

ソフトウェアアーキテクチャ

第14回 データベースシステム

環境情報学部

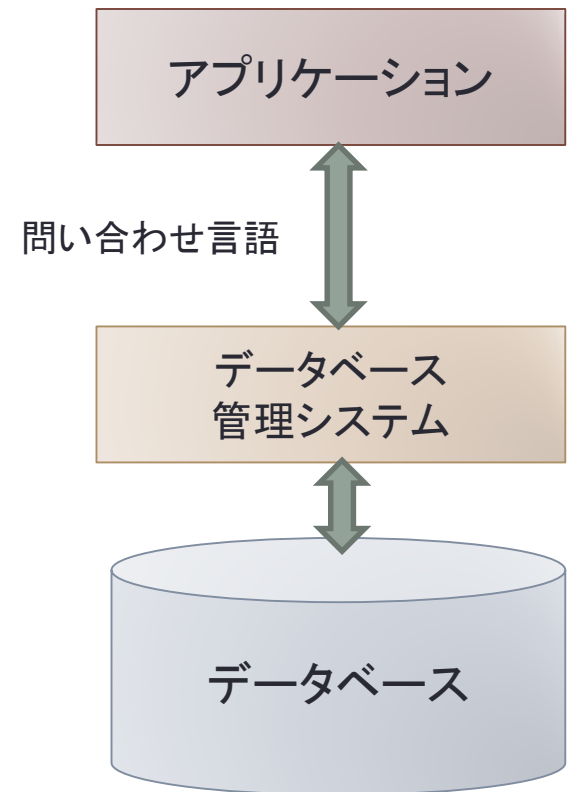
萩野 達也

lecture URL

<https://vu5.sfc.keio.ac.jp/slide/>

データベースシステム

- データベース
 - データをある形式にしたがって集めたもの
 - 検索などの処理が容易
- データベースシステムの種類
 - 関係データベース
 - オブジェクトデータベース
 - XML データベース
 - グラフデータベース
- データベース管理システム (DBMS)
 - データベースをコンピュータ上で管理するソフトウェア
 - Oracle Database
 - Microsoft SQL Server
 - PostgreSQL
 - MySQL
 - SQLite
 - IBM DB2
 - Infomix

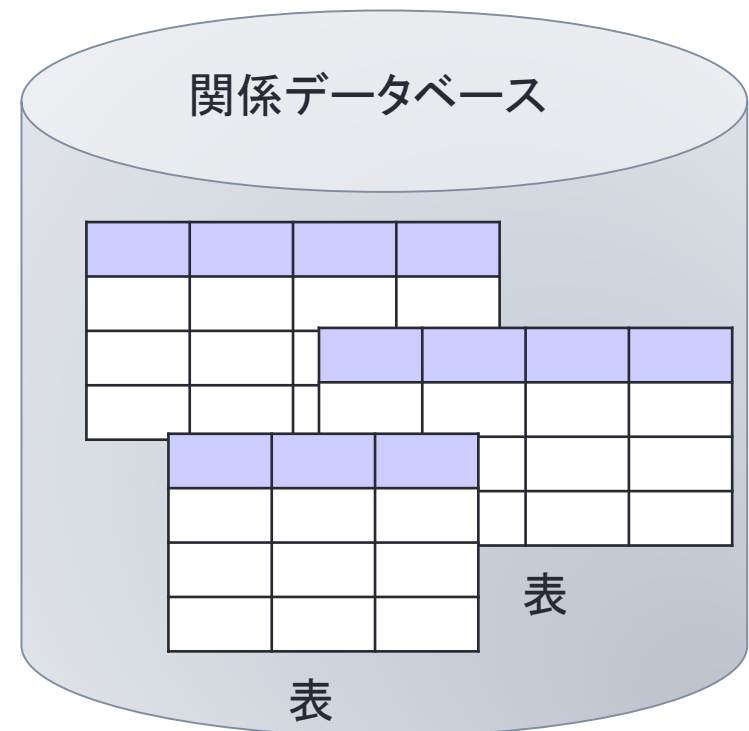


関係データベース

- 関係モデルに基づくデータベース
 - 関係 = 表
 - RDB = 表の集まり
- 関係演算により関係(表)を操作
 - 和(union)
 - 差(difference)
 - 交わり(intersection)
 - 制限(restriction, selection)
 - 射影(projection)
 - 結合(join)
- SQL
 - 関係演算に基づくRDBMS問い合わせ言語
 - INSERT
 - UPDATE
 - DELETE
 - SELECT

データベースモデル

- 階層モデル
- ネットワークモデル
- 関係モデル



例：授業履修者データ

履修者

テーブル名

属性名

属性(列)

組(行)

学籍番号	氏名	分野	科目名	学期	曜日	時限	教員名
9012345	藤沢太郎	先端	ソフトウェアアーキテクチャ	2022年春	月	3	萩野達也
9023456	遠藤花子	創造	数学と論理	2021年秋	火	2	青山敦
9012345	藤沢太郎	導入	情報基礎1	2021年秋	金	4	服部隆志
9023456	遠藤花子	先端	ソフトウェアアーキテクチャ	2021年春	月	3	萩野達也
:	:	:	:	:	:	:	:

- 関係をテーブル(表)としてあらわす
 - 属性(列)は順不同
 - それぞれの組(行)も順不同
 - 一つのセルには一つの値しか入れることはできない
 - 「情報基礎1」が4・5限の時には、「時限」に4と5を入れるわけにはいかない
 - 第1正規形

表の適切な分割・管理

- 関数従属関係にしたがって表を分割する
 - 一つの表では重複する部分があるなど, 更新が大変
 - 正規化
 - テーブルの組(行)を決めている属性(列)を主キーとする

履修者

学籍番号	授業コード
9012345	202201
9023456	202108
9012345	202102
9023456	202101
:	:

複合キー

授業

授業コード	科目コード	学期	曜日	時限	教員番号
202201	1234	2022年春	月	3	019
202108	2345	2021年秋	火	2	001
202102	3456	2021年秋	金	4	203
202101	1234	2021年春	月	3	019
:	:	:	:	:	:

主キー

学生

学籍番号	氏名
9012345	藤沢太郎
9023456	遠藤花子
:	:

主キー

科目

科目コード	分野	科目名
1234	先端	ソフトウェアアーキテクチャ
2345	創造	数学と論理
3456	導入	情報基礎1
:	:	:

主キー

教員

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志
:	:

主キー

テーブルの定義

```
CREATE TABLE 学生 (
  学籍番号 INTEGER PRIMARY KEY,
  氏名 VARCHAR(20));

CREATE TABLE 教員 (
  教員番号 INTEGER PRIMARY KEY,
  氏名 VARCHAR(20));

CREATE TABLE 科目 (
  科目コード INTEGER PRIMARY KEY,
  分野 CHAR(5),
  科目名 VARCHAR(20));

CREATE TABLE 授業 (
  授業コード INTEGER PRIMARY KEY,
  科目コード INTEGER
    REFERENCES 科目 (科目コード),
  学期 CHAR(6),
  曜日 CHAR(2),
  時限 INTEGER);

CREATE TABLE 履修者 (
  授業コード INTEGER
    REFERENCES 授業 (授業コード),
  学籍番号 INTEGER
    REFERENCES 学生 (学籍番号),
  PRIMARY KEY (授業コード, 学籍番号));
```

学生

学籍番号	氏名
9012345	藤沢太郎
9023456	遠藤花子
:	:

教員

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志
:	:

科目

科目コード	分野	科目名
1234	先端	ソフトウェアアーキテクチャ
2345	創造	数学と論理
3456	導入	情報基礎1
:	:	:

授業

授業コード	科目コード	学期	曜日	時限	教員番号
202201	1234	2022年春	月	3	019
202108	2345	2021年秋	火	2	001
202102	3456	2021年秋	金	4	203
202101	1234	2021年春	月	3	019
:	:	:	:	:	:

履修者

学籍番号	授業コード
9012345	202101
9023456	202008
9012345	202002
9023456	202001
:	:

関係演算

- 制限(restriction, selection)
 - 条件を満たす組(行)だけを取り出す

履修者

学籍番号	授業コード
9012345	202201
9023456	202108
9012345	202102
9023456	202101
:	:

学籍番号=9012345



学籍番号	授業コード
9012345	202201
9012345	202102
:	:

```
SELECT * FROM 履修者 WHERE 学籍番号='9012345'
```

- 射影(projection)
 - 指定した属性(列)だけを取り出す

授業

授業コード	科目コード	学期	曜日	時限	教員番号
202201	1234	2022年春	月	3	019
202108	2345	2021年秋	火	2	001
202102	3456	2021年秋	金	4	203
202101	1234	2021年春	月	3	019
:	:	:	:	:	:

科目コード,学期



科目コード	学期
1234	2022年春
2345	2021年秋
3456	2021年秋
1234	2021年春
:	:

```
SELECT 科目コード,学期 FROM 授業
```

関係演算(結合)

- 結合(join)

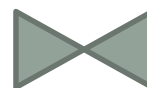
- 2つのテーブルの共通する値を使って表を結合する

授業

授業コード	科目コード	学期	曜日	時限	教員番号
202201	1234	2022年春	月	3	019
202108	2345	2021年秋	火	2	001
202102	3456	2021年秋	金	4	203
202101	1234	2021年春	月	3	019
:	:	:	:	:	:

教員

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志
:	:



授業と教員のテーブルを
教員番号で結合

授業コード	科目コード	学期	曜日	時限	教員番号	教員名
202201	1234	2022年春	月	3	019	萩野達也
202108	2345	2021年秋	火	2	001	青山敦
202102	3456	2021年秋	金	4	203	服部隆志
202101	1234	2021年春	月	3	019	萩野達也
:	:	:	:	:	:	:

```
SELECT * FROM 授業 JOIN 教員 ON (教員番号)
```

テーブルの更新

• 新しい組の挿入

- `INSERT INTO 教員 (教員番号, 氏名) VALUES ('002', '中村修')`
- 主キーは一意的でなくてはならない

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志

挿入



教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志
002	中村修

• 値の変更

- `UPDATE 教員 SET 氏名='はぎの' WHERE 教員番号='019'`

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志

更新



教員番号	教員名
019	はぎの
001	青山敦
203	服部隆志

• 組(行)の削除

- `DELETE FROM 教員 WHERE 教員番号='001'`
- 他のテーブルで参照されている組(行)の削除に注意

教員番号	教員名
019	萩野達也
001	青山敦
203	服部隆志

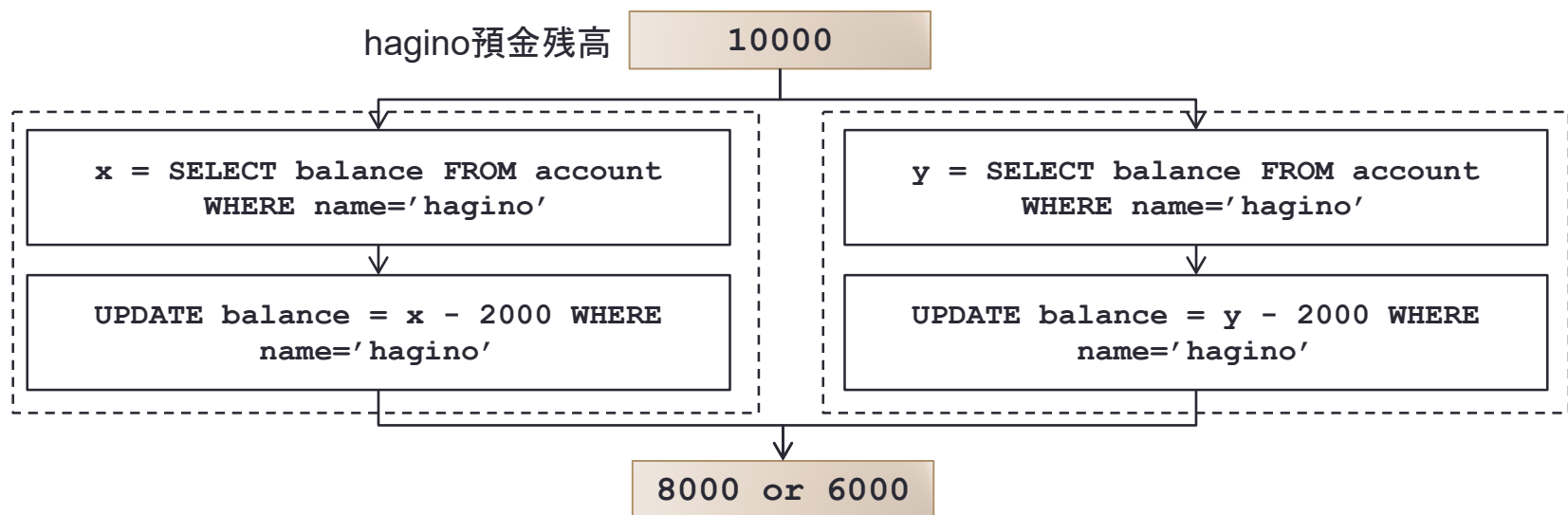
削除



教員番号	教員名
019	萩野達也
203	服部隆志

テーブル更新時の注意

- 同じ組(行)を複数回更新を行なうとおかしくなることがある
 - 銀行口座から同時に2000円引き出す



- ロック
 - 変更を行なう場合にはテーブルなどをロックすることが必要
 - LOCK
- トランザクション
 - クリティカルセクションを決める
 - BEGIN
 - COMMIT(END)
 - ROLLBACK(ABORT)

```
BEGIN
  LOCK TABLE account
  x = SELECT balance FROM account WHERE name='hagino'
  UPDATE balance = x - 2000 WHERE name='hagino'
COMMIT
```

