Employee Attendance Tracker

A Final Requirement to Database Administration
Presented to Professor Monina D. Barretto

Faculty of College of Computer and Information Sciences

Polytechnic University of the Philippines

Sta. Mesa, Manila

Mondia, Robbie A.

BSCS 3-1N

2020

Background

Written logs are about to be replaced by digital logs such as attendance, sales report, minutes of the meeting and the like. Although written logs are important and should not be thrown out of the picture, digitally tracked reports can be a helping hand specially on busy and hectic days.

This project was designed in accordance with the setup of an ESL company; English Development Pro-Asia Inc. where I used to work. There, we use a logbook where we write: Nickname, email address, date, time in, and later, time out that will serve as our attendance which I think kind of a hassle because there are a lot of details to fill in. Few months after, the management decided to use Google Forms as our attendance, although it is digital, the number of things to fill in is still the same so it did not really help remove the hassle.

With the Employee Attendance Tracker, attendance for the day will be recorded within just few clicks. This improved attendance log will bring less hassle to the employees that will log every day and it features a data management for the staff to easily navigate the log for the day or whenever they want to check something on the history.

Tables and Data Dictionary

employee_tbl – contains the data of employees working in English Pro-Asia.

Name	Data Type	Length	Description
EmpID	Alphanumeric	10	Unique ID of the employee
Lname	Alphanumeric	30	Last name of the employee
Fname	Alphanumeric	30	First name of the employee
MI	Alphanumeric	5	Middle initial of the employee
Nickname	Alphanumeric	15	Nickname of the employee and will be used to their respective nametag
Sex	Character	1	Sex of the employee
Birthdate	DATE		Date of birth of the employee
Age	Integer	2	Age of the employee
Address	Alphanumeric	50	Current and permanent address of the employee
Email	Alphanumeric	40	Email of the employee

Contact Number	**Alphanumeric**	**11**	**Contact number of the employee**
Employee Type	Alphanumeric	15	Category of the employee that falls under four types: - Staff - Regular - Part-timer - Provisional
Team Name	Alphanumeric	10	Team name division: - Sanji - Nico Robin - Nami - Luffy - Zoro
Teacher Class	Alphanumeric	30	Current class of the employee that varies on their quarterly performance: - Class A 160/hr. - Class B 140/hr. - Class C 120/hr.
Password	Alphanumeric	30	Password of admins and employees

attendanceLog_tbl – history of employee's daily attendance log.

Name	Data Type	Length	Description
EmpID	Alphanumeric	10	Unique ID of the employee
Date	DATE		Current date of the employee log. (Based on system time)
Time In	TIME		Current time when the employee logs in. (Based on system time)
Time Out	TIME		Current time when the employee logs out. (Based on system time)

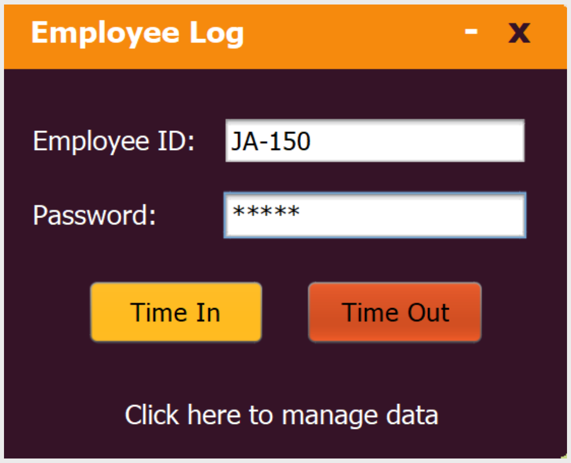
management_history – history of employee's data management.

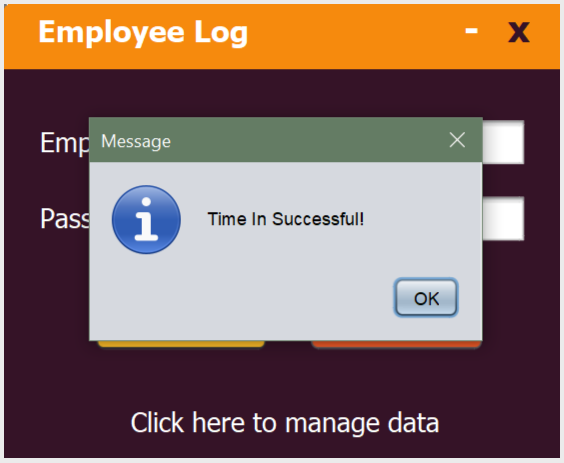
Name	Data Type	Length	Description
RecordCount	INT AUTO_INCREMENT	10	Record count
EmpID	Alphanumeric	10	Unique ID of the employee
Lname	Alphanumeric	30	Last name of the employee
Fname	Alphanumeric	30	First name of the employee
MI	Alphanumeric	5	Middle initial of the employee
Nickname	Alphanumeric	15	Nickname of the employee and will be used to their respective nametag
Sex	Character	1	Sex of the employee

Employee Type	Alphanumeric	15	Category of the employee that falls under four types: - Staff - Regular - Part-timer - Provisional
Team Name	Alphanumeric	10	Team name division: - Sanji - Nico Robin - Nami - Luffy - Zoro
Teacher Class	Alphanumeric	30	Current class of the employee that varies on their quarterly performance: - Class A 160/hr. - Class B 140/hr. - Class C 120/hr.
Password	Alphanumeric	30	Password of admins and employees
Action	Alphanumeric	15	Action taken from data management: - Added - Deleted - Old Record - Updated Record

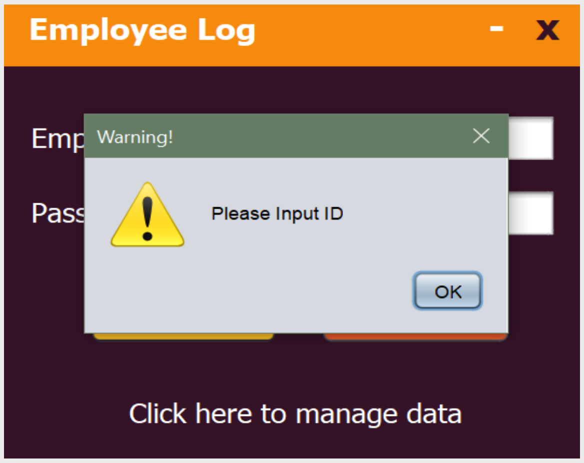
User Interface

Employee Log – An error catcher will check if the fields are empty or if the input of the user was validated against the database. It will then fire a warning to fill the fields first (if empty) or will prompt a warning if the credentials are incorrect or will warn again if multiple Time Ins are attempted because the user is only allowed to log once per day.

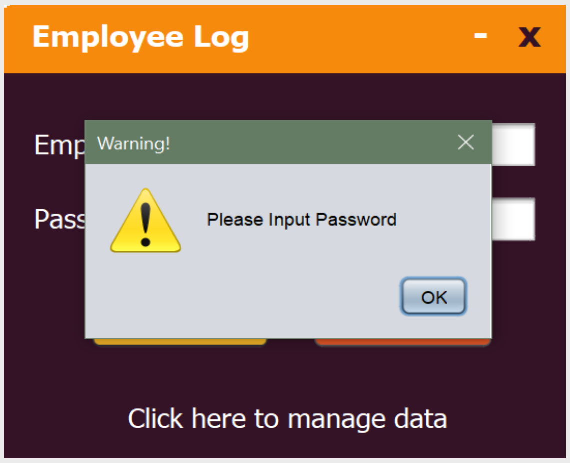




Time In Successful!

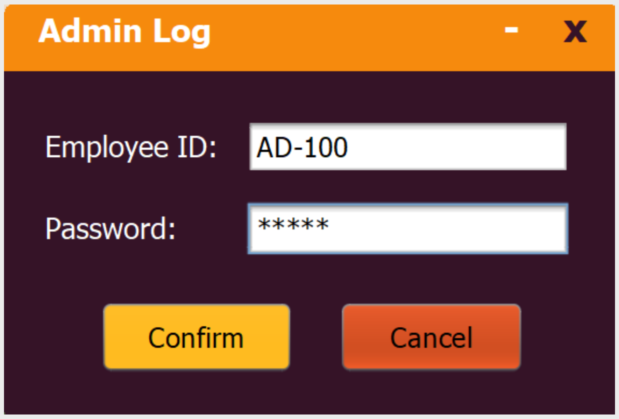


Please Input ID



Please Input Password

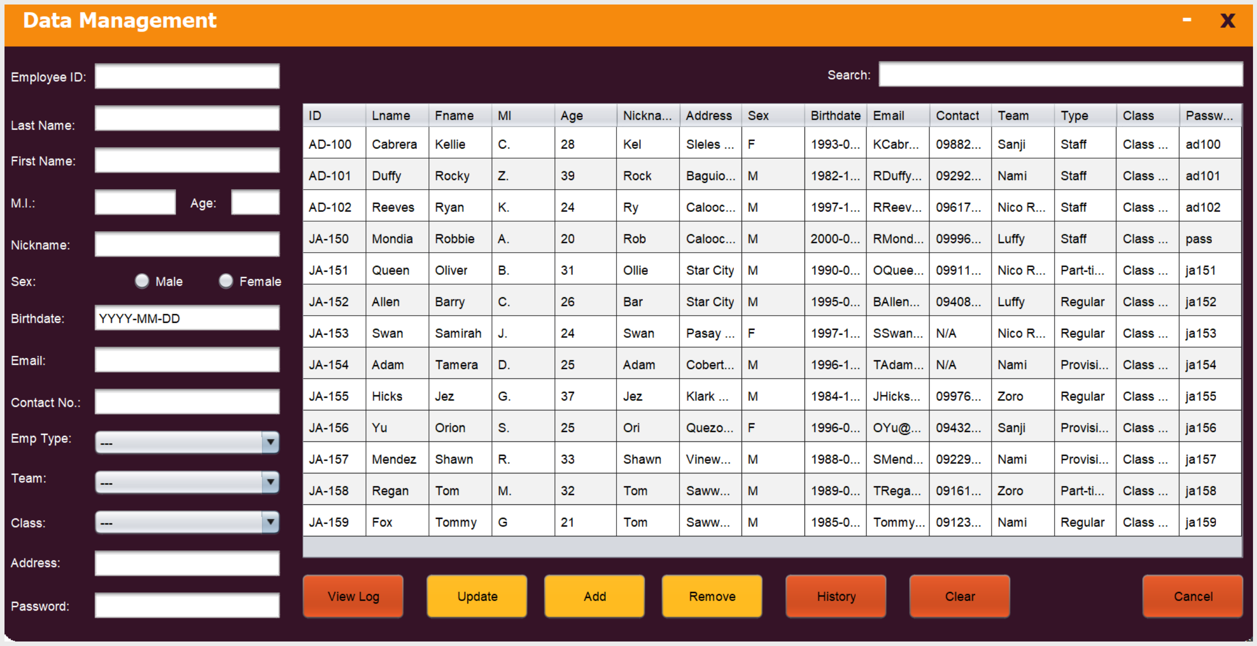
Admin Log – Users that was tagged as "Staff" are the ones that can access the data management table.



Employee ID: AD-100

Password: *****

Data Management – Window where admin/staff can add, delete, and update employee details.

