Database Management Systems

SQL (III)

Sample Database Schema

- Departments(did, dname, managerId)
- Emps(eid, firstname, lastname, salary, workdept, hireDate)
- Projects(pno, title, respEid, respDid)
- WorksOn(eid, pno, sdate, edate)

Existential Queries (I)

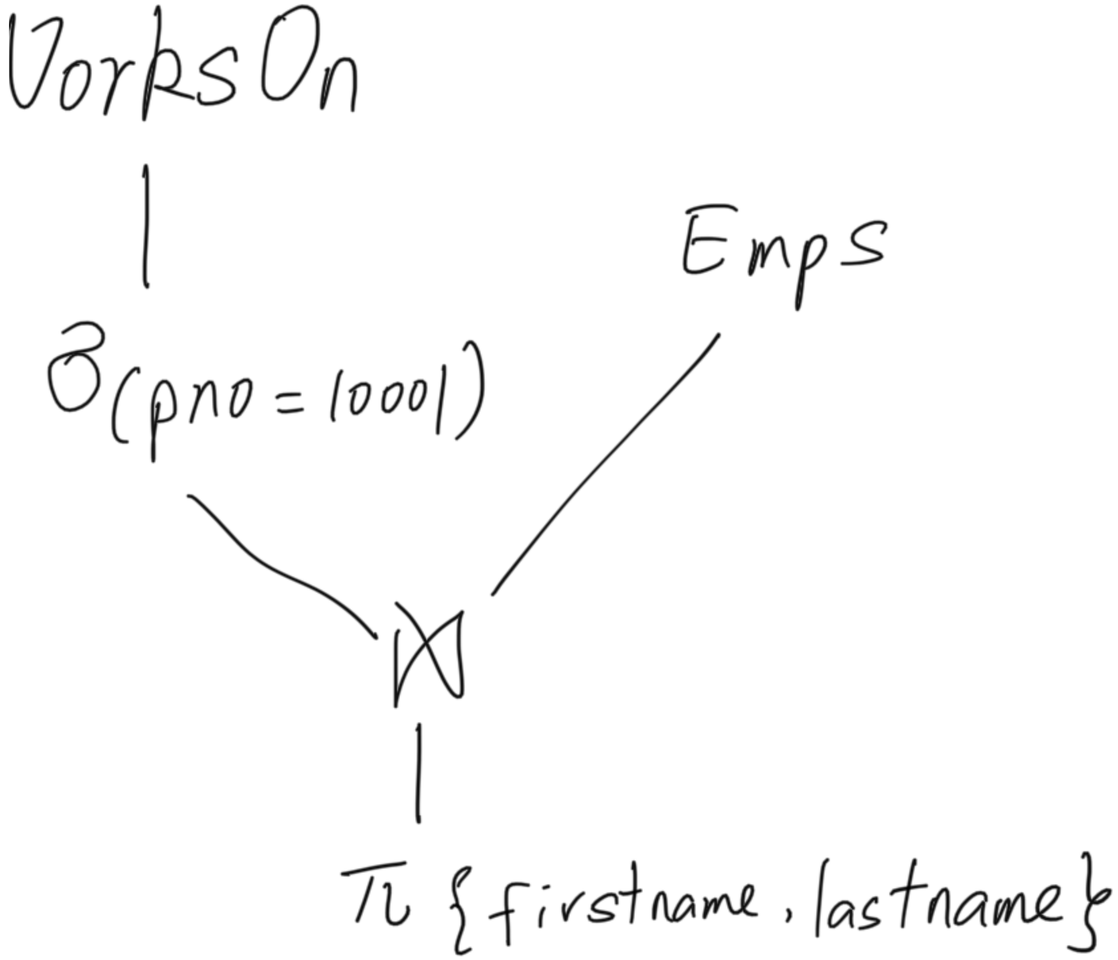
- List the name of each employee who worked on project 10001.

