润乾
RAQSOFT

应用计算中间件

润乾软件出品

www.raqsoft.com.cn

目录 CONTENTS

目录 CONTENTS

01

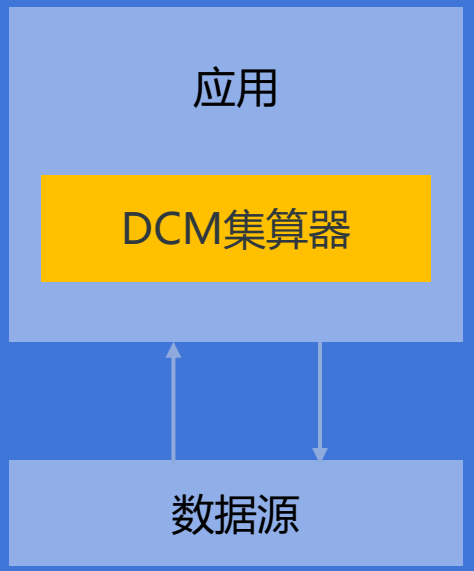
集算器是什么

数据计算中间件

集算器

- 位于数据源和应用之间提供通用计算服务的数据计算中间件（DCM, Data Computing Middleware）
- SPL（Structured Process Language）是集算器内置的程序语言
- 不依赖数据库的计算能力和可嵌入应用中使用是集算器的两个重要特征

独立计算、可嵌入使用的数据计算中间件



数据计算中间件特征

CHEASE

- Compatible
- Hot-deploy
- Efficient
- Agile
- Scalable
- Embeddable

数据计算中间件标准

CHEASE

C	H	E	A	S	E
兼容性	热部署	高性能	敏捷性	扩展性	集成性
Compatible	Hot-deploy	Efficient	Agile	Scalable	Embeddable

目录 CONTENTS

02

为什么需要计算中间件

计算无处不在

需求	• 报表统计、数据分析、业务处理，本质上都是计算 • 大量计算任务要被嵌入到应用系统内部
现状	• 大部分计算任务由数据库承担 • 封闭的数据库无法满足随时随地的计算需求

需要有不依赖于数据库的计算能力

现有技术的问题

数据库

沉重

数据库建设和使用都非常笨重、臃肿，无法很好适应灵活多变、随处可见的计算需求

封闭

数据库的计算要在库内才能实施，难以完成跨（异构）库计算、无库计算

数据入库费时费力费资源

数据库无法嵌入使用，限制灵活性

同理，Hadoop相关技术也面临同样的问题

结构性问题

现有技术的问题

SQL

数据库本身存在结构性问题，而数据库语言（SQL）则又成为影响数据库的另一因素：

难写

SQL不提倡过程，难以实现复杂计算，经常要写得很长

难调试

嵌套多层的复杂SQL很难调试

*扩展阅读：SQL像英语是个善意的错误

语法问题

SQL举例

计算目标：某支股票最长连续涨了多少交易日

```
select max(continuousDays)-1
from (select count(*) continuousDays
    from (select sum(changeSign) over(order by tradeDate) unRiseDays
        from (select tradeDate,
                case when closePrice>lag(closePrice) over(order by tradeDate)
                then 0 else 1 end changeSign
            from stock) )
    group by unRiseDays)
```

可以试试能不能读懂这句SQL，而这个计算用自然思维很好解决

*扩展阅读：SQL 的困难源于关系代数

现有技术的问题

JAVA

JAVA也常用于数据计算，但其算法实现比SQL还要复杂（缺少集合运算类库）

```
public static void groupCount(){
    groupBy mainJava = new groupBy();
    List<MainBean> list = mainJava.initList();
    List<MainBean> mainList = new ArrayList<MainBean>();
    Map<String, MainBean> map = new HashMap<String, MainBean>();
    MainBean totalBean = null;
    String employeeId = null;
    for(MainBean mainBean : list) {
        employeeId = mainBean.getEmployeeId();
        if (!map.containsKey(employeeId)) {
            totalBean = new MainBean();
            totalBean.setEmployeeId(employeeId);
            totalBean.setBuyNum(mainBean.getBuyNum());
            totalBean.setGoodsPrice(mainBean.getGoodsPrice());
            totalBean.setTotalBuyNum(mainBean.getTotalBuyNum());
            totalBean.setFreight(mainBean.getFreight());
            totalBean.setTotalGoodsPrice(mainBean.getGoodsPrice() * mainBean.getBuyNum());
            totalBean.setTotalPrice(mainBean.getGoodsPrice() * mainBean.getBuyNum() + mainBean.getFreight());
            mainList.add(totalBean);
            map.put(employeeId, totalBean);
        } else {
            totalBean = map.get(employeeId);
            totalBean.setTotalBuyNum(totalBean.getTotalBuyNum() + mainBean.getTotalBuyNum());
            totalBean.setFreight(totalBean.getFreight() + mainBean.getFreight());
            totalBean.setTotalGoodsPrice(totalBean.getTotalGoodsPrice() + mainBean.getGoodsPrice() * mainBean.getBuyNum());
            totalBean.setTotalPrice(totalBean.getTotalPrice() +
                    mainBean.getGoodsPrice() * mainBean.getBuyNum() + mainBean.getFreight());
        }
    }
```

这么长的代码仅仅实现了单个字段分组单个字段汇总的功能

语法问题

现有技术的问题

Python

Python也被用于数据计算

难以实现复杂计算

pandas不是专业结构化数据计算包，在处理分组有序等复杂运算时较为繁琐

集成性差

python几乎不具备集成性，很难嵌入到应用中实施计算

*扩展阅读：Pandas不擅长的结构化数据运算

语法与体系问题

目录 CONTENTS

02

集算器特性

SPL与集算器

C	H	E	A	S	E
兼容性	热部署	高性能	敏捷性	扩展性	集成性

JAVA开发	解释执行	并行计算	敏捷语法	分布式环境	易集成嵌入

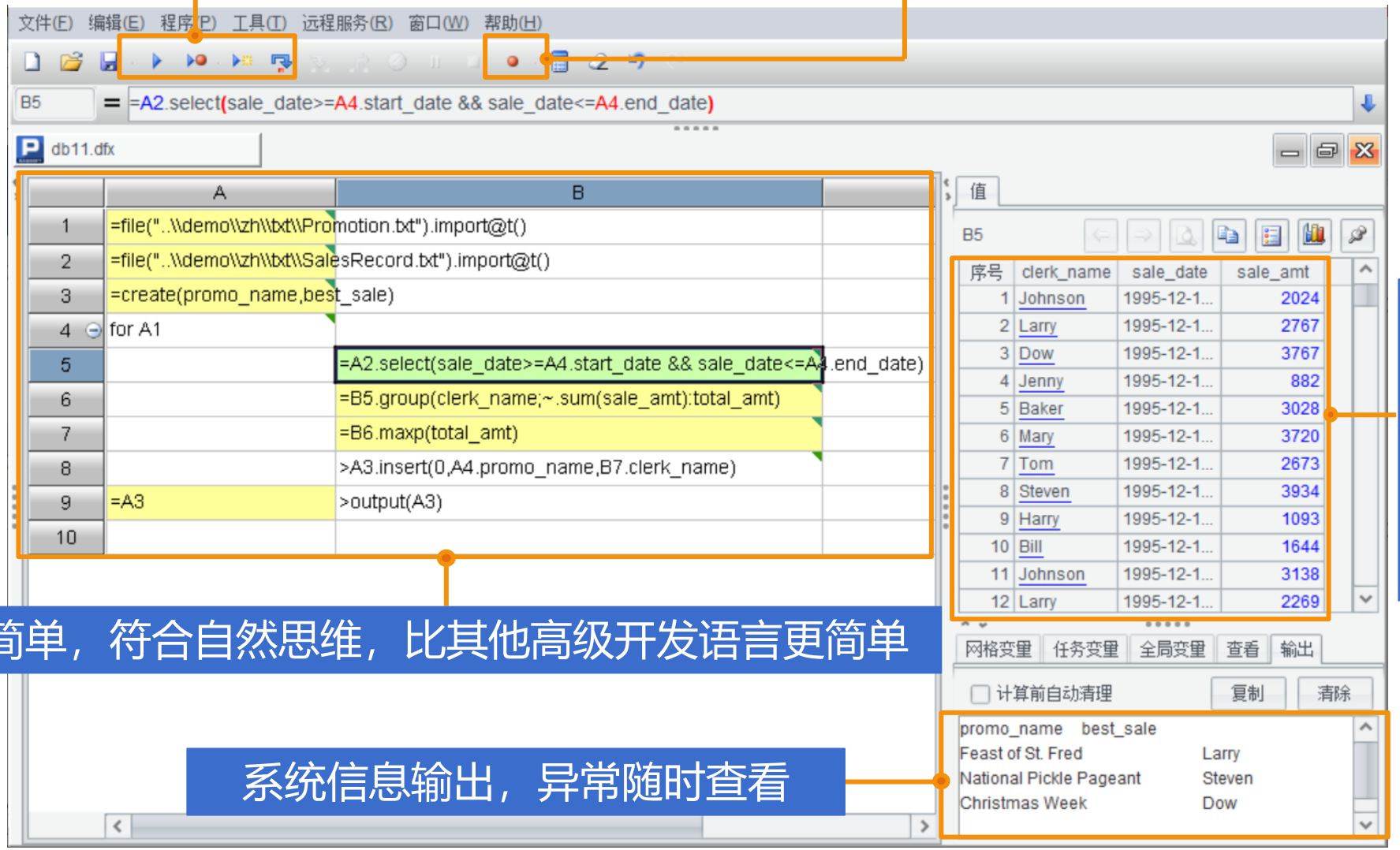
SPL
(Structured Process Language)

