

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Course Name: Database Management Systems

Course Number: CS/DSA – 4513

Section Number: 001

Semester and Year: Fall 2023

Instructor Name: Dr. Le Gruenwald

Author Name: Vasireddy Ujwala

Author Id: 113634633

Email-id: Ujwala.Vasireddy-1@ou.edu

Title: A JOB-SHOP ACCOUNTING SYSTEM

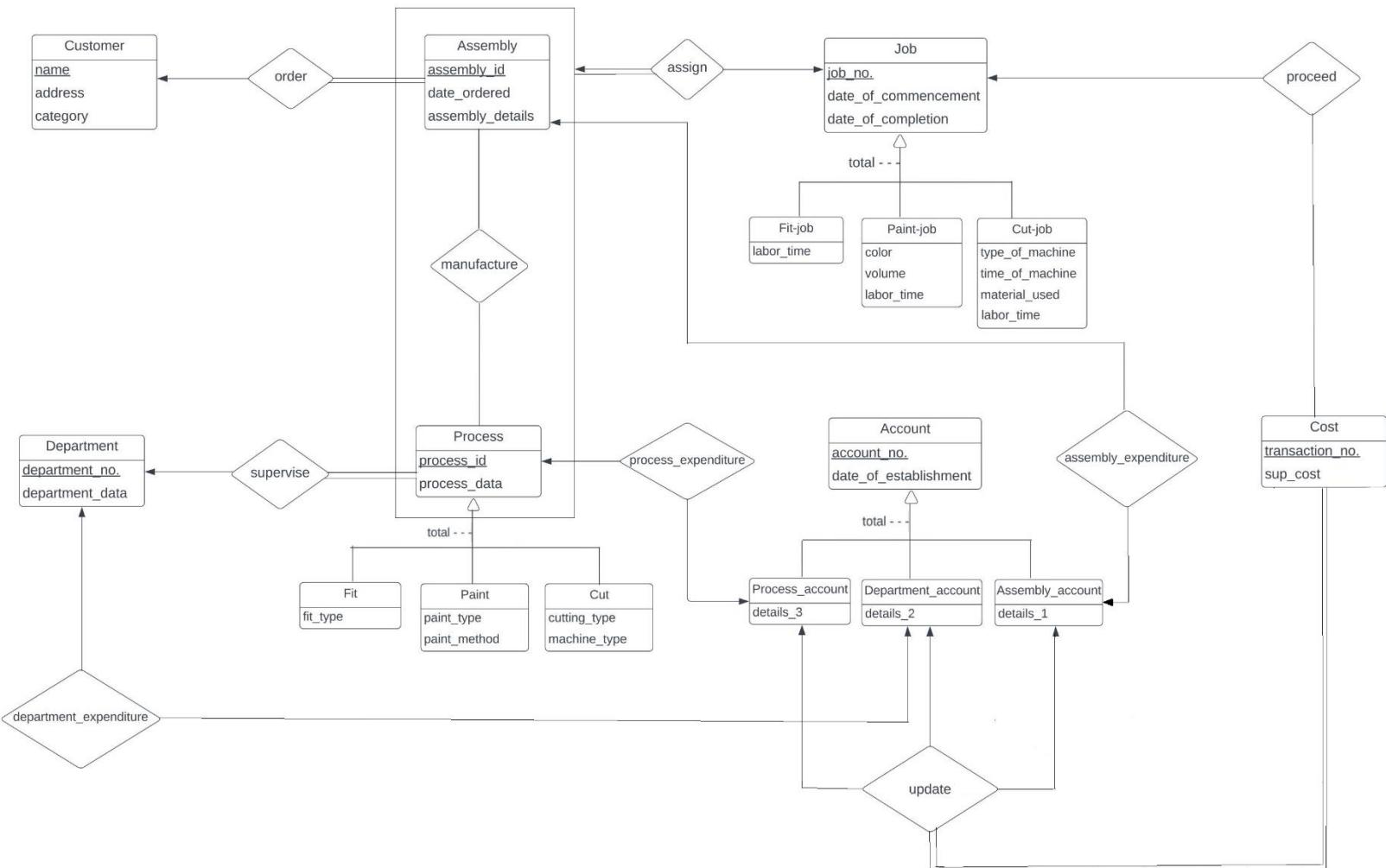
UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Task 1.	ER Diagram	3
Task 2.	Relational Database Schemas	4
Task 3.		
3.1	Discussion of storage structures for tables	5-6
3.2	Discussion of storage structures for tables (Azure SQL Database)	7-8
Task 4.	SQL statements and screenshots showing the creation of tables in Azure SQL Database	9-15
Task 5.		
5.1	SQL statements and Transact SQL stored procedures Implementing all queries (1-15 and error checking)	16-37
5.2	The Java source program and screenshots showing its successful compilation	38-65
Task 6.	Java program Execution	
6.1	Screenshots showing testing of query 1	66-70
6.2	Screenshots showing testing of query 2	70-75
6.3	Screenshots showing testing of query 3	76-85
6.4	Screenshots showing testing of query 4	86-93
6.5	Screenshots showing testing of query 5	94-96
6.6	Screenshots showing testing of query 6	97-100
6.7	Screenshots showing testing of query 7	100-102
6.8	Screenshots showing testing of query 8	103-105
6.9	Screenshots showing testing of query 9	105-108
6.10	Screenshots showing testing of query 10	109-110
6.11	Screenshots showing testing of query 11	111
6.12	Screenshots showing testing of query 12	112
6.13	Screenshots showing testing of query 13	113
6.14	Screenshots showing testing of query 14	114
6.15	Screenshots showing the testing of the import and export options	115-117
6.16	Screenshots showing the testing of quit option	118
6.17	Screenshots showing three different types of errors	119
Task 7.	Web database application and its execution	
7.1	Web database application source program and screenshots showing its successful compilation	120-129
7.2	Screenshots showing the testing of the Web database application	130-134

Task 1: ER DIAGRAM



Task 2: RELATIONAL DATABASE SCHEMAS

Customer (cname, caddress, category)
Assembly (assembly_id, date_ordered, assembly_details, cname)
Department (department_no, department_data)
Process (process_id, process_data, department_no)
Fit_process (process_id, fit_type)
Paint_process (process_id, paint_type, paint_method)
Cut_process (process_id, cutting_type, machine_type)
Job (job_no, date_of_commencement, date_of_completion)
Fit_job (job_no, fit_labor_time)
Paint_job (job_no, color, volume, paint_labor_time)
Cut_job (job_no, type_of_machine, time_of_machine, material_used, cut_labor_time)
manufacture (assembly_id, process_id, job_no)
Assembly_account (assembly_account_no, date_of_establishment, details_1)
Department_account (department_account_no, date_of_establishment, details_2)
Process_account (process_account_no, date_of_establishment, details_3)
assembly_expenditure (assembly_account_no, assembly_id)
department_expenditure (department_account_no, department_no)
process_expenditure (process_account_no, process_eid)
Cost (transaction_no, sup_cost, assembly_account_no, department_account_no, process_account_no)
proceed (transaction_no, job_no)

TASK 3.1: DISCUSSION OF STORAGE STRUCTURES FOR TABLES

Table Name	Query# Type	Search Key	Query Frequency	Selected file organization	Justifications
Customer	(1) Insertion		30/day	Index Sequential File on search key category	Given that range searches are prevalent in category, the Index Sequential file is the preferred choice for conducting such searches. The Index Sequential file employs a combination of a data file and an index, enabling quick access to specific ranges of data.
	(12) Range Search	category	100/day		
Assembly	(4) Insertion		40/day	Heap File	Heap files are considered advantageous for insertion operations due to their simplicity and efficiency in accommodating dynamic data growth.
Department	(2) Insertion		infrequent	Dynamic Hashing with a hash key job_no	Dynamic hashing is chosen because random search is more frequent and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(10) Random Search	job_no	20/day		
Process	(3) Insertion		infrequent	Dynamic Hashing with a hash key process_id	Dynamic hashing is chosen because random search is more frequent on process_id and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(10) Random Search	department_no	20/day		
	(11) Random Search	process_id	100/day		
Fit_process	(3) Insertion		infrequent	Heap File	Heap files are considered advantageous for insertion operations due to their simplicity and efficiency in accommodating dynamic data growth.
Paint_process				Heap File	Heap files are considered advantageous for insertion operations due to their simplicity and efficiency in
Cut_process	(3) Insertion		infrequent	Heap File	Heap files are considered advantageous for insertion operations due to their simplicity and efficiency in accommodating dynamic data growth.
Job	(6) Insertion		50/day	Dynamic Hashing with a hash key job_no	Dynamic hashing is chosen because random search is more frequent on job_no and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(7) Random Search(update)	job_no	50/day		
	(10) Random Search	date_of_completion	20/day		
	(11) Random Search	job_no	100/day		
Fit_job	(6) Insertion		50/day	Dynamic Hashing with a hash key job_no	Dynamic hashing may be preferable over heap files in scenarios where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a
	(7) Random Search (update)	job_no	50/day		
Paint_job	(6) Insertion		50/day	Dynamic Hashing with a hash key job_no	Dynamic hashing may be preferable over heap files in scenarios where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a specific key.
	(7) Random Search (update)	job_no	50/day		
	(14) Random Search (update)	job_no	1/week		
Cut_job	(6) Insertion		50/day	Dynamic Hashing with a hash key job_no	We don't consider Index Sequential file or a sparse index file here. Dynamic hashing may be preferable over heap files in scenarios where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a specific key.
	(7) Random Search (update)	job_no	50/day		
	(13) Range Search	job_no	1/month		
	(13) Deletion	job_no	1/month		

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Table Name	Query# Type	Search Key	Query Frequency	Selected file organization	Justifications
Manufacture	(4) Insertion		40/day	Dynamic Hashing with a hash key assembly_id	Dynamic hashing may be preferable where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a specific key. The search key assembly_id would be really helpful for update operation as well.
	(6) Random Search (update)	job_no	50/day		
	(10) Random Search	process_id	20/day		
	(11) Random Search	assembly_id	100/day		
Assembly_account	(5) Insertion		10/day	Dynamic Hashing with a hash key assembly_account_no	Dynamic hashing is chosen because random search is more frequent on assembly_account_no and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(8) Random Search (update)	assembly_account_no	50/day		
Department_account	(5) Insertion		10/day	Dynamic Hashing with a hash key department_account_no	Dynamic hashing is chosen because random search is more frequent on department_account_no and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(8) Random Search (update)	department_account	50/day		
Process_account	(5) Insertion		10/day	Dynamic Hashing with a hash key process_account_no	Dynamic hashing is chosen because random search is more frequent on process_account_no and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(8) Random Search (update)	process_account_no	50/day		
Assembly_expenditure	(5) Insertion		10/day	Dynamic Hashing with a hash key assembly_id	Dynamic hashing is chosen because random search is more frequent on assembly_id and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(5) Random Search (update)	assembly_account_no	10/day		
	(9) Random Search	assembly_id	200/day		
Department_expenditure	(5) Insertion		10/day	Dynamic Hashing with a hash key department_account_no	Dynamic hashing may be preferable over heap files in scenarios where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a specific key.
	(5) Random Search (update)	department_account	10/day		
Process_expenditure	(5) Insertion		10/day	Dynamic Hashing with a hash key process_account_no	Dynamic hashing may be preferable over heap files in scenarios where there is a need for adaptability to dynamic changes in data size and a desire for efficient random search operations based on a specific key.
	(5) Random Search (update)	process_account_no	10/day		
Cost	(8) Insertion		50/day	Dynamic Hashing with a hash key assembly_account_no	Dynamic hashing is chosen because random search is more frequent on assembly_account_no and particularly effective for random search with a hash key due to its ability to adapt dynamically to varying amounts of data and distribute it evenly across buckets.
	(9) Random Search	assembly_account_no	200/day		

Task 3.2: STUDY OF RELEVANT DOCUMENTATION

Some other possible indexes I could understand are:

Filtered Indexes:

Relevant for queries involving specific subsets of data. For example, if we frequently query a subset of records based on certain conditions, a filtered index on the relevant columns might improve performance.

Columnstore Indexes:

Suitable for analytical queries that involve aggregating large amounts of data. If we have analytics queries for reporting purposes, a columnstore index on relevant columns can significantly improve query performance.

Memory-Optimized Tables and Indexes:

Suitable for high transaction throughput and low-latency scenarios. If our workload involves frequent insertions and updates, especially for transactional data, memory-optimized tables and indexes can provide improved performance.

Azure SQL Database Hyperscale:

Relevant for very large databases with horizontal scaling requirements. If our dataset grows significantly, Azure SQL Database Hyperscale can offer automatic scaling and support for large volumes of data.

In-Memory OLTP:

Suitable for high-performance transaction processing. If you have queries that require low-latency access to transactional data, especially in scenarios with high concurrency, In-Memory OLTP can be beneficial.

JSON Indexing:

Relevant if your data includes JSON documents and you frequently query JSON data. JSON indexing can improve the performance of queries involving JSON data structures.

Full-Text Indexes:

Suitable for efficient searching of large amounts of text-based data. If you have queries that involve text search operations, full-text indexes can enhance the search performance.

Spatial Indexing:

Relevant if your application deals with spatial data. Spatial indexing can optimize queries related to geographical or geometric data, enhancing the performance of spatial queries.

Auto-Create Statistics and Auto-Update Statistics:

While not indexes, these features are crucial for the query optimizer to generate efficient execution plans. They play a significant role in optimizing query performance by ensuring that the query planner has up-to-date statistics for accurate decision-making.

Bitmapped Index:

In certain scenarios, especially when dealing with categorical data or when queries involve multiple boolean conditions, a bitmapped index can be considered. Bitmapped indexes use bitmaps to represent the presence or absence of values in a column, enabling fast searches for combinations of conditions.

Multilevel Index:

In scenarios where the dataset is large and a single-level index becomes impractical, a multilevel index can be considered. This hierarchical structure allows for more efficient navigation through the index.

Covering Index:

A covering index includes all the columns required to satisfy a query, eliminating the need to access the actual data rows. This can be beneficial for certain queries, especially when dealing with a specific subset of columns.

GiST (Generalized Search Tree):

In databases that support complex data types (e.g., geometric or geographic data), GiST indexing can be considered. GiST allows indexing of complex data structures beyond simple key-value pairs.

BRIN (Block Range INdex):

BRIN indexes are suitable for very large tables with naturally ordered data. They divide the table into blocks and create an index on the minimum and maximum values within each block.

Even though we can't use file structures like "Index Sequential File" and "Heap File" directly in Azure SQL Database, we do have significant influence over how tables are organized and indexed. By strategically setting up clustered indexes, primary keys, and other indexing features like secondary index, we can steer the database engine to work best for the kinds of queries you're running. It's like customizing the filing system to match exactly how you want to look up information.

Since the clustered indexes are by default in Azure, the non-clustered index I included are:

Table	Secondary Index
Customer	category (because of range search on it)
Process	department_no (because of random search on it)
Job	date_of_completion (because of random search on it)
assembly_expenditure	assembly_id (because of random search on it)
manufacture	job_no (because of random search update on it)
manufacture	assembly_id (because of random search on it)

Task 4: SQL statements and screenshots showing the creation of tables in Azure SQL Database

1. Create Customer table:

```
-- Create Customer table
CREATE TABLE Customer (
    cname VARCHAR(20) PRIMARY KEY,
    caddress VARCHAR(100),
    category INT NOT NULL,
    CONSTRAINT category_ck CHECK (category BETWEEN 1 AND 10),
);
```

Creating Index on category:

```
CREATE INDEX customer_category ON Customer(category);
```

Results Messages

Query succeeded: Affected rows: 0

2. Create Assembly table:

```
-- Create Assembly table
CREATE TABLE Assembly (
    assembly_id VARCHAR(20) PRIMARY KEY,
    date_ordered DATE,
    assembly_details VARCHAR(100),
    cname VARCHAR(20) FOREIGN KEY REFERENCES Customer(cname)
);
```

Results Messages

Query succeeded: Affected rows: 0

3. Create Department table:

```
--Create Department Table
CREATE TABLE Department (
    department_no INT PRIMARY KEY,
    department_data VARCHAR(100),
);
```

Results Messages

Query succeeded: Affected rows: 0

4. Create Process table:

```
--Create Process Table
CREATE TABLE Process (
process_id VARCHAR(20) PRIMARY KEY,
process_data VARCHAR(100),
department_no INT,
FOREIGN KEY (department_no) REFERENCES Department(department_no)
);
```

Creating Index on department_no:

```
CREATE INDEX process_department_no ON Process(department_no);
```

Results Messages

Query succeeded: Affected rows: 0

5. Create Fit_process table:

```
--Create Fit_process Table
CREATE TABLE Fit_process (
process_id VARCHAR(20) FOREIGN KEY REFERENCES Process(process_id),
fit_type VARCHAR(20),
PRIMARY KEY (process_id)
);
```

Results Messages

Query succeeded: Affected rows: 0

6. Create Paint_process table:

```
--Create Paint_process Table
CREATE TABLE Paint_process (
process_id VARCHAR(20) FOREIGN KEY REFERENCES Process(process_id),
paint_type VARCHAR(20),
paint_method VARCHAR(20),
PRIMARY KEY (process_id)
);
```

Results Messages

Query succeeded: Affected rows: 0

7. Create Cut_process table:

```
--Create Cut_process Table
CREATE TABLE Cut_process (
    process_id VARCHAR(20) FOREIGN KEY REFERENCES Process(process_id),
    cutting_type VARCHAR(20),
    machine_type VARCHAR(20),
    PRIMARY KEY (process_id)
);
```

Results Messages

Query succeeded: Affected rows: 0

8. Create Job table:

```
--Create Job Table
CREATE TABLE Job (
    job_no INT PRIMARY KEY,
    date_of_commencement DATE,
    date_of_completion DATE
);
```

Creating an index on date_of_completion:

```
CREATE INDEX job_date_of_completion ON Job(date_of_completion);
```

Results Messages

Query succeeded: Affected rows: 0

9. Create Fit_job table:

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

Results Messages

Query succeeded: Affected rows: 0

10. Create Paint_job table:

```
--Create Paint_job Table
CREATE TABLE Paint_job (
| job_no INT FOREIGN KEY REFERENCES Job(job_no),
color VARCHAR(20),
volume FLOAT,
paint_labor_time TIME,
PRIMARY KEY (job_no)
);
```

Results Messages

Query succeeded: Affected rows: 0

11. Create Cut_job table:

```
--Create Cut_job Table
CREATE TABLE Cut_job (
| job_no INT FOREIGN KEY REFERENCES Job(job_no),
job_machine_type VARCHAR(20),
machine_time TIME,
material_used VARCHAR(20),
cut_labor_time TIME,
PRIMARY KEY (job_no)
);
```

Results Messages

Query succeeded: Affected rows: 0

12. Create manufacture table:

```
--Create manufacture Table
CREATE TABLE manufacture (
assembly_id VARCHAR(20) FOREIGN KEY REFERENCES Assembly(assembly_id),
process_id VARCHAR(20) FOREIGN KEY REFERENCES Process(process_id),
job_no INT FOREIGN KEY REFERENCES Job(job_no),
PRIMARY KEY (assembly_id, process_id)
);
```

Creating an index on assembly_id:

```
CREATE INDEX manufacture_assembly_id ON manufacture(assembly_id);
```

Creating an index on job_no:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
CREATE INDEX manufacture_job_no ON manufacture(job_no);
```

Results Messages

Query succeeded: Affected rows: 0

13. Create Assembly_account table:

```
--Create Assembly Account Table
CREATE TABLE Assembly_account (
assembly_account_no INT PRIMARY KEY,
date_of_establishment DATE,
details_1 NUMERIC (10, 2)
);
```

Results Messages

Query succeeded: Affected rows: 0

14. Create Department_account table:

```
--Create Department Account Table
CREATE TABLE Department_account (
department_account_no INT PRIMARY KEY,
date_of_establishment DATE,
details_2 NUMERIC (10, 2)
);
```

Results Messages

Query succeeded: Affected rows: 0

15. Create Process_account table:

```
--Create Process Account Table
CREATE TABLE Process_account (
process_account_no INT PRIMARY KEY,
date_of_establishment DATE,
details_3 NUMERIC (10, 2)
);
```

Results Messages

Query succeeded: Affected rows: 0

16. Create assembly_expenditure table:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
-- Create Assembly Expenditure Table
CREATE TABLE assembly_expenditure (
    assembly_account_no INT,
    assembly_id VARCHAR(20),
    PRIMARY KEY (assembly_account_no),
    FOREIGN KEY (assembly_account_no) REFERENCES Assembly_account(assembly_account_no),
    FOREIGN KEY (assembly_id) REFERENCES Assembly(assembly_id)
);
```

Creating an index on assembly_id:

```
CREATE INDEX assembly_expenditure_assembly_id ON assembly_expenditure(assembly_id);
```

Results Messages

Query succeeded: Affected rows: 0

17. Creating department_expenditure table:

```
-- Create Department Expenditure Table
CREATE TABLE department_expenditure (
    department_account_no INT,
    department_no INT,
    PRIMARY KEY (department_account_no),
    FOREIGN KEY (department_account_no) REFERENCES Department_account(department_account_no),
    FOREIGN KEY (department_no) REFERENCES Department(department_no)
);
```

Results Messages

Query succeeded: Affected rows: 0

18. Creating process_expenditure table:

```
-- Create Process Expenditure Table
CREATE TABLE process_expenditure (
    process_account_no INT,
    process_id VARCHAR(20),
    PRIMARY KEY (process_account_no),
    FOREIGN KEY (process_account_no) REFERENCES Process_account(process_account_no),
    FOREIGN KEY (process_id) REFERENCES Process(process_id)
);
```

Results Messages

Query succeeded: Affected rows: 0

19. Create Cost table:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
-- Create Cost Table
CREATE TABLE Cost (
    transaction_no INT PRIMARY KEY,
    sup_cost NUMERIC(10, 2),
    assembly_account_no INT,
    department_account_no INT,
    process_account_no INT,
    FOREIGN KEY (assembly_account_no) REFERENCES Assembly_account(assembly_account_no),
    FOREIGN KEY (department_account_no) REFERENCES Department_account(department_account_no),
    FOREIGN KEY (process_account_no) REFERENCES Process_account(process_account_no)
);
```

Results Messages

Query succeeded: Affected rows: 0

20. Create proceed table:

```
--Create proceed Table
CREATE TABLE proceed (
    transaction_no INT FOREIGN KEY REFERENCES Cost(transaction_no),
    job_no INT FOREIGN KEY REFERENCES Job(job_no),
    PRIMARY KEY (transaction_no)
);
```

Results Messages

Query succeeded: Affected rows: 0

Task 5:

Task 5.1 SQL statements and Transact SQL stored procedures

1. Enter a new Customer

Enter a new Customer procedure:

```
1  CREATE PROCEDURE EnterNewCustomer
2      @cname VARCHAR(20),
3      @caddress VARCHAR(100),
4      @category INT,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Data validation
11         IF LEN(@cname) > 20
12             BEGIN
13                 -- If the name is too long
14                 SET @errorCode = 50001;
15                 SET @errorMessage = 'Invalid customer name. Name must be up to 20 characters.';
16                 RETURN;
17             END
18
19         IF @cname LIKE '%[0-9!@#$%^&*,.?":{}|<>]%'
20             BEGIN
21                 -- If the name contains special characters, raise an error
22                 SET @errorCode = 50002;
23                 SET @errorMessage = 'Invalid customer name. Name must be without special characters or numbers.';
24                 RETURN;
25             END
26
27         -- Check if the address is null
28         IF LEN(@caddress) = 0
29             BEGIN
30                 -- Check if the address is null
31                 IF LEN(@caddress) = 0
32                     BEGIN
33                         SET @errorCode = 50003;
34                         SET @errorMessage = 'Invalid customer address. Address must be provided.';
35                         RETURN;
36                     END
37                     -- Check if the address is too long
38                     IF LEN(@caddress) > 100
39                         BEGIN
40                             SET @errorCode = 50004;
41                             SET @errorMessage = 'Invalid customer address. Address must be up to 100 characters.';
42                             RETURN;
43                         END
44
45                     IF @category NOT BETWEEN 1 AND 10
46                         BEGIN
47                             -- If the category is in the specified range, raise an error
48                             SET @errorCode = 50005;
49                             SET @errorMessage = 'Invalid customer category. Category must be an integer between 1 and 10.';
50                             RETURN;
51                         END
52
53             -- Data manipulation
54             BEGIN
55                 INSERT INTO Customer (cname, caddress, category)
56                 VALUES (@cname, @caddress, @category);
57             END;
```

```

54      END;
55
56  END TRY
57  BEGIN CATCH
58      -- Handle errors
59      SET @errorCode = ERROR_NUMBER();
60
61      -- Check if the error is related to a primary key or unique constraint violation
62      IF @errorCode = 2627
63      BEGIN
64          SET @errorMessage = 'A customer with the same name or data already exists.';
65      END
66      ELSE
67      BEGIN
68          SET @errorMessage = ERROR_MESSAGE();
69      END
70
71      RETURN;
72  END CATCH;
73 END;

```

2. Enter a new department

Enter a new department procedure:

