SQL Training Examples with Trainings Table

# Step 1: Create Trainings Table and Insert Records

We begin by creating a sample table named 'Trainings' that will store details of training programs such as title, trainer, dates, fees, and location.   
This table will be used to demonstrate SQL functions, stored procedures, user-defined functions, subqueries, views, and triggers.

CREATE TABLE Trainings (  
 TrainingID INT PRIMARY KEY AUTO\_INCREMENT,  
 Title VARCHAR(100),  
 Trainer VARCHAR(50),  
 StartDate DATE,  
 EndDate DATE,  
 DurationDays INT,  
 Location VARCHAR(50),  
 Fees DECIMAL(10,2),  
 IsOnline BOOLEAN  
);  
  
-- Insert 10 sample records  
INSERT INTO Trainings (Title, Trainer, StartDate, EndDate, DurationDays, Location, Fees, IsOnline) VALUES  
('SQL Basics', 'Aarav Mehta', '2025-01-05', '2025-01-07', 3, 'Delhi', 5000, 0),  
('Advanced SQL', 'Priya Sharma', '2025-02-01', '2025-02-05', 5, 'Mumbai', 8000, 1),  
('Data Analysis', 'Rohan Gupta', '2025-03-10', '2025-03-12', 3, 'Bangalore', 6000, 1),  
('Python for Data', 'Sanya Kapoor', '2025-03-15', '2025-03-20', 6, 'Hyderabad', 7000, 0),  
('Machine Learning', 'Arjun Nair', '2025-04-01', '2025-04-10', 10, 'Chennai', 12000, 1),  
('Power BI', 'Nisha Jain', '2025-04-15', '2025-04-18', 4, 'Delhi', 6500, 0),  
('Excel Advanced', 'Vikram Rao', '2025-05-01', '2025-05-03', 3, 'Pune', 4000, 0),  
('Cloud Basics', 'Anjali Verma', '2025-05-10', '2025-05-12', 3, 'Mumbai', 5500, 1),  
('AI Concepts', 'Kunal Singh', '2025-06-01', '2025-06-07', 7, 'Bangalore', 15000, 1),  
('Big Data', 'Neha Patel', '2025-06-15', '2025-06-20', 6, 'Hyderabad', 10000, 0);

# Step 2: Built-in SQL Functions (10 Examples)

1. 1. Convert Title to uppercase

SELECT UPPER(Title) AS UpperTitle FROM Trainings;

1. 2. Extract year from StartDate

SELECT Title, YEAR(StartDate) AS StartYear FROM Trainings;

1. 3. Calculate training month

SELECT Title, MONTHNAME(StartDate) AS TrainingMonth FROM Trainings;

1. 4. Find string length of Title

SELECT Title, LENGTH(Title) AS TitleLength FROM Trainings;

1. 5. Round fees

SELECT Title, ROUND(Fees, -3) AS RoundedFees FROM Trainings;

1. 6. Replace word in Title

SELECT REPLACE(Title, 'SQL', 'Structured Query Language') FROM Trainings;

1. 7. Concatenate Title and Trainer

SELECT CONCAT(Title, ' by ', Trainer) AS TrainingDetails FROM Trainings;

1. 8. Days between start and end

SELECT Title, DATEDIFF(EndDate, StartDate) AS Duration FROM Trainings;

1. 9. Find minimum and maximum fees

SELECT MIN(Fees) AS MinFee, MAX(Fees) AS MaxFee FROM Trainings;

1. 10. Count trainings per location

SELECT Location, COUNT(\*) AS TrainingCount FROM Trainings GROUP BY Location;

# Step 3: Stored Procedures & Functions (10 Examples)

Stored procedures and user-defined functions help improve reusability and maintainability of SQL code.

