Module 5 – Introduction To Dbms

Q1: Create Table Name: Student and Exam

```
CREATE TABLE `students` (
  `Rollno` int(11) PRIMARY key NOT NULL AUTO_INCREMENT,
  `Name` varchar(25) DEFAULT NULL,
  `Branch` varchar(30) DEFAULT NULL
)
```

```
CREATE TABLE marks ( rollno INT, s_code VARCHAR(10) NOT NULL, marks INT NOT NULL, p_code VARCHAR(10) NOT NULL, FOREIGN KEY (rollno) REFERENCES students(rollno) );
```

				Rollno	Name	Branch
☐	🖉 Edit	⧉ Copy	⊖ Delete	1	jay	Comuper science
☐	🖉 Edit	⧉ Copy	⊖ Delete	2	suhani	Electronic & com
☐	🖉 Edit	⧉ Copy	⊖ Delete	3	kriti	Electronic & com

```
1  INSERT INTO marks (rollno, s_code, marks, p_code) VALUES
2  (1, 'CS11', 85, 'P1'),
3  (1, 'CS12', 90, 'P2'),
4  (2, 'EC101', 78, 'P1'),
5  (2, 'EC102', 82, 'P2'),
6  (3, 'EC101', 88, 'P1'),
7  (3, 'EC102', 79, 'P2');
```

```
1 SELECT s.rollno,s.name,m.p_code,m.marks,m.s_code
2 FROM students s
3 JOIN  marks m
4 ON  s.rollno = m.rollno;
```

rollno	name	p_code	marks	s_code
1	jay	P1	85	CS11
1	jay	P2	90	CS12
2	suhani	P1	78	EC101
2	suhani	P2	82	EC102
3	kriti	P1	88	EC101
3	kriti	P2	79	EC102

Q2: Create table given below: Employee and IncentiveTable

```
1 INSERT INTO Employees (F_name, L_name, salary, joining_date, department) VALUES
2 ('John', 'abraham', 10000, '1-jan-13 12.00.00AM', 'banking'),
3 ('michel', 'clark', 75000, '1-jan-13 12.00.00AM', 'insurance'),
4 ('roy', 'thomas', 80000, '1-feb-13 12.00.00AM', 'banking'),
5 ('tom', 'josh', 67000, '1-feb-13 12.00.00AM', 'insurance'),
6 ('jerry', 'pinto', 15000, '1-feb-13 12.00.00AM', 'insurance'),
7 ('philip', 'mathew', 25000, '1-jan-13 12.00.00AM', 'service'),
8 ('testname1', '123', 85000, '1-jan-13 12.00.00AM', 'service'),
9 ('testname2', 'Lname%', 57000, '1-feb-13 12.00.00AM', 'insurance');
0 |
```

```
1  INSERT INTO Incentives (employee_ref_id, incentive_date, incentive_amount) VALUES
2  (1, '2013-02-01', 5000.00),
3  (2, '2013-02-01', 3000.00),
4  (3, '2013-02-01', 4000.00),
5  (1, '2013-01-01', 4500.00),
6  (2, '2013-01-01', 3500.00);
```

Q3: Get First_Name from employee table using Tom name "Employee Name".

```
1  SELECT F_name
2  FROM Employees
3  WHERE F_name = 'tom';
4
```

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

```sql
SELECT F_name FROM Employees WHERE F_name = 'tom';
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

☐ Show all | Number of rows: 25 ⌄   Filter rows: Search this tab

Extra options

←T→   **F_name**

☐ ✎ Edit ⊰ᵢ Copy ⊖ Delete tom

Q4: Get FIRST_NAME, Joining Date, and Salary from employee table

```sql
1  SELECT F_name AS First_Name,
2        joining_date AS Joining_Date,
3        salary AS Salary
4  FROM Employees;
5  |
```

✔ Showing rows 0 - 7 (8 total, Query took 0.0002 seconds.)

