

FUTURE Inc.

DATABASE MANAGEMENT SYSTEM

Taught by Dr. Le Gruenwald

Xiang Yu | CS 4513 | 11/16/18

ID: 112991357

**Tasks Performed Page Number**

[Task 1 1](#_Toc530395203)

[1.1 ER Diagram 1](#_Toc530395204)

[1.2. Relational Database Schema 2](#_Toc530395205)

[Task 2. Data Dictionary 2](#_Toc530395206)

[Task 3 5](#_Toc530395207)

[3.1 Discussion of storage structure for table 5](#_Toc530395208)

[3.2 Discussion of storage structure for tables 6](#_Toc530395209)

[Task 4. SQL and text files showing the creation of tables 8](#_Toc530395210)

[Task 5. Script file showing the entire Java program and its successful compilation 12](#_Toc530395211)

[5.1RESULT PROMGRAMING 12](#_Toc530395212)

[5.2RESULT PROMGRAMING SQL TABLE 14](#_Toc530395213)

[5.3RESULT PROMGRAMING SQL QUERY 17](#_Toc530395214)

[5.4. JAVA CODE 28](#_Toc530395215)

[Task 6. Java program Execution 60](#_Toc530395216)

[6.1. Script file showing the testing of query 1 60](#_Toc530395217)

[6.2. Script file showing the testing of query 2 63](#_Toc530395218)

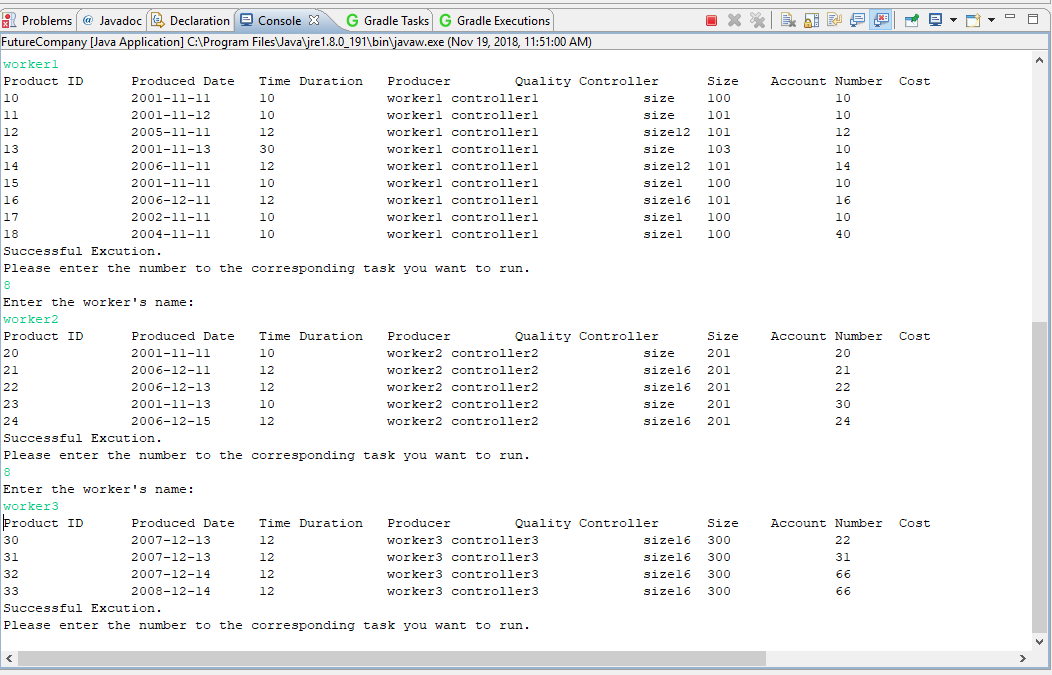
[6.3. Script file showing the testing of query 3 68](#_Toc530395219)

[6.4. Script file showing the testing of query 4 71](#_Toc530395220)

[6.5. Script file showing the testing of query5 74](#_Toc530395221)

[6.6 Script file showing the testing of query6 76](#_Toc530395222)

[6.7 Script file showing the testing of query7 78](#_Toc530395223)

[6.8 Script file showing the testing of query8 79](#_Toc530395224)

[6.9 Script file showing the testing of query9 80](#_Toc530395225)

[6.10Script file showing the testing of query10 81](#_Toc530395226)

[6.11Script file showing the testing of query11 82](#_Toc530395227)

[6.12Script file showing the testing of query12 83](#_Toc530395228)

[6.13Script file showing the testing of query13 84](#_Toc530395229)

[6.14Script file showing the testing of query14 85](#_Toc530395230)

[6.15Script file showing the testing of query15 86](#_Toc530395231)

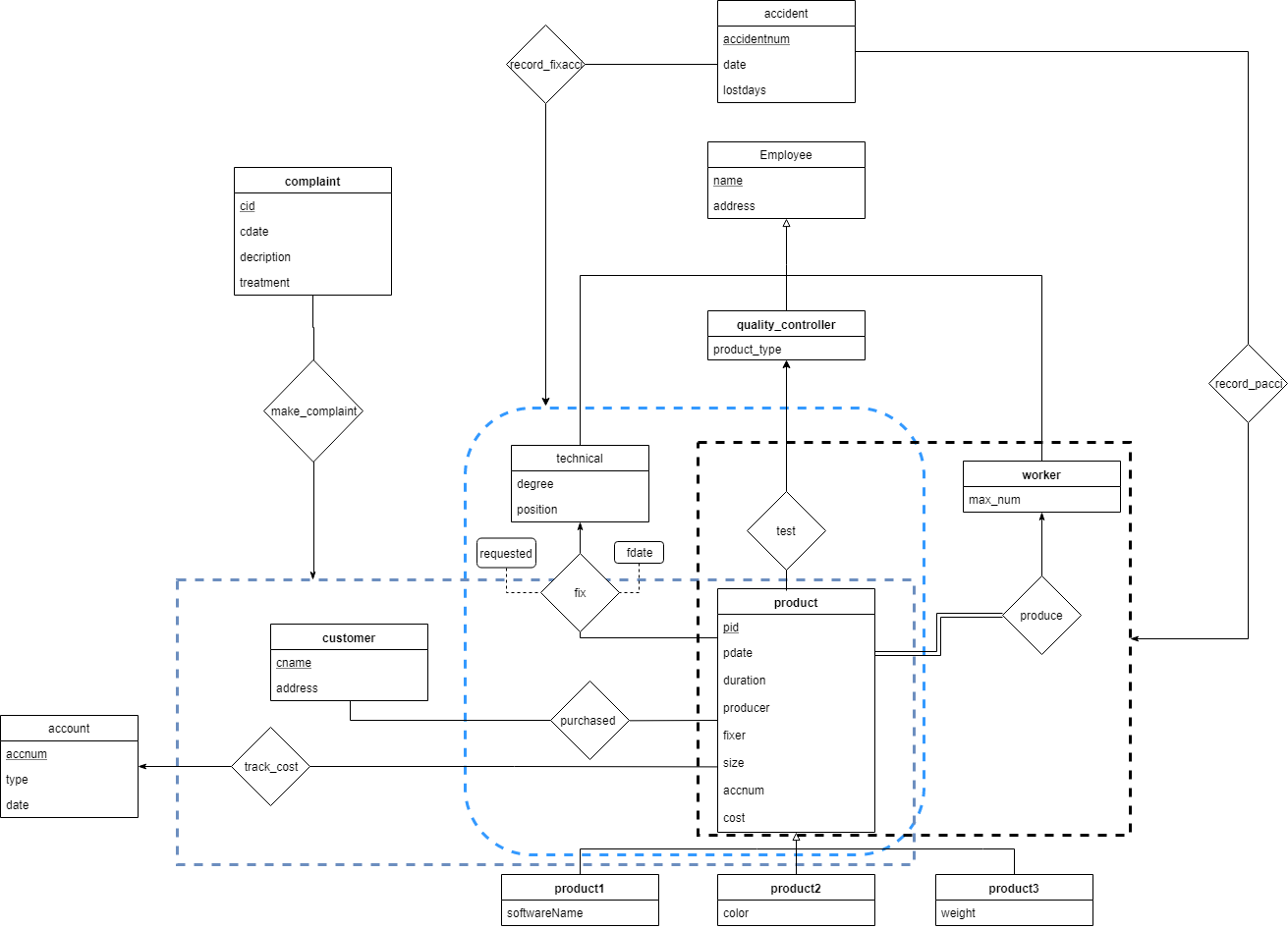
[6.17Script file showing the testing of query17 87](#_Toc530395232)

[6.19Script file showing the testing of query19 88](#_Toc530395233)

[6.19Script file showing the testing of query19 91](#_Toc530395234)

# Task 1

## 1.1 ER Diagram



## 1.2. Relational Database Schema

**Quality\_Controller (**name, product\_type**)**

**Employee (**name**,** address**)**

**Technical** (name, degree, position)

**Fix** (pid, name, fdate, requested)

**Test** (pid, name)

**Worker** (name, max\_num)

**Product** (pid, pdate, duration, producer, size, accnum, cost)

**Product1** (pid, software)

**Product2** (pid, color)

**Product3** (pid, weight)

**Account** (accnum, date, type)

**Customer** (cname,address)

**Purchased** (cname, pid)

**Complaint (**cid, cdate, description, treatment**)**

**Make\_Complaint** (cid, cname, pid)

**Accident** (accidentnum, accidentdate. lostday)

**Record\_pacci** (accidentnum, pid, name)

**Record\_Fixacci** (accidentnum, pid, name)

# Task 2. Data Dictionary

**PK = PRIMARY KEY, FK = FOREIGN KEY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TABLE NAME** | **COLUMN NAME** | **DATA TYPE** | **DATA SIZE (BYTES)** | **CONSTRAINT TYPE/TABLE** |
| Employee | name | VARCHAR | 255 | PK |
| Employee | address | VARCHAR | 255 | - |
| Quality\_Controller | name | VARCHAR | 255 | PK/Employee |
| Quality\_Controller | product\_type | VARCHAR | 50 | - |
| Technical | name | VARCHAR | 255 | PK/Employee |
| Technical | degree | VARCHAR | 10 | - |
| Technical | position | VARCHAR | 50 | - |
| Fix | pid | INT | 20 | PK/Product |
| Fix | name | VARCHAR | 255 | FK/Technical |
| Fix | fdate | VARCHAR | 50 | - |
| Fix | requested | VARCHAR | 50 | - |
| Worker | name | VARCHAR | 255 | PK/Employee |
| Worker | max\_num | INT | - | - |
| Product | pid | INT | - | PK |
| Product | pdate | DATE | - | - |
| Product | duration | INT | - | - |
| Product | producer | VARCHAR | 255 | FK/Worker |
| Product | tester | VARCHAR | 255 | FK/Qult\_Ctrl |
| Product | size | VARCHAR | 50 | - |
| Product | accnum | INT | - | FK/Account |
| Product | cost | INT | - |  |
| Product1 | pid | INT | - | PK/Product |
| Product1 | software | VARCHAR | 255 | - |
| Product2 | pid | INT | - | PK/Product |
| Product2 | color | VARCHAR | 50 | - |
| Product3 | pid | INT | - | PK/Product |
| Product3 | weight | VARCHAR | 50 | - |
| Account | accnum | INT | - | PK |
| Account | date | DATE | - | - |
| Account | type | VARCHAR | 50 | - |
| Customer | cname | VARCHAR | 255 | PK |
| Customer | address | VARCHAR | 255 | - |
| Purchased | cname | VARCHAR | 255 | PK/Customer |
| Purchased | pid | INT | - | PK/Product |
| Complaint | cid | INT | - | PK |
| Complaint | cdate | DATE | - | - |
| Complaint | description | VARCHAR | 255 | - |
| Complaint | treatment | VARCHAR | 50 | - |
| Make\_Complaint | cid | INT | - | PK/Complaint |
| Make\_Complaint | cname | VARCHAR | 255 | FK/ Purchased |
| Make\_Complaint | pid | INT | - | FK/ Purchased |
| Accident | accidentnum | INT | - | PK |
| Accident | accidentdate | DATE | - | - |
| Accident | lostday | INT | - | - |
| Record\_Fixacci | accidentnum | INT | - | FK/Accident |
| Record\_Fixacci | pid | INT | - |  |
| Record\_Fixacci | name | VARCHAR | 50 | FK/Technical |
| Record\_Pacci | accidentnum | INT | - | FK/Accident |
| Record\_Pacci | pid | INT | - |  |
| Record\_Pacci | name | VARCHAR | 50 | FK/Worker |

# Task 3

## 3.1 Discussion of storage structure for table

There are three main file organization such as heap file, index sequential file and dynamic hashing.

As I need to insert bulk data without ordering and don’t not need any searching, I could use heap file organization.

