新一代数据处理引擎

润乾软件　张政宏

目录

现状分析

文本结构化运算

入库时间比计算时间还要长

关联计算问题

开源工具的集成问题

SQL不好写

过程计算、组内计算、有序计算

存储过程性能差、难调试、耦合性强

开源数据库缺窗口函数、过程计算偏弱

多源计算

数据中心服务接口

广告计算ROI

Hadoop

学习成本高、维护成本高

SQL能力和传统数据库相差还很远

现状思考

现实情况

业务数据没那么大、用hadoop很重

相同的业务复杂度，开源工具尝试结果并不理想

大型商业软件很贵

内存越来越大、越来越廉价（160T）

单机性能越来越高（SSD、闪存）

我们需要怎样的数据处理工具？

开放体系

拥有数据库的全套计算能力，而不依赖数据库

集成性

和主流工程语言轻松集成

轻量级

杀鸡不要用牛刀

适用面

强描述能力

高性能

集算器介绍

润乾集算器

面向（半）结构化数据计算的程序设计语言

提供丰富的结构化计算类库，支持多线程并行

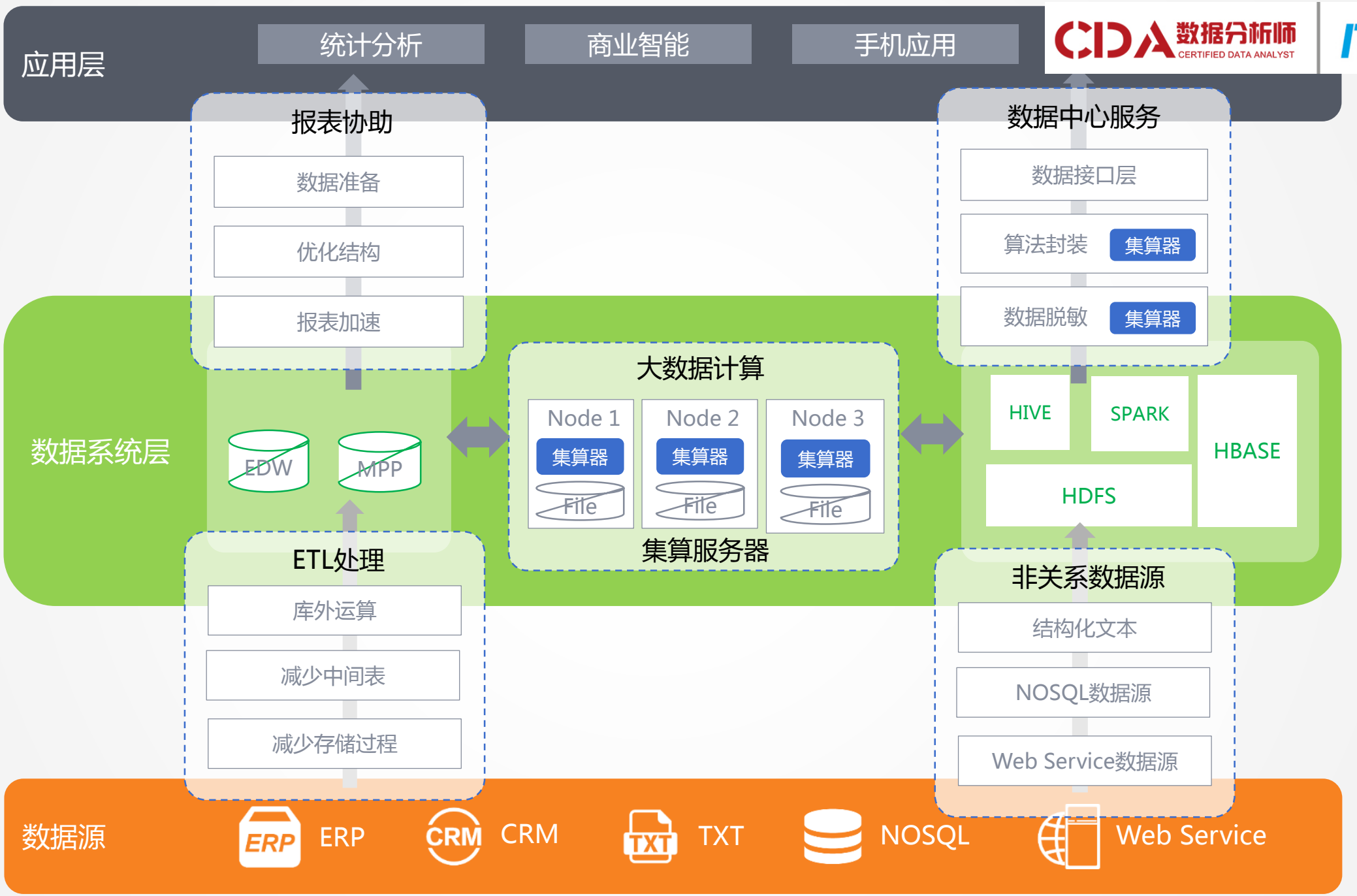
优势

实施交付：过程式与函数式编程结合、逻辑过程更贴近人的自然思维
提高开发效率、降低描述业务问题的难度

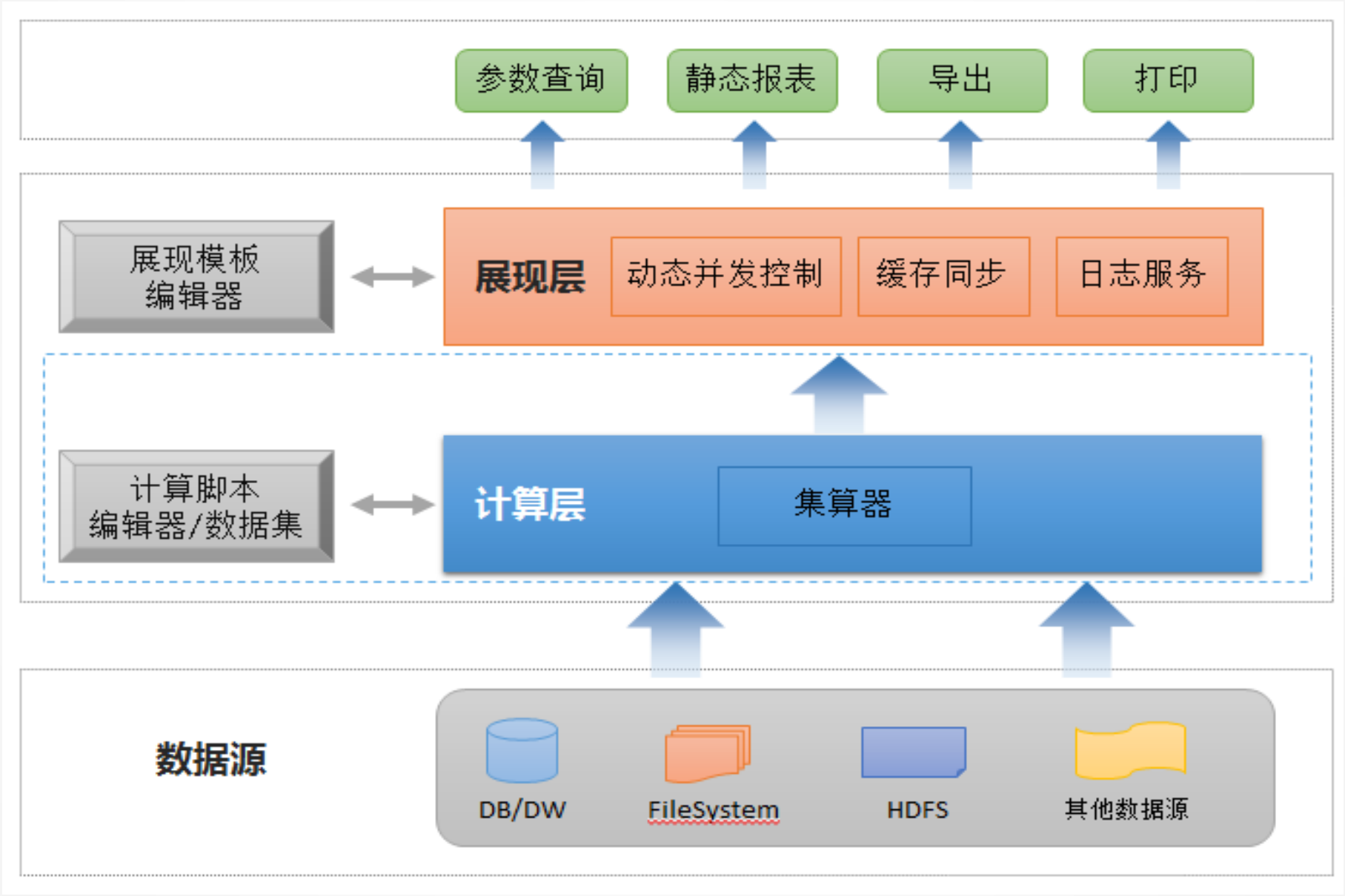
性能提升：面向数据特征的算法和类库

维护管理：优化结构体系、降低应用之间耦合度、方便维护和管理

应用场景


应用层
统计分析
商业智能
手机应用
报表协助
数据准备
优化结构
报表加速
数据中心服务
数据接口层
算法封装
集算器
数据脱敏
集算器
数据系统层
EDW
MPP
大数据计算
Node 1
集算器
File
Node 2
集算器
File
Node 3
集算器
File
集算服务器
HIVE
SPARK
HBASE
HDFS
ETL处理
库外运算
减少中间表
减少存储过程
非关系数据源
结构化文本
NOSQL数据源
Web Service数据源
数据源
ERP
CRM
