**9/30/2019**

Concepts to be modeled:

1. Entity: e.g. student, class (likely noun)
2. Relationship: e.g. teaches, advises, … (likely verb)
3. Attribute: e.g. FName, StuId, …
4. No need to be modeled.

<https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model>

ER have many notations.

F18 HW #3:

**Toy Employee System (TES)**  
  
This assignment models a drastically simplified component of TES for a company.

The company (Q: one company only?) has employee, each with a unique employee id.

Concepts:

1. Employee: entity
2. EmpId: attribute
   1. Unique: PK

The first name, last name, phone, and an optional email should be stored for each employee.

Concepts:

1. first name: property of an employee: attribute
2. last name: property of an employee: attribute
3. phone: property of an employee: attribute
4. email: property of an employee: attribute
   1. optional: allow null.

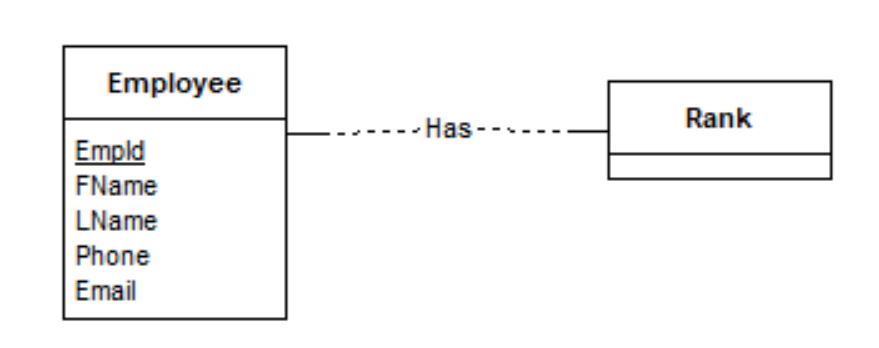
Every employee has a current rank. (noun-verb-noun)

1. Rank: (a) entity, (b) attribute
2. Current rank
3. Has: (a) relationship, (b) rank is an attribute of employee

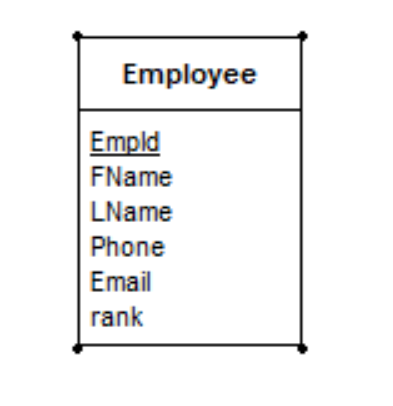
Entity versus attribute:

1. Entity: more complicated, more flexible. Exists by itself.
2. Attribute: more simple, value only.

(a)



(b)



These ranks are predefined. For example, employee X and Y may both be of the rank ‘Laboratory Supervisor II’, and Z may be of the rank ‘Office Manager I’. A rank may have a rank description.

Concept:

1. Rank description (desc): 2a, attribute of rank.
2. X, Y, Z: entity instance (object) of the entity type (class) Employee.
3. ‘Laboratory Supervisor II’: instance of Rank.

The start date of the current rank of every employee should be recorded.

Furthermore, TES also stores the rank history of the previous ranks of every employee, together with the start dates and end dates of these ranks. For example, the rank history of employee X may be:

* Laboratory Assistant I: 1/21/2015 to 1/20/2016.
* Laboratory Assistant II: 1/21/2016 to 1/20/2017.
* Laboratory Supervisor I: 1/21/2017 to 3/5/2018.

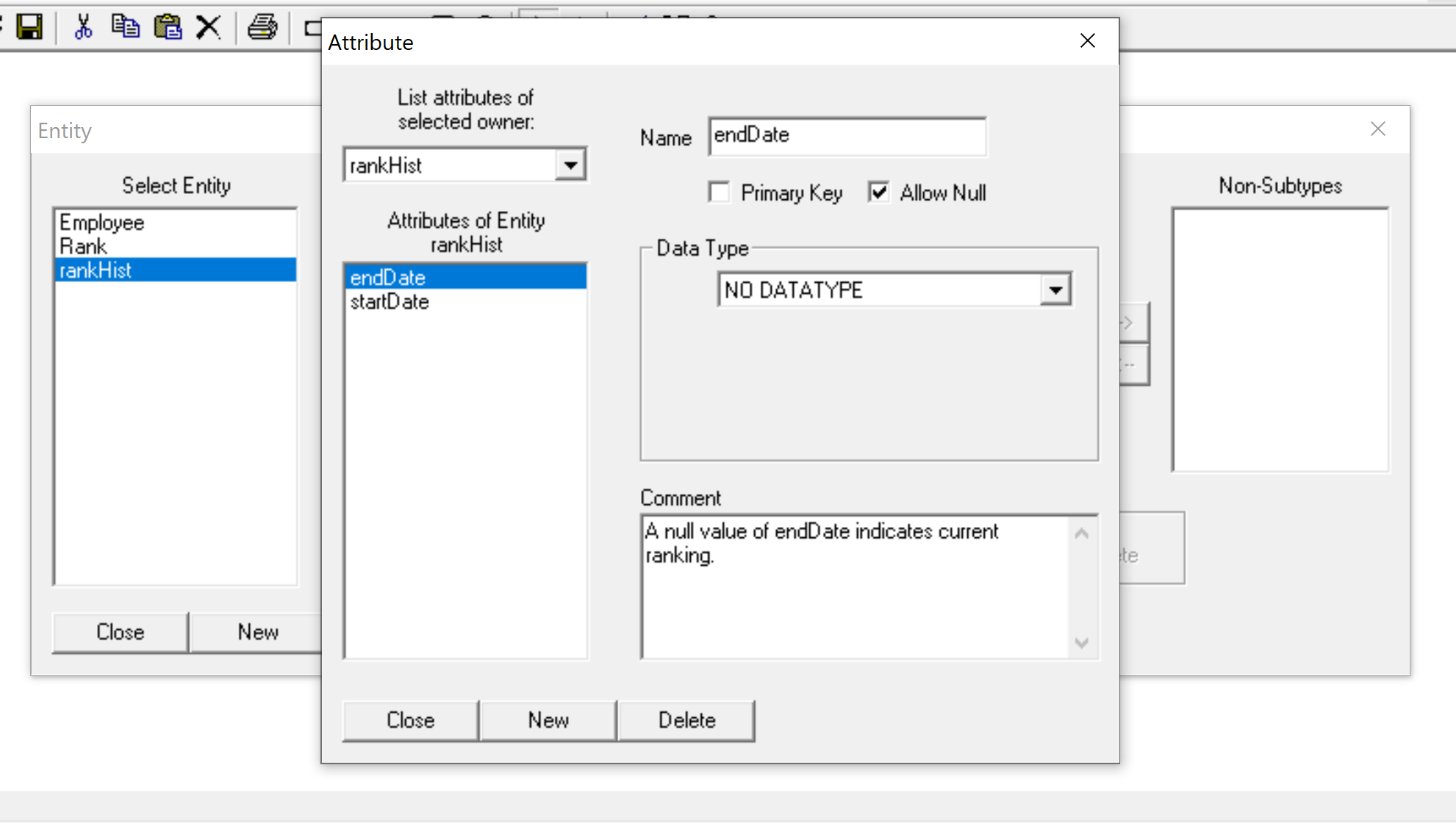
Concepts:

