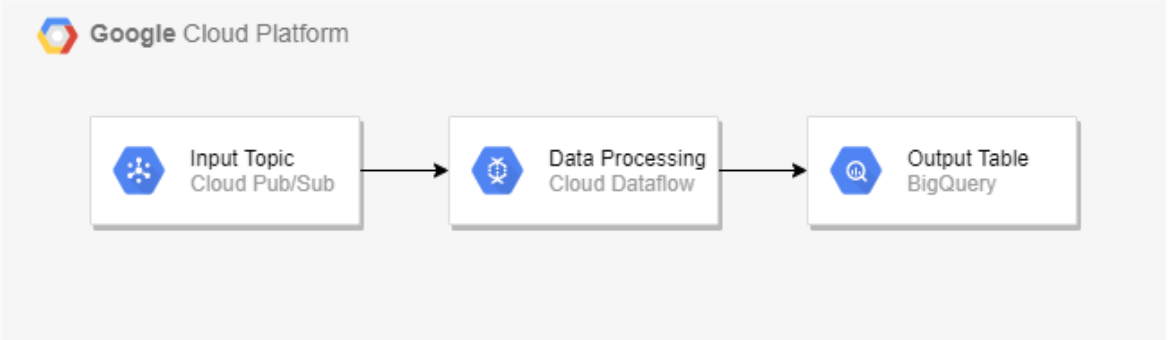
# Use case 1 - Pub/Sub to BigQuery with Dataflow

The goal of this use case is to implement a Dataflow job that reads a message from a Pub/Sub topic and writes its content to a BigQuery table.




**Message body:**

```
{"id": N, "name": "name1", "surname": "name2"}
```

Fields description:
id: integer;
name: string;
surname:string

The destination table on BigQuery has 3 fields and is called "**account**"(the table already exists). The fields have the same names and types as the Pub/Sub message above. All fields are nullable.

**Additional requirements:**

- Make the Pub/Sub topics used and the output BigQuery table name parametric (hint: use the option object of Apache Beam);
- Handle malformed messages (store them in a dlq topic) (hint: use the side output object of Apache Beam)

**Gradle command to run Dataflow job on GCP:**

```
LAB_ID="<N>"
```

```
gradle execute -DmainClass=<main_class> -Dexec.args="<custom_options_pipeline> --
runner=DataflowRunner --project=nttdata-c4e-bde --jobName=usecase1-labid-$LAB_ID --
region=europe-west4 --serviceAccount=c4e-uc1-sa-$LAB_ID@nttdata-c4e-
bde.iam.gserviceaccount.com --maxNumWorkers=1 --workerMachineType=n1-standard-1 --
gcpTempLocation=gs://c4e-uc1-dataflow-temp-$LAB_ID/temp --stagingLocation=gs://c4e-uc1-
dataflow-temp-$LAB_ID/staging --subnetwork=regions/europe-west4/subnetworks/subnet-
uc1-$LAB_ID --streaming"
```

**GCP resources to use:**

- Input Pub/Sub Topic: uc1-input-topic-<N>
- Input Pub/Sub Subscription: uc1-input-topic-sub-<N>
- Output BigQuery table: uc1_<N>.account
- Dlq Pub/Sub Topic: uc1-dlq-topic-<N>
- Dlq Pub/Sub Subscription: uc1-dlq-topic-sub-<N>
- Dataflow Job created bu the command: usecase1-labid-<N>

where <N> is the lab_id assigned to you

**Steps:**

1. From the GCP Cloud Shell, log in to impersonate the Dataflow job service account. Command:
   **gcloud auth application-default login**
2. Create a new repository on GitHub.com
3. Write Java code to implement an Apache Beam pipeline with the above requirements
4. Push the code on the repository created at the point 1
5. Clone/Pull the code from the repository to the cloud shell on GCP
6. Run the Dataflow job with the Gradle command shown above by replacing the following placeholders::
   - <main_class>: main class of the Java project;
   - <N> : the lab_id assigned to you;
   - <custom_options_pipeline>: set the custom job options defined in the code (for example the name of the topics used and the output table).
7. Test the pipeline by sending pub/sub messages on your topic and verify that the data is written correctly in the output table or in the dlq topic according to the message given in input (use the GCP UI).

**Useful links:**

- Apache Beam Programming Guide
- Class PubsubIO
- Class BigQueryIO

- How to impersonate a Service Account
- Dataflow job standard parameters
- How to publish a Pub/Sub message using GCP UI