入出力データを用いた バッチ処理プログラムから SQL への変換

竹之内 啓太[1,a)]　岡田 譲二[1,b)]　坂田 祐司[1,c)]

概要：長年のあいだ保守開発を続けているシステムは巨大化・複雑化し，企業のビジネス成長の足かせとなっている．本稿では，レガシー化したシステムの保守性向上を目的として，COBOL 等で書かれたバッチ処理プログラムを SQL に変換するアプローチを提案する．具体的には，既存のバッチ処理プログラムから入出力データを抽出し，それを入出力例として SQL の合成を行う．元のプログラムの冗長な構造に依存することなく，簡潔な表現をもつ SQL へ変換できるのが特長である．

1. はじめに

バッチ処理プログラムとは，一定期間ごとに大量のデータを一括に計算するバッチ処理を実行するためのシステムである．企業活動を支えるソフトウェアシステムの中でも，業務における売り上げの集計や給与の計算といった基幹業務を果たすものであり，その多くは数十年前に開発されてから現在まで保守され続け，社会的に大きな役割を担っている．一方で，バッチ処理プログラムのレガシー化が企業活動の重要な課題となっている．具体的には，COBOL やメインフレームといったレガシーな技術に依存していることや，長年にわたる保守開発によりシステムが巨大化していることにより，運用や保守に必要なコストが増大し，企業のビジネス成長の足かせとなっている．

これらの問題を解決するため，レガシー化したシステムを現代的な技術を用いて作り直すのがレガシーモダナイゼーションである．モダナイゼーションの手段として，プログラミング言語の変換は一般的に行われるアプローチである．たとえば，COBOL で書かれたバッチ処理プログラムを Java を用いたプログラムに変換することで，メインフレームに依存していた実行環境を刷新することができる．このさい，COBOL の 1 命令 (たとえば代入文) を 1 つの Java の文へ変換するストレートコンバージョンなどが主に用いられる．しかしながら，この変換によって得られたプログラムは COBOL の冗長な構造を保ったままであり，保守性を改善することにはつながりにくい．

[1] 株式会社 NTT データ
NTT DATA Corporation
[a)] Keita.Takenouchi@nttdata.com
[b)] Joji.Okada@nttdata.com
[c)] Yuji.Sakata@nttdata.com

効果的なモダナイゼーション手法の確立に向けて，我々の研究チームではレガシーなバッチ処理の分析を実施した [1]．バッチ処理は，帳票などのテーブル構造をもつファイルを読み書きするものであるが，プログラムが長大である場合であっても，テーブルの結合や，条件による行のフィルタリングなど，SQL で記述すると簡潔に表現される処理が多くを占めることが明らかとなった．

そこで本稿では，レガシーなバッチ処理プログラムのモダナイゼーションの手法として，入出力データのみを用いて SQL へ変換するアプローチを提案する．具体的には，レガシーなバッチ処理の入出力となるテーブルをリレーショナルデータベースのテーブルと見なし，その入力テーブルから出力テーブルのデータを作り出す SQL を合成することで変換を行う．変換後は SQL で書かれたバッチ処理を保守すればよく，保守コストの大幅な削減が見込まれる．

関連研究として，手続き型のソースコードを解析することで SQL で表現可能な部分を抽出する手法が存在する [2]．しかしながら，本アプローチのように，ソースコードそのものを解析することなくプログラムの変換を行うやり方は我々の調査の限りでは存在しない．

