DTCC

数，造，未，来

第十二届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2021

2021 年 10 月 18 日 - 20 日 | 北京国际会议中心

Spark
Hadoop
OLAP
HANA
MPP
Oracle

IT168.com ChinaUnix.net ITPUB

防微杜渐，浅谈规范化设计开发对数据库性能的影响

刘晨
中信建投证券股份有限公司

个人介绍

- 刘晨(网络ID：bisal)
- 管理学硕士
- 目前主要从事数据库应用研发能力提升和技术管理相关的工作
- Oracle ACE(Former)，腾讯TVP，拥有Oracle OCM/OCP、EXIN DevOps Master、SCJP等国际认证
- 国内首批Oracle YEP成员、OCMU联盟核心成员
- 《DevOps最佳实践》中文版合作译者
- ITPub/CSDN专家博主
- 微信公众号：bisal的个人杂货铺

提纲

- 案例介绍
- 现状和痛点
- 一些想法和实践

故障现象

一套OLTP系统，某天00:35左右，用户开始报障，出现业务办理响应缓慢的现象。

- 应用背景
  - 某核心OLTP系统
  - 7*24保障级别
- 数据库背景
  - Oracle 11.2.0.4 RAC
  - RedHat 64bit，20C256G

故障现象

```
zzz ***Sun [redacted]  00:30:51 CST [redacted]
procs      -----------memory---------- ---swap--   -----io----   --system--   -----cpu-----
 r  b   swpd     free   buff     cache   si   so     bi     bo     in     cs us sy id wa st
 8  0      0 83025696 2053580 35085296    0    0    199     71      0      0  3  1 96  0  0
 9  1      0 83004504 2053580 35085300    0    0 118950 261331 20653 26941 32  2 62  4  0
 9  0      0 83033232 2053596 35085684    0    0   6963  37617 22481 28820 36  4 58  3  0
zzz ***Sun [redacted]  00:31:25 CST [redacted]
procs      -----------memory---------- ---swap--   -----io----   --system--   -----cpu-----
 r  b   swpd     free   buff     cache   si   so     bi     bo     in     cs us sy id wa st
16  1      0 83024232 2053580 35086452    0    0    199     71      0      0  3  1 96  0  0
13  0      0 83031744 2053580 35086456    0    0  17885  34426 34482 35046 70  2 26  3  0
11  1      0 83029280 2053580 35086496    0    0   4947  36653 25445 24928 59  2 37  2  0
zzz ***Sun [redacted]  00:31:59 CST [redacted]
procs      -----------memory---------- ---swap--   -----io----   --system--   -----cpu-----
 r  b   swpd     free   buff     cache   si   so     bi     bo     in     cs us sy id wa st
26  1      0 83010248 2053580 35087148    0    0    200     71      0      0  3  1 96  0  0
28  0      0 82994256 2053580 35087152    0    0   3691  36714 33346 24931 98  2  0  0  0
27  0      0 82980752 2053580 35087204    0    0   4147  45835 36657 27377 98  1  0  0  0
zzz ***Sun [redacted]  00:32:34 CST [redacted]
procs      -----------memory---------- ---swap--   -----io----   --system--   -----cpu-----
 r  b   swpd     free   buff     cache   si   so     bi     bo     in     cs us sy id wa st
29  0      0 82672568 2053596 35088004    0    0    200     71      0      0  3  1 96  0  0
31  0      0 82674520 2053596 35088012    0    0   2444  20051 29031 19478 99  1  0  0  0
28  0      0 82680016 2053580 35088072    0    0   5939  19516 31583 22082 99  1  0  0  0
```

故障现象

Report Summary

Load Profile

	Per Second	Per Transaction	Per Exec	Per Call
DB Time(s):	1.9	0.1	0.02	0.01
DB CPU(s):	0.7	0.0	0.01	0.00
Redo size (bytes):	535,601.9	28,970.3		
Logical read (blocks):	576,462.2	31,180.4		
Block changes:	1,661.7	89.9		
Physical read (blocks):	187.2	10.1		
Physical write (blocks):	178.2	9.6		
Read IO requests:	187.2	10.1		
Write IO requests:	100.2	5.4		
Read IO (MB):	1.5	0.1		
Write IO (MB):	1.4	0.1		
Global Cache blocks received:	6.1	0.3		
Global Cache blocks served:	14.9	0.8		
User calls:	191.8	10.4		
Parses (SQL):	94.7	5.1		
Hard parses (SQL):	0.1	0.0		
SQL Work Area (MB):	1.1	0.1		
Logons:	0.5	0.0		
Executes (SQL):	97.0	5.2		
Rollbacks:	0.0	0.0		
Transactions:	18.5			

