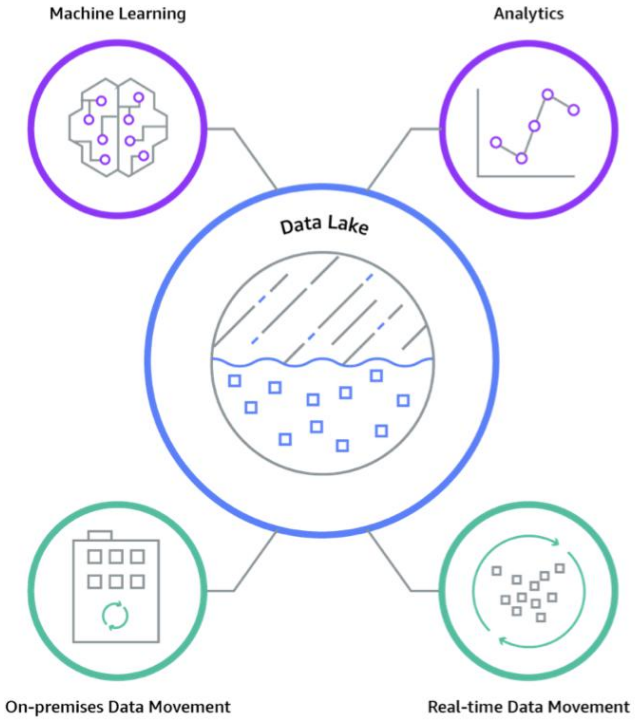
滴普湖仓一体架构探索与实践

吴小前

背景

什么是数据湖

AWS的定义：数据湖提供统一可扩展性的存储，保存结构化与非结构化数据。通常，这些数据以原始形式保存，自此之上，部署各种计算与分析引擎，从而挖掘数据价值。

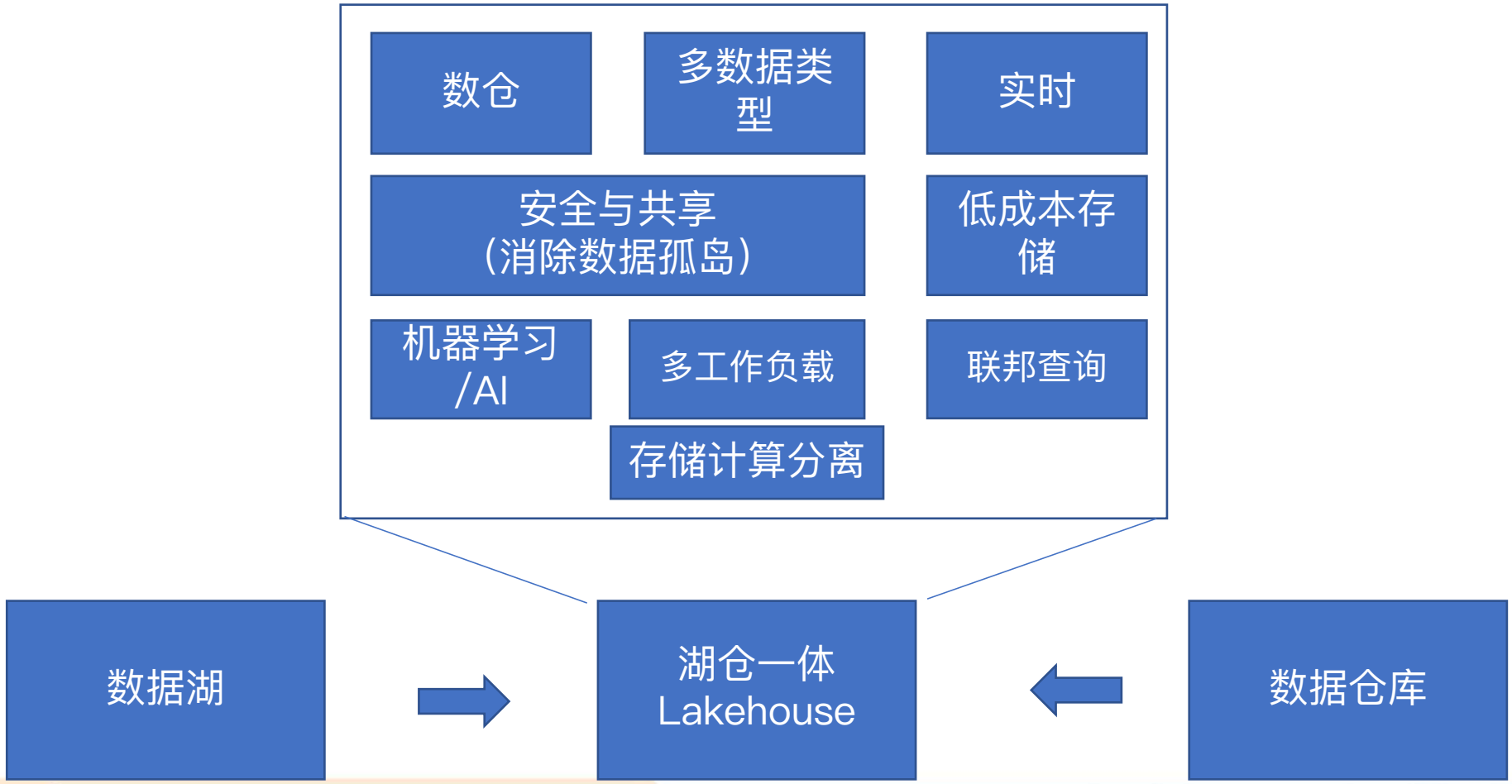


- 多工作负载（workload）
  - 流处理
  - 批处理
  - 机器学习
  - 交互式分析
- 多模数据，统一存储
  - 结构化
  - 半结构化
  - 非结构化
  - 二进制数据
- 数据管理
  - 安全与管控
  - 多种数据接入方式

数据仓库与数据湖对比

数据湖	vs	数据仓库
启动成本低 管理成本高	成本	启动成本高 易管理
结构化/半结构化/非结构化/二进制	数据类型	结构化/半结构化
流批处理，AI，数据挖掘和探索	计算负载	流批处理，交互式分析，BI，报表
Schema-on-read，安全与管控方面难度大	数据治理	数据质量高，易管理

数据平台向湖仓一体发展



构建lakehouse非易事

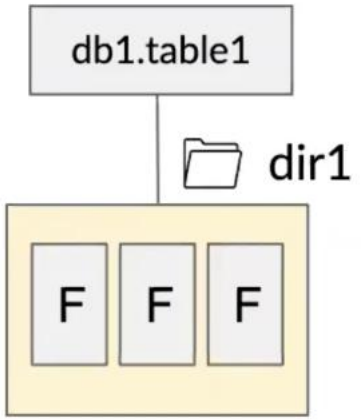
- 存储层
- 表存储引擎
- 数据处理
  - Spark，Flink，MR等
- 数据分析
  - SQL，BI and reporting， ML等
- 数据管控以及安全

