Software Defined Infrastructure

May. 2016

SMB-2 Detailed Design

M. Genkin

April 10/2017

Agenda

- SMB-2 overview
- Workloads and use cases
- Metrics
- Technical decisions
- Implementation
- Workloads
  - Synchronous-interactive
  - Asynchronous-batch
- Use Cases/Test Cases
  - Use case 1 – Synch-interactive multi-user
  - Use case 2 – Asynch-batch multi-user
  - Use case 3 – Mixed multi-user
  - Use case 4 – Mixed multi-tenant

SMB-2 Objective

- **Spark Multi-User Benchmark (SMB)** is designed to measure **resource manager performance under multi-user and multi-tenant conditions**:
  - Measure performance differences for resource manager software efficiency under
    - Short-running synchronous interactive jobs and queries executed in parallel by multiple users
    - Longer-running batch jobs and queries executed in parallel by multiple users
    - Mixed workloads – interactive+batch – executed in parallel by multiple users
    - Mixed workloads – interactive+batch – executed in parallel by multiple users with different qualities of service
      - Users from two different organizations with different QoS requirements execute a mixture of interactive and batch jobs on shared resources

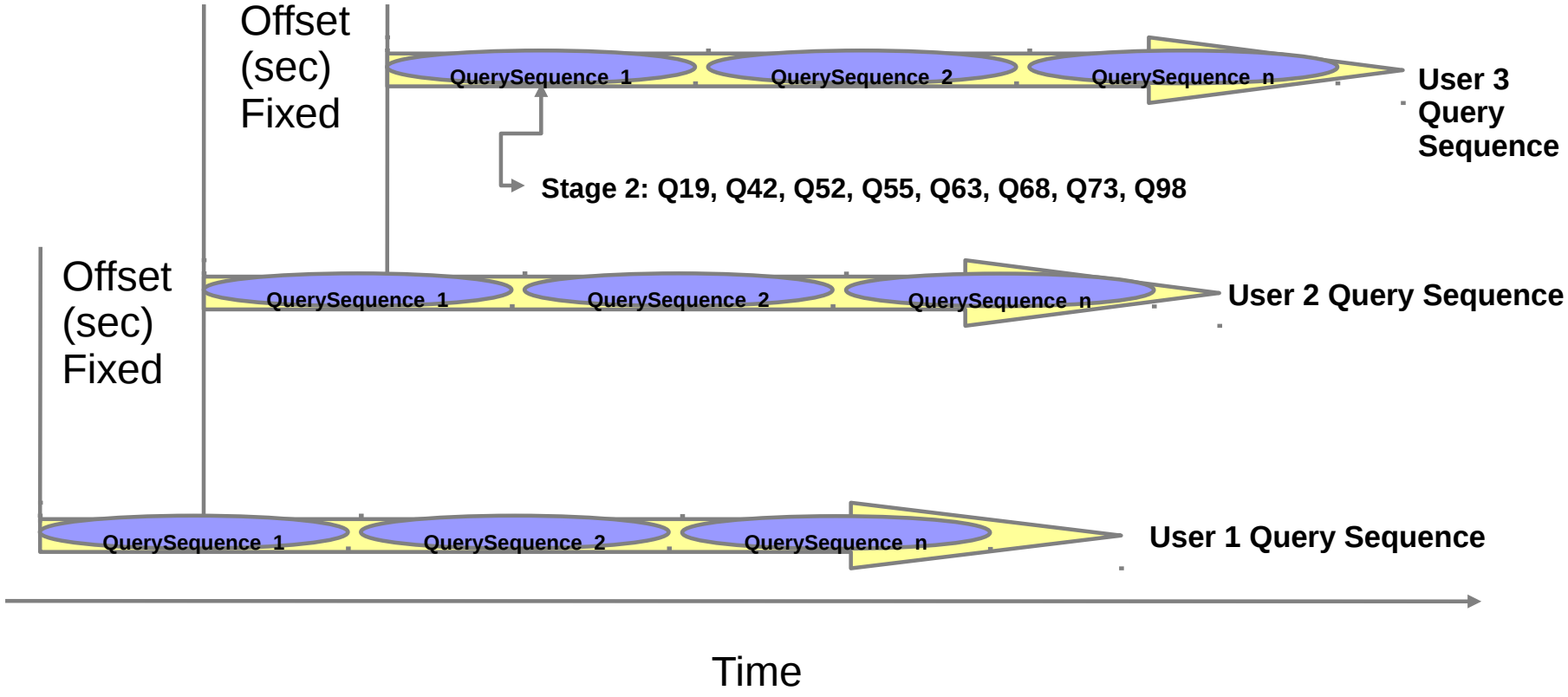
SMB-2 Use Cases And Workloads

- **SMB-2 will define 4 use cases (tests) based on two workload patterns:**
  - Workload patterns:
    1) Synchronous interactive workload
       - Short running queries – 5 to 20 sec duration
    2) Asynchronous batch workload
       - Longer running queries – 30 sec to 5 min duration
       - Machine learning jobs
  - Use cases:
    1) Synchronous interactive multi-user
       - All users have the same, equivalent QoS and weight
       - Uses workload pattern 1 only
    2) Asynchronous batch multi-user
       - All users have the same, equivalent QoS and weight
       - Uses workload pattern 2 only
    3) Mixed multi-user
       - Uses a mix of of workload 1 (50% of users) and workload 2 (50% of users)
       - All users have the same, equivalent QoS and weight
    – Mixed multi-tenant
       1) Uses a mix of of workload 1 (50% of users) and workload 2 (50% of users)
       - All users have different QoS and weight