I will use indexed sequential file for the table whose queries have range search

I will use dynamic hash file for the table whose queries have random search

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table Name** | **Type of Query** | **Search Key** | **Frequency** | **File Organization** |
| Employee | [1] insert  [6] random search | -------  name | 2/month  1/weak | Dynamic hash file |
| Quality\_controller | [1] insert  [16] random search | **--------**  name | 2/month | Dynamic hash file |
| Technical | [1] insert  [16] random search | **---------**  name | 2/month  1/3 months | Dynamic hash file |
| Fix | [2] insert  [10] random search  [12] random search  [14] random search | **--------**  requested  pid  name | 400/day  40/day  1/month  5/day | Dynamic hash file |
| Worker | [1] insert  [13] random search | **---------**  name | 2/month  10/month | Dynamic hash file |
| Product | [2] insert  [3] random search  [4] random search  [5] random search  [8] random search  [9] random search  [10]random search  [12] random search  [14] random search  [15] range search | **---------**  pid  pid  pid  name  pid  pid  pid  pid  pdate | 400/day  50/day  40/day  30/day  2000/day  400/day  40/day  1/month  5/day  5/day | Dynamic hash file |
| Product2 | [2] insert  [11] random search | **---------**  color | 400/day  5/month | Dynamic hash file |
| Product3 | [2] insert  [10] random search | **---------**  pid | 40/day | Dynamic hash file |
| Account | [4] insert | **---------** | 40/day | Heap file |
| Customer | [3] insert  [5] random search  [11] random search  [13] random search  [15] random search | **---------**  cname  cname  cname  cname | 50/day  30/day  5/month  10/month  5/day | Dynamic hash file |
| Purchased | [3] insert  [11] random search  [14] random search | **---------**  pid  pid | 50/day  5/month  5/day | Dynamic hash file |
| Complaint | [5] insert | **---------** | 30/day | Heap file |
| Make\_Complaint | [5] insert  [9] random search  [12] random search | **---------**  pid  pid | 30/day  400/day  1/month | Dynamic hash file |
| Accident | [6] insert  [17] range search | **---------**  accidentdate | 1/week  1/day | Indexed sequential file |
| Record\_Fixacci | [6] insert  [12] name | **---------** | 1/week  1/month | Dynamic hash file |
| Record\_pacc | [6] insert | **---------** | 1/week | heap |

## 3.2 Discussion of storage structure for tables

Because almost of search key above are primary keys which will be automatically indexed by the database system. Therefore, we need to index search key that are not primary key but are most searched in the table.

CREATE INDEX FixrequestedIndex

ON Fix (requested)

CREATE INDEX AccidentDateIndex

ON Accident (accidentdate)

CREATE INDEX ProductColorIndex

ON Product2 (color)

CREATE INDEX ProductDateIndex

ON Product (pdate)

# Task 4. SQL and text files showing the creation of tables

CREATE TABLE employee(

name VARCHAR(255) NOT NULL PRIMARY KEY,

address VARCHAR(255) NOT NULL

);

CREATE TABLE quality\_controller(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

product\_type VARCHAR(50) NOT NULL,

CONSTRAINT check\_type CHECK(product\_type in ('product 1', 'product 2', 'product 3'))

);

CREATE TABLE technical(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

degree VARCHAR(10) NOT NULL,

position VARCHAR(50) NOT NULL,

CONSTRAINT check\_degree CHECK(degree in ('BS', 'MS', 'Ph.D'))

);

CREATE TABLE worker(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

max\_mun INT NOT NULL

);

CREATE TABLE account(

accnum INT NOT NULL PRIMARY KEY,

accdate DATE NOT NULL,

acctype VARCHAR(50) NOT NULL,

CONSTRAINT check\_acctype CHECK(acctype in ('product1-account', 'product2-account','product1-account'))

);

CREATE TABLE product(

pid INT NOT NULL PRIMARY KEY,

pdate DATE NOT NULL,

duration INT NOT NULL,

producer VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES worker(name),

tester VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES quality\_controller(name),

accnum INT NOT NULL FOREIGN KEY REFERENCES account (accnum),

size VARCHAR(50) NOT NULL,

CONSTRAINT p\_unique UNIQUE(producer, tester, accnum)

);

CREATE TABLE product1(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

software VARCHAR(255)

);

CREATE TABLE product2(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

color VARCHAR(50) NOT NULL

);

CREATE TABLE product3(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

weight VARCHAR(50) NOT NULL

);

CREATE TABLE fix(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES technical(name),

fdate DATE NOT NULL,

requested VARCHAR(50) NOT NULL,

CONSTRAINT check\_request CHECK(requested in ('complaint', 'controller'))

);

CREATE TABLE customer(

cname VARCHAR(255) NOT NULL PRIMARY KEY,

address VARCHAR(255) NOT NULL

);

CREATE TABLE purchased(

cname VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES customer(cname),

pid INT NOT NULL UNIQUE FOREIGN KEY REFERENCES product(pid)

);

CREATE TABLE complaint(

cid INT NOT NULL PRIMARY KEY,

cdate DATE NOT NULL,

description VARCHAR(255) NOT NULL,

treatment VARCHAR(50) NOT NULL,

CONSTRAINT check\_treatment CHECK(treatment in ('refund', 'exchange'))

);

CREATE TABLE make\_complaint(

cid INT NOT NULL PRIMARY KEY,

cname VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES customer(cname),

pid INT NOT NULL FOREIGN KEY REFERENCES product(pid)

);

CREATE TABLE accident(

accidentnum INT NOT NULL PRIMARY KEY,

accidentdate DATE NOT NULL,

lostday INT NOT NULL

);

CREATE TABLE record\_fixacc(

accidentnum INT NOT NULL PRIMARY KEY REFERENCES accident(accidentnum),

pid INT NOT NULL FOREIGN KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES technical(name)

);

CREATE TABLE record\_pacci(

accidentnum INT NOT NULL PRIMARY KEY REFERENCES accident(accidentnum),

pid INT NOT NULL FOREIGN KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES worker(name)

);

CREATE TABLE track\_cost(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

accnum INT NOT NULL FOREIGN KEY REFERENCES account(accnum),

cost INT NOT NULL

);

CREATE INDEX FixrequestedIndex

ON Fix (requested)

CREATE INDEX AccidentDateIndex

ON Accident (accidentdate)

CREATE INDEX ProductColorIndex

ON Product2 (color)

CREATE INDEX ProductDateIndex

ON Product (pdate)

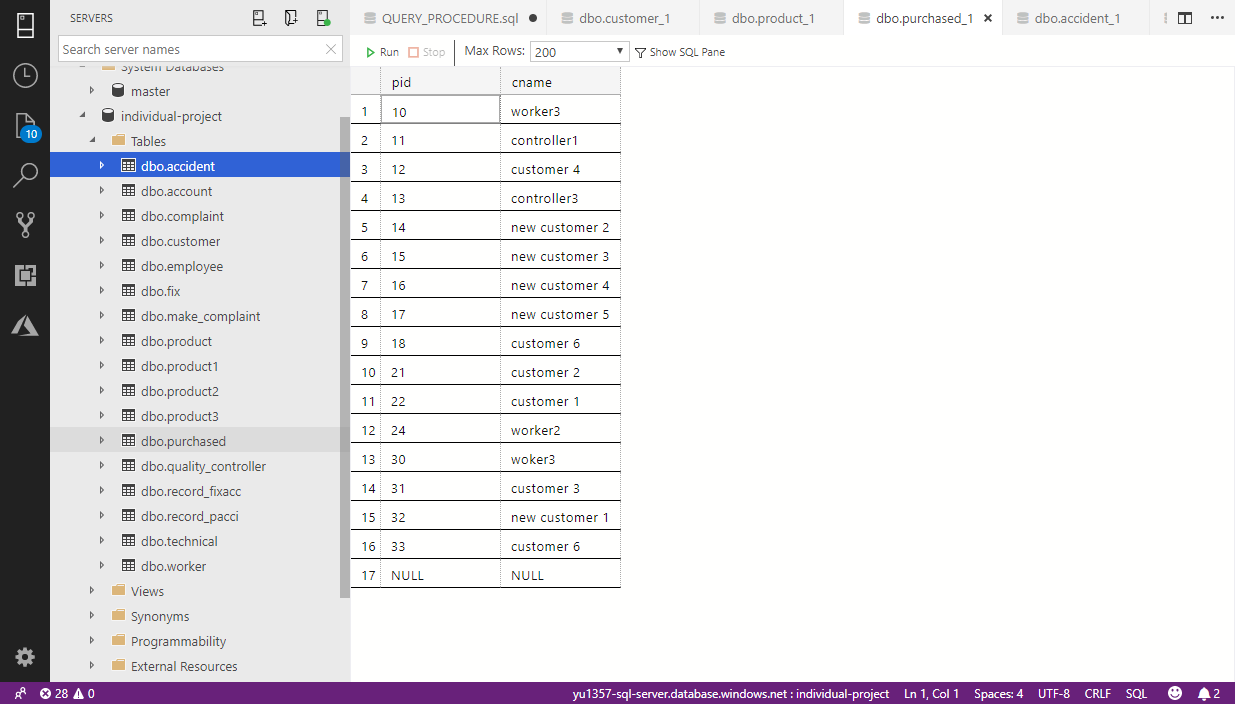
A screenshot of a cell phone

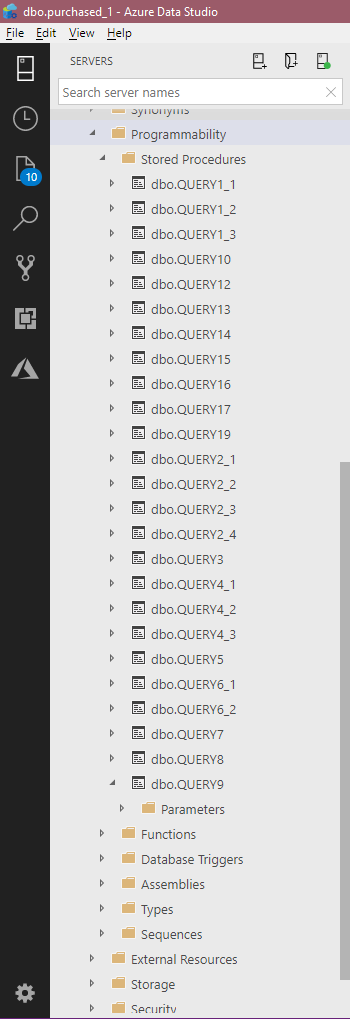
Description generated with very high confidence

RESULT OF CREATED TABLE

# Task 5. Script file showing the entire Java program and its successful compilation

## 5.1RESULT PROMGRAMING





## 5.2RESULT PROMGRAMING SQL TABLE

CREATE TABLE employee(

name VARCHAR(255) NOT NULL PRIMARY KEY,

address VARCHAR(255) NOT NULL

);

CREATE TABLE quality\_controller(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

product\_type VARCHAR(50) NOT NULL,

CONSTRAINT check\_type CHECK(product\_type in ('product 1', 'product 2', 'product 3'))

);

CREATE TABLE technical(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

degree VARCHAR(10) NOT NULL,

position VARCHAR(50) NOT NULL,

CONSTRAINT check\_degree CHECK(degree in ('BS', 'MS', 'Ph.D'))

);

CREATE TABLE worker(

name VARCHAR(255) NOT NULL PRIMARY KEY REFERENCES Employee(name),

max\_mun INT NOT NULL

);

CREATE TABLE account(

accnum INT NOT NULL PRIMARY KEY,

accdate DATE NOT NULL,

acctype VARCHAR(50) NOT NULL,

CONSTRAINT check\_acctype CHECK(acctype in ('product1-account', 'product2-account','product3-account'))

);

CREATE TABLE product(

pid INT NOT NULL PRIMARY KEY,

pdate DATE NOT NULL,

duration INT NOT NULL,

producer VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES worker(name),

tester VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES quality\_controller(name),

size VARCHAR(50) NOT NULL,

accnum INT NOT NULL FOREIGN KEY REFERENCES account (accnum),

cost INT NOT NULL,

);

CREATE TABLE product1(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

software VARCHAR(255)

);

CREATE TABLE product2(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

color VARCHAR(50) NOT NULL

);

CREATE TABLE product3(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

weight VARCHAR(50) NOT NULL

);

CREATE TABLE fix(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES technical(name),

fdate DATE NOT NULL,

requested VARCHAR(50) NOT NULL,

CONSTRAINT check\_request CHECK(requested in ('complaint', 'controller'))

);

CREATE TABLE customer(

cname VARCHAR(255) NOT NULL PRIMARY KEY,

address VARCHAR(255) NOT NULL

);

CREATE TABLE purchased(

pid INT NOT NULL PRIMARY KEY REFERENCES product(pid),

cname VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES customer(cname)

);

CREATE TABLE complaint(

cid INT NOT NULL PRIMARY KEY,

cdate DATE NOT NULL,

description VARCHAR(255) NOT NULL,

treatment VARCHAR(50) NOT NULL,

CONSTRAINT check\_treatment CHECK(treatment in ('refund', 'exchange'))

);

CREATE TABLE make\_complaint(

cid INT NOT NULL PRIMARY KEY REFERENCES complaint(cid),

cname VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES customer(cname),

pid INT NOT NULL FOREIGN KEY REFERENCES purchased(pid)

);

CREATE TABLE accident(

accidentnum INT NOT NULL PRIMARY KEY,

accidentdate DATE NOT NULL,

lostday INT NOT NULL

);

CREATE TABLE record\_fixacc(

accidentnum INT NOT NULL PRIMARY KEY REFERENCES accident(accidentnum),

pid INT NOT NULL FOREIGN KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES technical(name)

);

CREATE TABLE record\_pacci(

accidentnum INT NOT NULL PRIMARY KEY REFERENCES accident(accidentnum),

pid INT NOT NULL FOREIGN KEY REFERENCES product(pid),

name VARCHAR(255) NOT NULL FOREIGN KEY REFERENCES worker(name)

);

CREATE INDEX FixrequestedIndex

ON Fix (requested)

CREATE INDEX AccidentDateIndex

ON Accident (accidentdate)

CREATE INDEX ProductColorIndex

ON Product2 (color)

CREATE INDEX ProductDateIndex

ON Product (pdate)

## 5.3RESULT PROMGRAMING SQL QUERY

--1) Enter a new employee (2/month)

-------------------QUERY 1----------------------------------

--add Technical--

CREATE PROCEDURE QUERY1\_1

@name VARCHAR(255),

@address VARCHAR(255),

@degree VARCHAR(10),

@position VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[employee](name, address) VALUES(@name, @address);

INSERT INTO [dbo].[technical](name, degree, position) VALUES(@name, @degree, @position);

END

GO

--add Quality Controller

CREATE PROCEDURE QUERY1\_2

@name VARCHAR(255),

@address VARCHAR(255),

@product\_type VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[employee](name, address) VALUES(@name, @address);

INSERT INTO [dbo].[quality\_controller](name, product\_type) VALUES(@name, @product\_type);

END

GO

--add worker

CREATE PROCEDURE QUERY1\_3

@name VARCHAR(255),

@address VARCHAR(255),

@max\_num VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[employee](name, address) VALUES(@name, @address);

INSERT INTO [dbo].[worker](name, max\_mun) VALUES(@name, @max\_num);

END

GO

-------------------QUERY 1 end----------------------------------

--Enter a new product associated with the person

--who made the product, repaired the product if it is repaired, or checked the product (400/day).

