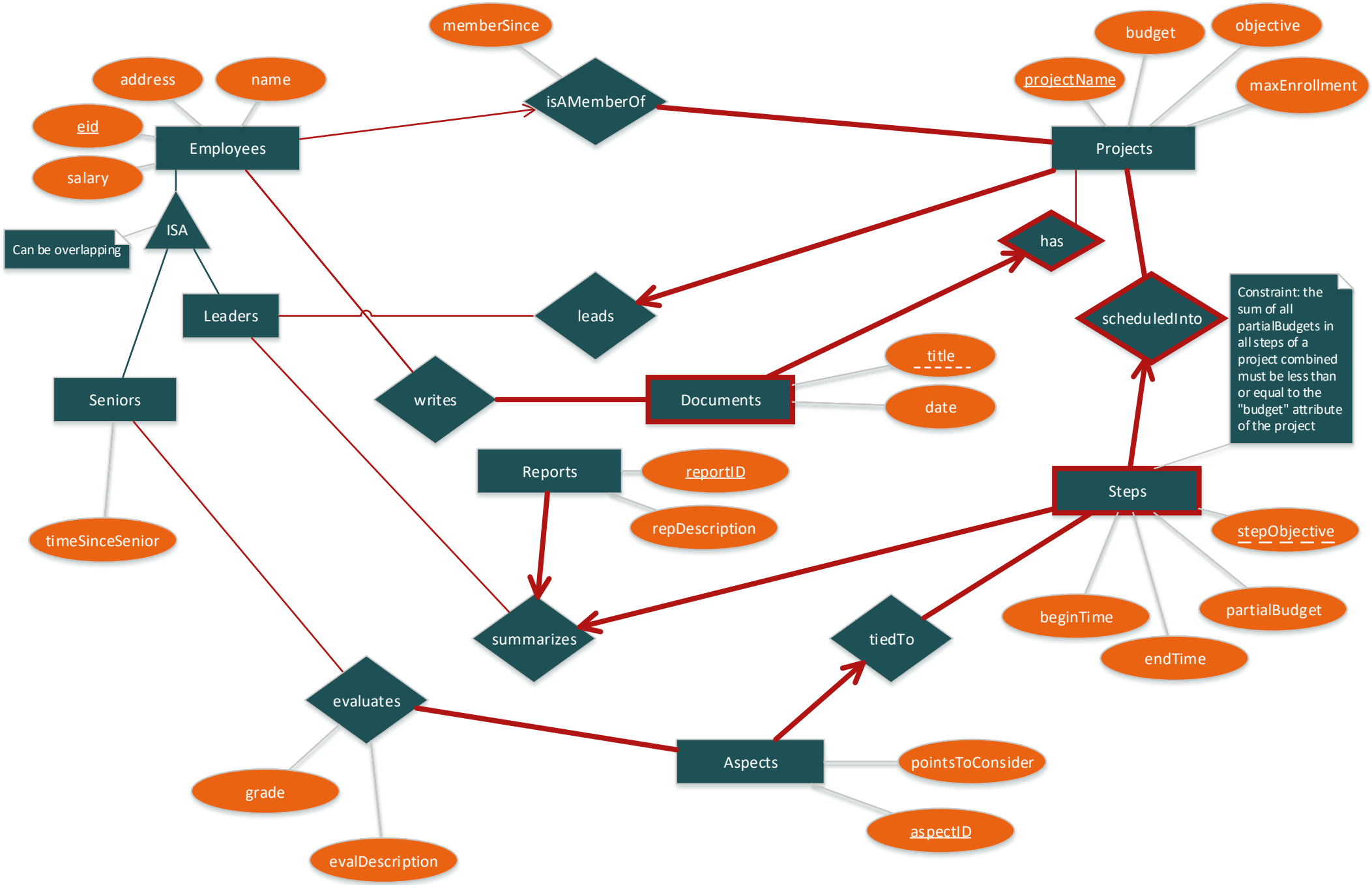
memberSince
address
name
budget
objective
projectName
maxEnrollment
isAMemberOf
eid
Employees
Projects
salary
ISA
Can be overlapping
has
leads
Leaders
scheduledInto
Constraint: the sum of all partialBudgets in all steps of a project combined must be less than or equal to the "budget" attribute of the project
title
writes
Documents
Seniors
date
Reports
reportID
Steps
repDescription
stepObjective
timeSinceSenior
beginTime
partialBudget
summarizes
tiedTo
endTime
evaluates
Aspects
pointsToConsider
grade
aspectID
evalDescription

Question 1)

Part 2)

Employees(eid, name, address, salary)

Seniors(eid, timeSinceSeniors) *(**eid** ref **Employees**)*

Leaders(eid)*(**eid** ref **Employees**)*

Projects(projectName, budget, objective, maxEnrollment, eid) *(**eid** ref **Leaders**)*

Membership(eid, projectName, memberSince) *(**eid** ref **Employees**, **projectName** ref **Projects**)*

