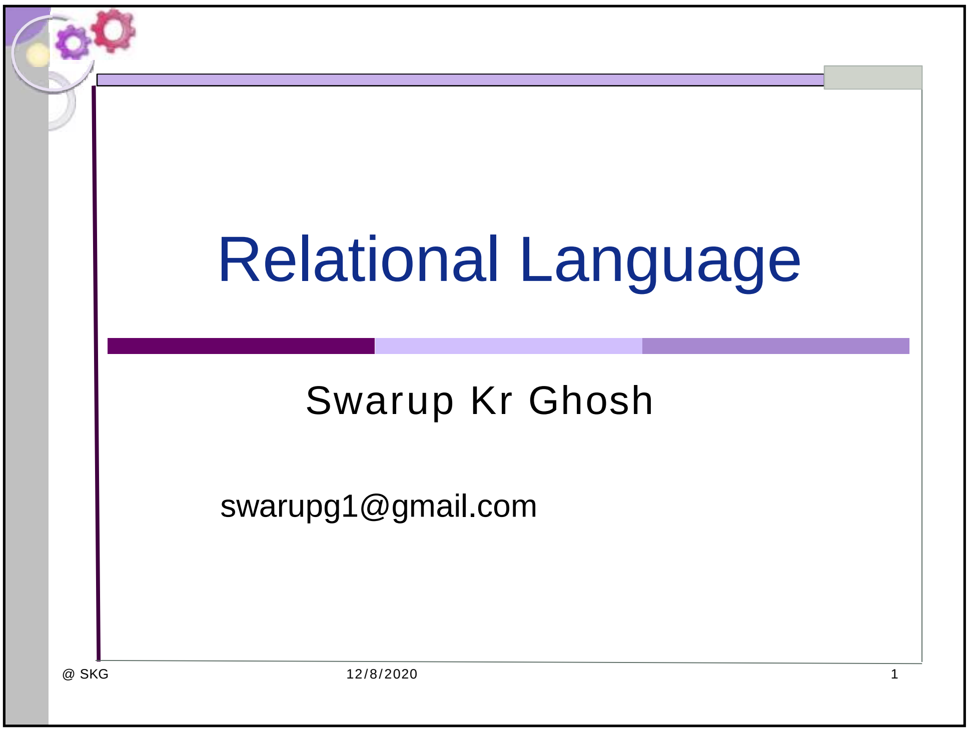
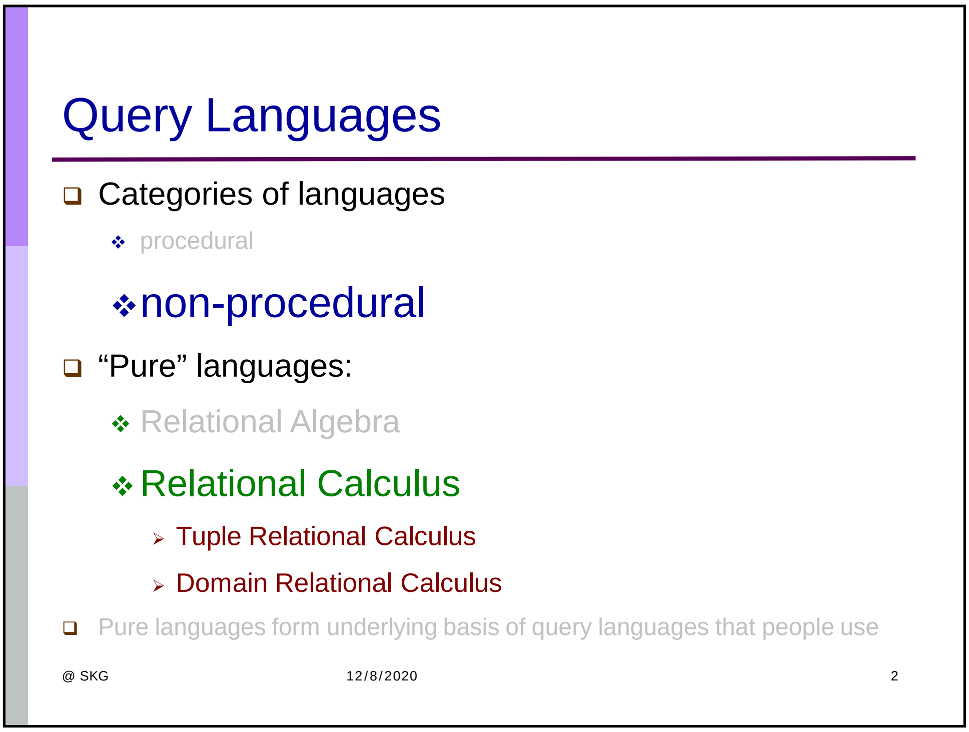


Relational Language

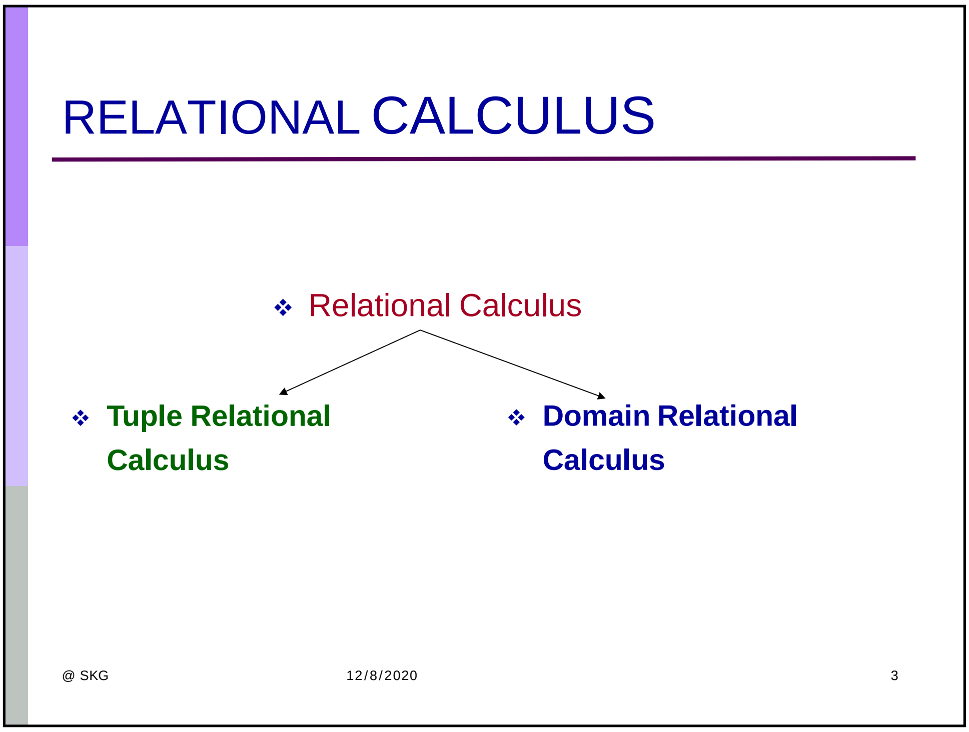
Swarup Kr Ghosh

swarupg1@gmail.com



Query Languages

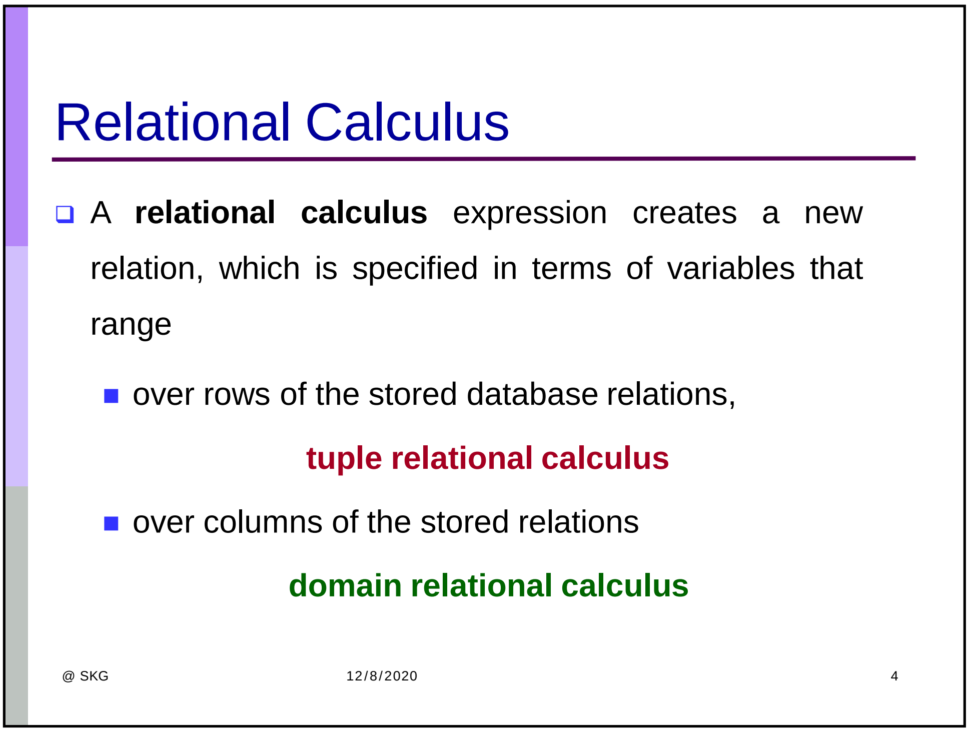
- Categories of languages
  - procedural
  - non-procedural
- “Pure” languages:
  - Relational Algebra
  - Relational Calculus
    - Tuple Relational Calculus
    - Domain Relational Calculus
- Pure languages form underlying basis of query languages that people use



RELATIONAL CALCULUS

- Relational Calculus
  - **Tuple Relational Calculus**
  - **Domain Relational Calculus**

