

# Kadiska - Network Statistic

- [Summary](#)
- [Tools: Talend](#)
  - [Detail job](#)
  - [Flow job](#)
  - [Access rights](#)
  - [Source](#)
  - [Format](#)
- [Destination](#)
  - [Location](#)
  - [Format](#)
  - [Sizing](#)
  - [Assessment](#)
- [Loading](#)
  - [1.1 Incremental Load](#)
  - [1.2 Full load](#)
  - [1.3. Reloading data](#)
  - [1.4 Plan to schedule](#)
  - [1.5 Timing](#)
- [Criticality](#)
- [Logging](#)
- [Known Error](#)

## Summary

This API is to extract the respond time of each application on each sites. However, not all Solvay sites have this program setup. Therefore, we can see only some sites only.

The API URL <https://app.kadiska.com/api/v1/query> (l\_VAR\_kadiska\_url\_query)

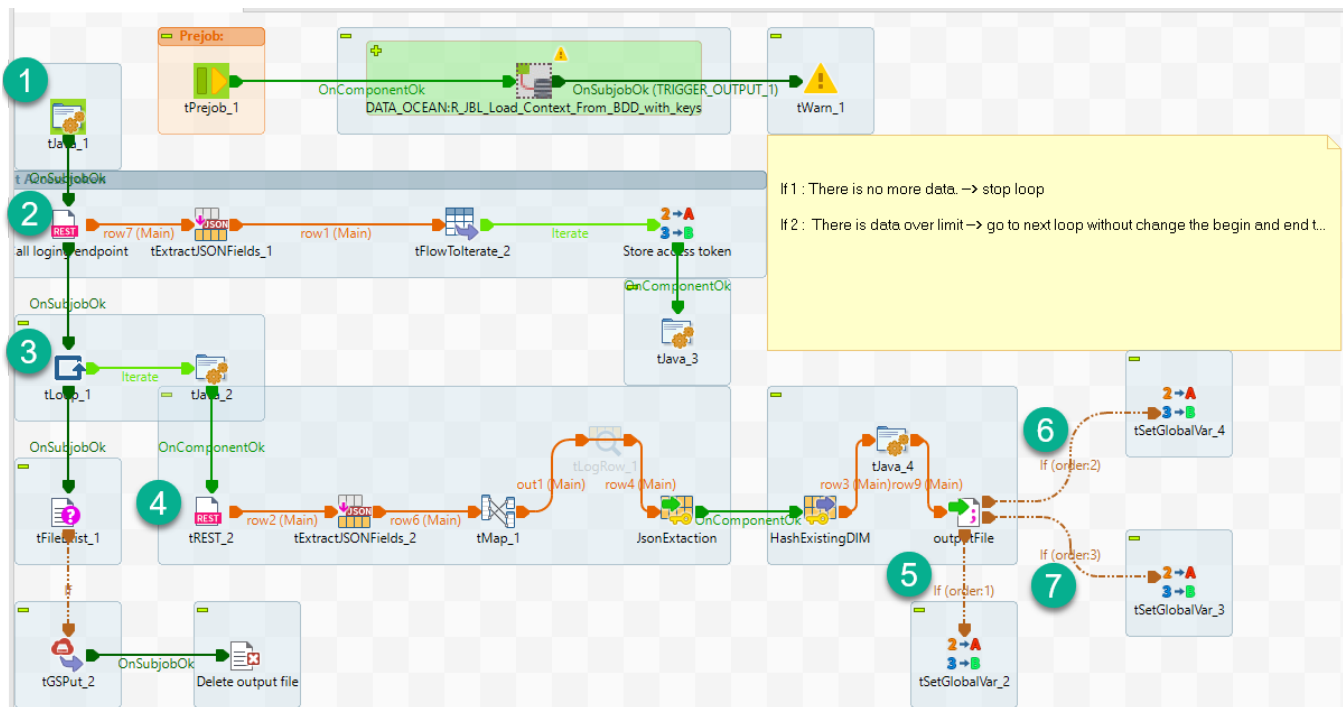
[Example of query](#) (Body)

Table	Incremental Load By	Incremental Flag	No. Fields	Talend Job	ODS Table	DM	DPL Table
<a href="#">Kadiska-rum</a>	Start time	KADISKA_RUM	83	F070_Kadiska_rum_to_ODS	ODS_KDK_0000_F001_I_H_rum	FACT_kdk_rum	V_FACT_kdk_rum

