データベース

第3回　リレーショナル代数

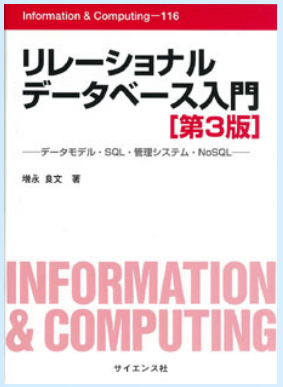
九州工業大学 情報工学部　尾下真樹

今回の内容

- 前回の復習
- リレーションの操作体系
  - リレーショナル代数とリレーショナル論理式
  - SQLとリレーション操作の関係
- リレーショナル代数
  - リレーショナル代数演算子
  - リレーショナル代数式

教科書・参考書

- 「リレーショナルデータベース入門 第3版」
  増永良文 著、サイエンス社（3,200円）
  - 3章（3.1～3.5節）



- 「データベースシステム 改訂2版」
  北川 博之 著、オーム社（3,200円）
  - 3章（3.3）

前回の復習

データモデル

- データモデル
  - データベースに格納するデータ構造（スキーマ）を記述するための枠組み
  - どのようにファイルやメモリにデータが格納されるかは気にせず、概念的なデータ構造を定義
  - 各データベースシステムはある特定のデータモデルをサポート
    - リレーショナルモデルが代表的
    - これまでにさまざまなデータモデルが開発されてきた
  - データモデルに基づいた操作言語が存在

リレーショナルデータモデル

- リレーショナルデータモデル
  - 表形式のデータ構造（リレーション）によりデータを格納するデータモデル
  - リレーション同士の演算によって、さまざまな処理を実現できる
  - 他のデータモデルと比べて、単純、データ独立性が高い、といった利点がある
    - ただし、可変長のデータや、データ構造が複雑なデータには不向き

スキーマとインスタンス

- リレーショナルデータベースの例
  - リレーション
    - 複数の属性の組み合わせによりデータを表現
  - スキーマ
    - リレーションの項目の型、属性制約、キー制約など
  - インスタンス
    - それぞれのデータ、表の各行に相当

学生

学生番号	氏名
0123001	織田 信長
0123002	豊臣 秀吉
0123003	徳川 家康

履修

科目番号	学生番号	成績
01	0123001	60
03	0123002	80
01	0123003	70

リレーションの整合性制約

- リレーションスキーマに、さまざまな制約を設定することで、整合性を保つことができる

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

属性値は各属性のドメイン制約に従う

主キー　外部キー

（超キー、候補キー）

あるリレーションの主キー

参照整合性制約

リレーションスキーマは第一正規形を満たす

キー制約の例

従業員（従業員番号, 氏名, 部門番号）

- 従業員番号は、全ての従業員に異なる番号が振られているとする
- 同じ部門には、同じ氏名の従業員は存在しないものとする

主キー {従業員番号} ※ 候補キーのどちらでも可
候補キー {従業員番号}, {氏名, 部門番号}
超キー {従業員番号}, {氏名, 部門番号},
{従業員番号, 氏名},
{従業員番号, 部門番号},
{従業員番号, 氏名, 部門番号}

参照整合性制約の例

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

主キー　外部キー

部門

部門番号	部門名
1	開発
2	営業
3	総務

主キー

この場合、従業員の部門番号は、必ず部門の部門番号（部門の主キー）に存在する必要がある
→ 参照整合性制約

リレーションの操作体系

データモデルと操作体系

- すべてのデータモデルは操作体系を持つ
  - データベースにデータを格納するだけでは意味がなく、格納されたデータに対して追加・削除・修正を行ったり、検索を行って操作結果を得たりすることが必要となるため

リレーションの操作

- データベースのデータは複数のリレーションにまたがって格納されている
- リレーションに対する何らかの操作を行なって、求めるデータを出力する必要がある

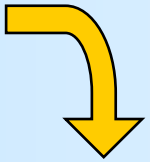
従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39

部門

部門番号	部門名
1	開発
2	営業

操作



氏名	部門名
織田 信長	開発

例：従業員「織田 信長」が所属する部門名を知りたい

リレーションの操作

- リレーションの操作
  - 主にデータの問い合わせ（検索）に利用
  - 複数のリレーションをもとに、求めるデータを新たなリレーションとして得る
- リレーションの操作体系
  - リレーショナル代数とリレーショナル論理式の２種類の操作体系がある

