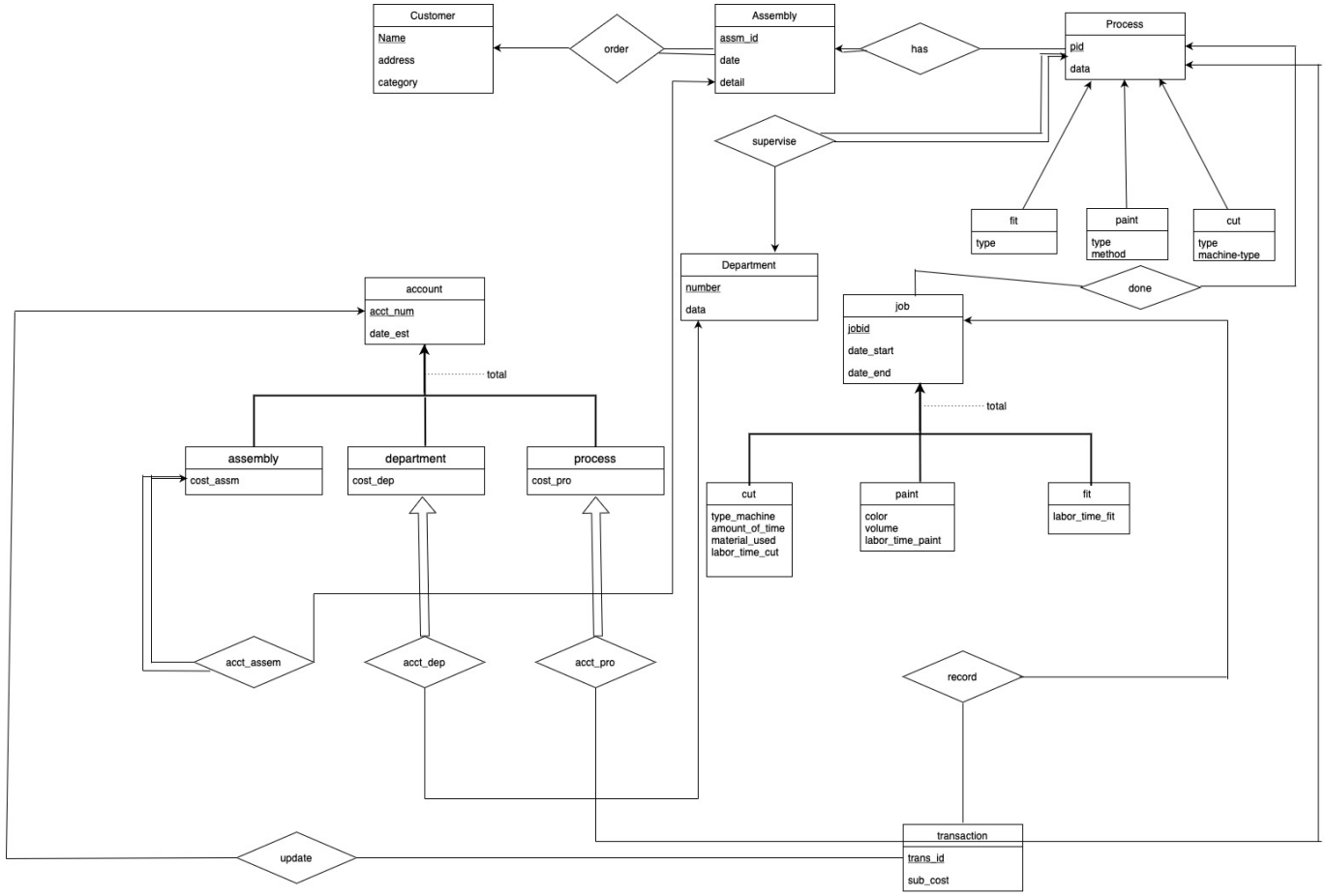
NAME: Oyindamola Obisesan
STUDENT ID: 113511813
EMAIL: OYIN.OBISESAN@OU.EDU
COURSE: CS/DSA 4513 – DATABASE MANAGEMENT
SECTION: 995
SECTION: ONLINE
SEMESTER: FALL 2021
INSTRUCTOR: DR. LE GRUENWALD
INDIVIDUAL PROJECT
SCORE:

Table of Contents

Task 1

1.1. ER Diagram



1.2. Relational Database

Customer (cust_Name, address, category)
Assembly (assm_id, pid, cust_name, date_assm, detail)
Order (cust_name, assm_id)
Has (assm_id, pid)
Processes (pid, dept_num, data)
processes_fit (pid, dept_num, type)
processes_paint (pid, dept_num, type, method)
processes_cut (pid, dept_num, type, machine_type)
job_cut (job_id, type_of_machine, amount_of_time, materials_used, labor_time_cut, date_start, date_end)
job_paint (job_id, color, volume, labor_time_paint, date_start, date_end)
job_fit (job_id, labor_time_fit, date_start, date_end)
department (dep_num, data)
acct_assem (acct_num, assm_id, date_est, cost_assm)
acct_dep (acct_num, dep_num, date_est, cost_dep)
acct_pro (acct_num, pid, date_est, cost_pro)
transactions (trans_id, sub_cost)
update (acct_num, trans_id)
record (trans_id, job_id)
done (job_id, pid, dep_num)

Task 2 - Data Dictionary

P/F	Field Name	Types	Size
Customer			
	Cust_name	Varchar	30
	Address	Varchar	40
	category	Numeric/integer	2
Assembly			
	Assm_ID	Varchar	12
	Date_assem	Date	3
	details	bit	64
Process	Pid	Varchar	12
	Process_data	Varchar	30
	Dept_num	integer	2,2
Process_fit	Pid	Varchar	12
	Fit_type	Varchar	12
	Dept_num	int	3
Process_ Paint	pid	Varchar	12
	paint_Type	Varchar	12
	paint_method	Varchar	12
	dept_num	Int	2
Process_cut	pid	Varchar	12
	cut_Type	Varchar	12