## Tools: Talend

### Detail job

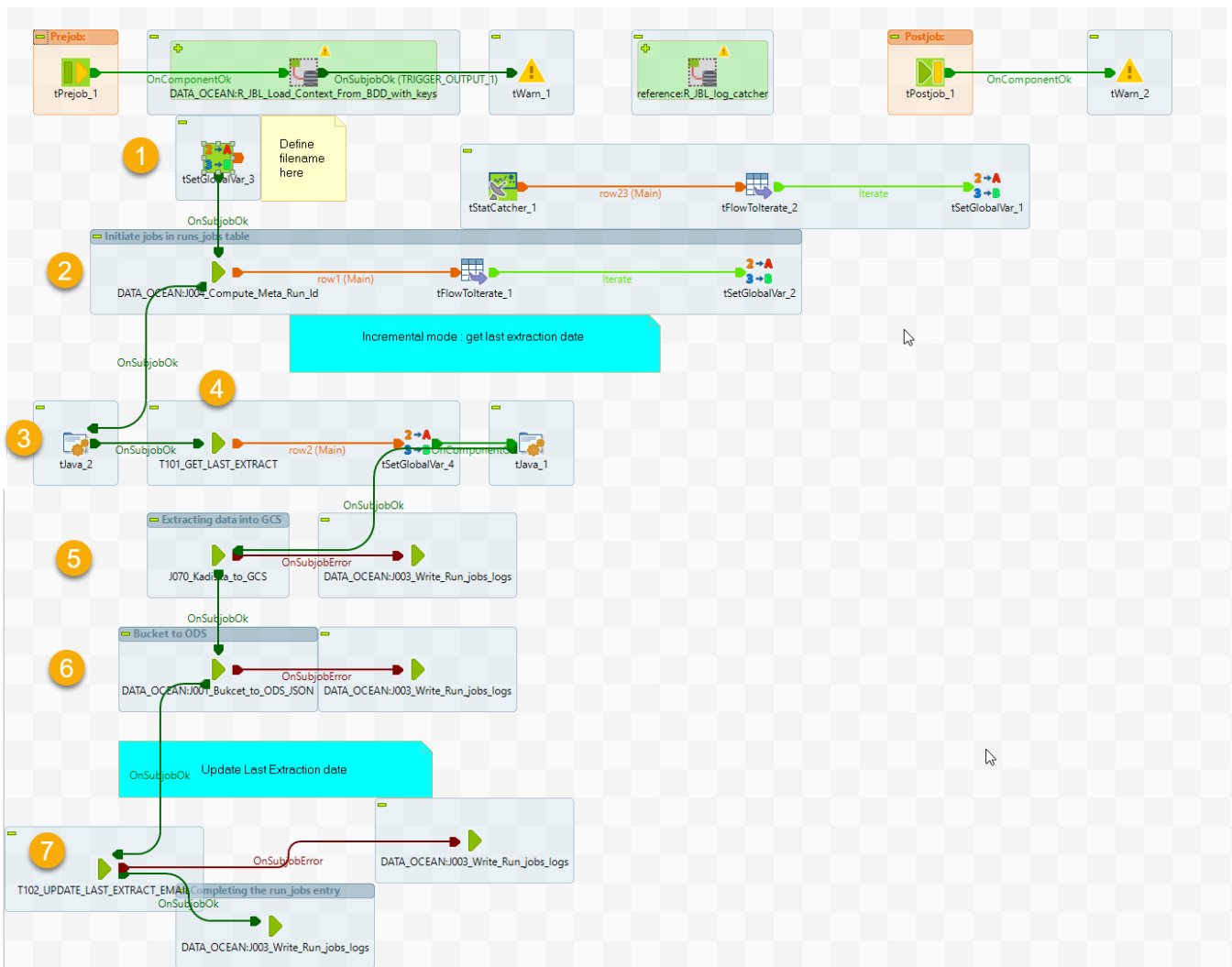
- J070\_Kadiska\_to\_GCS



1. Define the date
  - a. Time to select "Begin" to "End"
  - b. Time to stop "Stop"
  - c. Time today = "Now"
2. Login to Kadiska API on URL: <https://app.kadiska.com/api/v1/config/clients/client:5b77758db8/tokens> (l\_VAR\_kadiska\_url\_token) and keep the token as global variable
3. Start looping
4. Extract data from Kadiska API on URL: <https://app.kadiska.com/api/v1/query> (l\_VAR\_kadiska\_url\_query) controlling by following variables
5. In case [no more data] there is no data or the end time >= stop time, stop the loop
6. In case [there is data more than limit] there is data and number of line = limit, add offset + number of line
7. In case [no more data during the interval, go to next loop] there is data and number of line < limit or there is no data and End time < Stop, set offset = 0 and add Begin and End time 1 hour (interval)

