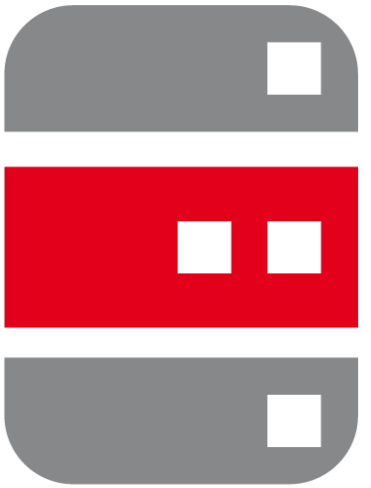
performing databases



Your reliability. Our concern.

performing databases
Your reliability. Our concern.

Oracle Core für Einsteiger: InMemory Column Store

Martin Klier

Performing Databases GmbH
Mitterteich

Referent

performing databases
Your reliability. Our concern.

- Martin Klier
- Lösungsarchitekt und Datenbankspezialist

- Fachliche Schwerpunkte:
  - Performanceoptimierung / Tuning
  - hochverfügbare Systeme
  - Architektur DBMS

- Linux seit 1997
- Oracle Database seit 2003

ORACLE ACE

Referent

performing databases
Your reliability. Our concern.

- Vorträge

IOUG
Independent oracle users group

COLLABORATE
April 2014

IOUG
independent oracle users group

COLLABORATE 12
TECHNOLOGY AND APPLICATIONS FORUM FOR THE ORACLE COMMUNITY

ORACLE OPEN WORLD

DOAG
Regionalgruppen

- Kontakt: martin.klier@performing-db.com
- Weblog: http://www.usn-it.de

Unternehmen

performing databases
Your reliability. Our concern.

- Spezialisten für Datenbanktechnik
  - Konzeptberatung und Vergabekompetenz
  - Architektur- und Systemplanung
  - Lizenzierung
  - Realisierung und Troubleshooting

- Kontakt
  - Performing Databases GmbH
    Wiesauer Straße 27
    95666 Mitterteich
  - Web: http://www.performing-databases.com
  - Twitter: @PerformingDB

Ziele

performing databases
Your reliability. Our concern.

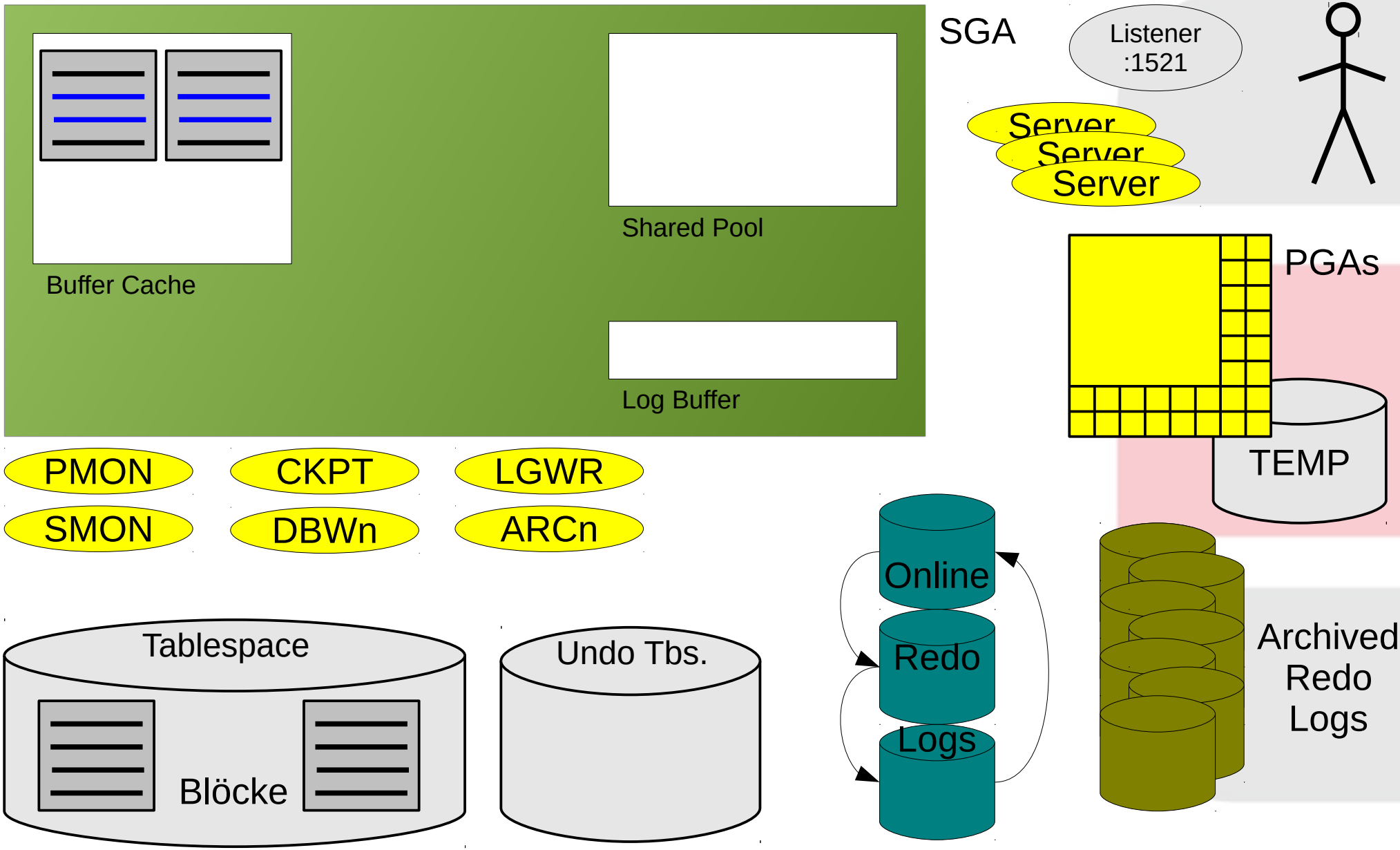
- Wiederholung Basisarchitektur
- Row- und Column basierte Daten
- Systemarchitektur mit InMemory Column Store
- Funktionsweise
  - Lesezugriffe
  - Transaktionen und Konsistenz
- Einsatzszenarien und Optimierungen

performing databases
Your reliability. Our concern.

Basics

Architekturschema
(vereinfacht)

performing databases
Your reliability. Our concern.



performing databases
Your reliability. Our concern.

