SD_Study Data Files Scanner and Analyzer

In the drug registration process, SD files (short for Study Data files) refer to electronic data submissions that contain structured datasets from nonclinical and clinical studies. These datasets are prepared following standardized data models to allow regulatory agencies (like the FDA, EMA, or local authorities) to efficiently review, validate, and analyze the study results.

Definition

SD files = Study Data Files — electronic data packages containing the raw and tabulated data from: **Nonclinical studies** (toxicology, pharmacology) **Clinical studies** (efficacy, safety, pharmacokinetics, etc.)

CDISC standards

- SDTM (Study Data Tabulation Model) – standardized structure for tabulated data.
- ADaM (Analysis Data Model) – datasets used for statistical analysis.
- SEND (Standard for Exchange of Nonclinical Data) – for preclinical (animal) data.

This notebook scans the current directory for files and analyzes specific file types using appropriate libraries:

- **.sdf files**: Structure Data File (Chemoinformatics) Plain text molecular data format used to store 3D structures, bonds, properties, and metadata - RDKit for chemical structure analysis
- **.xpt files**: SAS XPORT (SDTM Clinical Datasets) Contains clinical study data such as Demographics (DM), Adverse Events (AE), and Laboratory Results (LB) - pyreadstat for SAS transport files
- **.asnt files**: Textual representation of Abstract Syntax Notation One, sed in regulatory metadata, pharma labeling, and bioinformatics standards - Biopython for ASN.1 files

In [1]:
```
# Install required packages if not already installed
# Uncomment the following lines if you need to install packages

!pip install -q rdkit-pypi
!pip install -q pandas
!pip install -q biopython
```

In [2]:
```
import os
import glob
from pathlib import Path

# Scan current directory for files
current_dir = Path('.')
all_files = list(current_dir.rglob('*'))
files_only = [f for f in all_files if f.is_file()]

print(f"Found {len(files_only)} files in the current directory and subdirectories")

# Group files by extension
file_extensions = {}
for file_path in files_only:
    ext = file_path.suffix.lower()
    if ext not in file_extensions:
        file_extensions[ext] = []
    file_extensions[ext].append(file_path)

print("\nFiles by extension:")
for ext, files in file_extensions.items():
    print(f"{ext}: {len(files)} files")
    for file in files[:5]:  # Show first 5 files per extension
        print(f"  - {file}")
    if len(files) > 5:
        print(f"  ... and {len(files) - 5} more")
```

```
Found 25 files in the current directory and subdirectories

Files by extension:
: 5 files
  - .DS_Store
  - m5/.DS_Store
  - m5/53-clin-stud-reports/.DS_Store
  - m5/53-clin-stud-reports/study1234/.DS_Store
  - m5/53-clin-stud-reports/study1234/datasets/.DS_Store
.ipynb: 1 files
  - SD_Study-Data-files.ipynb
.xlsx: 1 files
  - sample_file_summary.xlsx
.json: 1 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/COMPOUND_CID_197365.json
.xml: 2 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/stf.xml
  - m5/53-clin-stud-reports/study1234/datasets/datasets/define.xml
.xpt: 3 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/lb.xpt
  - m5/53-clin-stud-reports/study1234/datasets/datasets/ae.xpt
  - m5/53-clin-stud-reports/study1234/datasets/datasets/dm.xpt
.asnt: 3 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/Conformer3D_COMPOUND_CID_197366.asnt
  - m5/53-clin-stud-reports/study1234/datasets/datasets/Structure2D_COMPOUND_CID_197365.asnt
  - m5/53-clin-stud-reports/study1234/datasets/datasets/assessment.asnt
.pdf: 3 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/annotated-crf.pdf
  - m5/53-clin-stud-reports/study1234/datasets/datasets/study1234-clin-report.pdf
  - m5/53-clin-stud-reports/study1234/datasets/datasets/study1234-sdtm-rg.pdf
.sdf: 3 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/compound.sdf
  - m5/53-clin-stud-reports/study1234/datasets/datasets/Structure2D_COMPOUND_CID_197365.sdf
  - m5/53-clin-stud-reports/study1234/datasets/datasets/Conformer3D_COMPOUND_CID_197366.sdf
.csv: 3 files
  - m5/53-clin-stud-reports/study1234/datasets/datasets/dm.csv
  - m5/53-clin-stud-reports/study1234/datasets/datasets/ae.csv
  - m5/53-clin-stud-reports/study1234/datasets/datasets/lb.csv
```

