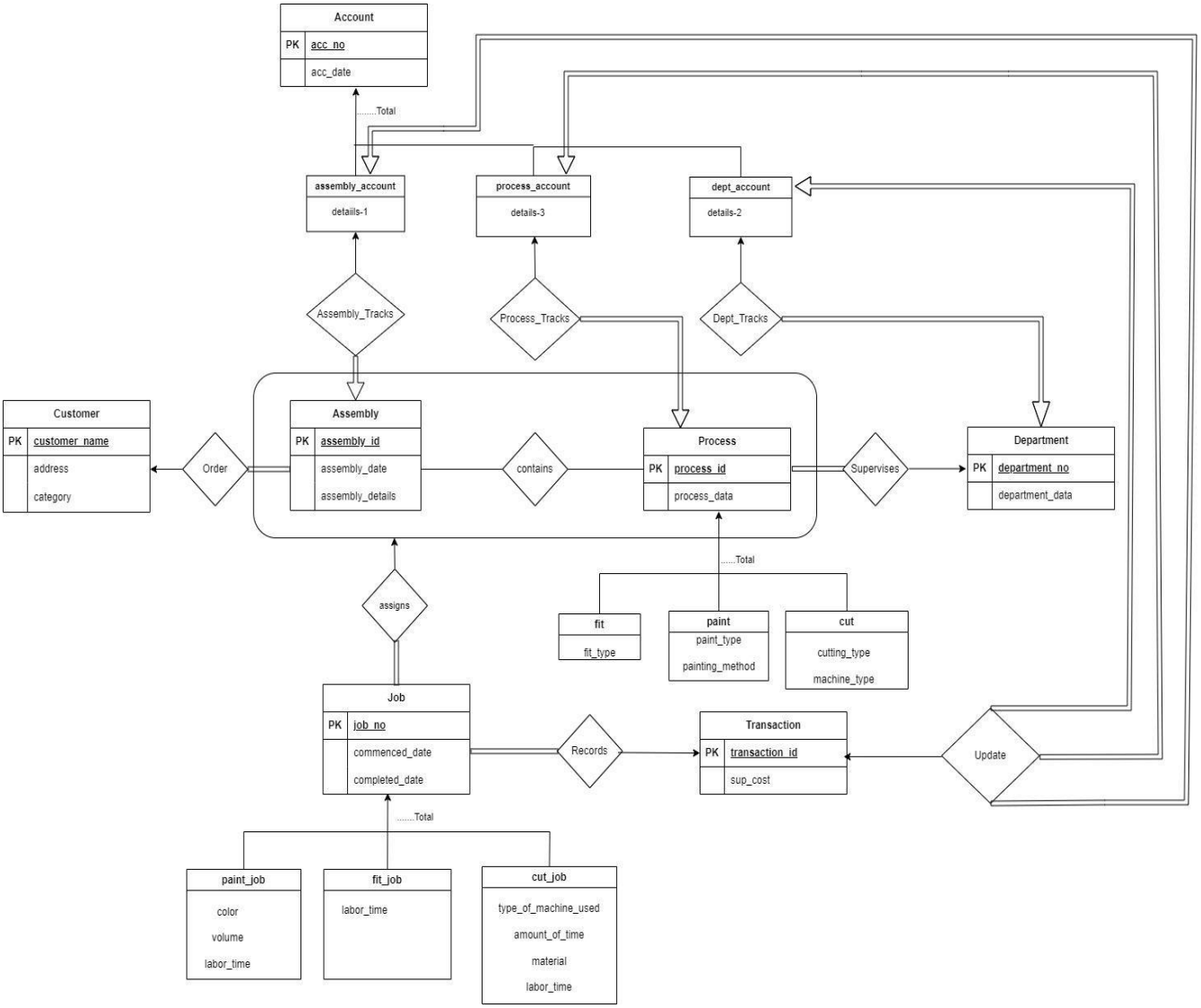
PROJECT TITLE: A JOB SHOP ACCOUNTING SYSTEM

Author's Name: Aishwarya Peri

TASK-1: ER Diagram



TASK-2: Relational Database Schema

=

1. Customer (**customer_name**, address, category)
2. Assembly(**assembly_id**, assembly_dates, assembly_details)
3. Process(**process_id**, process_data)
4. fit(**process_id** ,fit_type)
5. paint(**process_id** ,paint_type, painting_method)
6. cut(**process_id** ,cutting_type, machine_type)
7. Department(**department_no**, Department_data)
8. Account(**acc_no**, acc_date)
9. Assembly_account(**acc_no** ,details-1)
10. Process_account(**acc_no** ,details-2)
11. Dept_account(**acc_no** ,details-3)
12. Job(**job_no**, commenced_date, completed_date)
13. paint_job(**job_no** ,color, volume, labor_time)
14. Fit_job(**job_no** ,labor_time)
15. Cut_job(**job_no** ,type_of_machine_used, amount_of_time, material, labor_time)
16. Transaction(**transaction_id**, sup_cost)
17. Order(customer_name, **assembly_id**)
18. Contains(**assembly_id**, **process_id**)
19. Supervises(**process_id**, department_no)
20. Assigns(assembly_id, **job_no**)
21. Assembly_tracks(acc_no, **assembly_id**)
22. Process_tracks(acc_no, **process_id**)
23. Dept_tracks(acc_no, **department_no**)

TASK 3

3.1 Discussion of storage structures for tables

Table Name	Query# and Type	Search Key	Query Frequency	Selected File Organization	Justifications
Customer	**#1 Insertion #12 Range Search**	**Category**	**30 per day 100 per day**	**B+ Tree with Search key on category**	**As there is range search on category, B+ tree is preferred for range search**
Assembly	**#4 Insertion**		**40/ day**	**Heap File**	**Heap File is good for Insertion.**
Process	**#3 Insertion**		**Infrequent**	**Heap File**	**Heap File is good for Insertion.**
fit	**#3 Insertion**		**Infrequent**	**Heap File**	**Heap File is good for Insertion.**
paint	**#3 Insertion**		**Infrequent**	**Heap File**	**Heap File is good for Insertion.**
cut	**#3 Insertion**		**Infrequent**	**Heap File**	**Heap File is good for Insertion.**
Department	**#2 Insertion**		**Infrequent**	**Heap File**	**Heap File is good for Insertion.**
Account	**#5 Insertion**		**10/day**	**Heap File**	**Heap File is good for Insertion.**
Assembly_account	**#8 Insertion #5 Insertiion #9 Random Search**	**Assembly id**	**50/day 10/day 200/day**	**Dynamic Hashing with search key on assembly id**	**Dynamic Hashing is preferable for random search**

Process_account	**#8** **Insertion #5** **Insertiion**		**50/day** **10/day**	**Heap File**	**Heap File is good for Insertion.**
Dept_account	**#8 Insertion #5** **Insertiion**		**50/day** **10/day**	**Heap File**	**Heap File is good for Insertion.**
Job	**#7 Insertion #6** **Insertion #11** **Random Search**	**Assembly id**	**50/day** **50/day** **100/day**	**Dynamic Hashing with search key on assembly id**	**Dynamic Hashing is preferable for random search**
paint_job	**#7** **Insertion #14** **Random Search**	**Job Number**	**50/day** **1/week**	**Dynamic Hashing with search key on job number**	**Dynamic Hashing is preferable for random search**
Fit_job	**#7 Insertion**		**50/day**	**Heap file**	**Heap File is good for Insertion.**
Cut_job	**#7** **Insertion #13** **Range Search**	**Job number**	**50/day** **1/month**	**B+ Tree with Search key on job number**	**As there is range search on job number, B+ tree is preferred for range search**
Transaction	**#8 Insertion**		**50/day**	**Heap file**	**Heap File is good for Insertion.**
Order	**#4 Insertion**		**40/day**	**Heap file**	**Heap File is good for Insertion.**
Contains	**#4 Insertion**		**40/day**	**Heap file**	**Heap File is good for Insertion.**
Supervises	**#3 Insertion#11** **Random Search**	**Assembly id**	**Infrequent** **100/day**	**Dynamic Hashing with search key on assembly id**	**Dynamic Hashing is preferable for random search**

Assigns	**#6** **Insertion #11** **Random Search**	**Assembly id**	**50/day** **100/day**	**Dynamic Hashing with search key on assembly id**	**Dynamic Hashing is preferable for random search**
Assembly_Tracks	**#5 Insertiion #9 Random Search**	**Assembly id**	**10/day** **200/day**	**Dynamic Hashing with search key on assembly id**	**Dynamic Hashing is preferable for random search**
Processs_Tracks	**#5 Insertiion**		**10/day**	**Heap File**	**Heap File is good for Insertion.**
Department_Tracks	**#5 Insertiion**		**10/day**	**Heap File**	**Heap File is good for Insertion.**

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

A clustered index is by default created on primary key when primary key constraint is in table in AZURE SQL. We cannot create multiple clustered indexes on same table. Dynamic hashing is helpful in SQL when utilized in tables. Hash index can be created to make memory optimize tables but it is restricted in AZURE.

I have decided to use default primary indexes that are created on respected tables and create non cluster indexes for below:

Table Name	**Index Key**
Customer Table	Category
Job	Date Commenced

Task 4

SQL statements and screenshots showing the creation of tables in Azure SQL Database.

Creating index

```
create index cat on customer(category);
create index date on Job(commenced_date);
```

sages

```
:38:16 PM     Started executing query at Line 266
              Commands completed successfully.
              Total execution time: 00:00:00.094
```

1) Create Customer Table

```
----Creating customer table
create table customer(
    customer_name VARCHAR(200),
    cust_address VARCHAR (200) not null,
    category int not null,
    CONSTRAINT category_ck CHECK (category BETWEEN 1 and 10),
    PRIMARY KEY (customer_name)

)
```

```
Started executing query at Line 35
Commands completed successfully.
Total execution time: 00:00:00.066
```

2) Create department Table

```
----Creating Department table
create table Department(
    department_no INT,
    department_data VARCHAR (200) not null,
    PRIMARY KEY(department_no)
)
```

Messages

```
2:27:45 AM    Started executing query at Line 46
              Commands completed successfully.
              Total execution time: 00:00:00.083
```

3) Create process Table

```
----Creating process table
create table process(
    process_id INT,
    process_data VARCHAR(200) not null,
    PRIMARY KEY (process_id)
)
```

Messages

```
2:32:18 AM    Started executing query at Line 55
              Commands completed successfully.
              Total execution time: 00:00:00.093
```

4) Create Supervises Table

```
----Creating Supervises table
create table Supervises(
    process_id INT,
    department_no INT not null,
    PRIMARY KEY(process_id),
    FOREIGN KEY (process_id) REFERENCES process(process_id),
    FOREIGN KEY (department_no) REFERENCES Department(department_no),
)
```

Messages

```
2:33:29 AM      Started executing query at Line 62
                Commands completed successfully.
                Total execution time: 00:00:00.082
```

5) Create fit Table

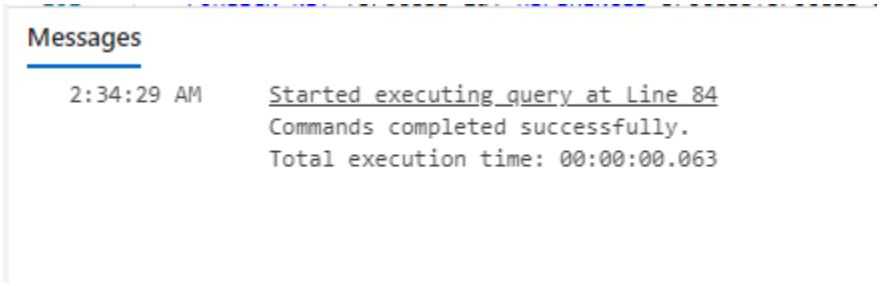
```
----Creating fit table
create table fit(
    process_id INT,
    fit_type VARCHAR(200),
    PRIMARY KEY (process_id),
    FOREIGN KEY (process_id) REFERENCES process(process_id)

)
```

Messages

```
2:34:01 AM      Started executing query at Line 74
                Commands completed successfully.
                Total execution time: 00:00:00.072
```

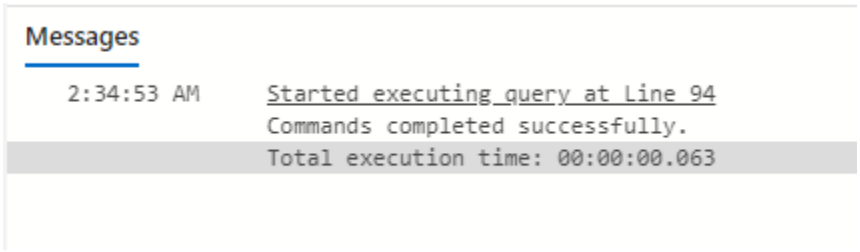
6) Create paint Table

```
----Creating paint table
create table paint(
    process_id INT,
    paint_type VARCHAR(200) not null,
    painting_method VARCHAR(60) not null,
    PRIMARY KEY (process_id),
    FOREIGN KEY (process_id) REFERENCES process(process_id)
)
```



7) Create cut Table

```
----Creating cut table
create table  cut(
    process_id INT,
    cutting_type VARCHAR(200) not null,
    machine_type VARCHAR(200) not null,
    PRIMARY KEY (process_id),
    FOREIGN KEY (process_id) REFERENCES process(process_id)
)
```



