



UNIVERSITI KUALA LUMPUR KAMPUS KOTA
MALAYSIAN INSTITUTE OF INFORMATION TECHNOLOGY
ASSESSMENT BRIEF

COURSE DETAILS	
CAMPUS	MIIT
COURSE NAME	ASP.NET WEB PROGRAMMING
COURSE CODE	ITD21203
COURSE LEADER	MDM HANA MUNIRA MUHD MUKHTAR (LO1)
PART-TIME LECTURER	SIR MOHD ZUHAIR MUHAMMAD KHAZANI (LO2) SIR EISOMULLAH LUKMAN (LO3) SIR EZZAT AKARUDIM SULAIMAN (LO4)
YEAR/SEMESTER	2 / 4

ASSESSMENT DETAILS	
TITLE/NAME	PROJECT
WEIGHT	20%
REQUIREMENT	1. ASP.NET <ul style="list-style-type: none"> a. Master Page b. User Controls c. Validation Controls d. Class & Object (Calculation)
PRESENTATION	WEEK 13

GROUP DETAILS	
NAME	1) MUHAMMAD IZZAT [REDACTED] [REDACTED]
GROUP	Group 8 L03 – B01

Introduction

For this project, I have chosen the QUESTION 2: which is CAR FOR SALE. This question requires an UI and a database **carforsale** to be created for CAR MANAGER to **insert** several car informations into the table. The information should then be **listed** on some webpage and able to be **viewed/filtered** by the range of cc or price. Apart from that, another web form is required for the CAR SALESMAN to **enter** their sales record such as PURCHASER's information and car details. Furthermore, create a web page that will **displays** the TOTAL SALES and will lists all cars that have been sold by the salesman.

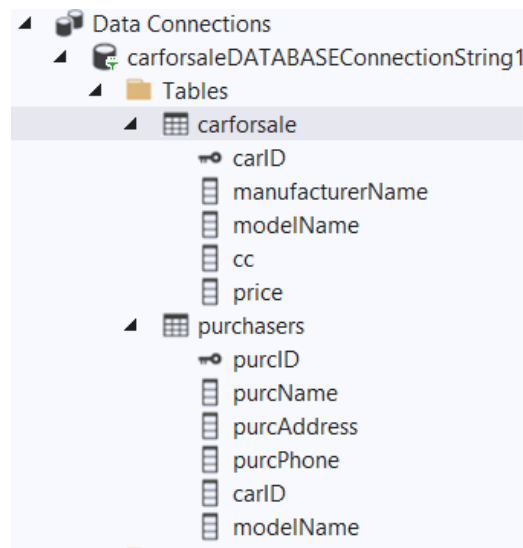
Table carforsale below shows the several information that should be inserted by the CAR MANAGER.

carID	manufacturerName	modelName	cc	price
1	Perodua	Ativa	1.0	RM 71, 200
2	Perodua	Aruz	1.5	RM 73, 266
3	Nissan	Almera	1.5	RM 92, 310
4	Nissan	X-Trail	2.5	RM 159, 888

Table carforsale

Data Structure

In my project, I've created two tables into my database. One is **carforsale** for CAR MANAGER to insert the car details, and the second is **purchasers** for CAR SALESMAN to record the purchaser's information and their car details.



For table **carforsale**, there are *carID* as PRIMARY KEY with IDENTITY(1,1), *manufacturerName*, *modelName*, *cc*, *price*. The data structure as in below:

dbo.carforsale [Design] | Update | Script File: dbo.carforsale.sql

Name	Data Type	Allow Nulls	Default
carID	int	<input type="checkbox"/>	
manufacturerName	varchar(50)	<input checked="" type="checkbox"/>	
modelName	varchar(50)	<input checked="" type="checkbox"/>	
cc	decimal(3,1)	<input checked="" type="checkbox"/>	
price	decimal(10,2)	<input checked="" type="checkbox"/>	

Keys (1)
 <unnamed> (Primary Key, Clustered: carID)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```