集算器 SPL Ware

简洁易用的开发环境

执行、调试执行、执行到光标、单步执行

设置断点



网格结果所见即所得，易于调试；方便引用中间结果

语法简单，符合自然思维，比其他高级开发语言更简单

系统信息输出，异常随时查看

敏捷语法

计算目标：某支股票最长连续涨了多少交易日

```
select max(continuousDays)-1
from (select count(*) continuousDays
      from (select sum(changeSign) over(order by tradeDate) unRiseDays
            from (select tradeDate,
                         case when closePrice>lag(closePrice) over(order by tradeDate)
                         then 0 else 1 end changeSign
                  from stock) )
      group by unRiseDays)
```

SQL解法

- SQL在使用窗口函数的情况下嵌套三层完成；
- 前面读懂了吗？

	A
1	=stock.sort(tradeDate)
2	=0
3	=A1.max(A2=if(closePrice>closePrice[-1],A2+1,0))

SPL解法

其实这个计算很简单，按照自然思维：先按交易日排序（行1），然后比较当天收盘价比前一天高就+1，否则就清零，最后求个最大值（行3）

专门设计的语法体系

SPL特别适合复杂过程运算

	A	B	C	D	E	F
1	=esProc.query("SELECT 订单ID AS 合同,订购日期 AS 日期,客户 ,订单金额 AS 金额,员工ID AS 销售 FROM 销售记录表 WHERE year(订购日期)=? OR year(订购日					
2	=esProc.query("select * from 员工表")					
3	>A1.run(销售=A2.select@1(编号:A1.销售))		/字段值 是记录			
4	=A1.group(销售)					
5	=create(销售,今年销售额,去年销售额,增长率,客户数,大客户数,大客户占比)					
6	for A4	=A6(1).销售.姓名				
7		=A6.select(year(日期)==年份).sum(金额)		/今年销售额		
8		=A6.select(year(日期)==年份-1).sum(金额)		/去年销售额		
9		=B8/B7-1	/增长率			
10		=A6.group(客户).(~.sum(金额))				
11		=B10.count()				
12		=B10.count(~>10000)	/大客户数			
13		=B12/B11				
14		>A5.insert(0,B6,B7,B8,B9,B11,B12,B13)				
15	result A5					

天然分步、层次清晰、直接引用单元格名**无需定义变量**

丰富的运算类库

专门针对结构化数据表设计

	A	B	C
1	=esProc.query("SELECT 订单ID AS 合同,订购日期 AS		/读取销售记录表
2	=A1.group(销售)		
3	=create(销售,今年销售额,去年销售额,客户数,大客户数)		
4	for A2	=A4(1).销售	
5		=A4.select(year(日期)==年份).sum(金额)	
6		=A4.select(year(日期)==年份-1).sum(金额)	
7		=A4.group(year(日期)).count(金额)	
8		=B7.count()	
9		=B7.count(~>=10000)	

分组、循环

	A	B	C
1	=esProc.query("select * from 员工表")		
2	=A1.select(性别=="男")		
3	=A1.select(出生日期>=date("1970-01-01"))		
4	=A2^A3	/交运算，统计晚于1970年出生的男员工	
5	=A2&A3	/并运算，统计男员工或者晚于1970年出生的员工	
6	=A2\A3	/运算，统计早于1970年出生的男员工	
7	=A4.sum(工资)		
8	=A5.avg(年龄)		
9	=A6.sort(出生日期)		
10	/集合作为基本数据类型		
11			

集合运算

	A	B	C
1	=file("交易记录.txt").import@t()		
2	=A1.sort(客户编码,交易日期)		
3	=A2.select(车辆型号=="捷达" || 车辆型号=="迈腾").dup@t()		
4	=A3.derive(interval(交易日期[-1],交易日期):间隔)		
5	=A4.select(车辆型号[-1]=="捷达" && 车辆型号=="迈腾" &&客户编码==客户编码[-1])		
6	=A5.avg(间隔)		
7			
8			
9			

排序、过滤

	A	B	C
1	=esProc.query("select * from 员工表")		
2	=A1.sort(入职日期)		
3	=A2.pmin(出生日期)	/出生最早的员工的记录序号	
4	=A2(to(A3-1))	/直接用序号访问成员	
5	=esProc.query("select * from 股价表 where 股票代码='000062'")		
6	=A5.sort(交易日期)		
7	=A6.pmax(收盘价)	/收盘价最高的那条记录的序号	
8	=A6.calc(A7,收盘价[收盘价[-1]-1)		
9			
10	/直接用序号访问成员		
11			

有序集合

多样性数据源

直接使用多个数据源混合计算，无需后台先将数据统一（ETL）后再计算

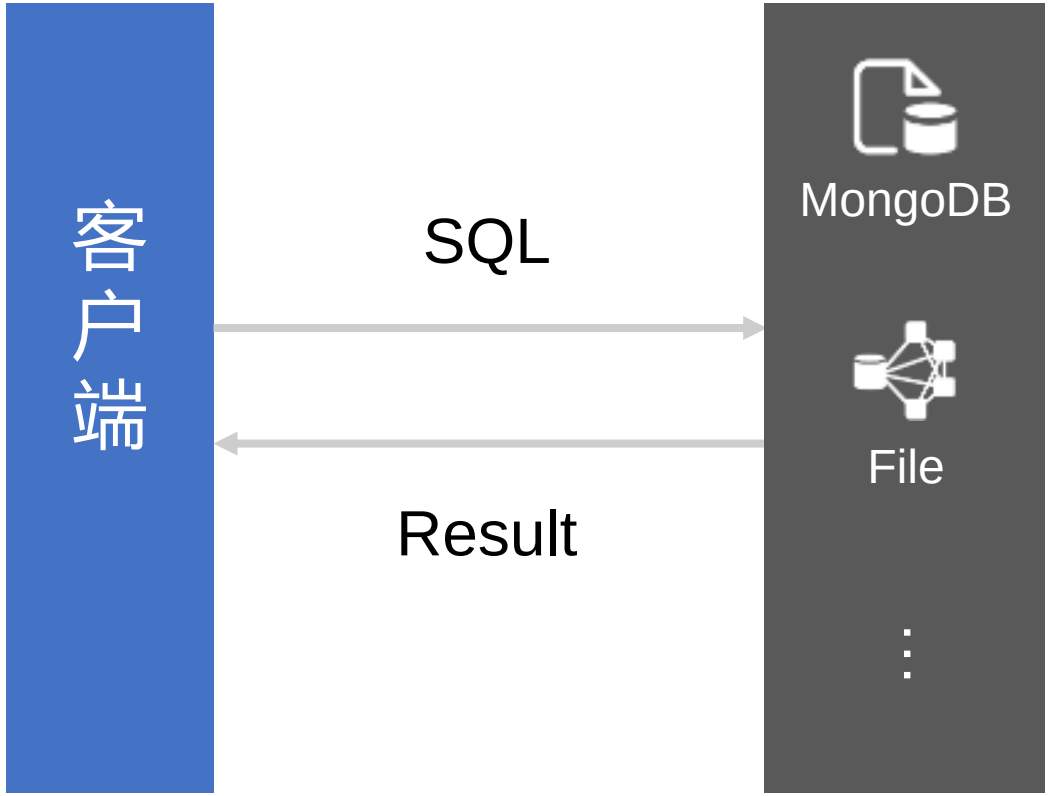
外部数据接口

- 商用 RDBMS：Oracle、MS SQL Server、DB2、Informix
- 开源 RDBMS：MySQL、PostgreSQL
- 开源 NOSQL：MongoDB、Redis、Cassandra、ElasticSearch
- Hadoop家族：HDFS、HIVE、HBase
- 应用软件：SAP ECC、BW
- 文件：Excel、Json、XML、TXT
- 其他：Http Restful、Web Services、OLAP4j 、...

