1(a) (seller, iid) can’t be a primary key. Because we can't determine a unique item by seller and iid.

1(b) For table BID, bidtime must be part of the key. Because one bidder can bid multiple times, and we need to differentiate according to the time of each bid.

1(c)

1. SELECT `user`.uid, COUNT(auction.iid) FROM `user` LEFT OUTER JOIN auction ON auction.seller=`user`.uid GROUP BY `user`.uid;
2. SELECT bid.seller,bid.iid,bid.starttime,MAX(bidprice) FROM auction,bid WHERE bid.seller=auction.seller AND bid.starttime = auction.starttime AND bid.iid=auction.iid AND bid.bidtime BETWEEN auction.starttime AND auction.endtime AND bid.bidprice>=auction.minbid GROUP BY bid.seller,bid.iid,bid.starttime;

(

OR:

SELECT bid.seller,bid.iid,bid.starttime,bid.bidprice FROM bid, auction WHERE bid.seller=auction.seller AND bid.starttime = auction.starttime AND bid.iid=auction.iid AND bid.bidtime BETWEEN auction.starttime AND auction.endtime AND bid.bidprice>=auction.minbid AND NOT EXISTS (SELECT \* FROM bid AS t1 WHERE t1.seller = bid.seller AND t1.starttime = bid.starttime AND t1.iid = bid.iid AND t1.bidprice > bid.bidprice);

)

1. SELECT MAX(bid.bidprice) FROM bid LEFT JOIN itemtype ON bid.iid = itemtype.iid WHERE itemtype.itemname='IPhone 12';

(

OR:

SELECT bid.bidprice FROM bid LEFT JOIN itemtype ON bid.iid = itemtype.iid WHERE itemtype.itemname='IPhone 12' AND NOT EXISTS (SELECT \* FROM bid AS t1 WHERE t1.seller = bid.seller AND t1.starttime = bid.starttime AND t1.iid = bid.iid AND t1.bidprice > bid.bidprice);

)

1. SELECT bid.bidder FROM bid, auction WHERE bid.seller=auction.seller AND bid.starttime = auction.starttime AND bid.iid=auction.iid AND bid.bidtime BETWEEN auction.starttime AND auction.endtime AND bid.bidprice>=auction.minbid AND NOT EXISTS (SELECT \* FROM (SELECT bid.seller,bid.iid,bid.starttime,bid.bidprice FROM bid, auction WHERE bid.seller=auction.seller AND bid.starttime = auction.starttime AND bid.iid=auction.iid AND bid.bidtime BETWEEN auction.starttime AND auction.endtime AND bid.bidprice>=auction.minbid AND NOT EXISTS (SELECT \* FROM bid AS t1 WHERE t1.seller = bid.seller AND t1.starttime = bid.starttime AND t1.iid = bid.iid AND t1.bidprice > bid.bidprice)) AS t2 WHERE t2.seller=bid.seller AND t2.iid=bid.iid AND t2.starttime=bid.starttime AND bid.bidprice=t2.bidprice) GROUP BY bid.bidder HAVING COUNT(bid.bidprice) > 10;
2. SELECT itemtype.itemname, t1.`condition`,AVG(t1.bidprice) FROM (SELECT bid.seller,bid.iid,bid.starttime,bid.bidtime, bid.bidprice, auction.`condition` FROM bid, auction WHERE bid.seller=auction.seller AND bid.starttime = auction.starttime AND bid.iid=auction.iid AND bid.bidtime BETWEEN auction.starttime AND auction.endtime AND bid.bidprice>=auction.minbid AND NOT EXISTS (SELECT \* FROM bid AS t1 WHERE t1.seller = bid.seller AND t1.starttime = bid.starttime AND t1.iid = bid.iid AND t1.bidprice > bid.bidprice)) AS t1, itemtype WHERE t1.iid = itemtype.iid AND YEAR(t1.bidtime)=2018 GROUP BY itemtype.itemname, t1.`condition`;
3. SELECT DISTINCT bid.bidder FROM bid WHERE bid.bidder=bid.seller;
4. SELECT DISTINCT CASE WHEN bidder1<bidder2 THEN bidder1 ELSE bidder2 END bu1, CASE WHEN bidder1<bidder2 THEN bidder2 ELSE bidder1 END bu2 FROM (SELECT DISTINCT t1.bidder bidder1,t2.bidder bidder2 FROM (SELECT \* FROM bid WHERE YEAR(starttime)=2018) AS t1, (SELECT \* FROM bid WHERE YEAR(starttime)=2018) AS t2 WHERE t1.seller=t2.seller AND t1.iid=t2.iid AND t1.starttime=t2.starttime AND t1.bidder<>t2.bidder GROUP BY t1.bidder,t2.bidder HAVING COUNT(DISTINCT t1.seller, t1.iid, t1.starttime)>=10) AS t3;

1(d)

