任务7-1 内存管理与优化

|  |  |
| --- | --- |
| 任务书 | |
| 一、任务描述与要求 | |
| 任务描述 | 学习计算机内存的基础知识，包括内存的分类、存储器层次结构（如缓存层次、主存等），以及虚拟内存的概念。 |
| 具体要求 | 1.了解计算机内存的层次结构；  2.识别内存泄漏问题。 |
| 二、任务目标 | |
| 知识目标 | * 理解内存管理的基本原理和方法； * 掌握常见的内存优化策略。 |
| 能力目标 | * 能够分析和解决内存相关问题； * 具备对程序进行内存优化的能力。 |
| 三、知识储备 | |
| **1.Spark的内存架构**  Spark的内存架构是其高性能计算的核心。它采用了统一内存管理模型，主要将内存分为执行内存、存储内存、用户内存和预留内存。执行内存用于存储计算过程中的临时数据，如shuffle、join、sort等操作产生的中间结果。存储内存则用于缓存RDD、DataFrame或临时表，通过将频繁访问的数据保存在内存中来提高处理速度。用户内存留给用户自定义的数据结构使用，而预留内存是系统保留的一小部分空间，用于防止OOM错误。    **2.内存管理策略**  Spark的内存管理策略是其性能优化的关键部分。在内存管理中有多种策略，但是最主要的策略只有两个，一个是统一内存管理，一个是动态占用机制。接着分别介绍这两种重要的管理策略：  统一内存管理:Spark 1.6版本之后引入的统一内存管理模型是Spark内存架构的一大革新。这个模型将执行内存和存储内存统一管理，极大地提高了内存利用率。在这个模型中，Spark将可用内存分为三个主要部分：执行内存、存储内存和用户内存。  动态占用机制:动态占用机制是统一内存管理模型的核心特性。它允许执行内存和存储内存之间相互借用空间，这种灵活性大大提高了内存利用率。 | |

|  |
| --- |
| 操作指南 |
| 一、制定计划 |
| |  |  |  |  | | --- | --- | --- | --- | | **序号** | **作业项目** | **序号** | **作业项目** | | 1 |  | 5 |  | | 2 |  | 6 |  | | 3 |  | 7 |  | | 4 |  | 8 |  | | 计划审核 | 审核意见： | | | |
| 二、实施方案 |
| 1. 生成模拟数据  在大数据处理的过程中，内存管理和性能优化是至关重要的。以下案例将帮助你理解如何在Apache Spark中进行内存管理和优化。通过生成模拟数据、进行简单的数据处理，并通过内存管理和性能调优技巧来提高Spark作业的效率。假设有一个模拟数据集，其中包含用户行为记录。使用Spark来处理这些数据，并通过优化内存管理和调优来提升性能安装完成后，首先，使用PySpark生成一个简单的用户行为数据集。具体代码如下：   |  | | --- | | from pyspark.sql import SparkSession  from pyspark.sql.functions import col, when  from pyspark.sql.types import IntegerType, StringType, StructType, StructField  # 创建SparkSession  spark = SparkSession.builder \  .appName("Memory Management and Optimization") \  .config("spark.sql.shuffle.partitions", "4") \  .config("spark.executor.memory", "2g") \  .config("spark.driver.memory", "2g") \  .getOrCreate()  # 生成数据  schema = StructType([  StructField("user\_id", StringType(), False),  StructField("action", StringType(), False),  StructField("timestamp", StringType(), False)  ])  data = [  ("user1", "click", "2024-08-25 10:00:00"),  ("user2", "view", "2024-08-25 10:05:00"),  ("user1", "purchase", "2024-08-25 10:15:00"),  ("user3", "click", "2024-08-25 10:20:00"),  ("user2", "purchase", "2024-08-25 10:25:00"),  ("user3", "view", "2024-08-25 10:30:00")  ]  df = spark.createDataFrame(data, schema=schema)  df.show() |   2. 数据处理  在数据处理阶段，可以对数据进行简单的转换和聚合操作，以模拟实际的数据处理任务。具体代码如下：   |  | | --- | | from pyspark.sql.functions import count  # 计算每个用户的行为次数  user\_action\_counts = df.groupBy("user\_id").agg(count("action").alias("action\_count"))  user\_action\_counts.show() |   3.