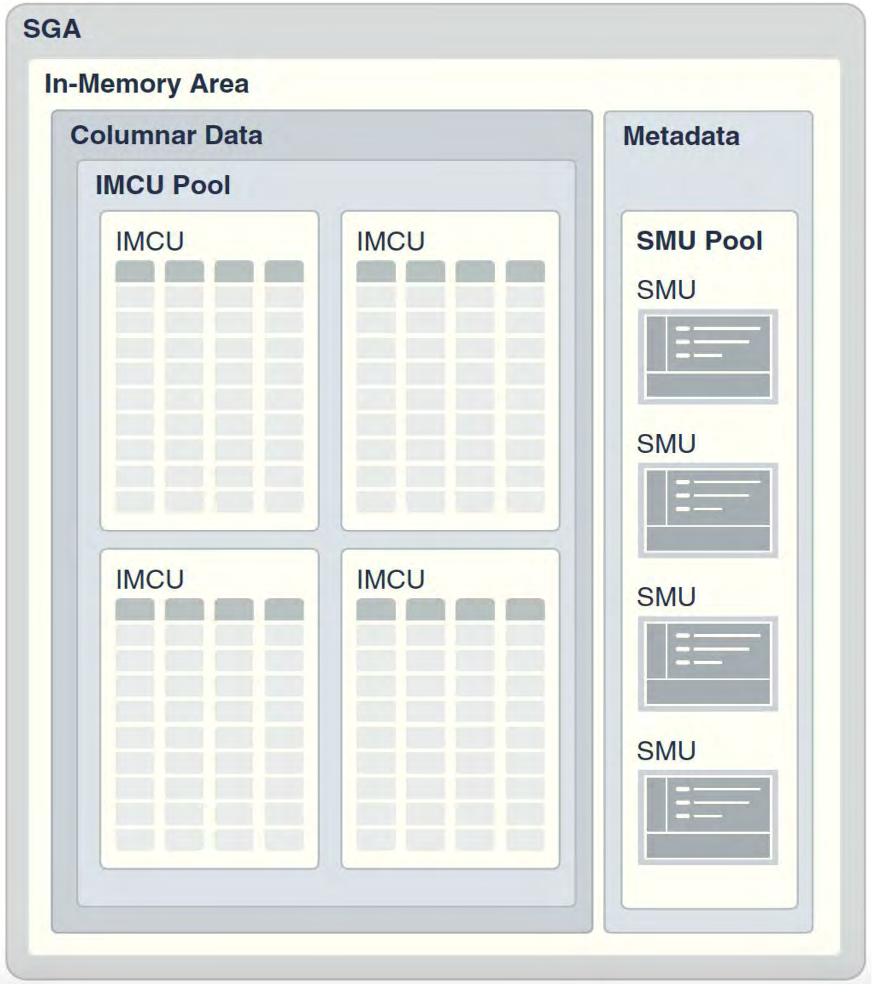
- Query can read the data without decompression
- Only decompress when the data need to return
- Read the compressed data can benefit from the CPU L2 / L3 cache
- Reducing memory writes

Different compression levels

COMPRESSION LEVEL	DESCRIPTION
NO MEMCOMPRESS	Data is populated without any compression
MEMCOMPRESS FOR DML	MEMCOMPRESS FOR DML
MEMCOMPRESS FOR QUERY LOW	Optimized for query performance (default)
MEMCOMPRESS FOR QUERY HIGH	Optimized for query performance as well as space saving
MEMCOMPRESS FOR CAPACITY LOW	Balanced with a greater bias towards space saving
MEMCOMPRESS FOR CAPACITY HIGH	Optimized for space saving

