データベース
【1:データベースシステムとは】

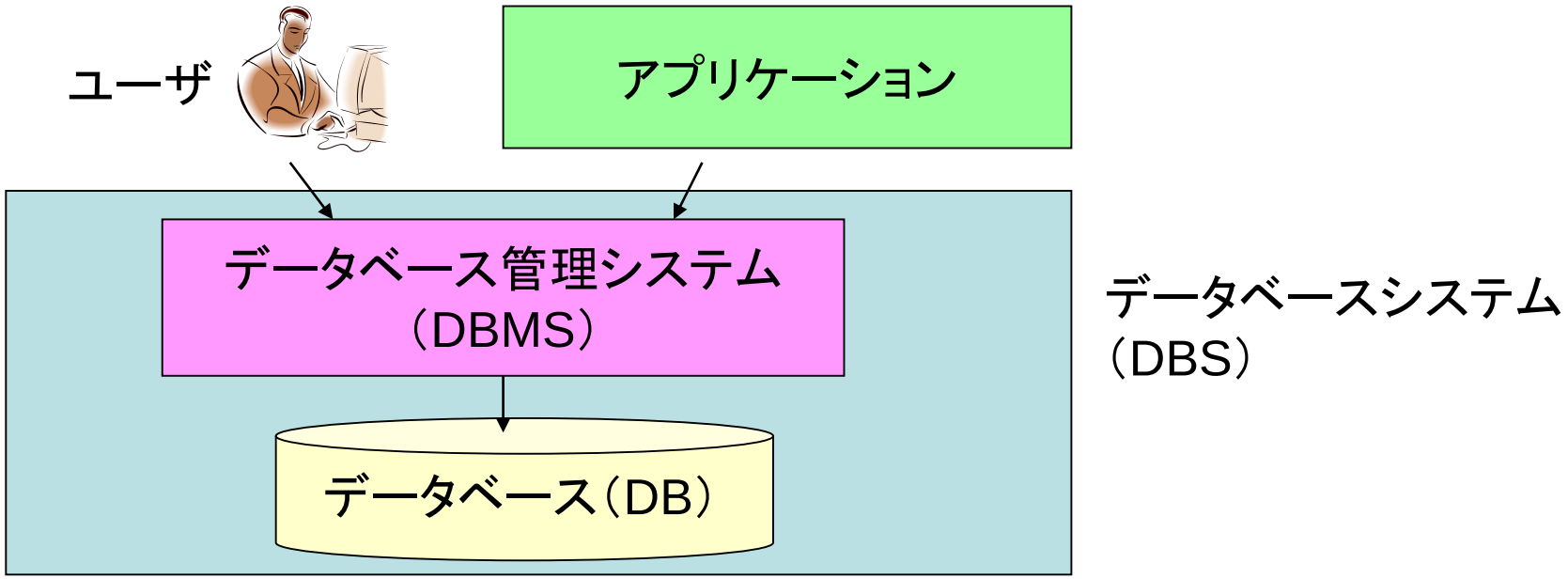
石川佳治

データベースシステムとは

- データベースシステム（database system）
  - 各種アプリケーションが扱うデータ資源を統合して蓄積管理
  - 効率的な共有，高度な利用
- アプリケーションシステムの例
  - ウェブサイト：ショッピングサイトなど
  - 人事管理，成績管理システム
- データベース（database, DB）
  - 複数の応用目的での共有を意図して組織的かつ永続的に格納されたデータ群

データベース管理システム（1）

- データベース管理システム（database management system, DBMS）
  - データベースを管理するためのソフトウェア
  - データベースシステム：DBMSとそれにより管理されるデータ群



