**Name : SHWETA BANGAD CWID: A20392438**

Contents

[PHASE V: REVIEW OF CURRENT PROJECT STATUS 2](#_Toc482118200)

[PHASE VI: DESIGNING THE PHYSICAL APPLICATION 2](#_Toc482118201)

[PHASE VII: LOADING THE TABLES WITH VALID DATA 6](#_Toc482118202)

[PHASE VIII: TESTING THE DATABASE SYSTEM 8](#_Toc482118203)

[PHASE IX: DATA ANALYTICS PERFORMED 14](#_Toc482118204)

[WEB PRESENCE: 17](#_Toc482118205)

[PHASE X: SYSTEM ANALYSIS AND VIEWPOINTS 26](#_Toc482118206)

# PHASE V: REVIEW OF CURRENT PROJECT STATUS

* In earlier phases, we had discussed few questions with the company and according to that we had the business rules to make the database with following entities.
* In my midterm project, I had 6 tables Accounting\_dept, Event\_table, Salesperson, Expense\_category, travel\_advance, Expense\_generated.
* In which all the entities were connected to another entity.
* This would have made my project a little complicated as I had to join many tables to get a single data.
* To make my project effective and efficient, we can make use of principles and facts of database management, data warehousing and analytics.
* So according to the rules of the database management, data warehousing, we could make use of the fact tables or the star schema, in which there are several dimension tables linked to one or more fact tables, where the measurable, quantitative data such as calculated parts can be stored.
* So, I had to make changes in my entities, making my entities as dimension tables and a fact table.
* Now the fact table in my project by using the principles of database management is the EXPENSE\_GENERATED table and the other tables are the dimension tables which give us all the information to put in fact table.

# PHASE VI: DESIGNING THE PHYSICAL APPLICATION

* The physical model includes all the entities with the proper primary key, foreign key to it. It gives the relationship between entities. So, for creating table following are the queries.

1. ACCOUNTING\_DEPT table:

DROP TABLE ACCOUNTING\_DEPT;

CREATE TABLE ACCOUNTING\_DEPT(

A\_ID INT NOT NULL,

A\_NAME VARCHAR (20),

A\_EMAIL VARCHAR (20),

USERNAME VARCHAR(20) ,

PASSWORD VARCHAR(20),

PRIMARY KEY (A\_ID)

);

1. SALESPERSON table:

DROP TABLE SALESPERSON;

CREATE TABLE SALESPERSON(

SP\_ID INT NOT NULL,

SP\_NAME VARCHAR (20),

SP\_EMAIL VARCHAR (20),

PRIMARY KEY (SP\_ID)

);

1. EXPENSE\_CATEGORY table:

DROP TABLE EXPENSE\_CATEGORY;

DROP TABLE EXPENSE\_CATEGORY;

CREATE TABLE EXPENSE\_CATEGORY(

SR\_NO INT NOT NULL,

TICKET\_FARE DECIMAL,

TAXI\_FARE DECIMAL,

HOTEL\_EXPENSE DECIMAL,

MEAL\_EXPENSE DECIMAL,

PRIMARY KEY (SR\_NO)

);

1. COMPANY\_EVENT table:

DROP TABLE COMPANY\_EVENTS;

CREATE TABLE COMPANY\_EVENTS(

EVENTid INT NOT NULL,

EVENT\_DESC VARCHAR(20),

STATUS VARCHAR(20),

PRIMARY KEY (EVENTid)

);

1. TRAVEL\_ADVANCE table:

DROP TABLE TRAVEL\_ADVANCE;

CREATE TABLE TRAVEL\_ADVANCE

(

CHECK\_NO INT,

AMOUNT\_GIVEN NUMBER,

TRAVELLING\_DATE DATE,

ISSUE\_DATE DATE,

PRIMARY KEY(CHECK\_NO));

1. EXPENSE\_GENERATED table:

DROP TABLE EXPENSE\_GENERATED;

CREATE TABLE EXPENSE\_GENERATED(

EXPENSE\_ID INT NOT NULL ,

EVENTid INT,

SP\_ID INT,

A\_ID INT,

CHECK\_NO INT,

SR\_NO INT,

TOTAL\_EXPENSE\_GENERATED NUMBER,

constraint fk\_EXPENSE\_GENERATED foreign key (EVENTid) references COMPANY\_EVENTS(EVENTid),

constraint fk\_EXPENSE\_GENERATED1 foreign key (SP\_ID) references SALESPERSON(SP\_ID),

constraint fk\_EXPENSE\_GENERATED2 foreign key (A\_ID) references ACCOUNTING\_DEPT(A\_ID),

constraint fk\_EXPENSE\_GENERATED3 foreign key (CHECK\_NO) references TRAVEL\_ADVANCE(CHECK\_NO),

constraint fk\_EXPENSE\_GENERATED4 foreign key (SR\_NO) references EXPENSE\_CATEGORY(SR\_NO),

PRIMARY KEY(EXPENSE\_ID)

);

* To make the Travel advance project more analytical and effective we can add a table which will give the name of cities where the travelers are visiting often.
* Therefore, with the approval of the team, we have decided to add a table named travelling\_city in travel advance project in which city\_code is the primary key and then in the travel\_advance table, we are using city\_code as a foreign key.
* Query for the TRAVELLING\_CITY table is shown below.

