

**Course name:** Database Management System

**Course number:** DSA 4513

**Section number:** 001

**Semester and year:** Fall 2023

**Instructor's name:** Dr. Le Gruenwald

**Author's name:** Leama Nolvenne Tah

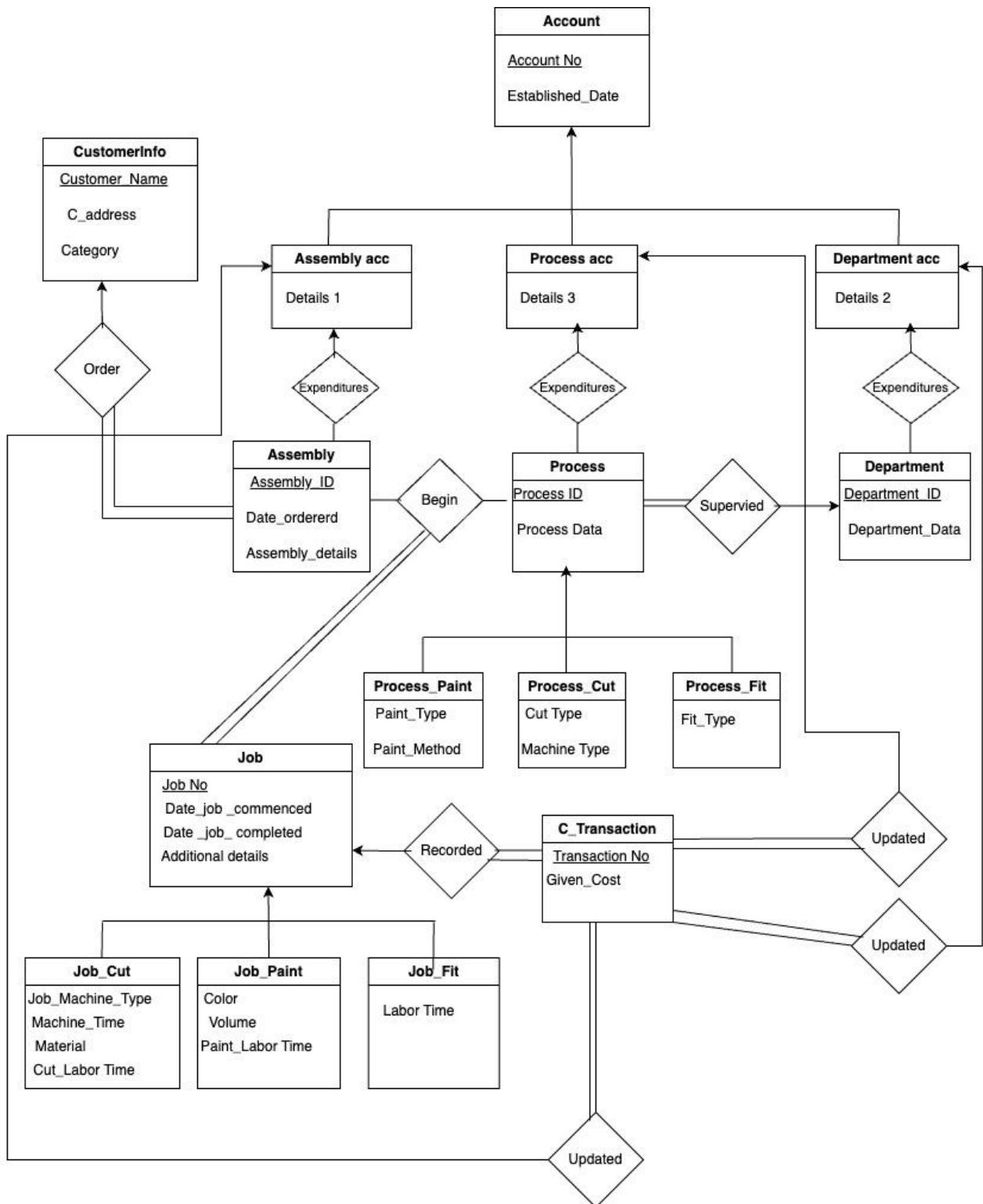
**ID:** 113634969

**Email address:** nolvenne.leama.tah-1@ou.edu

**Title of the project:** JOB-SHOP ACCOUNTING DATABASE SYSTEM

<b>Tasks Performed</b>	<b>Page Number</b>
Task 1. ER Diagram	3
Task 2. Relational Database Schemas	4
Task 3.	5-8
3.1. Discussion of storage structures for tables	5-7
3.2. Discussion of storage structures for tables (Azure SQL Database)	8
Task 4. SQL statements for the creation of tables in Azure SQL Database	8-13
Task 5.	14-61
5.1 SQL statements ( <b>and Transact SQL stored procedures, if any</b> )	14-22
Implementing all queries (1-15 and error checking)	
5.2 The Java source program and screenshots showing its successful compilation	23-61
Task 6. Java program Execution	61-90
Task 7. Web database application and its execution	61-70
7.1. Web database application source program and screenshots showing Its successful compilation	61-68
7.2. Screenshots showing the testing of the Web database application	70

**Task 1. :** Design an ER diagram to represent the Job-Shop Accounting database defined in part I.



**Task 2.** Convert the ER diagram in Task 1 to a Relational Database (i.e. a set of relational schemas).

customerInfo (Customer\_Name, C\_address, Category)

Assembly (Assembly\_ID, Date\_ordered, Assembly\_details, Customer Name)

Department (Department ID, Department Data)

Process1 (Process\_ID, Process\_Data, Department\_ID)

Process\_Fit (Process\_ID, Fit\_Type)

Process\_Paint (Process\_ID, Paint\_Type, Paint\_Method)

Process\_Cut (Process\_ID, Cut\_type, Machine\_type)

Job (Job\_No, Date\_job\_commenced, Date\_job\_completed, Assembly\_ID, Process\_ID)

Job\_Fit (Job\_No, Labor\_Time)

Job\_Paint (Job\_No, Color, Volume, Paint\_Labor\_Time)

Job\_Cut (Job\_No, Job\_Machine\_Type, Machine\_Time, Material, Cut\_Labor\_Time)

Department\_acc1 (Account\_No, Established\_Date, Date\_established, details\_2,

Department\_ID)

Assembly1\_acc (Account\_No, Established\_Date, Date\_established, details\_1, Assembly\_ID)

Process1\_acc (Account\_No, Established\_Date, Date\_established, details\_3, Process\_ID)

Transaction (Transaction\_No, Given Cost, Job\_No, Details 1, Details 2, Details 3)

### Task 3.

**3.1.** Discuss choices of appropriate storage structures for each relational table assuming that all types of storage structures discussed in class (Lecture Topic 4) are available.

Table Name	Query# and Type	Search Key	Query Frequency	Selected File Organization	Justifications
CustomerInfo	#1 Insertion		30 / day	B+ Tree with Index = Category	B+Tree is used to find all records with search key as Category in a specified range [lower bound, Upper Bound].
	#13 Range Search	Category	100 / day		
Assembly	#3 Insertion		40 / day	Heap File	Heap File is used to insert details without imposing any order.
Process1	#4 Insertion		Infrequent	Dynamic Extendable Hashing. Hash key (Process_ID)	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#8 Random Search	Process_ID / Department_ID	50 / day		
	#10 Random Search	Department_ID	20 / day		
	#11 Random Search	Proces_ID	100 / day		
	#12 Random Search	Proces_ID	20 / day		
Process_Fit	#4 insertion		Infrequent	Heap File	Heap File is used to insert details without imposing any order.
Process_Paint	#4 Insertion		Infrequent	Heap File	Heap File is used to insert details without imposing any order.
Process_Cut	#4 Insertion		Infrequent	Heap File	Heap File is used to insert details without imposing any order.

Department	#2 Insertion	Infrequent	Heap File		
Job	#6 Insertion		50 / day	Dynamic Extendable Hashing with hash key (Job_No).	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#7 Random Search	Job_No	50 / day		
	#8 Random Search	Job_No	50 / day		
	#10 Random search	Process_ID / Date_completed	20 /day		
	#11 Random Search	Assembly_ID	100 / day		
	#12 Random Search	Date_job_comple ted	20 / day		
	#14 Random Search	Job_No	1 / month		
Job_fit	#7 Insert		50 / day	Dynamic Extendable Hashing with hash key (Job_No).	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#10 Random Search	Job_No	20 /day		
	#12 Random Search	Job_No	20 /day		
Job_Paint	#7 Insertion		50 / day	Dynamic Extendable Hashing with hash key (Job_No).	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#10 Random Search	Job_No	20 / day		
	#12 Random Search	Job_No	20 / day		
	#15 Random Search	Job_No	1 / week		
Job_Cut	#7 Insertion		50 / day	B+ Tree with Index = Job_No	B+Tree is used to find all records with search key as Category in a
	#10	Job_No	20 /day		

	Random Search				specified range [lower bound, Upper Bound].
	#12 Random Search	Job_No	20 /day		
	#14 Range Search	Job_No	1 / month		
Department_ac	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (Account_No) .	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#8 Random Search	Department_ID	50 / day		
	#8 random Search (update)	Account_No	50 / day		
Assembly_accc	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (Account_No) .	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#8 Random Search	Assembly_ID	50 / day		
	#8 Random Search (update)	Account_No	50 / day		
	#9 Random Search	Assembly_ID	200 / day		
Process_acc	#5 Insertion		10 / day	Dynamic Extendable Hashing with hash key (Account_No) .	Dynamic hashing is used because it managed storage file and it is efficient for random search to retrieve data.
	#8 Random Search	Process_ID	50 / day		
	#8 Random Search (update)	Account_No	50 / day		
C_Transaction	#8 Insertion	50 / day	50 / day	Heap File	Heap File is used to insert details without imposing any order.

**3.2.** Discuss choices of storage structures for each relational table when implementing it in Azure SQL Database (if different from the previous choices specified in 3.1).

A relational database's implementation in Azure SQL Database to store each table is determined by several factors such as the nature of the data, access patterns, and performance considerations.

I do not change any of the search key.