1. user.uid  count(auction.iid) (user  user.uid=auction.seller auction)
2. ∏ bid.seller, bid.iid, bid.starttime, endprice (bid.seller,bid.iid,bid.starttime max(bid.bidprice) as endprice (σ auction.seller=bid.seller ∧ auction.iid=bid.iid ∧ auction.starttime=bid.starttime ∧ bid.bidtime<=auction.endtime ∧ bid.bidtime >= auction.starttime ∧ bid.bidprice >= auction.minbid (auction ⅹ bid)))
3.  max(bid.bidprice) (σ itemtype.itemname=’IPhone 12’ (bid bid.iid = itemtype.iid itemtype))
4. ∏ bid.bidder (σ bittimes>10 (bid.bidder count(bid.bidprice) as bidtimes ((∏ bid.bidder, bid.seller, bid.iid, bid.starttime, bit.bidtime, bidprice (σauction.seller=bid.seller ∧ auction.iid=bid.iid ∧ auction.starttime=bid.starttime ∧ bid.bidtime<=auction.endtime ∧ bid.bidtime >= auction.starttime ∧ bid.bidprice >= auction.minbid (auction ⅹ bid) ) ÷ ∏ bid.seller, bid.iid, bid.starttime, endprice (bid.seller,bid.iid,bid.starttime max(bid.bidprice) as endprice (σ auction.seller=bid.seller ∧ auction.iid=bid.iid ∧ auction.starttime=bid.starttime ∧ bid.bidtime<=auction.endtime ∧ bid.bidtime >= auction.starttime ∧ bid.bidprice >= auction.minbid (auction ⅹ bid)))))))
5. ∏ avg\_price (itemtype.itemname, t1.conditionAVG(bid.bidprice) as avg\_price (σt1.iid=itemtype.iid∧YEAR(t1.bidtime)=2018 ((∏ bid.seller, bid.iid, bid.starttime, endprice, auction.condition (bid.seller,bid.iid,bid.starttime max(bid.bidprice) as endprice (σ auction.seller=bid.seller ∧ auction.iid=bid.iid ∧ auction.starttime=bid.starttime ∧ bid.bidtime<=auction.endtime ∧ bid.bidtime >= auction.starttime ∧ bid.bidprice >= auction.minbid (auction ⅹ bid)))) as t1 ⅹ itemtype)))
6. ∏bid.bidder(σbid.bidder=bid.seller (bid))

1(e)

1. Can’t.
2. { <seller,bidder,iid,price> | ∃ seller,iid,starttime,endtime ( <seller,iid,starttime,endtime> ∈ auction) ∧ ∃ seller1,iid1,starttime1,bidtime (<seller1,iid1,starttime1,bidtime1, price1 > ∈ bid) ∧ seller=seller1 ∧ iid=iid1 ∧ starttime=starttime1 ∧ endtime>=bidtime1 ∧ startime<= bidtime1 ∧ ∀ seller2, iid2, starttime2,price2 (<seller2, iid2, starttime2, price2> ∈ bid = Ø ∧ seller1=seller2 ∧ iid1=iid2 ∧ starttime2=starttime1 ∧ price2>price1 ))}
3. { <price> | ∃ iid (<seller, iid, starttime, price> ∈ bid) ∧ ∃ iid1,itemname (<iid1, itemname> ∈ itemtype) ∧ iid=iid1 ∧ ∀ seller1, iid2, starttime1(<seller1, iid2, starttime1, price1> ∈ bid = Ø ∧ seller1=seller2 ∧ iid=iid2 ∧ starttime=starttime1 ∧ price1>price ) }
4. Can’t.
5. Can’t.

These problem i, iv, v all use the aggregation functions that are not supported by tuple calculus and domain calculus, such as avg,sum,count, etc., and cannot be converted into another description like max and min functions.

2.

1. SELECT DISTINCT \* FROM (SELECT CASE WHEN u1<u2 THEN u1 ELSE u2 END user1, CASE WHEN u1<u2 THEN u2 ELSE u1 END user2 FROM (SELECT uid u1, pinner u2 FROM (SELECT picture\_like.uid, picture.pinner FROM picture, picture\_like WHERE picture.pid=picture\_like.pid AND picture.pinner<>picture\_like.uid GROUP BY picture\_like.uid, picture.pinner HAVING COUNT(picture.pid)>=3) t1 WHERE EXISTS (SELECT \* FROM (SELECT picture\_like.uid, picture.pinner FROM picture, picture\_like WHERE picture.pid=picture\_like.pid AND picture.pinner<>picture\_like.uid GROUP BY picture\_like.uid, picture.pinner HAVING COUNT(picture.pid)>=3) t2 WHERE t2.pinner=t1.uid AND t2.uid=t1.pinner)) AS t3) t4;
2. SELECT picture.pid, COUNT(pinned\_time)-1 as res FROM picture GROUP BY picture.pid;
3. SELECT uname FROM user, board\_follower WHERE user.uid=board\_follower.follower AND board\_follower.bid IN (SELECT bid FROM picture WHERE picture.description LIKE '%cactus at sunset%');
4. SELECT uname FROM `user` WHERE uid IN (SELECT uid FROM picture\_review WHERE picture\_review.review\_time between now() and (now() - interval 24 hour) GROUP BY uid HAVING COUNT(picture\_review.review\_time)>10);