Primitivas de Transformacion

Propuesta de procesamiento de instancias utilizando SQL.

Introduccion

- Es una modificacion a la propuesta de Adriana de forma de exponer que las primitivas son expresables con SQL.
- Se redujo el planteo inicial:
  - No considerando las expresiones en Algebra Relacional correspondientes.
  - No se ataca el tema de la defincion de los esquemas (DDL).

P1 - Identity

- Entrada:
  - *Esquema origen:* R . Rel
- Procesamiento:
  - P1 = select * from R

Es necesario tener una sentencia SQL para esto ?

P2 - Data Filter

- Entrada:
  - *Esquema origen:* $R(A_1, ..., A_n)$. Rel
  - *Conjunto de atributos a filtrar:* X . $\{A_1, ..., A_n\}$
- Pre-procesamiento:
  - $A' = \{A'_1, ..., A'_m\} = \{A_1, ..., A_n\} - X$
- Procesamiento:
  - P2 = select $A'_1, ..., A'_m$ from R

P3 - Temporalization

- Entrada:
  - *Esquema origen:* R . Rel
  - *Valor de tiempo:* t . Time
- Procesamiento:
  - P2 = select *, t from R

P4.1 - Version Digits

- Entrada:
  - *Esquema origen:* $R(A_1, ..., A_n)$. Rel / A_1 . X . $Att_k(R)$
  - *Numero de versión:* n . String.
- Procesamiento(1):
  - P4.1 = select n || $A_1, ..., A_n$ from R

P4.2 - Key Extension

- Entrada:
  - *Esquema origen:* R . Rel
- Procesamiento:
  - P1 = select * from R

P5 - Foreing Key Update

- Entrada:
  - *Esquema origen:* $R(A_1, ..., A_n)$. Rel
  - *Esquema de la clave foránea:* $T(B_1, ..., B_m)$. Rel
  - *Antigua clave foránea:* X . $\{A_1, ..., A_n\}$
  - *Nueva clave foránea:* Y . $\{B_1, ..., B_m\}$
  - *Esquema de correspondencias:* $S(C_1, ..., C_l)$. Rel, $\{C_1, ..., C_l\} = (X \cup Y)$
- Pre-procesamiento:
  - $V = \{V_1, ..., V_j\}$ / $V = Y \cup (\{A_1, ..., A_n\} - X)$
- Procesamiento:
  - P5 = select $V_1, ..., V_j$ from R, S where R.X = S.X

P6.1 - DD-Ading 1-1

- Entrada:
  - *Esquema origen:* $R(A_1, ..., A_n)$. Rel
  - *Función de calculo:* f(X) / X . $\{A_1, ..., A_n\}$
- Procesamiento:
  - P6.1 = select *, f(X) from R

P6.2 - DD-Ading N-1

- Entrada:
  - *Esquemas origen:* $R(A_1, ..., A_n)$, $R_1(A'_1, ..., A'_{n'})$, ..., $R_n(A^n_1, ..., A^n_{n'n})$. Rel
  - *Función de calculo:* f(X) / X . $(\{A_1, ..., A_n\} \cup \{A'_1, ..., A'_{n'}\} \cup ... \cup \{A^n_{1,...,} A^n_{n'n}\})$
  - *Atributos de Join:* Y . $\{A_1, ..., A_n\}$. Y' . $\{A'_1, ..., A'_{n'}\}$ . ... . Y^n . $\{A^n_1, ..., A^n_{n'n}\}$
- Procesamiento:
  - P6.2 = select $A_1, ..., A_n$, f(X) from R, R_1, R_2, ..., R_n where $R.Y = R_1.Y'$ and $R_1.Y' = R_2.Y^2$ and ... and $R_{(n-1)}.Y^{(n-1)} = R_n.Y^n$

