GRIFFITH COLLEGE DUBLIN

Assignment Cover Sheet

Student name:	**Yen Lung Chen**		
Student number:	**2971978**		
Faculty:	**Computing Science**		
Course:	**HDC Computer Science**	**Stage/year:**	**1**
Subject:	**Relational Database**		
Study Mode:	Full time **V**	Part-time	
Lecturer Name:	**Gemma Deery**		
Assignment Title:	**RA/Mapping & Norm/SQL**		
No. of pages:	**15**		
Disk included?	Yes	No	
Additional Information:	(ie. number of pieces submitted, size of assignment, A2, A3 etc)		
Date due:	**3rd May 2018**		
Date submitted:	**3rd May 2018**		

Plagiarism disclaimer:

I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.

I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.

Signed: Yen Lung Chen **Date: 26/04/2018**

Please note: *Students **MUST** retain a hard / soft copy of **ALL** assignments as well as a receipt issued and signed by a member of Faculty as proof of submission.*

Consider the following schema:
Supplier(sid: integer, sname: string, address: string)
Part(pid: integer, pname: string, colour: string)
Catalog(sid: integer, pid: integer, cost: real)

The relation Supplier stores suppliers and the key of that relation is sid. The relation Part stores parts, and pid is the key of that relation. Finally, Catalogue stores which supplier supplies which part at which cost. The key is the combination of the two attributes sid and pid.

1. Write queries in relational algebra 3%

Supplier
sid
sname
address

Catalog
sid
pid
cost

Part
pid
pname
colour

Write the following queries in relational algebra.

1. Find the names of suppliers who supply some red part.
   Answer:
   $\prod_{sname}(\sigma_{colour="red"}(Supplier \bowtie Part \bowtie Catalog))$
   Duplicate rows are automatically eliminated, as relation is a set.
   Or the queries in the relational algebra could be
   $\prod_{sname}(\prod_{sid}(((\sigma_{colour="red"} Part) \bowtie Catalog) \bowtie Supplier))$

2. Find the IDs of suppliers who supply some red or green part.

$(\sigma_{colour="red" v colour="green"}Part)$
Pid
Pname
Colour

$((\sigma_{colour="red" v colour="green"}Part) \bowtie Catalog)$
Pid
sid
Pname
Colour
Cost

Catalog
sid
pid
cost

Answer: $\prod_{sid}((\sigma_{colour="red" v colour="green"}Part) \bowtie Catalog)$

3. Find the IDs of suppliers who supply some red part or are based at 21 George Street.

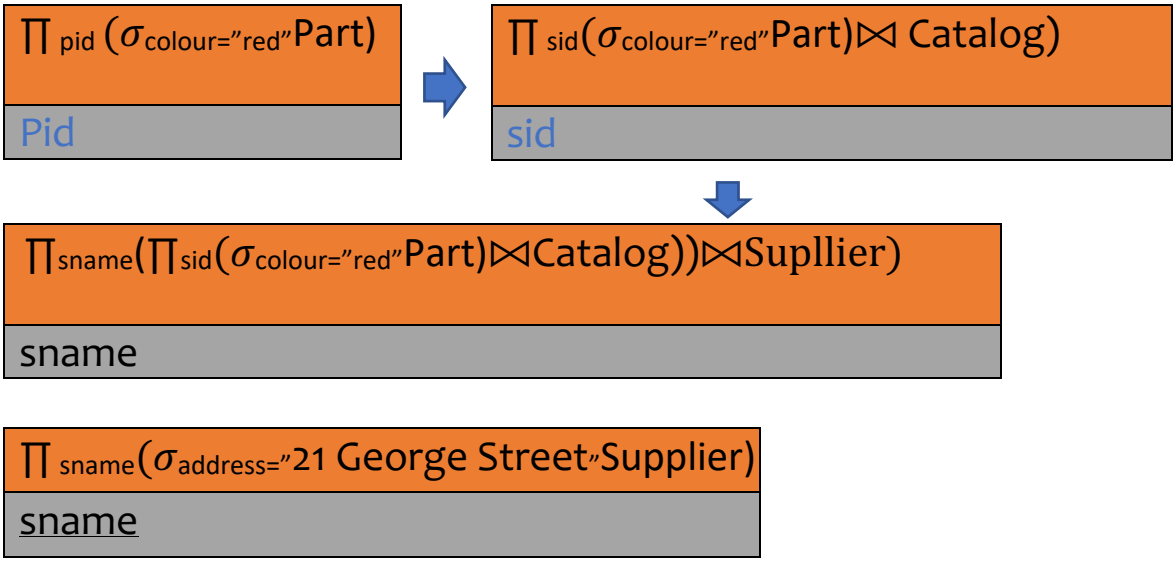
$\prod_{pid}(\sigma_{colour="red"}Part)$
Pid