Table	Index Key
customerInfo	Category
Process1	Department ID
Job1	Assembly ID
Department acc1	Department ID
Assembly1 acc	Assembly ID
Process1 acc	Process ID

**Task 4.** Construct SQL statements to create tables and implement them on Azure SQL Database.

```
-- Create Customer table
CREATE TABLE customerInfo (
    Customer_Name VARCHAR(20),
    C_address VARCHAR(100),
    Category INT,
    CONSTRAINT Category CHECK (Category BETWEEN 1 AND 10),
    PRIMARY KEY (Customer_Name)
);
SELECT*FROM customerInfo;
```

Customer_Name	C_address	Category
No results		

```
-- Create Assembly table
CREATE TABLE Assembly1 (
    Assembly_ID VARCHAR(10),
    Date_ordered DATE,
    Assembly_details VARCHAR(100),
    Customer_name VARCHAR(20),
    PRIMARY KEY (Assembly_ID),
    FOREIGN KEY (Customer_name) REFERENCES CustomerInfo(Customer_name)
);
SELECT*FROM Assembly1;
```

```
--Create Department Table
CREATE TABLE Department (
    Department_ID VARCHAR(10),
    Department_Data VARCHAR(100),
    PRIMARY KEY (Department_ID)
);
SELECT*FROM Department;
```

```
--Create Process Table
CREATE TABLE Process1 (
    Process_ID VARCHAR(10) PRIMARY KEY,
    Process_Data VARCHAR(100),
    Department_ID VARCHAR(10) FOREIGN KEY REFERENCES Department(Department_ID)
);
SELECT*FROM Process1;
```

Process_ID	Process_Data	Department_ID
No results		

```
--Create Process_Fit Table
CREATE TABLE Process_Fit (
    Process_ID VARCHAR(10) FOREIGN KEY REFERENCES Process1(Process_ID),
    Fit_Type VARCHAR(10),
```

```

PRIMARY KEY (Process_ID)
);
SELECT*FROM Process_Fit;

```

Process_ID	Fit_Type
No results	

```

--Create Process_Paint Table
CREATE TABLE Process_Paint (
    Process_ID VARCHAR(10) FOREIGN KEY REFERENCES Process1(Process_ID),
    Paint_Type VARCHAR(8),
    Paint_Method VARCHAR(8),
    PRIMARY KEY (Process_ID)
);
SELECT*FROM Process_Paint;

```

Process_ID	Paint_Type	Paint_Method
No results		

```

--Create Process_Cut Table
CREATE TABLE Process_Cut (
    Process_ID VARCHAR(10) FOREIGN KEY REFERENCES Process1(Process_ID),
    Cut_Type VARCHAR(8),
    Machine_type VARCHAR(10),
    PRIMARY KEY (Process_ID)
);
SELECT*FROM Process_Cut;

```

Process_ID	Cut_Type	Machine_type
No results		

```

--Create Job Table
CREATE TABLE Job1 (

```

```

Job_No INT PRIMARY KEY,
Date_job_commenced DATE,
Date_job_completed DATE,
Assembly_ID VARCHAR(10) FOREIGN KEY REFERENCES Assembly1(Assembly_ID),
Process_ID VARCHAR(10) FOREIGN key REFERENCES Process1(Process_ID)
);
SELECT*FROM Job1;

```

Job_No	Date_job_commenced	Date_job_completed	Assembly_ID	Process_ID
No results				

--Create Job\_Fit Table

```

CREATE TABLE Job_Fit (
    Job_No INT FOREIGN KEY REFERENCES Job1(Job_No),
    Labor_Time TIME,
    PRIMARY KEY (Job_No)
);
SELECT*FROM Job_Fit;

```

Job_No	Labor_Time
No results	

--Create Job\_Paint Table

```

CREATE TABLE Job_Paint (
    Job_No INT FOREIGN KEY REFERENCES Job1(Job_No),
    Color VARCHAR(10),
    Volume INT,
    Paint_Labor_Time TIME,
    PRIMARY KEY (Job_No)
);
SELECT*FROM Job_Paint;

```

Job_No	Color	Volume	Paint_Labor_Time
No results			

--Create Job\_Cut Table

```
CREATE TABLE Job_Cut (
    Job_No INT FOREIGN KEY REFERENCES Job1(Job_No),
    Job_Machine_Type VARCHAR(10),
    Machine_Time TIME,
    Material VARCHAR(6),
    Cut_Labor_Time TIME,
    PRIMARY KEY (Job_No)
);
SELECT*FROM Job_Cut;
```

Job_No	Job_Machine_Type	Machine_Time	Material	Cut_Labor_Time
No results				

--Create Department Account Table

```
CREATE TABLE Department_acc1(
    Account_No INT,
    Established_Date DATE,
    details_2 FLOAT,
    Department_ID VARCHAR(10) FOREIGN KEY REFERENCES Department(Department_ID) UNIQUE,
    PRIMARY KEY (Account_No)
);
SELECT*FROM Department_acc1;
```

Dept_Account_No	Established_Date	details_2	Department_ID
No results			

--Create Assembly Account

```
CREATE TABLE Assembly1_acc (
```

```

Account_No INT,
Established_Date DATE,
Details_1 FLOAT,
Assembly_ID VARCHAR(10) FOREIGN KEY REFERENCES Assembly1(Assembly_ID) UNIQUE,
PRIMARY KEY (Account_No)
);

```

```
SELECT*FROM Assembly1_acc;
```

Results    Messages

Account_No	Established_Date	Details_1	Assembly_ID
No results			

--Create Process Account

```

CREATE TABLE Process1_acc (
Account_No INT,
Established_Date DATE,
Details_3 FLOAT,
Process_ID VARCHAR(10) FOREIGN KEY REFERENCES Process1(Process_ID) UNIQUE,
PRIMARY KEY (Account_No)
);

```

Results    Messages

Account_No	Established_Date	Details_3	Process_ID
No results			

--Create Customer Transaction Table

```

CREATE TABLE C_Transaction (
Transaction_No VARCHAR(10) PRIMARY KEY,
Given_Cost FLOAT,
Job_No INT FOREIGN KEY REFERENCES Job1(Job_No),
Department_acc INT FOREIGN KEY REFERENCES Department_acc(Dept_Account_No),
Assembly1_acc INT FOREIGN KEY REFERENCES Assembly1_acc(Assembly_Account_No),
Process1_acc INT FOREIGN KEY REFERENCES Process1_acc(Process_Account_No)
);

```

```
SELECT*FROM C_Transaction;
```

Results	Messages
<input type="text"/> Search to filter items...	
Transaction_No	Given_Cost
Job_No	Department_acc
	Assembly1_acc
	Process1_acc
No results	

```

/*Create Index
*/
-- Create index on Phone Number - customer table
CREATE INDEX customer_Category ON CustomerInfo(Category);

-- Create index on Department_No - Process table
CREATE INDEX process_department ON Process1(Department_ID);

-- Create index on Assembly_ID - Job table
CREATE INDEX Assembly_Job ON job1(Assembly_ID);

-- Create index on Department_No - Department_acc table
CREATE INDEX Department_No_Acc ON Department_acc(Department_ID);

-- Create index on Assembly_ID - Assembly_acc table
CREATE INDEX Assembly_ID_acc ON Assembly1_acc(Assembly_ID);

-- Create index on process_ID - Process_acc table
CREATE INDEX Process_ID_acc ON Process1_acc(Process_ID);

```

**Results**      **Messages**

---

Query succeeded: Affected rows: 0

**Task 5a. (Task 5 and Task 6 together):** Write SQL statements for all queries (1-14) defined in part I.

SQL code to execute all queries

--CREATE PROCEDURES

-- Enter a new customer

```
CREATE PROCEDURE InsertCustomerInfo
    @Customer_Name VARCHAR(20),
    @C_address VARCHAR(100),
    @Category INT
AS
BEGIN
    INSERT INTO customerInfo
    (Customer_Name, C_address, Category)
    VALUES (@Customer_Name, @C_address, @Category)
END
GO
```

-- Enter a new Department

```
CREATE PROCEDURE InsertDepartmentInfo
    @Department_ID VARCHAR(10),
    @Department_Data VARCHAR(100)
AS
BEGIN
    INSERT INTO Department
    (Department_ID, Department_Data)
    VALUES (@Department_ID, @Department_Data)
END
GO
```

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

CREATE PROCEDURE InsertAssemblyInfo
    @Assembly_ID VARCHAR(10),
    @Date_ordered VARCHAR(10),
    @Assembly_details VARCHAR(100),
    @Customer_Name VARCHAR(20)

AS
BEGIN
    INSERT INTO Assembly1
    (
        Assembly_ID, Date_ordered, Assembly_details, Customer_Name)
    VALUES (@Assembly_ID, CAST (@Date_ordered as DATE), @Assembly_details, @Customer_Name)
END
GO
```

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

CREATE PROCEDURE InsertProcessIDdept
    @Process_ID VARCHAR(10),
    @Process_Data VARCHAR(100),
    @Department_ID VARCHAR(10)

AS
BEGIN
    INSERT INTO Process1
    (Process_ID, Process_Data, Department_ID)
    VALUES (@Process_ID, @Process_Data, @Department_ID)
END
GO
```

```
CREATE PROCEDURE InsertProcessFit
    @Process_ID VARCHAR(10),
    @Fit_Type VARCHAR(10)

AS
BEGIN
    INSERT INTO Process_Fit
    (Process_ID, Fit_Type)
```

```

VALUES (@Process_ID, @Fit_Type)
END

GO
CREATE PROCEDURE InsertProcessPaint
    @Process_ID VARCHAR(10),
    @Paint_Type VARCHAR(8),
    @Paint_Method VARCHAR(8)
AS
BEGIN
    INSERT INTO Process_Paint
    (Process_ID, Paint_Type, Paint_Method)
    VALUES (@Process_ID, @Paint_Type, @Paint_Method)
END
GO

CREATE PROCEDURE InsertProcessCut
    @Process_ID VARCHAR(10),
    @Cut_Type VARCHAR(8),
    @Machine_Type VARCHAR(10)
AS
BEGIN
    INSERT INTO Process_Cut
    (Process_ID, Cut_Type, Machine_Type)
    VALUES (@Process_ID, @Cut_Type, @Machine_Type)
END

GO
-- Create a new account and associate it with the
-- process, assembly, or department to which it is applicable
CREATE PROCEDURE InsertAccountDept
    @Account_No INT,
    @Details_2 FLOAT,
    @Department_ID VARCHAR(10)
AS
BEGIN
    INSERT INTO Department_acc

```

```
(Account_No, Details_2, Department_ID)
VALUES (@Account_No, @Details_2, @Department_ID)
END
```

```
GO
CREATE PROCEDURE InsertAccountAssembly
    @Account_No INT,
    @Details_1 FLOAT,
    @Assembly_ID VARCHAR(10)
AS
BEGIN
    INSERT INTO Assembly1_acc
        (Account_No, Details_1, Assembly_ID)
    VALUES (@Account_No, @Details_1, @Assembly_ID)
END
```

```
GO
CREATE PROCEDURE InsertAccountProcess
    @Account_No INT,
    @Details_3 FLOAT,
    @Process_ID VARCHAR(10)
AS
BEGIN
    INSERT INTO Process1_acc
        (Account_No, Details_3, Process_ID)
    VALUES (@Account_No, @Details_3, @Process_ID)
END
```

```
GO
-- Enter a new job, given its job-no, assembly-id, process-id, and date the job commenced
CREATE PROCEDURE InsertJobInfo
    @Job_No INT,
    @Date_job_commenced VARCHAR(10),
    @Assembly_ID VARCHAR(10),
    @Process_ID VARCHAR(7)
AS
BEGIN
```

```

INSERT INTO Job
(Job_No, Date_job_commenced, Assembly_ID, Process_ID)
VALUES (@Job_No, CAST(@Date_job_commenced as DATE), @Assembly_ID, @Process_ID)
END
GO

-- At the completion of a job, enter the date it completed
--and the information relevant to the type of job
CREATE PROCEDURE UpdateJobInfo
@Job_No INT,
@Date_job_completed VARCHAR(10)

AS
BEGIN
    UPDATE Job SET Date_job_completed = @Date_job_completed WHERE Job_No = @Job_No
END
GO

CREATE PROCEDURE InsertJobFitInfo
@Job_No INT,
@Labor_Time VARCHAR(10)

AS
BEGIN
    INSERT INTO Job_Fit
(Job_No, Labor_Time)
VALUES (@Job_No, CAST(@Labor_Time as TIME))
END
GO

CREATE PROCEDURE InsertJobPaintInfo
@Job_No INT,
@Color VARCHAR(10),
@Volume INT,
@Paint_Labor_Time VARCHAR(10)

AS
BEGIN
    INSERT INTO Job_Paint