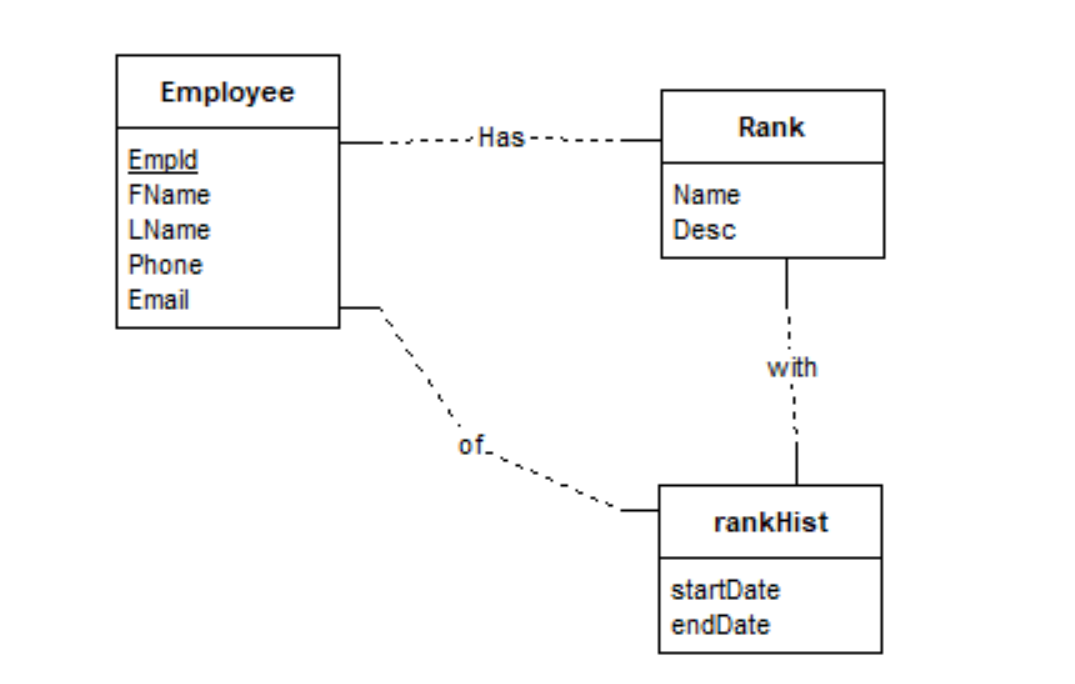
1. Current rank (of X): only one current rank: rankHIst: no EndDate
2. Previous ranks (of X): many: rankHist entity
3. Start date (of X as Laboratory Assistant I): attribute
   1. Property of X? No -> Not an attribute of entity Employee
4. End date (of X as Laboratory Assistant I): attribute

Modeling:

1. Rank history: entity: definition, a rank history is an employee having a specific rank with a start date and an end date (properties of a rank history)

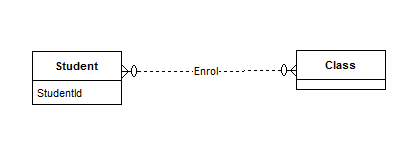
Her current rank may be ‘Laboratory Supervisor II’ since 3/6/2018.





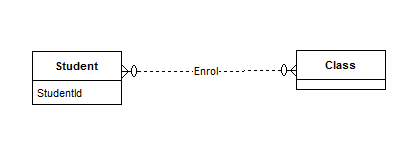
* Relationship cardinality (or *multiplicity*): number of entity instances to which another entity can map under the relationship.
* For binary relationship:
  1. one to one
  2. one to many
  3. many to many
* Furthermore, cardinality can be optional or mandatory.
  1. One can be (0..1) or (1..1)
  2. Many can be (0..\*) or (q..\*)

