**Step 1: Create the employees table**

sql

CopyEdit

CREATE TABLE employees (

emp\_id INT PRIMARY KEY,

emp\_name VARCHAR(50),

age INT,

gender VARCHAR(10),

dept VARCHAR(20),

salary INT,

experience INT

);

**✅ Step 2: Insert 10 rows into the table**

sql

CopyEdit

INSERT INTO employees VALUES (1, 'Akash', 28, 'Male', 'IT', 70000, 4);

INSERT INTO employees VALUES (2, 'Meena', 25, 'Female', 'HR', 50000, 2);

INSERT INTO employees VALUES (3, 'Ravi', 35, 'Male', 'Finance', 90000, 8);

INSERT INTO employees VALUES (4, 'Priya', 30, 'Female', 'IT', 85000, 6);

INSERT INTO employees VALUES (5, 'John', 45, 'Male', 'Admin', 60000, 12);

INSERT INTO employees VALUES (6, 'Sara', 26, 'Female', 'HR', 52000, 3);

INSERT INTO employees VALUES (7, 'Kiran', 38, 'Male', 'Finance', 95000, 10);

INSERT INTO employees VALUES (8, 'Divya', 29, 'Female', 'IT', 78000, 5);

INSERT INTO employees VALUES (9, 'Vinay', 41, 'Male', 'Admin', 61000, 11);

INSERT INTO employees VALUES (10, 'Anu', 32, 'Female', 'IT', 88000, 7);

**✅ Step 3: 20 SQL Statements with IF, IF ELSE, SWITCH using CASE**

**1. IF logic: Show employees with experience > 5 years**

SELECT emp\_name, experience,

CASE

WHEN experience > 5 THEN 'Experienced'

ELSE 'Fresher'

END AS experience\_level

FROM employees;

**2. IF ELSE logic: Show age category**

SELECT emp\_name, age,

CASE

WHEN age < 30 THEN 'Young'

ELSE 'Senior'

END AS age\_group

FROM employees;

**3. SWITCH logic (Simple CASE): Department code**

SELECT emp\_name, dept,

CASE dept

WHEN 'IT' THEN 'D001'

WHEN 'HR' THEN 'D002'

WHEN 'Finance' THEN 'D003'

WHEN 'Admin' THEN 'D004'

ELSE 'Unknown'

END AS dept\_code

FROM employees;

**4. Salary Band (using Searched CASE)**

SELECT emp\_name, salary,

CASE

WHEN salary < 60000 THEN 'Low'

WHEN salary BETWEEN 60000 AND 80000 THEN 'Medium'

ELSE 'High'

END AS salary\_band

FROM employees;

**5. Gender normalization**

SELECT emp\_name, gender,

CASE gender

WHEN 'Male' THEN 'M'

WHEN 'Female' THEN 'F'

ELSE 'O'

END AS gender\_short

FROM employees;

**6. Bonus eligibility**

SELECT emp\_name, salary,

CASE

WHEN salary > 80000 THEN 'Eligible'

ELSE 'Not Eligible'

END AS bonus\_status

FROM employees;

**7. Employee title by experience**

SELECT emp\_name, experience,

CASE

WHEN experience < 3 THEN 'Junior'

WHEN experience BETWEEN 3 AND 6 THEN 'Mid-level'

ELSE 'Senior'

END AS level

FROM employees;

**8. Categorize departments**

SELECT emp\_name, dept,

CASE

WHEN dept IN ('IT', 'Finance') THEN 'Core'

ELSE 'Support'

END AS dept\_category

FROM employees;

**9. Age + Experience logic (nested IF)**

SELECT emp\_name,

CASE

WHEN age < 30 AND experience < 5 THEN 'Growing'

WHEN age >= 30 AND experience >= 5 THEN 'Stable'

ELSE 'Shifting'

END AS career\_stage

FROM employees;

**10. Multiple departments switch logic**

SELECT emp\_name, dept,

CASE dept

WHEN 'IT' THEN 'Technology'

WHEN 'Finance' THEN 'Accounts'

WHEN 'HR' THEN 'People'

WHEN 'Admin' THEN 'Operations'

ELSE 'Other'

END AS dept\_fullform

FROM employees;