SDF Files Analysis (RDKit)

In [3]:
```
# Analyze SDF files using RDKit
try:
    from rdkit import Chem
    from rdkit.Chem import Descriptors

    sdf_files = file_extensions.get('.sdf', [])
    if sdf_files:
        print(f"Found {len(sdf_files)} SDF files")

        for sdf_file in sdf_files:
            print(f"\nAnalyzing {sdf_file}:")

            # Read SDF file
            suppl = Chem.SDMolSupplier(str(sdf_file))
            molecules = [mol for mol in suppl if mol is not None]

            print(f"  - Number of molecules: {len(molecules)}")

            if molecules:
                # Analyze first molecule as example
                mol = molecules[0]
                print(f"  - First molecule: {mol.GetProp('_Name') if mol.HasProp('_Name') else 'Unnamed'}")
                print(f"  - Molecular weight: {Descriptors.MolWt(mol):.2f}")
                print(f"  - Number of atoms: {mol.GetNumAtoms()}")
                print(f"  - Number of bonds: {mol.GetNumBonds()}")
                print(f"  - SMILES: {Chem.MolToSmiles(mol)}")
    else:
        print("No SDF files found")

except ImportError:
    print("RDKit not installed. Install with: pip install rdkit-pypi")
except Exception as e:
    print(f"Error analyzing SDF files: {e}")
```

```
Found 3 SDF files

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/compound.sdf:
  - Number of molecules: 0

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/Structure2D_COMPOUND_CID_197365.sdf:
  - Number of molecules: 1
  - First molecule: 197365
  - Molecular weight: 699.99
  - Number of atoms: 44
  - Number of bonds: 46
  - SMILES: CN1CCN(c2ccc3nc(-c4ccc5nc(CCCc6ccc(N(CCCl)CCCl)cc6)[nH]c5c4)[nH]c3c2)CC1.Cl.Cl.Cl

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/Conformer3D_COMPOUND_CID_197366.sdf:
  - Number of molecules: 1
  - First molecule: 197366
  - Molecular weight: 590.60
  - Number of atoms: 41
  - Number of bonds: 46
  - SMILES: CN1CCN(c2ccc3nc(-c4ccc5nc(CCCc6ccc(N(CCCl)CCCl)cc6)[nH]c5c4)[nH]c3c2)CC1
```

```
[16:12:02] ERROR: Atom line too short: '   -0.0015    1.2095    0.0000 C' on line 5
[16:12:02] ERROR: moving to the beginning of the next molecule
```

XPT Files Analysis (pyreadstat)

