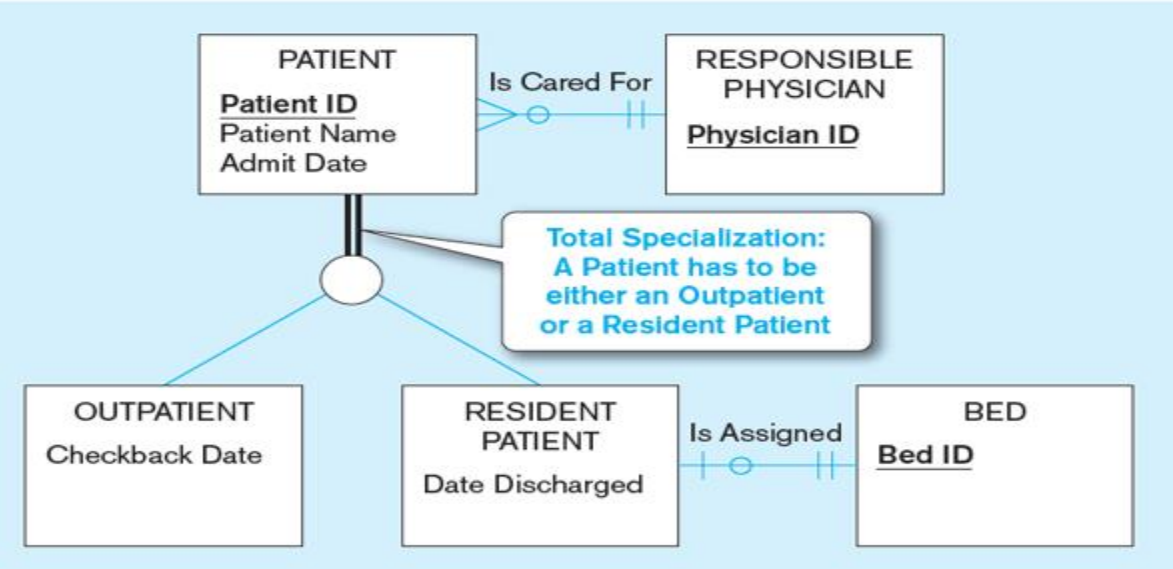
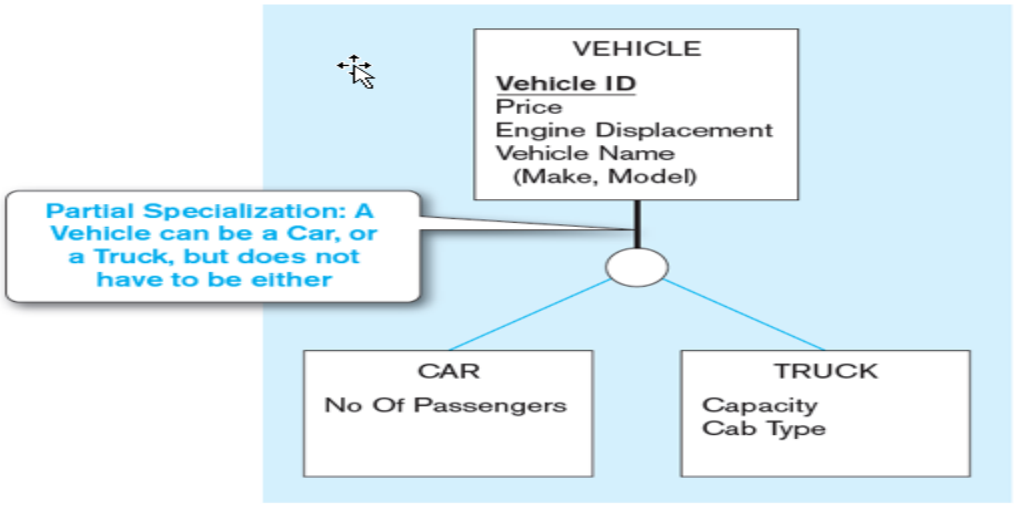
**A) Answer following questions briefly [15 points]**

