RisingWave x ClickHouse
实时分析新范式

Patrick Huang | RisingWave Labs

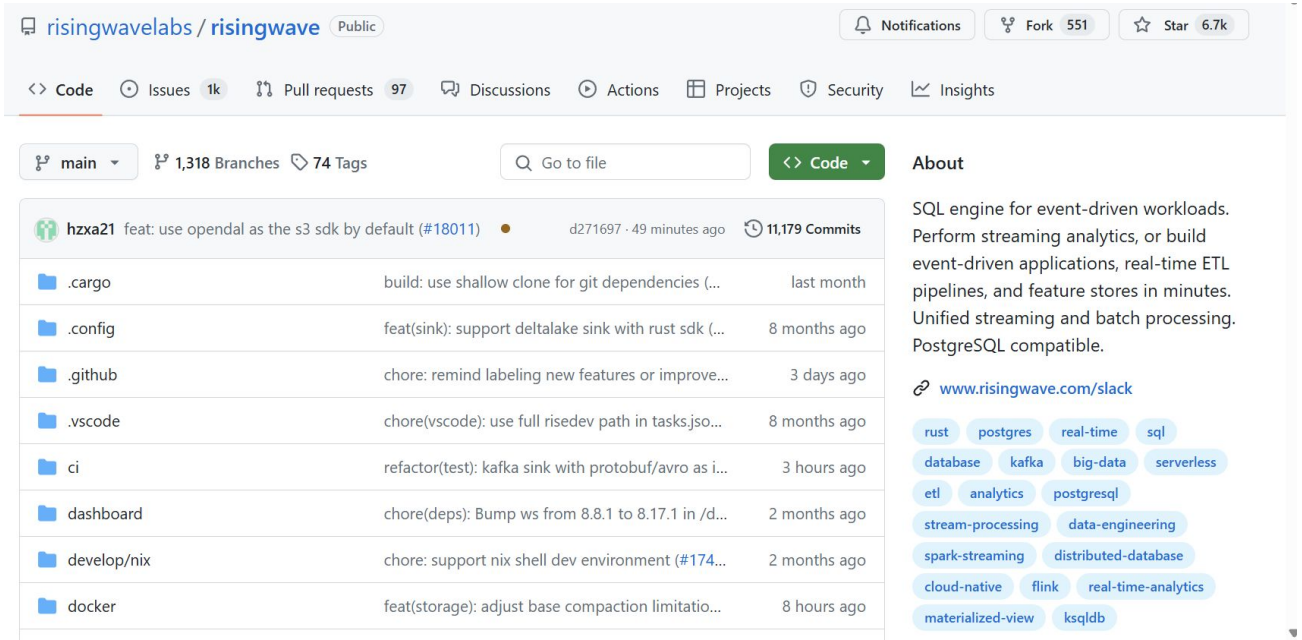
RisingWave

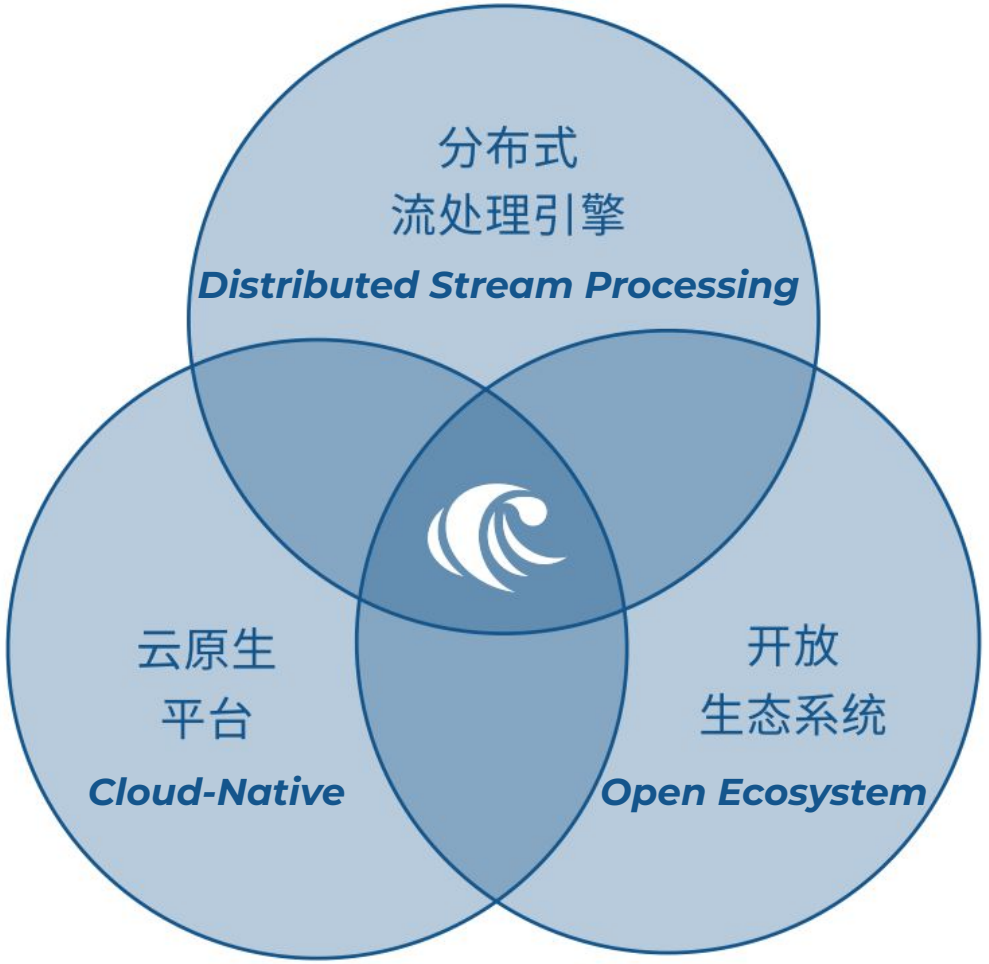
About Me

- Patrick Huang
  - RisingWave Labs -> VP of Engineering (DB Kernel & Storage)
  - Wechat Infra -> Staff Engineer (Large-scale Online Storage)
  - LinkedIn Infra -> Senior Engineer (Kafka & Next-gen Streaming Platform)
  - CMU DB Labs -> Research Assistant (Peloton DBMS)

RisingWave

SQL platform for Streaming





RisingWave

RisingWave 诞生于 2021 年初，开发团队由资深数据库研究院，于来自 AWS Readshift、Snowflake、Linkedin、Uber 等知名企业的技术专家共同组成。

我们的使命是为企业数据平台提供实时、可靠、高效、低门槛的事件处理工具，帮助企业利用实时数据实现业务增长。

经过三年的打磨，RisingWave已成为第三代流处理系统中最具代表性的产品，在全球范围内在互联网、金融、能源、航空航天、供应链、智能汽车等多个领域的生产环境中落地应用。目前，RisingWave 全球日活集群已达1700+个。

Ease of Use

- SQL first (PostgreSQL Compatible)
- Streaming + Serving
- Queryable Internal State via SQL

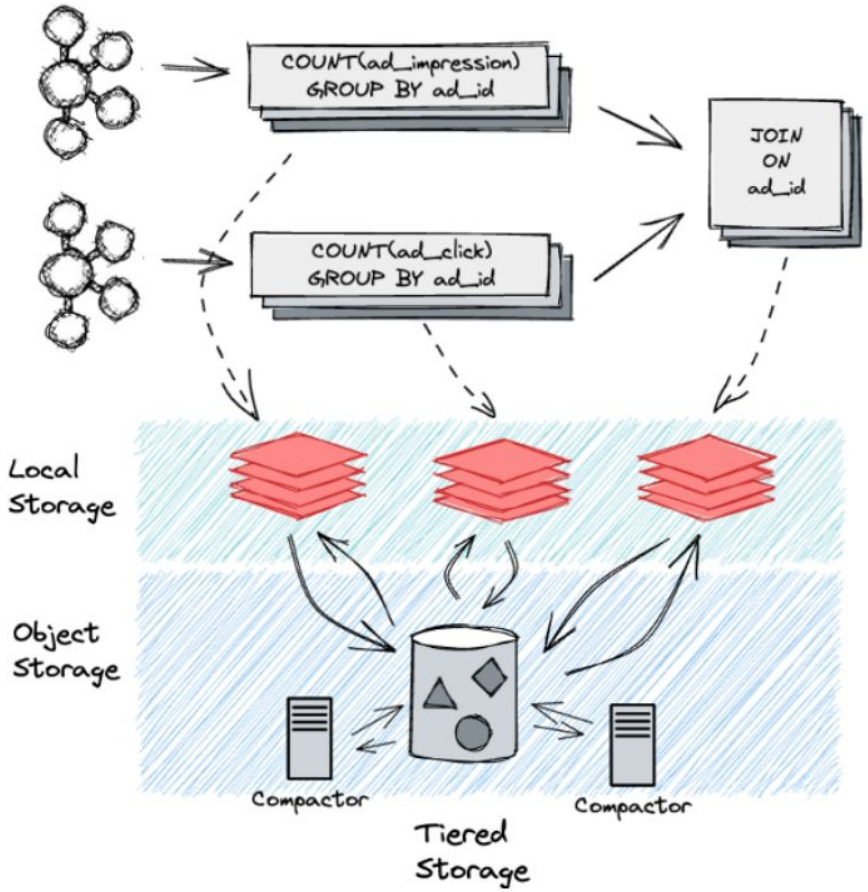
Quick Start on your laptop (Linux/macOS) with RisingWave Standalone

```
> curl https://risingwave.com/sh | sh
> ./risingwave
> psql -h localhost -p 4566 -d dev -U root
```

https://www.risingwavetutorial.com/

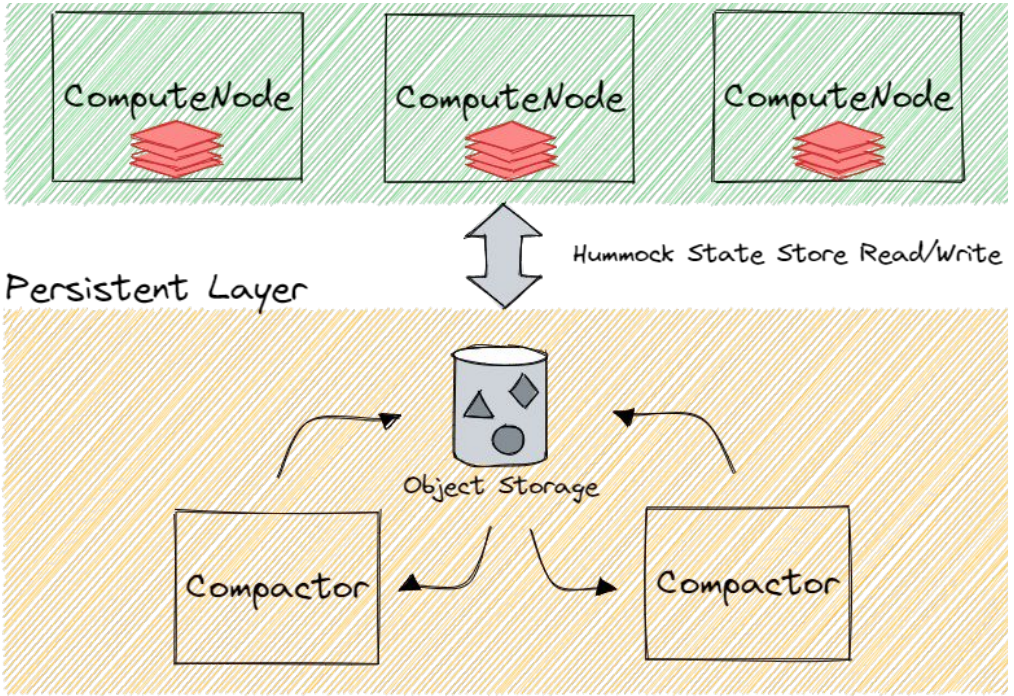
Robust

- Sub-second freshness
- Recovery & Scaling in seconds
- Large State Management
- 20+ Way Streaming Join