**11. Retirement eligibility**

SELECT emp\_name, age,

CASE

WHEN age >= 60 THEN 'Retired'

ELSE 'Active'

END AS status

FROM employees;

**12. Salary hike suggestion**

SELECT emp\_name, salary,

CASE

WHEN salary < 60000 THEN '10% Hike'

WHEN salary BETWEEN 60000 AND 80000 THEN '5% Hike'

ELSE 'No Hike'

END AS hike\_plan

FROM employees;

**13. Promotion chance**

SELECT emp\_name, experience,

CASE

WHEN experience >= 10 THEN 'High'

WHEN experience BETWEEN 5 AND 9 THEN 'Medium'

ELSE 'Low'

END AS promotion\_chance

FROM employees;

**14. Employee segment by salary**

SELECT emp\_name, salary,

CASE

WHEN salary < 50000 THEN 'Entry'

WHEN salary BETWEEN 50000 AND 75000 THEN 'Middle'

WHEN salary > 75000 THEN 'Executive'

END AS segment

FROM employees;

**15. Gender-wise greeting**

SELECT emp\_name, gender,

CASE gender

WHEN 'Male' THEN 'Hello Sir'

WHEN 'Female' THEN 'Hello Ma’am'

END AS greeting

FROM employees;

**16. IT or Not**

SELECT emp\_name, dept,

CASE

WHEN dept = 'IT' THEN 'Yes'

ELSE 'No'

END AS is\_it\_dept

FROM employees;

**17. Experience-based job title**

SELECT emp\_name, experience,

CASE

WHEN experience < 2 THEN 'Trainee'

WHEN experience < 5 THEN 'Executive'

WHEN experience < 8 THEN 'Manager'

ELSE 'Lead'

END AS title

FROM employees;

**18. Age eligibility for overseas travel**

SELECT emp\_name, age,

CASE

WHEN age < 25 THEN 'Too Young'

WHEN age BETWEEN 25 AND 40 THEN 'Eligible'

ELSE 'Needs Review'

END AS travel\_eligibility

FROM employees;

**19. Simple bonus flag**

SELECT emp\_name, salary,

CASE

WHEN salary > 85000 THEN 'Y'

ELSE 'N'

END AS bonus\_flag

FROM employees;

**20. IF + ELSE in WHERE logic (advanced filtering)**

SELECT emp\_name, salary

FROM employees

WHERE

CASE

WHEN dept = 'IT' THEN salary > 75000

ELSE salary > 60000

END;

**10 Real-World CASE Statement Examples for SQL Developer / ETL Roles**

**1. Categorize salary levels**

SELECT emp\_id, emp\_name, salary,

CASE

WHEN salary < 60000 THEN 'Low'

WHEN salary BETWEEN 60000 AND 80000 THEN 'Medium'

ELSE 'High'

END AS salary\_band

FROM employees;

🔹 *Used for analytics and segmentation.*

**2. Experience-based role assignment**

SELECT emp\_id, emp\_name, experience,

CASE

WHEN experience < 2 THEN 'Trainee'

WHEN experience BETWEEN 2 AND 5 THEN 'Executive'

WHEN experience > 5 THEN 'Manager'

END AS role\_level

FROM employees;

🔹 *Used in data mapping, HR analytics, reporting.*

**3. IF ELSE logic for bonus eligibility**

SELECT emp\_name, salary,

CASE

WHEN salary > 85000 THEN 'Eligible for Bonus'

ELSE 'Not Eligible'

END AS bonus\_status

FROM employees;

🔹 *Used in payroll or incentive logic.*

**4. Switch logic (simple CASE) for department codes**

SELECT emp\_name, dept,

CASE dept

WHEN 'IT' THEN 'D001'

WHEN 'HR' THEN 'D002'

WHEN 'Finance' THEN 'D003'

WHEN 'Admin' THEN 'D004'

ELSE 'D000'

END AS dept\_code

FROM employees;

🔹 *Used for dimensional modeling or master data mapping.*

**5. Gender classification for compliance reports**

SELECT emp\_id, emp\_name, gender,

CASE gender

WHEN 'Male' THEN 'M'

WHEN 'Female' THEN 'F'

ELSE 'Other'

END AS gender\_code

FROM employees;