TXT
NOSQL
Web Service

应用结构


参数查询
静态报表
导出
打印
展现模板
编辑器
展现层
动态并发控制
缓存同步
日志服务
计算脚本
编辑器/数据集
计算层
集算器
数据源
DB/DW
FileSystem
HDFS
其他数据源

集算器计算模型

离散性与集合化的有效结合

集合化是批量计算的基本能力

离散计算也不可或缺

离散性支持更彻底的集合化

离散性产生有序集合运算

离散数据集 = 集合运算 + 游离成员

=> 彻底集合化/有序计算

=> 更高的开发效率和执行效率

强描述能力 – 归纳举例

离散性

计算任务：张三和李四的年龄差与工资差

```
1  SELECT (SELECT age FROM employee WHERE name= ‘张三’ )
2  - (SELECT age FROM employee WHERE name= ‘李四’ )
3  FROM dual
4  SELECT (SELECT salary FROM employee WHERE name= ‘张三’ )
5  - (SELECT salary FROM employee WHERE name=‘李四’)
6  FROM dual
```

SQL

	A
1	=employee.select@1(name=="张三")
2	=employee.select@1(name=="李四")
3	=A1.age-A2.age
4	=A1.salary-A2.salary

集算脚本

引用式外键

计算任务：选出北京地区的交易记录

	A	
1	>交易记录.switch(地区,地区表:编号)	建立外键引用
2	=交易记录.select(地区.名称== “北京”)	用外键引用记录的字段过滤

分组子集

计算任务：用户在最后一次登录前三天内的登录次数

	A
1	=登录表. group(uid;~.max(logtime):last,~.count(interval(logtime,last)<=3):num)

