12/20/2022

Blastonbury Pop Festival

Database Systems 5COM2005-2022/2023 CourseWork

REPORT

Name: Dhyan Nilesh Patel
SRN NUMBER: 21008521

School of Engineering and Computer of Science

ASSIGNMENT BRIEFING SHEET (2022/23 Academic Year) – ANONYMOUS MARKING

Assignment Title	Coursework	**Submission Date**	20 December 2022
Module Title	Database Systems	**Module Code**	5COM2005
Tutor	**Deepak Panday**	**GROUP or INDIVIDUAL Assignment**	**Individual**

FOR INDIVIDUAL ASSIGNMENTS – *STUDENT TO COMPLETE*

By completing **BOX A** below, I certify that the submitted work is entirely mine and that any material derived or quoted from the published or unpublished work of other persons has been duly acknowledged. **[ref. UPR AS12, section 7 and UPR AS14 (Appendix III)].)].** I also certify, that any work with human participants has been carried out under an approved ethics protocol in accordance with UPR RE01.

*Please **ONLY** provide your ID (srn) number as this assignment will be anonymously marked*

BOX A

Student ID Number (SRN)
21008521

This sheet must be submitted with the assignment, and either BOX A or B filled in.

LATE SUBMISSION WILL ATTRACT A STANDARD LATENESS PENALTY.

1. For undergraduate modules, a score of 40% or above represents a pass mark.
2. For postgraduate modules, a score of 50% or above represents a pass mark.
3. For work submitted up to 5 working days late marked is capped to a bare pass (40% for undergraduate and 50% for postgraduate).
4. For work submitted more than 5 working days a mark of zero will be awarded for the assignment.

Part 1 – ORACLE Mark 50

Q1.1 Database Modelling / SQL marks 10

Construct an Entity-Relationship diagram to model the specification described below.

The following stages should be undertaken:

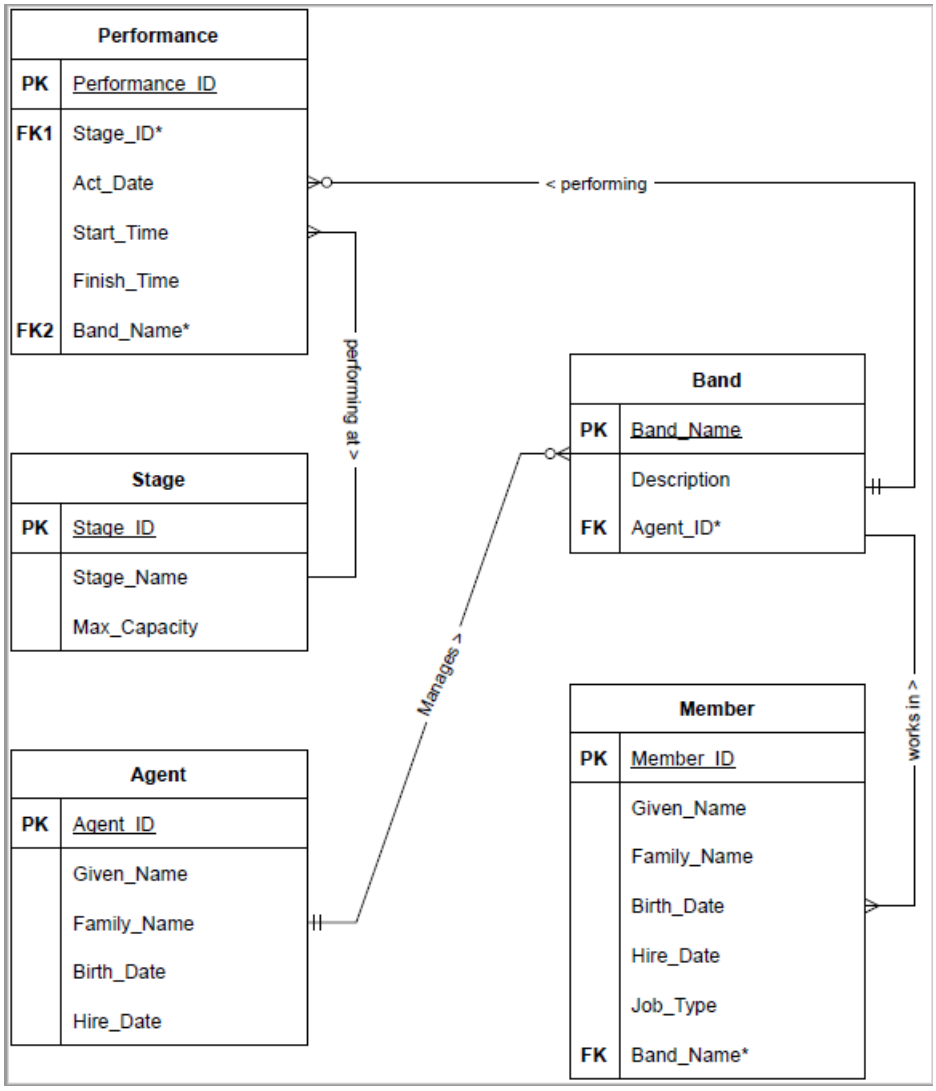
A. Identify the relevant Entities and label them appropriately (marks 3)
B. Link these with the relevant Relationships, these should be (marks 5)
    - Labelled appropriately
    - Define the type of relationship (1:1, 1:M or M:M)
    - Resolve any many-to-many relationships
    - Identify attributes for each entity and identify Primary and Foreign Keys (marks 2)

Your report needs to have ER-Diagram as shown the example below.

1.1 a) Identify the relevant Entities and label them appropriately

Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)
Band (Band_Name, Description, Agent_ID*)
Member (Member_ID, Given_Name, Family_Name, Birth_Date, Hire_Date, Job_Type, Band_Name*)
Stage (Stage_ID, Stage_Name, Max_Capacity)
Performance (Performance ID, Stage ID*, Act_Date, Start_Time, Finish_Time, Band_Name*)

1.1 b) ER diagram:



Q 1.2 Design Tables marks 10

Login to UH oracle through Oracle SQL Developer and create appropriate tables with all constraints which reflects the ER diagram you have designed in Q1.

You report needs to have all SQL script to create tables, add constraints on them.

SQL to create database in Oracle

```
-- Q 1.2  Design Tables

CREATE TABLE Agent
   (
   Agent_ID NUMBER(20) NOT NULL
   CONSTRAINT agent_agentid_pk PRIMARY KEY,
   Given_Name VARCHAR2(30),
   Family_Name VARCHAR2(30),
   Birth_Date DATE,
   Hire_Date DATE
   );

DESC Agent;

CREATE TABLE Band
   (
   Band_Name VARCHAR2(30) NOT NULL
   CONSTRAINT band_bandname_pk PRIMARY KEY,
   DESCRIPTION VARCHAR2(300),
   Agent_ID NUMBER(20) NOT NULL,
   CONSTRAINT band_agentid_fk FOREIGN KEY
   (Agent_ID) REFERENCES Agent(Agent_ID)
   );
```

```
DESC Band;

CREATE TABLE Member
   (
   Member_ID NUMBER(20) NOT NULL
   CONSTRAINT staff_staffid_pk PRIMARY KEY,
   Given_Name VARCHAR(30),
   Family_Name VARCHAR(30),
   Birth_Date DATE,
   Hire_Date DATE,
   Job_Type VARCHAR(60),
   Band_Name VARCHAR2(30) NOT NULL,
   CONSTRAINT staff_bandname_fk FOREIGN KEY
   (Band_Name) REFERENCES Band(Band_Name)
   );

DESC Member;

CREATE TABLE Stage
   (
   Stage_ID NUMBER(20) NOT NULL
   CONSTRAINT stage_stageid_pk PRIMARY KEY,
   Stage_Name VARCHAR(60),
   Max_Capacity NUMBER(25)
   );

DESC Stage;
```

```
CREATE TABLE Performance
   (
   Performance_ID NUMBER(20) NOT NULL
   CONSTRAINT perform_performid_pk PRIMARY KEY,
   Stage_ID NUMBER(20) NOT NULL,
   Act_Date DATE,
   Start_Time TIMESTAMP,
   Finish_Time TIMESTAMP,
   Band_Name VARCHAR2(30) NOT NULL,
   CONSTRAINT perform_bandname_fk FOREIGN KEY
   (Band_Name) REFERENCES Band(Band_Name),
   CONSTRAINT perform_stageid_fk FOREIGN KEY
   (Stage_ID) REFERENCES Stage(Stage_ID)
   );

DESC Performance;
```

Figure 1 shows SQL script to Create Agent Table, Band Table, Member Table, Stage Table and Performance Table.

SQL script to Create Agent Table, Band Table , Member Table, Stage Table and Performance Table.