```

1  CREATE PROCEDURE EnterNewDepartment
2      @department_no INT,
3      @department_data VARCHAR(100),
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(4000) OUTPUT
6  AS
7  BEGIN
8      BEGIN TRY
9          -- Data validation
10         IF @department_no <= 0
11             BEGIN
12                 -- If the department number is not positive, raise an error
13                 SET @errorCode = 50006;
14                 SET @errorMessage = 'Invalid department number. Department number must be a positive integer.';
15                 RETURN;
16             END
17
18         IF LEN(@department_data) > 100
19             BEGIN
20                 -- If the department data is too long, raise an error
21                 SET @errorCode = 50007;
22                 SET @errorMessage = 'Invalid department data. Data must be up to 100 characters.';
23                 RETURN;
24             END
25
26         -- Data manipulation
27         BEGIN
28             INSERT INTO Department (department_no, department_data)

```

```

23    |           RETURN;
24    |       END
25
26    -- Data manipulation
27    BEGIN
28        INSERT INTO Department (department_no, department_data)
29        VALUES (@department_no, @department_data);
30    END;
31
32
33    END TRY
34    BEGIN CATCH
35        -- Handle errors
36        SET @errorCode = ERROR_NUMBER();
37
38        -- Check if the error is related to a unique key violation
39        IF @errorCode = 2627
40        BEGIN
41            SET @errorMessage = 'Department number already exists. Please choose a unique department number.';
42        END
43        ELSE
44        BEGIN
45            SET @errorMessage = ERROR_MESSAGE();
46        END
47
48        RETURN;
49    END CATCH;
50 END;

```

3. Enter a new process_id and its department together with its type and information relevant to the type

Enter new process procedure:

```

1  CREATE PROCEDURE EnterNewProcess
2      @process_id VARCHAR(20),
3      @process_data VARCHAR(100),
4      @department_no INT,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Data validation
11         IF LEN(@process_data) = 0 OR LEN(@process_data) > 100
12         BEGIN
13             SET @errorCode = 50008;
14             SET @errorMessage = 'Invalid process data. Process data must be provided upto 100 characters.';
15             RETURN;
16         END
17
18         IF NOT EXISTS (SELECT 1 FROM Department WHERE department_no = @department_no)
19         BEGIN
20             SET @errorCode = 50009;
21             SET @errorMessage = 'Department does not exist. Please provide a valid department number.';
22             RETURN;
23         END
24
25
26         -- Data manipulation
27         INSERT INTO Process (process_id, process_data, department_no)
28         VALUES (@process_id, @process_data, @department_no);

```

```

26      -- Data manipulation
27      INSERT INTO Process (process_id, process_data, department_no)
28      VALUES (@process_id, @process_data, @department_no);
29
30
31      END TRY
32      BEGIN CATCH
33          -- Handle errors
34          SET @errorCode = ERROR_NUMBER();
35          SET @errorMessage = ERROR_MESSAGE();
36
37          RETURN;
38      END CATCH;
39  END;
40

```

3.1: Enter a fit process procedure:

```

1  CREATE PROCEDURE EnterNewFitProcess
2      @process_id VARCHAR(20),
3      @fit_type VARCHAR(20),
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(4000) OUTPUT
6  AS
7  BEGIN
8      BEGIN TRY
9          -- Data validation
10         IF LEN(@fit_type) = 0 OR LEN(@fit_type) > 20
11             BEGIN
12                 SET @errorCode = 50010;
13                 SET @errorMessage = 'Invalid input. Fit Type cannot be empty and it must be up to 20 characters.';
14                 RETURN;
15             END
16
17         -- Data manipulation
18         INSERT INTO Fit_Process (process_id, fit_type)
19         VALUES (@process_id, @fit_type);
20
21     END TRY
22     BEGIN CATCH
23         -- Handle errors
24         SET @errorCode = ERROR_NUMBER();
25         SET @errorMessage = ERROR_MESSAGE();
26         RETURN;
27     END CATCH;
28

```

3.2: Enter a paint process procedure:

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1  CREATE PROCEDURE EnterNewPaintProcess
2      @process_id VARCHAR(20),
3      @paint_type VARCHAR(20),
4      @paint_method VARCHAR(20),
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Data validation
11         IF LEN(@paint_type) = 0 OR LEN(@paint_type) > 20
12             BEGIN
13                 SET @errorCode = 50011;
14                 SET @errorMessage = 'Invalid input. Paint Type cannot be empty and must be up to 20 characters.';
15                 RETURN;
16             END
17
18         IF LEN(@paint_method) = 0 OR LEN(@paint_method) > 20
19             BEGIN
20                 SET @errorCode = 50012;
21                 SET @errorMessage = 'Invalid input. Paint Method cannot be empty and must be up to 20 characters.';
22                 RETURN;
23             END
24         -- Data manipulation
25         INSERT INTO Paint_Process (process_id, paint_type, paint_method)
26             VALUES (@process_id, @paint_type, @paint_method);
27     END TRY
28     BEGIN CATCH
29         -- Handle errors
30         SET @errorCode = ERROR_NUMBER();
31         SET @errorMessage = ERROR_MESSAGE();
32         RETURN;
33     END CATCH;
34 END;
```

3.3: Enter a cut process procedure:

```
1  CREATE PROCEDURE EnterNewCutProcess
2      @process_id VARCHAR(20),
3      @cutting_type VARCHAR(20),
4      @machine_type VARCHAR(20),
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Data validation
11         IF LEN(@cutting_type) = 0 OR LEN(@cutting_type) > 20
12             BEGIN
13                 SET @errorCode = 50013;
14                 SET @errorMessage = 'Invalid input. Cutting Type cannot be empty and must be up to 20 characters.';
15                 RETURN;
16             END
17
18         IF LEN(@machine_type) = 0 OR LEN(@machine_type) > 20
19             BEGIN
20                 SET @errorCode = 50014;
21                 SET @errorMessage = 'Invalid input. Machine Type cannot be empty and must be up to 20 characters.';
22                 RETURN;
23             END
24
25         -- Data manipulation
26         INSERT INTO Cut_Process (process_id, cutting_type, machine_type)
27             VALUES (@process_id, @cutting_type, @machine_type);
28             ...
29     END TRY
30     BEGIN CATCH
31         -- Handle errors
32         SET @errorCode = ERROR_NUMBER();
33         SET @errorMessage = ERROR_MESSAGE();
34         RETURN;
35     END CATCH;
36 END;
```

4. Enter a new assembly wit its customer_name, assembly_details, assembly_id and date ordered and associate it with one or more processes
- 4.1. Enter a new Assembly procedure:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1  CREATE PROCEDURE EnterNewAssembly
2      @assembly_id VARCHAR(20),
3      @date_ordered DATE,
4      @assembly_details VARCHAR(100),
5      @cname VARCHAR(20),
6      @errorCode INT OUTPUT,
7      @errorMessage NVARCHAR(4000) OUTPUT
8  AS
9  BEGIN
10     BEGIN TRY
11         -- Data validation
12         IF LEN(@assembly_details) = 0 OR LEN(@assembly_details) > 100
13             BEGIN
14                 SET @errorCode = 50015;
15                 SET @errorMessage = 'Invalid assembly details. Assembly details must be provided up to 100 characters.';
16                 RETURN;
17             END
18
19         IF NOT EXISTS (SELECT 1 FROM Customer WHERE cname = @cname)
20             BEGIN
21                 SET @errorCode = 50016;
22                 SET @errorMessage = 'Customer does not exist. Please provide a valid customer name.';
23                 RETURN;
24             END
25
26         -- Data manipulation
27         INSERT INTO Assembly (assembly_id, date_ordered, assembly_details, cname)
28             VALUES (@assembly_id, @date_ordered, @assembly_details, @cname);
29         -- Data manipulation
30         INSERT INTO Assembly (assembly_id, date_ordered, assembly_details, cname)
31             VALUES (@assembly_id, @date_ordered, @assembly_details, @cname);
32
33     END TRY
34     BEGIN CATCH
35         -- Handle errors
36         SET @errorCode = ERROR_NUMBER();
37         SET @errorMessage = ERROR_MESSAGE();
38
39         RETURN;
40     END CATCH;
41 
```

4.2: Enter a new manufacture procedure:

```

1  CREATE PROCEDURE EnterNewManufacture
2      @assembly_id VARCHAR(20),
3      @process_id VARCHAR(20),
4      @job_no INT,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Check if the combination of assembly_id and process_id already exists
11         IF EXISTS (SELECT 1 FROM manufacture WHERE assembly_id = @assembly_id AND process_id = @process_id)
12             BEGIN
13                 SET @errorCode = 50017;
14                 SET @errorMessage = 'Combination of assembly_id and process_id already exists. Please provide a unique combination.';
15                 RETURN;
16             END
17
18         -- Data manipulation
19         INSERT INTO manufacture (assembly_id, process_id, job_no)
20             VALUES (@assembly_id, @process_id, @job_no);
21     END TRY
22     BEGIN CATCH
23         -- Handle errors
24         SET @errorCode = ERROR_NUMBER();
25         SET @errorMessage = ERROR_MESSAGE();
26         RETURN;
27     END CATCH;
28 END;

```

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

5.1: Enter a new assembly account Procedure:

```

1  -- Procedure for Assembly_account
2  CREATE PROCEDURE InsertAssemblyAccount
3      @account_no INT,
4      @date_of_establishment DATE,
5      @details_1 NUMERIC(10,2),
6      @errorCode INT OUTPUT,
7      @errorMessage NVARCHAR(4000) OUTPUT
8  AS
9  BEGIN
10     BEGIN TRY
11         -- Insert the record into Assembly_account table
12         INSERT INTO Assembly_account (assembly_account_no, date_of_establishment, details_1)
13             VALUES (@account_no, @date_of_establishment, @details_1);
14
15         -- Success
16         SET @errorCode = 0;
17         SET @errorMessage = 'Assembly_account record inserted successfully.';
18     END TRY
19     BEGIN CATCH
20         -- Handle errors
21         SET @errorCode = 50018; -- Custom error code for assembly account
22         SET @errorMessage = ERROR_MESSAGE();
23     END CATCH;
24 END;
25

```

5.2: Enter a new department account procedure:

```
1  -- Procedure for Department_account
2  CREATE PROCEDURE InsertDepartmentAccount
3      @account_no INT,
4      @date_of_establishment DATE,
5      @details_2 NUMERIC(10, 2),
6      @errorCode INT OUTPUT,
7      @errorMessage NVARCHAR(4000) OUTPUT
8  AS
9  BEGIN
10     BEGIN TRY
11         -- Insert the record into Department_account table
12         INSERT INTO Department_account (department_account_no, date_of_establishment, details_2)
13         VALUES (@account_no, @date_of_establishment, @details_2);
14
15         -- Success
16         SET @errorCode = 0;
17         SET @errorMessage = 'Department_account record inserted successfully.';
18     END TRY
19     BEGIN CATCH
20         -- Handle errors
21         SET @errorCode = 50019; -- Custom error code for department account
22         SET @errorMessage = ERROR_MESSAGE();
23     END CATCH;
24 END;
25 |
```

5.3: Enter a new process account procedure:

```
1  -- Procedure for Process_account
2  CREATE PROCEDURE InsertProcessAccount
3      @account_no INT,
4      @date_of_establishment DATE,
5      @details_3 NUMERIC(10, 2),
6      @errorCode INT OUTPUT,
7      @errorMessage NVARCHAR(4000) OUTPUT
8  AS
9  BEGIN
10     BEGIN TRY
11         -- Insert the record into Process_account table
12         INSERT INTO Process_account (process_account_no, date_of_establishment, details_3)
13         VALUES (@account_no, @date_of_establishment, @details_3);
14
15         -- Success
16         SET @errorCode = 0;
17         SET @errorMessage = 'Process_account record inserted successfully.';
18     END TRY
19     BEGIN CATCH
20         -- Handle errors
21         SET @errorCode = 50020; -- Custom error code for process account
22         SET @errorMessage = ERROR_MESSAGE();
23     END CATCH;
24 END;
```

5.4: Enter a new assembly expenditure:

```

1  -- Procedure for Assembly_expenditure
2  CREATE PROCEDURE InsertAssemblyExpenditure
3      @assembly_account_no INT,
4      @assembly_id NVARCHAR(20),
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Insert the record into Assembly_expenditure table
11         INSERT INTO Assembly_expenditure (assembly_account_no, assembly_id)
12             VALUES (@assembly_account_no, @assembly_id);
13
14         -- Success
15         SET @errorCode = 0;
16         SET @errorMessage = 'Assembly_expenditure record inserted successfully.';
17     END TRY
18     BEGIN CATCH
19         -- Handle errors
20         SET @errorCode = 50021; -- Custom error code for assembly expenditure
21         SET @errorMessage = ERROR_MESSAGE();
22     END CATCH;
23 END;

```

5.5: Enter a new department expenditure procedure:

```

1  -- Procedure for Department_expenditure
2  CREATE PROCEDURE InsertDepartmentExpenditure
3      @department_account_no INT,
4      @department_no INT,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Insert the record into Department_expenditure table
11         INSERT INTO Department_expenditure (department_account_no, department_no)
12             VALUES (@department_account_no, @department_no);
13
14         -- Success
15         SET @errorCode = 0;
16         SET @errorMessage = 'Department_expenditure record inserted successfully.';
17     END TRY
18     BEGIN CATCH
19         -- Handle errors
20         SET @errorCode = 50022; -- Custom error code for department expenditure
21         SET @errorMessage = ERROR_MESSAGE();
22     END CATCH;
23 END;

```

5.6: Enter a new process expenditure procedure:

```

1  -- Procedure for Process_expenditure
2  CREATE PROCEDURE InsertProcessExpenditure
3      @process_account_no INT,
4      @process_id NVARCHAR(20),
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Insert the record into Process_expenditure table
11         INSERT INTO Process_expenditure (process_account_no, process_id)
12             VALUES (@process_account_no, @process_id);
13
14         -- Success
15         SET @errorCode = 0;
16         SET @errorMessage = 'Process_expenditure record inserted successfully.';
17     END TRY
18     BEGIN CATCH
19         -- Handle errors
20         SET @errorCode = 50023; -- Custom error code for process expenditure
21         SET @errorMessage = ERROR_MESSAGE();
22     END CATCH;
23 END;
24
25

```

6. Enter a new job, given its job_no, assembly_id, process_id and date the job commenced

6.1: Enter a new job procedure:

```

1  -- Procedure for entering a new job
2  CREATE PROCEDURE EnterNewJob
3      @job_no INT,
4      @date_of_commencement DATE,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(4000) OUTPUT
7  AS
8  BEGIN
9      BEGIN TRY
10         -- Insert the record into the Job table
11         INSERT INTO Job (job_no, date_of_commencement)
12             VALUES (@job_no, @date_of_commencement);
13
14         -- Success
15         SET @errorCode = 0;
16         SET @errorMessage = 'Job record inserted successfully.';
17     END TRY
18     BEGIN CATCH
19         -- Handle errors
20         SET @errorCode = ERROR_NUMBER();
21         SET @errorMessage = ERROR_MESSAGE();
22     END CATCH;
23 END;

```

6.2: Enter a fit job procedure:

```

1  --Procedure for inserting details into Fit_job
2  CREATE PROCEDURE InsertFitJobDetails
3      @job_no INT,
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(4000) OUTPUT
6  AS
7  BEGIN
8      BEGIN TRY
9          -- Insert the record into the Fit_job table
10         INSERT INTO Fit_job (job_no)
11             VALUES (@job_no);
12
13         -- Success
14         SET @errorCode = 0;
15         SET @errorMessage = 'Fit_job details inserted successfully.';
16     END TRY
17     BEGIN CATCH
18         -- Handle errors
19         SET @errorCode = ERROR_NUMBER();
20         SET @errorMessage = ERROR_MESSAGE();
21     END CATCH;
22 END;

```

6.3: Enter a paint job procedure:

```

1  -- Procedure for inserting details into Paint_job
2  CREATE PROCEDURE InsertPaintJobDetails
3      @job_no INT,
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(4000) OUTPUT
6  AS
7  BEGIN
8      BEGIN TRY
9          -- Insert the record into the Paint_job table
10         INSERT INTO Paint_job (job_no)
11             VALUES (@job_no);
12
13         -- Success
14         SET @errorCode = 0;
15         SET @errorMessage = 'Paint_job details inserted successfully.';
16     END TRY
17     BEGIN CATCH
18         -- Handle errors
19         SET @errorCode = ERROR_NUMBER();
20         SET @errorMessage = ERROR_MESSAGE();
21     END CATCH;
22 END;

```

6.4: Enter a cut job procedure:

```

1  --Procedure for inserting details into Cut_job
2  CREATE PROCEDURE InsertCutJobDetails
3      @job_no INT,
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(4000) OUTPUT
6  AS
7  BEGIN
8      BEGIN TRY
9          -- Insert the record into the Cut_job table
10         INSERT INTO Cut_job (job_no)
11             VALUES (@job_no);
12
13         -- Success
14         SET @errorCode = 0;
15         SET @errorMessage = 'Cut_job details inserted successfully.';
16     END TRY
17     BEGIN CATCH
18         -- Handle errors
19         SET @errorCode = ERROR_NUMBER();
20         SET @errorMessage = ERROR_MESSAGE();
21     END CATCH;
22 END;

```

6.5: Enter job in manufacture procedure:

```

1  CREATE PROCEDURE InsertManufactureJob (
2      @jobNumber INT,
3      @assemblyId VARCHAR(20),
4      @processId VARCHAR(20),
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(255) OUTPUT
7  )
8  AS
9  BEGIN
10     -- Check if the pair is legit existing in the manufacture table
11     IF NOT EXISTS (
12         SELECT 1
13         FROM manufacture
14         WHERE assembly_id = @assemblyId
15             AND process_id = @processId
16     )
17     BEGIN
18         -- Set error code and message for an invalid pair
19         SET @errorCode = 1;
20         SET @errorMessage = 'Invalid pair. This assembly_id and process_id combination does not exist in the manufacture table.';
21     END
22     ELSE
23     BEGIN
24         -- Update the manufacture table to associate the new job_no with the existing pair
25         UPDATE manufacture
26             SET job_no = @jobNumber
27             WHERE assembly_id = @assemblyId
28                 AND process_id = @processId;

```

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

7.1. Check for job procedure:

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1 -- Procedure to check if a job with the specified job number exists
2 CREATE PROCEDURE CheckJobExistence (
3     @jobNumber INT,
4     @existsFlag INT OUTPUT
5 )
6 AS
7 BEGIN
8     -- Check if the job number exists in the Job table
9     IF EXISTS (
10         SELECT 1
11         FROM Job
12         WHERE job_no = @jobNumber
13     )
14     BEGIN
15         -- Set existsFlag to 1 if the job exists
16         SET @existsFlag = 1;
17     END
18     ELSE
19     BEGIN
20         -- Set existsFlag to 0 if the job does not exist
21         SET @existsFlag = 0;
22     END
23 END;
24
```

7.2: Check job type procedure:

```
1 -- Procedure to determine the type of job (Fit, Paint, or Cut) based on the job number
2 CREATE PROCEDURE DetermineJob (
3     @jobNumber INT,
4     @jobType NVARCHAR(20) OUTPUT
5 )
6 AS
7 BEGIN
8     -- Determine the job type based on the presence in each table
9     IF EXISTS (
10         SELECT 1
11         FROM Fit_job
12         WHERE job_no = @jobNumber
13     )
14     BEGIN
15         SET @jobType = 'Fit Job';
16     END
17     ELSE IF EXISTS (
18         SELECT 1
19         FROM Paint_job
20         WHERE job_no = @jobNumber
21     )
22     BEGIN
23         SET @jobType = 'Paint Job';
24     END
25     ELSE IF EXISTS (
26         SELECT 1
27         FROM Cut_job
28         WHERE job_no = @jobNumber
29     )
30     BEGIN
31         SET @jobType = 'Cut Job';
32     END
33 END;
```

```

26      SELECT 1
27      FROM Cut_job
28      WHERE job_no = @jobNumber
29  )
30  BEGIN
31      SET @jobType = 'Cut Job';
32  END
33  ELSE
34  BEGIN
35      -- Set a default value if the job type cannot be determined
36      SET @jobType = 'Unknown Job';
37  END
38 END;

```

7.3: Enter fit information procedure:

```

1  -- Procedure to update Fit Job labor time
2  CREATE PROCEDURE UpdateFitJobLaborTime (
3      @jobNumber INT,
4      @fitLaborTime TIME,
5      @errorCode INT OUTPUT,
6      @errorMessage NVARCHAR(255) OUTPUT
7  )
8  AS
9  BEGIN
10     -- Check if the job number exists in the Fit Job table
11     IF EXISTS (
12         SELECT 1
13         FROM Fit_job
14         WHERE job_no = @jobNumber
15     )
16     BEGIN
17         -- Update the Fit Job table with the new fit_labor_time
18         UPDATE Fit_job
19             SET fit_labor_time = @fitLaborTime
20             WHERE job_no = @jobNumber;
21
22         -- Set success code and message
23         SET @errorCode = 0;
24         SET @errorMessage = 'Fit Job labor time updated successfully.';
25     END
26     ELSE
27     BEGIN
28         -- Set error code and message for job not found
29     END
30     ELSE
31     BEGIN
32         -- Set error code and message for job not found
33         SET @errorCode = 1;
34         SET @errorMessage = 'Fit Job with the specified job number does not exist.';
35     END
36 END;

```

7.4: Enter paint job information procedure:

```
1  -- Procedure to update Paint Job details
2  CREATE PROCEDURE UpdatePaintJobDetails (
3      @jobNumber INT,
4      @color VARCHAR(20),
5      @volume DOUBLE,
6      @paintLaborTime TIME,
7      @errorCode INT OUTPUT,
8      @errorMessage NVARCHAR(255) OUTPUT
9  )
10 AS
11 BEGIN
12     -- Check if the job number exists in the Paint Job table
13     IF EXISTS (
14         SELECT 1
15         FROM Paint_job
16         WHERE job_no = @jobNumber
17     )
18     BEGIN
19         -- Update Paint Job details
20         UPDATE Paint_job
21             SET color = @color,
22                 volume = @volume,
23                 paint_labor_time = @paintLaborTime
24             WHERE job_no = @jobNumber;
25
26         -- Set success code and message
27         SET @errorCode = 0;
28         SET @errorMessage = 'Paint Job details updated successfully.';
29
30     ELSE
31     BEGIN
32         -- Set error code and message if job number does not exist
33         SET @errorCode = 1;
34         SET @errorMessage = 'Error: Job with the specified job number does not exist in Paint Job table.';
35     END
36 END;
```

7.5: Enter cut job information procedure:

```

1  -- Procedure to update Cut Job details
2  CREATE PROCEDURE UpdateCutJobDetails (
3      @jobNumber INT,
4      @machineType VARCHAR(20),
5      @machineTime TIME,
6      @materialUsed VARCHAR(50),
7      @cutLaborTime TIME,
8      @errorCode INT OUTPUT,
9      @errorMessage NVARCHAR(255) OUTPUT
10 )
11 AS
12 BEGIN
13     -- Check if the job number exists in the Cut Job table
14     IF EXISTS (
15         SELECT 1
16         FROM Cut_job
17         WHERE job_no = @jobNumber
18     )
19     BEGIN
20         -- Update Cut Job details
21         UPDATE Cut_job
22             SET type_of_machine = @machineType,
23                 time_of_machine = @machineTime,
24                 material_used = @materialUsed,
25                 cut_labor_time = @cutLaborTime
26             WHERE job_no = @jobNumber;
27
28         -- Set success code and message

```

7.6: Check date of completion procedure:

```

1  CREATE PROCEDURE CheckDateCompletionGreaterThanCommencement (
2      @jobNumber INT,
3      @dateCompleted DATE,
4      @isValid BIT OUTPUT
5  )
6  AS
7  BEGIN
8      DECLARE @dateCommenced DATE;
9
10     -- Retrieve the date of commencement from the job table
11     SELECT @dateCommenced = date_of_commencement
12     FROM job
13     WHERE job_no = @jobNumber;
14
15     -- Check if the date of completion is greater than the date of commencement
16     IF @dateCompleted > @dateCommenced
17     BEGIN
18         SET @isValid = 1; -- Date is valid
19     END
20     ELSE
21     BEGIN
22         SET @isValid = 0; -- Date is not valid
23     END
24 END;
25

```

7.7: Enter date of completion procedure:

```

1  CREATE PROCEDURE UpdateCompletionDate
2      @jobNumber INT,
3      @newCompletionDate DATE,
4      @errorCode INT OUTPUT,
5      @errorMessage NVARCHAR(MAX) OUTPUT
6  AS
7  BEGIN
8      SET NOCOUNT ON;
9
10     BEGIN TRY
11         -- Validate that the new completion date is greater than the date of commencement
12         IF EXISTS (
13             SELECT 1
14             FROM job
15             WHERE job_no = @jobNumber
16             AND @newCompletionDate <= date_of_commencement
17         )
18         BEGIN
19             SET @errorCode = 1; -- You can define your error codes
20             SET @errorMessage = 'Error: Date of completion must be greater than the date of commencement.';
21         END
22     ELSE
23         BEGIN
24             -- Update the date_of_completion
25             UPDATE job
26             SET date_of_completion = @newCompletionDate
27             WHERE job_no = @jobNumber;
28
29             SET @errorCode = 0;
30             SET @errorMessage = 'Date of completion updated successfully.';
31         END
32     END TRY
33     BEGIN CATCH
34         SET @errorCode = ERROR_NUMBER();
35         SET @errorMessage = ERROR_MESSAGE();
36     END CATCH
37 END;
38

```

8. Enter a transaction_no and its sup-cost and update all the costs of the affected accounts by adding sup-cost to their current values of details