Rows & Columns

Row Data

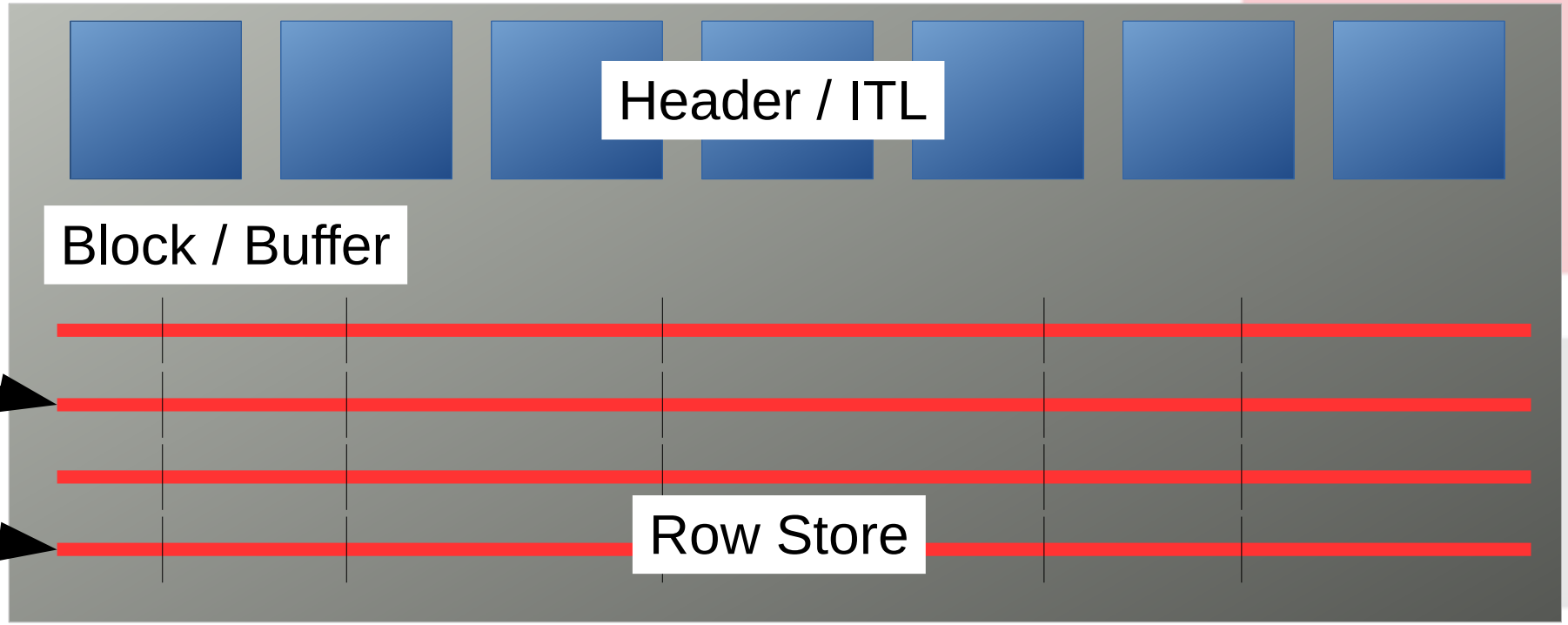
SELECT * … WHERE ID IN (3,36) ….

ID	NAME1	NAME2	STREET	ZIP	CITY
2	John	Doe	2 Wilbury Way	12345	Wesley
3	Alice	Allison	108 Agora Alley	23456	Amberg
4	Sally	Salinger	17 Samson St.	34567	Salisbury
..	..	..	..	..	..
..	..	..	..	..	..
..	..	..	..	..	..
36	..	..	..	..	..
..	..	..	..	..	..
57	Bob	Bobson	42 Century Ct.	56789	Taccanooga

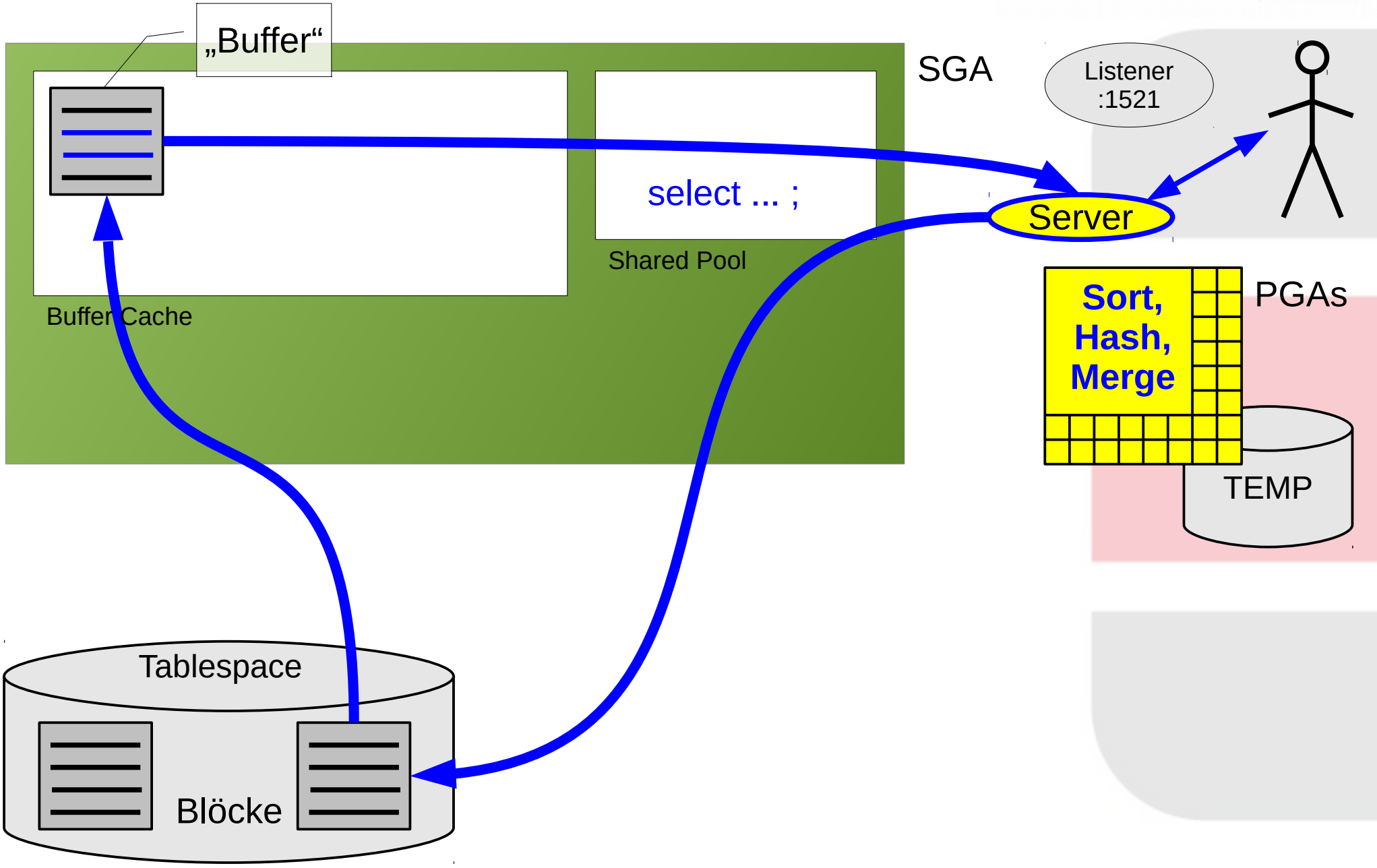
Row Data

performing databases
Your reliability. Our concern.

ID	NAME1	NAME2	STREET	ZIP	CITY
2	John	Doe	2 Wilbury Way	12345	Wesley
3	Alice	Allison	108 Agora Alley	23456	Amberg
4	Sally	Salinger	17 Samson St.	34567	Salisbury
8	Bob	Bobson	42 Century Ct.	56789	Taccanooga

