Course Name: Database Management Systems

Course Number: DSA – 4513

Section Number: 001

Semester and Year: Fall 2019

Instructor Name: Dr. Le Gruenwald

Author Name: Gowtham Teja Kanneganti

Author Id: 113426883

Email-id: gowthamkanneganti@ou.edu

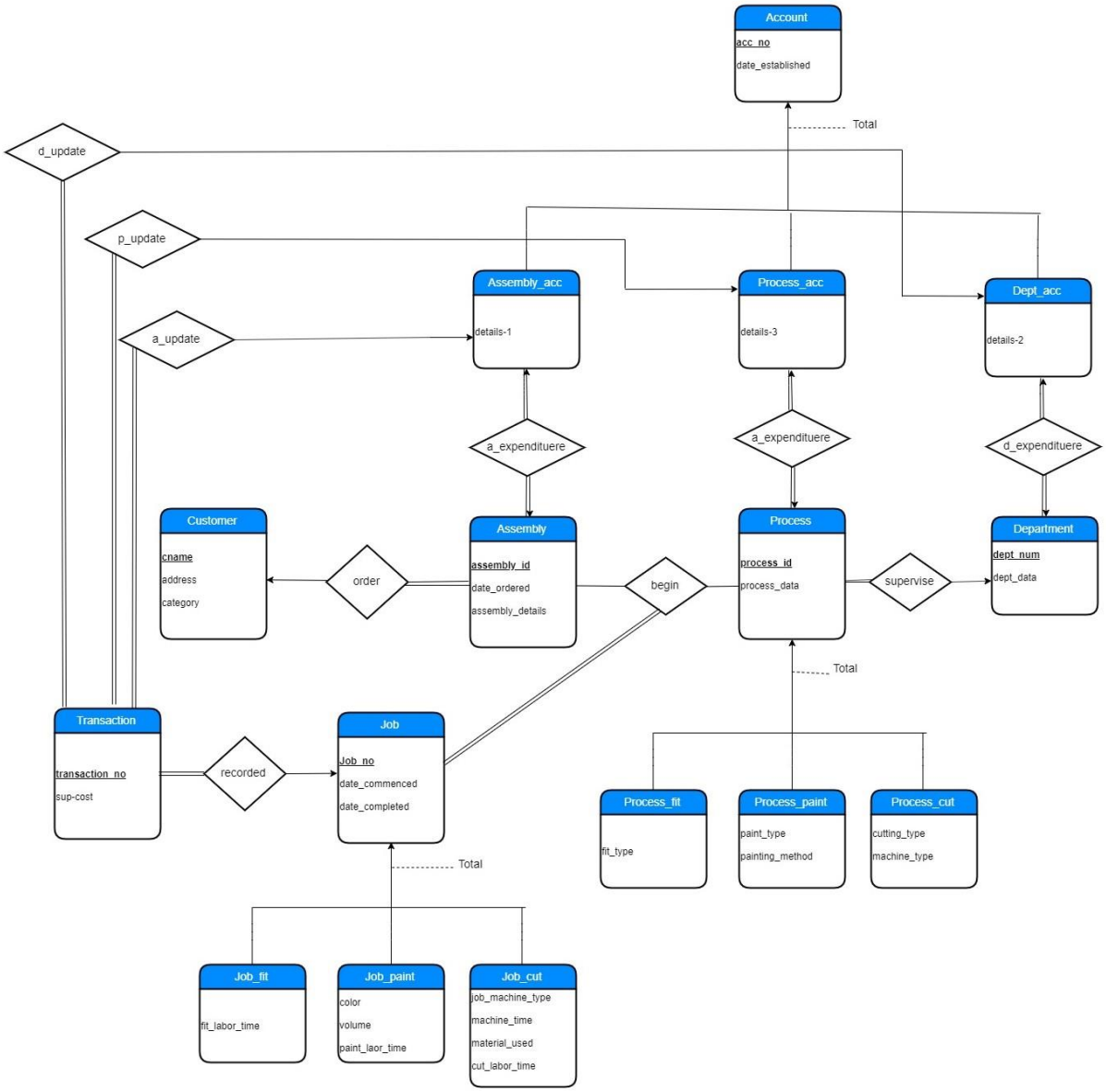
Title: A JOB-SHOP ACCOUNTING SYSTEM

Tasks Performed Page Number

Table of Contents

TASK 1.

1.1. ER Diagram



1.2. Relational Database Schema

Customer (cname, caddress, category)

Assembly (assembly_id, date_ordered, assembly_details, cname)

Department (dept_no, dept_data)

Process (process_id, process_data, dept_no)

Process_fit (process_id, fit_type)

Process_paint (process_id, paint_type, paint_method)

Process_cut (process_id, cutting_type, machine_type)

Job (job_no, date_commenced, date_completed, assembly_id, process_id)

Job_fit (job_no, fit_labor_time)

Job_paint (job_no, color, volume, paint_labor_time)

Job_cut (job_no, job_machine_type, machine_time, material_used, cut_labor_time)

Dept_acc (acc_no, date_established, details_2, dept_no)

Assembly_acc (acc_no, date_established, details_1, assembly_id)

Proecss_acc (acc_no, date_estableshed, details_3, process_id)

Transaction1 (t_no, sup_cost, job_no, dac_no, aacc_no, pacc_no)

TASK 2. Data Dictionary

- For VARCHAR, I am taking the value from the value given in Azure Database documentation. Size = (2 + num chars) in bytes.

Customer Table			
Column Name	Data Type	Size (bytes)	Constraints
cname	VARCHAR (20)	22	PRIMARY KEY
caddress	VARCHAR (100)	102	
category	INT	4	Constraint check between 1 and 10.

Assembly Table			
Column Name	Data Type	Size (bytes)	Constraints
assembly_id	VARCHAR (10)	12	PRIMARY KEY
date_ordered	DATE	3	
assembly_details	VARCHAR (100)	102	
cname	VARCHAR (20)	22	FORREIGN KEY TO Customer

Department Table			
Column Name	Data Type	Size (bytes)	Constraints
dept_no	VARCHAR (7)	9	PRIMARY KEY
dept_data	VARCHAR (100)	102	

Process Table			
Column Name	Data Type	Size (bytes)	Constraints
process_id	VARCHAR (7)	9	PRIMARY KEY
Process_data	VARCHAR (100)	102	
dept_no	VARCHAR (7)	9	FORREIGN KEY TO Department

Process_fit Table			
Column Name	Data Type	Size (bytes)	Constraints
process_id	VARCHAR (7)	9	PRIMARY KEY, FORREIGN KEY TO Process
fit_type	VARCHAR (5)	7	

Process_paint Table			
Column Name	Data Type	Size (bytes)	Constraints
process_id	VARCHAR (7)	9	PRIMARY KEY, FORREIGN KEY TO Process
paint_type	VARCHAR (6)	8	
Paint_method	VARCHAR (8)	10	

Process_cut Table			
Column Name	Data Type	Size (bytes)	Constraints
process_id	VARCHAR (7)	9	PRIMARY KEY, FORREIGN KEY TO Process
cutting_type	VARCHAR (7)	9	
machine_type	VARCHAR (9)	11	

Job Table			
Column Name	Data Type	Size (bytes)	Constraints
job_no	INT	4	PRIMARY KEY
date_commenced	DATE	3	
date_completed	DATE	3	
assembly_id	VARCHAR (10)	12	FORREIGN KEY TO Assembly
process_id	VARCHAR (7)	9	FOREIGN KEY TO Process

Job_fit Table			
Column Name	Data Type	Size (bytes)	Constraints
job_no	INT	4	PRIMARY KEY, FOREIGN KEY TO Job
fit_labor_time	TIME	3	

Job_paint Table			
Column Name	Data Type	Size (bytes)	Constraints
job_no	INT	4	PRIMARY KEY, FOREIGN KEY TO Job
color	VARCHAR (10)	12	
volume	INT	4	
pait_labor_time	TIME	3	

Job_cut Table			
Column Name	Data Type	Size (bytes)	Constraints
job_no	INT	4	PRIMARY KEY, FOREIGN KEY TO Job
job_machine_type	VARCHAR (10)	12	
machine_time	TIME	3	
material_used	VARCHAR (3)	5	
pait_labor_time	TIME	3	

Dept_acc Table			
Column Name	Data Type	Size (bytes)	Constraints
acc_no	INT	4	PRIMARY KEY
date_established	DATE	3	
details_2	FLOAT	8	
dept_no	VARCHAR (7)	9	FROREIGN KEY TO Department, UNIQUE constraint

Assembly_acc Table			
Column Name	Data Type	Size (bytes)	Constraints
acc_no	INT	4	PRIMARY KEY
date_established	DATE	3	
details_1	FLOAT	8	
assembly_id	VARCHAR (10)	12	FROREIGN KEY TO Assembly, UNIQUE constraint

Process_acc Table			
Column Name	Data Type	Size (bytes)	Constraints
acc_no	INT	4	PRIMARY KEY
date_established	DATE	3	
details_3	FLOAT	8	
process_id	VARCHAR (7)	9	FROREIGN KEY TO Department, UNIQUE constraint

Transaction1 Table			
Column Name	Data Type	Size (bytes)	Constraints
t_no	VARCHAR (10)	12	PRIMARY KEY
sup_cost	FLOAT	8	
job_no	INT	4	FROREIGN KEY TO Job
dacc_no	INT	4	FROREIGN KEY TO Department_acc
aacc_no	INT	4	FOREIGN KEY TO Assembly_acc
Pacc_no	INT	4	FOREIGN KEY TO Process_acc

TASK 3.

3.1 Discussion of storage structures for tables

Table Name	Query # and Type	Search Key	Query Frequency	Selected File Organization	Justifications
Customer	#1 Insertion		30 / day	B^+ Tree with Index = category	Since there is range search on category and B^+ Tree is preferred for range search
	#13 Range Search	Category	100 / day		
Assembly	#3 Insertion		40 / day	Heap File	Heap file is good for only Insertion
Process	#4 Insertion		Infrequent	Dynamic Extendable Hashing with hash key (Process_id)	Dynamic hashing is preferred, since Random search is more frequent. Process_id is chosen as hash key because Random search on process_id is more frequent.
	#8 Random Search	Process_id Dept-no	50 / day		
	#10 Random Search	Dept_no	20 / day		
	#11 Random Search	Process_id	100 / day		
	#12 Random Search	Proces_id	20 / day		
Process_fit	#4 insertion		Infrequent	Heap File	Heap file structure is preferred for only insertions.
Process_paint	#4 Insertion		Infrequent	Heap File	Heap file structure is preferred for only insertions.
Process_cut	#4 Insertion		Infrequent	Heap File	Heap file structure is

					preferred for only insertions.
Department	#2 Insertion		Infrequent	Heap File	Heap file structure is preferred for only insertions.
Job	#6 Insertion		50 / day	Dynamic Extendable Hashing with hash key (Job_no).	Dynamic hashing is preferred, since Random search is more frequent. Job_no is chosen as hash key because Random search on job_no is more frequent.
	#7 Random Search	Job_no	50 / day		
	#8 Random Search	Job_no	50 / day		
	#10 Random search	Process_id Date_completed	20 /day		
	#11 Random Search	Assembly_id	100 / day		
	#12 Random Search	Date_completed	20 / day		
	#14 Random Search	Job_no	1 / month		
Job_fit	#7 Insert		50 / day	Dynamic Extendable Hashing with hash key (Job_no).	Although there is insertion, I am taking Dynamic hashing because I feel we should be more concerned about random search.
	#10 Random Search	Job_no	20 /day		
	#12 Random Search	Job_no	20 / day		
Job_paint	#7 Insertion		50 / day	Dynamic Extendable	Although there is

	#10 Random Search	Job_no	20 / day	Hashing with hash key (Job_no).	insertion, I am taking Dynamic hashing because I feel we should be more concerned about random search.
	#12 Random Search	Job_no	20 / day		
	#15 Random Search	Job_no	1 / week		
Job_cut	#7 Insertion		50 / day	B^+ Tree with Index = job_no	Although less frequent there is range search on job_no. So, I am preferring B^+ Tree.
	#10 Random Search	Job_no	20 / day		
	#12 Random Search	Job_no	20 /day		
	#14 Range Search	Job_no	1 / month		
Dept_acc	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (Aacc_no).	Dynamic hashing is preferred, since Random search is more frequent. I am choosing hash key on account-no as this is the Primary key.
	#8 Random Search	Dept_no	50 / day		
	#8 random Search (update)	dacc_no	50 / day		
Assembly_accc	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (dacc_no).	Dynamic hashing is preferred, since Random search is more frequent. I am choosing hash key on
	#8 Random Search	Assembly_id	50 / day		
	#8 Random Search (update)	Aacc_no	50 / day		
	#9 Random Search	Assembly_id	200 / day		

					account-no as this is the Primary key.
Process_acc	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (pacc_no).	Dynamic hashing is preferred, since Random search is more frequent. I am choosing hash key on account-no as this is the Primary key.
	#8 Random Search	Process_id	50 / day		
	#8 Random Search (update)	pacc_no	50 / day		
Transaction1	#8 Insertion		50 / day	Heap File	Heap file structure is preferred for only insertions.

3.2 Discussion of storage structures for tables (Azure SQL Database)

Azure SQL by default creates a clustered index on primary key when primary key constraint is defined on a table. It is not possible to create more than one clustered index on a table. Most of my tables uses dynamic hashing. But hash indexes in azure can be created only in a memory optimized tables.