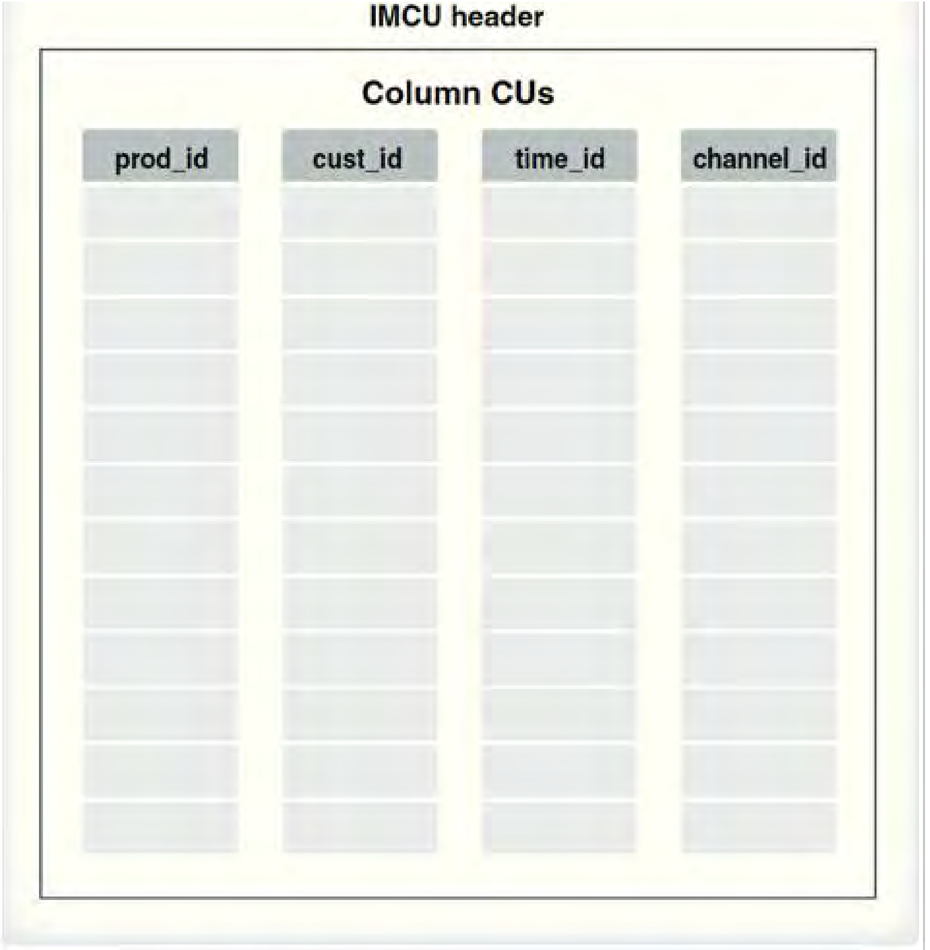
It is recommended to use the FOR QUERY compression algorithm, SQL queries execute directly on the compressed data

IMCU 、CU、 Local Dictionary



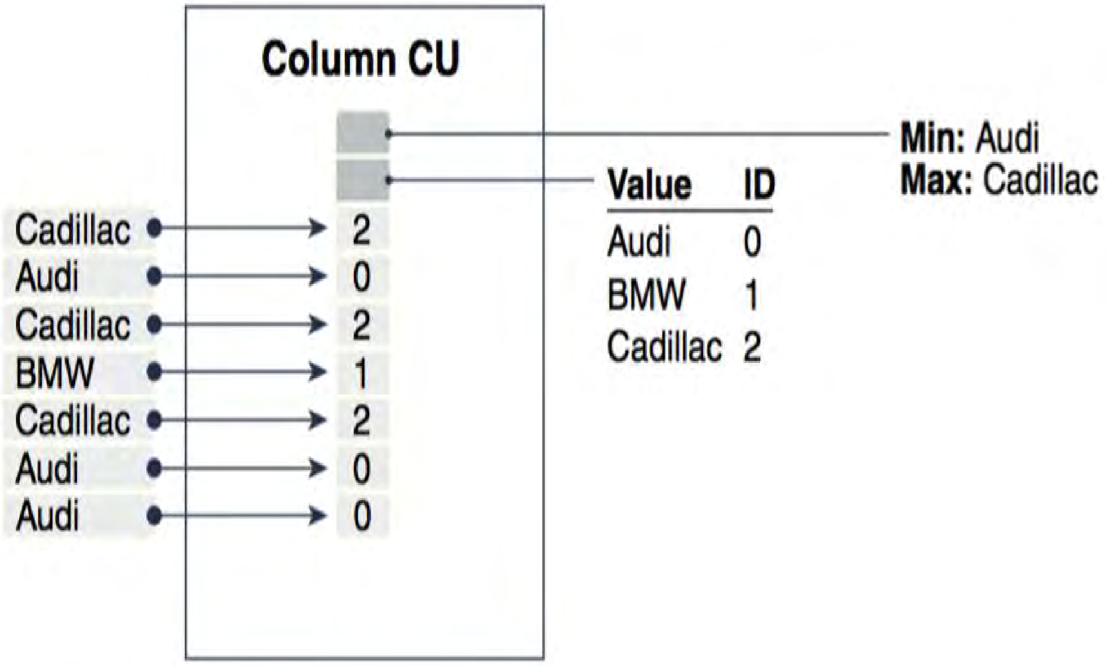
- The IM stores data for a single object (table, partition, materialized view) in a set of IMCUs.
- An IMCU stores columnar data for one and only one object.

