

PostgreSQL 技术布道 & 阿里云 PG 内核优化

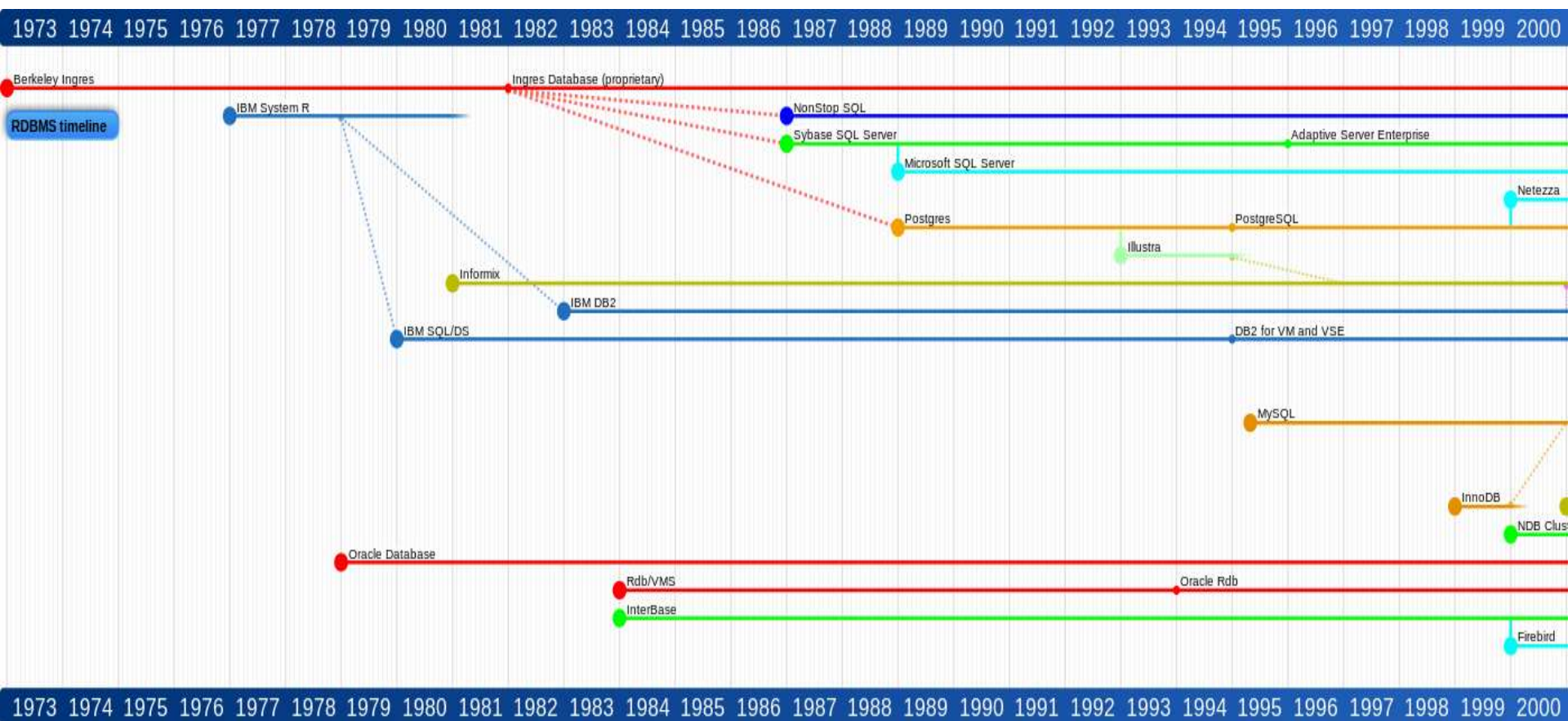
digoal

9/7/2016

议题

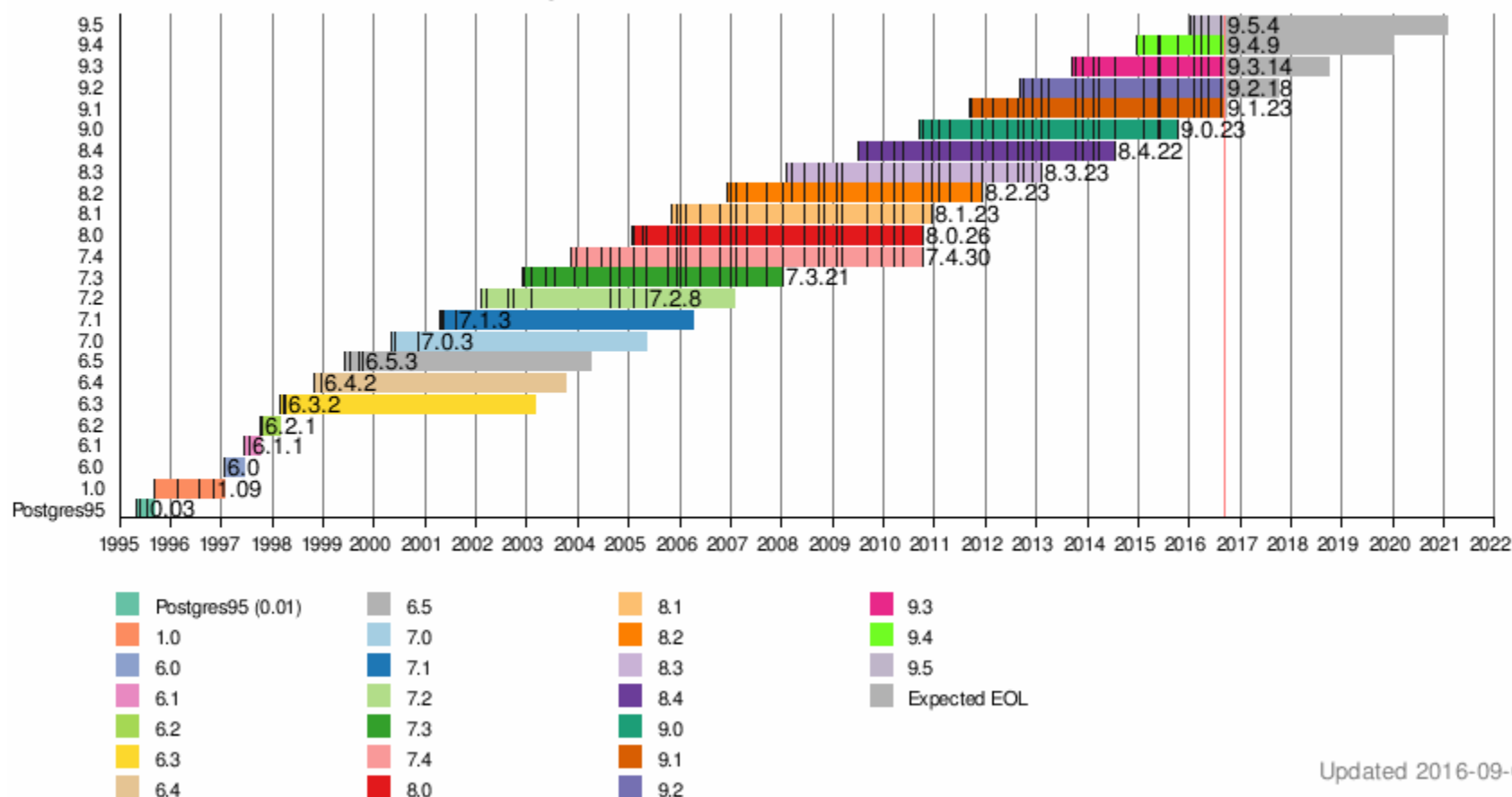
- PostgreSQL 前世今生
- PostgreSQL 特性
- PostgreSQL 适应的场景
- 最佳实践
- 阿里云PostgreSQL内核优化

PostgreSQL发展历史



版本迭代

PostgreSQL release timeline




Updated 2016-09-06

里程碑

7.2

2002年

- GIS



穿越回到2002

PostGIS(覆盖民用、军用、科研)

- 类型
 - 点
 - 二维、三维（经纬度、海拔）、多维、地址类型
 - 线
 - 闭合线段、开放线段、多点线段
 - 面
 - 圆、椭圆、矩形、长方形、正方形、规则多边形、不规则多边形、曲面
 - raster
 - GIS图数据, jpegs, tiffs, pngs, digital elevation models
 - 测绘、航天、天文应用
 - 拓扑
- 操作符
 - 点面判断、距离、面积、体积、叠加、相减、长度、弧度、夹角、pixel相关运算、raster相关运算
- 索引
 - 距离运算、距离排序、包含判断、相交判断、
- 函数
 - 区域内线段长度（城市道路长度）、路径成本(坡度、长度、权重)计算、最佳路径计算、区域面积（城市面积）



里程碑

8.4

- SQL:2008
- 窗口查询
- 并行恢复
- 递归查询

窗口查询 - 数据透视

- 数据结构

- 学号、**省份、城市、学校、年级、班级、科目**、分数

- 查询需求

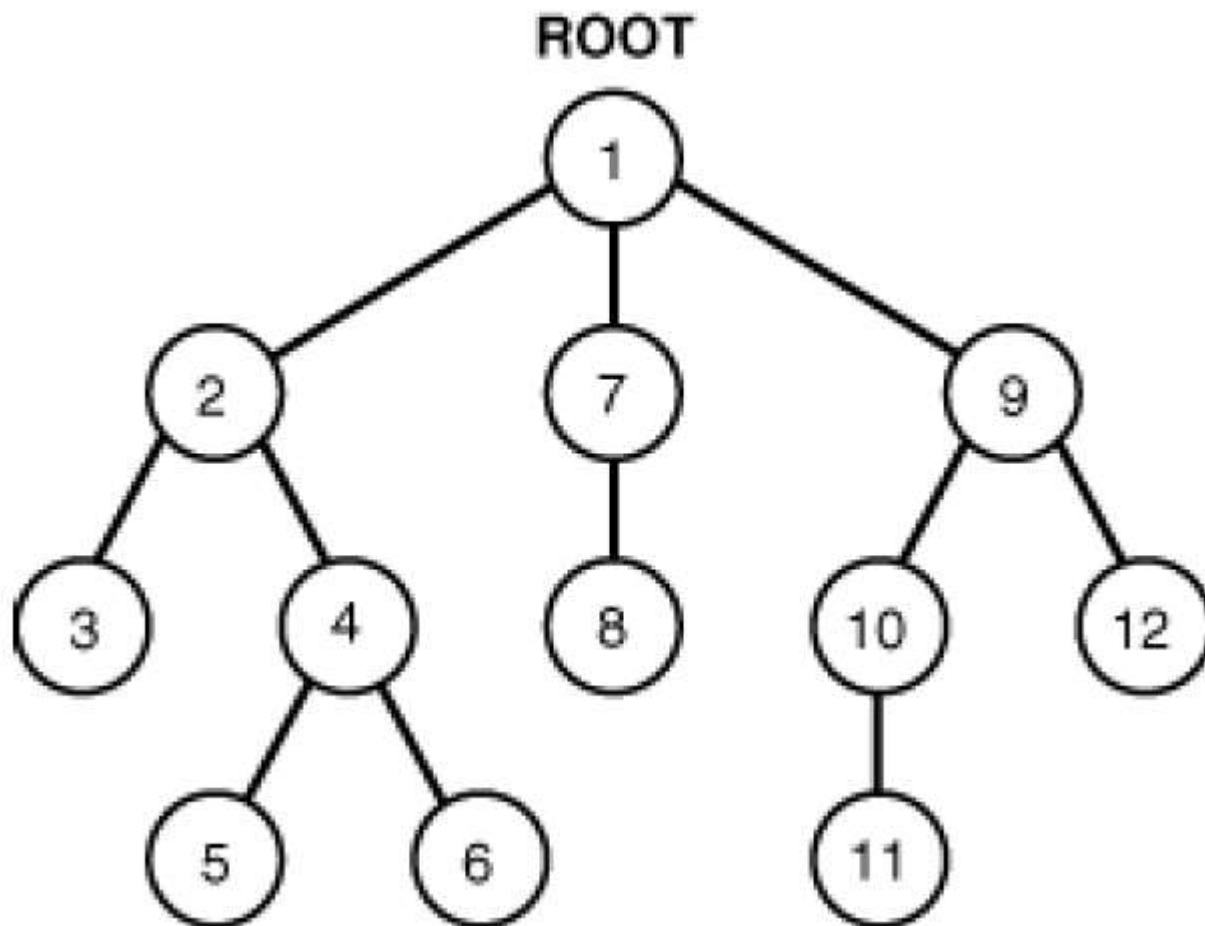
- 学号、(**窗口**)、分数、并列名次、名次、与第一名的分差、与前一名的分差、第一名到当前学生的平均分（帧）、处于哪个分数区间（比如分了**10**个区间）



- 语法

- **window_func()** over(partition by **part_col?** order by **col?**)
 - first_value()、lead(n)、lag(n)、dense_rank()、ntile(bucket)、avg(),

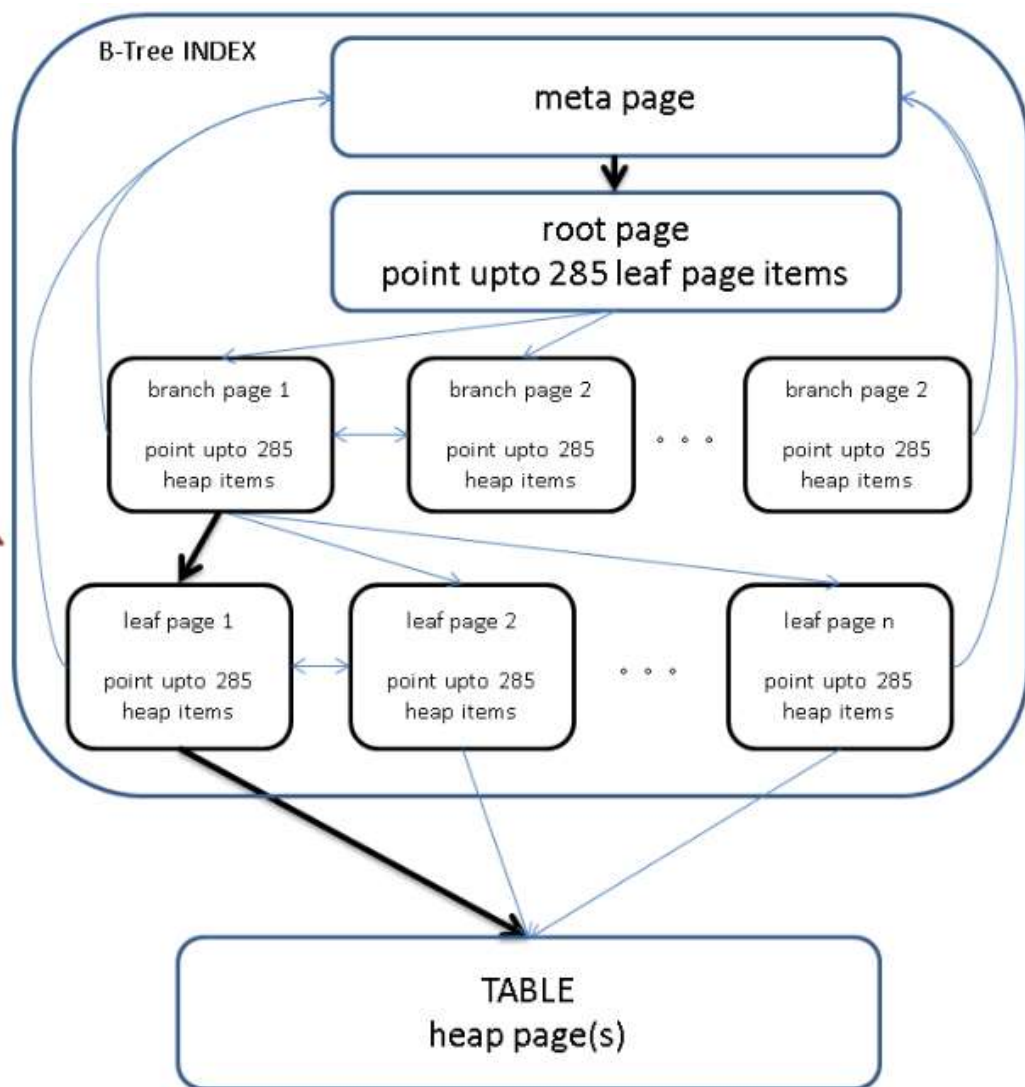
递归查询 - 支持树形结构数据



递归查询 - 优化count(distinct)

2级索引，包括1个root page，
1或多个branch page，多个
leaf page。
最多存储 285^3 条记录。
branch page是“双向链表”。

按层级收敛
减少数据块扫描



并行恢复

- 逻辑备份集恢复
- 多表并行还原
- 多个索引并行创建

里程碑

9.0

- 异步流式复制
- 快速大版本升级
- online code

异步流复制

- 基于REDO record，异步流式复制(不需要等到xlog切换)
- 延迟毫秒内
- 支持一主多备
- 备库read only时，不影响接收redo
- 备库read only时，不影响恢复（除非恢复时要擦除已对记录持锁的查询）
- 主备物理块级别一致

