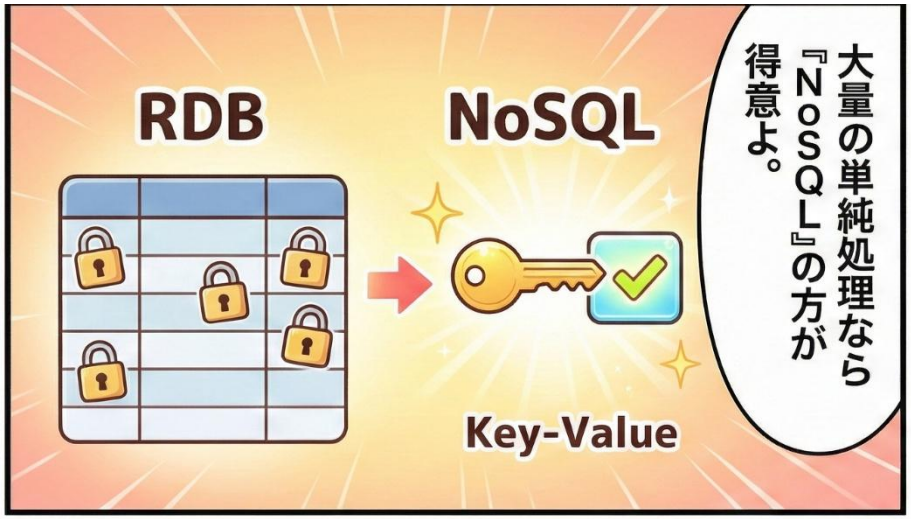
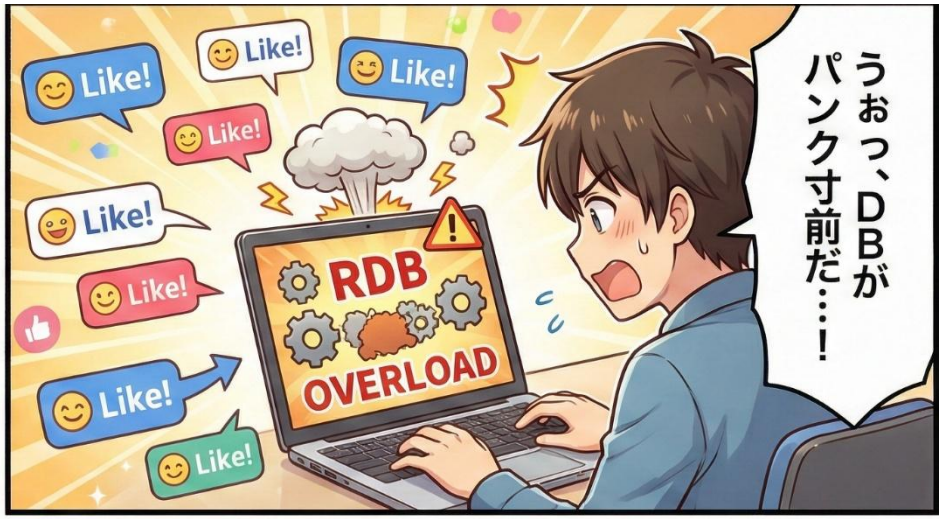
15. 種々のデータベースシステム、NoSQLデータベース