8) Create assembly Table

```
----Creating assembly table
create table assembly(
    assembly_id INT,
    assembly_date VARCHAR(200),
    assembly_details VARCHAR(200) not null,
    PRIMARY KEY (assembly_id)
)
```

Messages

```
2:35:58 AM     Started executing query at Line 104
               Commands completed successfully.
               Total execution time: 00:00:00.065
```

9) Create order Table

```
---- Creating Order table
create table Orders(
    assembly_id INT,
    customer_name VARCHAR(200) not null,
    PRIMARY KEY(assembly_id),
    FOREIGN KEY (assembly_id) REFERENCES assembly(assembly_id),
    FOREIGN KEY (customer_name) REFERENCES Customer(customer_name)
)
```

Messages

```
2:36:19 AM     Started executing query at Line 113
               Commands completed successfully.
               Total execution time: 00:00:00.063
```

10) Create contains Table

```
----Creating Contains table
create table Contain(
    assembly_id INT,
    process_id INT,
    PRIMARY KEY(assembly_id, process_id),
    FOREIGN KEY (assembly_id) REFERENCES assembly(assembly_id),
    FOREIGN KEY (process_id) REFERENCES Process(process_id)
)
```

Messages

```
2:36:44 AM     Started executing query at Line 123
               Commands completed successfully.
               Total execution time: 00:00:00.065
```

11) Create Account Table

```
----Creating Account table
Create table Account(
    acc_no INT,
    acc_date date not null,
    PRIMARY KEY(acc_no)
)
```

Messages

```
2:37:09 AM    Started executing query at Line 133
              Commands completed successfully.
              Total execution time: 00:00:00.075
```

12) Create process account table

```
----Creating Process account table
Create table process_account(
    acc_no INT,
    details_2 real,
    PRIMARY KEY(acc_no)
)
```

Messages

```
2:37:28 AM    Started executing query at Line 141
              Commands completed successfully.
              Total execution time: 00:00:00.064
```

13) Create process tracksTable

```
----Creating process tracks table
create table process_tracks(
    acc_no INT not null,
    process_id INT,
    PRIMARY KEY (process_id),
    FOREIGN KEY (acc_no) REFERENCES Account(acc_no),
    FOREIGN KEY (process_id) REFERENCES Process(process_id),
)
```

Messages

```
2:38:19 AM    Started executing query at Line 150
              Commands completed successfully.
              Total execution time: 00:00:00.066
```

14) Create assembly account table

```
----Creating Assembly account table
Create table Assembly_account(
    acc_no INT,
    details_1 real,
    PRIMARY KEY(acc_no)
)
```

Messages

```
2:38:41 AM    Started executing query at Line 161
              Commands completed successfully.
              Total execution time: 00:00:00.064
```

15) Create assembly tracks table

```
----Creating assembly tracks table
create table assembly_tracks(
    acc_no INT not null,
    assembly_id INT,
    PRIMARY KEY (assembly_id),
    FOREIGN KEY (acc_no) REFERENCES Account(acc_no),
    FOREIGN KEY (assembly_id) REFERENCES assembly(assembly_id),
)
```

Messages

```
2:39:04 AM    Started executing query at Line 170
              Commands completed successfully.
              Total execution time: 00:00:00.066
```

16) Create dept account table

```
----Creating Dept account table
Create table Dept_account(
    acc_no INT,
    details_3 real,
    PRIMARY KEY(acc_no)
)
```

Messages

```
2:39:36 AM    Started executing query at Line 181
              Commands completed successfully.
              Total execution time: 00:00:00.075
```

17) Create dept tracks table

```
----Creating dept tracks table
create table dept_tracks(
    acc_no INT not null,
    department_no int,
    PRIMARY KEY (department_no),
    FOREIGN KEY (acc_no) REFERENCES Account(acc_no),
    FOREIGN KEY (department_no) REFERENCES Department(department_no),
)
```

Messages

```
2:39:57 AM    Started executing query at Line 191
              Commands completed successfully.
              Total execution time: 00:00:00.072
```

18) Create Job Table

```
----Creating Job table
Create table Job(
    job_no INT,
    commenced_date date,
    completed_date date,
    PRIMARY KEY (job_no)
)
```

Messages

```
2:40:43 AM    Started executing query at Line 201
              Commands completed successfully.
              Total execution time: 00:00:00.066
```

19) Create Assigns Table

```
----Creating Assigns table
create table Assigns(
    job_no INT,
    assembly_id INT not null,
    process_id INT not null,
    PRIMARY KEY (job_no),
    FOREIGN KEY (job_no) REFERENCES Job(job_no),
    FOREIGN KEY (assembly_id) REFERENCES assembly(assembly_id),
    FOREIGN KEY (process_id) REFERENCES process(process_id),
)
```

Messages

```
2:40:59 AM    Started executing query at Line 211
              Commands completed successfully.
              Total execution time: 00:00:00.080
```

20) Create paint job table

```
----Creating paint job table
create table paint_job(
    job_no INT,
    color VARCHAR(200) not null,
    Volume FLOAT not null,
    labor_time INT not null,
    PRIMARY KEY (job_no),
    FOREIGN KEY (job_no) REFERENCES Job(job_no)
)
```

Messages

```
2:41:34 AM    Started executing query at Line 225
              Commands completed successfully.
              Total execution time: 00:00:00.064
```

21) Create fit job Table

```
----Creating fit job table
create table Fit_job(
    job_no INT,
    labor_time INT not null,
    PRIMARY KEY (job_no),
    FOREIGN KEY (job_no) REFERENCES Job(job_no)
)
```

Messages

```
2:41:53 AM    Started executing query at Line 236
              Commands completed successfully.
              Total execution time: 00:00:00.066
```

22) Create cut job table

```
---- Creating cut job table
create table Cut_job(
    job_no INT,
    type_of_machine_used VARCHAR(200) not null,
    amount_of_time time not null,
    material VARCHAR(200) not null,
    labor_time time not null,
    PRIMARY KEY (job_no),
    FOREIGN KEY (job_no) REFERENCES Job(job_no)
)
```

Messages

```
2:42:14 AM    Started executing query at Line 245
              Commands completed successfully.
              Total execution time: 00:00:00.067
```

23) Create transaction table

```
---- Creating Transaction table
create table Transactions(
    transaction_id int,
    sup_cost REAL not null,
    PRIMARY KEY(transaction_id)
)
```

Task 5

Procedures

1. Enter a new customer:

```
98    ----- for proc1
99    ----- Q1
100   drop procedure if exists proc1;
101   GO
102   CREATE PROCEDURE proc1
103       @customer_name VARCHAR(200),
104       @cust_address VARCHAR(200),
105       @category int
106
107   AS
108   BEGIN
109   insert into customer
110   (
111       customer_name,cust_address,category
112   )
113   VALUES(@customer_name,@cust_address,@category)
114   END
115
```

lessages

```
2:43:12 AM    Started executing query at Line 98
              Commands completed successfully.
2:43:12 AM    Started executing query at Line 102
              Commands completed successfully.
              Total execution time: 00:00:00.126
```

2. Enter a new department

```
161   ----- Q2
162   drop procedure if exists proc2;
163   GO
164   CREATE PROCEDURE proc2
165       @department_no int,
166       @department_data VARCHAR(200)
167
168   AS
169   BEGIN
170   insert into Department
171   (
172       department_no,department_data
173   )
174   VALUES(@department_no,@department_data)
175   END
176
```

Messages

```
2:44:41 AM    Started executing query at Line 161
              Commands completed successfully.
2:44:41 AM    Started executing query at Line 164
              Commands completed successfully.
              Total execution time: 00:00:00.191
```

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

```
----Q3
drop procedure if exists proc3

GO
CREATE PROCEDURE proc3
    @process_type VARCHAR(200),
    @process_id int,
    @process_data VARCHAR(200),
    @fit_type VARCHAR(200) = NULL,
    @paint_type VARCHAR(200)=NULL,
    @painting_method VARCHAR(200)= NULL,
    @cutting_type VARCHAR(200) = NULL,
    @machine_type VARCHAR(200)= NULL,
    @department_no INT
AS
BEGIN
insert into process
(
    process_id, process_data
)
VALUES(@process_id,@process_data)

insert into Supervises(
    process_id,department_no
)
VALUES(@process_id, @department_no)

IF @process_type = 'fit'

BEGIN
Insert into fit(
    process_id,fit_type
)
VALUES(@process_id,@fit_type)
END

IF @process_type = 'paint'
BEGIN
INSERT into paint(
    process_id,paint_type,painting_method
)
values (@process_id,@paint_type,@painting_method)
END
```

```
238
239    IF @process_type = 'cut'
240    BEGIN
241    insert into cut(
242        process_id,cutting_type,machine_type
243    )
244    values (@process_id,@cutting_type,@machine_type)
245
246    END
247    END
248
```

Messages

```
2:45:56 AM    Started executing query at Line 194
              Commands completed successfully.
2:45:56 AM    Started executing query at Line 199
              Commands completed successfully.
              Total execution time: 00:00:00.144
```

4. Enter a new assembly with its customer-name, assembly-details, assembly-id, and dateordered and associate it with one or more processes

```
283    drop procedure if exists proc4
284    GO
285    CREATE PROCEDURE proc4
286        @assembly_id int,
287        @assembly_date VARCHAR(200),
288        @assembly_details VARCHAR(200),
289        @cust_name VARCHAR(200)
290
291    AS
292    BEGIN
293
294    insert into assembly(
295        assembly_id,assembly_date,assembly_details
296    )
297    VALUES(@assembly_id, @assembly_date, @assembly_details)
298
299    insert into Orders(
300        assembly_id,customer_name
301    )
302    values(@assembly_id, @cust_name)
303
304    END
305    ------
306    drop procedure if exists Proc4_1;
307    GO
308    CREATE PROCEDURE Proc4_1
309        @process_id int,
310        @assembly_id int
311
312    AS
313    BEGIN
314    insert into Contain(
315        assembly_id,process_id
316    )
317    VALUES(@assembly_id,@process_id)
318
319    END;
```

Messages

```
2:47:42 AM    Started executing query at Line 283
              Commands completed successfully.
2:47:42 AM    Started executing query at Line 285
              Commands completed successfully.
2:47:42 AM    Started executing query at Line 308
              Commands completed successfully.
```

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

```
347
348   drop PROCEDURE if EXISTS proc5
349
350   GO
351   CREATE PROCEDURE proc5
352       @acc_no int,
353       @acc_date VARCHAR(200),
354       @acc_associate VARCHAR(200),
355       @details_2 REAL,
356       @details_1 REAL,
357       @details_3 REAL,
358       @department_no INT,
359       @process_id int,
360       @assembly_id INT
361
362   AS
363   BEGIN
364
365   insert into Account(
366       acc_no,acc_date
367   )VALUES(@acc_no,CAST(@acc_date AS date))
368
369    IF @acc_associate = 'process'
370    BEGIN
371    insert into process_account(
372        acc_no,details_2
373    )VALUES(@acc_no,@details_2)
374
375    INSERT into process_tracks(
376        acc_no,process_id
377    )VALUES(@acc_no,@process_id)
378    END
379
380    IF @acc_associate = 'assembly'
381    BEGIN
382    insert into Assembly_account(
383        acc_no,details_1
384    )VALUES(@acc_no,@details_1)
385    insert INTO assembly_tracks(
386        acc_no,assembly_id
387    )VALUES(@acc_no,@assembly_id)
388    END
389
390
391    IF @acc_associate = 'department'
392    BEGIN
393    insert into Dept_account(
394        acc_no,details_3
395    )VALUES(@acc_no,@details_3)
396
397    insert into dept_tracks(
398        acc_no, department_no
399    )VALUES(@acc_no,@department_no)
400    END
401    END
402
```

Messages

```
2:49:32 AM     Started executing query at Line 348
               Commands completed successfully.
2:49:32 AM     Started executing query at Line 351
               Commands completed successfully.
               Total execution time: 00:00:00.137
```

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

```
427    ---Q6
428
429    drop procedure if exists proc6
430
431    GO
432    CREATE PROCEDURE proc6
433        @job_no int,
434        @commenced_date VARCHAR(200),
435        @assembly_id int,
436        @process_id INT
437
438    AS
439    BEGIN
440
441    insert INTO Job(
442        job_no,commenced_date
443    )values(@job_no,@commenced_date)
444
445    insert into Assigns(
446        job_no,assembly_id,process_id
447    )VALUES(@job_no,@assembly_id,@process_id)
448
449    END
450
```

Messages

```
2:50:17 AM     Started executing query at Line 427
               Commands completed successfully.
2:50:17 AM     Started executing query at Line 432
               Commands completed successfully.
               Total execution time: 00:00:00.116
```