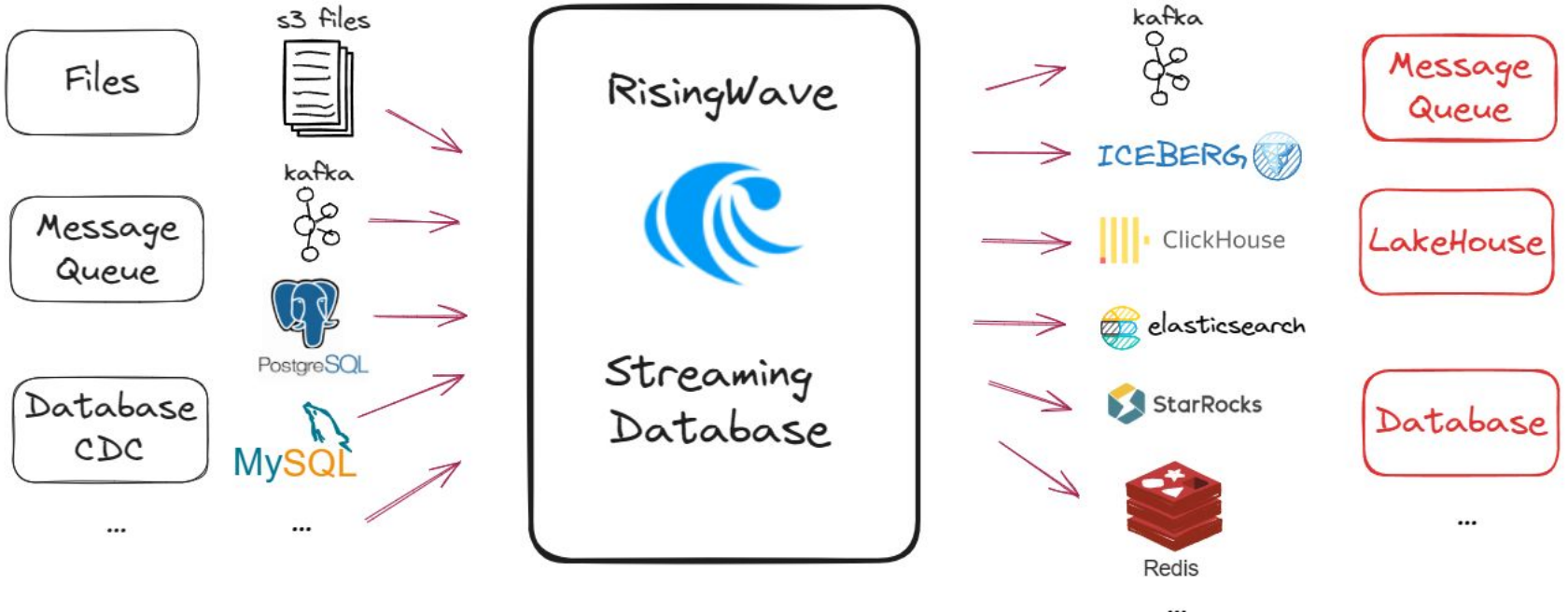
Cost Efficient

- Compute & Storage Disaggregation
- Tiered Storage + Local Cache
- Serverless Compaction
- Auto Scale Streaming Job
- No Vender Lock-in. Supported backend:
  - S3、Azure Blob、GCS、OSS、COS、OBS
  - MinIO、HDFS、DFS、...

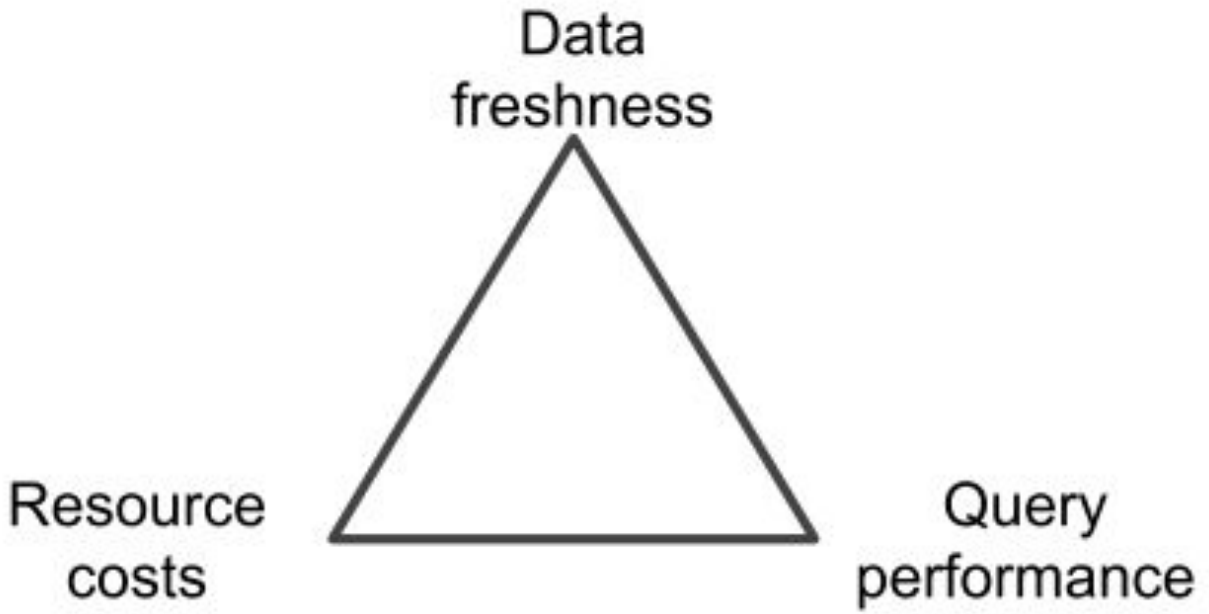


Open Ecosystem

- **Source**
  - MQ: Kafka、Pulsar、Repanda
  - CDC: Debezium、MySQL、PG
  - Apache Iceberg、Files
- **Sink**
  - Kafka、Pulsar、Repanda
  - Clickhouse、Starrocks、ES、Redis
  - Iceberg、Delta Lake



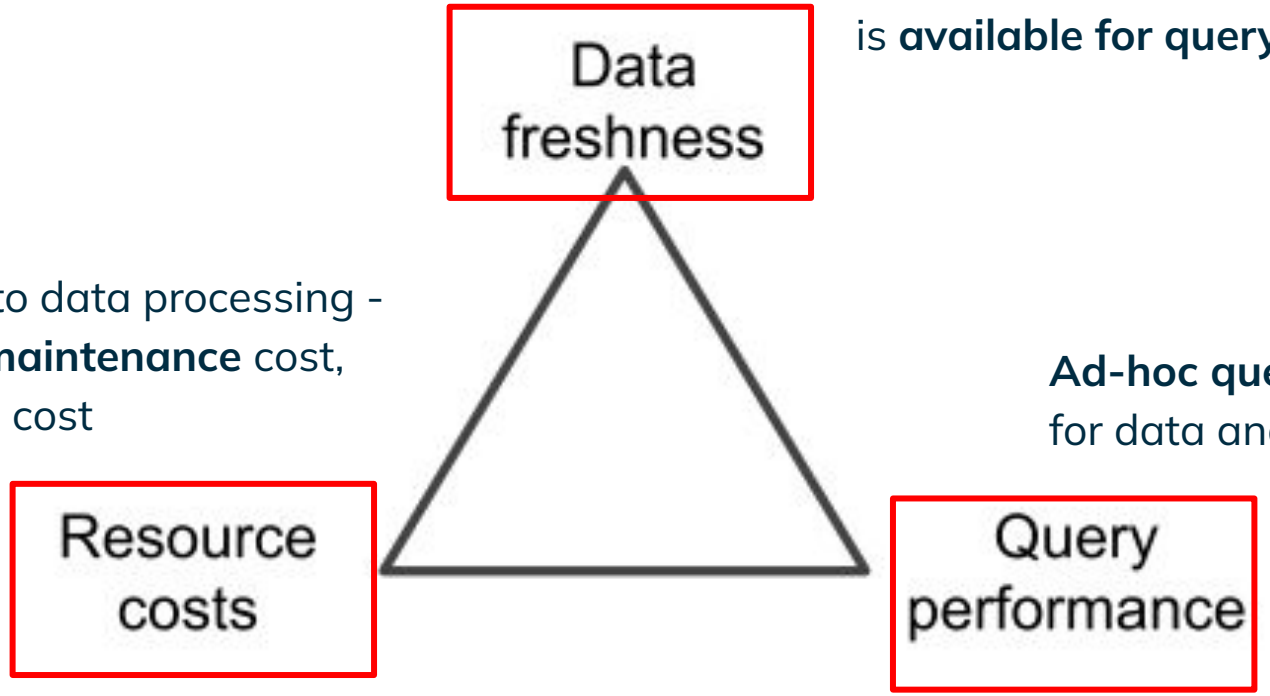
OLAP v.s Streaming: The Tradeoff Triangle



Reference:
Napa: Powering Scalable Data Warehousing with Robust Query Performance at Google. Proceedings of the VLDB Endowment (PVLDB), 14 (12)(2021), pp. 2986-2998

OLAP v.s Streaming: The Tradeoff Triangle

The time between a row is **available in source** to the time it is **available for querying**



Costs arise due to data processing - **ingestion** cost, **maintenance** cost, **query execution** cost

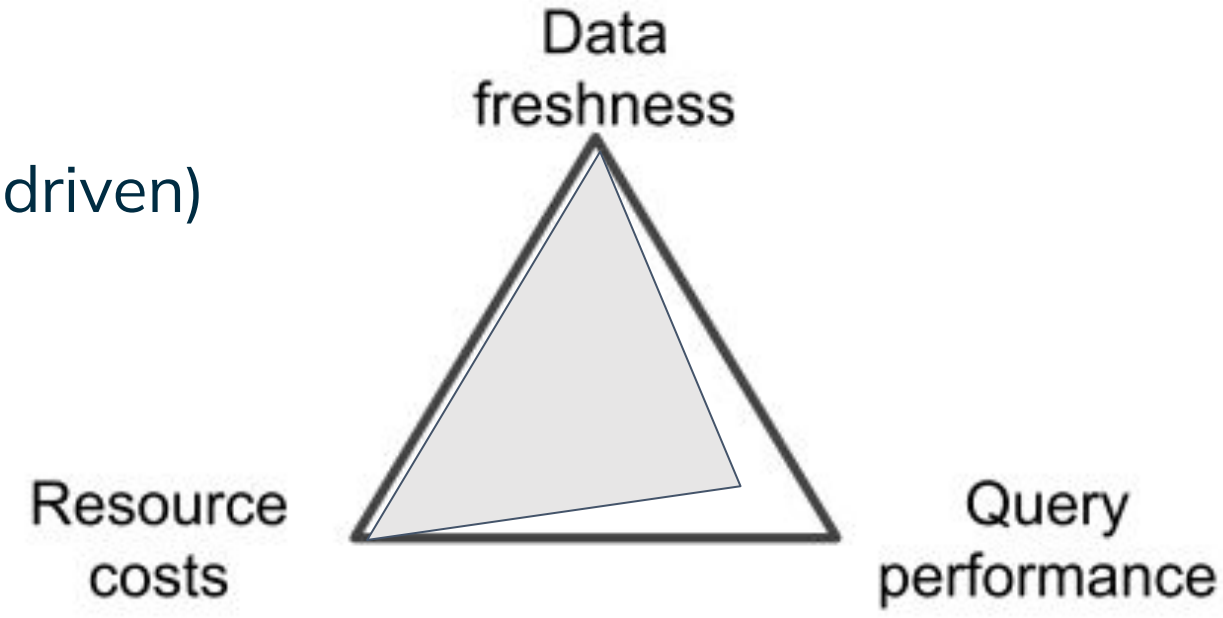
Ad-hoc query latency for data analytics.

Reference:
Napa: Powering Scalable Data Warehousing with Robust Query Performance at Google. Proceedings of the VLDB Endowment (PVLDB), 14 (12)(2021), pp. 2986-2998

OLAP v.s Streaming: The Tradeoff Triangle

RisingWave

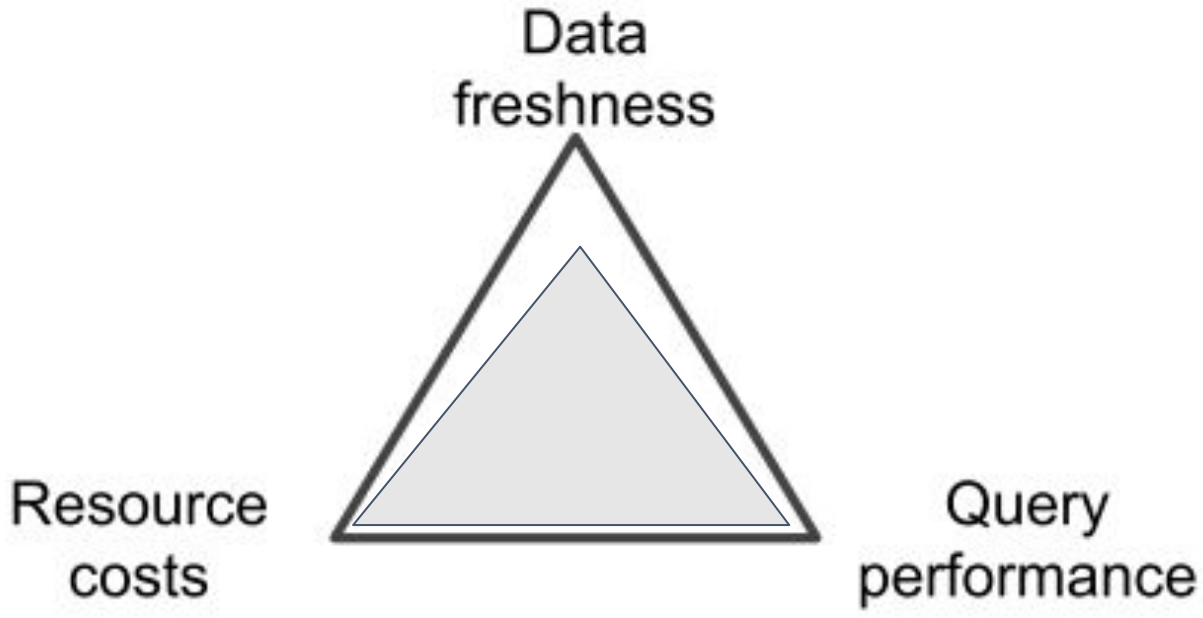
- Incremental Computation (Event-driven)
- Optimized for freshness
- Optimized for pre-defined query
- Row storage
  - Good for event-driven workload
  - Good for streaming state
  - Good for result serving