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
resmgr:cpu quantum	16,394	3538.2	216	51.4	Scheduler
DB CPU		2432.9		35.3	
db file sequential read	657,757	239.1	0	3.5	User I/O
log file sync	59,631	75.7	1	1.1	Commit
reliable message	133,749	57.7	0	.8	Other
gc cr disk read	443,199	55.1	0	.8	Cluster
gc current grant 2-way	146,522	25.6	0	.4	Cluster
control file sequential read	61,491	16.1	0	.2	System I/O
latch free	3,124	12.9	4	.2	Other
direct path write	24,998	11.7	0	.2	User I/O

故障现象

update A set a=:1, b=:2, c=:3 ... where id=:10 and update_time=:11;

Plan Statistics

- % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100

Stat Name	Statement Total	Per Execution	% Snap Total
Elapsed Time (ms)	5,919,151	572.89	85.94
CPU Time (ms)	2,001,590	193.73	82.27
Executions	10,332		
Buffer Gets	2,040,783,005	197,520.62	98.25
Disk Reads	20,664	2.00	3.06
Parse Calls	10,332	1.00	3.03
Rows	10,332	1.00	
User I/O Wait Time (ms)	16,458		
Cluster Wait Time (ms)	51		
Application Wait Time (ms)	0		
Concurrency Wait Time (ms)	804		
Invalidations	0		
Version Count	132		
Sharable Mem(KB)	348		

Execution Plan

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	UPDATE STATEMENT				3 (100)	
1	UPDATE					
2	TABLE ACCESS BY INDEX ROWID		1	113	3 (0)	00:00:01
3	INDEX UNIQUE SCAN		1		2 (0)	00:00:01

故障现象

夜维操作，

```
delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
```

参数:1 = 90(天)

参数:2 = 10000(条)

故障现象

期望:

第一次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
commit;
第二次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
commit;
...
第N次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
commit;

实际:

第一次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
第二次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
...
第N次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;
commit;

故障现象

- 数据库服务器的CPU idle降为0
- 受影响的业务对应的是一条UPDATE语句(update A set a=:1, b=:2, c=:3 ... where id=:10 and update_time=:11;)
- 该SQL的执行计划已经是比较高效的(唯一索引扫描)
- 该SQL涉及的表有并发执行的大数据量删除事务(delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;)

Bug 19791273

Bug 19791273 - Poor UPDATE SQL performance due to space search cache for updates on ASSM segment (Doc ID 19791273.8)

Description

This bug is only relevant when using ASSM Space Management (Bitmap Managed Segments)

This problem can occur if the fix for bug 13641076 is present.

That fix enables the use of space search cache in updates as well as inserts but this can degrade the performance of the UPDATE SQL operations.

This fix disables the space search cache for UPDATE operations to avoid the performance overhead.

Workaround

Setting event 10019 can help for specific UPDATE workload.

Bug 13641076

Bug 13641076 - High buffer gets for insert statement - rejection list does not fire (Doc ID 13641076.8)

Description

This bug is only relevant when using ASSM Space Management (Bitmap Managed Segments)

INSERT operations have a high amount of buffer gets when there is a concurrent uncommitted large DELETE for ASSM segments.

Note:

It is expected in ASSM that a first attempt to find space in a segment may need to visit many blocks in the case of an uncommitted concurrent delete, but subsequent executions of the cursor should use a "reject" list implemented in bug 4188324 and so skip reading of blocks already rejected.

Bug 13641076

Bug 13641076 - High buffer gets for insert statement - rejection list does not fire - superseded
(Doc ID 13641076.8)

Prior to this fix the "rejection" list is cursor based, and so if the DML cursor is closed and a new cursor used the reject list has to be rebuilt.
This fix changes the rejection list to be session based rather than cursor based.

This fix supersedes the fix for bug 4188324 and it can be disabled by the same event 10019 .

Rediscovery Notes
If INSERT operations show high amount of buffer gets when there is a concurrent uncommitted large DELETE, and subsequent DMLs on the same object in the same session do not seem to benefit from the rejection cache then this fix may help.