URL: https://www.kkaneko.jp/de/ds/index.html

金子邦彦





Nano Banana Pro を用いて作成

今回の内容

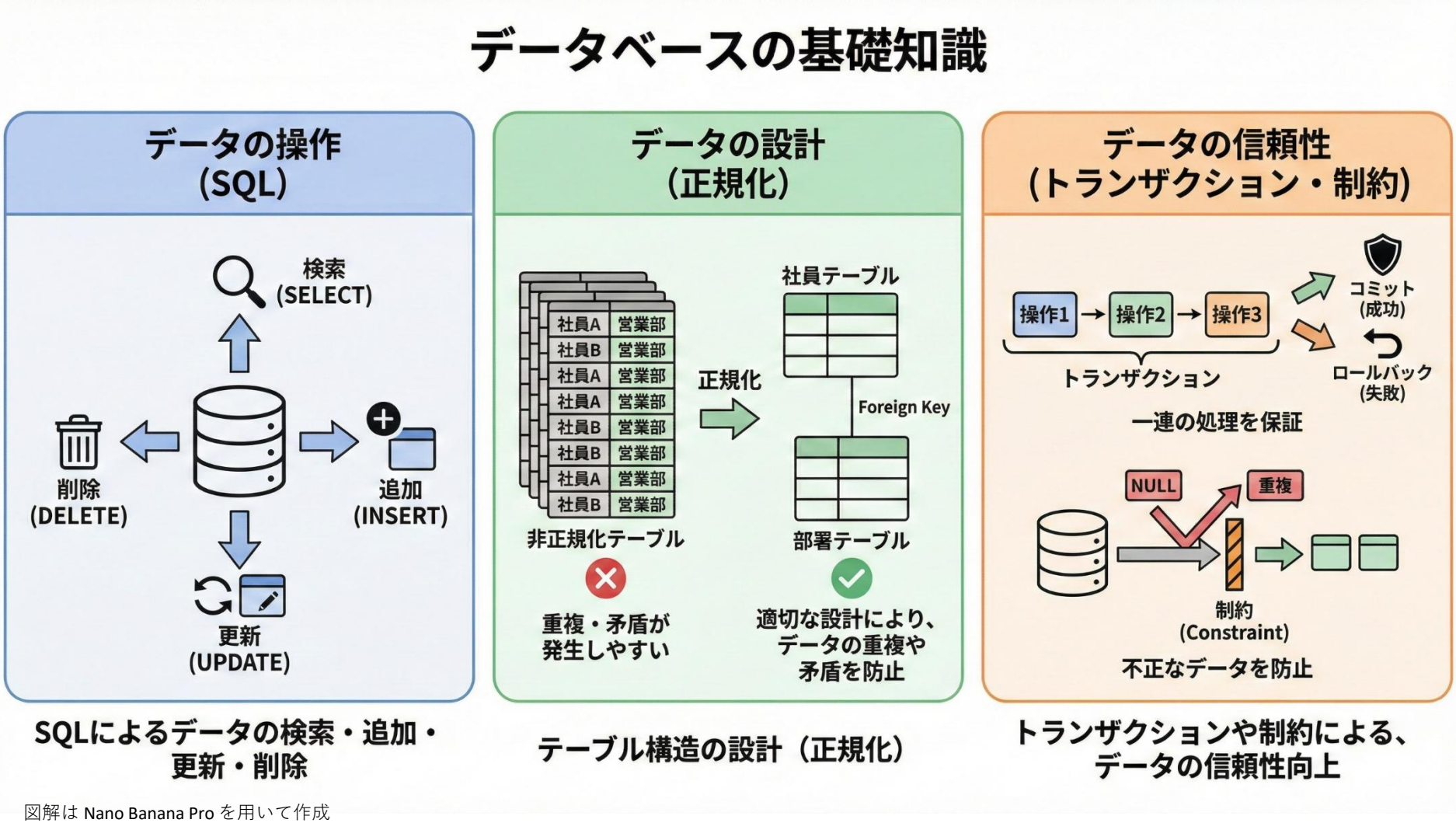
背景

- **リレーショナルデータベース**は、**データベース技術の基盤**
- 多くのシステムで利用されている
- **すべての場面で最適とは限らない**

目的

- リレーショナルデータベース**以外の選択肢**「**NoSQL**」を学ぶ
- **状況に応じたデータベース選択**ができるようになる

第1回から第14回で学んだ内容

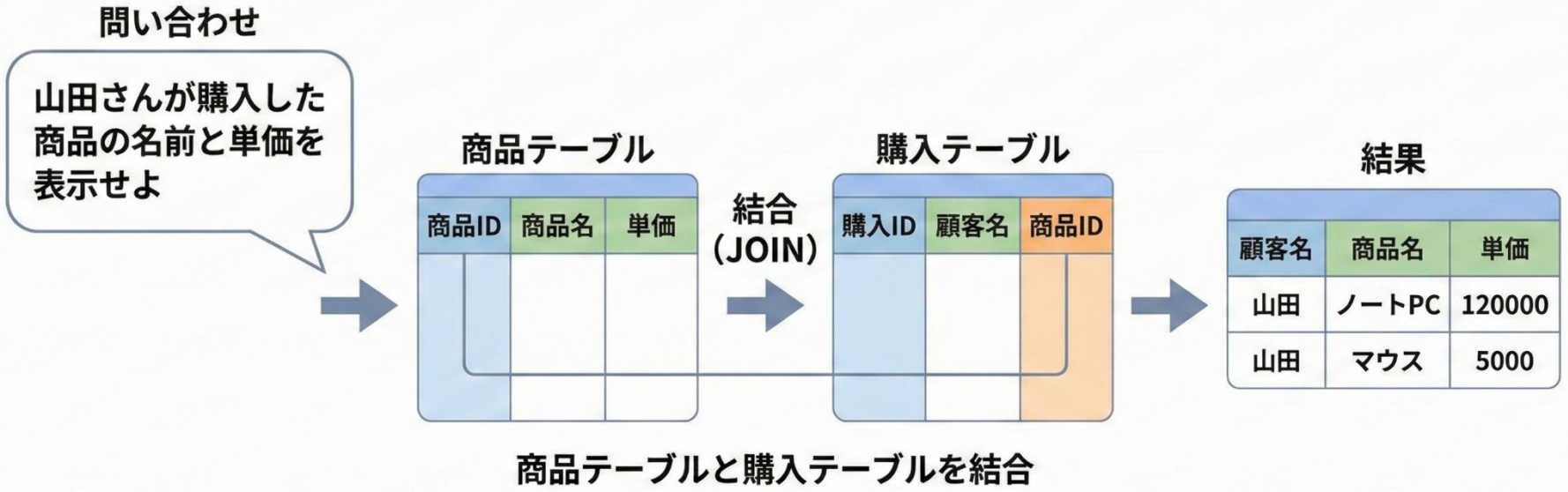


図解は Nano Banana Pro を用いて作成

これらはすべて**リレーショナルデータベースの技術**である。

リレーショナルデータベースが得意なこと①

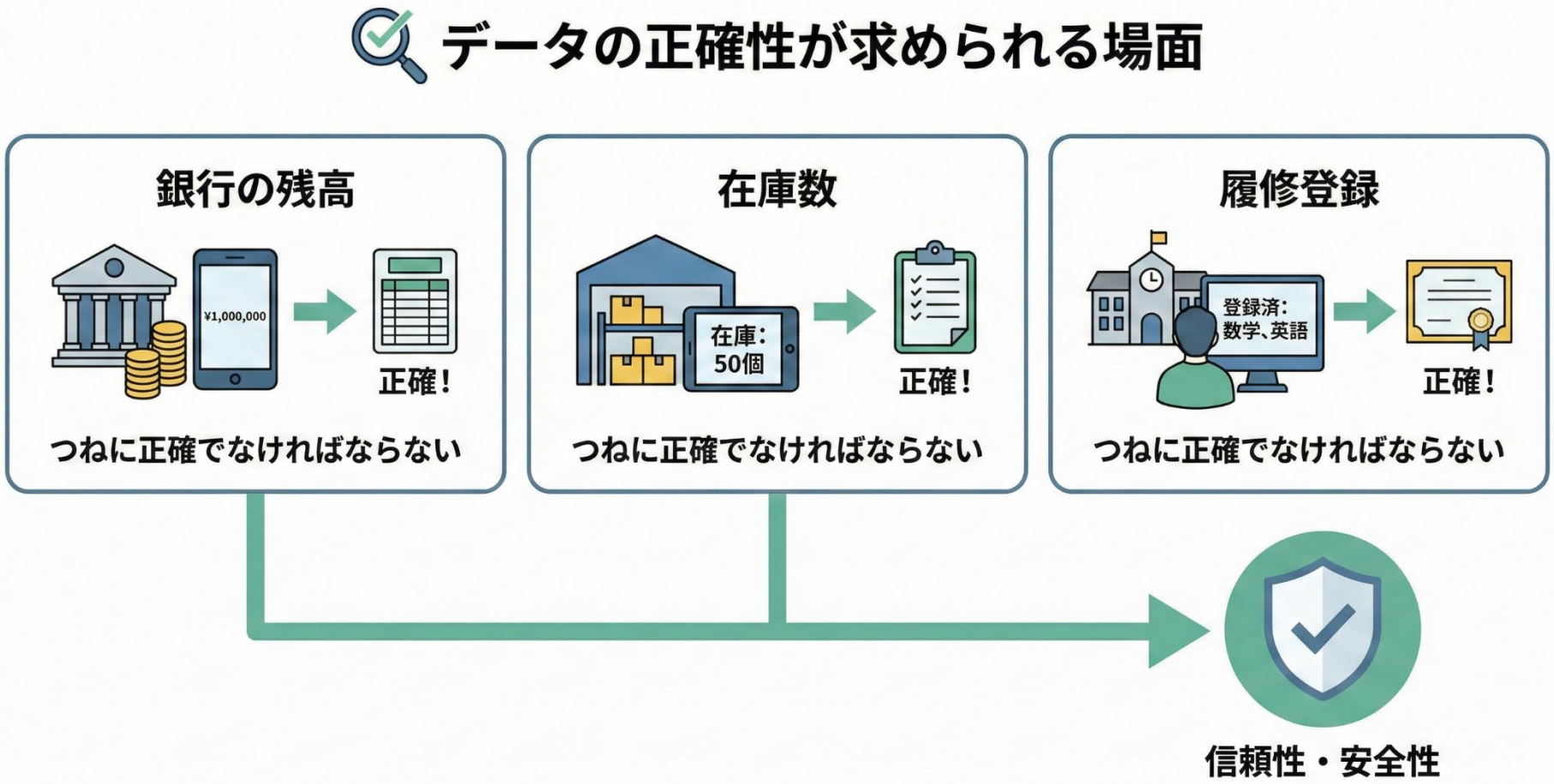
複数のテーブルを関連付けた問い合わせ



図解は Nano Banana Pro を用いて作成

リレーショナルデータベースは、このような複雑な問い合わせを得意とする。

リレーショナルデータベースが得意なこと②



リレーショナルデータベースは、**トランザクション**や**制約**により、データの信頼性を向上できる。

リレーショナルデータベースが苦手なこと

インターネットの普及により、**リレーショナルデータベースでは対応が難しい状況が生まれた。**

①　大量の単純なデータを高速に処理する場面

- SNSの「いいね」ボタンは、**1秒間に数万件**押される可能性がある。

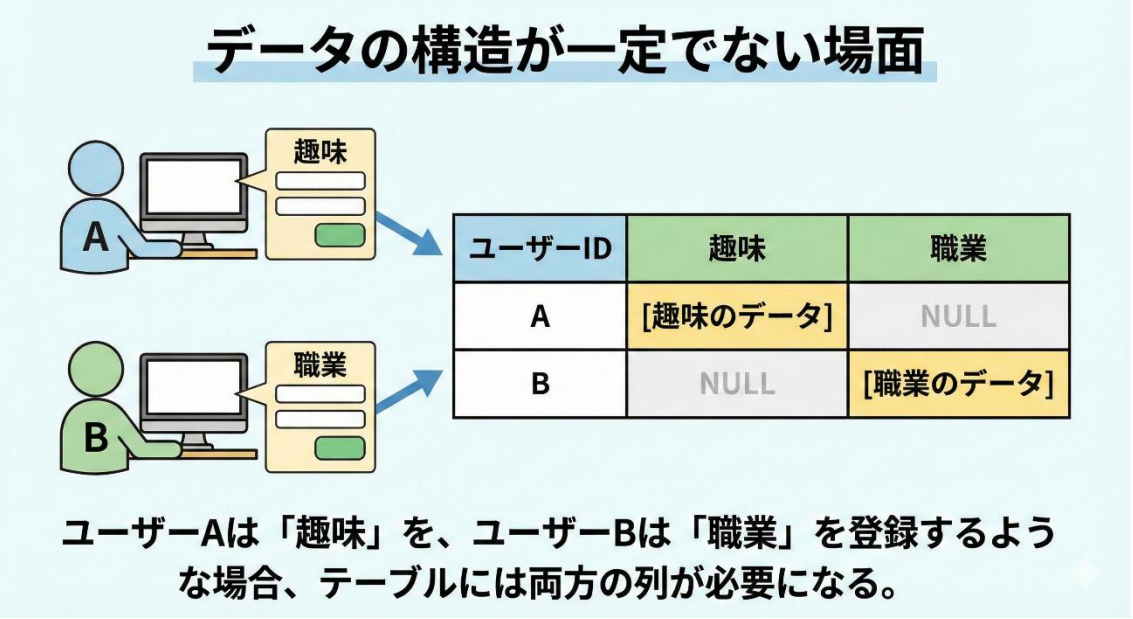
図解は Nano Banana Pro を用いて作成

- リレーショナルデータベースでは、1件ごとに**テーブルの制約を確認する処理**が発生。このような大量データの高速処理には**工夫が必要。**

リレーショナルデータベースが苦手なこと

②　データの構造が一定でない場面

- **ユーザーA**は「**趣味**」を、**ユーザーB**は「**職業**」を登録するような場合、**テーブルには両方の列が必要**になる。



図解は Nano Banana Pro を用いて作成

- 列数が増加、**新しい項目を追加**するたびに**テーブル構造の変更**が必要。

NoSQLデータベースとは

- **NoSQL（Not Only SQL）**は、**リレーショナルデータベース以外のデータベースの総称**である。

【基本的な考え方】

- テーブル形式にこだわらない
- **データ構造を柔軟に変更可能**
- **単純な読み書きを高速に処理**

【注意】

- NoSQLはリレーショナルデータベースを「置き換える」ものではない。それぞれに得意な場面があり、目的に応じて「使い分ける」ものである。

データベースシステムの種類

① リレーショナルデータベースシステム

	路線コード番号	事業者コード番号	路線名称一般	路線名称一般カナ
1	1001	3	中央新幹線	チュウオウシンカンセン
2	1002	3	東海道新幹線	トウカイドウシンカンセン
3	1003	4	山陽新幹線	サンヨウシンカンセン
4	1004	2	東北新幹線	トウホクシンカンセン

テーブル

② ドキュメント指向のデータベースシステム

JSONなどの**ドキュメント**に特化.

