Lab7_ANP_C7281_Having Clause PEDDA JAGADEESH AF0366969 ANP-C7281

Database Schema:

Use the same database scheme created in previous lab.

Assignment 1:

Task 1: Assume you are managing a database of student records, and you need to retrieve information about students born after June 16, 2009. What will be the SQL query for this?

Task 2: Assume you have a database containing a "Student" table with information about students, including their first names. You want to retrieve records of students whose first names start with either 'A' or 'J'. To achieve this, what will be your SQL query?

Task 3. Let's consider a scenario where you have a database with a "Student" table that contains information about students, including their first names and email addresses.

You want to retrieve records of students whose first name is not 'Alice' and whose email addresses contain the domain '@example.com'. To achieve this, what will be your SQL query?

Submission:

Create an SQL script file containing your solutions for all tasks (queries). Name the file "lab_assignment1.sql" Provide comments above each query to indicate the query's purpose.

Assignment 2:

Task1: Create a table Person with PersonID int, FirstName varchar(255),

LastName varchar(255) and age (int).

Make PersonID PRIMARY KEY.

Task2: Create a table Employee with emp_id int, first_name varchar(255)

last_name varchar(255) and age (int)

Make emp_id PRIMARY KEY.

Task 3: Insert data to Person table

Task 4: Insert data to Employee table

Task 5: Create a Union of two tables

Submission:

Create an SQL script file containing your solutions for the task. Name the file "lab_assignment2.sql" Provide comments above the query to indicate the query's purpose.

ChatGPT Exercise

Using ChatGPT generate SQL queries of the below problem.

Scenario 1: In a student grades database with tables for courses and grades, find the courses where the average grade is below a 'C' (consider 'C' as a passing grade). We have a "Course" table with the following columns:

CourseId, CourseName, CreditHours, and "Grade" table with the following a columns: StudentId(Foreign Key), CourseID((Foreign Key), Grade. you want to find courses where the average grade is below a "C". Generate the ChatGPT prompt for creating the queries for the above requirement.

SOLUTION:

Database Schema:

Use the same database scheme created in the previous lab.

Assignment 1:

Task 1: Assume you are managing a database of student records, and you need to retrieve information about students born after June 16, 2009. What will be the SQL query for this?

Code:

CREATE TABLE Students_data (

ID INT PRIMARY KEY,

First_Name VARCHAR(50),

```
Last_Name VARCHAR(50),
City VARCHAR(50),
Age INT,
Birth_Date DATE
);
```

```
mysql> CREATE TABLE Students_data (
    -> ID INT PRIMARY KEY,
-> First_Name VARCHAR(50),
-> Last_Name VARCHAR(50),
     -> City VARCHAR(50),
     -> Age INT,
     -> Birth_Date DATE
Query OK, 0 rows affected (0.07 sec)
mysql> describe Students_data;
                                   Null
                                                   Default
  Field
                  Type
                                            Key
                                                               Extra
                  int
                                   NO
                                            PRI
                                                   NULL
                  varchar(50)
varchar(50)
  First_Name
                                    YES
                                                    NULL
  Last_Name
                                   YES
                                                    NULL
  City
                  varchar(50)
                                    YES
                                                    NULL
                  int
                                    YES
                                                    NULL
  Birth_Date
                  date
                                   YES
                                                   NULL
 rows in set (0.00 sec)
```

Code:

INSERT INTO Students_data (ID, First_Name, Last_Name, City, Age, Birth_Date)

