

Phase II of Formula 1 Season 2023 Database Application Project

1. Trigger Implementations

The first trigger, named as `prevent_negative_duration`, automates the check of a negative number entrance for pit stop's duration during an insertion to `Pit_Stop` table.

```
DELIMITER //
CREATE TRIGGER prevent_negative_duration
BEFORE INSERT ON Pit_Stop
FOR EACH ROW
BEGIN
    -- Check if the duration is negative
    IF NEW.duration < 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Duration cannot be negative';
    END IF;
END //
DELIMITER ;
```

Here is the initial state of the `Pit_Stop` table:

	pid	reason	duration
▶	1	Tire change	0.18
	2	Refueling	0.16
	3	Repairing Damage	0.17
	4	Adjusting car setup	0.11
	5	Tire pressure adjustment	0.2
	6	Brake check	0.26
	7	Clearing debris	0.33
	8	Penalty	0.1
	9	Cooling System Error	0.14
	10	Windshield Cleaning	0.19

With the following query, the trigger is being tested and we expect to see an error message:

```
INSERT INTO Pit_Stop(pid,reason,duration) VALUES(11,'malfunction',-5);
```

After we run this code piece, we get the error message as we expected:

42 03:05:29 INSERT INTO Pit_Stop(pid,reason,duration) VALUES(11,'malfunction',-5) Error Code: 1644, Duration cannot be negative 0.000 sec

When we give a positive number for duration with the following query, we expect a successful insertion:

```
INSERT INTO Pit_Stop(pid,reason,duration) VALUES(11,'malfunction',5);
```

As we expected, insertion was completed successfully and here is the final state of the Pit_Stop table:

	pid	reason	duration
▶	1	Tire change	0.18
	2	Refueling	0.16
	3	Reparing Damage	0.17
	4	Adjusting car setup	0.11
	5	Tire pressure adjusment	0.2
	6	Brake check	0.26
	7	Clearing debris	0.33
	8	Penalty	0.1
	9	Cooling System Error	0.14
	10	Windshield Cleaning	0.19
	11	malfunction	5

The second trigger, named as restrict_supplier_country, automates the check that the given country during insertion into Engine_Supplier table for is a valid country.

```
DELIMITER //
CREATE TRIGGER restrict_supplier_country
BEFORE INSERT ON Engine_Supplier
FOR EACH ROW
BEGIN
    -- Check if the country is not in the allowed list
    IF NEW.country NOT IN ('Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola', 'Antigua and Barbuda', 'Argentina', 'Armenia', 'Australia', 'Austria',
'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bhutan', 'Bolivia', 'Bosnia and Herzegovina', 'Botswana',
'Brazil', 'Brunei', 'Bulgaria', 'Burkina Faso', 'Burundi', 'Cabo Verde', 'Cambodia', 'Cameroon', 'Canada', 'Central African Republic', 'Chad', 'Chile', 'China',
'Colombia', 'Comoros', 'Congo (Congo-Brazzaville)', 'Costa Rica', 'Croatia', 'Cuba', 'Cyprus', 'Czechia (Czech Republic)', 'Denmark', 'Djibouti', 'Dominica',
'Dominican Republic', 'Ecuador', 'Egypt', 'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Estonia', 'Eswatini (fmr. "Swaziland")', 'Ethiopia', 'Fiji', 'Finland',
'France', 'Gabon', 'Gambia', 'Georgia', 'Germany', 'Ghana', 'Greece', 'Grenada', 'Guatemala', 'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti', 'Holy See', 'Honduras',
'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy', 'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati',
'Korea (North)', 'Korea (South)', 'Kuwait', 'Kyrgyzstan', 'Laos', 'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Liechtenstein', 'Lithuania', 'Luxembourg',
'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta', 'Marshall Islands', 'Mauritania', 'Mauritius', 'Mexico', 'Micronesia', 'Moldova', 'Monaco',
'Mongolia', 'Montenegro', 'Morocco', 'Mozambique', 'Myanmar (formerly Burma)', 'Namibia', 'Nauru', 'Nepal', 'Netherlands', 'New Zealand', 'Nicaragua', 'Niger',
'Nigeria', 'North Macedonia (formerly Macedonia)', 'Norway', 'Oman', 'Pakistan', 'Palau', 'Palestine State', 'Panama', 'Papua New Guinea', 'Paraguay', 'Peru',
'Philippines', 'Poland', 'Portugal', 'Qatar', 'Romania', 'Russia', 'Rwanda', 'Saint Kitts and Nevis', 'Saint Lucia', 'Saint Vincent and the Grenadines', 'Samoa',
'San Marino', 'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia', 'Seychelles', 'Sierra Leone', 'Singapore', 'Slovakia', 'Slovenia',
'Solomon Islands', 'Somalia', 'South Africa', 'South Sudan', 'Spain', 'Sri Lanka', 'Sudan', 'Suriname', 'Sweden', 'Switzerland',
'Syria', 'Tajikistan', 'Tanzania', 'Thailand', 'Timor-Leste', 'Togo', 'Tonga', 'Trinidad and Tobago', 'Tunisia', 'Turkey',
'Turkmenistan', 'Tuvalu', 'Uganda', 'Ukraine', 'United Arab Emirates', 'United Kingdom', 'United States of America', 'Uruguay',
'Uzbekistan', 'Vanuatu', 'Venezuela', 'Vietnam', 'Yemen', 'Zambia', 'Zimbabwe')
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Invalid country for engine supplier.';
    END IF;
END //
DELIMITER ;
```