1 CREATE TABLE [dbo].[carforsale] (
2     [carID] INT IDENTITY (1, 1) NOT NULL,
3     [manufacturerName] VARCHAR (50) NULL,
4     [modelName] VARCHAR (50) NULL,
5     [cc] DECIMAL (3, 1) NULL,
6     [price] DECIMAL (10, 2) NULL,
7     PRIMARY KEY CLUSTERED ([carID] ASC)
8 );
9
  
```

As for table **purchasers**, there are *purcID* as a PRIMARY KEY with IDENTITY(3001, 1), *purcName*, *purcAddress*, *purcPhone*, *carID* as a FOREIGN KEY REFERENCES table **carforsale**, and *modelName*. This table is ideally created to easily link the price of bought car along with the purchaser id for calculating the total price later. Below is the data structure for the table:

dbo.purchasers [Design] x dbo.carforsale [Design]

Update Script File: dbo.purchasers.sql

	Name	Data Type	Allow Nulls	Default	
	purcID	int	<input type="checkbox"/>		
	purcName	varchar(70)	<input checked="" type="checkbox"/>		
	purcAddress	varchar(255)	<input checked="" type="checkbox"/>		
	purcPhone	int	<input checked="" type="checkbox"/>		
	carID	int	<input checked="" type="checkbox"/>		
	modelName	varchar(50)	<input checked="" type="checkbox"/>		
			<input type="checkbox"/>		

Keys (1)
 <unnamed> (Primary Key, Clustered: purcID)
Check Constraints (0)
Indexes (0)
Foreign Keys (1)
 <unnamed> (carforsale: carID)
Triggers (0)

Design T-SQL

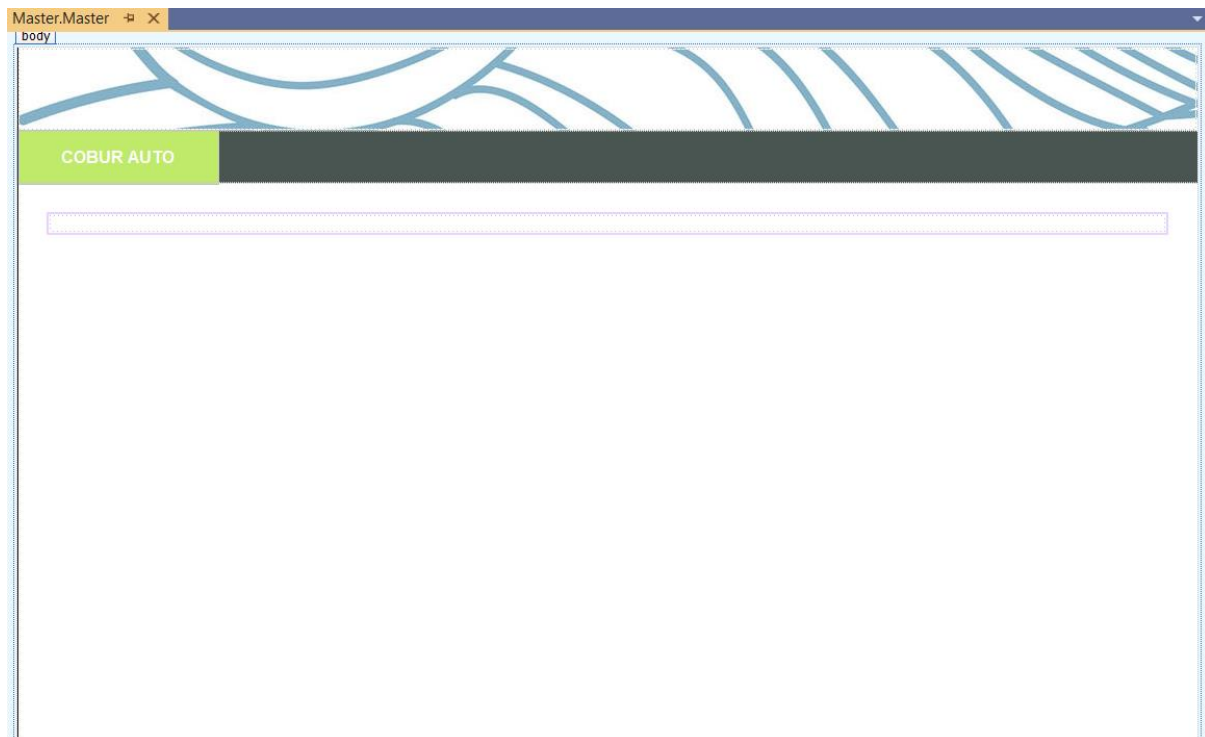
```

