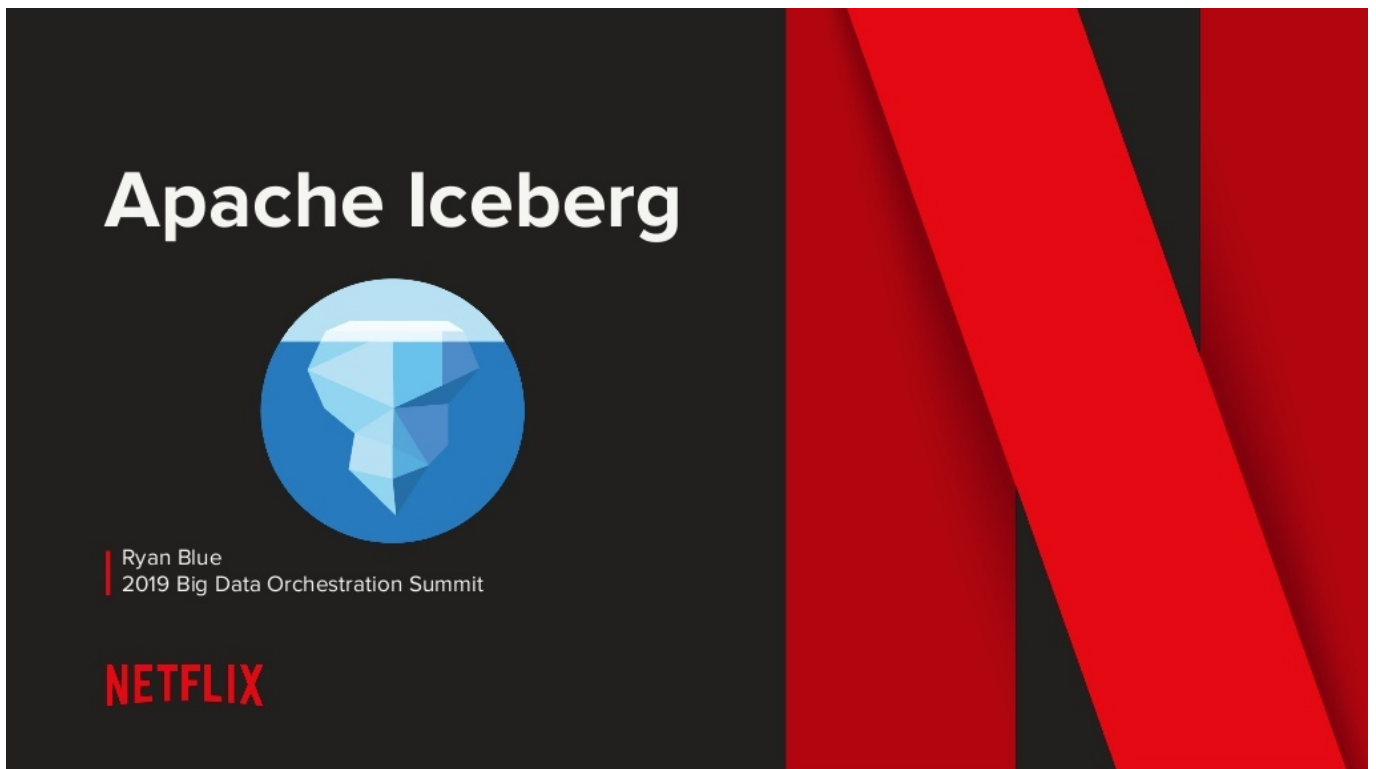


Apache iceberg : Netflix 数据仓库的基石



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Apache Iceberg 是一种用于跟踪超大规模表的新格式，是专门为对象存储（如S3）而设计的。本文将介绍为什么 Netflix 需要构建 Iceberg，Apache Iceberg 的高层次设计，并会介绍那些能够更好地解决查询性能问题的细节。



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

本文由 Ryan Blue 分享，他在 Netflix 从事开源数据项目，是 Apache Iceberg 的最初创建者之一，也是 Apache Spark, Parquet, 以及 Avro 贡献者。