内置接口，即装即用

文件SQL查询

通过SPL可以针对MongoDB和文件等使用SQL进行查询



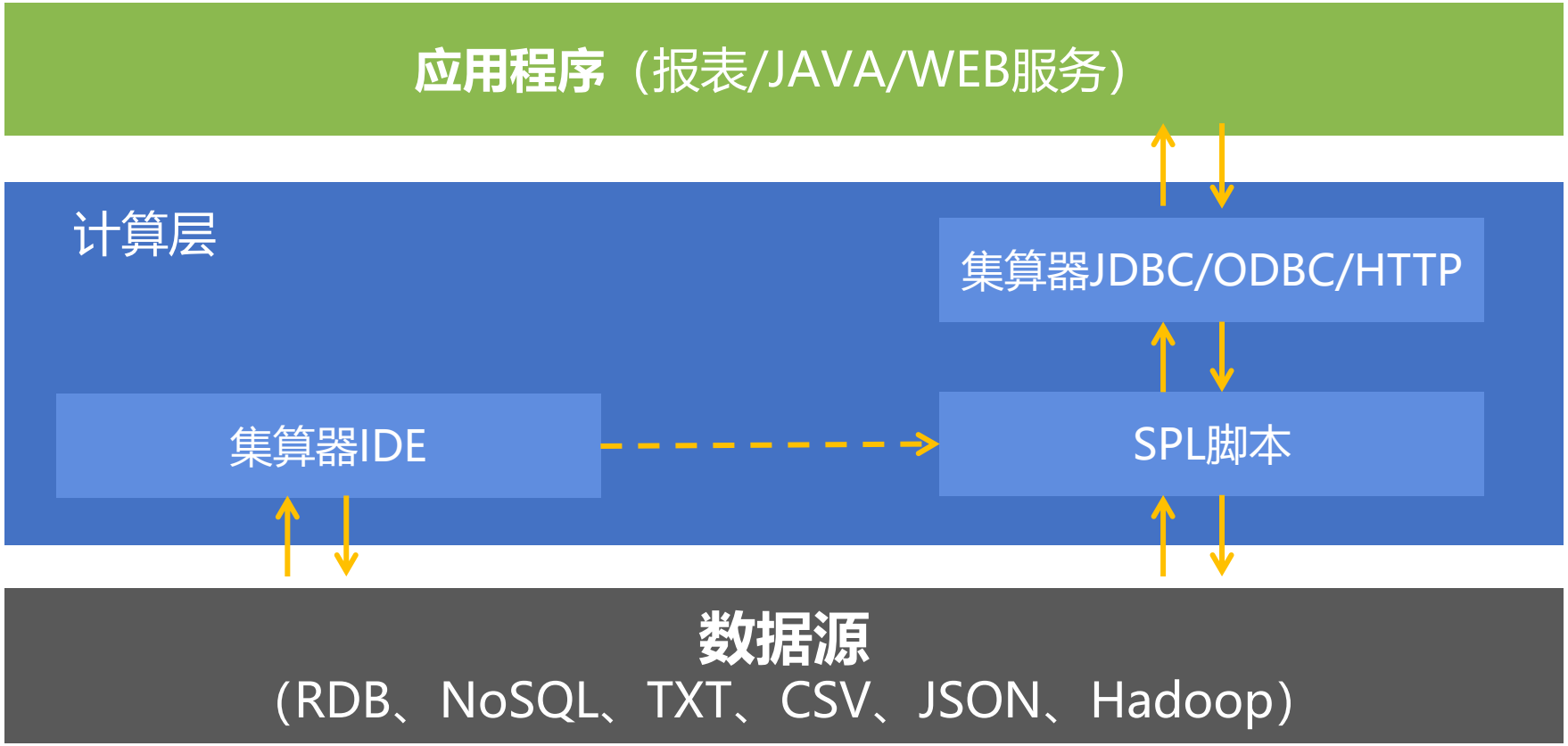
集算器赋予

NoSQL和文件

SQL查询能力

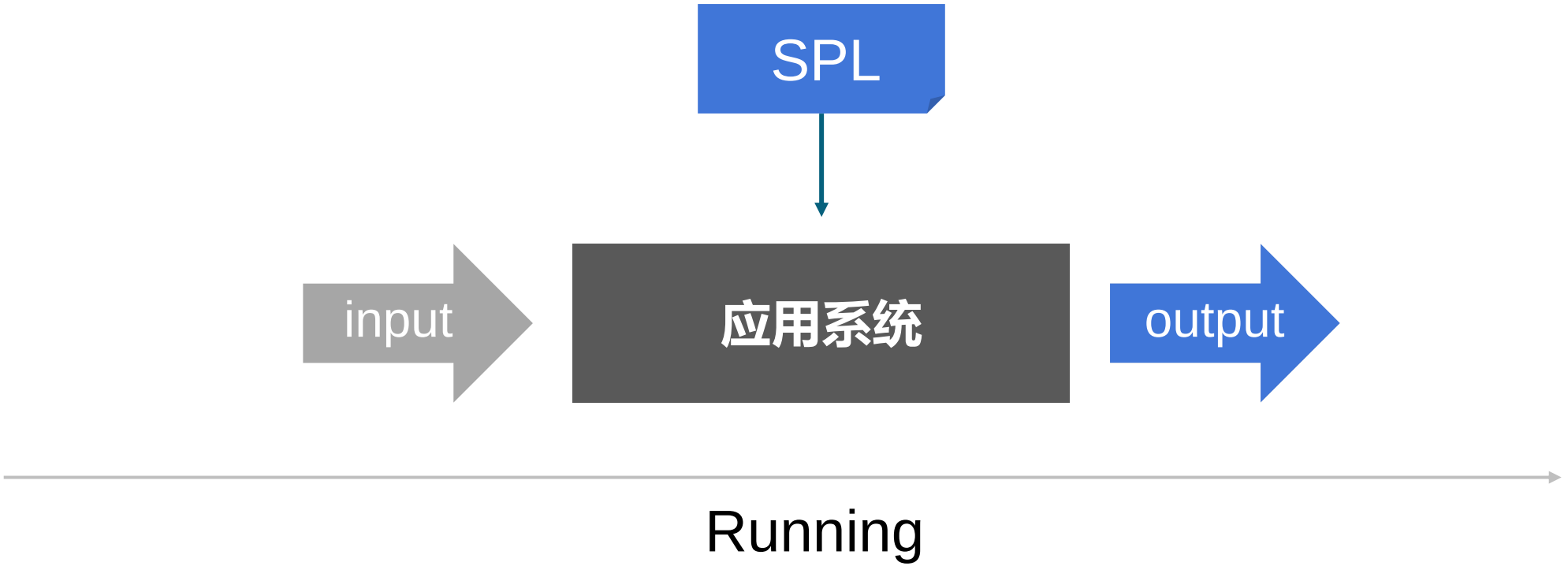
集成性

集算器使用JAVA开发，提供标准应用接口可无缝集成到应用中



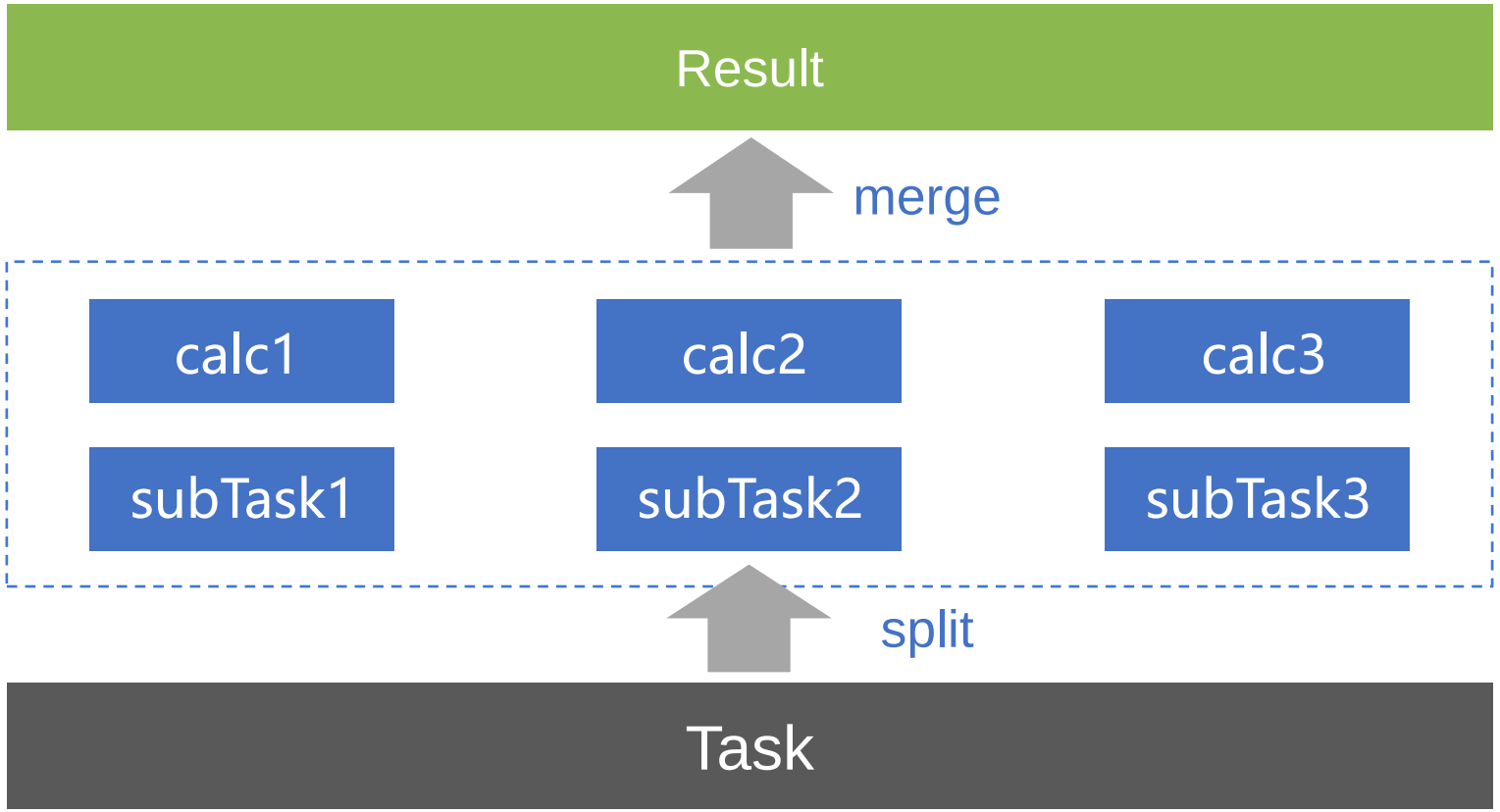
热切换

SPL解释执行，支持不停机热切换



