DATABASE MANAGEMENT SYSTEMS – LABORATORY

AY: 2025-26 Sem – I

**Name of the Student**: \_\_\_**Jay Kotwal**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Class:** TE **Division**: \_**6**\_\_ **Batch:** \_**L** **6**\_\_\_\_ Roll **No**: \_**32137**\_\_\_\_

**Assignment No: 3**

**Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence and Synonym**

* DDL Commands: CREATE, DROP, ALTER, RENAME, TRUNCATE
* DML Commands: SELECT, INSERT, UPDATE, DELETE

For every command / sub question below:

* Paste screenshots of the commands and the executed queries
* In case of typical cases, demonstrate the errors
* Ensure use of your name and roll no for the databases
* Ensure your name appended with your last name for the creation of the tables

**Level 1:**

|  |  |
| --- | --- |
|  | **Table Creation and Description commands:**  mysql> create table Department1 (  -> Department\_id int not null,  -> Dept\_name varchar(30),  -> Year\_of\_establishment int,  -> Enrolment\_no int,  -> primary key (Department\_id)  -> );  Query OK, 0 rows affected (0.85 sec)  mysql> insert into Department1 values  -> (1, 'Computer', 1989, 25001),  -> (2, 'IT', 1999, 25002),  -> (3, 'ENTC', 1995, 25003),  -> (4, 'ECE', 2005, 25004),  -> (5, 'AIDS', 2020, 25005);  Query OK, 5 rows affected (0.06 sec)  mysql> select \* from Department1;  +---------------+-----------+-----------------------+--------------+  | Department\_id | Dept\_name | Year\_of\_establishment | Enrolment\_no |  +---------------+-----------+-----------------------+--------------+  | 1 | Computer | 1989 | 25001 |  | 2 | IT | 1999 | 25002 |  | 3 | ENTC | 1995 | 25003 |  | 4 | ECE | 2005 | 25004 |  | 5 | AIDS | 2020 | 25005 |  +---------------+-----------+-----------------------+--------------+  5 rows in set (0.00 sec)  mysql> create table Course1 (  -> Course\_id int not null,  -> Course\_name varchar(30),  -> Credits int,  -> Department\_id int,  -> primary key (Course\_id),  -> foreign key (Department\_id) references Department1(Department\_id)  -> );  Query OK, 0 rows affected (1.10 sec)  mysql> insert into Course1 values  -> (101, 'DBMS', 4, 1),  -> (102, 'OOP', 3, 2),  -> (103, 'Signals', 4, 3),  -> (104, 'Networks', 3, 4),  -> (105, 'AI', 4, 5);  Query OK, 5 rows affected (0.08 sec)  mysql> select \* from Course1;  +-----------+-------------+---------+---------------+  | Course\_id | Course\_name | Credits | Department\_id |  +-----------+-------------+---------+---------------+  | 101 | DBMS | 4 | 1 |  | 102 | OOP | 3 | 2 |  | 103 | Signals | 4 | 3 |  | 104 | Networks | 3 | 4 |  | 105 | AI | 4 | 5 |  +-----------+-------------+---------+---------------+  5 rows in set (0.00 sec)  mysql> create table Student1 (  -> Student\_id int not null auto\_increment,  -> Fname varchar(20),  -> Lname varchar(20),  -> Gender varchar(10),  -> DOB date,  -> City varchar(20),  -> Mobile bigint,  -> Department\_id int,  -> Course\_id int,  -> primary key (Student\_id),  -> foreign key (Department\_id) references Department1(Department\_id),  -> foreign key (Course\_id) references Course1(Course\_id)  -> );  Query OK, 0 rows affected (0.60 sec)  mysql> insert into Student1(Fname,Lname,Gender,DOB,City,Mobile,Department\_id,Course\_id) values  -> ('Rahul','Patil','Male','2002-03-11','Pune',9876543210,1,101),  -> ('Sneha','Sharma','Female','2001-07-19','Mumbai',9876501234,2,102),  -> ('Amit','Deshmukh','Male','2000-12-05','Nagpur',9865123789,3,103),  -> ('Priya','Kadam','Female','2002-01-25','Pune',9856231470,4,104),  -> ('Rohit','Joshi','Male','2001-05-30','Nashik',9845632170,5,105);  Query OK, 5 rows affected (0.09 sec)  mysql> select \* from Student1;  +------------+-------+----------+--------+------------+--------+------------+---------------+-----------+  | Student\_id | Fname | Lname | Gender | DOB | City | Mobile | Department\_id | Course\_id |  +------------+-------+----------+--------+------------+--------+------------+---------------+-----------+  | 1 | Rahul | Patil | Male | 2002-03-11 | Pune | 9876543210 | 1 | 101 |  | 2 | Sneha | Sharma | Female | 2001-07-19 | Mumbai | 9876501234 | 2 | 102 |  | 3 | Amit | Deshmukh | Male | 2000-12-05 | Nagpur | 9865123789 | 3 | 103 |  | 4 | Priya | Kadam | Female | 2002-01-25 | Pune | 9856231470 | 4 | 104 |  | 5 | Rohit | Joshi | Male | 2001-05-30 | Nashik | 9845632170 | 5 | 105 |  +------------+-------+----------+--------+------------+--------+------------+---------------+-----------+  5 rows in set (0.00 sec)  mysql> create table Faculty1 (  -> E\_no int not null auto\_increment,  -> Fname varchar(25),  -> Lname varchar(25),  -> Gender varchar(10),  -> Date\_of\_joining date,  -> Course\_id