Item 217
git_comments:

1. As of 1.3.2, Moto doesn't support select_object_content yet.

git_commits:

1. **summary:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   message: [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. Closes #3227 from sekikn/AIRFLOW-2299
   label: code-design

github_issues:

1. **title:** Can one operator output to other operator input?
   body:
2. **title:** Can one operator output to other operator input?
   body:
3. **title:** Can one operator output to other operator input?
   body:
4. **title:** Can one operator output to other operator input?
   body:
5. **title:** Can one operator output to other operator input?
   body:
6. **title:** Can one operator output to other operator input?
   body:
7. **title:** Can one operator output to other operator input?
   body:
8. **title:** Can one operator output to other operator input?
   body:
9. **title:** Can one operator output to other operator input?
   body:
10. **title:** Can one operator output to other operator input?
    body:
    label: code-design
11. **title:** Can one operator output to other operator input?
    body:
    label: code-design
12. **title:** Can one operator output to other operator input?
    body:
13. **title:** Can one operator output to other operator input?
    body:
14. **title:** Can one operator output to other operator input?
    body:
    label: code-design
15. **title:** Can one operator output to other operator input?
    body:
16. **title:** Can one operator output to other operator input?
    body:
17. **title:** Can one operator output to other operator input?
    body:
18. **title:** Can one operator output to other operator input?
    body:
19. **title:** Can one operator output to other operator input?
    body:

github_issues_comments:

1. +1
2. +1
3. +1

4. This would shift Airflow DAGs from dependency graphs to dataflow graphs, which brings some interesting issues. Output data would need to be serialized to the Airflow database to ensure execution could continue if workers need to be restarted.
5. +1
6. I need a few use cases to understand what everyone is trying to achieve with this feature. It seems fairly straightforward to publish a message for a task downstream and have that task downstream pick it up, and bring it up into its context (making it available in the operator's `execute` method as well as in the templates). It's also safe to assume that the message could be a "pickleabe" python object of limited size (probably smaller than 1MB). Right? The question is how would the message be published? Is it an operator feature or a task feature? By that I mean is it baked into an operator, or is it baked when creating the task? Let's assume it's a task level feature. What if we had a parameter of `BaseOperator` called `drop_message` that expects a callable and calls it back after executing the task, passing it the context. Whatever is returned by that callable gets serialized and put into the database, and associated with that task instance. Downstream tasks can specify a `read_message`, most likely as a list of references to upstream tasks it's expecting messages from. These messages are made available in the context. Would that work for your use cases?
7. My use case: I have a mount folder. I need load the data to database. My steps: 1. Use BashOperator to list the data 2. Use HiveOperator to create tables and load data So, I want to use BashOperator output to HiveOperator params
8. And there are no way for you to know ahead of time what the filenames are going to be? Are all the files targeting the same table? Maybe you could write a FolderToHiveTransfer operator, or just a PythonOperator that does that. ?
9. Yep, the files are different tables. I do not how many, which table they have. I will try use pythonoperator do that.
10. **body:** I think it's a somewhat legit use case, though it'd be nice to have a more defined "contract" with whoever drops those files. If you could have predictable filenames/folder and time interval it really simplifies ETL. Not only it simplifies the ETL logic, but the way the metadata and matching logs are organized. But I understand that sometimes that's just the way it is and you have to deal with it...
    label: code-design
11. **body:** Keeping open as this feature is still needed. I'd like to hear as many use cases as possible before designing a solution.
    label: code-design
12. Hi, My scenario is : - One job is fetching an endpoint to get access_token via OAuth - My second job needs this access_token to perform the query on a second endpoint. I could do this in the same job but i had like to separate them. Regards,
13. Seems like a very small unit of work, you could easily squeeze both in the same `PythonOperator`. Are you concerned about minimizing the number of hits on the first endpoint?
14. **body:** I think the broader issue here is the difficulty of of building run-time dynamic pipelines, where the tasks and dependencies vary based on factors discovered during the run. I don't see a great way to handle that in Airflow. Here are some rambling thoughts... This manifests in two ways: 1. a task whose parameters depend on information discovered by earlier tasks (as described by @griffinqiu and @kwent) 2. a task whose existence depends on information discovered by earlier tasks (haven't seen anyone mention this yet, but I think it's the larger issue -- basically a superset of the first one) Issue 1) could be solved with a convenience function for serializing data to the database. That could be injected into the context, as @mistercrunch wrote, or simply persisted in a table (it seems like Variables, or something like Variables, are the right approach here. Using callbacks and context requires users to first store objects in context, then write a callback... it seems like the logic could be wrapped up nicely in a function and sanitized/checked for compatibility in the process. I will open a separate issue to discuss this.) I think, broadly speaking, that it's healthy to split up any logical task into small pieces even if the unit of work is small, if only because it leads to better monitoring of progress and errors. Issue 2) is much harder and @griffinqiu's issue could be viewed as a variant of it. Basically, we don't know in advance what tasks we need to run. I think the most common variant of this is "vertical" scaling (vertical in the context of the web UI), where I know what tasks I will run, but I don't know how many or with what parameters (i.e. how many vertical rows I'll see in the web UI). For example: - I want to process every file in a directory (local, FTP, etc.). I don't know the filenames in advance because someone may have put a new file in there since the last time. I also don't want to reprocess files I've already seen. As nice as it would be to have a formal contract with all data, often that's just not the case -- files show up in funny ways and times. So I know the tasks I need to run, but not the input or count - I want to run a pipeline with parameters that were produced by another process or pipeline. I don't know in advance how many different sets were produced, so I don't know how many times to run the pipeline. For example, build aggregations of data every X minutes, where X is an array of values generated externally that varies day to day. I don't know X until I reach that point in the pipeline and access it. Airflow is built around the concept of DAG discovery taking place entirely separately from running them. So this is hard to reconcile with the idea of dynamic DAG building. But perhaps a `DAGOperator` could generate a DAG at runtime and run it ad-hoc, then delete it. However I'm not sure how it would play with the interface (in terms of displaying progress and also knowing if tasks have already been run on subsequent runs). Food for thought... this

