Data Cleanup Implementation Guide — OpenSearch & Neptune

1. Objective

Over time, both **OpenSearch** and **Neptune** accumulate old, duplicate, or unused records.

These stale entries slow queries, inflate costs, and affect downstream reports.

This document provides a **developer-centric, step-by-step guide** to safely identify and remove stale data using **AWS Glue**, **SageMaker**, or **Lambda**—with automated scheduling and rollback.

2. Cleanup Overview

System	Purpose	Tools Used	Frequency
OpenSearch	Search indexes for dashboards / analytics	Python (Glue, SageMaker, CLI)	Weekly
Amazon Neptune	Graph DB (relationships & hierarchies)	Python (SageMaker / Glue)	Monthly

- Both jobs follow a consistent 4-phase approach:

Steps	Description
Detect	Identify stale, orphaned, or deleted data
Validate	Cross-check with source systems
Delete	Perform soft + hard deletes safely
Verify & Log	Log summary, audit results to S3

3. Prerequisites (Assumptions)

1. **Retention Rules**
    - OpenSearch: keep last **90 days**
    - Neptune: keep last **180 days**
2. **S3 Buckets**
    - s3://<bucket-name>/ — cleanup reports
    - s3://<stg bucket name>/ — ID exports / dry runs
    - s3://<bucket for opensearch snapshots>/ — OpenSearch backup snapshots
3. **IAM Roles**
    - Allow s3:*, cloudwatch:*, and respective service permissions:
        - OpenSearch: es:*
        - Neptune: neptune-db:*
4. **VPC Access**
    - Glue & SageMaker must be in the same VPC/subnets as OpenSearch & Neptune.
5. **Repo Layout**

```
infra/data_cleanup/
  config/
    cleanup_opensearch.json
    cleanup_neptune.json
  scripts/
    os_cleanup.py
    os_detect_export_ids.py
    neptune_cleanup_sagemaker.py
    neptune_cleanup_glue.py
```

4. OpenSearch Cleanup

4.1 (Sample) Configuration

config/cleanup_opensearch.json

```
{
  "host": "search-prod-opensearch.xxxxx.us-east-1.es.amazonaws.com",
  "port": 443,
  "index": "transactions",
  "auth_type": "basic",
  "username": "admin",
  "password": "REDACTED",
  "retention_days": 90,
```

```
  "snapshot_repo": "os-snapshots",
  "snapshot_bucket": "opensearch-snapshots",
  "report_bucket": "data-cleanup-reports",
  "report_prefix": "opensearch/"
}
```

4.2 One-Time Setup

Register snapshot repository (Kibana or CLI):

```
curl -X PUT "https://<DOMAIN>/_snapshot/os-snapshots" \
 -u admin:REDACTED \
 -H 'Content-Type: application/json' \
 -d '{ "type": "s3", "settings": { "bucket": "opensearch-snapshots" } }'
```

4.3 Step-by-Step Execution

Step 1: Dry Run — Count Stale Docs

```
curl -X GET "https://<DOMAIN>/<INDEX>/_count" \
 -u admin:REDACTED \
 -H 'Content-Type: application/json' \
 -d '{ "query": { "range": { "last_updated": { "lt": "now-90d/d" } } }
}'
```

Step 2: Export IDs (Optional)

Run in **Glue Python Shell** or **SageMaker**:

```
from opensearchpy import OpenSearch
import json, boto3
from datetime import datetime, timedelta

cfg = json.load(open('config/cleanup_opensearch.json'))
client = OpenSearch(
    hosts=[{'host': cfg['host'], 'port': cfg['port']}],
    http_auth=(cfg['username'], cfg['password']),
    use_ssl=True, verify_certs=True
)

cutoff = (datetime.utcnow() -
timedelta(days=cfg['retention_days'])).strftime('%Y-%m-%d')
query = {"query": {"range": {"last_updated": {"lt": cutoff}}}, "size":
10000}
scroll = client.search(index=cfg['index'], body=query, scroll='2m')
s3 = boto3.client('s3')
ids = [hit['_id'] for hit in scroll['hits']['hits']]
s3.put_object(
    Bucket=cfg['report_bucket'],
    Key=f"{cfg['report_prefix']}ids_{cfg['index']}.json",
    Body=json.dumps(ids)
)
print(f"Exported {len(ids)} stale IDs.")
```

Step 3: Soft Delete

```
curl -X POST "https://<DOMAIN>/<INDEX>/_update_by_query" \
 -u admin:REDACTED \
 -H 'Content-Type: application/json' \
 -d '{ "script": { "source": "ctx._source.is_deleted = true" },
       "query": { "range": { "last_updated": { "lt": "now-90d/d" } } } }
}'
```

Step 4: Hard Delete

```
curl -X POST "https://<DOMAIN>/<INDEX>/_delete_by_query" \
 -u admin:REDACTED \
 -H 'Content-Type: application/json' \
 -d '{ "query": { "term": { "is_deleted": true } } }'
```