针对分组子集的聚合运算较复杂时难以用简单聚合式写出，保留分组子集再结合分步计算则很容易

SQL不能保持子集，要用子查询在原集上附加信息，导致多次计算

```
1  WITH T AS
2        (SELECT uid,max(logtime) last FROM 登录表 GROUP BY uid)
3  SELECT T. uid,T.last,count(TT.logtime)
4        FROM T LEFT JOIN 登录表 TT ON T.uid=TT.uid
5        WHERE T.last-TT.logtime<=3 GROUP BY T.uid,T.last
```

强描述能力 – 归纳举例

非常规聚合

计算任务：列出用户首次登录的记录

	A
1	=登录表.group(uid).(~.minp(logtime))

聚合运算不一定总是SUM/COUNT这些，还可以理解为取出某个成员

有离散性时可以简单针对分组子集实施这种聚合

```
1  SELECT * FROM
2      (SELECT RANK() OVER(PARTITION BY uid ORDER BY logtime) rk, T.* FROM 登录表 T) TT
3  WHERE TT.rk=1;
```

有序分组

计算任务：一支股票最长连续上涨了多少天

	A
1	=股票.sort(交易日).group@i(收盘价<收盘价[-1]).max(~.len())

另一种和次序有关的分组，条件成立时产生新组

```
1  SELECT max(连续日数) FROM
2        (SELECT count(*) 连续日数 FROM
3              (SELECT SUM(涨跌标志) OVER ( ORDER BY 交易日) 不涨日数 FROM
4              ( SELECT 交易日,
5                    CASE WHEN  收盘价>LAG(收盘价) OVER( ORDER BY 交易日 THEN 0 ELSE 1 END 涨跌标志
6              FROM 股票 ))
7        GROUP BY 不涨日数)
```

分组有序计算

计算任务：找出连续上涨三天的股票

	A
1	=股票.sort(交易日).group(代码)
2	=A1.select((a=0,~.pselect(a=if(收盘价>收盘价[-1],a+1,0):3))>0).(代码)

分组子集与有序计算的组合

```
1  WITH A AS
2      (SELECT 代码,交易日, 收盘价-LAG(收盘价) OVER (PARITITION BY 代码 ORDER BY 涨幅) FROM 股票)
3  B AS
4      (SELECT 代码,
5          CASE WHEN 涨幅>0 AND
6              LAG(涨幅) OVER (PARTITION BY 代码 ORDER BY 交易日) >0  AND
7              LAG(涨幅,2) OVER PARTITION BY 代码 ORDER BY 交易日) >0
8              THEN 1 ELSE 0 END 三天连涨标志  FROM A)
9  SELECT distinct 代码 FROM B WHERE 三天连涨标志=1
```

高性能

遍历技术

- 延迟游标
- 遍历复用
- 聚合理解
- 有序游标

关联计算

- 区分JOIN
- 外键指针化
- 外键序号化
- 有序归并

并行计算

- 多线程
- 分段并行

集群计算

- 数据分布
- 集群维表

聚合理解

从一个集合计算出一个单值或另一个集合都可理解为聚合

高复杂度的排序问题转换为低复杂度的遍历问题

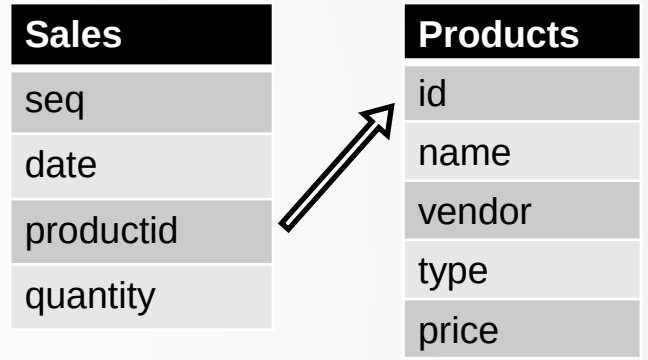
	A	
1	=file(“data.txt”).cursor@t()	
2	=A1.groups(;top(10,amount))	金额在前10名的订单
3	=A1.groups(area;top(10,amount))	每个地区金额在前10名的订单

高性能 – 面向数据特征

外键指针化

外键需要随机小量频繁访问

内存指针查找大幅提高性能



	A	
1	=file(“Products.txt”).import()	读入商品列表
2	=file(“Sales.txt”).import()	读入销售记录
3	>A2.switch(productid,A1:id)	建立指针式连接，把商品编号转换成指针
4	=A2.sum(quantity*productid.price)	计算销售金额，用指针方式引用商品单价

	Java指针连接	Oracle
单表无连接	0.57s	0.623s
五表外键连接	2.3s	5.1s

外键序号化

序号化相当于外存指针化

	A	
1	=file(“Products.txt”).import()	读入商品列表
2	=file(“Sales.txt”).cursor()	根据已序号化的销售记录建立游标
3	=A2.switch(productid,A1:#)	用序号定位建立连接指针，准备遍历
4	=A3.groups(;sum(quantity*productid.price))	计算结果

不需要再计算Hash值和比较

有序游标

复杂处理需要读出到程序内存中再处理

有序游标有效减少查找和遍历数量

	A	B	C
1	=file(“user.dat”).cursor@b()		/按用户id排序的源文件
2	for A1;id	…	/从游标中循环读入数据，每次读出一组id相同
3		…	/处理计算该组数据

