1DV503/1DT903 Database Technology and Modeling

Samuel Berg
School of Computer Science, Physics
and Mathematics, Linnaeus University, Sweden
sb224sc@student.lnu.se

Task 1 Functional Dependencies (15 points)

Details: Convert business statements into dependencies. Consider the relation DISK_DRIVE (Serial_number, Manufacturer, Model, Batch, Capacity, Retailer). Each tuple/record in the relation DISK_DRIVE contains information about a disk drive with a unique Serial_nuber, made by a manufacture, with a particular model number, released in a certain batch, which has a certain storage capacity and is sold by a certain retailer. For example, the tuple/record Disk_drive (‘1978619’, ‘WesternDigital’, ‘A2235X’, ’765234’, 500, ’CompUSA’) specifies that WesternDigital made a disk drive with serial number 1978619 and model number A2235X, released in batch 765234; it
is 500GB and sold by CompUSA.

Q: Write each of the following dependencies as an FD:

a) The manufacturer and serial number uniquely identifies the drive

   Answer: Manufacturer, Serial Number -> Drive - A manufacturer and a serial number only have one drive connected to them.

b) A model number is registered by a manufacturer and, therefore, can not be used by another manufacturer

   Answer: Model -> Manufacturer - For a given model number, there is only one manufacturer that corresponds to it.

c) All disk drivers in a particular batch are the same model

   Answer: Batch -> Model - All disk drivers included in a batch have the same model number, according to method.

Task 2 Normalization (15 points)

Details: Consider the following relation:
CAR_SALE(Car, Date_sold, Salesperson, Commission, Discount)
Assume that a car may be sold by multiple salespeople, and hence
{Car, Salesperson} is the primary key. Additionally, dependencies are:

Date_sold →Discount
Salesperson → Commission

Q: Based on the given primary key, is this relation in 1NF, 2ND, or 3NF? Why or why not? How would you successively normalize it completely?

Answer: As a result of having atomic values and no repeated groups, the connection belongs to 1NF.

Also, it belongs to 2NF since it only has one candidate key and all non-key qualities are totally dependent on the key in order to work. Discount and Commission are determined in this instance by Date sold and Salesperson, respectively.

Due to a transitive reliance between Salesperson and Discount through Date sold, it is not in 3NF, though. By establishing a new relation for Date sold and Discount, we may achieve 3NF by removing this transitive dependence.

CAR_SALE(Car, Date sold, Salesperson, Commission)

DATE_DISCOUNT(Date_sold, Discount)

Would be the first normalized relation.

As there are no transitive dependencies and all non-key qualities are completely dependent on the key, the second relation is now in 3NF.

It is not essential in this instance, although we might have done so if we wanted to separate the Salesperson and Commission into different relations.

Task 3 SQL queries using MySQL Workbench DBMS (60 points)

A)

Answer:

Query:

```
SELECT fname, lname

FROM employee

INNER JOIN works_on ON employee.ssn = works_on.essn

INNER JOIN project ON works_on.pno = project.pnumber

WHERE project.pname = "Computerization";
```

Result:

fname	lname
Franklin	Wong
Ahmad	Jabbar
Alicia	Zelaya

B)

Answer:

Query:

```
SELECT project.pnumber, department.dnumber, employee.lname,
employee.address, employee.bdate

FROM project

INNER JOIN department ON project.dnum = department.dnumber

INNER JOIN employee ON department.mgrssn = employee.ssn

WHERE project.plocation = "Houston";
```

Result:

pnumber	dnumber	lname	address	bdate
3	5	Wong	638 Voss, Houston, TX	1945-12-08
20	1	Borg	450 Stone, Houston, TX	1927-11-10

C)

Answer:

Query:

```
SELECT e.fname AS employee_first_name, e.lname AS employee_last_name, s.fname
AS supervisor_first_name, s.lname AS supervisor_last_name

FROM employee e

LEFT JOIN employee s ON e.superssn = s.ssn;
```

Result:

employee_first_name	employee_last_name	supervisor_first_name	supervisor_last_name
Jared	James		