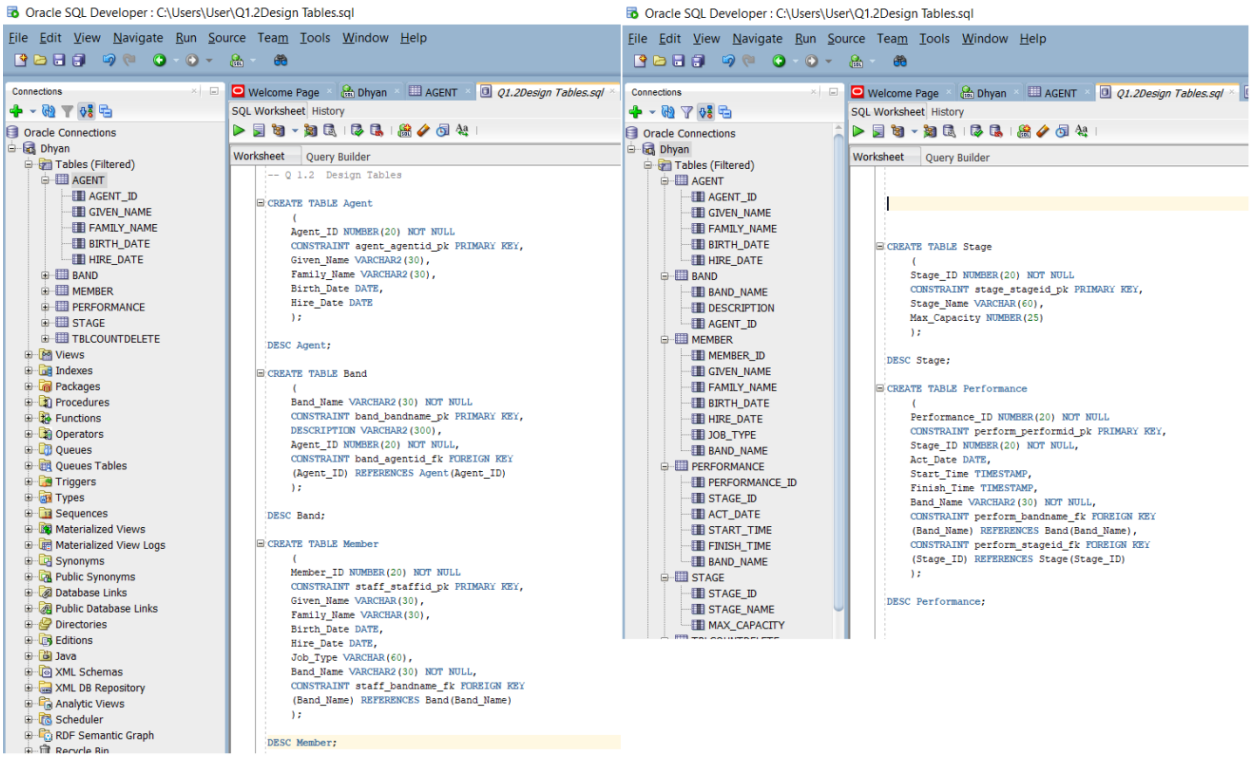


Figure 1

Q 1.3 Insert Data marks 10

Let us assume that there are two Agents- Agent1 and Agent2. The first Agent manages 3 bands and second agent manages 2 bands. There are in total 20 musicians distributed between the 5 bands, at least 5 different job types.

Now write an appropriate SQL statement (INSERT) to populate the tables with records under the above specification. You are free to consider any number of other objects do not provide in above specification.

Your report should have all SQL script to populate your table in ORACLE DB.

-- Q 1.3 Insert Data

1.3 a) Insert statements of 2 Agents:

-- Agent 1

```
INSERT INTO Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)
VALUES ( '2468101214', 'Rowan', 'Atkinson', TO_DATE('06/01/1990', 'DD/MM/YYYY'), TO_DATE('12/12/2020', 'DD/MM/YYYY'));
```

-- Agent 2

```
INSERT INTO Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)
VALUES ( '3691215182', 'Charlie', 'Chaplin', TO_DATE('16/04/1985', 'DD/MM/YYYY'), TO_DATE('25/06/2018', 'DD/MM/YYYY'));
```

1.3 b) Insert statements of 5 Bands:

-- Band 1

```
INSERT INTO Band (Band_Name, Description, Agent_ID)
VALUES ( 'Queen', 'They are the kings of arena rock and pop rock', '2468101214');
```

-- Band 2

```
INSERT INTO Band (Band_Name, Description, Agent_ID)
VALUES ( 'Rush', 'Rush is progressive rock god', '2468101214');
```

-- Band 3

INSERT INTO Band (Band_Name, Description, Agent_ID)

VALUES ('Yes', 'Yes played music in the blues-rock and folk-rock genres', '2468101214');

-- Band 4

INSERT INTO Band (Band_Name, Description, Agent_ID)

VALUES ('Maroon', 'American pop rock band from Los Angeles', '3691215182');

-- Band 5

INSERT INTO Band (Band_Name, Description, Agent_ID)

VALUES ('BTS', 'Kpop boy band from BigHit Entertainment', '3691215182');

1.3 c) Insert statements of 20 Members:

-- Band 1 = 4 Musicians

INSERT INTO Member VALUES ('21005078', 'James', 'Adler', TO_DATE('07/04/1996', 'DD/MM/YYYY'), TO_DATE('02/12/2016', 'DD/MM/YYYY'), 'Musician', 'Queen');

INSERT INTO Member VALUES ('21005079', 'Robert', 'Anderson', TO_DATE('03/07/1988', 'DD/MM/YYYY'), TO_DATE('07/11/2021', 'DD/MM/YYYY'), 'Composer', 'Queen');

INSERT INTO Member VALUES ('21005080', 'John', 'Beckett', TO_DATE('09/05/1990', 'DD/MM/YYYY'), TO_DATE('12/09/2019', 'DD/MM/YYYY'), 'Instrument Technician', 'Queen');

INSERT INTO Member VALUES ('21005081', 'Michael', 'Carson', TO_DATE('10/09/1994', 'DD/MM/YYYY'), TO_DATE('18/06/2017', 'DD/MM/YYYY'), 'Bassist', 'Queen');

-- Band 2 = 4 Musicians

INSERT INTO Member VALUES ('21006112', 'David', 'Brady', TO_DATE('12/11/2002', 'DD/MM/YYYY'), TO_DATE('19/01/2018', 'DD/MM/YYYY'), 'Musician', 'Rush');

```
INSERT INTO Member VALUES ( '21006113', 'William', 'Cooper', TO_DATE('14/02/1993', 'DD/MM/YYYY'), TO_DATE('09/09/2020', 'DD/MM/YYYY'), 'DJ', 'Rush');

INSERT INTO Member VALUES ( '21006114', 'Richard', 'Davis', TO_DATE('18/01/2001', 'DD/MM/YYYY'), TO_DATE('21/08/2019', 'DD/MM/YYYY'), 'Drummer', 'Rush');

INSERT INTO Member VALUES ( '21006115', 'Joseph', 'Dixon', TO_DATE('19/12/2000', 'DD/MM/YYYY'), TO_DATE('05/07/2021', 'DD/MM/YYYY'), 'Guitarist', 'Rush');

-- Band 3 = 4 Musicians

INSERT INTO Member VALUES ( '21007232', 'Thomas', 'Lennox', TO_DATE('22/11/1995', 'DD/MM/YYYY'), TO_DATE('22/02/2014', 'DD/MM/YYYY'), 'Musician', 'Yes');

INSERT INTO Member VALUES ( '21007233', 'Daniel', 'Lincoln', TO_DATE('13/08/1995', 'DD/MM/YYYY'), TO_DATE('24/12/2016', 'DD/MM/YYYY'), 'Keyboardist', 'Yes');

INSERT INTO Member VALUES ( '21007234', 'Mary', 'Mason', TO_DATE('17/10/1997', 'DD/MM/YYYY'), TO_DATE('16/11/2016', 'DD/MM/YYYY'), 'Bassist', 'Yes');

INSERT INTO Member VALUES ( '21007235', 'Jennifer', 'Wiley', TO_DATE('27/02/1987', 'DD/MM/YYYY'), TO_DATE('19/04/2018', 'DD/MM/YYYY'), 'Drummer', 'Yes');

-- Band 4 = 4 Musicians

INSERT INTO Member VALUES ( '21008356', 'Lisa', 'Walker', TO_DATE('25/02/2000', 'DD/MM/YYYY'), TO_DATE('09/06/2018', 'DD/MM/YYYY'), 'Musician', 'Maroon');

INSERT INTO Member VALUES ( '21008357', 'Steven', 'Wilson', TO_DATE('14/04/1994', 'DD/MM/YYYY'), TO_DATE('17/08/2019', 'DD/MM/YYYY'), 'Sound Technician', 'Maroon');

INSERT INTO Member VALUES ( '21008358', 'Paul', 'Parker', TO_DATE('18/11/1998', 'DD/MM/YYYY'), TO_DATE('15/10/2020', 'DD/MM/YYYY'), 'Event Manager', 'Maroon');

INSERT INTO Member VALUES ( '21008359', 'Ronald', 'Miler', TO_DATE('08/04/1999', 'DD/MM/YYYY'), TO_DATE('11/03/2020', 'DD/MM/YYYY'), 'Guitarist', 'Maroon');

-- Band 5 = 4 Musicians
```

INSERT INTO Member VALUES ('21009462', 'Sharon', 'Nash', TO_DATE('22/05/2002', 'DD/MM/YYYY'), TO_DATE('23/01/2017', 'DD/MM/YYYY'), 'Musician', 'BTS');

INSERT INTO Member VALUES ('21009463', 'Jacob', 'Smith', TO_DATE('26/08/1994', 'DD/MM/YYYY'), TO_DATE('16/02/2018', 'DD/MM/YYYY'), 'Singer', 'BTS');

INSERT INTO Member VALUES ('21009465', 'Stephen', 'Stone', TO_DATE('08/09/1996', 'DD/MM/YYYY'), TO_DATE('24/04/2020', 'DD/MM/YYYY'), 'Bassist', 'BTS');

INSERT INTO Member VALUES ('21009466', 'Alexander', 'Thompson', TO_DATE('06/03/1993', 'DD/MM/YYYY'), TO_DATE('26/06/2019', 'DD/MM/YYYY'), 'Composer', 'BTS');

Figure 2 shows that Data are successfully inserted.

