Import từ SQL sever:

//import dữ liệu từ sql sever vào spreadsheet

public void LoadDataToSpreadsheet(string connectionString, string[] tables)

{

MessageBox.Show("LoadDataToSpreadsheet(string connectionString, string[] tables)");

if (string.IsNullOrEmpty(connectionString))

{

MessageBox.Show("Không tìm thấy thông tin kết nối!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

using (SqlConnection connection = new SqlConnection(connectionString))

{

try

{

connection.Open();

IWorkbook workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM [{tableName}]";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

var sheet = GetOrCreateSheet(workbook, tableName);

for (int col = 0; col < dataTable.Columns.Count; col++)

{

sheet.Cells[0, col].Value = dataTable.Columns[col].ColumnName;

}

for (int row = 0; row < dataTable.Rows.Count; row++)

{

for (int col = 0; col < dataTable.Columns.Count; col++)

{

sheet.Cells[row + 1, col].Value = dataTable.Rows[row][col]?.ToString();

}

}

sheet.Columns.AutoFit(0, dataTable.Columns.Count);

}

//FormatTable();

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

workbook.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu LoadDataToSpreadsheet: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

Import từ Mysql:

public void LoadMySQLTablesToSpreadsheet(string connectionString, string[] tables)

{

MessageBox.Show("LoadMySQLTablesToSpreadsheet(string connectionString, string[] tables)");

using (MySqlConnection conn = new MySqlConnection(connectionString))

{

conn.Open();

var workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

try

{

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM `{tableName}`";

MySqlCommand cmd = new MySqlCommand(query, conn);

MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);

DataTable dt = new DataTable();

adapter.Fill(dt);

DevExpress.Spreadsheet.Worksheet sheet = workbook.Worksheets.FirstOrDefault(ws => ws.Name == tableName) ?? workbook.Worksheets.Add();

sheet.Name = tableName;

for (int col = 0; col < dt.Columns.Count; col++)

{

sheet.Cells[0, col].Value = dt.Columns[col].ColumnName;

}

for (int row = 0; row < dt.Rows.Count; row++)

{

for (int col = 0; col < dt.Columns.Count; col++)

{

sheet.Cells[row + 1, col].Value = dt.Rows[row][col]?.ToString();

}

}

}

}

finally

{

IWorkbook workbookFM = spreadsheetControl.Document;

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbookFM);

workbookFM.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

}

}

Import từ Excel:

private void importFromExcel(object sender, ItemClickEventArgs e)

{

XtraOpenFileDialog xtraOpenFileDialog = new XtraOpenFileDialog

{

Filter = "Excel Files|\*.xlsx;\*.xls",

Title = "Chọn file Excel để import"

};

if (xtraOpenFileDialog.ShowDialog() == DialogResult.OK)

{

string filePath = xtraOpenFileDialog.FileName;

spreadsheetControl.LoadDocument(filePath);

IWorkbook workbook = spreadsheetControl.Document;

foreach (DevExpress.Spreadsheet.Worksheet sheet in workbook.Worksheets)

{

sheet.Tag = new SheetSource

{

Type = SheetSource.SourceType.Excel,

SourcePathOrTableName = filePath

};

}

IWorkbook workbookFM = spreadsheetControl.Document;

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbookFM);

workbookFM.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

}

IMport flatfile:

private void ImportFlatFileToSpreadsheet(string filePath, string separator)

{

try

{

IWorkbook workbook = spreadsheetControl.Document;

DevExpress.Spreadsheet.Worksheet sheet = workbook.Worksheets.ActiveWorksheet;

workbook.BeginUpdate();

sheet.Cells.Clear();

string[] lines = File.ReadAllLines(filePath, Encoding.UTF8);

if (lines.Length == 0)

{

MessageBox.Show("File rỗng.");

return;

}

string[] headers = lines[0].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < headers.Length; col++)

{

sheet.Cells[0, col].Value = headers[col];

}

for (int row = 1; row < lines.Length; row++)

{

string[] values = lines[row].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < headers.Length; col++)

{

if (col < values.Length)

sheet.Cells[row, col].SetValue(values[col]);

}

}

sheet.Columns.AutoFit(0, headers.Length);

workbook.EndUpdate();

