Database Management Systems

Datalog (Domain/Tuple Calculus)

Datalog

- Datalog is a logical query language (“database logic”).
- Datalog query consists of if-then rules.
- Each of these rules expresses the idea that from certain combination of tuples in certain relations, we may infer that some other tuple must be in some other relation, or in the answer to a query.
- Datalog is a declarative language. (SQL is also declarative, while Relational Algebra is functional.)

Atoms

- An atom is a boolean function. There are two types of atoms in Datalog: Arithmetic and Predicate.

- Arithmetic atom: comparison between two arithmetic expressions

- Predicate (relational) atom: a predicate followed by its arguments
  - Each predicate takes a fixed number of arguments.

- predicate atom tests the membership:
  Lend(PA, PB, 1000) ==> Is it true that elements PA, PB, and 1000 form a tuple that is in the set Lend? ==> Is it true that person PA lend person PB $1000?

- When using Datalog to query database data, the predicates are database relations.
  Departments(101, ‘Public Relations’, 1001) ==> Is it true that the tuple <101, ‘Public Relations’, 1001> is in the table Departments in the database?

Rules and Queries

- A Datalog rule consists of:
  - a relational atom called the head
  - the symbol ::=, which often read as “if”
    - a body consisting of one or more atoms, called subgoals, which may be either relational or arithmetic. Subgoals are connected by AND, OR and NOT.
- A Datalog query is a collection of one or more rules.
- Datalog is relationally complete.

Extensional vs Intensional

- Predicate (Relational) atom can be either extensional or intensional.

- Extensional predicates: whose relations are stored in a database (they already exist).

- Intensional predicates: whose relations are computed by applying one or more Datalog rules (they exist because somebody’s intent).

Set Operations

- Union: use multiple rules with the same head
  Result(a, b) ::= rule1
  Result(a, b) ::= rule2

- Intersection: Two subgoals connected with AND
  Result(a, b) ::= T1(a, b) AND T2(a, b)

- Set difference: R AND NOT S
  Result(a, b) ::= T1(a, b) AND NOT T2(a, b)

Database Operations

- Projection:
  Result(fn, ln) ::= Emps(_, fn, ln, _, _, _)

- Selection:
  Result(fn, ln) ::= Emps(_, fn, ln, salary, _, _)
  AND salary > 50000

- Join: using same variables in multiple relational subgoals
  Result(fn, ln, dn) ::= Emps(_, fn, ln, _, did, _)
  AND Departments(did, dn, _)

Datalog Example

- For each employee who works in a department managed by Mary Smith, list his/her first and last name and the department's name.
  Result(fn, ln, dn) ::= Emps(eid, x, y, _, _, _)
  And Departments(did, dn, eid)
  And Emps(_, fn, ln, _, did, _)
  And x = 'Mary' And y = 'Smith'

- Equivalent one:
  Result(fn, ln, dn) ::= Emps(eid, 'Mary', 'Smith', _, _, _)
  And Departments(did, dn, eid)
  And Emps(_, fn, ln, _, did, _)

Safety (Unsafe Queries)

- Two potential unsafe factors:
  - top negation
    Result(dn) ::= NOT Departments(_, dn, _)
  - infinite domain involving arithmetic operations
    Result(x) ::= Emps(_, _, _, salary, _, _) AND x > salary
- Safety condition:
  Every variable that appears anywhere in the rule must appear in some non-negated, relational subgoal of the body.