$\prod_{sid}(\prod_{pid}(\sigma_{colour="red"}Part) \bowtie Catalog)$
sid

$\prod_{sid}(\sigma_{address="21 George Street"}Supplier)$
sid

Answer:
$\prod_{sid}(\prod_{pid}(\sigma_{colour="red"}Part) \bowtie Catalog) U (\prod_{sid}(\sigma_{address="21 George Street"}Supplier))$

4. Find the names of suppliers who supply some red part or are based at 21 George Street.

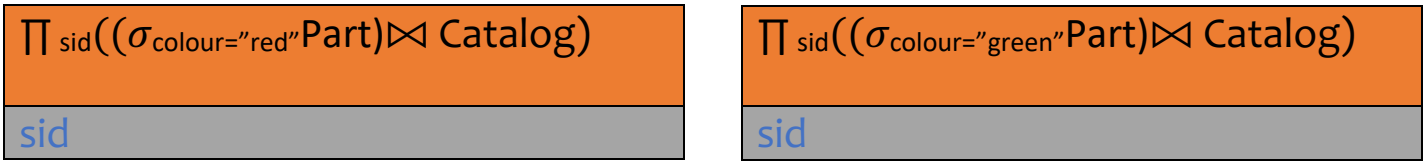


Answer:

$\prod_{sname}(\prod_{sid}(\sigma_{colour="red"}Part)\bowtie Catalog))\bowtie Supllier) \cup \prod_{sname}(\sigma_{address="21George\ Street"}Supplier)$

5. Find the IDs of suppliers who supply some red part and some green part.

(Hint: use intersection of relations or join the same relation several times)



Answer:

$\prod_{sid}((\sigma_{colour="red"Part})\bowtie Catalog) \cap \prod_{sid}((\sigma_{colour="green"Part})\bowtie Catalog)$

6. Find pairs of IDs such that the supplier with the first ID charges more for some part than the supplier with the secondID. (Hint: you may want to create temporary relations to get two copies of Catalog)

    1) Create two copies of Catalog and use Cartesian Product to compare each row of first table to each row of table 2 (Catalog 1 X Catalog 2)
    2) Only compare the same product in each tuple and different sid ($\sigma_{Catalog1.pid=Catalog2.pid \wedge Catalog1.sid!=Catalog2.sid}$(Catalog 1 X Catalog 2))
    3) Compare cost ($\sigma_{Catalog1.pid=Catalog2.pid \wedge Catalog1.sid!=Catalog2.sid \wedge Catalog1.cost>Catalog2.cost}$(Catalog 1 X Catalog 2))
    4) Project the sid
    5) Answer: $\prod_{Catalog1.sid,\ Catalog2.sid}(\sigma_{Catalog1.pid=Catalog2.pid \wedge Catalog1.sid!=Catalog2.sid \wedge Catalog1.cost>Catalog2.cost}$(Catalog 1 X Catalog 2))

7. Find the IDs of suppliers who supply only red parts. (Hint: A supplier supplies only red parts if it is not the case that the supplier offers a part that is not red. This question is a challenge!)

    1) Find the sid supply part which is not red : $\prod_{sid}((\sigma_{colour!=\text{"red"}}Part) \bowtie Catalog)$
    2) A supplier supplies only red parts if it is not the case that the supplier offers a part that is not red.
    3) Answer: $\prod_{sid}Supplier - \prod_{sid}((\sigma_{colour!=\text{"red"}}Part) \bowtie Catalog)$

8. Find the IDs of suppliers who supply every part. (Hint: A supplier supplies every part if it is not the case that there is some part which they do not supply. Use set difference and cartesian product. This question is a challenge, too!)

    1) Create a relation indicate each supplier ID supply all the part:
       $\prod_{sid,}$Supplier X $\prod_{pid}$Part
    2) Use division operator to find who actually supply every part:
       $\prod_{sid, pid}$(Catalog) / $\prod_{sid,}$Supplier X $\prod_{pid}$Part
    3) Find IDs of suppliers who supply every part:
       Answer:
       $\prod_{sid}$($\prod_{sid, pid}$(Catalog) /$\prod_{sid, pid}$(Supplier X Part))

2. Queries in relational algebra, what do they mean? 3%

For each of the following relational algebra queries, say in English what they mean.

1. πsname(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier)

Answer: Find the name of supplier who supply some red part less than cost 100

2. πsname(πsid(σcolour=”red” (Part) x σcost<100 (Catalog)) x Supplier)

Answer: Find the name of supplier who supply some red part less than cost 100

3. πsname(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier) ∩ πsname(σcolour=”green” (Part) x σcost<100 (Catalog) x Supplier)

πsname(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier):
set of name in suppliers supplying some red part less than cost 100

πsname(σcolour=”green” (Part) x σcost<100 (Catalog) x Supplier):
set of name in suppliers supplying some green part less than cost 100

Answer: Find the name of supplier who supply some red part and the supplier who supply green part less than cost 100

Note: two distinct suppliers with the same name may be returned by this query even if the first doesn’t supply a green part, and the second doesn’t supply a red part.