	Machine-type	Varchar	12
	Dept_num	Int	2
Job_ Cut	Job_id	Integer	2,2
	type_machine	Varchar	15
	amount_of_time	Numeric	3
	material_used	Varchar	30
	labor_time_cut	Numeric	3
	date_start	Date	3
	date_end	Date	3
job _ Paint	Job_id	Integer	5
	Color	Varchar	12
	Volume	Numeric	3
	Labor_time_paint	Numeric	3
	Date_start	date	3
	Date_end	date	3
Job_fit	Jobid	Integer	20
	labor_time_fit	Numeric	3
	date_start	Date	3
	date_end	Date	3
department	Dept_num	Integer	2
	Dep_data	varchar	30
Acct_assem	Acct_num	Integer	10
	Cost_assm	Numeric	10

	Date_est	Date	3
	Assm_id	Varchar	12
Department_acct	Acct_num	Int	10
	Dep_num	Int	2
	Cost_dep	Numeric	10
	Date_est	Date	3
Acct_pro	Acct_num	int	10
	pid	varchar	12
	Cost_pro	numeric	10
	Date_est	date	3
transaction	Trans_id	Varchar	15
	Sub_cost	Numeric	10
has	Assm_id	Varchar	12
	pid	varchar	12
order	Cust_name	Varchar	12
	Assm_id	varchar	12
update	Acct_num	Int	10
	Trans_id	varchar	15
record	Trans_id	Varchar	15
	Job_id	int	10
done	Job_id	Int	10
	Dep_num	Int	2
	pid	varchar	12

Task 3

3.1. Discussion of storage structures

Table Name	Query # and Type	Search Key	Query Frequency	Selected File Organization	Justification
Customer	1, insert 13, range search	category	30/day 100/day	Extendable Hashing on category	The hash function can be the category number since they are only 10. They can be searched in each of the bucket.
department	2, insert 12, random search	Dep_num	Infrequent 20/day	Index sequential file on dep_num	Since this is not done very frequently, the file can be arranged in sequential order with the dep_num and searched.
processes	3, insert		infrequent	Primary indexing on pid	Since this is not done very frequently, the file can be arranged in sequential order with the pid and searched.
Processes_cut	3, insert		infrequent	Primary indexing on pid	Since this is not done very frequently, the file can be arranged in sequential order with the pid and searched.
Processes_fit	3, insert		infrequent	Primary indexing on pid	Since this is not done very frequently, the file can be arranged in sequential order with the pid and searched.
Processes_paint	3, insert		infrequent	Primary indexing on pid	Since this is not done very frequently, the file can be arranged in sequential order

					with the pid and searched.
Acct_assem	5, insert 8, random search 9, random search	 Acct_num Assm_id	10/day 50/day 200/day	Index sequential file primary index on assm_id secondary index on acct_num	The primary search is done on assm_id because it occurs more frequently
acct_dep	(or 5, insert) 8, random search	 Acct_num	10/day 50/day	Index sequential file primary index on assm_id secondary index on acct_num	The primary search is done on assm_id because it occurs more frequently
acct_pro	(or 5, insert) 8, random search	 Acct_num	10/day 50/day	Index sequential file primary index on assm_id secondary index on acct_num	The primary search is done on assm_id because it occurs more frequently
assemblies	4, insert 11, random search	 Assm_id	 100/day	Extendable hashing on assm_id	This occurs frequently and needs to be accessed fast.
orders	4, insert		40/day	heap	
has	4, insert 6, insert		40/day 50/day	Index sequential	
done	6, insert 10, random search	 Dep_num	50/day 20/day	Index sequential on dept_num	The search is infrequent

job_cut	6, insert 7, update 10, random search 12, random search 14, deletion	 Job_id Date_end Date_end Job_id	50/day 50/day 20/day 20/day 1/month	Index sequential file Primary index on job_id And secondary index on date_end	The search is done on job_id because it occurs more frequently than date_end
transactions	8, insert		40/day	Sequential primary index on trans_id	
Job_fit	(or 6, insert) 7, update 10, random search 12, random search	 Job_id Date_end Date_end	50/day 50/day 20/day 20/day	Index sequential file Primary index on job_id And secondary index on date_end	The search is done on job_id because it occurs more frequently than date_end
Job_paint	(or 6, insert) 7, update 10, random search 12, random search 15, update	 Job_id Date_end Date_end Jod_id	50/day 50/day 20/day 20/day 1/week	Index sequential file Primary index on job_id And secondary index on date_end	The search is done on job_id because it occurs more frequently than date_end

3.2. Discussion of storage structures (Azure SQL Database)

The index sequential search was the database selected for use on this database. The primary key creates the index automatically. A secondary key is selected based on the other attributes that the queries search.

```
create NONCLUSTERED index customer on dbo.customer(category)
create NONCLUSTERED index done on dbo.done(dep_num)
create NONCLUSTERED index job_fit on dbo.job_fit(date_end)
create NONCLUSTERED index job_cut on dbo.job_cut(date_end)
create NONCLUSTERED index job_paint on dbo.job_paint(date_end)
```