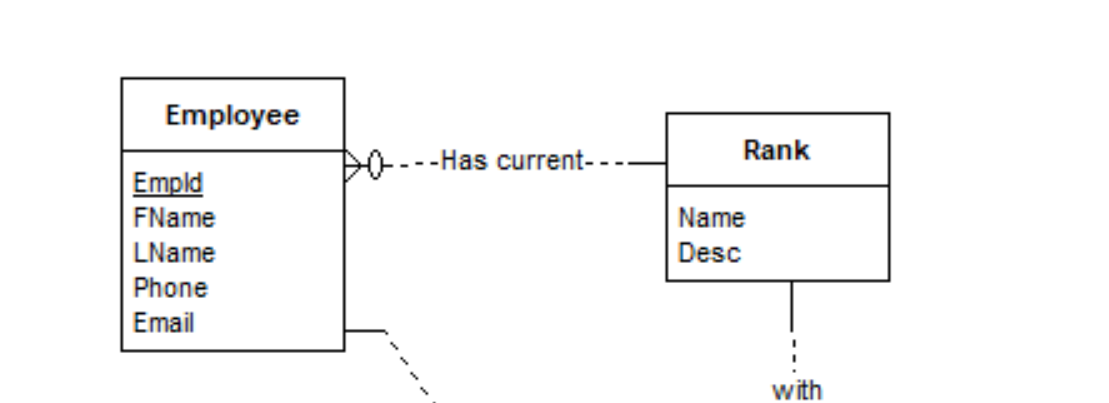
***Example:***



* Relationship cardinality (or *multiplicity*): number of entity instances to which another entity can map under the relationship.
* For binary relationship:
  1. one to one
  2. one to many
  3. many to many
* Furthermore, carindality can be optional or mandatory.
  1. One can be (0..1) or (1..1)
  2. Many can be (0..\*) or (q..\*)

