データベース
【4：リレーショナルデータモデル（2）】

石川　佳治

リレーショナル代数

リレーショナルデータモデルの操作体系

- リレーショナルデータモデルのデータ操作を規定した体系
  - リレーショナル代数（relational algebra）
  - リレーショナル論理（relational calculus）
  - 関係代数，関係論理ともいう

代数(algebra)とは

- 演算対象の集合＋演算子(operator)の集合で作られる数学的体系
- 例
  - ブール代数:{0, 1} + 論理演算(AND, OR, NOT等)
  - 群:単位元，逆元，結合則を満たす演算で規定される代数系
  - 線形代数:対象は行列＋行列に対する演算
- リレーショナル代数
  - 対象:リレーション
  - 演算子:1ないし複数のリレーションを引数にとり，リレーションを返す
  - 演算の組合せで複雑な操作を実現

基本的な演算子：和（1）

- 5つの基本的な演算子（和，差，直積，射影，選択）が存在：他の演算子はこれらから導出可能
- 和（union）
  - 和集合をとる
  - 二つのリレーション $R(A_1, \ldots, A_n)$, $S(B_1, \ldots, B_n)$ について，$R \cup S$ は

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

  - 制約：和両立（union compatibility）
    - R と S の次数は同じ
    - R と S の属性のドメインも同じ
  - 和集合の結果のリレーションが持つ属性名は R と同一

基本的な演算子:和(2)

R

A	B	C
a	b	c
d	a	e
a	d	c

S

A	B	C
b	f	a
d	a	e

$R \cup S$

A	B	C
a	b	c
d	a	e
a	d	c
b	f	a

基本的な演算子：和(3)

R

A	B	C
a	b	c
d	a	e
a	d	c

S

D	E	F
b	f	a
d	a	e

$R \cup S$

A	B	C
a	b	c
d	a	e
a	d	c
b	f	a

属性名の並びが異なっている場合，R の方を優先

基本的な演算子：差(1)

- 差(difference)
  - 差集合をとる

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

  - 制約：和両立
  - 差集合の結果のリレーションが持つ属性名は R と同一

基本的な演算子：差(2)

R

A	B	C
a	b	c
d	a	e
a	d	c

S

A	B	C
b	f	a
d	a	e

$R - S$

A	B	C
a	b	c
a	d	c

基本的な演算子：直積（1）

- 直積（caretesian product）
  - 二つのリレーション $R(A_1, \ldots, A_n)$, $S(B_1, \ldots, B_m)$ について，$R \times S$ は

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

  - $t * u$ は t と u を連結したタプル
  - 演算結果のリレーションの属性名
    - R, S の属性名に同じものがなければそのまま引き継ぐ
    - 同じものがある場合 $R.A_i$, $S.B_i$ などとして区別

基本的な演算子：直積(2)

R

A	B	C
a	b	c
d	a	e
a	d	c

S

D	E	F
a	f	d
d	c	b

$R \times S$

A	B	C	D	E	F
a	b	c	a	f	d
a	b	c	d	c	b
d	a	e	a	f	d
d	a	e	d	c	b
a	d	c	a	f	d
a	d	c	d	c	b

基本的な演算子：射影（1）

- 射影（projection）
  - リレーション $R(A_1, \ldots, A_n)$ が持つ属性のうち，指定した属性だけを残し他を削除する
  - $\{A_1', \ldots, A_m'\} \subseteq \{A_1, \ldots, A_n\}$ のとき

$$\pi_{A_1',\ldots,A_m'}(R) = \{t[A_1',\ldots,A_m'] \mid t \in R\}$$

  - $t[A_1', \ldots, A_m']$ は，タプル t から $A_1', \ldots, A_m'$ の値だけを残して他を除いたタプル

基本的な演算子：射影（2）

R

A	B	C
a	b	c
d	a	e
a	d	c

$\pi_{A,\,C}(R)$

A	C
a	c
d	e

基本的な演算子：選択（1）

- 選択（selection）
  - リレーション $R(A_1, \ldots, A_n)$ が持つタプルのうち，指定した条件を満たすものだけを残す
  - 選択条件
    1. 属性 A_i の値と定数 c の比較 $A_i\ \theta\, c$： θ は $=, <, >, \leq, \geq, \neq$ のいずれか
    2. 属性 A_i, A_j の比較条件 $A_i\ \theta\, A_j$： A_i, A_j のドメインは同一
    3. 上記の条件を $\wedge, \vee, \neg$ で組み合わせたもの
  - 選択条件 F を用いた選択 $\sigma_F(R)$

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

    - $P_F(t)$ はタプル t が条件 F を満たすとき真になる述語

基本的な演算子：選択(2)

