CPS 510 - Assignment 9

Shayaan Kirubakaran (Section 5) - T36

Kirusanth Palakanthan (Section 10) - Group 70 T36

Ahmed Hasan (Section 5) - T36

Website Link: https://webdev.scs.ryerson.ca/~kpalakan/assignment9.php

Login

The Ride & Pickup DBMS uses the Oracle database login built directly into the PHP code, meaning only users with valid Oracle credentials can run the application. Without the correct connection details, the webpage cannot access the database and no features will load. Access to the site itself is additionally protected by the TMU VPN, which restricts both the hosted webpage and the Oracle server to authenticated TMU users only. Because the VPN must be connected before the page can be opened, this acts as an automatic security layer that prevents unauthorized access. If the database needs to be restored, the webpage includes options to drop tables, create tables, and repopulate them with default data, allowing authorized users to safely reset the system at any time.

Code: (Assignment9.php)

```
<?php
// CPS510 - Assignment 9 (Ride & Pickup DBMS)
// SINGLE PHP FILE — Works with SQL files in same folder

// ============================================
// 1. ORACLE DATABASE CONNECTION
// ============================================
$conn = oci_connect(
    'kpalakan',
    '02102510',

'(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=152
1))
    (CONNECT_DATA=(SID=orcl)))'
);

if (!$conn) {
    $e = oci_error();
    die("<h2>DB Connection Failed</h2><pre>".$e['message']."</pre>");
}

// ============================================
// 2. START SESSION + CSRF
// ============================================
ini_set('session.cookie_httponly', 1);
ini_set('session.use_strict_mode', 1);
```

```
if (session_status() === PHP_SESSION_NONE) session_start();

if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(24));
}
$CSRF = $_SESSION['csrf_token'];

// ============================================
// 3. HELPERS
// ============================================
function h($s) { return htmlspecialchars((string)$s, ENT_QUOTES |
ENT_SUBSTITUTE, 'UTF-8'); }
function is_post() { return $_SERVER['REQUEST_METHOD'] === 'POST'; }
function check_csrf($token) {
    return isset($_SESSION['csrf_token']) &&
hash_equals($_SESSION['csrf_token'], $token);
}

// Whitelist our known table names here (adjust to match our schema)
$ALLOWED_TABLES = [
    'Address','Customer','Driver','Vehicle','Merchant','Location',
    'Service_Order','Payment','Rating_System'
];

function validate_identifier($name, $allowed) {
    if (!in_array($name, $allowed, true)) return false;
    return preg_match('/^[A-Za-z0-9_]+$/', $name);
}

// Basic client-side friendly validation helper (server-side fallback)
function validate_input_value($val) {
    // trim and limit length to avoid extremely long inputs
    $v = trim((string)$val);
    if (strlen($v) > 2000) return false;
    return $v;
}

// ============================================
// 4. SQL FUNCTIONS (safer handling)
// ============================================
function runSqlFile($conn, $filename) {
    if (!file_exists($filename)) {
        echo "<div class='error'>ERROR: File not found: " . h($filename) .
"</div>";
        return;
    }

    $sql = file_get_contents($filename);
    // split statements on semicolon + newline (keeps simple PL/SQL mostly
intact)
    $statements = preg_split('/;[\\s]*\r?\n/', $sql);
    foreach ($statements as $stmt) {
        $trim = trim($stmt);
        if ($trim === "") continue;
        $stid = oci_parse($conn, $trim);
        if (!@oci_execute($stid)) {
            $err = oci_error($stid);
            echo "<div class='error'>SQL execution error: " .
h($err['message']) . "</div>";
        }
    }
}

function runQuery($conn, $query, $bindings = []) {
    $stid = oci_parse($conn, $query);

    // Bind variables safely (oci_bind_by_name binds by reference)
    $refs = [];
    $i = 0;
    foreach ($bindings as $name => $val) {
        $i++;
        $var = "bv_" . $i;
        $$var = $val;
        $bindName = (strpos($name, ':') === 0) ? $name : ':' . $name;
        oci_bind_by_name($stid, $bindName, $$var);
        $refs[] = &$$var; // hold references to avoid garbage collection
    }

    if (!@oci_execute($stid)) {
        $err = oci_error($stid);
```

```php
        throw new RuntimeException("Query error: " . $err['message']);
    }
    return $stid;
}

function printTable($stid) {
    echo "<div class='table-wrap'><table class='result'>";
    echo "<thead><tr>";
    $ncols = oci_num_fields($stid);
    for ($i = 1; $i <= $ncols; $i++) {
        echo "<th>" . h(oci_field_name($stid, $i)) . "</th>";
    }
    echo "</tr></thead><tbody>";
    while ($row = oci_fetch_array($stid, OCI_ASSOC+OCI_RETURN_NULLS)) {
        echo "<tr>";
        foreach ($row as $col) {
            echo "<td>" . h($col) . "</td>";
        }
        echo "</tr>";
    }
    echo "</tbody></table></div>";
}

// ============================================
// 5. HANDLE ACTIONS (view, search, update, delete, run SQL files, run
queries)
// ============================================
$errors = [];
$messages = [];

if (isset($_GET['action'])) {
    $action = $_GET['action'];

    if (in_array($action, ['drop','create','populate','queries'], true)) {
        // these run SQL files or queries.sql
        $token_ok = isset($_GET['token']) &&
hash_equals($_SESSION['csrf_token'], $_GET['token']);
        if (!$token_ok) {
            $errors[] = "Missing or invalid token for action: " .
h($action);
```

```php
    } else {
        try {
            if ($action === "drop") {
                runSqlFile($conn, "drop_tables.sql");
                $messages[] = "Tables dropped successfully.";
            } elseif ($action === "create") {
                runSqlFile($conn, "create_tables.sql");
                $messages[] = "Tables created successfully.";
            } elseif ($action === "populate") {
                runSqlFile($conn, "populate_tables.sql");
                $messages[] = "Tables populated successfully.";
            } elseif ($action === "queries") {
                $messages[] = "Executing queries from queries.sql";
                $content = file_get_contents("queries.sql");
                $queries = preg_split('/;[\\s]*\r?\n/', $content);
                foreach ($queries as $q) {
                    $t = trim($q);
                    if ($t === "") continue;
                    $messages[] = $t;
                    try {
                        $stid = runQuery($conn, $t);
                        ob_start();
                        printTable($stid);
                        echo ob_get_clean();
                    } catch (Exception $e) {
                        $errors[] = $e->getMessage();
                    }
                }
            }
        } catch (Exception $e) {
            $errors[] = $e->getMessage();
        }
    }
}

// view table (GET)
if ($action === 'view' && isset($_GET['table'])) {
    $table = $_GET['table'];
    if (!validate_identifier($table, $ALLOWED_TABLES)) {
        $errors[] = "Invalid table name.";
        } else {
            try {
                $stid = runQuery($conn, "SELECT * FROM " . $table);
                $messages[] = "Viewing table: " . h($table);
                ob_start();
                printTable($stid);
                echo ob_get_clean();
            } catch (Exception $e) {
                $errors[] = $e->getMessage();
            }
        }
    }
}

// POST actions: search, update, delete
if (is_post()) {
    $act = $_POST['action'] ?? '';
    $token = $_POST['csrf_token'] ?? '';
    if (!check_csrf($token)) {
        $errors[] = "Invalid CSRF token.";
    } else {
        if ($act === 'search') {
            $t = trim($_POST['table'] ?? '');
            $field = trim($_POST['field'] ?? '');
            $value = $_POST['value'] ?? '';

            if (!validate_identifier($t, $ALLOWED_TABLES) ||
!preg_match('/^[A-Za-z0-9_]+$/', $field)) {
                $errors[] = "Invalid table or field name.";
            } else {
                $val = validate_input_value($value);
                if ($val === false) { $errors[] = "Invalid search value.";
}
                else {
                    try {
                        $sql = "SELECT * FROM $t WHERE $field = :val";
                        $stid = runQuery($conn, $sql, [':val' => $val]);
                        $messages[] = "Search results in " . h($t) . " for
" . h($field) . " = " . h($val);
```

```
                        ob_start(); printTable($stid); echo
ob_get_clean();
                    } catch (Exception $e) {
                        $errors[] = $e->getMessage();
                    }
                }
            }
        }

        if ($act === 'update') {
            $t = trim($_POST['table'] ?? '');
            $field = trim($_POST['field'] ?? '');
            $value = $_POST['value'] ?? '';
            $pk = trim($_POST['pk'] ?? '');
            $pkval = $_POST['pkval'] ?? '';

            if (!validate_identifier($t, $ALLOWED_TABLES) ||
                !preg_match('/^[A-Za-z0-9_]+$/', $field) ||
                !preg_match('/^[A-Za-z0-9_]+$/', $pk)) {
                $errors[] = "Invalid identifiers.";
            } else {
                $val = validate_input_value($value);
                $pkv = validate_input_value($pkval);
                if ($val === false || $pkv === false) $errors[] = "Invalid
values.";
                else {
                    try {
                        $sql = "UPDATE $t SET $field = :val WHERE $pk =
:pkval";
                        runQuery($conn, $sql, [':val' => $val, ':pkval' =>
$pkv]);
                        $messages[] = "Record updated in " . h($t);
                    } catch (Exception $e) {
                        $errors[] = $e->getMessage();
                    }
                }
            }
        }

        if ($act === 'delete') {
            $t = trim($_POST['table'] ?? '');
            $pk = trim($_POST['pk'] ?? '');
            $val = $_POST['val'] ?? '';

            if (!validate_identifier($t, $ALLOWED_TABLES) ||
!preg_match('/^[A-Za-z0-9_]+$/', $pk)) {
                $errors[] = "Invalid table or primary key name.";
            } else {
                $v = validate_input_value($val);
                if ($v === false) $errors[] = "Invalid value.";
                else {
                    try {
                        $sql = "DELETE FROM $t WHERE $pk = :val";
                        runQuery($conn, $sql, [':val' => $v]);
                        $messages[] = "Record deleted from " . h($t);
                    } catch (Exception $e) {
                        $errors[] = $e->getMessage();
                    }
                }
            }
        }
    }
}
?>

<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Ride & Pickup DBMS - CPS510</title>
<meta name="viewport" content="width=device-width,initial-scale=1">
<style>
:root{
  --bg:#f5f8ff; --card:#ffffff; --accent:#3b82f6; --accent-2:#06b6d4;
--muted:#53627a;
  --success:#16a34a; --danger:#ef4444;
}
*{box-sizing:border-box}
body{margin:0;font-family:Inter,Arial,Helvetica,sans-serif;background:line
ar-gradient(135deg,#e6eefc 0%,#f7fbff 100%);color:#0f172a;padding:22px}
```

