1. Consider a database used by a company to keep track of its employees. Every employee has a name and a unique employee number, and is either paid a salary or an hourly rate. The database must record the annual salary of salaried employees and the hourly rate for hourly employees. The company has several divisions, and each division has several departments. Each division has a unique name, but department names are only unique within their division (two different divisions can have departments with the same name, e.g., “sales”). Most employees (except for division managers) are assigned to a department. Each division and each department has an annual budget. Each division and each department is managed by one of the salaried employees (but management positions are occasionally temporarily vacant). A department cannot exist unless it belongs to a division.

Draw an E/R diagram to model the information given above as completely and accurately as possible. Document any additional assumptions that you make.

Note: There is a strong case to be made for introducing a Business Units entity set which generalizes the Departments and Divisions entity sets.
2. For the simple E/R diagram below, write two different relational schemas that implement the given design.

Solution 1:
Account (acctNum, balance)
Checking (acctNum, fee)
Savings (acctNum, interest)

Solution 2:
Checking (acctNum, balance, fee)
Savings (acctNum, balance, interest)

Solution 3:
Account (acctNum, balance, fee, interest)
3. Consider part of the relational schema for the automobile insurance company database we have used as an example in class:

   People(ssn, name, city, state)
   Autos(vin, make, year)
   Owns(ssn, vin)

Write an expression in relational algebra to find:

a) The names of everyone who lives in Boston.

b) The names of everyone who owns a 2003 DODGE.

c) The ssn’s of everyone who does not own a car.

Write SQL statements to:

d) Create the autos table.

e) Add a car with vin = 123, make = TOYOTA, and year = 2004 to the autos table.

\[
\begin{align*}
\text{a) } & \Pi_{\text{name}} \left( \pi_{\text{city=Boston}} \left( \text{people} \right) \right) \\
\text{b) } & \Pi_{\text{name}} \left( \pi_{\text{make=Dodge} \land \text{year=2003}} \left( \text{autos} \right) \Join \text{owns} \Join \text{people} \right) \\
\text{c) } & \Pi_{\text{ssn}} \left( \text{people} \right) - \Pi_{\text{ssn}} \left( \text{owns} \right) \\
\text{d) } & \text{create table autos (vin varchar(50) primary key, make varchar(20), year integer)} \\
\text{e) } & \text{insert into autos values (‘123’, ‘TOYOTA’, 2004)}
\end{align*}
\]
4. Complete the following java application to print out a list of the names of people living in Massachusetts. Assume the given string variables are defined correctly.

```java
import java.sql.*;

public class MassachusettsResidentList {
    public static void main(String[] argv) {
        String username = "scott";
        String password = "tiger";
        String driverClass = "com.mysql.jdbc.Driver";
        String url = "jdbc:mysql:dat.cis.umassd.edu/db1";

        try {
            Class.forName(driverClass);
            Connection dbConnection = DriverManager.getConnection(url, username, password);
            Statement stmt = dbConnection.createStatement();
            ResultSet results = stmt.executeQuery("select name from people where state = 'MA'");
            while (results.next()) {
                System.out.println(results.getString(1));
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                dbConnection.close();
            } catch (Exception e) {
            }
        }
    }
}
```