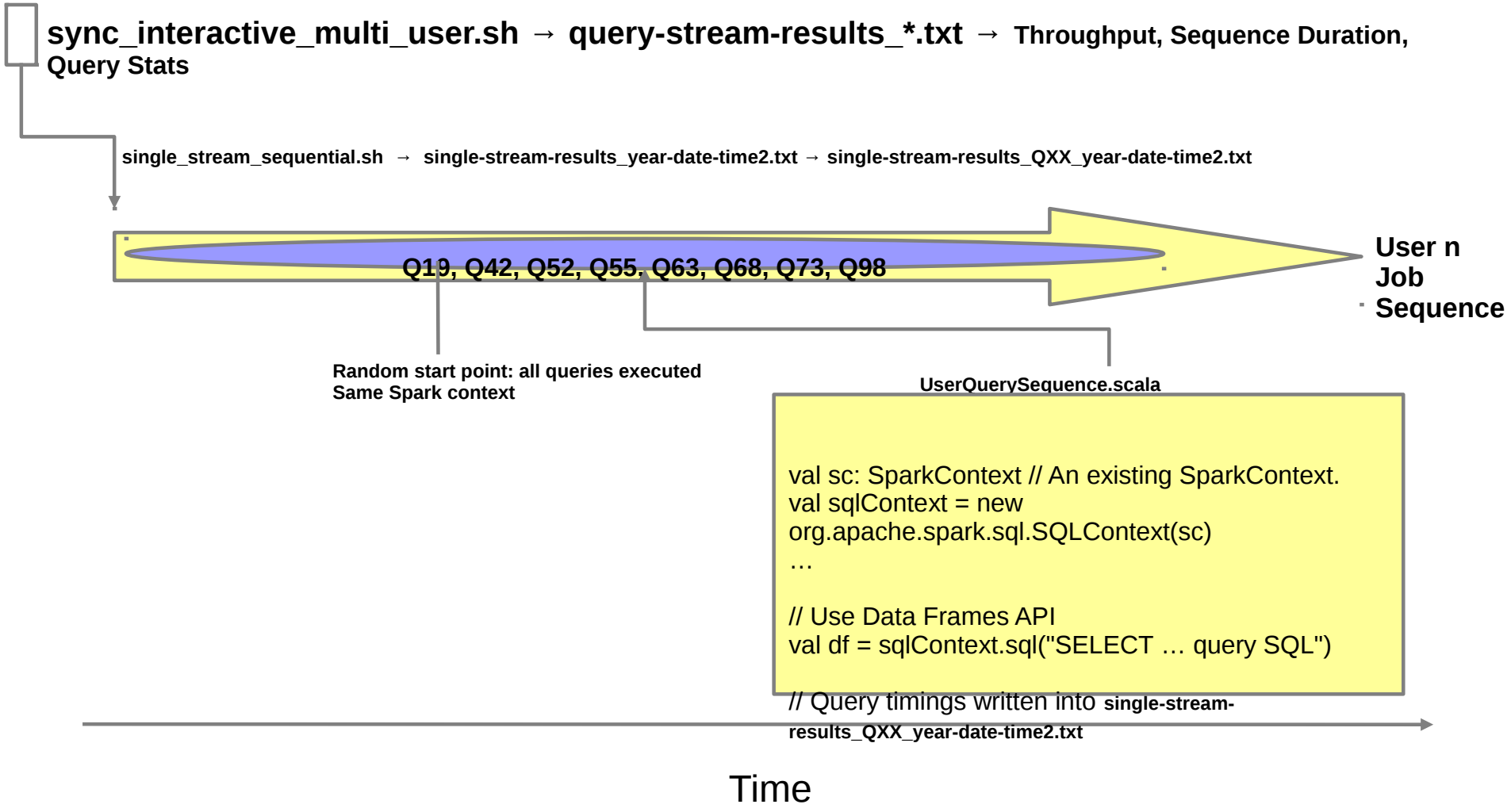
Technical Decisions

- **Use queries, schema and data generator from Cloudera GitHub**
  - IBM kit requires legal review/approval for exposure to outsiders
  - IBM kit is targeting Spark 2.0, although the desired query set should work with Spark 1.6.x
  - Possible leagal issues associated with using Cloudera artifacts – legal check in-progress
- User query sequences implemented as Scala program