In [4]:
```
# Analyze XPT files using pyreadstat
try:
    import pyreadstat

    xpt_files = file_extensions.get('.xpt', [])
    if xpt_files:
        print(f"Found {len(xpt_files)} XPT files")

        for xpt_file in xpt_files:
            print(f"\nAnalyzing {xpt_file}:")

            try:
                # Read XPT file using pyreadstat
                df, meta = pyreadstat.read_xport(str(xpt_file))

                print(f"  - Shape: {df.shape}")
                print(f"  - Columns: {list(df.columns)}")
                print(f"  - First 5 rows:\n{df.head()}")

                # Basic statistics
                print(f"  - Summary statistics:\n{df.describe()}")

            except Exception as file_error:
                print(f"  - Error reading {xpt_file}: {file_error}")

    else:
        print("No XPT files found")

except ImportError as e:
    print(f"Required packages not installed: {e}")
    print("Install with: pip install pyreadstat")
except Exception as e:
    print(f"Error analyzing XPT files: {e}")
```

```
Found 3 XPT files

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/lb.xpt:
  - Shape: (2, 5)
  - Columns: ['STUDYID', 'USUBJID', 'LBTEST', 'LBSTRESN', 'LBSTRESU']
  - First 5 rows:
  STUDYID  USUBJID      LBTEST  LBSTRESN LBSTRESU
0  ABC123  SUBJ001  Hemoglobin      14.2     g/dL
1  ABC123  SUBJ002     Glucose      88.0    mg/dL
  - Summary statistics:
       LBSTRESN
count   2.00000
mean   51.10000
std    52.18448
min    14.20000
25%    32.65000
50%    51.10000
75%    69.55000
max    88.00000

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/ae.xpt:
  - Shape: (2, 5)
  - Columns: ['STUDYID', 'USUBJID', 'AETERM', 'AESEV', 'AEREL']
  - First 5 rows:
  STUDYID  USUBJID    AETERM     AESEV      AEREL
0  ABC123  SUBJ001  Headache      MILD    RELATED
1  ABC123  SUBJ002    Nausea  MODERATE  UNRELATED
  - Summary statistics:
       STUDYID  USUBJID    AETERM AESEV    AEREL
count        2        2         2     2        2
unique       1        2         2     2        2
top     ABC123  SUBJ001  Headache  MILD  RELATED
freq         2        1         1     1        1

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/dm.xpt:
  - Shape: (2, 4)
  - Columns: ['STUDYID', 'USUBJID', 'AGE', 'SEX']
  - First 5 rows:
  STUDYID  USUBJID   AGE SEX
0  ABC123  SUBJ001  45.0   M
1  ABC123  SUBJ002  52.0   F
  - Summary statistics:
             AGE
count   2.000000
mean   48.500000
std     4.949747
min    45.000000
25%    46.750000
50%    48.500000
75%    50.250000
max    52.000000
```

ASNT Files Analysis (Biopython)

In [5]:
```
# Analyze ASNT files using Biopython and XML parsing
try:
    from Bio import SeqIO
    import xml.etree.ElementTree as ET

    def read_asnt(asnt_file):
        # Check if it's XML
        with open(asnt_file, 'r') as f:
            first_line = f.readline()
            if first_line.startswith('<?xml'):
                # It's XML, parse as XML
                tree = ET.parse(asnt_file)
                root = tree.getroot()
                return [root]  # Return as list with one element
            else:
                # Try as ASN.1 sequence
                records = list(SeqIO.parse(asnt_file, "genbank"))
                return records

    asnt_files = file_extensions.get('.asnt', [])
    if asnt_files:
        print(f"Found {len(asnt_files)} ASNT files")

        for asnt_file in asnt_files:
            print(f"\nAnalyzing {asnt_file}:")

            try:
                records = read_asnt(str(asnt_file))
                print(f"  - Number of records: {len(records)}")

                for i, record in enumerate(records[:3]):  # Show first 3 records
                    if hasattr(record, 'id'):  # BioPython record
                        print(f"  - Record {i+1}: {record.id}")
                        print(f"    Description: {record.description}")
                        print(f"    Sequence length: {len(record.seq)}")
                        print(f"    Sequence type: {record.seq.alphabet}")
                    else:  # XML element
                        print(f"  - XML Root: {record.tag}")
                        for child in record:
                            print(f"    {child.tag}: {child.text}")
                print("---")

            except Exception as e:
                print(f"  - Error reading {asnt_file}: {e}")
                # Fallback: read as text
                try:
                    with open(asnt_file, 'r') as f:
                        content = f.read()
                        print(f"  - File size: {len(content)} characters")
                        print(f"  - First 500 characters:\n{content[:500]}...")
                except Exception as e2:
                    print(f"  - Error reading as text: {e2}")
    else:
        print("No ASNT files found")

except ImportError:
    print("Biopython not installed. Install with: pip install biopython")
except Exception as e:
    print(f"Error analyzing ASNT files: {e}")
```

```
Found 3 ASNT files

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/Conformer3D_COMPOUND_CID_197366.asnt:
  - Number of records: 0
---

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/Structure2D_COMPOUND_CID_197365.asnt:
  - Number of records: 0
---

Analyzing m5/53-clin-stud-reports/study1234/datasets/datasets/assessment.asnt:
  - Number of records: 1
  - XML Root: AssessmentTemplate
    StudyID: STUDY1234
    Reviewer: Dr. Smith
    AssessmentDate: 2025-11-05
    Findings: 

    Recommendation: Approved for next review phase
---
```