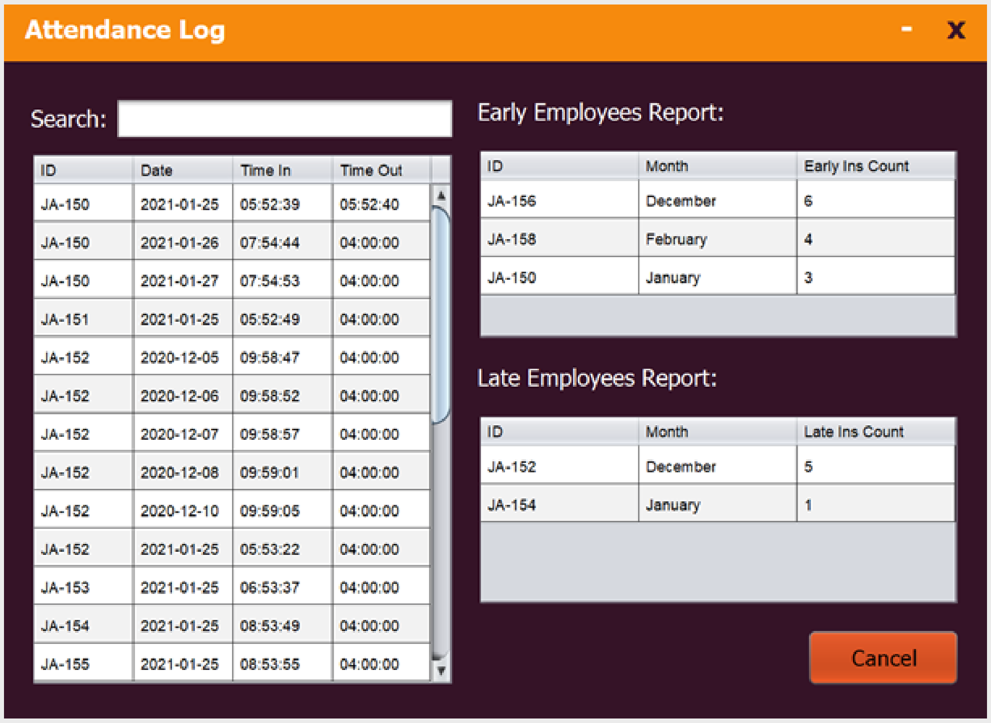


Employee ID:	
Last Name:	
First Name:	
M.I.:	
Age:	
Nickname:	
Sex:	
Birthdate:	YYYY-MM-DD
Email:	
Contact No.:	
Emp Type:	---
Team:	---
Class:	---
Address:	
Password:	

Search:

ID	Lname	Fname	MI	Age	Nickna...	Address	Sex	Birthdate	Email	Contact	Team	Type	Class	Passw...
AD-100	Cabrera	Kellie	C.	28	Kel	Sleles ...	F	1993-0...	KCabr...	09882...	Sanji	Staff	Class ...	ad100
AD-101	Duffy	Rocky	Z.	39	Rock	Baguio...	M	1982-1...	RDuffy...	09292...	Nami	Staff	Class ...	ad101
AD-102	Reeves	Ryan	K.	24	Ry	Calooc...	M	1997-1...	RReev...	09617...	Nico R...	Staff	Class ...	ad102
JA-150	Mondia	Robbie	A.	20	Rob	Calooc...	M	2000-0...	RMond...	09996...	Luffy	Staff	Class ...	pass
JA-151	Queen	Oliver	B.	31	Ollie	Star City	M	1990-0...	OQuee...	09911...	Nico R...	Part-ti...	Class ...	ja151
JA-152	Allen	Barry	C.	26	Bar	Star City	M	1995-0...	BAllen...	09408...	Luffy	Regular	Class ...	ja152
JA-153	Swan	Samirah	J.	24	Swan	Pasay ...	F	1997-1...	SSwan...	N/A	Nico R...	Regular	Class ...	ja153
JA-154	Adam	Tamera	D.	25	Adam	Cobert...	M	1996-1...	TAdam...	N/A	Nami	Provisi...	Class ...	ja154
JA-155	Hicks	Jez	G.	37	Jez	Klark ...	M	1984-1...	JHicks...	09976...	Zoro	Regular	Class ...	ja155
JA-156	Yu	Orion	S.	25	Ori	Quezo...	F	1996-0...	OYu@...	09432...	Sanji	Provisi...	Class ...	ja156
JA-157	Mendez	Shawn	R.	33	Shawn	Vinew...	M	1988-0...	SMend...	09229...	Nami	Provisi...	Class ...	ja157
JA-158	Regan	Tom	M.	32	Tom	Saww...	M	1989-0...	TRega...	09161...	Zoro	Part-ti...	Class ...	ja158
JA-159	Fox	Tommy	G	21	Tom	Saww...	M	1985-0...	Tommy...	09123...	Nami	Regular	Class ...	ja159

Attendance Log – Daily attendance log of employees and monthly most early ins and late report generation.


Attendance Log

Search:

ID	Date	Time In	Time Out
JA-150	2021-01-25	05:52:39	05:52:40
JA-150	2021-01-26	07:54:44	04:00:00
JA-150	2021-01-27	07:54:53	04:00:00
JA-151	2021-01-25	05:52:49	04:00:00
JA-152	2020-12-05	09:58:47	04:00:00
JA-152	2020-12-06	09:58:52	04:00:00
JA-152	2020-12-07	09:58:57	04:00:00
JA-152	2020-12-08	09:59:01	04:00:00
JA-152	2020-12-10	09:59:05	04:00:00
JA-152	2021-01-25	05:53:22	04:00:00
JA-153	2021-01-25	06:53:37	04:00:00
JA-154	2021-01-25	08:53:49	04:00:00
JA-155	2021-01-25	08:53:55	04:00:00

Early Employees Report:

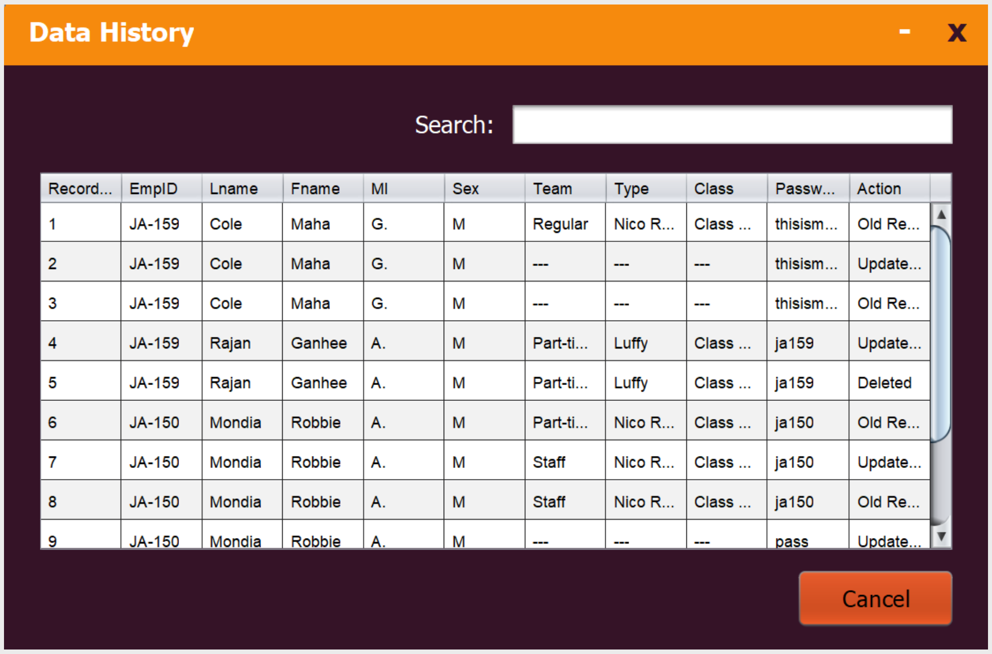
ID	Month	Early Ins Count
JA-156	December	6
JA-158	February	4
JA-150	January	3

Late Employees Report:

ID	Month	Late Ins Count
JA-152	December	5
JA-154	January	1

Cancel

Data History – Users that was tagged as "Staff" are the ones that can access the data management table.


Data History

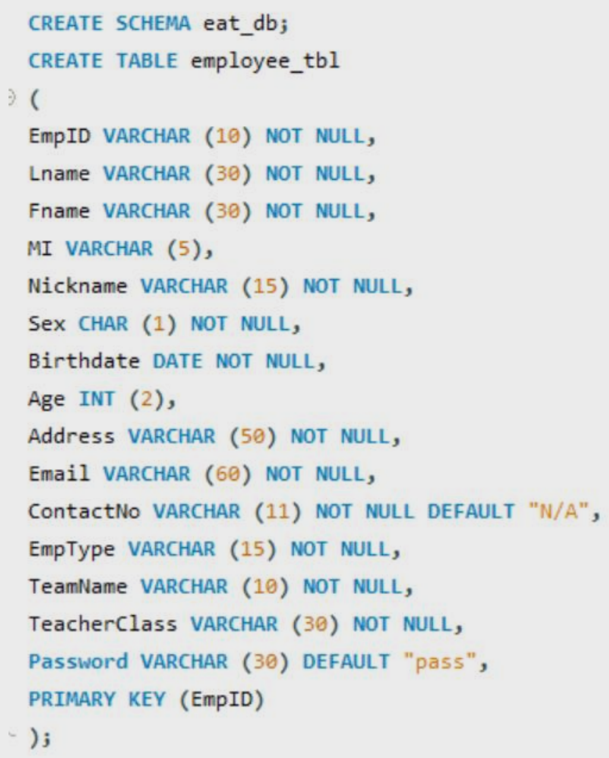
Search:

Record...	EmpID	Lname	Fname	MI	Sex	Team	Type	Class	Passw...	Action
1	JA-159	Cole	Maha	G.	M	Regular	Nico R...	Class ...	thisism...	Old Re...
2	JA-159	Cole	Maha	G.	M	---	---	---	thisism...	Update...
3	JA-159	Cole	Maha	G.	M	---	---	---	thisism...	Old Re...
4	JA-159	Rajan	Ganhee	A.	M	Part-ti...	Luffy	Class ...	ja159	Update...
5	JA-159	Rajan	Ganhee	A.	M	Part-ti...	Luffy	Class ...	ja159	Deleted
6	JA-150	Mondia	Robbie	A.	M	Part-ti...	Nico R...	Class ...	ja150	Old Re...
7	JA-150	Mondia	Robbie	A.	M	Staff	Nico R...	Class ...	ja150	Update...
8	JA-150	Mondia	Robbie	A.	M	Staff	Nico R...	Class ...	ja150	Old Re...
9	JA-150	Mondia	Robbie	A.	M	---	---	---	pass	Update...

