2019

Individual project

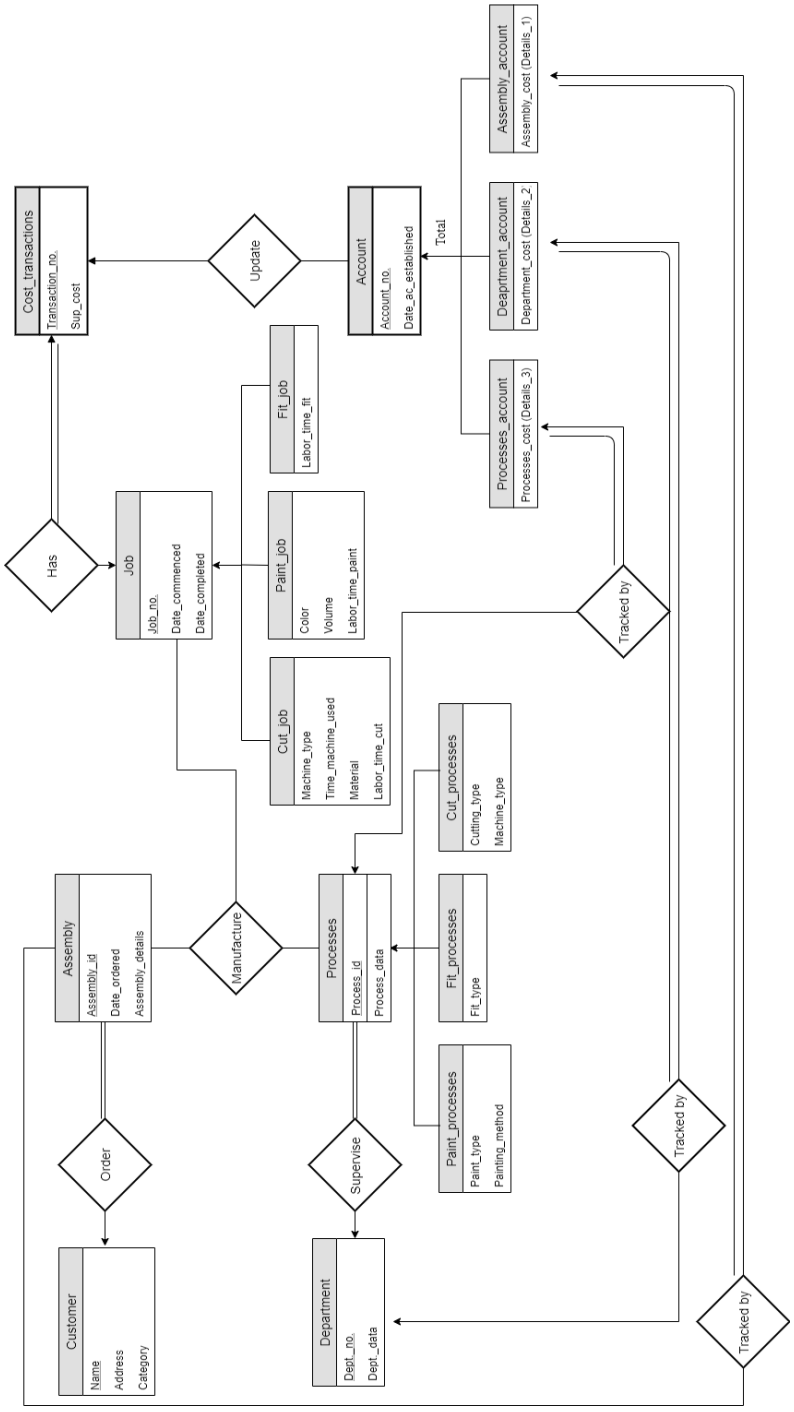
JOB-SHOP ACCOUNTING DATABASE MANAGEMENT SYSTEMS

GUPTA, ABHISHEK KUMAR | ABHIGUPTA@OU.EDU

Contents

Chapter 1

1.1 ER diagram



1.2 Relational Database System

Customer (Name, Address, Category)

Assembly (Assembly_id, name, Date_ordered, Assembly_details)

Department (Dept_no, Dept._data)

Process (process_id, dept_no, process_data)

Process_paint (Process_id, Paint_type, Painitng_method)

Process_fit (Process_id, Fit_type)

Process_cut (Process_id, Cutting_type, Machine_type)

Job (Job_no., Date_commenced, Date_completed

Cut_job (Job_no., Machine_type, Time_machine_used, Material, Labor_time_cut)

Paint_job (Job_no., Color, Volume, Labor_time_paint)

Fit_job (Job_no., Labor_time_fit)

Manufacture (Assembly_id, Process_id, job_no)

Account (Account_no, Date_ac_established)

Assembly_account (Account_no, assembly_id, Assembly_cost)

Department_account (Account_no, dept_no, Department_cost)

Processes_account (Account_no, process_id, processses_cost)

Cost_transactions (Transaction_no, Job_no, Sup_cost, account_no)

Update (transaction no, account no)

Chapter 2

Table name	Attribute name	types	size	constraints

Customer	name	varchar(80)	82	Primary key
	address	varchar(80)	82	
	category	int	4	
Assembly	assembly_id	int	4	Primary key
	date ordered	date	3	
	assembly details	varchar(80)	82	
	name	varchar(80)	82	Foreign key
Department	dept no	int	4	primary key
	dept data	varchar(80)	82	
process	process id	int	4	Primary key
	pricess data	varchar(80)	82	
	dept no	int	4	
paint process	process id	int	4	Primary key
	paint type	varchar(80)	82	
	painting method	varchar(80)	82	
fit process	process id	int	4	Primary key
	fit type	varchar(80)	82	
cut process	process id	int	4	Primary key
	cutting type	varchar(80)	82	
	machine type	varchar(80)	82	
job	job no	int	4	Primary key
	date commenced	date	3	
	date completed	date	3	
cut job	job no	int	4	Primary key
	machine type	varchar(80)	82	
	machine time used	varchar(80)	82	
	material	varchar(80)	82	
	labor time cut	numeric (5,2)	5	
paint job	job no	int	4	Primary key
	color	varchar(80)	82	
	volume	int	4	
	labor time paint	numeric (5,2)	5	

fit job	job no	int	4	Primary key
	labor time fit	numeric (5,2)	5	
manufacture	assembly id	int	4	Primary key
	process id	int	4	Primary key
	job no	int	4	Primary key
account	account no	int	4	Primary key
	date account established	date	3	
assembly account	account no	int	4	Primary key
	assembly cost	numeric (5,2)	5	
	assembly id	int	4	Foreign key
department account	account no	int	4	Primary key
	department cost	numeric (5,2)	5	
	dept no	int	4	Foreign key
process account	account no	int	4	Primary key
	process cost	numeric (5,2)	5	
	process id	int	4	Foreign key
cost transactions	transaction no	int	4	Primary key
	sup cost	numeric (5,2)	5	
	job no	int	4	Foreign key

Chapter 3:

Chapter 3.1

Table name	Query and type	Search Key	Query frequency	Selected file organization	Justifications
Customer	1, insertion	-	30	b tree	faster for searching as high frequency of search is involved
	13, range search	category	100		
Assembly	3, insertion	customer name	40	b tree	faster for searching as high frequency of search is involved
	5, random search	assembly id	10		
	6, insertion and random search	assembly id	50		
	8, insertion, random search	assembly id	50		
	9, random search	assembly id	200		
	12, random search	assembly id	20		
Department	2, insertion		infrequent	b tree	faster for searching as high frequency of search is involved
	5, random search	dept no	10		
	8, insertion, random search	dept no	50		
	10, random search	dept no	20		
	11, random search	dept no	100		
	12, random search	dept no	20		
process	4, insertion, random search		infrequent	b tree	faster for searching as high frequency of search is involved
	5, random search	process id	10		
	6, insertion and random search	process id	50		
	8, insertion, random search	process id	50		
	11, random search	process id	100		
paint process	4, insertion, random search		infrequent	Dynamic hashing	Because its good for insertion
fit process	4, insertion, random search		infrequent	Dynamic hashing	Because its good for insertion
cut process	4, insertion, random search		infrequent	Dynamic hashing	Because its good for insertion
job	6, insertion		50	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	6, insertion		50		
	7, insertion		50		
	7, random search	job no	50		

	10, random search	job no	20		
	12, random search	job no	20		
	14, range search	job no	infrequent		
cut job	7, insertion		50	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	7, random search	job no	50		
	10, random search	cut job time	20		
	14, deletion		infrequent		
paint job	7, insertion		50	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	7, random search	job no	50		
	10, random search	paint job time	20		
	15, random search and insertion	job no	1/7		
fit job	7, insertion		50	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	7, random search	job no	50		
	10, random search	fit job time	20		
account	5, insertion		10	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
assembly acocunt	5, insertion		10	Dynamic hashing	
	8, insertion		50		
	9, insertion		200		
department account	5, insertion		10	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	8, insertion		50		
process account	5, insertion		10	Dynamic hashing	Because its good for insertion as high volume insertion is ongoing
	8, insertion		50		
cost transaction	8, insertion		50	Dynamic hashing	Because its faster

Chapter 3.2

In SQL database I haven't specifically selected any storage structure. It is by default by Azure database.