リレーション操作体系

- リレーショナル代数
  - 基本的な演算子を定義
  - 演算子を使った式によって、求めるリレーションが得られるようにリレーションの操作を記述する
- リレーショナル論理式
  - 一階述語理論にもとづき、求めるリレーションの条件を宣言的に記述する
  - リレーショナル論理式には、さらに2種類がある
    - タプルリレーショナル理論
    - ドメインリレーショナル理論

リレーション操作体系の比較

- リレーション操作体系
  - リレーショナル代数
  - リレーショナル論理式
- 両者の比較
  - どちらも同等の記述力を持つ（Ullmanの定理）
  - どちらもあらゆる操作が記述できるというわけではない（例:推移的閉包など、どちらの操作体系でも記述できない操作もある）
- 本講義では、リレーショナル代数を主に説明
  - リレーショナル論理式については教科書を参照

実際のリレーション操作

- 実際のリレーショナルデータベースでは、SQLという言語を使って、よりプログラミング言語に近い形で問い合わせを記述する
- 今回学ぶ代数演算は、リレーショナルモデルがどのような演算をサポートするかという理論的な操作体系
  - 単にリレーショナルデータベースを利用するだけであれば不要
  - 専門的に利用するためには、内部でどのような演算が行われるのかを理解しておく必要がある

SQLとリレーション操作の関係

利用者

SQLという言語を使い
問い合わせを記述

結果を出力

データベースシステム

リレーションを管理

利用者は、SQLの書き方を、きちんと
理解しておく必要がある

システムが内部で自動的に行ってくれる
ので、全く知らなくても使えるが、専門的
に使うのであれば、理解が必要

問い合わせが行われたら、
リレーション操作を行って、
結果を求める
（リレーショナル代数式・
リレーショナル論理式）

今回の内容

- 前回の復習
- リレーションの操作体系
  - リレーショナル代数とリレーショナル論理式
  - SQLとリレーション操作の関係
- リレーショナル代数
  - リレーショナル代数演算子
  - リレーショナル代数式

リレーショナル代数

リレーショナル代数

- リレーション同士の演算
  - リレーショナルデータベースでは、リレーション同士の演算によって複雑な検索などの操作を行う
  - リレーションに対する演算の結果もリレーションになる
  - 体系的な演算を提供している
    - 数学的に扱えるということが重要
    - 具体的なシステム（プログラム）として実現するときには、また別の工夫が必要になる

リレーショナル代数演算子

- 2つのリレーション同士の演算（2項演算）
  - 和集合、差集合、共通集合、直積
- 単一リレーションに対しての演算（単項演算）
  - 射影、選択
- 結合演算（2項演算）
  - 結合、自然結合
- その他
  - 商、改名演算

和集合

- 和集合（union）

$$R \cup S = \{ t \mid t \in R \vee t \in S \}$$

  - 2つのリレーションのインスタンスの集合を足し合わせたもの
  - 2つのリレーションスキーマは同一でなければならない（和両立条件）
    - 次数が同じで、それぞれの属性のドメインも同じ
  - 通常、同一のリレーションスキーマが複数あることはなく、複数の演算結果に対して使われる

和集合演算の例

従業員１

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

従業員２

従業員番号	部門番号	氏名	年齢
002	2	豊臣 秀吉	45
005	2	伊達 政宗	15

従業員１＋従業員２ （和演算）

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60
005	2	伊達 政宗	15

差集合

- 差集合（difference）

$$R - S = \left\{ t \mid t \in R \wedge \neg t \in S \right\}$$

  - 片方のリレーションから、もう一方のリレーションを引いたもの
  - 他は、和集合演算と同じ

  - 通常、同一のリレーションスキーマが複数あることはなく、複数の演算結果に対して使われる

和集合演算・差集合演算の例

従業員１

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

従業員２

従業員番号	部門番号	氏名	年齢
002	2	豊臣 秀吉	45
005	2	伊達 政宗	15

従業員１＋従業員２ （和集合演算）

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60
005	2	伊達 政宗	15

従業員１－従業員２ （差集合演算）

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
003	3	徳川 家康	39
004	1	柴田 勝家	60

共通集合

- 共通集合（intersection）

$$R \cap S = R - (R - S)$$

  - 2つのリレーションの共通部分
  - 2つのリレーションは、和集合・差集合と同様に和両立条件を満たさなければならない
  - 共通集合は、差集合により表現可能
  - 通常、同一のリレーションスキーマが複数あることはなく、複数の演算結果に対して使われる