-------------------QUERY2----------------------------------

--add product1

CREATE PROCEDURE QUERY2\_1

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@accnum INT,

@cost INT,

@software VARCHAR(255)

AS

BEGIN

INSERT INTO [dbo].[product](pid, pdate, duration, producer, tester, size, accnum, cost) VALUES (@pid, @pdate, @duration, @producer, @tester, @size, @accnum, @cost);

INSERT INTO [dbo].[product1](pid, software) VALUES(@pid, @software)

END

GO

--add product2

CREATE PROCEDURE QUERY2\_2

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@accnum INT,

@cost INT,

@color VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[product](pid, pdate, duration, producer, tester, size, accnum, cost) VALUES (@pid, @pdate, @duration, @producer, @tester, @size, @accnum, @cost);

INSERT INTO [dbo].[product2](pid, color) VALUES(@pid, @color)

END

GO

--add Product 3

CREATE PROCEDURE QUERY2\_3

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@accnum INT,

@cost INT,

@weight VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[product](pid, pdate, duration, producer, tester, size, accnum, cost) VALUES (@pid, @pdate, @duration, @producer, @tester, @size, @accnum, @cost);

INSERT INTO [dbo].[product3](pid, weight) VALUES(@pid, @weight)

END

GO

--add fixer

CREATE PROCEDURE QUERY2\_4

@pid INT,

@name VARCHAR(255),

@fdate DATE,

@requested VARCHAR(50)

AS

BEGIN

INSERT INTO [dbo].[fix](pid, name, fdate, requested) VALUES(@pid, @name, @fdate, @requested)

END

GO

-------------------------------------------------query2 end-----------------------------------------

--Enter a customer associated with some products (50/day).

---------------Query 3-----------------------

CREATE PROCEDURE QUERY3

@cname VARCHAR(255),

@address VARCHAR(255),

@pid INT

AS

BEGIN

INSERT INTO [dbo].[customer](cname, address) VALUES(@cname, @address);

INSERT INTO [dbo].[purchased](cname, pid) VALUES(@cname, @pid);

END

GO

--4)Create a new account associated with a product (40/day).

-------------------Query4--------------

--product 1 without fixed. account associated

CREATE PROCEDURE QUERY4\_1

@accnum INT,

@accdate DATE,

@acctype VARCHAR(50),

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@software VARCHAR(255),

@cost INT

AS

BEGIN

INSERT INTO [dbo].[account](accnum, accdate, acctype) VALUES(@accnum, @accdate, @acctype);

EXEC QUERY2\_1 @pid = @pid, @pdate = @pdate, @duration = @duration, @producer = @producer, @tester = @tester, @size = size, @accnum = @accnum, @cost = @cost ,@software = @software;

END

GO

--product 2 without fixed account associated

CREATE PROCEDURE QUERY4\_2

@accnum INT,

@accdate DATE,

@acctype VARCHAR(50),

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@color VARCHAR(50),

@cost INT

AS

BEGIN

INSERT INTO [dbo].[account](accnum, accdate, acctype) VALUES(@accnum, @accdate, @acctype);

EXEC QUERY2\_2 @pid = @pid, @pdate = @pdate, @duration = @duration, @producer = @producer, @tester = @tester, @size = size, @accnum = @accnum, @cost = @cost, @color = @color;

END

GO

--product 3 without fixed account associated

CREATE PROCEDURE QUERY4\_3

@accnum INT,

@accdate DATE,

@acctype VARCHAR(50),

@pid INT,

@pdate DATE,

@duration INT,

@producer VARCHAR(255),

@tester VARCHAR(255),

@size VARCHAR(255),

@weight VARCHAR(50),

@cost INT

AS

BEGIN

INSERT INTO [dbo].[account](accnum, accdate, acctype) VALUES(@accnum, @accdate, @acctype);

EXEC QUERY2\_3 @pid = @pid, @pdate = @pdate, @duration = @duration, @producer = @producer, @tester = @tester, @size = size, @accnum = @accnum, @cost = @cost, @weitht = @weight;

END

GO

--5) Enter a complaint associated with a customer and product (30/day).

---------------Query5-------------

CREATE PROCEDURE QUERY5

@cid INT,

@cdate DATE,

@description VARCHAR(255),

@treatment VARCHAR(50),

@cname VARCHAR(255),

@pid INT

AS

BEGIN

INSERT INTO [dbo].[complaint](cid, cdate, [description], treatment) VALUES(@cid, @cdate, @description, @treatment);

INSERT INTO [dbo].[make\_complaint](cid, cname, pid) VALUES(@cid, @cname, @pid);

END

GO

--6) Enter an accident associated with appropriate employee and product (1/week).

---------------------Query 6--------------

----fixed accident

CREATE PROCEDURE QUERY6\_1

@accidentnum INT,

@accidentdate DATE,

@lostday INT,

@pid INT,

@name VARCHAR(255)

AS

BEGIN

INSERT INTO [dbo].[accident](accidentnum, accidentdate, lostday) VALUES(@accidentnum, @accidentdate, @lostday);

INSERT INTO [dbo].[record\_fixacc](accidentnum, pid, name) VALUES(@accidentnum, @pid, @name);

END

GO

----produce accident

CREATE PROCEDURE QUERY6\_2

@accidentnum INT,

@accidentdate DATE,

@lostday INT,

@pid INT,

@name VARCHAR(255)

AS

BEGIN

INSERT INTO [dbo].[accident](accidentnum, accidentdate, lostday) VALUES(@accidentnum, @accidentdate, @lostday);

INSERT INTO [dbo].[record\_pacci](accidentnum, pid, name) VALUES(@accidentnum, @pid, @name);

END

GO

-----------------------------Query6 end---------------------------

--7) Retrieve the date produced and time spent to produce a particular product (100/day).

-----------------------------Query7-------------------------------

CREATE PROCEDURE QUERY7

@pid INT,

@pdate DATE OUTPUT,

@duration INT OUTPUT

AS

BEGIN

SELECT @pdate = pdate, @duration = duration FROM [dbo].[product] WHERE pid = @pid;

END

GO

--8) Retrieve all products made by a particular worker (2000/day).

-----------------------Query 8---------------------

CREATE PROCEDURE QUERY8

@name VARCHAR(255)

AS

BEGIN

SELECT \*FROM [dbo].[Product]

WHERE producer = @name;

END

GO

--9)Retrieve the total number of errors a particular quality controller made.

--This is the total number of products certified by this controller and got some complaints (400/day).

-------------------------QUERY 9-------------------------------

CREATE PROCEDURE QUERY9

@name VARCHAR(255),

@count INT OUTPUT

AS

BEGIN

SELECT @count = COUNT(pid) FROM [dbo].[make\_complaint]

WHERE pid IN (SELECT pid FROM [dbo].[product] WHERE tester = @name);

END

GO

-- Retrieve the total costs of the products in the product3 category

--which were repaired at the request of a particular quality controller (40/day).

-------------------------------QUERY 10---------------------------------

CREATE PROCEDURE QUERY10

@name VARCHAR(255),

@total\_cost INT OUTPUT

AS

BEGIN

SELECT @total\_cost = SUM(cost) FROM [dbo].[product]

WHERE tester = @name AND pid IN (SELECT pid FROM [dbo].[fix] WHERE requested = 'controller')

AND pid IN (SELECT pid FROM [dbo].[product3])

END

GO

--11) Retrieve all customers who purchased all products of a particular color (5/month)

----------------------------------------QUERY 11---------------------------

CREATE PROCEDURE QUERY11

@color VARCHAR(50)

AS

BEGIN

WITH nn AS (SELECT cname FROM [dbo].[purchased] WHERE pid in (SELECT pid FROM [dbo].[product2] WHERE color = @color))

SELECT \*

FROM [dbo].[customer]

WHERE cname IN (SELECT cname FROM [dbo].[purchased] WHERE pid = (SELECT COUNT(pid) FROM [dbo].[product2] WHERE color = @color))

END

GO

--12)Retrieve the total number of work days lost due to accidents in repairing the products which got complaints (1/month).

------------------------------------------Query 12-------------------------------

CREATE PROCEDURE QUERY12

@totaldays INT OUTPUT

AS

BEGIN

SELECT @totaldays = SUM(lostday) FROM [dbo].[accident] WHERE accidentnum IN (SELECT accidentnum FROM [dbo].[record\_fixacc] WHERE pid IN (SELECT pid FROM [dbo].[make\_complaint]))

END

GO

--13)Retrieve all customers who are also workers (10/month)

----------------------------QUERY 13------------------

CREATE PROCEDURE QUERY13

AS

BEGIN

SELECT \* FROM [dbo].[customer] WHERE cname IN (SELECT name FROM [dbo].[worker])

END

GO

--14)Retrieve all the customers who have purchased the products made or certified or repaired by themselves (5/day).

------------------------------Query14-----------------------------

CREATE PROCEDURE QUERY14

AS

BEGIN

SELECT cname

FROM [dbo].[purchased]

WHERE cname IN (SELECT producer FROM [dbo].[product] WHERE [dbo].[product].pid = [dbo].[purchased].pid)

OR cname IN (SELECT tester FROM [dbo].[product] WHERE [dbo].[product].pid = [dbo].[purchased].pid)

OR cname IN (SELECT name FROM [dbo].[fix] WHERE [dbo].[fix].pid = [dbo].[purchased].pid);

END

GO

--15)Retrieve the average cost of all products made in a particular year (5/day).

---------------------------------QUERRY 15-----------------------

CREATE PROCEDURE QUERY15

@year INT,

@ave\_cost INT OUTPUT

AS

BEGIN

SELECT @ave\_cost = AVG(cost)

FROM [dbo].[product]

WHERE YEAR(pdate) = @year

END

GO

--16)Switch the position between a technical staff and a quality controller (1/ 3 months).

----------------------------------QUERY 16------------------------------

CREATE PROCEDURE QUERY16

@tec\_name VARCHAR(255),

@ctrl\_name VARCHAR(255)

AS

BEGIN

UPDATE [dbo].[product]

SET tester = @tec\_name

WHERE tester = @ctrl\_name;

UPDATE [dbo].[quality\_controller]

SET name = @tec\_name

WHERE name = @ctrl\_name;

UPDATE [dbo].[fix]

SET name = @ctrl\_name

WHERE name = @tec\_name

UPDATE [dbo].[technical]

SET name = @ctrl\_name

WHERE name = @tec\_name

END

GO

---17) Delete all accidents whose dates are in some range (1/day).

---------------------------------Query 17----------------------

CREATE PROCEDURE QUERY17

@startDate DATE,

@endDate DATE

AS

BEGIN

ALTER TABLE [dbo].[accident] NOCHECK CONSTRAINT ALL

DELETE FROM [dbo].[accident]

WHERE accidentdate <= @endDate AND accidentdate >= @startDate;

ALTER TABLE [dbo].[accident] CHECK CONSTRAINT ALL

END

GO

--(18) Import: enter new customers from a data file until the file is empty (the user must be asked to enter the input file name);

------------------------------------------Qury18---------------------------------

------------------USE QUERY3 FOR THIS

--19 Export: Retrieve all customers (in name order) and output them to a data file instead of screen (the user must be asked to enter the output file name);

--------------------------------------------QUERY 19-------------------

CREATE PROCEDURE QUERY19

AS

BEGIN

SELECT \*

FROM [dbo].[customer]

ORDER BY cname

END

GO

## 5.4. JAVA CODE

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.sql.BatchUpdateException;

import java.sql.CallableStatement;

import java.sql.Connection;

import java.sql.Date;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Scanner;

import com.microsoft.sqlserver.jdbc.SQLServerDataSource;

import au.com.bytecode.opencsv.CSVReader;

import au.com.bytecode.opencsv.CSVWriter;

public class FutureCompany{

public static void main(String[] args) throws SQLException {

// connect to database

String hostName = "yu1357-sql-server.database.windows.net";

String dbName = "individual-project";

String user = "yu1357";

String password = "Yx28408483";

String url = String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;hostNameInCertificate=\*.database.windows.net;loginTimeout=30;", hostName, dbName, user, password);

try(final Connection test = DriverManager.getConnection(url)){

final String schema = test.getSchema();

System.out.print("Successful connected to - Schema: " + schema);

System.out.println("");

}

catch(Exception e){

System.out.println("Unable to access the database!");

System.out.println(e.getMessage());

System.exit(0);

}

Connection connection = DriverManager.getConnection(url);

Scanner scan = new Scanner(System.in);

String menu =

"WELCOME TO THE Future Inc. DATABASE SYSTEM.\n"

+ "Please enter the number to the corresponding task you want to run.\n"

+ "\n"

+ "(1) Enter a new employee into the database.\n"

+ "(2) Enter a new product into database associated with the person who made the product, repaired the product if it is repaired, or checked the product.\n"

+ "(3) Enter a customer into database associated with some products.\n"

+ "(4) Create a new account associated with a product into databse.\n"

