CS 220
Relational Calculus

START RECORDING

Attendance Quiz: Relational Algebra

- **Name** each of these relational algebra operations and **briefly describe what they do**: σ, π, ∪, ∩, ρ, –, ÷, ×, ⋈, *
- Use these operations to form these queries:
  1. Retrieve the names of all employees
  2. Retrieve the name of the employee with SSN = 123456789
  3. Retrieve the names of the employees with SSN = 123456789 or SSN = 234567891
  4. Retrieve the name of each manager and the name of the department they manage
  5. Retrieve the names of each department with an office in Houston
  6. Retrieve the names and SSNs of the employees making more than $71,000

Employee

Full_Name	SSN	Salary
John Smith	123456789	70000
Jane Smith	234567891	71000
Franklin Wong	345678912	72000

Department

Name	ID	Mgr_SSN
Research	1	345678912
Administration	2	234567891

D_Locations

D_ID	Location
1	Houston
1	Boston
2	Boston

Finish Relational Algebra Slides

Today you will learn…

- How to retrieve information from a relational schema using a declarative language
  - In contrast, the relational algebra is imperative

Relational Query Languages

- Query = “retrieval program”
- Language examples:
  - Theoretical:
    1. Relational Algebra
    2. Relational Calculus
       a. tuple relational calculus (TRC)
       b. domain relational calculus (DRC)
  - Practical
    1. SQL (SEQUEL from System R)
    2. QUEL (Ingres)
    3. Datalog (Prolog-like)
- Theoretical QL’s:
  - give semantics to practical QL’s
  - key to understand query optimization in relational DBMSs

Chapter 8 Outline

- Unary Relational Operations: SELECT and PROJECT
- Relational Algebra Operations from Set Theory
- Binary Relational Operations: JOIN and DIVISION
- Additional Relational Operations
- Examples of Queries in Relational Algebra
- **The Tuple Relational Calculus**
- **The Domain Relational Calculus**

The Tuple Relational Calculus

- Relational Calculus
  - Specify **what you want**, not how to get it (i.e, declarative, not procedural)
- Relational algebra
  - Specify how to get the information you want (i.e, procedural)
- However…
  - Any retrieval that can be specified in basic **relational algebra** can also be specified in **relational calculus**!
  - We are getting closer to SQL

Tuple Variables and Range Relations

$$\{t \mid \text{COND}(t)\}$$

- **Tuple variable:** *t*
- **Satisfy:** COND(*t*)
- Specify:
    - **Range relation** *R* of *t*
        - *What tables are you interested in?*
    - Select particular combinations of tuples
        - *What filtering rules do you want to apply?*
    - Set of attributes to be retrieved (**requested attributes**)
        - *What attributes are you interested in?*

Expressions and Formulas in Tuple Relational Calculus

- General expression of tuple relational calculus is of the form:

$$\{t_1.A_j, t_2.A_k, \ldots, t_n.A_m \mid \text{COND}(t_1, t_2, \ldots, t_n, t_{n+1}, t_{n+2}, \ldots, t_{n+m})\}$$

- **Truth value** of an atom
  - Evaluates to either TRUE or FALSE for a specific combination of tuples
- **Formula** (Boolean condition)
  - Made up of one or more atoms connected via logical operators **AND**, **OR**, and **NOT**

Three Forms of Atoms

- $R(t_i)$
  - Range relation: used to show which table(s) tuples you are interested in
  - Example: EMPLOYEE(t)
- $t_i.A \{=, <, \leq, >, \geq, \neq\} c$
  - Filters the tuples
  - Example: t.first_name = “Smith”
- $t_i.A \{=, <, \leq, >, \geq, \neq\} t_j.B$
  - Filters the tuples
  - Example: t.first_name = t.last_name