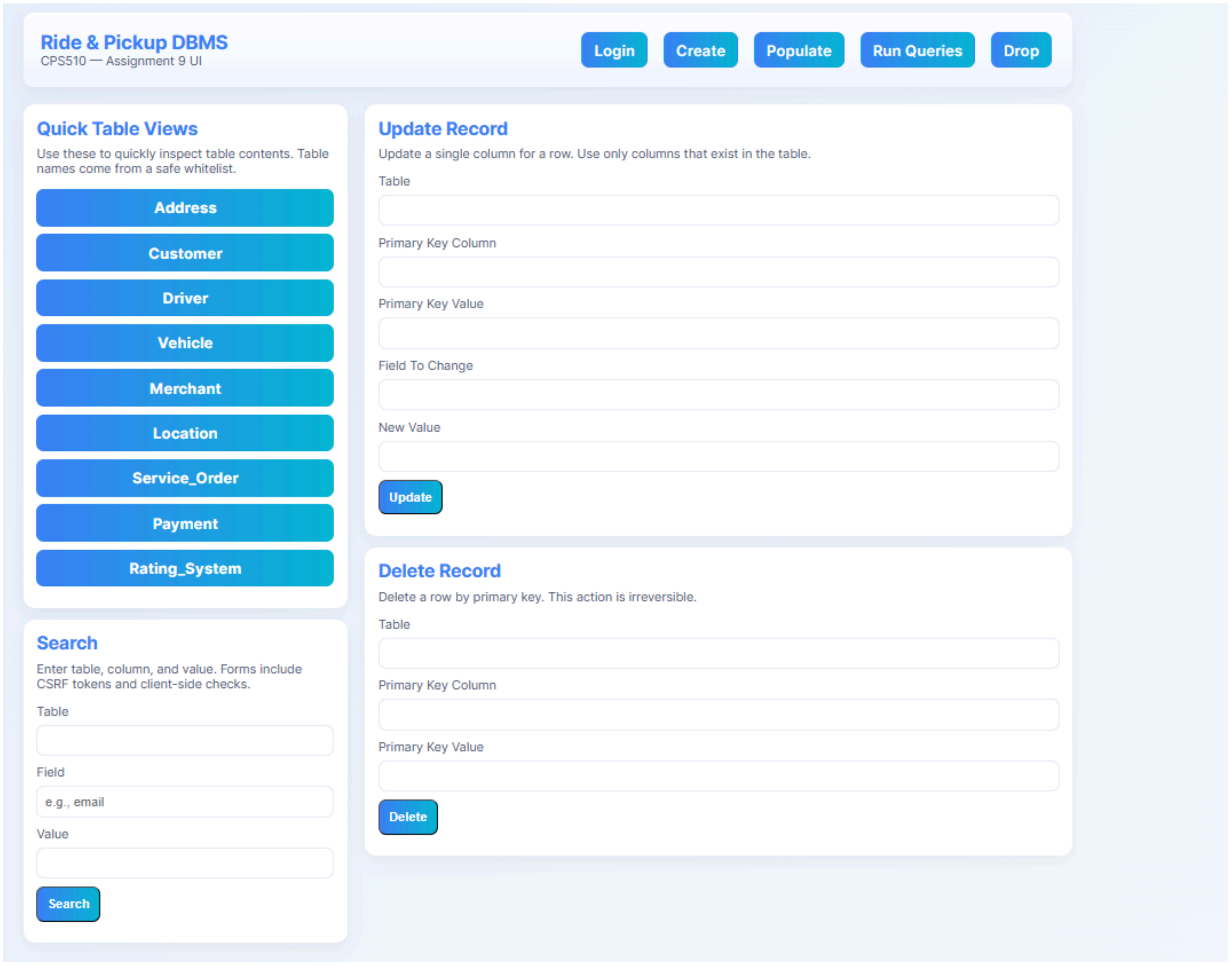
```
.container{max-width:1100px;margin:0 auto}
.header{display:flex;justify-content:space-between;align-items:center;padd
ing:14px
18px;border-radius:10px;background:linear-gradient(180deg,#ffffff,#f1f7ff)
;box-shadow:0 6px 18px rgba(15,23,42,0.06)}
.title{font-size:20px;color:var(--accent);margin:0}
.subtitle{font-size:13px;color:var(--muted)}
.top-actions
a{background:linear-gradient(90deg,var(--accent),var(--accent-2));color:#f
ff;padding:8px
12px;border-radius:8px;text-decoration:none;font-weight:600;margin-left:8p
x}
.layout{display:flex;gap:18px;margin-top:18px;align-items:flex-start}
.left{width:340px;flex-shrink:0}
.card{background:var(--card);padding:14px;border-radius:12px;box-shadow:0
6px 18px rgba(15,23,42,0.06)}
.btn{display:block;text-decoration:none;color:#fff;background:linear-gradi
ent(90deg,var(--accent),var(--accent-2));padding:10px;border-radius:8px;te
xt-align:center;margin:8px 0;font-weight:700}
.small-btn{display:inline-block;padding:8px
10px;border-radius:8px;background:#eef7ff;color:var(--accent);text-decorat
ion:none;border:1px solid rgba(59,130,246,0.12);margin:6px 4px}
.form-input{width:100%;padding:8px;border-radius:8px;border:1px solid
#e6eef6;margin-top:6px}
.label{font-size:13px;color:var(--muted);margin-top:10px;display:block}
.result{width:100%;border-collapse:collapse;margin-top:12px}
.result th, .result td{padding:8px;border:1px solid
#f0f4f8;text-align:left;font-size:14px}
.note{font-size:13px;color:var(--muted);margin-top:8px}
.msg{padding:10px;border-radius:8px;margin-bottom:10px;border-left:4px
solid var(--success);background:#f0fdf4;color:var(--success)}
.err{padding:10px;border-radius:8px;margin-bottom:10px;border-left:4px
solid var(--danger);background:#fff5f5;color:var(--danger)}
.footer{margin-top:18px;text-align:center;color:var(--muted);font-size:13p
x}
@media(max-width:980px){.layout{flex-direction:column}.left{width:100%}}
</style>
</head>
<body>
<div class="container">
```

```
  <div class="header">
    <div>
      <h1 class="title">Ride & Pickup DBMS</h1>
      <div class="subtitle">CPS510 — Assignment 9 UI</div>
    </div>
    <div class="top-actions">
      <a href="login.php" class="small-btn">Login</a>
      <!-- CSRF-protected links for sensitive SQL file actions (token
appended) -->
      <a href="?action=create&token=<?= h($CSRF) ?>"
class="small-btn">Create</a>
      <a href="?action=populate&token=<?= h($CSRF) ?>"
class="small-btn">Populate</a>
      <a href="?action=queries&token=<?= h($CSRF) ?>"
class="small-btn">Run Queries</a>
      <a href="?action=drop&token=<?= h($CSRF) ?>" class="small-btn"
onclick="return confirm('Drop tables? This is irreversible.');">Drop</a>
    </div>
  </div>

  <div class="layout">
    <div class="left">
      <div class="card">
        <h3 style="margin:0;color:var(--accent)">Quick Table Views</h3>
        <p class="note">Use these to quickly inspect table contents. Table
names come from a safe whitelist.</p>
        <?php foreach ($ALLOWED_TABLES as $tbl): ?>
          <a class="btn" href="?action=view&table=<?= h($tbl) ?>"><?=
h($tbl) ?></a>
        <?php endforeach; ?>
      </div>

      <div class="card" style="margin-top:12px">
        <h3 style="margin:0;color:var(--accent)">Search</h3>
        <p class="note">Enter table, column, and value. Forms include CSRF
tokens and client-side checks.</p>
        <form method="post" onsubmit="return validateSearch(this);"
style="margin-top:8px">
          <input type="hidden" name="action" value="search">
          <input type="hidden" name="csrf_token" value="<?= h($CSRF) ?>">
          <label class="label">Table</label>
          <input name="table" class="form-input" list="tables" required>
          <datalist id="tables">