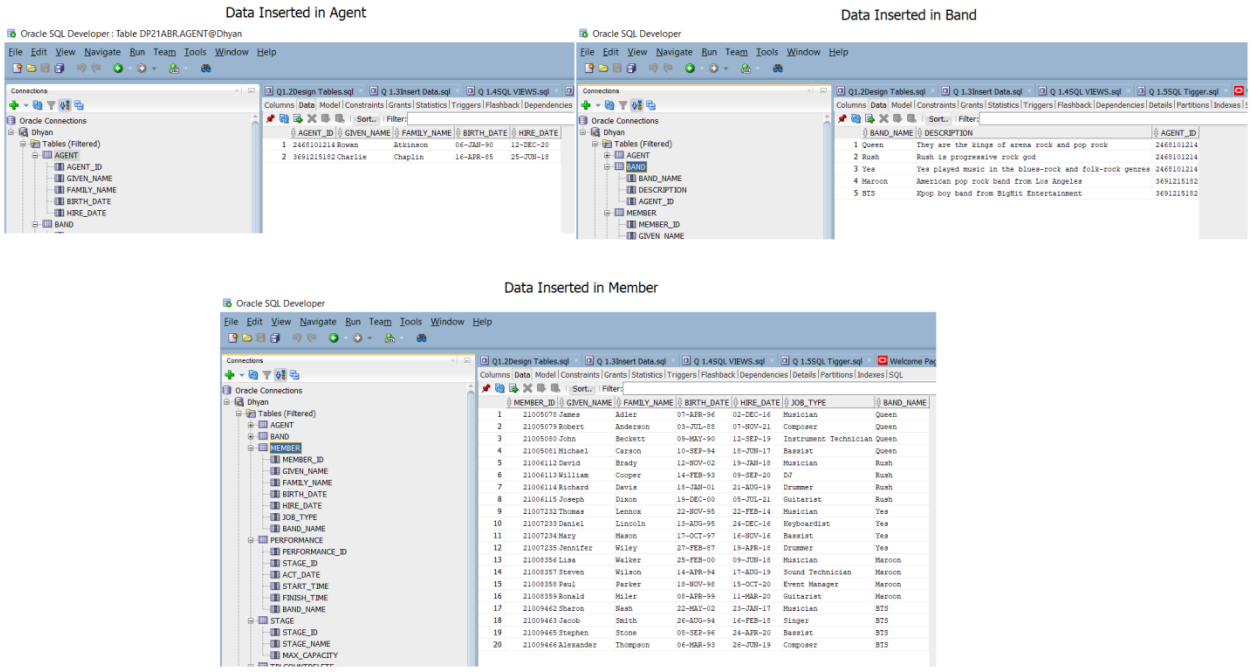


Figure 2

Q 1.4 SQL VIEWS marks 10

A. Create a view named ‘AgentJobs’ which display the number of different jobs carried out by each Agents.

B. Ensure that the view is READ-ONLY

C. Grant user -dp15aad access to the view.

Your report should have script of the view your have created, screen shot *of* ***Select * from AgentJobs*** and script to grant access.

1.4 A and B)

Creating view with read only option

```
CREATE VIEW AgentJobs
AS
SELECT Agent.Given_Name AS "Agent_Name",
COUNT(Member.Job_Type) AS "Number_Of_Jobs"
FROM Agent, Band, Member
WHERE Agent.Agent_ID = Band.Agent_ID
AND Band.Band_Name = Member.Band_Name
GROUP BY Agent.Given_Name
WITH READ ONLY;
```

Figure 3 shows screen-shot of select clause of the View created.

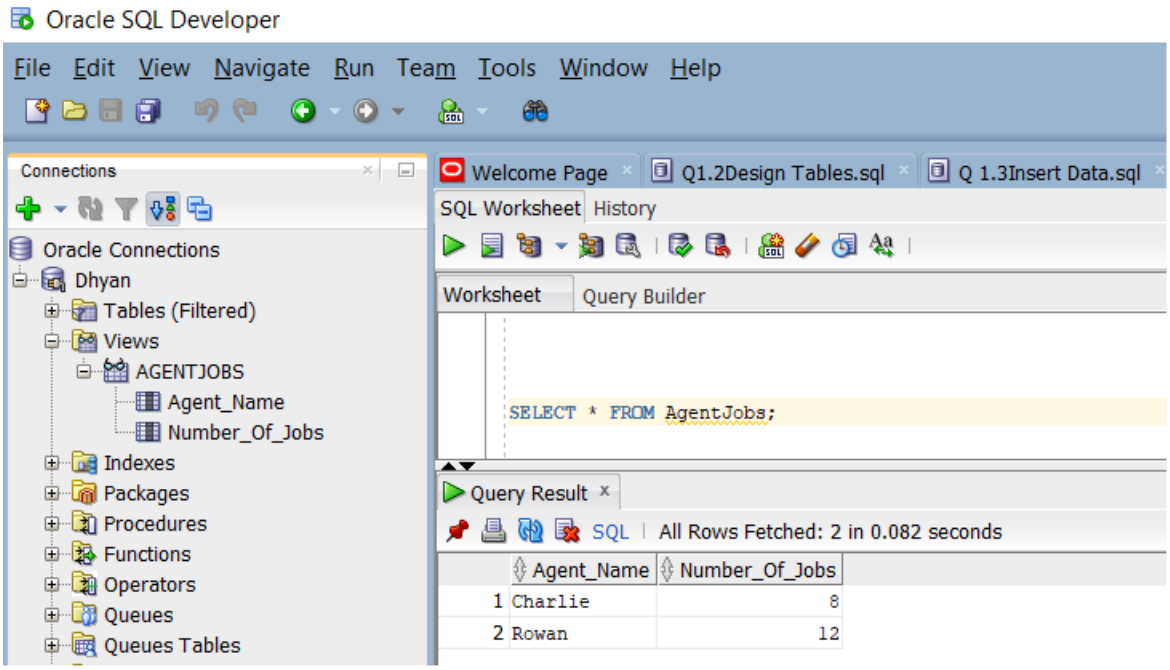


Figure 3

1.4 C) Grand Access to dp15aad

GRANT SELECT ON AgentJobs to dp15aad;

Figure 4 shows screen-shot of Grant Access.

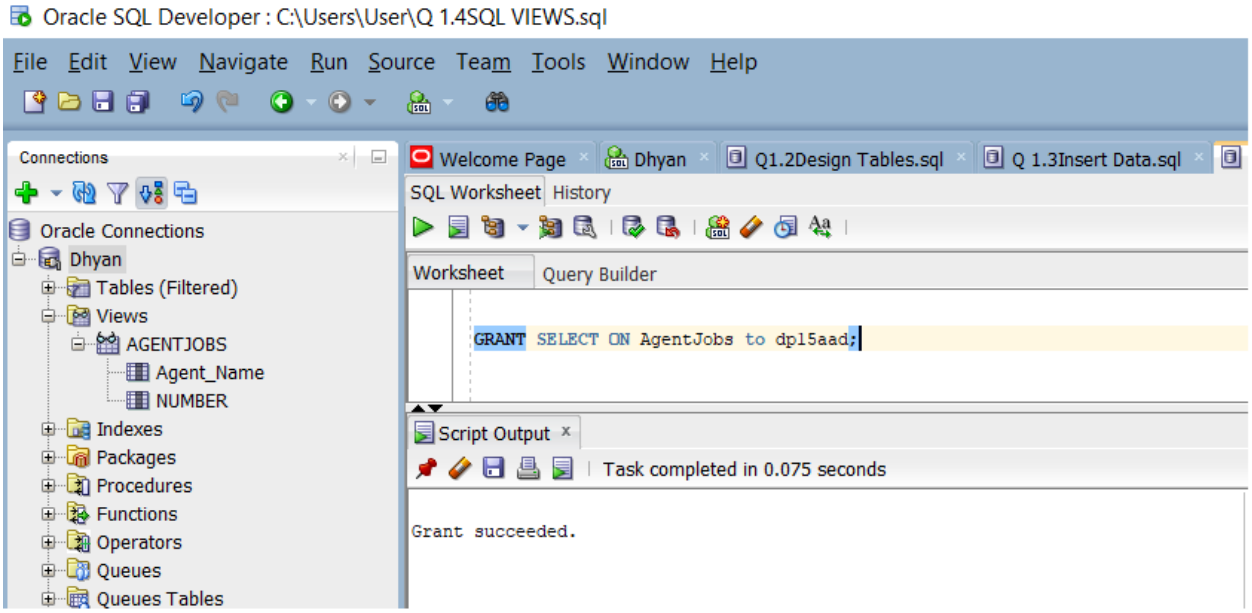


Figure 4

Q 1.5 SQL Tigger marks 10

Create a new table called tblCountDelete – which have only one attribute countDelete. The tblCountDelete keeps record to number of the time data is deleted from Agent table. Now write appropriate tigger which increase the countDelete in the tblCountDelete each time an entity is deleted from Agent table.

You report should have SQL for you tigger and evidence that your tigger is working well i.e. screen shot of the ***Select * from tblCountDelete***, before and after a record is deleted from Agent table. You can add a dummy data in your Agent table.

1.5 a) SQL statement to create tblCountDelete.

```
CREATE TABLE tblCountDelete
   (
   countDelete INT
   );
```

1.5 b) SQL statement to create tigger.

```
CREATE TRIGGER Deleted
AFTER DELETE ON Agent
FOR EACH ROW
BEGIN
 UPDATE tblCountDelete set countDelete = countDelete + 1;
END;
```

1.5 c) Evidence of trigger working well.

At First, I insert 0 into the tblCountDelete table:

```
INSERT INTO tblCountDelete VALUES(0);
```

Figure 5 shows screen-shot of creating a tblCountDelete with value 0.

