1. Dataset Description and Data Cleaning Process

Dataset Description

The data used in this assignment comes from the CSMAR (China Stock Market & Accounting Research) database, covering Chinese A-share listed companies from 2003 to 2024. The raw dataset contains **64,096** company-year observations across **5,698** listed companies, with 16 variables in total.

The variables fall into two main categories:

- **Financial data**: Total Assets, Total Liabilities, Shareholders' Equity, Revenue, Net Income, Operating Cash Flow
- **Corporate governance data**: CEO-Chairman Duality, Board Size, Number of Independent Directors, Property Rights Nature (SOE indicator), Largest Shareholder Ownership Ratio, Institutional Investor Shareholding Ratio, etc.

The original column names in CSMAR are coded (e.g., `A001000000a` actually refers to Total Assets), so the first step is to rename them into meaningful English abbreviations.

Data Cleaning Steps

1. **Rename columns**: Convert CSMAR coded column names to readable English names, e.g., `A001000000a` → `Total_Assets`, `B002000000b` → `Net_Income`
2. **Check missing values**: Revenue has ~1,505 missing entries, Property Rights Nature has ~2,106, CEO-Chairman Duality has ~2,286, and other variables have a few hundred each
3. **Drop missing observations**: First remove rows with missing financial data (assets, liabilities, equity, net income, cash flow), then remove rows with missing governance variables
4. **Remove anomalies**: Drop rows with Total Assets ≤ 0 (2 rows) and Total Equity = 0 (2 rows) to avoid errors in logarithmic and division operations
5. **Sort**: Sort by stock code and year to correctly compute year-over-year revenue growth for each company
6. **Type conversion**: Convert binary variables (SOE, DUAL) to integer type

In [13]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')

# Load data
df = pd.read_csv('常用变量查询（年度）.csv', dtype={'Stkcd': str})
```

```
print(f"Raw dataset size: {df.shape[0]} rows, {df.shape[1]} columns")
print(f"Year range: {df['accper'].min()} - {df['accper'].max()}")
print(f"Number of companies: {df['Stkcd'].nunique()}")
print(f"\nOriginal columns: {list(df.columns)}")
df.head()
```

```
Raw dataset size: 64096 rows, 16 columns
Year range: 2003 - 2024
Number of companies: 5698

Original columns: ['Stkcd', 'accper', 'stknme', 'A001000000a', 'A002000000', 'A00300
0000', 'B001101000', 'B002000000b', 'C001000000', 'ConcurrentPosition', 'Boardsize
2', 'IndDirector', 'PropertyRightsNature', 'Shrcr1', 'Shrcr4', 'InsInvestorProp']
```

Out[13]:

	Stkcd	accper	stknme	A001000000a	A002000000	A003000000	B001101000
0	000001	2003	平安银行	1.928510e+11	1.888859e+11	3.965084e+09	NaN
1	000001	2004	平安银行	2.042864e+11	1.996018e+11	4.684662e+09	NaN
2	000001	2005	平安银行	2.292164e+11	2.241734e+11	5.043042e+09	NaN
3	000001	2006	平安银行	2.605763e+11	2.541018e+11	6.474463e+09	NaN
4	000001	2007	平安银行	3.525394e+11	3.395333e+11	1.300606e+10	NaN

In [14]:

```
# Check missing values for each column
print("Missing values per column:")
print(df.isnull().sum())
print(f"\nTotal rows: {len(df)}")
```

```
Missing values per column:
Stkcd                     0
accper                    0
stknme                    0
A001000000a             350
A002000000              350
A003000000              350
B001101000             1505
B002000000b             350
C001000000              350
ConcurrentPosition     2286
Boardsize2              573
IndDirector             592
PropertyRightsNature   2106
Shrcr1                  569
Shrcr4                  569
InsInvestorProp         613
dtype: int64

Total rows: 64096
```

In [15]:

```python
# Rename columns: convert CSMAR codes to readable English names
rename_dict = {
    'A001000000a': 'Total_Assets',            # Total Assets
    'A002000000':  'Total_Debt',              # Total Liabilities
    'A003000000':  'Total_Equity',            # Total Shareholders' Equity
    'B001101000':  'Revenue',                 # Revenue
    'B002000000b': 'Net_Income',              # Net Income
    'C001000000':  'Operating_Cash_Flow',     # Operating Cash Flow
    'ConcurrentPosition':  'CEO_Chairman',           # CEO-Chairman Duality
    'Boardsize2':          'Board_Size',              # Board Size
    'IndDirector':         'Independent_Directors',   # Number of Independent Direc
    'PropertyRightsNature':'SOE_raw',                 # Property Rights Nature (1=S
    'Shrcr1':              'H1_Shareholding',         # Largest Shareholder Ownersh
    'Shrcr4':              'Top4_Shareholding',       # Top 4 Shareholders Ownershi
    'InsInvestorProp':     'Institutional_Shareholding' # Institutional Investor Sh
}
df.rename(columns=rename_dict, inplace=True)

# Step 1: Drop rows with missing key financial variables
key_financial = ['Total_Assets', 'Total_Debt', 'Total_Equity', 'Net_Income', 'Opera
before = len(df)
df.dropna(subset=key_financial, inplace=True)
print(f"Dropped rows with missing financial data: {before - len(df)}")

# Step 2: Drop rows with missing governance variables
gov_cols = ['CEO_Chairman', 'Board_Size', 'Independent_Directors', 'SOE_raw',
            'H1_Shareholding', 'Institutional_Shareholding']
before2 = len(df)
df.dropna(subset=gov_cols, inplace=True)
print(f"Dropped rows with missing governance data: {before2 - len(df)}")

# Step 3: Drop rows with missing Revenue (needed for Growth calculation)
before3 = len(df)
df.dropna(subset=['Revenue'], inplace=True)
print(f"Dropped rows with missing Revenue: {before3 - len(df)}")

# Step 4: Remove rows where Total_Assets <= 0 (log would be undefined)
before4 = len(df)
df = df[df['Total_Assets'] > 0]
print(f"Dropped rows with Total_Assets <= 0: {before4 - len(df)}")

# Step 5: Remove rows where Total_Equity == 0 (avoid division by zero)
before5 = len(df)
df = df[df['Total_Equity'] != 0]
print(f"Dropped rows with Total_Equity == 0: {before5 - len(df)}")

# Sort by company code and year
df = df.sort_values(['Stkcd', 'accper']).reset_index(drop=True)

print(f"\nCleaned dataset: {len(df)} rows, {df['Stkcd'].nunique()} companies")
print(f"Year range: {df['accper'].min()} - {df['accper'].max()}")
```

```
Dropped rows with missing financial data: 350
Dropped rows with missing governance data: 3500
Dropped rows with missing Revenue: 767
Dropped rows with Total_Assets <= 0: 2
Dropped rows with Total_Equity == 0: 2

Cleaned dataset: 59475 rows, 5533 companies
Year range: 2003 - 2024
```

In [16]:

```python
# ============ Calculate Control Variables ============

# Size: Natural logarithm of Total Assets
df['Size'] = np.log(df['Total_Assets'])

# Lev (Leverage): Total Debt / Total Assets
df['Lev'] = df['Total_Debt'] / df['Total_Assets']

# ROA (Return on Assets): Net Income / Total Assets
df['ROA'] = df['Net_Income'] / df['Total_Assets']

# Growth: Year-over-year percentage change in Revenue (grouped by company)
df['Growth'] = df.groupby('Stkcd')['Revenue'].pct_change() * 100

# CashFlow: Operating Cash Flow / Total Assets
df['CashFlow'] = df['Operating_Cash_Flow'] / df['Total_Assets']

# SOE (State-Owned Enterprise): Binary variable from Property Rights Nature
df['SOE'] = df['SOE_raw'].astype(int)

# DUAL: CEO-Chairman Duality (1 = CEO also serves as Chairman)
df['DUAL'] = df['CEO_Chairman'].astype(int)

# H1: Largest Shareholder Ownership Ratio
df['H1'] = df['H1_Shareholding']

# INSTSH: Institutional Investor Shareholding Ratio
df['INSTSH'] = df['Institutional_Shareholding']

# INDP: Independent Directors Ratio = Independent Directors / Board Size
df['INDP'] = df['Independent_Directors'] / df['Board_Size']