OLAP v.s Streaming: The Tradeoff Triangle

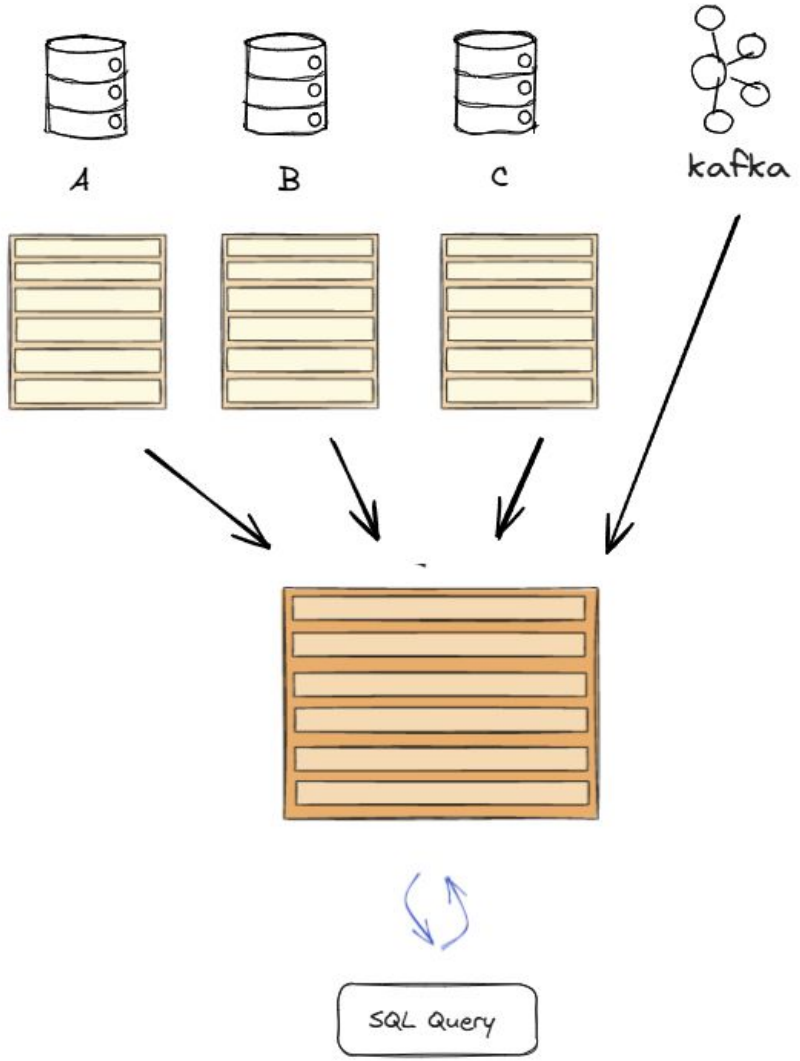
OLAP systems

- Full Computation (Query Driven)
- Optimized for query latency
- Optimized for ad-hoc query
- Columnar storage
  - Good for batch workload
  - Good for long-range scans
  - Good for interactive analytics



场景分析：实时大宽表

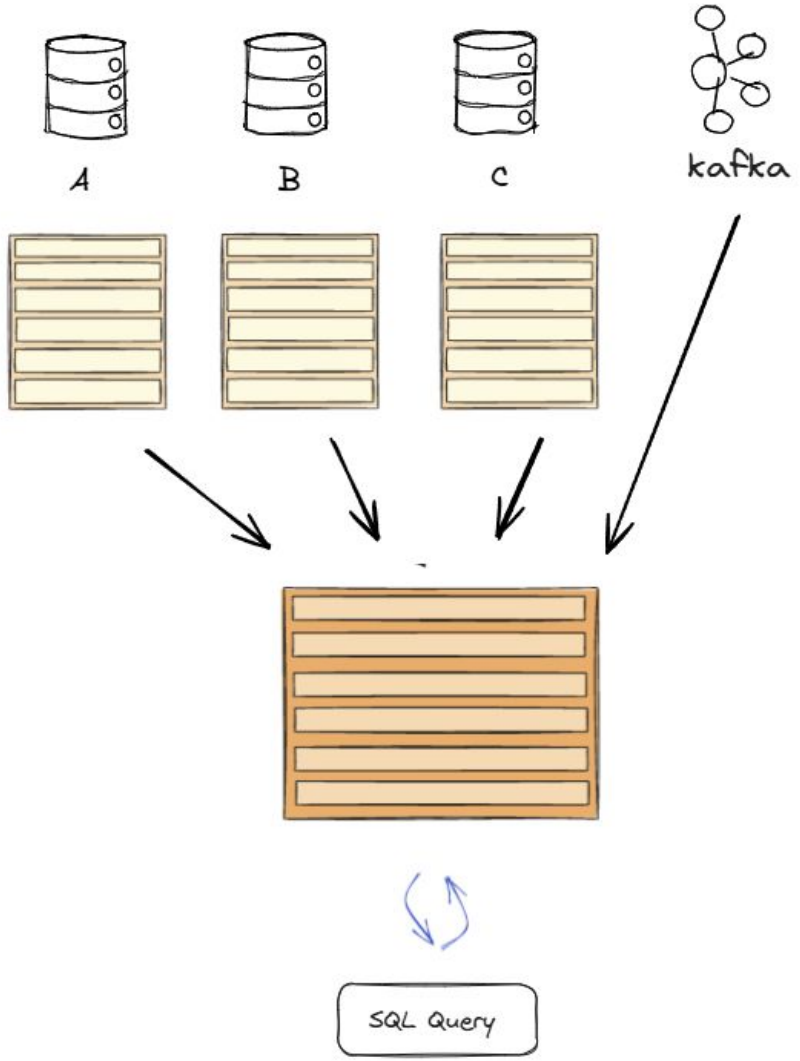
- 大宽表
  - 包含**大量列（字段）**的数据表
  - 相关的数据都在同一行，查询分析更方便
  - 用于**交互式查询分析**



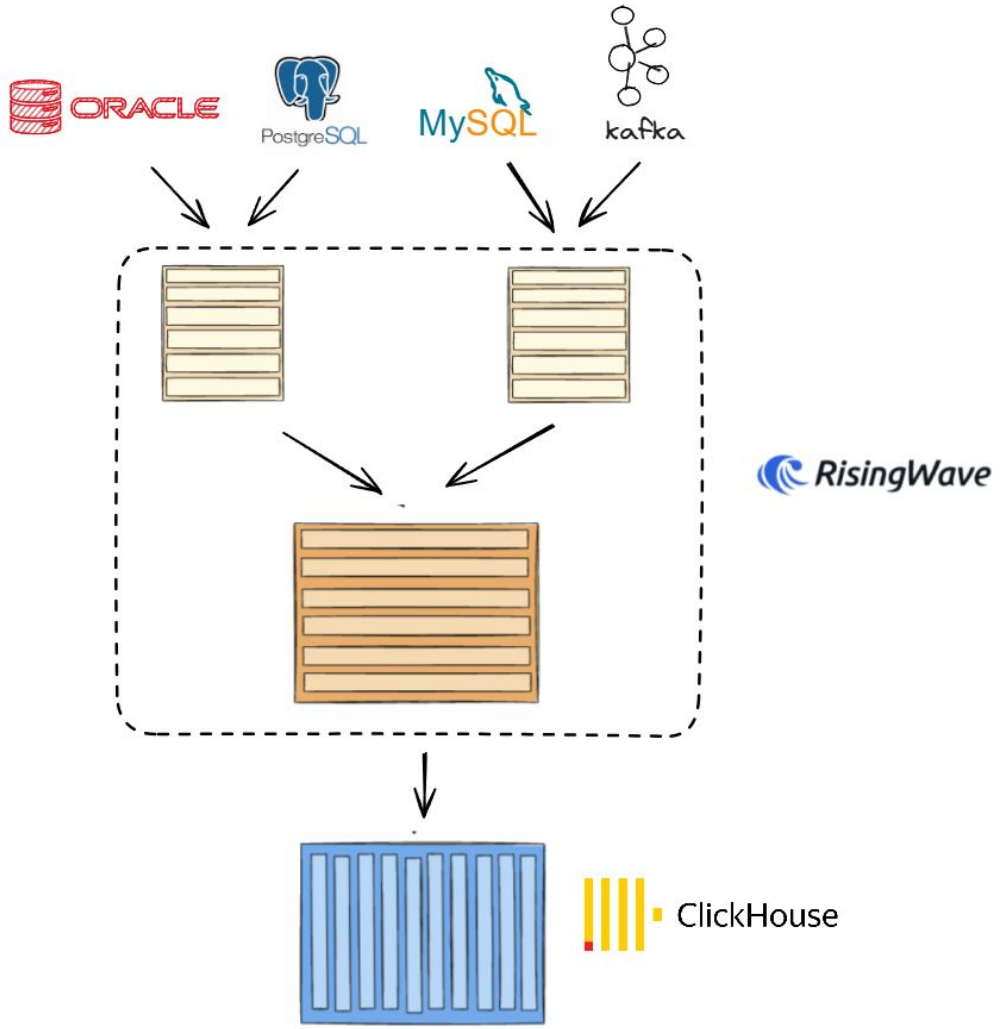
ClickHouse

场景分析：实时大宽表

- 实时打宽
  - 又称多流拼接，将来自**不同数据源**的多个表互相关联，打平成宽表
  - 打宽的pattern**较为固定**（基于ID列）
  - 打宽的**实时性**要求越来越高