Documents(title, date, projectName) *(**projectName** ref **Projects**)*

Writing(eid, title, projectName) *(**eid** ref **Employees**, **title** ref **Documents**, **projectName** ref **Projects**) (cannot reflect the participation constraint of **Documents** in the "writes" relation, from the E/R model)*

Reports(reportID, stepObjective, projectName, eid, repDescription) *(**stepObjective** ref **Steps**, **projectName** ref **Projects**, **eid** ref **Leaders**)*

Steps(stepObjective, projectName, partialBudget, beginTime, endTime, reportID, eid) *(**projectName** ref **Projects**, **reportID** ref **Reports**, **eid** ref **Leaders**) (cannot reflect the participation constraint of **Steps** in the "tiedTo" relation, from the E/R model)*

Aspects(aspectID, pointsToConsider, stepObjective, projectName)*(**stepObjective** ref **Steps**, **projectName** ref **Projects**) (cannot reflect the participation constraint of Aspects in the "evaluates" relation, from the E/R model)*

Evaluation(eid, grade, evalDescription, aspectID)*(**eid** ref **Seniors**, **aspectID** ref **Aspects**)*

Exercise 2

1.
   A. **Agree**. Every person can be the child of at most one other person. This is due to the key constraint from Person to ChildOf.
   B. **Agree**. Because people in the real world do not actually have a personID so this has been artificially created to identify between two people in the database.
   C. **Disagree**. There is no information in the diagram that supports this. Perhaps it would be necessary to add an attribute on ChildOf relation to enforce this.
   D. **Disagree**. There is no constraint that shows in relation ChildOf, personID not equal parentID. Due to lack of constraint, it is possible for personID = parentID.

2.
   A. **Disagree**. Doing that would mean all deposits would show as one entry between the customer and his/her account so individual deposits made by the customer would not be trackable but perhaps the sum of all deposits would still be trackable.
   B. **Agree**. Because in a make deposit relation as it is now, a single depositid can be linked to multiple customers and accounts, as long as there is one linked to it.
   C. **Disagree**. It is necessary because without it, A customer can make a deposit to his/her account without the deposit being recorded. An empty deposit would not make sense.
   D. **Disagree**. Because in that case a unique customer would be able to have only one deposit entry to a unique account and hence it would make the Deposits entity set useless.

Exercise 3

1) $\Pi_{city, streetaddr}(\sigma_{province='PQ'}(Branch))$

2) $\Pi_{custid}(\sigma_{balance>500}(\sigma_{atype='savings'}(Accounts)))$

3) $\Pi_{actid}(Customer \bowtie_{customer.name='Madea\ Fakename' \wedge Customer.custid = Account.custid \wedge Account.atype='Savings'} Account)$

4) $\Pi_{custid}(\sigma_{atype='checking' \wedge startdate - '2017-01-01' \geq 0}(Accounts)) \cap \Pi_{custid}(\sigma_{atype='savings' \wedge startdate - '2017-01-01' \geq 0}(Accounts))$

5) $\Pi_{actid}(\sigma_{transdate='2016-12-22' \wedge transtype='deposit'}(Transactions)) -$

$\Pi_{actid}(\sigma_{transdate='2016-12-22' \wedge transtype='withdraw'}(Transactions))$

6) $\rho(newBranch(BranchAddress), Branch(streetAddr)$

$\rho(T1, Employees \bowtie_{Employees.empid = newbranch.mgrid} newBranch.$

$\rho(T2, Customer \bowtie_{customer.name = T1.name \wedge customer.streetaddr = T1.streetaddr \wedge customer.city = T1.city \wedge customer.province = T1.province} T1)$

$\rho(T3, Account \bowtie_{Account.custid = T2.custid \wedge Account.atype='checking' \wedge Account.branchid = T2.branchid} T2)$

$\Pi_{name, branchAddress, city, province}(T3)$

7) $\rho(T1, Account \bowtie_{Account.custid = customer.custid} (\sigma_{name = 'Mone\ L\ Aunderer'}(Customer)))$

$\rho(T2, Transactions \bowtie_{transactions.accid = T1.accid} T1)$

$\rho(T3, Transactions \bowtie_{transactions.accid = T1.accid \wedge transactions.branchid = T1.branchid} T1)$

$\Pi_{tid}(\sigma_{transtype = withdraw}(T2 - T3))$

8) $\rho(T1, (Customer \bowtie_{customer.custid = account.custid \wedge account.atype = 'savings'} Account))$

$\rho(T2, (Transactions \bowtie_{transaction.accid = T1.accid \wedge transaction.transdate \geq '2017-01-01'} T1))$

$\rho(T3, T2)$

$\rho(T4, T3 \bowtie_{T3.custid = T2.custid \wedge T3.tid \neq T2.tid} T2)$

$\rho(T5, T4 - T2)$

$\Pi_{accid, name, custid}(T5)$