4. πsid(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier)
∩πsid(σcolour=”green” (Part) x σcost<100 (Catalog) x Supplier)

Answer: Find the ID of supplier who supply some red part and green part both less than cost 100

5. πsname(πsid,sname(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier) ∩ πsid,sname(σcolour=”green” (Part) x σcost<100 (Catalog) x Supplier))

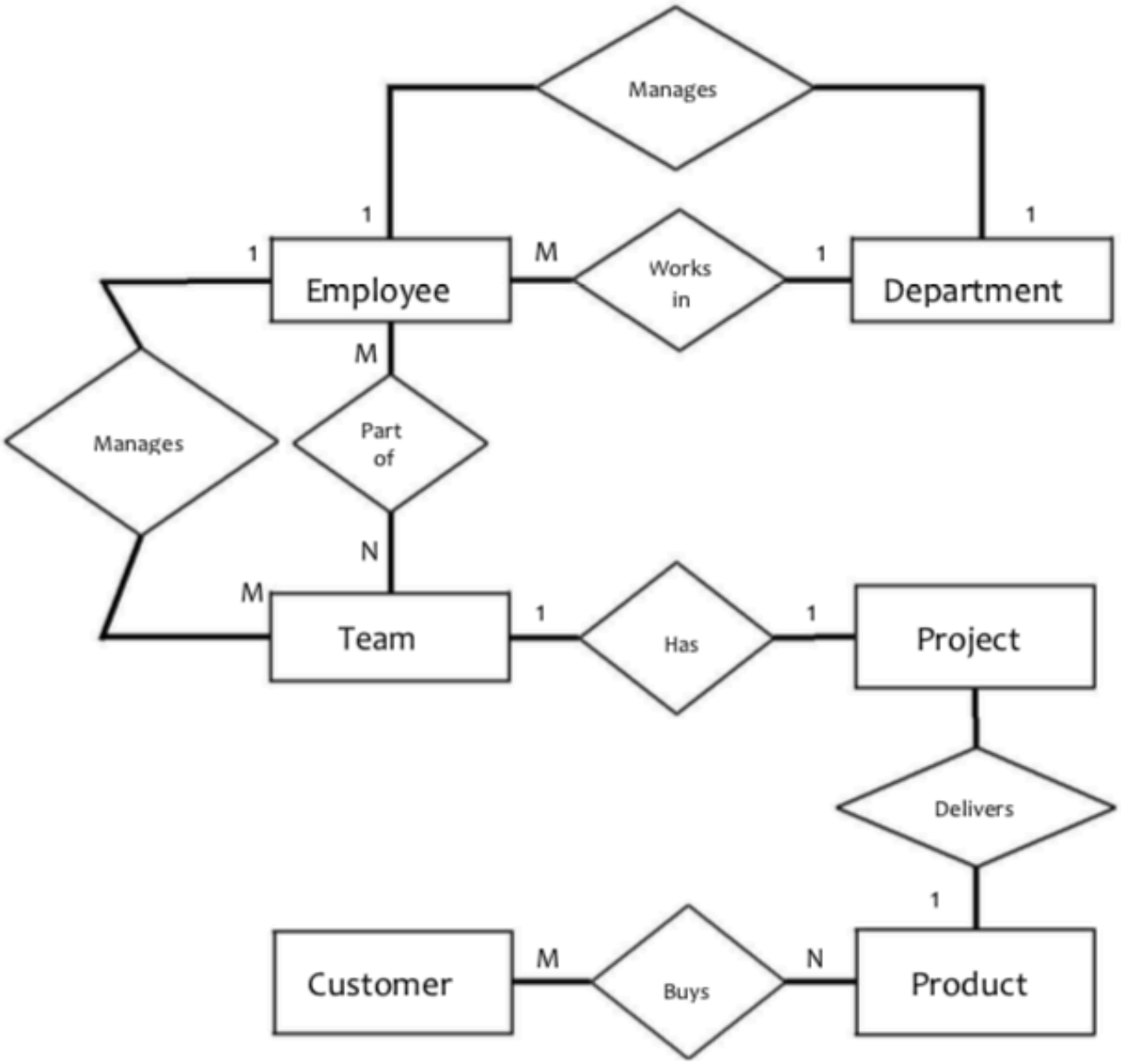
πsname(πsid,sname(σcolour=”red” (Part) x σcost<100 (Catalog) x Supplier):
Project the name of suppliers from the table contains two attributes: sid and sname, indicate from which red part less than cost 100 supplied from.