@ SKG 12/8/2020 3



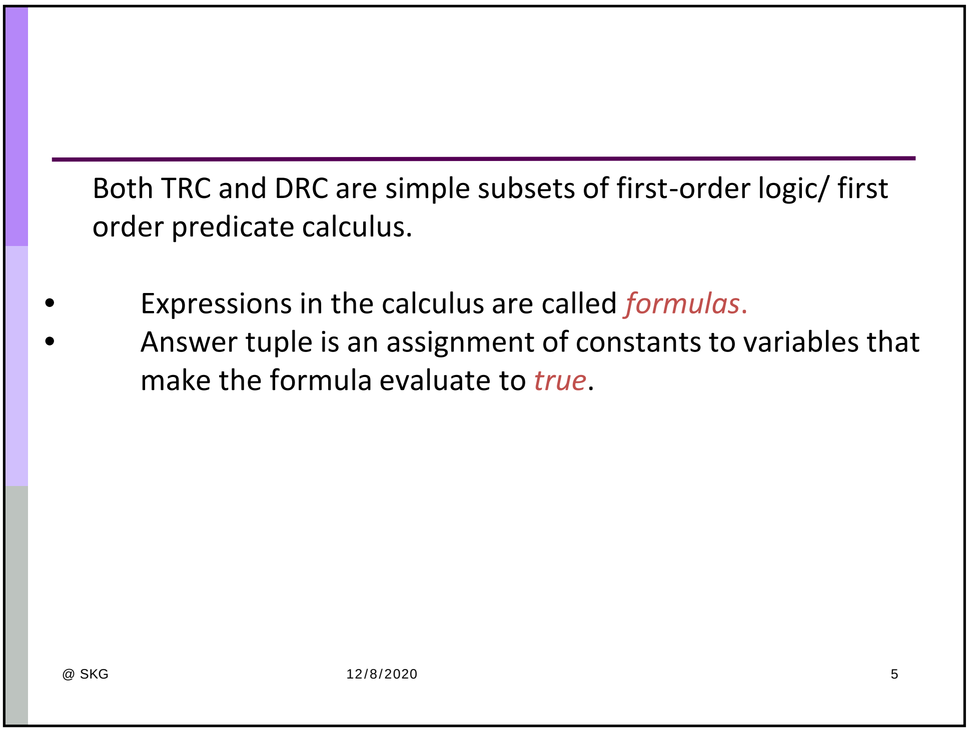
Relational Calculus

- A **relational calculus** expression creates a new relation, which is specified in terms of variables that range
  - over rows of the stored database relations,

    tuple relational calculus
  - over columns of the stored relations

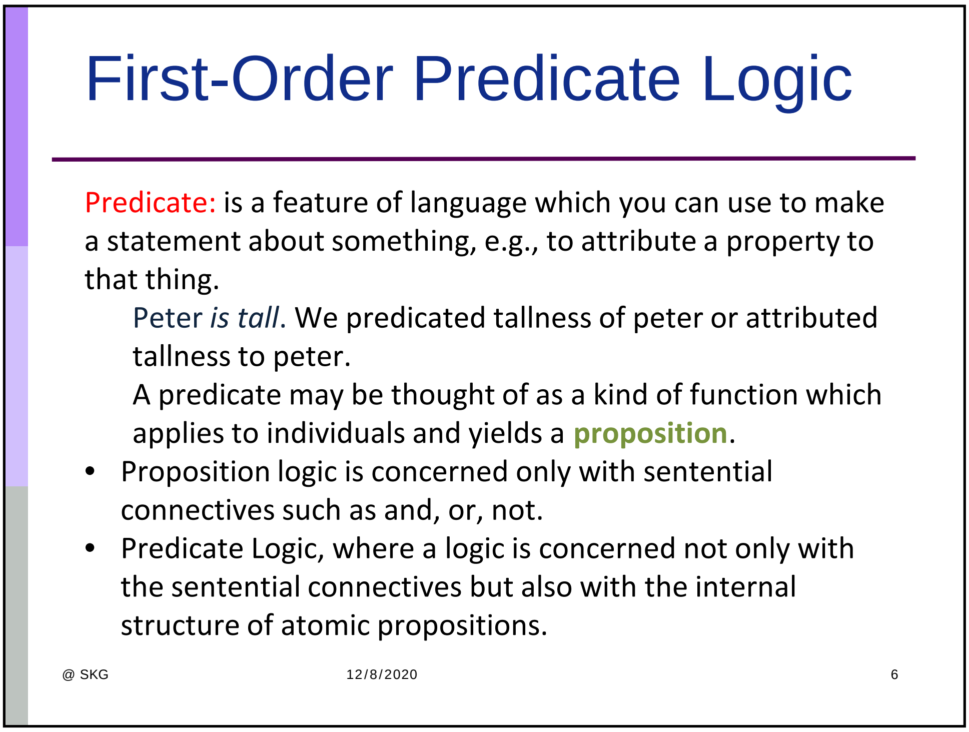
    domain relational calculus

@ SKG 12/8/2020 4



Both TRC and DRC are simple subsets of first-order logic/ first order predicate calculus.

- Expressions in the calculus are called *formulas.*
- Answer tuple is an assignment of constants to variables that make the formula evaluate to *true*.



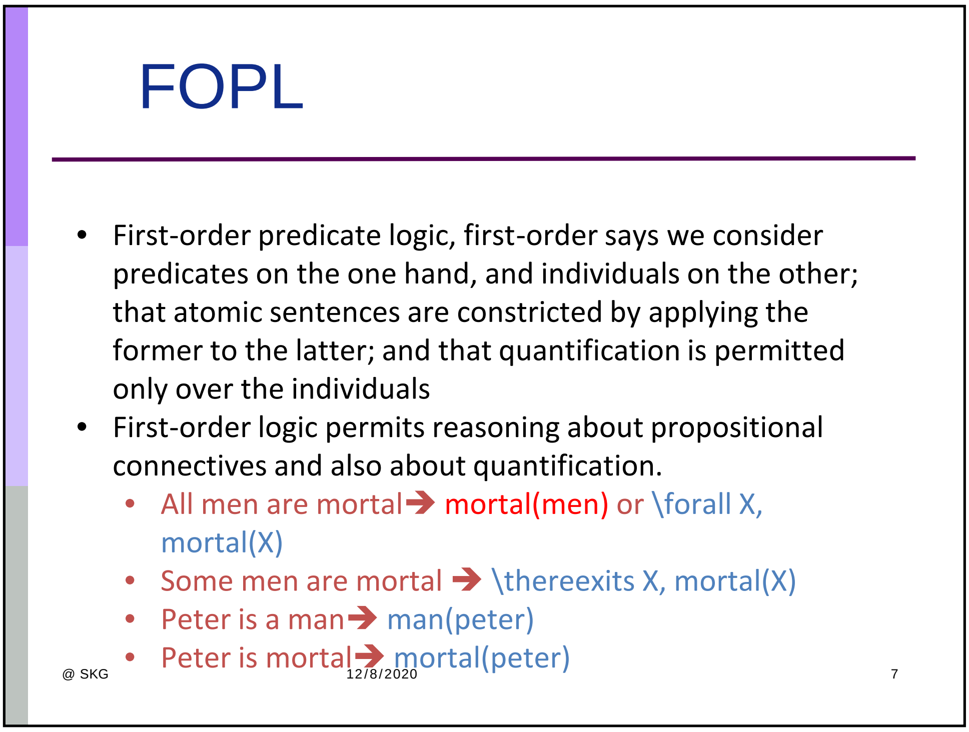
First-Order Predicate Logic

Predicate: is a feature of language which you can use to make a statement about something, e.g., to attribute a property to that thing.

Peter *is tall*. We predicated tallness of peter or attributed tallness to peter.

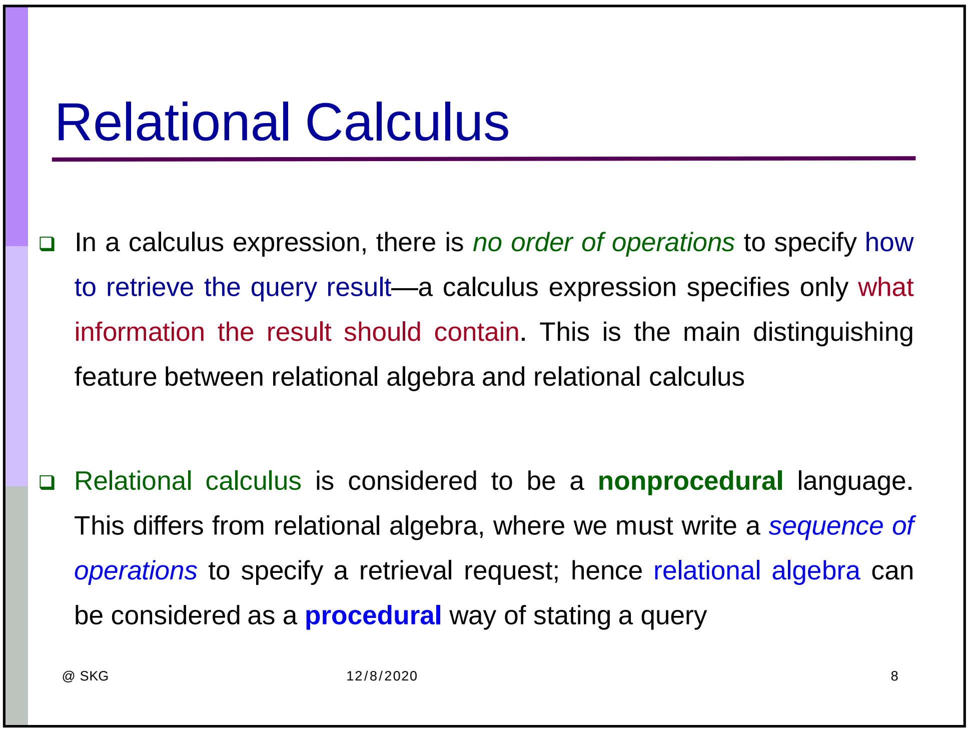
A predicate may be thought of as a kind of function which applies to individuals and yields a **proposition**.

- Proposition logic is concerned only with sentential connectives such as and, or, not.
- Predicate Logic, where a logic is concerned not only with the sentential connectives but also with the internal structure of atomic propositions.



