GROUP 09

BICYCLE RENTAL SYSTEM

SUMMARY OF CONTENTS

Physical Data Model

Physical Process Model

DBMS Screenshots

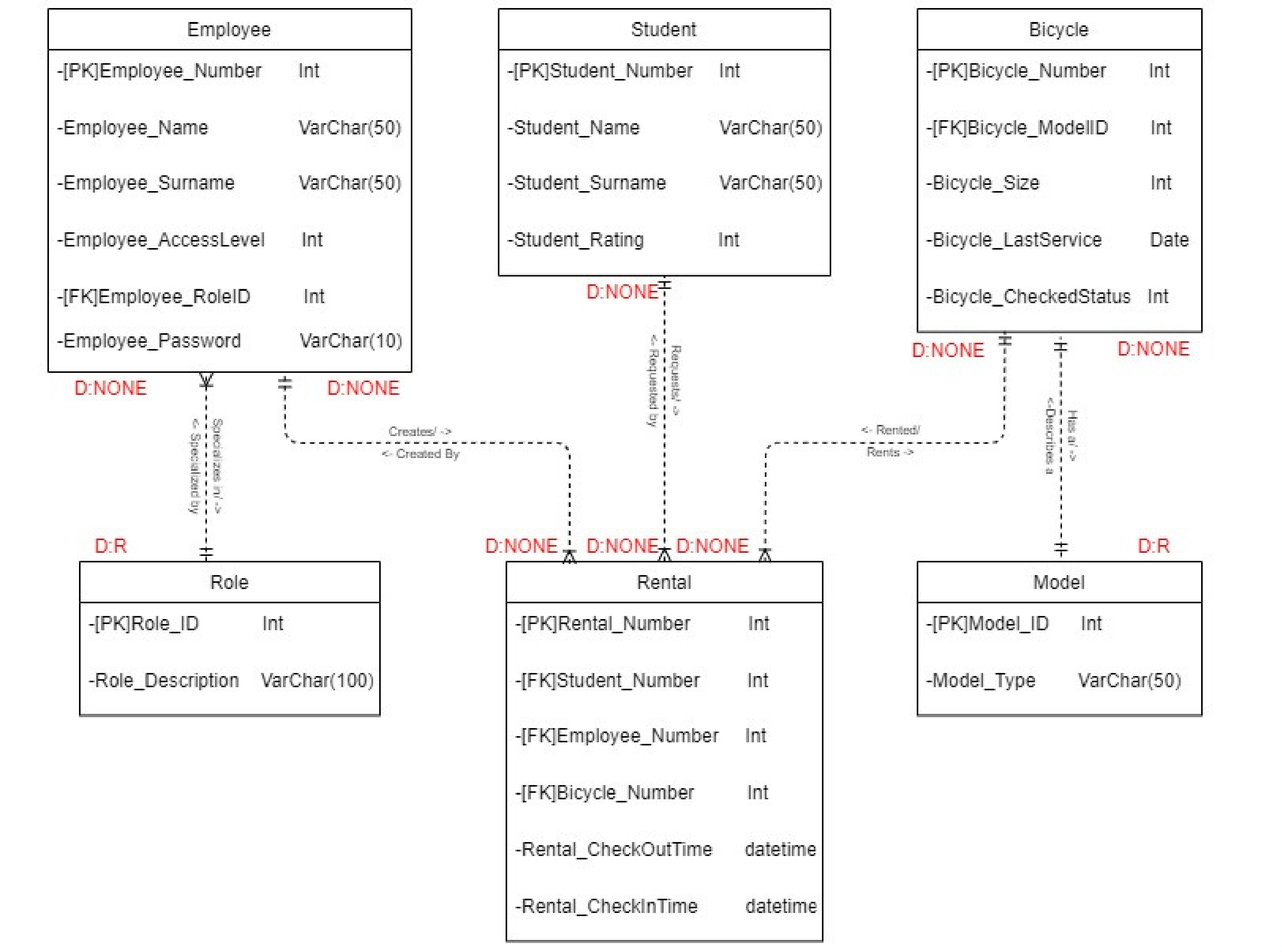
Example Code

Reports

User Manual

Detailed Diary

Physical Data Model



Employee	
-[PK]Employee_Number	Int
-Employee_Name	VarChar(50)
-Employee_Surname	VarChar(50)
-Employee_AccessLevel	Int
-[FK]Employee_RoleID	Int
-Employee_Password	VarChar(10)

Student	
-[PK]Student_Number	Int
-Student_Name	VarChar(50)
-Student_Surname	VarChar(50)
-Student_Rating	Int

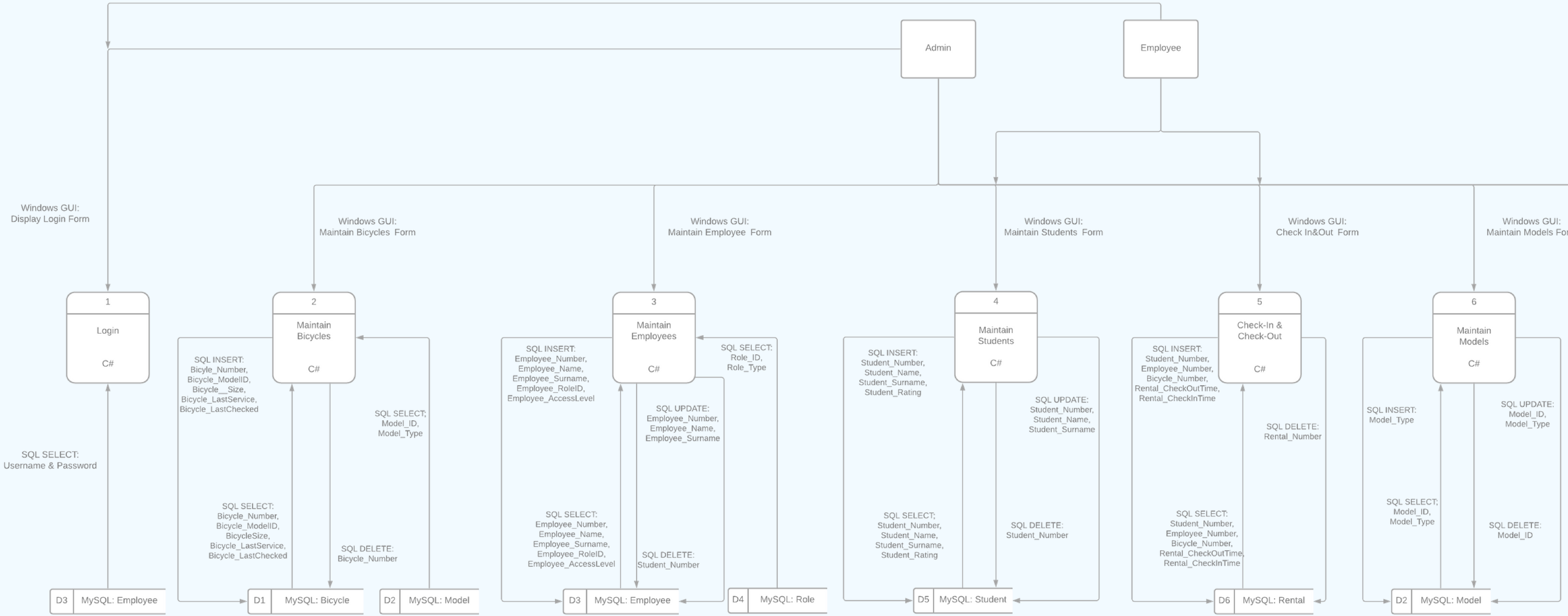
Bicycle	
-[PK]Bicycle_Number	Int
-[FK]Bicycle_ModelID	Int
-Bicycle_Size	Int
-Bicycle_LastService	Date
-Bicycle_CheckedStatus	Int

Role	
-[PK]Role_ID	Int
-Role_Description	VarChar(100)

Rental	
-[PK]Rental_Number	Int
-[FK]Student_Number	Int
-[FK]Employee_Number	Int
-[FK]Bicycle_Number	Int
-Rental_CheckOutTime	datetime
-Rental_CheckInTime	datetime

Model	
-[PK]Model_ID	Int
-Model_Type	VarChar(50)

Physical Process Model



Admin

Employee

Windows GUI:
Display Login Form

1
Login
C#

SQL SELECT:
Username & Password

D3 MySQL: Employee

Windows GUI:
Maintain Bicycles Form

2
Maintain Bicycles
C#

SQL INSERT:
Bicyle_Number,
Bicycle_ModelID,
Bicycle__Size,
Bicycle_LastService,
Bicycle_LastChecked

SQL SELECT:
Bicycle_Number,
Bicycle_ModelID,
BicycleSize,
Bicycle_LastService,
Bicycle_LastChecked

SQL DELETE:
Bicycle_Number

SQL SELECT;
Model_ID,
Model_Type

D1 MySQL: Bicycle

D2 MySQL: Model

Windows GUI:
Maintain Employee Form

3
Maintain Employees
C#

SQL INSERT:
Employee_Number,
Employee_Name,
Employee_Surname,
Employee_RoleID,
Employee_AccessLevel

SQL SELECT:
Employee_Number,
Employee_Name,
Employee_Surname,
Employee_RoleID,
Employee_AccessLevel

SQL UPDATE:
Employee_Number,
Employee_Name,
Employee_Surname

SQL DELETE:
Student_Number

SQL SELECT:
Role_ID,
Role_Type

D3 MySQL: Employee

D4 MySQL: Role

Windows GUI:
Maintain Students Form

4
Maintain Students
C#

SQL INSERT:
Student_Number,
Student_Name,
Student_Surname,
Student_Rating

SQL SELECT;
Student_Number,
Student_Name,
Student_Surname,
Student_Rating

SQL UPDATE:
Student_Number,
Student_Name,
Student_Surname

SQL DELETE:
Student_Number

D5 MySQL: Student

Windows GUI:
Check In&Out Form

5
Check-In & Check-Out
C#

SQL INSERT:
Student_Number,
Employee_Number,
Bicycle_Number,
Rental_CheckOutTime,
Rental_CheckInTime

SQL SELECT:
Student_Number,
Employee_Number,
Bicycle_Number,
Rental_CheckOutTime,
Rental_CheckInTime

SQL DELETE:
Rental_Number

D6 MySQL: Rental

Windows GUI:
Maintain Models Fo

6
Maintain Models
C#

SQL INSERT:
Model_Type

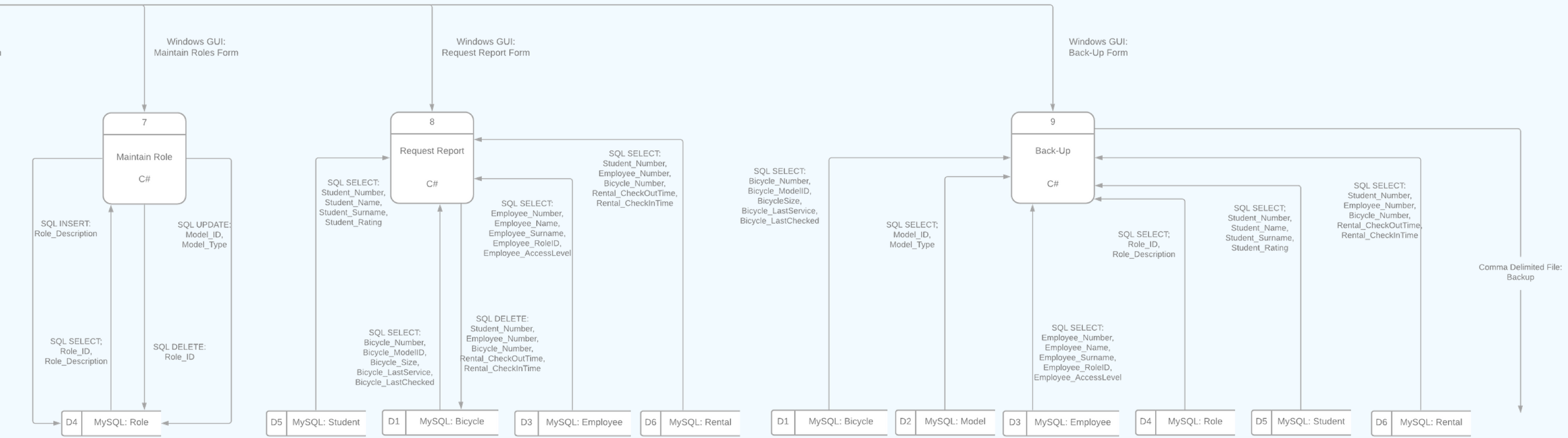
SQL SELECT;
Model_ID,
Model_Type

SQL UPDATE:
Model_ID,
Model_Type

SQL DELETE:
Model_ID

D2 MySQL: Model

Physical Process Model Cont



Windows GUI:
Maintain Roles Form

7
Maintain Role
C#

SQL INSERT:
Role_Description

SQL SELECT;
Role_ID,
Role_Description

SQL DELETE:
Role_ID

SQL UPDATE:
Model_ID,
Model_Type

D4 MySQL: Role

Windows GUI:
Request Report Form

8
Request Report
C#

SQL SELECT:
Student_Number,
Student_Name,
Student_Surname,
Student_Rating

SQL SELECT:
Bicycle_Number,
Bicycle_ModelID,
Bicycle_Size,
Bicycle_LastService,
Bicycle_LastChecked

