import pandas as pd

if 'transformer' not in globals():

    from mage\_ai.data\_preparation.decorators import transformer

if 'test' not in globals():

    from mage\_ai.data\_preparation.decorators import test

@transformer

def transform(df, \*args, \*\*kwargs):

    """

    Template code for a transformer block.

    Add more parameters to this function if this block has multiple parent blocks.

    There should be one parameter for each output variable from each parent block.

    Args:

        data: The output from the upstream parent block

        args: The output from any additional upstream blocks (if applicable)

    Returns:

        Anything (e.g. data frame, dictionary, array, int, str, etc.)

    """

*# Specify your transformation logic here*

    df['tpep\_pickup\_datetime'] = pd.to\_datetime(df['tpep\_pickup\_datetime'])

    df['tpep\_dropoff\_datetime'] = pd.to\_datetime(df['tpep\_dropoff\_datetime'])

    datetime\_dim = df[['tpep\_pickup\_datetime','tpep\_dropoff\_datetime']].drop\_duplicates().reset\_index(drop=True)

    datetime\_dim['pick\_hour'] = datetime\_dim['tpep\_pickup\_datetime'].dt.hour

    datetime\_dim['pick\_day'] = datetime\_dim['tpep\_pickup\_datetime'].dt.day

    datetime\_dim['pick\_month'] = datetime\_dim['tpep\_pickup\_datetime'].dt.month

    datetime\_dim['pick\_year'] = datetime\_dim['tpep\_pickup\_datetime'].dt.year

    datetime\_dim['pick\_weekday'] = datetime\_dim['tpep\_pickup\_datetime'].dt.weekday

    datetime\_dim['drop\_hour'] = datetime\_dim['tpep\_dropoff\_datetime'].dt.hour

    datetime\_dim['drop\_day'] = datetime\_dim['tpep\_dropoff\_datetime'].dt.day

    datetime\_dim['drop\_month'] = datetime\_dim['tpep\_dropoff\_datetime'].dt.month

    datetime\_dim['drop\_year'] = datetime\_dim['tpep\_dropoff\_datetime'].dt.year

    datetime\_dim['drop\_weekday'] = datetime\_dim['tpep\_dropoff\_datetime'].dt.weekday

    datetime\_dim['datetime\_id'] = datetime\_dim.index

    datetime\_dim = datetime\_dim[['datetime\_id', 'tpep\_pickup\_datetime', 'pick\_hour', 'pick\_day', 'pick\_month', 'pick\_year', 'pick\_weekday',

                             'tpep\_dropoff\_datetime', 'drop\_hour', 'drop\_day', 'drop\_month', 'drop\_year', 'drop\_weekday']]

    passenger\_count\_dim = df[['passenger\_count']].drop\_duplicates().reset\_index(drop=True)

    passenger\_count\_dim['passenger\_count\_id'] = passenger\_count\_dim.index

    passenger\_count\_dim = passenger\_count\_dim[['passenger\_count\_id','passenger\_count']]

    trip\_distance\_dim = df[['trip\_distance']].drop\_duplicates().reset\_index(drop=True)

    trip\_distance\_dim['trip\_distance\_id'] = trip\_distance\_dim.index

    trip\_distance\_dim = trip\_distance\_dim[['trip\_distance\_id','trip\_distance']]

    rate\_code\_type = {

        1:"Standard rate",

        2:"JFK",

        3:"Newark",

        4:"Nassau or Westchester",

        5:"Negotiated fare",

        6:"Group ride"

    }

    rate\_code\_dim = df[['RatecodeID']].drop\_duplicates().reset\_index(drop=True)

    rate\_code\_dim['rate\_code\_id'] = rate\_code\_dim.index

    rate\_code\_dim['rate\_code\_name'] = rate\_code\_dim['RatecodeID'].map(rate\_code\_type)

    rate\_code\_dim = rate\_code\_dim[['rate\_code\_id','RatecodeID','rate\_code\_name']]

    pickup\_location\_dim = df[['pickup\_longitude', 'pickup\_latitude']].drop\_duplicates().reset\_index(drop=True)

    pickup\_location\_dim['pickup\_location\_id'] = pickup\_location\_dim.index

    pickup\_location\_dim = pickup\_location\_dim[['pickup\_location\_id','pickup\_latitude','pickup\_longitude']]

    dropoff\_location\_dim = df[['dropoff\_longitude', 'dropoff\_latitude']].drop\_duplicates().reset\_index(drop=True)

    dropoff\_location\_dim['dropoff\_location\_id'] = dropoff\_location\_dim.index

    dropoff\_location\_dim = dropoff\_location\_dim[['dropoff\_location\_id','dropoff\_latitude','dropoff\_longitude']]

    payment\_type\_name = {

        1:"Credit card",

        2:"Cash",

        3:"No charge",

        4:"Dispute",

        5:"Unknown",

        6:"Voided trip"

    }

    payment\_type\_dim = df[['payment\_type']].drop\_duplicates().reset\_index(drop=True)

    payment\_type\_dim['payment\_type\_id'] = payment\_type\_dim.index

    payment\_type\_dim['payment\_type\_name'] = payment\_type\_dim['payment\_type'].map(payment\_type\_name)

    payment\_type\_dim = payment\_type\_dim[['payment\_type\_id','payment\_type','payment\_type\_name']]

    fact\_table = df.merge(passenger\_count\_dim, on='passenger\_count') \

             .merge(trip\_distance\_dim, on='trip\_distance') \

             .merge(rate\_code\_dim, on='RatecodeID') \

             .merge(pickup\_location\_dim, on=['pickup\_longitude', 'pickup\_latitude']) \

             .merge(dropoff\_location\_dim, on=['dropoff\_longitude', 'dropoff\_latitude'])\

             .merge(datetime\_dim, on=['tpep\_pickup\_datetime','tpep\_dropoff\_datetime']) \

             .merge(payment\_type\_dim, on='payment\_type') \

             [['VendorID', 'datetime\_id', 'passenger\_count\_id',

               'trip\_distance\_id', 'rate\_code\_id', 'store\_and\_fwd\_flag', 'pickup\_location\_id', 'dropoff\_location\_id',

               'payment\_type\_id', 'fare\_amount', 'extra', 'mta\_tax', 'tip\_amount', 'tolls\_amount',

               'improvement\_surcharge', 'total\_amount']]

    return {"datetime\_dim":datetime\_dim.to\_dict(orient="dict"),

    "passenger\_count\_dim":passenger\_count\_dim.to\_dict(orient="dict"),

    "trip\_distance\_dim":trip\_distance\_dim.to\_dict(orient="dict"),

    "rate\_code\_dim":rate\_code\_dim.to\_dict(orient="dict"),

    "pickup\_location\_dim":pickup\_location\_dim.to\_dict(orient="dict"),

    "dropoff\_location\_dim":dropoff\_location\_dim.to\_dict(orient="dict"),

    "payment\_type\_dim":payment\_type\_dim.to\_dict(orient="dict"),

    "fact\_table":fact\_table.to\_dict(orient="dict")}

@test

def test\_output(output, \*args) -> None:

    """

    Template code for testing the output of the block.

    """

    assert output is not None, 'The output is undefined'