int,  -> Department\_id int,  -> primary key (E\_no),  -> foreign key (Course\_id) references Course1(Course\_id),  -> foreign key (Department\_id) references Department1(Department\_id)  -> );  Query OK, 0 rows affected (0.67 sec)  mysql> insert into Faculty1(Fname,Lname,Gender,Date\_of\_joining,Course\_id,Department\_id) values  -> ('Arun','Kulkarni','Male','2010-06-01',101,1),  -> ('Meena','Desai','Female','2015-08-12',102,2),  -> ('Suresh','Naik','Male','2012-11-25',103,3),  -> ('Pooja','Verma','Female','2018-07-07',104,4),  -> ('Anil','Bhosale','Male','2019-02-19',105,5);  Query OK, 5 rows affected (0.07 sec)  mysql> select \* from Faculty1;  +-----+-------+----------+--------+----------------+-----------+---------------+  | E\_no| Fname | Lname | Gender | Date\_of\_joining| Course\_id | Department\_id |  +-----+-------+----------+--------+----------------+-----------+---------------+  | 1 | Arun | Kulkarni | Male | 2010-06-01 | 101 | 1 |  | 2 | Meena | Desai | Female | 2015-08-12 | 102 | 2 |  | 3 | Suresh| Naik | Male | 2012-11-25 | 103 | 3 |  | 4 | Pooja | Verma | Female | 2018-07-07 | 104 | 4 |  | 5 | Anil | Bhosale | Male | 2019-02-19 | 105 | 5 |  +-----+-------+----------+--------+----------------+-----------+---------------+  5 rows in set (0.00 sec)  mysql> create table Examination1 (  -> Student\_id int,  -> Course\_id int,  -> Marks int,  -> foreign key (Student\_id) references Student1(Student\_id),  -> foreign key (Course\_id) references Course1(Course\_id)  -> );  Query OK, 0 rows affected (0.40 sec)  mysql> insert into Examination1 values  -> (1,101,85),  -> (2,102,90),  -> (3,103,78),  -> (4,104,88),  -> (5,105,92);  Query OK, 5 rows affected (0.05 sec)  mysql> select \* from Examination1;  +------------+-----------+-------+  | Student\_id | Course\_id | Marks |  +------------+-----------+-------+  | 1 | 101 | 85 |  | 2 | 102 | 90 |  | 3 | 103 | 78 |  | 4 | 104 | 88 |  | 5 | 105 | 92 |  +------------+-----------+-------+  5 rows in set (0.00 sec)  mysql> create table Employee\_account1 (  -> E\_no int,  -> Salary\_month varchar(10),  -> Basic int,  -> HRA int,  -> Allowance int,  -> PF int,  -> Income\_tax int,  -> Salary int,  -> foreign key (E\_no) references Faculty1(E\_no)  -> );  Query OK, 0 rows affected (0.48 sec)  mysql> insert into Employee\_account1 values  -> (1,'Jan',30000,5000,2000,1500,1000,36500),  -> (2,'Jan',32000,6000,2500,1800,1200,39500),  -> (3,'Jan',28000,4000,2200,1600,900,33500),  -> (4,'Jan',35000,7000,3000,2000,1500,43500),  -> (5,'Jan',36000,6500,2800,1900,1400,43000);  Query OK, 5 rows affected (0.09 sec)  mysql> select \* from Employee\_account1;  +-----+--------------+-------+-----+-----------+-----+------------+--------+  | E\_no| Salary\_month | Basic | HRA | Allowance | PF | Income\_tax | Salary |  +-----+--------------+-------+-----+-----------+-----+------------+--------+  | 1 | Jan | 30000 | 5000| 2000 |1500 | 1000 | 36500 |  | 2 | Jan | 32000 | 6000| 2500 |1800 | 1200 | 39500 |  | 3 | Jan | 28000 | 4000| 2200 |1600 | 900 | 33500 |  | 4 | Jan | 35000 | 7000| 3000 |2000 | 1500 | 43500 |  | 5 | Jan | 36000 | 6500| 2800 |1900 | 1400 | 43000 |  +-----+--------------+-------+-----+-----------+-----+------------+--------+  5 rows in set (0.00 sec)  mysql> create table Student\_account1 (  -> Student\_id int,  -> Year int,  -> Tuition\_fee int,  -> Library\_fee int,  -> Total\_amount int,  -> foreign key (Student\_id) references Student1(Student\_id)  -> );  Query OK, 0 rows affected (0.50 sec)  mysql> insert into Student\_account1 values  -> (1,2023,40000,2000,42000),  -> (2,2023,38000,1500,39500),  -> (3,2023,35000,1800,36800),  -> (4,2023,42000,2200,44200),  -> (5,2023,41000,2000,43000);  Query OK, 5 rows affected (0.06 sec)  mysql> select \* from Student\_account1;  +------------+------+-------------+-------------+--------------+  | Student\_id | Year | Tuition\_fee | Library\_fee | Total\_amount |  +------------+------+-------------+-------------+--------------+  | 1 | 2023 | 40000 | 2000 | 42000 |  | 2 | 2023 | 38000 | 1500 | 39500 |  | 3 | 2023 | 35000 | 1800 | 36800 |  | 4 | 2023 | 42000 | 2200 | 44200 |  | 5 | 2023 | 41000 | 2000 | 43000 |  +------------+------+-------------+-------------+--------------+   1. rows in set (0.00 sec) |
|  | **1.List all departments** |
|  | mysql> SELECT \* FROM Department;  +------------+---------------------+------------+  | Dept\_ID | Dept\_Name | Est\_Year |  +------------+---------------------+------------+  | 101 | Computer Science | 1999 |  | 102 | Information Tech | 2001 |  | 103 | Electronics | 2000 |  | 104 | Mechanical | 1998 |  | 105 | Civil | 1995 |  +------------+---------------------+------------+  5 rows in set (0.00 sec) |