8.1. Enter a new transaction procedure:

```

1  CREATE PROCEDURE EnterNewTransaction
2      @transaction_no INT,
3      @sup_cost NUMERIC(10, 2),
4      @assembly_account_no INT,
5      @department_account_no INT,
6      @process_account_no INT
7  AS
8  BEGIN
9      -- Update Assembly_account details
10     UPDATE Assembly_account
11     SET details_1 = details_1 + (@sup_cost)
12     WHERE assembly_account_no = @assembly_account_no;
13
14     -- Update Department_account details
15     UPDATE Department_account
16     SET details_2 = details_2 + (@sup_cost)
17     WHERE department_account_no = @department_account_no;
18
19     -- Update Process_account details
20     UPDATE Process_account
21     SET details_3 = details_3 + (@sup_cost)
22     WHERE process_account_no = @process_account_no;
23
24     -- Insert into Cost table
25     INSERT INTO Cost (transaction_no, sup_cost, assembly_account_no, department_account_no, process_account_no)
26     VALUES (@transaction_no, @sup_cost, @assembly_account_no, @department_account_no, @process_account_no);
27 END;

```

8.1 Get total code procedure:

```

1  CREATE PROCEDURE GetTotalCost
2      @assembly_id VARCHAR(20),
3      @total_cost NUMERIC(10, 2) OUTPUT,
4      @error_message NVARCHAR(MAX) OUTPUT
5  AS
6  BEGIN
7      SET NOCOUNT ON;
8
9      BEGIN TRY
10         SELECT @total_cost = ISNULL(SUM(c.sup_cost), 0)
11         FROM Cost c
12         WHERE c.assembly_account_no IN (
13             SELECT aea.assembly_account_no
14             FROM assembly_expenditure aea
15             WHERE aea.assembly_id = @assembly_id
16         );
17
18         SET @error_message = NULL;
19     END TRY
20     BEGIN CATCH
21         SET @total_cost = 0;
22         SET @error_message = ERROR_MESSAGE();
23     END CATCH;
24 END;

```

9. Retrieve the total cost incurred on an assembly-id

Total cost procedure:

```

1  CREATE PROCEDURE GetTotalCost
2      @assembly_id VARCHAR(20),
3      @total_cost NUMERIC(10, 2) OUTPUT,
4      @error_message NVARCHAR(MAX) OUTPUT
5  AS
6  BEGIN
7      SET NOCOUNT ON;
8
9      BEGIN TRY
10         SELECT @total_cost = ISNULL(SUM(c.sup_cost), 0)
11         FROM Cost c
12         WHERE c.assembly_account_no IN (
13             SELECT aea.assembly_account_no
14             FROM assembly_expenditure aea
15             WHERE aea.assembly_id = @assembly_id
16         );
17
18         SET @error_message = NULL;
19     END TRY
20     BEGIN CATCH
21         SET @total_cost = 0;
22         SET @error_message = ERROR_MESSAGE();
23     END CATCH;
24 END;
25

```

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

```

1  CREATE PROCEDURE GetTotalLaborTimeInDepartment
2      @department_no INT,
3      @given_date DATE,
4      @total_labor_time FLOAT OUTPUT,
5      @error_message NVARCHAR(MAX) OUTPUT
6  AS
7  BEGIN
8      SET NOCOUNT ON;
9
10     BEGIN TRY
11         SELECT @total_labor_time = ISNULL(SUM(j.fit_labor_time + j.paint_labor_time + cj.cut_labor_time), 0)
12         FROM Job j
13         LEFT JOIN Fit_job fj ON j.job_no = fj.job_no
14         LEFT JOIN Paint_job pj ON j.job_no = pj.job_no
15         LEFT JOIN Cut_job cj ON j.job_no = cj.job_no
16         WHERE j.date_of_completion = @given_date
17         AND EXISTS (
18             SELECT 1
19             FROM manufacture m
20             INNER JOIN Process p ON m.process_id = p.process_id
21             WHERE m.job_no = j.job_no
22             AND p.department_no = @department_no
23         );
24
25         SET @error_message = NULL;
26     END TRY
27     BEGIN CATCH
28         SET @total_labor_time = 0;
29     END CATCH;
30 END;
31

```

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

```

1  CREATE PROCEDURE GetProcessesForAssembly
2    @assembly_id VARCHAR(20),
3    @process_details NVARCHAR(MAX) OUTPUT,
4    @error_message NVARCHAR(MAX) OUTPUT
5  AS
6  BEGIN
7    SET NOCOUNT ON;
8
9    BEGIN TRY
10      SELECT @process_details = COALESCE(@process_details + ', ', '') +
11        |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
12        p.process_data + ' (Department: ' + CONVERT(NVARCHAR(10), p.department_no) + ')'
13      FROM manufacture m
14      INNER JOIN Process p ON m.process_id = p.process_id
15      INNER JOIN Job j ON m.job_no = j.job_no
16      WHERE m.assembly_id = @assembly_id
17      ORDER BY j.date_of_completion;
18
19      SET @error_message = NULL;
20    END TRY
21    BEGIN CATCH
22      SET @process_details = NULL;
23      SET @error_message = ERROR_MESSAGE();
24    END CATCH;
25  END;

```

12. Retrieve the customers whose category is in the given range

```

1  CREATE PROCEDURE GetCustomersInRange
2    @min_category INT,
3    @max_category INT,
4    @customer_details NVARCHAR(MAX) OUTPUT,
5    @error_message NVARCHAR(MAX) OUTPUT
6  AS
7  BEGIN
8    SET NOCOUNT ON;
9
10   BEGIN TRY
11     SELECT @customer_details = STRING_AGG(cname, ', ')
12       |   |   |   |   |   |   |   |   |
13     FROM Customer
14     WHERE category BETWEEN @min_category AND @max_category
15     ORDER BY cname;
16
17     SET @error_message = NULL;
18   END TRY
19   BEGIN CATCH
20     SET @customer_details = NULL;
21     SET @error_message = ERROR_MESSAGE();
22   END CATCH;
23 END;

```

13. Delete all cut-jobs whose job_no is in given range

```

1  CREATE PROCEDURE DeleteCutJobsInRange
2      @min_job_no INT,
3      @max_job_no INT,
4      @error_message NVARCHAR(MAX) OUTPUT
5  AS
6  BEGIN
7      SET NOCOUNT ON;
8
9      BEGIN TRY
10         DELETE FROM Cut_job
11         WHERE job_no BETWEEN @min_job_no AND @max_job_no;
12
13         SET @error_message = NULL;
14     END TRY
15     BEGIN CATCH
16         SET @error_message = ERROR_MESSAGE();
17     END CATCH;
18 END;

```

14. Change the color of a given paint job

```

1  CREATE PROCEDURE ChangePaintJobColor
2      @job_number INT,
3      @new_color VARCHAR(20),
4      @error_message NVARCHAR(MAX) OUTPUT
5  AS
6  BEGIN
7      SET NOCOUNT ON;
8
9      BEGIN TRY
10         UPDATE Paint_job
11             SET color = @new_color
12             WHERE job_no = @job_number;
13
14         SET @error_message = NULL;
15     END TRY
16     BEGIN CATCH
17         SET @error_message = ERROR_MESSAGE();
18     END CATCH;
19 END;

```

Output of all Procedures:

Results	<u>Messages</u>
---------	-----------------

Query succeeded: Affected rows: 0

Task 5.2: Java application program that uses JDBC and Azure Database

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

```

185     case "1":
186         System.out.println("Please enter customer name:");
187         // Consume any remaining newline characters
188         sc.nextLine();
189         // Read customer name
190         final String cname = sc.nextLine();
191
192         System.out.println("Please enter customer address:");
193         // Read customer address
194         final String caddress = sc.nextLine();
195
196         System.out.println("Please enter customer category:");
197         final int category;
198         try {
199             // Read and parse category as an integer
200             category = Integer.parseInt(sc.nextLine().trim());
201         } catch (NumberFormatException e) {
202             // Handle invalid input for category
203             System.out.println("Error Message: Invalid input for category. Please enter a valid integer.");
204             break; // Terminate the loop or take appropriate action
205         }
206
207         System.out.println("Connecting to the database...");
208         try (Connection connection = DriverManager.getConnection(URL)) {
209             try (CallableStatement statement = connection.prepareCall(ENTER_NEW_CUSTOMER)) {
210                 // Set parameters for the stored procedure
211                 statement.setString(1, cname);
212                 statement.setString(2, caddress);
213                 statement.setInt(3, category);
214
215                 // Register output parameters for error code and message
216                 statement.registerOutParameter(4, Types.INTEGER);
217                 statement.registerOutParameter(5, Types.NVARCHAR);
218
219                 System.out.println("Dispatching the stored procedure...");
220                 // Execute the stored procedure
221                 statement.execute();
222
223                 // Retrieve error code and message from the stored procedure
224                 int aerrorCode = statement.getInt(4);
225                 String aerrorMessage = statement.getString(5);
226
227                 if (aerrorCode != 0) {
228                     // Error occurred, display error message
229                     System.out.println("Error Code: " + aerrorCode);
230                     System.out.println("Error Message: " + aerrorMessage);
231                     System.out.println("Failed to add customer. Returning to the main menu.");
232                 } else {
233                     // No error, success
234                     System.out.println("Done. Customer added.");
235                 }
236             } catch (SQLException e) {
237                 // Handle SQL exceptions during stored procedure execution
238                 System.out.println("Error: " + e.getMessage());
239                 System.out.println("Failed to add customer. Returning to the main menu.");
240             }
241         } catch (SQLException e) {
242             // Handle connection errors
243             System.out.println("Connection error: " + e.getMessage());
244         }
245         break;
246     }

```

2. Enter a new department (infrequent).

```

248     case "2":
249         // Prompt the user to enter the department number.
250         System.out.println("Please enter department number:");
251         // Read the entered department number.
252         int department_no = sc.nextInt();
253
254         // Consume the newline character left by previous sc.nextInt().
255         sc.nextLine();
256
257         // Prompt the user to enter department data.
258         System.out.println("Please enter department data:");
259         // Read the entered department data.
260         String department_data = sc.nextLine();
261
262         // Inform the user about the database connection process.
263         System.out.println("Connecting to the database...");
264
265         try (Connection connection = DriverManager.getConnection(URL)) {
266             // Prepare the stored procedure call to add a new department.
267             try (CallableStatement statement = connection.prepareCall(ENTER_NEW_DEPARTMENT)) {
268                 // Set the input parameters for the stored procedure.
269                 statement.setInt(1, department_no);
270                 statement.setString(2, department_data);
271
272                 // Register output parameters for error code and message.
273                 statement.registerOutParameter(3, Types.INTEGER);
274                 statement.registerOutParameter(4, Types.NVARCHAR);
275
276                 // Inform the user about dispatching the stored procedure.
277                 System.out.println("Dispatching the stored procedure...");
278                 // Execute the stored procedure.
279                 statement.execute();
280
281                 // Retrieve error code and message from the stored procedure.
282                 int berrortCode = statement.getInt(3);
283                 String berrortMessage = statement.getString(4);
284
285                 if (berrortCode != 0) {
286                     // Error occurred, display error message.
287                     System.out.println("Error Code: " + berrortCode);
288                     System.out.println("Error Message: " + berrortMessage);
289                     System.out.println("Failed to add department. Returning to the main menu.");
290                 } else {
291                     // No error, success.
292                     System.out.println("Done. Department added.");
293                 }
294             } catch (SQLException e) {
295                 // Handle SQL exceptions.
296                 System.out.println("Error: " + e.getMessage());
297                 System.out.println("Failed to add department. Returning to the main menu.");
298             }
299         } catch (SQLException e) {
300             // Handle connection-related exceptions.
301             System.out.println("Connection error: " + e.getMessage());
302         }
303         // Exit the case statement for adding a department.
304         break;
305     }

```

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

```

306     // Case "3" represents the user's choice to add a new process.
307     case "3":
308         System.out.println("Please enter process ID:");
309         sc.nextLine();
310         // read process id
311         String process_id = sc.nextLine();
312
313         System.out.println("Please enter process data:");
314         // read process data
315         String process_data = sc.nextLine();
316
317         System.out.println("Please enter department number:");
318
319         final int department_noc;
320         try {
321             department_noc = Integer.parseInt(sc.nextLine().trim());
322         } catch (NumberFormatException e) {
323             System.out.println("Error Message: Invalid input for department number. Please enter a valid integer.");
324             break; // Terminate the loop or take appropriate action
325         }
326
327         System.out.println("Connecting to the database...");
328
329         try (Connection connection = DriverManager.getConnection(URL)) {
330             try (CallableStatement statement = connection.prepareCall(ENTER_NEW_PROCESS)) {
331                 // Set parameters for the main process
332                 statement.setString(1, process_id);
333                 statement.setString(2, process_data);
334                 statement.setInt(3, department_noc);
335
336                 // Register output parameters for error code and message
337                 statement.registerOutParameter(4, Types.INTEGER);
338                 statement.registerOutParameter(5, Types.NVARCHAR);
339
340                 System.out.println("Dispatching the stored procedure for Process...");
341                 statement.execute();
342
343                 // Retrieve error code and message from the stored procedure
344                 int errorCode = statement.getInt(4);
345                 String errorMessage = statement.getString(5);
346
347                 if (errorCode != 0) {
348                     // Error occurred, display error message
349                     System.out.println("Error Code: " + errorCode);
350                     System.out.println("Error Message: " + errorMessage);
351                     System.out.println("Failed to add process. Returning to the main menu.");
352                 } else {
353                     // No error, success
354                     System.out.println("Done. Process added.");
355
356                     boolean processDetailsEnteredSuccessfully = false;
357
358                     while (!processDetailsEnteredSuccessfully) {
359                         // Ask the user to choose the type of process to add
360                         System.out.println("Choose the type of process to add:");
361                         System.out.println("1. Fit Process");
362                         System.out.println("2. Paint Process");
363                         System.out.println("3. Cut Process");
364
365                         int processTypeChoice = sc.nextInt();
366
367                         if (processTypeChoice == 1) {
368                             System.out.println("Fit Process selected.");
369
370                             // Add logic for Fit Process
371
372                             System.out.println("Process added successfully!");
373                             processDetailsEnteredSuccessfully = true;
374                         } else if (processTypeChoice == 2) {
375                             System.out.println("Paint Process selected.");
376
377                             // Add logic for Paint Process
378
379                             System.out.println("Process added successfully!");
380                             processDetailsEnteredSuccessfully = true;
381                         } else if (processTypeChoice == 3) {
382                             System.out.println("Cut Process selected.");
383
384                             // Add logic for Cut Process
385
386                             System.out.println("Process added successfully!");
387                             processDetailsEnteredSuccessfully = true;
388                         } else {
389                             System.out.println("Invalid choice. Please enter 1, 2, or 3.");
390                         }
391
392                         System.out.println("Do you want to add another process? (y/n)");
393                         String answer = sc.nextLine();
394                         if (answer.equalsIgnoreCase("n")) {
395                             System.out.println("Thank you for using the Job-Shop Accounting System!");
396                             System.out.println("Exiting program...");
397                             System.out.println("Goodbye!");
398                             System.out.println("Press any key to exit...");
399                             sc.nextLine();
400                             System.exit(0);
401                         }
402                     }
403
404                 }
405             }
406         }
407     }
408
409     System.out.println("Process added successfully!");
410
411     System.out.println("Do you want to add another process? (y/n)");
412     String answer = sc.nextLine();
413     if (answer.equalsIgnoreCase("n")) {
414         System.out.println("Thank you for using the Job-Shop Accounting System!");
415         System.out.println("Exiting program...");
416         System.out.println("Goodbye!");
417         System.out.println("Press any key to exit...");
418         sc.nextLine();
419         System.exit(0);
420     }
421
422 }

```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
366
367         switch (processTypeChoice) {
368             case 1:
369                 // Now, let's handle the Fit Process
370                 System.out.println("Please enter Fit Type:");
371                 sc.nextLine(); // Consume the newline character left by previous sc.nextInt()
372                 String fit_type = sc.nextLine();
373
374             try (CallableStatement fitStatement = connection.prepareCall(ENTER_NEW_FIT_PROCESS)) {
375                 // Set parameters for the Fit Process
376                 fitStatement.setString(1, process_id);
377                 fitStatement.setString(2, fit_type);
378
379                 // Register output parameters for error code and message
380                 fitStatement.registerOutParameter(3, Types.INTEGER);
381                 fitStatement.registerOutParameter(4, Types.NVARCHAR);
382
383                 System.out.println("Dispatching the stored procedure for Fit Process...");
384                 fitStatement.execute();
385
386                 // Retrieve error code and message from the stored procedure
387                 int errorCodeFitProcess = fitStatement.getInt(3);
388                 String errorMessageFitProcess = fitStatement.getString(4);
389
390                 if (errorCodeFitProcess != 0) {
391                     // Error occurred, display error message
392                     System.out.println("Error Code: " + errorCodeFitProcess);
393                     System.out.println("Error Message: " + errorMessageFitProcess);
394                     System.out.println("Failed to add Fit Process. Please try again.");
395                 } else {
396                     // No error, success
397                     System.out.println("Done. Fit Process added.");
398                     processDetailsEnteredSuccessfully = true;
399                 }
400             } catch (SQLException e) {
401                 // Handle SQL exceptions
402                 System.out.println("Error: " + e.getMessage());
403                 System.out.println("Failed to add Fit Process. Returning to process type selection.");
404             }
405             break;
406
407         case 2:
408             // Now, let's handle the Paint Process
409             System.out.println("Please enter Paint Type:");
410             sc.nextLine(); // Consume the newline character left by previous sc.nextInt()
411             String paint_type = sc.nextLine();
412
413             System.out.println("Please enter Paint Method:");
414             String paint_method = sc.nextLine();
415
416             try (CallableStatement paintStatement = connection.prepareCall(ENTER_NEW_PAINT_PROCESS)) {
417                 // Set parameters for the Paint Process
418                 paintStatement.setString(1, process_id);
419                 paintStatement.setString(2, paint_type);
420                 paintStatement.setString(3, paint_method);
421
422                 // Register output parameters for error code and message
423                 paintStatement.registerOutParameter(4, Types.INTEGER);
424                 paintStatement.registerOutParameter(5, Types.NVARCHAR);
425
426                 System.out.println("Dispatching the stored procedure for Paint Process...");
427                 paintStatement.execute();
428             }
429
430         }
431     }
432 }
```

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
429 // Retrieve error code and message from the stored procedure
430 int errorCodePaintProcess = paintStatement.getInt(4);
431 String errorMessagePaintProcess = paintStatement.getString(5);
432
433 if (errorCodePaintProcess != 0) {
434     // Error occurred, display error message
435     System.out.println("Error Code: " + errorCodePaintProcess);
436     System.out.println("Error Message: " + errorMessagePaintProcess);
437     System.out.println("Failed to add Paint Process. Please try again.");
438 } else {
439     // No error, success
440     System.out.println("Done. Paint Process added.");
441     processDetailsEnteredSuccessfully = true;
442 }
443
444 } catch (SQLException e) {
445     // Handle SQL exceptions
446     System.out.println("Error: " + e.getMessage());
447     System.out.println("Failed to add Paint Process. Returning to process type selection.");
448 }
449 break;
450
451 case 3:
452     // Now, let's handle the Cut Process
453     System.out.println("Please enter Cutting Type:");
454     sc.nextLine(); // Consume the newline character left by previous sc.nextLine()
455     String cuttingType = sc.nextLine();
456
457     System.out.println("Please enter Machine Type:");
458     String machineType = sc.nextLine();
459
460     try (CallableStatement cutStatement = connection.prepareCall(ENTER_NEW_CUT_PROCESS)) {
461         // Set parameters for the Cut Process
462         cutStatement.setString(1, process_id);
463         cutStatement.setString(2, cuttingType);
464         cutStatement.setString(3, machineType);
465
466         // Register output parameters for error code and message
467         cutStatement.registerOutParameter(4, Types.INTEGER);
468         cutStatement.registerOutParameter(5, Types.NVARCHAR);
469
470         System.out.println("Dispatching the stored procedure for Cut Process...");
471         cutStatement.execute();
472
473         // Retrieve error code and message from the stored procedure
474         int errorCodeCutProcess = cutStatement.getInt(4);
475         String errorMessageCutProcess = cutStatement.getString(5);
476
477         if (errorCodeCutProcess != 0) {
478             // Error occurred, display error message
479             System.out.println("Error Code: " + errorCodeCutProcess);
480             System.out.println("Error Message: " + errorMessageCutProcess);
481             System.out.println("Failed to add Cut Process. Please try again.");
482         } else {
483             // No error, success
484             System.out.println("Done. Cut Process added.");
485             processDetailsEnteredSuccessfully = true;
486         }
487     } catch (SQLException e) {
488         // Handle SQL exceptions
489         System.out.println("Error: " + e.getMessage());
490         System.out.println("Failed to add Cut Process. Returning to process type selection.");
491     }
492     break;
493
494     default:
495         System.out.println("Invalid choice. Returning to the main menu.");
496         break;
497     }
498 }
499 } catch (SQLException e) {
500     // Handle SQL exceptions
501     System.out.println("Error: " + e.getMessage());
502     System.out.println("Failed to add process. Returning to the main menu.");
503 }
504 } catch (SQLException e) {
505     System.out.println("Connection error: " + e.getMessage());
506 }
507 break;
```

4. Enter a new assembly with its customer-name, assembly-details, assembly-id, and date-ordered and associate it with one or more processes (40/day).

```
508 // Case "4" represents the user's choice to add a new assembly and manufacture record.
509 case "4":
510     // Get user input for Assembly table
511     System.out.println("Please enter assembly ID:");
512     sc.nextLine();
513     String assemblyId = sc.nextLine();
514     // read date ordered
515     String dateOrdered = getUserInputWithFormat("Please enter date ordered (YYYY-MM-DD):", "yyyy-MM-dd");
516     // read assembly details
517     System.out.println("Please enter assembly details:");
518     String assemblyDetails = sc.nextLine();
519     // read customer name
520     System.out.println("Please enter customer name:");
521     String assemblyName = sc.nextLine();
522
523     // Connecting to the database and executing stored procedure for Assembly
524     try (Connection connection = DriverManager.getConnection(URL)) {
525         try (CallableStatement assemblyStatement = connection.prepareCall(ENTER_NEW_ASSEMBLY)) {
526             assemblyStatement.setString(1, assemblyId);
527             assemblyStatement.setString(2, dateOrdered);
528             assemblyStatement.setString(3, assemblyDetails);
529             assemblyStatement.setString(4, assemblyName);
530
531             // Register output parameters for error code and message
532             assemblyStatement.registerOutParameter(5, Types.INTEGER);
533             assemblyStatement.registerOutParameter(6, Types.NVARCHAR);
534
535             System.out.println("Dispatching the stored procedure for Assembly...");
536             assemblyStatement.execute();
537
538             // Retrieve error code and message from the stored procedure
539             int derrorCode = assemblyStatement.getInt(5);
540             String derrormessage = assemblyStatement.getString(6);
541
542             if (derrorCode != 0) {
543                 // Error occurred, display error message
544                 System.out.println("Error Code: " + derrorCode);
545                 System.out.println("Error Message: " + derrormessage);
546                 System.out.println("Failed to add Assembly record. Returning to the main menu.");
547                 break; // Terminate the loop or take appropriate action
548             } else {
549                 // No error, success
550                 System.out.println("Done. Assembly record added.");
551             }
552         } catch (SQLException e) {
553             // Handle SQL exceptions
554             System.out.println("Error: " + e.getMessage());
555             System.out.println("Failed to add Assembly record. Returning to the main menu.");
556             break; // Terminate the loop or take appropriate action
557         }
558
559         // Ask the user if they want to enter a process for the specific assembly
560         String userChoice = "yes";
561
562         while (userChoice.equalsIgnoreCase("yes")) {
563             System.out.println("Do you want to enter process for this assembly? (yes/no):");
564             userChoice = sc.nextLine();
565
566             if (userChoice.equalsIgnoreCase("yes")) {
567                 // Get user input for Manufacture table
568                 System.out.println("Please enter process ID for Manufacture:");
569                 String processIdManufacture = sc.nextLine();
570
571                 try (CallableStatement processStatement = connection.prepareCall(ENTER_NEW_PROCESS)) {
572                     processStatement.setString(1, processIdManufacture);
573                     processStatement.setString(2, assemblyId);
574
575                     processStatement.registerOutParameter(3, Types.INTEGER);
576                     processStatement.registerOutParameter(4, Types.NVARCHAR);
577
578                     processStatement.execute();
579
580                     int derrorCodeProcess = processStatement.getInt(3);
581                     String derrormessageProcess = processStatement.getString(4);
582
583                     if (derrorCodeProcess != 0) {
584                         System.out.println("Error Code: " + derrorCodeProcess);
585                         System.out.println("Error Message: " + derrormessageProcess);
586                         System.out.println("Failed to add Process record. Returning to the main menu.");
587                         break;
588                     } else {
589                         System.out.println("Done. Process record added.");
590                     }
591                 } catch (SQLException e) {
592                     // Handle SQL exceptions
593                     System.out.println("Error: " + e.getMessage());
594                     System.out.println("Failed to add Process record. Returning to the main menu.");
595                     break;
596                 }
597             }
598         }
599     }
600 }
```

```

570     boolean isValidProcessId = false;
571
572
573     while (!isValidProcessId) {
574         try (CallableStatement manufactureStatement = connection.prepareCall(ENTER_NEW_MANUFACTURE)) {
575             manufactureStatement.setString(1, assemblyId);
576             manufactureStatement.setString(2, processIdManufacture);
577             manufactureStatement.setNull(3, Types.INTEGER); // Placeholder for job_no (null)
578
579             // Register output parameters for error code and message
580             manufactureStatement.registerOutParameter(4, Types.INTEGER);
581             manufactureStatement.registerOutParameter(5, Types.NVARCHAR);
582
583             System.out.println("Dispatching the stored procedure for Manufacture...");
584             manufactureStatement.execute();
585
586             // Retrieve error code and message from the stored procedure
587             int dderrorCode = manufactureStatement.getInt(4);
588             String dderrorMessage = manufactureStatement.getString(5);
589
590             if (dderrorCode != 0) {
591                 // Error occurred, display error message
592                 System.out.println("Error Code: " + dderrorCode);
593                 System.out.println("Error Message: " + dderrorMessage);
594                 System.out.println("Failed to add Manufacture record. Please enter process ID again:");
595
596                 // Prompt user for a new process ID
597                 processIdManufacture = sc.next();
598             } else {
599                 // No error, success
600                 System.out.println("Done. Manufacture record added.");
601                 isValidProcessId = true; // Exit the loop
602             }
603         } catch (SQLException e) {
604             // Handle SQL exceptions
605             System.out.println("Error: " + e.getMessage());
606             System.out.println("Failed to add Manufacture record. Please enter process ID again:");
607
608             // Prompt user for a new process ID
609             processIdManufacture = sc.next();
610         }
611     }
612 }
613 }
614 } catch (SQLException e) {
615     System.out.println("Connection error: " + e.getMessage());
616 }
617
618 break;
619