1. TRAVELLING\_CITY table:

CREATE TABLE TRAVELLING\_CITY(

CITY\_CODE INT NOT NULL,

CITY\_NAME VARCHAR(20),

PRIMARY KEY(CITY\_CODE)

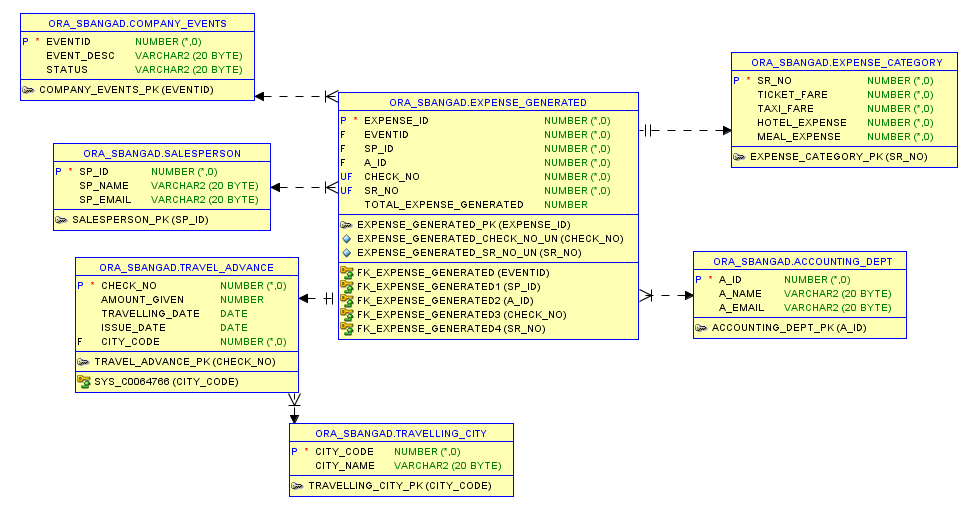
);

* By adding TRAVELLING\_CITY table, following changes in the database are required. We have to alter the travel\_advance table to add the city\_code column in it.

ALTER TABLE TRAVEL\_ADVANCE ADD CITY\_CODE INT;

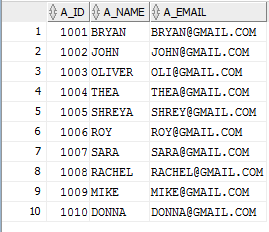
ALTER TABLE TRAVEL\_ADVANCE ADD FOREIGN KEY (CITY\_CODE) REFERENCES TRAVELLING\_CITY(CITY\_CODE);

* ENTITY RELATIONAL DIAGRAM:

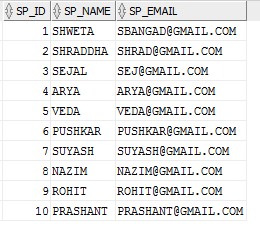


# PHASE VII: LOADING THE TABLES WITH VALID DATA

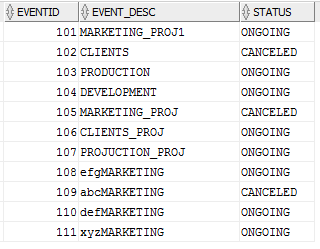
1. ACCOUNTING\_DEPT table:



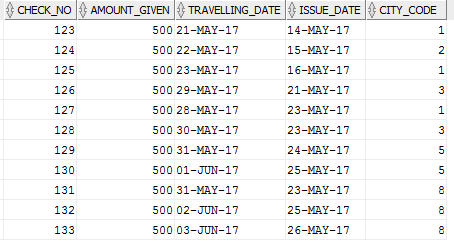
1. SALESPERSON table:



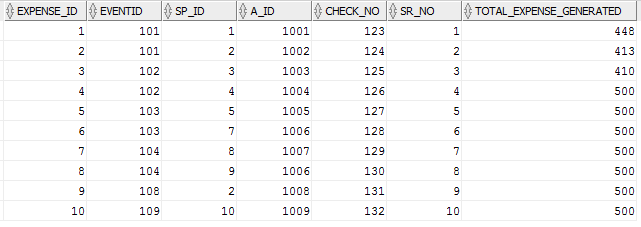
1. COMPANY\_EVENT table:



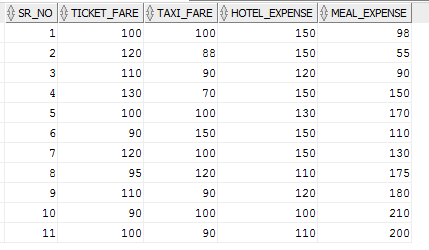
1. TRAVEL\_ADVANCE table:



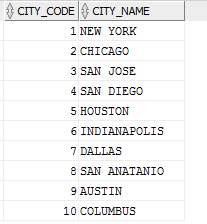
1. EXPENSE\_GENERATED table:



1. EXPENSE\_CATEGORY table:



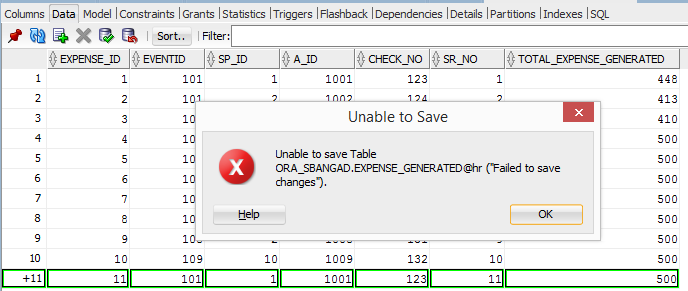
1. TRAVELLING\_CITY table:

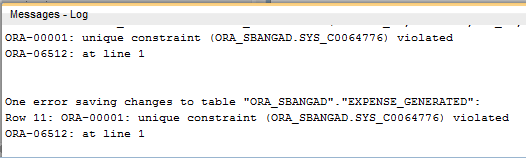


# PHASE VIII: TESTING THE DATABASE SYSTEM

1. In EXPENSE\_GENERATED table, if I would make an entry for 1st expense id with 123 check number and then again for 2nd expense id I use 123 check number then it makes an entry, but only one check number can be issued to a single salesperson for a single event.

According to the principles of database, this would make an insert anomaly. And thus, to avoid this anomaly we have to make changes in our EXPENSE\_GENERATED table. Following query will allow us to make the check number unique which would not allow us to make same entries. In this way we can test and make changes in the above table.





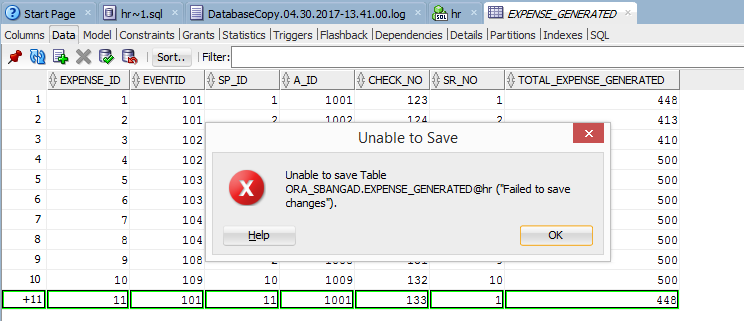
ALTER TABLE EXPENSE\_GENERATED

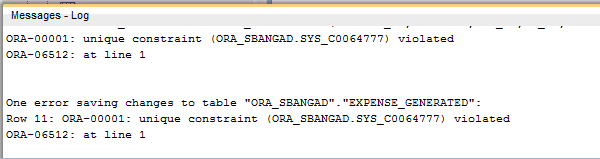
ADD UNIQUE (CHECK\_NO);

1. In EXPENSE-GENERATED table, each expense\_id is associated with one sr\_no, and this is also making an insert anomaly. So, to avoid this we need to update the the table by making this query run. Now if we are inserting the values in table with same sr\_no then it shows an error.

ALTER TABLE EXPENSE\_GENERATED

ADD UNIQUE (SR\_NO);



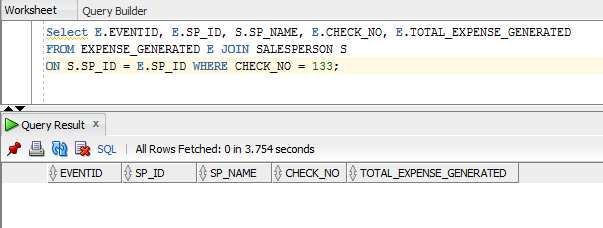


1. Selecting the data from 2 tables expense\_generated and salesperson where check number is 133, so for this there shows an empty row, because there is no row which is using check number 133.

Select E.EVENTID, E.SP\_ID, S.SP\_NAME, E.CHECK\_NO, E.TOTAL\_EXPENSE\_GENERATED

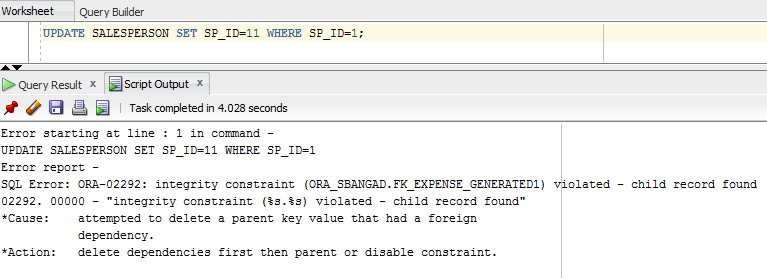
FROM EXPENSE\_GENERATED E JOIN SALESPERSON S

ON S.SP\_ID = E.SP\_ID WHERE CHECK\_NO = 133;



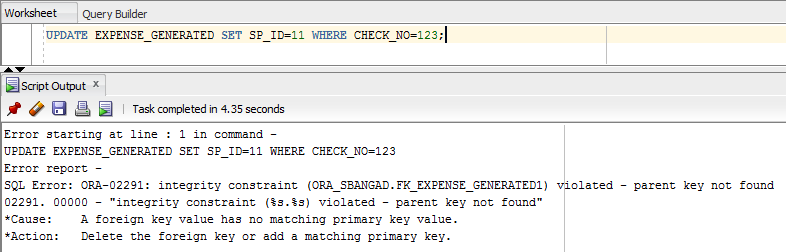
1. According to the principles of DBMS, we cannot update a column which is used as a foreign key in another table. So to avoid integrity issues, we cannot perform update query, otherwise we will have to update the same in the foreign key column as well.