Bug 4188324

Bug 4188324 - Wasted space possible in ASSM segments - superseded

(Doc ID 4188324.8)

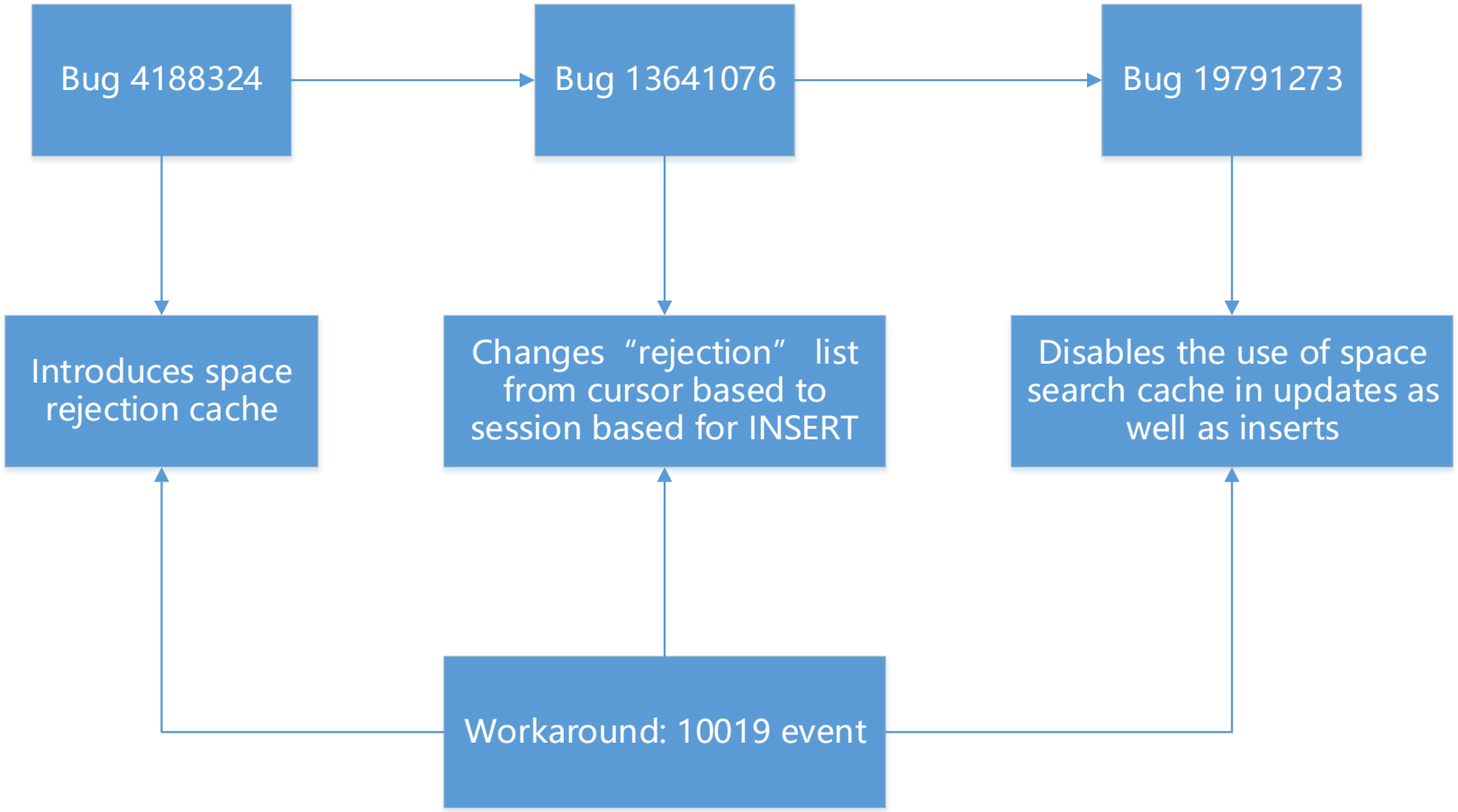
Description

Space may not always be reused when expected in an ASSM segment.

This fix tries to help maintain the "freeness" information about blocks in ASSM segments to help avoid excess wastage of space freed up by large transactions and keeps a memory list of blocks that has been rejected.

Thie list is known as the space rejection cache.

Bug关系复盘


Bug 4188324
Bug 13641076
Bug 19791273
Introduces space rejection cache
Changes “rejection” list from cursor based to session based for INSERT
Disables the use of space search cache in updates as well as inserts
Workaround: 10019 event

Bug 19791273

19791273	DISABLE SPACE SEARCH CACHE FOR UPDATES (Patch)	Linux x86-64 (American English)	11.2.0.4.190416
19791273	DISABLE SPACE SEARCH CACHE FOR UPDATES (Patch)	Linux x86-64 (American English)	11.2.0.4.170718

Patch 19791273: DISABLE SPACE SEARCH CACHE FOR UPDATES

Last Updated Apr 28, 2020 11:01 AM (1+ year ago)

Product Oracle Database - Enterprise Edition (More...)
Release Oracle 11.2.0.4.0
Platform Linux x86-64

Size Not Applicable
Download Access Software
Classification Defective
Patch Tag

This Patch is Obsolete. It cannot be Downloaded.

