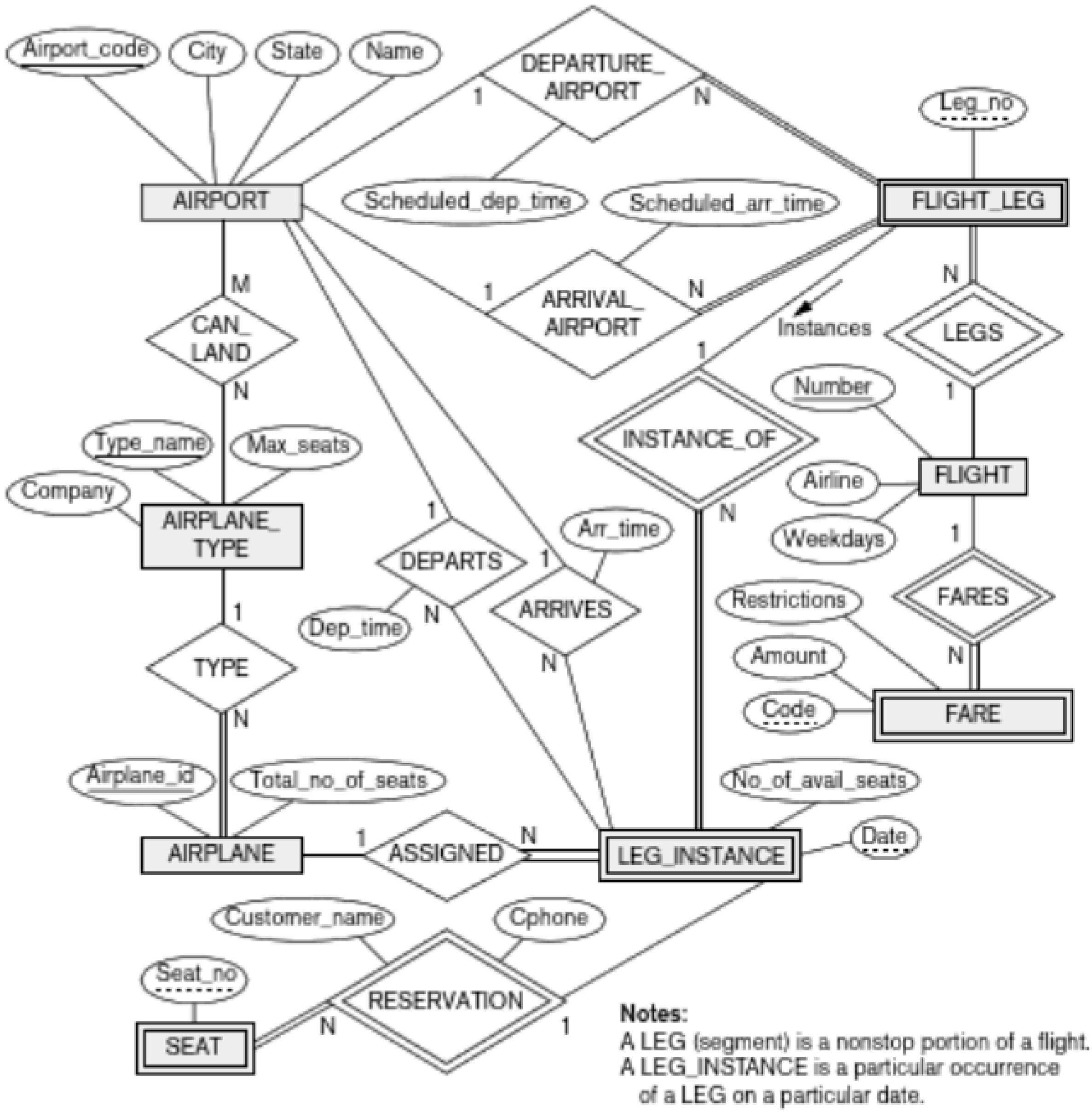
Q1- [7.5 mark] Given the following ER diagram, answer the following questions.