2. 提案するアプローチ

提案するアプローチは，以下の 2 つの手順からなる．

手順 1．既存のレガシーなバッチ処理プログラムから入出力データを抽出する．

手順 2．抽出した入出力データを入出力例として，それを実現する SQL を合成する．

「手順 1」に関しては，対象となるプログラムの単体試験データ（入力データとそれに対する期待値）をそのまま

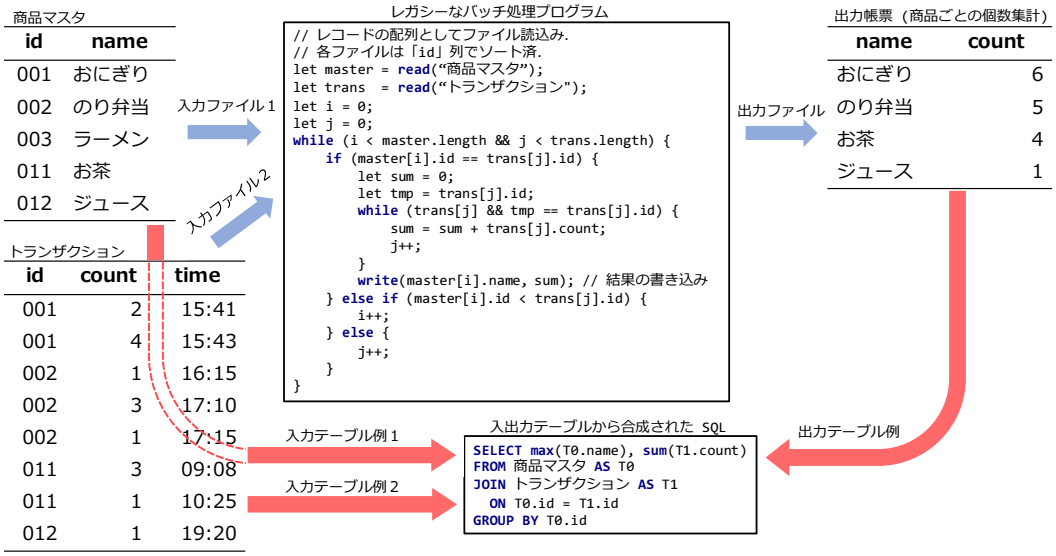

図 1　提案するアプローチの具体例．冗長なバッチ処理プログラムを簡潔な SQL へ変換する．

利用できる．試験データが存在しない場合でも，入力となるデータさえ作成できれば，実際にプログラムを実行することで出力データが得られる．「手順 2」に関しては，我々が開発した SQL 合成の手法 [3] 等が利用できる．この手法は，入出力テーブルの例からそれを実現する SQL の SELECT 文を合成する手法であり，レガシーなバッチ処理プログラムのテーブルの規模に対応しているのが特長である．この合成が成功した場合，元の冗長なプログラムから簡潔な SQL への変換が実現する．

図 1 の例を用いて手法の流れを具体的に説明する．この例は，2 つのテーブルを読み込み，その集計結果を新たなテーブルに書き込むバッチ処理である．ただし，このバッチ処理は，本来 COBOL 等のレガシーな言語で書かれるプログラムであるが，説明のため JavaScript 風の言語を用いて記述している．このプログラムは「商品マスタ」と「トランザクション」の 2 つのテーブルを読み込み，それぞれのテーブルがもつキーとなる列 (id 列) を一致させながら集計処理を行い，その結果を「出力帳票」テーブルに書き込む．このような結合処理はレガシーなシステムの開発において「マッチング」と呼ばれる一般的なものであり，図のようなループ構造を用いて実装される．

本アプローチでは，まずこのバッチ処理プログラムの 2 つの入力テーブル「商品マスタ」「トランザクション」にくわえて，出力テーブル「出力帳票」を抽出する．そして，これらのテーブルを入出力例として SQL の合成を行う．その結果，ループを用いて行われていた結合や集計処理が，SQL の JOIN 句や GROUP BY 句へ変換される．この SQL は元のプログラムの冗長な構造に依存しない簡潔な表現をもち，保守性が高いものである．

3. おわりに

本稿では，レガシーなバッチ処理プログラムの入出力データのみを用いて SQL へ変換するアプローチを提案した．理解が困難な長大な既存ソースコードを分析する必要がないうえ，元の構造に依存しない簡潔な表現をもつ SQL へ変換できるのが特長である．

一方で，いくつかの検討課題が残っている．たとえば，レガシーなバッチ処理には，文字列の結合や数値計算など値そのものを変更するようなものが存在する．このような処理は SQL の SELECT 文のみで表現することは難しく，さらに表現力の高いプログラムの合成技術が必要となる．また，変換後のプログラムの形は使用する入出力データにのみ依存するため，変換前後のプログラムの同値性が担保されるわけではない．元のプログラムの性質を網羅的に表現するような入出力データを用いることが望ましいが，必ずしもそれが容易であるとは限らない．

参考文献

[1] Joji, O., Takashi, I., Yuji, S. and Katsuro, I.: Towards Classification of Loop Idioms Automatically Extracted from Legacy Systems, *13th International Workshop on Software Clones*, pp. 34–35 (2019).
[2] Emani, K. V., Ramachandra, K., Bhattacharya, S. and Sudarshan, S.: Extracting Equivalent SQL from Imperative Code in Database Applications, *Proceedings of the 2016 International Conference on Management of Data*, pp. 1781–1796 (2016).
[3] 竹之内啓太，岡田譲二，坂田祐司：カラム数の大きなテーブルに対する入出力例からの SQL クエリの合成，ソフトウェアエンジニアリングシンポジウム (2019).