DTCC

第十四届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA

数智赋能 共筑未来

IT168.com ChinaUnix ITPUB

北京国际会议中心 | 2023/8/16-18

现代处理器之上的数据库

吕海波

个人介绍

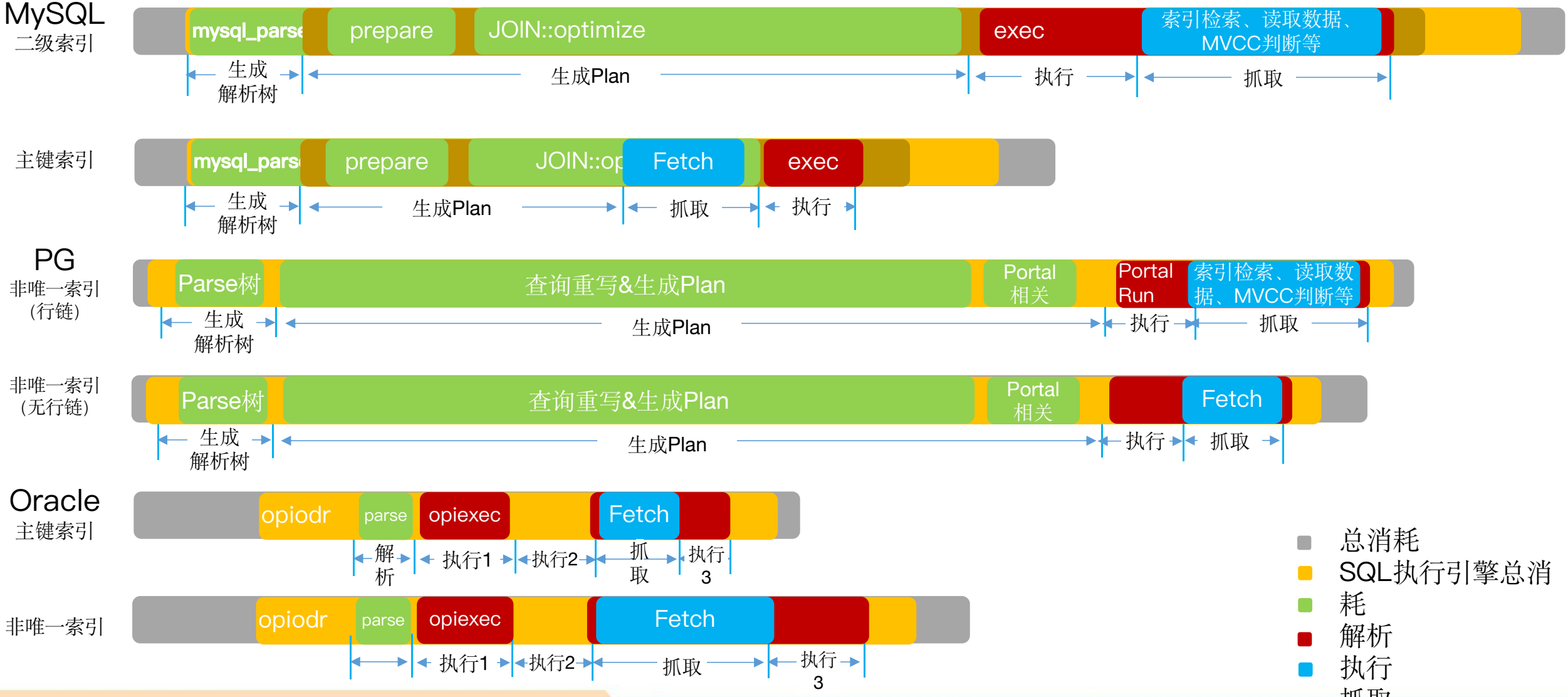
吕海波

北京大学 《开源软件开发基础与实践 – PostgreSQL数据库内核》课程校外导师

1996年进入IT行业，第一份工作是财务软件开发和财务软件讲师。至今27年软件行业从业经历，19年数据库相关工作经验。

曾在多家国内外巨头型互联网公司（阿里巴巴、京东、ebay、paypal）从事数据库管理与研究工作。出版技术书籍《Oracle内核技术揭密》，被誉为国内最深度解密Oracle算法原理的技术书籍。

现代处理器之上的数据库



现代处理器之上的数据库

PMC: Performance Monitor Counter, 性能监控计数器

PMU: Performance Monitor Unit, 性能监控单元

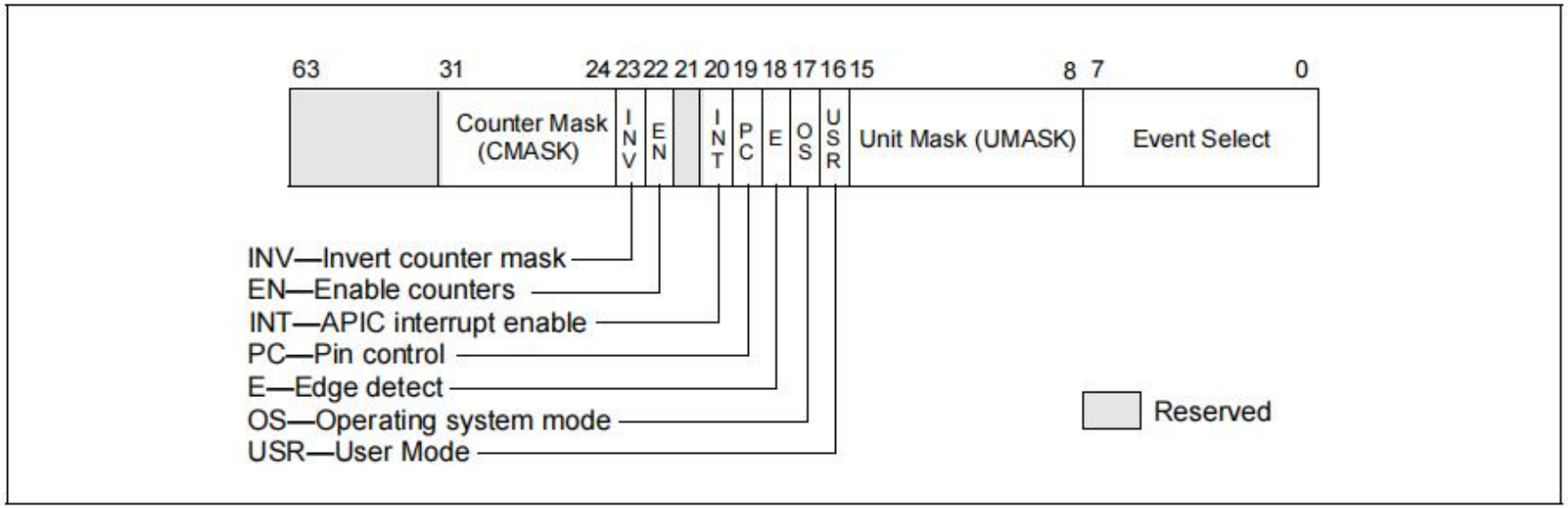


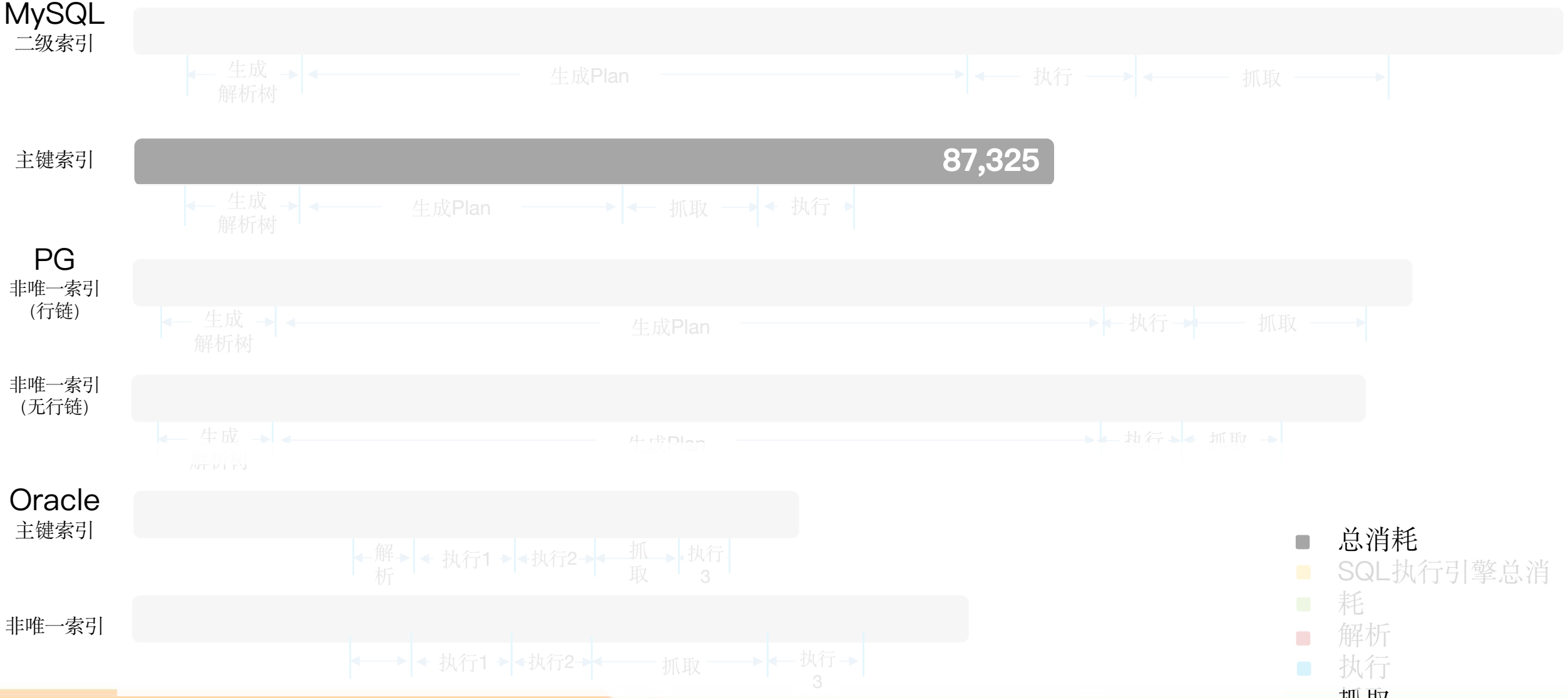
Figure 18-1. Layout of IA32_PERFEVTSELx MSRs