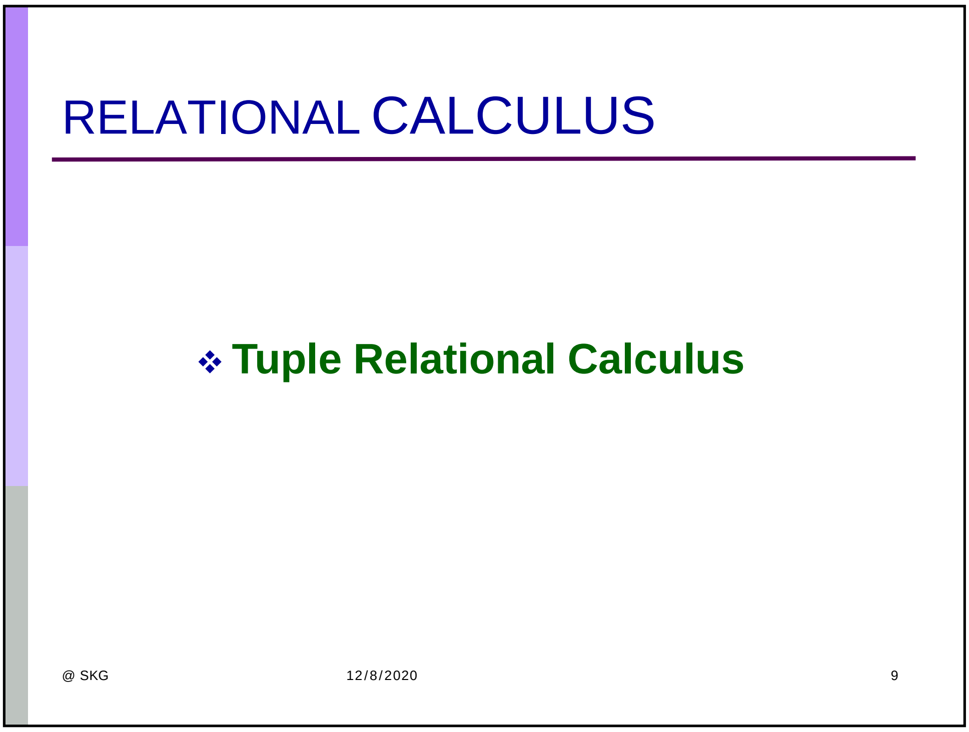
FOPL

- First-order predicate logic, first-order says we consider predicates on the one hand, and individuals on the other; that atomic sentences are constricted by applying the former to the latter; and that quantification is permitted only over the individuals
- First-order logic permits reasoning about propositional connectives and also about quantification.
  - All men are mortal→ mortal(men) or \forall X, mortal(X)
  - Some men are mortal → \thereexits X, mortal(X)
  - Peter is a man→ man(peter)
  - Peter is mortal→ mortal(peter)



Relational Calculus

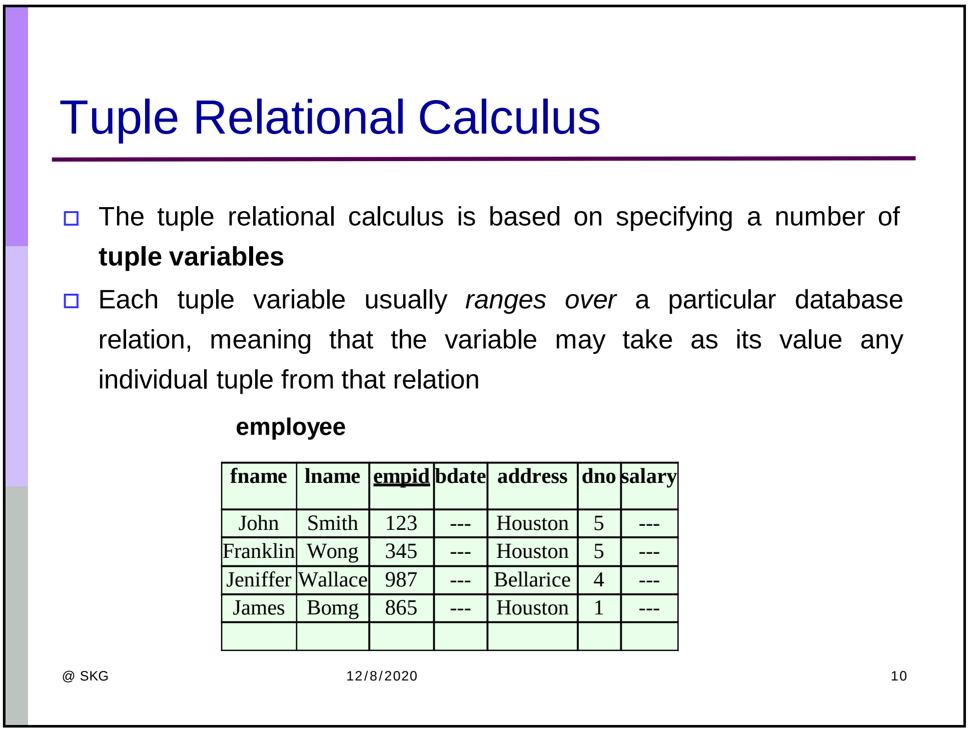
- ❑ In a calculus expression, there is *no order of operations* to specify how to retrieve the query result—a calculus expression specifies only what information the result should contain. This is the main distinguishing feature between relational algebra and relational calculus
- ❑ Relational calculus is considered to be a **nonprocedural** language. This differs from relational algebra, where we must write a *sequence of operations* to specify a retrieval request; hence relational algebra can be considered as a **procedural** way of stating a query



RELATIONAL CALCULUS

❖ Tuple Relational Calculus

@ SKG 12/8/2020 9



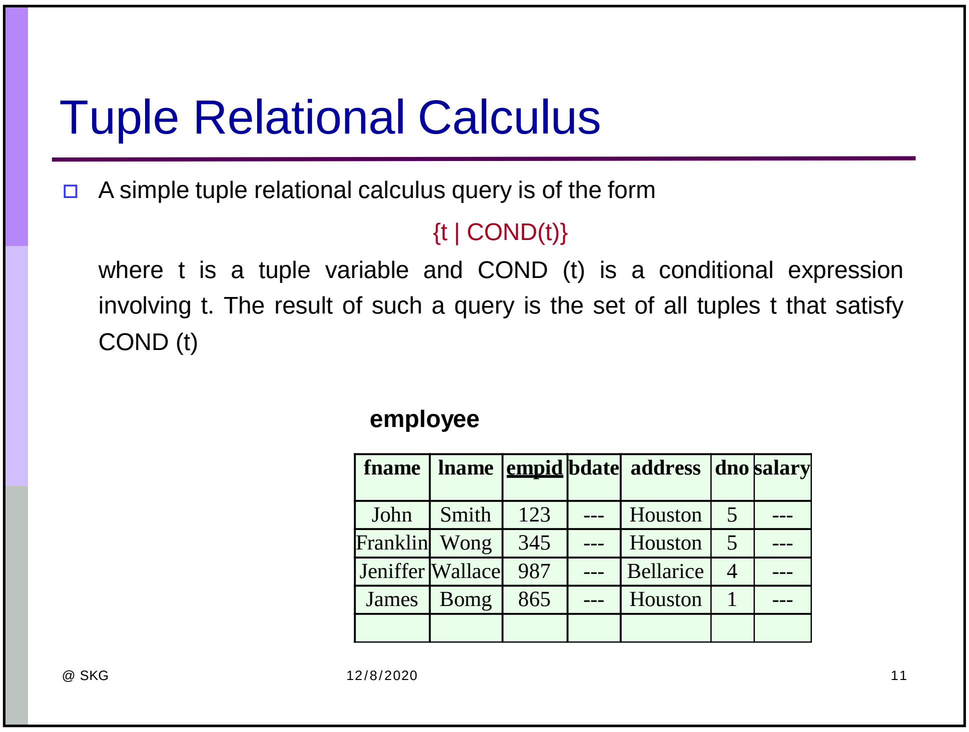
Tuple Relational Calculus

- □ The tuple relational calculus is based on specifying a number of **tuple variables**
- □ Each tuple variable usually *ranges over* a particular database relation, meaning that the variable may take as its value any individual tuple from that relation

employee

fname	lname	empid	bdate	address	dno	salary
John	Smith	123	---	Houston	5	---
Franklin	Wong	345	---	Houston	5	---
Jeniffer	Wallace	987	---	Bellarice	4	---
James	Bomg	865	---	Houston	1	---

@ SKG 12/8/2020 10



Tuple Relational Calculus

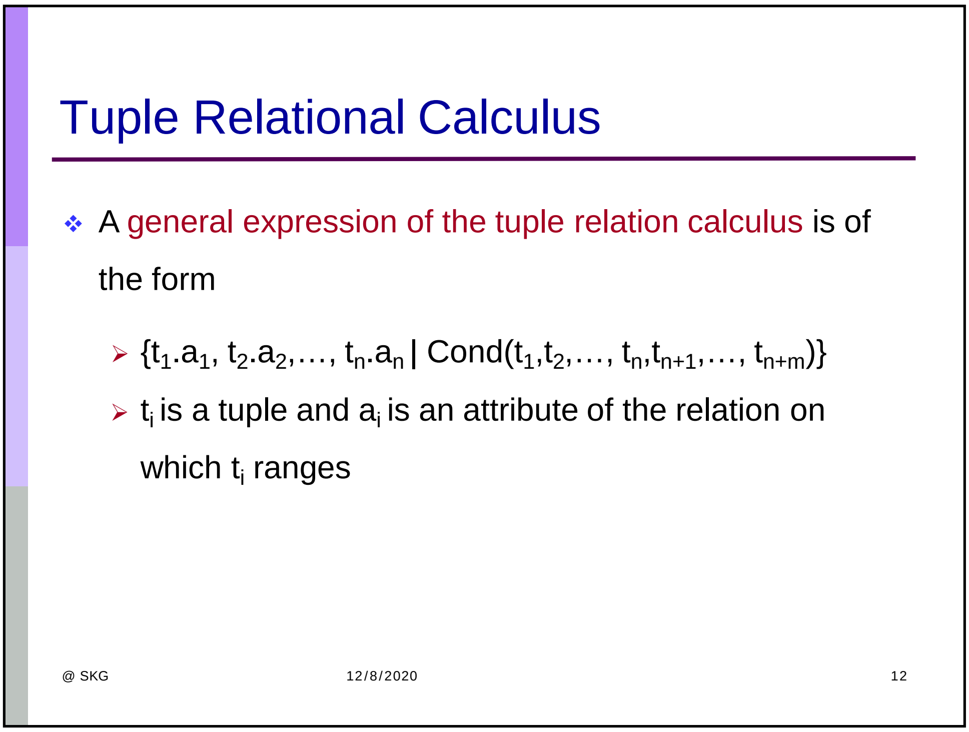
- A simple tuple relational calculus query is of the form

  {t | COND(t)}

  where t is a tuple variable and COND (t) is a conditional expression involving t. The result of such a query is the set of all tuples t that satisfy COND (t)

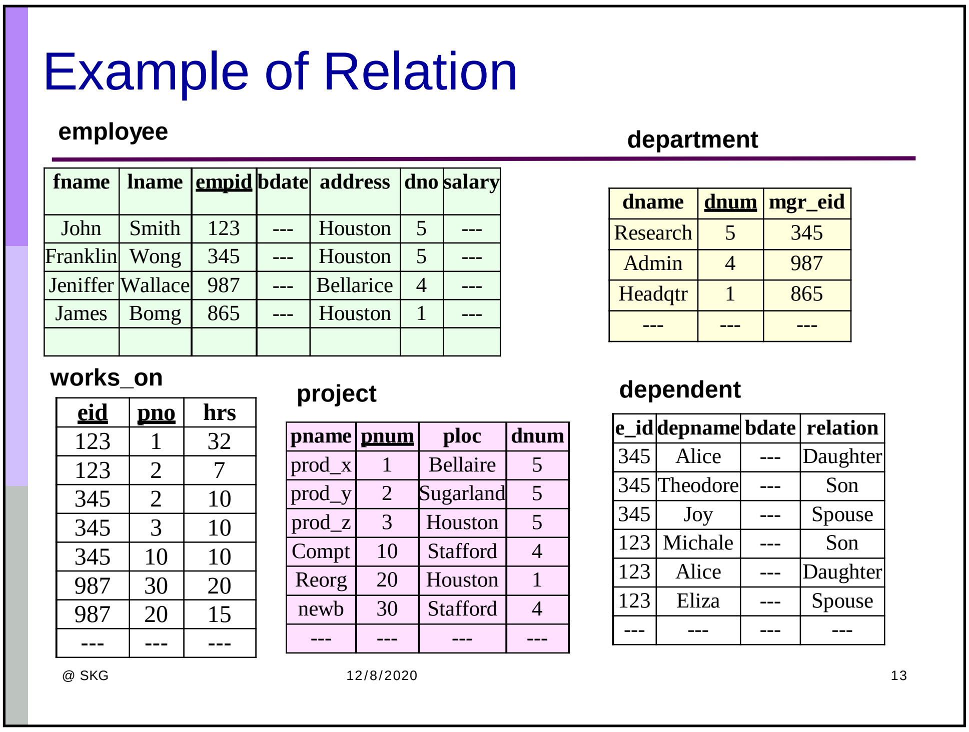
employee

fname	lname	<u>empid</u>	bdate	address	dno	salary
John	Smith	123	---	Houston	5	---
Franklin	Wong	345	---	Houston	5	---
Jeniffer	Wallace	987	---	Bellarice	4	---
James	Bomg	865	---	Houston	1	---