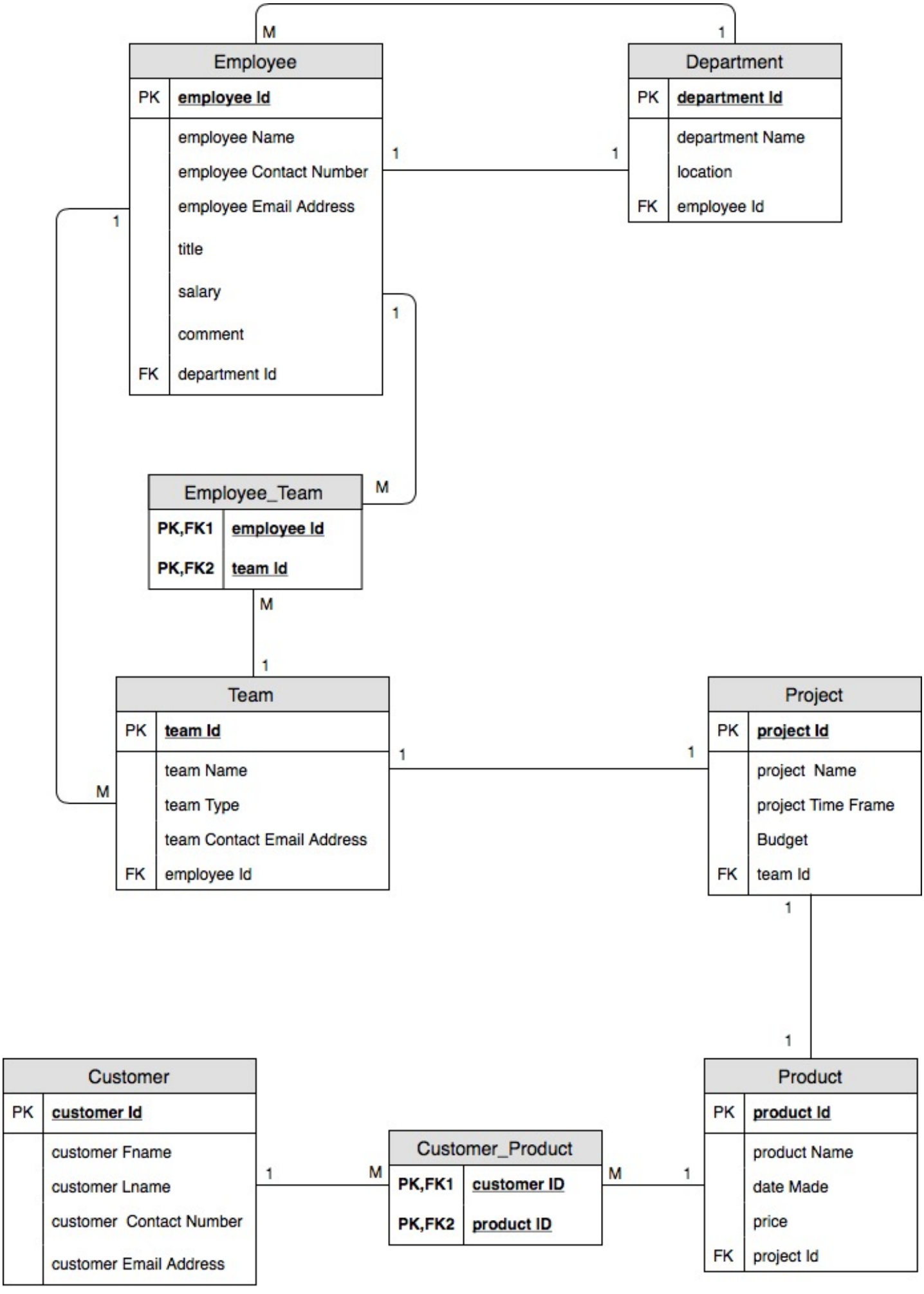
πsid,sname(σcolour=”green” (Part) x σcost<100 (Catalog) x Supplier)): Project the name and ID of suppliers from the joined table indicates from which green part less than cost 100 supplied from.

Answer:
Find the name of suppliers who supply some red part and some green part for less than 100.

3. Mapping & Normalisation 9%

a) Mapping the following Entity Relationship Diagram into relational schema. Choose appropriate attributes for each entity type. 5%

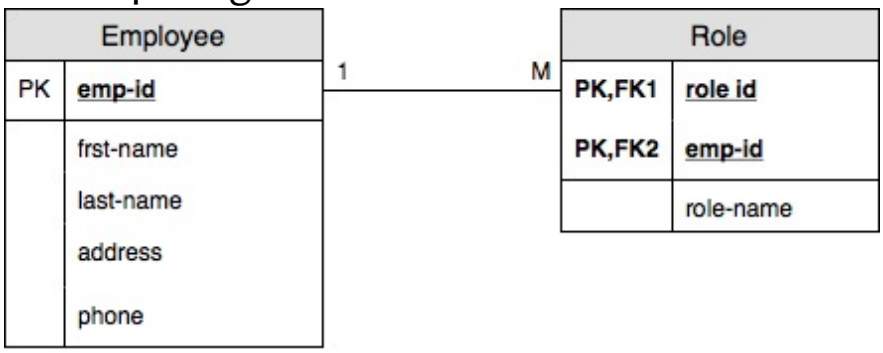



M
1
Employee
PK
employee Id
employee Name
employee Contact Number
employee Email Address
title
salary
comment
FK
department Id
Department
PK
department Id
department Name
location
FK
employee Id
1
1
1
1
1
M
Employee_Team
PK,FK1
employee Id
PK,FK2
team Id
M
1
Team
PK
team Id
team Name
team Type
team Contact Email Address
FK
employee Id
M
1
1
Project
PK
project Id
project Name
project Time Frame
Budget
FK
team Id
1
1
Customer
PK
customer Id
customer Fname
customer Lname
customer Contact Number
customer Email Address
1
M
Customer_Product
PK,FK1
customer ID
PK,FK2
product ID
M
1
Product
PK
product Id
product Name
date Made
price
FK
project Id