Summary

In [6]:
```
# Summary of analysis
print("=== ANALYSIS SUMMARY ===")
print(f"Total files scanned: {len(files_only)}")
print(f"File extensions found: {len(file_extensions)}")

# Check specific file types
special_types = ['.sdf', '.xpt', '.asnt']
for ext in special_types:
    count = len(file_extensions.get(ext, []))
    print(f"{ext.upper()} files: {count}")

print("\nAnalysis complete!")
```

```
=== ANALYSIS SUMMARY ===
Total files scanned: 25
File extensions found: 10
.SDF files: 3
.XPT files: 3
.ASNT files: 3

Analysis complete!
```

Enhanced Comprehensive File Summary Table with Detailed Content Columns

In [7]:
```
# Create enhanced summary table with detailed content columns
import pandas as pd

# Helper functions to extract detailed file information
def analyze_sdf_file(file_path):
    """Analyze SDF file and return detailed information."""
    try:
        from rdkit import Chem
        from rdkit.Chem import Descriptors

        suppl = Chem.SDMolSupplier(str(file_path))
        molecules = [mol for mol in suppl if mol is not None]

        if molecules:
            mol = molecules[0]
            return {
                'Number of molecules': len(molecules),
                'First molecule': mol.GetProp('_Name') if mol.HasProp('_Name') else 'Unnamed',
                'Molecular weight': f"{Descriptors.MolWt(mol):.2f}",
                'Number of atoms': mol.GetNumAtoms(),
                'Number of bonds': mol.GetNumBonds(),
                'SMILES': Chem.MolToSmiles(mol)
            }
        else:
            return {
                'Number of molecules': 0,
                'First molecule': 'N/A',
                'Molecular weight': 'N/A',
                'Number of atoms': 'N/A',
                'Number of bonds': 'N/A',
                'SMILES': 'N/A'
            }
    except Exception as e:
        return {
```

```python
            'Number of molecules': 'Error',
            'First molecule': f'Error: {str(e)}',
            'Molecular weight': 'Error',
            'Number of atoms': 'Error',
            'Number of bonds': 'Error',
            'SMILES': 'Error'
        }

def analyze_xpt_file(file_path):
    """Analyze XPT file and return detailed information."""
    try:
        import pyreadstat

        df, meta = pyreadstat.read_xport(str(file_path))

        # Get first 5 rows as formatted string
        first_rows = df.head().to_string(index=True)

        # Get summary statistics
        stats = df.describe().to_string()

        return {
            'Shape': str(df.shape),
            'Columns': str(list(df.columns)),
            'First 5 rows': first_rows,
            'Summary statistics': stats
        }
    except Exception as e:
        return {
            'Shape': f'Error: {str(e)}',
            'Columns': 'Error',
            'First 5 rows': 'Error',
            'Summary statistics': 'Error'
        }

def analyze_asnt_file(file_path):
    """Analyze ASNT file and return detailed information."""
    try:
        from Bio import SeqIO
        import xml.etree.ElementTree as ET

        # Check if it's XML
        with open(file_path, 'r') as f:
            first_line = f.readline()

        if first_line.startswith('<?xml'):
            # It's XML, parse as XML
            tree = ET.parse(str(file_path))
            root = tree.getroot()
            records = [root]
        else:
            # Try as ASN.1 sequence
            records = list(SeqIO.parse(str(file_path), "genbank"))

        if records:
            record = records[0]
            if hasattr(record, 'id'):  # BioPython record
                return {
                    'Number of records': len(records),
                    'XML Root': 'N/A (ASN.1 format)',
                    'StudyID': 'N/A',
                    'Reviewer': 'N/A',
                    'AssessmentDate': 'N/A',
                    'Findings': 'N/A',
                    'Recommendation': 'N/A'
                }
            else:  # XML element
                # Extract XML data
                xml_data = {}
                for child in record:
                    xml_data[child.tag] = child.text or ''

                return {
                    'Number of records': len(records),
                    'XML Root': record.tag,
                    'StudyID': xml_data.get('StudyID', ''),
                    'Reviewer': xml_data.get('Reviewer', ''),
                    'AssessmentDate': xml_data.get('AssessmentDate', ''),
                    'Findings': xml_data.get('Findings', ''),
                    'Recommendation': xml_data.get('Recommendation', '')
                }
