Slide 1 ~ Slide 5

Slide 6: Flink 아키텍처 다이어그램

이제 Flink의 전체 아키텍처를 살펴보겠습니다!

이 다이어그램은 말 그대로 Flink의 "신분증"입니다.

여러분, Flink의 아키텍처는 세 가지 주요 계층으로 나뉩니다:

APIs & Libraries 계층 (최상층):

- **왼쪽**: DataStream API - 스트림 데이터 전문 처리
- **오른쪽**: DataSet API - 배치 데이터 처리

이것은 마치 이중 언어 통역사와 같아서, "연속적인 대화"(스트림 처리)와 "책 한 권"(배치 처리) 모두를 처리할 수 있습니다.

Core 계층 (중간층):

이것은 Flink의 "두뇌"입니다 - **분산 스트림 데이터 플로우 런타임**. 마치 슈퍼 지휘관처럼 모든 데이터 처리 작업을 조정합니다.

Deploy 계층 (하단층):

Local, Cluster, Cloud 세 가지 배포 방식을 지원합니다. 이것은 Flink의 "세 가지 생활 방식"과 같습니다:

- **Local**: 재택근무
- **Cluster**: 대기업 출근
- **Cloud**: 세계를 여행하는 디지털 노마드

흥미롭게도, Flink는 만능 집사와 같아서, 실시간 주식 데이터 처리(스트림 처리)든 과거 판매 보고서 분석(배치 처리)이든 모두 능숙하게 처리할 수 있습니다!

Slide 1 ~ Slide 5

Slide 6:Flink架构图

现在我们来看Flink的整体架构！

这张图简直就是Flink的"身份证"。

大家看，Flink的架构分为三个主要层次**(céngcì)**：

APIs & Libraries层（顶层）：

- 左边是DataStream API，专门处理流数据
- 右边是DataSet API，处理批数据

这就像一个双语翻译官，既能处理"连续不断的对话"（流处理），也能处理"整本书"（批处理）。

Core层（中间）：这是Flink的"大脑"——分布式流数据流运行时。就像一个超级指挥官，协调所有的数据处理任务。

- **Streaming Dataflow Runtime**

Deploy层（底层）：

支持Local、Cluster、Cloud三种部署(bù shǔ)方式。这就像是Flink的"三种生活方式"：

- Local：像在家里工作
- Cluster：像在大公司上班
- Cloud：像是环游世界的数字游民(yóu mín)

有趣的是，Flink就像一个全能的管家，无论你是要处理实时股票数据（流处理），还是分析历史销售xiāoshòu报告（批处理），它都能胜任！

Slide 7: Flink의 탄생지

이 따뜻한 일러스트는 Flink의 기원 이야기를 보여줍니다!
이 그림을 보세요: 베를린 공과대학을 나타내는 건물이 오른쪽의 개발팀과 연결되어 있습니다. 이것은 마치 기술 창업의 동화 같은 이야기입니다!

2008년, 독일 베를린 공과대학의 어느 연구실에서 몇몇 똑똑한 연구원들이 한 가지 문제에 대해 고민하고 있었습니다: 어떻게 데이터 처리를 더 빠르고 스마트하게 만들 수 있을까?

《해리 포터》의 마법사들처럼, 그들은 코드의 "마법 지팡이"를 휘둘러 Flink라는 놀라운 도구를 만들어냈습니다.

아시나요? "Flink"라는 이름은 독일어로 "유연하고 민첩하다"는 의미입니다.
마치 민첩한 치타처럼 빠르게 흐르는 데이터를 빠르게 포착하고 처리할 수 있습니다.

베를린 공과대학은 우리에게 Flink뿐만 아니라 중요한 교훈도 주었습니다: 위대한 기술은 종종 학계의 상아탑에서 탄생하여 광활한 산업 세계로 날아간다는 것입니다.

Slide 7:Flink的诞生地

这张温馨的插图(chā tú)展示了Flink的起源故事!
看这幅图：一座代表柏林(**Bólín**)理工大学的建筑，连接着右边的开发团队。
这就像是一个科技创业的童话(tóng huà)故事!

2008年，在德国柏林理工大学的某个实验室里，几位聪明的研究人员正在思考一个问题：如何让数据处理变得更快、更智能？

就像《哈利·波特》中的魔法师一样，他们挥舞(huīwǔ)着代码的"魔法棒(mófǎbàng)"，创造出了Flink这个神奇的工具。

你们知道吗？
"Flink"这个名字在德语中的意思是"灵活、敏捷(mǐnjié)"。
就像一只敏捷的猎豹(lièbào，能够快速捕捉(bǔzhuō)并处理飞速流动的数据。

柏林理工大学不仅给了我们Flink，也给了我们一个重要的启示(qǐshì)：
伟大的技术往往诞生于学术的象牙塔(xiàngyátǎ)，
然后飞向广阔(guǎngkuò)的工业世界。

Slide 8: Flink 탄생 연도

2008년! 이 눈에 띄는 달력 아이콘을 보세요.
2008년은 기술 세계에 있어 특별한 해였습니다.
그 해에 글로벌 금융 위기가 발생했지만, 동시에 많은 위대한 기술이 탄생한 해이기도 합니다.
사람들이 여전히 경제 위기로 고민하고 있을 때, 독일의 한 연구원 그룹이 조용히 기술의 씨앗을 심고 있었습니다.

흥미롭게도, 2008년은 베이징에서 올림픽이 개최된 해이기도 합니다.
올림픽이 인간 체육의 극한을 보여준 것처럼, Flink도 나중에 데이터 처리 속도의 극한을 보여주었습니다.

당시 iPhone은 출시된 지 겨우 1년밖에 되지 않았고, Twitter는 여전히 소수의 소셜 플랫폼이었으며, 10년 후에 이렇게 방대한 실시간 데이터를 처리해야 할 줄은 아무도 예상하지 못했습니다.
하지만 Flink의 창시자들은 선견지명이 있었습니다.

그들은 기술 세계의 예언자처럼 미래의 데이터 홍수를 처리할 도구를 미리 준비해 두었습니다.
2008년의 연구 프로젝트에서 시작해 지금 Apache의 최상위 프로젝트가 되기까지, Flink는 10년 이상의 시간을 들여 한 가지 진리를 증명했습니다: **좋은 기술은 좋은 와인과 같아서 숙성할 시간이 필요하다는 것입니다.**

Slide 8:Flink诞生年份

2008年！看这个醒目(xǐng mù)的日历图标。
2008年对于科技世界来说是一个特殊(tèshū)的年份(niánfèn)。
那一年，全球金融危机爆发(bào fā)，但也是许多伟大技术诞生的年份。
就在人们还在为经济危机苦恼(kǔ nǎo)的时候，一群德国的研究人员正在悄悄(qiāo qiāo)地种下一颗技术的种子。

有趣的是，2008年也是奥运会在北京举办的年份。
就像奥运会展示了人类体育的极限(jí xiàn)一样，Flink后来也展示了数据处理速度的极限。

当时iPhone才刚刚发布一年，Twitter还是一个小众社交平台，没人能预料((yùliào))到十年后会有如此海量的实时数据需要处理。
但Flink的创造者们却有着(yǒu zhe)超前(chāoqián)的眼光(yǎnguāng)，

他们就像科技世界的预言家，提前为我们准备好了处理未来数据洪流的工具。

从2008年的一个研究项目，
到现在成为Apache的顶级项目，
Flink用了十多年时间证明了一个道理：
好的技术就像好酒，需要时间来酝酿(**yùn niàng**)。

Slide 9: Spark 탄생 연도

2009년! Spark가 화려하게 등장했습니다! 이 달력을 보세요. Spark는 Flink보다 1년 늦게 탄생했습니다.

