Practice Midterm Exam Key

You will have 90 minutes for this exam, although you should not need that much. This exam is open-book and open-note, but you may not use the computer for anything other than accessing DyKnow notes or PDF files from Moodle. Please take some time to check your work. If you need extra space, write on the back. There are a total of 30 points on this exam.

For several questions on this exam, you will need the following database:

Employee

ssn	ename	age	salary
314159265	Ray Diaz	35	89000
271828182	Nathan Logg	42	78000
161803398	Phyllis Bonacci	34	55000
030102999	Bryan Decimali	32	65000
141421356	Hiram Pottanoose	50	100000

Works

essn	did	hours
314159265	1	20
314159265	2	20
271828182	1	10
271828182	2	20
271828182	3	5
271828182	4	5
161803398	2	10
161803398	3	30
030102999	2	20
030102999	3	20
030102999	4	20
141421356	4	40

Department

deptid	dname	mgrssn	budget
1	Hardware	314159265	531000
2	Firmware	314159265	420000
3	Software	161803398	678000
4	Sleepwear	141421356	88000

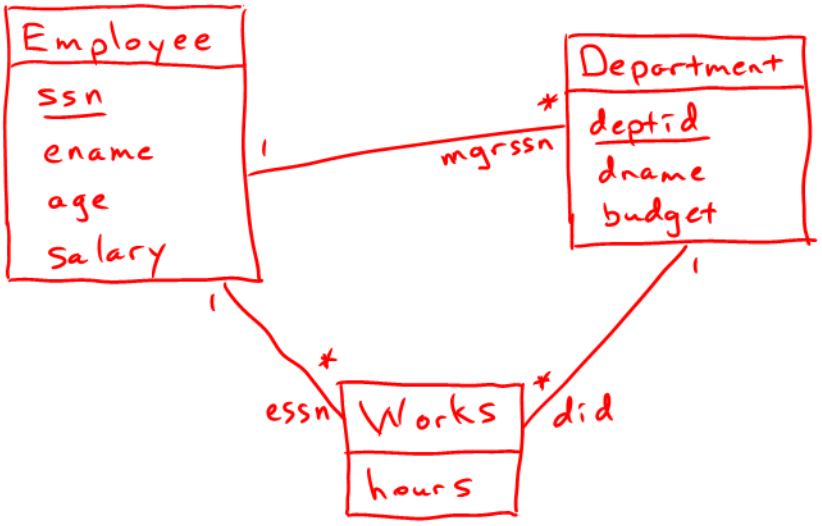
The Works table records the hours an employee spends working for a department each week; the mgrssn field identifies the manager of a department. Here is an SQL specification of these tables:

```
CREATE TABLE Employee (
  ssn INT not null,
  ename VARCHAR(20) not null,
  age INT not null,
  salary INT not null,
  PRIMARY KEY (ssn)
);

CREATE TABLE Department (
  deptid INT not null,
  dname VARCHAR(20) not null,
  mgrssn INT not null,
  budget INT not null,
  PRIMARY KEY (deptid),
  FOREIGN KEY (mgrssn) REFERENCES Employee
);

CREATE TABLE Works (
  essn INT not null,
  did INT not null,
  hours INT not null,
  FOREIGN KEY (essn) REFERENCES Employee,
  FOREIGN KEY (did) REFERENCES Department
);
```

1. (2 points) Draw a class diagram for the above database.



2. (3 points) List three different kinds of constraints expressed in the schema for this database.

primary key: ssn in Employee, deptid in Department

foreign key: essn from Works to Employee, etc.

null value: all attributes are marked "not null"

3. (1 point) What would be a useful integrity constraint for this database?

- 0 <= ssn < 1,000,000,000
- age limited to some valid range (e.g., 15-100)
- sum of hours worked per employee (per week) must be 40 (this is violated by the data...)

4. (8 points) For each of the following SQL queries against the above database, show the resulting table.

(a)
```
SELECT ename, salary
FROM Employee
WHERE age < 40;
```

Get names and salaries of employees under 40:

ename	salary
Ray Diaz	89000
Phyllis Bonacci	55000
Bryan Decimali	65000

(b)
```
SELECT dname, ename, budget, salary
FROM Employee, ~~Dept~~ Department
WHERE ssn = mgrssn
ORDER BY budget DESC;
```

Get department name, manager name, budget, and manager salary, in descending order by budget:

dname	ename	budget	salary
Software	Phyllis Bonacci	678000	55000
Hardware	Ray Diaz	531000	89000
Firmware	Ray Diaz	420000	89000
Sleepwear	Hiram Pottanoose	88000	100000

(c)
```
SELECT dname, SUM(salary)
FROM Employee, ~~Dept~~ Department, Works
WHERE ssn = essn AND deptid = did AND hours >= 20
GROUP BY deptid, dname;
```

Get total salary of employees with at least half-time assignment to each department:

dname	sumOfSalary
Hardware	89000
Firmware	232000
Software	120000
Sleepwear	165000

5. (8 points) Write the following queries in SQL against the above database schema.

   (a) Retrieve the names and ages of all employees who work in both the Firmware department and the Software department.

```
SELECT  e.ename, e.age
FROM    Employee e, Works w1, Works w2,
        Department d1, Department d2
WHERE   e.ssn = w1.essn  AND  e.ssn = w2.essn
  AND   w1.did = d1.deptid AND  w2.did = d2.deptid
  AND   d1.dname = 'Firmware'
  AND   d2.dname = 'Software'
```

f.k. conditions

   (b) Retrieve the names of all employees who are not managers.

```
SELECT  e.ename
FROM    Employee e
WHERE   e.ssn NOT IN
        (SELECT d.mgrssn
         FROM   Department d)
```

   (c) Retrieve the name of the manager of the department with the largest budget.

```
SELECT  e.ename
FROM    Employee e, Department d
WHERE   e.ssn = d.mgrssn
  AND   d.budget = (SELECT Max(d2.budget)
                    FROM   Department d2)
```

6. (2 points) We saw that the relational algebra join operation can be implemented using a product followed by a select. How can you use the join operation to implement the product?

product(T_1, T_2) = join(T_1, T_2, true)

↑ select all pairs of rows

7. (3 points) Finish this JDBC code fragment which executes the query from problem 4(a) above and prints the result in a simple table.

```
Statement stmt = conn.createStatement();
String query = "SELECT ename, salary FROM Employee WHERE age < 40";
ResultSet rs = stmt.executeQuery(query);

System.out.format("%20s %10s", "Employee name", "Salary");
while ( rs.next() ) {
    String ename = rs.getString("ename");
    int salary = rs.getInt("salary");

    System.out.format("%20s %10d", ename, salary );
}

rs.close();
```

(The format string `"%20s %10d"` says to print a string 20 columns wide, a space, and an integer 10 columns wide.)

8. (3 points) Suppose the Employee table in the above database had an additional column identifying each employee's supervisor (by giving their `ssn`). Does this represent a redundant relationship? Why or why not? What might be a better way to store this information?

- It is not redundant, because an employee may work in several departments — there does not need to be any connection between their supervisor and their department managers.
- If only some employees act as supervisors, their ssns could be kept in a separate "supervisor" table (a "subclass" of Employee);
  If the supervisor is one of the dept. mgrs. in which the employee works, could instead identify their "supervisory department" — this would track mgmt. changes.