4/23/2023

ARCHITECTURE

EMPLOYEE TRACKING SYSTEM

PARAMITA
ineuron

ABSTRACT

The Employee tracking application is prepared as a console-based core java project. The system tracks the performance of all registered employees in an organisation, involved in different projects. The detailed address of the employees is stored in a different entity and may be referred to when tracking of employees is necessary or their proximity to each other and involvement in different projects may need to be conveniently assigned.

The Manager entity of the system benefits the managerial staff to track the employees efficiently to know their allotment, department, progress, and scheduling data.

This system requires no web server and may be executed from a digital device within the onsite premises of an organisation which can effectively use it to understand the performance of its employees.

The system helps in

- Division of labour
- Understanding effectiveness of each employee
- Understanding employee potential
- Project constraints
- Effective scheduling
- Effective running of organisational framework
- Less cost involvement
- Tracking employee details

Architecture document generally indicates about the data arrangement of a system.

The system is primarily based on core java. the hardware software and other technical requirements of the system. Any platform and machine with an installed jdk can effectively run the application. As the project is developed in java it is:

- Portable
- Simple to understand and implement
- Any database can be used for stating minimum employee details
- Minimum storage and any RAM that effectively runs a java application can be used.
- The application uses the System Library which may vary from machine to machine.
- The application is developed in Eclipse IDE. It may be opened in any IDE or it may even run in command prompt.

The system to be developed is based on core java as a console-based project. The system is developed to track which employees are involved in which project and their details.

- Employees and their details are inserted in the system.
- Employee details, as Address may be updated in the system.
- Employees may be removed from a project.
- A Manager view is created in the manager class.
- No database and no connector are involved. Employees are stored in employee array.
- Linked hash map is used for a quick view of empid and employee_ name.
- The code is portable and easily maintained. It may well be enhanced and modified.

The architecture of a system generally shows the backend development of a system with its database and connectivity features.

Since this system is not connected to the database, it is not possible to show the system archcitecture with context to table creation.

The system might however be connected in future to implement database characteristics.

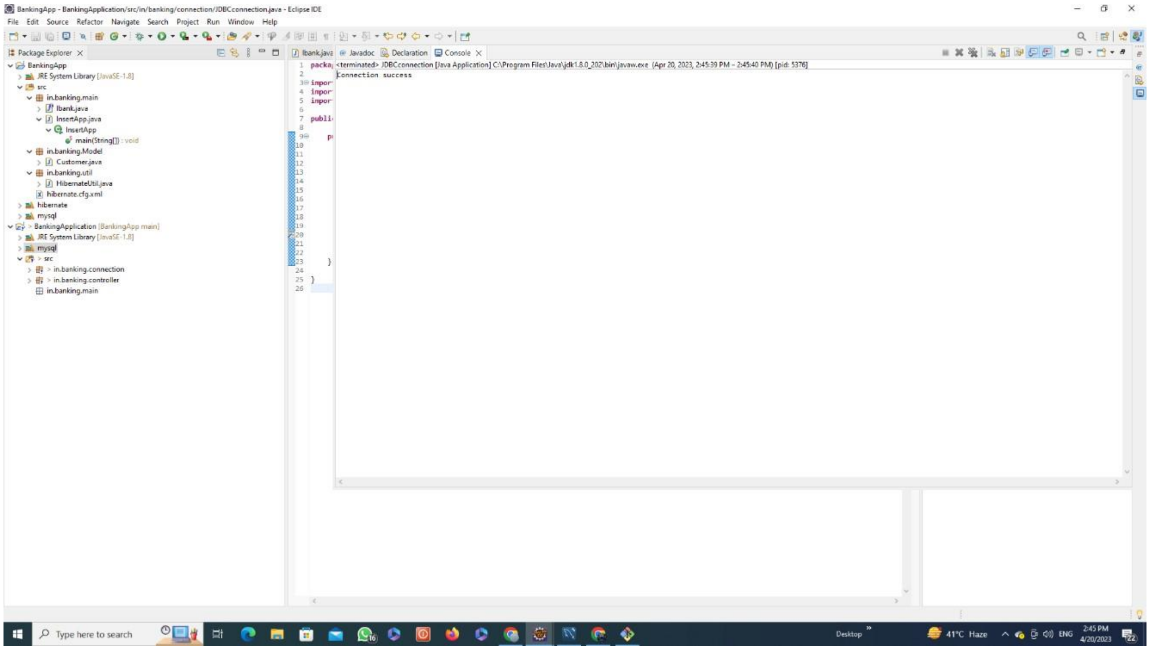
The mode of acceptance of data as input denotes the system architecture here.

The database of the system is prepared with MySQL workbench and kept ready.

Features of connectivity for smooth functioning of java code with the the system:

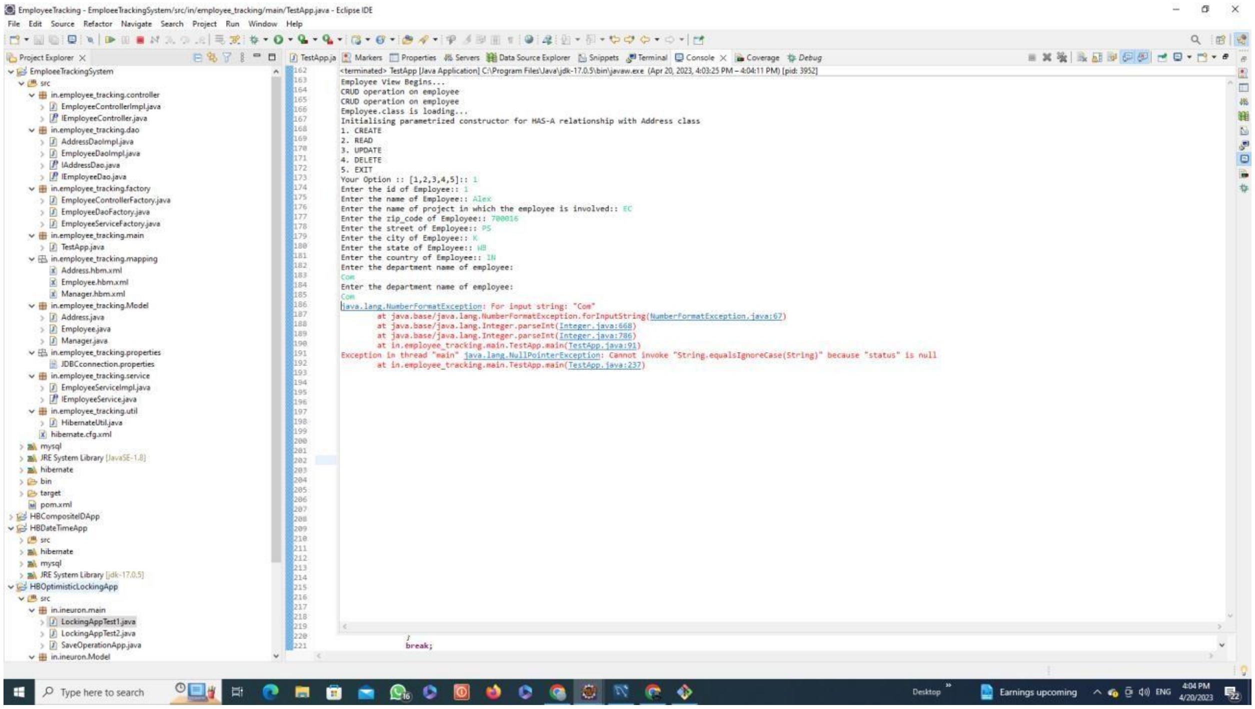
Entity name/ table name == class name

The connectivity of the system was checked with database for future implementation



BankingApp - BankingApplication/src/in/banking/connection/JDBCconnection.java - Eclipse IDE

