SQL Queries on Multiple Tables

```
SQL DDL Script
-------------------
-- MySQL Workbench Forward Engineering


-- -----------------------------------------------------
-- Schema hex_april_collegedb
-- -----------------------------------------------------
-- -----------------------------------------------------
-- Schema hex_april_collegedb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `hex_april_collegedb` DEFAULT CHARACTER SET utf8 ;
USE `hex_april_collegedb` ;

-- -----------------------------------------------------
-- Table `hex_april_collegedb`.`address`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `hex_april_collegedb`.`address` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `city` VARCHAR(255) NOT NULL,
  `state` VARCHAR(255) NULL,
  `country` VARCHAR(255) NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `hex_april_collegedb`.`student`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `hex_april_collegedb`.`student` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `email` VARCHAR(255) NOT NULL,
  `address_id` INT NOT NULL,
  PRIMARY KEY (`id`, `address_id`),
  INDEX `fk_student_address_idx` (`address_id` ASC)  ,
  CONSTRAINT `fk_student_address`
    FOREIGN KEY (`address_id`)
    REFERENCES `hex_april_collegedb`.`address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `hex_april_collegedb`.`department`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `hex_april_collegedb`.`department` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `hex_april_collegedb`.`course`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `hex_april_collegedb`.`course` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `credits` INT NULL,
  `fee` DOUBLE NULL DEFAULT 0,
  `department_id` INT NOT NULL,
  PRIMARY KEY (`id`, `department_id`),
  INDEX `fk_course_department1_idx` (`department_id` ASC)  ,
  CONSTRAINT `fk_course_department1`
    FOREIGN KEY (`department_id`)
    REFERENCES `hex_april_collegedb`.`department` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `hex_april_collegedb`.`student_course`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `hex_april_collegedb`.`student_course` (
  `student_id` INT NULL,
  `course_id` INT NULL,
  `id` INT NOT NULL AUTO_INCREMENT,
  `date_of_enrollment` DATE NULL,
  `discount` DOUBLE NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_student_has_course_course1_idx` (`course_id` ASC)  ,
  INDEX `fk_student_has_course_student1_idx` (`student_id` ASC)  ,
  CONSTRAINT `fk_student_has_course_student1`
    FOREIGN KEY (`student_id`)
    REFERENCES `hex_april_collegedb`.`student` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_student_has_course_course1`
    FOREIGN KEY (`course_id`)
    REFERENCES `hex_april_collegedb`.`course` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Queries with Insertions

```
use hex_april_collegedb;
show tables;

insert into address(city,state,country)
values
('mumbai','MS','India'),
('chennai','TN','India'),
('bhopal','MP','India'),
('delhi','Delhi','India'),
('pune','MS','India');

insert into student(name,email,address_id) values
('harry potter','harry@hogwards.com',1),
('ronald weasley','ronald@hogwards.com',1),
('hermione granger','hermione@hogwards.com',2),
('draco malfoy','draco@hogwards.com',3),
('ginni weasley','ginni@hogwards.com',4),
('neville longbottom','neville@hogwards.com',5);

insert into department(name) values
('IT'),('Dev'),('Testing');

insert into course(name,credits,fee,department_id)
values
```

```
('Java Programming', 120, 12000,2),
('Python Programming', 80, 8000,2),
('Selenium Admin', 60, 7000,3),
('Network Admin', 70, 8000,1);

-- date format must be yyyy-mm-dd and it shd go in DB as a string
insert into student_course(student_id,course_id,date_of_enrollment,discount)
values
(1,1,'2024-02-01',10),
(1,2,'2024-02-10',20),
(2,2,'2024-02-10',10),
(3,1,'2024-02-01',10),
(3,2,'2024-02-01',10),
(3,4,'2024-02-01',10),
(4,4,'2024-02-05',5),
(5,3,'2024-03-01',5);

-- Manual Mapping
/*
Q1. Display all students that are living in given city.

projection: students
criteria:  city=''
*/
select s.name,s.email,a.city
from student s,address a
where s.address_id = a.id AND a.city='mumbai';

/*
Output:
        name    email    city
        harry potter    harry@hogwards.com   mumbai
        ronald weasley  ronald@hogwards.com  mumbai
*/

/*
Q2. Display number of students that are living in each state.
projection: students
criteria: address
*/

select a.state,count(s.id)
from student s , address a
where s.address_id = a.id
group by a.state;

/*
state=MS
        1       harry potter    harry@hogwards.com   1       1       mumbai          MS
        India
        2       ronald weasley  ronald@hogwards.com  1       1       mumbai          MS
        India
        6       neville longbottom      neville@hogwards.com 5       5       pune    MS
        India
state=TN
        3       hermione granger        hermione@hogwards.com       2       2       chennaiTN
        India
state=MP
        4       draco malfoy    draco@hogwards.com   3       3       bhopal  MP      India
 state=Delhi
        5       ginni weasley   ginni@hogwards.com   4       4       delhi   Delhi   India
*/
```