            <?php foreach ($ALLOWED_TABLES as $t): ?><option value="<?=
h($t) ?>"></option><?php endforeach; ?>
          </datalist>
          <label class="label">Field</label>
          <input name="field" class="form-input" placeholder="e.g., email"
required>
          <label class="label">Value</label>
          <input name="value" class="form-input" required>
          <button class="btn" type="submit">Search</button>
        </form>
      </div>

    </div>

    <div class="right" style="flex:1">
      <?php foreach ($messages as $m): ?><div class="msg"><?= h($m)
?></div><?php endforeach; ?>
      <?php foreach ($errors as $e): ?><div class="err"><?= h($e)
?></div><?php endforeach; ?>

      <div class="card">
        <h3 style="margin:0;color:var(--accent)">Update Record</h3>
        <p class="note">Update a single column for a row. Use only columns
that exist in the table.</p>
        <form method="post" onsubmit="return validateUpdate(this);"
style="margin-top:8px">
          <input type="hidden" name="action" value="update">
          <input type="hidden" name="csrf_token" value="<?= h($CSRF) ?>">
          <label class="label">Table</label><input name="table"
class="form-input" list="tables" required>
          <label class="label">Primary Key Column</label><input name="pk"
class="form-input" required>
          <label class="label">Primary Key Value</label><input
name="pkval" class="form-input" required>
          <label class="label">Field To Change</label><input name="field"
class="form-input" required>
```

```
          <label class="label">New Value</label><input name="value"
class="form-input" required>
          <button class="btn" type="submit">Update</button>
        </form>
      </div>

      <div class="card" style="margin-top:12px">
        <h3 style="margin:0;color:var(--accent)">Delete Record</h3>
        <p class="note">Delete a row by primary key. This action is
irreversible.</p>
        <form method="post" onsubmit="return confirm('Delete record? This
cannot be undone.') && validateDelete(this);" style="margin-top:8px">
          <input type="hidden" name="action" value="delete">
          <input type="hidden" name="csrf_token" value="<?= h($CSRF) ?>">
          <label class="label">Table</label><input name="table"
class="form-input" list="tables" required>
          <label class="label">Primary Key Column</label><input name="pk"
class="form-input" required>
          <label class="label">Primary Key Value</label><input name="val"
class="form-input" required>
          <button class="btn" type="submit">Delete</button>
        </form>
      </div>

      <!-- If any table output was printed above (from actions), it will
show here because printTable echoes HTML -->
    </div>
  </div>

  <div class="footer">
    <div>a</div>
  </div>
</div>

<script>
// Client-side validations for UX only (server checks remain
authoritative)
function identOK(s) { return /^[A-Za-z0-9_]+$/.test(s); }

function validateSearch(f) {
  const table = f.table.value.trim(), field = f.field.value.trim();
  if (!identOK(table)) { alert('Invalid table name. Use alphanumeric and
underscores only.'); return false; }
  if (!identOK(field)) { alert('Invalid field name. Use alphanumeric and
underscores only.'); return false; }
  if (f.value.value.length > 2000) { alert('Value too long'); return
false; }
  return true;
}

function validateUpdate(f) {
  if (!identOK(f.table.value.trim()) || !identOK(f.pk.value.trim()) ||
!identOK(f.field.value.trim())) {
    alert('Table, PK column and field must be alphanumeric/underscore
only.'); return false;
  }
  return true;
}

function validateDelete(f) {
  if (!identOK(f.table.value.trim()) || !identOK(f.pk.value.trim())) {
    alert('Invalid identifiers.'); return false;
  }
  return true;
}
</script>
</body>
</html>
```

