Engineer

データウェアハウス

事業のデータを分析のために集めるデータベース

どのようなものか

データウェアハウスは、事業を動かすためではなく、事業についての問いに答えるために作るデータベースです。注文を受ける、出荷を手配する、請求書を発行する、サポートの電話を記録するといったシステムは、それぞれの仕事に適した形で、自分のデータを持っています。ウェアハウスは、それらすべてからコピーを受け取り、何年も保存します。複数のシステムにまたがる問いが、一週間分のエクスポート作業ではなく、一つのクエリーで済むように整理します。これを構築して運用することを、データウェアハウジングと呼びます。

コピー元のデータベースとの違いは、処理の性質です。注文システムは、一度に少数の行を、一分間に何千回も読み書きし、現在だけを保持します。顧客が引っ越すと、古い住所を上書きします。ウェアハウスへの書き込みはバッチで行い、読み取りは、一度に数百万行を走査するクエリーです。過去を意図的に残すため、去年の数字を、去年の事業の区分のまま集計し直せます。

Ralph Kimballは、1996年にウェアハウス業界へディメンショナルモデリングを広めました。Kimball Groupは、この種のシステムを二つの部屋に分けます。裏側の部屋には、ETLシステムを置きます。それを34のサブシステムに分け、四つの仕事にまとめています。発生源からのデータ抽出、クレンジングと共通定義への整合、プレゼンテーション領域への提供、全体の管理です。表側の部屋には、実際に人が問い合わせるプレゼンテーション領域を置きます。業務プロセスごとに整理し、各プロセスが記録する最も細かい粒度を保ち、速くするための集計結果を加え、ファクトテーブルとディメンションテーブルで構成します。その周りには、BIアプリケーションを置きます。個別のクエリー、標準レポート、ダッシュボード、スコアカード、分析アプリケーションです。Kimball Groupは、表側の部屋を「DW/BIシステムの対外的な顔」と呼んでいます。

構成は、これだけではありません。Kimball Group自身のアーキテクチャのページは、Corporate Information Factoryも説明しています。最小粒度のデータを、第三正規形に正規化した企業全体のウェアハウスに保持し、二段目のETLで、レポート用のディメンショナルなテーブルへ読み込む、ハブ・アンド・スポーク型の設計です。また、既存の正規化したウェアハウスから、ディメンショナルなプレゼンテーション領域へ供給する、混合型も説明しています。どの構成にも共通するのは、分離です。事業を動かすシステムは、事業を分析する場所にはしません。

ウェアハウスは製品名ではなく、役割です。チームが運用するサーバー上のPostgreSQLやMicrosoft SQL Serverでも、Google BigQueryのようなマネージドサービスでも構いません。GoogleはBigQueryを、クラウドのデータウェアハウスと分析ツールを組み合わせた、完全マネージドのデータプラットフォームと説明しています。BigQueryは、ストレージと計算資源を分け、それぞれを独立に拡張します。行ではなく列ごとにデータを保存し、ANSI標準のSQLでクエリーを受け付けます。

概念を判断に生かす

「データウェアハウスとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 営業、運用、経理の数値をまとめ、同じ会議に部門ごとに異なる答えを持ち寄る状況を解消します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 地理情報パイプラインを50倍に高速化 — 50x 地理データパイプラインの高速化

地理情報パイプラインを50倍に高速化 お問い合わせ →

なぜ重要なのか

企業が同じ問いを二つのシステムへ投げると、二つの答えが返ります。営業は注文時、経理は請求書の支払い時、サポートは問い合わせが届くたびに、顧客を数えます。それぞれの目的では間違っていませんが、互いの数字は一致しません。ウェアハウスでは、一つの定義について合意し、読み込み処理に明記します。そこから作るすべてのレポートが、同じ方法で数えるためです。