```

```

(Job_No, Color, Volume, Paint_Labor_Time)
VALUES (@Job_No, @Color, @Volume, CAST(@Paint_Labor_Time as TIME))
END
GO

```

```

CREATE PROCEDURE InsertJobCutInfo
@Job_No INT,
@Job_Machine_Type VARCHAR(10),
@Machine_Time VARCHAR(10),
@Material VARCHAR(6),
@Cut_Labor_Time VARCHAR(10)
AS
BEGIN
INSERT INTO Job_cut
(Job_No, Job_Machine_Type, Machine_Time, Material, Cut_Labor_Time)
VALUES (@Job_No, @Job_Machine_Type, CAST(@Machine_Time as TIME), @Material,
CAST(@Cut_Labor_Time as TIME)))
END
GO

```

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

```

CREATE PROCEDURE InsertTransactionInfo
@Transaction_No VARCHAR(10),
@Given_Cost FLOAT,
@Job_No INT
AS
BEGIN
DECLARE @Dept_acc_no INT,
@Assembly_acc_no INT,
@Process_acc_no INT;

SET @Dept_acc_no = (SELECT Account_No
FROM Department_acc, Process1, Job1
WHERE Process1.Process_ID = Job1.Process_ID AND Job1.Job_No = @Job_No AND
Department_acc.Department_ID = Process1.Department_ID);

```

```

SET @Assembly_acc_no = (SELECT Account_No
    FROM Assembly1_acc, Job1
    WHERE Assembly1_acc.Assembly_ID = Job1.Assembly_ID AND Job1.Job_No = @Job_No);

SET @Process_acc_no = (SELECT Account_No
    FROM Process1_acc, Job1
    WHERE Process1_acc.Process_ID = Job1.Process_ID AND Job1.Job_No = @Job_No);

INSERT INTO C_Transaction
(Transaction_No, Given_Cost, Job_No, Department_No, Assembly_acc, Process1_acc)
VALUES (@Transaction_No, @Given_Cost, @Job_No, @Dept_acc_no, @Assembly_acc_no, @Process_acc_no);

UPDATE Department_acc
SET Details_2 = Details_2 + @Given_Cost
WHERE Account_No = @Dept_acc_no;

UPDATE Assembly1_acc
SET Details_1 = Details_1 + @Given_Cost
WHERE Account_No = @Assembly_acc_no;

UPDATE Process1_acc
SET Details_3 = Details_3 + @Given_Cost
WHERE Account_No = @Process_acc_no;
END
GO

--Retrieve the cost incurred on an assembly_ID
CREATE PROCEDURE RetriveCost
    @Assembly_ID VARCHAR(10)

AS
BEGIN
    SELECT Details_1 FROM Assembly1_acc WHERE Assembly1_acc.Assembly_ID = @Assembly_ID;
END

GO
-- Retrieve the total labor time within a department for jobs completed

```

```

-- in the department during a given date

CREATE PROCEDURE RetrieveLaboreTime
    @Department_ID VARCHAR(10),
    @Date_job_completed VARCHAR(10)

AS
BEGIN
    DECLARE @Fit_Labor_Time FLOAT,
            @Paint_Labor_Time FLOAT,
            @Cut_Labor_Time FLOAT,
            @Total_Labor_Time FLOAT;

    SET @Fit_Labor_Time = (SELECT SUM(( DATEPART(hh, Labor_Time) * 3600 ) + ( DATEPART(mi,Labor_Time)
* 60 ) + DATEPART(ss,Labor_time))/60 as minute
    FROM Job_Fit WHERE Job_Fit.Job_No in (
        SELECT distinct(Job_No) FROM Job1
        WHERE Job1.Process_ID in (SELECT distinct(Process_ID) FROM Process1 WHERE Process1.Department_ID =
        @Department_ID) AND Job1.Date_job_completed = CAST(@Date_job_completed as DATE)));
    IF @Fit_Labor_Time IS NULL SET @Fit_Labor_Time = 0;

    SET @Paint_Labor_Time = (SELECT SUM(( DATEPART(hh, Paint_Labor_Time) * 3600 ) + ( DATEPART(mi,
    Paint_Labor_Time) * 60 ) + DATEPART(ss, Paint_Labor_Time))/60 as minute
    FROM Job_Paint WHERE Job_Paint.Job_No in (
        SELECT distinct(Job_No) FROM Job1
        WHERE Job1.Process_ID in (SELECT distinct(Process_ID) FROM Process1 WHERE Process1.Department_ID =
        @Department_ID) AND Job1.Date_job_completed =CAST(@Date_job_completed as DATE)));
    IF @Paint_Labor_Time IS NULL SET @Paint_Labor_Time = 0;

    SET @Cut_Labor_Time = (SELECT SUM(( DATEPART(hh, Cut_Labor_Time) * 3600 ) + ( DATEPART(mi,
    Cut_Labor_Time) * 60 ) + DATEPART(ss, Cut_Labor_Time))/60 as minute
    FROM Job_Cut WHERE Job_Cut.Job_No in (
        SELECT distinct(Job_No) FROM Job1
        WHERE Job1.Process_ID in (SELECT distinct(Process_ID) FROM Process1 WHERE Process1.Department_ID =
        @Department_ID) AND Job1.Date_job_completed =CAST(@Date_job_completed as DATE)));
    IF @Cut_Labor_Time IS NULL SET @Cut_Labor_Time = 0;

```

```

SET @Total_Labor_Time = @Fit_Labor_Time + @Paint_Labor_Time + @Cut_Labor_Time
SELECT @Total_Labor_Time
END
GO

-- Retrieve the processes through which a given assembly-id has passed so far (in date_job-commenced order) and
the department responsible for each process
CREATE PROCEDURE RetrieveProcessAssembly
    @Assembly_ID VARCHAR(10)
AS
BEGIN
    SELECT Job1.Date_job_commenced, Job1.Process_ID, Process1.Department_ID
    FROM Job1, Process1
    WHERE Job1.Assembly_ID = @Assembly_ID AND Process1.Process_ID = Job1.Process_ID
    ORDER BY 1 ;
END
GO
-- Retrieve the jobs (together with their type information and assembly-id) completed during a given date in a given
department
CREATE PROCEDURE RetrieveJobAssemblyFit
    @Date_job_completed VARCHAR(10),
    @Department_ID VARCHAR(10)
AS
BEGIN
    SELECT DISTINCT(Job1.Job_No), Job1.Assembly_ID, Job_Fit.Labor_Time
    FROM Job1, Job_Fit
    WHERE Date_job_completed = @Date_job_completed and Job1.Process_ID in (SELECT Process1.Process_ID
    FROM Process1 WHERE Department_ID = @Department_ID) AND Job_Fit.Job_No = Job1.Job_No;
END
GO

CREATE PROCEDURE RetrieveJobAssemblyPaint
    @Date_job_completed VARCHAR(10),

```

```

    @Department_ID VARCHAR(10)

AS
BEGIN
    SELECT DISTINCT(Job1.Job_No), Job1.Assembly_ID, Job_Paint.Color, Job_Paint.Volume,
    Job_Paint.Paint_Labor_Time
    FROM Job1, Job_Paint
    WHERE Date_job_completed = @Date_job_completed and Job1.Process_ID in (SELECT process_ID FROM
    Process1 WHERE Department_ID = @Department_ID) AND Job_Paint.Job_No = Job1.Job_No;

END
GO
CREATE PROCEDURE RetrieveJobAssemblyCut
    @Date_job_completed VARCHAR(10),
    @Department_ID VARCHAR(10)

AS
BEGIN
    SELECT DISTINCT(Job1.Job_No), Job1.Assembly_ID, Job_Cut.Job_Machine_Type, Job_Cut.Machine_Time,
    Job_Cut.Material, Job_Cut.Cut_Labor_Time
    FROM Job1, Job_Cut
    WHERE Date_job_completed = @Date_job_completed and Job1.Process_ID in (SELECT Process1.Process_ID
    FROM Process1 WHERE Department_ID = @Department_ID) AND Job_Cut.Job_No = Job1.Job_No;

END
GO
-- Retrieve the customers (in name order) whose category is in a given range
CREATE PROCEDURE RetrieveCustomerByCategory
    @Lower_b INT,
    @Upper_b INT

AS
BEGIN
    SELECT Customer_Name, Category AS Name FROM customerInfo
    WHERE Category >= @Lower_b AND Category <= @Upper_b
    ORDER BY 1 ;
END
GO

```

```

-- Delete all cut-jobs whose job-no is in a given range
CREATE PROCEDURE DeleteCutJob
    @Lower_b INT,
    @Upper_b INT
AS
BEGIN
    DELETE FROM Job_Cut WHERE Job_No >= @Lower_b AND Job_No <= @Upper_b
END

GO
-- Change the color of a given paint
CREATE PROCEDURE ChangePaintColor
    @Job_No INT,
    @Color VARCHAR(10)
AS
BEGIN
    UPDATE Job_Paint SET Color = @Color WHERE Job_No = @Job_No;
END
GO

CREATE PROCEDURE DeleteJobCutIdentified
    --@lower_b INT,
    -- @upper_b INT
AS
BEGIN
    DECLARE @JB INT

    SET @JB = (SELECT DISTINCT( Job_Cut.Job_No ) FROM Job_Cut WHERE Job_No >= 1 AND Job_No <= 10 )
    --DELETE FROM Job_cut WHERE job_no >= @lower_b AND job_no <= @upper_b
    SELECT @JB;
END

```

**Task 5b.** Write a Java application program that uses JDBC and Azure SQL Database to implement all SQL queries (options 1-14),

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

```
try {// initializing Scanner object
Scanner myScan = new Scanner(System.in));
//reading the input given by the user
Choice = myScan.nextInt();

//Depending on the choice made by the user, we are defining the operations to be done
switch (Choice) {

    case 1:
        // try and catch are used to not terminate loop in case of error.
        try {
            //Declaring the variables
            int Category;
            String Customer_Name, C_address;

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

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

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

            // Executing procedure for Query 1.
            final String Sql1 = "EXEC InsertCustomerInfo @Customer_Name = '"+Customer_Name+"', @C_address = '"+C_address+"', @Category = '"+Category+"';

            final Statement statement1 = connection.createStatement();
            statement1.executeUpdate(Sql1);
            System.out.println("Customer details inserted successfully.");
            System.out.println("=====");
        } catch (Exception e) {
            System.out.println("Error!. Returning to main menu");
        }
        break;
}
```