Reason

Bug 19791273 has been replaced by new bug 30758943

Note

The most recent replacement for this patch is 30758943.

Replacement Options

30758943	INSERT / DELETE FROM STREAMS$_APPLY_PROGRESS LEAVE FULL BLOCKS - SEGMENT GROWTH	Patch

第一次测试

1. 应用：

使用旧的夜维(无批量提交，虽然delete操作一次删除10000条，但是会在删除所有数据(20万)完成的时候，才会做commit，约需要2分钟，即需要让相应数据和回滚表空间的数据块处于事务进行中状态约2分钟)。

2. 数据库：

未设置10019，存在Bug 13641076的bugfix，未修复Bug 19791273。

3. 执行过程：

当夜维执行到第10次左右的10000条delete操作，应用的响应时间开始变长，数据库CPU idle最低降到了60%左右了，正常时间段，CPU idle一般为80%-90%左右的。

第一次测试

4. 数据：

这次夜维执行，22:10-22:12，总计2分钟左右，检索对应业务的update语句逻辑读，从下图可以看出，相比其他小时段，增长了将近1000倍，从SQL AWR看，这条update语句的逻辑读，大约是22万，

02点	358531
03点	231130
04点	211917
05点	239446
06点	509865
07点	855990
08点	681718
09点	585242
10点	773113
11点	878033
12点	823730
13点	781357
14点	713506
15点	797605
16点	735009
17点	761377
18点	711925
19点	576254
20点	516909
21点	476766
22点	481602117

Plan Statistics

- % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100

Stat Name	Statement Total	Per Execution	% Snap Total
Elapsed Time (ms)	468,897	219.42	64.72
CPU Time (ms)	460,483	215.48	81.02
Executions	2,137		
Buffer Gets	480,796,155	224,986.50	98.35
Disk Reads	4,274	2.00	2.88
Parse Calls	2,137	1.00	3.05
Rows	2,136	1.00	
User I/O Wait Time (ms)	4,771		
Cluster Wait Time (ms)	4		
Application Wait Time (ms)	0		
Concurrency Wait Time (ms)	136		
Invalidations	0		
Version Count	132		
Sharable Mem(KB)	949		

第二次测试

1. 应用：

使用旧的夜维，同时，业务切换至应用备份集群，重启应用备集群的连接池和应用，保证10019事件生效。

2. 数据库：

设置10019事件。

3. 执行过程：

夜维执行过程中，业务的响应时间，基本保持不变，数据库CPU idle一直在80%-90%左右的，可以说现在夜维的执行对正常业务基本无影响。

第二次测试

4. 数据：

这次夜维执行，还是2分钟左右，对应SQL AWR，显示update语句逻辑读，这次变成了44，

Plan Statistics

- % Total DB Time is the Elapsed Time of the SQL statement divided into the Total Database Time multiplied by 100

Stat Name	Statement Total	Per Execution	% Snap Total
Elapsed Time (ms)	6,605	5.61	2.67
CPU Time (ms)	1,708	1.45	1.58
Executions	1,178		
Buffer Gets	52,754	44.78	0.67
Disk Reads	3,517	2.99	2.41
Parse Calls	1,178	1.00	2.89
Rows	1,178	1.00	
User I/O Wait Time (ms)	5,043		
Cluster Wait Time (ms)	1		
Application Wait Time (ms)	0		
Concurrency Wait Time (ms)	0		
Invalidations	0		
Version Count	132		
Sharable Mem(KB)	949		

第三次测试

使用新夜维，即带批量提交的删除逻辑，从应用和数据库角度看，时间和CPU idle都和第二次测试相近，基本不存在delete对update的影响。

第一次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;

commit;

第二次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;