P6.3 - DD-Ading N-N

- Entrada:
  - *Esquemas origen:* $R(A_1, ..., A_n)$, $R_1(A'_1, ..., A'_{n'})$, ..., $R_n(A^n_1, ..., A^n_{n'n})$. Rel
  - *Expresión de agregación:* e(X) / X . $(\{A'_1, ..., A'_{n'}\} \cup ... \cup \{A^n_{1,...,} A^n_{n'n}\})$
  - *Atributos de Join:* Y . $\{A_1, ..., A_n\}$. Y' . $\{A'_1, ..., A'_{n'}\}$ . ... . Y^n . $\{A^n_1, ..., A^n_{n'n}\}$
  - *Atributos de agregación:* Z . $(\{A'_1, ..., A'_{n'}\} \cup ... \cup \{A^m_{1,...,} A^m_{n'n}\})$
- Procesamiento:
  - P6.3 = select $A_1, ..., A_n$, e(X) from R, R_1, R_2, ..., R_n where $R.Y = R_1.Y'$ and $R_1.Y' = R_2.Y^2$ and ... and $R_{(n-1)}.Y^{(n-1)} = R_n.Y^n$ group by $A_1, ..., A_n$, Z

P7 - Atribute Adding

- Entrada:
  - *Esquema origen:* R . Rel
  - *Valores de los atributos a agregar:* $\{b_1, ..., b_n\}$
- Procesamiento:
  - P7 = select *, $b_1, ..., b_n$ from R

P8 - Hierarchy Roll Up

- Entrada:
  - *Esquemas origen:*
    - $R_1(A_1, ..., A_n)$. Rel_M / . A . $\{A_1, ..., A_n\}$. A . $Att_{FK}(R_1, R_2)$
    - $R_2(B_1, ..., B_{n'})$. Rel_J / A . $\{B_1, ..., B_{n'}\}$. A . $Att_K(R_2)$
  - Atributos de medida: $Z = \{Z_1, ..., Z_k\} = Att_M(R_1)$, Z . $\{A_1, ..., A_n\}$
  - Agregaciones de los atributos: $\{e_1(Z_1), ..., e_k(Z_k)\}$
  - Nivel de la jerarquía: B / B . $\{B_1, ..., B_{n'}\}$. B . $Att_D(R_2)$
  - Atributos que por su granularidad salen de R_1: X / X . $\{A_1, ..., A_n\}$. X . $(Att_D(R_1)$ U $Att_M(R_1))$

P8 - Hierarchy Roll Up (II)

  - Atributos que por su granularidad salen de R_2: Y / Y . $\{B_1, ..., B_{n'}\}$. Y . $Att_D(R_2)$
  - Indica si genera nueva jerarquía: agg_h . Bolean
- Pre-procesamiento:
  - $V = \{V_1, ..., V_m\}$ / V = ((($\{A_1, ..., A_n\}$ – A) U B) – X) – Z
  - $V' = \{V'_1, ..., V'_{m'}\}$ / V' = $\{B_1, ..., B_{n'}\}$ – Y
- Procesamiento:
  - P8 = select $V_1, ..., V_m$, $e_1(Z_1), ..., e_k(Z_k)$ from R_1, R_2 where $R_1.A = R_2.A$ group by $V_1, ..., V_m$
  - Si se indica agg_h:
    - P8b = select distinct $V'_1, ..., V'_{m'}$ from R_2

P9 - Aggregate Generation

- Entrada:
  - *Esquemas origen:* $R(A_1, ..., A_n)$. Rel_M
  - *Atributos de medida:* $Z = \{Z_1, ..., Z_k\} = Att_M(R)$, Z . $\{A_1, ..., A_n\}$
  - *Agregaciones de los atributos:* $\{e_1(Z_1), ..., e_k(Z_k)\}$
  - *Atributos que salen de R:* X / X . $\{A_1, ..., A_n\}$. X . $(Att_D(R)$ U $Att_M(R))$
- Pre-procesamiento:
  - $V = \{V_1, ..., V_m\}$ / V = $(\{A_1, ..., A_n\}$ – X) – Z
- Procesamiento:
  - P9 = select $V_1, ..., V_m$, $e_1(Z_1), ..., e_k(Z_k)$ from R group by $V_1, ..., V_m$

P10 - Data Array Creation

- Entrada:
  - *Esquema origen:* $R(A_1, ..., A_n)$
  - *Atributo de valores predefinidos:* A . $\{A_1, ..., A_n\}$
  - *Expresión agregación:* e(A)
- Pre-procesamiento:
  - $V = \{V_1, ..., V_m\}$ / V = select distinct A from R
  - $B = \{B_1, ..., B_p\} = Att_M(R)$
  - $N = \{ N_{ij}$ / N_{ij} = "V_i" || "_" || "B_j", i=1..m, j=1..p}
  - $K = \{K_1, ..., K_{n-1}\} = \{A_1, ..., A_n\} – B – \{A\}$