2. Enter a new department (infrequent).

```
.01
.02
.03     case 2:
.04         // try and catch are used to not terminate loop in case of error.
.05         try {
.06
.07             // Declaring variables
.08             String Department_ID, Department_Data;
.09
.10            // Taking Department Number from user.
.11            System.out.println("Enter the Department ID\n");
.12            Department_ID = myScan.nextInt();
.13
.14            // Taking Department data from user.
.15            System.out.println("Enter the Department Data\n");
.16            Department_Data = myScan.nextLine();
.17
.18            // Executing Procedure for Query 2.
.19            final String Sql2 = "EXEC InsertDepartmentInfo @Department_ID = '"+Department_ID+"', @Department_Data = '"+Department_Data+"'"; 
.20
.21            final Statement statement2 = connection.createStatement();
.22            statement2.executeUpdate(Sql2);
.23
.24            System.out.println("Department details inserted successfully.");
.25            System.out.println("=====");
.26        } catch (Exception e) {
.27            System.out.println("Error!. Returning to main menu");
.28        }
.29    }
.30
.31
```

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

```
case 3:
    // try and catch are used to not terminate loop in case of error.
    try {
        // Declaring Variables
        String Assembly_ID, Assembly_details, Customer_Name;
        String Date_ordered;

        // Taking Assembly-id from user
        System.out.println("Enter Assembly ID");
        Assembly_ID = myScan.nextLine();

        // Taking date ordered from user
        System.out.println("Enter Date Ordered in yyyy-mm-dd format");
        Date_ordered = myScan.nextLine();

        // Taking Assembly details from user
        System.out.println("Enter Assembly Details");
        Assembly_details = myScan.nextLine();

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

        // Executing procedure for Query 3
        final String Sql3 = "EXEC InsertAssemblyInfo @Assembly_ID = '"+Assembly_ID+"', @Date_ordered = '"+Date_ordered+"', " +
                            "@Assembly_details = '"+Assembly_details+"', @Customer_Name = '"+Customer_Name+"'";

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

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

    } catch (Exception e) {
        System.out.println("Error!. Returning to main menu");
    }
    break;
```

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

```

169
170    // try and catch are used to not terminate loop in case of error.
171    try {
172
173        // Declaring Variables
174        String Process_ID, Process_Data, Department_ID;
175
176        // Taking process-id from the user
177        System.out.println("Enter Process ID");
178        Process_ID = myScan.next();
179
180        // Taking process data from the user
181        System.out.println("Enter Process Data");
182        Process_Data = myScan.next();
183
184        // Taking Department-no from the user
185        System.out.println("Enter Department ID");
186        Department_ID = myScan.next();
187
188        //Executing Procedure for Query 4
189        final String Sql4 = "EXEC InsertProcessIDdept @Process_ID = '"+Process_ID+"', " +
190            " @Process_Data = '"+Process_Data+"', @Department_ID = '"+Department_ID+"';"
191
192        final Statement statement4 = connection.createStatement();
193        statement4.executeUpdate(Sql4);
194
195        System.out.println("Process details inserted successfully.");
196        System.out.println("=====");
197
198        // Asking user to enter the type of procedure
199        System.out.println("Choose one of the following type of process:\n 1.Fit\n 2. Paint\n 3.Cut\n");
200        int choice1;
201        // Taking the type of process from the user
202        choice1 = myScan.nextInt();
203
204        if (choice1 ==1) {
205
206            // Declaring Variables
207            String Fit_Type;
208
209            // Taking fit type from user
210            System.out.println("Enter fit type\n");
211            Fit_Type = myScan.next();
212
213            // Executing Procedure to insert data in Process_fit table
214            final String Sql4_1 = "EXEC InsertProcessFit @Process_ID = '"+Process_ID+"', @Fit_Type = '"+Fit_Type+"';"
215
216            final Statement statement4_1 = connection.createStatement();
217            statement4_1.executeUpdate(Sql4_1);
218
219            System.out.println("Process_Fit details inserted successfully.");
220            System.out.println("=====");
221        }
222
223        System.out.println("=====");
224
225        if (choice1 ==2) {
226
227            // Declaring Variables
228            String Paint_Type, Paint_Method;
229
230            //Taking paint type from the user
231            System.out.println("Enter paint type");
232            Paint_Type = myScan.next();
233
234            // Taking paint method from the user
235            System.out.println("Enter paint method");
236            Paint_Method = myScan.next();
237
238            // Executing procedure to insert data into Process_paint table
239            final String Sql4_2 = "EXEC InsertProcessPaint @Process_ID = '"+Process_ID+"', " +
240                " @Paint_Type = '"+Paint_Type+"', @Paint_Method = '"+Paint_Method+"';"
241
242            final Statement statement4_2 = connection.createStatement();
243            statement4_2.executeUpdate(Sql4_2);
244
245            System.out.println("Process_Paint details inserted successfully.");
246            System.out.println("=====");
247        }
248
249        if (choice1 ==3) {
250
251            // Declaring Variables
252            String Cut_Type, Machine_Type;
253
254            //Taking cut type from the user
255            System.out.println("Enter cutting type");
256            Cut_Type = myScan.next();
257
258            // Take machine type from user
259            System.out.println("Enter machine type");
260            Machine_Type = myScan.next();
261
262            //Executing Procedure to insert data into Process_cut table.
263            final String Sql4_3 = "EXEC InsertProcessCut @Process_ID = '"+Process_ID+"', " +
264                " @Cut_Type = '"+Cut_Type+"', @Machine_Type = '"+Machine_Type+"';"
265
266            final Statement statement4_3 = connection.createStatement();
267            statement4_3.executeUpdate(Sql4_3);
268
269            System.out.println("Process_Cut record inserted successfully.");
270            System.out.println("=====");
271        }
272    } catch (Exception e) {
273        System.out.println("Error!. Returning to main menu");
274    }
275    break;

```

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

```

274
275     case 5:
276         // try and catch are used to not terminate loop in case of error.
277         try {
278
279             // Declaring Variables
280             int Account_No, choice2;
281
282             // Taking account number from user
283             System.out.println("Enter account number");
284             Account_No = myScan.nextInt();
285
286
287             // Asking user to provide the type of account
288             System.out.println("Choose one of the following type of account:\n 1. Department Account.\n 2. Assembly Account\n 3.Process Account\n");
289             choice2 = myScan.nextInt();
290
291
292             if (choice2 == 1) {
293                 // Declaring Variables
294                 float details_2;
295                 String Department_ID;
296                 String Established_Date;
297
298                 // Taking details-2 from the user
299                 System.out.println("Enter account details");
300                 details_2 = myScan.nextFloat();
301
302                 // Taking date account established from user
303                 System.out.println("Enter date account established");
304                 Established_Date = myScan.next();
305
306                 // Taking department no from the user
307                 System.out.println("Enter Department Number of the account");
308                 Department_ID = myScan.next();
309
310                 // Executing procedure to insert data into Department Account Table
311                 final String Sql5_1 = "EXEC InsertAccountDept5 @Account_No = '"+Account_No+"', @Established_Date = '"+Established_Date+"', "+
312                     "@details_2 = '"+details_2+"', @Department_ID = '"+Department_ID+"';";
313
314                 final Statement statement5_1 = connection.createStatement();
315                 statement5_1.executeUpdate(Sql5_1);
316
317                 System.out.println("Department_account record inserted successfully.");
318                 System.out.println("=====");
319             }
320
321             if (choice2 == 2) {
322
323                 // Declaring Variables
324                 float Details_1;
325                 String Assembly_ID;
326                 String Established_Date;
327
328                 // Taking details1 from user
329                 System.out.println("Enter account details");
330                 Details_1 = myScan.nextFloat();

```

```

331
332     // Taking date account established from user
333     System.out.println("Enter date account established");
334     Established_Date = myScan.next();
335
336     // Taking Assembly-ID from user
337     System.out.println("Enter Assembly id of the account");
338     Assembly_ID = myScan.next();
339
340     // Executing procedure to insert data into Assembly account
341     final String Sql5_2 = "EXEC InsertAccountAssembly4 @Account_No = '"+Account_No+"',@Established_Date = '"+Established_Date+"','"++
342         " @Details_1 = '"+Details_1+"',@Assembly_ID = '"+Assembly_ID+"';";
343
344     final Statement statement5_2 = connection.createStatement();
345     statement5_2.executeUpdate(Sql5_2);
346
347     System.out.println("Assembly_acc record inserted successfully.");
348     System.out.println("=====");
349 }
350
351 if (choice2 == 3) {
352
353     // Declaring Variables
354     float Details_3;
355     String Process_ID;
356     String Established_Date;
357
358     // Taking details from the user
359     System.out.println("Enter account details");
360     Details_3 = myScan.nextFloat();
361
362     // Taking date account established from user
363     System.out.println("Enter date account established");
364     Established_Date = myScan.next();
365
366     // Taking process-id from the user
367     System.out.println("Enter Process id of the account");
368     Process_ID = myScan.next();
369
370     // Executing the procedure to insert data into Process account
371     final String Sql5_3 = "EXEC InsertAccountProcess2 @Account_No = '"+Account_No+"',@Established_Date = '"+Established_Date+"','"++
372         " @Details_3 = '"+Details_3+"', @Process_ID = '"+Process_ID+"';";
373
374     final Statement statement5_3 = connection.createStatement();
375     statement5_3.executeUpdate(Sql5_3);
376
377     System.out.println("Process_acc record inserted successfully.");
378     System.out.println("=====");
379 }
380
381 } catch (Exception e) {
382     System.out.println("Error!. Returning to main menu");
383 }
384
385 break;
386

```

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

```

386
387     case 6:
388         // try and catch are used to not terminate loop in case of error.
389         try {
390
391             // Declaring Variables
392             String Date_job_commenced, Assembly_ID, Process_ID;
393             int Job_No;
394
395             // Taking Job-no from the user
396             System.out.println("Enter Job-No for a new job");
397             Job_No = myScan.nextInt();
398
399             // Taking assembly-id from the user
400             System.out.println("Enter assembly-ID for the job");
401             Assembly_ID = myScan.next();
402
403             // Taking process-id from user
404             System.out.println("Enter process-ID for the job");
405             Process_ID = myScan.next();
406
407             // Taking job commenced from user
408             System.out.println("Enter the commenced date for the job");
409             Date_job_commenced = myScan.next();
410
411             // Executing Procedure for Query 6
412             final String Sql6 = "EXEC InsertJobInfo1 @Job_No = '"+Job_No+"', @Date_job_commenced = '"+Date_job_commenced+"','"++
413                 " @Assembly_ID = '"+Assembly_ID+"', @Process_ID = '"+Process_ID+"';";
414
415             final Statement statement6 = connection.createStatement();
416             statement6.executeUpdate(Sql6);
417
418             System.out.println("Job record inserted successfully.");
419             System.out.println("=====");
420
421         } catch (Exception e) {
422             System.out.println("Error!. Returning to main menu");
423         }
424
425         break;
426