# BSIZE: Board Size (number of board members)
df['BSIZE'] = df['Board_Size'].astype(int)

# Winsorize continuous variables at 1% and 99% to mitigate outlier effects
from scipy.stats.mstats import winsorize
for col in ['Size', 'Lev', 'ROA', 'Growth', 'CashFlow', 'H1', 'INSTSH', 'INDP']:
    mask = df[col].notna()
    df.loc[mask, col] = winsorize(df.loc[mask, col], limits=[0.01, 0.01])

print("All control variables calculated (continuous variables winsorized at 1%/99%)")
print("\nPreview (first 10 rows):")
df[['Stkcd', 'accper', 'stknme', 'Size', 'Lev', 'ROA', 'Growth', 'CashFlow',
    'SOE', 'DUAL', 'H1', 'INSTSH', 'INDP', 'BSIZE']].head(10)
```

```
All control variables calculated (continuous variables winsorized at 1%/99%).

Preview (first 10 rows):
```

Out[16]:

	Stkcd	accper	stknme	Size	Lev	ROA	Growth	CashFlow	SOE
0	000001	2009	平安银行	26.083276	0.965177	0.008558	NaN	0.054769	0
1	000002	2003	万科A	23.080438	0.549243	0.053583	NaN	-0.139985	1
2	000002	2004	万科A	23.466324	0.594163	0.058750	20.174821	0.067501	1
3	000002	2005	万科A	23.813962	0.609809	0.065178	37.714101	0.038351	1
4	000002	2006	万科A	24.604993	0.649418	0.044418	69.035524	-0.062343	1
5	000002	2007	万科A	25.329380	0.661125	0.053125	99.048592	-0.104279	1
6	000002	2008	万科A	25.504375	0.674441	0.038913	15.383308	-0.000286	1
7	000002	2009	万科A	25.647679	0.670017	0.046727	19.245893	0.067244	1
8	000002	2010	万科A	26.083276	0.746861	0.040993	3.749591	0.010375	1
9	000002	2011	万科A	26.083276	0.770997	0.039160	41.544662	0.011443	1

2. Descriptive Statistics of Control Variables

The table below shows the descriptive statistics for all computed control variables, including sample size (N), mean, standard deviation, minimum, quartiles, median, and maximum. This gives an overview of how these variables are distributed across our sample.

In [17]:

```python
# Descriptive statistics table
control_vars = ['Size', 'Lev', 'ROA', 'Growth', 'CashFlow', 'SOE', 'DUAL', 'H1', 'I

desc = df[control_vars].describe().T
desc = desc[['count', 'mean', 'std', 'min', '25%', '50%', '75%', 'max']]
desc.columns = ['N', 'Mean', 'Std', 'Min', 'P25', 'Median', 'P75', 'Max']
desc['N'] = desc['N'].astype(int)
desc = desc.round(4)

print("Descriptive Statistics for Control Variables:")
print("=" * 100)
desc
```

```
Descriptive Statistics for Control Variables:
================================================================================
================
```

Out[17]:

	N	Mean	Std	Min	P25	Median	P75	Max
Size	59475	22.0120	1.3111	19.3995	21.0844	21.8266	22.7432	26.0833
Lev	59475	0.4355	0.2173	0.0515	0.2632	0.4267	0.5906	1.0062
ROA	59475	0.0297	0.0732	-0.3381	0.0106	0.0345	0.0644	0.1976
Growth	53942	16.2989	47.5269	-67.1467	-4.9687	9.3052	25.9742	310.2971
CashFlow	59475	0.0452	0.0728	-0.1864	0.0061	0.0453	0.0869	0.2501
SOE	59475	0.3742	0.4839	0.0000	0.0000	0.0000	1.0000	1.0000
DUAL	59475	0.2824	0.4501	0.0000	0.0000	0.0000	1.0000	1.0000
H1	59475	34.6686	15.0492	8.9886	22.9731	32.1665	44.9285	74.5657
INSTSH	59475	44.4965	24.2633	0.4544	24.8853	46.6005	63.9129	90.7484
INDP	59475	0.3724	0.0528	0.2727	0.3333	0.3333	0.4286	0.5714
BSIZE	59475	8.5753	1.7700	2.0000	7.0000	9.0000	9.0000	19.0000

In [18]:
```
# Save processed data
df.to_csv('processed_data.csv', index=False)
print("Processed data saved to processed_data.csv")
print(f"Final dataset: {len(df)} observations, {df['Stkcd'].nunique()} companies")
```

