一、Spark 知识点考纲（优秀高校教师视角）

一、Spark 概述（占比 10%）

1.1 Spark 定义与定位

- 1.1.1 Spark 的核心定位：基于内存的快速、通用、可扩展大数据分析计算引擎
- 1.1.2 Spark 与 Hadoop 的差异：计算模型（内存 vs 磁盘）、调度框架（自带 vs YARN 后集成）

1.2 Spark 内置模块

- 1.2.1 核心模块：Spark Core（基础功能、RDD API）、Spark SQL（结构化数据处理）、Spark Streaming（实时流计算）
- 1.2.2 扩展模块：MLlib（机器学习）、GraphX（图计算）

1.3 Spark 核心特点

- 1.3.1 性能优势：内存计算、惰性求值
- 1.3.2 功能优势：易整合、统一数据访问、兼容 Hive、标准数据连接（JDBC/ODBC）

二、Spark 运行模式（占比 10%）

2.1 本地模式（Local）

- 2.1.1 适用场景：开发调试
- 2.1.2 配置与运行：`--master local[K]`参数（K 为核数，* 为自动适配）、官方 PI 案例执行

2.2 YARN 模式（重点）

- 2.2.1 两种部署模式：Client（Driver 在客户端，交互调试）、Cluster（Driver 在 APPMaster，生产环境）

- 2.2.2 环境配置：yarn-site.xml 修改（关闭内存检查）、spark-env.sh 配置 YARN_CONF_DIR
- 2.2.3 历史服务配置：spark-defaults.conf 日志存储路径、spark-env.sh 历史服务端口（18080）

2.3 模式对比与端口

- 2.3.1 模式差异：Local/Standalone/YARN 的机器数、进程、所属框架对比
- 2.3.2 核心端口：4040（任务运行监控）、18080（Spark 历史服务）、8088（YARN 监控）、19888（Hadoop 历史服务）

三、Spark Core（占比 30%）

3.1 RDD 基础

- 3.1.1 RDD 定义与五大特性：弹性、不可变、可分区、并行计算、血缘依赖
- 3.1.2 RDD 创建方式：集合（parallelize）、外部存储（textFile）、其他 RDD 转换

3.2 RDD 分区与算子

- 3.2.1 分区规则：集合分区（手动指定 / 核数适配）、文件分区（totalSize/goalSize/splitSize 计算）
- 3.2.2 转换算子：
  - Value 型：map、flatMap、filter、distinct、sortBy
  - Key-Value 型：mapValues、groupByKey、reduceByKey（预聚合区别）、sortByKey
- 3.2.3 行动算子：collect（慎用）、count、first、take、saveAsTextFile、foreach/foreachPartition

3.3 RDD 核心机制

- 3.3.1 序列化：Java 序列化（通用但重）、Kryo 序列化（性能优，需注册自定义类）
- 3.3.2 依赖关系：窄依赖（一对一 / 多对一，无 Shuffle）、宽依赖（一对多，含 Shuffle）、Stage 划分（宽依赖数 + 1）
- 3.3.3 持久化：Cache（内存缓存，不切断血缘）、Checkpoint（磁盘存储，切断血缘，需配合 Cache 避免重复计算）

3.4 分布式共享变量

- 3.4.1 广播变量：只读变量，减少节点数据传输（创建：`sc.broadcast()`，访问：`.value()`）
- 3.4.2 键值对 RDD 分区器：Hash 分区（默认）、Range 分区（有序场景）

四、Spark SQL（占比 20%）

4.1 Spark SQL 基础

- 4.1.1 定义与优势：结构化数据处理模块，整合 SQL 与 Spark 编程
- 4.1.2 数据结构演进：RDD→DataFrame（带 Schema 的 RDD）→Dataset（强类型，`DataSet<Row>`-即 DataFrame）

4.2 编程入口与方式

- 4.2.1 SparkSession：替代 SQLContext/HiveContext，封装 SparkContext
- 4.2.2 三种编程方式：
  - 方法调用：Dataset API（map、sort、groupByKey）
  - SQL 方式：创建临时视图（createOrReplaceTempView）、执行 SQL 语句
  - DSL 方式：`col()`函数、字段操作（as、plus）、过滤（gt）

4.3 自定义函数与数据交互

- 4.3.1 自定义函数：UDF（一行进一行出）、UDAF（多行进一行出，Spark3.x 推荐 Aggregator）
- 4.3.2 数据加载与保存：
  - 文件格式：JSON（自动识别类型）、CSV（指定分隔符 / 表头）、Parquet（列式存储，默认 Snappy 压缩）
  - 外部系统：MySQL（JDBC 连接，需导入驱动）、Hive（enableHiveSupport，配置 hive-site.xml）