1  CREATE TABLE [dbo].[purchasers] (
2      [purcID]          INT          IDENTITY (3001, 1) NOT NULL,
3      [purcName]       VARCHAR (70)  NULL,
4      [purcAddress]    VARCHAR (255) NULL,
5      [purcPhone]      INT           NULL,
6      [carID]          INT           NULL,
7      [modelName]      VARCHAR (50)  NULL,
8      PRIMARY KEY CLUSTERED ([purcID] ASC),
9      FOREIGN KEY ([carID]) REFERENCES [dbo].[carforsale] ([carID])
10 );

```

Master Page

In Master page, I simply put a picture in the *header*, coloured the *content* background with teal gradient to make the page looks pleasant, then there is a *navigation menu* in the menu section with my made up car dealership's name COBUR AUTO.



Four options/navigations will appear once user hovered onto the navigation COBUR AUTO to redirect user to each four different pages. The four options are Overview, Total Sales, Salesman Page, and Manager Page.



Overview Page

This page will show the tables of data of both carforsale and purchasers that has been inserted by Manager and/or Salesman.



COBUR AUTO

Overview

CAR FOR SALE

carID	manufacturerName	modelName	cc	price
1	Perodua	Ativa	1.0	71200.00
2	Perodua	Arus	1.5	73266.00
3	Nissan	Almera	1.5	92310.00
4	Nissan	X-Trail	2.5	159888.00

PURCHASER DATA

purcID	purcName	purcAddress	purcPhone	carID	modelName
3001	Ammar Danish	No 9 Rah Residensi	1231241567	1	Ativa
3002	Saleem Chong	Jalan Inang 49 Taman Meru	1922353641	4	X-Trail

There is nothing much to do in this page as the page is only to view data to Manager or Salesman.

SQL to show table carforsale

SELECT statement:

```
SELECT [carID], [manufacturerName], [modelName], [cc], [price] FROM [carforsale]
```

SQL to show table purchasers

SELECT statement:

```
SELECT [purcID], [purcName], [purcAddress], [purcPhone], [carID], [modelName] FROM [purchasers]
```

Manager Page

This page is used by the Car Manager to *insert* a new car details into database table carforsale. The field that can be inserted is manufacturerName, modelName, cc, and price. Other than that, Manager can also *update* a new price of the existing car (only price that can be updated as we cannot change the specs of the car). This page will also shows the carforsale table so the manager can keep track of the data as manager inserting them.

manager.page

CAR TABLE

carID	manufacturerName	modelName	cc	price
1	Perodua	Ativa	1.0	71200.00
2	Perodua	Aruz	1.5	73266.00
3	Nissan	Almera	1.5	92310.00
4	Nissan	X-Trail	2.5	159888.00

Refresh Table

NEW CAR DETAILS

Manufacturer Name

Model Name

Cubic Centimetres (cc)

Car Price

* indicates required fields

Enter

UPDATE CAR PRICE

Car ID

New Price

* indicates required fields

Update Price

ENTER button

This button will execute sql to insert certain car details without bothering the primary key

```
protected void btnEnter_Click(object sender, EventArgs e)
{
    string sql;
    string manName, modelName;
    decimal cc, price;

    manName = txtMan.Text;
    modelName = txtModel.Text;
    cc = Convert.ToDecimal(txtCC.Text);
    price = Convert.ToDecimal(txtPrice.Text);

    sql = "insert into carforsale (manufacturerName, modelName, cc, price) " +
        "values('" + manName + "', '" + modelName + "', " + cc + ", " + price + ")";

    executeSQL(sql, "insert");
}
```

UPDATE button

This button will execute sql to update price field into the existing car ID

```
protected void btnUpdate_Click(object sender, EventArgs e)
{
    string sql;
    int carID;
    decimal newPrice;

    carID = Convert.ToInt32(txtCarID.Text);
    newPrice = Convert.ToDecimal(txtUpdatePrice.Text);

    sql = "update carforsale set price = '" + newPrice + "' where carID = '" + carID + "'";