Kimball Groupの道具は、適合ディメンションです。一つの顧客、一つの製品、一つのカレンダーを、ETLシステム内で集中的に管理し、必要とするすべてのファクトテーブルで共有します。同じ顧客ディメンションを参照する、営業のファクトテーブルとサポートのファクトテーブルは、行ごとに比較できます。バスアーキテクチャでは、業務プロセスと、それらが共有するディメンションを、アーキテクチャの設計図と呼ぶ行列に並べます。業務プロセスを一つずつ追加してウ

ェアハウスを作れるため、提供までの期間を短縮できるとしています。同じディメンションを使えば、注文を扱う部分を今四半期に、返品を扱う部分を次の四半期に提供し、すでに届けた部分を設計し直さずに済みます。

第二の理由は、履歴です。業務システムは現在の状態を保持しますが、ウェアハウスは、すべての状態を残せます。変化する属性を扱うKimballのタイプ2という技法は、新しい値と新しい代理キーを持つディメンション行を追加します。その時点以降に読み込むファクトは、すべて新しい行を指します。顧客の引っ越し前の売上は古い地域に残し、引っ越し後の売上は新しい地域へ結び付けます。少なくとも、適用開始日、有効期限、現在有効な行を示すフラグという三つの追加列があれば、どの日に何が正しかったかを問い合わせられます。

第三の理由は、単純に負荷です。三年分の注文を走査するレポートを、今日の注文を受け付けるデータベースで実行すべきではありません。購入手続きと、同じメモリー、ディスク、ロックを奪い合うためです。分析を専用のデータベースへ移せば、それぞれの仕事に合わせて調整できます。

代償は、ウェアハウスがコピーであることです。データの鮮度は最後に読み込んだ時点までで、正しさは、その読み込み規則の正しさ次第です。実行するスケジュールと、事業で合意した売上、顧客、一か月の定義、その両方に責任を持つ人が必要です。

実際の活用例

ディメンショナルなウェアハウスの中心は、最初に粒度を宣言したファクトテーブルです。ここでは、注文の一明細につき一行とし、その内容を説明するディメンションで囲みます。顧客ディメンションにはタイプ2の列を持たせ、顧客が引っ越したら上書きせず、二行目を作ります。

```
CREATE TABLE dim_customer (
    customer_key bigint PRIMARY KEY,   -- ウェアハウスが管理する代理キー
    customer_id  text NOT NULL,        -- 元のシステムでのキー
    region       text NOT NULL,
    valid_from   date NOT NULL,
    valid_to     date,                 -- 現在有効な行では空欄
    is_current   boolean NOT NULL
);

CREATE TABLE fact_order_line (
    date_key     integer NOT NULL REFERENCES dim_date,
    customer_key bigint  NOT NULL REFERENCES dim_customer,
    product_key  bigint  NOT NULL REFERENCES dim_product,
```

```
    quantity     integer NOT NULL,
    net_amount   numeric(12, 2) NOT NULL
);

SELECT d.year, c.region, sum(f.net_amount) AS revenue
FROM fact_order_line AS f
JOIN dim_date AS d USING (date_key)
JOIN dim_customer AS c USING (customer_key)
GROUP BY d.year, c.region
ORDER BY d.year, c.region;
```

各注文明細は、その時点で有効だった顧客行とともに読み込んでいるので、このクエリーは、購入時に顧客が住んでいた地域ごとの売上を返します。代わりに `customer_id` で結合し、 `is_current` で絞ると、過去全体を現在の地域で集計し直します。どちらも正当な問いであり、モデルは、データの二つ目のコピーを作らずに、両方へ答えられます。

データウェアハウス設計サービスが約束するのは、まさに最初の理由です。営業、業務、経理の数字を一か所に集め、二部門が同じ会議へ二つの答えを持ち込まなくなるようにします。より大きな規模では、創業者は2022年から2024年に、PostgreSQL、Microsoft SQL Server、Google BigQuery上の100のデータウェアハウス用データベースを設計し、運用しました。主に、空間座標やセンサーの読み取り値など、GISデータと時系列データを保持していました。最初の二つのエンジンでは、正規化したテーブルと非正規化したテーブルの構成、空間インデックス（PostgreSQLではPostGIS）、時系列のパーティション分割を組み合わせました。BigQueryでは、時間でパーティション分割し、クラスタリングしたテーブルに、圧縮と列指向ストレージを使いました。GISと時系列データへのクエリー応答時間は40–60%短くなり、この基盤により、データベースを使う組織が、リアルタイム分析ダッシュボードなどの新しいデータ製品を提供できました。

