1) Explain the term „range restricted clause“ with respect to the completion.

In a range restricted clause every variable in a clause occurs in at least one non negative body literal. That means, we have no existentials for variables in negative literals. As a result, when searching for a model we can restrict the search space by starting from a definite subprogram by removing all negative literals. Taking into account these literals later leads to the actual model.

Outlook: Depending on how we treat the negative literals we can come up with a slightly different semantics, the stable model semantics.

2) Prove that relational algebra has the same expressive power as non-recursive Datalog.

We can show that every relational operator can be mapped to an equivalent datalog expression and vice versa.

3) Are the following clauses range restricted?

1. \( p(x, a) :- q(x, y, a), \neg r(y, a). \) yes
2. \( p(x, a) :- q(a, a, x), \neg r(x, y). \) no
3. \( p(x, x) :- r(x, y). \) yes
4. \( p(x,y) :- y = a, \neg r(x,y). \) yes
5. \( p(x,y) :- y = a, \neg r(x,y), q(x,y,a). \) yes
4) Model the following scenario using Datalog and draw a dependency graph:

There are different types of events: lectures, tutorials and seminars. Every event takes place in a certain room and has a start, an end time and a title. An event may have other events as prerequisites. Professors, students and research assistants are persons. Every person has a name and an e-mail address. Additionally, every professor has a working group and every research assistant works in a working group. Events are given by professors or research assistants. Every tutorial belongs to a lecture. Students attend events.

event(X) :- lecture(X).
event(X) :- tutorial(X).
event(X) :- seminar(X).
event(X) :- prerequisite(X, Y).
event(Y) :- prerequisite(X, Y).
person(X) :- professor(X).
person(X) :- student(X).
person(X) :- researchAssistant(X).
professor(X) :- hasWorkingGroup(X,Y).
workingGroup(Y) :- hasWorkingGroup(Y).
researchAssistant(X) :- worksIn(X,Y).
workingGroup(Y) :- worksIn(X,Y).
tutorial(X) :- belongsTo(X, Y).
lecture(Y) :- belongsTo(X, Y).
student(X) :- attends (X, Y).
event(Y) :- attends(X, Y).
inconsistent :- event(X), not eventDetail(X, _, _, _, _).
inconsistent :- person(X), not personDetails(X, _, _).
inconsistent :- professor(X), not hasWorkingGroup(X, _).
inconsistent :- researchAssistant(X), not worksIn(X, _).
inconsistent :- tutorial(X), not belongsTo(X, _).
consistent :- not inconsistent.

Is the resulting program in non-recursive Datalog?

Yes.
5) Compute the least Herbrand models of the following programs using the immediate consequence operator $T_P$.

1. $p(a) :- p(x), q(x)$.
$p(f(x)) :- p(x)$.
$q(b)$.
$q(f(x)) :- q(x)$.

$I_0 = {}$
$I_1 = \{q(b)\}$
$I_2 = \{q(b), q(f(b))\}$
$I_3 = \{q(b), q(f(b)), q(f(f(b)))\}$
$I_\omega = \{q(b), q(f(b)), ..., q(f^n(b))\}$

2. $p(a)$.
$p(b)$.
$q(c)$.
$q(x) :- not r(x,b)$.
$r(x,x) :- p(x)$.

Not definite, not stratifiable. For the maximal definite subprogram we have:

$I_0 = {}$
$I_1 = \{p(a), p(b), q(c)\}$
$I_2 = \{p(a), p(b), q(c), r(a,a), r(b,b)\}$

3. $p_1(f(x)) :- p_1(x)$.
$p_2(a) :- p_1(x)$.
$p_2(f(x)) :- p_2(x)$.
$p_3(a) :- p_2(x)$.
$p_3(f(x)) :- p_3(x)$.
$p_4(a) :- p_3(x)$.
$p_4(f(x)) :- p_4(x)$.
$p_5(a) :- p_4(x)$.

$I_0 = {}$