```

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

type of job (50/day).

```
425
426     case 7:
427         // try and catch are used to not terminate loop in case of error.
428         try {
429             // Declaring Variables
430             String Date_job_completed;
431             int Job_No;
432
433             // Taking Job-no from user
434             System.out.println("Enter Job-No for the completed job");
435             Job_No = myScan.nextInt();
436
437             // Taking date completed from user
438             System.out.println("Enter job completed date in yyyy-mm-dd format");
439             Date_job_completed = myScan.next();
440
441             // Executing the Procedure for Query 7
442             final String Sql7 = "EXEC UpdateJobInfo1 @Job_No = '"+Job_No+"', @Date_job_completed = '"+Date_job_completed+"'"; 
443
444             final Statement statement7 = connection.createStatement();
445             statement7.executeUpdate(Sql7);
446
447             System.out.println("Job record Updated successfully.");
448             System.out.println("=====");
449
450             // Declaring Variables
451             int choice3;
452
453             // Asking the type of job from the user
454             System.out.println("Choose one of the following type of job:\n 1. Fit Job.\n 2. Paint Job\n 3.Cut Job\n");
455             choice3 = myScan.nextInt();
456
457             if (choice3 == 1) {
458                 // Declaring Variables
459                 String Labor_Time;
460
461                 // Taking fit labor time from user
462                 System.out.println("Enter the fit labor time fo Job in HH:MM:SS format.");
463                 Labor_Time = myScan.next();
464
465                 // Executing procedure to insert data into Job_fit table
466                 final String Sql7_1 = "EXEC InsertJobFitInfo @Job_No = '"+Job_No+"', @Labor_Time = '"+Labor_Time+"'"; 
467
468                 final Statement statement7_1 = connection.createStatement();
469                 statement7_1.executeUpdate(Sql7_1);
470
471                 System.out.println("Fit Job record inserted successfully.");
472                 System.out.println("=====");
473             }
474
475             if (choice3 == 2) {
476                 // Declaring Variables
477                 String Color, Paint_Labor_Time;
478                 int Volume;
```

```

482 // Taking color from user
483 System.out.println("Enter the paint color.");
484 Color = myScan.next();
485
486 // Taking labor time from user
487 System.out.println("Enter the paint job labor time in HH:MM:SS format.");
488 Paint_Labor_Time = myScan.next();
489
490 // Taking volume of paint from user
491 System.out.println("Enter the volume of paint.");
492 Volume = myScan.nextInt();
493
494 // Executing the procedure to insert data into Job_paint table
495 final String Sql7_2 = "EXEC InsertJobPaintInfo @Job_No = '"+Job_No+"', @Color = '"+Color+"', "+
496 " @Volume = '"+Volume+"', @Paint_Labor_Time = '"+Paint_Labor_Time+"'";+
497
498 final Statement statement7_2 = connection.createStatement();
499 statement7_2.executeUpdate(Sql7_2);
500
501 System.out.println("Paint Job record inserted successfully.");
502 System.out.println("=====");
503 }
504
505 if (choice3 == 3) {
506
507     // Declaring Variables
508     String Job_Machine_Type, Machine_Time, Material, Cut_Labor_Time;
509
510     // Taking machine type from user
511     System.out.println("Enter the Job Machine Type.");
512     Job_Machine_Type = myScan.next();
513
514     // Taking machine time from user
515     System.out.println("Enter the cut job machine time in HH:MM:SS format.");
516     Machine_Time = myScan.next();
517
518     // Taking material used from the user
519     System.out.println("Enter the material used.");
520     Material = myScan.next();
521
522     // Taking labor time from the user
523     System.out.println("Enter the cut job labor time in HH:MM:SS format.");
524     Cut_Labor_Time = myScan.next();
525
526     // Executing Procedure to insert data into Job_cut table
527     final String Sql7_3 = "EXEC InsertJobCutInfo @Job_No = '"+Job_No+"', @Job_Machine_Type= '"+Job_Machine_Type+"', "+
528 " @Machine_Time = '"+Machine_Time+"', @Material = '"+Material+"', @Cut_Labor_Time = '"+Cut_Labor_Time+"'";+
529
530     final Statement statement7_3 = connection.createStatement();
531     statement7_3.executeUpdate(Sql7_3);
532
533     System.out.println("Cut Job record inserted successfully.");
534     System.out.println("=====");
535 }
536 } catch (Exception e) {
537     System.out.println("Error!. Returning to main menu");
538 }
539 break;

```

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

```

539         break;
540     case 8:
541         // try and catch are used to not terminate loop in case of error.
542         try {
543
544             // Declaring Variables
545             String Transaction_No;
546             int Job_No;
547             float Given_Cost;
548
549             // Taking transaction number from the user
550             System.out.println("Enter the Transaction Number.");
551             Transaction_No = myScan.next();
552
553             // Taking sup-cost from the user
554             System.out.println("Enter the cost of the transaction.");
555             Given_Cost= myScan.nextFloat();
556
557             // Taking job-No from the user
558             System.out.println("Enter the Job number related to transaction.");
559             Job_No = myScan.nextInt();
560
561             // Executing procedure for Query 8.
562             final String Sql8 = "EXEC InsertTransactionInfo @Transaction_No = '"+Transaction_No+"', @Given_Cost = '"+Given_Cost+"', @Job_No = '"+Job_No+"'";+
563
564             final Statement statement8 = connection.createStatement();
565             statement8.executeUpdate(Sql8);
566
567             System.out.println("Transaction record inserted and related accounts updated successfully.");
568             System.out.println("=====");
569
570         } catch (Exception e) {
571             System.out.println("Error!. Returning to main menu");
572         }
573         break;

```

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

```
575
576     case 9:
577         // try and catch are used to not terminate loop in case of error.
578         try {
579             // Declaring Variables
580             String Assembly_ID3;
581
582             // Taking assembly-id from the user
583             System.out.println("Enter Assembly Id to retrieve the cost.");
584             Assembly_ID3 = myScan.next();
585
586             try {
587
588                 // Executing procedure for Query 9
589                 final String Sql9 = "EXEC RetriveCost @Assembly_ID = '"+Assembly_ID3+"'";
590
591                 try (final Statement statement9 = connection.createStatement();
592                      final ResultSet resultSet1 = statement9.executeQuery(Sql9)) {
593
594                     System.out.println(String.format("Cost incurred on Assembly-ID %s:", Assembly_ID3));
595                     while (resultSet1.next()) {
596                         System.out.println(String.format("%f",
597                                         resultSet1.getFloat(1)));
598
599                     }
600                 } catch (Exception e) {
601                     System.out.println("error!. Please try again.");
602                 }
603
604             } catch (Exception e) {
605                 System.out.println("Error!. Returning to main menu");
606             }
607             break;
608
609         ---- 12:
```

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

```
608 case 10:  
609     // try and catch are used to not terminate loop in case of error.  
610     try {  
611         // Declaring Variables  
612         String Department_No2,Date_job_completed1;  
613  
614         // Taking department-no from user  
615         System.out.println("Enter Department Number to get the total labor time.");  
616         Department_No2 = myScan.nextInt();  
617  
618         // Taking job-completed date from user  
619         System.out.println("Enter Job completed date to get the total labor time.");  
620         Date_job_completed1 = myScan.next();  
621  
622         // Executing the procedure for Query 10  
623         final String Sql10 = "EXEC RetrieveLaboreTime @Department_ID = '"+Department_No2+"', @Date_job_completed = '"+Date_job_completed1+"'";  
624  
625         try (final Statement statement10 = connection.createStatement();  
626              final ResultSet resultSet2 = statement10.executeQuery(Sql10)) {  
627  
628             System.out.println(String.format("Total labor-time in minutes for department number %s and date" +  
629                                 "job completed %s:", Department_No2, Date_job_completed1));  
630             while (resultSet2.next()) {  
631                 System.out.println(String.format("%s",  
632                                 resultSet2.getInt(1)));  
633             }  
634         }  
635     } catch (Exception e) {  
636         System.out.println("Error!. Returning to main menu");  
637     }  
638     break;  
639  
640 case 11:  
641  
642
```

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

```

541     case 11:
542         // try and catch are used to not terminate loop in case of error.
543         try {
544
545             // Declaring Variables
546             String Assembly_ID4;
547
548             // Taking assembly-id from user
549             System.out.println("Enter the Assembly Id to retrieve the processes.");
550             Assembly_ID4 = myScan.next();
551
552             // Executing procedure for query 11d
553             final String Sql11 = "EXEC RetrieveProcessAssembly @Assembly_ID = '"+Assembly_ID4+"'";
554
555             try (final Statement statement11 = connection.createStatement());
556                 final ResultSet resultSet3 = statement11.executeQuery(Sql11);
557
558                 System.out.println(String.format("The processes through which Assembly ID %s passed so far:", Assembly_ID4));
559                 while (resultSet3.next()) {
560                     System.out.println(String.format("%s | %s | %s",
561                         resultSet3.getString(1),
562                         resultSet3.getString(2),
563                         resultSet3.getString(3)));
564
565                 }
566             }
567         } catch (Exception e) {
568             System.out.println("Error!. Returning to main menu");
569         }
570         break;
571
572
573

```

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

```

574     case 12:
575         // try and catch are used to not terminate loop in case of error.
576         try {
577
578             // Declaring Variables
579             String Date_job_completed2, Department_No3;
580
581             // Taking date completed from user
582             System.out.println("Enter the date job is completed.");
583             Date_job_completed2 = myScan.next();
584
585             // Taking department number from user
586             System.out.println("Enter the department to retrieve the jobs.");
587             Department_No3 = myScan.next();
588
589             // Executing Procedure to retrieve fit jobs
590             final String Sql12_1 = "EXEC RetrieveJobAssemblyFit @Date_job_completed = '"+Date_job_completed2+"', @Department_ID = '"+Department_No3+"'";
591
592             try (final Statement statement12_1 = connection.createStatement());
593                 final ResultSet resultSet5 = statement12_1.executeQuery(Sql12_1);
594
595                 System.out.println(String.format("The fit jobs completed on date %s in department %s:", Date_job_completed2, Department_No3));
596                 while (resultSet5.next()) {
597                     System.out.println(String.format("%s | %s | %s",
598                         resultSet5.getString(1),
599                         resultSet5.getString(2),
600                         resultSet5.getString(3)));
601
602                 }
603
604             // Executing procedure to retrieve paint jobs
605             final String Sql12_2 = "EXEC RetrieveJobAssemblyPaint @Date_job_completed = '"+Date_job_completed2+"', @Department_ID = '"+Department_No3+"'";
606
607             try (final Statement statement12_2 = connection.createStatement());
608                 final ResultSet resultSet6 = statement12_2.executeQuery(Sql12_2);
609
610                 System.out.println(String.format("The Paint jobs completed on date %s in department %s:", Date_job_completed2, Department_No3));
611                 while (resultSet6.next()) {
612                     System.out.println(String.format("%s | %s | %s | %s | %s",
613                         resultSet6.getString(1),
614                         resultSet6.getString(2),
615                         resultSet6.getString(3),
616                         resultSet6.getString(4),
617                         resultSet6.getString(5)));
618
619                 }
620
621
622