## Flow job

- F070\_Kadiska\_rum\_to\_ODS



1. Define variable such as filename
2. Get meta\_run\_id
3. Define meta\_file\_name for incremental or reload which depending on I\_VAR\_kadiska\_q\_reload but the name is fixed to KADISKA\_RUM [RELOAD]
4. Get the last successfully load from table incremental\_loading
5. Call detail job
6. Load to ODS
7. Update the last successful date to incremental\_loading table

## Access rights

It is request to network team : Mohammed IDRISSEI and PRISCILA SANCHES DE BRITO-ex , Kadiska team = <https://kadiska.com/contact-us/>

Username/Password encrypt on variable g\_CNX\_KDK\_Password

## Source

<https://app.kadiska.com/api/v1/query>

## Format

JSON

## Destination

## Location

- Bucket = cs-ew1-prj-data-dm-dt-[dev]-staging/KADISKA/
- DataOean GCP = prj-data-dm-dt-[env]
- STG Table names =
  - prj-data-dm-dt-prod.STG.STG\_KDK\_0000\_0000\_F001\_I\_H\_rum
- ODS Table names =
  - prj-data-dm-dt-prod.ODS.ODS\_KDK\_0000\_F001\_I\_H\_rum
- DPL View names =
  - prj-data-dm-dt-[env].DPL.V\_FACT\_hlx\_work\_order

## Format

columnar format

## Sizing

## Assessment

## Loading

### 1.1 Incremental Load

It is control by variables:

- `L_VAR_kadiska_q_condition` : normally it is blank but it can add additional filter such as `["=", "watcher_name", ["$", "Google Calendar"]]`,
- `L_VAR_kadiska_q_interval` : during project design, we agree to have interval 1 hour = 3600000 milliseconds (should not be changed, agree on the project to keep 1 hour gap)
- `L_VAR_kadiska_q_limit` : max number of records to get from API each call (recommend 9999 which is max of API return)
- `L_VAR_kadiska_q_nloop` : max number of loop (recommend 5, increase more when need to reload in order to avoid many duplicated data)
- `L_VAR_kadiska_q_offset` : start index of the record from API query
- `L_VAR_kadiska_q_reload` : "incremental" for the incremental load, in case of reload change this variable to end time with epoch format (starting time will get from table `incremental_loading` on `meta_file_name = KADISKA_RUM_RELOAD`)
- `L_VAR_kadiska_email_flag` : "yes" when need Talend to send email to inform when the incremental load is not increasing on the `incremental_table`, which can use the loading duplicate data
- `L_VAR_kadiska_email_recipient` : list of email to get the email

### 1.2 Full load

N/A

### 1.3. Reloading data

It is control by variables:

- `L_VAR_kadiska_q_limit`: The number of records to get from API each time
- `L_VAR_kadiska_q_nloop`: the number of loop for each load. (the number of limit x nloop will be max record on the file to load each time to BQ)
- `L_VAR_kadiska_q_offset`: The number of record index from API. It starts with 0. Will change to another number only specific reload case
- `L_VAR_kadiska_q_reload`: If it is NOT "incremental", it will the full load with condition on this parameter such as [1724371200000]. It will be the end time ([epoch format](#)). The start date will always get from `incremental_loading` table with `meta_file_name = KADISKA_RUM_RELOAD`

#### How to:

1. Stop schedule job
2. Change the start date that is required to reload on

```
UPDATE STG.incremental_loading
```

```
SET meta_last_process_date = '2021-12-22 00:00:00' --the date that want to reload
```

where `meta_file_name = 'KADISKA_RUM_RELOAD'` -- identify the object that want to reload but the name to change must be `_RELOAD`. Without `_RELOAD` will use for incremental load only.