```

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

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
619 // CASE "5": represents the user's choice to add a new account and expenditure
620 case "5":
621     System.out.println("Enter account number:");
622     int accountNumber = sc.nextInt();
623
624     String dateEstablished = getUserInputWithFormat("Enter date of establishment (YYYY-MM-DD):", "yyyy-MM-dd");
625
626     System.out.println("Choose one of the following types of account:");
627     System.out.println("1. Assembly Account");
628     System.out.println("2. Department Account");
629     System.out.println("3. Process Account");
630
631     int accountTypeChoice = sc.nextInt();
632
633     System.out.println("Enter account details:");
634     sc.nextLine(); // Consume the newline character
635     BigDecimal accountDetails = new BigDecimal(sc.nextLine()); // Use BigDecimal for NUMERIC(10,2)
636
637     int eErrorCode;
638     String eErrorMessage;
639
640     try (Connection connection = DriverManager.getConnection(URL)) {
641         switch (accountTypeChoice) {
642             case 1:
643                 try (CallableStatement assemblyAccountStatement = connection.prepareCall(INSERT_ASSEMBLY_ACCOUNT)) {
644                     assemblyAccountStatement.setInt(1, accountNumber);
645                     assemblyAccountStatement.setString(2, dateEstablished);
646                     assemblyAccountStatement.setBigDecimal(3, accountDetails);
647
648                     assemblyAccountStatement.registerOutParameter(4, Types.INTEGER);
649                     assemblyAccountStatement.registerOutParameter(5, Types.NVARCHAR);
650
651                     System.out.println("Dispatching the stored procedure for Assembly Account...");
652                     assemblyAccountStatement.execute();
653
654                     eErrorCode = assemblyAccountStatement.getInt(4);
655                     eErrorMessage = assemblyAccountStatement.getString(5);
656
657                     // Check for errors in the assembly account addition
658                     if (eErrorCode != 0) {
659                         // Error occurred, display error message
660                         System.out.println("Error Code: " + eErrorCode);
661                         System.out.println("Error Message: " + eErrorMessage);
662                         System.out.println("Failed to add Assembly Account record. Returning to the main menu.");
663                         break; // Terminate the loop or take appropriate action
664                     }
665
666                     // Now, prompt the user for expenditure details
667                     System.out.println("Enter Assembly ID for Expenditure:");
668                     String eassemblyId = sc.nextLine();
669
670                     // Insert expenditure details into the assembly_expenditure table
671                     try (CallableStatement assemblyExpenditureStatement = connection.prepareCall(INSERT_ASSEMBLY_EXPENDITURE)) {
672                         assemblyExpenditureStatement.setInt(1, accountNumber);
673                         assemblyExpenditureStatement.setString(2, eassemblyId);
674
675                         assemblyExpenditureStatement.registerOutParameter(3, Types.INTEGER);
676                         assemblyExpenditureStatement.registerOutParameter(4, Types.NVARCHAR);
677
678                         assemblyExpenditureStatement.execute();
679
680                     }
681
682                 }
683             }
684         }
685     }
686 }
```

```

680
681     int aeErrorCode = assemblyExpenditureStatement.getInt(3);
682     String aeErrorMessage = assemblyExpenditureStatement.getString(4);
683
684     // Check for errors in the assembly expenditure addition
685     if (aeErrorCode != 0) {
686         // Error occurred, display error message
687         System.out.println("Error Code: " + aeErrorCode);
688         System.out.println("Error Message: " + aeErrorMessage);
689         System.out.println("Failed to add Assembly Expenditure record. Returning to the main menu.");
690         break; // Terminate the loop or take appropriate action
691     }
692 } catch (SQLException e) {
693     // Handle SQL exceptions for assembly expenditure statement
694     eErrorCode = 50021; // Custom error code for assembly expenditure
695     eErrorMessage = e.getMessage();
696
697     // Display error message
698     System.out.println("Error Code: " + eErrorCode);
699     System.out.println("Error Message: " + eErrorMessage);
700     System.out.println("Failed to add Assembly Expenditure record. Returning to the main menu.");
701     break; // Terminate the loop or take appropriate action
702 }
703
704 System.out.println("Assembly Account and Expenditure records added successfully.");
705 } catch (SQLException e) {
706     // Handle SQL exceptions for assembly account statement
707     eErrorCode = 50018; // Custom error code for assembly account
708     eErrorMessage = e.getMessage();
709
710     // Display error message
711     System.out.println("Error Code: " + eErrorCode);
712     System.out.println("Error Message: " + eErrorMessage);
713     System.out.println("Failed to add Assembly Account record. Returning to the main menu.");
714     break; // Terminate the loop or take appropriate action
715 }
716
717 break;
718
719 case 2:
720     try (CallableStatement departmentAccountStatement = connection.prepareCall(INSERT_DEPARTMENT_ACCOUNT)) {
721         departmentAccountStatement.setInt(1, accountNumber);
722         departmentAccountStatement.setString(2, dateEstablished);
723         departmentAccountStatement.setBigDecimal(3, accountDetails);
724
725         departmentAccountStatement.registerOutParameter(4, Types.INTEGER); // Assuming this is an INTEGER output parameter
726         departmentAccountStatement.registerOutParameter(5, Types.NVARCHAR);
727
728         System.out.println("Dispatching the stored procedure for Department Account...");
729         departmentAccountStatement.execute();
730
731         eErrorCode = departmentAccountStatement.getInt(4);
732         eErrorMessage = departmentAccountStatement.getString(5);
733
734         // Check for errors in the department account addition
735         if (eErrorCode != 0) {
736             // Error occurred, display error message
737             System.out.println("Error Code: " + eErrorCode);
738             System.out.println("Error Message: " + eErrorMessage);
739             System.out.println("Failed to add Department Account record. Returning to the main menu.");
740             break; // Terminate the loop or take appropriate action
741         }
742     }

```

```

741 // Now, prompt the user for expenditure details
742 System.out.println("Enter Department Number for Expenditure:");
743 int departmentNumber = sc.nextInt();
744
745 // Insert expenditure details into the department_expenditure table
746 try (CallableStatement departmentExpenditureStatement = connection.prepareCall(INSERT_DEPARTMENT_EXPENDITURE)) {
747     departmentExpenditureStatement.setInt(1, accountNumber);
748     departmentExpenditureStatement.setInt(2, departmentNumber);
749
750     departmentExpenditureStatement.registerOutParameter(3, Types.INTEGER);
751     departmentExpenditureStatement.registerOutParameter(4, Types.NVARCHAR);
752
753     departmentExpenditureStatement.execute();
754
755     int deErrorCode = departmentExpenditureStatement.getInt(3);
756     String deErrorMessage = departmentExpenditureStatement.getString(4);
757
758     // Check for errors in the department expenditure addition
759     if (deErrorCode != 0) {
760         // Error occurred, display error message
761         System.out.println("Error Code: " + deErrorCode);
762         System.out.println("Error Message: " + deErrorMessage);
763         System.out.println("Failed to add Department Expenditure record. Returning to the main menu.");
764         break; // Terminate the loop or take appropriate action
765     }
766 } catch (SQLException e) {
767     // Handle SQL exceptions for department expenditure statement
768     eErrorCode = 50022; // Custom error code for department expenditure
769     eErrorMessage = e.getMessage();
770
771     // Display error message
772     System.out.println("Error Code: " + eErrorCode);
773     System.out.println("Error Message: " + eErrorMessage);
774     System.out.println("Failed to add Department Expenditure record. Returning to the main menu.");
775     break; // Terminate the loop or take appropriate action
776 }
777
778 System.out.println("Department Account and Expenditure records added successfully.");
779 } catch (SQLException e) {
780     // Handle SQL exceptions for department account statement
781     eErrorCode = 50019; // Custom error code for department account
782     eErrorMessage = e.getMessage();
783
784     // Display error message
785     System.out.println("Error Code: " + eErrorCode);
786     System.out.println("Error Message: " + eErrorMessage);
787     System.out.println("Failed to add Department Account record. Returning to the main menu.");
788     break; // Terminate the loop or take appropriate action
789 }
790
791 break;
792
793 case 3:
794     try (CallableStatement processAccountStatement = connection.prepareCall(INSERT_PROCESS_ACCOUNT)) {
795         processAccountStatement.setInt(1, accountNumber);
796         processAccountStatement.setString(2, dateEstablished);
797         processAccountStatement.setBigDecimal(3, accountDetails);
798
799         processAccountStatement.registerOutParameter(4, Types.INTEGER);
800         processAccountStatement.registerOutParameter(5, Types.NVARCHAR);

```

```

801         System.out.println("Dispatching the stored procedure for Process Account...");
802         processAccountStatement.execute();
803
804         eErrorCode = processAccountStatement.getInt(4);
805         eErrorMessage = processAccountStatement.getString(5);
806
807         // Check for errors in the process account addition
808         if (eErrorCode != 0) {
809             // Error occurred, display error message
810             System.out.println("Error Code: " + eErrorCode);
811             System.out.println("Error Message: " + eErrorMessage);
812             System.out.println("Failed to add Process Account record. Returning to the main menu.");
813             break; // Terminate the loop or take appropriate action
814         }
815
816
817         // Now, prompt the user for expenditure details
818         System.out.println("Enter Process ID for Expenditure:");
819         String processId = sc.nextLine();
820
821         // Insert expenditure details into the process_expenditure table
822         try (CallableStatement processExpenditureStatement = connection.prepareCall(INSERT_PROCESS_EXPENDITURE)) {
823             processExpenditureStatement.setInt(1, accountNumber);
824             processExpenditureStatement.setString(2, processId);
825
826             processExpenditureStatement.registerOutParameter(3, Types.INTEGER);
827             processExpenditureStatement.registerOutParameter(4, Types.NVARCHAR);
828
829             processExpenditureStatement.execute();
830
831             int peErrorCode = processExpenditureStatement.getInt(3);
832             String peErrorMessage = processExpenditureStatement.getString(4);
833
834             // Check for errors in the process expenditure addition
835             if (peErrorCode != 0) {
836                 // Error occurred, display error message
837                 System.out.println("Error Code: " + peErrorCode);
838                 System.out.println("Error Message: " + peErrorMessage);
839                 System.out.println("Failed to add Process Expenditure record. Returning to the main menu.");
840                 break; // Terminate the loop or take appropriate action
841             }
842         } catch (SQLException e) {
843             // Handle SQL exceptions for process expenditure statement
844             eErrorCode = 50023; // Custom error code for process expenditure
845             eErrorMessage = e.getMessage();
846
847             // Display error message
848             System.out.println("Error Code: " + eErrorCode);
849             System.out.println("Error Message: " + eErrorMessage);
850             System.out.println("Failed to add Process Expenditure record. Returning to the main menu.");
851             break; // Terminate the loop or take appropriate action
852         }
853
854         System.out.println("Process Account and Expenditure records added successfully.");
855     } catch (SQLException e) {
856         // Handle SQL exceptions for process account statement
857         eErrorCode = 50020; // Custom error code for process account
858         eErrorMessage = e.getMessage();
859
860         // Display error message
861         System.out.println("Error Code: " + eErrorCode);
862         System.out.println("Error Message: " + eErrorMessage);
863         System.out.println("Failed to add Process Account record. Returning to the main menu.");
864         break; // Terminate the loop or take appropriate action
865     }
866
867     default:
868         System.out.println("Invalid choice. Returning to the main menu.");
869         return;
870     }
871
872     if (eErrorCode != 0) {
873         // Display error message
874         System.out.println("Error Code: " + eErrorCode);
875         System.out.println("Error Message: " + eErrorMessage);
876         System.out.println("Failed to add account. Returning to the main menu.");
877     } else {
878         System.out.println("Account record added successfully.");
879     }
880 } catch (SQLException e) {
881     // Handle SQL exceptions for connection
882     System.out.println("Connection error: " + e.getMessage());
883 }
884
885 break;
886
887

```

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

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
887
888 // case "6": represents the user's choice to add a new job
889 case "6":
890     // Insert into Job
891     System.out.println("Please enter job number:");
892     int jobNumber = sc.nextInt();
893
894     // Ask the user to enter the date of commencement
895     String dateCommenced = getUserInputWithFormat("Please enter date the job commenced (YYYY-MM-DD):", "yyyy-MM-dd");
896
897     try (Connection connection = DriverManager.getConnection(URL)) {
898         try (CallableStatement jobStatement = connection.prepareCall(ENTER_NEW_JOB)) {
899             jobStatement.setInt(1, jobNumber);
900             jobStatement.setString(2, dateCommenced);
901
902             // Register output parameters for error code and message
903             jobStatement.registerOutParameter(3, Types.INTEGER);
904             jobStatement.registerOutParameter(4, Types.NVARCHAR);
905
906             System.out.println("Dispatching the stored procedure for Job...");
907             jobStatement.execute();
908
909             // Retrieve error code and message from the stored procedure
910             int ferrorCode = jobStatement.getInt(3);
911             String ferrorMessage= jobStatement.getString(4);
912
913             if (ferrorCode != 0) {
914                 // Error occurred, display error message
915                 System.out.println("Error Code: " + ferrorCode);
916                 System.out.println("Error Message: " + ferrorMessage);
917                 System.out.println("Failed to add Job record. Returning to the main menu.");
918             } else {
919                 System.out.println("Job record added successfully.");
920
921                 // Ask the user to choose the type of job
922                 System.out.println("Choose the type of job:");
923                 System.out.println("1. Fit Job");
924                 System.out.println("2. Paint Job");
925                 System.out.println("3. Cut Job");
926                 int jobTypeChoice = sc.nextInt();
927                 sc.nextLine(); // Consume the newline character
928
929                 // Call the procedure to insert details based on the user's choice
930                 switch (jobTypeChoice) {
931                     // Inside the main switch statement
932                     case 1:
933                         // User's choice for Fit Job
934                         // Insert into Fit Job
935                         try (Connection fconnection = DriverManager.getConnection(URL)) {
936                             try (CallableStatement fitJobStatement = fconnection.prepareCall(INSERT_FIT)) {
937                                 fitJobStatement.setInt(1, jobNumber);
938
939                                 // Register output parameters for error code and message
940                                 fitJobStatement.registerOutParameter(2, Types.INTEGER);
941                                 fitJobStatement.registerOutParameter(3, Types.NVARCHAR);
942
943                                 System.out.println("Dispatching the stored procedure for Fit Job...");
944                                 fitJobStatement.execute();
945
946                                 // Retrieve error code and message from the stored procedure
947                                 int errorCodeFitJob = fitJobStatement.getInt(2);
948                                 String errorMessageFitJob = fitJobStatement.getString(3);
949
950                         }
```

```

949
950         if (errorCodeFitJob != 0) {
951             // Error occurred, display error message
952             System.out.println("Error Code: " + errorCodeFitJob);
953             System.out.println("Error Message: " + errorMessageFitJob);
954             System.out.println("Failed to add Fit Job details. Returning to the main menu.");
955         } else {
956             System.out.println("Fit Job details added successfully.");
957
958             // Ask the user if they want to insert Manufacture details
959             System.out.println("Do you want to insert assembly_id and process_id pairs for this job? (yes/no):");
960             String userInputManufacture = sc.nextLine().toLowerCase();
961
962             if (userInputManufacture.equals("yes")) {
963                 boolean insertMoreManufacture = true;
964
965                 while (insertMoreManufacture) {
966                     // Ask the user to enter the assembly_id and process_id pair
967                     System.out.println("Please enter assembly_id:");
968                     String fassemblyId = sc.nextLine();
969
970                     System.out.println("Please enter process_id:");
971                     String processId = sc.nextLine();
972
973                     // Call the procedure to insert details into the Manufacture table
974                     try (CallableStatement insertManufactureStatement = connection.prepareCall(INSERT_MANUFACTURE_JOB)) {
975                         insertManufactureStatement.setInt(1, jobNumber);
976                         insertManufactureStatement.setString(2, fassemblyId);
977                         insertManufactureStatement.setString(3, processId);
978
979                         // Register output parameters for error code and message
980                         insertManufactureStatement.registerOutParameter(4, Types.INTEGER);
981                         insertManufactureStatement.registerOutParameter(5, Types.NVARCHAR);
982
983                         System.out.println("Dispatching the stored procedure for Manufacture...");
984                         insertManufactureStatement.execute();
985
986                         // Retrieve error code and message from the stored procedure
987                         int errorCodeManufacture = insertManufactureStatement.getInt(4);
988                         String errorMessageManufacture = insertManufactureStatement.getString(5);
989
990                         if (errorCodeManufacture != 0) {
991                             // Error occurred, display error message
992                             System.out.println("Error Code: " + errorCodeManufacture);
993                             System.out.println("Error Message: " + errorMessageManufacture);
994                             System.out.println("Failed to add Manufacture details.");
995
996                             // Ask the user if they want to try entering assembly_id and process_id again
997                             System.out.println("Do you want to try entering assembly_id and process_id again? (yes/no):");
998                             String userInputTryAgain = sc.nextLine().toLowerCase();
999
1000                            if (userInputTryAgain.equals("no")) {
1001                                insertMoreManufacture = false; // Exit the loop
1002                            }
1003                        } else {
1004                            System.out.println("Manufacture details added successfully.");
1005
1006                            // Ask the user if they want to insert more assembly_id and process_id pairs
1007                            System.out.println("Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):");
1008                            String userInputMorePairs = sc.nextLine().toLowerCase();
1009
1010                            if (userInputMorePairs.equals("no")) {
1011                                insertMoreManufacture = false; // Exit the loop
1012                            }
1013                        }
1014                    }
1015                }
1016            }
1017        }
1018    }
1019
1020    if (userInput.equals("exit")) {
1021        System.out.println("Exiting the application. Goodbye!");
1022        System.exit(0);
1023    }
1024}

```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```

1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197

    insertManufactureStatement.setString(2, fassemblyId);
    insertManufactureStatement.setString(3, processId);

    // Register output parameters for error code and message
    insertManufactureStatement.registerOutParameter(4, Types.INTEGER);
    insertManufactureStatement.registerOutParameter(5, Types.NVARCHAR);

    System.out.println("Dispatching the stored procedure for Manufacture...");
    insertManufactureStatement.execute();

    // Retrieve error code and message from the stored procedure
    int errorCodeManufacture = insertManufactureStatement.getInt(4);
    String errorMessageManufacture = insertManufactureStatement.getString(5);

    if (errorCodeManufacture != 0) {
        // Error occurred, display error message
        System.out.println("Error Code: " + errorCodeManufacture);
        System.out.println("Error Message: " + errorMessageManufacture);
        System.out.println("Failed to add Manufacture details.");

        // Ask the user if they want to try entering assembly_id and process_id again
        System.out.println("Do you want to try entering assembly_id and process_id again? (yes/no):");
        String userInputTryAgain = sc.nextLine().toLowerCase();

        if (userInputTryAgain.equals("no")) {
            insertMoreManufacture = false; // Exit the loop
        } else {
            System.out.println("Manufacture details added successfully.");
        }
    } else {
        System.out.println("Manufacture details added successfully.");

        // Ask the user if they want to insert more assembly_id and process_id pairs
        System.out.println("Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):");
        String userInputMorePairs = sc.nextLine().toLowerCase();

        if (userInputMorePairs.equals("no")) {
            insertMoreManufacture = false; // Exit the loop
        }
    }

} catch (SQLException e) {
    // Handle SQL exceptions
    System.out.println("Error: " + e.getMessage());
    System.out.println("Failed to add Manufacture details. Returning to the main menu.");
    insertMoreManufacture = false; // Exit the loop on error
}

}
break;

case 3:
    // User's choice for Cut Job
    // Insert into Cut Job
    try (Connection ffcconnection = DriverManager.getConnection(URL)) {
        try (CallableStatement cutJobStatement = ffcconnection.prepareCall(INSERT_CUT)) {
            cutJobStatement.setInt(1, jobNumber);

            // Register output parameters for error code and message
            cutJobStatement.registerOutParameter(2, Types.INTEGER);
            cutJobStatement.registerOutParameter(3, Types.NVARCHAR);

            System.out.println("Dispatching the stored procedure for Cut Job...");
            cutJobStatement.execute();

            // Retrieve error code and message from the stored procedure
            int errorCodeCutJob = cutJobStatement.getInt(2);
            String errorMessageCutJob = cutJobStatement.getString(3);

            if (errorCodeCutJob != 0) {
                // Error occurred, display error message
                System.out.println("Error Code: " + errorCodeCutJob);
                System.out.println("Error Message: " + errorMessageCutJob);
                System.out.println("Failed to add Cut Job details. Returning to the main menu.");
            } else {
                System.out.println("Cut Job details added successfully.");
            }

            // Ask the user if they want to insert Manufacture details
            System.out.println("Do you want to insert assembly_id and process_id pairs for this job? (yes/no):");
            String userInputManufacture = sc.nextLine().toLowerCase();

            if (userInputManufacture.equals("yes")) {
                boolean insertMoreManufacture = true;

                while (insertMoreManufacture) {
                    // Ask the user to enter the assembly_id and process_id pair
                    System.out.println("Please enter assembly_id:");
                    String fffassemblyId = sc.nextLine();

                    System.out.println("Please enter process_id:");
                    String processId = sc.nextLine();

                    // Call the procedure to insert details into the Manufacture table
                    try (CallableStatement insertManufactureStatement = connection.prepareCall(INSERT_MANUFACTURE_JOB)) {
                        insertManufactureStatement.setInt(1, jobNumber);
                        insertManufactureStatement.setString(2, fffassemblyId);
                        insertManufactureStatement.setString(3, processId);

                        // Register output parameters for error code and message
                        insertManufactureStatement.registerOutParameter(4, Types.INTEGER);
                        insertManufactureStatement.registerOutParameter(5, Types.NVARCHAR);

                        System.out.println("Dispatching the stored procedure for Manufacture...");
                        insertManufactureStatement.execute();

                        // Retrieve error code and message from the stored procedure
                        int errorCodeManufacture = insertManufactureStatement.getInt(4);
                        String errorMessageManufacture = insertManufactureStatement.getString(5);

                        if (errorCodeManufacture != 0) {
                            // Error occurred, display error message
                            System.out.println("Error Code: " + errorCodeManufacture);
                            System.out.println("Error Message: " + errorMessageManufacture);
                            System.out.println("Failed to add Manufacture details.");

                            // Ask the user if they want to try entering assembly_id and process_id again
                            System.out.println("Do you want to try entering assembly_id and process_id again? (yes/no):");
                            String userInputTryAgain = sc.nextLine().toLowerCase();
                        }
                    }
                }
            }
        }
    }
}

```

```

1198         if (userInputTryAgain.equals("no")) {
1199             insertMoreManufacture = false; // Exit the loop
1200         }
1201     } else {
1202         System.out.println("Manufacture details added successfully.");
1203
1204         // Ask the user if they want to insert more assembly_id and process_id pairs
1205         String userInputMorePairs = sc.nextLine().toLowerCase();
1206
1207         if (userInputMorePairs.equals("no")) {
1208             insertMoreManufacture = false; // Exit the loop
1209         }
1210     }
1211 }
1212 } catch (SQLException e) {
1213     // Handle SQL exceptions
1214     System.out.println("Error: " + e.getMessage());
1215     System.out.println("Failed to add Manufacture details. Returning to the main menu.");
1216     insertMoreManufacture = false; // Exit the loop on error
1217 }
1218 }
1219 }
1220 }
1221 } catch (SQLException e) {
1222     // Handle SQL exceptions
1223     System.out.println("Error: " + e.getMessage());
1224     System.out.println("Failed to add Cut Job details. Returning to the main menu.");
1225 }
1226 }
1227 break;
1228
1229
1230 default:
1231     System.out.println("Invalid job type choice. Returning to the main menu.");
1232     break;
1233 }
1234 }
1235 }
1236 } catch (SQLException e) {
1237     // Handle SQL exceptions
1238     System.out.println("Error: " + e.getMessage());
1239     System.out.println("Failed to add job. Returning to the main menu.");
1240 }
1241 }
1242 break;

```

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

