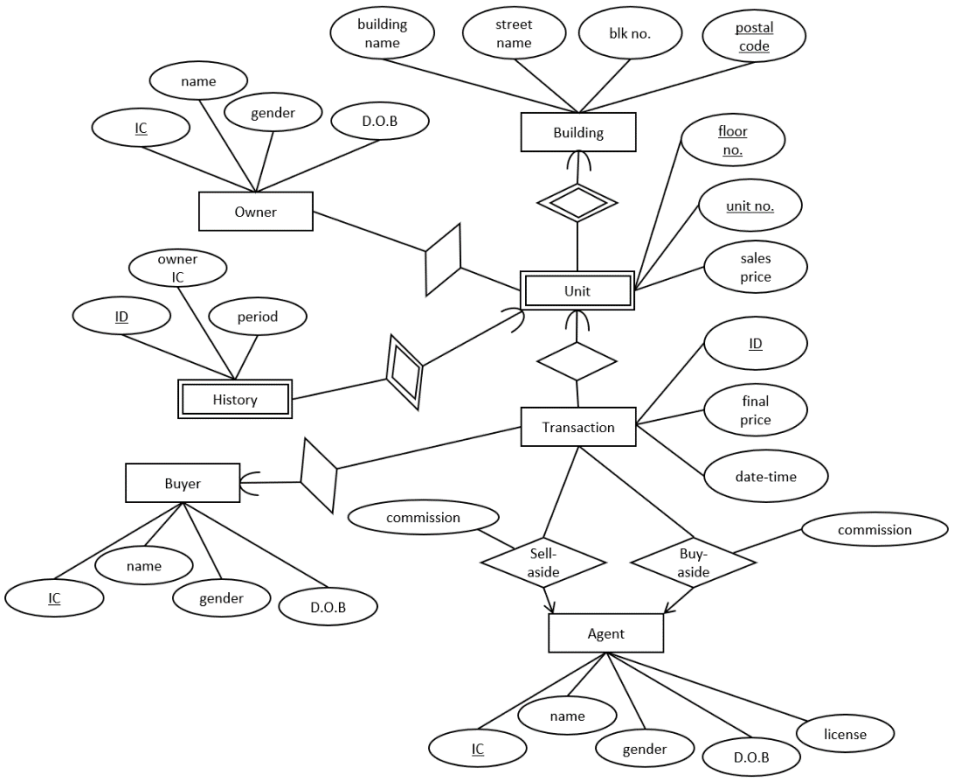
Solver: Chen Chongsong

1) a) i)



ii) Building (buildingName, streetName, blkNo., <u>postalCode</u>)
Unit (<u>postalCode</u>, <u>floorNo.</u>, <u>unitNo.</u>, salesPrice)
Owner (<u>IC</u>, name, gender, DOB)
Unit_Ownership (<u>postalCode</u>, <u>floorNo.</u>, <u>unitNo.</u>, <u>ownerIC</u>)
Buyer (<u>IC</u>, name, gender, DOB)
Agent (<u>IC</u>, name, gender, DOB, licenseNo.)
Transaction (<u>ID</u>, finalPrice, date-time, postalCode, floorNo., unitNo., buyerIC, sellAgentIC, buyAgentIC, sellAgentCommission, buyAgentCommission)

b) i)

$$R1 \coloneqq \gamma_{Category,MAX(Price)\rightarrow MaxPrice} Product$$
$$R2 \coloneqq \Pi_{PID}\sigma_{Product.Price=R1.MaxPrice}(Product \bowtie R1)$$

ii)

$$R1 \coloneqq \Pi_{\text{TID}\rightarrow\text{TID1,UID}\rightarrow\text{UID1}}\sigma_{PName=\text{"iPhone 8"}}(Transaction \bowtie Product)$$
$$R2 \coloneqq \Pi_{\text{TID1}\rightarrow\text{TID2,UID1}\rightarrow\text{UID2}}R1$$
$$R3 \coloneqq \sigma_{TID1>TID2}(R1 \times R2)$$
$$R4 \coloneqq \gamma_{(UID1,TID1,COUNT(TID2)\rightarrow NumPreviousTransactions)}R3$$
$$R5 \coloneqq \Pi_{UID1}\sigma_{NumPreviousTransaction=99}R4$$

iii)

$$R1 \coloneqq \Pi_{PID}\sigma_{Category=\text{"phone"}}Product$$
$$R2 \coloneqq \Pi_{\text{PID}}\sigma_{Category=\text{"laptop"}}Product$$
$$R3 \coloneqq \Pi_{UID,PID}Transaction$$

$$R4 \coloneqq (R3 \div R1) - (R3 \div R2)$$

2) a) Since F is not in the right-hand side of any FD, F must be contained in keys.
{A F}+ = {A F}
{B F}+ = {B F}
{C F}+ = {A B C F}
{D F}+ = {A B C D E F} "DF" is a key!
{E F}+ = {E F}

{A B F}+ = {A B F}
{A C F}+ = {A B C F}
{A E F}+ = {A E F}
{B C F}+ = {A B C F}
{B E F}+ = {A B C D E F} "BEF" is a key!
{C E F}+ = {A B C D E F} "CEF" is a key!
{A B C F}+ = {A B C F}

Therefore, 3 keys in total: DF, BEF, CEF

R is not in BCNF, since the left hand side of "D→AE" does not contain a key.
BCNF decomposition:

1. According to "D→AE", split R into R1(A, D, E), and R2(B, C, D, F).
   Now, in R1:
   {A}+ = {A}, {D}+ = {ADE}, {E}+ = {E}, {AE}+ = {AE},
   There is only one FD, ie. D→AE, related to R1, and the left-hand-side D is a key of R1, so R1 is in BCNF
2. From "C→A" and "AC→B", we can infer that "C→B".
   Therefore, we split R2 into R3(B, C) and R4(C, D, F).
3. From "D→AE" and "AD→C", we can infer that "D→C".
   Therefore, we split R4 into R5(C, D) and R6(D, F)

To summarise, R can be decomposed into ADE, BC, CD, DF.

Does that preserve all functional dependencies?
No, because only "D→AE" is preserved, and nothing else can be inferred from it.

b) R is not in 3NF, since for "D→AE", D is not a key, A is not contained in any key.
3NF decomposition:

1. Derive a minimal basis:
   a. S = {D→A, D→E, BE→C, BE→D, AD→C, AC→B, C→A}
   b. Remove "BE→C" since it is redundant.
      Now S = {D→A, D→E, BE→D, AD→C, AC→B, C→A}
   c. Simplify "AD→C" to "D→C", and "AC→B" to "C→B"
      Now S = {D→A, D→E, BE→D, D→C, C→B, C→A}
2. Combine FDs with the same LHS
   S becomes {D→AEC, BE→D, C→AB}
3. Create a table for each FD remained:
   R1(A, C, D, E) R2(B, D, E) R3(A, B, C)

4. Create a table R4(D, F) that contains a key
5. No redundant table to be removed

To summarise, R can be decomposed into ACDE, BDE, ABC, DF

3) a) i)
```
SELECT ISBN FROM Book
WHERE NoOfPages > (
        SELECT AVG(NoOfPages)
        FROM Book
) * 2;
```

ii)
```
WITH temporary AS (
        SELECT Author, COUNT(*) AS NoOfBooks
        FROM Book
        GROUP BY Author
)
SELECT t1.Author
FROM temporary AS t1
WHERE t1.NoOfBooks = (
        SELECT MAX(t2.NoOfBooks)
        FROM temporary AS t2
);
```

iii)
```
WITH BookPhilip AS (
        SELECT ISBN
        FROM Book
        WHERE Author = 'Philip S. Yu'
)
SELECT R.RDNR
FROM Reader AS R, Loan AS L, BookPhilip AS B
WHERE R.RDNR = L.ReaderNr AND L.ISBN = B.ISBN
GROUP BY R.RDNR
HAVING COUNT(DISTINCT B.ISBN) < (SELECT COUNT(*) FROM BookPhilip);
```

iv)
```
SELECT DISTINCT Surname
FROM Reader
WHERE City = 'New York';
```

b)
```
INSERT INTO Loan VALUES (
        SELECT RDNR FROM Reader
        WHERE Surname = 'Goh' AND Firstname = 'Andy',
        '123456',
        '4',
        '2017-12-31'
);
```
[As specified by Prof, any date would be OK]

4) a)
```
CREATE ASSERTION Q4_a CHECK (
        NOT EXISTS (
```

```
                SELECT W.number
                FROM Wards AS W, PatientInWard AS PIW, Patient AS P
                WHERE W.number = PIW.wardNumber AND PIW.pid = P.pid
                AND (P.gender = 'F' OR P.year <= 1957)
                GROUP BY W.number
                HAVING COUNT(P.pid) > W.number
        )
);
```

b)
```
CREATE TRIGGER Q4_b
AFTER INSERT ON R
REFERENCING NEW ROW AS NEW
FOR EACH ROW
WHEN EXISTS (
        SELECT * FROM R
        WHERE R.A = N.A AND R.B = N.C AND R.B <> N.B
)
BEGIN
        ROLLBACK;
END;
```

c)
```
CREATE VIEW UnLuckyDrinker AS
SELECT L.drinker
FROM Like AS L
GROUP BY L.drinker
HAVING COUNT(L.beer) > (
        SELECT COUNT(S.beer)
        FROM Frequent AS F, Sell AS S
        WHERE F.bar = S.bar
        AND F.drinker = L.drinker
        AND S.beer IN (
                SELECT L2.beer
                FROM Like AS L2
                WHERE L2.drinker = L.drinker
        )
);
```

The statement above compares the number of beers a person favours to the number of favoured beers a person can find in his/her frequent bar.
If you find that tricky, you can try the following statement: It checks whether each favoured beer is sold in his/her frequent bar.

```
CREATE VIEW UnLuckyDrinker AS
SELECT DISTINCT L.drinker
FROM Like AS L
WHERE L.beer NOT IN (
        SELECT S.beer
```

```
        FROM Frequent AS F, Sell AS S
        WHERE L.drinker = F.drinker AND F.bar = S.bar
);
```

d)
```
<results>
    <Applicants>
        <Applicant name=”A” appNum=”1”></Applicant>
        <Applicant name=”B” appNum=”2”></Applicant>
    </Applicants>
    <Choices>
        <Choice applicant=”1” code=”interesting” choiceNum=”10”
meritScore=”10”></Choice>
        <Choice applicant=”2” code=”exciting” choiceNum=”10”
meritScore=”10”></Choice>
    </Choices>
</results>
```

e) “interesting”
“exciting”

[This PYP is relatively difficult. Congratulations if you can work out most of the problems!]

--End of Answers--