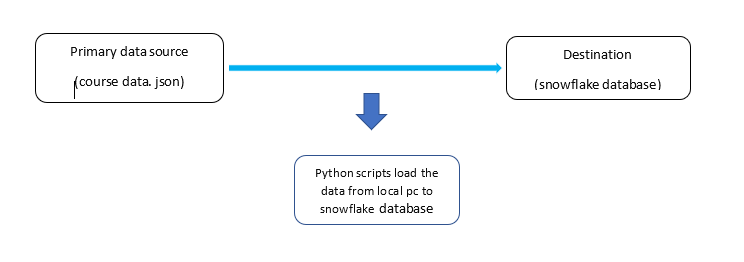
**Step one: -** Create a pipeline using python and suitable python libraries.



I used two python scripts to load the data in to snowflake database. The first script is dimention.py script, this script use to insert the data in to the dimension tables in star schema. The second script is the Fact.py, this script use to insert data in to the fact table in the star schema. The main reason to use two data loading scripts is to reduce the complexity. (first you need to run the dimension.py and secondly need to run the fact.py for the data load)

**python libraries imported for the scripts**

**import snowflake. Connector =** Snowflake Connector to connect to the Snowflake data warehouse and perform data access operations

**import json =** Read the json details

**import logging =** create error log files

**second step: -** Create a suitable star schema to analyze course sales.

In star schema there can be have one fact table and several number of associated dimension tables. Fact table contain measures and dimension table contain the details that describe the measures. I took course sales as the fact table measure to create the star schema.

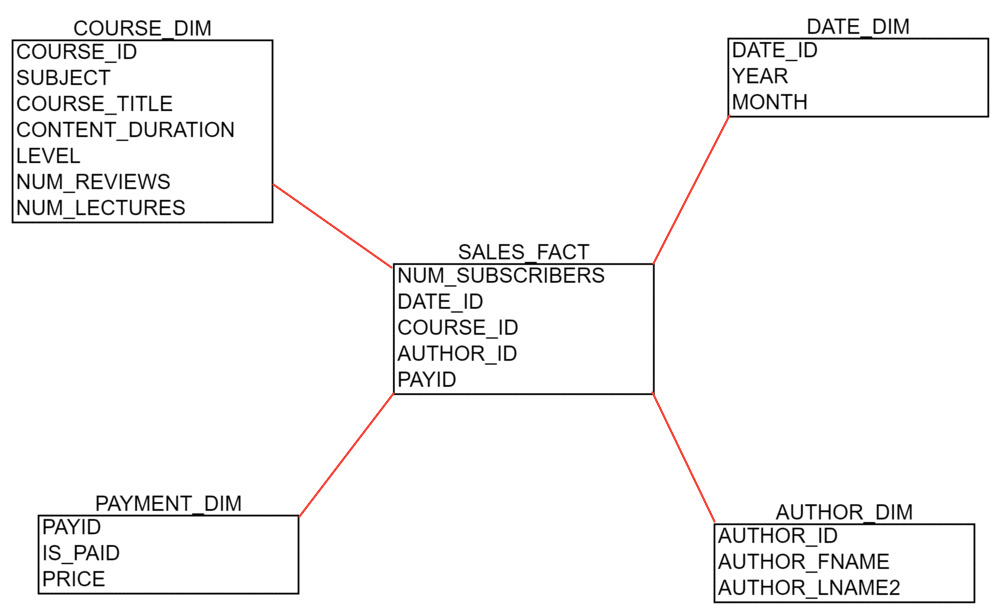
**Assumptions and modifications did to create the star schema.**

In the data source we couldn’t find a course sales data so we did assumption that each course subscriber is a course sale

From authorID we can extract three values (author code, author first name, author last name) so we split authorID and inserts that vales in to Author\_dim table.

From published timestamp we can extract two values (year, month) so we split published timestamp and insert that values in to Date\_dim table.

**diagram for the star schema**



As in the diagram there are 4-dimension tables that describe sales fact table. In the fact table num\_subscribers are the measure value and there are also 4 columns for the unique ids of four-dimension tables. According to the above design five tables are created in the WILEY\_DB inside the snowflake.