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="CGImap
0.0.2">
 <bounds minlat="54.0889580"
minlon="12.2487570" maxlat="54.0913900
maxlon="12.2524800"/>
 <node id="298884269" lat="54.0901746"
lon="12.2482632" user="SvenHRO"
uid="46882" visible="true" version="1"
changeset="676636" timestamp="2008-09-
```

ドキュメント

③ キー・バリュー形式のデータベースシステム

キー（鍵）　バリュー（値）

45343430	金子邦彦

キー・バリュー

その他、カラム指向型、グラフ型など

NoSQLの代表的な形式

本講義では、このうち**キー・バリュー型**と**ドキュメント型**を学ぶ。

形式	特徴
キー・バリュー型	最も単純な形式。**高速な読み書き**が可能
ドキュメント型	JSON形式でデータを格納。**柔軟な構造に対応**

まず、**キー・バリュー型**を説明する。
その後、**ドキュメント型**の**前提知識**として**JSON**を学び、ドキュメント型を説明する。

キー・バリュー型

- キー・バリュー型は、「**キー（鍵）**」を指定すると「**バリュー（値）**」が得られる形式。

例　**キー**　　**バリュー**

user001 →　　山田太郎

user002 →　　鈴木花子

count001 →　　12345

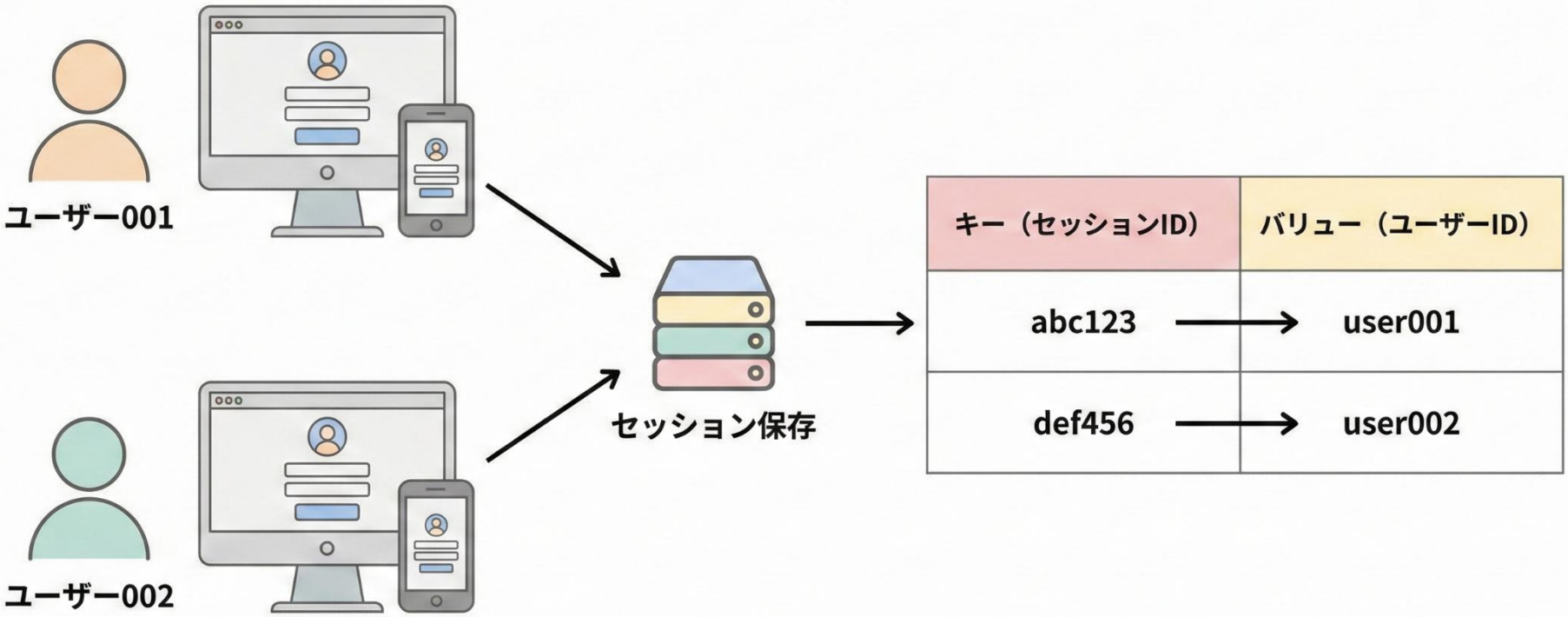
図解は Nano Banana Pro を用いて作成

考え方：ロッカーの番号（キー）を指定すると、中の荷物（バリュー）を取り出せる。

