

企業情報システムにおける非正規化構造データベースの優位性の証明

Supremacy of Unnormalised Database Designing on Enterprise Information System.

堀田正隆
Masataka Hotta

株式会社プロ・アクシス
ProAxis Ltd.

要旨

データベースを設計するに当たって、正規化すべきか、非正規化で良いかは情報システムを作る人たちの間で長らく問題になってきた。中々結論は出ず、理論データベースは正規化でなければならないが物理データベースは非正規化も許すという折衷策のような形でこれまで進行してきた。しかし、物理データベースについて言えば非正規化も「許す」ではなく、非正規化こそ「正しい」と言える。このことを実例を挙げて証明し、これまでの議論に決着をつける。但し、どうしても非正規化してはならない場合があるので、その境界はどこにあるか、その境界も明らかにする。なお、本論はリレーショナル・データベースを前提としている。

1. 正規化構造と非正規化構造とその他の構造

まず、正規化構造と非正規化構造とその他の構造とはどのようなものであるか確かめておくことにする。図1は発注伝票を表す。この発注伝票は、3行の明細があり納期は同じ日であることを表している。この伝票の内容をデータベースに記録する形式は3通りある。図2は二つのファイルで図1の伝票の内容を表している。納期は、伝票全体を通して一つなので図2の上のファイルが代表している。この構造を正規化構造と言う。図3はそれぞれの明細に同じ納期が書かれている。このように無駄に同じデータが書きこまれる構造を非正規化構造と言う。図4はヘッダー部のデータが先ずあって、その後明細が繰り返えされている。これがその他の構造になる。

メーカー名	発注日	納期
日本鋳化	20050612	20050625

メーカー名	発注日	原料名	数量
日本鋳化	20050612	PP-4028N	40
日本鋳化	20050612	SXCY50台湾	25
日本鋳化	20050612	ABS-VW-10	60

図2 正規化構造

メーカー名	: 日本鋳化
発注日	: 2005/6/12
納期	: 2005/6/25

原料名	数量
PP-4028N	40
SXCY50台湾	25
ABS-VW-10	60

図1 発注伝票の事例

メーカー名	発注日	納期	原料名	数量
日本鋳化	20050612	20050625	PP-4028N	40
日本鋳化	20050612	20050625	SXCY50台湾	25
日本鋳化	20050612	20050625	ABS-VW-10	60

図3 非正規化構造

メーカー名	発注日	納期	原料名	数量	原料名	数量	原料名	数量
日本鋳化	20050612	20050625	PP-4028N	40	SXCY50台湾	25	ABS-VW-10	60

図4 その他の構造

図4はデータ量としては他の二つより小さくなるが、明細行の数に制約があることとオーソドックスなSQL文でデータの読み書きができないという理由からシーケンシャルファイルの名残でよく見かけるが構造としては良くない。

2. 正規化構造が推奨されてきた理由

無駄なデータがデータベースに記録されないこととファイルの定義がそのままビジネス・ルールを表すという利点があって正規化構造が推奨されてきた。ファイルの定義がビジネス・ルールを表すというのは、図2のように正規化されていると、納期は伝票毎に決められており、数量は一つの伝票の中の原料毎に決められるというビジネス・ルールが分かるという意味である。

3. 正規化構造の問題点

図5の発注伝票の事例は図1と異なる形式になっている。図1は伝票毎に納期を決めているのに対し、図5はそれぞれの明細毎に納期を決めている。図5の伝票のデータを図2の正規化構造のデータベースに記録することはできない。ビジネス・ルールは常に変わるものであり、また、同じ会社でも異なる事業間では異なることが多い。例えば、同じ会社でA事業部は図2の発注伝票で、B事業部は図5の発注伝票であることは希なことではない。すなわち、ファイルの定義がそのままビジネス・ルールを表している正規化構造の利点は、将来の変更可能性や異なる事業間での一元性という基準で評価すれば逆に欠点となっている。

メーカー名	ピグメ化成	
発注日	2005/6/12	
原料名	納期	数量
ビルコン	2005/6/25	40
POMデルリン	2005/6/30	20
PETベース	2005/6/30	25

図5 発注伝票の事例

4. 非正規化構造による問題の解決

初めから図3のような非正規化の構造であれば、図2の伝票でも図5の伝票でもデータベース更新のアプリケーション少しを変えるだけでどちらでも対応可能である。非正規化構造の利点はこのような懐の深さにある。

