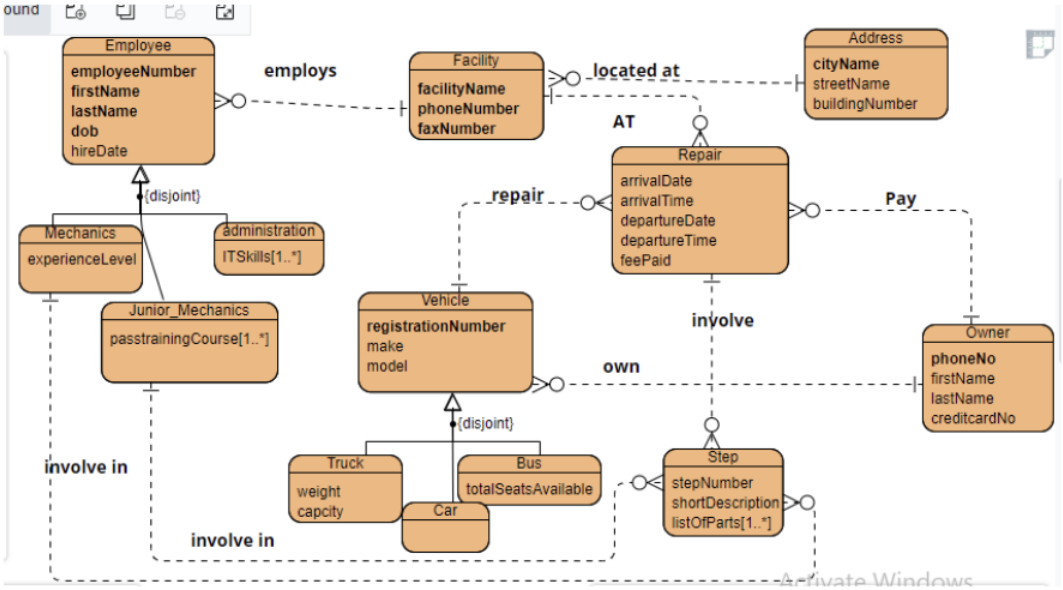
ERD:

Relationship between entities:

- 1. There are 1:M relationship between Facility entity and Employee entity
- 2. There are M:1 relationship between Facility entity and Address entity
- 3. There are 1:M relationship between Owner entity and Repair entity
- 4. There are M:1 relationship between Repair entity and Facility entity
- 5. There are 1:M relationship between Vehicle entity and Repair entity
- 6. There are M:1 relationship between Vehicle entity and Owner entity
- 7. There are M:1 relationship between Step entity and Repair entity
- 8. There are M:1 relationship between Step entity and Junior mechanic entity
- 9. There are 1:M relationship between Mechanic entity and Step entity



第二题：

Below are the steps to be followed to convert the given ER diagram to logical model and list relations from the diagram:

Step 1: Tables from entities:

The tables from the ERD will be as below:

1. EMPLOYEE: EMPLOYEE has next two generalized entities that are derived from EMPLOYEE i.e. subset of EMPLOYEE

2. DRIVER

3. MECHANIC: Mechanic employee has many experiences records

4. VEHICLE: Usedin trip

5. TRIP: The trip details to be stored in this

6. LEG: Trip has many departure and destinations as leg

Step 2: Multiple valued attribute experience in MECHANIC:

The relational database will not allow multiple values for single mechanic, thus the experience needs to be removed and taken to the other table as below:

MECHANIC_EXPERIENCE

Step 3: Relationships to be solved by references

1. EMPLOYEE is super entity of DRIVER and MECHANIC, thus the sub entities will have primary key which is foreign key from its parent entity i.e. empnum.

2. VEHICEL is owned by one MECHANIC, whereas a MECHANIC can own many VEHICLES, thus VEHICLE will store foreign key to its owner mechanic.

3. DRIVER drives many vehcles and a vehicle is driven by many drivers on different trips having trip date.

Thus this relationship will have TRIP, DRIVER and VEHICLE references in it under TRIP.

4. TRIP has none or many legs and LEG belongs to one TRIP, thus foreign key of trip is to be added in LEG.

Step 4: Final attributes in entities:

1. EMPLOYEE(empnum, fname, lname, dob)

2. DRIVER(driver_empnum(fk), licnum, status): licnum will be unique

3. MECHANIC(mech_empnum, qualnum): qualnum will be unique

4. VEHICLE(rego, capacity,weight, owner_mech_empnum(fk))

5. TRIP(vehicle_rego(fk), driver_empnum(fk), tdate) : Trip has dotted line with VEHCILE and DRIVER, which means a string relationship and

TRIP makes its primary key from other VEHILE and DRIVER entity's primary key.

6. LEG(LegID, departure, destination, vehicle_rego(fk), driver_empnum(fk), tdate(fk)

Step 4: Relational tables with keys:

Final list of relations is as below:

1. EMPLOYEE(**empnum**, fname, lname, dob)

Primary key: empnum

Foreign key:NA

2. DRIVER(**driver_empnum**, licnum, status)

Primary key: driver_empnum

Foreign key: driver_empnum from EMPLOYEE

3. MECHANIC(**mech_empnum**, qualnum)

Primary key: mech_empnum

Foreign key: mech_empnum from EMPLOYEE

4. VEHICLE(**rego**, capacity,weight, owner_mech_empnum)

Primary key: rego

Foreign key:owner_mech_empnum from MECHANIC

5. TRIP(**vehicle_rego, driver_empnum, tdate**)

Primary key: vehicle_rego, driver_empnum, tdate

Foreign key: Vehicle_Rego from VEHICLE and driver_empnum from DRIVER

6. LEG(**LegID**, departure, destination, vehicle_rego, driver_empnum, tdate)

Primary key: LegID

Foreign key: (vehicle_rego, driver_empnum, tdate) from TRIP

7. MECHANIC_EXPERIENCE(**mech_empnum, experience**)

Primary key: (mech_empnum, experience)