+ "(5) Enter a complaint associated with a customer and product.\n"

+ "(6) Enter an accident associated with appropriate employee and product.\n"

+ "(7) Retrieve the date produced and time spent to produce a particular product.\n"

+ "(8) Retrieve all products made by a particular worker.\n"

+ "(9) Retrieve the total number of errors a particular quality controller made. This is the total number of products certified by this controller and got some complaints.\n"

+ "(10) Retrieve the total costs of the products in the product3 category which were repaired at the request of a particular quality controller.\n"

+ "(11) Retrieve all customers who purchased all products of a particular color.\n"

+ "(12) Retrieve the total number of work days lost due to accidents in repairing the products which got complaints.\n"

+ "(13) Retrieve all customers who are also workers.\n"

+ "(14) Retrieve all the customers who have purchased the products made or certified or repaired by themselves.\n"

+ "(15) Retrieve the average cost of all products made in a particular year.\n"

+ "(16) Switch the position between a technical staff and a quality controller.\n"

+ "(17) Delete all accidents whose dates are in some range.\n"

+ "(18) Import: enter new customers from a data file until the file is empty (the user must be asked to enter the input file name).\n"

+ "(19) Export: Retrieve all customers (in name order) and output them to a data file instead of screen (the user must be asked to enter the output file name).\n"

+ "(20) Quit.";

System.out.println("\n" + menu + "\n");

while(true){

System.out.println("Please enter the number to the corresponding task you want to run.");

int task = Integer.parseInt(scan.nextLine());

String[] arg={};

String input;

switch (task){

case 1:

System.out.println("Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)");

switch(Integer.parseInt(scan.nextLine())){

case 1:

System.out.println("Enter your input in follwing format: Technical name, adress, dgree, technical position");

input = scan.nextLine();

arg = input.split(", ");

query1(connection, arg, 1);

break;

case 2:

System.out.println("Enter your input in follwing format: Quality Controller name, adress, product type(type like product # ex. product 1)");

input = scan.nextLine();

arg = input.split(", ");

query1(connection, arg, 2);

break;

case 3:

System.out.println("Enter your input in follwing format: Worker name, adress, max produces/day");

input = scan.nextLine();

arg = input.split(", ");

query1(connection, arg, 3);

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

break;

case 2:

System.out.println("Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)");

String pid = "";

switch(Integer.parseInt(scan.nextLine())){

case 1:

System.out.println("Enter your input in follwing format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used");

input = scan.nextLine();

arg = input.split(", ");

pid = arg[0];

query2(connection, arg, 1);

break;

case 2:

System.out.println("Enter your input in follwing format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, color");

input = scan.nextLine();

arg = input.split(", ");

pid = arg[0];

query2(connection, arg, 2);

break;

case 3:

System.out.println("Enter your input in follwing format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, weight");

input = scan.nextLine();

arg = input.split(", ");

pid = arg[0];

query2(connection, arg, 3);

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

if(pid.length() > 0){

System.out.println("Is the product repaired? 1 for yes, 0 for no");

if(Integer.parseInt(scan.nextLine())== 1){

System.out.println("Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)");

input = pid + ", " + scan.nextLine();

arg = input.split(", ");

query2(connection, arg, 4);

}

}

break;

case 3:

System.out.println("Enter your input in follwing format: customer name, adress, product ID");

input = scan.nextLine();

arg = input.split(", ");

query3(connection, arg);

break;

case 4:

System.out.println("What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)");

int temp = Integer.parseInt(scan.nextLine());

System.out.println("Enter your input in follwing format: account number, account established date(YYYY-MM-DD)");

input = scan.nextLine();

String pid4 = "";

switch(temp){

case 1:

System.out.println("Enter product " + temp +" associated by account by following format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, sotfware name major used, cost, ");

input = input + ", product1-account, " + scan.nextLine();

arg = input.split(", ");

pid4 = arg[3];

query4(connection, arg, 1);

break;

case 2:

System.out.println("Enter product " + temp +" associated by account by following format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, color, cost");

input = input + ", product2-account, " + scan.nextLine();

arg = input.split(", ");

pid4 = arg[3];

query4(connection, arg, 2);

break;

case 3:

System.out.println("Enter product " + temp +" associated by account by following format: ");

System.out.println("product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name, size, weight, cost");

input = input + ", product3-account, " + scan.nextLine();

arg = input.split(", ");

pid4 = arg[3];

query4(connection, arg, 3);

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

if(pid4.length() > 0){

System.out.println("Is the product repaired? 1 for yes, 0 for no");

if(Integer.parseInt(scan.nextLine())== 1){

System.out.println("Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)");

input = pid4 + ", " + scan.nextLine();

arg = input.split(", ");

query2(connection, arg, 4);

}

}

break;

case 5:

System.out.println("Enter your input in follwing format: compaint ID, compaint date(YYYY-MM-DD), description, treatment type in (refund/exchange),/n custemer name, product ID");

input = scan.nextLine();

arg = input.split(", ");

query5(connection, arg);

break;

case 6:

System.out.println("Enter your input in follwing format: accident number, accident date(YYYY-MM-DD), lost days, product ID, Employee name");

input = scan.nextLine();

arg = input.split(", ");

System.out.println("Enter 1 for fix accident, 2 for producing accident.");

query6(connection, arg, Integer.parseInt(scan.nextLine()));

break;

case 7:

System.out.println("Enter the product ID: ");

input = scan.nextLine();

query7(connection, input);

break;

case 8:

System.out.println("Enter the worker's name: ");

input = scan.nextLine();

query8(connection, input);

break;

case 9:

System.out.println("Enter the Quality Controller's name: ");

input = scan.nextLine();

query9(connection, input);

break;

case 10:

System.out.println("Enter the Quality Controller's name: ");

input = scan.nextLine();

query10(connection, input);

break;

case 11:

System.out.println("Enter color: ");

query11(connection,scan.nextLine());

break;

case 12:

query12(connection);

break;

case 13:

query13(connection);

break;

case 14:

query14(connection);

break;

case 15:

System.out.println("Enter the year(YYYY): ");

input = scan.nextLine();

query15(connection, input);

break;

case 16:

System.out.println("Enter the: technical name, Quality Controller name: ");

input = scan.nextLine();

arg = input.split(", ");

query16(connection, arg);

break;

case 17:

System.out.println("Enter the: start date, end date(YYYY-MM-DD): ");

input = scan.nextLine();

arg = input.split(", ");

query17(connection, arg);

break;

case 18:

System.out.println("Enter the input file: ");

input = scan.nextLine();

query18(connection, input);

break;

case 19:

System.out.println("Enter the output file: ");

input = scan.nextLine();

query19(connection, input);

break;

case 20:

System.out.println("Thank you for using Future Inc database, Bye!");

scan.close();

connection.close();

System.exit(0);

break;

}

}

}

/\*\*THIS IS QUERY 1\*/

public static void query1(Connection connection, String[] arg, int i){

try{

CallableStatement stmt;

switch(i){

case 1:

stmt = connection.prepareCall("{call QUERY1\_1(?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.execute();

break;

case 2:

stmt = connection.prepareCall("{call QUERY1\_2(?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.execute();

break;

case 3:

stmt = connection.prepareCall("{call QUERY1\_3(?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.execute();

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query2(Connection connection, String[] arg, int i){

try{

CallableStatement stmt;

switch(i){

case 1:

stmt = connection.prepareCall("{call QUERY2\_1(?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.execute();

break;

case 2:

stmt = connection.prepareCall("{call QUERY2\_2(?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.execute();

break;

case 3:

stmt = connection.prepareCall("{call QUERY2\_3(?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.execute();

break;

case 4:

stmt = connection.prepareCall("{call QUERY2\_4(?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.execute();

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query3(Connection connection, String[] arg){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY3(?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.execute();

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query4(Connection connection, String[] arg, int i){

try{

CallableStatement stmt;

switch(i){

case 1:

stmt = connection.prepareCall("{call QUERY4\_1(?,?,?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.setString(10, arg[9]);

stmt.setString(11, arg[10]);

stmt.execute();

break;

case 2:

stmt = connection.prepareCall("{call QUERY4\_2(?,?,?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.setString(10, arg[9]);

stmt.setString(11, arg[10]);

stmt.execute();

break;

case 3:

stmt = connection.prepareCall("{call QUERY4\_3(?,?,?,?,?,?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.setString(7, arg[6]);

stmt.setString(8, arg[7]);

stmt.setString(9, arg[8]);

stmt.setString(10, arg[9]);

stmt.setString(11, arg[10]);

stmt.execute();

break;

}

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query5(Connection connection, String[] arg){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY5(?,?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.setString(6, arg[5]);

stmt.execute();

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query6(Connection connection, String[] arg, int i){

try{

CallableStatement stmt;

switch(i){

case 1:

stmt = connection.prepareCall("{call QUERY6\_1(?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.execute();

break;

case 2:

stmt = connection.prepareCall("{call QUERY6\_2(?,?,?,?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.setString(3, arg[2]);

stmt.setString(4, arg[3]);

stmt.setString(5, arg[4]);

stmt.execute();

break;

default:

System.out.println("Fail to insert data! please try again!");

break;

}

System.out.println("Successful Excution.");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query7(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY7(?,?,?)}");

stmt.setString(1, in);

stmt.registerOutParameter("pdate", java.sql.Types.DATE);

stmt.registerOutParameter("duration", java.sql.Types.INTEGER);

stmt.execute();

System.out.println("DATE Duration Time");

System.out.println(stmt.getDate("pdate") + " " + stmt.getInt("duration"));

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query8(Connection connection, String arg){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY8(?)}");

stmt.setString(1, arg);

ResultSet rs = stmt.executeQuery();

System.out.format("%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n",

"Product ID", "Produced Date", "Time Duration", "Producer", "Quality Controller", "Size", "Account Number", "Cost");

while (rs.next())

{

int pid = rs.getInt("pid");

Date pdate = rs.getDate("pdate");

int duration = rs.getInt("duration");

String producer = rs.getString("producer");

String tester = rs.getString("tester");

String size = rs.getString("size");

int accnum = rs.getInt("accnum");

int cost = rs.getInt("cost");

// print the results

System.out.format("%s\t\t%s\t%s\t\t%s\t%s\t\t%s\t%s\t\t%s\n",

pid, pdate, duration, producer, tester, size, accnum, cost);

}

System.out.println("Successful Excution.");

stmt.close();

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

}

public static void query9(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY9(?,?)}");

stmt.setString(1, "worker 1");

stmt.registerOutParameter("count", java.sql.Types.INTEGER);

stmt.execute();

System.out.println("The number of error made by " + in + ": " + stmt.getInt("count"));

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query10(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY10(?,?)}");

stmt.setString(1, in);

stmt.registerOutParameter("total\_cost", java.sql.Types.INTEGER);

stmt.execute();

System.out.println("Total Cost of product 3 which repiration requested by "+ in + ":\t" +stmt.getInt("total\_cost"));

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query11(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY11(?)}");

stmt.setString(1, in);

ResultSet rs = stmt.executeQuery();

System.out.format("%s\t\t\t%s\n", "Customer Name", "Adress");

while (rs.next())

{

String cname = rs.getString("cname");

String address = rs.getString("address");

System.out.format("%s\t\t\t%s\n", cname, address);

}

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query12(Connection connection){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY12(?)}");

stmt.registerOutParameter("totaldays", java.sql.Types.INTEGER);

stmt.execute();

System.out.println("Total Lost day: " + " " + stmt.getInt("totaldays"));

}catch(Exception e){

System.out.println("Fail execution!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query13(Connection connection){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY13()}");

ResultSet rs = stmt.executeQuery();

System.out.format("%s\t\t\t%s\n", "Customer Name", "Adress");

while (rs.next())

{

String cname = rs.getString("cname");

String address = rs.getString("address");

// print the results

System.out.format("%s\t\t\t%s\n", cname, address);

}

stmt.close();

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query14(Connection connection){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY14()}");

ResultSet rs = stmt.executeQuery();

System.out.format("%s\n", "Customer Name");

while (rs.next())

{

String cname = rs.getString("cname");

// print the results

System.out.format("%s\n", cname);

}

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query15(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY15(?,?)}");

stmt.setString(1, in);

stmt.registerOutParameter("ave\_cost", java.sql.Types.INTEGER);

stmt.execute();

System.out.println("AVERAGE COST IN " + in + " is: " + stmt.getInt("ave\_cost"));

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query16(Connection connection, String[] arg){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY7(?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.execute();

System.out.println("Update Sucessfully!");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query17(Connection connection, String[] arg){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY17(?,?)}");

stmt.setString(1, arg[0]);

stmt.setString(2, arg[1]);

stmt.execute();

System.out.println("Executed Sucessfully!");

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query18(Connection connection, String in){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY3(?,?,?)}");

CSVReader reader = null;

reader = new CSVReader(new FileReader(in));

String[] agrs;

int lineNum = 0;

try {

while ((agrs = reader.readNext()) != null) {

lineNum++;

stmt.setString(1, agrs[0]);

stmt.setString(2, agrs[1]);

stmt.setString(3, agrs[2]);

stmt.addBatch();

}

} catch (NumberFormatException e1) {

e1.printStackTrace();

} catch (IOException e1) {

e1.printStackTrace();

}

int[] numRecords = new int[lineNum];

try {

numRecords = stmt.executeBatch();

} catch(BatchUpdateException e) {

numRecords = e.getUpdateCounts();

System.out.println("(ERROR) INSERTION exception: " + e.getMessage());

}

reader.close();

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

System.out.println("\n\n\n");

}

public static void query19(Connection connection, String out){

try{

CallableStatement stmt = connection.prepareCall("{call QUERY19()}");

CSVWriter writer = new CSVWriter(new FileWriter(out));

