EQUIVALENCE OF RA & RC

FUNDAMENTAL THEOREM OF DATABASE THEORY

Relational Algebra & Safe Relational Calculus are equally expressive

→ For every query in safe RC, we can always find an equiv. query in RA and vice versa

→ To translate, we need to assume that we have a **hybrid data model**

- Columns have names and they are ordered

(i.e. R over A, B, C means the 1st col. is A, 2nd col. is B, 3rd col. is C)

↳ RA cares about col. names, but not abt. the order

↳ RC cares about the ordering, but not abt. the names

ALGEBRA → CALCULUS

→ Given an RA expression E, we will translate into a FOL formula φ

→ To do this, we consider a translation **environment η**

↳ An injective map from attributes to variables

↳ For an attribute A, we assume $\eta(A) = x_A$ → attribute A is mapped by means of the environment η to a variable x_A

1) Base Relations

R over $A_1, \ldots, A_n$ is translated to $R(\eta(A_1), \ldots, \eta(A_n))$

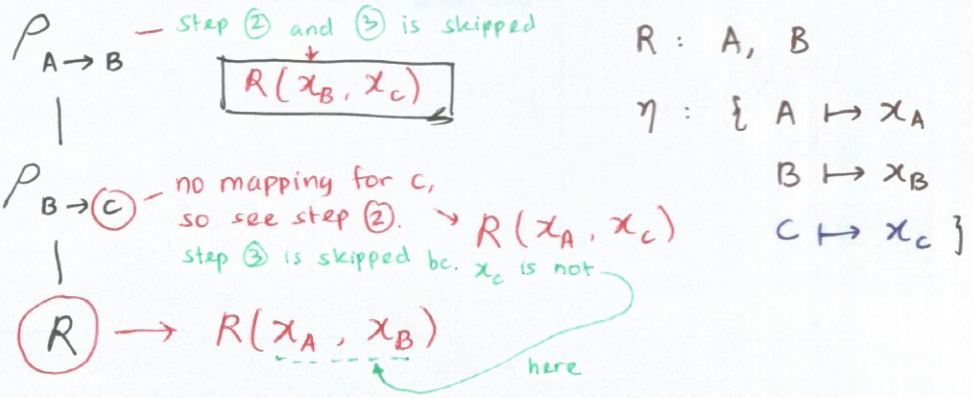
i.e. If R is a base relation over A, B and environment $\eta = \{A \mapsto x_A, B \mapsto x_B, \ldots\}$ then R is translated to $R(x_A, x_B)$

2) Renaming, $\rho_{OLD \to NEW}(E)$

1) Translate E to φ
2) If there is no mapping for NEW in η, add $\{NEW \mapsto x_{NEW}\}$
3) Replace every occurence of $\eta(NEW)$ in φ w/ a fresh variable (var. that has not been used)
4) Replace every (free) occurence of $\eta(OLD)$ in φ by $\eta(NEW)$

i.e. If R is a base relation over A, B

then $\rho_{A \to B}(\rho_{B \to C}(R))$ is translated to $R(x_B, x_C)$ → Explanation at the back



3) Projection

$\pi_a(E)$ is translated to $\exists X \varphi$

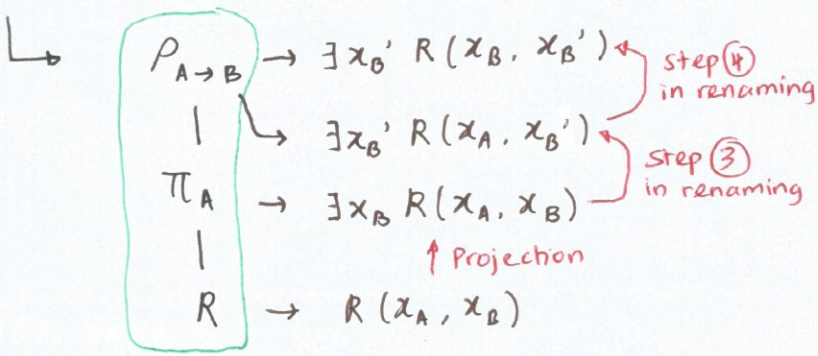
where φ is the translation of E

$[X = free(\varphi) - \eta(\alpha)]$ → attributes that are NOT projected become quantified