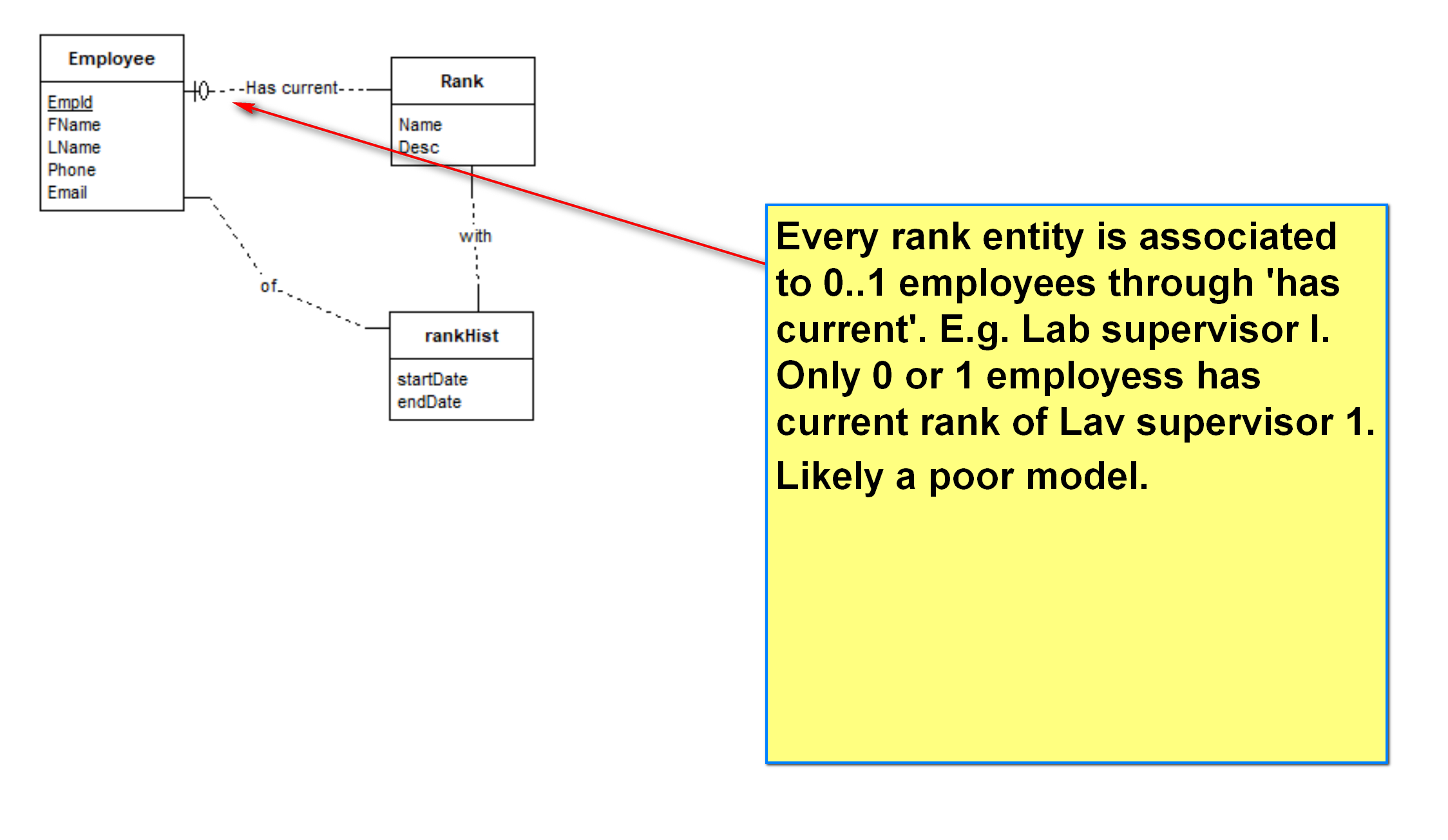
***Example:***

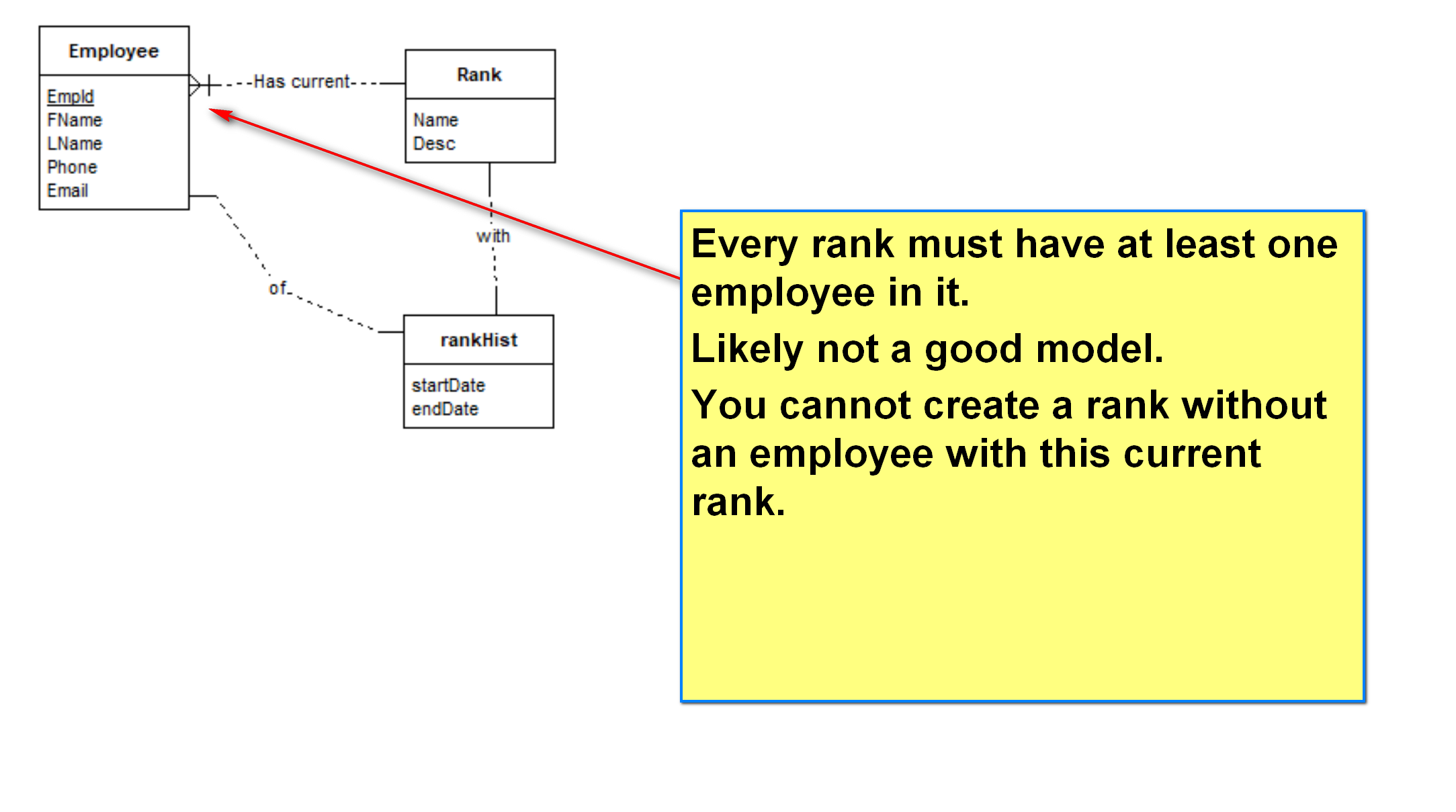


