**מיני פרויקט בסיסי נתונים - ניהול כרטיסים עבור רכבת**

**יונה אורונוב ואלעד יושעי**

**הקדמה**

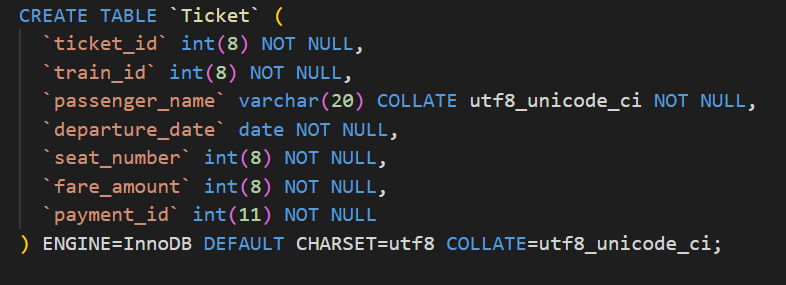
מטרת הפרויקט היא לייצג בסיס נתונים של מערכת לניהול כרטיסים בחברת רכבת כלשהי. לצורך כך יש צורך בטבלאות על מידע נוסף מעבר למידע על הכרטיס עצמו

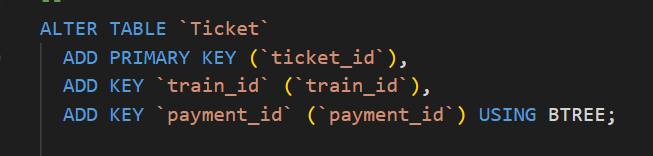
**שלב 1 -הגדרת טבלאות (שם הקובץ id20370336\_tickets)**

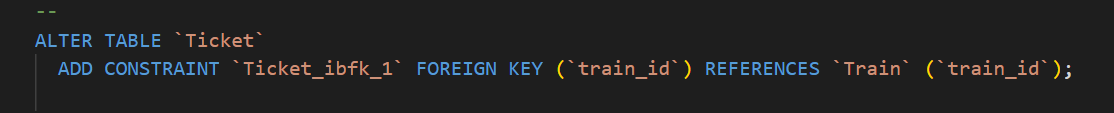
**Ticket**

טבלה שתחזיק את הנתונים על הכרטיס עצמו ותכיל מפתח אחד ומפתח זר אחד וסך הכל יהיה 8 שדות לכל כרטיס

| שם השדה | הסבר | טיפוס |
| --- | --- | --- |
| ticket\_id | מפתח,לכל כרטיס מספר מזהה משלו | INT |
| train\_id | מפתח זר, מזהה למפתח שהכרטיס מיועד עליו | INT |
| passenger\_name | שם הנוסע | VARCHAR |
| departure\_date | תאריך הנסיעה שעליו הכרטיס מיועד | DATE |
| departure\_time | זמן יציאה של הנסיעה של הכרטיס | DATE |
| arrival\_time | זמן הגעה של הנסיעה של הכרטיס | DATE |
| seat\_number | מספר מושב | INT |
| fare\_amount | מחיר הכרטיס | INT |

ליצירת טבלאות נשתמש בקוד phpmysql 

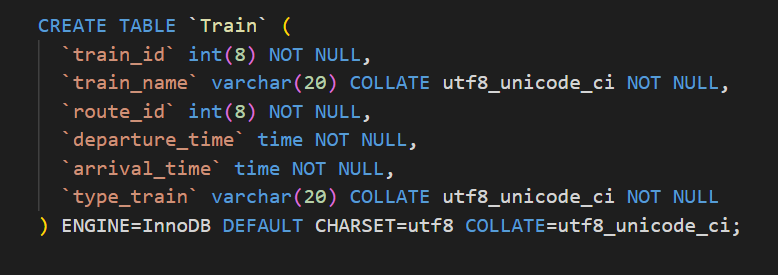
ונשתמש בקוד הבא על מנת ליצור קשרויות 



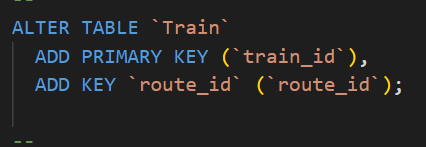
**Train**

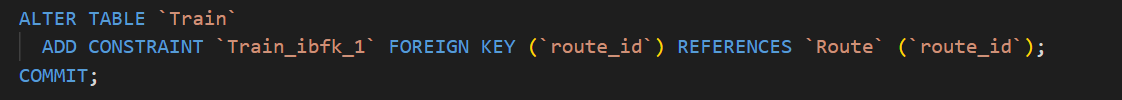
טבלה שתחזיק נתונים על כל הרכבות ומפתח אחד ו3 מפתחות זרים

| שם השדה | הסבר | טיפוס |
| --- | --- | --- |
| train\_id | מפתח, לכל רכבת מספר מזהה שלה | INT |
| train\_name | שם הרכבת | VARCHAR |
| route\_id | מפתח זר, המספר המזהה של הדרך שהרכבת נוסעת עליה | INT |
| departure\_station\_id | מפתח זר, מספר המזהה של התחנה שממנה הרכבת יוצאת | INT |
| arrival\_station\_id | מפתח זר, מספר המזהה של התחנה שממנה הרכבת מגיעה | INT |
| departure\_time | זמן היציאה של הרכבת | DATE |
| arrival\_time | זמן ההגעה של הרכבת | DATE |

ליצירת טבלאות נשתמש בקוד phpmysql 

ונשתמש בקוד הבא על מנת ליצור קשרויות

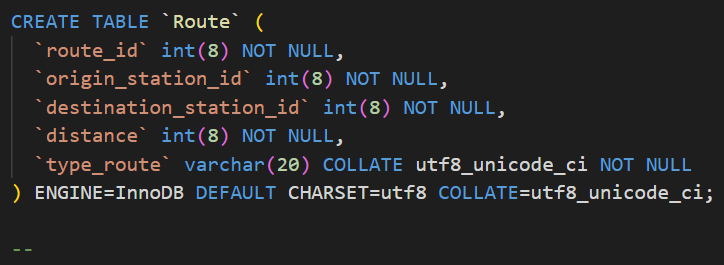




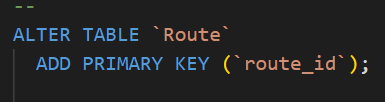
**Route**

הטבלה תחזיק נתונים על כל הדרכים שהרכבות נושאות עליהן ויש לזה מפתח אחד ו2 מפתחות זרים

| שם שדה | הסבר | טיפוס |
| --- | --- | --- |
| route\_id | מפתח, מספר מזהה של הדרך | INT |
| origin\_station\_id | מפתח זר, מספר מזהה של התחנה שממנה יוצאת הרכבת | INT |
| destination\_station\_id | מפתח זר, מספר מזהה של התחנה שעליה מגיעה הרכבת | INT |
| distance | המרחק של הדרך | INT |

ליצירת טבלאות נשתמש בקוד phpmysql 

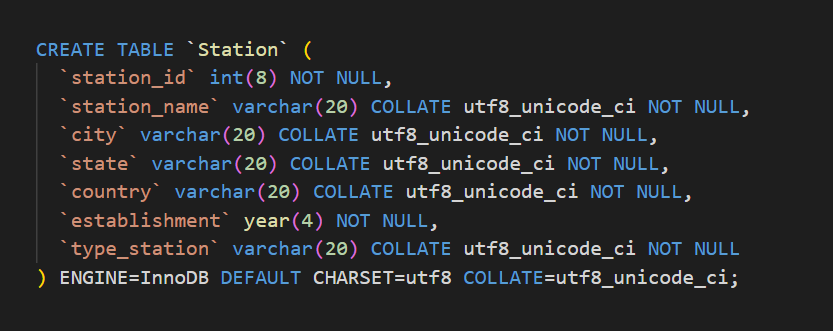
ונשתמש בקוד הבא על מנת ליצור קשרויות



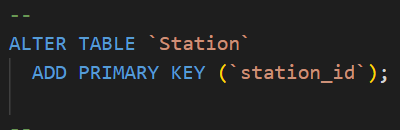
**Station**

טבלה עם הנתונים על כל תחנה שהרכבת עוצרת בה. הטבלה תכיל מפתח אחד בלי מפתחות זרים

| שדה | הסבר | טיפוס |
| --- | --- | --- |
| station\_id | מפתח, מספר מזהה של כל תחנה | INT |
| station\_name | שם התחנה | VARCHAR |
| city | שם העיר שהתחנה מצאת בה | VARCHAR |
| state | שם המחוז שהתחנה נמצאת בה | VARCHAR |
| country | שם המדינה שהתחנה נמצאת בה | VARCHAR |
| establishment | תאריך הרכבה | int |
| type\_station | סוג התחנה | VARCHER |

