第 25 章　データベースとファイルの設計

コンピュータ処理のきっかけ

私がソフトウェア技術者として仕事を始めてしばらくたった頃、コンピュータ処理のきっかけは 2 つといわれていた。それらを、「外部イベント」と「時間的イベント」と呼んだ。

外部イベントとは、「顧客が商品を購入する」というような情報システムの外側で何らかの「出来事」があり、それがトランザクションを発生させ、そのトランザクションがコンピュータ処理の対象になる、というものである。この場合のコンピュータ処理は、オンライン・リアルタイム処理が向いている[MCM84]。

もう 1 つの時間的イベントとは、「今日の閉店時間が来た」、「月末が来た」、「決算日になった」というような、あらかじめ決められた時間の到来をいう。これをきっかけに行うコンピュータ処理は、一日の営業の成果を締める、月間の成果を締める、決算処理を行う、というようなもので、バッチ処理が向いている[MCM84]。

コンピュータの処理能力が著しく向上し、IoT で集めたビッグデータが分析の対象になる時代になって、つまり外部イベントなどで発生したデータをコンピュータが容易に蓄積し、分析できる時代になって、もう 1 つ社内の担当者がコンピュータを動かして分析の結果を見たくなるような事態が生じて、今やもう 1 つの処理のきっかけが生まれている。それを、「内部イベント」と呼んでおこう。内部イベントの処理も、できることならオンライン処理が望ましい。

ここで、外部イベントはオンライン処理、時間的イベントはバッチ処理と決めつけたが、常にそうであるとは限らない。例えば、証券取引所は今のところ平日の午後 3 時にその日の取引を終える。つまりこれは時間的イベントであるが、顧客からの注文を受け付け、証券取引所に流している証券会社のオンライン情報システムはこの時間的イベントをオンラインで処理しなければならない。

以前ある会社で、経理処理の一部をオンラインで処理をして、今期になってから問い合わせをした時点まででいくら利益が上がっているかを表示するシステムを見たことがある。経理処理はバッチ処理向けのシステムの典型だが、その原則を変えて効果を上げていた。

このように処理の方法は、必要に応じて変更できる。しかし給与計算はバッチ処理のままであろうし、決算処理の大部分もバッチ作業で処理されるが普通だろう。ここでいいたかったことは、いくらコンピュータの処理能力が増大してもバッチ処理は無くならない、ということである。

データベースの設計

データベースの設計の方法については、以下で述べる「CRUD 図」作成以外は第 16 章で述べたので、ここではそれを繰り返さない。ある情報システムで 1 つしっかりとしたデータベースを構築し、全ての必要なデータをここに集約する場合、オンライン処理もバッチ処理もひっくるめてその情報システムで出力される全てのデータ項目をそのデータベースが管理する形にするのが良い。そのデータベースは第 16 章で述べた方法で設計することができる。ただしここでいう出力とは帳票や画面だけを指すのでは無く、ほかの情報システムに渡すメッセージやファイル等を含むことに留意してほしい。

第 16 章でも述べたことだが、ここで 1 つあえて繰り返しておきたいことがある。それは、データベースの構築に当たっては、「データの標準化」をしっかりと行っておく必要があるということである。データの標準化とは、「1 つのデータに 1 つの名前を付ける」ことである。1 つ

のデータに複数の名前が付けられていたり、2 つ以上のデータに同じ名前が付いていたりしてはいけない。コーポレート・データベースを構築する場合には、このデータン標準化の範囲は全社に及ぶことになる。

「CRUD 図」の作成

「CRUD 図」とはトランザクションごとに、データベースのレコードの「作成：Create（C）」、「参照：Read（R）」、「更新：Update（U）」、「削除：Delete（D）」のどの処理を行うかを対比させた図である。この図で、トランザクションとデータベースのレコードに対する処理の関係が明確になる。

この図を書くことで、時には「データベースのレコードを抹消するきっかけがない」というような設計ミスが明らかになったりすることがある。

バッチ処理のファイルの設計

バッチ処理とは、あるデータベースの全てのデータやある目的に沿って抽出された一部のデータ、一定時間（例えば、今日の開店から閉店まで）に入力されたトランザクションを全て蓄積したもの、などを処理の対象にし、ほかのマスターファイルから情報を取り込んだり、分類したり、必要な計算をしたりして、最終的には帳票などを作成する作業をいう。

ここではデータがファイルに格納され、バケツリレーのようにそれがプログラムからプログラムへと順次受け渡しをされて処理が進む。

この場合のこのファイルは、最終の出力を作成するための必要かつ充分なデータがあれば良い。先読み後書きができる順次ファイルがバッチ処理に向いているが、その中の構成等に特別のルールは無い。ただバッチ処理では分類（ソート）が不可欠なので、分類で使用されるソートキィはレコードの最初の部分に置くのが基本である。

このバッチ処理用のファイルは、データベースとは違って他の情報システムとの共有などは考えない方が良い。特定の出力のための専用のファイルの方が扱いやすい。

キィワード

外部イベント、トランザクション、時間的イベント、オンライン・リアルタイム処理、バッチ処理、内部イベント、データベース、データの標準化、CRUD 図、ファイル、分類、ソートキィ

略語

IoT：Internet of Things

参考文献とリンク先

[MCM84] Stephen McMenamin, John F. Palmar, “Essential Systems Analysis,” Addison Wesley, 1984.

（2016 年（平成 28 年）4 月 28 日　新規作成）
（2017 年（平成 29 年）1 月 18 日　一部追加）