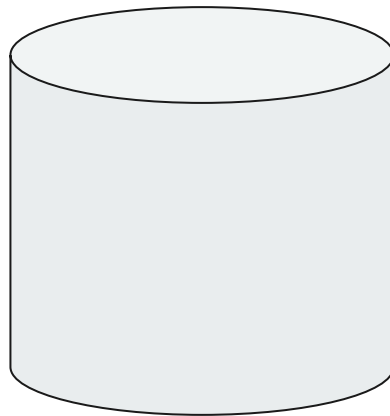


データベースとは？

大量の情報を保存し、コンピュータから
効率よくアクセスできるように加工した
データのあつまり



データベースとは？

由来は第二次世界大戦後の米軍

情報のアクセスを効率化するために

点在していた資料をひとつの基地にすべて集めた

⇒ これを情報(**Data**)の基地(**Base**)と呼んだ





システムがあれば、裏にはデータベースあり

ECサイトやSNSなどデータを扱うシステムの
ほとんどすべてにデータベースが使われている

システムとデータベースは切っても切れない関係



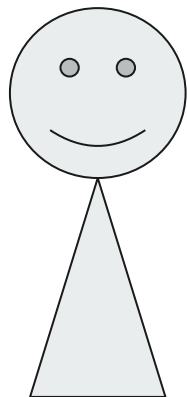
データベースの種類

- リレーショナルデータベース(RDB)
- グラフ型データベース
- キー・バリュー型データストア
- ドキュメント指向データベース

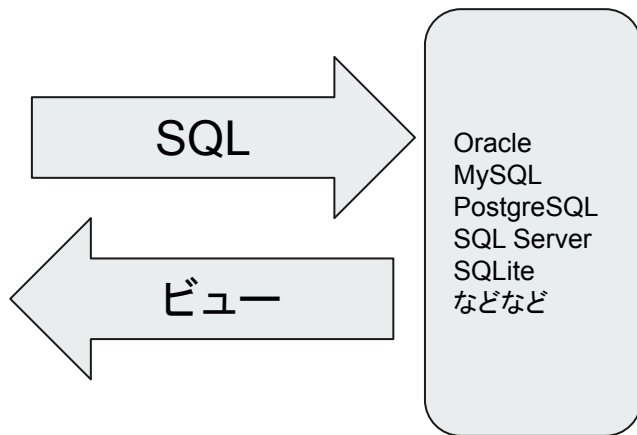
RDBが圧倒的に主流で、広く使われている

データベースの利用

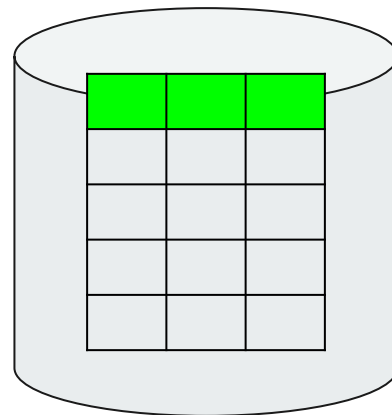
利用者



RDBMS



RDB
テーブル





データベース設計とは？

システム開発のひとつのプロセスで、

データベースに保持するデータに関する設計のこと

システムの拡張性やパフォーマンスに多大な影響を与える、システム開発において極めて重要なプロセス



システム開発のプロセス

1. 要件定義(なにをつくるか？)
2. 設計(どうつくるか？)←本コースのメインテーマ
3. 開発(実装)
4. テスト(期待通りに動くか？)



システム開発の設計のプロセスの内訳

- データベース設計(データの保持について決める)