Based on my understanding from section 3.1 and above statements, I have decided to go with the default primary indexes on each table and create additional non – clustered indexes on the following:

Table Name	Index Key
Customer	Category
Process	Dept_no
Job	Assembly_id
Dept_acc	Dept_no
Assembly_acc	Assembly_id
Process_acc	Process_id

TASK 4. SQL Statements and screenshots showing the creation of tables in Azure SQL Database

1. Create Customer table

```
-- Create Customer table
CREATE TABLE Customer (
    cname VARCHAR(20),
    caddress VARCHAR(100),
    category INT,
    CONSTRAINT category_ck CHECK (category BETWEEN 1 AND 10),
    PRIMARY KEY (cname)
);
```

```
Messages
    11:46:13 AM    Started executing query at Line 5
                   Commands completed successfully.
                   Total execution time: 00:00:00.032
```

Creating Index on Category:

```
-- Create index on category - customer table
CREATE INDEX customer_category ON customer(category);
```

```
Messages
    2:11:23 PM     Started executing query at Line 136
                   Commands completed successfully.
                   Total execution time: 00:00:00.044
```

2. Create Assembly table

```
-- Create Assembly table
CREATE TABLE Assembly (
    assembly_id VARCHAR(10), --aid01, aid02, ... ICREASE size based on   query frequency
    date_ordered DATE,
    assembly_details VARCHAR(100),
    cname VARCHAR(20),
    PRIMARY KEY (assembly_id),
    FOREIGN KEY (cname) REFERENCES Customer(cname)
);
```

```
Messages
    12:43:02 PM    Started executing query at Line 13
                   Commands completed successfully.
                   Total execution time: 00:00:00.041
```

3. Create Department Table

```sql
--Create Department Table
CREATE TABLE Department (
    dept_no VARCHAR(7), -- dept1, dept2, ...
    dept_data VARCHAR(100),
    PRIMARY KEY (dept_no)
);
```

```
Messages
    12:45:28 PM    Started executing query at Line 23
                   Commands completed successfully.
                   Total execution time: 00:00:00.032
```

4. Create Process Table

```sql
--Create Process Table
CREATE TABLE Process (
    process_id VARCHAR(7) PRIMARY KEY, -- proc1, proc2, ...
    process_data VARCHAR(100),
    dept_no VARCHAR(7) FOREIGN KEY REFERENCES Department(dept_no)
);
```

```
Messages
    12:46:48 PM    Started executing query at Line 30
                   Commands completed successfully.
                   Total execution time: 00:00:00.062
```

Creating Index on dept_no:

```
-- Create index on dept_no - Process table
CREATE INDEX process_dept ON process(dept_no);
```

```
Messages
    2:13:31 PM    Started executing query at Line 139
                  Commands completed successfully.
                  Total execution time: 00:00:00.075
```

5. Create Process_fit Table

```
--Create Process_fit Table
CREATE TABLE Process_fit (
    process_id VARCHAR(7) FOREIGN KEY REFERENCES Process(process_id),
    fit_type VARCHAR(5), --fit01, fit02, ...
    PRIMARY KEY (process_id)
);
```

```
Messages
    12:48:21 PM    Started executing query at Line 37
                   Commands completed successfully.
                   Total execution time: 00:00:00.043
```

6. Create Process_paint Table

```
--Create Process_paint Table
CREATE TABLE Process_paint (
    process_id VARCHAR(7) FOREIGN KEY REFERENCES Process(process_id),
    paint_type VARCHAR(6), -- type01, type02,...
    paint_method VARCHAR(8), -- method01, method02, ...
    PRIMARY KEY (process_id)
);
```

```
Messages
    12:50:25 PM    Started executing query at Line 44
                   Commands completed successfully.
                   Total execution time: 00:00:00.038
```

7. Create Process_cut Table

```
--Create Process_cut Table
CREATE TABLE Process_cut (
    process_id VARCHAR(7) FOREIGN KEY REFERENCES Process(process_id),
    cutting_type VARCHAR(7), --ctype01, ctype02, ...
    machine_type VARCHAR(9), --machine01, machine02, ...
    PRIMARY KEY (process_id)
);
```

```
Messages
Commands completed successfully.
Total execution time: 00:00:00
```

8. Create Job Table

```
--Create Job Table
CREATE TABLE Job (
    job_no INT PRIMARY KEY, -- job01, job02, ...
    date_commenced DATE,
    date_completed DATE,
    assembly_id VARCHAR(10) FOREIGN KEY REFERENCES Assembly(assembly_id),
    process_id VARCHAR(7) FOREIGN key REFERENCES Process(process_id)
);
```

```
Messages
6:12:38 PM    Started executing query at Line 60
              Commands completed successfully.
              Total execution time: 00:00:00.071
```

Creating Index on assembly_id:

```
-- Create index on assembly_id - Job table
CREATE INDEX assembly_job ON job(assembly_id);
```

```
Messages
    2:15:06 PM      Started executing query at Line 142
                    Commands completed successfully.
                    Total execution time: 00:00:00.061
```

9. Create Job_fit Table

```
--Create Job_fit Table
CREATE TABLE Job_fit (
    job_no INT FOREIGN KEY REFERENCES Job(job_no),
    fit_labor_time TIME,
    PRIMARY KEY (job_no)
);
```

```
Messages
    6:15:06 PM      Started executing query at Line 68
                    Commands completed successfully.
                    Total execution time: 00:00:00.051
```

10. Create Job_paint Table

```
--Create Job_paint Table
CREATE TABLE Job_paint (
    job_no INT FOREIGN KEY REFERENCES Job(job_no),
    color VARCHAR(10),
    volume INT, -- Assuming Volume of paint used will always be calculated to nearest integer
    paint_labor_time TIME,
    PRIMARY KEY (job_no)
);
```

```
Messages
    6:16:26 PM      Started executing query at Line 76
                    Commands completed successfully.
                    Total execution time: 00:00:00.091
```

11. Create Job_cut Table

```
--Create Job_cut Table
CREATE TABLE Job_cut (
    job_no INT FOREIGN KEY REFERENCES Job(job_no),
    job_machine_type VARCHAR(10), --jmachine01, jmachine02, ...
    machine_time TIME,
    material_used VARCHAR(3), --m01, m02, ...
    cut_labor_time TIME,
    PRIMARY KEY (job_no)
);
```

```
Messages
    6:17:18 PM      Started executing query at Line 85
                    Commands completed successfully.
                    Total execution time: 00:00:00.101
```

12. Create Department Account Table

```
--Create Department Account Table
CREATE TABLE Dept_acc (
    acc_no INT PRIMARY KEY,
    date_established DATE,
    details_2 FLOAT,
    dept_no VARCHAR(7) FOREIGN KEY REFERENCES Department(dept_no) UNIQUE
);
```

```
Messages
    6:18:49 PM      Started executing query at Line 95
                    Commands completed successfully.
                    Total execution time: 00:00:00.629
```

Creating Index on dept_no:

```
-- Create index on dept_no - Dept_acc table
CREATE INDEX dept_no_acc ON Dept_acc(dept_no);
```

```
Messages
    2:20:29 PM      Started executing query at Line 145
                    Commands completed successfully.
                    Total execution time: 00:00:00.061
```

13. Create Assembly Account

```
--Create Assembly Account
CREATE TABLE Assembly_acc (
    acc_no INT PRIMARY KEY,
    date_established DATE,
    details_1 FLOAT,
    assembly_id VARCHAR(10) FOREIGN KEY REFERENCES Assembly(assembly_id) UNIQUE
);
```

```
Messages
    6:19:52 PM      Started executing query at Line 103
                    Commands completed successfully.
                    Total execution time: 00:00:00.044
```

Creating Index on assembly_id:

```
-- Create index on assembly_id - Assembly_acc table
CREATE INDEX assembly_id_acc ON Assembly_acc(assembly_id);
```

```
Messages
    2:21:02 PM      Started executing query at Line 148
                    Commands completed successfully.
                    Total execution time: 00:00:00.042
```

14. Create Process Account

```
--Create Process Account
CREATE TABLE Process_acc (
    acc_no INT PRIMARY KEY,
    date_established DATE,
    details_3 FLOAT,
    process_id VARCHAR(7) FOREIGN KEY REFERENCES Process(process_id) UNIQUE
);
```

```
Messages
    6:20:33 PM      Started executing query at Line 111
                    Commands completed successfully.
                    Total execution time: 00:00:00.306
```

Creating Index on process_id:

```
-- Create index on process_id - process_acc table
CREATE INDEX process_id_acc ON process_acc(process_id);
```

15. Create Transaction Table

```
--Create Transaction Table
CREATE TABLE Transaction1 ( -- Added '1' because transaction is a keyword
    t_no VARCHAR(10) PRIMARY KEY,
    sup_cost FLOAT,
    job_no INT FOREIGN KEY REFERENCES Job(job_no),
    dacc_no INT FOREIGN KEY REFERENCES Dept_acc(acc_no),
    aacc_no INT FOREIGN KEY REFERENCES Assembly_acc(acc_no),
    pacc_no INT FOREIGN KEY REFERENCES Process_acc(acc_no)
);
```

```
Messages
    6:24:29 PM      Started executing query at Line 119
                    Commands completed successfully.
                    Total execution time: 00:00:00.064
```

Task 5

5.1 SQL statements for all queries (1 - 15) defined in part 1.

1. Enter a new Customer

```
CREATE PROCEDURE proc_1
    @cname VARCHAR(20),
    @caddress VARCHAR(100),
    @category INT
AS
BEGIN

INSERT INTO Customer
( -- Columns to insert data into
 [cname], [caddress], [category]
)
VALUES (@cname, @caddress, @category)
END
```

2. Enter a new Department

```
-- Enter a new Department

CREATE PROCEDURE proc_2

    @dept_no VARCHAR(7),

    @dept_data VARCHAR(100)

AS

BEGIN

    INSERT INTO Department

    ( -- Columns to insert data into

    [dept_no], [dept_data]

    )

    VALUES (@dept_no, @dept_data)

END
```

3. Enter a new assembly with its customer-name, assembly-details, assembly-id, and date-ordered

```
--Enter a new assembly with its customer-name, assembly-details, assembly-id,
-- and date-ordered
CREATE PROCEDURE proc_3
    @assembly_id VARCHAR(10),
    @data_ordered VARCHAR(10),
    @assembly_details VARCHAR(100),
    @cname1 VARCHAR(20)
AS
BEGIN
    INSERT INTO Assembly
    ( -- Columns to insert data into
    [assembly_id], [date_ordered], [assembly_details], [cname]
    )
    VALUES (@assembly_id, CAST (@data_ordered as DATE), @assembly_details, @cname1)
END
```

4. Enter a new process-id and its department together with its type and information relevant to the type

```
--Enter a new process-id and its department together
--with its type and information relevant to the type
CREATE PROCEDURE proc_4
    @process_id VARCHAR(7),
    @process_data VARCHAR(100),
    @dept_no1 VARCHAR(7)
AS
BEGIN
    INSERT INTO Process
    ( -- Columns to insert data into
    [process_id], [process_data], [dept_no]
    )
    VALUES (@process_id, @process_data, @dept_no1)
END
```

To insert data if it is fit-type:

```
CREATE PROCEDURE proc_4_1
    @process_id VARCHAR(7),
    @fit_type VARCHAR(5)
AS
BEGIN
    INSERT INTO Process_fit
    ( -- Columns to insert data into
    [process_id], [fit_type]
    )
    VALUES (@process_id, @fit_type)
END
```

To insert data if it is paint-type:

```
CREATE PROCEDURE proc_4_2
    @process_id VARCHAR(7),
    @paint_type VARCHAR(6),
    @paint_method VARCHAR(8)
AS
BEGIN
    INSERT INTO Process_paint
    ( -- Columns to insert data into
    [process_id], [paint_type], [paint_method]
    )
    VALUES (@process_id, @paint_type, @paint_method)
END
```

To insert data if it is cut-type:

```
CREATE PROCEDURE proc_4_3
    @process_id VARCHAR(7),
    @cutting_type VARCHAR(7),
    @machine_type VARCHAR(9)
AS
BEGIN
    INSERT INTO Process_cut
    ( -- Columns to insert data into
    [process_id], [cutting_type], [machine_type]
    )
    VALUES (@process_id, @cutting_type, @machine_type)
END
```

5. Create a new account and associate it with the process, assembly, or department to which it is applicable

For Department Account:

```
-- Create a new account and associate it with the
-- process, assembly, or department to which it is applicable
CREATE PROCEDURE proc_5_1
    @acc_no INT,
    @date_established VARCHAR(10),
    @details_2 FLOAT,
    @dept_no VARCHAR(7)
AS
BEGIN
    INSERT INTO Dept_acc
    ( -- Columns to insert data into
    [acc_no], [date_established], [details_2], [dept_no]
    )
    VALUES (@acc_no, CAST(@date_established as DATE), @details_2, @dept_no)
END
```

For Assembly Account:

```
CREATE PROCEDURE proc_5_2
    @acc_no INT,
    @date_established VARCHAR(10),
    @details_1 FLOAT,
    @assembly_id VARCHAR(10)
AS
BEGIN
    INSERT INTO Assembly_acc
    ( -- Columns to insert data into
    [acc_no], [date_established], [details_1], [assembly_id]
    )
    VALUES (@acc_no, CAST(@date_established as DATE), @details_1, @assembly_id)
END
```

For Process account:

```
CREATE PROCEDURE proc_5_3
    @acc_no INT,
    @date_established VARCHAR(10),
    @details_3 FLOAT,
    @process_id VARCHAR(7)
AS
BEGIN
    INSERT INTO Process_acc
    ( -- Columns to insert data into
    [acc_no], [date_established], [details_3], [process_id]
    )
    VALUES (@acc_no, CAST(@date_established as DATE), @details_3, @process_id)
END
```

6. Enter a new job, given its job-no, assembly-id, process-id, and date the job commenced

```
-- Enter a new job, given its job-no, assembly-id, process-id, and date the job commenced
CREATE PROCEDURE proc_6
    @job_no INT,
    @date_commenced VARCHAR(10),
    @assembly_id VARCHAR(10),
    @process_id VARCHAR(7)
AS
BEGIN
    INSERT INTO Job
    ( -- Columns to insert data into
    [job_no], [date_commenced], [assembly_id], [process_id]
    )
    VALUES (@job_no, CAST(@date_commenced as DATE), @assembly_id, @process_id)
END
```

7. At the completion of a job, enter the date it completed and the information relevant to the type of job

```
-- At the completion of a job, enter the date it completed
--and the information relevant to the type of job
CREATE PROCEDURE proc_7
    @job_no INT,
    @date_completed VARCHAR(10)


AS
BEGIN
    UPDATE Job SET date_completed = @date_completed WHERE job_no = @job_no
END
```

Insert in to fit job:

```
CREATE PROCEDURE proc_7_1
    @job_no INT,
    @fit_labor_time VARCHAR(8)
AS
BEGIN
    INSERT INTO Job_fit
    ( -- Columns to insert data into
    [job_no], [fit_labor_time]
    )
    VALUES (@job_no, CAST(@fit_labor_time as TIME))
END
```

Insert into paint job:

```
CREATE PROCEDURE proc_7_2
    @job_no INT,
    @color VARCHAR(10),
    @volume INT,
    @paint_labor_time VARCHAR(8)
AS
BEGIN
    INSERT INTO Job_paint
    ( -- Columns to insert data into
    [job_no], [color], [volume], [paint_labor_time]
    )
    VALUES (@job_no, @color, @volume, CAST(@paint_labor_time as TIME))
END
```

Insert into cut job:

```
CREATE PROCEDURE proc_7_3
    @job_no INT,
    @job_machine_type VARCHAR(10),
    @machine_time VARCHAR(8),
    @material_used VARCHAR(3),
    @cut_labor_time VARCHAR(8)
AS
BEGIN
    INSERT INTO Job_cut
    ( -- Columns to insert data into
    [job_no], [job_machine_type], [machine_time], [material_used], [cut_labor_time]
    )
    VALUES (@job_no, @job_machine_type, CAST(@machine_time as TIME), @material_used, CAST(@cut
_labor_time as TIME))
END
```