共通集合演算の例

従業員１

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

従業員２

従業員番号	部門番号	氏名	年齢
002	2	豊臣 秀吉	45
005	2	伊達 政宗	15

従業員１∩従業員２ （共通集合演算）

従業員番号	部門番号	氏名	年齢
002	2	豊臣 秀吉	45

直積

- 直積（cartesian product）

$$R \times S = \left\{ t * u \mid t \in R \wedge u \in S \right\}$$

（$t * u$はタプルtとuを連結したもの）

  - リレーションの各インスタンス同士を全て組み合わせて、連結したリレーションを得る
  - もとのリレーション名をつけて属性名を区別する
  - 単体ではあまり意味がなく、別の演算子（選択演算子）と組み合わせて使うことが多い
    - 結合演算子（詳しくは後で説明）

直積演算の例

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
004	1	柴田 勝家	60

×

部門

部門番号	部門名
1	開発
2	営業

＝

従業員. 従業員番号	従業員. 部門番号	従業員.氏名	従業員. 年齢	部門. 部門番号	部門. 部門名
001	1	織田 信長	48	1	開発
002	2	豊臣 秀吉	45	1	開発
004	1	柴田 勝家	60	1	開発
001	1	織田 信長	48	2	営業
002	2	豊臣 秀吉	45	2	営業
004	1	柴田 勝家	60	2	営業

リレーショナル代数演算子

- 2つのリレーション同士の演算（2項演算）
  - 和集合、差集合、共通集合、直積
- 単一リレーションに対しての演算（単項演算）
  - 射影、選択
- 結合演算（2項演算）
  - 結合、自然結合
- その他
  - 商、改名演算

注意：演算子の表記について

- 教科書（参考書）とは表記が異なるので注意
  - 「リレーショナルデータベース入門 」と「データベースシステム」で表記が異なる
    - 前者では角括弧 [ ] を使った表記、後者ではギリシャ文字を使った表記、が使われている
  - 本講義では、演算子の違いが分かりやすいよう、ギリシャ文字を使った表記に合わせている
    - 試験もこちらの表記を用いること
      - 後者の参考書を買っていなくとも、本資料の説明を理解すれば問題ない
      - 演算子の定義はどちらの参考書も同じ

射影

- 射影（projection）

（パイ）

$$\pi_{A'_1,\cdots,A'_m}(R)=\{t[A'_1,\cdots,A'_m]\mid t\in R\}$$

$$\{A'_1,\cdots,A'_m\}\subseteq\{A'_1,\cdots,A'_n\},\ m\le n$$

  - 指定した属性だけを残して、他の属性は削除
    - テーブルから必要な列（属性）のみを取り出す
  - 出力に必要ない属性や、直積で重複した属性を取り除くためなどに使用

射影演算の例

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

$\pi_{\text{氏名, 年齢}}$（従業員）

氏名	年齢
織田 信長	48
豊臣 秀吉	45
徳川 家康	39
柴田 勝家	60

射影は必要な属性（列）のみを取り出す

選択

- 選択（selection）

  （シグマ）

$$\sigma_F(R) = \{ t \mid t \in R \wedge P_F(t) \}$$

  - 選択条件 F で指定した条件を満たすインスタンスのみを残して、他のインスタンスは削除
    - テーブルから必要な行（インスタンス、タプル）のみを取り出す
  - 主にデータの検索処理に使われ、使用頻度はかなり高い

選択

- 選択の条件（下記のどれかの方法で記述）
  - 属性 A_i と 定数 c の比較演算 θ による比較

    $A_i \, \theta \, c$
  - 属性 A_i と 属性 A_j の比較演算子 θ による比較

    $A_i \, \theta \, A_j$
  - 上の2つの論理和（∨）、論理積（∧）、否定（¬）を用いて組み合わせたもの
- 比較演算子
  - θ は $=, <, >, \geq, \leq, \neq$ のどれか

（シータ）

射影演算・選択演算の例

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
003	3	徳川 家康	39
004	1	柴田 勝家	60

$\pi_{\text{氏名, 年齢}}$（従業員）

氏名	年齢
織田 信長	48
豊臣 秀吉	45
徳川 家康	39
柴田 勝家	60

射影は必要な属性（列）のみを取り出す

$\sigma_{\text{年齢}>45}$（従業員）

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
004	1	柴田 勝家	60

選択は必要なインスタンス（行）のみを取り出す

リレーショナル代数演算子

- 2つのリレーション同士の演算（2項演算）
  - 和集合、差集合、共通集合、直積
- 単一リレーションに対しての演算（単項演算）
  - 射影、選択
- 結合演算（2項演算）
  - 結合、自然結合
- その他
  - 商、改名演算

結合

- 結合（join）

$$R \bowtie_F S = \sigma_F \left(R \times S \right)$$

  - 実際の応用でよく使われる重要な演算子
    - 非常に重要！
  - 直積と選択を組み合わせた演算子
- 結合の種類
  - 等結合（比較演算子が = の場合、よく使われる）
  - θ-結合（= 以外の比較演算子の場合）

等結合の例

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
004	1	柴田 勝家	60

⋈ 部門番号＝部門番号

部門

部門番号	部門名
1	開発
2	営業

＝

従業員.従業員番号	従業員.部門番号	従業員.氏名	従業員.年齢	部門.部門番号	部門.部門名
001	1	織田 信長	48	1	開発
004	1	柴田 勝家	60	1	開発
002	2	豊臣 秀吉	45	2	営業

この例のように複数のリレーションの情報を組み合わせるために、結合が使われる

結合は直積と選択の組み合わせで計算される（次スライド）

等結合の例(1.直積演算)

従業員

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
004	1	柴田 勝家	60

×

部門

部門番号	部門名
1	開発
2	営業

＝

従業員. 従業員番号	従業員. 部門番号	従業員.氏名	従業員. 年齢	部門. 部門番号	部門. 部門名
001	1	織田 信長	48	1	開発
002	2	豊臣 秀吉	45	1	開発
004	1	柴田 勝家	60	1	開発
001	1	織田 信長	48	2	営業
002	2	豊臣 秀吉	45	2	営業
004	1	柴田 勝家	60	2	営業

等結合の例(2.選択演算)

σ 従業員.部門番号
＝部門.部門番号

従業員.従業員番号	従業員.部門番号	従業員.氏名	従業員.年齢	部門.部門番号	部門.部門名
001	1	織田 信長	48	1	開発
002	2	豊臣 秀吉	45	1	開発
004	1	柴田 勝家	60	1	開発
001	1	織田 信長	48	2	営業
002	2	豊臣 秀吉	45	2	営業
004	1	柴田 勝家	60	2	営業

この例のように複数のリレーションの情報を組み合わせるために、結合が使われる

＝

従業員.従業員番号	従業員.部門番号	従業員.氏名	従業員.年齢	部門.部門番号	部門.部門名
001	1	織田 信長	48	1	開発
004	1	柴田 勝家	60	1	開発
002	2	豊臣 秀吉	45	2	営業

自然結合

- 自然結合（natural join）

$$R \bowtie S = \pi_{A_1,\cdots A_n,B_1,\cdots B_m,C_1,\cdots C_k}\left(\sigma_{R.B_1=S.B_1\wedge\cdots\wedge R.B_m=S.B_m}\left(R\times S\right)\right)$$

$$R\left(A_1,\cdots A_n,B_1,\cdots B_m\right),\ S\left(B_1,\cdots B_m,C_1,\cdots C_k\right)$$

  - これも実際の応用でよく使われる重要な演算子
  - 2つのリレーションを同一の属性同士で等結合し、結合結果から同一の属性を取り除いたもの
    - 通常の等結合の結果には、同一の属性値を持つ属性が重複して存在することになり無駄

自然結合の例

従業員番号	部門番号	氏名	年齢
001	1	織田 信長	48
002	2	豊臣 秀吉	45
004	1	柴田 勝家	60

⋈

部門番号	部門名
1	開発
2	営業

部門番号が同じであるインスタンス同士の組み合わせになる

＝

従業員番号	部門番号	氏名	年齢	部門名
001	1	織田 信長	48	開発
002	2	豊臣 秀吉	45	営業
004	1	柴田 勝家	60	開発

※ 各属性のもとのリレーション名は省略できる

リレーショナル代数演算子

- 2つのリレーション同士の演算（2項演算）
  - 和集合、差集合、共通集合、直積
- 単一リレーションに対しての演算（単項演算）
  - 射影、選択
- 結合演算（2項演算）
  - 結合、自然結合
- その他
  - 商、改名演算子