```

1243 case "7":
1244     // Ask the user to enter the job number
1245     System.out.println("Please enter the job number:");
1246     int updateJobNumber = sc.nextInt();
1247
1248     // Loop to handle errors and ask for the correct date of completion
1249     while (true) {
1250         // Check if the job number exists and determine the job type
1251         try (Connection checkConnection = DriverManager.getConnection(URL)) {
1252             try (CallableStatement checkStatement = checkConnection.prepareCall(CHECK_JOB_EXISTENCE)) {
1253                 checkStatement.setInt(1, updateJobNumber);
1254                 checkStatement.registerOutParameter(2, Types.INTEGER);
1255
1256                 // Execute the stored procedure to check job existence
1257                 checkStatement.execute();
1258
1259                 // Retrieve the existsFlag from the stored procedure
1260                 int existsFlag = checkStatement.getInt(2);
1261
1262                 if (existsFlag != 0) {
1263                     boolean isValidDate = false;
1264                     String dateCompleted = null;
1265
1266                     while (!isValidDate) {
1267                         // Job exists, ask for date of completion
1268                         dateCompleted = getUserInputWithFormat("Please enter date the job completed (YYYY-MM-DD):", "yyyy-MM-dd");
1269
1270                         // Update the completion date for the job
1271                         try (CallableStatement updateCompletionDateStatement = checkConnection.prepareCall(UPDATE_COMPLETION_DATE)) {
1272                             updateCompletionDateStatement.setInt(1, updateJobNumber);
1273                             updateCompletionDateStatement.setString(2, dateCompleted);
1274                             updateCompletionDateStatement.registerOutParameter(3, Types.INTEGER);
1275                             updateCompletionDateStatement.registerOutParameter(4, Types.NVARCHAR);
1276
1277                             // Execute the stored procedure to update the completion date
1278                             updateCompletionDateStatement.execute();
1279
1280                             // Retrieve error code and message from the stored procedure
1281                             int errorCodeUpdateCompletionDate = updateCompletionDateStatement.getInt(3);
1282                             String errorMessageUpdateCompletionDate = updateCompletionDateStatement.getString(4);
1283
1284                             if (errorCodeUpdateCompletionDate != 0) {
1285                                 // Error occurred, display error message
1286                                 System.out.println("Error Code: " + errorCodeUpdateCompletionDate);
1287                                 System.out.println("Error Message: " + errorMessageUpdateCompletionDate);
1288                                 System.out.println("Failed to update completion date.");
1289
1290                                 // Prompt user for a correct date
1291                                 isValidDate = false;
1292                             } else {
1293                                 System.out.println("Completion date updated successfully.");
1294                                 isValidDate = true; // Exit the loop as the update was successful
1295                             }
1296                         } catch (SQLException e) {
1297                             // Handle SQL exceptions
1298                             System.out.println("Error: " + e.getMessage());
1299                             System.out.println("Failed to update completion date.");
1300                             isValidDate = false; // Prompt user for a correct date
1301                         }
1302                     }
1303                 }
1304             }
1305         }
1306     }

```

```

1304 // Determine the type of job (Fit, Paint, or Cut)
1305 String jobType;
1306 try (CallableStatement determineTypeStatement = checkConnection.prepareCall(DETERMINE_JOB_TYPE)) {
1307     determineTypeStatement.setInt(1, updateJobNumber);
1308     determineTypeStatement.registerOutParameter(2, Types.NVARCHAR);
1309
1310     // Execute the stored procedure to determine job type
1311     determineTypeStatement.execute();
1312
1313     // Retrieve the job type from the stored procedure
1314     jobType = determineTypeStatement.getString(2);
1315
1316     // Additional logic based on job type
1317     switch (jobType) {
1318         case "Fit Job":
1319             // Ask the user to enter the new fit_labor_time
1320             System.out.println("This is a Fit Job:");
1321             System.out.println("Please enter the new fit labor time (HH:mm:ss):");
1322             String fitLaborTimeString = sc.next();
1323             LocalTime fitLaborTime = LocalTime.parse(fitLaborTimeString);
1324
1325             // Call the stored procedure to update Fit Job labor time
1326             try (CallableStatement updateFitJobStatement = checkConnection.prepareCall(UPDATE_FIT_JOB_LABOR_TIME)) {
1327                 updateFitJobStatement.setInt(1, updateJobNumber);
1328                 updateFitJobStatement.setTimestamp(2, Time.valueOf(fitLaborTime));
1329                 updateFitJobStatement.registerOutParameter(3, Types.INTEGER);
1330                 updateFitJobStatement.registerOutParameter(4, Types.NVARCHAR);
1331
1332             // Execute the stored procedure to update Fit Job labor time
1333             updateFitJobStatement.execute();
1334
1335             // Retrieve error code and message from the stored procedure
1336             int errorCodeUpdateFitJob = updateFitJobStatement.getInt(3);
1337             String errorMessageUpdateFitJob = updateFitJobStatement.getString(4);
1338
1339             if (errorCodeUpdateFitJob != 0) {
1340                 // Error occurred, display error message
1341                 System.out.println("Error Code: " + errorCodeUpdateFitJob);
1342                 System.out.println("Error Message: " + errorMessageUpdateFitJob);
1343                 System.out.println("Failed to update Fit Job labor time.");
1344             } else {
1345                 System.out.println("Fit Job labor time updated successfully.");
1346             }
1347         } catch (SQLException e) {
1348             // Handle SQL exceptions
1349             System.out.println("Error: " + e.getMessage());
1350             System.out.println("Failed to update Fit Job labor time.");
1351         }
1352     }
1353     break;
1354
1355     case "Paint Job":
1356         // Ask the user to enter the new details for Paint Job
1357         System.out.println("This is a Paint Job:");
1358         System.out.println("Please enter the new color:");
1359         String paintColor = sc.next();
1360
1361         System.out.println("Please enter the new volume:");
1362         double paintVolume = sc.nextDouble();
1363
1364         System.out.println("Please enter the new paint labor time (HH:mm:ss):");
1365         String paintLaborTimeString = sc.next();
1366         LocalTime paintLaborTime = LocalTime.parse(paintLaborTimeString);
1367
1368     }
1369 }

```

```

1366 // Call the stored procedure to update Paint Job details
1367 try (CallableStatement updatePaintJobStatement = checkConnection.prepareCall(UPDATE_PAINT_JOB_DETAILS)) {
1368     updatePaintJobStatement.setInt(1, updateJobNumber);
1369     updatePaintJobStatement.setString(2, paintColor);
1370     updatePaintJobStatement.setDouble(3, paintVolume);
1371     updatePaintJobStatement.setTime(4, Time.valueOf(paintLaborTime));
1372     updatePaintJobStatement.registerOutParameter(5, Types.INTEGER);
1373     updatePaintJobStatement.registerOutParameter(6, Types.NVARCHAR);
1374
1375     // Execute the stored procedure to update Paint Job details
1376     updatePaintJobStatement.execute();
1377
1378     // Retrieve error code and message from the stored procedure
1379     int errorCodeUpdatePaintJob = updatePaintJobStatement.getInt(5);
1380     String errorMessageUpdatePaintJob = updatePaintJobStatement.getString(6);
1381
1382     if (errorCodeUpdatePaintJob != 0) {
1383         // Error occurred, display error message
1384         System.out.println("Error Code: " + errorCodeUpdatePaintJob);
1385         System.out.println("Error Message: " + errorMessageUpdatePaintJob);
1386         System.out.println("Failed to update Paint Job details.");
1387     } else {
1388         System.out.println("Paint Job details updated successfully.");
1389     }
1390 } catch (SQLException e) {
1391     // Handle SQL exceptions
1392     System.out.println("Error: " + e.getMessage());
1393     System.out.println("Failed to update Paint Job details.");
1394 }
1395
1396 break;
1397
case "Cut Job":
1398     // Ask the user to enter the new details for Cut Job
1399     System.out.println("This is a Cut Job:");
1400     System.out.println("Please enter the new type of machine:");
1401     String machineType = sc.nextLine();
1402
1403     System.out.println("Please enter the new time of machine (HH:mm:ss):");
1404     String machineTimeString = sc.nextLine();
1405     LocalTime machineTime = LocalTime.parse(machineTimeString);
1406
1407     System.out.println("Please enter the new material used:");
1408     String machineUsed = sc.nextLine();
1409
1410     System.out.println("Please enter the new cut labor time (HH:mm:ss):");
1411     String cutLaborTimeString = sc.nextLine();
1412     LocalTime cutLaborTime = LocalTime.parse(cutLaborTimeString);
1413
1414     // Call the stored procedure to update Cut Job details
1415     try (CallableStatement updateCutJobStatement = checkConnection.prepareCall(UPDATE_CUT_JOB_DETAILS)) {
1416         updateCutJobStatement.setInt(1, updateJobNumber);
1417         updateCutJobStatement.setString(2, machineType);
1418         updateCutJobStatement.setTime(3, Time.valueOf(machineTime));
1419         updateCutJobStatement.setString(4, machineUsed);
1420         updateCutJobStatement.setTime(5, Time.valueOf(cutLaborTime));
1421         updateCutJobStatement.registerOutParameter(6, Types.INTEGER);
1422         updateCutJobStatement.registerOutParameter(7, Types.NVARCHAR);
1423
1424         // Execute the stored procedure to update Cut Job details
1425         updateCutJobStatement.execute();
1426
1427
1428     // Retrieve error code and message from the stored procedure
1429     int errorCodeUpdateCutJob = updateCutJobStatement.getInt(6);
1430     String errorMessageUpdateCutJob = updateCutJobStatement.getString(7);
1431
1432     if (errorCodeUpdateCutJob != 0) {
1433         // Error occurred, display error message
1434         System.out.println("Error Code: " + errorCodeUpdateCutJob);
1435         System.out.println("Error Message: " + errorMessageUpdateCutJob);
1436         System.out.println("Failed to update Cut Job details.");
1437     } else {
1438         System.out.println("Cut Job details updated successfully.");
1439     }
1440 } catch (SQLException e) {
1441     // Handle SQL exceptions
1442     System.out.println("Error: " + e.getMessage());
1443     System.out.println("Failed to update Cut Job details.");
1444 }
1445
1446 break;
1447
default:
1448     System.out.println("Invalid job type. Returning to the main menu.");
1449     break;
1450 }
1451
1452
1453     break; // Exit the loop as the update was successful
1454 } else {
1455     // Error: Job does not exist
1456     System.out.println("Error: Job with the specified job number does not exist. Returning to the main menu.");
1457     break; // Exit the loop as the job does not exist
1458 }
1459 } catch (SQLException e) {
1460     // Handle SQL exceptions
1461     System.out.println("Error: " + e.getMessage());
1462     System.out.println("Failed to update job details. Returning to the main menu.");
1463     break; // Exit the loop in case of a SQL exception
1464 }
1465
1466 }
1467
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).

```

1468 // Prompt the user to enter transaction details for a specific case (case "8")
1469 case "8":
1470     System.out.println("Please enter transaction-no:");
1471     sc.nextLine();
1472     // read transaction-no
1473     int transaction_no = sc.nextInt();
1474
1475     System.out.println("Please enter sup-cost:");
1476     // read sup-cost
1477     double sup_cost = sc.nextDouble();
1478
1479     System.out.println("Do you want to enter assembly account? (yes/no):");
1480     String enterAssembly = sc.next().toLowerCase();
1481     int assembly_account_no = 0;
1482
1483     // Check if the user wants to enter an assembly account
1484     if (enterAssembly.equals("yes")) {
1485         System.out.println("Please enter assembly account no:");
1486         assembly_account_no = sc.nextInt();
1487     }
1488
1489     System.out.println("Do you want to enter department account? (yes/no):");
1490     String enterDepartment = sc.next().toLowerCase();
1491     int department_account_no = 0;
1492
1493     // Check if the user wants to enter a department account
1494     if (enterDepartment.equals("yes")) {
1495         System.out.println("Please enter department account no:");
1496         department_account_no = sc.nextInt();
1497     }
1498
1499     System.out.println("Do you want to enter process account? (yes/no):");
1500     String enterProcess = sc.next().toLowerCase();
1501     int process_account_no = 0;
1502
1503     // Check if the user wants to enter a process account
1504     if (enterProcess.equals("yes")) {
1505         System.out.println("Please enter process account no:");
1506         process_account_no = sc.nextInt();
1507     }
1508
1509     System.out.println("Connecting to the database...");
1510
1511     int accountsUpdatedCount = 0;
1512
1513     // Establish a database connection
1514     try (Connection connection = DriverManager.getConnection(URL)) {
1515         try (CallableStatement statement = connection.prepareCall(ENTER_NEW_TRANSACTION)) {
1516             // Set parameters for the transaction
1517             statement.setInt(1, transaction_no);
1518             statement.setDouble(2, sup_cost);
1519
1520             // Check and set assembly account if provided
1521             if (assembly_account_no != 0) {
1522                 statement.setInt(3, assembly_account_no);
1523                 accountsUpdatedCount++;
1524                 System.out.println("Updating assembly account.");
1525             } else {
1526                 statement.setNull(3, Types.INTEGER);
1527             }
1528         }
1529     }

```

```

1528
1529         // Check and set department account if provided
1530         if (department_account_no != 0) {
1531             statement.setInt(4, department_account_no);
1532             accountsUpdatedCount++;
1533             System.out.println("Updating department account.");
1534         } else {
1535             statement.setNull(4, Types.INTEGER);
1536         }
1537
1538         // Check and set process account if provided
1539         if (process_account_no != 0) {
1540             statement.setInt(5, process_account_no);
1541             accountsUpdatedCount++;
1542             System.out.println("Updating process account.");
1543         } else {
1544             statement.setNull(5, Types.INTEGER);
1545         }
1546
1547         System.out.println("Dispatching the stored procedure for Transaction...");
1548
1549         // Execute the stored procedure for the transaction
1550         boolean accountsUpdated = statement.execute();
1551
1552         // Display the number of accounts updated
1553         if (accountsUpdatedCount > 0) {
1554             System.out.println("Cost updated for " + accountsUpdatedCount + " account(s).");
1555         } else {
1556             System.out.println("No cost update.");
1557         }
1558     } catch (SQLException e) {
1559         // Handle SQL exceptions
1560         System.out.println("Error: " + e.getMessage());
1561         System.out.println("Failed to update costs. Returning to the main menu.");
1562     }
1563 } catch (SQLException e) {
1564     System.out.println("Connection error: " + e.getMessage());
1565 }
1566
1567 break;
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612

```

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

```

1570
1571         // Prompt the user to enter an assembly id to retrieve total cost (case "9")
1572         case "9":
1573             System.out.println("Enter an assembly id to retrieve total cost:");
1574             String assemblyId = sc.next();
1575
1576             // Establish a database connection
1577             try (Connection connection = DriverManager.getConnection(URL)) {
1578                 // Retrieve total cost from the Cost table using a stored procedure
1579                 try (CallableStatement costStatement = connection.prepareCall(GET_TOTAL_COST)) {
1580                     costStatement.setString(1, assemblyId);
1581                     costStatement.registerOutParameter(2, Types.DOUBLE);
1582                     costStatement.registerOutParameter(3, Types.NVARCHAR);
1583
1584                     // Execute the stored procedure to get total cost
1585                     costStatement.execute();
1586
1587                     // Retrieve total cost and error message from the stored procedure
1588                     double totalCost = costStatement.getDouble(2);
1589                     String errorMessage = costStatement.getString(3);
1590
1591                     // Display the total cost or an error message, if any
1592                     if (errorMessage != null) {
1593                         System.out.println("Error Message: " + errorMessage);
1594                     } else {
1595                         System.out.println("Total cost incurred on assembly_id " + assemblyId + ": " + totalCost);
1596                     }
1597                 } catch (SQLException e) {
1598                     // Handle SQL exceptions for the cost statement
1599                     int errorCode23 = 50025; // Custom error code
1600                     String errorMessage23 = e.getMessage();
1601                     System.out.println("Error Code: " + errorCode23);
1602                     System.out.println("Error Message: " + errorMessage23);
1603                     System.out.println("Failed to retrieve total cost. Returning to the main menu.");
1604                 }
1605             } catch (SQLException e) {
1606                 // Handle SQL exceptions for the database connection
1607                 System.out.println("Connection error: " + e.getMessage());
1608             }
1609
1610
1611
1612

```

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

```

1616 // Prompt the user to enter department number and date to retrieve total labor time (case "10")
1617 case "10":
1618     System.out.println("Enter department number:");
1619     String departmentNumberCase10 = sc.next();
1620
1621     System.out.println("Enter the date (YYYY-MM-DD):");
1622     String givenDateCase10 = sc.next();
1623
1624     // Establish a database connection
1625     try (Connection connection = DriverManager.getConnection(URL)) {
1626         // Retrieve total labor time from Process, Manufacture, and Job tables using a stored procedure
1627         try (CallableStatement totalLaborTimeStatement = connection.prepareCall(GET_TOTAL_LABOR_TIME_IN_DEPARTMENT)) {
1628             totalLaborTimeStatement.setString(1, departmentNumberCase10);
1629             totalLaborTimeStatement.setString(2, givenDateCase10);
1630             totalLaborTimeStatement.registerOutParameter(3, Types.FLOAT);
1631             totalLaborTimeStatement.registerOutParameter(4, Types.NVARCHAR);
1632
1633             // Execute the stored procedure to get total labor time
1634             totalLaborTimeStatement.execute();
1635
1636             // Retrieve total labor time and error message from the stored procedure
1637             float totalLaborTime = totalLaborTimeStatement.getFloat(3);
1638             String errorMessageCase10 = totalLaborTimeStatement.getString(4);
1639
1640             // Display the total labor time or an error message, if any
1641             if (errorMessageCase10 != null) {
1642                 System.out.println("Error Message: " + errorMessageCase10);
1643             } else {
1644                 System.out.println("Total labor time in department " + departmentNumberCase10 +
1645                     " for jobs completed on " + givenDateCase10 + ": " + totalLaborTime + " minutes");
1646             }
1647         } catch (SQLException e) {
1648             // Handle SQL exceptions for the total labor time statement
1649             int errorCodeCase10 = 50020; // Custom error code
1650             String errorMessageCase10 = e.getMessage();
1651             System.out.println("Error Code: " + errorCodeCase10);
1652             System.out.println("Error Message: " + errorMessageCase10);
1653             System.out.println("Failed to retrieve total labor time. Returning to the main menu.");
1654         }
1655     } catch (SQLException e) {
1656         // Handle SQL exceptions for the database connection
1657         System.out.println("Connection error: " + e.getMessage());
1658     }
1659     break;
1660

```

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).

```

1665     // Case 11
1666     // Prompt the user to enter assembly_id to retrieve processes (case "11")
1667     case "11":
1668         System.out.println("Enter assembly_id to retrieve processes:");
1669         String assemblyId = sc.nextInt();
1670
1671         // Establish a database connection
1672         try (Connection connection = DriverManager.getConnection(URL)) {
1673             // Retrieve processes for the given assembly_id using a stored procedure
1674             try (CallableStatement processesStatement = connection.prepareCall(GET_PROCESSES_FOR_ASSEMBLY)) {
1675                 processesStatement.setString(1, assemblyId);
1676                 processesStatement.registerOutParameter(2, Types.NVARCHAR);
1677                 processesStatement.registerOutParameter(3, Types.NVARCHAR);
1678
1679                 // Execute the stored procedure to get processes
1680                 processesStatement.execute();
1681
1682                 // Retrieve process details and error message from the stored procedure
1683                 String processDetails = processesStatement.getString(2);
1684                 String errorMessageCase11 = processesStatement.getString(3);
1685
1686                 // Display the processes or an error message, if any
1687                 if (errorMessageCase11 != null) {
1688                     System.out.println("Error Message: " + errorMessageCase11);
1689                 } else {
1690                     System.out.println("Processes for assembly_id " + assemblyId + ":\\n" + processDetails);
1691                 }
1692             } catch (SQLException e) {
1693                 // Handle SQL exceptions for the processes statement
1694                 int errorCodeCase11 = 50020; // Custom error code
1695                 String errorMessageCase11 = e.getMessage();
1696                 System.out.println("Error Code: " + errorCodeCase11);
1697                 System.out.println("Error Message: " + errorMessageCase11);
1698                 System.out.println("Failed to retrieve processes. Returning to the main menu.");
1699             }
1700         } catch (SQLException e) {
1701             // Handle SQL exceptions for the database connection
1702             System.out.println("Connection error: " + e.getMessage());
1703         }
1704     break;
1705 }

```

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

```

1707     // Case 12
1708     // Prompt the user to enter the minimum and maximum category to retrieve customers (case "12")
1709     case "12":
1710         System.out.println("Enter the minimum category:");
1711         int minCategory = sc.nextInt();
1712
1713         System.out.println("Enter the maximum category:");
1714         int maxCategory = sc.nextInt();
1715
1716         // Establish a database connection
1717         try (Connection connection = DriverManager.getConnection(URL)) {
1718             // Retrieve customers in the given category range using a stored procedure
1719             try (CallableStatement customersStatement = connection.prepareCall(GET_CUSTOMERS_IN_RANGE)) {
1720                 customersStatement.setInt(1, minCategory);
1721                 customersStatement.setInt(2, maxCategory);
1722                 customersStatement.registerOutParameter(3, Types.NVARCHAR);
1723                 customersStatement.registerOutParameter(4, Types.NVARCHAR);
1724
1725                 // Execute the stored procedure to get customers
1726                 customersStatement.execute();
1727
1728                 // Retrieve customer details and error message from the stored procedure
1729                 String customerDetails = customersStatement.getString(3);
1730                 String errorMessageCase12 = customersStatement.getString(4);
1731
1732                 // Display the customers or an error message, if any
1733                 if (errorMessageCase12 != null) {
1734                     System.out.println("Error Message: " + errorMessageCase12);
1735                 } else {
1736                     System.out.println("Customers in the category range " + minCategory + " to " + maxCategory + ":\\n" + customerDetails);
1737                 }
1738             } catch (SQLException e) {
1739                 // Handle SQL exceptions for the customers statement
1740                 int errorCodeCase12 = 50021; // Custom error code
1741                 String errorMessageCase12 = e.getMessage();
1742                 System.out.println("Error Code: " + errorCodeCase12);
1743                 System.out.println("Error Message: " + errorMessageCase12);
1744                 System.out.println("Failed to retrieve customers. Returning to the main menu.");
1745             }
1746         } catch (SQLException e) {
1747             // Handle SQL exceptions for the database connection
1748             System.out.println("Connection error: " + e.getMessage());
1749         }
1750     break;
1751 }

```

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

```

1753 // Case 13
1754     // Prompt the user to enter the range for job numbers to delete cut-jobs (case "13")
1755     case "13":
1756         System.out.println("Enter the range for job numbers (minJobNo and maxJobNo):");
1757         int minJobNo = sc.nextInt();
1758         int maxJobNo = sc.nextInt();
1759
1760         // Establish a database connection
1761         try (Connection connection = DriverManager.getConnection(URL)) {
1762             // Delete cut-jobs within the specified job number range using a stored procedure
1763             try (CallableStatement deleteCutJobsStatement = connection.prepareCall(DELETE_CUT_JOBS_IN_RANGE)) {
1764                 deleteCutJobsStatement.setInt(1, minJobNo);
1765                 deleteCutJobsStatement.setInt(2, maxJobNo);
1766                 deleteCutJobsStatement.registerOutParameter(3, Types.NVARCHAR);
1767
1768                 // Execute the stored procedure to delete cut-jobs
1769                 deleteCutJobsStatement.execute();
1770
1771                 // Retrieve error message from the stored procedure
1772                 String errorMessageCase13 = deleteCutJobsStatement.getString(3);
1773
1774                 // Display success message or an error message, if any
1775                 if (errorMessageCase13 != null) {
1776                     System.out.println("Error Message: " + errorMessageCase13);
1777                 } else {
1778                     System.out.println("Cut-jobs within the specified range have been deleted.");
1779                 }
1780             } catch (SQLException e) {
1781                 // Handle SQL exceptions for delete cut-jobs statement
1782                 int errorCodeCase13 = 50023; // Custom error code
1783                 String errorMessageCase13 = e.getMessage();
1784                 System.out.println("Error Code: " + errorCodeCase13);
1785                 System.out.println("Error Message: " + errorMessageCase13);
1786                 System.out.println("Failed to delete cut-jobs. Returning to the main menu.");
1787             }
1788         } catch (SQLException e) {
1789             // Handle SQL exceptions for the database connection
1790             System.out.println("Connection error: " + e.getMessage());
1791         }
1792     break;
1793