特徴：**構造が単純**で、**キー**から**直接バリュー**を**特定**できるため、**高速に取り出せる。**

キー・バリュー型の用途

（例）ログイン状態の管理

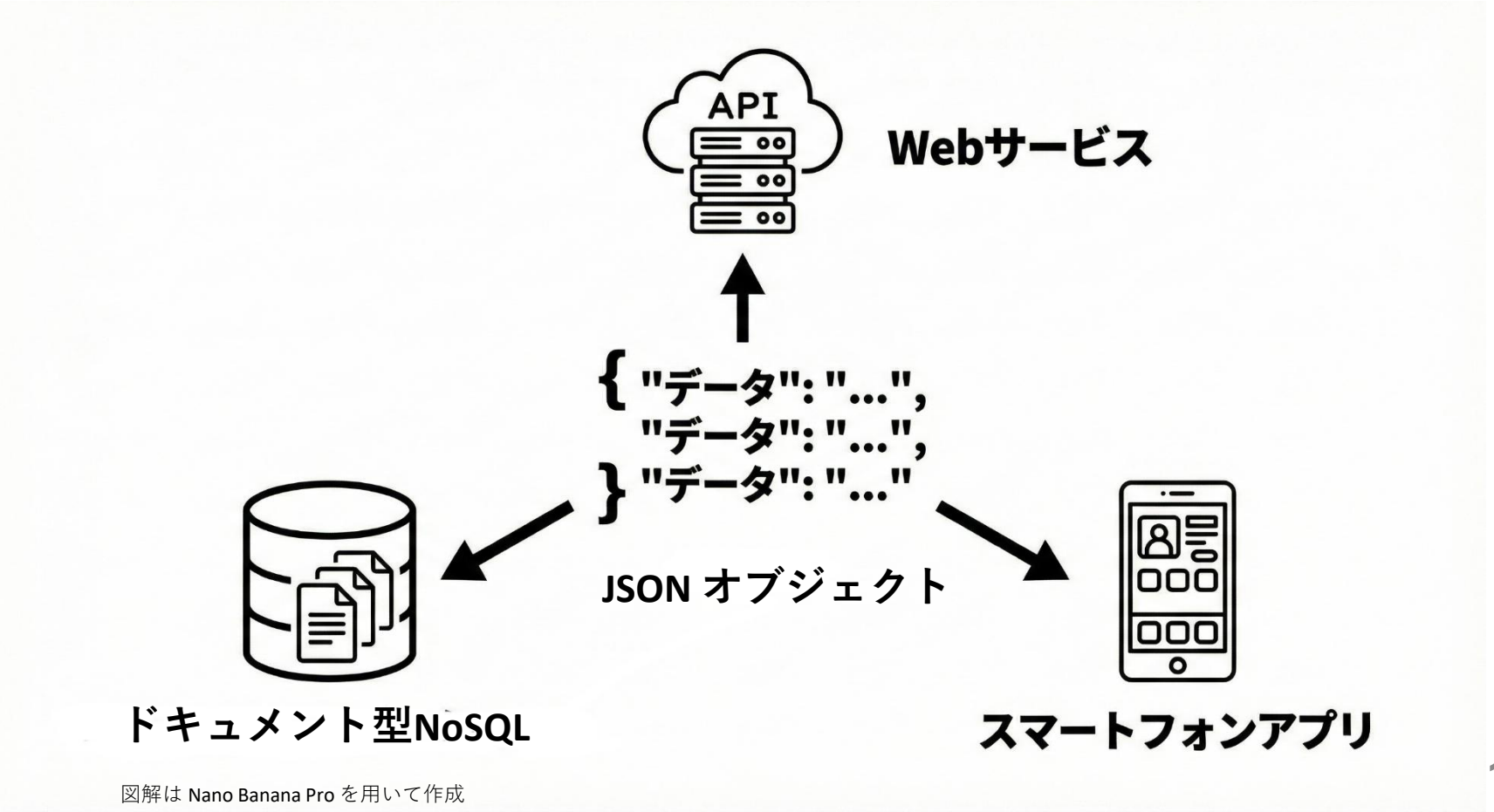


図解は Nano Banana Pro を用いて作成

大量の単純なデータを高速に読み書きする場面

さまざまなシステムのデータ交換に使用されるJSON

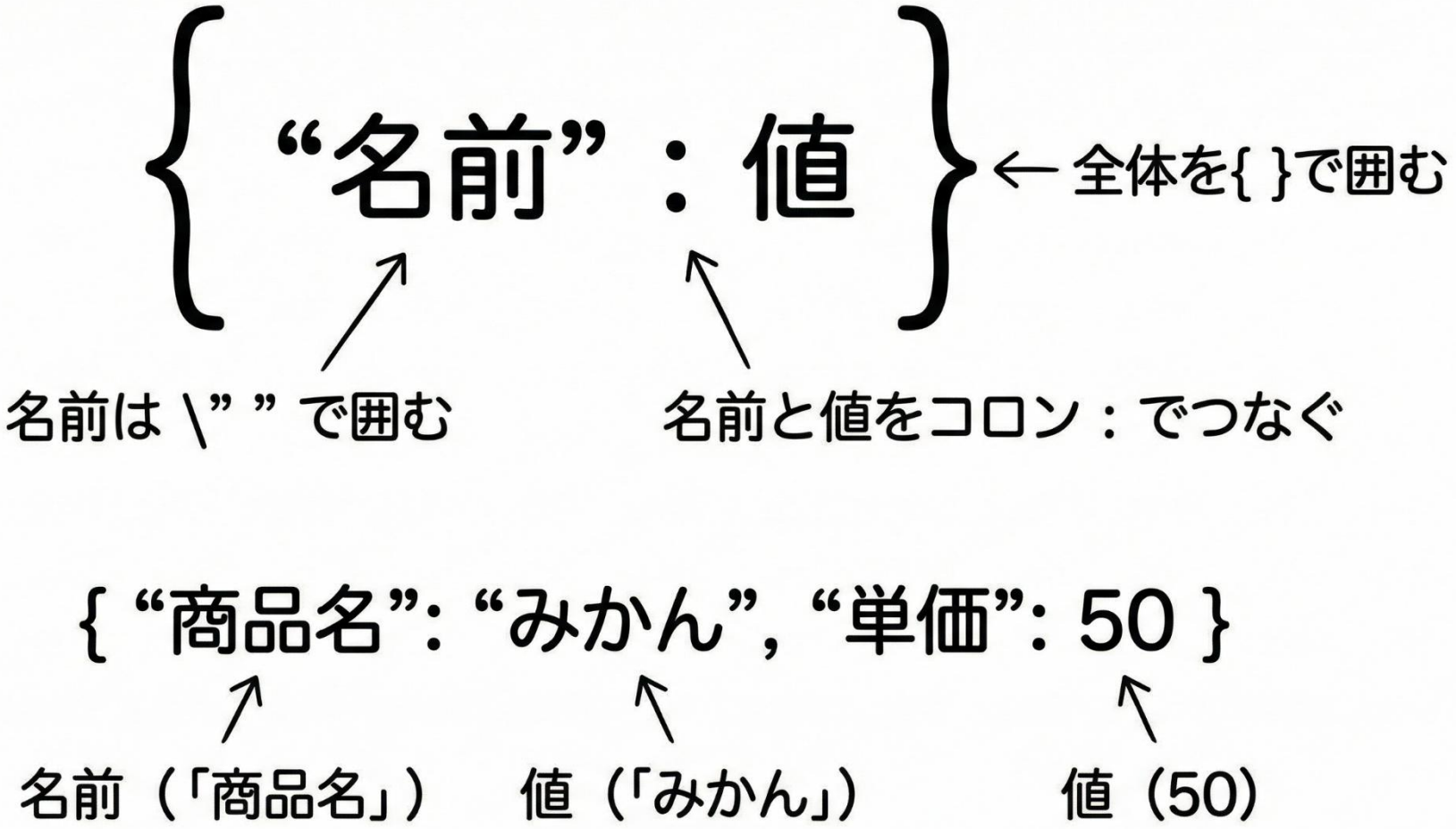
ドキュメント型NoSQL の基礎となる JSON を説明する



図解は Nano Banana Pro を用いて作成

JSON の書き方

JSONは「**名前**」と「**値**」の**組み合わせ**で**データを表現**する。



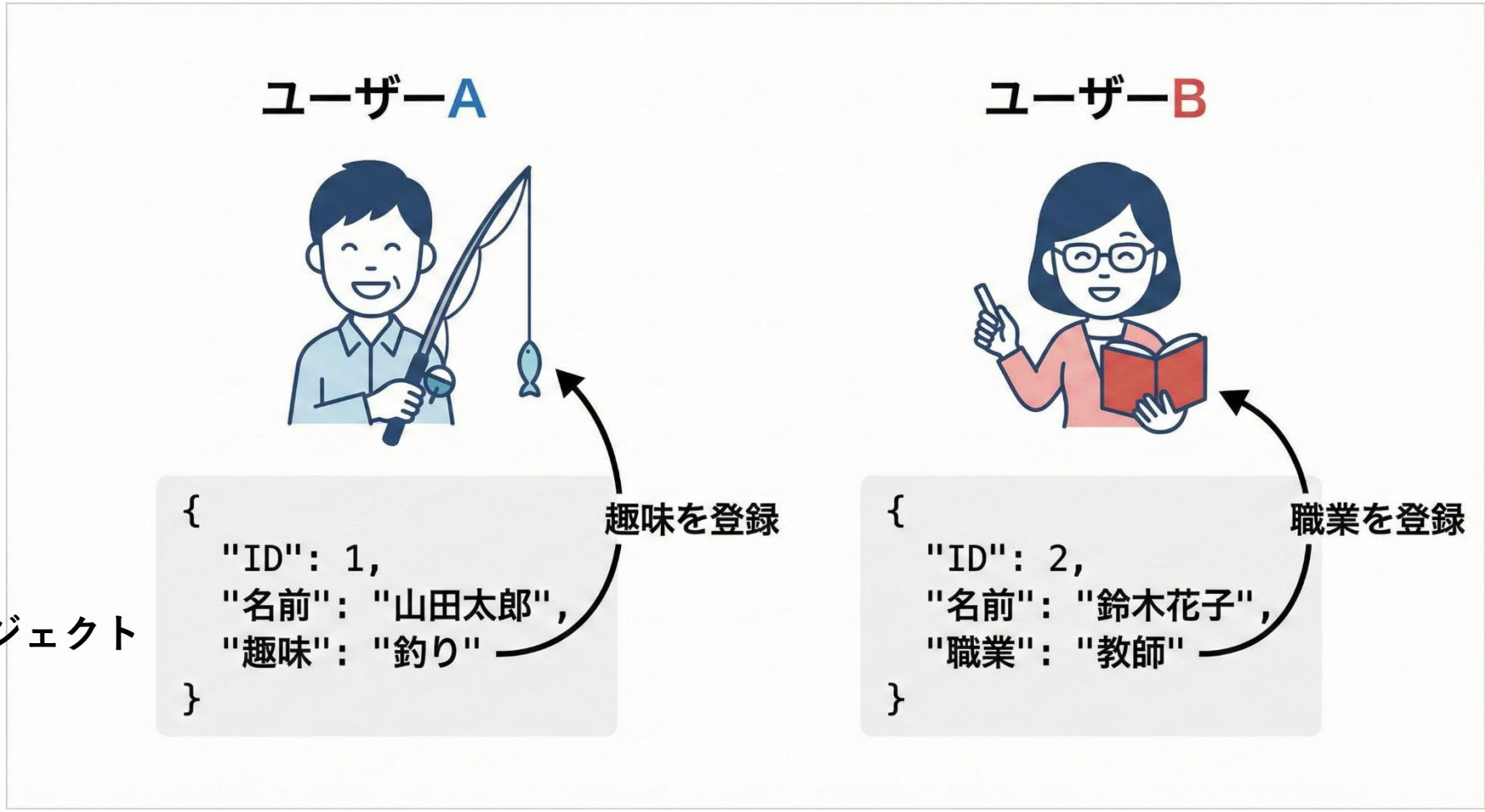
図解は Nano Banana Pro を用いて作成

JSONの値の種類

JSONの値には**種類**がある。基本は「文字列」と「数値」。

種類	書き方	例
文字列	**ダブルクォーテーションで囲む**	**"みかん"**
数値	**そのまま書く**	**50**

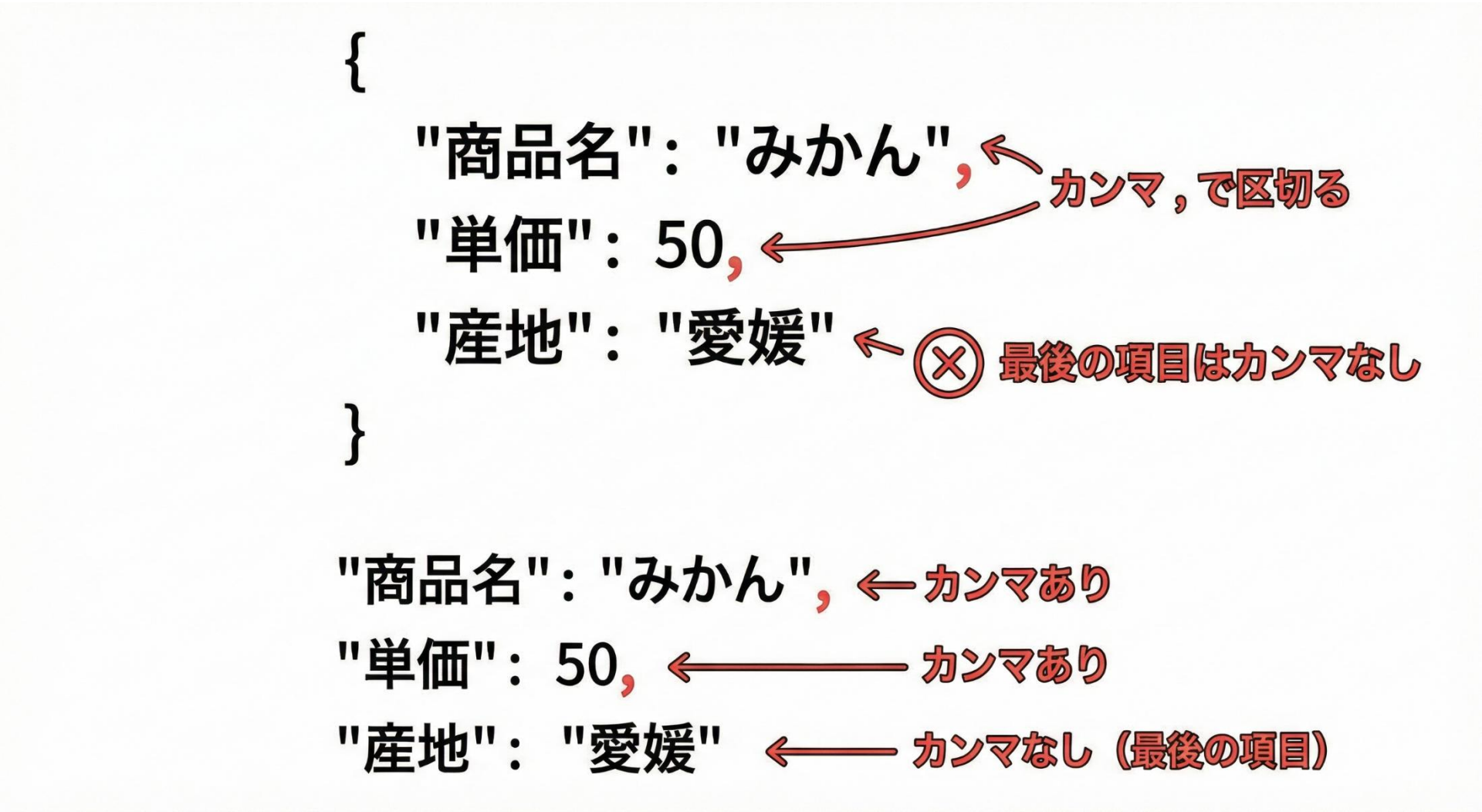
JSON の実例



JSON
オブジェクト

図解は Nano Banana Pro を用いて作成

JSONの複数項目

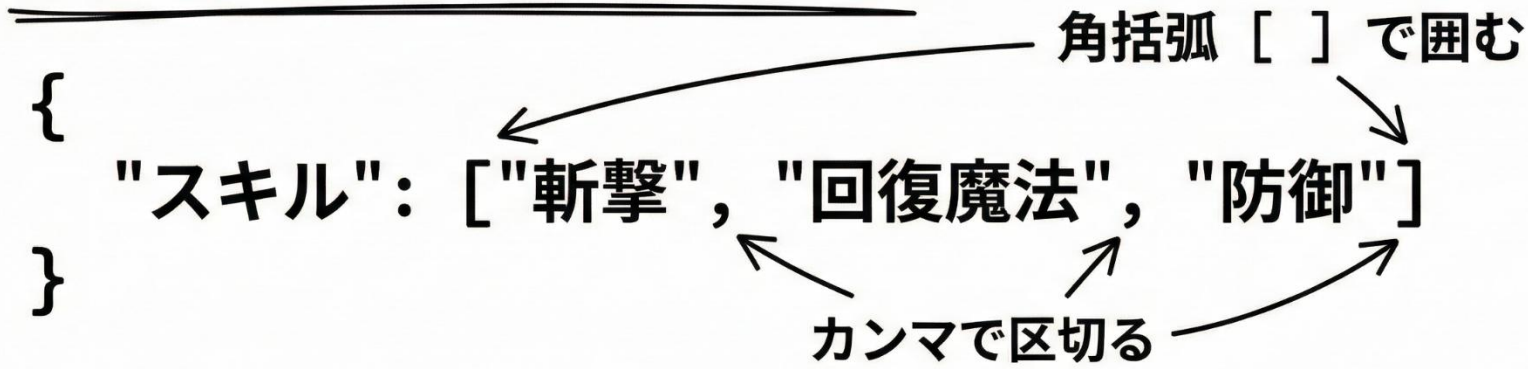


図解は Nano Banana Pro を用いて作成

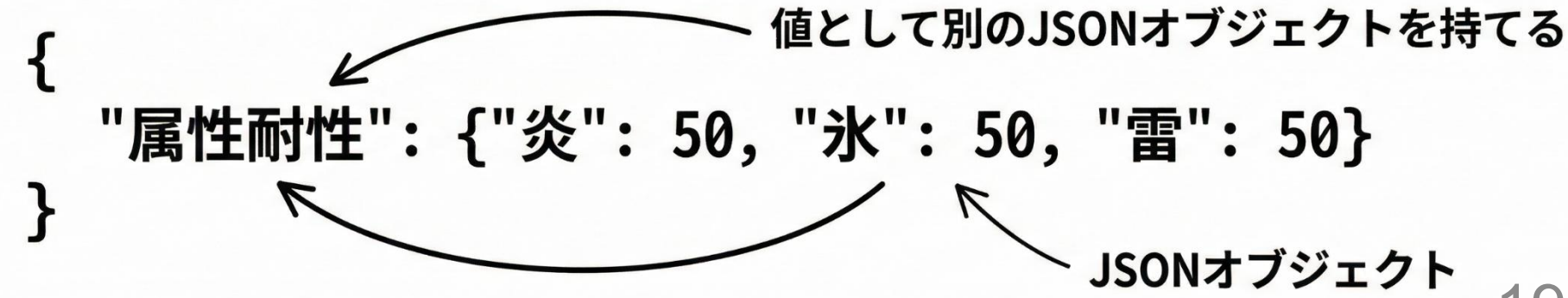
JSONの配列とネスト

配列とネストを組み合わせることで、**複雑なデータも表現可能。**

配列（複数の値をまとめる）