SQL DELETE:
Student_Number,
Employee_Number,
Bicycle_Number,
Rental_CheckOutTime,
Rental_CheckInTime

SQL SELECT:
Employee_Number,
Employee_Name,
Employee_Surname,
Employee_RoleID,
Employee_AccessLevel

SQL SELECT:
Student_Number,
Employee_Number,
Bicycle_Number,
Rental_CheckOutTime,
Rental_CheckInTime

D5 MySQL: Student

D1 MySQL: Bicycle

D3 MySQL: Employee

D6 MySQL: Rental

Windows GUI:
Back-Up Form

9
Back-Up
C#

SQL SELECT:
Bicycle_Number,
Bicycle_ModelID,
BicycleSize,
Bicycle_LastService,
Bicycle_LastChecked

SQL SELECT;
Model_ID,
Model_Type

SQL SELECT:
Employee_Number,
Employee_Name,
Employee_Surname,
Employee_RoleID,
Employee_AccessLevel

SQL SELECT;
Role_ID,
Role_Description

SQL SELECT;
Student_Number,
Student_Name,
Student_Surname,
Student_Rating

SQL SELECT:
Student_Number,
Employee_Number,
Bicycle_Number,
Rental_CheckOutTime,
Rental_CheckInTime

Comma Delimited File:
Backup

D1 MySQL: Bicycle

D2 MySQL: Model

D3 MySQL: Employee

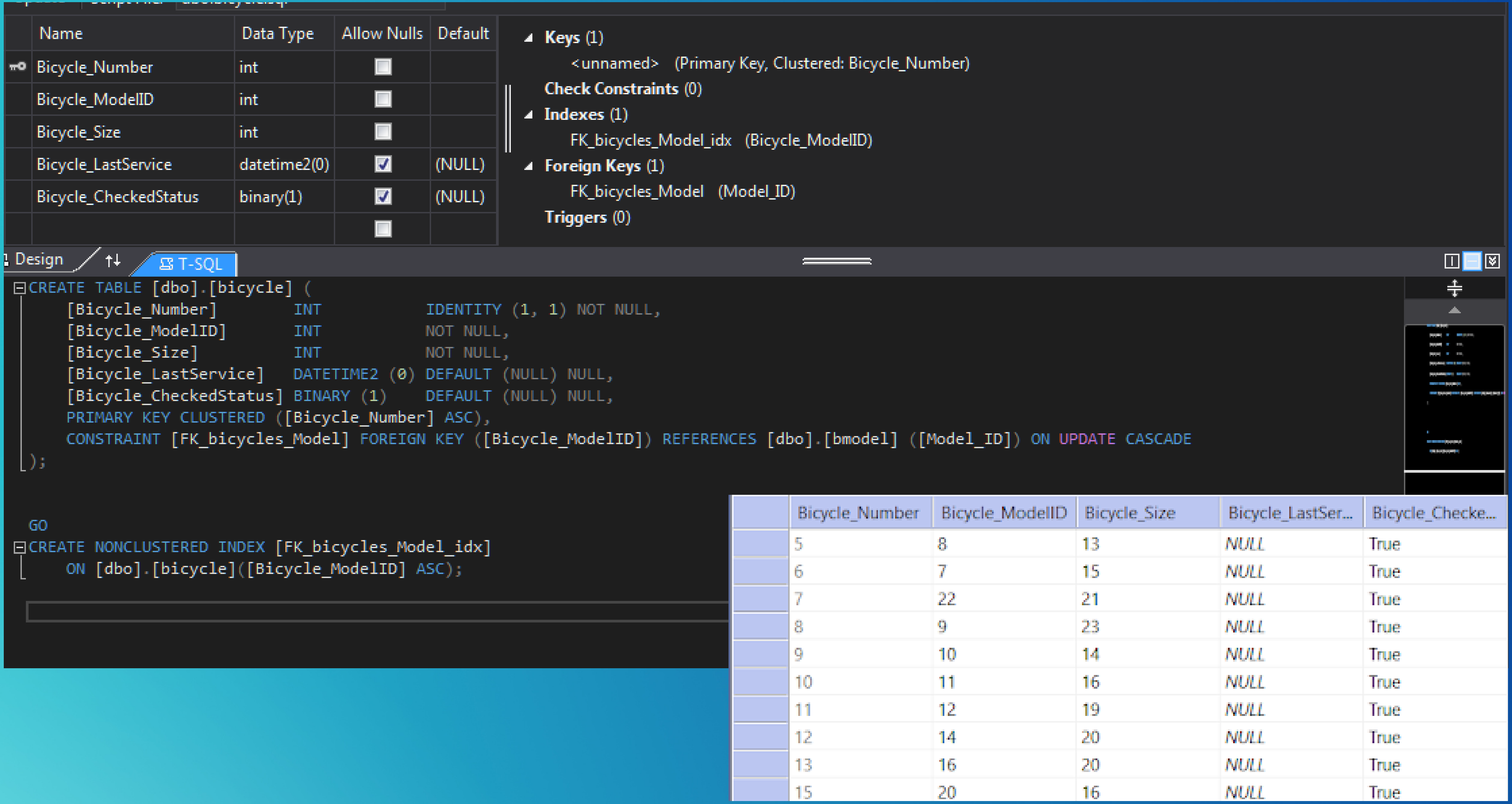
D4 MySQL: Role

D5 MySQL: Student

D6 MySQL: Rental

DBMS SCREENSHOTS

1)EMPLOYEE TABLE



	Name	Data Type	Allow Nulls	Default
🔑	Bicycle_Number	int	☐	
	Bicycle_ModelID	int	☐	
	Bicycle_Size	int	☐	
	Bicycle_LastService	datetime2(0)	☑	(NULL)
	Bicycle_CheckedStatus	binary(1)	☑	(NULL)
			☐	

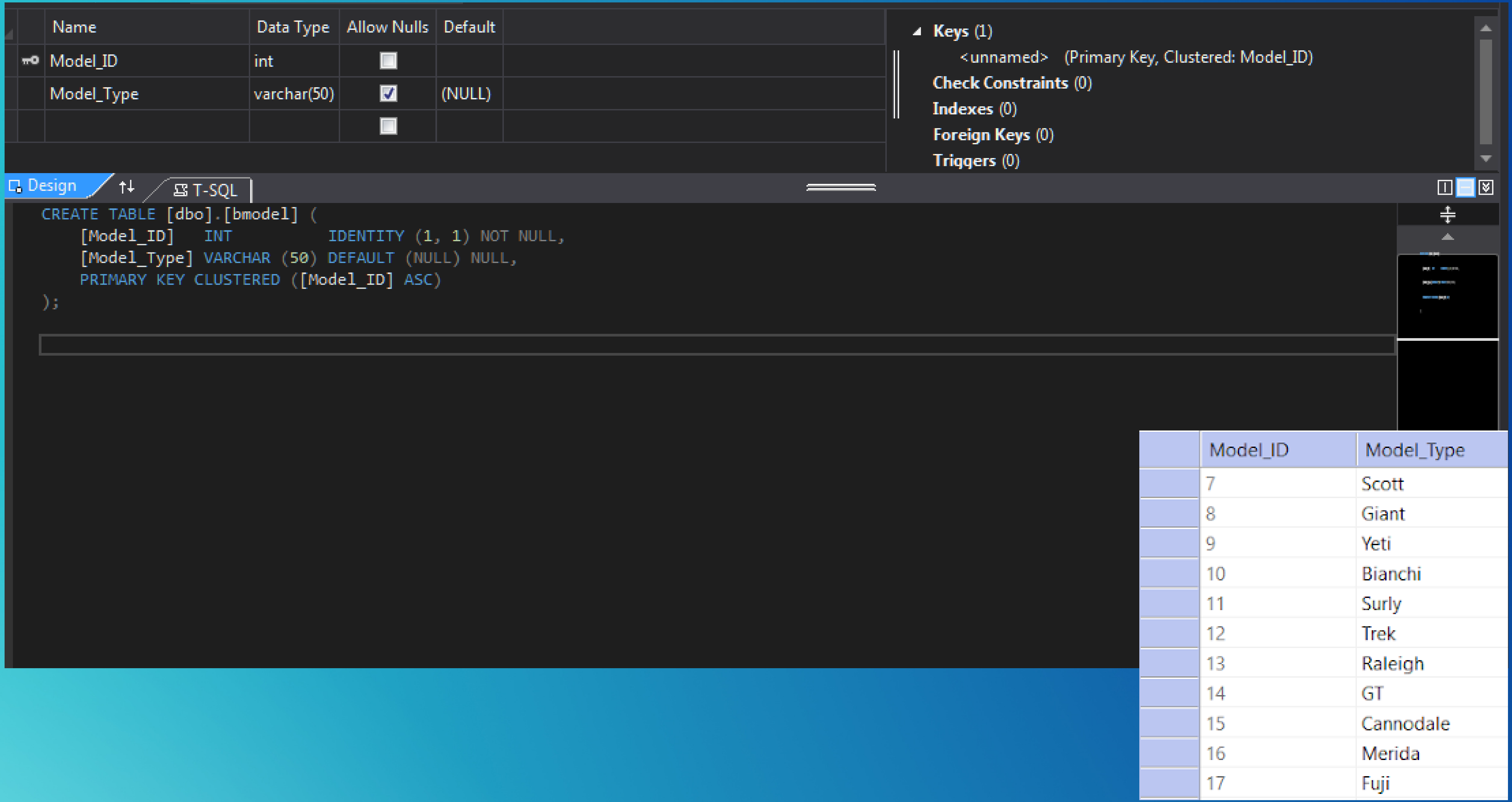
Keys (1)
<unnamed> (Primary Key, Clustered: Bicycle_Number)
Check Constraints (0)
Indexes (1)
FK_bicycles_Model_idx (Bicycle_ModelID)
Foreign Keys (1)
FK_bicycles_Model (Model_ID)
Triggers (0)

```
CREATE TABLE [dbo].[bicycle] (
    [Bicycle_Number]        INT           IDENTITY (1, 1) NOT NULL,
    [Bicycle_ModelID]       INT           NOT NULL,
    [Bicycle_Size]          INT           NOT NULL,
    [Bicycle_LastService]   DATETIME2 (0) DEFAULT (NULL) NULL,
    [Bicycle_CheckedStatus] BINARY (1)    DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Bicycle_Number] ASC),
    CONSTRAINT [FK_bicycles_Model] FOREIGN KEY ([Bicycle_ModelID]) REFERENCES [dbo].[bmodel] ([Model_ID]) ON UPDATE CASCADE
);


GO
CREATE NONCLUSTERED INDEX [FK_bicycles_Model_idx]
    ON [dbo].[bicycle]([Bicycle_ModelID] ASC);
```

Bicycle_Number	Bicycle_ModelID	Bicycle_Size	Bicycle_LastSer...	Bicycle_Checke...
5	8	13	NULL	True
6	7	15	NULL	True
7	22	21	NULL	True
8	9	23	NULL	True
9	10	14	NULL	True
10	11	16	NULL	True
11	12	19	NULL	True
12	14	20	NULL	True
13	16	20	NULL	True
15	20	16	NULL	True

DBMS SCREENSHOTS

2)MODEL TABLE



	Name	Data Type	Allow Nulls	Default
🔑	Model_ID	int	□	
	Model_Type	varchar(50)	☑	(NULL)
			□	