IMCU 、CU、 Local Dictionary



- A Column Compression Unit (CU) is contiguous storage for a single column in an IMCU.
- Every IMCU has one or more CUs.

IMCU 、CU、 Local Dictionary



- A CU is divided into a body and a header
- The header contains metadata about the values stored in the CU body
- It may also contain a local Dictionary
- The local Dictionary is a sorted list of the distinct values in that column and their corresponding dictionary codes

Let

A exception of Local Dictionary

```
SQL> select /*+ parallel(16) */ count(*),count(distinct
id) from c1;

COUNT(*)          COUNT(DISTINCTID)
--------------         ------------------------------
138572154         138572154
```

The table c1 has only one column, its value is generated according to a sequence.

For primary columns, date columns, or the number of distinct value are very high columns, the local dictionary takes up a lot of space

A exception of Local Dictionary

Compression method	Original	After compression	Ratio
memcompress for query low	1688 M	2177 M	-30%

Bigger than original

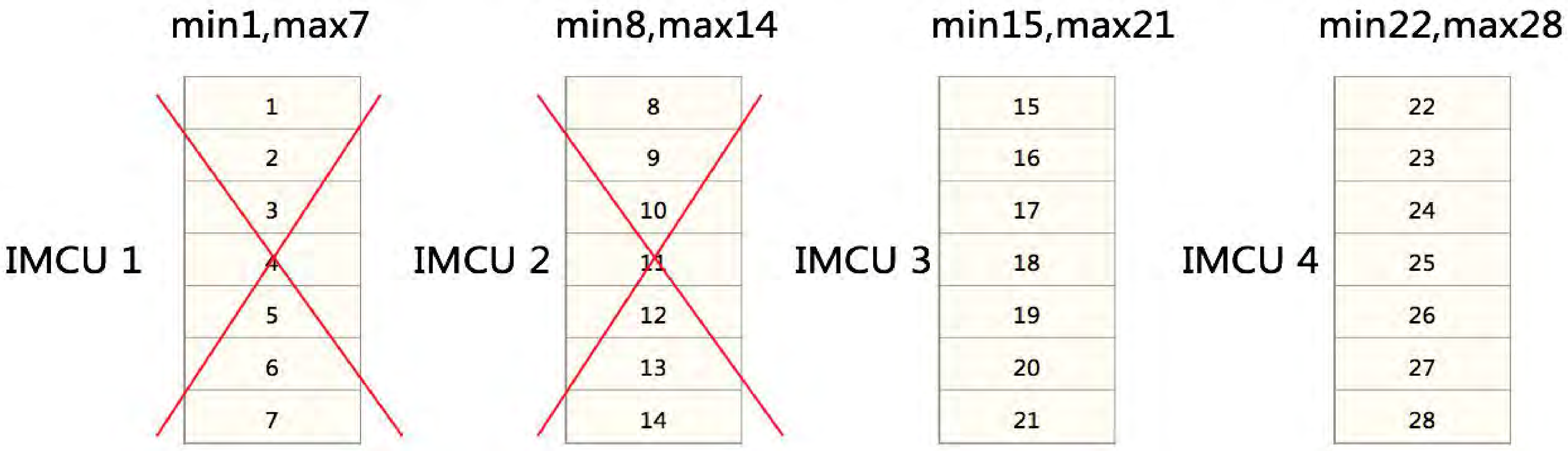
Data Skipping

- The traditional btree indexes have no advantage in the data analysis
- Data skipping technology the major memory database vendors have
- Tell the database quickly which blocks do not need to be accessed
- Automatically create and maintain, only exists in memory
- Reduce

Storage Index

- The Storage Index is already available in the first release of Exadata in 2008
- Now this feature has been migrated to IM
- Tell the database which blocks do not need to visit
- Each CU ‘s head records the maximum and minimum values

Storage Index



For example, for queries such as Where prod_id> 14 and prod_id <29, according to the maximum minimum information recorded by the CU header ,the IMCU 1 and IMCU 2 are skipped directly during the scan.