商

- 商（division, quotient）

$$R \div S = \left\{ t \mid t \in R\left(A_1, \cdots A_m\right) \wedge \left(\forall u \in S, \left(t, u\right) \in R\right) \right\}$$

  - ただし、リレーション R, S について、S の属性は、R の属性の部分集合

$$R\left(A_1, \cdots A_n\right), S\left(A_m, \cdots A_n\right)$$

  - RからSを除いた属性について、全てのSとの組み合わせがRに存在するものの集合を求める
  - $\left(R \times S\right) \div S = R$ が成り立つ
  - 特殊な演算なので、あまり使われない

商の例

参加者

プロジェクト番号	従業員番号
P1	001
P1	002
P1	003
P2	002
P2	003
P2	004
P3	002
P3	003

÷

プロジェクト

プロジェクト番号
P1
P2
P3

＝

従業員番号
002
003

全てのプロジェクト番号との組が存在する、従業員番号が求まる
（全プロジェクトに参加している従業員が求まる）

改名演算

- 改名演算

  （デルタ）

  $$\delta_{A \leftarrow B}(R)$$

  - 自然結合や商を適用するために、名前を変える演算
    - 属性A の名前を、属性 B に変更する
  - 自然結合や商では、属性の名前が一致することが条件になるので、属性の名前が異なっている場合に前もって名前を変更する
  - 使用例は後で紹介

リレーショナル代数演算子のまとめ

- 2つのリレーション同士の演算（2項演算）
  - 和集合、差集合、共通集合、直積
- 単一リレーションに対しての演算（単項演算）
  - 射影、選択
- 結合演算（2項演算）
  - 結合、自然結合
- その他
  - 商、改名演算

リレーショナル代数演算子の主な用途

- 選択、射影
  - 必要なデータ（表の行）や属性（表の列）を取り出すために使用
- 結合
  - 複数のリレーションを組み合わせるために使用
- 和集合、差集合、共通集合
  - 複数の演算結果同士を組み合わせるときに使用
- 直積
  - 直接は使用しない（結合を定義する上で重要）
- 商（あまり使わない）

リレーショナル代数式

- リレーショナル代数式
  - これまでに出てきたリレーショナル代数演算子を使って、問い合わせを記述したもの
  - 各種演算を組み合わせることでいろいろな問い合わせを記述できる
  - 「リレーショナル代数表現」とも呼ばれる

リレーションの例

学生（学生番号、氏名）

科目（科目番号, 科目名, 単位数）

履修（科目番号, 学生番号, 成績）

学生

学生番号	氏名
0123001	織田 信長
0123002	豊臣 秀吉
0123003	徳川 家康
…	…

履修

科目番号	学生番号	成績
001	00001	90
001	00002	80
002	00001	90
002	00003	70
…	…	…

科目

科目番号	科目名	単位
001	データベース	2
002	プログラミング	3
…	…	…

リレーショナル代数式の例1

- 科目番号002の履修者の学生番号と成績の一覧

※ 演算子の順番（1. 選択、2. 射影）に注意

$$\pi_{\text{学生番号,成績}}\left(\sigma_{\text{科目番号}=\text{‘002’}}\ \text{履修}\right)$$

2. 射影（π）により、学生番号と成績の属性のみを取り出す

履修

科目番号	学生番号	成績
001	00001	90
001	00002	80
002	00001	90
002	00003	70

学生番号	成績
00001	90
00003	70

1. 選択（σ）により、科目番号が ‘002’ の履修のデータを取り出す

リレーショナル代数式の例2

- 学生番号00001の学生が履修した科目名と成績の一覧

$$\pi_{\text{科目名,成績}}(\text{科目} \bowtie (\sigma_{\text{学生番号}=\text{‘00001’}} \text{履修}))$$

履修

科目番号	学生番号	成績
001	00001	90
001	00002	80
002	00001	90
002	00003	70

$\bowtie$

科目

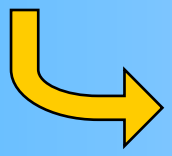
科目番号	科目名	単位
001	データベース	2
002	プログラミング	3

リレーショナル代数式の例2

- 学生番号00001の学生が履修した科目名と成績の一覧

$$\pi_{科目名,成績}(科目 \bowtie (\sigma_{学生番号='00001'} 履修))$$

科目番号	学生番号	成績	科目名	単位
001	00001	90	データベース	2
002	00001	90	プログラミング	3



科目名	成績
データベース	90
プログラミング	90

リレーショナル代数式の例2

- 学生番号00001の学生が履修した科目名と成績の一覧

$$\pi_{\text{科目名,成績}}(\text{科目} \bowtie (\sigma_{\text{学生番号='00001'}} \text{履修}))$$