COMPANY Database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

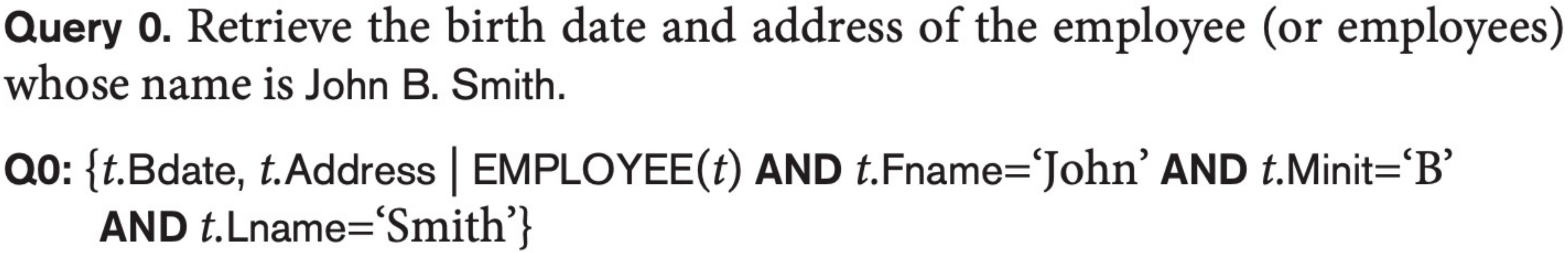
WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Simple Query

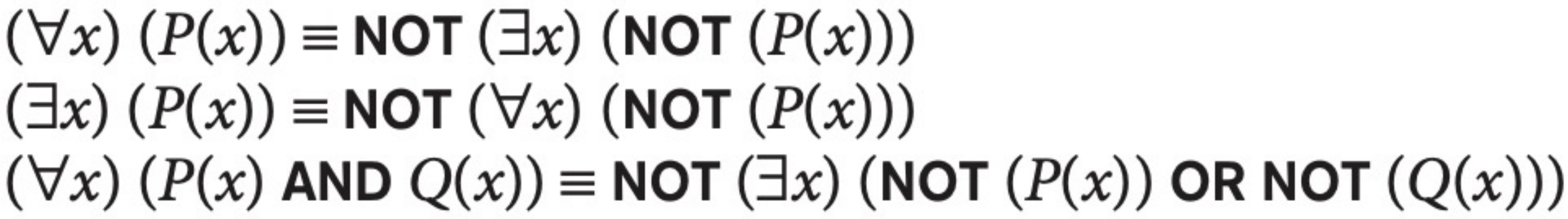


Query 0. Retrieve the birth date and address of the employee (or employees) whose name is John B. Smith.

Q0: {*t*.Bdate, *t*.Address | EMPLOYEE(*t*) **AND** *t*.Fname='John' **AND** *t*.Minit='B'
 AND *t*.Lname='Smith'}

Existential and Universal Quantifiers

- **Universal quantifier:** (∀t)(F)
  - TRUE when F is TRUE for **every** t
- **Existential quantifier** (∃t)(F)
  - TRUE when F is TRUE for **at least one** t
- Possible to transform between each other:
  - Negation (preceded by NOT)
  - AND and OR replace one another
  - A negated formula becomes unnegated, and vice-versa
  - For example:

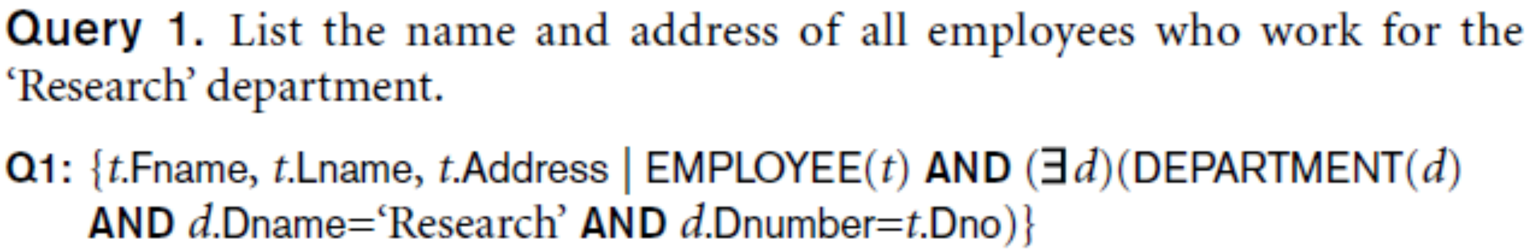


$$(\forall x)\ (P(x)) \equiv \textbf{NOT}\ (\exists x)\ (\textbf{NOT}\ (P(x)))$$

$$(\exists x)\ (P(x)) \equiv \textbf{NOT}\ (\forall x)\ (\textbf{NOT}\ (P(x)))$$

$$(\forall x)\ (P(x)\ \textbf{AND}\ Q(x)) \equiv \textbf{NOT}\ (\exists x)\ (\textbf{NOT}\ (P(x))\ \textbf{OR NOT}\ (Q(x)))$$