ליצירת טבלאות נשתמש בקוד phpmysql 

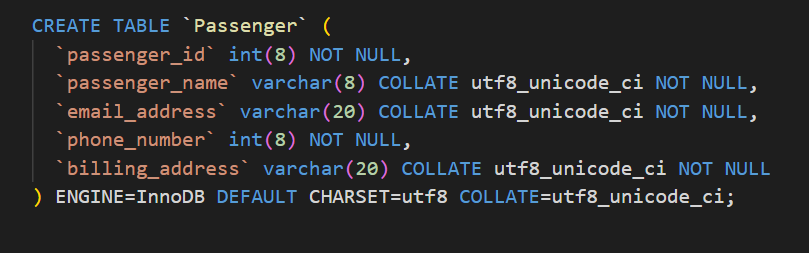
ונשתמש בקוד הבא על מנת ליצור קשרויות



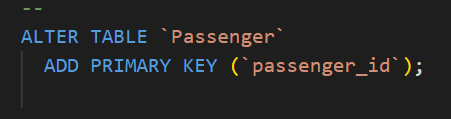
**Passenger**

הטבלה תחזיק נתונים על כל אדם שרכש כרטיס ואת כל הנתונים הדרושים, לטבלה יהיה מפתח אחד ולא יהיה מפתחות זרים

| שדה | הסבר | טיפוס |
| --- | --- | --- |
| passenger\_id | מפתח, מספר מזהה של כל נוסע | INT |
| passenger\_name | השם כל הנוסע | VARCHAR |
| email\_address | האימייל של הנוסע | VARCHAR |
| phone\_number | המספר הטלפון של הנוסע | INT |
| billing\_address | הכתובת למשלוח חשבונית לנוסע | VARCHAR |

ליצירת טבלאות נשתמש בקוד phpmysql 

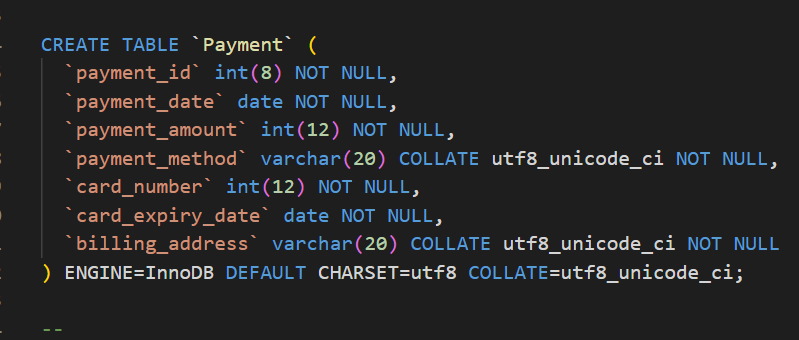
ונשתמש בקוד הבא על מנת ליצור קשרויות



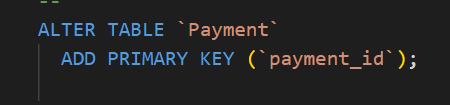
**Payment**

הטבלה תכיל נתונים על הרכישה עצמה שבוצע על מנת לקנות את הכרטיס. בטבלה יהיה מפתח אחד ומפתח זר אחד

| שדה | הסבר | טיפוס |
| --- | --- | --- |
| payment\_id | מפתח,מספר מזהה של הרכישה | INT |
| ticket\_id | מפתח זר, המספר המזהה של הכרטיס שהמשתמש קיבל בעקבות הרכישה | INT |
| payment\_date | תאריך ביצוע הרכישה | DATE |
| card\_number | מספר כרטיס אשראי | INT |
| card\_expiry\_date | תאריך התוקף של האשראי | DATE |
| billing\_address | כתובת שרוכש | VARCHAR |
| payment\_amount | עלות הרכישה | INT |
| payment\_method | צורת החיוב | VARCHAR |

ליצירת טבלאות נשתמש בקוד phpmysql 

ונשתמש בקוד הבא על מנת ליצור קשרויות



**מילוי הטבלאות**

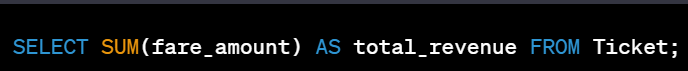
על מנת למלא את הטבלאות נשתמש בשאילתת SQL מסוג INSERT. בהתחלה על מנת לבצע את ההכנה נכתוב לבד את השאילתה בהתאם לטבלה שעליה היא מיועדת עם הערכים המתאימים. לאחר מכן על מנת להכניס הרבה ערכים (1000 לכל טבלה) כתבנו סקריפט פייתון לכל טבלה אשר יוצר 1000 שאילתות שמכניסות 1000 אובייקטים לטבלאות ( נמצא בקובץ ZIP Insert שיש שמה לכל טבלה וטבלה את הסקריפט שלו). לבסוף נריץ את התוצאה ב PHPMYADMIN והערכים יוכנסו

**שלב 2 - יצירת שאילתות**

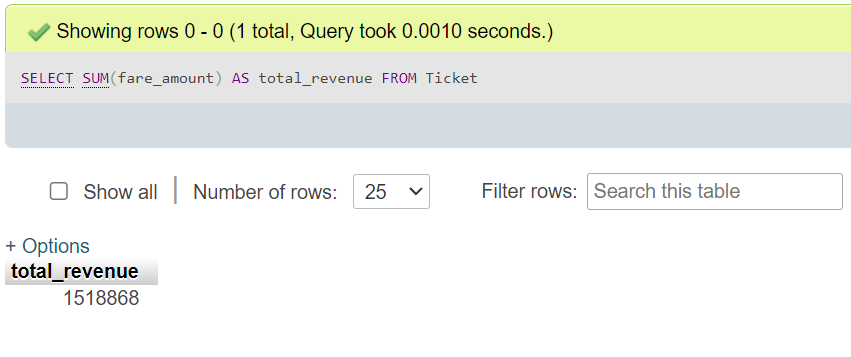
בשלב זה נכתוב שאליתות אשר ירוצו על בסיס הנתונים שלנו ויחזירו לנו פלטים בהתאם לשאילתה.