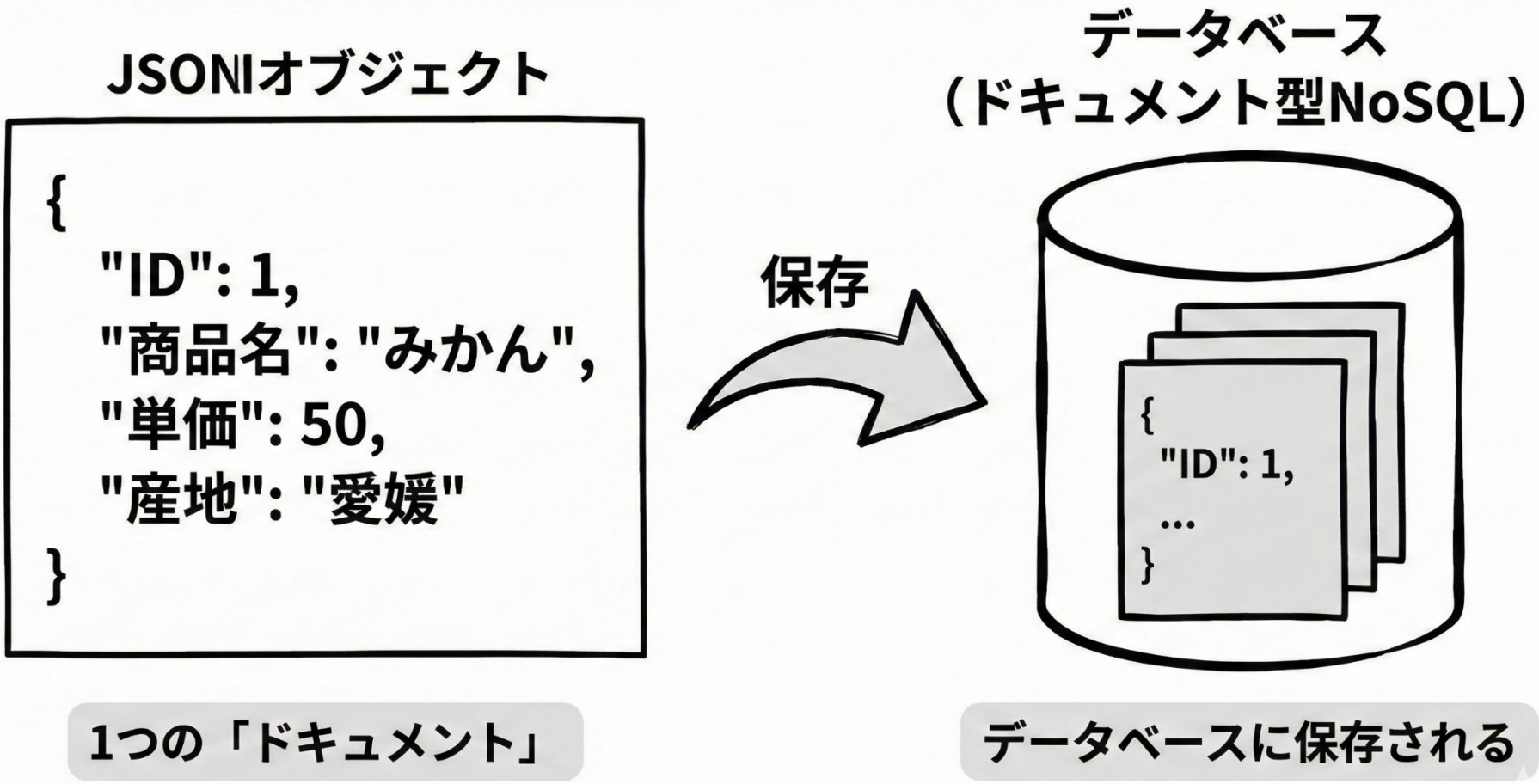


ネスト（JSONオブジェクトの中にJSONオブジェクト）



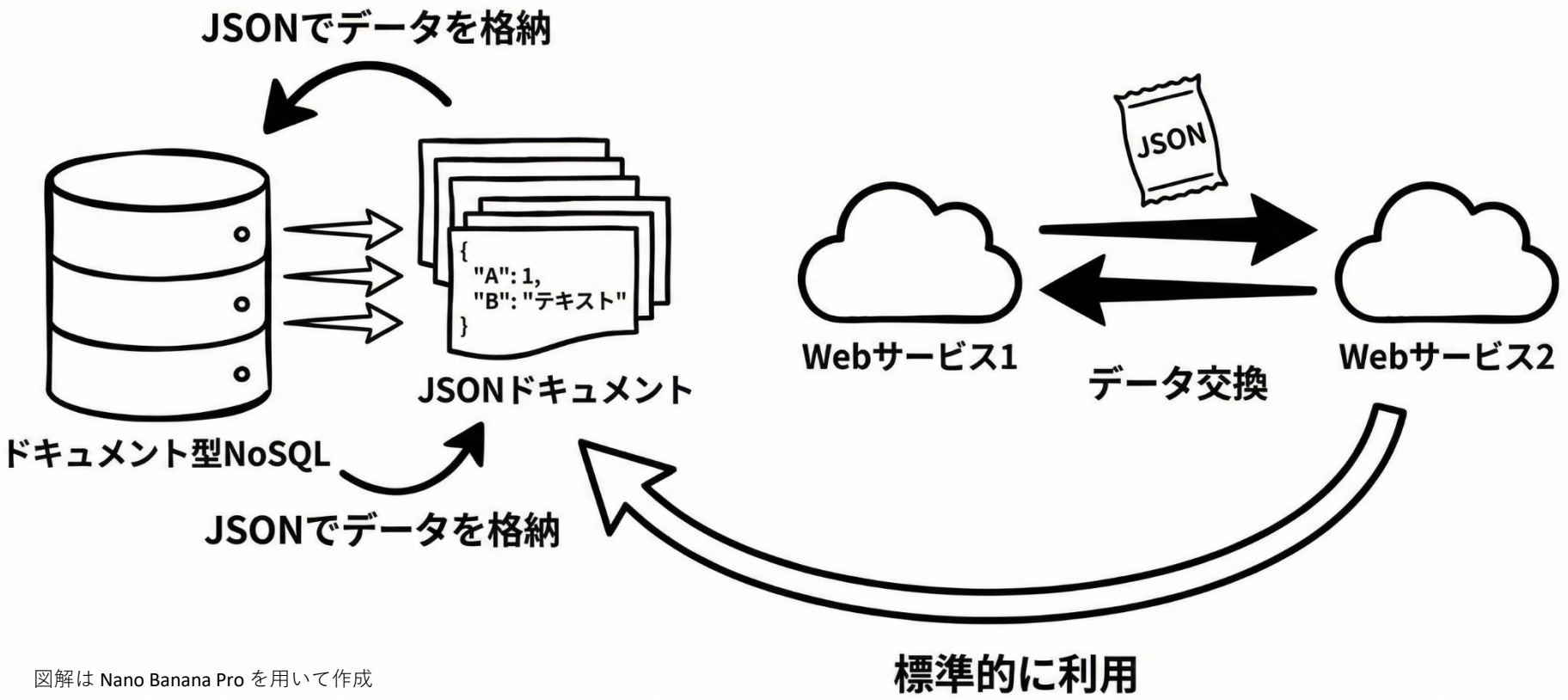
図解は Nano Banana Pro を用いて作成

ドキュメント型NoSQL と JSON



図解は Nano Banana Pro を用いて作成

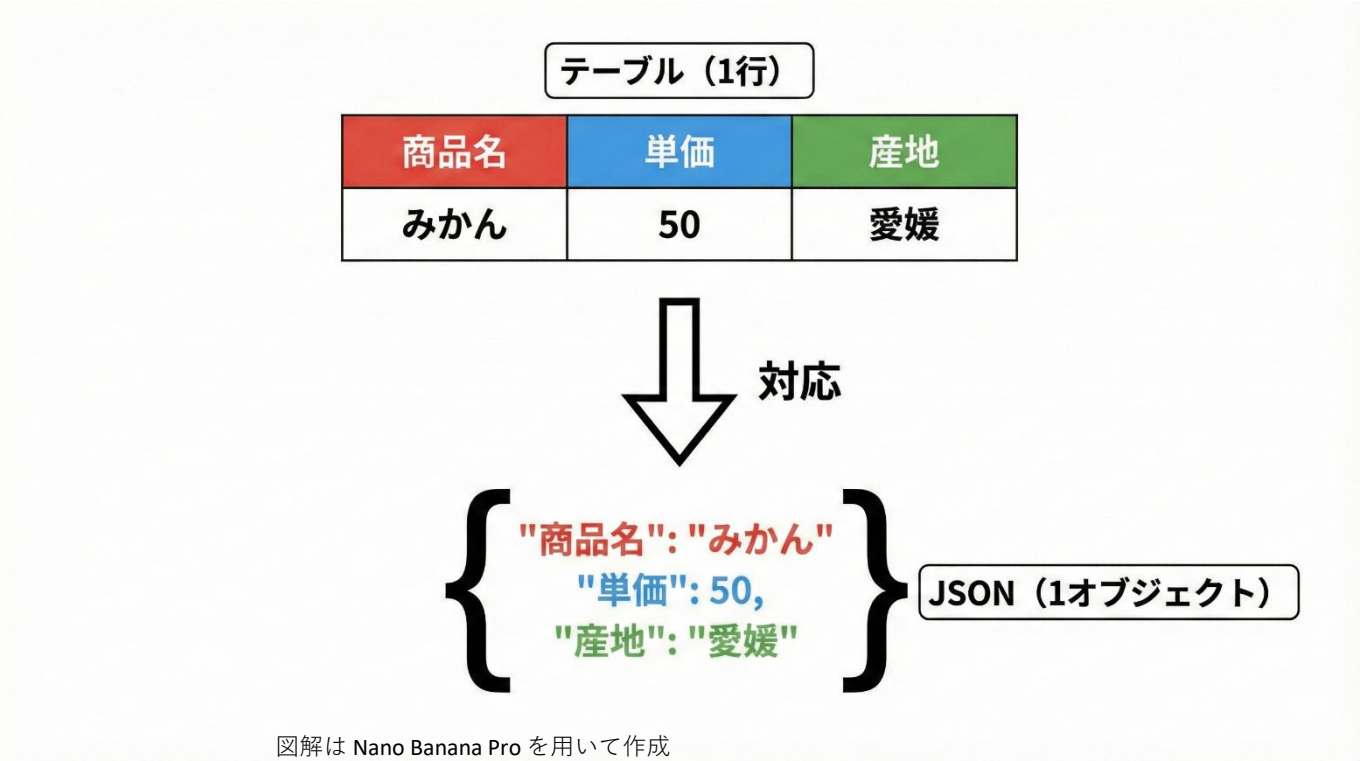
JSOによるドキュメント型 NoSQL と Web サービスの連携



図解は Nano Banana Pro を用いて作成

JSONとテーブルの対応

同じデータを、**テーブル**と**JSON**の両方で表現できる。

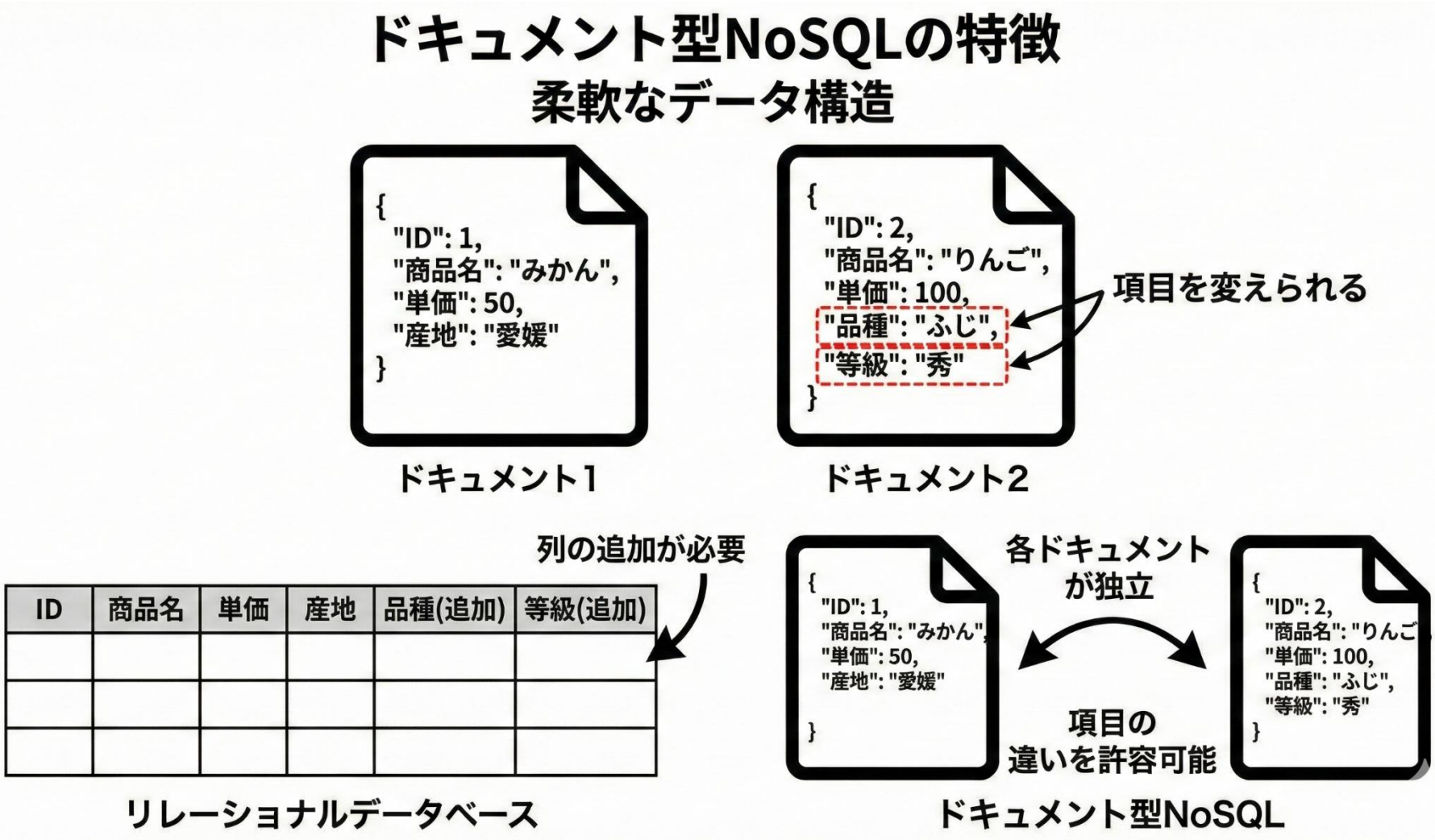


図解は Nano Banana Pro を用いて作成

対応関係

- テーブルの1行 → 1つのJSONオブジェクト
- 列名 → JSONの名前
- セルの値 → JSONの値

ドキュメント型 NoSQL の特徴



図解は Nano Banana Pro を用いて作成

リレーショナルデータベースとNoSQLの比較

観点	リレーショナルデータベース	NoSQL
データ形式	**テーブル（行と列）**	**キー・バリュー、JSONなど**
構造の変更	**列の追加が必要**	**柔軟に対応可能**
複雑な問い合わせ	**得意**	**苦手な場合が多い**
大量データの単純処理	**負担が大きい**	**得意**
データの正確性保証	**得意（ACID特性を完全にサポート）**	**限定的（一部のみサポート、または結果整合性を採用）**

- 補足：**ACID特性**

