CS4337 : Database Systems
Term Project Write-up

Matthew McMillian
mgm160130@utdallas.edu

December 1, 2018

1. Problem Description:

   Our objective in this project is to design and implement a database based on the given criteria. Our database resembles that of a business, containing tables relating to employees, customers, products, etc. Using the information given, we are tasked with designing multiple distinct diagrams such as a Relational, Conceptual EER, and Logical EER diagram. Then, we are tasked with defining views to display a subset of our data and we are supposed to implement queries for data retrieval.

2. Project Questions:

   (a) Can you think of 5 more rules (other than the ones explicitly described above) that are likely to be used in a company?

   - Departments would have teams of employees. In leu of this, employees would also belong to teams.
   - As well as marketing sites, there could be offices that different (and sometimes the same departments) and employees would work at, and it might be useful to keep track of this information.
   - As we've seen in the textbook, for insurance reasons it may be beneficial to keep track of employees dependents.
   - As we've seen in the textbook, teams or groups of employees have projects to work on. Adding a project entity to keep track of the work done by employees would be beneficial.
   - If we have projects, we would need a relation for a department to control a project (since it has to be assigned somewhere).

(b) Is the ability to model super-class/subclass relationships likely to be important in such an environment? Why or why not?

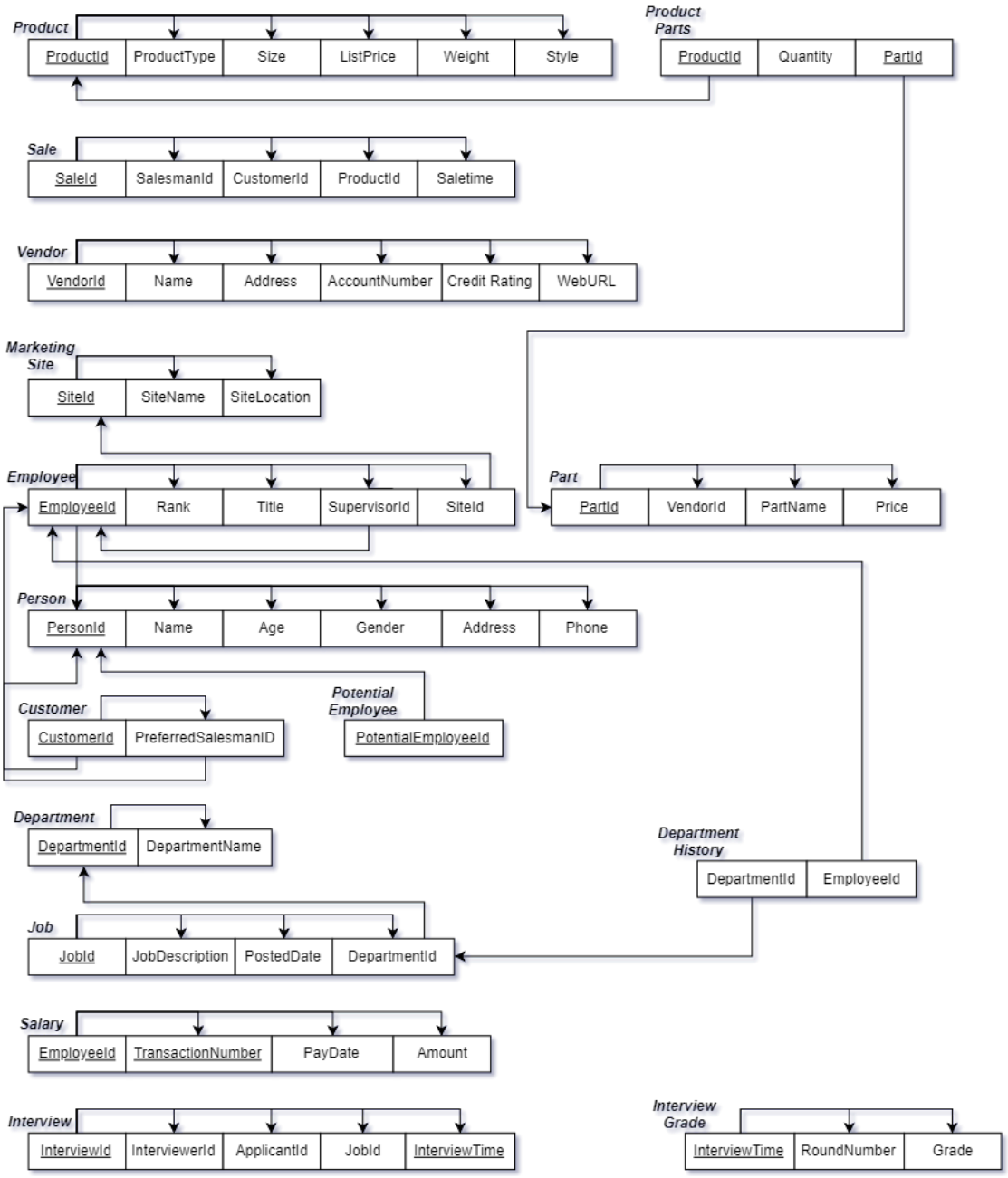
- It is important to model this superclass / subclass relationship. In this environment, we have many different types of people that we can interact with at the company. These people also have differing attributes to them. For example, we would not want a customer to have a salary. More generally, we don't want or care to keep track of a customers information such as their name, sex, birth date, ... since it would be useless information. Having the ability to categorize these types of potential people can make it easy to store data efficiently and not overpopulate tables.

(c) Justify using a Relational DBMS like Oracle for this project.