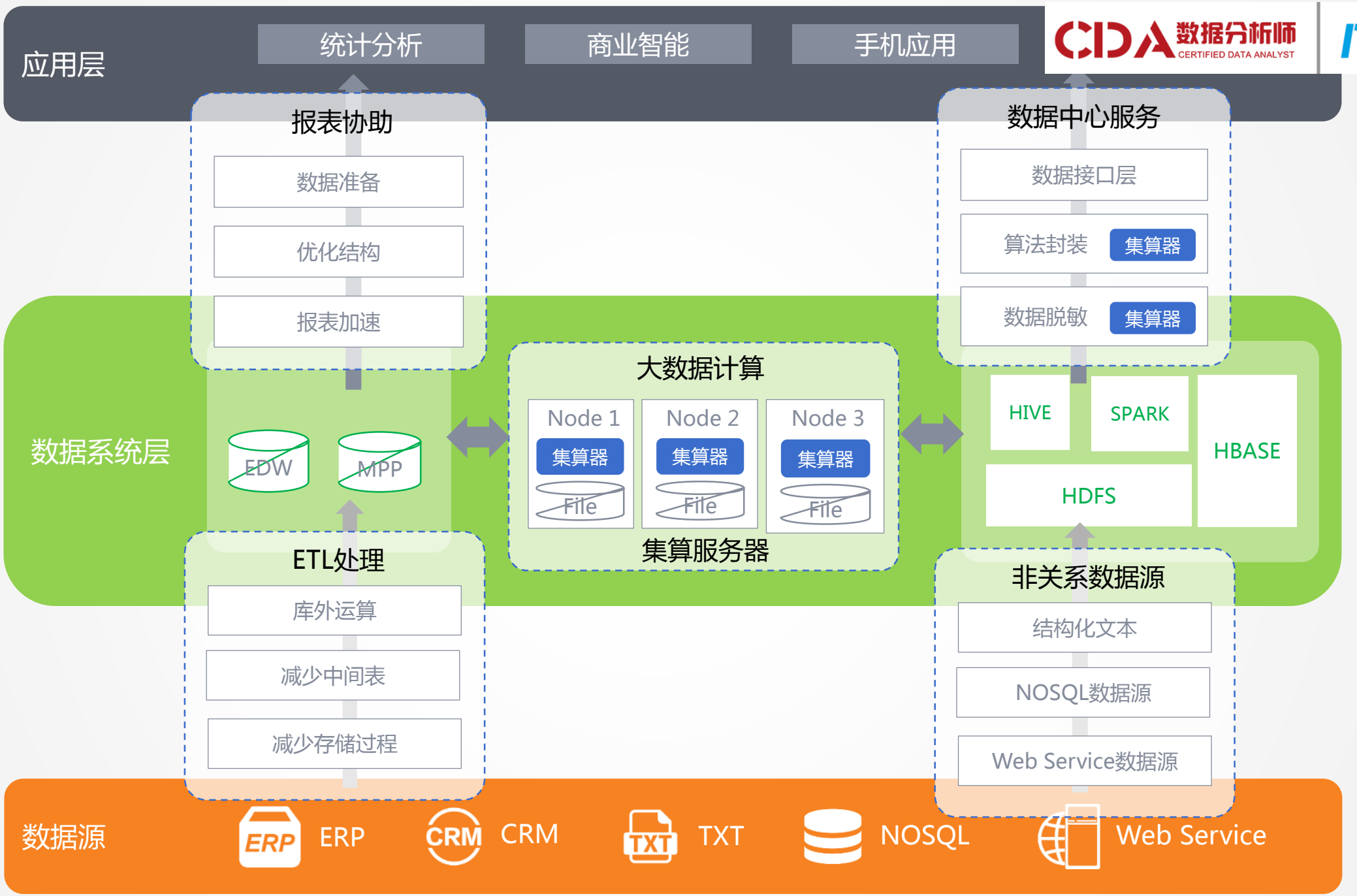
有序归并

同维表和主子表连接可以先排序后变成有序归并

追加数据的再排序也仍然是低成本的归并计算

	A	
1	=file(“Order.txt”).cursor@t()	订单游标，按订单id排序
2	=file(“Detail.txt”).cursor@t()	订单明细游标，也按订单id排序
3	=joinx(A1:O,id;A2:D,id)	有序归并连接，仍返回游标
4	=A3.groups(O.area;sum(D.amount))	按地区分组汇总金额，地区字段在主表中，金额字段在明细子表中

应用场景


应用层
统计分析
商业智能
手机应用
报表协助
数据准备
优化结构
报表加速
数据中心服务
数据接口层
算法封装
集算器
数据脱敏
集算器
数据系统层
EDW
MPP
大数据计算
Node 1
集算器
File
Node 2
集算器
File
Node 3
集算器
File
集算服务器
HIVE
SPARK
HBASE
HDFS
ETL处理
库外运算
减少中间表
减少存储过程
非关系数据源
结构化文本
NOSQL数据源
Web Service数据源
数据源
ERP
CRM
TXT
NOSQL
Web Service

