OCL Constraints- Case Study 1

1. **context** Branch:: nameIsKey() : Boolean

   post:

   result=Branch.allInstances()-> **select**(b|b.name=self.name)->size()=1

2. **context** BranchType:: nameIsKey() : Boolean

   post:

   result=BranchType.allInstances()-> **select**(b|b.name=self.name)->size()=1

3. **context** Country:: nameIsKey() : Boolean

   post:

   result=Country.allInstances-> **select**(b|b.name=self.name)->size()=1

4. **context** PerformanceIndicator:: nameIsKey() : Boolean

   post:

   result=PerformanceIndicator.allInstances()-> **select**(b|b.name=self.name)->size()=1

5. **context** ServiceDepot:: nameIsKey() : Boolean

   post:

   result=ServiceDepot.allInstances()-> **select**(s|s.name=self.name)->size()=1

6. **context** CarModel:: nameIsKey() : Boolean

   post:

   result=CarModel.allInstances-> **select**(b|b.name=self.name)->size()=1

7. **context** CarGroup:: nameIsKey() : Boolean

post:

result=CarGroup.allInstances()-> **select**(b|b.name=self.name)->size()=1

8. **context** RentalDuration:: nameIsKey() : Boolean

post:

result=RentalDuration.allInstances()-> **select**(b|b.name=self.name)->size()=1

9. **context** Discount:: nameIsKey() : Boolean

post:

result=Discount.allInstances()-> **select**(b|b.name=self.name)->size()=1

10. **context** MakeRental:: pickUpBranch() : Branch

post:

let branches:Set(Branch) = Branch.allInstances() -> **select**(name=self.pickUpId)

in

branches->size()=1 implies result=branches->**any()**

11. **context** MakeRental:: dropOffBranch() : Branch

post:

let branches:Set(Branch) = Branch.allInstances()-> **select**(name=self.dropOffId)

in

branches->size()=1 implies result=branches->**any()**

12. **context** PendantCarOrder:: idIsKey() : Boolean

post:

result=PendantCarOrder.allInstances()-> select(b|b.id=self.id)->size()=1

13. **context** EU_RentPerson:: idIsKey() : Boolean

post:

result=EU_RentPerson.allInstances() ->select(p|p.id=self.id)->size()=1

14. **context** DemandXModel:: demand() : Integer

post:

let pendantRes:Reservation= Reservation.allInstances()->
select(r|r.beginning.date()=tomorrow())-> **select**(r| r.pickUpBranch=self.branch and r.car-> isEmpty())
in
result=pendantRes.requestedModel-> **select**(m|m=d.carModel)->size()

15. **context** DemandXGroup:: demand() : Integer

post:

let pendantRes:Reservation= Reservation.allInstances()->
select(r|r.beginning.date()=tomorrow())-> **select**(r|
r.pickUpBranch=self.branch and r.car-> **isEmpty()**)
in
result=pendantRes.requestedGroup-> **select**(m|m=d.carGroup)->size()

16. **context** DemandXModel:: demand() : Integer

post:

let pendantRes:Reservation= Reservation.allInstances()->
select(r|r.beginning.date()=tomorrow())-> **select**(r| r.pickUpBranch=self.branch and r.car-> isEmpty())
in

```
        result=pendantRes.requestedModel-> select(m|m=d.BikeModel)->size()
```

```
17.     context DemandXGroup:: demand() : Integer
    post:
        let pendantRes:Reservation= Reservation.allInstances()->
        select(r|r.beginning.date()=tomorrow())-> select(r |
        r.pickUpBranch=self.branch and r.car-> isEmpty())
        in
        result=pendantRes.requestedGroup-> select(m|m=d.BikeGroup)->size()
```

```
18.     context CarGroup:: totalOrder() : Boolean
    post:
        let isWorse (w,b:CarGroup):Boolean= b.worse=w or
        isWorse(w,b.worse)
        let isBetter (b,w:CarGroup):Boolean= w.better=b or
        isBetter(b,w.better)
        in
        result = CarGroup.allInstances()->one (cg|cg.worse->isEmpty())
        and CarGroup.allInstances()->one (cg|cg.better->isEmpty()) and
        CarGroup.allInstances()->forall (cg1,cg2| isWorse(cg1,cg2)
        implies not isBetter (cg1,cg2) and isBetter (cg1,cg2) implies
        not isWorse (cg1,cg2))
```

```
19.     context RentalDuration:: totalOrder() : Boolean
    post:
        let isShorter(s,l:RentalDuration):Boolean= l.shorter=s or
        isShorter(s,l.shorter)
        let isLonger(l,s:RentalDuration):Boolean= s.longer=l or
        isLonger(l,s.longer)
        in
        result = RentalDuration.allInstances()->one (rd| rd.shorter-> isEmpty())
```

and RentalDuration.allInstances() -> **one** (rd|rd.longer->isEmpty()) **and**
RentalDuration.allInstances()-> **forAll** (rd1,rd2| isShorter (rd1,rd2)
implies not isLonger (rd1,rd2) **and** isLonger (rd1,rd2) implies not isShorter (rd1,rd2))

20. **context** ExistingRentalDuration:: duration() : RentalDuration

post:

let rentDuration:Set (RentalDuration)= RentalDuration. allInstances()->
select(rd| rd.name=self.durationName)
in
rentDuration->notEmpty() implies result= rentDuration-> **any()**
and self. perf= durationLimit

21. **context** ExistingPerformanceIndicator:: perfInd() : PerformanceIndicator

post:

let perf: Set(PerformanceIndicator)= PerformanceIndicator. allInstances()->
select(pi|pi.name=self.name)
in
perf->notEmpty() implies result=perf-> **any()**
and self. perf= performanceLevel