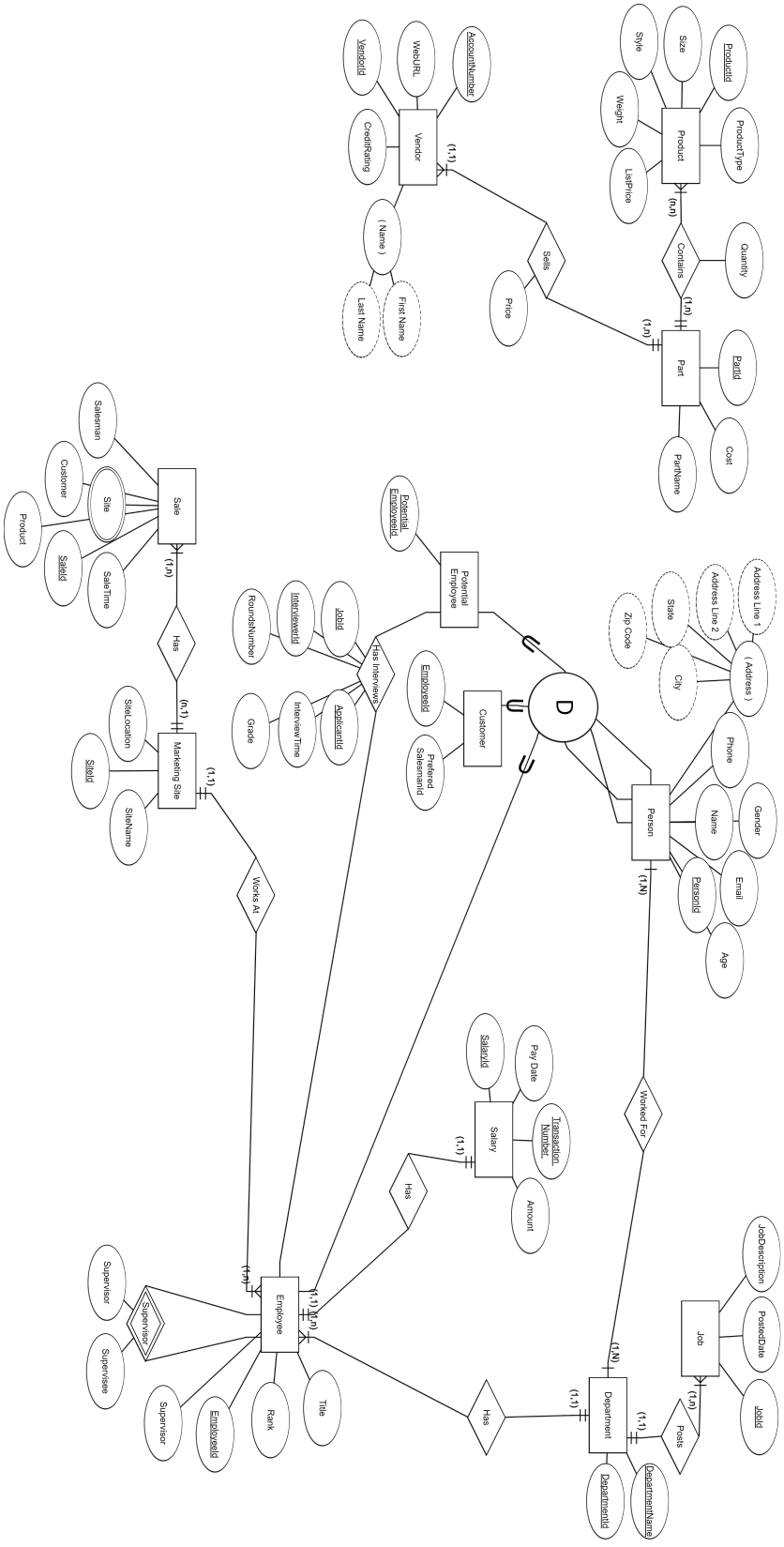
- In a company it is important to have consistent information. Especially in an enterprise environment, data is flowing in and out at rapid speeds. Because of this we need a database structure that supports atomic operations (CRUD operations) so that we can preserve the integrity of our fast-paced data. Another thing that a company structure has is a variety of data. Since almost all the entities in our database are related in some way, it may be useful to perform joins to get specific subsets of data. In a Non-Relational DBMS joins are nonexistent. To even simulate what a join can do, you would have to perform multiple queries which can be very inefficient. Due to our atomicity and relational constraints listed above, it is safe to say that in this scenario a Relational DBMS can be used to store our data.

3. Project Diagrams:

- Relational Diagram: (Note this is in 3NF)



- EER Diagram:


Vendor
VendorId
WebURL
AccountNumber
CreditRating
(Name)
Last Name
First Name
Sells
Price
(1,1)
(1,n)
Product
Size
ProductId
Style
ProductType
Weight
ListPrice
Contains
Quantity
(0,n)
(1,n)
Part
PartId
PartName
Cost
Potential Employee
PotentialEmployeeId
D
Customer
EmployeeId
PreferedSalesmanId
Has Interviews
JobId
InterviewId
RoundsNumber
InterviewTime
ApplicantId
Grade
Person
Address Line 1
Address Line 2
(Address)
State
City
Zip Code
Phone
Name
Gender
PersonId
Email
Age
(1,N)
Sale
Salesman
Customer
Site
Product
SaleId
SaleTime
(1,n)
Has
(0,1)
Marketing Site
SiteLocation
SiteId
SiteName
(1,1)
Works At
Worked For
Salary
SalaryId
Pay Date
Transaction Number
Amount
(1,1)
Has
Job
JobDescription
PosteDate
JobId
(1,N)
(1,1)
Department
DepartmentId
DepartmentName
Posts
(1,n)
(1,1)
Has
Employee
Supervisor
Supervisee
Supervisor
EmployeeId
Rank
Title
(1,n)
(1,1)(0,n)
Supervisor

4. SQL Statements

- **Database Views:**

```
CREATE VIEW View1 (Average_Salary, EmployeeId) AS (
  SELECT SUM(S.Amount)/COUNT(S.Amount), S.EmployeeId FROM
      Employee E, Salary S WHERE (E.EmployeeId = S.
      EmployeeId) GROUP BY S.EmployeeId
);

CREATE VIEW View2 (Passed_Interview_Rounds, JobId,
    ApplicantId) AS (
  SELECT COUNT(*), X.JobId, X.POTENTIALEMPLOYEEID FROM
    (SELECT DISTINCT I.INTERVIEWID, I.JobId, PE.
       POTENTIALEMPLOYEEID
       FROM PotentialEmployee PE, Interview I
      WHERE (I.APPLICANTID = PE.POTENTIALEMPLOYEEID)) X,
          Interview_Grade IG
   WHERE IG.INTERVIEWID = X.INTERVIEWID AND IG.GRADE > 60
     GROUP BY X.POTENTIALEMPLOYEEID, X.JobId
);

CREATE VIEW View3 (Number_Of_Items, ProductType) AS (
  SELECT COUNT(*), P.PRODUCTTYPE FROM PRODUCT P GROUP BY P
      .PRODUCTTYPE
);

CREATE VIEW View4 (Part_Purchase_Cost, Product) AS (
  SELECT SUM(X.PRICE * X.QUANTITY), X.PRODUCTTYPE
   FROM (SELECT DISTINCT PART.PRICE, PP.Quantity, PRODUCT.
       PRODUCTTYPE
            FROM PRODUCT PRODUCT, PART PART, PRODUCT_PARTS
                PP
           WHERE (PP.PRODUCTID = PRODUCT.PRODUCTID AND PART
               .PARTID = PP.PARTID)) X
   GROUP BY X.PRODUCTTYPE
);
```

- **Database Creation:**

```
CREATE TABLE PRODUCT (
    ProductId NUMBER(9) NOT NULL,
    ProductType VARCHAR2(20) NOT NULL,
    ProductSize NUMBER(9,2) NOT NULL,
    ListPrice NUMBER(9,2) NOT NULL,
    Weight NUMBER(9,2) NOT NULL,
    ProductStyle VARCHAR2(20) NOT NULL,

   PRIMARY KEY (ProductId)
);
--DROP TABLE PRODUCT CASCADE CONSTRAINTS;

CREATE TABLE PERSON (
    PersonId NUMBER(9) NOT NULL,
    Name VARCHAR2(40) NOT NULL,
    Age NUMBER(3) NOT NULL,
    Gender VARCHAR2(15) NOT NULL,
    Address VARCHAR2(60) NOT NULL,
    Phone NUMBER(10) NOT NULL,
    Email VARCHAR2(30) NOT NULL,

   PRIMARY KEY (PersonId)
);
--DROP TABLE PERSON CASCADE CONSTRAINTS;

CREATE TABLE EMPLOYEE (
    EmployeeId NUMBER(9) NOT NULL,
    Rank VARCHAR2(40) NOT NULL,
    Title VARCHAR2(40) NOT NULL,
    SupervisorId NUMBER(9),
    SiteId NUMBER(9) NOT NULL,

   PRIMARY KEY (EmployeeId)
);
--DROP TABLE EMPLOYEE CASCADE CONSTRAINTS;

CREATE TABLE Customer (
    CustomerId NUMBER(9) NOT NULL,
    PreferredSalesman NUMBER(9) NOT NULL,

   PRIMARY KEY (CustomerId)
);

