リレーショナルデータベースの基礎（まとめ）

* リレーションスキーマ

この資料では，リレーションスキーマは「リレーション名(属性名リスト)」であるとする．例えば，リレーションTが，表1の通りであるとする．このとき，リレーションTのリレーションスキーマはT(name, score, student\_name)である．

表1 リレーションTの例

|  |  |  |
| --- | --- | --- |
| name | score | student\_name |
| Database | 80 | KK |
| Database | 95 | AA |
| Database | 80 | LL |
| Programming | 85 | KK |
| Programming | 75 | LL |

* **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では，データベースを，リレーションの集まりとして記述する．一貫性制約の記述では，主キー制約，一意制約，非空制約，ドメイン制約，参照整合性制約などの所定の制約を記述する．

* **リレーショナルデータモデルでのデータ操作の体系**

リレーショナルデータモデルでのデータ操作の体系は，リレーショナル代数演算およびリレーショナル関係論理を基礎とする．

■　**リレーショナル代数**

リレーショナル代数は，集合論に基礎を置く体系である．集合演算と，リレーショナル代数に特有の演算がある．リレーショナル代数とリレーショナル関係論理は，土台となる数理が違うものの，相互に等価であるので，リレーショナル代数のみを説明する．

　【集合演算】

　　和集合，差集合，共通集合，直積集合

　【リレーショナル代数に特有の演算】

　　射影，選択，結合，商

**■ 和両立**

リレーション*R*(*A*1, *A*2, …, *An*) と*S*(*B*1, *B*2, …, *Bm*) が和両立であるとは，次の2条件を満足しているときを言う．

(1) *R*と*S*の次数が等しい．つまり *n* = *m*

(2) 各*i*（1≦*i*≦n）について，*A*iと*B*iのドメインが等しい．

■　和集合演算

*R*とSを和両立なリレーションであるとする．*R*と*S*の和集合演算は「*R*∪*S*」と書く．定義は次の通りである．

*R*∪*S* = { *t* | *t* ∈ *R* ∨ *t* ∈ *S* }

■　差集合演算

*R*とSを和両立なリレーションであるとする．*R*と*S*の差集合演算は「*R*－*S*」と書く．定義は次の通りである．

*R*－*S* = { *t* | *t* ∈ *R* ∧ ￢( *t* ∈ *S* ) }

■　共通集合演算

*R*とSを和両立なリレーションとする．*R*と*S*の共通集合演算は「*R*∩*S*」と書く．定義は次の通りである．

*R*∩*S* = { *t* | *t* ∈ *R* ∧ *t* ∈ *S* }

■　直積集合演算

*R*(*A*1, *A*2, …, *A*n) と*S*(*B*1, *B*2, …, *B*m)をリレーションとする，*R*と*S*の直積集合演算は「*R*×*S*」と書く．定義は次の通りである．

*R*×*S* = { (*r*, *s*) | *r* ∈ *R* ∧ *s* ∈ *S* }

但し，「(*r*, *s*)」と書いているのは，*r* = (*a*1, *a*2, …, *a*n), *s* = (*b*1, *b*2, …, *b*m) とするときに，(*r*, *s*) = (*a*1, *a*2, …, *a*n, *b*1, *b*2, …, *b*m) なる *n* + *m* 項のタップルである．

■　射影演算

*R*(*A*1, *A*2, …, *A*n) をリレーションとする．全属性集合 {*A*1, *A*2, …, *A*n}の部分集合 *X* = {*A*i1, *A*i2, …, *A*ik}について，*R*の*X*上の射影は，「*R*[*X*]」あるいは「*R*[*A*i1, *A*i2, …, *A*ik]」と書く．定義は次の通りである．

*R*[*A*i1, *A*i2, …, *A*ik] = { *u* | *u* ∈ *dom*(*A*i1)×*dom*(*A*i2)×…×*dom*(*A*ik)

∧ (∃*t*∈*R*) (*t*[*A*i1] = *u*[*A*i1]∧*t*[*A*i2] = *u*[*A*i2]∧ … ∧*t*[*A*ik] = *u*[*A*ik])}