    executeSQL(sql, "update");
}
```


For validation, it is divided into two sections which are group1 and group2. Validation group1 is to validate the NEW CAR DETAILS section and group 2 is for UPDATE CAR PRICE section.

NEW CAR DETAILS

NEW CAR DETAILS

Manufacturer Name
 *

Model Name
 *

Cubic Centimetres (cc)
 *

Car Price
 *

* indicates required fields

Required field validator is set on all of the field, it would only shows star(*) with reminder below so that another related validator can be fit as well.

NEW CAR DETAILS

Manufacturer Name

Model Name

Cubic Centimetres (cc)

 enter cc from 1.0 to 4.0

Car Price

 enter valid price

* indicates required fields

Range validator is set on two of them:

- 1) Cubic Centimetres –
Minimum 1.0 with Maximum 4.0
- 2) Car Price –
Minimum 0, Maximum 1000000

UPDATE CAR PRICE

UPDATE CAR PRICE

Car ID
 *

New Price
 *

* indicates required fields

Required field validator been set on all of the field, it would only shows star(*) with indication below so that another related validator can fit as well.

UPDATE CAR PRICE

Car ID

enter valid car ID

New Price

enter valid price

* indicates required fields

Range validator is set on two of them:

- 1) Car ID –
Minimum 1 with Maximum 100
- 2) Car Price –
Minimum 0, Maximum 1000000

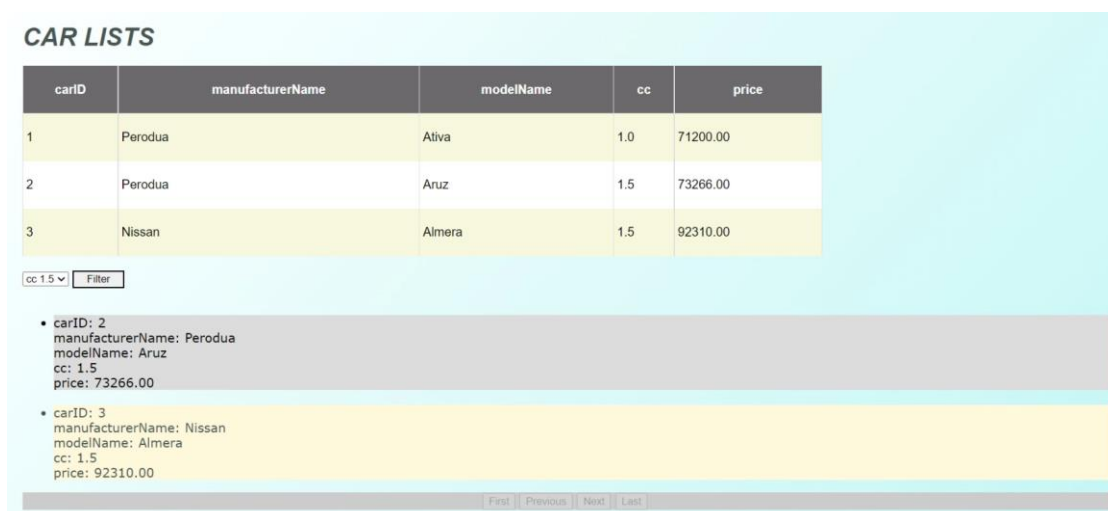
Once the inputs has passed the validator, the entered values will be filled into the table carforsale.

Salesman Page

This page is used by Car Salesman to view the list of cars that have been entered by the Manager. The Salesman can filter to find cars by the range of cc. The filtered car will be shown in ListView1.



Filter button



Code behind page for filter button:

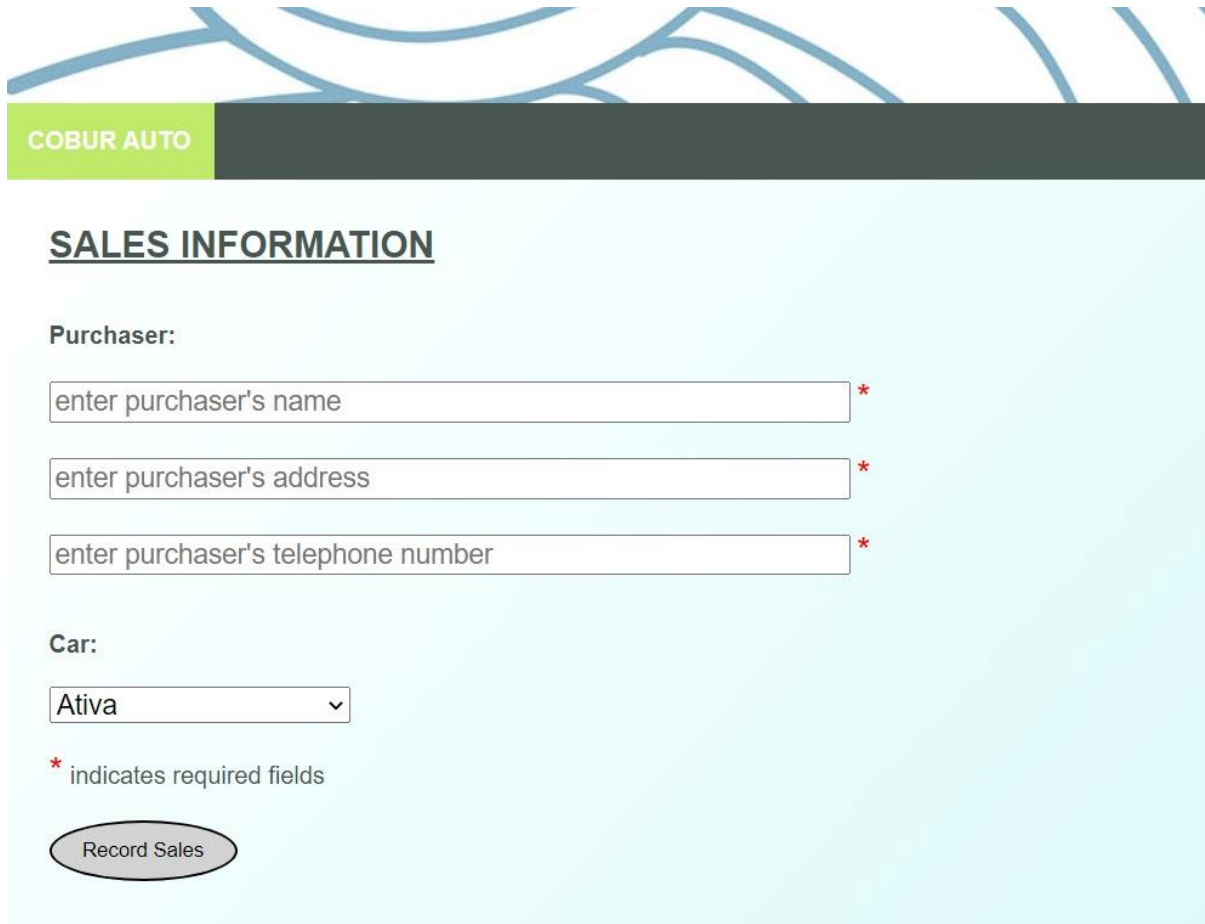
```
protected void btnFilter_Click(object sender, EventArgs e)
{
    string sql;
    SqlCommand command;
    DataTable dt = new DataTable();
    decimal cc = Convert.ToDecimal(ddlFilter.SelectedValue);