快速大版本升级

- 元数据（1万张表 - 约十几MB）
- 元数据导出导入
- 不需要动用户的数据文件
- 统计信息需要重新收集

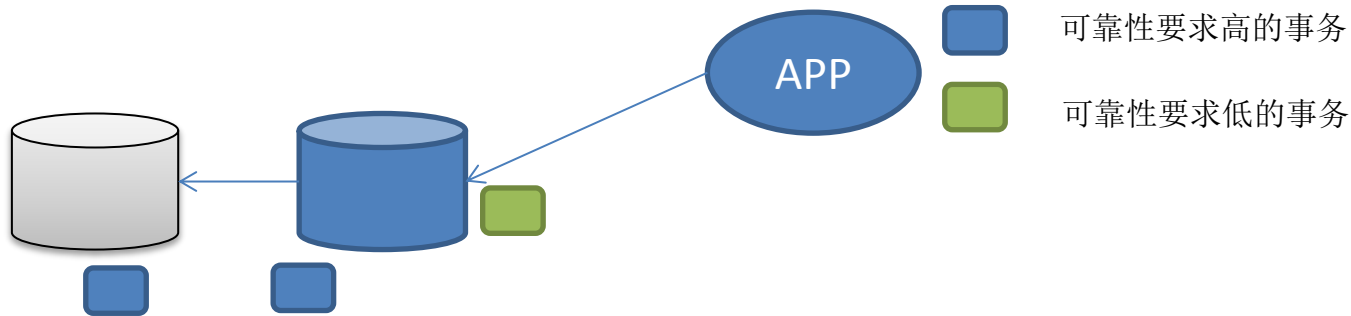
里程碑

9.1

- 同步流式复制
- KNN索引支持
- FDW接口

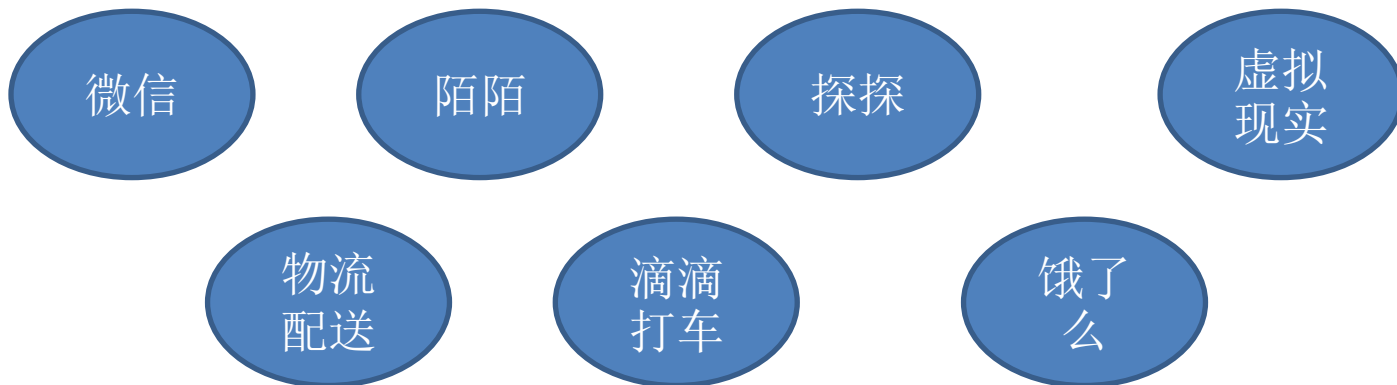
同步流复制

- 用户可以根据事务可靠性要求，选择本事务是否需要同步复制。

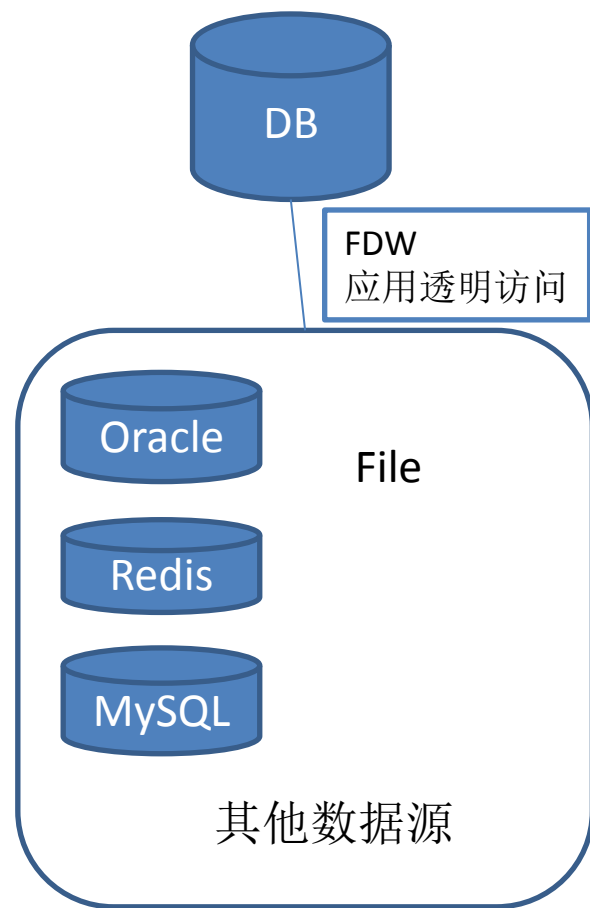
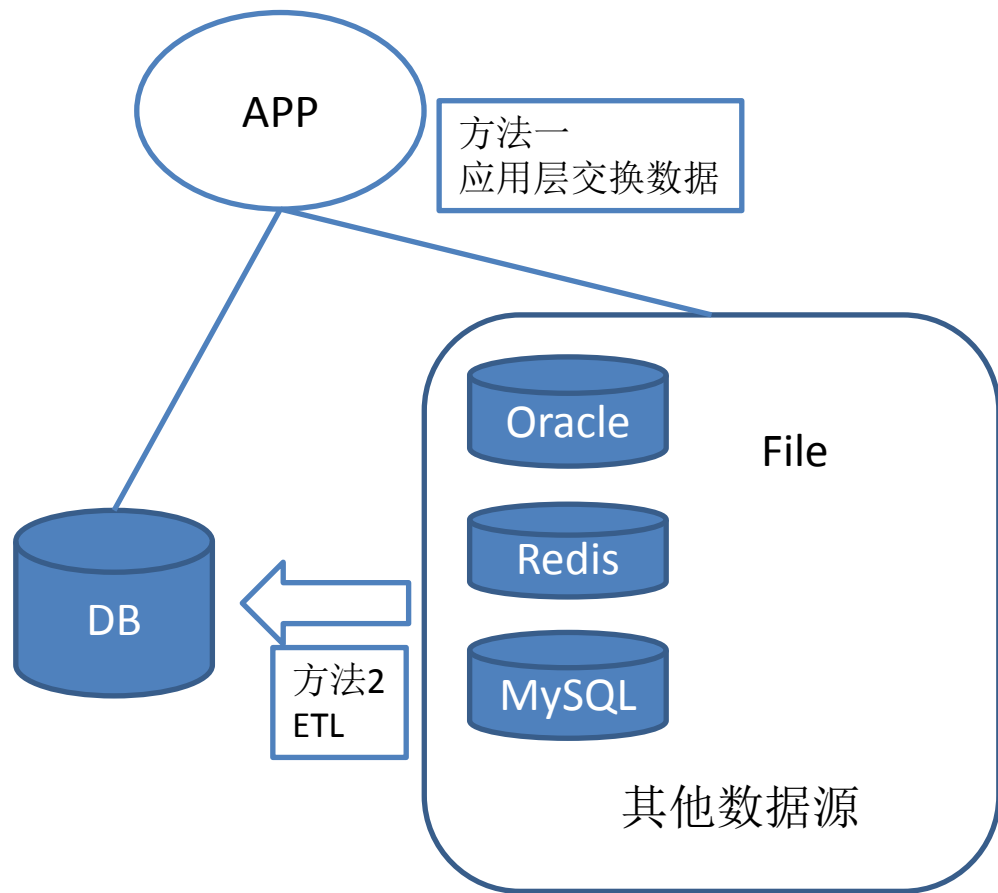


KNN查询索引支持

- 近邻查询
 - 根据经纬度查询最近用户
 - 根据数值查询最相邻数值
 - 根据文本相似度查询最相似文本
- 距离排序



FDW接口



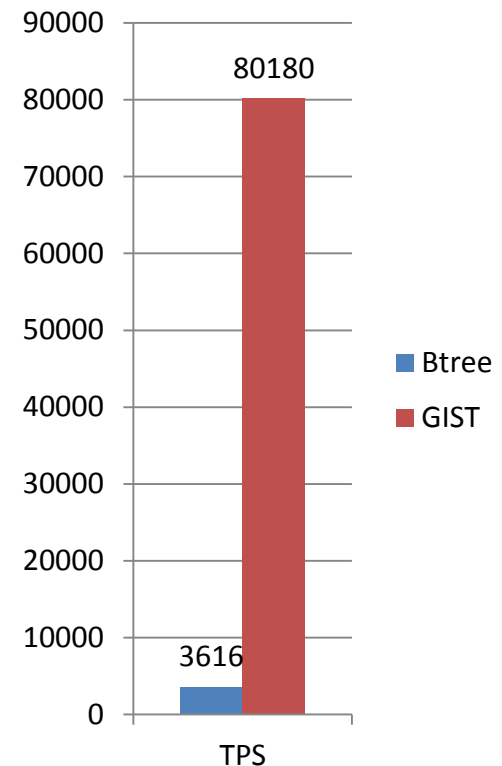
里程碑

9.2

- SP-GiST索引
- 范围类型
- JSON
- 支持plv8
- 级联流式复制

GiST取代B-Tree用于范围匹配查询

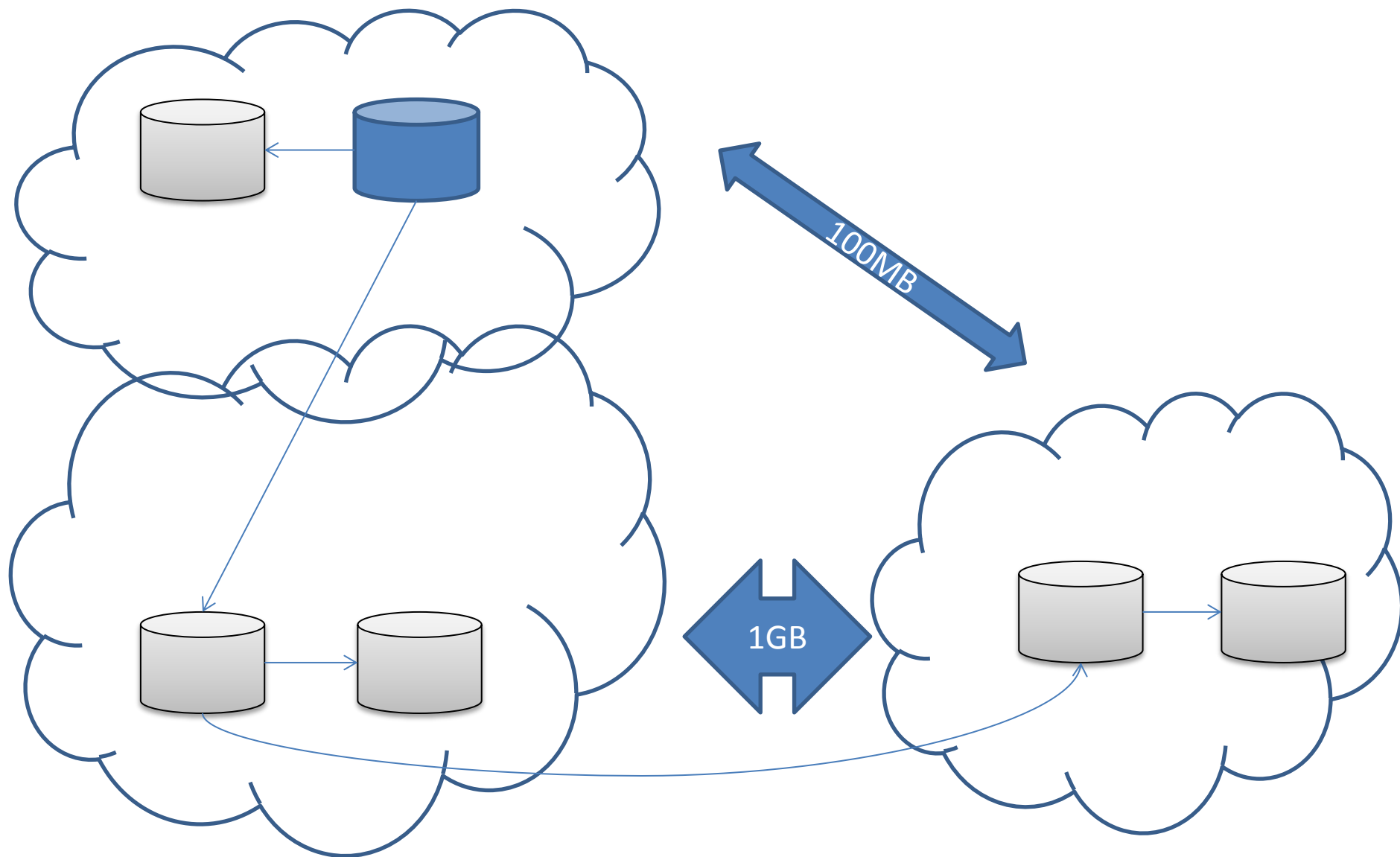
- 快速范围查询，**智能DNS**，判断来源IP落在哪个IP地址段内
- 传统方法
 - postgres=# create table tbl(id int,ip_start int8,ip_end int8);
 - postgres=# create index idx_tbl on tbl using **btree(ip_start,ip_end);**
- 使用范围类型
 - postgres=# create table tbl_r(id int,ip_range int8range);
 - postgres=# create index idx_tbl_r on tbl_r using **spgist(ip_range);**
 - 或
 - postgres=# create index idx_tbl_r1 on tbl_r using **gist(ip_range);**
- 传统范围查询
 - postgres=# select * from tbl where ? **between ip_start and ip_end;**
- 使用范围类型的范围匹配操作符，利用gist/spgist索引
 - postgres=# select * from tbl_r where **ip_range @> ?**
 - OR 不改变原有数据结构，使用 函数索引
 - create index idx on tbl using gist (int8range(ip_start,ip_end+1));
 - select * from tbl where int8range(ip_start,ip_end+1) @> ?;



JSON支持

- 非结构化数据JSON支持
- PLV8引擎，服务端编程
- JUST IO
- no op,am,func support

级联复制

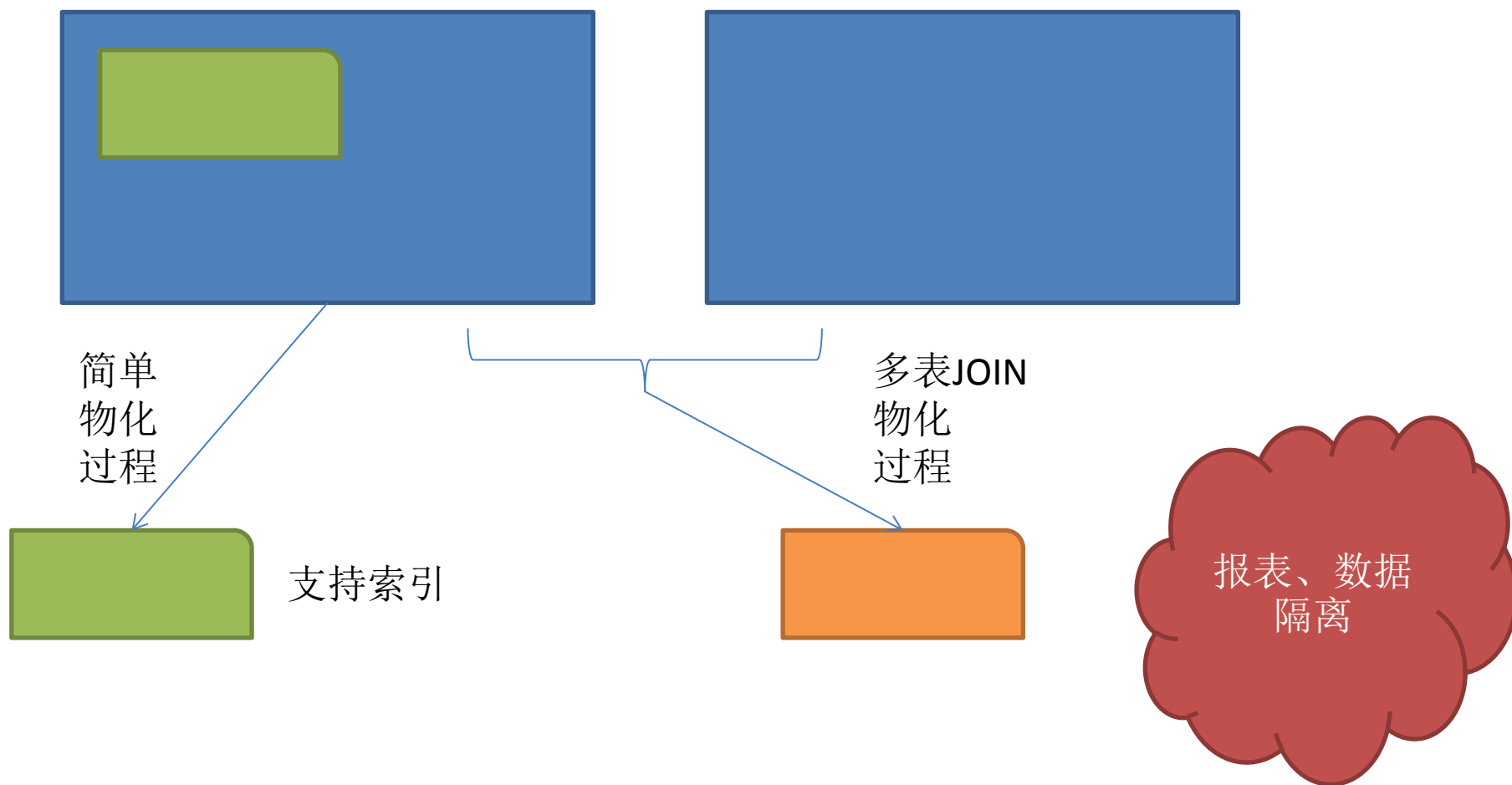


里程碑

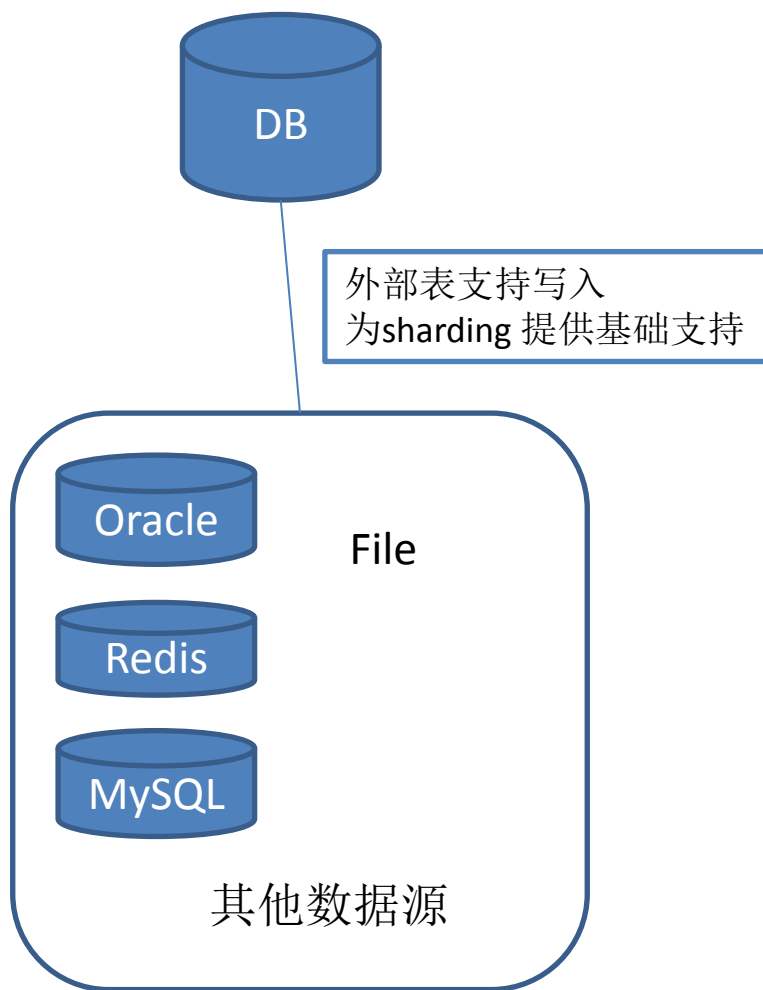
9.3

- 物化视图
- 简单视图
支持DML
- 可写FDW
接口
- 事件触发器

物化视图



可写FDW



事件触发器

- 保护DBA，防止DBA误操作



当drop或truncate table时

触发function1

function1中，根据触发事件的用户，删除或truncate重要的数据时
抛出异常

保护DBA，防止DBA误操作

里程碑

9.4

- JSONB
- 增量物化视图
- 逻辑流式复制
- 缓存预热
- 正则查询支持索引

JSONB

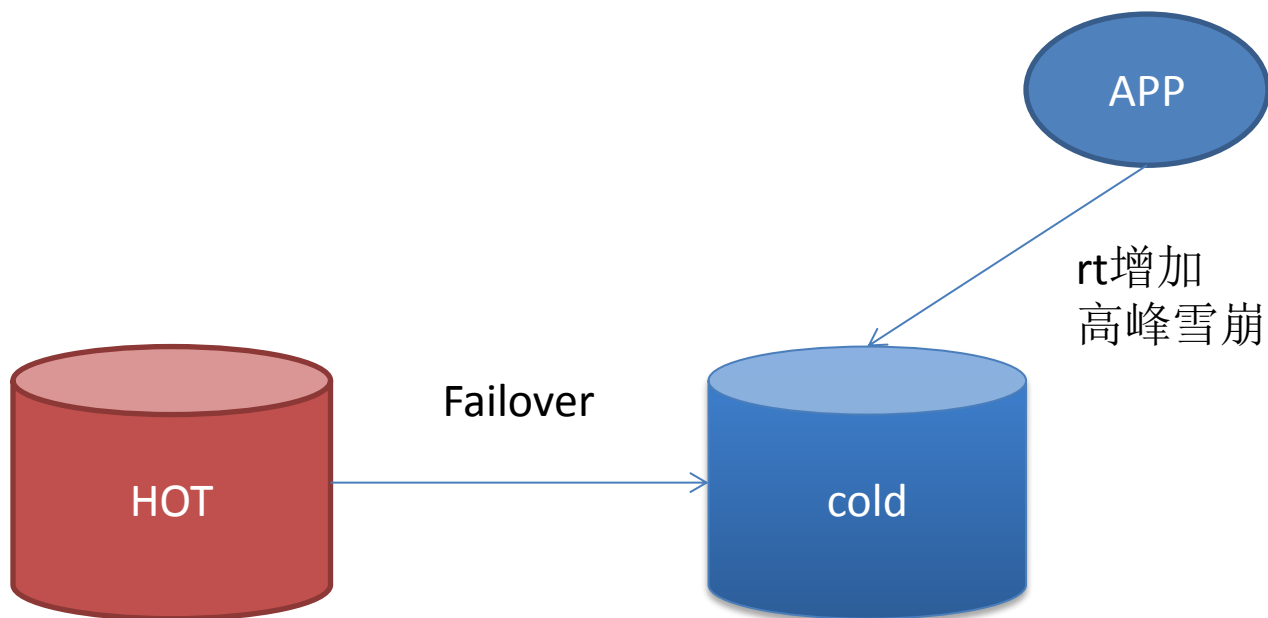
- 全面支持
 - IO、OP、AM、func
- 任意KEY或VALUE的索引支持
- VALUE支持numeric, string, time, array, 嵌套json等
- 支持常用的JSON类型查询，构造符