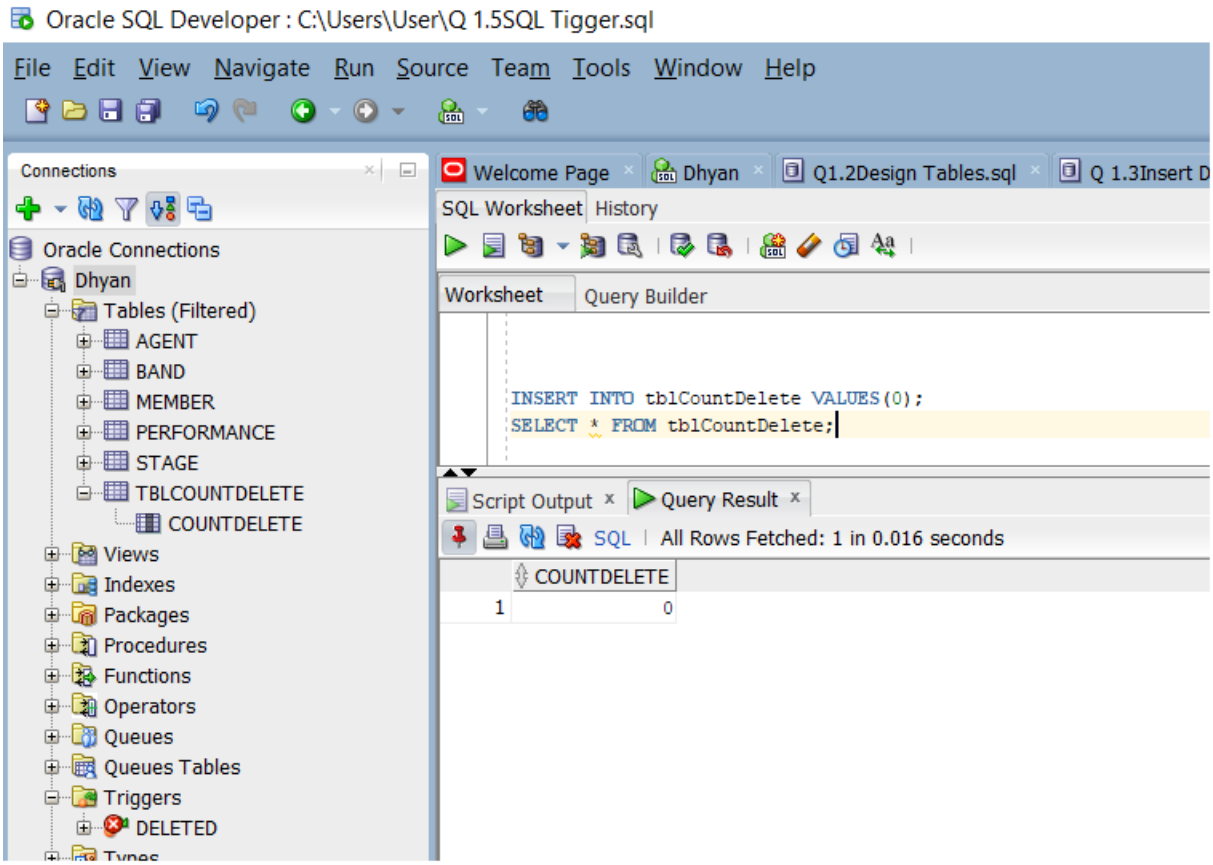


Figure 5

1.5 d) SQL to add three Dummy data in Job

-- Dummy Data 1

INSERT INTO Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)

VALUES ('12345', 'Dummy1', 'Data1', TO_DATE('10/01/1995', 'DD/MM/YYYY'), TO_DATE('15/11/2015', 'DD/MM/YYYY'));

-- Dummy Data 2

INSERT INTO Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)

VALUES ('12346', 'Dummy2', 'Data2', TO_DATE('20/05/1990', 'DD/MM/YYYY'), TO_DATE('25/05/2016', 'DD/MM/YYYY'));

-- Dummy Data 3

INSERT INTO Agent (Agent_ID, Given_Name, Family_Name, Birth_Date, Hire_Date)

VALUES ('12347', 'Dummy3', 'Data3', TO_DATE('10/10/1989', 'DD/MM/YYYY'), TO_DATE('13/10/2017', 'DD/MM/YYYY'));

1.5 e) Removing the three dummy Data from Job.

```
--Delete Data 1
DELETE FROM Agent WHERE Agent_ID = 12345;
--Delete Data 2
DELETE FROM Agent WHERE Agent_ID = 12346;
--Delete Data 3
DELETE FROM Agent WHERE Agent_ID = 12347;
```

1.5 f) Check if trigger is working or not.

View the table tblCountDelete:

```
SELECT * FROM tblCountDelete
```

Figure 6 shows screen-shot of the viewing tblCountDelete after removing three dummy Agents.

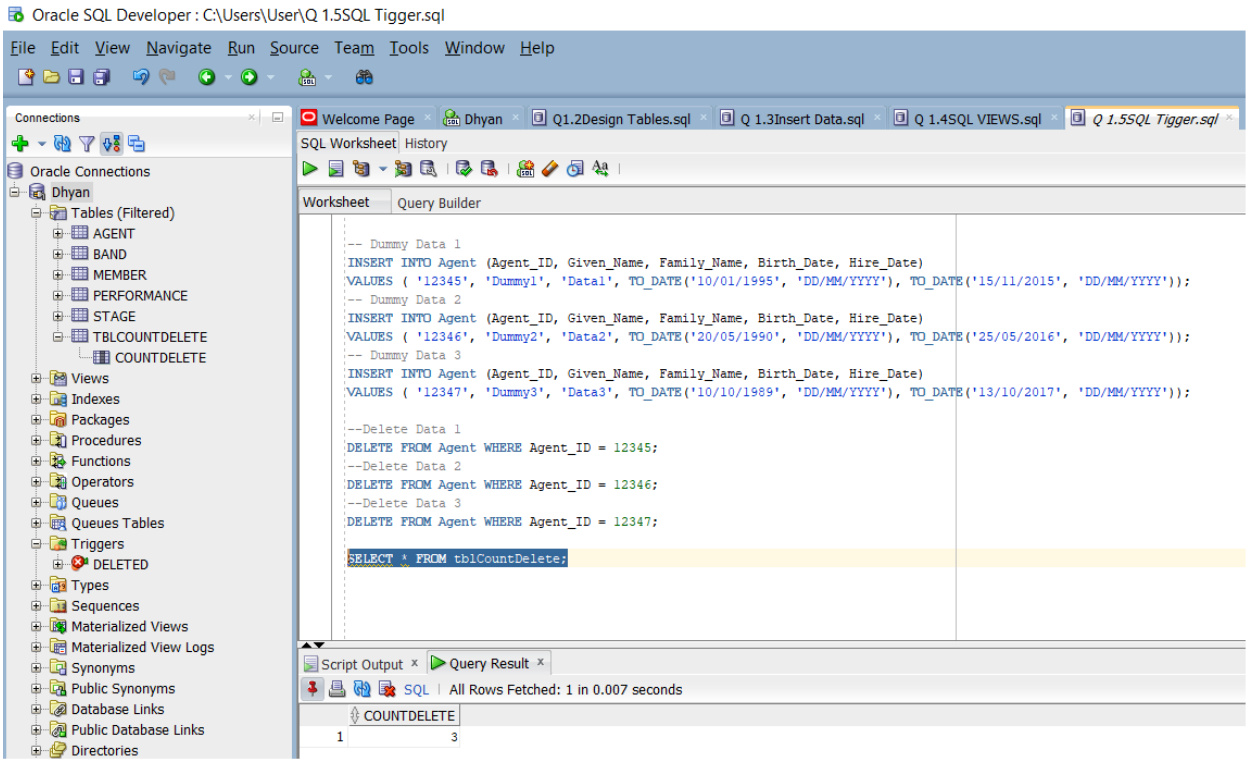


Figure 6

Part -2 MYSQL – PHP Mark 30

Q 2.1 Moving Dataset to MySQL marks 20

Please install WAMP in your computer and run all services under WAMP. Export the dataset from ORACLE to MySQL server. Make sure all tables, constraints, records, views and tiggers are moved to MySQL.

Your report should have screen shot of WAMP server running in your computer, screen of all tables and records, screen shot of script of view and script of tigger in MySQL.