P10 - Data Array Creation (II)

- Procesamiento:
  - T_1 = select $K_1, ..., K_{n-1}$, $e(B_1)$ as $N_{11}, ..., e(B_p)$ as N_{p1} from R where A = V_1 group by $K_1, ..., K_{n-1}$

  ...
  - T_m = select $K_1, ..., K_{n-1}$, $e(B_1)$ as $N_{1m}, ..., e(B_p)$ as N_{pm} from R where A = V_m group by $K_1, ..., K_{n-1}$
  - P10 = select $K_1, ..., K_{n-1}, N_{11}, ..., N_{pm}$ from $T_1, ..., T_m$ where $T_1.K=T_2.K$ and...and $T_{m-1}.K=T_m.K$

P11.1 - Vertical Partition

- Entrada:
  - *Esquema origen:* R . Rel
  - *Atributos que nunca cambian:* Y . Att(R)
  - *Atributos que algunas veces cambian:* Z . Att(R), Z . Y = .
  - *Atributos que cambian muchas veces:* W . Att(R), W . Y = . . W . Z = .
- Pre-procesamiento:
  - $Y' = \{Y'_1, ..., Y'_{n'}\}$ / $Y' = Att_K(R)$. Y
  - $Z' = \{Z'_1, ..., Z'_{n'}\}$ / $Z' = Att_K(R)$. Z
  - $W' = \{W'_1, ..., W'_{n'}\}$ / $W' = Att_K(R)$. W

P11.1 - Vertical Partition (II)

- Procesamiento:
  - P11.1.1 = select $Y'_1, \ldots, Y'_{n'}$ from R
  - P11.1.2 = select $Z'_1, \ldots, Z'_{n''}$ from R
  - P11.1.3 = select $W'_1, \ldots, W'_{n'''}$ from R

P11.2 - Horizontal Partition

- Entrada:
  - *Esquema origen:* R . Rel
  - *Condición de Historizacion:* c(X) / X . Att(R)
- Procesamiento:
  - P11.2.1 = select * from R where c(X)
  - P11.2.2 = select * from R where not(c(X))

P12.1 - De-Normalized Hierarchy Generation

- Entrada:
  - *Esquemas origen:* $R_1, \ldots, R_n$. Rel
  - *Atributos de la Jerarquía:* $J = \{J_1, \ldots, J_m\}$
  - *Clave de la Jerarquía:* k . $\{J_1, \ldots, J_m\}$
- Pre-procesamiento:
  - $S' = \{S'_1, \ldots, S'_n\} = (Att(R_1) - J)$. k

  ...
  - $S^n = \{S'_1, \ldots, S'_{n}n\} = (Att(R_n) - J)$. k
  - $s_{i(i+1)} = Att(R_i)$. J . $Att(R_{i+1})$, i=1..(n-1)

P12.1 - De-Normalized Hierarchy Generation (II)

- Procesamiento:
  - P13.0 = select distinct $J_1, \ldots, J_m$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and ... and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - P13.1 = select $S'_1, \ldots, S'_{n'}$ from R_1, P13.0 where $R_1.k$ = P13.0.k

  ...
  - P13.n = select $S'_1, \ldots, S'_{n}n$ from R_n, P13.0 where $R_1.k$ = P13.0.k

P12.2 - Snowflake Hierarchy Generation

- Entrada:
  - *Esquemas origen:* $R_1, \ldots, R_n$. Rel
  - *Conjunto ordenado de atributos de la Jerarquía:* $J = \{J_1, \ldots, J_m\}$
  - *Clave de la Jerarquía:* k . $\{J_1, \ldots, J_m\}$
- Pre-procesamiento:
  - $S' = \{S'_1, \ldots, S'_n\} = (Att(R_1) - J)$. k

  ...
  - $S^n = \{S'_1, \ldots, S'_{n}n\} = (Att(R_n) - J)$. k
  - $s_{i(i+1)} = Att(R_i)$. J . $Att(R_{i+1})$, i=1..(n-1)

P12.2 - Snowflake Hierarchy Generation (II)

- Procesamiento:
  - T1 = select distinct $J_1, \ldots, J_m$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and ... and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - P13.J1 = select distinct J_1, J_2 from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and ... and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$

  ...
  - P13.J(m-1) = select distinct $J_{(m-1)}, J_m$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and ... and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - P13.1 = select $S'_1, \ldots, S'_{n'}$ from R_1, T1 where $R_1.k$ = T1.k

