DTCC 2010-2022

第十三届中国数据库技术大会

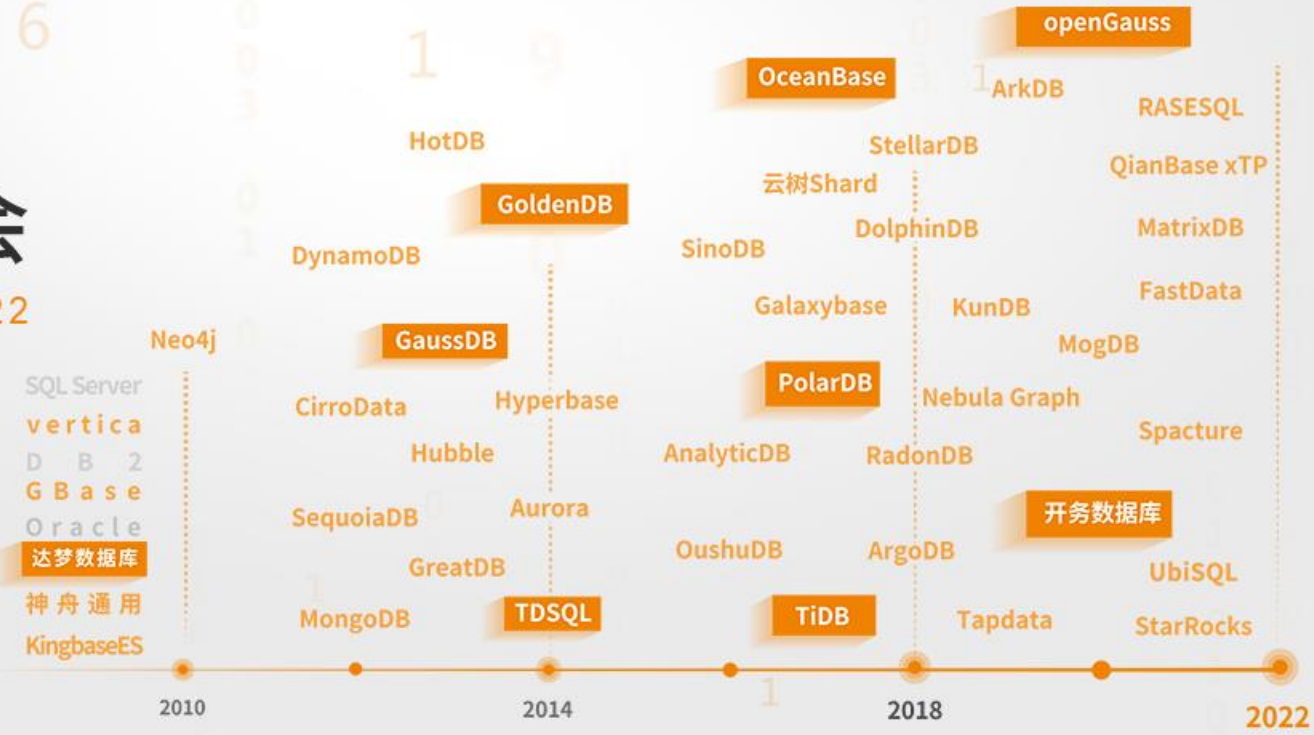
DATABASE TECHNOLOGY CONFERENCE CHINA 2022

数据智能 价值创新

IT168.com ChinaUnix.net ITPUB

线上直播 | 2022/12/14-16

数据来源：数据库产品上市商用时间

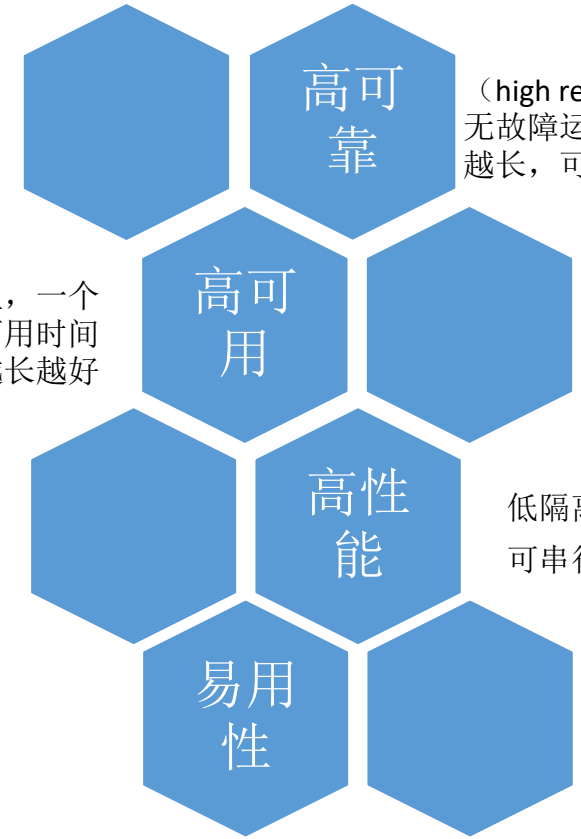


分布式数据库的分布式事务处理技术

李海翔

目录

1：分布式事务型数据库的理念与背景



（high reliability）定义为一个服务连续无故障运行的时间，无故障运行的时间越长，可靠性就越高。 99.999%

定义为在足够长的时间里，一个服务可用的时间，服务可用时间越长越好

低隔离别+高性能

可串行化+低性能

可靠性需求：根据其目标和实现方法的不同，可分为三个级别：

级别	目标	实现方法
1	减少系统的软、硬件故障	硬件：简化电路设计、提高生产工艺、进行可靠性试验等 软件：软件可靠性设计、软件可靠性测试等
2	即使发生故障，系统功能也不受影响	设备和链路的冗余设计、部署倒换策略、提高倒换成功率
3	尽管发生故障导致功能受损，但系统能够快速恢复	提供故障检测、诊断、隔离和恢复技术

