poor man's CDC home assignment

Goal

The goal of this exercise is to **create a data replication pipeline** for a single table.

Given a table in a source database, you should **provide a sync process to a target database table**.

What does it mean? We want to clone (and keep an hourly updated copy) of a database table.

The table should

Deletes are hard-deletes and should be reflected in the sync as well (destination database should omit deleted rows).

Source table

Structure

column name	type	
id	integer	PK (index/autoincrement/unique)
s3_path	nvarchar(1024)	
format	nvarchar(5)	
type	tinyint	enumeration (image:1 , video:2)
updated_at	timestamp	

Example

id	s3_path	format	type	updated_at
1	bucket/path/filename	jpg	1	2021-06-01T22:00:00
2	bucket/path/filename	mp4	2	2021-06-01T23:10:12

3 hour window example content (asterisk denotes changed rows)

1st hour

```
|id      |   s3_path       |  format       |  type       |
updated_at       |
|--------------------------------------------------------------------------
----|
|1       |   /a.jpg        |  jpg          |  1          | 2021-01-
01T00:01 |
|2       |   /b.jpg        |  jpg          |  1          | 2021-01-
01T00:20 |
|3       |   /c.jpg        |  jpg          |  1          | 2021-01-
01T00:30 |
```

2nd hour

```
|id      |   s3_path       |  format       |  type       |
updated_at       |
|-------------------------------------------------------------------------
----|
|1       |   /xx.jpg       |  jpg          |  1          | 2021-01-
01T01:21 |
|2       |   /b.jpg        |  jpg          |  1          | 2021-01-
01T00:20 |
|3       |   /c.jpg        |  jpg          |  1          | 2021-01-
01T00:30 |
|4       |   /d.mp4        |  mp4          |  2          | 2021-01-
01T01:30 |
|5       |   /e.jpg        |  jpg          |  1          | 2021-01-
01T01:40 |
```

3rd hour

```
|id      |   s3_path       |  format       |  type       |
updated_at       |
|-------------------------------------------------------------------------
----|
|1       |   /x.jpg        |  jpg          |  1          | 2021-01-
01T01:21 |
|2       |   /zz.gif       |  gif          |  1          | 2021-01-
01T02:30 |


|4       |   /d.mp4        |  mp4          |  2          | 2021-01-
01T01:30 |
|5       |   /e.jpg        |  jpg          |  1          | 2021-01-
01T01:40 |
|6       |   /f.jpg        |  jpg          |  1          | 2021-01-
01T01:40 |
```

Constraints

- Each row in the table has an id column (primary key auto-increment) and updated_at column(database timestamp , not indexed). Both columns are automatically populated during insertion of new record. (see table structure below)
- updated_at column is re-populated upon update event in the source DB.(see table structure below)
- Table width is 5 columns and length is up to 100M rows (the process should apply for 50 rows table and 100M rows table)
- DML (insert/update/delete) throughput is ~10 operations/second
- We cant utilize the source's database CDC (change data capture) capabilities (e.g. we can't setup nor ingest bin log). For that matter we do not own this DB , and have only **read access** to it (meaning we can't change anything in it or it's tables and indexes)
- The replication process should not interfere break or block production flows.
- row level deletes are "hard" deletes (the row is deleted from the table)

Assignment Rules:

- Source db and dest db should reside on the same DB technology (sqlite/ mysql/ postgresql) - they should reside on two different databases / schemas. you cannot join nor copy directly between the to DBs
- Process runtime
  - the sync process should be written in python
  - using airflow is a plus.
- since there is no actual writes to source DB the pipeline should start with seeding the change to source db (as described in 3 hour window example)
- sync process should address insert / updated and deleted columns . replica table should look exactly like the source
- Expected outcome is a fully functional python CLI application (or airflow DAG)
- validation steps are plus
- alerting steps are plus
- addressing schema evolution (what happens when a new column is added to the source) is a plus

Open Questions

Q1: your source table size is less than 1000 rows and increases in rate of 1/day what is the cost/effective solution you would recommend.

Q2: what are the available strategies for handling schema changes ?

Submitting the assignment:

1. send zipped folder with python code and sqlite database file
2. Provide both project Installation and Execution procedures in the README.md file and include them in the project.
3. Send an email containing:
   a. A link to the project repository at Github.
   b. An answer to the open question (either as a separate DOC or as part of the email body).