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

Table Creation

```
drop table if exists records
drop table if exists orders
drop table if exists processes
drop table if exists processes_fit
drop table if exists processes_paint
drop table if exists processes_cut
drop table if exists has
drop table if exists job_cut
drop table if exists job_paint
drop table if exists job_fit
drop table if exists acct_assem
drop table if exists acct_dep
drop table if exists acct_pro
drop table if exists updates
drop table if exists records
drop table if exists transactions
drop table if exists done
drop table if exists department
drop table if exists assemblies
drop table if exists customer


create table customer(
    cust_name varchar(30) primary key,
    addr varchar(30),
    category integer
    );

create table assemblies(
    assm_id varchar(12),
    pid varchar(12),
    cust_name varchar(30) foreign key references customer (cust_name),
    dat_assm date,
    detail varchar(40),
    primary key (assm_id, pid, cust_name)
    );

create table orders(
    --assm_id varchar(12) foreign key references assemblies (assm_id),
    assm_id varchar(12),
    cust_name varchar(30) foreign key references customer (cust_name),
    primary key (cust_name, assm_id)
    );
```

```
create table has(
    --assm_id varchar(12) foreign key references assemblies (assm_id),
    assm_id varchar(12),
    pid varchar(12),
    --job_id integer,
    primary key (assm_id, pid)
);

create table department(
    dept_num integer primary key,
    dep_data varchar(30)
    );

create table processes(
    pid varchar(12),
    process_data varchar(30),
    --assm_id varchar(12) foreign key references assemblies(assm_id),
    --assm_id varchar(12),
    dept_num integer foreign key references department(dept_num),
    primary key (pid, dept_num)
    );

create table processes_fit(
    pid varchar(12),
    fit_type varchar(12),
    --assm_id varchar(12) foreign key references assemblies(assm_id),
    --assm_id varchar(12),
    dept_num integer foreign key references department(dept_num),
    primary key (pid, dept_num)
    );

create table processes_paint(
    pid varchar(12),
    paint_type varchar(12),
    paint_method varchar(20),
    dept_num integer foreign key references department(dept_num),
    primary key (pid, dept_num)
    );

create table processes_cut(
    pid varchar(12),
    cut_type varchar(12),
    machine_type varchar (12),
    dept_num integer foreign key references department(dept_num),
    primary key (pid, dept_num)
    );

create table job_cut(
```

```
    job_id integer primary key,
    type_of_mac varchar(15),
    amt_time numeric,
    mat_used varchar(30),
    labor_time_cut numeric,
    date_start date,
    date_end date
);

create table job_paint(
    job_id integer primary key,
    color varchar(12),
    volume numeric,
    labor_time_paint numeric,
    date_start date,
    date_end date
);

create table job_fit(
    job_id integer primary key,
    labor_time_fit numeric,
    date_start date,
    date_end date
);

create table acct_assem(
    acct_num integer,
    assm_id varchar(12),
    date_est date,
    cost_assm numeric,
    primary key(acct_num, assm_id)
    );

create table acct_dep(
    acct_num integer,
    dept_num integer foreign key references department (dept_num),
    date_est date,
    cost_dept numeric
    primary key (acct_num, dept_num)
    );

create table acct_pro(
    pid varchar(12),
    acct_num integer,
    date_est date,
    cost_pro numeric,
    primary key (acct_num, pid)
    );
```

```sql
create table transactions(
    trans_id varchar(15) primary key,
    sub_cost numeric
    );

create table updates(
    acct_num integer,
    trans_id varchar(15) foreign key references transactions(trans_id),
    primary key (acct_num, trans_id)
);

create table records(
    trans_id varchar(15) foreign key references transactions(trans_id),
    job_id integer,
    primary key (trans_id, job_id)
);

create table done(
    dep_num integer foreign key references department (dept_num),
    pid varchar(12),
    job_id integer
    primary key (job_id,pid, dep_num)
);
```

Task 5

5.1. SQL statements and T-SQL implementing queries 1-15

5.1.1. SQL

```sql
--1
insert into customer
values
('Oyindamola', '2900 Beaumont', 2),


--2
insert into department
values
(7, 'this is Mewbourne');


--3
insert into department
values
(9, 'DSA');
```

```sql
insert into processes
values
('40', 9, 'process?')

update processes_fit
set fit_type = 'straight'
where pid = '40'


--4
insert into assemblies
values
('78', '67', 'Kome', '23-05-2021', 'assembly')

insert into orders
values
('Kome', '78')

insert into has
values
('78', '67')


--5
insert into acct_assem
values
(2333455, '6', '2021-04-04', 567)

--6

insert into job_fit(job_id, date_start)
values
(56, '2021-04-05');

insert into done
values
(34, '45', 56)

insert has
values
('67', '56')

--7
update job_fit
set
    date_end = '2021-04-06',
    labor_time_fit = 5
```

```sql
where job_id = 56



--8
insert into transactions
values
(345, 45)


--9
select sum (cost_assm)
from acct_assem
where assm_id = 34

--10
select sum (total_labor) from (
    select sum(labor_time_cut) as total_labor from job_cut where job_id in (select
job_id from done where dept_num = 2) and date_end = '2021-04-03' union all
    select sum(labor_time_cut) as total_labor from job_fit where job_id in (select
job_id from done where dept_num = 2) and date_end = '2021-04-03' union all
    select sum(labor_time_cut) as total_labor from job_paint where job_id in (select
job_id from done where dept_num = 2) and date_end = '2021-04-03'
) total_labor_time
--x group by dep_num

--11
select done.job_id, dept_num
from assemblies


inner join
done on assemblies.pid = done.pid
right join
    (select job_id, date_start, date_end
    from job_cut
    UNION
    select job_id, date_start, date_end
    from job_paint
    UNION
    select job_id, date_start, date_end
    from job_fit) aa
    on done.job_id = aa.job_id
    where assm_id = 45
order by date_start

--12a
select *
```

```
from assemblies
inner join
done on assemblies.pid = done.pid
right join
job_cut on done.job_id = job_cut.job_id


--12b
select *
from assemblies
inner join
done on assemblies.pid = done.pid
right join
job_fit on done.job_id = job_fit.job_id

--12c
select *
from assemblies
inner join
done on assemblies.pid = done.pid
right join
job_paint on done.job_id = job_paint.job_id

--13
select cust_name
from customer
where category between 4 and 5
order by cust_name asc


-- 14
delete
from job_cut
where job_id between 40 and 60

--15
update job_paint
set color = 'red'
where job_id = 56
```

5.1.2. T-SQL

```
drop procedure query1
go
create PROCEDURE query1
```

```
    @cust_name varchar(30),
    @addr varchar(30),
    @category integer

    as
    BEGIN
        insert into customer (cust_name, addr, category)
        values (@cust_name, @addr, @category)

    end


drop procedure query2

go
create PROCEDURE query2
    @dept_num integer,
    @dep_data varchar(30)

as
BEGIN
    insert into department (dept_num, dep_data)
    values (@dept_num, @dep_data)

end

drop procedure query3
go
create procedure query3
    @pid varchar(12),
    @process_data varchar(30),
    @dept_num integer,
    @type_process varchar(30),
    @cut_type varchar(30),
    @machine_type VARCHAR(30),
    @fit_type VARCHAR(30),
    @paint_type varchar(30),
    @paint_method varchar(30)
as
BEGIN
    --insert department (dept_num)
    --values (@dept_num)

    INSERT processes (pid, process_data, dept_num)
    values (@pid, @process_data, @dept_num)

if @type_process = 'cut'
    insert into processes_cut (pid, cut_type, machine_type, dept_num)
```

```
    values (@pid, @cut_type, @machine_type, @dept_num)
if @type_process = 'fit'
    insert into processes_fit (pid, fit_type, dept_num)
    values (@pid, @fit_type, @dept_num)
if @type_process = 'paint'
    insert into processes_paint (pid, paint_type, paint_method, dept_num)
    values (@pid, @paint_type, @paint_method, @dept_num)
end

drop PROCEDURE query4

go
CREATE procedure query4
    @assm_id varchar(12),
    @pid integer,
    @cust_name varchar(30),
    @dat_assm date,
    @detail varchar(30),
    @dep_num INTEGER,
    @job_id integer

as
begin

    --insert customer (cust_name)
    --values (@cust_name)

    insert into assemblies (assm_id, pid, cust_name, dat_assm, detail)
    values (@assm_id, @pid, @cust_name, @dat_assm, @detail)

    insert orders (assm_id, cust_name)
    values (@assm_id, @cust_name)

    insert into has (assm_id, pid)
    values (@assm_id, @pid)

end

drop procedure query5

go
create procedure query5
    @acct_num integer,
    @pid varchar(12),
    @assm_id varchar(12),
    @date_est date,
    @dept_num integer,
    @type_acct varchar(12),
```

```
    @cost_assm numeric,
    @cost_dept numeric,
    @cost_pro numeric
as
begin
if (@type_acct = 'assembly')
    begin
        insert into acct_assem (acct_num, assm_id, date_est, cost_assm)
        values (@acct_num, @assm_id, @date_est, @cost_assm)
    end
if (@type_acct = 'department')
    begin
        insert into acct_dep (acct_num, dept_num, date_est, cost_dept)
        values (@acct_num, @dept_num, @date_est, @cost_dept)
    end

if (@type_acct = 'process')
    begin
        insert into acct_pro (pid, acct_num, date_est, cost_pro)
        values (@pid, @acct_num, @date_est, @cost_pro)
    end

end

drop procedure query6


GO
create procedure query6
    @job_id INTEGER,
    @date_start date,
    @dept_num integer,
    @pid varchar(12),
    @assm_id varchar(12),
    @type_job varchar (12)

as
begin
    --INSERT into job_fit (job_id, date_start)
    --values (@job_id, @date_start)

    --insert into department(dept_num)
    --values (@dept_num)
    insert into done (dep_num, pid, job_id)
    values (@dept_num, @pid, @job_id)

    insert into has (assm_id, pid)
    values (@assm_id, @pid)
```

```
if @type_job = 'fit'
    INSERT into job_fit (job_id, date_start)
    values (@job_id, @date_start)
if @type_job = 'cut'
    insert into job_cut (job_id, date_start)
    values (@job_id, @date_start)
if @type_job = 'paint'
    insert into job_paint (job_id, date_start)
    values (@job_id, @date_start)




end

drop procedure query7

go
create procedure query7
    @date_end date,
    @labor_time_cut numeric,
    @labor_time_fit numeric,
    @labor_time_paint numeric,
    @job_id integer,
    @type_job varchar(12),
    @type_of_mac varchar(15),
    @amt_time numeric,
    @mat_used varchar(30),
    @volume numeric,
    @color varchar(12)

as

BEGIN

begin
    update job_cut
    set
        date_end = @date_end,
        labor_time_cut = @labor_time_cut,
        type_of_mac = @type_of_mac,
        amt_time = @amt_time,
        mat_used = @mat_used
    where job_id = @job_id and @type_job = 'cut'

end
```

```
begin
    update job_fit
    set
        date_end = @date_end,
        labor_time_fit = @labor_time_fit
    where job_id = @job_id and @type_job = 'fit'
end

begin
    update job_paint
    set
        date_end = @date_end,
        labor_time_paint = @labor_time_paint,
        volume = @volume,
        color = @color
    where job_id = @job_id and @type_job = 'paint'

end

END

drop procedure query8

go
create procedure query8
    @trans_id varchar(15),
    @sub_cost numeric,
    @acct_type varchar(15),
    @assm_id varchar(15),
    @acct_num integer,
    @date_est date

as
begin
    insert into transactions (trans_id, sub_cost)
    values (@trans_id, @sub_cost)
end

begin
update acct_assem
set
    cost_assm = cost_assm + @sub_cost,
    date_est = @date_est
where acct_num = @acct_num and @acct_type = 'assembly'
end

begin
update acct_dep
```

```
set
    cost_dept = cost_dept + @sub_cost,
    date_est = @date_est
where acct_num = @acct_num and @acct_type = 'department'
end

begin
update acct_pro
set
    cost_pro = cost_pro + @sub_cost,
    date_est = @date_est
where acct_num = @acct_num and @acct_type = 'assembly'
end



drop PROCEDURE query9
go
create procedure query9
    @assm_id varchar (12)
as
begin
    select sum (cost_assm) as total_cost
    from acct_assem
    where assm_id = @assm_id
end

drop procedure query10
GO
create procedure query10
    @dept_num integer,
    @date_end date
as
begin
select sum (total_labor) from (
    select sum(labor_time_cut) as total_labor from job_cut where job_id in (select
job_id from done where dep_num = @dept_num) and date_end = @date_end union all
    select sum(labor_time_fit) as total_labor from job_fit where job_id in (select
job_id from done where dep_num = @dept_num) and date_end = @date_end union all
    select sum(labor_time_paint) as total_labor from job_paint where job_id in (select
job_id from done where dep_num = @dept_num) and date_end = @date_end
) total_labor_time

end


drop procedure query11
```

```
GO
create procedure query11
    @assm_id varchar(12)
as
begin
    select done.job_id, dep_num
    from assemblies
    inner join
    done on assemblies.pid = done.pid
    right join
        (select job_id, date_start, date_end
        from job_cut
        UNION
        select job_id, date_start, date_end
        from job_paint
        UNION
        select job_id, date_start, date_end
        from job_fit) aa
        on done.job_id = aa.job_id
        where assm_id = @assm_id
    order by date_start

end

drop procedure query12

GO
create PROCEDURE query12
    @job_type varchar(12),
    @date_end date,
    @dept_num integer

as
begin
if @job_type = 'cut'
    select *
    from assemblies
    inner join
    done on assemblies.pid = done.pid
    right join
    job_cut on done.job_id = job_cut.job_id
    where date_end = @date_end and dep_num = @dept_num

if @job_type = 'fit'
    select *
    from assemblies
    inner join
    done on assemblies.pid = done.pid
```

```
    right join
    job_fit on done.job_id = job_fit.job_id
    where date_end = @date_end and dep_num = @dept_num

if @job_type = 'paint'
    select *
    from assemblies
    inner join
    done on assemblies.pid = done.pid
    right join
    job_paint on done.job_id = job_paint.job_id
    where date_end = @date_end and dep_num = @dept_num
end



go
create procedure query13
    @cat_1 integer,
    @cat_2 integer

as
BEGIN
    select cust_name
    from customer
    where category between @cat_1 and @cat_2
    order by cust_name asc
end


go
create procedure query14
    @job_id_1 integer,
    @job_id_2 INTEGER
as
begin
    delete
    from job_cut
    where job_id between @job_id_1 and @job_id_2
end


go
create procedure query15
    @job_id integer,
    @color varchar(15)
as
begin
```

```
    update job_paint
    set color = @color
    where job_id = @job_id

end

drop procedure query16

go
create procedure query16
    @cust_name varchar (30),
    @addr varchar(30),
    @category integer
as
begin
    insert into customer
    values(@cust_name, @addr, @category)
end

drop procedure query17

go
create procedure query17
    @cat_1 integer,
    @cat_2 integer
as
begin
    select cust_name
    from customer
    where category between @cat_1 and @cat_2
order by cust_name asc
end
```