现代处理器之上的数据库

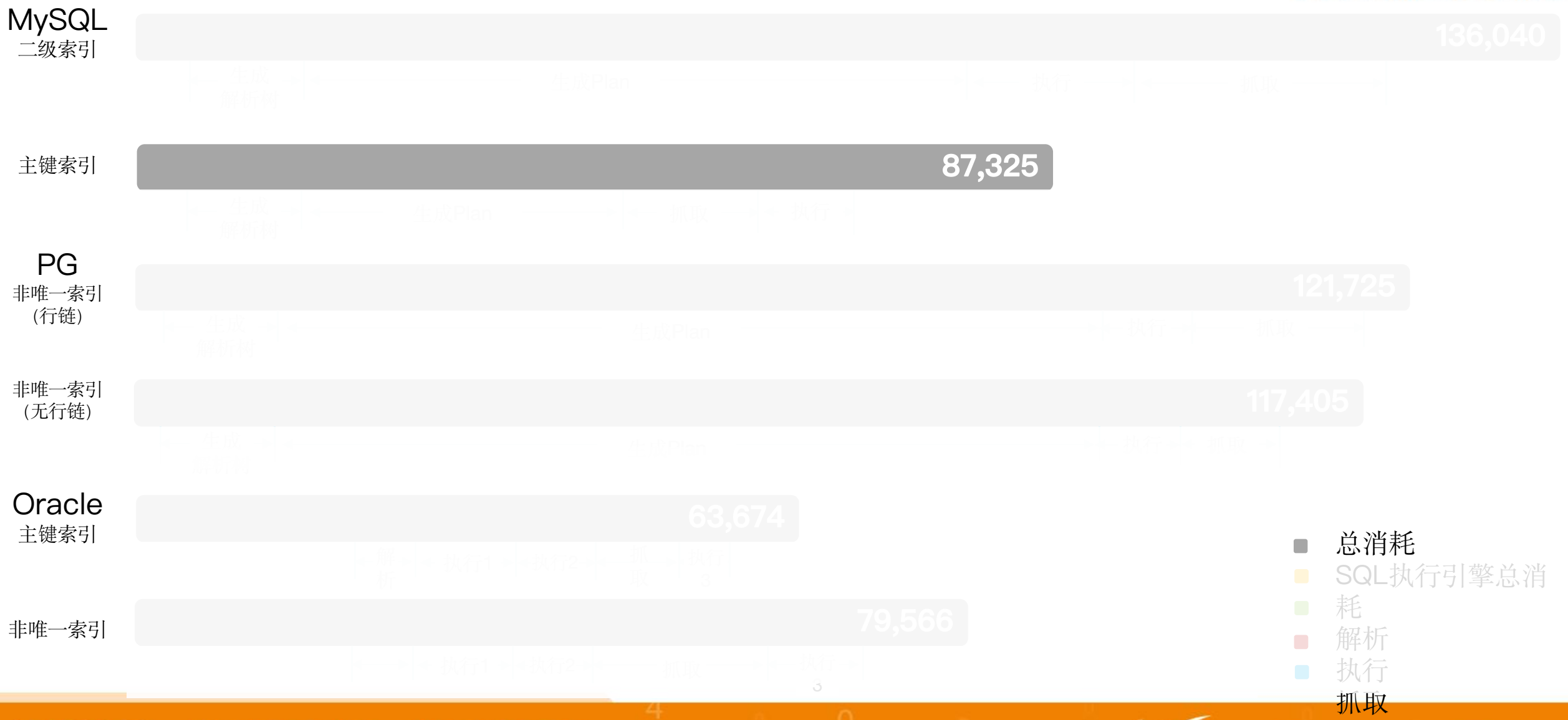

```
[root@localhost diting]# perf stat -e instructions:u,cycles:u -t 16564
^C
 Performance counter stats for thread id '16564':

          87,325      instructions:u             #    0.18  insn per cycle
         490,694      cycles:u

       1.472712575 seconds time elapsed

[root@localhost diting]#
```

```
mysql> select * from vage1 where id1=1;
+-----+----------------------+------+----------------------+
| id1 | c1                   | id2  | c2                   |
+-----+----------------------+------+----------------------+
|   1 | AAAAAAAAAAAAAAAAAAAA |  101 | BBBBBBBBBBBBBBBBBBB |
+-----+----------------------+------+----------------------+
1 row in set (0.00 sec)

mysql>
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select PROCESSLIST_ID, THREAD_OS_ID from performance_schema.threads where
IST_INFO%';
+----------------+--------------+
| PROCESSLIST_ID | THREAD_OS_ID |
+----------------+--------------+
|             14 |        16564 |
+----------------+--------------+
1 row in set (0.00 sec)

mysql>
```

现代处理器之上的数据库



现代处理器之上的数据库



➢ SQL执行过程

- 解析：生成解析树、查询树、计划树
- 执行：准备好沿着计划树，执行SQL
- 抓取：索引检索、读取行/列数据
- 其他：收发网络包，等，Others

现代处理器之上的数据库


```
\-----others
          |
          > mysql_parse
            |
            > JOIN::optimize
            |
            < JOIN::optimize
            |
            > JOIN::exec
                      |
                      > row_search_mvcc
                        |
                        > btr_cur_search_to_nth_level
                        < btr_cur_search_to_nth_level
                        > row_sel_get_clust_rec_for_mysql...
                        < row_sel_get_clust_rec_for_mysql
                        > row_sel_store_mysql_rec...
                        < row_sel_store_mysql_rec
                      < row_search_mvcc
                      |
            <JOIN::exec
            |
          < mysql_parse
          |
          others
```

解析

执行

抓取

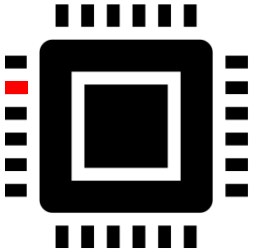
执行

现代处理器之上的数据库


```
\-----others
        |
        > mysql_parse
          |
          > JOIN::optimize
          |
          < JOIN::optimize
          |
          > JOIN::exec
                |
                > row_search_mvcc
                  |
                  > btr_cur_search_to_nth_level
                  < btr_cur_search_to_nth_level
                  > row_sel_get_clust_rec_for_mysql...
                  < row_sel_get_clust_rec_for_mysql
                  > row_sel_store_mysql_rec...
                  < row_sel_store_mysql_rec
                < row_search_mvcc
                |
          <JOIN::exec
          |
        < mysql_parse
        |
        others
```

JOIN::exec

JOIN::exec

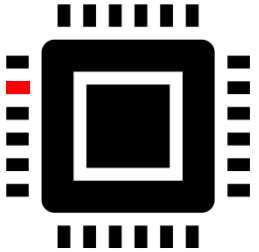


(1).
共用所有CPU计数器

(2).
继续目标程序的运行

现代处理器之上的数据库

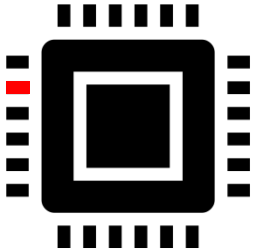
MySQL/PG/Oracle —— OS：Linux ——



ptrace

现代处理器之上的数据库

MySQL/PG/Oracle —— OS：Linux ——



ptrace

现代处理器之上的数据库

(1).
打开所需PMC计数器

(2).
继续目标程序的运行

现代处理器之上的数据库


```
\-----others
        |
        > pg_parse_query
        < pg_parse_query
        > pg_analyze_and_rewrite_fixedparams
        < pg_analyze_and_rewrite_fixedparams
        > pg_plan_queries
        < pg_plan_queries
        > PortalDefineQuery & PortalStart
        < PortalDefineQuery & PortalStart
        > PortalRun
          |
          > index_getnext_slot
                  |
                  < _bt_search
                  |
                  < _bt_search
                  > index_fetch_heap
                  < index_fetch_heap
                  > heap_hot_search_buffer
                  < heap_hot_search_buffer
          < index_getnext_slot
          |
        < PortalRun
        others
```

解析

执行

抓取

执行

现代处理器之上的数据库

opiodr

opi :接口层，oracle program interface。 odr: Oracle D…Request

kksParseCursor

火山模型
解析层

opiexec

qertbFetchByRowID

qerixFetchUniqueInde
x

qerflFetchOutside

qertbFetch(query execute rowsource table access fetch)

火山模型
执行层

ktrgtc2(一致读)

kcbgtcr (索引检索)