ResultSet rs = stmt.executeQuery();

String in[] = new String[2];

while (rs.next()){

in[0] = rs.getString("cname");

in[1] = rs.getString("address");

writer.writeNext(in);

}

writer.close();

}catch(Exception e){

System.out.println("Fail to insert data!");

e.printStackTrace();

}

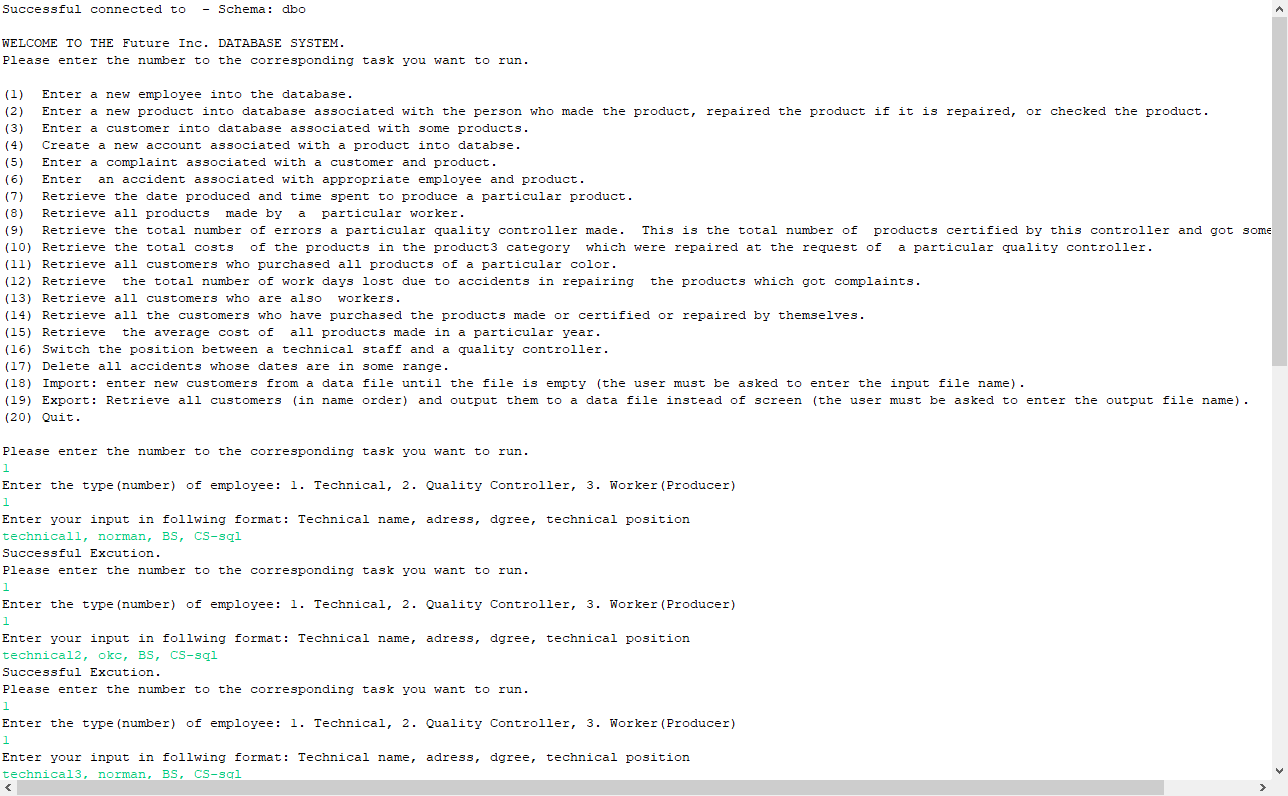
System.out.println("\n\n\n");

}

}

# Task 6. Java program Execution

## 6.1. Script file showing the testing of query 1



Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

1

Enter your input in follwing format: Technical name, adress, dgree, technical position

technical1, norman, BS, CS-sql

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

1

Enter your input in follwing format: Technical name, adress, dgree, technical position

technical2, okc, BS, CS-sql

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

1

Enter your input in follwing format: Technical name, adress, dgree, technical position

technical3, norman, BS, CS-sql

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

1

Enter your input in follwing format: Technical name, adress, dgree, technical position

technical4, norman, BS, CS-sql

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

2

Enter your input in follwing format: Quality Controller name, adress, product type(type like product # ex. product 1)

controller1, norman, product 1

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

2

Enter your input in follwing format: Quality Controller name, adress, product type(type like product # ex. product 1)

controller2, norman, product 2

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

2

Enter your input in follwing format: Quality Controller name, adress, product type(type like product # ex. product 1)

controller3, norman, product 3

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

3

Enter your input in follwing format: Worker name, adress, max produces/day

worker1, norman, 10

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

3

Enter your input in follwing format: Worker name, adress, max produces/day

worker2, norman, 20

Successful Excution.

Please enter the number to the corresponding task you want to run.

1

Enter the type(number) of employee: 1. Technical, 2. Quality Controller, 3. Worker(Producer)

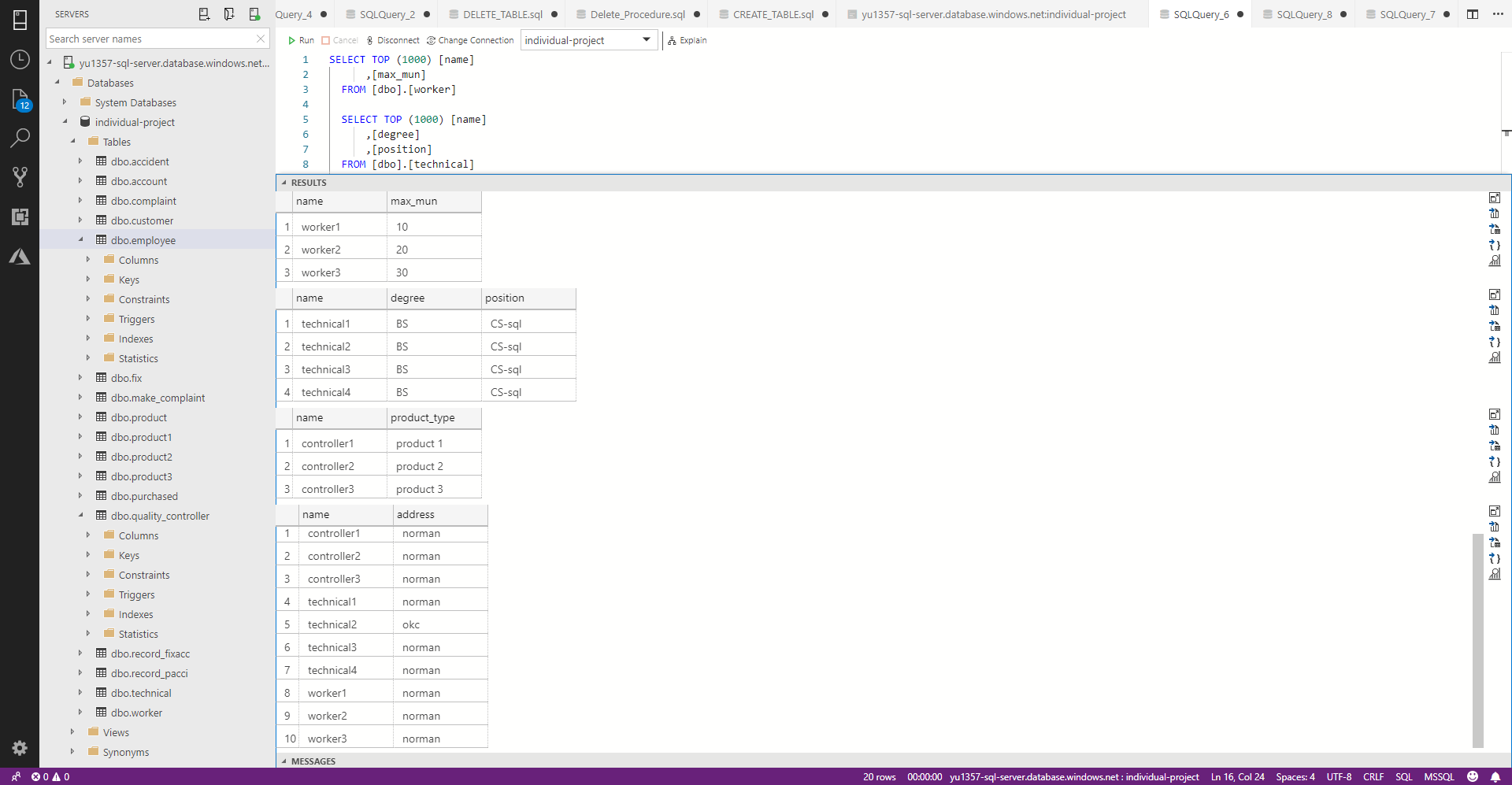
3

Enter your input in follwing format: Worker name, adress, max produces/day

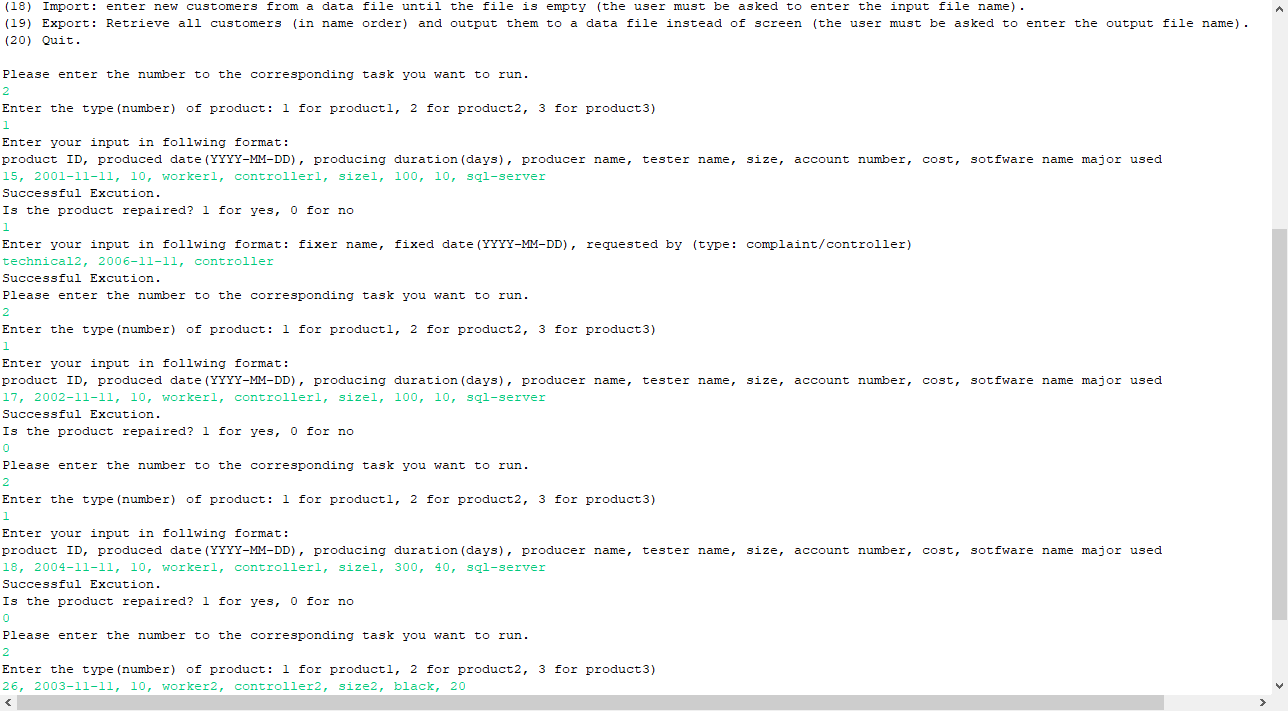
worker3, norman, 30

Successful Excution.

Please enter the number to the corresponding task you want to run.