Tuple Relational Calculus

- A general expression of the tuple relation calculus is of the form
  - $\{t_1.a_1, t_2.a_2, \ldots, t_n.a_n \mid Cond(t_1, t_2, \ldots, t_n, t_{n+1}, \ldots, t_{n+m})\}$
  - t_i is a tuple and a_i is an attribute of the relation on which t_i ranges



Example of Relation

employee

fname	lname	empid	bdate	address	dno	salary
John	Smith	123	---	Houston	5	---
Franklin	Wong	345	---	Houston	5	---
Jeniffer	Wallace	987	---	Bellarice	4	---
James	Bomg	865	---	Houston	1	---

department

dname	dnum	mgr_eid
Research	5	345
Admin	4	987
Headqtr	1	865
---	---	---

works_on

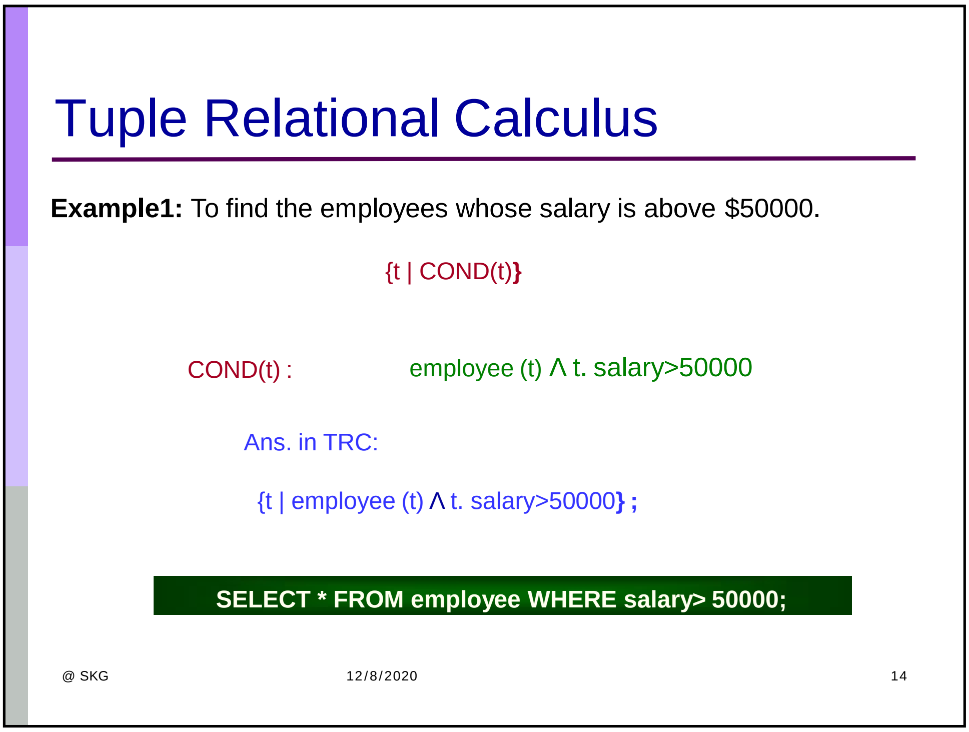
eid	pno	hrs
123	1	32
123	2	7
345	2	10
345	3	10
345	10	10
987	30	20
987	20	15
---	---	---

project

pname	pnum	ploc	dnum
prod_x	1	Bellaire	5
prod_y	2	Sugarland	5
prod_z	3	Houston	5
Compt	10	Stafford	4
Reorg	20	Houston	1
newb	30	Stafford	4
---	---	---	---

dependent

e_id	depname	bdate	relation
345	Alice	---	Daughter
345	Theodore	---	Son
345	Joy	---	Spouse
123	Michale	---	Son
123	Alice	---	Daughter
123	Eliza	---	Spouse
---	---	---	---



Tuple Relational Calculus

Example1: To find the employees whose salary is above $50000.

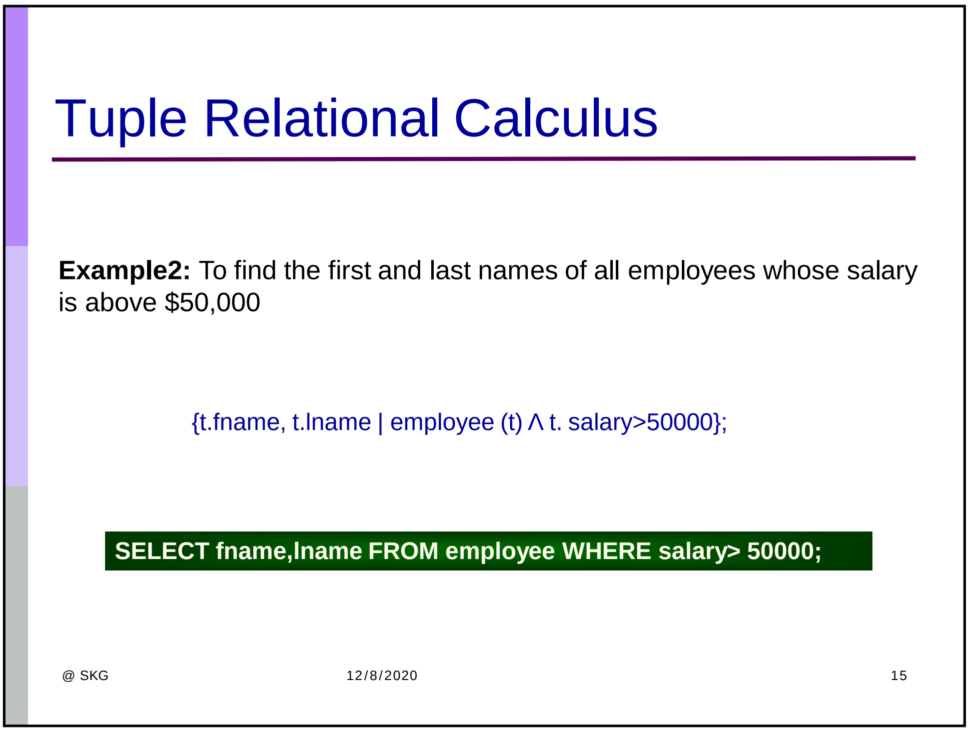
{t | COND(t)**}**

COND(t) : employee (t) Λ t. salary>50000

Ans. in TRC:

{t | employee (t) Λ t. salary>50000**} ;**

SELECT * FROM employee WHERE salary> 50000;

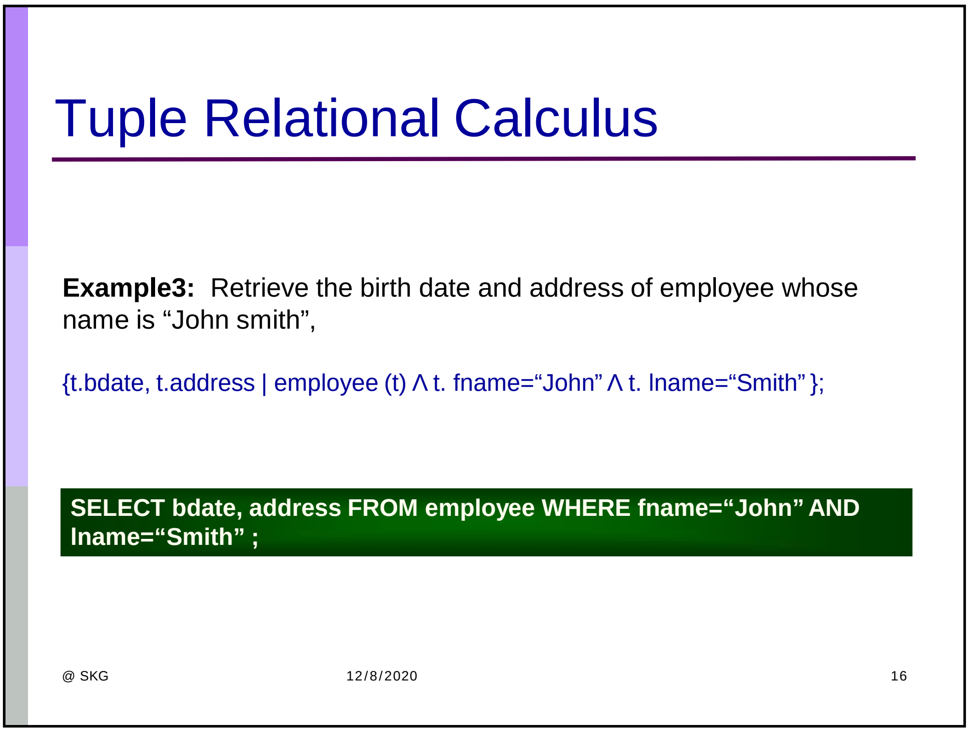


Tuple Relational Calculus

Example2: To find the first and last names of all employees whose salary is above $50,000

{t.fname, t.lname | employee (t) Λ t. salary>50000};

SELECT fname,lname FROM employee WHERE salary> 50000;



Tuple Relational Calculus

Example3: Retrieve the birth date and address of employee whose name is “John smith”,

{t.bdate, t.address | employee (t) Λ t. fname=“John” Λ t. lname=“Smith” };

SELECT bdate, address FROM employee WHERE fname=“John” AND lname=“Smith” ;