关注 过往记忆大数据 公众号并在后台回复 Iceberg 关键字获取本文 PPT。



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Apache Iceberg 是由 Netflix 开发开源的，其于 2018年11月16日进入 Apache 孵化器，是 Netflix 公司数据仓库基础。在功能上和我们熟悉的 Delta Lake 或者 Apache Hudi 类似，但各有优缺点。任何东西的诞生都是有其背后的原因，那么为什么 Netflix 需要开发 Apache Iceberg ？

5-year Challenges

- Smarter processing engines
 - CBO, better join implementations
 - Result set caching, materialized views
- Reduce manual data maintenance
 - Data librarian services
 - Declarative instead of imperative

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

在 Netflix，他们希望有更智能的处理引擎，比如有 CBO 优化，更好的 Join 实现，缓存结果集以及物化视图等功能。同时，他们也希望减少人工维护数据。

Problem Whack-a-mole

- Unsafe operations are everywhere
 - Writing to multiple partitions
 - Renaming a column
- Interaction with object stores causes major headaches
 - Eventual consistency to performance problems
 - Output committers can't fix it
- Endless scale challenges

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Netflix 面临的问题包括：1、不安全的操作随处可见；2、和对象存储交互有时候会出现很大的问题；3、无休止的可扩展性挑战。

为了解决这些问题，Iceberg 诞生了。那么 Iceberg 是什么？

What is Iceberg?



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

iceberg 是一种可伸缩的表存储格式，内置了许多最佳实践。

Iceberg is a scalable format for tables with a lot of best practices built in.

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

什么？是一种存储格式？可使我们已经有 Parquet，Avro 以及 ORC 这些格式了，为什么还要设计一种新格式？

A format?

We already have Parquet, Avro and ORC . . .

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

A table format

- File formats help you modify or skip data in a single file
- Table formats do the same thing for a collection of files

- To demonstrate this, consider Hive tables . . .

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

iceberg 允许我们在一个文件里面修改或者过滤数据；当然多个文件也支持这些操作。为了展示这点，我们来看看一张 Hive 表。

Hive Tables

- Key idea: **organize data in a directory tree**

```
date=20180513/  
  |- hour=18/  
    |- ...  
  |- hour=19/  
    |- part-000.parquet  
    |- ...  
    |- part-031.parquet  
  |- hour=20/  
    |- ...
```

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Hive 表的核心思想是把数据组织成目录树，如上所述。

Hive Tables

- Filter: WHERE **date = '20180513' AND hour = 19**

```
date=20180513/  
|- hour=18/  
|   |- ...  
|- hour=19/  
|   |- part-000.parquet  
|   |- ...  
|   |- part-031.parquet  
|- hour=20/  
|   |- ...
```

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

如果我们需要过滤数据，可以在 where 里面添加分区相关的信息。

Hive Metastore

- Problem: **too much directory listing** for large tables
- Solution: use HMS to track partitions

```
date=20180513/hour=19 -> hdfs://.../date=20180513/hour=19
date=20180513/hour=20 -> hdfs://.../date=20180513/hour=20
```

- The file system still tracks the files in each partition . . .

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

带来的问题是如果一张表有很多分区，我们需要使用 HMS (Hive MetaStore) 来记录这些分区，同时底层的文件系统（比如 HDFS）仍然需要在每个分区里面记录这些分区数据。

Hive Tables: Problems

- State is kept in both the **metastore** and in a **file system**
- Changes are not atomic without locking
- Requires directory listing
 - $O(n)$ listing calls, $n = \#$ matching partitions
 - Eventual consistency breaks correctness

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

这就导致我们需要在 HMS 和 文件系统里面同时保存一些状态信息；因为缺乏锁机制，所以对上面两个系统进行修改也不能保证原子性。

Hive Tables: Benefits

- Everything supports Hive tables*
 - Engines: Hive, Spark, Presto, Flink, Pig
 - Tools: Hudi, NiFi, Flume, Sqoop
- **Simplicity** and **ubiquity** have made Hive tables indispensable
- The whole ecosystem uses the same at-rest data!

