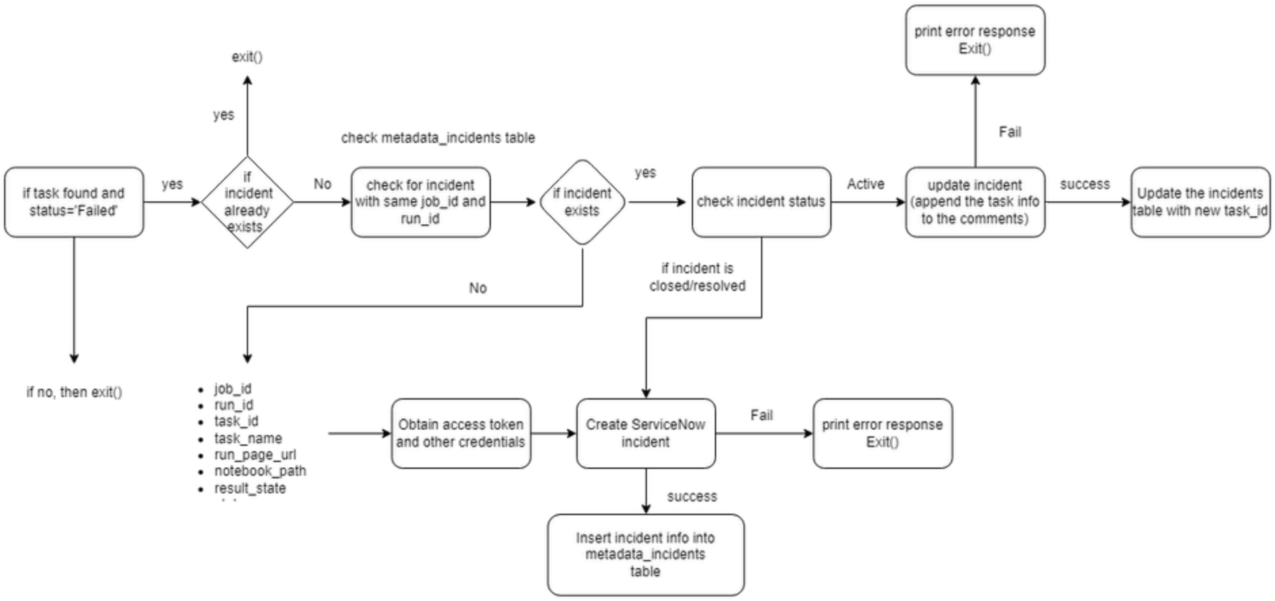
EDW_ServiceNow_Incident_Ticket_Generation

1. Overview

The idea is to have only a single incident per job run. If there is a failed task in the audit metadata table for which no incident exists, we go ahead and check if there is any incident created in the incident table with the same job_id and run_id. This means these two tasks are from the same workflow, and job run but different tasks.

If an incident exists, then we check if the incident is still open. If so, then we go ahead and update the existing incident with the new task info in the activity log(comments) section. Refer to the flowchart below for detailed information.



exit()
yes
check metadata_incidents table
print error response Exit()
Fail
if task found and status='Failed'
yes
if incident already exists
No
check for incident with same job_id and run_id
if incident exists
yes
check incident status
Active
update incident (append the task info to the comments)
success
Update the incidents table with new task_id
No
if incident is closed/resolved
if no, then exit()
job_id
run_id
task_id
task_name
run_page_url
notebook_path
result_state
Obtain access token and other credentials
Create ServiceNow incident
Fail
print error response Exit()
success
Insert incident info into metadata_incidents table

2. Obtaining data from 'workflow_integration_parameters' table

The client credentials, other info required to create ServiceNow incident, URL endpoints specific to environment (DEV, PROD, QA) are stored in the 'workflow_integration_parameters' table.

Note that the secret name and scope name is specified in the table. Both scope name and secret name are required to retrieve secret value from the databricks secrets

dbutils.secrets.get(scope_name,secret_name).