이것은 마치 연이은 해에 태어난 두 형제와 같아서, 비슷한 사명을 가지고 있지만 서로 다른 성격을 가지고 있습니다.
2009년은 "후금융 위기" 시대로, 전 세계가 새로운 성장 동력을 찾고 있었습니다.
캘리포니아 대학 버클리의 연구원들도 생각하고 있었습니다: 어떻게 데이터 처리를 더 빠르게 만들 수 있을까?
그들의 답은 바로 Spark였습니다 - 번개처럼 빠른 빅데이터 처리 엔진.

흥미로운 우연의 일치: 2009년은 비트코인이 탄생한 해이기도 합니다.

이 해는 "데이터 혁명"의 중요한 출발점이었다고 할 수 있습니다.
Spark와 비트코인, 하나는 전통적인 데이터를 처리하고 하나는 새로운 디지털 화폐를 만들어 각자의 분야에서 큰 파장을 일으켰습니다.
Spark의 이름은 흥미롭습니다 - "불꽃(spark)". 작은 불꽃 하나가 초원 전체를 태울 수 있듯이, Spark도 정말로 빅데이터 처리 분야 전체에 열정을 불붙였습니다.
2009년부터 지금까지, Spark는 이미 빅데이터 처리의 벤치마크 중 하나가 되었습니다.

Slide 9: Spark诞生年份

2009年！Spark闪亮登场了！
看这个日历，Spark比Flink晚了一年诞生。

这就像是两个出生在相邻**(xiāng lín)**年份的兄弟，有着相似的使命，却有着不同的性格。

2009年，正是"后金融危机"时代，全世界都在寻找(xúnzhǎo)新的增长点(zēngzhǎng diǎn)。
加州大学伯克利分校的研究人员们也在思考：如何让数据处理变得更快？

他们的答案就是Spark—一个像闪电一样快速的大数据处理引擎**(yǐn qíng)**。

有个有趣的巧合：2009年也是比特币诞生的年份。
可以说，这一年是"数据革命"的重要起点。
Spark和比特币，一个处理传统数据，一个创造新型数字货币，都在各自的领域掀起(xiānqǐ)了巨浪(jùlàng)。

Spark的名字很有意思——"火花"。
正如一个小小的火花可以点燃整(diǎnrán)个草原，Spark也确实点燃了整个大数据处理领域的热情。

从2009年到现在，Spark已经成为了大数据处理的标杆(biāo gān)之一。

Slide 10: Spark의 탄생지

이제 Spark의 "고향"을 보게 되었습니다 - **캘리포니아 대학 버클리!**

이 유명한 교장은 미국 서부 해안 최고의 공립 대학 중 하나를 나타냅니다.

버클리는 기술 혁신의 "실리콘밸리 두뇌"와 같아서, 실리콘밸리에서 불과 수십 킬로미터 떨어져 있으며, 학술적 분위기와 창업 정신이 완벽하게 융합되어 있습니다.

2009년 버클리 캠퍼스를 상상해보세요:
학생들이 배낭을 메고 햇빛에 비춰진 돌길을 걸어가고, 어느 컴퓨터 연구실에서는 몇몇 대학원생들이 MapReduce를 어떻게 더 빠르게 만들 수 있을지 논의하고 있습니다.
그들은 아마도 커피를 마시면서 화이트보드에 아키텍처 다이어그램을 그리다가 누군가가 "데이터를 메모리에 보관하면 어떨까?"라고 말했을 것입니다 - 그렇게 Spark의 핵심 아이디어가 탄생했습니다!
버클리는 애플의 창립자부터 구글의 엔지니어에 이르기까지 수많은 기술 엘리트를 배출했습니다.
Spark의 탄생은 이 학교의 기술 혁신 역사에 굵직한 한 획을 더했습니다.
이것은 마치 기술 버전의 "아메리칸 드림"과 같습니다:
캠퍼스의 한 아이디어에서 시작해 결국 세상을 바꾸는 기술로 성장하는 것입니다.

Slide 10:Spark的诞生地

现在我们看到了Spark的"老家"——加州大学伯克利分校！

这个著名的校徽代表着美国西海岸最顶尖(dǐng jiān)的公立大学之一。

伯克利分校就像是技术创新的"硅谷大脑(Guīgǔ Dànǎo)"，距离硅谷只有几十公里，学术氛围和创业精神在这里完美融合。

想象一下2009年的伯克利校园：学生们背着书包走过被阳光照耀(**zhàoyào**)的石径，在某个计算机实验室里，几个研究生正在讨论如何让MapReduce变得更快。
他们可能一边喝着咖啡，一边在白板上画着架构图，突然有人说："为什么不把数据保存在内存里呢？"—就这样，Spark的核心思想诞生了！

伯克利分校培养出了无数的科技精英，从苹果公司的创始人到谷歌的工程师。

Spark的诞生更是为这所学校的科技创新历史添上了(**tiān shàng**) 浓墨重彩(nóng mò zhòng cǎi)的一笔 (bǐ) 。
这就像是一个科技版的"美国梦"：
从校园里的一个想法，
最终成长为改变世界的技术。

Slide 11: Flink 생태계

와! 이 그림은 Flink의 풍부한 생태계를 보여줍니다!
이 아키텍처 다이어그램을 보세요. 마치 하이테크 레고 블록 시스템과 같습니다! 각 컴포넌트마다 고유한 역할이 있고, 결합하면 강력한 데이터 처리 애플리케이션을 구축할 수 있습니다.

Programming APIs (프로그래밍 인터페이스):
Java, Scala를 지원하며, 심지어 Hadoop MapReduce도 지원합니다! 이것은 마치 "언어 유엔"과 같아서, 어떤 프로그래밍 언어를 사용하든 Flink가 이해할 수 있습니다.

Data Access (데이터 액세스):
왼쪽의 빨간 원으로 표시된 컴포넌트들이 정말 흥미롭습니다 - 전통적인 파일 시스템부터 현대의 Kafka까지, 마치 데이터의 "만능 어댑터"처럼 어떤 형식의 데이터든 받아들일 수 있습니다.