🔹 *Used for standardizing values during ETL loads.*

**6. Determine travel eligibility based on age and experience**

SELECT emp\_name, age, experience,

CASE

WHEN age < 25 THEN 'Not Eligible'

WHEN experience < 3 THEN 'Needs Training'

ELSE 'Eligible for Travel'

END AS travel\_status

FROM employees;

🔹 *Used in HR dashboards or travel modules.*

**7. Create flags for reporting KPIs**

SELECT emp\_id, salary,

CASE

WHEN salary > 80000 THEN 1

ELSE 0

END AS high\_earner\_flag

FROM employees;

🔹 *Used in BI reports and dashboards.*

**8. Assign performance rating based on salary and experience**

SELECT emp\_id, emp\_name, salary, experience,

CASE

WHEN salary >= 80000 AND experience >= 5 THEN 'A'

WHEN salary >= 60000 THEN 'B'

ELSE 'C'

END AS performance\_rating

FROM employees;

🔹 *Used for performance scoring in appraisal systems.*

**9. Convert department into business unit name**

SELECT emp\_name, dept,

CASE

WHEN dept IN ('IT', 'Finance') THEN 'Core Business'

WHEN dept IN ('HR', 'Admin') THEN 'Support Services'

ELSE 'Unknown'

END AS business\_unit

FROM employees;

🔹 *Used in data warehouse dimension lookups.*

**10. Dynamic ORDER BY using CASE**

SELECT emp\_name, dept, salary

FROM employees

ORDER BY

CASE dept

WHEN 'IT' THEN 1

WHEN 'HR' THEN 2

WHEN 'Finance' THEN 3

ELSE 4

END;

🔹 *Used to control custom sort orders in reporting queries.*

**Student Table**

CREATE TABLE student (

id INT PRIMARY KEY,

name VARCHAR(50),

address VARCHAR(100),

course VARCHAR(50),

age INT,

fees DECIMAL(10, 2)

);

**3. 🧠 Assign scholarship eligibility based on age and course**

SELECT name, age, course,

CASE

WHEN age < 21 AND course = 'Computer Science' THEN 'Eligible'

ELSE 'Not Eligible'

END AS scholarship\_status

FROM student;

**4. 📊 Fee discount suggestion (like IF ELSE logic)**

SELECT name, course, fees,

CASE

WHEN course = 'Biology' THEN fees \* 0.90 -- 10% discount

ELSE fees

END AS final\_fee

FROM student;

**5. 🏷️ Short code for each course (like SWITCH)**

SELECT name, course,

CASE course

WHEN 'Computer Science' THEN 'CS'

WHEN 'Mathematics' THEN 'MATH'

WHEN 'Physics' THEN 'PHY'

WHEN 'Chemistry' THEN 'CHEM'

WHEN 'Biology' THEN 'BIO'

ELSE 'OTH'

END AS course\_code

FROM student;

**6. 📈 Rank students by age group**

SELECT name, age,

CASE

WHEN age < 20 THEN 'Teen'

WHEN age BETWEEN 20 AND 22 THEN 'Young Adult'

ELSE 'Adult'

END AS age\_group

FROM student;

**7. 🏠 Region-wise grouping from address (using LIKE)**

SELECT name, address,

CASE

WHEN address LIKE '%Maple%' THEN 'Zone A'

WHEN address LIKE '%Oak%' THEN 'Zone B'

WHEN address LIKE '%Pine%' THEN 'Zone C'

WHEN address LIKE '%Elm%' THEN 'Zone D'

ELSE 'Other Zone'

END AS region

FROM student;

**8. 🎓 Assign class year based on age**

SELECT name, age,

CASE

WHEN age <= 19 THEN '1st Year'

WHEN age = 20 THEN '2nd Year'

WHEN age = 21 THEN '3rd Year'

ELSE 'Final Year'

END AS class\_year

FROM student;

**9. 📚 Program type: Technical or Non-Technical**

SELECT name, course,

CASE

WHEN course = 'Computer Science' THEN 'Technical'

WHEN course IN ('Mathematics', 'Physics', 'Chemistry') THEN 'Non-Technical'

ELSE 'Life Science'

END AS program\_type

FROM student;