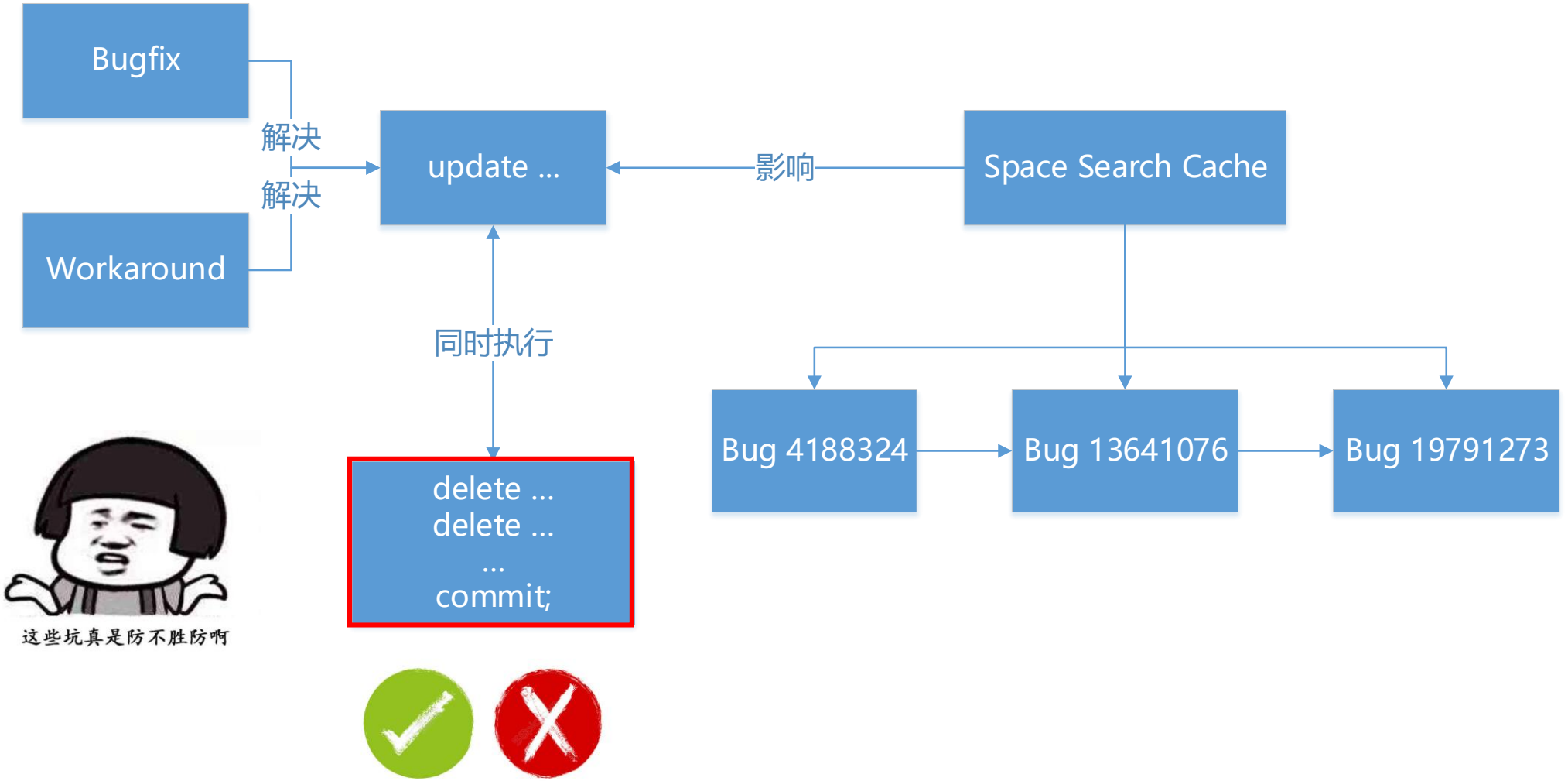
commit;

...

第N次执行：delete from A where update_time <= trunc(sysdate) - :1 and rownum <= :2;

commit;

回顾


Bugfix
Workaround
解决
解决
update ...
影响
Space Search Cache
同时执行
delete ...
delete ...
...
commit;
Bug 4188324
Bug 13641076
Bug 19791273
这些坑真是防不胜防啊

