

TIS

一站式基于TIDB的搜索中台(terminator index search)

百岁

大纲

背景
功能
架构
实现

背景

背景

现状

TiDB 是一款定位于在线事务处理/在线分析处理（HTAP: Hybrid Transactional/Analytical Processing）的融合型数据库产品，实现了一键水平伸缩，强一致性的多副本数据安全，分布式事务，实时 OLAP 等重要特性。同时兼容 MySQL 协议和生态，迁移便捷，运维成本极低

潜在的无法满足的需求

- 1.基于TiDB的模糊查询
- 2.OLAP场景下无法进行进行基于地理位置的匹配和排序
- 3.等等，基于传统搜索引擎之上做的操作

现状

如何补短板

用户通过ES和TiDB打通，在自己企业内部实现企业应用搜索功能

存在问题

- 1.都是做项目性质，全部黑屏化操作。没有形成行业标准，各家企业重复造轮子，浪费资源
- 2.搜索技术门槛相对较高，BAT大企业已经形成人才黑洞，中小企业很难找到满足要求的搜索人才
- 3.随着大数据到来，企业内部亟需搜索引擎技术为企业各条业务先提供支持

功能

功能

将搜索引擎技术中台化

1. 将数据库相关的技术构建成基于TiDB的一站式，
开箱即用的中台产品
2. 通过中台产品将团队角色由全职保姆转变成搜索
技术咨询师

特性

一键安装

利用ansible实现分布式环境中一键安装、更新等操作

丰富的插件机制

基于UI的插件机制，方便扩展Solr、Lucene的底层功能扩展点。适应私有云，公有云、混合云环境

一站式开箱即用

TIS安装完成之后，用户就可以轻松在TIS平台上对索引实例进行，创建、数据结构(Schema)更新，数据刷新操作、以及最终的删除等操作

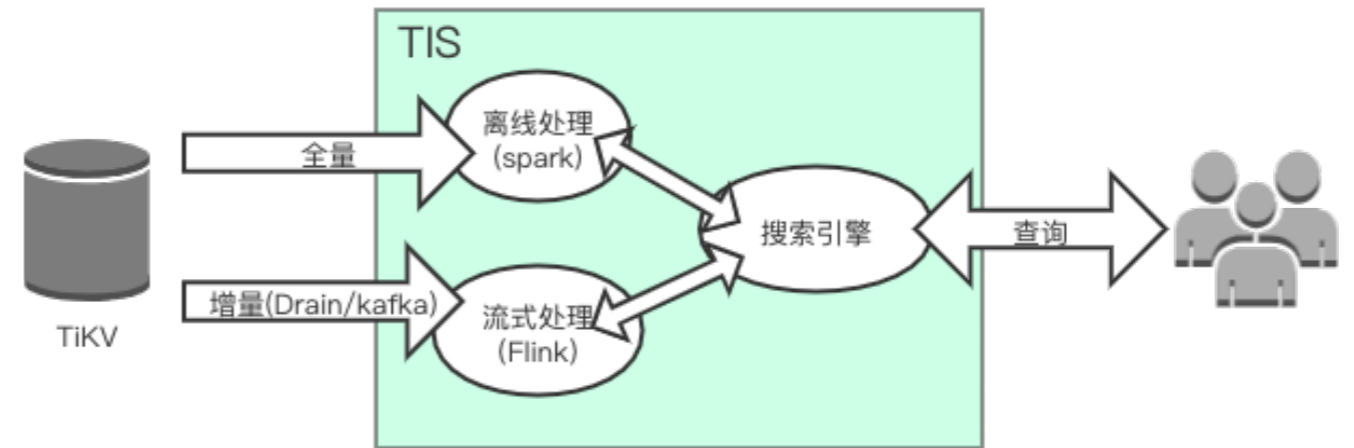
批流统一

通过配置一个数据流规则，可以自动生成全量数据构建(批处理)和增量管道(流处理)脚本，且能自动部署，大幅提交工作效率

架构

架构

TIS整体分为三个组件



1. 离线处理

如已安装TiSpark则直接使用基于TiKV的RDD构建宽表，如未安装则可以利用插件切换到Aliyun MaxComput，或者自建Hive，Spark环境通过TiKV导出数据构建宽表

