| **Task** | **Description** | **Expected Result** | **Risks** |
| --- | --- | --- | --- |
| Step 1 | In the lower test environment, compare the outcomes of both workflows (one with custom task and another with the current production version). | Both workflows produce the same output, but the custom workflow should have a shorter run time, demonstrating performance improvement without compromising data quality. | Discrepancies may lead to defect analysis and turnaround effort, impacting the test completion timeline and potentially delaying TSR production on Monday. |
| Step 2 | In the pre-production environment, rerun the custom task after Databricks resolved the previous job run issue. Ensure the custom workflow runs without any scheduling or job run problems. | The custom workflow should not fail, and orchestration should run within the benchmark timeline. | A solution may be needed in the production environment to avoid full data refresh while replacing the existing workflow with the custom one. |
| Step 3 | Perform non-functional high-level cost and performance testing on Tuesday and Wednesday, analyzing costs by monitoring the workflow similar to production. | The cost analysis should demonstrate cost savings from the custom workflow implementation. | The storage cost analysis cannot be fully done due to shared accounts with other sources. The workflow run scheduling pattern may need to be changed in pre-prod to see differences in the cost, which could be time-consuming. It is essential to check with Sumit if this approach aligns with the release planned for next week Thursday. |