多Master复制

- 基于redo的逻辑复制
- 数据冲突handler
- 自定义冲突handler

防止雪崩

- 缓存预热
- set query timeout



高效(秒、毫秒级)模糊检索和分词

- 英语分词性能: ~ 900万 words每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
- 中文分词性能: ~ 400万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
- 英文分词+插入性能: ~ 666万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
- 中文分词+插入性能: ~ 290万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)

- 高效(秒、毫秒级) 模糊查询、正则匹配 (支持中文, 支持GIN索引查询)
- 传统方式 全表扫描, 百亿数据的查询响应至少是小时级别。
- PostgreSQL使用GIN R-TREE索引可以将查询时间缩短到秒级。

— 公安刑侦、车牌、地址、邮箱。。。查询、

```
select 'postregsql' % 'postgresql';
      postgres=# select similarity('postregsql','postgresql');
      similarity
      -----
             0.375
      (1 row)
```

```
select * from tbl where info ~ '^???6888$';
select * from tbl where info ~ '^???688?$';
```

性能与ElasticSearch
差不多

前后遮挡的才是高手
但是也逃不过
PostgreSQL的法眼



里程碑

9.5

- BRIN索引接口
- 多核性能增强
- OLAP增强
- 多维分析
- customSCAN接口
- 采样接口
- GPU并行加速器

流式数据 - 块级索引 BRIN

- 支持适合流式数据的索引
 - 某电子商务网站的用户浏览行为记录
 - 每分钟70万条记录(一天约10亿记录)
 - 每10分钟统计一次最近的行为并合并结果(范围扫描), 15个统计维度



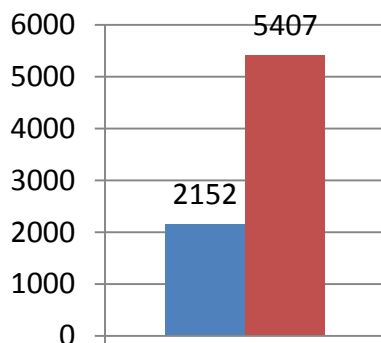
BRIN索引 和 B-tree索引的对比

- btree (全记录索引)

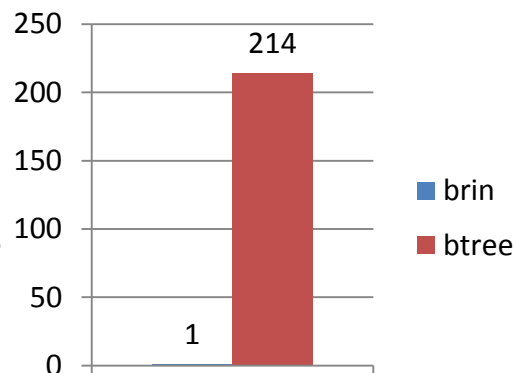
- value1,ctid
- value2,ctid
-
- **big**
- full

B-tree
大而全

BRIN
小而美



插入数据(ms)

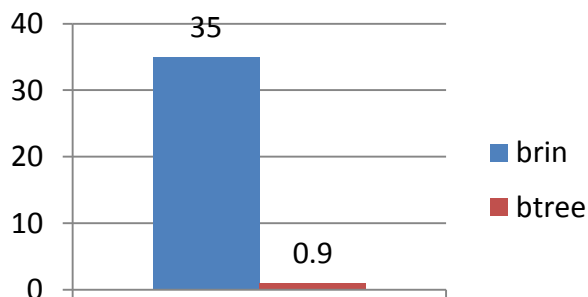


索引大小(MB)

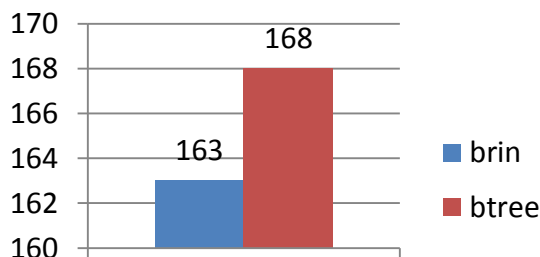
- brin (块粒度索引)

- blockid_left, blockid_right, value_min, value_max, allnull?, hasnull?
-
- **small**
- lossy

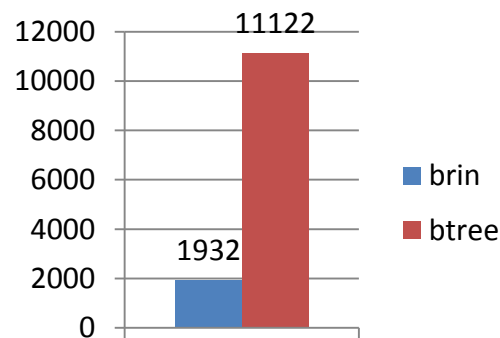
流式数据
范围
增量计算



精确查询(ms)



范围查询(ms)
超过单个BLOCK



创建索引(ms)

GIS数据结合窗口、多维分析

解决了哪几个问题

1. 避免冗余扫描和计算
 1. 每个大范围的数据只需要扫一次
2. 解决灵活多变的多维透视需求

多个字段任意组合，指定组合的聚合
查询结果快速查询

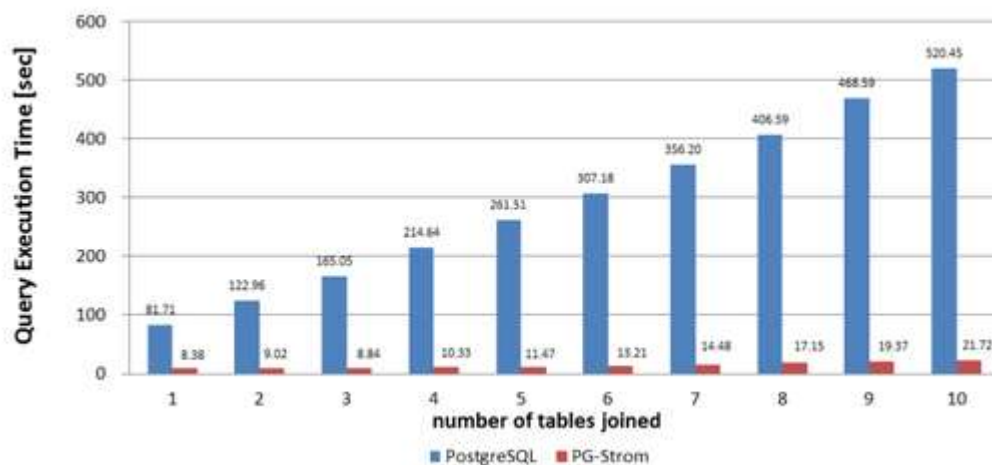
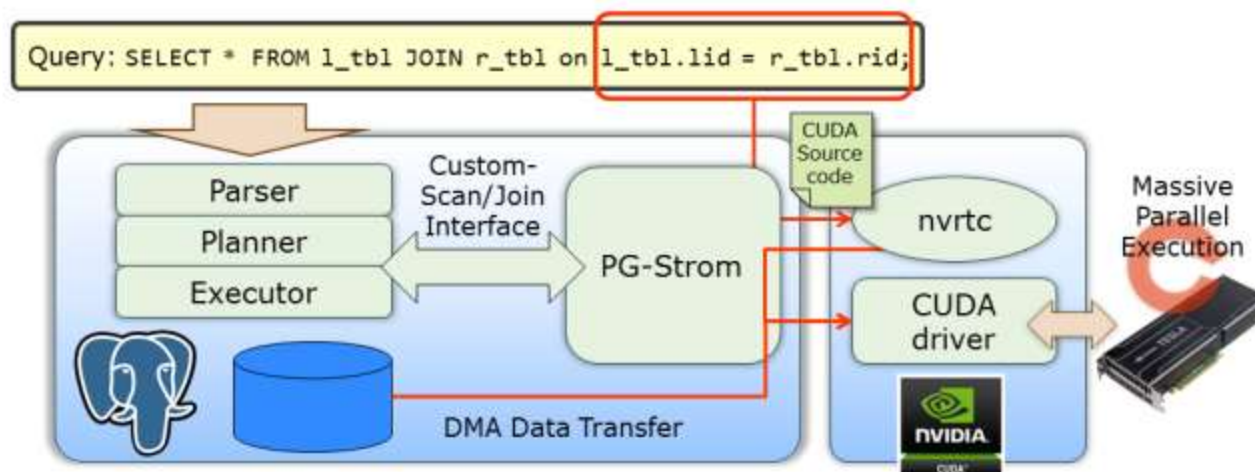
Grouping Sets

Cube

Rollup



GPU并行计算



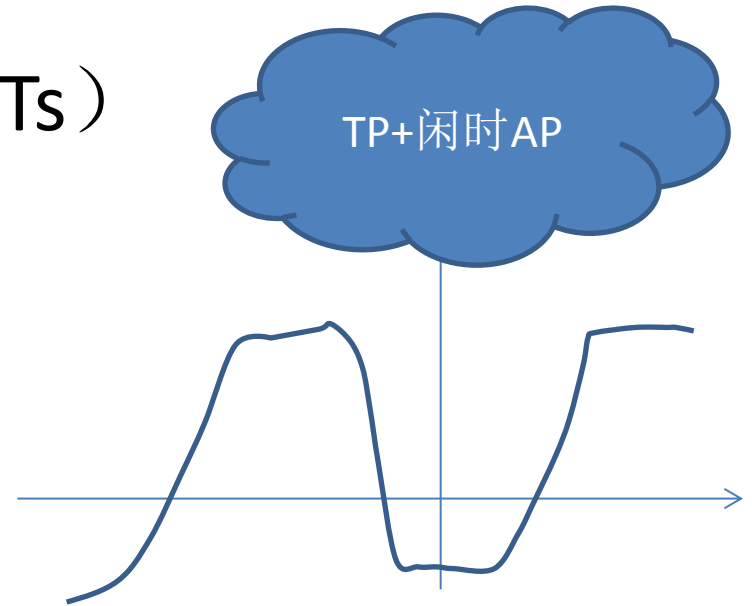
里程碑

9.6

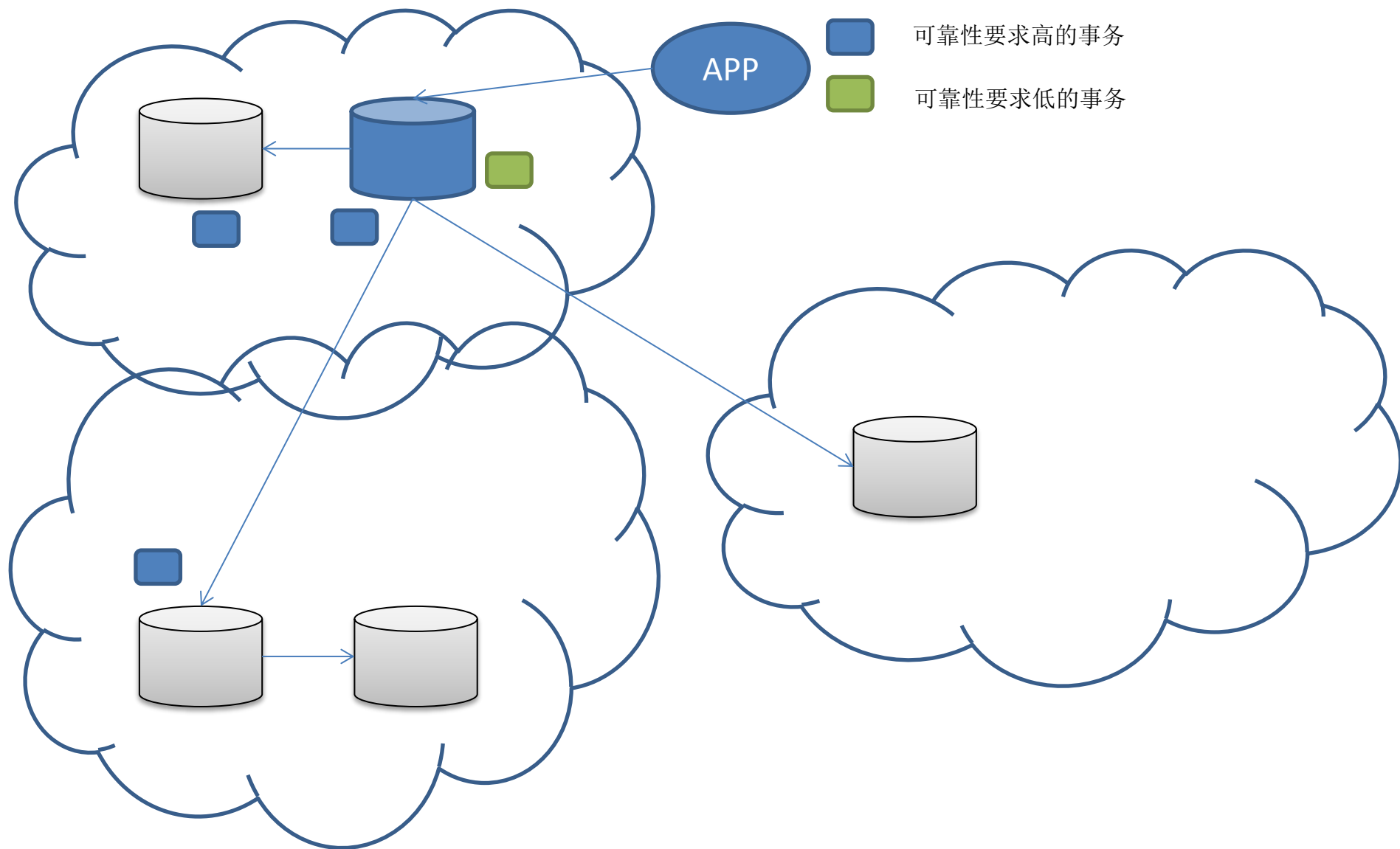
- 多核并行计算
- 多副本同步流式复制
- snapshot too old
- FDW join, sort, update, delete 下推
- 多核增强 (500W+) QPS

并行计算 - 精准营销

- 用户画像数据（标签 - BITS）
- 数据量亿级
- 用户提交查询条件(BITs)
- bit and (包含什么bit)
- bit xor (不包含什么bit)
- 性能提升与CPU成倍数提升(几百秒 -> 几秒)



金融级 - 多副本可靠性



前端 - 任意维度勾选

- 任意维度组合
- bloom index

实例名

Select...

Q

高级搜索

实例名:

用户连接:

用户信息:

可用区:

请选择

集群名:

请选择

库类型:

请选择

库版本:

请选择

实例类型:

请选择

网络类型:

请选择

VIP类型:

请选择

TOP类型:

请选择

实例状态:

请选择

锁定模式:

请选择

服务状态:

请选择

是否非标:

请选择

SQL WALL:

请选择

业务类型:

请选择

实例角色:

请选择

网络MODE:

请选择

过期时间:

至

创建时间:

至

搜索

里程碑

其他插件

GIS业务 - 最佳路径计算

Routing Functions

- **All pairs** - All pair of vertices.
 - **pgr_floydWarshall** - Floyd-Warshall's Algorithm
 - **pgr_johnson** - Johnson's Algorithm
- **pgr_astar** - Shortest Path A*
- **pgr_bdAstar** - Bi-directional A* Shortest Path
- **pgr_bdDijkstra** - Bi-directional Dijkstra Shortest Path
- **dijkstra** - Dijkstra family functions
 - **pgr_dijkstra** - Dijkstra's shortest path algorithm.
 - **pgr_dijkstraCost** - Use pgr_dijkstra to calculate the costs of the shortest paths.
- **Driving Distance** - Driving Distance
 - **pgr_drivingDistance** - Driving Distance
 - Post processing
 - **pgr_alphaShape** - Alpha shape computation
 - **pgr_pointsAsPolygon** - Polygon around set of points
- **pgr_ksp** - K-Shortest Path
- **pgr_trsp** - Turn Restriction Shortest Path (TRSP)
- **pgr_tsp** - Traveling Sales Person



支持线段双向权重设定

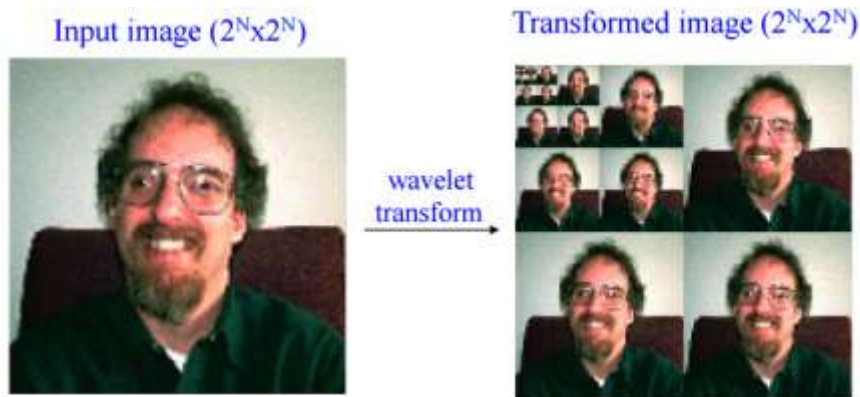
上坡: + weight

下坡: - weight



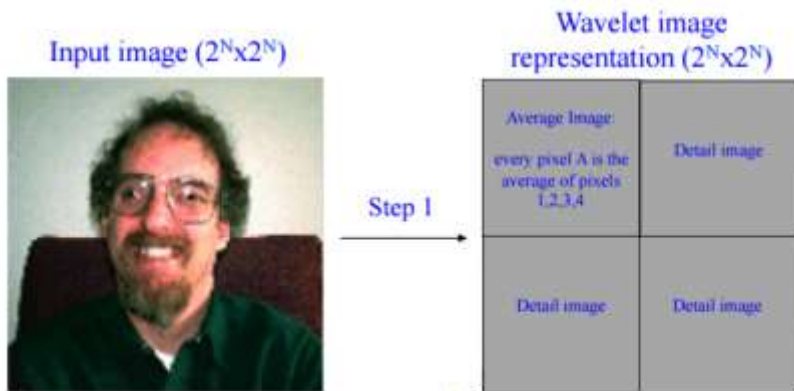
图像搜索

Wavelet算法
下沉至PG内核



The Haar 2-D Wavelet Transform

The 2-D Haar Wavelet Transform corresponds to a modification of this minimal recursive transform