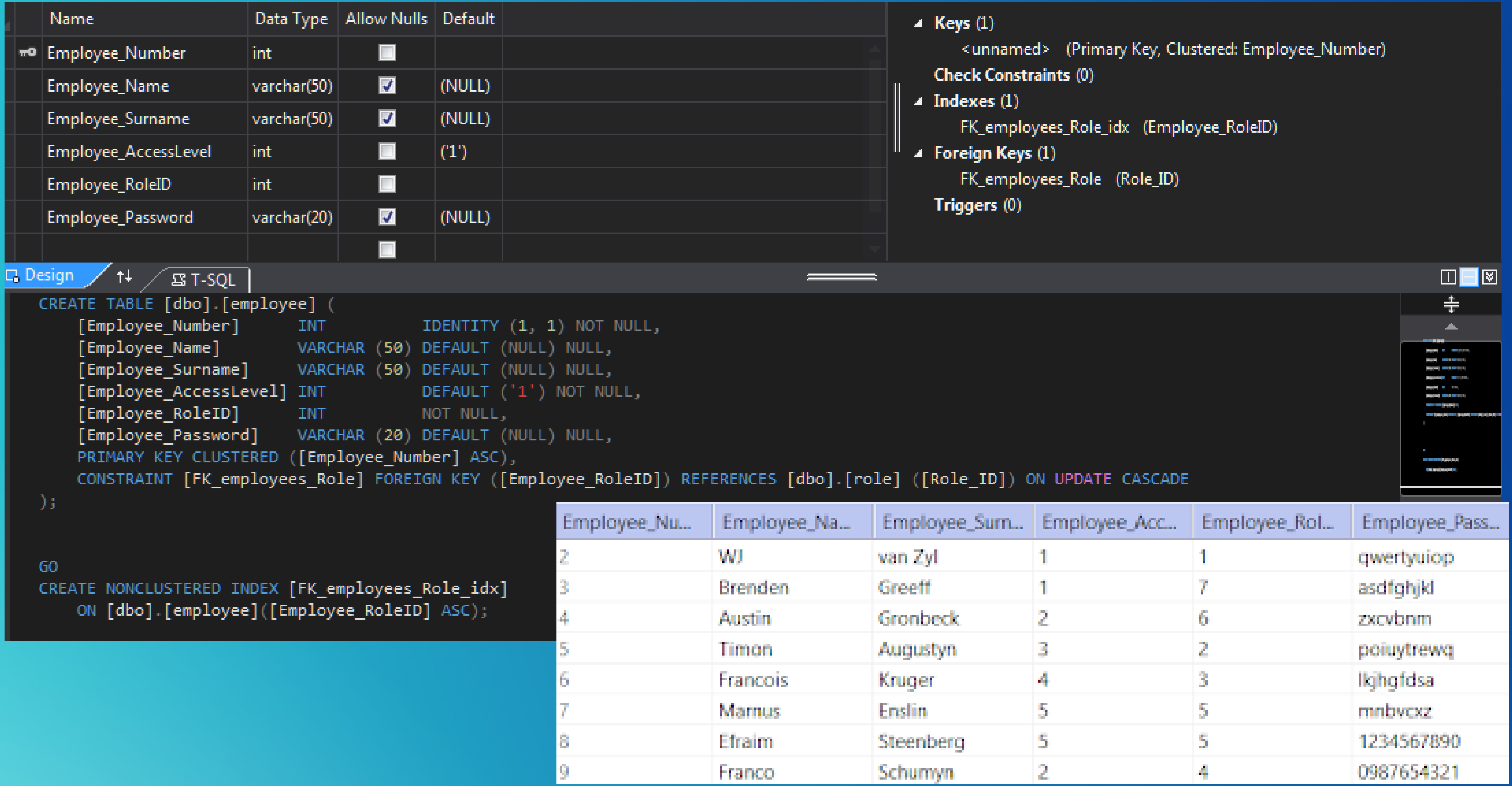
Keys (1)
<unnamed> (Primary Key, Clustered: Model_ID)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[bmodel] (
    [Model_ID]   INT          IDENTITY (1, 1) NOT NULL,
    [Model_Type] VARCHAR (50) DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Model_ID] ASC)
);
```

	Model_ID	Model_Type
	7	Scott
	8	Giant
	9	Yeti
	10	Bianchi
	11	Surly
	12	Trek
	13	Raleigh
	14	GT
	15	Cannodale
	16	Merida
	17	Fuji

DBMS SCREENSHOTS

3)EMPLOYEE TABLE



Name	Data Type	Allow Nulls	Default
Employee_Number	int	□	
Employee_Name	varchar(50)	☑	(NULL)
Employee_Surname	varchar(50)	☑	(NULL)
Employee_AccessLevel	int	□	('1')
Employee_RoleID	int	□	
Employee_Password	varchar(20)	☑	(NULL)
		□	

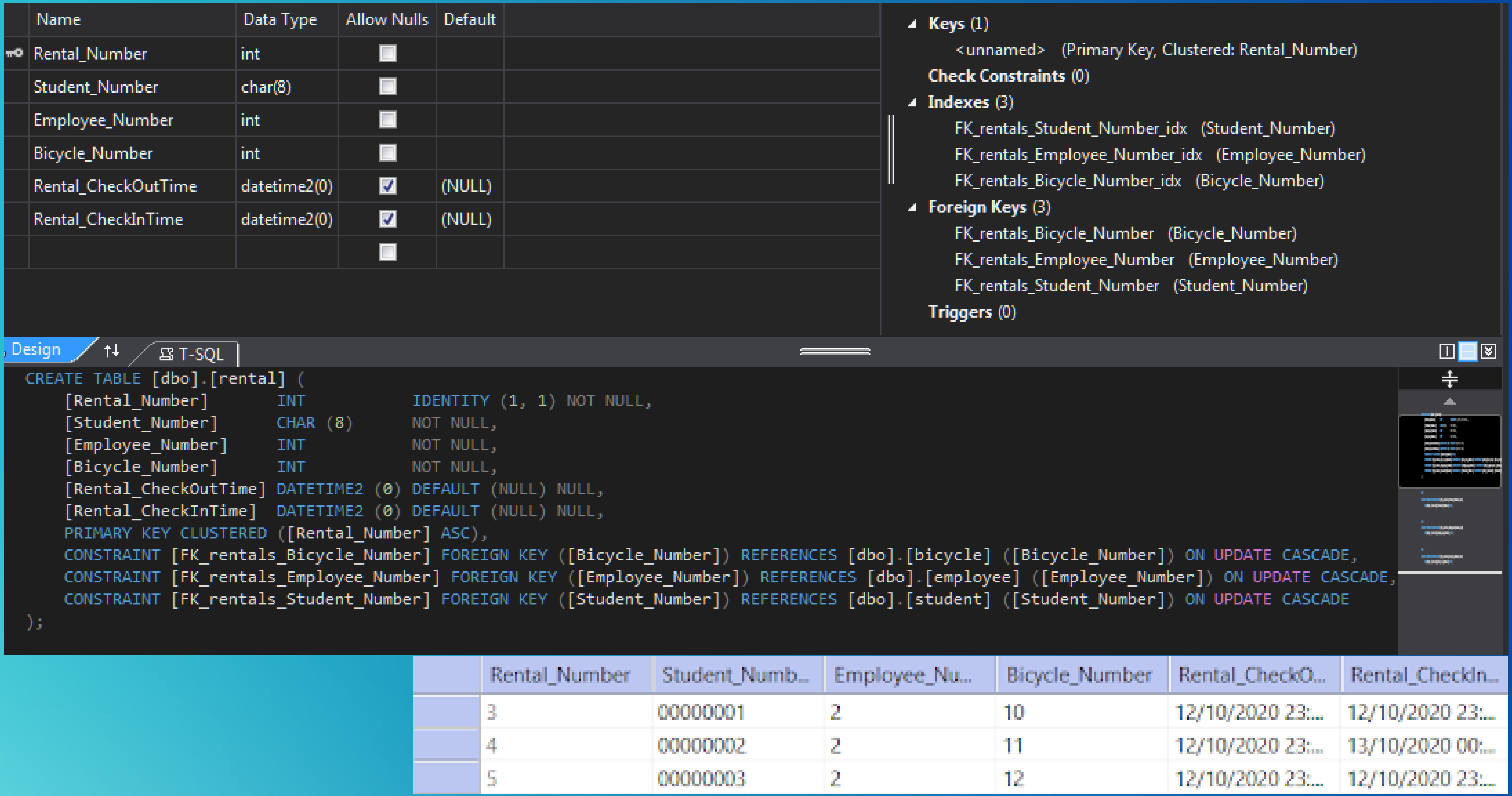
Keys (1)
<unnamed> (Primary Key, Clustered: Employee_Number)
Check Constraints (0)
Indexes (1)
FK_employees_Role_idx (Employee_RoleID)
Foreign Keys (1)
FK_employees_Role (Role_ID)
Triggers (0)

```
CREATE TABLE [dbo].[employee] (
    [Employee_Number]      INT          IDENTITY (1, 1) NOT NULL,
    [Employee_Name]        VARCHAR (50) DEFAULT (NULL) NULL,
    [Employee_Surname]     VARCHAR (50) DEFAULT (NULL) NULL,
    [Employee_AccessLevel] INT          DEFAULT ('1') NOT NULL,
    [Employee_RoleID]      INT          NOT NULL,
    [Employee_Password]    VARCHAR (20) DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Employee_Number] ASC),
    CONSTRAINT [FK_employees_Role] FOREIGN KEY ([Employee_RoleID]) REFERENCES [dbo].[role] ([Role_ID]) ON UPDATE CASCADE
);


GO
CREATE NONCLUSTERED INDEX [FK_employees_Role_idx]
    ON [dbo].[employee]([Employee_RoleID] ASC);
```

Employee_Nu...	Employee_Na...	Employee_Surn...	Employee_Acc...	Employee_Rol...	Employee_Pass...
2	WJ	van Zyl	1	1	qwertyuiop
3	Brenden	Greeff	1	7	asdfghjkl
4	Austin	Gronbeck	2	6	zxcvbnm
5	Timon	Augustyn	3	2	poiuytrewq
6	Francois	Kruger	4	3	lkjhgfdsa
7	Marnus	Enslin	5	5	mnbvcxz
8	Efraim	Steenberg	5	5	1234567890
9	Franco	Schumyn	2	4	0987654321

DBMS SCREENSHOTS

4)RENTAL TABLE



Name	Data Type	Allow Nulls	Default
Rental_Number	int	□	
Student_Number	char(8)	□	
Employee_Number	int	□	
Bicycle_Number	int	□	
Rental_CheckOutTime	datetime2(0)	☑	(NULL)
Rental_CheckInTime	datetime2(0)	☑	(NULL)
		□	

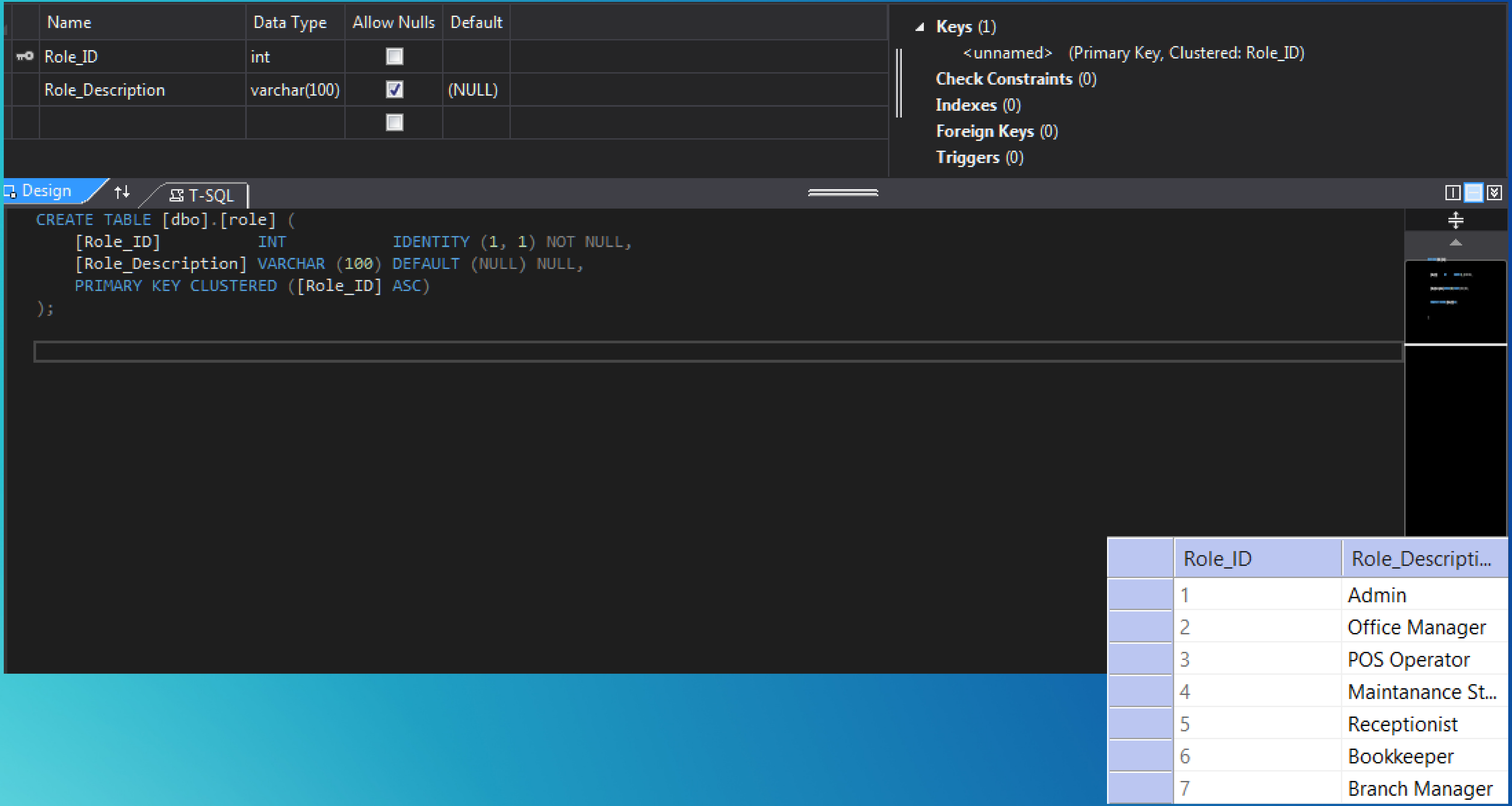
- **Keys** (1)
  - <unnamed> (Primary Key, Clustered: Rental_Number)
- **Check Constraints** (0)
- **Indexes** (3)
  - FK_rentals_Student_Number_idx (Student_Number)
  - FK_rentals_Employee_Number_idx (Employee_Number)
  - FK_rentals_Bicycle_Number_idx (Bicycle_Number)
- **Foreign Keys** (3)
  - FK_rentals_Bicycle_Number (Bicycle_Number)
  - FK_rentals_Employee_Number (Employee_Number)
  - FK_rentals_Student_Number (Student_Number)
- **Triggers** (0)