**10. 🎯 Custom sort order for reporting**

SELECT name, course

FROM student

ORDER BY

CASE course

WHEN 'Computer Science' THEN 1

WHEN 'Mathematics' THEN 2

WHEN 'Physics' THEN 3

WHEN 'Chemistry' THEN 4

WHEN 'Biology' THEN 5

ELSE 6

END;

***Longest case statement***

***CREATE TABLE students (***

***id INT PRIMARY KEY,***

***name VARCHAR(50),***

***age INT,***

***gender VARCHAR(10),***

***department VARCHAR(50),***

***gpa DECIMAL(3, 2),***

***attendance\_percent INT,***

***city VARCHAR(50),***

***fee\_status VARCHAR(20)***

***);***

***INSERT INTO students VALUES***

***(1, 'Alice', 20, 'Female', 'Computer Science', 3.9, 95, 'Chennai', 'Paid'),***

***(2, 'Bob', 22, 'Male', 'Mathematics', 2.5, 75, 'Mumbai', 'Unpaid'),***

***(3, 'Charlie', 23, 'Male', 'Physics', 3.1, 80, 'Delhi', 'Paid'),***

***(4, 'Divya', 19, 'Female', 'Chemistry', 3.7, 90, 'Chennai', 'Unpaid'),***

***(5, 'Ethan', 21, 'Male', 'Biology', 2.9, 60, 'Kolkata', 'Unpaid'),***

***(6, 'Fathima', 20, 'Female', 'Computer Science', 3.2, 85, 'Chennai', 'Paid'),***

***(7, 'Gopal', 24, 'Male', 'Mathematics', 1.9, 55, 'Mumbai', 'Unpaid'),***

***(8, 'Hema', 22, 'Female', 'Physics', 3.6, 92, 'Delhi', 'Paid'),***

***(9, 'Iqbal', 20, 'Male', 'Chemistry', 2.0, 70, 'Kolkata', 'Unpaid'),***

***(10, 'Jaya', 23, 'Female', 'Biology', 3.8, 98, 'Chennai', 'Paid');***

**3. Longest CASE Statement: Student Evaluation Report**

We will evaluate students based on multiple fields like GPA, attendance\_percent, fee\_status, and department.

SELECT

id, name, gender, department, gpa, attendance\_percent, fee\_status, city,

CASE

-- GPA-based academic performance

WHEN gpa >= 3.8 THEN 'Top Performer'

WHEN gpa BETWEEN 3.5 AND 3.79 THEN 'Excellent'

WHEN gpa BETWEEN 3.0 AND 3.49 THEN 'Good'

WHEN gpa BETWEEN 2.5 AND 2.99 THEN 'Average'

WHEN gpa BETWEEN 2.0 AND 2.49 THEN 'Needs Improvement'

ELSE 'Failing'

END AS academic\_status,

CASE

-- Attendance remarks

WHEN attendance\_percent >= 95 THEN 'Outstanding Attendance'

WHEN attendance\_percent >= 85 THEN 'Very Good Attendance'

WHEN attendance\_percent >= 75 THEN 'Satisfactory Attendance'

WHEN attendance\_percent >= 60 THEN 'Low Attendance'

ELSE 'Critical Attendance'

END AS attendance\_remark,

CASE

-- Final Evaluation combining GPA + Attendance + Fee

WHEN gpa >= 3.5 AND attendance\_percent >= 90 AND fee\_status = 'Paid' THEN 'Scholarship Eligible'

WHEN gpa >= 3.0 AND attendance\_percent >= 85 AND fee\_status = 'Paid' THEN 'Good Standing'

WHEN gpa >= 2.5 AND attendance\_percent >= 70 AND fee\_status = 'Unpaid' THEN 'Fee Defaulter - Warning'

WHEN gpa < 2.5 AND attendance\_percent < 60 THEN 'On Academic Probation'

WHEN fee\_status = 'Unpaid' THEN 'Payment Pending'

ELSE 'Regular'

END AS final\_remark,

CASE department

WHEN 'Computer Science' THEN 'Tech Department'

WHEN 'Mathematics' THEN 'Science & Logic'

WHEN 'Physics' THEN 'Core Science'

WHEN 'Chemistry' THEN 'Life Sciences'

WHEN 'Biology' THEN 'Medical Science'

ELSE 'Other Department'

END AS department\_group

FROM students;