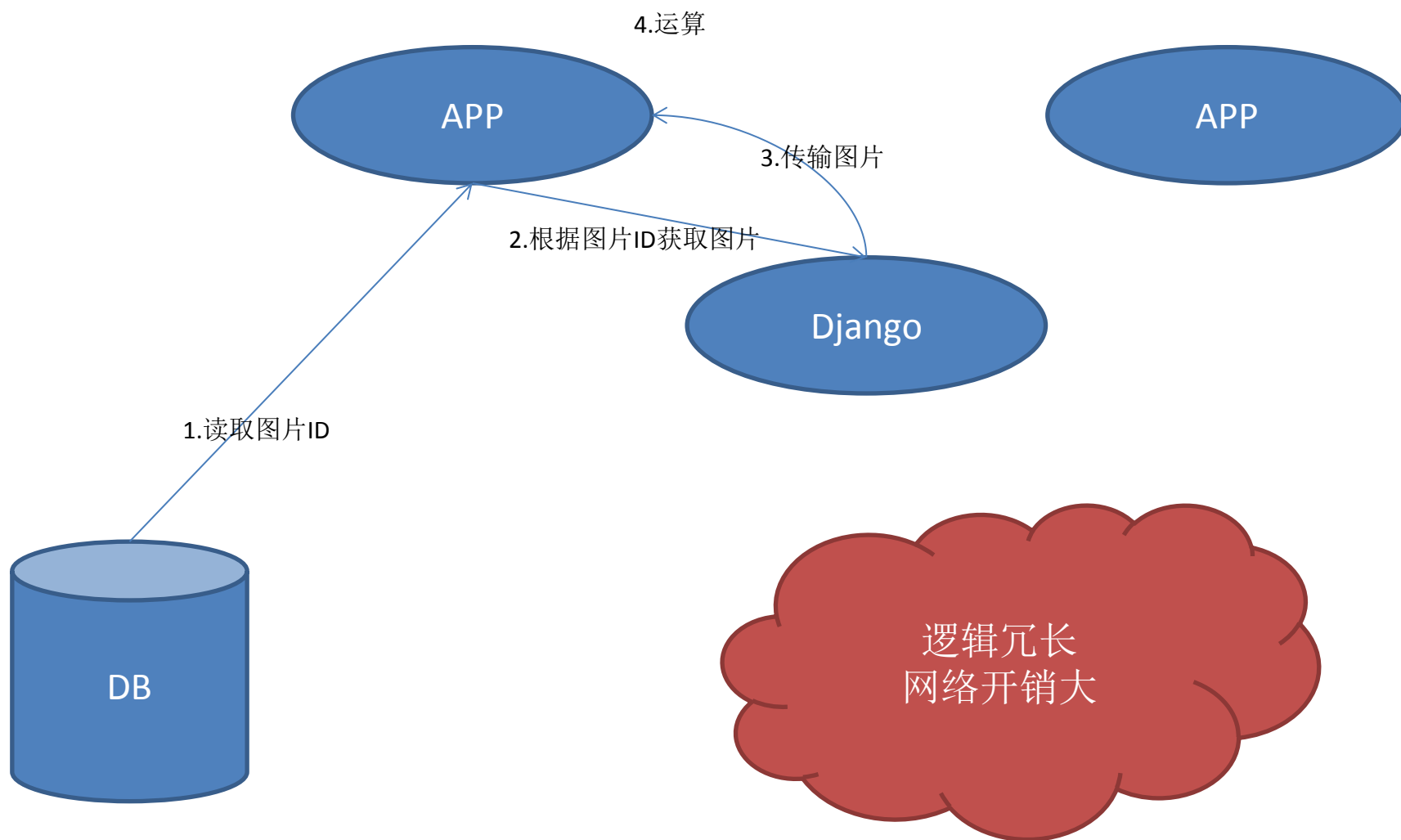


相似图片 | 相似人脸

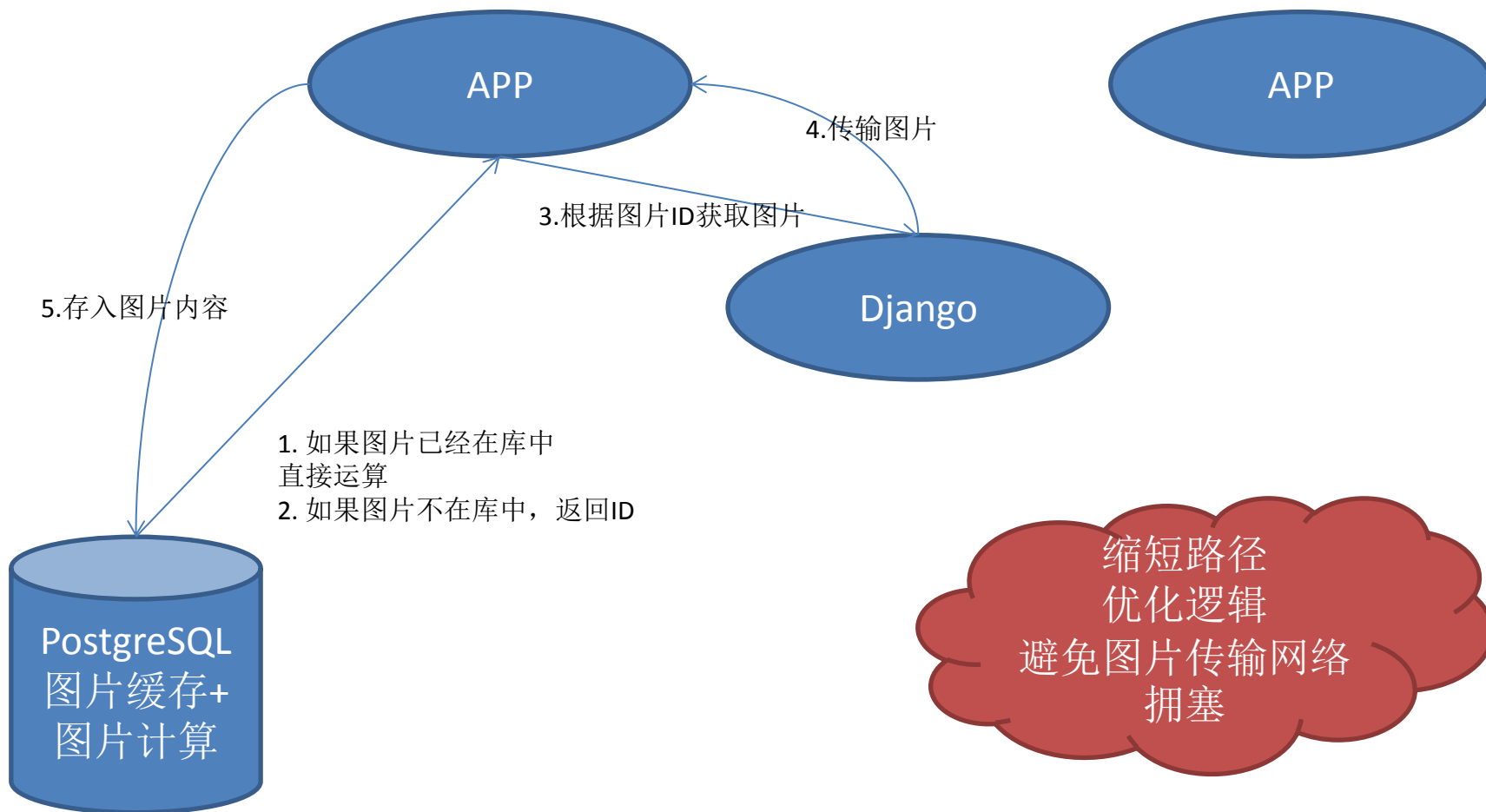
更多 >>



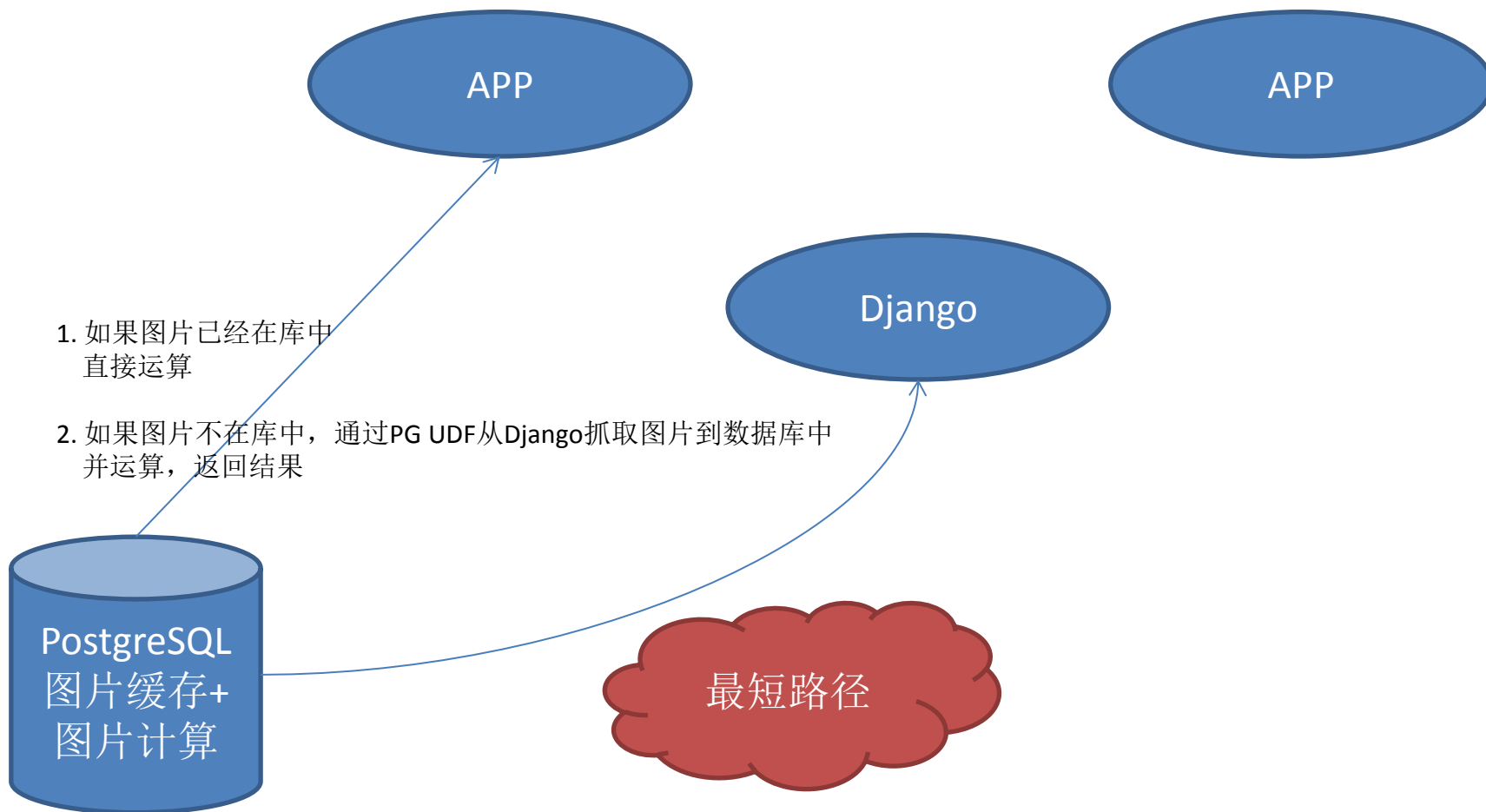
图像搜索



图像搜索



图像搜索



基因测序

- DNA库，如何配对，下一代最优

New data types:

- DNA_SEQUENCE
- RNA_SEQUENCE
- AA_SEQUENCE
- ALIGNED_DNA_SEQUENCE
- ALIGNED_RNA_SEQUENCE
- ALIGNED_AA_SEQUENCE
- Type modifiers:
 - CASE_SENSITIVE / CASE_INSENSITIVE
 - FLC / IUPAC / ASCII
 - SHORT / DEFAULT / REFERENCE (only DNA)



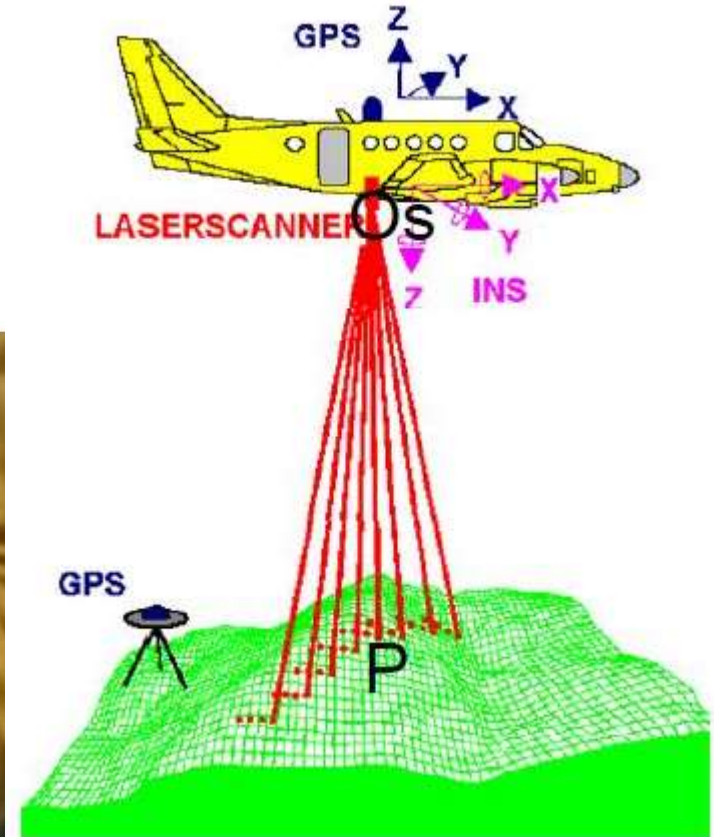
3D数据处理(raster,pgpointcloud)

位置、RGB、密度、材质、。。。位点多元属性



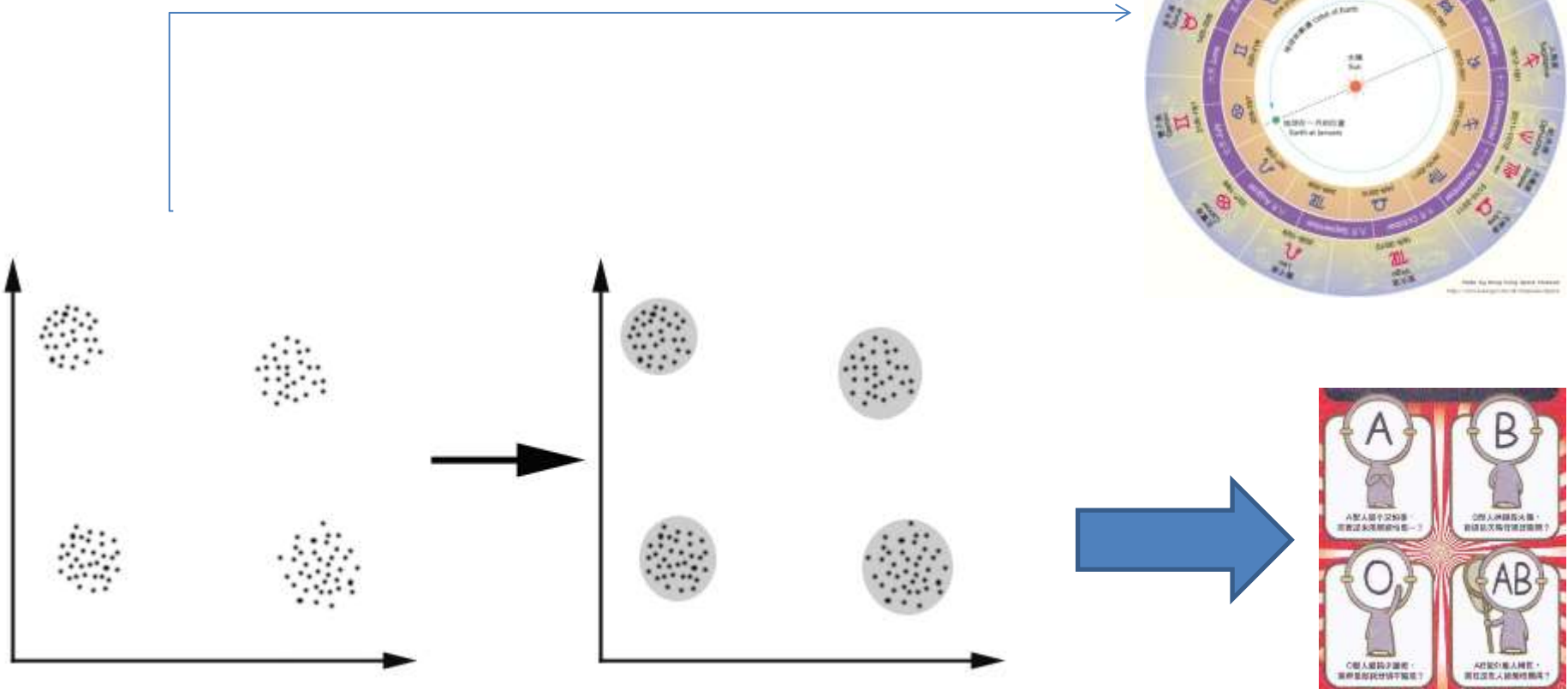
3D打印是近几年兴起的一种技术，除了存储物体表面的位置信息，还有颜色，密度等信息。

而3D扫描其实在军用领域很早以前就有了。



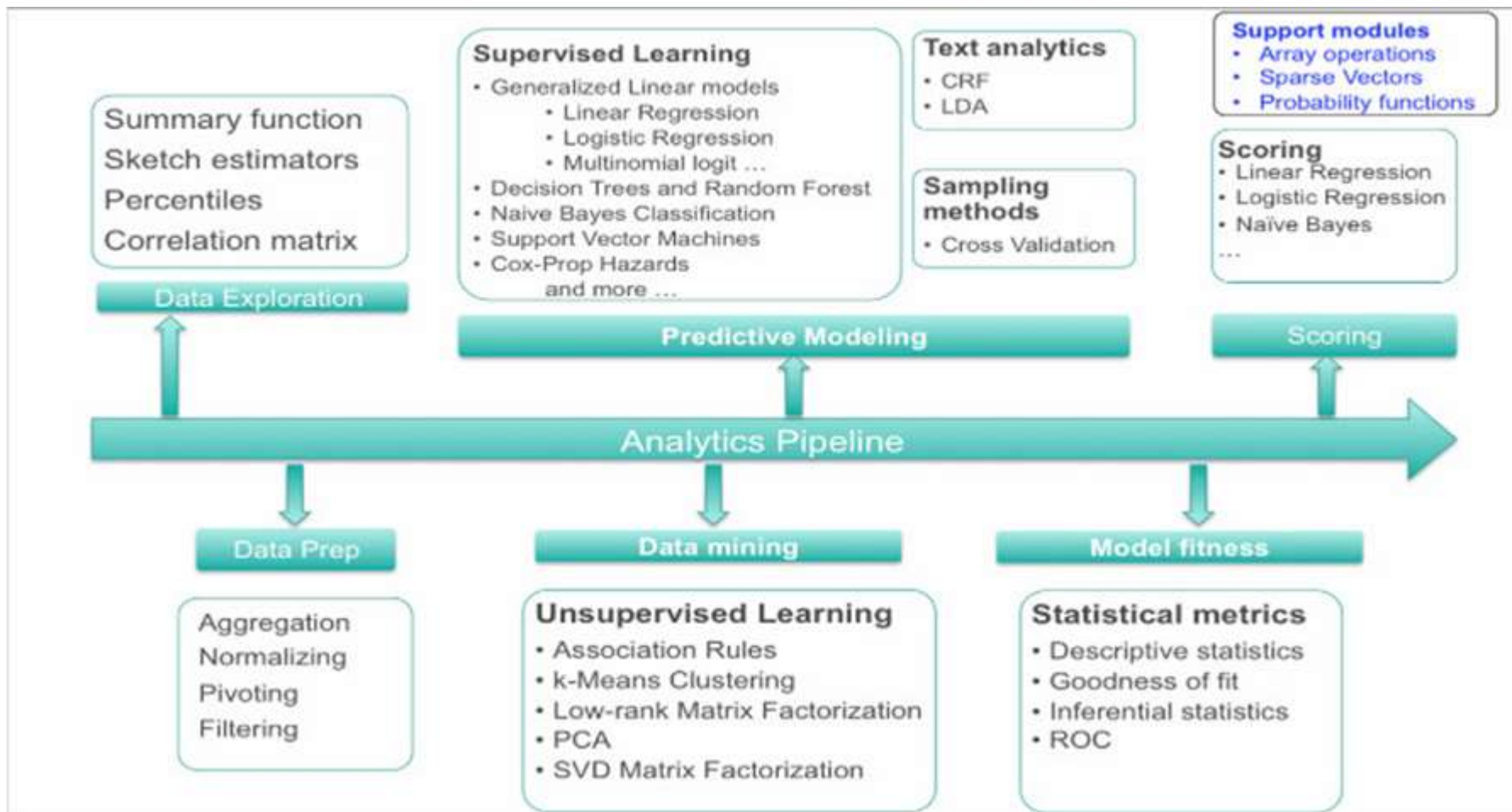
一条SQL搞定聚类分析

- 一条SQL搞定聚类分析
- `SELECT kmeans(ARRAY[columns,...], K) OVER (...), * FROM samples;`