Queries Using Quantifiers

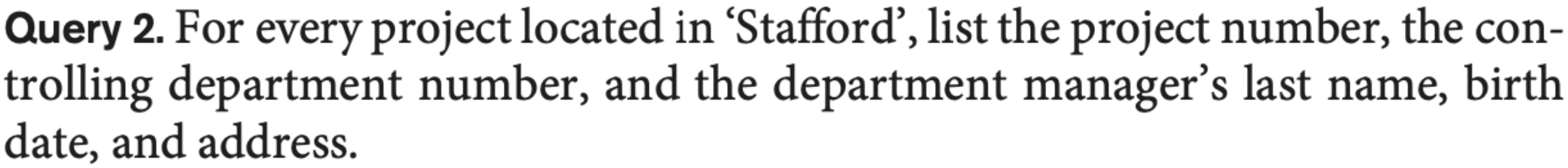


Query 1. List the name and address of all employees who work for the 'Research' department.

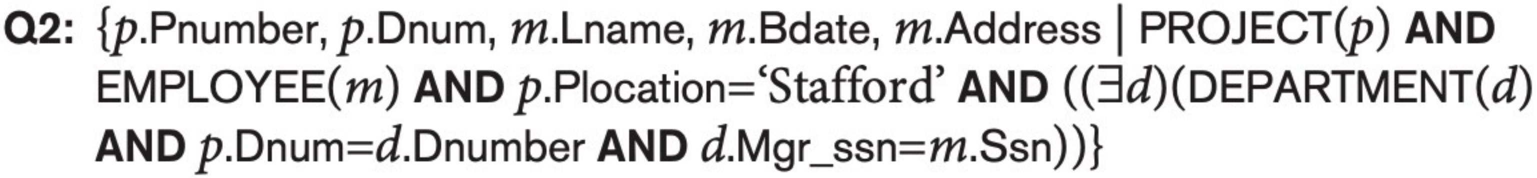
Q1: $\{t$.Fname, t.Lname, t.Address | EMPLOYEE(t) **AND** $(\exists d)$(DEPARTMENT(d) **AND** d.Dname='Research' **AND** d.Dnumber=t.Dno$)\}$

- Quantifiers are needed for multi-table queries
- Variable *d* is *bound* to the existential quantifier
  - Whereas *t* is *free* (i.e., not bound to a quantifier)
  - All free variables should appear to the left of the bar
  - Bound variables shouldn't appear to the left of the bar