  ...
  - P13.n = select $S'_1, \ldots, S'_n$n from R_n, T1 where $R_1.k$ = T1.k

P12.3 - Free Decomposition – Hierarchy Generation

- Entrada:
  - *Esquemas origen:* $R_1, \ldots, R_n$. Rel
  - *Conjunto ordenado de atributos de la Jerarquía:* J = $\{J_1, \ldots, J_m\}$
  - *Descomposición de la Jerarquía:* D = $\{D_i / D_i = \{J^i_1, \ldots, J^i_q\}$. J_i i=1..p}
  - *Clave de la Jerarquía:* k . $\{J_1, \ldots, J_m\}$
- Pre-procesamiento:
  - S' = $\{S'_1, \ldots, S'_n\}$ = (Att(R_1) – J) . k
  - …
  - S^n = $\{S'_1, \ldots, S'_n n\}$ = (Att(R_n) – J) . k
  - $s_{i(i+1)}$ = Att(R_i) . J . Att(R_{i+1}), i=1..(n-1)

P12.3 - Free Decomposition – Hierarchy Generation (II)

- Procesamiento:
  - T1 = select distinct $J_1, \ldots, J_m$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and … and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - P13.J1 = select distinct $J'_1, \ldots, J'_q$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and … and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - …
  - P13.Jp = select distinct $J^p_1, \ldots, J^p_q p$ from $R_1, \ldots, R_n$ where $R_1.s_{12} = R_2.s_{12}$ and … and $R_{(n-1)}.s_{(n-1)n} = R_n.s_{(n-1)n}$
  - P13.1 = select $S'_1, \ldots, S'_n$ from R_1, T1 where $R_1.k$ = T1.k
  - …
  - P13.n = select $S'_1, \ldots, S'_n n$ from R_n, T1 where $R_1.k$ = T1.k

P13 - Minidimension Break Off

- Entrada:
  - *Esquema origen:* R . Rel
  - *Función de clave:* f
  - *Atributos de la mini dimensión:* X = $\{X_1, \ldots, X_n\}$. Att(R)
- Pre-procesamiento:
  - $\{R'_1, \ldots, R'_m\}$ = Att(R) - X
- Procesamiento:
  - T1 = select f as F_i * from R
  - P13.1 = select F_i $X_1, \ldots, X_n$ from T1
  - P13.2 = select F_i $R'_1, \ldots, R'_m$ from T1

P14 - New Dimension Crossing

- Entrada:
  - *Esquema origen:* R_1, R_2 . Rel
  - *Atributos de Join:* A, A . Att(R_1), A . Att(R_2)
  - *Atributos que se excluyen de R_1:* Y_1 . Att(R_1)
  - *Atributos que se excluyen de R_2:* Y_2 . Att(R_2)
- Pre-procesamiento:
  - Y'_1 = $\{y'_1, \ldots, y'_n\}$ = Att(R_1) – Y_1
  - Y'_2 = $\{y''_1, \ldots, y''_m\}$ = Att(R_2) – Y_2
- Procesamiento:
  - P14 = select distinct $y'_1, \ldots, y'_n, y''_1, \ldots, y''_m$ from R_1, R_2 where $R_1.A = R_2.A$

Posibles lineas de trabajo

- Definicion mas formal de la semantica de las primitivas.
- Tentativa de implementacion de carga por cada primitiva.
- Completar el trabajo sobre Algebra Relacional.
- Encarar el tema de propiedades de las primitivas.
- Encarar el tema sobre DDL
  - Lenguaje para expresar esto ?

Dudas que surgieron

- Cuando se define mal la aplicación de una primitiva y por ejemplo se pierde la clave primaria que pasa ?
  - Esto se asume como un error del diseñador y no se verifica el error ?
  - O se imponen restricciones que se verifican el definir la primitiva que llevan a que estos errores no se cometan ?. Esta alternativa tiene el aspecto importante de que impacta el diseño.
- Existe la sospecha de que unas primitivas se pueden definir en funcion de las otras

Conclusiones

- Todas las primitivas expuestas son expresables utilizando unicamente SQL, si bien es necesario cierto pre-procesamiento y los datos son parametricos en los esquemas de entrada.
- Hay algunas lineas de trabajo posible, habria que estudiar como atacarlas.