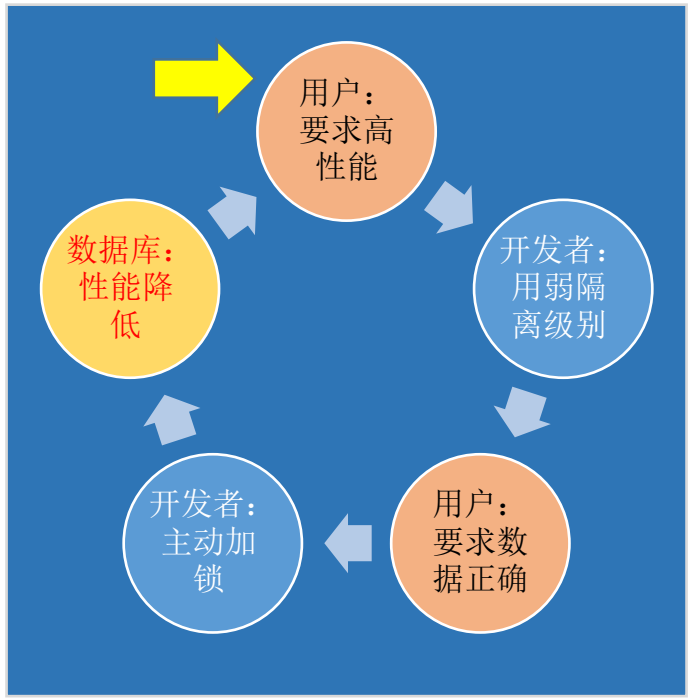
例如：
1 MySQL可串行化，性能很差
2 MySQL是单机系统

推测：
分布式的可串行化，是不是性能更差？

1.2 单机数据库的强一致+低性能

怪1：对一致性和性能的认知有误

数据库使用怪圈



Select…For Update

灵魂问题：

Q1：我们能精准地说出，什么情况下应该用For Update、什么情况下不用吗？

现实问题：

Q2：为什么我们需要精准地知道什么情况下应该用For Update、什么情况下不用？

1.2 单机数据库的强一致+低性能

测试目的：**验证用户主动加锁、不加锁的性能差异**

背景：Percona Server 8.0.18-9：验证用户主动加锁、不加锁的性能差异

背景：测试的recordcount设置为4000w，数据实际容量为90GB，稍大于buffer pool的大小， YCSB的读写4:1混合场景

	不加锁	用户加锁	
并发数	ops	ops	加锁影响
32	197377	183284	-7.14%
64	331071	180651	-45.43%
128	324274	97402	-69.96%
256	356958	170988	-52.10%
512	325337	162320	-50.11%
1024	283012	151466	-46.48%

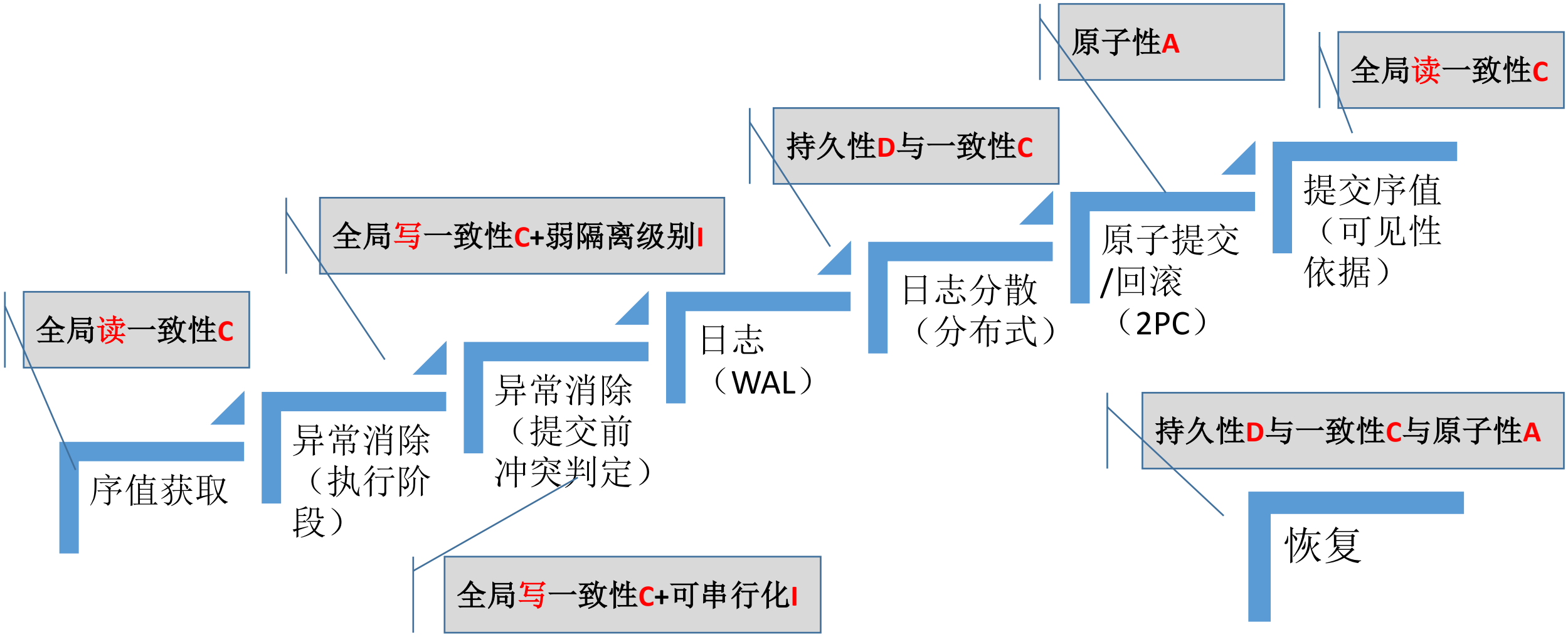
RC隔离级别下的性能数据

	不加锁	主动加锁	
并发数	ops	ops	加锁影响
32	167611	154499	-7.82%
64	263614	186389	-29.29%
128	336917	97356	-71.10%
256	356980	170766	-52.16%
512	326180	162960	-50.04%
1024	283079	151236	-46.57%

