Graded Problems for Module 4

The graded problems in module 4 involve SELECT statements for single table problems with conditions, joins with two tables, and row summaries involving single tables. You should execute the statements using Oracle, MySQL, or PostgreSQL.

To facilitate grading, please number the SQL statements and format them neatly. You do not need to show the result tables. Indicate in the beginning of your document if you used Oracle or MySQL.

1. List the city, state, and zip codes in the customer table. Your result should not have duplicates. (Hint: The DISTINCT keyword eliminates duplicates.)

```
SELECT DISTINCT city, state, zip
FROM customer;
```

2. List the name, department, phone number, and email address of employees with a phone number beginning with “3-”.

```
SELECT empname, department, phone, email
FROM employee
WHERE phone LIKE '3-%';
```

3. List all columns of the resource table with a rate between $10 and $20. Sort the result by rate.

```
SELECT *
FROM resourcetbl
WHERE rate BETWEEN 10 AND 20
ORDER BY rate;
```

4. List the event requests with a status of “Approved” or “Denied” and an authorized date in July 2018. Include the event number, authorization date, and status in the output. (Hint: see the examples in Module 4 for date constants in Oracle and MySQL.)

```
SELECT eventno, dateauth, status
FROM eventrequest
WHERE (status IN ('Approved', 'Denied') AND (dateauth BETWEEN '01-Jul-2018' AND '31-07-2018'));
```

5. List the location number and name of locations that are part of the “Basketball arena”. Your WHERE clause should not have a condition involving the facility number compared to a constant (“F101”). Instead, you should use a condition on the FacName column for the value of “Basketball arena”.

```
SELECT location.locno, location.locname, facility.facname
FROM location
INNER JOIN facility
ON facility.facno = location.facno
WHERE facname = 'Basketball arena';
```

6. For each event plan, list the plan number, count of the event plan lines, and sum of the number of resources assigned. For example, plan number “P100” has 4 lines and 7 resources assigned. You only need to consider event plans that have at least one line.

```
SELECT planno, COUNT(*) AS EventPlanLineCount, SUM(numberfld) AS ResourceSum
FROM eventplanline
GROUP BY planno
```

```
HAVING COUNT(*) >= 1;
```