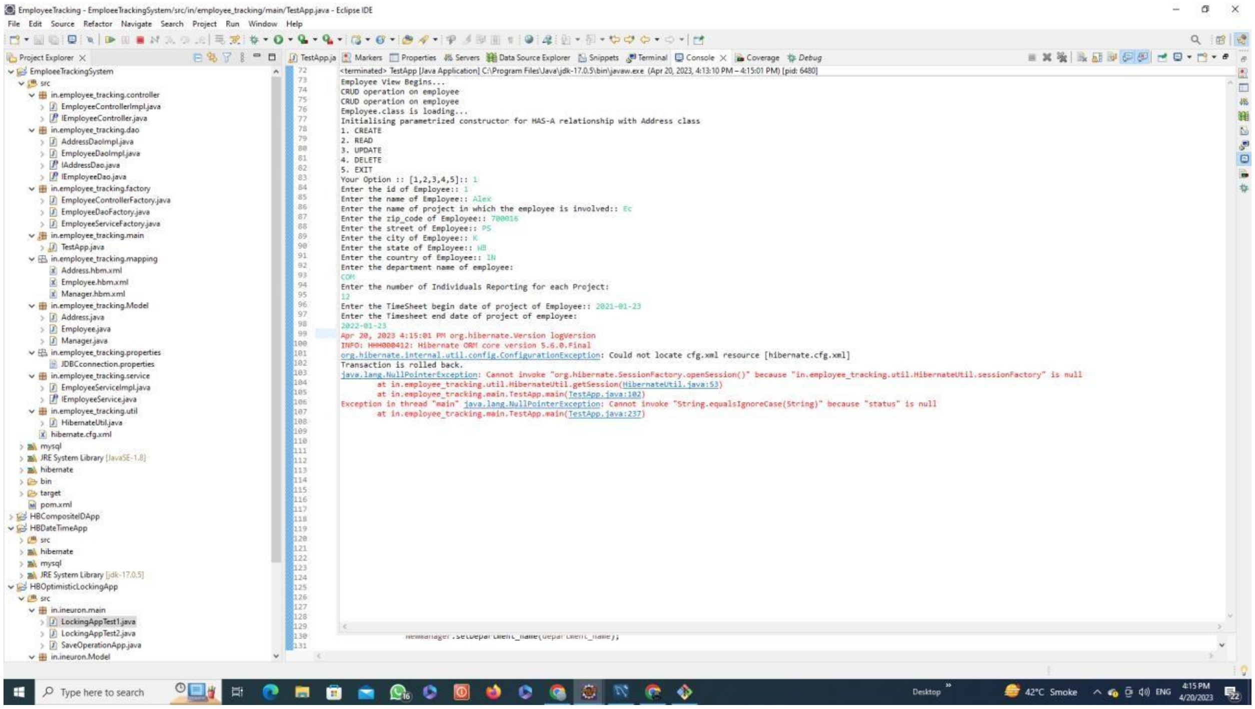
```
<terminated> JDBCconnection [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.exe  (Apr 20, 2023, 2:45:39 PM – 2:45:40 PM) [pid: 5376]
Connection success
```



EmployeeTracking - EmploeeTrackingSystem/src/in/employee_tracking/main/TestApp.java - Eclipse IDE

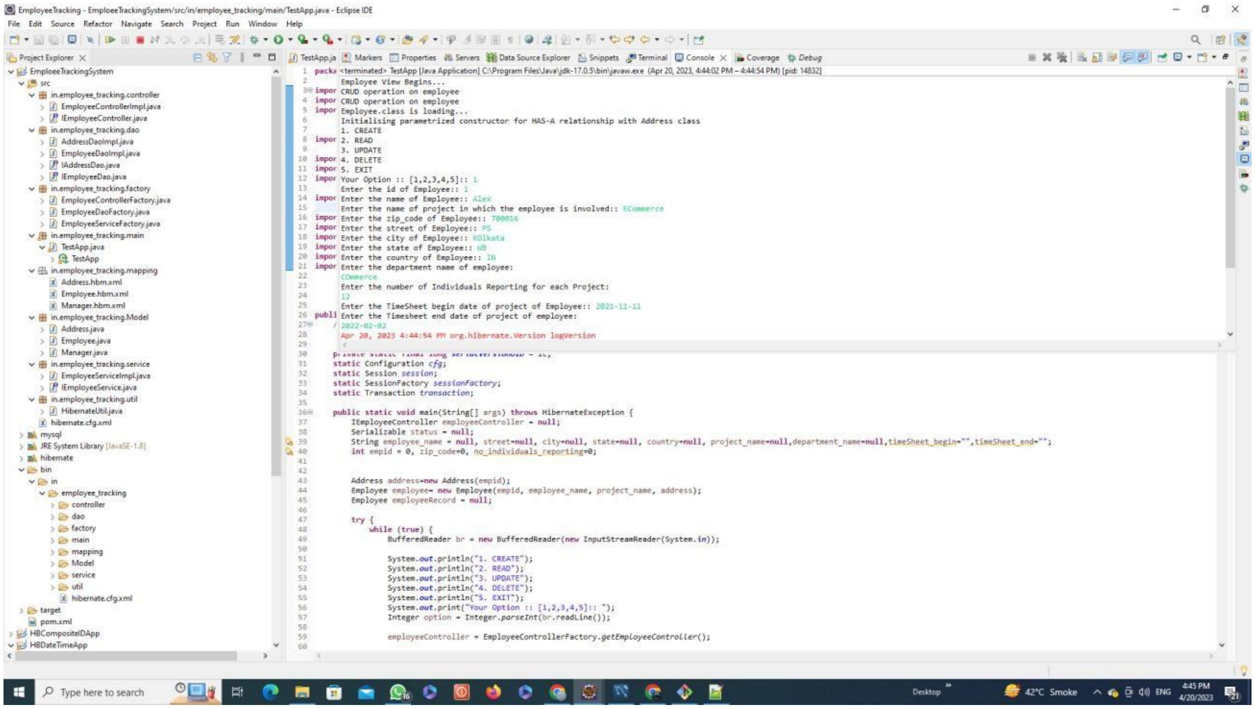
```
<terminated> TestApp [Java Application] C:\Program Files\Java\jdk-17.0.5\bin\javaw.exe (Apr 20, 2023, 4:03:25 PM – 4:04:11 PM) [pid: 3952]
Employee View Begins...
CRUD operation on employee
CRUD operation on employee
Employee.class is loading...
Initialising parametrized constructor for HAS-A relationship with Address class
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Your Option :: [1,2,3,4,5]:: 1
Enter the id of Employee:: 1
Enter the name of Employee:: Alex
Enter the name of project in which the employee is involved:: EC
Enter the zip_code of Employee:: 700016
Enter the street of Employee:: PS
Enter the city of Employee:: K
Enter the state of Employee:: WB
Enter the country of Employee:: IN
Enter the department name of employee:
Com
Enter the department name of employee:
Com
java.lang.NumberFormatException: For input string: "Com"
	at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
	at java.base/java.lang.Integer.parseInt(Integer.java:668)
	at java.base/java.lang.Integer.parseInt(Integer.java:786)
	at in.employee_tracking.main.TestApp.main(TestApp.java:91)
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.equalsIgnoreCase(String)" because "status" is null
	at in.employee_tracking.main.TestApp.main(TestApp.java:237)
```

The system faced problems with hibernate integration on the following so, the process was terminated:



EmployeeTracking - EmploeeTrackingSystem/src/in/employee_tracking/main/TestApp.java - Eclipse IDE

```
<terminated> TestApp [Java Application] C:\Program Files\Java\jdk-17.0.5\bin\javaw.exe (Apr 20, 2023, 4:13:10 PM – 4:15:01 PM) [pid: 6480]
Employee View Begins...
CRUD operation on employee
CRUD operation on employee
Employee.class is loading...
Initialising parametrized constructor for HAS-A relationship with Address class
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Your Option :: [1,2,3,4,5]:: 1
Enter the id of Employee:: 1
Enter the name of Employee:: Alex
Enter the name of project in which the employee is involved:: Ec
Enter the zip_code of Employee:: 700016
Enter the street of Employee:: PS
Enter the city of Employee:: K
Enter the state of Employee:: WB
Enter the country of Employee:: IN
Enter the department name of employee:
COM
Enter the number of Individuals Reporting for each Project:
12
Enter the TimeSheet begin date of project of Employee:: 2021-01-23
Enter the Timesheet end date of project of employee:
2022-01-23
Apr 20, 2023 4:15:01 PM org.hibernate.Version logVersion
INFO: HHH000412: Hibernate ORM core version 5.6.0.Final
org.hibernate.internal.util.config.ConfigurationException: Could not locate cfg.xml resource [hibernate.cfg.xml]
Transaction is rolled back.
java.lang.NullPointerException: Cannot invoke "org.hibernate.SessionFactory.openSession()" because "in.employee_tracking.util.HibernateUtil.sessionFactory" is null
	at in.employee_tracking.util.HibernateUtil.getSession(HibernateUtil.java:53)
	at in.employee_tracking.main.TestApp.main(TestApp.java:102)
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.equalsIgnoreCase(String)" because "status" is null
	at in.employee_tracking.main.TestApp.main(TestApp.java:237)
```



```
<terminated> TestApp [Java Application] C:\Program Files\Java\jdk-17.0.5\bin\javaw.exe (Apr 20, 2023, 4:44:02 PM – 4:44:54 PM) [pid: 14832]
Employee View Begins...
CRUD operation on employee
CRUD operation on employee
Employee.class is loading...
Initialising parametrized constructor for HAS-A relationship with Address class
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Your Option :: [1,2,3,4,5]:: 1
Enter the id of Employee:: 1
Enter the name of Employee:: Alex
Enter the name of project in which the employee is involved:: ECommerce
Enter the zip_code of Employee:: 700016
Enter the street of Employee:: PS
Enter the city of Employee:: KOlkata
Enter the state of Employee:: WB
Enter the country of Employee:: IN
Enter the department name of employee:
COmmerce
Enter the number of Individuals Reporting for each Project:
12
Enter the TimeSheet begin date of project of Employee:: 2021-11-11
Enter the Timesheet end date of project of employee:
2022-02-02
Apr 20, 2023 4:44:54 PM org.hibernate.Version logVersion
```

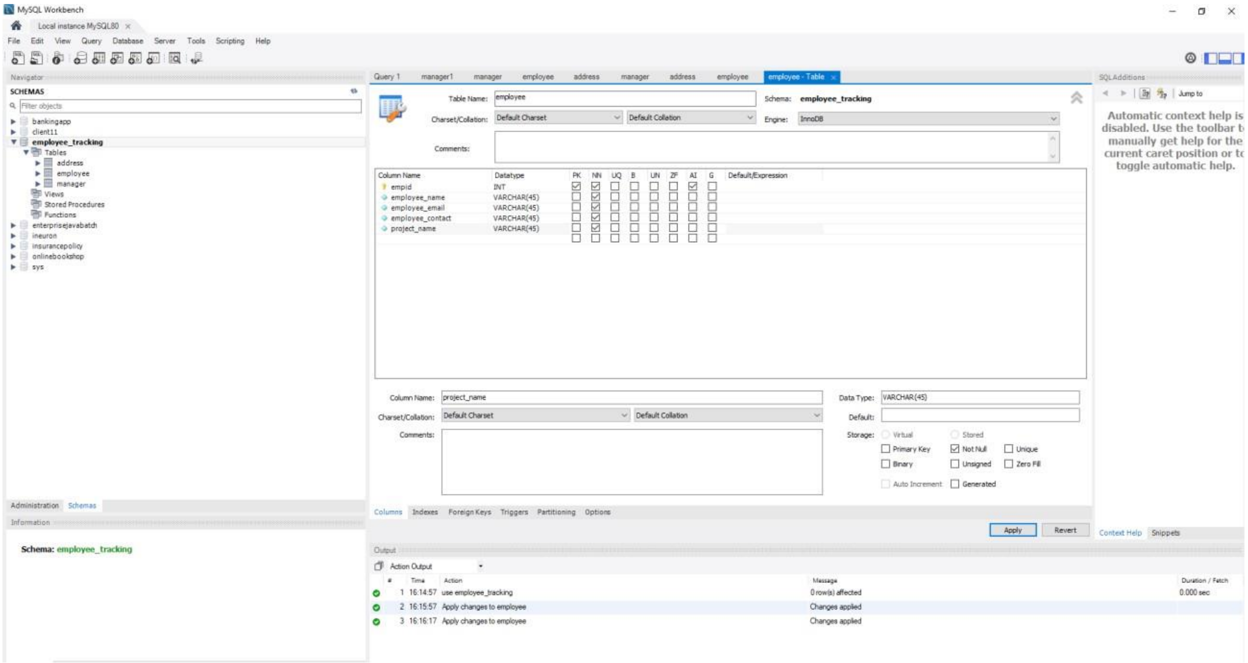
```java
    private static final long serialVersionUID = 1L;
    static Configuration cfg;
    static Session session;
    static SessionFactory sessionFactory;
    static Transaction transaction;

    public static void main(String[] args) throws HibernateException {
        IEmployeeController employeeController = null;
        Serializable status = null;
        String employee_name = null, street=null, city=null, state=null, country=null, project_name=null,department_name=null,timeSheet_begin="",timeSheet_end="";
        int empid = 0, zip_code=0, no_individuals_reporting=0;


        Address address=new Address(empid);
        Employee employee= new Employee(empid, employee_name, project_name, address);
        Employee employeeRecord = null;

        try {
            while (true) {
                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                System.out.println("1. CREATE");
                System.out.println("2. READ");
                System.out.println("3. UPDATE");
                System.out.println("4. DELETE");
                System.out.println("5. EXIT");
                System.out.print("Your Option :: [1,2,3,4,5]:: ");
                Integer option = Integer.parseInt(br.readLine());

                employeeController = EmployeeControllerFactory.getEmployeeController();

```

The present system is purely in core java and the database connectivity is not executed for the time being.

The tables created in the Schema employee_ tracking are:

Table name: employee

Primary key : empid


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
bankingapp
client11
employee_tracking
Tables
address
employee
manager
Views
Stored Procedures
Functions
enterprisejavabatch
ineuron
insurancepolicy
onlinebookshop
sys
Administration Schemas
Information
Query 1 manager1 manager employee address manager address employee employee - Table
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.
Columns Indexes Foreign Keys Triggers Partitioning Options
Apply Revert
Context Help Snippets
Output
Action Output

Schema: employee_tracking

Table Name: employee

Schema: employee_tracking

Charset/Collation: Default Charset / Default Collation

Engine: InnoDB

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
empid	INT	☑	☑	□	□	□	□	☑	□	
employee_name	VARCHAR(45)	□	☑	□	□	□	□	□	□	
employee_email	VARCHAR(45)	□	☑	□	□	□	□	□	□	
employee_contact	VARCHAR(45)	□	☑	□	□	□	□	□	□	
project_name	VARCHAR(45)	□	☑	□	□	□	□	□	□	
		□	□	□	□	□	□	□	□	

Column Name: project_name

Data Type: VARCHAR(45)

Charset/Collation: Default Charset / Default Collation

Default:

Comments:

