



| DataFun.

数据湖ICEBERG在小米的落地与实践

小米-李培殿



目录 CONTENT

01 Iceberg 技术简介

02 Iceberg在小米
的应用实践

03 基于 Iceberg 的
流批一体的探索

04 未来规划



| DataFun.

01

Iceberg技术简介



Iceberg 简介



Apache Iceberg is an **open table format** for huge analytic datasets.

Iceberg adds tables to compute engines including Spark, Trino, PrestoDB, Flink and Hive using a high-performance table format that works just like a SQL table.

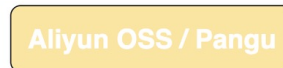
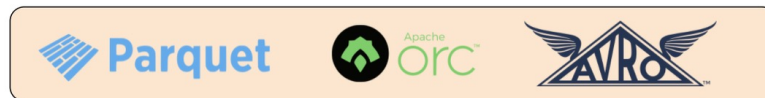


Iceberg 简介

存储与计算分离

计算引擎选择更灵活

屏蔽底层文件存储细节，对外暴露都是一张Iceberg表



文件布局

Metadata(元数据文件)

- 记录了最新的快照信息和历史快照信息，以及最新的Schema信息。

Snapshot(快照文件)

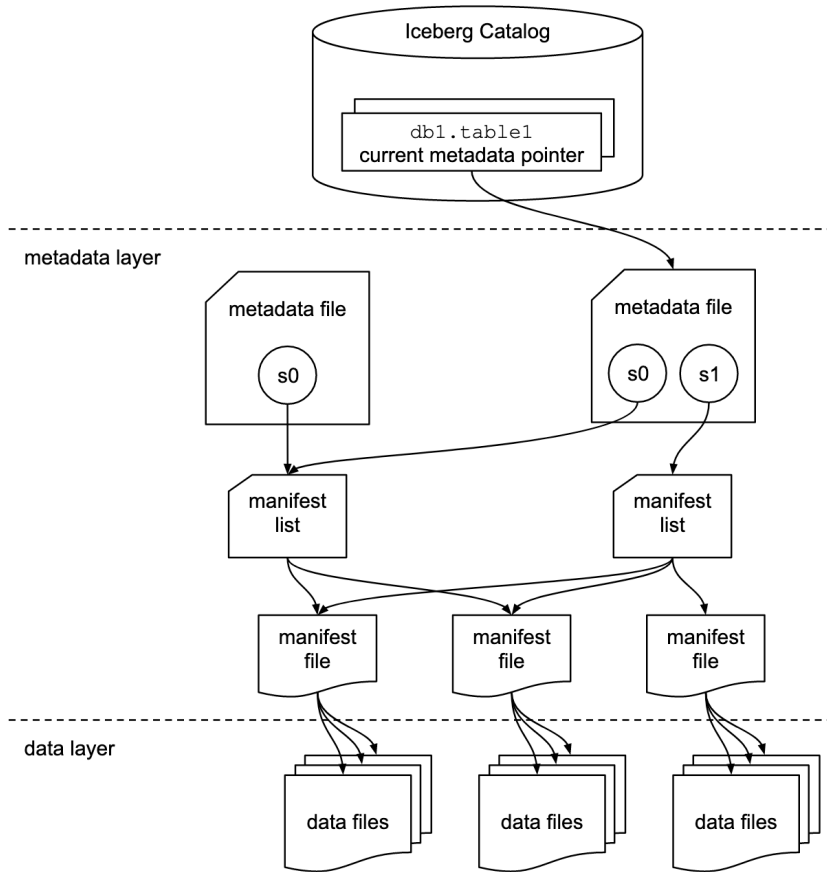
- 每次事务提交都会生成一个Snapshot。记录了本次提交新增的清单文件和历史清单文件列表(Manifest List)。

Manifest(清单文件)

- 记录了本次事务写入的文件和分区的对应关系，以及字段统计信息(最大值、最小值)。

Data File(数据文件)