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

```
477  GO
478  CREATE PROCEDURE proc7
479      @job_no int,
480      @completed_date VARCHAR(200),
481      @job_type VARCHAR(200),
482      @labor_time VARCHAR(200),
483      @color VARCHAR(200),
484      @volume int,
485      @type_of_machine_used VARCHAR(200),
486      @amount_of_time VARCHAR(200),
487      @material VARCHAR(200)
488
489  AS
490  BEGIN
491
492  update Job SET completed_date = @completed_date WHERE job_no = @job_no
493
494  if @job_type = 'fit job'
495  BEGIN
496  insert into Fit_job(
497      job_no,labor_time
498  )VALUES(@job_no,cast(@labor_time as time))
499  END
500
```

```
501    if @job_type = 'paint job'
502    BEGIN
503    insert into paint_job(
504        color,volume,labor_time,job_no
505    )
506    VALUES (@color,@volume,cast(@labor_time as time),@job_no)
507    END
508
509    if @job_type = 'cut job'
510    BEGIN
511    INSERT INTO Cut_job(
512        type_of_machine_used, amount_of_time,material,labor_time,job_no
513    )
514    VALUES(@type_of_machine_used,cast(@amount_of_time as time),@material,cast(@labor_time as time),@job_no)
515
516    END
517    END
518
```

Messages

```
2:51:27 AM    Started executing query at Line 473
              Commands completed successfully.
2:51:27 AM    Started executing query at Line 478
              Commands completed successfully.
              Total execution time: 00:00:00.110
```

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)

```
267
268   drop PROCEDURE if EXISTS proc8
269
270   GO
271   CREATE PROCEDURE proc8
272
273       @transaction_id INT,
274       @sup_cost real,
275       @acc_no INT
276   AS
277   BEGIN
278
279   insert into Transactions(
280       transaction_id,sup_cost
281   )VALUES(@transaction_id,@sup_cost)
282
283
284   update Assembly_account
285   set details_1 = details_1 + @sup_cost
286   WHERE acc_no = @acc_no
287
288   update process_account
289   set details_2 = details_2 + @sup_cost
290   WHERE acc_no = @acc_no
291
292   update Dept_account
293   set details_3 = details_3 + @sup_cost
294   WHERE acc_no = @acc_no
295   END
296
```

Messages

```
3:38:44 PM     Started executing query at Line 267
               Commands completed successfully.
3:38:44 PM     Started executing query at Line 271
               Commands completed successfully.
               Total execution time: 00:00:00.164
```

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

```
300   ---Q9
301
302   drop PROCEDURE if exists proc9
303
304   GO
305   create PROCEDURE proc9
306       @assembly_id int
307   AS
308   BEGIN
309       SELECT [details_1]
310       FROM Assembly_account Aa
311       INNER JOIN assembly_tracks Atr ON Atr.acc_no = Aa.acc_no
312       WHERE assembly_id = @assembly_id
313   end;
314
315
316   exec proc9 @assembly_id = 203;
317
318   select * from assembly_tracks;
319
```

Results | Messages

```
1:23:36 PM     Started executing query at Line 318
               (3 rows affected)
               Total execution time: 00:00:00.078
```

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

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

```
585   drop PROCEDURE if exists proc11
586
587   GO
588   Create PROCEDURE proc11
589       @assembly_id int
590
591   AS
592   BEGIN
593
594   select Job.commenced_date,Assigns.process_id, Supervises.department_no
595   from Job,Assigns,Supervises
596   where Assigns.assembly_id = @assembly_id and Assigns.process_id = Supervises.process_id and Assigns.job_no = Job.job_no order by Job.commenced_date;
597
598   END
599
```

Messages

```
2:52:15 AM    Started executing query at Line 585
              Commands completed successfully.
2:52:15 AM    Started executing query at Line 588
              Commands completed successfully.
              Total execution time: 00:00:00.120
```

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

```
603   ----Q12
604
605   drop PROCEDURE if exists proc12;
606
607   Go
608   CREATE PROCEDURE proc12
609       @low_cat INT,
610       @high_cat INT
611
612   AS
613   BEGIN
614
615   select * from customer
616   where category >= @low_cat AND category <= @high_cat
617   order by customer_name;
618
619   END
620
```

Messages

```
2:52:47 AM    Started executing query at Line 605
              Commands completed successfully.
2:52:47 AM    Started executing query at Line 608
              Commands completed successfully.
              Total execution time: 00:00:00.121
```

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

```
623    ----Q13
624
625    drop PROCEDURE if exists proc13
626
627    GO
628    CREATE PROCEDURE proc13
629        @low_jobno int,
630        @high_jobno int
631
632    AS
633    BEGIN
634    DELETE From Cut_job where job_no between @low_jobno and @high_jobno;
635
636    END
637
```

Messages

```
2:53:21 AM     Started executing query at Line 625
               Commands completed successfully.
2:53:21 AM     Started executing query at Line 628
               Commands completed successfully.
               Total execution time: 00:00:00.119
```

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

```
644    ----Q14
645    drop PROCEDURE if exists proc14
646    GO
647    create PROCEDURE proc14
648        @color VARCHAR(200),
649        @job_no VARCHAR(200)
650
651    AS
652    BEGIN
653    UPDATE paint_job set color = @color where job_no = @job_no;
654
655    END
656
```

Messages

```
2:53:50 AM     Started executing query at Line 644
               Commands completed successfully.
2:53:50 AM     Started executing query at Line 647
               Commands completed successfully.
               Total execution time: 00:00:00.117
```

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

```java
import java.io.File;
public class ip {

    // Database credentials
    final static String HOSTNAME = "peri0015-sql-server.database.windows.net";
    final static String DBNAME = "cs-dsa-4513-sql-db";
    final static String USERNAME = "peri0015";
    final static String PASSWORD = "Gokaraju@0525";

    // Database connection string
    final static String URL =
    String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trustServerCertificate=false;hostNameInCertificate=*.dat
    HOSTNAME, DBNAME, USERNAME, PASSWORD);


    // User input prompt//
    final static String PROMPT =
            "\nPlease select one of the options below: \n" +
            "1. Enter a new customer (30/day); \n" +
            "2. Enter a new department (infrequent) ;\n" +
            "3. Enter a new process-id and its department together with its type and information relevant to the type (infrequent) ; \n" +
            "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) ; \n" +
            "5. Create a new account and associate it with the process, assembly, or department to which it is applicable (10/day) ; \n" +
            "6. Enter a new job, given its job-no, assembly-id, process-id, and date the job commenced (50/day) ; \n" +
            "7. At the completion of a job, enter the date it completed and the information relevant to the type of job (50/day) ; \n" +
            "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) ; \n" +
            "9. Retrieve the total cost incurred on an assembly-id (200/day) ; \n" +
            "10. Retrieve the total labor time within a department for jobs completed in the department during a given date (20/day) ; \n" +
            "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) ; \n" +
            "12. Retrieve the customers (in name order) whose category is in a given range (100/day) ; \n" +
            "13. Delete all cut-jobs whose job-no is in a given range (1/month) ; \n" +
            "14. Change the color of a given paint job (1/week) ; \n" +
            "15. Import: enter new customers from a data file until the file is empty"
            + " (the user must be asked to enter the input file name) ; \n" +
            "16. Export: Retrieve the customers (in name order) whose category is in a given range "
            + "and output them to a data file instead of screen (the user must be asked to enter the output file name) ; \n" +
            "17. Quit";

```

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

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

        System.out.println("Welcome to the application - peri!");
        final Scanner sc = new Scanner(System.in); // Scanner is used to collect the user input
        String option = ""; // Initialize user option selection as nothing
        while (!option.equals("17")) { // As user for options until option 4 is selected
            System.out.println(PROMPT); // Print the available options
            option = sc.next(); // Read in the user option selection

            switch (option) { // Switch between different options

                case "1": // Insert customer details
                 try {
                    System.out.println("Enter customer name:");
                    sc.nextLine();
                    String customer_name = sc.nextLine();

                    System.out.println("Enter customer address:");
                    String customer_address= sc.nextLine();

                    System.out.println("Enter category");
                    int category = sc.nextInt();

                    System.out.println("Connecting to the database...");
                   // Get a database connection and prepare a query statement
                    try (final Connection connection = DriverManager.getConnection(URL)) {
                        try (
                            final PreparedStatement statement = connection.prepareStatement
                            ("EXEC proc1 @customer_name = ?, @cust_address = ?,@category = ?")) {
                            // Populate the query template with the data collected from the user

                            statement.setString(1, customer_name);
                            statement.setString(2, customer_address);
                            statement.setInt(3, category);

                            System.out.println("Dispatching the query...");
                            // Actually execute the populated query
                            final int rows_inserted = statement.executeUpdate();
                            System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
                        }
                    }
                        }
                    }
                    catch(Exception error) {
                        error.printStackTrace();
                        System.out.println("error in case 1");
                    }

                            break;
```

2. Enter a new department (infrequent).

```
99
100         case "2": // Insert department details
101             try {
102             System.out.println("Enter department number");
103             int department_no = sc.nextInt(); //
104
105             System.out.println("Enter department data:");
106             sc.nextLine();
107             String department_data = sc.nextLine();
108
109
110             System.out.println("Connecting to the database...");
111          // Get a database connection and prepare a query statement
112             try (final Connection connection = DriverManager.getConnection(URL)) {
113                 try (
114                     final PreparedStatement statement = connection.prepareStatement("EXEC proc2 @department_no = ?,@department_data = ?")) {
115                     // Populate the query template with the data collected from the user
116                     statement.setInt(1, department_no);
117                     statement.setString(2, department_data);
118
119
120                     System.out.println("Dispatching the query...");
121                     // Actually execute the populated query
122                     final int rows_inserted = statement.executeUpdate();
123                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
124                 }
125             }
126     }
127     catch(Exception error) {
128         error.printStackTrace();
129         System.out.println("error in case 2");
130     }
131
132             break;
133
```

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

```
        case "3": // Insert customer details
            try {
            System.out.println("Enter process type:");
            sc.nextLine();
            String process_type = sc.nextLine();

            System.out.println("Enter process id");
            int process_id = sc.nextInt(); //

            System.out.println("Enter process data:");
            sc.nextLine();
            String process_data = sc.nextLine();

            System.out.println("Enter fit type:");
            String fit_type = sc.nextLine();

            System.out.println("Enter paint type:");
            String paint_type = sc.nextLine();

            System.out.println("Enter painting method:");
            String painting_method = sc.nextLine();


            System.out.println("Enter cut type:");
            String cut_type = sc.nextLine();

            System.out.println("Enter machine type:");
            String machine_type = sc.nextLine();

            System.out.println("Enter department number");
            int dept_no = sc.nextInt(); //
```

```
    System.out.println("Connecting to the database...");
// Get a database connection and prepare a query statement
    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (
            final PreparedStatement statement = connection.prepareStatement
            ("exec proc3 @process_type = ?, @process_id = ?, @process_data = ?, @fit_type = ?, @paint_type = ?, @painting_method    = ?, @cutting_
            {
            // Populate the query template with the data collected from the user
            statement.setString(1, process_type);
            statement.setInt(2, process_id);
            statement.setString(3, process_data);
            statement.setString(4, fit_type);
            statement.setString(5, paint_type);
            statement.setString(6, painting_method);
            statement.setString(7, cut_type);
            statement.setString(8, machine_type);
            statement.setInt(9, dept_no);



            System.out.println("Dispatching the query...");
            // Actually execute the populated query
            final int rows_inserted = statement.executeUpdate();

            System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
        }
    }
    }
    catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 3");
    }

    break;
```

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

```
case "4": // Insert customer details
    try {
    System.out.println("Enter assembly id");
    int assid = sc.nextInt(); //

    System.out.println("Enter assembly date:");
    sc.nextLine();
    String assdate = sc.nextLine();

    System.out.println("Enter assembly details:");
    String assdetails = sc.nextLine();

    System.out.println("Enter the customer name:");
    String customer_name = sc.nextLine();

    System.out.println("Enter no of process:");
    int no_of_process = sc.nextInt();

    System.out.println("Connecting to the database...");
    // Get a database connection and prepare a query statement
    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (
            final PreparedStatement statement = connection.prepareStatement
            ("EXEC proc4 @assembly_id = ?, @assembly_date = ?, @assembly_details = ?, @cust_name = ?;")) {
            // Populate the query template with the data collected from the user
            statement.setInt(1, assid);
            statement.setString(2, assdate);
            statement.setString(3, assdetails);
            statement.setString(4, customer_name);

            System.out.println("Dispatching the query...");
            // Actually execute the populated query
            final int rows_inserted = statement.executeUpdate();

            System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
        }
    }
```

```
241         for(int i = 1 ; i <= no_of_process; i++) {
242
243             System.out.println("enter assembly id:");
244             int assid1 = sc.nextInt();
245
246             System.out.println("enter the process id");
247             int assproid = sc.nextInt();
248
249             try (final Connection connection1 = DriverManager.getConnection(URL)) {
250                 try(
251                     final PreparedStatement statement1 = connection1.prepareStatement("EXEC Proc4_1 @process_id = ?, @assembly_id = ?;")) {
252                     // Populate the query template with the data collected from the user
253                     statement1.setInt(1, assproid);
254                     statement1.setInt(2,assid1);
255
256
257                     final int rows_inserted1 = statement1.executeUpdate();
258
259                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted1));
260
261                 }
262             }
263         }
264 }
265 catch(Exception error) {
266     error.printStackTrace();
267     System.out.println("error in case 4");
268 }
269
270
271         break;
```

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