```sql
SELECT F_name AS First_Name, joining_date AS Joining_Date, salary AS Salary FROM Employees;
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

☐ Show all | Number of rows: 25 ⌄ Filter rows: Search this table | Sort by key: None

Extra options

←T→			First_Name	Joining_Date	Salary
☐ 🖉 Edit ⌗ Copy ⊖ Delete			John	2013-01-01 12:00:00	10000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			michel	2013-01-01 12:00:00	75000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			roy	2013-02-01 12:00:00	80000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			tom	2013-02-01 12:00:00	67000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			jerry	2013-02-01 12:00:00	15000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			philip	2013-01-01 12:00:00	25000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			testname1	2013-01-01 12:00:00	85000
☐ 🖉 Edit ⌗ Copy ⊖ Delete			testname2	2013-02-01 12:00:00	57000

Q5: Get all employee details from the employee table order by First_Name Ascending and Salary descending?

```
1  SELECT *
2  FROM Employees
3  ORDER BY F_name ASC, salary DESC;
4
```

	employee_id	F_name ▲ 1	L_name	salary ▼ 2	joining_date	department
☐ 🖉 Edit ᠄ Copy ⊖ Delete	5	jerry	pinto	15000	2013-02-01 12:00:00	insurance
☐ 🖉 Edit ᠄ Copy ⊖ Delete	1	John	abraham	10000	2013-01-01 12:00:00	banking
☐ 🖉 Edit ᠄ Copy ⊖ Delete	2	michel	clark	75000	2013-01-01 12:00:00	insurance
☐ 🖉 Edit ᠄ Copy ⊖ Delete	6	philip	mathew	25000	2013-01-01 12:00:00	service
☐ 🖉 Edit ᠄ Copy ⊖ Delete	3	roy	thomas	80000	2013-02-01 12:00:00	banking
☐ 🖉 Edit ᠄ Copy ⊖ Delete	7	testname1	123	85000	2013-01-01 12:00:00	service
☐ 🖉 Edit ᠄ Copy ⊖ Delete	8	testname2	Lname%	57000	2013-02-01 12:00:00	insurance
☐ 🖉 Edit ᠄ Copy ⊖ Delete	4	tom	josh	67000	2013-02-01 12:00:00	insurance

Q6: Get employee details from employee table whose first name contains 'J'.

```
1 SELECT *
2 FROM Employees
3 WHERE F_name LIKE '%J%';
```

←T→			employee_id	F_name	L_name	salary	joining_date	department
☐	Edit	Copy	Delete	1	John	abraham	10000	2013-01-01 12:00:00	banking
☐	Edit	Copy	Delete	5	jerry	pinto	15000	2013-02-01 12:00:00	insurance

---

Q7,8: Get department wise maximum salary from employee table order by 8. salary ascending?

```
1 SELECT department, MAX(salary) AS max_salary
2 FROM Employees
3 GROUP BY department
4 ORDER BY max_salary ASC;
5
```

←T→			department	max_salary ▲ 1
☐	Edit	Copy	Delete	insurance	75000
☐	Edit	Copy	Delete	banking	80000
☐	Edit	Copy	Delete	service	85000

Q9: Select First_Name, incentive amount from employee and incentives table for those employees who have incentives and incentive amount greater than 3000

```
1  SELECT e.F_name AS first_name, i.incentive_amount
2  FROM Employees e
3  JOIN Incentives i ON e.employee_id = i.employee_ref_id
4  WHERE i.incentive_amount > 3000;
```

first_name	incentive_amount
John	5000
roy	4000
John	4500
michel	3500

Q10: Create After Insert trigger on Employee table which insert records in viewtable

```sql
1  CREATE TABLE viewtable (
2      employee_id INT,
3      F_name VARCHAR(50),
4      L_name VARCHAR(50),
5      salary DECIMAL(10, 2),
6      joining_date DATE,
7      department VARCHAR(50)
8  );
```

```sql
DELIMITER //

CREATE TRIGGER after_employee_insertes
AFTER INSERT ON Employees
FOR EACH ROW
BEGIN
    INSERT INTO viewtable (employee_id, F_name, L_name, salary, joining_date, department)
    VALUES (NEW.employee_id, NEW.F_name, NEW.L_name, NEW.salary, NEW.joining_date, NEW.department);
END;
//

DELIMITER ;
```

```sql
1  INSERT INTO Employees (F_name, L_name, salary, joining_date, department)
2  VALUES ('John', 'Doe', 60000, '2024-01-15', 'Engineering');
3
```

employee_id	F_name	L_name	salary	joining_date	department
12	John	Doe	60000.00	2024-01-15	Engineering
12	John	Doe	60000.00	2024-01-15	Engineering

