Talend User Component tGoogleAnalyticsInput

Purpose

This component addresses the needs of gathering Google Analytics data for a large number of profiles and fine-grained detail data.
The component uses the Core Reporting API 3.0 and the Authentication API OAuth 2.0 final.
To provide the ability to run in multiple iterations the component has special capabilities to avoid multiple logins through iterations. Usually automated processes should not use personal accounts. This requirement is addressed by using a service account, which are the only preferred way to login into Google Analytics for automated processes.
Please in case of problems check the checklist at the end of this document.

Talend-Integration

This component can be found in the palette under Business->Google
This component provides an output flow and several return values.

Parameters

Parameters to connect to Google Analytics (setup client)

Property	Content
Application Name	Not necessary, but recommended by Google. Simple provide the name of your application gathering data. ***Required***
Service Account Email	The email address of the service account. Google creates this address within the process of creating a service account. ***Required***
Key File (*.p12)	The Service Account Login works with private key file for authentication. In the process of creating a service account you download this file. ***Required***

Parameters to define the query

Property	Content
View-ID	Set here the View-ID (formally known as Profile-ID). It is a 10-digit number (fast growing number range!) ***Required!***
Start Date	All queries need always a time range (only date, not time). The value must be a String with the pattern “yyyy-MM-dd”. ***Required!***
End Date	Time range end. If you want gather data for one date, use start date as end date. The value must be a String with the pattern “yyyy-MM-dd”. ***Required!***
Dimensions	Dimensions are like group clauses. These dimensions will group the metric values. See advise for notations below. Separate multiple dimensions with a comma. ***Not required (since release 1.5)***
Metrics	Things you want to measure. Separate multiple metrics with a comma. See advise for notations below. ***Required!***
Filters	Contains all used filters as concatenated string. See advise for notation below
Sorts	Contains all sort criteria as concatenated string. Separate multiple dimensions/metrics with a comma. See advise for notation below
Segment-ID	Segments are stored filters within Google Analytics. You need the numeric ID of the segment.
Sampling Level	Google Analytics can collect the result based on sampled data. This attribute tells Google Analytics which kind of sampling should be used (in case of sampling is necessary because of the amount of data). These are the possible values: DEFAULT: It is a balance between Speed and precision FASTER: use more sampled data but the result returns faster HIGHER_PRECISION: use less sampled data and it takes longer to get the result

Deliver Totals Data Set	The API provides a totals record. This can be used to calculate percentage values or check results. This data set will be delivered (as first row) if option is checked or will be omitted if option is not checked. Date values (e.g. ga:date) remains empty (null) in the totals record.
Normalize Output	If true the component normalizes the otherwise flat record into two normalized outputs (dimensions and metrics). For every raw record with its columns for dimensions and metrics this option creates one record per raw-record and dimension / metric. E.g. if the component in the flat mode would create 3 records with 4 dimensions and 2 metrics it will create for the dimensions-flow 3 x 4 records and for the metrics flow 3 x 2 records.

Explanation for the Normalized Output

The normalized output as made for scenarios in which the job will be configured with metrics and dimensions at runtime. In this use case it is not possible to declare the appropriated schema for the flat output. The normalization creates 2 read only output schemas:

Dimensions

Column	Type	Meaning
ROW_NUM	int	The row number from the original flat result row. It identifies the records, which belongs to together.
DIMENSION_NAME	String	Name of the dimension
DIMENSION_VALUE	String	Value of the dimension

Metrics

Column	Type	Meaning
ROW_NUM	int	The row number from the original flat result row. It identifies the records, which belongs together.
METRIC_NAME	String	Name of the metric
METRIC_VALUE	Double	Value of the metric

Advice for notation

For dimensions, metrics, filters and sorts you have to use the notation from the Google Core API:
https://developers.google.com/analytics/devguides/reporting/core/dimsmets

Filters can be concatenated with OR or AND operator.
Separate filter expressions with a comma means OR
Separate filter expressions with a semicolon means AND

Filter comparison operators:

Operator	Meaning
“==”	Exact match to include
“!=”	Exact match to exclude
“=~”	Regex match to include (only for strings)
“!~”	Regex match to exclude (only for strings)
“>=”	Greater or equals than (only for numbers)
“=@”	Contains string
“!@”	Does not contains string
“>”	Greater than (only for numbers)
“<=”	Lower or equals than (only for numbers)
“<”	Lower than (only for numbers)

Building Flat Schema for Component

In the schema you need an amount of columns equals to the sum of the number of dimensions and metrics.
Columns in the schema must start at first with dimensions (if provided) and ends with metrics.
Schema column types must match to the data types of the dimensions and metrics. The schema column names can differ from the names of dimensions and metrics. Only the order and there types are important.
In Talend schema columns must follow the Java naming rules therefore avoid writing ga:xxx instead use the name without the ga: namespace prefix.
Important: For date dimensions (e.g. ga:date) you must specify the date pattern as “yyyyMMdd” if you want it as Date typed value.

