**companydata Data Layer Documentation**

Add companydata.jar and mysql-connector-java-8.0.17.jar files to your classpath if running locally or to the “lib” folder under the WEB-INF folder for the war file.

You have to be on campus or using VPN.

You will need to add: import companydata.\*; to the top of your class that wants to access the Data Layer.

To use it (see included TestDataLayer.java example):

DataLayer dl = null;

try {

dl = new DataLayer("yourusername");

//call the DataLayer method you want

} catch (Exception e) {

//deal with the error – can be thrown by any of the methods below

} finally {

dl.close();

}

**Methods** (all throw an exception if there is an error, such as the requested object doesn’t exist)**:**

//Delete all Departments, Employees and Timecards for a company

//Returns the number of rows deleted

**public int deleteCompany(String companyName)**

//Insert a department

//Returns the inserted Department

**public Department insertDepartment(Department department)**

//Get all Departments for a given company

//Returns a list of Department objects

**public List<Department> getAllDepartment(String companyName)**

//Get a given Department for a given company

//Returns the requested Department

**public Department getDepartment(String companyName, int dept\_id**)

//Get a given Department for a given company

//Returns the requested Department

**public Department getDepartment(String companyName, String dept\_no**)

//Update a Department

//Returns the updated Department

**public Department updateDepartment(Department department)**

//Delete a given Department for a given company

//Returns the number of rows deleted

**public int deleteDepartment(String company, int dept\_id)**

//Insert an Employee

//Returns the inserted Employee

**public Employee insertEmployee(Employee employee)**

//Get a list of Employees for a given company

//Returns the list of Employee objects

**public List<Employee> getAllEmployee(String companyName)**

//Get the requested Employee

//Returns the requested Employee

**public Employee getEmployee(int emp\_id)**

//Update an Employee

//Returns the updated Employee object

**public Employee updateEmployee(Employee employee)**

//Delete the given Employee

//Returns the number of rows deleted

**public int deleteEmployee(int emp\_id)**

//Insert a Timecard

//Returns the inserted Timecard

**public Timecard insertTimecard(Timecard timecard)**

//Get all Timecards for a given Employee

//Returns the list of Timecard objects

**public List<Timecard> getAllTimecard(int emp\_id)**

//Get the requested Timecard

//Returns the requested Timecard

**public Timecard getTimecard(int timecard\_id)**

//Update a given Timecard

//Returns the updated Timecard

**public Timecard updateTimecard(Timecard timecard)**

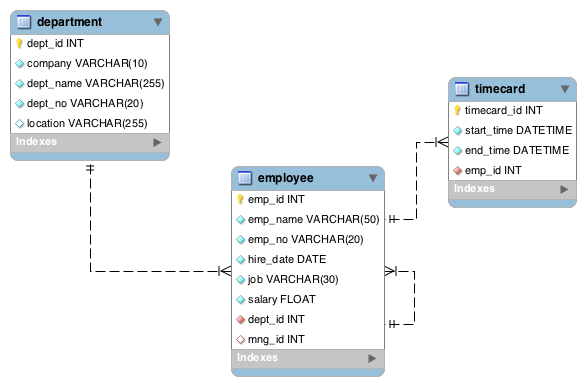
//Delete the given Timecard

//Returns the number of rows deleted

**public int deleteTimecard(int timecard\_id)**

//Close the connection

**public void close()**



The **employee** table has two foreign keys:

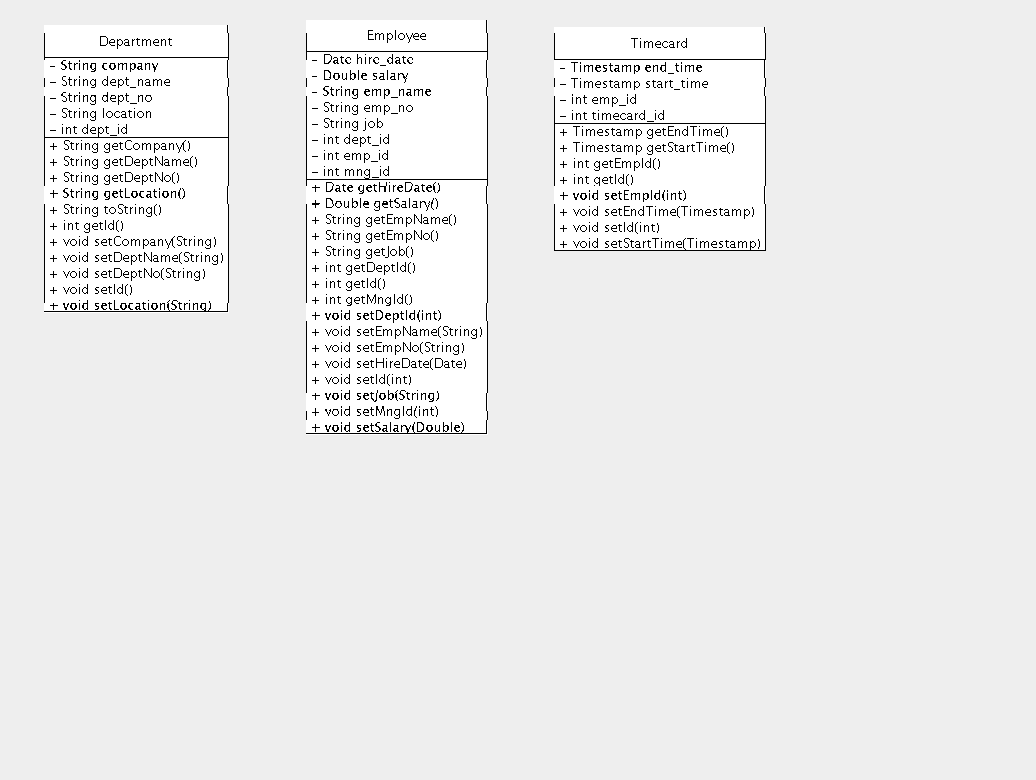
1. mng\_id references **employee**.emp\_id
2. dept\_id references **department**.dept\_id

The **employee** table has an additional unique index of emp\_no besides the the primary key

The **timecard** table has one foreign key:

1. emp\_id references **employee**.emp\_id

The **department** table has one additional unique index of dept\_no besides the primary key



**Each class has constructors of**:

**Employee**:

Employee(String emp\_name, String emp\_no, Date hire\_date,

String job, Double salary, int dept\_id, int mng\_id)

Employee(int emp\_id, String emp\_name, String emp\_no, Date hire\_date,

String job, Double salary, int dept\_id, int mng\_id)

**Department**:

Department(String company, String dept\_name, String dept\_no,

String location)

Department(int dept\_id, String company, String dept\_name,

String dept\_no, String location)

Department(String company, int dept\_id)

Department()

**Timecard**:

Timecard(Timestamp start\_time, Timestamp end\_time, int emp\_id)

Timecard(int timecard\_id, Timestamp start\_time, Timestamp end\_time,

int emp\_id)

Timecard()