UPDATE SALESPERSON SET SP\_ID=11 WHERE SP\_ID=1;



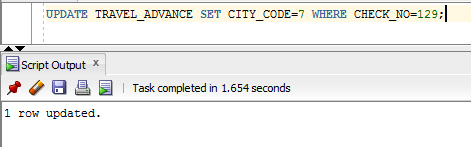
1. As check number is a unique constraint in expense\_generated table and there is no entry of sp\_id = 11 in salesperson table, so it won’t update the table. Thus this update query will not run properly.

UPDATE EXPENSE\_GENERATED SET SP\_ID=11 WHERE CHECK\_NO=123;

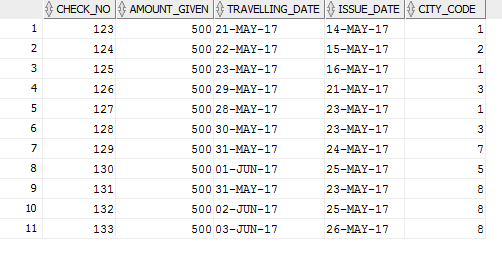


1. In this, by using update query we can update row city code by 7 of check number 129 without having to change the whole information from travelling city table. Thus if, the table has a primary key and then it is used as foreign key by another table it becomes easy to update the row.

UPDATE TRAVEL\_ADVANCE SET CITY\_CODE=7 WHERE CHECK\_NO=129;

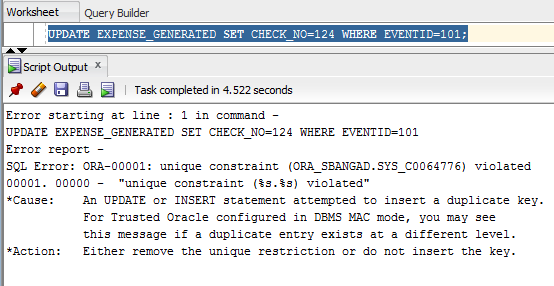


Here the city\_code is changed to 7 where check number is 129.



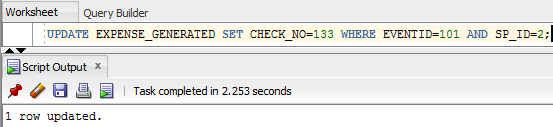
1. As check\_no is unique constraint it will not make any updates otherwise it would be an update anomaly.

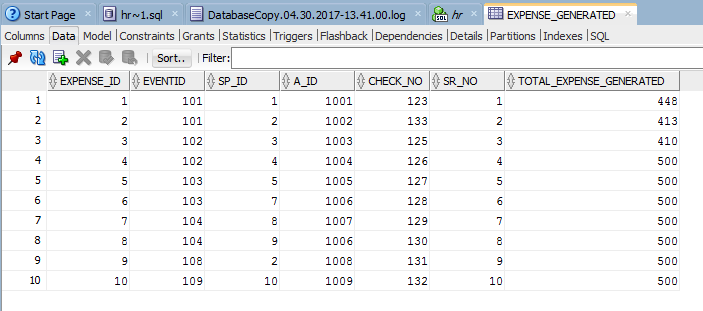
UPDATE EXPENSE\_GENERATED SET CHECK\_NO=124 WHERE EVENTID=101;



1. Here we are testing the integrity of table where we keep in mind the unique constraints, primary key, foreign key, so that there is no wrong data updated. Even if we try to update the anomaly, it will show errors.

UPDATE EXPENSE\_GENERATED SET CHECK\_NO=133 WHERE EVENTID=101 AND SP\_ID=2;





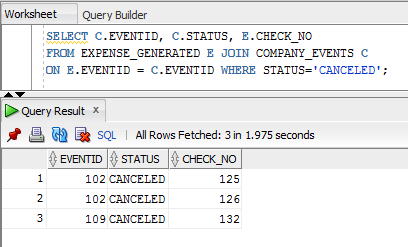
# PHASE IX: DATA ANALYTICS PERFORMED

1. Here we can perform analysis of which events in the company are cancelled and which check number has to be canceled so as to take money back from the salesperson in the company associated with canceled checks.

SELECT C.EVENTID, C.STATUS, E.CHECK\_NO

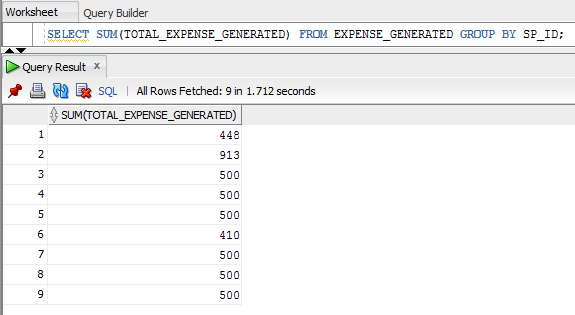
FROM EXPENSE\_GENERATED E JOIN COMPANY\_EVENTS C

ON E.EVENTID = C.EVENTID WHERE STATUS='CANCELED';