RR隔离级别下的性能数据

读写场景，用户加锁
性能损失很大

1.3 分布式数据库因分布带来的变化



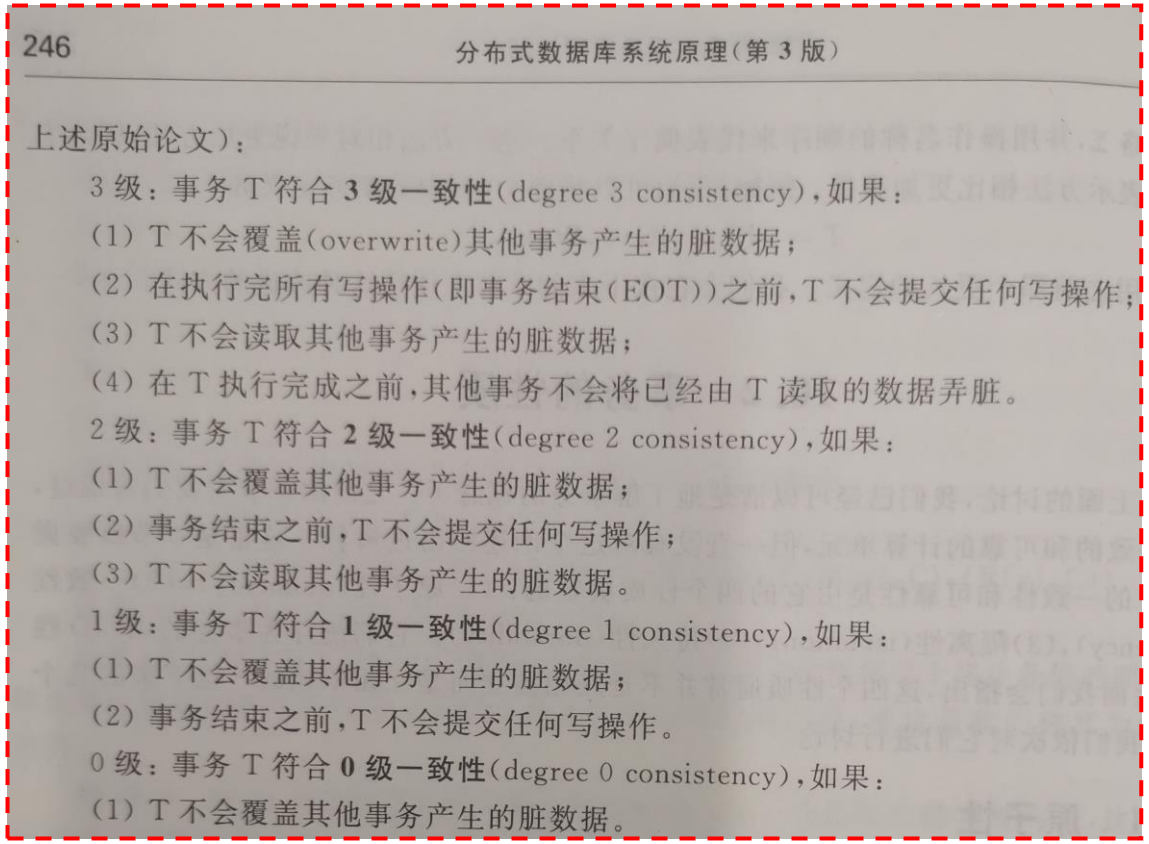
目录

2：分布式事务处理技术——2.1一致性与隔离性

Consistency (Correctness): *Consistency (database systems)*

Consistency ensures that a transaction can only bring the database **from one valid state to another**, maintaining database invariants: any data written to the database must be valid **according to all defined rules**,x including constraints, cascades, triggers, and any combination thereof.

https://wikipedia.org/wiki/Consistency
https://en.wikipedia.org/wiki/ACID#Consistency_(Correctness)



246 分布式数据库系统原理(第3版)

上述原始论文)：

3级：事务T符合**3级一致性**(degree 3 consistency)，如果：
(1) T不会覆盖(overwrite)其他事务产生的脏数据；
(2) 在执行完所有写操作(即事务结束(EOT))之前，T不会提交任何写操作；
(3) T不会读取其他事务产生的脏数据；
(4) 在T执行完成之前，其他事务不会将已经由T读取的数据弄脏。

2级：事务T符合**2级一致性**(degree 2 consistency)，如果：
(1) T不会覆盖其他事务产生的脏数据；
(2) 事务结束之前，T不会提交任何写操作；
(3) T不会读取其他事务产生的脏数据。

1级：事务T符合**1级一致性**(degree 1 consistency)，如果：
(1) T不会覆盖其他事务产生的脏数据；
(2) 事务结束之前，T不会提交任何写操作。

0级：事务T符合**0级一致性**(degree 0 consistency)，如果：
(1) T不会覆盖其他事务产生的脏数据。

Jim Gray, Raymond A. Lorie, Gianfranco R. Putzolu, and Irving L. Traiger. 1976. Granularity of Locks and Degrees of Consistency in a Shared Data Base.