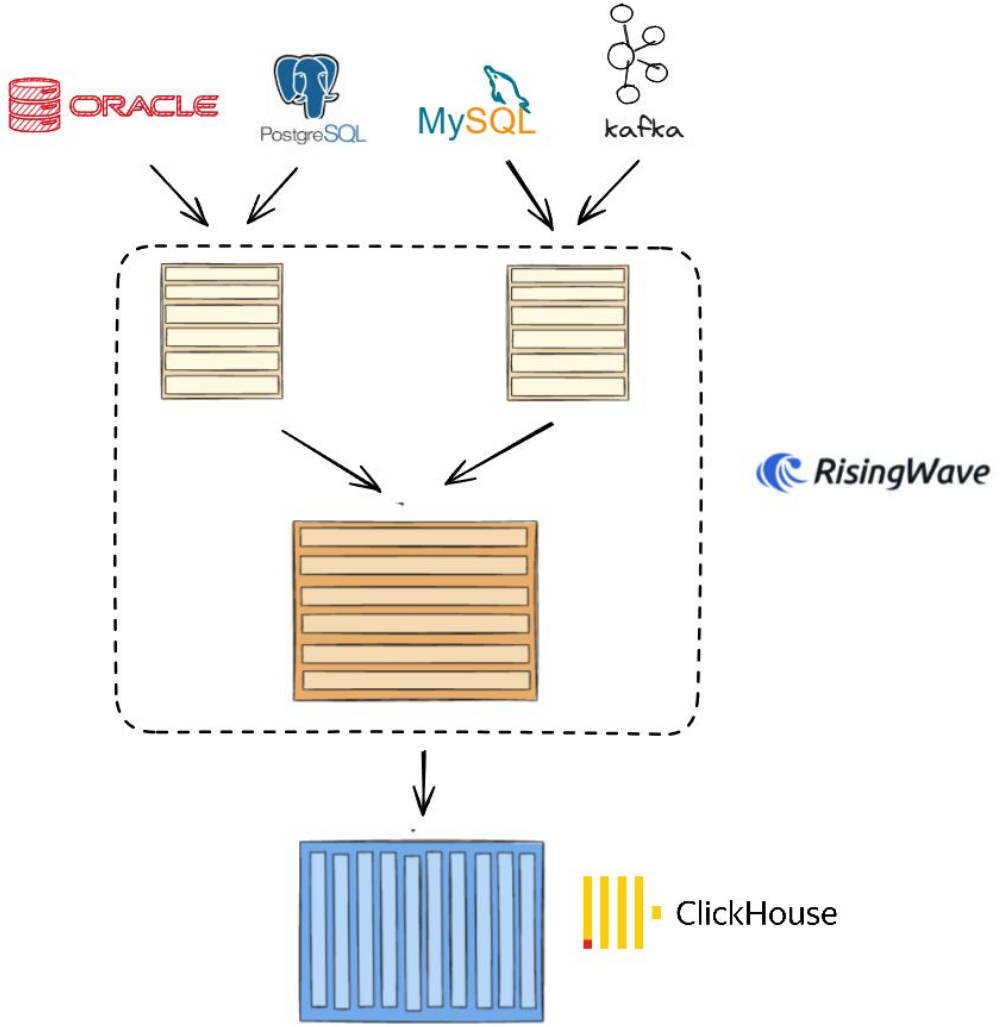


RisingWave x ClickHouse：实时打宽场景分析



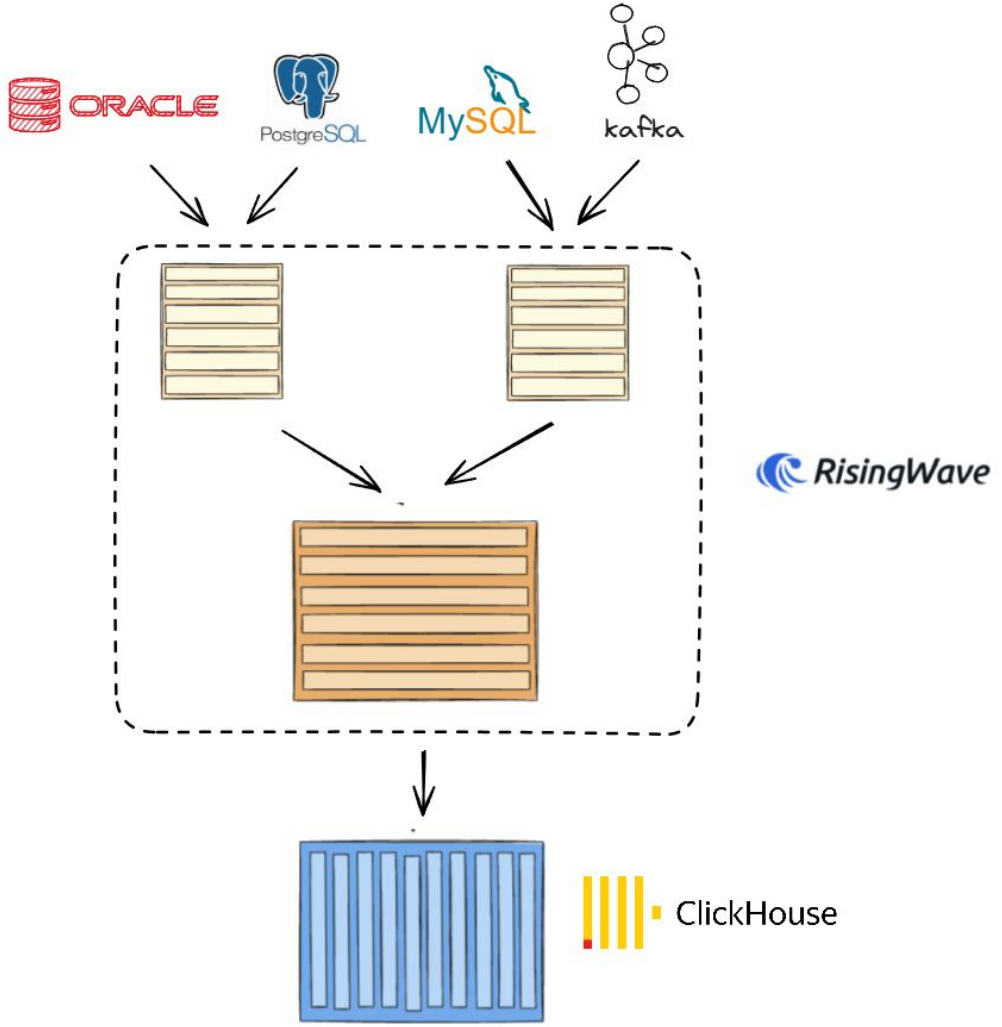
RisingWave x ClickHouse：实时打宽场景分析

- 挑战#1：数据来自不同数据源
- 挑战#2：超大状态维护
- 挑战#3：流式数仓写入



实时打宽挑战#1：数据来自不同数据源

- 来自DB的变更（CDC）：
  - MySQL
  - PostgreSQL
  - Oracle
  - TiDB、MongoDB、...
- 来自事件流（MQ）
  - Kafka
  - Pulsar
  - MQTT



RisingWave基础概念：SOURCE

- Source支持消费多种数据源：Kafka、Plusar、Kinses、S3
- 消息Encoding支持：AVRO、JSON、PROTOBUF、CSV、BYTES
- 消息队列支持指定消费开始位置
- 支持Schema Registry

```
CREATE SOURCE ad_click (
    bid_id BIGINT,
    click_timestamp TIMESTAMPZ
) WITH (
    connector = 'kafka',
    topic = 'ad_click',
    properties.bootstrap.server = 'xxx',
    scan.startup.mode = 'earliest'
) FORMAT PLAIN ENCODE AVRO (
   schema.registry = 'xxx'
);
```

RisingWave基础概念：TABLE

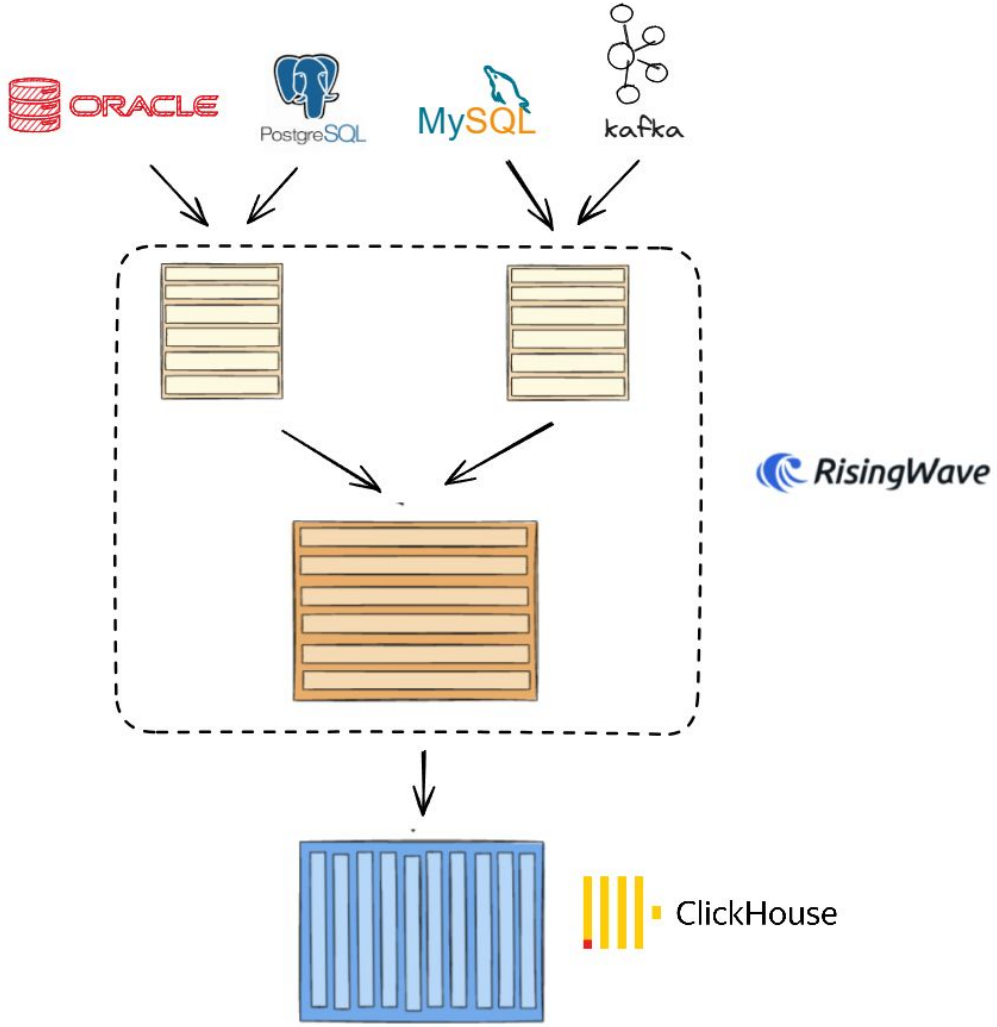
- Table可以消费所有Source支持的数据源
- Table会物化数据到表，支持主键
- Table支持对接上游CDC
  - OLTP数据库（MySQL、PostgreSQL、Oracle、TiDB等）
  - NoSQL数据库（MongoDB等）
- Table支持DML，可增删改
- 消息Format支持：PLAIN、DEBEZIUM、CANAL、MAXWELL、UPSERT

```
CREATE TABLE orders (
    order_id int,
    price decimal,
    PRIMARY KEY (order_id)
) WITH (
 connector = 'mysql-cdc',
 hostname = '127.0.0.1',
 port = '3306',
 username = 'root',
 password = 'xxx',
 database.name = 'mydb',
 table.name = 'orders',
);
```

实时打宽挑战#2：超大状态维护

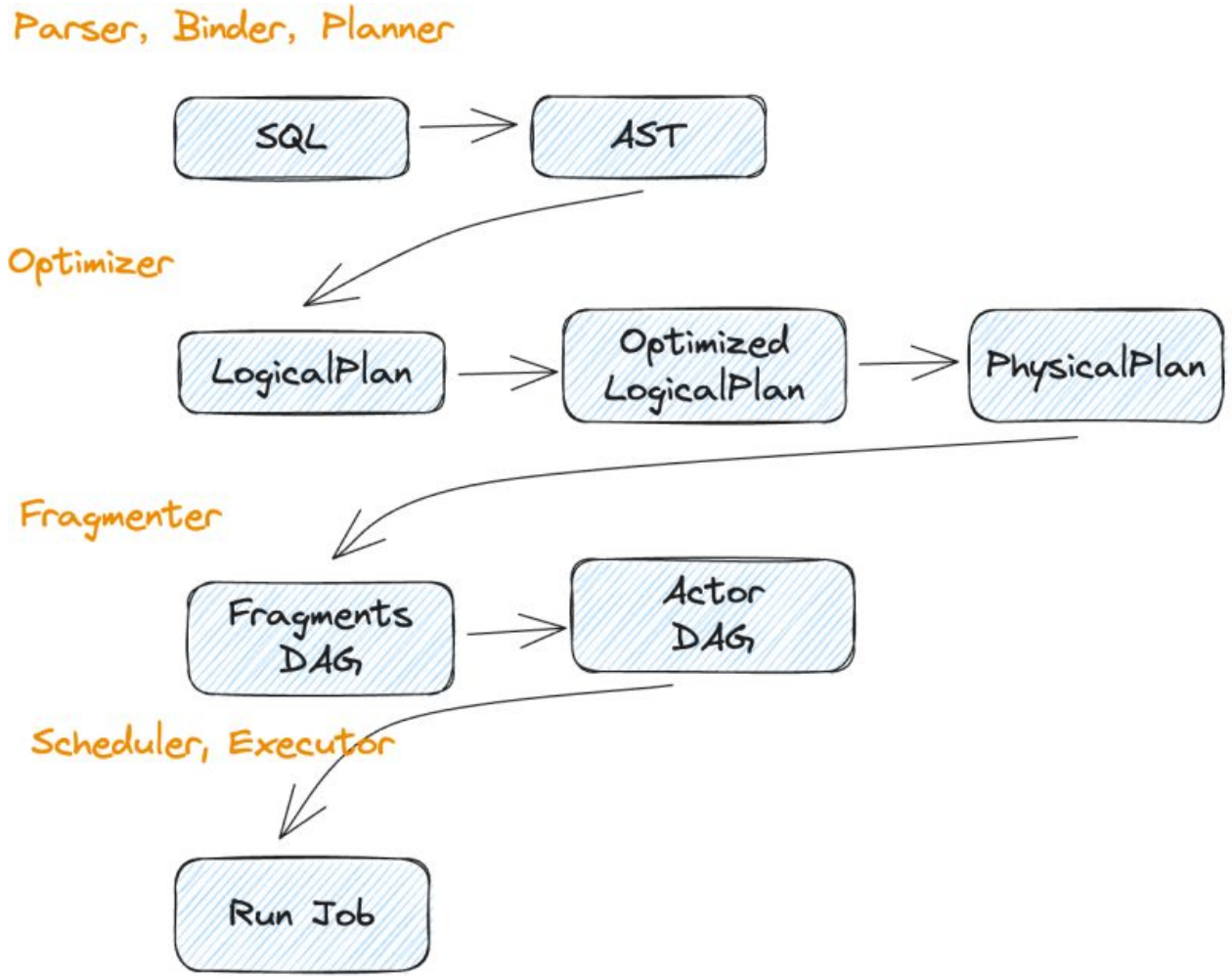
- Watermark、Window Join等高级Streaming特性不一定适用
  - 宽表的更新不一定有很强的时间局域性
  - 历史数据的修复订正也需要同步到宽表