机器学习UDF库

- SQL接口机器学习库MADLib (支持几百个机器学习库函数、对应各种数学模型), PL/R, PL/Python



线性回归例子

- MADLib库(支持**几百**个机器学习库函数、对应**各种数学模型**)，PL/R, PL/Python
- 例子
 - p元线性回归
 - $y_1 = b_0 + b_1x_{11} + b_2x_{12} + \dots + b_px_{1p} + \epsilon_1$
 - $y_2 = b_0 + b_1x_{21} + b_2x_{22} + \dots + b_px_{2p} + \epsilon_2$
 -
 - 求截距，斜率。
 - 预测 y_n
 - $y_n = b_0 + b_1x_{n1} + b_2x_{n2} + \dots + b_px_{np} + \epsilon_n$

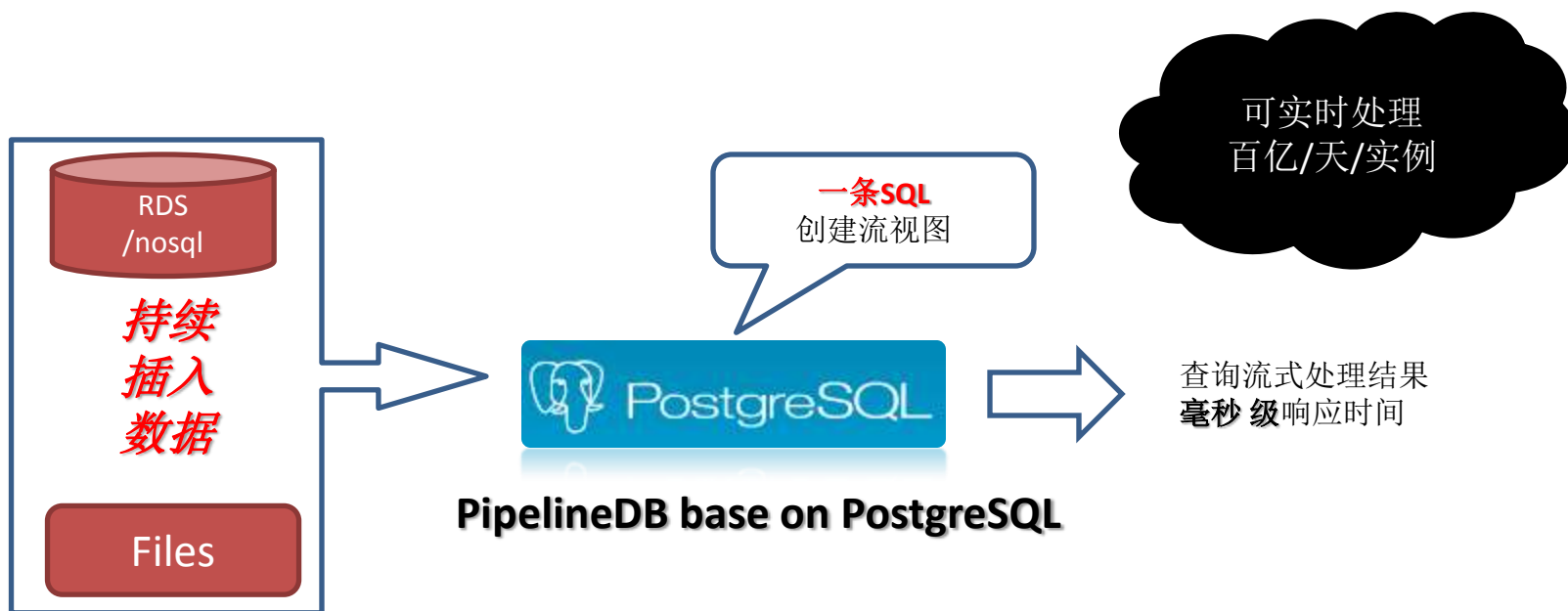
股市有风险
入市需谨慎

```
linregr_train( source_table,  
               out_table,  
               dependent_varname,  
               independent_varname,  
               grouping_cols,  
               heteroskedasticity_option  
)
```



一条SQL搞定流式实时处理

- 传统流式计算开发门槛高
- PostgreSQL 一条SQL搞定流式实时处理
 - 实时计算某WEB站点的请求延迟(**SLA实时监测**), 90%RT低于多少毫秒, 95%RT低于多少毫秒, 99%RT低于多少毫秒。
 - **实时营销**效果反馈 (实时检测某营销活动周边1公里的人流、车流)
 - 实时趋势**预测** (如股价, 温度, 湿度。。。)
 - 非流式处理的话, 响应时间在**十分钟级** (假设**十亿级**数据量)



超轻锁- 秒杀特性

- 超轻锁 (advisory LOCK) 解决高并发锁竞争问题
 - 手段：在CPU运算发现行锁之前就知道是不是有冲突，大大缩短CPU计算资源，等待资源

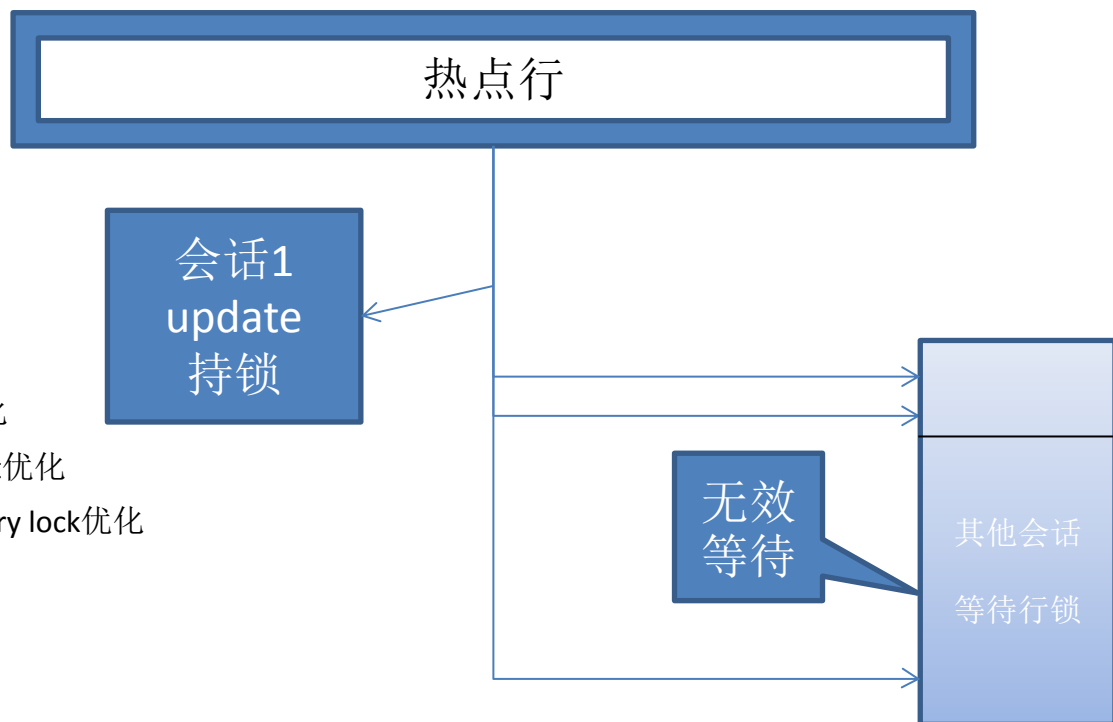
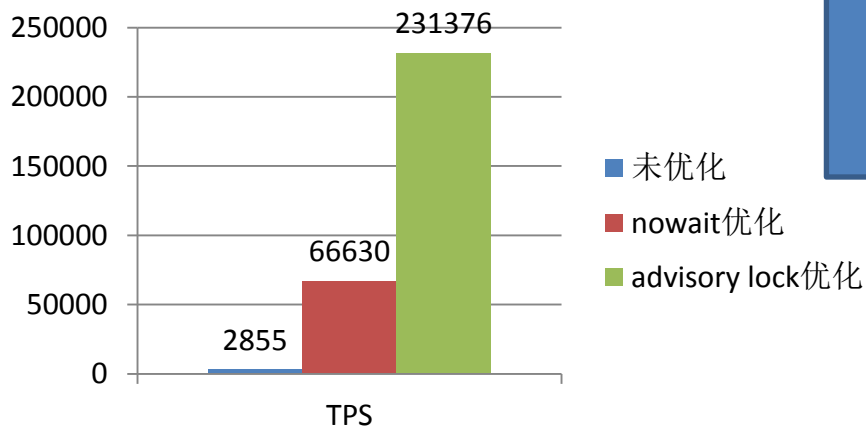
传统 - 行锁弊端

1. 无效等待多

2. 无效等待用户

长时间占用会话资源

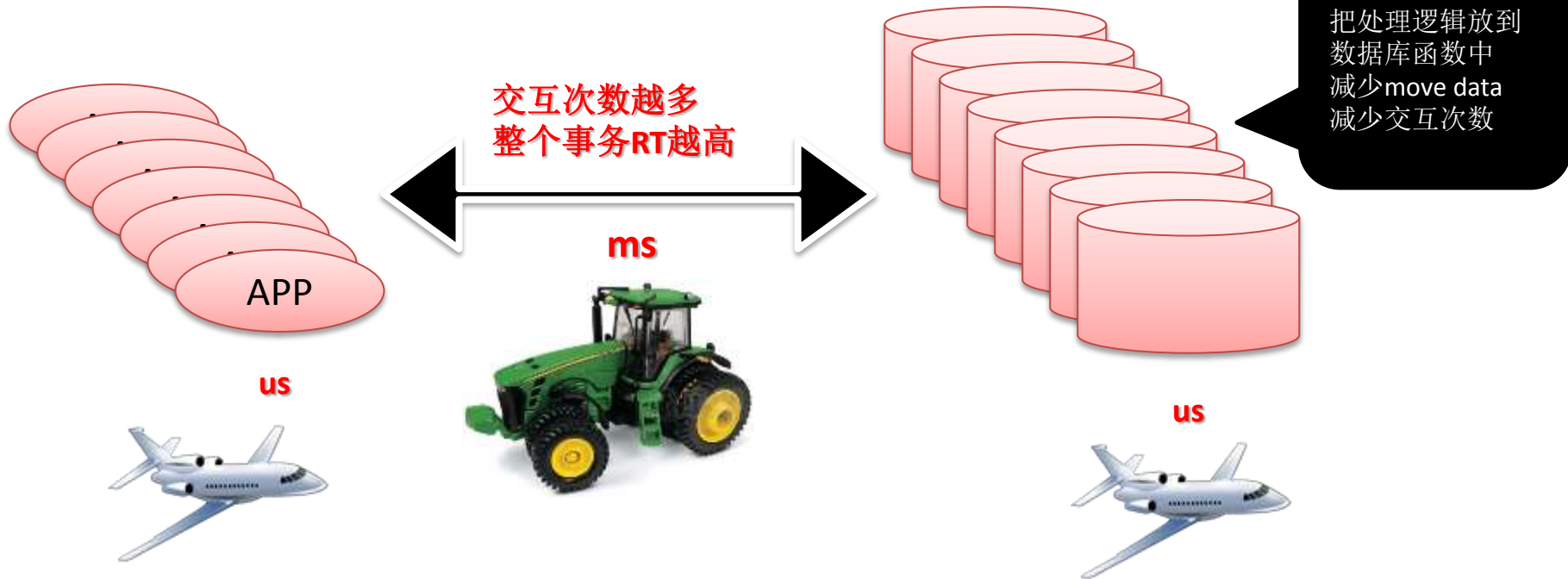
3. 发现锁冲突的代码路径长
需要进行大量CPU运算



数据库编程能力

- 数据库端函数编程
 - 支持 C, Python, R, perl, java, tcl, PHP,
- 解决move data带来的网络RT瓶颈

硬件发展迅速
sharding后数据库IO、运算能力
已经不再是瓶颈



数据库编程能力

TPC-C

新建订单 9 QUERY

支付 11 QUERY

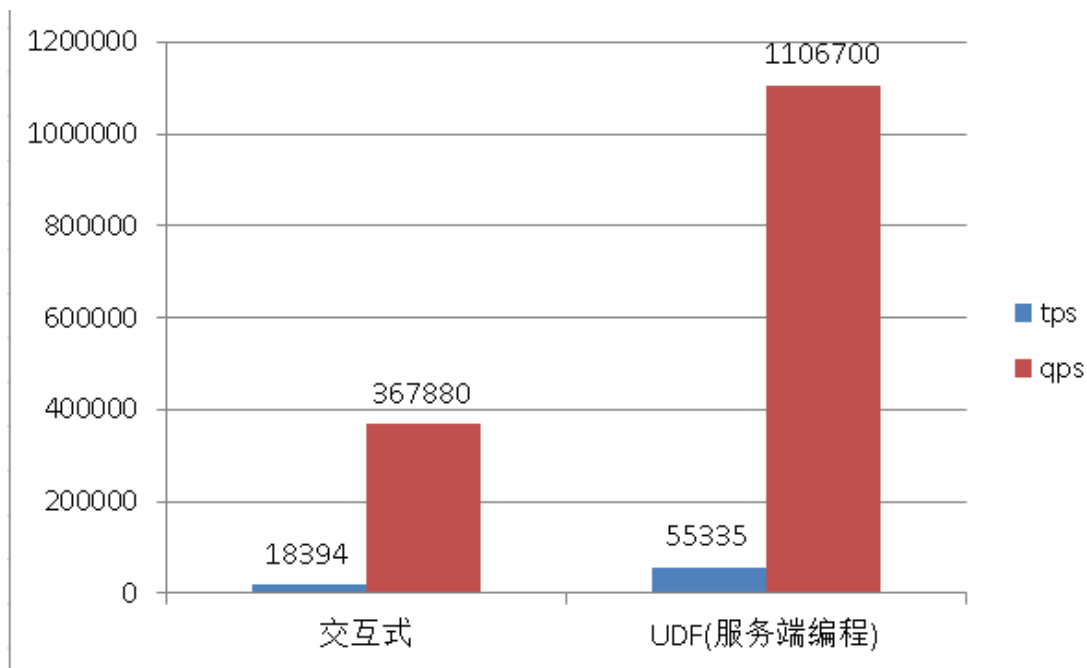
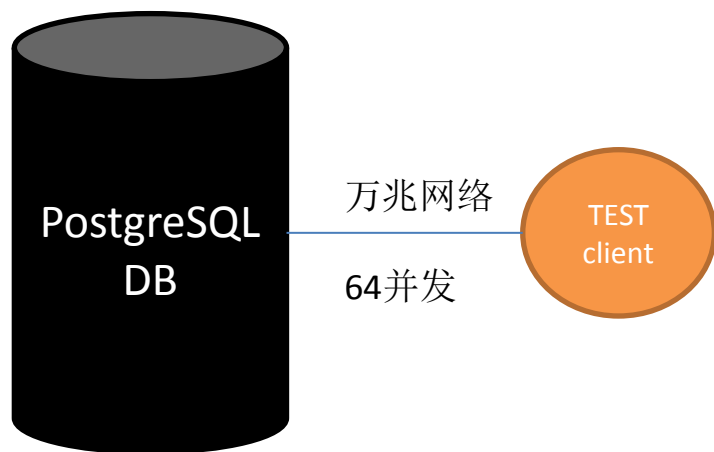
订单状态查询 6 QUERY

发货 7 QUERY

对比20条QUERY(pk I,U,D,S)的事务TPS

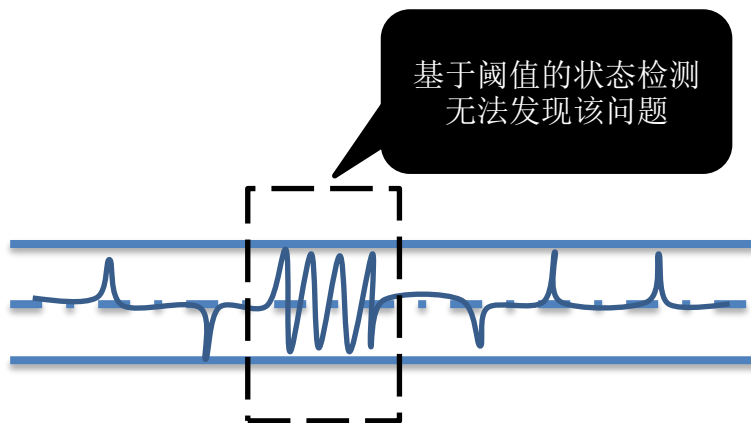
1.交互式（业务逻辑在client完成）

2.服务端编程模式（业务逻辑封装在数据库FUNC完成）



复杂查询

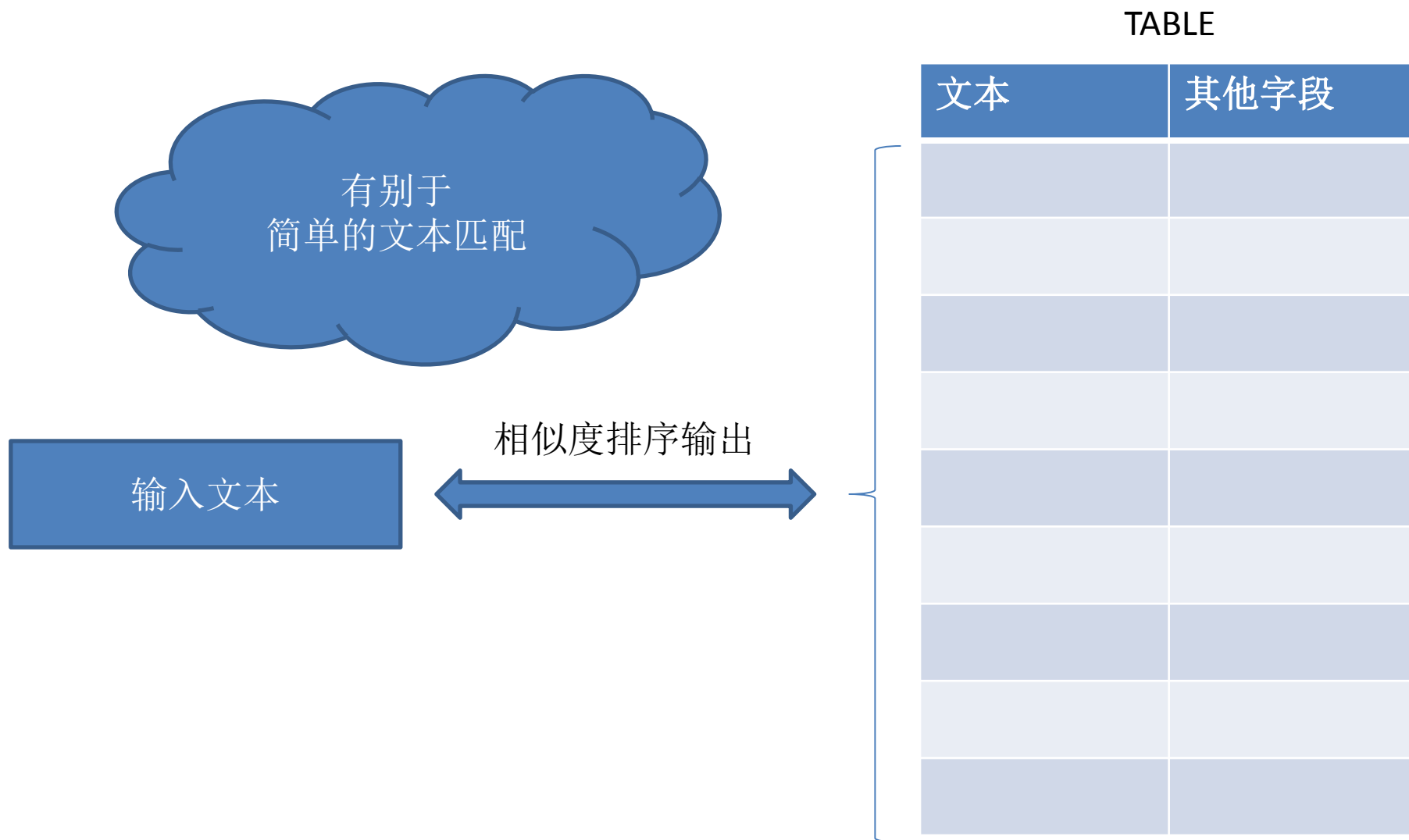
- 遗传算法、HASH JOIN、HASH 聚合
 - 解决多表查询效率问题，在分析场景中比传统嵌套循环性能提升100倍以上
- **传统**状态监测，基于阈值或**状态**值
 - 适合离散值
 - 覆盖不到的**问题**：抖动、趋势异常。
- **多维度复杂查询**
- 抖动监测，基于**方差**
 - 适合连续值
- 趋势监测，基于**相关性**
 - 例如 时间相关性、属性相关性