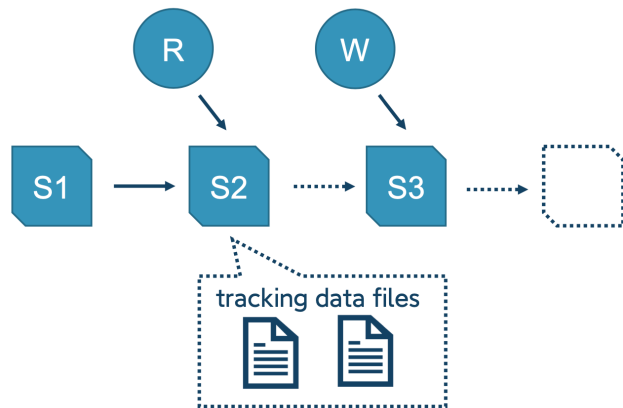
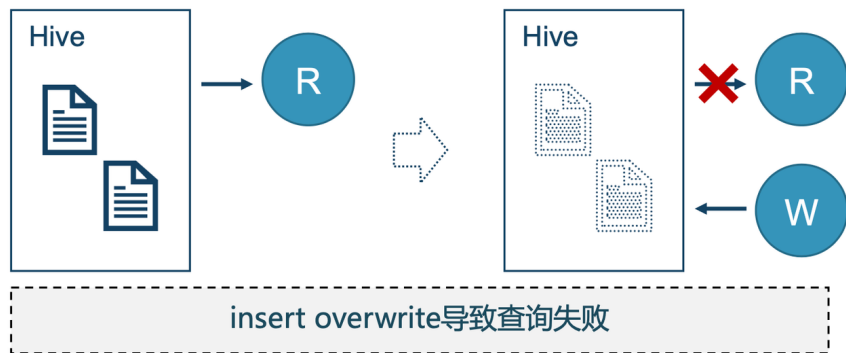
- 实际写入的数据文件，如Parquet、Avro等格式文件。