5.2. Java Source Program and Screenshots

5.2.1. Java Source Program

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

```
import java.io.FileNotFoundException;

public class Project1 {

    // Database credentials
    final static String HOSTNAME = "obis000-sql-server.database.windows.net";
    final static String DBNAME = "cs-dsa-4513-sql-db";
    final static String USERNAME = "obis0000";
    final static String PASSWORD = "Anec3030";

 // Database connection string
    final static String URL =
String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;password=%s;encrypt=true;trust
ServerCertificate=false;hostNameInCertificate=*.database.windows.net;loginTimeout=30;",
            HOSTNAME, DBNAME, USERNAME, PASSWORD);

    // Query templates
    final static String QUERY_TEMPLATE_1 = "EXEC query1 @cust_name = ?, @addr = ?,
@category = ?;";
    final static String QUERY_TEMPLATE_2 = "EXEC query2 @dept_num = ?, @dep_data =
?;";
    final static String QUERY_TEMPLATE_3 = "EXEC query3 @pid = ?, @process_data = ?,
@dept_num = ?, @type_process = ?, @cut_type = ?, @machine_type = ?, @fit_type = ?,
@paint_type = ?, @paint_method = ?;";
    //final static String QUERY_TEMPLATE_3 = "EXEC query3 @pid = ?, @process_data = ?,
@dept_num = ?, @type_process = ?, @cut_type = ?, machine_type = ?, @fit_type = ?,
@paint_type = ?, @paint_method = ?;";
    final static String QUERY_TEMPLATE_4 = "EXEC query4 @assm_id = ?, @pid = ?,
@cust_name = ?, @dat_assm = ?, @detail = ?, @job_id = ?, @dep_num = ?;";
    final static String QUERY_TEMPLATE_5 = "EXEC query5 @acct_num = ?, @pid = ?,
@assm_id = ?, @date_est = ?, @dept_num = ?, @type_acct = ?, @cost_assm = ?, @cost_dept =
?, @cost_pro = ?;";
    final static String QUERY_TEMPLATE_6 = "EXEC query6 @job_id = ?, @date_start = ?,
@dept_num = ?, @pid = ?, @assm_id = ?, @type_job = ?;";
    final static String QUERY_TEMPLATE_7 = "EXEC query7 @date_end = ?,
@labor_time_cut = ?, @labor_time_fit = ?, @labor_time_paint = ?, @job_id = ?, @type_job =
?, @type_of_mac = ?, @amt_time = ?, @mat_used = ?, @volume = ?, @color = ?;";
    final static String QUERY_TEMPLATE_8 = "EXEC query8 @trans_id = ?, @sub_cost = ?,
@acct_type = ?, @assm_id = ?, @acct_num = ?, @date_est = ?;";
    final static String QUERY_TEMPLATE_9 = "EXEC query9 @assm_id = ?;";
    final static String QUERY_TEMPLATE_10 = "EXEC query10 @dept_num = ?, @date_end =
?;";
    final static String QUERY_TEMPLATE_11 = "EXEC query11 @assm_id = ?;";
    final static String QUERY_TEMPLATE_12 = "EXEC query12 @job_type = ?, @date_end =
?, @dept_num = ?;";
    final static String QUERY_TEMPLATE_13 = "EXEC query13 @cat_1 = ?, @cat_2 = ?;";
```

```
    final static String QUERY_TEMPLATE_14 = "EXEC query14 @job_id_1 = ?, @job_id_2 =
?;";
    final static String QUERY_TEMPLATE_15 = "EXEC query15 @job_id = ?, @color = ?;";
    final static String QUERY_TEMPLATE_16 = "EXEC query15 @cust_name = ?, @addr = ?,
@category = ?;";
    final static String QUERY_TEMPLATE_17 = "EXEC query15 @cat_1 = ?, @cat_2 = ?;";

    // User input prompt//
    final static String PROMPT =
            "\nPlease select one of the options below: \n" +
            "1) Insert new customer; \n" +
            "2) Insert new department; \n" +
            "3) Insert new process; \n" +
            "4) Insert new assembly; \n" +
            "5) Insert new account; \n" +
            "6) Insert new job; \n" +
            "7) Insert new job_end; \n" +
            "8) Insert update acct; \n" +
            "9) retrieve total labor time; \n" +
            "10)retreive total_labor_time; \n" +
            "11)retreive assm_id; \n" +
            "12)retrieve job; \n" +
            "13)retrieve category range; \n" +
            "14)delete job; \n" +
            "15)update paint; \n" +
            "16)import; \n" +
            "17)export; \n" +
            "18) Exit!";

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

        System.out.println("Welcome to the sample application!");

        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("3")) { // As user for options until option 3 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 a new CUSTOMER option
                // Collect the new customer data from the user
                System.out.println("Please enter Customer name:");
                sc.nextLine();
                final String cust_name = sc.nextLine(); // Read in the user input of customer name
```

```
            System.out.println("Please enter address:");
            // Preceding nextInt, nextFloar, etc. do not consume new line characters from the user
input.
            // We call nextLine to consume that newline character, so that subsequent nextLine
doesn't return nothing.
            //sc.nextLine();
            final String addr = sc.nextLine(); // Read in user input of address (white-spaces
allowed).


            System.out.println("Please enter category");
            //sc.nextLine();
            final int category = sc.nextInt(); // Read in user input of category



            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(QUERY_TEMPLATE_1)) {
                  // Populate the query template with the data collected from the user
                  statement.setString(1, cust_name);
                  statement.setString(2, addr);
                  //statement.setInt(3, y_of_e);
                  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));
               }
            }

            break;

         case "2": // Insert a new student option
            // Collect the new department data from the user
            System.out.println("Please enter integer  department:");
            final int dept_num = sc.nextInt(); // Read in the user input of dept_num

            System.out.println("Please enter department details:");
            // Preceding nextInt, nextFloar, etc. do not consume new line characters from the user
input.
            // We call nextLine to consume that newline character, so that subsequent nextLine
doesn't return nothing.
```

```
            sc.nextLine();
            final String dep_data = sc.nextLine(); // Read in user input of student department
number



            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(QUERY_TEMPLATE_2)) {
                  // Populate the query template with the data collected from the user
                  statement.setInt(1, dept_num);
                  statement.setString(2, dep_data);



                  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));
               }
            }

            break;

         case "3": // Insert a new process option
            // Collect the new process data from the user
            System.out.println("Please enter process id:");
