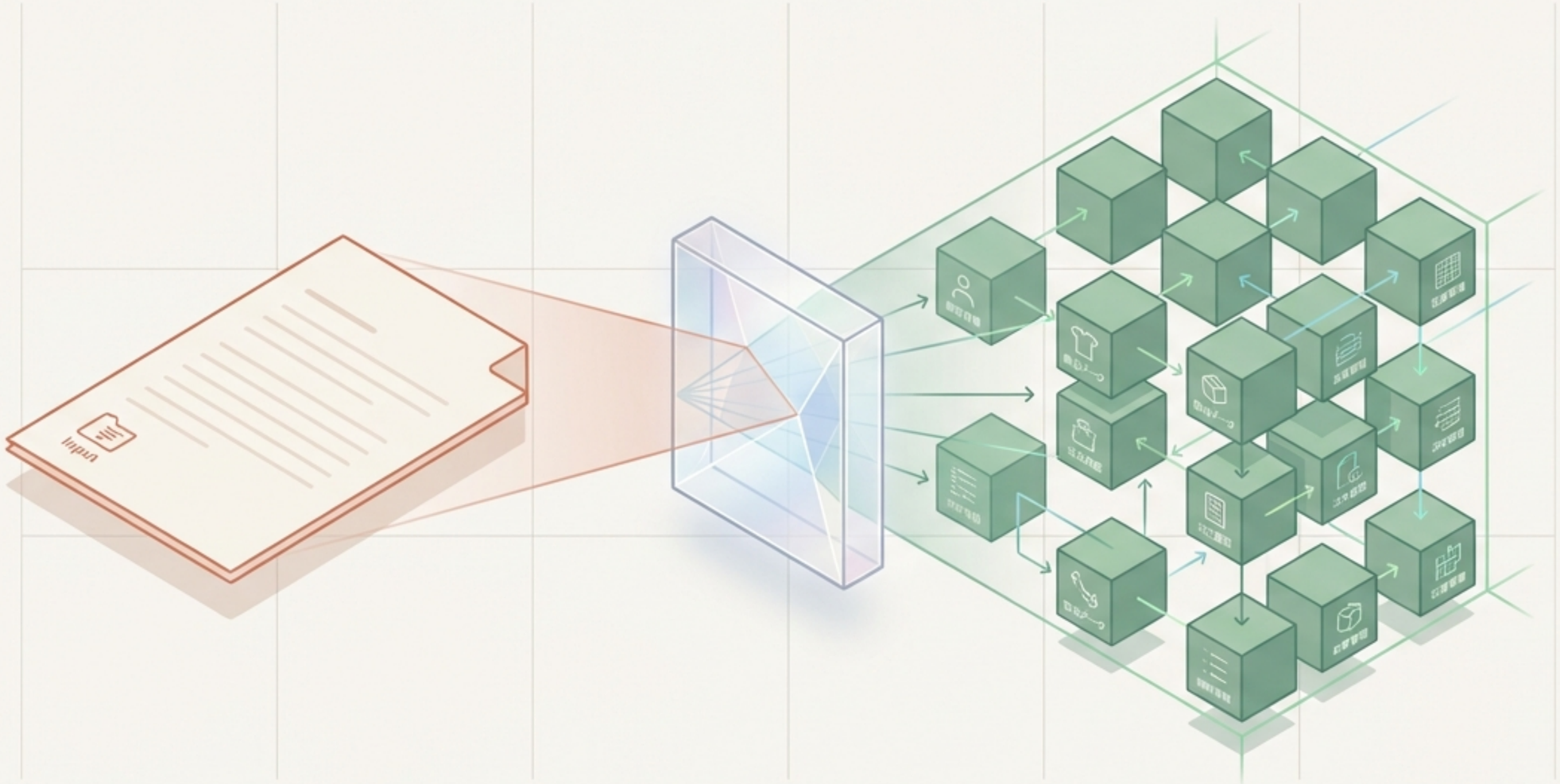