- 別解
  - こちらでも良いが、通常は上の書き方をする
    - 選択は科目には関係ないため結合の前に適用

$$\pi_{\text{科目名,成績}}(\sigma_{\text{学生番号='00001'}}(\text{科目} \bowtie \text{履修}))$$

リレーショナル代数式のポイント

- ポイントを押さえれば、自分が出力したいリレーションを代数式で書くことは簡単！
  - 選択、射影、結合の組み合わせ
  - 最終的に得たい属性を射影で取得
  - 最終的に得たい属性を持つリレーションは必ず入力として必要
  - 複数のリレーションは結合を使って組み合わせる
  - 条件を満たすインスタンス（行）のみを選択で取り出す

リレーショナル代数式のポイント

- 例:学生番号00001の学生が履修した科目名と成績の一覧
  - 科目名、成績の属性が必要 → 射影で出力
  - 科目名、成績の属性が必要 → 科目、履修の２つのリレーションが必要 → 両者を結合
  - 条件「学生番号00001の学生が履修」→ 履修のリレーションからこの条件にもとづいて選択

$$\pi_{\text{科目名,成績}}\left(\text{科目} \bowtie \left(\sigma_{\text{学生番号}=\text{‘0001’}} \text{履修}\right)\right)$$

演習問題

- 終了期限までに受験する
- 終了時刻前・制限時間内に解答を提出する
  - 解答が途中でも、時間切れになると打ち切られるので、注意する
- 解答後に正答や解説が表示されるので、確認して、間違えた問題を復習すること
- ログイン不能や操作ミス等による回答失敗には、一切対応しない

今回の内容

- 前回の復習
- リレーションの操作体系
  - リレーショナル代数とリレーショナル論理式
  - SQLとリレーション操作の関係
- リレーショナル代数
  - リレーショナル代数演算子
  - リレーショナル代数式

リレーショナル代数式の例3

- 科目番号001の科目について、学生番号00001の学生よりも成績が良かった学生の学生番号の一覧
  - 2つの問い合わせの組み合わせ
  - 改名演算子が必要になる

リレーショナル代数式の例3

- 科目番号001の科目について、学生番号00001の学生よりも成績が良かった学生の学生番号の一覧
  1. 学生番号00001、科目番号001の成績 → 履修’
  2. 科目番号001、履修‘の成績よりも成績が大きい

履修

科目番号	学生番号	成績
001	00001	80
001	00002	70
002	00001	80
001	00003	90

履修‘

科目番号	学生番号	成績
001	00001	80

（σ科目番号=001∧
学生番号=00001 履修））

リレーショナル代数式の例3

- 1. 学生番号00001、科目番号001の成績

（σ科目番号=001∧学生番号=00001履修）

履修

科目番号	学生番号	成績
001	00001	80
001	00002	70
002	00001	80
001	00003	90

履修‘

科目番号	学生番号	成績
001	00001	80

リレーショナル代数式の例3

- 2. 科目番号001、履修‘の成績よりも成績が大きい
  - データ同士の比較なので選択ではなく結合を使う

$$(\sigma_{\text{科目番号}=001}\ \text{履修}) \bowtie_{\text{成績}>\text{成績}} \text{履修'})$$

履修

	科目番号	学生番号	成績
×	001	00001	80
×	001	00002	70
	002	00001	80
○	001	00003	90

履修‘

科目番号	学生番号	成績
001	00001	80

同じリレーション同士の演算では成績>成績 などと書くと混乱するので、改名演算子を用いて区別

リレーショナル代数式の例3

- 2. 科目番号001、履修‘の成績よりも成績が大きい

$$(\sigma_{\text{科目番号=001}}\ \text{履修} \bowtie_{\text{成績>成績}}\ \text{履修'})$$

– 改名演算子を使って属性名を読み替える

$$(\sigma_{\text{科目番号=001}}\ \text{履修} \bowtie_{\text{成績>成績'}}\ \delta_{\text{成績}\leftarrow\text{成績'}}\ \text{履修'}))$$