Kcbgtcr(回表)

kdsttgr(全表扫描)

kcbgtcr

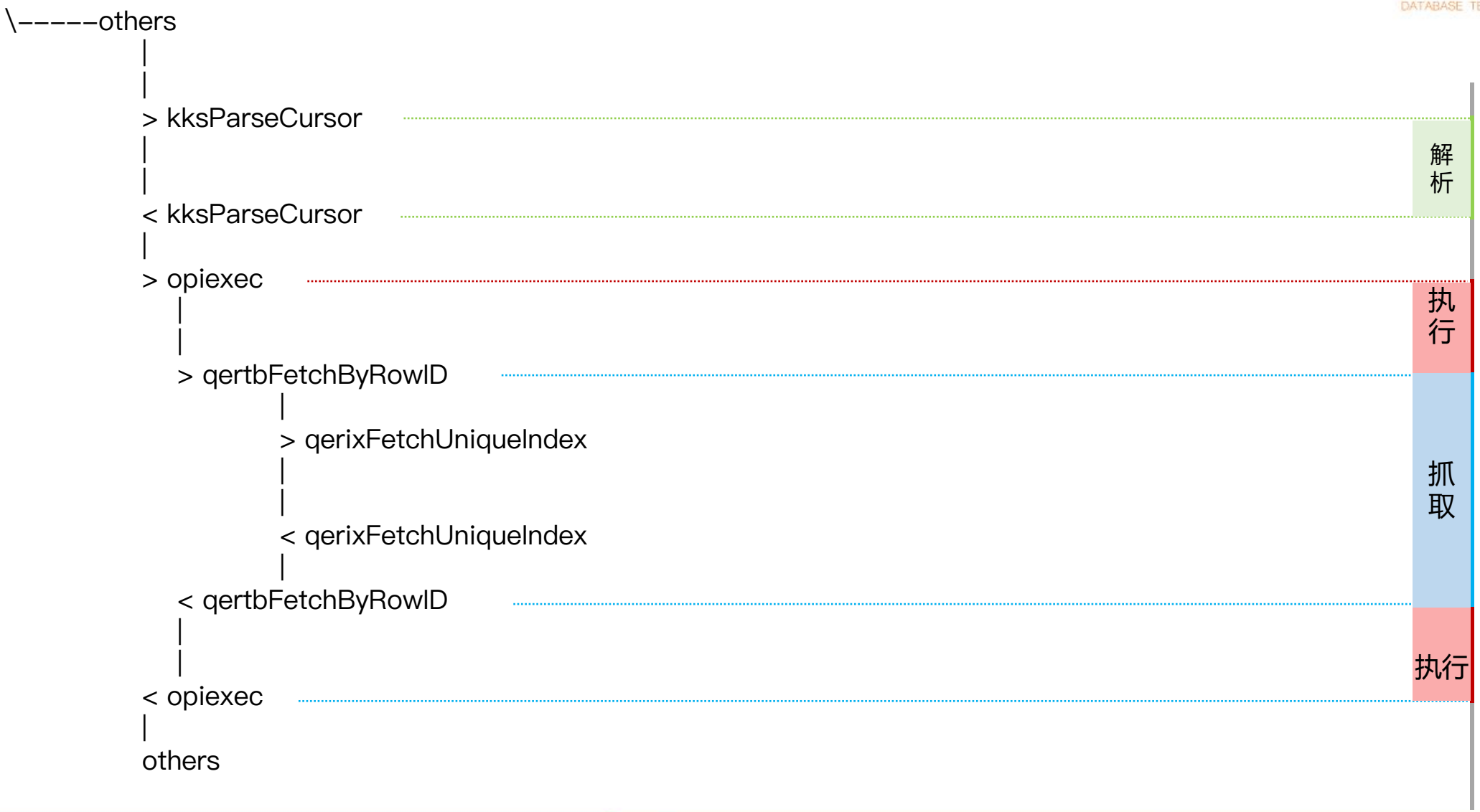
抓取或
逻辑读层

Id	Operation
0	SELECT STATEMENT
1	TABLE ACCESS BY INDEX ROWID
* 2	INDEX UNIQUE SCAN

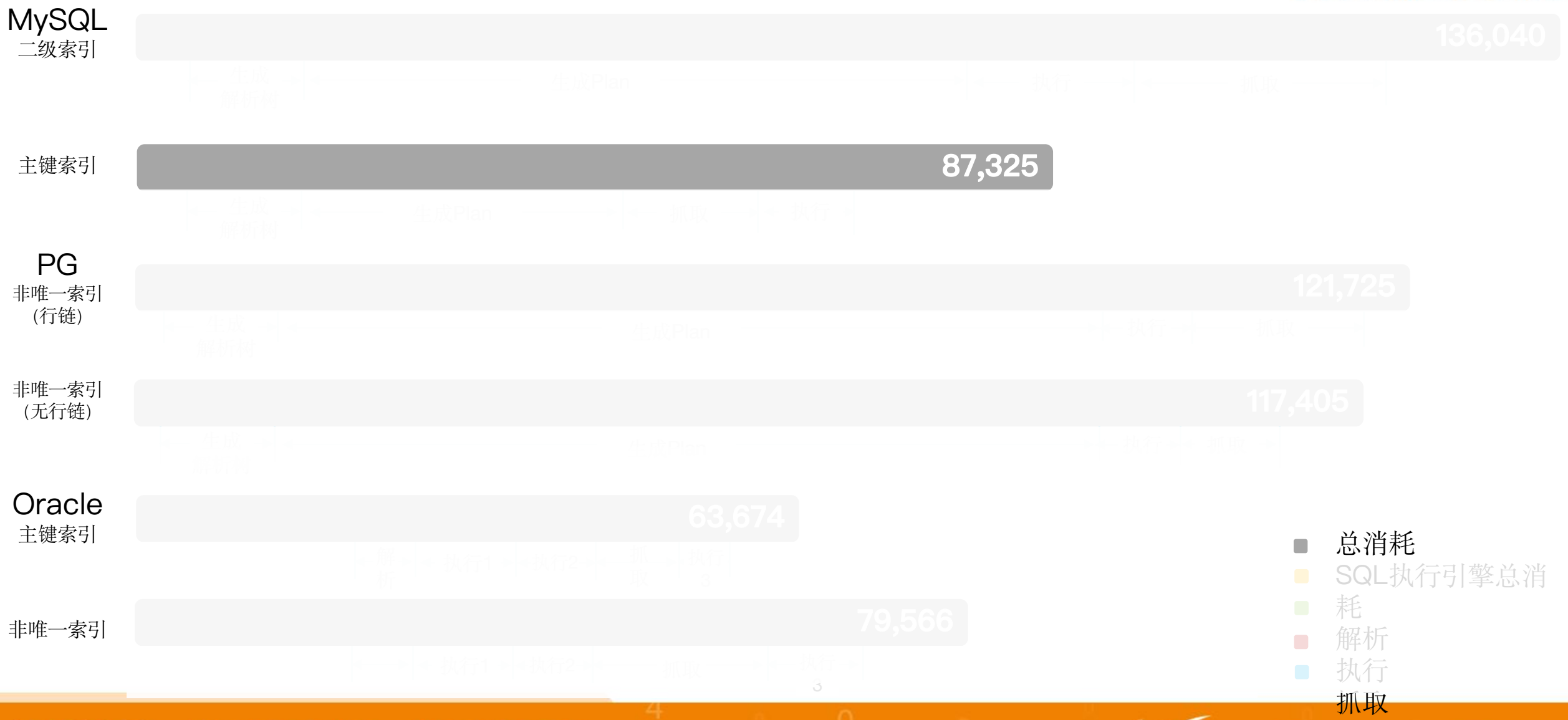
目标SQL:

Select * from vage where id1=1

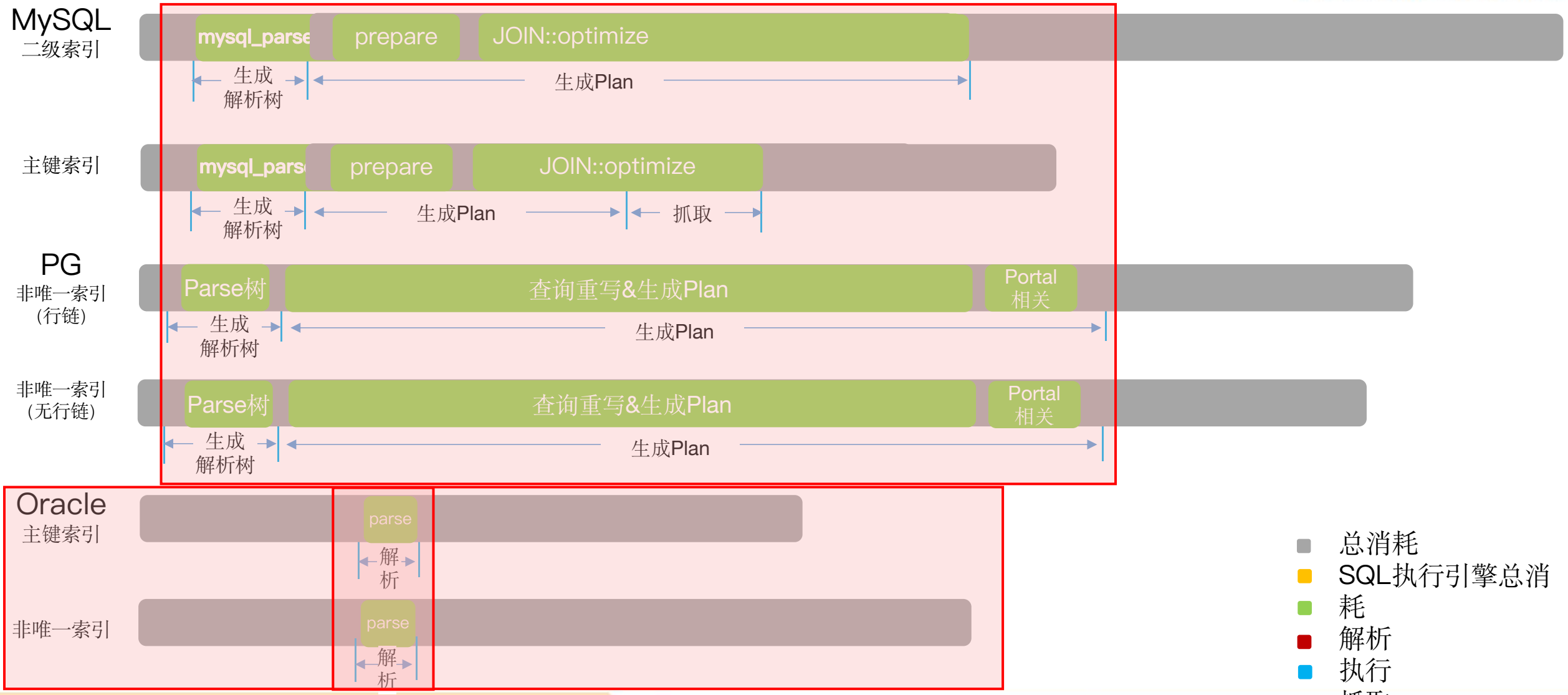
现代处理器之上的数据库



现代处理器之上的数据库



现代处理器之上的数据库



现代处理器之上的数据库

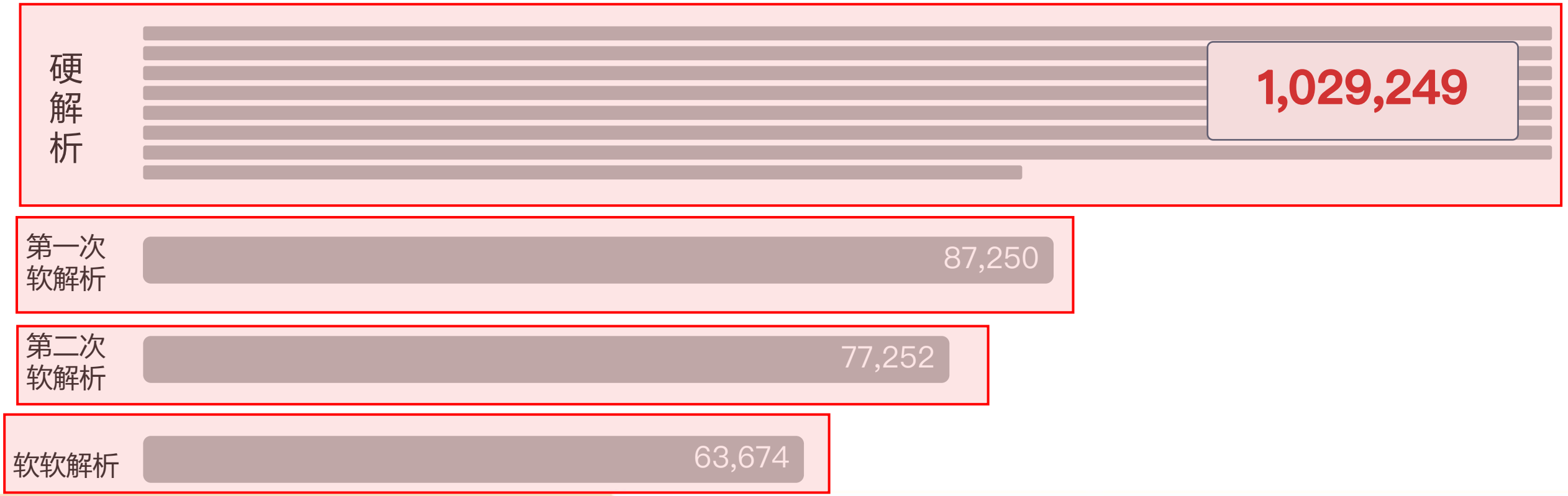
现代处理器之上的数据库

Select … from … where … …

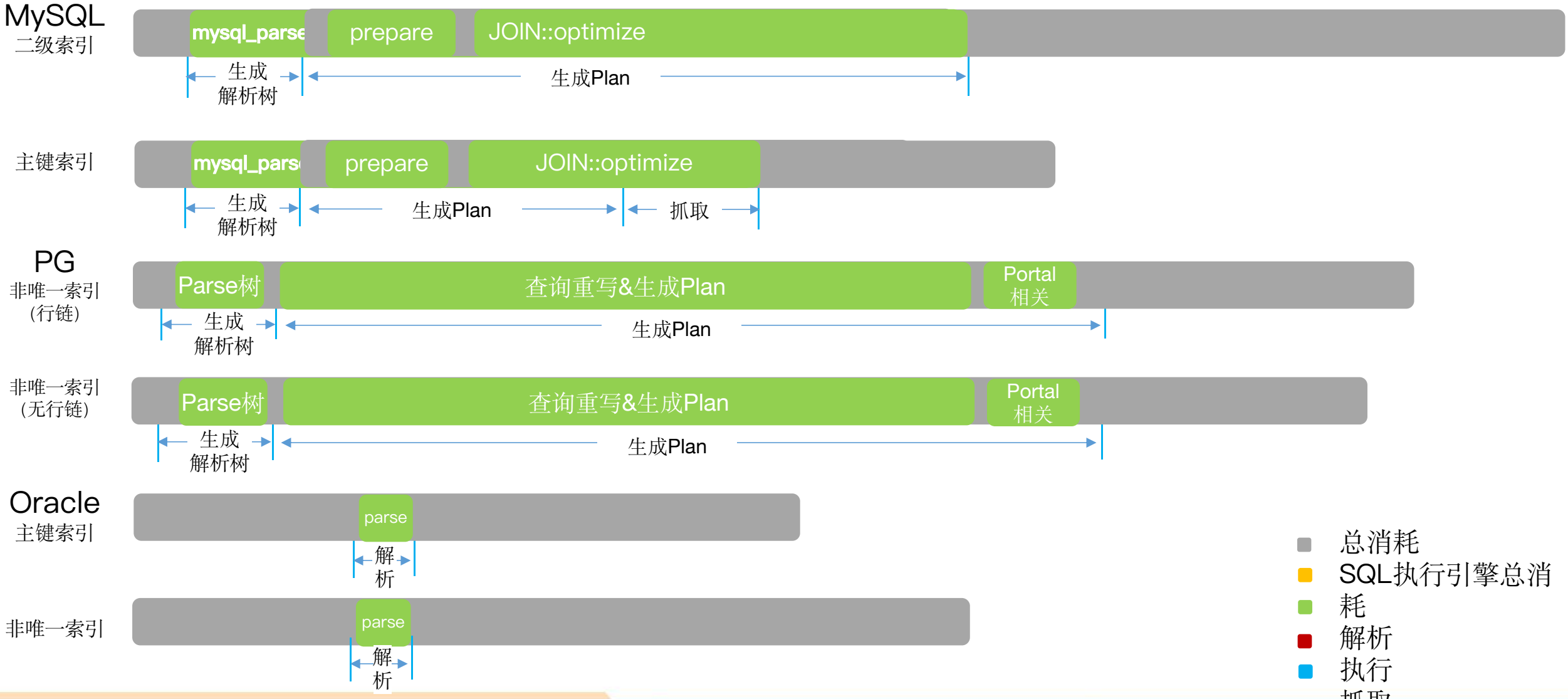
硬解析
生成Plan

共享池

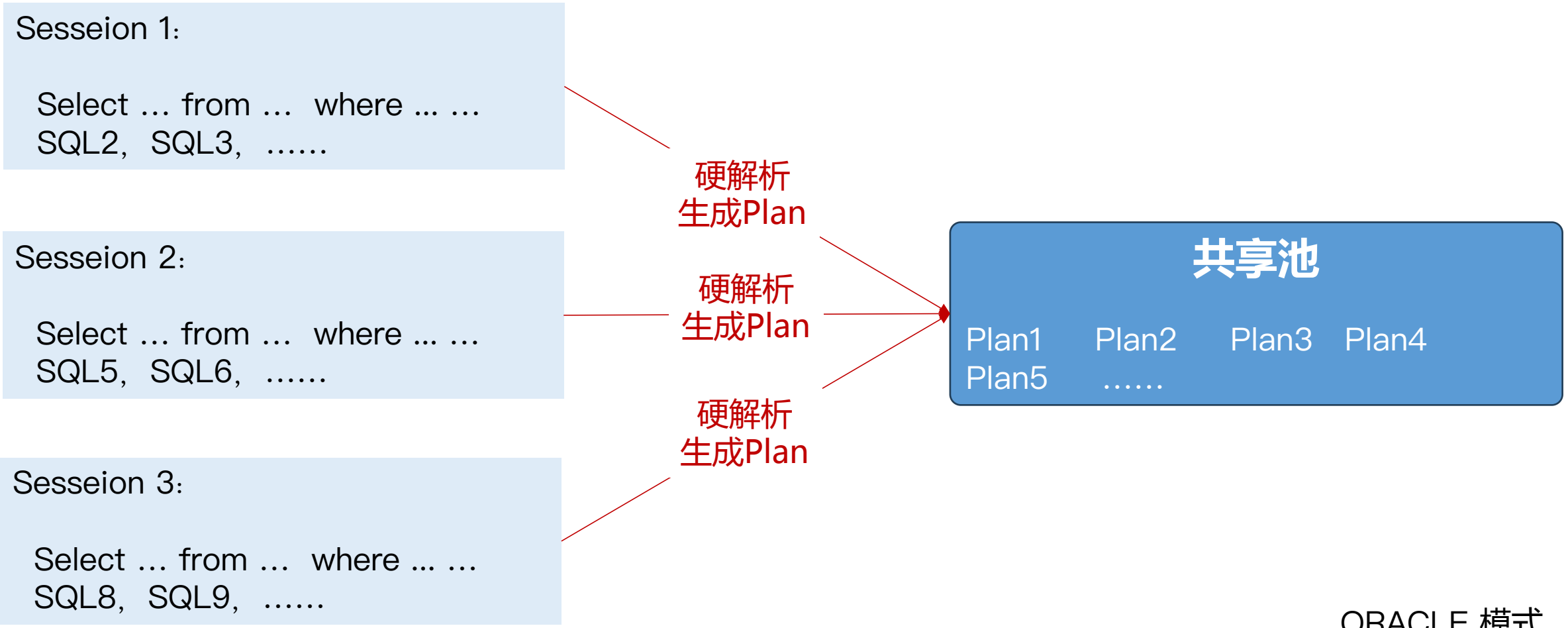
Plan1 Plan2 Plan3 Plan4
Plan5 ……



现代处理器之上的数据库

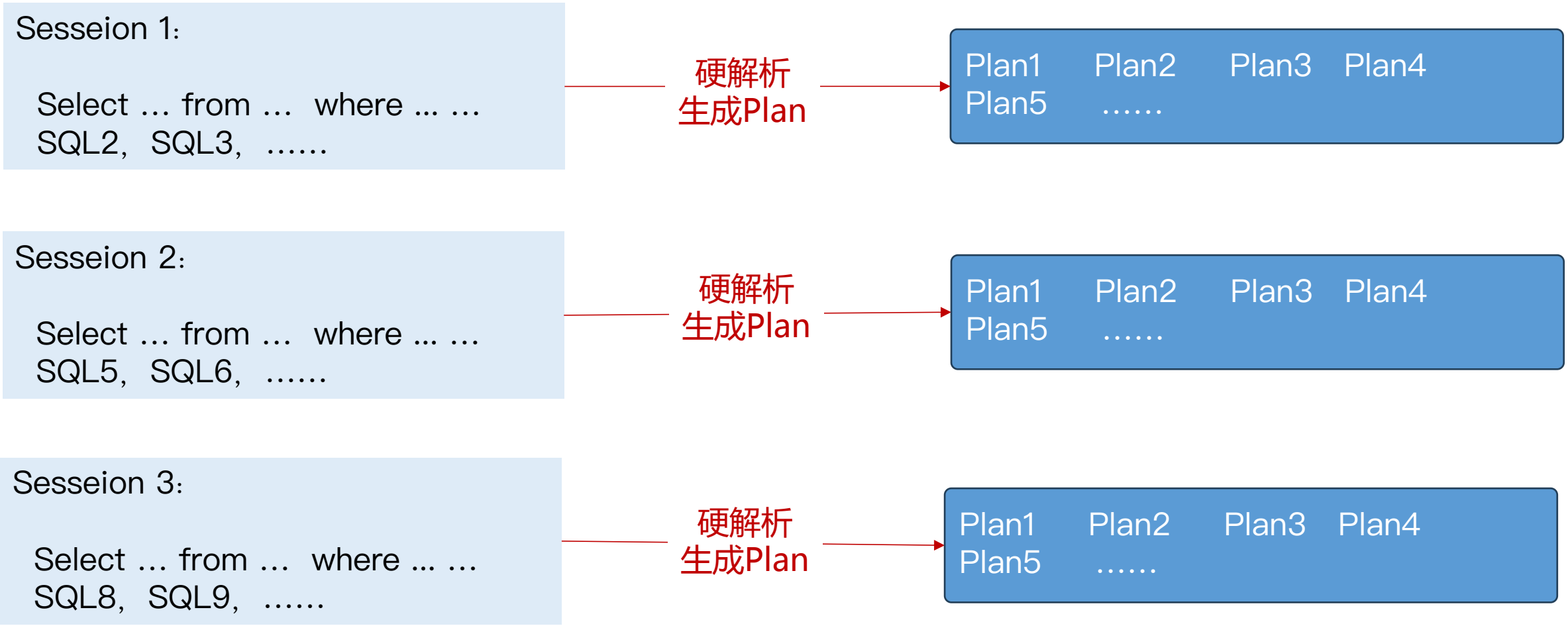


现代处理器之上的数据库



ORACLE 模式

现代处理器之上的数据库



PostgreSQL 模式

现代处理器之上的数据库


```
Driver driver = new org.postgresql.Driver();
Properties prop = new Properties();
prop.setProperty("user", user);
prop.setProperty("password", password);
```

连接数据库

```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        id1 = rs.getInt(1);
        ......
}
```

第一次
执行SQL

```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

第2到6次
执行SQL

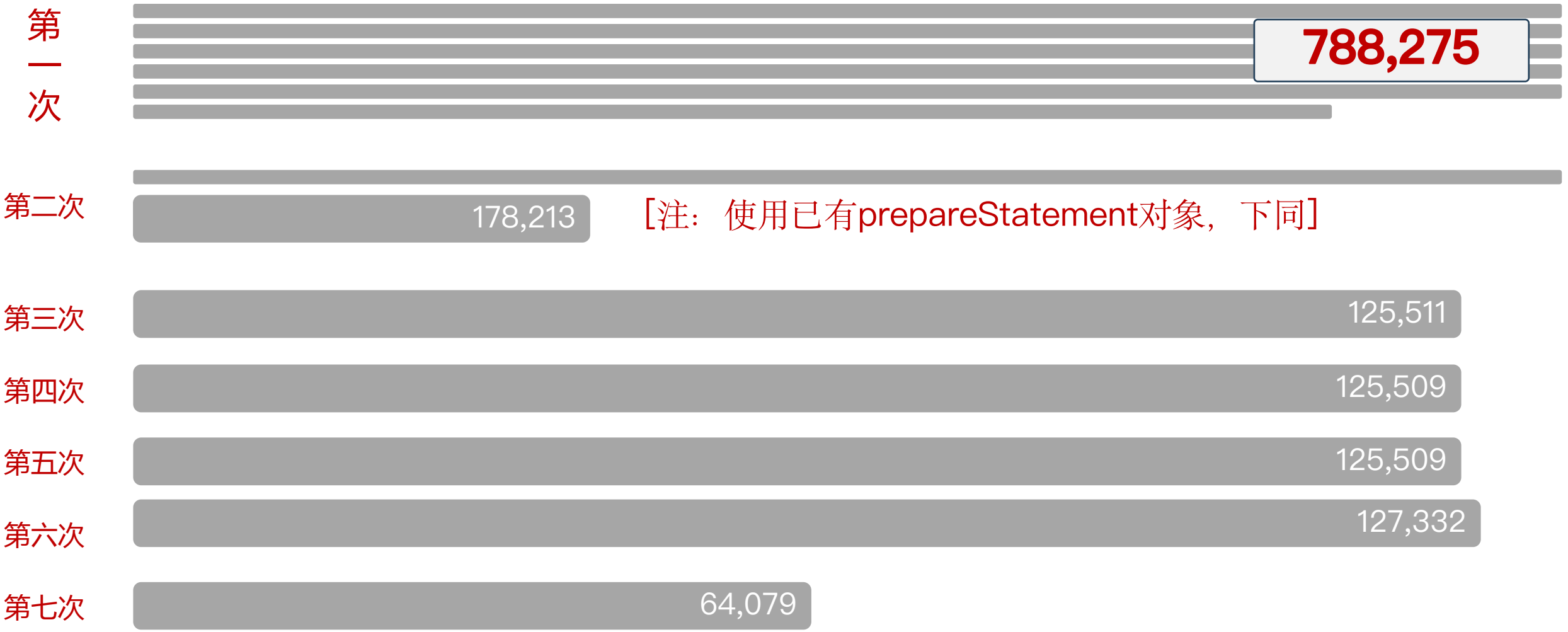
```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

第7次以后
执行SQL

现代处理器之上的数据库

创建prepareStatement对象 prepareStatement SQL 绑定 执行 抓取	第一次
绑定 执行 抓取 绑定 执行 抓取 绑定 执行 抓取 绑定 执行 抓取 绑定 执行 抓取	2到6次
绑定 执行 抓取	第7次

现代处理器之上的数据库



现代处理器之上的数据库

现代处理器之上的数据库


```
st1 = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st1.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        id1 = rs.getInt(1);
        ......
}
```

```
st2 = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st2.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

```
st3 = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st3.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

不需保持PreparedStatement对象

现代处理器之上的数据库


```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        id1 = rs.getInt(1);
        ......
}
```

```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