リレーショナルデータベースは、**原子性**（Atomicity）、**一貫性**（Consistency）、**独立性**（Isolation）、**永続性**（Durability）の4つの特性を備えた**トランザクション処理**をサポートする。NoSQLの多くは、処理速度を優先するため、これらの**特性の一部**を**緩和**している。

使い分けの考え方

リレーショナルデータベースを選ぶ場面

- 複数のテーブルを関連付けて管理したい
- データの信頼性を保証する必要がある（銀行、在庫管理、履修登録）
- 複雑な条件での検索や集計が必要

NoSQLを選ぶ場面

- **大量のデータ**を**高速**に読み書きしたい（**SNSの「いいね」**）
- **データの構造が決まっていない**、または**変化する**
- **単純なアクセス**が中心（**IDでデータを取得**するなど）

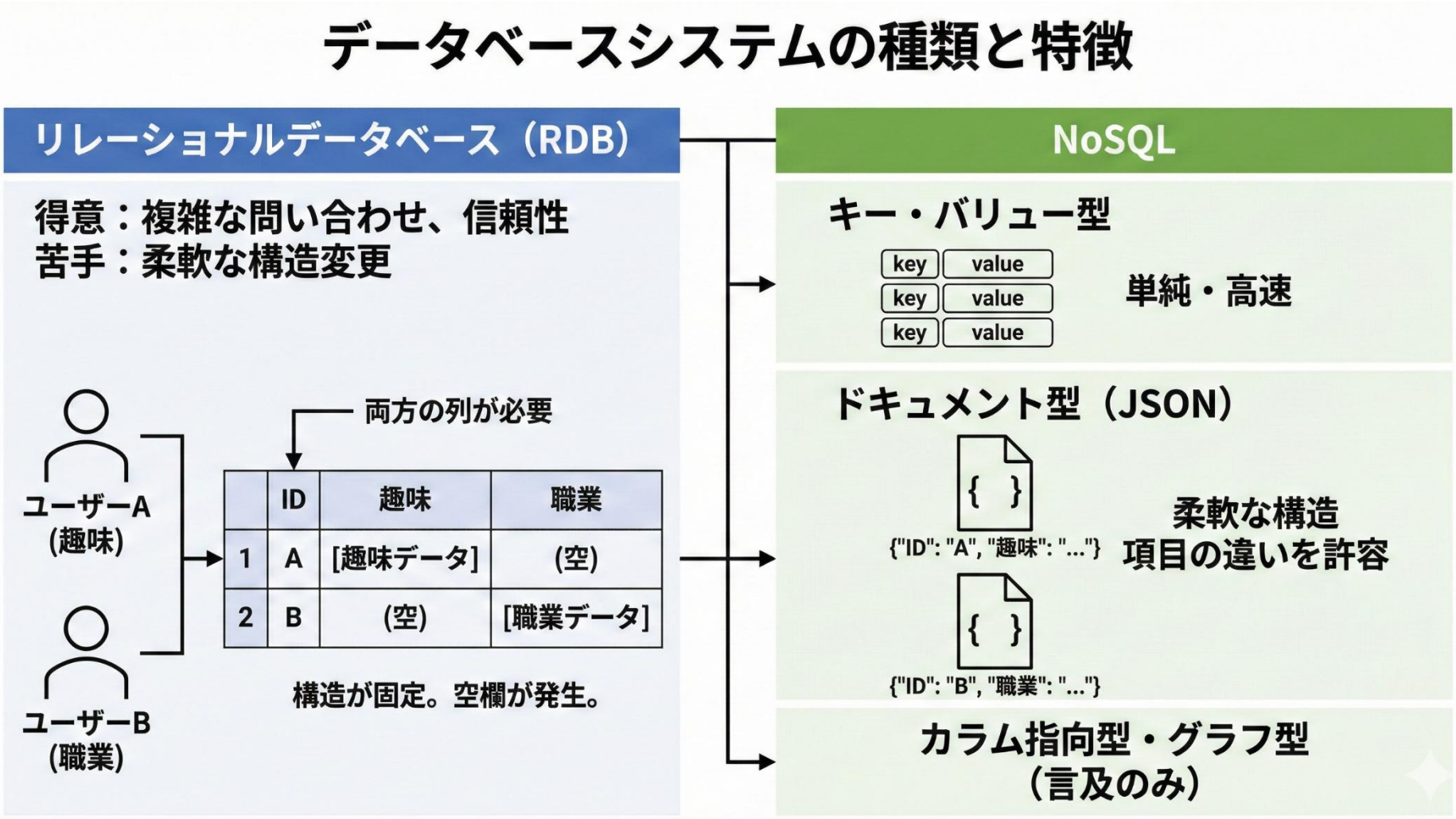
実際のシステムでの利用例

実際のシステムでは、**リレーショナルデータベース**と**NoSQL**を**組み合わせ**て利用することが多い。

データの種類	選択するデータベース	理由
会員情報、注文履歴	**リレーショナルデータベース**	**データの信頼性の保証が必要**
「いいね」の数、閲覧履歴	**NoSQL**	**大量データ、高速処理**

データの性質に応じて、適切なデータベースを選択する。

全体まとめ


データベースシステムの種類と特徴
リレーショナルデータベース（RDB）
得意：複雑な問い合わせ、信頼性
苦手：柔軟な構造変更
両方の列が必要
ユーザーA
(趣味)
ユーザーB
(職業)
ID
趣味
職業
1
A
[趣味データ]
(空)
2
B
(空)
[職業データ]
構造が固定。空欄が発生。
NoSQL
キー・バリュー型
key
value
key
value
key
value
単純・高速
ドキュメント型（JSON）
{ }
{"ID": "A", "趣味": "..."}