Chapter 4: SQL statements and screenshot of tables creates

Code to create tables are below:

```
-- create table customer
CREATE TABLE Customer(
    c_name VARCHAR(80) PRIMARY KEY,
    c_address VARCHAR(80),
    category INTEGER,
)
Create Index a on Customer(c_name)

--create assembly
CREATE TABLE js_Assembly(
    assembly_id INTEGER PRIMARY KEY,
    date_ordered DATE,
    assembly_details VARCHAR(80),
    c_name VARCHAR(80),
    FOREIGN KEY (c_name) REFERENCES customer
)
Create Index b on js_Assembly(assembly_id)

-- create table department
CREATE TABLE Department(
    dept_no INTEGER PRIMARY KEY,
    dept_data VARCHAR(80),
)
Create Index c on department(dept_no)

CREATE TABLE Processes(
    process_id integer PRIMARY KEY,
    process_data VARCHAR(80),
    dept_no INTEGER,
    FOREIGN KEY (dept_no) REFERENCES department
)
Create Index d on processes(process_id)

CREATE TABLE Paint_processes(
    process_id INTEGER PRIMARY KEY,
    paint_type VARCHAR(80),
    painting_method VARCHAR(80),
    FOREIGN KEY (process_id) REFERENCES processes
)
Create Index e on paint_processes(process_id)

CREATE TABLE Fit_processes(
    process_id INTEGER PRIMARY KEY,
    fit_type VARCHAR(80),
    FOREIGN KEY (process_id) REFERENCES processes
)
```

```
Create Index f on fit_processes(process_id)

CREATE TABLE Cut_processes(
    process_id INTEGER PRIMARY KEY,
    cutting_type VARCHAR(80),
    machine_type VARCHAR(80),
    FOREIGN KEY (process_id) REFERENCES processes
)
Create Index g on cut_processes(process_id)

CREATE TABLE Job(
    job_no INTEGER PRIMARY KEY,
    date_commenced DATE,
    date_completed DATE
)
Create Index f on job(job_no)

CREATE TABLE Cut_job(
    job_no INTEGER PRIMARY KEY,
    machine_type VARCHAR(80),
    time_machine_used VARCHAR(80),
    material VARCHAR(80),
    labor_time_cut NUMERIC(5,2),
    FOREIGN key (job_no) REFERENCES job
)
Create Index f on cut_job(job_no)

CREATE TABLE Paint_job(
    job_no INTEGER PRIMARY KEY,
    color VARCHAR(80),
    volume NUMERIC(5,2),
    labor_time_paint NUMERIC(5,2),
    FOREIGN key (job_no) REFERENCES job
)
Create Index g on paint_job(job_no)


CREATE TABLE Fit_job(
    job_no INTEGER PRIMARY KEY,
    labor_time_fit NUMERIC(5,2),
    FOREIGN key (job_no) REFERENCES job
)
Create Index h on fit_job(job_no)


CREATE TABLE manufacture(
    assembly_id INTEGER,
    process_id INTEGER ,
    job_no INTEGER,
    PRIMARY KEY (assembly_id, process_id, job_no),
    FOREIGN KEY (assembly_id) REFERENCES js_Assembly,
    FOREIGN KEY (process_id) REFERENCES processes,
    FOREIGN key (job_no) REFERENCES job
)
```

```
Create Index g on manufacture(job_no)


CREATE TABLE Account(
    account_no INTEGER PRIMARY KEY,
    date_ac_established DATE
)
Create Index g on account(account_no)


CREATE TABLE Assembly_account(
    account_no INTEGER PRIMARY KEY,
    assembly_cost NUMERIC(5,2),
    assembly_id INTEGER,
    FOREIGN KEY (account_no) REFERENCES account,
    FOREIGN KEY (assembly_id) REFERENCES js_Assembly
)
Create Index h on assembly_account(account_no)

CREATE TABLE Department_account(
    account_no INTEGER PRIMARY KEY,
    department_cost NUMERIC(5,2),
    dept_no INTEGER
    FOREIGN KEY (account_no) REFERENCES account,
    FOREIGN KEY (dept_no) REFERENCES department
)
Create Index i on department_account(account_no)


CREATE TABLE Process_account(
    account_no INTEGER PRIMARY KEY,
    process_cost NUMERIC(5,2),
    process_id integer,
    FOREIGN KEY (process_id) REFERENCES processes,
    FOREIGN KEY (account_no) REFERENCES account
)
Create Index j on process_account(account_no)


CREATE TABLE Cost_transactions(
    transaction_no int PRIMARY key,
    sup_cost NUMERIC(5,2),
    job_no INTEGER,
    FOREIGN KEY (job_no) REFERENCES job
)
Create Index k on cost_transactions(transaction_no)


CREATE TABLE update_ac(
    transaction_no INTEGER PRIMARY KEY,
    account_no INTEGER
    FOREIGN KEY (transaction_no) REFERENCES cost_transactions,
    FOREIGN KEY (account_no) REFERENCES account
)
```

```
Create Index l on update_ac(transaction_no)
```

Code to select aal the tables after creating them :

```
SELECT * from customer
SELECT * from js_assembly
SELECT * from department
SELECT * from processes
SELECT * from paint_processes
SELECT * from fit_processes
SELECT * from cut_processes
SELECT * from manufacture
SELECT * from job
SELECT * from cut_job
SELECT * from paint_job
SELECT * from fit_job
SELECT * from account
SELECT * from assembly_account
SELECT * from department_account
SELECT * from process_account
SELECT * from Cost_transactions
SELECT * from update_ac
```

Created tables:

	c_name	c_address	category

	assembly_id	date_ordered	assembly_details	c_name

	dept_no	dept_data

	process_id	process_data	dept_no

	process_id	paint_type	painting_method

	process_id	fit_type

	process_id	cutting_type	machine_type

	assembly_id	process_id	job_no

	job_no	date_commenced	date_completed

	job_no	machine_type	time_machine_us…	material	labor_time_cut

	job_no	color	volume	labor_time_paint

	job_no	labor_time_fit

	account_no	date_ac_establi…

	account_no	assembly_cost	assembly_id

	account_no	department_cost	dept_no

	account_no	process_cost	process_id

	transaction_no	sup_cost	job_no

	transaction_no	account_no

Chapter 5: Java source file and screenshots

SQL statements for all queries for 1-15 and created procedures for Java application

```
/* query 1 procedure for creating new customer */
GO
CREATE PROCEDURE new_customer
    @c_name VARCHAR(80),
    @c_address VARCHAR (80),
    @category INTEGER

AS
BEGIN
    INSERT into Customer(
        c_name,
        c_address,
        category
    )

    VALUES
        (@c_name, @c_address, @category)

    END
```

```
GO

/* query 2 procedure for creating new department */
GO
CREATE PROCEDURE new_department
    @dept_no INTEGER,
    @dept_data VARCHAR(80)

AS
BEGIN
    INSERT into Department(
        dept_no,
        dept_data
    )

    VALUES
        (@dept_no, @dept_data)

    END


/* query 3 procedure for creating new assembly */
GO
CREATE PROCEDURE new_assembly
    @assembly_id INTEGER,
    @date_ordered DATE,
    @assembly_details VARCHAR(80),
    @name VARCHAR(80)

AS
BEGIN
    INSERT into js_Assembly(
        assembly_id,
        date_ordered,
        assembly_details,
        c_name
    )

    VALUES
        (@assembly_id, @date_ordered, @assembly_details, @name)

    END

/* query 4 procedure for entering new process */
GO
CREATE PROCEDURE new_process
    @process_id integer,
    @process_data VARCHAR(80),
    @dept_no INTEGER

AS
BEGIN
    INSERT into Processes(
```

```
        process_id,
        process_data,
        dept_no
    )

    VALUES
        (@process_id, @process_data, @dept_no)

    END

/* query 5 */

GO
CREATE PROCEDURE new_account
   @account_no INTEGER,
   @date_ac_established DATE

AS
BEGIN
    INSERT into Account(
        account_no,
        date_ac_established
    )
    VALUES
    (@account_no, @date_ac_established)
    END

GO
CREATE PROCEDURE new_process_account
    @account_no INTEGER,
    @process_cost NUMERIC(5,2),
    @process_id integer

AS
BEGIN
    INSERT into process_account(
        account_no,
        process_cost,
        process_id
    )
    VALUES
    (@account_no,@process_cost,@process_id)
    END
DROP PROCEDURE new_assembly_account
GO
CREATE PROCEDURE new_assembly_account
    @account_no INTEGER,
    @assembly_cost NUMERIC(5,2),
    @assembly_id INTEGER
AS
BEGIN
    INSERT INTO Assembly_account(
        account_no,
        assembly_cost,
```

```
        assembly_id
    )
    VALUES
    (@account_no, @assembly_cost, @assembly_id)
END

DROP PROCEDURE new_assembly_account


GO
CREATE PROCEDURE new_department_account
    @account_no INTEGER,
    @department_cost NUMERIC(5,2),
    @dept_no INTEGER
AS
BEGIN
    INSERT INTO department_account(
        account_no ,
        department_cost ,
        dept_no
    )
    VALUES
    (@account_no, @department_cost, @dept_no)
    END

/* query 6*/
drop PROCEDURE new_job
GO
CREATE PROCEDURE new_job
    @new_job_no integer,
    @new_date_commenced DATE,
    @new_assembly_id INTEGER,
    @new_process_id INTEGER

As BEGIN
    INSERT into Job(job_no, date_commenced)
    VALUES(@new_job_no, @new_date_commenced)
    INSERT into manufacture (assembly_id, process_id, job_no)
    VALUES(@new_assembly_id, @new_process_id, @new_job_no)

END

/* query 7 */
--DROP procedure completion_job
GO
CREATE PROCEDURE completion_job
    @job_no INTEGER,
    @date_completed DATE

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

```
GO
CREATE PROCEDURE completed_cut_job
    @job_no INTEGER,
    @machine_type VARCHAR(80),
    @time_machine_used VARCHAR(80),
    @material VARCHAR(80),
    @labor_time_cut NUMERIC(5,2)

AS BEGIN
    INSERT into Cut_job (job_no, machine_type, time_machine_used, material, labor_time_cut)
    VALUES(@job_no, @machine_type, @time_machine_used, @material, @labor_time_cut)
END

GO
CREATE PROCEDURE completed_paint_job
    @job_no INTEGER,
    @color VARCHAR(80),
    @volume NUMERIC(5,2),
    @labor_time_paint NUMERIC(5,2)