- SQL Inner/Outer Join更适用于打宽场景
  - 标准SQL（PG）可轻松构建Streaming Pipeline
  - 大状态是常态



RisingWave：SQL即流处理

- 基于SQL构建流作业
- 丰富的查询优化
  - 列裁剪，Filter下推
  - 子查询解关联
  - Join Ordering
  - Agg改写
  - 共享SubPlan
  - 公共表达式提取
  - 自动索引选择
  - Streaming改写



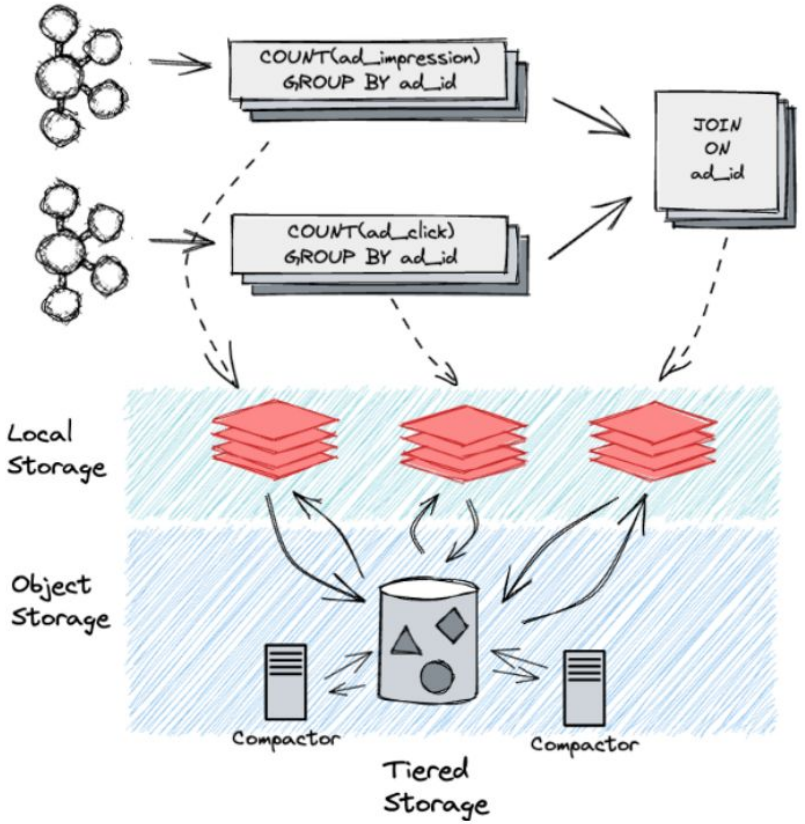
RisingWave基础概念：MATERIALIZED VIEW

- Materialized View增量实时维护流处理结果
- 支持MV-on-MV构建层级Streaming Pipeline
- 支持丰富的SQL语法
  - Join, Agg, Filter, 集合运算, 窗口函数, 子查询, UDF, Grouping Sets, CTE, Lateral, Watermark, ...
  - PG常见表达式, Lambda表达式, 半结构化数据处理函数
- 支持实时查询Materialized View结果

```
CREATE MATERIALIZED VIEW ad_ctr AS
SELECT
    ad_clicks.ad_id AS ad_id,
    ad_clicks.clicks_count :: NUMERIC / ad_impressions.impressions_count AS ctr
FROM
    (
        SELECT
            ad_impression.ad_id AS ad_id,
            COUNT(*) AS impressions_count
        FROM
            ad_impression
        GROUP BY
            ad_id
    ) AS ad_impressions
    JOIN (
        SELECT
            ai.ad_id,
            COUNT(*) AS clicks_count
        FROM
            ad_click AS ac
            LEFT JOIN ad_impression AS ai ON ac.bid_id = ai.bid_id
        GROUP BY
            ai.ad_id
    ) AS ad_clicks
    ON ad_impressions.ad_id = ad_clicks.ad_id;
```

RisingWave：多流Join

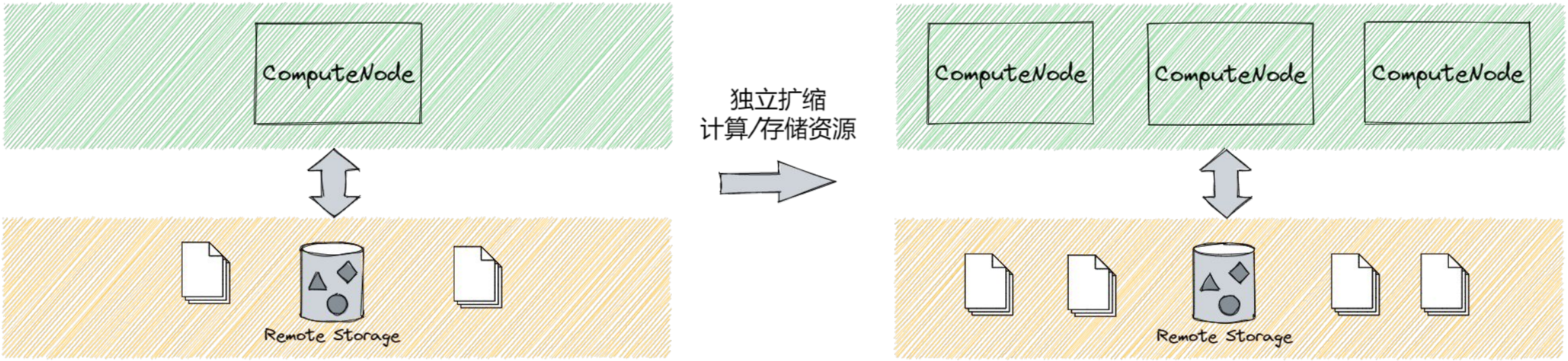
- 多流Join
  - Regular Join：Inner + Outer
  - Interval Join
  - Temporal Join
  - Windows Join (Watermark)
- TB级别长周期状态管理
  - 算子状态持久化在对象存储，无单机状态上限
  - 基于Shared Storage的自研存储引擎



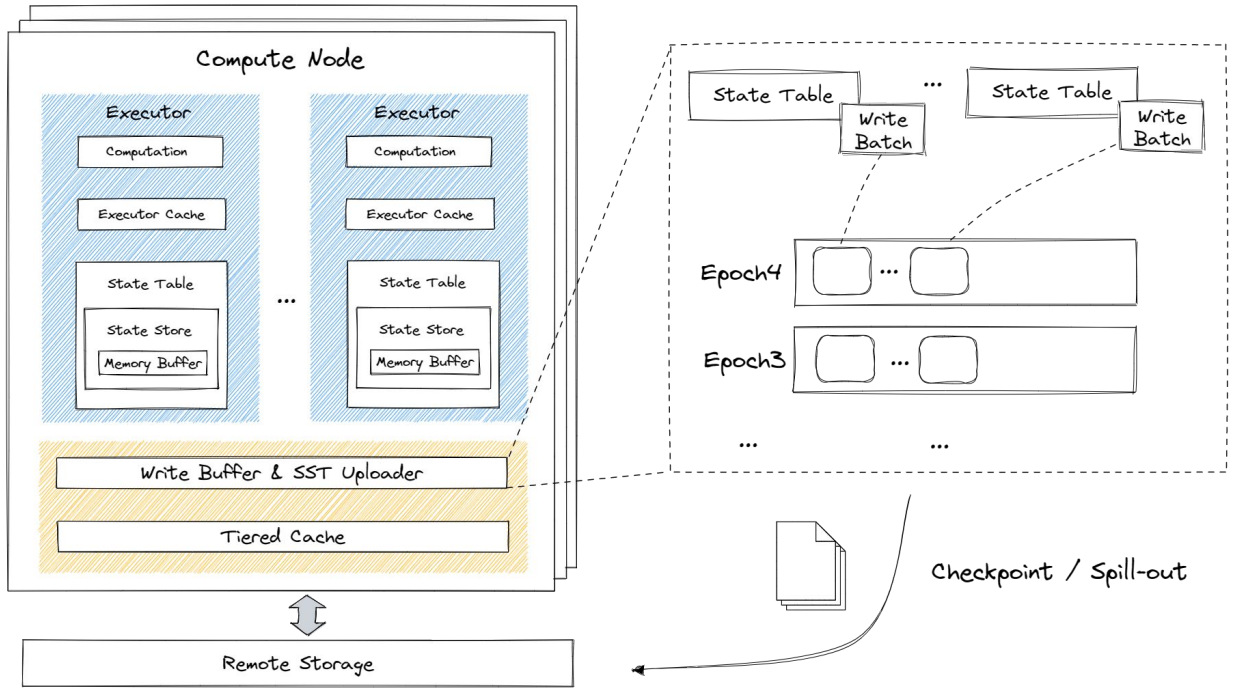
- 生产案例：基于多流Join构建CDC流表打宽
  - 集群包含10+个包含多流Join的物化视图，其中最复杂的包含20路多流Join
  - 状态总量~500GB

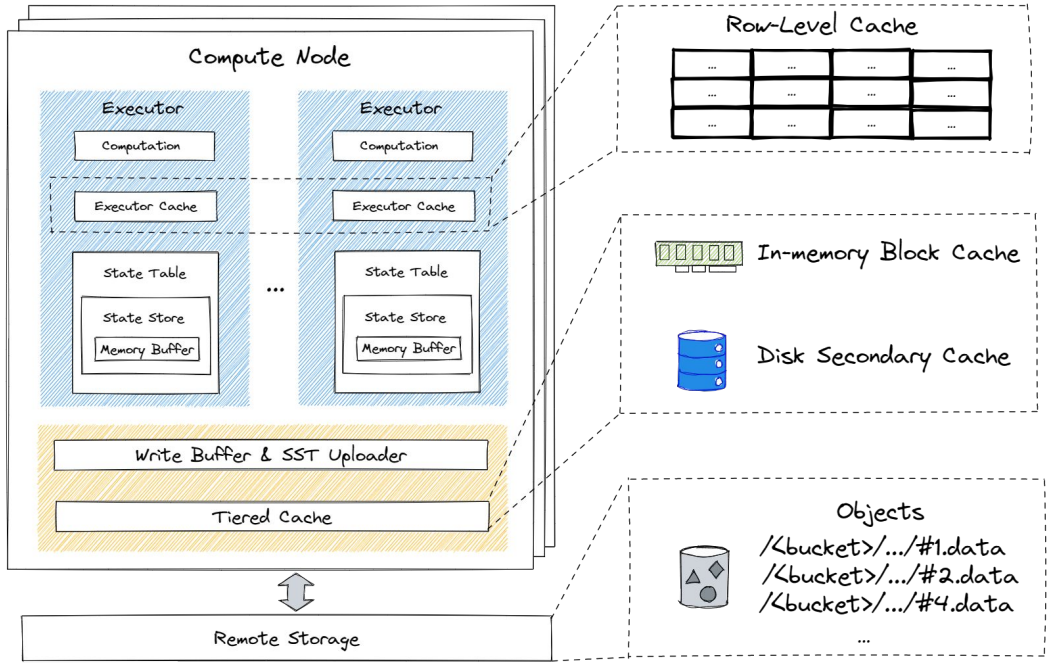
存算分离：大状 态 != 高成本

- 状态存储持久化在成本低廉的远端Remote Object Store
- 存算分离的架构让计算和存储资源独立扩缩，且实现秒级扩缩容，无须腾挪数据
- 自研云原生LSM存储引擎，支持Serverless Compaction，根据compaction负载auto scale