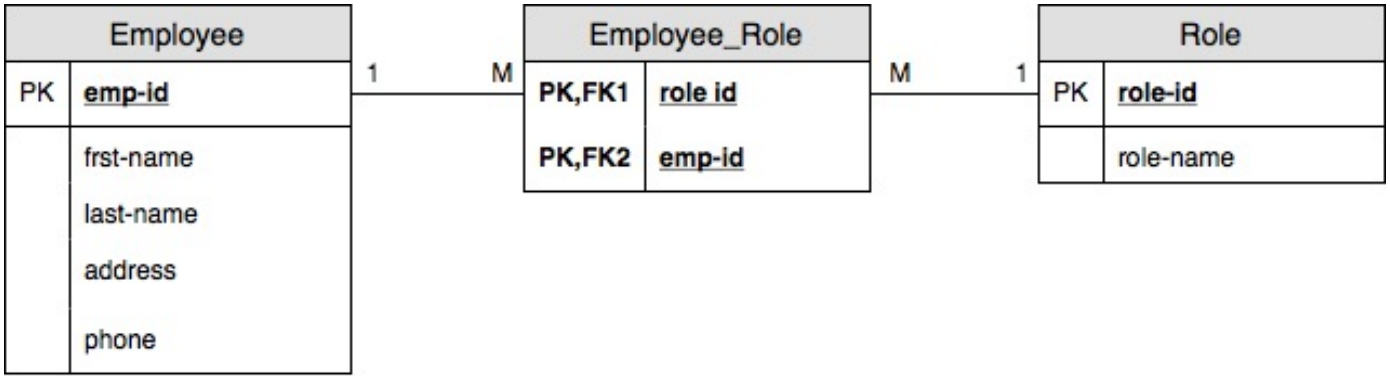
b) Database normalization – consider the following relation: 5%
Employee (emp-id, frst-name, last-name, address, phone, role-id, role-name)
Let's assume each employee has one address and one phone number but each employee within the company is given more than one roles.
For example an employee may take responsibility of providing telephone support and face to face support.

• State whether the given table structure is in 1NF. If not, make appropriate changes so that it is in 1NF.
No, the given table structure is not in 1NF. A relation is in 1NF if, and only if, it contains no repeating attributes or groups of attributes. In other words each of the column in a table should contain one value and just one value.

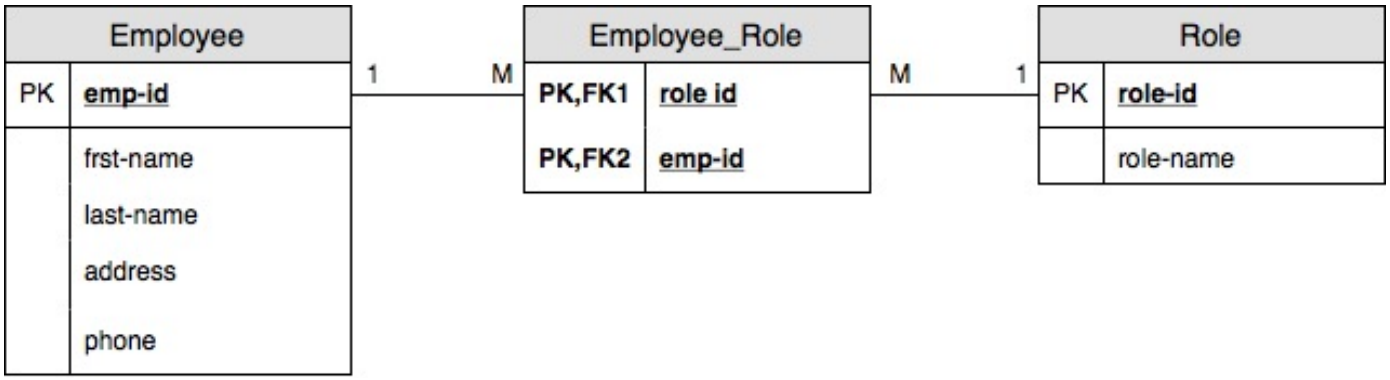
Decomposing:



• State whether the given table structure is in 2NF. If not, make appropriate changes so that it is in 2NF.
No, the given table structure is not in 2NF. No non-key fields may depend on a portion of the primary key in 2NF. Check all of the non-key attributes against each part of the key attributes to ensure they are functionally dependent on it.

