UNTAR
Universitas Tarumanagara

Terakreditasi BAN PT A (Unggul) QS STARS RATING SYSTEM 2019 JABEE CPA AUSTRALIA ICAEW CHARTERED ACCOUNTANTS

UNTAR untuk INDONESIA

DATABASE SYSTEMS
TK13021

TEKNIK INFORMATIKA

UNIVERSITAS TARUMANAGARA

Course Schedule

1. Introduction to Databases
2. Database Environment
3. The Relational Model
4. Relational Algebra
5. Relational Calculus
6. Database Planning, Design, and Administration

7 – 9 Entity-Relationship Modeling

10–12 Normalization

13. Conceptual Database Design
14. Logical Database Design

Chapter 5

Relational Calculus

Relational Calculus

- **Relational calculus query specifies *what* is to be retrieved rather than *how* to retrieve it.**
  - **No description of how to evaluate a query.**
- **In first-order logic (or predicate calculus), *predicate* is a truth-valued function with arguments.**
- **When we substitute values for the arguments, function yields an expression, called a *proposition*, which can be either true or false.**

Relational Calculus

- **If predicate contains a variable (e.g. '*x* is a member of staff'), there must be a range for *x*.**
- **When we substitute some values of this range for *x*, proposition may be true; for other values, it may be false.**
- **When applied to databases, relational calculus has forms: *tuple* and *domain.***

Tuple Relational Calculus

- **Interested in finding tuples for which a predicate is true. Based on use of tuple variables.**
- **Tuple variable is a variable that 'ranges over' a named relation: i.e., variable whose only permitted values are tuples of the relation.**
- **Specify range of a tuple variable *S* as the Staff relation as:**

  Staff(S)
- **To find set of all tuples S such that P(S) is true:**

  {S | P(S)}

Tuple Relational Calculus - Example

- **To find details of all staff earning more than £10,000:**

  $\{S \mid Staff(S) \wedge S.salary > 10000\}$

- **To find a particular attribute, such as salary, write:**

  $\{S.salary \mid Staff(S) \wedge S.salary > 10000\}$

Tuple Relational Calculus

- **Can use two *quantifiers* to tell how many instances the predicate applies to:**
  - **Existential quantifier ∃ (‘there exists’)**
  - **Universal quantifier ∀ (‘for all’)**

- **Tuple variables qualified by ∀ or ∃ are called *bound* variables, otherwise called *free* variables.**

Tuple Relational Calculus

- **Existential quantifier used in formulae that must be true for at least one instance, such as:**

$$\text{Staff}(S) \wedge (\exists B)(\text{Branch}(B) \wedge (B.\text{branchNo} = S.\text{branchNo}) \wedge B.\text{city} = \text{‘London’})$$

- **Means ‘There exists a Branch tuple with same branchNo as the branchNo of the current Staff tuple, *S*, and is located in London’.** (terdapat tuple branch dengan branchNo yang sama dengan branchNo dari tuple Staff , S, dan berlokasi di London’)

Tuple Relational Calculus

- **Universal quantifier is used in statements about every instance, such as:**

  (∀B) (B.city ≠ 'Paris')

- **Means 'For all Branch tuples, the address is not in Paris'.**

- **Can also use ~(∃B) (B.city = 'Paris') which means 'There are no branches with an address in Paris'.**

Tuple Relational Calculus

- Formulae should be unambiguous and make sense.
- A (well-formed) formula is made out of <u>atoms</u>:
  - $R(S_i)$, where S_i is a tuple variable and R is a relation
  - $S_i.a_1 \; \theta \; S_j.a_2$
  - $S_i.a_1 \; \theta \; c$
- Can recursively build up formulae from atoms:
  - An atom is a formula
  - If F_1 and F_2 are formulae, so are their conjunction, $F_1 \wedge F_2$; disjunction, $F_1 \vee F_2$; and negation, $\sim F_1$
  - If F is a formula with free variable X, then $(\exists X)(F)$ and $(\forall X)(F)$ are also formulae.

Example - Tuple Relational Calculus

a) List the names of all managers who earn more than £25,000.

$\{S.fName, S.lName \mid Staff(S) \wedge S.position = \text{'Manager'} \wedge S.salary > 25000\}$

b) List the staff who manage properties for rent in Glasgow.

$\{S \mid Staff(S) \wedge (\exists P)\ (PropertyForRent(P) \wedge (P.staffNo = S.staffNo) \wedge P.city = \text{'Glasgow'})\}$

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Example - Tuple Relational Calculus

c) List the names of staff who currently do not manage any properties.

$$\{S.fName, S.lName \mid Staff(S) \wedge (\sim(\exists P)(PropertyForRent(P) \wedge (S.staffNo = P.staffNo)))\}$$

Or

$$\{S.fName, S.lName \mid Staff(S) \wedge ((\forall P)(\sim PropertyForRent(P) \vee \sim(S.staffNo = P.staffNo)))\}$$

Example - Tuple Relational Calculus

- List the names of clients who have viewed a property for rent in Glasgow.

$$\{C.fName, C.lName \mid Client(C) \wedge ((\exists V)(\exists P)$$
$$(Viewing(V) \wedge PropertyForRent(P) \wedge$$
$$(C.clientNo = V.clientNo) \wedge$$
$$(V.propertyNo=P.propertyNo) \wedge P.city = \text{‘Glasgow’}))\}$$

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Client

clientNo	fName	lName	telNo	prefType	maxRent
CR76	John	Kay	0207-774-5632	Flat	425
CR56	Aline	Stewart	0141-848-1825	Flat	350
CR74	Mike	Ritchie	01475-392178	House	750
CR62	Mary	Tregear	01224-196720	Flat	600

Viewing

clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-04	too small
CR76	PG4	20-Apr-04	too remote
CR56	PG4	26-May-04	
CR62	PA14	14-May-04	no dining room
CR56	PG36	28-Apr-04	

Other Languages

- **Transform-oriented languages are non-procedural languages that use relations to transform input data into required outputs (e.g. SQL).**

- **Graphical languages provide user with picture of the structure of the relation. User fills in example of what is wanted and system returns required data in that format (e.g. QBE).**

Other Languages

- **4GLs can create complete customized application using limited set of commands in a user-friendly, often menu-driven environment.**

- **Some systems accept a form of *natural language*, sometimes called a 5GL, although this development is still a an early stage.**

Latihan Soal TRC

1. Tampilkan hotelNo yang memiliki harga > 100
2. Tampilkan roomNo dan hotelNo yang bertipe Deluxe
3. Tampilkan hotelName untuk hotel yang berada di kota London
4. Tampilkan hotelName untuk semua hotel yang memiliki harga > 50

The following tables form part of a database held in a relational DBMS:

Hotel	(hotelNo, hotelName, city)
Room	(roomNo, hotelNo, type, price)
Booking	(hotelNo, guestNo, dateFrom, dateTo, roomNo)
Guest	(guestNo, guestName, guestAddress)

where Hotel contains hotel details and hotelNo is the primary key;
Room contains room details for each hotel and (roomNo, hotelNo) forms the primary key;
Booking contains details of bookings and (hotelNo, guestNo, dateFrom) forms the primary key;
Guest contains guest details and guestNo is the primary key.

Thank You

Reference: Database Systems A Practical Approach to Design, Implementation, and Management Fourth Edition.

Thomas M. Connolly and Carolyn E. Begg

UNTAR
Universitas Tarumanagara

UNTAR untuk INDONESIA