- SQL — SELECT ... FROM ... WHERE ...
- Relational Algebra (RA) — $\sigma, \pi, \bowtie, \delta, \cup, \setminus$
- Relational Calculus (RC) — $\forall x F,\ \exists x F,\ F \wedge G,\ F \vee G, \neg F$
- **Datalog** — $\approx$ RC + Recursion

EXAMPLE: Given relations `employee(Emp, Salary, DeptNo)` and `dept(DeptNo, Mgr)`, find all (employee, manager) pairs:

- SQL:

```
SELECT Emp, Mgr
FROM employee, dept
WHERE employee.DeptNo = dept.DeptNo
```

- RA: $\pi_{\mathtt{Emp,Mgr}}(\mathtt{employee} \bowtie \mathtt{dept})$
- RC: $F(\mathtt{Emp,Mgr}) =$
  $\exists \mathtt{Salary,\ DeptNo} : (\mathtt{employee(Emp,\ Salary,\ DeptNo)} \wedge \mathtt{dept(DeptNo,Mgr)})$
- **Datalog**: `boss(Emp,Mgr) ← employee(Emp, Salary, DeptNo), dept(DeptNo,Mgr)`

- A **relational database** is given as a set of **facts**:

```
employee(john, 40000, toys). ...
employee(mary, 65000, cs). ...
...
dept(cs, mary). ...
```

- A **Datalog program** defines **views** by means of **rules** of the form **Head** ← **Body**:

```
boss(Emp,Mgr) ← employee(Emp, Salary, DeptNo), dept(DeptNo,Mgr)
highpaid(Emp) ← employee(Emp, Salary, _), Salary > 60000
```

- **EDB**: extensionally defined relations (facts): `employee/3, dept/2`
- **IDB**: intensionally (i.e., rule-) defined relations (views): `boss/2`
- A **query** is a view with a distinguished **answer**/n relation:
  `answer(Emp,Mgr) ← employee(Emp, Salary, DeptNo), dept(DeptNo,Mgr)`

Notation:

- **lowercase**: **relation names** (`employee/3, highpaid/1, ...`) and **constants** (aka data values: `john`, `toys`, `50000`, ...)
- **UPPERCASE/Capitalized**: **variables** (`Emp, X, ...`) (“_” means: don't care)

Relational operations have concise representations! Examples:

```
sel(X,Y) :- p(X,Y), X=a, not X=Y.   % SELECT some tuples from p(X,Y)

proj(X) :- p(X,Y).                  % PROJECT on the first argument

join(X,Y,Z) :- p(X,Y), q(Y,Z).      % JOIN p(A,B), q(C,D) s.t. B=C

prod(X,Y) :- p(X), q(Y).            % PRODUCT of p(X) and q(Y)

intersect(X) :- p(X), q(X).         % INTERSECTION of p(X), q(X)

diff(X) :- p(X), not q(X).          % SET-DIFFERENCE: p(X) \ q(X)

union(X) :- p(X).                   % UNION of p(X), ...
union(X) :- q(X).                   % ... and q(X)
```

Rules have a "logical reading" (i.e., rules are formulas):

$$\forall X \; (\, \mathtt{diff}(X) \;\leftarrow\; \mathtt{p}(X) \wedge \neg\, \mathtt{q}(X) \,).$$
$$\forall X \; (\, \mathtt{union}(X) \;\leftarrow\; \mathtt{p}(X) \vee \mathtt{q}(X) \,).$$

Relations can be defined (directly or indirectly) in terms of themselves ($\Rightarrow$ **recursive** relations/rules):

```
e(a,b). e(b,c).  ...          % FACTS (EDB-relation: e/2)

tc(X,Y) :- e(X,Y).            % RULES (IDB-relation: tc/2)
tc(X,Y) :- e(X,Z), tc(Z,Y).
```

- **Bottom-Up** Evaluation (**Fixpoint Semantics**): Apply rules **iteratively** (in so-called T_P-**rounds**) until a **fixpoint** is reached $(P := Facts \cup Rules)$:

$$\boxed{\begin{array}{l} I_0 := \emptyset \\ I_{n+1} := I_n \cup T_P(I_n) \end{array}}$$

The sequence $I_0 \subseteq I_1 \subseteq I_2 \ldots$ converges[1] to the **least fixpoint** $lfp(T_P)$ of T_P

T_P (**Immediate Consequences**) operator:

$$T_P(I) := \{\sigma(Head) \mid (Head \leftarrow Body) \in P,\; I \models \sigma(Body)\}$$

[1] proviso: P positive

```
e(a,b). e(b,c). e(c,d).         % FACTS (EDB-relation: e/2)

tc(X,Y) :- e(X,Y).              % RULES (IDB-relation: tc/2)
tc(X,Y) :- e(X,Z), tc(Z,Y).
```

n	I_n
0	$\emptyset$
1	$\{\mathbf{e(a,b)}, \mathbf{e(b,c)}, \mathbf{e(c,d)}\}$
2	$\{e(a,b), e(b,c), e(c,d)\} \cup \{\mathbf{tc(a,b)}, \mathbf{tc(b,c)}, \mathbf{tc(c,d)}\}$
3	$\{e(a,b), e(b,c), e(c,d)\} \cup \{tc(a,b), tc(b,c), tc(c,d), \mathbf{tc(a,c)}, \mathbf{tc(b,d)}\}$
4	$\{e(a,b), e(b,c), e(c,d)\} \cup \{tc(a,b), tc(b,c), tc(c,d), tc(a,c), tc(b,d), \mathbf{tc(a,d)}\}$

$\Rightarrow$ if the longest path in e/2 has length n, then $O(n)$ rounds are needed!

(Exercise: how about the following rule? `tc(X,Y)` $\leftarrow$ `tc(X,Z), tc(Z,Y).`)