怪2：一致性没有形式化标准

2：分布式事务处理技术——2.1一致性与隔离性

怪3：一致性与数据异常与隔离级别的关系，不明确

- Q1：有多少种数据异常?
- Q2：数据异常之间的关系是什么?
- Q3：数据异常和变量、并发事务的个数的关系?
- Q4：谓词类数据异常（如幻读）、非谓词类数据异常的关系（如不可重复读）
- Q5：数据一致性的定义未统一
- Q6：隔离级别的定义未统一

- A1：收集了业界提出的近20种异常，TDSQL提出20+种新的数据异常
- A2：提出数据异常的通用定义，涵盖所有数据异常（包括已知和本项研究新发现的）
- A3：把相关因素统一在一个模型下（数据状态模型）；对数据异常分类（基于数学的形式化定义加以分类，找出每类异常的“特征”）
  ——彻底解释清楚数据异常是什么（用环形象化表达所有数据异常）
- A4：把两种貌似不同角度的数据异常，统一在了一个形式化定义的框架下
- A5：直接用数据异常的角度阐述数据一致性概念（**重新定义一致性**），而不是用可串行化的理论阐述数据一致性概念（形象且直接）
- A6：先对数据异常分类，然后在分类的基础上定义隔离级别，使得隔离级别定义：**完整**（基于全部数据异常）、**有层次**（基于数据异常分类）、**打破传统对隔离级别的认知**（传统观点，认为各种隔离级别会有不同性能）
  ——弱化隔离级别的作用（有助于简化并发访问控制算法）

2：分布式事务处理技术——2.1一致性与隔离性

表 1　现存数据异常汇总

编号	异常名称(异常名称,文献,年份)	形式化定义
1	Dirty Write,[16] 1992,[18] 1995	$W_1[x] \dots W_2[x] \dots (C_1 \text{ or } A_1) \text{ and } (C_2 \text{ or } A_2) \text{ in any order}$
2	Lost Update,[18] 1995	$R_1[x] \dots W_2[x] \dots W_1[x] \dots C_1$
3	Dirty Read,[16] 1992,[18] 1995	$W_1[x] \dots R_2[x] \dots (A_1 \text{ and } C_2 \text{ in either order})$
4	Aborted Reads,[20] 2000,[19] 2015	$W_1(x1:i) \dots R_2(x1:i) \dots (A_1 \text{ and } C_2 \text{ in any order})$
5	Fuzzy OR Non-Repeatable Read,[16] 1992	$R_1[x] \dots W_2[x] \dots C_2 \dots R_1[x] \dots C_1$
6	Phantom,[16] 1992	$R_1[P] \dots P_2[y \text{ in } P] \dots C_2 \dots R_1[P] \dots C_1$
7	Intermediate Reads,[20] 2000,[19] 2015	$W_1(x1:i) \dots R_2(x1:i) \dots W_1(x1:j) \dots C_2$
8	Read Skew,[18] 1995	$R_1[x] \dots W_2[x] \dots W_2[y] \dots C_2 \dots R_1[y] \dots (C_1 \text{ or } A_1)$
9	未命名的异常,[21] 2000	$R_3[y]R_1[x]W_1[x]R_1[y]W_1[y]C_1\ R_2[x]W_2[x]R_2[z]W_2[z]C_2\ R_3[z]C_3$
10	fractured reads,[31] 2014,[3] 2017	$R_1[x_0] \dots W_2[x_1] \dots W_2[y_1] \dots C_2 \dots R_1[y_1]$
11	Serial-Concurrent-Phenomenon,[25] 2014	$R_1[x_0] \dots W_2[x_1] \dots W_2[y_1] \dots C_2 \dots R_1[y_1]$
12	Cross-Phenomenon,[25] 2014	$R_1[x_0] \dots R_2[y_0] \dots W_3[x_1] \dots C_3 \dots W_4[y_1]C_4 \dots R_2[x_1] \dots R_1[y_1]$
13	long fork anomaly,[3] 2017	$R_4[x_0] \dots W_1[x_1] \dots R_3[y_0] \dots R_3[x_1] \dots W_2[y_1] \dots R_4[y_1]$
14	causality violation anomaly,[3] 2017	$R_3[x_0] \dots W_1[x_1] \dots C_1 \dots R_2[x_1] \dots W_2[y_1] \dots C_2 \dots R_3[y_1]$
15	A read-only transaction anomaly,[11] 1982,[22] 2004	$R_2(x_0,0)R_2(y_0,0)R_1(y_0,0)W_1(y_1,20)C_1\ R_3(x_0,0)R_3(y_1,20)C_3\ W_2(x_2,-11)C_2$
16	Write Skew,[18] 1995	$R_1[x] \dots R_2[y] \dots W_1[y] \dots W_2[x] \dots (C_1 \text{ and } C_2 \text{ occur})$
17	Predicate-Based Write Skew,[23] 2005	$R_1(\{x_0 \text{ in } P\}) \dots R_2(\{y_0 \text{ in } P\}) \dots W_1[\{y_1 \text{ in } P\}] \dots C_1 \dots W_2[\{x_1 \text{ in } P\}]$
18	读半已提交数据异常(分布式数据库) ,[33] 2019	$R_1[x] \dots W_2[x] \dots W_2[y] \dots C_2 \dots R_1[y] \dots C_1$

SQL标准对于隔离级别的定义，还可信吗？

（ANSI）可串行化的定义，还可信吗？

2：分布式事务处理技术——2.1一致性与隔离性

软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2022,33(3):909−930 [doi: 10.13328/j.cnki.jos.006442]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

数据库管理系统中数据异常体系化定义与分类*

李海翔[1], 李晓燕[2], 刘 畅[1], 杜小勇[3,4], 卢 卫[3,4], 潘安群[1]

