Spark Streaming 강의 스크립트

슬라이드 1: 강의 시작

今天我们要学习的是 Spark Streaming (스파크 스트리밍)。

今天的内容会比较丰富，我们将从基础概念开始，一步一步深入到实际的编程实践中。

请大家准备好电脑，跟着我一起来探索流式数据处理 (stream processing) 的世界。

SL -02 正式课程大纲

好的，我们看到正确的课程大纲了。今天我们要学习的 Spark Streaming 主要包括以下几个部分：

8.1 概述 (Overview) - 我们会从基本概念开始

8.2 编写Structured Streaming程序的基本步骤 - 这是实践部分

8.3 输入源 (Input Sources) - 了解数据从哪里来

8.4 输出操作 (Output Operations) - 学习结果如何输出

8.5 容错处理 (Fault Tolerance) - 这是企业级应用的关键

8.6 迟到数据处理 (Late Data Handling) - 处理实际场景中的复杂情况

8.7 查询的管理和监控 (Query Management and Monitoring) - 生产环境必备技能

SL-03 슬라이드 3: 8.1 概述 - 章节导入

好，现在我们正式开始第8.1节的学习。概述部分我们分为三个小节：

- **8.1.1 基本概念 (Basic Concepts)** - 理解什么是流式处理(**chǔ lǐ**)
- **8.1.2 两种处理模型 (Two Processing Models)** - 微批处理和持续(**chí xù**)处理
- **8.1.3 Structured Streaming和Spark SQL、Spark Streaming关系** - 理解它们之间的联系和区别(**qū bié**)

这三个部分将帮助大家建立对 Structured Streaming 的整体认识。

SL-04 슬라이드 4: 8.1.1 基本概念

这是今天课程的核心概念，请大家仔细听。

Structured Streaming 的关键思想是什么？

就是将实时数据流（real-time data stream）视为一张正在不断添加数据的表（unbounded table）。

这个概念很重要，我再说一遍：**数据流就等于无界表，数据流中的新数据就等于附加到无界表上的新的数据行**。

你们看这个图，左边是传统的数据流概念 —— 一个一个的数据包在流动。

右边是 Structured Streaming 的概念 —— 把这些数据看作是不断向表格中添加新行。

这样做有什么好处呢？

我们可以把流计算(stream computing) 等同于在一个静态表上的批处理查询(batch query)。

Spark会在不断添加数据的无界输入表上运行计算，并进行增量查询 (incremental query)。

这就像你在Excel表格中不断添加新行，然后对整个表格进行统计分析一样简单！

SL-05 슬라이드 5: 处理模型详解

现在我们来看看这个处理模型是如何工作的。

大家看这个图，很清楚地展示了 **Structured Streaming编程模型**：

时间轴(shí jiān zhóu)：每秒触发一次 (trigger once per second)

在 **t=1时刻**：

- 输入：所有到达的数据
- 查询：对这些数据进行处理
- 结果：得到t=1时刻所有数据的处理结果
- 输出：将结果输出

在 **t=2时刻**：

- 输入：包括之前的数据加上新到达的数据
- 查询：对累积**(lěi jī)**的所有数据进行处理
- 结果：得到t=2时刻所有数据的处理结果
- 输出：输出更新后的结果

注意这里的关键点：每次都是对**所有数据**进行查询，不是只处理新数据。这保证了结果的一致性 (consistency)。

这就是为什么叫"无界表" —— 表格在不断增长，但查询逻辑保持不变。

SL-06 슬라이드 6: 8.1.2 两种处理模型 - 微批处理

现在我们来学习两种不同的处理模型。首先是**微批处理 (wēi pī chǔ lǐ Micro-batch Processing)**。

Structured Streaming 默认使用微批处理执行模型(mó xíng)。什么意思呢？

想象一下，Spark流计算引擎 (stream computing engine) 就像一个勤劳**(qín láo)**的工人，它会：

1. **定期检查**流数据源 (stream data source)
2. **收集**自上一批次结束后到达的新数据
3. **执行批量查询**处理这些数据

大家看这个图：

- Spark引擎在中心
- 周围有日志 (log) 管理
- 数据流进来后，被分成一个个**微批处理 (micro-batch)**
- 每个微批处理启动多个**短任务 (rèn wù short tasks)**

重要特点：在微批处理之前，待处理数据的偏移量 (**piān yí liàng** offset) 会写入预写日志 (write-ahead log)，这是为了容错。

延迟时间：数据到达和得到处理并输出结果之间的延时超过100毫秒。这对大多数应用来说已经足够快了。

SL-07 슬라이드 7: 持续处理模型

第二种模型(mó xíng)是**持续处理（Continuous Processing)**，这是从Spark 2.3.0版本开始引入的实验性(shí yàn xìng)功能。

为什么需要持续处理？因为有些应用需要**毫秒级延迟（millisecond latency)**！

持续处理模式下，Spark的工作方式完全不同：

- **不再**根据触发器来周期性启动任务
- **而是**启动一系列的连续读取、处理和写入结果的长**时间运行的任务（long-running tasks)**

看这个图：

- 引擎在查询开始后启动一系列长时间运行的任务
- 数据到来后**立刻**被任务处理
- 各个"时期（epoch)"的已处理数据偏移量被**异步**写**入**预写日志

这就像流水线生产 —— 工人一直在工作台前，零件(**零件língjiàn** 부품)一来就立刻处理，而不是等积累一堆再处理。

关键优势：可以实现毫秒级的实时响应！

SL-08 슬라이드 8: 8.1.3 与其他组件的关系

现在我们来理解 Structured Streaming 在 Spark 生态系统中的位置。

Spark Streaming处理的是源源不断的数据流，但有个重要区别：

数据抽象 (Data Abstraction) 不同：

- **Spark Streaming** 采用 **DStream** (本质上就是一系列RDD)
- **Structured Streaming** 采用 **DataFrame**

这个区别很重要！DataFrame 提供了更高级的抽象和更好的性能优化。

与 **Spark SQL 的关系：** Structured Streaming 可以使用 Spark SQL 的 DataFrame/Dataset 来处理数据流。虽然 Spark SQL 也是采用 DataFrame 作为数据抽象，但是：

- **Spark SQL** 只能处理静**态的数据**
- **Structured Streaming** 可以处理**结构化的数据流**

所以说，Structured Streaming 就将 **Spark SQL** 和 **Spark Streaming** 二者的特性结合了起来 —— 既有 SQL 的易用性，又有流处理的实时性。

SL-09 슬라이드 9: 功能和性能对比

继续我们的对比分析。

功能兼容性： Structured Streaming 可以对 DataFrame/Dataset 应用前面章节提到的各种操作：

- **select** - 选择列 xuǎn zé liè
- **where** - 条件过滤 tiáo jiàn guò lǜ
- **groupBy** - 分组聚合 **fēn zǔ jù hé**
- **map** - 映射转换 **yìng shè zhuǎn huàn**
- **filter** - 数据过滤
- **flatMap** - 扁平化映射 biǎn píng huà yìng shè

这意味着你们学过的 DataFrame 操作都可以直接用在流处理上！

性能对比： 这里有个重要的性能提升：

- **Spark Streaming** 只能实现**秒级**的实时响应(xiǎng yìng)
- **Structured Streaming** 采用了全新的设计方式：
    - 微批处理模型：可以实现**100毫秒级别**的实时响应
    - 持续处理模型：可以支持**毫秒级**的实时响应

这是一个巨大的性能跃升(yuè sheng)！从秒级到毫秒级，意味着更多实时应用成为可能。

이 다음부터 오는 내용은 적절하게 슬라이드를 보면서 설명할 것