8. Enter a transaction-no and its sup-cost and update all the costs (details) of the affected accounts by adding sup-cost to their current values of details

```
-- Enter a transaction-no and its sup-cost and update all the costs (details) of the affected
--accounts by adding sup-cost to their current values of details
CREATE PROCEDURE proc_8
    @t_no VARCHAR(10),
    @sup_cost FLOAT,
    @job_no INT
AS
BEGIN
    DECLARE @dacc_no INT,
            @aacc_no INT,
            @pacc_no INT;

    SET @aacc_no = (SELECT acc_no
```

```
                    FROM Assembly_acc, Job
                    WHERE Assembly_acc.assembly_id = Job.assembly_id AND Job.job_no = @job_no)
;


    SET @pacc_no = (SELECT acc_no
                    FROM Process_acc, Job
                    WHERE Process_acc.process_id = Job.process_id AND Job.job_no = @job_no);


    SET @dacc_no = (SELECT acc_no
                    FROM Dept_acc, Process, Job
                    WHERE Process.process_id = Job.process_id AND Job.job_no = @job_no AND Dep
t_acc.dept_no = Process.dept_no);


    INSERT INTO Transaction1
    ( -- Columns to insert data into
    [t_no], [sup_cost], [job_no], [dacc_no], [aacc_no], [pacc_no]
    )
    VALUES (@t_no, @sup_cost, @job_no, @dacc_no, @aacc_no, @pacc_no);


    UPDATE Dept_acc
    SET details_2 = details_2 + @sup_cost
    WHERE acc_no = @dacc_no;


    UPDATE Assembly_acc
    SET details_1 = details_1 + @sup_cost
    WHERE acc_no = @aacc_no;


    UPDATE Process_acc
    SET details_3 = details_3 + @sup_cost
    WHERE acc_no = @pacc_no;
END
```

9. Retrieve the cost incurred on an assembly-id

```
--Retrieve the cost incurred on an assembly-id

CREATE PROCEDURE proc_9

    @assembly_id VARCHAR(10)



AS

BEGIN

    SELECT details_1 FROM Assembly_acc WHERE Assembly_acc.assembly_id = @assembly_id;

END
```

10. Retrieve the total labor time within a department for jobs completed in the department during a given date

```
-- Retrieve the total labor time within a department for jobs completed
-- in the department during a given date
CREATE PROCEDURE proc_10
    @dept_no VARCHAR(7),
    @date_completed VARCHAR(10)

AS
BEGIN
    DECLARE @fl_time FLOAT,
            @pl_time FLOAT,
            @cl_time FLOAT,
            @tl_time FLOAT;
    SET @fl_time = (SELECT  SUM(( DATEPART(hh, fit_labor_time) * 3600 ) + ( DATEPART(mi, fit_l
abor_time) * 60 ) + DATEPART(ss, fit_labor_time))/60 as minute
FROM Job_fit WHERE Job_fit.job_no in (
SELECT distinct(job_no) FROM Job
WHERE Job.process_id in (SELECT distinct(process_id) FROM Process WHERE Process.dept_no = @dep
t_no) AND Job.date_completed = CAST(@date_completed as DATE)));

IF @fl_time IS NULL SET @fl_time = 0;

    SET @pl_time = (SELECT  SUM(( DATEPART(hh, paint_labor_time) * 3600 ) + ( DATEPART(mi,  pa
int_labor_time) * 60 ) + DATEPART(ss,  paint_labor_time))/60 as minute
FROM Job_paint WHERE Job_paint.job_no in (
SELECT distinct(job_no) FROM Job
WHERE Job.process_id in (SELECT distinct(process_id) FROM Process WHERE Process.dept_no = @dep
t_no) AND Job.date_completed =CAST(@date_completed as DATE)));

IF @pl_time IS NULL SET @pl_time = 0;

    SET @cl_time = (SELECT  SUM(( DATEPART(hh, cut_labor_time) * 3600 ) + ( DATEPART(mi, cut_l
abor_time) * 60 ) + DATEPART(ss, cut_labor_time))/60 as minute
```

```
FROM Job_cut WHERE Job_cut.job_no in (
SELECT distinct(job_no) FROM Job
WHERE Job.process_id in (SELECT distinct(process_id) FROM Process WHERE Process.dept_no = @dep
t_no) AND Job.date_completed =CAST(@date_completed as DATE)));

IF @cl_time IS NULL SET @cl_time = 0;

    SET @tl_time = @fl_time + @pl_time + @cl_time
    SELECT  @tl_time
END
```

11. Retrieve the processes through which a given assembly-id has passed so far (in date-commenced order) and the department responsible for each process

```
-- Retrieve the processes through which a given assembly-id has passed so far (in date-
commenced order) and the department responsible for each process

CREATE PROCEDURE proc_11

    @assembly_id VARCHAR(10)

AS

BEGIN

    SELECT Job.date_commenced, Job.process_id, Process.dept_no

    FROM Job, Process

    WHERE Job.assembly_id = @assembly_id AND Process.process_id = Job.process_id

    ORDER BY 1 ;


END
```

12. Retrieve the jobs (together with their type information and assembly-id) completed during a given date in a given department

To retrieve fit jobs:

```
-- Retrieve the jobs (together with their type information and assembly-
id) completed during a given date in a given department

CREATE PROCEDURE proc_12_1

    @date_completed VARCHAR(10),

    @dept_no VARCHAR(7)

AS

BEGIN

    SELECT DISTINCT(Job.job_no), Job.assembly_id, Job_fit.fit_labor_time
```

```
    FROM Job, Job_fit

    WHERE date_completed = @date_completed and Job.process_id in (SELECT process.process_id FR
OM Process WHERE dept_no = @dept_no) AND Job_fit.job_no = Job.job_no;



END
```

To retrieve paint jobs:

```
CREATE PROCEDURE proc_12_2

    @date_completed VARCHAR(10),

    @dept_no VARCHAR(7)

AS

BEGIN

    SELECT DISTINCT(Job.job_no), Job.assembly_id, Job_paint.color, Job_paint.volume, Job_paint
.paint_labor_time

    FROM Job, Job_paint

    WHERE date_completed = @date_completed and Job.process_id in (SELECT process.process_id FR
OM Process WHERE dept_no = @dept_no) AND Job_paint.job_no = Job.job_no;



END
```

To retrieve cut jobs:

```
CREATE PROCEDURE proc_12_3

    @date_completed VARCHAR(10),

    @dept_no VARCHAR(7)

AS

BEGIN

    SELECT DISTINCT(Job.job_no), Job.assembly_id, Job_cut.job_machine_type, Job_cut.machine_ti
me, Job_cut.material_used, Job_cut.cut_labor_time

    FROM Job, Job_cut

    WHERE date_completed = @date_completed and Job.process_id in (SELECT process.process_id FR
OM Process WHERE dept_no = @dept_no) AND Job_cut.job_no = Job.job_no;



END
```

13. Retrieve the customers (in name order) whose category is in a given range

```
-- Retrieve the customers (in name order) whose category is in a given range
CREATE PROCEDURE proc_13
    @lower_b INT,
    @upper_b INT
AS
BEGIN
    SELECT cname, category AS name FROM Customer
    WHERE category >= @lower_b AND category <= @upper_b
    ORDER BY 1 ;
END
```

14. Delete all cut-jobs whose job-no is in a given range

```
-- Delete all cut-jobs whose job-no is in a given range
CREATE PROCEDURE proc_14
    @lower_b INT,
    @upper_b INT
AS
BEGIN
    DELETE FROM Job_cut WHERE job_no >= @lower_b AND job_no <= @upper_b
END
```

15. Change the color of a given paint job

```
-- Change the color of a given paint
CREATE PROCEDURE proc_15
    @job_no INT,
    @color VARCHAR(10)
AS
BEGIN
    UPDATE Job_paint SET color = @color WHERE job_no = @job_no;
END
```

Executing all above Procedures: - I do not want to keep screenshot after running individual Procedures. This will make grading difficult for you. So, I am placing screenshot after executing the final procedure.

```
Messages
9:15:53 PM    Started executing query at Line 516
              Commands completed successfully.
              Total execution time: 00:00:00.041
```

5.2 Java application program that uses JDBC and Azure SQL Database to implement all SQL queries (options 1-17)

1. Enter a new customer (30/day).

```java
// try and catch are used to not terminate loop in case of error.
try {

//Declaring the variables
int category;
String cname, caddress;

//Taking customer name from user
System.out.println("Enter the Customer Name");
cname = myScan.next();

// Taking Customer Address from the user
System.out.println("Enter the Customer Address");
caddress = myScan.next();

// Taking customer category from the user
System.out.println("Enter the Customer Category");
category = myScan.nextInt();

// Executing procedure for Query 1.
final String Sql1 = "EXEC proc_1 @cname = '"+cname+"', @caddress = '"+caddress+"', @category = '"+category+"';";

final Statement statement1 = connection.createStatement();
    statement1.executeUpdate(Sql1);

    System.out.println("Customer record inserted successfully.");
    System.out.println("========================================");
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

2. Enter a new department (infrequent).

```
    // try and catch are used to not terminate loop in case of error.
    try {

    // Declaring variables
    String dept_no, dept_data;

    // Taking Department Number from user.
    System.out.println("Enter the Department Number\n");
    dept_no = myScan.next();

    // Taking Department data from user.
    System.out.println("Enter the Department Data\n");
    dept_data = myScan.next();

    // Executing Procedure for Query 2.
    final String Sql2 = "EXEC proc_2 @dept_no = '"+dept_no+"', @dept_data = '"+dept_data+"';";

    final Statement statement2 = connection.createStatement();
        statement2.executeUpdate(Sql2);

        System.out.println("Department record inserted successfully.");
        System.out.println("========================================");
    } catch (Exception e) {
        System.out.print("You got an error!. Returning to main menu");
    }
    break;
```

3. <u>Enter a new assembly with its customer-name, assembly-details, assembly-id, and date-ordered (40/day).</u>

```
    // try and catch are used to not terminate loop in case of error.
    try {
        // Declaring Variables
    String assembly_id, assembly_details, cname1;
    String date_ordered;

    // Taking Assembly-id from user
    System.out.println("Enter Assembly ID\n");
    assembly_id = myScan.next();

    // Taking date ordered from user
    System.out.println("Enter Date Ordered\n");
    date_ordered = myScan.next();

    // Taking Assembly details from user
    System.out.println("Enter Assembly Details\n");
    assembly_details = myScan.next();

    // Taking customer name from user
    System.out.println("Enter the Customer Name\n");
    cname1 = myScan.next();

    // Executing procedure for Query 3
    final String Sql3 = "EXEC proc_3 @assembly_id = '"+assembly_id+"', @data_ordered = '"+date_ordered+"'," +
                "@assembly_details = '"+assembly_details+"', @cname1 = '"+cname1+"';";

    final Statement statement3 = connection.createStatement();
        statement3.executeUpdate(Sql3);

        System.out.println("Assembly record inserted successfully.");
        System.out.println("========================================");

    } catch (Exception e) {
        System.out.print("You got an error!. Returning to main menu");
    }
    break;
```

4. <u>Enter a new process-id and its department together with its type and information relevant to the type (infrequent).</u>

Code to update Process table

```
// try and catch are used to not terminate loop in case of error.
try {

// Declaring Variables
String process_id, process_data, dept_no1;

// Taking process-id from the user
System.out.println("Enter Process ID\n");
process_id = myScan.next();

// Taking process data from the user
System.out.println("Enter Process Data\n");
process_data = myScan.next();

// Taking Department-no from the user
System.out.println("Enter Department No\n");
dept_no1 = myScan.next();

//Executing Procedure for Query 4
final String Sql4 = "EXEC proc_4 @process_id = '"+process_id+"'," +
        " @process_data = '"+process_data+"', @dept_no1 = '"+dept_no1+"';";

final Statement statement4 = connection.createStatement();
    statement4.executeUpdate(Sql4);

    System.out.println("Process record inserted successfully.");
    System.out.println("========================================");

// Asking user to enter the type of procedure
System.out.println("Choose one of the following type of process:\n 1.Fit\n 2. Paint\n 3.Cut\n");
int choice1;
// Taking the type of process from the user
choice1 = myScan.nextInt();
```

Code to update Process fit table

```
if (choice1 ==1) {

    // Declaring Variables
    String fit_type;

    // Taking fit type from user
    System.out.println("Enter fit type\n");
    fit_type = myScan.next();

    // Executing Procedure to insert data in Process_fit table
    final String Sql4_1 = "EXEC proc_4_1 @process_id = '"+process_id+"',  @fit_type = '"+fit_type+"';";

    final Statement statement4_1 = connection.createStatement();
        statement4_1.executeUpdate(Sql4_1);

        System.out.println("Process_fit record inserted successfully.");
        System.out.println("========================================");
}
```

Code to update Process Paint

```
if (choice1 ==2) {

    // Declaring Variables
    String paint_type, paint_method;

    //Taking paint type from the user
    System.out.println("Enter paint type");
    paint_type = myScan.next();

    // Taking paint method from the user
    System.out.println("Enter paint method");
    paint_method = myScan.next();

    // Executing procedure to insert data into Process_paint table
    final String Sql4_2 = "EXEC proc_4_2 @process_id = '"+process_id+"', " +
            " @paint_type = '"+paint_type+"', @paint_method = '"+paint_method+"';";

    final Statement statement4_2 = connection.createStatement();
        statement4_2.executeUpdate(Sql4_2);

        System.out.println("Process_paint record inserted successfully.");
        System.out.println("========================================");
}
```

Code to update Process Cut

```
if (choice1 ==3) {

    // Declaring Variables
    String cutting_type, machine_type;

    //Taking cut type from the user
    System.out.println("Enter cutting type");
    cutting_type = myScan.next();

    // Take machine type from user
    System.out.println("Enter machine type");
    machine_type = myScan.next();

    //Executing Procedure to insert data into Process_cut table.
    final String Sql4_3 = "EXEC proc_4_3 @process_id = '"+process_id+"', " +
            " @cutting_type = '"+cutting_type+"', @machine_type = '"+machine_type+"';";

    final Statement statement4_3 = connection.createStatement();
        statement4_3.executeUpdate(Sql4_3);

        System.out.println("Process_cut record inserted successfully.");
        System.out.println("========================================");
}
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

5. Create a new account and associate it with the process, assembly, or department to which it is applicable (10/day).

Code to ask basic account details

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
int acc_no, choice2;
String date_established;

// Taking account number from user
System.out.println("Enter account number");
acc_no = myScan.nextInt();

// Taking date account established from user
System.out.println("Enter date account established");
date_established = myScan.next();

// Asking user to provide the tye of account
System.out.println("Choose one of the following type of account:\n 1. Department Account.\n 2. Assembly Account\n 3.Process Account\n");
choice2 = myScan.nextInt();
```

Code to add Department Account

```
if (choice2 == 1) {
    // Declaring Variables
    float details2;
    String dept_no2;

    // Taking details-2 from the user
    System.out.println("Enter account details");
    details2 = myScan.nextFloat();

    // Taking department no from the user
    System.out.println("Enter Department Number of the account");
    dept_no2 = myScan.next();

    // Executing procedure to insert data into Department Account Table
    final String Sql5_1 = "EXEC proc_5_1 @acc_no = '"+acc_no+"',  @date_established = '"+date_established+"','"+
            " @details_2 = '"+details2+"', @dept_no = '"+dept_no2+"';";

    final Statement statement5_1 = connection.createStatement();
        statement5_1.executeUpdate(Sql5_1);

        System.out.println("Dept_acc record inserted successfully.");
        System.out.println("========================================");
}
```

Code to add Assembly account