　※「*R*[*A*i1, *A*i2, …, *A*ik]」の代わりに「π*A*i1, *A*i2, …, *A*ik(*R*)」のように書くこともある．

表2(a)のリレーションTに対する選択T[score] は，表2(b) に示すようなリレーションになる．

表2 射影の例

(a) リレーションTの例（表1の再掲）

|  |  |  |
| --- | --- | --- |
| name | score | student\_name |
| Database | 80 | KK |
| Database | 95 | AA |
| Database | 80 | LL |
| Programming | 85 | KK |
| Programming | 75 | LL |

(b) (a)のリレーションTに対する射影T[score]

|  |
| --- |
| score |
| 80 |
| 95 |
| 80 |
| 85 |
| 75 |

■　選択演算

*R*(*A*1, *A*2, …, *A*n) をリレーションとする．*R*の属性*A*iと*A*j上の -選択演算を「*R*[*A*i* A*j]」と書く．定義は次の通りである．** は2項述語である．

*R*[*A*i *A*j] = { *t* | *t* ∈ *R* ∧ *t*[*A*i] ** *t*[*A*j] }

*R*の属性*A*iと値*c*に関する -選択演算を「*R*[*A*i* c*]」と書く．定義は次の通りである．

*R*[*A*i *c*] = { *t* | *t* ∈ *R* ∧ *t*[*A*i] ** *c* }

表2-Y(a)のリレーションTに対する選択T[score] は，表2-Y(b) に示すようなリレーションになる．

表3選択の例

表1のリレーションTに対する選択T[score > 85]

|  |  |  |
| --- | --- | --- |
| name | score | student\_name |
| Database | 95 | AA |
| Programming | 85 | KK |

■　リレーショナル代数での比較演算子

比較演算子は2項述語である．リレーショナル代数では，数値及び文字列に関する比較演算子**として，** ∈ **＞，≧，＝，≦，＜，≠} の6種類を考えるのが普通である．

※ SQL では「<=」, 「>=」, 「<>」のように書く．SQL と混同しないこと．

■　θ-結合演算

*R*(*A*1, *A*2, …, *A*n) と*S*(*B*1, *B*2, …, *B*m)をリレーションとする．*R*の属性*A*iと*S*の属性*B*j上のθ-結合演算を「*R*[*A*i *B*j]*S*」と書く．定義は次の通りである． は比較演算子であり，2項演算であることに注意せよ．

*R*[*A*iθ*B*j]**S** は，*R* と *S* の直積集合 (*R*×*S*) の中から「*R*.*A*iθ*S*.*B*j」を満足する要素を選ん

だもの

θ-結合演算は，直積演算と，-選択演算を使って，次のように定義することもできる．

*R*[*A*i *B*j]*S* = (*R*×*S*) [*R*.*A*i *S.B*j]

リレーショナル代数式を使って次のように書くことができる．

*R*[*A*i*B*j]*S* = { (*t*, *u*) | *t* ∈ *R* ∧ *u* ∈ *S* ∧ *t*[*A*i] *u*[*B*j] }

但し，「(*t*, *u*)」と書いているのは，*t* = (*a*1, *a*2, …, *a*n), *u* = (*b*1, *b*2, …, *b*m) とするときに，(*t*, *u*) = (*a*1, *a*2, …, *a*n, *b*1, *b*2, …, *b*m) なる *n* + *m* 項のタップルである．

※「*R*[*A*i *B*j]*S*」の代わりに「*R A*i *B*j*S*」のような書き方をすることもある．

※　ここでは，*R*の単一の属性*A*iと*S*の単一の属性*B*j の結合演算の説明にとどめる．複数の属性での結合演算については，説明を割愛する．

■　結合属性

*R*(*A*1, *A*2, …, *A*n) と*S*(*B*1, *B*2, …, *B*m)をリレーションとする．この2つのリレーションの共通属性*C*1, *C*2, …, *C*k は次の性質を満たす．

(1) {*A*1, *A*2, …, *A*n}∩{*B*1, *B*2, …, *B*m} = {*C*1, *C*2, …, *C*k}

(2) 各*Ci*（1≦*i*≦*k*）について，元のリレーション*R*と*S*におけるドメインが等しい．つまり

dom(*R*.*C*i) = dom(*S*.*C*i)