Deployment (배포):
오른쪽 빨간 원의 Hadoop YARN은 특히 주목할 만합니다. 이것은 Flink에게 강력한 "집주인"을 찾아준 것과 같아서, Hadoop 생태계에서 안심하고 살 수 있게 합니다.

흥미로운 비유: 전통적인 데이터 처리 시스템이 구식 필름 카메라라면, Flink 생태계는 현대의 디지털 일안 반사식 DSLR카메라와 같습니다 - 기능이 강력할 뿐만 아니라 선택할 수 있는 다양한 렌즈와 액세서리가 있어 전문가급 "데이터 사진"을 찍을 수 있습니다.

Slide 11:Flink生态系统

哇！这张图展示了Flink丰富的生态系统！

看这个架构图，简直就像一个高科技的乐高积木(Lègāo Jīmù)系统！
每个组件都有自己的作用，
组合起来就能构建强大的数据处理应用。

Programming APIs（编程接口）：
支持Java、Scala，甚至还有Hadoop MapReduce！
这就像是一个"语言联合国"，无论你说哪种编程语言，Flink都能听懂。

Data Access（数据访问）： 左边那些被红圈(hóng quān)标注的组件真有趣——从传统的文件系统到现代的Kafka，就像是一个数据的"万能插座(chā zuò)"，什么格式的数据都能接。

Deployment（部署）： 右边红圈里的Hadoop YARN特别值得注意。
这就像是给Flink找了一个强大的"房东(fáng dōng)"，让它能在Hadoop的生态系统中安心居住。

有个有趣的比喻：如果说传统的数据处理系统像是老式的胶片(jiāo piàn)相机，那么Flink生态系统就像是现代的数码单反(shùmǎ dānfǎn)—不仅功能强大，还有各种镜头和配件可以选择，让你能够拍出专业级的"数据照片"。

Slide 12: Apache Spark vs Apache Flink 비교표

이 페이지들은 Spark와 Flink의 상세한 비교를 보여줍니다. 마치 두 슈퍼히어로의 "스킬 카드" 대결과 같습니다!

핵심 구현:

- **Spark**: Scala (우아한 함수형 프로그래밍 언어)
- **Flink**: Java (안정적이고 신뢰할 수 있는 엔터프라이즈급 언어)

이것은 마치 "예술가 vs 엔지니어"의 대결입니다!

프로그래밍 인터페이스:

- **Spark**: Java, Python, R 언어 지원 - 진정한 "다중언어 학자"
- **Flink**: DataSet API는 Java, Scala, Python 지원, DataStream API는 Java와 Scala 지원

이것은 마치 두 명의 통역사와 같습니다. Spark는 유엔 통역사에 가깝고, Flink는 전문 기술 통역사에 가깝습니다.

Slide 10:Apache Spark vs Apache Flink对比表

这几页展示了Spark和Flink的详细对比，就像是两个超级英雄的"技能卡片"对决！

核心实现：

- Spark：Scala（优雅的函数式编程语言）
- Flink：Java（稳重可靠的企业级语言）

这就像是"艺术家 vs 工程师"的对决！

编程接口：

- Spark：支持Java、Python、R语言，真是一个"多语言学霸"
- Flink：DataSet API支持→Java、Scala、Python，DataStream API支持→Java和Scala

这就像是两个翻译官，Spark更像是联合国翻译，Flink更像是专业技术翻译

계산 모델 (가장 흥미로운 부분!):

- **Spark**: 데이터 조각 집합 기반의 마이크로 배치 처리 모델
- **Flink**: 연산자 기반의 연속 스트림 모델

상상해보세요:

- Spark는 효율적인 공장 조립 라인과 같아서, 연속적인 데이터를 작은 조각으로 잘라 처리합니다. 마치 긴 국수를 작은 조각으로 자르는 것과 같습니다.
- Flink는 진정한 강물과 같아서, 데이터가 물처럼 끊임없이 흐릅니다.

장단점:

이것은 핵심 "전투력" 비교입니다!

- **Spark**: 스트림 처리에 지연이 있으며, 초 단위 계산만 지원 (강하지만 약간 둔한 역사와 같음)
- **Flink**: 스트림 컴퓨팅은 Storm과 성능이 비슷하며, 밀리초 단위 계산 지원 (민첩한 닌자와 같음)

비유하자면: 실시간 주식 거래 데이터를 처리해야 한다면, Spark는 "1초만 기다려"라고 말할 수 있지만, Flink는 "이미 처리 끝났어"라고 말할 것입니다!

计算模型（这是最精彩的部分！）：

- Spark：基于数据片集合的微批处理模型
- Flink：基于操作符的连续流模型

想象一下：

- Spark就像是一个高效的工厂流水线，把连续的数据切成小块来处理，就像把一条长面条切成小段一样。
- 而Flink就像是一条真正的河流，数据像水一样连续不断地流动。

优缺点(yōu quē diǎn)：

这是关键的"战斗力"对比！

- Spark：流式处理有延时，只支持秒级计算（像是一个强壮但略显笨重的大力士）
- Flink：流式计算与Storm性能相当，支持毫秒级计算（像是一个敏捷的忍者）

打个比方：如果你要处理实时股票交易数据，Spark可能会说"等我一秒钟"，而Flink会说"我已经处理完了"！

Slide 13: 성능 비교 차트

이 차트는 정말 직관적입니다! 마치 데이터 처리의 "올림픽"과 같습니다!

이 성능 차트를 보면, X축은 노드 수(10, 30, 60)를 나타내고, Y축은 처리 능력을 나타냅니다. 세 개의 선이 세 명의 "선수"를 나타냅니다:

- **파란선**: Flink (착실하게 상승, 마라톤 챔피언 같음)
- **빨간선**: Spark (중간 성능, 전천후 운동선수 같음)
- **녹색선**: Hadoop(MR) (급속 상승, 로켓 같음)

흥미롭게도, 노드 수가 증가할 때 Hadoop MapReduce의 성능 향상이 가장 두드러지지만, 이것은 마치 대포로 모기를 잡는 것과 같습니다 - 강력하지만 약간 과도할 수 있습니다.

Flink와 Spark의 성능은 더 안정적이고 실용적이며, 마치 경험이 풍부한 두 운전자처럼 다양한 도로 상황에서 최상의 운전 경험을 유지하는 방법을 알고 있습니다.

이 차트는 우리에게 중요한 진리를 알려줍니다: **절대적인 "최고"는 없고, 가장 적합한 것만 있을 뿐입니다.**

기술을 선택하는 것은 교통수단을 선택하는 것과 같습니다 - 시내 통근에는 전기차, 장거리 여행에는 고속철도, 각각의 장점이 있습니다.