Screenshots:

Home Screen



Ride & Pickup DBMS
CPS510 — Assignment 9 UI

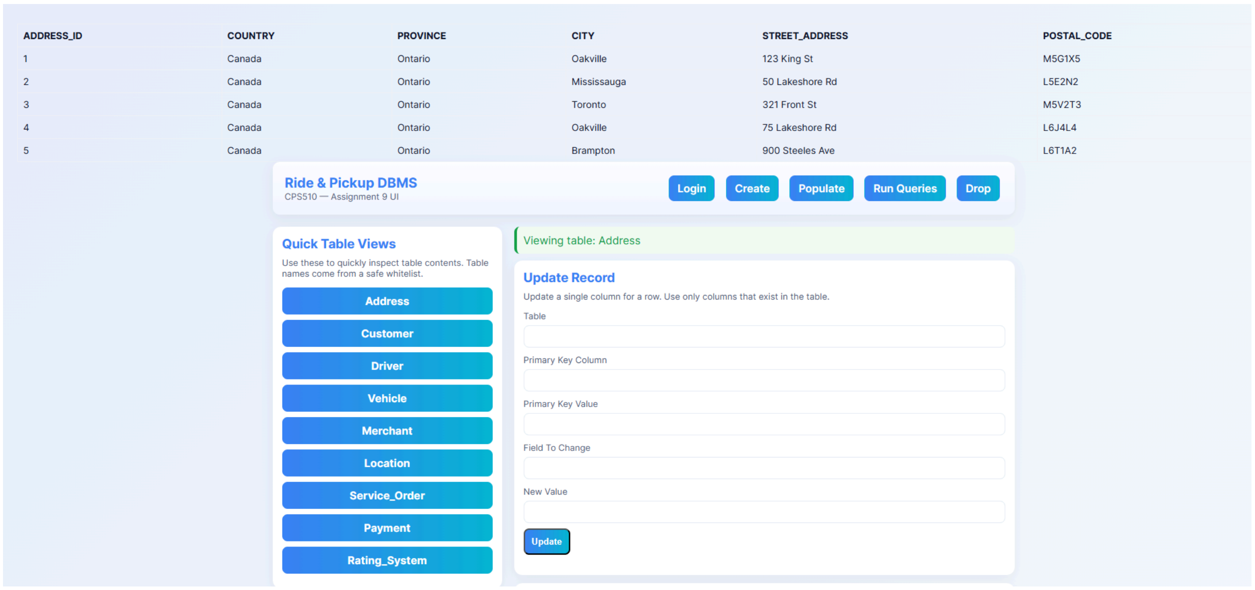
Quick Table Views
Use these to quickly inspect table contents. Table names come from a safe whitelist.

Search
Enter table, column, and value. Forms include CSRF tokens and client-side checks.
Table
Field
Value

Update Record
Update a single column for a row. Use only columns that exist in the table.
Table
Primary Key Column
Primary Key Value
Field To Change
New Value

Delete Record
Delete a row by primary key. This action is irreversible.
Table
Primary Key Column
Primary Key Value

Viewing Table:



ADDRESS_ID	COUNTRY	PROVINCE	CITY	STREET_ADDRESS	POSTAL_CODE
1	Canada	Ontario	Oakville	123 King St	M5G1X5
2	Canada	Ontario	Mississauga	50 Lakeshore Rd	L5E2N2
3	Canada	Ontario	Toronto	321 Front St	M5V2T3
4	Canada	Ontario	Oakville	75 Lakeshore Rd	L6J4L4
5	Canada	Ontario	Brampton	900 Steeles Ave	L6T1A2