リレーショナル代数式の例3

- 科目番号001の科目について、学生番号00001の学生よりも成績が良かった学生の学生番号の一覧
  1. 学生番号00001、科目番号001の成績
  2. 科目番号001、履修‘の成績よりも成績が大きい
  – 改名演算子が必要になる

$$\pi_{\text{学生番号}}((\sigma_{\text{科目番号}=001}\ \text{履修}) \bowtie_{\text{成績}>\text{成績'}} \delta_{\text{成績}\leftarrow\text{成績'}}(\sigma_{\text{科目番号}=001 \wedge \text{学生番号}=00001}\ \text{履修}))$$

リレーショナル代数式の例4

- 履修者が一人もいない科目の科目名の一覧
  - 履修者が1人でもいる科目を取り出す
    - 科目と履修の自然結合
    - リレーションは集合なので、同じ科目が複数あっても、射影の時点で重複は取り除かれる
  - 全体の科目から引く
    - 履修者が一人もいない科目が得られる

$$(\pi_{\text{科目名}}\ \text{科目}) - (\pi_{\text{科目名}}\ (\text{科目} \bowtie \text{履修}))$$

まとめ

- リレーションの操作体系
  - リレーショナル代数とリレーショナル論理式
  - SQLとリレーション操作の関係
- リレーショナル代数
  - リレーショナル代数演算子
  - リレーショナル代数式

SQLとリレーション操作の関係

利用者

SQLという言語を使い
問い合わせを記述

結果を出力

データベースシステム

リレーションを管理

利用者は、SQLの書き方を、きちんと
理解しておく必要がある

システムが内部で自動的に行ってくれる
ので、全く知らなくても使えるが、専門的
に使うのであれば、理解が必要

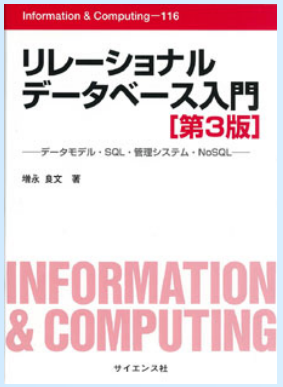
問い合わせが行われたら、
リレーション操作を行って、
結果を求める
（リレーショナル代数式・
リレーショナル論理式）

次回予告

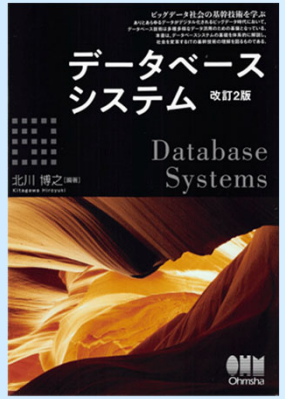
- 次回（第4回）
  - データベース言語SQL（1）
- 第5回
  - データベース実践演習
  - リレーショナルデータベースシステム（PostgreSQL）を実際に体験する
- 第6回
  - データベース言語SQL（2）
  - SQLのより詳しい書き方を学ぶ
  - 実際のデータベースシステムでSQLを体験する

教科書・参考書

- 「リレーショナルデータベース入門 第3版」
  増永良文 著、サイエンス社（3,200円）
  - 5章（5.1～5.4、5.7）



- 「データベースシステム 改訂2版」
  北川 博之 著、オーム社（3,200円）
  - 4章（4.1～4.4.3）



データベースサーバ利用申請

- データベース演習用のサーバにユーザ登録を行うため、設定されている期限までに、Moodleから申請を行うこと
  - 利用申請を行わなければ、演習が行えない
  - 学生番号と九工大IDの情報を提出する
    - 学生番号と九工大IDの違いに注意する
      - 学生番号：学生証に記載されている番号
        - » 212C0123（数字3文字＋C＋数字4文字の形式）
      - 九工大ID：LiveCampus や Moodle のユーザ名
        - » abcd1234（英字4文字＋数字4文字の形式）
    - 半角文字で、大文字と小文字を区別して、入力する

演習問題

- Moodleの演習問題を受験する
- 終了時刻前・制限時間内に解答を提出する
  - 解答が途中でも、時間切れになると打ち切られるので、注意する
- 解答後に正答や解説が表示されるので、確認して、間違えた問題を復習しておくこと
- ログイン不能や操作ミス等による回答失敗には、一切対応しない