```

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

```

1796 // Prompt the user to enter job number and new color for a paint job (case "14")
1797 case "14":
1798     System.out.println("Enter job number for paint job:");
1799     int jobNumberw = sc.nextInt();
1800
1801     System.out.println("Enter new color:");
1802     String newColor = sc.next();
1803
1804     // Establish a database connection
1805     try (Connection connection = DriverManager.getConnection(URL)) {
1806         // Change the color of the given paint job using a stored procedure
1807         try (CallableStatement changeColorStatement = connection.prepareCall(CHANGE_PAINT_JOB_COLOR)) {
1808             changeColorStatement.setInt(1, jobNumberw);
1809             changeColorStatement.setString(2, newColor);
1810             changeColorStatement.registerOutParameter(3, Types.NVARCHAR);
1811
1812             // Execute the stored procedure to change the color
1813             changeColorStatement.execute();
1814
1815             // Retrieve error message from the stored procedure
1816             String errorMessageCase14 = changeColorStatement.getString(3);
1817
1818             // Display success message or an error message, if any
1819             if (errorMessageCase14 != null) {
1820                 System.out.println("Error Message: " + errorMessageCase14);
1821             } else {
1822                 System.out.println("Color changed successfully for paint job " + jobNumberw);
1823             }
1824         } catch (SQLException e) {
1825             // Handle SQL exceptions for changing color statement
1826             int errorCodeCase14 = 50022; // Custom error code
1827             String errorMessageCase14 = e.getMessage();
1828             System.out.println("Error Code: " + errorCodeCase14);
1829             System.out.println("Error Message: " + errorMessageCase14);
1830             System.out.println("Failed to change color. Returning to the main menu.");
1831         }
1832     } catch (SQLException e) {
1833         // Handle SQL exceptions for the database connection
1834         System.out.println("Connection error: " + e.getMessage());
1835     }
1836
1837

```

15. Import.file:

```

1840 // Prompt the user to enter the input file name for adding customers (case "15")
1841 case "15":
1842     System.out.println("Please enter the input file name:");
1843     String inputFileName = sc.nextLine();
1844
1845 // Print current working directory for debugging
1846 System.out.println("Current Directory: " + System.getProperty("user.dir"));
1847
1848 // Open the file and read customer data until the file is empty
1849 try (Scanner fileScanner = new Scanner(new File(inputFileName))) {
1850     while (fileScanner.hasNext()) {
1851         String cname15 = fileScanner.nextLine();
1852         String caddress15 = fileScanner.nextLine();
1853         int category15;
1854
1855         try {
1856             // Attempt to parse category as an integer
1857             category15 = Integer.parseInt(fileScanner.nextLine().trim());
1858         } catch (NumberFormatException e) {
1859             // Handle invalid input for category
1860             System.out.println("Error Message: Invalid input for category. Please enter a valid integer.");
1861             continue; // Move to the next iteration
1862         }
1863
1864         // Establish a database connection
1865         try (Connection connection = DriverManager.getConnection(URL)) {
1866             // Add a new customer from the file using a stored procedure
1867             try (CallableStatement statement = connection.prepareCall(ENTER_NEW_CUSTOMER)) {
1868                 statement.setString(1, cname15);
1869                 statement.setString(2, caddress15);
1870                 statement.setInt(3, category15);
1871                 statement.registerOutParameter(4, Types.INTEGER);
1872                 statement.registerOutParameter(5, Types.NVARCHAR);
1873
1874                 // Execute the stored procedure to add the customer
1875                 statement.execute();
1876
1877                 // Retrieve error code and message from the stored procedure
1878                 int errorCode = statement.getInt(4);
1879                 String errorMessage = statement.getString(5);
1880
1881                 // Display success message or an error message, if any
1882                 if (errorCode != 0) {
1883                     System.out.println("Error Code: " + errorCode);
1884                     System.out.println("Error Message: " + errorMessage);
1885                     System.out.println("Failed to add customer from file.");
1886                 } else {
1887                     System.out.println("Customer added from file: " + cname15);
1888                 }
1889             } catch (SQLException e) {
1890                 System.out.println("Error: " + e.getMessage());
1891                 System.out.println("Failed to add customer from file.");
1892             }
1893         } catch (SQLException e) {
1894             System.out.println("Connection error: " + e.getMessage());
1895         }
1896     }
1897 } catch (FileNotFoundException e) {
1898     System.out.println("File not found: " + inputFileName);
1899 }
1900
1901

```

16. export.file:

```

1904
1905 // Case 16: Export customers to a data file based on category range
1906 // Prompt the user to enter the output file name and category range for exporting customers (case "16")
1907 case "16":
1908     System.out.println("Please enter the output file name:");
1909     String outputFileName = sc.next();
1910     System.out.println("Please enter the category range (e.g., 1-5):");
1911     String categoryRange = sc.next();
1912
1913     // Parse the category range
1914     String[] range = categoryRange.split("-");
1915     int minCategoryyy = Integer.parseInt(range[0]);
1916     int maxCategoryyy = Integer.parseInt(range[1]);
1917
1918     // Establish a database connection and export customers to a file
1919     try (Connection connection = DriverManager.getConnection(URL);
1920          Statement statement = connection.createStatement();
1921          ResultSet resultSet = statement.executeQuery("SELECT cname, caddress, category FROM Customer " +
1922             "WHERE category BETWEEN " + minCategoryyy + " AND " + maxCategoryyy + " ORDER BY cname");
1923          PrintWriter writer = new PrintWriter(outputFileName)) {
1924
1925         while (resultSet.next()) {
1926             String cname16 = resultSet.getString("cname");
1927             String caddress16 = resultSet.getString("caddress");
1928             int category16 = resultSet.getInt("category");
1929
1930             // Write customer details to the output file
1931             writer.println(cname16);
1932             writer.println(caddress16);
1933             writer.println(category16);
1934
1935             System.out.println("Exported customer: " + cname16);
1936         }
1937         System.out.println("Customers exported to file: " + outputFileName);
1938     } catch (SQLException | FileNotFoundException e) {
1939         // Handle SQL and file-related exceptions
1940         System.out.println("Error: " + e.getMessage());
1941         System.out.println("Failed to export customers to file.");
1942     }
1943     break;
1944

```

17. Quit option:

```

1956
1957
1958
1959
1960
1961
    // Handle case "17" - Do nothing, the while loop will terminate upon the next iteration
case "17":
    System.out.println("Exiting!"); // Display exit message
    break;

```

Error Handling for 3 different queries:**Query 1:**

```

Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
1
Please enter customer name:
abc
Please enter customer address:
aaa
Please enter customer category:
Error Message: Invalid input for category. Please enter a valid integer.
Please select one of the options below:

```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
1  
Please enter customer name:  
A234  
Please enter customer address:  
ssdgh  
Please enter customer category:  
1  
Connecting to the database...  
Dispatching the stored procedure...  
Error Code: 50002  
Error Message: Invalid customer name. Name must be without special characters or numbers.  
Failed to add customer. Returning to the main menu.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
1  
Please enter customer name:  
abc  
Please enter customer address:  
ashj  
Please enter customer category:  
dhf  
Error Message: Invalid input for category. Please enter a valid integer.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
1  
Please enter customer name:  
Swarupa  
Please enter customer address:  
ash  
Please enter customer category:  
3  
Connecting to the database...  
Dispatching the stored procedure...  
Error Code: 2627  
Error Message: A customer with the same name or data already exists.  
Failed to add customer. Returning to the main menu.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Query 6:

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1001  
Please enter date the job commenced (YYYY-MM-DD):  
2020-01-01  
Dispatching the stored procedure for Job...  
Error Code: 2627  
Error Message: Violation of PRIMARY KEY constraint 'PK__Job__6E3281CA507D4168'. Cannot insert duplicate key in object 'dbo.Job'. The duplicate key value is (1001). Failed to add Job record. Returning to the main menu.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1990  
Please enter date the job commenced (YYYY-MM-DD):  
299  
Invalid date format. Please enter the date in yyyy-MM-dd format.  
Please enter date the job commenced (YYYY-MM-DD):
```

Query 7:

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
109  
Error: Job with the specified job number does not exist. Returning to the main menu.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1001  
Please enter date the job completed (YYYY-MM-DD):  
2000-11-11  
Error Code: 1  
Error Message: Error: Date of completion must be greater than the date of commencement.  
Failed to update completion date.  
Please enter date the job completed (YYYY-MM-DD):
```

Task 6: Java Program Execution

Testing of query 1: 5 queries

```
WELCOME TO THE JOB-SHOP ACCOUNTING DATABASE SYSTEM

Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
1
Please enter customer name:
Swarupa
Please enter customer address:
123 Main Street, Anytown, USA
Please enter customer category:
2
Connecting to the database...
Dispatching the stored procedure...
Done. Customer added.
```

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
1
Please enter customer name:
Siva
Please enter customer address:
456 Oak Avenue, Cityville, CA
Please enter customer category:
5
Connecting to the database...
Dispatching the stored procedure...
Done. Customer added.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
1  
Please enter customer name:  
Santhu  
Please enter customer address:  
789 Maple Lane, Suburbia, TX  
Please enter customer category:  
6  
Connecting to the database...  
Dispatching the stored procedure...  
Done. Customer added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
1  
Please enter customer name:  
Mallu  
Please enter customer address:  
101 Pine Road, Countryside, NY  
Please enter customer category:  
9  
Connecting to the database...  
Dispatching the stored procedure...  
Done. Customer added.
```

**UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM**

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
1
Please enter customer name:
Rupa
Please enter customer address:
202 Cedar Boulevard, Metropolis, FL
Please enter customer category:
6
Connecting to the database...
Dispatching the stored procedure...
Done. Customer added.
```

Output of query 1:

dbo.Customer X			
Create New Row Save Refresh Discard Delete Row			
<input type="text"/> Search to filter items...			
cname	caddress	category	
Malli	101 Pine Road, Countryside, NY	9	
Rupa	202 Cedar Boulevard, Metropolis, FL	6	
Santhu	789 Maple Lane, Suburbia, TX	6	
Siva	456 Oak Avenue, Cityville, CA	5	
Swarupa	123 Main Street, Anytown, USA	2	

Testing of query 2: 5 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
2  
Please enter department number:  
101  
Please enter department data:  
data1  
Connecting to the database...  
Dispatching the stored procedure...  
Done. Department added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
2  
Please enter department number:  
102  
Please enter department data:  
data2  
Connecting to the database...  
Dispatching the stored procedure...  
Done. Department added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
2  
Please enter department number:  
103  
Please enter department data:  
data3  
Connecting to the database...  
Dispatching the stored procedure...  
Done. Department added.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
2
Please enter department number:
104
Please enter department data:
data4
Connecting to the database...
Dispatching the stored procedure...
Done. Department added.
```

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
2
Please enter department number:
105
Please enter department data:
data5
Connecting to the database...
Dispatching the stored procedure...
Done. Department added.
```

Output of query 2:

dbo.Department						
		Create New Row	Save	Refresh	Discard	Delete Row
<input type="text"/> Search to filter items...						
department_no	department_data					
101	data1					
102	data2					
103	data3					
104	data4					
105	data5					

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Testing of query 3: 10 queries

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P01  
Please enter process data:  
pdata1  
Please enter department number:  
101  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
3  
Please enter Cutting Type:  
ctype1  
Please enter Machine Type:  
mtype1  
Dispatching the stored procedure for Cut Process...  
Done. Cut Process added.
```



```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P02  
Please enter process data:  
pdata2  
Please enter department number:  
102  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
1  
Please enter Fit Type:  
ftype2  
Dispatching the stored procedure for Fit Process...  
Done. Fit Process added.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P03  
Please enter process data:  
pdata3  
Please enter department number:  
103  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
2  
Please enter Paint Type:  
ptype3  
Please enter Paint Method:  
pmethod3  
Dispatching the stored procedure for Paint Process...  
Done. Paint Process added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P04  
Please enter process data:  
pdata4  
Please enter department number:  
104  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
1  
Please enter Fit Type:  
ftype4  
Dispatching the stored procedure for Fit Process...  
Done. Fit Process added.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P05  
Please enter process data:  
pdata5  
Please enter department number:  
105  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
1  
Please enter Fit Type:  
ftype5  
Dispatching the stored procedure for Fit Process...  
Done. Fit Process added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P06  
Please enter process data:  
pdata6  
Please enter department number:  
105  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
2  
Please enter Paint Type:  
ptype6  
Please enter Paint Method:  
pmethod6  
Dispatching the stored procedure for Paint Process...  
Done. Paint Process added.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P07  
Please enter process data:  
pdata7  
Please enter department number:  
105  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
2  
Please enter Paint Type:  
ptype7  
Please enter Paint Method:  
pmethod7  
Dispatching the stored procedure for Paint Process...  
Done. Paint Process added.
```



```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P08  
Please enter process data:  
pdata8  
Please enter department number:  
104  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
3  
Please enter Cutting Type:  
ctype8  
Please enter Machine Type:  
mtype8  
Dispatching the stored procedure for Cut Process...  
Done. Cut Process added.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P09  
Please enter process data:  
pdata9  
Please enter department number:  
103  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
3  
Please enter Cutting Type:  
ctype9  
Please enter Machine Type:  
mtype9  
Dispatching the stored procedure for Cut Process...  
Done. Cut Process added.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
3  
Please enter process ID:  
P10  
Please enter process data:  
pdata10  
Please enter department number:  
102  
Connecting to the database...  
Dispatching the stored procedure for Process...  
Done. Process added.  
Choose the type of process to add:  
1. Fit Process  
2. Paint Process  
3. Cut Process  
1  
Please enter Fit Type:  
ftype10  
Dispatching the stored procedure for Fit Process...  
Done. Fit Process added.
```

Outputs of query 3:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Process 

Create New Row Save Refresh Discard Delete Row

Search to filter items...

process_id	process_data	department_no
P01	pdata1	101
P02	pdata2	102
P03	pdata3	103
P04	pdata4	104
P05	pdata5	105
P06	pdata6	105
P07	pdata7	105
P08	pdata8	104
P09	pdata9	103
P10	pdata10	102

dbo.Fit_process 

Create New Row Save Refresh Discard Delete Row

Search to filter items...

process_id	fit_type
P02	ftype2
P04	ftype4
P05	ftype5
P10	ftype10

dbo.Paint_process 

Create New Row Save Refresh Discard Delete Row

Search to filter items...

process_id	paint_type	paint_method
P03	ptype3	pmethod3
P06	ptype6	pmethod6
P07	ptype7	pmethod7

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Cut_process ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

process_id	cutting_type	machine_type
P01	ctype1	mtype1
P08	ctype8	mtype8
P09	ctype9	mtype9

Testing of query 4: 10 queries

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
4
Please enter assembly ID:
A01
Please enter date ordered (YYYY-MM-DD):
2020-11-11
Please enter assembly details:
adetails1
Please enter customer name:
Siva
Dispatching the stored procedure for Assembly...
Done. Assembly record added.
Do you want to enter process for this assembly? (yes/no):
no
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit
```

4

Please enter assembly ID:

A02

Please enter date ordered (YYYY-MM-DD):

2017-10-10

Please enter assembly details:

adetails2

Please enter customer name:

Siva

Dispatching the stored procedure for Assembly...

Done. Assembly record added.

Do you want to enter process for this assembly? (yes/no):

yes

Please enter process ID for Manufacture:

P02

Dispatching the stored procedure for Manufacture...

Done. Manufacture record added.

Do you want to enter process for this assembly? (yes/no):

yes

Please enter process ID for Manufacture:

P03

Dispatching the stored procedure for Manufacture...

Done. Manufacture record added.

Do you want to enter process for this assembly? (yes/no):

no

Please select one of the options below:

```
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit
```

4

Please enter assembly ID:

A03

Please enter date ordered (YYYY-MM-DD):

2018-09-09

Please enter assembly details:

adetails3

Please enter customer name:

Siva

Dispatching the stored procedure for Assembly...

Done. Assembly record added.

Do you want to enter process for this assembly? (yes/no):

no

|

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

- (3) Enter a new Process and information related to the type:
- (4) Enter a new Assembly and associate it with one or more processes:
- (5) Enter a new Account and associate it with the one which it is applicable:
- (6) Enter a new Job:
- (7) Enter the date of completion for a job and information of its type:
- (8) Enter a new Cost Transaction and update all the affected accounts:
- (9) Retrieve the total cost incurred on an assembly-id:
- (10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
- (11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
- (12) Retrieve the customers whose category is in a given range:
- (13) Delete all cut-jobs whose job-no is in a given range:
- (14) Change the color of a given paint job:
- (15) Import: enter new customers from a data file until the file is empty:
- (16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
- (17) Quit

4

Please enter assembly ID:

A04

Please enter date ordered (YYYY-MM-DD):

2019-12-19

Please enter assembly details:

adetails4

Please enter customer name:

Swarupa

Dispatching the stored procedure for Assembly...

Done. Assembly record added.

Do you want to enter process for this assembly? (yes/no):

yes

Please enter process ID for Manufacture:

P05

Dispatching the stored procedure for Manufacture...

Done. Manufacture record added.

Do you want to enter process for this assembly? (yes/no):

yes

Please enter process ID for Manufacture:

P06

Dispatching the stored procedure for Manufacture...

Done. Manufacture record added.

Do you want to enter process for this assembly? (yes/no):

no

Please select one of the options below:

- (1) Enter a new Customer:
- (2) Enter a new Department:
- (3) Enter a new Process and information related to the type:
- (4) Enter a new Assembly and associate it with one or more processes:
- (5) Enter a new Account and associate it with the one which it is applicable:
- (6) Enter a new Job:
- (7) Enter the date of completion for a job and information of its type:
- (8) Enter a new Cost Transaction and update all the affected accounts:
- (9) Retrieve the total cost incurred on an assembly-id:
- (10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
- (11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
- (12) Retrieve the customers whose category is in a given range:
- (13) Delete all cut-jobs whose job-no is in a given range:
- (14) Change the color of a given paint job:
- (15) Import: enter new customers from a data file until the file is empty:
- (16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
- (17) Quit

4

Please enter assembly ID:

A05

Please enter date ordered (YYYY-MM-DD):

2018-11-26

Please enter assembly details:

adetails5

Please enter customer name:

Santhu

Dispatching the stored procedure for Assembly...

Done. Assembly record added.

Do you want to enter process for this assembly? (yes/no):

no

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
4  
Please enter assembly ID:  
A06  
Please enter date ordered (YYYY-MM-DD):  
2015-03-25  
Please enter assembly details:  
adetails6  
Please enter customer name:  
Santhu  
Dispatching the stored procedure for Assembly...  
Done. Assembly record added.  
Do you want to enter process for this assembly? (yes/no):  
yes  
Please enter process ID for Manufacture:  
P07  
Dispatching the stored procedure for Manufacture...  
Done. Manufacture record added.  
Do you want to enter process for this assembly? (yes/no):  
yes  
Please enter process ID for Manufacture:  
P08  
Dispatching the stored procedure for Manufacture...  
Done. Manufacture record added.  
Do you want to enter process for this assembly? (yes/no):  
yes  
Please enter process ID for Manufacture:  
P09  
Dispatching the stored procedure for Manufacture...  
Done. Manufacture record added.  
Do you want to enter process for this assembly? (yes/no):  
no
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
4  
Please enter assembly ID:  
A07  
Please enter date ordered (YYYY-MM-DD):  
2021-04-14  
Please enter assembly details:  
adetails7  
Please enter customer name:  
Mallu  
Dispatching the stored procedure for Assembly...  
Done. Assembly record added.  
Do you want to enter process for this assembly? (yes/no):  
no
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
4  
Please enter assembly ID:  
A08  
Please enter date ordered (YYYY-MM-DD):  
2018-02-03  
Please enter assembly details:  
adetails8  
Please enter customer name:  
Rupa  
Dispatching the stored procedure for Assembly...  
Done. Assembly record added.  
Do you want to enter process for this assembly? (yes/no):  
no  
  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
4  
Please enter assembly ID:  
A09  
Please enter date ordered (YYYY-MM-DD):  
2009-10-10  
Please enter assembly details:  
adetails9  
Please enter customer name:  
Malli  
Dispatching the stored procedure for Assembly...  
Done. Assembly record added.  
Do you want to enter process for this assembly? (yes/no):  
yes  
Please enter process ID for Manufacture:  
P01  
Dispatching the stored procedure for Manufacture...  
Done. Manufacture record added.  
Do you want to enter process for this assembly? (yes/no):  
yes  
Please enter process ID for Manufacture:  
P02  
Dispatching the stored procedure for Manufacture...  
Done. Manufacture record added.  
Do you want to enter process for this assembly? (yes/no):  
no
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(3) Enter a new Process and information related to the type:  

(4) Enter a new Assembly and associate it with one or more processes:  

(5) Enter a new Account and associate it with the one which it is applicable:  

(6) Enter a new Job:  

(7) Enter the date of completion for a job and information of its type:  

(8) Enter a new Cost Transaction and update all the affected accounts:  

(9) Retrieve the total cost incurred on an assembly-id:  

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

(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  

(12) Retrieve the customers whose category is in a given range:  

(13) Delete all cut-jobs whose job-no is in a given range:  

(14) Change the color of a given paint job:  

(15) Import: enter new customers from a data file until the file is empty:  

(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  

(17) Quit  

4  

Please enter assembly ID:  

A10  

Please enter date ordered (YYYY-MM-DD):  

2010-12-19  

Please enter assembly details:  

adetails10  

Please enter customer name:  

Santhu  

Dispatching the stored procedure for Assembly...  

Done. Assembly record added.  

Do you want to enter process for this assembly? (yes/no):  

yes  

Please enter process ID for Manufacture:  

P05  

Dispatching the stored procedure for Manufacture...  

Done. Manufacture record added.  

Do you want to enter process for this assembly? (yes/no):  

yes  

Please enter process ID for Manufacture:  

P07  

Dispatching the stored procedure for Manufacture...  

Done. Manufacture record added.  

Do you want to enter process for this assembly? (yes/no):  

no
```

Outputs of query 4:

```
1  SELECT assembly_id, FORMAT(date_ordered, 'yyyy-MM-dd') AS date_ordered, assembly_details, cname
2  FROM Assembly;
3
```

dbo.Assembly X Query 3 X

Run Cancel query

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

Search to filter items...

assembly_id	date_ordered	assembly_details	cname
A01	2020-11-11	adetails1	Siva
A02	2017-10-10	adetails2	Siva
A03	2018-09-09	adetails3	Siva
A04	2019-12-19	adetails4	Swarupa
A05	2018-11-26	adetails5	Santhu
A06	2015-03-25	adetails6	Santhu
A07	2021-04-14	adetails7	Malli
A08	2018-02-03	adetails8	Rupa
A09	2009-10-10	adetails9	Malli
A10	2010-12-19	adetails10	Santhu

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.manufacture 

 Create New Row  Save  Refresh  Discard  Delete Row

Search to filter items...

assembly_id	process_id	job_no
A02	P02	
A02	P03	
A04	P05	
A04	P06	
A06	P07	
A06	P08	
A06	P09	
A09	P01	
A09	P02	
A10	P05	
A10	P07	

Testing of query 5: 10 queries

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
5
Enter account number:
1111
Enter date of establishment (YYYY-MM-DD):
1990-03-27
Choose one of the following types of account:
1. Assembly Account
2. Department Account
3. Process Account
1
Enter account details:
1000.00
Dispatching the stored procedure for Assembly Account...
Enter Assembly ID for Expenditure:
A01
Assembly Account and Expenditure records added successfully.
Account record added successfully.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
2222  
Enter date of establishment (YYYY-MM-DD):  
1991-04-24  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
2  
Enter account details:  
2000.00  
Dispatching the stored procedure for Department Account...  
Enter Department Number for Expenditure:  
102  
Department Account and Expenditure records added successfully.  
Account record added successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
3333  
Enter date of establishment (YYYY-MM-DD):  
1992-01-28  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
3  
Enter account details:  
3000.00  
Dispatching the stored procedure for Process Account...  
Enter Process ID for Expenditure:  
P03  
Process Account and Expenditure records added successfully.  
Account record added successfully.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
4444  
Enter date of establishment (YYYY-MM-DD):  
1993-09-09  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
1  
Enter account details:  
4000.00  
Dispatching the stored procedure for Assembly Account...  
Enter Assembly ID for Expenditure:  
A04  
Assembly Account and Expenditure records added successfully.  
Account record added successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
5555  
Enter date of establishment (YYYY-MM-DD):  
1994-08-01  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
2  
Enter account details:  
5000.00  
Dispatching the stored procedure for Department Account...  
Enter Department Number for Expenditure:  
105  
Department Account and Expenditure records added successfully.  
Account record added successfully.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
6666  
Enter date of establishment (YYYY-MM-DD):  
1995-02-26  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
3  
Enter account details:  
6000.00  
Dispatching the stored procedure for Process Account...  
Enter Process ID for Expenditure:  
P06  
Process Account and Expenditure records added successfully.  
Account record added successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
5  
Enter account number:  
7777  
Enter date of establishment (YYYY-MM-DD):  
1996-07-06  
Choose one of the following types of account:  
1. Assembly Account  
2. Department Account  
3. Process Account  
1  
Enter account details:  
7000.00  
Dispatching the stored procedure for Assembly Account...  
Enter Assembly ID for Expenditure:  
A07  
Assembly Account and Expenditure records added successfully.  
Account record added successfully.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit
```

5

Enter account number:

8888

Enter date of establishment (YYYY-MM-DD):

1997-06-07

Choose one of the following types of account:

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

1

Enter account details:

8000.00

Dispatching the stored procedure for Assembly Account...|

Enter Assembly ID for Expenditure:

A07

Assembly Account and Expenditure records added successfully.

Account record added successfully.

Please select one of the options below:

```
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit
```

5

Enter account number:

9999

Enter date of establishment (YYYY-MM-DD):

1998-12-11

Choose one of the following types of account:

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

3

Enter account details:

9000.00

Dispatching the stored procedure for Process Account...|

Enter Process ID for Expenditure:

P03

Process Account and Expenditure records added successfully.

Account record added successfully.

