I.

```
Person(string name, string ssn primary key, Date dob)
Boat(string name primary key, int tonnage)
Race(string name primary key, string winnerBoat)
  foreign key: winnerBoat -> Boat.name
Ownership(string owner, string boat, Date begin, Date end)
  foreign key: owner -> Person.ssn
               boat -> Boat.name
```

II.

1. For the boats who won the "Americas Cup" title, return the (boat, owner) object pairs. The query result should have type set<struct { Boat boat, Person owner }>.

answer:

```
select struct(boat: o.boat, owner: p)
from o in ownerships, p in o.coOwners
where 'Americas Cup' in o.boat.racesWon
```

2. Find the boat(s) ever owned by "Jack Sparrow". The query result should have type set<Boat>.

answer:

```
select o.boat
from p in persons, o in ownerships
where p.name = 'Jack Sparrow' and p in o.coOwners
```

3. Now assume that the definition of class Person is enriched with the declaration

    relationship set<Ownership> ownerships inverse Ownership::coOwners;

    and redo query II.2 exploiting this relationship.

answer:

```
select o.boat
from p in persons, o in p.ownerships
where p.name = 'Jack Sparrow'
```

4. Find the boat(s) most recently owned by "Jack Sparrow". The query result should have type set<Boat>.

answer:

```
define has_ownerships(owner_name) as
select o
from p in persons, o in p.ownerships
where p.name = owner_name

select o.boat
from o in has_ownerships('Jack Sparrow')
where o.end is null
    or ( for all o1 in has_ownerships('Jack Sparrow') : o1.end != null
and o.end >= o1.end)
;
```

5. Dropping the assumption of point 3., find the owners (return the objects themselves) of all "Americas Cup"-winning boats.

answer:

```
select p
from p in persons
```

```
where for all owners in (
    select o.coOwners
    from o in ownerships
    where 'Americas Cup' in o.boat.racesWon
  ) : p in owners
```

III. Express queries II.1, II.2, II.4 and II.5 in QBE, on schema of point I.

1.

```
Race | name           winnerBoat
-----------------------------
     | America Cups   _b

Ownership | owner, boat, begin, end
-----------------------------------
          | _o     _b

result | boat, owner
--------------------
  I.   | _b    _o
```

2.

```
Person | name,          ssn,   dob
---------------------------------
       | Jack Sparrow   _p

Ownership | owner, boat, begin, end
-----------------------------------
          | _p     _b

result | boat
-------------
  I.   | _b
```

4. NOTE: I use ^ as negative symbol

```
# stage 1: get all boats owned by Jack, return (boat, end)
Person | name,          ssn,   dob
---------------------------------
       | Jack Sparrow   _p

Ownership | owner, boat, begin, end
-----------------------------------
          | _p     _b           _e

boat_end | boat, end
--------------------
  I.     | _b    _e

# stage 2: get all end dates that are not latest
boat_end | boat, end
--------------------
         |       _e1
```

```
boat_end | boat, end
-------------------
         |       _e2

# condition
-----------------------------------------------------------------
| (_e1 < _e2 and _e2 != null) or ( _e1 != null and _e2 == null) |
-----------------------------------------------------------------

not_latest | end
----------------
I.         | _e1


# stage 3: return boats with end date not in 'not_latest'
boat_end | boat, end
--------------------
         | _b,   _e1

not_latest | end
----------------
  ^        | _e1

result | boat
-------------
  I.   | _b

5. NOTE: I use ^ as negative symbol
# stage 1
Race | name           winnerBoat
----------------------------
     | America Cups   _b

Ownership | owner, boat, begin, end
-----------------------------------
   ^      | _o     _b

bad_owner | owner
-----------------
   I.     | _o


#stage2
Person | name, ssn,   dob
--------------------------------
       |       _o

bad_owner | owner
-----------------
   ^      | _o

result | owner
--------------------
  I.   | _o
```