i.e. If R is a base relation over A, B

then $\pi_A(R)$ is translated into $\exists x_B \, R(x_A, x_B)$

i.e. $\rho_{A \to B}(\pi_A(R))$



4) Selection

$\sigma_\theta(E)$ is translated to $\varphi \wedge \eta(\theta)$

where φ is the translation of E

$\eta(\theta)$ is obtained from θ by replacing each attribute A by $\eta(A)$

i.e. If R is a base relation over A, B

then $\sigma_{A=B}(R)$ is translated to $R(x_A, x_B) \wedge (x_A = x_B)$

5) Product

$E_1 \times E_2$ is translated to $\varphi_1 \wedge \varphi_2$

6) Union

$E_1 \cup E_2$ is translated to $\varphi_1 \vee \varphi_2$

7) Difference

$E_1 - E_2$ is translated to $\varphi_1 \wedge \neg \varphi_2$

EXAMPLE

Customer : CustID, Name

Account : Number, CustID

$\eta = \{ \text{CustID} \mapsto x_1, \text{Name} \mapsto x_2, \text{Number} \mapsto x_3 \}$

Translate Customer ⋈ Account to RC.

$$= \pi_{\substack{\text{CustID, Name,} \\ \text{Number}}} \left(\sigma_{\text{CustID} = \text{CustID}'} \left(\text{Customer} \times \rho_{\text{CustID} \to \text{CustID}'} (\text{Account}) \right) \right)$$

$\pi_{\text{CustID, Name, Number}}$

$\exists x_1'$ •

$\sigma_{\text{CustID} = \text{CustID}'}$ • $\wedge\ x_1 = x_1'$

$\times$ Customer $(x_1, x_2) \wedge$ Account (x_3, x_1')

Customer

Customer (x_1, x_2)

$\rho_{\text{CustID} = \text{CustID}'}$

Account (x_3, x_1')

Account

Account (x_3, x_1)

$\therefore \exists x_1'$ Customer $(x_1, x_2) \wedge$ Account $(x_3, x_1') \wedge x_1 = x_1'$

ACTIVE DOMAIN IN RA

→ Given a schema, you can always write a RA expression tht. gives you a unary table whose contents are precisely all elements of the active domain of the database

> For relation R over attributes $A_1, \dots, A_n$,
>
> $Adom(R)$ is given by $\rho_{A_1 \to A}(\pi_{A_1}(R)) \cup \cdots \cup \rho_{A_n \to A}(\pi_{A_n}(R))$
>
> Projection over each attribute of R and renaming all these projections to the same attribute name (in order to take the union)

→ For a database D, $\boxed{Adom(D) = \bigcup_{R \in D} Adom(R)}$

For each relation in the database

→ We denote by $\boxed{Adom_N}$ the RA expression that, on a database D, returns a table...

1) w/ a single column named N
2) consisting of all elems. of $Adom(D)$

CALCULUS ⟶ ALGEBRA

→ Given a FOL formula φ, we will translate into an RA expression E.

→ The formula φ must satisfy the ff. assumptions:

- No $\forall x$, $\rightarrow$ (implication), $\neg(\neg A)$ (double negation)
- No distinct pair of quantifiers binds the same variable name
- No variable name occurs both free and bound
- No variable name is repeated within a predicate → $\{x \mid R(x,x)\} \equiv \{x \mid \exists y\ R(x,y) \wedge (x = y)\}$
- No constants in predicates
- No atoms of the form $x\ \mathbf{op}\ x$ / $c_1\ \mathbf{op}\ c_2$ (2 constants to T/F → we can reduce; can be simplified)