```
if (choice2 == 2) {

    // Declaring Variables
    float details1;
    String assembly_id1;

    // Taking details1 from user
    System.out.println("Enter account details");
    details1 = myScan.nextFloat();

    // Taking assembly-id from user
    System.out.println("Enter Assembly id of the account");
    assembly_id1 = myScan.next();

    // Executing procedure to insert data into Assembly account
    final String Sql5_2 = "EXEC proc_5_2 @acc_no = '"+acc_no+"',  @date_established = '"+date_established+"','"+
            " @details_1 = '"+details1+"',  @assembly_id = '"+assembly_id1+"';";

    final Statement statement5_2 = connection.createStatement();
        statement5_2.executeUpdate(Sql5_2);

        System.out.println("Assembly_acc record inserted successfully.");
        System.out.println("========================================");
}
```

Code to add Process Account

```
if (choice2 == 3) {

    // Declaring Variables
    float details3;
    String process_id1;

    // Taking details from the user
    System.out.println("Enter account details");
    details3 = myScan.nextFloat();

    // Taking process-id from the user
    System.out.println("Enter Process id of the account");
    process_id1 = myScan.next();

    // Executing the procedure to insert data into Process account
    final String Sql5_3 = "EXEC proc_5_3 @acc_no = '"+acc_no+"',  @date_established = '"+date_established+"','"+
            " @details_3 = '"+details3+"',  @process_id = '"+process_id1+"';";

    final Statement statement5_3 = connection.createStatement();
        statement5_3.executeUpdate(Sql5_3);

        System.out.println("Process_acc record inserted successfully.");
        System.out.println("========================================");
}
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

6. Enter a new job, given its job-no, assembly-id, process-id, and date the job commenced (50/day).

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String date_commenced, assembly_id2, process_id2;
int job_no;

// Taking Job-no from the user
System.out.println("Enter Job-no for a new job");
job_no = myScan.nextInt();

// Taking assembly-id from the user
System.out.println("Enter assembly-id for the job");
assembly_id2 = myScan.next();

// Taking process-id from user
System.out.println("Enter process-id for the job");
process_id2 = myScan.next();

// Taking job commenced from user
System.out.println("Enter date commenced for the job");
date_commenced = myScan.next();

// Executing Procedure for Query 6
final String Sql6 = "EXEC proc_6 @job_no = '"+job_no+"', @date_commenced = '"+date_commenced+"','"+
        " @assembly_id = '"+assembly_id2+"', @process_id = '"+process_id2+"';";

final Statement statement6 = connection.createStatement();
    statement6.executeUpdate(Sql6);

    System.out.println("Job record inserted successfully.");
    System.out.println("========================================");

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

7. At the completion of a job, enter the date it completed and the information relevant to the type of job (50/day).

Code to update Job table

```
// try and catch are used to not terminate loop in case of error.
try {
    // Declaring Variables
String date_completed;
int job_no1;

// Taking Job-no from user
System.out.println("Enter Job-no for for the completed job");
job_no1 = myScan.nextInt();

// Taking date completed from user
System.out.println("Enter job completed date in yyyy-mm-dd format");
date_completed = myScan.next();

// Executing the Procedure for Query 7
final String Sql7 = "EXEC proc_7 @job_no = '"+job_no1+"', @date_completed = '"+date_completed+"';";

final Statement statement7 = connection.createStatement();
    statement7.executeUpdate(Sql7);

    System.out.println("Job record Updated successfully.");
    System.out.println("========================================");

    // Declaring Variables
int choice3;

// Asking the type of job from the user
System.out.println("Choose one of the following type of job:\n 1. Fit Job.\n 2. Paint Job\n 3.Cut Job\n");
choice3 = myScan.nextInt();
```

Code to insert into Job fit table

```
if (choice3 == 1) {

    // Declaring Variables
    String fit_labor_time;

    // Taking fit labor time from user
    System.out.println("Enter the fit job labor time in HH:MM:SS format.");
    fit_labor_time = myScan.next();

    // Executing procedure to insert data into Job_fit table
    final String Sql7_1 = "EXEC proc_7_1 @job_no = '"+job_no1+"', @fit_labor_time = '"+fit_labor_time+"';";

    final Statement statement7_1 = connection.createStatement();
        statement7_1.executeUpdate(Sql7_1);

        System.out.println("Fit Job record inserted successfully.");
        System.out.println("========================================");
}
```

Code to insert into Job paint table

```
if (choice3 == 2) {

    // Declaring Variables
    String color, paint_labor_time;
    int volume;

    // Taking color from user
    System.out.println("Enter the paint color.");
    color = myScan.next();

    // Taking labor time from user
    System.out.println("Enter the paint job labor time in HH:MM:SS format.");
    paint_labor_time = myScan.next();

    // Taking volumme of paint from user
    System.out.println("Enter the volume of paint.");
    volume = myScan.nextInt();

    // Executing the procedure to insert data into Job paint table
    final String Sql7_2 = "EXEC proc_7_2 @job_no = '"+job_no1+"', @color = '"+color+"', "+
            " @volume = '"+volume+"', @paint_labor_time = '"+paint_labor_time+"';";

    final Statement statement7_2 = connection.createStatement();
        statement7_2.executeUpdate(Sql7_2);

        System.out.println("Paint Job record inserted successfully.");
        System.out.println("========================================");
}
```

Code to insert into Job cut table

```
if (choice3 == 3) {

    // Declaring Variables
    String job_machine_type, machine_time, material_used, cut_labor_time;

    // Taking machine type from user
    System.out.println("Enter the Job Machine Type.");
    job_machine_type = myScan.next();

    // Taking machine time from user
    System.out.println("Enter the cut job machine time in HH:MM:SS format.");
    machine_time = myScan.next();

    // Taking material used from the user
    System.out.println("Enter the material used.");
    material_used = myScan.next();

    // Taking labor time from the user
    System.out.println("Enter the cut job labor time in HH:MM:SS format.");
    cut_labor_time = myScan.next();

    // Executing Procedure to insert data into Job_cut table
    final String Sql7_3 = "EXEC proc_7_3 @job_no = '"+job_no1+"', @job_machine_type= '"+job_machine_type+"', "+
    " @machine_time = '"+machine_time+"', @material_used = '"+material_used+"', @cut_labor_time = '"+cut_labor_time+"';";

    final Statement statement7_3 = connection.createStatement();
        statement7_3.executeUpdate(Sql7_3);

        System.out.println("Cut Job record inserted successfully.");
        System.out.println("========================================");
}
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

8. Enter a transaction-no and its sup-cost and update all the costs (details) of the affected accounts by adding sup-cost to their current values of details (50/day).

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String t_no;
int job_no2;
float sup_cost;

// Taking transaction number from the user
System.out.println("Enter the Transaction Number.");
t_no = myScan.next();

// Taking sup-cost from the user
System.out.println("Enter the cost of the transaction.");
sup_cost = myScan.nextFloat();

// Taking job-no from the user
System.out.println("Enter the Job number related to transaction.");
job_no2 = myScan.nextInt();

// Executing procedure for Query 8.
final String Sql8 = "EXEC proc_8 @t_no = '"+t_no+"', @sup_cost = '"+sup_cost+"', @job_no = '"+job_no2+"';";

final Statement statement8 = connection.createStatement();
    statement8.executeUpdate(Sql8);

    System.out.println("Transaction record inserted and related accounts updated successfully.");
    System.out.println("========================================");

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

9. Retrieve the cost incurred on an assembly-id (200/day).

```
// try and catch are used to not terminate loop in case of error.
try {
    // Declaring Variables
String assembly_id3;

// Taking assembly-id from the user
System.out.println("Enter Assembly Id to retrieve the cost.");
assembly_id3 = myScan.next();

try {

// Executing procedure for Query 9
final String Sql9 = "EXEC proc_9 @assembly_id = '"+assembly_id3+"';";

try (final Statement statement9 = connection.createStatement();
        final ResultSet resultSet1 = statement9.executeQuery(Sql9)) {

        System.out.println(String.format("Cost incurred on assembly-id %s:", assembly_id3));
        while (resultSet1.next()) {
            System.out.println(String.format("%f",
                resultSet1.getFloat(1)));

            }
        }
} catch (Exception e) {
    System.out.println("You got into an error!. Please try again.");
}

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

10. <u>Retrieve the total labor time within a department for jobs completed in the department during a given date (20/day).</u>

```java
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String dept_no2,date_completed1;

// Taking department-no from user
System.out.println("Enter Department Number to get the toatal labor time.");
dept_no2 = myScan.next();

// Taking job-completed date from user
System.out.println("Enter Job completed date to get the toatal labor time.");
date_completed1 = myScan.next();

// Executing the procedure for Query 10
final String Sql10 = "EXEC proc_10 @dept_no = '"+dept_no2+"', @date_completed = '"+date_completed1+"';";

try (final Statement statement10 = connection.createStatement();
        final ResultSet resultSet2 = statement10.executeQuery(Sql10)) {

        System.out.println(String.format("Total labor-time in minutes for department number %s and date" +
                " job completed %s:", dept_no2, date_completed1));
        while (resultSet2.next()) {
            System.out.println(String.format("%s",
                resultSet2.getInt(1)));

            }
        }
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

11. <u>Retrieve the processes through which a given assembly-id has passed so far (in date-commenced order) and the department responsible for each process (100/day).</u>

```java
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String assembly_id4;

// Taking assembly-id from user
System.out.println("Enter the Assembly Id to retrieve the processes.");
assembly_id4 = myScan.next();

// Executing procedure for Query 11d
final String Sql11 = "EXEC proc_11 @assembly_id = '"+assembly_id4+"';";

try (final Statement statement11 = connection.createStatement();
        final ResultSet resultSet3 = statement11.executeQuery(Sql11)) {

        System.out.println(String.format("The processes through which Assembly Id %s passed so far:", assembly_id4));
        while (resultSet3.next()) {
            System.out.println(String.format("%s | %s | %s",
                resultSet3.getString(1),
                resultSet3.getString(2),
                resultSet3.getString(3)));

            }
        }
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

12. <u>Retrieve the jobs (together with their type information and assembly-id) completed during a given date in a given department (20/day).</u>

Code to get fit jobs

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String date_completed2, dept_no3;

// Taking date completed from user
System.out.println("Enter the date job is completed.");
date_completed2 = myScan.next();

// Taking department number from user
System.out.println("Enter the department to retrieve the jobs.");
dept_no3 = myScan.next();

// Executing Procedure to retrieve fit jobs
final String Sql12_1 = "EXEC proc_12_1 @date_completed = '"+date_completed2+"', @dept_no = '"+dept_no3+"';";

try (final Statement statement12_1 = connection.createStatement();
        final ResultSet resultSet5 = statement12_1.executeQuery(Sql12_1)) {

        System.out.println(String.format("The fit jobs completed on date %s in department %s:", date_completed2, dept_no3));
        while (resultSet5.next()) {
            System.out.println(String.format("%s | %s | %s",
                resultSet5.getString(1),
                resultSet5.getString(2),
                resultSet5.getString(3)));

            }
        }
```

Code to get paint jobs

```
        }
// Executing procedure to retrieve paint jobs
final String Sql12_2 = "EXEC proc_12_2 @date_completed = '"+date_completed2+"', @dept_no = '"+dept_no3+"';";

try (final Statement statement12_2 = connection.createStatement();
        final ResultSet resultSet6 = statement12_2.executeQuery(Sql12_2)) {

        System.out.println(String.format("The Paint jobs completed on date %s in department %s:", date_completed2, dept_no3));
        while (resultSet6.next()) {
            System.out.println(String.format("%s | %s | %s | %s | %s",
                resultSet6.getString(1),
                resultSet6.getString(2),
                resultSet6.getString(3),
                resultSet6.getString(4),
                resultSet6.getString(5)));

            }
        }
```

Code to get Cut Jobs

```
// Executing procedure to retrieve cut jobs
final String Sql12_3 = "EXEC proc_12_3 @date_completed = '"+date_completed2+"', @dept_no = '"+dept_no3+"';";

try (final Statement statement12_3 = connection.createStatement();
        final ResultSet resultSet7 = statement12_3.executeQuery(Sql12_3)) {

        System.out.println(String.format("The Cut jobs completed on date %s in department %s:", date_completed2, dept_no3));
        while (resultSet7.next()) {
            System.out.println(String.format("%s | %s | %s | %s | %s | %s",
                resultSet7.getString(1),
                resultSet7.getString(2),
                resultSet7.getString(3),
                resultSet7.getString(4),
                resultSet7.getString(5),
                resultSet7.getString(6)));

            }
        }
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}

break;
```

13. <u>Retrieve the customers (in name order) whose category is in a given range (100/day).</u>

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
int lower_b, upper_b;

// Taking lower bound of category from user
System.out.println("Enter the lower bound of the category.");
lower_b = myScan.nextInt();

// Taking upper bound of category from user
System.out.println("Enter the upper bound of the category.");
upper_b = myScan.nextInt();

// Executing procedure for Query 13
final String Sql13 = "EXEC proc_13 @lower_b = '"+lower_b+"', @upper_b = '"+upper_b+"';";

try (final Statement statement13 = connection.createStatement();
        final ResultSet resultSet4 = statement13.executeQuery(Sql13)) {

        System.out.println("The customers in the given range of category are");
        while (resultSet4.next()) {
            System.out.println(String.format("%s | %s",
                resultSet4.getString(1),
                resultSet4.getString(2)));

            }
        }
} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

14. <u>Delete all cut-jobs whose job-no is in a given range (1/month).</u>

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
int lower_b1, upper_b1;

// Taking lower bound job-no from user
System.out.println("Enter the lower bound of the cut job-no.");
lower_b1 = myScan.nextInt();

// Taking upper bound job-no from user
System.out.println("Enter the upper bound of the cut job-no.");
upper_b1 = myScan.nextInt();

// Executing procedure for Query 14
final String Sql14 = "EXEC proc_14 @lower_b = '"+lower_b1+"', @upper_b = '"+upper_b1+"';";

final Statement statement14 = connection.createStatement();
    statement14.executeUpdate(Sql14);

    System.out.println("Deleted all cut jobs in the given range of job-no.");
    System.out.println("========================================");

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

15. <u>Change the color of a given paint job (1/week).</u>

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
String color;
int job_no3;

// Taking job-no from user
System.out.println("Enter the paint job-no.");
job_no3 = myScan.nextInt();

// Taking color from user
System.out.println("Enter the new color for the job-no.");
color = myScan.next();

// Executing procedure for Query 15
final String Sql15 = "EXEC proc_15 @job_no = '"+job_no3+"', @color = '"+color+"';";

final Statement statement15 = connection.createStatement();
    statement15.executeUpdate(Sql15);

    System.out.println("Changed the color of a given paint job.");
    System.out.println("========================================");

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

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

```
try {
    // Declaring Variables
String file_name, line;

// Taking Input file name from user
System.out.println("Enter the file-name to Import data.");
file_name = myScan.next();

// Creating new Fiel object
File file = new File(file_name);

// Creating new Scanner Object
Scanner sc = new Scanner(file);

// While loop to read all lines in the input file
while(sc.hasNextLine()) {
    line = sc.nextLine();

    // Dividing line to parts separated by Delimiter (",")
    String[] parts = line.split(",");

    String cname2 = parts[0];
    String caddress1 = parts[1];
    int category1 = Integer.parseInt(parts[2]);

    // Execute procedure for Query 16
    final String Sql16 = "EXEC proc_1 @cname = '"+cname2+"', @caddress = '"+caddress1+"', @category = '"+category1+"';";

    final Statement statement16 = connection.createStatement();
        statement16.executeUpdate(Sql16);

}

} catch (Exception e) {
    System.out.print("You got an error!. Returning to main menu");
}
break;
```

17. Export: Retrieve the customers (in name order) whose category is in a given range and output them to a data file instead of screen (the user must be asked to enter the output file name).