    sql = "select * from dbo.carforsale where cc = '" + cc + "' ";

    connection.Open();
    command = new SqlCommand(sql, connection);
    SqlDataAdapter da = new SqlDataAdapter(command);
    da.Fill(dt);
    command.ExecuteNonQuery();

    if (dt.Rows.Count > 0)
    {
        ListView1.DataSource = dt;
        ListView1.DataBind();
    }
    else
    {
        Response.Write("No Record");
    }
    command.Dispose();
    connection.Close();
}
```

Salesman can then click the `Record Sales` on **Salesman page** button to redirect to the **Sales Info page** to record sales made. The record will contain purcName, purcAddress, purcPhone, and Car Details (CarID, modelName).



COBUR AUTO

SALES INFORMATION

Purchaser:

enter purchaser's name *

enter purchaser's address *

enter purchaser's telephone number *

Car:

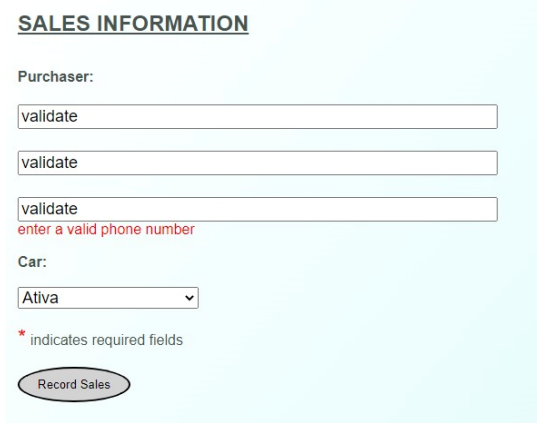
Ativa ▾

* indicates required fields

Record Sales

Several validator has been set onto those fields;

- 1) Required Field Validator:
 - All of the textboxes
- 2) Regular Expression Validator:
 - txtphone = 11 digits



SALES INFORMATION

Purchaser:

validate

validate

validate
enter a valid phone number

Car:

Ativa ▾

* indicates required fields

Record Sales

Once the inputs passed all the validator, Salesman may then click on the `Record Sales` to record all data into table purchasers and to redirect Salesman to the **Total Sales page**.

Record Sales button

This button will execute sql to insert purchasers data without bothering the purcID