```
st = conn.prepareStatement("select id1, id2, c1, c2 from vage2 where id1=?");
st.setInt(1, 12345);
rs = st.executeQuery();
while (rs.next()) {
        ......
```

一次解析

多次执行

无需解析
只有执行、抓取

无需解析
只有执行、抓取

现代处理器之上的数据库

Load Profile

	Per Second	Per Transaction	Per Exec	Per Call
DB Time(s):	89.1	0.3	0.00	0.00
DB CPU(s):	10.2	0.0	0.00	0.00
Redo size:	4,402,349.2	12,689.5		
Logical reads:	636,317.4	1,834.1		
Block changes:	8,018.4	23.1		
Physical reads:	8,487.8	24.5		
Physical writes:	1,421.3	4.1		
User calls:	53,075.8	153.0		
Parses:	20,109.9	58.0		
Hard parses:	5.7	0.0		
W/A MB processed:	20.2	0.1		
Logons:	3.0	0.0		
Executes:	20,113.7	58.0		
Rollbacks:	40.6	0.1		
Transactions:	346.9			

Instance Efficiency Percentages (Target 100%)

Buffer Nowait %:	99.43	Redo NoWait %:	99.99
Buffer Hit %:	99.02	In-memory Sort %:	100.00
Library Hit %:	99.95	Soft Parse %:	99.97