{ }
{"ID": "B", "職業": "..."}
柔軟な構造
項目の違いを許容
カラム指向型・グラフ型
（言及のみ）

演習の目的

演習の目的

- リレーショナルデータベースとNoSQL（JSON形式）の両方でデータ表現を体験する。

この演習で確認すること

- リレーショナルデータベースにおけるテーブル設計
- JSONの構文
- テーブルのデータをJSON形式で表現

演習のねらい

- **同じデータ**を**リレーショナルデータベース**用と**NoSQL**用の両方で設計し、**それぞれの特徴**を実感

ゲームのキャラクターデータを題材とする

キャラクターごとに**装備数**、**スキル数**、**属性耐性の有無**が異なる

キャラクター	HP	MP	装備	スキル	属性耐性
勇者	**120**	**30**	**鉄の剣、革の鎧、力の指輪**	**斬撃、回復魔法、防御**	**なし**
魔法使い	**60**	**100**	**樫の杖、魔法のローブ**	**ファイア、ブリザド、サンダー、ケアル、バリア**	**炎:50、氷:50、雷:50**
盗賊	**80**	**20**	**短剣、黒装束**	**二刀流、隠密、毒攻撃、盗む**	**なし**

問1

- このデータをリレーショナルデータベースで管理するためのテーブル設計を示しなさい。

ヒント

- 1つのセルに複数の値（装備やスキル）を入れることはできない
- 繰り返し項目は別テーブルに分離する

問2

- 勇者と魔法使いのデータをJSON形式で記述しなさい。属性耐性を持たないキャラクターについては、「属性耐性」の項目を省略してよい。

ヒント

- 配列とネストを活用する

解答（問1）キャラクターテーブル

基本情報を管理するテーブルを作成する。

（装備、スキル、属性耐性は繰り返し項目のため、別テーブルに分離）

キャラクターテーブル