**שאילתא 1**

השאילתה תרוץ על הטבלה Ticket ותחזיר את הרווח של החברה ממכירת הכרטיסים בעזרת שימוש בפונקציה sum על השדה fare\_amount שיש בטבלה



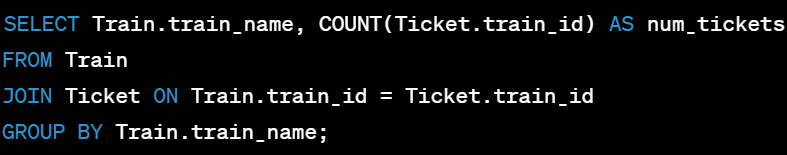
לאחר הרצת השאילתה ניתן לראות שהתוצאה היא



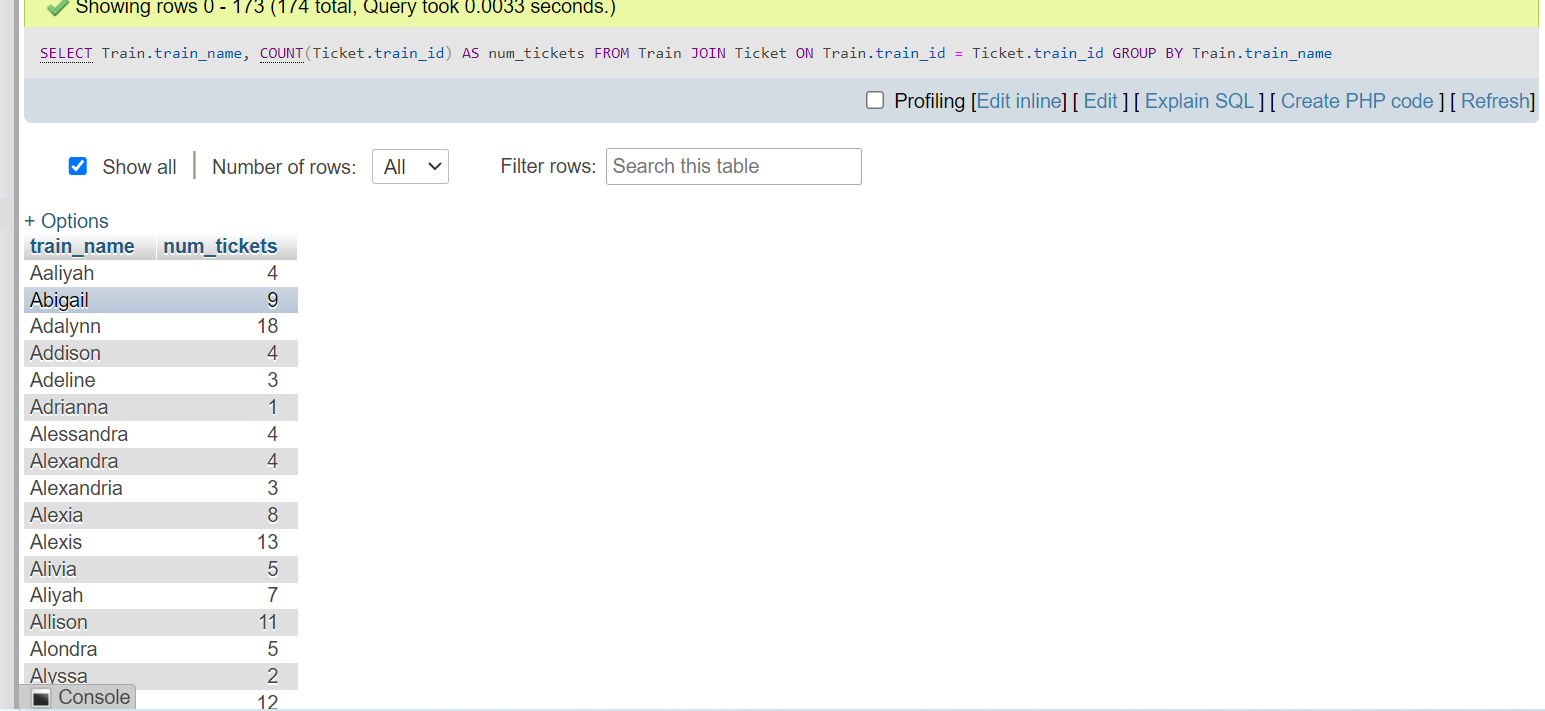
תוצאה בTICKET.SQL

**שאילתא 2**

השאילתה תרוץ על הטבלה Tickets ותחזיר את מספר הכרטיסים שנמכרו עבור כל רכבת. הרכבת תבצע צירוף בין טבלה Train לטבלה Tickets עם סינון של id משותף בין הכרטיס לרכבת. לאחר מכן השאילתה תקבץ לקבוצות את הכרטיסים בעזרת פונקציה GROUP לפי השדה train\_name שיש בטבלת Train. לאחר מכן שאילתא תשתמש בפונקציה COUNT שתמנע את הכרטיסים על פי השדה train\_id שיש בטבלה Ticket



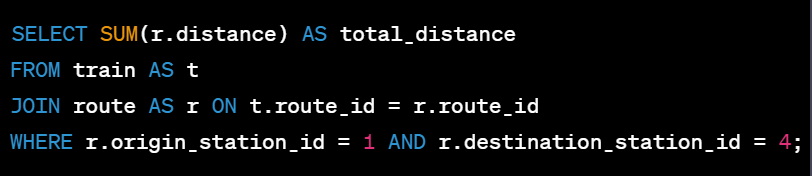
לאחר הרצת שאילתא ניתן לראות את התוצאה המוצגת לכל שם הרכבת כמה כרטיסים נמכרו לה



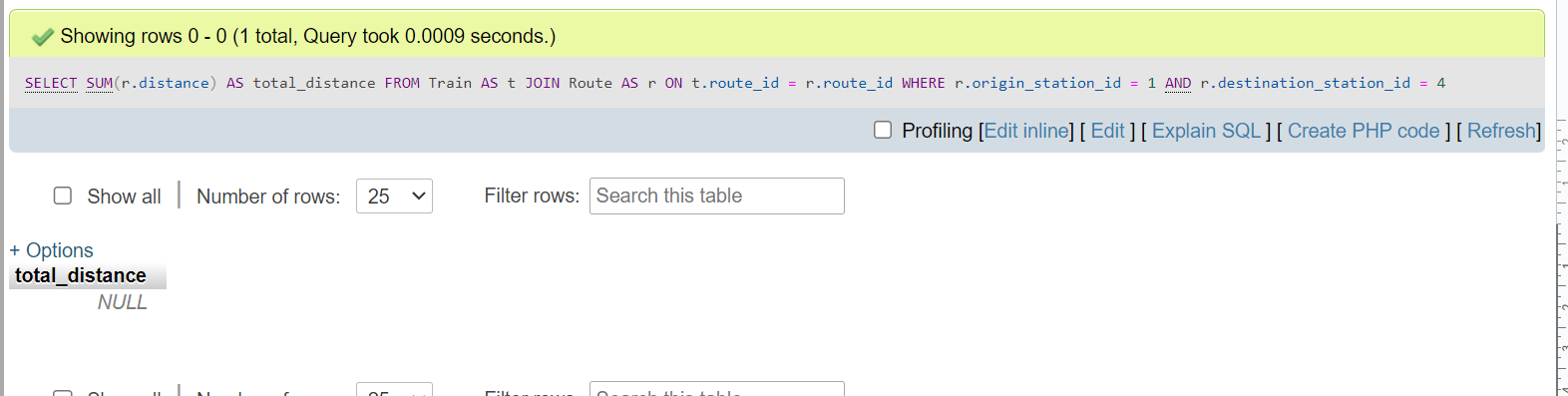
תוצאה מלאה בקובץ TRAIN.SQL

**שאילתא 3**

השאילתה תסכום בעזרת פונקצית sum את המרחק שכל הרכבות עשו בין 2 תחנות ספציפיות. השאילתה מצרפת את הטבלה של הרכבות ואת הטבלה של הדרכים בעזרת פונקציה JOIN ולאחר מכן מבצעת סינון בעזרת השוואה של ID של הדרך שווה. לאחר מכן יש סינון על פי הדרך מתי שגם תחנת המוצא וגם תחנת היעד שוות למה שצריך