4.4 实战场景：结构化数据分析

- 4.4.1 多表关联：用户行为表 / 城市表 / 产品表 Join
- 4.4.2 窗口函数：`rank() over (partition by ... order by ...)`（如区域热门商品 Top3）

五、Spark Streaming（占比 15%）

5.1 Streaming 基础

- 5.1.1 定义与核心抽象：实时流计算组件，DStream（连续 RDD 序列，批次间独立）
- 5.1.2 架构原理：Receiver（数据接收）、JobGenerator（生成 Job）、JobScheduler（调度执行）

5.2 DStream 操作

- 5.2.1 无状态转换：map、flatMap、filter、reduceByKey（批次内独立）
- 5.2.2 窗口操作：window（窗口时长 / 滑动步长，需为批次整数倍）、reduceByKeyAndWindow
- 5.2.3 输出操作：print（调试）、foreachRDD（通用，如写入 MySQL）、saveAsTextFiles（慎用，小文件多）

5.3 集成与关闭

- 5.3.1 与 Kafka 集成：Direct 模式（Spark3.x 仅支持），配置 Consumer 参数（BOOTSTRAP_SERVERS、GROUP_ID）
- 5.3.2 优雅关闭：外部触发（如 HDFS 文件存在），监控线程 +`stop(true, true)`

六、Spark 内核与调优（占比 10%）

6.1 任务执行与 Shuffle

- 6.1.1 本地化调度：五级级别（PROCESS_LOCAL>NODE_LOCAL>RACK_LOCAL>NO_PREF>ANY）
- 6.1.2 Shuffle 原理：
  - HashShuffle：优化前（每个 Task 对应 Reduce 数文件）、优化后（合并 buffer，`spark.shuffle.consolidateFiles=true`）
  - SortShuffle：排序 + 溢写 + 合并；bypassShuffle（触发条件：Reduce 数≤200 且非聚合算子）

6.2 内存管理

- 6.2.1 内存类型：堆内（JVM 控制，OOM 风险）、堆外（直接申请，减少 GC）
- 6.2.2 内存分配：统一内存管理（存储 / 执行内存动态占用，`spark.memory.fraction=0.6`）

6.3 容错机制

- 6.3.1 Task 失败重试：失败次数≤最大重试次数，黑名单机制（避免重复调度到失败节点）
- 6.3.2 RDD 容错：血缘依赖（重算丢失分区）、Checkpoint（持久化中间结果）

七、实战案例（占比 5%）

7.1 基础案例：WordCount

- 7.1.1 本地调试：Local 模式代码编写、断点调试
- 7.1.2 集群运行：打包部署、HDFS 输入输出路径配置

7.2 综合案例：

- 7.2.1 Top10 热门品类：点击 / 下单 / 支付数统计、排序（自定义 Comparable）
- 7.2.2 区域热门商品 Top3：多表 Join、UDAF 实现城市分布备注、窗口函数排序

二、Spark 知识点类型划分（优秀学生视角）

一、需记忆的知识点（占比 40%）

对应考纲模块	具体知识点
Spark 概述	1. Spark 定义与定位；2. Spark 与 Hadoop 的核心差异；3. Spark 内置模块（Core/SQL/Streaming）及功能；4. Spark 四大特点（易整合、统一数据访问、兼容 Hive、标准连接）

运行模式	1. 各模式适用场景（Local - 调试、YARN - 生产）；2. YARN Client/Cluster 模式差异；3. 核心端口（4040/18080/8088）；4. 模式对比（机器数、进程、所属框架）
Spark Core	1. RDD 五大特性；2. 序列化类型（Java/Kryo）差异；3. 持久化级别（MEMORY_ONLY/MEMORY_AND_DISK 等）及含义；4. 广播变量定义与作用；5. 算子分类（转换 / 行动）及功能描述
Spark SQL	1. DataFrame/Dataset 与 RDD 的差异；2. SparkSession 的作用（替代 SQLContext/HiveContext）；3. UDF/UDAF 定义（UDF 一行进一行出、UDAF 多行进一行出）；4. 数据格式特点（Parquet 列式存储、JSON 自动识别类型）
Spark Streaming	1. DStream 定义（连续 RDD 序列）；2. 窗口操作参数要求（窗口时长 / 滑动步长为批次整数倍）；3. Kafka Direct 模式特点；4. 优雅关闭触发逻辑（外部文件监控）
内核与调优	1. 本地化调度五级级别及优先级；2. Shuffle 类型触发条件（bypassShuffle：Reduce≤200 + 非聚合算子）；3. 统一内存管理参数（`spark.memory.fraction=0.6`）；4. Task 失败重试机制（黑名单作用）