1. 1. Stored Procedure: Get all trainings by location

DELIMITER $$  
CREATE PROCEDURE GetTrainingsByLocation(IN loc VARCHAR(50))  
BEGIN  
 SELECT \* FROM Trainings WHERE Location = loc;  
END$$  
DELIMITER ;

1. 2. Stored Procedure: Insert new training

DELIMITER $$  
CREATE PROCEDURE AddTraining(  
 IN title VARCHAR(100), IN trainer VARCHAR(50),  
 IN startDate DATE, IN endDate DATE,  
 IN duration INT, IN location VARCHAR(50),  
 IN fees DECIMAL(10,2), IN isOnline BOOLEAN  
)  
BEGIN  
 INSERT INTO Trainings (Title, Trainer, StartDate, EndDate, DurationDays, Location, Fees, IsOnline)  
 VALUES (title, trainer, startDate, endDate, duration, location, fees, isOnline);  
END$$  
DELIMITER ;

1. 3. Function: Calculate training cost per day

DELIMITER $$  
CREATE FUNCTION CostPerDay(fees DECIMAL(10,2), duration INT)  
RETURNS DECIMAL(10,2)  
DETERMINISTIC  
BEGIN  
 RETURN fees / duration;  
END$$  
DELIMITER ;

1. 4. Function: Check if training is long (>5 days)

DELIMITER $$  
CREATE FUNCTION IsLongTraining(duration INT)  
RETURNS VARCHAR(20)  
DETERMINISTIC  
BEGIN  
 RETURN IF(duration > 5, 'Long Training', 'Short Training');  
END$$  
DELIMITER ;

1. 5. Function: Categorize fees

DELIMITER $$  
CREATE FUNCTION FeeCategory(fee DECIMAL(10,2))  
RETURNS VARCHAR(20)  
DETERMINISTIC  
BEGIN  
 RETURN CASE   
 WHEN fee < 5000 THEN 'Low Cost'  
 WHEN fee BETWEEN 5000 AND 10000 THEN 'Medium Cost'  
 ELSE 'High Cost'  
 END;  
END$$  
DELIMITER ;

# Step 4: Advanced SQL Topics (10 Examples)

1. 1. Subquery: Trainings with fees above average

SELECT \* FROM Trainings WHERE Fees > (SELECT AVG(Fees) FROM Trainings);

1. 2. Subquery: Trainings with max duration

SELECT \* FROM Trainings WHERE DurationDays = (SELECT MAX(DurationDays) FROM Trainings);

1. 3. View: Online trainings

CREATE VIEW OnlineTrainings AS SELECT Title, Trainer, Fees FROM Trainings WHERE IsOnline = 1;

1. 4. View: Expensive trainings

CREATE VIEW ExpensiveTrainings AS SELECT Title, Fees FROM Trainings WHERE Fees > 10000;

1. 5. Trigger: Prevent fees less than 3000

DELIMITER $$  
CREATE TRIGGER trg\_MinFees  
BEFORE INSERT ON Trainings  
FOR EACH ROW  
BEGIN  
 IF NEW.Fees < 3000 THEN  
 SIGNAL SQLSTATE '45000'  
 SET MESSAGE\_TEXT = 'Fees cannot be less than 3000';  
 END IF;  
END$$  
DELIMITER ;

1. 6. Trigger: Auto set DurationDays

DELIMITER $$  
CREATE TRIGGER trg\_SetDuration  
BEFORE INSERT ON Trainings  
FOR EACH ROW  
BEGIN  
 SET NEW.DurationDays = DATEDIFF(NEW.EndDate, NEW.StartDate);  
END$$  
DELIMITER ;

1. 7. Nested subquery: Trainings by trainer with highest fee

SELECT \* FROM Trainings WHERE Trainer = (SELECT Trainer FROM Trainings ORDER BY Fees DESC LIMIT 1);

1. 8. Inline view: Rank trainings by fees

SELECT Title, Fees, RANK() OVER (ORDER BY Fees DESC) AS FeeRank FROM Trainings;

1. 9. Update with subquery: Increase fees of cheaper trainings

UPDATE Trainings SET Fees = Fees \* 1.1 WHERE Fees < (SELECT AVG(Fees) FROM (SELECT \* FROM Trainings) t);

1. 10. Delete with subquery: Delete trainings longer than average

DELETE FROM Trainings WHERE DurationDays > (SELECT AVG(DurationDays) FROM (SELECT \* FROM Trainings) t);