Advanced Option Parameters

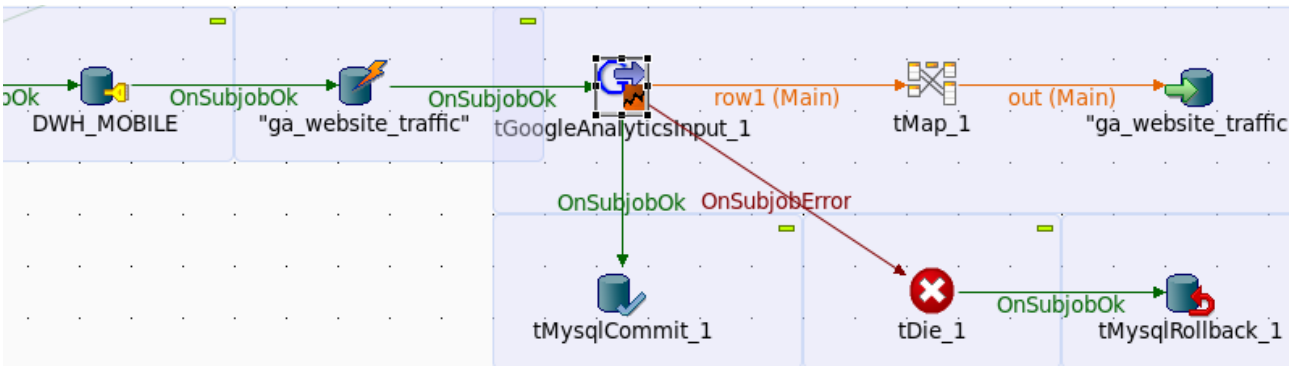
Property	Content
Timeout in s	How long should the component wait for getting the first result and fetching all result with one internal iteration
Static Time Offset (to past)	Within the process of login, the component requests an access token and use the local time stamp (because these tokens will expire after a couple of seconds) Google rejects all requests to access tokens when the request is in the future compared to the timestamp in Google servers. If you experience such kind of problems, this options let the requests appear to be more in the past (5-10s was recognized as good time shift)
Fetch Size	This is the amount of data the component fetches at once. The value is used to set the max_rows attribute. max_rows means not the absolute amount of data! The component manages setting the start index to get all data. To achieve this, the component iterates as long as the last result set are completely fetched.
Local Number Format	You can get numbers in various formats. Here you can define the locale in which format double or float values are should textual format by the API.
Reuse Client for Iterations	If you use this component in iterations it is strongly recommended to set this option. It saves time to authenticate unnecessary often and avoids problems because of max amount of connects per time range.
Distinct Name Extension	The client will be kept with an automatically created name: Talend-Name-Component name + job name. In case this is not distinct enough, you can specify an additional extension to the name.

Return values

Return value	Content
ERROR_MESSAGE	Last error message
NB_LINE	Number of delivered lines (only set if normalization is not used)
CONTAINS_SAMPLED_DATA	True if data are sampled, means not exactly calculated. This can happen if you query to many details.
SAMPLE_SIZE	The amount of datasets used for the query
SAMPLE_SPACE	The amount of available datasets for this query
TOTAL_AFFECTED_ROWS	Number of rows, which are collected by Google to calculate the result.
NB_LINE_DIMENSIONS	Number of normalized dimension records (only set if normalization is used)
NB_LINE_METRICS	Number of normalized metric records (only set if normalization is used)