The impact of enable and disable storage index

Retrieve 10% rows out of a 20 GB table:

1.

```
select /*+ full(wxh) INMEMORY_PRUNING */count(object_name),count(object_type) from wxh where id>1 and id<10000000;
```

2.

```
select /*+ full(wxh) NO_INMEMORY_PRUNING */count(object_name),count(object_type) from wxh where id>1 and id<100000000;
```

The impact of enable and disable storage index

SQL	Elapsed Time
select /*+ full(wxh) INMEMORY_PRUNING */ count(object_name),count(object_type) from wxh where id>1 and id<10000000;	30 ms
select /*+ full(wxh) NO_INMEMORY_PRUNING */ count(object_name),count(object_type) from wxh where id>1 and id<100000000;	160 ms

Speed up 5x

- The (NO_)INMEMORY_PRUNING hint can enable/disable storage indexes.
- You haven' t a reason to disable the storage index in the production environment.

Comparing "in memory" with In-Memory

Retrieve 15% rows out of a 20 GB table:

1.

```
select /*+ full(wxh) */count(object_name) from wxh where object_id>1 and object_id<10000;
```

2.

```
select /*+ full(wxh) */count(object_name) from wxh where object_id>1 and object_id<10000;
```

Comparing "in memory" with In-Memory

The IM is 110 times faster than buffer cache

Metric	Buffer Cache	In-Memory	Ratio
task-clock (msec)	11684 ms	106 ms	**110**
cycles	41,357,721,213	179,523,876	230
instructions	41,453,927,963	425,354,074	97
in[illegible]s per cycle	1	2.37	0.42

Speed up 110x

CPU time

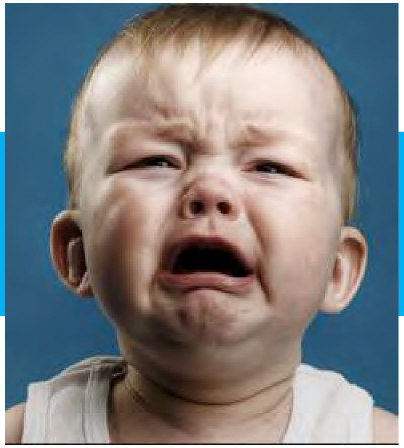
IM vs Parallel

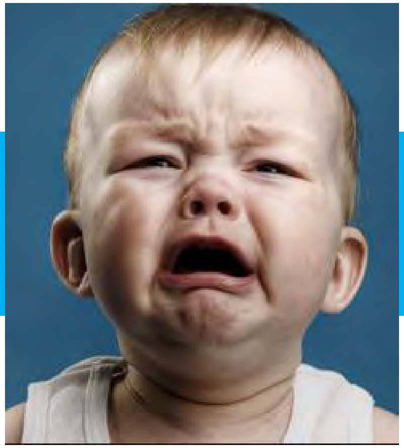
- Parallel Execution doesn’t mean “work smarter”
- You’re actually willing to accept to “work harder”
- IM is smart
- IM+Parallel is a best practice
  - More slaves
  - more PGA_AGGREGATE_TARGET
- IM has a Bigger IPC, insns per cycle , higher is better

When to Use Oracle Database In-Memory

Life cycle of a query:

Retrieve





- Column format
- Storage Index
- Compression
- SIMD

The Solution of accelerate data processing

DB GEEK

- **In-Memory** accelerates join operations through Bloom filter
- **Join** (12CR2 New feature) join groups eliminate the performance overhead of decompressing and hashing column values
- **In-Memory** also converts the join into a filter operation
- **Virtual** (12CR2 New feature), you can further improve performance for some CPU resource-intensive queries