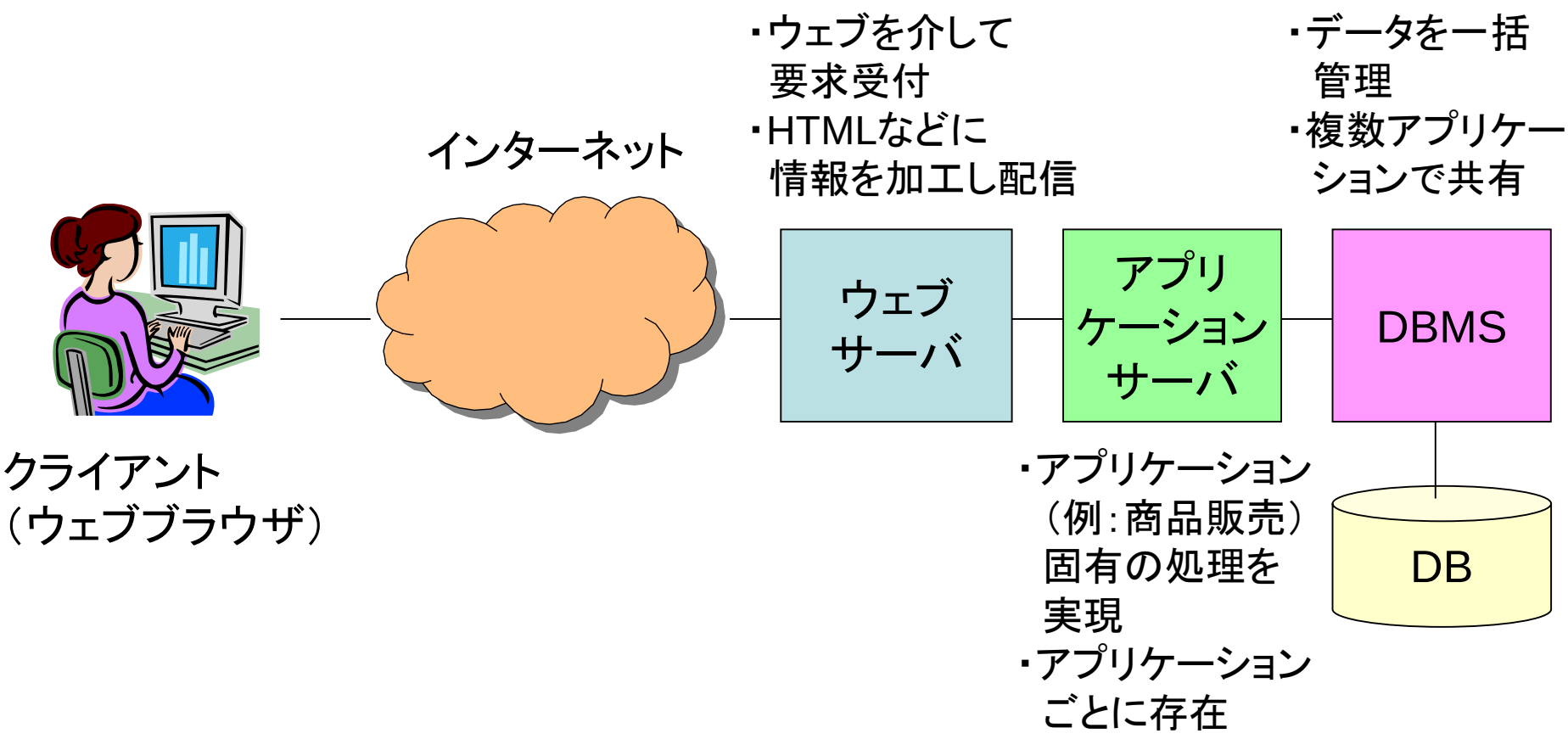
データベース管理システム（2）

- 商用のDBMS
  - Oracle
  - DB2（IBM）
  - SQL Server（Microsoft）
  - HiRDB（日立）
- フリーのDBMS
  - PostgreSQL
  - MySQL
  - Firebird
  - SQLite：組込み用DBMS

3階層モデル（three tier model）（1）

- インターネットサービスで一般的なシステム構成
- システムの拡張性，柔軟性を実現
- 3階層のシステム構成
  - プレゼンテーション層（フロントエンド）
    - ウェブサーバによる実現
    - ユーザインタフェースを実現
  - アプリケーション層（ミッドティア）
    - アプリケーションサーバによる実現
    - アプリケーション固有の処理
  - データベース層（バックエンド）
    - DBMSによる実現
    - データを管理

3階層モデル(three tier model)(2)



データベースシステムの必要性(1)

- DBMSを用いないデータ管理にはいくつかの問題が存在
  - ファイルによるデータ管理など

1. データとアプリケーションの相互依存
   - アプリケーションは，データ格納やアクセス法の詳細を強く意識しなければならない
   - アプリケーションとファイル構造が相互依存
   - 異なったファイル構造で格納されたデータを利用できない
   - 不統一性，不整合性，冗長性の原因

データベースシステムの必要性(2)

2. 整合性維持機能の欠如

- 不正な更新，誤った入力の排除が難しい
- 例：複数ファイルで学生の学籍番号が不一致
- アプリケーションプログラムでチェックが必要

3. 不十分な機密保護

- 複数ユーザによるデータ共有：アクセス制御（読出し，書込み）が必要
- ファイル単位のアクセス制御は可能だが，データ単位のアクセス制御は困難

データベースシステムの必要性(3)

4. 複数ユーザの同時アクセス

- 複数ユーザによる更新を同時に行うとさまざまな異状，矛盾が発生
- 有用性を維持しつつ，複数ユーザへの対処を図ることが重要

5. 不十分な障害時データ保護

- システム障害への対応は容易ではない
  - プログラムのエラー
  - システムダウン
  - ディスククラッシュ
- 障害時にもデータを保護し，すみやかな復旧を図る機能が求められる

