(a) Write SQL statement to create and populate SECTION table with referential constraint. You need to show how to insert one tuple into the table.
(b) Write a SQL statement to retrieve the names of all senior students majoring in CS.
(c) Write a SQL statement to retrieve the names of all courses taught by Professor Anderson in 2004 and 2005.
(d) Retrieve the name and major departments of all students who do not have any grade A in any of their courses.
(e) Retrieve the name and major departments of all students who has at least one grade A in any of their courses.

Answer: Review the following SQL statements

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
DROP SCHEMA IF EXISTS simpleUniversity ;
CREATE SCHEMA IF NOT EXISTS simpleUniversity ;
```
USE simpleUniversity;

drop table if exists Course;

cREATE TABLE Course (  
  CourseName varchar(30) NOT NULL,  
  CourseNumber varchar(10) primary key,  
  NumOfCredits tinyint,  
  Department varchar(10) not null  
) ENGINE = InnoDB;

drop table if exists Section;

cREATE TABLE Section (  
  SectionId int not null,  
  CourseNo varchar(10) not null,  
  Semester char(6) not null,  
  Year Date not null,  
  Instructor varchar(20) not null,  
  primary key (SectionId, CourseNo, Semester, Year),  
  constraint fkSection foreign key (CourseNo) references Course(CourseNumber) on delete cascade on update cascade  
)engine = innoDB;

drop table if exists Prerequisite;

cREATE TABLE Prerequisite (  
  CourseNo varchar(10) not null,  
  PrerequisiteNo varchar(10) not null,  
  primary key (CourseNo, PrerequisiteNo),  
  constraint fkPrerequisite1 foreign key (CourseNo) references Course(CourseNumber) on delete no action on update cascade,  
  constraint fkPrerequisite2 foreign key (PrerequisiteNo) references Course(CourseNumber) on delete cascade on update cascade  
);

drop table if exists Student;

cREATE TABLE Student (  
  StudentNumber char(9) primary key,  
  name varchar(20) not null,  
  Classification tinyint not null,  
  DepartmentName vararch(10) not null  
) ENGINE = InnoDB;

drop table if exists GradeReport;

cREATE TABLE GradeReport;
CREATE TABLE `Course` (  
`CourseNo` char(9) not null,  
`Title` varchar(50) not null,  
`CourseId` int not null,  
`Credits` int not null,  
`Department` varchar(10) not null  
);  

CREATE TABLE `Section` (  
`SectionId` int not null,  
`CourseId` char(9) not null,  
`OfferedTerm` varchar(20) not null,  
`Year` int not null,  
`Location` varchar(20) not null,  
primary key (`SectionId`),  
constraint fkCourse foreign key (`CourseId`) references `Course`(`CourseNo`)  
);  

CREATE TABLE `Prerequisite` (  
`CourseId1` char(9) not null,  
`CourseId2` char(9) not null  
);  

CREATE TABLE `Student` (  
`StudentNumber` char(9) not null,  
`FirstName` varchar(20) not null,  
`LastName` varchar(20) not null,  
`ClassYear` int not null,  
`Major` varchar(10) not null  
);  

CREATE TABLE `GradeReport` (  
`StudentNumber` char(9) not null,  
`SectionId` int not null,  
`Grade` ENUM ('A', 'B', 'C', 'D', 'F')  
);  

FOREIGN KEY (`StudentNumber`) REFERENCES `Student`(`StudentNumber`);  

FOREIGN KEY (`SectionId`) REFERENCES `Section`(`SectionId`);  

SET SQL_MODE=@OLD_SQL_MODE;  
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;  
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;  

-- database population --  
insert into `Course` values  
('Intro to Computer Science', 'CS1310', 4, 'CS'),  
('Data Structures', 'CS3320', 4, 'CS'),  
('Discrete Mathematics', 'MATH2410', 4, 'MATH'),  
('Database Systems', 'CS3380', 4, 'CS');  

insert into `Section` values  
(85, 'MATH2410', 'Fall', '2004-09-01', 'King'),  
(92, 'CS1310', 'Fall', '2004-09-01', 'Anderson'),  
(102, 'CS3320', 'Spring', '2005-01-20', 'Knuth'),  
(112, 'MATH2410', 'Fall', '05-09-01', 'Chang'),  
(119, 'CS1310', 'Fall', '2005-09-01', 'Anderson'),  
(135, 'CS3380', 'Fall', '2005-09-01', 'Stone');  

insert into `Prerequisite` values  
('CS3380', 'CS3320'),  
('CS3380', 'MATH2410'),  
('CS3320', 'CS1310');  

insert into `Student` values  
('17', 'Bill Smith', 1, 'CS'),  
('8', 'Bob Brown', 2, 'CS'),  
('11', 'Nick Feiner', 4, 'MATH');  

insert into `GradeReport` values  
('17', 112, 'B'),  
('17', 119, 'C'),  
('8', 85, 'A'),  
('17', 135, 'A'),  
('8', 92, 'A'),  
('11', 85, 'B');  

select * from `Course`;  
select * from `Section`;
select * from Prerequisite;
select * from GradeReport;
select * from Student;

-- (b) Write a SQL statement to retrieve the names of all senior students majoring in CS --
select name from student
    where classification = 4 and DepartmentName = 'CS';

select name as 'Student Name' from student
    where classification = 4 and DepartmentName = 'MATH';

-- (c) Write a SQL statement to retrieve the names of all courses taught by Professor Anderson in 2004 and 2005.
SELECT CourseName
    FROM Course, Section
    WHERE Course.CourseNumber = Section.CourseNo
    AND Instructor = 'ANDERSON'
    AND (YEAR(YEAR) = '2004' OR YEAR(YEAR) = '2005');

SELECT CourseName
    FROM Course as C inner join Section S
    on C.CourseNumber = S.CourseNo
    AND Instructor = 'Anderson'
    AND (YEAR(YEAR) = '2004');

-- (d) Retrieve the name and major departments of all students who do not have any grade A in any of their courses.

-- first select student number who has at least one grade 'A'
Select StudentNumber, name as StudentName, DepartmentName as Major
    from Student S
    where exists
        (select StudentNo
            from GradeReport AS GR
            where
                GR.StudentNo = S.StudentNumber
                and Grade = 'A');

-- Retrieve the name and major departments of all students who do not have any grade A in any of their courses.
Select StudentNumber, name as StudentName, DepartmentName as Major
    from Student S
    where not exists
        (Select StudentNo
from GradeReport AS GR
    where
        GR.StudentNo = S.StudentNumber
        and Grade = 'A');

-- Retrieve the name and major departments of all students who do not have all A grade

Select StudentNumber, name as StudentName, DepartmentName as Major
    from Student S
    where exists
        (Select StudentNo
            from GradeReport AS GR
            where
                GR.StudentNo = S.StudentNumber
                and NOT(Grade = 'A'));

-- Retrieve the name and major departments of all students who do have all A grade

Select StudentNumber, name as StudentName, DepartmentName as Major
    from Student S
    where not exists
        (Select StudentNo
            from GradeReport AS GR
            where
                GR.StudentNo = S.StudentNumber
                and NOT(Grade = 'A'));