## 6.2. Script file showing the testing of query 2



Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

1

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used

15, 2001-11-11, 10, worker1, controller1, size1, 100, 10, sql-server

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical2, 2006-11-11, controller

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

1

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used

17, 2002-11-11, 10, worker1, controller1, size1, 100, 10, sql-server

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

1

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used

18, 2004-11-11, 10, worker1, controller1, size1, 300, 40, sql-server

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

26, 2003-11-11, 10, worker2, controller2, size2, black, 20

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

1

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used

12, 2005-11-11, 12, worker1, controller1, size12, 101, 12, vs

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical1, 2006-11-11, controller

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

1

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, sotfware name major used

14, 2006-11-11, 12, worker1, controller1, size12, 101, 14, vs

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

2

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, color

16, 2006-12-11, 12, worker1, controller1, size16, 101, 16, vs

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

2

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, color

21, 2006-12-11, 12, worker2, controller2, size16, 101, 21, yellow

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical2, 2007-11-11, controller

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

2

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, color

22, 2006-12-13, 12, worker2, controller2, size16, 101, 22, black

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

2

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, color

24, 2006-12-15, 12, worker2, controller2, size16, 101, 24, black

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical2, 2007-11-11, complaint

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

3

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, weight

30, 2007-12-13, 12, worker3, controller3, size16, 300, 22, 40kg

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical3, 2008-11-11, complaint

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

3

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, weight

31, 2007-12-13, 12, worker3, controller3, size16, 300, 31, 31kg

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

3

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, weight

32, 2007-12-14, 12, worker3, controller3, size16, 300, 66, 55kg

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

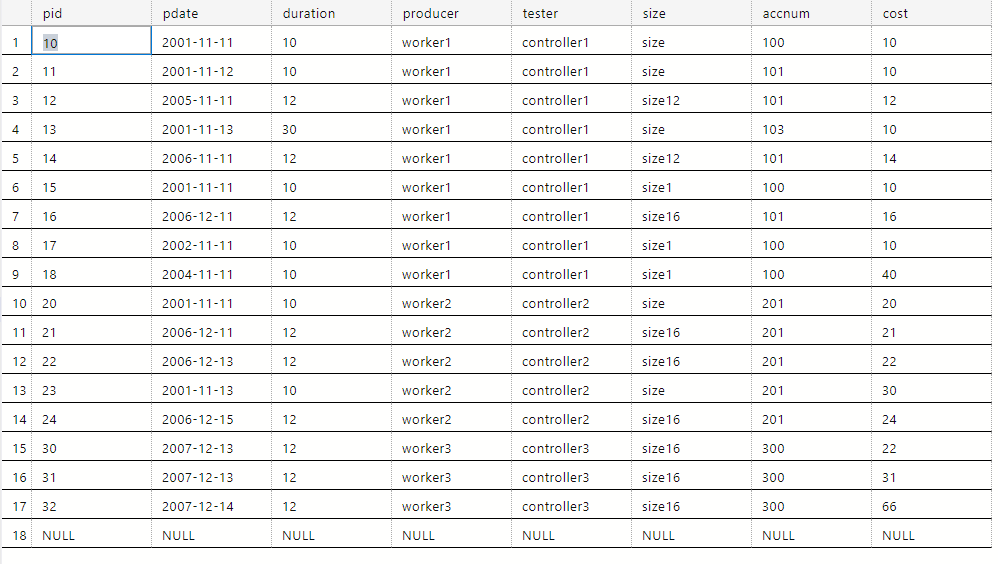
1

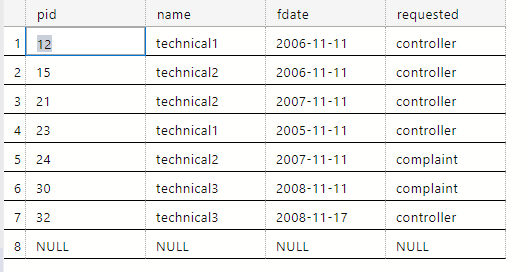
Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical3, 2008-11-17, controller

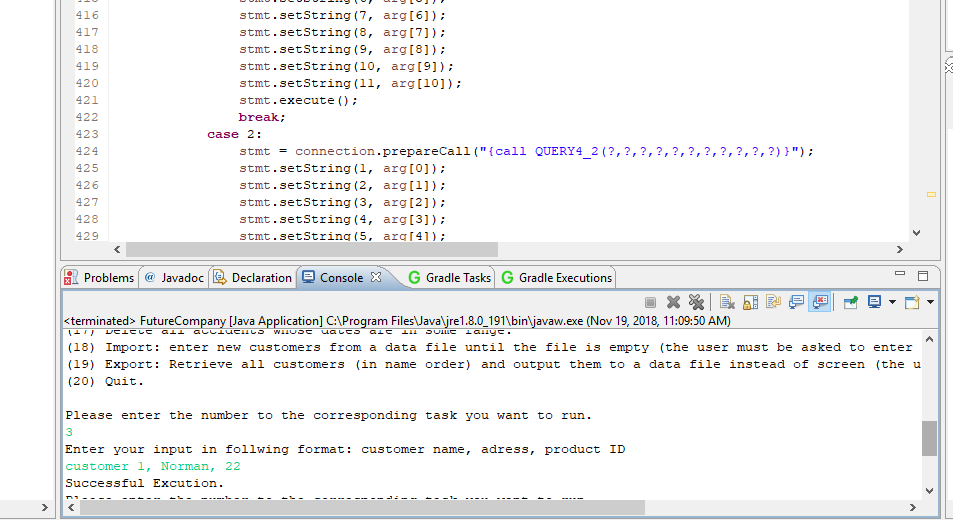
Successful Excution.

Please enter the number to the corresponding task you want to run.





## 6.3. Script file showing the testing of query 3



Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

customer 1, Norman, 22

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

customer 2, Norman, 21

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

customer 3, Norman, 31

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

woker3, Norman, 30

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

customer 4, Norman, 12

Successful Excution.

Enter your input in follwing format: customer name, adress, product ID

worker2, Norman, 24

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

worker3, Norman, 10

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

controller1, Norman, 11

Successful Excution.

Please enter the number to the corresponding task you want to run.

3

Enter your input in follwing format: customer name, adress, product ID

controller3, Norman, 13

Successful Excution.

Please enter the number to the corresponding task you want to run.

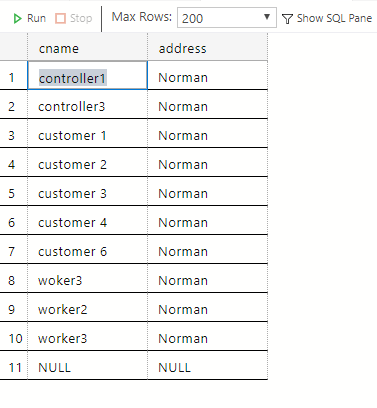
3

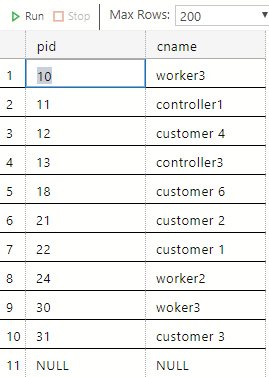
Enter your input in follwing format: customer name, adress, product ID

customer 6, Norman, 18

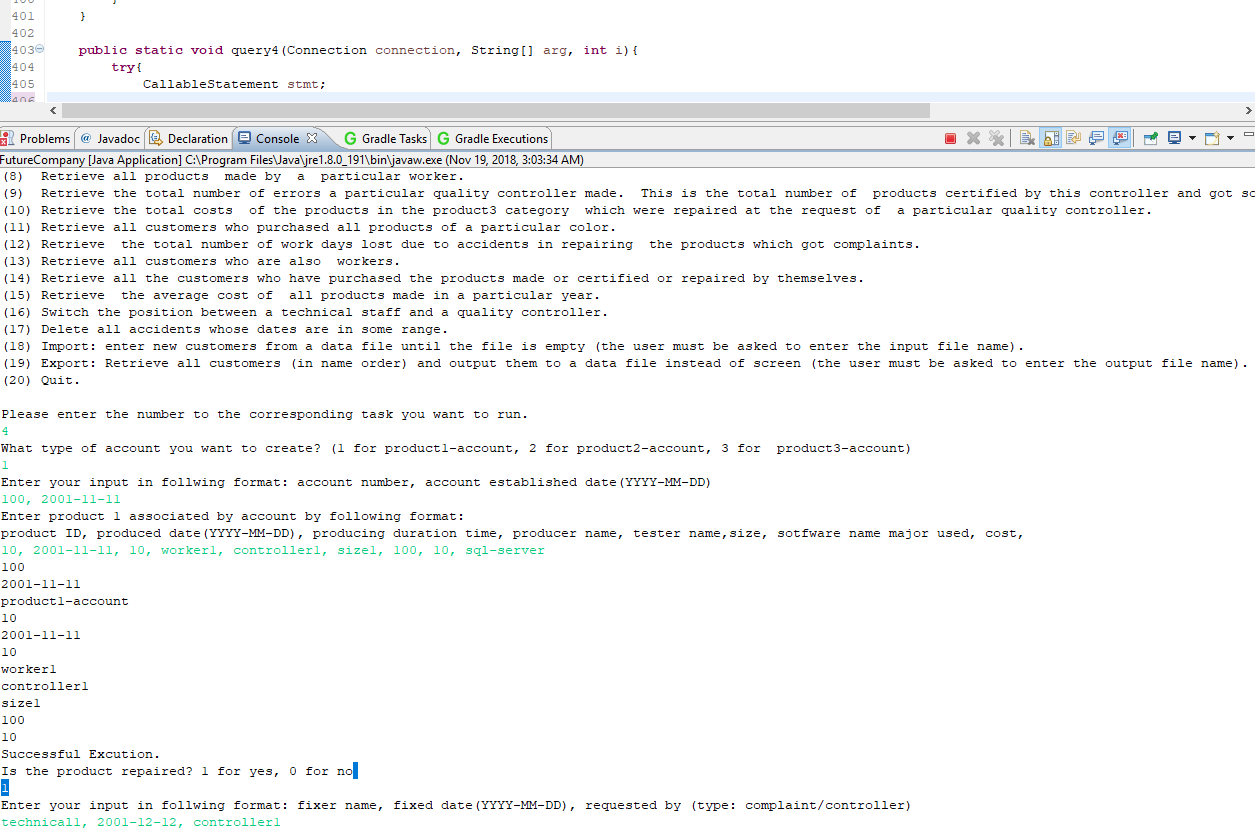
Successful Excution.

Please enter the number to the corresponding task you want to run.





## 6.4. Script file showing the testing of query 4



Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

1

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

103, 2001-11-11

Enter product 1 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, sotfware name major used, cost,

13, 2001-11-13, 30, worker1, controller1, size1, 100, 10, sql-server

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical1, 2005-11-11, comtronller

Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

2

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

200, 2001-11-11

Enter product 2 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, color, cost

10, 2001-11-11, 10, worker1, controller1, size1, 100, 10, sql-server

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

0

Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

2

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

201, 2001-11-11

Enter product 2 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, color, cost

20, 2001-11-11, 10, worker2, controller2, size2, blue, 20

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical1, 2005-11-11, contronller

Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

2

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

203, 2001-11-13

Enter product 2 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name,size, color, cost

23, 2001-11-13, 10, worker2, controller2, size2, green, 30

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical1, 2005-11-11, controller

Successful Excution.

Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

3

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

300, 2001-11-11

Enter product 3 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name, size, weight, cost

30, 2001-11-11, 30, worker3, controller3, size3, 3kg, 20

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical2, 2005-11-11, controller

Please enter the number to the corresponding task you want to run.

4

What type of account you want to create? (1 for product1-account, 2 for product2-account, 3 for product3-account)

3

Enter your input in follwing format: account number, account established date(YYYY-MM-DD)

302, 2001-11-11

Enter product 3 associated by account by following format:

product ID, produced date(YYYY-MM-DD), producing duration time, producer name, tester name, size, weight, cost

32, 2001-11-11, 30, worker3, controller3, size3, 3kg, 20

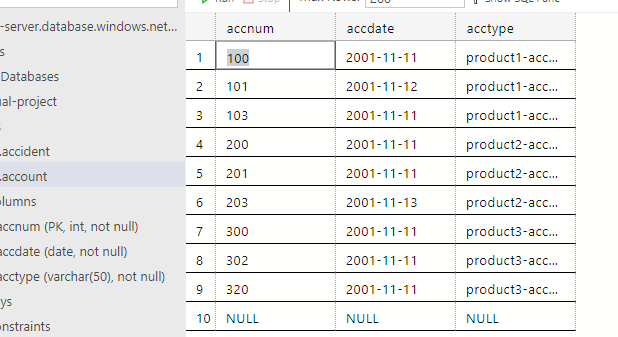
Successful Excution.

Is the product repaired? 1 for yes, 0 for no

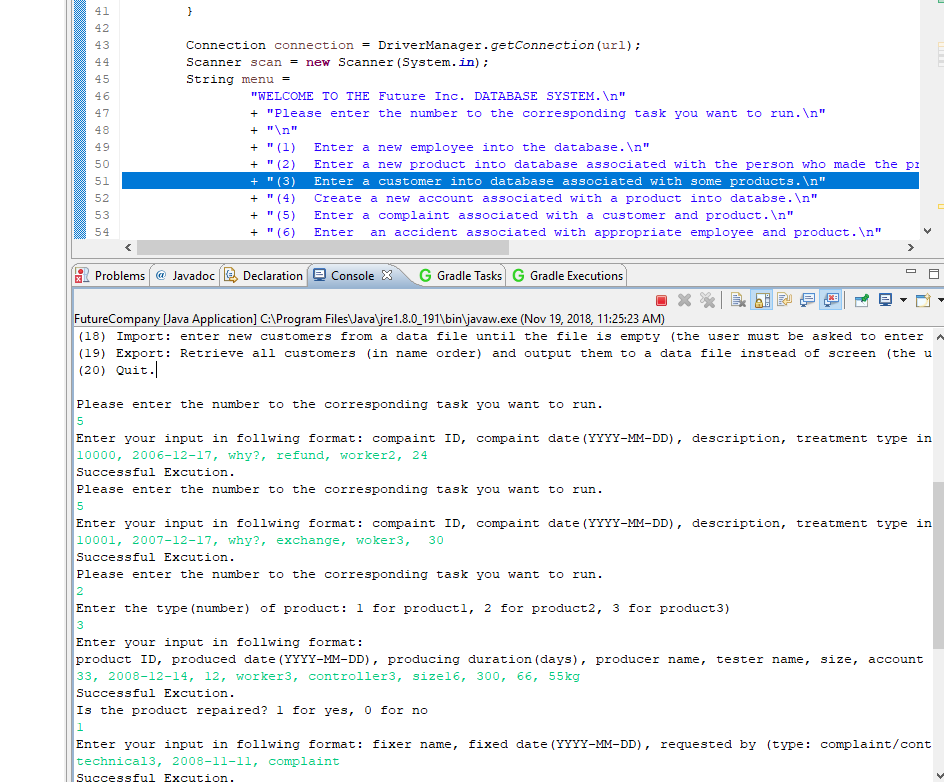
1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical1, 2005-11-11, controller



## 6.5. Script file showing the testing of query5



Please enter the number to the corresponding task you want to run.

5

Enter your input in follwing format: compaint ID, compaint date(YYYY-MM-DD), description, treatment type in (refund/exchange),/n custemer name, product ID

10000, 2006-12-17, why?, refund, worker2, 24

Successful Excution.