2.1 a) Installation of WAMP or XAMPP and running MySQL and Apache Server

Figure 7 shows screen-shot XAMPP Installed and Running in the computer

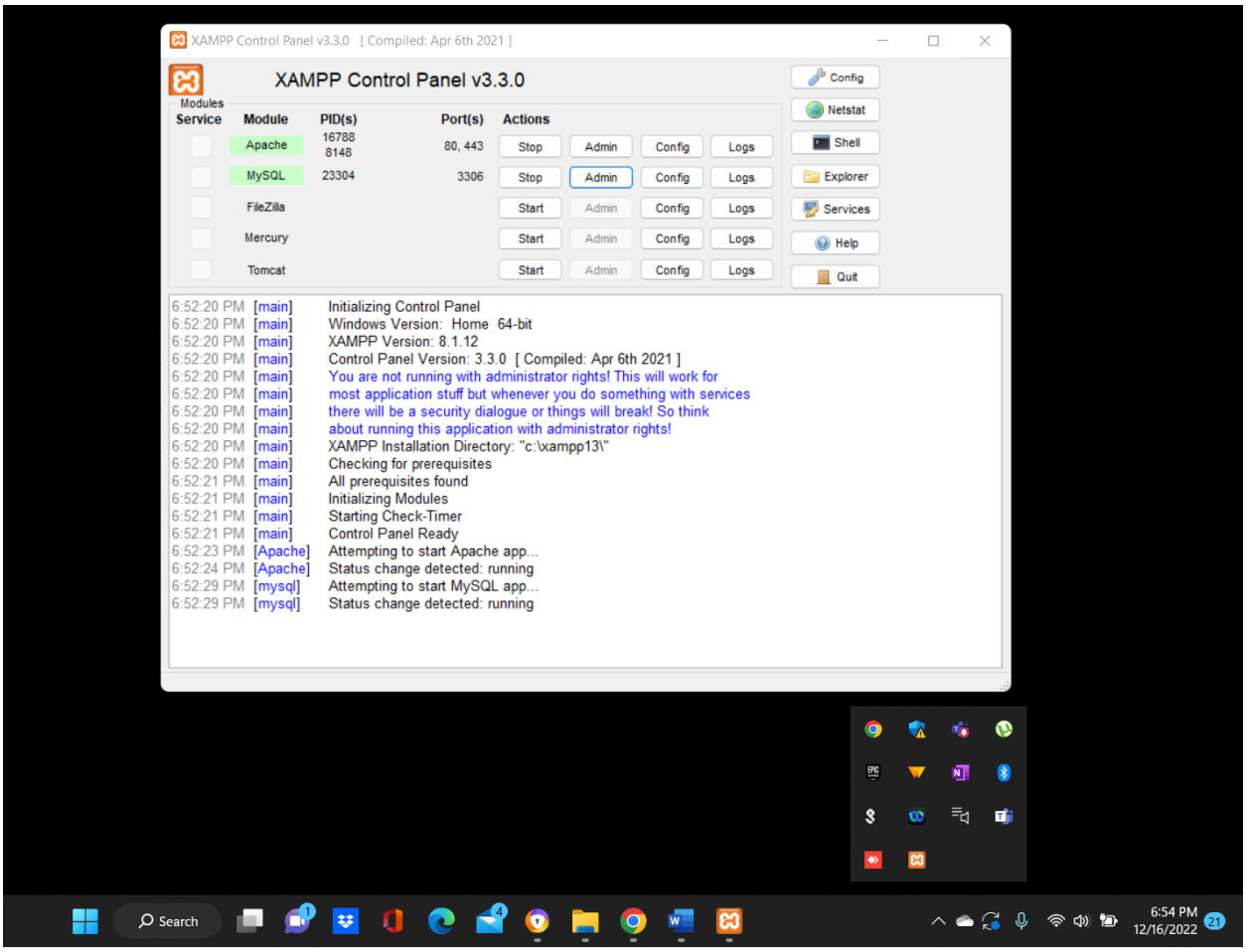


Figure 7

2.1 b) Creating database in MySQL

Figure 8 shows screen-shot of creating a new database new_db in MySQL using phpMyAdmin.

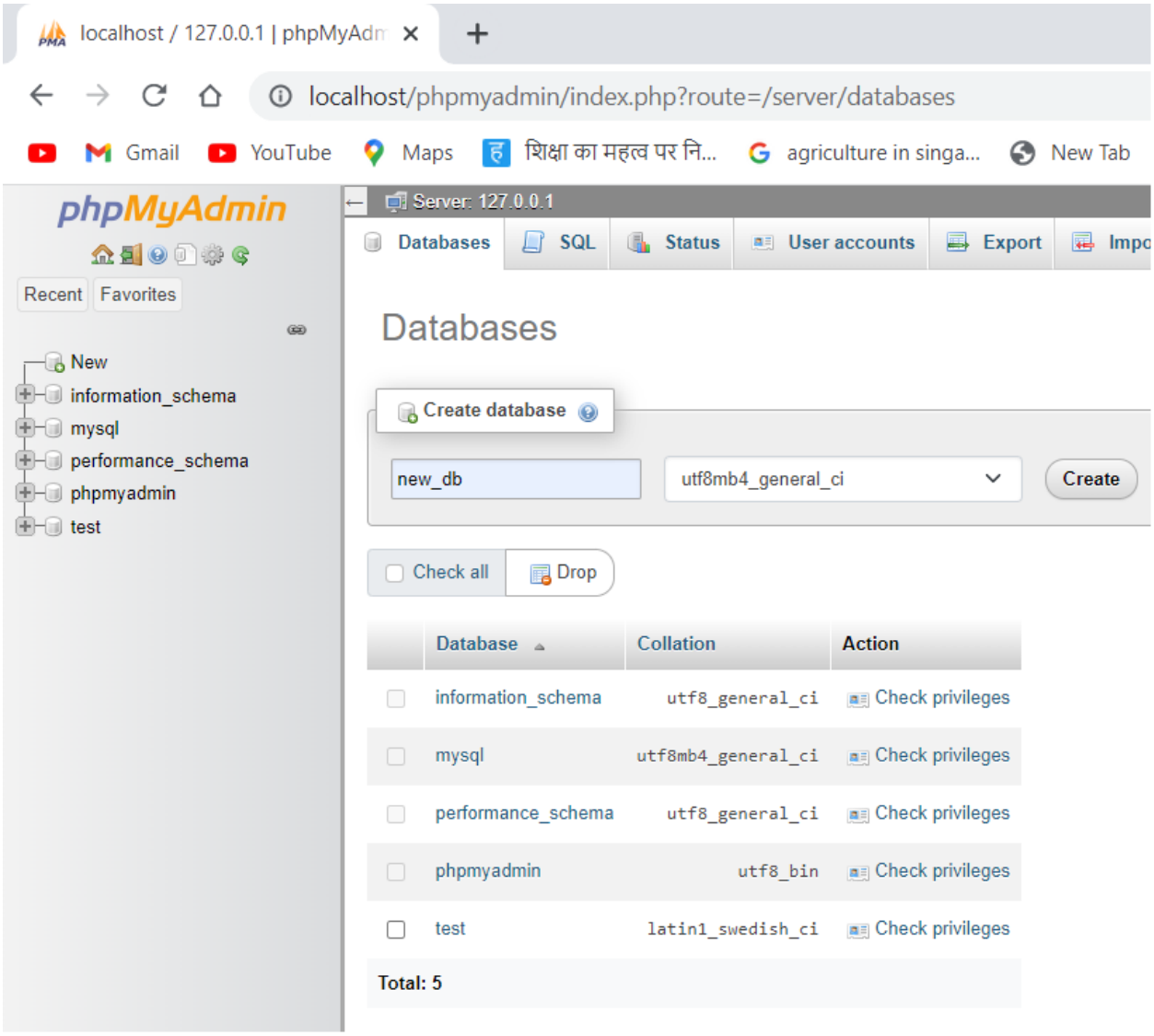


Figure 8

2.1 c) Create table in MySQL

CREATE TABLE Agent(

Agent_ID DECIMAL(20) NOT NULL,

Given_Name VARCHAR(30),

Family_Name VARCHAR(30),

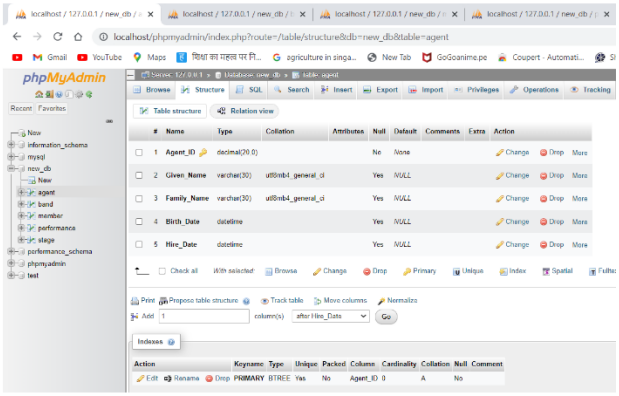
Birth_Date DATETIME,

```
Hire_Date DATETIME,
CONSTRAINT `pk_agentid_agent` PRIMARY KEY(Agent_ID)
);

CREATE TABLE Band(
Band_Name VARCHAR(30) NOT NULL,
DESCRIPTION VARCHAR(300),
Agent_ID DECIMAL(20) NOT NULL,
CONSTRAINT `pk_bandname_band` PRIMARY KEY(Band_Name),
CONSTRAINT `fk_agentid_band` FOREIGN KEY (Agent_ID) REFERENCES Agent(Agent_ID)
);

CREATE TABLE Member(
Member_ID DECIMAL(20) NOT NULL,
Given_Name VARCHAR(30),
Family_Name VARCHAR(30),
Birth_Date DATETIME,
Hire_Date DATETIME,
Job_Type VARCHAR(60),
Band_Name VARCHAR(20) NOT NULL,
CONSTRAINT `pk_memberid_member` PRIMARY KEY(Member_ID),
CONSTRAINT `fk_bandname_member` FOREIGN KEY (Band_Name) REFERENCES Band(Band_Name)
);

CREATE TABLE Stage(
Stage_ID DECIMAL(20) NOT NULL,
Stage_Name VARCHAR(60),
Max_Capacity DECIMAL(25),
CONSTRAINT `pk_stageid_stage` PRIMARY KEY(Stage_ID)
);
```

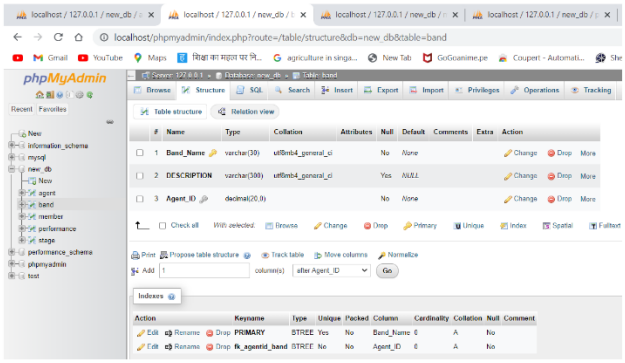
```
CREATE TABLE Performance(
Performance_ID DECIMAL(20) NOT NULL,
Stage_ID DECIMAL(20) NOT NULL,
Act_Date DATETIME,
Start_Time DATETIME,
Finish_Time DATETIME,
Band_Name VARCHAR(30) NOT NULL,
CONSTRAINT `pk_performid_performance` PRIMARY KEY(Performance_ID),
CONSTRAINT `fk_bandname_performance` FOREIGN KEY (Band_Name) REFERENCES Band(Band_Name),
CONSTRAINT `fk_stageid_performance` FOREIGN KEY (Stage_ID) REFERENCES Stage(Stage_ID)
);
```

Figure 9 shows that Agent Table, Band Table and Member Table was successfully created.