2. 流式处理

基于FlinkSql构建索引增量通道

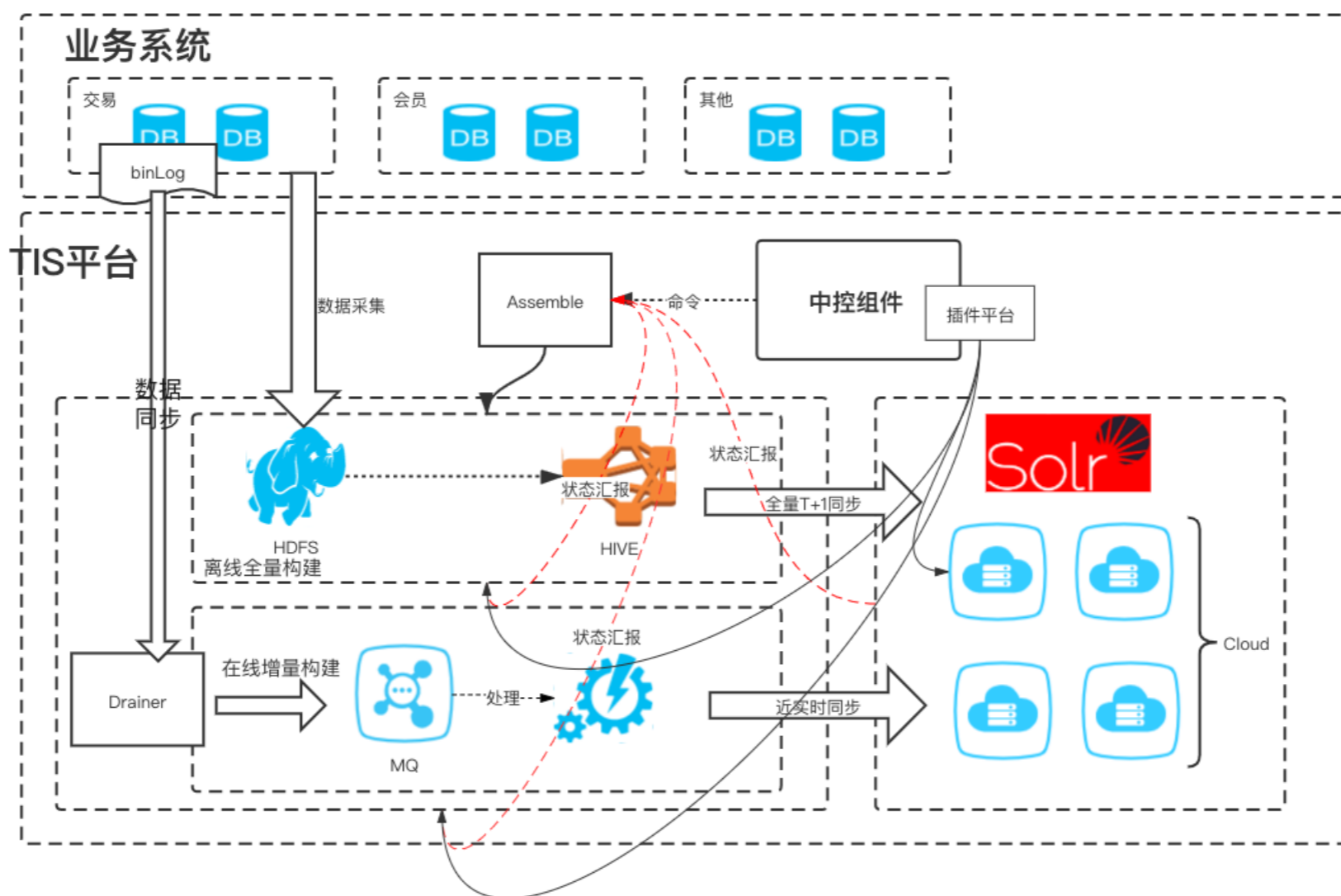
3. 搜索引擎

离线构建的索引全量和增量通道产生的数据汇聚到搜索引擎中，现搜索引擎基于Solr

以上三个组件无缝整合，在用户视角感觉不出三个组件时间界限

架构详细

1. 中控组件中定义业务TiDB数据源
2. 中控组件中可定义各种插件给离线全量、在线增量构建通道、搜索引擎节点使用
3. 通过中控组件可观测TIS中各组件的状态，并且可以下发各种命令



