達人に学ぶDB設計徹底指南書

木本ニッキー

2024/06/18

システム開発部

## 

## 1.この本を読んで

　私はこの本を読んで、論理設計と物理設計のトレードオフを様々な観点から考察される望ましい論理設計・物理設計について学び、目指すべきデータベース設計を行うエンジニアは、バランスの良い、正しいトレードオフを導き出せるようになることであると考えた。全体的な内容は、データベースやその設計に関する基礎知識、現場に求められるデータベースについてであるが、本書で取り扱う疑問の多くは「プログラムは動作するが、二択のうちどちらの方法が望ましいか」というものが大体である。そしてその答えについて明確な定義はなく、筆者の考えに基づく「最適解」について書かれている。答えがない、あるいは間違った設計が多数存在する中で、基礎知識と良い・悪い例から望ましいトレードオフを導き出すことが、読者に学んでほしいことではないかと考えた。これらのことから、このレポートでは本書の内容と筆者の考えについてまとめ、思うところを記述していく。

## 2.データベース設計を行う前に

　この本の序盤は設計方法ではなく、設計するデータベースに関する基礎知識と、設計を行う開発の進め方や設計の段階についての説明であった。最も印象に残ったことは、「データベースを制する者がシステムを制す」という言葉で、プログラムからデータの設計をするのではなく、データに合わせてシステムをつくるということを学んだ。また、データベース設計を外部スキーマ、概念スキーマ、内部スキーマという3層スキーマモデルに分けて考えたときに、概念スキーマである論理設計がデータ独立性を保証するうえで重要であることについても述べられている。これらに関して、システム全体を見たときの概念スキーマの重要性は、そのシステムの良し悪しを決めると言っても良いくらいに大きな役割を担っていると感じた。さらに論理設計と物理設計について学んだが、物理設計の比重が多かったことに驚いた。それは、概念スキーマである論理設計のほうが重要度が大きく、行う工程も多いと予想していたからである。しかし、ここで物理設計のほうが多く取り上げられている理由は明白で、論理設計は物理層の制約にとらわれないものであり、原則としてデータベース設計は論理設計が物理設計に先立つからである。この章における物理設計はテーブル・インデックスの定義の他に、ハードウェアのサイジング、ストレージの冗長構成決定、バックアップ設計について書かれており、今までの内容や日常的に触れてこなかった分野であったためイメージしづらく理解が困難だったが、パフォーマンスとの兼ね合いや実務において問題になりがちであることで、実務に入る前に必ず理解しなければならないと考えた。

## 3.論理設計の手順

　先ほどまではなぜ物理設計に関する記述が多いのかと述べたが、3章から5章までは論理設計に関する記述であることがわかった。これらのことは、データベース設計を学ぶことにおいて、どちらも捨てがたいほど重要な内容であることだ。正規化の手順は実際のテーブルを利用して実例を交えながら第1〜5正規形について説明されていた。これらはデータの冗長性をなくすための行為であり、SQLを学んだ際にも「なぜSQLについて学ぶのか」と考えたときに一番の要因として挙げられたものであったがゆえに、これらの操作について理解すると同時に、以前まで学んでいたSQLについての理解が深まった。主に関数従属性について説明されており、段階を踏んで理解することができたが、現実にはカラム数は10個に収まることが少ないと予想されるため、かなりの量のカラム数を持つテーブルを扱うことを考えたときに、正しく正規化できるかの不安は高まっていった。次に正規化されたテーブルの関係を視覚的に理解しやすくするためのER図の書き方について記述されていた。テーブルだけでは見えづらかった関係を明示することは画期的であり、以前にテーブルを正規化した時の曖昧な理解をここで払拭できたのではないかと感じた。IE表記法やIDEF1Xについての書き方だけでなく、リレーショナルデータベースでは多対多の関係を作ってはいけないなどのテーブルの関係に関する知識についても理解が深まった。次に論理設計とパフォーマンスについて書かれていたが、正規化されたテーブルの結合は非常にコストが高いという内容だった。この時点で、今まで学んだ正規化は基礎的な内容だから学んだのであって実用的でないのかと疑ってしまった。実際にこれらを解決する方法にSQLの他にも非正規化というものが考えられ、集計データなどを保持することで検索を高速化できる利点もある。しかし、非正規化は更新のパフォーマンスやデータのリアルタイム性など、考慮すべきことが多くなってしまうので、非正規化は最後の手段として考えられ、基本的には正規化をするというようになっている。ここでの重要なポイントは正規化と検索SQLのパフォーマンスのトレードオフであると考えられる。さらに、パフォーマンスに関してインデックスと統計情報について述べられている。インデックスはSQLのパフォーマンス改善のためのよく知られている手段である。SQLを学ぶ上であまり触れてこなかったが、パフォーマンス向上について考えたときにとても有効で、設計するときにインデックスが効率よく利用できているかによってパフォーマンスに差が出てくる。しかし、なんでもかんでもインデックスを作ればよいというものではなく、適切な場面や使える場面が限られているため、注意深く利用する必要がある。また、非常に有効な手段であるとはいってもデメリットも存在することを忘れてはならない。これと同時に統計情報についても書かれており、どのようにしてSQLを受け取ってテーブルにアクセスするかの流れを確認する必要がある。エンジニアがSQLの実行計画立案に直接的に関与することはないが、統計情報収集のタイミングや対象など、統計情報の収集をどのように行うかが重要になってくる。インデックスと統計情報は非正規化の前段階にあるパフォーマンス改善のための有効手段だが、今まで扱わなかった分、イメージを掴めず何度も反復して読み直し、内容の理解に時間を要してしまった。内容が難しい程実務において力が発揮されるので、この本において重要なポイントの一つであると考えられる。

## 4.論理設計のバッドノウハウとグレーノウハウ

　これらは今まで学んだ内容を否定するかのような手法で現実に存在する設計である。私はこの章を読んで初めて、表紙にある「正しい考え方」という言葉の意味について理解した。ここではバッドノウハウはやってはいけないことと書かれているが、世の中のシステムには、正規化も何もかも無視した論理設計が溢れかえっているという現実がある。バッドノウハウは可読性を悪化させ、それを見た人が理解しにくいものとなってしまうことや、設計変更が難しくなってしまうというデメリットを持つ。私はバッドノウハウが持つデメリットはそこに収まらず、解消方法があるにもかかわらずそれをしないさぼり癖が付くことや、そのシステムを更新する人たちに迷惑をかけるという人間性に欠けた手法であると感じ取った。しかし、バッドノウハウとは違いグレーノウハウはメリットとデメリットが拮抗している部分に位置している。使い方は簡単ではないが、正しく使うことでメリットにもなるが、当然その方法が正しいかを客観的に判断することは難しいため、バッドノウハウはもちろん、グレーノウハウも発生させないためにデータを綺麗にするというデータクレンジングを行うべきだと考えた。グレーノウハウに関して、エンジニアとしてより良い設計を目指すためにメリットだけでも抽出するべきかもしれないが、デメリットが大きくなる可能性とシステムに関わる多くの人たちのことを考えると、メリットになることが保証されない限り使わないことのほうが賢明であると考える。

## 5.データベース設計の方法と重要性

　この本を読む前は、以前読んでいたSQLについての知識を用いてデータベースの設計方法について学ぶものだと思っていた。実際に読み始めてから、設計方法だけでなく、その必要性や実務におけるソフトウェアやパフォーマンスについての考察や説明、それぞれの操作の意味や利点欠点などの情報が事細かに書かれていたため、根柢の考え方と共に設計方法を学ぶことができた。知識の理解に関して、方程式の暗記のように覚えるより、成り立ちや考え方を同時に理解することによって頭にしみつきやすく、暗記的な解法よりも納得できる形で設計することができる。ただのデータベースの設計説明書よりも、実務への不安が和らぐような内容であった。また、前半で著者に述べられた「こうするべき」という作法を、後半に反例を出すことによって本当に今までのことが正しいのかと疑問に感じさせるというところにも大きく興味を持った。それと同時に、なぜ正しいデータベースの設計方法を疑わせるようなことをするのか考えたが、そもそも著者は設計の方法というよりかは「トレードオフ」に大きな意味があると考えていたことが気になった。実際に異なるテーマの章においてもトレードオフという言葉は必要不可欠であり、それぞれの利点と欠点を学ぶことこそが、現場において求められる重要なものではないかと考えられる。