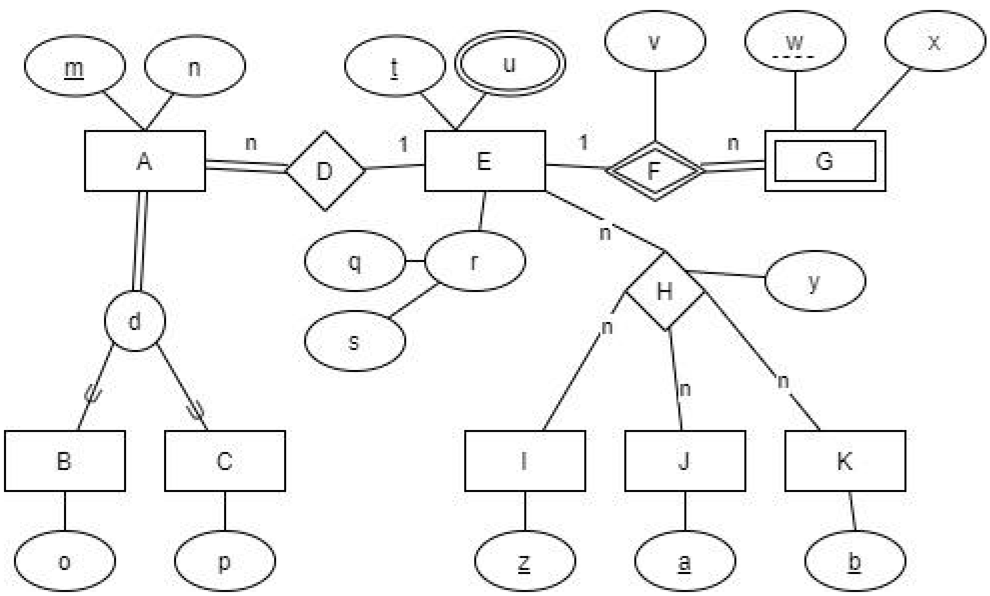

a) (0.5 mark) An airplane can have multiple types. **False**
b) (0.5 mark) There might be airports that don't accept any airplane types. **True**
c) (1 mark) What is the primary key of table CAN_LAND? [**Airport_code, Type_name**]
d) (0.5 mark) It is possible that in the final schema, there is no specific table for AIRPLANE_TYPE. **False**

e) (1 mark) What are the weak entities in this diagram? **Flight_leg, leg_instance, seat, fare**
f) (2 marks) According to this diagram, a weak entity can be related to a weak entity itself in a hierarchy where the final primary key is the combination of all previous keys. What is the primary key of SEAT? **Leg_no, Date, Seat_no**
g) (0.5 mark) An airplane must be assigned to a LEG_INSTANCE. **False**
h) (1 mark) According to this design, you can insert the seats of an AIRPLANE after inserting the airplane and before inserting the LEG_INSTANCE records. **False**
i) (0.5 mark) All the seats should be reserved. **True or False**

Q2. [10 marks] Transform the ER diagram into a relational schema using the methods discussed in class.

Note 1: Remember to include your foreign keys in the bold format, and underline primary key attributes:
T(attribute1, attribute2, **attribute3**) here attribute1 and attribute2 are together the primary key, and attribute 3 is the foreign key for table T.

Note 2: For specialisation, use option 8A from slides:
*``We create a relational table for the superclass and create a relational table for each subclass. The primary key of each of the subclass is the primary key of the superclass.``*




A(m, n, **t**) or A(m, n) and D(m, **t**)    2 marks [both mappings are possible]
B(**m**, o) 1 mark
C(**m**, p) 1 mark
E(t, q, s) 1 mark
U(**t**, u) 1 mark
FG(**t**, w, x, v) 1.5 marks
H(**t**, **z**, **a**, **b**, y) 1 marks
I(z) 0.5 mark
J(a) 0.5 mark
K(b) 0.5 mark

Q3- [3 marks] Which of the functional dependencies might hold given the data in relation R(A,B,C,D). Please circle all FDs that may hold.

| **A** | **B** | **C** | **D** |
|---|---|---|---|
| 1 | 3 | 4 | 3 |
| 2 | 3 | 3 | 2 |
| 2 | 1 | 2 | 1 |
| 3 | 2 | 1 | 4 |
| 4 | 2 | 1 | 4 |

| | |
|---|---|
| 1 | C → B |
| 2 | B → C |
| 3 | {C, D} → A |
| 4 | {C, D} → {A, B} |
| 5 | {C, D} → B |
| 6 | {A, B, C} → D |

C → B **holds**
{C, D} → B **holds**
{A, B, C} → D **holds**
**The other three don't hold**