Execute to Parse %:	0.02	Latch Hit %:	91.05
Parse CPU to Parse Elapsd %:	1.43	% Non-Parse CPU:	99.84

现代处理器之上的数据库

Shared Pool for PostgreSQL开源团队

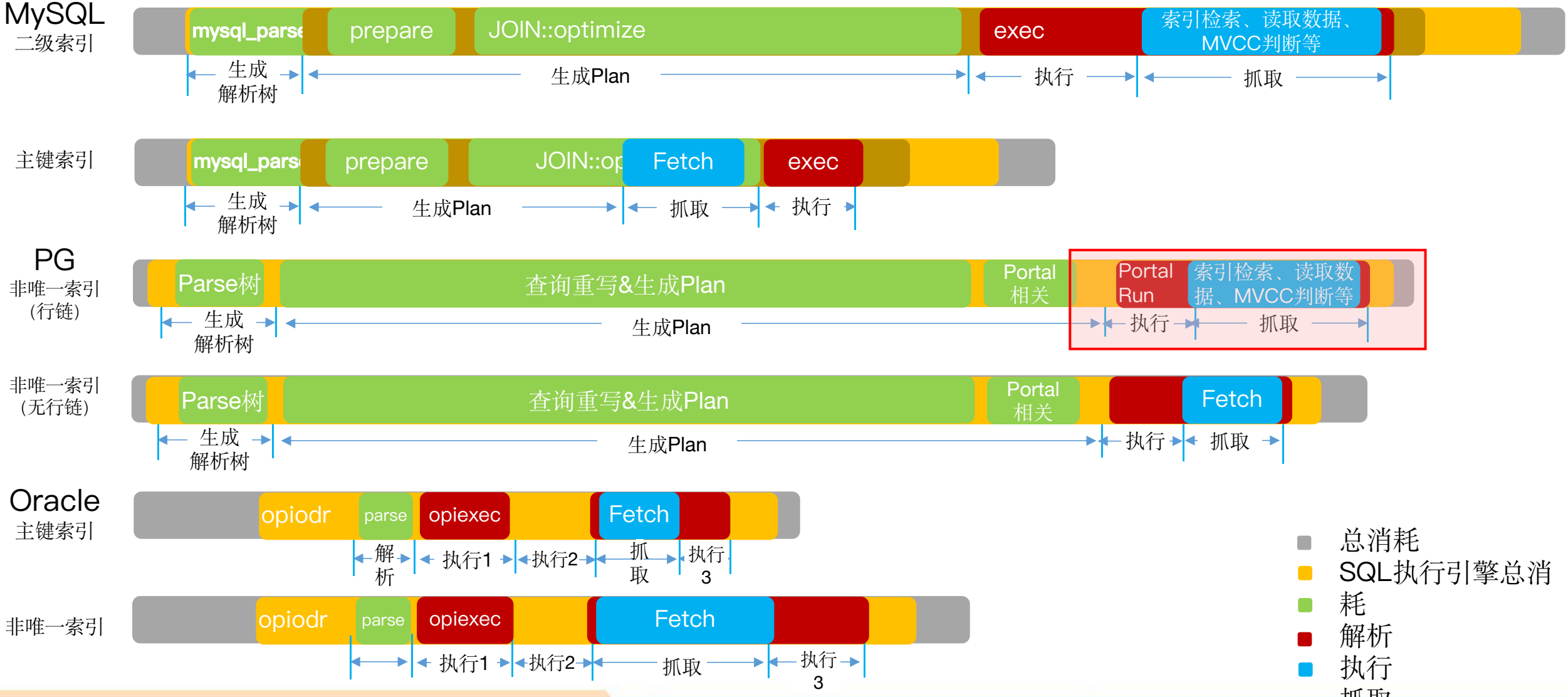
- **北京大学软微学院《开源软件开发基础及实践 – PostgreSQL内核》课程**

我们是国内最强数据库团队之一，成员不乏各省市状元级别的北大优秀学子，还有在数据库、操作系统、计算机体系结构等基础领域摸爬滚打十数年的大牛，更有十几年前就在阿里担任P8的超牛带队。

这样的团队，怀着开源的理想，会打造什么的东东？

IT知识刺客

现代处理器之上的数据库



现代处理器之上的数据库

➢ CPU内的PMC：performance monitor counter

- 性能监控计数器：performance monitor counter
- CPU内置
- Intel/AMD 有千个量级PMC，国产CPU也有百个量级的PMC。

➢ PMC的作用：官方说法，用于对程序进行profiling

- Profiling：来于“侧写”，包含剖析、画像之意
- 几百个计数器，足以完成对程序的“画像”，也足够回答：程序的“好”、“坏”
- 数据库，是**特征明显**的程序

现代处理器之上的数据库

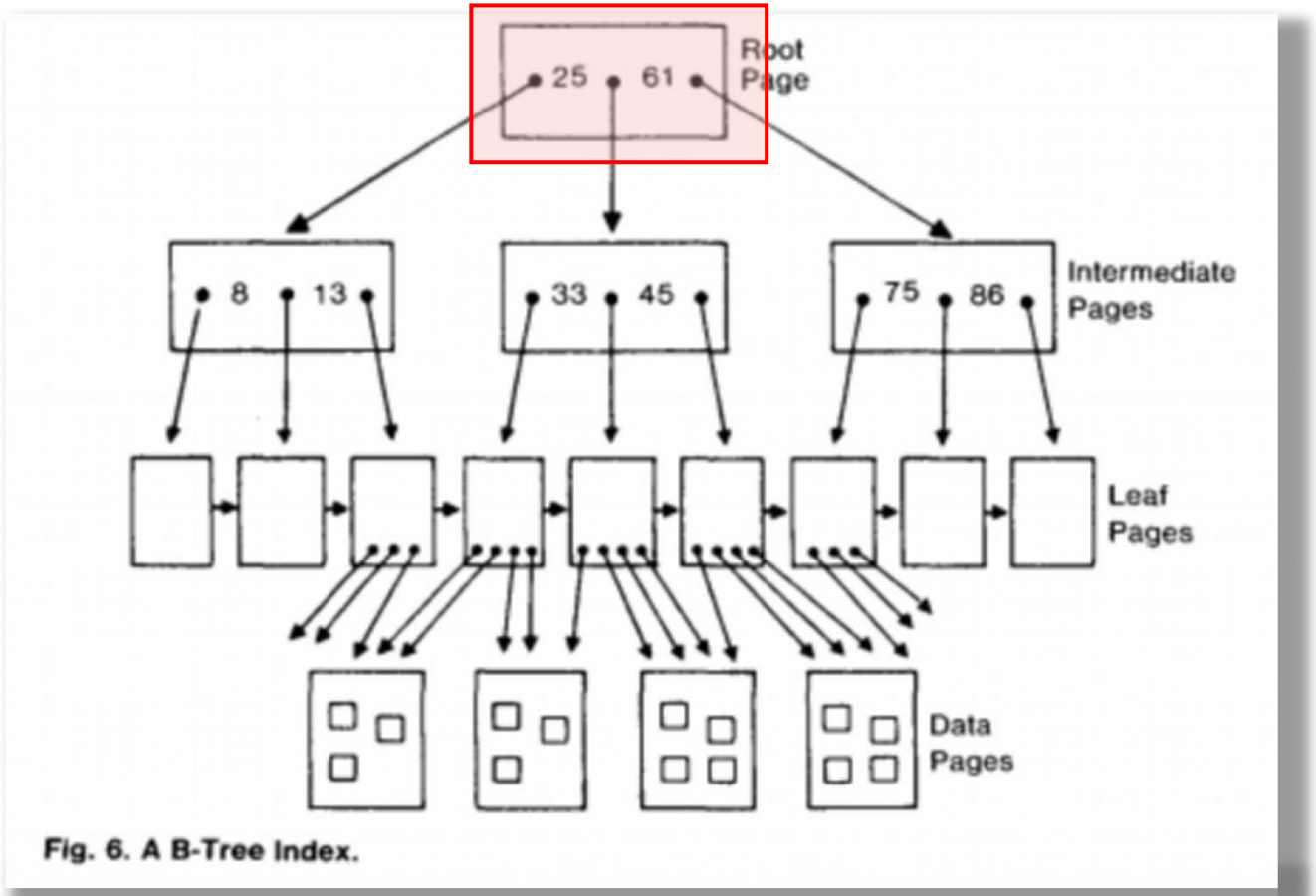
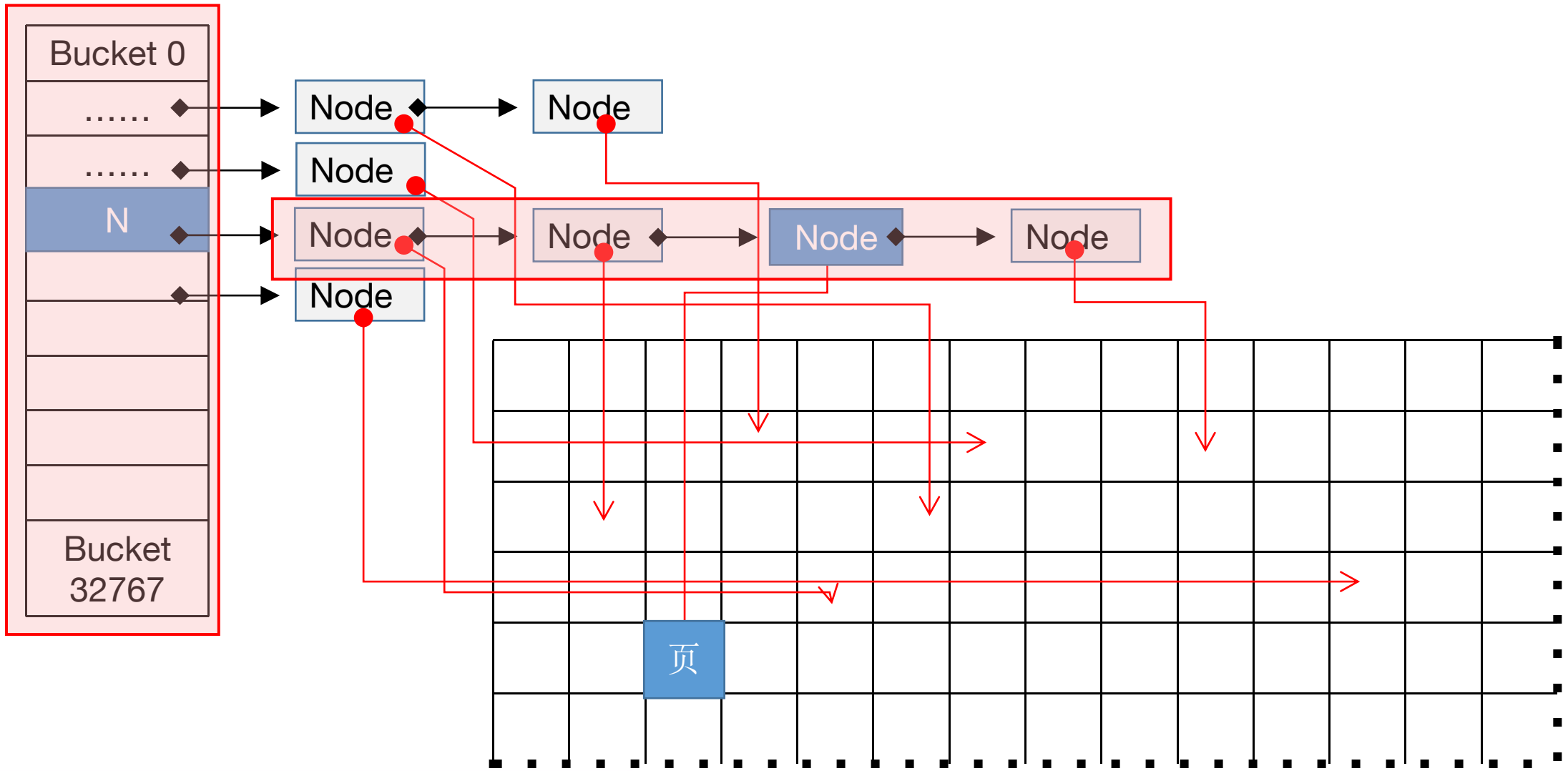
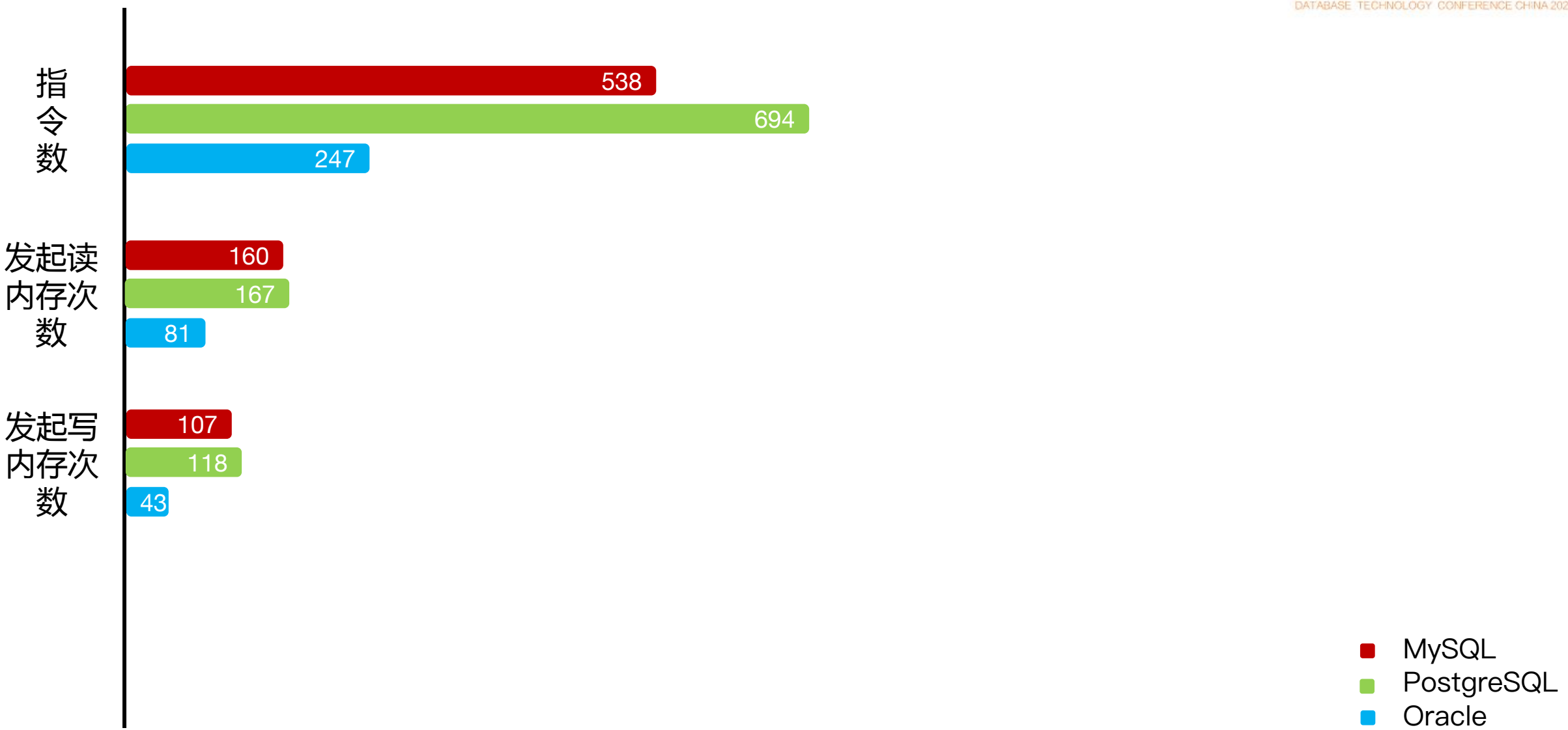


Fig. 6. A B-Tree Index.

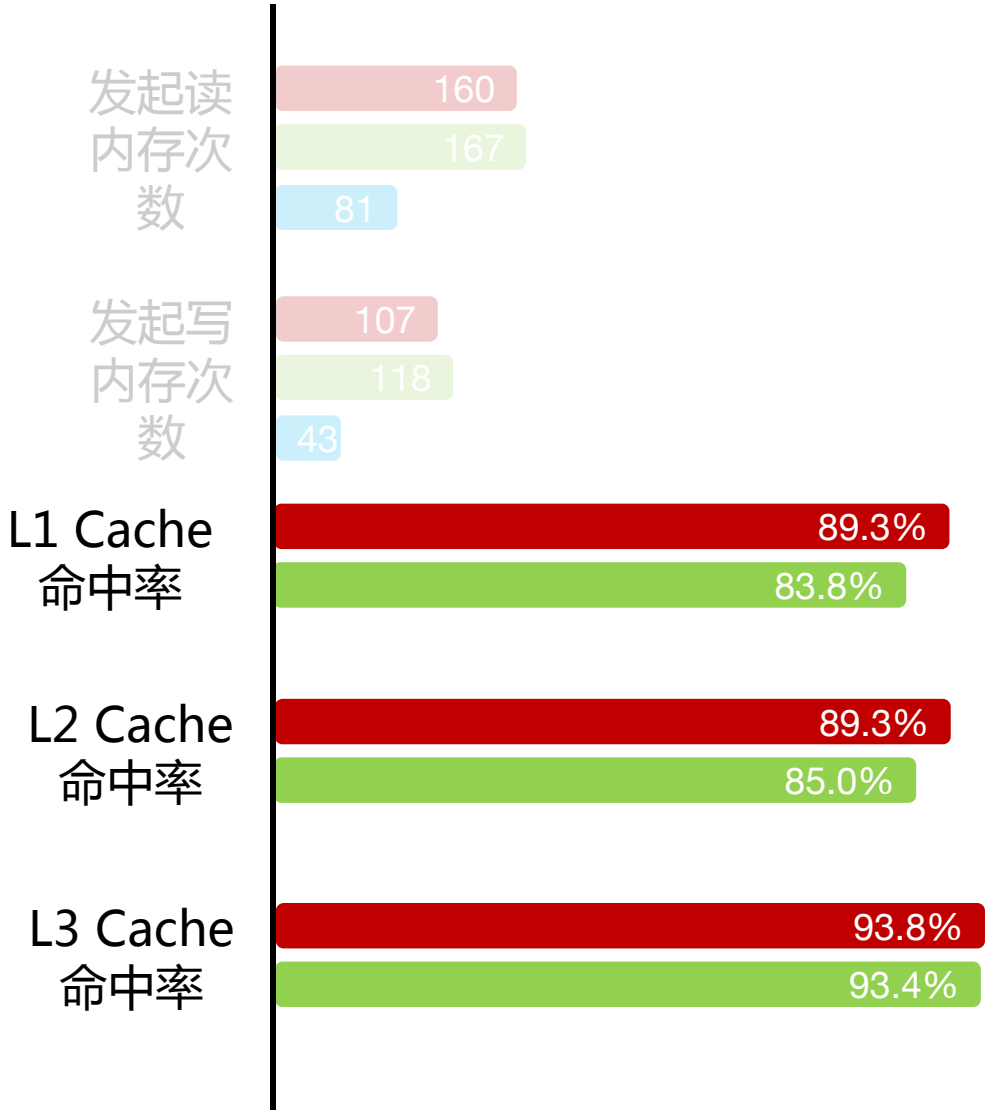
现代处理器之上的数据库



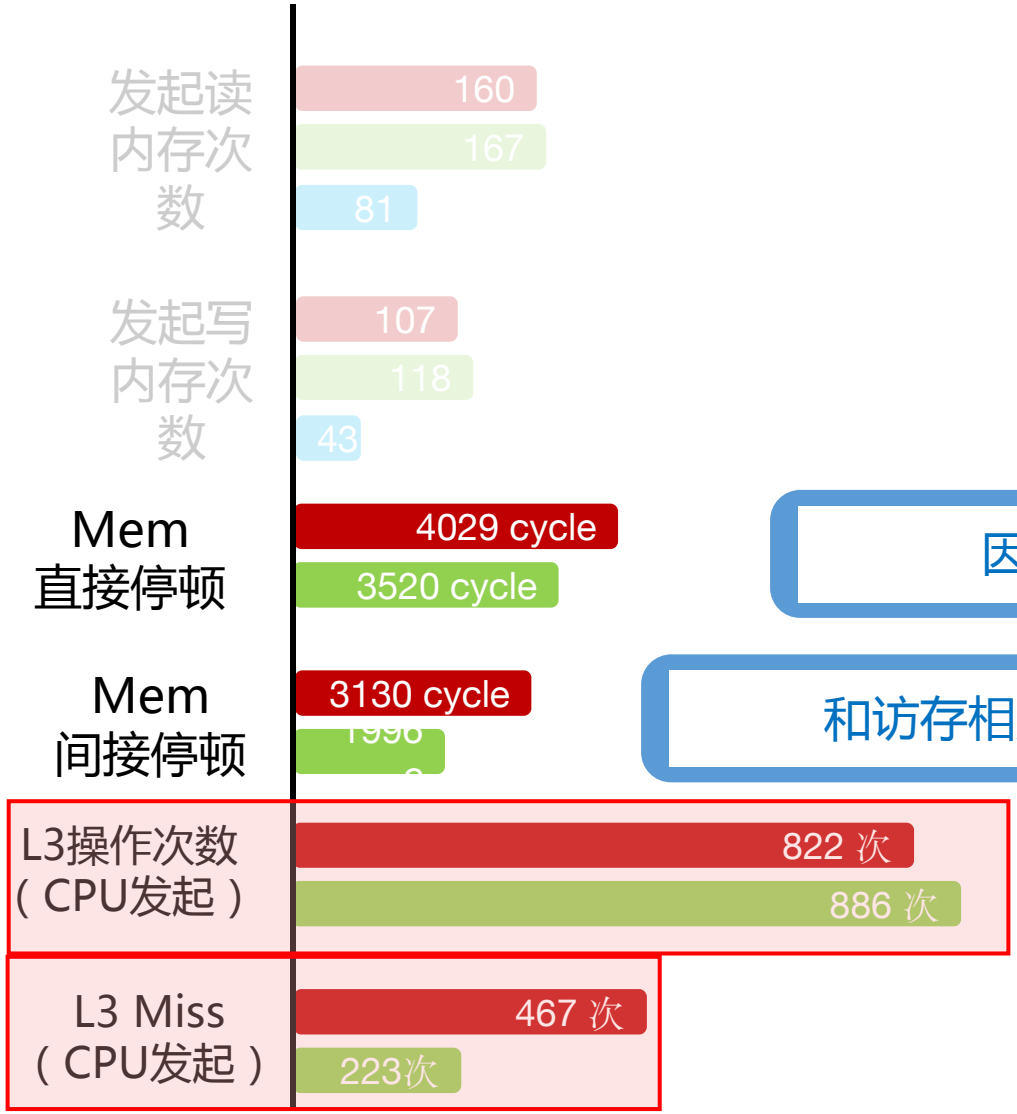
