# 《大数据综合项目实训》练习02

## 一、业务背景

某电商平台需要对**商品基础信息数据**进行集中管理与处理，需要完成一次完整流程：
**商品数据采集 → Kafka 缓冲 → Spark on YARN 处理 → 写入 Hive → 查询验证**

日志文件：商品信息数据文件 `test02.log`

数据格式如下：

| 商品ID | 商品名称 | 商品分类 | 商品价格 | 上架时间 |
| --- | --- | --- | --- | --- |
| P3001 | 经典文学全集 | 图书 | 198 | 2025-02-01 09:00:00 |
| P3002 | 数据分析入门书籍 | 图书 | 128 | 2025-02-02 10:30:00 |
| P3003 | 春季休闲外套 | 服装 | 359 | 2025-02-03 11:15:00 |
| P3004 | 男士运动T恤 | 服装 | 159 | 2025-02-04 14:20:00 |
| P3005 | 有机坚果礼盒 | 食品 | 269 | 2025-02-05 16:40:00 |
| P3006 | 进口巧克力组合 | 食品 | 199 | 2025-02-06 09:25:00 |
| P3007 | 儿童绘本套装 | 图书 | 149 | 2025-02-06 10:50:00 |
| P3008 | 女士针织毛衣 | 服装 | 299 | 2025-02-07 11:30:00 |
| P3009 | 低糖即食燕麦 | 食品 | 89 | 2025-02-07 15:05:00 |
| P3010 | 经典名著精装版 | 图书 | 228 | 2025-02-08 16:20:00 |

## 二、内容

### 第 1 题：启动与检查环境

#### 1.1 启动 HDFS

在 master 执行：

```
start-dfs.sh
```

#### 1.2 启动 YARN

在 slave1 执行：

```
start-yarn.sh
```

### 1.3 启动 Zookeeper

在三台节点执行：

```
zkServer.sh start
```

### 1.4 启动 Kafka

在三台节点执行：

```
kafka-server-start.sh -daemon /opt/apps/kafka/config/server.properties
```

## 第 2 题：Kafka 创建 Topic 并验证

### 2.1 创建 Topic

在 slave1 节点创建 Kafka 主题，主题名称：`test_topic_02`

```
kafka-topics.sh \
--create \
--topic test_topic_02 \
--bootstrap-server master:9092 \
--partitions 3 \
--replication-factor 1
```

### 2.2 验证 Topic

```
kafka-topics.sh --list --bootstrap-server master:9092
```

### 2.3 启动 Kafka 消费者

在 slave1 节点执行：

```
kafka-console-consumer.sh \
--bootstrap-server master:9092,slave1:9092,slave2:9092 \
--topic test_topic_02 \
--from-beginning
```

---

## 第 3 题：Kafka 生产者发送商品数据

### 3.1 准备商品数据文件

将日志文件 `test02.log` 复制到 slave2 的指定目录 `/opt/data/flume/logs`：

```
mkdir -p /opt/data/flume/logs
```

### 3.2 在slave2将商品数据发送到 Kafka 主题

```
cat /opt/data/flume/logs/test02.log | \
kafka-console-producer.sh \
--broker-list master:9092,slave1:9092,slave2:9092 \
--topic test_topic_02
```

**3.3 在slave1节点查看 Kafka 消费结果**

---

## 第 4 题：Hive 建表

**任务：创建 `test_table_02` 商品信息表**

**表含义说明：**商品信息表用于记录电商平台中商品的基础信息，包括商品名称、分类、价格及上架时间等内容。

**字段说明**

| 中文字段 | 字段名 | 数据类型 | 说明 |
| --- | --- | --- | --- |
| 商品ID | product_id | STRING | 商品唯一标识 |
| 商品名称 | product_name | STRING | 商品名称 |
| 商品分类 | category | STRING | 商品所属分类 |
| 商品价格 | price | DOUBLE | 商品销售价格 |
| 上架时间 | create_time | STRING | 商品上架时间 |

**完成下面的Hive任务**

**4.1 在 master节点启动Hive环境：**

```
# 启动元数据服务
hive --service metastore &
# 启动Hive
hive
```

**4.2 选择数据库**

```
use default;
```

**4.3 删除数据表**

```
DROP TABLE IF EXISTS test_table_02;
```

**4.4 创建商品信息表**

```
CREATE TABLE IF NOT EXISTS test_table_02 (
  product_id   STRING,
  product_name STRING,
  category     STRING,
  price        DOUBLE,
  create_time  STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

#### 4.5 查看数据表

```
show tables;
```

---

## 第 5 题：在 spark-shell 中完成 Kafka → Spark → Hive 数据处理任务

在 FinalShell 新建一个master节点窗口，启动 spark-shell：

```
spark-shell \
--master yarn \
--deploy-mode client \
--driver-memory 512m \
--conf spark.driver.maxResultSize=256m \
--conf spark.executor.instances=1 \
--conf spark.executor.cores=1 \
--conf spark.executor.memory=512m \
--conf spark.sql.shuffle.partitions=2 \
--conf spark.sql.catalogImplementation=hive \
--conf spark.driver.extraClassPath=/opt/apps/hive/conf \
--conf spark.executor.extraClassPath=/opt/apps/hive/conf \
--packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.1.1
```

#### 5.1 从 Kafka 读取商品数据

```
val kafkaDF = (
    spark.read
  .format("kafka")
  .option("kafka.bootstrap.servers", "master:9092")
  .option("subscribe", "test_topic_02")
  .load()
)
```

#### 5.2 提取 Kafka 中的 value 字段

```
val lines = kafkaDF.selectExpr("CAST(value AS STRING) AS line")
```

### 5.3 解析商品 CSV 数据（5 列）

```
val dataDF = lines.selectExpr(
  "split(line, ',')[0] AS product_id",
  "split(line, ',')[1] AS product_name",
  "split(line, ',')[2] AS category",
  "cast(split(line, ',')[3] AS double) AS price",
  "split(line, ',')[4] AS create_time"
)
```

### 5.4 切换数据库并查看表

```
spark.sql("USE default")
spark.sql("show tables").show(false)
```

### 5.5 将 DataFrame 写入 Hive 表

```
dataDF.coalesce(1).write.mode("append").insertInto("test_table_02")
```

### 5.6 查询 Hive 表前5行数据，验证写入结果

```
spark.sql("SELECT * FROM test_table_02 LIMIT 5").show(false)
```