- Will not implement query qualification substitution parameters as in TPC-DS
  - Instead will randomize start point in query sequence execution
  - Each user querly sequence will involve all queries but starting point will vary randomly
- Data stored as csv files on HDFS

SMB-2: Sync-Interactive Workload



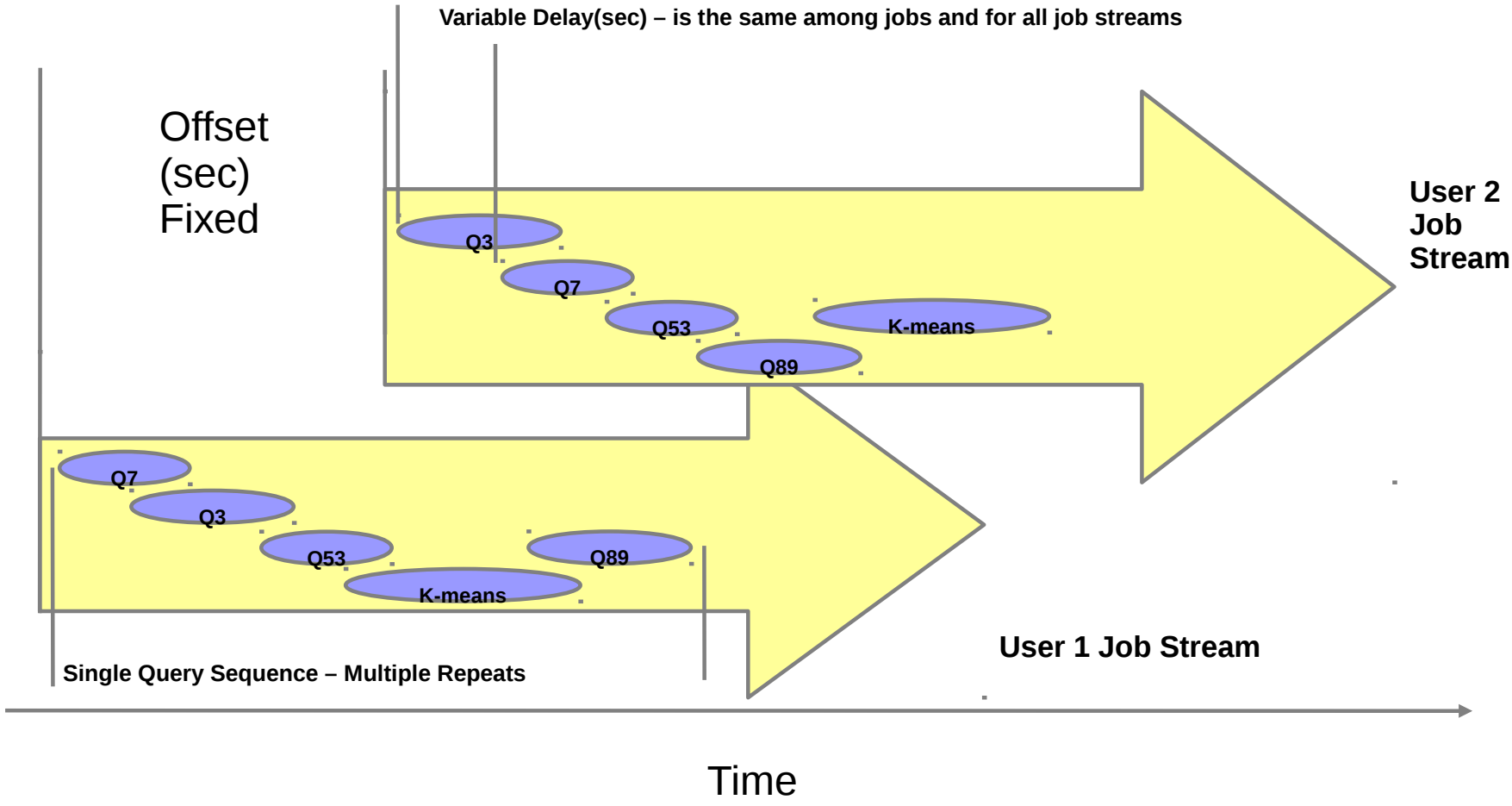
SMB-2 Sync-Interactive Workload Impl.