↑ 本コースのテーマ

- アプリケーション設計(提供機能を決める)
- インターフェース設計(使用画面などを決める)
- など



DBの種類ごとに、DB設計のやり方は異なる

- リレーショナルデータベース(RDB)
- グラフ型データベース
- キー・バリュー型データストア

DBの種類ごとにデータベース設計のやり方は異なる

本コースでは、RDBの設計について学ぶ



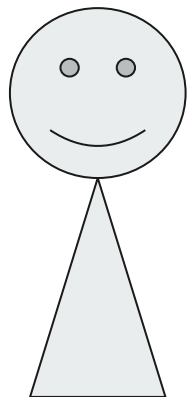
3層スキーマ

データベースは、3層のスキーマ(枠組み)からなる

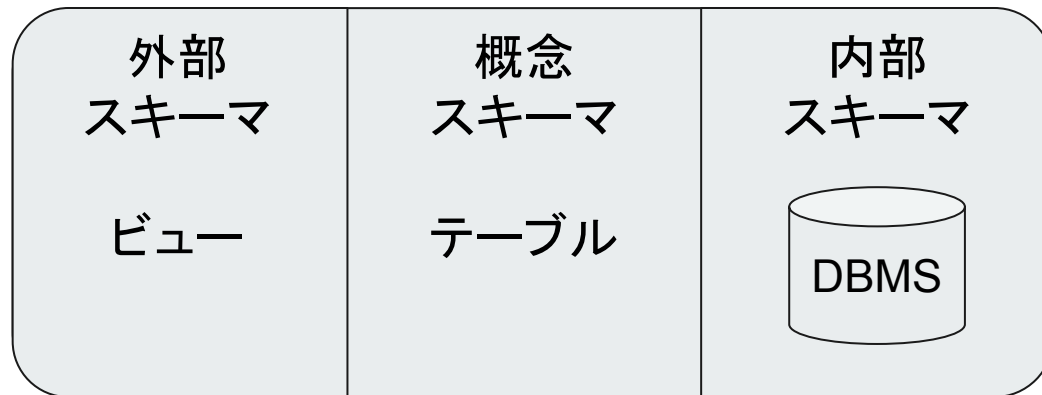
- 外部スキーマ(ユーザーから見たDB、ビュー)
- 概念スキーマ(開発者から見たDB、テーブル)
- 内部スキーマ(DBMSから見たDB、データの物理的配置)

3層スキーマのイメージ

利用者



システム



外部スキーマと概念スキーマ

ビュー(外部スキーマ)

name	height
佐藤	170.2
鈴木	151.5
高橋	182.1
田中	163.5
渡辺	157.8
伊藤	173.0
山本	166.4
中村	144.1
小林	168.7
加藤	178.6

SELECT name, height
FROM user;



テーブル(概念スキーマ)

id	name	height	weight	age	job_id
1	佐藤	170.2	65.2	60	4
2	鈴木	151.5	50.3	53	2
3	高橋	182.1	85.1	31	8
4	田中	163.5	70.6	36	3
5	渡辺	157.8	55.8	62	7
6	伊藤	173.0	65.3	75	1
7	山本	166.4	49.1	25	5
8	中村	144.1	56.9	45	7
9	小林	168.7	90.1	38	3
10	加藤	178.6	78.5	26	6

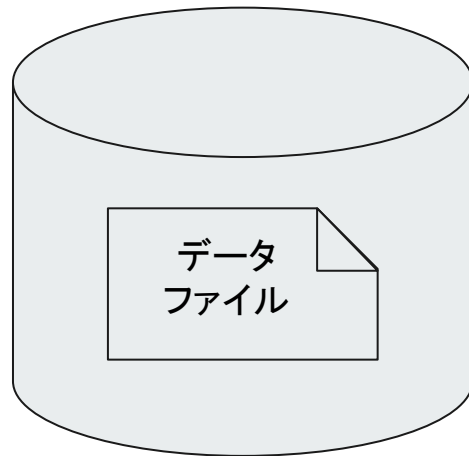
概念スキーマと内部スキーマ

テーブル(概念スキーマ)

id	name	height	weight	age	job_id
1	佐藤	170.2	65.2	60	4
2	鈴木	151.5	50.3	53	2
3	高橋	182.1	85.1	31	8
4	田中	163.5	70.6	36	3
5	渡辺	157.8	55.8	62	7
6	伊藤	173.0	65.3	75	1
7	山本	166.4	49.1	25	5
8	中村	144.1	56.9	45	7
9	小林	168.7	90.1	38	3
10	加藤	178.6	78.5	26	6