Scenario 1

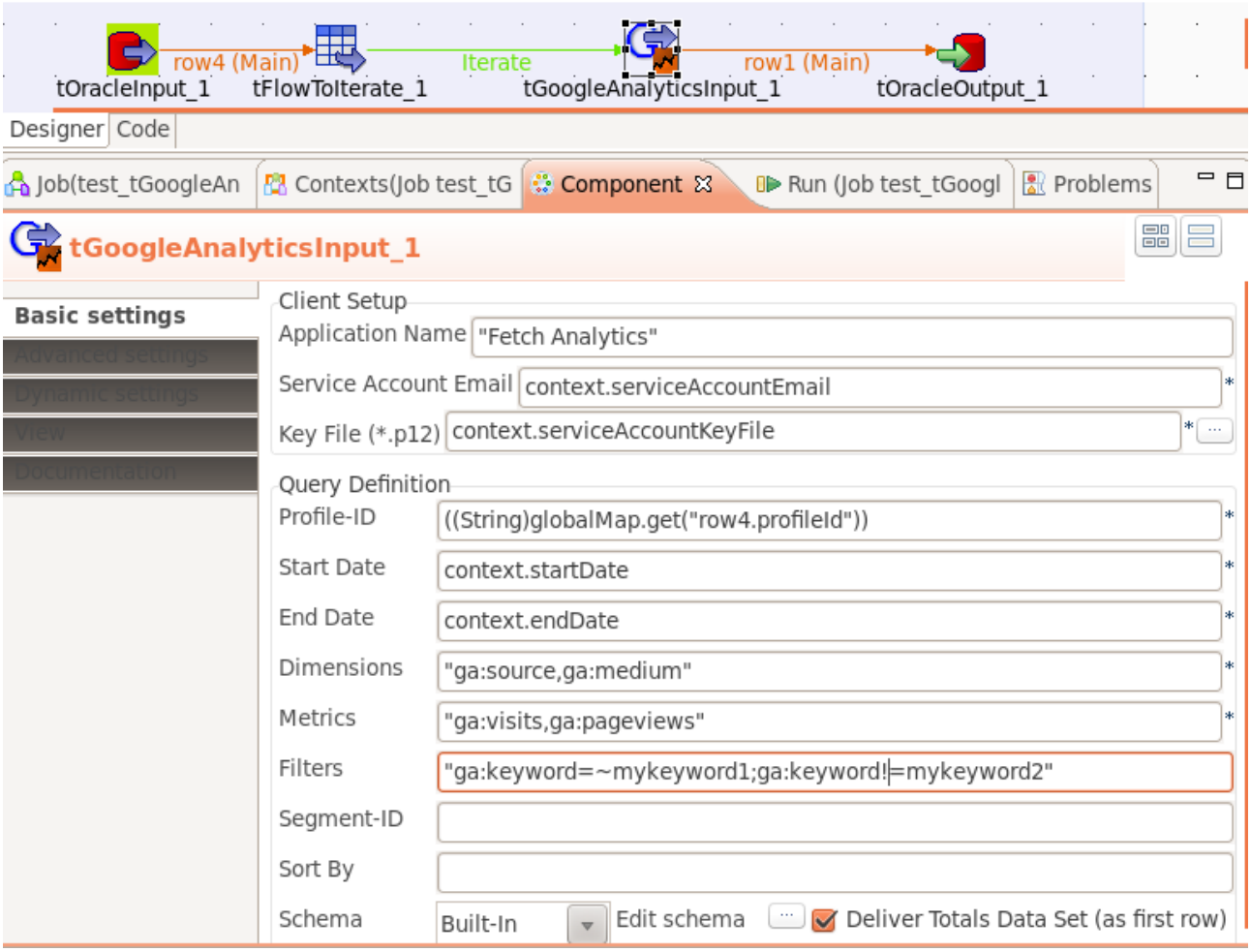
Profiles and filters are stored in a database.



In this scenario Google Analytics data are fetched in a sub job. The data will be deleted before inserting to provide restart capabilities.
This job can be used as embedded job in a surrounding job in a iteration.