数据湖的问题

- 容易出现数据不一致
  - 数据追加或者更新都是件麻烦的事情
  - 处理作业发生问题，可能出现不完整的数据
- 无法支持并发读写
  - 批处理与实时处理共存的需求
  - 多计算引擎数据访问需求难以协同
- 运维问题
  - 大量小文件
  - 数据质量问题

Hive Table的问题

- 数据访问和修改低效
  - HMS以及元数据设计的问题
  - 大量目录访问
- 跨分区和多作业并发修改会出现数据不一致
- 分区操作复杂，使用者深度参与
  - WHERE event_ts >= ‘2021-05-10 12:00:00’ AND event_year >= ‘2021’ AND event_month >= ‘05’ AND (event_day >= ‘10’ OR event_month >= '06')
- 统计成本过高
  - CBO



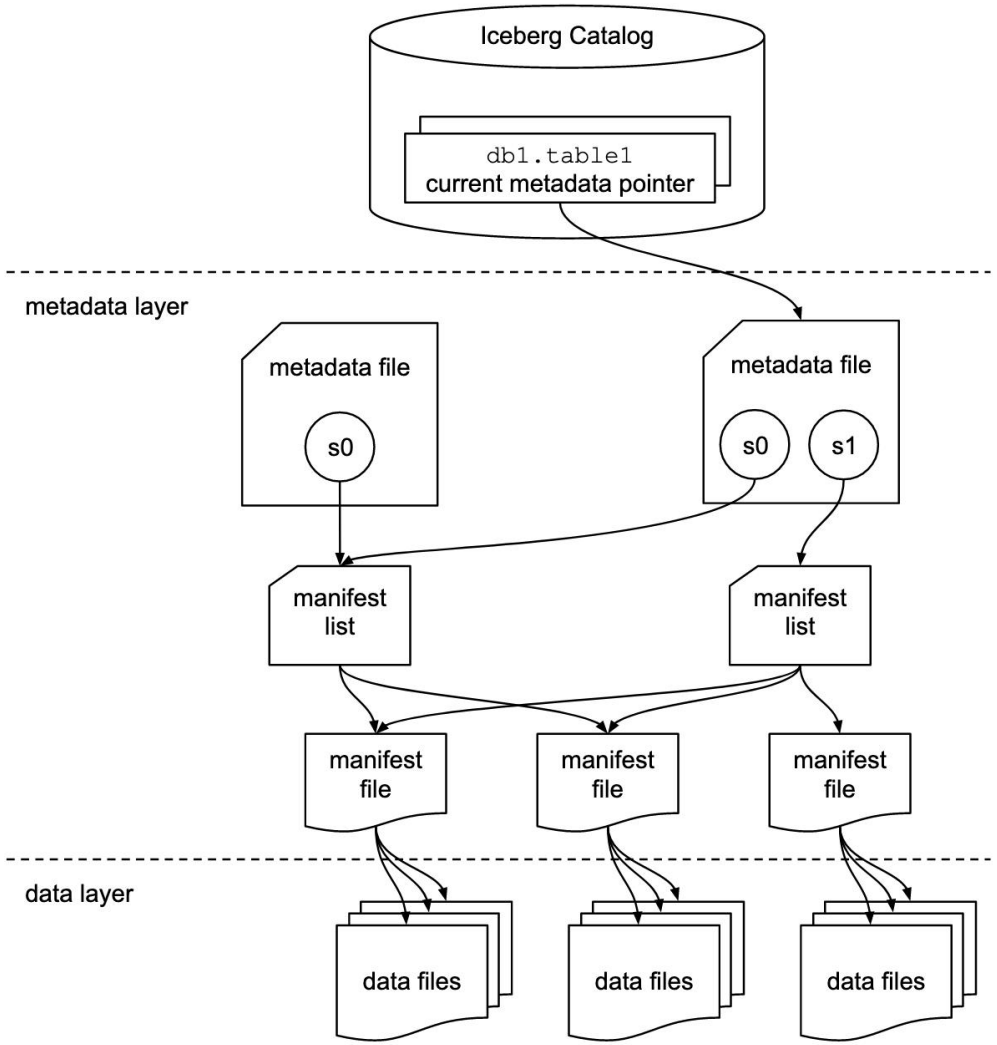
表内容保存在目录中

Apache Iceberg

Iceberg：3种开源解决方案之一

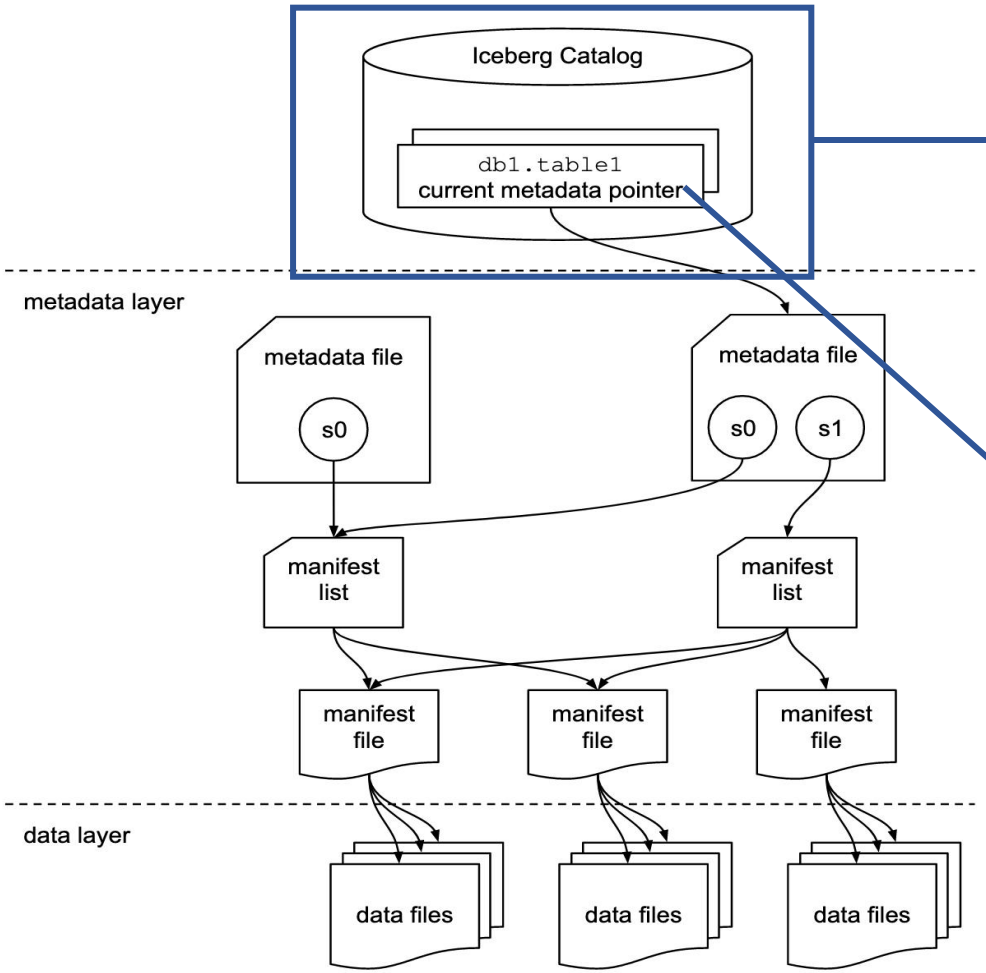
Apache Iceberg is an open table format for huge analytic datasets.

- 支持事务隔离(ACID)
  - 多任务和多引擎并发读写
- 近实时，且能够胜任小批更新
- 支持历史版本
- 隐式分区以及Partition变更



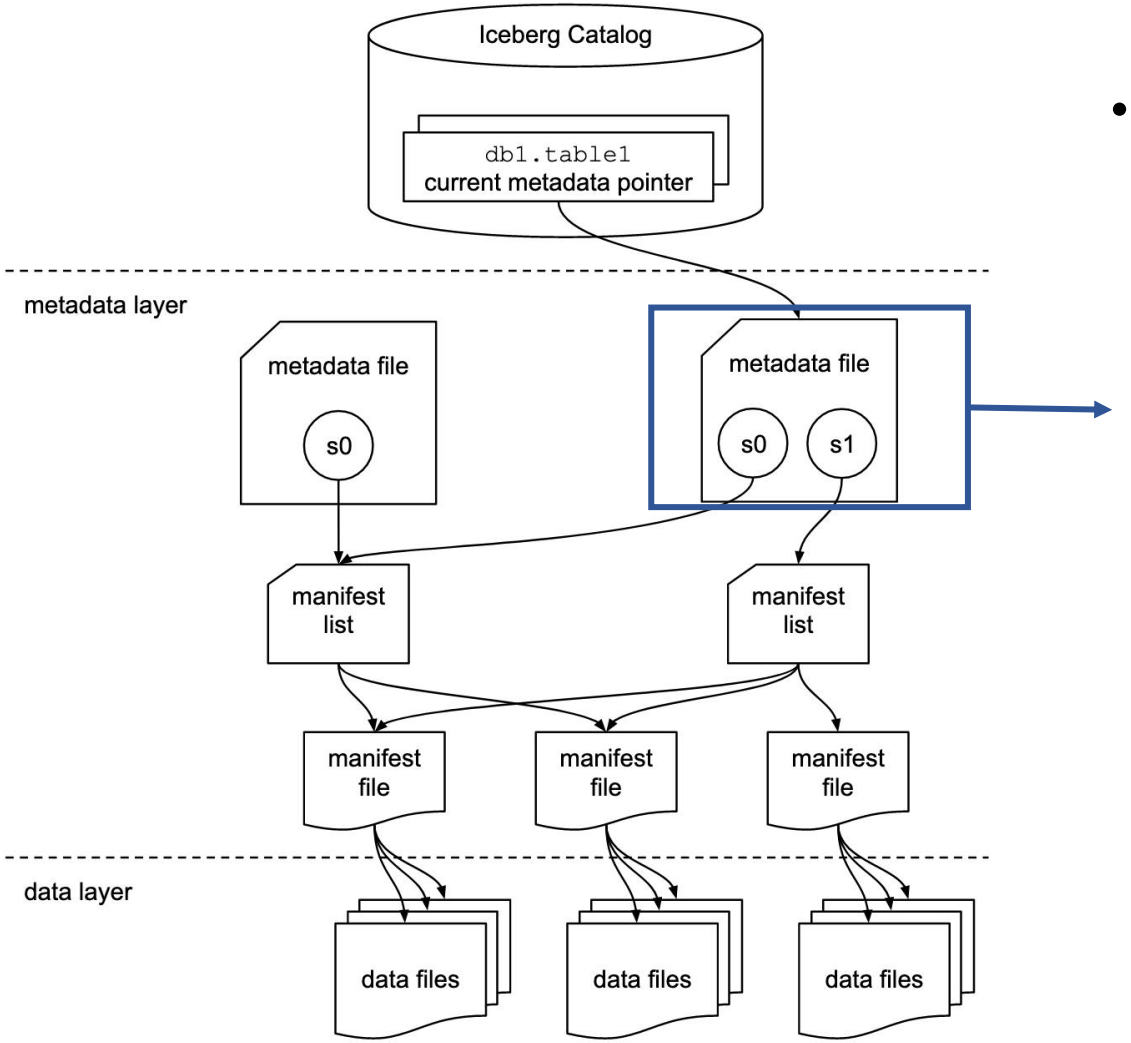
图片来源：iceberg官网

Iceberg catalog



- Iceberg Catalog
  - 保存“当前元数据指针”的地方
  - 修改此指针时需要原子操作
- Table1的当前指针
  - 这个“指针”记录table1的元数据在文件系统的位置

