



Link to creative demo:

https://drive.google.com/file/d/133Bjg6FK0ENUzeTox06E-wx3DbFX5-gT/view?usp=share_link



```
1 DELIMITER $$
2
3 DROP PROCEDURE IF EXISTS ApplyDiscountByCriteria $$
4
5 CREATE PROCEDURE ApplyDiscountByCriteria(
6     IN p_genre VARCHAR(255),
7     IN p_release_date VARCHAR(255),
8     IN p_game_name VARCHAR(255),
9     IN p_discount_rate DECIMAL(5,2))
10 BEGIN
11     -- Declare necessary variables and the cursor
12     DECLARE done INT DEFAULT FALSE;
13     DECLARE v_gameName VARCHAR(255);
14     DECLARE v_oldPrice DECIMAL(10,4);
15     DECLARE v_oldFinalPrice DECIMAL(10,4);
16     DECLARE v_newPrice DECIMAL(10,4);
17
18     DECLARE gameCursor CURSOR FOR
19         -- do not allow making a discount on bookmarked games, so we can handle that separately later
20         SELECT g.gameName, g.initialPrice
21         FROM Games AS g JOIN GameGenre AS gg ON g.gameName = gg.gameName
22         WHERE (gg.gameGenre = p_genre OR p_genre = '')
23             AND (g.releaseDate = p_release_date OR p_release_date = '')
24             AND (g.gameName LIKE CONCAT('%', p_game_name, '%'))
25             AND g.gameName NOT IN (SELECT g2.gameName
26                                   FROM Games g2 NATURAL JOIN Bookmarks b);
27
28     DECLARE bookmarksCursor CURSOR FOR
29         -- now handle bookmarks separately
30         SELECT g.gameName, g.initialPrice, g.finalPrice
31         FROM Games AS g JOIN GameGenre AS gg ON g.gameName = gg.gameName
32         WHERE (gg.gameGenre = p_genre OR p_genre = '')
33             AND (g.releaseDate = p_release_date OR p_release_date = '')
34             AND (g.gameName LIKE CONCAT('%', p_game_name, '%'))
35             AND g.gameName IN (SELECT g2.gameName
36                               FROM Games g2 NATURAL JOIN Bookmarks b);
37
38     -- Declare a continue handler for NOT FOUND condition
39     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
40
41     -- Open the cursor
42     OPEN gameCursor;
43
44     -- Start the loop
45     read_loop: LOOP
46         -- Fetch the next game from the cursor
47         FETCH gameCursor INTO v_gameName, v_oldPrice;
48
49         -- Check if the cursor reached the end of fetched records
50         IF done THEN
51             LEAVE read_loop;
52         END IF;
53
54         -- Calculate the new price after applying the discount
55         SET v_newPrice = v_oldPrice * (1 - p_discount_rate);
56
57         -- Update the finalPrice in the Games table
58         UPDATE Games
59         SET finalPrice = v_newPrice
60         WHERE gameName = v_gameName;
61
62         -- Continue to the next iteration of the loop
63     END LOOP;
64
65     -- Close the cursor
66     CLOSE gameCursor;
67
68     SET done = false;
69
70     -- Open the cursor
71     OPEN bookmarksCursor;
72
73     -- Start the loop
74     read_loop_two: LOOP
75         -- Fetch the next game from the cursor
76         FETCH bookmarksCursor INTO v_gameName, v_oldPrice, v_oldFinalPrice;
77
78         -- Check if the cursor reached the end of fetched records
79         IF done THEN
80             LEAVE read_loop_two;
81         END IF;
82
83         -- Calculate the new price after applying the discount
84         SET v_newPrice = v_oldPrice * (1 - p_discount_rate);
85
86         -- Only apply discount to bookmarked games when cheaper
87         IF v_newPrice < v_oldFinalPrice THEN
88             -- Update the finalPrice in the Games table
89             UPDATE Games
90             SET finalPrice = v_newPrice
91             WHERE gameName = v_gameName;
92         END IF;
93         -- Continue to the next iteration of the loop
94     END LOOP;
95
96     -- Close the cursor
97     CLOSE bookmarksCursor;
98
99     -- After closing the cursor, SELECT the updated records, along with lowest min ever price
100    SELECT g.gameName, gg.gameGenre, g.releaseDate, pl.old_price, pl.new_price, (SELECT MIN(pl2.new_price) FROM price_change_log pl2 WHERE pl.gameName = pl2.gameName GROUP BY pl2.gameName) as lowest_price
101    FROM price_change_log AS pl JOIN Games AS g ON pl.gameName = g.gameName JOIN GameGenre AS gg ON g.gameName = gg.gameName
102    WHERE pl.changed_at >= NOW() - INTERVAL 1 SECOND;
103 END $$
```



```
1  CREATE TRIGGER log_price_change
2  AFTER UPDATE ON Games
3  FOR EACH ROW
4  BEGIN
5      IF NEW.finalPrice <> OLD.finalPrice THEN
6          INSERT INTO price_change_log (gameName, old_price, new_price, changed_at)
7          VALUES (OLD.gameName, OLD.finalPrice, NEW.finalPrice, NOW());
8      END IF;
9  END$$
10
11 DELIMITER ;
12
```