[1](腾讯科技(北京)有限公司, 北京 100080)
[2](北京大学 数学科学学院 信息与计算科学系, 北京 100871)
[3](数据工程与知识工程教育部重点实验室(中国人民大学), 北京 100872)
[4](中国人民大学 信息学院, 北京 100872)
通信作者: 李海翔, E-mail: blueseali@tencent.com

摘 要: 数据异常尚没有统一的定义, 其含义是指可能破坏数据库一致性状态的特定数据操作模式. 已知的数据异常有脏写、脏读、不可重复读、幻读、丢失更新、读偏序和写偏序等. 为了提高并发控制算法的效率, 数据异常也被用于定义隔离级别, 采用较弱的隔离级别以提高事务处理系统的效率. 体系化地研究了数据异常以及对应

数据库管理系统中数据异常体系化定义与分类（jos.org.cn）
http://www.jos.org.cn/jos/article/abstract/6442?st=search

研究内容：

1 定义什么是数据异常？（只包括实体类数据异常）

2 对数据异常分类

3 用数据异常定义隔离级别

4 用数据异常体系化研究结果的结论，描述并发算法

研究方法和结果：

1 基于数学：扩展了冲突图（把3种冲突关系扩展为9种），使得冲突图可以包括事务的所有操作（增加了回滚和提交操作）

2 基于扩展后的冲突图：用扩展的冲突有向环图，定义什么是数据异常

3 效果：扩展了冲突图的作用（从可串行化调度 扩展为 定义所有数据异常）