```

```

?1
?2
?3 // Executing procedure to retrieve cut jobs
?4 final String Sql12_3 = "EXEC RetrieveJobAssemblyCut @Date_job_completed = '"+Date_job_completed2+"', @Department_ID = '"+Department_No3+"'";
?5
?6 try (final Statement statement12_3 = connection.createStatement();
?7     final ResultSet resultSet7 = statement12_3.executeQuery(Sql12_3)) {
?8
?9     System.out.println(String.format("The Cut jobs completed on date %s in department %s:", Date_job_completed2, Department_No3));
?10    while (resultSet7.next()) {
?11        System.out.println(String.format("%s | %s | %s | %s | %s | %s",
?12            resultSet7.getString(1),
?13            resultSet7.getString(2),
?14            resultSet7.getString(3),
?15            resultSet7.getString(4),
?16            resultSet7.getString(5),
?17            resultSet7.getString(6)));
?18    }
?19 }
?20 } catch (Exception e) {
?21     System.out.println("Error!. Returning to main menu");
?22 }
?23
?24 break;

```

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

```

747 case 13:
748     // try and catch are used to not terminate loop in case of error.
749     try {
750
751         // Declaring Variables
752         int Lower_b, Upper_b;
753
754         // Taking lower bound of category from user
755         System.out.println("Enter the lower bound of the category.");
756         Lower_b = myScan.nextInt();
757
758         // Taking upper bound of category from user
759         System.out.println("Enter the upper bound of the category.");
760         Upper_b = myScan.nextInt();
761
762         // Executing procedure for Query 13
763         final String Sql13 = "EXEC RetrieveCustomerByCategory @Lower_b = '"+Lower_b+"', @Upper_b = '"+Upper_b+"'";
764
765     try (final Statement statement13 = connection.createStatement();
766         final ResultSet resultSet4 = statement13.executeQuery(Sql13)) {
767
768         System.out.println("The customers in the given range of category are");
769         while (resultSet4.next()) {
770             System.out.println(String.format("%s | %s",
771                 resultSet4.getString(1),
772                 resultSet4.getString(2)));
773
774         }
775     } catch (Exception e) {
776         System.out.println("Error!. Returning to main menu");
777     }
778
779     break;
780

```

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

```

781     case 14:
782         // try and catch are used to not terminate loop in case of error.
783         try {
784             // Declaring Variables
785             int lower_b1, upper_b1;
786
787             // Taking lower bound job-no from user
788             System.out.println("Enter the lower bound of the cut job-no.");
789             lower_b1 = myScan.nextInt();
790
791             // Taking upper bound job-no from user
792             System.out.println("Enter the upper bound of the cut job-no.");
793             upper_b1 = myScan.nextInt();
794
795             // Executing procedure for Query 14
796             final String Sql14 = "EXEC DeleteCutJob @Lower_b = '"+lower_b1+"', @Upper_b = '"+upper_b1+"'";;
797
798             final Statement statement14 = connection.createStatement();
799             statement14.executeUpdate(Sql14);
800
801             System.out.println("Deleted all cut jobs in the given range of Job-No.");
802             System.out.println("=====");
803
804         } catch (Exception e) {
805             System.out.println("Error!. Returning to main menu");
806         }
807
808         break;
809

```

## 15. Change the color of a given paint job (1/week).

```

---  

810     case 15:
811         // try and catch are used to not terminate loop in case of error.
812         try {
813
814             // Declaring Variables
815             String Color;
816             int Job_No3;
817
818             // Taking job-no from user
819             System.out.println("Enter the paint Job-No.");
820             Job_No3 = myScan.nextInt();
821
822             // Taking color from user
823             System.out.println("Enter the new color for the job-no.");
824             Color = myScan.next();
825
826             // Executing procedure for Query 15
827             final String Sql15 = "EXEC ChangePaintColor @Job_No = '"+Job_No3+"', @Color = '"+Color+"'";;
828
829             final Statement statement15 = connection.createStatement();
830             statement15.executeUpdate(Sql15);
831
832             System.out.println("Changed the color of a given paint job.");
833             System.out.println("=====");
834
835         } catch (Exception e) {
836             System.out.println("Error!. Returning to main menu");
837         }
838
839         break;
840

```

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

```

840     case 16:
841         // try and catch are used to not terminate loop in case of error.
842         try {
843             // Declaring Variables
844             String file_name, line;
845
846             // Taking Input file name from user
847             System.out.println("Enter the file-name to Import data.");
848             file_name = myScan.next();
849
850             // Creating new File object
851             File file = new File(file_name);
852
853             // Creating new Scanner Object
854             Scanner sc = new Scanner(file);
855
856             // While loop to read all lines in the input file
857             while(sc.hasNextLine()) {
858                 line = sc.nextLine();
859
860                 // Dividing line to parts separated by Delimiter ","
861                 String[] parts = line.split(",");
862
863
864                 String Customer_Name= parts[0];
865                 String C_address = parts[1];
866                 int Category = Integer.parseInt(parts[2]);
867
868                 // Execute procedure for Query 16
869                 final String Sql16 = "EXEC InsertCustomerInfo @Customer_Name = '"+Customer_Name+"', @C_address = '"+C_address+"', @Category = '"+Category+"'";
870
871                 final Statement statement16 = connection.createStatement();
872                 statement16.executeUpdate(Sql16);
873             }
874             sc.close();
875         }
876
877
878         catch (Exception e) {
879             System.out.print("Error!. Returning to main menu: ");
880         }
881

```

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

```

384     case 17:
385         // try and catch are used to not terminate loop in case of error.
386         try {
387
388             // Declaring Variables
389             String File_name1, Lower_b2, Upper_b2;
390
391             // Enter the filename to output result
392             System.out.println("Enter the file-name to Export data.");
393             File_name1 = myScan.next();
394
395             // Taking lower bound of category from user
396             System.out.println("Enter the lower bound of category.");
397             Lower_b2 = myScan.next();
398
399             // Taking upper bound of category from user
400             System.out.println("Enter the upper bound of category.");
401             Upper_b2 = myScan.next();
402
403             // Creating new file writer Object
404             FileWriter fw = new FileWriter(File_name1);
405
406             // Executing Procedure(for Query 13) to get output
407             final String Sql17 = "EXEC RetrieveCustomerByCategory1- @Lower_b2 = '"+Lower_b2+"', @Upper_b2 = '"+Upper_b2+"' ";
408
409
410             try (final Statement statement17 = connection.createStatement();
411                  final ResultSet resultSet4 = statement17.executeQuery(Sql17)) {
412
413                 //System.out.println("The customers in the given range of category are");
414                 while (resultSet4.next()) {
415
416                     //fw.write(String.format("%s,%s", resultSet4.getString(1), resultSet4.getString(2)));
417                     fw.write(resultSet4.getString(1) + "," + resultSet4.getString(2) + "\n");
418                 }
419
420                 fw.close();
421                 } catch (SQLException e) {
422                     e.getCause().getMessage();
423                 }
424
425             } catch (Exception e) {
426                 System.out.println("Error!. Returning to main menu");
427             }
428             break;
429

```

18. Quit

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

## Task 6: execution of Queries

```
=====  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====  
1  
Enter the Customer Name  
Lisa  
Enter the Customer Address  
Boston  
Enter the Customer Category  
1  
Customer details inserted successfully.  
=====
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT
```

---

```
1  
Enter the Customer Name  
John  
Enter the Customer Address  
Chicago  
Enter the Customer Category  
2  
Customer details inserted successfully.
```

---

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT
```

---

```
1  
Enter the Customer Name  
Rita  
Enter the Customer Address  
Oklahoma  
Enter the Customer Category  
4  
Customer details inserted successfully.
```

---

```

Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
1
Enter the Customer Name
Ben
Enter the Customer Address
Michigan
Enter the Customer Category
6
Customer details inserted successfully.

```

customerInfo Table: -

Results    Messages

---

Search to filter items...

Customer_Name	C_address	Category
Ben	Michigan	6
John	Chicago	2
Lisa	Boston	1
Rita	Oklahoma	4

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers within a range of category
14. Delete all cut-jobs within a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record within a range of category.
18. QUIT
=====
```

**2**

Enter the Department ID

depth5

Enter the Department Data

ghte2

Department details inserted successfully.

### Department Table: -

Department_ID	Department_Data
Dept1	CS1
Dept2	TB2
Dept3	Phy4
depth5	ghte2

```
-----  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
3  
Enter Assembly ID  
Assbl1  
Enter Date Ordered in yyyy-mm-dd format  
1999-12-23  
Enter Assembly Details  
Assembly23  
Enter the Customer Name  
Lisa  
Assembly details inserted successfully.  
=====
```

```
-----  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
3  
Enter Assembly ID  
Assbl2  
Enter Date Ordered in yyyy-mm-dd format  
2000-03-12  
Enter Assembly Details  
Assembly2  
Enter the Customer Name  
John  
Assembly details inserted successfully.  
=====
```

```
Choose one Option:
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====  
3  
Enter Assembly ID  
Assbl3  
Enter Date Ordered in yyyy-mm-dd format  
1898-04-12  
Enter Assembly Details  
Assembl67  
Enter the Customer Name  
Ben  
Assembly details inserted successfully.  
=====  
Choose one Option:
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====  
3  
Enter Assembly ID  
Assbl4  
Enter Date Ordered in yyyy-mm-dd format  
2023-11-12  
Enter Assembly Details  
Assembl45  
Enter the Customer Name  
Rita  
Assembly details inserted successfully.
```

```

Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
4
Enter Process ID
Proc2
Enter Process Data
ProcessT2
Enter Department ID
Dept2
Process details inserted successfully.
=====
Choose one of the following type of process:
1.Fit
2. Paint
3.Cut

2
Enter paint type
Paint2
Enter paint method
PaintMachine
Process_Paint details inserted successfully.

