

实验5：实时统计与恢复

2学时 · 起点镜像 L05

今天的成果：**新数据更新窗口，保留状态后能继续运行**

实验任务与90分钟安排

阶段	时间	成果
概念与手算	15分钟	10秒窗口及均值
补全与运行	30分钟	五条窗口结果
保留状态重启	20分钟	同输入结果保持一致
实时观察与核对	25分钟	新文件更新、告警解释

本实验使用本地Spark和六条传感器记录。

批处理与持续查询

方式	输入视角	执行方式
批处理	当前一份有限数据	处理完成后结束
持续查询	持续到达的新文件	小批次增量更新状态
availableNow	启动时可见的数据	处理到当前输入末尾后结束

三者都可用确定样例核对结果。

本次使用Structured Streaming

有限的传感器JSON文件持续到达

使用DataFrame表达窗口计算

小批次更新统计结果

理论课的DStream基于RDD微批次；本课程采用Structured Streaming DataFrame API。

起点与三个目录

```
labctl stop  
cd ~/work/lab05
```

input: 完整输入文件

checkpoint: 进度与状态

result.json: 用于核对的小型结果

传感器记录与schema

```
{ "event_time": "2026-10-27T08:00:01Z",  
  "sensor": "A", "value": 20.0 }
```

列	Spark类型	含义
event_time	TimestampType	UTC事件时间
sensor	StringType	传感器编号
value	DoubleType	本次测量值

输入、状态与结果的职责

路径	内容	恢复时处理
input	已发布完整JSON文件	保留
checkpoint	进度、提交记录与聚合状态	保留原查询状态
result.json	当前窗口结果表	核对并继续更新
live-input等	独立实时演示数据	与规定样例分开

同一查询使用固定的输入与checkpoint组合。

事件时间与到达时间

`event_time` 表示事件何时发生。

文件落到input的时刻是它到达系统的时间。

旧时间戳的文件，也可能现在才到达。

六条事件的时间顺序

事件时间UTC	传感器	值
08:00:01	A	20
08:00:05	A	30
08:00:09	B	40
08:00:12	A	18
08:00:14	B	22
08:00:21	A	26

左闭右开的窗口边界

时间	所属10秒窗口
08:00:00	08:00:00, 08:00:10)
08:00:09	08:00:00, 08:00:10)
08:00:10	08:00:10, 08:00:20)
08:00:20	08:00:20, 08:00:30)

左端点属于本窗口，右端点进入下一窗口。

窗口统计的数量与均值

对于窗口W和传感器s:

$$n_{W,s} = |E_{W,s}|, \quad \bar{x}_{W,s} = \frac{\sum_{e \in E_{W,s}} value_e}{n_{W,s}}$$

08:00:00窗口的A: $n = 2$, $\bar{x} = (20 + 30)/2 = 25$ 。

按窗口和传感器共同分组，保留两列分组键。

10秒滚动窗口

08:00:00窗口： [08:00:00, 08:00:10)

传感器	窗口内数值	均值
A	20、30	25
B	40	40

10秒整的事件属于下一个窗口。

微批次、触发间隔与窗口长度

参数	决定的内容
processingTime=5秒	持续查询检查与处理的触发节奏
maxFilesPerTrigger=2	每批最多读取两个文件
window=10秒	按事件时间划分的统计区间
生成间隔2秒	新文件到达节奏

窗口长度与实际一次处理耗时分别观察。

窗口表达式与结果列名

```
F.window("event_time", "10 seconds")  
F.count("*").alias("count")  
F.avg("value").alias("avg")
```

把窗口与sensor放入 `groupBy`，聚合结果命名为 `summary`。

给定写入器读取 `window`、`sensor`、`count`、`avg`。

补全一个聚合表达式

在stream.py中得到summary:

- 按10秒window和sensor分组

- 统计记录数, 列名count

- 统计value均值, 列名avg

其余读取、写入和退出代码已经提供。

完整文件的原子发布

生成器采用以下顺序：

写入隐藏临时文件 `.event-0000.tmp`

写完一条完整JSON并关闭文件

原子改名为 `event-0000.json`

Spark读取正式输入文件

正式文件名保持唯一，已发布内容保持稳定。

complete模式与结果替换

每个批次向写入函数提供完整聚合表

写入器生成临时JSON，再替换结果文件

当前结果包含截至本批次的全部窗口

相同完整结果再次写出时，文件内容保持相同

本例的输出是一个小型本地结果表。

先处理固定输入

```
python generate.py --interval 0  
spark-submit stream.py --once  
cat result.json  
labctl check lab05
```

availableNow处理当前可见文件后退出。

核对全部结果

窗口起点	传感器	数量	均值
08:00:00	A	2	25
08:00:00	B	1	40
08:00:10	A	1	18
08:00:10	B	1	22
08:00:20	A	1	26

恢复前后的比较方法

```
cp result.json first-result.json
spark-submit stream.py --once 2>&1 | tee restart.log
diff first-result.json result.json
ls checkpoint/commits checkpoint/offsets
```

同一查询、同一输入和状态，且无新文件时，结果保持一致。

checkpoint中的三类信息

信息	作用
输入进度	已识别与处理的文件位置
批次提交	已完成批次的记录
聚合状态	窗口计数、和等中间状态

原查询恢复时使用原目录；新查询采用新目录。

保留状态再运行

同一输入、同一查询、同一checkpoint:

```
spark-submit stream.py --once  
cat result.json
```

无新文件时，统计结果保持一致。

持续观察新文件

终端1

```
spark-submit stream.py --input live-input \  
    --checkpoint live-checkpoint --output live-  
result.json
```

终端2

```
python generate.py --output live-input --interval 2
```

使用独立演示目录，与规定核对目录分开保存。

实时观察的三个终端

终端	任务
1	spark-submit持续查询
2	生成器分批发布新文件
3	查看live-result.json与日志

先启动查询，再发布数据，最后停止查询。

每次观察记录最新batch编号与窗口结果。

告警是一个可解释的条件

本次个人修改：窗口均值大于30时告警。

B在08:00:00窗口：40，告警

A在08:00:00窗口：25，不告警

等于30是否告警？按“大于”定义，不告警

文件进度与业务事件去重

输入操作	结果口径
保留原文件与原checkpoint重启	按已记录进度继续
相同事件写入另一个新文件	作为新输入参与本例聚合
按业务event_id去重	需要明确额外规则和状态

实训包含event_id，用于进一步讨论业务重复。

checkpoint与查询恢复

记录文件处理进度与聚合状态

同一查询重启后继续处理

文件进度与业务去重分别设计

修改窗口或查询结构时，旧状态可能不兼容

新查询用新checkpoint；保留旧目录用于原查询恢复。

有限输入的状态策略

complete模式保留所有窗口聚合状态

有限输入采用完整状态保留策略

持续无限输入会使状态增长

同内容换文件名再次投放可能再次计数

本次恢复采用原查询、原输入与原checkpoint。

状态规模与watermark扩展

基础任务：有限输入、complete模式、保留全部窗口。

持续输入时，窗口状态会随时间增长。

扩展设计需确定：

- 允许多晚到达的事件

- 何时清理旧窗口状态

- 迟到事件到来后如何处理

- 输出模式与状态清理的匹配关系

常见错误与诊断

现象	检查
找不到summary	TODO变量名与异常是否替换
列名不存在	count、avg及window结构
结果未更新	新文件、查询日志、输出路径
重启后报状态不兼容	查询结构与checkpoint组合
count增加超出预期	新文件是否包含重复事件

按输入、查询、状态、输出顺序定位。

报告截图核对

图1 首次窗口结果（报告3 处理已有文件）

图2 实时更新前（报告4 保留状态重启与实时更新）

图3 实时更新后（报告4 保留状态重启与实时更新）

截图内容与图号按Word模板填写，直接插入对应位置。

提交Word报告

“源码与运行说明”： stream.py全文、 alert.py全文或stream.py中独立告警输出代码

完成各步骤实际结果、规定截图和已有书面栏目

AI使用情况： 简短说明或一张截图； 未使用填“未使用”

保存后重新打开， 核对文本和截图清晰可读

正式提交一份填写完成的Word报告。