4 过程：基于图大小（顶点数和边数，定义无穷多个异常）和图的边的特征（边的个数有限），研究用数据异常之间的关系，对数据异常进行分类

5 结果：建立起体系化研究数据异常的框架（回答了数据异常有无穷多个，但可以归类为3种）

6 应用：定义了隔离级别（基于全部数据异常定义隔离级别，使得隔离级别不再遗漏某个数据异常）、初步描述了并发访问算法

2：分布式事务处理技术——2.1一致性与隔离性

存在的现象 | 问题 | 核心技术+过往结果 | 新未来

把语义形式化

把无限个数的异常的体系完备化

扩充了冲突可串行化

一致性定义1：传统教科书的定义方式

不违反完整性约束

一致性遵从用户语义，事务调度器难实现用户语义

一致性定义2：Jim Gray的定义方式

一致性分为四种级别，分别消除四种数据异常

用有限个数的数据异常表示一致性，不完备

传统一致性理论：缺乏形式化定义

一致性理论和实践割裂

工程实践：另起炉灶

数据库引擎实现

基于冲突可串行化实现一致性的事务调度器

冲突可串行化不能表示所有数据异常

①定义不相同

②理论不能指导实践

③冲突依赖图

①一致性理论缺失

偏序环图形式化定义一致性

偏序对与偏序环

数据异常标准化

异常分类，变无穷为有限种类

分类后的异常满射为一致性模型

②数据库不可衡量

不能定义所有异常

冲突关系与冲突环

冲突关系个数有限

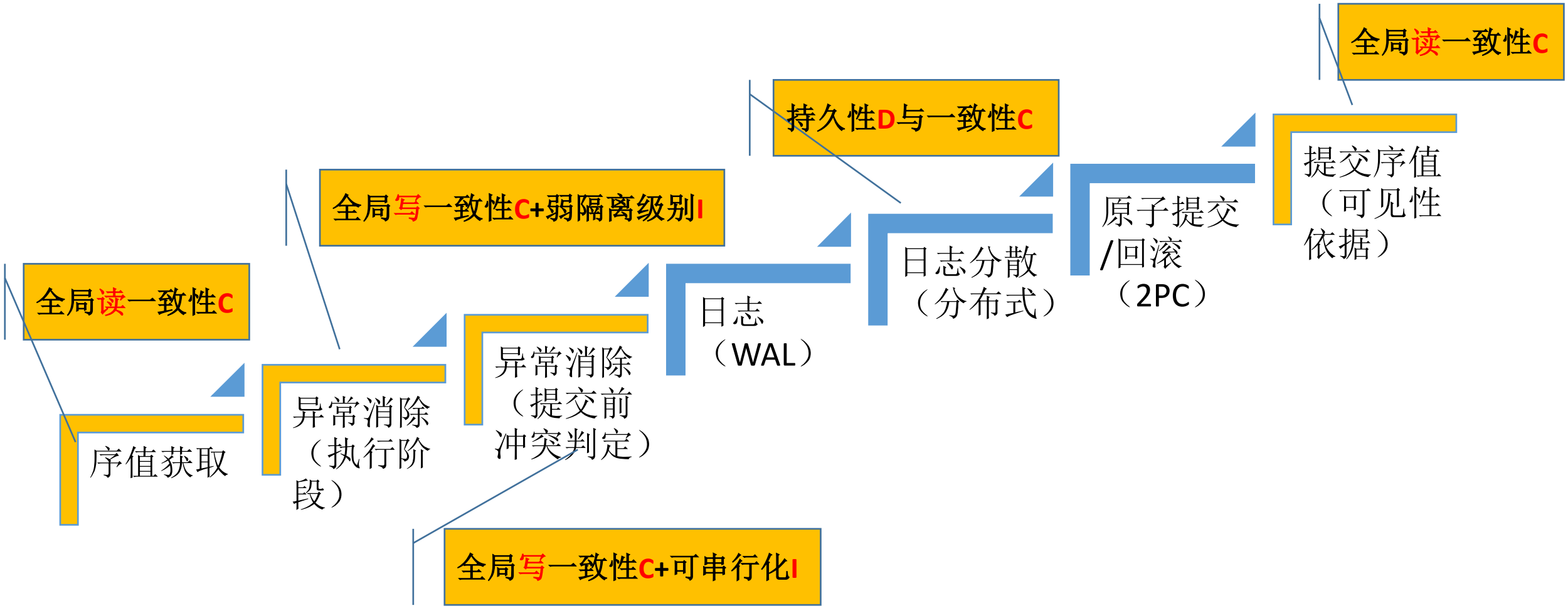
①偏序环图包涵冲突依赖图

②形式化的理论，可检测数据库的实现