            sc.nextLine();
            final String pid3 = sc.nextLine(); // Read in the user input of process id

            System.out.println("Please enter process data:");
            //sc.nextLine();
            final String process_data = sc.nextLine(); // Read in the user input of process data

            System.out.println("Please enter department num:");
            final int dept_num3 = sc.nextInt(); // Read in the user input of department number

            System.out.println("Please enter type_process:");
            sc.nextLine();
            final String type_process = sc.nextLine(); // Read in the user input of process type

            System.out.println("Please enter cut type:");
            //sc.nextLine();
```

```
            final String cut_type = sc.nextLine(); // Read in the user input of cut type

            System.out.println("Please enter machine type:");
            //sc.nextLine();
            final String machine_type = sc.nextLine(); // Read in the user input of machine type

            System.out.println("Please enter fit type:");
            //sc.nextLine();
            final String fit_type = sc.nextLine(); // Read in the user input of fit type

            System.out.println("Please enter paint type:");
            //sc.nextLine();
            final String paint_type = sc.nextLine(); // Read in the user input of paint type

             System.out.println("Please enter paint method:");
             // Preceding nextInt, nextFloar, etc. do not consume new line characters from the user
input.
             // We call nextLine to consume that newline character, so that subsequent nextLine
doesn't return nothing.
             //sc.nextLine();
             final String paint_method = sc.nextLine(); // Read in user input of student First Name
(white-spaces allowed).

             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(QUERY_TEMPLATE_3)) {
                    // Populate the query template with the data collected from the user

                    statement.setString(1, pid3);
                    statement.setString(2, process_data);
                    statement.setInt(3, dept_num3);
                    statement.setString(4, type_process);
                    statement.setString(5, cut_type);
                    statement.setString(6, machine_type);
                    statement.setString(7, fit_type);
                    statement.setString(8, paint_type);
                    statement.setString(9, paint_method);


                    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));
```

```
            }
        }

        break;


    case "4": // Insert a new assembly option
        // Collect the new student data from the user
            System.out.println("Please enter assembly ID:");
            sc.nextLine();
        final String assm_id = sc.nextLine();

        System.out.println("Please enter process id:");
        final int pid = sc.nextInt();

        System.out.println("Please enter customer name:");
            sc.nextLine();
        final String cust_name1 = sc.nextLine();

        System.out.println("Please enter assembly date:");
            //sc.nextLine();
        final String dat_assm = sc.nextLine();

        System.out.println("Please enter assembly detail:");
            //sc.nextLine();
        final String detail = sc.nextLine();

            System.out.println("Please enter integer  department:");
        final int dep_num = sc.nextInt();

        System.out.println("Please enter JOB ID:");
        // Preceding nextInt, nextFloar, etc. do not consume new line characters from the user
input.
        // We call nextLine to consume that newline character, so that subsequent nextLine
doesn't return nothing.
        //sc.nextLine();
        final int job_id = 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(QUERY_TEMPLATE_4)) {
                // Populate the query template with the data collected from the user
```

```
            statement.setString(1, assm_id);
            statement.setInt(2, pid);
            statement.setString(3,  cust_name1);
            statement.setString(4, dat_assm);
            statement.setString(5,  detail);
            statement.setInt(6, dep_num);
            statement.setInt(7, job_id);


            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));
        }
    }

    break;



case "5": // Insert a new student option
    // Collect the new student data from the user
            System.out.println("Please enter account number:");
            //sc.nextLine();
    final int acct_num5 = sc.nextInt();

    System.out.println("Please enter process id:");
    sc.nextLine();
    final String pid5 = sc.nextLine();

    System.out.println("Please enter assembly id:");
    //sc.nextLine();
    final String assm_id5 = sc.nextLine();

    System.out.println("Please enter date est:");
            //sc.nextLine();
    final String date_est = sc.nextLine();

    System.out.println("Please enter department number:");
            //sc.nextLine();
    final int dept_num5 = sc.nextInt();

    System.out.println("Please enter type:");
            sc.nextLine();
    final String type_acct = sc.nextLine();
```

```
            System.out.println("Please enter cost of assem:");
            //sc.nextLine();
            final float cost_assm = sc.nextFloat();

            System.out.println("Please enter cost of dept:");
            //sc.nextLine();
            final float cost_dept = sc.nextFloat();

            System.out.println("Please enter cost of process:");
            //sc.nextLine();
            final float cost_pro = sc.nextFloat();


            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(QUERY_TEMPLATE_5)) {
                  // Populate the query template with the data collected from the user
                  statement.setInt(1,acct_num5);
                  statement.setString(2,  pid5);
                  statement.setString(3, assm_id5);
                  statement.setString(4, date_est);
                  statement.setInt(5, dept_num5);
                  statement.setString(6, type_acct);
                  statement.setFloat(7,  cost_assm);
                  statement.setFloat(8,  cost_dept);
                  statement.setFloat(9, cost_pro);


                  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));
               }
            }

            break;

         case "6": // Insert a new job option
            // Collect the new job data from the user

                  System.out.println("Please enter job id:");
            final int job_id6 = sc.nextInt();
```

```
            System.out.println("Please enter date start:");