内存管理与优化  在Spark应用程序中，合理配置内存可以显著提升性能。在创建SparkSession时设置了executor.memory和driver.memory参数。根据数据量和处理复杂度，可以适当调整这些参数。  通过设置shuffle.partitions参数来控制Shuffle过程中数据的分区数量，进而优化性能。在上述代码中，将其设置为4，这通常适用于数据量较小的情况。  4.使用缓存  对于需要多次访问的数据，可以使用cache()或persist()方法将数据缓存到内存中，从而加快后续的访问速度。具体代码如下：   |  | | --- | | # 缓存数据  df.cache()  # 重新计算，以便触发缓存  user\_action\_counts = df.groupBy("user\_id").agg(count("action").alias("action\_count"))  user\_action\_counts.show() |   5.优化广播变量  在处理涉及小表和大表的连接操作时，可以使用广播变量来优化性能。对于小表，可以将其广播到所有节点，从而减少Shuffle操作的开销。具体代码如下：   |  | | --- | | from pyspark.sql.functions import broadcast  # 假设有一个小表作为广播变量  small\_table = spark.createDataFrame([  ("user1", "Group1"),  ("user2", "Group2"),  ("user3", "Group3")  ], ["user\_id", "group"])  # 广播小表  broadcast\_small\_table = broadcast(small\_table)  # 连接操作  joined\_df = df.join(broadcast\_small\_table, on="user\_id", how="inner")  joined\_df.show() | |
| 三、实施记录 |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **序号** | **关键步骤记录** |  | **序号** | **解决问题记录** | | 1 |  |  | 1 |  | | 2 |  |  | 2 |  | | 3 |  |  | 3 |  | | 4 |  |  | 4 |  | | 5 |  |  | 5 |  | | 6 |  |  | 6 |  | | 7 |  |  | 7 |  | | 8 |  |  | 8 |  | | 9 |  |  | 9 |  | |
| 四、任务拓展 |
| 在之前的案例中，使用的是静态模拟数据集。在真实的场景中，需要处理的是实时到来的数据流。可以使用Kafka作为数据源，接收实时数据，并通过Spark Structured Streaming来进行处理。 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 评价标准 | | | | | | | | |
| 一、基本信息 | | | | | | | | |
| 项目名称 | | 单元七 Spark性能调优 | | | | | | |
| 任务名称 | | 任务7-1 内存管理与优化 | | | | | | |
| 班级 |  | | 学号 | |  | 姓名 | |  |
| 组名 |  | | 学时 | |  | 日期 | |  |
| 组员  分工 |  | | |  | | |  | |
|  | | |  | | |  | |
| 二、任务检查评价单 | | | | | | | | |
| |  |  |  | | --- | --- | --- | | **评价内容** | **评价标准** | **得分** | | 知识运用  （20分） | 掌握相关理论知识，理解本次任务要求，制定详细计划，计划条理清晰，逻辑正确（20分） |  | | 理解相关理论知识，能根据本次任务要求、制定合理计划（15分） | | 了解相关理论知识，有制定计划（10分） | | 无制定计划（0分） | | 专业技能  （40分） | 结果验证全部满足。（40分） |  | | 结果验证只有一个功能不能实现，其它功能全部实现（30分） | | 结果验证只有一个功能实现，其它功能全部没有实现（20分） | | 结果验证功能均未实现（0分） | | 核心素养  （20分） | 具有良好的自主学习能力、分析解决问题的能力、整个任务过程中有指导他人（20分） |  | | 具有较好的学习能力和分析解决问题的能力，任务过程中无指导他人（15分） | | 能够主动学习并收集信息，有请教他人进行解决问题的能力（10分） | | 不主动学习（0分） | | 课堂纪律  （20分） | 设备无损坏、设备摆放整齐、工位区域内保持整洁、无干扰课堂秩序（20分） |  | | 设备无损坏、无干扰课堂秩序（15分） | | 无干扰课堂秩序（10分） | | 干扰课堂秩序（0分） | | 得分（满分100） | |  | | 综合评价 | ☆☆☆☆☆ | | | | | | | | | | |