```
273         case "5": // Insert customer details
274             try {
275             System.out.println("Enter account number:");
276             sc.nextLine();
277             int acc_no = sc.nextInt();
278 
279             System.out.println("Enter account date:");
280             sc.nextLine();
281             String acc_date = sc.nextLine(); //
282 
283             System.out.println("Enter acc associated to:");
284             String acc_associate = sc.nextLine();
285 
286             System.out.println("Enter details2 for process:");
287             float details2 = sc.nextFloat();
288 
289             System.out.println("Enter details1 for assembly");
290             float details1 = sc.nextFloat();
291 
292             System.out.println("Enter details3 for dept:");
293             Float details3 = sc.nextFloat();
294 
295 
296             System.out.println("Enter department number:");
297             int accdeptno = sc.nextInt();
298 
299             System.out.println("Enter process id:");
300             int accproid = sc.nextInt();
301 
302             System.out.println("Enter assembly id:");
303             int accassid = sc.nextInt(); //
304 
305 
306             System.out.println("Connecting to the database...");
307         // Get a database connection and prepare a query statement
308             try (final Connection connection = DriverManager.getConnection(URL)) {
309                 try (
310                     final PreparedStatement statement = connection.prepareStatement
311                     ("exec proc5 @acc_no = ?, @acc_date = ?, @acc_associate = ?, @details_2 = ?, @details_1 = ?, @details_3 = ?, @department_no =  ?
312                     {
313                     // Populate the query template with the data collected from the user
314                     statement.setInt(1, acc_no);
315                     statement.setString(2, acc_date);
316                     statement.setString(3, acc_associate);
317                     statement.setFloat(4, details2);
318                     statement.setFloat(5, details1);
319                     statement.setFloat(6, details3);
320                     statement.setInt(7, accdeptno);
321                     statement.setInt(8, accproid);
322                     statement.setInt(9, accassid);
323 
324 
325 
326                     System.out.println("Dispatching the query...");
327                     // Actually execute the populated query
328                     final int rows_inserted = statement.executeUpdate();
329 
330                     System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
331                 }
332             }
333     }
334     catch(Exception error) {
335         error.printStackTrace();
336         System.out.println("error in case 5");
337     }
338 
339             break;
340 
```

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

```
340
341                 case "6" :
342                     try {
343                     System.out.println("enter the job no");
344                     sc.nextLine();
345                     String job_no = sc.nextLine();
346
347                     System.out.println("Enter job commenced date:");
348                     sc.nextLine();
349                     String commenced_date = sc.nextLine(); //
350
351                     System.out.println("Enter assembly id");
352                     int jassid = sc.nextInt();
353
354                     System.out.println("Enter process id");
355                     int jproid = sc.nextInt();
356
357                     System.out.println("Connecting to the database...");
358                  // Get a database connection and prepare a query statement
359                     try (final Connection connection = DriverManager.getConnection(URL)) {
360                         try (
361                             final PreparedStatement statement = connection.prepareStatement
362                             ("exec proc6 @job_no = ?, @commenced_date = ?, @assembly_id = ? , @process_id = ? ;"))
363                             {
364                             // Populate the query template with the data collected from the user
365                             statement.setString(1, job_no);
366                             statement.setString(2, commenced_date);
367                             statement.setInt(3, jassid);
368                             statement.setInt(4, jproid);
369
370                             System.out.println("Dispatching the query...");
371                             // Actually execute the populated query
372                             final int rows_inserted = statement.executeUpdate();
373
374                             System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
375                         }
376                     }
377                     }
378                     catch(Exception error) {
379                         error.printStackTrace();
380                         System.out.println("error in case 6");
381                     }
382                      break;
```

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

```
case "7" :
    try {
    System.out.println("enter the job no");
    sc.nextLine();
    String job_no1 = sc.nextLine();

    System.out.println("Enter job completed date:");
    String completed_date = sc.nextLine(); //

    System.out.println("Enter job type");
    String job_type = sc.nextLine();

    System.out.println("Enter labor time");
    String labor_time = sc.nextLine();


    System.out.println("Enter color");
    String color = sc.nextLine();

    System.out.println("Enter volume");
    String volume = sc.nextLine();


    System.out.println("Enter type of machine");
    String type_of_machine = sc.nextLine();

    System.out.println("Enter amount of time:");
    String amt_of_time = sc.nextLine();

    System.out.println("Enter the material:");
    String material = sc.nextLine();
```

```
        System.out.println("Connecting to the database...");
    // Get a database connection and prepare a query statement
        try (final Connection connection = DriverManager.getConnection(URL)) {
            try (
                final PreparedStatement statement = connection.prepareStatement
                ("exec proc7 @job_no = ?, @completed_date = ?, @job_type = ?, @labor_time = ?, @color = ?, @volume = ?, @type_of_machine_used =
                {
                // Populate the query template with the data collected from the user
                statement.setString(1, job_no1);
                statement.setString(2, completed_date);
                statement.setString(3, job_type);
                statement.setString(4, labor_time);
                statement.setString(5, color);
                statement.setString(6, volume);
                statement.setString(7, type_of_machine);
                statement.setString(8, amt_of_time);
                statement.setString(9, material);

                System.out.println("Dispatching the query...");
                // Actually execute the populated query
                final int rows_inserted = statement.executeUpdate();

                System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
            }
        }
}catch(Exception error) {
    error.printStackTrace();
    System.out.println("error in case 7");
}
```

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).

```
case "8" :
    try {
    System.out.println("enter the transaction number");
    sc.nextLine();
    int trans_no = sc.nextInt();

    System.out.println("Enter supcost");
    float supcost = sc.nextFloat(); //

    System.out.println("Enter account number");
    int acc_number = sc.nextInt();

    System.out.println("Connecting to the database...");
 // Get a database connection and prepare a query statement
    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (
            final PreparedStatement statement = connection.prepareStatement
            ("exec proc8 @transaction_id = ?, @sup_cost = ?, @acc_no = ?;"))
            {
            // Populate the query template with the data collected from the user
            statement.setInt(1, trans_no);
            statement.setFloat(2, supcost);
            statement.setInt(3, acc_number);

            System.out.println("Dispatching the query...");
            // Actually execute the populated query
            final int rows_inserted = statement.executeUpdate();

            System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
        }
    }
    }
    catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 8");
    }
```

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

```
case "9" :
    try {
    System.out.println("Enter assembly id:");
    int ass_id = sc.nextInt(); //

    System.out.println("Connecting to the database...");

    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (  final PreparedStatement statement = connection.prepareStatement("exec proc9 @assembly_id = ?;")) {


                statement.setInt(1,ass_id);

                ResultSet resultSet = statement.executeQuery();

                System.out.println("results of query 9: \n");
                sc.nextLine();

                System.out.println("Total cost on assembly id: \n");
                System.out.println("total cost\n");

                // Unpack the tuples returned by the database and print them out to the user
                while (resultSet.next()) {
                    System.out.println(String.format(   "%s",

                        resultSet.getInt(1)));

                }
        }
    }
    }
    catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 9");
    }
    break;
```

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

```
case "11" :
    try {
    System.out.println("Enter assembly id:");
    int assid = sc.nextInt(); //


    System.out.println("Connecting to the database...");

    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (  final PreparedStatement statement = connection.prepareStatement("Exec proc11 @assembly_id = ?;")) {


                statement.setInt(1, assid);

                ResultSet resultSet = statement.executeQuery();

                System.out.println("results of query 11: \n");
                sc.nextLine();

                System.out.println("Contents of the customer table: \n");
                System.out.println("Commenced date | Process_id     | Department_no \n");

                // Unpack the tuples returned by the database and print them out to the user
                while (resultSet.next()) {
                    System.out.println(String.format(   "%s      | %s     |       %s  ",
                        resultSet.getString(1),
                        resultSet.getString(2),
                        resultSet.getInt(3)));

                }
        }
    }
    }
    catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 11");
    }


    break;
```

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

```
case "12" :
    try {
    System.out.println("Enter low category:");
    int low_cat = sc.nextInt(); //

    System.out.println("Enter high category:");
    int high_cat = sc.nextInt();

    System.out.println("Connecting to the database...");

    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (  final PreparedStatement statement = connection.prepareStatement("Exec proc12 @low_cat = ?, @high_cat = ?;")) {


                statement.setInt(1, low_cat);
                statement.setInt(2, high_cat);

                ResultSet resultSet = statement.executeQuery();

                System.out.println("results of query 12: \n");
                sc.nextLine();

                System.out.println("Contents of the customer table: \n");
                System.out.println("cust_name | cust_address        | category  \n");

                // Unpack the tuples returned by the database and print them out to the user
                while (resultSet.next()) {
                    System.out.println(String.format(   "%s   |     %s     |       %s  ",
                        resultSet.getString(1),
                        resultSet.getString(2),
                        resultSet.getInt(3)));

                }
        }
    }
    }
    catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 12");
    }
    break;
```

13. Delete all cut-jobs whose job-no is in a given range (1/month). 14. Change the color of a given paint job (1/week).

```
case "13" :
    try {
    System.out.println("Enter low jobno:");
    int low_jobno = sc.nextInt(); //

    System.out.println("Enter high jobno:");
    int high_jobno = sc.nextInt();

    System.out.println("Connecting to the database...");
   // Get a database connection and prepare a query statement
    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (
            final PreparedStatement statement = connection.prepareStatement
            ("exec proc13 @low_jobno = ? ,@high_jobno = ?;"))
            {
            // Populate the query template with the data collected from the user
            statement.setInt(1, low_jobno);
            statement.setInt(2, high_jobno);


            System.out.println("Dispatching the query...");
            // Actually execute the populated query
            final int rows_inserted = statement.executeUpdate();

            System.out.println(String.format("Done. rows deleted.", rows_inserted));
        }
    }
}catch(Exception error) {
    error.printStackTrace();
    System.out.println("error in case 13");
}

    break;
```

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

```
case "14" :
    try {
    System.out.println("enter the job no");
    sc.nextLine();
    String job_no2 = sc.nextLine();


    System.out.println("Enter color to be changed");
    String color1 = sc.nextLine();


    System.out.println("Connecting to the database...");
 // Get a database connection and prepare a query statement
    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (
            final PreparedStatement statement = connection.prepareStatement
            ("exec proc14 @job_no = ?,  @color = ?;"))
            {
            // Populate the query template with the data collected from the user
            statement.setString(1, job_no2);
            statement.setString(2, color1);


            System.out.println("Dispatching the query...");
            // Actually execute the populated query
            final int rows_inserted = statement.executeUpdate();

            System.out.println(String.format("Done. %d row updated.", rows_inserted));
        }
    }
    }catch(Exception error) {
        error.printStackTrace();
        System.out.println("error in case 14");
    }

    break;
```

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

```
case "15":
    try {
        String file_name,line;
        System.out.println("enter the file name to read");
        file_name = sc.next();

        File file = new File(file_name);

        Scanner sc1 = new Scanner(file);

        while(sc1.hasNextLine()) {
            line = sc1.nextLine();

            String[] parts = line.split(",");

            String cust_name = parts[0];
            String cust_address = parts[1];
            int insert_category = Integer.parseInt(parts[2]);

            final String query15 = "EXEC proc1 @customer_name = '"+cust_name+"', @cust_address = '"+cust_address+"' , @category = '"+insert_categ

            try(final Connection connection = DriverManager.getConnection(URL)){
                try(final PreparedStatement statement = connection.prepareStatement(query15)){
                    final int rows_inserted = statement.executeUpdate();

                    System.out.println(String.format("Done. %d rows inserted.", rows_inserted));
                }
            }
        }
    }catch(Exception error) {
            error.printStackTrace();
            System.out.println("You got an error in import");
        }
    break;
```

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

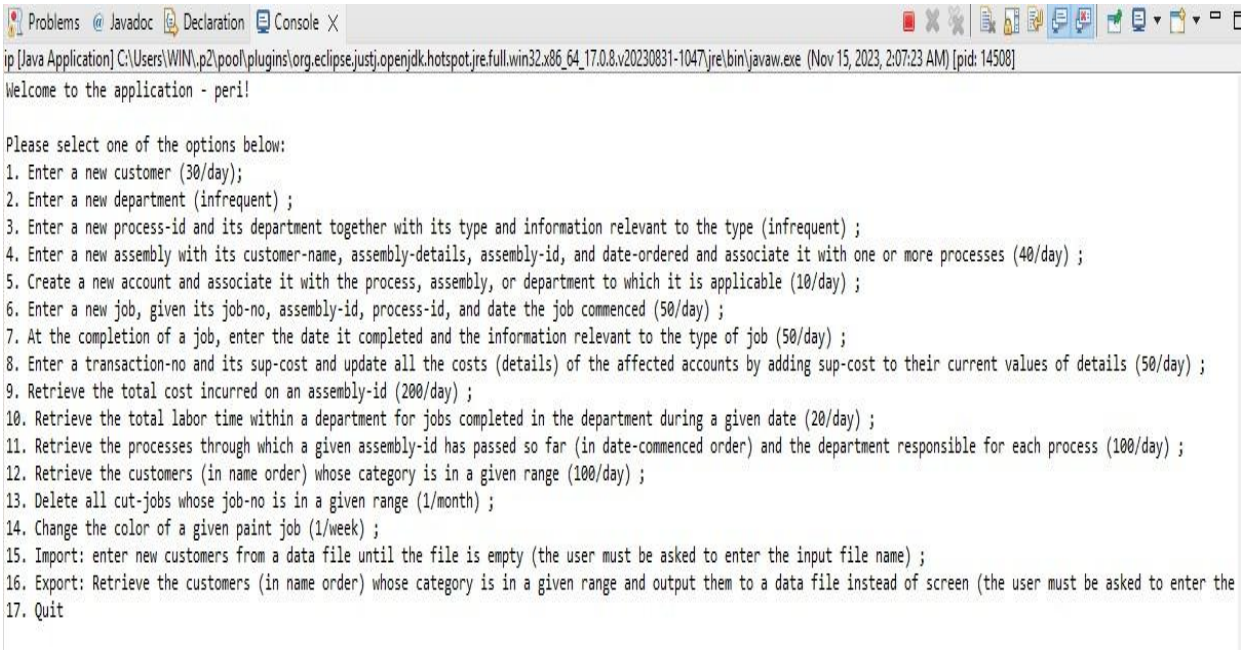
```
case "16":
    try {
    String file_name;
    int low_category, high_category;

