About data:

The dataset for this use case contains heartbeat data collected from health devices of 5 individuals per hour over a period of 5 months - January 2020 to May 2020. Be mindful about the fact that 2020 was a leap year.

For each month there are two types of files:

1. File ending with month number. E.g.: health\_tracker\_data\_2020\_1.json
2. file ending with \_late. This file contains records that arrived late. For instance some records for January month arrived in February. So these records will be present in 1\_late.json and not in 2.json. This file may or may not be present for each month. E.g.: health\_tracker\_data\_2020\_1\_late.json

File name format:

1. health\_tracker\_data\_2020\_<month>.json
2. health\_tracker\_data\_2020\_<month>\_late.json

Here month value is:

1. 1 for January
2. 2 for February
3. 3 for March
4. 4 for April
5. 5 for May

Data location: [https://hadoop-and-big-data.s3-us-west-2.amazonaws.com/fitness-tracker/<file\_name>.json](https://hadoop-and-big-data.s3-us-west-2.amazonaws.com/fitness-tracker/%3cfile_name%3e.json)

You can use databricks community edition to solve this challenge. Spin up a cluster in databricks workspace and open a notebook to get started.

Challenge:

For this exercise, you’ll be required to do the following:

1. Read raw file and ingest into delta table.
2. Perform batch upload of new data.
3. Perform data quality checks and find issues with quality of data.
4. If there are any issues, then rectify the data and update the table.
5. Scrub personal data of a user.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Task | Status | Remark |
| 01 | Ingest the file for month of January from the remote location and get it into a raw directory in dbfs. (Bronze layer) You can either download all the files to your local machine and ingest it or you can use python urllib to retrieve and ingest file at a dbfs location. |  |  |
| 02 | Review the data and analyse it. What’s the format of data? List out columns present in data along with their data type |  |  |
| 03 | Extract the data for first month and visualise it. |  |  |
| 04 | Transform the incoming table by splitting Unix timestamp column into date and time. |  |  |
| 05 | Create an empty delta table. |  |  |
| 06 | Store the transformed data into the delta table |  |  |
| 07 | Add comments to the columns of the table. Comments can make your tables easier to read and maintain |  |  |
| 08 | Create an aggregated layer (gold layer) of the heart rate data for analytics purposes. Calculate minimum, average, maximum heart rate recorded by each device for month of January |  |  |
| 09 | Read data for February month and append it to Delta table. Can you verify how many records were added in both the inserts (step 6 and step 9). |  |  |
| 10 | Analyse the data and check for data quality. Do you think the data is intact? Hint - Check description of dataset. |  |  |
| 11 | If there is any discrepancy, then find the issue. List down your approach for checking data quality and for identifying data quality issues. |  |  |
| 11 | List out any broken readings in the data set and total count of broken readings. |  |  |
| 12 | Design a logic to fix the broken readings - e.g.: assign a value, calculate averages, use SQL functions etc. |  |  |
| 13 | Merge the modified broken readings into the delta table. |  |  |
| 14 | Read data for February month late file and append it to Delta table. |  |  |
| 15 | Analyse the data and check for data quality. Do you think the data is intact? If not, then do the needful to rectify the data. |  |  |
| 16 | Read the data for March month and insert into the table. If any error occurs, then include required options in your code to insert new records |  |  |
| 17 | Mask name column of the user of device ID 1 in the entire table. |  |  |
| 18 | Delete the records of device ID 5. Ensure that it has been deleted in a way that data cannot be recovered at all |  |  |