```
CREATE TABLE [dbo].[rental] (
    [Rental_Number]       INT           IDENTITY (1, 1) NOT NULL,
    [Student_Number]      CHAR (8)      NOT NULL,
    [Employee_Number]     INT           NOT NULL,
    [Bicycle_Number]      INT           NOT NULL,
    [Rental_CheckOutTime] DATETIME2 (0) DEFAULT (NULL) NULL,
    [Rental_CheckInTime]  DATETIME2 (0) DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Rental_Number] ASC),
    CONSTRAINT [FK_rentals_Bicycle_Number] FOREIGN KEY ([Bicycle_Number]) REFERENCES [dbo].[bicycle] ([Bicycle_Number]) ON UPDATE CASCADE,
    CONSTRAINT [FK_rentals_Employee_Number] FOREIGN KEY ([Employee_Number]) REFERENCES [dbo].[employee] ([Employee_Number]) ON UPDATE CASCADE,
    CONSTRAINT [FK_rentals_Student_Number] FOREIGN KEY ([Student_Number]) REFERENCES [dbo].[student] ([Student_Number]) ON UPDATE CASCADE
);
```

Rental_Number	Student_Numb...	Employee_Nu...	Bicycle_Number	Rental_CheckO...	Rental_CheckIn...
3	00000001	2	10	12/10/2020 23:...	12/10/2020 23:...
4	00000002	2	11	12/10/2020 23:...	13/10/2020 00:...
5	00000003	2	12	12/10/2020 23:...	12/10/2020 23:...

DBMS SCREENSHOTS

5)ROLE TABLE



	Name	Data Type	Allow Nulls	Default
🔑	Role_ID	int	□	
	Role_Description	varchar(100)	☑	(NULL)
			□	

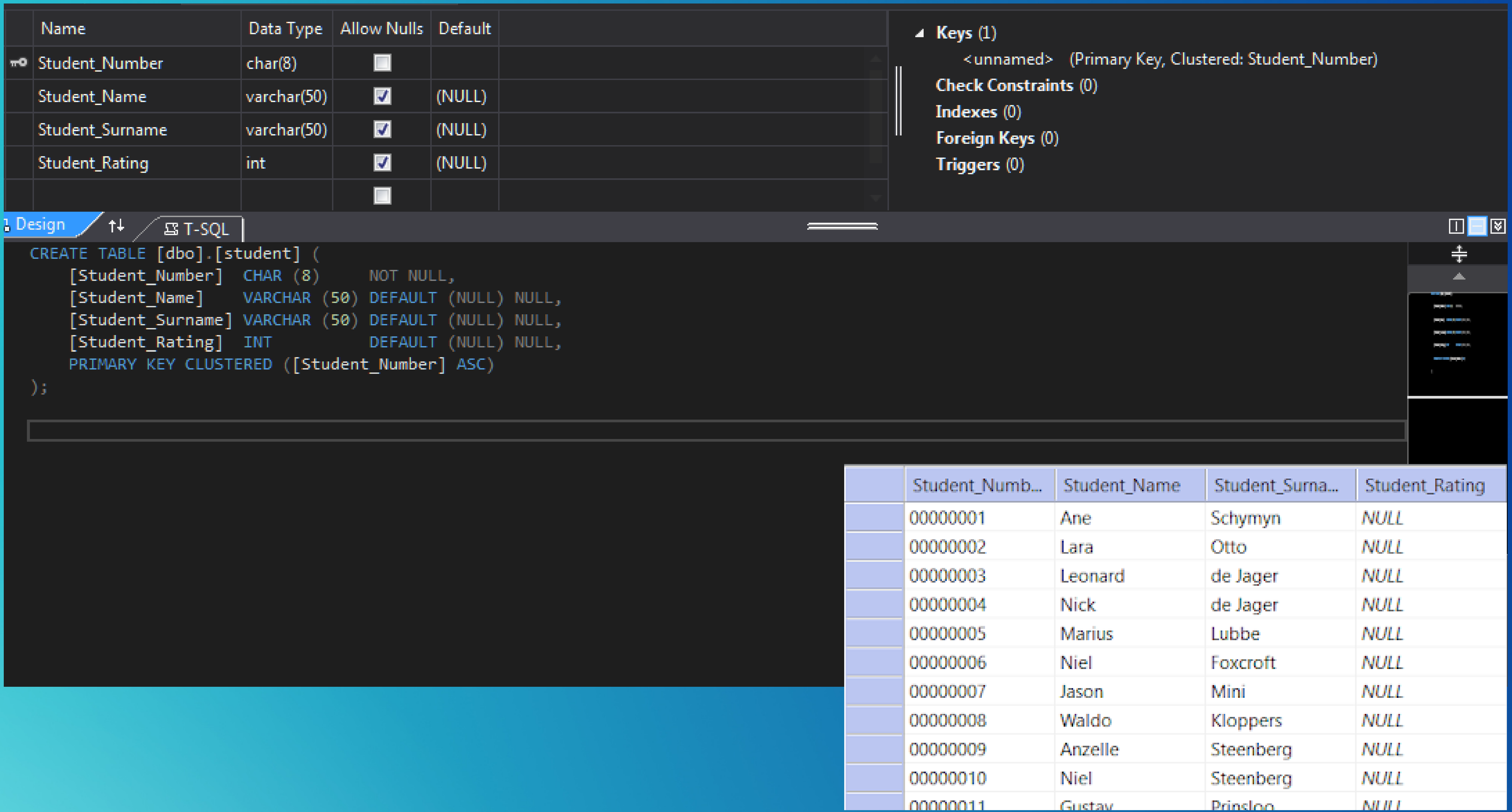
Keys (1)
<unnamed> (Primary Key, Clustered: Role_ID)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[role] (
    [Role_ID]          INT           IDENTITY (1, 1) NOT NULL,
    [Role_Description] VARCHAR (100) DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Role_ID] ASC)
);
```

Role_ID	Role_Descripti...
1	Admin
2	Office Manager
3	POS Operator
4	Maintanance St...
5	Receptionist
6	Bookkeeper
7	Branch Manager

DBMS SCREENSHOTS

6)STUDENT TABLE



	Name	Data Type	Allow Nulls	Default
🔑	Student_Number	char(8)	□	
	Student_Name	varchar(50)	☑	(NULL)
	Student_Surname	varchar(50)	☑	(NULL)
	Student_Rating	int	☑	(NULL)
			□	