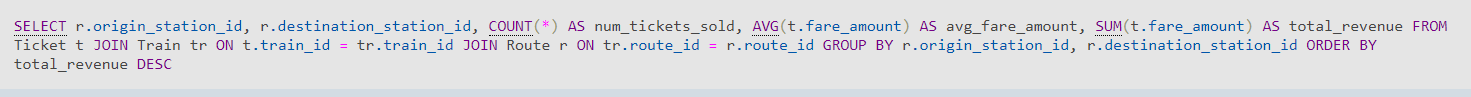
ולכן ניתן לראות שאין כזאת רכבת

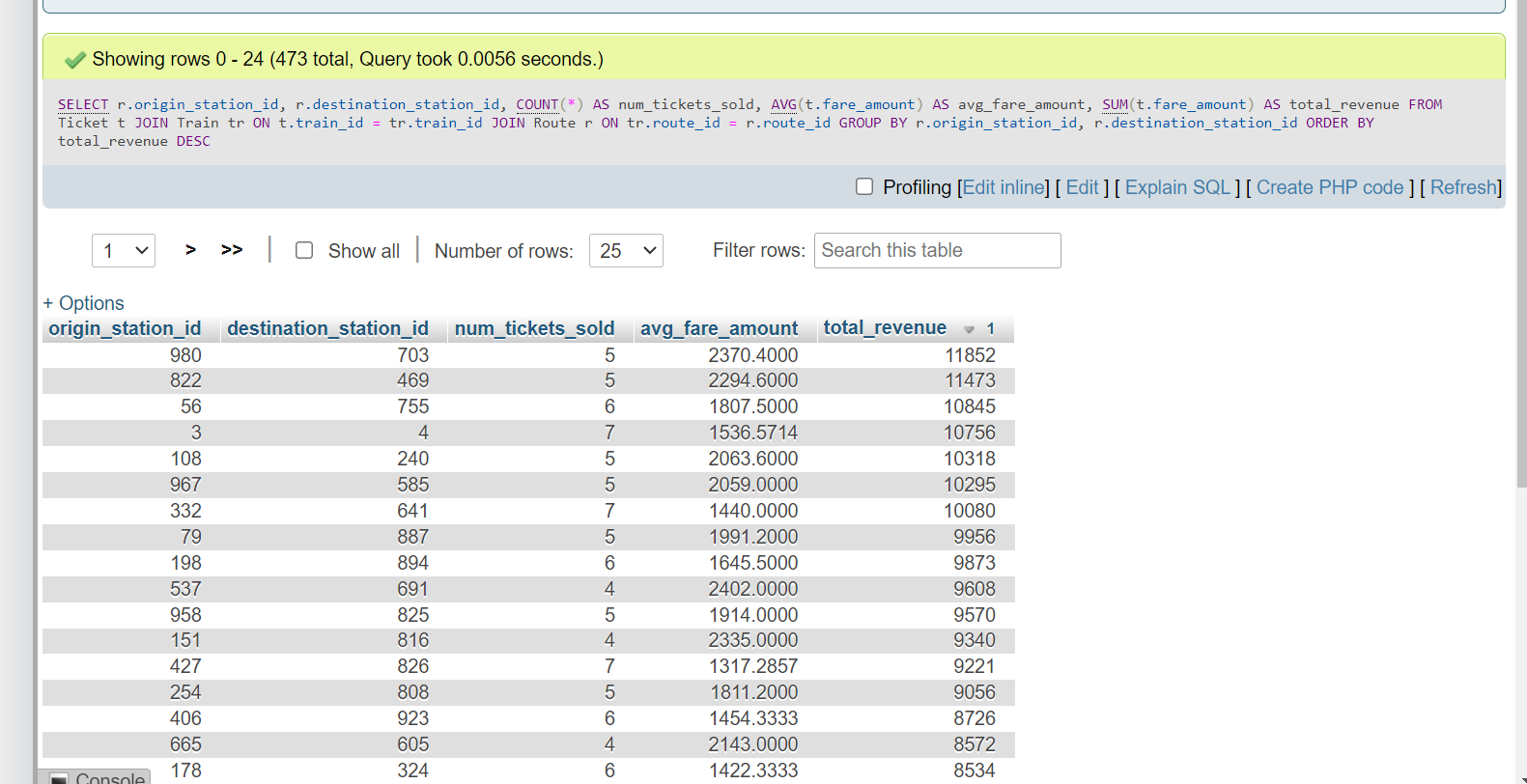


תוצאה ב TRAIN(1).SQL

**שאליתא 4**

שאילתא מוצאת את כמות הרווח לכל דרך עם הכמות הנסיעה הממוצעת סכום המכירה הכללי של הכרטיסים. זה קורה בעזרת שילוב של הטבלאות של הכרטיסים של הדרך ושל הרכבות בעזרת סינון של ON רק איפה שיש שוויון ולבסוף מבצעים קיבוץ לפי תחנת יעד תחנת המוצא ויש מיון לפי הנסיעה הכללית