Jon	Jones	Jared	James
Justin	Mark	Jared	James
Brad	Knight	Jared	James
John	Smith	Franklin	Wong
Evan	Wallis		
Josh	Zell	Evan	Wallis
Andy	Vile	Evan	Wallis
Tom	Brand	Evan	Wallis
Jenny	Vos	Josh	Zell
Chris	Carter	Josh	Zell
Kim	Grace		
Jeff	Chase	Kim	Grace
Franklin	Wong	James	Borg
Alex	Freed		
Bonnie	Bays	Alex	Freed
Alec	Best	Alex	Freed
Sam	Snedden	Alex	Freed
Joyce	English	Franklin	Wong
John	James		
Nandita	Ball	John	James
Bob	Bender		
Jill	Jarvis	Bob	Bender
Kate	King	Bob	Bender
Lyle	Leslie	Jill	Jarvis
Billie	King	Lyle	Leslie
Jon	Kramer	Lyle	Leslie
Ray	King	Billie	King
Gerald	Small	Kate	King
Arnold	Head	Kate	King
Helga	Pataki	Kate	King
Naveen	Drew	Gerald	Small
Carl	Reedy	Naveen	Drew
Sammy	Hall	Carl	Reedy
Red	Bacher	Sammy	Hall
Ramesh	Narayan	Franklin	Wong
James	Borg		
Jennifer	Wallace	James	Borg
Ahmad	Jabbar	Jennifer	Wallace
Alicia	Zelaya	Jennifer	Wallace

D)

Answer:

Query:

```
SELECT *
FROM employee
WHERE address LIKE BINARY '%Atlanta, GA%';
```

Result:

	fname	minit	lname	ssn	bdate	address	sex	salary	superssn	dno
	Jared	D	James	111111100	1966-10-10	123 Peachtree, Atlanta, GA	M	85000		6
	Jon	C	Jones	111111101	1967-11-14	111 Allgood, Atlanta, GA	M	45000	111111100	6
	Justin		Mark	111111102	1966-01-12	2342 May, Atlanta, GA	M	40000	111111100	6
	Brad	C	Knight	111111103	1968-02-13	176 Main St., Atlanta, GA	M	44000	111111100	6
*										

E)

Answer:

Query:

```
SELECT *
FROM employee
WHERE MONTH(bdate) = 11;
```

Result:

	fname	minit	lname	ssn	bdate	address	sex	salary	superssn	dno
	Jon	C	Jones	111111101	1967-11-14	111 Allgood,	M	45000	111111100	6

						Atlanta, GA				
	Jenny	F	Vos	222222204	1967-11-11	263 Mayberry, Milwaukee, WI	F	61000	222222201	7
	James	E	Borg	888665555	1927-11-10	450 Stone, Houston, TX	M	55000		1
*										

F)

Answer:

Query:

```
SELECT d.dname, AVG(e.salary) AS average_salary
FROM department d
JOIN employee e ON d.dnumber = e.dno
GROUP BY d.dname;
```

Result:

dname	average_salary
Administration	31000.0000
Hardware	63450.0000
Headquarters	55000.0000
Research	33250.0000
Sales	40821.4286
Software	60000.0000

G)

Answer:

Query:

```
SELECT fname, lname
FROM employee
WHERE ssn NOT IN (SELECT essn FROM works_on);
```

Result:

fname	lname
Bob	Bender
Kate	King

H)

Answer:

Query:

```
SELECT e.fname, e.lname
FROM employee e
JOIN works_on w ON e.ssn = w.essn
JOIN project p ON w.pno = p.pnumber
WHERE e.dno = 5
AND e.salary > 30000
        AND p.pname = 'ProductZ';
```

Result:

fname	lname
Franklin	Wong
Ramesh	Narayan

I)

Answer:

Query:

```
SELECT e.fname, e.lname
FROM employee e
JOIN employee m ON e.superssn = m.ssn
WHERE m.ssn = '333445555'
AND e.address LIKE BINARY '%Houston, TX%';
```

Result:

fname	lname

John	Smith
Joyce	English

J)

Answer:

Query:

```
SELECT e.fname, e.lname
FROM employee e
JOIN (
SELECT dno, MAX(salary) AS max_salary
from employee
GROUP BY dno
) m ON e.dno = m.dno AND e.salary = m.max_salary;
```

Result:

fname	**lname**
Jared	James
Evan	Wallis
Franklin	Wong
Bob	Bender
James	Borg
Jennifer	Wallace

K)

Answer:

Query:

```
SELECT d.dnumber, d.dname, COUNT(*) AS num_employees
FROM department d
JOIN employee e ON d.dnumber = e.dno
GROUP BY d.dnumber
HAVING AVG(e.salary) > 30000;
```

Result:

dnumber	dname	num_emplyees
1	Headquarters	1
4	Administration	3
5	Research	4
6	Software	8
7	Hardware	10
8	Sales	14

L)

Answer:

Query:

```
SELECT d.dependent_name, d.relationship

FROM dependent d

JOIN employee e ON d.essn = e.ssn

JOIN employee s ON e.superssn = s.ssn

WHERE s.ssn = '333445555'

ORDER BY d.dependent_name ASC;
```

Result:

dependent_name	relationship
Alice	Daughter
Elizabeth	Spouse
Michael	Son