次に図6は、使用する原料の基本情報を記録した原料マスターの事例を示している。図6の構造から規格番号（バージョンのようなもの）と納品時の梱包単位は原料毎に決まっていることが読み取れる。一方、単価とリードタイムは同じ原料でも仕入先（メーカー名）毎に異なるものであることが理解できる。

原料名	規格番号	梱包単位
PP-4028N	1	12
SXCY50台湾	2	50
ABS-VW-10	1	6

原料名	メーカー名	単価	リードタイム
PP-4028N	日本鋳化	980	10
PP-4028N	ピグメ化成	1100	5
PP-4028N	新日本ベスト	965	12
SXCY50台湾	日本鋳化	470	7
ABS-VW-10	ピグメ化成	2850	20

図6 正規化構造による原料マスター

ここで生産上の都合からある一仕入先に限って梱包単位に特例を設けたいとする。実際の業務では決して珍しくないことだが、図6の構造をもってしては不可能である。実務を行う者にとってこのような情報システムの硬直性は大変迷惑なものと感ずるものである。

図7は非正規化構造にした原料マスターである。同じ原料であれば、規格番号と梱包単位に同じデータが並ぶ無駄な構造になっている。しかし、先に述べた一仕入先に限って梱包単位に特例を設けたいという要望が受けられる構造になっている。点線の丸で囲んだ箇所はそのことを示している（本来は12でなければならぬところが2になっている）。

原料名	メーカー名	規格番号	梱包単位	単価	リードタイム
PP-4028N	日本鋳化	1	12	980	10
PP-4028N	新日本ベスト	1	12	965	12
PP-4028N	ピグメ化成	1	2	1100	5
SXCY50台湾	日本鋳化	2	50	470	7
ABS-VW-10	ピグメ化成	1	6	2850	20

図7 非正規化構造による原料マスター

以上述べた通り、非正規化構造のデータベースは正規化構造のデータベースに比し、多くの点で優っている。纏めると、

- 1) 将来ビジネス・ルールの変更があっても対応可能
- 2) 同じ企業においてビジネス・ルールが異なっても一元化可能
- 3) 特例を設定可能

5. 非正規化の限界

どんな場合でも非正規化にしてしまえるかというそうではない。図8は非正規化の基本構造を示している。常に含む、含まれるの関係が成立してなければならない。図9は図7の非正規化の原料マスターの構造の模式図である。図8に示した非正規化の基本構造は保たれている。

図10は図7の原料マスターに検査項目名と域値を加えた模式図である。この場合は図8の基本構造から外れている。従って、分割されなければならない。このような場合を非正規化の限界という。この判定は一々模式図を書かずとも機械的に行うことできる。