**i) Difference between TOTAL and PARTIAL completeness constraint rules for super-type and sub-type entity**

**TOTAL completeness constraint:**
Instance of a super-type must also be a member of at least one subtype




**PARTIAL completeness constraint:**
Instance of a super-type may not be a member of any subtype.



**ii) Difference between TRUCATE, DELETE, and DROP SQL commands**

TRUNCTE: Is a DDL command that deletes all rows of the table. It is self-commit and once table is truncated, it cannot be rollback. Truncate applies to entire table and NO filter condition can be specified. After the table is truncated, the table structure remains in database with NO record.

DELETE: Is a DML command that deletes records from the table for a given criteria (condition). Delete requires COMMIT or ROLLBACK to confirm or undo the changes.

DROP: Is a DDL command. It is self-commit and once table is dropped, it cannot be rollback. DROP applies to entire table and NO filter condition can be specified. After the table is dropped, both structure and data are removed from the database.

**iii) Explain different delete rules for referential integrity constraint.**

**Delete rules for referential integrity constraints.**

**Restrict–**don't allow delete of "parent" side rows if related rows exist in "dependent" side

**Cascade**–automatically delete "dependent" side rows that correspond with the "parent" side row to be deleted

**Set-to-Null**–set the foreign key value in the dependent side rows to null

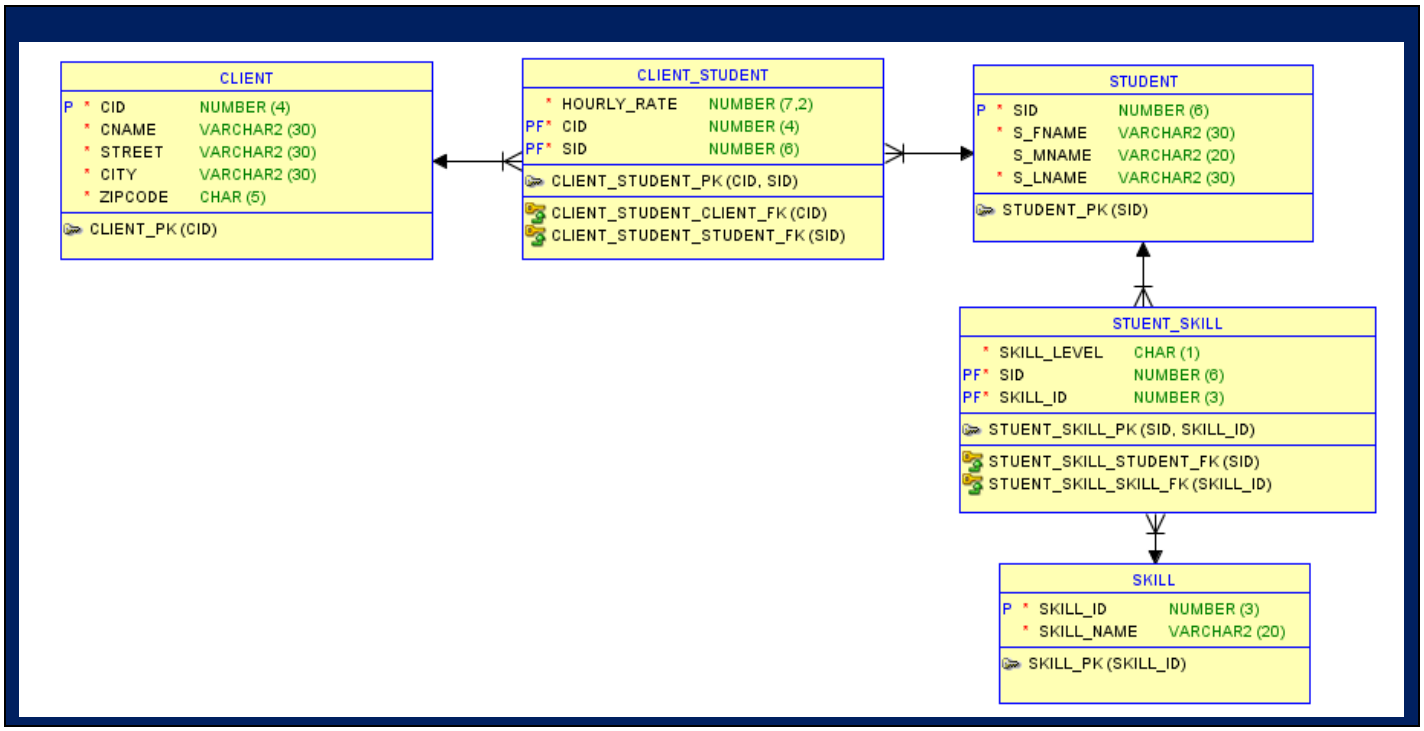
**B) Draw ERD (relational model) for following problem set [40 points]**