Cancel

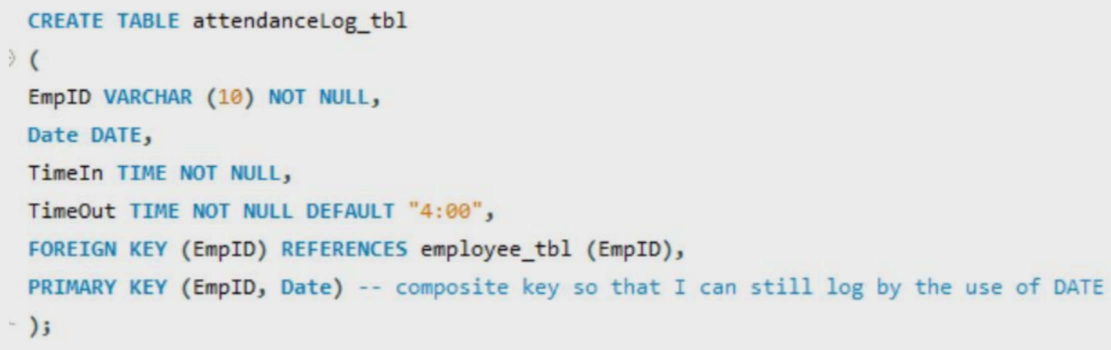
SQL Queries and Output

Employee Table



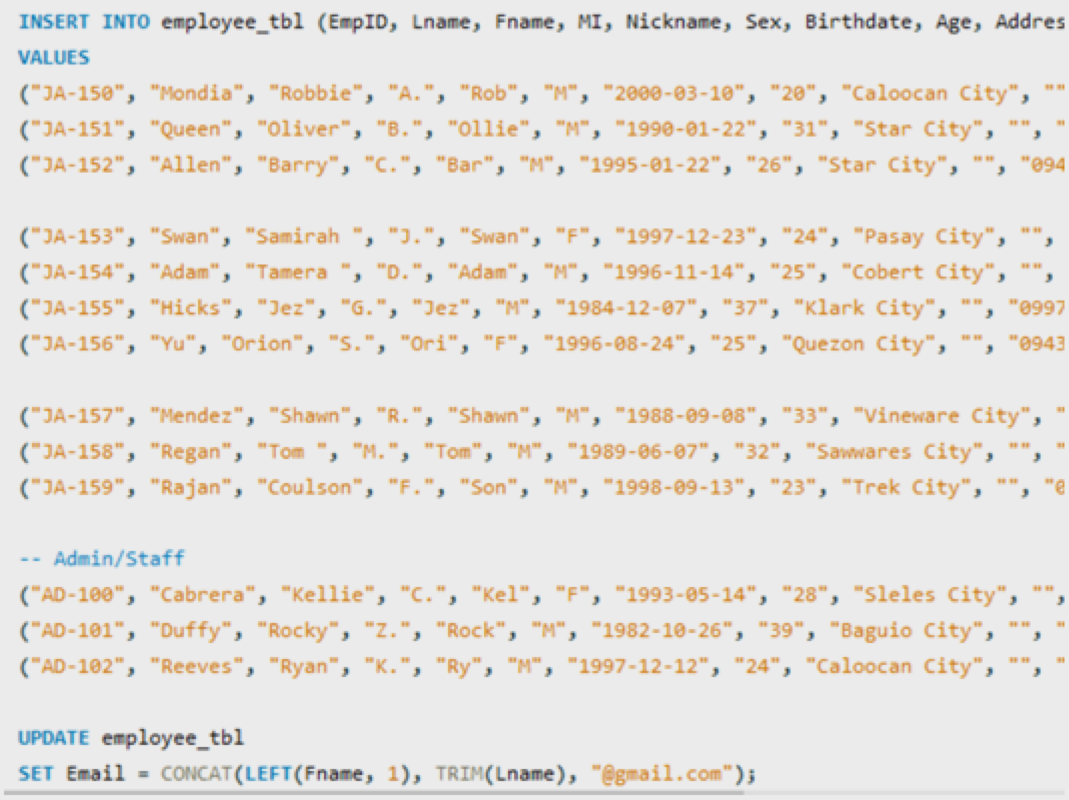
```sql
CREATE SCHEMA eat_db;
CREATE TABLE employee_tbl
(
EmpID VARCHAR (10) NOT NULL,
Lname VARCHAR (30) NOT NULL,
Fname VARCHAR (30) NOT NULL,
MI VARCHAR (5),
Nickname VARCHAR (15) NOT NULL,
Sex CHAR (1) NOT NULL,
Birthdate DATE NOT NULL,
Age INT (2),
Address VARCHAR (50) NOT NULL,
Email VARCHAR (60) NOT NULL,
ContactNo VARCHAR (11) NOT NULL DEFAULT "N/A",
EmpType VARCHAR (15) NOT NULL,
TeamName VARCHAR (10) NOT NULL,
TeacherClass VARCHAR (30) NOT NULL,
Password VARCHAR (30) DEFAULT "pass",
PRIMARY KEY (EmpID)
);
```

Attendance Log Table



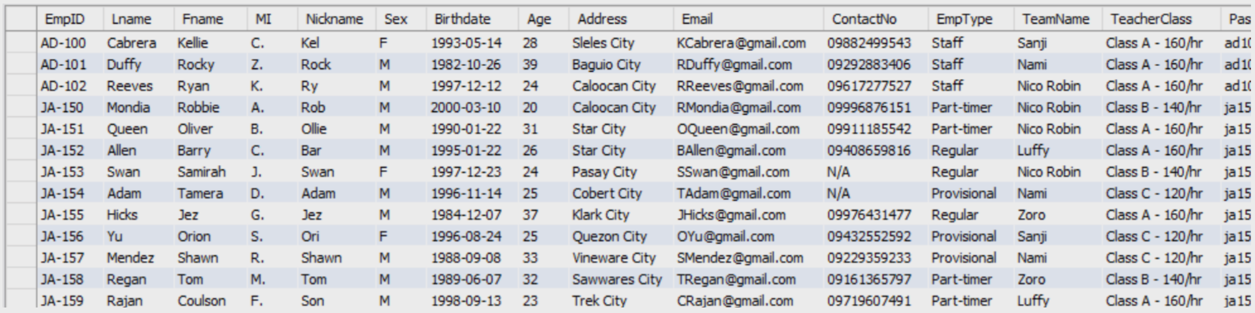
```sql
CREATE TABLE attendanceLog_tbl
(
EmpID VARCHAR (10) NOT NULL,
Date DATE,
TimeIn TIME NOT NULL,
TimeOut TIME NOT NULL DEFAULT "4:00",
FOREIGN KEY (EmpID) REFERENCES employee_tbl (EmpID),
PRIMARY KEY (EmpID, Date) -- composite key so that I can still log by the use of DATE
);
```

Test Data



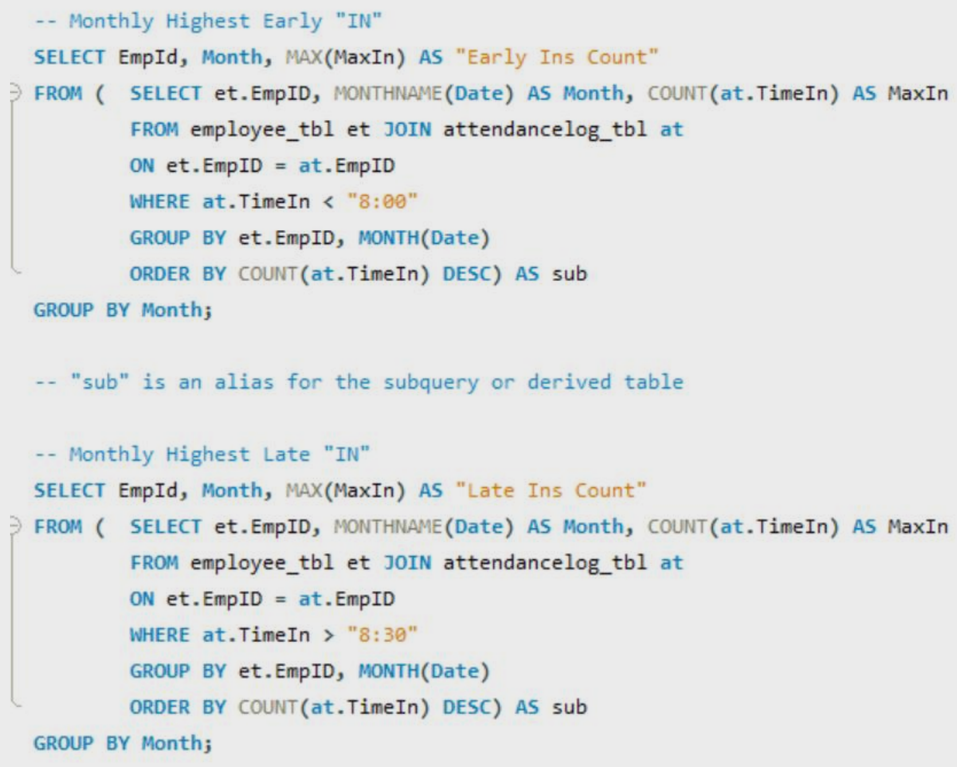
```
INSERT INTO employee_tbl (EmpID, Lname, Fname, MI, Nickname, Sex, Birthdate, Age, Addres
VALUES
("JA-150", "Mondia", "Robbie", "A.", "Rob", "M", "2000-03-10", "20", "Caloocan City", ""
("JA-151", "Queen", "Oliver", "B.", "Ollie", "M", "1990-01-22", "31", "Star City", "", "
("JA-152", "Allen", "Barry", "C.", "Bar", "M", "1995-01-22", "26", "Star City", "", "094

("JA-153", "Swan", "Samirah ", "J.", "Swan", "F", "1997-12-23", "24", "Pasay City", "",
("JA-154", "Adam", "Tamera ", "D.", "Adam", "M", "1996-11-14", "25", "Cobert City", "",
("JA-155", "Hicks", "Jez", "G.", "Jez", "M", "1984-12-07", "37", "Klark City", "", "0997
("JA-156", "Yu", "Orion", "S.", "Ori", "F", "1996-08-24", "25", "Quezon City", "", "0943

("JA-157", "Mendez", "Shawn", "R.", "Shawn", "M", "1988-09-08", "33", "Vineware City", "
("JA-158", "Regan", "Tom ", "M.", "Tom", "M", "1989-06-07", "32", "Sawwares City", "", "
("JA-159", "Rajan", "Coulson", "F.", "Son", "M", "1998-09-13", "23", "Trek City", "", "0

-- Admin/Staff
("AD-100", "Cabrera", "Kellie", "C.", "Kel", "F", "1993-05-14", "28", "Sleles City", "",
("AD-101", "Duffy", "Rocky", "Z.", "Rock", "M", "1982-10-26", "39", "Baguio City", "", "
("AD-102", "Reeves", "Ryan", "K.", "Ry", "M", "1997-12-12", "24", "Caloocan City", "", "

UPDATE employee_tbl
SET Email = CONCAT(LEFT(Fname, 1), TRIM(Lname), "@gmail.com");
```

Employee Table + Test Data Output



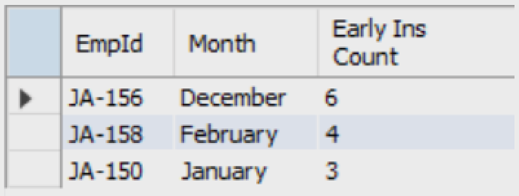
EmpID	Lname	Fname	MI	Nickname	Sex	Birthdate	Age	Address	Email	ContactNo	EmpType	TeamName	TeacherClass	Pas
AD-100	Cabrera	Kellie	C.	Kel	F	1993-05-14	28	Sleles City	KCabrera@gmail.com	09882499543	Staff	Sanji	Class A - 160/hr	ad1
AD-101	Duffy	Rocky	Z.	Rock	M	1982-10-26	39	Baguio City	RDuffy@gmail.com	09292883406	Staff	Nami	Class A - 160/hr	ad1
AD-102	Reeves	Ryan	K.	Ry	M	1997-12-12	24	Caloocan City	RReeves@gmail.com	09617277527	Staff	Nico Robin	Class A - 160/hr	ad1
JA-150	Mondia	Robbie	A.	Rob	M	2000-03-10	20	Caloocan City	RMondia@gmail.com	09996876151	Part-timer	Nico Robin	Class B - 140/hr	ja15
JA-151	Queen	Oliver	B.	Ollie	M	1990-01-22	31	Star City	OQueen@gmail.com	09911185542	Part-timer	Nico Robin	Class A - 160/hr	ja15
JA-152	Allen	Barry	C.	Bar	M	1995-01-22	26	Star City	BAllen@gmail.com	09408659816	Regular	Luffy	Class A - 160/hr	ja15
JA-153	Swan	Samirah	J.	Swan	F	1997-12-23	24	Pasay City	SSwan@gmail.com	N/A	Regular	Nico Robin	Class B - 140/hr	ja15
JA-154	Adam	Tamera	D.	Adam	M	1996-11-14	25	Cobert City	TAdam@gmail.com	N/A	Provisional	Nami	Class C - 120/hr	ja15
JA-155	Hicks	Jez	G.	Jez	M	1984-12-07	37	Klark City	JHicks@gmail.com	09976431477	Regular	Zoro	Class A - 160/hr	ja15
JA-156	Yu	Orion	S.	Ori	F	1996-08-24	25	Quezon City	OYu@gmail.com	09432552592	Provisional	Sanji	Class C - 120/hr	ja15
JA-157	Mendez	Shawn	R.	Shawn	M	1988-09-08	33	Vineware City	SMendez@gmail.com	09229359233	Provisional	Nami	Class C - 120/hr	ja15
JA-158	Regan	Tom	M.	Tom	M	1989-06-07	32	Sawwares City	TRegan@gmail.com	09161365797	Part-timer	Zoro	Class B - 140/hr	ja15
JA-159	Rajan	Coulson	F.	Son	M	1998-09-13	23	Trek City	CRajan@gmail.com	09719607491	Part-timer	Luffy	Class A - 160/hr	ja15