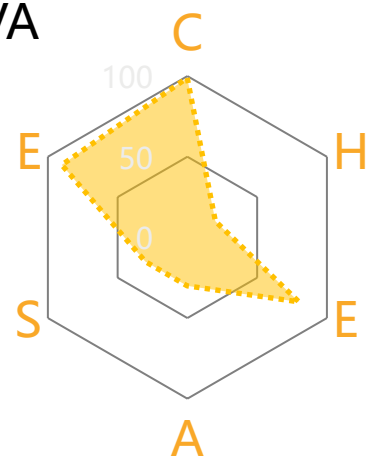
多线程并行

方便地针对单任务实施多线程计算

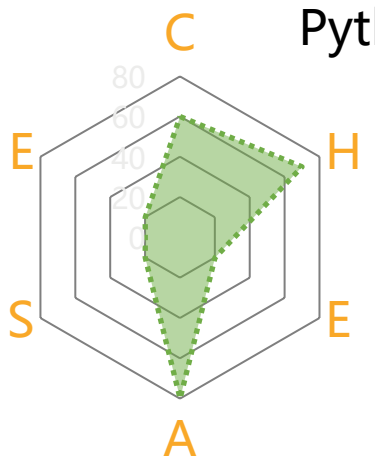


SPL能力圈

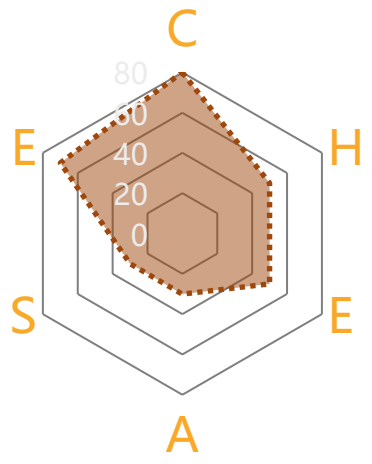
JAVA



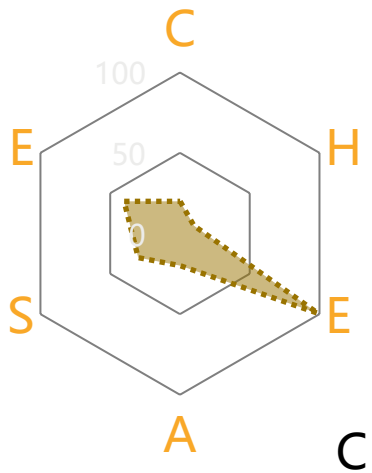
Python

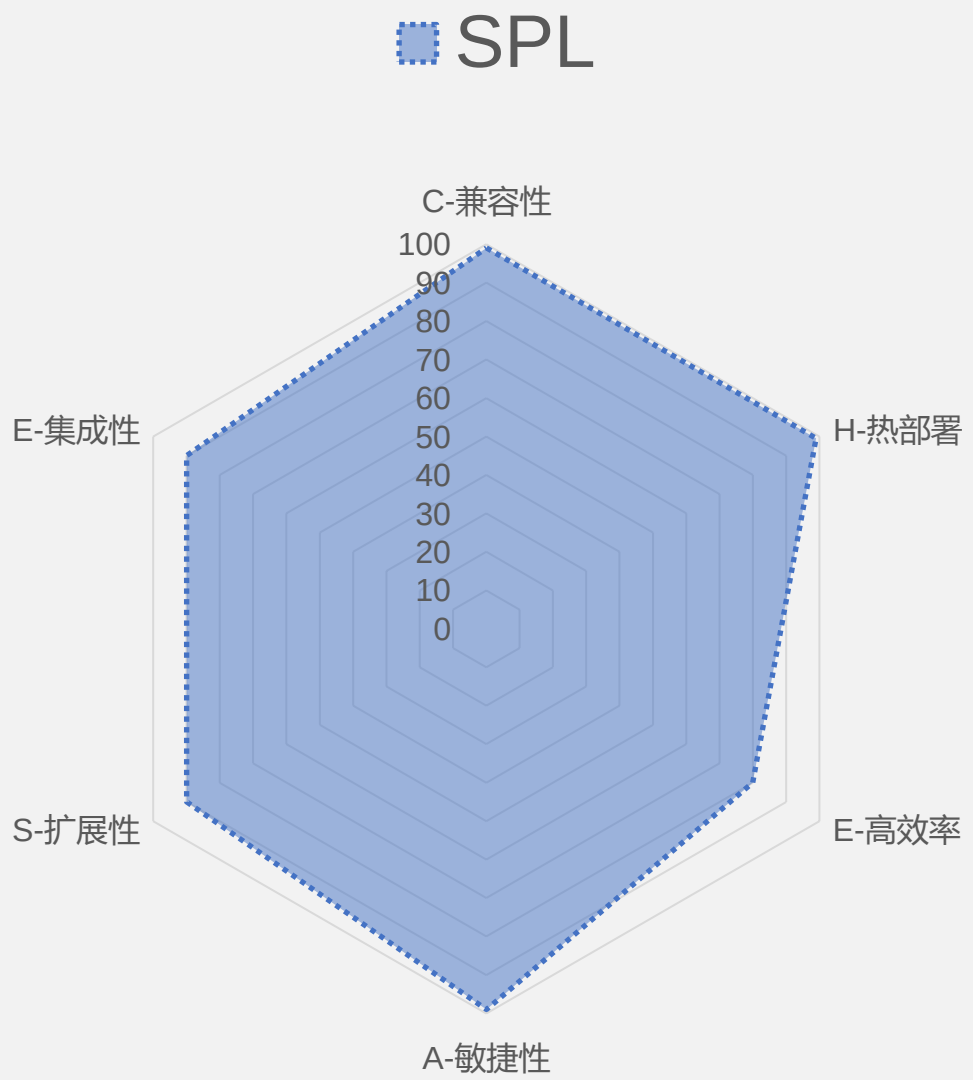


SQL



C++





目录 CONTENTS

04