共通属性のことを結合属性ともいう．

■　自然結合演算

*R*(*A*1, *A*2, …, *A*n) と*S*(*B*1, *B*2, …, *B*m)をリレーションとする．***R*と*S*の自然結合演算を「*R* \* *S*」と書く**．自然結合演算は，結合演算と，射影演算を使って，次のように表すことができる．

*R* \* *S* = (*R*[*C*1 = *C*1, *C*2 = *C*2,…,*C*k = *C*k]*S*) [*A*1, *A*2, …,*A*n, *D*1, *D*2, …,*D*m-k],

但し，*C*1, *C*2, …, *C*k は，*A*と*B* の結合属性．

また，{*B*1, *B*2, …, *B*m}－{*C*1, *C*2, …, *C*k} = {*D*1, *D*2, …,*D*m-k}とする．

※「*R* \* *S*」の代わりに「*R S*」のように書くこともある．

■　商演算

*R*(*A*1, *A*2, …, *A*n-m,*B*1, *B*2, …, *B*m) を*n*次のリレーション，*S*(*B*1, *B*2, …, *B*m)を*m*次のリレーションとする（但し, m < n）．*R*を*Sで*割った商を「*R*÷*S*」と書く．定義は次の通りである．

*R*÷*S* = { *t* | *t* ∈*R*[*A*1, *A*2, …, *A*n-m] ∧ (∀*u*∈*S*) ((*t, u)* ∈*R* )}

但し，「(*t*, *u*)」と書いているのは，*t* = (*a*1, *a*2, …, *a*n-m), *u* = (*b*1, *b*2, …, *b*m) とするときに，(*t*, *u*) = (*a*1, *a*2, …, *a*n, *b*1, *b*2, …, *b*m) なる *n* 項のタップルである．「∀」は，全称作用素であり，全て（for all）を表す．

「*R*[*A*1, *A*2, …, *A*n-m]」と書いているのは射影演算であり，「{ *u* | *u* ∈ *dom*(*A*1)×*dom*(*A*2)×…×*dom*(*A*n-m) ∧ (∃*t*∈*R*) (*t*[*A*1] = *u*[*A*1]∧*t*[*A*2] = *u*[*A*2]∧ … ∧*t*[*A*n-m] = *u*[*A*n-m])}」のことである．

表4. リレーショナル代数のまとめ

|  |  |  |
| --- | --- | --- |
|  | 記法 | 定義  ※　定義では論理式などを使用．論理式の記法は，集合を基礎とする論理式にしている． |
| 和集合演算 | *R*∪*S* | *R*∪*S* = { *t* | *t* ∈ *R* ∨ *t* ∈ *S* } |
| 差集合演算 | *R*－*S* | *R*－*S* = { *t* | *t* ∈ *R* ∧ ￢( *t* ∈ *S* ) } |
| 共通集合演算 | *R*∩*S* | *R*∩*S* = { *t* | *t* ∈ *R* ∧ *t* ∈ *S* } |
| 直積集合演算 | *R*×*S* | *R*×*S* = { (*r*, *s*) | *r* ∈ *R* ∧ *s* ∈ *S* }  ※ (*r*, *s*) は，*r* = (*a*1, *a*2, …, *a*n), *s* = (*b*1, *b*2, …, *b*m) とするときに，(*r*, *s*) = (*a*1, *a*2, …, *a*n, *b*1, *b*2, …, *b*m) なる *n* + *m* 項のタップル |
| 射影演算 | *R*[*A*i1, *A*i2, …, *A*ik] | *R*[*A*i1, *A*i2, …, *A*ik] = { *u* | *u* ∈ *dom*(*A*i1)×*dom*(*A*i2)×…×*dom*(*A*ik)  ∧ (∃*t*∈*R*) (*t*[*A*i1] = *u*[*A*i1]∧*t*[*A*i2] = *u*[*A*i2]∧ … ∧*t*[*A*ik] = *u*[*A*ik])} |
| 選択演算 | *R*[*A*i *A*j] | *R*[*A*i *A*j] = { *t* | *t* ∈ *R* ∧ *t*[*A*i]  *t*[*A*j] } |
| *R*[*A*i *c*] | *R*[*A*i *c*] = { *t* | *t* ∈ *R* ∧ *t*[*A*i] *c* } |
| 結合演算 | 一般の結合演算 *R*[*A*i*B*j]*S* | *R*[*A*i *B*j]*S* = { (*t*, *u*) | *t* ∈ *R* ∧ *u* ∈ *S* ∧*t*[*A*i]  *u*[*B*j] }  ※ (*t*, *u*)は，*t* = (*a*1, *a*2, …, *a*n), *u* = (*b*1, *b*2, …, *b*m) とするときに，(*t*, *u*) = (*a*1, *a*2, …, *a*n, *b*1, *b*2, …, *b*m) なる *n* + *m* 項のタップル |
| 自然結合  *R* \* *S* | *R* \* *S* = (*R*[*C*1 = *C*1, *C*2 = *C*2,…,*C*k = *C*k]*S*) [*A*1, *A*2, …,*A*n, *D*1, *D*2, …,*D*m-k],  ※ *C*1, *C*2, …, *C*k は，*A*と*B* の結合属性,  {*B*1, *B*2, …, *B*m}－{*C*1, *C*2, …, *C*k} = {*D*1, *D*2, …,*D*m-k} |
| 商演算 | *R*÷*S* | *R*÷*S* = { *t* | *t* ∈*R*[*A*1, *A*2, …, *A*n-m] ∧ (∀*u*∈*S*) ((*t, u)* ∈*R* )} |