AS BEGIN
    INSERT into paint_job (job_no, color, volume, labor_time_paint)
    VALUES(@job_no, @color, @volume, @labor_time_paint )
END
--drop PROCEDURE completed_fit_job
GO
CREATE PROCEDURE completed_fit_job
    @job_no INTEGER,
    @labor_time_fit NUMERIC(5,2)

AS BEGIN
    INSERT into fit_job(job_no, labor_time_fit)
    VALUES(@job_no, @labor_time_fit)
END

/* Query 8 */
--DROP PROCEDURE cost_transaction_entry

GO
CREATE PROCEDURE cost_transaction_entry
   @transaction_no INTEGER,
   @sup_cost NUMERIC(5,2),
   @transaction_job_no INTEGER

AS BEGIN
   INSERT into Cost_transactions(transaction_no, sup_cost, job_no)
   VALUES (@transaction_no, @sup_cost, @transaction_job_no)
END

drop PROCEDURE update_process_ac
GO
CREATE PROCEDURE update_process_ac
    @tran_no INTEGER,
```

```
    @process_sup_cost NUMERIC(5,2)


AS BEGIN
    DECLARE @process_ac integer
    SET @process_ac =
    (select account_no
    FROM Process_account
    WHERE process_id IN(
            select process_id
            from manufacture
            where job_no in (
                select job_no
                from Cost_transactions
                WHERE transaction_no = @tran_no
                ))
    )

    UPDATE Process_account
    SET process_cost = process_cost + @process_sup_cost
    WHERE account_no = @process_ac

END

drop PROCEDURE update_assembly_ac

GO
CREATE PROCEDURE update_assembly_ac
    @tran_no INTEGER,
    @assembly_sup_cost NUMERIC(5,2)


AS BEGIN
    DECLARE @assembly_ac integer
    SET @assembly_ac =
    (select account_no
    FROM assembly_account
    WHERE assembly_id IN(
            select assembly_id
            from manufacture
            where job_no in (
                select job_no
                from Cost_transactions
                WHERE transaction_no = @tran_no
                ))
    )

    UPDATE assembly_account
    SET assembly_cost = assembly_cost + @assembly_sup_cost
    WHERE account_no = @assembly_ac

END

--drop PROCEDURE update_dept_ac
```

```
GO
CREATE PROCEDURE update_dept_ac
    @tran_no INTEGER,
    @dept_sup_cost NUMERIC(5,2)


AS BEGIN
    DECLARE @dept_ac integer
    SET @dept_ac =
    (select account_no
    FROM department_account
    WHERE dept_no IN(
        select dept_no
        from Department
        where dept_no IN (
            select dept_no
            from Processes
            where process_id in (
                select process_id
                from manufacture
                WHERE job_no IN (
                    select job_no
                    FROM Cost_transactions
                    WHERE transaction_no = @tran_no
                )
                )))
    )

    UPDATE department_account
    SET department_cost = department_cost + @dept_sup_cost
    WHERE account_no = @dept_ac

END

/* query 9 */

GO
CREATE PROCEDURE ret_assembly_cost
    @ret_assembly_id INTEGER

as BEGIN
   SELECT Assembly_cost
   FROM Assembly_account
   WHERE assembly_id = @ret_assembly_id

end

/* query 10 */

GO
CREATE PROCEDURE tot_labor_time
    @dept_num INTEGER,
    @start_compl_date VARCHAR (80),
```

```
    @end_compl_date VARCHAR (80)

AS
BEGIN

DECLARE @total_labor_cut_time NUMERIC (5,2)
SET @total_labor_cut_time =(
SELECT sum(labor_time_cut)
FROM Cut_job
WHERE job_no in(SELECT job_no
FROM Job
WHERE date_completed BETWEEN @start_compl_date AND @end_compl_date
INTERSECT
SELECT job_no
FROM Job
WHERE job_no IN (
    select job_no
    FROM manufacture
    where process_id IN(
        select process_id
        FROM Processes
        WHERE dept_no = @dept_num))))

DECLARE @total_labor_paint_time NUMERIC (5,2)
SET @total_labor_paint_time =(
SELECT sum(labor_time_paint)
FROM paint_job
WHERE job_no in(SELECT job_no
FROM Job
WHERE date_completed BETWEEN @start_compl_date AND @end_compl_date
INTERSECT
SELECT job_no
FROM Job
WHERE job_no IN (
    select job_no
    FROM manufacture
    where process_id IN(
        select process_id
        FROM Processes
        WHERE dept_no = @dept_num))))

DECLARE @total_labor_fit_time NUMERIC (5,2)
SET @total_labor_fit_time =(
SELECT sum(labor_time_fit)
FROM fit_job
WHERE job_no in(SELECT job_no
FROM Job
WHERE date_completed BETWEEN @start_compl_date AND @end_compl_date
INTERSECT
SELECT job_no
FROM Job
WHERE job_no IN (
    select job_no
    FROM manufacture
```

```
    where process_id IN(
        select process_id
        FROM Processes
        WHERE dept_no = @dept_num))))

DECLARE @total_labor_time NUMERIC(5,2)
SET @total_labor_time = @total_labor_cut_time + @total_labor_fit_time + @total_labor_paint_tim
e

SELECT @total_labor_time

END

/* query 11 */

GO
CREATE PROCEDURE ret_process
    @assembly_id_process INTEGER

AS BEGIN

Select process_id, dept_no
From Processes
Where process_id in (
    Select process_id
    from manufacture
    where assembly_id = @assembly_id_process)

END

/* query 12 */
--DROP PROCEDURE ret_jobs
Go
CREATE PROCEDURE ret_jobs
    @start_comp_date VARCHAR(80),
    @end_comp_date VARCHAR(80),
    @department_no INTEGER

AS
BEGIN

SELECT job_no
FROM Job
WHERE date_completed BETWEEN @start_comp_date AND @end_comp_date
INTERSECT
SELECT job_no
FROM Job
WHERE job_no IN (
    select job_no
    FROM manufacture
    where process_id IN(
        select process_id
        FROM Processes
        WHERE dept_no = @department_no
```

```
    )
)
END

/* query 13 */

go
CREATE PROCEDURE ret_customer
 @start_cat INTEGER,
 @end_cat INTEGER

AS BEGIN
  SELECT c_name
  FROM Customer
  WHERE category BETWEEN @start_cat and @end_cat

END

/* query 14 */

GO
CREATE PROCEDURE delete_cut_jobs
  @start_job_no INTEGER,
  @end_job_no INTEGER

AS BEGIN
   DELETE from Cut_job
   WHERE job_no BETWEEN @start_job_no and @end_job_no

END

EXEC delete_cut_jobs @start_job_no = 1, @end_job_no=3;

/* query 15 */
GO
CREATE PROCEDURE change_color
    @color VARCHAR(80),
    @col_job_no INTEGER

AS
BEGIN
    UPDATE Paint_job
    SET
    color = @color
    WHERE job_no = @col_job_no
END
```

Java code to run procedures created in SQL:

```
import java.sql.Connection;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.CallableStatement;
import java.util.Scanner;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

public class IP_Chapter5_Gupta_Abhishek
{

//Defining class for Procedure  to enter a new customer details
       public static void new_customer(String c_name, String c_address, int category)
       {
         String query = "{ call new_customer(?, ?, ?) }"; //Calling procedure #1
         ResultSet rs;

       // Connect to database
         final String hostName = "userID-sql-server.database.windows.net";
         final String dbName = "my-sql-db";
         final String user = "userID";
         final String password = "Password";
         final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
         try
         (final Connection connection = DriverManager.getConnection(url))
         {
              final String schema = connection.getSchema();
              System.out.println("Successful connection - Schema:" + schema);
              System.out.println("query1: enter the new customer");
             try
             (final Connection conn = DriverManager.getConnection(url);
                    CallableStatement stmt = conn.prepareCall(query))
             {

       //Providing parameters to the procedure
              stmt.setString(1, c_name);
              stmt.setString(2, c_address);
              stmt.setInt(3, category);
              rs = stmt.executeQuery();                    //Query execution
              }
             catch (SQLException ex)
             {
              System.out.println(ex.getMessage());
              }
             }
         catch (SQLException e)
```

```
        {
                throw new RuntimeException(e);
            }
        }

//Defining class for Procedure to enter new department
       public static void new_department(int dept_no, String dept_data)
       {
        String query = "{ call new_department(?, ?) }"; //Calling procedure #1
        ResultSet rs;

       // Connect to database
        final String hostName = "userID-sql-server.database.windows.net";
        final String dbName = "my-sql-db";
        final String user = "userID";
        final String password = "Password";
        final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
        try
        (final Connection connection = DriverManager.getConnection(url))
        {
             final String schema = connection.getSchema();
             System.out.println("Successful connection - Schema:" + schema);
             System.out.println("query2: enter the new department");
            try
            (final Connection conn = DriverManager.getConnection(url);
                    CallableStatement stmt = conn.prepareCall(query))
            {

       //Providing parameters to the procedure
             stmt.setInt(1, dept_no);
             stmt.setString(2, dept_data);
             rs = stmt.executeQuery();                    //Query execution
             }
            catch (SQLException ex)
            {
             System.out.println(ex.getMessage());
             }
            }
        catch (SQLException e)
        {
                throw new RuntimeException(e);
            }
        }


//Defining class for Procedure to enter new assembly
             public static void new_assembly(int assembly_id, String date_ordered,
String assembly_details, String name)
             {
              String query = "{ call new_assembly(?, ?,?,?) }"; //Calling procedure
#1
              ResultSet rs;
```