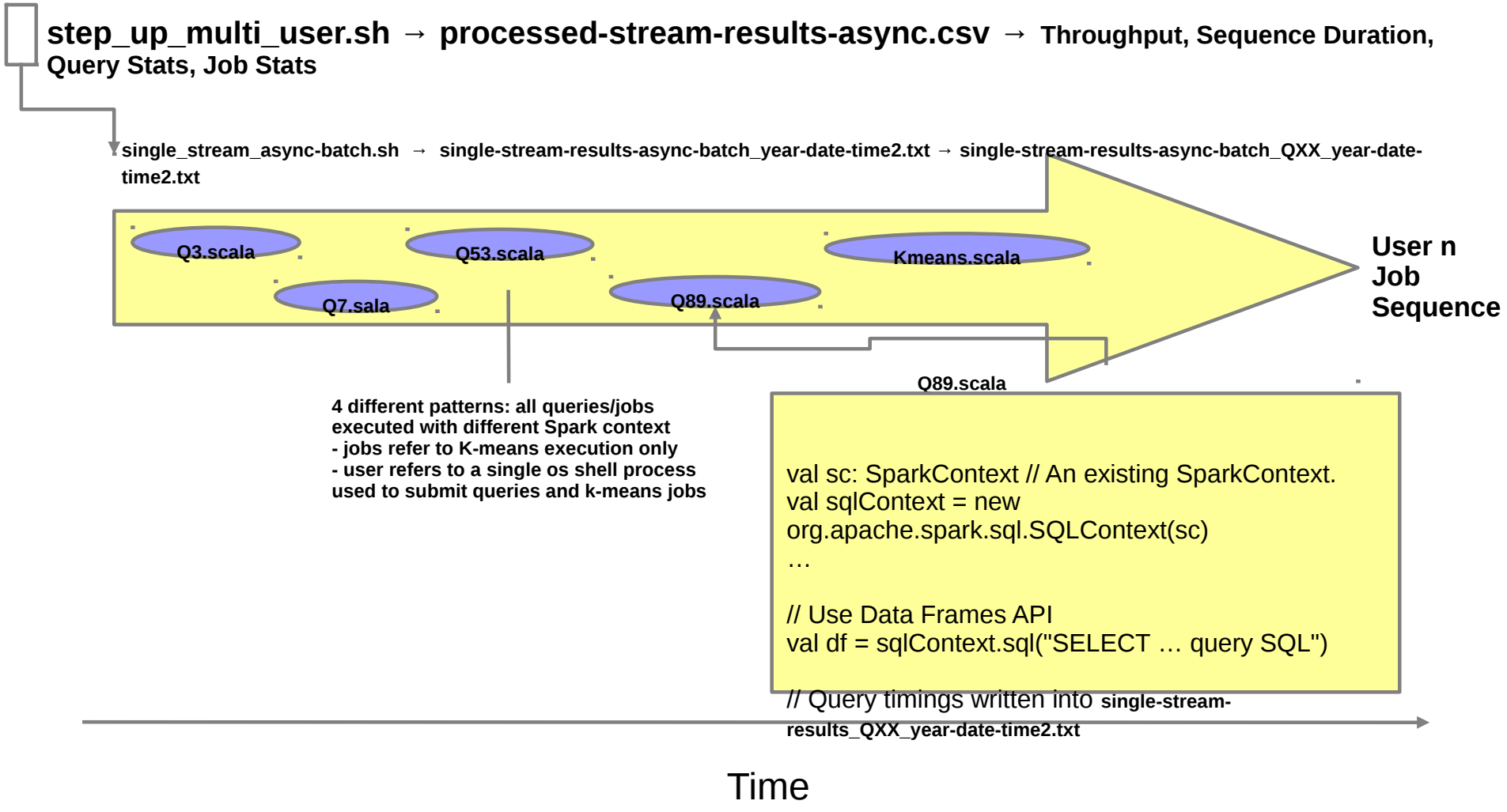
UserQuerySequence.scala

- **Implementation details:**
  - Query sequences:
    1) Q19, Q42, Q52, Q55, Q63, Q68, Q73, Q98
    2) Q42, Q52, Q55, Q19, Q68, Q73, Q98, Q63
    3) Q52, Q19, Q55, Q68, Q73, Q98, Q63, Q42
    4) Q68, Q19, Q55, Q73, Q98, Q63, Q42, Q52
    5) Q63, Q19, Q55, Q98, Q73, Q42, Q52, Q68
    6) Q73, Q63, Q19, Q55, Q98, Q42, Q68, Q52
    7) Q98, Q68, Q63, Q19, Q55, Q42, Q73, Q52
    8) Q55, Q52, Q68, Q63, Q19, Q42, Q73, Q98
    9) Q19, Q98, Q42, Q52, Q55, Q63, Q68, Q73
    10) Q42, Q19, Q68, Q52, Q55, Q73, Q98, Q63
  - One of the above query sequences is chosen at random when each query job sequence is started.
  - This ensures that most users are not running the same query at tha same time

SMB-2 Benchmark
Asynch-Batch Workload



SMB-2 Async-Batch Workload Impl.



single_stream_async-batch.sh

- **Implementation details:**
  - Query sequences:
    1) Q3, Q7, Q53, Q89, K-means
    2) K-means, Q53, Q89, Q7, Q4
    3) Q53, Q3, K-means, Q89, Q7
    4) Q89, K-means, Q7, Q3, Q89
  - One of the above query sequences is chosen at random when each query job sequence is started.
  - This ensures that most users are not running the same query at the same time

Use Case 1 – Sync-Interactive Multi-User

- **Implementation details:**
  - **step_up_multi_user.sh**
    - **N users/streams (total)**
    - **M interactive users (must be less than total)**
      - **M=N for this use case i.e. all interactive**
    - **Offset**
    - **X iterations**
  - **Summary:**
  - **New parameter added to step_up_multi_user.sh to convey how many interactive vs. batch streams to start, in this case all streams run synchronous interactive workload**

Use Case 2 – Async-Batch Multi-User

- **Implementation details:**
  - **step_up_multi_user.sh**
    - **N users/streams (total)**
    - **M interactive users (must be less than total)**
      - **M=0 for this use case i.e. no interactive users**
    - **Offset**
    - **X iterations**
    - **Summary:**