3. Change context I\_VAR\_kadiska\_q\_reload to the end date with epoch format
4. Recheck the variables: limit, nloop, offset
5. Run the job to load until KADISKA\_RUM\_RELOAD is reach the end time that we enter on I\_VAR\_kadiska\_q\_reload. Meaning reload is complete
6. Change the context I\_VAR\_kadiska\_q\_reload to be "incremental"
7. Reschedule the job

## 1.4 Plan to schedule

Every hour

## 1.5 Timing

5 minutes

## Criticality

Low

## Logging

1. Check the last loading

```
select * from STG.incremental_loading
where meta_file_name like '%KADISKA_RUM'
order by meta_file_name
```

meta_file_name	meta_source_system	meta_last_process_date
KADISKA_RUM	KDK	2024-08-23 07:00:00 UTC

2. Check the loading records / error

```
select job.job_name , job.meta_start_date , logs.meta_run_id , logs.meta_source_system , logs.meta_step , logs.meta_status , logs.meta_num_lines , logs
.meta_error_lines from STG.log_tables logs join STG.run_jobs job on logs.meta_run_id = job.meta_run_id
where logs.meta_run_id in ( SELECT meta_run_id FROM STG.run_jobs order by meta_start_date desc limit 100 )
and job_name like '%Kadiska%'
and meta_start_date > DATE_SUB ( CURRENT_TIMESTAMP () , INTERVAL 1 DAY )
order by meta_start_date desc
```

job_name	meta_start_date	meta_run_id	meta_source_system	meta_step	meta_status	meta_num_lines	meta_error_lines
F070_Kadiska_rum_to_ODS	2024-08-23 07:03:54.538313 ...	dd72673a-00ba-49ac-8f3b-47f...	STG_KDK_0000_0000_F001_L...	Staging to ODS	OK	139	0
F070_Kadiska_rum_to_ODS	2024-08-23 07:03:54.538313 ...	dd72673a-00ba-49ac-8f3b-47f...	KDK_DT_0000_0000_F001_202...	Bucket to Staging	OK	139	0
F070_Kadiska_rum_to_ODS	2024-08-23 06:03:16.352499 ...	0c1105df-1a5c-4b16-93be-0f9f...	KDK_DT_0000_0000_F001_202...	Bucket to Staging	OK	68	0
F070_Kadiska_rum_to_ODS	2024-08-23 06:03:16.352499 ...	0c1105df-1a5c-4b16-93be-0f9f...	STG_KDK_0000_0000_F001_L...	Staging to ODS	OK	68	0
F070_Kadiska_rum_to_ODS	2024-08-23 05:02:43.847255 ...	0fdb4b3-5028-4a31-a190-ebb...	STG_KDK_0000_0000_F001_L...	Staging to ODS	OK	56	0
F070_Kadiska_rum_to_ODS	2024-08-23 05:02:43.847255 ...	0fdb4b3-5028-4a31-a190-ebb...	KDK_DT_0000_0000_F001_202...	Bucket to Staging	OK	56	0

3. Check duplicated data / missing data

```
select meta_business_date , meta_run_id , time_begin , count (*) from `ODS.ODS_KDK_0000_F001_I_H_rum`
group by meta_business_date , meta_run_id , time_begin
order by meta_business_date desc , time_begin desc
```

## Known Error

1. If the last nloop times, not cover all records by limit, it will move to next hour (happen only loading > 1 hour gap). Check on Logging script 3 if the number of records = limit, it may have more data
2. When there is no data loading and it API return on postman like picture below

```
1 {
2   "detail": [
3     {
4       "msg": "Too Many Requests"
5     }
6   ]
7 }
```

The log in TMC will be

Error tExtractJSONFields\_2 - Missing property in path \$('[data]

It is error on Kadiska about the limit of run times / day.

For non available data - Contact Solvay Network team - [mohamed.idrissi@solvay.com](mailto:mohamed.idrissi@solvay.com), [priscila.sanchesdebrito-ext@solvay.com](mailto:priscila.sanchesdebrito-ext@solvay.com) , [vinicius.cosendey@solvay.com](mailto:vinicius.cosendey@solvay.com)

**Can also contact Kadiska team directly** <https://kadiska.com/contact-us/> (needs account created by Vinicius) adding in the ticket as cc Network team (Mohammed, Priscila and Vinicius)

- to create a request /case > **Process\_ How to open Kadiska Cases/Incidents**
- Other contacts: email at [support@netskope.com](mailto:support@netskope.com), or via telephone using one of our regional support numbers:

US: 1-800-685-2098

UK: 44-8455280141

Australia: 1-800-505-486

Europe: 44-8455280141

Singapore: 80-0130-2191

India: 00080-0100-4400