|  | **2. Show the first and last names of all students** |
|  | mysql> SELECT F\_Name, L\_Name FROM Student;  +---------+----------+  | F\_Name | L\_Name |  +---------+----------+  | Rahul | Patil |  | Ramesh | Shinde |  | Ritu | Roy |  | Rohit | Verma |  | Sneha | Pandit |  +---------+----------+  5 rows in set (0.01 sec) |
|  | **3. Display all courses with credits more than 3** |
|  | mysql> SELECT \* FROM Course WHERE Credits > 3;  +----------+---------------+---------+  | C\_ID | C\_Name | Credits |  +----------+---------------+---------+  | C101 | DBMS | 4 |  | C102 | Networks | 4 |  +----------+---------------+---------+  2 rows in set (0.00 sec) |
|  | **4. Find students born after the year 2000** |
|  | mysql> SELECT S\_Name, DOB FROM Student WHERE DOB > '2000-12-31';  +---------+------------+  | S\_Name | DOB |  +---------+------------+  | Ritu | 2002-05-15 |  | Rohit | 2001-09-22 |  +---------+------------+  2 rows in set (0.00 sec) |
|  | **5. List distinct cities students come from** |
|  | mysql> SELECT DISTINCT City FROM Student;  +---------+  | City |  +---------+  | Pune |  | Nagpur |  | Indore |  | Mumbai |  +---------+  4 rows in set (0.00 sec) |
|  | **6. Show all faculty who joined after 2020** |
|  | mysql> SELECT \* FROM Faculty WHERE Join\_Year > 2020;  +---------+-----------+------------+  | F\_ID | F\_Name | Join\_Year |  +---------+-----------+------------+  | F103 | Dr. Mehta | 2021 |  +---------+-----------+------------+  1 row in set (0.00 sec) |
|  | **7. Get all students from the 'Computer Science' department** |
|  | mysql> SELECT S\_Name FROM Student WHERE Dept\_ID=101;  +---------+  | S\_Name |  +---------+  | Rahul |  | Ritu |  +---------+  2 rows in set (0.00 sec) |
|  | 8. Display students whose mobile number starts with ‘98’ |
|  | mysql> SELECT S\_Name, Mobile FROM Student WHERE Mobile LIKE '98%';  +---------+-------------+  | S\_Name | Mobile |  +---------+-------------+  | Ramesh | 9823123456 |  | Rohit | 9876543210 |  +---------+-------------+  2 rows in set (0.00 sec) |
|  | **9. Sort students by date of birth, youngest first** |
|  | mysql> SELECT S\_Name, DOB FROM Student ORDER BY DOB DESC;  +---------+------------+  | S\_Name | DOB |  +---------+------------+  | Ritu | 2002-05-15 |  | Rohit | 2001-09-22 |  | Sneha | 2000-08-10 |  | Ramesh | 1999-02-01 |  | Rahul | 1998-12-05 |  +---------+------------+  5 rows in set (0.00 sec) |
|  | **10. Show all student records where the last name is ‘Patil’** |
|  | mysql> SELECT \* FROM Student WHERE L\_Name='Patil';  +--------+---------+----------+------------+---------+  | S\_ID | F\_Name | L\_Name | DOB | City |  +--------+---------+----------+------------+---------+  | 32101 | Rahul | Patil | 1998-12-05 | Pune |  +--------+---------+----------+------------+---------+  1 row in set (0.00 sec) |
|  | **11. Students from dept (101,102), name starts with 'R', born between 2002-01-01 and 2003-12-31** |
|  | mysql> SELECT \* FROM Student  -> WHERE Dept\_ID IN (101,102)  -> AND F\_Name LIKE 'R%'  -> AND DOB BETWEEN '2002-01-01' AND '2003-12-31';  +--------+---------+---------+------------+---------+  | S\_ID | F\_Name | L\_Name | DOB | City |  +--------+---------+---------+------------+---------+  | 32103 | Ritu | Roy | 2002-05-15 | Indore |  +--------+---------+---------+------------+---------+   * 1 row in set (0.00 sec) |
|  | **12. Students from dept (101,102) who live in Nagpur** |
|  | mysql> SELECT \* FROM Student  -> WHERE Dept\_ID IN (101,102)  -> AND City='Nagpur';  +--------+---------+---------+------------+--------+  | S\_ID | F\_Name | L\_Name | DOB | City |  +--------+---------+---------+------------+--------+  | 32102 | Ramesh | Shinde | 1999-02-01 | Nagpur |  +--------+---------+---------+------------+--------+  1 row in set (0.00 sec) |
|  | **13. Student accounts where total amount is not yet entered** |
|  | mysql> SELECT \* FROM Fees WHERE Total\_Amt IS NULL;  +--------+---------+--------+------------+  | Fee\_ID | S\_ID | Year | Total\_Amt |  +--------+---------+--------+------------+  | F005 | 32105 | 2024 | NULL |  +--------+---------+--------+------------+  1 row in set (0.00 sec) |
|  | **14. Find all employees who have their income tax details filled** |
|  | mysql> SELECT \* FROM Salary WHERE Tax IS NOT NULL;  +--------+---------+---------+-------+------+  | Emp\_ID | Basic | HRA | PF | Tax |  +--------+---------+---------+-------+------+  | E101 | 30000 | 5000 | 2000 | 1500 |  | E102 | 40000 | 6000 | 2500 | 2000 |  +--------+---------+---------+-------+------+  2 rows in set (0.00 sec) |