Slide 13:性能对比图表

这张图表太直观了！就像是一场数据处理的"奥运会"！

看这个性能图表，X轴代表节点数量（10、30、60），Y轴代表处理能力。

三条线代表三个"选手"：

- 蓝线：Flink（稳步上升，像马拉松冠军）
- 红线：Spark（中等表现，像全能运动员）
- 绿线：Hadoop(MR)（急速攀升，像火箭一样）

有趣的是，当节点数量增加时，Hadoop MapReduce的性能提升最为显著，但这就像是用大炮打蚊子——虽然威力强大，但可能有点过度了。

Flink和Spark的表现更加稳定和实用，就像是两个经验丰富的老司机，知道如何在不同路况下保持最佳的驾驶体验。

这个图表告诉我们一个重要道理：没有绝对的"最好"，只有最适合的。

选择技术就像选择交通工具——市内通勤(tōng qín)用电动车，长途旅行用高铁，各有各的优势。

Slide 14: 기생유, 하생량!

이 슬라이드는 너무 극적입니다!
《삼국지》에서 주유의 고전적인 대사를 빌려왔습니다.
이 비교를 보세요:

Spark의 장점:

- ✓ 좋은 커뮤니티
- ✓ 기업 지원이 빠름

Flink의 특징:

- ✓ 불온불화(不愠不火, 그다지 뜨겁지도 차갑지도 않음)

이것은 마치 성격이 완전히 다른 두 스타와 같습니다:

- Spark는 인기 아이돌과 같아서 팬이 많고 상업적 가치가 높으며 어디를 가든 환영받습니다
- Flink는 실력파 배우에 가까워서 그다지 유명하지는 않을 수 있지만 전문 능력은 매우 강합니다

"기생유, 하생량(既生瑜, 何生亮)"이라는 말은 사실 기술 세계의 흥미로운 현상을 반영합니다: 종종 기능이 유사한 기술이 동시에 존재합니다.
하지만 이런 "경쟁"은 사실 좋은 일입니다. 안드로이드와 iOS의 경쟁이 있었기에 우리가 더 나은 스마트폰 경험을 즐길 수 있는 것처럼 말입니다.
Spark와 Flink의 관계는 코카콜라와 펩시콜라와 같습니다 - 표면적으로는 경쟁자이지만 실제로는 전체 산업의 발전을 함께 추진하고 있습니다.
사용자가 최종적으로 이익을 얻습니다. 더 많은 선택권과 더 나은 기술을 갖게 되기 때문입니다.

Slide 14:既生瑜，何生亮！

这张幻灯片太有戏剧性了！
借用了《三国演义》中周瑜的经典台词。

看这个对比：

Spark的优势：

- ✓社区好，
- ✓企业支持早

Flink的特点：

- 不愠不火，

这就像是两个性格完全不同的明星：

- Spark就像是一个人气偶像(ǒuxiàng)，粉丝(fěnsī)众多，商业价值高，到哪里都有人欢迎
- Flink更像是一个实力派演员，可能不那么出名，但专业能力很强

"既生瑜，何生亮"这句话其实反映了技术世界的一个有趣现象：往往会有功能相似的技术同时存在。
但这种"竞争"其实是件好事，就像有了安卓和iOS的竞争，我们才能享受到更好的手机体验。

Spark和Flink的关系就像是可口可乐和百事可乐——表面上是竞争对手，实际上是在共同推动(tuī dòng)整个行业的发展。

用户最终受益(shòu yì)，因为我们有了更多选择，更好的技术。

Slide 15: Google과 Flink

보세요! Google이라는 기술 거인이 우정의 손을 내밀었습니다!
이 이미지는 중요한 전환점을 보여줍니다: Google이 Flink를 지원하기 시작했습니다.
이 녹색 달러 기호는 투자와 지원을 나타내며, 그 손은 Google의 기술력을 나타냅니다.

이것은 매우 흥미로운 발전입니다. Google 자체가 빅데이터 기술의 원조라는 것을 알아야 합니다 - MapReduce, BigTable, GFS 같은 기술들은 모두 Google이 처음 제안한 것입니다.
이제 Google이 Flink를 지원하기로 선택한 것은 마치 무림 맹주가 신흥 문파를 인정하는 것과 같습니다.

이런 장면을 상상해보세요: Google의 엔지니어들이 회의실에 앉아 논의합니다: "우리에게는 이미 자체 데이터 처리 기술이 있는데, 왜 Flink를 지원해야 하지?"
답은 간단합니다: Flink가 스트림 처리 분야에서 정말로 독특한 장점을 가지고 있기 때문이며, Google조차도 이를 인정했습니다!
이것은 마치 최고급 레스토랑의 주방장이 자신만의 요리가 있지만 다른 훌륭한 레시피도 인정하는 것과 같습니다.

Slide 15: Google与Flink

看！Google这个科技巨人伸出了友谊之手！

这张图片展示了一个重要的转折(**zhuǎn zhé**)点：
Google开始支持Flink。
这个绿色的美元符号代表着投资和支持，
而那只手代表着Google的技术实力。

这是一个非常有意思的发展。要知道，Google本身就是大数据技术的鼻祖(bízǔ)——MapReduce、BigTable、GFS这些技术都是Google首先提出的。

现在Google选择支持Flink，就像是武林盟主(**méng zhǔ**)对一个新兴门派的认可。

想象一下这样的场景：Google的工程师们坐在会议室里讨论：
"我们已经有了自己的数据处理技术，为什么还要支持Flink？

"答案很简单：因为Flink在流处理方面确实有独特的优势，连Google都认可了！

这就像是一个顶级餐厅的主厨， 虽然拥有自己的招牌菜，但也会认可其他优秀的食谱(shí pǔ)。

Slide 16: 천하통일?

이 질문은 정말 날카롭습니다! Google이 "천하통일"을 원할까요?

화면은 여전히 Google을 나타내는 손이지만, 이번에는 제목이 "천하통일?"로 바뀌었습니다 - 이 질문은 마치 기술 버전의《왕좌의 게임》과 같습니다!

이것은 역사의 많은 사례를 떠올리게 합니다:

- Microsoft는 한때 운영 체제 세계를 통일하려 했습니다
- Facebook은 전 세계의 소셜 네트워크를 연결하려 했습니다
- 이제 Google이 빅데이터 처리 분야를 통일하려는 걸까요?

하지만 현실은 종종 상상보다 복잡합니다. 자연 생태계처럼 기술 세계도 다양성이 필요합니다:

- 어떤 기술은 실시간 데이터 처리에 적합
- 어떤 기술은 과거 데이터 처리에 적합
- 어떤 기술은 소규모 회사에 적합
- 어떤 기술은 대기업에 적합

Google이 아무리 강력해도 "천하통일"은 현실적이지 않을 수 있고, 꼭 좋은 일도 아닙니다. 다양화된 기술 생태계가 더 건강합니다. 마치 숲에 다양한 식물과 동물이 필요한 것처럼 말입니다.

아마도 Google의 진정한 목표는 "통일"이 아니라 "통합"일 것입니다 - 다양한 기술들이 더 잘 협력할 수 있도록 하는 것입니다.