データファイル(内部スキーマ)

データファイルを
テーブルとして
扱えるように設定





論理設計と物理設計

論理設計(概念スキーマの設計)

→ データを管理するためのデータモデルの設計

物理設計(内部スキーマの設計)

→ DDLによる実装やストレージの構成などの設計



概念スキーマはなぜ必要？

データファイル(内部スキーマ)から直接

ビュー(外部スキーマ)を取り出すでも良い？

→ **データ独立性**を確保するのに、

概念スキーマは絶対に必要



データ独立性とは？

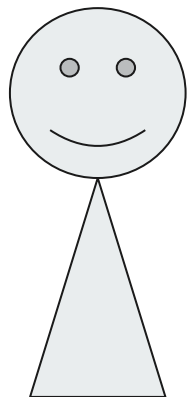
外部スキーマを変更しても、内部スキーマを変更する必要がない状態(**論理的データ独立性**)

もしくは、内部スキーマを変更しても、外部スキーマを変更する必要がない状態(**物理的データ独立性**)

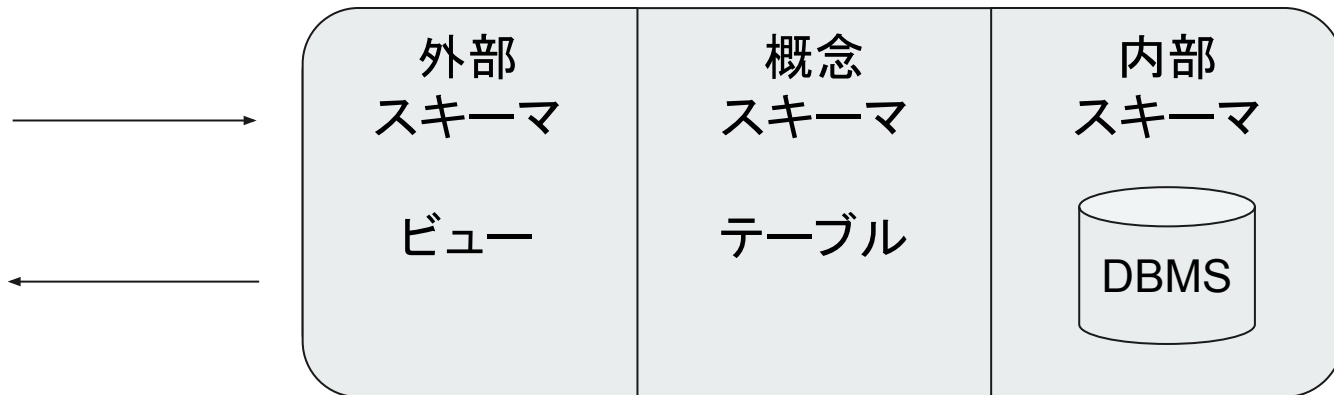
のことを、データ独立性があるという


3層スキーマによるデータ独立性

利用者



システム





3層スキーマのまとめ

- DBは外部スキーマ、概念スキーマ、内部スキーマの3層からなる
- RDBにおいてそれぞれは、ビュー、テーブル、データの物理的な配置に対応する
- 概念スキーマが他のスキーマの変更を吸収する緩衝材の役割を担うことで、データの独立性が確保できる



論理設計は超重要

逆に言うと、概念スキーマを変更すると、

外部スキーマにも内部スキーマにも影響が出る

→ 概念スキーマの設計(論理設計)は超重要

→ 質の高い論理設計のために、知識を身に着けよう



コラム:3層スキーマの別の解釈

概念スキーマは内部スキーマを、**隠**ぺいしていると考え
こともできる

データの物理的な配置は、ユーザーには理解しづらい

→ テーブルというわかりやすいインターフェースに変換す
ることで、扱いやすくしているとも考えられる