Question 1: Which of the following statements are true
1. Each superkey is a superset of some candidate key.
2. Each primary key is also a candidate key, but there may be candidate keys that are not primary keys.

a) only 1 is true
b) only 2 is true
c) both 1 and 2 are true
d) neither 1 nor 2 are true

Answer C

The following two questions refer to the relational scheme R (A, B, C, D, E , F, G, H) and the following functional dependencies over R:

A → BCD
AD → E
EFG → H
F → GH

Question 2: Based on the functional dependencies, there is one key for R.

What is it ?   AF

Question 3: One of the four functional dependencies can be removed without altering the key.

Which one ?   EFG -> H

Question 4

Suppose that two relations R(A, B) and S(A, B) have exactly the same schema. Consider the following equalities in relational algebra?

    I. $R \cap S = R - (R - S)$
    II. $R \cap S = S - (S - R)$
    III. $R \cap S = R$ NATURAL-JOIN S
    IV. $R \cap S = R \times S$

Which of the equalities hold in relational algebra?

        (A) I only
        (B) I and II only
        (C) I, II and III
        (D) I, II, III and IV

ANSWER: (c   )

Question 5

Which of the following statements are true about weak entity sets:

    1. A weak entity set cannot have a primary key.
    2. A weak entity set must have a local attribute in primary key
    3. A weak entity must borrow an attribute from another entity set to form a primary key.

        (a) None of them.
        (b) (1) and (2) only.
        (c) (2) and (3) only.
        (d) (3).

ANSWER: (d   )

Question 6: Consider the following instances of relations R(A, B, C, D, E) and S (P, Q, R, S):

A	B	C	D	E
7	4	1	2	3
1	5	3	1	1
1	6	3	2	4
5	5	1	2	3

P	Q	R	S
7	2	1	4
5	2	1	2
1	2	3	7

Which of the following inclusion dependencies hold over the instances of relation R and S given above?

I)   PQ⊆ AD
II)   RS ⊆ CD
III)   PQR ⊆ ADC

a) I & III   b) II & III   c) I & II   d) I only

ANSWER: (a   )

Question: 7

The table Arc(x,y) currently has the following tuples (note there are duplicates): (1,2), (1,2), (2,3), (3,4), (3,4), (4,1), (4,1), (4,1), (4,2). Compute the result of the query:
```
SELECT a1.x, a2.y, COUNT(*)
FROM Arc a1, Arc a2
WHERE a1.y = a2.x
GROUP BY a1.x, a2.y;
```

Which of the following tuples is in the result?

I) (1,3,2)

II) (2,4,6)

III) (3,1,4)

IV) (3,1,6)

a) I) and III)   b) I) and II)   c) I) and IV)   d) II and III

ANSWER: (C   )

Question: 8

The relation R(a,b) may have duplicate tuples. Which of the following queries has a result that is **guaranteed** not to have duplicates, regardless of what tuples R contains?

I)   SELECT a FROM R WHERE a = 1

II) SELECT MAX(b) FROM R GROUP BY a

III) SELECT a, b FROM R GROUP BY a, b

IV) SELECT a FROM R WHERE a NOT IN (SELECT a FROM R)

a) III) and IV)   b) I) and II)   c) III only   d) I and III

ANSWER: (a   )

PART II

Question 11 (25 pts)
Consider the following database schema:

Frequents(kid, store)
Sells(store, candy)
Likes(kid, candy)

Table **Frequents** indicates what candy stores a kid likes to visit. Table **Sells** shows which candy each store sells. Table **Likes** tells which candy a kid likes.

Express the following queries. When you use relational algebra, you can either write a single expression, or write a sequential relational algebra program with a series of assignments, or draw a tree. If you think it is impossible to express a particular query in relational algebra, say so and justify your answer.

(a) (*4 pts*) Use relational algebra to find the kids who like "Hersheys" or "Mars" candy.

$H \leftarrow \Pi_{KID}(\sigma_{CANDY = \text{"HERSHEYS"}}(LIKES))$
$M \leftarrow \Pi_{KID}(\sigma_{CANDY = \text{"MARS"}}(LIKES))$
BOTH $\leftarrow$ H UNION M
$\Pi_{KID}(BOTH)$

(b) (*6 pts*) Use relational algebra to find the kids that frequent at least one candy store that sells a candy they like.

$R \leftarrow \Pi_{KID, STORE}(LIKES \bowtie_{NATURAL\text{-}JOIN} SELLS)$
$Sol \leftarrow \Pi_{KID}(\sigma_{F.STORE=R.STORE \wedge F.KID = R.KID}(F \text{ x } R))$

(c) (6 *pts*) Assume each kid likes at least one candy and frequents at least one store. Use relational algebra to find the kids that frequent *only* stores that serve *some* candy that they like.

$R \leftarrow \Pi_{KID, STORE} (LIKES_{\ NATURAL\text{-}JOIN}\ SELLS)$
$S \leftarrow \Pi_{KID} (FREQUENTS - R)$
$Sol \leftarrow \Pi_{KID} (FREQUENTS) - S$

(d) (4 *pts*) Use **SQL** to list the stores (in alphabetical order) that sell more than 10 different candies.

```
Select store
From sells
Group by stores
Having count(distinct candy) > 10;
```

(e) (5 *pts*) Use **SQL** to list the kids whose liked candies are all sold by the store "Starbucks."