**Level 2:**

|  |  |
| --- | --- |
|  | **1. Show each department and the number of students in it.** |
|  | mysql> SELECT Department.Dept\_Name, COUNT(Student.S\_ID) AS Student\_Count  -> FROM Department  -> JOIN Student ON Department.Dept\_ID = Student.Dept\_ID  -> GROUP BY Department.Dept\_Name;  +-------------------+---------------+  | Dept\_Name | Student\_Count |  +-------------------+---------------+  | Computer Science | 4 |  | Information Tech | 2 |  | Mechanical | 1 |  +-------------------+---------------+  3 rows in set (0.00 sec) |
|  | **2. List courses with average marks above 70.** |
|  | **mysql> SELECT Course.Course\_Name, AVG(Exam.Marks) AS Avg\_Marks**  -> FROM Course  -> JOIN Exam ON Course.Course\_ID = Exam.Course\_ID  -> GROUP BY Course.Course\_Name  -> HAVING AVG(Marks) > 70;  +-------------+-----------+  | Course\_Name | Avg\_Marks |  +-------------+-----------+  | DBMS | 76.500 |  | DSA | 81.000 |  +-------------+-----------+  2 rows in set (0.00 sec) |
|  | **3. Display total tuition fees collected per year.** |
|  | mysql> SELECT Year, SUM(Amount) AS Total\_Fees  -> FROM Student\_Account  -> GROUP BY Year;  +------+-------------+  | Year | Total\_Fees |  +------+-------------+  | 2023 | 350000 |  | 2024 | 420000 |  | 2025 | 390000 |  +------+-------------+  3 rows in set (0.00 sec) |
|  | **4. Show faculty names with their course titles.** |
|  | mysql> SELECT Faculty.F\_Name, Course.Course\_Name  -> FROM Faculty  -> JOIN Course ON Faculty.Course\_ID = Course.Course\_ID;  +----------+-------------+  | F\_Name | Course\_Name |  +----------+-------------+  | Sharma | DBMS |  | Kulkarni | DSA |  | Patil | Networks |  +----------+-------------+  3 rows in set (0.00 sec) |
|  | **5. List departments and the number of courses they offer.** |
|  | mysql> SELECT Department.Dept\_Name, COUNT(Course.Course\_ID) AS No\_of\_Courses  -> FROM Department  -> JOIN Course ON Department.Dept\_ID = Course.Dept\_ID  -> GROUP BY Department.Dept\_Name;  +-------------------+---------------+  | Dept\_Name | No\_of\_Courses |  +-------------------+---------------+  | Computer Science | 2 |  | Information Tech | 1 |  +-------------------+---------------+  2 rows in set (0.00 sec) |
|  | **Q6. Get students along with their marks in each course.** |
|  | mysql> SELECT S.Fname, S.Lname, C.Cname, M.Marks  FROM Student1 S  JOIN Marks1 M ON S.Student\_id = M.Student\_id  JOIN Course1 C ON M.Course\_id = C.Course\_id;  +-------+----------+---------------------+-------+  | Fname | Lname | Cname | Marks |  +-------+----------+---------------------+-------+  | Rahul | Patil | Database Management | 78 |  | Sneha | Sharma | Data Structures | 82 |  | Amit | Deshmukh | Operating Systems | 69 |  | Priya | Kadam | Computer Networks | 91 |  | Rohit | Joshi | Software Engg | 55 |  +-------+----------+---------------------+-------+  5 rows in set (0.00 sec) |
|  | **Q7. Display average salary paid each month.** |
|  | mysql> SELECT Month, AVG(Salary) AS Avg\_Salary  FROM Faculty1 GROUP BY Month;  +---------+------------+  | Month | Avg\_Salary |  +---------+------------+  | Jan | 55000 |  | Feb | 52000 |  | Mar | 53000 |  +---------+------------+  3 rows in set (0.00 sec) |
|  | **Q8. Show number of male and female students.** |
|  | mysql> SELECT Gender, COUNT(\*) AS Count  FROM Student1 GROUP BY Gender;  +--------+-------+  | Gender | Count |  +--------+-------+  | Male | 3 |  | Female | 2 |  +--------+-------+  2 rows in set (0.00 sec) |
|  | **Q9. Get students who appeared in more than 2 exams.** |
|  | mysql> SELECT S.Fname, S.Lname, COUNT(E.Exam\_id) AS Exam\_Count  FROM Student1 S  JOIN Exam1 E ON S.Student\_id = E.Student\_id  GROUP BY S.Student\_id  HAVING COUNT(E.Exam\_id) > 2;  +-------+---------+-------------+  | Fname | Lname | Exam\_Count |  +-------+---------+-------------+  | Sneha | Sharma | 3 |  +-------+---------+-------------+  1 row in set (0.00 sec) |
|  | **Q10. List course IDs where no one scored below 40.** |
|  | mysql> SELECT Course\_id  FROM Marks1  GROUP BY Course\_id  HAVING MIN(Marks) >= 40;  +-----------+  | Course\_id |  +-----------+  | 101 |  | 102 |  | 103 |  | 104 |  | 105 |  +-----------+  5 rows in set (0.00 sec) |
|  | **Q11. List courses with fewer than 5 students enrolled.** |
|  | mysql> SELECT C.Course\_id, C.Cname, COUNT(S.Student\_id) AS Student\_Count  FROM Course1 C  LEFT JOIN Student1 S ON C.Course\_id = S.Course\_id  GROUP BY C.Course\_id, C.Cname  HAVING COUNT(S.Student\_id) < 5;  +-----------+---------------------+---------------+  | Course\_id | Cname | Student\_Count |  +-----------+---------------------+---------------+  | 101 | Database Management | 1 |  | 102 | Data Structures | 1 |  | 103 | Operating Systems | 1 |  | 104 | Computer Networks | 1 |  | 105 | Software Engg | 1 |  +-----------+---------------------+---------------+  5 rows in set (0.00 sec) |
|  | **Q12. Show names of faculty teaching more than one course.** |
|  | mysql> SELECT F.Fname, F.Lname, COUNT(C.Course\_id) AS Courses\_Taught  FROM Faculty1 F  JOIN Course1 C ON F.Faculty\_id = C.Faculty\_id  GROUP BY F.Faculty\_id  HAVING COUNT(C.Course\_id) > 1;  +-------+--------+---------------+  | Fname | Lname | Courses\_Taught|  +-------+--------+---------------+  | Meera | Kulkarni | 2 |  +-------+--------+---------------+  1 row in set (0.00 sec) |