Example of Relation

employee

fname	lname	empid	bdate	address	dno	salary
John	Smith	123	---	Houston	5	---
Franklin	Wong	345	---	Houston	5	---
Jeniffer	Wallace	987	---	Bellarice	4	---
James	Bomg	865	---	Houston	1	---

department

dname	dnum	mgr_eid
Research	5	345
Admin	4	987
Headqtr	1	865
---	---	---

works_on

eid	pno	hrs
123	1	32
123	2	7
345	2	10
345	3	10
345	10	10
987	30	20
987	20	15
---	---	---

project

pname	pnum	ploc	dnum
prod_x	1	Bellaire	5
prod_y	2	Sugarland	5
prod_z	3	Houston	5
Compt	10	Stafford	4
Reorg	20	Houston	1
newb	30	Stafford	4
---	---	---	---

dependent

e_id	depname	bdate	relation
345	Alice	---	Daughter
345	Theodore	---	Son
345	Joy	---	Spouse
123	Michale	---	Son
123	Alice	---	Daughter
123	Eliza	---	Spouse
---	---	---	---

Tuple Relational Calculus-example

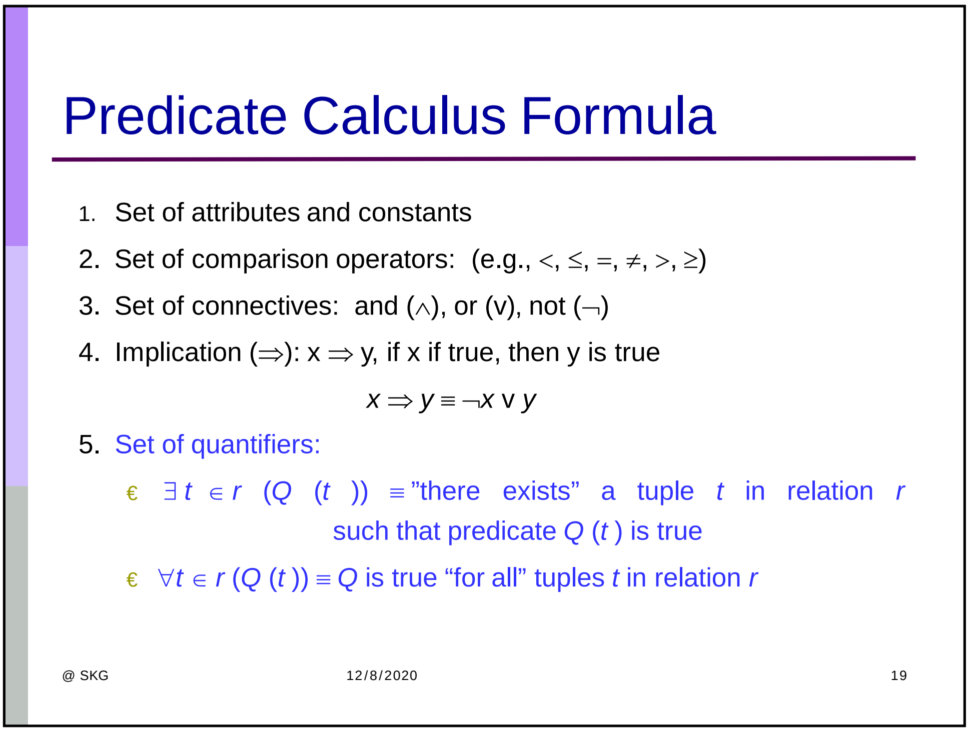
- **Example 4:** Retrieve the name and address of all employees who work for the ‘Research’ department.

  Ans :

  {t.fname, t.lname, t.address | employee (t) ∧ ((∃d) (department (d) ∧ d.dname=“research” ∧ d.dnum=t.dno))}

- The *only* *free tuple variables* in a relational calculus expression should be those that appear to the left of the bar (|).

```
SELECT fname, lname, address FROM employee, department WHERE
dname=“Research” AND dnum =dno ;
```



Predicate Calculus Formula

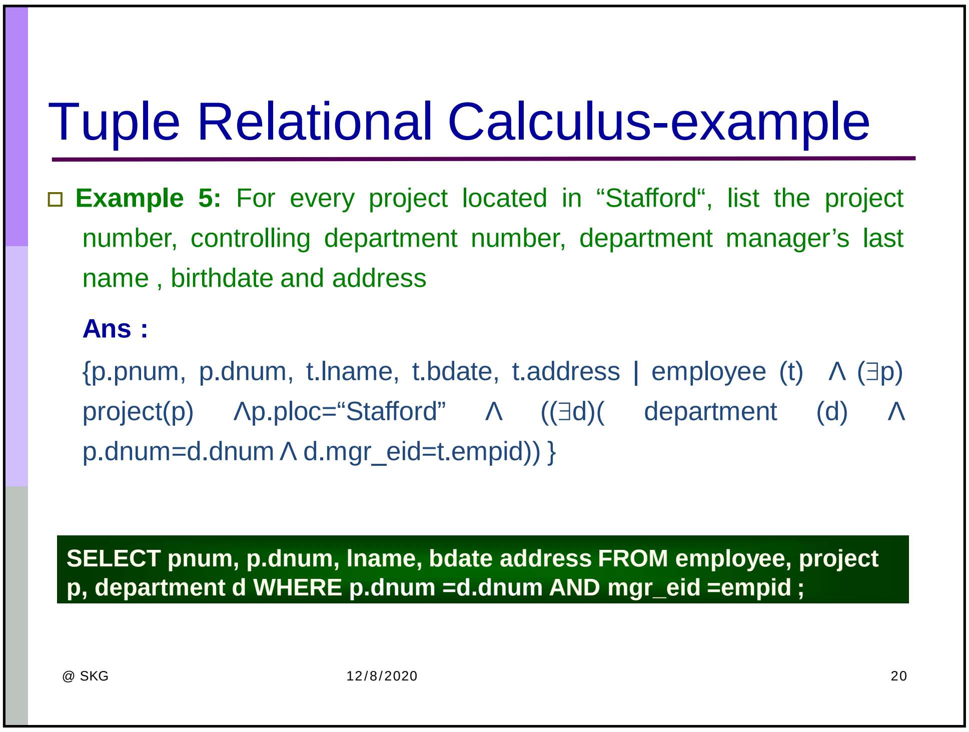
1. Set of attributes and constants
2. Set of comparison operators: (e.g., $<, \leq, =, \neq, >, \geq$)
3. Set of connectives: and ($\wedge$), or (v), not ($\neg$)
4. Implication ($\Rightarrow$): x $\Rightarrow$ y, if x if true, then y is true

$$x \Rightarrow y \equiv \neg x \vee y$$

5. Set of quantifiers:

€ $\exists\, t \in r\ (Q\ (t\ ))\ \equiv$ "there exists" a tuple *t* in relation *r* such that predicate *Q* (*t*) is true

€ $\forall t \in r\ (Q\ (t\ )) \equiv Q$ is true "for all" tuples *t* in relation *r*



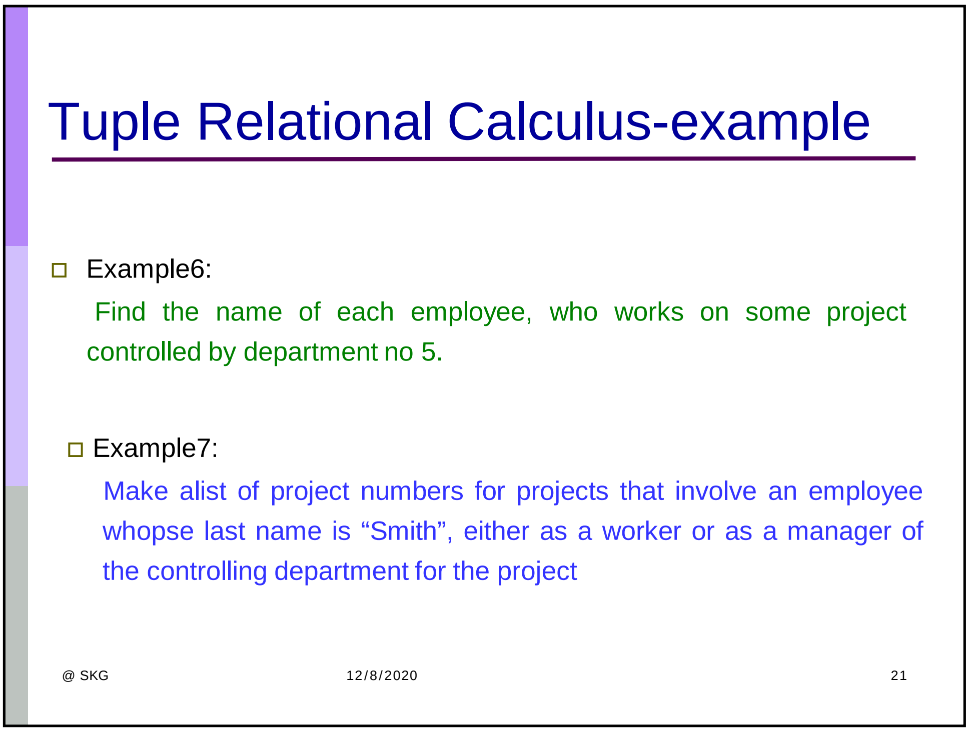
Tuple Relational Calculus-example

□ **Example 5:** For every project located in "Stafford", list the project number, controlling department number, department manager's last name , birthdate and address

Ans :

{p.pnum, p.dnum, t.lname, t.bdate, t.address | employee (t) Λ (∃p) project(p) Λp.ploc="Stafford" Λ ((∃d)(department (d) Λ p.dnum=d.dnum Λ d.mgr_eid=t.empid)) }

SELECT pnum, p.dnum, lname, bdate address FROM employee, project p, department d WHERE p.dnum =d.dnum AND mgr_eid =empid ;



Tuple Relational Calculus-example

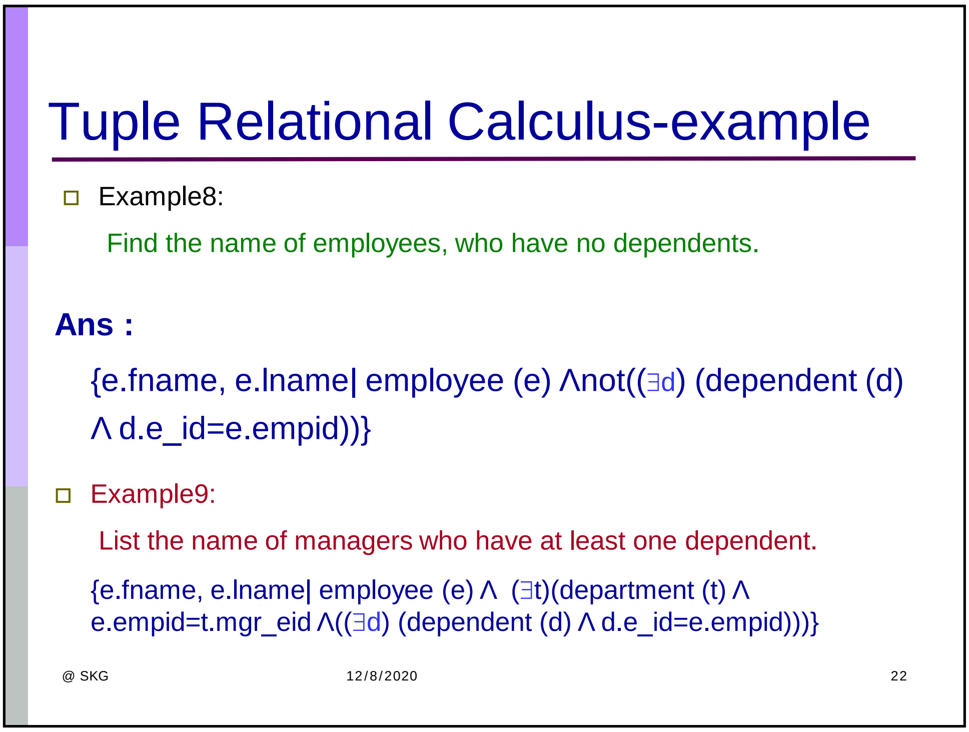
- □ Example6:

  Find the name of each employee, who works on some project controlled by department no 5.