データベースシステムによるデータ管理

- DBMSを用いたデータ管理の利点
  - データをアプリケーションと独立に管理：多目的利用が可能
  - 関連するデータの統合による無用な重複や不整合の除去
  - データの意味やデータの相互関係の把握が容易になる
  - データの表現法やその管理方法を標準化しやすくなる

DBMSの機能（1）

1. データ記述・操作系

- 対象データとそれに対する操作に関する共通の枠組み「データモデル」を提供
- データモデル（data model）
  - 論理的なレベルでのデータ記述と操作が可能
  - データの物理的な格納形態や検索手順の詳細に依存しない
- リレーショナルデータモデル
  - 代表的かつ最も利用されているデータモデル
  - データベースを表（リレーション）の集まりと捉える

リレーショナルデータモデル

- データベースを論理的な表として表現

学生

学籍番号	氏名	学部	住所
123	山田一郎	工学部	名古屋市
159	田中花子	工学部	一宮市
201	鈴木二郎	理学部	名古屋市

教員

名前	所属	役職
高橋	工学部	教授
中村	工学部	助教授
伊藤	理学部	教授

指導教員

教員	学籍番号
高橋	123
高橋	159
伊藤	201

DBMSの機能（2）

2. 効率の良いデータアクセス機構

- データに合った効率よい格納方式を利用
- 検索処理の効率化のための索引を設定
- 二次記憶へのアクセス時間を減らす
- 問合せ最適化：与えられた問合せに対して，効率的な処理手続きをプランニング

3. 整合性の保持

- データモデルを用いてデータの構造や関連を明示的に管理
- 整合性の制約の維持管理をDBMSに任せる
- 例：学籍番号の一意性のチェックをDBMSに

DBMSの機能（3）

4. 機密保護
   - 各ユーザが可能な操作をきめ細かく指定可能
   - 検索，修正，挿入，削除など
5. 同時実行制御（concurrency control）
   - トランザクション機能を支援
   - トランザクション（transaction）：アプリケーションから見たときのひとまとまりの処理単位
     - 商品販売の例：①在庫チェック，②在庫数を-1，③売上げ記載
   - 複数トランザクションの同時実行：誤った更新を防ぐ
6. 障害回復（recovery）
   - 何らかの障害でトランザクションが中止された場合，状態を復元，データの整合性をとる

データベースシステムに関する基本概念

- スキーマとインスタンス
- 抽象化の3レベル
- データ独立性
- データベース言語

スキーマとインスタンス

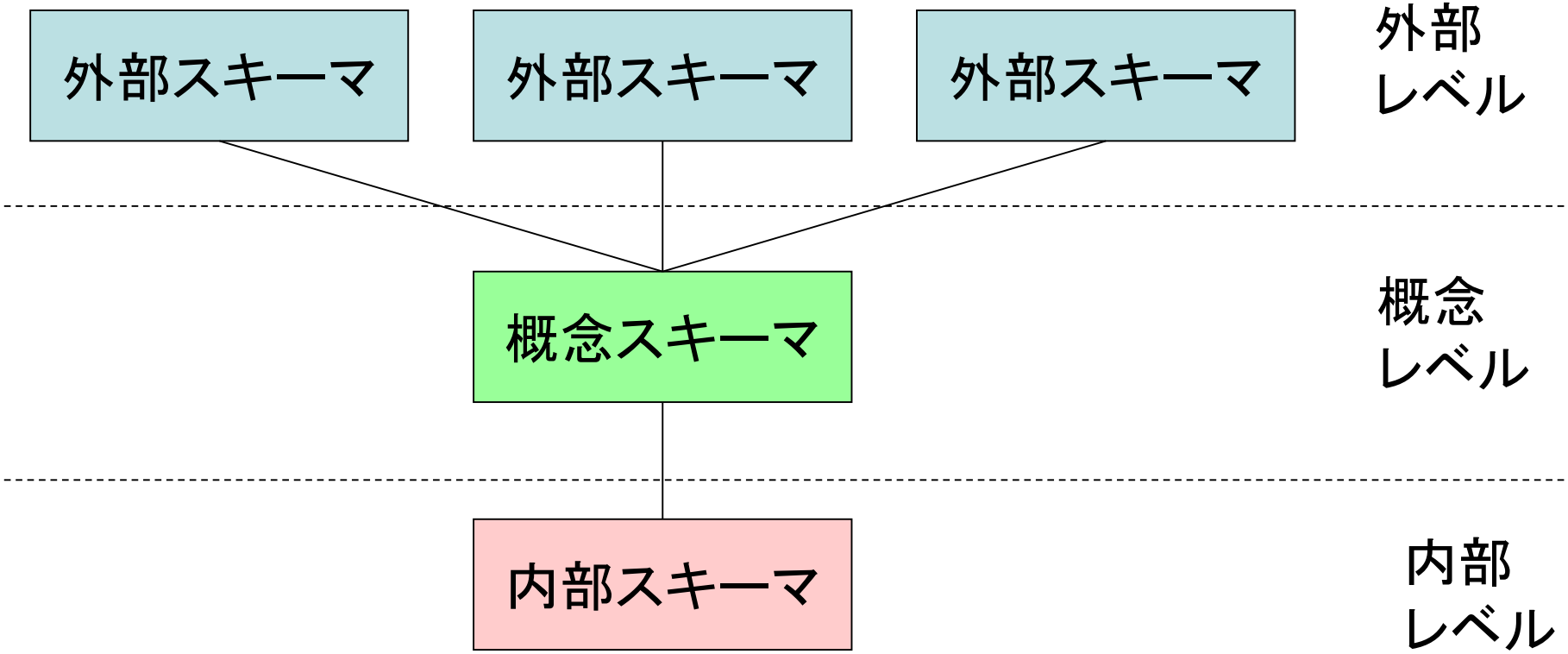
- スキーマ（schema）
  - データベース中のデータの構造，形式，関連，整合性制約などを記述したもの
  - 例：学生リレーションは学籍番号，氏名，学部，住所からなる
- インスタンス（instance）
  - スキーマに基づくデータ群
  - リレーショナルデータベースの場合，表に含まれるデータを指す