## 6.著者の「トレードオフ」の考察の意味

　著者はなぜ答えることが難しい「トレードオフ」についてこだわっているのかを考えたとき、現実問題としてその「トレードオフ」について考えられていない設計がこの世の中に溢れているのはなぜだろうかと疑問に感じた。読み始めた頃は正規化やテーブルにおける関数従属性などの概念スキーマについての知識が複雑で理解しにくく、曖昧に理解しながら設計を行っているからではないか、また、その手続きが複雑でさぼっているからではないかと考えていた。3章以降での演習問題に実際に正規化をするものがあったが、カラム数の少ない単純なテーブルであっても、カラムごとの関係や関数従属性について理解が難しく多くの時間を要してしまっていた。パフォーマンス改善の方法なども一見するとその解決は見えづらいものである。しかし、読み進めていくとこれら行き詰っていた箇所は基礎中の基礎でありプロのエンジニアがこのような理由で業務を怠るはずがないと考えた。この疑問を抱えながら読み進めると、設計を行う上でとてつもない量の「トレードオフ」について考えなくてはならないことを学んだ。それは論理設計と物理設計の比重によるものだけでなく、データの整合性とパフォーマンスの「トレードオフ」やバックアップにおけるバックアップコストとリカバリコストの「トレードオフ」、正規化と検索SQLのパフォーマンス「トレードオフ」など、どちらも捨てがたいものの取捨選択を何度も強いられなければならないものであった。さらにそれらは状況によって最適な答えが変わってしまうものであり、考えることに相当なリソースを要することとなる。その状況とはビジネスロジックなどの制約条件や、開発に伴うコスト、扱うデータの数や種類など要因は数え切れず、担当したシステムがもともと悪い設計であり、改修に当たってそれらに従わなければならないという場合もあるという。しかし、著者がトレードオフについて多く言及するということから、データベース設計をマスターするにあたってトレードオフを考える難しさを受け入れ理解することが、世の中の悪い設計を根絶する方法であることが考えられる。本書の内容の順序として、データベースの基礎知識、設計方法、バッド・グレーノウハウとなっているのは、正しい設計方法について理解してから世の中で起きている設計方法の問題を知ることで、正しい設計方法を形式的に覚えるのではなく、悪い設計と比較したり、そのシチュエーションにおけるトレードオフの関係をとらえることによって、設計に関する知識に収まらず、その知識の根柢や重要なポイントを押さえられるようになっているのではないかと考えた。

## 7.本書を読んで考える、なるべきエンジニア像

　本書を読み、データベースの基礎知識、設計方法、バッドノウハウなどについて学んだ。これらから得た率直な感想は、当たり前だがやり方を学ぶだけでは意味がないということである。このことは内容の約1/3が設計方法以外のものであることと、順序の後半に現実問題の提訴が行われていることが証明していると考えられる。実際に設計方法のみの教本だった場合、演習問題をこなすことはできるだろうが、現場に出れる程の一人前になったとは思えない。基礎知識のほかに様々なトレードオフの関係を理解すること、バッドノウハウの問題について知り、根絶に向き合うことによって、ようやくデータベース設計のスタートラインに立てるのだと考えた。それらを知って設計方法に戻ることで、正規化・ER図の作成の重要さを深く理解でき、基礎知識の大切さを考えられるようになっている。私は本書を読み、「データベースがシステムの中心であり、システム開発の中心でもある」ということや、まず正規化を行い、そのあとにパフォーマンスとのトレードオフの関係を考えることなど、基本を順守できるようなエンジニアになることを目指そうと考えた。これらの考えが疎かになっていれば、設計やトレードオフについて問題解決にすることができずに目をそむけてしまい、バッドノウハウを利用してしまったり、適切なトレードオフの選択ができなくなるからである。研究や技術が発展し、データベース設計方法などの常識が今後も目まぐるしく移り変わることは避けられないが、本書で学んだ基本的であり順守しなければならないことを忘れなければ、時代の変化に対応でき、新たに発生する問題にも向き合うことができるのではないかと考える。柔軟な対応の中に考え方の確固たる軸を築くことで、一人前のエンジニアを目指したいと思った。

## 8.本書の伝えたいこと

　本書を通して伝えたいことは、「データベースエンジニアの仕事は、シビアなトレードオフにある相反したベクトルの均衡点を探すことにある」ということではないかと考えた。論理設計を物理設計に優先すべきというものや、正規化によっておこるデータの整合性と検索SQLのパフォーマンス悪化、非正規化やバッドノウハウによるパフォーマンスの最大化とそれによって起こるデータのリアルタイム性の低下と更新異常のリスクなど、基礎知識を固めた先には多くのトレードオフによる問題がある。さらに実際の開発において、「予算」や「使っているDBMSの機能上の制限」など、二つに収まらないほどのベクトルまで考慮しなければならない。筆者はそれらの多次元ベクトルの方程式を解くことが、エンジニアには求められると記述している。私はこのことから、データベースの基礎知識や設計方法など、本書において学んだことは、その多次元ベクトルの方程式を解くための材料であり、筆者が述べているエンジニアとして求められることについて向き合えるようになることが、本書の伝えたいことではないかと考えた。

以上