现代处理器之上的数据库



现代处理器之上的数据库



现代处理器之上的数据库

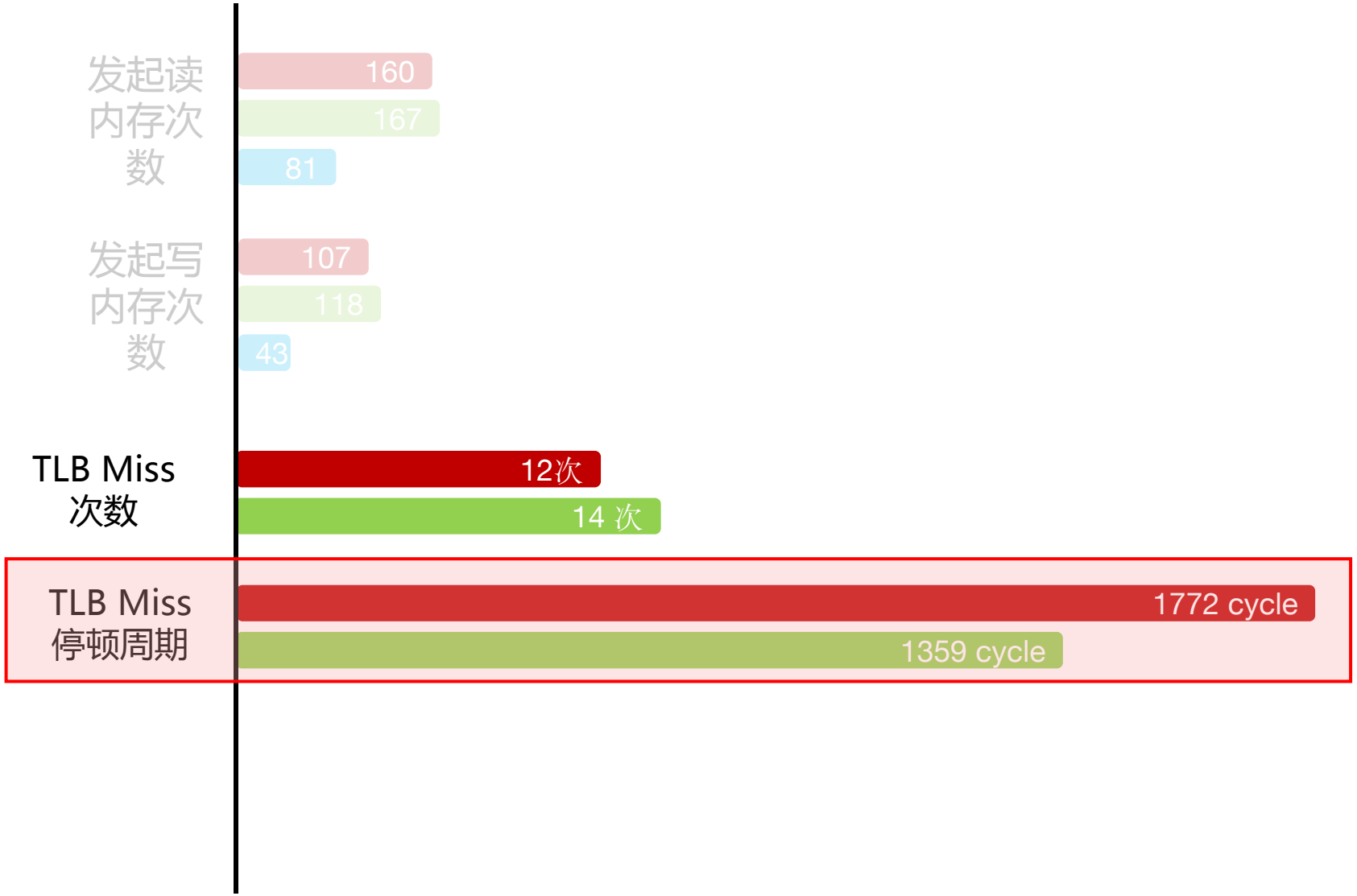


因访存指令延迟直接造成的停顿

和访存相关的停顿，如TLB Miss等

- MySQL
- PostgreSQL
- Oracle

现代处理器之上的数据库



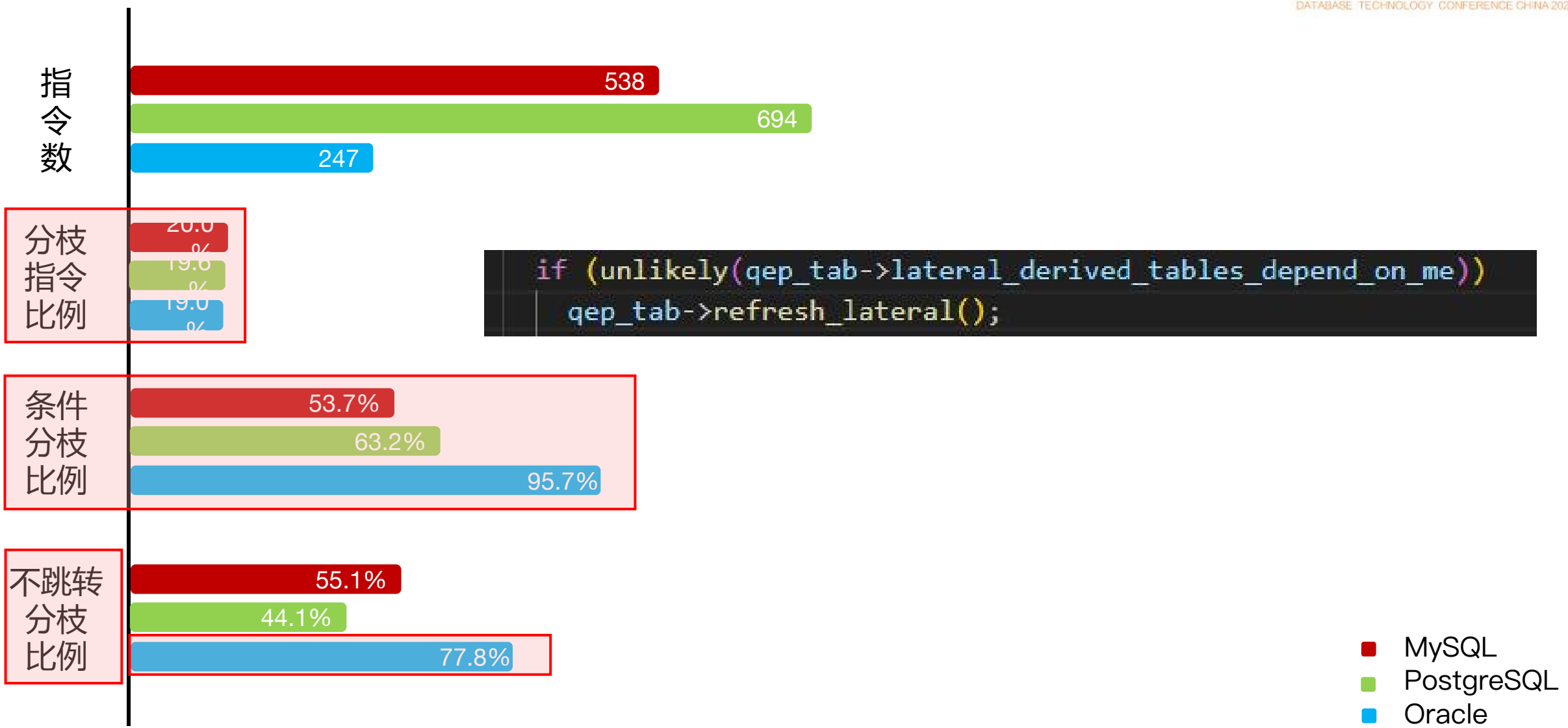
现代处理器之上的数据库

UOP ：微码

MySQL
PostgreSQL
Oracle

The End

现代处理器之上的数据库




```
if (unlikely(qep_tab->lateral_derived_tables_depend_on_me))
  qep_tab->refresh_lateral();
```

现代处理器之上的数据库

	PostgreSQL		国产数据库A		国产数据库B		Oracle	Oracle RAC	
指令数量	176,339	1.80	1,057,173	10.79	2,221,808	22.68	97,975	149,519	1.53
分枝指令	35,202	1.89	180,826	9.73	426,557	22.95	18,586	28,234	1.52
不跳转分枝	12,411	1.21	50,528	4.93	190,279	18.55	10,258	13,265	1.29
内存读	45,471	1.56	366,060	12.52	577,411	19.75	29,237	50,697	1.73
内存写	33,569	1.96	220,328	12.84	408,369	23.80	17,159	31,992	1.86
主存流量	96KB	1.68	186KB	3.26	407KB	7.14	57KB		
流水线停顿次数	555,109	4.57	2,014,078	16.60	1,147,604	9.46	121,336		

THANKS