4.4 Glue/SageMaker Job — Automated Cleanup

scripts/os_cleanup.py

```
from opensearchpy import OpenSearch
from datetime import datetime, timedelta
import json, boto3

cfg = json.load(open('config/cleanup_opensearch.json'))
client = OpenSearch(
    hosts=[{'host': cfg['host'], 'port': cfg['port']}],
    http_auth=(cfg['username'], cfg['password']),
    use_ssl=True, verify_certs=True
)

cutoff = (datetime.utcnow() -
timedelta(days=cfg['retention_days'])).strftime('%Y-%m-%d')
count = client.count(index=cfg['index'], body={"query": {"range":
{"last_updated": {"lt": cutoff}}}})['count']

client.update_by_query(index=cfg['index'], body={
  "script": {"source": "ctx._source.is_deleted = true"},
  "query": {"range": {"last_updated": {"lt": cutoff}}}
}, conflicts="proceed")

resp = client.delete_by_query(index=cfg['index'], body={"query":
{"term": {"is_deleted": True}}}, conflicts="proceed")

report = {
  "system": "OpenSearch",
  "index": cfg['index'],
  "cutoff": cutoff,
  "soft_marked_count": count,
  "hard_deleted": resp.get('deleted', 0),
  "ts": datetime.utcnow().isoformat()
}
boto3.client('s3').put_object(
  Bucket=cfg['report_bucket'],

Key=f"{cfg['report_prefix']}report_{cfg['index']}_{datetime.utcnow().str
ftime('%Y%m%d_%H%M%S')}.json",
  Body=json.dumps(report)
)
```

5. Neptune Cleanup

5.1 Configuration

config/cleanup_neptune.json

```
{
  "endpoint_wss":
"wss://my-neptune.cluster-xyz.us-east-1.neptune.amazonaws.com:8182/greml
in",
  "retention_days": 180,
  "batch_size": 500,
  "report_bucket": "data-cleanup-reports",
  "report_prefix": "neptune/",
  "staging_bucket": "data-cleanup-staging"
}
```

5.2 SageMaker Cleanup Script

scripts/neptune_cleanup_sagemaker.py

```
from gremlin_python.driver import client
from datetime import datetime, timedelta
import json, boto3, time

cfg = json.load(open('config/cleanup_neptune.json'))
g = client.Client(cfg['endpoint_wss'], 'g')
cutoff = (datetime.utcnow() -
timedelta(days=cfg['retention_days'])).strftime('%Y-%m-%d')

# Step 1: Detect
count_q = f"g.V().has('lastActive', lt('{cutoff}')).count()"
inactive_count = g.submit(count_q).all().result()[0]
print("Inactive vertices:", inactive_count)

# Step 2: Soft delete
soft_q = f"g.V().has('lastActive',
lt('{cutoff}')).property('status','deleted')"
g.submit(soft_q).all().result()

# Step 3: Hard delete in batches
ids_q = "g.V().has('status','deleted').id()"
ids = g.submit(ids_q).all().result()
for i in range(0, len(ids), cfg['batch_size']):
    batch = ids[i:i+cfg['batch_size']]
    drop_q = "g.V(" + ",".join([f\"'{x}'\" for x in batch]) + ").drop()"
    g.submit(drop_q).all().result()
    time.sleep(0.3)
print("Cleanup done.")

# Step 4: Report
boto3.client('s3').put_object(
    Bucket=cfg['report_bucket'],

Key=f"{cfg['report_prefix']}report_{datetime.utcnow().strftime('%Y%m%d_%
H%M%S')}.json",
    Body=json.dumps({
        "system": "Neptune",
        "cutoff": cutoff,
        "soft_marked": inactive_count,
        "hard_deleted": len(ids),
        "ts": datetime.utcnow().isoformat()
    })
)
```

5.3 Glue Cleanup (Automated Monthly)

scripts/neptune_cleanup_glue.py

```
from gremlin_python.driver import client
from datetime import datetime, timedelta
import boto3, json, time

cfg = json.load(open('config/cleanup_neptune.json'))
g = client.Client(cfg['endpoint_wss'], 'g')
s3 = boto3.client('s3')

cutoff = (datetime.utcnow() -
timedelta(days=cfg['retention_days'])).strftime('%Y-%m-%d')
inactive = g.submit(f"g.V().has('lastActive',
lt('{cutoff}')).count()").all().result()[0]
g.submit(f"g.V().has('lastActive',
lt('{cutoff}')).property('status','deleted')").all().result()

deleted_total = 0
while True:
    ids =
g.submit(f"g.V().has('status','deleted').limit({cfg['batch_size']}).id()
").all().result()
    if not ids:
        break
    q = "g.V(" + ",".join([f\"'{x}'\" for x in ids]) + ").drop()"
    g.submit(q).all().result()
    deleted_total += len(ids)
    time.sleep(0.2)

s3.put_object(
  Bucket=cfg['report_bucket'],

Key=f"{cfg['report_prefix']}report_{datetime.utcnow().strftime('%Y%m%d_%
H%M%S')}.json",
  Body=json.dumps({
    "system": "Neptune",
    "cutoff": cutoff,
    "soft_marked": inactive,
    "hard_deleted": deleted_total,
    "ts": datetime.utcnow().isoformat()
  })
)
```

---

6. Scheduling

Platform	Task	Frequency	Trigger
Glue	OpenSearch cleanup	Weekly	cron(0 21 ? * SUN *)
Glue	Neptune cleanup	Monthly	cron(0 21 1 * ? *)
SageMaker	Manual review	On-demand	Developer-triggered
Lambda + EventBridge	Small daily deletes	Optional	Auto trigger

---

7. Verification & Rollback

Step	Validation
OpenSearch	GET /_count and index health check
Neptune	g.V().count() and g.E().count() post cleanup
Rollback	Restore snapshot (OpenSearch/Neptune) to new cluster/index
Reports	All cleanup runs log to s3://data-cleanup-reports/...

Requirements.txt file

```
boto3>=1.28.0
botocore>=1.31.0
opensearch-py>=2.4.2
gremlinpython>=3.6.4
aiobotocore>=2.12.0
requests>=2.31.0
urllib3>=2.0.4
python-dateutil>=2.9.0
pandas>=2.1.0
pyyaml>=6.0.2
tqdm>=4.66.1
retrying>=1.3.4
ipykernel>=6.25.0
jupyterlab>=4.0.5
rich>=13.6.0
click>=8.1.7
```