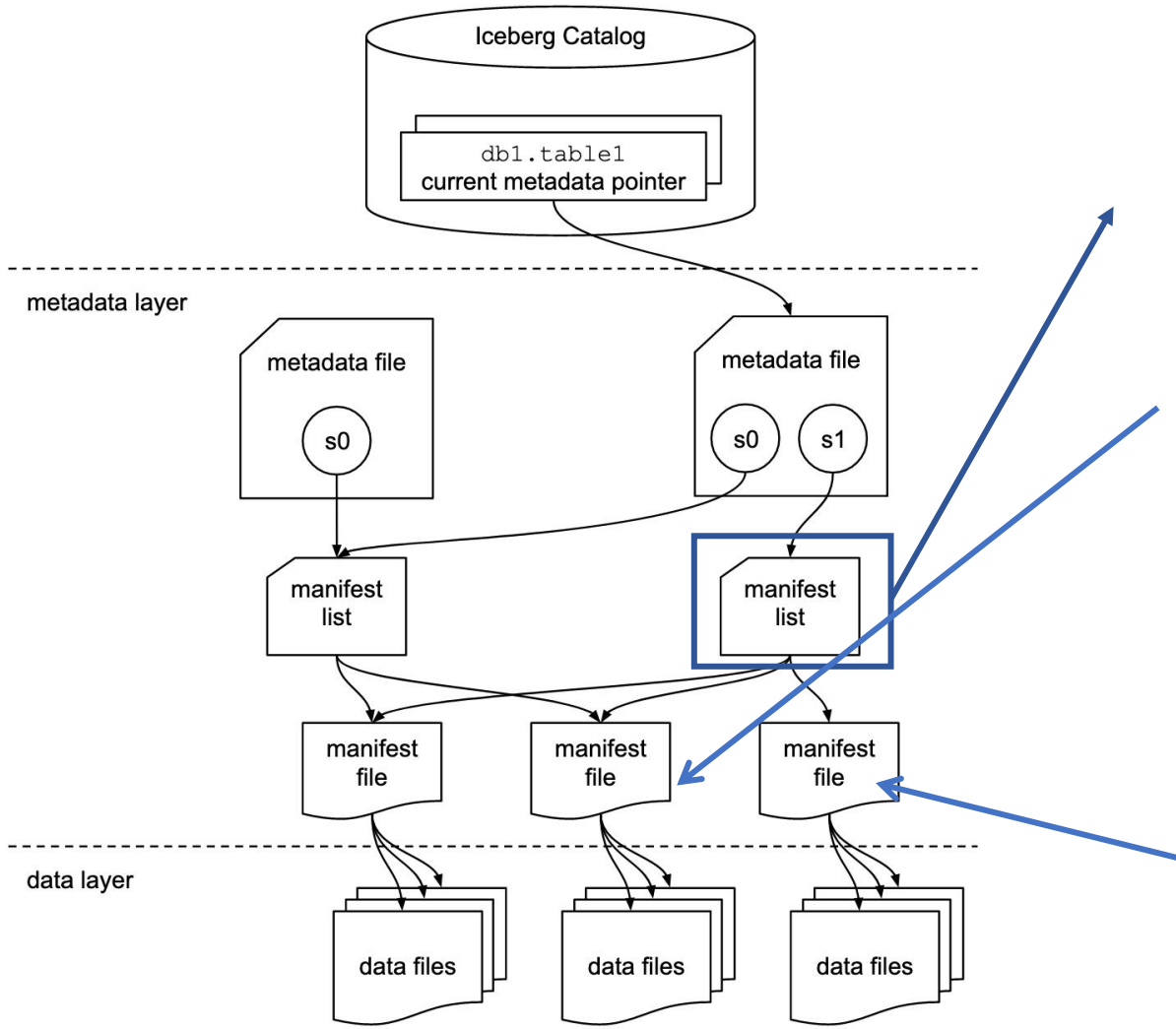
元数据文件



- **Metadata file保存基于timeline的元数据**

```
{
    "table-uuid": "<uuid>",
    "location": "/path/to/table/dir",
    "schema": {...},
    "partition-spec": [{partition-details}, ...],
    "current-snapshot-id": <snapshot-id>,
    "snapshots": [{
        "snapshot-id": <snapshot-id>,
        "manifest-list": "/path/to/manifest/list.avro"
    }, ...],
}
```

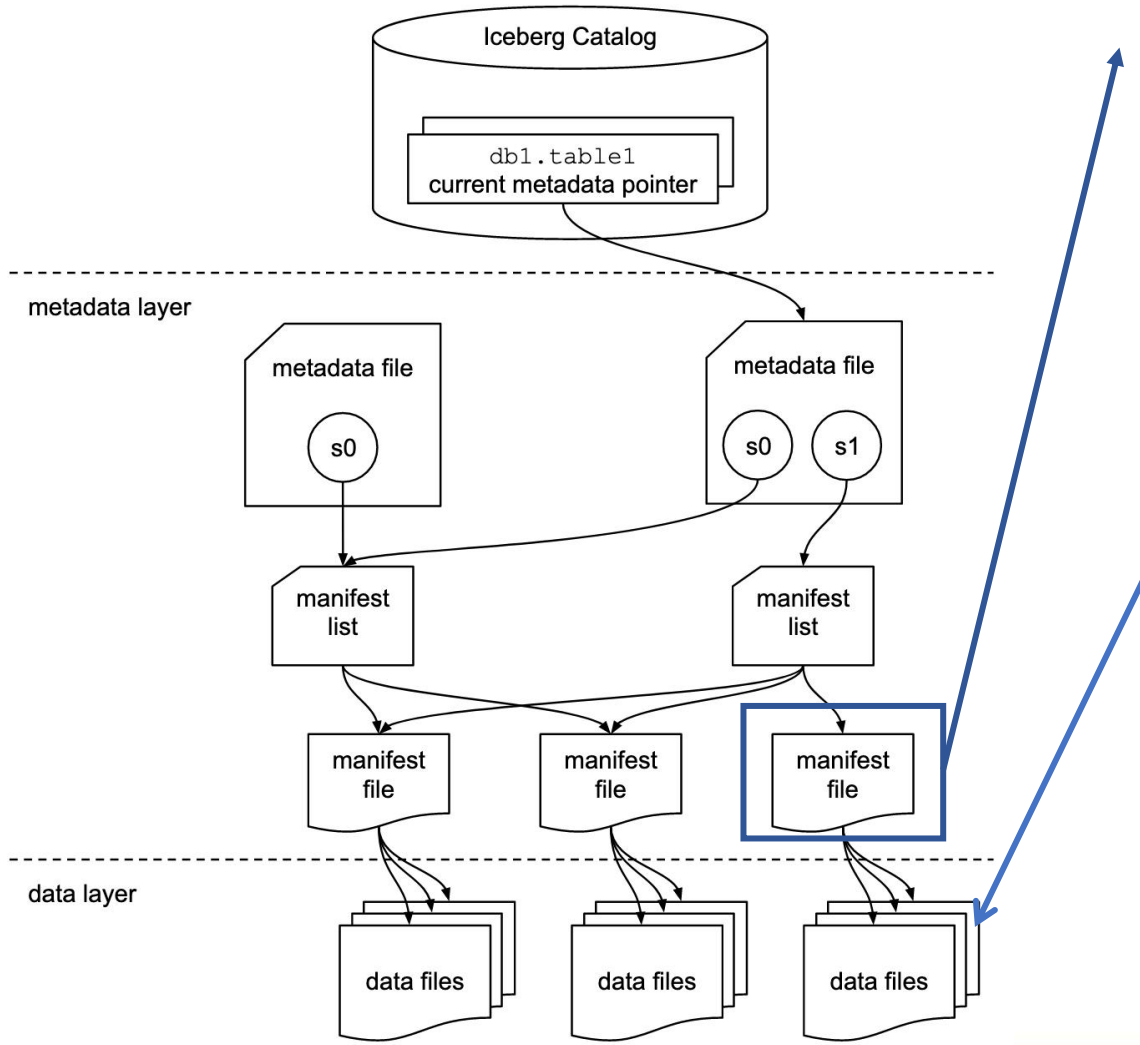
Manifest list



- **Manifest list**

```
{
    "manifest-path":
        "/path/to/manifest/file.avro",
    "added-snapshot-id": <snapshot-id>,
    "partition-spec-id": <partition-spec-id>,
    "partitions": [{partition-info}, ...],
    ...
}
{
    "manifest-path":
        "/path/to/manifest/file2.avro",
    "added-snapshot-id": <snapshot-id>,
    "partition-spec-id": <partition-spec-id>,
    "partitions": [{partition-info}, ...],
    ...
}
```

