MINI\_ASSIGNMENT-1

ABDUL AABD KHAN

APPROACH ( USING SNOWSQL):

* Installing snowsql
* Making internal and name stage with cmds:

use database DEMO;

use schema dbt;

use role demo\_role;

create table MINI\_XML(

MINI\_COL\_XML VARIANT

);

SELECT \* FROM information\_schema.tables WHERE table\_name="MINI\_XML";

SELECT \* FROM information\_schema.tables WHERE table\_name='MINI\_XML';

desc table mini\_xml;

create or replace FILE FORMAT xml\_load TYPE='XML' COMPRESSION='AUTO'

DESCRIBE FILE FORMAT XML\_LOAD;

show file formats like '%xml\_load'

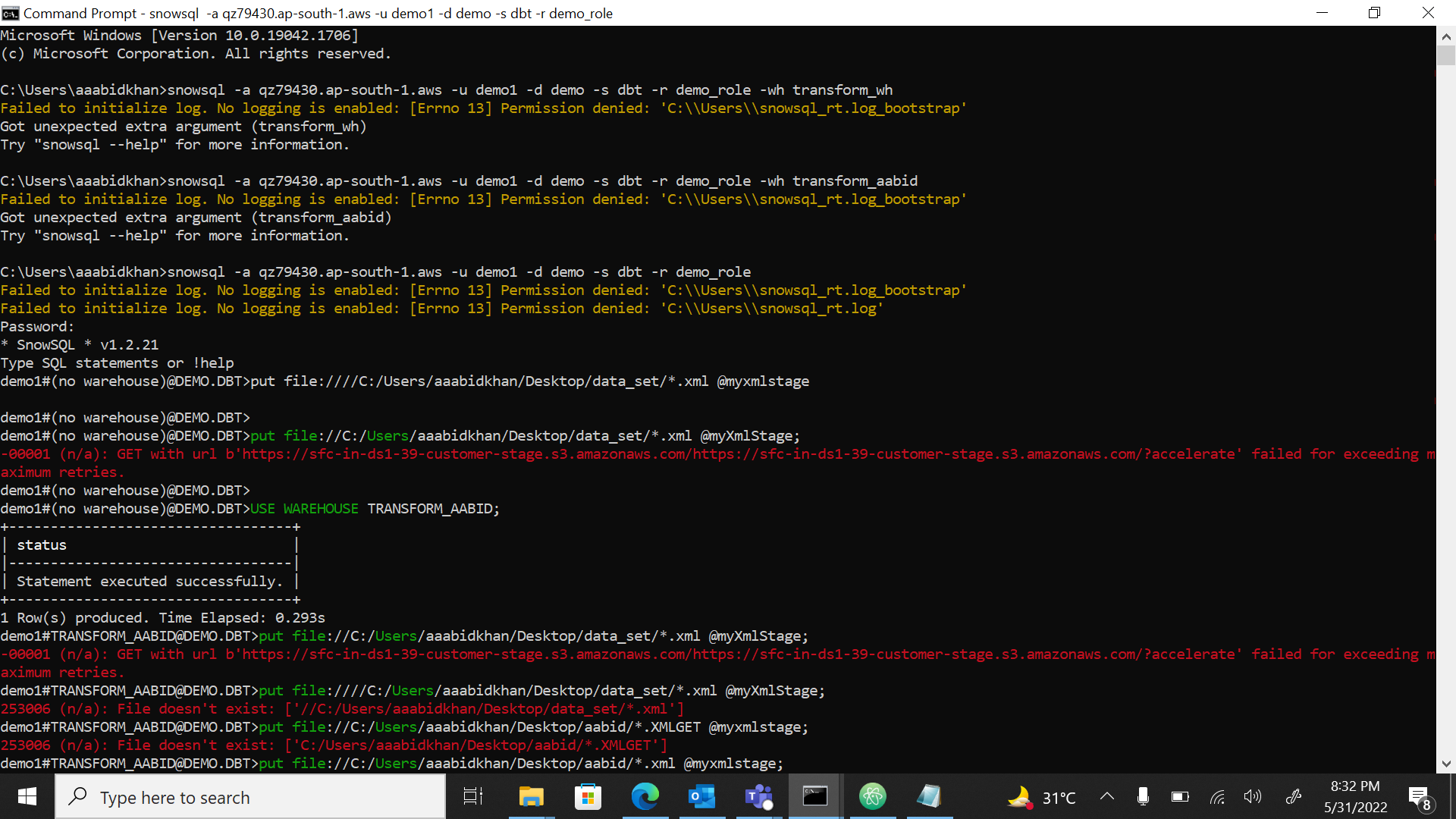
select \* from information\_schema.file\_formats