1. This will let the company analyze which sp\_id has spent how much in total for the travel.

SELECT SUM(TOTAL\_EXPENSE\_GENERATED) FROM EXPENSE\_GENERATED GROUP BY SP\_ID;

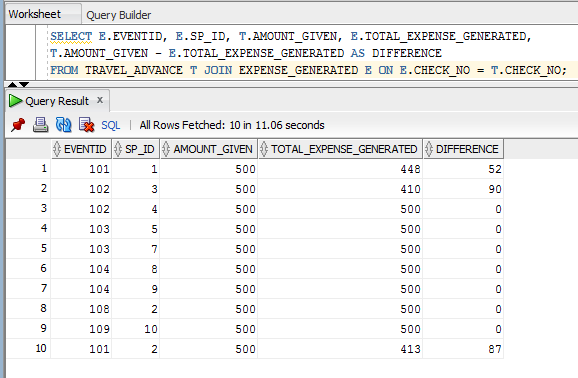


1. According to the business rules defined in previous phases, we need to know the difference between travel amount issued to salesperson and how much he spent so as to know how much money the company has to pay to salesperson or has to get return from salesperson.

SELECT E.EVENTID, E.SP\_ID, T.AMOUNT\_GIVEN, E.TOTAL\_EXPENSE\_GENERATED,

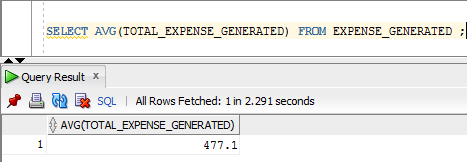
T.AMOUNT\_GIVEN - E.TOTAL\_EXPENSE\_GENERATED AS DIFFERENCE

FROM TRAVEL\_ADVANCE T JOIN EXPENSE\_GENERATED E ON E.CHECK\_NO = T.CHECK\_NO;



1. This will let us analyze the future predictions of average expenses done by each salesperson and according to this the company can decide how much money is enough for one salesperson for one travel.

SELECT AVG(TOTAL\_EXPENSE\_GENERATED) FROM EXPENSE\_GENERATED ;



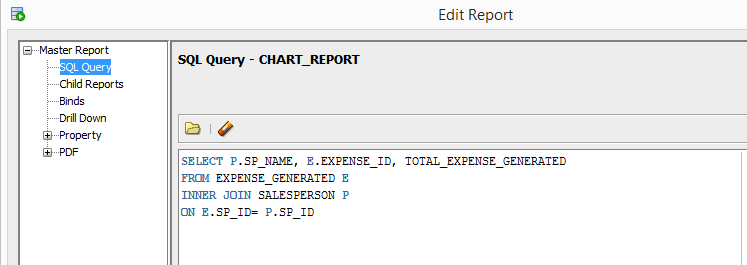
1. Report generated for the salesperson versus how much he spent for each travel will let us know about each salespersons expense details. This query will give us the series, group and value of the report.
2. I,2,3…..10 are the expense\_ids for which the values are the amount spent by salesperson named as total\_expense\_generated foreach salesperson in the company.

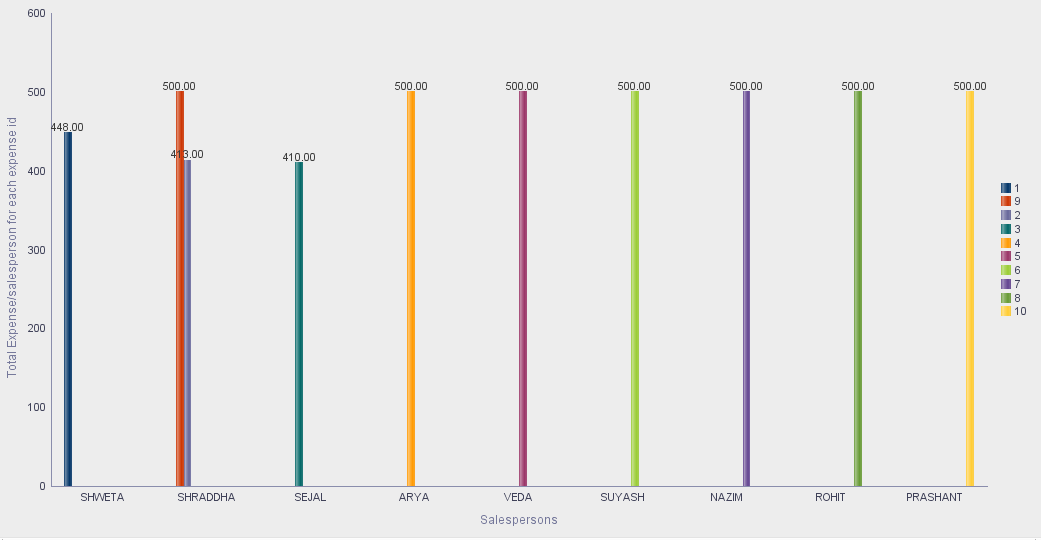
SELECT P.SP\_NAME, E.EXPENSE\_ID, TOTAL\_EXPENSE\_GENERATED