事务性

避免
脏数据

读写
分离

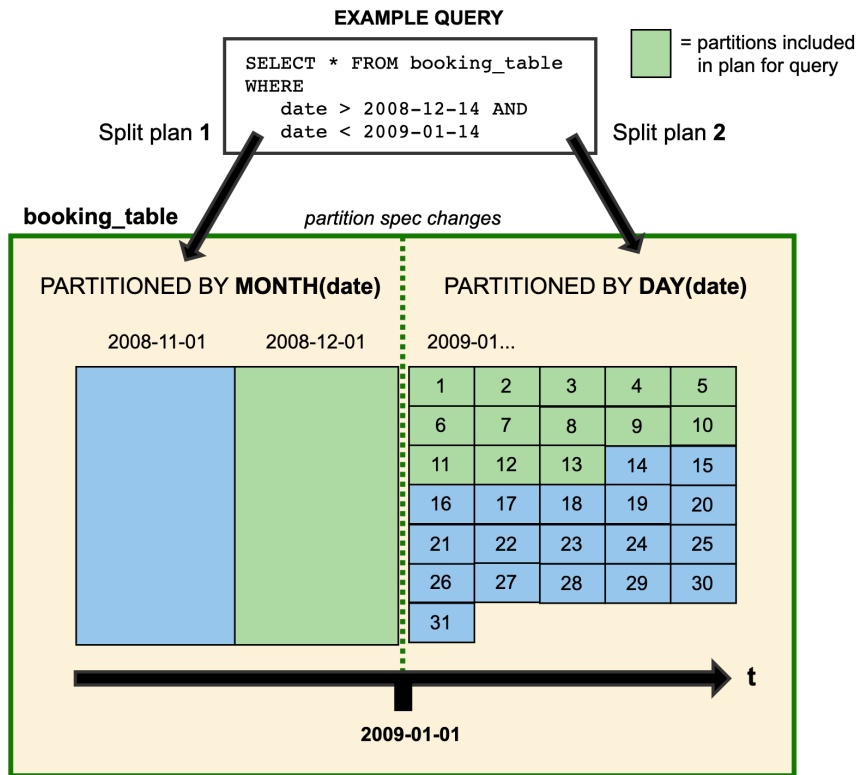


隐式分区

根据数据自动推断分区

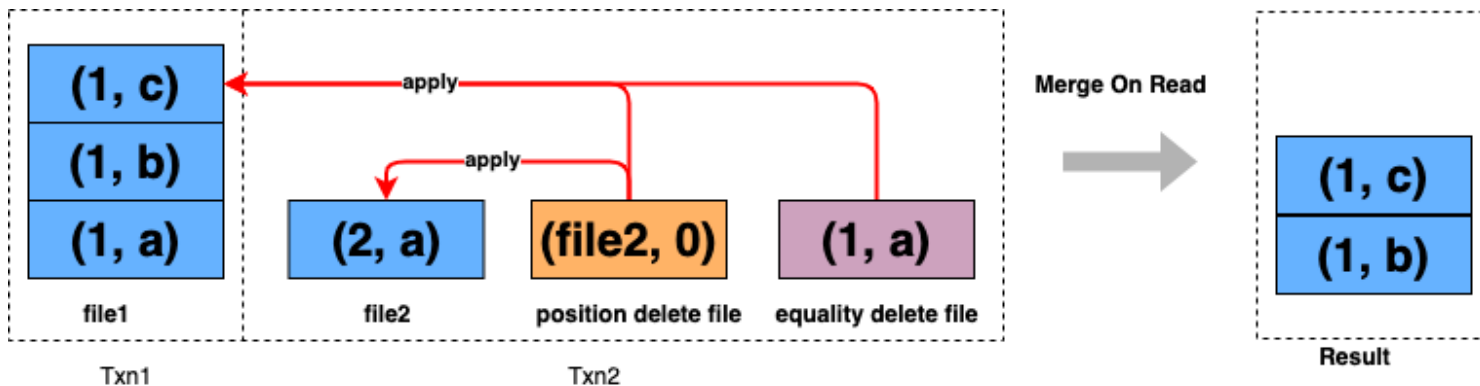
分区不和目录强绑定

灵活的分区变更



行级更新

format version	更新方式
V1	Copy On Write
V2	Copy On Write、Merge On Read





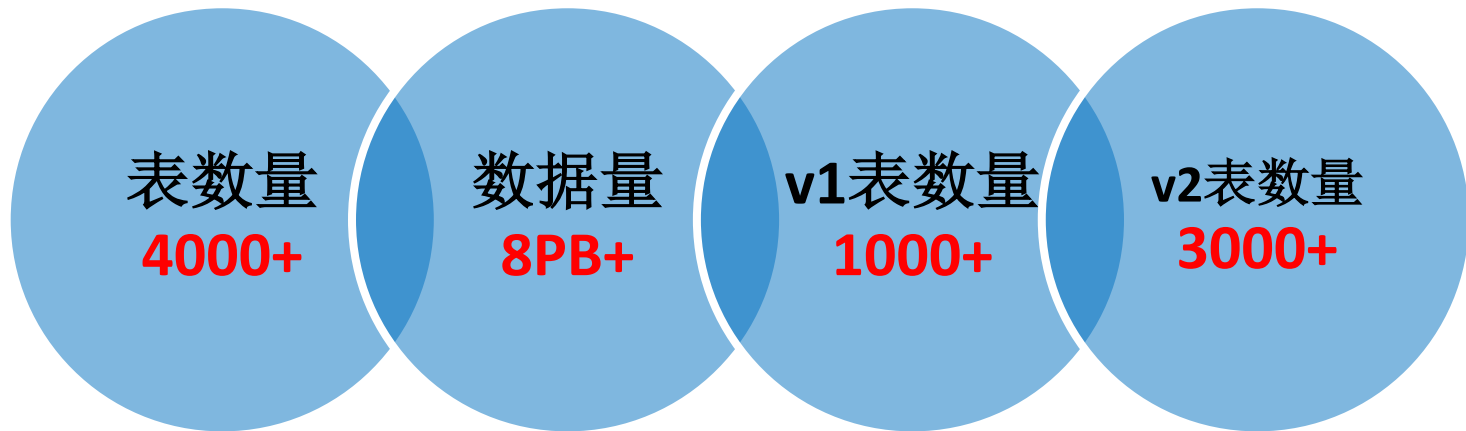
| DataFun.