- □ Example7:

  Make alist of project numbers for projects that involve an employee whopse last name is “Smith”, either as a worker or as a manager of the controlling department for the project

@ SKG 12/8/2020 21



Tuple Relational Calculus-example

- □ Example8:

  Find the name of employees, who have no dependents.

Ans :

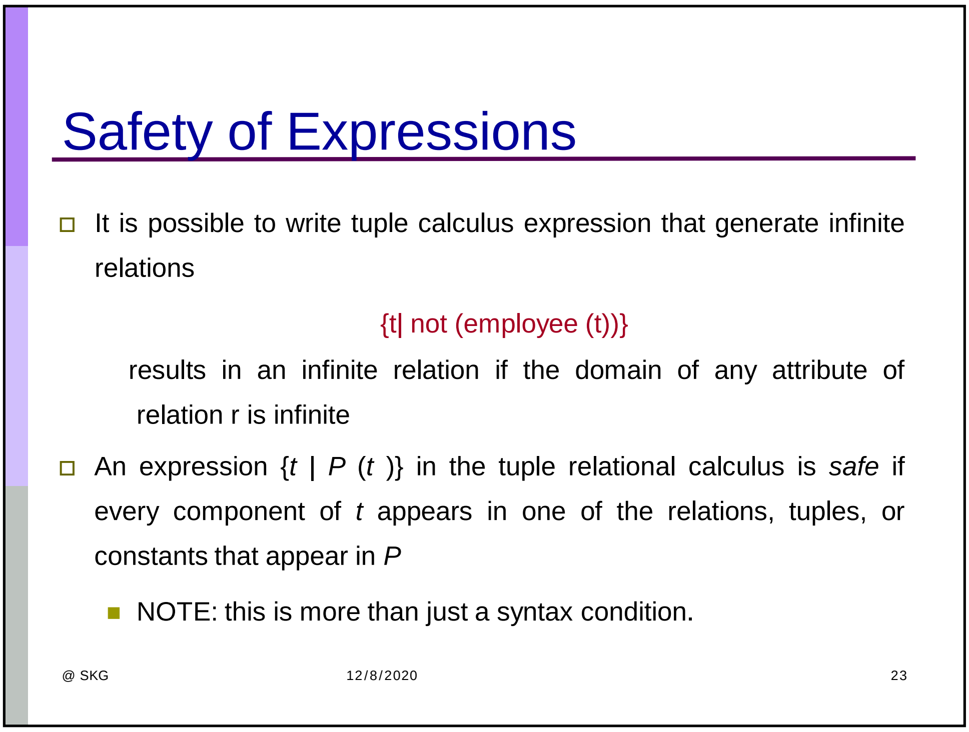
{e.fname, e.lname| employee (e) Λnot((∃d) (dependent (d) Λ d.e_id=e.empid))}

- □ Example9:

  List the name of managers who have at least one dependent.

  {e.fname, e.lname| employee (e) Λ (∃t)(department (t) Λ e.empid=t.mgr_eid Λ((∃d) (dependent (d) Λ d.e_id=e.empid)))}

@ SKG 12/8/2020 22



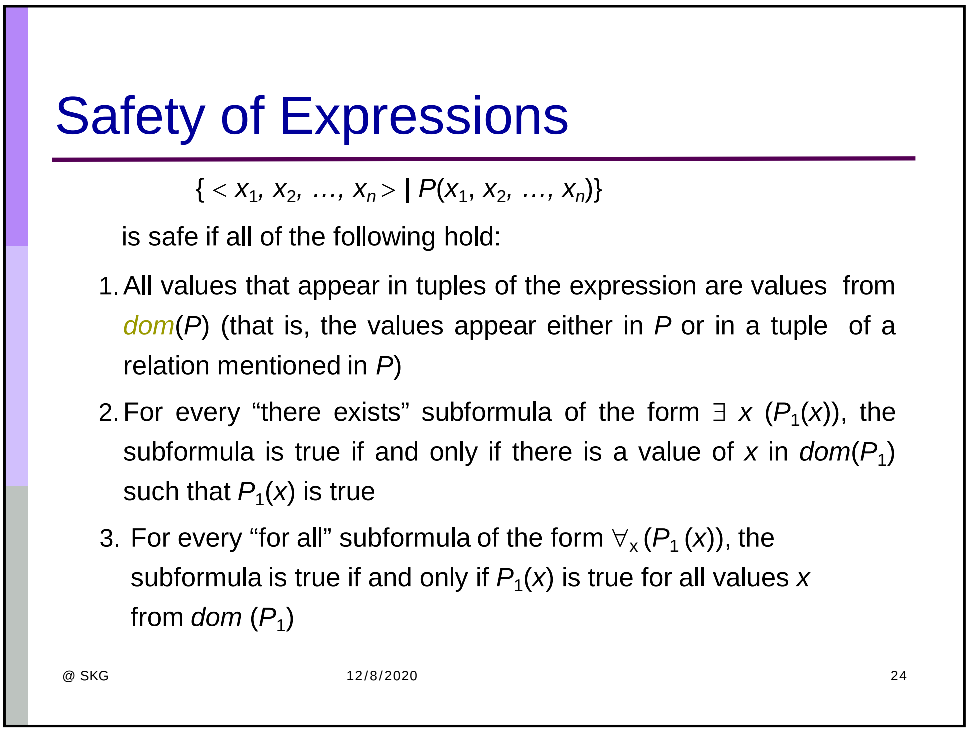
Safety of Expressions

- It is possible to write tuple calculus expression that generate infinite relations

  {t| not (employee (t))}

  results in an infinite relation if the domain of any attribute of relation r is infinite

- An expression {*t* | *P* (*t*)} in the tuple relational calculus is *safe* if every component of *t* appears in one of the relations, tuples, or constants that appear in *P*
  - NOTE: this is more than just a syntax condition.

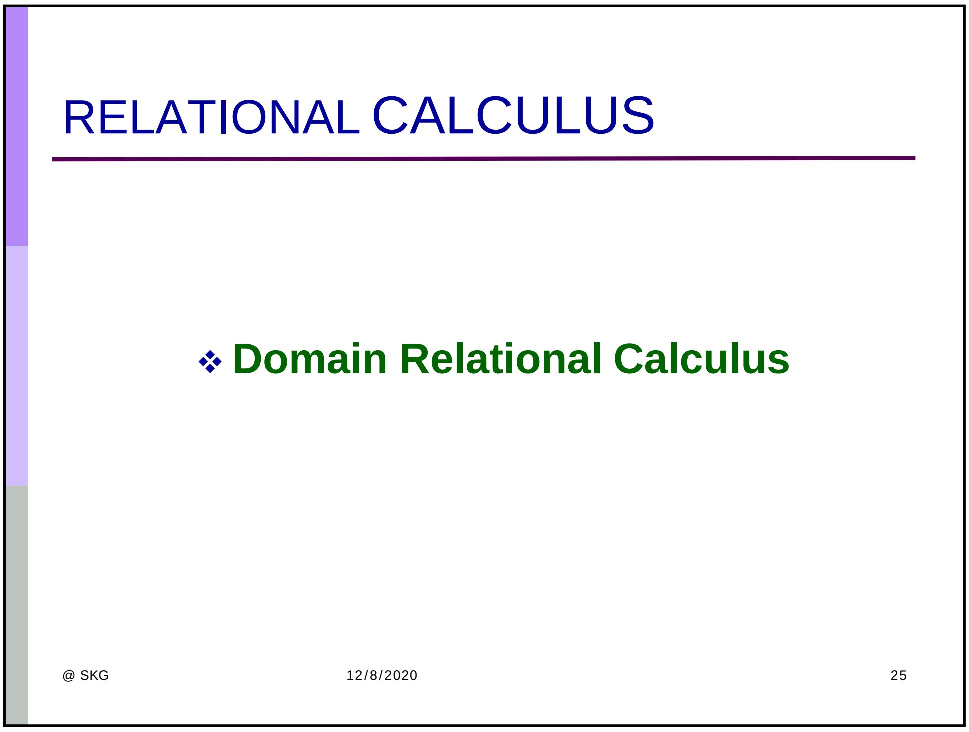


Safety of Expressions

$\{ < x_1, x_2, \ldots, x_n > | P(x_1, x_2, \ldots, x_n)\}$

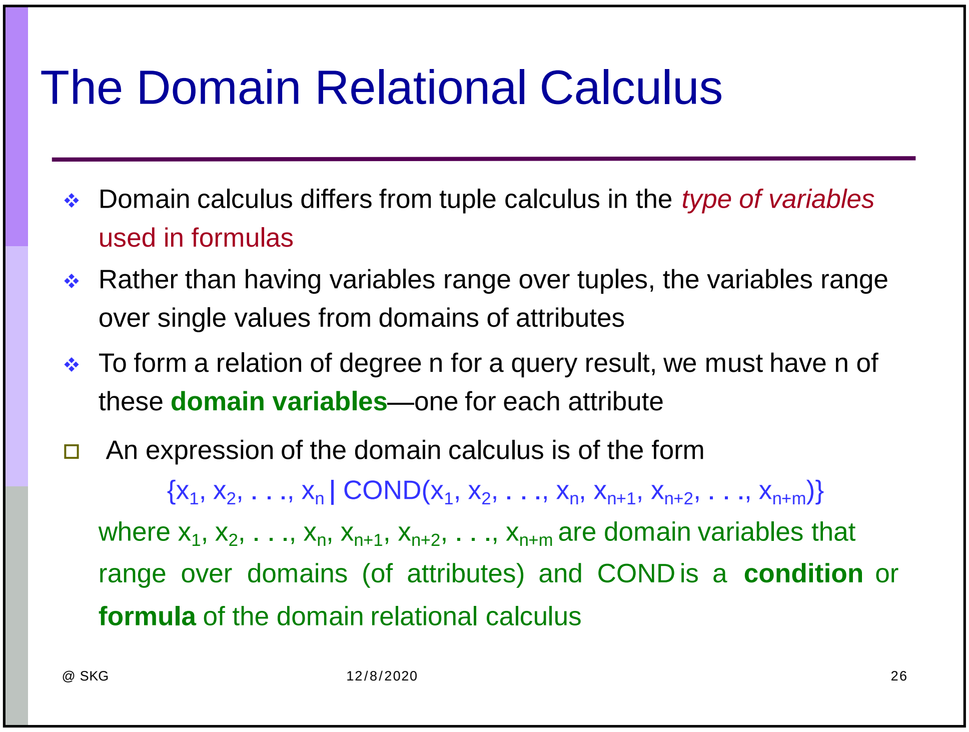
is safe if all of the following hold:

1. All values that appear in tuples of the expression are values from *dom*(*P*) (that is, the values appear either in *P* or in a tuple of a relation mentioned in *P*)
2. For every "there exists" subformula of the form $\exists\, x\, (P_1(x))$, the subformula is true if and only if there is a value of *x* in $dom(P_1)$ such that $P_1(x)$ is true
3. For every "for all" subformula of the form $\forall_x (P_1 (x))$, the subformula is true if and only if $P_1(x)$ is true for all values *x* from *dom* (P_1)



RELATIONAL CALCULUS

- **Domain Relational Calculus**



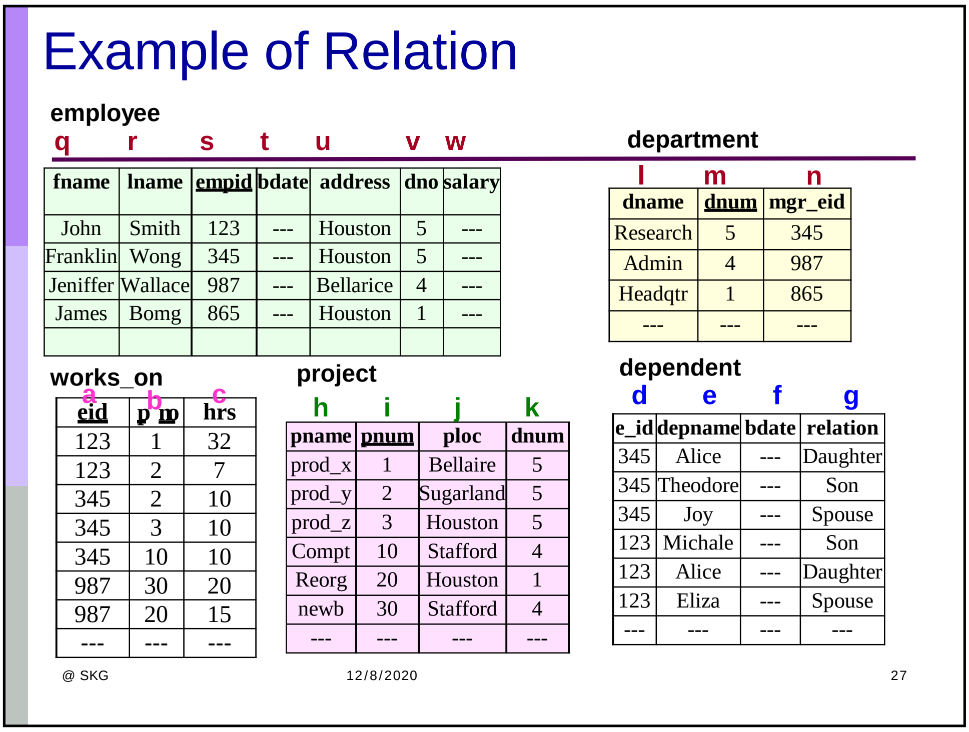
The Domain Relational Calculus

- Domain calculus differs from tuple calculus in the *type of variables* used in formulas
- Rather than having variables range over tuples, the variables range over single values from domains of attributes
- To form a relation of degree n for a query result, we must have n of these **domain variables**—one for each attribute

□ An expression of the domain calculus is of the form

$$\{x_1, x_2, . . ., x_n \mid COND(x_1, x_2, . . ., x_n, x_{n+1}, x_{n+2}, . . ., x_{n+m})\}$$

where $x_1, x_2, . . ., x_n, x_{n+1}, x_{n+2}, . . ., x_{n+m}$ are domain variables that range over domains (of attributes) and COND is a **condition** or **formula** of the domain relational calculus



Example of Relation

employee

q r s t u v w

fname	lname	empid	bdate	address	dno	salary
John	Smith	123	---	Houston	5	---
Franklin	Wong	345	---	Houston	5	---
Jeniffer	Wallace	987	---	Bellarice	4	---
James	Bomg	865	---	Houston	1	---

department

l m n

dname	dnum	mgr_eid
Research	5	345
Admin	4	987
Headqtr	1	865
---	---	---

works_on

a b c

eid	p no	hrs
123	1	32
123	2	7
345	2	10
345	3	10
345	10	10
987	30	20
987	20	15
---	---	---

project

h i j k

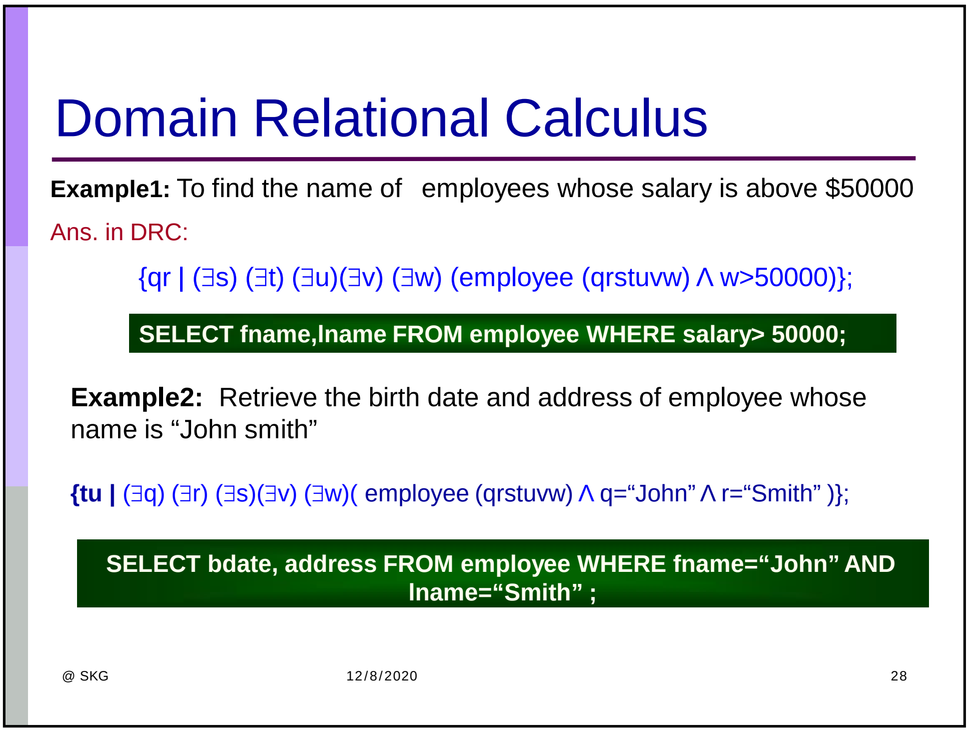
pname	pnum	ploc	dnum
prod_x	1	Bellaire	5
prod_y	2	Sugarland	5
prod_z	3	Houston	5
Compt	10	Stafford	4
Reorg	20	Houston	1
newb	30	Stafford	4
---	---	---	---

dependent

d e f g

e_id	depname	bdate	relation
345	Alice	---	Daughter
345	Theodore	---	Son
345	Joy	---	Spouse
123	Michale	---	Son
123	Alice	---	Daughter
123	Eliza	---	Spouse
---	---	---	---

@ SKG 12/8/2020 27



Domain Relational Calculus

Example1: To find the name of employees whose salary is above $50000

Ans. in DRC:

{qr | (∃s) (∃t) (∃u)(∃v) (∃w) (employee (qrstuvw) Λ w>50000)};

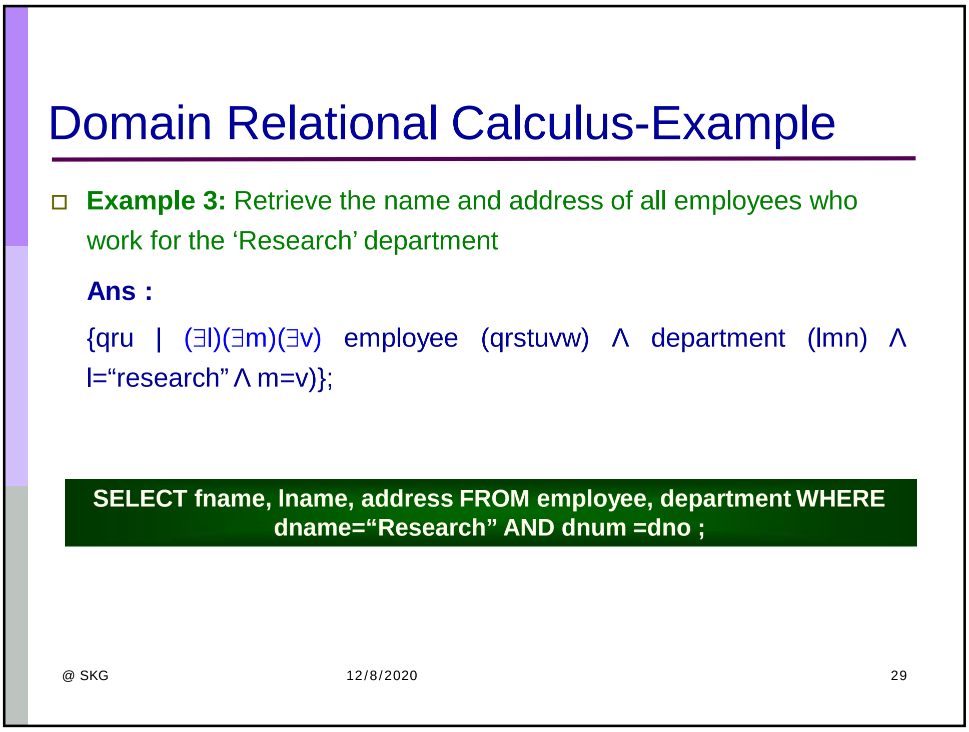
SELECT fname,lname FROM employee WHERE salary> 50000;

Example2: Retrieve the birth date and address of employee whose name is "John smith"

{tu | (∃q) (∃r) (∃s)(∃v) (∃w)(employee (qrstuvw) Λ q="John" Λ r="Smith")};

SELECT bdate, address FROM employee WHERE fname="John" AND lname="Smith" ;