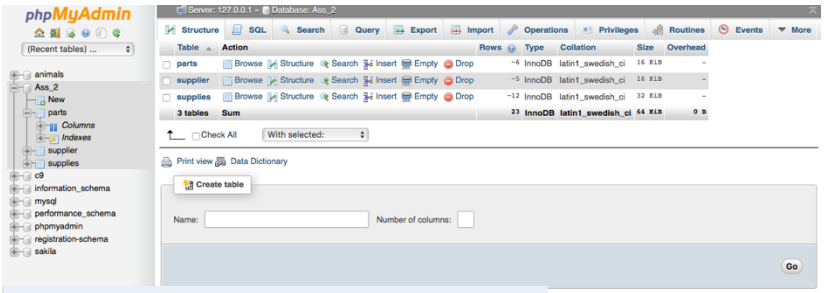


• State whether the given table structure is in 3NF. If not, make appropriate changes so that it is in 3NF.
No, the given table structure is not in 3NF.
Third Normal From (3FN): No fields may depend on other non- key fields. In other words, each field in a record should contain information about the entity that is defined by the primary key.



4. SQL 5%

Run the SQL – DDL and DML statements provided in assignment2__DDL_DML.sql file (on Moodle). This will create 3 tables and populate them with data.
As the file is in .sql format, to save time you can copy and paste when executing these statements.
a) Read the statements when you execute them so you understand the tables, columns, and their relationships. Once you have run all the statements successfully, now create and execute the following queries:




```
mysql> use Ass_2;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| Ass_2              |
| animals            |
| c9                 |
| mysql              |
| performance_schema |
| phpmyadmin         |
| registration-schema |
| sakila             |
+--------------------+
9 rows in set (0.00 sec)

mysql> use Ass_2;
Database changed
mysql>
```

b) How many relationships exist between these tables? Specify their type & cardinality.


```
mysql> describe parts;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| partNum | char(2)      | NO   | PRI | NULL    |       |
| name    | char(10)     | NO   |     | NULL    |       |
| colour  | char(8)      | NO   |     | NULL    |       |
| weight  | decimal(3,1) | NO   |     | NULL    |       |
| city    | varchar(10)  | NO   |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> describe supplier;
+-------------+------------+------+-----+---------+-------+
| Field       | Type       | Null | Key | Default | Extra |
+-------------+------------+------+-----+---------+-------+
| supplierNum | char(2)    | NO   | PRI | NULL    |       |
| name        | char(10)   | NO   |     | NULL    |       |
| status      | tinyint(4) | NO   |     | NULL    |       |
| city        | varchar(10)| NO   |     | NULL    |       |
+-------------+------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> describe  supplies;
+-------------+------------+------+-----+---------+-------+
| Field       | Type       | Null | Key | Default | Extra |
+-------------+------------+------+-----+---------+-------+
| supplierNum | char(2)    | NO   | PRI | NULL    |       |
| partNum     | char(2)    | NO   | PRI | NULL    |       |
| quantity    | smallint(6)| NO   |     | NULL    |       |
+-------------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

c) List all the records in supplier, parts, & supplies tables. One table at a time.

```
mysql> select* from parts;
+---------+-------+--------+--------+--------+
| partNum | name  | colour | weight | city   |
+---------+-------+--------+--------+--------+
| P1      | Nut   | Red    |   12.0 | London |
| P2      | Bolt  | Green  |   17.0 | Paris  |
| P3      | Screw | Blue   |   17.0 | Oslo   |
| P4      | Screw | Red    |   14.0 | London |
| P5      | Cam   | Blue   |   12.0 | Paris  |
| P6      | Cog   | Red    |   19.0 | London |
+---------+-------+--------+--------+--------+
6 rows in set (0.00 sec)
```

```
mysql> select* from supplier;
+-------------+-------+--------+--------+
| supplierNum | name  | status | city   |
+-------------+-------+--------+--------+
| S1          | Smith |     20 | London |
| S2          | Jones |     10 | Paris  |
| S3          | Blake |     20 | Paris  |
| S4          | Clark |     20 | London |
| S5          | Adams |     30 | Athens |
+-------------+-------+--------+--------+
5 rows in set (0.00 sec)
```

```
mysql> select *from supplies;
+-------------+---------+----------+
| supplierNum | partNum | quantity |
+-------------+---------+----------+
| S1          | P1      |      300 |
| S1          | P2      |      200 |
| S1          | P3      |      400 |
| S1          | P4      |      200 |
| S1          | P5      |      100 |
| S1          | P6      |      100 |
| S2          | P1      |      300 |
| S2          | P2      |      400 |
| S3          | P2      |      200 |
| S4          | P2      |      200 |
| S4          | P4      |      300 |
| S4          | P5      |      400 |
+-------------+---------+----------+
12 rows in set (0.01 sec)
```

d) Update the supplier table to reflect a change in supplier's status to the value 20 for all suppliers from Paris.