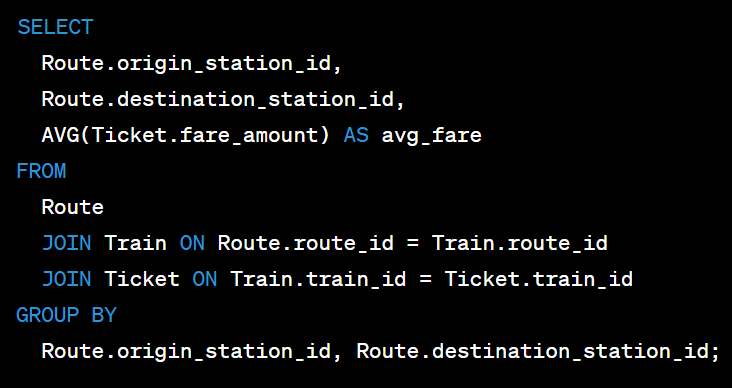
והתוצאה היא 

קובץ התוצאות המלא הוא

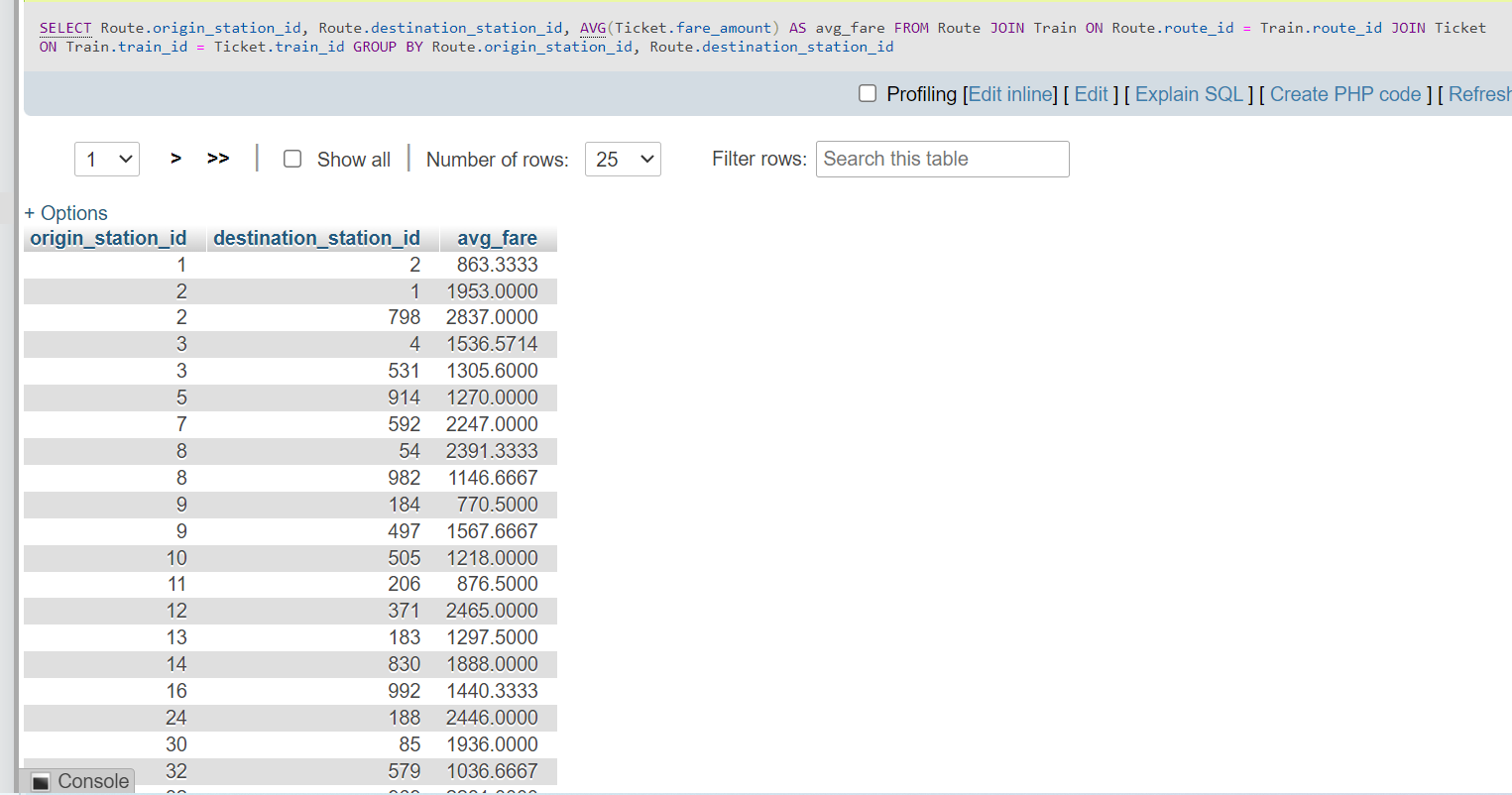
r.sql

**שאילתא 5**

התוצאה מחזירה את הFARE\_AMOUNT הממוצע לכל דרך והיא מחזירה את התחנת ההתחלה והתחנת הסוף של הדרך. שאילתא מחברת בין ROUTE TRAIN TICKET בעזרת JOIN עם סינון בעזרת IN ולבסוף מקבצת על פי תחנת יעד תחנת המוצא של הדרך



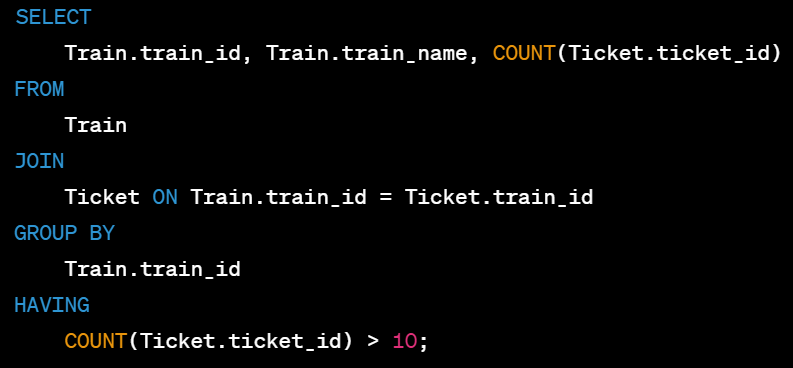
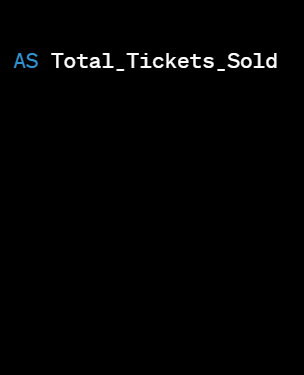
תוצאה של הרצה היא



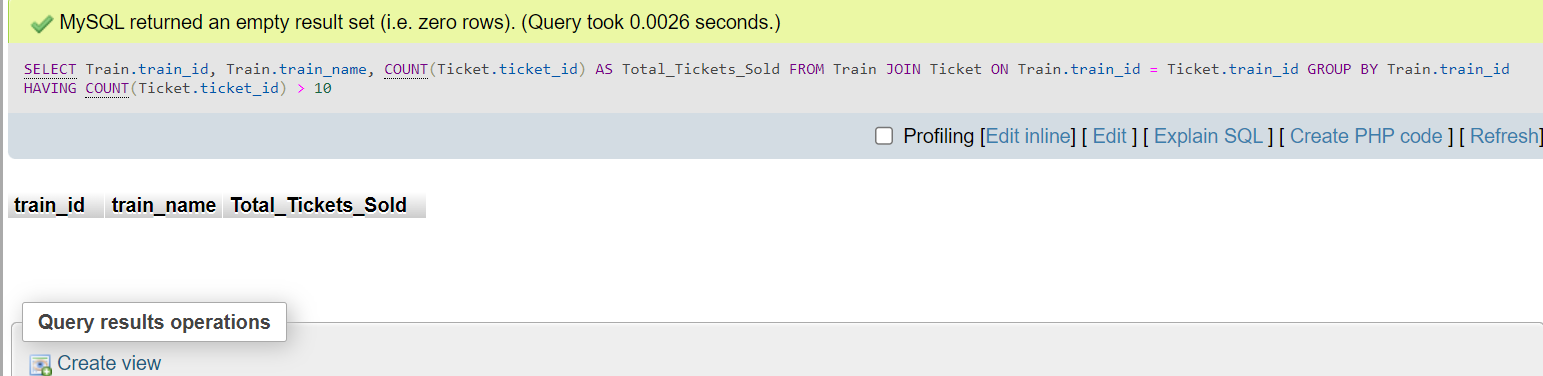
התוצאות המלאות בROUTE.SQL

**שאילתא 6**

השאילתה מחזירה את השם והID של הרכבת וכמות הכרטיסים שנמכרו לאותה רכבת בתנאי שהכמות מעל 10. הפונקציה מצרפת את הטבלה של הרכבת ואת הטבלה של הכרטיסים יחד איפה שיש שוויון בין ה ID של הרכבת לID ששמור בכרטיס המציין את הרכבת עליו הוא משתייך. לאחר מכן יש צירוף בעזרת ה ID של הרכבת פונקציה GROUP. ולבסוף סינון של הקבוצות איפה יש יותר מ10 בעזרת הפונקציות COUNT ו HAVING



וניתן לראות שזה החזיר נתונים ריקים



**שלב 3 - איחוד טבלאות**

בשלב הזה בפרויקט שלנו אנחנו נרצה להרחיב את בסיס הנתונים שלנו ולכן נקבל את הטבלאות של זוג אחר בקורס ונשלב את הטבלאות יחד. הטבלאות שקיבלנו מהזוג האחר הן טבלאות של בסיס נתונים שמנהלת עובדים כולמר טבלאות אשר יכילו נתונים על כל העובדים ועל הדרך שאיתם אנחנו נחשב את המשכורות ויצירת טבלאות לכל סוג, בנוסף אנחנו היא מטפלת גם במה שקשור לימי החופש שיש וכו'

הטבלאות שקיבלנו הן :

עובד (employee)- המכילה פרטים בסיסיים הנוגעים לכל עובד בחברה (ת"ז, שם, מגדר, דרגה ניהולית, ותק, היקף משרה, ומספר שעות נוספות)

טבלת היעדרויות - טבלה השומרת את הנתונים עבור כל היעדרות עובד: מספר ימי ההיעדרות והאם ההיעדרות גוררת ניכוי מהשכר (חופשה) או לא (מחלה). עובד יכול להיעדר מספר פעמים בחודש, וכל תקופת היעדרות מיוצגת על ידי שורה בטבלה בנפרד

טבלת מס - טבלת נקודות זיכוי במס הכנסה השומרת עבור כל עובד את הפרטים הרלוונטיים (מצב משפחתי, מקום מגורים וכו').

שכר ברוטו - טבלת נקודות זיכוי במס הכנסה השומרת עבור כל עובד את הפרטים הרלוונטיים (מצב משפחתי, מקום מגורים וכו').

טבלת סוגי עובדים - נרצה להגדיר סוגי עובדים ולכן לכל סוג ניצור טבלה משלו אשר תירוש מהטבלה של עובד כללי

· מאבטח (שם תחנה, סוג נשק ['Glock', 'Uzi', 'M16']).

· מנהלן (יש תואר [bool], כפל תואר [bool]).

· קופאי (שם תחנה).

· מהנדס (יש תואר [bool], כפל תואר [bool]).

· פקח (שם תחנה, תפקיד ['signalman','controller']).

· נהג (סוג רישיון ['A', 'B', 'C'], מספר שעות עבודה בסיכון).