Queries Using Quantifiers

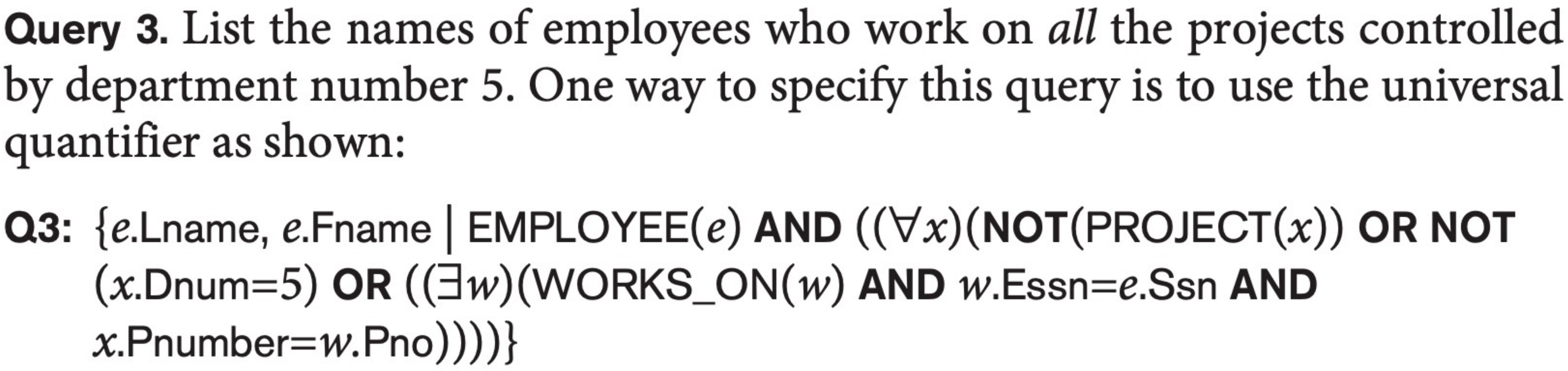


Query 2. For every project located in ‘Stafford’, list the project number, the controlling department number, and the department manager’s last name, birth date, and address.



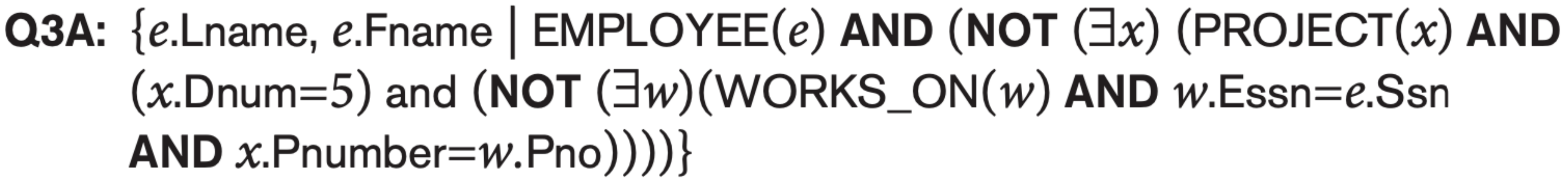
Q2: {*p*.Pnumber, *p*.Dnum, *m*.Lname, *m*.Bdate, *m*.Address | PROJECT(*p*) **AND**
EMPLOYEE(*m*) **AND** *p*.Plocation=‘Stafford’ **AND** ((∃*d*)(DEPARTMENT(*d*)
AND *p*.Dnum=*d*.Dnumber **AND** *d*.Mgr_ssn=*m*.Ssn))}

Queries Using Quantifiers



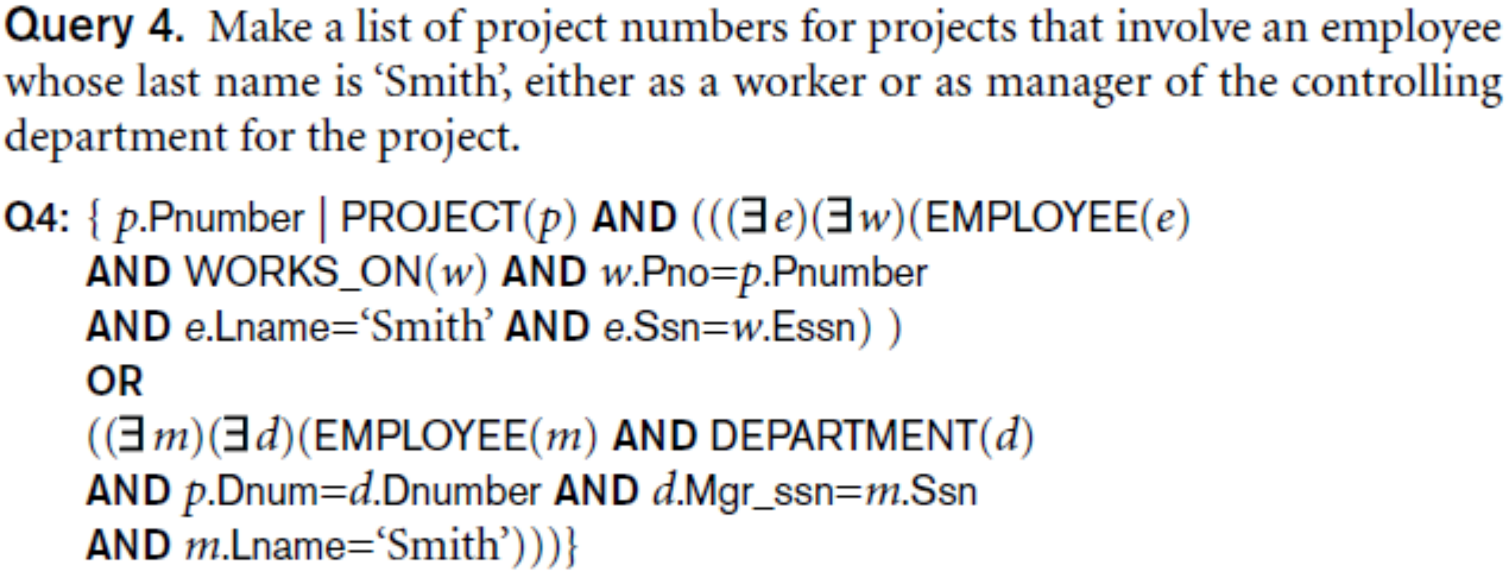
Query 3. List the names of employees who work on *all* the projects controlled by department number 5. One way to specify this query is to use the universal quantifier as shown:

Q3: {*e*.Lname, *e*.Fname | EMPLOYEE(*e*) **AND** ((∀*x*)(**NOT**(PROJECT(*x*)) **OR NOT** (*x*.Dnum=5) **OR** ((∃*w*)(WORKS_ON(*w*) **AND** *w*.Essn=*e*.Ssn **AND** *x*.Pnumber=*w*.Pno))))}



Q3A: {*e*.Lname, *e*.Fname | EMPLOYEE(*e*) **AND** (**NOT** (∃*x*) (PROJECT(*x*) **AND** (*x*.Dnum=5) and (**NOT** (∃*w*)(WORKS_ON(*w*) **AND** *w*.Essn=*e*.Ssn **AND** *x*.Pnumber=*w*.Pno))))}

Queries Using Quantifiers



Query 4. Make a list of project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as manager of the controlling department for the project.

Q4: { *p*.Pnumber | PROJECT(*p*) **AND** (((∃*e*)(∃*w*)(EMPLOYEE(*e*)

AND WORKS_ON(*w*) **AND** *w*.Pno=*p*.Pnumber

AND e.Lname='Smith' **AND** e.Ssn=*w*.Essn))

OR

((∃*m*)(∃*d*)(EMPLOYEE(*m*) **AND** DEPARTMENT(*d*)

AND *p*.Dnum=*d*.Dnumber **AND** *d*.Mgr_ssn=*m*.Ssn

AND *m*.Lname='Smith')))}

Queries Using Quantifiers

Query 6. List the names of employees who have no dependents.

Q6: $\{e.\text{Fname}, e.\text{Lname} \mid \text{EMPLOYEE}(e)$ **AND** (**NOT** $(\exists d)(\text{DEPENDENT}(d)$
AND $e.\text{Ssn}=d.\text{Essn}))\}$

Q6A: $\{e.\text{Fname}, e.\text{Lname} \mid \text{EMPLOYEE}(e)$ **AND** $((\forall d)($**NOT**$(\text{DEPENDENT}(d))$
OR NOT$(e.\text{Ssn}=d.\text{Essn})))\}$

Query 7. List the names of managers who have at least one dependent.

Q7: $\{e.\text{Fname}, e.\text{Lname} \mid \text{EMPLOYEE}(e)$ **AND** $((\exists d)(\exists \rho)(\text{DEPARTMENT}(d)$
AND $\text{DEPENDENT}(\rho)$ **AND** $e.\text{Ssn}=d.\text{Mgr\_ssn}$ **AND** $\rho.\text{Essn}=e.\text{Ssn}))\}$

Safe Expressions

- Guaranteed to yield a finite number of tuples as its result
  - Otherwise expression is called **unsafe**
- Which is safe?
  - {t | NOT EMPLOYEE(t)}
  - {t | EMPLOYEE(t)}

The Domain Relational Calculus

- Differs from tuple calculus in type of variables used in formulas
  - Variables range over single values from domains of attributes
- You won’t see it on the homework, exams, etc., but you should know it exists

- SQL: Based on **Tuple** Relational Calculus
- QBE (Query-By-Example): Based on **Domain** Relational Calculus

The Domain Relational Calculus (cont'd.)