is a problem we deal with in our pipeline so I will continue to think about it and would love any other perspective.
label: code-design

15. For what it's worth, +1. This is a showstopper for our [Stanford] adoption, as much as we otherwise like airflow. For use cases and similar, consider Apache Spark's docs and particularly: http://spark.apache.org/docs/latest/streaming-programming-guide.html Regarding @jlowin's excellent analysis, 1 is a must have, and 2 a nice to have (we can call a bunch of no -op operators that return straight away when not appropriate, but not creating/calling them at all would be better!)
16. We're currently sketching a solution for this here, please join the PR and comment on the implementatoin details as it shapes up: https://github.com/airbnb/airflow/pull/232 I think it's taking shape nicely with a very integrated model. Now keep in mind that Airflow is a batch workflow engine, not a realtime stream processing engine like Storm or Spark Streaming, and that the current model works extremely well, at scale, for what it is intended. A similar system at Facebook that inspired some of the design decisions behind Airflow is used by hundreds of data people everyday to satisfy a very large array of use cases. I'm convinced there would be a way to solve your use cases with Airflow as it is today, but agree that cross task communication can be more elegant in many cases. @azaroth42, we use Airflow to trigger Spark, Cascading, Hive, MR, ML, Ruby, Python, MySql, and all sorts of other tasks at Airbnb. Airflow is the orchestra director, it glues all of these systems together in a nice symphony For the chunks of pipelines that you'd want to be very dynamic and shape-shifting (I'd like more specific use-cases still to understand what you are all up to!), maybe something external to Airflow like some platform, program, or service can satisfy these use cases. Assuming that this thing works in batch, Airflow would just coordinate this service with others.
17. http://pythonhosted.org/airflow/concepts.html#xcoms

github_pulls:

1. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
2. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression

- Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`

3. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
4. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
5. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always

need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
6. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
7. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative

mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`

8. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`
9. **title:** [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator
   body: Make sure you have checked _all_ steps below. ### JIRA - [x] My PR addresses the following [Airflow JIRA](https://issues.apache.org/jira/browse/AIRFLOW/) issues and references them in the PR title. For example, "\[AIRFLOW-XXX\] My Airflow PR" - https://issues.apache.org/jira/browse/AIRFLOW-2299 - In case you are fixing a typo in the documentation you can prepend your commit with \[AIRFLOW-XXX\], code changes always need a JIRA issue. ### Description - [x] Here are some details about my PR, including screenshots of any UI changes: Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. This PR upgrades boto3 to 1.7.0+, since [it supports S3 Select from that version] (https://github.com/boto/boto3/blob/develop/CHANGELOG.rst#170). I checked [boto3's Upgrading Notes] (https://boto3.readthedocs.io/en/latest/guide/upgrading.html) and confirmed that changes referred there doesn't affect Airflow. ### Tests - [x] My PR adds the following unit tests __OR__ does not need testing for this extremely good reason: - Added: - tests.hooks.test_s3_hook:TestS3Hook.test_select_key - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_select_expression - Updated and renamed: - tests.operators.test_s3_file_transform_operator:TestS3FileTransformOperator.test_execute_with_transform_script ### Commits - [x] My commits all reference JIRA issues in their subject lines, and I have squashed multiple commits if they address the same issue. In addition, my commits follow the guidelines from "[How to write a good git commit message](http://chris.beams.io/posts/git-commit/)": 1. Subject is separated from body by a blank line 2. Subject is limited to 50 characters 3. Subject does not end with a period 4. Subject uses the imperative mood ("add", not "adding") 5. Body wraps at 72 characters 6. Body explains "what" and "why", not "how" ### Documentation - [x] In case of new functionality, my PR adds documentation that describes how to use it. - When adding new operators/hooks/sensors, the autoclass documentation generation needs to be added. ### Code Quality - [x] Passes `git diff upstream/master -u -- "*.py" | flake8 --diff`

github_pulls_comments:

1. # [Codecov](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=h1) Report > Merging [#3227] (https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=desc) into [master] (https://codecov.io/gh/apache/incubator-airflow/commit/c7a472ed6b0d8a4720f57ba1140c8cf665757167?src=pr&el=desc) will **decrease** coverage by `<.01%`. > The diff coverage is `80.95%`. [![Impacted file tree graph](https://codecov.io/gh/apache/incubator-airflow/pull/3227/graphs/tree.svg?width=650&token=WdLKlKHOAU&height=150&src=pr)](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=tree) ```diff @@ Coverage Diff @@ ## master #3227 +/- ## ========== - Coverage 75.35% 75.35% -0.01%

========== Files 195 195 Lines 14553 14565 +12 ========== + Hits 10966 10975 +9 - Misses 3587 3590 +3 ``` | [Impacted Files](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=tree) | Coverage Δ | | |---|---|---| | [airflow/hooks/S3_hook.py](https://codecov.io/gh/apache/incubator-airflow/pull/3227/diff?src=pr&el=tree#diff-YWlyZmxvdy9ob29rcy9TM19ob29rLnB5) | `94.59% <80%> (-0.69%)` | :arrow_down: | | [airflow/operators/s3_file_transform_operator.py](https://codecov.io/gh/apache/incubator-airflow/pull/3227/diff?src=pr&el=tree#diff-YWlyZmxvdy9vcGVyYXRvcnMvczNfZmlsZV90cmFuc2Zvcm1fb3BlcmF0b3IucHk=) | `93.61% <81.25%> (-1.39%)` | :arrow_down: | | [airflow/models.py](https://codecov.io/gh/apache/incubator-airflow/pull/3227/diff?src=pr&el=tree#diff-YWlyZmxvdy9tb2RlbHMucHk=) | `87.33% <0%> (-0.05%)` | :arrow_down: | ------ [Continue to review full report at Codecov](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=continue). > **Legend** - [Click here to learn more](https://docs.codecov.io/docs/codecov-delta) > `Δ = absolute <relative> (impact)`, `ø = not affected`, `? = missing data` > Powered by [Codecov](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=footer). Last update [c7a472e...4ec4ef9](https://codecov.io/gh/apache/incubator-airflow/pull/3227?src=pr&el=lastupdated). Read the [comment docs](https://docs.codecov.io/docs/pull-request-comments).