```
	// Connect to database
		final String hostName = "userID-sql-server.database.windows.net";
		final String dbName = "my-sql-db";
		final String user = "userID";
		final String password = "Password";
		final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
		try
		(final Connection connection = DriverManager.getConnection(url))
		{
		 final String schema = connection.getSchema();
		 System.out.println("Successful connection - Schema:" + schema);
		 System.out.println("query3: enter the new assembly");
		try
		(final Connection conn = DriverManager.getConnection(url);
			CallableStatement stmt = conn.prepareCall(query))
		{

	//Providing parameters to the procedure
		 stmt.setInt(1, assembly_id);
		 stmt.setString(2, date_ordered);
		 stmt.setString(3, assembly_details);
		 stmt.setString(4, name);
		 rs = stmt.executeQuery();                    //Query execution
		 }
		catch (SQLException ex)
		{
		 System.out.println(ex.getMessage());
		 }
		}
		catch (SQLException e)
		{
			throw new RuntimeException(e);
		 }
		}

//Defining class for Procedure  to enter a new process
	public static void new_process(int process_id, String process_data, int
dept_no)
		{
		 String query = "{ call new_process(?, ?, ?) }"; //Calling procedure #1
		 ResultSet rs;

	// Connect to database
		final String hostName = "userID-sql-server.database.windows.net";
		final String dbName = "my-sql-db";
		final String user = "userID";
		final String password = "Password";
		final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
```

```
                try
                (final Connection connection = DriverManager.getConnection(url))
                {
                     final String schema = connection.getSchema();
                     System.out.println("Successful connection - Schema:" + schema);
                     System.out.println("query4: enter the new process");
                    try
                    (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                    {

        //Providing parameters to the procedure
                     stmt.setInt(1, process_id);
                     stmt.setString(2, process_data);
                     stmt.setInt(3, dept_no);
                     rs = stmt.executeQuery();                    //Query execution
                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
                catch (SQLException e)
                {
                           throw new RuntimeException(e);
                     }
                }

//Defining class for Procedure to enter new account
              public static void new_account(int account_no, String
date_ac_established)
              {
                String query = "{ call new_account(?, ?) }"; //Calling procedure #1
                ResultSet rs;

        // Connect to database
                final String hostName = "userID-sql-server.database.windows.net";
                final String dbName = "my-sql-db";
                final String user = "userID";
                final String password = "Password";
                final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
                try
                (final Connection connection = DriverManager.getConnection(url))
                {
                     final String schema = connection.getSchema();
                     System.out.println("Successful connection - Schema:" + schema);
                     System.out.println("query5: enter the new account");
                    try
                    (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                    {
```

```
                    stmt.setInt(1, account_no);
                    stmt.setString(2, date_ac_established);
                    rs = stmt.executeQuery();
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                           throw new RuntimeException(e);
                    }
               }


//Defining class for Procedure to enter new process account
             public static void new_process_account(int account_no, float
process_cost, int process_id)
             {
              String query = "{ call new_process_account(?, ?, ?) }";
              ResultSet rs;

              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    System.out.println("query5: enter the new process account, if
any");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, account_no);
                    stmt.setFloat(2, process_cost);
                    stmt.setInt(3, process_id);
                    rs = stmt.executeQuery();
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
```

```
                        throw new RuntimeException(e);
                    }
                }

//Defining class for Procedure to enter new assembly account
            public static void new_assembly_account(int account_no, float
assembly_cost, int assembly_id)
            {
              String query = "{ call new_assembly_account(?, ?, ?) }";
              ResultSet rs;

              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                   System.out.println("query5: enter the new assembly account, if
any");
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                          CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setInt(1, account_no);
                   stmt.setFloat(2, assembly_cost);
                   stmt.setInt(3, assembly_id);
                   rs = stmt.executeQuery();
                   }
                  catch (SQLException ex)
                  {
                   System.out.println(ex.getMessage());
                   }
                  }
              catch (SQLException e)
              {
                        throw new RuntimeException(e);
                   }
              }

//Defining class for Procedure to enter new department account
            public static void new_department_account(int account_no, float
department_cost, int dept_no)
            {
              String query = "{ call new_department_account(?, ?, ?) }";
              ResultSet rs;

              final String hostName = "userID-sql-server.database.windows.net";
```

```
            final String dbName = "my-sql-db";
            final String user = "userID";
            final String password = "Password";
            final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
            try
            (final Connection connection = DriverManager.getConnection(url))
            {
                 final String schema = connection.getSchema();
                 System.out.println("Successful connection - Schema:" + schema);
                 System.out.println("query5: enter the new department account, if
any");
                try
                (final Connection conn = DriverManager.getConnection(url);
                        CallableStatement stmt = conn.prepareCall(query))
                {

                 stmt.setInt(1, account_no);
                 stmt.setFloat(2, department_cost);
                 stmt.setInt(3, dept_no);
                 rs = stmt.executeQuery();
                 }
                catch (SQLException ex)
                {
                 System.out.println(ex.getMessage());
                 }
                }
            catch (SQLException e)
            {
                        throw new RuntimeException(e);
                 }
            }


//Defining class for Procedure to create a new job query 6
            public static void new_job(int new_job_no, String new_date_commenced,
int new_assembly_id, int new_process_id)
            {
             String query = "{ call new_job(?, ?,?,?) }"; //Calling procedure #1
             ResultSet rs;

      // Connect to database
            final String hostName = "userID-sql-server.database.windows.net";
            final String dbName = "my-sql-db";
            final String user = "userID";
            final String password = "Password";
            final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
            try
            (final Connection connection = DriverManager.getConnection(url))
            {
```

```
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    System.out.println("query6: enter the new job");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

      //Providing parameters to the procedure
                    stmt.setInt(1, new_job_no);
                    stmt.setString(2, new_date_commenced);
                    stmt.setInt(3, new_assembly_id);
                    stmt.setInt(4, new_process_id);
                    rs = stmt.executeQuery();                    //Query execution
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                           throw new RuntimeException(e);
                    }
               }

//Defining class for Procedure to create a new job query 7
             public static void completion_job(int job_no, String date_completed)
             {
               String query = "{ call completion_job(?, ?) }"; //Calling procedure #1
               ResultSet rs;

      // Connect to database
               final String hostName = "userID-sql-server.database.windows.net";
               final String dbName = "my-sql-db";
               final String user = "userID";
               final String password = "Password";
               final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
               try
               (final Connection connection = DriverManager.getConnection(url))
               {
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    System.out.println("query7: enter the new job");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, job_no);
                    stmt.setString(2, date_completed);
                    rs = stmt.executeQuery();
```

```
                    }
                catch (SQLException ex)
                {
                 System.out.println(ex.getMessage());
                 }
                }
            catch (SQLException e)
            {
                        throw new RuntimeException(e);
                 }
            }

//Defining class for Procedure to create a completed_cut_job query 7
             public static void completed_cut_job(int job_no, String
machine_type,String time_machine_used,String material,float labor_time_cut )
             {
              String query = "{ call completed_cut_job(?, ?,?,?,?) }"; //Calling
procedure #1
              ResultSet rs;

       // Connect to database
              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                   //System.out.println("query7: enter the new job");
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                         CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setInt(1, job_no);
                   stmt.setString(2, machine_type);
                   stmt.setString(3, time_machine_used);
                   stmt.setString(4, material);
                   stmt.setFloat(5, labor_time_cut);
                   rs = stmt.executeQuery();
                   }
                  catch (SQLException ex)
                  {
                   System.out.println(ex.getMessage());
                   }
                  }
              catch (SQLException e)
              {
                         throw new RuntimeException(e);
```

```
                    }
                }

//Defining class for Procedure to create a completed_paint_job query 7
             public static void completed_paint_job(int job_no, String color, float
volume,float labor_time_paint )
             {
               String query = "{ call completed_paint_job(?, ?,?,?) }"; //Calling
procedure #1
               ResultSet rs;

       // Connect to database
               final String hostName = "userID-sql-server.database.windows.net";
               final String dbName = "my-sql-db";
               final String user = "userID";
               final String password = "Password";
               final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
               try
               (final Connection connection = DriverManager.getConnection(url))
               {
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    //System.out.println("query7: enter the new job");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, job_no);
                    stmt.setString(2, color);
                    stmt.setFloat(3, volume);
                    stmt.setFloat(4, labor_time_paint);
                    rs = stmt.executeQuery();
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                           throw new RuntimeException(e);
                    }
               }

//Defining class for Procedure to create a completed_paint_job query 7
             public static void completed_fit_job(int job_no, float labor_time_fit )
             {
               String query = "{ call completed_fit_job(?, ?) }"; //Calling procedure
#1
               ResultSet rs;
```

```
	// Connect to database
		final String hostName = "userID-sql-server.database.windows.net";
		final String dbName = "my-sql-db";
		final String user = "userID";
		final String password = "Password";
		final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
		try
		(final Connection connection = DriverManager.getConnection(url))
		{
		     final String schema = connection.getSchema();
		     System.out.println("Successful connection - Schema:" + schema);
		     //System.out.println("query7: enter the new job");
		    try
		    (final Connection conn = DriverManager.getConnection(url);
		           CallableStatement stmt = conn.prepareCall(query))
		    {

		     stmt.setInt(1, job_no);
		     stmt.setFloat(2, labor_time_fit);
		     rs = stmt.executeQuery();
		     }
		    catch (SQLException ex)
		    {
		     System.out.println(ex.getMessage());
		     }
		    }
		catch (SQLException e)
		{
		           throw new RuntimeException(e);
		     }
		}

//Defining class for Procedure to entry cost transaction 8
		public static void cost_transaction_entry(int transaction_no, float
sup_cost, int transaction_job_no )
		{
		  String query = "{ call cost_transaction_entry(?, ?, ?) }"; //Calling
procedure #1
		  ResultSet rs;

	// Connect to database
		final String hostName = "userID-sql-server.database.windows.net";
		final String dbName = "my-sql-db";
		final String user = "userID";
		final String password = "Password";
		final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
		try
		(final Connection connection = DriverManager.getConnection(url))
```