```
mysql> update supplier set status=20 where city="Paris";
Query OK, 1 row affected (0.01 sec)
Rows matched: 2  Changed: 1  Warnings: 0

mysql> select * from supplier;
+-------------+-------+--------+--------+
| supplierNum | name  | status | city   |
+-------------+-------+--------+--------+
| S1          | Smith |     20 | London |
| S2          | Jones |     20 | Paris  |
| S3          | Blake |     20 | Paris  |
| S4          | Clark |     20 | London |
| S5          | Adams |     30 | Athens |
+-------------+-------+--------+--------+
5 rows in set (0.01 sec)
```

e) Show the number of suppliers in each city ordered from highest to lowest.

```
mysql> select city, count(city) from supplier group by city order by count(city) desc;
+--------+-------------+
| city   | count(city) |
+--------+-------------+
| Paris  |           2 |
| London |           2 |
| Athens |           1 |
+--------+-------------+
3 rows in set (0.00 sec)
```

f) List only the name of all the parts except the Red parts.

```
mysql> select name from parts
    -> where colour !="Red";
+-------+
| name  |
+-------+
| Bolt  |
| Screw |
| Cam   |
+-------+
3 rows in set (0.00 sec)
```

g) Show all entries from the supplies table with their corresponding part names and supplier names. Rename the columns to appropriate ones.

```
mysql> select supplier.name Sname, supplies.supplierNum,supplies.partNum,parts.name Pname, quantity from supplier join supplies on suppli
er.supplierNum=supplies.supplierNum join parts on supplies.partNum=parts.partNum;
+-------+-------------+---------+-------+----------+
| Sname | supplierNum | partNum | Pname | quantity |
+-------+-------------+---------+-------+----------+
| Smith | S1          | P1      | Nut   |      300 |
| Smith | S1          | P2      | Bolt  |      200 |
| Smith | S1          | P3      | Screw |      400 |
| Smith | S1          | P4      | Screw |      200 |
| Smith | S1          | P5      | Cam   |      100 |
| Smith | S1          | P6      | Cog   |      100 |
| Jones | S2          | P1      | Nut   |      300 |
| Jones | S2          | P2      | Bolt  |      400 |
| Blake | S3          | P2      | Bolt  |      200 |
| Clark | S4          | P2      | Bolt  |      200 |
| Clark | S4          | P4      | Screw |      300 |
| Clark | S4          | P5      | Cam   |      400 |
+-------+-------------+---------+-------+----------+
12 rows in set (0.00 sec)
```

h) Show the names of all suppliers that appear more than once in the supplies table.

```
mysql> select name from supplier join supplies on supplier.supplierNum=supplies.supplierNum group by supplies.supplierNum having count(su
pplies.supplierNum)>1;
+-------+
| name  |
+-------+
| Smith |
| Jones |
| Clark |
+-------+
3 rows in set (0.00 sec)
```

i) Supplier with supplierNum = S4 has closed down his business. Delete all the records related to this supplier from all relevant tables.

Note: MySQL provides a more effective way called `ON DELETE CASCADE` referential action for a foreign key that allows you to delete data from child tables automatically when you delete the data from the parent table. For this assignment the original database file is altered in .txt and imported to myphpadmin. In addition, the action of this query will affect the following two queries due to deletion of supplierNum = S4.

Shows the CREATE TABLE statement that creates the named table to make sure `ON DELETE CASCADE` clause at the end of the foreign key constraint definition.

```
-------------------------------------------------+
| supplies | CREATE TABLE `supplies` (
  `supplierNum` char(2) NOT NULL,
  `partNum` char(2) NOT NULL,
  `quantity` smallint(6) NOT NULL,
  PRIMARY KEY (`supplierNum`,`partNum`),
  KEY `partNum` (`partNum`),
  CONSTRAINT `supplies_ibfk_1` FOREIGN KEY (`supplierNum`) REFERENCES `supplier` (`supplierNum`) ON DELETE CASCADE,
  CONSTRAINT `supplies_ibfk_2` FOREIGN KEY (`partNum`) REFERENCES `parts` (`partNum`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
+----------+----------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------
```

Delete:

```
mysql> delete from supplier where supplierNum ="S4";
Query OK, 1 row affected (0.00 sec)

mysql> select* from supplier;
+-------------+-------+--------+--------+
| supplierNum | name  | status | city   |
+-------------+-------+--------+--------+
| S1          | Smith |     20 | London |
| S2          | Jones |     10 | Paris  |
| S3          | Blake |     20 | Paris  |
| S5          | Adams |     30 | Athens |
+-------------+-------+--------+--------+
4 rows in set (0.00 sec)
```

Check

```
mysql> select* from supplies;
+-------------+---------+----------+
| supplierNum | partNum | quantity |
+-------------+---------+----------+
| S1          | P1      |      300 |
| S1          | P2      |      200 |
| S1          | P3      |      400 |
| S1          | P4      |      200 |
| S1          | P5      |      100 |
| S1          | P6      |      100 |
| S2          | P1      |      300 |
| S2          | P2      |      400 |
| S3          | P2      |      200 |
+-------------+---------+----------+
9 rows in set (0.00 sec)
```

j) List all the parts except those with a quantity of 200.

Note: assuming exclude the total quantity of 200 of each parts

1) Join two tables