**Step three**: - Write 5 analytical queries based one your star schema.

(1)Rank () function is a window that assigns a rank to each row. This function Apply on the result set in the query and Rank them in descending order based on the order by column value. In the below query we e gave rank to the result set using **NUM\_REVIEWS and order them in descending order**

**SELECT**

**COURSE\_ID,**

**COURSE\_TITLE,**

**SUBJECT,**

**RANK () OVER (**

**ORDER BY NUM\_REVIEWS DESC**

**) rank\_column**

**FROM**

**"WILEY\_DB"."PUBLIC"."COURSE\_DIM";**

(2) Sql inner join use to join several columns of multiple tables and create temporary join table, based on that join table we can analyze the data. In below query I joined all the NUM\_SUBSCRIBERS, COURSE\_ID, COURSE\_TITLE, LEVEL columns that LEVEL = 'All Levels' and NUM\_SUBSCRIBERS < 100

**select d.NUM\_SUBSCRIBERS,d.COURSE\_ID,d2.COURSE\_TITLE,d2.LEVEL from "WILEY\_DB"."PUBLIC"."SALES" as d join "WILEY\_DB"."PUBLIC"."COURSE\_DIM" as d2 where d2.LEVEL = 'All Levels' and d.NUM\_SUBSCRIBERS < 100**

(3)Rank () function with **partition By** clause. There are three course levels in the COURSE\_DIM table so if want to give a rank for each level separately we need to use **partition By** clause.

**SELECT**

**COURSE\_ID,**

**COURSE\_TITLE,**

**NUM\_REVIEWS ,**

**SUBJECT,**

**LEVEL,**

**RANK () OVER (**

**PARTITION BY LEVEL**

**ORDER BY NUM\_REVIEWS DESC**

**) partition\_rank**

**FROM**

**"WILEY\_DB"."PUBLIC"."COURSE\_DIM";**

(4)CUME\_DIST () function to calculate a cumulative distribution of a value within a group of values. We calculate cumulative distribution for order by CONTENT\_DURATION and DESC order

**SELECT**

**COURSE\_ID,**

**COURSE\_TITLE,**

**SUBJECT,**

**NUM\_REVIEWS,**

**CUME\_DIST() OVER (**

**PARTITION BY LEVEL**

**ORDER BY CONTENT\_DURATION DESC**

**) cume\_dist**

**FROM "WILEY\_DB"."PUBLIC"."COURSE\_DIM"**

(5)Below query only filter and join the free paid All level courses

**select d.IS\_PAID,d.PRICE,d2.SUBJECT,d2.COURSE\_TITLE,d2.LEVEL from**

**"WILEY\_DB"."PUBLIC"."PAYMENT\_DIM" as d join "WILEY\_DB"."PUBLIC"."COURSE\_DIM" as d2**

**where d.IS\_PAID = FALSE and d2.LEVEL = 'All Levels'**

(6) PERCENT\_RANK () Percentage rank calculate specific percentage value for each row. We can order the rows based on the decedent order of percentage and get a analytical idea about data. Below we apply the the precent rank on the result set order by the num\_lectures in a course and the order the rows according to decedent order

**SELECT**

**COURSE\_ID,**

**COURSE\_TITLE,**

**NUM\_REVIEWS,**

**SUBJECT,**

**PERCENT\_RANK () OVER (**

**ORDER BY NUM\_LECTURES DESC**

**) presentage\_rank**

**FROM**

**"WILEY\_DB"."PUBLIC"."COURSE\_DIM";**

(7) Below query provides the details of all the authors that published their courses in month of December before 2015

**select d.DATE\_ID,d.YEAR,d.MONTH,d2.AUTHOR\_ID,d2.AUTHOR\_FNAME from**

**"WILEY\_DB"."PUBLIC"."DATE\_DIM" as d join "WILEY\_DB"."PUBLIC"."AUTHOR\_DIM" as d2 where YEAR <'2015' AND MONTH = '12'**

**Issues faced during the project**

(1) This is the first time that used the snowflake platform so at the very beginning it took little bit time to learn how to use tools in the platform.

(2) Faced some issues when loading data in to the database due to the bad quality of source data.

In course\_title json key there are some error data that can’t insert in to the d database.

EX: - If there is extra double quotation in the data it gave an error when insertion in to the database.

 "165":"How to remove 'Risk from Penny Stocks - Options Trading\"",

Solution: - cleaned the data source and made insert process easy.

If we use that that error data include json file it will create some issues for the data insertion.