- 案例介绍
- 现状和痛点
- 一些想法和实践

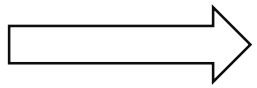
现状和痛点

现状

- 自研 + 采购
- 员工 + 外包
- 传统 + 创新

痛点

- 技术栈难统一
- 技术能力不平衡
- 技术规范难落地



- 技术难沉淀
- 规范难落地
- 标准难实施

数据库设计开发中的一些常见心态

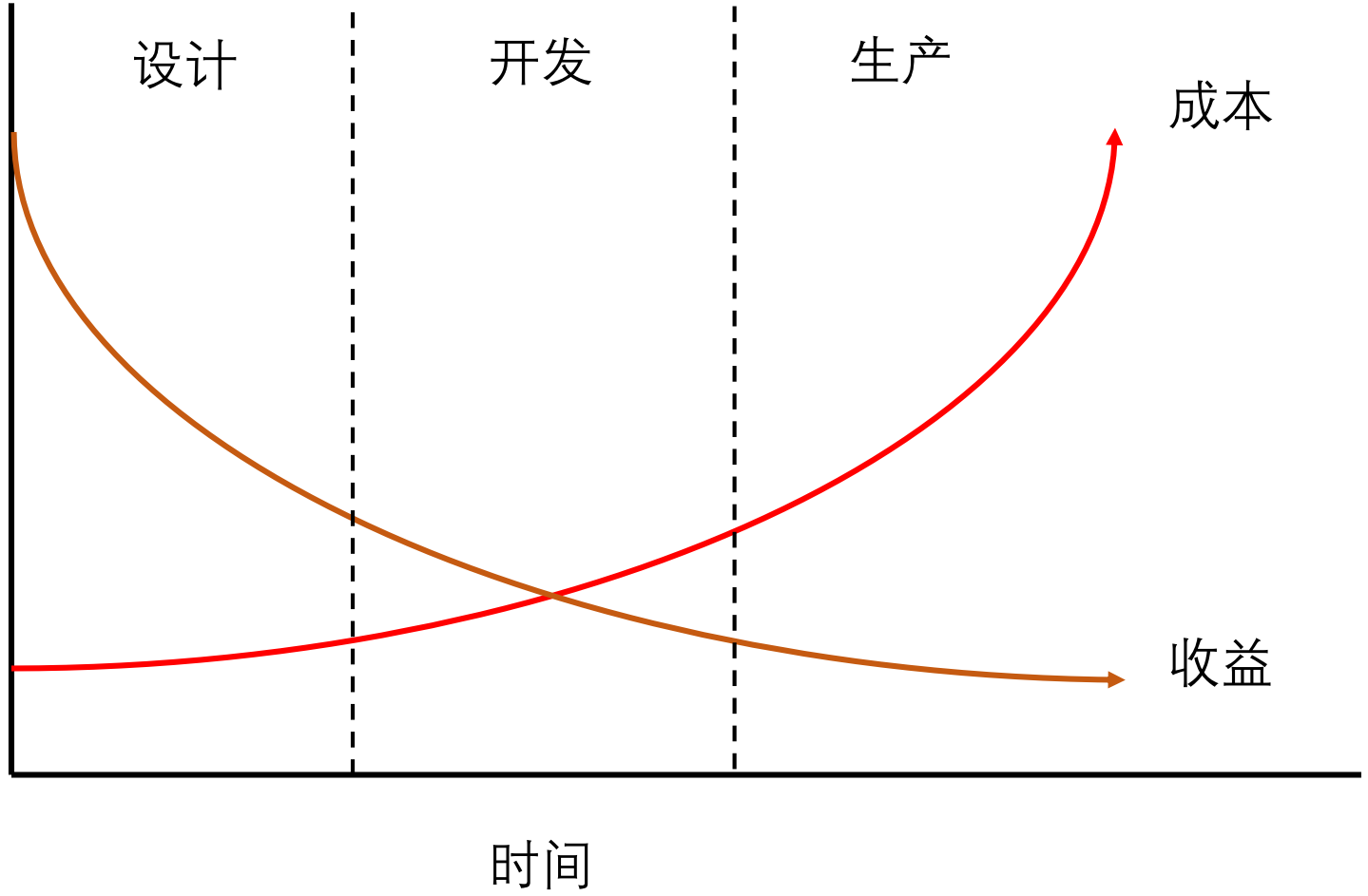
- ✓ 程序都是通过框架访问数据库的，数据库就是“黑盒”，会用就行，不用关心他的运行机制。
- ✓ 表、索引等数据库对象的设计能满足当前需求，行了，够了。
- ✓ 优化？很简单，就是建索引。
- ✓ 如果SQL返回值“看着”正确，就是他了。
- ✓ SQL在测试环境能跑，而且执行很快，上生产，一定没问题。
- ✓ 隐式转换、NULL值的不合理判断、重复索引、不必要的全表扫描等问题经常出现在我们日常的设计开发过程中。
- ✓ 迫于项目进度等方面的压力，功能实现优先级高于性能，且没有快速了解数据库设计开发质量的直观方法。
- ✓ …