Manifest文件

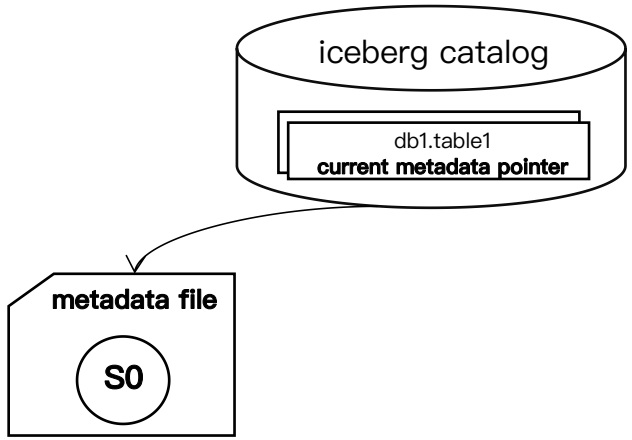


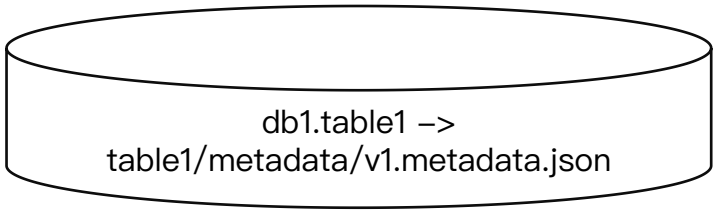
- **Manifest文件：保存数据文件列表，以及每个文件的统计信息等**

```
{
    "data-file": {
        "file-path": "/path/to/data/file.orc",
        "file-format": "ORC",
        "partition": {"<partition-field>":{"<data-type>": <value>}},
        "record-count": <num-record>,
        "null-value-counts": [{
            "column-index": "1", "value": 4
        }, ...],
        "lower-bounds": [{
            "column-index": "1", "value": "xxx"
        }, ...],
        "upper-bounds": [{
            "column-index": "1", "value": "yyyy"
        }, ...]
    }
    ....
}
```

创建表

```
CREATE TABLE db1.table1 (
    order_id bigint,
    customer_id bigint,
    order_amount DECIMAL(10,2),
    order_ts TIMESTAMP
)
PARTITIONED BY (hour(order_ts));
```

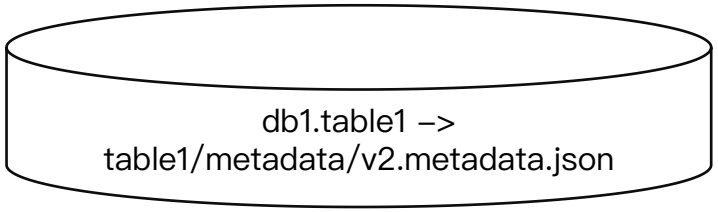




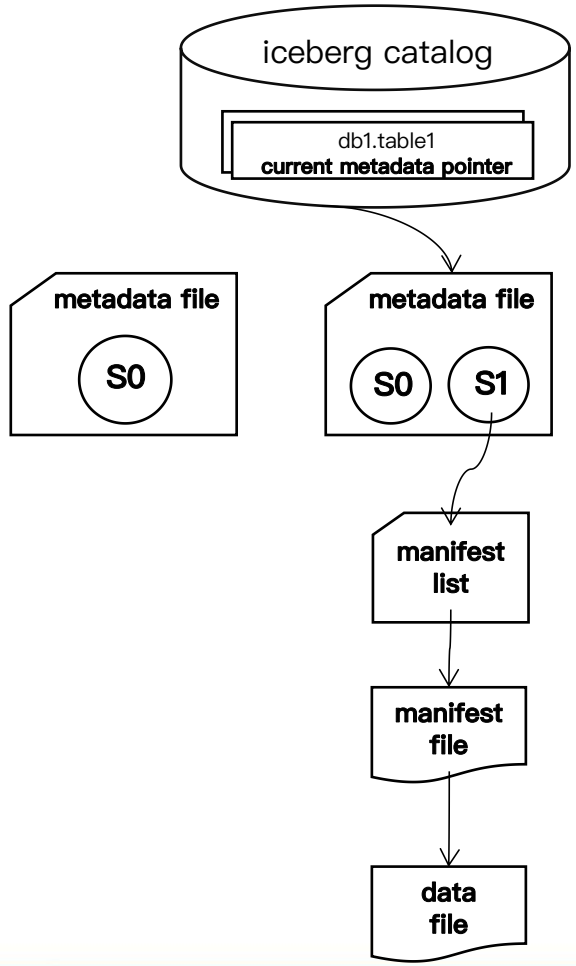

```
table1
├── data
└── metadata
    └── v1.metadata.json
```

插入数据


```
INSERT INTO db1.table1 VALUES (
    322,
    765,
    23.1,
    '2021-09-13 12:21:39'
);
```

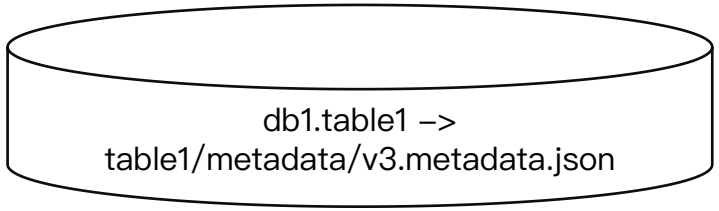



```
table1
├── data
│   └── order_ts_hour=2021-09-13-12
│       └── 0000-1-cae5f.parquet
└── metadata
    ├── d8f9-ad19-4e.avro
    ├── snap-2938-1-4302.avro
    ├── v1.metadata.json
    └── v2.metadata.json
```