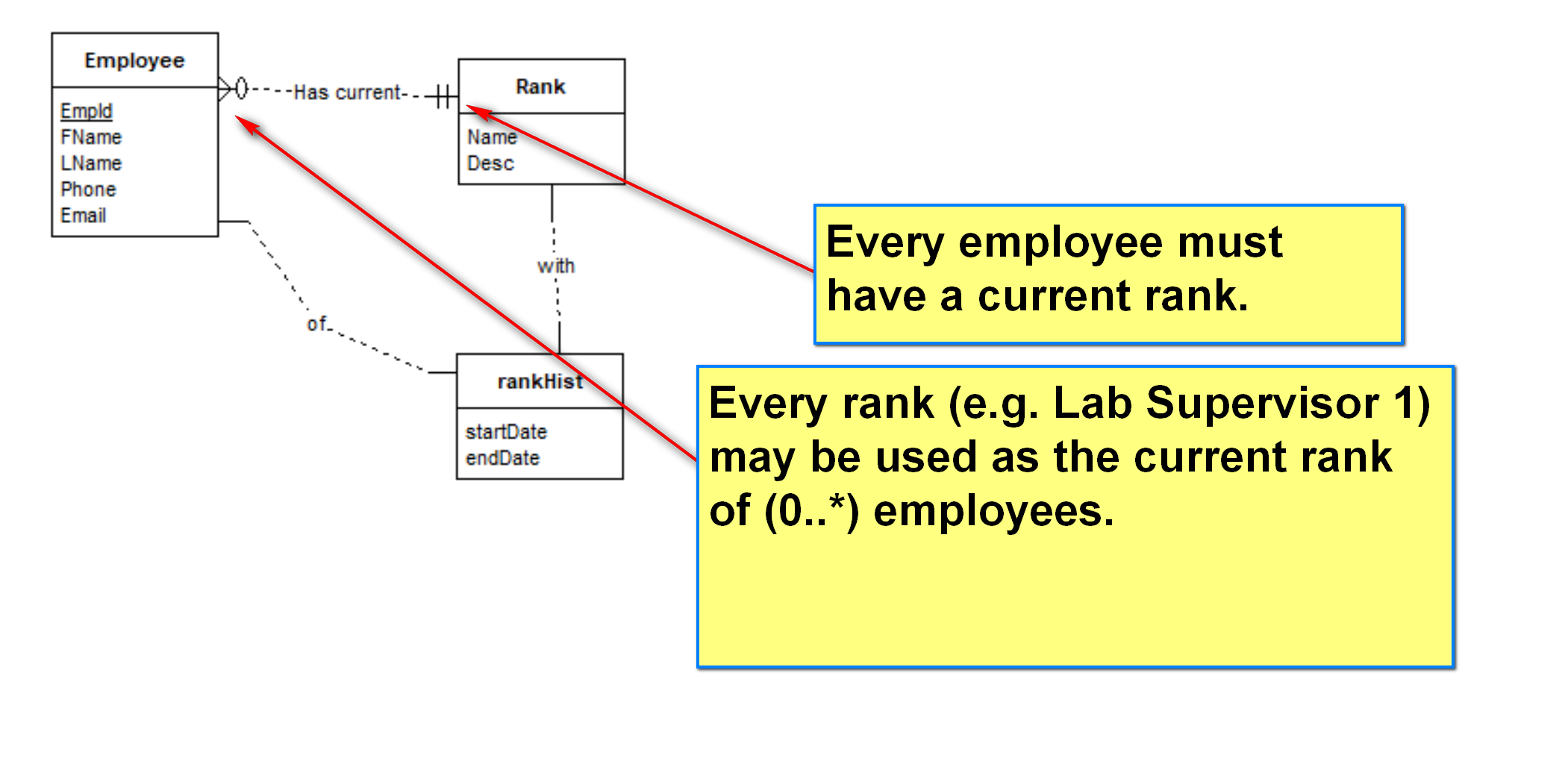
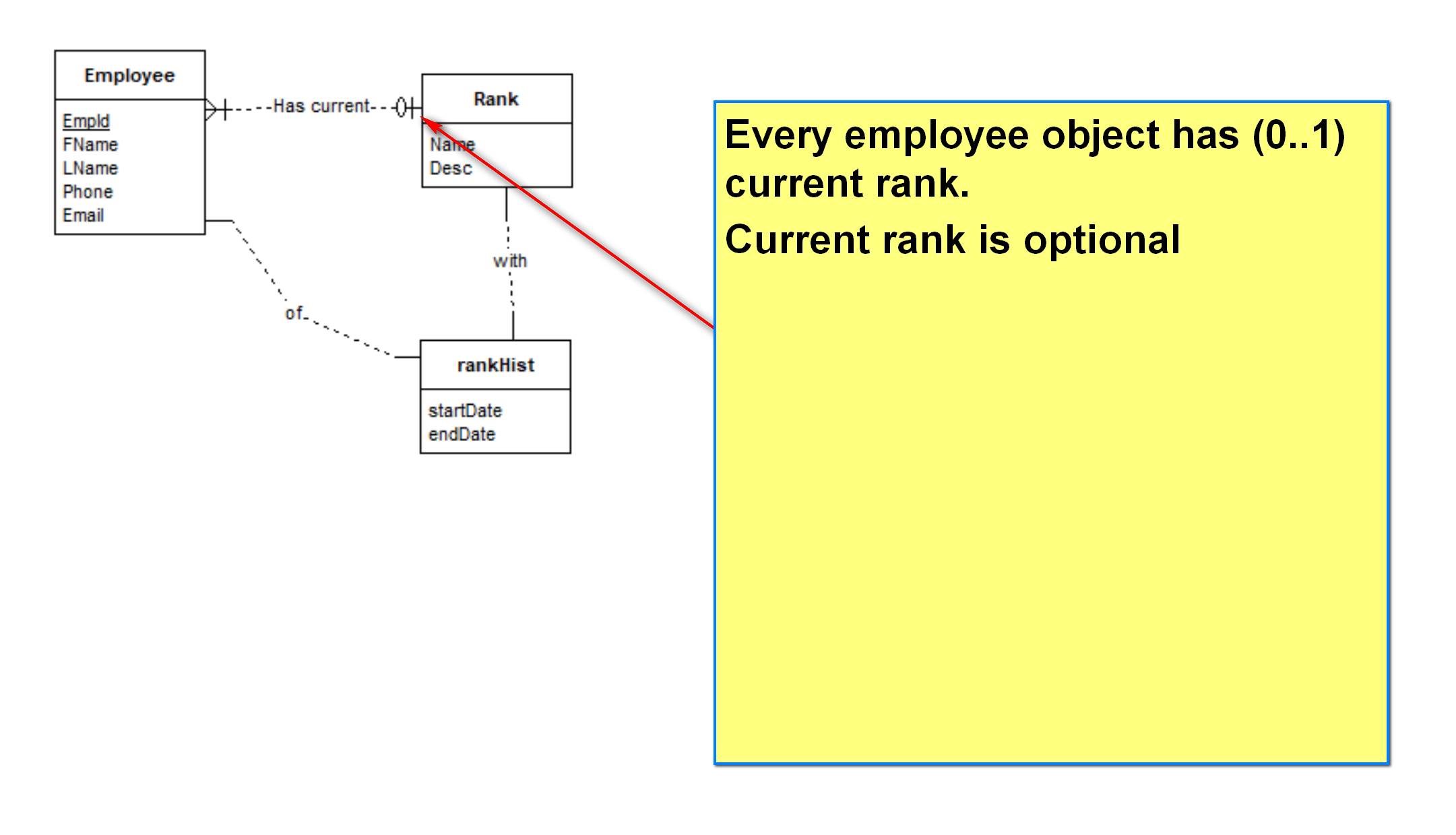