```

```
44 CREATE TABLE PotentialEmployee (
45    PotentialEmployeeId NUMBER(9) NOT NULL,
46
47   PRIMARY KEY (PotentialEmployeeId)
48 );
49
50 CREATE TABLE Interview (
51    InterviewId NUMBER(9) NOT NULL,
52    InterviewerId NUMBER(9) NOT NULL,
53    ApplicantId NUMBER(9) NOT NULL,
54    JobId NUMBER(9) NOT NULL,
55
56   PRIMARY KEY (InterviewId)
57 );
58 --DROP TABLE Interview CASCADE CONSTRAINTS;
59
60 CREATE TABLE Sale (
61    SaleId NUMBER(9) NOT NULL,
62    ProductId NUMBER(9) NOT NULL,
63    CustomerId NUMBER(9) NOT NULL,
64    SalesmanId NUMBER(9) NOT NULL,
65    SaleTime DATE NOT NULL,
66
67   PRIMARY KEY (SaleId)
68 );
69
70 --CREATE TABLE Sale_Site (
71 --   SalesmanId NUMBER(9) NOT NULL,
72 --   SiteId NUMBER(9) NOT NULL,
73 --
74 --   PRIMARY KEY (SalesmanId)
75 --);
76 --DROP TABLE SALE_SITE CASCADE CONSTRAINTS;
77
78 CREATE TABLE MarketingSite (
79    SiteId NUMBER(9) NOT NULL,
80    SiteName VARCHAR2(20) NOT NULL,
81    SiteLocation VARCHAR2(30) NOT NULL,
82
83   PRIMARY KEY (SiteId)
84 );
85
86 CREATE TABLE Salary (
87    EmployeeId NUMBER(9) NOT NULL,
88    TransactionNumber NUMBER(9) NOT NULL,
```

```
89      PayDate DATE NOT NULL,
90      Amount NUMBER(9,2) NOT NULL,
91
92      CONSTRAINT EMP_TRS_UNI UNIQUE (EmployeeId,
            TransactionNumber)
93  --  PRIMARY KEY (EmployeeId)
94  );
95  --DROP TABLE SALARY CASCADE CONSTRAINTS;
96
97  --CREATE TABLE Salary_Transaction (
98  --  TransactionNumber NUMBER(9) NOT NULL,
99  --  PayDate DATE NOT NULL,
100 --  Amount NUMBER(9,2) NOT NULL
101 --);
102
103 --DROP TABLE SALARY_TRANSACTION CASCADE CONSTRAINTS;
104
105 CREATE TABLE Vendor (
106     VendorId NUMBER(9) NOT NULL,
107     Name VARCHAR2(30) NOT NULL,
108     Address VARCHAR2(60) NOT NULL,
109     AccountNumber NUMBER(9) NOT NULL,
110     CreditRating NUMBER(6) NOT NULL,
111     WebURL VARCHAR2(30) NOT NULL,
112
113     PRIMARY KEY (VendorId)
114 );
115 --DROP TABLE VENDOR CASCADE CONSTRAINTS;
116
117 CREATE TABLE Job (
118     JobId NUMBER(9) NOT NULL,
119     JobDescription VARCHAR2(60) NOT NULL,
120     PostedDate DATE NOT NULL,
121     DepartmentId NUMBER(9) NOT NULL,
122
123     PRIMARY KEY (JobId)
124 );
125
126 CREATE TABLE Interview_Grade (
127     InterviewID NUMBER(9) NOT NULL,
128     RoundNumber NUMBER(1) NOT NULL,
129     Grade NUMBER(3) NOT NULL,
130
131     CONSTRAINT INT_GRADE_UNI UNIQUE (InterviewID,
            RoundNumber)
```

```
132 --   PRIMARY KEY (InterviewID)
133 );
134 --DROP TABLE INTERVIEW_GRADE CASCADE CONSTRAINTS;
135
136
137 CREATE TABLE Part (
138    PartId NUMBER(9) NOT NULL,
139    PartName VARCHAR(30) NOT NULL,
140    Price NUMBER(5,2) NOT NULL,
141    VendorId NUMBER(9) NOT NULL,
142
143   PRIMARY KEY (PartId)
144 );
145 --DROP TABLE PART CASCADE CONSTRAINTS;
146
147 CREATE TABLE Department (
148    DepartmentId NUMBER(9) NOT NULL,
149    DepartmentName VARCHAR2(30) NOT NULL,
150
151   PRIMARY KEY (DepartmentId)
152 );
153
154 CREATE TABLE Product_Parts (
155    ProductId NUMBER(9) NOT NULL,
156    Quantity NUMBER(9) NOT NULL,
157    PartId NUMBER(9) NOT NULL
158 );
159 --DROP TABLE PRODUCT_PARTS CASCADE CONSTRAINTS;
160
161 CREATE TABLE Department_History (
162      DepartmentId NUMBER(9) NOT NULL,
163      EmployeeId NUMBER(9) NOT NULL
164 );
165
166 ALTER TABLE Department_History ADD FOREIGN KEY (
      DepartmentId) REFERENCES Department(DepartmentId);
167 ALTER TABLE Department_History ADD FOREIGN KEY (EmployeeId
      ) REFERENCES Employee(EmployeeId);
168
169 ALTER TABLE Employee ADD FOREIGN KEY (EmployeeId)
     REFERENCES Person(PersonId);
170 ALTER TABLE Employee ADD FOREIGN KEY (SiteId) REFERENCES
      MarketingSite(SiteId);
171
172 ALTER TABLE Customer ADD FOREIGN KEY (CustomerId)
```

```
    REFERENCES Person(PersonId);
173 ALTER TABLE Customer ADD FOREIGN KEY (PreferredSalesman)
    REFERENCES Employee(EmployeeId);
174
175 ALTER TABLE PotentialEmployee ADD FOREIGN KEY (
    PotentialEmployeeId) REFERENCES Person(PersonId);
176
177 ALTER TABLE Sale ADD FOREIGN KEY (ProductId) REFERENCES
    Product(ProductId);
178 ALTER TABLE Sale ADD FOREIGN KEY (CustomerId) REFERENCES
    Customer(CustomerId);
179 ALTER TABLE Sale ADD FOREIGN KEY (SalesmanId) REFERENCES
    Employee(EmployeeId);
180
181 ALTER TABLE Sale_Site ADD FOREIGN KEY (SalesmanId)
    REFERENCES Employee(EmployeeId);
182 ALTER TABLE Sale_Site ADD FOREIGN KEY (SiteId) REFERENCES
    MarketingSite(SiteId);
183
184 ALTER TABLE Salary ADD FOREIGN KEY (EmployeeId) REFERENCES
     Employee(EmployeeId);
185
186 --ALTER TABLE Salary_Transaction ADD FOREIGN KEY (
    TransactionNumber) REFERENCES Salary(TransactionNumber)
    ;
187
188 ALTER TABLE Interview ADD FOREIGN KEY (InterviewerId)
    REFERENCES Employee(EmployeeId);
189 ALTER TABLE Interview ADD FOREIGN KEY (ApplicantId)
    REFERENCES PotentialEmployee(PotentialEmployeeId);
190 ALTER TABLE Interview ADD FOREIGN KEY (JobId) REFERENCES
    Job(JobId);
191
192 ALTER TABLE Job ADD FOREIGN KEY (DepartmentId) REFERENCES
    Department(DepartmentId);
193
194 ALTER TABLE Part ADD FOREIGN KEY (VendorID) REFERENCES
    VENDOR(
195
196 ALTER TABLE Interview_Grade ADD FOREIGN KEY (InterviewId)
    REFERENCES Interview(InterviewId);
197
198 ALTER TABLE Product_Parts ADD FOREIGN KEY (ProductId)
    REFERENCES Product(ProductId);
199 ALTER TABLE Product_Parts ADD FOREIGN KEY (PartId)
```

```
REFERENCES Part(PartId);
```