ID	名前	HP	MP
1	**勇者**	**120**	**30**
2	**魔法使い**	**60**	**100**
3	**盗賊**	**80**	**20**

解答（問1）装備・スキルテーブル

装備テーブル

キャラクターID	装備名
1	鉄の剣
1	革の鎧
1	力の指輪
2	樫の杖
2	魔法のローブ
3	短剣
3	黒装束

スキルテーブル

キャラクターID	スキル名
1	斬撃
1	回復魔法
1	防御
2	ファイア
2	ブリザド
2	サンダー
2	ケアル
2	バリア
3	二刀流
3	隠密
3	毒攻撃
3	盗む

解答（問1）属性耐性テーブル

属性耐性テーブル

キャラクターID	属性	耐性値
2	炎	50
2	氷	50
2	雷	50

設計のポイント

- **合計4つのテーブル**が必要
- キャラクターIDで各テーブルを**関連付け**
- 勇者と盗賊は属性耐性を持たないため、属性耐性テーブルには登場しない

解答（問2）JSON形式

勇者

{ "名前": "勇者", "HP": 120, "MP": 30, "装備": ["鉄の剣", "革の鎧", "力の指輪"], "スキル": ["斬撃", "回復魔法", "防御"] }

魔法使い

{ "名前": "魔法使い", "HP": 60, "MP": 100, "装備": ["樫の杖", "魔法のローブ"], "スキル": ["ファイア", "ブリザド", "サンダー", "ケアル", "バリア"], "属性耐性": {"炎": 50, "氷": 50, "雷": 50} }

盗賊

{ "名前": "盗賊", "HP": 80, "MP": 20, "装備": ["短剣", "黒装束"], "スキル": ["二刀流", "隠密", "毒攻撃", "盗む"] }

補足

- 勇者は属性耐性を持たないため、「属性耐性」の項目を省略している。これはドキュメント型の柔軟性を示す例である