Join Statements + Subquery



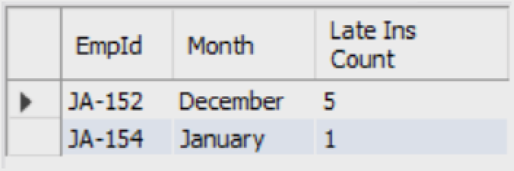
```
-- Monthly Highest Early "IN"
SELECT EmpId, Month, MAX(MaxIn) AS "Early Ins Count"
FROM (  SELECT et.EmpID, MONTHNAME(Date) AS Month, COUNT(at.TimeIn) AS MaxIn
        FROM employee_tbl et JOIN attendancelog_tbl at
        ON et.EmpID = at.EmpID
        WHERE at.TimeIn < "8:00"
        GROUP BY et.EmpID, MONTH(Date)
        ORDER BY COUNT(at.TimeIn) DESC) AS sub
GROUP BY Month;

-- "sub" is an alias for the subquery or derived table

-- Monthly Highest Late "IN"
SELECT EmpId, Month, MAX(MaxIn) AS "Late Ins Count"
FROM (  SELECT et.EmpID, MONTHNAME(Date) AS Month, COUNT(at.TimeIn) AS MaxIn
        FROM employee_tbl et JOIN attendancelog_tbl at
        ON et.EmpID = at.EmpID
        WHERE at.TimeIn > "8:30"
        GROUP BY et.EmpID, MONTH(Date)
        ORDER BY COUNT(at.TimeIn) DESC) AS sub
GROUP BY Month;
```

Month Highest Early IN Count Output



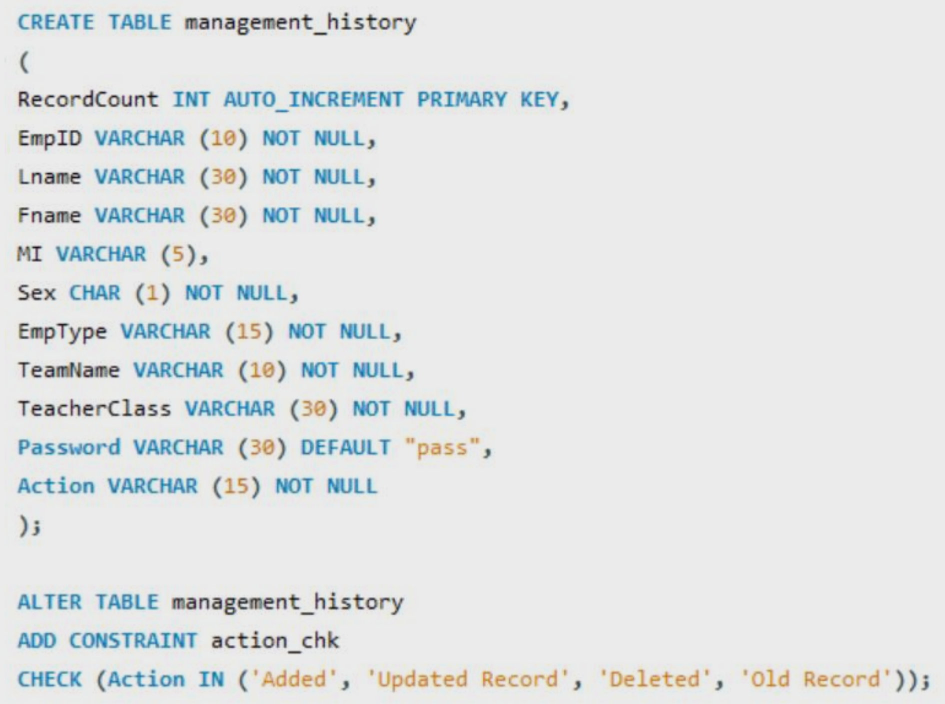
	EmpId	Month	Early Ins Count
▶	JA-156	December	6
	JA-158	February	4
	JA-150	January	3

Month Highest Late IN Count Output



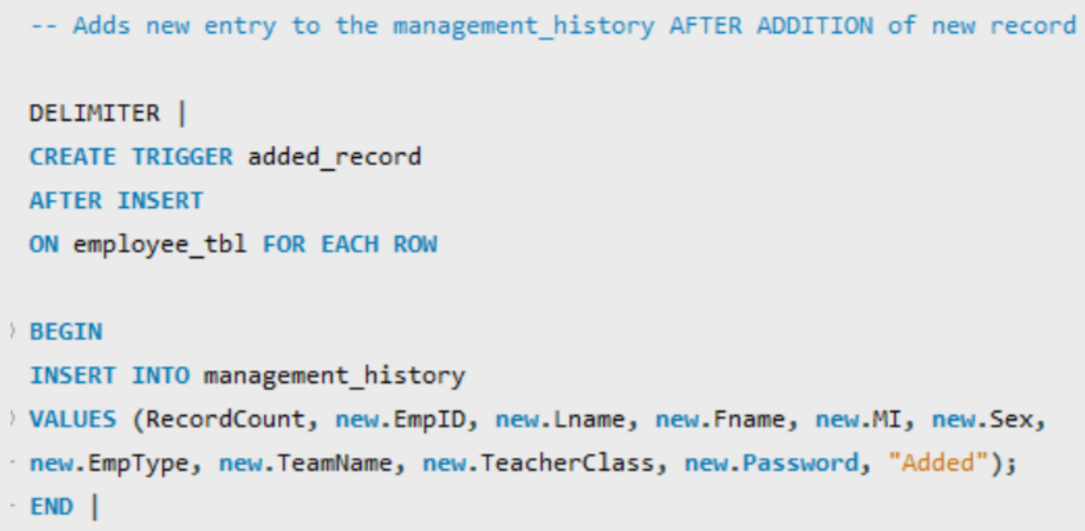
	EmpId	Month	Late Ins Count
▶	JA-152	December	5
	JA-154	January	1

Trigger Table



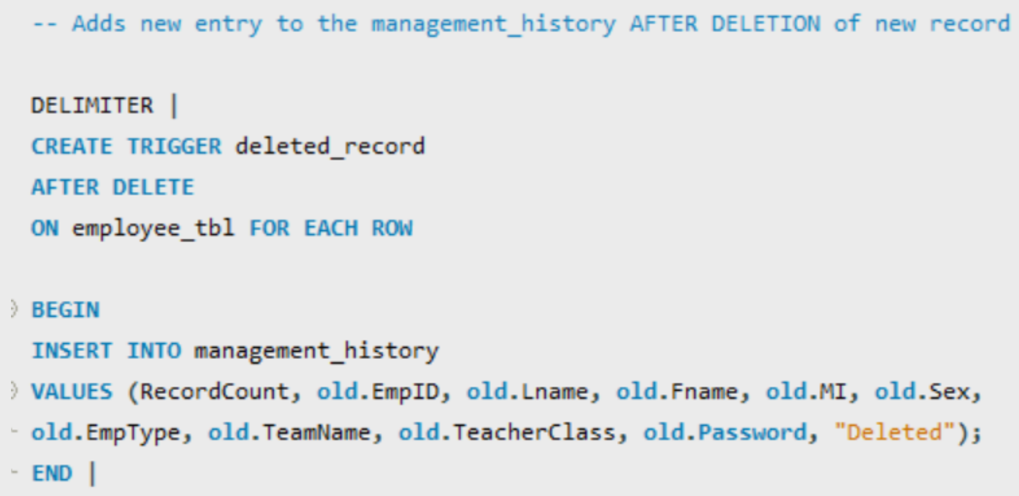
```
CREATE TABLE management_history
(
RecordCount INT AUTO_INCREMENT PRIMARY KEY,
EmpID VARCHAR (10) NOT NULL,
Lname VARCHAR (30) NOT NULL,
Fname VARCHAR (30) NOT NULL,
MI VARCHAR (5),
Sex CHAR (1) NOT NULL,
EmpType VARCHAR (15) NOT NULL,
TeamName VARCHAR (10) NOT NULL,
TeacherClass VARCHAR (30) NOT NULL,
Password VARCHAR (30) DEFAULT "pass",
Action VARCHAR (15) NOT NULL
);

ALTER TABLE management_history
ADD CONSTRAINT action_chk
CHECK (Action IN ('Added', 'Updated Record', 'Deleted', 'Old Record'));
```

Add Record Trigger



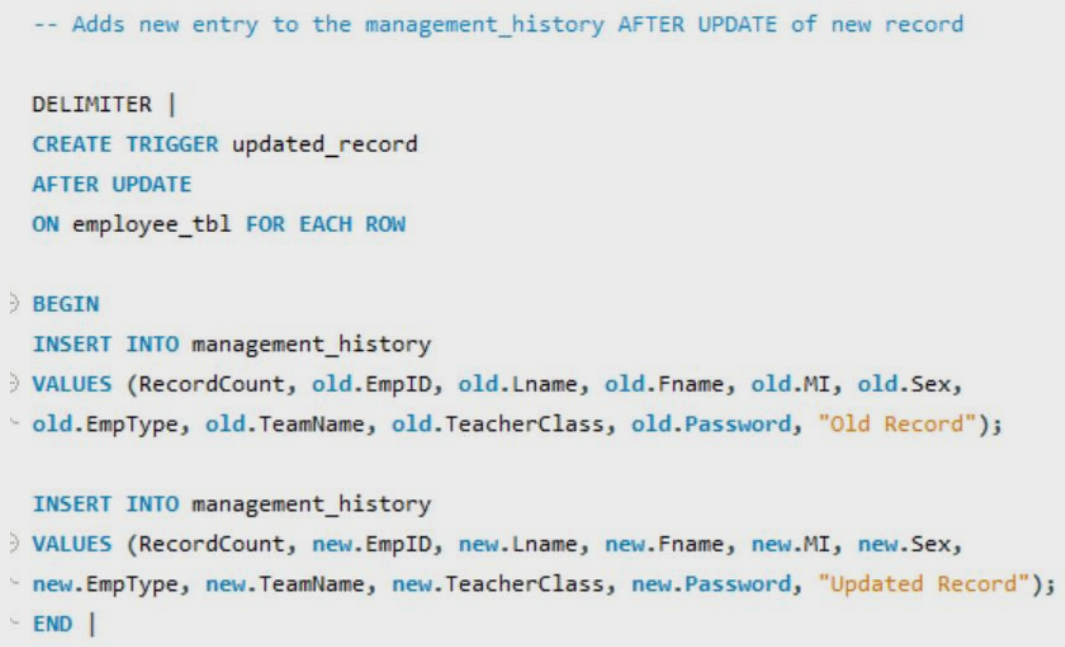
```
-- Adds new entry to the management_history AFTER ADDITION of new record

DELIMITER |
CREATE TRIGGER added_record
AFTER INSERT
ON employee_tbl FOR EACH ROW

BEGIN
INSERT INTO management_history
VALUES (RecordCount, new.EmpID, new.Lname, new.Fname, new.MI, new.Sex,
new.EmpType, new.TeamName, new.TeacherClass, new.Password, "Added");
END |
```

Delete Record Trigger



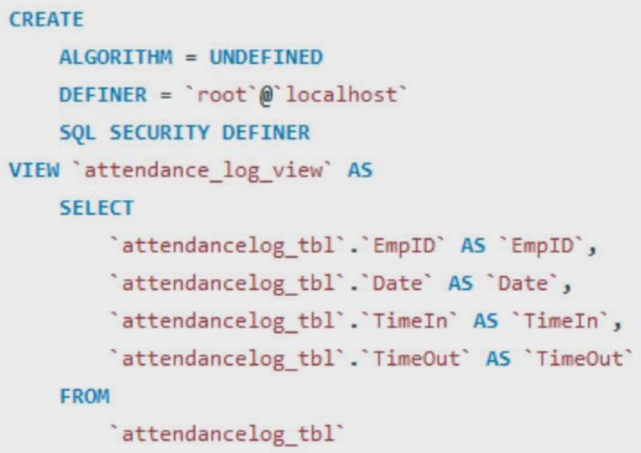
```
-- Adds new entry to the management_history AFTER DELETION of new record

DELIMITER |
CREATE TRIGGER deleted_record
AFTER DELETE
ON employee_tbl FOR EACH ROW

BEGIN
INSERT INTO management_history
VALUES (RecordCount, old.EmpID, old.Lname, old.Fname, old.MI, old.Sex,
old.EmpType, old.TeamName, old.TeacherClass, old.Password, "Deleted");
END |
```