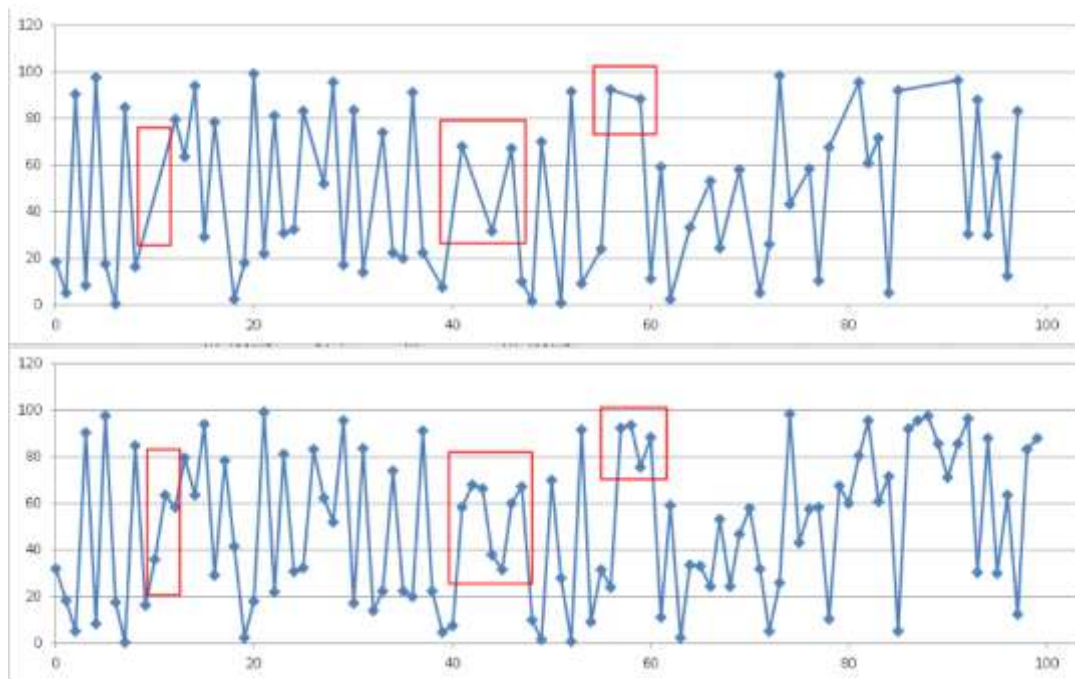
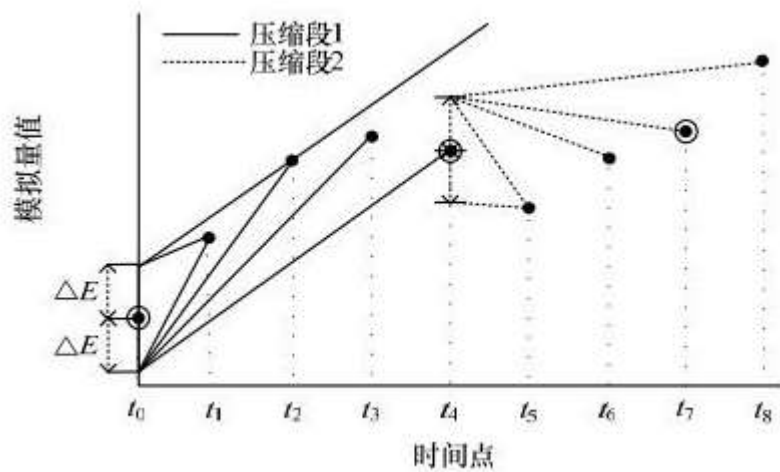
估值计算

- HLL
- 按时间统计UV、新增用户数，。。。。。。

文本挖掘- 相似度



物联网 - 旋转门压缩



议题

- PostgreSQL 前世今生
- **PostgreSQL 社区理念**
- PostgreSQL 适应的场景
- 最佳实践
- 阿里云PostgreSQL内核优化

JSON

HINT

GIS数据处理

多维分析

FDW数据泵

时序数据处理

异步消息

音频处理

立体几何

平面几何

基因处理

图数据处理

分词

旋转门压缩

数据走势预测

正则走索引

下棋

估值计算

探索宇宙

3D打印

GPU并行计算

流式计算

机器学习

文本挖掘

对接ES\Kafka\...

MR

LBS应用

一切皆可扩展 - 取自最终用户开源项目(pgxn, github, pgfoundry, sourceforge, 社区.....)



历经43年进化 - 相当成熟的底盘技术

开放式接口

SQL兼容性

AGG\WINDOW

HOOK language handler

类型扩展(IO,OP,AM,FUNC)

FDW handler scan handler

支持数十种流行编程语言
编写服务端函数

SQL-2013 递归查询 MVCC 优化器 8种索引方法,20种大类

ECPG 窗口查询 HASH JOIN 秒杀 bt,hash,gin,brin,rum
bloom,gist,sp-gist

可靠性

扩展性

REDO

流式复制

多副本

块级增量备份&PITR

CPU并行计算

读写分离

水平分库

多主同步

PostgreSQL - 可编程数据库

传统数据库

能存取固定类型
能用固定模式处理固定类型

传统数据库
眼里

这是别墅



可编程数据库

能存取固定类型
能用固定模式处理固定类型
能新增自定义类型
能新增自定义数据处理和访问方法

可编程数据库
眼里

这是别墅
同时允许扩展其用途
迪斯尼
农场
赌场
KTV
银行
超级市场
医院

。 。 。 。 。 。

继Ingres之后又一个改变世界的产品

- OLAP
 - Greenplum、AsterData、matrixDB、Paracle、Redshift、Illustra, Informix, Netezza、某些国产数据库。。。
- OLTP
 - EDB、国家电网xxDB、某些国产数据库
- 流式数据库
 - pipelineDB
- NewSQL
 - Postgres-XL, Postgres-XC, Postgres-XZ, 。。。



http://amturing.acm.org/stonebraker_1172121.pdf

http://amturing.acm.org/award_winners/stonebraker_1172121.cfm

http://amturing.acm.org/vp/stonebraker_1172121.cfm

单节点性能指标参考数据

- 秒杀
 - 8 Core, 23万 qps
- KNN近邻查询
 - 16 Core, 100亿数据, 64并发, KNN查询平均响应时间0.848毫秒, qps 74151.
- 模糊查询、正则匹配
 - 8Host, 16Core, 1008亿数据, 前后模糊、正则匹配, 秒级响应
- 分词
 - 英语分词性能: ~ 900万 words每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
 - 中文分词性能: ~ 400万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
 - 英文分词+插入性能: ~ 666万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
 - 中文分词+插入性能: ~ 290万 字每秒 (Intel(R) Xeon(R) CPU X7460 @ 2.66GHz)
- 并行计算
 - CPU并行 32Core, 16亿(90GB), count (*) 7秒, bit(and, xor) 16秒, 非并行(141秒, 488秒).
 - GPU并行 (1张 1亿 table join 9张 10万 table) 21秒, 非并行520秒.

单节点性能指标参考数据

- 数据装载
 - 32Core, 512G, 2*Aliflash SSD
 - 连续24小时多轮数据批量导入测试(平均每条记录长度360字节, 时间字段索引)
 - 每轮测试插入12TB数据
 - 506万行/s, 1.78 GB/s, 全天插入4372亿, 154TB数据
 - (为什么这么快?) (BRIN, HEAP, 动态扩展FILE, prealloc XLOG, reuse XLOG)
- TPC-B (1 Select : 3 Update : 1 Insert)
 - 32Core, 512G, 2*Aliflash SSD 10亿数据量, 11万tps, 77万qps
 - Select-Only 100万tps (即使应用缓存失效, 也无大碍)
- TPC-C (新建订单45,支付43,订单查询4,发货4,库存查询4)
 - 4000个仓库, 400GB数据, 平均每笔事务10几条SQL
 - 12Core, 256GB, intel SSD , 61万TPmC (IO瓶颈严重,理论上可以达到200万)
- LinkBench (Facebook 社交关系应用)
 - 1亿个node, 4亿条关系, (32Core, 2 SSD, 512G)
 - (添加NODE, 更新NODE, 删除NODE, 获取NODE信息, 添加关系, 删除关系, 更新关系, 关系总数查询, 获取多个关系, 获取关系列表)
 - 12万 ops (默认测试用例)

议题

- PostgreSQL 前世今生
- PostgreSQL 特性
- **PostgreSQL 适应的场景**
- 最佳实践
- 阿里云PostgreSQL内核优化

PostgreSQL适应场景

- 适应广泛的行业与业务场景
 - GIS, 物联网, 互联网, 企业, ERP, 多媒体,
- TP + simple AP
- 单库20TB不算多
- 要求主备严谨一致的场景不二之选

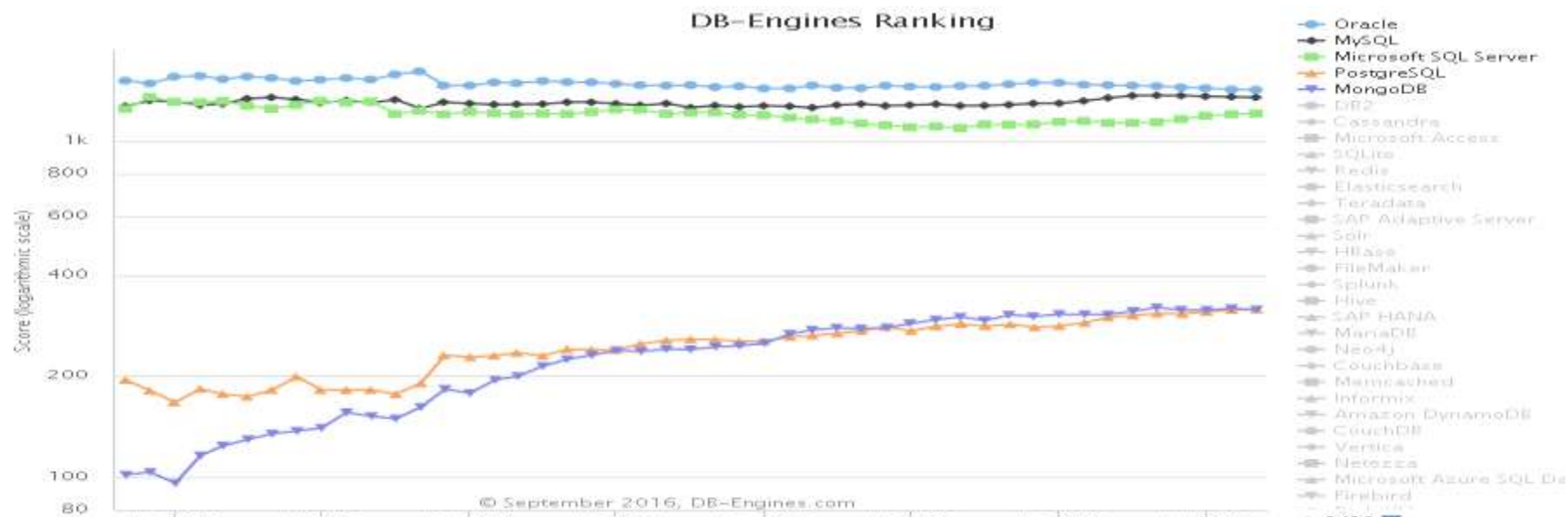
行业应用

- 生物制药 {Affymetrix(基因芯片), 美国化学协会, gene(结构生物学应用案例), ...}
- 电子商务 { CD BABY, etsy(与淘宝类似), whitepages, flightstats, Endpoint Corporation, 阿里巴巴 ...}
- 学校 {加州大学伯克利分校, 哈佛大学互联网与社会中心, .LRN, 莫斯科国立大学, 悉尼大学, 武汉大学, 人民大学, 上海交大, 华东师范 ...}
- 金融 {Journyx, LLC, trusecommerce(类似支付宝), 日本证券交易交所, 邮储银行, 同花顺, 平安科技...}
- 游戏 {MobyGames, 斯凯网络 ...}
- 政府 {美国国家气象局, 印度国家物理实验室, 联合国儿童基金, 美国疾病控制和预防中心, 美国国务院, 俄罗斯杜马, 国家电网, 12306...}
- 医疗 {calorieking, 开源电子病历项目, shannon医学中心, ...}
- 制造业 {Exoteric Networks, 丰田, 捷豹路虎}
- 媒体 {IMDB.com, 美国华盛顿邮报国会投票数据库, MacWorld, 绿色和平组织, ...}
- 开源项目 {Bricolage, Debian, FreshPorts, FLPR, PostGIS, SourceForge, OpenACS, Gforge, ...}
- 零售 {ADP, CTC, Safeway, Tsutaya, Rockport, ...}
- 科技 {Sony, MySpace, Yahoo, Afiliast, APPLE, 富士通, Omnitel, Red Hat, Sirius IT, SUN, 国际空间站, Instagram, Disqus, 去哪儿, 腾讯, 华为, 中兴, 云游, 智联招聘, 高德地图, 饿了么 ...}
- 通信 {Cisco, Juniper, NTT(日本电信), 德国电信, Optus, Skype, Tlestra(澳洲电讯), 中国移动...}
- 物流 {SF}

2010

2016

发展趋势



Jul 7, 1996 – May 7, 2016

Contributions to master, excluding merge commits

Contributions: **Commits** ▼



议题

- PostgreSQL 前世今生
- PostgreSQL 特性
- PostgreSQL 适应的场景
- 最佳实践
- 阿里云PostgreSQL内核优化

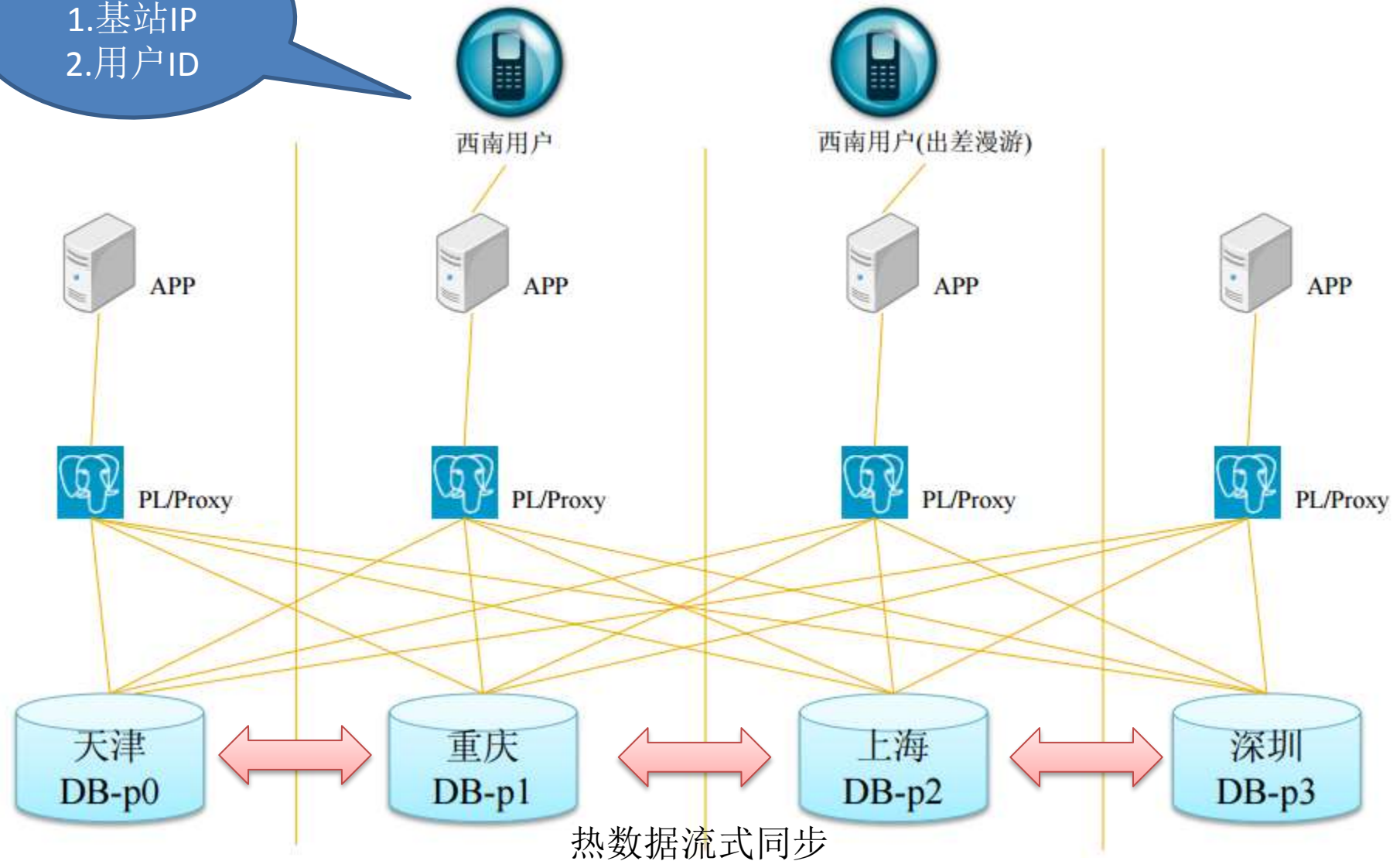
最佳实践

- 读写分离
 - 流式复制 + pgpool-II
 - 时延检测, HINT, FUNC黑白名单, 负载反馈, 心跳
- 水平分库
 - 内核支持 FDW based sharding, 下推, 应用透明, 支持任意操作(包括跨库事务, 跨库JOIN等)。
 - plproxy, 自由定义路由策略, 不限于哈希取模, 一致性哈希, 尾号等。
 - New SQL产品Postgres-XL, CitusDB, Postgres-XC
- 单元化
 - 多主逻辑复制(BDR)
 - 内置冲突处理handler, 开放冲突自定义handler接口

PL/Proxy 水平分库

路由策略

1. 基站IP
2. 用户ID



最佳实践

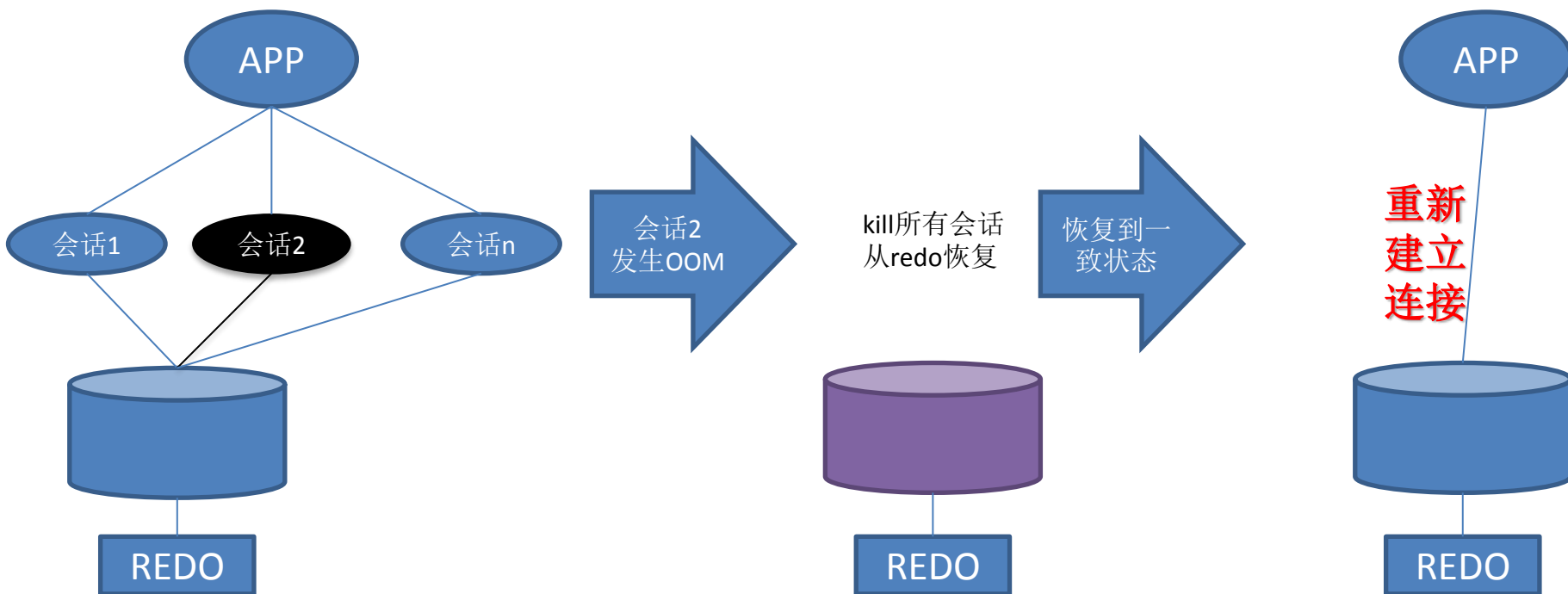
- 扩容、版本升降级
 - 逻辑增量复制、平滑扩容、升级
- 多副本高可用高可靠
 - 多机房，任意指定节点数同步流式数据复制
 - 选举(pg_raft)
- 备份恢复
 - 基于备库、块级增量，时间点恢复
 - rewind(强一致退化技术、跨时间线)
- 监控、诊断
 - Zabbix\Nagios\AWR