Please enter the number to the corresponding task you want to run.

5

Enter your input in follwing format: compaint ID, compaint date(YYYY-MM-DD), description, treatment type in (refund/exchange),/n custemer name, product ID

10001, 2007-12-17, why?, exchange, woker3, 30

Successful Excution.

Please enter the number to the corresponding task you want to run.

2

Enter the type(number) of product: 1 for product1, 2 for product2, 3 for product3)

3

Enter your input in follwing format:

product ID, produced date(YYYY-MM-DD), producing duration(days), producer name, tester name, size, account number, cost, weight

33, 2008-12-14, 12, worker3, controller3, size16, 300, 66, 55kg

Successful Excution.

Is the product repaired? 1 for yes, 0 for no

1

Enter your input in follwing format: fixer name, fixed date(YYYY-MM-DD), requested by (type: complaint/controller)

technical3, 2008-11-11, complaint

Successful Excution.

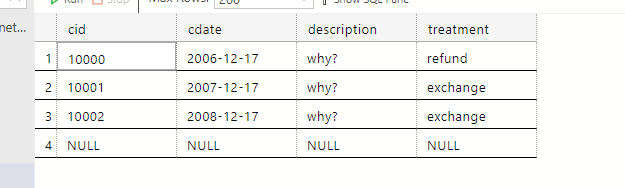
Please enter the number to the corresponding task you want to run.

3

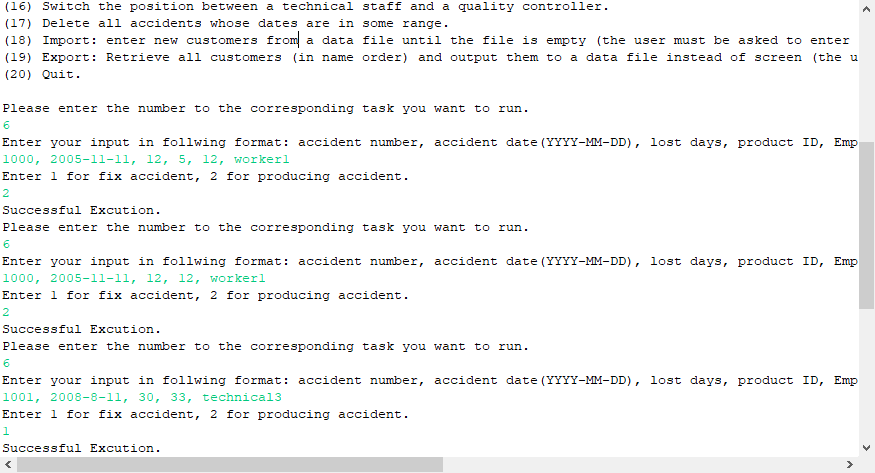
Enter your input in follwing format: customer name, adress, product ID

customer 10, Norman, 33

Successful Excution.



## 6.6 Script file showing the testing of query6



Please enter the number to the corresponding task you want to run.

6

Enter your input in follwing format: accident number, accident date(YYYY-MM-DD), lost days, product ID, Employee name

1000, 2005-11-11, 12, 5, 12, worker1

Enter 1 for fix accident, 2 for producing accident.

2

Successful Excution.

Please enter the number to the corresponding task you want to run.

6

Enter your input in follwing format: accident number, accident date(YYYY-MM-DD), lost days, product ID, Employee name

1000, 2005-11-11, 12, 12, worker1

Enter 1 for fix accident, 2 for producing accident.

2

Successful Excution.

Please enter the number to the corresponding task you want to run.

6

Enter your input in follwing format: accident number, accident date(YYYY-MM-DD), lost days, product ID, Employee name

1001, 2008-8-11, 30, 33, technical3

Enter 1 for fix accident, 2 for producing accident.

1

Please enter the number to the corresponding task you want to run.

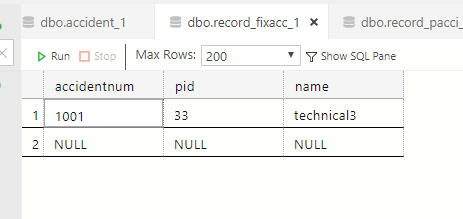
6

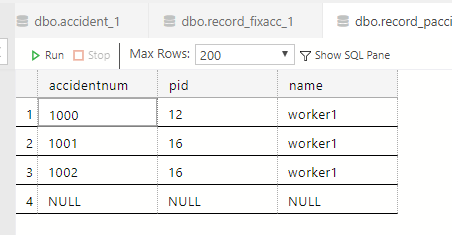
Enter your input in follwing format: accident number, accident date(YYYY-MM-DD), lost days, product ID, Employee name

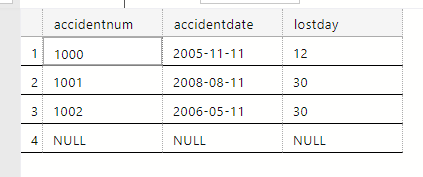
1002, 2006-5-11, 30, 16, worker1

Enter 1 for fix accident, 2 for producing accident.

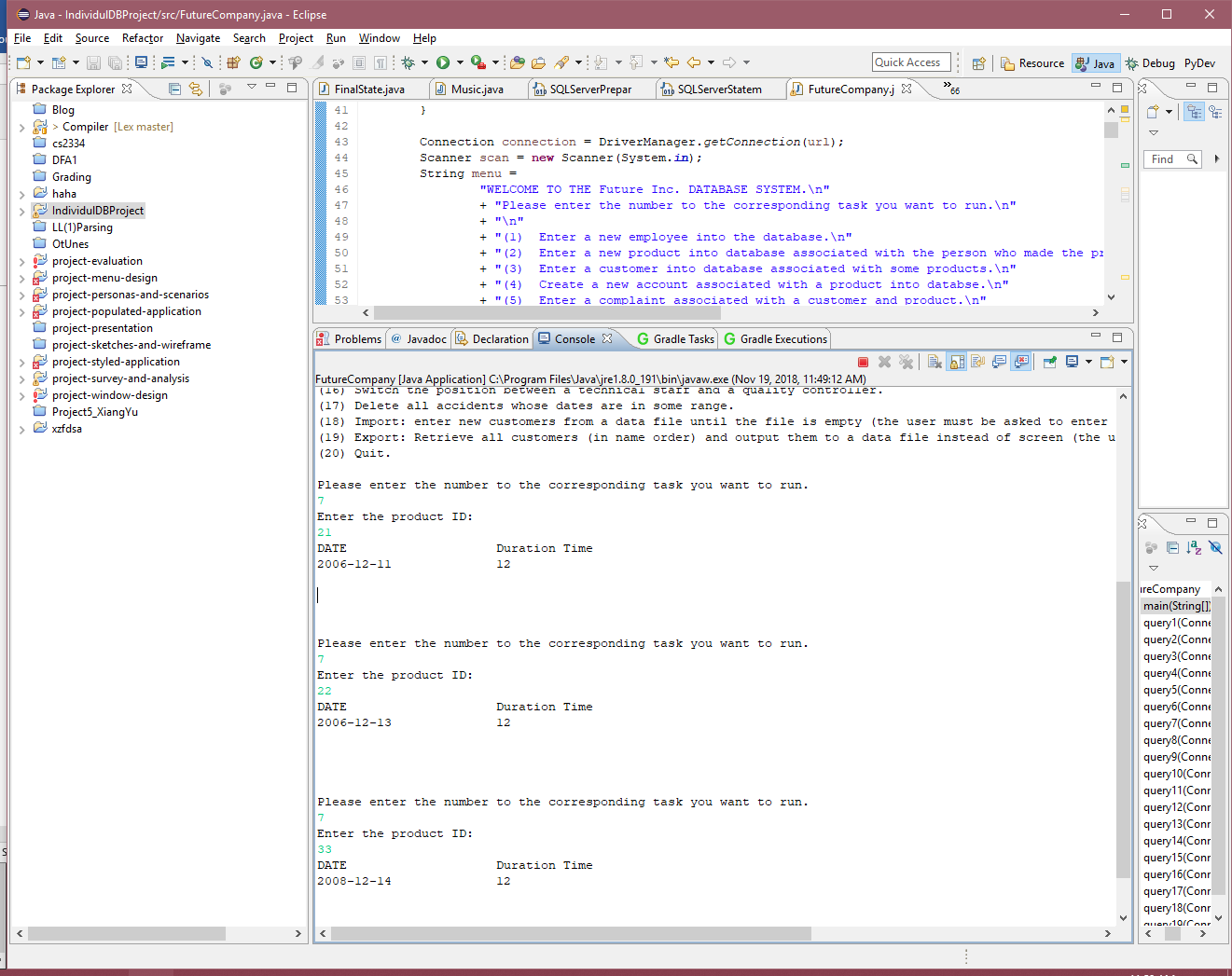
2





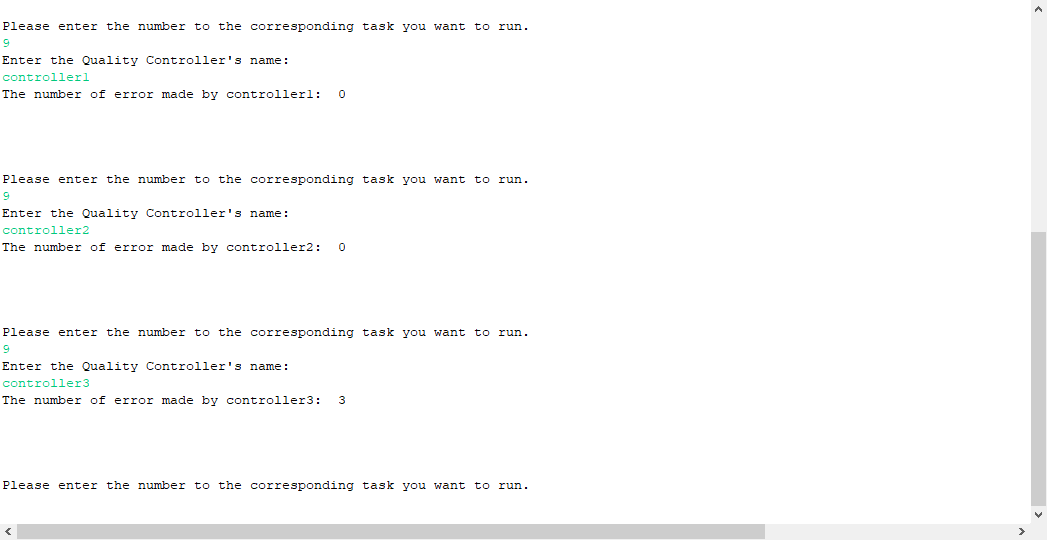


## 6.7 Script file showing the testing of query7

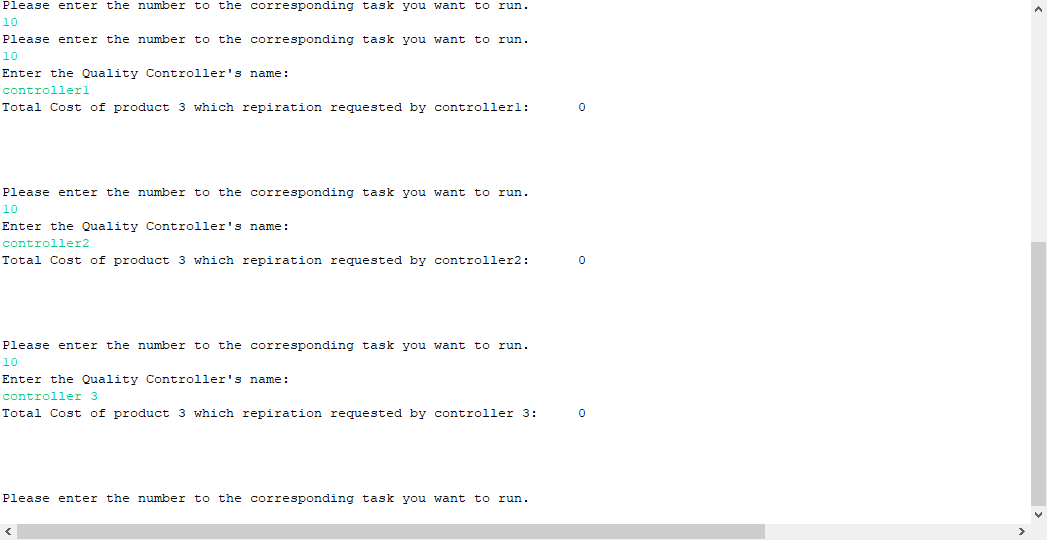


## 6.8 Script file showing the testing of query8

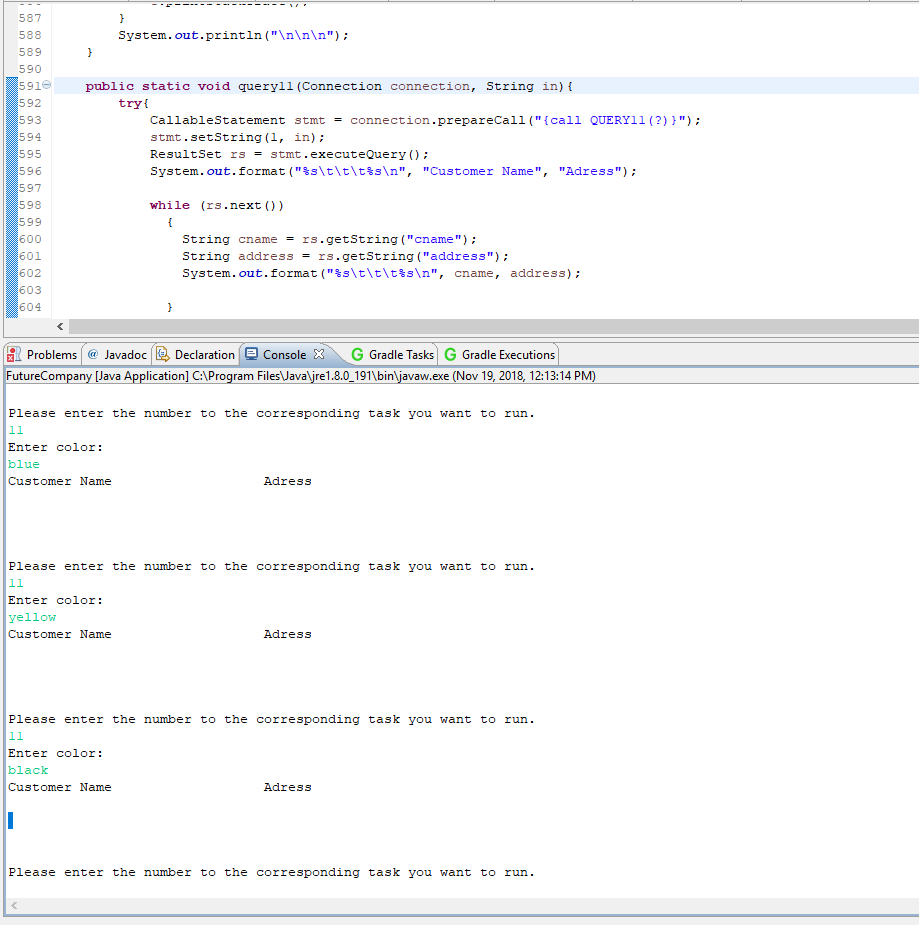
## 6.9 Script file showing the testing of query9