应用场景

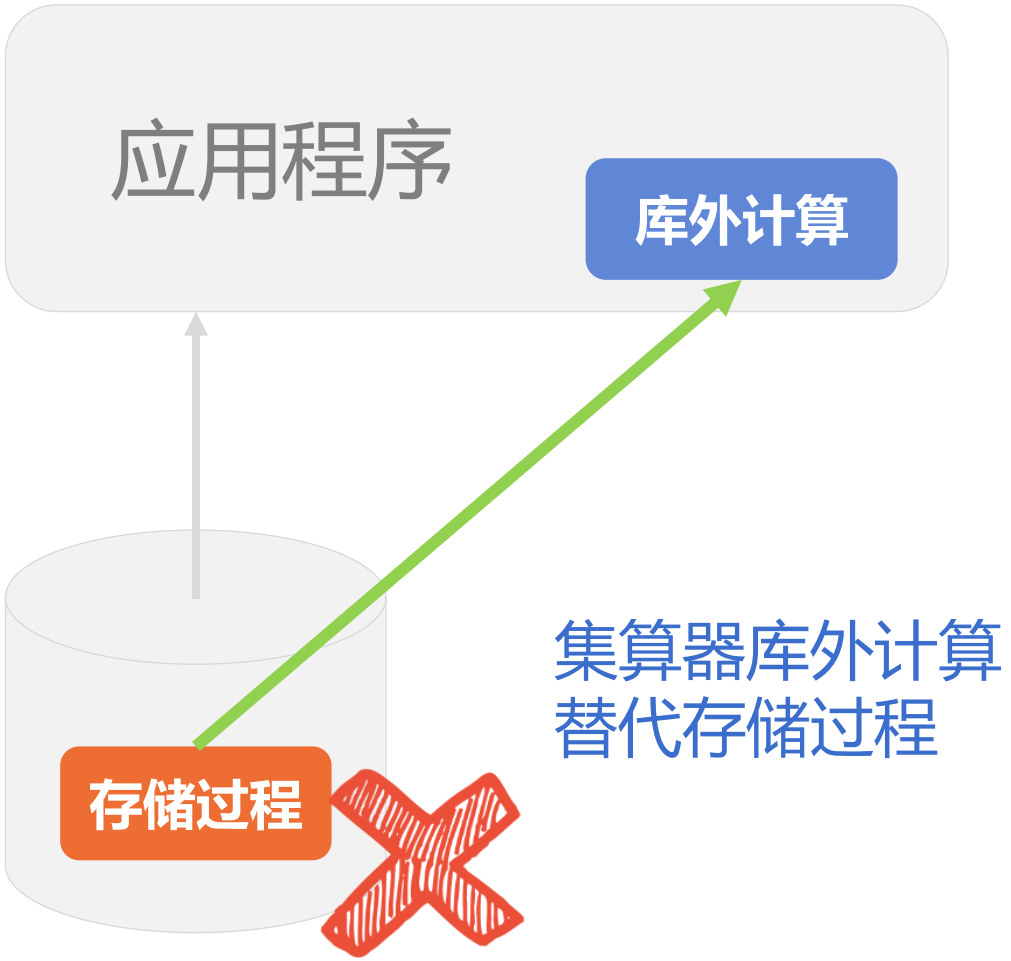
减少存储过程

存储过程的目的

- 数据整理
- 呈现准备

存储过程的问题

- 造成应用内与应用间耦合
- 安全性低
- 移植性差



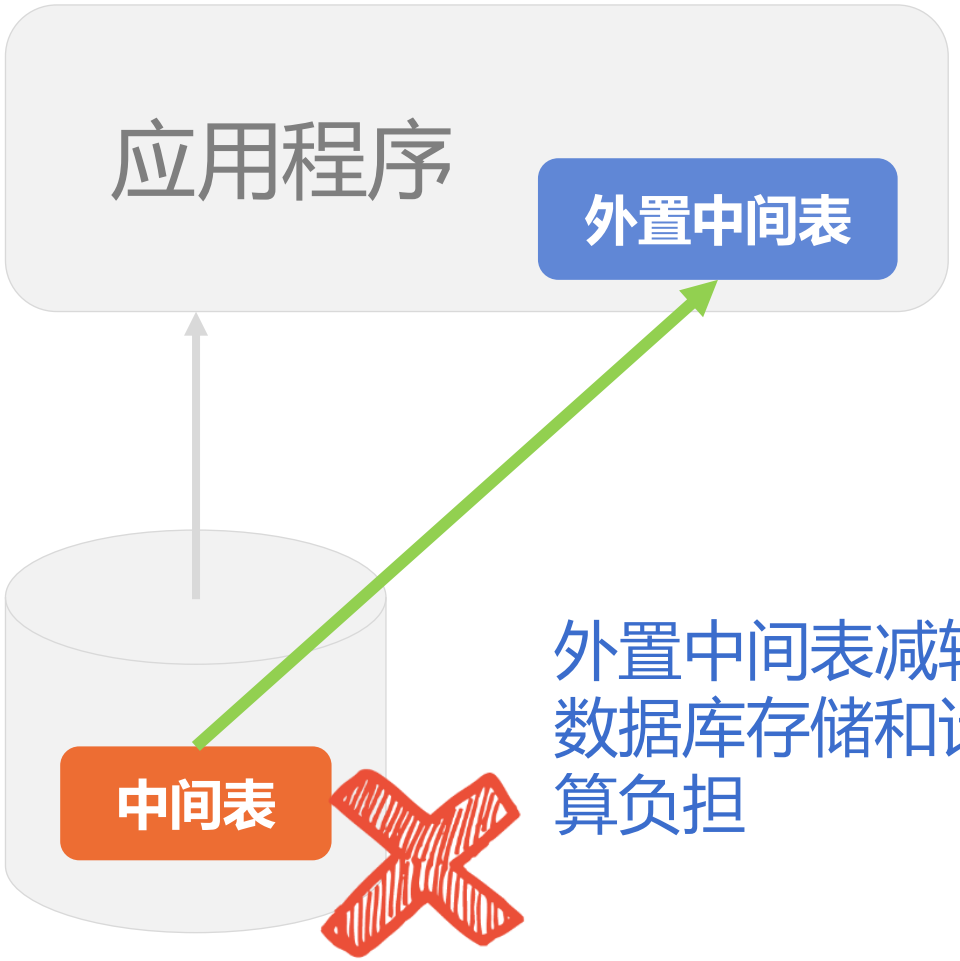
中间表替代

中间表的目的

- 获得更高计算效率

中间表的问题

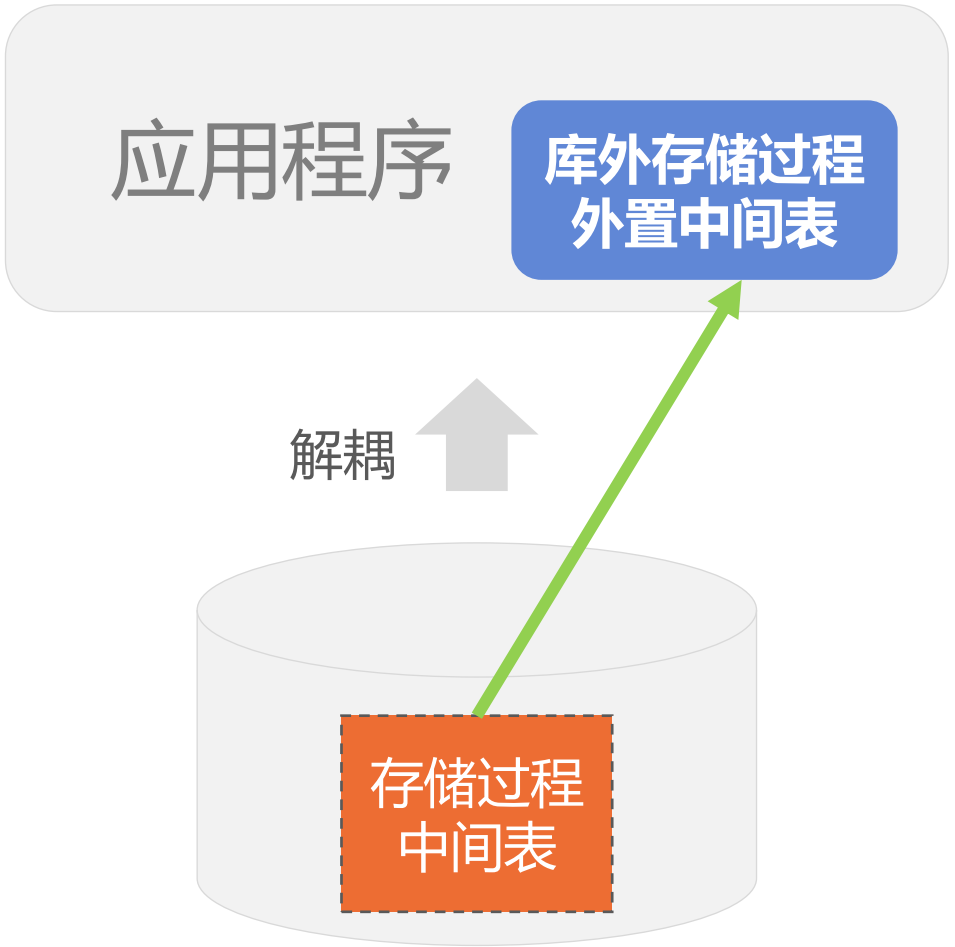
- 造成应用内与应用间耦合
- 过多引发数据库管理问题
- 影响数据库容量和性能