3. Data Table

CREATE TABLE data (

OrderID INT NOT NULL,

CustomerName VARCHAR(20) NOT NULL,

CustomerLocation VARCHAR(20) NOT NULL,

ProductLine VARCHAR(20) NOT NULL,

UnitPrice INT NOT NULL,

Quantity INT NOT NULL,

Total INT NOT NULL

);

insert into data values(1, 'Sarah Lee', 'Mexico City', 'Electronics', 150, 1, 150),(2, 'Michael Wong', 'Toronto', 'Furniture', 300, 1, 300),(3, 'Emily Davis', 'San Francisco', 'Furniture', 150, 3, 450),(4, 'David Kim', 'Vancouver', 'Clothing', 50, 5, 250),(5, 'Sophia Patel', 'Tokyo', 'Electronics', 250, 2, 500),(6, 'Liam Nguyen', 'Mexico City', 'Furniture', 400, 1, 400),(7, 'Isabella Rossi', 'Toronto', 'Clothing', 75, 3, 225),(8, 'Ethan Müller', 'San Francisco', 'Electronics', 180, 2, 360),(9, 'Olivia Sato', 'Vancouver', 'Furniture', 350, 1, 350),(10, 'Noah Dupont', 'Tokyo', 'Clothing', 60, 4, 240),(11, 'Emma Hernandez', 'Mexico City', 'Electronics', 220, 2, 440),(12, 'Jacob Kowalski', 'Toronto', 'Furniture', 280, 2, 560),(13, 'Ava Morales', 'San Francisco', 'Clothing', 55, 5, 275),(14, 'William Tanaka', 'Vancouver', 'Electronics', 190, 3, 570),(15, 'Mia Dupuis', 'Tokyo', 'Furniture', 320, 1, 320),(16, 'Alexander Ivanov', 'Mexico City', 'Clothing', 65, 4, 260),(17, 'Isabella Garcia', 'Toronto', 'Electronics', 230, 2, 460),(18, 'Daniel Moreno', 'San Francisco', 'Furniture', 290, 2, 580),(19, 'Sophia Nguyen', 'Vancouver', 'Clothing', 70, 3, 210),(20, 'John Smith', 'Tokyo', 'Electronics', 200, 2, 400);

**CASE Statements (1–20)**

**1. Product Category Label (Simple CASE)**

sql

CopyEdit

SELECT

OrderID, ProductLine,

CASE ProductLine

WHEN 'Electronics' THEN 'Tech'

WHEN 'Furniture' THEN 'Home'

WHEN 'Clothing' THEN 'Fashion'

ELSE 'Other'

END AS ProductCategory

FROM data;

**2. High or Low Order Value (Searched CASE)**

sql

CopyEdit

SELECT

OrderID, Total,

CASE

WHEN Total >= 500 THEN 'High Value'

ELSE 'Low Value'

END AS OrderValueCategory

FROM data;

**3. Location Grouping (Simple CASE)**

sql

CopyEdit

SELECT

OrderID, CustomerLocation,

CASE CustomerLocation

WHEN 'Toronto' THEN 'Canada'

WHEN 'Vancouver' THEN 'Canada'

WHEN 'Mexico City' THEN 'Mexico'

WHEN 'Tokyo' THEN 'Japan'

WHEN 'San Francisco' THEN 'USA'

ELSE 'Other'

END AS Country

FROM data;

**4. Discount Eligibility (Searched CASE)**

sql

CopyEdit

SELECT

OrderID, Quantity,

CASE

WHEN Quantity >= 3 THEN 'Eligible'

ELSE 'Not Eligible'

END AS DiscountStatus

FROM data;

**5. Bulk Purchase Label**

sql

CopyEdit

SELECT

OrderID,

CASE

WHEN Quantity >= 5 THEN 'Bulk Order'

ELSE 'Regular Order'

END AS PurchaseType

FROM data;

**6. Shipping Tier (Based on Total)**

sql

CopyEdit

SELECT

OrderID, Total,

CASE

WHEN Total >= 500 THEN 'Free Shipping'

WHEN Total >= 300 THEN 'Standard Shipping'

ELSE 'Shipping Charges Apply'

END AS ShippingTier

FROM data;

