Q5: Student Information Table with DML, DDL, and TCL

```
CREATE TABLE Student (
   RollNo INT PRIMARY KEY,
   Name VARCHAR(100),
   Course VARCHAR(50),
   Marks INT
);

-- DML Commands
INSERT INTO Student VALUES (1, 'Alice', 'DBMS', 85);
UPDATE Student SET Marks = 90 WHERE RollNo = 1;
DELETE FROM Student WHERE Marks < 40;

-- TCL Commands
SAVEPOINT sp1;
ROLLBACK TO sp1;
COMMIT;
```

Q6: Views in Employee and Department

```
CREATE VIEW EmpDeptSummary AS
SELECT e.EmpID, e.Name, d.DeptName
FROM Employee e
JOIN Department d ON e.DeptID = d.DeptID;

-- Output of View
SELECT * FROM EmpDeptSummary;

-- | EmpID | Name  | DeptName |
-- | 101   | Alice | IT       |
-- | 102   | Bob   | IT       |
-- | 103   | Carol | HR       |
```

Q7: PLSQL Program with Function

```
CREATE OR REPLACE FUNCTION AddTwoNumbers(x IN NUMBER, y IN NUMBER)
RETURN NUMBER IS
BEGIN
   RETURN x + y;
END;

-- Output
-- SELECT AddTwoNumbers(2, 3) FROM dual;
-- Result: 5

-- Example for Sqrt
-- SELECT SQRT(2) FROM dual;
-- Result: 1.41
```

Q8: Database with Constraints and Trigger

```
CREATE TABLE Accounts (
   AccID INT PRIMARY KEY,
   Name VARCHAR(100),
   Balance INT CHECK (Balance >= 0)
);

CREATE OR REPLACE TRIGGER BalanceCheck
BEFORE INSERT OR UPDATE ON Accounts
FOR EACH ROW
BEGIN
   IF :NEW.Balance < 0 THEN
      RAISE_APPLICATION_ERROR(-20001, 'Negative balance not allowed');
   END IF;
END;

-- Test Trigger
-- INSERT INTO Accounts VALUES (1, 'John', -100);
-- Output: Error - Negative balance not allowed
```

Q9 - Q12: Follow Ups

Q9: Same as Q4 - Reuse queries from Employee and Department.
Q10: Follow Q4 - Already included above.
Q11: Follow Q6 with Index

```
CREATE INDEX idx_name ON Employee(Name);
```

Q12: Follow Q3 - Arithmetic and exception handling.

Q13: Student Registration ERD (2NF)

```
CREATE TABLE Course (
   CourseID INT PRIMARY KEY,
   CourseName VARCHAR(100)
);

CREATE TABLE Registration (
   RegID INT PRIMARY KEY,
   StudentName VARCHAR(100),
   CourseID INT,
   FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

-- In 2NF, partial dependencies are removed.
-- Sample Query:
INSERT INTO Course VALUES (1, 'DBMS');
INSERT INTO Registration VALUES (1001, 'Alice', 1);
SELECT * FROM Registration;

-- Output:
```

```
-- | RegID | StudentName | CourseID |
-- | 1001  | Alice       | 1        |
```