問題が起きるところ

一つのファクトテーブルに二つの粒度を入れると、どの合計でも一部の金額を二重に数えます。 注文には明細があり、注文全体に属する配送料もあります。各明細に配送料を入れると、明細の合計では、明細の数だけ数えます。Kimball Groupの四段階の設計手順は、業務プロセスを選ぶ直後、ディメンションやファクトに名前を付ける前の、二番目に粒度の宣言を置きます。異なる粒度を、一つのファクトテーブルで共有してはいけない、という規則です。 注文全体の料金は、注文単位の別のファクトテーブルに保持するか、事業で合意した規則に従って、明細へ配分します。

顧客が引っ越すと、去年のレポートが変わります。 顧客行の地域を単純に上書きすると、過去の売上もすべて、顧客とともに新しい地域へ移ってしまいます。一月に承認したレポートが、六月には別の内容になります。履歴が重要な属性には、タイプ2を使います。現在の行を閉じ、新しい代理キーで行を追加します。一つのデータベーストランザクションで行い、どの読み込み処理にも、両方の行が現在有効だと見える瞬間を作らないようにします。

```
BEGIN;
UPDATE dim_customer
SET valid_to = current_date, is_current = false
WHERE customer_id = 'C-1042' AND is_current;
INSERT INTO dim_customer
    (customer_key, customer_id, region, valid_from, valid_to, is_current)
VALUES (90731, 'C-1042', 'North', current_date, NULL, true);
COMMIT;
```

二つのチームが二つの顧客ディメンションを作り、終わらせるはずだった議論をウェアハウスが繰り返します。 新しい分析分野を扱う部分は、自分用の顧客一覧や製品一覧のコピーを持たせた方が早く提供できます。しかし、それぞれのコピーがずれます。一方は閉鎖したアカウントを外し、もう一方は異なる方法で重複をまとめます。共有するディメンションは、ETLシステム内で一度だけ作り、新しいファクトテーブルには、それを使わせます。バス行列が確認表になります。業務プロセスを加える前に、新しいファクトテーブルが、既存のどのディメンションを指す必要があるかを示します。

レポートが、準備用のテーブルを直接参照します。 発生源からの生のコピーは、まず裏側の部屋へ届きます。読み込みが済んでいるので、そこを直接問い合わせたくなります。しかし、そのテーブルには、各発生源独自のコード、重複、欠損があり、発生源が変わるたびに変わります。その上に作ったレポートは、ウェアハウスが解くはずだった問題を引き継ぎます。裏側の部屋はETLシステム専用にし、レポート、ダッシュボード、分析担当者には、プレゼンテーション領域だけを使わせます。数字を一致させるのは、そこにある適合ディメンションと、宣言した粒度です。

参考資料

- KimballのDW/BIシステムの技術アーキテクチャ — 裏側と表側の部屋、ETLのサブシステム、メタデータ、ハブ・アンド・スポーク型と混合型の選択肢です。
- 企業データウェアハウスのバスアーキテクチャ — 適合ディメンション、バス行列、業務プロセスを一つずつ追加する構築方法です。

- 四段階のディメンショナル設計手順 — 業務プロセス、粒度、ディメンション、ファクトを、この順で決めます。
- タイプ2：新しい行の追加 — 新しい行、新しい代理キー、三つの追加列によって、変わる属性の履歴を保存する方法です。
- BigQueryの概要 — Googleのマネージドウェアハウスです。ストレージと計算資源を分け、データを列ごとに保存します。

用語集

実際の制約に合わせて選ぶ

代表的なデータセット、そこから答えを得たい問い、現在のボトルネックをお知らせください。品質、処理量、復旧の測り方を合意できます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「データウェアハウスとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く 連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/data-warehouse/