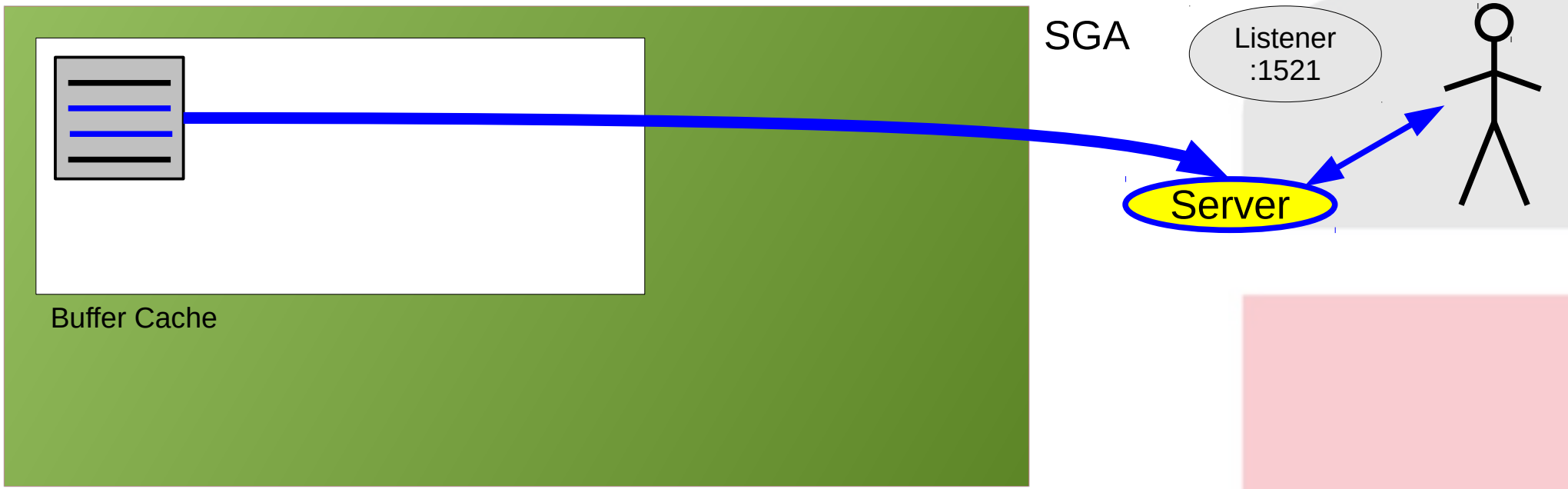


Lesevorgang Row



Lesevorgang Row

performing databases
Your reliability. Our concern.



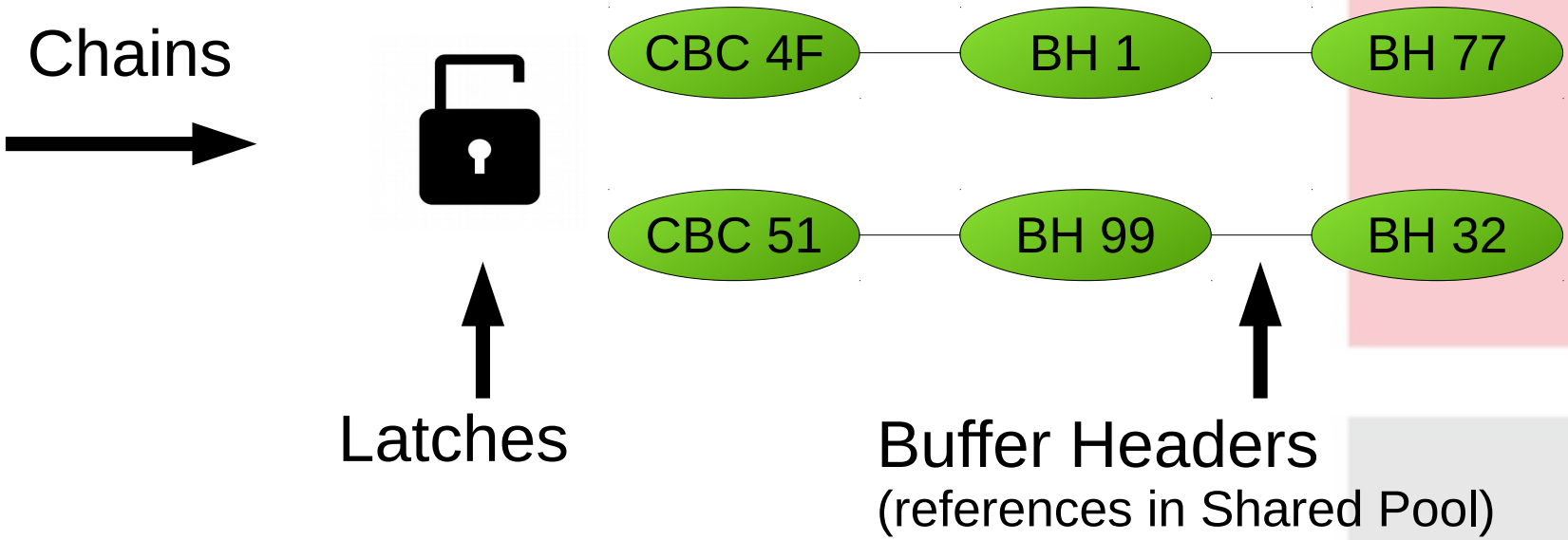
Maximum
1,000,000 rows per second
by design.

(Tirthankar Lahiri, Oracle VP)

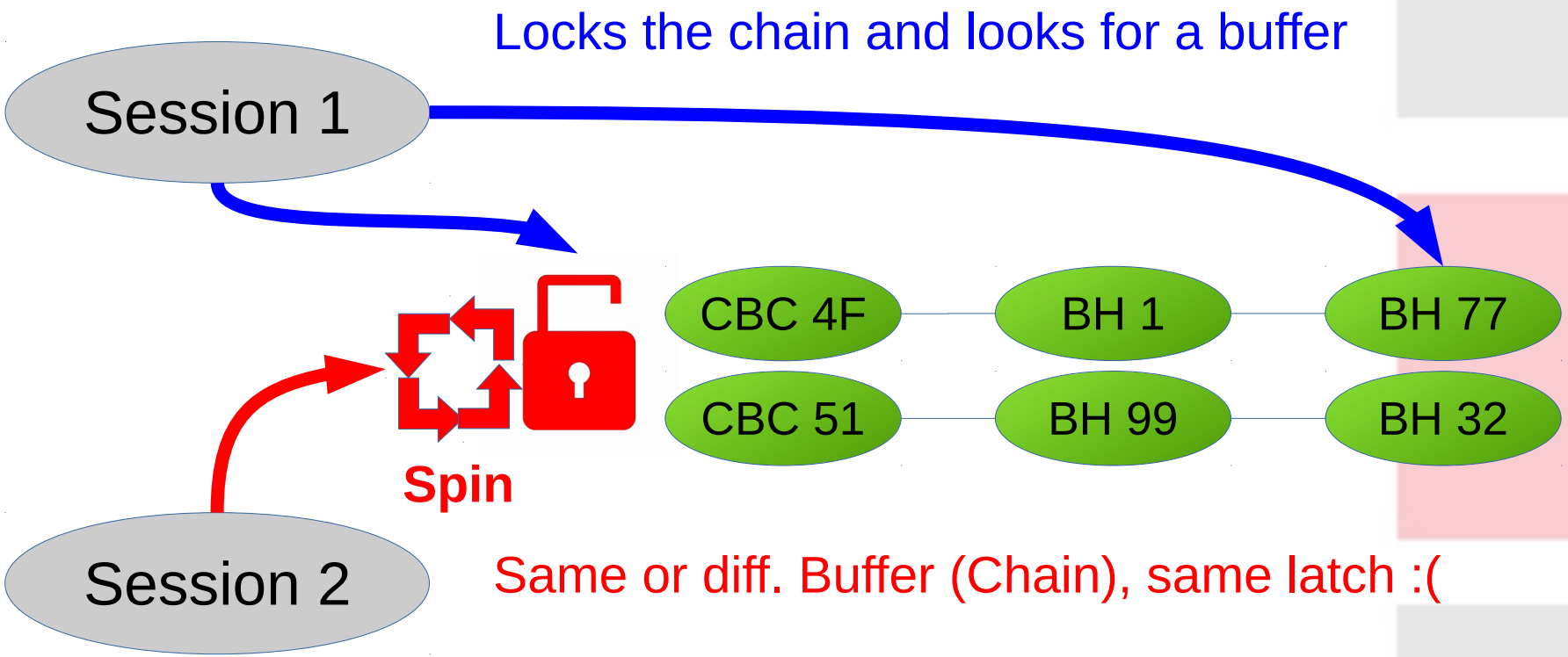
Buffer Cache Access

performing databases
Your reliability. Our concern.

Cache Buffer Chains: Is this block in the BC?



Buffer Cache Access



performing databases
Your reliability. Our concern.