外置中间表减轻数据库存储和计算负担

数据库解耦

- 将存储过程和中间表外置到应用中
- 数据库仅承担存储和少量（通用）计算
- 应用与数据库解耦，易维护，易扩展



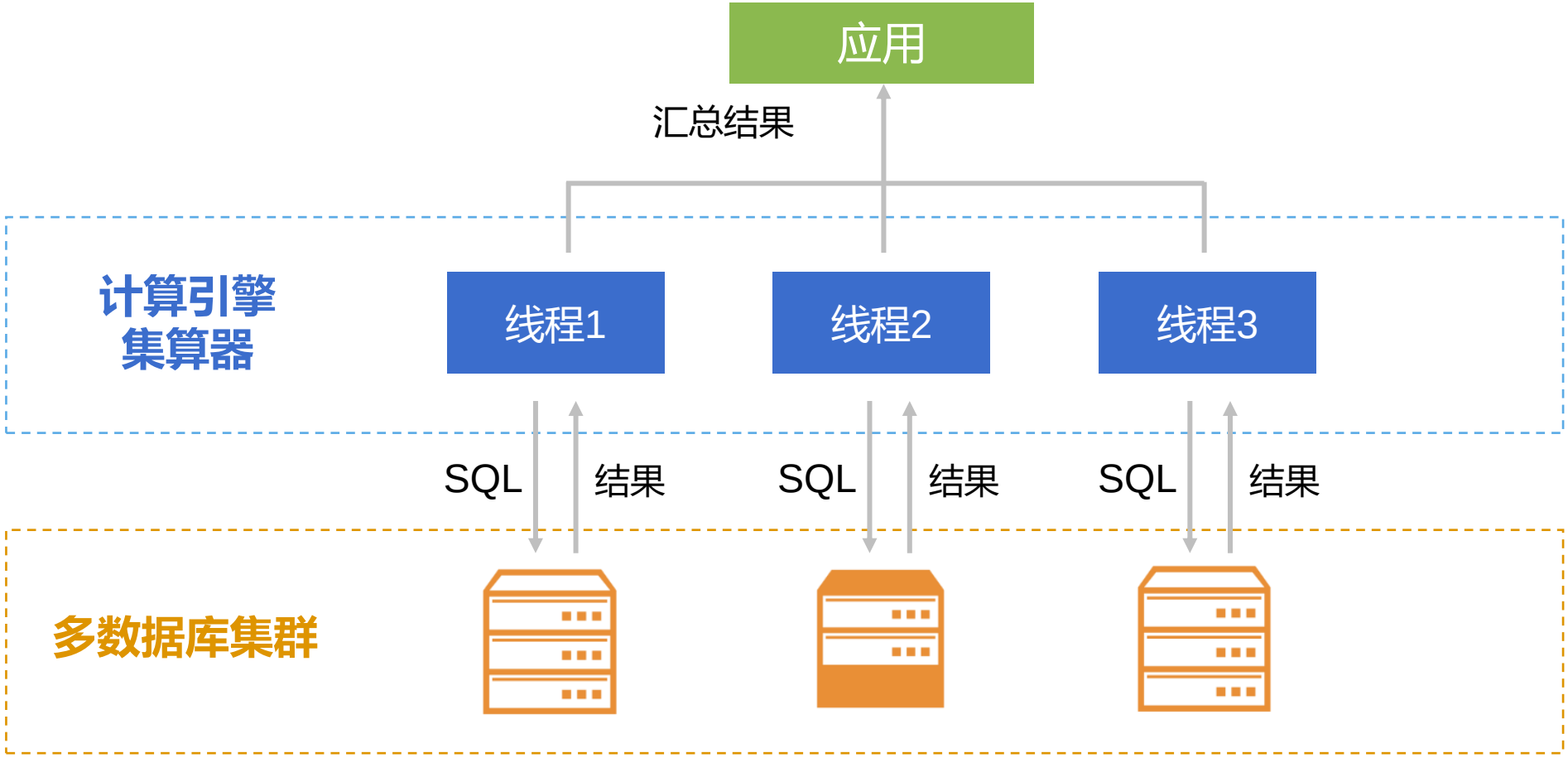
RDB协助

集算器还可以辅助RDB计算，提升RDB能力

- SQL迁移将标准SQL转化为各类数据库“方言”
- 并行取数提升性能
- 跨库/异构库计算进一步增强RDB能力

跨库集群

借助集算器的跨库与并行能力实现多数据库集群计算



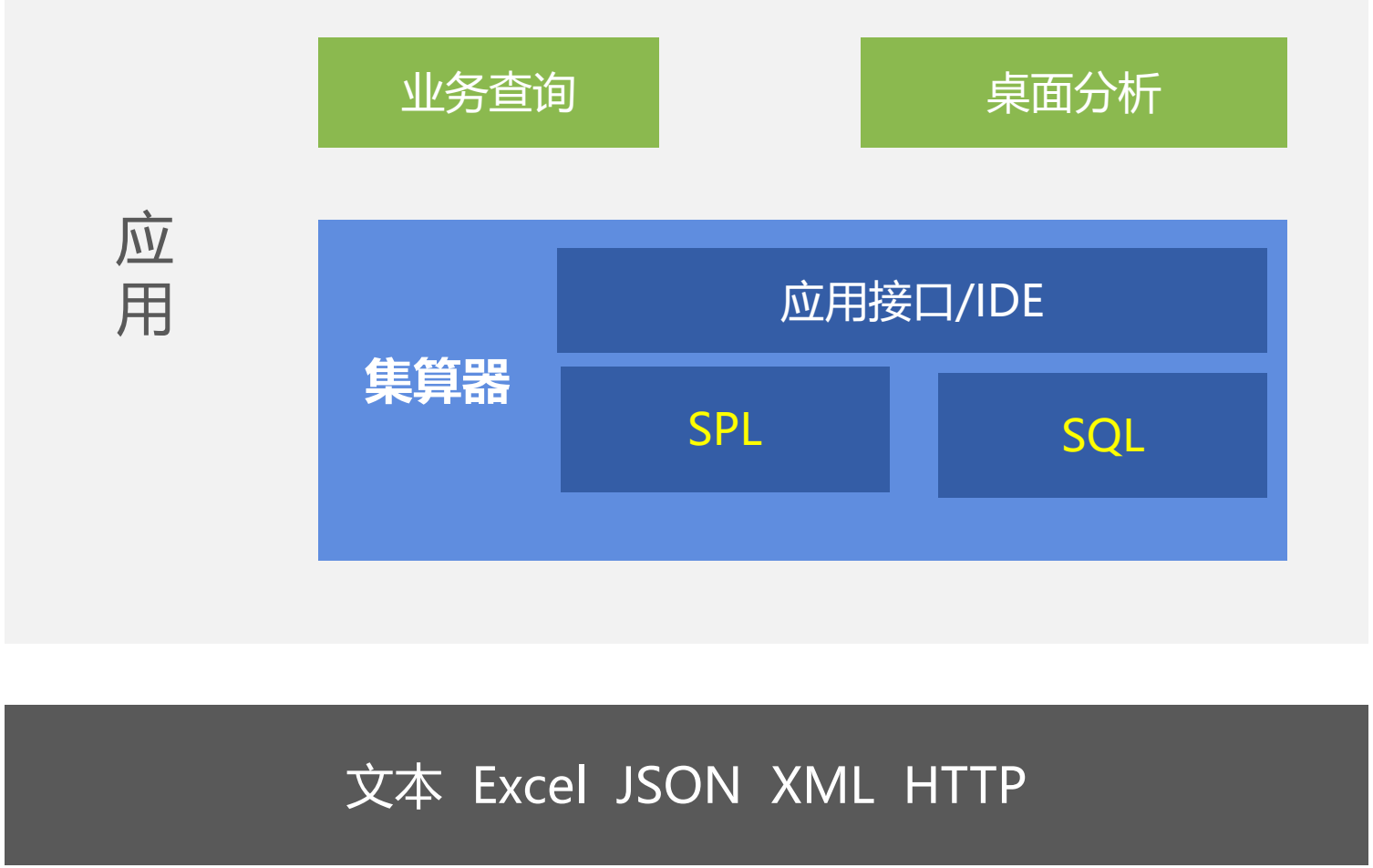
多样性数据源实时混算

集算器同时关联多个数据源进行实时数据混算，提供跨源T+0查询

- 嵌入应用计算
- 异构源实时混算
- 冷热数据实时混算
- 数据无需物理同库

文件计算

可以通过SPL原生语法进行文件计算，同时支持SQL查文件，简单方便



- 提供两种文件计算方式
- 熟悉SQL可以零成本上手

MongoDB计算

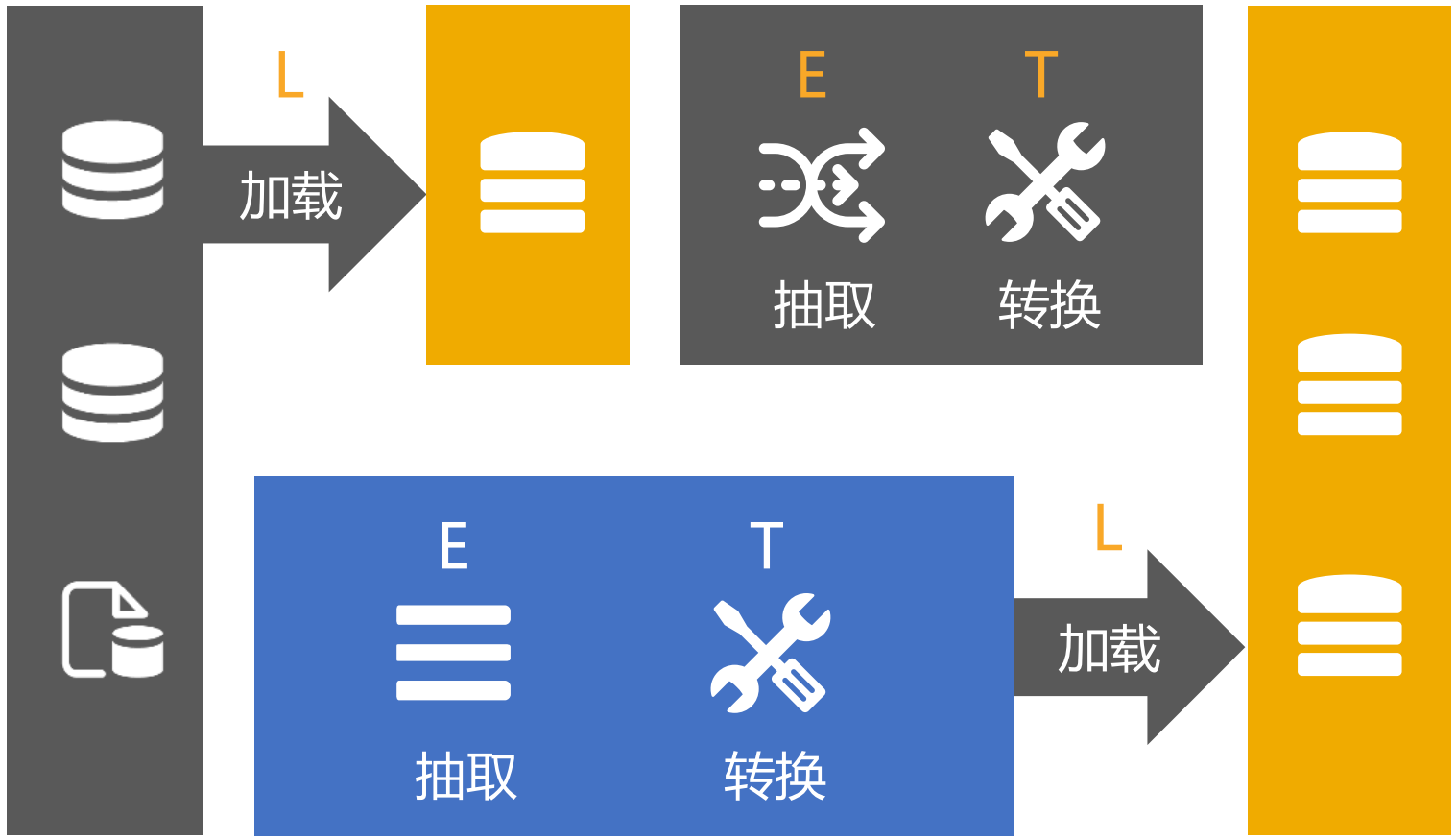
集算器可以强化MongoDB计算能力，简化计算过程

- 使得MongoDB达到或强于RDB的计算能力
- 增强计算能力后，充分发挥MongoDB原有优势