非关系数据源

多样性数据普遍存在

txt,csv,xlsx

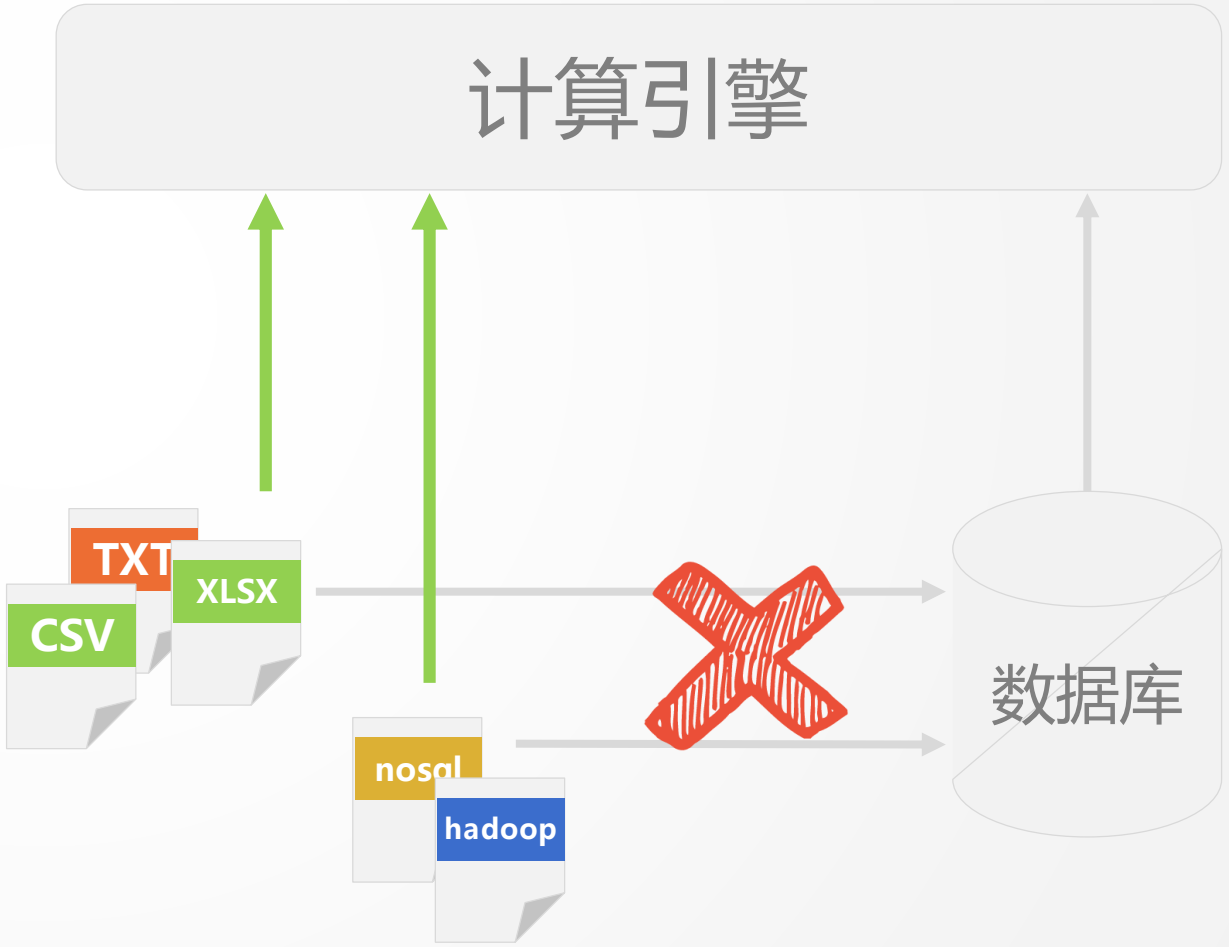
nosql,hadoop

json,xml

直接计算多样性数据

不需要建设专门的数据库及转入工作

结构简单、实时性更好



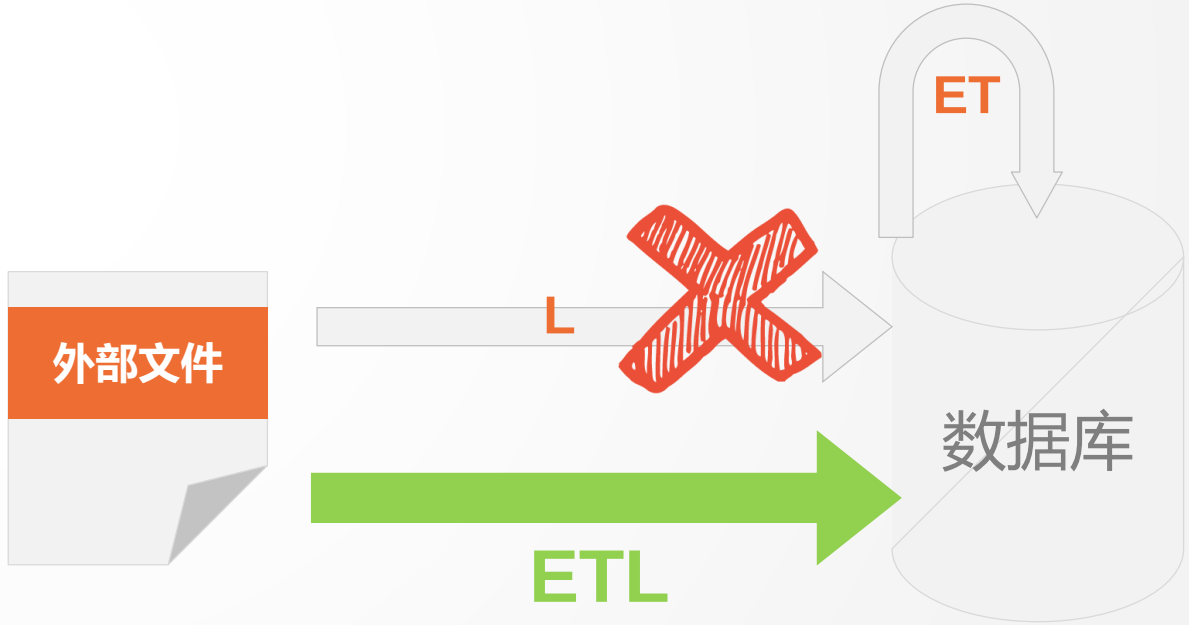
ETL处理

缺乏库外计算能力扰乱ETL过程

ETL？ELT？LET？

加大数据库负担

库外计算实现合理的ETL



协助报表-减少存储过程

存储过程的目的

数据整理

报表查询

存储过程的问题

应用内与应用间耦合

安全性与易管理性

库外计算替代存储过程

协助报表-减少冗余中间表

