Slowly Changing Dimension Type 2

In dimensional modeling, it is very important to determine how the change of data in the source system reflects in dimension tables in the data warehouse system. This phenomenon is known as slowly changing dimensions. This term comes from the reason that dimensions accumulate their changes at the slow rate in comparison with facts in the fact table.

A brief overview of slowly changing dimension can be obtained from below link:

http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html
https://en.wikipedia.org/wiki/Slowly_changing_dimension

There are three most common slowly change dimension types, which are known as slowly changing dimension type 1, type 2 and type 3. Let's examine type 2 and way to handle the situation in a greater detail.

To implement a SCD type 2, I am using below Yelp user data file downloaded from the Kaggle Yelp data competition.

Download the data file from below link:

https://www.kaggle.com/yelp-dataset/yelp-dataset/version/6#yelp_user.csv

Yelp data file details:

1326100 unique user records with below attributes:

```
user_id, name, review_count, yelping_since, useful, funny, cool,
fans, elite, average_stars, compliment_hot, compliment_more,
compliment_profile, compliment_cute, compliment_list,
compliment_note, compliment_plain, compliment_cool,
compliment_funny, compliment_writer, compliment_photos, date
```

To obtain the data Yelp user monthly data files, download the file from above link and follow the steps in below Python code file

https://github.com/sahilbhange/slowly-changing-dimention/blob/master/Yelp_user.py

Objective: Task is to identify the Yelp user attribute (like average_rating, fans, review count, etc) changes over time and maintain every version of user record if there is any change in the attribute.

If any of the below customer attribute changes on a given day, a new effective record of customer will be created and an old record will be expired with appropriate expiry date.

```
review_count, useful, funny, cool, fans, average_stars,
compliment_hot, compliment_more, compliment_profile,
compliment_cute, compliment_list, compliment_note,
compliment_plain, compliment_cool, compliment_funny,
compliment_writer, compliment_photos
```

Example: For a given Yelp user, on process date '20180731', below is the review_count, useful and average_stars values,

proc_dt	user_id	name	review_count	useful	average_stars
7/31/18	jYnkJR3T8yCERXywoVhWYA	Hugo	48	15	3.73

For process date '20180831', its review_count, useful and average_stars values changed as below:

proc_dt	user_id	name	review_count	useful	average_stars
8/31/18	jYnkJR3T8yCERXywoVhWYA	Hugo	60	30	4.5

Thus, in the historical table, the user versions will be stored as below where older record is expired with date '20180730' and new record will be an effective with the expiry date '20991231' as below:

proc_dt	user_id	name	review_count	useful	average_stars	eff_dt	exp_dt
7/31/18	jYnkJR3T8yCERXywoVhWYA	Hugo	48	15	3.73	7/31/18	7/30/18
8/31/18	jYnkJR3T8yCERXywoVhWYA	Hugo	60	30	4.5	8/31/18	12/31/99