This is the state of Engine_Supplier table initially:

	esid	ename	country
▶	1	Ferrari	Italy
	2	Mercedes	Germany
	3	Honda	Japan
	4	Renault	France
	5	Ford	France
	6	Audi	German
	7	Alpine	France
	8	McLaren	Britain
	9	Cosworth	Britain
	10	BMW	Germany

We're testing the trigger with following query, by entering an invalid country name and we expect to get the error message:

```
INSERT INTO Engine_Supplier(esid,ename,country) VALUES(11,'Toyota','Tokyo');
```

As expected, the insertion was failed, and we got our error message:

```
48 03:21:51 INSERT INTO Engine_Supplier(esid,ename,country) VALUES(11,'Toyota','Tokyo') Error Code: 1644. Invalid country for engine supplier. 0.000 sec
```

If we try to make the insertion with a valid country name with following query, we expect a successful attempt:

```
INSERT INTO Engine_Supplier(esid,ename,country) VALUES(11,'Toyota','Japan');
```

The attempt became successful, and the final state of table is:

	esid	ename	country
▶	1	Ferrari	Italy
	2	Mercedes	Germany
	3	Honda	Japan
	4	Renault	France
	5	Ford	France
	6	Audi	German
	7	Alpine	France
	8	McLaren	Britain
	9	Cosworth	Britain
	10	BMW	Germany
	11	Toyota	Japan

2. Stored Procedure Implementations

The first stored procedure, named as addEngineSupplier, takes the relevant values as parameters and inserts the given values into Engine_Supplier table. After the insertion, it gives a message that the insertion is completed successfully.

```
DELIMITER //
```

```
CREATE PROCEDURE addEngineSupplier(  
    IN new_esid INT,  
    IN new_ename CHAR(50),  
    IN new_country CHAR(50)  
)  
BEGIN  
    -- Insert new engine supplier into Engine_Supplier table  
    INSERT INTO Engine_Supplier (esid, ename, country)  
    VALUES (new_esid, new_ename, new_country);  
  
    -- Provide a success message  
    SELECT 'Engine Supplier added successfully' AS Message;  
  
END //
```

```
DELIMITER ;
```

We can call the procedure with following query to make an insertion into Engine_Supplier table:

```
CALL addEngineSupplier(20,'Opel','Germany');
```

After the successful insertion, it gives the following message:

	Message
►	Engine Supplier added successfully

The initial and final state of Engine_Supplier table are:

	esid	ename	country
►	1	Ferrari	Italy
	2	Mercedes	Germany
	3	Honda	Japan
	4	Renault	France
	5	Ford	France
	6	Audi	German
	7	Alpine	France
	8	McLaren	Britain
	9	Cosworth	Britain
	10	BMW	Germany

	esid	ename	country
►	1	Ferrari	Italy
	2	Mercedes	Germany
	3	Honda	Japan
	4	Renault	France
	5	Ford	France
	6	Audi	German
	7	Alpine	France
	8	McLaren	Britain
	9	Cosworth	Britain
	10	BMW	Germany
	20	Opel	Germany