Rows & Columns

Columnar Data

SELECT ID …. WHERE ZIP ….;

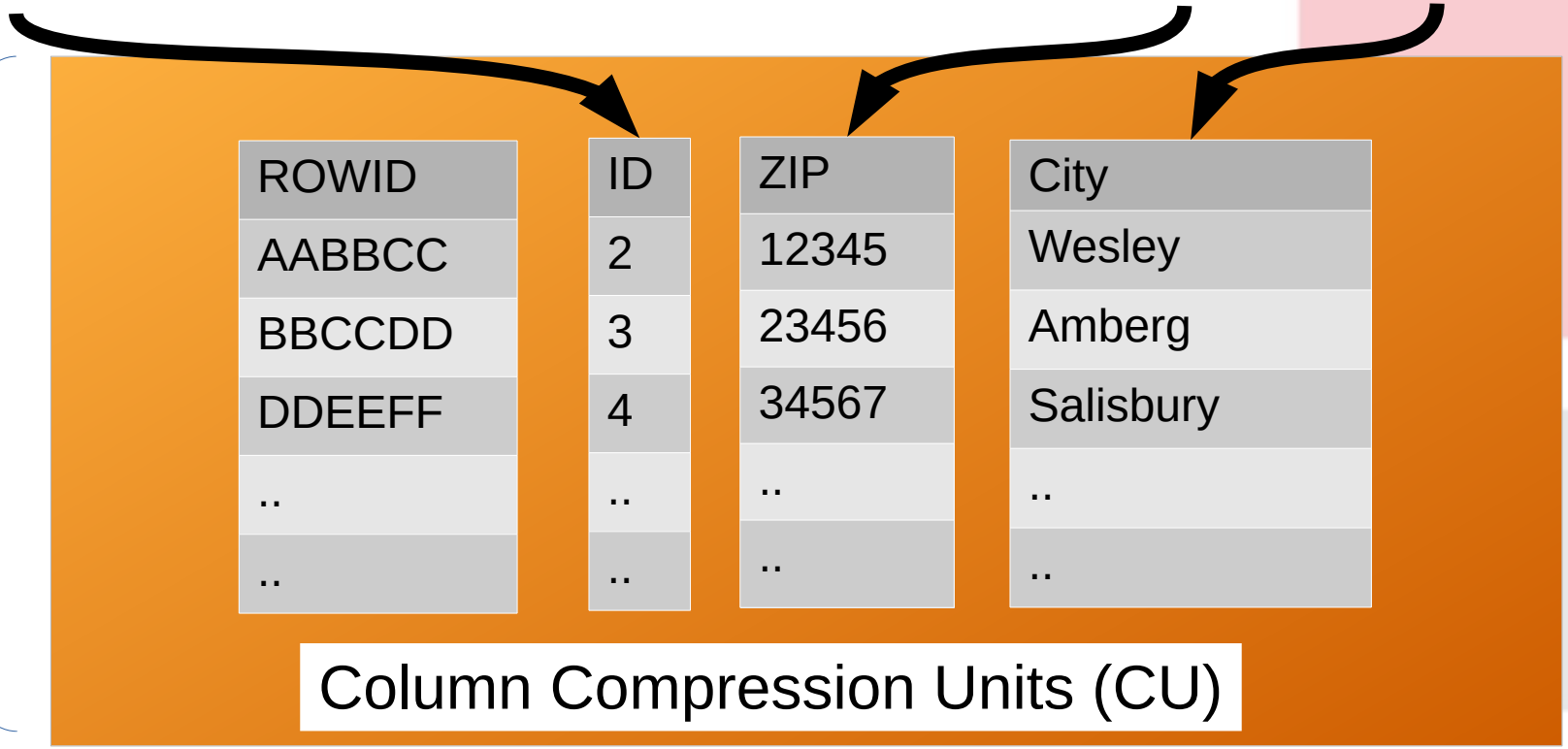
SELECT COUNT(*) …. WHERE ZIP ….;

ID	NAME1	NAME2	STREET	ZIP	CITY
2	John	Doe	2 Wilbury Way	12345	Wesley
3	Alice	Allison	108 Agora Alley	23456	Amberg
4	Sally	Salinger	17 Samson St.	34567	Salisbury
..	..	..	..	..	..
..	..	..	..	..	..
..	..	..	..	..	..
36	..	..	..	..	..
..	..	..	..	..	..
57	Bob	Bobson	42 Century Ct.	56789	Taccanooga

Columnar Data

performing databases
Your reliability. Our concern.

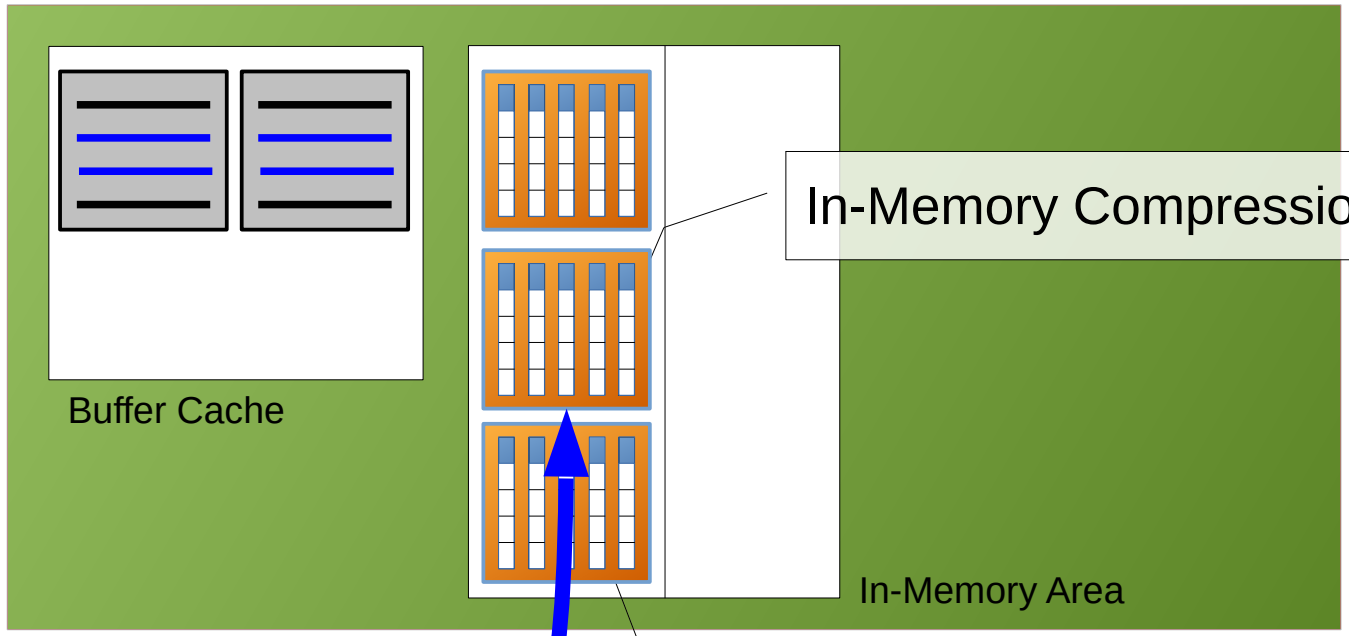
ID	NAME1	NAME2	STREET	ZIP	CITY
2	John	Doe	2 Wilbury Way	12345	Wesley
3	Alice	Allison	108 Agora Alley	23456	Amberg
4	Sally	Salinger	17 Samson St.	34567	Salisbury
..	..	..	..	..	..
..	..	..	..	..	..



performing databases
Your reliability. Our concern.