Agent Table



Band Table



Member Table

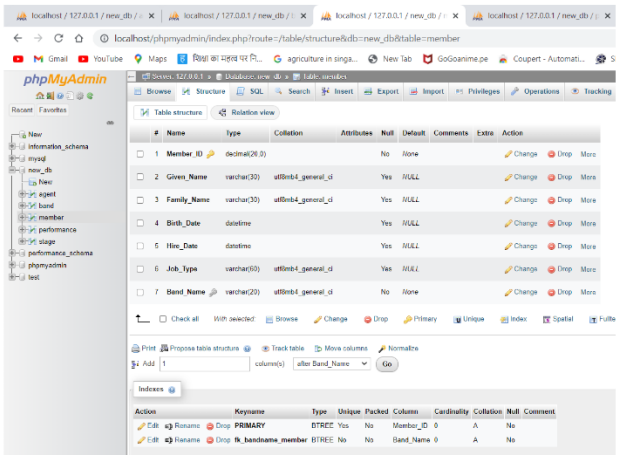
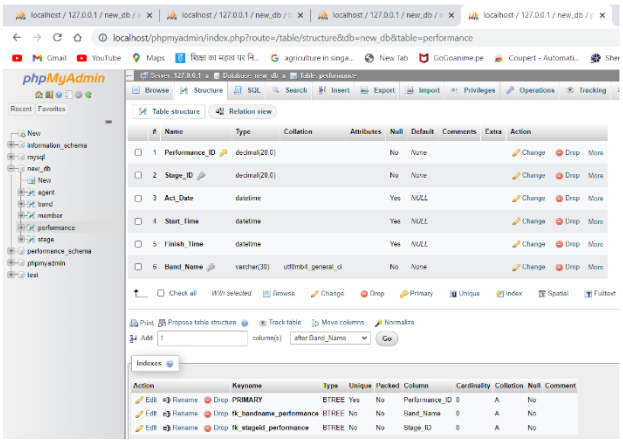


Figure 9

Figure 10 shows that Performance Table and Stage Table was successfully created.

Performance Table



Stage Table

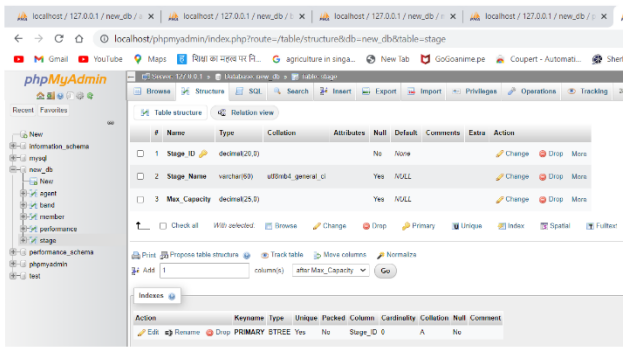


Figure 10

2.1 d) Insert Statement in MySQL

INSERT INTO `agent`(`Agent_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`) VALUES ('2468101214', 'Rowan', 'Atkinson', '1990-01-06', '2020-12-12');

INSERT INTO `agent`(`Agent_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`) VALUES ('3691215182', 'Charlie', 'Chaplin', '1985-04-16', '2018-06-25');

INSERT INTO `band`(`Band_Name`, `DESCRIPTION`, `Agent_ID`) VALUES ('Queen','They are the kings of arena rock and pop rock','2468101214');

INSERT INTO `band`(`Band_Name`, `DESCRIPTION`, `Agent_ID`) VALUES ('Rush','Rush is progressive rock god','2468101214');

INSERT INTO `band`(`Band_Name`, `DESCRIPTION`, `Agent_ID`) VALUES ('Yes','Yes played music in the blues-rock and folk-rock genres','2468101214');

INSERT INTO `band`(`Band_Name`, `DESCRIPTION`, `Agent_ID`) VALUES ('Maroon','American pop rock band from Los Angeles','3691215182');

INSERT INTO `band`(`Band_Name`, `DESCRIPTION`, `Agent_ID`) VALUES ('BTS','Kpop boy band from BigHit Entertainment','3691215182');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21007232', 'Thomas', 'Lennox','1995-11-22', '2014-02-22', 'Musician', 'Yes');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21007233', 'Daniel', 'Lincoln', '1995-08-13', '2016-12-24', 'Keyboardist', 'Yes');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21007234', 'Mary', 'Mason', '1997-10-17', '2016-11-16', 'Bassist', 'Yes');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21007235', 'Jennifer', 'Wiley', '1987-02-27', '2018-04-19', 'Drummer', 'Yes');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21008356', 'Lisa', 'Walker', '2000-02-25', '2018-06-09', 'Musician', 'Maroon');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21008357', 'Steven', 'Wilson', '1994-04-14', '2019-08-17', 'Sound Technician', 'Maroon');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21008358', 'Paul', 'Parker', '1998-11-18', '2020-10-15', 'Event Manager', 'Maroon');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21008359', 'Ronald', 'Miler', '1999-04-08', '2020-03-11', 'Guitarist', 'Maroon');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21009462', 'Sharon', 'Nash', '2002-05-22', '2017-01-23', 'Musician', 'BTS');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21009463', 'Jacob', 'Smith', '1994-08-26', '2018-02-16', 'Singer', 'BTS');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21009465', 'Stephen', 'Stone', '1996-09-08', '2020-04-24', 'Bassist', 'BTS');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21009466', 'Alexander', 'Thompson', '1993-03-06', '2019-06-26', 'Composer', 'BTS');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21005078', 'James', 'Adler', '1996-04-07', '2016-12-02', 'Musician', 'Queen');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21005079', 'Robert', 'Anderson', '1988-07-03', '2021-11-07', 'Composer', 'Queen');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21005080', 'John', 'Beckett', '1990-05-09', '2019-09-12', 'Instrument Technician', 'Queen');

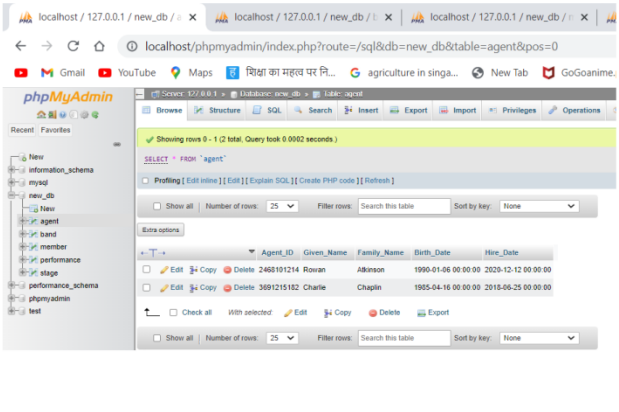
INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21005081', 'Michael', 'Carson', '1994-09-10', '2017-06-18', 'Bassist', 'Queen');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ('21006112', 'David', 'Brady', '2002-11-12', '2018-01-19', 'Musician', 'Rush');

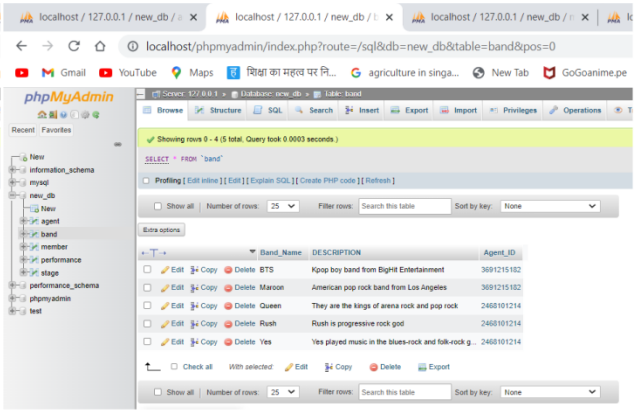
```
INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ( '21006113', 'William', 'Cooper', '1993-02-14', '2020-09-09', 'DJ', 'Rush');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ( '21006114', 'Richard', 'Davis', '2001-01-18', '2019-08-21', 'Drummer', 'Rush');

INSERT INTO `member`(`Member_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`, `Job_Type`, `Band_Name`) VALUES ( '21006115', 'Joseph', 'Dixon', '2000-12-19', '2021-07-05', 'Guitarist', 'Rush');
```

Figure 11 shows that all rows have successfully been inserted in Table Agent, Table Band and Table Member.

2 Rows Inserted in Agent



5 Rows Inserted in Band



20 Rows Inserted in Member

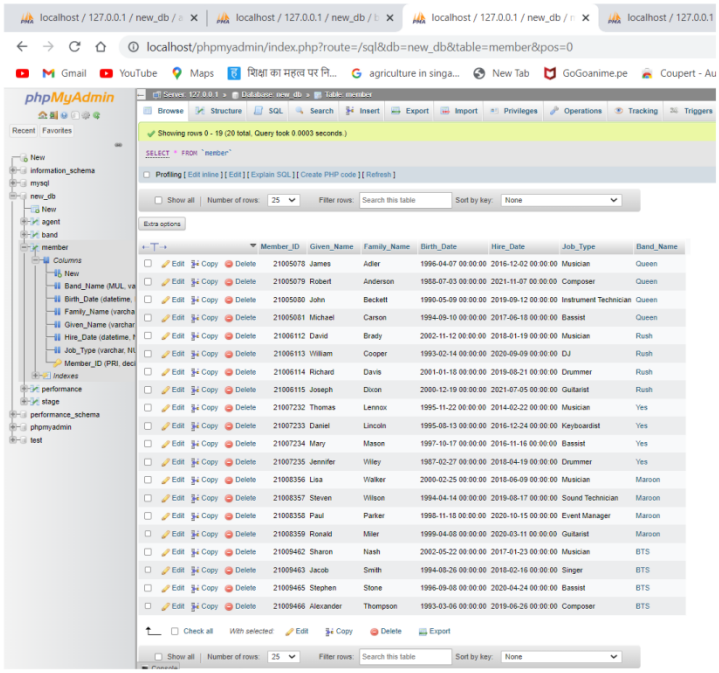


Figure 11

2.1 e) View in MySQL

```
CREATE VIEW AgentJobs AS
SELECT Agent.Given_Name AS "Agent_Name",
COUNT(Job_Type) AS "Number_Of_Jobs"
FROM Agent JOIN Band ON Agent.Agent_ID = Band.Agent_ID
JOIN Member ON Member.Band_Name = Band.Band_Name
GROUP BY Agent.Given_Name;
```