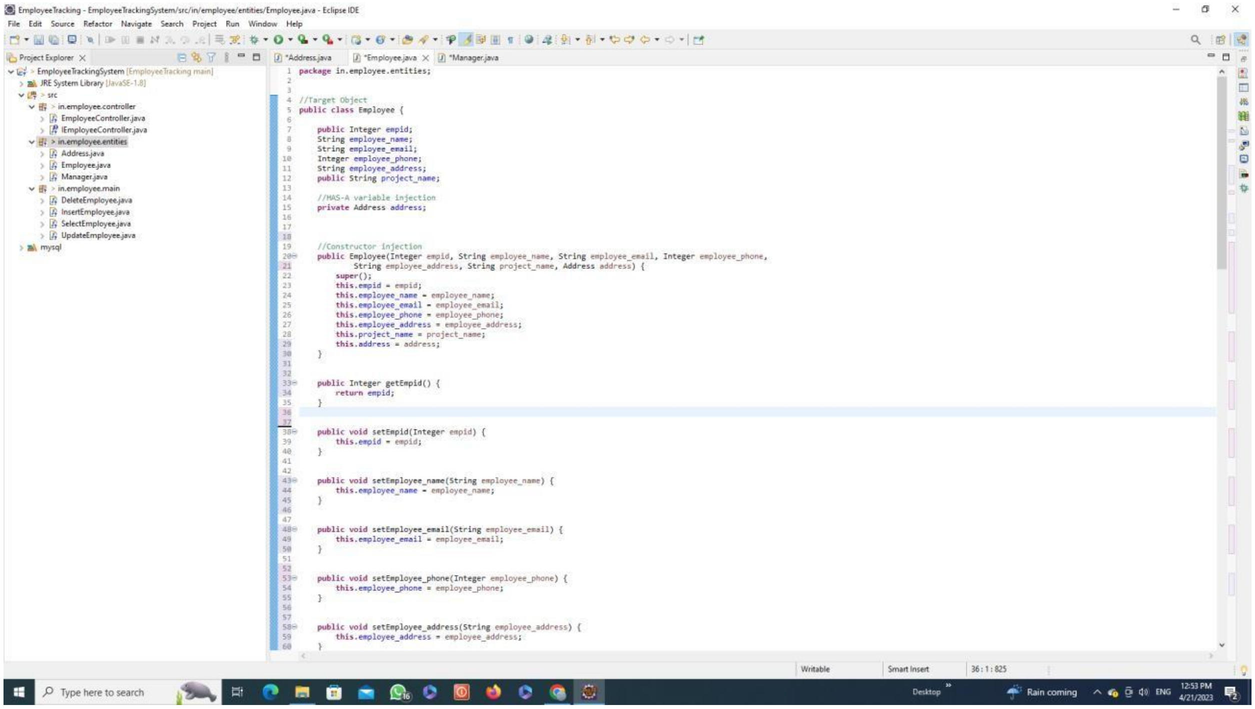
Storage: ○ Virtual ○ Stored

□ Primary Key ☑ Not Null □ Unique

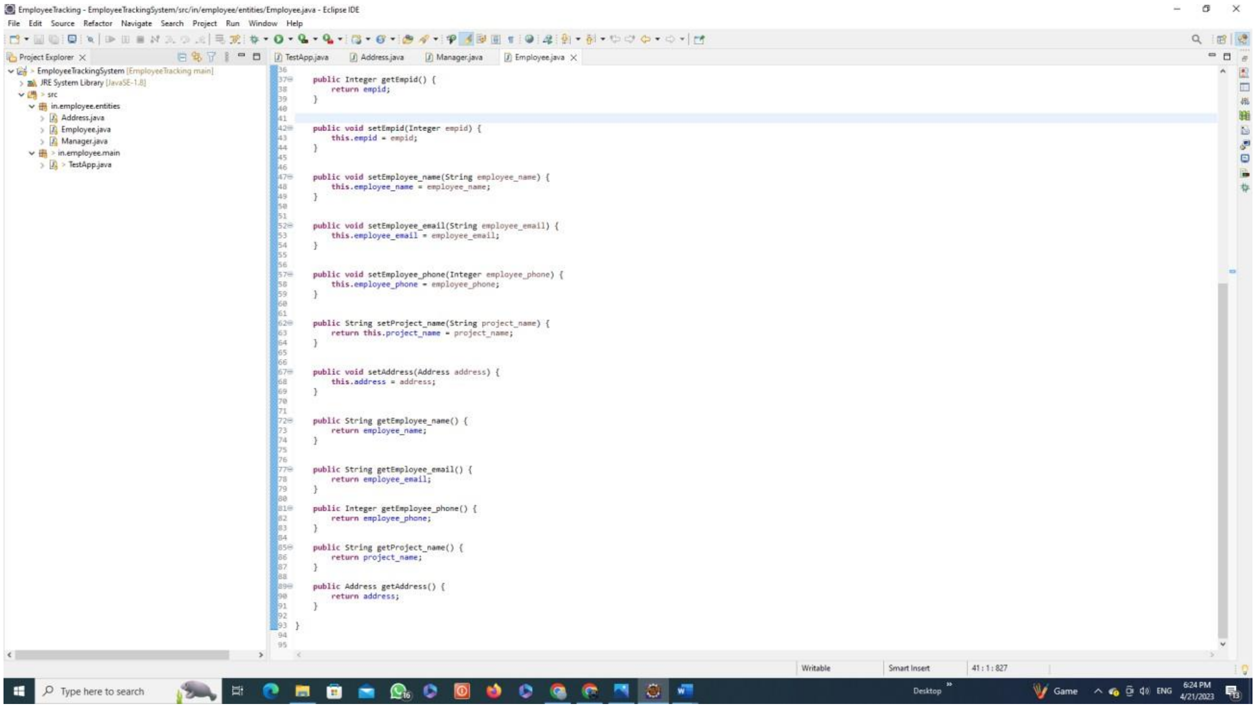
□ Binary □ Unsigned □ Zero Fill

□ Auto Increment □ Generated

```
#  Time      Action                         Message              Duration / Fetch
1  16:14:57  use employee_tracking          0 row(s) affected    0.000 sec
2  16:15:57  Apply changes to employee      Changes applied
3  16:16:17  Apply changes to employee      Changes applied
```



```java
package in.employee.entities;


//Target Object
public class Employee {

    public Integer empid;
    String employee_name;
    String employee_email;
    Integer employee_phone;
    String employee_address;
    public String project_name;

    //HAS-A variable injection
    private Address address;



    //Constructor injection
    public Employee(Integer empid, String employee_name, String employee_email, Integer employee_phone,
            String employee_address, String project_name, Address address) {
        super();
        this.empid = empid;
        this.employee_name = employee_name;
        this.employee_email = employee_email;
        this.employee_phone = employee_phone;
        this.employee_address = employee_address;
        this.project_name = project_name;
        this.address = address;
    }


    public Integer getEmpid() {
        return empid;
    }


    public void setEmpid(Integer empid) {
        this.empid = empid;
    }


    public void setEmployee_name(String employee_name) {
        this.employee_name = employee_name;
    }


    public void setEmployee_email(String employee_email) {
        this.employee_email = employee_email;
    }


    public void setEmployee_phone(Integer employee_phone) {
        this.employee_phone = employee_phone;
    }


    public void setEmployee_address(String employee_address) {
        this.employee_address = employee_address;
    }
```



EmployeeTracking - EmployeeTrackingSystem/src/in/employee/entities/Employee.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
EmployeeTrackingSystem [EmployeeTracking main]
JRE System Library [JavaSE-1.8]
src
in.employee.entities
Address.java
Employee.java
Manager.java
in.employee.main
TestApp.java
TestApp.java
Address.java
Manager.java
Employee.java
Writable
Smart Insert
41 : 1 : 827
Type here to search
Desktop
Game
ENG
6:24 PM
4/21/2023

```java

    public Integer getEmpid() {
        return empid;
    }


    public void setEmpid(Integer empid) {
        this.empid = empid;
    }


    public void setEmployee_name(String employee_name) {
        this.employee_name = employee_name;
    }


    public void setEmployee_email(String employee_email) {
        this.employee_email = employee_email;
    }


    public void setEmployee_phone(Integer employee_phone) {
        this.employee_phone = employee_phone;
    }


    public String setProject_name(String project_name) {
        return this.project_name = project_name;
    }


    public void setAddress(Address address) {
        this.address = address;
    }


    public String getEmployee_name() {
        return employee_name;
    }


    public String getEmployee_email() {
        return employee_email;
    }

    public Integer getEmployee_phone() {
        return employee_phone;
    }

    public String getProject_name() {
        return project_name;
    }

    public Address getAddress() {
        return address;
    }

}