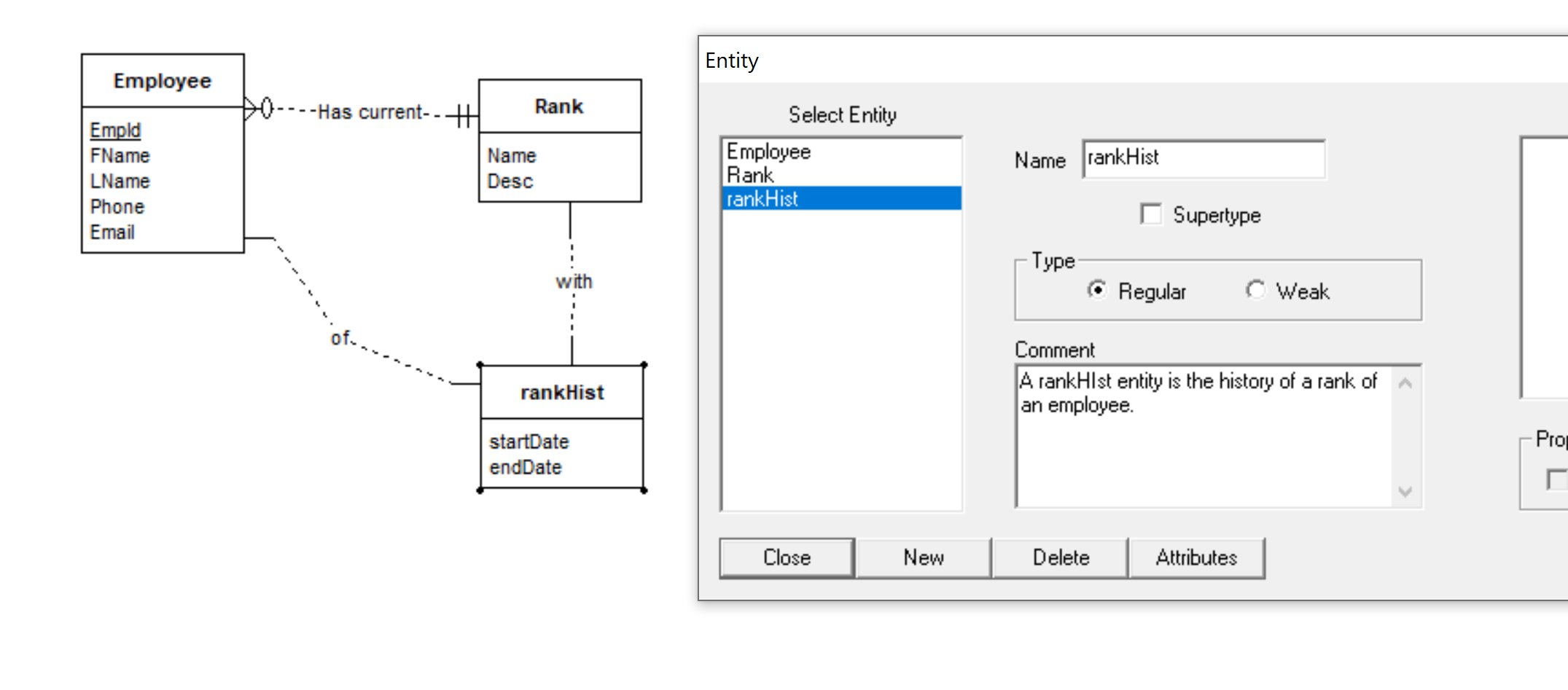
Every rank object is related to (0..\*) employees through the relationship ‘has current’:

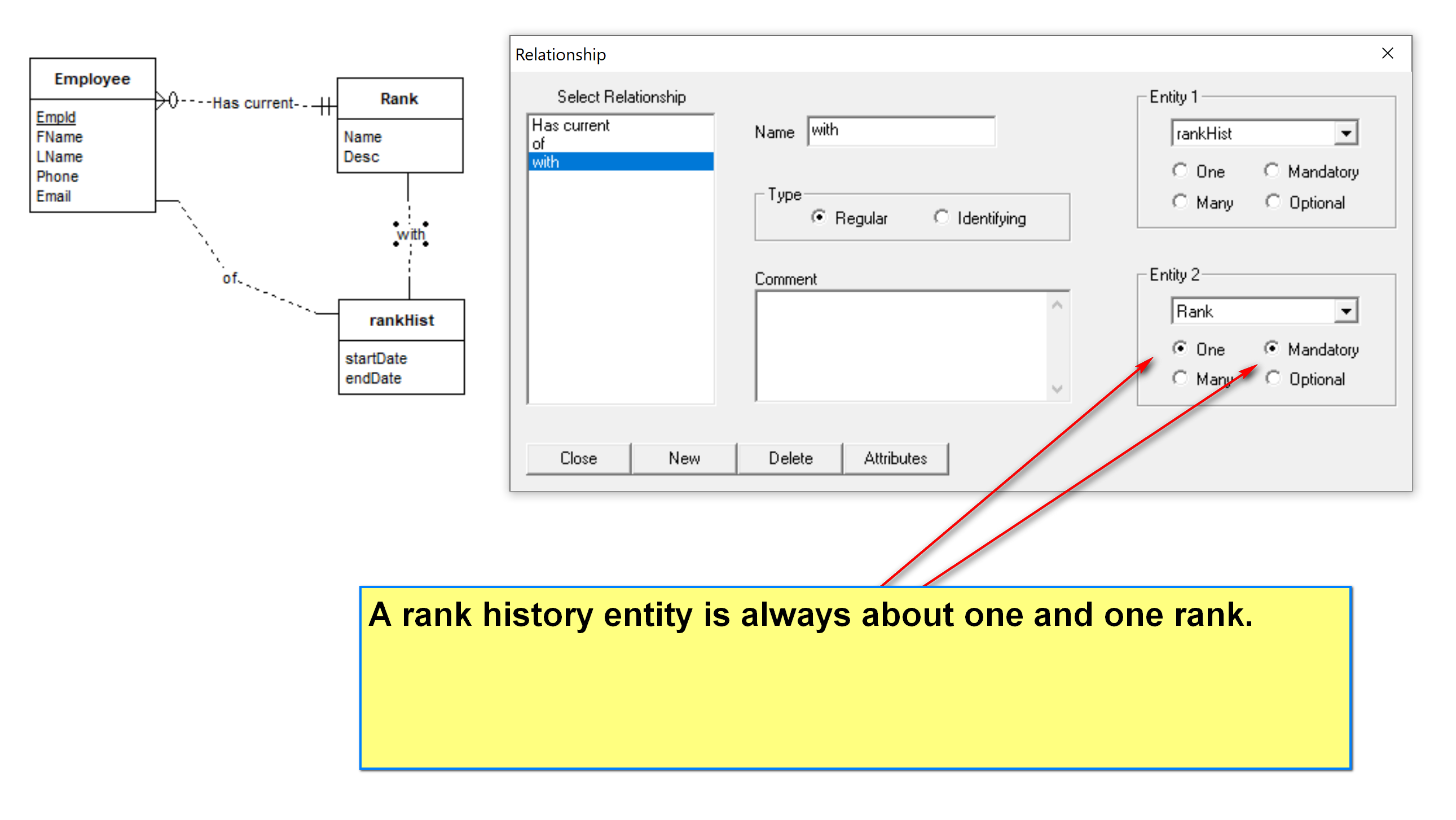
0 or many employees have the same current rank (e.g. lab supervisor I)