Slide 16:一统天下？

这个问题太犀利(**xī lì**)了！Google会不会想要"一统天下"？

画面还是那只代表Google的手，但这次标题变成了"一统天下？"——这个问题简直像是科技版的《权力的游戏》！

这让我想起了历史上的很多例子：

- 微软曾经(céngjīng)想要统一操作系统世界
- Facebook想要连接全世界的社交网络
- 现在Google是不是想要统一大数据处理领域？

但现实往往比想象复杂。

就像自然界的生态系统一样，技术世界也需要多样性：

- 有的技术适合处理实时数据
- 有的技术适合处理历史数据
- 有的技术适合小公司
- 有的技术适合大企业

Google虽然强大，但"一统天下"可能并不现实，也不一定是好事。

多样化的技术生态更健康，就像一个森林需要各种不同的植物和动物一样。

也许Google的真正目标不是"统一"，而是"整合"——让不同的技术能够更好地协作。

Slide 17: Apache Beam 프로그래밍 인터페이스

와! 이 그림은 Apache Beam의 "진화사"를 보여줍니다!
"The Evolution of Apache Beam"의 시연 그림을 보세요. 마치 기술 발전의 서사시 영화를 보는 것 같습니다!
왼쪽에서 오른쪽으로, 완전한 기술 생태계 진화를 볼 수 있습니다:

- **가장 왼쪽**: 전통적인 MapReduce (그 고전적인 빨간 육각형)
- **중간**: 다양한 Google 내부 기술 (빨간 로고)
- **그 다음**: Google Cloud Dataflow
- **가장 오른쪽**: Apache Beam의 로고

아래 흰색 영역은 지원되는 다양한 실행 엔진을 보여줍니다: Hadoop, Spark, Flink 등등.
빨간 원으로 표시된 "프로그래밍 인터페이스" 개념이 특히 중요합니다.
이것은 마치 "번역 다리"를 구축한 것과 같습니다 - 한 가지 언어로 코드를 작성하지만 여러 다른 "기계"에서 실행할 수 있습니다.
상상해보세요, 이것은 마치 책을 한 번 쓰면 Kindle, iPad, 종이책 등 다양한 매체에서 읽을 수 있는 것과 같습니다.

Apache Beam의 이념은 매우 간단합니다: **"Write once, run anywhere"** (한 번 작성하면 어디서나 실행)
이것이 프로그래머의 궁극적인 꿈이 아닐까요?

Slide 17:Apache Beam编程接口

哇！这张图展示了Apache Beam的"进化史"！
看这个"The Evolution of Apache Beam"的演示图，就像是看一部科技发展的史诗(**shǐshī**)电影！
从左到右，我们看到了一个完整的技术生态演进：

- 最左边是传统的MapReduce（那个经典的红色六边形）
- 中间展示了各种Google的内部技术（红色标志）
- 然后是Google Cloud Dataflow
- 最右边是Apache Beam的标志

下方那个白色区域展示了支持的各种执行引擎：Hadoop、Spark、Flink等等。

被红圈标注的"编程接口"概念特别重要。
这就像是搭建了一座"翻译桥梁"——你用一种语言写代码，
但可以在多种不同的"机器"上运行。

想象一下，这就像是写了一本书，可以在Kindle、iPad、纸质书等各种媒介(**méi jiè**)上阅读(**yuè dú**)。

Apache Beam的理念很简单："Write once, run anywhere"（一次编写，到处运行）。
这不就是程序员的终极梦想吗？

Slide 19: Beam 모델 아키텍처

이 그림은 Beam의 기술 아키텍처를 보여줍니다. 말 그대로 완벽한 "기술 샌드위치"입니다!

이 계층 구조를 보세요:

최상층: Beam Java, Other Languages, Beam Python

이것은 세 개의 서로 다른 "접수원"과 같아서, 어떤 프로그래밍 언어를 사용하든 전문 서비스 직원이 응대합니다.

중간층: Beam Model: Pipeline Construction

이것은 Beam의 "두뇌"로, 당신의 아이디어를 기계가 이해할 수 있는 "설계도"로 번역하는 역할을 합니다.

하층: Runner A, Cloud Dataflow, Runner B

이것들은 진짜 "시공팀"으로, 설계도를 현실로 만드는 역할을 합니다.

최하층: 세 개의 Execution (실행)

이것은 최종 "건설 노동자"로, 실제 작업을 수행합니다.

이 아키텍처 설계는 특히 교묘합니다: 마치 번역 회사와 같아서, 고객은 어떤 언어로든 요구사항을 제출할 수 있고, 회사 내부에는 표준 처리 프로세스가 있으며, 그런 다음 다양한 실행 팀을 선택하여 작업을 완료할 수 있습니다.

상상해보세요: Python으로 데이터 처리 프로그램을 작성하면, 그것이 Spark에서 실행될 수도 있고, Flink에서 실행될 수도 있으며, 심지어 Google Cloud에서도 실행될 수 있습니다 –

이것이 바로 Beam의 매력입니다!

Slide 19: Beam模型架构

这张图展示了Beam的技术架构，简直就是一个完美的"技术三明治"！

看这个分层结构：

顶层：Beam Java、Other Languages、Beam Python

这就像是三个不同的"接待员"，无论你说哪种编程语言，都有专门的服务人员接待你。

中间层:Beam Model:Pipelineb construction

这是Beam的"大脑"，负责把你的想法翻译成机器能理解的"施工图纸"。

下层:Runner A,Cloud Dataflow、Runner B

这些是真正的"施工队"，负责把图纸变成现实。

底层：三个Execution（执行）

这是最终的"建筑工人"，干实际的活儿。

这个架构设计特别巧妙：就像一个翻译公司，客户可以用任何语言提出需求，公司内部有标准的处理流程，然后可以选择不同的执行团队来完成工作。

想象一下：你用Python写了一个数据处理程序，

它可以在Spark上运行，也可以在Flink上运行，甚至可以在Google Cloud上运行—

这就是Beam的魅力(mèi lì)所在！

Slide 20: 과녁 중앙의 Spark

이 그림은 정말 흥미롭습니다! Spark가 과녁의 정중앙에 놓여 있습니다!

이 디자인을 보세요: 검은색 원형 과녁, 녹색 십자 조준선, 노란색 Spark 로고가 정확히 중심에 위치합니다. 이것은 말 그대로 "기술 사격" 게임입니다!
이 이미지는 강한 메시지를 전달합니다: 빅데이터 처리 전장에서 많은 "궁수들"이 Spark를 목표로 삼고 있다는 것입니다.
하지만 여기서 "사격"은 Spark를 제거하려는 것이 아니라 그것과 경쟁하고 초월하려는 것입니다.

다른 각도에서 보면, 이것은 Spark의 중요한 지위를 설명합니다 - 충분히 중요한 기술만이 모두의 관심의 초점이 될 수 있습니다.
마치 농구장의 스타 플레이어가 항상 중점적으로 방어받는 것처럼 말입니다.

흥미롭게도, 이런 "조준당하는" 상태가 오히려 Spark의 성공을 증명합니다.
기술 세계에서 모방당하는 것이 최고의 찬사이고, 도전받는 것이 최대의 인정입니다.