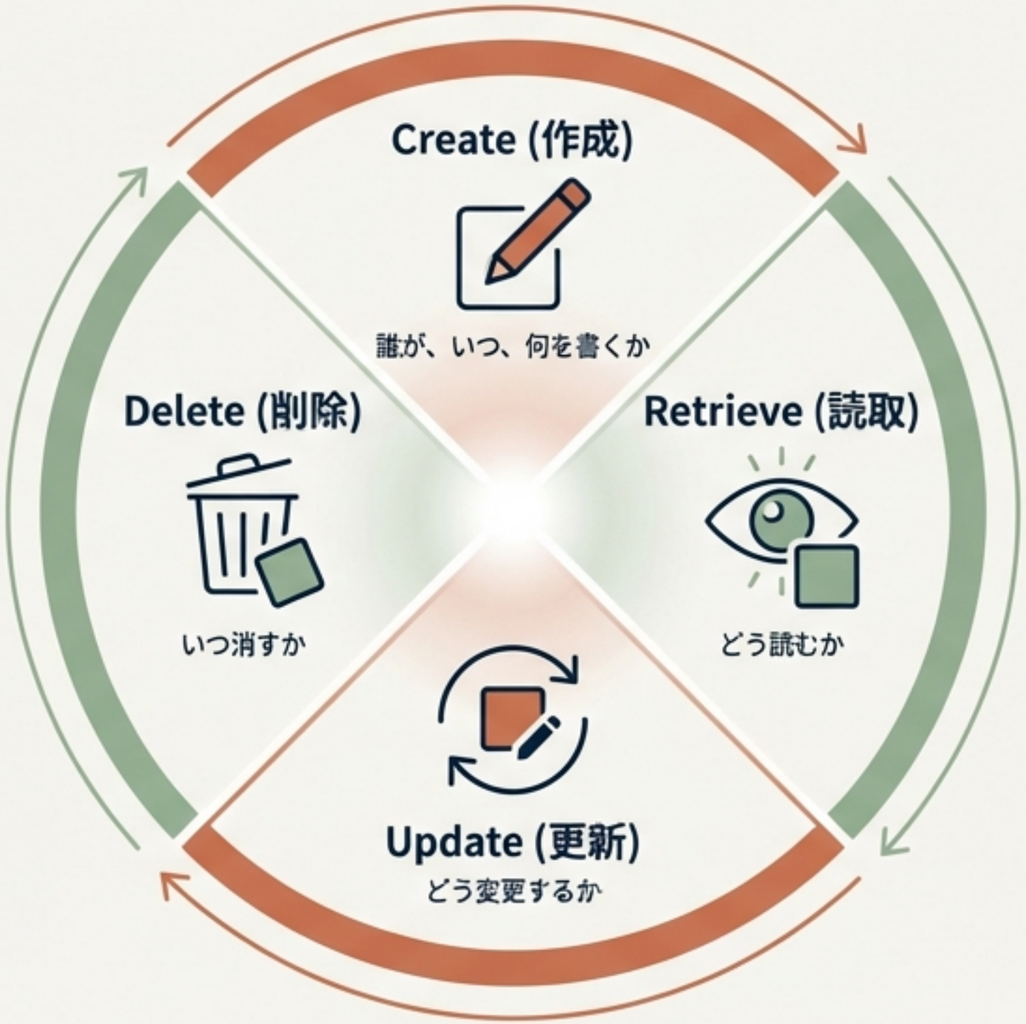
システムの心臓部を理解する - データモデル入門