- 案例介绍
- 现状和痛点
- 一些想法和实践

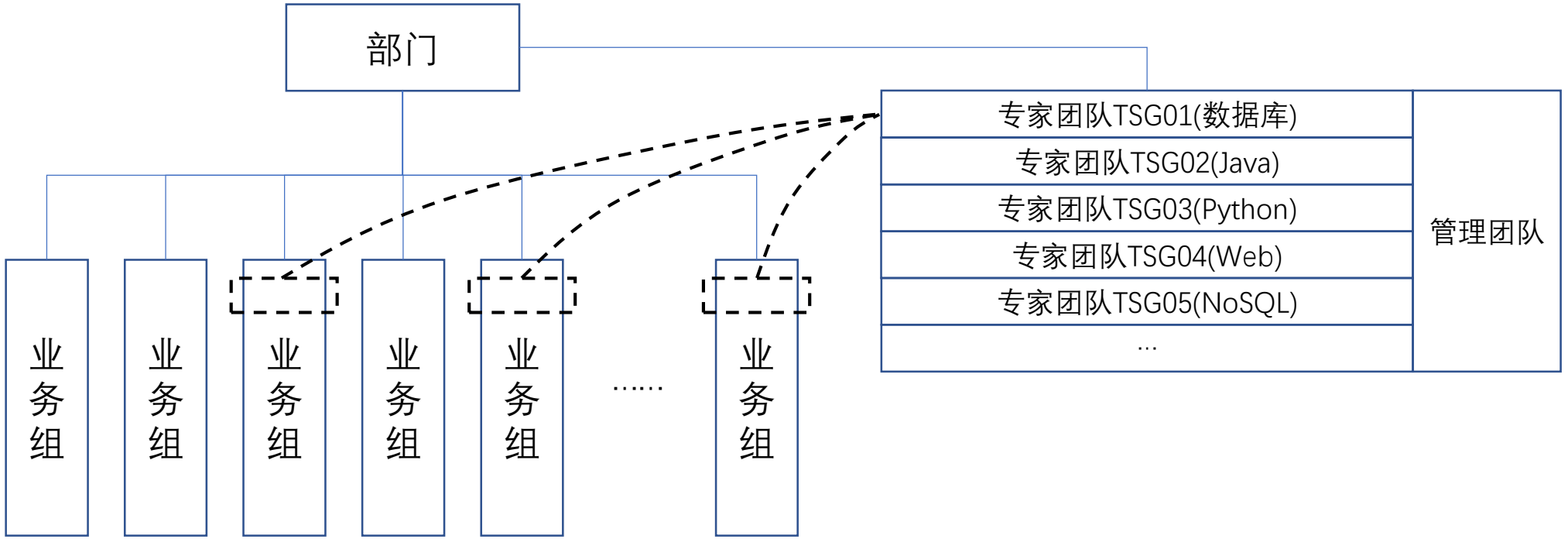
如何提升数据库设计开发能力？

- ✓ 不要被“表象”、“想当然”迷惑
- ✓ 需要“知其然”，更要“知其所以然”
- ✓ 培养“优化前置”的数据库设计开发思想
- ✓ 多动手、多实践、多总结

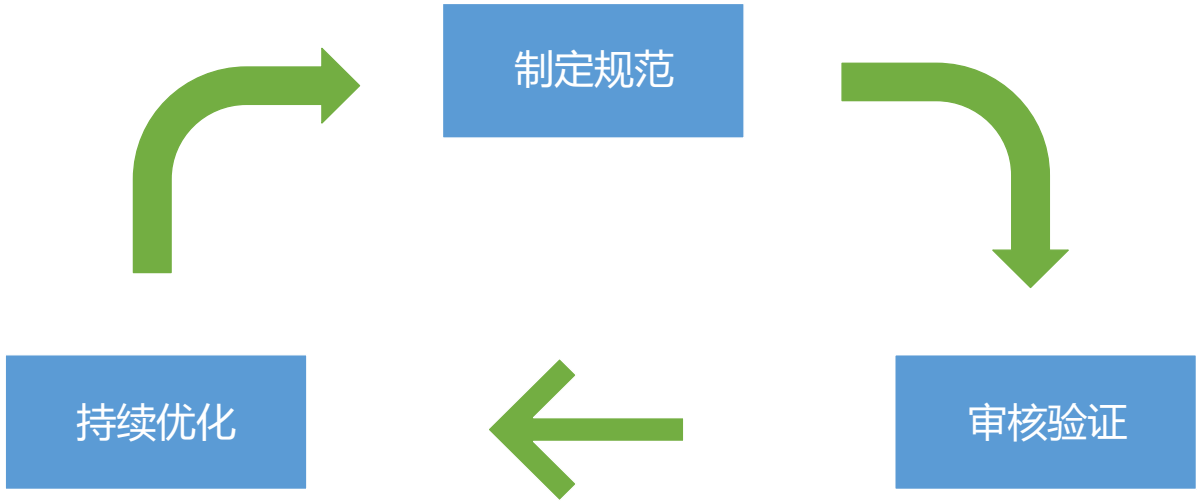
优化的成本和收益



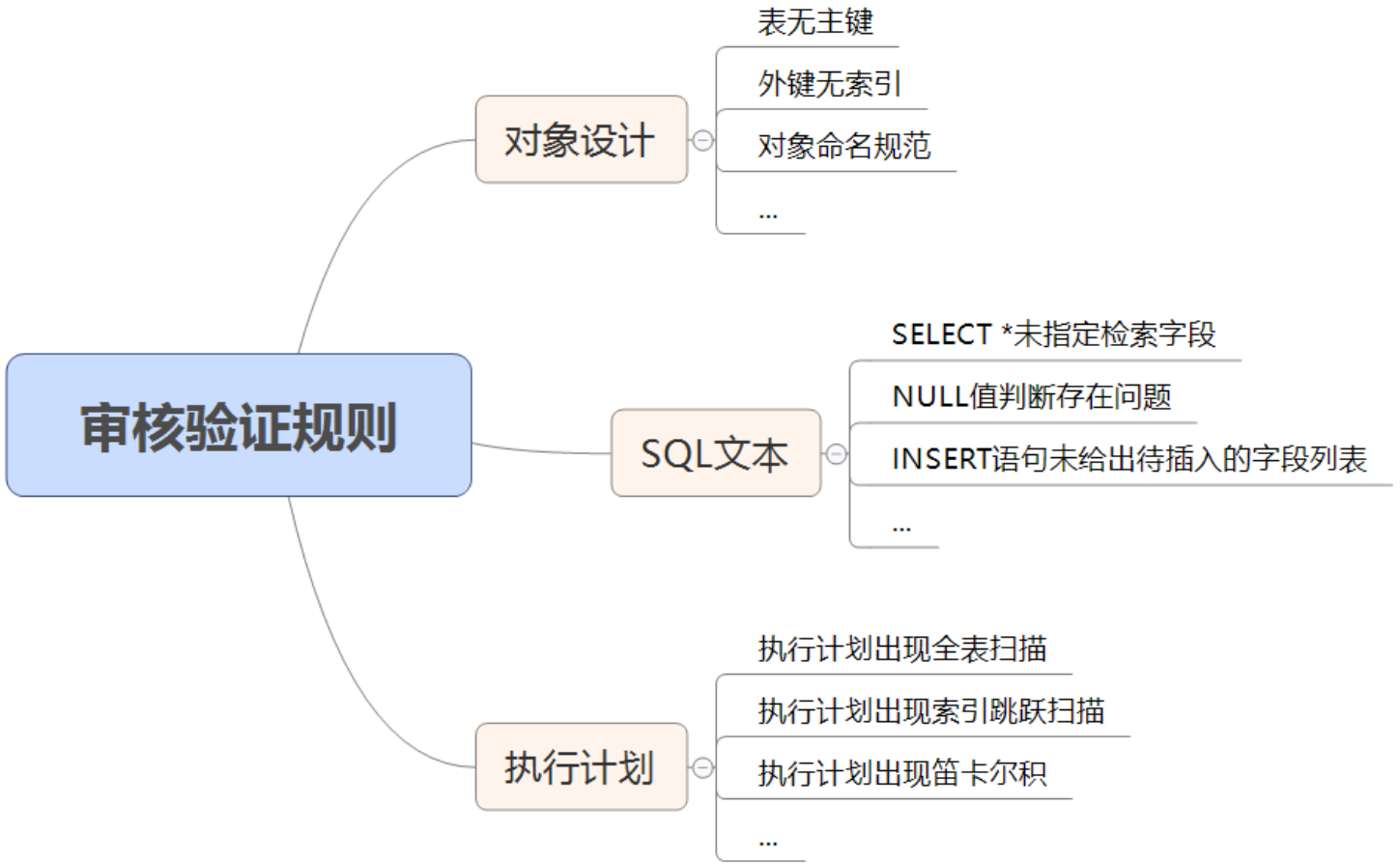
组建技术专家团队


部门
业务组
业务组
业务组
业务组
业务组
……
业务组
专家团队TSG01(数据库)
专家团队TSG02(Java)
专家团队TSG03(Python)
专家团队TSG04(Web)
专家团队TSG05(NoSQL)
…
管理团队

“规范-验证-优化”的闭环



审核验证规则


审核验证规则
对象设计
表无主键
外键无索引
对象命名规范
...
SQL文本
SELECT *未指定检索字段
NULL值判断存在问题
INSERT语句未给出待插入的字段列表
...
执行计划
执行计划出现全表扫描
执行计划出现索引跳跃扫描
执行计划出现笛卡尔积
...

回顾

- 案例介绍
- 现状和痛点
- 一些想法和实践

THANKS