```
// try and catch are used to not terminate loop in case of error.
try {

    // Declaring Variables
    String file_name1, lower_b2, upper_b2;

    // Enter the filename to output result
    System.out.println("Enter the file-name to Export data.");
    file_name1 = myScan.next();

    // Taking lower bound of category from user
    System.out.println("Enter the lower bound of category.");
    lower_b2 = myScan.next();

    // Taking upper bound of category from user
    System.out.println("Enter the upper bound of category.");
    upper_b2 = myScan.next();

    // Creating new file writer Object
    FileWriter fw = new FileWriter(file_name1);

    // Executing Procedure(for Query 13) to get output
    final String Sql17 = "EXEC proc_13 @lower_b = '"+lower_b2+"', @upper_b = '"+upper_b2+"';";
```

```
     try (final Statement statement17 = connection.createStatement();
             final ResultSet resultSet4 = statement17.executeQuery(Sql17)) {

             //System.out.println("The customers in the given range of category are");
             while (resultSet4.next()) {

                 //fw.write(String.format("%s,%s", resultSet4.getString(1), resultSet4.getString(2)));
                 fw.write(resultSet4.getString(1) + "," + resultSet4.getString(2) + "\n");
                 }
             fw.close();
             } catch (SQLException e) {
                 e.getCause().getMessage();
             }


     } catch (Exception e) {
         System.out.print("You got an error!. Returning to main menu");
     }
 break;
```

18. <u>Quit</u>

```
case 18:
    System.out.println("You choose to Quit! Bye!");
```

Task 6. Java Program Execution

6.1 Screenshots showing the testing of query 1

Showing implementation for 2 queries in java:

```
Working Directory = C:\Users\kgt04\OneDrive\Documents\DSA 4513\Java\SampleAzureSQLProject
Successful connection - Schema:dbo
==========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. ENter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transsaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible
responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the colot of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
1
Enter the Customer Name
Ford
Enter the Customer Address
Dearborn
Enter the Customer Category
1
Customer record inserted successfully.
==========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. ENter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transsaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible
responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
```

13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the colot of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
1
Enter the Customer Name
Harman
Enter the Customer Address
Novi
Enter the Customer Category
2

Affected Customer Table: -

```
675    SELECT * FROM Customer;
```

Results Messages

	cname	caddress	category
1	Avion	Chennai	4
2	Ford	Dearborn	1
3	Harman	Novi	2
4	IFB	Bangalore	5
5	Nvidia	CA	3

6.2 Screenshots showing the testing of query 2

Showing Implementation of 2 queries

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. ENter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transsaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)

14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the colot of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
2
Enter the Department Number

dept1
Enter the Department Data

dept_data1
Department record inserted successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. ENter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transsaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the colot of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
2
Enter the Department Number

dept2
Enter the Department Data

dept_data2
Department record inserted successfully.

Affected Department Table: -

```
675    SELECT * FROM Department;
```

Results Messages

	dept_no	dept_data
1	dept1	dept_data1
2	dept2	dept_data2
3	dept3	dept_data3
4	dept4	dept_data4
5	dept5	dept_data5

6.3 Screenshots showing the testing of query 3

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
3
Enter Assembly ID
aid01
Enter Date Ordered in yyyy-mm-dd format
2019-01-01
Enter Assembly Details
a-d-1
Enter the Customer Name
Ford
Assembly record inserted successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)

3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
3
Enter Assembly ID
aid02
Enter Date Ordered in yyyy-mm-dd format
2019-02-03
Enter Assembly Details
a-d-2
Enter the Customer Name
Ford
Assembly record inserted successfully.

Affected Assembly Table: -

```
SELECT * FROM Assembly;
```

Results | Messages

	assembly_id	date_order…	assembly_details	cname
1	aid01	2019-01-01	a-d-1	Ford
2	aid02	2019-02-03	a-d-2	Ford
3	aid03	2019-01-15	a-d-3	Harman
4	aid04	2019-02-15	a-d-4	Nvidia
5	aid05	2019-02-10	a-d-5	Nvidia
6	aid06	2019-03-05	a-d-6	IFB
7	aid07	2019-03-05	a-d-7	Avion
8	aid08	2019-04-04	a-d-8	IFB
9	aid09	2019-03-20	a-d-9	Avion
10	aid10	2019-05-05	a-d-10	Ford

6.4 Screenshots showing the testing of query 4

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
4
Enter Process ID
proc001
Enter Process Data
pdata1
Enter Department No
dept1
Process record inserted successfully.
==========================================
Choose one of the following type of process:
 1.Fit
 2. Paint
 3.Cut

1
Enter fit type

fit01
Process_fit record inserted successfully.
==========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
```

11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
4
Enter Process ID
proc002
Enter Process Data
pdata2
Enter Department No
dept2
Process record inserted successfully.
==========
Choose one of the following type of process:
1.Fit
2. Paint
3.Cut

2
Enter paint type
type01
Enter paint method
method01
Process_paint record inserted successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
4
Enter Process ID
proc003
Enter Process Data

pdata3
Enter Department No
dept1
Process record inserted successfully.
==========
Choose one of the following type of process:
1.Fit
2. Paint
3.Cut

3
Enter cutting type
ctype01
Enter machine type
machine01

Affected Process Table: -

```
674
675    SELECT * FROM Process;
```

Results Messages

	process_id	process_da...	dept_no
1	proc001	pdata1	dept1
2	proc002	pdata2	dept2
3	proc003	pdata3	dept1
4	proc004	pdata4	dept3
5	proc005	pdata5	dept3
6	proc006	pdata6	dept5
7	proc007	pdata7	dept5
8	proc008	pdata8	dept3
9	proc009	pdata9	dept4
10	proc010	pdata10	dept4

Affected Process fit Table: -

```
675    SELECT * FROM Process_fit;
```

Results | Messages

	process_id	fit_type
1	proc001	fit01
2	proc004	fit02
3	proc009	fit01
4	proc010	fit01

Affected Process paint Table: -

```
675    SELECT * FROM Process_paint;
```

Results | Messages

	process_id	paint_type	paint_meth...
1	proc002	type01	method01
2	proc005	type02	method02
3	proc007	type01	method02

Affected Process cut Table: -

```
675    SELECT * FROM Process_cut;
```

Results | Messages

	process_id	cutting_ty...	machine_ty...
1	proc003	ctype01	machine01
2	proc006	ctype02	machine02
3	proc008	ctype03	machine03

6.5 Screenshots showing the testing of query 5

Code Showing insertion of new account of each type: -

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
5
Enter account number
1
Enter date account established
2018-12-15
Choose one of the following type of account:
 1. Department Account.
 2. Assembly Account
 3.Process Account

1
Enter account details
2000
Enter Department Number of the account
dept1
Dept_acc record inserted successfully.
==========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
```

8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
5
Enter account number
1001
Enter date account established
2019-01-01
Choose one of the following type of account:
1. Department Account.
2. Assembly Account
3.Process Account

2
Enter account details
10
Enter Assembly id of the account
aid01
Assembly_acc record inserted successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
5
Enter account number
10001
Enter date account established
2019-01-01

Choose one of the following type of account:
1. Department Account.
2. Assembly Account
3.Process Account

3
Enter account details
5
Enter Process id of the account
proc001
Process_acc record inserted successfully.

Affected Department Account Table: -

```
675    SELECT * FROM Dept_acc;
```

Results Messages

	acc_no	date_established	details_2	dept_no
1	1	2018-12-15	2000	dept1
2	2	2018-12-15	90	dept2
3	3	2018-12-15	75	dept3
4	4	2018-12-15	4	dept4
5	5	2018-12-15	5	dept5

Affected Assembly Account Table: -

```
SELECT * FROM Assembly_acc;
```

Results | Messages

	acc_no	date_established	details_1	assembly_id
1	1001	2019-01-01	10	aid01
2	1002	2019-02-03	15	aid02
3	1003	2019-01-15	8	aid03
4	1004	2019-02-15	0	aid04
5	1005	2019-02-10	7	aid05
6	1006	2019-03-05	12	aid06
7	1007	2019-03-05	10	aid07
8	1008	2019-04-04	23	aid08
9	1009	2019-03-20	18	aid09
10	1010	2019-05-05	5	aid10

Affected Process Account Table: -

```
SELECT * FROM Process_acc;
```

Results | Messages

	acc_no	date_established	details_3	process_id
1	10001	2019-01-01	5	proc001
2	10002	2018-12-15	20	proc002
3	10003	2018-12-15	70	proc003
4	10004	2018-12-15	50	proc004
5	10005	2018-12-15	60	proc005
6	10006	2018-12-15	45	proc006
7	10007	2018-12-15	90	proc007
8	10008	2018-12-15	40	proc008
9	10009	2018-12-15	98	proc009
10	10010	2018-12-15	79	proc010

6.6 Screenshots showing the testing of query 6

Code Showing insertion of new Job: -

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
========================================
6
Enter Job-no for a new job
1
Enter assembly-id for the job
aid01
Enter process-id for the job
proc001
Enter date commenced for the job
2019-01-01
Job record inserted successfully.
========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
```

15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
6
Enter Job-no for a new job
2
Enter assembly-id for the job
aid01
Enter process-id for the job
proc002
Enter date commenced for the job
2019-01-01
Job record inserted successfully.

Affected Job Table: -

```
681    SELECT * FROM Job;
```

Results Messages

	job_no	date_commenced	date_completed	assembly_id	process_id
1	1	2019-01-01	NULL	aid01	proc001
2	2	2019-01-01	NULL	aid01	proc002
3	3	2019-02-03	NULL	aid02	proc003
4	4	2019-02-15	NULL	aid04	proc004
5	5	2019-02-10	NULL	aid05	proc005
6	6	2019-03-05	NULL	aid06	proc006
7	7	2019-02-10	NULL	aid05	proc004
8	8	2019-03-20	NULL	aid09	proc009
9	9	2019-03-20	NULL	aid09	proc004
10	10	2019-03-05	NULL	aid07	proc009

6.7 Screenshots showing the testing of query 7

Code Showing insertion of each Job type: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)

7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
7
Enter Job-no for for the completed job
1
Enter job completed date in yyyy-mm-dd format
2019-01-31
Job record Updated successfully.
==========
Choose one of the following type of job:
1. Fit Job.
2. Paint Job
3.Cut Job

1
Enter the fit job labor time in HH:MM:SS format.
05:30:00
Fit Job record inserted successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
7
Enter Job-no for for the completed job
2
Enter job completed date in yyyy-mm-dd format

```
2019-01-31
Job record Updated successfully.
==========================================
Choose one of the following type of job:
 1. Fit Job.
 2. Paint Job
 3.Cut Job

2
Enter the paint color.
yellow
Enter the paint job labor time in HH:MM:SS format.
1:30:00
Enter the volume of paint.
2
Paint Job record inserted successfully.
==========================================
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
7
Enter Job-no for for the completed job
3
Enter job completed date in yyyy-mm-dd format
2019-02-28
Job record Updated successfully.
==========================================
Choose one of the following type of job:
 1. Fit Job.
 2. Paint Job
 3.Cut Job

3
Enter the Job Machine Type.
jmachine01
Enter the cut job machine time in HH:MM:SS format.
4:00:00
Enter the material used.
```

m01
Enter the cut job labor time in HH:MM:SS format.
7:00:00
Cut Job record inserted successfully.

Affected Job Table: -

```
SELECT * FROM Job;
```

Results | Messages

	job_no	date_commenced	date_completed	assembly_id	process_id
1	1	2019-01-01	2019-01-31	aid01	proc001
2	2	2019-01-01	2019-01-31	aid01	proc002
3	3	2019-02-03	2019-02-28	aid02	proc003
4	4	2019-02-15	2019-02-28	aid04	proc004
5	5	2019-02-10	2019-02-28	aid05	proc005
6	6	2019-03-05	2019-03-31	aid06	proc006
7	7	2019-02-10	2019-02-28	aid05	proc004
8	8	2019-03-20	2019-03-31	aid09	proc009
9	9	2019-03-20	2019-03-31	aid09	proc004
10	10	2019-03-05	2019-03-31	aid07	proc009

Affected Job Fit Table: -

```
SELECT * FROM Job_fit;
```

Results | Messages

	job_no	fit_labor_time
1	1	05:30:00
2	5	03:00:00
3	8	02:00:00
4	9	02:00:00

Affected Job Paint Table: -

```
SELECT * FROM Job_paint;
```

Results | Messages

	job_no	color	volume	paint_labor_time
1	2	yellow	2	01:30:00
2	4	red	1	03:00:00
3	10	green	3	02:30:00

Affected Job Cut Table: -

```
SELECT * FROM Job_cut;
```

Results | Messages

	job_no	job_machine_type	machine_ti…	material_used	cut_labor_time
1	3	jmachine01	04:00:00	m01	07:00:00
2	6	jmachine02	04:00:00	m02	00:30:00
3	7	jmachine01	02:00:00	m03	03:00:00

6.8 Screenshots showing the testing of query 8

Code Showing insertion of 2 Transactions: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)

7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
8
Enter the Transaction Number.
t001
Enter the cost of the transaction.
35
Enter the Job number related to transaction.
1
Transaction record inserted and related accounts updated successfully.
==========
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
8
Enter the Transaction Number.
t002
Enter the cost of the transaction.
50
Enter the Job number related to transaction.
2
Transaction record inserted and related accounts updated successfully.