02

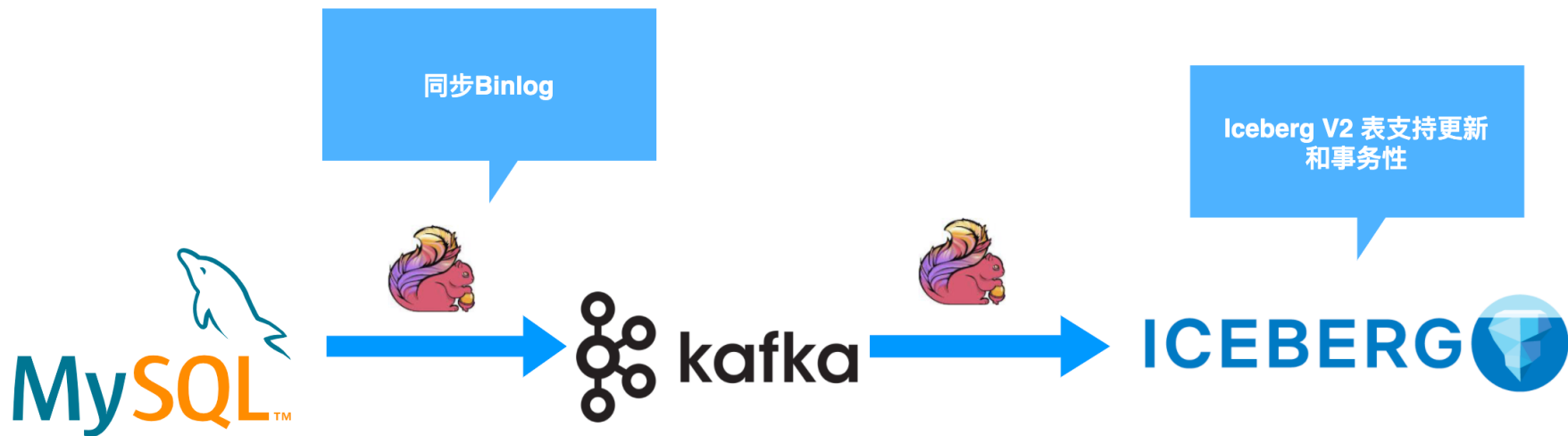
Iceberg在小米的 应用实践



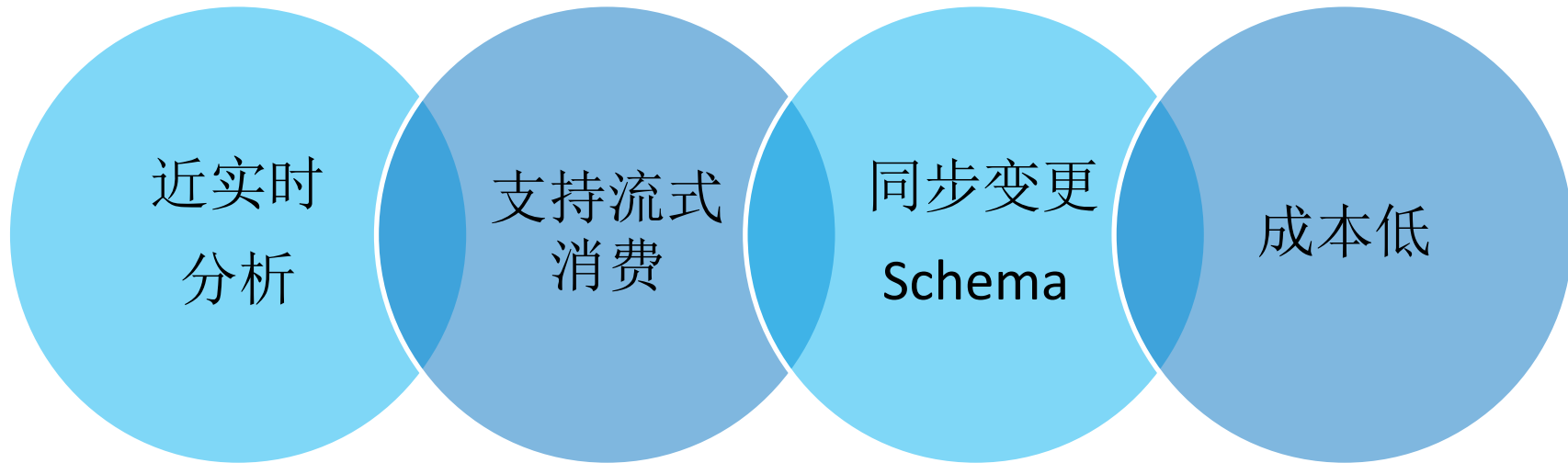
数据湖在小米应用现状



ChangLog 数据入湖



ChangLog 数据入湖优点



ChangLog 入湖分区的选择-自增 Id 为主键

Bucket 分区

- 数据均匀分布
- 所有分区都可能需要 Compaction
- 随着数据量增加，分区数不变

id_bucket=0

id_bucket=1

id_bucket=2

id_bucket=3

Truncate 分区

- 数据只写入最近几个分区
- 只对最近几个分区 Compaction
- 随着数据量增加分区数递增

id_trunc_1000000=0

id_trunc_1000000=1000000

id_trunc_1000000=2000000

id_trunc_1000000=3000000



ChangeLog 数据入湖产品化

映射关系

源表MySQL: tbl_...

Catalog集群:mysql_..._sale 库名 sale

字段名称(33)	字段类型	字段描述
id	INT(10) UNSIGNED	自增主键
order_id	BIGINT(19)	
site_id	INT(10) UNSIGNED	
com_id	BIGINT(19)	
customer_id	BIGINT(19)	
org_id	VARCHAR(20)	
code	VARCHAR(80)	
type_id	INT(10)	
com_type_id	BIGINT(19)	
com_type_name	VARCHAR(100)	
amount	DECIMAL(12, 2) UNSIGNED	
duce	DECIMAL(12, 2)	

目标表Iceberg: tbl_...