```

Table name: address

Primary key: zip_ code

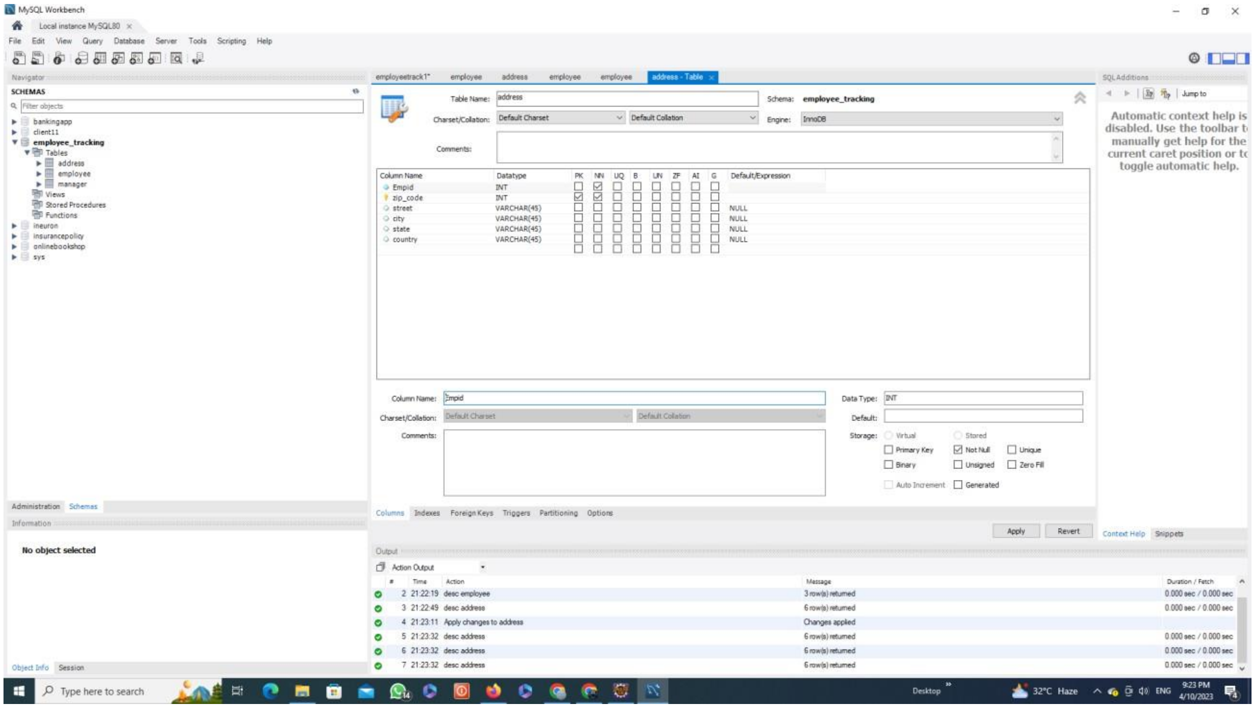


Table Name: address

Schema: **employee_tracking**

Charset/Collation: Default Charset | Default Collation

Engine: InnoDB

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Empid	INT	□	☑	□	□	□	□	□	□	
zip_code	INT	☑	☑	□	□	□	□	□	□	
street	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
city	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
state	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
country	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
		□	□	□	□	□	□	□	□	

Column Name: Empid

Data Type: INT

Charset/Collation: Default Charset | Default Collation

Default:

Comments:

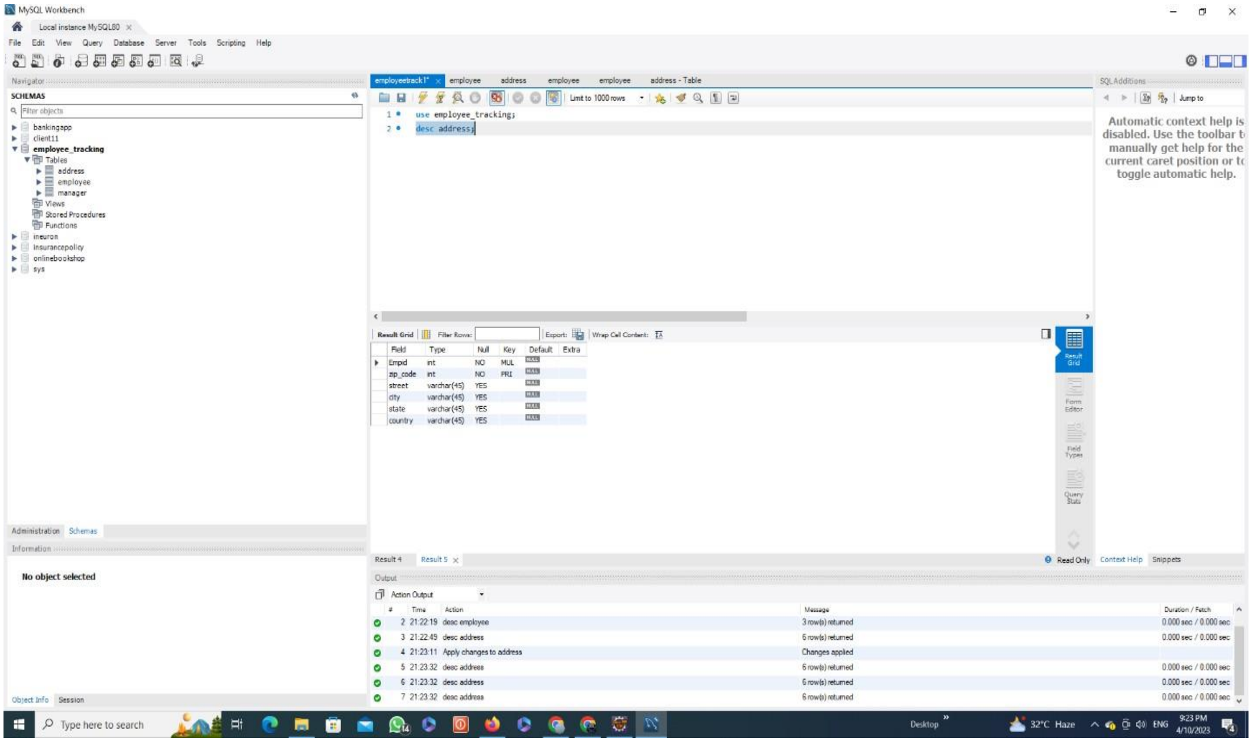
Storage: ○ Virtual ○ Stored

□ Primary Key ☑ Not Null □ Unique

□ Binary □ Unsigned □ Zero Fill

□ Auto Increment □ Generated

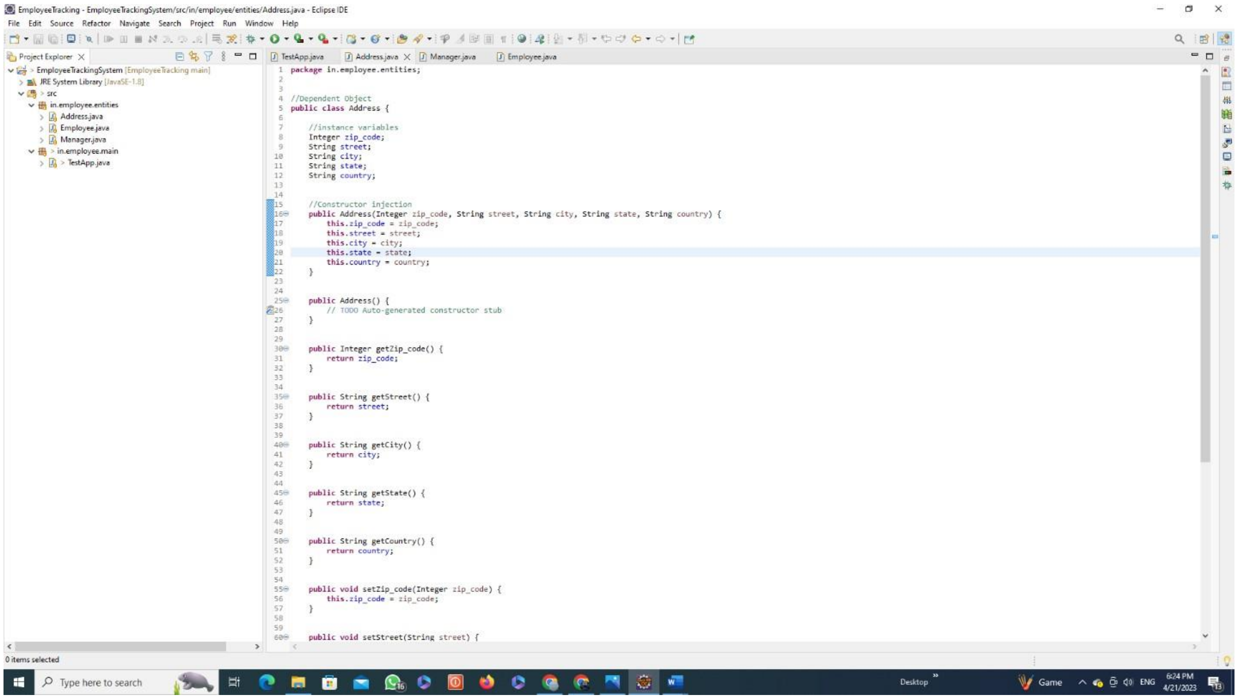
```
#  Time      Action                     Message           Duration / Fetch
2  21:22:19  desc employee              3 row(s) returned  0.000 sec / 0.000 sec
3  21:22:49  desc address               6 row(s) returned  0.000 sec / 0.000 sec
4  21:23:11  Apply changes to address   Changes applied
5  21:23:32  desc address               6 row(s) returned  0.000 sec / 0.000 sec
6  21:23:32  desc address               6 row(s) returned  0.000 sec / 0.000 sec
7  21:23:32  desc address               6 row(s) returned  0.000 sec / 0.000 sec
```


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
employeetrack1* employee address employee employee address - Table
Limit to 1000 rows
SCHEMAS
bankingapp
client11
employee_tracking
Tables
address
employee
manager
Views
Stored Procedures
Functions
ineuron
insurancepolicy
onlinebookshop
sys
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
No object selected
Result 4 Result 5
Read Only Context Help Snippets
Action Output

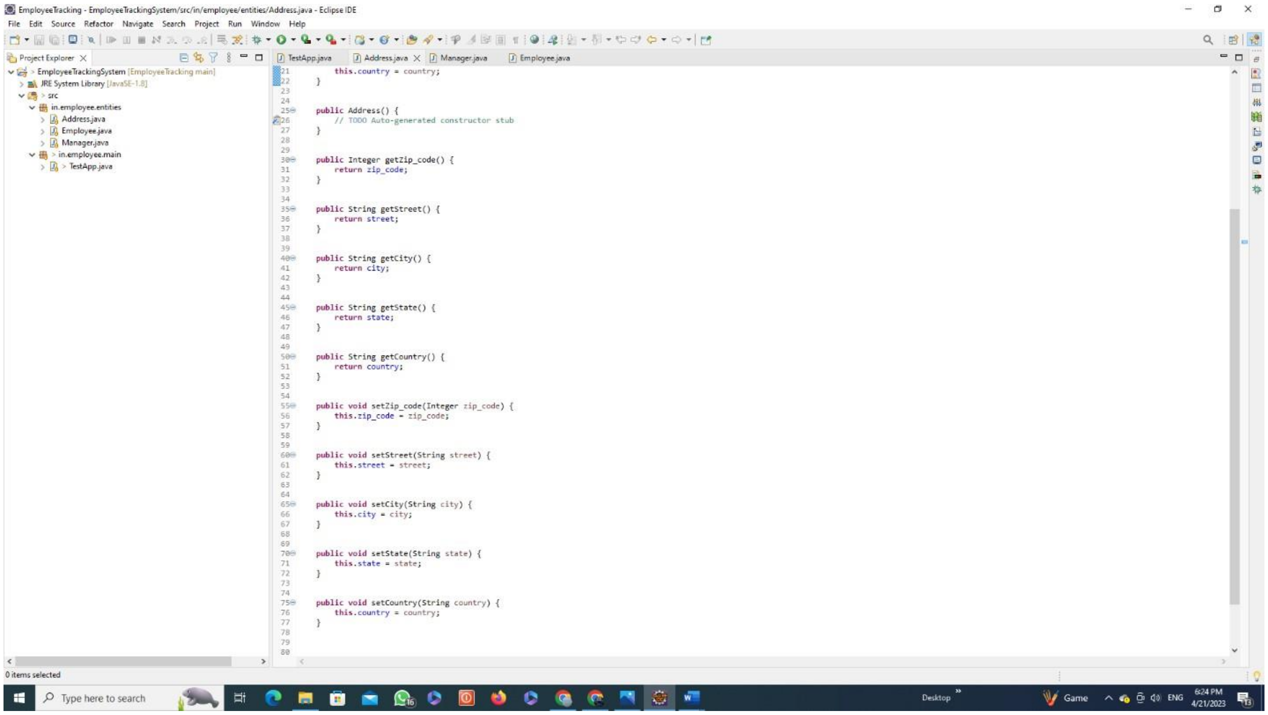
```sql
use employee_tracking;
desc address;
```

Field	Type	Null	Key	Default	Extra
Empid	int	NO	MUL	NULL	
zip_code	int	NO	PRI	NULL	
street	varchar(45)	YES		NULL	
city	varchar(45)	YES		NULL	
state	varchar(45)	YES		NULL	
country	varchar(45)	YES		NULL	

```
#  Time      Action                      Message           Duration / Fetch
2  21:22:19  desc employee               3 row(s) returned 0.000 sec / 0.000 sec
3  21:22:49  desc address                6 row(s) returned 0.000 sec / 0.000 sec
4  21:23:11  Apply changes to address    Changes applied
5  21:23:32  desc address                6 row(s) returned 0.000 sec / 0.000 sec
6  21:23:32  desc address                6 row(s) returned 0.000 sec / 0.000 sec
7  21:23:32  desc address                6 row(s) returned 0.000 sec / 0.000 sec
```