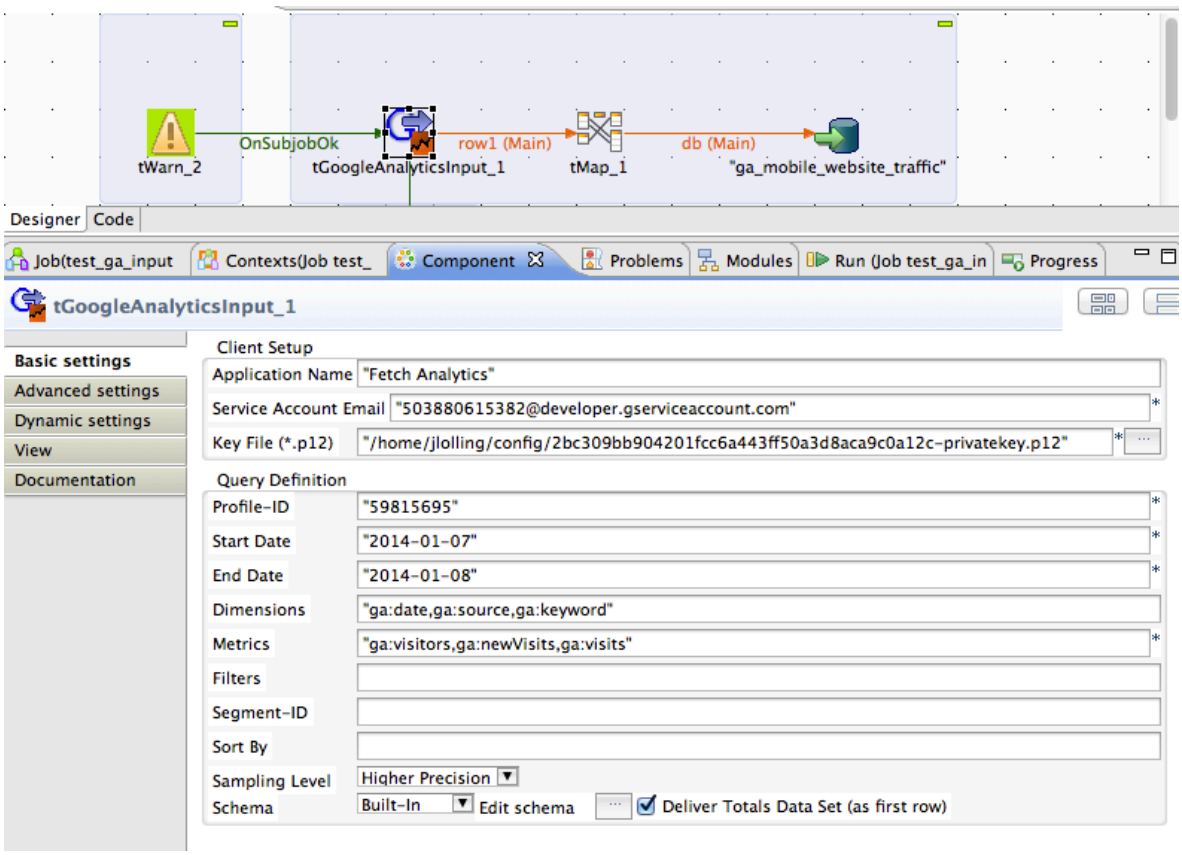
Scenario 2

Iterate through profiles stored in a database table.