VALUES

(1, 'Aarav', 'Sharma', 'Mumbai', 23, '2000-01-15'),

- (2, 'Vivaan', 'Verma', 'Delhi', 22, '2001-02-22'),
- (3, 'Diya', 'Patel', 'Bangalore', 21, '2002-03-30'),
- (4, 'Aanya', 'Reddy', 'Hyderabad', 20, '2003-04-12'),
- (5, 'Ishaan', 'Singh', 'Chennai', 19, '2004-05-19'),
- (6, 'Anaya', 'Kumar', 'Pune', 18, '2005-06-05'),
- (7, 'Arjun', 'Nair', 'Kochi', 17, '2006-07-20'),
- (8, 'Aadhya', 'Mehta', 'Ahmedabad', 16, '2007-08-25'),
- (9, 'Aryan', 'Joshi', 'Surat', 15, '2008-09-10'),
- (10, 'Anvi', 'Bose', 'Kolkata', 14, '2009-10-18'),
- (11, 'Vihaan', 'Das', 'Lucknow', 13, '2010-11-30'),
- (12, 'Mira', 'Roy', 'Jaipur', 12, '2011-12-25'),
- (13, 'Reyansh', 'Chopra', 'Chandigarh', 11, '2012-01-15'),
- (14, 'Aarohi', 'Kapoor', 'Indore', 10, '2013-02-22'),
- (15, 'Kabir', 'Malhotra', 'Bhopal', 9, '2014-03-30'),
- (16, 'Anaya', 'Rao', 'Patna', 8, '2015-04-12'),
- (17, 'Advik', 'Srivastava', 'Thane', 7, '2016-05-19'),
- (18, 'Myra', 'Ahuja', 'Nashik', 6, '2017-06-05'),
- (19, 'Ayaan', 'Iyer', 'Vadodara', 6, '2017-06-05'),
- (20, 'Anika', 'Menon', 'Nagpur', 5, '2018-07-20'),
- (21, 'Arnav', 'Jain', 'Agra', 23, '2000-01-15'),
- (22, 'Saanvi', 'Shah', 'Varanasi', 22, '2001-02-22'),
- (23, 'Krishna', 'Khan', 'Gurgaon', 21, '2002-03-30'),
- (24, 'Riya', 'Singh', 'Noida', 20, '2003-04-12'),

- (25, 'Dev', 'Chaudhary', 'Ghaziabad', 19, '2004-05-19'),
- (26, 'Sara', 'Pandey', 'Faridabad', 18, '2005-06-05'),
- (27, 'Kabir', 'Srinivasan', 'Meerut', 17, '2006-07-20'),
- (28, 'Anvi', 'Rathore', 'Rajkot', 16, '2007-08-25'),
- (29, 'Vihaan', 'Dutta', 'Jodhpur', 15, '2008-09-10'),
- (30, 'Mira', 'Ghosh', 'Gwalior', 14, '2009-10-18'),
- (31, 'Reyansh', 'Chatterjee', 'Coimbatore', 13, '2010-11-30'),
- (32, 'Aarohi', 'Bhattacharya', 'Jabalpur', 12, '2011-12-25'),
- (33, 'Kabir', 'Mukherjee', 'Vijayawada', 11, '2012-01-15'),
- (34, 'Anaya', 'Chatterjee', 'Madurai', 10, '2013-02-22'),
- (35, 'Advik', 'Gandhi', 'Raipur', 9, '2014-03-30'),
- (36, 'Myra', 'Sen', 'Kota', 8, '2015-04-12'),
- (37, 'Ayaan', 'Dasgupta', 'Guwahati', 7, '2016-05-19'),
- (38, 'Anika', 'Roy', 'Chandigarh', 6, '2017-06-05'),
- (39, 'Arnav', 'Basu', 'Hubli-Dharwad', 5, '2018-07-20'),
- (40, 'Saanvi', 'Nair', 'Tiruchirappalli', 23, '2000-01-15'),
- (41, 'Krishna', 'Patel', 'Mysore', 22, '2001-02-22'),
- (42, 'Riya', 'Reddy', 'Salem', 21, '2002-03-30'),
- (43, 'Dev', 'Sharma', 'Tirunelveli', 20, '2003-04-12'),
- (44, 'Sara', 'Verma', 'Malegaon', 19, '2004-05-19'),
- (45, 'Kabir', 'Singh', 'Gorakhpur', 18, '2005-06-05'),
- (46, 'Anvi', 'Gupta', 'Guntur', 17, '2006-07-20'),
- (47, 'Vihaan', 'Joshi', 'Bikaner', 16, '2007-08-25'),

- (48, 'Mira', 'Shetty', 'Amravati', 15, '2008-09-10'),
- (49, 'Reyansh', 'Nath', 'Cuttack', 14, '2009-10-18'),
- (50, 'Aarohi', 'Bhatt', 'Bhubaneswar', 13, '2010-11-30');

```
Output:

mysql> INSERT INTÓ Students_data (ĪD, First_Name, Last_Name, City, Age, Birth_Date)
    -v VALUES
    -(1, 'Aarav', 'Sharma', 'Mumbai', 23, '2000-01-15'),
    -(2, 'Vivaan', 'Verma', 'Delhi', 22, '2001-02-22'),
    -(3, 'Diya', 'Patel', 'Bangalore', 21, '2002-03-30'),
    -(4, 'Aanya', 'Reddy, 'Hyderabad', 20, '2003-04-12'),
    -(5, 'Ishaan', 'Singh', 'Chennai', 19, '2004-05-19'),
    -(6, 'Anya', 'Kumar', 'Pune', 18, '2005-06-05'),
    -(7, 'Arjun', 'Nair', 'Kochi', 17, '2006-07-20'),
    -(8, 'Aadhya', 'Mehta', 'Ahmedbad', 16, '2007-08-25'),
    -(9, 'Aryan', 'Joshi', 'Surat', 15, '2008-09-10'),
    -(10, 'Anvi', 'Bose', 'Ko'lkata', 14, '2009-10-18'),
    -(11, 'Vihaan', 'Bose', 'Ko'lkata', 14, '2009-10-18'),
    -(12, 'Mira', 'Roy', 'Jaipur', 12, '2011-12-25'),
    -(13, 'Reyansh', 'Chopra', 'Chandigarh', 11, '2012-01-15'),
    -(14, 'Aarohi', 'Kapoor', 'Indore', 10, '2013-02-22'),
    -(16, 'Anaya', 'Rao', 'Patna', 'Bhopal', 9, '2014-03-30'),
    -(16, 'Anaya', 'Rao', 'Patna', 'Bhopal', 9, '2014-03-30'),
    -(16, 'Anaya', 'Rao', 'Patna', 'Bhopal', 9, '2014-03-30'),
    -(18, 'Myra', 'Ahuja', 'Nashik', 6, '2017-06-05'),
    -(22, 'Saanvi', 'Shah', 'Vardodara', 6, '2017-06-05'),
    -(22, 'Saanvi', 'Shah', 'Vardodara', 6, '2017-06-05'),
    -(24, 'Riya', 'Singh', 'Noida', 20, '2003-04-12'),
    -(25, 'Dev', 'Chaudhary', 'Garziabad', 19, '2004-05-19'),
    -(26, 'Sara', 'Patna', 'Sarai', 'Agra', 23, '2000-01-15'),
    -(22, 'Kabir', 'Srinivasan', 'Meerut', 17, '2006-07-20'),
    -(22, 'Kabir', 'Srinivasan', 'Meerut', 17, '2006-07-20'),
    -(23, 'Krishna', 'Ghoth', 'Gurgaon', 21, '2009-00-18'),
    -(26, 'Sara', 'Patna', 'Patna'
                                                                                                                                                                                                                                         (43,
(44,
(45,
                                                                                                                                                                     Sara
                                                                                                                                                            'Kabir'
'Anvi',
   -> (47, 'Vihaan', 'Gupta', 'Guntur'

-> (47, 'Vihaan', 'Joshi', 'Bikan

-> (48, 'Mira', 'Shetty', 'Amrava

-> (49, 'Reyansh', 'Nath', 'Cutta

-> (50, 'Aarohi', 'Bhatt', 'Bhuba

Query OK, 50 rows affected (0.21 sec)

Records: 50 Duplicates: 0 Warnings:
```

mysql> select * from Students_data;						
ID	First_Name	Last_Name	City	Age	 Birth_Date	
1	Aạrav	Sharma	Mumbai	23	2000-01-15	
2	Vivaan	Verma	Delhi	22	2001-02-22	
3	Diya	Patel	Bangalore	21	2002-03-30	
4	Aanya	Reddy	Hyderabad	20	2003-04-12	
4 5 6	Ishaan	Singh	Chennai	19	2004-05-19	
7	Anaya	Kumar	Pune Kochi	18 17	2005-06-05	
8	Arjun	Nair Mehta	Ahmedabad	16	2006-07-20 2007-08-25	
9	Aadhya Aryan	Joshi	Surat	15	2007-08-23	
10	Anvi	Bose	Kolkata	14	2008-09-10	
	Vihaan	Das	Lucknow	13	2010-11-30	
12	Mira	Roy	Jaipur	12	2011-12-25	
13	Reyansh	Chopra	Chandigarh	11	2012-01-15	
14	Aarohi	Kapoor	Indore	10	2013-02-22	
i 15 i	Kabir	Malhotra	Bhopal		2014-03-30	
i 16 i	Anaya	Rao	Patna	9 8	i 2015-04-12 i	
17	Advik	Srivastava	Thane	7	2016-05-19	
18	Myra	Ahuja	Nashik	6	2017-06-05	
19	Ayaan	Iyer	Vadodara	6	2017-06-05	
20	Anika	Menon	Nagpur	5	2018-07-20	
21	Arnav .	Jain	Agra .	23	2000-01-15	
22	Saanvi	Shah	Varanasi	22	2001-02-22	
23	Krishna	Khan	Gurgaon	21	2002-03-30	
24	Riya	Singh	Noida	20	2003-04-12	
25	Dev	Chaudhary	Ghaziabad	19	2004-05-19	
26 27	Sara Kabir	Pandey Srinivasan	Faridabad Meerut	18 17	2005-06-05 2006-07-20	
28	Anvi	Rathore	Rajkot	16	2006-07-20	
29	Vihaan	Dutta	Jodhpur	15	2007-08-23	
30	Mira	Ghosh	Gwalior	14	2009-10-18	
31	Reyansh	Chatterjee	Coimbatore	13	2010-11-30	
32	Aarohi	Bhattacharya	Jabalpur	12	2011-12-25	
j 33 j	Kabir	Mukherjee	Vijayawada	11	2012-01-15	
j 34 j	Anaya	Chatterjee	Maďurai	10	2013-02-22	
35	Advik	Gandhi	Raipur	9	2014-03-30	
36	Myra	Sen	Kota	8	2015-04-12	
37	Ayaan	Dasgupta	Guwahati	7	2016-05-19	
38	Anika	Roy	Chandigarh	6	2017-06-05	
39	Arnav .	Basu	Hubli-Dharwad	5	2018-07-20	
40	Saanvi	Nair	Tiruchirappalli	23	2000-01-15	
41	Krishna	Patel	Mysore	22	2001-02-22	
42	Riya	Reddy	Salem Tirunelveli	21 20	2002-03-30 2003-04-12	
43	Dev Sara	Sharma Verma	Malegaon	19	2003-04-12 2004-05-19	
44	Kabir	Singh	Gorakhpur	18	2004-03-19	
46	Anvi	Gupta	Guntur	17	2003-00-03	
47	Vihaan	Joshi	Bikaner	16	2007-08-25	
48	Mira	Shetty	Amravati	15	2008-09-10	
49	Reyansh	Nath	Cuttack	14	2009-10-18	
50	Aarohi	Bhatt	Bhubaneswar	13	2010-11-30	
++		·		 	 	
50 row	s in set (0.0	O sec)				

Code:

SELECT * FROM Students_data WHERE birth_date > '2009-06-16';

Output:

10	mysql:	> SELECT * FRO	OM Students_data	a WHERE birth_da	te > '20	009-06-16';
11 Vihaan Das Lucknow 13 2010-11-30 12 Mira Roy Jaipur 12 2011-12-29 13 Reyansh Chopra Chandigarh 11 2012-01-19 14 Aarohi Kapoor Indore 10 2013-02-22 15 Kabir Malhotra Bhopal 9 2014-03-30 16 Anaya Rao Patna 8 2015-04-12 17 Advik Srivastava Thane 7 2016-05-19 18 Myra Ahuja Nashik 6 2017-06-09	ID	First_Name	Last_Name	City	Age	Birth_Date
20	11 12 13 14 15 16 17 18 19 20 30 31 32 33 34 35 36 37 38 39 49	Vihaan Mira Reyansh Aarohi Kabir Anaya Advik Myra Ayaan Anika Mira Reyansh Aarohi Kabir Anaya Advik Myra Advik Myra Advik Myra Advik Myra Ayaan Anika	Das Roy Chopra Kapoor Malhotra Rao Srivastava Ahuja Iyer Menon Ghosh Chatterjee Bhattacharya Mukherjee Chatterjee Gandhi Sen Dasgupta Roy Basu Nath	Lucknow Jaipur Chandigarh Indore Bhopal Patna Thane Nashik Vadodara Nagpur Gwalior Coimbatore Jabalpur Vijayawada Madurai Raipur Kota Guwahati Chandigarh Hubli-Dharwad	14 13 12 11 10 9 8 7 6 5 14 13 12 11 10 9 8 7 6	2009-10-18 2010-11-30 2011-12-25 2012-01-15 2013-02-22 2014-03-30 2015-04-12 2016-05-19 2017-06-05 2017-06-05 2018-07-20 2009-10-18 2010-11-30 2011-12-25 2012-01-15 2013-02-22 2014-03-30 2015-04-12 2016-05-19 2017-06-05 2018-07-20 2009-10-18 2010-11-30

Task 2: Assume you have a database containing a "Student" table with information about students, including their first names. You want to retrieve records of students whose first names start with either 'A' or 'J'. To achieve this, what will be your SQL query?

Code:

SELECT * FROM Students_data

WHERE first_name LIKE 'A%' OR first_name LIKE 'J%';

mysql> SELECT * FROM Students_data -> WHERE first_name LIKE 'A%' OR first_name LIKE 'J%';						
ID	First_Name	Last_Name	City	Age	Birth_Date	
++ 1 4 6 7 8 9 10 14 16 17 20 21 28 32 34 35 37 38 39 46 50	Aarav Aanya Anaya Arjun Aadhya Aryan Anvi Aarohi Anaya Advik Ayaan Anika Arnav Anvi Aarohi Anaya Advik Ayaan Anika Arnav Anvi Aarohi Anaya Advik Ayaan Anika Arnav Anika	Sharma Reddy Kumar Nair Mehta Joshi Bose Kapoor Rao Srivastava Iyer Menon Jain Rathore Bhattacharya Chatterjee Gandhi Dasgupta Roy Basu Gupta Bhatt	Mumbai Hyderabad Pune Kochi Ahmedabad Surat Kolkata Indore Patna Thane Nagpur Agra Rajkot Jabalpur Madurai Raipur Guwahati Chandigarh Hubli-Dharwad Guntur Bhubaneswar	23 20 18 17 16 15 14 10 8 7 6 5 23 16 12 10 9 7 6 5 17	2000-01-15 2003-04-12 2005-06-05 2006-07-20 2007-08-25 2013-02-22 2015-04-12 2007-08-25 2016-05-19 2017-06-05 2018-07-20 2016-05-19 2017-06-05 2018-07-20 2016-05-19 2017-06-05 2018-07-20 2016-05-20 2016-07-20 2010-11-30	
22 rows in set (0.20 sec)						

Task 3. Let's consider a scenario where you have a database with a "Student" table that contains information about students, including their first names and email addresses. You want to retrieve records of students whose first name is not 'Alice' and whose email addresses contain the domain '@example.com'. To achieve this, what will be your SQL query?

```
Code:

CREATE TABLE Student (

id INT AUTO_INCREMENT PRIMARY KEY,

first_name VARCHAR(255),

email VARCHAR(255)
);
```

```
mysql> CREATE TABLE Student (
-> id INT AUTO_INCREMENT PRIMARY KEY,
-> first_name VARCHAR(255),
-> email VARCHAR(255)
-> );-- Step 1: Create the Student table
Query OK, 0 rows affected (0.05 sec)
```

Code:

SELECT * FROM Student

WHERE first_name <> 'Alice' AND email LIKE '% @example.com';

Output:

```
mysql> select * from Student;
  id | first_name
                      email
                      alice@example.com
   1
       Alice
   2
       John
                       john@example.com
   3
                       jane@example.com
       Jane
   4
                      bob@example.com
       Bob
   5
       Anna
                      anna@example.com
5 rows in set (0.00 sec)
mysql> SELECT * FROM Student
-> WHERE first_name <> 'Alice' AND email LIKE '‰example.com';
  id | first_name |
                      email
   23
                      john@example.com
jane@example.com
       John
       Jane
                      bob@example.com
       Bob
                      anna@example.com
       Anna
4 rows in set (0.00 sec)
```

Submission:

Create an SQL script file containing your solutions for all tasks (queries). Name the file

```
"lab_assignment1.sql" Provide comments above each query to indicate the
query's
purpose.
Assignment 2:
Task1: Create a table Person with PersonID int, FirstName varchar(255),
LastName varchar(255) and age (int). Make PersonID PRIMARY KEY.
Code:
CREATE TABLE Person (
PersonID INT AUTO_INCREMENT PRIMARY KEY,
FirstName VARCHAR(255),
LastName VARCHAR(255),
Age INT
);
Output:
```

```
mysql> CREATE TABLE Person (
    -> PersonID INT AUTO_INCREMENT PRIMARY KEY,
    -> FirstName VARCHAR(255),
    -> LastName VARCHAR(255),
    -> Age INT
    ->);
Query OK, 0 rows affected (0.26 sec)
```

```
Task2: Create a table Employee with emp_id int, first_name varchar(255) last_name varchar(255) and age (int ) Make emp_id PRIMARY KEY.
```

```
Code:

CREATE TABLE Employee (
emp_id INT AUTO_INCREMENT PRIMARY KEY,
first_name VARCHAR(255),
last_name VARCHAR(255),
age INT
);
```

```
mysql> CREATE TABLE Employee
            emp_id INT AUTO_INCREMENT PRIMARY KEY,
             irst_name VARCHAR(255),
            last_name VARCHAR(255),
           age INT
Query OK, O rows affected (0.28 sec)
mysql> describe Employee;
  Field
                                Nu11
                                              Default
                Type
                                       Key
                                                          Extra
  emp_id
                                                          auto_increment
                int
                                NO
                                        PRI
                                               NULL
                varchar(255)
varchar(255)
                                YES
   irst_name
                                               NULL
  last_name
                                YES
                                               NULL
                                YES
  age
 rows in set (0.00 sec)
```

Task 3: Insert data to the Person table

Code:

```
-- Insert data into the Person table
```

```
('John', 'Doe', 30),

('Jane', 'Smith', 25),

('Alice', 'Johnson', 28),

('Bob', 'Brown', 35),

('Emily', 'Davis', 22);
```

Output:

mysql> select * from person;				
PersonID	FirstName	LastName	Age	
1 2 3 4	John Jane Alice Bob Emily	Doe Smith Johnson Brown Davis	30 25 28 35 22	
+++ 5 rows in set (0.00 sec)				

Task 4: Insert data to Employee table

Code:

-- Insert data into the Employee table

INSERT INTO Employee (first_name, last_name, age)

VALUES

('Michael', 'Scott', 45);

```
('Jim', 'Halpert', 32);
('Pam', 'Beesly', 30);
('Dwight', 'Schrute', 38);
('Stanley', 'Hudson', 50);
```

mysql> select * from Employee;				
emp_id	first_name	last_name	age	
1 2 3 4 5	Michael Jim Pam Dwight Stanley	Scott Halpert Beesly Schrute Hudson	45 32 30 38 50	
++ 5 rows in set (0.00 sec)				

