ODIP 4.3 修正パッチ（P1040306005845）リリースノート

2024/3/15

（株）インテリジェント・モデル

目　次

A. 変更内容

1. TRUNCATE 文のオプション化

(1) DBMS 設定ファイルのオプションの追加

DBMS 設定ファイル（<dbms>.properties）に下記のオプションを追加しました。出力がテーブルかつユーザビューのロードタイプが「全置換」のとき、sql.template.truncate で指定した SQL 文を使用してテーブルの全行を削除します。

オプション	説明
sql.template.truncate = [truncate statement with <tablename> variable]	テーブルの全行を削除する SQL 文のテンプレートを指定します。変数の<tablename>は、実行時にテーブル名に置換されます。このオプションの指定がない場合には、これまでのバージョンと同じ既定の構文を使用します。
sql.commit.before.truncate = [true|false]	テーブルの全行を削除する SQL 文を発行する直前に commit を行うか否かを true か false で指定します。DBMS によっては、特定の命令（例えば、DB2 for luw の TRUNCATE 文）は，トランザクション内の最初にだけ実行可能な場合があります。このオプションを true に設定すると、テーブルの全行を削除する SQL 文を、確実にトランザクション内で最初に実行することができます。

(2) db2.properties の変更

DB2 for luw の DBMS 設定ファイル（db2.properties）に、次の行を追加しました。この変更により、従来は DB2 のテーブルに対する全行削除は ALTER TABLE 文によって行っていましたが、この変更により、DB2 for luw 9.6 以降でサポートされる TRUNCATE TABLE 文によって行うようになります。従来のバージョンと同じ SQL を実行するためには sql.template.truncate に”ALTER TABLE <tablename> ACTIVATE NOT LOGGED INITIALLY WITH EMPTY TABLE”を指定してください。

sql.template.truncate = TRUNCATE TABLE <tablename> IMMEDIATE
sql.commit.before.truncate = true

2. ロード処理関連ファイル名の改定

ロード処理で ODIP が自動的に作成するデータファイル(.dat)、制御ファイル(.ctl/.fmt)、ログファイル（.log)、不良ファイル（.bad）それぞれのファイル名を変更しました。

(1) オプションの追加

以前のバージョンと互換性を保つために、次のオプションを設定ファイル（odip.ini）に追加しました。job.loader.filename の既定値は random です。tablename を指定すると、以前のバージョンと同じ動きになります。job.loader.delete.log.file の既定値は false です。

odip.ini の追加オプション

オプション	説明
job.loader.filename = [**random** | tablename]	DBMS が提供するローダを使用する際のデータファイル、制御ファイル、ログファイル、エラーファイルのファイル名の取得方法を指定します。このオプションは、以前のバージョンの動作を再現したい場合に使用します。
job.loader.delete.log.file = [true | **false**]	DBMS のローダによるロード処理後に、ロードログファイルを削除するか否かを指定します。 true を指定すると、ロード処理の正常終了後に、データファイル以外の DBMS のローダが出力するログファイル、不良ファイルを削除します。

(2) ロードファイル名の変更

① random

JavaVM の API で提供される、ランダムな一時ファイル名を使用します。一時ファイル名は、例えば"odip123456789012456.dat"のように"odip"で始まり、文字列とランダムな数字の組合せになります。一時ファイル名はロードファイルの出力先ディレクトリ内で一意性が保たれます。このオプションを指定することで、同じテーブルへのロード処理を同時に複数実行しても、ファイルの競合によるエラーは発生しません。同じテーブルへのロード処理が複数同時に実行される可能性がある場合、random を指定してください。

② tablename

例えば“A00000001_1.dat”のように、ロード先のテーブル名に日時、連番など必要な識別子を付加したファイル名を使用します。このオプションを指定すると、同じテーブルへのロード処理を同時に複数実行した場合に、タイミングによってはファイルの競合によるエラーが発生する場合があります。このオプションは、以前のバージョンとの互換性を維持したい場合に指定します。