Scenario 3

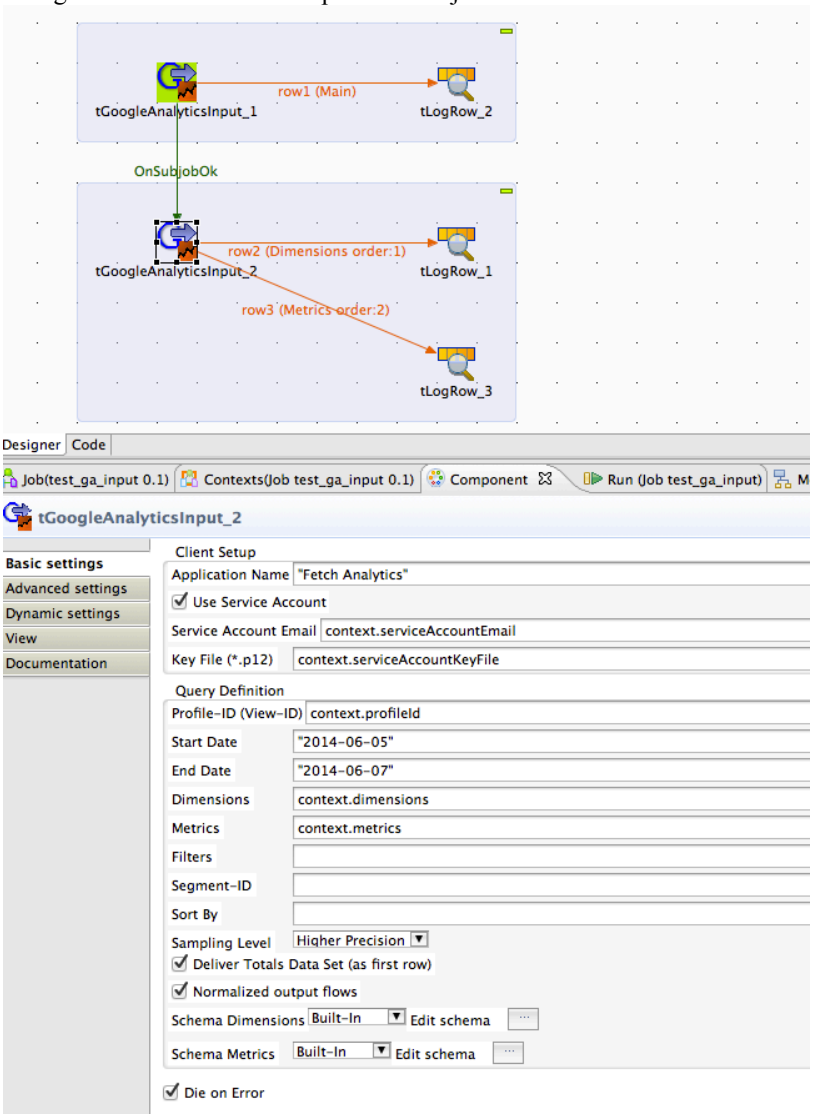
In case of the data are based on sampled data, you can control the sampling.



With the help of the new attribute Sampling Level you can control the way Google Analytics collects the necessary data. E.g. with the sampling level “Higher Precision” you can use a larger sampling size. The back draft, it will take much longer to get the result.
If there is no sampling, this will have no effect.

Scenario 4

Using flat and normalized output in a test job.



The dimensions was set to: "ga:date,ga:source,ga:keyword", The metrics was set to: "ga:visitors,ga:newVisitors"
It is not necessary to know this configuration at runtime because the component recognizes the dimensions and metrics from the result set metadata.