In-Memory Architecture

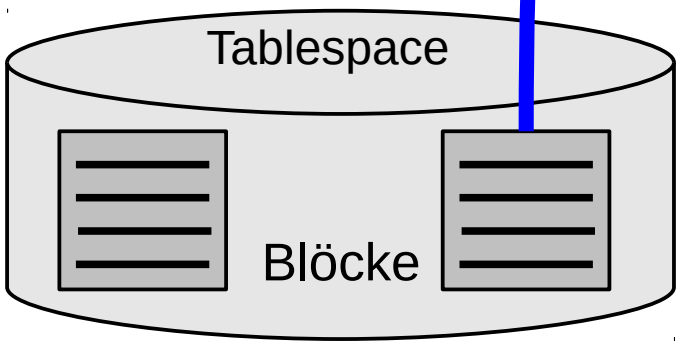
In-Memory Population



IMCO
W001
W002

IMCU- or 1MB pool

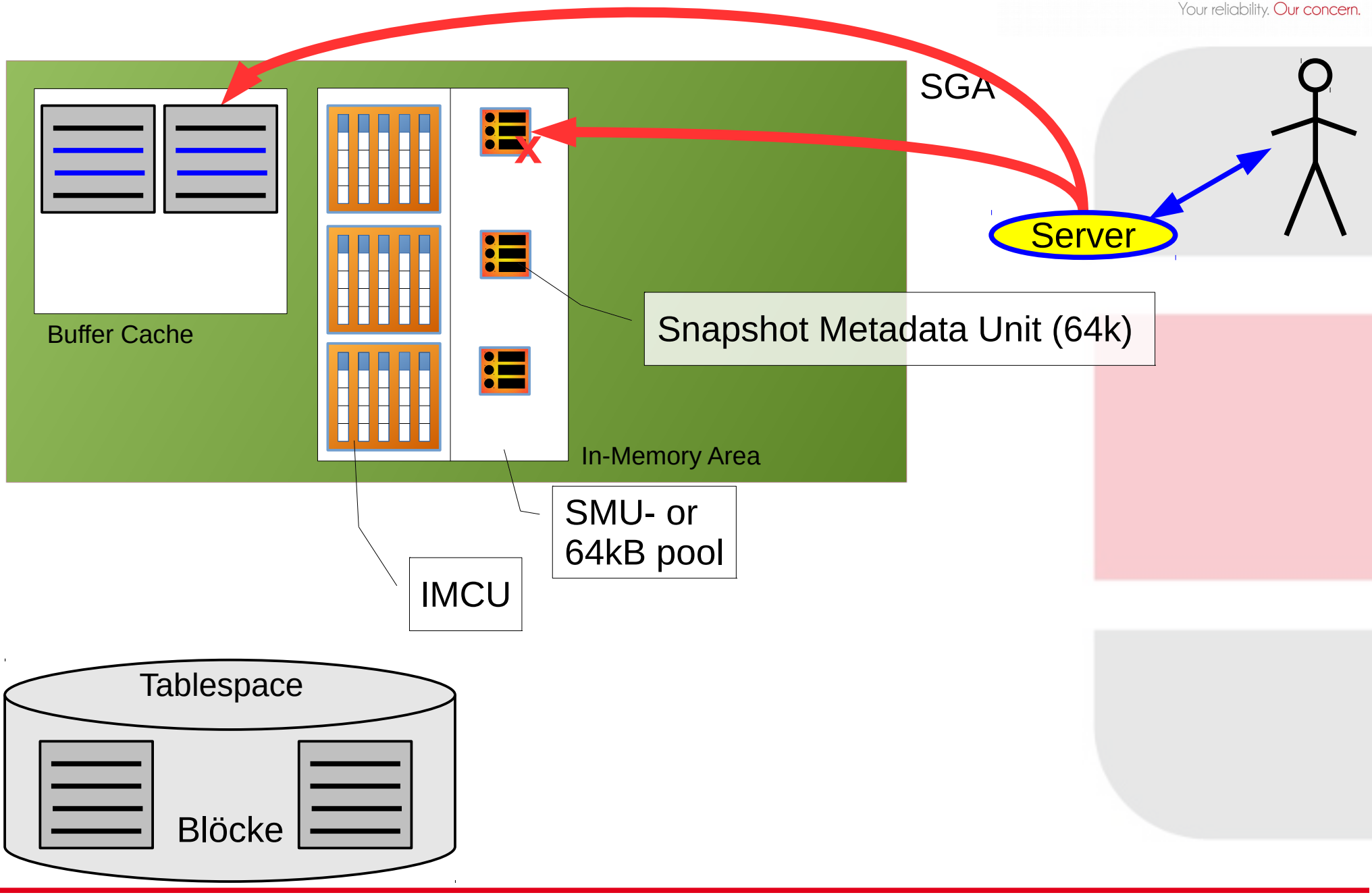
In-Memory Coordinator Process



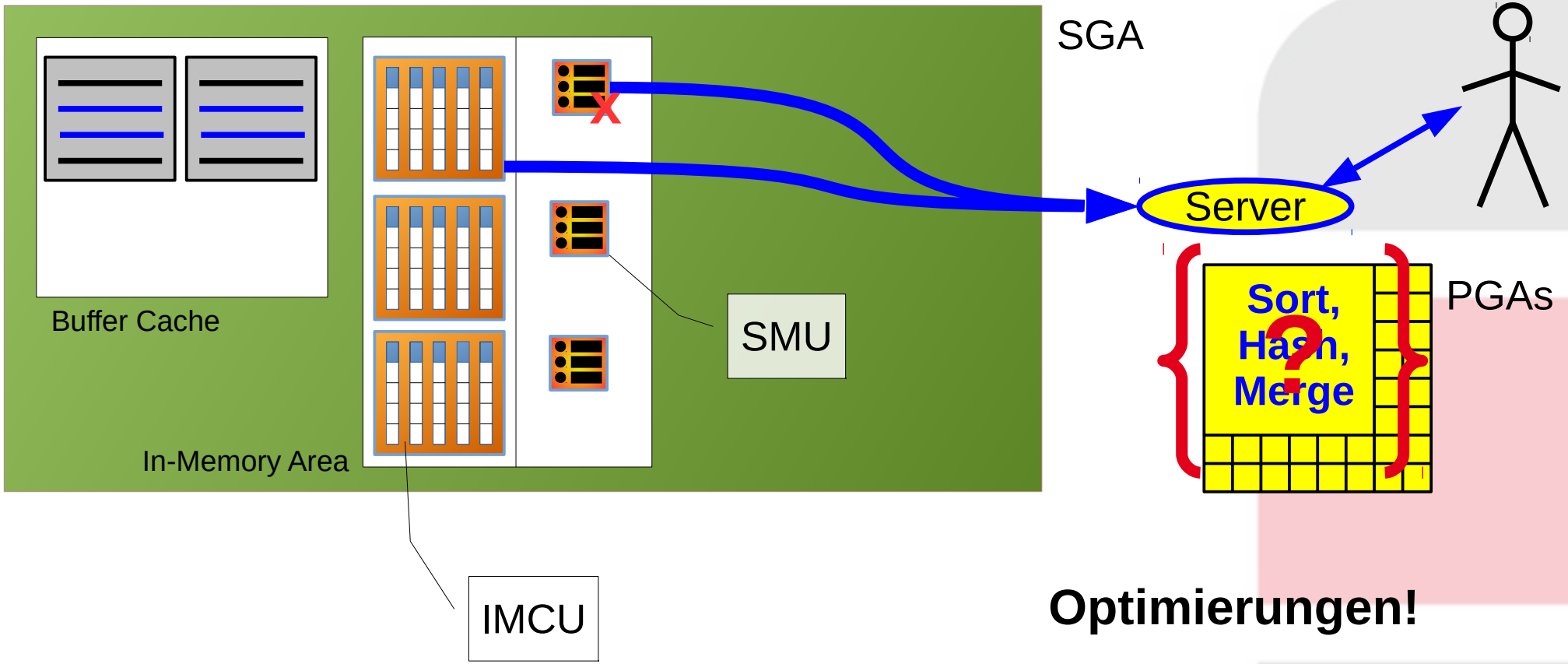
Order-Priority per table

- critical
- high
- medium
- low
- none (=on demand ?)