なぜ情報は「バラバラ」に保存されるのか？



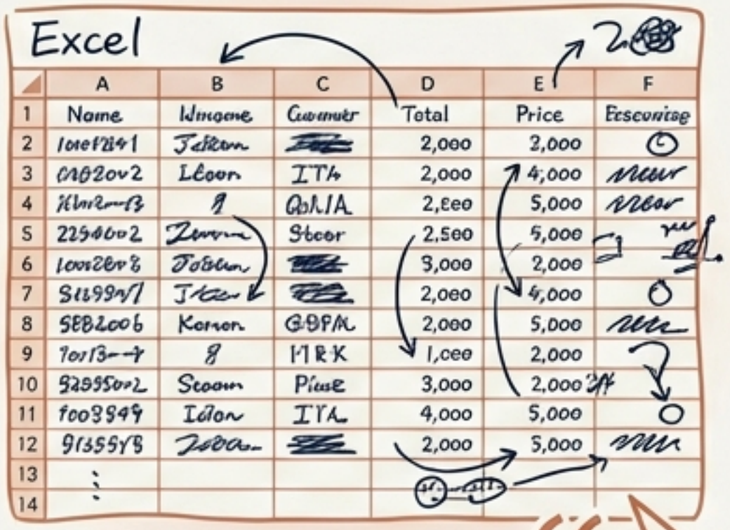
システムとは、「データを扱う仕組み」である

システムは画面や機能が異なっても、その本質は「データを書いて、読んで、更新して、削除する」ことにあります。



人間の直感 vs システムの論理

非エンジニアにとっての「壁」



「全部1枚のシートで見たい」

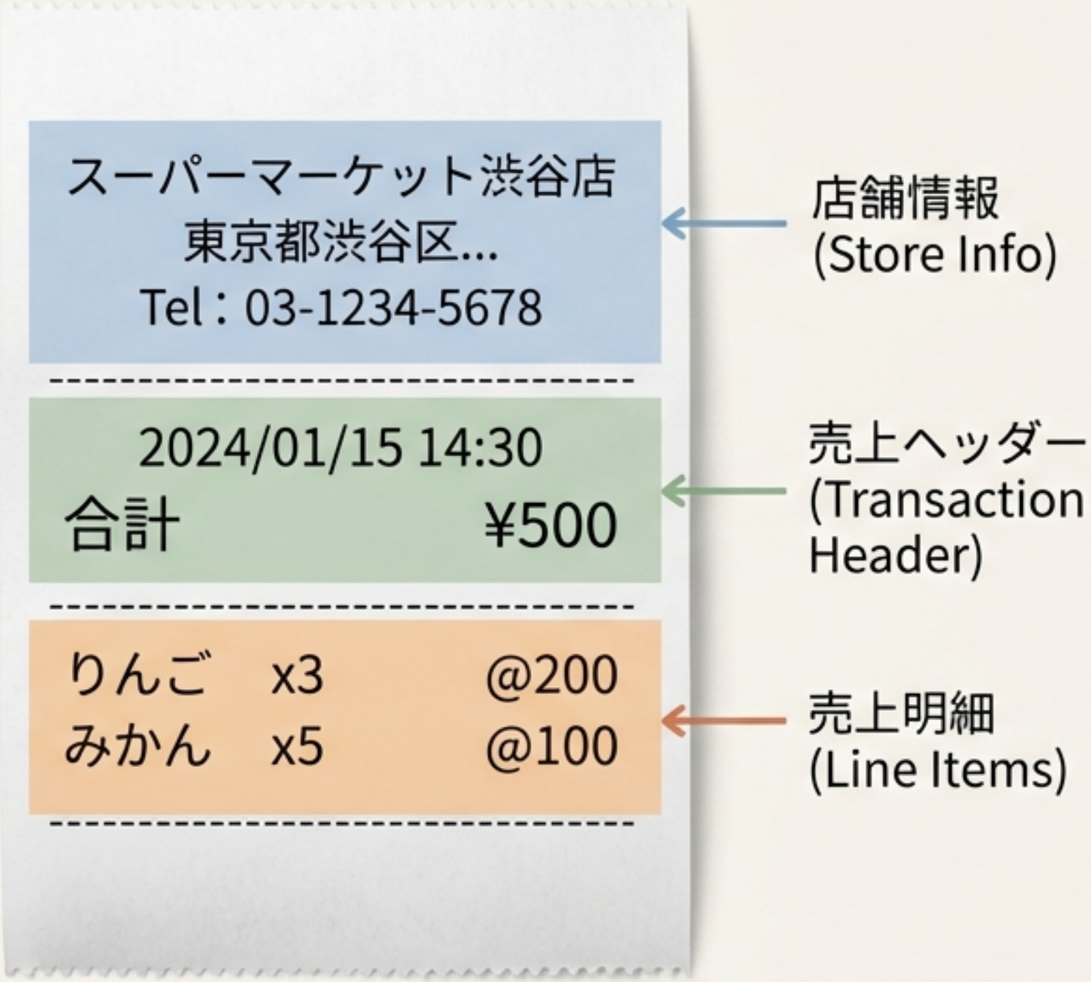
人間の直感
(Natural Intuition)

「効率のために分割して保存する」

システムの論理
(System Logic)

なぜデータを分割して保存する必要があるのか？

「レシート」から学ぶデータ構造



1枚の紙に見えるレシートは、実は3つの異なる「情報の層」で構成されています。

- **店舗情報：取引先の情報（固定）**
- **売上ヘッダー：いつ、いくら買ったか（1回限り）**
- **売上明細：具体的に何を買ったか（複数行）**

「分ける」ことで、データは強くなる

Bad: The Mega-Row

渋谷店	東京	03-xxxx	1/15	りんご	3	200	みかん	5	100	...		

店舗情報が繰り返され、商品が増えるたびに横に伸びる（非効率）

店舗テーブル（Store）

ID	店舗名	電話番号
001	渋谷店	03-xxxx

売上テーブル（Sales）

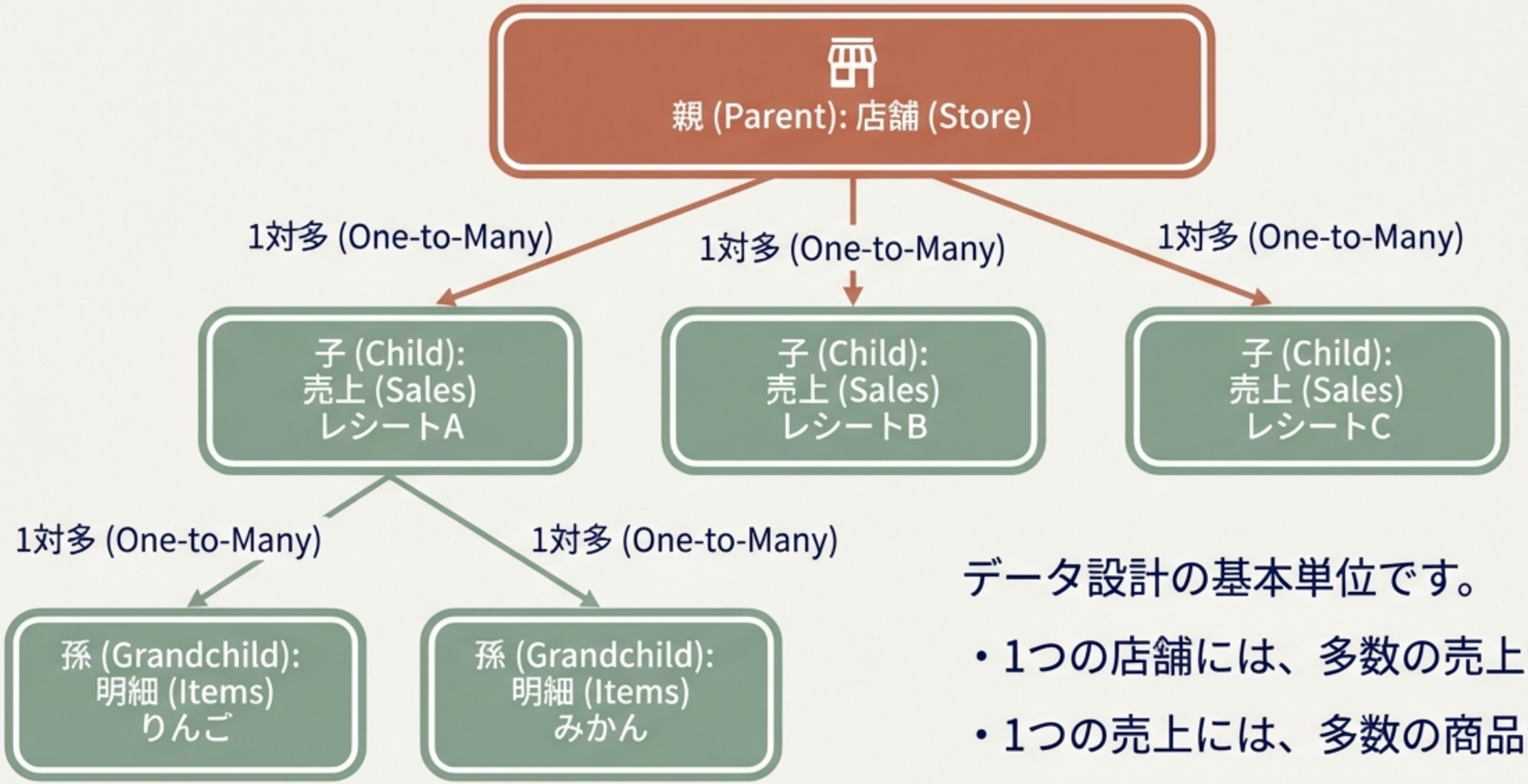
ID	Store_ID	日付
S001	001	1/15

明細テーブル（Items）

Item	Qty	単価
りんご	3	200
みかん	5	100

👉分割することで、店舗の電話番号が変わっても1箇所修正するだけで、
全ての過去・未来のデータに反映されます。

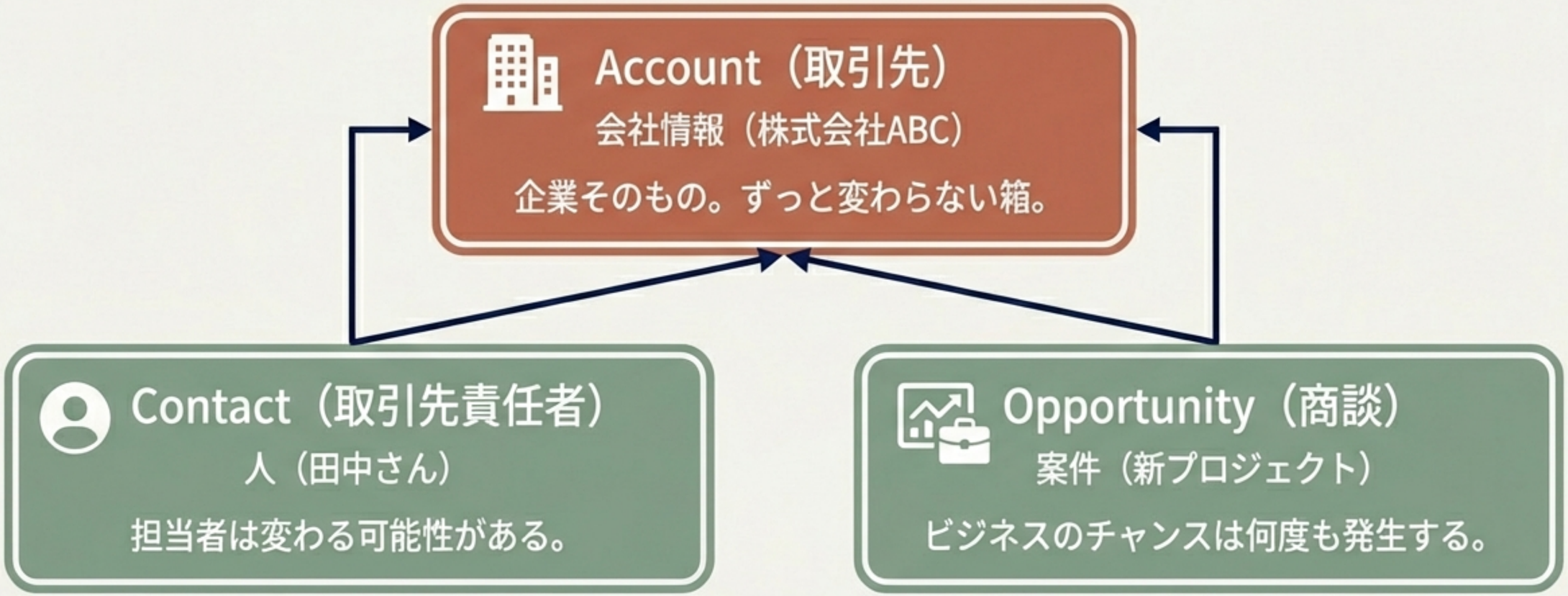
親子関係（1対多）を理解する



データ設計の基本単位です。

- 1つの店舗には、多数の売上が紐づく
- 1つの売上には、多数の商品が紐づく

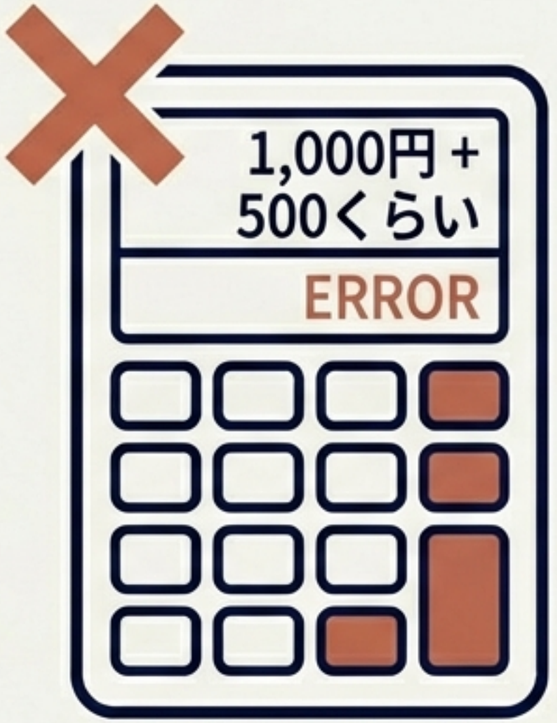
Salesforceにおける「3つの箱」



レシートと同じ論理です。「会社（Who）」と「取引（Transaction）」を分けることで、担当者が変わったり、新しい商談が始まったりしても柔軟に対応できます。

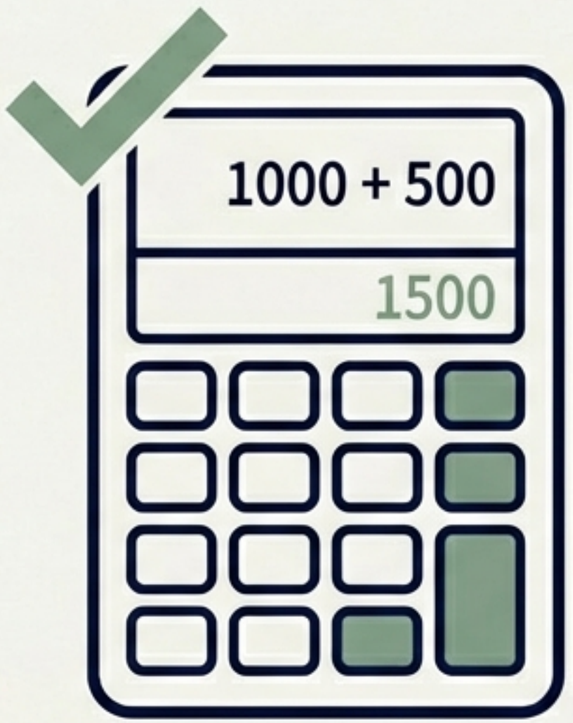
データには「型」がある

コンピュータに正確に計算させるためのルール



テキスト型（Text）

文字が混ざると計算できない

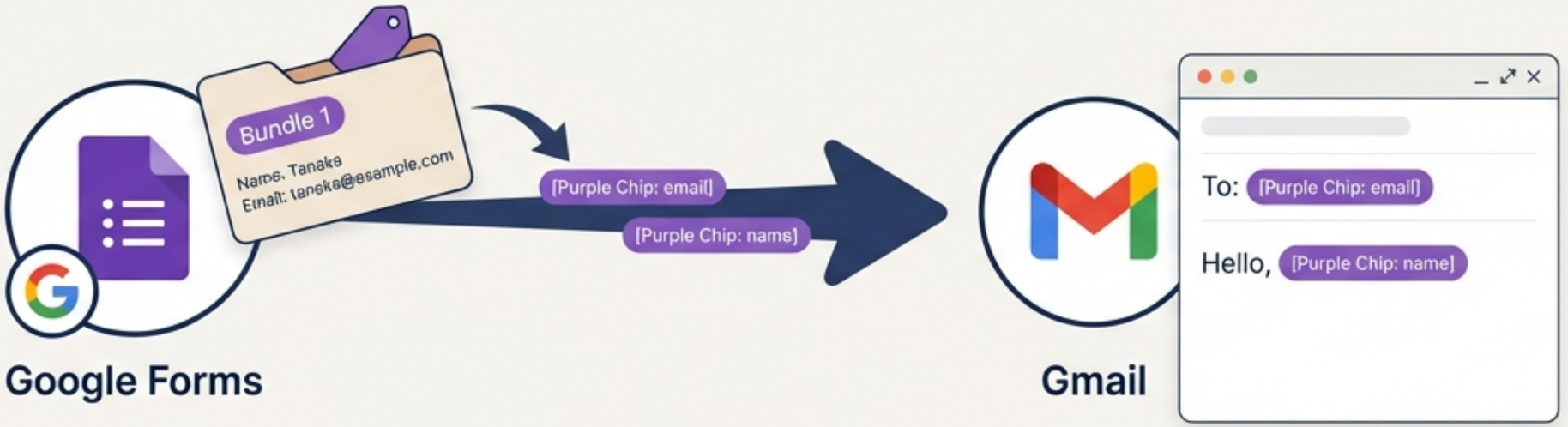


数値型（Number）

純粋な数字なら計算可能

- **テキスト**
  名前やメモ
- **数値**
  金額や数量（計算用）
- **日付**
  2024/01/15
- **チェックボックス**
  Yes/No
- **参照**
  他のIDへのリンク

データが流れる時の形：Bundle

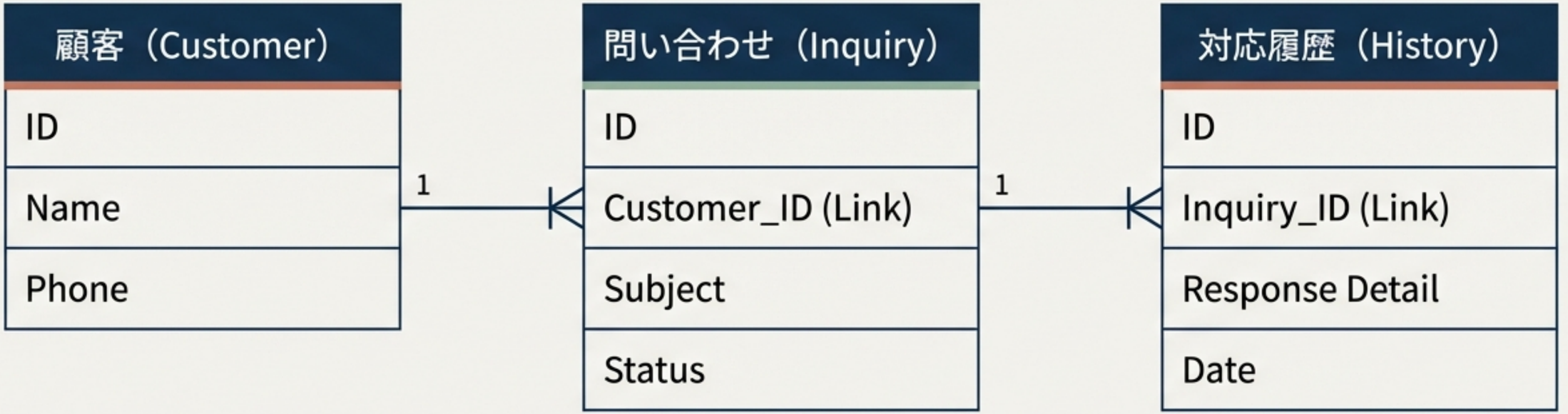


マッピング (Mapping)：変数（紫色のチップ）を宛先に埋め込む

オートメーションでは、データは「Bundle」という塊で流れます。ある場所から来たデータを、次の場所の項目に「マッピング」することで自動化が成立します。

実践：顧客対応システムを設計する

要件：顧客、問い合わせ、対応履歴を管理したい



【ポイント】
対応履歴を顧客テーブルの中に書かないこと。問い合わせは無限に増える可能性があるため、別のテーブル（1対多）として切り出します。

よくある間違いと対策

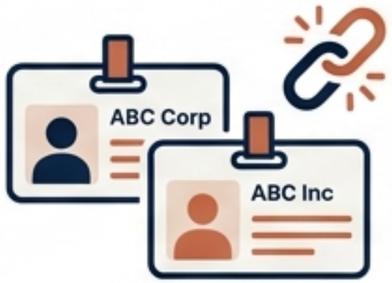
The Mega-Sheet

間違い (Mistake)

全部を1つのシートにまとめる

対策 (Solution)

行ではなく列が増え続けるのを防ぐため、テーブルを分ける。



Ignoring IDs

間違い (Mistake)

名前で紐づける

対策 (Solution)

名前は変わる可能性がある。必ず不変の「ID」で管理する。

Ignoring Types

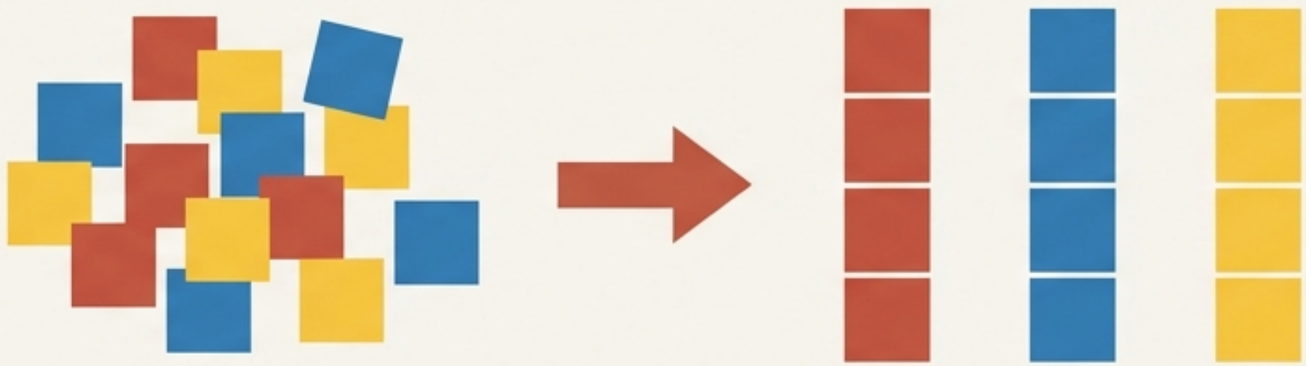
間違い (Mistake)

