7.

There are three foreign keys :-

1. Attribute serial_no of relation OPTION that references relation CAR
2. The Attribute salesperson_id of relation SALE refers that relation SALESPERSON
3. The attribute serial_no of relation SALE that references relation CAR

SALE

Saleperson_id	Serial_no	Date	Sale_price
1	99	24-1-20	100
1	100	25-1-20	90
2	50	1-1-20	150

SALESPERSON

Saleperson_id	name	phone
1	RK	9886341416
2	PK	7286009239

If we try to

INSERT INTO SALE VALUES (3, 44, 1-2-20, 200)

This is violation of referential integrity constraints since salesperson_id 3 not exists

INSERT INTO SALE VALUES (2, 66, 2-2-20, 400)

This is not violating because salesperson_id 2 exists

10.

a) one possible set of operations for the following update is the following :

INSERT < FNO, LNO, DT, SEAT_NO, CUST_NAME, CUST_PHONE > into SEAT_RESERVATION;

modify the LEG_INSTANCE TUPLE with the condition: (FLIGHT_NUMBER = FNO AND LEG_NUMBER = LNO AND DATE = DT) by setting NUMBER_OF_AVAILABLE_SEATS = NUMBER_OF_AVAILABLE_SEATS - 1 ;

These operations should be repeated for each LEG of the fight on which a reservation is made. This assumes that the reservation has only one seat. More complex operations will be needed for a more realistic reservation that may reserve several seats at once

b) we would check that NUMBER_OF_AVAILABLE_SEATS on each LEG_INSTANCE of flight is greater than 1 before doing any reservation, and that the SEAT_NUMBER being reserved in SEAT_RESERVATION is available

c) The INSERT operation into SEAT_RESERVATION will check all the key, entity integrity, referential integrity constraints for the relation. The check that NUMBER_OF_AVAILABE_SEATS on each LEG_INSTANCE of the flight is greater than 1 does not fall into any of the above types of constraints (It is a general semantic integrity constraint)

19003118

12. a) Names of parts that cost less than 20.00

SQL> SELECT * FROM PARTS WHERE Price < 20;

Algebraic expression

$$temp1 \leftarrow \sigma_{Price < 20}(PARTS)$$

$$Results \leftarrow \pi_{Pno, Pname, Qoh, Price, QLevel}(temp1)$$

b) Names and cities of employees who have taken olders for parts costing more than 50.00

SELECT Ename, City FROM EMPLOYEES e, ZIP_CODES z, ORDES o, ODETAIL o1, PARTS p WHERE p.Price > 50 AND p.Pno = o1.Pno AND o.Ono = o1.Ono AND e.Eno = o.Eno AND e.zip = z.zip ;

g) Names of customers who have placed exactly two olders

SELECT Cname, count(Ono) as total FROM CUSTOMERS c, ORDERS o WHERE c.Cno = o.Ono AND total = 2 ;

c) pairs of customer number values of customers who live in the same ZIP code

```
SELECT c.cno, c1.cno FROM customers c, CUSTOMERS c1
WHERE c.zip = c1.zip AND c.cno < c1.cno ;
```

d) Names of customers who ordered parts from employees living in wichita

```
SELECT DISTINCT cname FROM CUSTOMERS c, ORDERS o,
EMPLOYEES e, ZIP_CODES z. WHERE c.cno = o.ono AND
c.zip = z.zip AND z.city = 'wichita' ;
```

e) names of customers who have ordered parts costing less than 20

```
SELECT DISTINCT cname FROM CUSTOMERS c, PARTS p,
ODETAILS o, ORDERS o1 WHERE p.price < 20
AND p.pno = o.pno AND o.ono = o1.ono AND
c.cno = o1.cno ;
```

f) Names of customers who have not placed any order

```
SELECT cname, count(ono) as total FROM
CUSTOMERS c, ORDERS o. WHERE c.cno = o.ono
AND total = 0 ;
```