Q11,12: Create table given below: Salesperson and Customer

```
1 CREATE TABLE Salesperson (
2     SNo INT PRIMARY KEY,
3     SName VARCHAR(50),
4     City VARCHAR(50),
5     Comm DECIMAL(4, 2)
6 );
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐ 1	SNo 🔑	int(11)			No	None			✏ Change	⊖ Drop	More
☐ 2	SName	varchar(50)	utf8mb4_general_ci		Yes	NULL			✏ Change	⊖ Drop	More
☐ 3	City	varchar(50)	utf8mb4_general_ci		Yes	NULL			✏ Change	⊖ Drop	More
☐ 4	Comm	decimal(4,2)			Yes	NULL			✏ Change	⊖ Drop	More

```
1 INSERT INTO Salesperson (SNo, SName, City, Comm) VALUES
2 (1001, 'Peel', 'London', 0.12),
3 (1002, 'Serres', 'San Jose', 0.13),
4 (1004, 'Motika', 'London', 0.11),
5 (1007, 'Rafkin', 'Barcelona', 0.15),
6 (1003, 'Axelrod', 'New York', 0.10);
```

			SNo	SName	City	Comm
☐	✏ Edit	ᴴ Copy ⊖ Delete	1001	Peel	London	0.12
☐	✏ Edit	ᴴ Copy ⊖ Delete	1002	Serres	San Jose	0.13
☐	✏ Edit	ᴴ Copy ⊖ Delete	1003	Axelrod	New York	0.10
☐	✏ Edit	ᴴ Copy ⊖ Delete	1004	Motika	London	0.11
☐	✏ Edit	ᴴ Copy ⊖ Delete	1007	Rafkin	Barcelona	0.15

```
CREATE TABLE Customer (
    CNo INT PRIMARY KEY,
    CName VARCHAR(50),
    City VARCHAR(50),
    Rating INT,
    SNo INT,
    FOREIGN KEY (SNo) REFERENCES Salesperson(SNo)
);
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐ 1	**SNo** 🔑	int(11)			No	*None*			🖊 Change	⊖ Drop	More
☐ 2	**SName**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			🖊 Change	⊖ Drop	More
☐ 3	**City**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			🖊 Change	⊖ Drop	More
☐ 4	**Comm**	decimal(4,2)			Yes	*NULL*			🖊 Change	⊖ Drop	More

```
INSERT INTO Customer (CNo, CName, City, Rating, SNo) VALUES
(201, 'Hoffman', 'London', 100, 1001),
(202, 'Giovanne', 'Roe', 200, 1003),
(203, 'Liu', 'San Jose', 300, 1002),
(204, 'Grass', 'Barcelona', 100, 1007),
(206, 'Clemens', 'London', 100, 1007),
(207, 'Pereira', 'Roe', 100, 1004);
```

			CNo	CName	City	Rating	SNo
☐	Edit	Copy	Delete	201	Hoffman	London	100	1001
☐	Edit	Copy	Delete	202	Giovanne	Roe	200	1003
☐	Edit	Copy	Delete	203	Liu	San Jose	300	1002
☐	Edit	Copy	Delete	204	Grass	Barcelona	100	1007
☐	Edit	Copy	Delete	206	Clemens	London	100	1007
☐	Edit	Copy	Delete	207	Pereira	Roe	100	1004

Q13: All Customer name whose rating is more than 100.

```
1 SELECT CName
2 FROM Customer
3 WHERE Rating > 100;
4
```

			CName
☐	Edit	Copy	Delete	Giovanne
☐	Edit	Copy	Delete	Liu

Q14: Names and cities of all salespeople in London with commission above 0.12

```
1  SELECT SName, City
2  FROM Salesperson
3  WHERE City = 'London' AND Comm > 0.12;
4  |
```

✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)

SELECT SName, City FROM Salesperson WHERE City = 'London' AND Comm > 0.12;

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

SName	City

Query results operations

Q15: All salespeople either in Barcelona or in London

```
1  SELECT SName, City
2  FROM Salesperson
3  WHERE City = 'Barcelona' OR City = 'London';
4
```

←T→			▼ SName	City
☐ 🖉 Edit 📋 Copy ⊖ Delete			Peel	London
☐ 🖉 Edit 📋 Copy ⊖ Delete			Motika	London
☐ 🖉 Edit 📋 Copy ⊖ Delete			Rafkin	Barcelona

Q16: All salespeople with commission between 0.10 and 0.12. (Boundary values should be excluded).

```
1  SELECT SName, City, Comm
2  FROM Salesperson
3  WHERE Comm > 0.10 AND Comm < 0.12;
4
```

				SName	City	Comm
☐	✏ Edit	Copy	⊖ Delete	Motika	London	0.11

Q17: All customers excluding those with rating <= 100 unless they are located in Rome

```
SELECT CName, City, Rating
FROM Customer
WHERE Rating > 100 OR City = 'Rome';
```

				CName	City	Rating
☐	✏ Edit	Copy	⊖ Delete	Giovanne	Roe	200
☐	✏ Edit	Copy	⊖ Delete	Liu	San Jose	300

Q18: Write a SQL statement that displays all the information about all salespeople

```
1 SELECT *
2 FROM Salesperson;
3
```

				SNo	SName	City	Comm
☐	Edit	Copy	⊖ Delete	1001	Peel	London	0.12
☐	Edit	Copy	⊖ Delete	1002	Serres	San Jose	0.13
☐	Edit	Copy	⊖ Delete	1003	Axelrod	New York	0.10
☐	Edit	Copy	⊖ Delete	1004	Motika	London	0.11
☐	Edit	Copy	⊖ Delete	1007	Rafkin	Barcelona	0.15

Q19: From the following table, write a SQL query to find orders that are delivered by a salesperson with ID. 5001. Return ord_no, ord_date, purch_amt. Sample table: orders

```
1 CREATE TABLE Salesmen (
2     salesman_id INT PRIMARY KEY,
3     name VARCHAR(50),
4     city VARCHAR(50),
5     commission DECIMAL(4,2)
6 );
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐ 1	**salesman_id** 🔑	int(11)			No	*None*			🖉 Change	⊖ Drop	More
☐ 2	**name**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			🖉 Change	⊖ Drop	More
☐ 3	**city**	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			🖉 Change	⊖ Drop	More
☐ 4	**commission**	decimal(4,2)			Yes	*NULL*			🖉 Change	⊖ Drop	More

```
8  INSERT INTO Salesmen (salesman_id, name, city, commission) VALUES
9  (5001, 'James Hoog', 'New York', 0.15),
10 (5002, 'Nail Knite', 'Paris', 0.13),
11 (5005, 'Pit Alex', 'London', 0.11),
12 (5006, 'Mc Lyon', 'Paris', 0.14),
13 (5007, 'Paul Adam', 'Rome', 0.13),
14 (5003, 'Lauson Hen', 'San Jose', 0.12);
15
```

←T→				salesman_id	name	city	commission
☐	🖉 Edit	⫶ Copy	⊖ Delete	5001	James Hoog	New York	0.15
☐	🖉 Edit	⫶ Copy	⊖ Delete	5002	Nail Knite	Paris	0.13
☐	🖉 Edit	⫶ Copy	⊖ Delete	5003	Lauson Hen	San Jose	0.12
☐	🖉 Edit	⫶ Copy	⊖ Delete	5005	Pit Alex	London	0.11
☐	🖉 Edit	⫶ Copy	⊖ Delete	5006	Mc Lyon	Paris	0.14
☐	🖉 Edit	⫶ Copy	⊖ Delete	5007	Paul Adam	Rome	0.13