Catalog集群:iceberg_zjyprc_hadoop 库名 mysql

字段名称(33)	字段类型	字段描述	指定key
id	int		否
order_id	long		是
site_id	int		否
com_id	long		否
customer_id	long		否
org_id	string		否
code	string		否
type_id	int		否
com_type_id	long		否
com_type_n...	string		否
amount	decimal(12,2)		否
duce	decimal(38,18)		否



日志数据入湖



隐式分区避免数据漂移问题



隐式分区保证延迟数据正确分区



Flink + Iceberg 事务性保证数据不丢不重




支持 Schema 同步变更

映射关系

源表Talos: 10.10.10.100

Catalog集群:talos 库名:default

字段名称(11)	字段类型	字段描述
time	string	
did	string	
type	string	
msg	string	
host	string	
	string	
	string	
	string	
pdid	string	
logid	string	
DATE_FORMAT(talos...	EXPRESSION	时间分区

目标表Iceberg: d[REDACTED] →

Catalog集群:iceberg 库名: t

字段名称(11)	字段类型	字段描述
time	string	消息创建时间(毫秒)
did	string	设备ID
type	string	消息类型
msg	string	消息内容
host	string	主机名
...	string	...
...	string	...
c	string	...
pdid	string	...
logid	string	...
date	int	日期

Iceberg-governance 服务

Compaction 服务

- 合并小文件、
merge delete files

Expire Snapshots 服务

- 过期 snapshots

Orphan Files clean 服务

- 清理孤儿文件



Hive 升级 Iceberg

压缩方式	存储格式	Parquet + ZSTD 存储节约
UNCOMPRESSED	TEXT	80%
SNAPPY	SequenceFile	30%+
SNAPPY	Parquet	30%+
GZIP	Parquet	5%

- ✓ 在 Compaction 中配置更高的 compression level 获得更高的压缩率

Hive 升级 Iceberg 产品化

查看数据地图→

Catalog集群: hive op 库名: tmp

0人已收藏

☆ 收藏

[查看新链路](#)

...

预览数据

表信息

分区记录

HDFS目录

链路升级

变更记录

数据同步

选择分区范围:




开始日期

→

结束日期

* 此处停

批量同步

Hive表分区	Hive分区大小	Iceberg表分区	Iceberg分区大小	同步状态 查看同步作业	操作
date=20220502		date=20220502		已完成	开始同步
date=20220501		date=20220501		已完成	开始同步

共2条

<

1

>

10 条/页 ▾

写入作业升级 [查看完整血缘→](#)

Hive写入作业	作业类型	所属工作流	负责人	Iceberg写入作业	作业类型	操作
----------	------	-------	-----	-------------	------	----