mongoDB

ETL

传统ETL往往先L再ET，费时费力；通过集算器可以实现真正的ETL过程

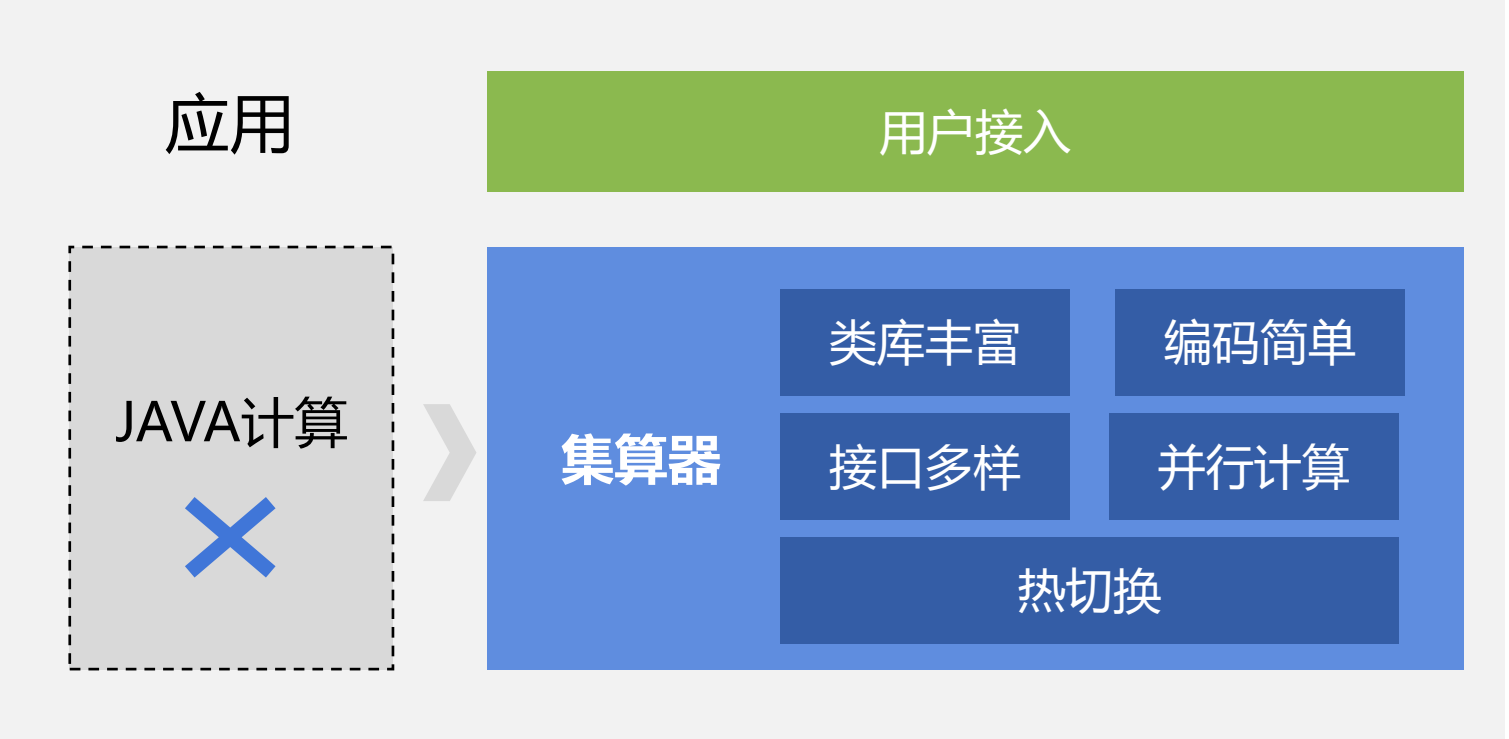


ETL复杂计算

- 库外处理，为数据库减负
- 减少IO，缩短时间窗口
- 实现简单灵活
- 允许多源混合处理

JAVA计算替代

使用集算器替代JAVA计算可以降低编码难度，提升运算效率



- JAVA编码难度高，集算器更加简洁高效
- JAVA不支持热切换，SPL解释执行支持热切换

应用解耦

使用JAVA做处理在线计算会导致与应用耦合度过高，集算器可以解耦计算模块，单独运行维护

JAVA

模块化困难

Java程序必须和主应用一起编译打包，耦合度高

难以热切换

使用Java编写的算法有修改后会导致整个应用重新编译部署，很难做到热切换。

集算器

模块化简单

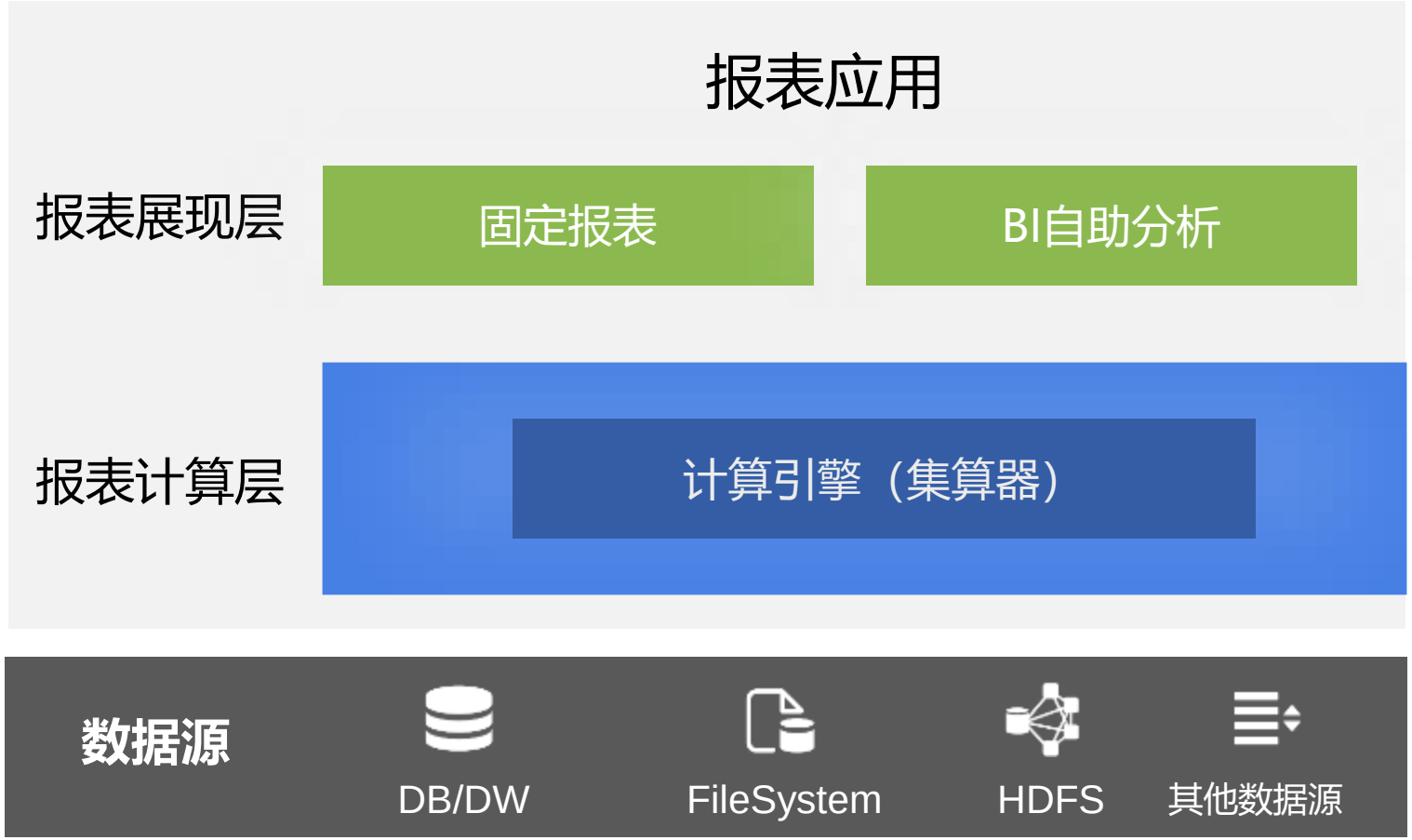
集算器脚本文件可以单独维护，方便模块化

容易热切换

集算器是解释执行的语言，很容易做到热切换

报表内计算

集算器嵌入报表应用中完成报表数据准备工作（计算引擎），弥补报表工具本身计算能力不足



- 提高报表开发效率
- 降低报表与应用耦合度
- 降低报表与数据源耦合度