이것은 마치 무협 소설의 줄거리와 같습니다: 일단 무림제일이 되면 수많은 도전자들이 찾아와 무술을 겨루게 됩니다.
Spark는 지금 바로 그런 "무림맹주"입니다.

Slide 20:Spark在靶心中

这张图太有意思了！Spark被放在了靶心的正中央！

看这个设计：一个黑色的圆形靶子，绿色的十字瞄准线，黄色的Spark标志正好位于中心。这简直就是一个"技术射击"的游戏！

这个图像传达了一个强烈的信息：在大数据处理的战场上，很多"弓箭手"都把Spark当作了目标。
但这里的"射击"不是要消灭Spark，而是要与它竞争，超越它。

从另一个角度看，这也说明了Spark的重要地位——只有足够重要的技术才会成为大家关注的焦点。

就像篮球场上的明星球员总是被重点防守一样。

有趣的是，这种"被瞄准"的状态反而证明了Spark的成功。在技术世界里，被模仿是最高的赞美，被挑战是最大的认可。

这就像是武侠小说中的情节：一旦你成为了武林第一，就会有无数的挑战者前来切磋。

Spark现在就是这样的"武林盟主"。

Slide 21: ✅ 정답: B

Slide 22: ✅ 정답: C

Slide 23: ✅ 정답: B

Slide 24: ✅ 정답: C

Slide 25: ✅ 정답: C

Slide 26: ✅ 정답: C

Slide 27: ✅ 정답: D

Slide 28: ✅ 정답: C

Slide 29: ✅ 정답: C

제9문

Apache Beam 아키텍처의 "Pipeline Construction" 계층은 주로 무엇을 담당하나요?

- A. 구체적인 데이터 처리 작업 실행
- B. 프로그래밍 인터페이스 제공
- C. 사용자 코드를 실행 계획으로 변환
- D. 클러스터 리소스 관리

Slide 30: ✅ 정답: C

Slide 21:✅ 正确答案：B

Slide 22:✅ 正确答案：C

Slide 23:✅ 正确答案：B

Slide 24:✅ 正确答案：C

Slide 25:✅ 正确答案：C

Slide 26:✅ 正确答案：C

Slide 27:✅ 正确答案：D

Slide 28: ✅ 正确答案：C

Slide 29: ✅ 正确答案：C

第9题

Apache Beam架构中的"Pipeline Construction"层主要负责什么？

A. 执行具体的数据处理任务

B. 提供编程接口

C. 将用户代码转换为执行计划

D. 管理集群资源

Slide 30: ✅ 正确答案：C

제 11 문 강의 내용에 따르면, 한 회사가 배치 처리와 스트림 처리를 모두 지원하는 빅데이터 플랫폼을 구축하면서 벤더 락인(vendor lock-in)을 피하고자 할 때, 가장 합리적인 기술 선택은 무엇입니까?

A. 커뮤니티 지원이 좋은 Apache Spark 만 사용한다
B. Google 이 지원하는 Apache Flink 만 사용한다
C. Apache Beam 을 통합 프로그래밍 인터페이스로 사용하여 Spark, Flink 등 여러 엔진에서 실행할 수 있도록 한다
D. Spark 와 Flink 두 개의 독립적인 시스템을 동시에 배포한다

정답: C *
답안 해설:
Apache Beam 의 핵심 설계 철학은 "한 번 작성하고 어디서나 실행"(write once, run anywhere)입니다. 강의 16 페이지의 아키텍처 다이어그램에 따르면, Beam 은 통합된 Pipeline Construction 레이어를 제공하며, 서로 다른 Runner(Spark Runner, Flink Runner, Cloud Dataflow 등)를 통해 다양한 실행 엔진에서 실행될 수 있습니다. 이러한 설계는 벤더 락인을 방지하면서 배치 처리와 스트림 처리를 모두 지원합니다. 선택지 A 와 B 는 모두 기술 스택의 단일 의존성을 야기하고, 선택지 D 는 시스템 복잡도와 유지보수 비용을 증가시킵니다. Beam 의 추상화 레이어 설계는 바로 이 문제를 해결하기 위한 것으로, 기업이 필요에 따라 유연하게 하위 실행 엔진을 전환할 수 있도록 합니다.

第 11 题
根据课程内容，如果一家公司需要构建一个既支持批处理又支持流处理的大数据平台，并且希望避免 vendor lock-in（供应商锁定），最合理的技术选择是什么？

A. 只使用 Apache Spark，因为它社区支持好
B. 只使用 Apache Flink，因为 Google 支持它
C. 使用 Apache Beam 作为统一编程接口，可以在 Spark、Flink 等多个引擎上运行
D. 同时部署 Spark 和 Flink 两套独立系统

正确答案：C
答案解析：
Apache Beam 的核心设计理念就是"一次编写，到处运行"（write once, run anywhere）。根据课程第 16 页的架构图，Beam 提供了统一的 Pipeline Construction 层，可以通过不同的 Runner（如 Spark Runner、Flink Runner、Cloud Dataflow）在不同的执行引擎上运行。这种设计避免了供应商锁定，同时支持批处理和流处理。选项 A 和 B 都会造成技术栈的单一依赖，选项 D 则会增加系统复杂度和维护成本。Beam 的抽象层设计正是为了解决这个问题，让企业可以根据需求灵活切换底层执行引擎。

제 12 문 Flink(2008 년)와 Spark(2009 년)의 발전 과정을 볼 때, Spark 의 시장 수용도가 한때 Flink 를 앞섰지만 Google 이 나중에 Flink 를 지원하기로 선택한 이유는 무엇이며, 이것은 어떤 기술 트렌드를 보여줍니까?

A. Google 의 선택은 무작위적이며 특별한 의미가 없다
B. 초기에는 시장 성숙도와 커뮤니티 생태계가 기술적 우위보다 중요하지만, 실시간 스트림 처리 수요가 증가하면서 기술 아키텍처의 선진성이 더 중요해진다
C. Flink 가 Spark 보다 1 년 늦었기 때문에 기술이 반드시 더 좋다
D. 단지 Flink 가 유럽에서 개발되었고 Google 이 국제화를 지원하고 싶어서이다

정답: B
답안 해설:
이 문제는 기술 진화 법칙에 대한 이해를 평가합니다. 강의 12 페이지의 "기생유, 하생량(既生瑜, 何生亮)" 비교에 따르면, Spark 의 장점은 "좋은 커뮤니티"와 "조기 기업 지원"이며, Flink 는 "미온적"으로 묘사됩니다. 그러나 10 페이지의 비교표를 보면 Flink 는 스트림 처리에서 본질적인 우위를 가지고 있습니다: 연산자 기반의 연속 스트림 모델로 밀리초 단위 컴퓨팅을 지원하는 반면, Spark 는 마이크로 배치 처리 모델을 채택하여 초 단위 컴퓨팅만 지원합니다.
사물인터넷, 실시간 추천 시스템 등 저지연이 요구되는 애플리케이션 시나리오가 증가하면서 Flink 의 기술 아키텍처 우위가 두드러지기 시작했습니다. Google 과 같은 기술 거인이 Flink 를 지원하기로 선택한 것(13-14 페이지)은 기술이 성숙된 후에는 아키텍처의 선진성이 초기 생태계 우위를 넘어선다는 것을 보여줍니다. 이는 또한 Google 이 Beam 을 개발한 이유를 설명합니다 - 기존 생태계를 활용하면서도 기술적으로 선진적인 엔진을 지원하기 위해서입니다.