- Interpretation: if and only if there exists a record showing an employee ever worked on project 10001, then this employee’s name should be in the result.

- SQL:
  ```
  select distinct firstname, lastname
  from   Emps natural join WorksOn
  where pno = 10001;
  ```

Solution

- Datalog:
  Result(fn, ln) ::=
  WorksOn(eid, 10001, _, _)
  AND Emps(eid, fn, ln, _, _, _)



Existential Queries (II)

- List the name of each employee who NEVER worked on project 10001.
- Interpretation: if there exists a record showing an employee ever worked on project 10001, then this employee's name should NOT be in the result.

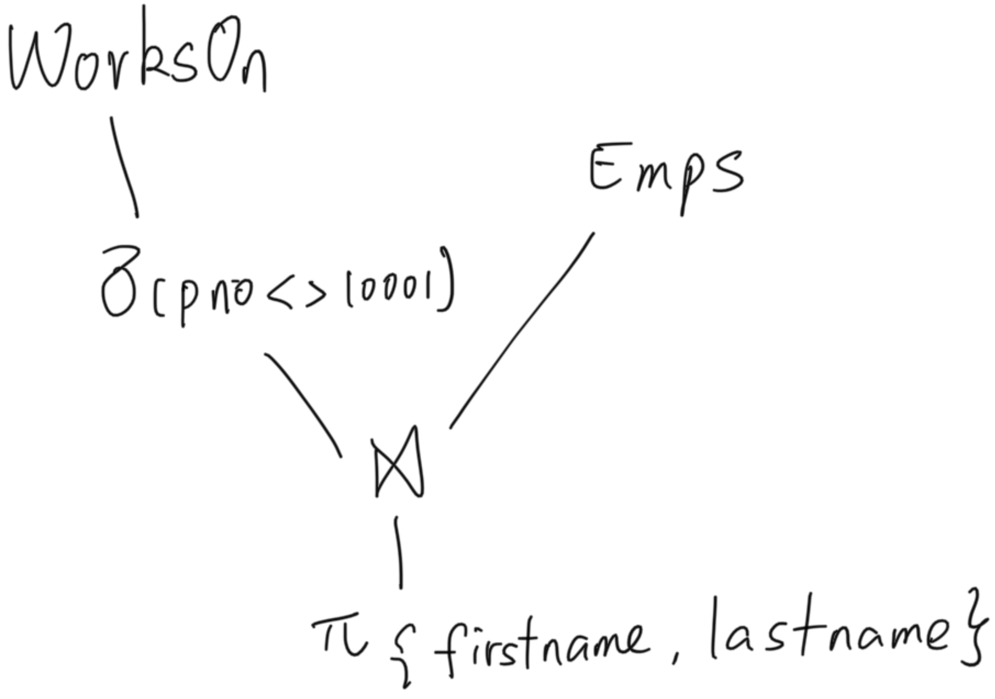
Wrong Query

- Datalog:

  ```
  Result(fn, ln) ::=
     WorksOn(eid, pno, _, _)
   AND Emps(eid, fn, ln, _, _, _)
   AND pno <> 10001
  ```

- Interpretation:
  List the name of each employee who worked on a project whose pno is not 10001. (One can work on Project 10001 and a project whose pno is not 10001.)

- Tuple level negation is not enough.