Affected Transaction Table: -

```
SELECT * FROM Transaction1;
```

Results | Messages

	t_no	sup_cost	job_no	dacc_no	aacc_no	pacc_no
1	t001	35	1	1	1001	10001
2	t002	50	2	2	1001	10002
3	t003	20	3	1	1002	10003
4	t004	50	1	1	1001	10001
5	t005	5	4	3	1004	10004
6	t006	10	7	3	1005	10004
7	t007	20	9	3	1009	10004
8	t008	25	5	3	1005	10005
9	t009	15	6	5	1006	10006
10	t010	10	3	1	1002	10003

Affected Department Account Table: -

```
SELECT * FROM Dept_acc;
```

Results | Messages

	acc_no	date_established	details_2	dept_no
1	1	2018-12-15	2115	dept1
2	2	2018-12-15	140	dept2
3	3	2018-12-15	135	dept3
4	4	2018-12-15	4	dept4
5	5	2018-12-15	20	dept5

Affected Assembly Account Table: -

```
692
693    SELECT * FROM Assembly_acc;
```

Results | Messages

	acc_no	date_established	details_1	assembly_id
1	1001	2019-01-01	145	aid01
2	1002	2019-02-03	45	aid02
3	1003	2019-01-15	8	aid03
4	1004	2019-02-15	5	aid04
5	1005	2019-02-10	42	aid05
6	1006	2019-03-05	27	aid06
7	1007	2019-03-05	10	aid07
8	1008	2019-04-04	23	aid08
9	1009	2019-03-20	38	aid09
10	1010	2019-05-05	5	aid10

Affected Process Account Table: -

```
694
695    SELECT * FROM Process_acc;
```

Results | Messages

	acc_no	date_established	details_3	process_id
1	10001	2019-01-01	90	proc001
2	10002	2018-12-15	70	proc002
3	10003	2018-12-15	100	proc003
4	10004	2018-12-15	85	proc004
5	10005	2018-12-15	85	proc005
6	10006	2018-12-15	60	proc006
7	10007	2018-12-15	90	proc007
8	10008	2018-12-15	40	proc008
9	10009	2018-12-15	98	proc009
10	10010	2018-12-15	79	proc010

6.9 Screenshots showing the testing of query 9

Three different queries of Query 9:

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
9
Enter Assembly Id to retrieve the cost.
aid01
Cost incurred on assembly-id aid01:
145.000000
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
9
Enter Assembly Id to retrieve the cost.
```

aid02
Cost incurred on assembly-id aid02:
45.000000
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
9
Enter Assembly Id to retrieve the cost.
aid04
Cost incurred on assembly-id aid04:
5.000000

6.10 Screenshots showing the testing of query 10

Three different queries for Query 10: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)

16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
10
Enter Department Number to get the total labor time.
dept1
Enter Job completed date to get the total labor time.
2019-01-31
Total labor-time in minutes for department number dept1 and date job completed 2019-01-31:
330
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
10
Enter Department Number to get the total labor time.
dept3
Enter Job completed date to get the total labor time.
2019-02-28
Total labor-time in minutes for department number dept3 and date job completed 2019-02-28:
540
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)

13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
10
Enter Department Number to get the total labor time.
dept4
Enter Job completed date to get the total labor time.
2019-03-31
Total labor-time in minutes for department number dept4 and date job completed 2019-03-31:
270

6.11 Screenshots showing the testing of query 11

Three different queries for Query 11: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
11
Enter the Assembly Id to retrieve the processes.
aid01
The processes through which Assembly Id aid01 passed so far:
2019-01-01 | proc001 | dept1
2019-01-01 | proc002 | dept2
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)

4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
11
Enter the Assembly Id to retrieve the processes.
aid04
The processes through which Assembly Id aid04 passed so far:
2019-02-15 | proc004 | dept3
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
11
Enter the Assembly Id to retrieve the processes.
aid09
The processes through which Assembly Id aid09 passed so far:
2019-03-20 | proc009 | dept4
2019-03-20 | proc004 | dept3

6.12 Screenshots showing the testing of query 12

Three different queries for Query 12: -

In my Java program, it will show jobs for each type and if specific job type is not available that section will be empty.

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========================================
12
Enter the date job is completed.
2019-01-31
Enter the department to retrieve the jobs.
dept1
The fit jobs completed on date 2019-01-31 in department dept1:
1 | aid01 | 05:30:00.0000000
The Paint jobs completed on date 2019-01-31 in department dept1:
The Cut jobs completed on date 2019-01-31 in department dept1:
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
```

12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
12
Enter the date job is completed.
2019-02-28
Enter the department to retrieve the jobs.
dept3
The fit jobs completed on date 2019-02-28 in department dept3:
5 | aid05 | 03:00:00.0000000
The Paint jobs completed on date 2019-02-28 in department dept3:
4 | aid04 | red | 1 | 03:00:00.0000000
The Cut jobs completed on date 2019-02-28 in department dept3:
7 | aid05 | jmachine01 | 02:00:00.0000000 | m03 | 03:00:00.0000000
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
12
Enter the date job is completed.
2019-03-31
Enter the department to retrieve the jobs.
dept4
The fit jobs completed on date 2019-03-31 in department dept4:
8 | aid09 | 02:00:00.0000000
The Paint jobs completed on date 2019-03-31 in department dept4:
10 | aid07 | green | 3 | 02:30:00.0000000
The Cut jobs completed on date 2019-03-31 in department dept4:

6.13 Screenshots showing the testing of query 13

Three different queries for Query 13: -

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
========================================
13
Enter the lower bound of the category.
1
Enter the upper bound of the category.
5
The customers in the given range of category are
Avion | 4
Ford | 1
Harman | 2
IFB | 5
Nvidia | 3
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the
process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
```

13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
13
Enter the lower bound of the category.
2
Enter the upper bound of the category.
4
The customers in the given range of category are
Avion | 4
Harman | 2
Nvidia | 3
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
13
Enter the lower bound of the category.
1
Enter the upper bound of the category.
3
The customers in the given range of category are
Ford | 1
Harman | 2
Nvidia | 3

6.14 Screenshots showing the testing of query 14

Three different queries for Query 14: -

Job Cut Table before deletion: -

```
690    SELECT * FROM Job_cut;
691
```

Results | Messages

	job_no	job_machine_type	machine_ti…	material_used	cut_labor_time
1	3	jmachine01	04:00:00	m01	07:00:00
2	6	jmachine02	04:00:00	m02	00:30:00
3	7	jmachine01	02:00:00	m03	03:00:00

1. Deleting all cut-jobs whose job-no is in range 1-4 (inclusive):

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible
for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option
17).
18. QUIT (Option 18)
==========================================
14
Enter the lower bound of the cut job-no.
1
Enter the upper bound of the cut job-no.
4
```

Deleted all cut jobs in the given range of job-no.

SQL Job Cut Table after running above program:

```
689
690    SELECT * FROM Job_cut;
691
```

Results | Messages

	job_no	job_machine_type	machine_ti…	material_used	cut_labor_time
1	6	jmachine02	04:00:00	m02	00:30:00
2	7	jmachine01	02:00:00	m03	03:00:00

2. Deleting all cut-jobs whose job-no is in range 8-10 (inclusive):

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
14
Enter the lower bound of the cut job-no.
8
Enter the upper bound of the cut job-no.
10
Deleted all cut jobs in the given range of job-no.

SQL Job Cut Table after running above program:

```
SELECT * FROM Job_cut;
```

Results | Messages

	job_no	job_machine_type	machine_ti…	material_used	cut_labor_time
1	6	jmachine02	04:00:00	m02	00:30:00
2	7	jmachine01	02:00:00	m03	03:00:00

3. Deleting all cut-jobs whose job-no is in range 6-9 (inclusive):

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
14
Enter the lower bound of the cut job-no.
6
Enter the upper bound of the cut job-no.
9
Deleted all cut jobs in the given range of job-no.

SQL Job Cut Table after running above program:

```
690    SELECT * FROM Job_cut;
691
```

Results | Messages

job_no	job_machine_type	machine_time	material_used	cut_labor_time

6.15 Screenshots showing the testing of query 15

Three different queries for Query 15: -

Job Cut Table before changing paint: -

```
688    SELECT * FROM Job_paint;
689
690    SELECT * FROM Job_cut;
691
```

Results | Messages

	job_no	color	volume	paint_labor_time
1	2	yellow	2	01:30:00
2	4	red	1	03:00:00
3	10	green	3	02:30:00

1. Change the color of job 10 to white:

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)

15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
15
Enter the paint job-no.
10
Enter the new color for the job-no.
white
Changed the color of a given paint job.

SQL Job Cut Table after running above program:

```
SELECT * FROM Job_paint;

SELECT * FROM Job_cut;
```

Results | Messages

	job_no	color	volume	paint_labor_time
1	2	yellow	2	01:30:00
2	4	red	1	03:00:00
3	10	white	3	02:30:00

2. Change the color of job 4 to yellow:

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17
18. QUIT (Option 18)
==========
15

Enter the paint job-no.
4
Enter the new color for the job-no.
yellow
Changed the color of a given paint job.

SQL Job Cut Table after running above program:

```
SELECT * FROM Job_paint;

SELECT * FROM Job_cut;
```

Results | Messages

	job_no	color	volume	paint_labor_time
1	2	yellow	2	01:30:00
2	4	yellow	1	03:00:00
3	10	white	3	02:30:00

3. Change the color of job 2 to red:

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
15
Enter the paint job-no.
2
Enter the new color for the job-no.

red
Changed the color of a given paint job.

SQL Job Cut Table after running above program:

```
SELECT * FROM Job_paint;

SELECT * FROM Job_cut;
```

Results | Messages

	job_no	color	volume	paint_labor_time
1	2	red	2	01:30:00
2	4	yellow	1	03:00:00
3	10	white	3	02:30:00

6.16 Screenshots showing the testing of query 16

Screenshot of the Input.txt file: -

Input - Notepad

File Edit Format View Help

```
c1,a1,4
c2,a2,6
c3,a3,9
c4,a4,8
c5,a5,5
```

Implementing Query 16 in Java: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)

7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
16
Enter the file-name to Import data.
Input.txt

Customer table after executing Query 16: -

```
SELECT * FROM Customer;
```

Results | Messages

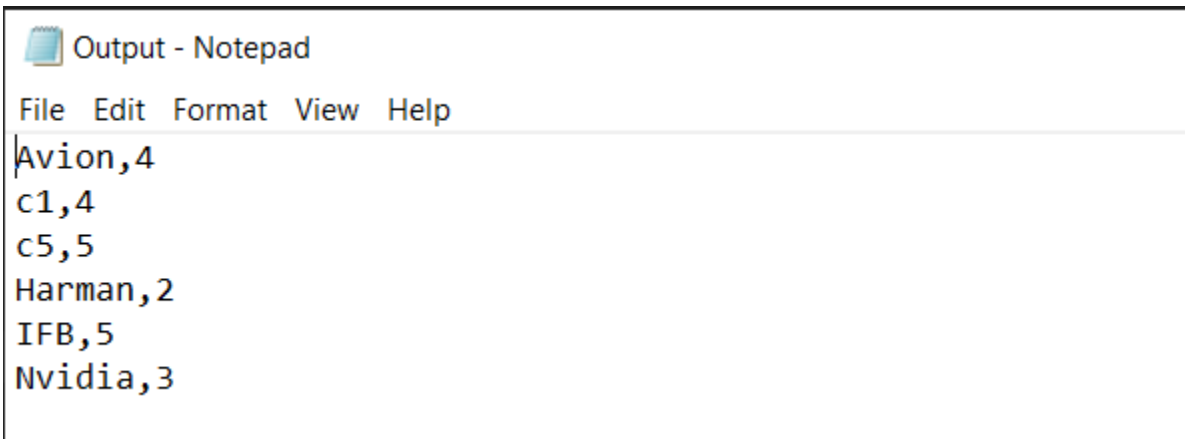
	cname	caddress	category
1	Avion	Chennai	4
2	c1	a1	4
3	c2	a2	6
4	c3	a3	9
5	c4	a4	8
6	c5	a5	5
7	Ford	Dearborn	1
8	Harman	Novi	2
9	IFB	Bangalore	5
10	Nvidia	CA	3

6.17 Screenshots showing the testing of query 17

Implementing Query 17 in Java: -

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
17
Enter the file-name to Export data.
Output.txt
Enter the lower bound of category.
2
Enter the upper bound of category.
5

Screenshot of the Output.txt file: -



6.18 Screenshot showing Quit option

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)

3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
18
You choose to Quit! Bye!

6.19 Checking Errors

1. Error1: Primary Key Error
   Here I am trying to insert data into Customer table with primary key value same as the one existing in table.

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
1

Enter the Customer Name
Ford
Enter the Customer Address
mi
Enter the Customer Category
3
You got an error!. Returning to main menu
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)

2. Error 2: Giving Incorrect date format
   Here I am entering Incorrect date created for Assembly Table

You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option 17).
18. QUIT (Option 18)
==========
3
Enter Assembly ID
aid11
Enter Date Ordered in yyyy-mm-dd format
2019-31-31
Enter Assembly Details
ad11
Enter the Customer Name
Nvidia
You got an error!. Returning to main menu

3. Error 3: Foreign key reference
   Here to enter data into Process table giving new department number that is not there in Department table. (Department-no is a Foreign key in Process table)

```
You have the Following Options to Choose:
1. Enter a new Customer (Option 1)
2. Enter a new Department (Option 2)
3. Enter a new assembly (Option 3)
4. Enter a new process and information related to type (Option 4)
5. Create a new account and associate it with the one to which it is applicable (Option 5)
6. Enter a new Job (Option 6)
7. Enter date job is completed and information related to type of job (Option 7)
8. Enter a transaction details and update all of the affected accounts (Option 8)
9. Retrieve the cost incurred on an assembly-id (Option 9)
10. Retrieve the total labor time within a department for jobs completed during a given date (Option 10)
11. Retrieve the processes through which a given assembly-id has passed so far and the department responsible
for the process (Option 11)
12. Retrieve all jobs completed during the given date in a given department (Option 12)
13. Retrieve the customers whose category is in a given range (Option 13)
14. Delete all cut-jobs whose job-no is in a given range (Option 14)
15. Change the color of a given paint job (Option 15)
16. Import: enter new customers from a data file until the file is empty (Option 16).
17. Export: Retrieve the customers whose category is in a given range and output them to a data file(Option
17).
18. QUIT (Option 18)
==========================================
4
Enter Process ID
proc011
Enter Process Data
pdata11
Enter Department No
d
You got an error!. Returning to main menu
```

Task 7. Web database application and its execution

7.1 Web database application source program and screenshots showing its successful compilations

1. Data Hander: In this code I am creating Datahandler1 class with the properties to get all customers, retrieve customers in the given category and add customer from the form.

```
package jsp_azure_test;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class DataHandler1 {
	private Connection conn;
// Azure SQL connection credentials
	private String server = "kann0002-sql-server.database.windows.net";
	private String database = "cs-dsa-4513-sql-db";
	private String username = "kann0002";
	private String password = "*******";
// Resulting connection string
	final private String url =

	String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;enc
rypt=true;trustServerCertificate=false;hostNameInCertificate=*.database.windows.net;lo
ginTimeout=30;",
				server, database, username, password);
// Initialize and save the database connection
	private void getDBConnection() throws SQLException {
		if (conn != null) {
			return;
		}
		this.conn = DriverManager.getConnection(url);
	}
	// Return the result of selecting everything from the Customer table
	public ResultSet getAllCustomers() throws SQLException {
		getDBConnection(); // Prepare the database connection
		// Prepare the SQL statement
		final String sqlQuery = "SELECT * FROM Customer;";
```

```
		final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
		// Execute the query
return stmt.executeQuery();
	}

	// Return the result of selecting Customers with their category in the given range
from Customer table
	public ResultSet retrieveCustomers(int lower_b, int upper_b) throws
SQLException {
				getDBConnection(); // Prepare the database connection
				//final String sqlQuery = "EXEC proc_13 @lower_b =
'"+lower_b+"', @upper_b = '"+upper_b+"';";
				// Prepare the SQL statement
				final String sqlQuery = "SELECT cname AS name, category
FROM Customer" +
						"	WHERE category >= ? AND category <= ?
ORDER BY 1 ;";
				final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
				// Replace the '?' in the above statement with the given attribute
values
				stmt.setInt(1, lower_b);
				stmt.setInt(2, upper_b);
				// Execute the query
	return stmt.executeQuery();
		}
	// Inserts a record into the Customer table with the given attribute values
	public boolean addCustomer(
				String cname, String address, int category)
						throws SQLException {
		getDBConnection(); // Prepare the database connection
		// Prepare the SQL statement
		final String sqlQuery =
								"INSERT INTO Customer " +
								"(cname, caddress,
category) " +
								"VALUES " +
								"(?, ?, ?)";
		final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
		// Replace the '?' in the above statement with the given attribute values
		stmt.setString(1, cname);
		stmt.setString(2, address);
		stmt.setInt(3, category);
```

```
            // Execute the query, if only one record is updated, then we indicate
success by returning true
            return stmt.executeUpdate() == 1;
        }
}
```