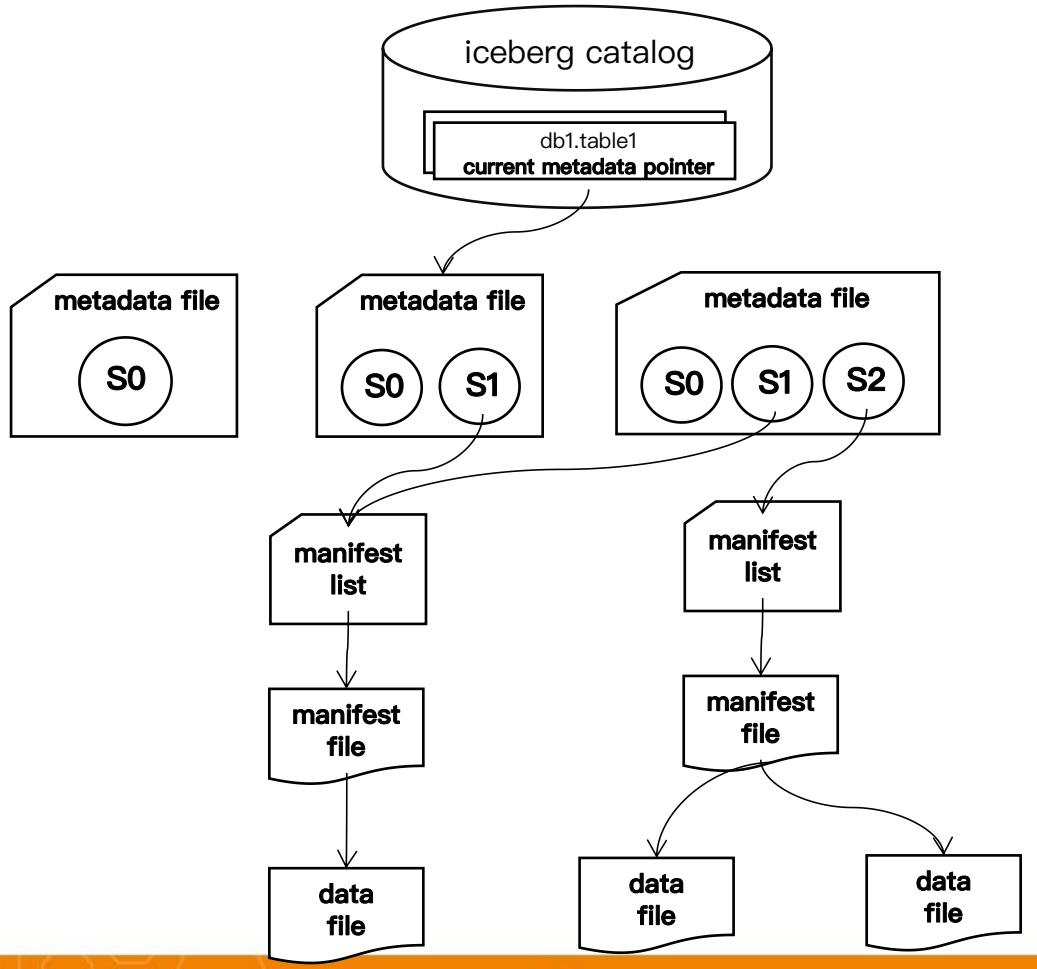


插入更多数据

INSERT INTO db1.table1 select * from table1_stage

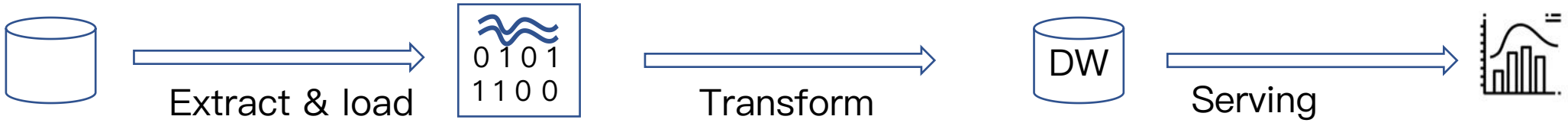


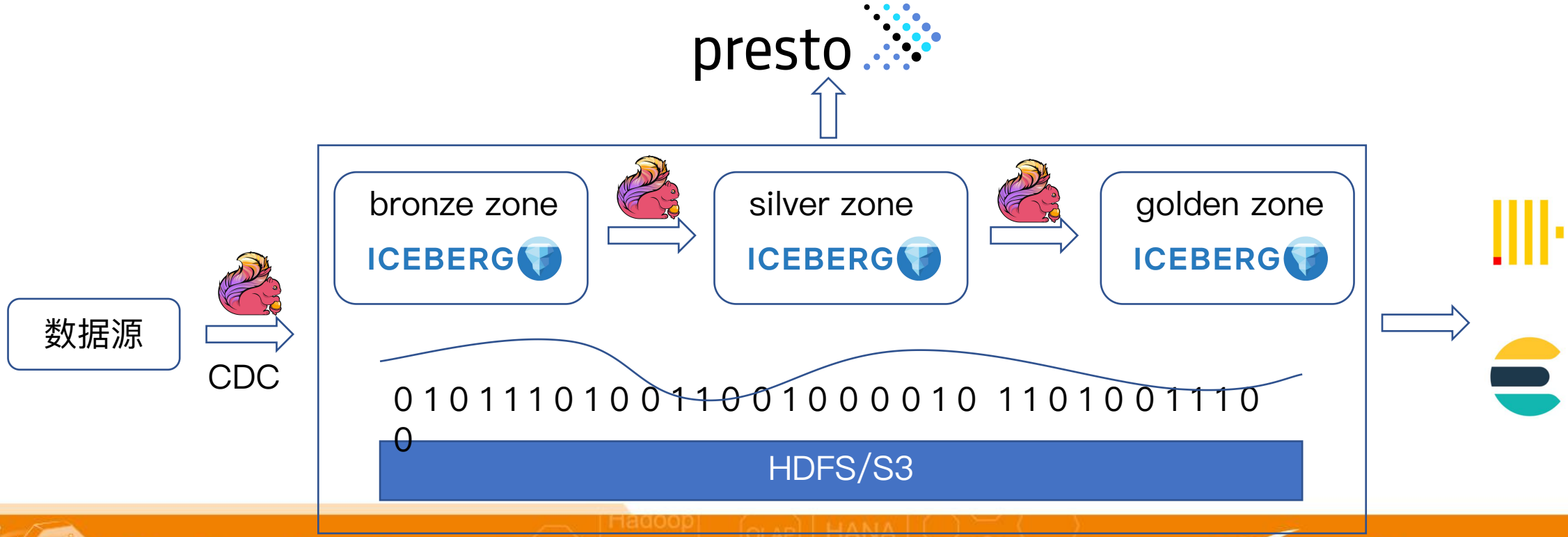

```
table1
├── data
│   ├── order_ts_hour=2021-09-13-12
│   │   ├── 0000-1-34a3d.parquet
│   │   └── 0000-1-cae5f.parquet
│   └── order_ts_hour=2021-09-15-13
│       └── 0000-3-1ee8a.parquet
└── metadata
    ├── 5de1-d3f8-54.avro
    ├── d8f9-ad19-4e.avro
    ├── snap-2938-1-4302.avro
    ├── snap-3261-3-9561.avro
    ├── v1.metadata.json
    ├── v2.metadata.json
    └── v3.metadata.json
```