- **Database Insertion (For Testing):**

```
1  --<View 1>
2  INSERT INTO MarketingSite(SiteId, SiteName, SiteLocation)
       VALUES (1, 'Mathematics Research', 'San Jose');
3
4  INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
       Phone, Email) VALUES (2048, 'Matthew McMillian', 20, '
       Male', 'UT Dallas', 3335557777, 'testemail@ut.edu');
5  INSERT INTO EMPLOYEE (EmployeeId, Rank, Title,
       SupervisorId, SiteId) VALUES (2048, 'Lead Director', '
       Vice President', NULL, 1);
6
7  INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
       Phone, Email) VALUES (2049, 'Brandon Tran', 21, 'Male',
        'UT Dallas', 4448889999, 'testemail@ut.edu');
8  INSERT INTO EMPLOYEE (EmployeeId, Rank, Title,
       SupervisorId, SiteId) VALUES (2049, 'Director', 'UX
       Designer', 2048, 1);
9
10 INSERT INTO SALARY (EmployeeId, TransactionNumber, PayDate
       , Amount) VALUES (2048, 1000, TO_DATE('2018/02/01','
       yyyy/mm/dd'), 10.00);
11 INSERT INTO SALARY (EmployeeId, TransactionNumber, PayDate
       , Amount) VALUES (2048, 1001, TO_DATE('2018/02/02','
       yyyy/mm/dd'), 40.00);
12 INSERT INTO SALARY (EmployeeId, TransactionNumber, PayDate
       , Amount) VALUES (2048, 1002, TO_DATE('2018/02/02','
       yyyy/mm/dd'), 240.00);
13
14 INSERT INTO SALARY (EmployeeId, TransactionNumber, PayDate
       , Amount) VALUES (2049, 1000, TO_DATE('2018/02/01','
       yyyy/mm/dd'), 30.00);
15 INSERT INTO SALARY (EmployeeId, TransactionNumber, PayDate
       , Amount) VALUES (2049, 1001, TO_DATE('2018/027/02','
       yyyy/mm/dd'), 80.00);
16 COMMIT;
17
18 --<View 2>
19 INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
       Phone, Email) VALUES (1048, 'Seong Wang', 20, 'Male', '
       UT Dallas', 1112223333, 'testemail@ut.edu');
20 INSERT INTO POTENTIALEMPLOYEE (POTENTIALEMPLOYEEID) VALUES
       (1048);
21
```

```
22 INSERT INTO DEPARTMENT (DepartmentId, DepartmentName)
     VALUES (3, 'Mathematics');
23 INSERT INTO JOB (JobId, JobDescription, PostedDate,
     DepartmentId) VALUES (3, 'Math Research Intern',
     TO_DATE('2018/05/03','yyyy/mm/dd'), 3);
24
25 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
     ApplicantId, JobId) VALUES (10, 2048, 1048, 3);
26 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (10, 1, 60);
27 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (10, 2, 70);
28 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (10, 3, 50);
29 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (10, 4, 80);
30 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (10, 5, 90);
31
32 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (82, 1, 60);
33 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (82, 2, 70);
34 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (82, 3, 50);
35 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (82, 4, 80);
36 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (82, 5, 90);
37
38 INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
     Phone, Email) VALUES (1049, 'Kion Smith', 20, 'Male', '
     UT Dallas', 3454563456, 'testemail@ut.edu');
39 INSERT INTO POTENTIALEMPLOYEE (POTENTIALEMPLOYEEID) VALUES
     (1049);
40
41 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
     ApplicantId, JobId) VALUES (11, 2048, 1049, 3);
42 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (11, 1, 30);
43 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (11, 2, 20);
44 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
     Grade) VALUES (11, 3, 50);
45 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
```

```
    Grade) VALUES (11, 4, 60);
46 INSERT INTO INTERVIEW_GRADE (InterviewId, RoundNumber,
    Grade) VALUES (11, 5, 70);
47
48 --<View 3, 4>
49 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (1, 'Crowbar',
    20, 19.99, 5.31, 'Red-Metal');
50 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (3, 'Crowbar',
    20, 19.99, 5.31, 'Blue-Metal');
51 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (4, 'Crowbar',
    20, 21.99, 5.31, 'Yellow-Metal');
52 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (2, 'Portal Gun
    ', 23, 119.99, 10.31, 'Aperture White');
53 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (6, 'Textbook',
     15, 120.99, 2.59, 'Statistical Analysis');
54 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (7, 'Textbook',
     15, 220.99, 2.59, 'Numerical Analysis');
55 INSERT INTO PRODUCT (ProductId, ProductType, ProductSize,
    ListPrice, Weight, ProductStyle) VALUES (8, 'Textbook',
     15, 260.99, 2.59, 'Database Analysis');
56
57 INSERT INTO PART (PartId, PartName, Price, VendorId)
    VALUES (1, 'Handle', 3.99, 1);
58 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
     (2, 'Metal Bob', 1.99, 2);
59 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (1, 1, 1);
60 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (1, 1, 2);
61 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (4, 1, 1);
62 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (4, 1, 2);
63 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (3, 1, 1);
64 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
    VALUES (3, 1, 2);
65
66 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
```

```
    (3, 'Quantum Parts', 6.99, 2);
67 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
    (4, 'Metallic Plating', 0.99, 3);
68 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (2, 3, 3);
69 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (2, 4, 4);