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) quit
5
Enter account number:
10000
Enter date of establishment (YYYY-MM-DD):
1999-11-12
Choose one of the following types of account:
1. Assembly Account
2. Department Account
3. Process Account
2
Enter account details:
10000.00
Dispatching the stored procedure for Department Account...
Enter Department Number for Expenditure:
102
Department Account and Expenditure records added successfully.
Account record added successfully.
```

Outputs of query 5:

```
1  SELECT assembly_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_1
2  FROM Assembly_account;
3
```

The screenshot shows a SQL query results page. At the top, there is a toolbar with buttons for Run, Cancel query, Save query, Export data as, and Show all. Below the toolbar, there are two tabs: Results (which is selected) and Messages. A search bar is located above the results table. The results table has three columns: assembly_account_no, date_of_establishment, and details_1. The data is as follows:

assembly_account_no	date_of_establishment	details_1
1111	1990-03-27	1000.00
4444	1993-09-09	4000.00
7777	1996-07-06	7000.00
8888	1997-06-07	8000.00

```
1  SELECT department_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_2
2  FROM Department_account;
3
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Query 1 × dbo.Department_account ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

department_account_no	date_of_establishment	details_2
2222	1991-04-24	2000.00
5555	1994-08-01	5000.00
10000	1999-11-12	10000.00

```
1 SELECT process_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_3
2 FROM Process_Account;
3
```

Query 1 × dbo.Process_Account ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

process_account_no	date_of_establishment	details_3
3333	1992-01-28	3000.00
6666	1995-02-26	6000.00
9999	1998-12-11	9000.00

Testing of query 6: 10 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(11) Retrieve the processes through which a given assembly_id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1001  
Please enter date the job commenced (YYYY-MM-DD):  
2020-01-01  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
3  
Dispatching the stored procedure for Cut Job...  
Cut Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A02  
Please enter process_id:  
P02  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A09  
Please enter process_id:  
P02  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no  
  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1002  
Please enter date the job commenced (YYYY-MM-DD):  
2019-02-02  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
2  
Dispatching the stored procedure for Paint Job...  
Paint Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A02  
Please enter process_id:  
P03  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A10  
Please enter process_id:  
P05  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) quit  
6  
Please enter job number:  
1003  
Please enter date the job commenced (YYYY-MM-DD):  
2018-03-03  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
2  
Dispatching the stored procedure for Paint Job...  
Paint Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A04  
Please enter process_id:  
P05  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A09  
Please enter process_id:  
P01  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no  
  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) quit  
6  
Please enter job number:  
1004  
Please enter date the job commenced (YYYY-MM-DD):  
2017-04-04  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
3  
Dispatching the stored procedure for Cut Job...  
Cut Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A10  
Please enter process_id:  
P07  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1005  
Please enter date the job commenced (YYYY-MM-DD):  
2016-05-05  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
1  
Dispatching the stored procedure for Fit Job...  
Fit Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A06  
Please enter process_id:  
P07  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no  
  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1006  
Please enter date the job commenced (YYYY-MM-DD):  
2015-06-06  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
1  
Dispatching the stored procedure for Fit Job...  
Fit Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A06  
Please enter process_id:  
P08  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A06  
Please enter process_id:  
P09  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1007  
Please enter date the job commenced (YYYY-MM-DD):  
2014-07-07  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
1  
Dispatching the stored procedure for Fit Job...  
Fit Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
no  
  
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1008  
Please enter date the job commenced (YYYY-MM-DD):  
2013-08-08  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
2  
Dispatching the stored procedure for Paint Job...  
Paint Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
no
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1009  
Please enter date the job commenced (YYYY-MM-DD):  
2012-09-09  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
2  
Dispatching the stored procedure for Paint Job...  
Paint Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
yes  
Please enter assembly_id:  
A04  
Please enter process_id:  
P06  
Dispatching the stored procedure for Manufacture...  
Manufacture details added successfully.  
Do you want to insert more assembly_id and process_id pairs for this job? (yes/no):  
no
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
6  
Please enter job number:  
1010  
Please enter date the job commenced (YYYY-MM-DD):  
2021-10-10  
Dispatching the stored procedure for Job...  
Job record added successfully.  
Choose the type of job:  
1. Fit Job  
2. Paint Job  
3. Cut Job  
3  
Dispatching the stored procedure for Cut Job...  
Cut Job details added successfully.  
Do you want to insert assembly_id and process_id pairs for this job? (yes/no):  
no
```

Outputs of query 6:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1  SELECT job_no, FORMAT(date_of_commencement, 'yyyy-MM-dd') AS date_of_commencement, date_of_completion
2  FROM Job;
3
```

dbo.Job × Query 2 ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

job_no	date_of_commencement	date_of_completion
1001	2020-01-01	
1002	2019-02-02	
1003	2018-03-03	
1004	2017-04-04	
1005	2016-05-05	
1006	2015-06-06	
1007	2014-07-07	
1008	2013-08-08	
1009	2012-09-09	
1010	2021-10-10	

dbo.Fit_job ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	fit_labor_time
1005	
1006	
1007	

dbo.Paint_job ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	color	volume	paint_labor_time
1002			
1003			
1008			
1009			

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Cut_job ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	type_of_machine	time_of_machine	material_used	cut_labor_time
1001				
1004				
1010				

dbo.manufacture ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

assembly_id	process_id	job_no
A02	P02	1001
A09	P02	1001
A02	P03	1002
A10	P05	1002
A04	P05	1003
A09	P01	1003
A10	P07	1004
A06	P07	1005
A06	P08	1006
A06	P09	1006
A04	P06	1009

Testing of query 7: 10 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1001  
Please enter date the job completed (YYYY-MM-DD):  
2000-11-11  
Error Code: 1  
Error Message: Error: Date of completion must be greater than the date of commencement.  
Failed to update completion date.  
Please enter date the job completed (YYYY-MM-DD):  
2021-01-01  
Completion date updated successfully.  
This is a Cut Job:  
Please enter the new type of machine:  
type1  
Please enter the new time of machine (HH:mm:ss):  
01:01:01  
Please enter the new material used:  
material1  
Please enter the new cut labor time (HH:mm:ss):  
10:10:10  
Cut Job details updated successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1002  
Please enter date the job completed (YYYY-MM-DD):  
2020-02-02  
Completion date updated successfully.  
This is a Paint Job:  
Please enter the new color:  
White  
Please enter the new volume:  
243.89  
Please enter the new paint labor time (HH:mm:ss):  
20:20:20  
Paint Job details updated successfully.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1003  
Please enter date the job completed (YYYY-MM-DD):  
2019-03-03  
Completion date updated successfully.  
This is a Paint Job:  
Please enter the new color:  
Black  
Please enter the new volume:  
4599.999  
Please enter the new paint labor time (HH:mm:ss):  
13:03:03  
Paint Job details updated successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1004  
Please enter date the job completed (YYYY-MM-DD):  
2018-04-04  
Completion date updated successfully.  
This is a Cut Job:  
Please enter the new type of machine:  
type2  
Please enter the new time of machine (HH:mm:ss):  
02:02:02  
Please enter the new material used:  
material2  
Please enter the new cut labor time (HH:mm:ss):  
20:20:20  
Cut Job details updated successfully.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1005  
Please enter date the job completed (YYYY-MM-DD):  
2017-05-05  
Completion date updated successfully.  
This is a Fit Job:  
Please enter the new fit labor time (HH:mm:ss):  
15:15:05  
Fit Job labor time updated successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1006  
Please enter date the job completed (YYYY-MM-DD):  
2016-06-06  
Completion date updated successfully.  
This is a Fit Job:  
Please enter the new fit labor time (HH:mm:ss):  
16:06:06  
Fit Job labor time updated successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1007  
Please enter date the job completed (YYYY-MM-DD):  
2015-07-07  
Completion date updated successfully.  
This is a Fit Job:  
Please enter the new fit labor time (HH:mm:ss):  
17:07:07  
Fit Job labor time updated successfully.
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1008  
Please enter date the job completed (YYYY-MM-DD):  
2014-08-08  
Completion date updated successfully.  
This is a Paint Job:  
Please enter the new color:  
Blue  
Please enter the new volume:  
421.678  
Please enter the new paint labor time (HH:mm:ss):  
18:09:09  
Paint Job details updated successfully.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
7  
Please enter the job number:  
1009  
Please enter date the job completed (YYYY-MM-DD):  
2013-09-09  
Completion date updated successfully.  
This is a Paint Job:  
Please enter the new color:  
Black  
Please enter the new volume:  
67.3  
Please enter the new paint labor time (HH:mm:ss):  
19:09:09  
Paint Job details updated successfully.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
7
Please enter the job number:
1010
Please enter date the job completed (YYYY-MM-DD):
2012-10-10
Error Code: 1
Error Message: Error: Date of completion must be greater than the date of commencement.
Failed to update completion date.
Please enter date the job completed (YYYY-MM-DD):
2023-03-03
Completion date updated successfully.
This is a Cut Job:
Please enter the new type of machine:
type3
Please enter the new time of machine (HH:mm:ss):
03:03:30
Please enter the new material used:
material3
Please enter the new cut labor time (HH:mm:ss):
13:30:30
Cut Job details updated successfully.
```

Outputs of query 7:

```
1  SELECT job_no, FORMAT(date_of_commencement, 'yyyy-MM-dd') AS date_of_commencement,
2  FORMAT(date_of_completion, 'yyyy-MM-dd') AS date_of_completion
3  from Job;
```

dbo.Job × Query 4 ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

job_no	date_of_commencement	date_of_completion
1001	2020-01-01	2021-01-01
1002	2019-02-02	2020-02-02
1003	2018-03-03	2019-03-03
1004	2017-04-04	2018-04-04
1005	2016-05-05	2017-05-05
1006	2015-06-06	2016-06-06
1007	2014-07-07	2015-07-07
1008	2013-08-08	2014-08-08
1009	2012-09-09	2013-09-09
1010	2021-10-10	2023-03-03

UJWALA VASIREDDY
 113634633
 INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Fit_job

job_no	fit_labor_time
1005	15:15:05
1006	16:06:06
1007	17:07:07

dbo.Paint_job

job_no	color	volume	paint_labor_time
1002	White	243.89	20:20:20
1003	Black	4599.999	13:03:03
1008	Blue	421.678	18:09:09
1009	Black	67.3	19:09:09

dbo.Cut_job

job_no	type_of_machine	time_of_machine	material_used	cut_labor_time
1001	type1	01:01:01	material1	10:10:10
1004	type2	02:02:02	material2	20:20:20
1010	type3	03:03:30	material3	13:30:30

Testing of query 8: 10 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
1110  
Please enter sup-cost:  
10.10  
Do you want to enter assembly account? (yes/no):  
yes  
Please enter assembly account no:  
1111  
Do you want to enter department account? (yes/no):  
yes  
Please enter department account no:  
2222  
Do you want to enter process account? (yes/no):  
no  
Connecting to the database...  
Updating assembly account.  
Updating department account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 2 account(s).
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
2220  
Please enter sup-cost:  
20.20  
Do you want to enter assembly account? (yes/no):  
no  
Do you want to enter department account? (yes/no):  
no  
Do you want to enter process account? (yes/no):  
no  
Connecting to the database...  
Dispatching the stored procedure for Transaction...  
No cost update.
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
3330  
Please enter sup-cost:  
30.30  
Do you want to enter assembly account? (yes/no):  
yes  
Please enter assembly account no:  
4444  
Do you want to enter department account? (yes/no):  
no  
Do you want to enter process account? (yes/no):  
yes  
Please enter process account no:  
3333  
Connecting to the database...  
Updating assembly account.  
Updating process account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 2 account(s).  
  
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
4440  
Please enter sup-cost:  
40.40  
Do you want to enter assembly account? (yes/no):  
yes  
Please enter assembly account no:  
7777  
Do you want to enter department account? (yes/no):  
yes  
Please enter department account no:  
5555  
Do you want to enter process account? (yes/no):  
yes  
Please enter process account no:  
6666  
Connecting to the database...  
Updating assembly account.  
Updating department account.  
Updating process account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 3 account(s).
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
5550  
Please enter sup-cost:  
50.50  
Do you want to enter assembly account? (yes/no):  
no  
Do you want to enter department account? (yes/no):  
no  
Do you want to enter process account? (yes/no):  
yes  
Please enter process account no:  
6666  
Connecting to the database...  
Updating process account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 1 account(s).
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
6660  
Please enter sup-cost:  
60.60  
Do you want to enter assembly account? (yes/no):  
yes  
Please enter assembly account no:  
8888  
Do you want to enter department account? (yes/no):  
yes  
Please enter department account no:  
5555  
Do you want to enter process account? (yes/no):  
no  
Connecting to the database...  
Updating assembly account.  
Updating department account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 2 account(s).
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
7770  
Please enter sup-cost:  
70.70  
Do you want to enter assembly account? (yes/no):  
yes  
Please enter assembly account no:  
8888  
Do you want to enter department account? (yes/no):  
no  
Do you want to enter process account? (yes/no):  
no  
Connecting to the database...  
Updating assembly account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 1 account(s).
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
8880  
Please enter sup-cost:  
80.80  
Do you want to enter assembly account? (yes/no):  
no  
Do you want to enter department account? (yes/no):  
no  
Do you want to enter process account? (yes/no):  
yes  
Please enter process account no:  
3333  
Connecting to the database...  
Updating process account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 1 account(s).
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
9990  
Please enter sup-cost:  
90.90  
Do you want to enter assembly account? (yes/no):  
no  
Do you want to enter department account? (yes/no):  
yes  
Please enter department account no:  
10000  
Do you want to enter process account? (yes/no):  
no  
Connecting to the database...  
Updating department account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 1 account(s).
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
8  
Please enter transaction-no:  
1001  
Please enter sup-cost:  
100.10  
Do you want to enter assembly account? (yes/no):  
no  
Do you want to enter department account? (yes/no):  
yes  
Please enter department account no:  
2222  
Do you want to enter process account? (yes/no):  
yes  
Please enter process account no:  
9999  
Connecting to the database...  
Updating department account.  
Updating process account.  
Dispatching the stored procedure for Transaction...  
Cost updated for 2 account(s).
```

Outputs of query 8:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Cost 

 Create New Row  Save  Refresh  Discard  Delete Row

Search to filter items...

transaction_no	sup_cost	assembly_account_no	department_account_no	process_account_no
1001	100.10		2222	9999
1110	10.10	1111	2222	
2220	20.20			
3330	30.30	4444		3333
4440	40.40	7777	5555	6666
5550	50.50			6666
6660	60.60	8888	5555	
7770	70.70	8888		
8880	80.80			3333
9990	90.90		10000	

```

1  SELECT assembly_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_1
2  FROM Assembly_account;
3

```

dbo.Assembly_account  Query 16 

 Run  Cancel query  Save query  Export data as  Show all

Results 

Search to filter items...

assembly_account_no	date_of_establishment	details_1
1111	1990-03-27	1010.10
4444	1993-09-09	4030.30
7777	1996-07-06	7040.40
8888	1997-06-07	8131.30

```

1  SELECT department_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_2
2  FROM department_account;
3

```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Query 1 × dbo.Department_account ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

department_account_no	date_of_establishment	details_2
2222	1991-04-24	2110.20
5555	1994-08-01	5101.00
10000	1999-11-12	10113.90

```
1 SELECT process_account_no, FORMAT(date_of_establishment, 'yyyy-MM-dd') AS date_of_establishment, details_3
2 FROM process_account;
3
```

Query 1 × dbo.Process_account ×

Run Cancel query Save query Export data as Show all

Results Messages

Search to filter items...

department_account_no	date_of_establishment	details_2
2222	1991-04-24	2110.20
5555	1994-08-01	5101.00
10000	1999-11-12	10113.90

Testing and output of query 9: 3 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
9
```

Enter an assembly id to retrieve total cost:

A01

Total cost incurred on assembly id A01: 10.1

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
9
```

Enter an assembly id to retrieve total cost:

A04

Total cost incurred on assembly_id A04: 30.3

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
9
```

Enter an assembly id to retrieve total cost:

A07

Total cost incurred on assembly_id A07: 171.7

Testing and output of query 10: 3 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
10  
Enter department number:  
101  
Enter the date (YYYY-MM-DD):  
2021-01-01  
Total labor time in department 101 for jobs completed on 2021-01-01: 0.0 minutes
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
10  
Enter department number:  
105  
Enter the date (YYYY-MM-DD):  
2017-05-05  
Total labor time in department 105 for jobs completed on 2017-05-05: 915.0 minutes
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
10  
Enter department number:  
103  
Enter the date (YYYY-MM-DD):  
2020-02-02  
Total labor time in department 103 for jobs completed on 2020-02-02: 1220.0 minutes
```

Testing and output of query 11: 3 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
11  
Enter assembly_id to retrieve processes:  
A10  
Processes for assembly_id A10:  
Process ID: P07, Department: data5  
Process ID: P05, Department: data5
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
11  
Enter assembly_id to retrieve processes:  
A02  
Processes for assembly_id A02:  
Process ID: P03, Department: data3  
Process ID: P02, Department: data2
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
11  
Enter assembly_id to retrieve processes:  
A06  
Processes for assembly_id A06:  
Process ID: P08, Department: data4  
Process ID: P09, Department: data3  
Process ID: P07, Department: data5
```

Testing and output of query 12: 3 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
12  
Enter the minimum category:  
3  
Enter the maximum category:  
8  
Customers in the category range 3 to 8:  
Rupa - 202 Cedar Boulevard, Metropolis, FL\nSanthu - 789 Maple Lane, Suburbia, TX\nSiva - 456 Oak Avenue, Cityville, CA
```

Color changed successfully for paint job 1002

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
12  
Enter the minimum category:  
1  
Enter the maximum category:  
4  
Customers in the category range 1 to 4:  
Swarupa - 123 Main Street, Anytown, USA
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
12  
Enter the minimum category:  
3  
Enter the maximum category:  
4  
Customers in the category range 3 to 4:  
null
```

Testing of query 13: 3 queries

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
13  
Enter the range for job numbers (minJobNo and maxJobNo):  
1002  
1004  
Cut-jobs within the specified range have been deleted.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
13  
Enter the range for job numbers (minJobNo and maxJobNo):  
1001  
1002  
Cut-jobs within the specified range have been deleted.
```

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
13  
Enter the range for job numbers (minJobNo and maxJobNo):  
1002  
1004  
Cut-jobs within the specified range have been deleted.
```

Output of query 13:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Paint_job ×

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	color	volume	paint_labor_time
1002	Silver	243.89	20:20:20
1003	Black	4599.999	13:03:03
1008	Red	421.678	18:09:09
1009	Blue	67.3	19:09:09

Testing of query 14:

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
14
Enter job number for paint job:
1002
Enter new color:
Silver
Color changed successfully for paint job 1002
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
14
Enter job number for paint job:
1008
Enter new color:
Red
Color changed successfully for paint job 1008

Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
14
Enter job number for paint job:
1009
Enter new color:
Blue
Color changed successfully for paint job 1009
```

Outputs of query 14:

dbo.Cut_job

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	type_of_machine	time_of_machine	material_used	cut_labor_time
1004	type2	02:02:02	material2	20:20:20
1010	type3	03:03:30	material3	13:30:30

dbo.Cut_job

Create New Row Save Refresh Discard Delete Row

Search to filter items...

job_no	type_of_machine	time_of_machine	material_used	cut_labor_time
1010	type3	03:03:30	material3	13:30:30

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

dbo.Cut_job				
	Create New Row	Save	Refresh	Discard
<input type="text"/> Search to filter items...				
job_no	type_of_machine	time_of_machine	material_used	cut_labor_time
No results				

Testing of query 15:

```
Please select one of the options below:
(1) Enter a new Customer:
(2) Enter a new Department:
(3) Enter a new Process and information related to the type:
(4) Enter a new Assembly and associate it with one or more processes:
(5) Enter a new Account and associate it with the one which it is applicable:
(6) Enter a new Job:
(7) Enter the date of completion for a job and information of its type:
(8) Enter a new Cost Transaction and update all the affected accounts:
(9) Retrieve the total cost incurred on an assembly-id:
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:
(12) Retrieve the customers whose category is in a given range:
(13) Delete all cut-jobs whose job-no is in a given range:
(14) Change the color of a given paint job:
(15) Import: enter new customers from a data file until the file is empty:
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:
(17) Quit
15
Please enter the input file name:
import.txt
Customer added from file: Alice Johnson
Customer added from file: Michael Rodriguez
Customer added from file: Emily Patel
Customer added from file: James Nguyen
Customer added from file: Sophia Kim
```

Output of query 15:

import	
File	Edit
Jasmine Patel	
321 Willow Lane, Citytown, State, 12345	
2	
Malik Johnson	
456 Birch Street, Villageland, State, 23456	
4	
Isabella Kim	
789 Cedar Avenue, Townsville, State, 34567	
6	
Andre Thompson	
890 Pine Lane, Suburbville, State, 45678	
7	
Sophia Rodriguez	
567 Oak Road, Hamletown, State, 56789	
8	

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

Testing of query 16:

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
16  
Please enter the output file name:  
export.txt  
Please enter the category range (like 1-5):  
5-9  
Exported customer: Emily Patel  
Exported customer: James Nguyen  
Exported customer: Malli  
Exported customer: Michael Rodriguez  
Exported customer: Rupa  
Exported customer: Santhu  
Exported customer: Siva  
Customers exported to file: export.txt
```

Output for query 16:

The screenshot shows a software window titled "export". The menu bar includes "File", "Edit", and "View". The main area displays a list of customers with their details:

Customer Name	Address	Category
Emily Patel	789 Pine Street, Unit 15, Lakeside, TX 75001	6
James Nguyen	101 Maple Lane, Building C, Harmony City, FL 33123	9
Malli	101 Pine Road, Countryside, NY	9
Michael Rodriguez	456 Oak Avenue, Suite 302, Rivertown, CA 90210	5
Rupa	202 Cedar Boulevard, Metropolis, FL	6
Santhu	789 Maple Lane, Suburbia, TX	6
Siva	456 Oak Avenue, Cityville, CA	5

Testing and output for query 17:

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
Please select one of the options below:  
(1) Enter a new Customer:  
(2) Enter a new Department:  
(3) Enter a new Process and information related to the type:  
(4) Enter a new Assembly and associate it with one or more processes:  
(5) Enter a new Account and associate it with the one which it is applicable:  
(6) Enter a new Job:  
(7) Enter the date of completion for a job and information of its type:  
(8) Enter a new Cost Transaction and update all the affected accounts:  
(9) Retrieve the total cost incurred on an assembly-id:  
(10) Retrieve the total labor time within a department for jobs completed in the department during a given date:  
(11) Retrieve the processes through which a given assembly-id has passed so far and the department responsible for each process:  
(12) Retrieve the customers whose category is in a given range:  
(13) Delete all cut-jobs whose job-no is in a given range:  
(14) Change the color of a given paint job:  
(15) Import: enter new customers from a data file until the file is empty:  
(16) Export: Retrieve the customers whose category is in a given range and output them to a data file:  
(17) Quit  
17  
Exiting!
```