2. @sekikn if the file storing encoded string, the `Payload` returned is bytes. At https://github.com/sekikn/incubator-airflow/blob/288fca445ffcad718d39f413eddd8712a18dbf85/airflow/hooks/S3_hook.py#L248, `".join()` will raise exception. ``` File "/usr/local/lib/python3.6/site-packages/airflow/hooks/S3_hook.py", line 249, in select_key for event in response['Payload'] TypeError: sequence item 0: expected str instance, bytes found ```

github_pulls_reviews:

1. Did you check in your own environment if this breaks anything?
2. This will break the API, can you add the arguments to the end?
3. I've been running Airflow with boto3 1.7.4 in a few days and didn't see any problem so far (my environment is small and not for production though...)
4. Sorry if I misunderstand, but I think it doesn't break the API since `apply_defaults` decorator does not allow positional argument. Users have to use keyword arguments here, so parameter position doesn't matter.
5. Ah check

jira_issues:

1. **summary:** Add S3 Select functionarity to S3FileTransformOperator
   description: S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it, but it's inefficient if the original file is large but the necessary part is small. S3 Select, [which became GA recently|https://aws.amazon.com/about-aws/whats-new/2018/04/amazon-s3-select-is-now-generally-available/], can improve its efficiency and usablitily.
2. **summary:** Add S3 Select functionarity to S3FileTransformOperator
   description: S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it, but it's inefficient if the original file is large but the necessary part is small. S3 Select, [which became GA recently|https://aws.amazon.com/about-aws/whats-new/2018/04/amazon-s3-select-is-now-generally-available/], can improve its efficiency and usablitily.
3. **summary:** Add S3 Select functionarity to S3FileTransformOperator
   description: S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it, but it's inefficient if the original file is large but the necessary part is small. S3 Select, [which became GA recently|https://aws.amazon.com/about-aws/whats-new/2018/04/amazon-s3-select-is-now-generally-available/], can improve its efficiency and usablitily.
4. **summary:** Add S3 Select functionarity to S3FileTransformOperator
   description: S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it, but it's inefficient if the original file is large but the necessary part is small. S3 Select, [which became GA recently|https://aws.amazon.com/about-aws/whats-new/2018/04/amazon-s3-select-is-now-generally-available/], can improve its efficiency and usablitily.
   label: code-design
5. **summary:** Add S3 Select functionarity to S3FileTransformOperator
   description: S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it, but it's inefficient if the original file is large but the necessary part is small. S3 Select, [which became GA recently|https://aws.amazon.com/about-aws/whats-new/2018/04/amazon-s3-select-is-now-generally-available/], can improve its efficiency and usablitily.

jira_issues_comments:

1. Commit 6e82f1d7c9fa391c636a0155cdb19aa6cbda0821 in incubator-airflow's branch refs/heads/master from [~sekikn] [https://git-wip-us.apache.org/repos/asf?p=incubator-airflow.git;h=6e82f1d] [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. Closes #3227 from sekikn/AIRFLOW-2299
2. Commit 6e82f1d7c9fa391c636a0155cdb19aa6cbda0821 in incubator-airflow's branch refs/heads/master from [~sekikn] [https://git-wip-us.apache.org/repos/asf?p=incubator-airflow.git;h=6e82f1d] [AIRFLOW-2299] Add S3 Select functionarity to S3FileTransformOperator Currently, S3FileTransformOperator downloads the whole file from S3 before transforming and uploading it. Adding extraction feature using S3 Select to this operator improves its efficiency and usablitily. Closes #3227 from sekikn/AIRFLOW-2299
3. Issue resolved by pull request #3227 [https://github.com/apache/incubator-airflow/pull/3227]