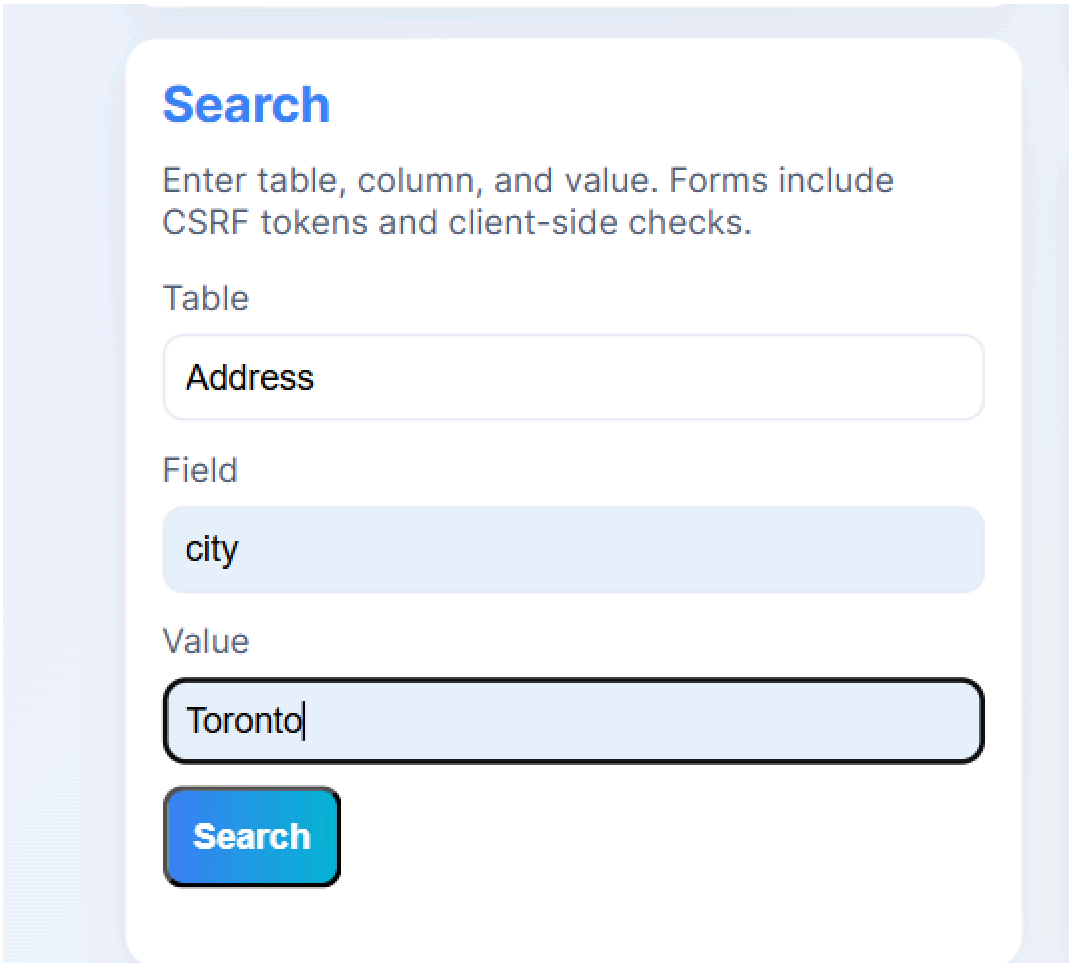
Ride & Pickup DBMS
CPS510 — Assignment 9 UI

Quick Table Views
Use these to quickly inspect table contents. Table names come from a safe whitelist.

Viewing table: Address

Update Record
Update a single column for a row. Use only columns that exist in the table.
Table
Primary Key Column
Primary Key Value
Field To Change
New Value

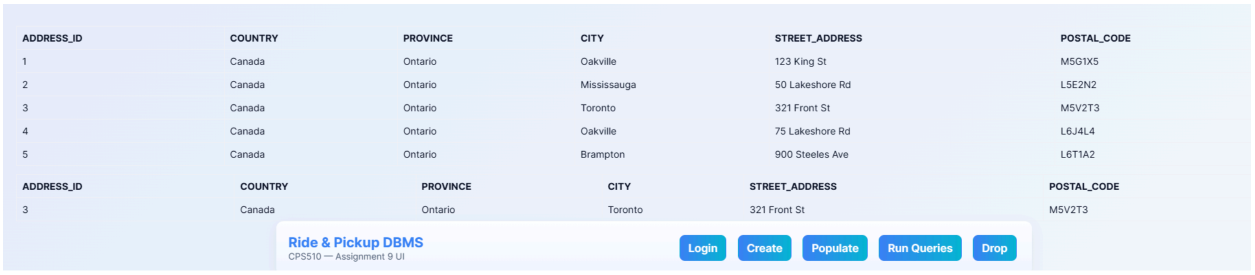
Search function:



Search

Enter table, column, and value. Forms include CSRF tokens and client-side checks.

