CS306 PROJECT PHASE 2

Ayberk Savaş

Spring 2025 Sabanci University

Introduction

This project aims to enhance the functionality and maintainability of a watch sales database by implementing one trigger and one stored procedure, as required for a solo participant. The trigger focuses on automating the logging of sales transactions, while the stored procedure simplifies stock tracking. These additions improve data integrity, usability, and operational efficiency in managing watch sales and inventory.

TRIGGER

To improve functionality and maintain ease of use, I implemented a _WATCH_LOG_ table to keep track of watches sold. In my system, each watch is stored in the _WATCH_ table regardless of whether it has been sold or not. A watch that hasn't been purchased yet has NULL value in its order_id field. Once it is sold, the order_id is updated with the corresponding order. To automate the logging process, I created the logWatchSale trigger, which detects when a watch's order_id changes from NULL to a valid value. This trigger then inserts the watch's details into the _WATCH_LOG_ table. In summary, available watches are stored in the _WATCH_ table with order_id = NULL, and once they are purchased, they are automatically logged in the _WATCH_LOG_ table for tracking and reporting purposes.

```
CREATE TABLE _WATCH_LOG_ (
  log_id      INT AUTO_INCREMENT PRIMARY KEY,
  watch_id    CHAR(20),
  watch_name  CHAR(20),
  order_id    CHAR(20),
  sold_at     TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
DELIMITER $$
CREATE TRIGGER logWatchSale
AFTER UPDATE ON _WATCH_
FOR EACH ROW
BEGIN
    IF OLD.order_id IS NULL AND NEW.order_id IS NOT NULL THEN
        INSERT INTO _WATCH_LOG_ (watch_id, watch_name, order_id)
        VALUES (NEW.watch_id, NEW.watch_name, NEW.order_id);
    END IF;
END$$

DELIMITER ;
```

STORED PROCEDURE

In my implementation, each watch is stored as a separate record on the _WATCH_ table. To determine the available stock of a specific model, we simply count the number of watches with the given watch_name that have a NULL value for order_id, indicating they haven't been sold yet. The stored procedure GetStock automates this process. By providing the name of the watch model—for example, CALL GetStock('Kanno')—the procedure returns the current number of unsold units for that model, making stock tracking efficient and user-friendly.

```
CREATE PROCEDURE GetStock(IN model_name CHAR(20))
BEGIN
    SELECT watch_name, COUNT(*) AS available_stock
    FROM _WATCH_
    WHERE watch_name = model_name AND order_id IS NULL
    GROUP BY watch_name;
END$$

DELIMITER ;
```