Update Record Trigger



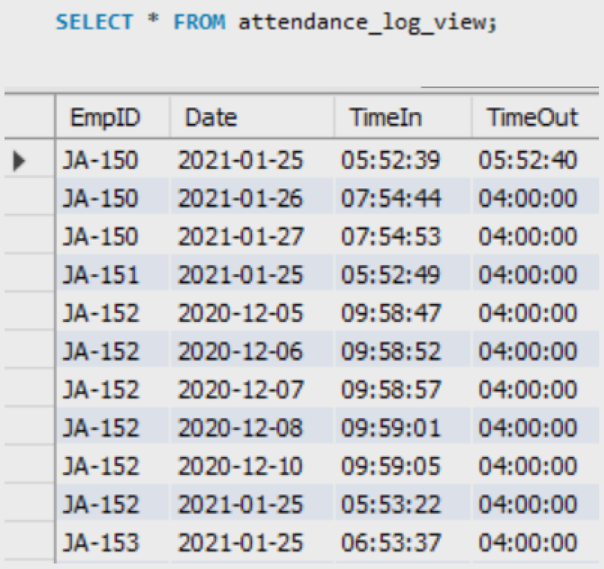
```
-- Adds new entry to the management_history AFTER UPDATE of new record

DELIMITER |
CREATE TRIGGER updated_record
AFTER UPDATE
ON employee_tbl FOR EACH ROW

BEGIN
INSERT INTO management_history
VALUES (RecordCount, old.EmpID, old.Lname, old.Fname, old.MI, old.Sex,
old.EmpType, old.TeamName, old.TeacherClass, old.Password, "Old Record");

INSERT INTO management_history
VALUES (RecordCount, new.EmpID, new.Lname, new.Fname, new.MI, new.Sex,
new.EmpType, new.TeamName, new.TeacherClass, new.Password, "Updated Record");
END |
```

Attendance Log View



```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `attendance_log_view` AS
    SELECT
        `attendancelog_tbl`.`EmpID` AS `EmpID`,
        `attendancelog_tbl`.`Date` AS `Date`,
        `attendancelog_tbl`.`TimeIn` AS `TimeIn`,
        `attendancelog_tbl`.`TimeOut` AS `TimeOut`
    FROM
        `attendancelog_tbl`
```

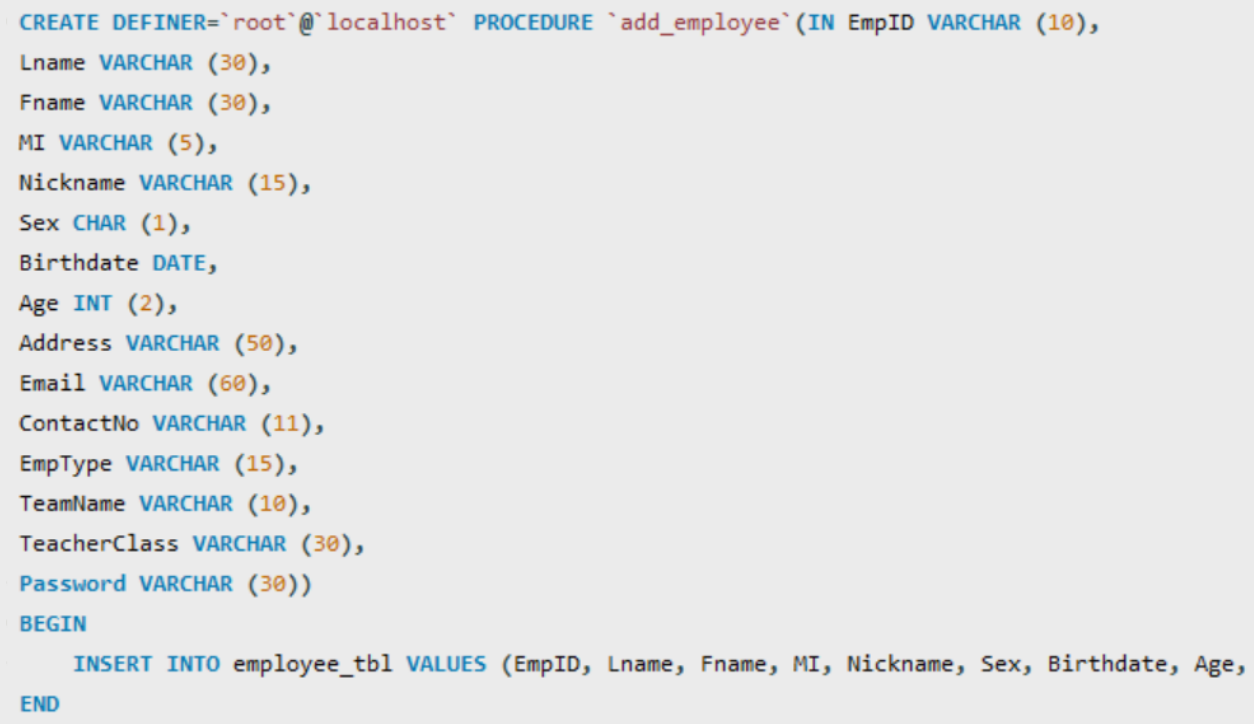
Attendance Log View Output



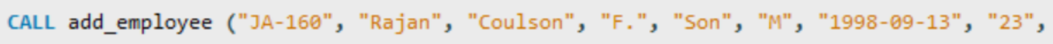
```
SELECT * FROM attendance_log_view;
```

EmpID	Date	TimeIn	TimeOut
JA-150	2021-01-25	05:52:39	05:52:40
JA-150	2021-01-26	07:54:44	04:00:00
JA-150	2021-01-27	07:54:53	04:00:00
JA-151	2021-01-25	05:52:49	04:00:00
JA-152	2020-12-05	09:58:47	04:00:00
JA-152	2020-12-06	09:58:52	04:00:00
JA-152	2020-12-07	09:58:57	04:00:00
JA-152	2020-12-08	09:59:01	04:00:00
JA-152	2020-12-10	09:59:05	04:00:00
JA-152	2021-01-25	05:53:22	04:00:00
JA-153	2021-01-25	06:53:37	04:00:00

Stored Procedure: Add (Insert) Employee

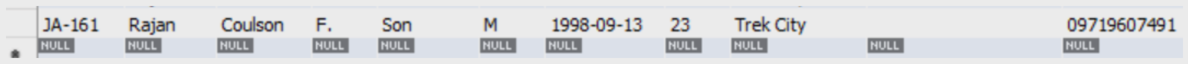


```
CREATE DEFINER=`root`@`localhost` PROCEDURE `add_employee`(IN EmpID VARCHAR (10),
Lname VARCHAR (30),
Fname VARCHAR (30),
MI VARCHAR (5),
Nickname VARCHAR (15),
Sex CHAR (1),
Birthdate DATE,
Age INT (2),
Address VARCHAR (50),
Email VARCHAR (60),
ContactNo VARCHAR (11),
EmpType VARCHAR (15),
TeamName VARCHAR (10),
TeacherClass VARCHAR (30),
Password VARCHAR (30))
BEGIN
    INSERT INTO employee_tbl VALUES (EmpID, Lname, Fname, MI, Nickname, Sex, Birthdate, Age,
END
```



```
CALL add_employee ("JA-160", "Rajan", "Coulson", "F.", "Son", "M", "1998-09-13", "23",
```

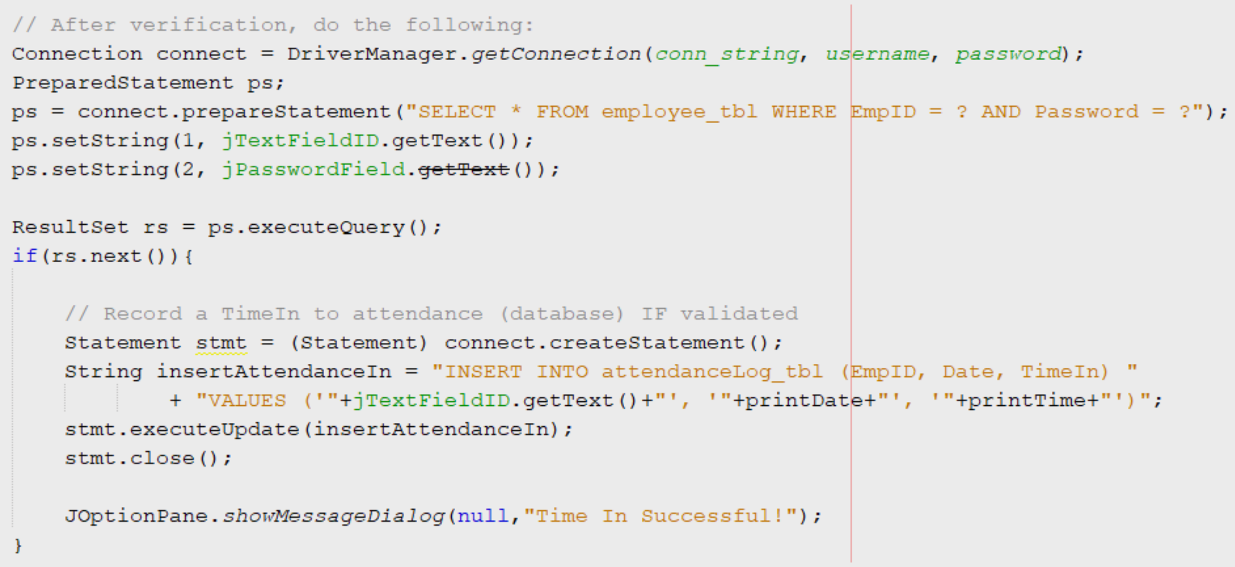
Employee Table Record



JA-161	Rajan	Coulson	F.	Son	M	1998-09-13	23	Trek City		09719607491
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

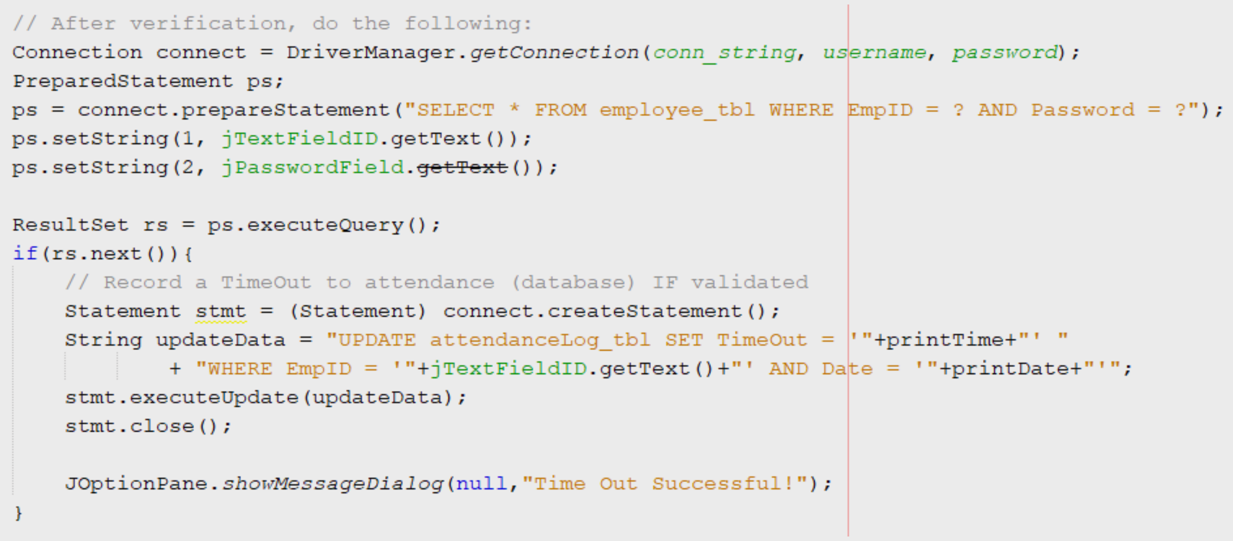
Java Codes + SQL Queries

Time In Button – Inserts record to attendanceLog_tbl after validation of EmpID and Password.