        else:
            return {
                'Number of records': 0,
                'XML Root': 'N/A',
                'StudyID': 'N/A',
                'Reviewer': 'N/A',
                'AssessmentDate': 'N/A',
                'Findings': 'N/A',
                'Recommendation': 'N/A'
            }
    except Exception as e:
        return {
            'Number of records': f'Error: {str(e)}',
            'XML Root': 'Error',
            'StudyID': 'Error',
            'Reviewer': 'Error',
            'AssessmentDate': 'Error',
            'Findings': 'Error',
            'Recommendation': 'Error'
        }

# Filter for only the three file types
target_extensions = ['.xpt', '.sdf', '.asnt']

# Create summary data with detailed columns
summary_data = []
for ext in target_extensions:
    if ext in file_extensions:
        files = file_extensions[ext]

        for file_path in files:
            file_name = file_path.name

            # Initialize row data
            row = {
                'File Name': file_name,
                'Extension': ext.upper(),
                'File Path': str(file_path)
            }

            # Add file-specific detailed columns
            if ext == '.sdf':
                sdf_info = analyze_sdf_file(file_path)
                row.update({
                    'Number of molecules': sdf_info['Number of molecules'],
                    'First molecule': sdf_info['First molecule'],
                    'Molecular weight': sdf_info['Molecular weight'],
                    'Number of atoms': sdf_info['Number of atoms'],
                    'Number of bonds': sdf_info['Number of bonds'],
                    'SMILES': sdf_info['SMILES']
                })
                # Add empty columns for other file types
                row.update({
                    'Shape': '',
                    'Columns': '',
                    'First 5 rows': '',
                    'Summary statistics': '',
                    'Number of records': '',
                    'XML Root': '',
                    'StudyID': '',
                    'Reviewer': '',
                    'AssessmentDate': '',
                    'Findings': '',
                    'Recommendation': ''
                })

            elif ext == '.xpt':
                xpt_info = analyze_xpt_file(file_path)
                row.update({
                    'Shape': xpt_info['Shape'],
                    'Columns': xpt_info['Columns'],
                    'First 5 rows': xpt_info['First 5 rows'],
                    'Summary statistics': xpt_info['Summary statistics']
                })
                # Add empty columns for other file types
                row.update({
                    'Number of molecules': '',
                    'First molecule': '',
                    'Molecular weight': '',
                    'Number of atoms': '',
                    'Number of bonds': '',
                    'SMILES': '',
                    'Number of records': '',
                    'XML Root': '',
                    'StudyID': '',
                    'Reviewer': '',
                    'AssessmentDate': '',
                    'Findings': '',
                    'Recommendation': ''
                })

            elif ext == '.asnt':
                asnt_info = analyze_asnt_file(file_path)
                row.update({
                    'Number of records': asnt_info['Number of records'],
                    'XML Root': asnt_info['XML Root'],
                    'StudyID': asnt_info['StudyID'],
                    'Reviewer': asnt_info['Reviewer'],
                    'AssessmentDate': asnt_info['AssessmentDate'],
                    'Findings': asnt_info['Findings'],
                    'Recommendation': asnt_info['Recommendation']
                })
                # Add empty columns for other file types
                row.update({
                    'Number of molecules': '',
                    'First molecule': '',
                    'Molecular weight': '',
                    'Number of atoms': '',
                    'Number of bonds': '',
                    'SMILES': '',
                    'Shape': '',
                    'Columns': '',
                    'First 5 rows': '',
                    'Summary statistics': ''
                })

            summary_data.append(row)
```

```python

# Create DataFrame and display
summary_df = pd.DataFrame(summary_data)
summary_df
```

```
[16:12:03] ERROR: Atom line too short: '   -0.0015    1.2095    0.0000 C' on line 5
[16:12:03] ERROR: moving to the beginning of the next molecule
```