**7. Customer Segment (Name-based)**

sql

CopyEdit

SELECT

OrderID, CustomerName,

CASE

WHEN CustomerName LIKE 'S%' THEN 'Segment A'

WHEN CustomerName LIKE 'M%' THEN 'Segment B'

ELSE 'Other Segment'

END AS CustomerSegment

FROM data;

**8. City Popularity (Simple)**

sql

CopyEdit

SELECT

CustomerLocation,

CASE CustomerLocation

WHEN 'Tokyo' THEN 'High Demand'

WHEN 'Toronto' THEN 'High Demand'

ELSE 'Normal Demand'

END AS CityStatus

FROM data;

**9. Pricing Tier**

sql

CopyEdit

SELECT

UnitPrice,

CASE

WHEN UnitPrice >= 300 THEN 'Premium'

WHEN UnitPrice >= 200 THEN 'Mid-range'

ELSE 'Economy'

END AS PriceTier

FROM data;

**10. Customer Type Based on Total & Location**

sql

CopyEdit

SELECT

CustomerName,

CASE

WHEN Total > 500 AND CustomerLocation = 'Toronto' THEN 'VIP Canada'

WHEN Total > 500 THEN 'VIP International'

ELSE 'Regular'

END AS CustomerType

FROM data;

**11. Risk Category (based on unit price and quantity)**

sql

CopyEdit

SELECT

OrderID,

CASE

WHEN UnitPrice > 200 AND Quantity > 3 THEN 'High Risk'

WHEN UnitPrice > 200 THEN 'Moderate Risk'

ELSE 'Low Risk'

END AS RiskCategory

FROM data;

**12. Weekend Order Simulation**

sql

CopyEdit

SELECT

OrderID,

CASE

WHEN OrderID % 2 = 0 THEN 'Weekend Order'

ELSE 'Weekday Order'

END AS OrderDay

FROM data;

**13. Label Based on OrderID Range**

sql

CopyEdit

SELECT

OrderID,

CASE

WHEN OrderID <= 5 THEN 'Early Orders'

WHEN OrderID BETWEEN 6 AND 15 THEN 'Mid Orders'

ELSE 'Recent Orders'

END AS OrderPeriod

FROM data;

**14. Currency Suggestion by City**

sql

CopyEdit

SELECT

CustomerLocation,

CASE CustomerLocation

WHEN 'Tokyo' THEN 'JPY'

WHEN 'Mexico City' THEN 'MXN'

WHEN 'Toronto' THEN 'CAD'

ELSE 'USD'

END AS PreferredCurrency

FROM data;

**15. Quantity Alert**

sql

CopyEdit

SELECT

Quantity,

CASE

WHEN Quantity >= 4 THEN 'Stock Check Needed'

ELSE 'Stock OK'

END AS InventoryAlert

FROM data;

**16. Gift Eligible Customers**

sql

CopyEdit

SELECT

CustomerName,

CASE

WHEN Total >= 500 AND Quantity >= 2 THEN 'Gift Eligible'

ELSE 'No Gift'

END AS GiftStatus

FROM data;

**17. Sales Tax Band**

sql

CopyEdit

SELECT

Total,

CASE

WHEN Total > 500 THEN '18% GST'

WHEN Total > 300 THEN '12% GST'

ELSE '5% GST'

END AS TaxRate

FROM data;

**18. Clothing Return Policy**

sql

CopyEdit

SELECT

ProductLine,

CASE

WHEN ProductLine = 'Clothing' THEN 'Return within 7 days'

ELSE 'No Returns'

END AS ReturnPolicy

FROM data;

**19. Profit Margin Band (Assume margin = 25%)**

sql

CopyEdit

SELECT

Total,

CASE

WHEN Total \* 0.25 > 100 THEN 'High Profit'

WHEN Total \* 0.25 > 50 THEN 'Moderate Profit'

ELSE 'Low Profit'

END AS ProfitMarginBand

FROM data;

**20. Sales Tag Based on Combination**

sql

CopyEdit

SELECT

OrderID,

CASE

WHEN ProductLine = 'Electronics' AND Total > 500 THEN 'Hot Deal'

WHEN ProductLine = 'Clothing' AND Quantity >= 4 THEN 'Clearance'

ELSE 'Normal Sale'

END AS SaleTag

FROM data;