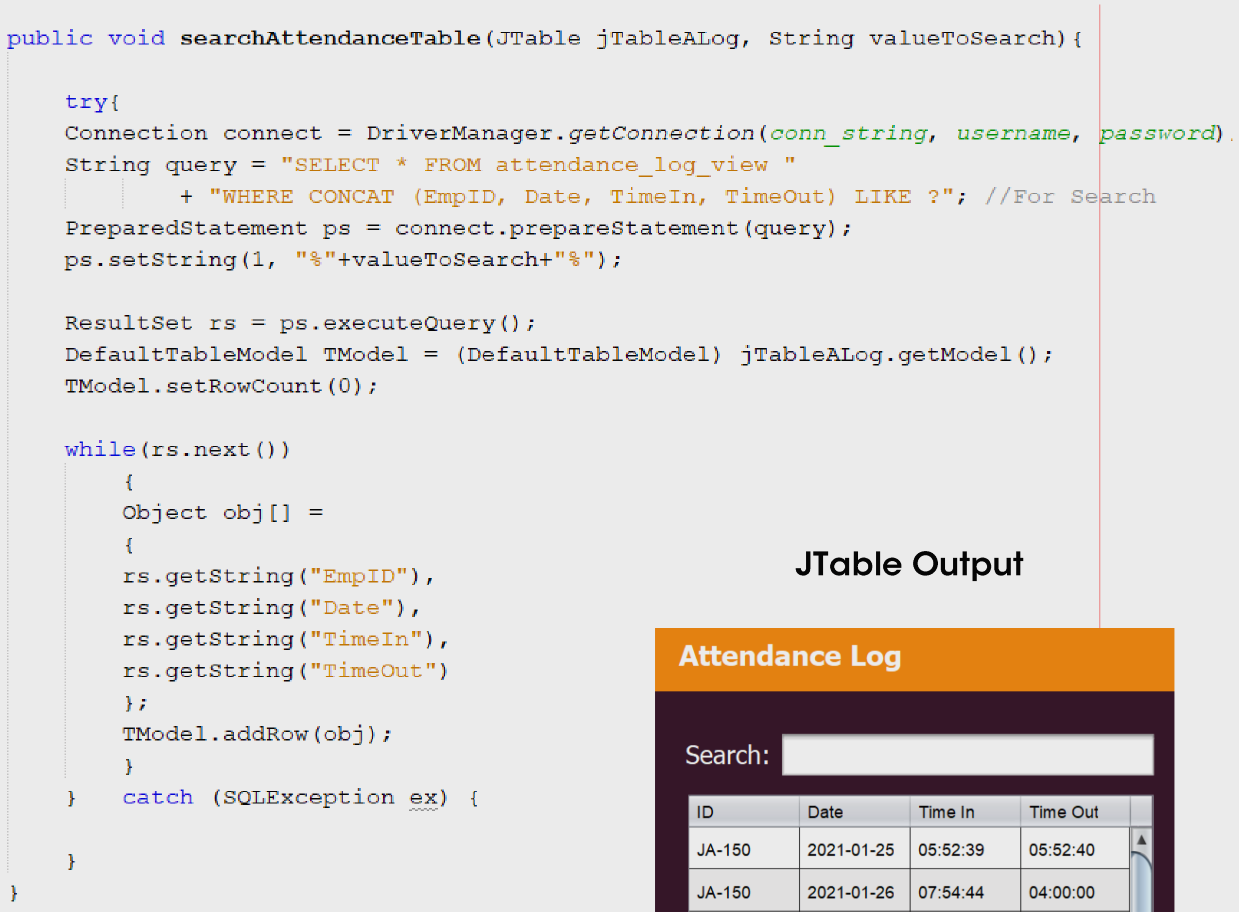
```
// After verification, do the following:
Connection connect = DriverManager.getConnection(conn_string, username, password);
PreparedStatement ps;
ps = connect.prepareStatement("SELECT * FROM employee_tbl WHERE EmpID = ? AND Password = ?");
ps.setString(1, jTextFieldID.getText());
ps.setString(2, jPasswordField.~~getText~~());

ResultSet rs = ps.executeQuery();
if(rs.next()){

    // Record a TimeIn to attendance (database) IF validated
    Statement stmt = (Statement) connect.createStatement();
    String insertAttendanceIn = "INSERT INTO attendanceLog_tbl (EmpID, Date, TimeIn) "
            + "VALUES ('"+jTextFieldID.getText()+"', '"+printDate+"', '"+printTime+"')";
    stmt.executeUpdate(insertAttendanceIn);
    stmt.close();

    JOptionPane.showMessageDialog(null,"Time In Successful!");
}
```

Time Out Button – Updates attendanceLog_tbl's TimeOut column after validation of EmpID and Password.



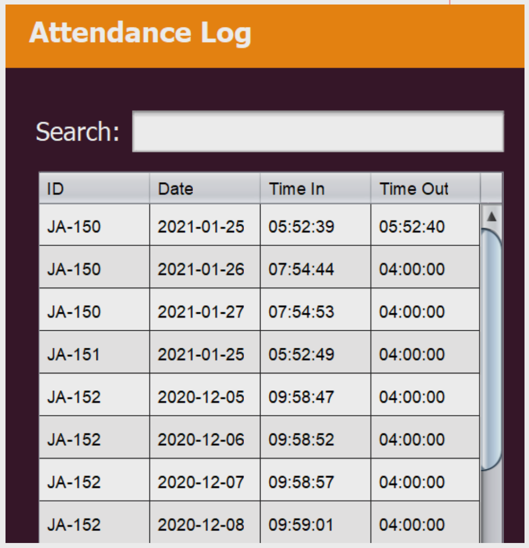
```
// After verification, do the following:
Connection connect = DriverManager.getConnection(conn_string, username, password);
PreparedStatement ps;
ps = connect.prepareStatement("SELECT * FROM employee_tbl WHERE EmpID = ? AND Password = ?");
ps.setString(1, jTextFieldID.getText());
ps.setString(2, jPasswordField.~~getText~~());

ResultSet rs = ps.executeQuery();
if(rs.next()){
    // Record a TimeOut to attendance (database) IF validated
    Statement stmt = (Statement) connect.createStatement();
    String updateData = "UPDATE attendanceLog_tbl SET TimeOut = '"+printTime+"' "
            + "WHERE EmpID = '"+jTextFieldID.getText()+"' AND Date = '"+printDate+"'";
    stmt.executeUpdate(updateData);
    stmt.close();

    JOptionPane.showMessageDialog(null,"Time Out Successful!");
}
```

Attendance Log – ResultSet will scan the selected view (attendance_log_view) then display the results to the table in the program.



```
public void searchAttendanceTable(JTable jTableALog, String valueToSearch){

    try{
    Connection connect = DriverManager.getConnection(conn_string, username, password)
    String query = "SELECT * FROM attendance_log_view "
            + "WHERE CONCAT (EmpID, Date, TimeIn, TimeOut) LIKE ?"; //For Search
    PreparedStatement ps = connect.prepareStatement(query);
    ps.setString(1, "%"+valueToSearch+"%");

    ResultSet rs = ps.executeQuery();
    DefaultTableModel TModel = (DefaultTableModel) jTableALog.getModel();
    TModel.setRowCount(0);

    while(rs.next())
        {
        Object obj[] =
        {
        rs.getString("EmpID"),
        rs.getString("Date"),
        rs.getString("TimeIn"),
        rs.getString("TimeOut")
        };
        TModel.addRow(obj);
        }
    }   catch (SQLException ex) {

    }
}
```

JTable Output

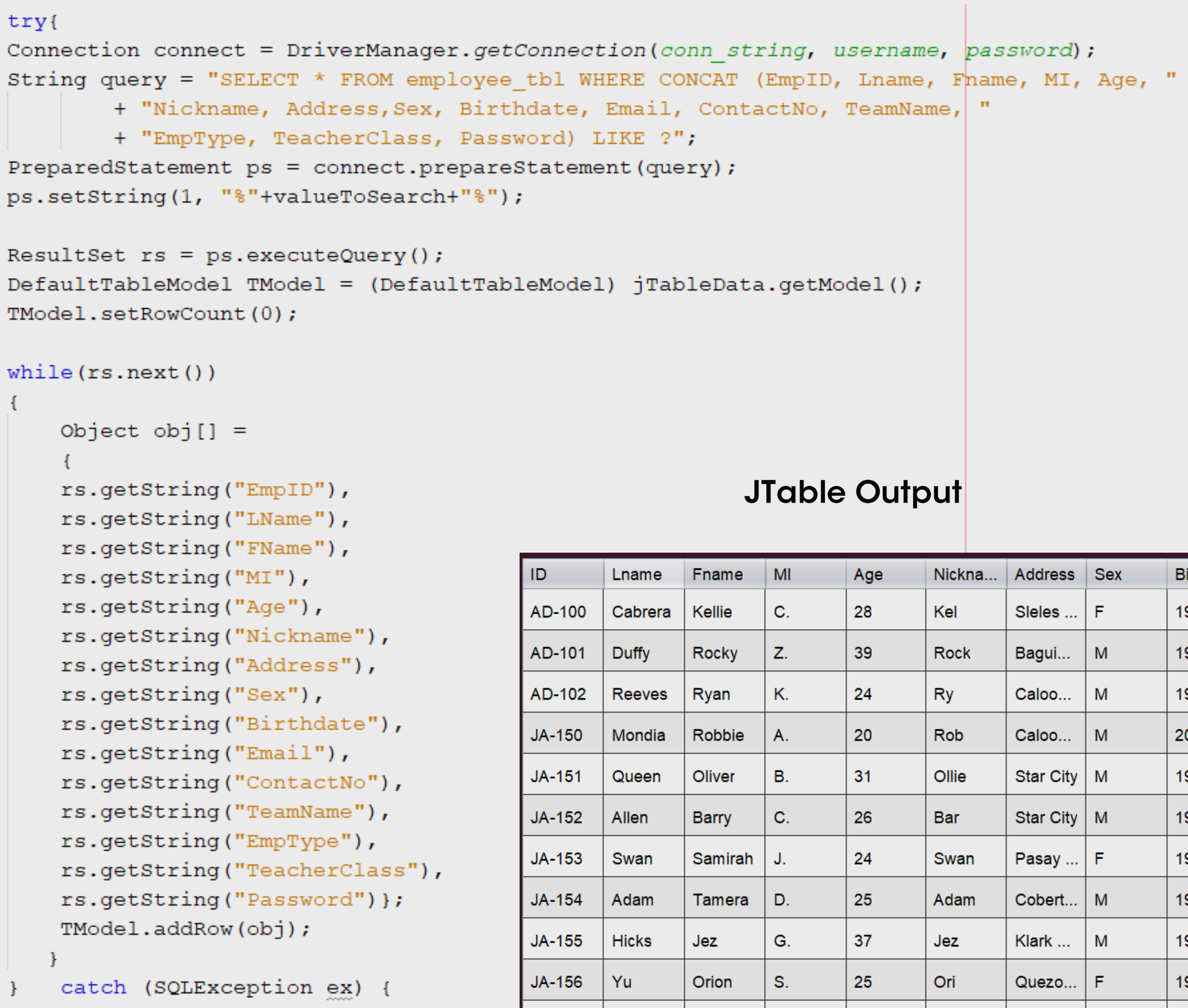


Attendance Log

Search:

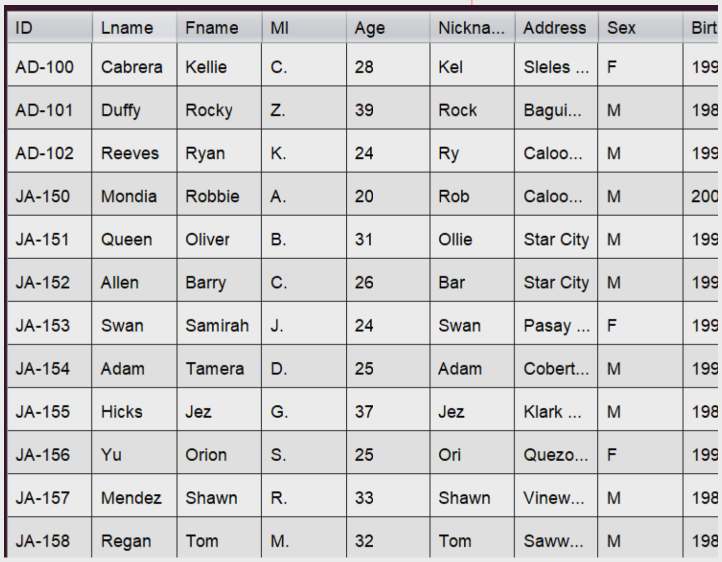
ID	Date	Time In	Time Out
JA-150	2021-01-25	05:52:39	05:52:40
JA-150	2021-01-26	07:54:44	04:00:00
JA-150	2021-01-27	07:54:53	04:00:00
JA-151	2021-01-25	05:52:49	04:00:00
JA-152	2020-12-05	09:58:47	04:00:00
JA-152	2020-12-06	09:58:52	04:00:00
JA-152	2020-12-07	09:58:57	04:00:00
JA-152	2020-12-08	09:59:01	04:00:00