→ Next, we define an **environment η**

↳ An injective map from variables to attributes

↳ For a variable x, we assume $\eta(x) = A_x$

1) Predicate

→ Let R be over attributes $A_1, \ldots, A_n$,

$R(x_1, \ldots, x_n)$ is translated to $\rho_{A_1 \to \eta(x_1), \ldots, A_n \to \eta(x_n)}(R)$

i.e. For R over attributes A, B, C,

$R(x, y, z)$ is translated into $\rho_{A \to A_x,\ B \to A_y,\ C \to A_z}(R)$

$\eta : \{ x \mapsto A_x,\ y \mapsto A_y,\ z \mapsto A_z \}$

2) Existential Quantification

$\exists x\, \varphi$ is translated to $\pi_{\eta(X - \{x\})}(E)$

where E is the translation of φ

$X = \text{free}(\varphi)$

i.e. For φ w/ free variables x, y, z and translation E,

$\exists y\, \varphi$ is translated into $\pi_{A_x, A_z}(E)$

3) Comparisons

x **op** y is translated to $\sigma_{\eta(x)\ \mathbf{op}\ \eta(y)}(\text{Adom}_{\eta(x)} \times \text{Adom}_{\eta(y)})$

x **op** c is translated to $\sigma_{\eta(x)\ \mathbf{op}\ c}(\text{Adom}_{\eta(x)})$

i.e. → $x = y$ is translated to $\sigma_{A_x = A_y}(\text{Adom}_{A_x} \times \text{Adom}_{A_y})$

→ $x > 1$ is translated to $\sigma_{A_x > 1}(\text{Adom}_{A_x})$

4) Negation

translation of φ

$\neg\varphi$ is translated into $\left(\underset{x \in \text{free}(\varphi)}{\times} \text{Adom}_{\eta(x)} \right) - E$

i.e. For φ w/ free variables x, y and translation E,

$\neg\varphi$ is translated to $\text{Adom}_{A_x} \times \text{Adom}_{A_y} - E$

$\eta: \{ x \mapsto A_x, y \mapsto A_y \}$

5) Disjunction

$\varphi_1 \vee \varphi_2$ is translated to $E_1 \times \left(\underset{x \in X_2 - X_1}{\times} \text{Adom}_{\eta(x)} \right) \cup E_2 \times \left(\underset{x \in X_1 - X_2}{\times} \text{Adom}_{\eta(x)} \right)$

where for $i \in \{1,2\}$, E_i is the translation of φ_i

$X_i = \text{free}(\varphi_i)$

i.e. R: A, B

S: B, C

$\eta: \{ x \mapsto A, y \mapsto B, z \mapsto C \}$

$R(x,y) \vee S(y,z)$ is translated to...

$R \times \text{Adom}_C \cup S \times \text{Adom}_A$

Note that we do not need renaming

6) Conjunction

→ Same as disjunction but use $\cap$ instead of $\cup$

EXAMPLE

Customer : CustID, Name

Account : Number, CustID

Translate $\exists x_4 \, \text{Customer}(x_1, x_2) \wedge \text{Account}(x_3, x_4) \wedge x_1 = x_4$

$\eta: \{ x_1 \mapsto \text{CustID}, x_2 \mapsto \text{Name}, x_3 \mapsto \text{Number}, x_4 \mapsto \text{CustID}' \}$

$\pi_{\text{CustID, Name, Number}}$ (5) $\leftarrow \exists x_4$ $(\text{Customer} \times \text{Adom}_{\text{Number}} \times \text{Adom}_{\text{CustID}'}) \cap$ (4) $\times \text{Adom}_{\text{Name}}$

(5) $\wedge$

(1) $\text{Customer}(x_1, x_2)$ — Customer

(2) $\times \text{Adom}_{\text{CustID}} \cap$ (3) $\times \text{Adom}_{\text{Number}}$

(4) $\wedge$

(2) $\text{Account}(x_3, x_4)$ — $\rho_{\text{CustID} \to \text{CustID}'}(\text{Account})$

$x_1 = x_4$ (3) — $\sigma_{\text{CustID} = \text{CustID}'}(\text{Adom}_{\text{CustID}} \times \text{Adom}_{\text{CustID}'})$