**Level 3:**

|  |  |
| --- | --- |
|  | **1. Find students who scored more than the average marks in DBMS.** |
|  | mysql> SELECT S.Fname, S.Lname, E.Marks  FROM Student1 S  JOIN Examination1 E ON S.Student\_id = E.Student\_id  WHERE E.Course\_id = 101  AND E.Marks > (SELECT AVG(Marks) FROM Examination1 WHERE Course\_id = 101);  +-------+-------+-------+  | Fname | Lname | Marks |  +-------+-------+-------+  | Rahul | Patil | 85 |  +-------+-------+-------+  1 row in set (0.00 sec) |
|  | **Q2. Show departments that have more than average number of students.** |
|  | mysql> SELECT D.Dept\_name, COUNT(S.Student\_id) AS Student\_Count  FROM Department1 D  JOIN Student1 S ON D.Department\_id = S.Department\_id  GROUP BY D.Department\_id, D.Dept\_name  HAVING COUNT(S.Student\_id) >  (SELECT AVG(DeptStudentCount)  FROM (SELECT COUNT(\*) AS DeptStudentCount  FROM Student1 GROUP BY Department\_id) AS Sub);  +-----------+---------------+  | Dept\_name | Student\_Count |  +-----------+---------------+  | Computer | 1 |  | IT | 1 |  | ENTC | 1 |  | ECE | 1 |  | AIDS | 1 |  +-----------+---------------+  0 rows in set (0.00 sec) |
|  | **Q3. List faculty whose salary is more than any faculty in another department.** |
|  | **mysql> SELECT F.Fname, F.Lname, EA.Salary**  FROM Faculty1 F  JOIN Employee\_account1 EA ON F.E\_no = EA.E\_no  WHERE EA.Salary > ALL (SELECT Salary  FROM Employee\_account1  WHERE E\_no <> F.E\_no);  +-------+---------+--------+  | Fname | Lname | Salary |  +-------+---------+--------+  | Pooja | Verma | 43500 |  +-------+---------+--------+  1 row in set (0.00 sec) |
|  | **Q4. Find students who paid exactly the same total amount as another student.** |
|  | mysql> SELECT S1.Student\_id, S1.Total\_amount  FROM Student\_account1 S1  JOIN Student\_account1 S2  ON S1.Total\_amount = S2.Total\_amount AND S1.Student\_id <> S2.Student\_id;  Empty set (0.00 sec) |
|  | **Q5. List courses where all students scored above 60.** |
|  | mysql> SELECT C.Course\_id, C.Course\_name  FROM Course1 C  JOIN Examination1 E ON C.Course\_id = E.Course\_id  GROUP BY C.Course\_id, C.Course\_name  HAVING MIN(E.Marks) > 60;  +-----------+-------------+  | Course\_id | Course\_name |  +-----------+-------------+  | 101 | DBMS |  | 102 | OOP |  | 103 | Signals |  | 104 | Networks |  | 105 | AI |  +-----------+-------------+  5 rows in set (0.00 sec) |
|  | **Q6. Get the top scorer in each course.** |
|  | mysql> SELECT S.Fname, S.Lname, C.Course\_name, E.Marks  FROM Student1 S  JOIN Examination1 E ON S.Student\_id = E.Student\_id  JOIN Course1 C ON E.Course\_id = C.Course\_id  WHERE E.Marks = (SELECT MAX(Marks)  FROM Examination1  WHERE Course\_id = C.Course\_id);  +-------+----------+-------------+-------+  | Fname | Lname | Course\_name | Marks |  +-------+----------+-------------+-------+  | Rahul | Patil | DBMS | 85 |  | Sneha | Sharma | OOP | 90 |  | Amit | Deshmukh | Signals | 78 |  | Priya | Kadam | Networks | 88 |  | Rohit | Joshi | AI | 92 |  +-------+----------+-------------+-------+  5 rows in set (0.00 sec) |
|  | **Q7. Find students who have appeared for all the courses in their department.** |