Data Management Table – ResultSet will scan the selected table (employee_tbl) then display the results to the table in the program.



```
try{
Connection connect = DriverManager.getConnection(conn_string, username, password);
String query = "SELECT * FROM employee_tbl WHERE CONCAT (EmpID, Lname, Fname, MI, Age, "
        + "Nickname, Address,Sex, Birthdate, Email, ContactNo, TeamName, "
        + "EmpType, TeacherClass, Password) LIKE ?";
PreparedStatement ps = connect.prepareStatement(query);
ps.setString(1, "%"+valueToSearch+"%");

ResultSet rs = ps.executeQuery();
DefaultTableModel TModel = (DefaultTableModel) jTableData.getModel();
TModel.setRowCount(0);

while(rs.next())
{
    Object obj[] =
    {
    rs.getString("EmpID"),
    rs.getString("LName"),
    rs.getString("FName"),
    rs.getString("MI"),
    rs.getString("Age"),
    rs.getString("Nickname"),
    rs.getString("Address"),
    rs.getString("Sex"),
    rs.getString("Birthdate"),
    rs.getString("Email"),
    rs.getString("ContactNo"),
    rs.getString("TeamName"),
    rs.getString("EmpType"),
    rs.getString("TeacherClass"),
    rs.getString("Password")};
    TModel.addRow(obj);
   }
}   catch (SQLException ex) {
```

JTable Output

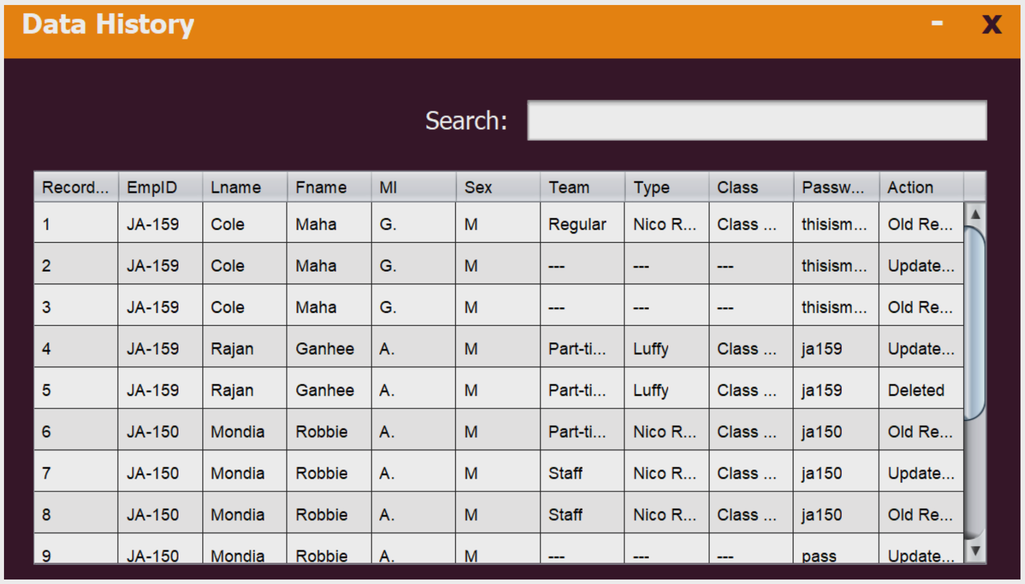


ID	Lname	Fname	MI	Age	Nickna...	Address	Sex	Birt
AD-100	Cabrera	Kellie	C.	28	Kel	Sleles ...	F	199
AD-101	Duffy	Rocky	Z.	39	Rock	Bagui...	M	198
AD-102	Reeves	Ryan	K.	24	Ry	Caloo...	M	199
JA-150	Mondia	Robbie	A.	20	Rob	Caloo...	M	200
JA-151	Queen	Oliver	B.	31	Ollie	Star City	M	199
JA-152	Allen	Barry	C.	26	Bar	Star City	M	199
JA-153	Swan	Samirah	J.	24	Swan	Pasay ...	F	199
JA-154	Adam	Tamera	D.	25	Adam	Cobert...	M	199
JA-155	Hicks	Jez	G.	37	Jez	Klark ...	M	198
JA-156	Yu	Orion	S.	25	Ori	Quezo...	F	199
JA-157	Mendez	Shawn	R.	33	Shawn	Vinew...	M	198
JA-158	Regan	Tom	M.	32	Tom	Saww...	M	198

History – ResultSet will scan the selected table (management_history) then display the results to the table in the program.

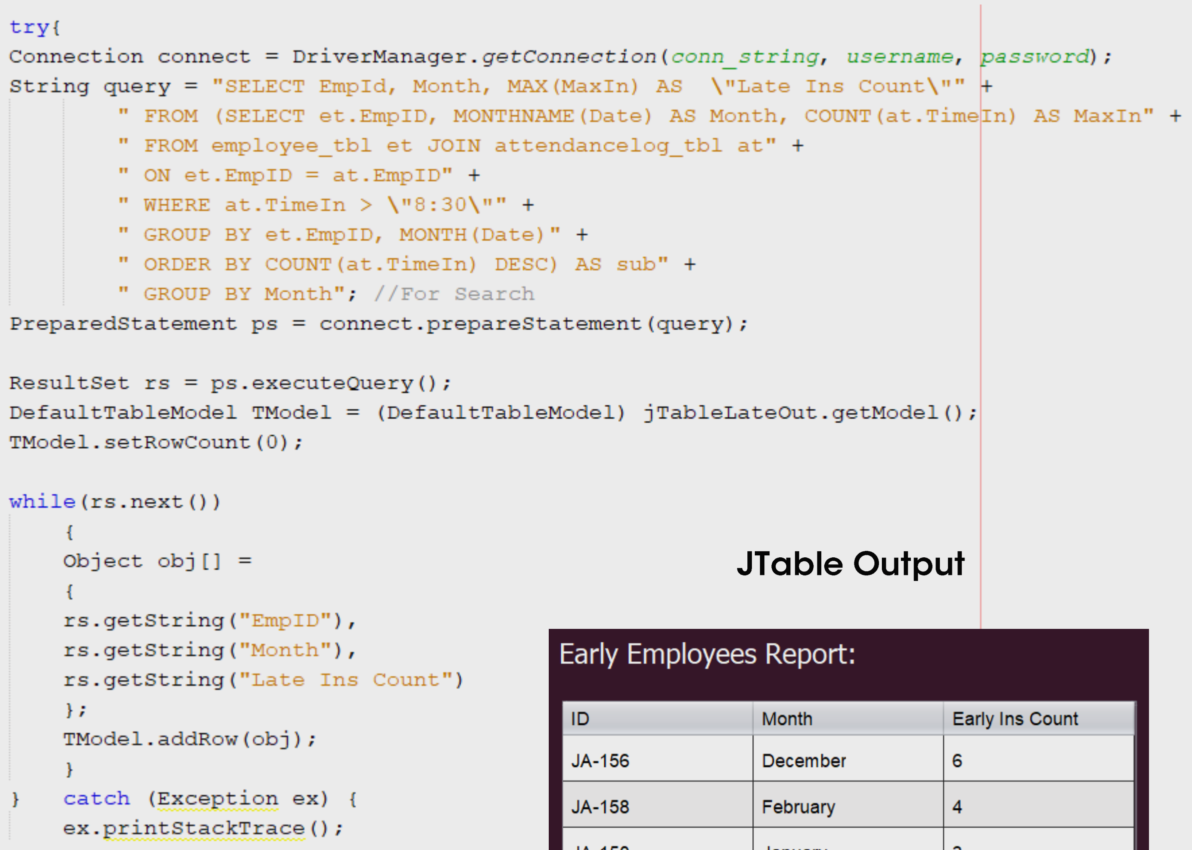
```
try{
Connection connect = DriverManager.getConnection(conn_string, username, password);
String query = "SELECT * FROM management_history "
        + "WHERE CONCAT (RecordCount, EmpID, Lname, Fname, MI, Sex, EmpType, TeamName, "
        + "TeacherClass, Password, Action) LIKE ?"; //For Search
PreparedStatement ps = connect.prepareStatement(query);
ps.setString(1, "%"+valueToSearch+"%");

ResultSet rs = ps.executeQuery();
DefaultTableModel TModel = (DefaultTableModel) jTableALog.getModel();
TModel.setRowCount(0);

while(rs.next())
    {
    Object obj[] =
    {
    rs.getString("RecordCount"),
    rs.getString("EmpID"),
    rs.getString("Lname"),
    rs.getString("Fname"),
    rs.getString("MI"),
    rs.getString("Sex"),
    rs.getString("EmpType"),
    rs.getString("TeamName"),
    rs.getString("TeacherClass"),
    rs.getString("Password"),
    rs.getString("Action")
    };
    TModel.addRow(obj);
    }
}   catch (SQLException ex) {
```

JTable Output



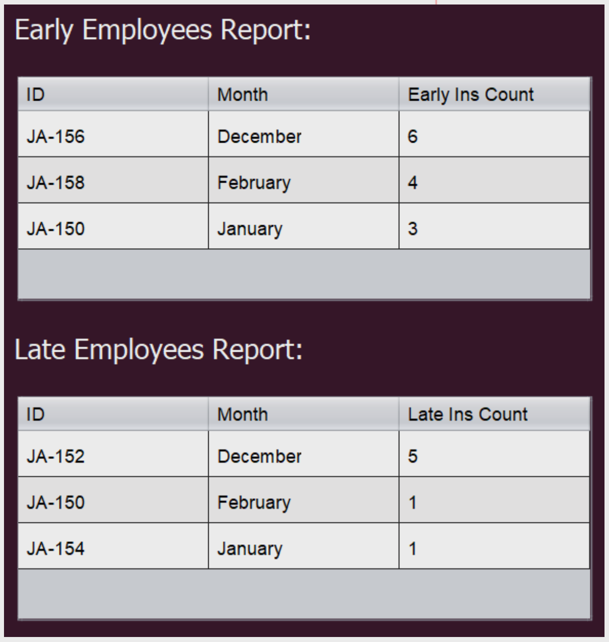
Record...	EmpID	Lname	Fname	MI	Sex	Team	Type	Class	Passw...	Action
1	JA-159	Cole	Maha	G.	M	Regular	Nico R...	Class ...	thisism...	Old Re...
2	JA-159	Cole	Maha	G.	M	---	---	---	thisism...	Update...
3	JA-159	Cole	Maha	G.	M	---	---	---	thisism...	Old Re...
4	JA-159	Rajan	Ganhee	A.	M	Part-ti...	Luffy	Class ...	ja159	Update...
5	JA-159	Rajan	Ganhee	A.	M	Part-ti...	Luffy	Class ...	ja159	Deleted
6	JA-150	Mondia	Robbie	A.	M	Part-ti...	Nico R...	Class ...	ja150	Old Re...
7	JA-150	Mondia	Robbie	A.	M	Staff	Nico R...	Class ...	ja150	Update...
8	JA-150	Mondia	Robbie	A.	M	Staff	Nico R...	Class ...	ja150	Old Re...
9	JA-150	Mondia	Robbie	A.	M	---	---	---	pass	Update...