70
71 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
    (5, 'High-Quality Paper', 4.99, 2);
72 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
    (6, 'Cardboard Cover', 0.99, 3);
73 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (6, 1, 5);
74 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (6, 50, 6);
75 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (7, 1, 5);
76 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (7, 50, 6);
77 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (8, 1, 5);
78 INSERT INTO Product_Parts (ProductId, Quantity, PartId)
   VALUES (8, 50, 6);
79
80 --<Query 1>
81 INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
    Phone, Email) VALUES (1001, 'Hellen Cole', 30, 'Female'
    , '999 New York City', 2819995555, 'testemail@ut.edu');
82 INSERT INTO POTENTIALEMPLOYEE (POTENTIALEMPLOYEEID) VALUES
    (1001);
83
84 INSERT INTO JOB (JobId, JobDescription, PostedDate,
    DepartmentId) VALUES (11111, 'Summer Computer Science
    Intern', TO_DATE('2018/02/01','yyyy/mm/dd'), 3);
85
86 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
    ApplicantId, JobId) VALUES (1, 2048, 1001, 11111);
87 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
    ApplicantId, JobId) VALUES (2, 2049, 1001, 11111);
88
89 --<Query 2>
90 INSERT INTO DEPARTMENT (DepartmentId, DepartmentName)
   VALUES (12, 'Marketing');
91 INSERT INTO JOB (JobId, JobDescription, PostedDate,
```

```
    DepartmentId) VALUES (86, 'Marketing Intern, Fall',
    TO_DATE('2011/03/03','yyyy/mm/dd'), 12);
92  INSERT INTO JOB (JobId, JobDescription, PostedDate,
    DepartmentId) VALUES (87, 'Marketing Intern, Summer',
    TO_DATE('2011/03/05','yyyy/mm/dd'), 12);
93  INSERT INTO JOB (JobId, JobDescription, PostedDate,
    DepartmentId) VALUES (88, 'Marketing Intern, Spring',
    TO_DATE('2011/03/02','yyyy/mm/dd'), 12);
94  INSERT INTO JOB (JobId, JobDescription, PostedDate,
    DepartmentId) VALUES (89, 'Marketing Intern, Winter',
    TO_DATE('2011/03/09','yyyy/mm/dd'), 12);
95  INSERT INTO JOB (JobId, JobDescription, PostedDate,
    DepartmentId) VALUES (90, 'On-Site Sales Intern',
    TO_DATE('2012/04/09','yyyy/mm/dd'), 12);
96
97  --<Query 4>
98  INSERT INTO MarketingSite(SiteId, SiteName, SiteLocation)
    VALUES (2, 'Omega', 'Houston');
99  INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
    Phone, Email) VALUES (2050, 'Hector Quear', 27, 'Male',
     'Austin', 3335557777, 'testemail@ut.edu');
100 INSERT INTO EMPLOYEE (EmployeeId, Rank, Title,
    SupervisorId, SiteId) VALUES (2050, 'Salesman', 'Sr.',
    2048 , 2);
101
102 INSERT INTO PERSON (PersonId, Name, Age, Gender, Address,
    Phone, Email) VALUES (4001, 'John Bimbo', 42, 'Male', '
    999 Los Muertos', 6775663444, 'testemail@ut.edu');
103 INSERT INTO CUSTOMER (CustomerId, PreferredSalesman)
    VALUES (4001, 2049);
104
105 INSERT INTO SALE (SaleId, ProductId, CustomerId,
    SalesmanId, SaleTime) VALUES (301, 1, 4001, 2049,
    TO_DATE('2011/04/03','yyyy/mm/dd'));
106 INSERT INTO SALE (SaleId, ProductId, CustomerId,
    SalesmanId, SaleTime) VALUES (302, 1, 4001, 2049,
    TO_DATE('2011/03/01','yyyy/mm/dd'));
107
108 INSERT INTO SALE (SaleId, ProductId, CustomerId,
    SalesmanId, SaleTime) VALUES (311, 3, 4001, 2050,
    TO_DATE('2011/04/03','yyyy/mm/dd'));
109 INSERT INTO SALE (SaleId, ProductId, CustomerId,
    SalesmanId, SaleTime) VALUES (312, 2, 4001, 2050,
    TO_DATE('2011/02/01','yyyy/mm/dd'));
110
```

```
111 --<Query 6>
112 INSERT INTO SALE (SaleId, ProductId, CustomerId,
        SalesmanId, SaleTime) VALUES (612, 8, 4001, 2050,
        TO_DATE('2011/02/05','yyyy/mm/dd'));
113 INSERT INTO SALE (SaleId, ProductId, CustomerId,
        SalesmanId, SaleTime) VALUES (613, 7, 4001, 2050,
        TO_DATE('2011/02/06','yyyy/mm/dd'));
114
115 --<Query 7>
116 INSERT INTO JOB (JobId, JobDescription, PostedDate,
        DepartmentId) VALUES (101, 'Receptionist Intern',
        TO_DATE('2011/01/06','yyyy/mm/dd'), 12);
117
118 --<Query 8>
119 INSERT INTO JOB (JobId, JobDescription, PostedDate,
        DepartmentId) VALUES (12345, 'Busniess Insider Intern',
         TO_DATE('2011/01/06','yyyy/mm/dd'), 12);
120 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
        ApplicantId, JobId) VALUES (83, 2048, 2049, 12345);
121 INSERT INTO INTERVIEW (InterviewId, InterviewerId,
        ApplicantId, JobId) VALUES (82, 2048, 1048, 12345);
122
123 --<Query 11>
124 -- Insert Matt into each department for testing
125 INSERT INTO DEPARTMENT_HISTORY(DepartmentId, EmployeeId)
        VALUES (3, 2048);
126 INSERT INTO DEPARTMENT_HISTORY(DepartmentId, EmployeeId)
        VALUES (3, 2049);
127 INSERT INTO DEPARTMENT_HISTORY(DepartmentId, EmployeeId)
        VALUES (12, 2048);
128
129 --<Query 13>
130 -- Add person who applies for and passes all jobs, and add
         someone who applies for all jobs but does not pass
        them all
131
132 --<Query 15>
133 INSERT INTO VENDOR (VendorId, Name, Address, AccountNumber
        , CreditRating, WebURL) VALUES (2, 'Jacks Parts', '500
        W Parker St', 999888777, 200, 'www.test.com');
134 INSERT INTO PART (PartId, PartName,Price, VendorId) VALUES
         (15, 'Cup', 2.99, 2);
135
136 INSERT INTO VENDOR (VendorId, Name, Address, AccountNumber
        , CreditRating, WebURL) VALUES (3, 'Bills Parts', '300
```

```
        W Parker St', 124432134, 210, 'www.test.com');
137 INSERT INTO PART (PartId, PartName, Price, VendorId) VALUES
        (17, 'Cup', 1.99, 3);
```