ロックの種類

- テーブルロック
 - 変更するテーブル全体をロック
 - ロックの管理が簡単
 - 同時読み書きが多い場合にパフォーマンスに問題
- 行ロック
 - テーブルの行毎にロックする
 - ロックの競合が少なくなる
 - バグの発生率が高くなる
 - デッドロックの検出の必要がある
- ページロック
 - テーブルロックと行ロックの中間

デッドロック

- 複数人が互いにロックすることにより先に進めなくなる
 - データベース以外でも並行処理では同じことが起こる



- デッドロックの検出
 - 依存関係を調べてデッドロックしていることを見つける
 - ロールバックさせて, 再度実行する
- デッドロックしないためには
 - 同じ順でテーブル(資源)をロックする

読み出しロックと書き込みロック

- 読み出しロック
 - テーブルの中身を読む
 - 複数人が同時に読んでも問題ない
 - 書き込みを禁止する
- 書き込みロック
 - テーブルに変更を行なう
 - 他の読み・書きを禁止する
 - 書き込み以前の状態を読み出し可能とする場合もある
- 読み出しロック中のテーブルの最後への追加
 - 読み出しと同時に実行できる場合がある

問い合わせ処理の最適化

- 問い合わせでは複数のテーブルを結合する
 - 結合演算は可換および結合則が成り立つ
 - $A \bowtie B = B \bowtie A$ および $(A \bowtie B) \bowtie C = A \bowtie (B \bowtie C)$
 - テーブルの結合 (join) 順を選ぶ
 - 制限を適用する順を選ぶ
- テーブルのデータのアクセス方法
 - 順番に読むかランダムに読むか
 - 主キーを使ってアクセスするのか

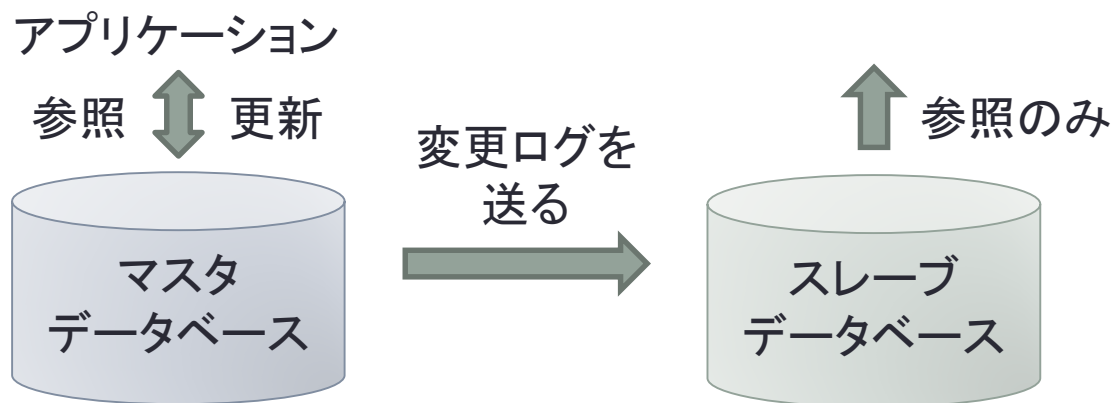
- 例: 2022年春学期に科目を履修している学生を抽出する

```
SELECT DISTINCT s.学籍番号, s.氏名 FROM 履修者 r, 学生 s, 授業 l
WHERE r.学籍番号=s.学籍番号 AND r.授業コード=l.授業コード AND l.学期='2022春'
```

- 戦略1
 - 履修者のテーブルを順番に調べ、対応する授業を探し、学期が2022年春であった場合、対応する学生を探す
- 戦略2
 - 学生のテーブルを順番に調べ、履修者の中から履修している授業を探し、その授業が2022年春であるかを調べる
- 戦略3
 - 授業のテーブルを順番に調べ、2022年春である場合に、履修者テーブルから履修者を探し、対応する学生を探す

データベースの複製

- マスタスレーブ方式の複製
 - MySQLなどがサポート
 - マスタはテーブルに対する変更をログとして記録
 - スレーブはマスタのログを読んで自分のテーブルを更新
- ログの記録方式
 - SQLステートメントを記録
 - 変更された行の内容を記録
 - 両者をミックス



ストレージエンジンの実装

- データベースは永続記憶
 - 電源を切って再起動しても前の状態から開始
 - データはファイルに保存
- ファイルの取り扱いの注意
 - fsyncあるいはfdatasyncを用いて確実に物理媒体に書き込む
 - ログを使うことにより不完全な状態をなくす
 - ログは追加書きのみ
- キーによるインデックス
 - B tree
 - ハッシュ

まとめ

- データベース
 - データベース管理システム
- 関係データベース
 - 関係演算
 - 基本演算(制限, 射影, 結合)
 - SQL
- ロック
 - デッドロック