Early and Late Ins Report – First, String query performs the SQL statements then the ResultSet will scan the joined tables. Finally, it will display the scanned data to the java table. The two tables have almost the same codes, the only difference is the "where" clause: `WHERE at.TimeIn < \"8:00\""`. This is not applicable to EPRO-Asia Inc. because there is no fixed Time In (< 8:00 am) there.



```
try{
Connection connect = DriverManager.getConnection(conn_string, username, password);
String query = "SELECT EmpId, Month, MAX(MaxIn) AS  \"Late Ins Count\"" +
        " FROM (SELECT et.EmpID, MONTHNAME(Date) AS Month, COUNT(at.TimeIn) AS MaxIn" +
        " FROM employee_tbl et JOIN attendancelog_tbl at" +
        " ON et.EmpID = at.EmpID" +
        " WHERE at.TimeIn > \"8:30\"" +
        " GROUP BY et.EmpID, MONTH(Date)" +
        " ORDER BY COUNT(at.TimeIn) DESC) AS sub" +
        " GROUP BY Month"; //For Search
PreparedStatement ps = connect.prepareStatement(query);

ResultSet rs = ps.executeQuery();
DefaultTableModel TModel = (DefaultTableModel) jTableLateOut.getModel();
TModel.setRowCount(0);

while(rs.next())
    {
    Object obj[] =
    {
    rs.getString("EmpID"),
    rs.getString("Month"),
    rs.getString("Late Ins Count")
    };
    TModel.addRow(obj);
    }
}   catch (Exception ex) {
    ex.printStackTrace();
```

JTable Output



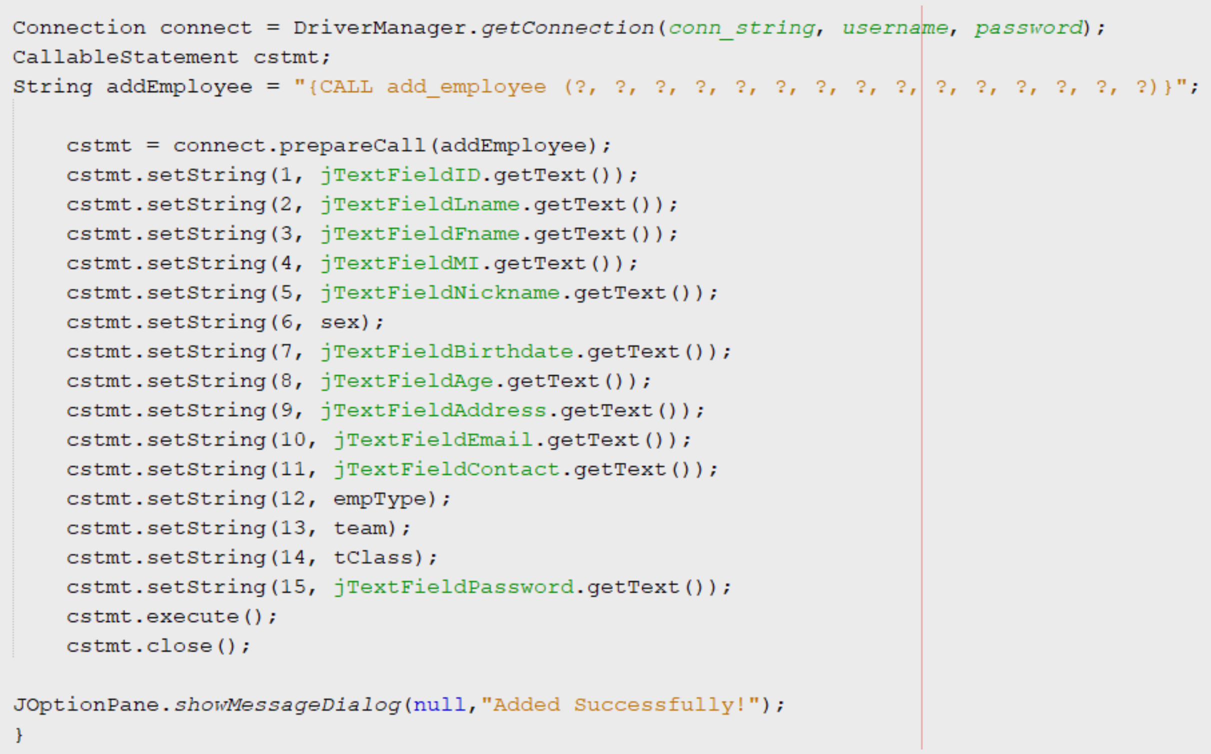
Early Employees Report:

ID	Month	Early Ins Count
JA-156	December	6
JA-158	February	4
JA-150	January	3

Late Employees Report:

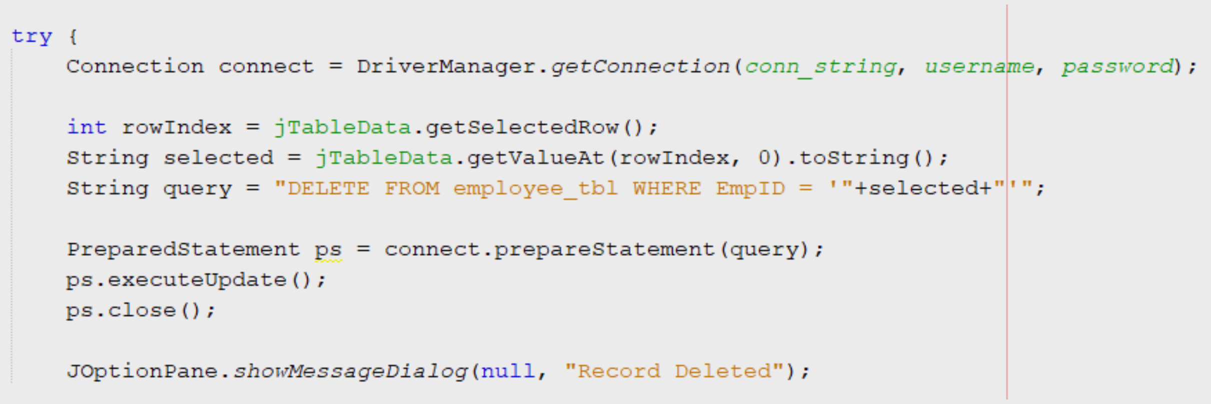
ID	Month	Late Ins Count
JA-152	December	5
JA-150	February	1
JA-154	January	1

Add Button (Insert) – Calls the stored procedure and adds the data to the employee_tbl according to the inputs in the text fields.



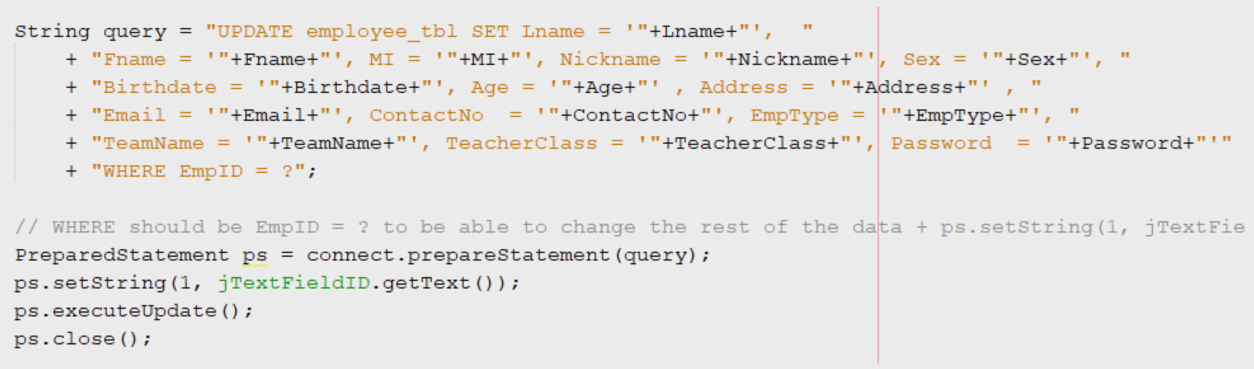
```
Connection connect = DriverManager.getConnection(conn_string, username, password);
CallableStatement cstmt;
String addEmployee = "{CALL add_employee (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)}";

    cstmt = connect.prepareCall(addEmployee);
    cstmt.setString(1, jTextFieldID.getText());
    cstmt.setString(2, jTextFieldLname.getText());
    cstmt.setString(3, jTextFieldFname.getText());
    cstmt.setString(4, jTextFieldMI.getText());
    cstmt.setString(5, jTextFieldNickname.getText());
    cstmt.setString(6, sex);
    cstmt.setString(7, jTextFieldBirthdate.getText());
    cstmt.setString(8, jTextFieldAge.getText());
    cstmt.setString(9, jTextFieldAddress.getText());
    cstmt.setString(10, jTextFieldEmail.getText());
    cstmt.setString(11, jTextFieldContact.getText());
    cstmt.setString(12, empType);
    cstmt.setString(13, team);
    cstmt.setString(14, tClass);
    cstmt.setString(15, jTextFieldPassword.getText());
    cstmt.execute();
    cstmt.close();

JOptionPane.showMessageDialog(null,"Added Successfully!");
}
```

Remove Button (Delete) – Deletes the selected row in the Java table.



```
try {
    Connection connect = DriverManager.getConnection(conn_string, username, password);

    int rowIndex = jTableData.getSelectedRow();
    String selected = jTableData.getValueAt(rowIndex, 0).toString();
    String query = "DELETE FROM employee_tbl WHERE EmpID = '"+selected+"'";

    PreparedStatement ps = connect.prepareStatement(query);
    ps.executeUpdate();
    ps.close();

    JOptionPane.showMessageDialog(null, "Record Deleted");
```

Update Button – Updates the employee_tbl once clicked. There is a limitation here however, all data can be updated except the EmpID because an "*SQLIntegrityConstraintViolationException: Cannot delete or update a parent row: a foreign key constraint fails*" will occur. The program updates based on the selected row on the java table and existing ID only. For instance, we have JA-150 and we will input a JA-200 on the text field, a prompt saying "Record Updated" will occur, but nothing will happen because JA-200 does not exist in the first place, and the program may not work anymore if we change the ID. Basically, we cannot change EmpID.



```
String query = "UPDATE employee_tbl SET Lname = '"+Lname+"',  "
    + "Fname = '"+Fname+"', MI = '"+MI+"', Nickname = '"+Nickname+"', Sex = '"+Sex+"', "
    + "Birthdate = '"+Birthdate+"', Age = '"+Age+"' , Address = '"+Address+"' , "
    + "Email = '"+Email+"', ContactNo  = '"+ContactNo+"', EmpType = '"+EmpType+"', "
    + "TeamName = '"+TeamName+"', TeacherClass = '"+TeacherClass+"', Password  = '"+Password+"'"
    + "WHERE EmpID = ?";

// WHERE should be EmpID = ? to be able to change the rest of the data + ps.setString(1, jTextFie
PreparedStatement ps = connect.prepareStatement(query);
ps.setString(1, jTextFieldID.getText());
ps.executeUpdate();
ps.close();
```

- **References:**

- Talathi, S. (2017, September 11). *HOW TO CONNECT MYSQL DATABASE IN JAVA USING NETBEANS 8.2 JAVA TUTORIALS FOR BEGINNERS* [Video]. YouTube. https://www.youtube.com/watch?v=qRrMc99OYLM
- Codec. (2018, January 21). *How to connect Java Project with MySQL Workbench* [Video]. YouTube. https://www.youtube.com/watch?v=ThTgN90PA44
- 1BestCsharp blog. (2019, February 12). *Java Project Tutorial With Source Code - Part 1/2* [Video]. YouTube. https://www.youtube.com/watch?v=gDD7_KUIm58&list=PLFDH5bKmoNqwlu2RaI1GrvtHbOqO-J5Kb
- 1BestCsharp blog. (2019, February 12). *Java Project Tutorial With Source Code - Part 2/2* [Video]. YouTube. https://www.youtube.com/watch?v=LU51dpZ47Cc&list=PLFDH5bKmoNqwlu2RaI1GrvtHbOqO-J5Kb&index=2
- Naravani, M. (2017, September 23). 11. *Inserting data into MySQL database using jFrame form in Netbeans Java - Part - 2* [Video]. YouTube. https://www.youtube.com/watch?v=17DQ5uYJkw4
- 1BestCsharp blog. (2017, August 14). JAVA - *How To Design Login And Register Form In Java Netbeans* [Video]. YouTube. https://www.youtube.com/watch?v=XAowXcmQ-kA
- DeX PY. (2021, January 23). *Membuat Login dan Sign Up di Netbeans dengan MySQL Database* [Video]. YouTube. https://www.youtube.com/watch?v=FaCMHJZu4KU