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

当然 Hive 这样维护表也不是没有好处。这种设计使得很多引擎（Hive、Spark、Presto、Flink、Pig）都支持读写 Hive 表，同时支持很多第三方工具。简单和透明使得 Hive 表变得不可或缺的。

Iceberg

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Iceberg's Goals

- **An open spec and community for at-rest data interchange**
 - Maintain a clear spec for the format
 - Design for multiple implementations across languages
 - Support needs across projects to avoid fragmentation

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Iceberg 的目标包括：1、成为静态数据交换的开放规范，维护一个清晰的格式规范，支持多语言

, 支持跨项目的需求等。

Iceberg's Goals

- **Improve scale and reliability**
 - Work on a single node, scale to a cluster
 - All changes are atomic, with serializable isolation
 - Native support for cloud object stores
 - Support many concurrent writers

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

2、提升扩展性和可靠性。能够在单个节点上运行，也能在集群上运行。所有的修改都是原子性的，串行化隔离。原生支持云对象存储，支持多并发写。

Iceberg's Goals

- **Fix persistent usability problems**
 - In-place evolution for schema and layout (no side-effects)
 - Hide partitioning: insulate queries from physical layout
 - Support time-travel, rollback, and metadata inspection
 - Configure tables, not jobs
- Tables should have **no unpleasant surprises**

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

3、修复持续的可用性问题，比如模式演进，分区隐藏，支持时间旅行、回滚等。

Iceberg's Design

- Key idea: **track all files in a table** over time
 - A **snapshot** is a complete list of files in a table
 - Each write produces and commits a new snapshot



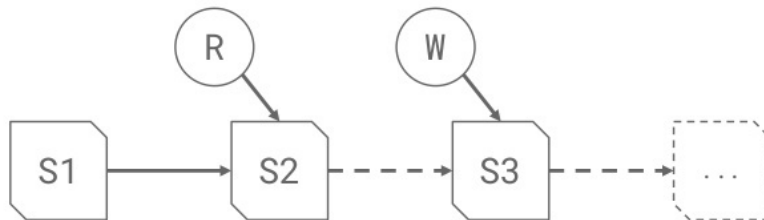
NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Iceberg 主要设计思想：记录表在所有时间的所有文件，和 Delta Lake 或 Apache Hudi 一样，支持 snapshot，其是表在某个时刻的完整文件列表。每一次写操作都会生成一个新的快照。

Iceberg's Design

- Readers use the current snapshot
- Writers optimistically create new snapshots, then commit



NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

读取数据的时候使用当前的快照，Iceberg 使用乐观锁机制来创建新的快照，然后提交。

Iceberg Design Benefits

- All changes are atomic
- No expensive (or inconsistent) file system operations
- Snapshots are indexed for scan planning on a single node
- CBO metrics are reliable
- Versions for incremental updates and **materialized views**

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Iceberg 这么设计的好处是：

- 所有的修改都是原子性的；
- 没有耗时的文件系统操作；
- 快照是索引好的，以便加速读取；
- CBO metrics 信息是可靠的；
- 更新支持版本，支持物化视图。

Iceberg at Netflix



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Scale

- Production tables: **tens of petabytes, millions of partitions**
 - Scan planning fits on a single node
 - Advanced filtering enables more use cases
 - Overall performance is better
- Low latency queries are faster for large tables

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

Iceberg 在 Netflix 生产环境维护着数十 TB

的数据，数百万个分区。对大表进行查询能够提供低延迟的响应。

Concurrency

- Production Flink pipeline writing in 3 AWS regions
- Lift service moving data into a single region
- Merge service compacting small files

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

生产环境中使用 Flink 管道在 3 个 AWS regions 写数据。Lift 服务将数据移到一个 region。Merge 服务对小文件进行合并。

Usability

- Rollback is popular
- Metadata tables
 - Track down the version a job read
 - Find the process that wrote a bad version

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

可用性方面：回滚是家常便饭。

Future Work

- Spark vectorization for faster bulk reads
 - Presto vectorization already done
- Row-level delete encodings
 - MERGE INTO
 - ID equality predicates

NETFLIX

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：iteblog_hadoop

未来工作：1、支持 Spark 向量化以便实现快速的 bulk read，Presto 向量化已经支持。2、行级别的删除，支持 MERGE INTO 等。

本博客文章除特别声明，全部都是原创！
原创文章版权归过往记忆大数据（[过往记忆](#)）所有，未经许可不得转载。
本文链接: [【】](#)（）