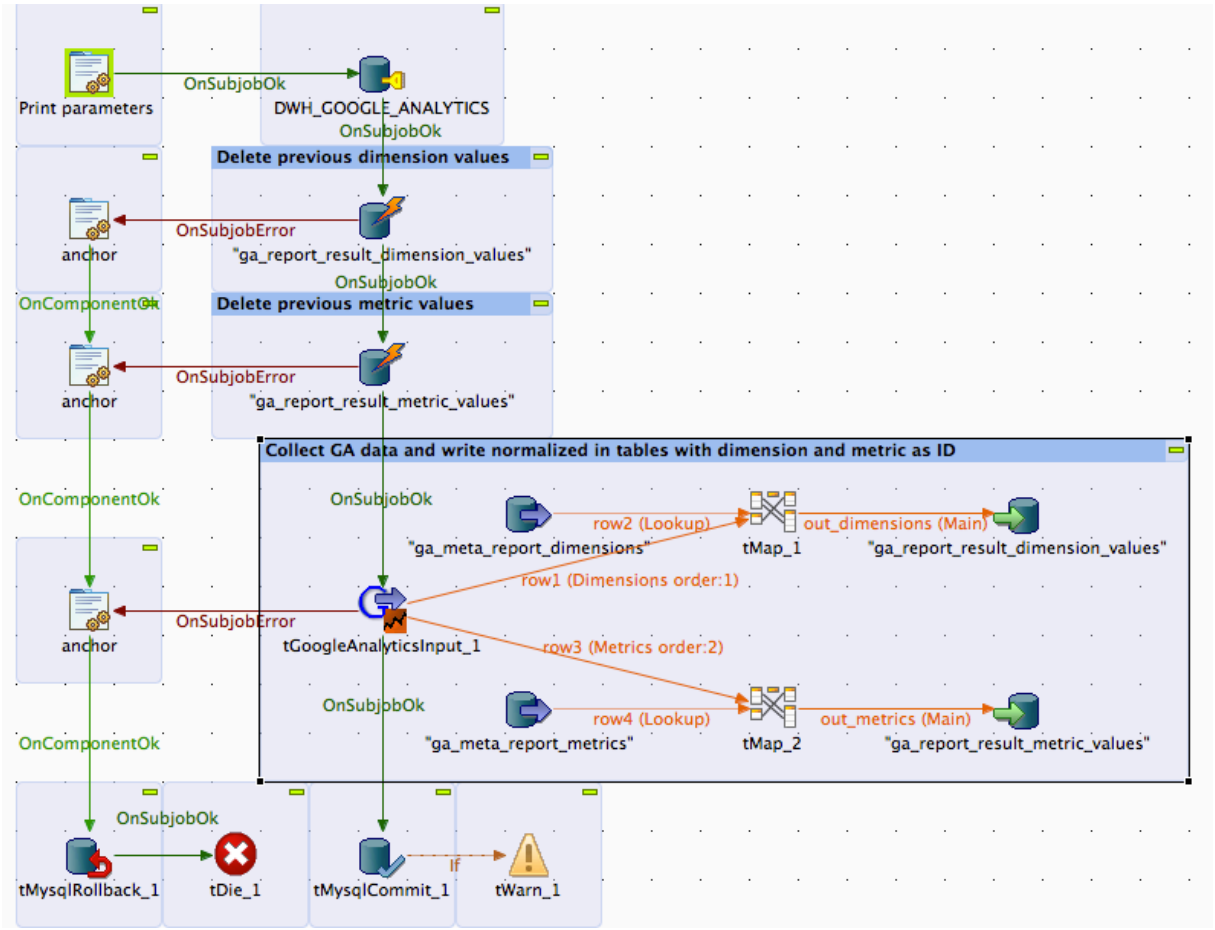
Here the outputs:

```
.--------+------+--------------+--------+---------.
|                    tLogRow_2                     |
|=-------+------+--------------+--------+--------=|
|date    |source|keyword       |visitors|newVisits|
|=-------+------+--------------+--------+--------=|
|total   |total |total         |3       |2        |
|20140605|google|(not provided)|1       |1        |
|20140606|google|(not provided)|2       |1        |
'--------+------+--------------+--------+---------'
```

```
.-------+--------------+--------------.
|              tLogRow_1              |
|=------+--------------+--------------=|
|ROW_NUM|DIMENSION_NAME|DIMENSION_VALUE|
|=------+--------------+--------------=|
|0      |ga:date       |total          |
|0      |ga:source     |total          |
|0      |ga:keyword    |total          |
|1      |ga:date       |20140605       |
|1      |ga:source     |google         |
|1      |ga:keyword    |(not provided) |
|2      |ga:date       |20140606       |
|2      |ga:source     |google         |
|2      |ga:keyword    |(not provided) |
'-------+--------------+--------------'
```

```
.-------+------------+------------.
|            tLogRow_3            |
|=------+------------+-----------=|
|ROW_NUM|METRIC_NAME |METRIC_VALUE|
|=------+------------+-----------=|
|0      |ga:visitors |3.0         |
|0      |ga:newVisits|2.0         |
|1      |ga:visitors |1.0         |
|1      |ga:newVisits|1.0         |
|2      |ga:visitors |2.0         |
|2      |ga:newVisits|1.0         |
'-------+------------+------------'
```

Next a real live scenario for using the normalized output in conjunction with the usage of the meta-data (gathered with the component tGoogleAnalyticsManagement):



This job is designed to gather the data for one day and one report (a combination of a view, dimensions, metrics and filters very much like a custom report in the Google Analytics dashboard).
This job gets the view-ID, dimensions, metrics and filters as context variables and will be called as much there are queries and dates to process.
The tMaps exchanges the dimension names and metric names with their numeric ids and adds a report-ID and the current date into the output flow for the database.
To get this job restartable everything is done within a transaction and the previous data for the report and the date will be deleted at first.
By the way, take note about the way to handle errors, this is very easy and avoid implementing the error handling multiple times. The anchors are tJava components without any code.

It is supposed to use gather the Analytics metadata to be sure you have access to all necessary data and to be able to build a star schema for the dimensions and metrics. Take a look at the component tGoogleAnalyticsManagement (today I would name it more like metadata but anyway).

Configuration checklist:

1. Is the email of the service account added to all relevant views (profiles)?
2. Is the system time of the host running the job synchronized with a NTP server?
3. Is the Google Analytics API enabled in the Google API Console?

Tip:
Check your report at first in the Google Analytics API Explorer to get an idea if the data works for you.