Tiered Cache：大状态 != 低性能



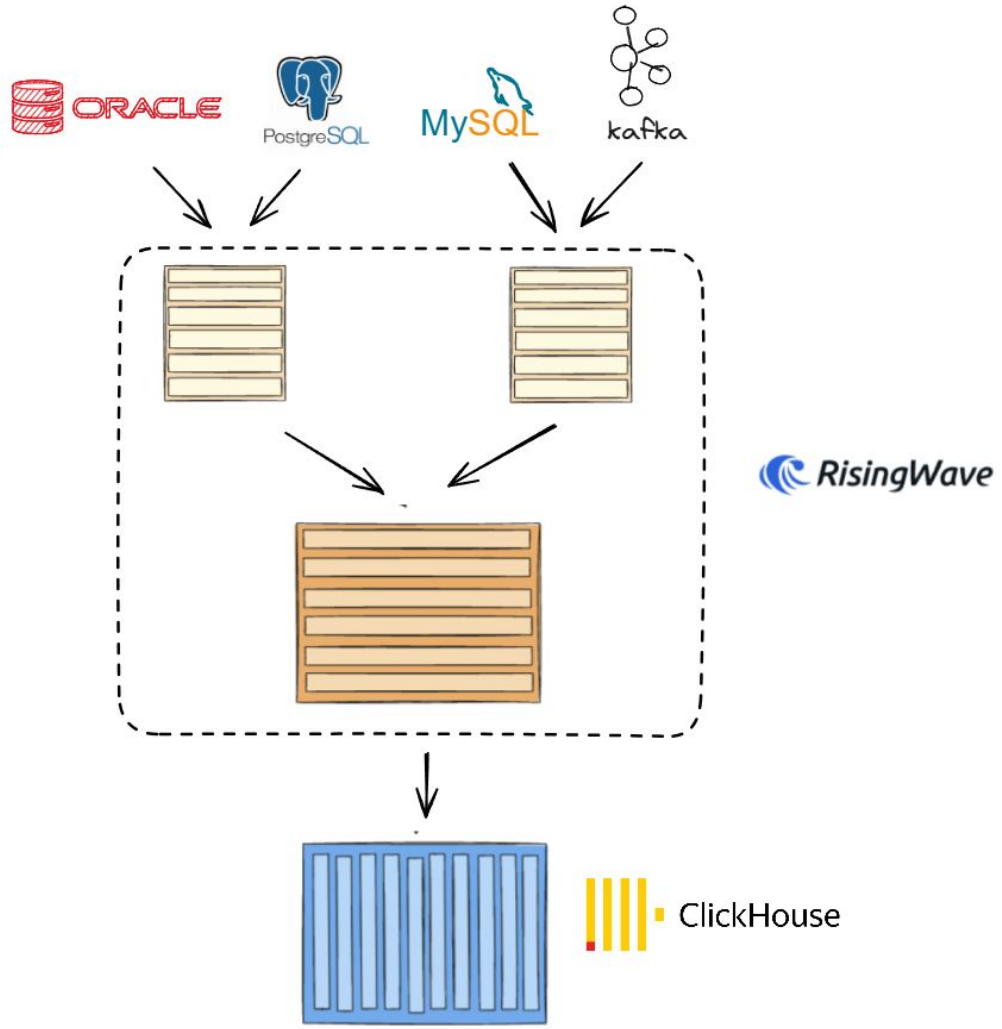


- 计算节点维护Write Buffer攒批写入远端存储
- Checkpoint触发Force Flush
- 大状态可提前Spill

- 计算节点维护Tiered Cache提升读性能
- 热数据缓存在最靠近计算的算子Row Cache
- 温数据缓存在节点的Block Cache

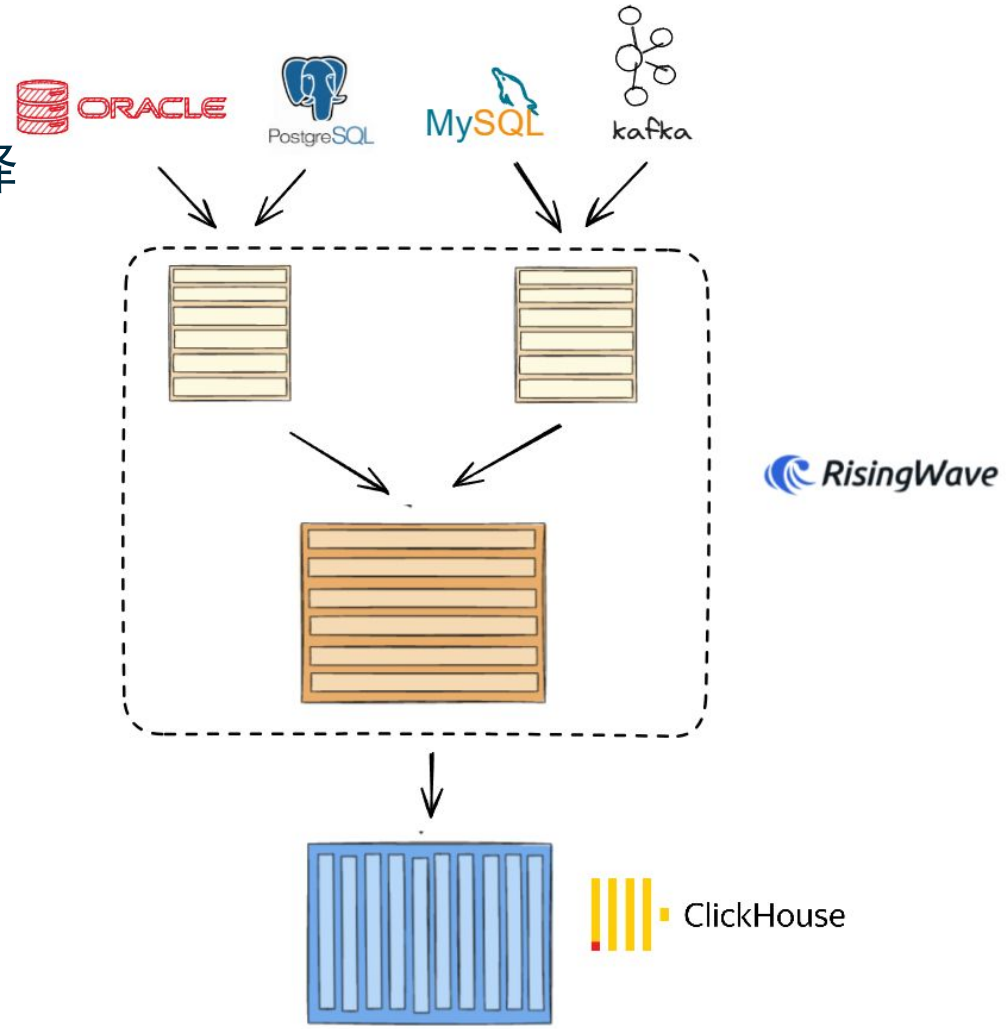
实时打宽挑战#3：流式数仓写入

过去，许多OLAP系统只能支持批量写入，无法高效支持近实时写入与更新（upsert）



实时打宽挑战#3：流式数仓写入

- ClickHouse天然支持流式写入，同时丰富的表引擎选择让流式写入更为灵活
- RisingWave通过Sink对ClickHouse进行流式写入
  - CollapsingMergeTree引擎家族与ReplacingMergeTree引擎家族支持upsert写入
  - 其余引擎家族支持append-only写入
- RisingWave支持“写解耦”(Sink Decoupling)
  - 更灵活地攒批写入下游Sink
  - 同时隔离下游Sink故障和流作业



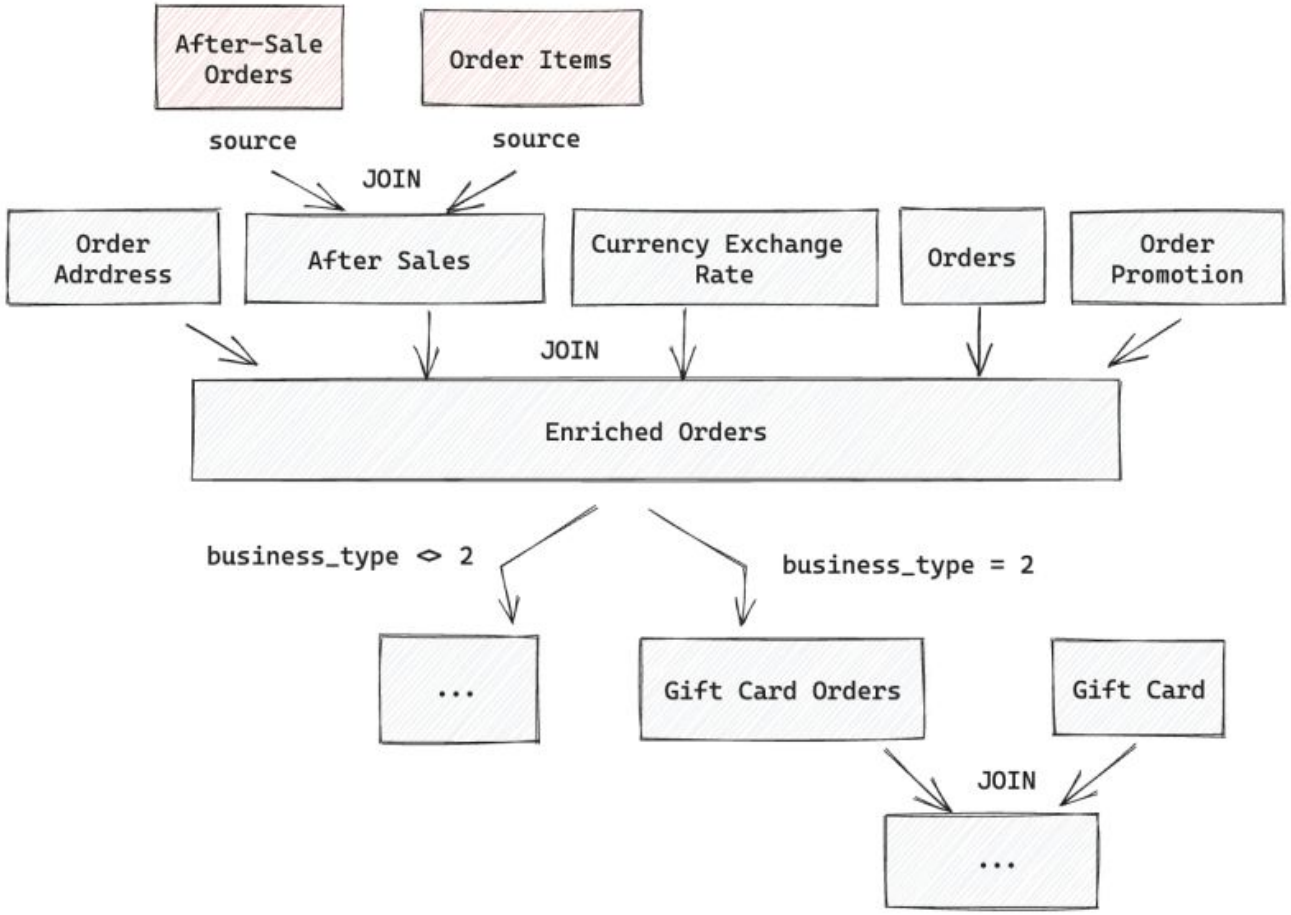
RisingWave基础概念：SINK

- 通过Sink发送数据到多种下游系统
- 支持的Connector：Kafka、JDBC、Redis、Clickhouse、StarRocks、Doris、ElasticSearch、Cassandra、Iceberg...
- 支持的Format：APPEND_ONLY、UPSERT、DEBEZIUM
- SINK的输入可以是Table/Materialized View，也可以是SQL query

```
CREATE SINK sink_clickhouse
FROM impression_cnt
WITH (
    connector = 'clickhouse',
    type = 'upsert',
    primary_key = 'ad_id',
    clickhouse.url = '${CLICKHOUSE_URL}',
    clickhouse.user = '${CLICKHOUSE_USER}',
    clickhouse.password = '${CLICKHOUSE_PASSWORD}',
    clickhouse.database = 'default',
    clickhouse.table='impression_cnt'
);
```

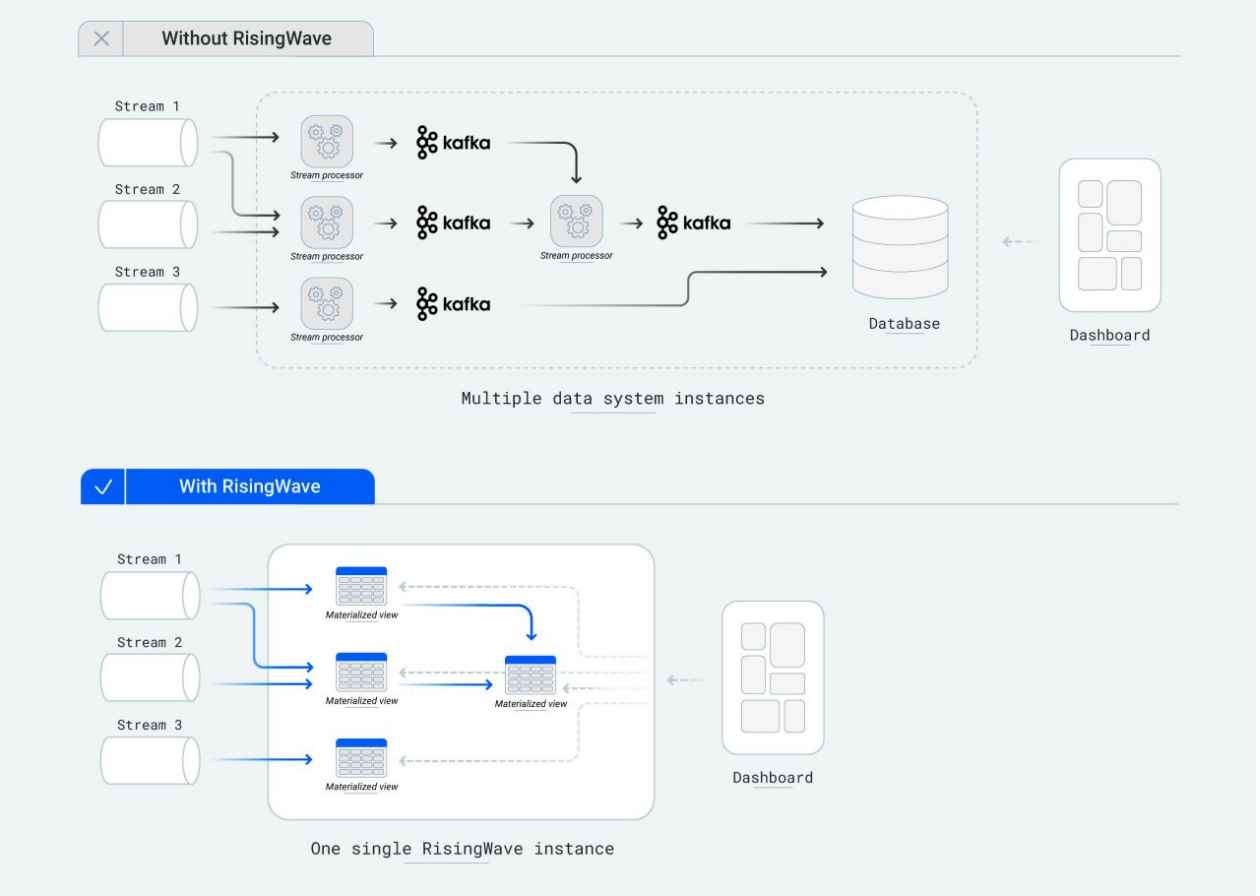
基于宽表做进一步实时分析：多层数据表

- 实时分析链路中往往有分层结构
  - 分层构建打宽链路
  - 初步流表打宽后，不同业务对同一宽表有不同的实时加工需求
- RisinWave支持MV-on-MV分层构建流作业，不需要经过Kafka中转
- RisingWave支持DML对Table进行数据订正，下游MV数据自动根据依赖修复