Query 0. List the birth date and address of the employee whose name is 'John B. Smith'.

Q0: $\{u, v \mid (\exists q)\ (\exists r)\ (\exists s)\ (\exists t)\ (\exists w)\ (\exists x)\ (\exists y)\ (\exists z)$
$(\text{EMPLOYEE}(qrstuvwxyz)\ \textbf{AND}\ q=\text{'John'}\ \textbf{AND}\ r=\text{'B'}\ \textbf{AND}\ s=\text{'Smith'})\}$

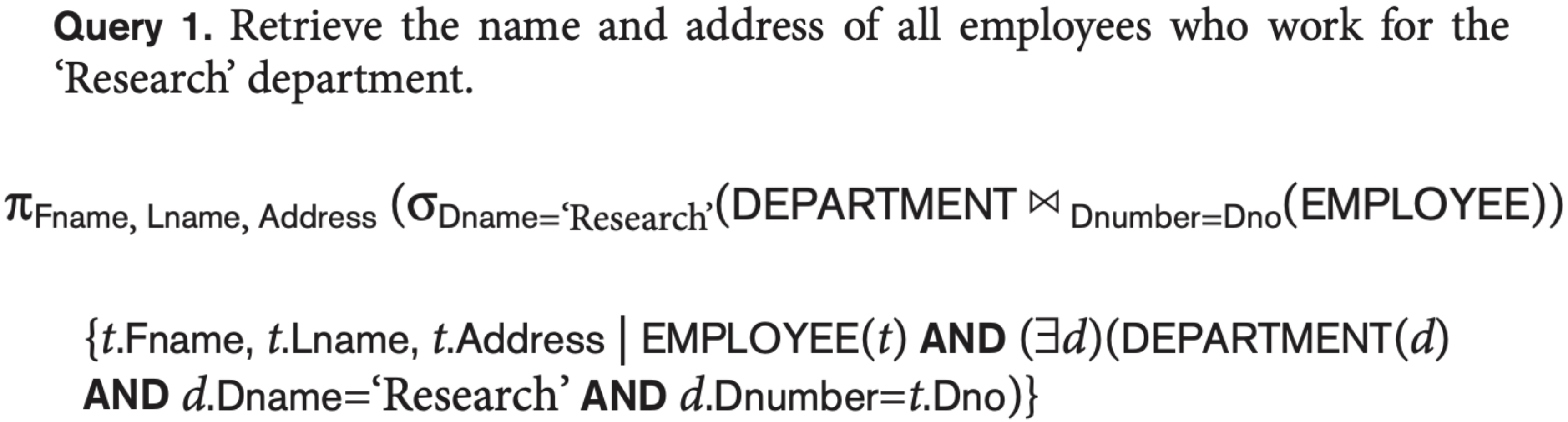
Or using QBE-style shorthand:

Q0A: $\{u, v \mid \text{EMPLOYEE}(\text{'John'}, \text{'B'}, \text{'Smith'}, t, u, v, w, x, y, z)\}$

We won't use the Domain Relational Calculus in this course!

Formal Languages for Relational Model

- Relational algebra
  - Imperative: specify **how** to get what you want
- Tuple/domain relational calculus
  - Declarative: specify **what** you want
- Possible to convert queries between the two



Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

$$\pi_{\text{Fname, Lname, Address}}\ (\sigma_{\text{Dname='Research'}}(\text{DEPARTMENT} \bowtie_{\text{Dnumber=Dno}}(\text{EMPLOYEE})))$$

$$\{t.\text{Fname},\ t.\text{Lname},\ t.\text{Address} \mid \text{EMPLOYEE}(t)\ \textbf{AND}\ (\exists d)(\text{DEPARTMENT}(d)\ \textbf{AND}\ d.\text{Dname='Research'}\ \textbf{AND}\ d.\text{Dnumber}=t.\text{Dno})\}$$