The second stored procedure, named as addPitStop, takes the new information and inserts those into Pit_Stop table. Finally, it gives a success message.

```
DELIMITER //
```

⊖

```
CREATE PROCEDURE addPitStop(  
    IN new_pid INT,  
    IN new_reason CHAR(100),  
    IN new_duration REAL  
)  
BEGIN  
    -- Insert new pit stop into Pit_Stop table  
    INSERT INTO Pit_Stop (pid, reason, duration)  
    VALUES (new_pid, new_reason, new_duration);  
  
    -- Provide a success message  
    SELECT 'Pit Stop added successfully' AS Message;  
END //
```

```
DELIMITER ;
```

This procedure can be called with the following query:

```
CALL addPitStop(30,'health issue',1.7);
```

After the successful completion of insertion, we get the message:

	Message
►	Pit Stop added successfully

The state of Pit_Stop table at the beginning and end of insertion:

	pid	reason	duration		pid	reason	duration
▶	1	Tire change	0.18	▶	1	Tire change	0.18
	2	Refueling	0.16		2	Refueling	0.16
	3	Repairing Damage	0.17		3	Repairing Damage	0.17
	4	Adjusting car setup	0.11		4	Adjusting car setup	0.11
	5	Tire pressure adjustment	0.2		5	Tire pressure adjustment	0.2
	6	Brake check	0.26		6	Brake check	0.26
	7	Clearing debris	0.33		7	Clearing debris	0.33
	8	Penalty	0.1		8	Penalty	0.1
	9	Cooling System Error	0.14		9	Cooling System Error	0.14
	10	Windshield Cleaning	0.19		10	Windshield Cleaning	0.19
					30	health issue	1.7

3. Web Access Module

The first web interface works with the stored procedure addEngineSupplier. To design the webpage, index_engine_supplier.html is written. With this html file, we're creating the webpage in which we will get inputs from the user to insert a new entry into Engine_Supplier table. To be able to execute the insertion, html file is connected to insert_engine_supplier.php as it will be explained now.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Add Engine Supplier</title>
</head>
<body>

  <h2>Insert New Engine Supplier</h2>

  <form action="insert_engine_supplier.php" method="POST">
    <label for="esid">Engine Supplier ID:</label><br>
    <input type="number" id="esid" name="esid" required><br><br>

    <label for="ename">Engine Supplier Name:</label><br>
    <input type="text" id="ename" name="ename" required><br><br>

    <label for="country">Country:</label><br>
    <input type="text" id="country" name="country" required><br><br>

    <input type="submit" value="Submit">
  </form>

</body>
</html>
```

In insert_engine_supplier.php, we first establish the connection of database and check that if it is completed successfully. Using the data which user is entered, we call the stored procedure addEngineSupplier and a message appears on the screen if the insertion is completed without a problem.

```

1  <?php
2  $servername = "localhost";
3  $username = "root";
4  $password = "";
5  $dbname = "mydatabase";
6
7  $conn = new mysqli(hostname: $servername, username: $username, password: $password, database: $dbname);
8
9  if ($conn->connect_error) {
10     die("Connection failed: " . $conn->connect_error);
11 }
12
13 $esid = $_POST['esid'];
14 $ename = $_POST['ename'];
15 $country = $_POST['country'];
16
17 $sql = "CALL addEngineSupplier(?, ?, ?)";
18
19 $stmt = $conn->prepare(query: $sql);
20 $stmt->bind_param(types: "iss", var: &$esid, vars: &$ename, $country);
21
22 if ($stmt->execute()) {
23     echo "<p>Engine Supplier added successfully!</p>";
24 } else {
25     echo "<p>Error: " . $stmt->error . "</p>";
26 }
27
28 $stmt->close();
29 $conn->close();
30 ?>
31

```

To view the table, a separate php file is written, named as view_engine_supplier.php. As above, the connection is established and checked first. Afterwards, we fetched the data we needed from our database with using a SQL query, which are the components of Engine_Supplier table. Finally, we're outputting the results within a table.