      - **New parameter added to step_up_multi_user.sh to convey how many interactive vs. batch streams to start, in this case all streams run asynchronous, batch workload**

Use Case 3 – Mixed Multi-User

- **Implementation details:**
  - **step_up_multi_user.sh**
    - **N users/streams (total)**
    - **M interactive users (must be less than total)**
      - **M=1/2N for this use case i.e. 50% interactive**
    - **Offset**
    - **X iterations**
  - **Summary:**
    - **New parameter added to step_up_multi_user.sh to convey how many interactive vs. batch streams to start, in this case 50% of the user streams run synchronous interactive workload, and 50% run asynchronous batch workload.**

Use Case 4 – Mixed Multi-Tenant

- **Implementation details:**
  - **step_up_multi_user.sh**
    - **N users/streams (total)**
    - **M interactive users (must be less than total)**
      - **M=1/2N for this use case i.e. 50% interactive**
    - **Offset**
    - **X iterations**
    - **Summary:**
      - **New parameter added to step_up_multi_user.sh to convey how many interactive vs. batch streams to start, in this case 50% of the user streams run synchronous interactive workload, and 50% run asynchronous batch workload.**
      - **In addition to script parameters for this use case a different weighting of resource allocation will be used:**
        - **70% of CPU and Memory for sync-interactive user streams**
        - **30% of CPU and Memory for async-batch user streams**

SMB Benchmark Theory And Metrics

- Coarse metrics:
  - Total test duration
  - Combined query throughput
- Per-query metrics:
  - Duration of each query executed is proportional to resources allocated by the resource manager
  - Plot of query duration for all queries vs. test duration should show a pattern simiar to figure on the right
  - Query duration data can be used to calculate key metrics related to resource manager efficiency:
    - Throughput
    - Query duration
    - Query duration variance