二、需手动推导的知识点（占比 25%）

对应考纲模块	具体知识点
Spark Core	1. 文件 RDD 分区数计算： - 步骤：①计算 totalSize（文件总大小）；②goalSize =totalSize/numSplits；③splitSize=max (minSize, min (goalSize, blockSize))；④按 1.1 倍原则切分 2. Stage 划分： - 逻辑：数宽依赖个数，Stage 数 = 宽依赖数 + 1 3. 依赖关系判断： - 窄依赖：父 RDD 分区仅被子 RDD 一个分

	区依赖（如 map/filter）； - 宽依赖：父 RDD 分区被子 RDD 多个分区依赖（如 groupByKey/reduceByKey）
内核与调优	1. Shuffle 流程推导： - HashShuffle 优化前后对比（优化前：Task 数 ×Reduce 数文件；优化后：合并 buffer，减少文件数）； - SortShuffle 流程（排序→溢写→合并） 2. 统一内存管理动态占用规则： - 存储内存不足时，借用执行内存空闲区域； - 执行内存不足时，强制回收存储内存借用部分； - 双方均不足时，数据落盘 3. 任务数计算： - 逻辑：Stage 中最后一个 RDD 的分区数 = 该 Stage 的 Task 数
实战案例	1. Top10 热门品类排序逻辑推导： - 优先级：点击数＞下单数＞支付数； - 自定义 Comparable 接口实现比较逻辑 2. 区域热门商品城市备注推导： - UDAF 预聚合（统计城市点击数）→排序取前 2→剩余归为 “其他”→计算占比

三、需实操的知识点（占比 35%）

对应考纲模块	具体知识点
运行模式	1. Local 模式实操： - 解压 Spark 安装包、执行官方 PI 案例（bin/spark-submit --class org.apache.spark.examples.SparkPi --master local[2] 示例jar 10）； - 4040 端口查看任务监控 2. YARN 模式实操： - 配置 yarn-site.xml（关闭内存检查）、spark-env.sh（YARN_CONF_DIR）； - 启动 HDFS/YARN 集群、提交任务（`--master yarn`）； - 配置历史服务（spark-defaults.conf 日志路径、18080 端口访问）
Spark Core	1. RDD 算子实操： - 代码编写：parallelize 创建集合 RDD、textFile 读取文件 RDD、map/flatMap/filter/reduceByKey 算子组合使用； - 异常处理：Windows 下 Hadoop 依赖问题（

	配置 HADOOP_HOME、复制 hadoop.dll 到 System32） 2. 序列化实操： - Kryo 序列化配置（`spark.serializer=org.apache.spark.serializer.KryoSerializer`+ 注册自定义类）； - 解决序列化异常（自定义类实现 Serializable） 3. 持久化实操： - Cache 使用（`rdd.cache()`）、Checkpoint 配置（`sc.setCheckpointDir()`+`rdd.checkpoint()`）
Spark SQL	1. 编程实操： - SparkSession 创建（`SparkSession.builder().config(conf).getOrCreate()`）； - SQL 方式：创建临时视图 + 执行 SQL（如-`select * from t1 where age>18`）； - DSL 方式：`col()`函数、字段操作（`col("age").plus(1)`） 2. 自定义函数实操： - UDF：定义函数（如添加后缀）、注册（`spark.udf().register()`）、SQL 调用； - UDAF：继承 Aggregator 实现求平均年龄 3. 数据交互实操： - 读取 CSV：- 指定 header / 分隔符（`option("header","true").option("sep",",").csv()`）； - 读写 MySQL：配置 JDBC 参数（user/password/URL）、`read.jdbc()`/`write.jdbc()`； - 集成 Hive：添加 hive-site.xml、-`enableHiveSupport()`
Spark Streaming	1. 环境配置： - 添加 Kafka 依赖（-spark-streaming-kafka-0-10_2.12）； - 日志配置（log4j2.properties 关闭冗余日志） 2. 代码实操： - StreamingContext 创建（`new JavaStreamingContext("local[*]", "name", Duration.apply(3000))`）； - Kafka 集成：`KafkaUtils.createDirectStream()`配置 Consumer 参数； - 窗口操作：`window(Duration.apply(12000), Duration.apply(6000))`； - 优雅关闭：监控 HDFS 文件（`fs.exists(new Path("/stopSpark"))`）+-`javaStreamingContext.stop(true, true)`
实战案例	1. WordCount 实操： - 本地调试：代码编写（textFile→flatMap→

	mapToPair→reduceByKey）、断点查看 RDD 转换； - 集群运行：打包（maven package）、上传 jar、`spark-submit`提交（指定 class/ master/ 输入输出路径） 2. Top10 热门品类实操： - 数据转换：文本文件→UserVisitAction 对象； - 统计逻辑：flatMap 拆分点击 / 下单 / 支付记录→reduceByKey 聚合→sortBy 排序； - 序列化配置：注册自定义类（UserVisitAction /CategoryCountInfo）

> （注：文档部分内容可能由 AI 生成）