```

```

<terminated> Performed_Questions [Java Application] /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java (Nov 16, 2021)
Choose one option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
4
Enter Process ID
Proc1
Enter Process Data
ProcessT1
Enter Department ID
Dept1
Process details inserted successfully.
=====
Choose one of the following type of process:
1.Fit
2. Paint
3.Cut

1
Enter fit type
Fit1
Process_Fit details inserted successfully.
=====
```

```

Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whithin a range of category
14. Delete all cut-jobs whithin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whithin a range of category.
18. QUIT
=====
4
Enter Process ID
Proc3
Enter Process Data
ProcessT3
Enter Department ID
Dept3
Process details inserted successfully.
=====
Choose one of the following type of process:
1.Fit
2. Paint
3.Cut
3
Enter cutting type
CutMachine
Enter machine type
Machine1
Process_Cut record inserted successfully.
=====

```

## Process1 Table

Results    Messages

Search to filter items...		
Process_ID	Process_Data	Department_ID
Proc1	ProcessT1	Dept1
Proc2	ProcessT2	Dept2
Proc3	ProcessT3	Dept3

```
|Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers within a range of category  
14. Delete all cut-jobs within a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record within a range of category.  
18. QUIT  
=====
```

```
5  
Enter account number
```

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

```
1  
Enter account details
```

```
12  
Enter date account established
```

```
2010-12-23
```

```
Enter Department Number of the account
```

```
Dept1
```

```
Department_account record inserted successfully.
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers within a range of category
14. Delete all cut-jobs within a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record within a range of category.
18. QUIT
=====
```

5

Enter account number

45

Choose one of the following type of account:

1. Department Account.
2. Assembly Account
3. Process Account

1

Enter account details

456

Enter date account established

2001-01-27

Enter Department Number of the account

Dept2

Department\_account record inserted successfully.

## Department\_acc Table

Results    Messages

Search to filter items...

Account_No	Established_Date	details_2	Department_ID
1	2010-12-23T00:00:00.0000000	12	Dept1
45	2001-01-27T00:00:00.0000000	456	Dept2

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers within a range of category
14. Delete all cut-jobs within a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record within a range of category.
18. QUIT
=====
```

```
5
Enter account number
```

```
56
Choose one of the following type of account:
```

- 1. Department Account.
- 2. Assembly Account
- 3. Process Account

```
2
Enter account details
```

```
657
Enter date account established
```

```
2009-07-10
Enter Assembly id of the account
```

```
Assbl1
Assembly_acc record inserted successfully.
=====
C:\Windows\system32>
```

```
-----  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
5  
Enter account number  
79
```

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

```
2  
Enter account details  
98  
Enter date account established  
2000-10-23  
Enter Assembly id of the account  
Assbl2  
Assembly_acc record inserted successfully.  
=====
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT
```

---

```
5
```

```
Enter account number
```

```
6
```

```
Choose one of the following type of account:
```

- 1. Department Account.
- 2. Assembly Account
- 3.Process Account

```
3
```

```
Enter account details
```

```
78
```

```
Enter date account established
```

```
2009-05-12
```

```
Enter Process id of the account
```

```
Proc1
```

```
Process_acc record inserted successfully.
```

---

```
|Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
5  
Enter account number  
8
```

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

```
3  
Enter account details  
987  
Enter date account established  
1999-02-12  
Enter Process id of the account  
Proc2  
Process_acc record inserted successfully.  
=====
```

```
=====  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
6  
Enter Job-No for a new job  
2  
Enter assembly-ID for the job  
Assbl2  
Enter process-ID for the job  
Proc2  
Enter the commenced date for the job  
2022-12-23  
Job record inserted successfully.  
=====
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whithin a range of category
14. Delete all cut-jobs whithin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whithin a range of category.
18. QUIT
=====
```

6

Enter Job-No for a new job

7

Enter assembly-ID for the job

Assbl1

Enter process-ID for the job

Proc1

Enter the commenced date for the job

2023-12-24

Job record inserted successfully.

=====
Choose one Option:

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whithin a range of category
14. Delete all cut-jobs whithin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whithin a range of category.
18. QUIT
=====
```

6

Enter Job-No for a new job

4

Enter assembly-ID for the job

Assbl3

Enter process-ID for the job

Proc1

Enter the commenced date for the job

2021-06-23

Job record inserted successfully.

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
7
Enter Job-No for the completed job
4
Enter job completed date in yyyy-mm-dd format
2023-06-21
Job record Updated successfully.
=====
```

```
Choose one of the following type of job:
1. Fit Job.
2. Paint Job
3.Cut Job

2
Enter the paint color.
Blue
Enter the paint job labor time in HH:MM:SS format.
06:20:00
Enter the volume of paint.
1
Paint Job record inserted successfully.
=====
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
7
```

```
Enter Job-No for the completed job
```

```
7
```

```
Enter job completed date in yyyy-mm-dd format
```

```
2023-12-01
```

```
Job record Updated successfully.
```

```
=====
Choose one of the following type of job:
```

- 1. Fit Job.
- 2. Paint Job
- 3.Cut Job

```
1
```

```
Enter the fit labor time fo Job in HH:MM:SS format.
```

```
03:30:00
```

```
Fit Job record inserted successfully.
```

Choose one Option:

1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT

---

7

Enter Job-No for the completed job

2

Enter job completed date in yyyy-mm-dd format

2023-12-12

Job record Updated successfully.

---

Choose one of the following type of job:

1. Fit Job.
2. Paint Job
- 3.Cut Job

3

Enter the Job Machine Type.

machine1

Enter the cut job machine time in HH:MM:SS format.

02:00:00

Enter the material used.

M1

Enter the cut job labor time in HH:MM:SS format.

03:00:00

Cut Job record inserted successfully.

---

## Job1 Table

[Results](#) [Messages](#)

Search to filter items...

Job_No	Date_job_commenced	Date_job_completed	Assembly_ID	Process_ID
2	2022-12-23T00:00:00.0000000	2023-12-12T00:00:00.0000000	Assbl2	Proc2
4	2021-06-23T00:00:00.0000000	2023-06-21T00:00:00.0000000	Assbl3	Proc1
7	2023-12-24T00:00:00.0000000	2023-12-01T00:00:00.0000000	Assbl1	Proc1

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
8
Enter the Transaction Number.
1
Enter the cost of the transaction.
300
Enter the Job number related to transaction.
2
Transaction record inserted and related accounts updated successfully.
=====
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
8
Enter the Transaction Number.
2
Enter the cost of the transaction.
200
Enter the Job number related to transaction.
4
Transaction record inserted and related accounts updated successfully.
=====
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whithin a range of category  
14. Delete all cut-jobs whithin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whithin a range of category.  
18. QUIT  
=====
```

```
9  
Enter Assembly Id to retrieve the cost.  
Assbl1  
Cost incurred on Assembly-ID Assbl1:  
657.000000
```

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whithin a range of category  
14. Delete all cut-jobs whithin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whithin a range of category.  
18. QUIT  
=====
```

```
10  
Enter Department Number to get the total labor time.  
Dept1  
Enter Job completed date to get the total labor time.  
2023-12-12  
Total labor-time in minutes for department number Dept1 and date job completed 2023-12-12:  
0  
Choose one Option:
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whithin a range of category
14. Delete all cut-jobs whithin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whithin a range of category.
18. QUIT
=====
```

```
11
Enter the Assembly Id to retrieve the processes.
```

```
Assbl1
The processes through which Assembly ID Assbl1 passed so far:
2023-12-24 | Proc1 | Dept1
=====
```

```
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whithin a range of category
14. Delete all cut-jobs whithin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whithin a range of category.
18. QUIT
=====
```

```
12
Enter the date job is completed.
```

```
2023-12-12
Enter the department to retrieve the jobs.
```

```
Dept2
The fit jobs completed on date 2023-12-12 in department Dept2:
The Paint jobs completed on date 2023-12-12 in department Dept2:
The Cut jobs completed on date 2023-12-12 in department Dept2:
2 | Assbl2 | machine1 | 02:00:00.0000000 | M1 | 03:00:00.0000000
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
13
Enter the lower bound of the category.
```

```
2
Enter the upper bound of the category.
```

```
4
The customers in the given range of category are
```

```
John | 2
Rita | 4
^   ^ ..
```

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====
```

```
14
Enter the lower bound of the cut job-no.
```

```
3
Enter the upper bound of the cut job-no.
```

```
7
Deleted all cut jobs in the given range of Job-No.
=====
```

```
=====
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT
```

---

15

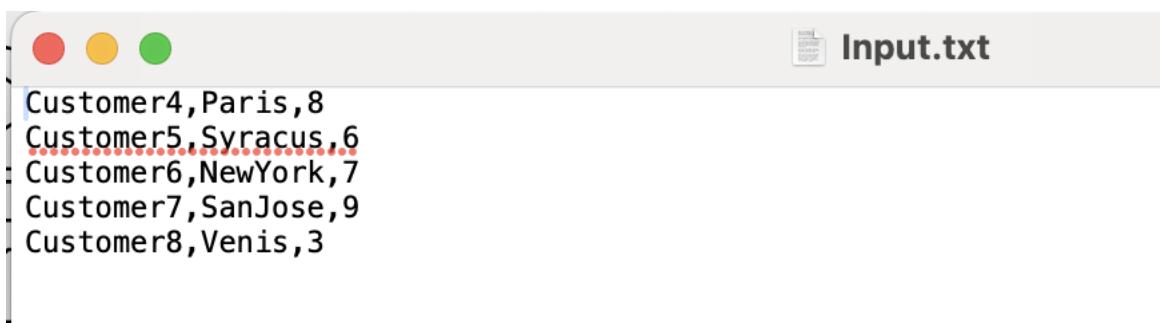
Enter the paint Job-No.

4

Enter the new color for the job-no.

Red

Changed the color of a qiven paint job.



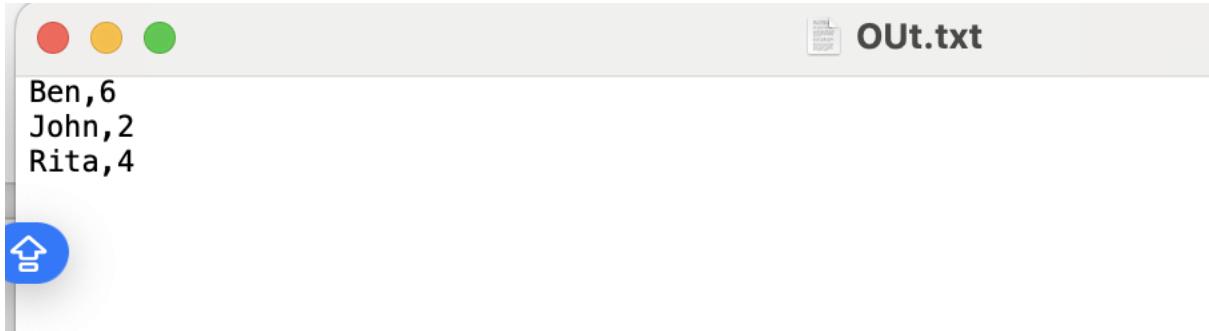
```

===== CONNECTION =====
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====

16
Enter the file-name to Import data.
Input.txt
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====

17
Enter the file-name to Export data.
OUT.txt
Enter the lower bound of category.
2
Enter the upper bound of category.
6

```



Results    Messages

Search to filter items...

Customer_Name	C_address	Category
Ben	Michigan	6
Customer4	Paris	8
Customer5	Syracus	6
Customer6	NewYork	7
Customer7	SanJose	9
Customer8	Venis	3
John	Chicago	2
Lisa	Boston	1
Rita	Oklahoma	4

=====

Choose one Option:

1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT

=====

18

You choose to Quit!

