1. SIMPLE QUERIES (SELECT), BASIC FETCHES (NO BUSINESS LOGIC)

These are pure, direct SELECTs. They:

- Populate dropdown lists
- Display raw data
- DO NOT apply logic
- DO NOT calculate
- DO NOT edit anything
- Maybe implemented as VIEWS if repeated

These queries DO NOT need stored procedures (but you can wrap them if you want consistency).

A. User Fetch Queries

1. Get user by ID
2. Get user by username (login)
3. Get all users
4. Get all drivers
5. Get all operators
6. Get all passengers
7. Get all company representatives

B. Vehicle Fetch Queries

8. Get all vehicles
9. Get vehicles by driver
10. Get vehicle docs
11. Get vehicle inspections
12. Get service types for vehicle

→ A good place for views
Examples:

- vwVehicleDetails
- vwDriverVehicles

C. Trips (non-recursive layer)

13. Get all total trips
14. Get total trips by user
15. Get simple trip segments (no chaining)
16. Get segment details with driver & passenger
17. Get payment for a trip
18. Get payments by user
19. Get GDPR logs

D. Messaging

20. Get messages by trip
21. Get latest message preview

These require NO triggers, NO heavy SQL — just SELECTs.

2. ADVANCED QUERIES (MUST BE STORED PROCEDURES)

Includes **recursion, calculations, edits, logging, cascading, inserts, updates, deletes**, and **reports**.

This category determines the majority of your marks.

A. MODIFYING DATA (Insert/Update/Delete) → ALWAYS STORED PROCEDURES

Because grading states “all data management must use SPs”.

SP list (required):

1. Insert User

- Hash password

- Validate type
- Log insertion (Trigger INSERT INTO Log table)

2. Update User

- Validate
- Write to log via trigger

3. Delete User (GDPR)

- Must call internal procedures
- Must NOT delete payment history
- Cascade or pseudo-delete personal data
- Log deletion

4. Insert Vehicle

- Validate driver
- Log insert
- Auto-create default service types if needed
- Cascading service types

5. Insert Driver Document

- Validate document type
- Trigger logs audit trail

6. Insert Vehicle Document

- Similar logic
- Cascade updates on expiration date if needed

7. Insert/Update/Delete Trip Segment

- Must validate vehicle availability
- Must log each edit

8. Insert/Update/Delete Payment

- Update TotalTrip
- Validate minimum pricing
- Trigger logs operator actions

9. Insert GDPR Request

- Write log entry
- Auto-generate workflow state

B. ADVANCED QUERY LOGIC (non-editing) → ALWAYS STORED PROCEDURES

These handle **calculations**, **routing**, **recursive logic**, **aggregation**, **filtering**, and **complex joins**.

10. Recursive Trip Assembly

- Build complete segment chain
- CTE or Cursor method

11. Bridge ↔ Geofence Pathfinding

- Required for "transport with intermediate points"
- Must return reachable segments

12. Dynamic Driver/Vehicle Assignment

- Nearest driver
- Within same geofence
- Matches service type
- Driver availability
- Vehicle condition
- Trip readiness

13. Dynamic Price Calculation Procedure

- Applies minimum pricing rules
- Calculates costs based on service type
- Handles automatic campaign adjustments

C. ANALYTIC REPORTS (Mandatory SPs)

Each report:

- Up to 3 filters

- Up to 3 grouping options
- SP receives parameters (NULL = no filter)

Required reports:

14. Trip Statistics Report

Filters:

- Time range
- Service type
- Location

Groupings:

- Service type
- Day
- Month

15. Driver Performance Report

Filters:

- Driver ID
- Time range
  Groupings:
- Month
- Service Type

Outputs:

- Trip counts
- Avg rating
- Earnings (JOIN Payment)

16. Driver Earnings Report

Filters:

- Driver ID
- Year
  Groupings:
- Month

17. Passenger Usage Report

Filters:

- User ID
- Time range
  Groupings:
- Service type

18. Most Expensive Trips

Filters:

- Service type
  Groupings:
- None (top 10)

19. Peak Usage Report

Filters:

- Time range
  Groupings:
- Hour
- Day
- Month

20. Vehicle Utilization Report

Filters:

- Vehicle ID
- Time range
  Groupings:
- Day
- Month

21. GDPR Audit Report

Filters:

- Requester
- Operator

- Time Range

3. TRIGGERS YOU MUST IMPLEMENT (for marks)

Based on grading:

✓ Semantic constraints
✓ Logging/versioning
✓ Auditing
✓ Precommit checks

Minimum required:

A. Log changes (INSERT/UPDATE/DELETE)

1. trgUserLog
2. trgDriverDocLog
3. trgVehicleDocLog
4. trgTripSegmentLog
5. trgPaymentAudit

B. Semantic constraints

6. Prevent expired documents
7. Prevent driver under 18
8. Prevent impossible trip times (Departure_Time < Arrive_Time)

C. Cascading

9. Cascading delete OR pseudo-delete user data (GDPR)
10. Cascading updates for vehicle reassignments
11. Cascading deletions for temp Trip_Segment edits

4. VIEWS YOU SHOULD IMPLEMENT (optional but useful)

Views increase readability and help PHP avoid multi-joins.

Recommended:

1. vwDriverVehicles
2. vwVehicleDetails
3. vwUserFeedbackSummary
4. vwTripWithPayment
5. vwTripSegmentExpanded
6. vwGDPRHistory

Views DO NOT replace SPs (reports still require T-SQL logic).

5. MINIMAL PHP FILES (REVISED FOR TRIGGERS + LOGGING + SP USE)

No logic in PHP except determining **which stored procedure to call**.

CORE

- db_connection.php
- login.php
- logout.php
- dashboard.php

USERS

- user_profile.php
- user_edit.php
- user_delete_gdpr.php → calls SP

DRIVERS/VEHICLES

- driver_dashboard.php
- vehicle_list.php
- vehicle_details.php
- vehicle_upload_doc.php

TRIPS

- trip_list.php
- trip_details.php → calls recursive SP
- trip_request.php

PAYMENTS

- payment_details.php

GDPR

- gdpr_request.php
- gdpr_status.php

REPORTS

- report_trips.php
- report_driver_perf.php
- report_earnings.php
- report_usage_patterns.php

Each report-page:

- reads filter inputs
- chooses which SP to call
- DOES NOT apply filtering in PHP