```
1 CREATE TABLE Orders (
2     ord_no INT PRIMARY KEY,
3     purch_amt DECIMAL(10,2),
4     ord_date DATE,
5     customer_id INT,
6     salesman_id INT,
7     FOREIGN KEY (salesman_id) REFERENCES Salesmen(salesman_id)
8 );
```

```sql
1  INSERT INTO Orders (ord_no, purch_amt, ord_date, customer_id, salesman_id) VALUES
2  (70001, 150.50, '2012-10-05', 3005, 5002),
3  (70009, 270.65, '2012-09-10', 3001, 5005),
4  (70002, 65.26, '2012-10-05', 3002, 5001),
5  (70004, 110.50, '2012-08-17', 3009, 5003),
6  (70007, 948.50, '2012-09-10', 3005, 5002),
7  (70005, 2400.60, '2012-07-27', 3007, 5001),
8  (70008, 5760.00, '2012-09-10', 3002, 5001),
9  (70010, 1983.43, '2012-10-10', 3004, 5006),
10 (70003, 2480.40, '2012-10-10', 3009, 5003),
11 (70012, 250.45, '2012-06-27', 3008, 5002),
12 (70011, 75.29, '2012-08-17', 3003, 5007),
13 (70013, 3045.60, '2012-04-25', 3002, 5001);
```

←T→			ord_no	purch_amt	ord_date	customer_id	salesman_id
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70001	150.50	2012-10-05	3005	5002
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70002	65.26	2012-10-05	3002	5001
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70003	2480.40	2012-10-10	3009	5003
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70004	110.50	2012-08-17	3009	5003
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70005	2400.60	2012-07-27	3007	5001
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70007	948.50	2012-09-10	3005	5002
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70008	5760.00	2012-09-10	3002	5001
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70009	270.65	2012-09-10	3001	5005
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70010	1983.43	2012-10-10	3004	5006
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70011	75.29	2012-08-17	3003	5007
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70012	250.45	2012-06-27	3008	5002
☐	✏ Edit	⟩⟨ Copy ⊖ Delete	70013	3045.60	2012-04-25	3002	5001

```
1 SELECT ord_no, ord_date, purch_amt
2 FROM Orders
3 WHERE salesman_id = 5001;
4 |
```

←T→				ord_no	ord_date	purch_amt
☐	🖉 Edit	ϟ Copy	⊖ Delete	70002	2012-10-05	65.26
☐	🖉 Edit	ϟ Copy	⊖ Delete	70005	2012-07-27	2400.60
☐	🖉 Edit	ϟ Copy	⊖ Delete	70008	2012-09-10	5760.00
☐	🖉 Edit	ϟ Copy	⊖ Delete	70013	2012-04-25	3045.60

Q20: From the following table, write a SQL query to select a range of products whose price is in the range Rs.200 to Rs.600. Begin and end values are included. Return pro_id, pro_name, pro_price, and pro_com.

```
1 CREATE TABLE item_mast (
2     PRO_ID INT PRIMARY KEY,
3     PRO_NAME VARCHAR(50),
4     PRO_PRICE DECIMAL(10, 2),
5     PRO_COM INT
6 );
```

```
1  INSERT INTO item_mast (PRO_ID, PRO_NAME, PRO_PRICE, PRO_COM) VALUES
2  (101, 'Mother Board', 3200.00, 15),
3  (102, 'Key Board', 450.00, 16),
4  (103, 'ZIP drive', 250.00, 14),
5  (104, 'Speaker', 550.00, 16),
6  (105, 'Monitor', 5000.00, 11),
7  (106, 'DVD drive', 900.00, 12),
8  (107, 'CD drive', 800.00, 12),
9  (108, 'Printer', 2600.00, 13),
10 (109, 'Refill cartridge', 350.00, 13),
11 (110, 'Mouse', 250.00, 12);
12
```

				PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
☐	Edit	Copy	Delete	101	Mother Board	3200.00	15
☐	Edit	Copy	Delete	102	Key Board	450.00	16
☐	Edit	Copy	Delete	103	ZIP drive	250.00	14
☐	Edit	Copy	Delete	104	Speaker	550.00	16
☐	Edit	Copy	Delete	105	Monitor	5000.00	11
☐	Edit	Copy	Delete	106	DVD drive	900.00	12
☐	Edit	Copy	Delete	107	CD drive	800.00	12
☐	Edit	Copy	Delete	108	Printer	2600.00	13
☐	Edit	Copy	Delete	109	Refill cartridge	350.00	13
☐	Edit	Copy	Delete	110	Mouse	250.00	12