FROM EXPENSE\_GENERATED E

INNER JOIN SALESPERSON P

ON E.SP\_ID= P.SP\_ID





## WEB PRESENCE:

**Employee Login:**

<%@ Page Language = "VB" %>

<%@ Import Namespace = "System.Data.OleDb" %>

<!DOCTYPE html>

<html xmlns = "http://www.w3.org/1999/xhtml">

<head id="Head1" runat = "server">

<title>Connection</title>

<script runat = "server">

Sub Login\_Click(Src As Object, E As EventArgs)

Try

'Connect to the Database

Dim cnAccess As New OleDbConnection(

"Provider = Microsoft.Jet.OLEDB.4.0;" &

"Data Source = C:\Users\Shweta\OneDrive\Documents\TALogin.mdb")

cnAccess.Open()

Dim semp\_user\_name As String

semp\_user\_name = emp\_user\_name.Text.Trim

Dim semp\_password As String

semp\_password = emp\_password.Text.Trim

'Construct the SELECT statement

Dim sSelectSQL As String

'Create the SQL Select Statement

sSelectSQL = "SELECT \* FROM employee\_tbl WHERE emp\_user\_name= '" & semp\_user\_name & "' and emp\_password= '" & semp\_password & "'"

'Create the OleDbCommand object

Dim cmdSelect As New OleDbCommand(sSelectSQL, cnAccess)

Dim drEmp As OleDbDataReader, sbResults As New StringBuilder()

drEmp = cmdSelect.ExecuteReader()

sbResults.Append("<table>")

If drEmp.HasRows Then

Session("emp\_user\_name") = semp\_user\_name

Response.Redirect("Expense\_report.aspx")

Else

Response.Write("Username password does not match!")

End If

Catch ex As Exception

Response.Write(ex.Message)

Response.Write("Connection Failed")

End Try

End Sub

</script>

</head>

<body style = "font-family:Tahoma;">

<h3>TRAVEL ADVANCE Employee Login</h3>

<form runat = "server" id = "form1">

<table>

<tr>

</tr>

<tr>

<td>Employee username: </td>

<td><asp:Textbox id = "emp\_user\_name" runat = "server" /></td>

</tr>

<tr>

</tr>

<tr>

<td>Employee password: </td>

<td><asp:Textbox id = "emp\_password" runat = "server" /></td>

</tr>

</table>

<br />

<asp:Button Text = "Login" OnClick = "Login\_Click"

runat = "server" ID = "Button1" />

<p>

<asp:Label id = "msg" runat = "server" />

</p>

</form>

<div></div>

</body>

</html>

**Expense\_report:**

<%@ Page Language = "VB" %>

<%@ Import Namespace = "System.Data.OleDb" %>

<!DOCTYPE html>

<html xmlns = "http://www.w3.org/1999/xhtml">

<head id="Head1" runat = "server">

<title>Connection</title>

<script runat = "server">

Sub Insert\_Click(Src As Object, E As EventArgs)

Try

'Connect to the Database

Dim cnAccess As New OleDbConnection(

"Provider = Microsoft.Jet.OLEDB.4.0;" &

"Data Source = C:\Users\Shweta\OneDrive\Documents\TALogin.mdb")

cnAccess.Open()

Dim sHotel\_Expenses, sMeal\_Expenses, sTaxi\_Expenses, sTicket\_Expenses As Integer

Dim suser, sInsertSQL As String

sHotel\_Expenses = Hotel\_Expenses.Text

sMeal\_Expenses = Meal\_Expenses.Text

sTaxi\_Expenses = Taxi\_Expenses.Text

sTicket\_Expenses = Ticket\_Expenses.Text

suser = Session("emp\_user\_name")

'Constr7uct the insert statement

sInsertSQL = "INSERT INTO expense\_report(" &

"[Hotel\_Expenses], [Meal\_Expenses], [Taxi\_Expenses], [Ticket\_Expenses], [emp\_user\_name]) VALUES" &

"('" & sHotel\_Expenses & "','" & sMeal\_Expenses & "','" & sTaxi\_Expenses & "','" & sTicket\_Expenses & "','" & suser & "');"

'Construct the OleDbCommand object

Dim cmdInsert As New OleDbCommand(sInsertSQL, cnAccess)

'since this is not a query, we do not expect to return data

cmdInsert.ExecuteNonQuery()

Response.Write("Data Recorded!")

Catch ex As Exception

Response.Write(ex.Message)

Response.Write("Connection Failed")

End Try

End Sub

</script>

</head>

<body style = "font-family:Tahoma;">

<h3>Enter your Expenses for Travel</h3>

<form runat = "server" id = "form1">

<table>

<tr>

<td>Hotel Expense: </td>

<td><asp:Textbox id = "Hotel\_Expenses" runat = "server" /></td>

</tr>

<tr>

<td>Meal Expense: </td>

<td><asp:Textbox id = "Meal\_Expenses" runat = "server" /></td>

</tr>

<tr>

<td>Taxi Expense: </td>

<td><asp:Textbox id = "Taxi\_Expenses" runat = "server" /></td>

</tr>

<tr>

<td>Ticket Expense: </td>

<td><asp:Textbox id = "Ticket\_Expenses" runat = "server" /></td>

</tr>

</table>

<br />

<asp:Button Text = "Insert" OnClick = "Insert\_Click"

runat = "server" ID = "Button1" />

<p>

<asp:Label id = "msg" runat = "server" />

</p>

<br />

</form>

<div></div>

</body>

</html>

**AccountantLogin:**

<%@ Page Language = "VB" %>

<%@ Import Namespace = "System.Data.OleDb" %>

<!DOCTYPE html>

<html xmlns = "http://www.w3.org/1999/xhtml">

<head id="Head1" runat = "server">

<title>Connection</title>

<script runat = "server">

Sub Login\_Click(Src As Object, E As EventArgs)