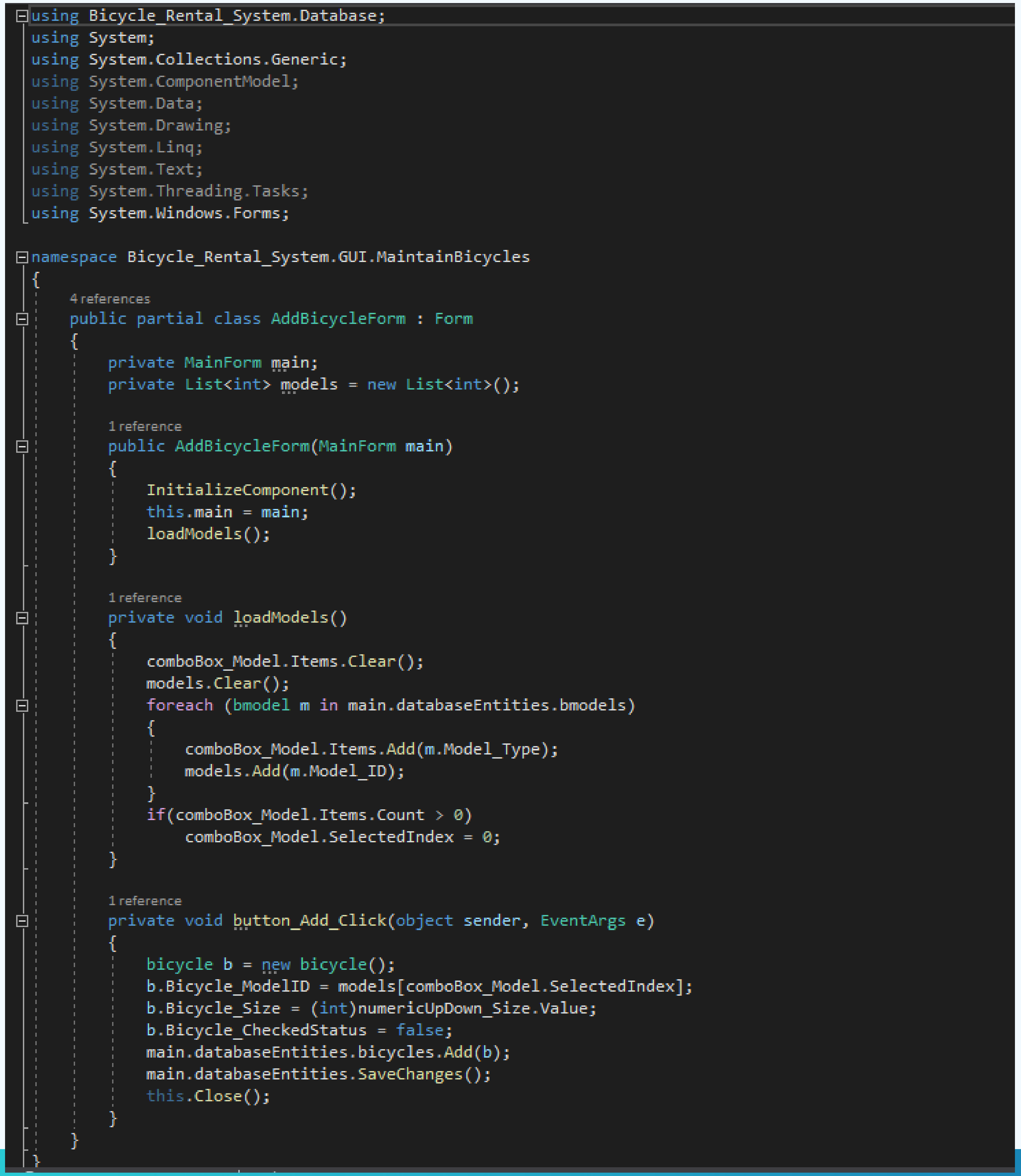
Keys (1)
<unnamed> (Primary Key, Clustered: Student_Number)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[student] (
    [Student_Number]  CHAR (8)     NOT NULL,
    [Student_Name]    VARCHAR (50) DEFAULT (NULL) NULL,
    [Student_Surname] VARCHAR (50) DEFAULT (NULL) NULL,
    [Student_Rating]  INT          DEFAULT (NULL) NULL,
    PRIMARY KEY CLUSTERED ([Student_Number] ASC)
);
```

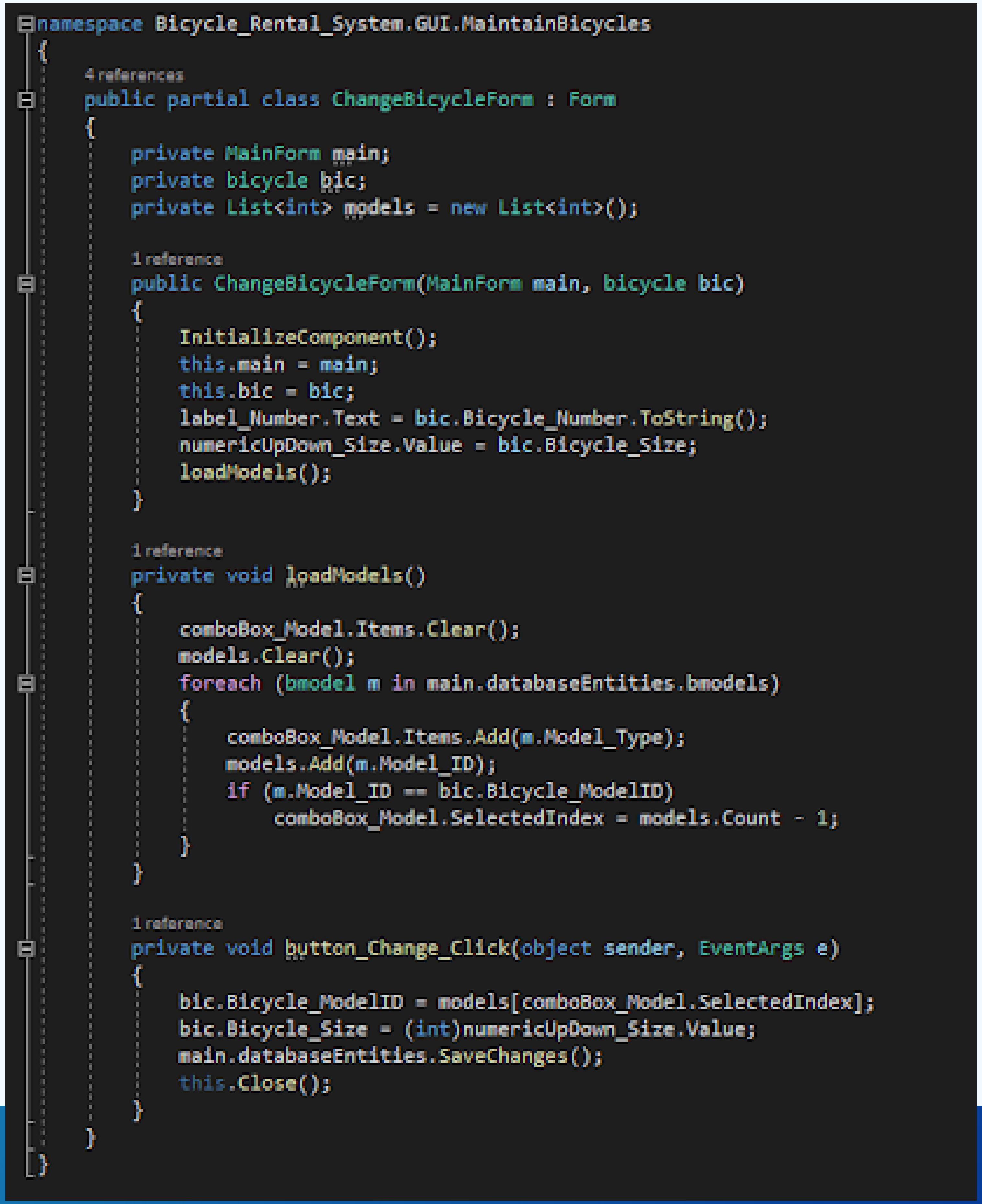
	Student_Numb...	Student_Name	Student_Surna...	Student_Rating
	00000001	Ane	Schymyn	*NULL*
	00000002	Lara	Otto	*NULL*
	00000003	Leonard	de Jager	*NULL*
	00000004	Nick	de Jager	*NULL*
	00000005	Marius	Lubbe	*NULL*
	00000006	Niel	Foxcroft	*NULL*
	00000007	Jason	Mini	*NULL*
	00000008	Waldo	Kloppers	*NULL*
	00000009	Anzelle	Steenberg	*NULL*
	00000010	Niel	Steenberg	*NULL*
	00000011	Gustav	Prinsloo	*NULL*

Example Code

Add and Edit Bicycle



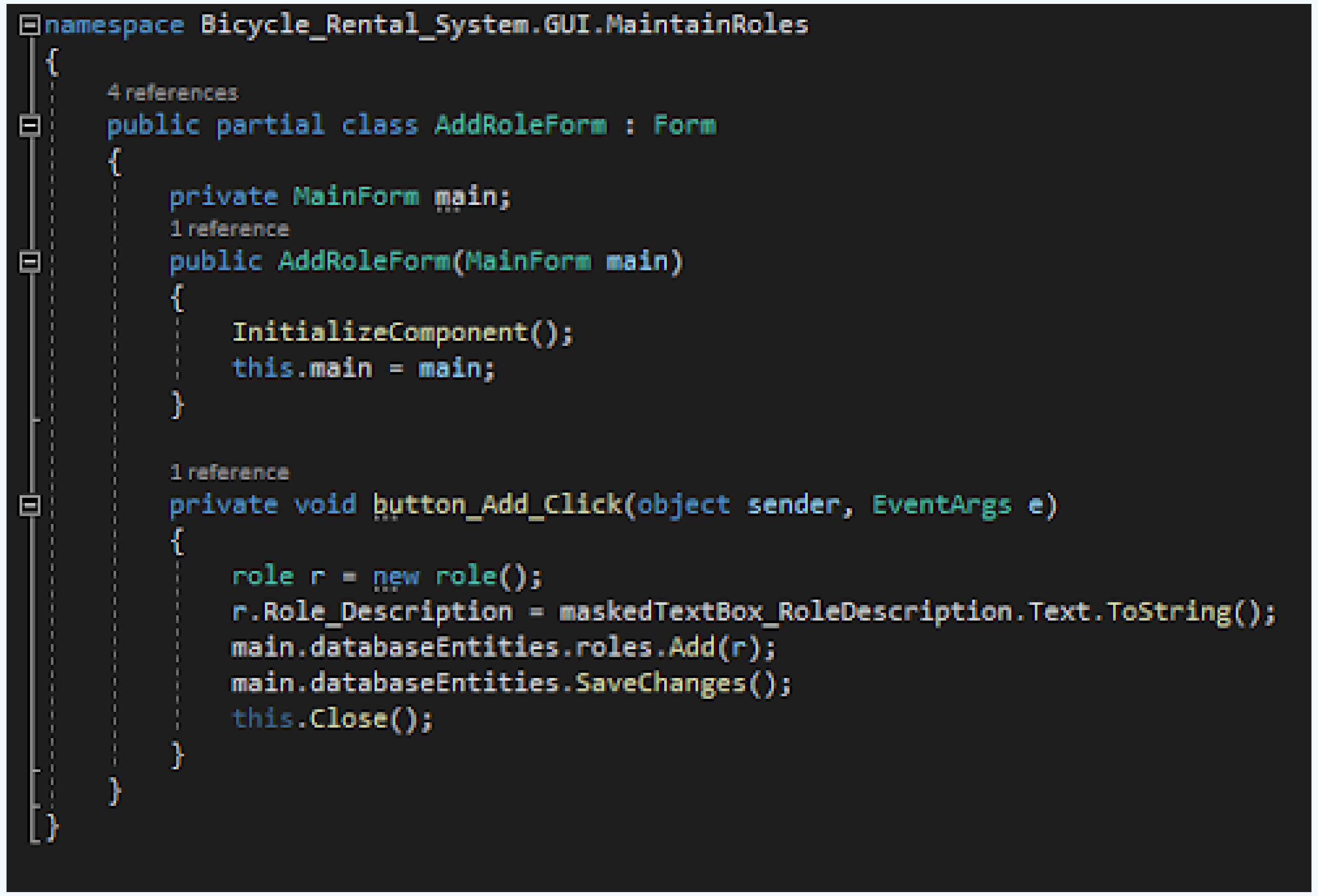
```
using Bicycle_Rental_System.Database;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Bicycle_Rental_System.GUI.MaintainBicycles
{
    4 references
    public partial class AddBicycleForm : Form
    {
        private MainForm main;
        private List<int> models = new List<int>();

        1 reference
        public AddBicycleForm(MainForm main)
        {
            InitializeComponent();
            this.main = main;
            loadModels();
        }

        1 reference
        private void loadModels()
        {
            comboBox_Model.Items.Clear();
            models.Clear();
            foreach (bmodel m in main.databaseEntities.bmodels)
            {
                comboBox_Model.Items.Add(m.Model_Type);
                models.Add(m.Model_ID);
            }
            if(comboBox_Model.Items.Count > 0)
                comboBox_Model.SelectedIndex = 0;
        }

        1 reference
        private void button_Add_Click(object sender, EventArgs e)
        {
            bicycle b = new bicycle();
            b.Bicycle_ModelID = models[comboBox_Model.SelectedIndex];
            b.Bicycle_Size = (int)numericUpDown_Size.Value;
            b.Bicycle_CheckedStatus = false;
            main.databaseEntities.bicycles.Add(b);
            main.databaseEntities.SaveChanges();
            this.Close();
        }
    }
}
```



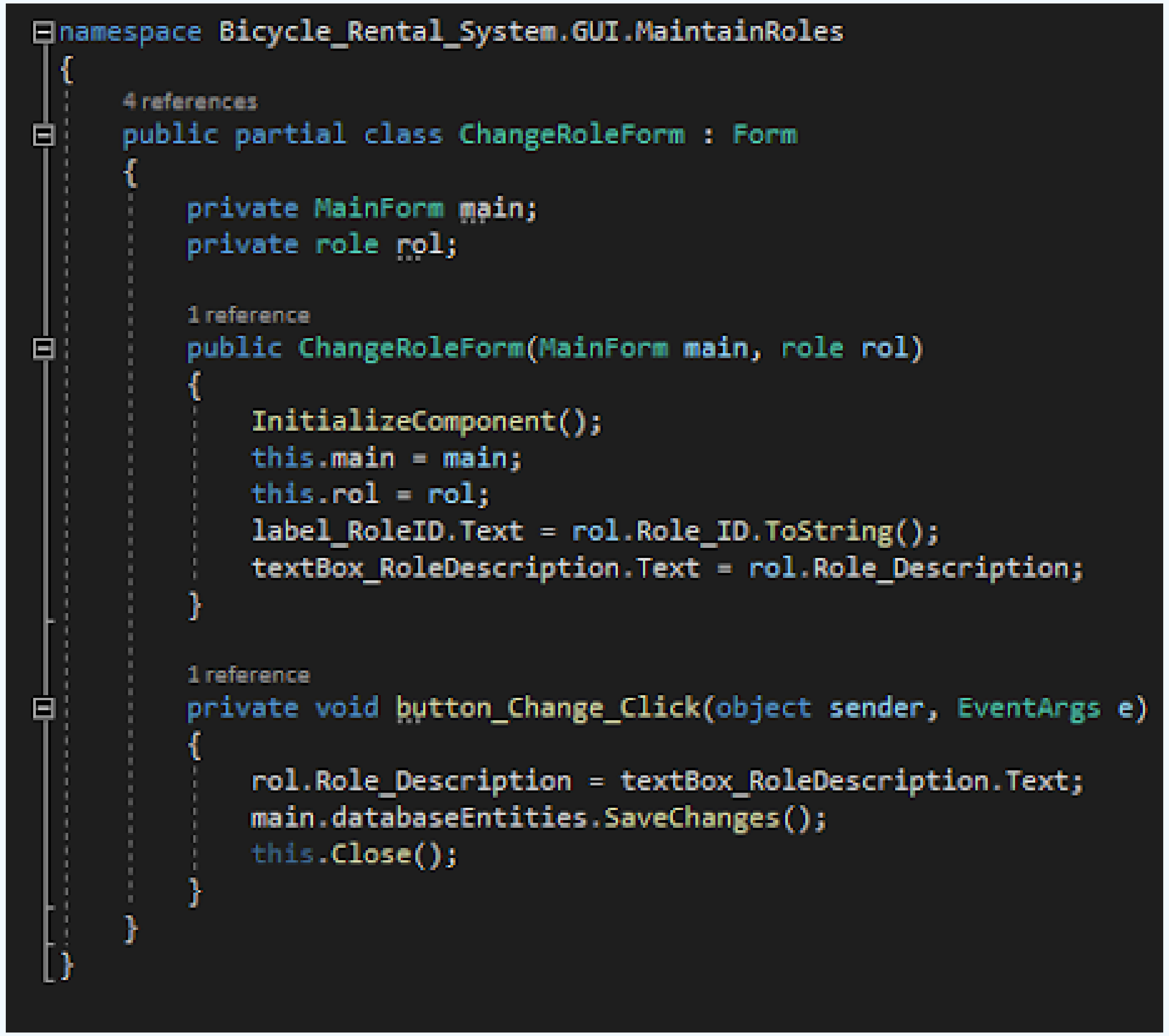
```
namespace Bicycle_Rental_System.GUI.MaintainBicycles
{
    4 references
    public partial class ChangeBicycleForm : Form
    {
        private MainForm main;
        private bicycle bic;
        private List<int> models = new List<int>();

