# 3-Hour Practical Evaluation – MySQL & Python WebSocket

## Section A – MySQL Practical (40 marks)

1. Database Setup & Normalization (5 marks)  
- Create normalized tables for an e-commerce order system:  
 customers(customer\_id, name, email, city)  
 orders(order\_id, customer\_id, order\_date, amount)  
 order\_items(item\_id, order\_id, product\_name, quantity, price)  
- Define primary keys, foreign keys, and indexes for optimal performance.

2. Complex Query – Top Customers Per City (7 marks)  
- Write a single query to find top 3 customers per city based on total amount spent in the last 12 months.

3. Duplicate Removal (5 marks)  
- In the orders table, delete duplicate rows keeping only the record with the smallest order\_id for each (customer\_id, order\_date).

4. Pivot Report (8 marks)  
- Create a query to display total orders per month for 2024, showing months as columns.

5. Query Optimization (7 marks)  
- You are given the query:  
SELECT \* FROM orders WHERE YEAR(order\_date) = 2024 AND amount > 500 ORDER BY order\_date DESC;  
- Rewrite it for maximum efficiency using indexes and avoiding functions on indexed columns.

6. Trigger Implementation (8 marks)  
- Create a MySQL trigger that automatically logs all deleted orders into a table deleted\_orders with columns: (order\_id, deleted\_at, deleted\_by).

## Section B – Python WebSocket Practical (40 marks)

1. Basic WebSocket Server (8 marks)  
- Create a WebSocket server that accepts connections and sends 'Welcome <client\_id>' upon connection.

2. Broadcast Chat Server (8 marks)  
- Extend the server to broadcast messages from any client to all connected clients.

3. Rate Limiting (6 marks)  
- Implement rate limiting: no client can send more than 5 messages in 10 seconds. If exceeded, disconnect them.

4. Idle Timeout (6 marks)  
- Disconnect clients if they are idle for more than 30 seconds.

5. Authentication (6 marks)  
- Require clients to send an authentication token in the first message. Disconnect if invalid.

6. Client Script (6 marks)  
- Write a Python WebSocket client that connects, authenticates, sends a message every 5 seconds, and prints all received messages.

## Section C – Integrated Task (20 marks)

Scenario:  
You are building a real-time product inventory dashboard where updates from multiple sellers come in through WebSockets and are stored in MySQL.

1. MySQL Table Design (5 marks)  
- Create the inventory\_updates table with proper indexing to handle high-frequency inserts and allow fast retrieval of latest stock per product.

2. Integration Code (15 marks)  
- Write Python WebSocket server code that:  
 - Accepts messages in JSON format:  
 {  
 "seller\_id": 12,  
 "product\_id": 45,  
 "quantity": 120,  
 "updated\_at": "2025-08-08T10:30:00"  
 }  
 - Inserts into MySQL using connection pooling.  
 - Broadcasts the latest stock update to all connected clients in real-time.