Table	Address
Field	city
Value	Toronto

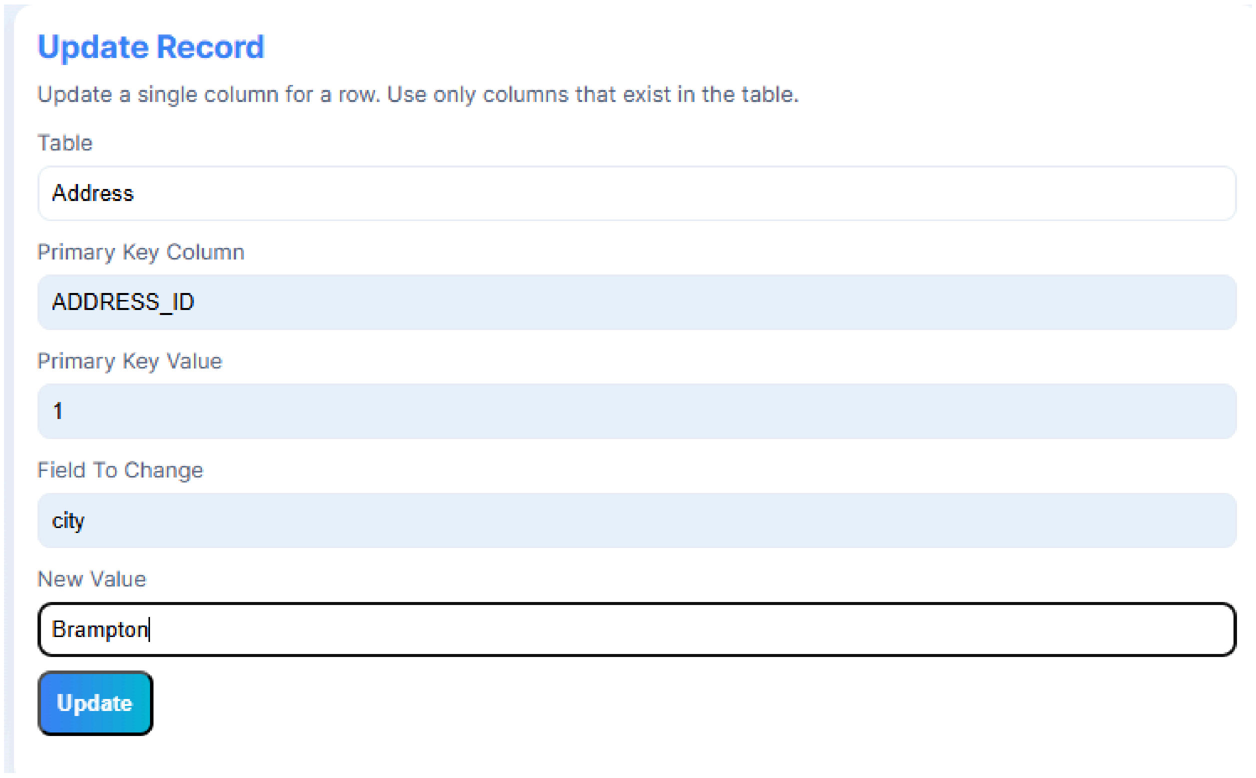


ADDRESS_ID	COUNTRY	PROVINCE	CITY	STREET_ADDRESS	POSTAL_CODE
1	Canada	Ontario	Oakville	123 King St	M5G1X5
2	Canada	Ontario	Mississauga	50 Lakeshore Rd	L5E2N2
3	Canada	Ontario	Toronto	321 Front St	M5V2T3
4	Canada	Ontario	Oakville	75 Lakeshore Rd	L6J4L4
5	Canada	Ontario	Brampton	900 Steeles Ave	L6T1A2

ADDRESS_ID	COUNTRY	PROVINCE	CITY	STREET_ADDRESS	POSTAL_CODE
3	Canada	Ontario	Toronto	321 Front St	M5V2T3

Ride & Pickup DBMS
CPS510 — Assignment 9 UI

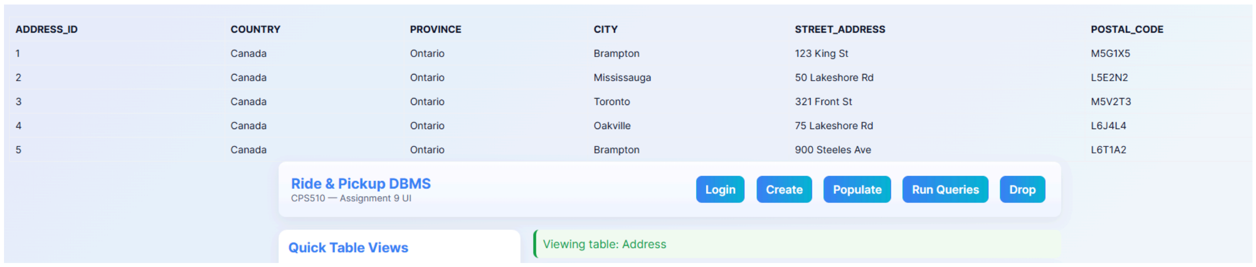
Update function:



Update Record

Update a single column for a row. Use only columns that exist in the table.

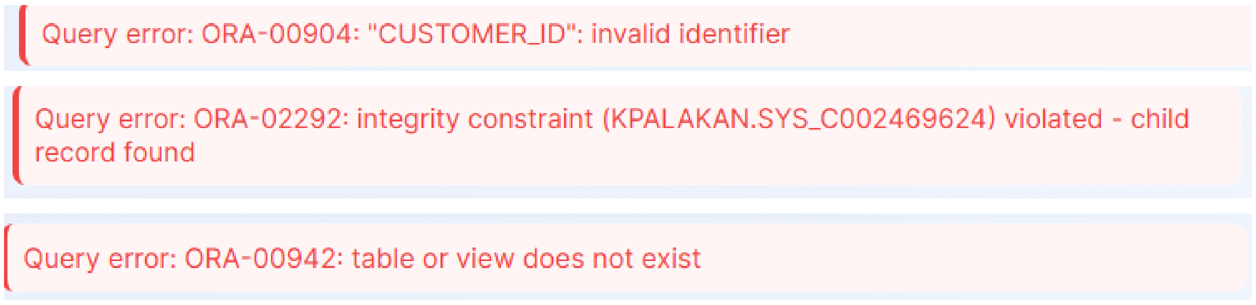
Table	Address
Primary Key Column	ADDRESS_ID
Primary Key Value	1
Field To Change	city
New Value	Brampton



ADDRESS_ID	COUNTRY	PROVINCE	CITY	STREET_ADDRESS	POSTAL_CODE
1	Canada	Ontario	Brampton	123 King St	M5G1X5
2	Canada	Ontario	Mississauga	50 Lakeshore Rd	L5E2N2
3	Canada	Ontario	Toronto	321 Front St	M5V2T3
4	Canada	Ontario	Oakville	75 Lakeshore Rd	L6J4L4
5	Canada	Ontario	Brampton	900 Steeles Ave	L6T1A2

Viewing table: Address

Example of error handling:

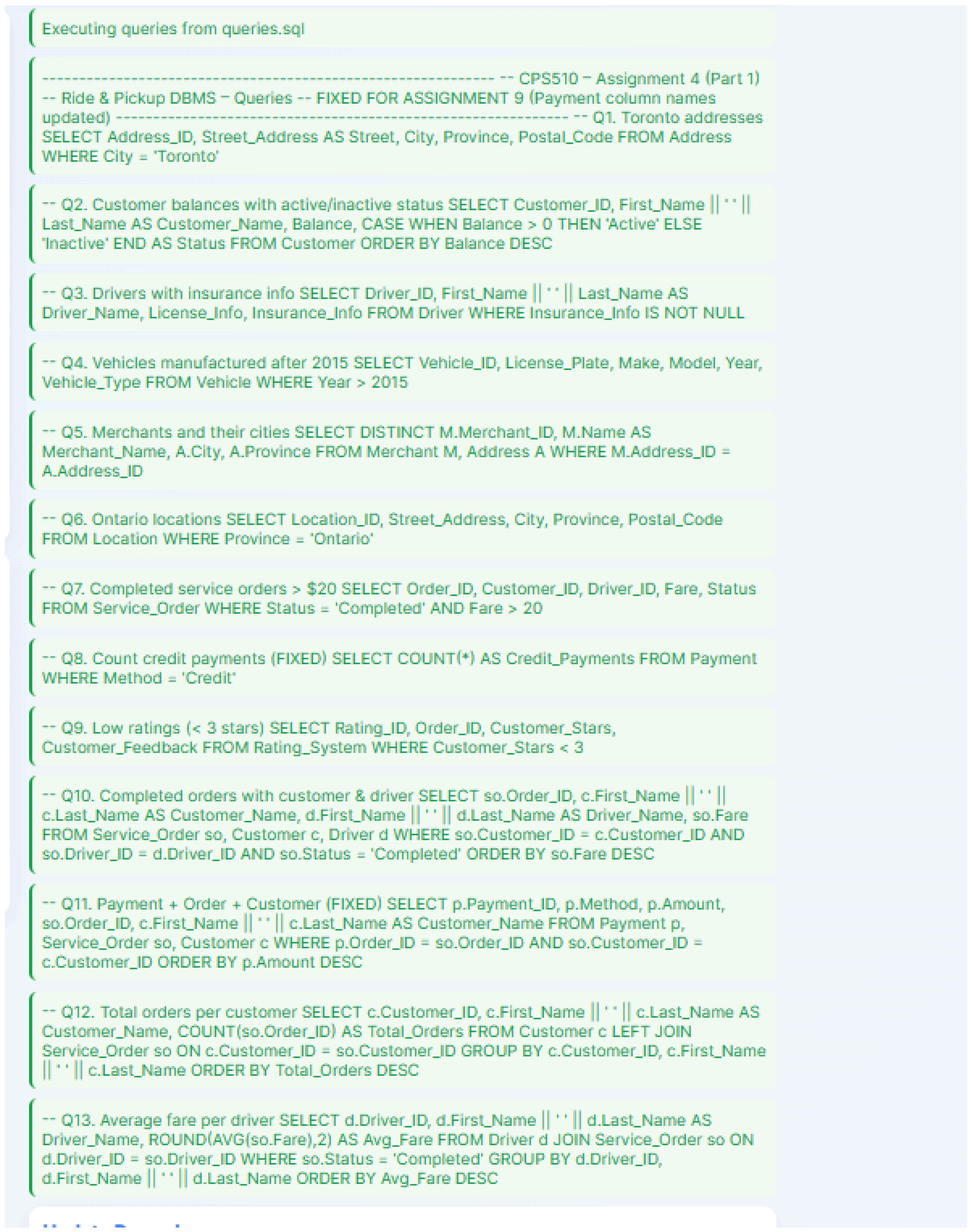


Query error: ORA-00904: "CUSTOMER_ID": invalid identifier

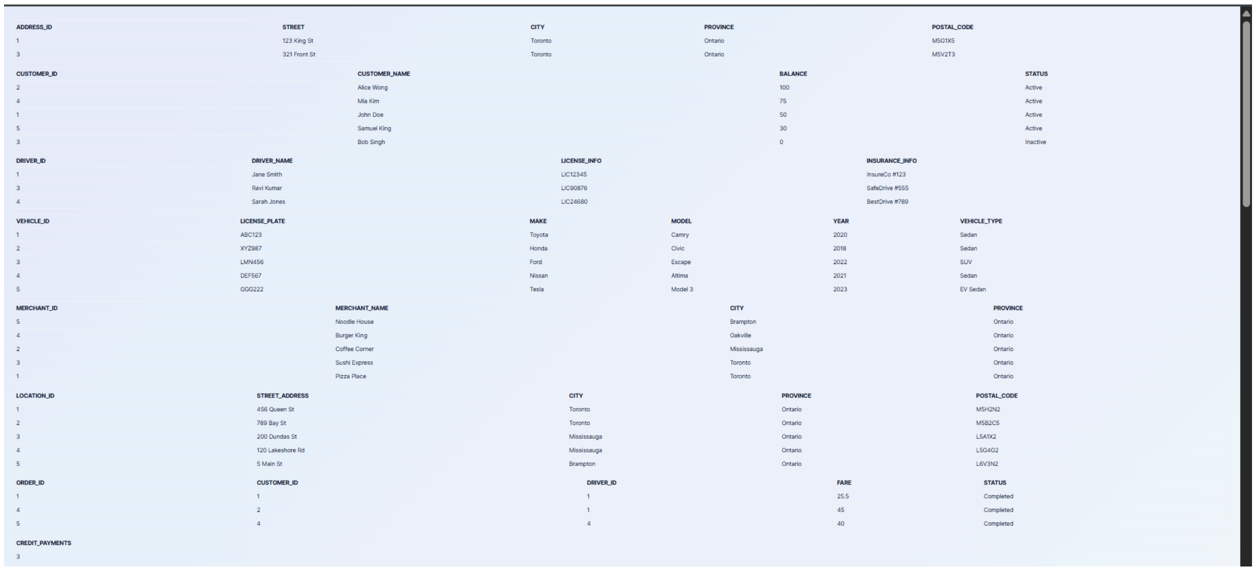
Query error: ORA-02292: integrity constraint (KPALAKAN.SYS_C002469624) violated - child record found

Query error: ORA-00942: table or view does not exist

Advanced Queries



```
Executing queries from queries.sql

------------------------------------------------------------ -- CPS510 – Assignment 4 (Part 1)
-- Ride & Pickup DBMS – Queries -- FIXED FOR ASSIGNMENT 9 (Payment column names
updated) ------------------------------------------------------------ -- Q1. Toronto addresses
SELECT Address_ID, Street_Address AS Street, City, Province, Postal_Code FROM Address
WHERE City = 'Toronto'

-- Q2. Customer balances with active/inactive status SELECT Customer_ID, First_Name || ' ' ||
Last_Name AS Customer_Name, Balance, CASE WHEN Balance > 0 THEN 'Active' ELSE
'Inactive' END AS Status FROM Customer ORDER BY Balance DESC

-- Q3. Drivers with insurance info SELECT Driver_ID, First_Name || ' ' || Last_Name AS
Driver_Name, License_Info, Insurance_Info FROM Driver WHERE Insurance_Info IS NOT NULL

-- Q4. Vehicles manufactured after 2015 SELECT Vehicle_ID, License_Plate, Make, Model, Year,
Vehicle_Type FROM Vehicle WHERE Year > 2015

-- Q5. Merchants and their cities SELECT DISTINCT M.Merchant_ID, M.Name AS
Merchant_Name, A.City, A.Province FROM Merchant M, Address A WHERE M.Address_ID =
A.Address_ID

-- Q6. Ontario locations SELECT Location_ID, Street_Address, City, Province, Postal_Code
FROM Location WHERE Province = 'Ontario'

-- Q7. Completed service orders > $20 SELECT Order_ID, Customer_ID, Driver_ID, Fare, Status
FROM Service_Order WHERE Status = 'Completed' AND Fare > 20

-- Q8. Count credit payments (FIXED) SELECT COUNT(*) AS Credit_Payments FROM Payment
WHERE Method = 'Credit'

-- Q9. Low ratings (< 3 stars) SELECT Rating_ID, Order_ID, Customer_Stars,
Customer_Feedback FROM Rating_System WHERE Customer_Stars < 3

-- Q10. Completed orders with customer & driver SELECT so.Order_ID, c.First_Name || ' ' ||
c.Last_Name AS Customer_Name, d.First_Name || ' ' || d.Last_Name AS Driver_Name, so.Fare
FROM Service_Order so, Customer c, Driver d WHERE so.Customer_ID = c.Customer_ID AND
so.Driver_ID = d.Driver_ID AND so.Status = 'Completed' ORDER BY so.Fare DESC

-- Q11. Payment + Order + Customer (FIXED) SELECT p.Payment_ID, p.Method, p.Amount,
so.Order_ID, c.First_Name || ' ' || c.Last_Name AS Customer_Name FROM Payment p,
Service_Order so, Customer c WHERE p.Order_ID = so.Order_ID AND so.Customer_ID =
c.Customer_ID ORDER BY p.Amount DESC

-- Q12. Total orders per customer SELECT c.Customer_ID, c.First_Name || ' ' || c.Last_Name AS
Customer_Name, COUNT(so.Order_ID) AS Total_Orders FROM Customer c LEFT JOIN
Service_Order so ON c.Customer_ID = so.Customer_ID GROUP BY c.Customer_ID, c.First_Name
|| ' ' || c.Last_Name ORDER BY Total_Orders DESC

-- Q13. Average fare per driver SELECT d.Driver_ID, d.First_Name || ' ' || d.Last_Name AS
Driver_Name, ROUND(AVG(so.Fare),2) AS Avg_Fare FROM Driver d JOIN Service_Order so ON
d.Driver_ID = so.Driver_ID WHERE so.Status = 'Completed' GROUP BY d.Driver_ID,
d.First_Name || ' ' || d.Last_Name ORDER BY Avg_Fare DESC
```