```
Processed data saved to processed_data.csv
Final dataset: 59475 observations, 5533 companies
```

3. Additional Observations and Issues

Here are some things I noticed while working on this assignment:

Missing Data

The raw dataset has quite a few missing values. The most affected columns are **ConcurrentPosition** (2,286 missing), **PropertyRightsNature** (2,106 missing), and **Revenue** (1,505 missing). Other variables have a few hundred missing entries each. After cleaning, the dataset went from 64,096 rows down to **59,475 rows** covering **5,533 companies**.

Growth is a special case — since it is computed as the year-over-year percentage change in revenue, the first year of each company will always be NaN (no prior year for comparison). This is expected behavior, not a data error. That is why Growth has fewer observations (~53,942) compared to other variables.

Outlier Handling

A small number of companies had **Total Assets equal to zero or negative**, which would cause `-inf` when taking the logarithm and errors in division. I removed these anomalous observations (only 4 rows in total). Additionally, I applied **1% and 99% winsorization** to all continuous variables, which caps extreme values at the 1st and 99th percentiles. This is a standard practice in financial research.

Missing Variables in the Dataset

The assignment requires computing **MBR (Market-to-Book Ratio)**, which needs Market Capitalization data. However, this field is not available in our dataset. To compute it, one would need to download market value data separately from the CSMAR stock trading database and merge it in.

Similarly, **MSH (Management Shareholding Ratio)** is also not present in our dataset and would need to be obtained from a different CSMAR module. Therefore, these two variables could not be calculated.

Key Findings from Descriptive Statistics

- **Size**: Mean = 22.01, Std = 1.31, ranging from 19.40 to 26.08. The log-transformed distribution looks approximately normal.
- **Lev (Leverage)**: Mean = 0.44, meaning that on average about 44% of total assets are financed by debt — a moderate level.
- **ROA**: Mean = 0.03 (3%), which is relatively low overall. The median (0.035) is slightly higher than the mean, suggesting that some loss-making firms are pulling the average down.
- **Growth**: Mean = 16.3%, but with a very high standard deviation of 47.5%, indicating large variation across companies. The range spans from -67% to 310%.
- **SOE**: Mean = 0.37, indicating that about 37% of the sample firms are state-owned enterprises.
- **DUAL**: Mean = 0.28, meaning about 28% of firms have the CEO also serving as the Chairman of the Board.
- **H1 (Largest Shareholder)**: Mean = 34.7%, median = 32.2%, reflecting the common "one dominant shareholder" phenomenon in Chinese listed companies.
- **INSTSH (Institutional Investors)**: Mean = 44.5%, showing relatively high institutional investor participation in A-share markets.
- **INDP (Independent Director Ratio)**: Mean = 0.37, median = 0.33 (exactly 1/3), which is consistent with China's regulatory requirement that independent directors make up at least one-third of the board.
- **BSIZE (Board Size)**: Mean = 8.6, median = 9. Most companies have boards of 7 to 9 members.

Notes on Data Processing

1. When computing Growth, it is essential to **group by company** before applying `pct_change()`. Applying it directly to the entire column would mix different companies' data and produce incorrect results.
2. Although rare, rows with Total Assets ≤ 0 do exist and must be filtered out before taking the logarithm.
3. Winsorization effectively reduces the influence of extreme outliers on both descriptive statistics and subsequent regression analyses.