Try

'Connect to the Database

Dim cnAccess As New OleDbConnection(

"Provider = Microsoft.Jet.OLEDB.4.0;" &

"Data Source = C:\Users\Shweta\OneDrive\Documents\TALogin.mdb")

cnAccess.Open()

Dim sAccountant\_name As String

sAccountant\_name = accountant\_name.Text.Trim

Dim sAccountant\_password As String

sAccountant\_password = accountant\_password.Text.Trim

'Construct the SELECT statement

Dim sSelectSQL As String

'Create the SQL Select Statement

sSelectSQL = "SELECT \* FROM accountant\_tbl WHERE Accountant\_name= '" & sAccountant\_name & "' and Accountant\_password= '" & sAccountant\_password & "'"

'Create the OleDbCommand object

Dim cmdSelect As New OleDbCommand(sSelectSQL, cnAccess)

Dim drEmp As OleDbDataReader, sbResults As New StringBuilder()

drEmp = cmdSelect.ExecuteReader()

sbResults.Append("<table>")

If drEmp.HasRows Then

Response.Redirect("Travel\_Advance.aspx")

Else

Response.Write("Username password does not match!")

End If

Catch ex As Exception

Response.Write(ex.Message)

Response.Write("Connection Failed")

End Try

End Sub

</script>

</head>

<body style = "font-family:Tahoma;">

<h3>Accountant Login</h3>

<form runat = "server" id = "form1">

<table>

<tr>

<td>Accountant username: </td>

<td><asp:Textbox id = "accountant\_name" runat = "server" /></td>

</tr>

<tr>

<td>Accountant password: </td>

<td><asp:Textbox id = "accountant\_password" runat = "server" /></td>

</tr>

</table>

<br />

<asp:Button Text = "Login" OnClick = "Login\_Click"

runat = "server" ID = "Button1" />

<p>

<asp:Label id = "msg" runat = "server" />

</p>

</form>

<div></div>

</body>

</html>

**Travel\_Advance**

<%@ Page Language="VB" %>

<%@ Import Namespace="System.Data.OleDb" %>

<%@ Import Namespace="System.IO" %>

<%@ Import Namespace="System.Drawing" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head id="Head1" runat="server">

<title>Connection</title>

<script runat="server">

Public Class PictureBox

Property Image As Drawing.Bitmap

End Class

Sub Search\_Click(Src As Object, E As EventArgs)

Try

'Connect to the Database

Dim cnAccess As New OleDbConnection(

"Provider = Microsoft.Jet.OLEDB.4.0;" &

"Data Source = C:\Users\Shweta\OneDrive\Documents\TALogin.mdb")

cnAccess.Open()

Dim semp\_user\_name As String

semp\_user\_name = emp\_user\_name.Text.Trim

'Construct the SELECT statement

Dim sSelectSQL As String

'Create the SQL Select Statement

sSelectSQL = "SELECT \* FROM expense\_report WHERE ([emp\_user\_name] Like '" & semp\_user\_name & "')"

'Create the OleDbCommand object

Dim cmdSelect As New OleDbCommand(sSelectSQL, cnAccess)

Dim drEmp As OleDbDataReader, sbResults As New StringBuilder()

drEmp = cmdSelect.ExecuteReader()

sbResults.Append("<table>")

Do While drEmp.Read()

sbResults.Append("<table>")

sbResults.Append("<tr><td><b>Expense ID: </b>")

sbResults.Append(drEmp.GetInt32(0))

sbResults.Append("</td></tr><tr><td><b> Total Expense: </b>")

'Response.Write(Convert.ToInt32(drEmp.GetString(1)))

sbResults.Append(Convert.ToInt32(drEmp.GetString(1)) + Convert.ToInt32(drEmp.GetString(2)) + Convert.ToInt32(drEmp.GetString(3)) + Convert.ToInt32(drEmp.GetString(4)))

sbResults.Append("")

sbResults.Append("</table>")

sbResults.Append("<br></br>")

Loop

sbResults.Append("</table>")

msg.Text = sbResults.ToString()

cnAccess.Close()

cnAccess = Nothing

If sbResults.ToString().Length < 16 Then

Response.Write("Data Not Found")

Else

Response.Write("Data Found!")