```
1  SELECT PRO_ID, PRO_NAME, PRO_PRICE, PRO_COM
2  FROM item_mast
3  WHERE PRO_PRICE BETWEEN 200 AND 600;
```

←T→			▼ PRO_ID	PRO_NAME	PRO_PRICE	PRO_COM
☐	Edit Copy	Delete	102	Key Board	450.00	16
☐	Edit Copy	Delete	103	ZIP drive	250.00	14
☐	Edit Copy	Delete	104	Speaker	550.00	16
☐	Edit Copy	Delete	109	Refill cartridge	350.00	13
☐	Edit Copy	Delete	110	Mouse	250.00	12

Q21: From the following table, write a SQL query to calculate the average price for a manufacturer code of 16. Return avg.

```
1 SELECT AVG(PRO_PRICE) AS avg_price
2 FROM item_mast
3 WHERE PRO_COM = 16;
4
```

avg_price

500.000000

Q22: From the following table, write a SQL query to display the pro_name as 'Item Name' and pro_priceas 'Price in Rs.'

```
1 SELECT PRO_NAME AS 'Item Name', PRO_PRICE AS 'Price in Rs.'
2 FROM item_mast;
```

←T→				Item Name	Price in Rs.
☐	Edit	Copy	⊖ Delete	Mother Board	3200.00
☐	Edit	Copy	⊖ Delete	Key Board	450.00
☐	Edit	Copy	⊖ Delete	ZIP drive	250.00
☐	Edit	Copy	⊖ Delete	Speaker	550.00
☐	Edit	Copy	⊖ Delete	Monitor	5000.00
☐	Edit	Copy	⊖ Delete	DVD drive	900.00
☐	Edit	Copy	⊖ Delete	CD drive	800.00
☐	Edit	Copy	⊖ Delete	Printer	2600.00
☐	Edit	Copy	⊖ Delete	Refill cartridge	350.00
☐	Edit	Copy	⊖ Delete	Mouse	250.00

Q23: From the following table, write a SQL query to find the items whose prices are higher than or equal to $250. Order the result by product price in descending, then product name in ascending. Return pro_name and pro_price.

```
1 SELECT PRO_NAME, PRO_PRICE
2 FROM item_mast
3 WHERE PRO_PRICE >= 250
4 ORDER BY PRO_PRICE DESC, PRO_NAME ASC;
  |
```

←T→				PRO_NAME ▲ 2	PRO_PRICE ▼ 1
☐	✎ Edit	⚡ Copy	⊖ Delete	Monitor	5000.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Mother Board	3200.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Printer	2600.00
☐	✎ Edit	⚡ Copy	⊖ Delete	DVD drive	900.00
☐	✎ Edit	⚡ Copy	⊖ Delete	CD drive	800.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Speaker	550.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Key Board	450.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Refill cartridge	350.00
☐	✎ Edit	⚡ Copy	⊖ Delete	Mouse	250.00
☐	✎ Edit	⚡ Copy	⊖ Delete	ZIP drive	250.00

Q24: From the following table, write a SQL query to calculate average price of the items for each company. Return average price and company code.

```
1 SELECT PRO_COM AS company_code, AVG(PRO_PRICE) AS average_price
2 FROM item_mast
3 GROUP BY PRO_COM;
```

←T→				company_code	average_price
☐	✎ Edit	⚡ Copy	⊖ Delete	11	5000.000000
☐	✎ Edit	⚡ Copy	⊖ Delete	12	650.000000
☐	✎ Edit	⚡ Copy	⊖ Delete	13	1475.000000
☐	✎ Edit	⚡ Copy	⊖ Delete	14	250.000000
☐	✎ Edit	⚡ Copy	⊖ Delete	15	3200.000000
☐	✎ Edit	⚡ Copy	⊖ Delete	16	500.000000