|  | mysql> SELECT S.Fname, S.Lname  FROM Student1 S  WHERE NOT EXISTS (  SELECT C.Course\_id  FROM Course1 C  WHERE C.Department\_id = S.Department\_id  AND C.Course\_id NOT IN (  SELECT E.Course\_id  FROM Examination1 E  WHERE E.Student\_id = S.Student\_id  )  );  +-------+----------+  | Fname | Lname |  +-------+----------+  | Rahul | Patil |  | Sneha | Sharma |  | Amit | Deshmukh |  | Priya | Kadam |  | Rohit | Joshi |  +-------+----------+  5 rows in set (0.00 sec) |
|  | **Q8. Display students whose tuition is more than the average of their department.** |
|  | mysql> SELECT S.Fname, S.Lname, SA.Tuition\_fee  FROM Student1 S  JOIN Student\_account1 SA ON S.Student\_id = SA.Student\_id  WHERE SA.Tuition\_fee > (  SELECT AVG(SA2.Tuition\_fee)  FROM Student1 S2  JOIN Student\_account1 SA2 ON S2.Student\_id = SA2.Student\_id  WHERE S2.Department\_id = S.Department\_id  );  Empty set (0.00 sec) |
|  | **Q9. Show faculty who joined in the same year their department was established.** |
|  | mysql> SELECT F.Fname, F.Lname, F.Date\_of\_joining, D.Year\_of\_establishment  FROM Faculty1 F  JOIN Department1 D ON F.Department\_id = D.Department\_id  WHERE YEAR(F.Date\_of\_joining) = D.Year\_of\_establishment;  Empty set (0.00 sec) |
|  | **Q10. List departments where no student failed in any course.** |
|  | mysql> SELECT D.Dept\_name  FROM Department1 D  WHERE D.Department\_id NOT IN (  SELECT S.Department\_id  FROM Student1 S  JOIN Examination1 E ON S.Student\_id = E.Student\_id  WHERE E.Marks < 40  );  +-----------+  | Dept\_name |  +-----------+  | Computer |  | IT |  | ENTC |  | ECE |  | AIDS |  +-----------+  5 rows in set (0.00 sec) |
|  | **Q11. List faculty with total salary paid over time.** |
|  | mysql> SELECT F.Fname, F.Lname, SUM(EA.Salary) AS Total\_Salary  FROM Faculty1 F  JOIN Employee\_account1 EA ON F.E\_no = EA.E\_no  GROUP BY F.E\_no, F.Fname, F.Lname;  +-------+---------+--------------+  | Fname | Lname | Total\_Salary |  +-------+---------+--------------+  | Arun | Kulkarni| 36500 |  | Meena | Desai | 39500 |  | Suresh| Naik | 33500 |  | Pooja | Verma | 43500 |  | Anil | Bhosale | 43000 |  +-------+---------+--------------+  5 rows in set (0.00 sec) |
|  | **Q12. Display the course in which the maximum number of students appeared.** |
|  | mysql> SELECT C.Course\_name, COUNT(E.Student\_id) AS Student\_Count  FROM Course1 C  JOIN Examination1 E ON C.Course\_id = E.Course\_id  GROUP BY C.Course\_id, C.Course\_name  ORDER BY Student\_Count DESC  LIMIT 1;  +-------------+---------------+  | Course\_name | Student\_Count |  +-------------+---------------+  | DBMS | 1 |  +-------------+---------------+  1 row in set (0.00 sec) |
|  | **Q13. Get the youngest student in each department.** |
|  | mysql> SELECT D.Dept\_name, S.Fname, S.Lname, S.DOB  FROM Student1 S  JOIN Department1 D ON S.Department\_id = D.Department\_id  WHERE S.DOB = (  SELECT MAX(S2.DOB)  FROM Student1 S2  WHERE S2.Department\_id = S.Department\_id  );  +-----------+-------+----------+------------+  | Dept\_name | Fname | Lname | DOB |  +-----------+-------+----------+------------+  | Computer | Rahul | Patil | 2002-03-11 |  | IT | Sneha | Sharma | 2001-07-19 |  | ENTC | Amit | Deshmukh | 2000-12-05 |  | ECE | Priya | Kadam | 2002-01-25 |  | AIDS | Rohit | Joshi | 2001-05-30 |  +-----------+-------+----------+------------+  5 rows in set (0.00 sec) |