2. Add New customer Form - Code

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Add Customer</title>
</head>
<body>
<h2>Add Customer</h2>
<!--
Form for collecting user input for the new customer record.
Upon form submission, add_new_customer.jsp file will be invoked.
-->
<form action="add_new_customer.jsp">
<!-- The form organized in an HTML table for better clarity. -->
<table border=1>
<tr>
<th colspan="2">Enter the Customer Data:</th>
</tr>
<tr>
<td>Customer Name:</td>
<td><div style="text-align: center;">
<input type=text name=cname>
</div></td>
</tr>
<tr>
<td>Customer Address:</td>
<td><div style="text-align: center;">
<input type=text name=address>
</div></td>
</tr>
<tr>
<td>Category:</td>
<td><div style="text-align: center;">
<input type=text name=category>
</div></td>
</tr>
<tr>
<td><div style="text-align: center;">
<input type=reset value=Clear>
</div></td>
<td><div style="text-align: center;">
<input type=submit value=Insert>
```

```
</div></td>
</tr>
</table>
</form>
</body>
</html>
```

3. Add new Customer code: -

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Query Result</title>
</head>
<body>
<%@page import="jsp_azure_test.DataHandler1"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Array"%>
<%
// The handler is the one in charge of establishing the connection.
DataHandler1 handler1 = new DataHandler1();
// Get the attribute values passed from the input form.
String cname = request.getParameter("cname");
String address = request.getParameter("address");
String category = request.getParameter("category");

/*
* If the user hasn't filled out all the cname, address and category. This is very
simple
checking.
*/
if (cname.equals("") || address.equals("") || category.equals("")) {
response.sendRedirect("add_new_customer_form.jsp");
} else {
int category1 = Integer.parseInt(category);
// Now perform the query with the data from the form.
boolean success = handler1.addCustomer(cname, address, category1);
if (!success) { // Something went wrong
%>
<h2>There was a problem inserting the course</h2>
<%
} else { // Confirm success to the user
%>
<h2>The Customer Details:</h2>
Page 23 of 23
<ul>
<li>Customer Name: <%=cname%></li>
<li>Address: <%=address%></li>
```

```
<li>Category: <%=category%></li>
</ul>
<h2>Was successfully inserted.</h2>
<a href="get_all_customers.jsp">See all Customers.</a>
<%
}
}
%>
</body>
</html>
```

4. Code to get all customers: -

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
      pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
      <head>
      <meta charset="UTF-8">
            <title>Customers</title>
      </head>
      <body>
      <%@page import="jsp_azure_test.DataHandler1"%>
      <%@page import="java.sql.ResultSet"%>
      <%
            // We instantiate the data handler here, and get all the customers from
the database
            final DataHandler1 handler = new DataHandler1();
            final ResultSet customers = handler.getAllCustomers();
      %>
      <!-- The table for displaying all the movie records -->
      <table cellspacing="2" cellpadding="2" border="1">
            <tr> <!-- The table headers row -->
                  <td align="center">
                        <h4>Customer</h4>
                  </td>
                  <td align="center">
                        <h4>Address</h4>
                  </td>
                  <td align="center">
                        <h4>Category</h4>
            </tr>
            <%
                  while(customers.next()) { // For each Customer record returned...
                        // Extract the attribute values for every row returned
                        final String cname = customers.getString("cname");
                        final String caddress = customers.getString("caddress");
                        final String category = customers.getString("category");
                        out.println("<tr>"); // Start printing out the new table
row
                        out.println( // Print each attribute value
```

```
                        "<td align=\"center\">" + cname +
                        "</td><td align=\"center\"> " + caddress+
                        "</td><td align=\"center\"> " + category + "</td>");
                    out.println("</tr>");
            }
            %>
        </table>
</body>
</html>
```

5. Retrieve Customer Form (For Query 13): -

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Retrieve customers given category range</title>
</head>
<body>
<h2>Give Customers Range</h2>
<!--
Form for collecting user input for the new customer record.
Upon form submission, retrieve_customers.jsp file will be invoked.
-->
<form action="retrieve_customers.jsp">
<!-- The form organized in an HTML table for better clarity. -->
<table border=1>
<tr>
<th colspan="2">Enter the Category Range:</th>
</tr>
<tr>
<td>Lower Bound:</td>
<td><div style="text-align: center;">
<input type=text name=lower_b>
</div></td>
</tr>
<tr>
<td>Upper Bound:</td>
<td><div style="text-align: center;">
<input type=text name=upper_b>
</div></td>
</tr>
<tr>
<td><div style="text-align: center;">
<input type=reset value=Clear>
</div></td>
<td><div style="text-align: center;">
<input type=submit value=Insert>
</div></td>
</tr>
</table>
</form>
```

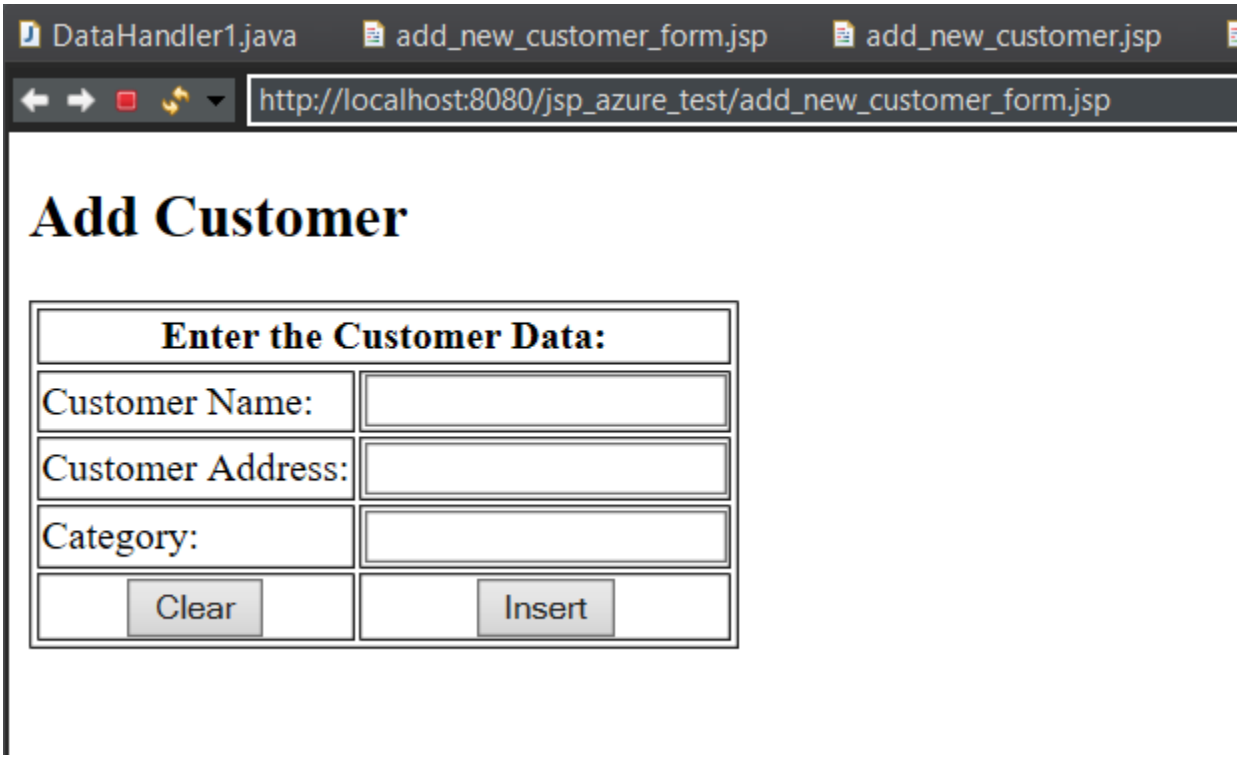
```
	</body>
	</html>
```

6. Code to retrieve customers: -

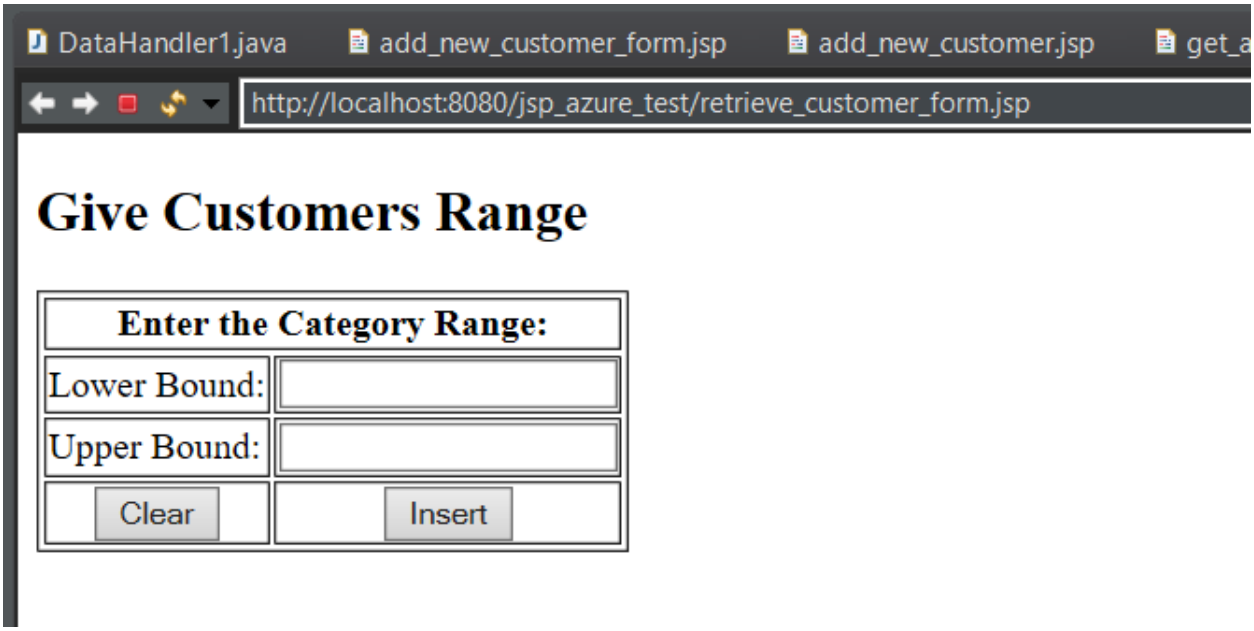
```
	<%@ page language="java" contentType="text/html; charset=UTF-8"
	pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
	<head>
	<meta charset="UTF-8">
		<title>Customers</title>
	</head>
	<body>
	<%@page import="jsp_azure_test.DataHandler1"%>
	<%@page import="java.sql.ResultSet"%>
	<%
		// We instantiate the data handler here, and get all the customers from
the database
		final DataHandler1 handler = new DataHandler1();
	// Get the attribute values passed from the input form.
		String lower_b = request.getParameter("lower_b");
		String upper_b = request.getParameter("upper_b");


		if (lower_b.equals("") || upper_b.equals("")) {
		response.sendRedirect("retrieve_customer_form.jsp");
		} else {
		int lower_b1 = Integer.parseInt(lower_b);
		int upper_b1 = Integer.parseInt(upper_b);
		// Now perform the query with the data from the form.
		final ResultSet customers = handler.retrieveCustomers(lower_b1,
upper_b1);
	%>
	<!-- The table for displaying all the Customer records -->
	<table cellspacing="2" cellpadding="2" border="1">
		<tr> <!-- The table headers row -->
			<td align="center">
				<h4>Customer</h4>
			</td>
			<td align="center">
				<h4>category</h4>
			</td>
		<%
			while(customers.next()) { // For each Customer record returned...
				// Extract the attribute values for every row returned
				final String cname = customers.getString("name");
				final String category = customers.getString("category");
				out.println("<tr>"); // Start printing out the new table
row
				out.println( // Print each attribute value
					"<td align=\"center\">" + cname +
```

```
                                "</td><td align=\"center\"> " + category + "</td>");
                        out.println("</tr>");
                }
        }
                %>
        </table>
    </body>
</html>
```