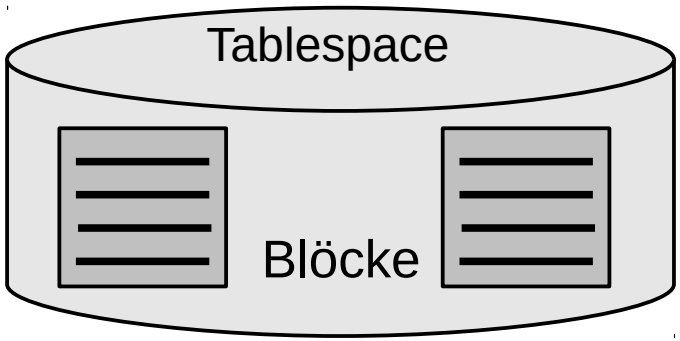
In-Memory Data Change



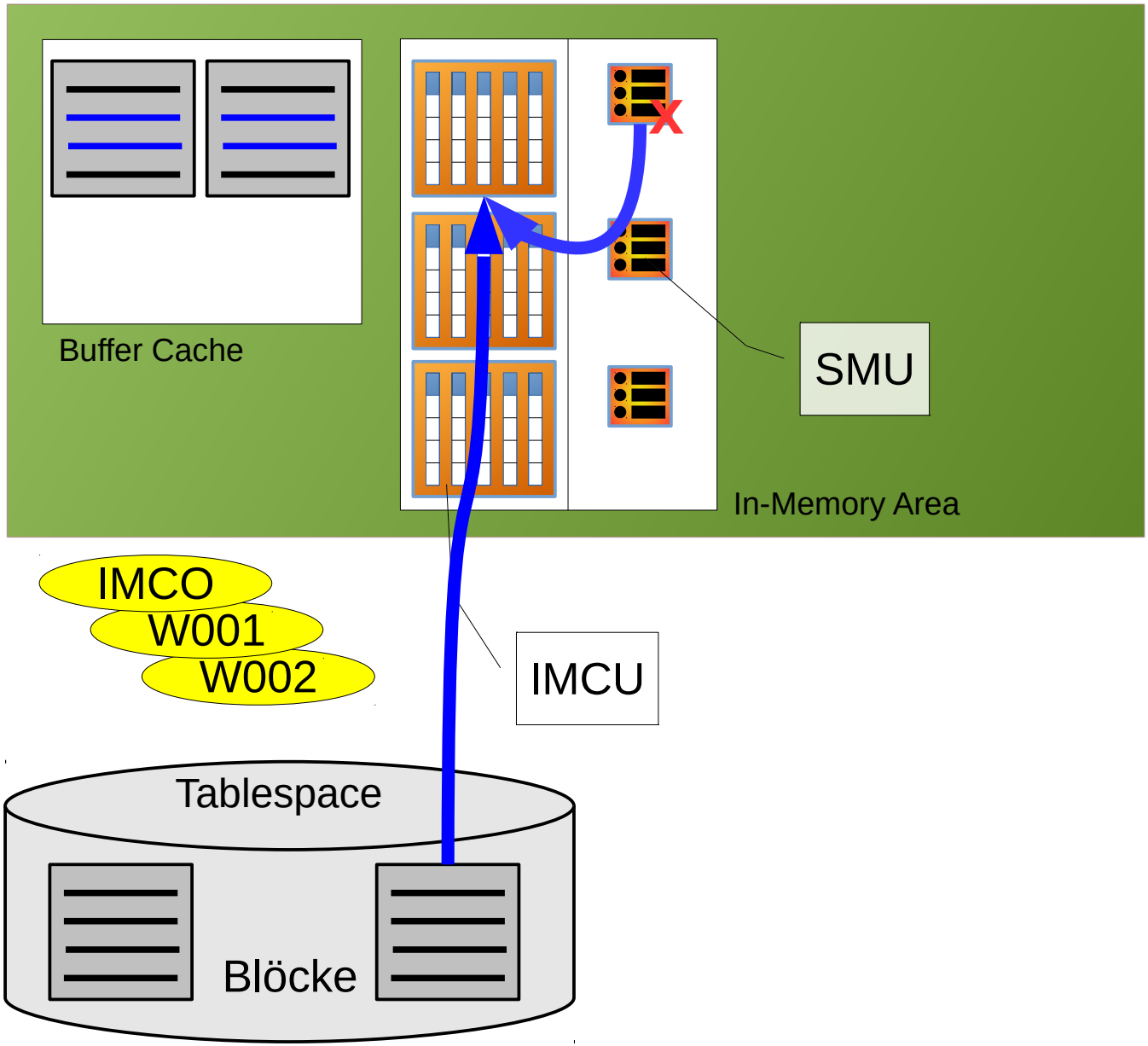
Lesevorgang Columnar



Optimierungen!



In-Memory RePopulation



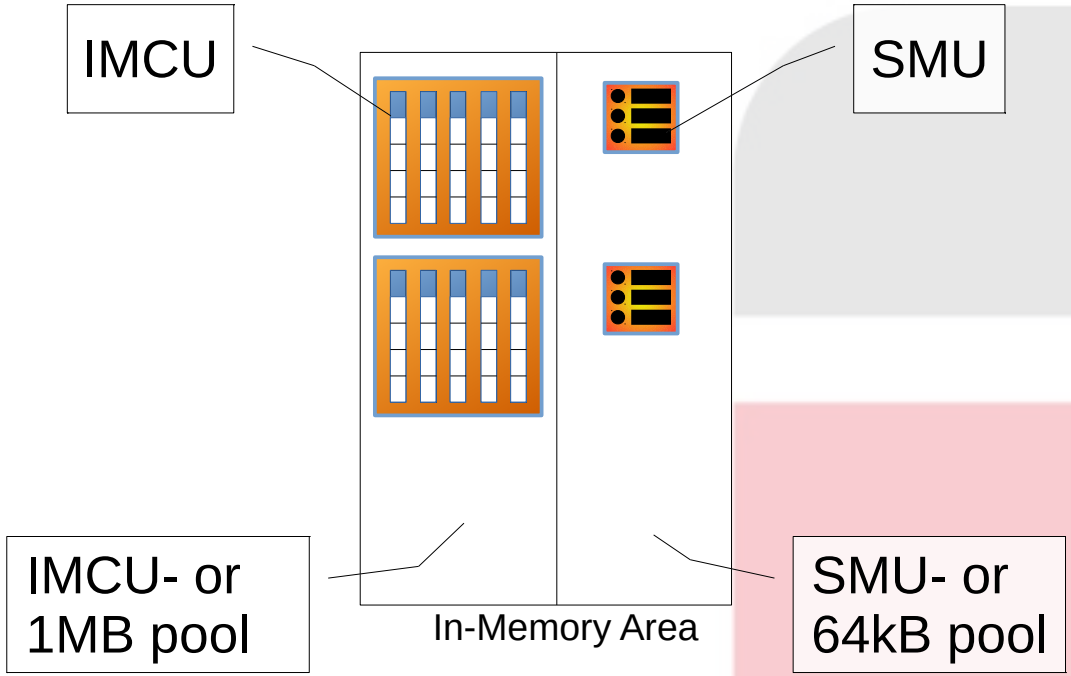
When?

- Threshold per IMCU
- Trickle (constant)

InMemory Area

performing databases
Your reliability. Our concern.

- Geteilt in
  - 1MB-Pool (IMCUs)
  - 64k-Pool (SMUs)



- **KEIN** Least-Recently-Used-Mechanismus (LRU)

  First come, first serve!