**Level 4:**

|  |  |
| --- | --- |
|  | **1. Get each student's full name, department name, and total marks** |
|  | mysql> select concat(s.Fname,' ',s.Lname) as Student, d.Dept\_name, e.Marks  -> from Student1 s  -> join Department1 d on s.Department\_id=d.Department\_id  -> join Examination1 e on s.Student\_id=e.Student\_id;  +-----------+-----------+-------+  | Student | Dept\_name | Marks |  +-----------+-----------+-------+  | Rahul Patil | Computer | 85 |  | Sneha Sharma | IT | 90 |  | Amit Deshmukh| ENTC | 78 |  | Priya Kadam | ECE | 88 |  | Rohit Joshi | AIDS | 92 |  +-----------+-----------+-------+  5 rows in set (0.00 sec) |
|  | **2. Show net salary for every employee**  **(Net salary = Basic + HRA + Allowance − PF − Tax)** |
|  | mysql> select e.E\_no, f.Fname, f.Lname,  -> (ea.Basic+ea.HRA+ea.Allowance - ea.PF - ea.Income\_tax) as Net\_Salary  -> from Faculty1 f  -> join Employee\_account1 ea on f.E\_no=ea.E\_no  -> join Faculty1 e on e.E\_no=ea.E\_no;  +-----+--------+----------+------------+  | E\_no| Fname | Lname | Net\_Salary |  +-----+--------+----------+------------+  | 1| Arun | Kulkarni | 36500 |  | 2| Meena | Desai | 39500 |  | 3| Suresh | Naik | 33500 |  | 4| Pooja | Verma | 43500 |  | 5| Anil | Bhosale | 43000 |  +-----+--------+----------+------------+  5 rows in set (0.00 sec) |
|  | **3. Students with number of courses and average score** |
|  | mysql> select s.Student\_id, concat(s.Fname,' ',s.Lname) as Student,  -> count(e.Course\_id) as Courses\_Taken, avg(e.Marks) as Avg\_Marks  -> from Student1 s  -> left join Examination1 e on s.Student\_id=e.Student\_id  -> group by s.Student\_id;  +------------+---------------+---------------+-----------+  | Student\_id | Student | Courses\_Taken | Avg\_Marks |  +------------+---------------+---------------+-----------+  | 1 | Rahul Patil | 1 | 85.00 |  | 2 | Sneha Sharma | 1 | 90.00 |  | 3 | Amit Deshmukh | 1 | 78.00 |  | 4 | Priya Kadam | 1 | 88.00 |  | 5 | Rohit Joshi | 1 | 92.00 |  +------------+---------------+---------------+-----------+  5 rows in set (0.00 sec) |
|  | **4. Student who paid the highest total fee** |
|  | mysql> select s.Student\_id, concat(s.Fname,' ',s.Lname) as Student, sa.Total\_amount  -> from Student1 s  -> join Student\_account1 sa on s.Student\_id=sa.Student\_id  -> order by sa.Total\_amount desc limit 1;  +------------+-------------+--------------+  | Student\_id | Student | Total\_amount |  +------------+-------------+--------------+  | 4 | Priya Kadam | 44200 |  +------------+-------------+--------------+  1 row in set (0.00 sec) |
|  | **5. Departments with ≥1 faculty and ≥10 students** |
|  | mysql> select d.Dept\_name  -> from Department1 d  -> join Student1 s on d.Department\_id=s.Department\_id  -> join Faculty1 f on d.Department\_id=f.Department\_id  -> group by d.Dept\_name  -> having count(distinct s.Student\_id)>=10;  Empty set (0.00 sec) |
|  | **6. Fees collected and students per year** |
|  | mysql> select Year, sum(Total\_amount) as Total\_Fees, count(Student\_id) as Students  -> from Student\_account1  -> group by Year;  +------+------------+----------+  | Year | Total\_Fees | Students |  +------+------------+----------+  | 2023 | 205500 | 5 |  +------+------------+----------+  1 row in set (0.00 sec) |
|  | **7. Students not in any exam** |
|  | mysql> select concat(Fname,' ',Lname) as Student  -> from Student1  -> where Student\_id not in (select Student\_id from Examination1);  Empty set (0.00 sec) |