抽象化の3レベル（1）

- 内部レベル（internal level）
  - 物理的な格納のレベル
  - 内部スキーマ（internal schema）で構成を規定
- 概念レベル（conceptual level）
  - データベース全体を論理的に記述
  - 概念スキーマ（conceptual schema）
- 外部レベル（external level）
  - アプリケーションごとのデータベースシステムへの視点に対応
  - 外部スキーマ（external schema）
  - ビュー（view）とも呼ぶ

抽象化の3レベル（2）

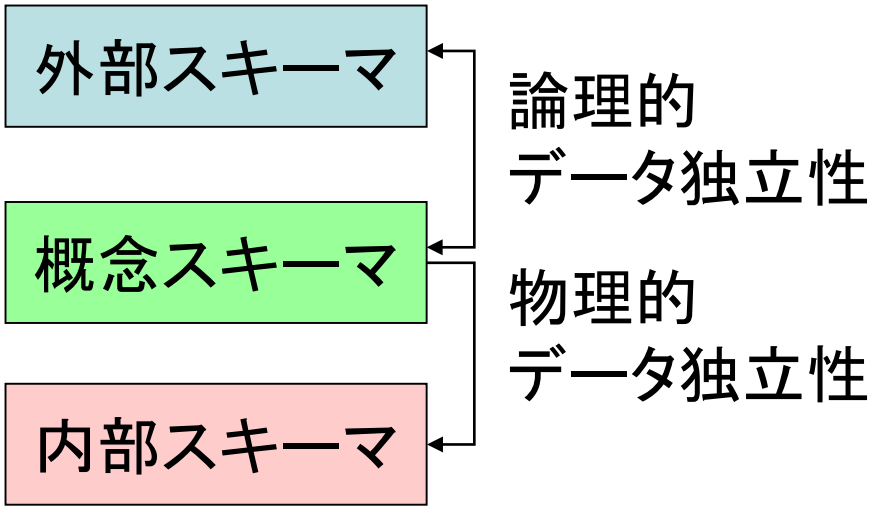
ANSI/SPARCモデル



データ独立性（data independence）

- データをアプリケーションプログラムから分離して組織化
- 2種類のデータ独立性
  - 論理的データ独立性（logical data independence）
  - 物理的データ独立性（physical data independence）

- 例：概念スキーマを変更する場合
  - 実世界の要求により
  - 外部スキーマに影響を与えない範囲であれば，アプリケーションは独立



データベース言語

- データベース言語（database language）
  - データモデルに基づくデータ記述ならびにデータ操作のための言語
  - DBMSが機能を提供
- データ定義言語（data definition language, DDL）
  - スキーマ記述を行う
- データ操作言語（data manipulation language, DML）
  - インスタンス操作を行う言語
- SQL
  - 標準データベース言語
  - データ定義と操作の機能を併せ持つ

データベースシステムの構成

ユーザ

問合せ

アプリケーション
プログラマ

アプリケーションプログラム

データベース
管理者（DBA）

スキーマ

データ操作言語処理系

データ定義言語処理系

同時実行
制御情報

データマネジャ

システム
カタログ

ファイルマネジャ

障害回復用
情報

データベース