**Consider a consulting firm, Worth Work Inc.(WWI) with following entities and attributes.**

**CLIENT (cID, cName, cAddress)**
**STUDENTS (sID, sName, sSkill)**

**WWI is a consulting company that provides university students opportunity to work for it client companies. WWI keeps record of potential students and their skills. WWI prefers students with multiple skills in order to place them effectively at client projects. Students are being paid on hourly rate and that can vary client to client and student to student. Each student will have at least one work opportunities. Assume that students can have many skills such as Database principles, Database management, Project management, Web enabled databases, Data analytics etc. WWI also intends to record skill level (H for High, M for Medium, and L for Low) for each skill that students have.**

**Draw ERD (Relational Model) with proper relationships among entities by resolving any composite or multivalued attributes. Within ERD, for each entity list all attributes and their datatypes and size, mandatory/optional, primary key and foreign key wherever applicable. Cleary state assumptions you have made (if any) for the database designs.**



**C) For ERD in section C, write following SQL DDL commands [15 points]**

**i) To create table STUDENT with Primary key.**

```
CREATE TABLE student (
    sid            NUMBER(6) NOT NULL,
    s_fname    VARCHAR2(30) NOT NULL,
    s_mname   VARCHAR2(20),
    s_lname    VARCHAR2(30) NOT NULL
);

ALTER TABLE student ADD CONSTRAINT student_pk
PRIMARY KEY ( sid );
```

**ii) To add a constraint to enforce that field cID in COURSE table has minimum 4 digits**

```
ALTER TABLE course ADD CONSTRAINT  check_course_cid
CHECK ( cid >=1000)
```