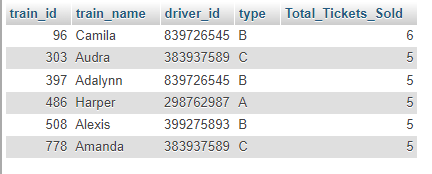
לאחר שילוב הטבלאות לא אמור להיות לנו בעיות של אי סנכרון ואינטגרציה צריכה להיות פשוטה כי גם אין שוב טבלאות משיקות ולכן גם השאילתות אשר יצרנו בחלק 2 לא אמורות להשתנות

**שלב 4 - יצירת שאילתות אינטגרציה**

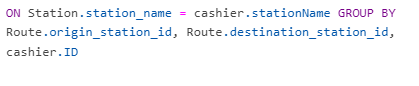
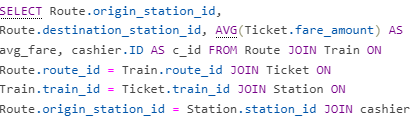
בחלק הזה אנחנו ניצור שאילתות על הטבלאות הנתונים החדשות והישנות.

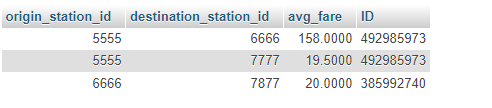
בהתחלה נרחיב 2 שאילתות ישנות להיות תקפות גם על החדשות

1. נרחיב את שאילתא 3 כך שהפעם היא תתאים לכל רכבת שיש לה יותר מ 4 כרטיסים שנמכרו היא תתן גם נהג ותציג את סוג הרישיון שיש לו ואת המספר המזהה של אותו נהג. הפונקציה תבחר את הנהג באופן רנדומלי 

והתוצאה שנקבל היא 

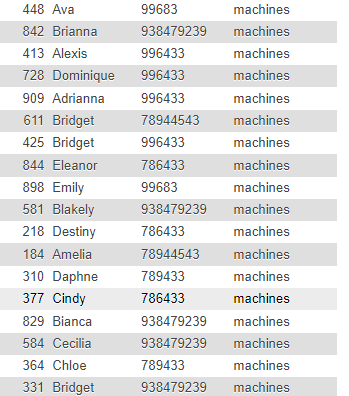
1. נרחיב את שאילתה 2 בשביל כדי שהפעם גם תדפיס המספר המזהה של הקופאי בתחנת היציאה. נעשה את זה בעזרת פעולת JOIN פעולת השוואה פשוטה



והתוצאה שנקבל היא 

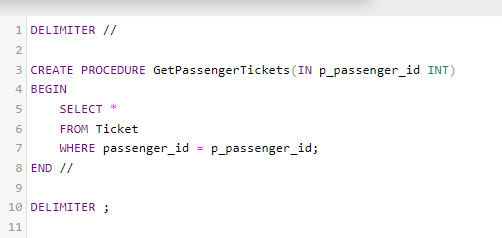
1. ניצור טבלת VIEW כך שתכיל לכל רכבת מהנדס אחת שתחום המקצוע שלו זה מכונאי ונגביל את מספר ההנדסאים לאחד 

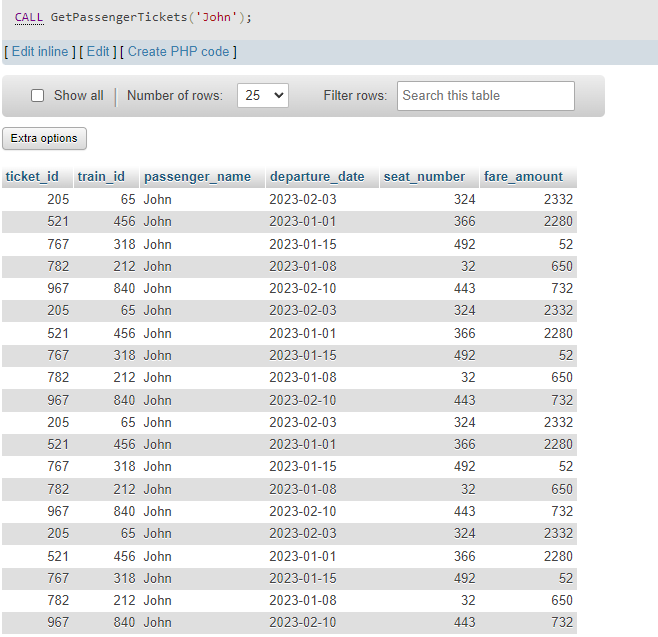
והתוצאה שנקבל היא הנתונים על הרכבת המספר המזהה של המהנדס

חלק מהתוצאות הן 

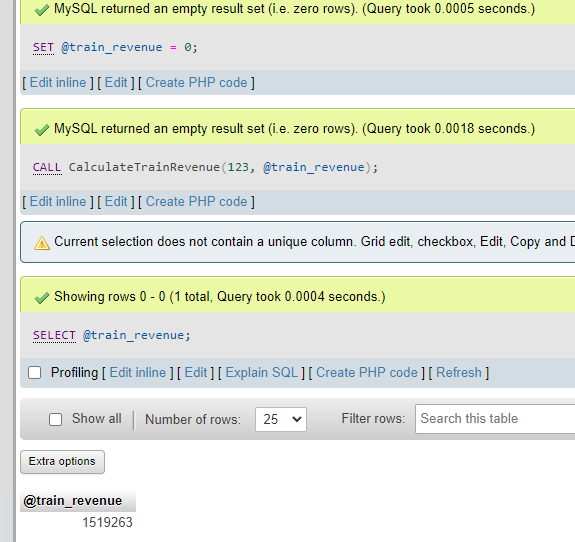
1. ניצור טבלת VIEW שתתן לנו נותנים על שילוב של הטבלאות הישנות של דרך ושל תחנה ונשלב עם הנתונים של המוכר 

התוצאות שנקבל הן 

1. ניצור שאילתא שמתי שהיא מקבלת שם של נוסע היא תחזיר את כל הפרטי כרטיס שלו 

ונראה שימוש לזה 

1. ניצור שאילתה שמקבלת ID של רכבת ותחזיר לנו את סך המרחק שהיא עשתה 

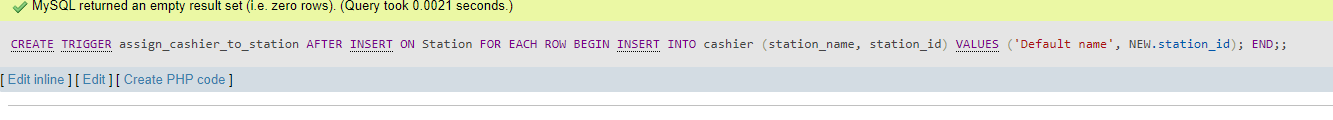
ונראה שימוש לזה 

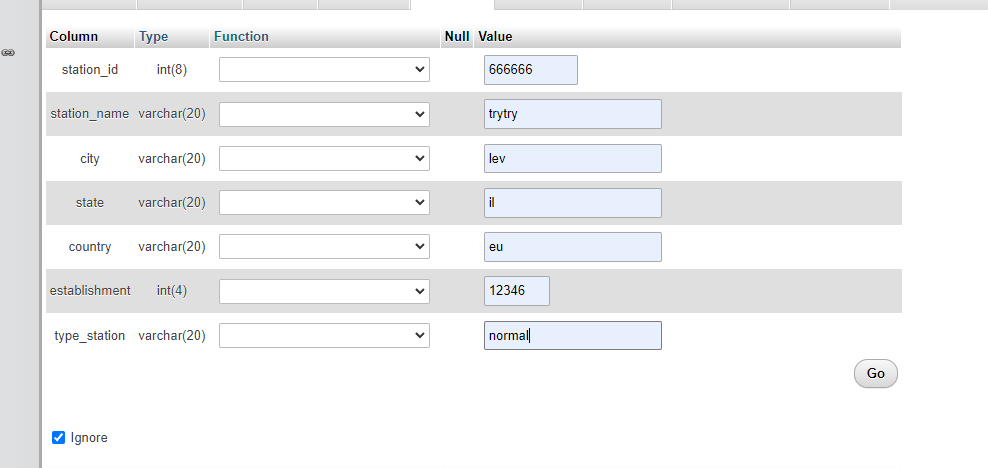
1. שאילתא תחזיר עבור כל דרך את כמות הכסף שנכנס ממכירת כרטיסים לדרך הזאת