Screenshot after executing Add-new_customer_form: -

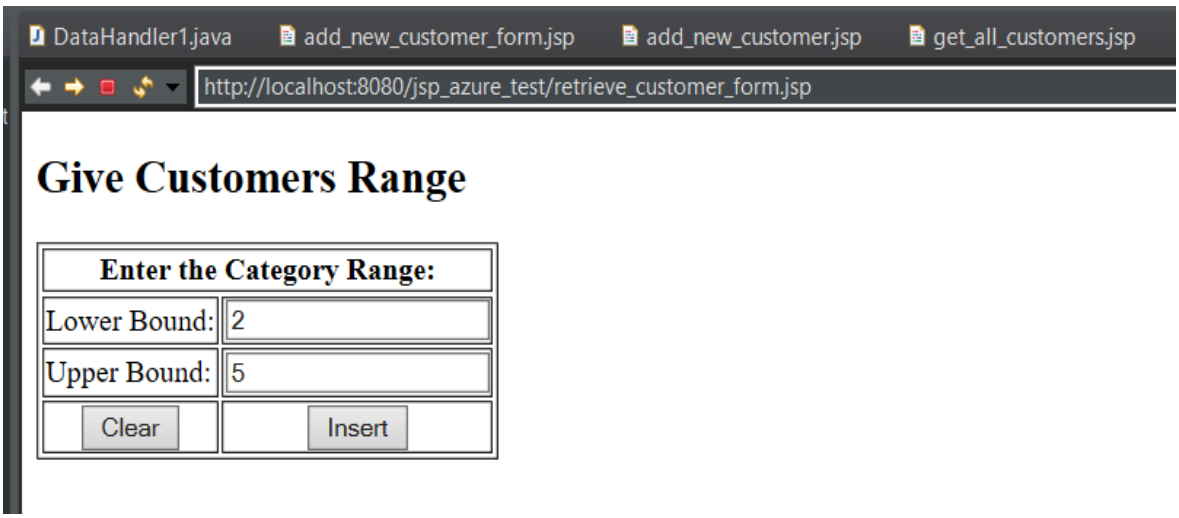


Screenshot after executing retrieve_customer_form: -

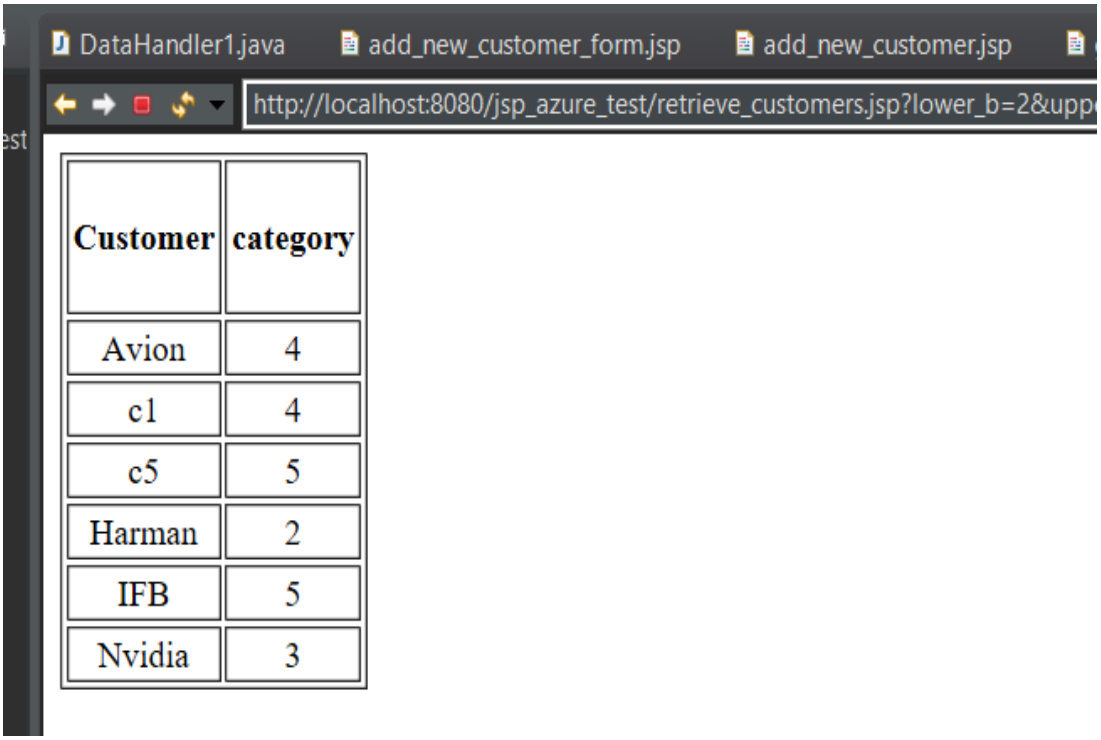


7.2 Screenshots showing the testing of the Web database application

1. First Query 13: -
   Retrieve Form Screenshot with values

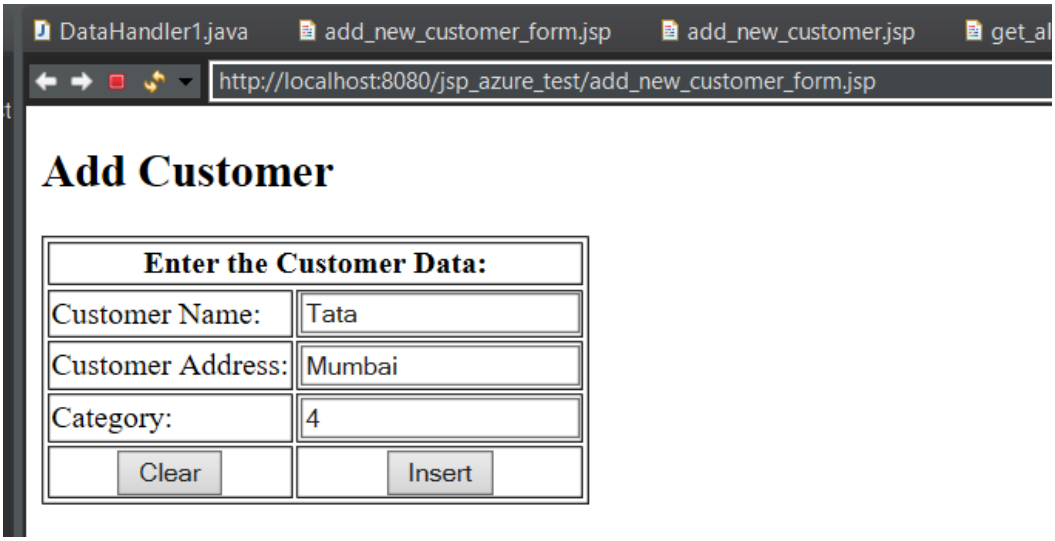


   Result after retrieving customers

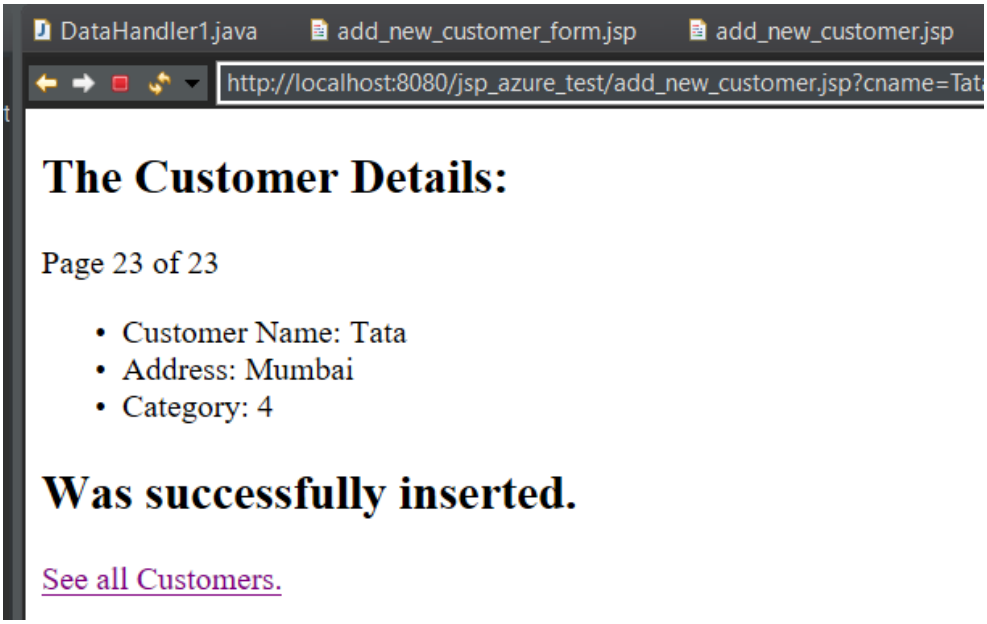


Customer	category
Avion	4
c1	4
c5	5
Harman	2
IFB	5
Nvidia	3

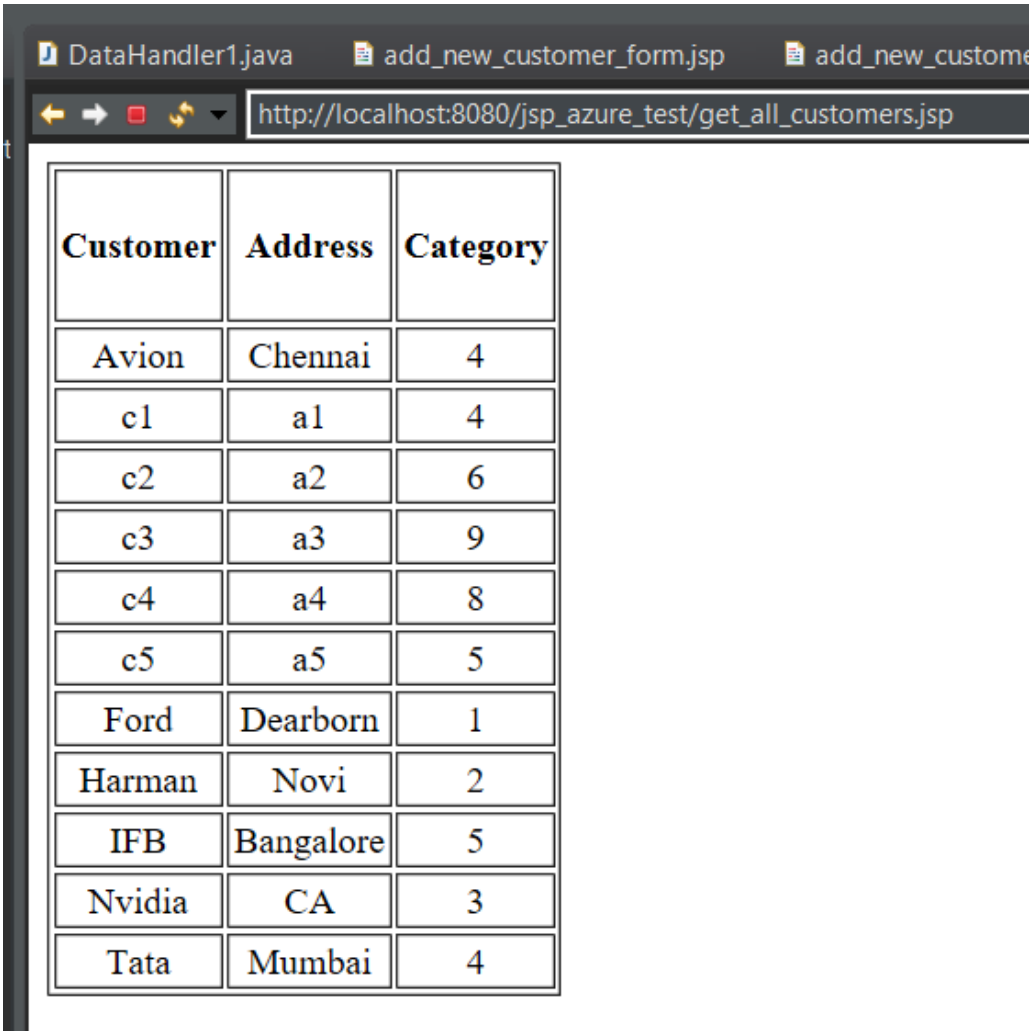
2. <u>Second Query 1: -</u>
   Add customer Form Screenshot with values



Result after adding new Customer:



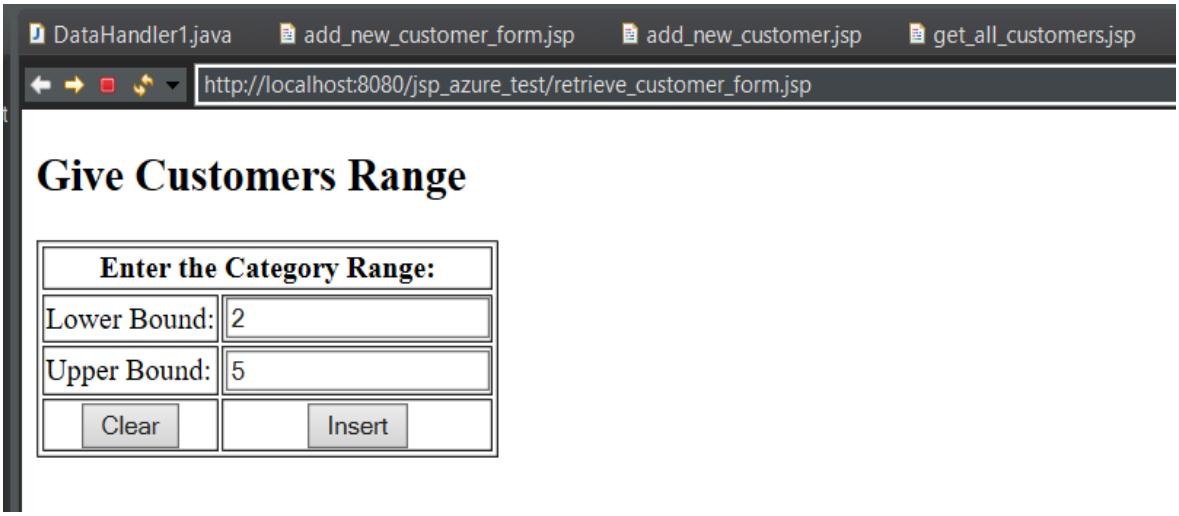
Viewing all customers: -



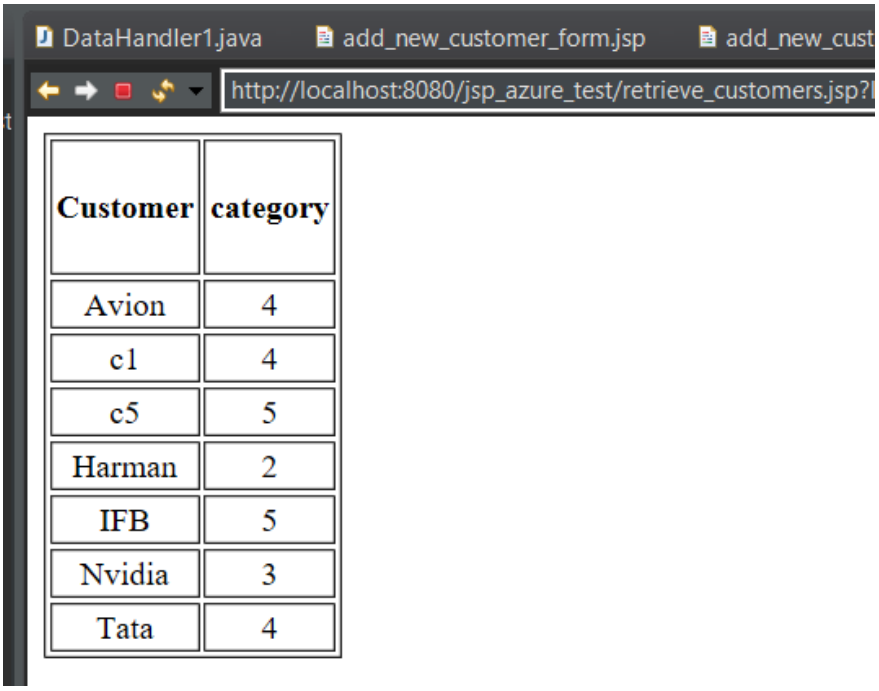
Customer	Address	Category
Avion	Chennai	4
c1	a1	4
c2	a2	6
c3	a3	9
c4	a4	8
c5	a5	5
Ford	Dearborn	1
Harman	Novi	2
IFB	Bangalore	5
Nvidia	CA	3
Tata	Mumbai	4

3. <u>Third Query 13: -</u>

   Retrieve Form Screenshot with values



Result after retrieving customers



Customer	category
Avion	4
c1	4
c5	5
Harman	2
IFB	5
Nvidia	3
Tata	4

THANK YOU!