**Check error on Primary key**

```
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
1  
Enter the Customer Name  
Lisa  
Enter the Customer Address  
Boston  
Enter the Customer Category  
5  
Error!. Returning to main menu
```

### Error on data format

```
=====  
Choose one Option:  
1. Enter a new Customer  
2. Enter a new Department  
3. Enter a new assembly  
4. Enter a new process and information related to type  
5. Create a new account and associate it with the one of the 3 type  
6. Enter a new Job  
7. Enter the date job is completed and the type of job  
8. Enter a transaction details and update all accounts  
9. Retrieve the cost for an assembly-id  
10. Retrieve the total labor time,a department ID for jobs completed during a given date  
11. Retrieve the processes through which a given assembly-id has passed with the department ID  
12. Retrieve all jobs completed during the given date by a given department  
13. Retrieve the customers whitin a range of category  
14. Delete all cut-jobs whitin a range Job-No.  
15. Change the color of paint job.  
16. Import new customers detail from a data file.  
17. Retrieve and export customers record whitin a range of category.  
18. QUIT  
=====
```

```
7  
Enter Job-No for the completed job  
90-78-12  
Error!. Returning to main menu  
Choose one Option:
```

### Error on foreign key

```
=====
Choose one Option:
1. Enter a new Customer
2. Enter a new Department
3. Enter a new assembly
4. Enter a new process and information related to type
5. Create a new account and associate it with the one of the 3 type
6. Enter a new Job
7. Enter the date job is completed and the type of job
8. Enter a transaction details and update all accounts
9. Retrieve the cost for an assembly-id
10. Retrieve the total labor time,a department ID for jobs completed during a given date
11. Retrieve the processes through which a given assembly-id has passed with the department ID
12. Retrieve all jobs completed during the given date by a given department
13. Retrieve the customers whitin a range of category
14. Delete all cut-jobs whitin a range Job-No.
15. Change the color of paint job.
16. Import new customers detail from a data file.
17. Retrieve and export customers record whitin a range of category.
18. QUIT
=====

4
Enter Process ID
Proc5
Enter Process Data
Process6
Enter Department ID
90
Error!. Returning to main menu
Choose one Option.
```

**Task 7. (23 points):** Write a Web database application using Azure SQL Database and JSP which provides the Web pages for query 1 and query 12.

#### Handler Code:

```
package Final_Project;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class DataHandlerP {

    private Connection conn;
    // Azure SQL connection credentials
    private String server = "tah0001.database.windows.net";
    private String database = "cs_dsa_4513_F23";
    private String username = "tah0001";
    private String password = "*****";
```

```

// Resulting connection string
final private String url =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trustServerCertif
icate=false;hostNameInCertificate=*.database.windows.net;loginTimeout=30;",
server, database, username, password);

// Initialize and save the database connection
private void getDBConnection() throws SQLException {
if (conn != null) {
return;
}
this.conn = DriverManager.getConnection(url);
}

// Return the result of selecting everything from the Customer table
public ResultSet getAllCustomers() throws SQLException {
getDBConnection(); // Prepare the database connection
// Prepare the SQL statement
final String sqlQuery = "SELECT * FROM CustomerInfo";
final PreparedStatement stmt = conn.prepareStatement(sqlQuery);

// Execute the query
return stmt.executeQuery();
}

// Return the result of selecting Customers with their Category in the given range from Customer table
public ResultSet retrieveCustomers(int Lower_b, int Upper_b) throws SQLException {
getDBConnection(); // Prepare the database connection
// Prepare the SQL statement
final String sqlQuery = "SELECT Customer_Name, Category FROM customerInfo" +
" WHERE Category >= ? AND Category <= ? ORDER BY 1 ";
final PreparedStatement stmt = conn.prepareStatement(sqlQuery);

// Replace the '?' in the above statement with the given attribute values
stmt.setInt(1, Lower_b);
stmt.setInt(2, Upper_b);

// Execute the query
return stmt.executeQuery();
}

// Inserts a record into the CustomerInfo table with the given attribute values
public boolean addCustomer(String Customer_Name, String C_address, int Category)

```

```

throws SQLException {getDBConnection(); // Prepare the database connection
                    // Prepare the SQL statement
final String sqlQuery ="INSERT INTO customerInfo " +"(Customer_Name, C_address, Category) "
+"VALUES " +"(?, ?, ?);"
final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
// Replace the '?' in the above statement with the given attribute values
stmt.setString(1, Customer_Name);
stmt.setString(2, C_address);
stmt.setInt(3, Category);
// Execute the query, if only one record is updated, then we indicate success by returning true
return stmt.executeUpdate() == 1;
}
}

```

### Add new customer Code

```

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

```

```

String Name = request.getParameter("Customer_Name");
String Address = request.getParameter("C_address");
String category = request.getParameter("Category");
/*
 * If the user hasn't filled out all the Customer_name, address and category. This is very simple
checking.
*/
if (Name.equals("") || Address.equals("") || category.equals("")) {
    response.sendRedirect("Add_new_customer_form.jsp");
} else {
    int Category1= Integer.parseInt(category);
    // Now perform the query with the data from the form.
    boolean success = handler.addCustomer(Name,Address, Category1);
    if (!success) { // Something went wrong
        %>
        <h2>There was a problem inserting the course</h2>
        <%
    } else { // Confirm success to the user
        %>
        <h2>The Customer Details:</h2>
    }
    Page 23 of 23
    <ul>
        <li>Customer Name: <%=Name%></li>
        <li>Address: <%=Address%></li>
        <li>Category: <%=category%></li>
    </ul>
    <h2>Was successfully inserted.</h2>
    <a href="get_all_customers.jsp">See all Customers.</a>
    <%
}
}

```

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

### Add new 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 customer record.

Upon form submission, add\_new\_customer.jsp file will be invoked.

-->

```
<form method= "POST" action= "Add_New_Customer.jsp">  
<!-- The form organized in an HTML table for better clarity. -->  
<table border= 1>  
<tr>  
<th colspan= "2">Enter the Customer Data:</th>  
</tr>  
<tr>  
<td>Customer Name:</td>  
<td><div style="text-align: center;">  
<input type= text name= "Customer_Name">  
</div></td>  
</tr>  
<tr>
```

```

<td>Customer Address:</td>
<td><div style="text-align: center;">
<input type="text" name= "C_address">
</div></td>
</tr>
<tr>
<td>Category:</td>
<td><div style="text-align: center;">
<input type="text" name= "Category">
</div></td>
</tr>
<tr>
<td><div style="text-align: center;">
<input type="reset" value= "Clear">
</div></td>
<td><div style="text-align: center;">
<input type="submit" value= "Insert">
</div></td>
</tr>
</table>
</form>
</body>
</html>

```

### Get all Customer Code

```

<%@ page language= "java" contentType= "text/html; charset=UTF-8"
    pageEncoding= "UTF-8"%>
<!DOCTYPE html>
<html>
    <head>

```

```

<meta charset="UTF-8">
    <title>Customers</title>
</head>
<body>
<%@page import= "Final_Project.DataHandlerP"%>
<%@page import= "java.sql.ResultSet"%>
<%
    // We instantiate the data handler here, and get all the customers from the database
    final DataHandlerP handler1 = new DataHandlerP();
    final ResultSet customers = handler1.getAllCustomers();

%>
<!-- The table for displaying all the Customer records -->
<table border="1">
    <tr> <!-- The table headers row -->
        <td align="center">
            <h4>Customer</h4>
        </td>
        <td align="center">
            <h4>Address</h4>
        </td>
        <td align="center">
            <h4>Category</h4>
        </td>
    </tr>
    <%
        while(customers.next()) { // For each Customer record returned...
            // Extract the attribute values for every row returned
            final String Customer_Name =
customers.getString("Customer_Name");
            final String C_address = customers.getString("C_address");
            final String Category = customers.getString("Category");
            out.println("<tr>"); // Start printing out the new table row

```

```

        out.println( // Print each attribute value
            "<td align=\"center\">" + Customer_Name +
            "</td><td align=\"center\"> " + C_address+
            "</td><td align=\"center\"> " + Category + "</td>");
        out.println("</tr>");
    }
%>
</table>
</body>
</html>

```

### Retrieve Customer Code

```

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

```

```

String upper_b = request.getParameter("Upper_b");

if (lower_b.equals("")||upper_b.equals("")) {
    response.sendRedirect("retrieve_customer_form.jsp");
} else {
    int Lower_b1 = Integer.parseInt(lower_b);
    int Upper_b1 = Integer.parseInt(upper_b);
    // Now perform the query with the data from the form.
    final ResultSet customers = handler.retrieveCustomers(Lower_b1, Upper_b1);

%>
<!-- The table for displaying all the Customer records -->
<table border="1">
    <tr> <!-- The table headers row -->
        <td align="center">
            <h4>Customer</h4>
        </td>
        <td align="center">
            <h4>category</h4>
        </td>
    <%
        while(customers.next()) { // For each Customer record returned...
            // Extract the attribute values for every row returned
            final String Customer_Name =
            customers.getString("Customer_Name");
            final String Category = customers.getString("Category");
            out.println("<tr>"); // Start printing out the new table row
            out.println( // Print each attribute value
                "<td align=\"center\">" + Customer_Name +
                "</td><td align=\"center\"> " + Category + "</td>");
            out.println("</tr>");
        }
    <%>

```

```

        }
        %>
    </table>
</body>
</html>
```

### Retrieve Customer Form Code

```

DOCTYPE html>

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

```
<tr>
<td>Upper Bound:</td>
<td><div style="text-align: center;">
<input type=text name= "Upper_b">
</div></td>
</tr>
<tr>
<td><div style="text-align: center;">
<input type=reset value= Clear>
</div></td>
<td><div style="text-align: center;">
<input type=submit value= Insert>
</div></td>
</tr>
</table>
</form>
</body>
</html>
```

### Screenshot

# Add Customer

Enter the Customer Data:	
Customer Name:	Remi
Customer Address:	Tunisia
Category:	8
<input type="button" value="Clear"/>	<input type="button" value="Insert"/>

---

## The Customer Details:

Page 23 of 23

- Customer Name: Remi
- Address: Tunisia
- Category: 8

**Was successfully inserted.**

[See all Customers.](#)

---

<b>Customer</b>	<b>Address</b>	<b>Category</b>
Ben	Michigan	6
Customer4	Paris	8
Customer5	Syracus	6
Customer6	New York	7
Customer7	San Jose	9
Customer8	Venis	3
John	Chicago	2
Lisa	Boston	1
Remi	Tunisia	8
Rita	Oklahoma	4

---

[Retrieve Customer details](#)

# Give Customers Range

<b>Enter the Category Range:</b>	
Lower Bound:	2
Upper Bound:	8
<input type="button" value="Clear"/>	<input type="button" value="Insert"/>

---

Customer	category
Ben	6
Customer4	8
Customer5	6
Customer6	7
Customer8	3
John	2
Remi	8
Rita	4

---