议题

- PostgreSQL 前世今生
- PostgreSQL 特性
- PostgreSQL 适应的场景
- 最佳实践
- 阿里云PostgreSQL内核优化

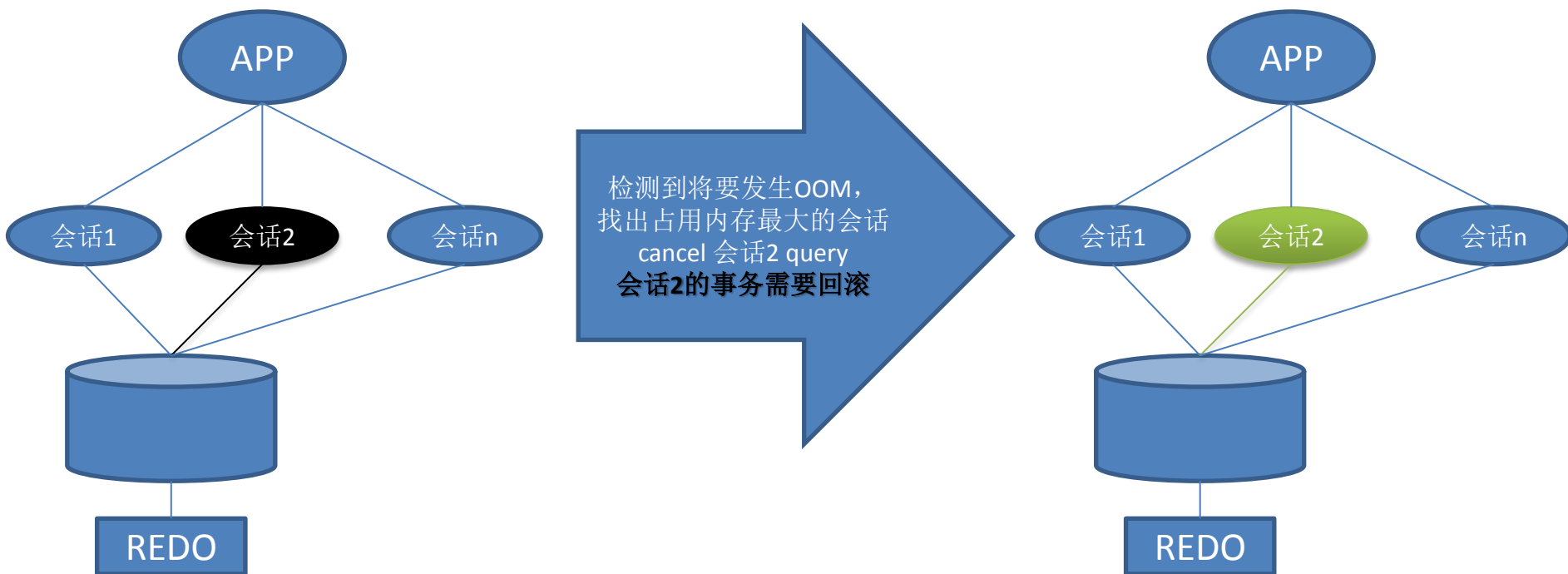
OOM 大面积影响业务问题分析

- 数据库发生OOM后
- 数据库crash，进入自动recovery状态
- 较大面积影响业务



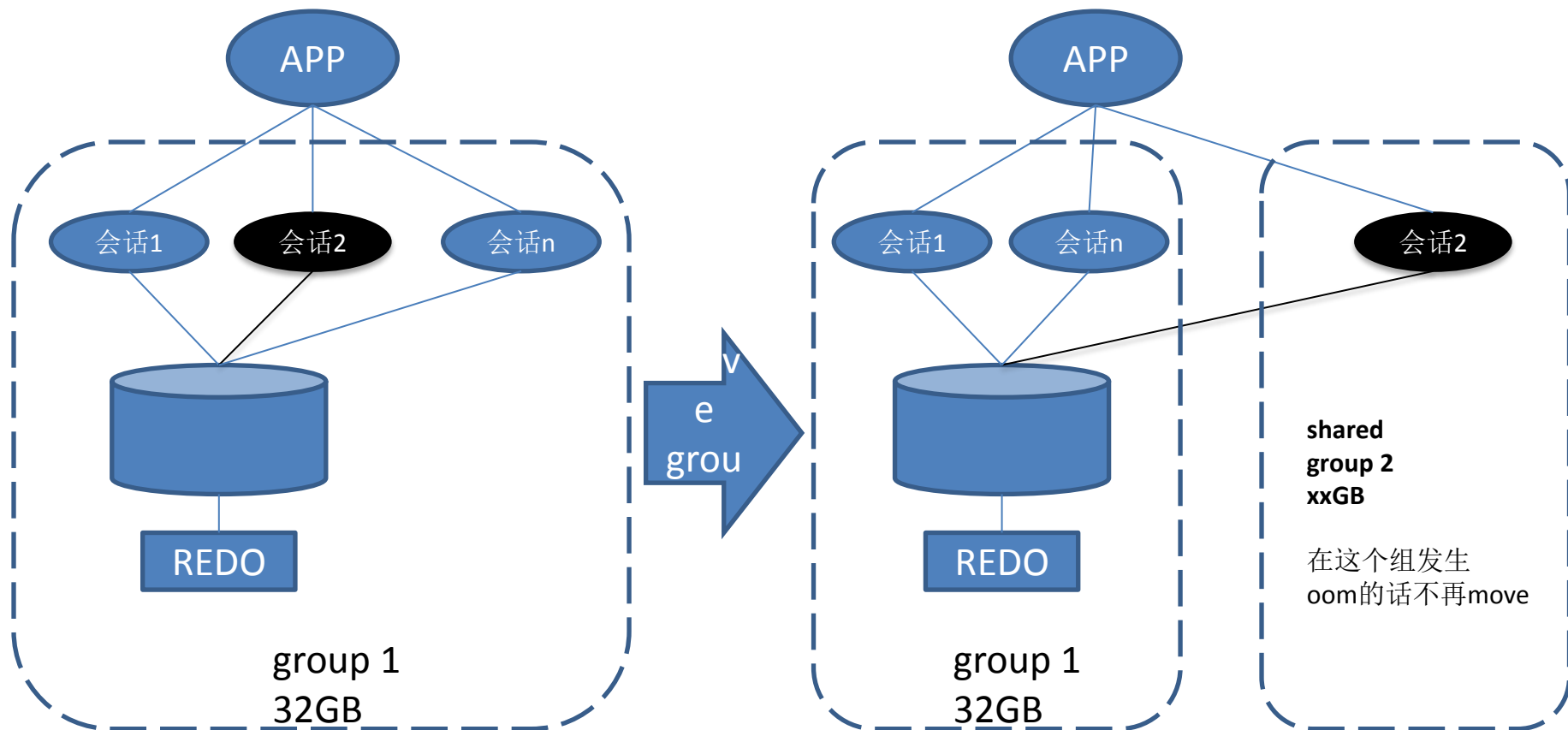
ApsaraDB PG解决OOM问题1

- 异步监控，发现快要OOM时，
- 使用USR2信号，cancel 较大内存的query。
- 不影响所有会话，仅仅影响占据较大内存的会话，需回滚。



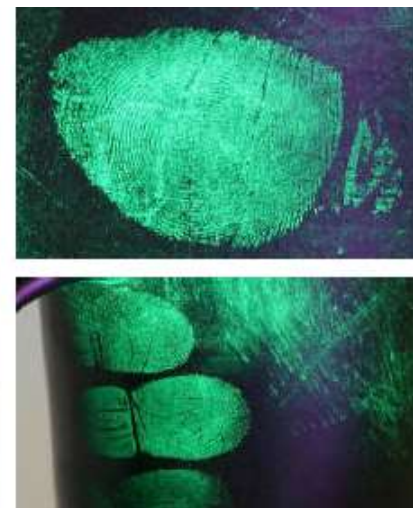
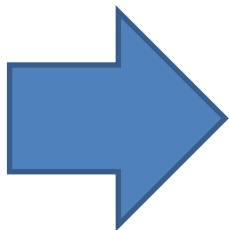
ApsaraDB PG解决OOM问题2