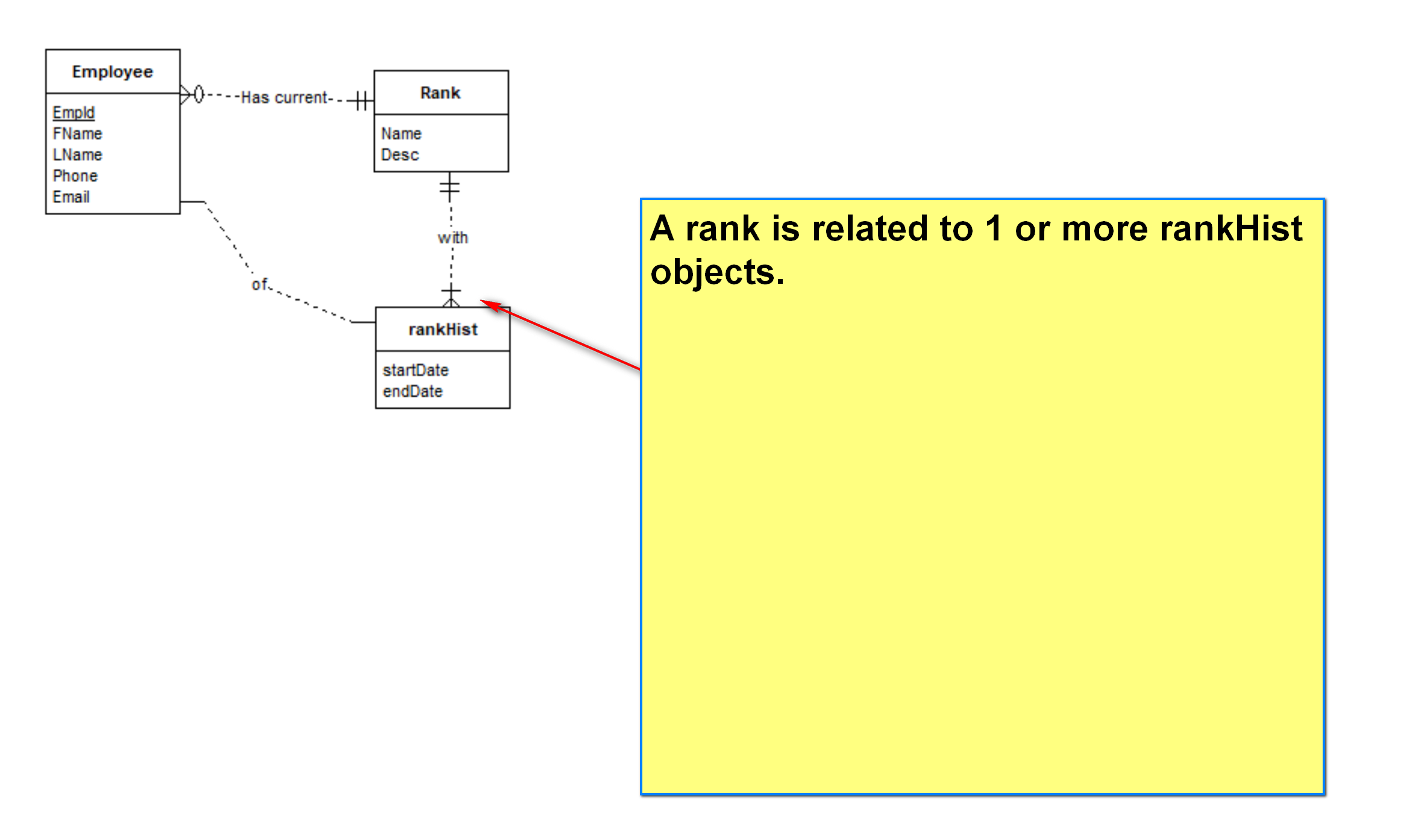


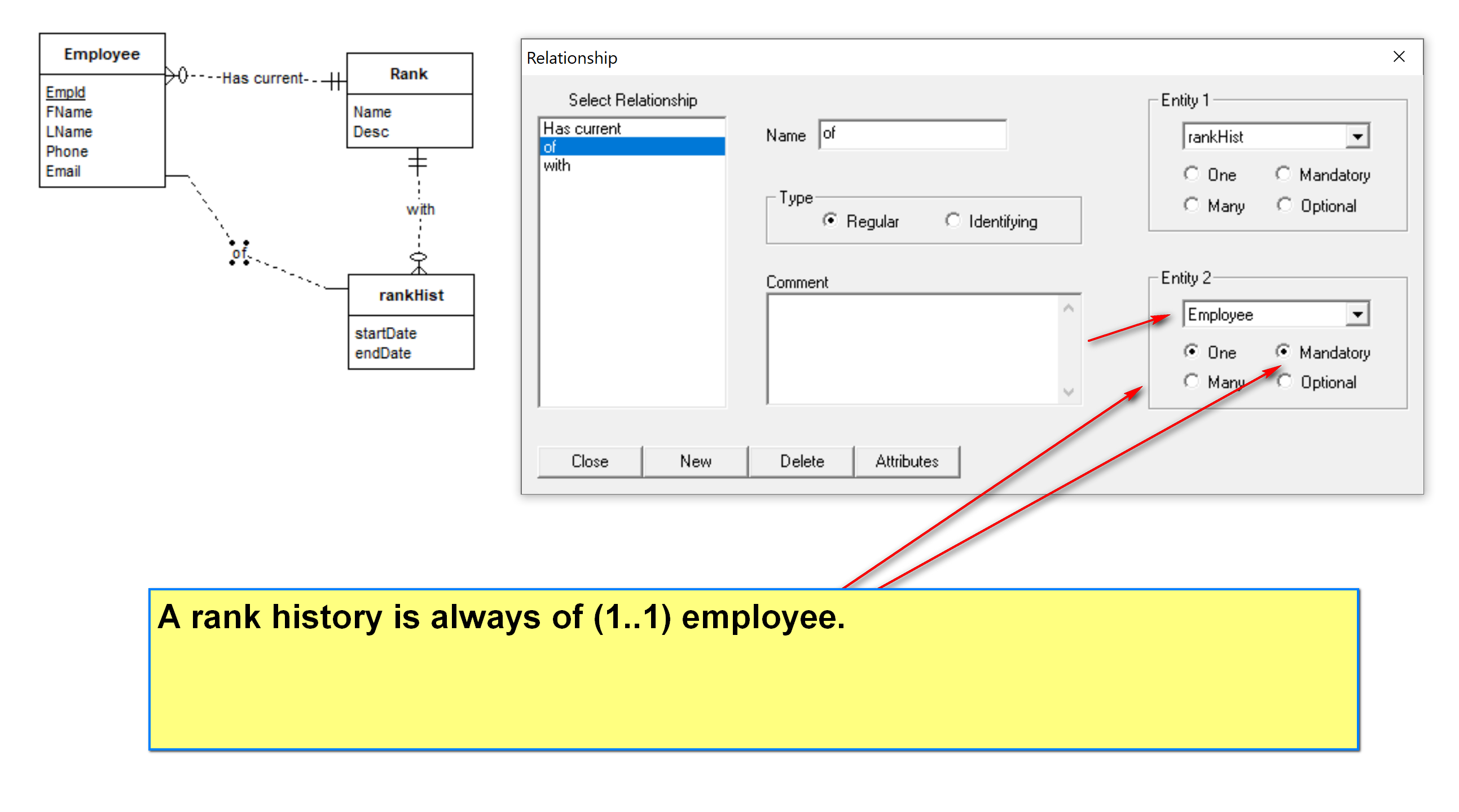












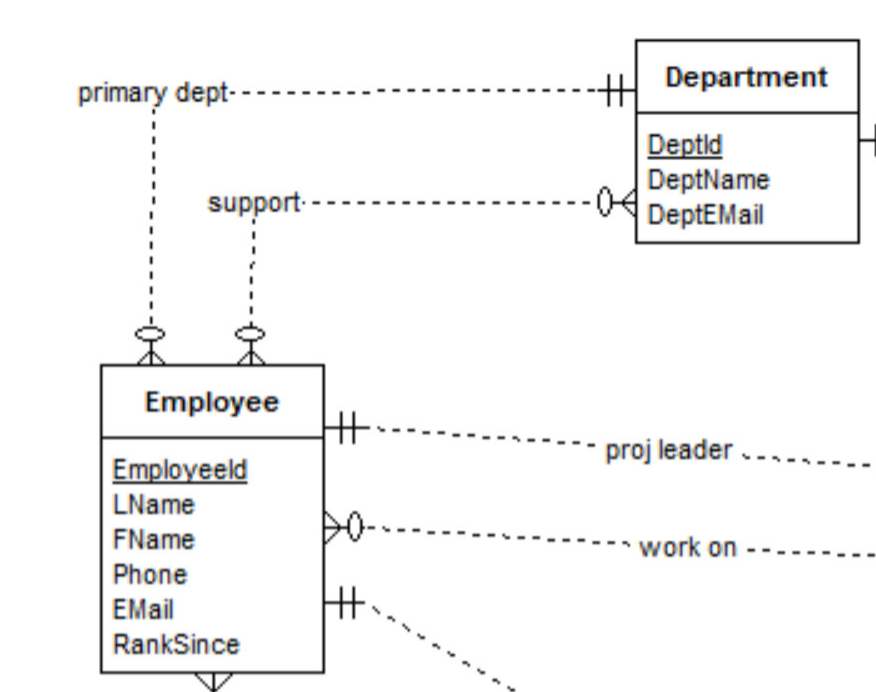
Each rank has a unique level that is used to determine the pay range. Several ranks may belong to the same level. For example, both ‘Laboratory Supervisor II’ and ‘Office Manager I’ may belong to the same level ‘Supervisor Level II’. A level may come with a description.

There are departments, with a unique DeptId. The department name and email address should also be stored. An employee is always assigned to a unique department as his primary department. He may support any number of additional department.

There are projects, each with a unique id, a name and an optional description. A project always has an employee as the project leader. It may have any number of additional employees working on it. A project always has a lead department.

Suggested solutions:





There are departments (entity), with a unique DeptId (attribute of Department: PK). The department name and email address (attributes of Department) should also be stored. An employee is always assigned to a unique (one and only one in the multiplicity) department as his primary department. (noun: employee entity) – verb (relationship):assigned to as primary – noun: department entity) (Assumption: a department may have 0..\* employee as its primary department.) He (employee object) may (therefore support 0..\* department) support any number of additional (relationship) department (noun: entity).