            sc.nextLine();
            final String date_start = sc.nextLine();

            System.out.println("Please enter integer  department:");
            final int dept_num6 = sc.nextInt();



                System.out.println("Please enter process id:");
            final int pid6 = sc.nextInt();


                System.out.println("Please enter assembly ID:");
                sc.nextLine();
            final String assm_id6 = sc.nextLine();

            System.out.println("Please enter job_type:");
                sc.nextLine();
            final String type_job6 = 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(QUERY_TEMPLATE_6)) {
                  // Populate the query template with the data collected from the user
                  statement.setInt(1, job_id6);
                  statement.setString(2, date_start);
                  statement.setInt(3, dept_num6);
                  statement.setInt(4, pid6);
                  statement.setString(5, assm_id6);
                  statement.setString(6, type_job6);

                  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));
               }
            }

            break;
```

```
case "7": // Insert a new job option
    // Collect the new job data from the user

            System.out.println("Please enter date end:");
     sc.nextLine();
     final String date_end = sc.nextLine();

           System.out.println("Please enter  labor time cut:");
    final float labor_time_cut = sc.nextFloat();

    System.out.println("Please enter  labor time fit:");
    final float labor_time_fit = sc.nextFloat();

    System.out.println("Please enter  labor time paint:");
    final float labor_time_paint = sc.nextFloat();


    System.out.println("Please enter job id:");
    final int job_id7 = sc.nextInt();


           System.out.println("Please enter job type:");
           sc.nextLine();
    final String type_job = sc.nextLine();


    System.out.println("Please enter type_of_mac:");
           //sc.nextLine();
    final String type_mac = sc.nextLine();

    System.out.println("Please enter amt time:");
    final float amt_time = sc.nextFloat();


    System.out.println("Please enter mat used:");
           sc.nextLine();
    final String mat_used = sc.nextLine();


    System.out.println("Please enter volume");
    final float volume = sc.nextFloat();

    System.out.println("Please enter color:");
           sc.nextLine();
    final String color = 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(QUERY_TEMPLATE_7)) {
                  // Populate the query template with the data collected from the user
                  statement.setString(1, date_end);
                  statement.setFloat(2, labor_time_cut);
                  statement.setFloat(3, labor_time_fit);
                  statement.setFloat(4, labor_time_paint);

                  statement.setInt(5, job_id7);

                  statement.setString(6, type_job);
                  statement.setString(7, type_mac);
                  statement.setFloat(8, amt_time);
                  statement.setString(9, mat_used);
                  statement.setFloat(10, volume);
                  statement.setString(11, color);

                  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));
               }
            }

            break;

         case "8": // Insert a new transaction option
            // Collect the new transaction data from the user

                  System.out.println("Please enter trans id:");
                  sc.nextLine();
            final String trans_id = sc.nextLine();

            System.out.println("Please enter subcost:");
                  //sc.nextLine();
            final float sub_cost = sc.nextFloat();

            System.out.println("Please enter acct type:");
                  sc.nextLine();
            final String acct_type = sc.nextLine();
```

```
            System.out.println("Please enter assembly ID:");
                  //sc.nextLine();
            final String assm_id8 = sc.nextLine();

            System.out.println("Please enter acct num:");
            //sc.nextLine();
            final int acct_num = sc.nextInt();

            System.out.println("Please enter date est:");
            sc.nextLine();
            final String date_est8 = 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(QUERY_TEMPLATE_8)) {
                  // Populate the query template with the data collected from the user
                  statement.setString(1, trans_id);
                  statement.setFloat(2, sub_cost);
                  statement.setString(3, acct_type);
                  statement.setString(4, assm_id8);
                  statement.setInt(5, acct_num);
                  statement.setString(6, date_est8);



                  //statement.setFloat(3, age_);
                  //statement.setInt(4,did);


                  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));
               }
            }

            break;


         case "9":
```

```
            System.out.println("Please enter assembly ID:");
                  sc.nextLine();
            final String assm_id9 = 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(QUERY_TEMPLATE_9)) {
                  // Populate the query template with the data collected from the user

                  statement.setString(1, assm_id9);

                  try(
                        final ResultSet resultSet = statement.executeQuery()){

                        while (resultSet.next()) {
                        System.out.println(String.format("\ttotal_cost: %s",
                           resultSet.getString(1)));
                        System.out.println("\n");

                        }
                           }
               }
            }

            break;


         case "10":



            System.out.println("Please department num:");

            final int dept_num10 = sc.nextInt();
            //final int total_labor_time = sc.nextInt();
            System.out.println("Please date completed:");

            final String date_end10 = sc.nextLine();
            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(QUERY_TEMPLATE_10)) {
                  // Populate the query template with the data collected from the user

                  statement.setInt(1, dept_num10);
                  statement.setString(2, date_end10);

                  try(
                        final ResultSet resultSet = statement.executeQuery()){

                        while (resultSet.next()) {
                        System.out.println(String.format("\ttotal_labor_time: %s",
                           resultSet.getString(1)));
                        System.out.println("\n");