Task 7: Web Database Application and its Execution

DataHandler.java:

```

1 package jsp_azure;
2 import java.sql.Connection;
3 import java.sql.ResultSet;
4 import java.sql.SQLException;
5 import java.sql.DriverManager;
6 import java.sql.PreparedStatement;
7
8 public class DataHandler {
9     private Connection conn;
10    // Azure SQL connection credentials
11    private String server = "vasi0011-sql-server.database.windows.net";
12    private String database = "cs-dsa-4513-sql-db";
13    private String username = "vasi001";
14    private String password = "*****";
15    // Resulting connection string
16    final private String url =
17        String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trustServerCertificate=false;hostNameInCertificate=%s.database.windows.net;loginTimeout=30");
18        server, database, username, password);
19    // Initialize and save the database connection
20    // Initialize and save the database connection
21    private void getDBConnection() throws SQLException {
22        if (conn != null) {
23            return;
24        }
25        this.conn = DriverManager.getConnection(url);
26    }
27
28    // Return the result of selecting everything from the Customer table
29    public ResultSet getAllCustomers() throws SQLException {
30        getDBConnection(); // Prepare the database connection
31
32        // Prepare the SQL statement
33        final String sqlQuery = "SELECT * FROM Customer";
34        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
35
36        // Execute the query
37        return stmt.executeQuery();
38    }
39
40    // Return the result of selecting Customers with their category in the given range from Customer table
41    public ResultSet retrieveCustomers(int lower_b, int upper_b) throws SQLException {
42        getDBConnection(); // Prepare the database connection
43
44        // Return the result of selecting Customers with their category in the given range from Customer table
45        public ResultSet retrieveCustomers(int lower_b, int upper_b) throws SQLException {
46            getDBConnection(); // Prepare the database connection
47
48            // Prepare the SQL statement
49            final String sqlQuery = "SELECT cname AS name, category FROM Customer" +
50                " WHERE category >= ? AND category <= ? ORDER BY 1 ";
51            final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
52
53            // Replace the '?' in the above statement with the given attribute values
54            stmt.setInt(1, lower_b);
55            stmt.setInt(2, upper_b);
56
57            // Execute the query
58            return stmt.executeQuery();
59        }
60
61        // Inserts a record into the Customer table with the given attribute values
62        public boolean addCustomer(String cname, String caddress, int category) throws SQLException {
63            getDBConnection(); // Prepare the database connection
64
65            // Prepare the SQL statement for adding a new customer
66            final String sqlQuery = "INSERT INTO Customer (cname, caddress, category) VALUES (?, ?, ?)";
67            final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
68
69            // Replace the '?' in the above statement with the given attribute values
70            stmt.setString(1, cname);
71            stmt.setString(2, caddress);
72            stmt.setInt(3, category);
73
74            // Execute the query, if only one record is updated, then we indicate success by returning true
75            return stmt.executeUpdate() == 1;
76        }
77    }
78 }
```

welcome.jsp:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4   "http://www.w3.org/TR/html4/loose.dtd">
5<html>
6<head>
7   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8   <title>Welcome to Job-Shop Accounting</title>
9<style>
10  body {
11    font-family: 'Arial', sans-serif;
12    margin: 20px;
13    padding: 20px;
14    background-color: #f4f4f4;
15    display: flex;
16    flex-direction: column;
17    align-items: center;
18    justify-content: center;
19    height: 100vh;
20  }
21
22  h1 {
23    color: #333;
24  }
25
26  .card-container {
27    display: flex;
28    justify-content: space-around;
29    margin-top: 20px;
30  }
31
32  .card {
33    width: 400px;
34    padding: 20px;
35    background-color: #eee; /* Gray background color */
36    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
37    border-radius: 8px;
38    text-align: center;
39  }
40
41  h2 {
42    color: #333;
43  }
44
45  a {
46    display: block;
47    margin: 10px 0;
48    padding: 10px;
49    background-color: #3498db;
50    color: #fff;
51    text-decoration: none;
52    text-align: center;
53    border-radius: 5px;
54    transition: background-color 0.3s;
55  }
56
57  a:hover {
58    background-color: #2980b9;
59  }
60 </style>
61 </head>
62<body>
63   <h1>Welcome to Job-Shop Accounting</h1>
64
65<div class="card-container">
66   <!-- Add New Customer Card -->
67<div class="card">
68   <h2>Add New Customer</h2>
69   <a href="add_new_customer_form.jsp">Go to Add New Customer</a>
70 </div>
71
72   <!-- Retrieve Customers Card -->
73<div class="card">
74   <h2>Retrieve Customers</h2>
75   <a href="retrieve_customer_form.jsp">Go to Retrieve Customers</a>
76 </div>
77 </div>
78 </body>
79 </html>
```

add_new_customer.jsp:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4   "http://www.w3.org/TR/html4/loose.dtd">
5<html>
6<head>
7   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8   <title>Add New Customer Result</title>
9</head>
10<body>
11   <%@page import="jsp_azure.DataHandler"%>
12   <%@page import="java.sql.ResultSet"%>
13   <%@page import="java.sql.Array"%>
14
15<%
16 // The handler is the one in charge of establishing the connection.
17 DataHandler handler1 = new DataHandler();
18
19 // Get the attribute values passed from the input form.
20 String cname = request.getParameter("cname");
21 String address = request.getParameter("address");
22 String category = request.getParameter("category");
23
24 /*
25 * If the user hasn't filled out all the cname, address and category. This is very simple checking.
26 */
27 if (cname == null || address == null || category == null || cname.trim().isEmpty() || address.trim().isEmpty() || category.trim().isEmpty()) {
28   response.sendRedirect("add_new_customer_form.jsp");
29 } else {
30   try {
31     int category1 = Integer.parseInt(category);
32
33     // Now perform the query with the data from the form.
34     boolean success = handler1.addCustomer(cname, address, category1);
35
36     if (!success) {
37       <h2>There was a problem inserting the customer.</h2>
38     } else {
39       <h2>The Customer Details:</h2>
40       <ul>
41         <li>Customer Name: <%=cname%></li>
42         <li>Address: <%=address%></li>
43         <li>Category: <%=category1%></li>
44       </ul>
45       <h2>Was successfully inserted.</h2>
46       <a href="get_all_customers.jsp">See all Customers.</a>
47     }
48   } catch (NumberFormatException e) {
49     <h2>Error: Invalid input for category. Please enter a valid integer.</h2>
50     <a href="add_new_customer_form.jsp">Go back to the form</a>
51   }
52 }
53 <%
54 <%
55 <%
56 <%
57 <%
58 <%
59 <%
60 </body>
61 </html>
```

add_new_customer_forms.jsp:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1 <!DOCTYPE html>
2<html>
3<head>
4     <meta charset="UTF-8">
5     <title>Add Customer</title>
6<style>
7     body {
8         font-family: Arial, sans-serif;
9         display: flex;
10        justify-content: center;
11        align-items: center;
12        height: 100vh;
13        margin: 0;
14        background-color: #f4f4f4;
15    }
16
17    .card {
18        width: 70%;
19        background-color: #fff;
20        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
21        border-radius: 8px;
22        padding: 20px;
23    }
24
25    h2 {
26        text-align: center;
27    }
28
29    table {
30        margin: auto;
31        border-collapse: collapse;
32        width: 100%;
33    }
34
35    th, td {
36        padding: 10px;
37        text-align: left;
38        border-bottom: 1px solid #ddd;
39    }
40
41    input[type="text"] {
42        width: calc(100% - 16px);
43        padding: 8px;
44        box-sizing: border-box;
45    }
46
47    input[type="reset"], input[type="submit"] {
48        padding: 10px;
49        width: calc(50% - 6px);
50        box-sizing: border-box;
51    }
52
53    input[type="reset"] {
54        background-color: #f44336;
55        color: white;
56        border: none;
57    }
58
59    input[type="submit"] {
60        background-color: #4CAF50;
61        color: white;
62        border: none;
63    }
64
65    div {
66        text-align: center;
67    }
68
69    .error {
70        color: #f44336;
71    }
72</style>
73<body>
74    <div class="card">
75        <h2>Add Customer</h2>
76
77        <!-- Form for collecting user input for the new customer record. -->
78        <form action="add_new_customer.jsp" method="post">
79            <!-- The form organized in an HTML table for better clarity. -->
80            <table>
81                <tr>
82                    <th colspan="2">Enter the Customer Data:</th>
83                </tr>
84                <tr>
85                    <td>
```

UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
| 85          <tr>
86              <td>Customer Name:</td>
87              <td>
88                  <div>
89                      <input type="text" name="cname" required>
90                  </div>
91              </td>
92          </tr>
93          <tr>
94              <td>Customer Address:</td>
95              <td>
96                  <div>
97                      <input type="text" name="address" required>
98                  </div>
99              </td>
100         </tr>
101         <tr>
102             <td>Category:</td>
103             <td>
104                 <div>
105                     <input type="text" name="category" required>
106                 </div>
107             </td>
108         </tr>
109         <tr>
110             <td colspan="2">
111                 <div>
112                     <input type="reset" value="Clear">
113                     <input type="submit" value="Insert">
114                 </div>
115             </td>
116         </tr>
117     </table>
118 </form>
119
120     <!-- Display error messages, if any -->
121     <%
122         String errorMessage = (String)request.getAttribute("errorMessage");
123         if (errorMessage != null && !errorMessage.isEmpty()) {
124             %>
125                 <div class="error"><%=errorMessage%></div>
126             <%
127
128                 <td colspan="2">
129                     <div>
130                         <input type="reset" value="Clear">
131                         <input type="submit" value="Insert">
132                     </div>
133                 </td>
134             </tr>
135         </table>
136     </form>
137
138     <!-- Display error messages, if any -->
139     <%
140         String errorMessage = (String)request.getAttribute("errorMessage");
141         if (errorMessage != null && !errorMessage.isEmpty()) {
142             %>
143                 <div class="error"><%=errorMessage%></div>
144             <%
145         }
146     <%
147
148     </div>
149 </body>
150 </html>
151 |
```

get_all_customers.jsp:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <meta charset="UTF-8">
6     <title>Customers</title>
7     <style>
8         body {
9             font-family: Arial, sans-serif;
10            display: flex;
11            justify-content: center;
12            align-items: center;
13            height: 100vh;
14            margin: 0;
15            background-color: #f4f4f4;
16        }
17
18        .card {
19            width: 70%;
20            background-color: #fff;
21            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
22            border-radius: 8px;
23            padding: 20px;
24        }
25
26        h4 {
27            margin: 0;
28        }
29
30        table {
31            margin: auto;
32            border-collapse: collapse;
33            width: 100%;
34            margin-top: 20px;
35        }
36
37        th, td {
38            padding: 10px;
39            text-align: left;
40            border-bottom: 1px solid #ddd;
41        }
42     </style>
43 </head>
44 <body>
45     <%@page import="jsp.azure.DataHandler"%>
46     <%@page import="java.sql.ResultSet"%>
47
48     <%
49         // We instantiate the data handler here, and get all the customers from the database
50         final DataHandler handler = new DataHandler();
51         final ResultSet customers = handler.getAllCustomers();
52     %>
53
54     <div class="card">
55         <h2>All Customers</h2>
56
57         <!-- The table for displaying all the customer records -->
58         <table>
59             <tr> <!-- The table headers row -->
60                 <th align="center"><h4>Customer</h4></th>
61                 <th align="center"><h4>Address</h4></th>
62                 <th align="center"><h4>Category</h4></th>
63             </tr>
64
65             <%
66                 while (customers.next()) { // For each Customer record returned...
67                     // Extract the attribute values for every row returned
68                     final String cname = customers.getString("cname");
69                     final String caddress = customers.getString("caddress");
70                     final String category = customers.getString("category");
71
72                     out.println("<tr>"); // Start printing out the new table row
73                     out.println("// Print each attribute value
74                         "<td align="center">" + cname +
75                         "</td><td align="center">" + caddress +
76                         "</td><td align="center">" + category + "</td>"); // Print each attribute value
77                     out.println("</tr>"); // End printing out the new table row
78                 }
79             %>
80         </table>
81     </div>
82 </body>
83 </html>
```

retrieve_customers.jsp

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <meta charset="UTF-8">
6     <title>Customers</title>
7     <style>
8         body {
9             font-family: Arial, sans-serif;
10        }
11
12        h4 {
13            margin: 0;
14        }
15
16        table {
17            margin: auto;
18            border-collapse: collapse;
19            width: 50%;
20            margin-top: 20px;
21        }
22
23        th, td {
24            padding: 10px;
25            text-align: left;
26            border-bottom: 1px solid #ddd;
27        }
28    </style>
29 </head>
30 <body>
31     <%@page import="jsp_azure.DataHandler"%>
32     <%@page import="java.sql.ResultSet"%>
33
34     <%
35     // We instantiate the data handler here and get customers from the database
36     final DataHandler handler = new DataHandler();
37
38     // Get the attribute values passed from the input form.
39     String lower_b = request.getParameter("lower_b");
40     String upper_b = request.getParameter("upper_b");
41
42     if (lower_b == null || upper_b == null || lower_b.trim().isEmpty() || upper_b.trim().isEmpty()) {
43         response.sendRedirect("retrieve_customer_form.jsp");

```

```

43         response.sendRedirect("retrieve_customer_form.jsp");
44     } else {
45         try {
46             int lower_b1 = Integer.parseInt(lower_b);
47             int upper_b1 = Integer.parseInt(upper_b);
48
49             // Now perform the query with the data from the form.
50             final ResultSet customers = handler.retrieveCustomers(lower_b1, upper_b1);
51         %}
52
53         <!-- The table for displaying all the Customer records -->
54         <table>
55             <tr> <!-- The table headers row -->
56                 <th align="center"><h4>Customer</h4></th>
57                 <th align="center"><h4>Category</h4></th>
58             </tr>
59
60             <%
61             while (customers.next()) { // For each Customer record returned...
62                 // Extract the attribute values for every row returned
63                 final String cname = customers.getString("name");
64                 final String category = customers.getString("category");
65
66                 out.println("<tr>"); // Start printing out the new table row
67                 out.println(" // Print each attribute value
68                     <td align="center">" + cname +
69                     "</td><td align="center"> " + category + "</td>");
70                 out.println("</tr>");
71             }
72         }
73         catch (NumberFormatException e) {
74             // Handle the case where the input is not a valid integer
75         %}
76         <h2>Error: Invalid input for category range. Please enter valid integers.</h2>
77         <a href="retrieve_customer_form.jsp">Go back to the form</a>
78     <%
79     }
80     %
81     </table>
82 </body>
83 </html>
84 
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

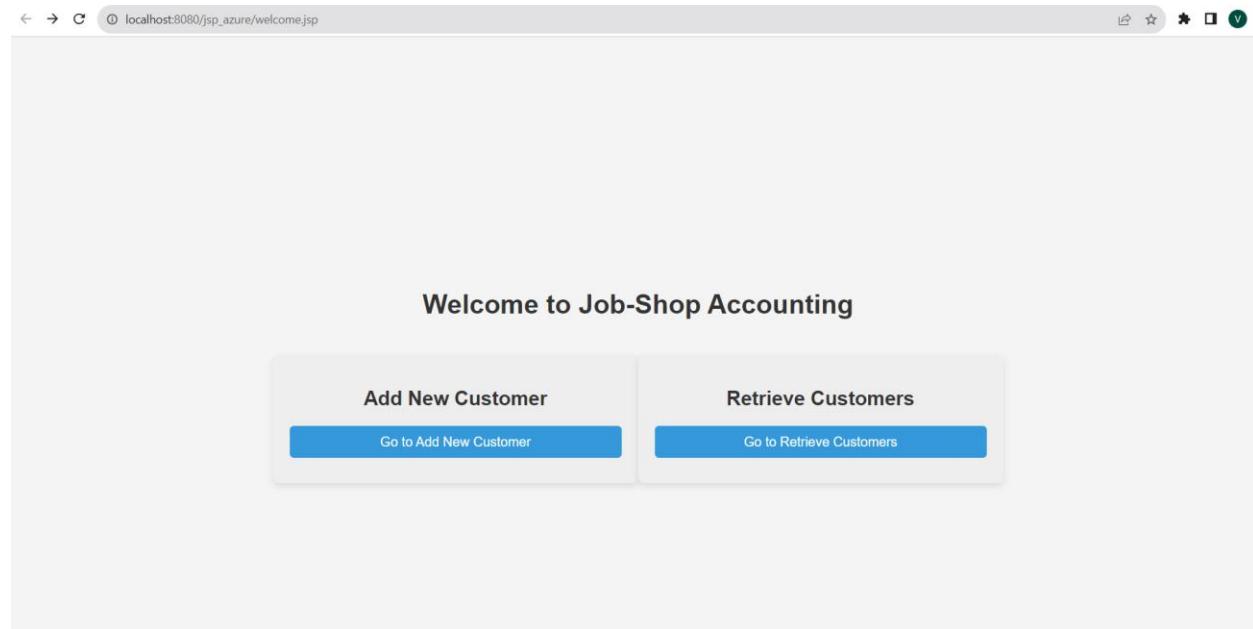
retrieve_customer_form.jsp

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Retrieve Customers Given Category Range</title>
6     <style>
7         body {
8             font-family: Arial, sans-serif;
9             display: flex;
10            justify-content: center;
11            align-items: center;
12            height: 100vh;
13            margin: 0;
14            background-color: #f4f4f4;
15        }
16
17        .card {
18            width: 50%;
19            background-color: #fff;
20            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
21            border-radius: 8px;
22            padding: 20px;
23        }
24
25        h2 {
26            text-align: center;
27        }
28
29        table {
30            margin: auto;
31            border-collapse: collapse;
32            width: 100%;
33            margin-top: 20px;
34        }
35
36        th, td {
37            padding: 10px;
38            text-align: left;
39            border-bottom: 1px solid #ddd;
40        }
41
42        input[type="text"] {
43            width: calc(100% - 16px);
44
45            width: calc(100% - 16px);
46            padding: 8px;
47            box-sizing: border-box;
48        }
49
50        input[type="reset"], input[type="submit"] {
51            padding: 10px;
52            width: calc(50% - 6px);
53            box-sizing: border-box;
54        }
55
56        input[type="reset"] {
57            background-color: #f44336;
58            color: white;
59            border: none;
60        }
61
62        input[type="submit"] {
63            background-color: #4CAF50;
64            color: white;
65            border: none;
66        }
67
68        div {
69            text-align: center;
70        }
71    </style>
72 </head>
73 <body>
74     <div class="card">
75         <h2>Retrieve Customers Given Category Range</h2>
76
77         <!-- Form for collecting user input for retrieving customer records -->
78         <form action="retrieve_customers.jsp" method="post">
79             <!-- The form organized in an HTML table for better clarity. -->
80             <table>
81                 <tr>
82                     <th colspan="2">Enter the Category Range:</th>
83                 </tr>
84                 <tr>
85                     <td>Lower Bound:</td>
86                     <td>
87                         <div>
```

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

```
83         <td>Lower Bound:</td>
84         <td>
85             <div>
86                 <input type="text" name="lower_b" required>
87             </div>
88         </td>
89     </tr>
90     <tr>
91         <td>Upper Bound:</td>
92         <td>
93             <div>
94                 <input type="text" name="upper_b" required>
95             </div>
96         </td>
97     </tr>
98     <tr>
99         <td colspan="2">
100            <div>
101                <input type="reset" value="Clear">
102                <input type="submit" value="Retrieve">
103            </div>
104        </td>
105    </tr>
106  </table>
107 </form>
108 </div>
109 </body>
110 </html>
```

Outputs of the web application:



UJWALA VASIREDDY

113634633

INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

The screenshot shows a web browser window with the URL `localhost:8080/jsp_azure/retrieve_customer_form.jsp`. The page title is "Retrieve Customers Given Category Range". Below the title, there is a form with two input fields: "Lower Bound" containing the value "1" and "Upper Bound" containing the value "9". There are two buttons at the bottom: a red "Clear" button and a green "Retrieve" button.

The screenshot shows a web browser window with the URL `localhost:8080/jsp_azure/retrieve_customers.jsp`. The page displays a table with two columns: "Customer" and "Category". The data rows are:

Customer	Category
Alice Johnson	4
Emily Patel	6
James Nguyen	9
Malli	9
Michael Rodriguez	5
Rupa	6
Santhu	6
Siva	5
Sophia Kim	1
Swarupa	2

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

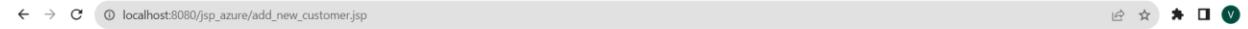
localhost:8080/jsp_azure/add_new_customer_form.jsp

Add Customer

Enter the Customer Data:

Customer Name:	Mehedi
Customer Address:	West Bengal
Category:	7

Clear **Insert**



The Customer Details:

- Customer Name: Mehedi
- Address: West Bengal
- Category: 7

Was successfully inserted.

[See all Customers.](#)

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

localhost:8080/jsp_azure/get_all_customers.jsp

Customer	Address	Category
Alice Johnson	123 Main Street, Apt 4B, Springfield, IL 62701	4
Emily Patel	789 Pine Street, Unit 15, Lakeside, TX 75001	6
James Nguyen	101 Maple Lane, Building C, Harmony City, FL 33123	9
Malli	101 Pine Road, Countryside, NY	9
Mehedi	West Bengal	7
Michael Rodriguez	456 Oak Avenue, Suite 302, Rivertown, CA 90210	5
Rupa	202 Cedar Boulevard, Metropolis, FL	6
Santhu	789 Maple Lane, Suburbia, TX	6
Siva	456 Oak Avenue, Cityville, CA	5
Sophia Kim	543 Elm Street, Floor 8, Mountainview, NY 10022	1
Swarupa	123 Main Street, Anytown, USA	2

localhost:8080/jsp_azure/retrieve_customer_form.jsp

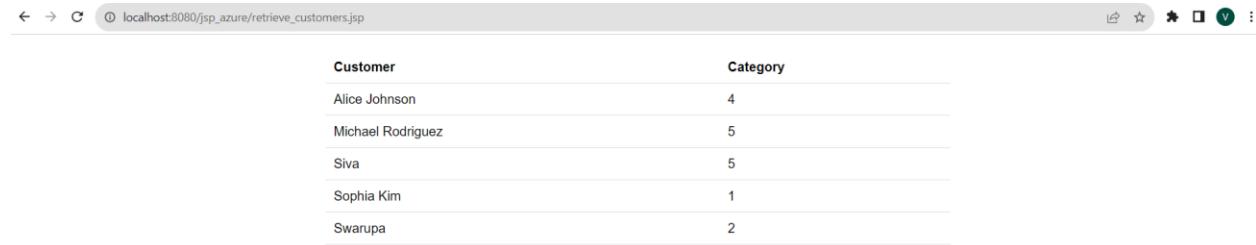
Retrieve Customers Given Category Range

Enter the Category Range:

Lower Bound:

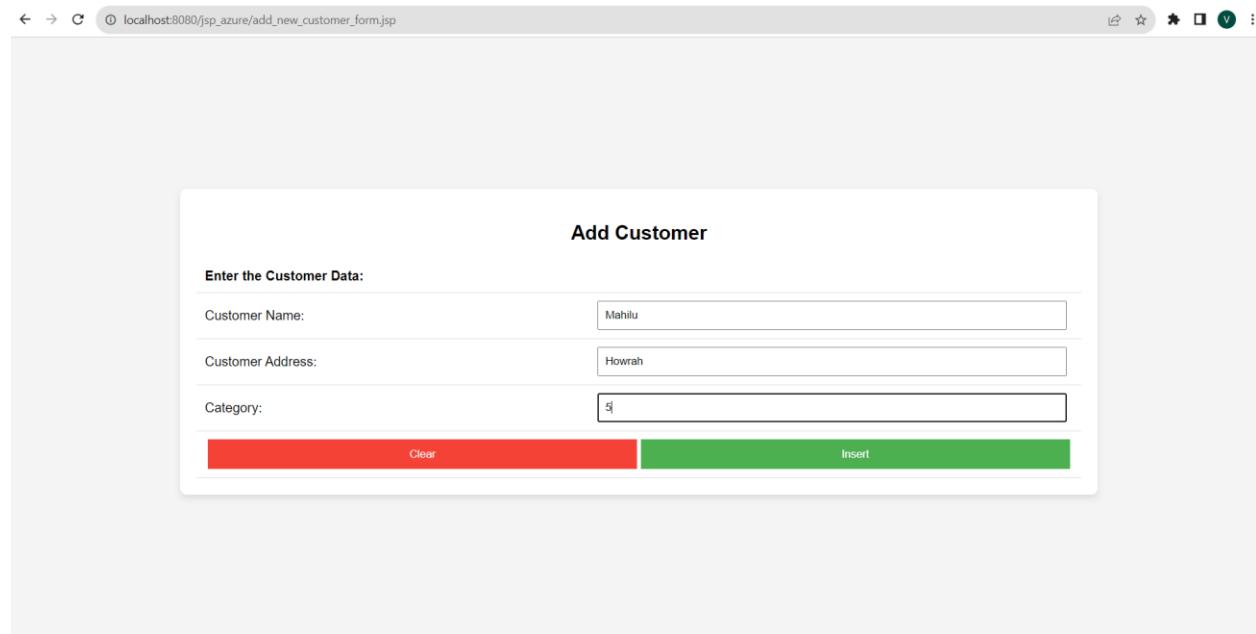
Upper Bound:

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM



A screenshot of a web browser window displaying a table of customer data. The table has two columns: 'Customer' and 'Category'. The data rows are:

Customer	Category
Alice Johnson	4
Michael Rodriguez	5
Siva	5
Sophia Kim	1
Swarupa	2



A screenshot of a web browser window showing an 'Add Customer' form. The form is titled 'Add Customer' and contains the following fields:

Enter the Customer Data:

Customer Name:	<input type="text" value="Mahilu"/>
Customer Address:	<input type="text" value="Howrah"/>
Category:	<input type="text" value="5"/>

At the bottom of the form are two buttons: 'Clear' (red background) and 'Insert' (green background).

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

← → ⌂ localhost:8080/jsp_azure/add_new_customer.jsp

The Customer Details:

- Customer Name: Mahilu
 - Address: Howrah
 - Category: 5

Was successfully inserted.

[See all Customers.](#)

All Customers

Customer	Address	Category
Alice Johnson	123 Main Street, Apt 4B, Springfield, IL 62701	4
Emily Patel	789 Pine Street, Unit 15, Lakeside, TX 75001	6
James Nguyen	101 Maple Lane, Building C, Harmony City, FL 33123	9
Mahilu	Howrah	5
Malli	101 Pine Road, Countryside, NY	9
Mehedi	West Bengal	7
Michael Rodriguez	456 Oak Avenue, Suite 302, Rivertown, CA 90210	5
Rupa	202 Cedar Boulevard, Metropolis, FL	6
Santhu	789 Maple Lane, Suburbia, TX	6
Siva	456 Oak Avenue, Cityville, CA	5
Sophia Kim	543 Elm Street, Floor 8, Mountainview, NY 10022	1
Swarupa	123 Main Street, Anytown, USA	2

UJWALA VASIREDDY
113634633
INDIVIDUAL PROJECT: A JOB-SHOP ACCOUNTING SYSTEM

The screenshot shows a web browser window with the URL `localhost:8080/jsp_azure/retrieve_customer_form.jsp`. The page contains a central modal dialog with the title "Retrieve Customers Given Category Range". Inside the dialog, there are two input fields: "Lower Bound" with the value "2" and "Upper Bound" with the value "1". Below the inputs are two buttons: a red "Clear" button and a green "Retrieve" button.

The screenshot shows a web browser window with the URL `localhost:8080/jsp_azure/retrieve_customers.jsp`. The page displays a table with two columns: "Customer" and "Category". The data rows are:

Customer	Category
Alice Johnson	4
Emily Patel	6
Mahilu	5
Mehedi	7
Michael Rodriguez	5
Rupa	6
Santhu	6
Siva	5
Swarupa	2