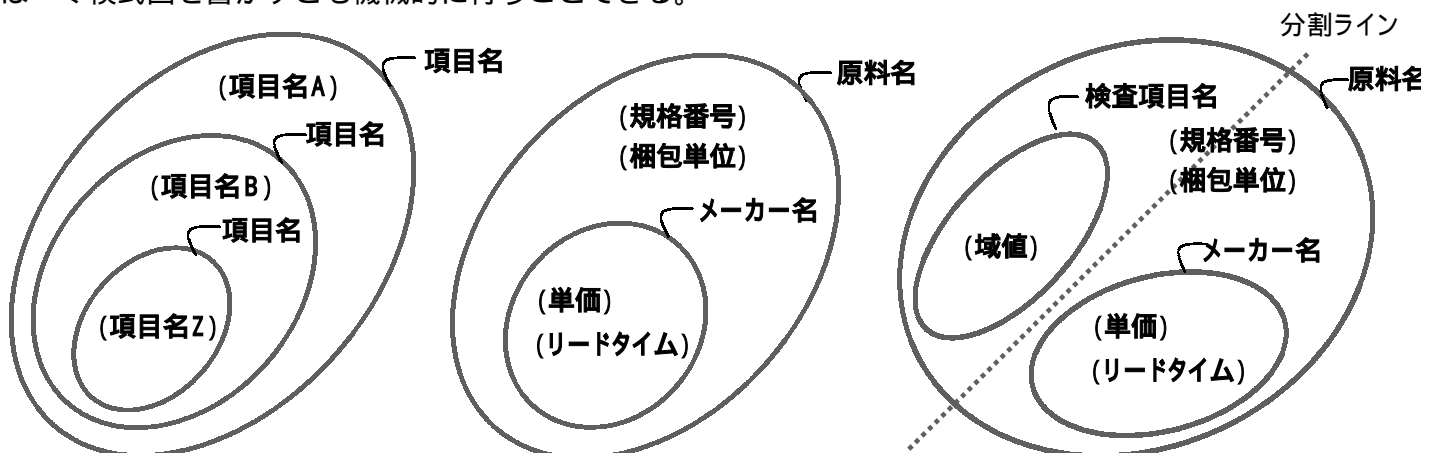


図8 非正規化の基本構造

図9 図7の模式図

図10 非正規化の基本構造に反する構造

非正規化の限界にはもう一例ある。加減算可能な量を表すデータ項目が最下位層以外の層にある場合、非正規化の限界に達している。最下位層とは図7で言えば項目名Zのことで、従って、項目名AまたはBが量を表すデータ項目である時、非正規化の限界に達している。

6. データの拡散の問題

非正規化することで同じデータが複数のファイルに拡散されることが懸念されてきた。ここで先ずデータの拡散の意味を確かめておくことにする。データの拡散の意味には、データの重複記録という意味と同じ事実が多くつのファイルに書かれるという意味の二つが考えられる。前者の弊害は記憶量の肥大（冗長）であり、後者の弊害は情報システム複雑化である。この弊害の違いで二者の違いが分かるはずである。

もし、拡散を前者の意味とするならば非正規化により確実にそうなる。後者の意味とするならば非正規化によってもたらされるものではない。同じ事実を複数のファイルに拡散させないことを厳格に行うということ（One Fact One Placeと言う）は非正規化においても実現可能である。この詳しいことについては別のテーマで改めて取り上げることにするが、これまで取り上げた事例を見る限り事実の拡散は起こっていないはずである。

7. データの更新時間に関する問題

非正規化によりデータの肥大（冗長）が起こることにより、データの更新時間に悪い影響を及ぼすことも懸念されることである。図2と図3に戻って、図2は1レコードの更新で済む。図3は3レコードの更新が必要になる。場合によっては1000レコードの更新となることが考えられる。例えそうであっても、基本的には図2の更新時間と図3の更新時間の間に大きな差はないと考えられる。図3で更新される一連のレコード群は物理的にほとんど同じところに配置されていると考えられるからである。

8. まとめ（柔らかい手法、固い手法）

ビジネス・ルールをデータベースの構造で表そうとする考え方（データオリエンテッド）は一見スマートに見えるが何か変更が起きると立ちどころに行き詰まってしまう脆さを抱えている。非正規化構造は、ビジネス・ルールをデータベース構造に反映させようという発想自体がないので、アンチデータオリエンテッドあると言える。しかし、少々ビジネス・ルールが変わろうと影響を受けない懐の深さを備えている。

過去を振り返ると約20年前にソフトウェア・エンジニアリングは重要な分岐点にさしかかっていた。この時の焦眉の課題は、システム構築過程であまりにも要求仕様の変更が多く、これによる非効率を何とかしなければならぬということであった。この時、ソフトウェア・エンジニアリングとして2つの選択肢があった。一つは変更を容易に可能にするエンジニアリングを確立する道（柔らかい手法）。もう一つは十分な分析をして変更を少なくするエンジニアリングを確立する道（固い手法）であった。

そして、ソフトウェア・エンジニアリングは後者の固い手法を選んで突き進んだ。この線上に正規化論もある。しかし、今もって要求定義と現状分析の良い方法とか現状分析が良くできる人への要求度が高いことは何を意味するのか。固い手法の難しさはなまじのものでなかったのである。本論には20年前の固い手法の選択が誤りであったとの主張が根底にある。どんなに困難に見えても柔らかい手法を確立して乗りきるべきであった。

現在、情報システムを構築する上で時間がかかるのは、将来も変わることはないかという検証と複数の部門間で異なることの多いビジネス・ルールの統合である。こういうことに時間を使うのは不毛である。将来のことは分からない、また違うものは違うままで良いと割り切って考えるべきである。そのための初歩として、データベースは正規化構造より懐の深い非正規化構造にしておくのが正しい選択である。データ構造は変えずにアプリケーションのバリエーションで乗りきろうと考え方を変えることが必要である。