MessageBox.Show("Đã import dữ liệu thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi import file: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

**Sua lai:**

public enum SourceType { Database, Excel, CSV, Generated }

public class SheetSource

{

public SourceType Type { get; set; }

public string SourcePathOrTableName { get; set; } // Path hoặc tên bảng SQL

public string ConnectionString { get; set; } // Nếu là Database thì lưu thêm

}

private Dictionary<string, SheetSource> sheetSources = new(); // key = tên sheet

--------------

public void LoadDataToSpreadsheet(string connectionString, string[] tables)

{

if (string.IsNullOrEmpty(connectionString))

{

MessageBox.Show("Không tìm thấy thông tin kết nối!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

using SqlConnection connection = new(connectionString);

try

{

connection.Open();

IWorkbook workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM [{tableName}]";

SqlDataAdapter adapter = new(query, connection);

DataTable dataTable = new();

adapter.Fill(dataTable);

var sheet = GetOrCreateSheet(workbook, tableName);

for (int col = 0; col < dataTable.Columns.Count; col++)

sheet.Cells[0, col].Value = dataTable.Columns[col].ColumnName;

for (int row = 0; row < dataTable.Rows.Count; row++)

for (int col = 0; col < dataTable.Columns.Count; col++)

sheet.Cells[row + 1, col].Value = dataTable.Rows[row][col]?.ToString();

sheet.Columns.AutoFit(0, dataTable.Columns.Count);

sheet.Tag = new SheetSource

{

Type = SourceType.Database,

SourcePathOrTableName = tableName,

ConnectionString = connectionString

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

workbook.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu: " + ex.Message);

}

}

----------------------------------

public void LoadMySQLTablesToSpreadsheet(string connectionString, string[] tables)

{

using MySqlConnection conn = new(connectionString);

conn.Open();

var workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM `{tableName}`";

MySqlCommand cmd = new(query, conn);

MySqlDataAdapter adapter = new(cmd);

DataTable dt = new();

adapter.Fill(dt);

var sheet = GetOrCreateSheet(workbook, tableName);

for (int col = 0; col < dt.Columns.Count; col++)

sheet.Cells[0, col].Value = dt.Columns[col].ColumnName;

for (int row = 0; row < dt.Rows.Count; row++)

for (int col = 0; col < dt.Columns.Count; col++)

sheet.Cells[row + 1, col].Value = dt.Rows[row][col]?.ToString();

sheet.Columns.AutoFit(0, dt.Columns.Count);

sheet.Tag = new SheetSource

{

Type = SourceType.Database,

SourcePathOrTableName = tableName,

ConnectionString = connectionString

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

workbook.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

---------------

private void importFromExcel(object sender, ItemClickEventArgs e)

{

XtraOpenFileDialog dialog = new() { Filter = "Excel Files|\*.xlsx;\*.xls" };

if (dialog.ShowDialog() != DialogResult.OK) return;

string filePath = dialog.FileName;

spreadsheetControl.LoadDocument(filePath);

var workbook = spreadsheetControl.Document;

foreach (var sheet in workbook.Worksheets)

{

sheet.Tag = new SheetSource

{

Type = SourceType.Excel,

SourcePathOrTableName = filePath

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

UpdateSheetListToComboBox();

UpdateTreeView();

}

-------------------

private void ImportFlatFileToSpreadsheet(string filePath, string separator)

{

IWorkbook workbook = spreadsheetControl.Document;

var sheet = workbook.Worksheets.ActiveWorksheet;

workbook.BeginUpdate();

sheet.Cells.Clear();

string[] lines = File.ReadAllLines(filePath);

if (lines.Length == 0)

{

MessageBox.Show("File rỗng");

return;

}

string[] headers = lines[0].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < headers.Length; col++)

sheet.Cells[0, col].Value = headers[col];

for (int row = 1; row < lines.Length; row++)

{

string[] values = lines[row].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < values.Length && col < headers.Length; col++)

sheet.Cells[row, col].Value = values[col];

}

sheet.Columns.AutoFit(0, headers.Length);

workbook.EndUpdate();

sheet.Tag = new SheetSource

{

Type = SourceType.CSV,

SourcePathOrTableName = filePath

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

MessageBox.Show("Import thành công!");