```
                {
                     final String schema = connection.getSchema();
                     System.out.println("Successful connection - Schema:" + schema);
                     //System.out.println("query7: enter the new job");
                    try
                    (final Connection conn = DriverManager.getConnection(url);
                            CallableStatement stmt = conn.prepareCall(query))
                    {

                     stmt.setInt(1, transaction_no);
                     stmt.setFloat(2, sup_cost);
                     stmt.setInt(3, transaction_job_no);
                     rs = stmt.executeQuery();
                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
                catch (SQLException e)
                {
                            throw new RuntimeException(e);
                     }
                }

//Defining class for Procedure to update process account of given cost transaction 8
             public static void update_process_ac(int tran_no, float
process_sup_cost)
             {
              String query = "{ call update_process_ac(?, ?) }"; //Calling procedure
#1
              ResultSet rs;

       // Connect to database
              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                   //System.out.println("query7: enter the new job");
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                          CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setInt(1, tran_no);
                   stmt.setFloat(2, process_sup_cost);
```

```
                    rs = stmt.executeQuery();
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                           throw new RuntimeException(e);
                    }
               }

//Defining class for Procedure to update assembly account of given cost transaction 8
             public static void update_assembly_ac(int tran_no, float
assembly_sup_cost)
             {
              String query = "{ call update_assembly_ac(?, ?) }"; //Calling
procedure #1
               ResultSet rs;

      // Connect to database
               final String hostName = "userID-sql-server.database.windows.net";
               final String dbName = "my-sql-db";
               final String user = "userID";
               final String password = "Password";
               final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
               try
               (final Connection connection = DriverManager.getConnection(url))
               {
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    //System.out.println("query7: enter the new job");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, tran_no);
                    stmt.setFloat(2, assembly_sup_cost);
                    rs = stmt.executeQuery();
                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                           throw new RuntimeException(e);
                    }
               }
```

```
//Defining class for Procedure to update department account of given cost transaction
8
              public static void update_dept_ac(int tran_no, float dept_sup_cost)
              {
                String query = "{ call update_dept_ac(?, ?) }"; //Calling procedure #1
                ResultSet rs;

       // Connect to database
                final String hostName = "userID-sql-server.database.windows.net";
                final String dbName = "my-sql-db";
                final String user = "userID";
                final String password = "Password";
                final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
                try
                (final Connection connection = DriverManager.getConnection(url))
                {
                     final String schema = connection.getSchema();
                     System.out.println("Successful connection - Schema:" + schema);
                     //System.out.println("query7: enter the new job");
                    try
                    (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                    {

                     stmt.setInt(1, tran_no);
                     stmt.setFloat(2, dept_sup_cost);
                     rs = stmt.executeQuery();
                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
                catch (SQLException e)
                {
                           throw new RuntimeException(e);
                     }
                }

//Defining class for Procedure retrieve cost incurred on assembly id query 9
              public static void ret_assembly_cost(int ret_assembly_id)
              {
                String query = "{ call ret_assembly_cost(?) }"; //Calling procedure #1
                ResultSet rs;

       // Connect to database
                final String hostName = "userID-sql-server.database.windows.net";
                final String dbName = "my-sql-db";
                final String user = "userID";
                final String password = "Password";
```

```
            final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
            try
            (final Connection connection = DriverManager.getConnection(url))
            {
                 final String schema = connection.getSchema();
                 System.out.println("Successful connection - Schema:" + schema);
                try
                (final Connection conn = DriverManager.getConnection(url);
                        CallableStatement stmt = conn.prepareCall(query))
                {

                 stmt.setInt(1, ret_assembly_id);
                 rs = stmt.executeQuery();
                 while (rs.next()) {              //Displaying table from SQL
database
                        System.out.println(String.format("%f",
                                    rs.getFloat(1)));
                    }

                 }
                catch (SQLException ex)
                {
                 System.out.println(ex.getMessage());
                 }
                }
            catch (SQLException e)
            {
                        throw new RuntimeException(e);
                 }
            }

//Defining class for Procedure retrieve total labor time query 10
            public static void tot_labor_time(int dept_num, String start_compl_date,
String end_compl_date)
            {
             String query = "{ call tot_labor_time(?,?,?) }";
             ResultSet rs;

             final String hostName = "userID-sql-server.database.windows.net";
             final String dbName = "my-sql-db";
             final String user = "userID";
             final String password = "Password";
             final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
             try
             (final Connection connection = DriverManager.getConnection(url))
             {
                 final String schema = connection.getSchema();
                 System.out.println("Successful connection - Schema:" + schema);
                try
```

```
                    (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                    {

                     stmt.setInt(1, dept_num);
                     stmt.setString(2, start_compl_date);
                     stmt.setString(3, end_compl_date);
                     rs = stmt.executeQuery();
                     while (rs.next()) {              //Displaying table from SQL
database
                            System.out.println(String.format("%f",
                                         rs.getFloat(1)));
                        }

                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
                catch (SQLException e)
                {
                           throw new RuntimeException(e);
                     }
                }

//Defining class for Procedure retrieve process and department for query 11
              public static void ret_process(int assembly_id_process)
              {
                String query = "{ call ret_process(?) }";
                ResultSet rs;

                final String hostName = "userID-sql-server.database.windows.net";
                final String dbName = "my-sql-db";
                final String user = "userID";
                final String password = "Password";
                final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
                try
                (final Connection connection = DriverManager.getConnection(url))
                {
                     final String schema = connection.getSchema();
                     System.out.println("Successful connection - Schema:" + schema);
                    try
                    (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                    {

                     stmt.setInt(1, assembly_id_process);
                     rs = stmt.executeQuery();
                     while (rs.next()) {              //Displaying table from SQL
database
                            System.out.println(String.format("%d",
```

```
                                        rs.getInt(1)));
                        }
                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
                catch (SQLException e)
                {
                        throw new RuntimeException(e);
                     }
                }

//Defining class for Procedure to retrieve jobs as per query 12
             public static void ret_jobs(String start_comp_date, String
end_comp_date, int deptartment_no)
             {
              String query = "{ call ret_jobs(?, ?, ?) }";
              ResultSet rs;

              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                         CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setString(1, start_comp_date);
                   stmt.setString(2, end_comp_date);
                   stmt.setInt(3, deptartment_no);
                   rs = stmt.executeQuery();
                   while (rs.next()) {              //Displaying table from SQL
database
                         System.out.println(String.format("%d",
                                        rs.getInt(1)));
                        }
                     }
                    catch (SQLException ex)
                    {
                     System.out.println(ex.getMessage());
                     }
                    }
              catch (SQLException e)
```

```
                {
                        throw new RuntimeException(e);
                    }
                }


//Defining class for Procedure retrieve customer as per query 13
             public static void ret_customer(int start_cat, int end_cat)
             {
               String query = "{ call ret_customer(?, ?) }"; //Calling procedure #1
               ResultSet rs;

       // Connect to database
               final String hostName = "userID-sql-server.database.windows.net";
               final String dbName = "my-sql-db";
               final String user = "userID";
               final String password = "Password";
               final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
               try
               (final Connection connection = DriverManager.getConnection(url))
               {
                    final String schema = connection.getSchema();
                    System.out.println("Successful connection - Schema:" + schema);
                    System.out.println("query13: enter the start and end category to
retrieve customer");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, start_cat);
                    stmt.setInt(2, end_cat);
                    rs = stmt.executeQuery();

             while (rs.next()) {              //Displaying table from SQL database
                    System.out.println(String.format("%s",
                                rs.getString(1)));
                  }

                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
               catch (SQLException e)
               {
                        throw new RuntimeException(e);
                    }
               }

//Defining class for Procedure to delete cut jobs as per query 14
```

```
            public static void delete_cut_jobs(int start_job_no, int end_job_no)
            {
              String query = "{ call delete_cut_jobs(?, ?) }"; //Calling procedure
#1
              ResultSet rs;

      // Connect to database
              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                   System.out.println("query14: enter the start and end job no to
delete cut jobs");
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                         CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setInt(1, start_job_no);
                   stmt.setInt(2, end_job_no);
                   rs = stmt.executeQuery();

                   }
                  catch (SQLException ex)
                  {
                   System.out.println(ex.getMessage());
                   }
                  }
              catch (SQLException e)
              {
                         throw new RuntimeException(e);
                   }
              }

//Defining class for Procedure to change color as per query 15
            public static void change_color(String color, int col_job_no)
            {
              String query = "{ call change_color(?, ?) }";
              ResultSet rs;


              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
```

```
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
                   System.out.println("query15: enter the color and end job no to
change color");
                  try
                  (final Connection conn = DriverManager.getConnection(url);
                         CallableStatement stmt = conn.prepareCall(query))
                  {

                   stmt.setString(1, color);
                   stmt.setInt(2, col_job_no);
                   rs = stmt.executeQuery();

                   }
                  catch (SQLException ex)
                  {
                   System.out.println(ex.getMessage());
                   }
                  }
              catch (SQLException e)
              {
                         throw new RuntimeException(e);
                   }
              }


//Defining class for Procedure retrieve customer and export to text file as per query
17
            public static void export_customer(int start_cat, int end_cat) throws
IOException
            {
              String query = "{ call ret_customer(?, ?) }"; //Calling procedure #1
              ResultSet rs;

      // Connect to database
              final String hostName = "userID-sql-server.database.windows.net";
              final String dbName = "my-sql-db";
              final String user = "userID";
              final String password = "Password";
              final String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
host NameInCertificate=*.database.windows.net;loginTimeout=30;", hostName, dbName,
user, password);
              try
              (final Connection connection = DriverManager.getConnection(url))
              {
                   final String schema = connection.getSchema();
                   System.out.println("Successful connection - Schema:" + schema);
```