        1 reference
        public ChangeBicycleForm(MainForm main, bicycle bic)
        {
            InitializeComponent();
            this.main = main;
            this.bic = bic;
            label_Number.Text = bic.Bicycle_Number.ToString();
            numericUpDown_Size.Value = bic.Bicycle_Size;
            loadModels();
        }

        1 reference
        private void loadModels()
        {
            comboBox_Model.Items.Clear();
            models.Clear();
            foreach (bmodel m in main.databaseEntities.bmodels)
            {
                comboBox_Model.Items.Add(m.Model_Type);
                models.Add(m.Model_ID);
                if (m.Model_ID == bic.Bicycle_ModelID)
                    comboBox_Model.SelectedIndex = models.Count - 1;
            }
        }

        1 reference
        private void button_Change_Click(object sender, EventArgs e)
        {
            bic.Bicycle_ModelID = models[comboBox_Model.SelectedIndex];
            bic.Bicycle_Size = (int)numericUpDown_Size.Value;
            main.databaseEntities.SaveChanges();
            this.Close();
        }
    }
}
```

Example Code

Add and Edit Roles



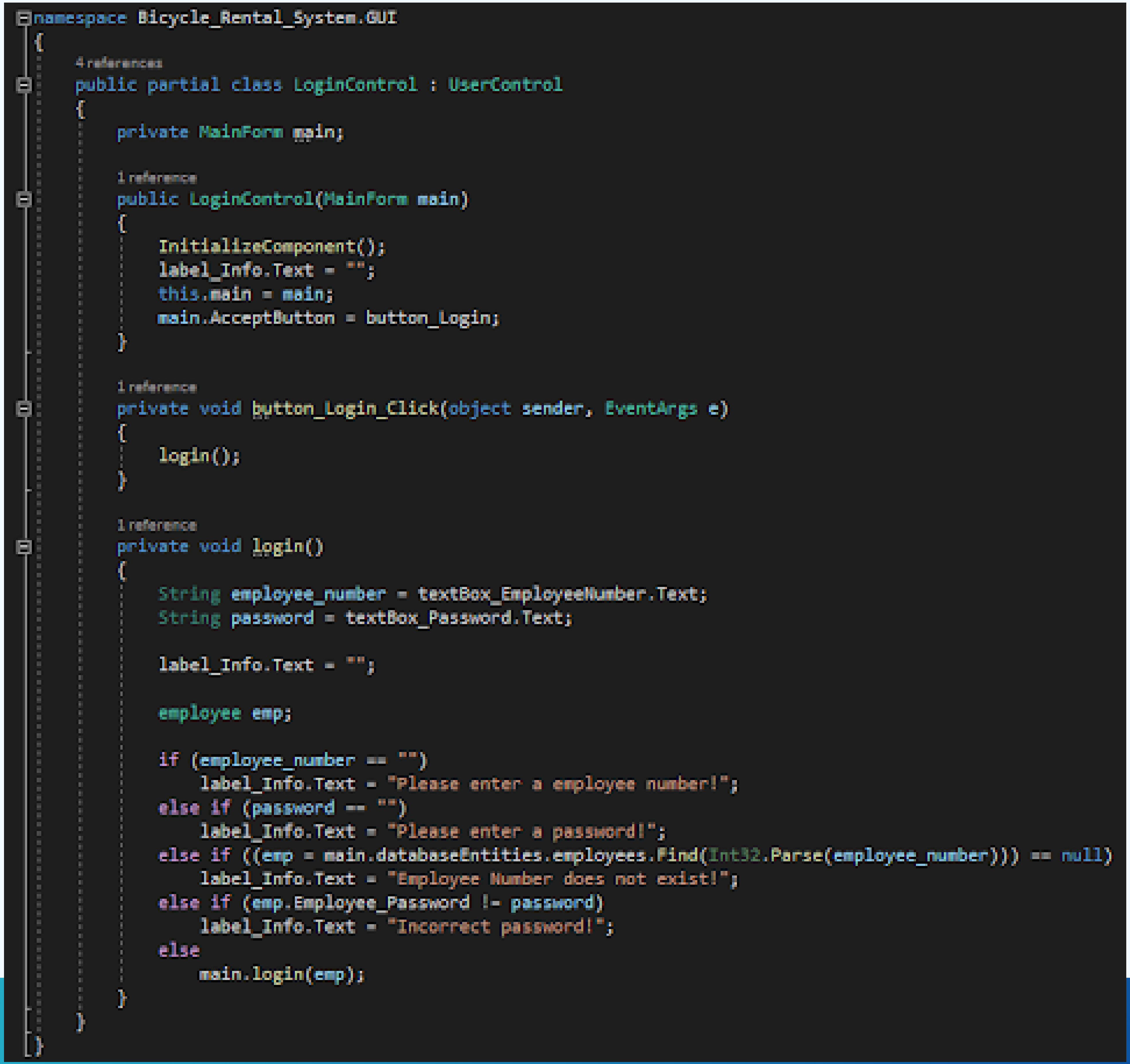
```
namespace Bicycle_Rental_System.GUI.MaintainRoles
{
    4 references
    public partial class AddRoleForm : Form
    {
        private MainForm main;
        1 reference
        public AddRoleForm(MainForm main)
        {
            InitializeComponent();
            this.main = main;
        }

        1 reference
        private void button_Add_Click(object sender, EventArgs e)
        {
            role r = new role();
            r.Role_Description = maskedTextBox_RoleDescription.Text.ToString();
            main.databaseEntities.roles.Add(r);
            main.databaseEntities.SaveChanges();
            this.Close();
        }
    }
}
```



```
namespace Bicycle_Rental_System.GUI.MaintainRoles
{
    4 references
    public partial class ChangeRoleForm : Form
    {
        private MainForm main;
        private role rol;

        1 reference
        public ChangeRoleForm(MainForm main, role rol)
        {
            InitializeComponent();
            this.main = main;
            this.rol = rol;
            label_RoleID.Text = rol.Role_ID.ToString();
            textBox_RoleDescription.Text = rol.Role_Description;
        }

        1 reference
        private void button_Change_Click(object sender, EventArgs e)
        {
            rol.Role_Description = textBox_RoleDescription.Text;
            main.databaseEntities.SaveChanges();
            this.Close();
        }
    }
}
```

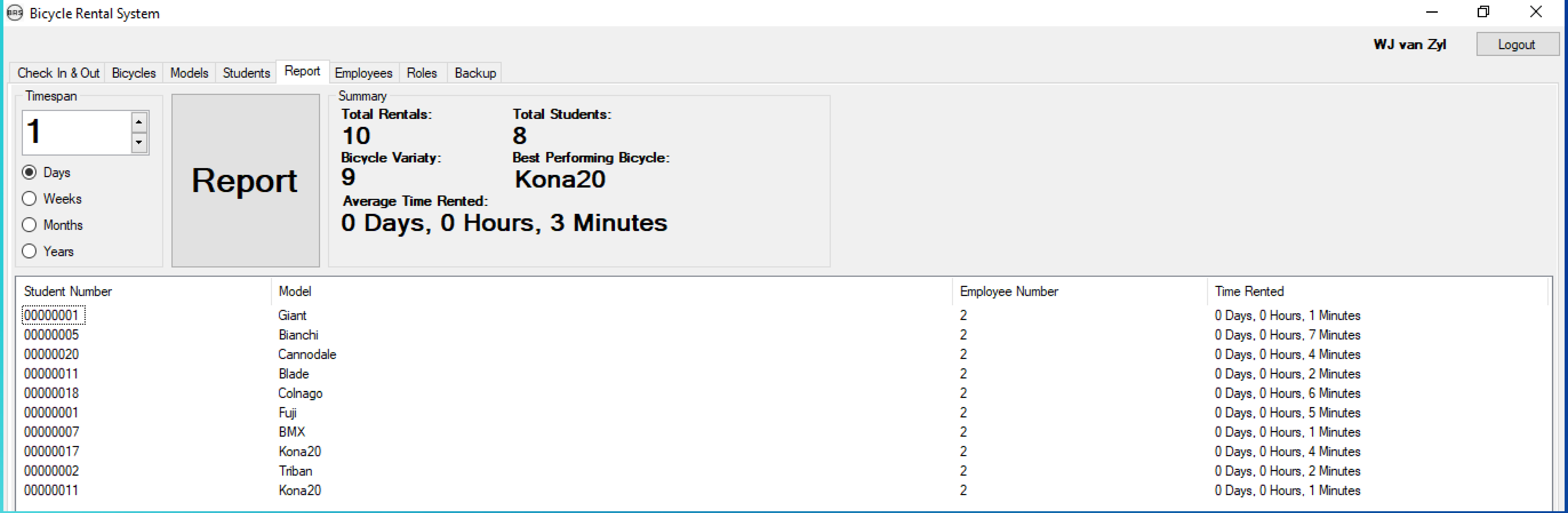
Example Code

Login



```
namespace Bicycle_Rental_System.GUI
{
    4 references
    public partial class LoginControl : UserControl
    {
        private MainForm main;

        1 reference
        public LoginControl(MainForm main)
        {
            InitializeComponent();
            label_Info.Text = "";
            this.main = main;
            main.AcceptButton = button_Login;
        }

        1 reference
        private void button_Login_Click(object sender, EventArgs e)
        {
            login();
        }