- **Database Queries:**

```
1  --<Query 1>
2  SELECT X.InterviewerId, P.Name
3    FROM (SELECT DISTINCT I.InterviewerId
4              FROM EMPLOYEE E, Person P, INTERVIEW I
5             WHERE (I.ApplicantId = P.PersonId AND P.Name = '
                  Hellen Cole' AND I.JobId = '11111')) X, Person
                   P
6   WHERE X.InterviewerId = P.PersonId;
7
8  --<Query 2>
9  SELECT J.JobId
10   FROM Department D, Job J
11  WHERE (J.DepartmentId = D.DepartmentId AND J.POSTEDDATE
       BETWEEN TO_DATE('2011/03/01','yyyy/mm/dd') and TO_DATE
       ('2011/03/31','yyyy/mm/dd'));
12
13 --<Query 3>
14 SELECT E.EmployeeId, P.Name FROM EMPLOYEE E, Person P
     WHERE (E.EmployeeId = P.PersonId AND E.SupervisorId IS
     NULL);
15
16 --<Query 4>
17 SELECT M.SiteId, M.SiteLocation
18   FROM SALE S, EMPLOYEE E, MARKETINGSITE M
19  WHERE (S.SALETIME NOT BETWEEN TO_DATE('2011/03/01','yyyy/
       mm/dd') AND TO_DATE('2011/03/31','yyyy/mm/dd') AND S.
       SALESMANID = E.EmployeeId AND E.SITEID = M.SITEID)
20
21    MINUS
22
23 SELECT M.SiteId, M.SiteLocation
24   FROM SALE S, EMPLOYEE E, MARKETINGSITE M
25  WHERE (S.SALETIME BETWEEN TO_DATE('2011/03/01','yyyy/mm/
       dd') AND TO_DATE('2011/03/31','yyyy/mm/dd') AND S.
       SALESMANID = E.EmployeeId AND E.SITEID = M.SITEID);
26
27 --<Query 5 Is this what he wants?>
28 SELECT X.JOBID, X.JOBDESCRIPTION
29   FROM (SELECT J.JOBID, J.JOBDESCRIPTION, J.POSTEDDATE,
         SYSDATE - J.POSTEDDATE AS DURATION
30             FROM JOB J) X
31  WHERE X.DURATION > 30;
32
```

```
33 --<Query 6>
34 SELECT P.Name, P.PERSONID
35   FROM Person P, Product Pr, Sale S
36  WHERE (Pr.LISTPRICE > 200 AND S.SALESMANID = P.PERSONID
       AND S.PRODUCTID = Pr.PRODUCTID);
37
38 --<Query 7>
39 SELECT DISTINCT D.DEPARTMENTID, D.DEPARTMENTNAME
40   FROM Department D, Job J
41  WHERE (J.DepartmentId = D.DepartmentId AND J.POSTEDDATE
       NOT BETWEEN TO_DATE('2011/01/01','yyyy/mm/dd') and
       TO_DATE('2011/02/01','yyyy/mm/dd'))
42
43  MINUS
44
45 SELECT DISTINCT D.DEPARTMENTID, D.DEPARTMENTNAME
46   FROM Department D, Job J
47  WHERE (J.DepartmentId = D.DepartmentId AND J.POSTEDDATE
       BETWEEN TO_DATE('2011/01/01','yyyy/mm/dd') and TO_DATE
       ('2011/02/01','yyyy/mm/dd'));
48
49 --<Query 8>
50 SELECT DISTINCT P.PersonId, P.Name, D.DEPARTMENTNAME
51   FROM Department D, Employee E, Interview I, Job J,
         Person P
52  WHERE (I.APPLICANTID = E.EmployeeId AND E.EmployeeId = P.
       PersonId AND I.JobId = '12345' AND I.JobId = J.JobId
       AND J.DepartmentId = D.DepartmentId);
53
54 --<Query 9>
55 SELECT MAX(COUNT(P.ProductType)) FROM SALE S, PRODUCT P
     WHERE S.PRODUCTID = P.PRODUCTID GROUP BY P.PRODUCTTYPE;
56
57 --<Query 10>
58 SELECT X.PRODUCTTYPE
59   FROM (SELECT X.PRODUCTTYPE, AVG(X.LISTPRICE - X.PTOT) AS
         PTOT
60             FROM (SELECT DISTINCT P.PRODUCTID, PRODUCT.
                 PRODUCTTYPE, PRODUCT.LISTPRICE, SUM(PART.
                 PRICE * P.QUANTITY) AS PTOT
61                         FROM PART PART, PRODUCT PRODUCT,
                             PRODUCT_PARTS P
62            WHERE (P.PARTID = PART.PARTID AND P.PRODUCTID =
                 PRODUCT.PRODUCTID) group by P.PRODUCTID,
                 PRODUCT.PRODUCTTYPE, PRODUCT.LISTPRICE) X GROUP
```