CREATE OR REPLACE STAGE myXmlStage file\_format = xml\_load

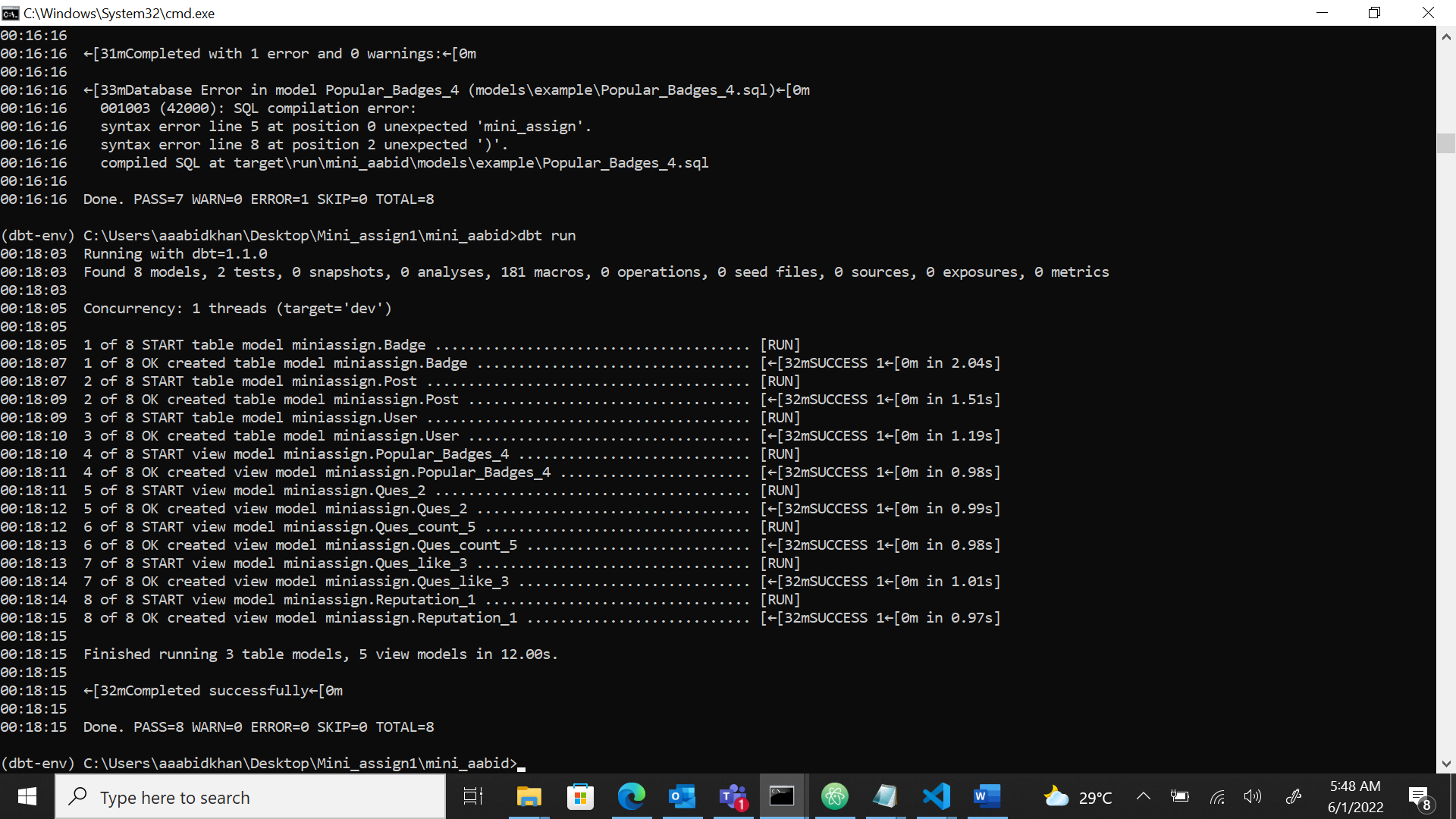
show stages like '%myXmlStage'

desc stage myXmlStage

select \* from information\_schema.stages where stage\_name = 'MYXMLSTAGE'

* Got error: unable to load data set using put method
* Screenshot: 

APPROACH 2( USING PYTHON SCRIPT AND WEB UI):

* Created python file to convert the large xml files into smaller files.
* CODE:
* import xml.etree.ElementTree as ET
* context = ET.iterparse('Posts.xml', events=('end', ))
* parts = []
* title = 'posts-'
* count = 0
* for event, elem in context:
* if len(parts) > 7500:
* filename = format(title + str(count) + ".xml")
* with open("" + filename, 'w') as f:
* f.write("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n")
* f.write("<posts>\n")
* f.write("\n".join(parts))
* f.write("</posts>")
* parts = []
* count += 1
* if elem.tag == 'row':
* parts.append(ET.tostring(elem).decode())
* filename = format(title + str(count) + ".xml")
* with open(filename, 'w') as f:
* f.write("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n")
* f.write("<posts>\n")
* f.write("\n".join(parts))
* f.write("</posts>")
* Then created table through web ui and loaded data (chunks).
* Created virtual env and run cmd “dbt init Aabid\_mini” to initialize dbt
* Run the code through “dbt run”
* 
* All the table and views created run successfully.