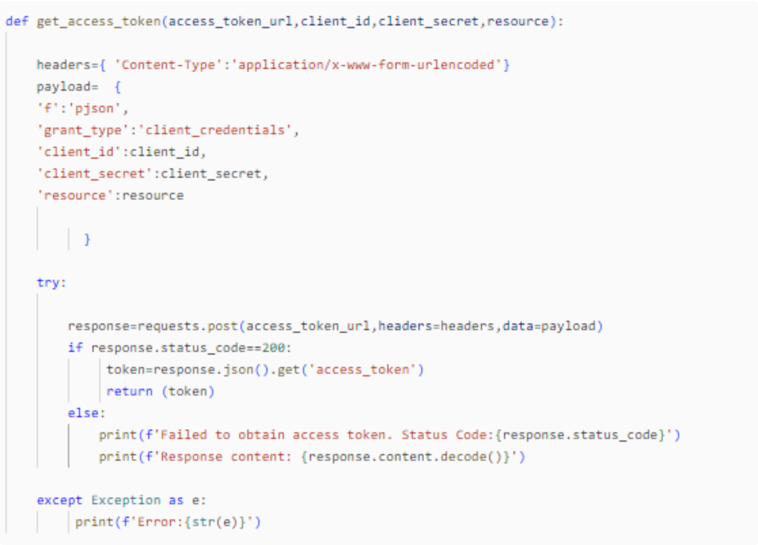
3. Retrieve failed tasks from metadata table

retrieve the info of all the failed tasks from audit metadata table and create a ServiceNow incident for each task if the incident does not exist in 'metadata_incidents' table

select JOB_ID, TASK_NAME, RUN_ID, JOB_STATUS, EXCEPTION from table_name

The information required to population description fields can be directly retrieved from this table.

4. Obtaining Access Token to use the ServiceNowAPI Services



```
def get_access_token(access_token_url,client_id,client_secret,resource):

    headers={ 'Content-Type':'application/x-www-form-urlencoded'}
    payload=  {
    'f':'pjson',
    'grant_type':'client_credentials',
    'client_id':client_id,
    'client_secret':client_secret,
    'resource':resource

          }

    try:

        response=requests.post(access_token_url,headers=headers,data=payload)
        if response.status_code==200:
             token=response.json().get('access_token')
             return (token)
        else:
            print(f'Failed to obtain access token. Status Code:{response.status_code}')
            print(f'Response content: {response.content.decode()}')

    except Exception as e:
         print(f'Error:{str(e)}')
```

5. Creating ServiceNow incident

creating incident with necessary information (urgency, impact, cmdb_ci, caller_id, location, assignment_group, short_description and description)

HTTP Verb: POST

URL: https://xxxxxxxxxxxxxxxxxxxx/apg-001-servicenow/v1/now/table/incident/

Header Key/Value Pairs:

- Authorization: "Bearer access-token"
- Accept: application/json

Body (raw json)

{
    "incident_object”: {

```
    "urgency":
    "impact":
    "cmdb_ci":
    "caller_id":
    "location":
   "assignment_group":

            "short_description":"test",
            "description":"test purpose"
   }
}
```

If the incident is successfully created, you will see the http code 200.

6. Update existing incident if task has same job_id and run_id

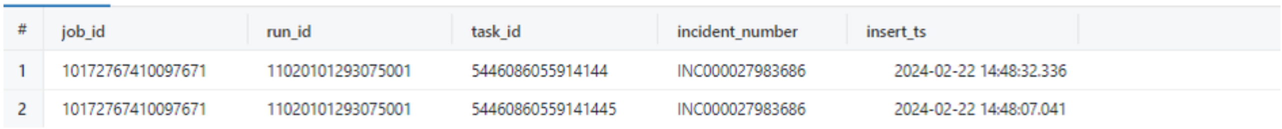
If the incident is not created for a particular task in the metadata table, then we check if an incident has been created for any other task with the same job_id and run_id.

If an incident is found, then retrieve the status of that incident to check if it is closed/resolved or still open. If the incident is still active, then update the task info in the activity (comments) field as shown below.

7. Insert into 'metadata_incidents' table

If the incident is successfully created or updated, the info(job_id, run_id, task_id, incident_number, insert_ts) is logged into the 'metadata_incidents' table.

For both records two records, the incident number is same since they are from the same workflow and job run. As said earlier, the idea is to have one single incident per job run.



#	job_id	run_id	task_id	incident_number	insert_ts
1	10172767410097671	11020101293075001	5446086055914144	INC000027983686	2024-02-22 14:48:32.336
2	10172767410097671	11020101293075001	54460860559141445	INC000027983686	2024-02-22 14:48:07.041