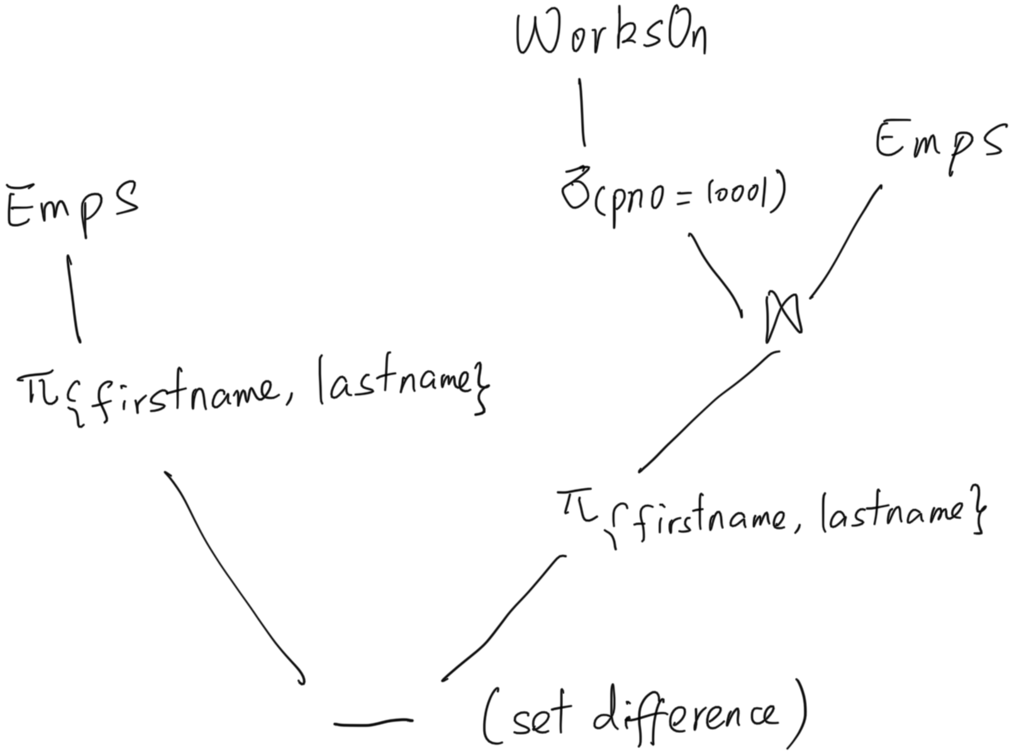


Solution

- Datalog:
  Result(fn, ln) ::=
  Emps(eid, fn, ln, _, _, _)
  AND
  NOT WorksOn(eid, 10001, _, _)

- Interpretation:
  For an employee, there doesn't exist such a record in WorksOn that shows this employee ever worked on project 10001

- There must be a table level negation.



Sub-Queries (I)

- predicate: EXISTS

  ```
  Select firstname, lastname
  From   Emps E
  Where not exists (Select *
                    From   WorksOn W
                    Where W.eid = E.eid
                        And W.pno = 10001);
  ```

- check membership operator: IN

  ```
  Select firstname, lastname
  From   Emps
  Where eid Not In (Select eid
                    From   WorksOn
                    Where pno = 10001);
  ```

Sub-Queries (II)

- Compare against a set of values

- List the name of the employee(s) who has the highest salary:

  ```
  Select firstname, lastname
  From   Emps
  Where salary >= All (select salary from Emps);
  ```

- List the name of the employee(s) whose salary is not the lowest:

  ```
  Select firstname, lastname
  From  Emps
  Where Salary > Some (select salary from Emps);
  ```

- Keyword Any? Is it equivalent to All or Some?

NULL values

- a special sentinel value to indicate a data item doesn’t exist in the database
- it usually means one of the following two things:
    - not applicable, such as the spouse_name column for an employee who is still single
    - we don’t know, such as the spouse_name column for an employee who is married but doesn’t want to disclose his/her partner’s name
- to determine whether a column is null, we must use the special operator “is null” or “is not null”
- the usual comparison operators against a null value will ALWAYS return false. “null = null” is false in database.

Outer Join

- In inner join, unmatched rows will be removed from the result
- There are three types of outer join:
    - Left join: unmatched rows from only the left table will be kept in the result
    - Right join: unmatched rows from only the right table will be kept in the result
    - Full outer join: unmatched rows from both the left and the right tables will be kept in the result
- Default “join” means inner join, default “full join” means “full outer join”.