第 12 题
从 Flink（2008 年）和 Spark（2009 年）的发展历程来看，为什么 Spark 在市场上的接受度一度超过 Flink，但 Google 后来却选择支持 Flink？这说明了什么技术趋势？

A. Google 的选择是随机的，没有特别意义
B. 早期市场成熟度和社区生态比技术优势更重要，但随着实时流处理需求增长，技术架构的先进性变得更关键
C. Flink 比 Spark 晚一年，所以技术一定更好
D. 只是因为 Flink 是欧洲开发的，Google 想支持国际化

正确答案：B

答案解析：
这道题考查对技术演进规律的理解。根据课程第 12 页的"既生瑜，何生亮"对比，Spark 的优势在于"社区好"和"企业支持早"，而 Flink 被描述为"不愠不火"。但从第 10 页的对比表可以看出，Flink 在流处理方面有本质优势：基于操作符的连续流模型，支持毫秒级计算，而 Spark 采用微批处理模型，只支持秒级计算。随着物联网、实时推荐系统等对低延迟要求越来越高的应用场景增多，Flink 的技术架构优势开始凸显。Google 作为技术巨头选择支持 Flink（第 13-14 页），说明在技术成熟后，架构的先进性会超越早期的生态优势。这也解释了为什么 Google 要开发 Beam——既要利用现有生态，又要支持技术先进的引擎。

제 13 문 11 페이지의 성능 비교 그래프를 보면, Flink 와 Spark 는 서로 다른 노드 수에서 성능 곡선이 거의 겹치지만, Hadoop MapReduce 의 증가 곡선은 명확히 더 가파릅니다. 이 현상에 대한 가장 합리적인 기술적 설명은 무엇입니까?

A. Flink 와 Spark 가 동일한 코드베이스를 사용한다
B. Flink 와 Spark 는 모두 인메모리 컴퓨팅 모델을 채택하여 유사한 확장성 특성을 가지며, MapReduce 는 디스크 I/O 가 병목이 되어 확장성이 떨어진다
C. 그래프 데이터에 오류가 있으며, Flink 의 성능이 가장 좋아야 한다
D. 노드 수는 성능에 영향을 미치지 않는다

****정답: B****

****답안 해설:****

이 문제는 기저 기술 아키텍처에 대한 이해를 평가합니다. 비록 강의에서 명확히 설명하지는 않았지만, 그래프로부터 추론할 수 있습니다: Flink 와 Spark 는 모두 차세대 빅데이터 프레임워크로서, 둘 다 인메모리 컴퓨팅 모델을 채택했기 때문에 확장성 측면에서 유사한 성능을 보입니다. 두 프레임워크의 성능 곡선은 거의 겹쳐져 있으며(둘 다 60 노드에서 약 1000-1500 수준의 성능), 이는 배치 처리 시나리오에서 확장 특성이 비슷하다는 것을 보여줍니다. 반면 Hadoop MapReduce 는 1 세대 빅데이터 프레임워크로서 중간 결과 저장을 위해 디스크 I/O 에 의존합니다. 노드 수가 증가할 때 컴퓨팅 능력은 향상되지만, 디스크 I/O 가 주요 병목이 되어 성능 증가 곡선이 더 가파르게 나타납니다(60 노드에서 약 7000 에 달하며, 이는 Flink 와 Spark 보다 훨씬 높습니다). 이러한 대비는 기술 세대 간 차이를 부각시킵니다: 2 세대 프레임워크(Flink/Spark)는 인메모리 컴퓨팅을 통해 I/O 병목을 최적화했지만, 높은 확장성 시나리오에서는 유사한 성능을 보이며, 이들 간의 진정한 차이는 배치 처리가 아닌 스트림 처리(10 페이지 비교표)에서 나타납니다.

第 13 题

观察第 11 页的性能对比图，Flink 和 Spark 在不同节点数量下的性能曲线几乎重叠，但 Hadoop MapReduce 的增长曲线明显更陡。这个现象最合理的技术解释是什么？

A. Flink 和 Spark 都使用了相同的代码库
B. Flink 和 Spark 都采用内存计算模型，具有相似的扩展性特征；而 MapReduce 的磁盘 I/O 成为瓶颈，导致扩展性较差
C. 图表数据有误，应该 Flink 性能最好
D. 节点数量对性能没有影响

正确答案：B

答案解析：

这道题考查对底层技术架构的理解。虽然课程没有明确说明，但从图表可以推断：Flink 和 Spark 都是新一代大数据框架，都采用了内存计算模型，因此在扩展性上表现相似。两者性能曲线几乎重叠（都在 60 节点时约 1000-1500 的性能水平），说明它们在批处理场景下的扩展特性相近。而 Hadoop MapReduce 作为第一代大数据框架，依赖磁盘 I/O 进行中间结果存储，当节点数增加时，虽然计算能力提升，但磁盘 I/O 成为主要瓶颈，导致性能增长曲线更陡（60 节点时达到约 7000，远高于 Flink 和 Spark）。这个对比凸显了技术代际差异：第二代框架（Flink/Spark）通过内存计算优化了 I/O 瓶颈，但在高扩展性场景下表现相似，而它们之间的真正差异体现在流处理（第 10 页对比表）而非批处理上。

제 4 문 9 페이지의 Flink 생태계 다이어그램에 따르면, 기존 Hadoop 생태계를 Flink 로 마이그레이션하면서 HDFS 와 Kafka 에 대한 지원을 유지하려면, 어떤 레이어의 호환성에 중점을 두어야 합니까?

A. 프로그래밍 API 의 호환성만 고려하면 된다
B. Data Access 레이어(데이터 소스 연결)와 Deployment 레이어(배포 환경)의 호환성을 동시에 고려해야 한다
C. 핵심 런타임만 교체하면 된다
D. 호환성을 전혀 고려할 필요 없이 직접 교체하면 된다

****정답: B****

****답안 해설:****

이 문제는 시스템 아키텍처와 마이그레이션 전략에 대한 이해를 평가합니다. 9 페이지의 Flink 생태계 다이어그램을 보면, Flink 아키텍처는 여러 레이어로 나뉩니다. Data Access 레이어에서 Flink 는 HDFS 와 Kafka 등의 데이터 소스에 연결해야 하며(다이어그램에 명확히 표시됨), Deployment 레이어에서는 Hadoop YARN 을 리소스 관리자로 지원해야 합니다(다이어그램에도 표시됨). 따라서 마이그레이션 시 다음을 보장해야 합니다: 1) Flink 의 커넥터가 기존 HDFS 와 Kafka 에 올바르게 접근할 수 있어야 하고, 2) Flink 가 기존 Hadoop YARN 클러스터에서 배포되고 실행될 수 있어야 합니다. 프로그래밍 API 만 고려하는 것(선택지 A)은 하위 인프라 통합 문제를 간과하게 되고, 런타임만 교체하는 것(선택지 C)은 기존 데이터 소스 및 배포 환경과의 단절을 초래하며, 선택지 D 의 "직접 교체"는 비현실적입니다. 실제로 Flink 는 설계 시 Hadoop 생태계와의 호환성을 고려했으며, 이것이 바로 기업들이 Flink 를 채택할 수 있는 중요한 이유입니다.