中间表的由来

运算复杂或数据量大

再次计算的能力

中间表的问题

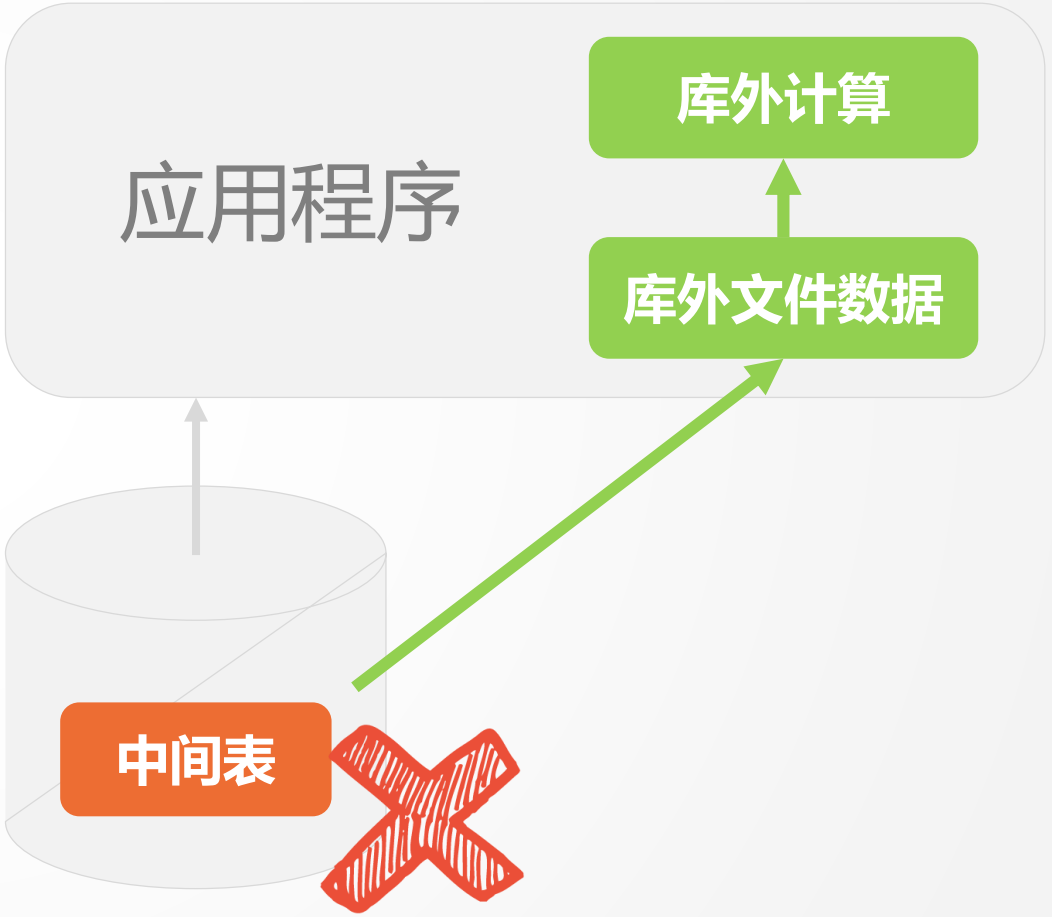
数量众多占用数据库资源

线性结构导致管理困难

库外计算将中间数据外置

计算不依赖于数据库，其它能力不需要

绑定应用、树状结构、易于管理



协助报表-并行取数

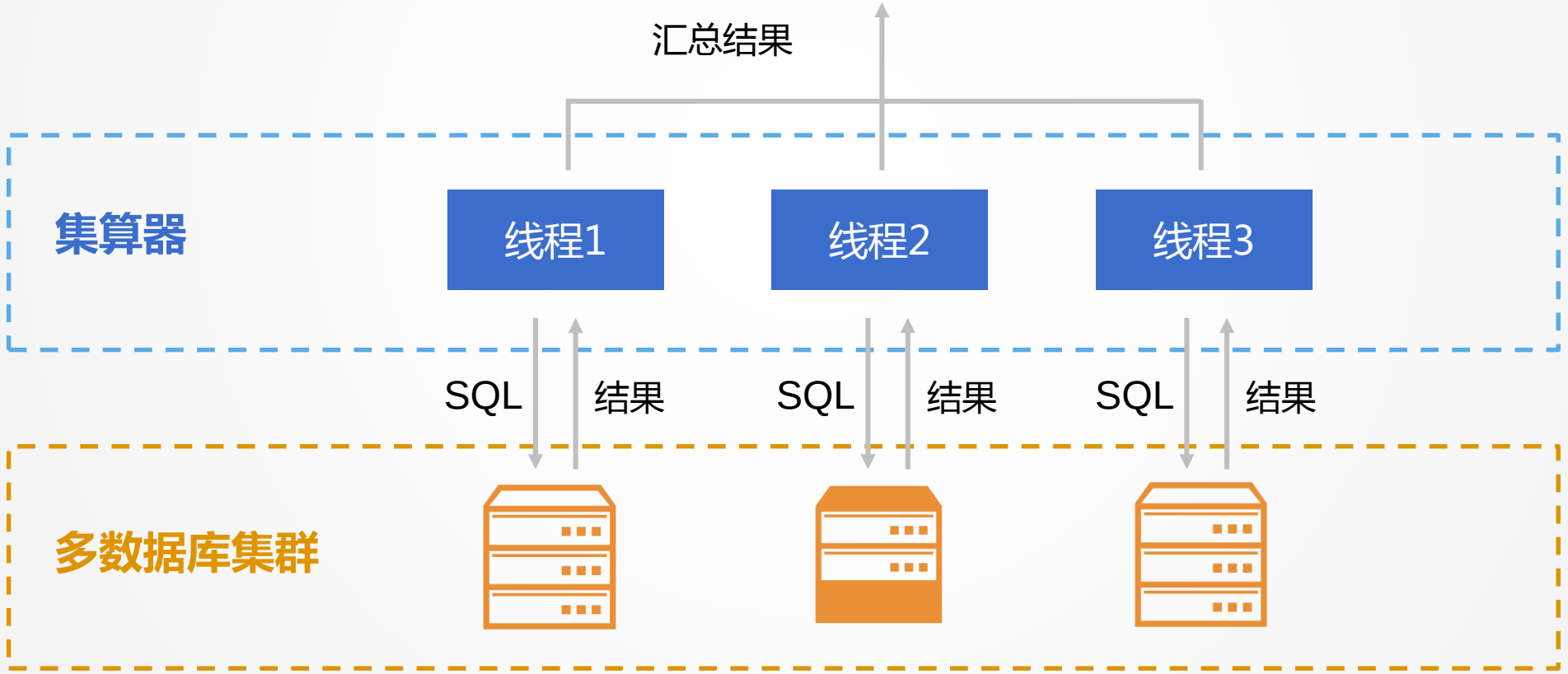
JDBC性能瓶颈

计算引擎多线程取数

	A	B	C
1	fork 4	=connect(db)	/分4线程
2		=B1.query@x(“select * from T where part=?”,A1)	/分别取每一段
3	=A1.conj()		/合并结果