上記の8つの演算は必ずしも独立ではない．例えば，共通集合は，差集合を使って*R*∩*S = R*－(*R*－*S*) のようになる．結合は，直積と選択を表すことができる．商は，直積と射影と差を使って表すことができる．

■　リレーションの分解 (decomposition)

あるリレーションが，リレーションスキーマ *R*(*A*1, *A*2, … , *A*n) のインスタンスであるとする．このリレーションの，m 個のリレーション *X*1, *X*2, …, *X*m への分解は次のように定義される．

*X*1, *X*2, …, *X*m を*R*の全属性集合{*A*1, *A*2,…, *A*n}の部分集合で，

*X*1∪*X*2∪…∪*X*m = {*A*1, *A*2,…, *A*n}

なる条件を満たすとき（*X*iと*X*jは共通集合をもってよい），このリレーションを*m*個の射影*R*[*X*1], *R*[*X*2], …, *R*[*X*m]で置き換えることを，このリレーションの分解という．各*R*[*X*i]を*R*の分解成分という．

■　関数従属性

リレーションスキーマ***R***(*A*1, *A*2, …, *A*l, *B*1, *B*2, …, *B*m, *C*1, *C*2, …, *C*n) に関数従属性*A*1, *A*2, …, *A*l → *B*1, *B*2, …, *B*m が存在するとは，次の場合をいう．

リレーションスキーマ**Ｒ**の任意のインスタンスの任意の2タップルt と t' について，それらの*A*1, *A*2, …, *A*l値が等しければ，*B*1, *B*2, …, *B*m 値が等しいということが必ず成り立つ．

* 属性名の記法

複数のリレーションで属性名が同じことがありえるため，属性名Aと書いただけでは，どのリレーションであるかのどの属性であるかを特定できない場合がある．特定のリレーションTの特定の属性名Aを特定したい場合，リレーション名Tと属性名Aを連結してT.Aのように書く場合がある．

* **べき集合の例**

集合{みかん，りんご，バナナ}から構成されるべき集合は，{φ, {みかん}, {りんご}, {バナナ}, {みかん, りんご}, {りんご, バナナ}, {バナナ, みかん}, {みかん, りんご, バナナ}}である．

**■　第一正規形**

ドメインとして，分解不可能な「単純値」のみを対象とすること．

■ SQL 列制約 (column-constraint)

SQL列制約の代表的なものを下に列挙する．

PRIMARY KEY ・・・ 主キー制約

NOT NULL ・・・ 非空制約

UNIQUE ・・・ 一意制約

CHECK (<expression>) ・・・ 更新時にチェックされる式 (expression)

FOREIGN KEY (<column-name>) REFERENCES <foreign-table> (<column-name>, ...) ・・・ 参照整合性制約（つまり外部キー）