}

---------------

public void SaveSheetBackToOriginalSource(Worksheet sheet)

{

if (!sheetSources.TryGetValue(sheet.Name, out var source))

{

SaveSheetToSQL(sheet); // Không có nguồn => mặc định lưu vào DB

return;

}

switch (source.Type)

{

case SourceType.Database:

SaveSheetToSQL(sheet, source);

break;

case SourceType.Excel:

spreadsheetControl.SaveDocument(source.SourcePathOrTableName);

break;

case SourceType.CSV:

SaveSheetToCSV(sheet, source.SourcePathOrTableName);

break;

default:

SaveSheetToSQL(sheet);

break;

}

}

---------------

private void SaveSheetToSQL(Worksheet sheet, SheetSource source = null)

{

// Lưu vào bảng mới nếu không có source hoặc tùy chỉnh với source

string tableName = source?.SourcePathOrTableName ?? $"Sheet\_{sheet.Name}\_{DateTime.Now:yyyyMMddHHmmss}";

string connectionString = source?.ConnectionString ?? DEFAULT\_CONNECTION\_STRING;

DataTable table = SpreadsheetToDataTable(sheet);

using SqlConnection conn = new(connectionString);

conn.Open();

using SqlCommand drop = new($"IF OBJECT\_ID('{tableName}', 'U') IS NOT NULL DROP TABLE [{tableName}];", conn);

drop.ExecuteNonQuery();

using SqlCommand create = new($"CREATE TABLE [{tableName}] (" +

string.Join(", ", table.Columns.Cast<DataColumn>().Select(c => $"[{c.ColumnName}] NVARCHAR(MAX)")) + ");", conn);

create.ExecuteNonQuery();

foreach (DataRow row in table.Rows)

{

string values = string.Join(", ", row.ItemArray.Select(v => $"N'{v.ToString().Replace("'", "''")}'"));

string query = $"INSERT INTO [{tableName}] VALUES ({values})";

using SqlCommand insert = new(query, conn);

insert.ExecuteNonQuery();

}

}

private void SaveSheetToCSV(Worksheet sheet, string filePath)

{

using StreamWriter writer = new(filePath, false, Encoding.UTF8);

int rowCount = sheet.GetUsedRange().RowCount;

int colCount = sheet.GetUsedRange().ColumnCount;

for (int row = 0; row < rowCount; row++)

{

List<string> values = new();

for (int col = 0; col < colCount; col++)

values.Add(sheet.Cells[row, col].Value?.ToString() ?? "");

writer.WriteLine(string.Join(",", values));

}

}

private DataTable SpreadsheetToDataTable(Worksheet sheet)

{

DataTable dt = new();

int colCount = sheet.GetUsedRange().ColumnCount;

int rowCount = sheet.GetUsedRange().RowCount;

for (int col = 0; col < colCount; col++)

dt.Columns.Add(sheet.Cells[0, col].Value.ToString());

for (int row = 1; row < rowCount; row++)

{

DataRow dr = dt.NewRow();

for (int col = 0; col < colCount; col++)

dr[col] = sheet.Cells[row, col].Value?.ToString();

dt.Rows.Add(dr);

}

return dt;

}

Sua tiep:

// SheetSource.cs

public class SheetSource

{

public enum SourceType { Database, Excel, CSV, Unknown }

public SourceType Type { get; set; }

public string SourcePathOrTableName { get; set; } // Table name for DB, file path for Excel/CSV

public string ConnectionString { get; set; } // Only for Database

}

// Add a Dictionary to track source of each sheet

private Dictionary<string, SheetSource> sheetSources = new Dictionary<string, SheetSource>();

// Updated LoadDataToSpreadsheet

public void LoadDataToSpreadsheet(string connectionString, string[] tables)

{

if (string.IsNullOrEmpty(connectionString))

{

MessageBox.Show("Không tìm thấy thông tin kết nối!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

using (SqlConnection connection = new SqlConnection(connectionString))

{

try

{

connection.Open();

IWorkbook workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM [{tableName}]";

