1. **/\*5.13\*/**
2. /\*
3. a.
4. We need to know the number of attributes and names of attributes of r to decide the number and names of columns in the table.
5. \*/
6. /\*
7. b.
8. We can use the JDBC methods getColumnCount() and getColumnName(int) to get the required information.
9. \*/
10. //c.
11. **static** **void** printTable(String r)
12. {
13. **try**
14. {
15. Class.forName("oraclejdbc.driver.OracleDriver"); Connection conn =  DriverManager.getConnection(    "jdbc:oracle:thin:@db.yale.edu:2000:univdb",user,passwd);   Statement stmt = conn.createStatement();
16. ResultSet rs = stmt.ExecuteQuery(r);
17. ResultSetMetaData rsmd = rs.getMetaData();
18. **int** count = rsmd.getColumnCount(); System.out.println("<tr>");
19. **for**(**int** i=1;i<=count;i++)
20. {
21. System.out.println("<td>"+rsmd.getColumnName(i)+"</td>");
22. }
23. System.out.println("</tr>");
24. **while**(rs.next()
25. {
26. System.out.println("<tr>");
27. **for**(**int** i=1;i<=count;i++)
28. {
29. System.out.println("<td>"+rs.getString(i)+"</td>");
30. }
31. System.out.println("</tr>");
32. }
33. stmt.close();
34. conn.close();
35. }
36. **catch**(SQLException sqle)
37. {
38. System.out.println("SQLException : " + sqle);
39. }
40. }
41. /\*
42. 5.15. a.
43. \*/
44. **create** **function** avg\_salary(cname **varchar**(15))
45. **returns** **integer**
46. **declare** result **integer**;
47. **select** avg(salary) **into** result
48. **from** works
49. **where** works.company\_name = cname
50. **return** result;
51. **end**
52. **select** company\_name
53. **from** works
54. **where** avg\_salary(company\_name)>avg\_salary("First Bank Corporation")
55. /\*
56. 5.15. b.
57. \*/
58. **select** company\_name
59. **from** works
60. **group** **by** company\_name
61. **group** **by** company\_name
62. **having** avg(salary)>(**select** avg(salary)
63. **from** works
64. /\*
65. 5.21
66. We define triggers **for** each relationship whose **primary** **key** **is** referenced **by** a **foreign** **key** **to** another relationship. The **trigger** **is** fired whenever a tuple **is** removed **from** the reference relationship. The **action** performed **by** the **trigger** **is** **to** access all reference relationships and **delete** all tuples whose **foreign** **key** attribute **values** are the same **as** the **primary** **key** attribute **values** **of** the deleted tuples in the reference relationship. These triggers will handle the **on** **delete** **cascade** operation.
67. \*/