Question 1

(1)Relational algebra expression:

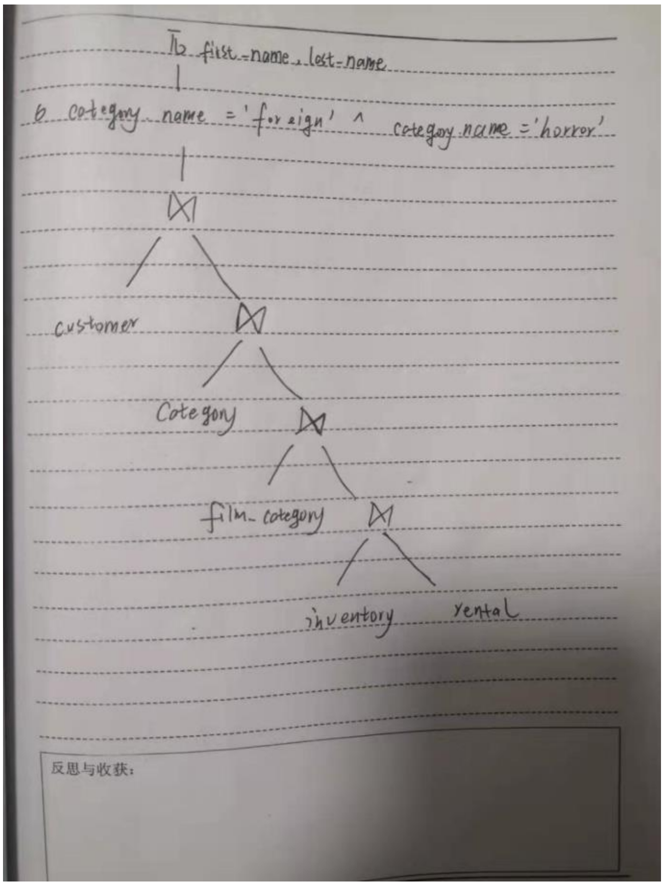
$\Pi_{first\_name,last\_name}$ ($\sigma_{Category.name='\ foreign'\ \wedge\ category.name='\ horror'}$

((((Customer $\bowtie$Rental) $\bowtie$Inventory) $\bowtie$Film) $\bowtie$Film_category))

(2) SQL statement:

```
select first_name as FIRST_NAME,last_name as Last_name from customer as c
join rental r on c.customer_id = r.customer_id
join inventory i on r.inventory_id = i.inventory_id
join film f on i.film_id = f.film_id
join film_category fc on f.film_id = fc.film_id
join category c2 on fc.category_id = c2.category_id
where c2.name=' foreign' or c2.name = ' horror'
group by first_name, last_name
```

(3) Expression tree:



Question 2

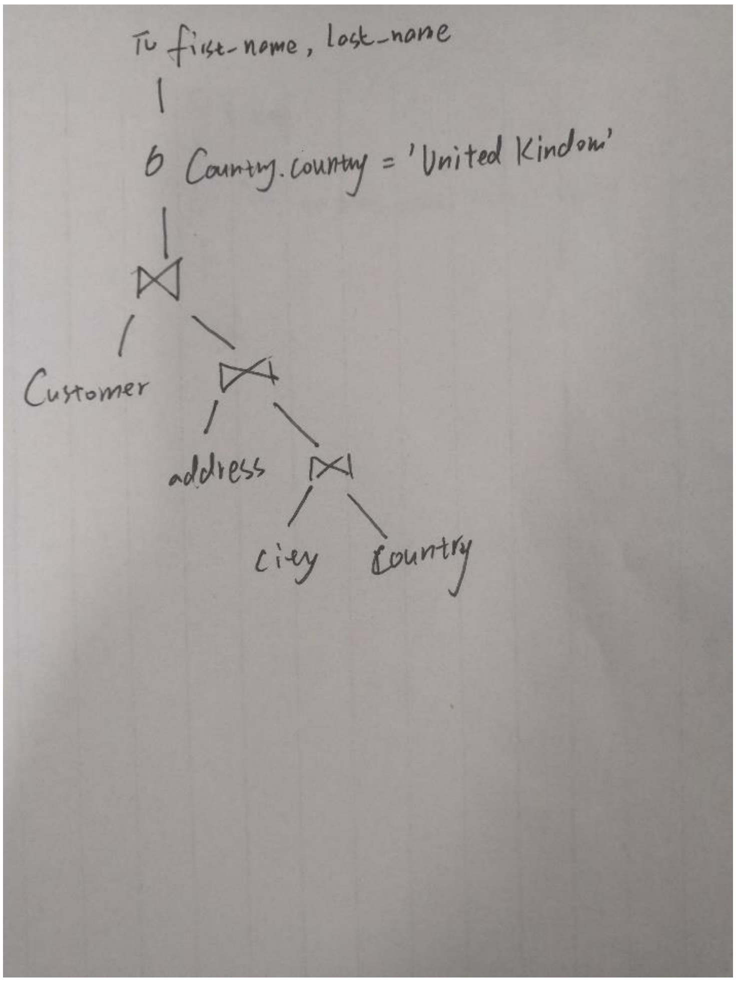
(1)Relational algebra expression:

Πfirst_name,lasrt_name (σ Country.country=’ United Kingdom’ (((Customer ⋈ address) ⋈city) ⋈country))

(2) SQL statement:

select first_name as FIRST_NAME,last_name as LAST_NAME from customer
as c
join address a on c.address_id = a.address_id
join city c2 on a.city_id = c2.city_id
join country c3 on c2.country_id = c3.country_id
where c3.country = 'United Kingdom'

(3) Expression tree:



Question 3

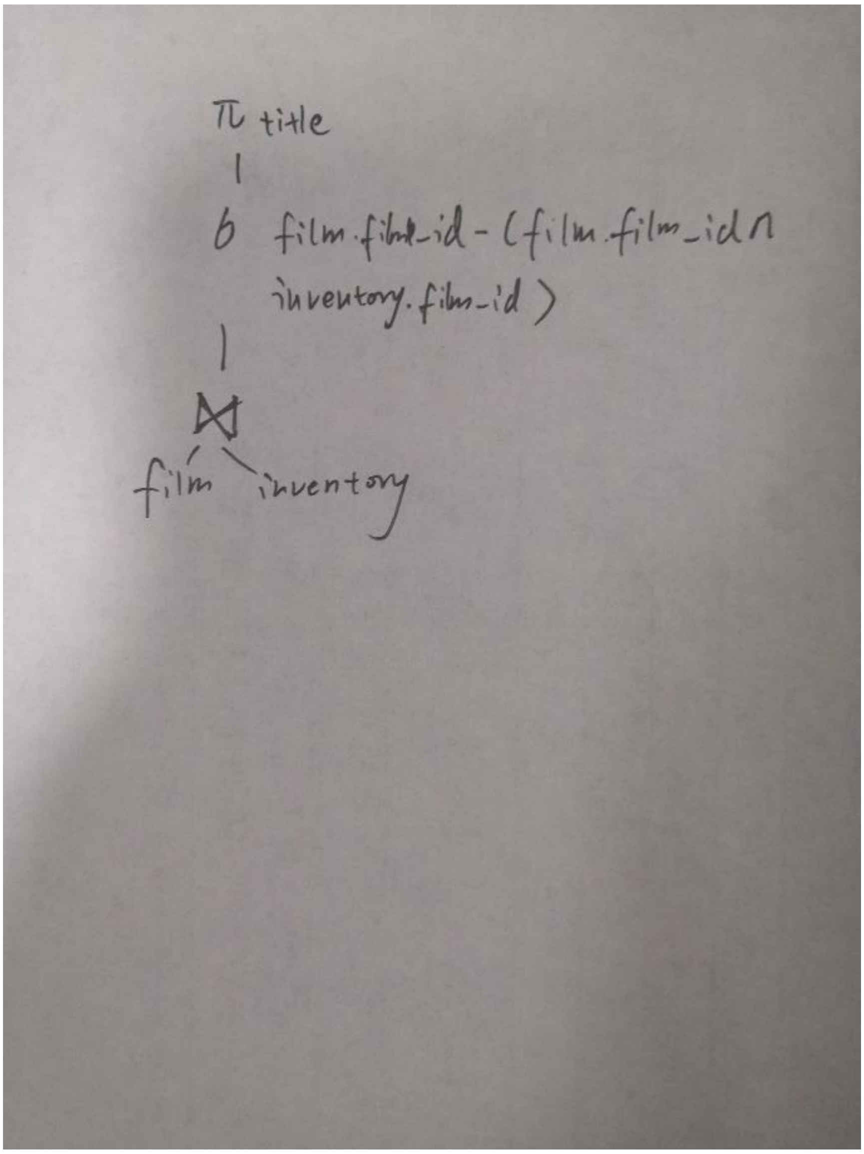
(1)Relational algebra expression:

Πtitle (σ (film.film_id- (film.film_id∩inventory.film_id)(film ⋈nventory)))

(2) SQL statement:

select f.title from film as f
join inventory i on f.film_id = i.film_id
where f.film_id not in(i.film_id)

(3) Expression tree:



Question 4

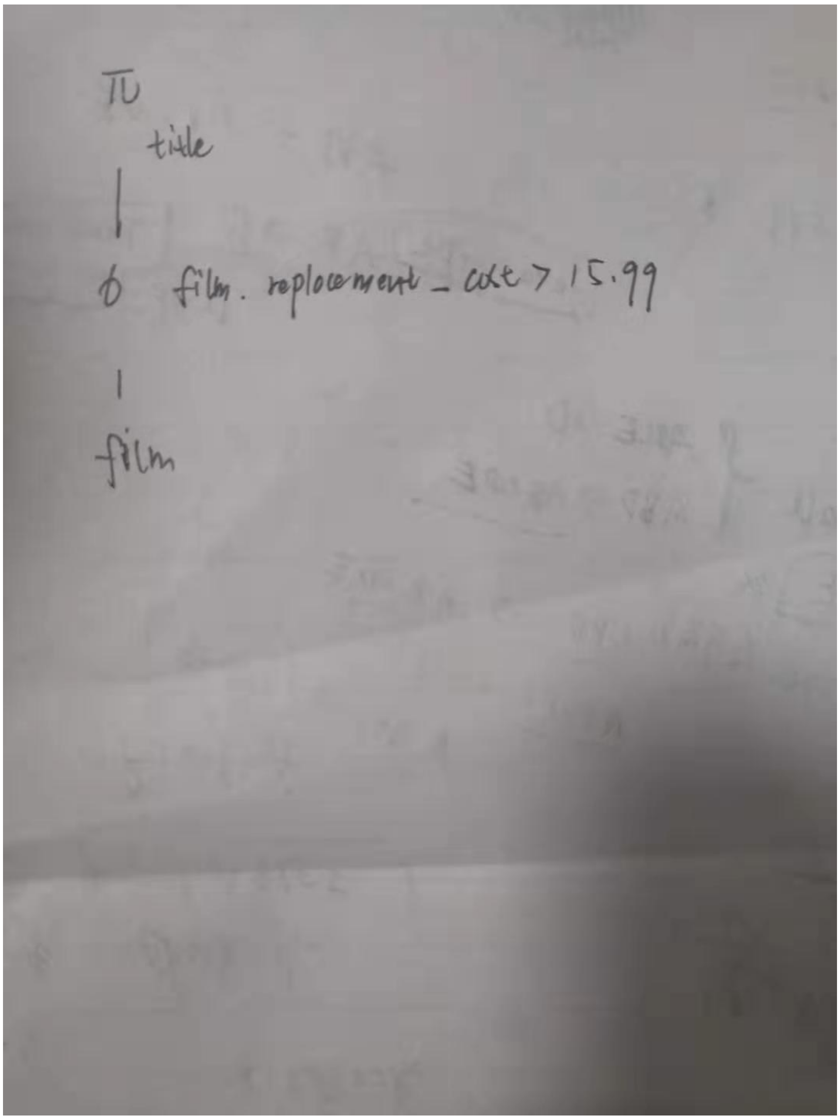
(1)Relational algebra expression:

Πtitle (σ film. replacement_cost >15.99 (Film))

(2) SQL statement:

select title as TITLE from film as f
where f.replacement_cost >15.99

(3) Expression tree:



Question 5

The natural connection takes the common attributes of the two relations A and B as the root, and the values of these common attributes in the two tables are spliced together as a new tuple, and the remaining parts that cannot be spliced are all discarded. This gives you a new relationship.

Natural joins are also a different operation. We can first make the Cartesian product of the A and B relations, then make the equality judgment based on the same attributes in the two relations, remove the unequal tuples, and finally simplify the attributes to get the result of the natural connection.

id	name
1	Aa
3	Ac
4	Ad
8	Ae

id	names
1	Ba
2	Bb
3	Bc
5	Be
6	Bf

Only two tables, Table_A, Table_B

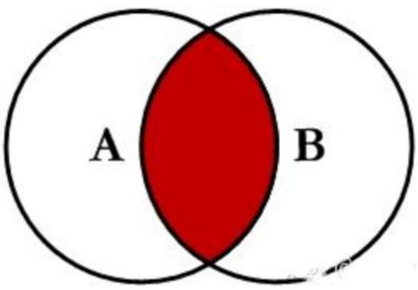
Note the ids of the two tables, A has1, 3, 4, 8； B has 1, 2, 3, 5, 6, and fields are name and names .The field data is in lowercase The letter is preceded by an A or B to make it easier to distinguish which table it belongs to.

Use the equal (=) operator in the join condition to compare the column values of the joined column, but it uses the select list to indicate the columns included in the query result set and removes the duplicate columns in the join table.

Query the result, note that the duplicate column has been deleted, the number of columns is only 3, which is also the difference from the equivalent connection

id	name	names
1	Aa	Ba
3	Ac	Bc

According to the condition, find the intersection of the data of Table A and Table B, but the field has been deduplicated, and the Venn diagram is



So the intersection is equal to the natural join.

My example:

In order to cooperate with the test, two tables (shuguo and weiguo) are specially built by me, and some test data are added, in which the characters of wuguo are repeatedly recorded You can see the data in the two tables in the screenshot below.

Table weiguo:

sanguo.public.weiguo [@localhost]

5 rows   Tx: Auto

<Filter criteria>

	id	name	age
1	1	caocao	24
2	2	DIANWEI	23
3	3	GUOJIA	22
4	4	SUNQUAN	24
5	5	ZHOUYU	24

Table shuguo:

<Filter criteria>

	id	name	age
1	1	LIUBEI	24
2	2	GUANYU	23
3	3	ZHANGFEI	22
4	4	SUNQUAN	24
5	5	ZHOUYU	24

Now I test them with the code of natural connection and intersection respectively.

natural join SQL statements:

```sql
select * from shuguo as s
natural join weiguo as w
order by id;
```

Results of natural join

	id	name	age
1	4	SUNQUAN	24
2	5	ZHOUYU	24

Intersection SQL statements:

```sql
select *
from shuguo intersect
select *
from weiguo
order by id;
```

Results of Intersection

	id	name	age
1	4	SUNQUAN	24
2	5	ZHOUYU	24

The results using natural join and intersection are obviously the same

In summary, you can clearly see that anything that can be achieved using intersection can be achieved by using natural joins-you need to prove that the result of the intersection of A and B is the same as the result of the natural connection of A and B

Question 6

Answer: NO

Natural join does not increase expressive power of relational algebra. Because it can be instead of the group of projection, selection and cross-product.

Assume the following two tables A and B.

Table A	
NUMBER	CITY
1	SHIJIAZHUANG
2	CHENGDU
3	BEIJING

Table B	
NUMBER	NAME
1	DUDU
3	MINMIN
5	XIAOXAIO

In the case of the natural join, the two tables are searched by using the following statement.

σ Table B ⋈ Table A.

In the case of the group of projection, selection and cross-product, the two tables are searched by using the following statement.

Π A.NUMBER, A.CITY, B.NAME (σ (A.NUMBER = 1 ^B.NUMBER = 3) (A ⋈ B))

The results of table C can be obtained through the above two methods. So, it can be concluded that Natural join does not increase expressive power of relational algebra.

Table C		
NUMBER	CITY	NAME
1	SHIJIAZHUANG	DUDU
3	BEIJING	MINMIN

Question 7

The answer is： NO
The Intersection operator does not increase expressive power to Relational Algebra.
Assume that there are two tables, table A and table B. And the Intersection method to get two intersect the parts.
The code is shown below: σ (A ∩ B)
At the same time, there are two different methods which are can get the same result.
The Natural Join: σ (A - (A - B)).
And Difference methods: σ (A ⋈ B).
The results obtained by the above three methods are the same. Since the statements using the intersection method can be replaced, it is concluded that the Intersection operator does not increase expressive power to Relational Algebra.

Question 8

The answer is：YES
The rename operator can increase expressive power.
Assume there are two tables, table A and table B. Through self-join, the rename operator can be used for various types of queries.
In order to know which table the ID belongs to, you need to use the Rename Operator method for all attribute columns. The specific code is shown below:
ρA(A.ID, A.Name)(A) ⋈ ρB(B.ID, B.Name)(B)
The following new code can be used to avoid duplicate attribute columns and A, B:
π(A.Name, B.Name) (σ (A.Name < B.Name) (ρA (A.ID, A.Name) (A) ⋈ ρB(B.ID, B.Name)(B))

Question 9

the maximum number of possible values for A is 30
Because B, C → D, B has two values, and C has three values. So D has a maximum of 6 values. In addition, D, E → A. Then D has 6 values, E has 5 values, D and E together determine A, then A can have 30 values at most.

Question 10

StudentID, ModuleID → ModuleMark:
The ModuleMark is determined by the StudentID, ModuleID. That is to say, when a student with a StudentID selects a module (this module uses ModuleID), it will finally get the marks of this module.(ModuleMark).When a student's StudentID, ModuleID is equal, then it must have an equivalent ModuleMark
ModuleID → Term
ModuleID determines Term. When two people belong to a certain ModuleID at the same time, then both of them must belong to this Term.

Question 11

The correct answer is ABE → C

11. $\because AB \to C \quad AE \to D \quad D \to B$.

(1) $\therefore \{DB\}$ X.

~~{DB~~

(2) $\{AB\}$

$\because AB \to C \therefore \{ABC\} \quad \therefore \cancel{B} AB \to D$ X

(3) $\{ABE\}$

$\because AB \to C \therefore \{ABEC\}$.

$AE \to D \therefore \{ABECD\} \quad \therefore ABE \to C$.

(4) $\{ABC\}$

$\because AB \to C \therefore \{ABC\} \quad \therefore \cancel{C} \therefore ABC \to DE$ X.

Question 12

The correct answer is BCA

12.

∵ E→D, DA→B, D→B, BA→E.

①. {CE}

∵ E→D : {CED}

D→B : {CEDB} ×.

Becouse it does not include all atteributes, it is not the key

②. {BE}

∵ E→D : {BED} ×

~~∵ D→B~~

Because it does not include all attributes, it is not the key.

③. {BEA}

∵ E→D : {BEAD} ×

~~∵ DA→B~~

Because it does not include all attributes, it is not the key.

④. {BCA}

∵ BA→E : {BCAE} ✓

∵ E→D : {BCADE}.

Becouse it includes all attributes., it is the key.

Question 13

The correct answer is (6, 2, 8, 4)

Because according to the rules and definitions of functional dependencies and A → B and B → D we can know: A can decide B, and B can decide D at the same time. And because in relation R (A, B, C, D) currently has only the tuple (1,2,3,4). So (1,2,3,4) is legal.

Then we use the exclusion method here, like (1, 3, 4, 5), we know,when A = 1, B must be 2. So this answer is wrong.

Similarly, when (9, 8, 3, 7), Like (1, 3, 4, 5), it is incorrect..
We finally look at (6, 2, 8, 4), because B → D, so when B = 2, D must be 4, so it may be legal.

Question 14

Answer: (B, C), (B,D), (D,A)
Step1:
1) Check the left side is a key or not.
B → C :
Using B → C get {B, C}
The closure of B: {B, C} does not contain all the attyributes of the relation.
So, B is not a key and B → C satisfies BCNF.
D → A:
Using D → A get {D, A}
The closure of D: {D, A} does not contain all the attyributes of the relation.
So, D is not a key and D → A satisfies BCNF.
BA → D:
Using BA → D get {B, A, D} and using B → C get {B, A, D, C}
The closure of BA: { B, A, D, C } contain all the attyributes of the relation.
So, BA is a key and D → A not violates BCNF.
CD →B:
Using CD → B get {C, D, B} and using D → A get {C, D, B, A}
The closure of CD: { C, D, B, A } contain all the attyributes of the relation.
So,CD is a key and CD → B not violates BCNF.
2) Pick B → C which is violates BCNF.
Step 2: Start to Decompose R
S1 (B, C): The functional dependency B →C where the B is a key, S1 satisfies
BCNF.
S2 (A, B, D):
BA → D: Closure of BA is {B, A, D}, BA is a key. BA → D not satisfies BCNF.D → A: Closure of D is {D, A}, D is not a key. D → A satisfies BCNF.
Step 3: Decompose S2
S2.1 (B,D)
S2.2 (D,A)
So, the result of decompose R is S1 (B, C), S2.1(B,D), S2.2(D,A).

Question 15

The correct answer is ABCD → E

15.

solution:

progress:

∵ ABD is a key for R. ∴ the following dependencies must be avaiable:

ABD->CE ,

ABCD->E ,

ABDE->C ,

∴ only NO.4 ~~is contains~~ is included in the available dependencies,

so ABDE->C is the answer.

Question 16

The answer is {E,C},{D, A, B}, {D, A, E}

Accorading to E → C and DA → B, the compute key for R (A, D, E) can be get.

Decompsotion start with E→ C:
1. {E,C}:
E → C can be applied in {E,C} relation and E on the left side is a key. {E,C} satisfy BCNF.
2. {E, A, B, D}:
DA → B can be applied in {E, A, B, D} relation and DA on the left side is not a key.
Decompsotion start with DA → B:
1. {D, A, B}:
DA → B can be applied in {D, A, B} relation and DA on the left side is a key. {D, A, B} satisfy BCNF.
2. {D, A, E}:
{D, A, E} is key for R, {D, A, E}satisfy BCNF.
The decomposition of R into BCNF will be {E,C},{D, A, B}, {D, A, E}