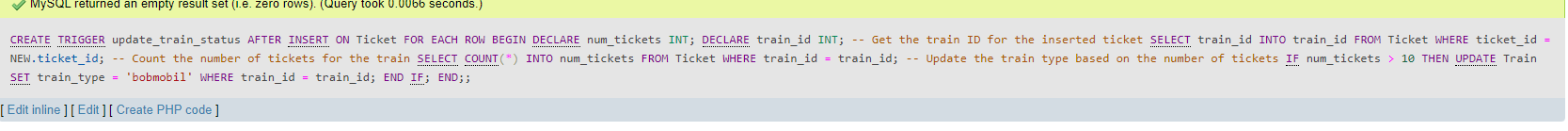
כאשר 480 ו562 זה ערכים שניתן להחליף. בדוגמה הזאת התוצאה זה 

1. ניצור שאילתא שתחזיר לנו את שמות כל העבודים ופרטי תחנה שהמשתמש יכניס לדוגמא עבור 

הקלט נבון יצא לנו 

1. נעשה תרגיל - שכל פעם שאנחנו מוסיפים תחנה חדשה אז היא גם גורמת לשמירה של קופאי ברירת מחדל 

ונראה אופן פעולה  יוצא לנו 

1. נוסיף טריגר נוסף בכל פעם שיש הוספת כרטיס אנחנו נבדוק אם לרכבת יש מעל 10 כרטיסים שנמכרו עליה ואם באמת יש אז אנחנו נשנה את הסוג שלה להיות מסוג בובמוביל

**שלב 5 - ממשק משתמש :**

התבקשנו בשלב זה לגרום לשימוש נח ויעיל בבסיס הנתונים

על מנת לעשות זאת הורדנו את בסיס הנתונים למחשב האישי וכתבנו קוד בphp שירוץ על הlocal host וייגרום לכך להתבצע

הקוד הבא גורם לכך שיהיה ניתן לגשת לבסיס הנתונים שלנו בצורה קלה ולבצע בו מספר פעולות:

1. למחוק נתונים

2. לשנות אותם

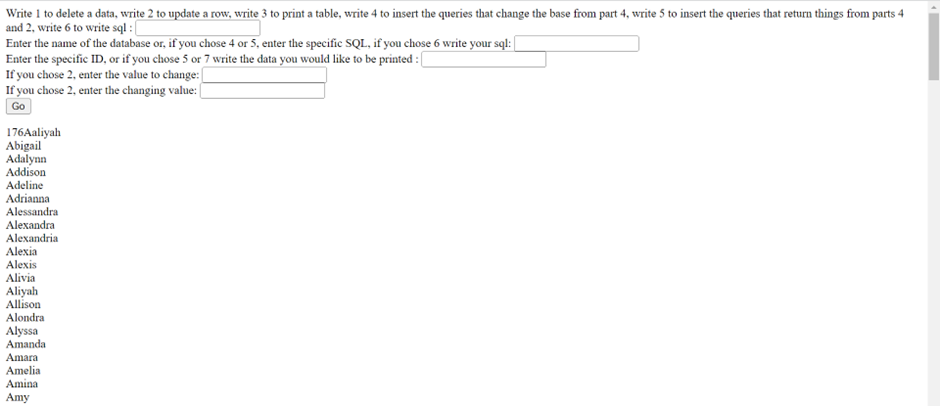
3. להדפיס ערכים ספציפיים מטבלה

4. להריץ את השאילתות שכתבנו בשלב 4 שמשנות נתונים

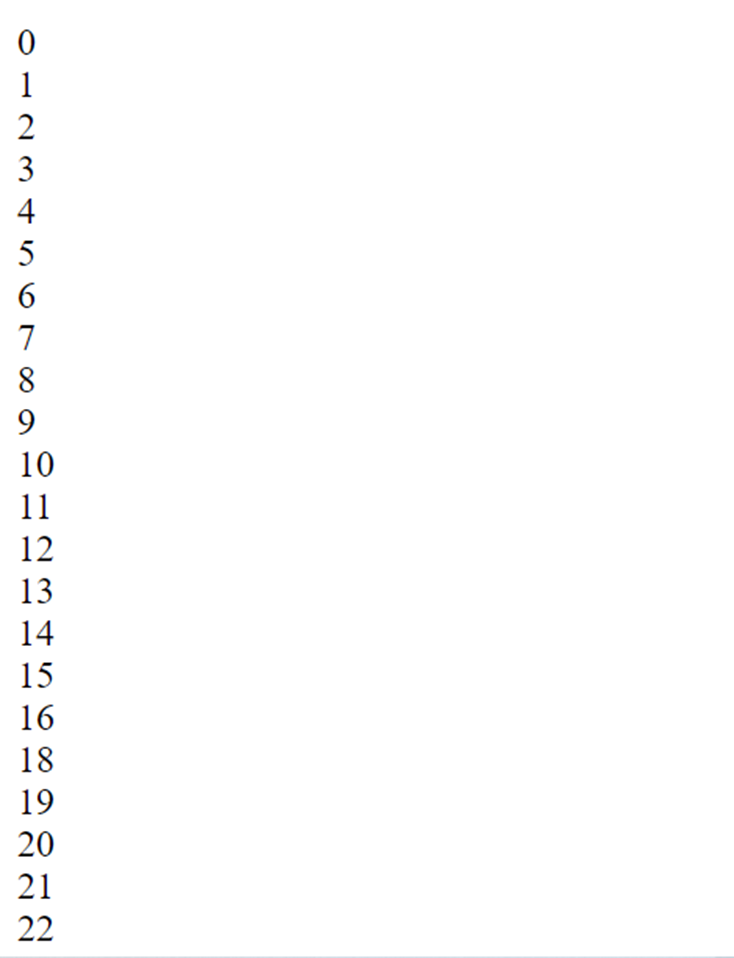
5. להריץ את השאילתות משלבים 2 ו4 שמחזירות נתונים

6. לכתוב שאילתה באופן עצמאי

דוגמאות לשימוש באתר:



המשתמש כתב שאילתה בעצמו והחזרנו את הערכים שהוא ביקש



ניתן לראות שהמשתמש בחר למחוק את ערך 17 ולאחר מכן להדפיס את כל הid של train

להלן הקוד שכתבנו:

<html>

<body>

<form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">

Write 1 to delete a data, write 2 to update a row, write 3 to print a table,

write 4 to insert the queries that change the base from part 4, write 5 to insert the queries that return things from parts 4 and 2, write 6 to write sql : <input type="text" name="value1"><br>

Enter the name of the database or, if you chose 4 or 5, enter the specific SQL, if you chose 6 write your sql: <input type="text" name="value2"><br>

Enter the specific ID, or if you chose 5 or 7 write the data you would like to be printed : <input type="text" name="value3"><br>

If you chose 2, enter the value to change: <input type="text" name="value4"><br>

If you chose 2, enter the changing value: <input type="text" name="value5"><br>

<input type="submit" value="Go">

</form>

<?php

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// collect values from input fields

$action = $\_POST['value1'];

$name = $\_POST['value2'];

$what = $\_POST['value3'];

$val1 = $\_POST['value4'];

$val2 = $\_POST['value5'];

if (empty($action) || empty($name)) {

echo "Action is empty";

} else {

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "project1";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if ($action == 1) {

// Delete a record

$sql = "DELETE FROM " . $name . " WHERE " . $name . "\_id = " . $what;

if ($conn->query($sql) === TRUE) {

echo "Record deleted successfully";

} else {

echo "Error deleting record: " . $conn->error;

}

} elseif ($action == 2) {

// Update a record

$sql = "UPDATE " . $name . " SET " . $val1 . " = '" . $val2 . "' WHERE " . $name . "\_id = " . $what;

if ($conn->query($sql) === TRUE) {

echo "Record updated successfully";

} else {

echo "Error updating record: " . $conn->error;