|  | **8. Students with fee records for 2023, 2024, 2025** |
|  | mysql> select Student\_id  -> from Student\_account1  -> group by Student\_id  -> having count(distinct Year)=3;  Empty set (0.00 sec) |
|  | **9. Faculty whose course has the lowest average marks** |
|  | mysql> select f.Fname,f.Lname,avg(e.Marks) as Avg\_Marks  -> from Faculty1 f  -> join Examination1 e on f.Course\_id=e.Course\_id  -> group by f.E\_no  -> order by Avg\_Marks asc limit 1;  +--------+------+-----------+  | Fname |Lname | Avg\_Marks |  +--------+------+-----------+  | Suresh | Naik | 78.00 |  +--------+------+-----------+  1 row in set (0.00 sec) |
|  | **10. Departments where every student paid > ₹50,000** |
|  | mysql> select d.Dept\_name  -> from Department1 d  -> join Student1 s on d.Department\_id=s.Department\_id  -> join Student\_account1 sa on s.Student\_id=sa.Student\_id  -> group by d.Department\_id  -> having min(sa.Total\_amount)>50000;  Empty set (0.00 sec) |
|  | **11. Each course with highest, lowest, avg marks** |
|  | mysql> select c.Course\_name, max(e.Marks) as Max\_Marks, min(e.Marks) as Min\_Marks, avg(e.Marks) as Avg\_Marks  -> from Course1 c  -> join Examination1 e on c.Course\_id=e.Course\_id  -> group by c.Course\_name;  +-------------+-----------+-----------+-----------+  | Course\_name | Max\_Marks | Min\_Marks | Avg\_Marks |  +-------------+-----------+-----------+-----------+  | DBMS | 85 | 85 | 85.00 |  | OOP | 90 | 90 | 90.00 |  | Signals | 78 | 78 | 78.00 |  | Networks | 88 | 88 | 88.00 |  | AI | 92 | 92 | 92.00 |  +-------------+-----------+-----------+-----------+  5 rows in set (0.00 sec) |
|  | **12. Students with number of failed courses (<40)** |
|  | mysql> select s.Student\_id, concat(s.Fname,' ',s.Lname) as Student,  -> sum(case when e.Marks<40 then 1 else 0 end) as Failed\_Courses  -> from Student1 s  -> left join Examination1 e on s.Student\_id=e.Student\_id  -> group by s.Student\_id;  +------------+---------------+----------------+  | Student\_id | Student | Failed\_Courses |  +------------+---------------+----------------+  | 1 | Rahul Patil | 0 |  | 2 | Sneha Sharma | 0 |  | 3 | Amit Deshmukh | 0 |  | 4 | Priya Kadam | 0 |  | 5 | Rohit Joshi | 0 |  +------------+---------------+----------------+  5 rows in set (0.00 sec) |
|  | **13. Students from Pune scoring >80** |
|  | mysql> select s.Fname,s.Lname,e.Marks  -> from Student1 s  -> join Examination1 e on s.Student\_id=e.Student\_id  -> where s.City='Pune' and e.Marks>80;  +-------+-------+-------+  | Fname | Lname | Marks |  +-------+-------+-------+  | Rahul | Patil | 85 |  +-------+-------+-------+  1 row in set (0.00 sec) |
|  | **14. Department-wise fee collection per year** |
|  | mysql> select d.Dept\_name, sa.Year, sum(sa.Total\_amount) as Total\_Fees  -> from Department1 d  -> join Student1 s on d.Department\_id=s.Department\_id  -> join Student\_account1 sa on s.Student\_id=sa.Student\_id  -> group by d.Dept\_name, sa.Year;  +-----------+------+------------+  | Dept\_name | Year | Total\_Fees |  +-----------+------+------------+  | Computer | 2023 | 42000 |  | IT | 2023 | 39500 |  | ENTC | 2023 | 36800 |  | ECE | 2023 | 44200 |  | AIDS | 2023 | 43000 |  +-----------+------+------------+  5 rows in set (0.00 sec) |
|  | **15. Students sharing city with at least one faculty** |
|  | mysql> select distinct concat(s.Fname,' ',s.Lname) as Student, s.City  -> from Student1 s  -> where s.City in (select distinct 'Pune'); -- ⚠ sample, since faculty cities not stored  Empty set (0.00 sec) |