**iii) To add foreign key for associate entities, if any**

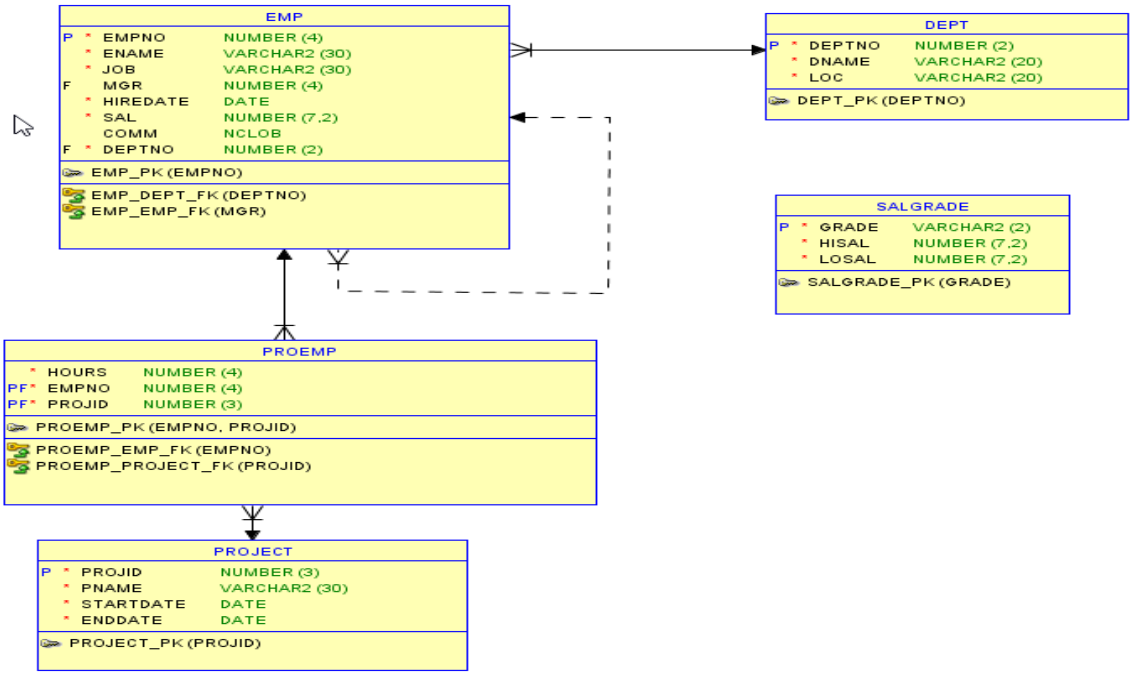
```
ALTER TABLE client_student
   ADD CONSTRAINT client_student_client_fk FOREIGN KEY ( cid )
      REFERENCES client ( cid );

ALTER TABLE client_student
   ADD CONSTRAINT client_student_student_fk FOREIGN KEY ( sid )
      REFERENCES student ( sid );

ALTER TABLE stuent_skill
   ADD CONSTRAINT stuent_skill_skill_fk FOREIGN KEY ( skill_id )
      REFERENCES skill ( skill_id );

ALTER TABLE stuent_skill
   ADD CONSTRAINT stuent_skill_student_fk FOREIGN KEY ( sid )
   REFERENCES student ( sid );
```

**D) Consider following Relational Model and answer questions [15 points]**




i) **List name of managers and number of employees that each manager is managing. Use appropriate use of column alias and list the result in descending order of number of employees that managers are managing**

```
SELECT b.ename "Manager Name",
            count(*) "Number of Emploees managing"
FROM ap_emp a JOIN ap_emp b ON a.mgr=b.empno
GROUP by  b.ename
ORDER BY 2 DESC;
```

ii) **Write SQL to retrieve employee's number, name, salary, and number of years (round number with no decimal points) they are working as of today. Use proper heading for number of years in result set. Exclude employees who are not eligible for receiving commission. Display salary with currency sign '$' and heading as 'Monthly Salary in USD'. Arrange the result set in descending order of seniority of employees.**

```
SELECT empno, ename,
            to_char(sal,'$99999.99') "Monthly Salary in USD",
round((sysdate-hiredate)/365,0) " Number of Years of Seniority"
FROM ap_emp
WHERE comm IS NOT NULL
ORDER BY 4 DESC;
```

iii) **List employee name, project name and number of hours worked on project. Use appropriate column alias and arrange result in order of number of hours for projects.**

```
SELECT a.ename "Employee Name" ,c.pname "Project Name",
            b.hours "Number of project hours"
FROM ap_emp a JOIN ap_proemp b ON a.empno=b.empno JOIN
          ap_project c ON b.projid=c.projid
ORDER BY 3;
```

iv) **List total salary of each department, only for those departments which total salary is higher than the total salary of employees working in SALES department.**

```
SELECT deptno,sum(sal) “Total Salary”
FROM ap_emp
GROUP BY deptno
HAVING sum(sal) > (select sum(sal) from ap_emp where
             deptno=(select deptno from ap_dept where
            dname='SALES'))
;
```

v) **List name of manager if they have any employee that not assigned any project. List name of such employee along with their managers name.**

```
SELECT  b.ename "Manager Name" , a.ename "Employee Name"
FROM ap_emp a JOIN ap_emp b ON a.mgr=b.empno
WHERE a.empno NOT IN (select empno from ap_proemp)
;
```