"""Module for generating comparison reports"""

import pandas as pd

import numpy as np

from typing import Dict, List, Optional

import os

def generate\_comparison\_report(source\_df: pd.DataFrame, target\_df: pd.DataFrame,

                           column\_mapping: Dict[str, str], join\_keys: Optional[List[str]],

                           output\_path: str) -> None:

    """Generate pandas-based comparison report"""

    try:

        # Get source and target columns

        source\_cols = list(column\_mapping.keys())

        target\_cols = [column\_mapping[col] for col in source\_cols]

        # Select relevant columns and copy DataFrames

        source\_compare\_df = source\_df[source\_cols].copy()

        target\_compare\_df = target\_df[target\_cols].copy()

        # Rename target columns to match source

        target\_compare\_df.columns = source\_cols

        # Convert all columns to string and handle nulls

        for col in source\_cols:

            source\_compare\_df[col] = source\_compare\_df[col].fillna('').astype(str)

            target\_compare\_df[col] = target\_compare\_df[col].fillna('').astype(str)

        # Perform merge

        if isinstance(join\_keys, (list, tuple)) and join\_keys:

            # Use provided join keys if they exist in source columns

            valid\_keys = [k for k in join\_keys if k in source\_cols]

            if valid\_keys:

                merged\_df = pd.merge(

                    source\_compare\_df,

                    target\_compare\_df,

                    on=valid\_keys,

                    how='outer',

                    indicator=True

                )

            else:

                # Fall back to index-based comparison if no valid join keys

                merged\_df = pd.merge(

                    source\_compare\_df,

                    target\_compare\_df,

                    left\_index=True,

                    right\_index=True,

                    how='outer',

                    indicator=True

                )

        else:

            # Use index-based comparison if no join keys provided

            merged\_df = pd.merge(

                source\_compare\_df,

                target\_compare\_df,

                left\_index=True,

                right\_index=True,

                how='outer',

                indicator=True

            )

        # Calculate statistics

        stats = {

            'total\_rows\_source': len(source\_compare\_df),

            'total\_rows\_target': len(target\_compare\_df),

            'matching\_rows': len(merged\_df[merged\_df['\_merge'] == 'both']),

            'source\_only\_rows': len(merged\_df[merged\_df['\_merge'] == 'left\_only']),

            'target\_only\_rows': len(merged\_df[merged\_df['\_merge'] == 'right\_only'])

        }

        # Calculate match percentage

        total\_rows = len(source\_compare\_df)

        match\_percentage = (stats['matching\_rows'] / total\_rows \* 100) if total\_rows > 0 else 0

        # Generate summary

        summary = f"""

        Comparison Summary:

        - Total Rows in Source: {stats['total\_rows\_source']}

        - Total Rows in Target: {stats['total\_rows\_target']}

        - Matching Rows: {stats['matching\_rows']}

        - Source Only Rows: {stats['source\_only\_rows']}

        - Target Only Rows: {stats['target\_only\_rows']}

        - Match Percentage: {match\_percentage:.2f}%

        """

        # Generate column statistics

        col\_stats = []

        for col in source\_cols:

            col\_stats.append({

                'Column': col,

                'Source\_Count': len(source\_compare\_df[col].dropna()),

                'Source\_Unique': source\_compare\_df[col].nunique(),

                'Target\_Count': len(target\_compare\_df[col].dropna()),

                'Target\_Unique': target\_compare\_df[col].nunique(),

                'Match\_Status': 'Match' if source\_compare\_df[col].equals(target\_compare\_df[col]) else 'Mismatch'

            })

        col\_stats\_df = pd.DataFrame(col\_stats)

        col\_stats\_html = col\_stats\_df.to\_html(classes='table table-striped', index=False)

        # Generate HTML for mismatched records

        source\_only = merged\_df[merged\_df['\_merge'] == 'left\_only']

        target\_only = merged\_df[merged\_df['\_merge'] == 'right\_only']

        source\_only\_html = source\_only.to\_html(classes='table table-striped', index=False) if not source\_only.empty else '<p>No records unique to source</p>'

        target\_only\_html = target\_only.to\_html(classes='table table-striped', index=False) if not target\_only.empty else '<p>No records unique to target</p>'

        # Write HTML report

        with open(output\_path, 'w') as f:

            f.write(f"""

            <!DOCTYPE html>

            <html lang="en">

            <head>

                <title>Data Comparison Report</title>

                <style>

                    body {{ font-family: Arial, sans-serif; margin: 20px; }}

                    .report {{ max-width: 1200px; margin: 0 auto; }}

                    .section {{ margin: 20px 0; padding: 20px; border: 1px solid #ddd; border-radius: 5px; }}

                    .match {{ color: green; }}

                    .mismatch {{ color: red; }}

                    table {{ border-collapse: collapse; width: 100%; }}

                    th, td {{ border: 1px solid #ddd; padding: 8px; text-align: left; }}

                    th {{ background-color: #f5f5f5; }}

                </style>

            </head>

            <body>

                <div class="report">

                    <h1>Data Comparison Report</h1>

                    <div class="section">

                        <h2>Summary</h2>

                        <pre>{summary}</pre>

                    </div>

                    <div class="section">

                        <h2>Column Statistics</h2>

                        {col\_stats\_html}

                    </div>

                    <div class="section">

                        <h2>Mismatched Records</h2>

                        <h3>Records only in Source:</h3>

                        {source\_only\_html}

                        <h3>Records only in Target:</h3>

                        {target\_only\_html}

                    </div>

                </div>

            </body>

            </html>

            """)

    except Exception as e:

        # Generate error report

        error\_html = f"""

        <!DOCTYPE html>

        <html lang="en">

        <head><title>Comparison Error Report</title></head>

        <body>

            <h1>Error Generating Comparison Report</h1>

            <p>An error occurred: {str(e)}</p>

            <p>Please check that:</p>

            <ul>

                <li>All required columns exist in both datasets</li>

                <li>Join keys are valid column names</li>

                <li>Data types are compatible for comparison</li>

            </ul>

        </body>

        </html>

        """

        with open(output\_path, 'w') as f:

            f.write(error\_html)

        raise Exception(f"Error generating comparison report: {str(e)}")