- 弹性OOM



security invoker 安全陷阱

- 函数提权陷阱

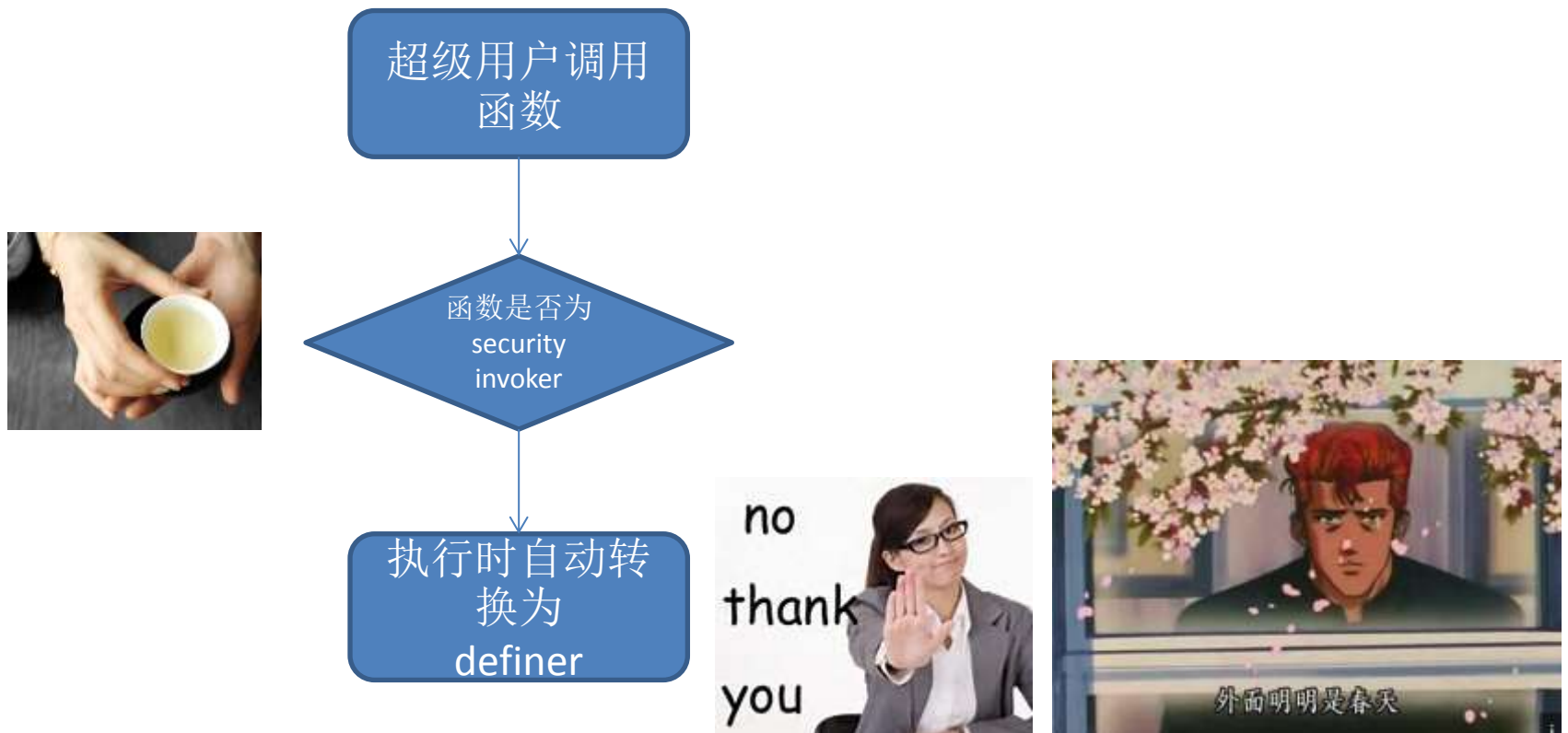


创建内含越权操作的函数
将函数设置为 security invoker

通过视图，触发器，规则等手段，
诱导超级用户或其他目标用户执行

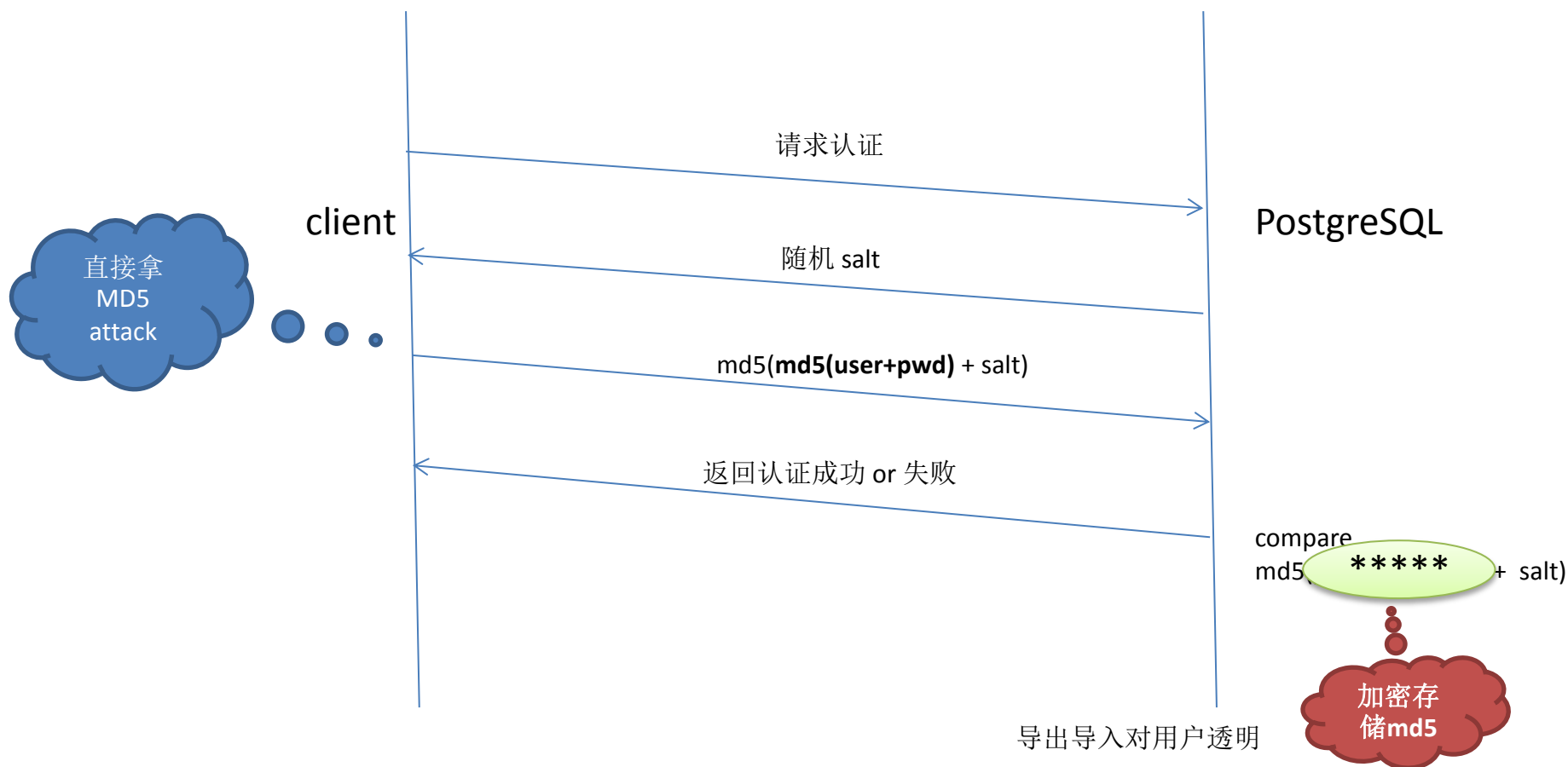
ApsaraDB PG **FIX** 提权安全陷阱

- 普通用户创建的security invoker函数，超级用户调用时自动转换为security definer

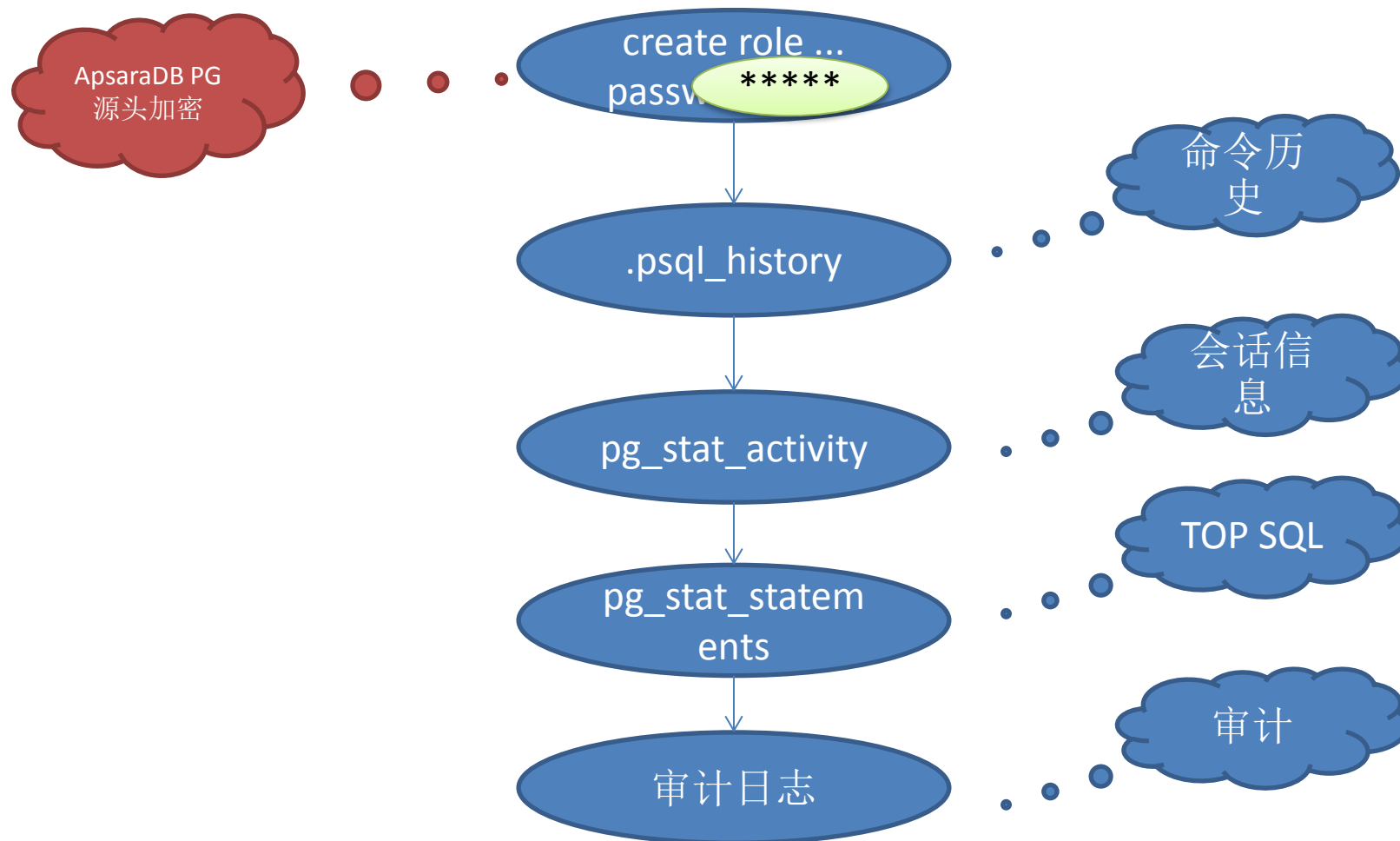


密码md5安全加固

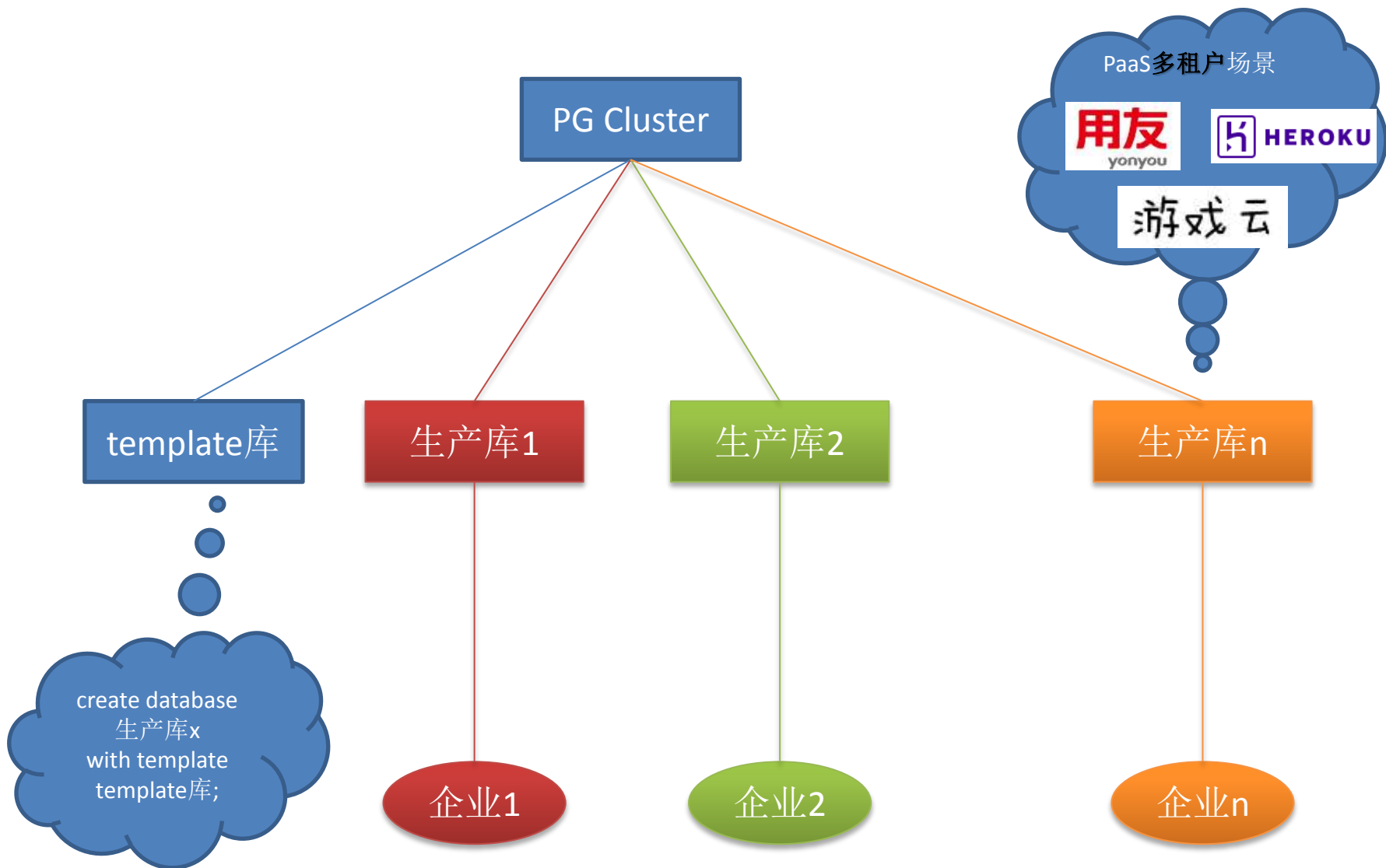
- 认证过程



封堵密码泄露漏洞



频繁建库业务背景

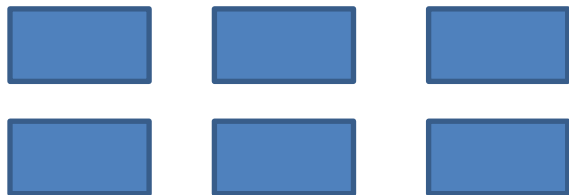


频繁建库瓶颈分析

模板库



一个基础模板库有几百个文件



create
database

copy_dir

生产库

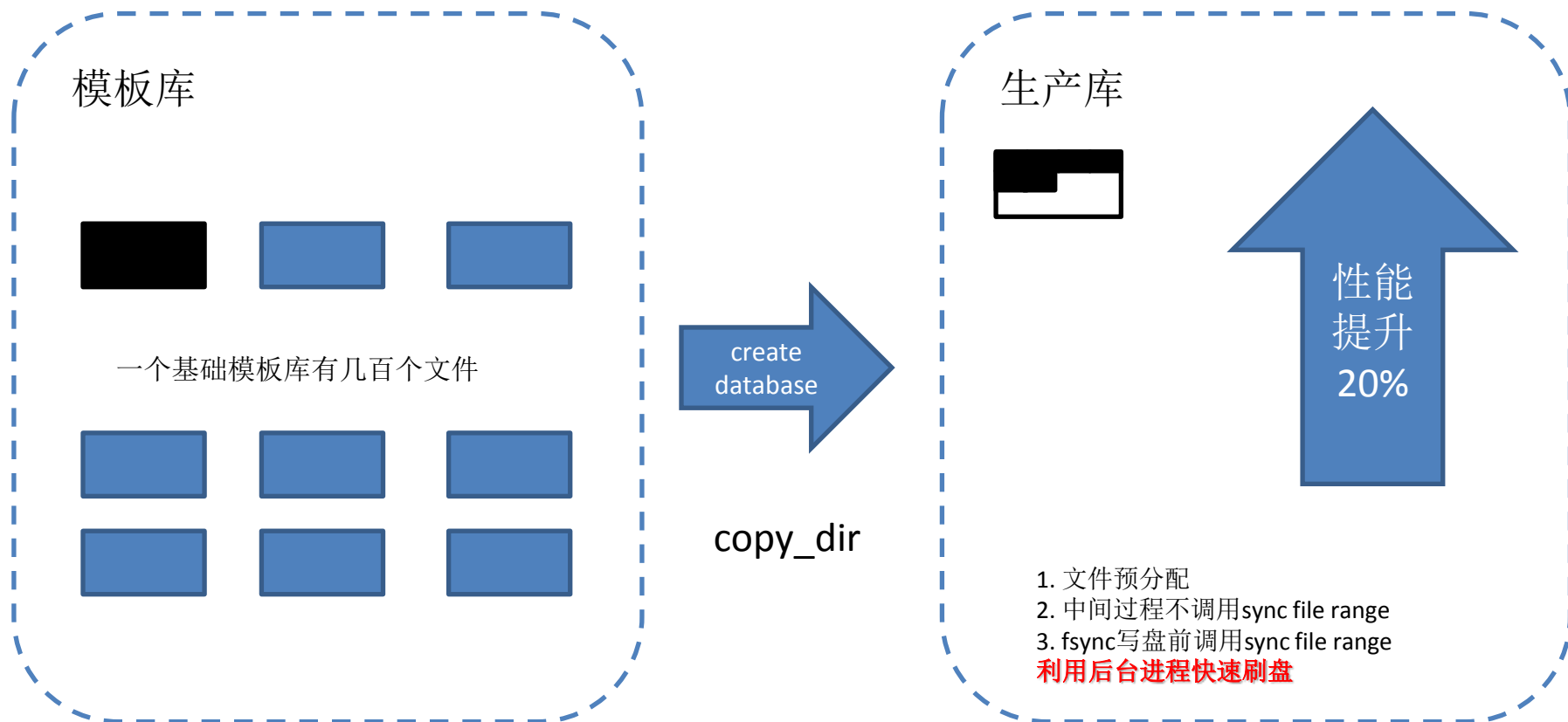


create database需要拷贝几百个文件。

追加形式COPY文件,
(每次copy COPY_BUF_SIZE = (8 * BLCKSZ))

每write一个BUF单位,
需要调用sync file range。
用户进程的IO较多, 不利于合并。

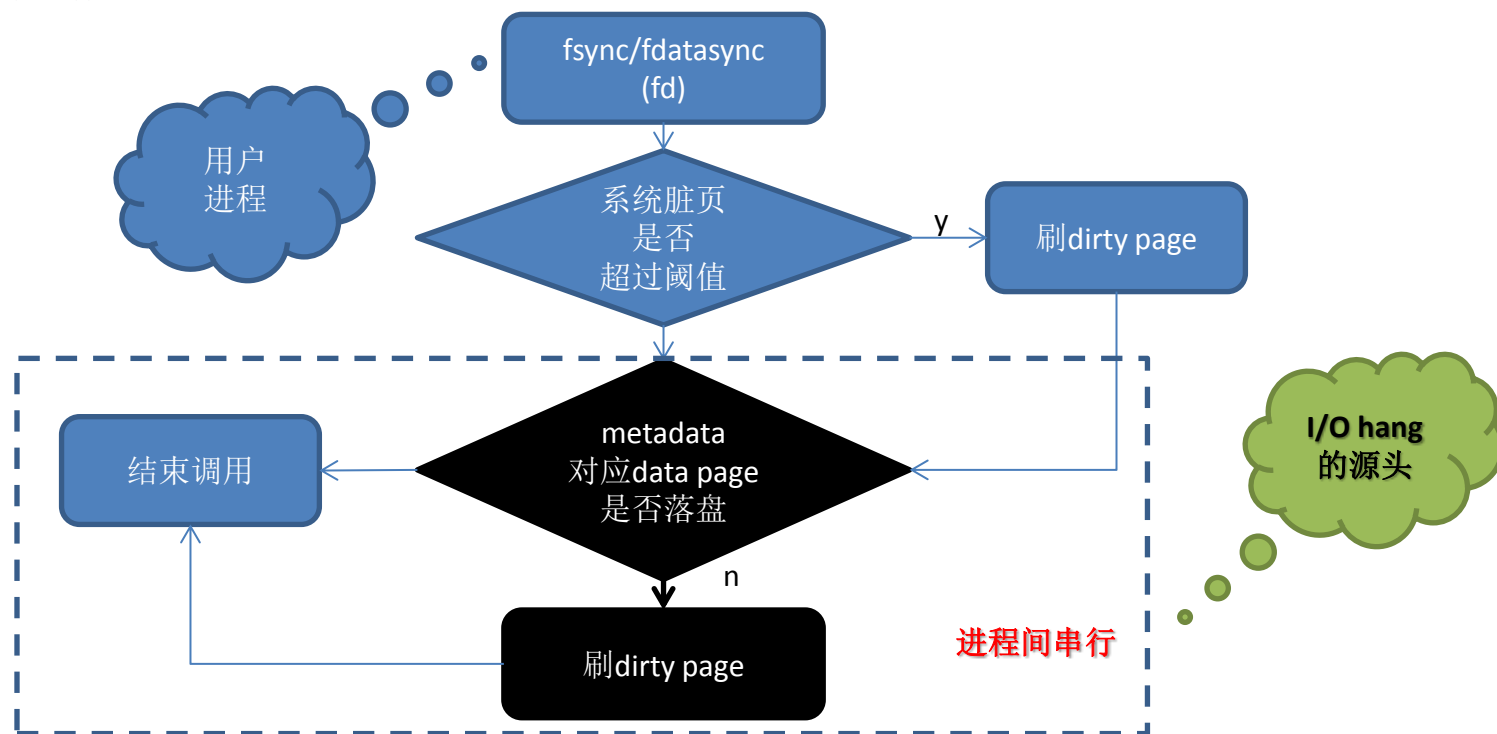
频繁建库内核优化



I/O hang的问题分析

- **ext4 data=ordered**

- metadata很小，但是写 metadata 串行
- metadata落盘前，必须确保对应的data已落盘
- 文件修改的时间戳变更、文件大小变更都涉及写metadata
- 脏页超过内核设置"vm.dirty_ratio或vm.dirty_bytes"时，用户进程调用fsync时，需要刷额外的脏页

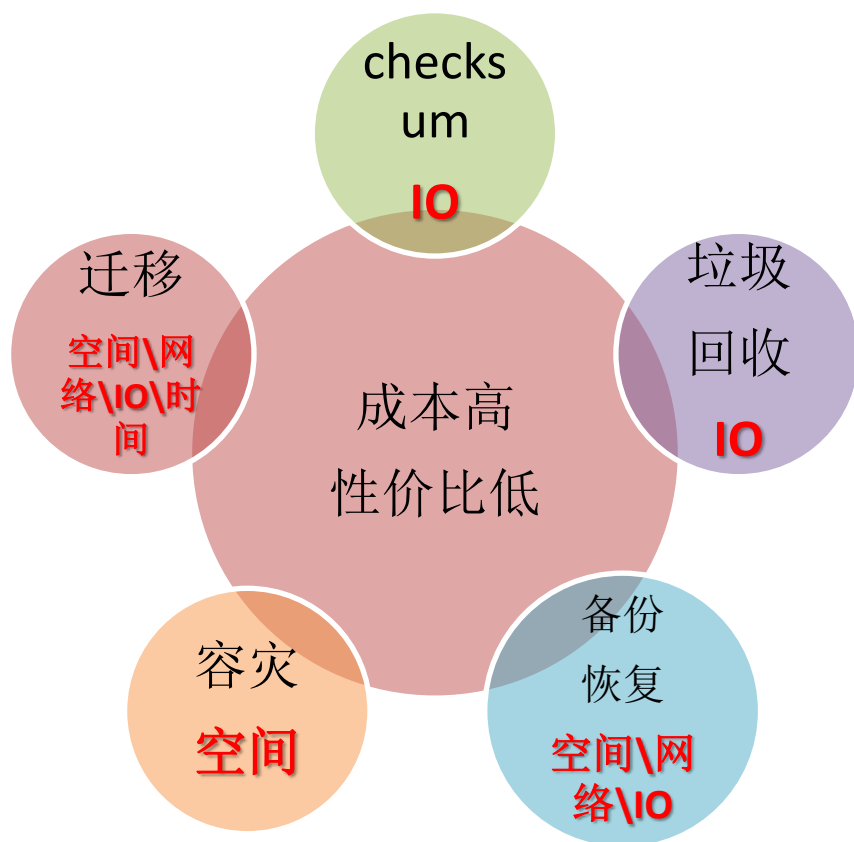


内核优化解决I/O hang的问题

- 优化原则
 - 尽量不改变metadata(modify time, size)
 - 尽量在fsync前，预处理dirty page
 - 尽量不触发进程写dirty page的阈值
- create database 优化
 - 预分配，fsync前 分段sync file range
- 检查点优化
 - 排序，sync file range。减少离散IO，减少dirty page；
 - 最后再调用fsync。降低fsync时含有大量dirty page概率；
- clog性能优化
 - 加大buffer、checkpoint时统一刷盘
 - （社区版本，即时刷盘）
- 非内核优化
 - data=writeback

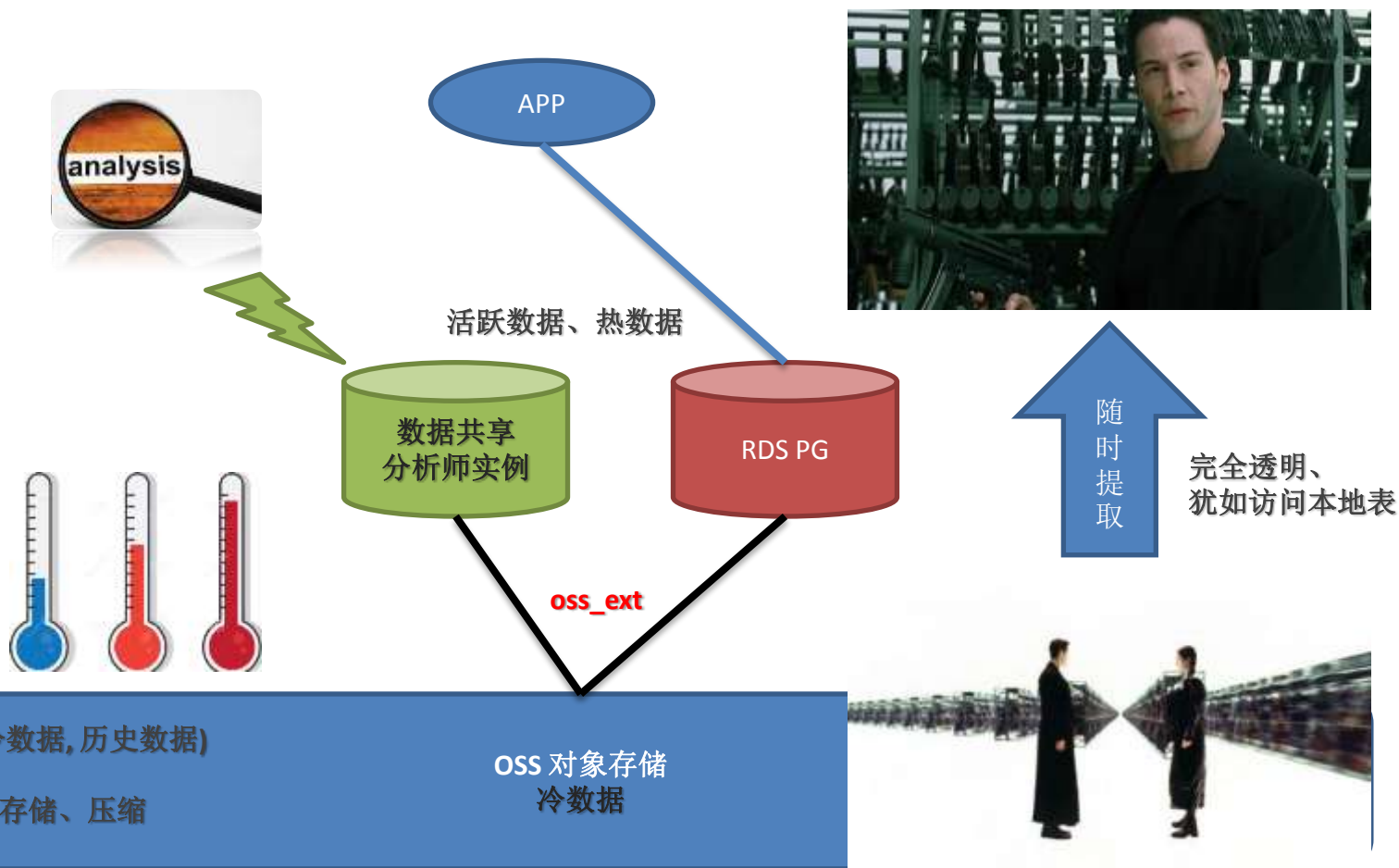
丝般顺滑

大实例的问题分析



oss_ext解决大实例问题

- 透明冷热分离 解决大实例问题



社区版本Roadmap

- https://wiki.postgresql.org/wiki/PostgreSQL10_Roadmap
- 基于流的逻辑复制
 - ApsaraDB PG 已具备
- 多核并行继续增强
- 内置分区表语法（支持hash\range\list分区）
- 动态编译query (JIT)、向量计算
- 内核内置sharding
 - Postgres-XL feed back to PostgreSQL
 - FDW 分布式特性持续增强
- 热插拔存储引擎
 - in-memory 列存储引擎、in-memory 行存储引擎、undo引擎
- 块级增量备份(通过page LSN可以分辨块变化，加入block change track)
 - pg_rman已实现
- 部分备份与恢复（类似Oracle的表空间恢复）
- 页级压缩
- 内置AWR

私人奉献PostgreSQL十年布道精华

- <https://yq.aliyun.com/groups/29>
- <http://blog.163.com/digoal@126>
- <https://github.com/digoal/blog>
- <https://yq.aliyun.com/articles/59251>

- 开发
- 管理
- 调优
- 排错
- 内核
- 最佳实践

谢谢观赏

