In [38]:

```
import pandas as pd
import xarray as xr
import zarr
from datetime import datetime
import os

# Constants
GHCND_URL_TEMPLATE = "https://www.ncei.noaa.gov/data/global-historical
WBAN_CODES = ["14739", "23169", "94846"]
ZARR_ARCHIVE_PATH = "ghcnd_archive.zarr"

def fetch_data(wban_code):
    url = GHCND_URL_TEMPLATE.format(wban_code)
    return pd.read_csv(url, low_memory=False)

def transform_data(df):
    df = df[['DATE', 'PRCP', 'TMAX', 'TMIN']].copy()
    df.columns = ['time', 'precp', 'tmax', 'tmin']
    df['time'] = pd.to_datetime(df['time'])
    df.set_index('time', inplace=True)
    return df



def build_ghcnd_archive(wban_codes):
    datasets = []
    for code in wban_codes:
        df = fetch_data(code)
        df = transform_data(df)
        ds = xr.Dataset.from_dataframe(df)
        ds = ds.expand_dims({"ghcn_id": [code]})
        datasets.append(ds)

    combined_ds = xr.concat(datasets, dim="ghcn_id")
    combined_ds.to_zarr(ZARR_ARCHIVE_PATH, mode="w", consolidated=True
    print("Archived successfully!")

def update_ghcnd_archive(wban_codes):
    archive = xr.open_zarr(ZARR_ARCHIVE_PATH)
    datasets = []
    for code in wban_codes:
        df = fetch_data(code)
        df = transform_data(df)
        ds = xr.Dataset.from_dataframe(df)
        ds = ds.expand_dims({"ghcn_id": [code]})
        datasets.append(ds)

    combined_ds = xr.concat(datasets, dim="ghcn_id")
    combined_ds.to_zarr(ZARR_ARCHIVE_PATH, mode="a", append_dim="time"
    print("Archive updated successfully!")
```

```
Archive built successfully
Archive updated successfully
```

In [39]:
```
# Build the archive from scratch
build_ghcnd_archive(WBAN_CODES)
```

```
Archive built successfully
```

In [40]:
```
# Simulate a daily update
update_ghcnd_archive(WBAN_CODES)
```

```
Archive updated successfully
```

In [41]:
```
archive = xr.open_zarr("ghcnd_archive.zarr")
print(archive)
```

```
<xarray.Dataset>
Dimensions:  (ghcn_id: 3, time: 64706)
Coordinates:
  * ghcn_id  (ghcn_id) <U5 '14739' '23169' '94846'
  * time     (time) datetime64[ns] 1936-01-01 1936-01-02 ... 2024-07-
29
Data variables:
    precp    (ghcn_id, time) float64 dask.array<chunksize=(2, 16177),
meta=np.ndarray>
    tmax     (ghcn_id, time) float64 dask.array<chunksize=(2, 16177),
meta=np.ndarray>
    tmin     (ghcn_id, time) float64 dask.array<chunksize=(2, 16177),
meta=np.ndarray>
```

### How would you orchestrate this system to run at scale?

I would use a workflow orchestration tool to schedule daily updates, such as <strong>Databricks Workflow</strong> or any other tools based on the team's preference. The tool would manage the ETL process, schedule tasks, and provide monitoring, also the processed archived files can be shared to colleage effortless.

### What major risks would this system face?

Data Quaility: If the source data changes structure or Changes in external APIs<strong>(ncei.noaa.gov Down)</strong> can disrupt the ETL process.It could break the ETL process. For example, missing of some day's data for some of the station.

Failed data downloads due to network problems could lead to incomplete or missing data.

Handling over 100k stations requires efficient data processing and storage management to avoid performance bottlenecks.

### What are the next set of enhancements you would add?

Add some checks to make sure the data we're getting is complete, accurate, and consistent. This will help us catch any issues early on.

Improve our error handling and logging so we can quickly diagnose and fix any problems. This means better diagnostics and quicker troubleshooting.

Fine-tune how we store data in Zarr to handle larger datasets efficiently. We'll tweak the chunking strategy based on how the data will be used next to make sure it's fast and efficient.

Chuncking stratgies and the task execution frequency will be based on the next step usage of the data.

Update the code to use multithreading and Spark (Databricks) for better scalability. This will help us efficiently handle data from over 100,000 stations.

### How would you improve the clarity of this assignment?

Specify the required frequency task execution, which will impact the chuncking stratgies pretty much, as well as task execution progress as well as cost of task execution

Clarify whether the script should handle historical data backfilling.

Provide this project background information as well as potential usage of the output archieve file, which can help interviewer to optimize the script

In [ ]:

In [ ]: