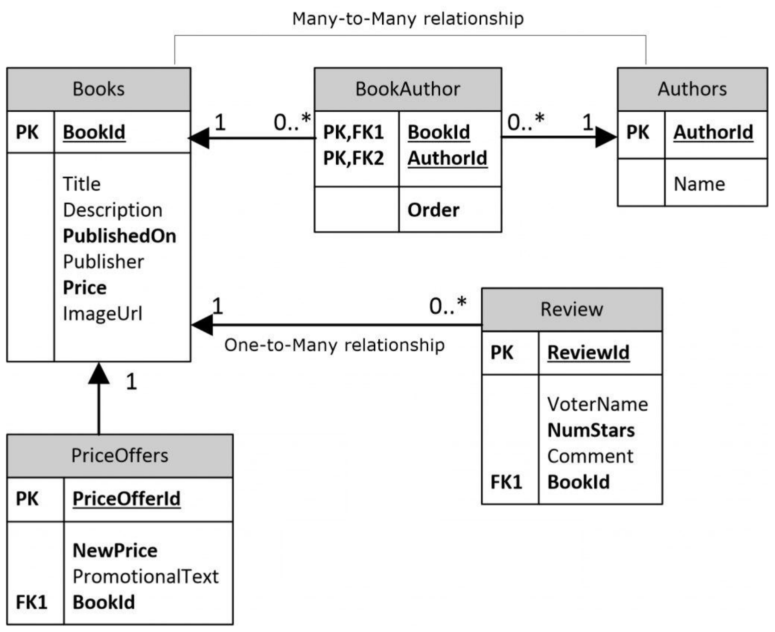
Homework 5

Your Name: Naga Satya Silpa Annadevara

Student ID: A20517818

---

1. Use DDL to answer the following questions. Note: you do not need to run SQL in Oracle. Just give the answers for the following requests [40]



1). Use DDL to create a database named as "BookInfo". Create the tables, except the table "BookAuthor" [5]

ANSWER:

```
CREATE DATABASE BookInfo;

CREATE TABLE Books (
BookID INT NOT NULL,
Description VARCHAR2(100),
Title VARCHAR2(30),
PublishedOn DATE,
Publisher VARCHAR2(20),
Price FLOAT,
ImageUrl VARCHAR2(100)
CONSTRAINT Books_PK PRIMARY KEY (BookID)
);
```

```
CREATE TABLE PriceOffers (
PriceOfferID INT NOT NULL,
NewPrice DECIMAL(6,2),
PromotionalText VARCHAR2(100)
BookID INT NOT NULL,
CONSTRAINT PriceOffers_PK PRIMARY KEY (PriceOfferID),
CONSTRAINT PriceOffers_FK1 FOREIGN KEY (BookID) REFERENCES Books (BookID)
);


CREATE TABLE Authors (
AuthorID INT NOT NULL,
Name VARCHAR2(30),
CONSTRAINT Authors _PK PRIMARY KEY (AuthorID)
);


CREATE TABLE Review (
ReviewID INT NOT NULL,
VoterName VARCHAR2(40),
NumStars INT NOT NULL,
Comment VARCHAR2(300),
BookID INT NOT NULL,
CONSTRAINT Review_PK PRIMARY KEY (ReviewID),
CONSTRAINT Review_FK1 FOREIGN KEY (BookID) REFERENCES Books (BookID)
);
```

2). Create the table BookAuthor without definition of keys. [5]

ANSWER:

```
CREATE TABLE BookAuthor (
BookID INT NOT NULL,
AuthorID INT NOT NULL,
Order VARCHAR2 (30)
);
```

3). Add PK and FK to the table BookAuthor [5]

ANSWER:

```
ALTER TABLE BookAuthor (
ADD CONSTRAINT BookAuthor_PK PRIMARY KEY (BookID, AuthorID),
ADD CONSTRAINT BookAuthor_FK1 FOREIGN KEY (BookID) REFERENCES Books
(BookID),
```

ADD CONSTRAINT BookAuthor_FK2 FOREIGN KEY (AuthorID) REFERENCES Authors (AuthorID)
);

4). Drop table “Authors” [5]

ANSWER:

ALTER TABLE BookAuthor DROP CONSTRAINT BookAuthor_FK2;
DROP TABLE Authors

5). Add a new attribute “ISBN” to the table “Books”, and set it as primary key [5]

ANSWER:

As Books table already has a Primary key (BookID), we can solve this question in 2 ways.

1st WAY:

Removing BookID as Primary key and setting ISBN as a new primary key in the Books Table as follows:

(In order to drop primary key (BookID) in books table, first we need to drop bookID in other tables that has been set as foreign keys):
ALTER TABLE PriceOffers DROP CONSTRAINT PriceOffers_FK1;
ALTER TABLE Review DROP CONSTRAINT Review_FK1;
ALTER TABLE BookAuthor DROP CONSTRAINT BookAuthor_FK1;

(Then drop primary key (BookID) in Books table):
ALTER TABLE Books
DROP CONSTAINT Books_PK;

(Then add new attribute ISBN and set it as a new PK in books table):
ALTER TABLE Books
ADD COLUMN ISBN INT NOT NULL,
ADD CONSTRAINT Books_PK PRIMARY KEY (ISBN);

(To have a connection with other tables we need to set ISBN as a FK in other tables)
ALTER TABLE PriceOffers
ADD CONSTRAINT PriceOffers_FK1 FOREIGN KEY (ISBN) REFERENCES Books (ISBN);

ALTER TABLE Review
ADD CONSTRAINT Review_FK1 FOREIGN KEY (ISBN) REFERENCES Books (ISBN);

ALTER TABLE BookAuthor
ADD CONSTRAINT BookAuthor_FK1 FOREIGN KEY (ISBN) REFERENCES Books (ISBN);

2nd WAY:

Keeping BookID as Primary key and setting BookID and ISBN together as a primary key in the Books Table as follows:

(To set BookID and ISBN together as a pk in books table, first we need to drop bookid alone as a pk in books table. To do so, we need to remove bookid in other tables that has been set as a fk's):

```
ALTER TABLE PriceOffers DROP CONSTRAINT PriceOffers_FK1;
ALTER TABLE Review DROP CONSTRAINT Review_FK1;
ALTER TABLE BookAuthor DROP CONSTRAINT BookAuthor_FK1;
```

(Then drop BookID as a Pk in books table):

```
ALTER TABLE Books
DROP CONSTAINT Books_PK;
```

(Then add new column ISBN and set both bookID and ISBN together as a PK in books table):

```
ALTER TABLE Books
ADD ISBN INT NOT NULL,
ADD CONSTRAINT Books_PK PRIMARY KEY (BookID,ISBN);
```

(As we are keeping BookID as one of the pk's in book table, we need to re-assign it as a FK in other tables that we dropped previously. To do so, we need to alter these tables again and re-add bookID as a FK in these tables):

```
ALTER TABLE PriceOffers
ADD CONSTRAINT PriceOffers_FK1 FOREIGN KEY (BookID) REFERENCES Books (BookID);

ALTER TABLE Review
ADD CONSTRAINT Review_FK1 FOREIGN KEY (BookID) REFERENCES Books (BookID);

ALTER TABLE BookAuthor
ADD CONSTRAINT BookAuthor_FK1 FOREIGN KEY (BookID) REFERENCES Books (BookID);
```

6). In the table "Books", we have the attribute "publisher" which refers to the publisher name. Now, we want to have a new entity "Publisher" with attributes, such as PublisherID, PublisherName, Address, Country, Tel, ContactPersonName, ContactPersonalEmail. Use DDL to make relevant changes. [5]

ANSWER:

(create a new table called Publisher and add constraints in it):

```
CREATE TABLE Publisher (
PublisherID INT NOT NULL,
PublisherName VARCHAR2(50),
Address VARCHAR2(200),
Country VARCHAR2(30),
Tel INT,
ContactPersonName VARCHAR2(50),
ContactPersonalEmail VARCHAR2(40)
ADD CONSTRAINT Publisher_PK PRIMARY KEY (PublisherID)
ADD CONSTRAINT Publisher_FK1 FOREIGN KEY (BookID) REFERENCES)
Books(BookID)   (#one way of adding FK here – adding bookid as FK in publisher table)
);
```

(Other way of adding FK here): (#adding publisher id as FK in books table)

```
ALTER TABLE Books
ADD COLUMN PublisherID INT NOT NULL,
ADD CONSTRAINT Books_FK1 FOREIGN KEY (PublisherID) REFERENCES
Publisher(PublisherID);
```

7). Add default value 9.9 to "Price" in table "Books" [5]

ANSWER:

We can write the syntax in different ways according to the professor ppt slides:

Way 1:

```
ALTER TABLE Books
MODILFY Price FLOAT DEFAULT '9.9';
```

Way 2:

```
ALTER TABLE Books
ALTER COLUMN Price
SET DEFAULT '9.9';
```

Way 3:

```
ALTER TABLE Books
SET Price DEFAULT '9.9';
```

8). Update the datatype as varchar(200) for ImageUrl in table "Books" [5]

ANSWER:

We can solve this in 2 ways:

1st WAY:

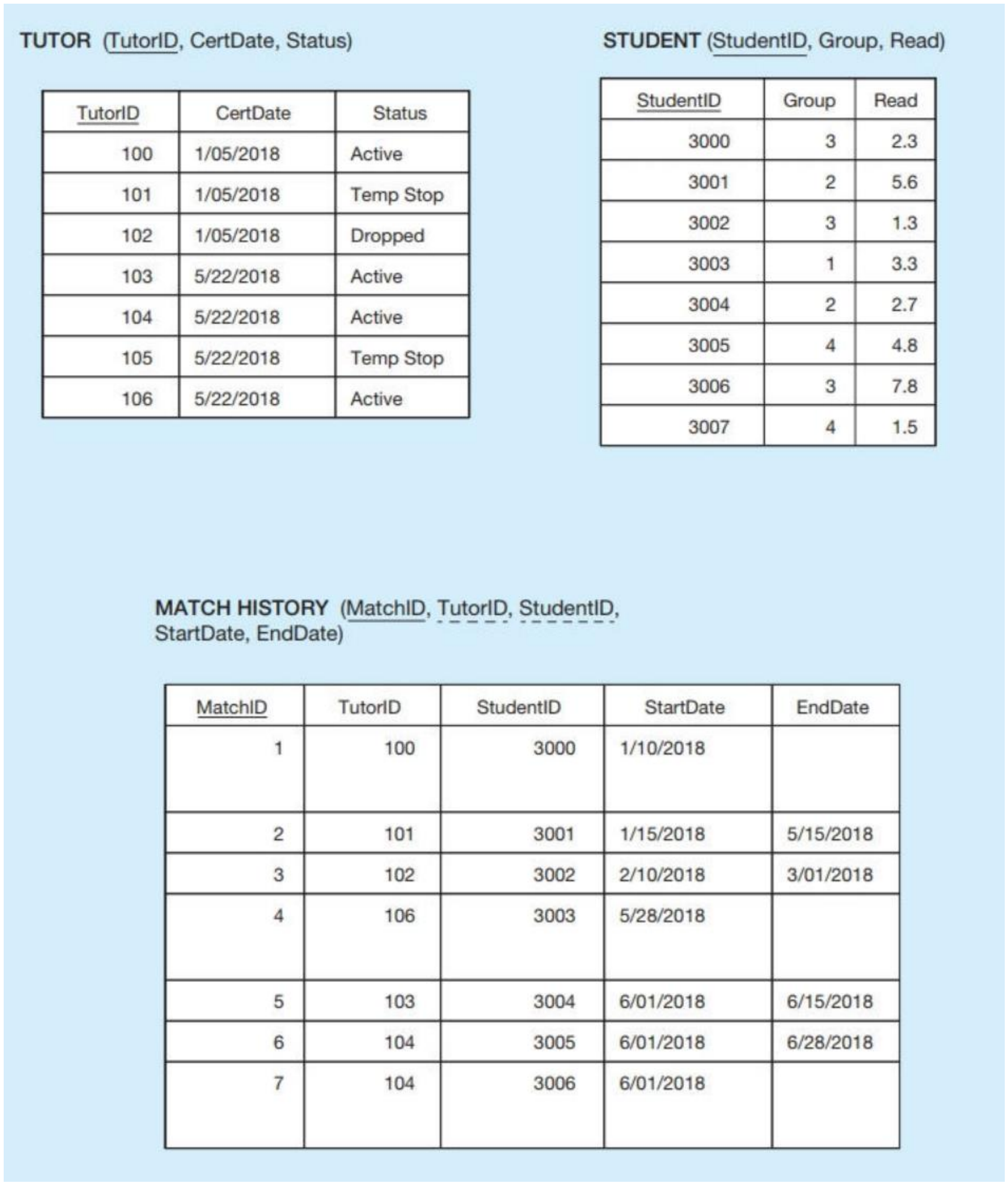
```
ALTER TABLE Books
MODIFY COLUMN ImageUrl VARCHAR2(200);
```

2nd WAY:

```
ALTER TABLE Books
ADD COLUMN ImageUrl2 VARCHAR2(200);          #adding a temporary column ImageUrl2

UPDATE Books
SET ImageUrl2 = TO_CHAR(ImageUrl);                    #Copy values from ImageUrl

ALTER TABLE Books
DROP COLUMN ImageUrl;                                 #drop old column ImageUrl

ALTER TABLE Books
RENAME COLUMN ImageUrl2 to ImageUrl;                  #Rename column to ImageUrl
```

2. By using the sample of the data tables below, using SQL to answer the following questions [40]

Note: the tables below just gave you sample data, you should assume that there are many more rows in each table. These sample rows are just used for you to better understand the values in each table. <u>You do not need to run them in Oracle</u>



TUTOR (TutorID, CertDate, Status)

TutorID	CertDate	Status
100	1/05/2018	Active
101	1/05/2018	Temp Stop
102	1/05/2018	Dropped
103	5/22/2018	Active
104	5/22/2018	Active
105	5/22/2018	Temp Stop
106	5/22/2018	Active

STUDENT (StudentID, Group, Read)

StudentID	Group	Read
3000	3	2.3
3001	2	5.6
3002	3	1.3
3003	1	3.3
3004	2	2.7
3005	4	4.8
3006	3	7.8
3007	4	1.5

MATCH HISTORY (MatchID, TutorID, StudentID, StartDate, EndDate)

MatchID	TutorID	StudentID	StartDate	EndDate
1	100	3000	1/10/2018	
2	101	3001	1/15/2018	5/15/2018
3	102	3002	2/10/2018	3/01/2018
4	106	3003	5/28/2018	
5	103	3004	6/01/2018	6/15/2018
6	104	3005	6/01/2018	6/28/2018
7	104	3006	6/01/2018	

1) how many tutors have a status of temp stop? Which tutors are active?

ANSWER:

SELECT COUNT (TutorID) (As it is asked 'how many tutors', we need to use 'count')
FROM Tutor
WHERE Status = 'Temp stop';

SELECT TutorID
FROM Tutor
WHERE Status = 'Active';

2) List the IDs of the tutors who are currently tutoring more than 1 student.

ANSWER:

```
SELECT TutorID
FROM Match History
GROUP BY TutorID
HAVING COUNT(StudentID) > 1;
```

(having is a conditional expression used to assign conditions on groups)

3) Get the list of student groups and also return the number of students in each group.

ANSWER:

```
SELECT Group, COUNT (*)
FROM Student
GROUP BY Group;
```

```
Select group, count(studentid)
From student
Group by group
```

(Or)

```
SELECT StudentID, COUNT (*)
FROM Student
GROUP BY StudentID
HAVING COUNT (Group) >1;
```

4) Which student has the highest Read score? And what is the score for that.

ANSWER:

We can solve this in 2 ways by using order by statement:

1st way :

```
SELECT StudentID, Read
FROM Student
ORDER BY Read DESC
FETCH FIRST 1 ROW ONLY;
```

2nd way:

```
SELECT StudentID, MAX(Read)
FROM Student
GROUP BY StudentID
```

```
ORDER BY MAX(Read) DESC
FETCH FIRST 1 ROW ONLY;
```

5) Show the average, max, min Read scores per student group.

ANSWER:

```
SELECT Group, AVG(Read) AS “AVERAGE”, MAX(Read), MIN(Read)
FROM Student
GROUP BY Group;
```

3. use the EnrollSys database (W6_Enrollment.sql) we introduced in the class, and write down the SQL statements to answer the following questions. [20]

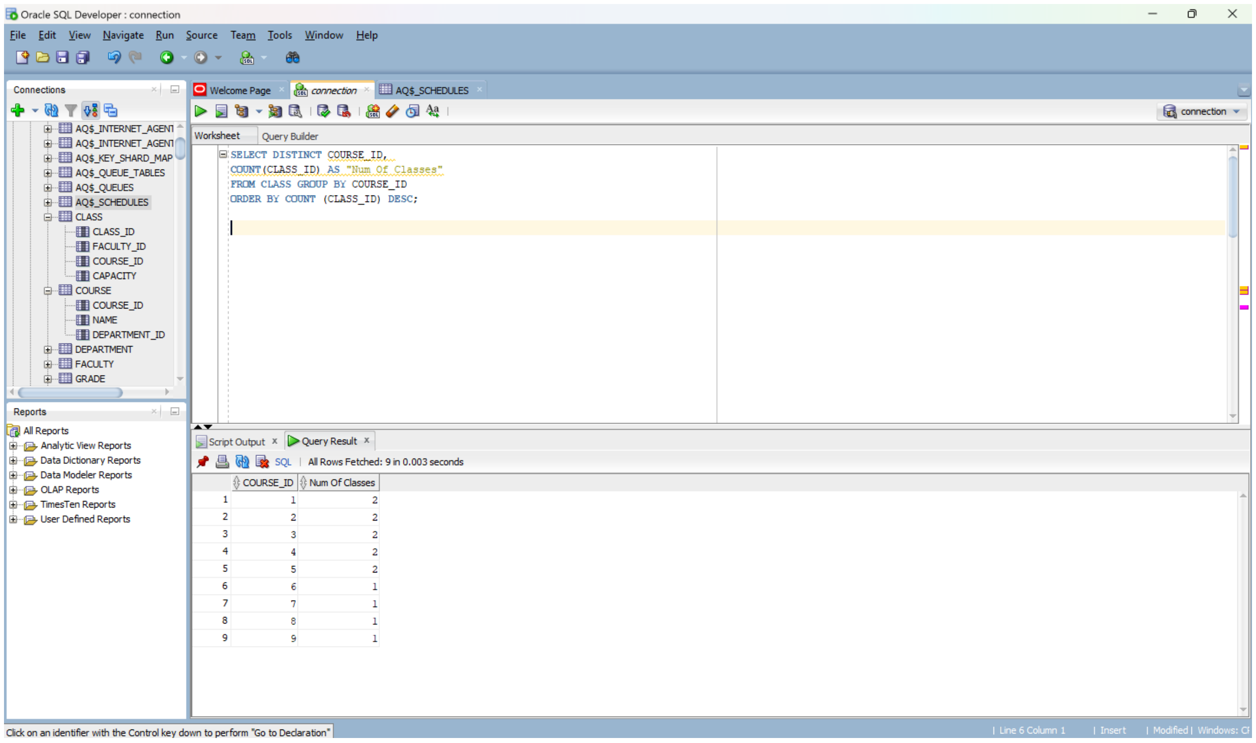
Note:

- You should provide your SQL statement, as well as the snapshot of the outputs in your Oracle
- Use a single SQL query for the following questions.
- Note that you should have the capability to write down correct SQL statement without running/examining them in the Oracle.

1). Return a list of unique courses along with the number of classes associated with it, rank the courses by the number of classes in descending order. Note: a class can be considered as a section in a course

ANSWER:

```
SELECT DISTINCT COURSE_ID, COUNT(CLASS_ID) AS "Num Of Classes"
FROM CLASS
GROUP BY COURSE_ID
ORDER BY COUNT (CLASS_ID) DESC;
```



```
AQ$_INTERNET_AGENT
AQ$_INTERNET_AGENT
AQ$_KEY_SHARD_MAP
AQ$_QUEUE_TABLES
AQ$_QUEUES
AQ$_SCHEDULES
CLASS
    CLASS_ID
    FACULTY_ID
    COURSE_ID
    CAPACITY
COURSE
    COURSE_ID
    NAME
    DEPARTMENT_ID
DEPARTMENT
FACULTY
GRADE
```

```sql
SELECT DISTINCT COURSE_ID,
COUNT(CLASS_ID) AS "Num Of Classes"
FROM CLASS GROUP BY COURSE_ID
ORDER BY COUNT (CLASS_ID) DESC;
```

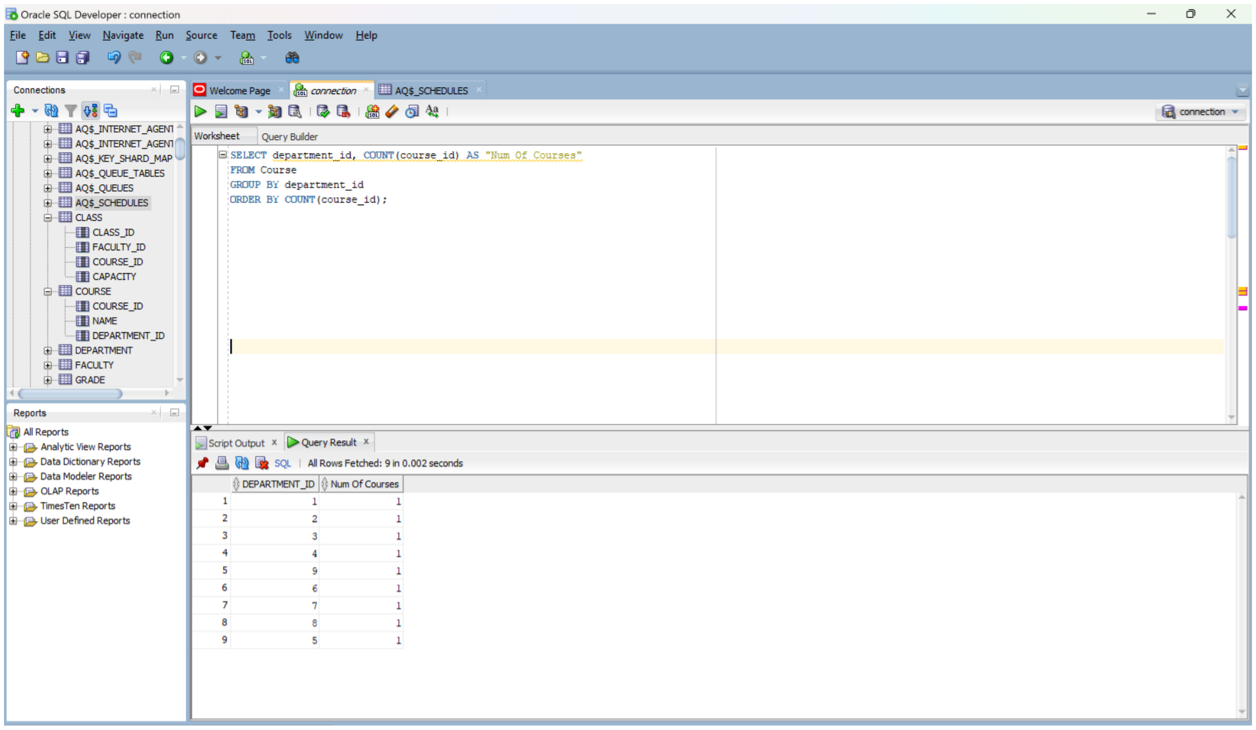
```
SQL | All Rows Fetched: 9 in 0.003 seconds

     COURSE_ID  Num Of Classes
1            1               2
2            2               2
3            3               2
4            4               2
5            5               2
6            6               1
7            7               1
8            8               1
9            9               1
```

```
Click on an identifier with the Control key down to perform "Go to Declaration"    | Line 6 Column 1 | Insert | Modified | Windows: CI
```

2. Return a list of unique departments along with the number of courses operated by the department.

ANSWER:

SELECT distinct department_id, COUNT(course_id) AS "Num Of Courses"
FROM Course
GROUP BY department_id
ORDER BY COUNT(course_id);



```
SELECT department_id, COUNT(course_id) AS "Num Of Courses"
FROM Course
GROUP BY department_id
ORDER BY COUNT(course_id);
```

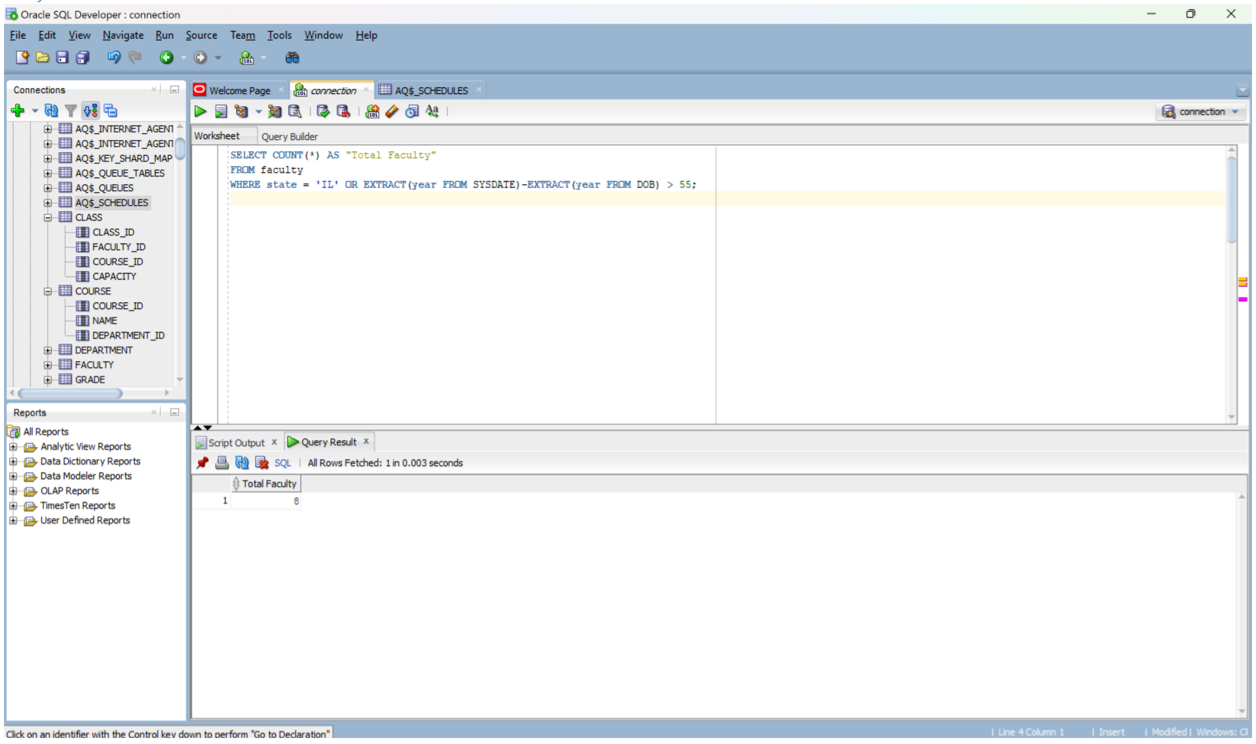
All Rows Fetched: 9 in 0.002 seconds

	DEPARTMENT_ID	Num Of Courses
1	1	1
2	2	1
3	3	1
4	4	1
5	9	1
6	6	1
7	7	1
8	8	1
9	5	1

3. Find how many faculties are from IL, or age is > 55.

ANSWER:

SELECT COUNT(*) AS "Total Faculty"
FROM faculty
WHERE state = 'IL' OR EXTRACT(year FROM SYSDATE)-EXTRACT(year FROM DOB) > 55;



```
SELECT COUNT(*) AS "Total Faculty"
FROM faculty
WHERE state = 'IL' OR EXTRACT(year FROM SYSDATE)-EXTRACT(year FROM DOB) > 55;
```

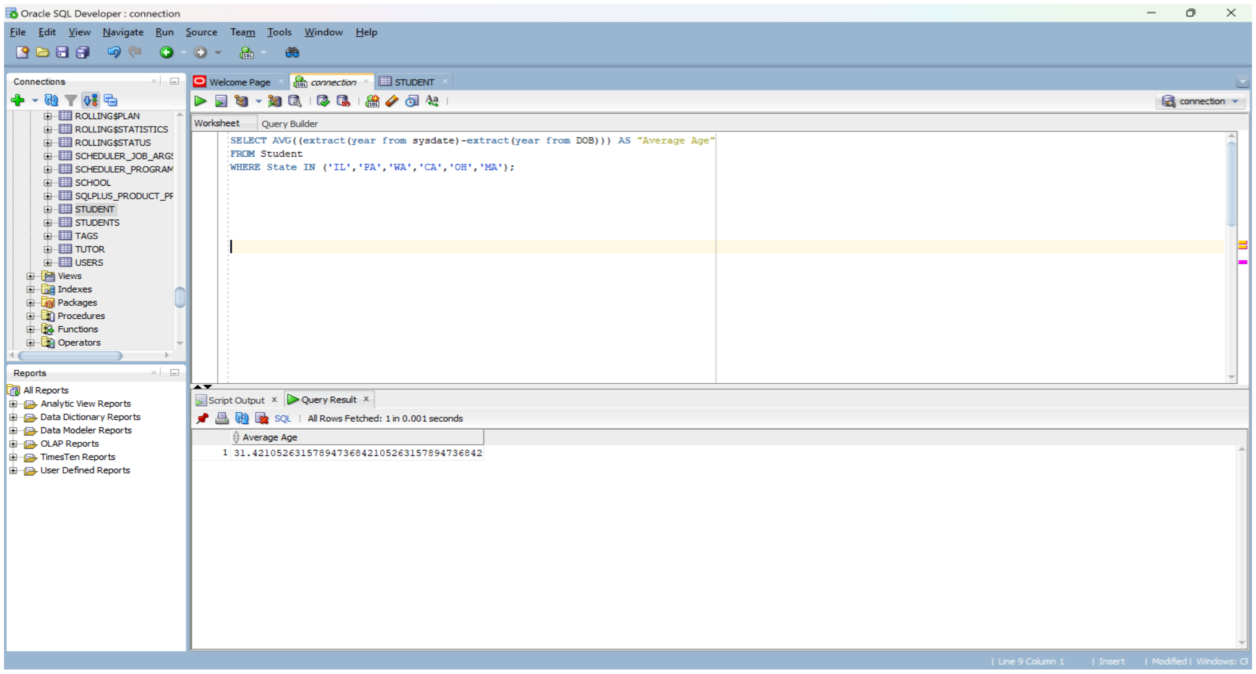
All Rows Fetched: 1 in 0.003 seconds

	Total Faculty
1	8

4. Find the average age for students who are from IL, PA, WA, CA, OH, MA

ANSWER:

SELECT AVG((extract(year from sysdate)-extract(year from DOB))) AS "Average Age"
FROM Student
WHERE State IN ('IL','PA','WA','CA','OH','MA');



```
SELECT AVG((extract(year from sysdate)-extract(year from DOB))) AS "Average Age"
FROM Student
WHERE State IN ('IL','PA','WA','CA','OH','MA');
```