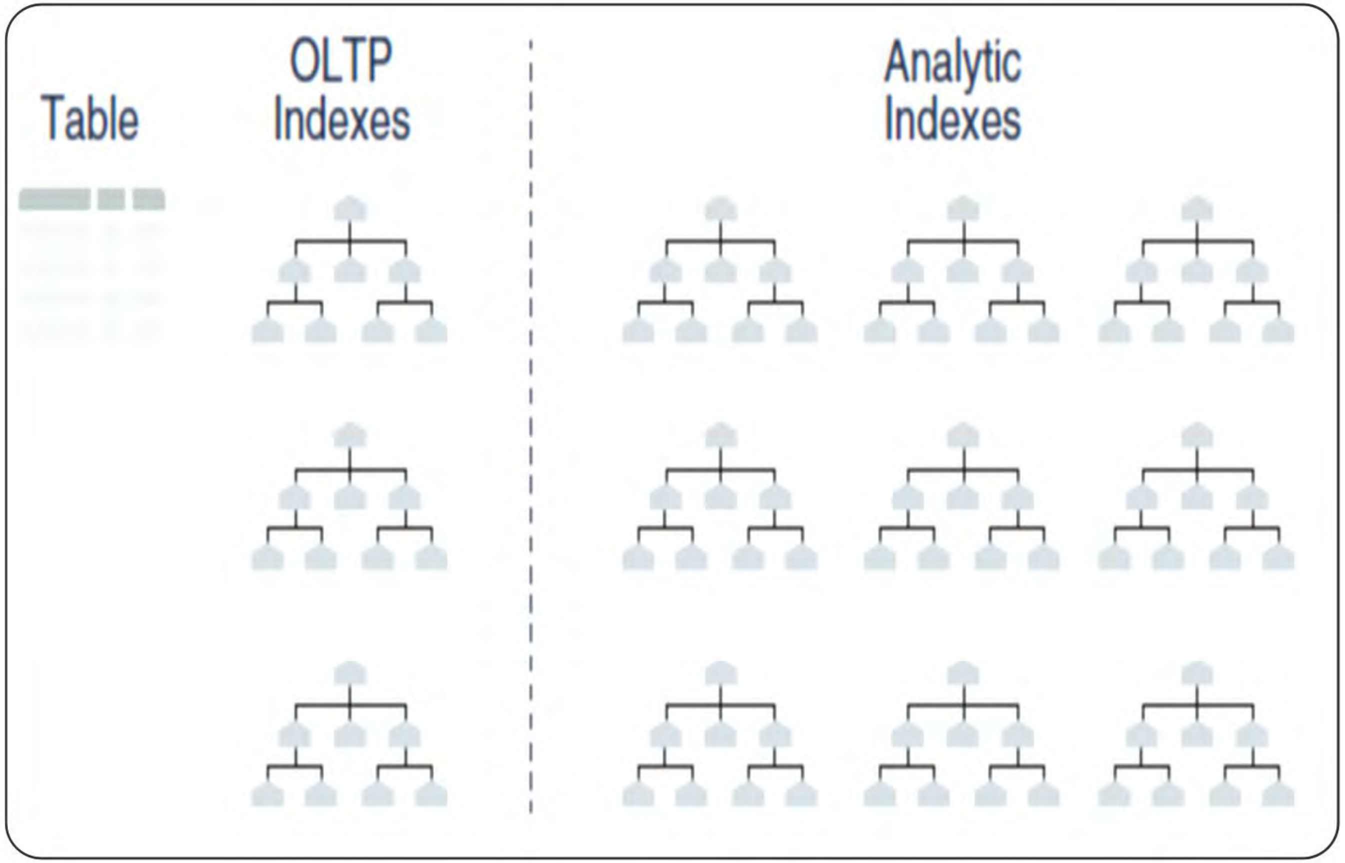
The

The Solution of accelerate data processing

? Why oracle try its best to convert the join into a filter operation

Because,Filter is very efficient through SIMD vector Processing.

The impact of enable IM on OLTP



- IM can indirectly improve the performance of OLTP system
- These analytic indexes are no longer needed
- Reduce the overhead of maintaining the indexes

A coin, there are always two sides

The impact of enabling IM on OLTP

DB GEEK

TPS drops by *20%* after enabling IM on related tables

Based on the swingbench tool, 25 concurrent users, 10G of data volume.

The Advantage of Oracle IM

- Column Format
- SIMD
- Data Skipping
- Compression

The technology about high performance these IM database vendors are using is similar

The Advantage of Oracle IM

	Oracle Database In-Momory 12.2.0.1	SAP HANA SPS11	Microsoft SQL Server 2016	IBM DB2 BLU 10.5	MemSQL 5.0
Release date	Apr 2017	November 2015	June 2016(TBD)	June 2013	March 2016
Columnar format	✓	✓	✓	✓	✓
Compression	✓	✓	✓	✓	✓
SIMD vector processing	✓	✓	✓	✓	✓
Data skipping	✓	✓	✓	✓	✓
In-memory aggregation	✓				
In-memory OLTP optimizations			✓		✓
Size not limited by DRAM	✓	✓	✓	✓	✓
Cluster(scale-out) support	✓	✓			✓
Dual-format in one database	✓			✓	
Consistent updates	✓	✓	✓		
100% application transparency	✓				
Persistence of Column Store	✓	✓	✓	✓	✓
Query on Secondary Replica	✓		✓		✓
Materialized View with Column Store	✓				
Integration with R	✓	✓	✓	✓	✓
Memory-Only columns		✓		✓	

The Advantage of Oracle IM

- Scalability
  - RAC
  - Active Data Guard
- Application transparency
- Mixed workload support

These enterprise features is real advantage of oracle IM

Thank You!