- 提升报表计算性能

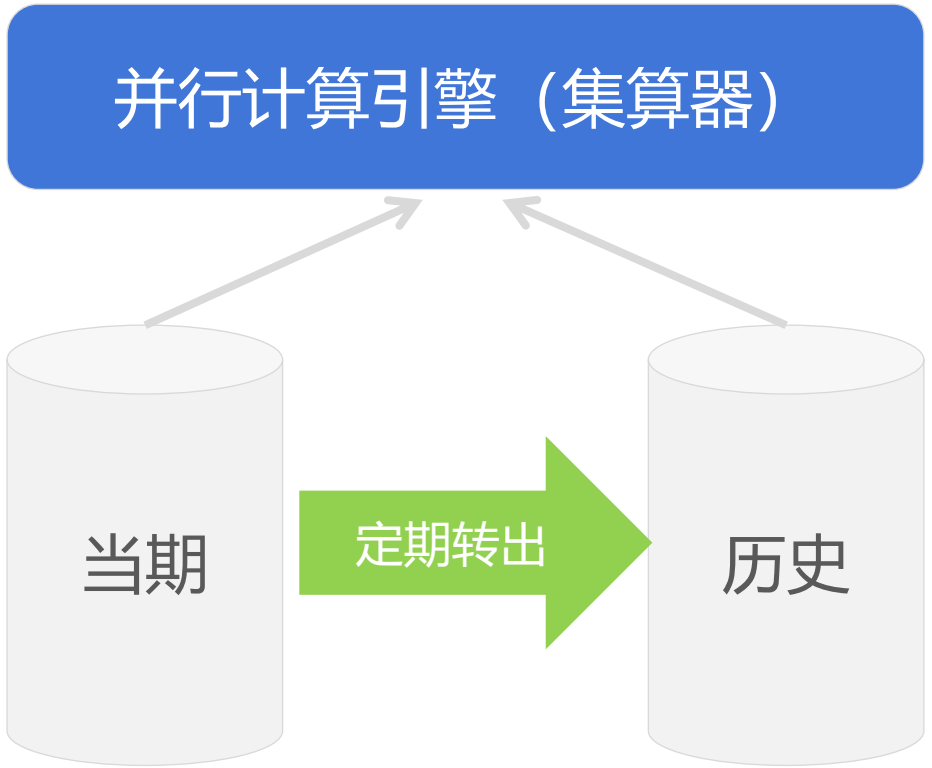
T+0查询报表

T+0问题

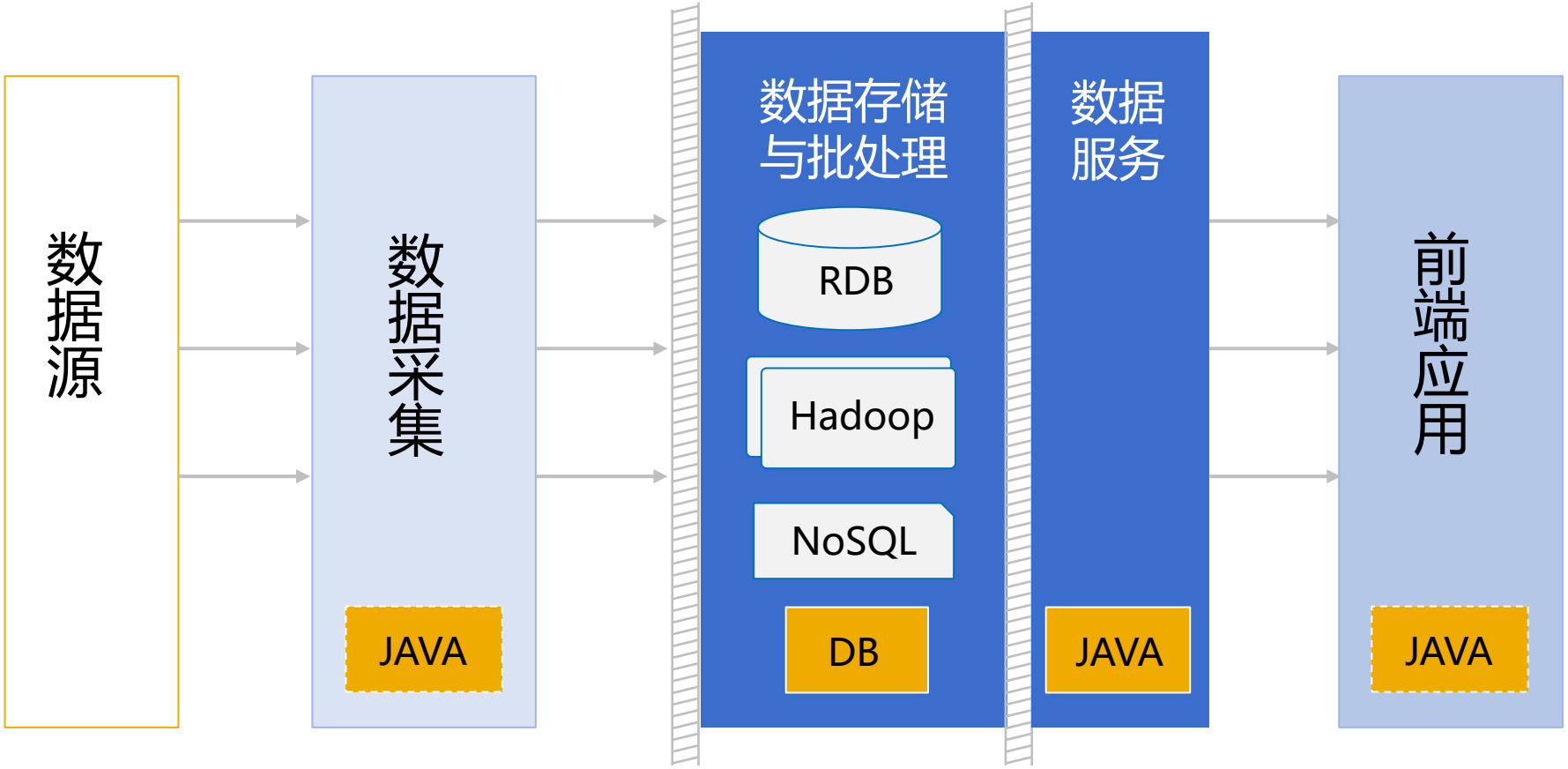
- 交易一致性要求关系数据库
- 历史与当期同库，数据量太大
- 历史与当期异库，跨库计算困难

库外计算实现并行跨库计算

- 历史数据还可文件化



当前数据中台和微服务架构



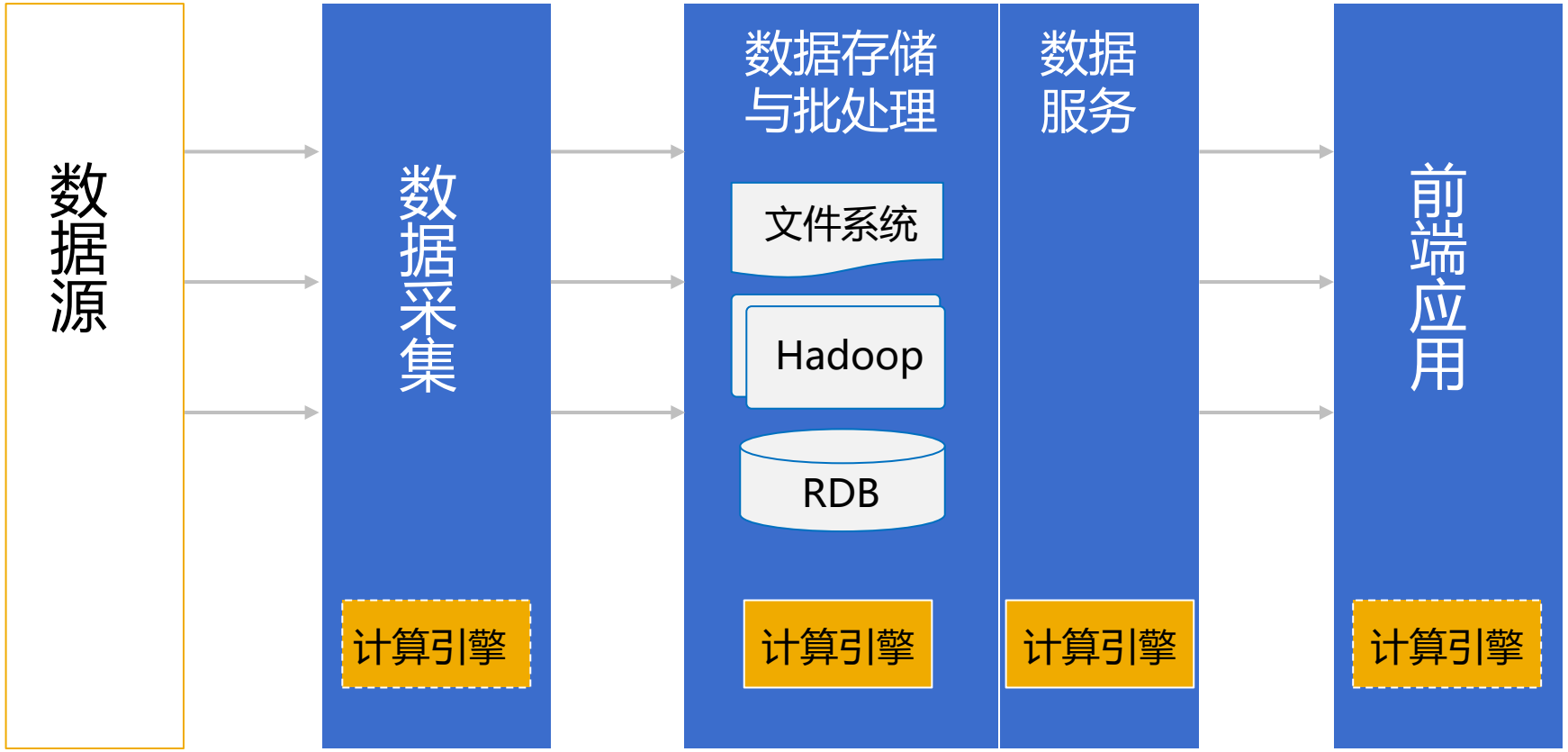
混合库计算 ✗

实时数据源 ✗

算法热切换 ✗

只有数据进入“库里”才能实施计算，数据采集或分析应用中计算只能硬编码

引入开放计算引擎后的数据中台与微服务架构



- 混合库计算
- 实时数据源
- 算法热切换

开放的计算能力分布在涉及计算的各个阶段；没有“库”的封闭，体系更加开放

更多资料

SPL CookBook

http://www.raqsoft.com.cn/wx/SPL-cookbook.html

应用案例

http://c.raqsoft.com.cn/article/1565768237788

润乾
RAQSOFT

THANKS

创新技术 推动应用进步

www.raqsoft.com.cn