  => **Denke: „Tablespace“** nicht ~~„Cache“~~

performing databases
Your reliability. Our concern.

Using Column Store

With Or Without You

performing databases
Your reliability. Our concern.

```
select /*+ noparallel no_inmemory */ count(*) from imtable

Plan hash value: 1601702119

-------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |  26650   (1)| 00:00:02 |
|   1 |  SORT AGGREGATE    |         |     1 |             |          |
|   2 |   TABLE ACCESS FULL| IMTABLE |  5810K|  26650   (1)| 00:00:02 |
-------------------------------------------------------------------------
```

+

```
select /*+ noparallel */ count(*) from imtable

Plan hash value: 1601702119

-----------------------------------------------------------------------------------
| Id  | Operation                   | Name    | Rows  | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |         |     1 |   1140   (3)| 00:00:01 |
|   1 |  SORT AGGREGATE             |         |     1 |             |          |
|   2 |   TABLE ACCESS INMEMORY FULL| IMTABLE |  5810K|   1140   (3)| 00:00:01 |
-----------------------------------------------------------------------------------
```

When and When Not

performing databases
Your reliability. Our concern.

```
SQL_ID  a202cw0jn1trn, child number 0
-------------------------------------
select /*+ noparallel */ object_id, count(*) x from imtable group by
object_id

Plan hash value: 186096447

-------------------------------------------------------------------------------------------
| Id  | Operation                   | Name    | E-Rows |E-Bytes|E-Temp | Cost (%CPU)| E-Time   |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |         |        |       |       |  7741 (100)|          |
|   1 |  HASH GROUP BY              |         |  90895 |   443K|    66M|  7741   (4)| 00:00:01 |
|   2 |   TABLE ACCESS INMEMORY FULL| IMTABLE |   5810K|    27M|       |  1174   (5)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

+

```
SQL_ID  9n0h8a6c4jn5d, child number 0
-------------------------------------
select /*+ noparallel */ object_id, count(*) x  from imtable where
object_id=82345 group by object_id

Plan hash value: 1486712282

-----------------------------------------------------------------------------------------
| Id  | Operation             | Name    | E-Rows |E-Bytes| Cost (%CPU)| E-Time   |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |         |        |       |     3 (100)|          |
|   1 |  SORT GROUP BY NOSORT |         |     64 |   320 |     3   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN    | I_OID_1 |     64 |   320 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------
```

Operation Pushdown I


```
SQL_ID  91063q5ku7j07, child number 0
-------------------------------------
select /*+ noparallel */ object_id, count(*) x  from imtable where
object_id>70000 and object_id<85000 group by object_id

Plan hash value: 186096447

-------------------------------------------------------------------------------------------
| Id  | Operation                   | Name    | E-Rows |E-Bytes| Cost (%CPU)| E-Time   |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |         |        |       |  1207 (100)|          |
|   1 |  HASH GROUP BY              |         |  14854 | 74270 |  1207   (8)| 00:00:01 |
|*  2 |   TABLE ACCESS INMEMORY FULL| IMTABLE |    949K|  4636K|  1175   (6)| 00:00:01 |
-------------------------------------------------------------------------------------------

Query Block Name / Object Alias (identified by operation id):
-------------------------------------------------------------

   1 - SEL$1
   2 - SEL$1 / IMTABLE@SEL$1

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

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('12.1.0.2')
      DB_VERSION('12.1.0.2')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      FULL(@"SEL$1" "IMTABLE"@"SEL$1")
      USE_HASH_AGGREGATION(@"SEL$1")
      END_OUTLINE_DATA
  */

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

   2 - inmemory(("OBJECT_ID">70000 AND "OBJECT_ID"<85000))
       filter(("OBJECT_ID">70000 AND "OBJECT_ID"<85000))
```

Operation Pushdown II

performing databases
Your reliability. Our concern.

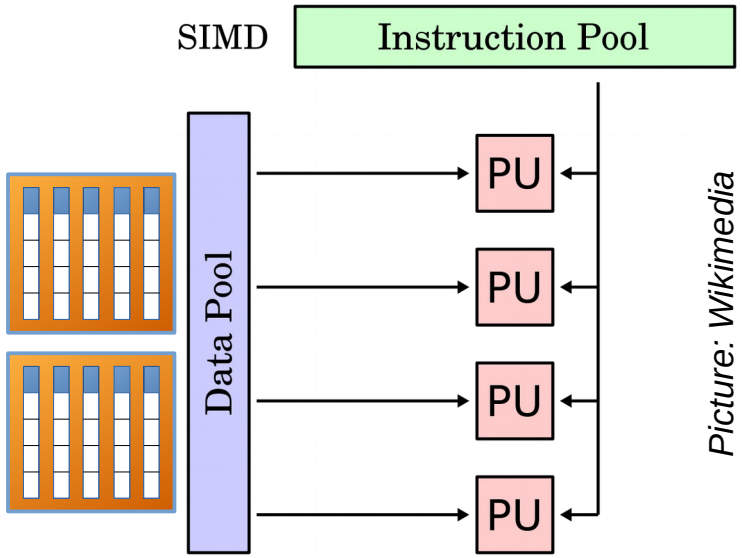
```
select /*+ noparallel */ count(*)
from bigthing bt, imtable imt
where bt.OBJECT_ID=imt.OBJECT_ID
  and imt.OBJECT_ID between 80000 and 85000
```

```
Plan hash value: 2908376589

-------------------------------------------------------------------------------------------------
| Id  | Operation                          | Name     | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |          |     1 |    10 |       | 11873   (5)| 00:00:01 |
|   1 |  SORT AGGREGATE                    |          |     1 |    10 |       |            |          |
|*  2 |   HASH JOIN                        |          |  8835K|    84M|  5264K| 11873   (5)| 00:00:01 |
|   3 |    JOIN FILTER CREATE              | :BF0000  |   316K|  1546K|       |  1173   (5)| 00:00:01 |
|*  4 |     TABLE ACCESS INMEMORY FULL     | IMTABLE  |   316K|  1546K|       |  1173   (5)| 00:00:01 |
|   5 |    JOIN FILTER USE                 | :BF0000  |  2537K|    12M|       |  8362   (6)| 00:00:01 |
|*  6 |     TABLE ACCESS INMEMORY FULL     | BIGTHING |  2537K|    12M|       |  8362   (6)| 00:00:01 |
-------------------------------------------------------------------------------------------------