实践方案

实时数据湖方案


Extract & load
0 1 0 1
1 1 0 0
Transform
DW
Serving

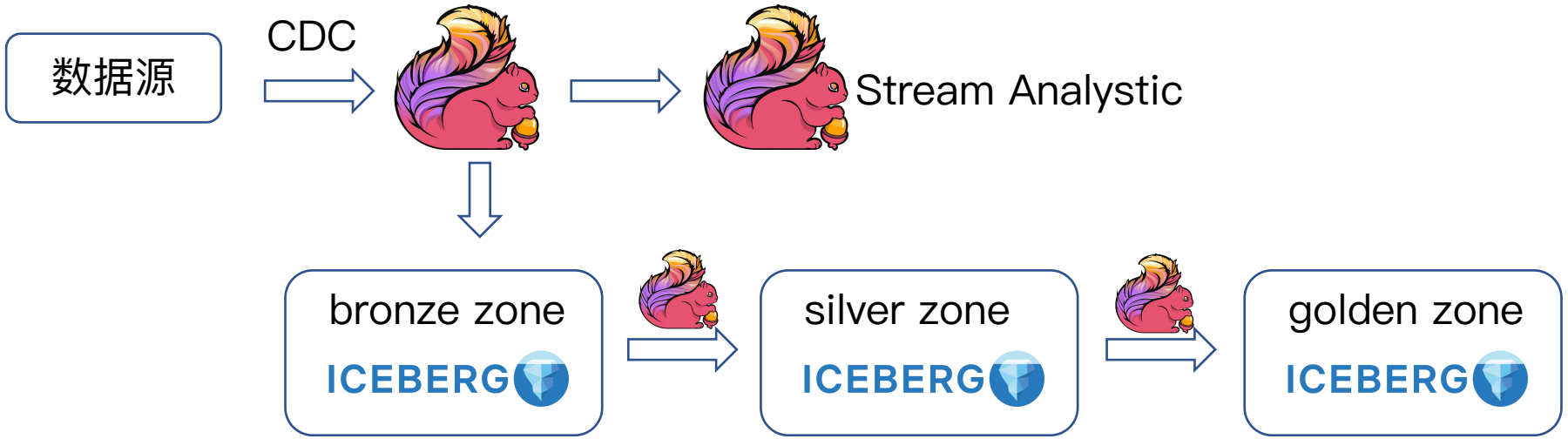

presto
bronze zone
ICEBERG
silver zone
ICEBERG
golden zone
ICEBERG
数据源
CDC
0 1 0 1 1 1 0 1 0 0 1 1 0 0 1 0 0 0 0 1 0 1 1 0 1 0 0 1 1 1 0 0
HDFS/S3