```

1  <?php
2  $servername = "localhost";
3  $username = "root";
4  $password = "";
5  $dbname = "mydatabase";
6
7  $conn = new mysqli(hostname: $servername, username: $username, password: $password, database: $dbname);
8
9
10 if ($conn->connect_error) {
11     die("Connection failed: " . $conn->connect_error);
12 }
13
14
15 $sql = "SELECT esid, ename, country FROM Engine_Supplier";
16 $result = $conn->query(query: $sql);
17
18 if ($result->num_rows > 0) {
19     echo "<table border='1'><tr><th>Engine Supplier ID</th><th>Engine Supplier Name</th><th>Country</th></tr>";
20     while($row = $result->fetch_assoc()) {
21         echo "<tr><td>" . $row['esid'] . "</td><td>" . $row['ename'] . "</td><td>" . $row['country'] . "</td></tr>";
22     }
23     echo "</table>";
24 } else {
25     echo "No engine suppliers found.";
26 }
27
28 $conn->close();
29 ?>
30

```

After placing those files into htdocs subfile of xampp file, we tested our web interfaces. By going to http://localhost/index_engine_supplier.html, we can access the interface we designed, and it looks like this:

Insert New Engine Supplier

Engine Supplier ID:

Engine Supplier Name:

Country:

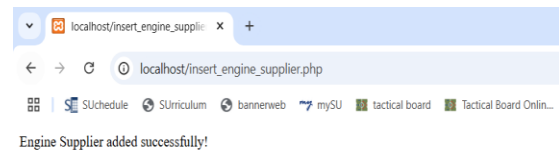
After a successful insertion, a message appears on the page:

Insert New Engine Supplier

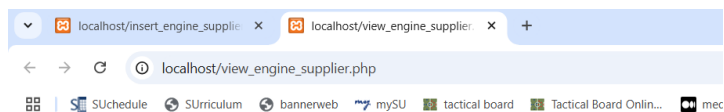
Engine Supplier ID:

Engine Supplier Name:

Country:



To view the status of the Engine_Supplier, the link http://localhost/view_engine_supplier.php can be used:



Engine Supplier ID	Engine Supplier Name	Country
1	Ferrari	Italy
2	Mercedes	Germany
3	Honda	Japan
4	Renault	France
5	Ford	France
6	Audi	German
7	Alpine	France
8	McLaren	Britain
9	Cosworth	Britain
10	BMW	Germany
15	Opel	Germany
20	Opel	Germany

A similar approach is used to design the web interface of addPitStop procedure. The index_pit_stop.html contains the following as its functionality is quite similar to previous one:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Add Pit Stop</title>
7 </head>
8 <body>
9
10   <h2>Insert New Pit Stop</h2>
11
12   <form action="insert_pit_stop.php" method="POST">
13     <label for="pid">Pit Stop ID:</label><br>
14     <input type="number" id="pid" name="pid" required><br><br>
15
16     <label for="reason">Reason:</label><br>
17     <input type="text" id="reason" name="reason" required><br><br>
18
19     <label for="duration">Duration (seconds):</label><br>
20     <input type="number" step="any" id="duration" name="duration" required><br><br>
21
22     <input type="submit" value="Submit">
23   </form>
24
25 </body>
26 </html>
```