Out[7]:

	File Name	Extension	File Path	Shape	Columns	First 5 rows	Summary statistics	Number of molecules	First molecule
0	lb.xpt	.XPT	m5/53-clin-stud-reports/study1234/datasets/dat...	(2, 5)	['STUDYID', 'USUBJID', 'LBTEST', 'LBSTRESN', '...	STUDYID USUBJID LBTEST LBSTRESN LBSTR...	LBSTRESN\ncount 2.00000\nmean 51.10...		
1	ae.xpt	.XPT	m5/53-clin-stud-reports/study1234/datasets/dat...	(2, 5)	['STUDYID', 'USUBJID', 'AETERM', 'AESEV', 'AER...	STUDYID USUBJID AETERM AESEV AE...	STUDYID USUBJID AETERM AESEV AER...		
2	dm.xpt	.XPT	m5/53-clin-stud-reports/study1234/datasets/dat...	(2, 4)	['STUDYID', 'USUBJID', 'AGE', 'SEX']	STUDYID USUBJID AGE SEX\n0 ABC123 SUBJ0...	AGE\ncount 2.000000\nmean 48....		
3	compound.sdf	.SDF	m5/53-clin-stud-reports/study1234/datasets/dat...					0	N/A
4	Structure2D_COMPOUND_CID_197365.sdf	.SDF	m5/53-clin-stud-reports/study1234/datasets/dat...					1	197365
5	Conformer3D_COMPOUND_CID_197366.sdf	.SDF	m5/53-clin-stud-reports/study1234/datasets/dat...					1	197366
6	Conformer3D_COMPOUND_CID_197366.asnt	.ASNT	m5/53-clin-stud-reports/study1234/datasets/dat...						
7	Structure2D_COMPOUND_CID_197365.asnt	.ASNT	m5/53-clin-stud-reports/study1234/datasets/dat...						
8	assessment.asnt	.ASNT	m5/53-clin-stud-reports/study1234/datasets/dat...						

	Molecular weight	Number of atoms	Number of bonds	SMILES	Number of records	XML Root	StudyID	Reviewer	AssessmentDate	Findings	Recommendation
0											
1											
2											
3	N/A	N/A	N/A	N/A							
4	699.99	44	46	CN1CCN(c2ccc3nc(-c4ccc5nc(CCCc6ccc(N(CCCl)CCCl...							
5	590.60	41	46	CN1CCN(c2ccc3nc(-c4ccc5nc(CCCc6ccc(N(CCCl)CCCl...							
6					0	N/A	N/A	N/A	N/A	N/A	N/A
7					0	N/A	N/A	N/A	N/A	N/A	N/A
8					1	AssessmentTemplate	STUDY1234	Dr. Smith	2025-11-05	\n	Approved for next review phase

Save Summary Table to CSV

In [8]:

```python
# Save the summary table to CSV
csv_filename = 'SD_Study-Data-files.csv'
summary_df.to_csv(csv_filename, index=False)
print(f"Summary table saved to {csv_filename}")
print(f"CSV file contains {len(summary_df)} rows and {len(summary_df.columns)} columns")
print(f"Columns: {list(summary_df.columns)}")
```

```
Summary table saved to SD_Study-Data-files.csv
CSV file contains 9 rows and 20 columns
Columns: ['File Name', 'Extension', 'File Path', 'Shape', 'Columns', 'First 5 rows', 'Summary statistics', 'Number of molecules', 'First molecule', 'Molecular weight', 'Number of atoms', 'Number of bonds', 'SMILES', 'Number of records', 'XML Root', 'StudyID', 'Reviewer', 'AssessmentDate', 'Findings', 'Recommend
ation']
```