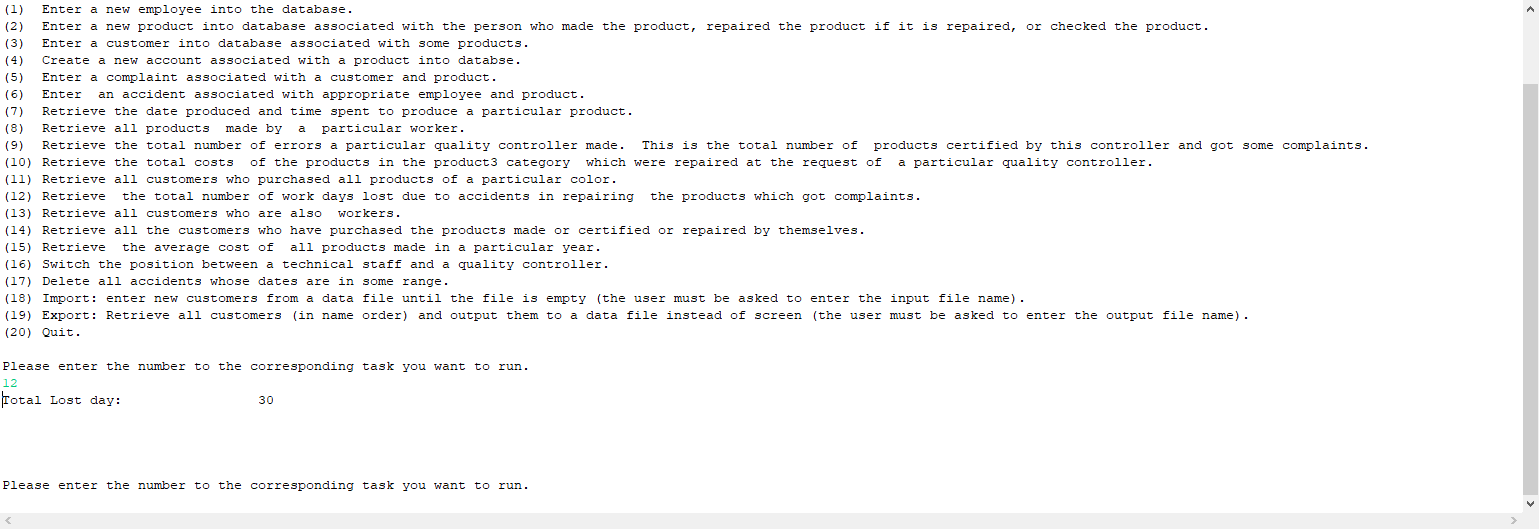
## 6.10Script file showing the testing of query10



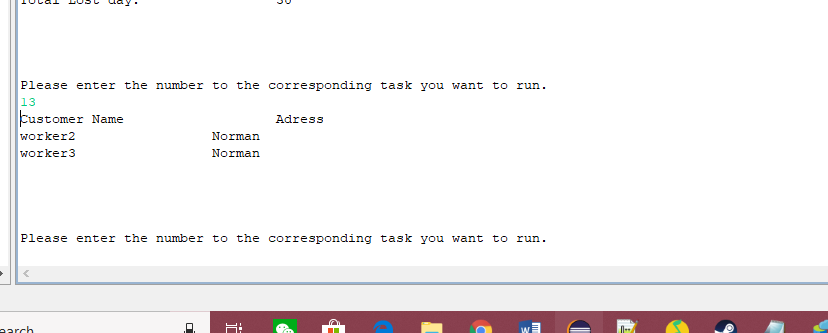
## 6.11Script file showing the testing of query11



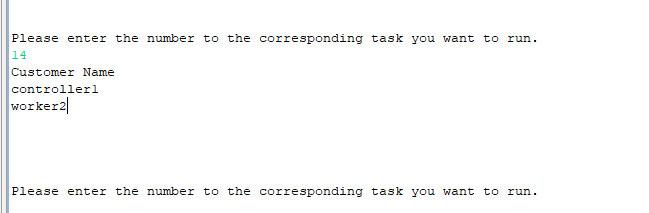
## 6.12Script file showing the testing of query12



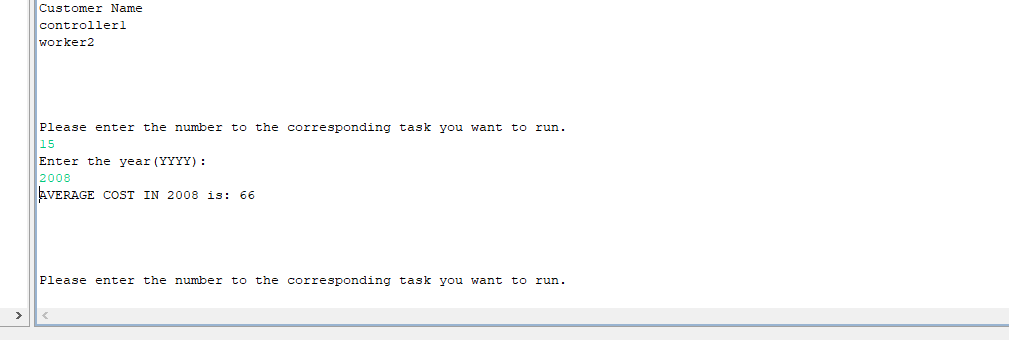
## 6.13Script file showing the testing of query13



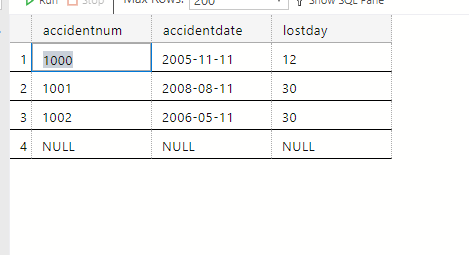
## 6.14Script file showing the testing of query14

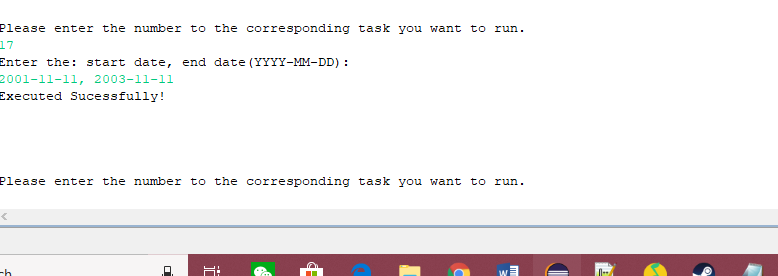


## 6.15Script file showing the testing of query15



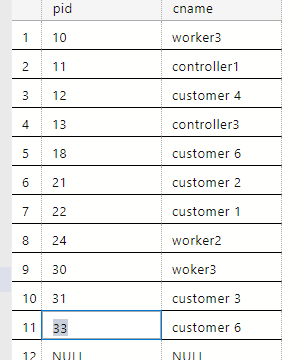
## 6.17Script file showing the testing of query17

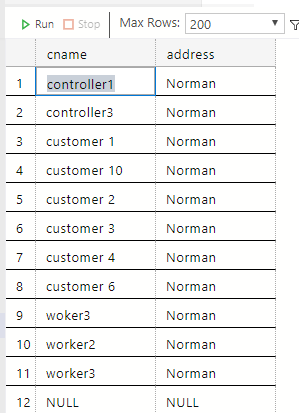




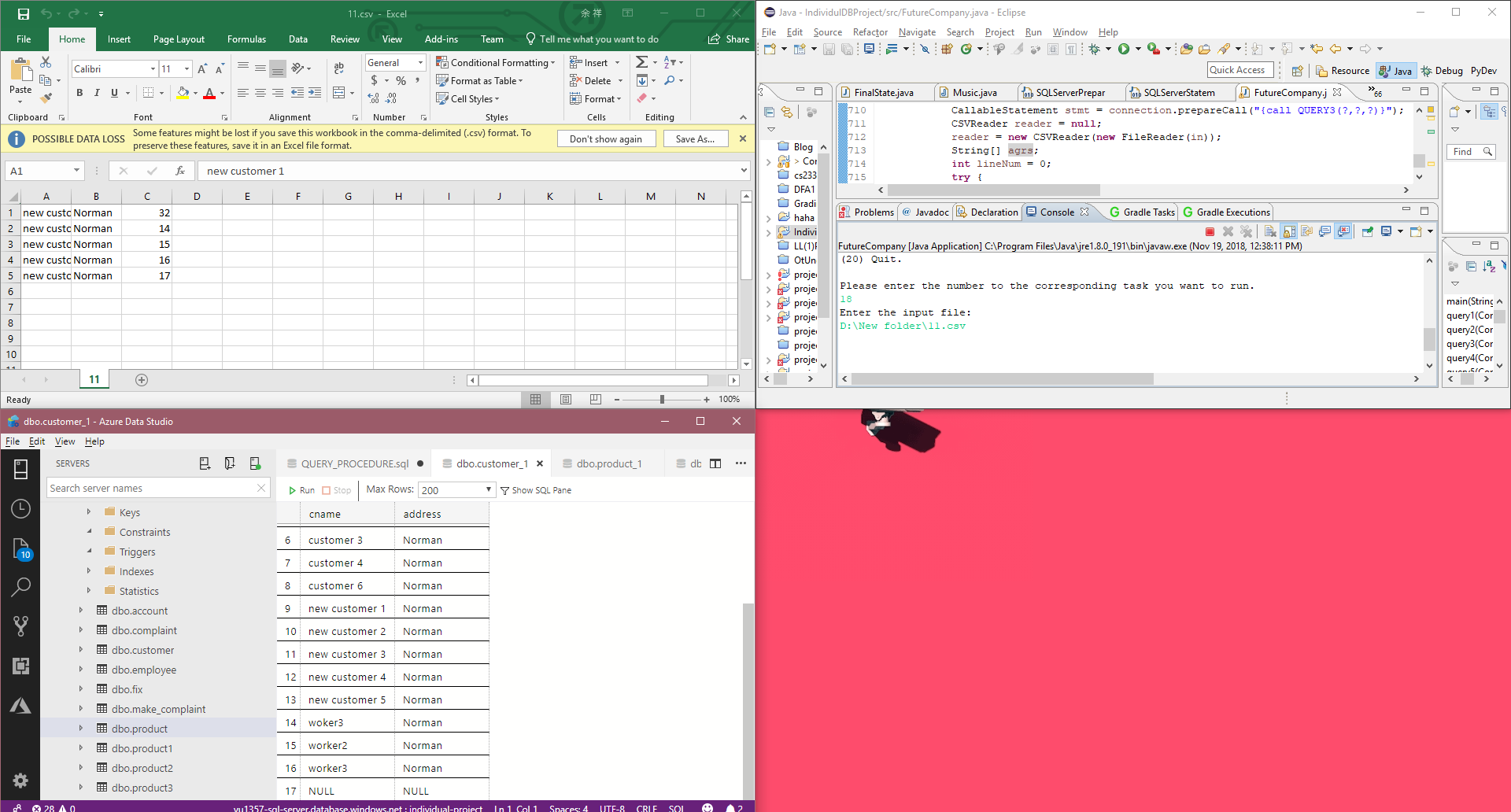
## 6.19Script file showing the testing of query19

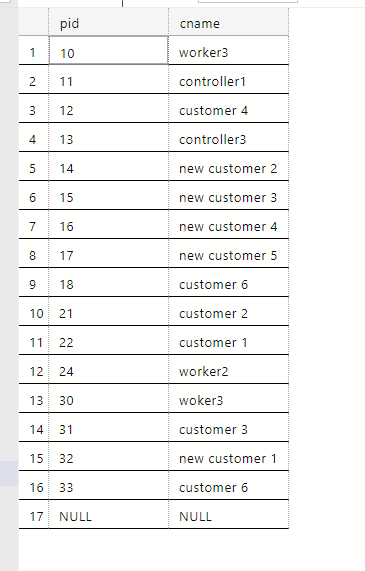
BEFORE:





AFTER:





## 6.19Script file showing the testing of query19