22. **context** ExistingCar:: car() : Car

post:

let carI: Set(Car)=Car.allInstances()-> **select**(c|c.registrationNumber=self.regNumber)
in
carI->**notEmpty()** implies result=carI->**any()**

23. **context** ExistingCarGroup:: carG() : CarGroup

post:

let carGr:Set(CarGroup)= carGroup.allInstances()-> **select**(cG| cG.name=self.carGroup)

in

carGr->notEmpty() implies result=carGr-> **any()**

24. **context** ExistingCarModel:: carM() : CarModel

post:

let carMod: Set(CarModel)=CarModel.allInstances()-> **select**(cM|

cM.name=self.carModel)

in

carMod->notEmpty() implies result=carMod-> **any()**

25. **context** ExistingDiscount:: discount() : Discount

post:

let dis: Set(Discount)= Discount.allInstances()-> **select**(d|d.name=self.discountName)

in

dis->notEmpty() implies result= dis-> **any()**

26. **context** ExistingRentalDuration:: duration() : RentalDuration

post:

let rentDuration:Set (RentalDuration)=RentalDuration. allInstances()->**select** (rd|

rd.name= self.durationName)

in

rentDuration->notEmpty() implies result= rentDuration-> **any()**

27. **context** ExistingPerformanceIndicator:: perfInd() :PerformanceIndicator

post:

let perf: Set(PerformanceIndicator)= PerformanceIndicator. allInstances()->

select(pi|pi.name=self.name)

in

perf->notEmpty() implies result=perf-> **any()**

28. **context** NewCarGroupDurationPrice:: apply() : CarGroup

post:

cgdp.oclIsNew() **and** cgdp.oclIsTypeOf(CarGroupDurationPrice)
and cgdp.price=self.price **and** cgdp.carGroup=self.carG **and**
cgdp.rentalDuration=duration

29. **context** NewCGDPForNewDuration:: apply()

post:

cgdp.oclIsNew() **and** cgdp.oclIsTypeOf(CarGroupDurationPrice)
and cgdp.price=self.price **and** cgdp.carGroup=self.carG **and**
cgdp.rentalDuration=duration

30. **context** ExtendedCarAllocationDefinitions:: 2upgradePossible(): Boolean

post:

result= **if** self.2upgradeGroup->isNotEmpty()
then(self.curBranch.nextDayR.car-> **collect**(carGroup)-> **includes**
(upgradeGroup) or self.groupAvail(self.upgradeGroup)) **and**
self.groupAvail(self.2upgradeGroup)->isNotEmpty **and**
self.groupAvail(self.2upgradeGroup).quantity@pre **-**
self.demXGroup->select(d|d.carGroup=self.2upgradeGroup).
demand@pre >0.1*self.groupQuota(self.curBranch,
self.2upgradeGroup)
else
False

31. **context** ExtendedCarAllocationDefinitions::downgradePossible():Boolean

post:

```
        result= if self.downgradeGroup->isNotEmpty()
                then self.groupAvail(self.downgradeGroup)->isNotEmpty and
                self.groupAvail(self.downgradeGroup).quantity@pre -
                self.demXGroup->select(d|d.carGroup=self.downgradeGroup).
                demand@pre >0.1*self.groupQuota(self.curBranch,
                self.downgradeGroup)
                else
                False
```

32. **context** WithSurplus:: allInstances() : Boolean

post:

```
        CalculateOwnCars.allInstances()-> select(c|c.answerSurplus)
```

33. **context** WithLack:: allInstances() : Boolean

post:

```
        CalculateOwnCars.allInstances()-> select(c|c.answerLack)
```

34. **context** CarAllocationDefinitions:: demXModel() : DemandXModel

post:

```
        result=self.reservation.pickUpBranch.demandXModel
```

35. **context** CarAllocationDefinitions:: demXGroup() : DemandXGroup

post:

```
        result=self.reservation.pickUpBranch.demandXGroup
```

36. **context** RequestTransfer:: apply() : Boolean

post:

self.**oclIsTypeOf**(MoveCars).^apply() **and** self.otherBranch.
carsAvailable@pre->intersection(self.otherBranch.car->
select(c|c.**oclIsKindOf**(BeingTransferredCar) **and**
c.**oclIsTypeOf**(BeingTransferredCar).destination=self.askingBranch))->
size()=movedCars

37. **context** DoTransfer:: apply() : Boolean
post:
self.**oclIsTypeOf**(MoveCars).^apply() **and** self.askingBranch.
carsAvailable@pre->intersection(self.askingBranch.car->
select(c|c.**oclIsKindOf**(BeingTransferredCar) **and**
c.**oclIsTypeOf**(BeingTransferredCar).destination=self.otherBranch))->size()=movedCars

38. **context** EndOfMaintenance:: carWasBeingMaintained() : Boolean
post:
result=self.car.**oclIsTypeOf**(MaintenanceScheduled) **and**
self.car.**oclAsType**(MaintenanceScheduled).beginningDate<now())

39. **context** EndOfRepairs:: carWasBeingRepaired() : Boolean
post:
result=self.car.**oclIsTypeOf**(RepairsScheduled) **and**
self.car.**oclAsType**(RepairsScheduled).beginningDate< now())

40. **context** RequestTransfer:: apply() : Boolean
post:
self.**oclAsType**(MoveCars).^apply() **and** self.otherBranch.
carsAvailable@pre ->intersection (self.otherBranch.car ->
select(c|c.**oclIsKindOf**(BeingTransferredCar) **and**
c.oclAsType(BeingTransferredCar).destination= self.askingBranch))->size()=movedCars