R

A	B	C
a	b	c
d	a	e
a	d	c

$\sigma_{C\,=\,c}(R)$

A	B	C
a	b	c
a	d	c

その他の演算子：結合（1）

- 有用な操作パターンに対応した演算子を定義
- 結合（join）
  - 直積と選択の組合せ
  - しばしば用いられる有用な演算
  - 結合条件 $F = A_i \ \theta \ B_j$
    - θ は $=, <, >, \leq, \geq, \neq$ のいずれか
  - 結合は以下のリレーションを導出

$$R \bowtie_F S = \{t * u \mid t \in R \wedge u \in S \wedge P_F(t, u)\}$$

- $P_F(t, u)$ は t , u が結合条件 F を満たすとき真になる述語

その他の演算子：結合（2）

- 結合演算は直積と選択の組合せで定義可能

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

- 等結合（equi-join）
  - 比較演算子 θ が = である場合
- θ 結合（ θ -join）
  - 比較演算子 θ が = 以外の場合
- $\wedge$, $\vee$, $\neg$ で複数の条件を組み合わせて結合条件としてもよい

その他の演算子：結合(3)

R

A	B
1	3
2	5
3	4

S

C	D	E
4	5	3
4	4	6
5	5	7

$R \bowtie_{B=C} S$

A	B	C	D	E
2	5	5	5	7
3	4	4	5	3
3	4	4	4	6

等結合

$R \bowtie_{B<C} S$

A	B	C	D	E
1	3	4	5	3
1	3	4	4	6
1	3	5	5	7
3	4	5	5	7

θ結合

その他の演算子：自然結合（1）

- **自然結合**（natural join）
  - 同じ属性名の属性について等結合を行う処理はしばしば発生
  - 自然結合はそのような状況に特化
  - 二つのリレーション $R(A_1, \ldots, A_n, B_1, \ldots, B_m)$, $S(B_1, \ldots, B_m, C_1, \ldots, C_k)$ について，以下のように定義

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

その他の演算子：自然結合（2）

R

A	B	C
a	b	c
b	c	d
c	d	e
d	e	f

S

B	C	D
b	c	f
d	e	a
d	e	c

$R \bowtie S$

A	B	C	D
a	b	c	f
c	d	e	a
c	d	e	c

その他の演算子：共通部分集合（1）

- 共通部分集合（intersection）
  - 二つのリレーションの交わりをとる

$$R \cap S = \{t \mid t \in R \wedge t \in S\}$$

  - 制約：R, S は和両立である
  - 差演算で表現可能

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

その他の演算子：共通部分(2)

R

A	B	C
a	b	c
d	a	e
a	d	c

S

A	B	C
b	f	a
d	a	e

$R \cap S$

A	B	C
d	a	e

その他の演算子：商（1）

プロジェクトメンバ

プロジェクト番号	従業員番号
p1	1
p1	2
p1	3
p2	2
p2	3
p3	2
p3	3

プロジェクト

プロジェクト番号
p1
p2
p3

プロジェクトメンバ
÷ プロジェクト

従業員番号
2
3

登録されているすべてのプロジェクトの
メンバとなっている従業員の番号を求めよ

その他の演算子：商(2)

- 商(division, quotient)
  - 二つのリレーション $R(A_1, \ldots, A_n)$, $S(A_m, \ldots, A_n)$ について定義
  - ただし, $1 < m \le n$ で同じ名前の属性のドメインは同一

$$R \div S = \pi_{A_1,\ldots,A_{m-1}}(R) - \pi_{A_1,\ldots,A_{m-1}}((\pi_{A_1,\ldots,A_{m-1}}(R) \times S) - R)$$

  - 以下の性質が成立

$$(R \times S) \div S = R$$

その他の演算子：商（3）

- 例の説明（$n = 2, m = 2$)
- $R(A_1, A_2)$：プロジェクトメンバ（従業員番号，プロジェクト番号）
- $S(A_2)$ ：プロジェクト（プロジェクト番号）
- 第1ステップ
  - 従業員とプロジェクトの考えうるすべての組合せを作成

$\pi_{A1}(R) \times S$
$= \pi_{従業員番号}$(プロジェクトメンバ)
$\times$ プロジェクト

従業員番号	プロジェクト番号
1	p1
1	p2
1	p3
2	p1
2	p2
2	p3
3	p1
3	p2
3	p3

その他の演算子：商(4)

- ステップ2
  - ステップ1の結果から「プロジェクト」を引く
  - 一部のプロジェクトにしか参加していない従業員の情報が得られる
- ステップ3
  - 従業員番号について射影
  - 一部のプロジェクトにしか参加していない従業員の番号を得る

$$(\pi_{A1}(R) \times S) - R$$
$$= [\text{ステップ1の結果}] - R$$

従業員番号	プロジェクト番号
1	p2
1	p3

⇩

$$\pi_{A1}((\pi_{A1}(R) \times S) - R)$$
$$= \pi_{\text{従業員番号}}([\text{ステップ2の結果}])$$