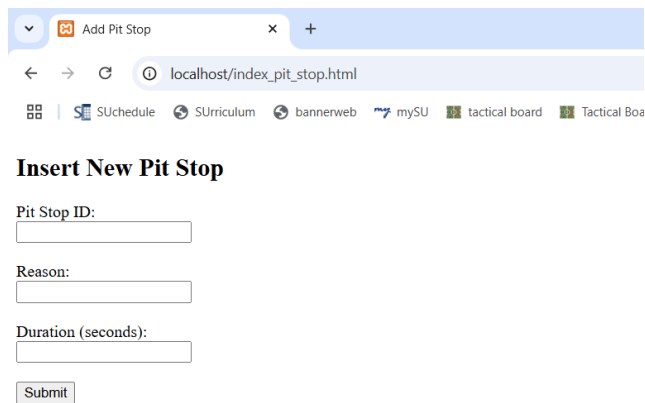

To execute the insertion, we need to call the procedure we have written, and it happens via `insert_pit_stop.php`. Beginning with connection to database, we moved on with calling the procedure:

```
1  <?php
2
3  $servername = "localhost";
4  $username = "root";
5  $password = "";
6  $dbname = "mydatabase";
7
8
9  $conn = new mysqli(hostname: $servername, username: $username, password: $password, database: $dbname);
10
11
12  if ($conn->connect_error) {
13      die("Connection failed: " . $conn->connect_error);
14  }
15
16
17  $pid = $_POST['pid'];
18  $reason = $_POST['reason'];
19  $duration = $_POST['duration'];
20
21
22  $sql = "CALL addPitStop(?, ?, ?)";
23
24  |
25  $stmt = $conn->prepare(query: $sql);
26  $stmt->bind_param(types: "iss", var: &$amp;pid, vars: &$amp;reason, $duration);
27
28  // Execute the stored procedure
29  if ($stmt->execute()) {
30      echo "<p>Pit Stop added successfully!</p>";
31  } else {
32      echo "<p>Error: " . $stmt->error . "</p>";
33  }
34
35  $stmt->close();
36  $conn->close();
37  ?>
```

Again, to view the table on web, `view_pit_stop.php` is implemented. The php file connects to database, selects the relevant data with SQL query and displays the information in a table format.

```
1  <?php
2  // Database connection
3  $servername = "localhost";
4  $username = "root";
5  $password = "";
6  $dbname = "mydatabase";
7
8  $conn = new mysqli(hostname: $servername, username: $username, password: $password, database: $dbname);
9
10
11  if ($conn->connect_error) {
12      die("Connection failed: " . $conn->connect_error);
13  }
14
15
16  $sql = "SELECT pid, reason, duration FROM Pit_Stop";
17  $result = $conn->query(query: $sql);
18
19  if ($result->num_rows > 0) {
20
21      echo "<table border='1'><tr><th>Pit Stop ID</th><th>Reason</th><th>Duration</th></tr>";
22      while($row = $result->fetch_assoc()) {
23          echo "<tr><td>" . $row['pid'] . "</td><td>" . $row['reason'] . "</td><td>" . $row['duration'] . "</td></tr>";
24      }
25      echo "</table>";
26  } else {
27      echo "No pit stops found.";
28  }
29
30  $conn->close();
31  ?>
```

After the placement of files into xampp -> htdocs, the path http://localhost/index_pit_stop.html can be used to access to following page which allows insertion with the input taken from user:



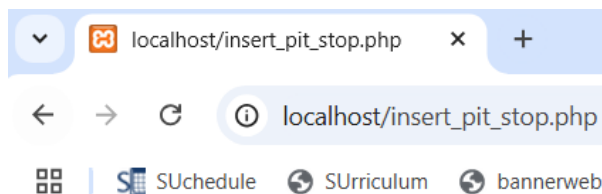
Insert New Pit Stop

Pit Stop ID:

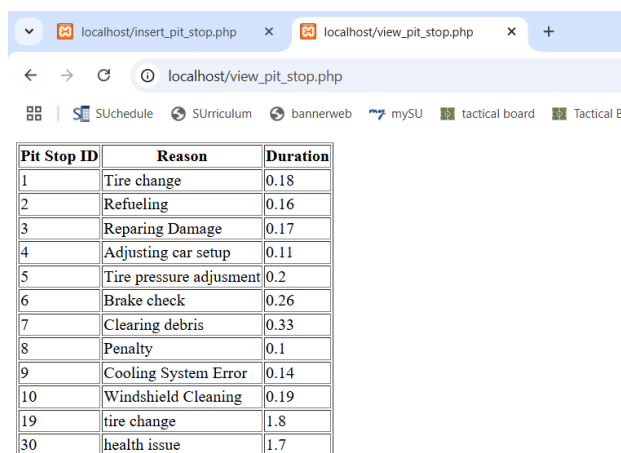
Reason:

Duration (seconds):

If we make a successful insertion, the message will appear as the following:



The final state of the Pit_Stop table can be viewed from http://localhost/view_pit_stop.php:



Pit Stop ID	Reason	Duration
1	Tire change	0.18
2	Refueling	0.16
3	Reparing Damage	0.17
4	Adjusting car setup	0.11
5	Tire pressure adjustment	0.2
6	Brake check	0.26
7	Clearing debris	0.33
8	Penalty	0.1
9	Cooling System Error	0.14
10	Windshield Cleaning	0.19
19	tire change	1.8
30	health issue	1.7