```
                    System.out.println("query13: enter the start and end category to
retrieve customer");
                   try
                   (final Connection conn = DriverManager.getConnection(url);
                           CallableStatement stmt = conn.prepareCall(query))
                   {

                    stmt.setInt(1, start_cat);
                    stmt.setInt(2, end_cat);
                    rs = stmt.executeQuery();

                    FileWriter fstream = new
FileWriter("C:\\Users\\abhig\\Desktop\\DBMS\\Individual project\\exportFile.txt");
                    BufferedWriter out = new BufferedWriter(fstream);
                    while (rs.next()) {
                            out.write((rs.getString(1)) + ", ");
                            out.newLine();
                    }
                    System.out.println("Completed writing into text file");
                    out.close();

                    }
                   catch (SQLException ex)
                   {
                    System.out.println(ex.getMessage());
                    }
                   }
                catch (SQLException e)
                {
                           throw new RuntimeException(e);
                    }
                }

       public static void main(String[] args) throws IOException
       {
              String nl = System.getProperty("line.separator");    //Defining line
separator
              System.out.println("1. Insert new customer by selecting option 1" + nl +
                           "2. Insert new department by selecting option 2" + nl +
                           "3. Insert new assembly by selecting option 3" + nl +
                           "4. Insert new process details by selecting option 4" + nl
+
                           "5. Insert new account details by selecting option 5" + nl
+
                           "6. Insert new job option 6" + nl +
                           "7. Insert the job completion details option 7" + nl +
                           "8. Update all the sup cost option 8" + nl +
                           "9. Retrieve cost incurred in assembly ID option 9" + nl +
                           "10.Retrieve total labor time option 10" + nl +
                           "11.Retrieve process through which assembly id has passed
option 11" + nl +
                           "12.Retrieve job no between completion date option 12" +
nl +
                           "13.Retrieve customer name between category option 13" +
nl +
```

```
                        "14.Delete cut jobs between jo no. option 14" + nl +
                        "15.Update color for given jo no. option 15" + nl +
                        "17.Export the customer name by option 17" + nl
                        + "18. Exit");

            while (true)
            {
                  System.out.println("Please Enter Option Number: ");//Obtaining
user input for option
                  Scanner user_input = new Scanner(System.in);
                  int option = user_input.nextInt();

                  if (option == 1)
                  {
                        System.out.println("Enter New Customer's name:
");//Performer ID input by user
                        String c_name = user_input.next();

                        System.out.println("Enter New customer's address:
");//Performer age input by user
                        String c_address = user_input.next();

                        System.out.println("Enter New customer's category:
");//Performer name input by user
                        int category = user_input.nextInt();

                        new_customer(c_name, c_address, category);//Providing
input to procedure 1
                  }

                  else if (option == 2)
                  {
                        System.out.println("Enter New Department's number: ");
                        int dept_no = user_input.nextInt();

                        System.out.println("Enter New department's data: ");
                        String dept_data = user_input.next();
                        new_department(dept_no, dept_data);
                  }

                  else if (option == 3)
                  {
                        System.out.println("Enter New assembly ID: ");
                        int assembly_id = user_input.nextInt();

                        System.out.println("Enter date ordered for assembly: ");
                        String date_ordered = user_input.next();

                        System.out.println("Enter assembly details: ");
                        String assembly_details = user_input.next();

                        System.out.println("Enter customer name who ordered
assembly : ");
                        String name = user_input.next();
```

```
                        new_assembly(assembly_id, date_ordered, assembly_details,
name);
                }

                else if (option == 4)
                {
                        System.out.println("Enter new process id: ");
                        int process_id = user_input.nextInt();

                        System.out.println("Enter the process data: ");
                        String process_data = user_input.next();

                        System.out.println("Enter the departemnt no for this
process: ");
                        int dept_no = user_input.nextInt();

                        new_process(process_id, process_data, dept_no);//Providing
input to procedure 1
                }

                else if (option == 5)
                {
                        System.out.println("Enter the account no: ");
                        int account_no = user_input.nextInt();

                        System.out.println("Enter the date established: ");
                        String date_ac_established = user_input.next();

                        new_account(account_no, date_ac_established);

                        System.out.println("Enter the type account related to
(process1, assembly2 or dept3: 1, 2, 3? ");
                        int ac_type = user_input.nextInt();

                        if (ac_type == 1)
                        {
                                System.out.println("Enter the process cost:
");
                                float process_cost = user_input.nextFloat();

                                System.out.println("Enter the process id: ");
                                int process_id = user_input.nextInt();

                                new_process_account(account_no, process_cost,
process_id);
                        }

                        if (ac_type == 2)
                        {
                                System.out.println("Enter the assembly cost:
");
                                float assembly_cost = user_input.nextFloat();

                                System.out.println("Enter the assembly id:
");
```

```
                                        int assembly_id = user_input.nextInt();

                                        new_assembly_account(account_no,
assembly_cost, assembly_id);
                                }

                                if (ac_type == 3)
                                {
                                        System.out.println("Enter the department
cost: ");
                                        float department_cost =
user_input.nextFloat();

                                        System.out.println("Enter the department
number: ");
                                        int dept_no = user_input.nextInt();

                                        new_department_account(account_no,
department_cost, dept_no);
                                }


                        }

                        else if (option == 6)
                        {
                                System.out.println("Enter New job number: ");
                                int new_job_no = user_input.nextInt();

                                System.out.println("Enter date commenced: ");
                                String new_date_commenced = user_input.next();

                                System.out.println("Enter assembly id: ");
                                int new_assembly_id = user_input.nextInt();

                                System.out.println("Enter process id: ");
                                int new_process_id = user_input.nextInt();

                                new_job(new_job_no, new_date_commenced, new_assembly_id,
new_process_id);
                        }

                        else if (option == 7)
                        {
                                System.out.println("Enter the job no: ");
                                int job_no = user_input.nextInt();

                                System.out.println("Enter the date completed: ");
                                String date_completed = user_input.next();

                                completion_job(job_no, date_completed);

                                System.out.println("Enter the type of job (cut1, paint2 or
fit3: 1, 2, 3? ");
                                int job_type = user_input.nextInt();
```

```
                        if (job_type == 1)
                        {
                                System.out.println("Enter the machine type:
");
                                String machine_type = user_input.next();
                                System.out.println("Enter the time machine
used: ");
                                String time_machine_used = user_input.next();
                                System.out.println("Enter the material used:
");
                                String material = user_input.next();
                                System.out.println("Enter the labor time cut:
");
                                float labor_time_cut =
user_input.nextFloat();

                                completed_cut_job(job_no, machine_type,
time_machine_used, material, labor_time_cut);
                        }

                        if (job_type == 2)
                        {
                                System.out.println("Enter the color: ");
                                String color = user_input.next();
                                System.out.println("Enter the volume used:
");
                                float volume = user_input.nextFloat();
                                System.out.println("Enter the labor time
paint: ");
                                float labor_time_paint =
user_input.nextFloat();

                                completed_paint_job(job_no, color, volume,
labor_time_paint);
                        }

                        if (job_type == 3)
                        {
                                System.out.println("Enter the labor time fit:
");
                                float labor_time_fit =
user_input.nextFloat();

                                completed_fit_job(job_no, labor_time_fit);
                        }


                }

                else if (option == 8)
                {
                        System.out.println("Enter the transaction no: ");
                        int transaction_no = user_input.nextInt();
```

```
                        System.out.println("Enter the sup cost: ");
                        Float sup_cost = user_input.nextFloat();

                        System.out.println("Enter the transaction job no: ");
                        int transaction_job_no = user_input.nextInt();

                        System.out.println("Enter the process sup cost: ");
                        Float process_sup_cost = user_input.nextFloat();

                        System.out.println("Enter the assembly sup cost: ");
                        Float assembly_sup_cost = user_input.nextFloat();

                        System.out.println("Enter the department sup cost: ");
                        Float dept_sup_cost = user_input.nextFloat();

                        cost_transaction_entry(transaction_no, sup_cost,
transaction_job_no);
                        update_assembly_ac(transaction_no, assembly_sup_cost);
                        update_process_ac(transaction_no, process_sup_cost);
                        update_dept_ac(transaction_no, dept_sup_cost);
                }

                else if (option == 9)
                {
                        System.out.println("Enter assembly id: ");
                        int ret_assembly_id = user_input.nextInt();

                        ret_assembly_cost(ret_assembly_id);
                }

                else if (option == 10)
                {
                        System.out.println("Enter dept_no: ");
                        int dept_num = user_input.nextInt();

                        System.out.println("Enter start of job completion date:
");
                        String start_compl_date = user_input.next();

                        System.out.println("Enter end of job completion date: ");
                        String end_compl_date = user_input.next();

                        tot_labor_time(dept_num, start_compl_date,
end_compl_date);
                }

                else if (option == 11)
                {
                        System.out.println("Enter assembly id: ");
                        int assembly_id_process = user_input.nextInt();
                        ret_process(assembly_id_process);
                }

                else if (option == 12)
                {
```

```
                        System.out.println("Enter start of job completion date:
");
                        String start_comp_date = user_input.next();

                        System.out.println("Enter end of job completion date: ");
                        String end_comp_date = user_input.next();

                        System.out.println("Enter dept_no: ");
                        int department_no = user_input.nextInt();

                        ret_jobs(start_comp_date, end_comp_date, department_no);
                }

                else if (option == 13)
                {
                        System.out.println("Enter start category: ");
                        int start_cat = user_input.nextInt();

                        System.out.println("Enter end category: ");
                        int end_cat = user_input.nextInt();

                        ret_customer(start_cat, end_cat);
                }

                else if (option == 14)
                {
                        System.out.println("Enter start job no.: ");
                        int start_job_no = user_input.nextInt();

                        System.out.println("Enter end job no.: ");
                        int end_job_no = user_input.nextInt();