报表协助-跨库计算

异构数据库集群



报表协助-T+0查询报表

T+0问题

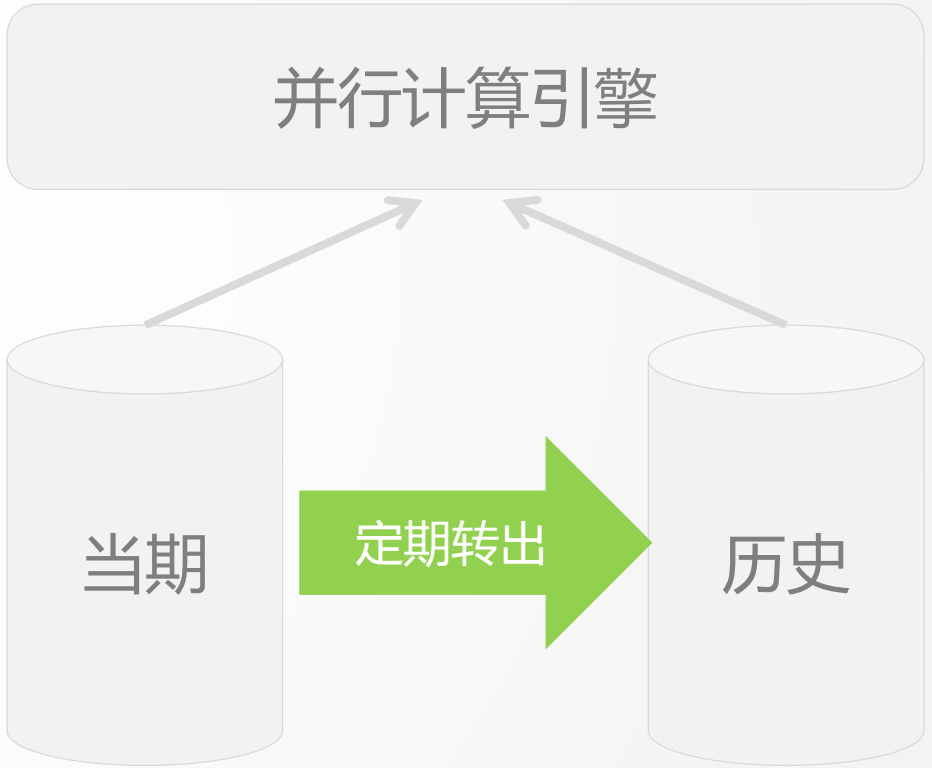
交易一致性要求关系数据库

历史与当期同库，数据量太大

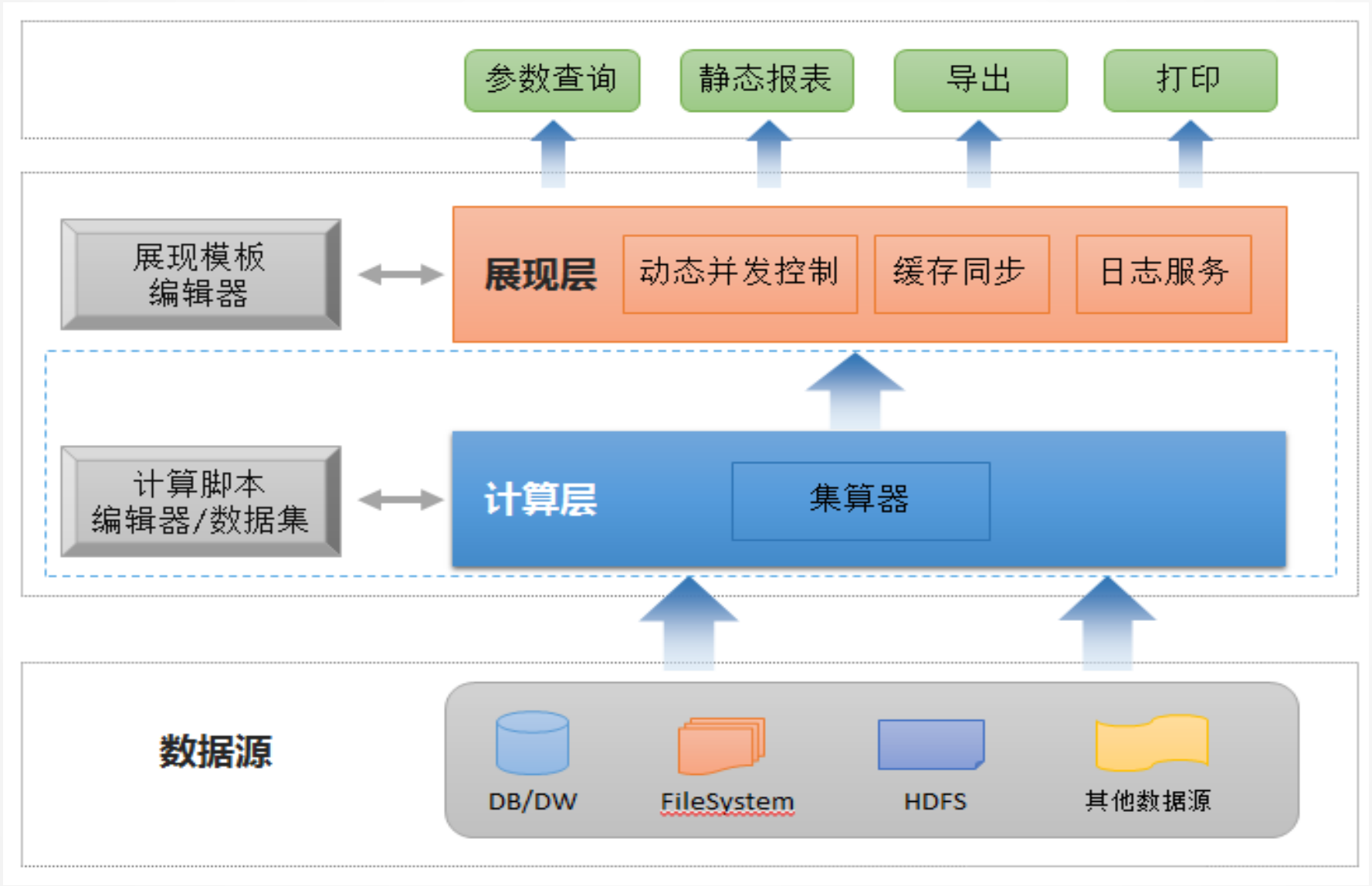
历史与当期异库，跨库计算困难

库外计算实现并行跨库计算

历史数据还可文件化



协助报表-优化应用结构


参数查询
静态报表
导出
打印
展现模板
编辑器
展现层
动态并发控制
缓存同步
日志服务
计算脚本
编辑器/数据集
计算层
集算器
数据源
DB/DW
FileSystem
HDFS
其他数据源

数据中心服务

数据中心的特征与要求

数据库群