ADDRESS_ID	STREET	CITY	PROVINCE	POSTAL_CODE
1	123 King St	Toronto	Ontario	M5G1X5
3	321 Front St	Toronto	Ontario	M5V2T3

CUSTOMER_ID	CUSTOMER_NAME	BALANCE	STATUS
2	Alice Wong	100	Active
4	Mia Kim	75	Active
1	John Doe	50	Active
5	Samuel King	30	Active
3	Bob Singh	0	Inactive

DRIVER_ID	DRIVER_NAME	LICENSE_INFO	INSURANCE_INFO
1	Jane Smith	LIC12345	InsureCo #123
3	Ravi Kumar	LIC90876	SafeDrive #555
4	Sarah Jones	LIC24680	BestDrive #789

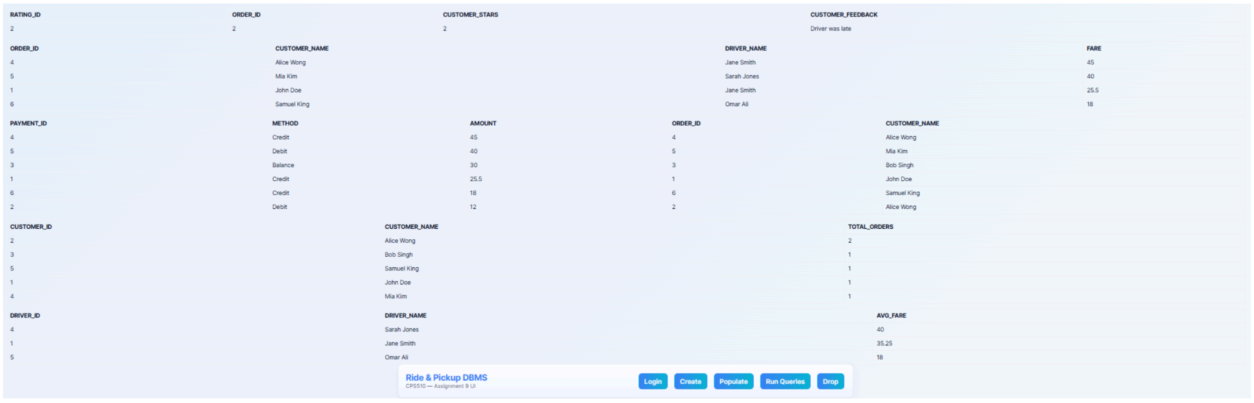
VEHICLE_ID	LICENSE_PLATE	MAKE	MODEL	YEAR	VEHICLE_TYPE
1	ABC123	Toyota	Camry	2020	Sedan
2	XYZ987	Honda	Civic	2018	Sedan
3	LMN456	Ford	Escape	2022	SUV
4	DEF567	Nissan	Altima	2021	Sedan
5	GGG222	Tesla	Model 3	2023	EV Sedan

MERCHANT_ID	MERCHANT_NAME	CITY	PROVINCE
5	Noodle House	Brampton	Ontario
4	Burger King	Oakville	Ontario
2	Coffee Corner	Mississauga	Ontario
3	Sushi Express	Toronto	Ontario
1	Pizza Place	Toronto	Ontario

LOCATION_ID	STREET_ADDRESS	CITY	PROVINCE	POSTAL_CODE
1	456 Queen St	Toronto	Ontario	M5H2N2
2	789 Bay St	Toronto	Ontario	M5B2C5
3	200 Dundas St	Mississauga	Ontario	L5A1X2
4	120 Lakeshore Rd	Mississauga	Ontario	L5G4G2
5	5 Main St	Brampton	Ontario	L6V3N2

ORDER_ID	CUSTOMER_ID	DRIVER_ID	FARE	STATUS
1	1	1	25.5	Completed
4	2	1	45	Completed
5	4	4	40	Completed

CREDIT_PAYMENTS
3



RATING_ID	ORDER_ID	CUSTOMER_STARS	CUSTOMER_FEEDBACK
2	2	2	Driver was late

ORDER_ID	CUSTOMER_NAME	DRIVER_NAME	FARE
4	Alice Wong	Jane Smith	45
5	Mia Kim	Sarah Jones	40
1	John Doe	Jane Smith	25.5
6	Samuel King	Omar Ali	18

PAYMENT_ID	METHOD	AMOUNT	ORDER_ID	CUSTOMER_NAME
4	Credit	45	4	Alice Wong
5	Debit	40	5	Mia Kim
3	Balance	30	3	Bob Singh
1	Credit	25.5	1	John Doe
6	Credit	18	6	Samuel King
2	Debit	12	2	Alice Wong

CUSTOMER_ID	CUSTOMER_NAME	TOTAL_ORDERS
2	Alice Wong	2
3	Bob Singh	1
5	Samuel King	1
1	John Doe	1
4	Mia Kim	1

DRIVER_ID	DRIVER_NAME	AVG_FARE
4	Sarah Jones	40
1	Jane Smith	35.25
5	Omar Ali	18