                        delete_cut_jobs(start_job_no, end_job_no);
                }

                else if (option == 15)
                {
                        System.out.println("Enter new color: ");
                        String color = user_input.next();

                        System.out.println("Enter job no.: ");
                        int col_job_no = user_input.nextInt();

                        change_color(color, col_job_no);
                }

                else if (option == 17)
                {
                        System.out.println("Enter start category: ");
                        int start_cat = user_input.nextInt();

                        System.out.println("Enter end category: ");
                        int end_cat = user_input.nextInt();

                        export_customer(start_cat, end_cat);
```

```
                }

                else if (option == 18)
                {
                        System.out.println("Quit option successfully exeuted");
                        System.exit(0);//Breaking while loop
                }
        }

    }
}
```

Chapter 6 java programm execution

6.1: Screenshot showing testing of query 1

c_name	c_address	category
Abhishek	Sarkeys	1
Bikash	Stephenson st	5
Michael	Commons apt	5
Rahul	Boyd st	2
Sachin	Stephenson st	3

6.2: Screenshot showing testing of query 2

dept_no	dept_data
1	clerk
2	manager
3	staff
4	clerk
5	manager

6.3: Screenshot showing testing of query 3

assembly_id	date_ordered	assembly_details	c_name
1	2019-01-01	square	Abhishek
2	2019-01-02	rectangle	Bikash
3	2019-01-03	circle	Michael
4	2019-01-04	sqaure	Rahul
5	2019-01-05	rectangle	sachin
6	2019-01-06	cube	Abhishek
7	2019-01-07	cuboid	bikash
8	2019-01-08	sphere	Michael
9	2019-01-09	Radial	Rahul
10	2019-01-10	rectangle	sachin

6.4: Screenshot showing testing of query 4

process_id	process_data	dept_no
1	cut	1
2	paint	2
3	fit	3
4	cut	4
5	paint	5
6	fit	1
7	cut	2
8	paint	3
9	fit	4
10	cut	5

6.5: Screenshot showing testing of query 5

account_no	date_ac_established
1	2018-01-01
2	2018-01-02
3	2018-01-03
4	2018-01-04
5	2018-01-05
6	2018-01-06
7	2018-01-07
8	2018-01-08
9	2018-01-09
10	2018-01-10

account_no	assembly_cost	assembly_id
2	100.00	1
5	200.00	2
8	300.00	3

account_no	department_cost	dept_no
3	100.00	1
6	200.00	2
9	300.00	3

account_no	process_cost	process_id
1	100.00	1
4	200.00	2
7	300.00	3
10	400.00	4

6.6: Screenshot showing testing of query 6

assembly_id	process_id	job_no
1	1	1
1	1	2
1	1	3
2	2	4
2	2	5
2	2	6
3	3	7
3	3	8
3	3	9
4	4	10

job_no	date_commenced	date_completed
1	2019-02-01	*NULL*
2	2019-02-02	*NULL*
3	2019-02-03	*NULL*
4	2019-02-04	*NULL*
5	2019-02-05	*NULL*
6	2019-02-06	*NULL*
7	2019-02-07	*NULL*
8	2019-02-08	*NULL*
9	2019-02-09	*NULL*
10	2019-02-10	*NULL*

6.7: Screenshot showing testing of query 7

job_no	date_commenced	date_completed
1	2019-02-01	2019-03-01
2	2019-02-02	2019-03-02
3	2019-02-03	2019-03-03
4	2019-02-04	2019-03-04
5	2019-02-05	2019-03-05
6	2019-02-06	2019-03-06
7	2019-02-07	2019-03-07
8	2019-02-08	2019-03-08
9	2019-02-09	2019-03-09
10	2019-02-10	2019-03-10

job_no	machine_type	time_machine_used	material	labor_time_cut
1	saw	1	aluminum	1.00
4	drill	4	iron	4.00
7	polish	7	copper	7.00

job_no	color	volume	labor_time_paint
2	blue	2.00	2.00
5	yellow	5.00	5.00
8	purple	8.00	8.00

job_no	labor_time_fit
3	3.00
6	6.00
9	9.00
10	10.00

6.8: Screenshot showing testing of query 8

transaction_no	sup_cost	job_no
1	100.00	1
2	200.00	2
3	300.00	3
4	400.00	4
5	500.00	5
6	600.00	6
7	700.00	7
8	800.00	8
9	900.00	9
10	600.00	10

account_no	assembly_cost	assembly_id
2	290.00	1
5	730.00	2
8	800.00	3

account_no	department_cost	dept_no
3	320.00	1
6	640.00	2
9	700.00	3

account_no	process_cost	process_id
1	290.00	1
4	730.00	2
7	900.00	3
10	600.00	4

6.9: Screenshot showing testing of query 9

```
Please Enter Option Number:
9
Enter assembly id:
1
Successful connection - Schema:dbo
290.000000
Please Enter Option Number:
9
Enter assembly id:
2
Successful connection - Schema:dbo
730.000000
Please Enter Option Number:
9
Enter assembly id:
3
Successful connection - Schema:dbo
800.000000
Please Enter Option Number:
```

6.10: Screenshot showing testing of query 10

```
Please Enter Option Number:
10
Enter dept_no:
1
Enter start of job completion date:
02-02-2019
Enter end of job completion date:
04-01-2019
Successful connection - Schema:dbo
6.000000
Please Enter Option Number:
10
Enter dept_no:
2
Enter start of job completion date:
02-02-2019
Enter end of job completion date:
04-01-2019
Successful connection - Schema:dbo
15.000000
Please Enter Option Number:
10
Enter dept_no:
3
Enter start of job completion date:
02-02-2019
Enter end of job completion date:
04-01-2019
Successful connection - Schema:dbo
24.000000
Please Enter Option Number:
```

6.11: Screenshot showing testing of query 11

```
Please Enter Option Number:
11
Enter assembly id:
1
Successful connection - Schema:dbo
1
Please Enter Option Number:
11
Enter assembly id:
2
Successful connection - Schema:dbo
2
Please Enter Option Number:
11
Enter assembly id:
3
Successful connection - Schema:dbo
3
Please Enter Option Number:
```

6.12: Screenshot showing testing of query 12

```
Please Enter Option Number:
12
Enter start of job completion date:
02-01-2019
Enter end of job completion date:
04-01-2019
Enter dept_no:
1
Successful connection - Schema:dbo
1
2
3
Please Enter Option Number:
12
Enter start of job completion date:
02-01-2019
Enter end of job completion date:
04-01-2019
Enter dept_no:
2
Successful connection - Schema:dbo
4
5
6
Please Enter Option Number:

12
Enter start of job completion date:
02-01-2019
Enter end of job completion date:
04-01-2019
Enter dept_no:
3
Successful connection - Schema:dbo
7
8
9
Please Enter Option Number:
```

6.13: Screenshot showing testing of query 13

```
Please Enter Option Number:
13
Enter start category:
1
Enter end category:
4
Successful connection - Schema:dbo
query13: enter the start and end category to retrieve customer
Abhishek
Rahul
Sachin
Please Enter Option Number:
13
Enter start category:
1
Enter end category:
5
Successful connection - Schema:dbo
query13: enter the start and end category to retrieve customer
Abhishek
Bikash
Michael
Rahul
Sachin
Please Enter Option Number:

13
Enter start category:
5
Enter end category:
9
Successful connection - Schema:dbo
query13: enter the start and end category to retrieve customer
Bikash
Michael
Please Enter Option Number:
```

6.14: Screenshot showing testing of query 14

```
Please Enter Option Number:
14
Enter start job no.:
7
Enter end job no.:
8
Successful connection - Schema:dbo
query14: enter the start and end job no to delete cut jobs
The statement did not return a result set.
Please Enter Option Number:
14
Enter start job no.:
8
Enter end job no.:
9
Successful connection - Schema:dbo
query14: enter the start and end job no to delete cut jobs
The statement did not return a result set.
Please Enter Option Number:
14
Enter start job no.:
9
Enter end job no.:
10
Successful connection - Schema:dbo
query14: enter the start and end job no to delete cut jobs
The statement did not return a result set.
Please Enter Option Number:
```

6.15: Screenshot showing testing of query 15

Before:

job_no	color	volume	labor_time_paint
2	white	2.00	2.00
5	yellow	5.00	5.00
8	purple	8.00	8.00

```
Please Enter Option Number:
15
Enter new color:
red
Enter job no.:
2
Successful connection - Schema:dbo
query15: enter the color and end job no to change color
The statement did not return a result set.
Please Enter Option Number:
15
Enter new color:
blue
Enter job no.:
5
Successful connection - Schema:dbo
query15: enter the color and end job no to change color
The statement did not return a result set.
Please Enter Option Number:
15
Enter new color:
brown
Enter job no.:
8
Successful connection - Schema:dbo
query15: enter the color and end job no to change color
The statement did not return a result set.
Please Enter Option Number:
```

After program execution:

job_no	color	volume	labor_time_paint
2	red	2.00	2.00
5	blue	5.00	5.00
8	brown	8.00	8.00

6.16: Screenshot showing testing of query 16

6.17: Screenshot showing testing of query 17

```
Please Enter Option Number:
17
Enter start category:
1
Enter end category:
8
Successful connection - Schema:dbo
query13: enter the start and end category to retrieve customer
Completed writing into text file
Please Enter Option Number:
```

The output from exported text file is:

Abhishek,

Bikash,

Michael,

Rahul,

Sachin,

6.18: Screenshot showing testing of query 18