(Select kid from likes) EXCEPT
(select kid from likes where candy NOT in (select candy from sells where store = 'starbucks');

The next three questions are based on the following information

1. Manufacturers have a name, which we may assume is unique, an address, and a phone number.
2. Products have a model number and a type (e.g., television set). Each product is made by one manufacturer, and different manufacturers may have different products with the same model number. However, you may assume that no manufacturer would have two products with the same model number.
3. Customers are identified by their unique social security number. They have email addresses, and physical addresses. Several customers may live at the same (physical) address, but we assume that no two customers have the same email address.
4. An order has a unique order number, and a date. An order is placed by one customer. For each order, there are one or more products ordered, and there is a quantity for each product on the order.

Question 11: (20 points) In the space below, draw an E/R diagram that represents the above information. Do not forget to underline the keys as usual.
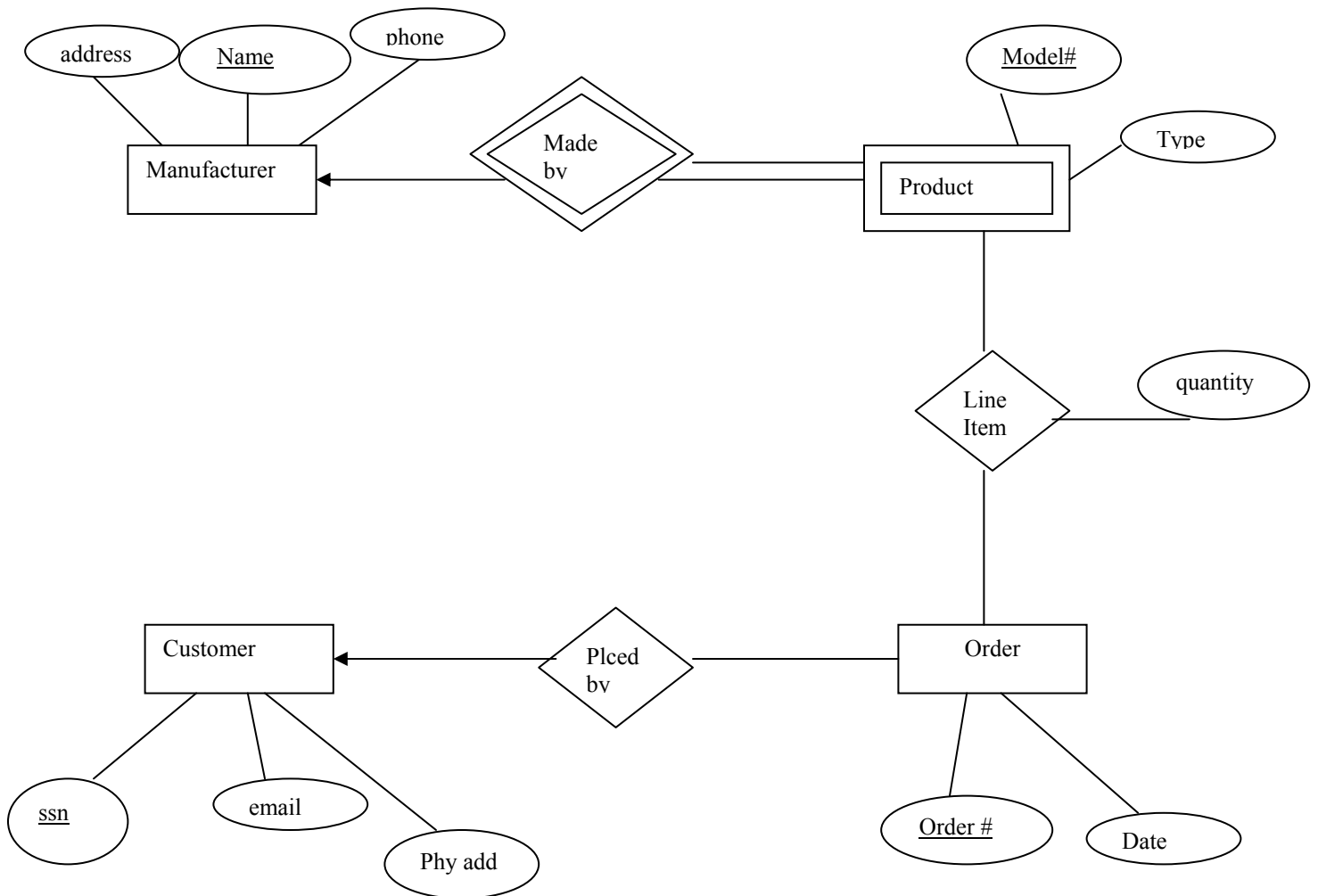
Question 12: (15 points) Convert the E/R diagram in Question 11 to the relational model. Remember to specify not just the relational scheme but also integrity constraints including keys as well as inclusion dependencies.

Manufacturer (address, <u>name</u>, phone)
Product (<u>p_name, model_no</u>, type)
Customer (<u>ssn</u>, email, address)
Order (<u>order_no</u>, date)
Line_item (<u>p_name, model_no, order_no</u>)
Placed_by (ssn, <u>order_no</u>)
Product [p_name] ⊆ Manufacturer [name]
Placed_by [ssn] ⊆ Customer [ssn]
Line_item [name, model_no] ⊆ Product [name, model_no]