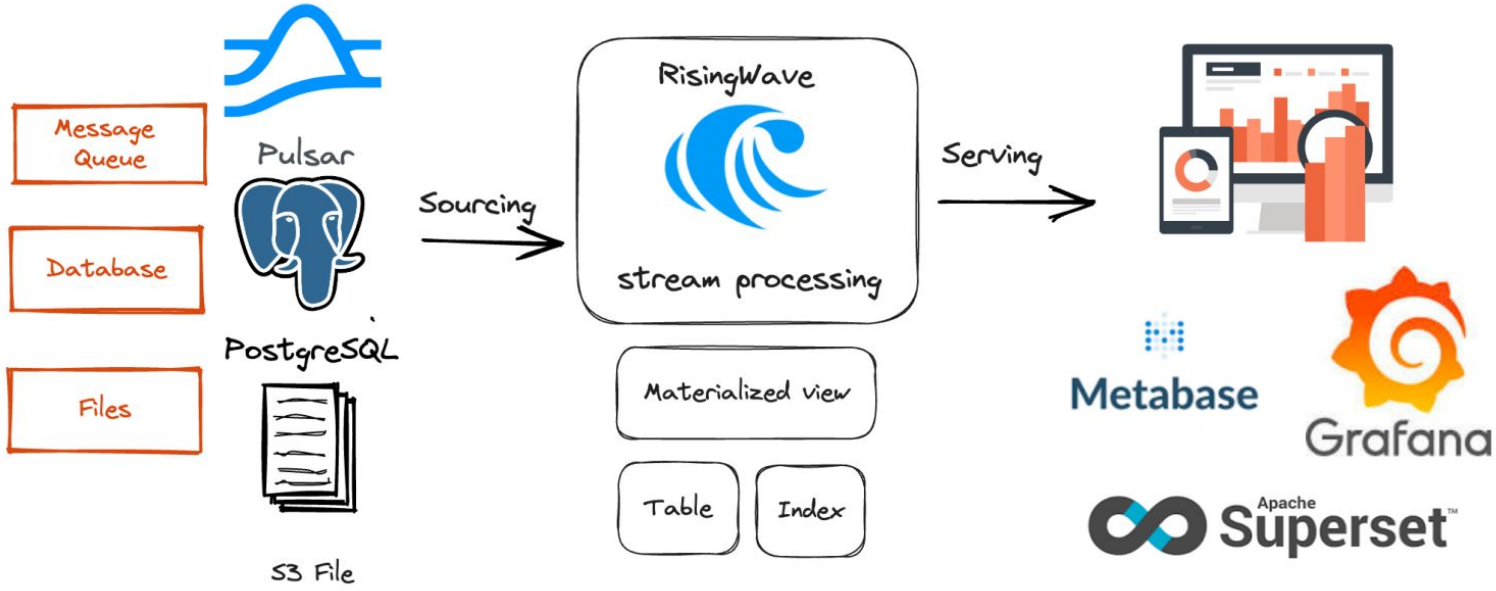
基于宽表做进一步实时分析：多层数据表

- RisinWave支持MV-on-MV分层构建流作业，不需要经过Kafka中转
- RisingWave支持DML对Table进行数据订正，下游MV数据自动根据依赖修复



基于宽表做进一步实时分析：实时看板

- RisingWave Table/Materialized View结果支持SQL查询
- 支持Serving和简单批查询、支持Streaming和Serving隔离
- 用户可基于宽表构建下游MV，轻松对接Metabase/Superset等BI工具构建实时看板



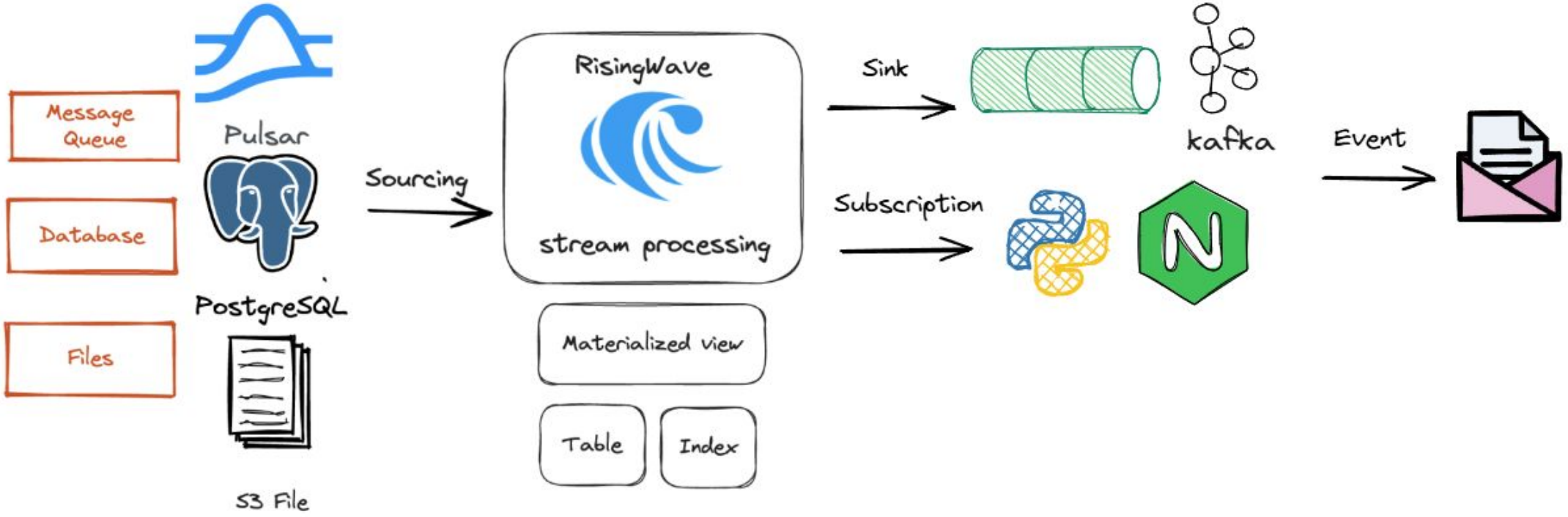
RisingWave基础概念：INDEX

- 索引可以创建在Table和Materialized View上
- 加速Serving查询
- 支持指定Include列、Distributed列
- 支持表达式索引
- 自动索引选择

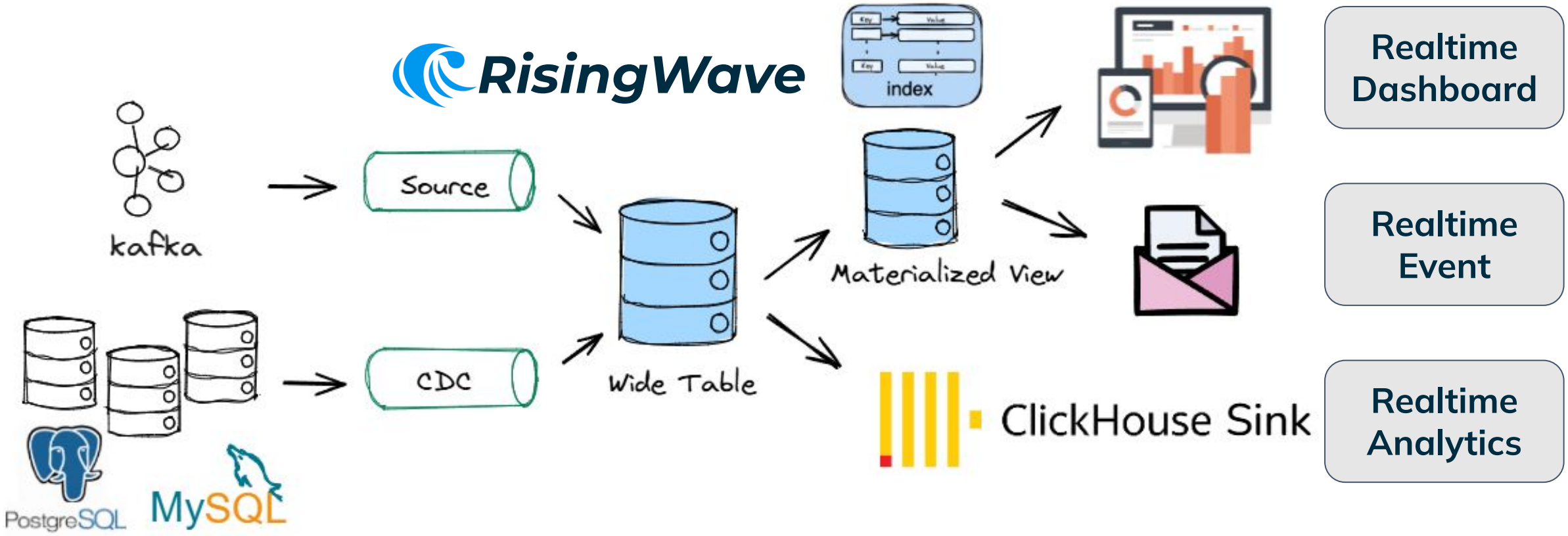
```
-- customers 加速点查
CREATE INDEX idx_c_phone on customers(c_phone);

SELECT * FROM customers where c_phone = '123456789';

-- orders 加速join
CREATE INDEX idx_o_custkey on orders(o_custkey);

SELECT * FROM customers JOIN orders ON c_custkey = o_custkey;

-- json 表达式加速
CREATE TABLE t (j jsonb, v1 int, v2 int);
CREATE INDEX idx1 on t(j->>'k1');

SELECT * FROM t WHERE j->>'k1' = 'abc';
```

基于宽表做进一步实时分析: Event Notification

- RisingWave MV变更支持Sink到下游MQ
- RisingWave MV变更支持通过Subscription进行订阅
- 用户可基于宽表构建下游MV，通过MQ/Subscription订阅MV变更，构建事件驱动的业务与应用。如规则引擎、异常告警、指标监控等。



RisingWave x ClickHouse：实时分析新范式



RisingWave 1.x -> 2.0

- RisingWave Premium
  - Enterprise Edition + Premium Support for Self-Hosting RisingWave Cluster
- Unified experience for Streaming & Batch
- Empower Event-Driven Applications
- Automatic Schema Mapping & Schema Change
- Severless Backfill
- More SQL features

START YOUR STREAM PROCESSING JOURNEY WITH

A SINGLE LINE OF CODE!