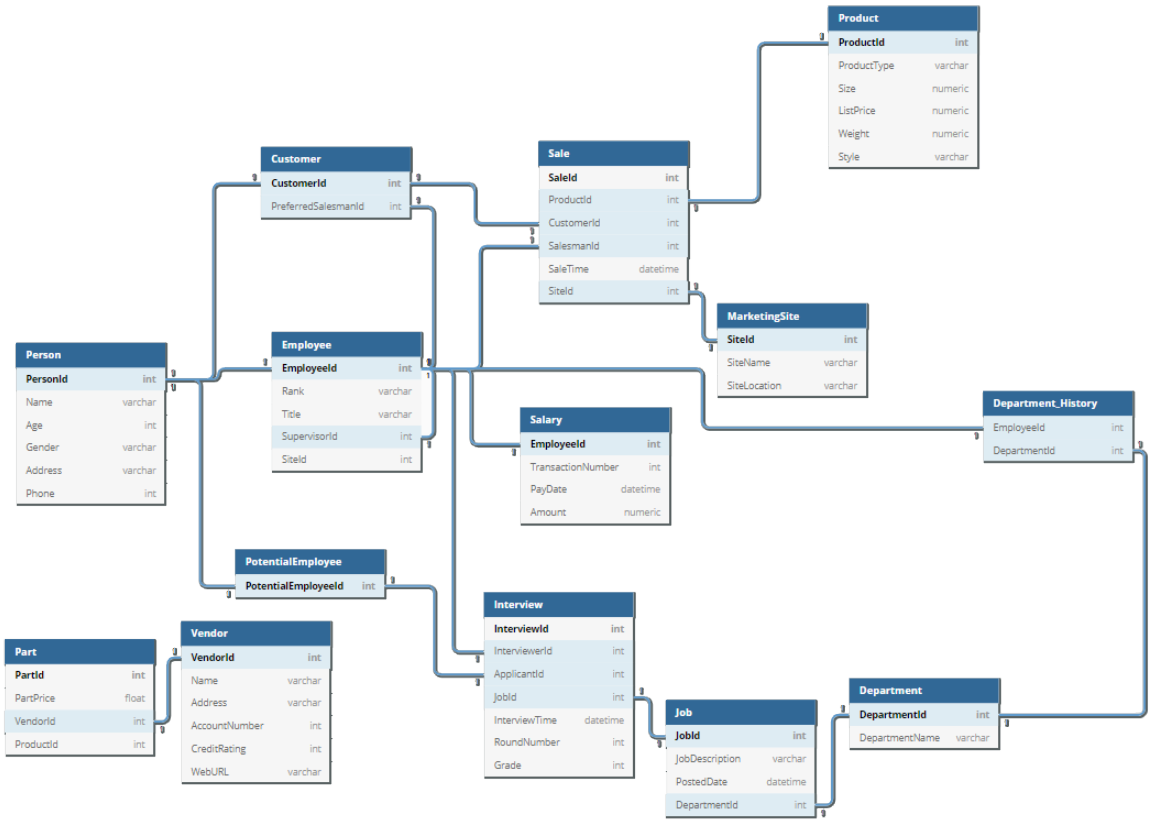
```
                BY X.PRODUCTTYPE) X
63  WHERE X.PTOT = (SELECT MAX(AVG(X.LISTPRICE - X.PTOT))
64                      FROM (SELECT DISTINCT P.PRODUCTID,
                            PRODUCT.PRODUCTTYPE, PRODUCT.
                            LISTPRICE, SUM(PART.PRICE * P.
                            QUANTITY) AS PTOT
65                              FROM PART PART, PRODUCT PRODUCT
                                , PRODUCT_PARTS P
66                              WHERE (P.PARTID = PART.PARTID
                                AND P.PRODUCTID = PRODUCT.
                                PRODUCTID) group by P.
                                PRODUCTID, PRODUCT.
                                PRODUCTTYPE, PRODUCT.
                                LISTPRICE) X GROUP BY X.
                                PRODUCTTYPE);
67
68  --SELECT MAX(AVG(X.LISTPRICE - X.PTOT)) FROM
69  --(SELECT DISTINCT P.PRODUCTID, PRODUCT.PRODUCTTYPE,
      PRODUCT.LISTPRICE, SUM(PART.PRICE * P.QUANTITY) AS PTOT
70  --  FROM PART PART, PRODUCT PRODUCT, PRODUCT_PARTS P
71  -- WHERE (P.PARTID = PART.PARTID AND P.PRODUCTID = PRODUCT
      .PRODUCTID) group by P.PRODUCTID, PRODUCT.PRODUCTTYPE,
      PRODUCT.LISTPRICE) X GROUP BY X.PRODUCTTYPE;
72  --
73  --SELECT X.PRODUCTTYPE, AVG(X.LISTPRICE - X.PTOT) AS PTOT
74  --  FROM (SELECT DISTINCT P.PRODUCTID, PRODUCT.PRODUCTTYPE
      , PRODUCT.LISTPRICE, SUM(PART.PRICE * P.QUANTITY) AS
      PTOT
75  --            FROM PART PART, PRODUCT PRODUCT, PRODUCT_PARTS
       P
76  --          WHERE (P.PARTID = PART.PARTID AND P.PRODUCTID =
       PRODUCT.PRODUCTID) group by P.PRODUCTID, PRODUCT.
      PRODUCTTYPE, PRODUCT.LISTPRICE) X
77  -- GROUP BY X.PRODUCTTYPE;
78
79  --<Query 11 *Dis requires some more info, :thinking:>
80  SELECT T.Name, T.PersonId
81    FROM (SELECT P.Name, P.PersonId, COUNT(DH.DepartmentId)
        AS DEPT_COUNT
82              FROM Person P, Department_History DH
83            WHERE (P.PersonId = DH.EmployeeId) GROUP BY P.
                Name, P.PersonId) T,
84            (SELECT COUNT(D.DepartmentId) AS DEPT_MAX FROM
                Department D) X
85  WHERE (T.DEPT_COUNT = X.DEPT_MAX);
```

```
86
87 --<Query 12>
88 SELECT P.Name, P.Email
89   FROM PERSON P, (SELECT PERSONID, JOBID, SUM(GRADE) /
       COUNT(ROUNDNUMBER) AS AVERAGE_SCORE, COUNT(
       ROUNDNUMBER) AS ROUNDCOUNT
90                       FROM (SELECT DISTINCT P.PERSONID, IG.
                             ROUNDNUMBER, IG.GRADE, I.JobId
91                                   FROM JOB J, PERSON P,
                                     POTENTIALEMPLOYEE PE,
                                     INTERVIEW_GRADE IG,
                                     INTERVIEW I
92                                  WHERE (IG.INTERVIEWID = I.
                                    INTERVIEWID AND I.
                                    APPLICANTID = PE.
                                    POTENTIALEMPLOYEEID AND I.
                                    APPLICANTID = P.PERSONID))
                                    group by JOBID, PERSONID) X
93  WHERE X.ROUNDCOUNT >=5 AND X.AVERAGE_SCORE >= 70 AND X.
      PERSONID = P.PERSONID;
94
95 --<Query 13, Whats the difference between this one and
     12?>
96 -- Select people who got every job they applied for...
97 --SELECT PERSONID, JOBID, SUM(GRADE) / COUNT(ROUNDNUMBER)
     AS AVERAGE_SCORE, COUNT(ROUNDNUMBER) AS ROUNDCOUNT
98 --                       FROM (SELECT DISTINCT P.PERSONID,
     P.NAME, P.EMAIL, IG.ROUNDNUMBER, IG.GRADE, I.JobId
99 --                             FROM JOB J, PERSON P,
     POTENTIALEMPLOYEE PE, INTERVIEW_GRADE IG, INTERVIEW I
100 --                            WHERE (IG.INTERVIEWID = I.
     INTERVIEWID AND I.APPLICANTID = PE.POTENTIALEMPLOYEEID
     AND I.APPLICANTID = P.PERSONID)) group by JOBID,
     PERSONID;
101 --
102 --SELECT B.PERSONID, COUNT(B.JOBID) FROM (SELECT DISTINCT
     P.PERSONID, I.JobId
103 --                       FROM PERSON P, POTENTIALEMPLOYEE
     PE, INTERVIEW_GRADE IG, INTERVIEW I
104 --                       WHERE (IG.INTERVIEWID = I.
     INTERVIEWID AND I.APPLICANTID = PE.POTENTIALEMPLOYEEID
     AND I.APPLICANTID = P.PERSONID)) B group by B.PERSONID;
105
106 SELECT G.NAME, G.Phone, G.Email
107   FROM (SELECT P.Name, P.Phone, P.Email, P.PersonId, COUNT
```

```
        (X.JOBID) AS JOB_COUNT
108     FROM PERSON P, (SELECT PERSONID, JOBID, SUM(GRADE) /
          COUNT(ROUNDNUMBER) AS AVERAGE_SCORE, COUNT(
          ROUNDNUMBER) AS ROUNDCOUNT
109                     FROM (SELECT DISTINCT P.PERSONID, P.
                          NAME, P.EMAIL, IG.ROUNDNUMBER, IG
                          .GRADE, I.JobId
110                       FROM JOB J, PERSON P,
                            POTENTIALEMPLOYEE PE,
                            INTERVIEW_GRADE IG,
                            INTERVIEW I
111                       WHERE (IG.INTERVIEWID = I.
                            INTERVIEWID AND I.
                            APPLICANTID = PE.
                            POTENTIALEMPLOYEEID AND I.
                            APPLICANTID = P.PERSONID))
                             group by JOBID, PERSONID)
                              X
112   WHERE (X.ROUNDCOUNT >=5 AND X.AVERAGE_SCORE >= 70 AND X
        .PERSONID = P.PERSONID) GROUP BY P.Name, P.Email, P.
        PersonId, P.Phone) G,
113                 (SELECT B.PERSONID, COUNT(B.JOBID) AS
                      JOB_COUNT FROM (SELECT DISTINCT P.
                      PERSONID, I.JobId
114                   FROM PERSON P, POTENTIALEMPLOYEE PE,
                          INTERVIEW_GRADE IG, INTERVIEW I
115                   WHERE (IG.INTERVIEWID = I.INTERVIEWID
                          AND I.APPLICANTID = PE.
                          POTENTIALEMPLOYEEID AND I.
                          APPLICANTID = P.PERSONID)) B group
                           by B.PERSONID) X
116  WHERE (X.PERSONID = G.PERSONID AND G.JOB_COUNT = X.
       JOB_COUNT);
117
118 --<Query 14>
119 SELECT X.PERSONID, X.NAME
120   FROM (SELECT P.PERSONID, P.NAME, AVG(S.AMOUNT) AS
        AVERAGE FROM PERSON P, EMPLOYEE E, SALARY S WHERE (P.
        PERSONID = E.EMPLOYEEID AND S.EMPLOYEEID = E.
        EMPLOYEEID) GROUP BY P.PERSONID, P.NAME) X
121  WHERE (X.AVERAGE = (SELECT MAX(AVG(S.AMOUNT)) FROM SALARY
        S GROUP BY S.EMPLOYEEID));
122 --SELECT P.PERSONID, P.NAME, AVG(S.AMOUNT) AS AVERAGE FROM
      PERSON P, EMPLOYEE E, SALARY S WHERE (P.PERSONID = E.
      EMPLOYEEID AND S.EMPLOYEEID = E.EMPLOYEEID) GROUP BY P.
```

```
        PERSONID, P.NAME;
123
124 --<Query 15>
125 SELECT V.VENDORID, V.NAME
126   FROM PART P, VENDOR V
127  WHERE P.PARTNAME = 'Cup' AND V.VENDORID = P.VENDORID AND
          P.PRICE = (SELECT MIN(P.PRICE) FROM PART P WHERE P.
          PARTNAME = 'Cup');
```