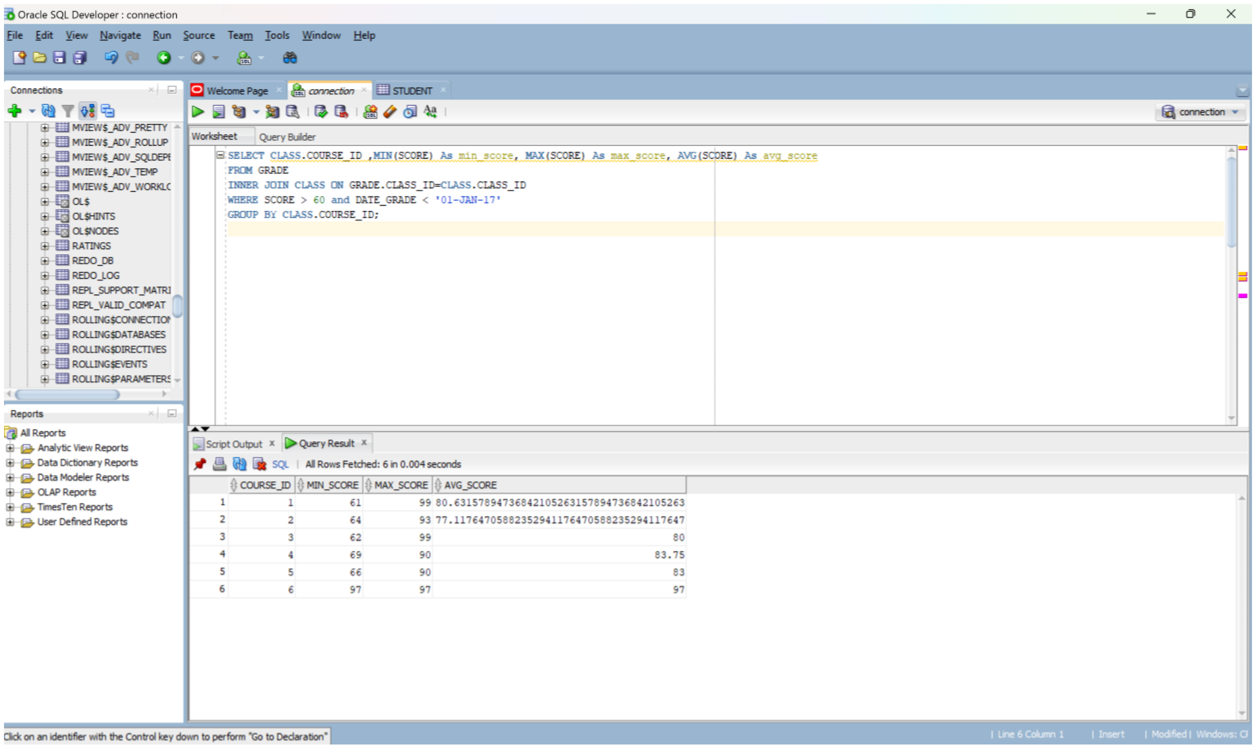
All Rows Fetched: 1 in 0.001 seconds

```
  Average Age
1 31.421052631578947368421052631578947368 42
```

5. return the min, max and avg score of the unique courses, only for the courses graded before 2017, and the min grade is > 60.

ANSWER:

SELECT CLASS.COURSE_ID ,MIN(SCORE) As "min_score", MAX(SCORE) As "max_score", AVG(SCORE) As "avg_score"
FROM GRADE
INNER JOIN CLASS ON GRADE.CLASS_ID=CLASS.CLASS_ID
WHERE GRADE.SCORE > 60 and (EXTRACT(YEAR FROM GRADE.DATE_GRADE))<2017
GROUP BY CLASS.COURSE_ID;

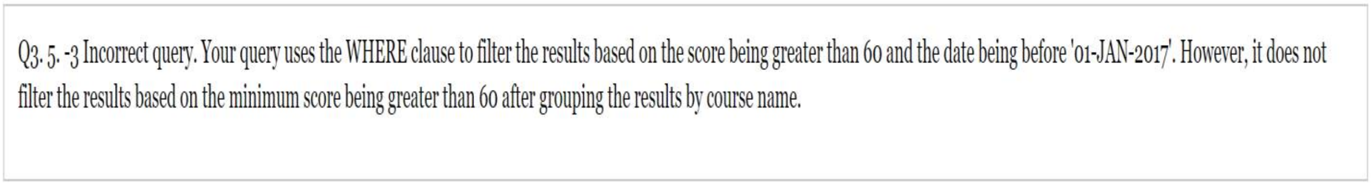


```sql
SELECT CLASS.COURSE_ID ,MIN(SCORE) As min_score, MAX(SCORE) As max_score, AVG(SCORE) As avg_score
FROM GRADE
INNER JOIN CLASS ON GRADE.CLASS_ID=CLASS.CLASS_ID
WHERE SCORE > 60 and DATE_GRADE < '01-JAN-17'
GROUP BY CLASS.COURSE_ID;
```

All Rows Fetched: 6 in 0.004 seconds

	COURSE_ID	MIN_SCORE	MAX_SCORE	AVG_SCORE
1	1	61	99	80.631578947368421052631578947368421052 63
2	2	64	93	77.117647058823529411764705882352941176 47
3	3	62	99	80
4	4	69	90	83.75
5	5	66	90	83
6	6	97	97	97

Feedback & score: I got 97. (updated from 92 to 95) but in the price default question don't use quotation marks for value 9.9 as its data type is float.



Q3. 5. -3 Incorrect query. Your query uses the WHERE clause to filter the results based on the score being greater than 60 and the date being before '01-JAN-2017'. However, it does not filter the results based on the minimum score being greater than 60 after grouping the results by course name.