一致性理论和实践统一

1 并发算法
2 分布式并发算法
3 死锁检测
4 一致性的体系化、形式化定义（ACID+CAP）

2：分布式事务处理技术——2.1一致性与隔离性

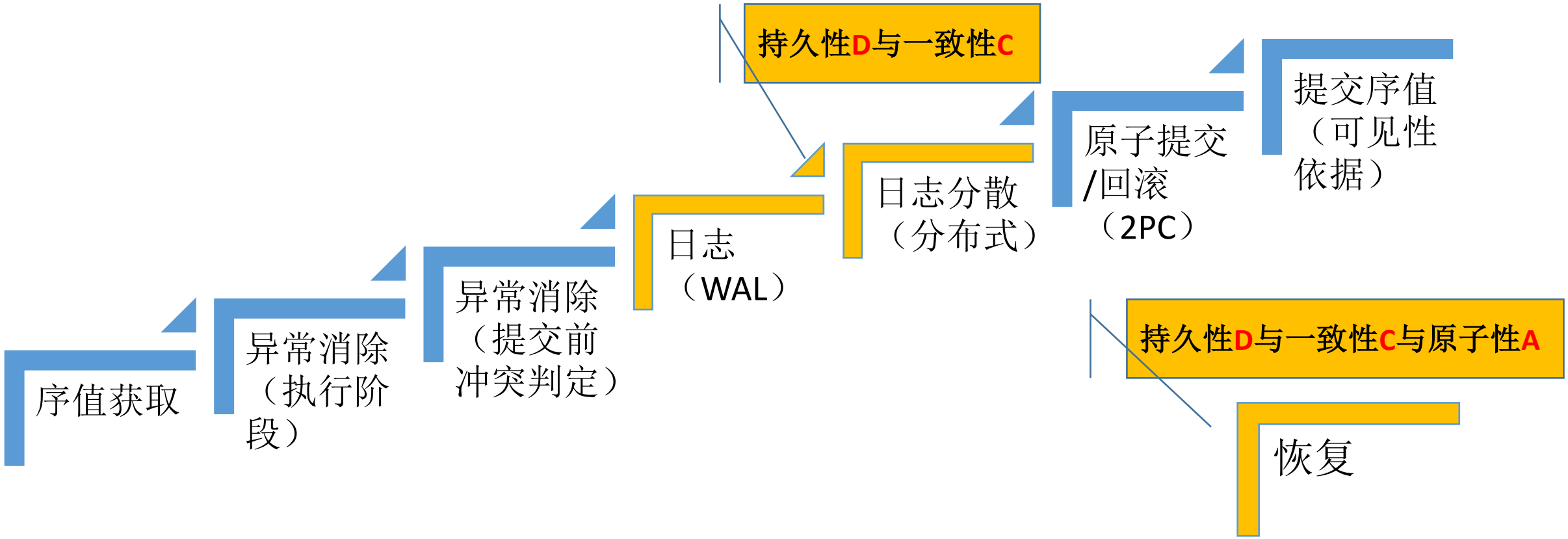


2：分布式事务处理技术——2.1一致性与隔离性

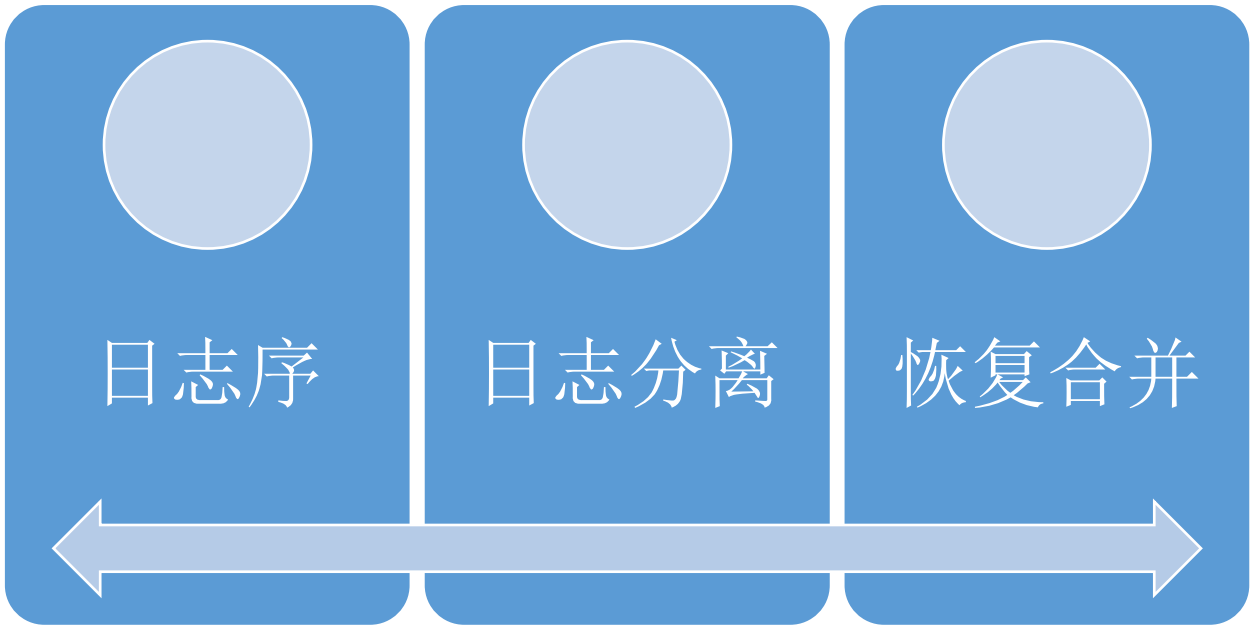
不一致性 == 有数据异常

一致性 == 无数据异常

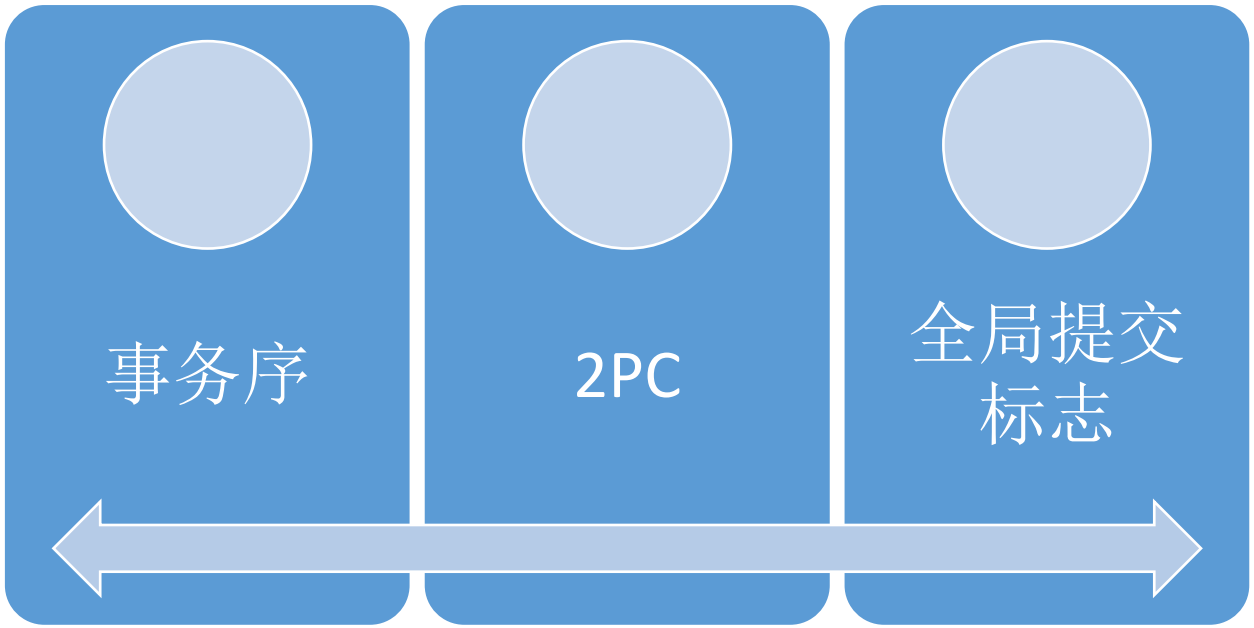
2：分布式事务处理技术——2.2 持久性与可靠性



2：分布式事务处理技术——2.2 持久性与可靠性

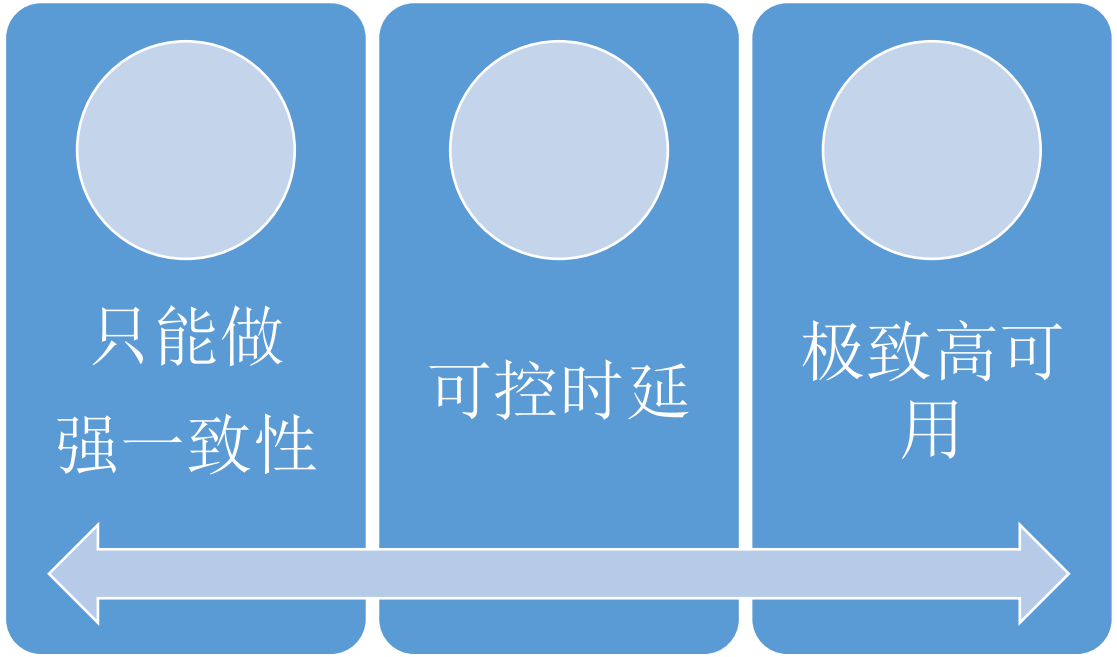


2：分布式事务处理技术——2.3 原子性与一致性

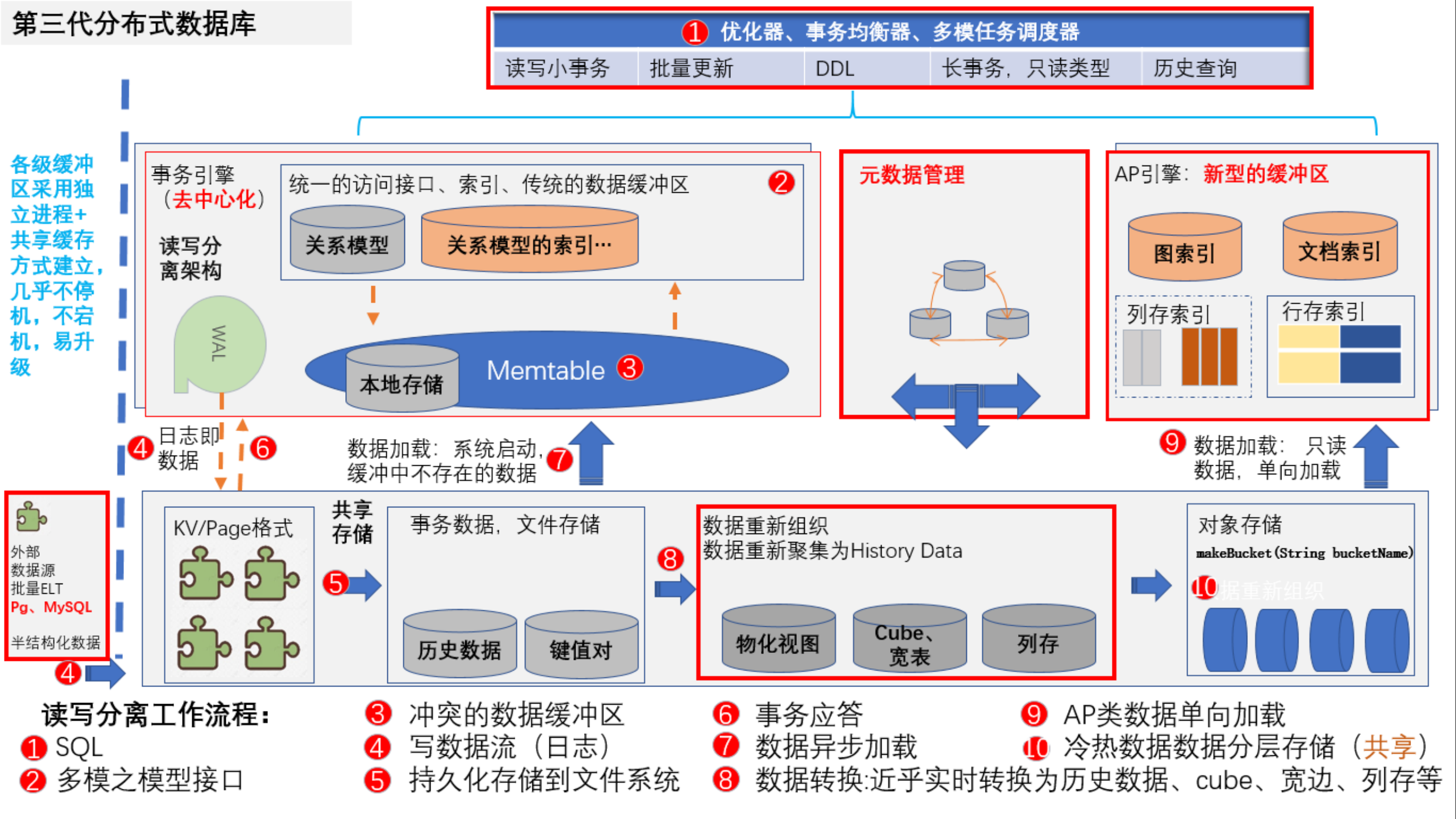


目录

3：第三代分布式数据库



第三代分布式数据库



读写分离工作流程：

1. SQL
2. 多模之模型接口
3. 冲突的数据缓冲区
4. 写数据流（日志）
5. 持久化存储到文件系统
6. 事务应答
7. 数据异步加载
8. 数据转换:近乎实时转换为历史数据、cube、宽边、列存等
9. AP类数据单向加载
10. 冷热数据数据分层存储（共享）

3：第三代分布式数据库

	Spanner	CRDB	OB	veDB