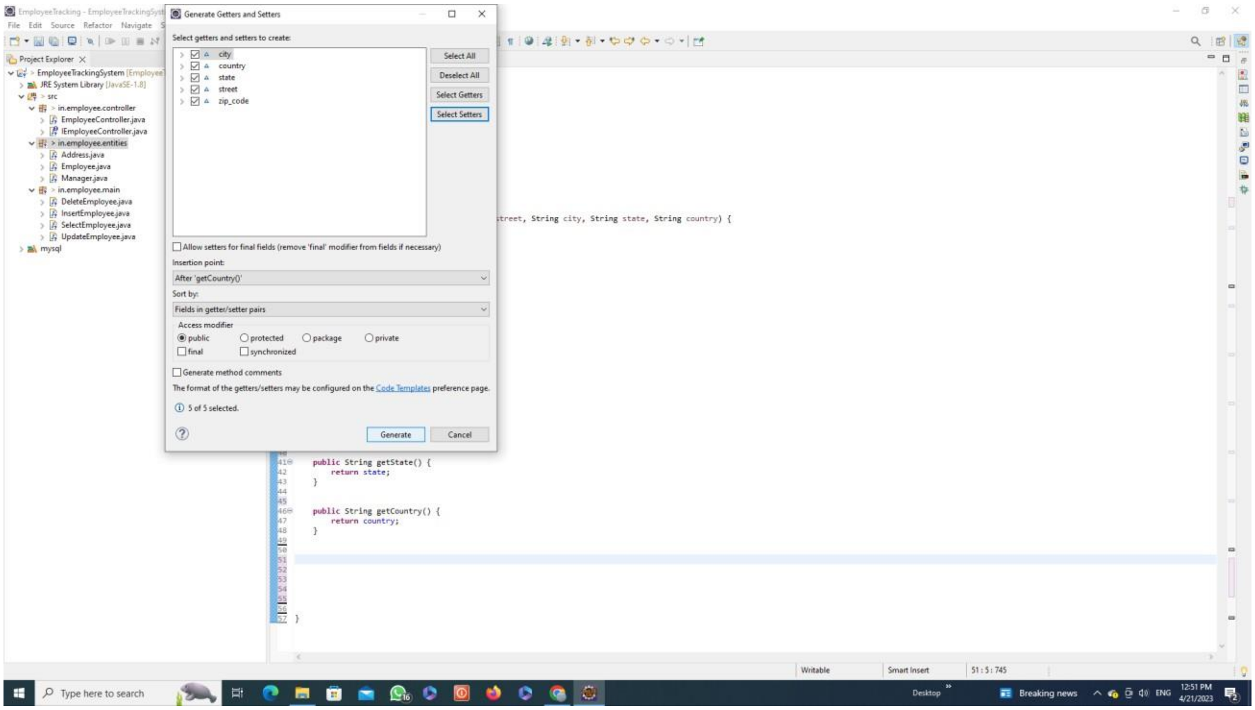

EmployeeTracking - EmployeeTrackingSystem/src/in/employee/entities/Address.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
EmployeeTrackingSystem [EmployeeTracking main]
JRE System Library [JavaSE-1.8]
src
in.employee.entities
Address.java
Employee.java
Manager.java
in.employee.main
TestApp.java
TestApp.java Address.java Manager.java Employee.java
0 items selected

```java
package in.employee.entities;


//Dependent Object
public class Address {

    //instance variables
    Integer zip_code;
    String street;
    String city;
    String state;
    String country;


    //Constructor injection
    public Address(Integer zip_code, String street, String city, String state, String country) {
        this.zip_code = zip_code;
        this.street = street;
        this.city = city;
        this.state = state;
        this.country = country;
    }


    public Address() {
        // TODO Auto-generated constructor stub
    }


    public Integer getZip_code() {
        return zip_code;
    }


    public String getStreet() {
        return street;
    }


    public String getCity() {
        return city;
    }


    public String getState() {
        return state;
    }


    public String getCountry() {
        return country;
    }


    public void setZip_code(Integer zip_code) {
        this.zip_code = zip_code;
    }


    public void setStreet(String street) {
```



EmployeeTracking - EmployeeTrackingSystem/src/in/employee/entities/Address.java - Eclipse IDE

```
        this.country = country;
    }


    public Address() {
        // TODO Auto-generated constructor stub
    }


    public Integer getZip_code() {
        return zip_code;
    }


    public String getStreet() {
        return street;
    }


    public String getCity() {
        return city;
    }


    public String getState() {
        return state;
    }


    public String getCountry() {
        return country;
    }


    public void setZip_code(Integer zip_code) {
        this.zip_code = zip_code;
    }


    public void setStreet(String street) {
        this.street = street;
    }


    public void setCity(String city) {
        this.city = city;
    }


    public void setState(String state) {
        this.state = state;
    }


    public void setCountry(String country) {
        this.country = country;
    }
```



Generate Getters and Setters

Select getters and setters to create:

- ☑ city
- ☑ country
- ☑ state
- ☑ street
- ☑ zip_code

□ Allow setters for final fields (remove 'final' modifier from fields if necessary)

Insertion point:
After 'getCountry()'

Sort by:
Fields in getter/setter pairs

Access modifier
● public ○ protected ○ package ○ private
□ final □ synchronized

□ Generate method comments

The format of the getters/setters may be configured on the Code Templates preference page.

ⓘ 5 of 5 selected.

```
treet, String city, String state, String country) {

    public String getState() {
        return state;
    }


    public String getCountry() {
        return country;
    }







}
```

Table name: manager

No primary key of its own [weak entity]

In project set up manger class extends the Employee class

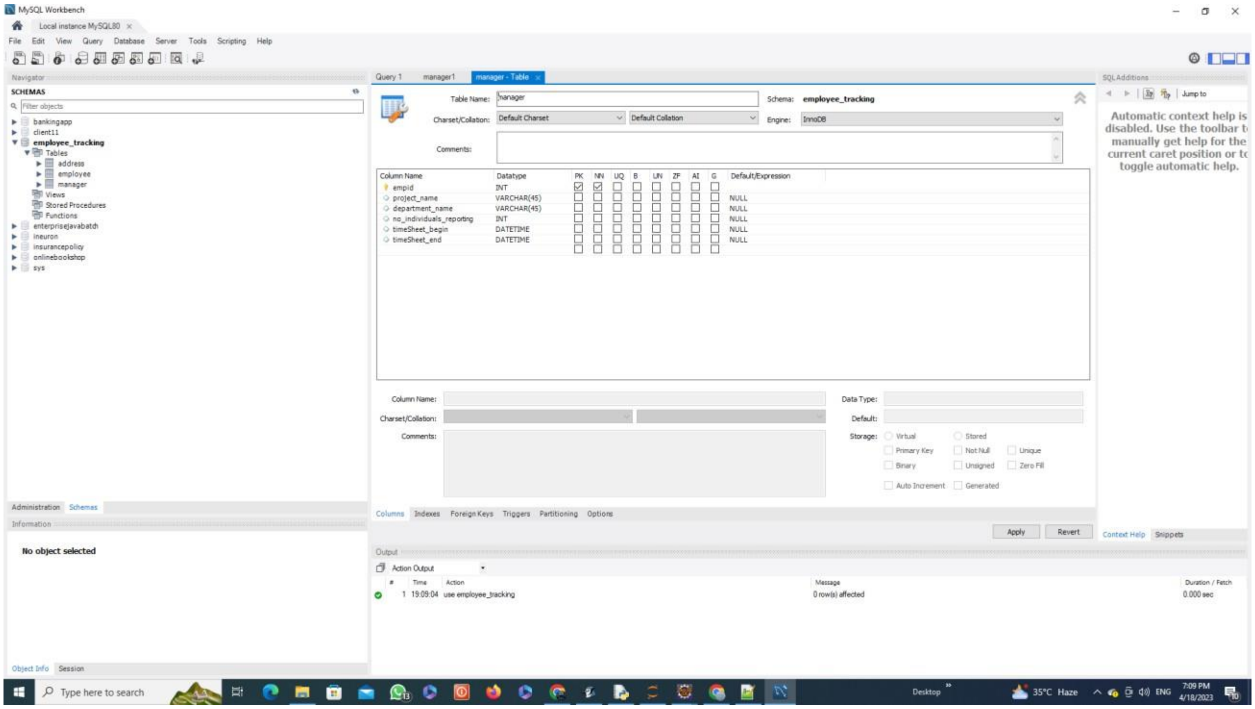


Table Name: manager

Schema: **employee_tracking**

Charset/Collation: Default Charset / Default Collation

Engine: InnoDB

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
empid	INT	☑	☑	□	□	□	□	□	□	
project_name	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
department_name	VARCHAR(45)	□	□	□	□	□	□	□	□	NULL
no_individuals_reporting	INT	□	□	□	□	□	□	□	□	NULL
timeSheet_begin	DATETIME	□	□	□	□	□	□	□	□	NULL
timeSheet_end	DATETIME	□	□	□	□	□	□	□	□	NULL
		□	□	□	□	□	□	□	□	

Column Name:

Data Type:

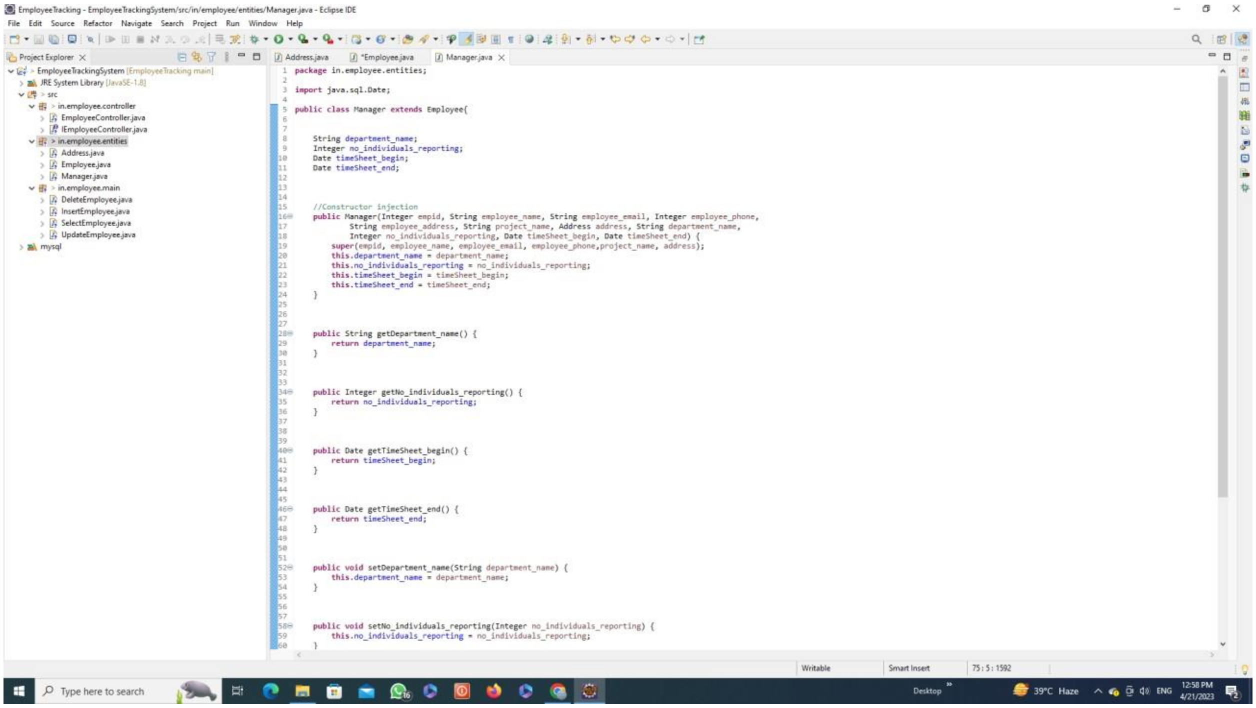
Charset/Collation:

Default:

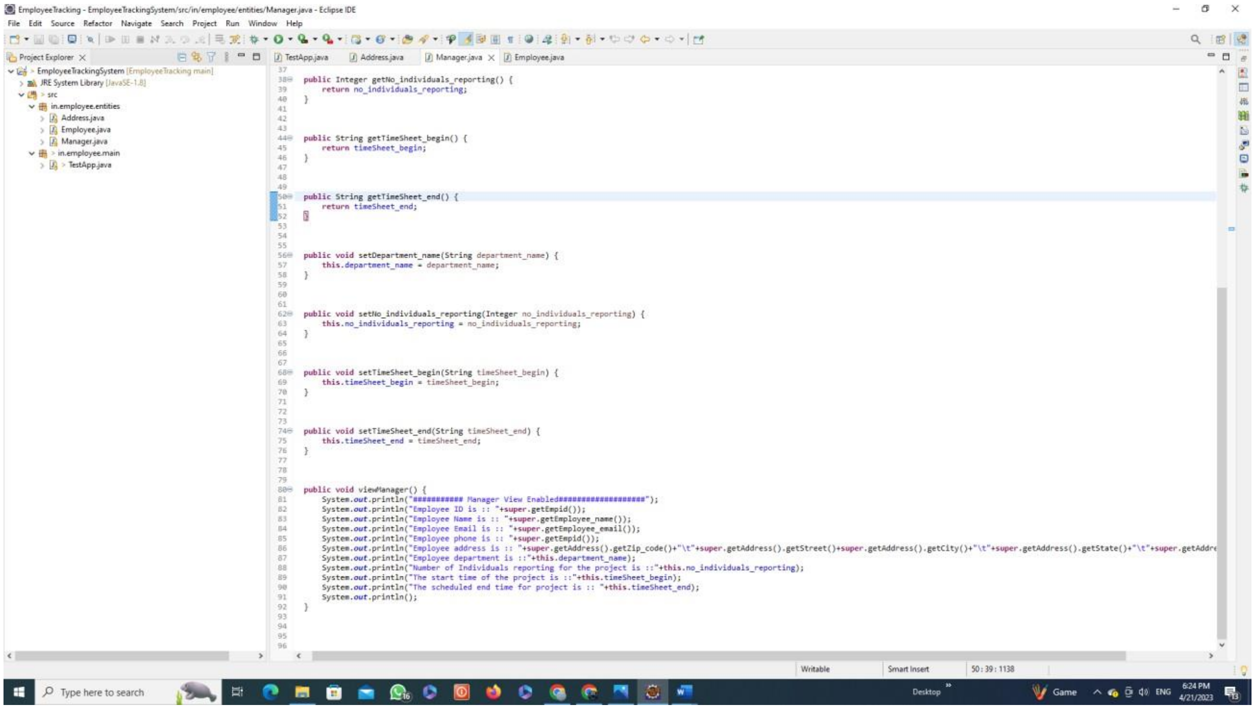
Comments:

Storage: ○ Virtual ○ Stored
□ Primary Key □ Not Null □ Unique
□ Binary □ Unsigned □ Zero Fill
□ Auto Increment □ Generated

#	Time	Action	Message	Duration / Fetch
1	19:09:04	use employee_tracking	0 row(s) affected	0.000 sec



```java
package in.employee.entities;

import java.sql.Date;

public class Manager extends Employee{


    String department_name;
    Integer no_individuals_reporting;
    Date timeSheet_begin;
    Date timeSheet_end;



    //Constructor injection
    public Manager(Integer empid, String employee_name, String employee_email, Integer employee_phone,
            String employee_address, String project_name, Address address, String department_name,
            Integer no_individuals_reporting, Date timeSheet_begin, Date timeSheet_end) {
        super(empid, employee_name, employee_email, employee_phone,project_name, address);
        this.department_name = department_name;
        this.no_individuals_reporting = no_individuals_reporting;
        this.timeSheet_begin = timeSheet_begin;
        this.timeSheet_end = timeSheet_end;
    }



    public String getDepartment_name() {
        return department_name;
    }



    public Integer getNo_individuals_reporting() {
        return no_individuals_reporting;
    }



    public Date getTimeSheet_begin() {
        return timeSheet_begin;
    }



    public Date getTimeSheet_end() {
        return timeSheet_end;
    }



    public void setDepartment_name(String department_name) {
        this.department_name = department_name;
    }



    public void setNo_individuals_reporting(Integer no_individuals_reporting) {
        this.no_individuals_reporting = no_individuals_reporting;
    }
```



```java

    public Integer getNo_individuals_reporting() {
        return no_individuals_reporting;
    }



    public String getTimeSheet_begin() {
        return timeSheet_begin;
    }



    public String getTimeSheet_end() {
        return timeSheet_end;
    }



    public void setDepartment_name(String department_name) {
        this.department_name = department_name;
    }



    public void setNo_individuals_reporting(Integer no_individuals_reporting) {
        this.no_individuals_reporting = no_individuals_reporting;
    }



    public void setTimeSheet_begin(String timeSheet_begin) {
        this.timeSheet_begin = timeSheet_begin;
    }



    public void setTimeSheet_end(String timeSheet_end) {
        this.timeSheet_end = timeSheet_end;
    }



    public void viewManager() {
        System.out.println("########### Manager View Enabled###################");
        System.out.println("Employee ID is :: "+super.getEmpid());
        System.out.println("Employee Name is :: "+super.getEmployee_name());
        System.out.println("Employee Email is :: "+super.getEmployee_email());
        System.out.println("Employee phone is :: "+super.getEmpid());
        System.out.println("Employee address is :: "+super.getAddress().getZip_code()+"\t"+super.getAddress().getStreet()+super.getAddress().getCity()+"\t"+super.getAddress().getState()+"\t"+super.getAddre
        System.out.println("Employee department is ::"+this.department_name);
        System.out.println("Number of Individuals reporting for the project is ::"+this.no_individuals_reporting);
        System.out.println("The start time of the project is ::"+this.timeSheet_begin);
        System.out.println("The scheduled end time for project is :: "+this.timeSheet_end);
        System.out.println();
    }




```