```
protected void btnRecord_Click(object sender, EventArgs e)
{
    string sql;
    string name, address, modelName;
    int phone, carID;

    name = txtName.Text;
    address = txtAddress.Text;
    phone = Convert.ToInt32(txtPhone.Text);
    carID = Convert.ToInt32(ddlCar.SelectedValue);
    modelName = ddlCar.SelectedItem.Text;

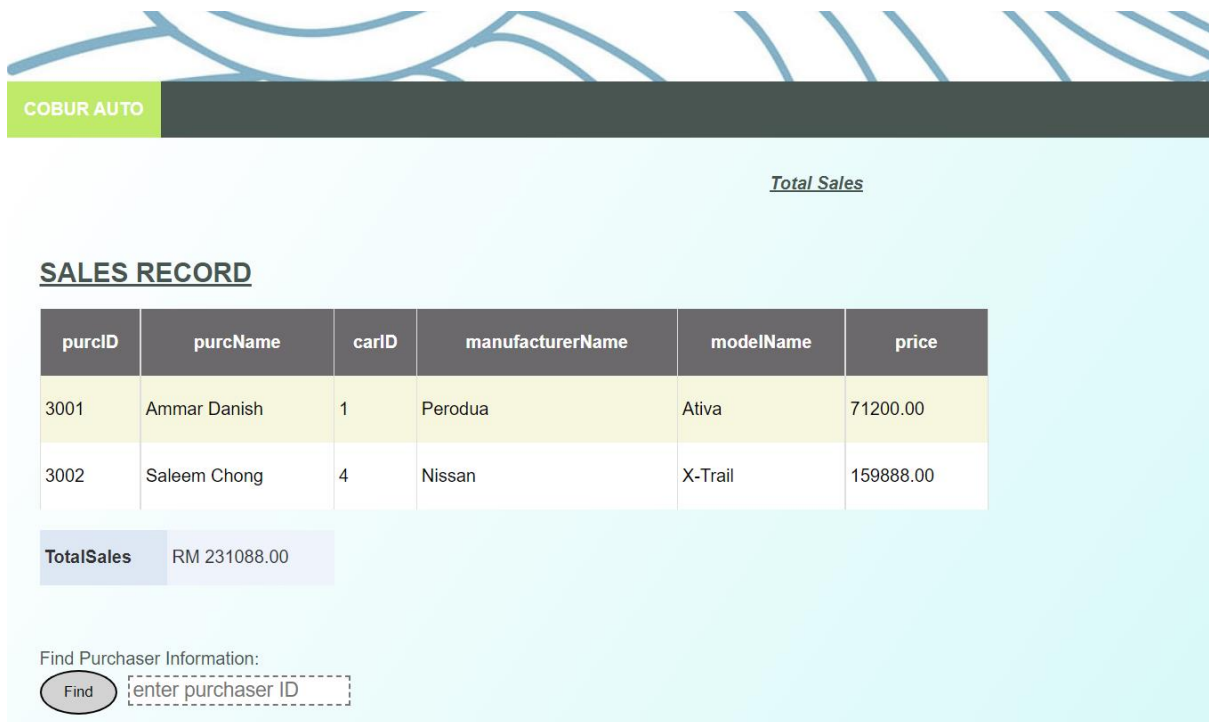
    sql = "insert into purchasers (purcName, purcAddress, purcPhone, carID, modelName) values " +
        "(" + name + ", " + address + ", " + phone + ", " + carID + ", " + modelName + ")";

    executeSQL(sql, "insert");