@ SKG 12/8/2020 28



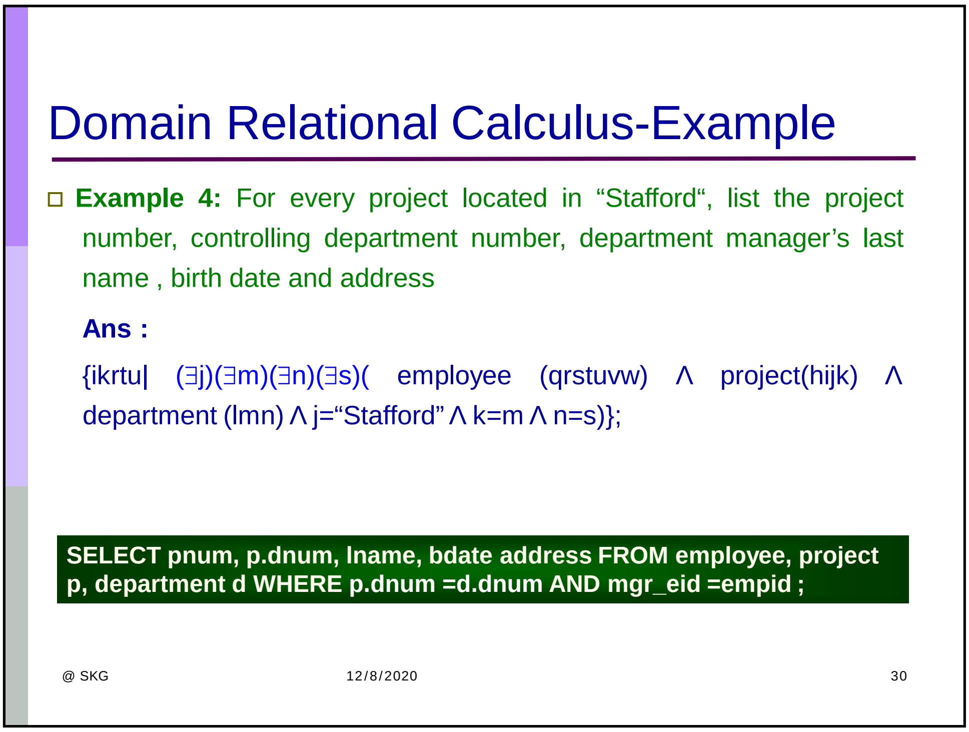
Domain Relational Calculus-Example

- **Example 3:** Retrieve the name and address of all employees who work for the ‘Research’ department

Ans :

{qru | (∃l)(∃m)(∃v) employee (qrstuvw) Λ department (lmn) Λ l=“research” Λ m=v)};

```
SELECT fname, lname, address FROM employee, department WHERE
dname=“Research” AND dnum =dno ;
```



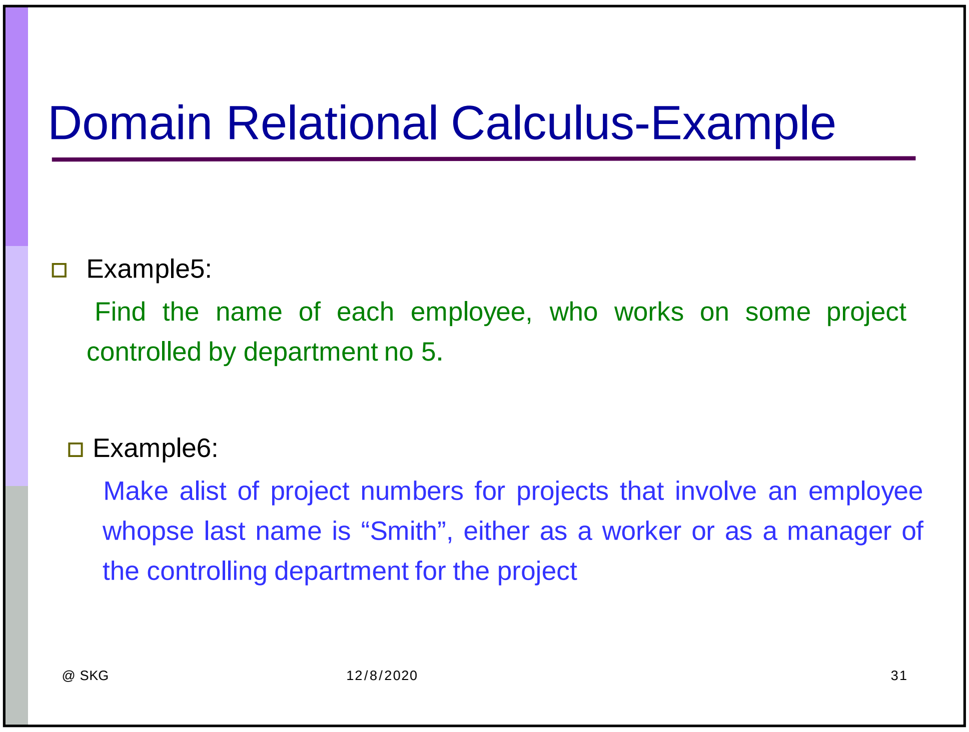
Domain Relational Calculus-Example

- **Example 4:** For every project located in “Stafford“, list the project number, controlling department number, department manager’s last name , birth date and address

Ans :

{ikrtu| (∃j)(∃m)(∃n)(∃s)(employee (qrstuvw) Λ project(hijk) Λ department (lmn) Λ j=“Stafford” Λ k=m Λ n=s)};

```
SELECT pnum, p.dnum, lname, bdate address FROM employee, project
p, department d WHERE p.dnum =d.dnum AND mgr_eid =empid ;
```



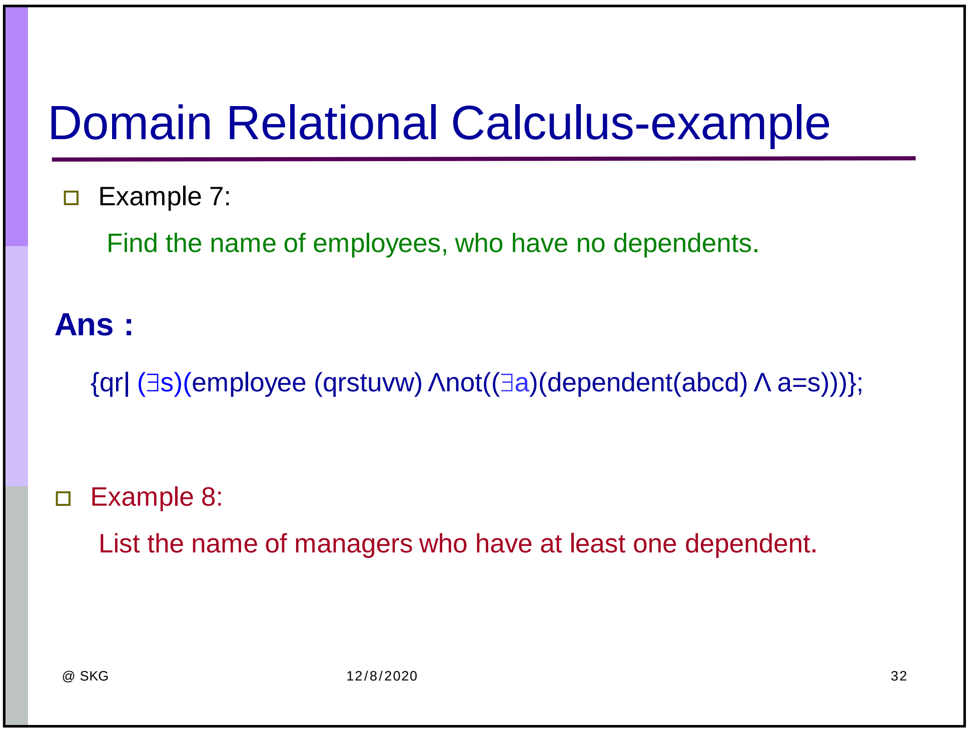
Domain Relational Calculus-Example

- Example5:

  Find the name of each employee, who works on some project controlled by department no 5.

- Example6:

  Make alist of project numbers for projects that involve an employee whopse last name is “Smith”, either as a worker or as a manager of the controlling department for the project



Domain Relational Calculus-example

- Example 7:

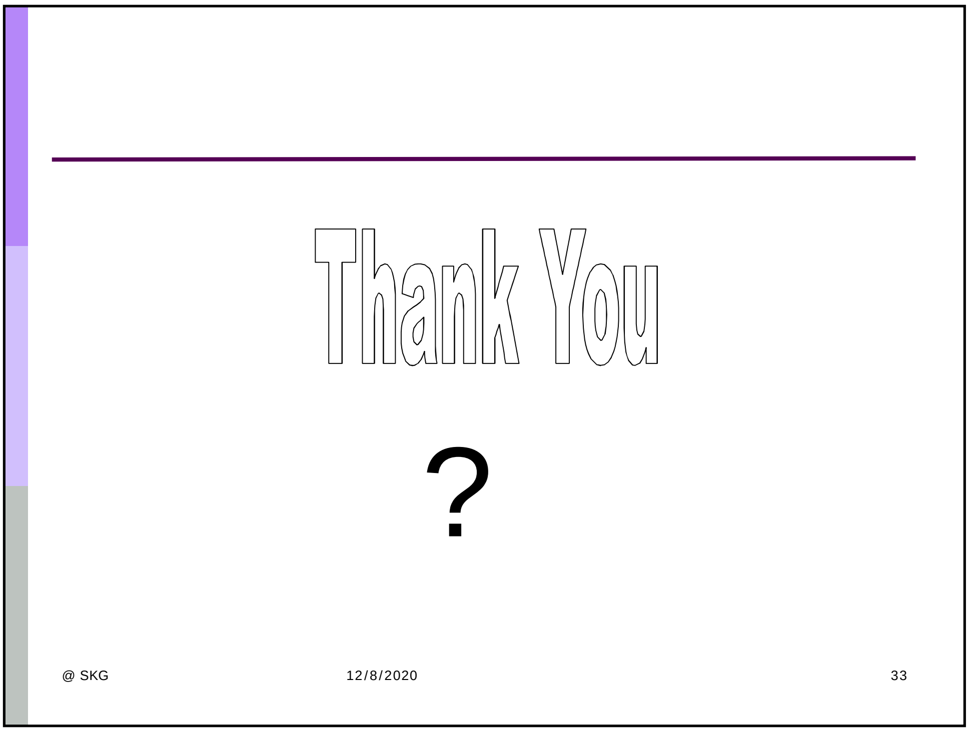
  Find the name of employees, who have no dependents.

Ans :

{qr| (∃s)(employee (qrstuvw) Λnot((∃a)(dependent(abcd) Λ a=s)))};

- Example 8:

  List the name of managers who have at least one dependent.



Thank You

?

@ SKG 12/8/2020 33