日付や数値をテキストにする

対策 (Solution)

計算や並び替えができなくなるため、厳密な「型」を守る。

「正規化」という考え方



これまで解説した「テーブルを分ける」技術は、専門用語で「正規化」と呼ばれます。

目的：データの重複と依存を排除すること

第1正規形、第2正規形...という定義を暗記する必要はありません。

「同じ情報が複数箇所にあるのは危険信号」

「このデータは誰（何）に属するか？」

この2点を意識するだけで、十分な設計ができます。

システム視点を手に入れる

System = CRUD

システムとは、データを作成・更新・削除する仕組みである。

Split to Strengthen

情報を適切なテーブルに分割し、重複をなくす。

Think 1-to-Many

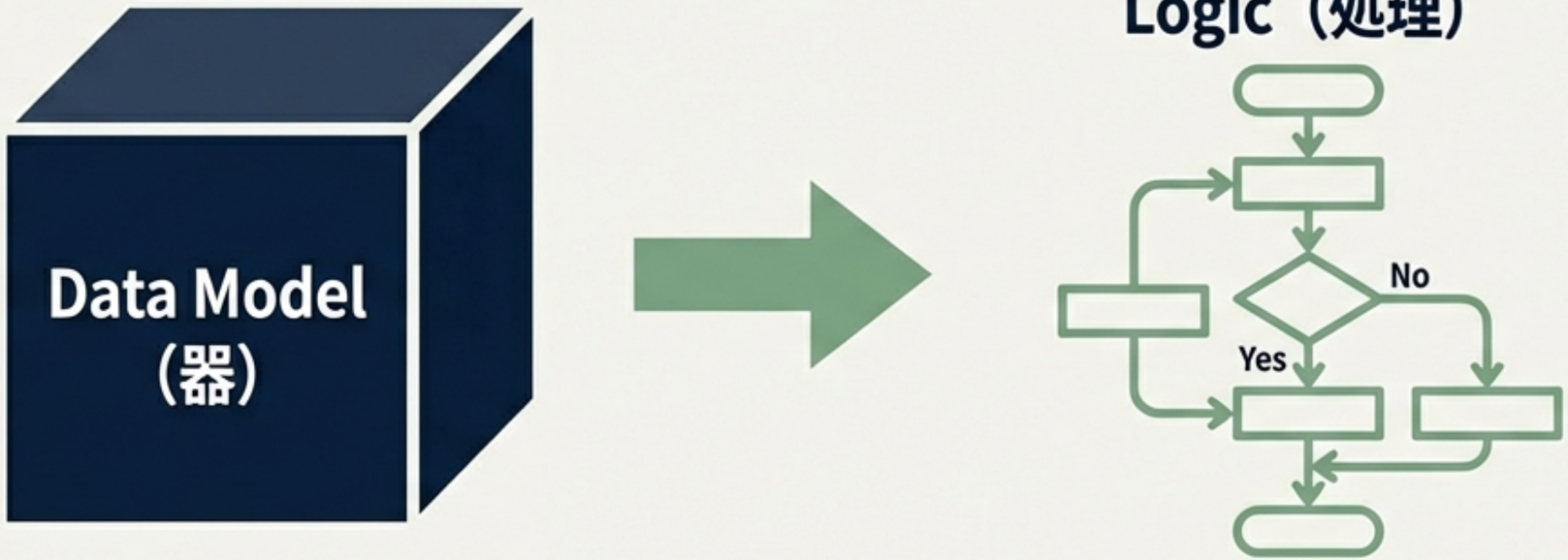
「親（1）」と「子（多）」の関係を見極める。

Respect the Type

数値、日付、テキスト。型を厳密に定義する。

Next Steps

データ構造の次は、「ロジック」へ



データモデルはあくまでデータを保存する「器」です。
器の構造を理解した今、次は中身をどう操作するかを学びます。

Coming Soon: 変数、ループ、条件分岐（プログラミング的思考）