}

} elseif ($action == 3) {

// Retrieve a list

// Assuming the list name is provided in the 'fname' input field

$sql = "SELECT \* FROM " . $name;

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

while ($row = $result->fetch\_assoc()) {

echo $row[$what] . "<br>";

}

} else {

echo "0 results";

}

} elseif ($action == 4) {

// Custom queries

if ($name == 1) {

$sql = "SELECT

Train.train\_id,

Train.train\_name,

(

SELECT driver.ID

FROM driver

ORDER BY RAND()

LIMIT 1

) AS driver\_id,

(

SELECT driver.License

FROM driver

WHERE driver.ID = driver\_id

) AS type,

COUNT(Ticket.ticket\_id) AS Total\_Tickets\_Sold

FROM

Train

JOIN

Ticket ON Train.train\_id = Ticket.train\_id

GROUP BY

Train.train\_id

HAVING

COUNT(Ticket.ticket\_id) > 4";

} elseif ($name == 2) {

$sql = "SELECT

Route.origin\_station\_id,

Route.destination\_station\_id,

AVG(Ticket.fare\_amount) AS avg\_fare,

cashier.ID AS c\_id

FROM Route

JOIN Train ON Route.route\_id = Train.route\_id

JOIN Ticket ON Train.train\_id = Ticket.train\_id

JOIN Station ON Route.origin\_station\_id = Station.station\_id

JOIN cashier ON Station.station\_name = cashier.stationName

GROUP BY Route.origin\_station\_id, cashier.ID, Route.destination\_station\_id";

} elseif ($name == 3) {

$sql = "CREATE VIEW RouteStationCashierView AS

SELECT

Route.route\_id,

Route.origin\_station\_id,

Route.destination\_station\_id,

Station.station\_name,

cashier.ID,

cashier.stationName

FROM

Route

JOIN

Station ON Route.origin\_station\_id = Station.station\_id

JOIN

cashier ON Station.station\_name = cashier.stationName";

} elseif ($name == 4) {

$sql = "CREATE VIEW EngineerTrainView AS

SELECT

Train.train\_id,

Train.train\_name,

engineer.ID AS engineer\_id,

engineer.type AS engineer\_type

FROM

Train

JOIN engineer ON engineer.type = 'machines'

ORDER BY RAND()";

} elseif ($name == 5) {

$sql = "CREATE PROCEDURE GetPassengerTickets(IN p\_passenger\_name VARCHAR(20))

BEGIN

SELECT t.ticket\_id, t.train\_id, t.passenger\_name, t.departure\_date, t.seat\_number, t.fare\_amount

FROM Ticket t

JOIN Passenger p ON t.passenger\_name = p.passenger\_name

WHERE p.passenger\_name = p\_passenger\_name";

} elseif ($name == 6) {

$sql = "CREATE PROCEDURE CalculateTrainRevenue(IN train\_id INT, OUT total\_revenue INT)

BEGIN

SELECT SUM(fare\_amount) INTO total\_revenue

FROM Ticket

WHERE train\_id = train\_id";

} elseif ($name == 7) {

$sql = "SELECT SUM(Ticket.fare\_amount) AS total\_revenue

FROM Ticket

JOIN Train ON Ticket.train\_id = Train.train\_id

JOIN Route ON Train.route\_id = Route.route\_id

WHERE Route.origin\_station\_id = ? AND Route.destination\_station\_id = ?";

} elseif ($name == 8) {

$sql = "CREATE TRIGGER assign\_cashier\_to\_station

AFTER INSERT ON Station

FOR EACH ROW

BEGIN

INSERT INTO cashier (ID, stationName)

VALUES (NEW.station\_id, 'Default Cashier')";

} elseif ($name == 9) {

$sql = "SELECT s.station\_name, s.city, s.country, c.ID

FROM Station s

JOIN Cashier c ON s.station\_name = c.stationName

WHERE s.station\_name = <station\_name>";

} elseif ($name == 10) {

$sql = "DELIMITER //

CREATE TRIGGER update\_train\_status

AFTER INSERT ON Ticket

FOR EACH ROW

BEGIN

DECLARE num\_tickets INT;

DECLARE train\_id INT;

-- Get the train ID for the inserted ticket

SELECT train\_id INTO train\_id FROM Ticket WHERE ticket\_id = NEW.ticket\_id;

-- Count the number of tickets for the train

SELECT COUNT(\*) INTO num\_tickets FROM Ticket WHERE train\_id = train\_id";

}

$result = $conn->query($sql);

}elseif ($action == 5) {

// Custom queries

if ($name == 1) {

$sql = "SELECT

Train.train\_id,

Train.train\_name,

(

SELECT driver.ID

FROM driver

ORDER BY RAND()

LIMIT 1

) AS driver\_id,

(

SELECT driver.License

FROM driver

WHERE driver.ID = driver\_id

) AS type,

COUNT(Ticket.ticket\_id) AS Total\_Tickets\_Sold

FROM

Train

JOIN

Ticket ON Train.train\_id = Ticket.train\_id

GROUP BY

Train.train\_id

HAVING

COUNT(Ticket.ticket\_id) > 4";

} elseif ($name == 2) {

$sql = "SELECT

Route.origin\_station\_id,

Route.destination\_station\_id,

AVG(Ticket.fare\_amount) AS avg\_fare,

cashier.ID AS c\_id

FROM Route

JOIN Train ON Route.route\_id = Train.route\_id

JOIN Ticket ON Train.train\_id = Ticket.train\_id

JOIN Station ON Route.origin\_station\_id = Station.station\_id

JOIN cashier ON Station.station\_name = cashier.stationName

GROUP BY Route.origin\_station\_id, cashier.ID, Route.destination\_station\_id";

} elseif ($name == 3) {

$sql = "SELECT r.origin\_station\_id, r.destination\_station\_id, COUNT(\*) AS num\_tickets\_sold,

AVG(t.fare\_amount) AS avg\_fare\_amount, SUM(t.fare\_amount) AS total\_revenue

FROM Ticket t

JOIN Train tr ON t.train\_id = tr.train\_id

JOIN Route r ON tr.route\_id = r.route\_id

GROUP BY r.origin\_station\_id, r.destination\_station\_id

ORDER BY total\_revenue DESC";

} elseif ($name == 4) {

$sql = "SELECT

Route.origin\_station\_id,

Route.destination\_station\_id,

AVG(Ticket.fare\_amount) AS avg\_fare

FROM

Route

JOIN Train ON Route.route\_id = Train.route\_id

JOIN Ticket ON Train.train\_id = Ticket.train\_id

GROUP BY

Route.origin\_station\_id, Route.destination\_station\_id";

} elseif ($name == 5) {

$sql = "SELECT

Train.train\_id, Train.train\_name, COUNT(Ticket.ticket\_id) AS Total\_Tickets\_Sold

FROM

Train

JOIN

Ticket ON Train.train\_id = Ticket.train\_id

GROUP BY

Train.train\_id

HAVING

COUNT(Ticket.ticket\_id) > 10";

} elseif ($name == 6) {

$sql = "SELECT SUM(r.distance) AS total\_distance

FROM train AS t

JOIN route AS r ON t.route\_id = r.route\_id

WHERE r.origin\_station\_id = 1 AND r.destination\_station\_id = 4";

} elseif ($name == 7) {

$sql = "SELECT Train.train\_name, COUNT(Ticket.train\_id) AS num\_tickets

FROM Train

JOIN Ticket ON Train.train\_id = Ticket.train\_id

GROUP BY Train.train\_name";

} elseif ($name == 8) {

$sql = "SELECT SUM(fare\_amount) AS total\_revenue FROM Ticket";

}

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

while ($row = $result->fetch\_assoc()) {

echo $row[$what] . "<br>";

}

} else {

echo "0 results";

}

}elseif ($action == 6) {

$result = $conn->query($name);

echo $result->num\_rows;

if ($result->num\_rows > 0) {

while ($row = $result->fetch\_assoc()) {

echo $row[$what] . "<br>";

}

} else {

echo "0 results";

}

}else {

echo "Invalid action";

}

$conn->close();

}

}

?>

</body>

</html>