5. Dependency Diagram:



6. Database States:

Customer:

	CUSTOMERID	PREFERREDSALESMAN
1	4001	2049

Department:

	DEPARTMENTID	DEPARTMENTNAME
1	3	Mathematics
2	12	Marketing

Department History:

	DEPARTMENTID	EMPLOYEEID
1	3	2048
2	12	2048
3	3	2049

Employee:

	EMPLOYEEID	RANK	TITLE	SUPERVISORID	SITEID
1	2048	Lead Director	Vice President	(null)	1
2	2049	Director	UX Designer	2048	1
3	2050	Salesman	Sr.	2048	2

Interview:

	INTERVIEWID	INTERVIEWERID	APPLICANTID	JOBID
1	10	2048	1048	3
2	11	2048	1049	3
3	1	2048	1001	11111
4	2	2049	1001	11111
5	81	2048	1049	12345
6	82	2048	1048	12345
7	83	2048	2049	12345

Interview Grade:

	INTERVIEWID	ROUNDNUMBER	GRADE
1	82	2	70
2	82	3	50
3	82	4	80
4	82	1	60
5	82	5	90
6	10	1	60
7	10	2	70
8	10	3	50
9	10	4	80
10	10	5	90
11	11	1	30
12	11	2	20
13	11	3	50
14	11	4	60
15	11	5	70

Job:

	JOBID	JOBDESCRIPTION	POSTEDDATE	DEPARTMENTID
1	3	Math Research Intern	03-MAY-18	3
2	11111	Summer Computer Science Intern	01-FEB-18	3
3	86	Marketing Intern, Fall	03-MAR-11	12
4	87	Marketing Intern, Summer	05-MAR-11	12
5	88	Marketing Intern, Spring	02-MAR-11	12
6	89	Marketing Intern, Winter	09-MAR-11	12
7	90	On-Site Sales Intern	09-APR-12	12
8	101	Receptionist Intern	06-JAN-11	12
9	12345	Busniess Insider Intern	06-JAN-11	12

MarketingSite:

	SITEID	SITENAME	SITELOCATION
1	1	Mathematics Research	San Jose
2	2	Omega	Houston

Part:

	PARTID	PARTNAME	PRICE	VENDORID
1	1	Handle	3.99	1
2	2	Metal Bob	1.99	2
3	3	Quantum Parts	6.99	2
4	4	Metallic Plating	0.99	3
5	5	High-Quality Paper	4.99	2
6	6	Cardboard Cover	0.99	3
7	15	Cup	2.99	2
8	17	Cup	1.99	3

Person:

	PERSONID	NAME	AGE	GENDER	ADDRESS	PHONE	EMAIL
1	2048	Matthew McMillian	20	Male	UT Dallas	3335557777	testemail@ut.edu
2	2049	Brandon Tran	21	Male	UT Dallas	4448889999	testemail@ut.edu
3	1048	Seong Wang	20	Male	UT Dallas	1112223333	testemail@ut.edu
4	1049	Kion Smith	20	Male	UT Dallas	3454563456	testemail@ut.edu
5	1001	Hellen Cole	30	Female	999 New York City	2819995555	testemail@ut.edu
6	2050	Hector Quear	27	Male	Austin	3335557777	testemail@ut.edu
7	4001	John Bimbo	42	Male	999 Los Muertos	6775663444	testemail@ut.edu

Potential Employee:

	POTENTIALEMPLOYEEID
1	1048
2	1049
3	1001

Product:

	PRODUCTID	PRODUCTTYPE	PRODUCTSIZE	LISTPRICE	WEIGHT	PRODUCTSTYLE
1	1	Crowbar	20	19.99	5.31	Red-Metal
2	3	Crowbar	20	19.99	5.31	Blue-Metal
3	4	Crowbar	20	21.99	5.31	Yellow-Metal
4	2	Portal Gun	23	119.99	10.31	Aperture White
5	6	Textbook	15	120.99	2.59	Statistical Analysis
6	7	Textbook	15	220.99	2.59	Numerical Analysis
7	8	Textbook	15	260.99	2.59	Database Analysis

Product Parts:

	PRODUCTID	QUANTITY	PARTID
1	1	1	1
2	1	1	2
3	4	1	1
4	4	1	2
5	3	1	1
6	3	1	2
7	2	3	3
8	2	4	4
9	6	1	5
10	6	50	6
11	7	1	5
12	7	50	6
13	8	1	5
14	8	50	6

Salary:

	EMPLOYEEID	TRANSACTIONNUMBER	PAYDATE	AMOUNT
1	2048	1000	01-JUL-18	10
2	2048	1001	02-JUL-18	40
3	2049	1000	01-JUL-18	30
4	2049	1001	02-JUL-18	80
5	2048	1002	02-JUL-18	240

Sale:

	SALEID	PRODUCTID	CUSTOMERID	SALESMANID	SALETIME
1	301	1	4001	2049	03-APR-11
2	302	1	4001	2049	01-MAR-11
3	311	3	4001	2050	03-APR-11
4	312	2	4001	2050	01-FEB-11
5	612	8	4001	2050	05-FEB-11
6	613	7	4001	2050	06-FEB-11

Vendor:

	VENDORID	NAME	ADDRESS	ACCOUNTNUMBER	CREDITRATING	WEBURL
1	2	Jacks Parts	500 W Parker St	999888777	200	www.test.com
2	3	Bills Parts	300 W Parker St	124432134	210	www.test.com

View 1:

	AVERAGE_SALARY	EMPLOYEEID
1	96.666666666666666666666666666666666666667	2048
2	55	2049

View 2:

	PASSED_INTERVIEW_ROUNDS	JOBID	APPLICANTID
1	3	3	1048
2	1	3	1049
3	3	12345	1048

View 3:

	NUMBER_OF_ITEMS	PRODUCTTYPE
1	3	Textbook
2	3	Crowbar
3	1	Portal Gun

View 4:

	PART_PURCHASE_COST	PRODUCT
1	54.49	Textbook
2	5.98	Crowbar
3	24.93	Portal Gun