        1 reference
        private void login()
        {
            String employee_number = textBox_EmployeeNumber.Text;
            String password = textBox_Password.Text;

            label_Info.Text = "";

            employee emp;

            if (employee_number == "")
                label_Info.Text = "Please enter a employee number!";
            else if (password == "")
                label_Info.Text = "Please enter a password!";
            else if ((emp = main.databaseEntities.employees.Find(Int32.Parse(employee_number))) == null)
                label_Info.Text = "Employee Number does not exist!";
            else if (emp.Employee_Password != password)
                label_Info.Text = "Incorrect password!";
            else
                main.login(emp);
        }
    }
}
```

REPORTS



WJ van Zyl

Timespan

1

● Days
○ Weeks
○ Months
○ Years

Report

Summary

Total Rentals: 10

Total Students: 8

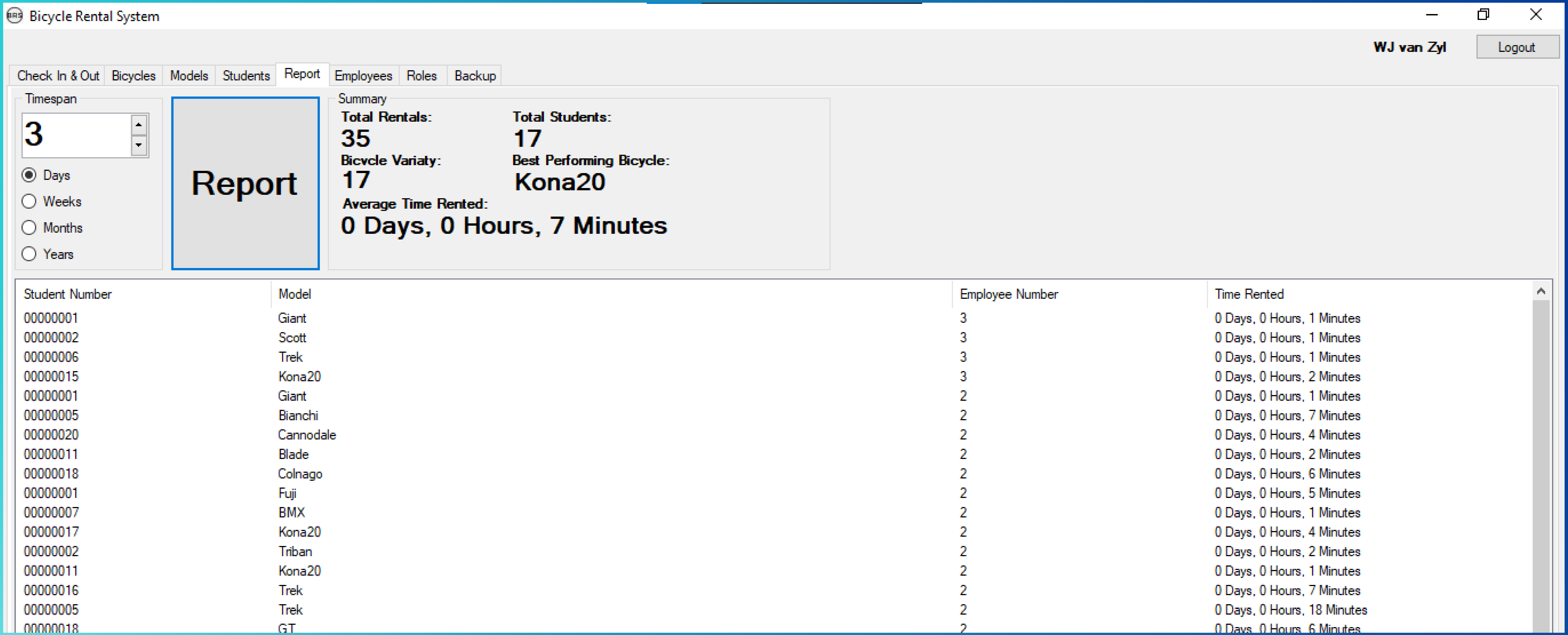
Bicycle Variaty: 9

Best Perfoming Bicycle: Kona20

Average Time Rented: 0 Days, 0 Hours, 3 Minutes

Student Number	Model	Employee Number	Time Rented
00000001	Giant	2	0 Days, 0 Hours, 1 Minutes
00000005	Bianchi	2	0 Days, 0 Hours, 7 Minutes
00000020	Cannodale	2	0 Days, 0 Hours, 4 Minutes
00000011	Blade	2	0 Days, 0 Hours, 2 Minutes
00000018	Colnago	2	0 Days, 0 Hours, 6 Minutes
00000001	Fuji	2	0 Days, 0 Hours, 5 Minutes
00000007	BMX	2	0 Days, 0 Hours, 1 Minutes
00000017	Kona20	2	0 Days, 0 Hours, 4 Minutes
00000002	Triban	2	0 Days, 0 Hours, 2 Minutes
00000011	Kona20	2	0 Days, 0 Hours, 1 Minutes

REPORTS CONT



Timespan

3

● Days
○ Weeks
○ Months
○ Years

Summary

Total Rentals:
35

Total Students:
17

Bicycle Variaty:
17

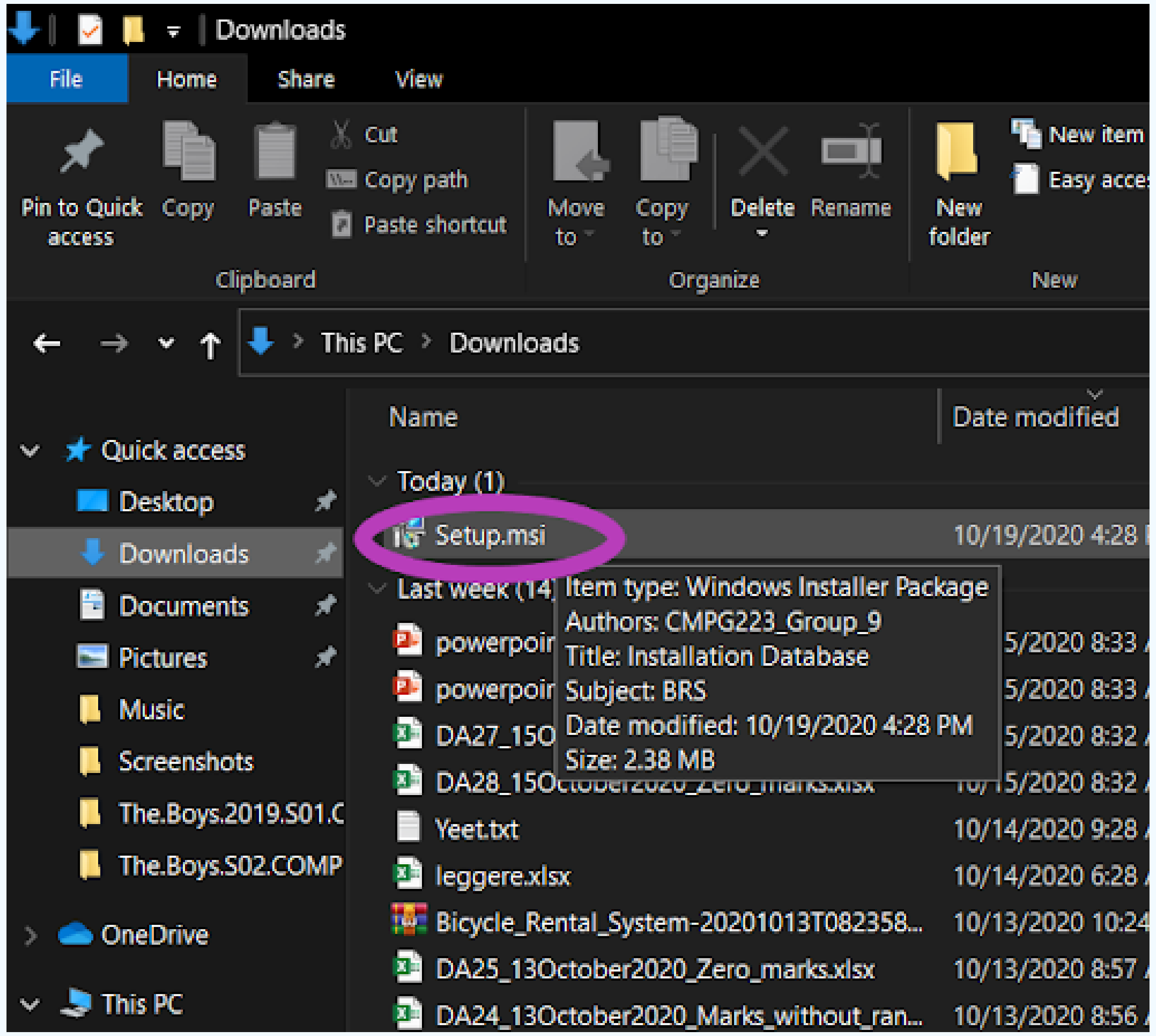
Best Performing Bicycle:
Kona20

Average Time Rented:
0 Days, 0 Hours, 7 Minutes

Student Number	Model	Employee Number	Time Rented
00000001	Giant	3	0 Days, 0 Hours, 1 Minutes
00000002	Scott	3	0 Days, 0 Hours, 1 Minutes
00000006	Trek	3	0 Days, 0 Hours, 1 Minutes
00000015	Kona20	3	0 Days, 0 Hours, 2 Minutes
00000001	Giant	2	0 Days, 0 Hours, 1 Minutes
00000005	Bianchi	2	0 Days, 0 Hours, 7 Minutes
00000020	Cannodale	2	0 Days, 0 Hours, 4 Minutes
00000011	Blade	2	0 Days, 0 Hours, 2 Minutes
00000018	Colnago	2	0 Days, 0 Hours, 6 Minutes
00000001	Fuji	2	0 Days, 0 Hours, 5 Minutes
00000007	BMX	2	0 Days, 0 Hours, 1 Minutes