依赖的语言、构件

前台界面

Angular9.0 、 ng-zorro、 g6等

服务层

Java1.8、 scala、 struts2、 spring4、 solr等

实现



1. 定义数据源
2. 定义数据流
3. 索引实例
4. 增量通道
5. 插件管理

实现-数据源定义

The screenshot displays a web-based interface for managing data sources. At the top, there is a navigation bar with a logo and three menu items: '我的索引', '基础管理', and '离线数据'. Below this, a breadcrumb trail shows '数据源管理'. On the left side, there is a search bar and a tree view containing 'employees' and 'dept_emp'. The 'employees' item is highlighted with a blue selection bar and a red circle. A red arrow points from this circle to the 'SQL' field in the table information panel on the right. The table information panel is titled '表信息' and contains a table with two rows: '表名' (employees) and '数据库名' (employees). Below the table is a large text area labeled 'SQL' containing the following query:

```
SELECT emp_no,birth_date,first_name,last_name,gender,hire_date
FROM employees
```

实现-定义数据流

保存设置名称

The screenshot displays a data flow configuration interface. In the background, a data flow diagram is visible on a grid, showing a source node labeled 'supply_commodityxxxx' (represented by a gear icon) connected to a target node labeled 'employees' (represented by a document icon). A modal dialog box titled '设置数据流名称' (Set Data Flow Name) is centered on the screen. The dialog has a close button (X) in the top right corner. It contains a text input field with the label '* 名称:' (Name) and the value 'test_flow'. At the bottom of the dialog are two buttons: '取消' (Cancel) and '确定' (Confirm). In the top right corner of the main interface, there are two buttons: '保存' (Save) and '关闭' (Close).

实现-索引实例

对索引进行查询

The screenshot shows a search interface with a sidebar on the left and a main content area. The sidebar contains several menu items: 主控台, 查询, 配置变更, 实时通道, 全量构建, 监控, and 操作历史. The '查询' (Query) item is circled in red. The main content area has a search bar with the text 'POJO' and a dropdown menu. Below the search bar, there are several configuration options: Sort (create_time desc), Start/Rows (0 / 3), Distrib (引擎节点), Debug (off), Facet (checked), and FQ (facetField, facetPrefix, facetQuery). The '高级 ^' (Advanced ^) link is circled in red. The search results are displayed in a table with columns for IP address, dept_no, from_date, to_date, emp_no, and _version_.

search4test2

POJO

query:Solr查询语法

:

Sort: create_time desc Start/Rows: 0 / 3

Distrib: 引擎节点 Debug:

Facet: facetField facetPrefix facetQuery FQ: + Add

Raw: FI: 选择

Query 命中:329601 高级 ^

[0]192.168.28.200 dept_no:d007 from_date:1995-02-19 to_date:9999-01-01 emp_no:10455 _version_:20201014161509

[0]192.168.28.200 dept_no:d002 from_date:2000-01-06 to_date:9999-01-01 emp_no:10456 _version_:20201014161509

[0]192.168.28.200 dept_no:d006 from_date:1999-10-23 to_date:9999-01-01 emp_no:10457 _version_:20201014161509

点击打开查询页面

点击打开高级搜索项设置

实现-创建增量通道

创建成功可观测增量执行的流量等指标



实现-插件配置

- 借鉴了jenkins的插件实现机制-基于UI的插件管理平台
- 将TIS中各可变化功能点进行抽象，形成扩展点可进行扩展
- 在TIS平台中对接口参数实例化

存储 宽表构建 索引构建容器 数据导出

保存

▼ 存储

com.qlangtech.tis.offline.FileSystemFactory

com.qlangtech.tis.hdfs.impl.HdfsFileSystemFactory

* name:

* hdfsAddress:

* rootDir:

* hdfsSiteContent:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <property>
    <name>dfs.nameservices</name>
    <value>daily-cdh</value>
  </property>
  <property>
    <name>dfs.client.failover.proxy.provider.daily-cdh</name>
    <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
  </property>
  <property>
    <name>dfs.ha.automatic-failover.enabled.daily-cdh</name>
    <value>true</value>
  </property>
  <property>
    <name>dfs.ha.namenodes.daily-cdh</name>
    <value>namenode228,namenode295</value>
  </property>
  <property>

```

添加 ▼

> 宽表构建

谢谢观看