```
mysql> select * from supplies join  parts on supplies.partNum=parts.partNum;
+-------------+---------+----------+---------+-------+--------+--------+--------+
| supplierNum | partNum | quantity | partNum | name  | colour | weight | city   |
+-------------+---------+----------+---------+-------+--------+--------+--------+
| S1          | P1      |      300 | P1      | Nut   | Red    |   12.0 | London |
| S2          | P1      |      300 | P1      | Nut   | Red    |   12.0 | London |
| S1          | P2      |      200 | P2      | Bolt  | Green  |   17.0 | Paris  |
| S2          | P2      |      400 | P2      | Bolt  | Green  |   17.0 | Paris  |
| S3          | P2      |      200 | P2      | Bolt  | Green  |   17.0 | Paris  |
| S1          | P3      |      400 | P3      | Screw | Blue   |   17.0 | Oslo   |
| S1          | P4      |      200 | P4      | Screw | Red    |   14.0 | London |
| S1          | P5      |      100 | P5      | Cam   | Blue   |   12.0 | Paris  |
| S1          | P6      |      100 | P6      | Cog   | Red    |   19.0 | London |
+-------------+---------+----------+---------+-------+--------+--------+--------+
9 rows in set (0.00 sec)
```

2) Sum and group

```
mysql> select parts.name,supplies.partNum, sum(quantity)  from supplies join  parts on supplies.partNum=parts.partNum group by supplies.p
artNum;
+-------+---------+---------------+
| name  | partNum | sum(quantity) |
+-------+---------+---------------+
| Nut   | P1      |           600 |
| Bolt  | P2      |           800 |
| Screw | P3      |           400 |
| Screw | P4      |           200 |
| Cam   | P5      |           100 |
| Cog   | P6      |           100 |
+-------+---------+---------------+
6 rows in set (0.00 sec)
```

3) List all the parts except those with a quantity of 200.

```
mysql> select parts.name,supplies.partNum, sum(quantity)  from supplies join  parts on supplies.partNum=parts.partNum group by supplies.p
artNum having sum(quantity)!="200";
+-------+---------+---------------+
| name  | partNum | sum(quantity) |
+-------+---------+---------------+
| Nut   | P1      |           600 |
| Bolt  | P2      |           800 |
| Screw | P3      |           400 |
| Cam   | P5      |           100 |
| Cog   | P6      |           100 |
+-------+---------+---------------+
5 rows in set (0.00 sec)
```

k) List part names, their colour, and the supplier(s) who supply them.

1) Join three tables required

```
mysql> select * from supplier join supplies on supplier.supplierNum=supplies.supplierNum join parts on supplies.partNum=parts.partNum;
+-------------+-------+--------+--------+-------------+---------+----------+---------+-------+--------+--------+--------+
| supplierNum | name  | status | city   | supplierNum | partNum | quantity | partNum | name  | colour | weight | city   |
+-------------+-------+--------+--------+-------------+---------+----------+---------+-------+--------+--------+--------+
| S1          | Smith |     20 | London | S1          | P1      |      300 | P1      | Nut   | Red    |   12.0 | London |
| S1          | Smith |     20 | London | S1          | P2      |      200 | P2      | Bolt  | Green  |   17.0 | Paris  |
| S1          | Smith |     20 | London | S1          | P3      |      400 | P3      | Screw | Blue   |   17.0 | Oslo   |
| S1          | Smith |     20 | London | S1          | P4      |      200 | P4      | Screw | Red    |   14.0 | London |
| S1          | Smith |     20 | London | S1          | P5      |      100 | P5      | Cam   | Blue   |   12.0 | Paris  |
| S1          | Smith |     20 | London | S1          | P6      |      100 | P6      | Cog   | Red    |   19.0 | London |
| S2          | Jones |     10 | Paris  | S2          | P1      |      300 | P1      | Nut   | Red    |   12.0 | London |
| S2          | Jones |     10 | Paris  | S2          | P2      |      400 | P2      | Bolt  | Green  |   17.0 | Paris  |
| S3          | Blake |     20 | Paris  | S3          | P2      |      200 | P2      | Bolt  | Green  |   17.0 | Paris  |
+-------------+-------+--------+--------+-------------+---------+----------+---------+-------+--------+--------+--------+
9 rows in set (0.00 sec)
```

2) List columns

```
mysql> select supplier.name, parts.name, parts.colour  from supplier join supplies on supplier.supplierNum=supplies.supplierNum join part
s on supplies.partNum=parts.partNum;
+-------+-------+--------+
| name  | name  | colour |
+-------+-------+--------+
| Smith | Nut   | Red    |
| Smith | Bolt  | Green  |
| Smith | Screw | Blue   |
| Smith | Screw | Red    |
| Smith | Cam   | Blue   |
| Smith | Cog   | Red    |
| Jones | Nut   | Red    |
| Jones | Bolt  | Green  |
| Blake | Bolt  | Green  |
+-------+-------+--------+
9 rows in set (0.00 sec)
```