第 14 题

根据第 9 页的 Flink 生态系统图，如果要将现有的 Hadoop 生态系统迁移到 Flink，同时保持对 HDFS 和 Kafka 的支持，需要重点关注哪些层面的兼容性？

A. 只需要关注编程 API 的兼容性
B. 需要同时考虑 Data Access 层（数据源连接）和 Deployment 层（部署环境）的兼容性
C. 只需要更换核心运行时
D. 完全不需要考虑兼容性，直接替换即可

正确答案：B

答案解析：

这道题考查对系统架构和迁移策略的理解。从第 9 页的 Flink 生态系统图可以看出，Flink 架构分为多个层次。在 Data Access 层，Flink 需要连接到 HDFS 和 Kafka 等数据源（图中明确标注了这些组件）；在 Deployment 层，需要支持 Hadoop YARN 作为资源管理器（图中也有标注）。因此，迁移时必须确保：1）Flink 的连接器能够正确访问现有的 HDFS 和 Kafka；2）Flink 能够在现有的 Hadoop YARN 集群上部署运行。仅仅关注编程 API（选项 A）会忽略底层基础设施的集成问题；只更换运行时（选项 C）会导致与现有数据源和部署环境的脱节；而选项 D 的"直接替换"是不现实的。实际上，Flink 设计时就考虑了与 Hadoop 生态的兼容性，这也是它能够被企业采用的重要原因。

제 5 문 강의 내용을 종합적으로 고려할 때, 다음 중 Flink 나 Spark 를 직접 사용하는 것보다 "Beam + Flink Runner" 조합을 사용하기에 가장 적합한 시나리오는 무엇입니까?

A. 단순한 배치 처리만 필요한 오프라인 데이터 분석 프로젝트

B. 여러 클라우드 플랫폼(Google Cloud, AWS 등)에 배포해야 하고 지연시간 요구사항이 매우 높은 실시간 추천 시스템

C. 로컬 개발 환경에서만 실행되는 프로토타입 프로젝트

D. 스트림 처리가 전혀 필요 없는 데이터 웨어하우스 프로젝트

****정답: B****

****답안 해설:****

이 문제는 기술 선택에 대한 종합적 판단 능력을 평가합니다. "Beam + Flink Runner" 조합의 장점은 다음과 같습니다: 1) Beam 은 통합 프로그래밍 인터페이스를 제공하여 크로스 플랫폼 배포를 지원합니다(16 페이지 아키텍처 다이어그램에서 Google Cloud Dataflow, Spark, Flink 등 여러 Runner 에서 실행 가능함을 보여줌). 2) Flink 는 Runner 로서 밀리초 단위의 저지연 스트림 처리 능력을 제공합니다(10 페이지 비교표). 따라서 여러 클라우드 플랫폼에 배포해야 하고 지연시간 요구사항이 높은 시나리오(예: 실시간 추천 시스템)에서 이 조합이 가장 적합합니다: Beam 은 코드 이식성을 보장하고, Flink 는 스트림 처리 성능을 보장합니다. 선택지 A 의 단순 배치 처리는 이렇게 복잡한 기술 스택이 필요하지 않고, 선택지 C 의 프로토타입 프로젝트는 단일 프레임워크를 사용하는 것이 더 간단하며, 선택지 D 의 데이터 웨어하우스 프로젝트는 스트림 처리 능력이 필요 없어 전통적인 배치 처리 도구를 사용하면 됩니다. 이 문제의 본질은 "언제 기술의 조합 가치가 단일 기술보다 큰가"를 이해하는 것입니다 - 요구사항이 "크로스 플랫폼 이식성"과 "고성능 스트림 처리"를 동시에 포함할 때, Beam + Flink 조합이 의미를 갖습니다.

第 15 题

综合考虑课程内容，以下哪个场景最适合使用"Beam + Flink Runner"的组合，而不是直接使用 Flink 或 Spark？

A. 一个只需要简单批处理的离线数据分析项目

B. 一个需要在多个云平台（Google Cloud、AWS 等）上部署，且对延迟要求极高的实时推荐系统

C. 一个只在本地开发环境运行的原型项目

D. 一个完全不需要流处理的数据仓库项目

正确答案：B

答案解析：

这道题考查对技术选型的综合判断能力。"Beam + Flink Runner"组合的优势在于：1）Beam 提供了统一的编程接口，支持跨平台部署（第 16 页架构图显示可以在 Google Cloud Dataflow、Spark、Flink 等多个 Runner 上执行）；2）Flink 作为 Runner 提供了毫秒级的低延迟流处理能力（第 10 页对比表）。因此，对于需要在多个云平台部署且对延迟要求高的场景（如实时推荐系统），这个组合最合适：Beam 确保代码可移植性，Flink 确保流处理性能。选项 A 的简单批处理不需要这么复杂的技术栈；选项 C 的原型项目使用单一框架更简单；选项 D 的数据仓库项目不需要流处理能力，使用传统批处理工具即可。这道题的本质是理解"什么时候技术的组合价值大于单一技术"——当需

求同时包含"跨平台可移植性"和"高性能流处理"时，Beam + Flink 的组合才有意义。

Slide 28: 강의 마무리

좋습니다, 이번 수업은 여기서 마치겠습니다!

오늘 우리는 "빅데이터 시대"의 전말을 알아보았고, 정보화 물결, 세 가지 지원 기술부터 데이터가 어떻게 생성되고 어떻게 진화하는지까지 배웠습니다.

다음 수업에서는 계속해서 "빅데이터의 핵심 기술"을 탐구할 것입니다.

마지막으로 질문 하나: 오늘 여러분 자신도 매일 "데이터를 생산"하고 있다는 것을 발견하셨나요? 😉

감사합니다 여러분, 수업 끝! 👏👏👏

Slide 28:

好了，我们本节课到这里就结束啦！

今天我们认识了"大数据时代"的来龙去脉，从信息化浪潮、三大支撑技术，到数据如何产生、如何演变。

下节课我们会继续探讨"大数据的关键技术"。

最后提问一下：你今天有没有发现，自己每天其实也在"生产数据"呢？😉

谢谢大家，下课！👏👏👏