新五数据



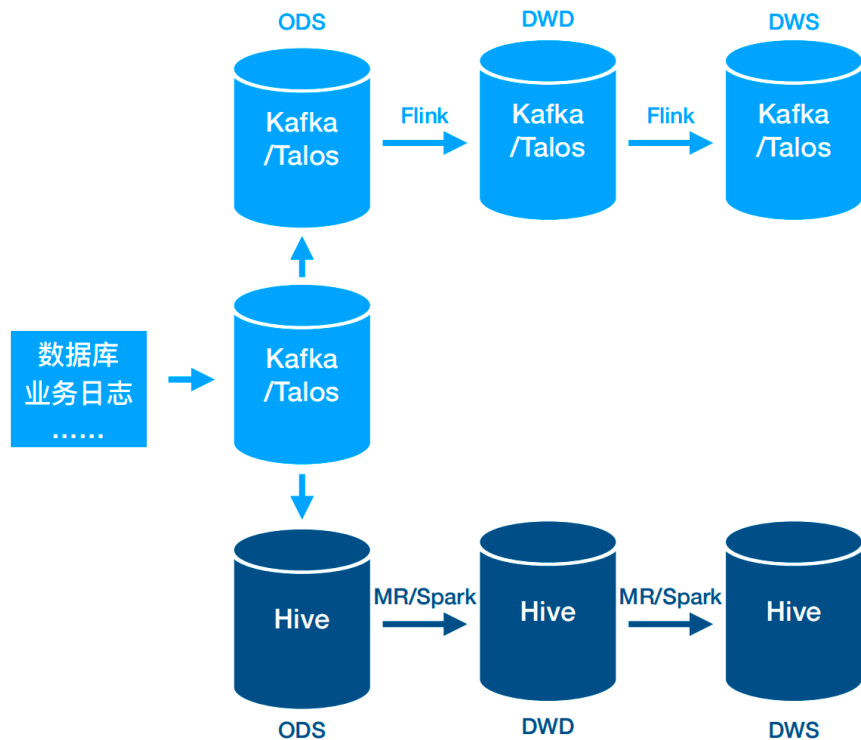


03

基于 Iceberg 的 流批一体的探索



Lambda 架构

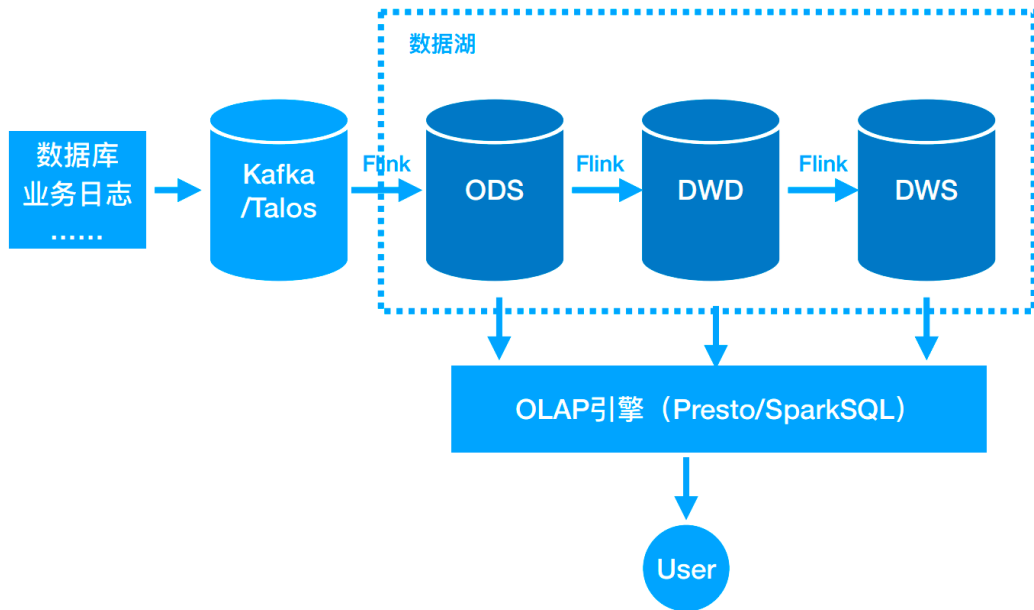


- 实时链路提供时效性
- 离线链路提供准确性
- 离线数据支持回溯
- 离线数据可供OLAP查询



- 实时链路不支持OLAP 查询
- 实时链路回溯能力有限
- 两套存储，存储成本高
- 两套代码，开发维护成本高
- 实时离线数据不一致

数据湖架构



- ✓ Iceberg 存储上统一
- ✓ Flink 计算引擎统一
- ✓ 支持回溯
- ✓ 支持 OLAP 查询
- ✓ 支持构建变更流

为什么需要离线作业来修数据？

Flink 状态过期导致没 Join 上

Watermark 设置导致延迟数据丢失

Lookup Join 完成后维表发生了变更



Overwrite VS Merge Into ?

```
MERGE INTO prod.db.target t -- a target table
USING (SELECT ...) s        -- the source updates
ON t.id = s.id AND t.date=20220601 -- condition to find updates for target rows
WHEN MATCHED AND s.op = 'delete'
THEN DELETE
WHEN MATCHED AND t.count IS NULL AND s.op = 'increment'
THEN UPDATE SET t.count = 0
WHEN MATCHED AND s.op = 'increment'
THEN UPDATE SET t.count = t.count + 1w
WHEN NOT MATCHED
THEN INSERT *
```

Overwrite VS Merge Into ?