    System.out.println("Enter the file name");
    file_name = sc.next();

    System.out.println("Enter lower category number");
    low_category = sc.nextInt();


    System.out.println("Enter high category number");
    high_category = sc.nextInt();

    FileWriter fw = new FileWriter(file_name);

    final String query16 = "EXEC proc12 @low_cat = '"+low_category+"', @high_cat = '"+high_category+"';";

    try (final Connection connection = DriverManager.getConnection(URL)) {
        try (  final PreparedStatement statement = connection.prepareStatement(query16)) {

            final ResultSet resultset = statement.executeQuery();

            while(resultset.next()) {
                fw.write(resultset.getString(1)+"\n");
            }
            fw.close();
        }
        }
    }catch(Exception error) {
        error.printStackTrace();
        System.out.println("You got an error in export");
    }

    break;
```

(17) Quit

```
case "17": // Quiting
    System.out.println("Quiting! Good-bye!");
    break;
default: // option not selected from 1 - 4, re-prompt the user for the correct one
    System.out.println(String.format(
        "option not selected from 1 - 17: %s\n" +
        "Please try again!",
        option));
    break;
```

Successful compliation of java code:

Select options from 1-17:



TASK 6 Java Program Execution:

Screenshots of testing query 1:

Tables before insertion:

Customer Table:

Results Messages

	customer_name	cust_address	category

Inserting Customers:

1st Insertion:

```
17. Quit
1
Enter customer name:
john
Enter customer address:
redbud lane
Enter category
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion:

```
17. Quit
1
Enter customer name:
olivia
Enter customer address:
Prairie View Drive
Enter category
2
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
1
Enter customer name:
Alex
Enter customer address:
Thunderbird Avenue
Enter category
3
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
1
Enter customer name:
Yamin
Enter customer address:
Sooner Street
Enter category
4
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
1
Enter customer name:
Ashley
Enter customer address:
Cedar Ridge Road
Enter category
5
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion:

	customer_name	cust_address	category
1	Alex	Thunderbird Avenue	3
2	Ashley	Cedar Ridge Road	5
3	john	redbud lane	1
4	olivia	Prairie View Drive	2
5	Yamin	Sooner Street	4

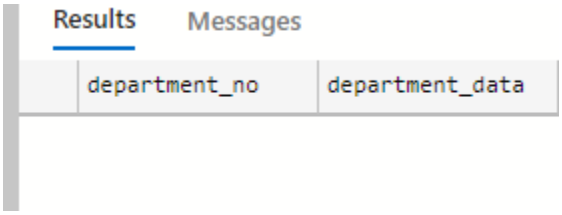
Below is the screenshot of more complete data which has been taken directly from AZURE.

Results Messages

	customer_name	cust_address	category
1	Abrar	Tulsa Circle	1
2	Alex	Thunderbird Avenue	3
3	Andres	Elmwood Avenue	4
4	Ashley	Cedar Ridge Road	5
5	Caleb	Mustang Lane	3
6	Ciarra	Pecan Lane	8
7	Emily	wildflower way	5
8	Jamie	Keystone Terrace	10
9	john	redbud lane	1
10	Kara	Oakwood Drive	7
11	Kendal	Rolling Hills Dri…	6
12	Logan	Cypress street	2
13	Nesma	Tulsa Turnpike	10
14	olivia	Prairie View Drive	2
15	Rebecca	Sunset Avenue	9
16	Ted	Bison Boulevard	6
17	Timmy	Sagebrush Court	5
18	Tombe	Cherokee Trail	7
19	Ximena	Bison Crossing	9
20	Yamin	Sooner Street	4

Screenshots of testing query 2:

Tables before insertion:



Inserting Department:

1st Insertion:

```
17. Quit
2
Enter department number
101
Enter department data:
DD1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion:

```
2
Enter department number
102
Enter department data:
DD2
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
2
Enter department number
103
Enter department data:
DD3
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
2
Enter department number
104
Enter department data:
DD4
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
2
Enter department number
105
Enter department data:
DD5
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

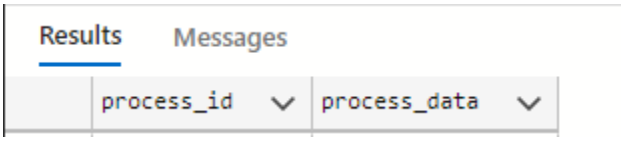
Screenshots of table after above insertion:

Results Messages

	department_no	department_data
1	101	DD1
2	102	DD2
3	103	DD3
4	104	DD4
5	105	DD5
6	106	DD6

Screenshots of testing query 3:

Tables before insertion:



1st Insertion

```
3
Enter process type:
FIT
Enter process id
1
Enter process data:
FIT1
Enter fit type:
THREADED FIT
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
101
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion

```
3
Enter process type:
CUT
Enter process id
2
Enter process data:
CUT1
Enter fit type:
NULL
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
JIGSAW CUTTING
Enter machine type:
JIGSAW POWER TOOL
Enter department number
102
Connecting to the database...
Dispatching the query...
```

3rd Insertion

```
17. Quit
3
Enter process type:
PAINT
Enter process id
3
Enter process data:
PAINT1
Enter fit type:
NULL
Enter paint type:
TYPE01
Enter painting method:
ROLLING
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
103
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion

```
17. Quit
3
Enter process type:
FIT
Enter process id
4
Enter process data:
FIT2
Enter fit type:
INTERLOCKING FIT
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
104
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion

```
17. Quit
3
Enter process type:
FIT
Enter process id
5
Enter process data:
FIT3
Enter fit type:
THREADED FIT
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
105
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

6th Insertion

```
3
Enter process type:
CUT
Enter process id
6
Enter process data:
CUT2
Enter fit type:
NULL
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
BAND CUTTING
Enter machine type:
BANDSHAW MACHINE
Enter department number
106
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

7th Insertion

```
3
Enter process type:
FIT
Enter process id
7
Enter process data:
FIT3
Enter fit type:
THREADED FIT
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
101
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

8th Insertion

```
3
Enter process type:
PAINT
Enter process id
8
Enter process data:
PAINT2
Enter fit type:
NULL
Enter paint type:
TYPE02
Enter painting method:
BRUSHING
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
103
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

9th Insertion

```
17. Quit
3
Enter process type:
PAINT
Enter process id
9
Enter process data:
PAINT3
Enter fit type:
NULL
Enter paint type:
TYPE03
Enter painting method:
AIRBRUSHING
Enter cut type:
NULL
Enter machine type:
NULL
Enter department number
102
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

10th Insertion

```
17. Quit
3
Enter process type:
CUT
Enter process id
10
Enter process data:
CUT3
Enter fit type:
NULL
Enter paint type:
NULL
Enter painting method:
NULL
Enter cut type:
LASER CUTTING
Enter machine type:
LASER
Enter department number
104
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion:

Results Messages

	process_id	process_data
1	1	FIT1
2	2	CUT1
3	3	PAINT1
4	4	FIT2
5	5	FIT3
6	6	CUT2
7	7	FIT3
8	8	PAINT2
9	9	PAINT3
10	10	CUT3

	process_id	department_no
1	1	101
2	2	102
3	3	103
4	4	104
5	5	105
6	6	106
7	7	101
8	8	103
9	9	102
10	10	104

	process_id	fit_type
1	1	THREADED FIT
2	4	INTERLOCKING FIT
3	5	THREADED FIT
4	7	THREADED FIT

	process_id	cutting_type	machine_type
1	2	JIGSAW CUTTING	JIGSAW POWER TOOL
2	6	BAND CUTTING	BANDSHAW MACHINE
3	10	LASER CUTTING	LASER

	process_id	paint_type	painting_method
1	3	TYPE01	ROLLING
2	8	TYPE02	BRUSHING
3	9	TYPE03	AIRBRUSHING

Below is the screenshot of more complete data which has been taken directly from AZURE.

Results Messages

	process_id	process_data
1	1	FIT1
2	2	CUT1
3	3	PAINT1
4	4	FIT2
5	5	FIT3
6	6	CUT2
7	7	FIT3
8	8	PAINT2
9	9	PAINT3
10	10	CUT3
11	11	paint4
12	12	cut4

Results Messages

	process_id	department_no
1	1	101
2	2	102
3	3	103
4	4	104
5	5	105
6	6	106
7	7	101
8	8	103
9	9	102
10	10	104
11	11	101
12	12	105

Results Messages

	process_id	fit_type
1	1	THREADED FIT
2	4	INTERLOCKING FIT
3	5	THREADED FIT
4	7	THREADED FIT

Results Messages

	process_id	cutting_type	machine_type
1	2	JIGSAW CUTTING	JIGSAW POWER TOOL
2	6	BAND CUTTING	BANDSHAW MACHINE
3	10	LASER CUTTING	LASER
4	12	Band Cutting	bandsaw Machine

Results Messages

	process_id	paint_type	painting_method
1	3	TYPE01	ROLLING
2	8	TYPE02	BRUSHING
3	9	TYPE03	AIRBRUSHING
4	11	type04	Rolling

Results Messages

	assembly_id	assembly_date	assembly_details

	assembly_id	process_id

	assembly_id	customer_name

Screenshots of testing query 4:

1st Insertion:

```
17. Quit
4
Enter assembly id
206
Enter assembly date:
2022-02-15
Enter assembly details:
details6
Enter the customer name:
Ted
Enter no of process:
3
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
206
enter the process id
6
Done. 1 rows inserted.
enter assembly id:
206
enter the process id
10
Done. 1 rows inserted.
enter assembly id:
206
enter the process id
4
Done. 1 rows inserted.
```

2nd Insertion:

```
17. Quit
4
Enter assembly id
201
Enter assembly date:
2022-01-01
Enter assembly details:
details1
Enter the customer name:
John
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
201
enter the process id
1
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
4
Enter assembly id
202
Enter assembly date:
2022-03-03
Enter assembly details:
details3
Enter the customer name:
olivia
Enter no of process:
2
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
202
enter the process id
1
Done. 1 rows inserted.
enter assembly id:
202
enter the process id
2
Done. 1 rows inserted.
```

4th Insertion:

```
17. Quit
4
Enter assembly id
203
Enter assembly date:
2022-04-11
Enter assembly details:
details3
Enter the customer name:
Alex
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
203
enter the process id
3
Done. 1 rows inserted.
```

5th Insertion:

```
17. Quit
4
Enter assembly id
204
Enter assembly date:
2022-05-25
Enter assembly details:
details4
Enter the customer name:
Yamin
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
204
enter the process id
4
Done. 1 rows inserted.
```

6th Insertion:

```
17. Quit
4
Enter assembly id
205
Enter assembly date:
2022-06-07
Enter assembly details:
details5
Enter the customer name:
Ashley
Enter no of process:
2
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
205
enter the process id
5
Done. 1 rows inserted.
enter assembly id:
205
enter the process id
2
Done. 1 rows inserted.
```

7th Insertion:

```
17. Quit
4
Enter assembly id
207
Enter assembly date:
2022-07-18
Enter assembly details:
details7
Enter the customer name:
kara
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
207
enter the process id
7
Done. 1 rows inserted.
```

8th Insertion:

```
17. Quit
4
Enter assembly id
208
Enter assembly date:
2022-08-30
Enter assembly details:
details8
Enter the customer name:
Ciarra
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
208
enter the process id
8
Done. 1 rows inserted.
```

9th Insertion:

```
4
Enter assembly id
209
Enter assembly date:
10/12/2022
Enter assembly details:
details9
Enter the customer name:
Rebbeca
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
209
enter the process id
9
Done. 1 rows inserted.
```

10th Insertion:

```
4
Enter assembly id
210
Enter assembly date:
2022-11-22
Enter assembly details:
details10
Enter the customer name:
Jamie
Enter no of process:
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
enter assembly id:
210
enter the process id
10
Done. 1 rows inserted.
```

Screenshots of table after above insertion

	assembly_id	assembly_date	assembly_details
1	201	2022-01-01	details1
2	202	2022-03-03	details3
3	203	2022-04-11	details3
4	204	2022-05-25	details4
5	205	2022-06-07	details5
6	206	2022-02-15	details6
7	207	2022-07-18	details7
8	208	2022-08-30	details8
9	209	10/12/2022	details9
10	210	2022-11-22	details10

	assembly_id	process_id
1	201	1
2	202	1
3	202	2
4	203	3
5	204	4
6	205	2
7	205	5
8	206	4
9	206	6
10	206	10

	assembly_id	customer_name
1	201	John
2	202	olivia
3	203	Alex
4	204	Yamin
5	205	Ashley
6	206	Ted
7	207	kara
8	208	Ciarra
9	210	Jamie

Screenshots of testing query 5:

Tables before insertion:

1st Insertion:

```
17. Quit
5
Enter account number:
1111
Enter account date:
2022-01-01
Enter acc associated to:
process
Enter details2 for process:
1000
Enter details1 for assembly
0
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
1
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion:

```
17. Quit
5
Enter account number:
2222
Enter account date:
2022-02-15
Enter acc associated to:
assembly
Enter details2 for process:
0
Enter details1 for assembly
450
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
0
Enter assembly id:
201
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
5
Enter account number:
3333
Enter account date:
2022-03-03
Enter acc associated to:
department
Enter details2 for process:
0
Enter details1 for assembly
0
Enter details3 for dept:
1250
Enter department number:
103
Enter process id:
0
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
17. Quit
5
Enter account number:
4444
Enter account date:
2022-04-11
Enter acc associated to:
assembly
Enter details2 for process:
0
Enter details1 for assembly
320
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
0
Enter assembly id:
202
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
17. Quit
5
Enter account number:
5555
Enter account date:
2022-05-25
Enter acc associated to:
process
Enter details2 for process:
200
Enter details1 for assembly
0
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
5
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

6th Insertion:

```
17. Quit
5
Enter account number:
6666
Enter account date:
2022-06-07
Enter acc associated to:
process
Enter details2 for process:
300
Enter details1 for assembly
0
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
6
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

7th Insertion:

```
17. Quit
5
Enter account number:
8888
Enter account date:
2022-08-30
Enter acc associated to:
assembly
Enter details2 for process:
0
Enter details1 for assembly
697
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
0
Enter assembly id:
203
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

8th Insertion:

```
17. Quit
5
Enter account number:
7777
Enter account date:
2022-07-18
Enter acc associated to:
department
Enter details2 for process:
0
Enter details1 for assembly
0
Enter details3 for dept:
200
Enter department number:
105
Enter process id:
0
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

9th Insertion:

```
17. Quit
5
Enter account number:
9999
Enter account date:
2022-10-12
Enter acc associated to:
department
Enter details2 for process:
0
Enter details1 for assembly
0
Enter details3 for dept:
467
Enter department number:
101
Enter process id:
0
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

10th Insertion:

```
17. Quit
5
Enter account number:
1000
Enter account date:
2022-11-22
Enter acc associated to:
process
Enter details2 for process:
840
Enter details1 for assembly
0
Enter details3 for dept:
0
Enter department number:
0
Enter process id:
10
Enter assembly id:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion

Results Messages

	acc_no	acc_date
1	1000	2022-11-22
2	1111	2022-01-01
3	2222	2022-02-15
4	3333	2022-03-03
5	4444	2022-04-11
6	5555	2022-05-25
7	6666	2022-06-07
8	7777	2022-07-18
9	8888	2022-08-30
10	9999	2022-10-12

	acc_no	details_1
1	2222	450
2	4444	320
3	8888	697

	acc_no	assembly_id
1	2222	201
2	4444	202
3	8888	203

	acc_no	details_2
1	1000	840
2	1111	1000
3	5555	200
4	6666	300

	acc_no	process_id
1	1111	1
2	5555	5
3	6666	6
4	1000	10

	acc_no	details_3
1	3333	1250
2	7777	200
3	9999	467

	acc_no	department_no
1	9999	101
2	3333	103
3	7777	105

Screenshots of testing query 6:

Tables before insertion:

Results Messages

	job_no	commenced_date	completed_date

	job_no	assembly_id	process_id

Insertion1

```
16. Export: Retrieve the customers
17. Quit
6
enter the job no
1
Enter job commenced date:
2022-10-11

Enter assembly id
201
Enter process id
1
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.

Please select one of the options be
```

2nd Insertion:

```
17. Quit
6
enter the job no
2
Enter job commenced date:
2022-09-12

Enter assembly id
202
Enter process id
2
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
6
enter the job no
3
Enter job commenced date:
2022-08-03

Enter assembly id
203
Enter process id
3
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
17. Quit
6
enter the job no
4
Enter job commenced date:
2022-07-15

Enter assembly id
204
Enter process id
4
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
17. Quit
6
enter the job no
5
Enter job commenced date:
2022-02-03

Enter assembly id
205
Enter process id
5
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

6th Insertion:

```
17. Quit
6
enter the job no
6
Enter job commenced date:
2022-01-03

Enter assembly id
206
Enter process id
6
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

7th Insertion:

```
17. Quit
6
enter the job no
7
Enter job commenced date:
2022-08-06

Enter assembly id
207
Enter process id
7
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

8th Insertion

```
16. Export: Retrieve the customer
17. Quit
6
enter the job no
8
Enter job commenced date:
2022-03-14

Enter assembly id
208
Enter process id
8
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

9th Insertion

```
17. Quit
6
enter the job no
9
Enter job commenced date:
2022-04-25

Enter assembly id
206
Enter process id
9
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

10th Insertion

```
17. Quit
6
enter the job no
10
Enter job commenced date:
2022-05-31

Enter assembly id
205
Enter process id
10
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion

Results Messages

	job_no	commenced_date	completed_date
1	1	1900-01-01	NULL
2	2	1900-01-01	NULL
3	3	1900-01-01	NULL
4	4	1900-01-01	NULL
5	5	1900-01-01	NULL
6	6	1900-01-01	NULL
7	7	1900-01-01	NULL
8	8	1900-01-01	NULL
9	9	1900-01-01	NULL
10	10	1900-01-01	NULL

	job_no	assembly_id	process_id
1	1	201	1
2	2	202	2
3	3	203	3
4	4	204	4
5	5	205	5
6	6	206	6
7	7	207	7
8	8	208	8
9	9	206	9
10	10	205	10

Results Messages

	job_no	commenced_date	completed_date
3	3	1900-01-01	NULL
4	4	1900-01-01	NULL
5	5	1900-01-01	NULL
6	6	1900-01-01	NULL
7	7	1900-01-01	NULL
8	8	1900-01-01	NULL
9	9	1900-01-01	NULL
10	10	1900-01-01	NULL
11	11	2022-09-18	NULL
12	12	2022-06-11	NULL
13	13	2022-04-16	NULL
14	14	2022-03-29	NULL
15	15	2022-06-17	NULL

	job_no	assembly_id	process_id
3	3	203	3
4	4	204	4
5	5	205	5
6	6	206	6
7	7	207	7
8	8	208	8
9	9	206	9
10	10	205	10
11	11	204	11
12	12	203	12
13	13	202	3
14	14	201	4
15	15	208	8

Screenshots of testing query 7:

Tables before insertion:

Results Messages

	job_no	commenced_date	completed_date
1	1	1900-01-01	NULL
2	2	1900-01-01	NULL
3	3	1900-01-01	NULL
4	4	1900-01-01	NULL
5	5	1900-01-01	NULL
6	6	1900-01-01	NULL
7	7	1900-01-01	NULL
8	8	1900-01-01	NULL
9	9	1900-01-01	NULL
10	10	1900-01-01	NULL

	job_no	labor_time

	job_no	color	Volume	labor_time

1st Insertion:

```
17. Quit
7
enter the job no
1
Enter job completed date:
2022-12-10
Enter job type
paint job
Enter labor time
12:00:00
Enter color
blue
Enter volume
150
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion:

```
17. Quit
7
enter the job no
2
Enter job completed date:
2022-11-15
Enter job type
fit job
Enter labor time
08:00:00
Enter color
NULL
Enter volume
0
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
7
enter the job no
3
Enter job completed date:
2022-12-4
Enter job type
cut job
Enter labor time
07:30:00
Enter color
NULL
Enter volume
0
Enter type of machine
jigsaw power tool
Enter amount of time:
12:30:00
Enter the material:
wood
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
17. Quit
7
enter the job no
4
Enter job completed date:
2022-8-19
Enter job type
fit job
Enter labor time
09:00:00
Enter color
NULL
Enter volume
0
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
17. Quit
7
enter the job no
5
Enter job completed date:
2022-5-4
Enter job type
paint job
Enter labor time
10:00:00
Enter color
grey
Enter volume
125
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

6th Insertion:

```
17. Quit
7
enter the job no
6
Enter job completed date:
2022-10-4
Enter job type
fit job
Enter labor time
11:30:00
Enter color
NULL
Enter volume
0
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

7th Insertion:

```
17. Quit
7
enter the job no
7
Enter job completed date:
2022-10-7
Enter job type
cut job
Enter labor time
05:00:00
Enter color
NULL
Enter volume
0
Enter type of machine
bandsaw machine
Enter amount of time:
4:00:00
Enter the material:
wood
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

8th Insertion:

```
17. Quit
7
enter the job no
8
Enter job completed date:
2022-7-15
Enter job type
fit job
Enter labor time
07:30:00
Enter color
NULL
Enter volume
0
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

9th Insertion:

```
17. Quit
7
enter the job no
9
Enter job completed date:
2022-6-26
Enter job type
cut job
Enter labor time
08:00:00
Enter color
NULL
Enter volume
0
Enter type of machine
jigsaw power tool
Enter amount of time:
11:00:00
Enter the material:
wood
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

10th Insertion:

```
17. Quit
7
enter the job no
10
Enter job completed date:
2022-8-1
Enter job type
paint job
Enter labor time
09:30:00
Enter color
red
Enter volume
140
Enter type of machine
NULL
Enter amount of time:
NULL
Enter the material:
NULL
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion

Results Messages

	job_no	commenced_date	completed_date
1	1	1900-01-01	2022-12-10
2	2	1900-01-01	2022-11-15
3	3	1900-01-01	2022-12-04
4	4	1900-01-01	2022-08-19
5	5	1900-01-01	2022-05-04
6	6	1900-01-01	2022-10-04
7	7	1900-01-01	2022-10-07
8	8	1900-01-01	2022-07-15
9	9	1900-01-01	2022-06-26
10	10	1900-01-01	2022-08-01

	job_no	labor_time
1	2	08:00:00
2	4	09:00:00
3	6	11:30:00
4	8	07:30:00

	job_no	color	Volume	labor_time
1	1	blue	150	12:00:00
2	5	grey	125	10:00:00
3	10	red	140	09:30:00

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	3	jigsaw power tool	12:30:00	wood	07:30:00
2	7	bandsaw machine	04:00:00	wood	05:00:00
3	9	jigsaw power tool	11:00:00	wood	08:00:00

Below is the screenshot of more complete data which has been taken directly from AZURE.

Results Messages

	job_no	commenced_date	completed_date
6	6	1900-01-01	2022-10-04
7	7	1900-01-01	2022-10-07
8	8	1900-01-01	2022-07-15
9	9	1900-01-01	2022-06-26
10	10	1900-01-01	2022-08-01
11	11	2022-09-18	2022-11-19
12	12	2022-06-11	2022-12-12
13	13	2022-04-16	2022-05-17
14	14	2022-03-29	2022-02-26
15	15	2022-06-17	2022-09-18

	job_no	labor_time
1	2	08:00:00
2	4	09:00:00
3	6	11:30:00
4	8	07:30:00
5	12	12:30:00
6	14	05:00:00

	job_no	color	Volume	labor_time
1	1	blue	150	12:00:00
2	5	grey	125	10:00:00
3	10	red	140	09:30:00
4	13	grey	110	06:00:00

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	3	jigsaw power tool	12:30:00	wood	07:30:00
2	7	bandsaw machine	04:00:00	wood	05:00:00
3	9	jigsaw power tool	11:00:00	wood	08:00:00
4	11	Laser	06:30:00	glass	10:00:00
5	15	Laser	03:00:00	glass	08:30:00

Screenshots of testing query 8:

Tables before insertion:

Results Messages

	acc_no	details_3
1	3333	1250
2	7777	200
3	9999	467

	acc_no	details_2
1	1000	840
2	1111	1000
3	5555	200
4	6666	300

	acc_no	details_1
1	2222	450
2	4444	320
3	8888	697

1st Insertion:

```
17. Quit
8
enter the transaction number
123
Enter supcost
150
Enter account number
1111
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

2nd Insertion:

```
17. Quit
8
enter the transaction number
456
Enter supcost
200
Enter account number
222
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

3rd Insertion:

```
17. Quit
8
enter the transaction number
105
Enter supcost
250
Enter account number
333
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

4th Insertion:

```
17. Quit
8
enter the transaction number
230
Enter supcost
300
Enter account number
4444
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

5th Insertion:

```
17. Quit
8
enter the transaction number
984
Enter supcost
350
Enter account number
5555
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

6th Insertion:

```
17. Quit
8
enter the transaction number
420
Enter supcost
400
Enter account number
6666
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

7th Insertion:

```
17. Quit
8
enter the transaction number
134
Enter supcost
450
Enter account number
7777
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

8th Insertion:

```
17. Quit
8
enter the transaction number
145
Enter supcost
500
Enter account number
8888
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

9th Insertion:

```
17. Quit
8
enter the transaction number
279
Enter supcost
550
Enter account number
9999
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

10th Insertion:

```
17. Quit
8
enter the transaction number
326
Enter supcost
600
Enter account number
1000
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Screenshots of table after above insertion

Results Messages

	acc_no	details_3
1	3333	1250
2	7777	650
3	9999	1017

	acc_no	details_2
1	1000	1440
2	1111	1150
3	5555	550
4	6666	700

	acc_no	details_1
1	2222	450
2	4444	620
3	8888	1197

Screenshots of testing query 9:

1st Insertion:

```
17. Quit
9
Enter assembly id:
201
Connecting to the database...
results of query 9:

Total cost on assembly id:

total cost

450
```

2nd Insertion:

```
17. Quit
9
Enter assembly id:
202
Connecting to the database...
results of query 9:

Total cost on assembly id:

total cost

620
```

3rd Insertion:

```
17. Quit
9
Enter assembly id:
203
Connecting to the database...
results of query 9:

Total cost on assembly id:

total cost

1197
```

Screenshots of testing query 11:

1st Insertion:

```
17. Quit
11
Enter assembly id:
206
Connecting to the database...
results of query 11:

Contents of the customer table:

Commenced date | Process_id     | Department_no

1900-01-01       |      6          |       106
1900-01-01       |      9          |       102
```

2nd Insertion:

```
11
Enter assembly id:
203
Connecting to the database...
results of query 11:

Contents of the customer table:

Commenced date | Process_id     | Department_no

1900-01-01       |      3          |       103
2022-06-11       |      12         |       105
```

3rd Insertion:

```
11
Enter assembly id:
208
Connecting to the database...
results of query 11:

Contents of the customer table:

Commenced date | Process_id     | Department_no

1900-01-01       |      8          |       103
2022-06-17       |      8          |       103
```

Screenshots of testing query 12:

1st Insertion:

```
12
Enter low category:
1
Enter high category:
4
Connecting to the database...
results of query 12:

Contents of the customer table:

cust_name | cust_address               | category

Abrar   |       Tulsa Circle       |       1
Alex   |       Thunderbird Avenue         |       3
Andres   |      Elmwood Avenue      |       4
Caleb   |       Mustang Lane       |       3
john   |       redbud lane        |       1
Logan   |       Cypress street     |       2
olivia   |      Prairie View Drive         |       2
Yamin   |       Sooner Street      |       4
```

2nd Insertion:

```
17. Quit
12
Enter low category:
3
Enter high category:
7
Connecting to the database...
results of query 12:

Contents of the customer table:

cust_name | cust_address                | category

Alex   |        Thunderbird Avenue         |       3
Andres   |      Elmwood Avenue      |       4
Ashley   |      Cedar Ridge Road           |       5
Caleb   |       Mustang Lane        |       3
Emily   |       wildflower way      |       5
Kara   |        Oakwood Drive       |       7
Kendal   |      Rolling Hills Drive        |       6
Ted   |         Bison Boulevard     |       6
Timmy   |       Sagebrush Court     |       5
Tombe   |        Cherokee Trail     |       7
Yamin   |       Sooner Street       |       4
```

3rd Insertion:

```
17. Quit
12
Enter low category:
5
Enter high category:
9
Connecting to the database...
results of query 12:

Contents of the customer table:

cust_name | cust_address                | category

Ashley   |      Cedar Ridge Road           |       5
Ciarra   |       Pecan Lane         |       8
Emily   |       wildflower way      |       5
Kara   |        Oakwood Drive       |       7
Kendal   |      Rolling Hills Drive        |       6
Rebecca   |     Sunset Avenue       |       9
Ted   |         Bison Boulevard     |       6
Timmy   |       Sagebrush Court     |       5
Tombe   |        Cherokee Trail     |       7
Ximena   |      Bison Crossing      |       9
```

Screenshots of testing query 13:

Tables before insertion:

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	3	jigsaw power tool	12:30:00	wood	07:30:00
2	7	bandsaw machine	04:00:00	wood	05:00:00
3	9	jigsaw power tool	11:00:00	wood	08:00:00
4	11	Laser	06:30:00	glass	10:00:00
5	15	Laser	03:00:00	glass	08:30:00

Insertion1:

```
17. Quit
13
Enter low jobno:
3
Enter high jobno:
6
Connecting to the database...
Dispatching the query...
Done. rows deleted.
```

Output

Results Messages

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	7	bandsaw machine	04:00:00	wood	05:00:00
2	9	jigsaw power tool	11:00:00	wood	08:00:00
3	11	Laser	06:30:00	glass	10:00:00
4	15	Laser	03:00:00	glass	08:30:00

Insertion2:

```
17. Quit
13
Enter low jobno:
7
Enter high jobno:
10
Connecting to the database...
Dispatching the query...
Done. rows deleted.
```

Output

Results Messages

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	11	Laser	06:30:00	glass	10:00:00
2	15	Laser	03:00:00	glass	08:30:00

Insertion 3:

```
17. Quit
13
Enter low jobno:
14
Enter high jobno:
20
Connecting to the database...
Dispatching the query...
Done. rows deleted.
```

Output

Results Messages

	job_no	type_of_machine_used	amount_of_time	material	labor_time
1	11	Laser	06:30:00	glass	10:00:00

Screenshots of testing query 14:

Results Messages

	job_no	color	Volume	labor_time
1	1	blue	150	12:00:00
2	5	grey	125	10:00:00
3	10	red	140	09:30:00
4	13	grey	110	06:00:00

Insertion1:

```
17. Quit
14
enter the job no
1
Enter color to be changed
pink
Connecting to the database...
Dispatching the query...
Done. 1 row updated.
```

Insertion 2:

```
17. Quit
14
enter the job no
10
Enter color to be changed
green
Connecting to the database...
Dispatching the query...
Done. 1 row updated.
```

Insertion 3:

```
17. Quit
14
enter the job no
13
Enter color to be changed
orange
Connecting to the database...
Dispatching the query...
Done. 1 row updated.
```

Output:

Results Messages

	job_no	color	Volume	labor_time
1	1	pink	150	12:00:00
2	5	grey	125	10:00:00
3	10	green	140	09:30:00
4	13	orange	110	06:00:00

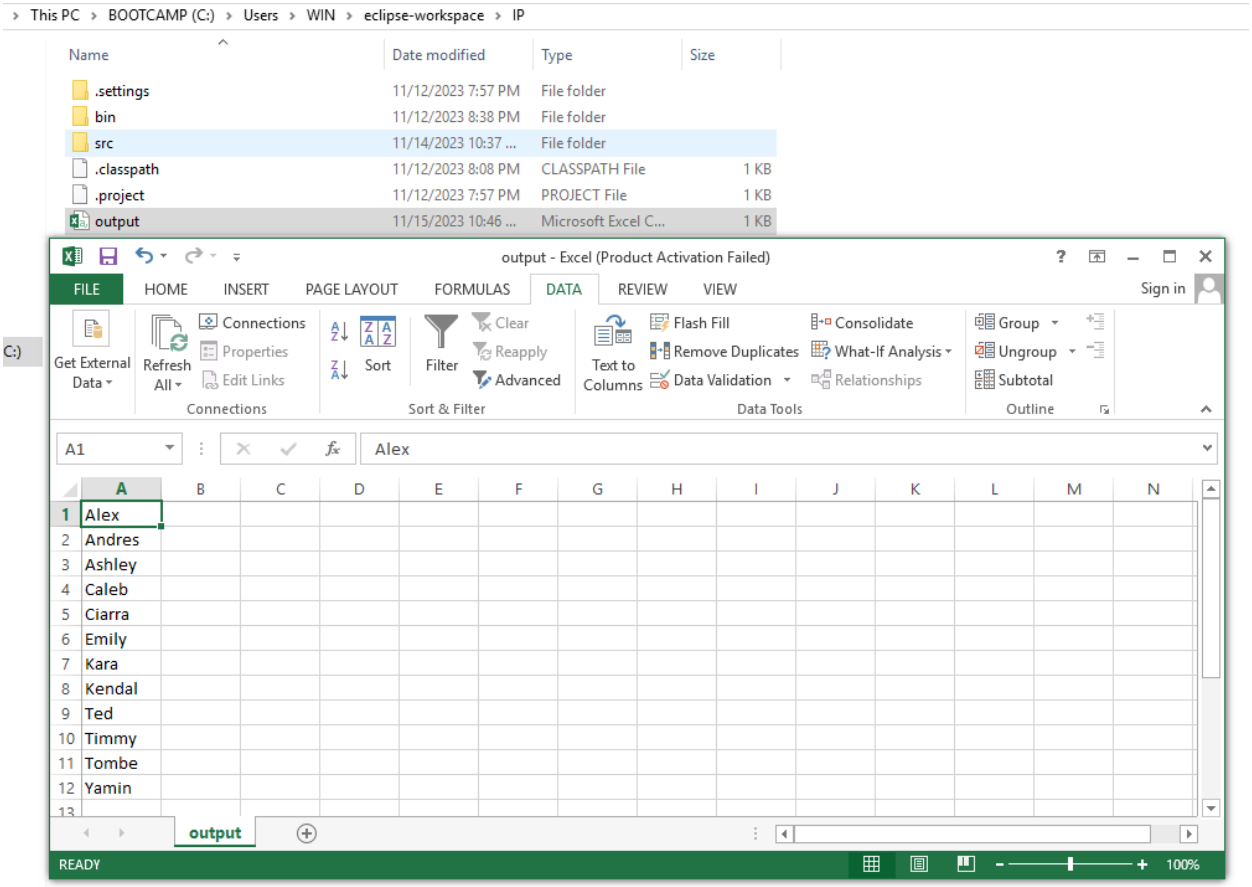
Query15:

Screenshots of testing query 16:

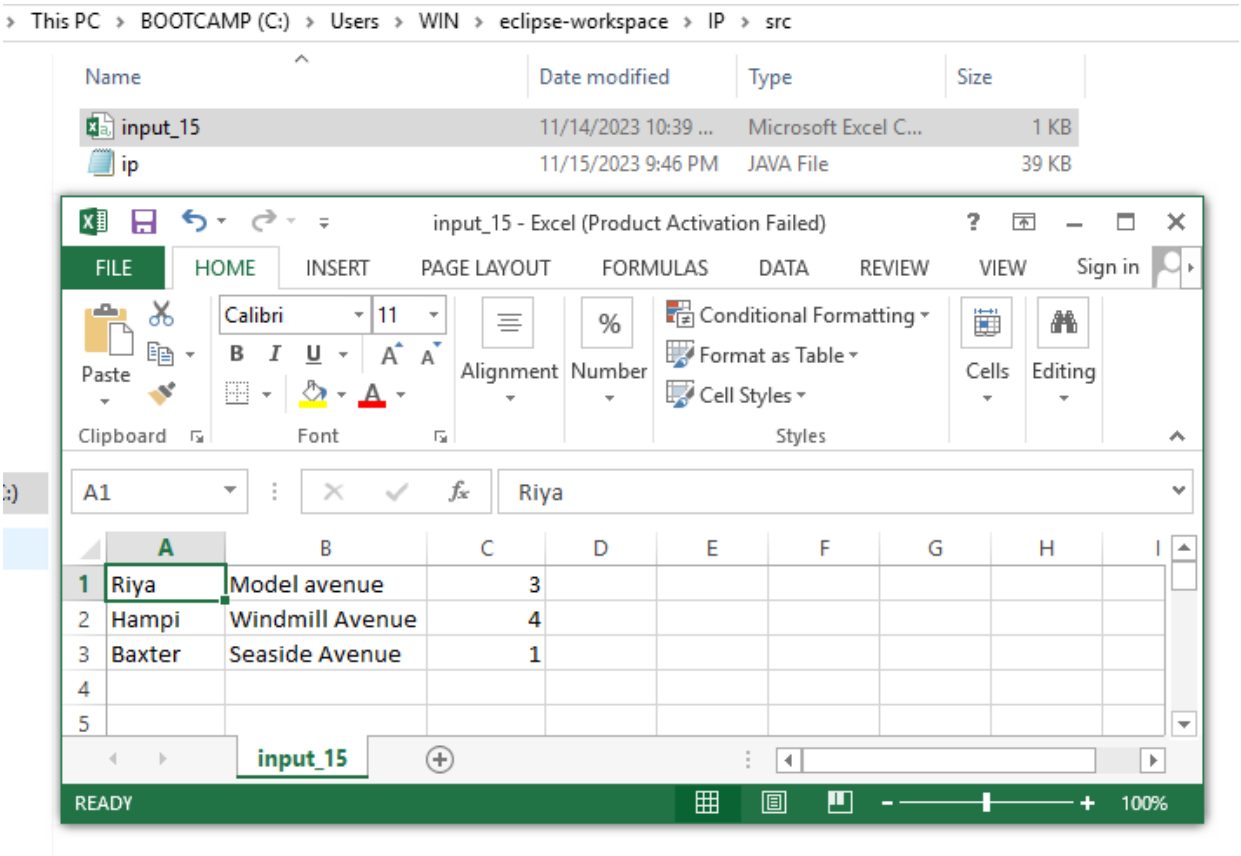
Insertion1:

```
16
Enter the file name
output.csv
Enter lower category number
3
Enter high category number
8
```

Output:


This PC › BOOTCAMP (C:) › Users › WIN › eclipse-workspace › IP
Name
Date modified
Type
Size
.settings
11/12/2023 7:57 PM
File folder
bin
11/12/2023 8:38 PM
File folder
src
11/14/2023 10:37 ...
File folder
.classpath
11/12/2023 8:08 PM
CLASSPATH File
1 KB
.project
11/12/2023 7:57 PM
PROJECT File
1 KB
output
11/15/2023 10:46 ...
Microsoft Excel C...
1 KB
output - Excel (Product Activation Failed)
FILE
HOME
INSERT
PAGE LAYOUT
FORMULAS
DATA
REVIEW
VIEW
Sign in
Get External Data
Refresh All
Connections
Properties
Edit Links
Sort
Filter
Clear
Reapply
Advanced
Text to Columns
Flash Fill
Remove Duplicates
Data Validation
Consolidate
What-If Analysis
Relationships
Group
Ungroup
Subtotal
Connections
Sort & Filter
Data Tools
Outline
A1
Alex
Andres
Ashley
Caleb
Ciarra
Emily
Kara
Kendal
Ted
Timmy
Tombe
Yamin
output
READY
100%