    Response.Redirect("TotalSales.aspx");
}
```

Total Sales Page

This page contains table of purchasers data along with the car price and the total sales made by salesman below the table.



COBUR AUTO

Total Sales

SALES RECORD

purcID	purcName	carID	manufacturerName	modelName	price
3001	Ammar Danish	1	Perodua	Ativa	71200.00
3002	Saleem Chong	4	Nissan	X-Trail	159888.00

TotalSales RM 231088.00

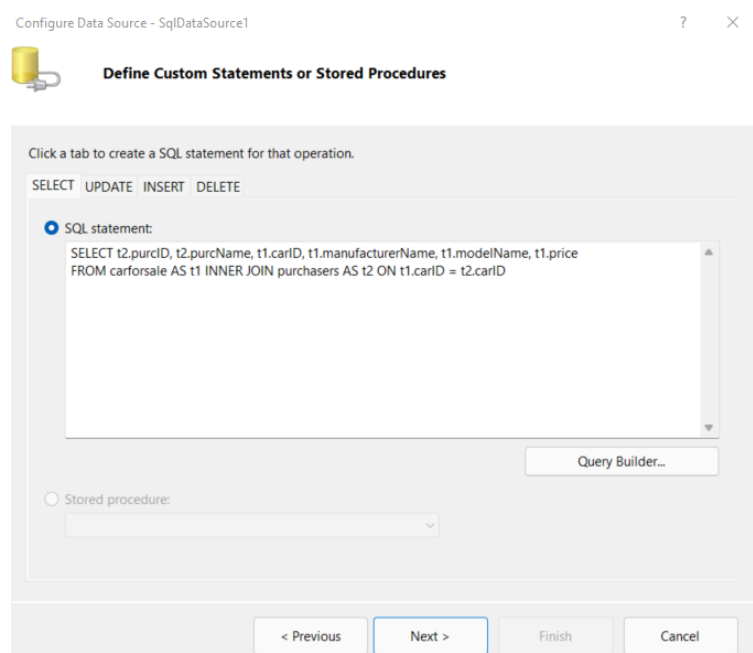
Find Purchaser Information:

Find

----->

SQL used to show the table of purchasers that also joined by carPrice from table carforsale.

! purcAddress and purcPhone will not be shown.



Configure Data Source - SqlDataSource1

Define Custom Statements or Stored Procedures

Click a tab to create a SQL statement for that operation.

SELECT UPDATE INSERT DELETE

☒ SQL statement:

```
SELECT t2.purcID, t2.purcName, t1.carID, t1.manufacturerName, t1.modelName, t1.price
FROM carforsale AS t1 INNER JOIN purchasers AS t2 ON t1.carID = t2.carID
```

☐ Stored procedure:

Query Builder...

< Previous Next > Finish Cancel

Configure Data Source - SqlDataSource2

Define Custom Statements or Stored Procedures

Click a tab to create a SQL statement for that operation.

SELECT UPDATE INSERT DELETE

☒ SQL statement:

```
SELECT FORMAT(SUM(t1.price), 'RM0.00') AS TotalSales
FROM carforsale AS t1
INNER JOIN purchasers AS t2 ON t1.carID = t2.carID
```

Query Builder...

☐ Stored procedure:

< Previous Next > Finish Cancel

←-----

SQL used to show the total price of sales made by Salesman

The Find button is used to find the details of the purchasers, which in this case is purcAddress and purcPhone as it is not shown on the table above

Find Purchaser Information:

Find validate * enter valid ID

range validator is set on the field with minimum 3000 and maximum 3999

Find Purchaser Information:

Find 3001

purcID	3001
purcName	Ammar Danish
purcAddress	No 9 Rah Residensi
purcPhone	1231241567
modelName	Ativa

Code behind page for Find button

```
protected void btnFind_Click(object sender, EventArgs e)
{
    string sql;
    SqlCommand command;
    DataTable dt = new DataTable();
    int purcID = Convert.ToInt32(txtFind.Text);

    sql = "select * from dbo.purchasers where purcID = '" + purcID + "' ";

    connection.Open();
    command = new SqlCommand(sql, connection);
    SqlDataAdapter da = new SqlDataAdapter(command);
    da.Fill(dt);
    command.ExecuteNonQuery();

    if (dt.Rows.Count > 0)
    {
        DetailsView1.DataSource = dt;
        DetailsView1.DataBind();
    }
    else
    {
        Response.Write("No Record");
    }
    command.Dispose();
    connection.Close();
}
```

THANK YOU