```
1. Insert new customer by selecting option 1
2. Insert new department by selecting option 2
3. Insert new assembly by selecting option 3
4. Insert new process details by selecting option 4
5. Insert new account details by selecting option 5
6. Insert new job option 6
7. Insert the job completion details option 7
8. Update all the sup cost option 8
9. Retrieve cost incurred in assembly ID option 9
10.Retrieve total labor time option 10
11.Retrieve process through which assembly id has passed option 11
12.Retrieve job no between completion date option 12
13.Retrieve customer name between category option 13
14.Delete cut jobs between jo no. option 14
15.Update color for given jo no. option 15
17.Export the customer name by option 17
18. Exit
Please Enter Option Number:
18
Quit option successfully exeuted
```

6.19 Screenshot showing three types of errors:

1.) violation of primary key

```
Please Enter Option Number:
1
Enter New Customer's name:
Abhishek
Enter New customer's address:
boyd
Enter New customer's category:
2
Successful connection - Schema:dbo
query1: enter the new customer
Violation of PRIMARY KEY constraint 'PK__Customer__29397C80304B4269'. Cannot insert
duplicate key in object 'dbo.Customer'. The duplicate key value is (Abhishek).
Please Enter Option Number:
```

2) violation of data type

```
Please Enter Option Number:
2
Enter New Department's number:
abc
```

```
Exception in thread "main" java.util.InputMismatchException
	at java.base/java.util.Scanner.throwFor(Scanner.java:939)
	at java.base/java.util.Scanner.next(Scanner.java:1594)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
	at IP_Chapter5_Gupta_Abhishek.main(IP_Chapter5_Gupta_Abhishek.java:1037)
```

3) Error when no foreign key exist

```
Please Enter Option Number:
4
Enter new process id:
13
Enter the process data:
paint
Enter the departemnt no for this process:
17
Successful connection - Schema:dbo
query4: enter the new process
The INSERT statement conflicted with the FOREIGN KEY constraint
"FK__Processes__dept___4C564A9F". The conflict occurred in database "my-sql-db",
table "dbo.Department", column 'dept_no'.
Please Enter Option Number:
```

Chapter 7. Web database application and its execution

```
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 DataHandler  {

private Connection conn;

//Azure SQL connection credentials
	private String server = "userID-sql-server.database.windows.net";
	private String database = "my-sql-db";
	private String username = "userID";
	private String password = "Password";

	// Resulting connection string
	final private String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;
trustServerCertificate=false;hostNameInCertificate=*.database.windows.net;loginTimeou
t=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 movie_night table
public ResultSet getAllcustomers()
		throws SQLException
{
	getDBConnection();
	final String sqlQuery = "SELECT * FROM Customer;";
	final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
	return stmt.executeQuery();
	}
	// Inserts a record into the movie_night table with the given attribute values
public boolean add_new_customer(String name, String add, int categ)
		throws SQLException
		{
		getDBConnection();

		// Prepare the database connection


		// Prepare the SQL statement
				final String sqlQuery =
						"INSERT INTO Customer "
						+ "(name, add, categ) "
						+ "VALUES "
						+ "(?, ?, ?)";
				final PreparedStatement stmt =
conn.prepareStatement(sqlQuery);
		// Replace the '?' in the above statement with the given attribute
values
		stmt.setString(1, name);
		stmt.setString(2, add);
		stmt.setInt(3, categ);

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

	public ResultSet ret_customer_java(int categ1, int categ2) throws
SQLException//Defining class for Procedure 1
	{

	  getDBConnection();

		// Prepare the database connection
```

```
            // Prepare the SQL statement
                        final String sqlQuery = "{ call ret_customer(?, ?) }";

                        final PreparedStatement stmt =
conn.prepareStatement(sqlQuery);
         // Replace the '?' in the above statement with the given attribute values
                    stmt.setInt(1, categ1);
                    stmt.setInt(2, categ2);


                return stmt.executeQuery();


            }
    }
```

####Code for jsp files:Add customer

```
<%@ 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.DataHandler"%>
  <%@page import="java.sql.ResultSet"%>
  <%@page import="java.sql.Array"%>
  <%
  // The handler is the one in charge of establishing the connection.
  DataHandler handler = new DataHandler();
```

```
// Get the attribute values passed from the input form.
String startTime = request.getParameter("start_time");
String movieName = request.getParameter("movie_name");
String durationString = request.getParameter("duration_min");

/*
 * If the user hasn't filled out all the time, movie name and duration. This is very simple checking.
 */
if (startTime.equals("") || movieName.equals("") || durationString.equals("")) {
   response.sendRedirect("IP_Chapter7_Gupta_Abhishek_add_customer_form.jsp");
} else {
   int duration = Integer.parseInt(durationString);

   // Now perform the query with the data from the form.
   boolean success = handler.add_new_customer(startTime, movieName, duration);
   if (!success) { // Something went wrong
      %>
         <h2>There was a problem inserting the course</h2>
      <%
   } else { // Confirm success to the user
      %>
      <h2>The customer:</h2>

      <ul>
         <li>Name: <%= startTime %></li>
         <li>Address: <%= movieName %></li>
         <li>Category: <%= durationString %></li>

      </ul>
```

```
        <h2>Was successfully inserted.</h2>

        <a href="IP_Chapter7_Gupta_Abhishek_add_customer_form.jsp">See all movie nights.</a>
        <%
      }
   }
   %>
   </body>
</html>
```

####Code for jsp files: customer form

```
<!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 movie_night record.
         Upon form submission, add_movie.jsp file will be invoked.
      -->
      <form action="IP_Chapter7_Gupta_Abhishek_add_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=start_time>
    </div></td>
  </tr>
  <tr>
    <td>Address:</td>
    <td><div style="text-align: center;">
    <input type=text name=movie_name>
    </div></td>
  </tr>
  <tr>
    <td>Category:</td>
    <td><div style="text-align: center;">
    <input type=text name=duration_min>
    </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>
```

####Code for jsp files: get all customer

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
  <meta charset="UTF-8">
    <title>Movie Nights</title>
  </head>
  <body>
    <%@page import="jsp_azure_test.DataHandler"%>
    <%@page import="java.sql.ResultSet"%>
    <%
      // We instantiate the data handler here, and get all the movies from the database
      final DataHandler handler = new DataHandler();
      final ResultSet movies = 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>Name</h4>
```

```
          </td>
          <td align="center">
            <h4>Address</h4>
          </td>
          <td align="center">
            <h4>Category</h4>
          </td>
        </tr>

        <%
           while(movies.next()) { // For each movie_night record returned...
              // Extract the attribute values for every row returned
              final String time = movies.getString("c_name");
              final String name = movies.getString("c_address");
              final String duration = movies.getString("category");

              out.println("<tr>"); // Start printing out the new table row
              out.println( // Print each attribute value
                 "<td align=\"center\">" + time +
                 "</td><td align=\"center\"> " + name +
                 "</td><td align=\"center\"> " + duration + "</td>");
              out.println("</tr>");
           }
        %>
      </table>
   </body>
</html>
```

####Code for jsp files: retrieve customer

```
<%@ 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.DataHandler"%>
   <%@page import="java.sql.ResultSet"%>
   <%@page import="java.sql.Array"%>
   <%
   // The handler is the one in charge of establishing the connection.
   DataHandler handler = new DataHandler();

   // Get the attribute values passed from the input form.
   String startTimestring = request.getParameter("start_time");
   String movieNamestring = request.getParameter("movie_name");

   /*
    * If the user hasn't filled out all the time, movie name and duration. This is very simple checking.
    */
   if (startTimestring.equals("") || movieNamestring.equals("")) {
      response.sendRedirect("IP_Chapter7_Gupta_Abhishek_ret_cust_jsp");
   } else {
      int startTime = Integer.parseInt(startTimestring);
```

```
    int movieName = Integer.parseInt(movieNamestring);

    // Now perform the query with the data from the form.
    ResultSet rs = handler.ret_customer_java(startTime, movieName);
    %>
    <table cellspacing="2" cellpadding="2" border="1">
      <tr> <!-- The table headers row -->
       <td align="center">
        <h4>Name</h4>
       </td>

      </tr>

      <%

      while(rs.next())
      { // For each movie_night record returned...
            // Extract the attribute values for every row returned
            final String time = rs.getString("c_name");

            out.println("<tr>"); // Start printing out the new table row
            out.println( // Print each attribute value
                "<td align=\"center\">" + time + "</td>");
            out.println("</tr>");
        }
      %>
     </table>
  </body>
</html>
```

```
        <h2>Query 13 Result</h2>




        <h2>Was successfully displayed.</h2>
        <%
      }
   %>
   </body>
</html>
```

####Code for jsp files: retrieve customer form

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>Range search of customer</title>
   </head>
   <body>
      <h2>Range search of customer</h2>
      <!--
         Form for collecting user input for the new movie_night record.
         Upon form submission, add_movie.jsp file will be invoked.
      -->
```

```
<form action="IP_Chapter7_Gupta_Abhishek_ret_cust.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>Category range start:</td>
      <td><div style="text-align: center;">
      <input type=text name=start_time>
      </div></td>
    </tr>
    <tr>
      <td>Category range termination:</td>
      <td><div style="text-align: center;">
      <input type=text name=movie_name>
      </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>
```

7.1. Web database application source program and screenshots showing Its successful compilation

All the above program was successfully executed and below are the snapshot

7.2. Screenshots showing the testing of the Web database application

Query 13 to retrieve customer

Range search of customer

Enter the Customer Data:	
Category range start:	3
Category range termination:	7
Clear	Insert

Result of the query

Name
Bikash
Michael
Sachin

Query 13 Result

Was successfully displayed.

For query 1: inseritng customer

Add Customer

Enter the Customer Data:	
Customer Name:	Samuel
Address:	Dallas
Category:	7
Clear	Insert

The customer:

- Name: Samuel
- Address: Dallas
- Category: 7

Was successfully inserted.

See all movie nights.

Running query 13 again to retrieve inserted customer

Range search of customer

Enter the Customer Data:	
Category range start:	3
Category range termination:	7
Clear	Insert

Name
Bikash
Michael
Sachin
samuel

Query 13 Result

Was successfully displayed.