End If

Catch ex As Exception

Response.Write(ex.Message)

Response.Write("Connection Failed")

End Try

End Sub

</script>

</head>

<body style="font-family: Tahoma;">

<h3>Enter Employee Name</h3>

<form runat="server" id="form1">

<table>

<tr>

<td>Employee Name: </td>

<td>

<asp:TextBox ID="emp\_user\_name" runat="server" /></td>

</tr>

</table>

<br />

<asp:Button Text="Search" OnClick="Search\_Click"

runat="server" ID="Button1" />

<p>

<asp:Label ID="msg" runat="server" />

</p>

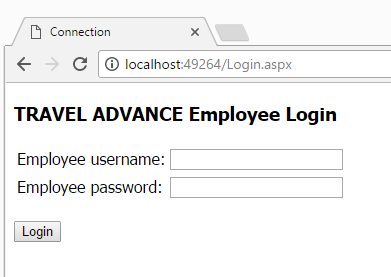
</form>

<div></div>

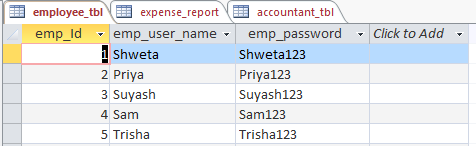
</body>

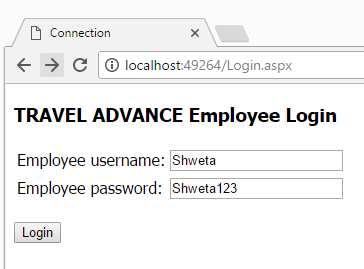
</html>

1. Employee already have their usernames and password. The login page for employees is :

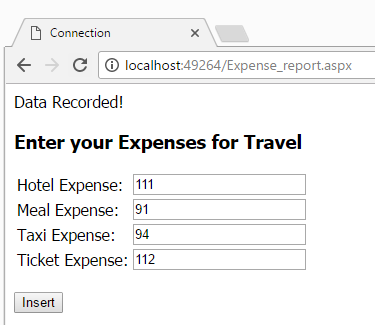


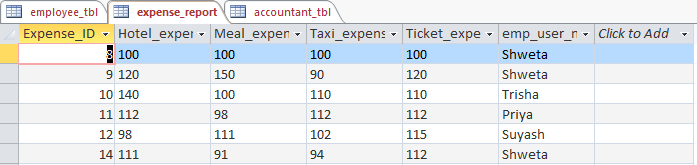
1. Username and password must match for employee and if does it is redirected to expense\_report page:



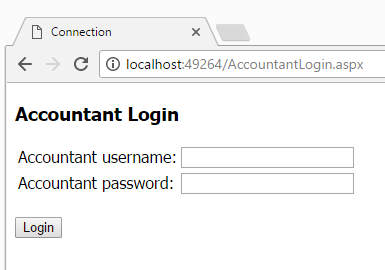


1. Expense\_report page will record expenses done by the employee who logged in and will save in the table named expense\_report shown below. Data recorded for that employee is saved.

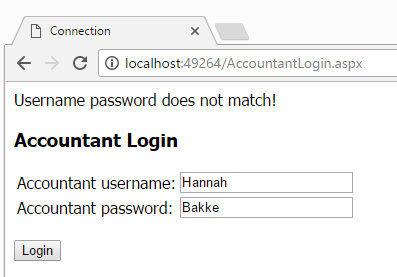




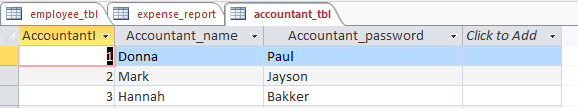
1. After employees fill their expenses, then the accountant can login and see all the records of each employee. First the accountant login page looks like this:



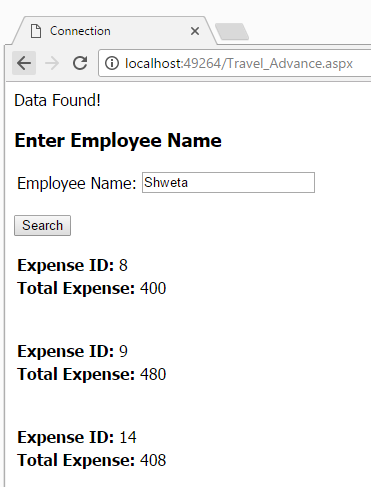
1. If the accountant enters wrong username and password it will show error.



**Accountant table:**



1. After the accountant login, he can find the employees expenses just by typing in employees name. he can see every expense done by that employee.



# PHASE X: SYSTEM ANALYSIS AND VIEWPOINTS

* The overall system of travel advance is very effective and efficient as the team has made use of proper principles, standards and rules of data ware housings as well as data analytics and the database management system.
* The travel advance system has been modified to the star schema type where there is a fact table and the dimensional tables, because of which the analytics and querying becomes easy otherwise we have to join lot of tables to create a simple analysis.
* Overall the system becomes easy to use and analyses better for further changes if needed. We have made system which gives all the reports about the salesperson , how much he has spent and how much money he was given.
* The data is handled properly as all the anomalies were removed and the testing was done properly in order to retrieve data and analyze data effectively.
* Before the company used the electronic sheets where there were confidentiality issues, integrity issues, which were all solved in the travel advance database system.
* According to the database principles our data team has made use of all the tools required for analyzing the data, for data mining, machine learning which will help to give better predictive analysis to improve the system.
* The prediction for further management of accounts can be done by using tools that will give the predictive analysis about the average money spent by the salespersons in a month. How many events generally happens in a month and by that how many events can happen in next month can be found.
* The design, implementation and management is totally focused in this system to make it effective and efficient.