00000017	Kona20	2	0 Days, 0 Hours, 4 Minutes
00000002	Triban	2	0 Days, 0 Hours, 2 Minutes
00000011	Kona20	2	0 Days, 0 Hours, 1 Minutes
00000016	Trek	2	0 Days, 0 Hours, 7 Minutes
00000005	Trek	2	0 Days, 0 Hours, 18 Minutes
00000018	GT	2	0 Days, 0 Hours, 6 Minutes

User Manual

1)

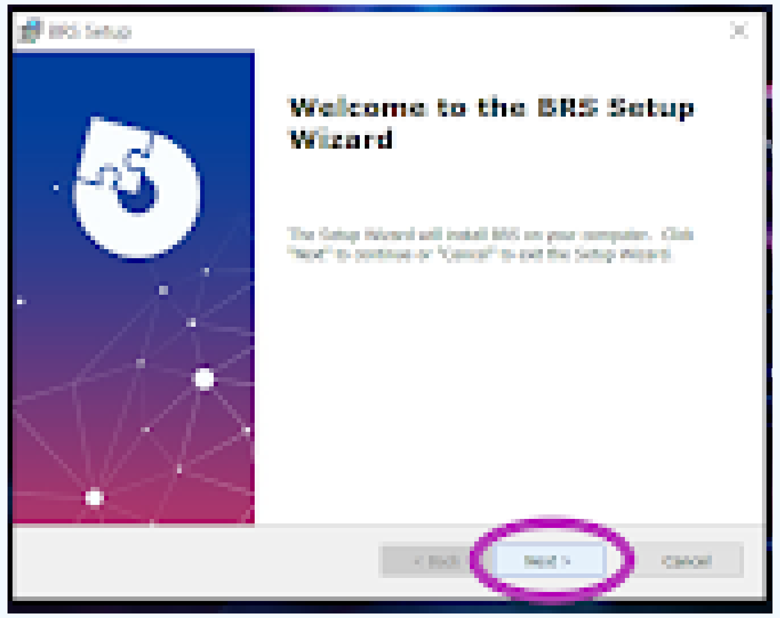


Setup.msi 10/19/2020 4:28

Item type: Windows Installer Package
Authors: CMPG223_Group_9
Title: Installation Database
Subject: BRS
Date modified: 10/19/2020 4:28 PM
Size: 2.38 MB

powerpoir 5/2020 8:33
powerpoir 5/2020 8:33
DA27_15O 5/2020 8:32
DA28_15October2020_Zero_marks.xlsx 10/15/2020 8:32
Yeet.txt 10/14/2020 9:28
leggere.xlsx 10/14/2020 6:28
Bicycle_Rental_System-20201013T082358... 10/13/2020 10:24
DA25_13October2020_Zero_marks.xlsx 10/13/2020 8:57
DA24_13October2020_Marks_without_ran... 10/13/2020 8:56

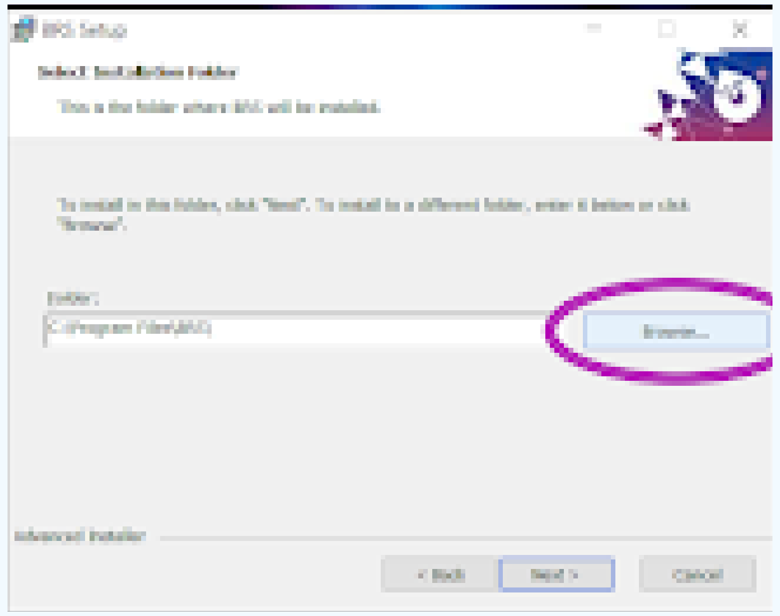
2)



Welcome to the BRS Setup Wizard

The Setup Wizard will install BRS on your computer. Click "Next" to continue or "Cancel" to exit the Setup Wizard.

3)



Select Installation Folder

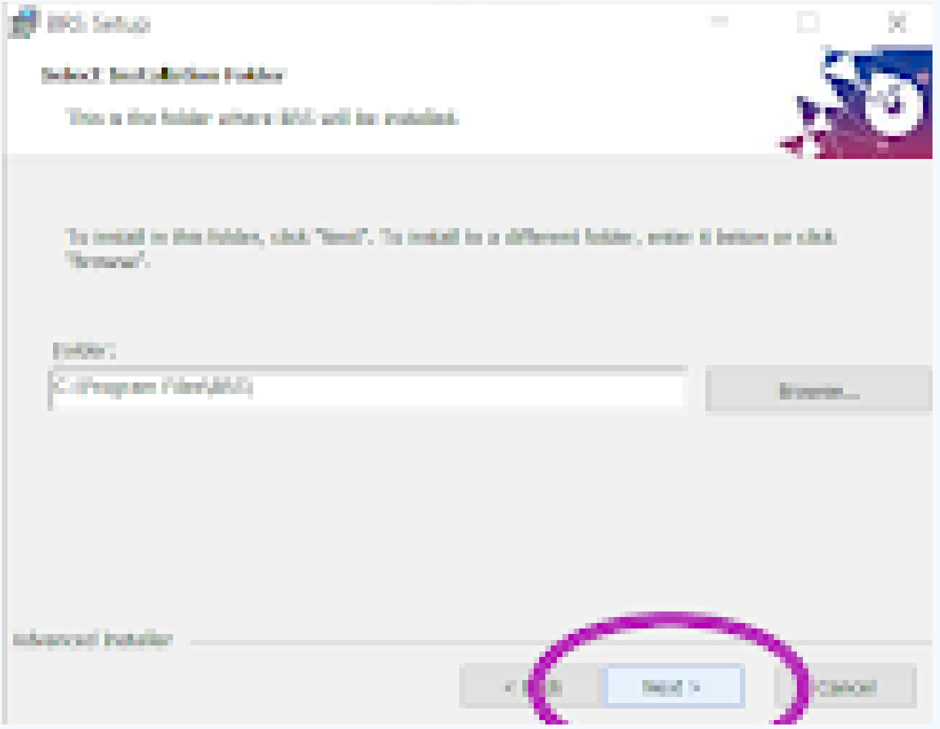
This is the folder where BRS will be installed.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder:
C:\Program Files\BRS\

User Manual

4)



BRS Setup

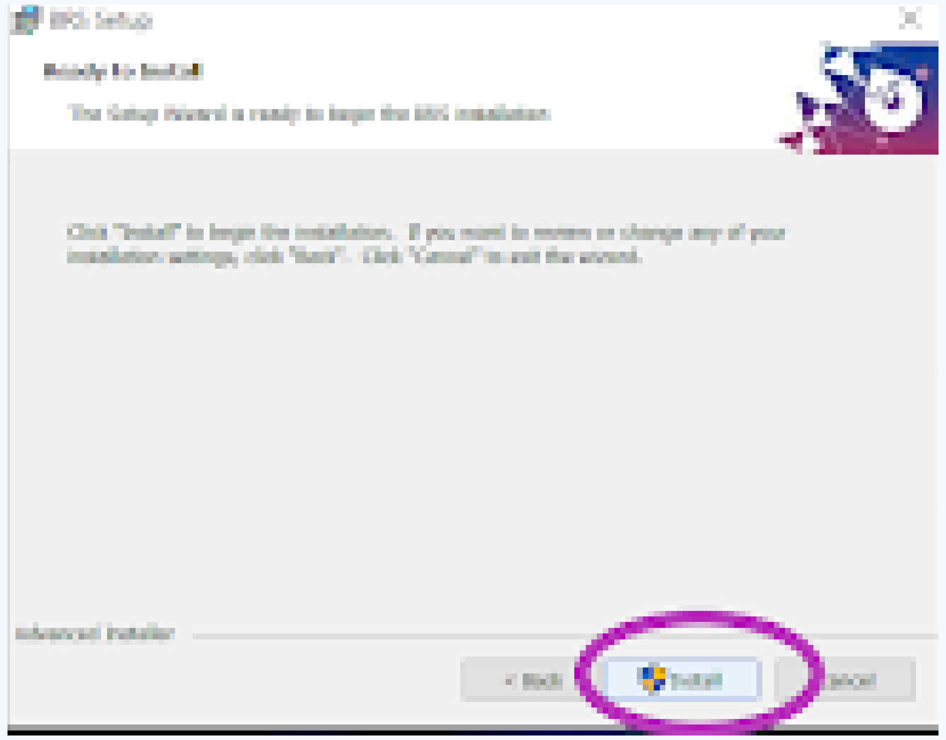
Select Installation Folder

This is the folder where BRS will be installed.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder:
C:\Program Files\BRS\

5)



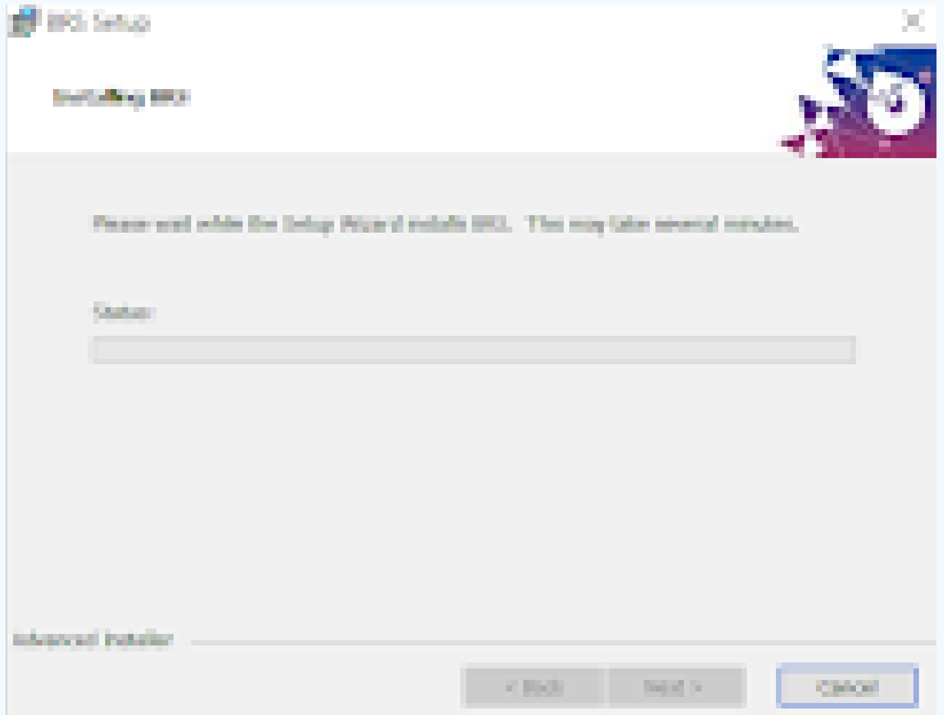
BRS Setup

Ready to Install

The Setup Wizard is ready to begin the BRS installation.

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click "Back". Click "Cancel" to exit the wizard.

6)



BRS Setup

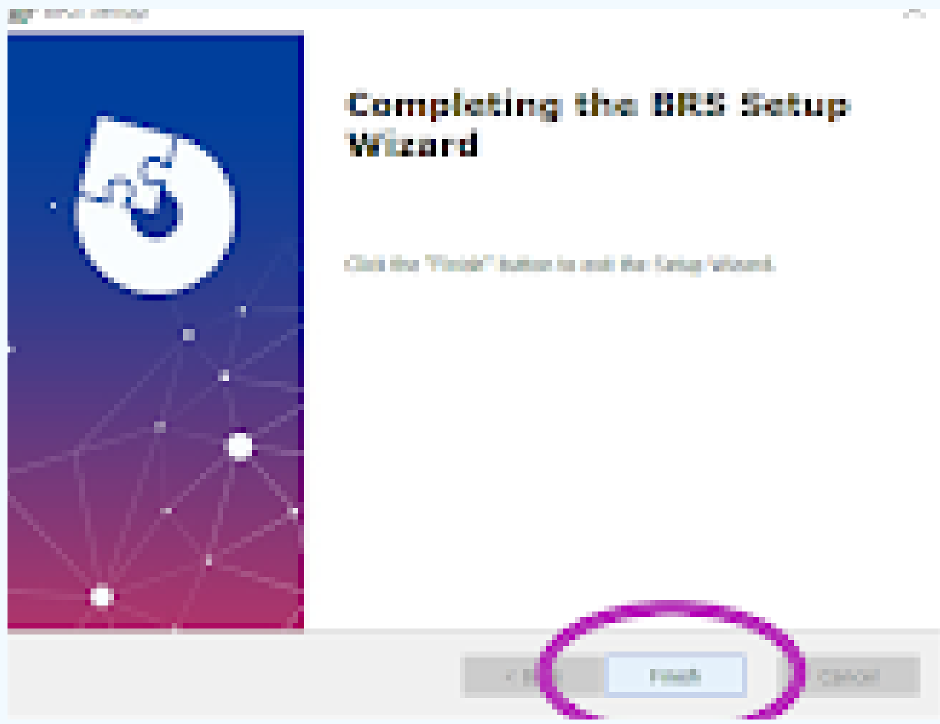
Installing BRS

Please wait while the Setup Wizard installs BRS. This may take several minutes.

Status:

User Manual

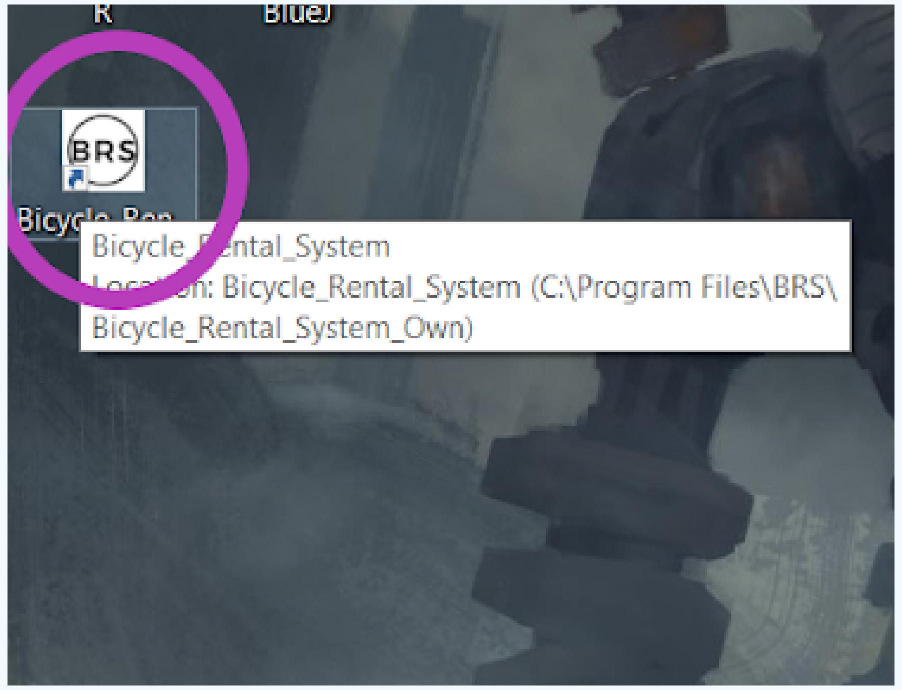
7)



Completing the BRS Setup Wizard

Click the "Finish" button to exit the Setup Wizard.

8)



Bicycle_Rental_System
Location: Bicycle_Rental_System (C:\Program Files\BRS\
Bicycle_Rental_System_Own)

SYSTEM DEMONSTRATION

GROUP 09 MEMBERS

Austin Gronbeck - 30627605
Timon Augustyn - 28585984
Efraim Steenberg - 29652537
Marnus Enslin - 32377630
WJ van Zyl - 30119170
Francois Kruger - 30198356
Brenden Greeff - 29972396