Data Engineering Challenge

At Data Engineering we have been tasked to create a CLI tool that extracts a list of users from an external provider API and loads them into a SQLite database, performing any data transformation as needed.

The API exposes the following endpoint:

URL: http://de-challenge.ltvco.com/v1/users
Method: `GET`
Query Arguments:

- **api_key**: API key used to authenticate into the service (**required**)
- **created_at**: Date used to filter the users based on their creation date (**required**) (YYYY-MM-DD)

Successful Response:

```
[
  {
    "active": "true",
    "address": {
      "state": "OK",
      "zip": "43574"
    },
    "contact": {
      "email": "trentonhuels@cruickshank.io",
      "phone": "8207992744"
  },
    "created_at": 337193736372486642,
    "dob": "1982-03-28",
    "gender": "male",
    "name": {
      "first": "Emma",
      "last": "McClure"
    },
    "user_id": "a7d8feae-cac5-40c2-8272-53b4089636c7"
  }
]
```

Error Response:

```
{
  "error": "error message"
}
```

The data dictionary offered by the external provided is attached below:

Field	Data Type	Description	Allowed Values	Nullable
active	string	Indicates if the user is currently active	• “true” • “false”	No
address	object	The user address information		No
address.state	string	The 2 characters US state abbreviation		No
address.zip	string	The 5 digit ZIP code		No
contact	object	The user contact information		Yes
contact.email	string	The email address		Yes
contact.phone	string	The phone number		Yes
created_at	integer	The UNIX timestamp with nanosecond precision when the user was created		No
dob	string	The date of birth in YYYY-MM-DD format		Yes
gender	string	The gender of the user	• “male” • “female”	No
name	object	The user name information		No
name.first	string	The first name		No
name.last	string	The last name		No
user_id	string	The user identification value		No

The table where the users need to be loaded have the following schema:

```
CREATE TABLE `user` (
  `id` integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  `user_id` varchar(36) NOT NULL DEFAULT '',
  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  `first_name` varchar(255) NOT NULL DEFAULT '',
  `last_name` varchar(255) NOT NULL DEFAULT '',
  `date_of_birth` date DEFAULT NULL,
```

```
  `gender` varchar(6) NOT NULL DEFAULT '',
  `phone` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `state` varchar(2) NOT NULL DEFAULT '',
  `zip` integer NOT NULL,
  `active` bool NOT NULL
);
```

Your mission is to build a CLI tool to process the user data

- The tool should receive the date to process as an argument.
- The tool should retrieve the users from the API using the input date, store them into the provided database and exit.
- It must be implemented using Golang >= 1.13 or Python 3.
    - If you chose Golang you should include a *go.mod* and *go.sum* files containing all the dependencies you used.
    - If you chose Python you should include a *requirements.txt* file containing all the dependencies you used.
- There must be a README file that explains how to set up and use the tool.
- Code must be in GitHub.

Extra Credit

- Include a Dockerfile
- Add unit testing

Assets

- The SQLite database can be downloaded from https://s3.amazonaws.com/ltvco-challenge/de-challenge.db
- The API key to authenticate into the service is ec093dd5-bbe3-4d8e-bdac-314b40afb796

Additional information

- When you get started, please provide the URL to the project on GitHub so we can follow along.
- If you get stuck or have any questions, **please** do not hesitate to reach out to us.