Task 5: Create Union of two tables

Code:

SELECT first_name AS Name, last_name AS Surname, age AS Age FROM Employee

UNION

SELECT FirstName AS Name, LastName AS Surname, Age FROM Person;

```
nysql> SELECT first_name AS Name, last_name AS Surname, age AS Age FROM Employee
    -> SELECT FirstName AS Name, LastName AS Surname, Age FROM Person;
                        Age
                           45
32
30
 Michael
             Scott
             Halpert
Beesly
 Jim
 Pam
                           38
50
30
             Schrute
 Dwight
 Stanley
John
                           25
28
             Johnson
 Alice
                           35
22
             Brown
  rows in set (0.00 sec)
```

Submission:

Create an SQL script file containing your solutions for the task. Name the file

"lab_assignment2.sql" Provide comments above the query to indicate the query's

purpose.

ChatGPT Exercise

Using ChatGPT generate SQL queries of the below problem.

Scenario 1: In a student grades database with tables for courses and grades, find the courses where the average grade is below a 'C' (consider 'C' as a passing grade).

We have a "Course" table with the following columns:

CourseId, CourseName, CreditHours, and "Grade" table with the following columns: StudentId(Foreign Key), CourseID((Foreign Key), Grade. you want to find courses where the average grade is below a "C". Generate the ChatGPT prompt for creating the queries for the above requirement.

```
Code:
-- Create the Course table
CREATE TABLE Course (
CourseId INT PRIMARY KEY,
CourseName VARCHAR(100),
CreditHours INT
);
-- Create the Grade table
CREATE TABLE Grade (
StudentId INT,
CourseId INT,
Grade CHAR(1),
FOREIGN KEY (CourseId) REFERENCES Course(CourseId)
);
```

```
mysql> -- Create the Grade table
mysql> CREATE TABLE Grade (
                      StudentId INT,
                      CourseId INT,
                      Grade CHAR(1)
                      FOREIGN KEY (CourseId) REFERENCES Course(CourseId)
-> );
Query OK, O rows affected (0.06 sec)
mysql> -- Insert values into the Course table
mysql> -- Insert values into the Course table
mysql> INSERT INTO Course (CourseId, CourseName, CreditHours) VALUES
    -> (1, 'Mathematics', 4),
    -> (2, 'Physics', 3),
    -> (3, 'Chemistry', 4),
    -> (4, 'Biology', 3),
    -> (5, 'History', 2);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
mysql> -- Insert values into the Grade table
mysql> INSERT INTO Grade (StudentId, CourseId, Grade) VALUES
-> (101, 1, 'A'),
-> (102, 1, 'B'),
-> (103, 1, 'C'),
        -> (101, 1,
-> (102, 1,
-> (103, 1,
        -> (102, 1,

-> (103, 1,

-> (101, 2,

-> (102, 2,

-> (103, 2,

-> (101, 3,

-> (102, 3,

-> (103, 3,

-> (101, 4,

-> (102, 4,
         -> (102, 4,
                                 'D'
         -> (103, 4,
         -> (101, 5,
                                 'B'
        -> (102, 5,
-> (103, 5, 'C');
Query OK, 15 rows affected (0.01 sec)
Records: 15 Duplicates: 0 Warnings: 0
```

Code:

-- Assuming the following grade point values:

```
-- 'A' = 4.0

-- 'B' = 3.0

-- 'C' = 2.0

-- 'D' = 1.0

-- 'F' = 0.0
```

-- SQL query to find courses where the average grade is below a 'C'

SELECT c.CourseId, c.CourseName, c.CreditHours

FROM Course c

JOIN Grade g ON c.CourseId = g.CourseId

GROUP BY c.CourseId, c.CourseName, c.CreditHours

HAVING AVG(

CASE g.Grade

WHEN 'A' THEN 4.0

WHEN 'B' THEN 3.0

WHEN 'C' THEN 2.0

WHEN 'D' THEN 1.0

WHEN 'F' THEN 0.0

ELSE NULL

END

) < 2.0;

THANN YOU, SIN