```
/*
Output:
        state   count(s.id)
        Delhi   1
        MP      1
        MS      3
        TN      1
*/
/*
Q3. Display courses that belong to given department
projection:courses
criteria:department
*/

select c.*
from course c,department d
where c.department_id = d.id AND d.name="DEV";

/* Output:
        id      name    credits fee     department_id
        1       Java Programming        120     12000   2
        2       Python Programming      80      8000    2
*/
/*
Q4. Display number of courses for each department.
projection: courses
criteria: department

*/
 select d.name,count(c.id)
 from course c, department d
 where c.department_id = d.id
 group by d.name;

/*
d.name=Dev
        1       Java Programming        120     12000   2       2       Dev
        2       Python Programming      80      8000    2       2       Dev
d.name=Testing
        3       Selenium Admin          60      7000    3       3       Testing
dname=It
        4       Network Admin 70        8000    1       1       IT
Output:
        name    count(c.id)
        Dev     2
        IT      1
        Testing 1
*/

/*
Q5. Display students that have enrolled in given course.
projection: student
criteria: course
*/

select s.*,c.name
from student s,student_course sc, course c
where s.id=sc.student_id AND sc.course_id = c.id AND c.name='Network Admin';

/*
Output:
        id      name    email                                           address_id
        name
        3       hermione granger        hermione@hogwards.com       2               Network
Admin
        4       draco malfoy    draco@hogwards.com                  3               Network
Admin
*/
/*
Q6. Display students associated with given department.
 projection: student
 criteria: department
*/

select distinct s.name , s.id, s.email
from student s, student_course sc, course c, department d
where s.id=sc.student_id
         AND sc.course_id=c.id
   AND c.department_id = d.id
   AND d.name='Dev';

/*
        name    id      email
        harry potter    1       harry@hogwards.com
        ronald weasley  2       ronald@hogwards.com
        hermione granger        3       hermione@hogwards.com
*/

/*
Q7. Display number of students associated with each department.
project: students
criteria: department
*/
```

```
select d.name,count(distinct s.id) as number_of_students_associated

from student s, student_course sc, course c, department d

where s.id=sc.student_id

            AND sc.course_id=c.id

    AND c.department_id = d.id

group by d.name;


/*

d.name='Dev'

          1         harry potter      harry@hogwards.com   1        1        1        1        2024-02-01
          10        1         Java Programming        120      12000    2        2        Dev

          3         hermione granger        hermione@hogwards.com        2        3        1        4
          2024-02-01        10        1         Java Programming        120      12000    2        2        Dev

          1         harry potter      harry@hogwards.com   1        1        2        2        2024-02-10
          20        2         Python Programming     80       8000     2        2        Dev

          2         ronald weasley  ronald@hogwards.com 1        2        2        3        2024-02-10
          10        2         Python Programming     80       8000     2        2        Dev

          3         hermione granger        hermione@hogwards.com        2        3        2        5
          2024-02-01        10        2         Python Programming     80       8000     2        2        Dev

d.name='Testing'

          5         ginni weasley     ginni@hogwards.com   4        5        3        8        2024-03-01
          5         3         Selenium Admin          60       7000     3        3        Testing

d.name='IT'

          3         hermione granger        hermione@hogwards.com        2        3        4        6
          2024-02-01        10        4         Network Admin 70        8000     1        1        IT

          4         draco malfoy      draco@hogwards.com   3        4        4        7        2024-02-05
          5         4         Network Admin 70        8000     1        1        IT

*/


/*

Q8. Display students that have enrolled before given date

projection: student

criteria: student_course

*/
```

```
select *
from student s, student_course sc
where s.id = sc.student_id AND sc.date_of_enrollment <= '2024-02-05';

/*
Q9. Display courses for which the discount of more than 5% is given.
projection: course
criteria: student_course
*/

select distinct c.id, c.name, c.fee
from course c, student_course sc
where c.id = sc.course_id AND sc.discount>5;

/*
Q10. Display avg discount given for each course
projection: student_course
criteria: course
*/
select c.name,AVG(sc.discount)
from course c, student_course sc
where c.id = sc.course_id
group by c.name;

/* OUtput:
        name    AVG(sc.discount)
        Java Programming        10
        Network Admin 7.5
        Python Programming      13.333333333333334
        Selenium Admin          5
```

```
*/

/*
Q11. Display avg discount given to each student
projection: student_course
criteria: student
*/

select s.name, AVG(sc.discount)
from student s, student_course sc
where s.id = sc.student_id
group by s.name;

/* O/P:
        name    AVG(sc.discount)
        draco malfoy     5
        ginni weasley    5
        harry potter     15
        hermione granger         10
        ronald weasley  10
*/
```