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

   2 - access("BT"."OBJECT_ID"="IMT"."OBJECT_ID")
   4 - inmemory("IMT"."OBJECT_ID">=80000 AND "IMT"."OBJECT_ID"<=85000)
       filter("IMT"."OBJECT_ID">=80000 AND "IMT"."OBJECT_ID"<=85000)
   6 - inmemory("BT"."OBJECT_ID">=80000 AND "BT"."OBJECT_ID"<=85000 AND
              SYS_OP_BLOOM_FILTER(:BF0000,"BT"."OBJECT_ID"))
       filter("BT"."OBJECT_ID">=80000 AND "BT"."OBJECT_ID"<=85000 AND
              SYS_OP_BLOOM_FILTER(:BF0000,"BT"."OBJECT_ID"))
```

Optimierungen

performing databases
Your reliability. Our concern.

- Vector Processing (Single Instruction Multiple Data SIMD)
  z.B. Intel AVX / 256bit Register



Picture: Wikimedia

>1,000,000,000
rows per second
(Tirthankar Lahiri, Oracle VP)

Optimierungen

performing databases
Your reliability. Our concern.

- Operation Pushdown: InMemory Scan (*)
  (ähnlich Exadata Smart Scan)
  - Prädikat-Prüfung im Column Store (*)
  - Aggregation im Scan
  - Bloom Filter im IMCU Scan (*)

- In-Memory Storage Index
  (Auslassen unnötiger IMCUs bei In-Memory Scan)

- Predicate Optimization
  (wird eine IMCUs ohnehin vollständig benötigt, erfolgt auf ihr keine Prüfung des Prädikats)

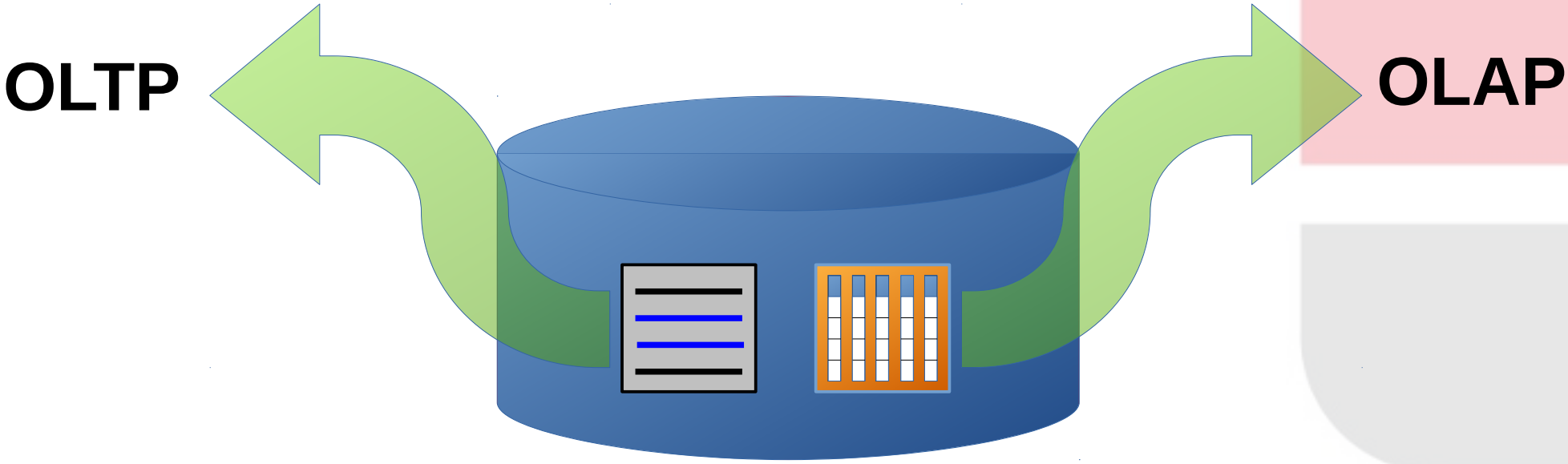
performing databases
Your reliability. Our concern.

Abschluß

Vorteile IM Column St.

performing databases
Your reliability. Our concern.

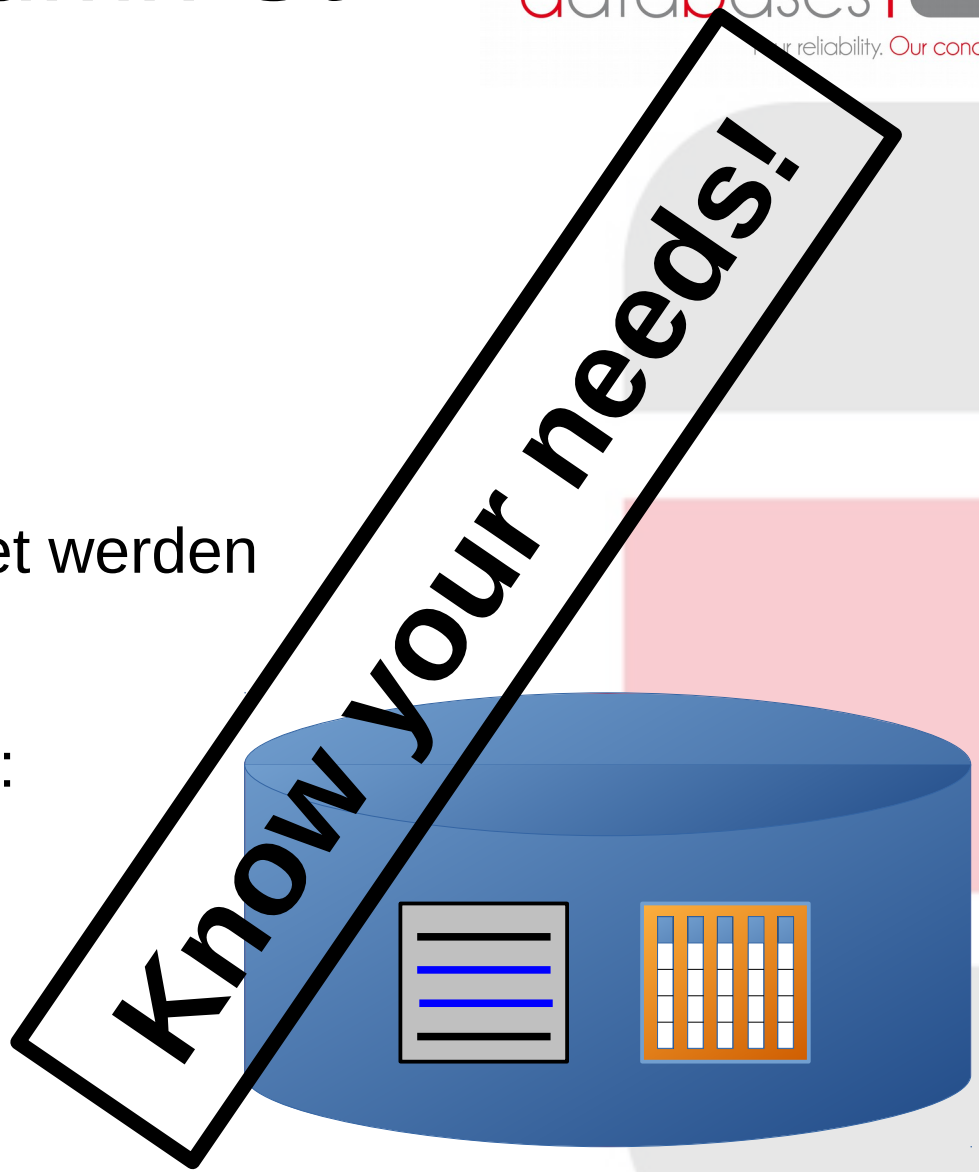
- Schneller bei spaltenbasierten Auswertungen
- Optimizer nutzt „Shortcuts“ der Architektur
- Voll transparent, ideal für Mischbetrieb auf selbem Datenbestand



Nachteile IM Column St.

performing databases
reliability. Our concern.

- Ressourcenbedarf
  - RAM
  - CPU für Compression
- Inhalt muss gepflegt / verwaltet werden
  => Advisor!
- Eingeschränkt deterministisch:
  Nutzen stark abhängig von
  - Workload
  - Inhalt
  - Zustand
- Skalierung im RAC ohne Ausfallsicherheit (Nur auf Engineered Systems)



performing databases
Your reliability. Our concern.

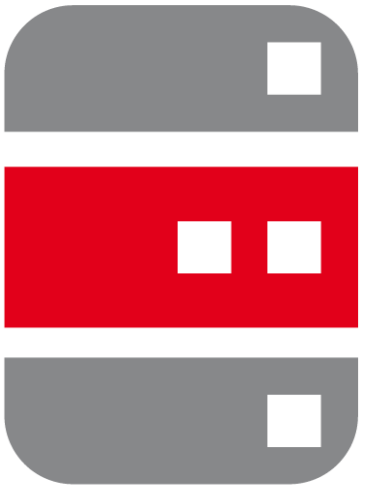
InMemory ist cool.
Cool reicht nicht.
UseCase muß passen.

#FiveWordTechHorrors „InMemory solves all performance problems“

performing databases
Your reliability. Our concern.

Download Präsentation und Whitepaper
http://www.performing-databases.com

performing databases



Your reliability. Our concern.