优劣分析

+ 存储层统一
+ 降低存储成本
+ 实现了部分业务的流批统一，达到分钟级延时
+ 解决了小文件问题，提升了查询速度

– 达不到秒级实时
– 实时开发难度变大

部分业务实现秒级实时



一套代码批和流两种运行模式，实现开发效率提升

未来实现Late batching，去掉批处理，节省计算资源

Fastdata数据平台介绍

Fastdata实时数据分析平台 架构

体验层
- SQL分析
- 数据开发
- 机器学习 高级分析
- 数据探索

计算层
- 流引擎（流批一体）
- 分析引擎（联邦查询/多维分析/物化视图）

元数据
- 元数据服务（HMS）
- 数据源透析
- Caching
- 权限与安全

存储层
- Parquet/orc/avro/CSV/json
- Iceberg表引擎
- S3/HDFS

基础服务
- IAM
- OAM
- 集群管理
- 统一运维
- 安全控制
- 自动化部署
- 日志服务
- 云服务

数据集成

数据源
- SaaS应用
- OLTP
- ERP
- CRM
- 文件
- 传感器
- 设备
- 社交

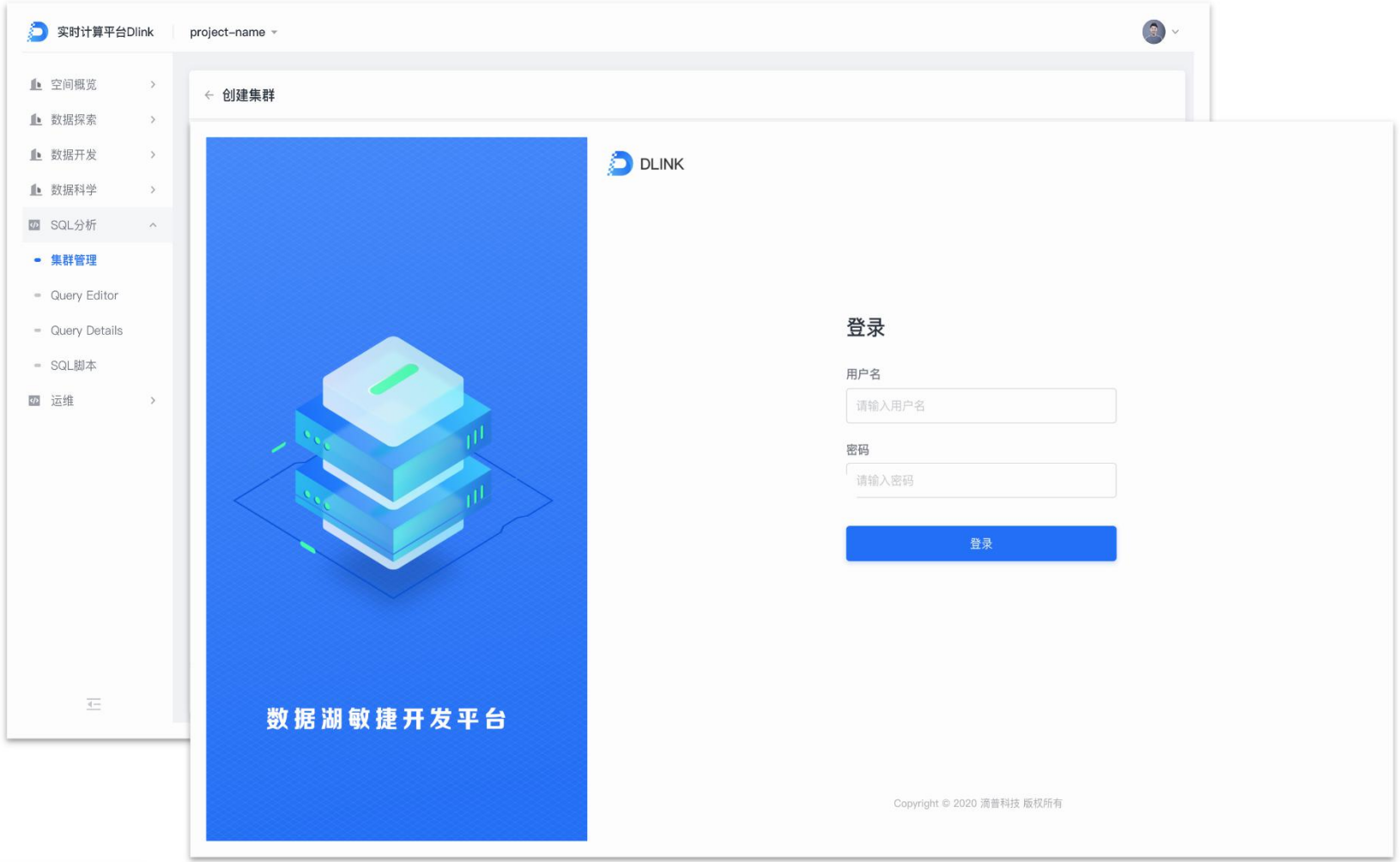
Fastdata敏捷开发平台

支持作业开发、数据处理、数据分析、数据模型、元数据 等功能。

优势：简单、中立、敏捷、低成本

特性：

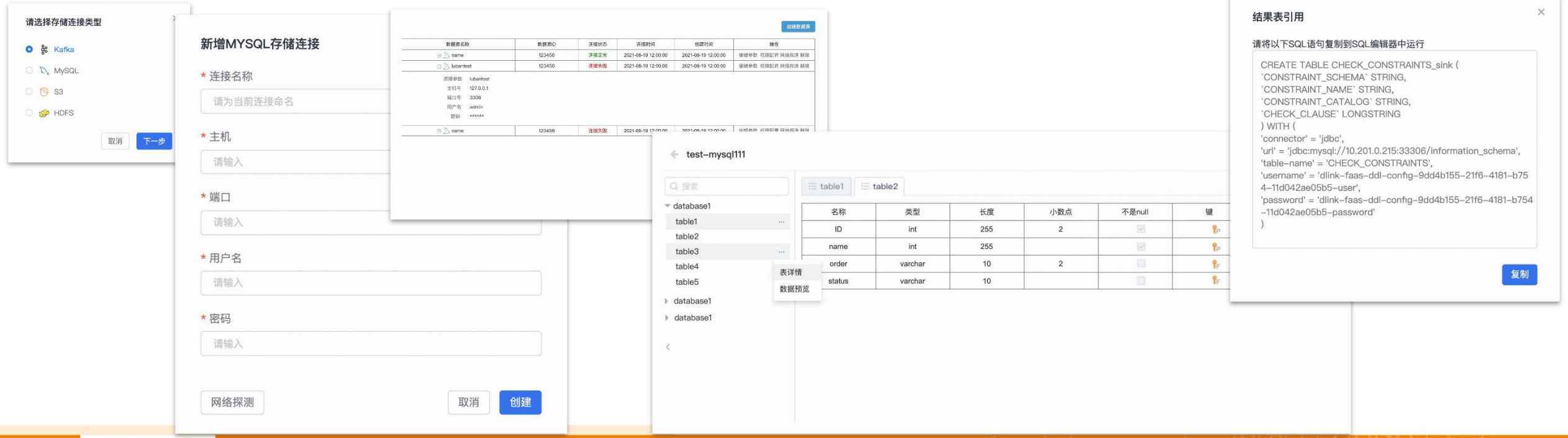
1、空间概览（数据安全）
2、数据探索
3、数据开发
4、数据科学
5、SQL分析
6、运维监控



特性：数据探索

数据探索能力：数据源数据profiling

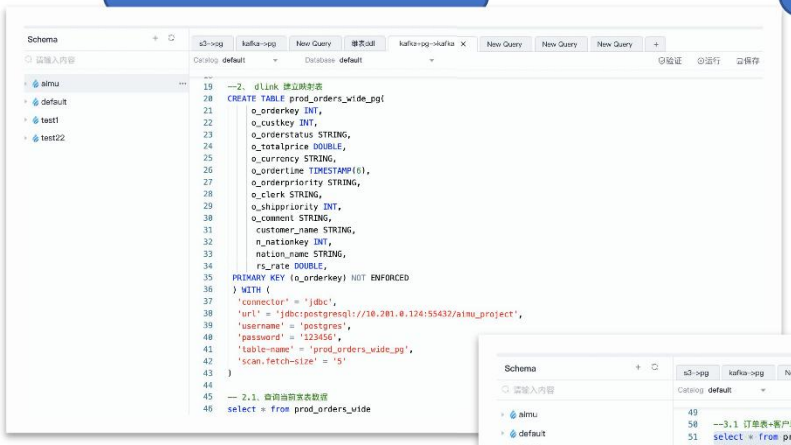
创建数据源 → 数据源分配 → 数据预览 → 数据应用

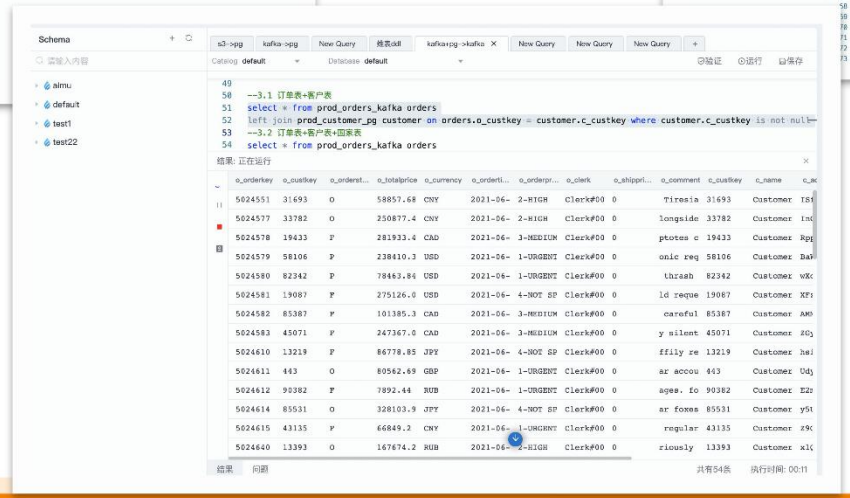


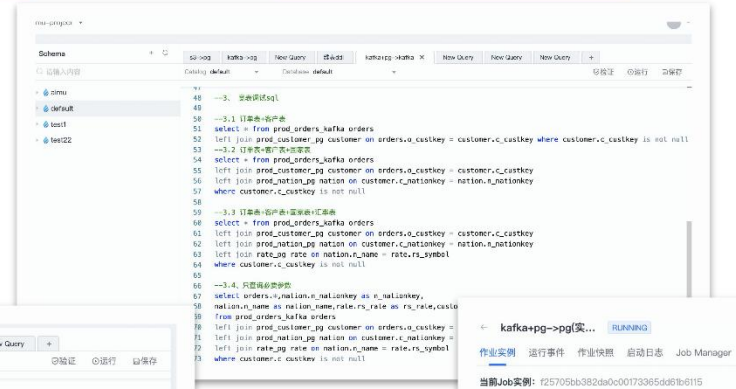
特性：数据开发

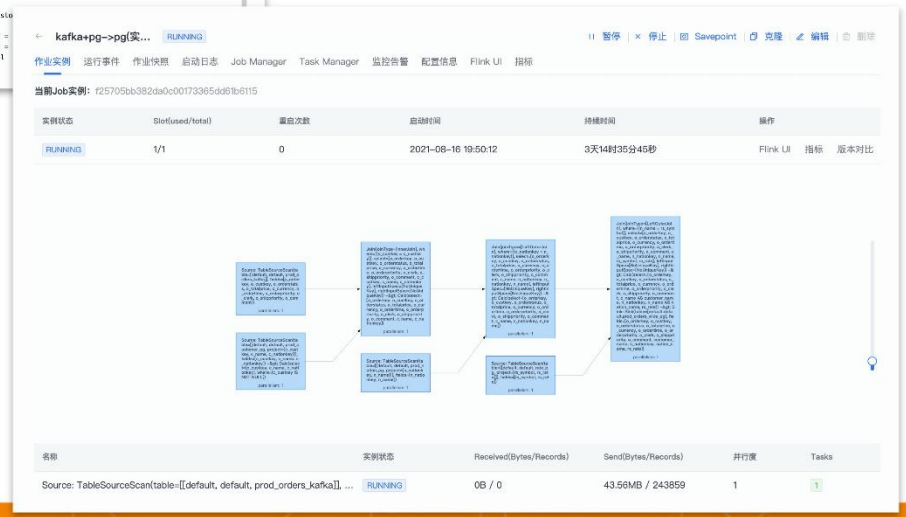
数据开发能力：sql全流程开发

构建映射表 → sql预览 → 调试sql → sql作业









特性： 数据科学

数据科学能力：

数据模型

在线机器学习

数据挖掘

谢谢