(3) ログファイル名、不良ファイル名の変更

ロード用一時ファイルの変更に合わせて、ロード処理で作成されるログファイル、不良ファイルなどの、拡張子を除いたファイル名をデータファイルと同じ名前に変更しました。また、job.loader.delete.log.file を true にすることで、ロード処理の正常終了後にログファイル、不良ファイルなどを削除します。以前のバージョンでは、これらのファイル名は前回実行時のファイル名と同じになることが多く、実行時に上書きされていました。本改定によって、実行ごとにユニークなファイル名のファイルが作成されます。自動的に削除することで、ディレクトリ内にこれらのファイルが残存することを防ぐことができます。

3. Snowflake 対応の拡張

(1) execcopy コマンドの追加

管理単位を定義せずに、CSV ファイルから Snowflake のテーブルへデータをコピーするためのコマンド execcopy を、ODIP トランスフォーマに追加しました。execcopy は、コピー元データソース、コピー先データソース、データセットなどを指定して実行します。オプションの詳細、実行例はトランスフォーマリファレンスガイドを参照してください。

現在のバージョンでは、次の制限があります。

- CSV ファイルから Snowflake のテーブルへ、ローダ（snowsql COPY）を使用したコピーのみ対応しています。コピー元が CSV ファイル以外、コピー先が Snowflake のテーブル以外を指定すると実行時にエラーになります。
- コピー元にファイルパス（-from_file_path）を指定した場合は、カンマ区切り、文字列の囲み文字が””、エンコードが UTF-8、ヘッダなしの CSV ファイルを前提として実行します。
- コピー先データソースで、Snowflake の COPY コマンドでサポートされていない文字セットを指定するとエラーになります。
- コピー元 CSV ファイルのレイアウト（カラムの順序、カラム数）は、コピー先テーブルと完全に一致する必要があります。
- 出力先のテーブルが存在しない場合、データセット定義に従って CREATE TABLE

します。レイアウトがデータセットの定義と一致しているかのチェックは行いません。エラーになるかどうかは Snowflake のローダ（snowsql COPY）に依存します。

・コピー元ファイル名のまま Snowflake のステージへアップロードします。同一テーブルに対して同時に execcopy を実行する場合は、ユニークな名前に変更して実行してください。

(2) ファイルの分割機能の追加

Snowflake のテーブルへデータをロードする際に、自動的にファイルを分割した後にステージへのアップロードおよびロードを行うためのオプションを追加しました。この機能を有効にすることで、Snowflake へのデータ出力処理の時間短縮が見込めます。

① 追加オプション

ファイルを分割するために、snowflake.properties に次のオプションを追加しました。ファイルを分割する場合、次の何れかのオプションを有効にしてください。

オプション名	説明	既定値
loader.file.split.size	分割するファイルのサイズをバイト数で指定します。 KB、MB、GB を使用できます。 元ファイルのサイズが設定値よりも小さいとき、ファイルを分割せずに従来通りの 1 ファイルをロードします。	-1 (無効)
loader.file.split.lines	分割するファイルの行数を指定します。 元ファイルの行数が設定値よりも少ないときも、ファイルを分割するロジックが動き、新しく作られた 1 ファイルをロードします。	-1 (無効)
loader.file.split.number	2 以上の分割するファイルの数を指定します。分割するファイルのサイズを指定された数で割りますが、行末で分割するため、分割したファイルのサイズは必ずしも一致しません。	-1 (無効)

② 制限事項

ロードファイルの分割機能には、次の制限があります。

・ファイル分割の設定はジョブごと、出力テーブルごとの指定はできません。

・改行コードが LF または CRLF のみ分割できます。

(3) ファイルの圧縮機能の追加

Snowflake のテーブルへデータをロードする際に、自動的にファイルを圧縮した後にス

テージへのアップロードおよびロードを行うためのオプションを追加しました。この機能を有効にすることで、Snowflake へのデータ出力処理の時間短縮が見込めます。

① 追加オプション

ファイルを圧縮するために、snowflake.properties に次のオプションを追加しました。

オプション名	説明	既定値
loader.file.gzip	true を指定すると、データファイルを gzip コマンドでローカルファイルを圧縮した後にステージに転送、Snowflake へロードを行います。Windows 環境で実行するためには Linux コマンドの gzip と同等のコマンドを導入する必要があります。ODIP からは gzip <ファイル名>を呼び出します。	false