SqlDataAdapter adapter = new SqlDataAdapter(query, connection);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

var sheet = GetOrCreateSheet(workbook, tableName);

for (int col = 0; col < dataTable.Columns.Count; col++)

sheet.Cells[0, col].Value = dataTable.Columns[col].ColumnName;

for (int row = 0; row < dataTable.Rows.Count; row++)

for (int col = 0; col < dataTable.Columns.Count; col++)

sheet.Cells[row + 1, col].Value = dataTable.Rows[row][col]?.ToString();

sheet.Columns.AutoFit(0, dataTable.Columns.Count);

sheet.Tag = new SheetSource

{

Type = SheetSource.SourceType.Database,

SourcePathOrTableName = tableName,

ConnectionString = connectionString

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

workbook.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu LoadDataToSpreadsheet: " + ex.Message);

}

}

}

// Updated LoadMySQLTablesToSpreadsheet

public void LoadMySQLTablesToSpreadsheet(string connectionString, string[] tables)

{

using (MySqlConnection conn = new MySqlConnection(connectionString))

{

conn.Open();

var workbook = spreadsheetControl.Document;

workbook.BeginUpdate();

try

{

foreach (string tableName in tables)

{

string query = $"SELECT \* FROM `{tableName}`";

MySqlCommand cmd = new MySqlCommand(query, conn);

MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);

DataTable dt = new DataTable();

adapter.Fill(dt);

DevExpress.Spreadsheet.Worksheet sheet = GetOrCreateSheet(workbook, tableName);

for (int col = 0; col < dt.Columns.Count; col++)

sheet.Cells[0, col].Value = dt.Columns[col].ColumnName;

for (int row = 0; row < dt.Rows.Count; row++)

for (int col = 0; col < dt.Columns.Count; col++)

sheet.Cells[row + 1, col].Value = dt.Rows[row][col]?.ToString();

sheet.Tag = new SheetSource

{

Type = SheetSource.SourceType.Database,

SourcePathOrTableName = tableName,

ConnectionString = connectionString

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

}

finally

{

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

workbook.EndUpdate();

UpdateSheetListToComboBox();

UpdateTreeView();

}

}

}

// Updated ImportFromExcel

private void importFromExcel(object sender, ItemClickEventArgs e)

{

XtraOpenFileDialog openFileDialog = new XtraOpenFileDialog

{

Filter = "Excel Files|\*.xlsx;\*.xls",

Title = "Chọn file Excel để import"

};

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

string filePath = openFileDialog.FileName;

spreadsheetControl.LoadDocument(filePath);

IWorkbook workbook = spreadsheetControl.Document;

foreach (var sheet in workbook.Worksheets)

{

sheet.Tag = new SheetSource

{

Type = SheetSource.SourceType.Excel,

SourcePathOrTableName = filePath

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

}

AdjustColumnWidth();

FormatHeaderRow();

MoveSheet1ToEnd(workbook);

UpdateSheetListToComboBox();

UpdateTreeView();

}

}

// Updated ImportFlatFileToSpreadsheet

private void ImportFlatFileToSpreadsheet(string filePath, string separator)

{

try

{

IWorkbook workbook = spreadsheetControl.Document;

DevExpress.Spreadsheet.Worksheet sheet = workbook.Worksheets.ActiveWorksheet;

workbook.BeginUpdate();

sheet.Cells.Clear();

string[] lines = File.ReadAllLines(filePath, Encoding.UTF8);

if (lines.Length == 0) return;

string[] headers = lines[0].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < headers.Length; col++)

sheet.Cells[0, col].Value = headers[col];

for (int row = 1; row < lines.Length; row++)

{

string[] values = lines[row].Split(new string[] { separator }, StringSplitOptions.None);

for (int col = 0; col < headers.Length; col++)

if (col < values.Length)

sheet.Cells[row, col].SetValue(values[col]);

}

sheet.Columns.AutoFit(0, headers.Length);

sheet.Tag = new SheetSource

{

Type = SheetSource.SourceType.CSV,

SourcePathOrTableName = filePath

};

sheetSources[sheet.Name] = (SheetSource)sheet.Tag;

workbook.EndUpdate();