Foreign key: mech_empnum from MECHANIC

**Subset method is used in generalizing EMPLOYEE -> DRIVER and MECHANIC

第三题

第四题：

1.

```
INSERT INTO ORDERS VALUES (777, '007', '2021-07-19')
INSERT INTO ORDER_DETAILS VALUES(777, 'Golden Bolt', 22,
0.44),

(777, 'Silver Screw', 5, 0.22)



--We need to insert data first into orders table then
only we can insert data into order_details table due to
foreign key constraint.
--1ST query insert the data into Orders table.
--2nd query inserts the date into order_details table.
```

2.

```
DELETE FROM ORDER_DETAILS WHERE ORDER_ID = 777
DELETE FROM ORDERS WHERE ORDER_ID = 777
```

```
--We need to delete data from child table first then only
we can delete data from parent table due to foreign key
constraint.
-- first query deletes data from order_details table.
--second query deletes data from orders table.
```

3.

```
ALTER TABLE ORDER_DETAILS ADD CONSTRAINT FK_01 FOREIGN
KEY(PRODUCT_NAME) REFERENCES PRODUCTS(PRODUCT_NAME) ON
UPDATE CASCADE
UPDATE PRODUCTS SET PRODUCT_NAME = 'PLATINUM BOLT' WHERE
PRODUCT_NAME = 'Golden bolt'


--We can not simply update a foreign key because it will
affect the integrity contraints of foreign key.
--first query alters table to add a ON UPDATE CASCADE
constraint to the foreign key which means any update done
on parent key will automatically updates the foreign key
column in child table.
-- last query updates the column in parent table and
column in child table gets auto updated.
```

4.

```
SELECT * INTO OLD_ORDERS FROM ORDERS WHERE
YEAR(ORDER_DATE) < 2000
SELECT * INTO OLD_ORDER_DETAILS FROM ORDER_DETAILS WHERE
ORDER_ID IN (SELECT ORDER_ID FROM OLD_ORDERS)
DELETE FROM ORDER_DETAILS WHERE ORDER_ID IN (SELECT
ORDER_ID FROM OLD_ORDER_DETAILS)
DELETE FROM ORDERS WHERE ORDER_ID IN (SELECT ORDER_ID
FROM OLD_ORDERS)


--We have to first insert data into parent table then
only we can insert data into child table.
--For deletion it is reversed we need to delete data from
child table then only we can delete data from parent
table.
```

第五题

- **Find a customer code (CUSTOMER_CODE) and company name (COMPANY_NAME) of all customers who submitted at least one order in 2019. (2 marks)**

Answer:

```
Select cus.CUSTOMER_CODE, cus.COMPANY_NAME from CUSTOMER
cus INNER JOIN ORDERS ord  on
cus.CUSTOMER_CODE=ord.Customer_CODE where
YEAR(ord.ORDER_DATE) = '2019'
```

Explanation: To get matched record from two table, inner join gets implemented and with Year function it is checked whether the year is of 2019.

- **Find a customer code (CUSTOMER_CODE) and company name (COMPANY_NAME) of all customers who submitted no orders in 2019. (2 marks)**

Answer:

```
Select cus.CUSTOMER_CODE, cus.COMPANY_NAME from CUSTOMER
cus INNER JOIN ORDERS ord  on
cus.CUSTOMER_CODE=ord.Customer_CODE where
YEAR(ord.ORDER_DATE) <> '2019'
```

Explanation: To get matched record from two table, inner join gets implemented and with Year function it is checked whether the year is not (i.e., <> operator used) 2019.

- **Find a customer code (CUSTOMER_CODE) and company name (COMPANY_NAME) of all customers who submitted at least 50 orders in 2019. (2 marks)**

Answer:

```
Select cus.CUSTOMER_CODE, cus.COMPANY_NAME, COUNT(*) as
count from CUSTOMER cus INNER JOIN ORDERS ord  on
cus.CUSTOMER_CODE=ord.Customer_CODE  where
YEAR(ord.ORDER_DATE) = '2019' GROUP BY ord.CUSTOMER_CODE
HAVING COUNT(*) > 50
```

Explanation: In above as explained previous functions are used apart from that here having and groupby are used additionally. Having is used to get the

count of the rows and in above query it checked whetehr the count is greater than 50 and group by groups same customers in a group.

- **Find a unique order identifier (ORDER_ID) of all orders that included both products Golden Bolts and Silver Screws. (2 marks)**

Answer:

```
Select DISTICT ORDER_ID from ORDER_DETAIL where
PRODUCT_NAME in ('Golden Bolts, Silver Screws')
```

Explanation: To get unique ID's DISTINCT Function is used and product name is checked using in operator.

- **Find a company name (COMPANY_NAME) of all customers whose company name starts from a letter X and who did not provide information about their address (CITY and COUNTRY).**

Answer:

```
Select * from CUSTOMER WHERE COMPANY_NAME LIKE 'X%' and
CITY IS NULL and COUNTRY IS NULL
```

Explanation: % sign provide as a wildcard. In given requirement want to find all records with company nema starts with X while giving 'X%' it will retuen all company name starting with x and IS NUll retirns all values which are empty (i.e., NOt given by customers),