従業員番号
1

その他の演算子：商（5）

- ステップ4
  - すべての従業員番号から，ステップ3で得られた「一部のプロジェクトにしか属していない従業員の従業員番号」を引く

$$\pi_{A1}(R) - \pi_{A1}((\pi_{A1}(R) \times S) - R)$$
$$= \pi_{従業員番号}(R) - [ステップ3の結果]$$

従業員番号
2
3

その他の演算子：改名演算子

- 改名（renaming）
  - 属性名を変更する
  - 直積演算などにおける名前の衝突，自然結合・商演算における名前の一致のために利用

R

A	B	C
a	b	c
d	a	e

⇓

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

D	B	C
a	b	c
d	a	e

リレーショナル代数式(1)

- リレーショナル代数式（relational algebra expression）
  1. リレーション名はリレーショナル代数式
  2. リレーション定数はリレーショナル代数式
  3. リレーショナル代数式をオペランドとしたリレーショナル代数演算子の適用はリレーショナル代数式

  ※ リレーショナル定数：データベース中に存在せず，問合せ記述の目的のためだけに用いるリレーション

リレーショナル代数式(2)

- Q1：科目番号002の科目の履修者の学籍番号と成績の一覧

$$\pi_{学籍番号,\ 成績}(\sigma_{科目番号='002'}(履修))$$

履修

科目番号	学籍番号	成績
001	00001	90
001	00002	80
002	00001	90
002	00003	70

選択

学籍番号,
成績について
射影

学籍番号	成績
00001	90
00003	70

教科書 図3.3(p.34)の
データベースを想定

リレーショナル代数式(3)

- Q2:学籍番号00001の学生が履修した科目の科目番号，科目名，成績の一覧

$$\pi_{科目番号, 科目名, 成績}(科目 \bowtie (\sigma_{学籍番号='00001'}(履修))$$

科目

科目番号	科目名	単位数
001	データベース	2
002	システムプログラム	3

$\sigma_{学籍番号='00001'}$履修

科目番号	学籍番号	成績
001	00001	90
002	00001	90

$科目 \bowtie (\sigma_{学籍番号='00001'}履修)$

科目番号	科目名	単位数	学籍番号	成績
001	データベース	2	00001	90
002	システムプログラム	3	00001	90

射影

リレーショナル代数式(4)

- Q3:情報工学専攻のいずれかの学生が履修した科目の科目番号と科目名の一覧

$$\pi_{科目番号,科目名}(科目 \bowtie 履修 \bowtie (\sigma_{専攻=‘情報工学’}(学生)))$$

$科目 \bowtie 履修 \bowtie (\sigma_{専攻=‘情報工学’}(学生)))$

科目番号	科目名	単位数	成績	学籍番号	氏名	専攻	住所
001	データベース	2	90	00001	山田一郎	情報工学	東京都×××
002	システムプログラム	3	90	00001	山田一郎	情報工学	東京都×××
001	データベース	2	80	00002	鈴木明	情報工学	茨城県△△△

射影

リレーショナル代数式(5)

- Q4:科目番号001の科目に関して学籍番号00002の学生よりも成績のよかった学生の学籍番号の一覧

$$\pi_{\text{学籍番号}}(\sigma_{\text{科目番号}='001'}(\text{履修}) \bowtie_{\text{成績}>\text{成績00002}} \delta_{\text{成績}\leftarrow\text{成績00002}}(\pi_{\text{成績}}(\sigma_{\text{科目番号}='001' \wedge \text{学籍番号}='00002'}(\text{履修}))))$$

$\sigma_{\text{科目番号}='001' \wedge \text{学籍番号}='00002'}(\text{履修})$

科目番号	学籍番号	成績
001	00002	80

成績について射影
属性名変更

成績00002
80

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

科目番号	学籍番号	成績
001	00001	90
001	00002	80

成績>成績2002の条件でθ結合

科目番号	学籍番号	成績	成績0002
001	00001	90	80

射影

リレーショナル代数式(6)

- Q5:実習課題のない科目の科目番号と科目名の一覧

$$\pi_{科目番号,\,科目名}(科目) - \pi_{科目番号,\,科目名}(科目 \bowtie 実習課題)$$

科目 $\bowtie$ 実習課題

科目番号	科目名	単位数	課題番号	課題名
001	データベース	2	01	データモデリング
001	データベース	2	02	データベース設計
001	データベース	2	03	SQL
002	システムプログラム	3	01	Cプログラミング
002	システムプログラム	3	02	システムコール

科目番号, 科目名
について射影:
実習課題を持つ科目
の科目番号と科目名

科目番号	科目名
001	データベース
002	システムプログラム

これを第一項から引く
⇒この場合は空リレーション