架构	SN+文件级共享存储	SN	SN	SN+日志级共享存储
事务强一致性	严格可串行化	因果+可串行化	SI	严格可串行化、可串行化
多写架构	--	可串行化多写	SI多写	可串行化多写
可扩展	--	扩展性较弱	千台级规模扩展	存储无限扩展 计算千台级扩展
高可用	数据多副本提供系统级的高可用，事务级瞬时闪断	数据可灵活分布，数据多副本提供系统级的高可用	数据多副本提供系统级的高可用	数据级高可用：日志服务器； 内存级高可用：主主互备、RO切换
高可靠	--	更资料截取（Change Data Capture，CDC）、备份与恢复、管理API，以及99.99%正常运行时间	数据备份和日志备份	库、表级备份恢复 日志级备份恢复 数据块级备份恢复
可控时延	超大时延超低性能	高时延（全球级、时延未做特定设计）低性能	高时延、扩展节点获高性能	低时延高性能、扩展节点获更高性能


openGauss
OceanBase
ArkDB
RASESQL
HotDB
StellarDB
QianBase xTP
云树Shard
GoldenDB
DolphinDB
MatrixDB
DynamoDB
SinoDB
FastData
Galaxybase
KunDB
SQL Server
vertica
D B 2
GBase
Oracle
达梦数据库
Spacture
开务数据库
UbiSQL
神舟通用
TDSQL
TiDB
Tapdata
StarRocks
KingbaseES
2010
2014
2018

THANKS