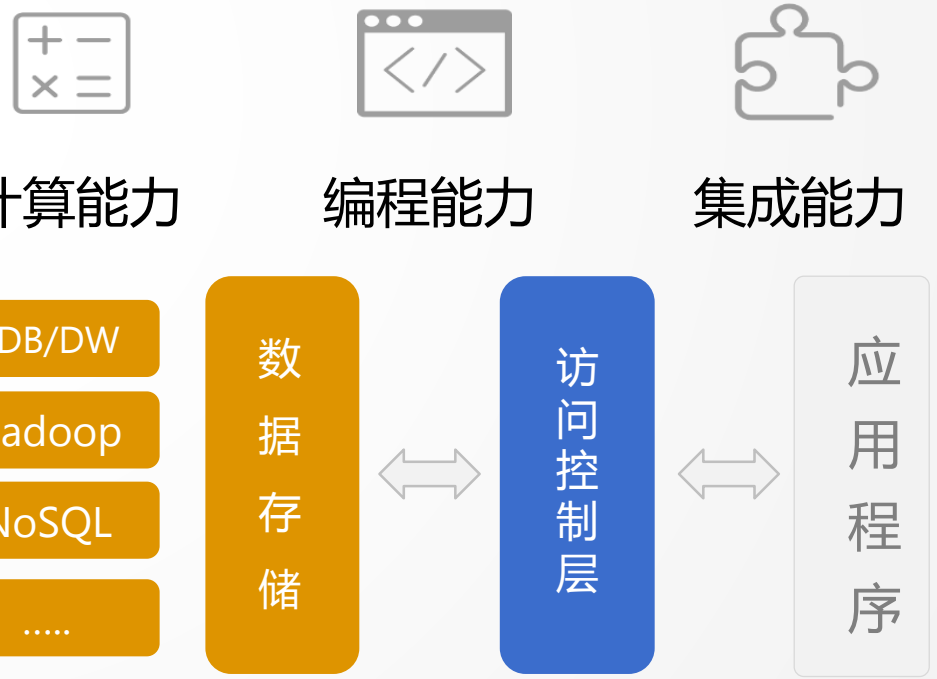
多样性数据源

服务式接口

访问受控

数据脱敏

库外计算引擎实现数据库中心访问层



大数据计算

集算器

创新大数据计算引擎

Hadoop是庞大的重型解决方案，投入了相当多资源解决容错，适合大规模集群

集算器是轻量级产品，适合中小规模集群，不需要太强的容错能力，技术门槛低

MPPSQL

关系数据库存在的问题

硬件成本、实施成本

集算器函数式的编程方式，逻辑过程写起来更贴近人的自然思维，提高开发效率、降低描述业务问题的难度

集算器是中小规模的大数据计算平台

润乾
RAQSOFT

创新技术 推动应用进步

www.raqsoft.com.cn

谢谢大家！