Overwrite

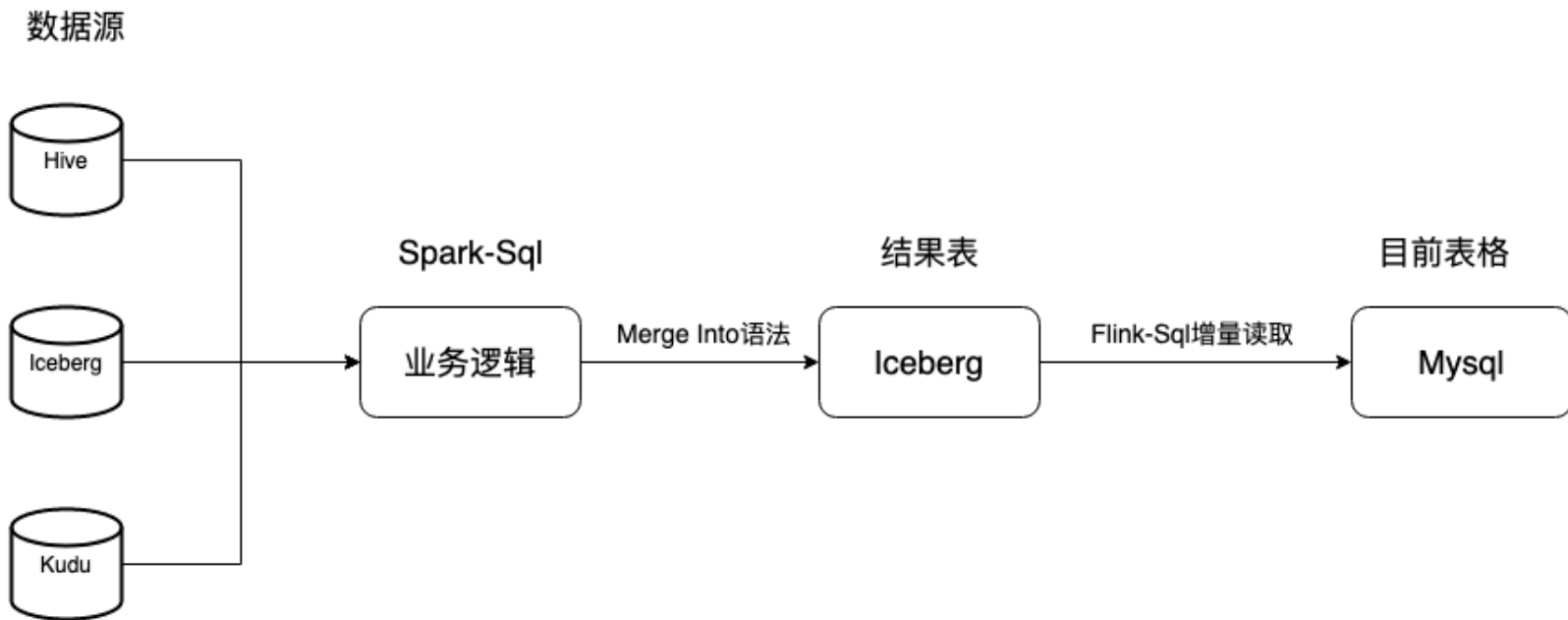
- 分区覆盖
- 语法简单
- 性能好
- 可能导致下游消费波动

Merge Into

- 增量更新
- 语法复杂
- 性能不如 **overwrite**
- 下游只消费变更数据



Merge Into 增量同步数据



✓ Merge Into 更新 Iceberg, Flink 同步变更至下游

不同的分区带来的问题

ProcessTime 分区

- 实时总是写入 T 分区
- 离线修正 T-1 分区
(Overwrite)
- ✓ 实时离线处理的数据
无交集

EventTime 分区

- 实时写入对应分区
- 离线修复历史全量
(Merge Into)
- ✓ 实时离线处理的数据
存在交集