Figure 12 shows the screen shot of SELECT * FROM `agentjobs`;

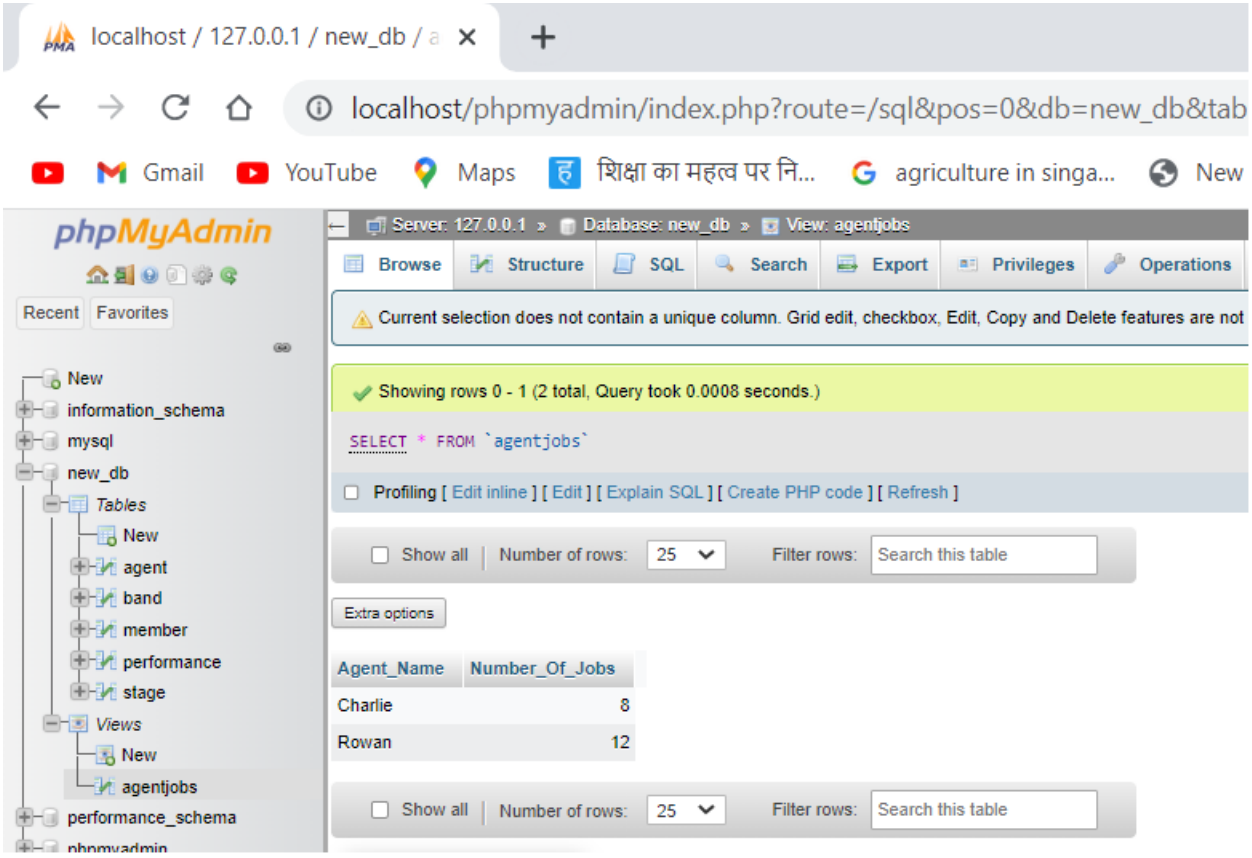


Figure 12

2.1 f) Trigger in MySQL

```
CREATE TABLE tblCountDelete(
countDelete INT
);

INSERT INTO `tblcountdelete`(`countDelete`) VALUES (0);
```

Figure 13 shows the screen shot of tblCountDelete before deleting record from Agent table.

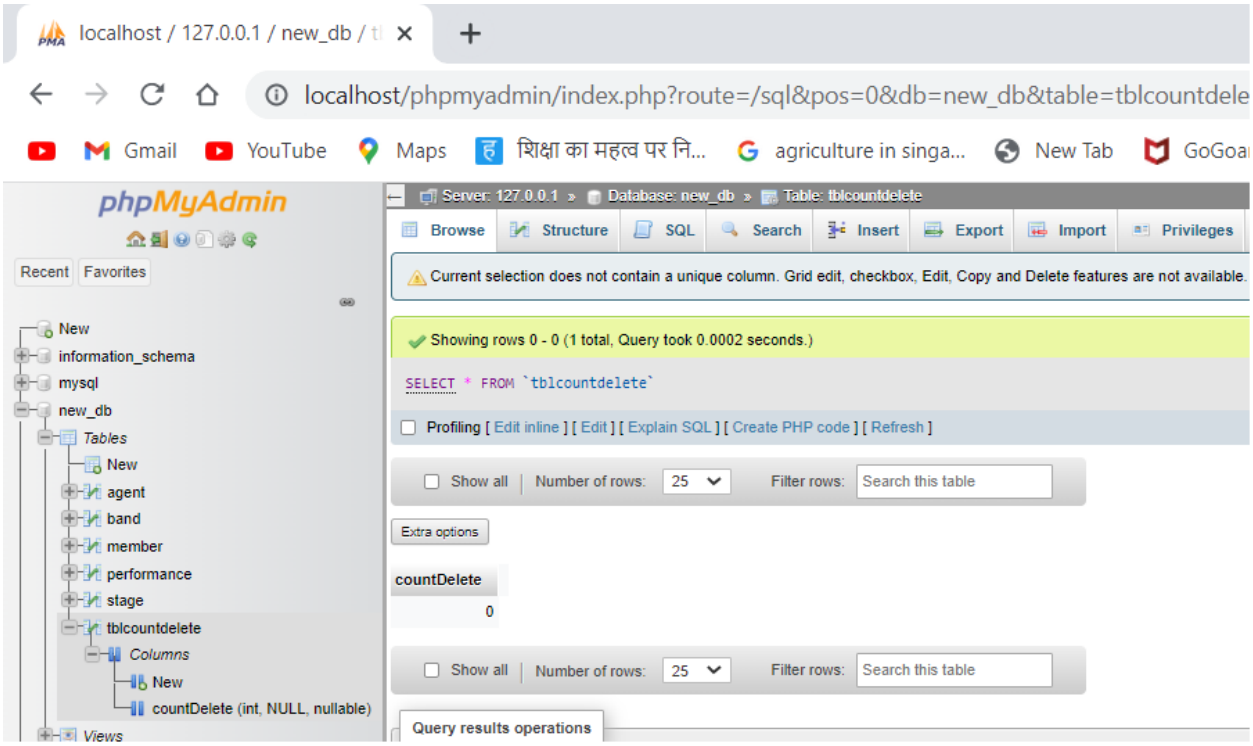


Figure 13

```
DELIMITER $$
CREATE TRIGGER Deleted
AFTER DELETE
ON Agent
FOR EACH ROW
BEGIN
```

```
UPDATE tblCountDelete SET countDelete = countDelete + 1;
END $$
```

2.1 g) SQL to add three Dummy data in Job

```
INSERT INTO `agent`(`Agent_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`)
VALUES ( '12345', 'Dummy1', 'Data1', '1995-01-10', '2015-11-15');
INSERT INTO `agent`(`Agent_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`)
VALUES ( '12346', 'Dummy2', 'Data2', '1990-05-20', '2016-05-25');
INSERT INTO `agent`(`Agent_ID`, `Given_Name`, `Family_Name`, `Birth_Date`, `Hire_Date`)
VALUES ( '12347', 'Dummy3', 'Data3', '1989-10-10', '2017-10-13');
```

2.1 h) Removing the three dummy Data from Job

```
DELETE FROM `agent` WHERE Agent_ID = 12345;
DELETE FROM `agent` WHERE Agent_ID = 12346;
DELETE FROM `agent` WHERE Agent_ID = 12347;
```

2.1 i) Check if trigger is working or not

View the table tblCountDelete:

```
SELECT * FROM tblCountDelete;
```

Figure 14 shows the screen shot of tblCountDelete after deleting record from Agent table.