Query16:


This PC › BOOTCAMP (C:) › Users › WIN › eclipse-workspace › IP › src
Name
Date modified
Type
Size
input_15
11/14/2023 10:39 ...
Microsoft Excel C...
1 KB
ip
11/15/2023 9:46 PM
JAVA File
39 KB
input_15 - Excel (Product Activation Failed)
FILE
HOME
INSERT
PAGE LAYOUT
FORMULAS
DATA
REVIEW
VIEW
Sign in
Paste
Calibri
11
Alignment
Number
Conditional Formatting
Format as Table
Cell Styles
Cells
Editing
Clipboard
Font
Styles
A1
Riya
A
B
C
D
E
F
G
H
1 Riya Model avenue 3
2 Hampi Windmill Avenue 4
3 Baxter Seaside Avenue 1
4
5
input_15
READY
100%

Screenshots of testing query 17:

```
17
Quiting! Good-bye!
```

Detecting Errors:

Error 1: Primary Key Error

```
16. Export: Retrieve the customers (in n
17. Quit
1
Enter customer name:
John
Enter customer address:
norman
Enter category
5
Connecting to the database...
Dispatching the query...
error in case 1
```

Error 2: Incorrect Data type related Insertion

```
17. Quit
7
enter the job no
16
Enter job completed date:
2022-06-08
Enter job type
3
Enter labor time
high
Enter color
null
Enter volume
null
Enter type of machine
null
Enter amount of time:
null
Enter the material:
null
Connecting to the database...
Dispatching the query...
com.microsoft.sqlserver.jdbc.SQLServerException: Error converting data type nvarchar to int.
```

Error 3: Foreign key reference error

```
7
enter the job no
18
Enter job completed date:
2022-09-08
Enter job type
job
Enter labor time
12:00:00
Enter color
null
Enter volume
0
Enter type of machine
null
Enter amount of time:
null
Enter the material:
null
Connecting to the database...
Dispatching the query...
Done. 0 rows inserted.
```

Task 7: Web database application and its execution

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

Screenshots of successful compilation:

1) Data handler

DataHandler.java | Peri_Aishwarya_IP_Task7_ad... | Peri_Aishwarya_IP_Task7_ad... | Peri_Aishwarya_IP_Task7_get... | Peri_Aishwarya_IP_Task7_retr... | Peri_Aishwarya_IP_Task7_retr...

```java
package Aishwarya;

import java.sql.Statement;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
public class DataHandler {
    private Connection conn;
    // Azure SQL connection credentials
    private String server = "peri0015-sql-server.database.windows.net";
    private String database = "cs-dsa-4513-sql-db";
    private String username = "peri0015";
    private String password = "Gokaraju@0525";

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

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

    // Return the result of selecting everything from the movie_night table
    public ResultSet getAllCustomers() throws SQLException {
        getDBConnection();
        final String sqlQuery = "select * from customer;"; // query same as case 12
        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
        return stmt.executeQuery();
    }

    // Inserts a record into the movie_night table with the given attribute values
    public boolean addCustomer(
        String customer_name, String cust_address, int category) throws SQLException {
        getDBConnection(); // Prepare the database connection
        // Prepare the SQL statement
        final String sqlQuery =
        "INSERT INTO customer (customer_name, cust_address, category) " +
        "VALUES " +
        "(?, ?, ?)"; // query same as case 1
        final PreparedStatement stmt = conn.prepareStatement(sqlQuery);
        // Replace the '?' in the above statement with the given attribute values
        stmt.setString(1, customer_name);
        stmt.setString(2, cust_address);
        stmt.setInt(3, category);

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

    // Return the result of selecting everything from the movie_night table
    public ResultSet getretriveCustomers(int low_cat, int high_cat) throws SQLException {
        getDBConnection();
        System.out.println(low_cat);
        final String sqlQuery = "select * from customer where customer.category >= "+low_cat+" and category <= "+high_cat+" order by customer.customer_name;"; // query s
        Statement stmt = conn.createStatement();

        return stmt.executeQuery(sqlQuery);
    }
}
```

Activate Win
Go to Settings t

2) Add customer form

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Add New Customers</title>
    </head>
    <body>
        <h2>Add new customer data in the fields</h2>
        <!--
            Form for collecting user input for the customer record.
            Upon form submission, add_movie.jsp file will be invoked.
        -->
        <form action="Peri_Aishwarya_IP_Task7_add_customer.jsp">
            <!-- The form organized in an HTML table for better clarity. -->
            <table border=1>
                <tr>
                    <th colspan="2">Enter the Customer Data:</th>
                </tr>
                <tr>
                    <td>Customer name:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=customer_name>
                    </div></td>
                </tr>
                <tr>
                    <td>Customer address:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=cust_address>
                    </div></td>
                </tr>
                <tr>
                    <td>Category from 1 - 10:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=category>
                    </div></td>
                </tr>

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

3) Add Customer

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Query Result</title>
</head>
    <body>
    <%@page import="Aishwarya.DataHandler"%>
    <%@page import="java.sql.ResultSet"%>
    <%@page import="java.sql.Array"%>
    <%
    // The handler is the one in charge of establishing the connection.
    DataHandler handler1 = new DataHandler();

    // Get the attribute values passed from the input form.
    String customer_name = request.getParameter("customer_name");
    String cust_address = request.getParameter("cust_address");
    String category = request.getParameter("category");

    /*
     * If the user hasn't filled out all the customer name, customer address and category. This is very simple checking.
     */
    if (customer_name.equals("") || cust_address.equals("") || category.equals("")) {
        response.sendRedirect("Peri_Aishwarya_IP_Task7_add_customer_form.jsp");
    } else {
        int category1 = Integer.parseInt(category);

        // Now perform the query with the data from the form.
        boolean success = handler1.addCustomer(customer_name, cust_address, category1);
        if (!success) { // Something went wrong
            %>
                <h2>There was a problem inserting the course</h2>
            <%
            } else { // Confirm success to the user
            %>
            <h2>Customer details:</h2>

            <ul>
                <li>Customer Name: <%=customer_name%></li>
                <li>Customer Address: <%=cust_address%></li>
                <li>Category: <%=category%></li>

            </ul>
```

```jsp
                <li>Category: <%=category%></li>

            </ul>

            <h2>Was successfully inserted.</h2>

            <a href="Peri_Aishwarya_IP_Task7_get_all_customers.jsp">See all Customers.</a>
            <%
        }
    }
    %>
    </body>
</html>
```

4) Get customer

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
        <title>Customers</title>
    </head>
    <body>
        <%@page import="Aishwarya.DataHandler"%>
        <%@page import="java.sql.ResultSet"%>
        <%
            // We instantiate the data handler here, and get all the customers from the database
            final DataHandler handler = new DataHandler();
            final ResultSet customers = handler.getAllCustomers();
        %>
        <!-- The table for displaying all the movie records -->
        <table cellspacing="2" cellpadding="2" border="1">
            <tr> <!-- The table headers row -->
              <td align="center">
                <h4>Customer</h4>
              </td>
              <td align="center">
                <h4>Customer address</h4>
              </td>
              <td align="center">
                <h4>category</h4>
              </td>

            </tr>
            <%
               while(customers.next()) { // For each customer record returned...
                    // Extract the attribute values for every row returned
                    final String customer_name = customers.getString("customer_name");
                    final String cust_address = customers.getString("cust_address");
                    final String category = customers.getString("category");


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

5) Retrieve customer form

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Retrieve customers from given category </title>
    </head>
    <body>
        <h2>enter category range from 1 - 10</h2>
        <!--
            Form for collecting user input for the customer record.
            Upon form submission, add_movie.jsp file will be invoked.
        -->
        <form action="Peri_Aishwarya_IP_Task7_retrive_customer.jsp">
            <!-- The form organized in an HTML table for better clarity. -->
            <table border=1>
                <tr>
                    <th colspan="2">category range from 1 - 10</th>
                </tr>
                <tr>
                    <td>low_cat</td>
                    <td><div style="text-align: center;">
                    <input type=text name=start_time>
                    </div></td>
                </tr>
                <tr>
                    <td>high_cat</td>
                    <td><div style="text-align: center;">
                    <input type=text name=movie_name>
                    </div></td>
                </tr>
                <tr>
                    <td><div style="text-align: center;">
                    <input type=reset value=Clear>
                    </div></td>
                    <td><div style="text-align: center;">
                    <input type=submit value=Insert>
                    </div></td>
                </tr>
            </table>
        </form>
    </body>
</html>

```

6) Retrive customer

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
        <title>Customers</title>
    </head>
    <body>
        <%@page import="Aishwarya.DataHandler"%>
        <%@page import="java.sql.ResultSet"%>
        <%
            // We instantiate the data handler here, and get all the customers from the database
            final DataHandler handler1 = new DataHandler();

            //String low_cat  = request.getParameter("low_cat");
            //String high_cat = request.getParameter("high_cat");

            int low_cat = Integer.parseInt(request.getParameter("low_cat"));
            System.out.println(low_cat);
            int high_cat = Integer.parseInt(request.getParameter("high_cat"));

            final ResultSet customers = handler1.getretriveCustomers(low_cat,high_cat);

        %>
        <!-- The table for displaying all the movie records -->
        <table cellspacing="2" cellpadding="2" border="1">
            <tr> <!-- The table headers row -->
              <td align="center">
                <h4>Customer name</h4>
              </td>
              <td align="center">
                <h4>Cust_address</h4>
              </td>
              <td align="center">
                <h4>Category</h4>
              </td>


            </tr>
            <%
            while(customers.next()) { // For each customer record returned...
                // Extract the attribute values for every row returned
                final String customer_name = customers.getString("customer_name");
                final String cust_address = customers.getString("cust_address");
                final int category = customers.getInt("category");

```

```
46                 final int category = customers.getInt("category");
47
48
49                 out.println("<tr>"); // Start printing out the new table row
50                 out.println( // Print each attribute value
51                      "<td align=\"center\">" + customer_name +
52                      "</td><td align=\"center\"> " + cust_address +
53                      "</td><td align=\"center\"> " + category + "</td>");
54                 out.println("</tr>");
55             }
56                %>
57           </table>
58     </body>
59 </html>
60
```

7.2 Screenshots showing the testing of the Web database application

1. Add customer form:

enter category range from 1 - 10

category range from 1 - 10	
low_cat	
high_cat	
Clear	Insert

For 12: 12. Retrieve the customers (in name order) whose category is in a given range

enter category range from 1 - 10

category range from 1 - 10	
low_cat	4
high_cat	9
Clear	Insert

Customer	Customer address	category
Andres	Elmwood Avenue	4
Ashley	Cedar Ridge Road	5
Ciarra	Pecan Lane	8
Emily	wildflower way	5
Jamie	Keystone Terrace	10
Kara	Oakwood Drive	7
Kendal	Rolling Hills Drive	6
Rebecca	Sunset Avenue	9
ronda	india	5
Ted	Bison Boulevard	6
Timmy	Sagebrush Court	5
Tombe	Cherokee Trail	7
Ximena	Bison Crossing	9
Yamin	Sooner Street	4

For 1: Enter a new customer

Add new customer data in the fields

Enter the Customer Data:	
Customer name:	Aish
Customer address:	india
Category from 1 - 10:	6
Clear	Insert

Customer details:

- Customer Name: Aish
- Customer Address: india
- Category: 6

Was successfully inserted.

See all Customers.

Customer	Customer address	category
Abrar	Tulsa Circle	1
Aish	india	6
Alex	Thunderbird Avenue	3
Andres	Elmwood Avenue	4
Ashley	Cedar Ridge Road	5
Caleb	Mustang Lane	3
Ciarra	Pecan Lane	8
Emily	wildflower way	5
Jamie	Keystone Terrace	10
John	Redbud Lane	1
Kara	Oakwood Drive	7
Kendal	Rolling Hills Drive	6
Logan	Cypress street	2
Nesma	Tulsa Turnpike	10
Olivia	Prairie View Drive	2
Rebecca	Sunset Avenue	9
ronda	india	5
Ted	Bison Boulevard	6
Timmy	Sagebrush Court	5
Tombe	Cherokee Trail	7
Ximena	Bison Crossing	9
Yamin	Sooner Street	4

For 12:

enter category range from 1 - 10

category range from 1 - 10	
low_cat	4
high_cat	9
Clear	Insert

Customer	Customer address	category
Aish	india	6
Andres	Elmwood Avenue	4
Ashley	Cedar Ridge Road	5
Ciarra	Pecan Lane	8
Emily	wildflower way	5
Jamie	Keystone Terrace	10
Kara	Oakwood Drive	7
Kendal	Rolling Hills Drive	6
Rebecca	Sunset Avenue	9
ronda	india	5
Ted	Bison Boulevard	6
Timmy	Sagebrush Court	5
Tombe	Cherokee Trail	7
Ximena	Bison Crossing	9
Yamin	Sooner Street	4

Thank you