## Part 1: Database & Query Optimization (MySQL)

Scenario:  
You are given a dataset of shipments stored in a MySQL table:

CREATE TABLE shipments (  
 id BIGINT PRIMARY KEY AUTO\_INCREMENT,  
 order\_id VARCHAR(50),  
 carrier\_name VARCHAR(50),  
 ship\_date DATETIME,  
 delivery\_date DATETIME,  
 weight DECIMAL(10,2),  
 origin\_province VARCHAR(20),  
 destination\_province VARCHAR(20),  
 status VARCHAR(20),  
 cost DECIMAL(10,2)  
);

Tasks:  
1. Propose 3 indexes you would add and explain why.

1. order\_id frequently will be use for searching shipment using order\_id
2. index on status and delivery\_date (it will be use for filtration and report purpose frequently as per requirement in part3
3. index on ship\_date, carrier\_name and delivery\_date, it is require for aggregation type query as per requirement of part1 query requirement, average delivery time per carrier for last 30 days.

2. Write a SQL query to:  
 - Find the average delivery time per carrier for the last 30 days.

Query:

1. Select carrier\_name, AVG(TIMESTAMPDIFF(HOUR, ship\_date, delivery\_date)) AS avg\_hours FROM shipments WHERE ship\_date >= NOW() - INTERVAL 30 DAY GROUP BY carrier\_name

- Identify shipments where the delivery time exceeded the carrier’s average by more than 25%.

Query:

1. Select a. order\_id, a.carrier\_name, a.delivery\_hours, b.avg\_hours, round((a.delivery\_hours-b.avg\_hours)/b.avg\_hours \*100, 2) as prcntile from (Select order\_id, carrier\_name, TIMESTAMPDIFF(HOUR, ship\_date, delivery\_date) AS delivery\_hours from shipments) as a join  
    (SELECT carrier\_name, AVG(TIMESTAMPDIFF(HOUR, ship\_date, delivery\_date)) AS avg\_hours FROM shipments WHERE ship\_date >= NOW() - INTERVAL 30 DAY GROUP BY carrier\_name) AS b on b.carrier\_name=a.carrier\_name where a.delivery\_hours > b.avg\_hours having prcntile > 25.0

or

1. Select a. order\_id, a.carrier\_name, a.delivery\_hours, b.avg\_hours, (b.avg\_hours+(b.avg\_hours\*25/100)) as increasevalue FROM (SELECT order\_id, carrier\_name, TIMESTAMPDIFF(HOUR, ship\_date, delivery\_date) AS delivery\_hours FROM shipments) AS a JOIN   
   (SELECT carrier\_name, AVG(TIMESTAMPDIFF(HOUR, ship\_date, delivery\_date)) AS avg\_hours FROM shipments WHERE ship\_date >= NOW() - INTERVAL 30 DAY GROUP BY carrier\_name)   
   AS b ON b.carrier\_name=a.carrier\_name WHERE a.delivery\_hours > b.avg\_hours HAVING a.delivery\_hours > increasevalue

3. Describe how you would optimize performance if the table grows to 100 million rows.

Ans:

1. We can use table portioning option on date or on id base.
2. We can check all query which running for the table and create index as per query running on it.
3. Avoid select \* query, use with column name instead of \*.
4. We can check the hardware of server if running on normal hdd use ssd and increase ram if not enough in server.
5. We can go with table switching concept monthly base table creation with (table + monthyear) and create lookup table of table switching.

## Part 4: Data Scenario Discussion (Short Write-Up)

In under 300 words:  
Describe a challenging data migration or performance issue you solved — what was the root cause, how did you identify it, and what was your fix?

Ans:

1. Use of index as per given requirement write query and create index accordingly and check the query index usage using explain command where type=all showing it means index is not using and scanning all records, checked queries with single index and checked with compound index.
2. Checked csv files and clean the data like null values field, remove duplicate value on the basis of order\_id, fill the 0 value where cost field is null or blank after completion this send to next stage where data uploading and keeping rejects and success records in output after upload completion. Next stage send notification of upload with summary report having details of rejected, success and taken time in process start and end time.
3. Records was uploading one by one and find it was running slow to complete upload it take more time, to resolve the issue use 1000 insertion at a time.