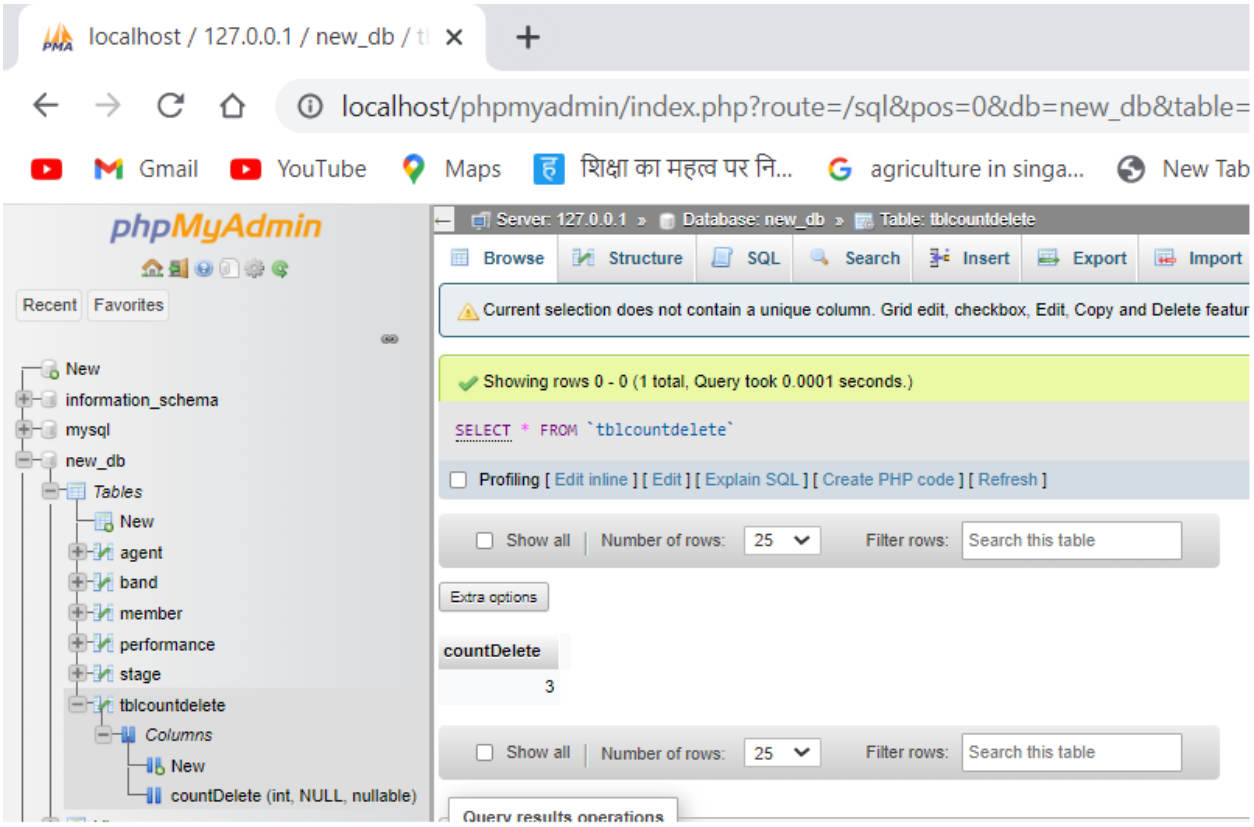


Figure 14

Q 2.2 FETCHING DATA IN PHP marks 10

Write an appropriate PHP script to display results from AgentJobs view in your localserver.

Your report should have PHP script that you used to connect MySQL database and PHP script to display all result from AgentJobs view in webpage. Your report should have screen shot of your webpage displaying the data from AgentJobs view.

Your report should have

- PHP script that you used to connect MySQL dataset
- PHP script that you used to display the results.

- Screen shot of the webpage displaying the result.

2.2 a) PHP script that you used to connect MySQL dataset

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbName = "new_db";
$port = 3306;

// Create connection
$conn = new mysqli($servername, $username, $password, $dbName, $port);

?>
```

2.2 b) PHP script that you used to display the results.

```
<?php
include "dbConnection.php";
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}
$sql = "select * from AgentJobs";
$result = $conn->query($sql);

echo "<html><body>Agent Jobs<hr color='green'>";
echo "<table><tr><th>Agent Name</th><th>Number of Jobs</th></tr>";
```

```
if ($result->num_rows > 0) {

  while($row = $result->fetch_assoc()) {
    echo "<tr><td>" . $row['Agent_Name']."</td><td>". $row['Number_Of_Jobs']."</td></tr>";
  }
  } else {
    echo "0 results";
}
echo "</table></body></html>";
$conn->close();
?>
```

2.2 C) Screen shot of the webpage displaying the result.

Figure 15 shows the screen shot of webpage displaying the result.

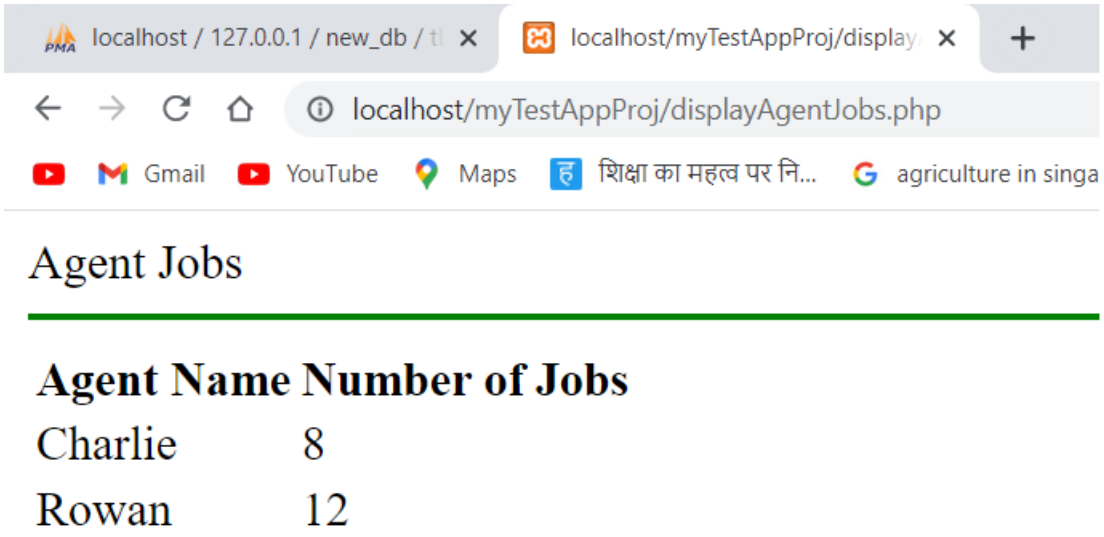


Figure 15

Part -3 MYSQL – JAVA Marks 20

Q 3.1 Using JDBC marks 20

Using JDBC engine in JAVA, display all results from Agent table in the screen.

Note- you can populate data in command prompt. [No need to use java swing].

Your report should have

- Java code to connect to MySQL server using JDBC
- Java code to display results
- Screen shot of the populated results

- **I have used netbean8.1 IDE to develop the application which display data from new_db database to the desktop (console). For this I used JDBC, which is available in mysql-connector-5.1.18.jar.**

Figure 16 shows project configuration with necessary jar files.

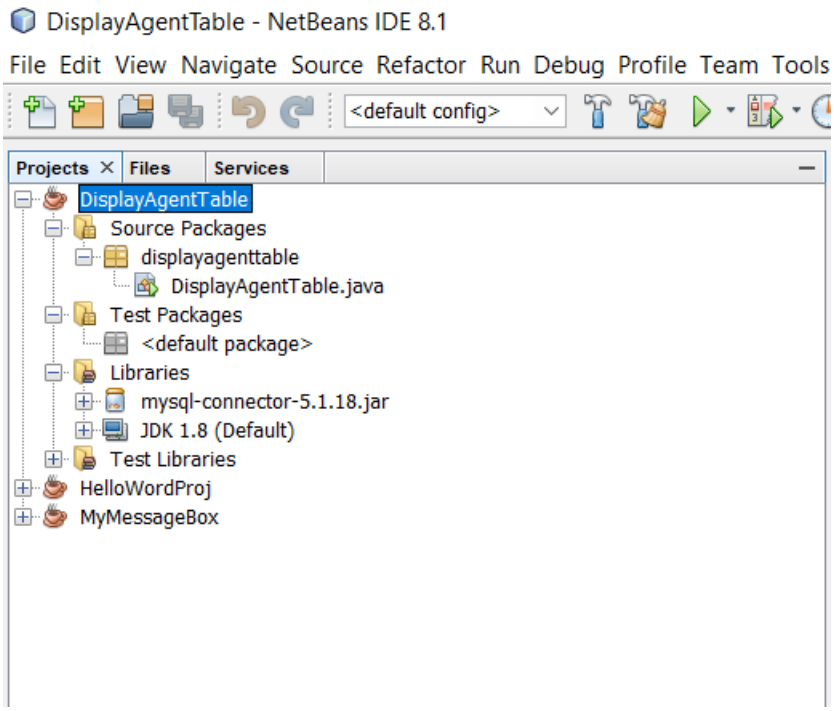


Figure 16

3.1 a) and b)

Java code to connect to coursework01 database in MySQL and display results from Agent table.

package displayagenttable;

```
import java.sql.*;
public class DisplayAgentTable {


   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      // TODO code application logic here
      Connection conn = null;
      try
      {
         Class.forName("com.mysql.jdbc.Driver");
         conn = DriverManager.getConnection("jdbc:mysql://localhost/new_db?user=root");
          Statement stmt = conn.createStatement();
          ResultSet rs = stmt.executeQuery("SELECT Agent_ID, Given_Name, Family_Name,
Birth_Date, Hire_Date FROM Agent");
          while (rs.next()) {
             System.out.print("\nAgent_ID:      " + rs.getString("Agent_ID"));
             System.out.print(", Given_Name:    " + rs.getString("Given_Name"));
             System.out.print(", Family_Name:   " + rs.getString("Family_Name"));
             System.out.print(", Birth_Date:    " + rs.getString("Birth_Date"));
             System.out.print(", Hire_Date:     " + rs.getString("Hire_Date"));
          }
      }catch(Exception e)
      {
         e.printStackTrace();
      }
   }
```

}

3.1 c) Screen shot of the populated results.

Figure 17 shows screen shot of the results from Agent table.

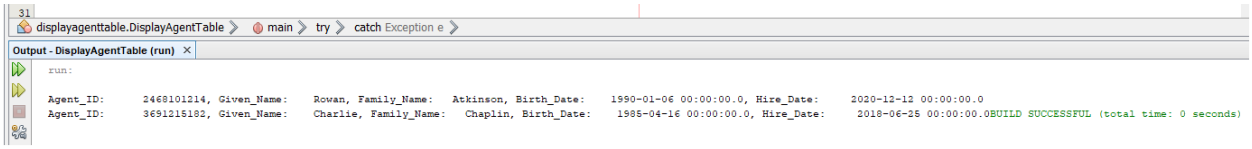


Figure 17

**********END REPORT**********

- I have submitted all the files that I have designed during the project such as Java, Php, Sql and so on.