Q4- [1.5 marks] Given relation Enrol(<u>sid</u>, <u>cid</u>, cname), and assuming cid → cname, answer the following questions:

| sid | cid | cname |
|---|---|---|
| s1 | infs3 | Math |
| s1 | infs4 | DB |
| s2 | infs3 | Math |
| s3 | infs2 | ML |

| |
|---|
| 1) Give an example of insertion anomaly |
| 2) Give an example of deletion anomaly |
| 3) Give an example of modification anomaly |

1) [0.5 mark] Insert infs5, Distributed Systems without knowing the value of sid.
2) [0.5 mark] Deleting s3 removes all information about infs2.
3) [0.5 mark] Changing Math to Advanced Math in row 1, cid → cname doesn't hold anymore because of row 3 (s2). (don't have to talk about cid → cname explicitly).

Q5- [4 mark] Find the closure of A and all candidate keys of A(ABCDEFGH) given the following functional dependencies.
$A \rightarrow B$
$B \rightarrow C$
$C \rightarrow D$
$\{A, E\} \rightarrow F$
$F \rightarrow G$
$E \rightarrow A$

**{A}+ = ABCD**
**Candidate keys: EH**

Q6- [4 marks] You have a relation R(A, B, C, D, E) with the following functional dependencies:
A, B → {C, D, E}
A → C
B → D

a) Is R in BCNF?
b) In one sentence explain why it is or it is not BCNF?
c) If R is not in BCNF, decompose it into BCNF.

Answer:

**a) No**
**b) Because A is not a superkey and is on left hand side. The same about B. Any of them is OK.**
**c) ABE, AC, BD is the resulting decomposition. [3 marks, one for each table]**

**Q7**- [10 marks] Write the asked SQL queries using the below schema. *Remove duplicates* in all the requested queries,and *don't* include any unwanted attributes.

Product(pid, pname, brand)
ProductInstance(**pid**,validStartDate, validEndDate, price)
Customer(cid, cname)
Order(oid, **cid**, orderdate)
OrderItem(**oid, pid, validStartDate**, quantity)

Note: validStartDate is the date a price becomes valid. For a product, the current price is the price of a productinstance record where validEndDate is NULL (the price is still valid). When the price of a product changes, the validEndDate of the product is set to current datetime and a new record is added to ProductInstance where the validStartDate is set to current datetime and the validEndDate is set to NULL (means it is still valid).

A- [3 marks] Find the names of all the customers that have purchased any product between 19/04/2020 20:30 and 20/04/2020 14:30.

**SELECT DISTINCT cname FROM Order, Customer WHERE Order.cid=Customer.cid AND orderdate BETWEEN 19/04/2020 20:30 AND 20/04/2020 14:30.**
**Note: Students might use natural join, or use <>= operators instead of between keyword. Also they might use aliases for tables e.g. Order o.**

B- [3 marks] Find pair of product names (as in pname1, pname2) that are ordered together e.g. they both exist in a single order (ignore quantity) and the number of times they are ordered together (ignoring quantity) and sort them by the number of times they are ordered together in descending order.

Note: This is very similar to ``*other customers buy these items together*`` feature in online shops such as Amazon. Also in physical shops if these items are put close to each other, it will increase sale (e.g. pasta and tomato sauce).

```
SELECT p1.pname, p2.pname, count(*) FROM Product p1, OrderItem oi1, Product p2,
OrderItem oi2
WHERE
p1.pid=oi1.pid AND p2.pid=oi2.pid
AND
oi1.oid=oi2.oid
GROUP BY p1.pname, p2.pname
ORDER BY count(*) DESC
```

C- [4 marks] Find brand names whose total ordered quantity (the sum of all quantities of all products for that brand) is more than the average ordered quantity of all brands (find the total ordered quantity from each brand, then average them).

Guide:
Assume we have brand=LG with two products : TV and Phone
TV ordered 2 times each time with quantity 5.
Phone ordered 3 times each time with quantity 1.
Total ordered quantity from brand LG = (2 * 5) + (3 * 1) = 13
Now find this number for all brands, and find brands for which this number is more than the average of the same number among all brands.

```
SELECT DISTINCT p.brand FROM Product p, OrderItem oi
WHERE
p.pid=oi.pid
GROUP BY p.brand
HAVING
SUM(quantity) > (
SELECT AVG(q) FROM SELECT p2.brand, SUM(oi2.quantity) AS q FROM Product p2,
OrderItem oi2
WHERE
p2.pid=oi2.pid
GROUP BY p2.brand
)
```