② 制限事項

・ファイル圧縮の設定はジョブごと、出力テーブルごとの指定はできません。
・ファイル分割なしで圧縮のみ有効な場合、execcopy を実行すると実行後は元ファイルは圧縮ファイルのみが残ります。
・圧縮は gzip コマンドを使用しているため、Unix/Linux のみ利用可能ですが、Unix/Linux の gzip と互換性のある gzip コマンドを導入することで Windows 環境でも動作可能です。（gzip コマンドは ODIP では提供していません）

(4) その他 Snowflake 関連の修正

① Snowflake のスキーマ名、テーブル名にワイルドカード文字「_」が含まれるとき、JDBC ドライバのメタデータ検索の応答が遅くなる場合がある問題を修正しました。

② 従来のバージョンでは、データを Snowflake にロードするには~/.snowsql/config、または実行ユーザの環境変数 SNOWSQL_PWD に Snowflake のパスワードを設定する必要がありましたが、出力データソースのパスワードを動的に設定するように変更しました。この改定で、loader.command の cmd /c set SNOWSQL_PWD=<pass>&のような指定も不要になります。

③ Snowflake 関連機能の改定に伴い、snowflake.properties のロード関連コマンドのテンプレートの一部を変更しました。

4. その他の修正

(1) JVM モードのジョブを複数同時に実行すると、タイミングによってジョブのキャンセルができなくなる問題を修正しました。このとき、showserver のジョブ一覧に対象のジョブが表示されなくなる問題も合わせて修正しました。

(2) COBOL 固定長ファイルへの出力で SUBSTRB 関数や CONCATB 関数などを使用する場合、データソース情報で設定したエンコーディングが使用されず、期待したバイト数での切り出しなどができない問題を修正しました。

B. バージョンアップによる影響

既存の定義への影響はありません。

C. パッチの適用方法

本パッチは、次の ODIP 製品に適用してください。

- ODIP アドミニストレータ v4.3
- ODIP オペレーションマネージャ v4.3
- ODIP リポジトリマネージャ v4.3
- ODIP プロセスマネージャ v4.3
- ODIP リポジトリサーバ v4.3
- ODIP トランスフォーマ v4.3

1. ライブラリファイル、設定ファイル、スクリプトファイルの更新

実行中の ODIP 製品を終了し、ODIP_P1040306005845 フォルダに格納されているライブラリファイル、設定ファイル、スクリプトファイルを、表 1 のファイルのコピー先に上書きコピーしてください。

表 1　ODIP_P1040306005845 のフォルダ構成及びファイルのコピー先

ODIP_P1040306005845		ファイルのコピー先
lib		
	ADM	ODIP アドミニストレータの lib フォルダ

		OPE		ODIP オペレーションマネージャの lib フォルダ
		RPM		ODIP リポジトリマネージャの lib フォルダ
		RPS		ODIP リポジトリサーバの lib フォルダ
		TFM		ODIP トランスフォーマの lib フォルダ
	config			
		ADM		
			jdbcsample	ODIP アドミニストレータの config 配下の jdbcsample フォルダ
		OPE		
			jdbcsample	ODIP オペレーションマネージャの config 配下の jdbcsample フォルダ
		RPM		
			jdbcsample	ODIP リポジトリマネージャの config 配下の jdbcsample フォルダ
		TFM		
			jdbcsample	ODIP トランスフォーマの config 配下の jdbcsample フォルダ
	bin			
		TFM		
			unix	ODIP トランスフォーマ(Unix/Linux)の bin フォルダ
			win	ODIP トランスフォーマ(Windows)の bin フォルダ

2. パッチ適用後の確認

パッチ適用後は、各製品を起動し、表 2 の確認方法に従って確認を行ってください。

表 2　パッチ適用後の確認方法

製品名	確認方法
ODIP アドミニストレータ	ヘルプメニューから“ODIP について”を選択し、選択されたすべてのビルド ID が 1040306005845 であることを確認してください。
ODIP オペレーションマネージャ	
ODIP リポジトリマネージャ	
ODIP プロセスマネージャ	
ODIP リポジトリサーバ	ODIP リポジトリマネージャのツールメニューから"ORMS サーバ情報"を選択し、表示されたすべてのビルド ID が 1040306005845 であることを確認してください。
ODIP トランスフォーマ	ODIP トランスフォーマを起動し、showserver コマンドを、オプションに“-info version”を指定して実行してください。表示されたすべてのビルド ID が 1040306005845 であることを確認してください。

以　上