MessageBox.Show("Đã import dữ liệu thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi import file: " + ex.Message);

}

}

Hàm đọc từ sheetsource:

public void SaveSheetToOriginalSource()

{

IWorkbook workbook = spreadsheetControl.Document;

foreach (Worksheet sheet in workbook.Worksheets)

{

if (sheet.IsEmpty) continue;

if (sheet.Tag is SheetSource source)

{

switch (source.Type)

{

case SheetSource.SourceType.Excel:

SaveSheetToExcel(sheet, source.SourcePathOrTableName);

break;

case SheetSource.SourceType.CSV:

SaveSheetToCSV(sheet, source.SourcePathOrTableName, ","); // Mặc định phân cách là dấu phẩy

break;

case SheetSource.SourceType.Database:

SaveSheetToDatabase(sheet, source.SourcePathOrTableName);

break;

default:

MessageBox.Show($"Không thể lưu sheet '{sheet.Name}': nguồn không xác định.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

break;

}

}

else

{

// Nếu không có thông tin nguồn => Lưu vào SQL Server với tên bảng là tên sheet

SaveSheetToDatabase(sheet, sheet.Name);

}

}

MessageBox.Show("Đã lưu tất cả sheet theo nguồn gốc thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void SaveSheetToExcel(Worksheet sheet, string filePath)

{

IWorkbook tempWorkbook = new Workbook();

Worksheet tempSheet = tempWorkbook.Worksheets[0];

tempSheet.CopyFrom(sheet);

tempSheet.Name = sheet.Name;

tempWorkbook.SaveDocument(filePath, DocumentFormat.Xlsx);

}

private void SaveSheetToCSV(Worksheet sheet, string filePath, string separator)

{

using (StreamWriter writer = new StreamWriter(filePath, false, Encoding.UTF8))

{

int rowCount = sheet.GetUsedRange().RowCount;

int colCount = sheet.GetUsedRange().ColumnCount;

for (int row = 0; row < rowCount; row++)

{

List<string> rowValues = new List<string>();

for (int col = 0; col < colCount; col++)

{

var value = sheet.Cells[row, col].Value.ToString();

value = value.Contains(separator) ? $"\"{value}\"" : value;

rowValues.Add(value);

}

writer.WriteLine(string.Join(separator, rowValues));

}

}

}

private void SaveSheetToDatabase(Worksheet sheet, string tableName)

{

if (string.IsNullOrEmpty(GlobalConnectionString))

{

MessageBox.Show("Không tìm thấy chuỗi kết nối!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

using (SqlConnection connection = new SqlConnection(GlobalConnectionString))

{

connection.Open();

SqlCommand command = connection.CreateCommand();

SqlTransaction transaction = connection.BeginTransaction();

command.Connection = connection;

command.Transaction = transaction;

try

{

// Tạo bảng mới

var usedRange = sheet.GetUsedRange();

int colCount = usedRange.ColumnCount;

int rowCount = usedRange.RowCount;

List<string> columnNames = new List<string>();

for (int col = 0; col < colCount; col++)

{

string colName = sheet.Cells[0, col].Value.ToString();

colName = string.IsNullOrWhiteSpace(colName) ? $"Column{col}" : colName;

columnNames.Add($"[{colName}] NVARCHAR(MAX)");

}

command.CommandText = $"IF OBJECT\_ID('{tableName}', 'U') IS NOT NULL DROP TABLE [{tableName}]; " +

$"CREATE TABLE [{tableName}] ({string.Join(",", columnNames)});";

command.ExecuteNonQuery();

// Insert dữ liệu

for (int row = 1; row < rowCount; row++)

{

List<string> values = new List<string>();

for (int col = 0; col < colCount; col++)

{

string value = sheet.Cells[row, col].Value.ToString().Replace("'", "''");

values.Add($"N'{value}'");

}

string insertCmd = $"INSERT INTO [{tableName}] VALUES ({string.Join(",", values)});";

command.CommandText = insertCmd;

command.ExecuteNonQuery();

}

transaction.Commit();

}

catch (Exception ex)

{

transaction.Rollback();

MessageBox.Show($"Lỗi khi lưu sheet '{sheet.Name}' vào database: {ex.Message}", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}