Install RisingWave for macOS and Linux

```
curl https://risingwave.com/sh | sh
```

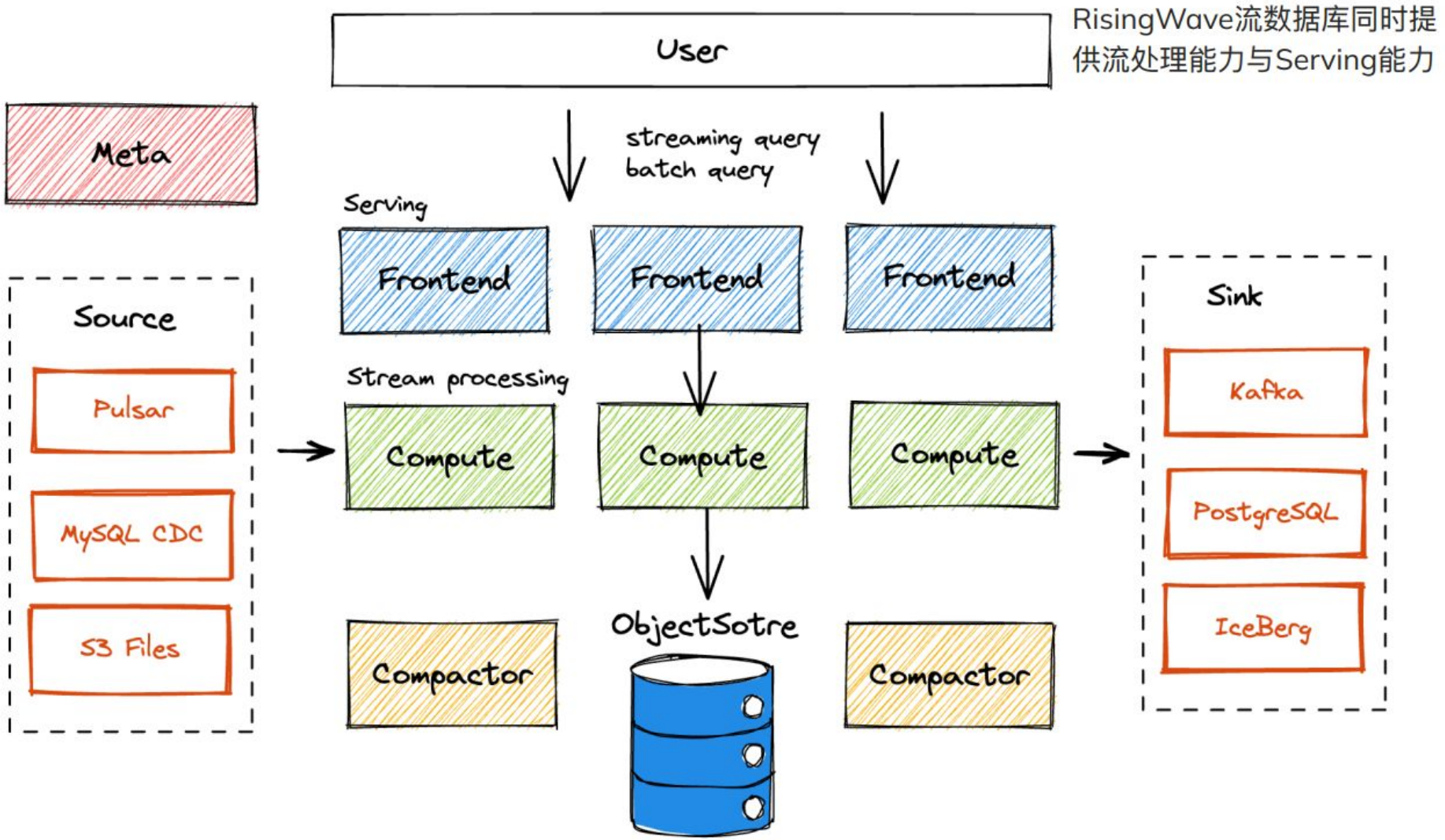
关注 RisingWave 订阅号
获取更多

扫一扫进入
社区用户微信交流群

GitHub:	**官网:**	**Slack:**	**B站:**	**知乎:**
risingwave.com/github	risingwave.com	risingwave.com/slack	RisingWave 中文开源社区	RisingWave 中文开源社区

RisingWave

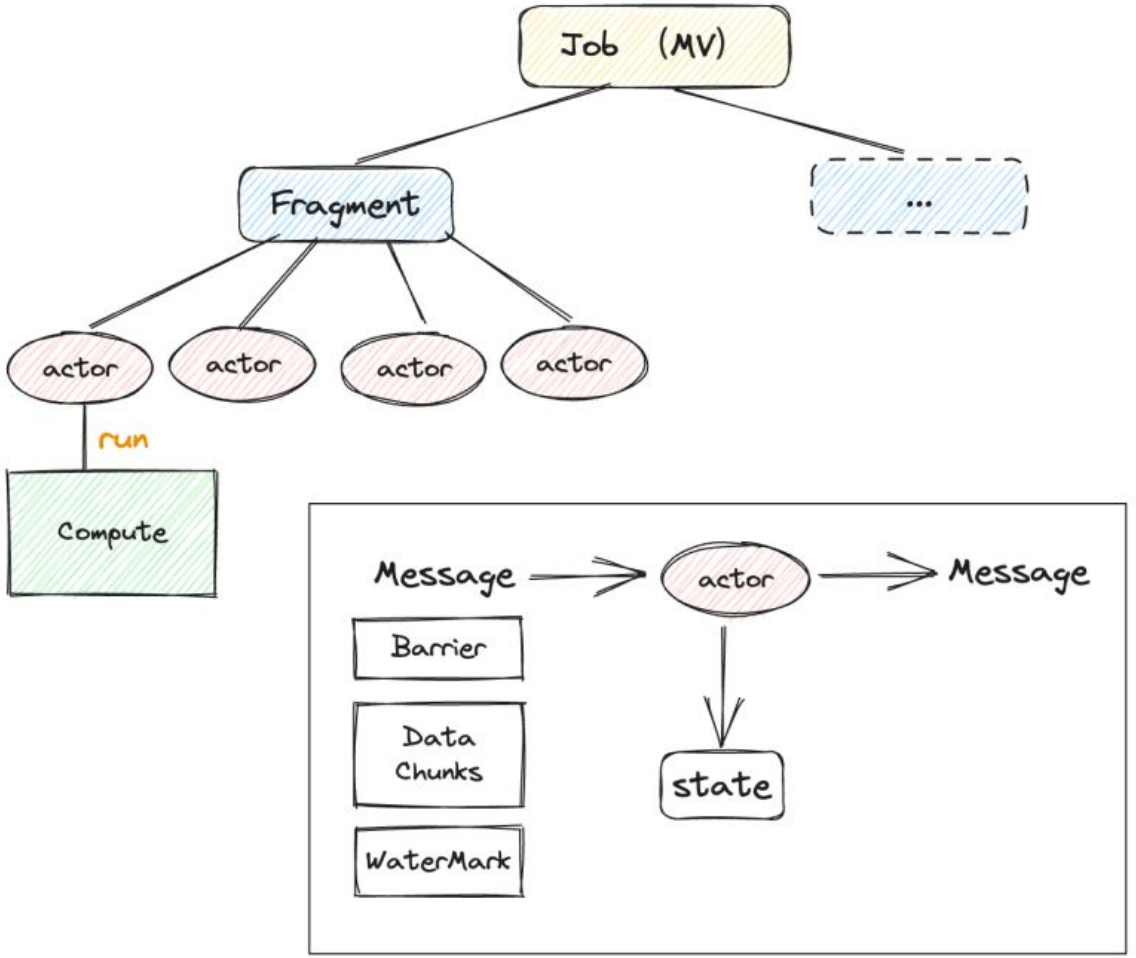
Backup Slides：RisingWave架构



Backup Slides：执行器

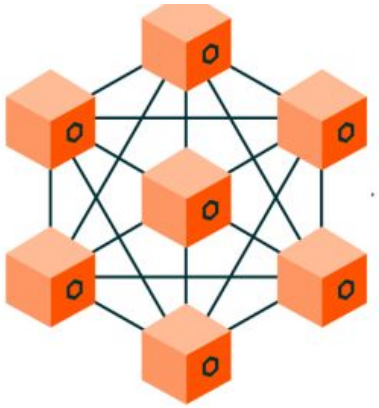
- 根据算子特性和集群规模自动划分Fragment和并行度

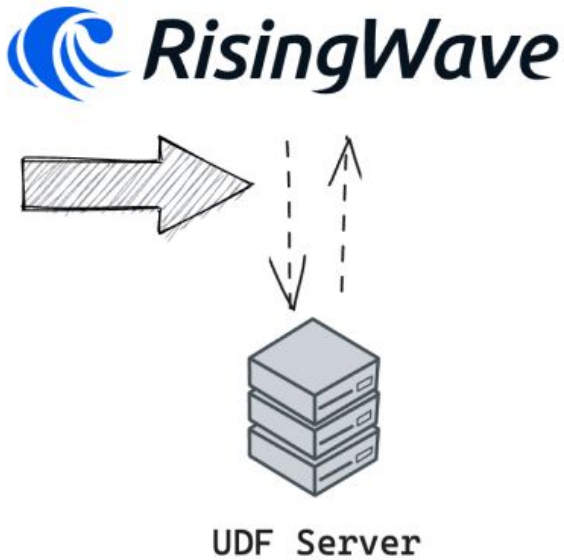
- 基于Actor模型实现并发计算
  - 任务Fragment调度到Actor执行
  - Actor单线程，避免并发问题
  - Actor间传递消息通讯
  - Actor有独立状态



Backup Slides：UDF

- RPC：支持Python，Java UDF Server进行自定义流处理逻辑
- Embedded：支持WASM，Javascript，Python，SQL等Embedded UDF



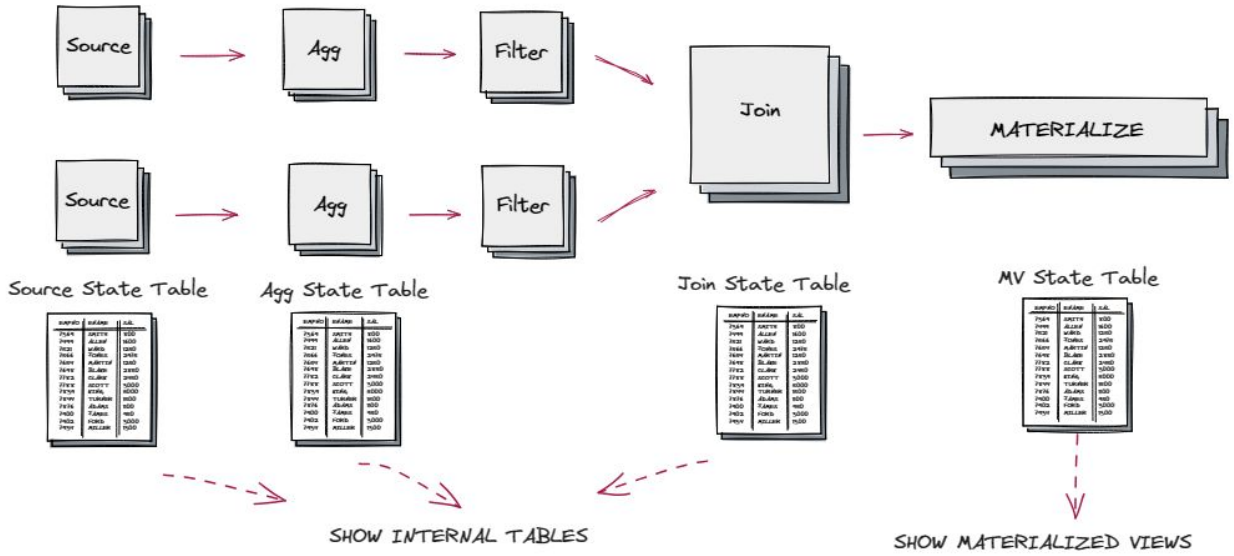


- 生产案例：基于Python UDF构建实时风控指标

Backup Slides：Queryable State

- 流算子内部状态抽象成关系型State Table
- SHOW INTERNAL TABLES查看表名
- State Table支持SQL查询

- 适用场景
  - 排查线上数据问题
  - 优化流作业SQL
  - 学习流算子的状态管理机制




```
dev=> SHOW INTERNAL TABLES;
                                                  Name
-----------------------------------------------------------------------------------------------
 __internal_demo_cdc_4_source_1005
 __internal_ad_events_2_source_1003
 __internal_user_impression_count_6005_hashjoindegreeleft_7007
 __internal_user_impression_count_6005_hashjoindegreeright_7009
 __internal_user_impression_count_6005_hashjoinleft_7006
 __internal_user_impression_count_6007_streamscan_7011
 __internal_pg_ads_6002_source_7003
 __internal_user_impression_count_6004_hashaggstate_7005
 __internal_user_impression_count_6006_streamscan_7010
 __internal_user_impression_count_6005_hashjoinright_7008
(10 rows)
```

Backup Slides: Queryable State

- 流算子内部状态抽象成关系型State Table
- SHOW INTERNAL TABLES查看表名
- State Table支持SQL查询

```
dev=> SELECT * FROM __internal_user_impression_count_6004_hashaggstate_7005 limit 5;
          ad_events_user_id           | count
--------------------------------------+-------
 000c5060-5130-4674-9e44-6db8b4a7c17c |   270
 0011512d-893b-4103-bad8-9f1c04476a96 |   260
 001f3398-5be3-43ef-a71c-300c399f64a0 |   258
 00213256-69f7-487c-aff3-8e0912bf7f0d |   290
 00234e5d-b9ab-41e9-8271-5206bc27aecc |   253
(5 rows)
```

- 适用场景
  - 排查线上数据问题
  - 优化流作业SQL
  - 学习流算子的状态管理机制

```
dev=> SELECT * FROM __internal_ad_events_2_source_1003;
 partition_id |                                offset_info
--------------+-------------------------------------------------------------------
 0            | {"split_info": {"partition": 0, "start_offset": 3496935, "stop_offs.
              |.et": null, "topic": "ad-events"}, "split_type": "kafka"}
(1 row)
```