Merge Into 的隔离级别带来的问题

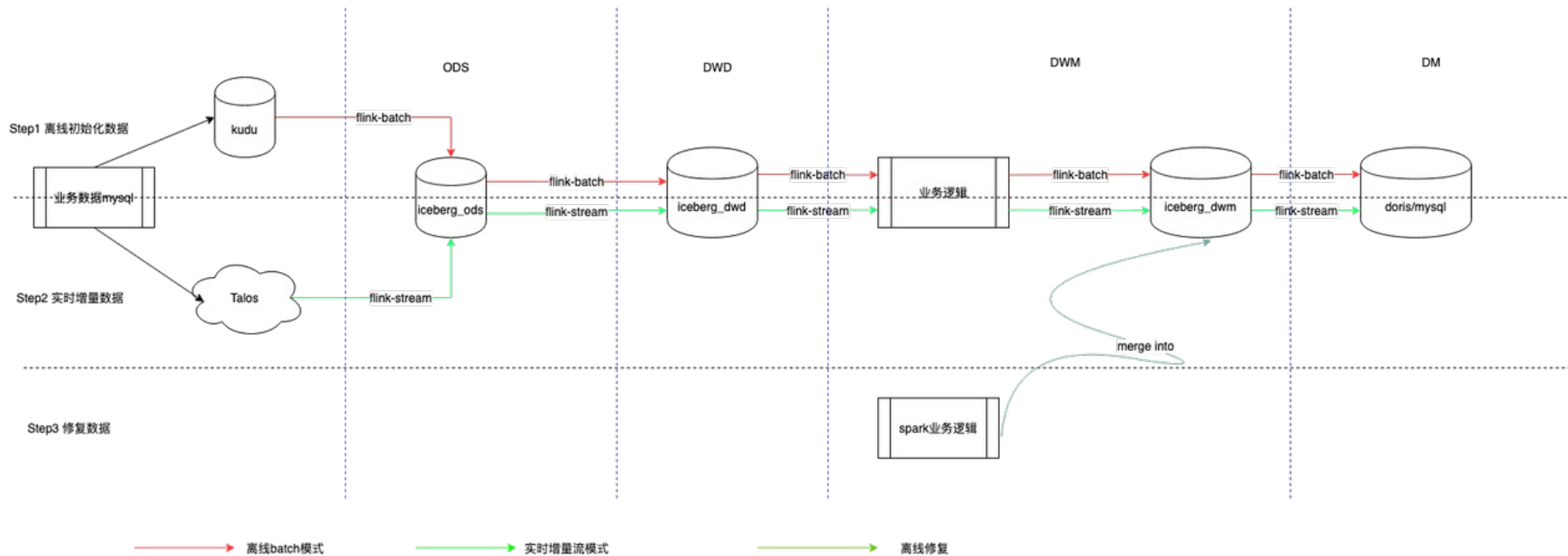
Serializable isolation level

- 与提交过程中其他已成功提交事务冲突则本次提交失败
- 离线作业失败概率增加
- ✓ 使用 **Serializable isolation level** , 实时只处理 T 分区的变更, T-N 分区变更由离线修正

Snapshot isolation level

- 覆盖提交过程中其他已提交事务的更新
- 丢失中间事务的更新

构建流批一体的链路



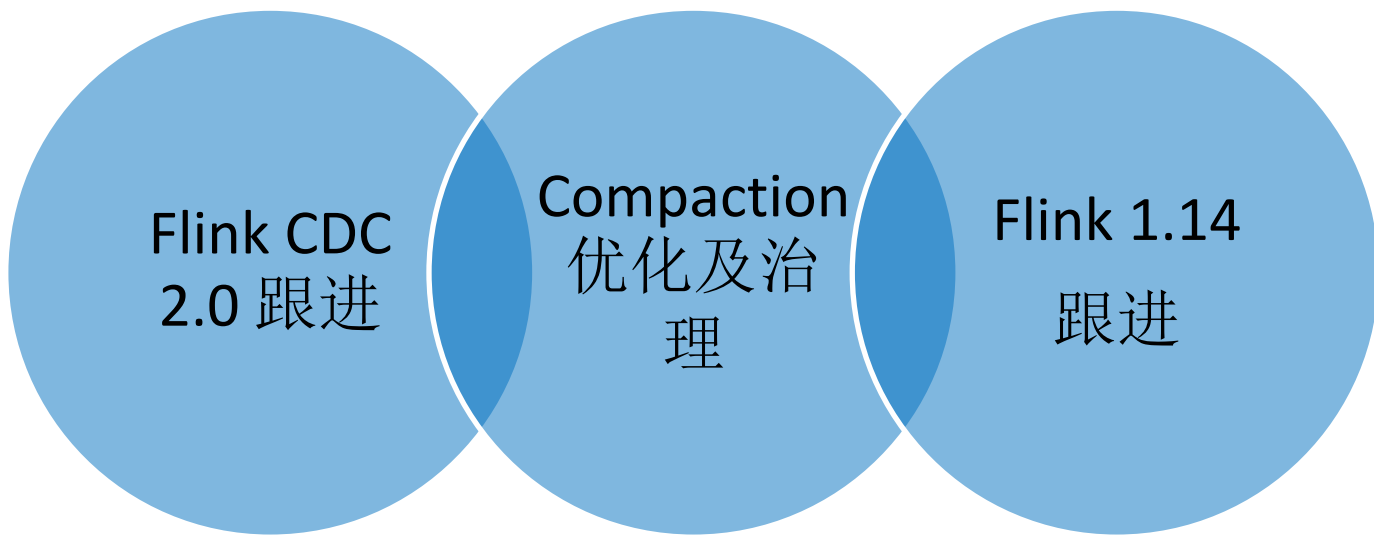


04

未来规划



未来规划



非常感谢您的观看



| DataFun.