                        }
                           }
               }
            }

            break;

         case "11":



            System.out.println("Please enter assembly ID:");
                  sc.nextLine();
            final String assm_id11 = sc.nextLine();
            //final int total_labor_time = sc.nextInt();
            //final String date_end10 = 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(QUERY_TEMPLATE_11)) {
                  // Populate the query template with the data collected from the user

                  statement.setString(1, assm_id11);
                  //statement.setString(2, date_end10);

                  try(
```

```
                    final ResultSet resultSet = statement.executeQuery()){

                    while (resultSet.next()) {
                    System.out.println(String.format("%s | %s",
                       resultSet.getString(1), resultSet.getString(2)));
                    System.out.println("\n");

                    }
                       }
            }
         }

         break;

      case "12":



         System.out.println("Please enter job type:");
               sc.nextLine();
         final String job_type = sc.nextLine();

         System.out.println("Please enter date ended:");
               //sc.nextLine();
         final String date_end12 = sc.nextLine();

         System.out.println("Please enter department number:");
               //sc.nextLine();
         final int dept_num12 = sc.nextInt();

         //final int total_labor_time = sc.nextInt();
         //final String date_end10 = 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(QUERY_TEMPLATE_12)) {
               // Populate the query template with the data collected from the user

               statement.setString(1, job_type);
               statement.setString(2, date_end12);
               statement.setInt(3, dept_num12);

               try(
```

```
                    final ResultSet resultSet = statement.executeQuery()){

                    while (resultSet.next()) {
                    System.out.println(String.format("%s | %s | %s | %s | %s | %s | %s",
                       resultSet.getString(1), resultSet.getString(2), resultSet.getString(3),
resultSet.getString(4), resultSet.getString(5), resultSet.getString(6), resultSet.getString(7)));
                    System.out.println("\n");

                    }
                       }
            }
         }

         break;

      case "13":




         System.out.println("Please enter category 1:");
               //sc.nextLine();
         final int cat_1 = sc.nextInt();

         System.out.println("Please enter category 2:");
               //sc.nextLine();
         final int cat_2 = sc.nextInt();

         //final int total_labor_time = sc.nextInt();
         //final String date_end10 = 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(QUERY_TEMPLATE_13)) {
               // Populate the query template with the data collected from the user



               statement.setInt(1, cat_1);
               statement.setInt(2, cat_2);

               try(
                     final ResultSet resultSet = statement.executeQuery()){
```

```
                    while (resultSet.next()) {
                    System.out.println(String.format("\tcust_name: %s",
                       resultSet.getString(1)));
                    System.out.println("\n");

                    }
                       }
                }
            }

            break;

        case "14":

            System.out.println("Please enter job_id 1:");
                  //sc.nextLine();
            final int job_id_1 = sc.nextInt();

            System.out.println("Please enter job_id 2:");
                  //sc.nextLine();
            final int job_id_2 = sc.nextInt();

            //final int total_labor_time = sc.nextInt();
            //final String date_end10 = 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(QUERY_TEMPLATE_14)) {
                  // Populate the query template with the data collected from the user

                  //statement.setString(1, job_type);
                  //statement.setString(2, date_end12);

                  statement.setInt(1, job_id_1);
                  statement.setInt(2, job_id_2);


                  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 updated.", rows_inserted));
```

```
            }
        }

        break;

    case "15":

        System.out.println("Please enter job_id:");
            //sc.nextLine();
        final int job_id15 = sc.nextInt();

        System.out.println("Please enter color:");
            sc.nextLine();
        final String color15 = sc.nextLine();

        //final int total_labor_time = sc.nextInt();
        //final String date_end10 = 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(QUERY_TEMPLATE_15)) {
                // Populate the query template with the data collected from the user

                //statement.setString(1, job_type);
                //statement.setString(2, date_end12);

                statement.setInt(1, job_id15);
                statement.setString(2, color15);


                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 updated.", rows_inserted));


            }
        }

        break;
```

```
        case "16": // Import: enter new customer from a data file until the file is empty.

            sc.nextLine();

            System.out.println("Please enter the input filename:");
            final String input_file = sc.nextLine();

            try {
                File myObj = new File(input_file+".txt");

                Scanner myReader = new Scanner(myObj);

                while (myReader.hasNextLine()) {
                   for(int k=0; k<1; k++) {
                     String data = myReader.nextLine();
                       System.out.println(data);

                     try (final Connection connection = DriverManager.getConnection(URL)) {
                       try (
                          final PreparedStatement statement =
connection.prepareStatement(QUERY_TEMPLATE_16)) {



                          // Populate the query template with the data collected from the user
                          statement.setString(1, data);
                          data = myReader.nextLine();
                          statement.setString(2, data);
                          data = myReader.nextLine();
                          statement.setString(3, data);



                          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));
                       }
                    }

                    }
                }
                myReader.close();
              } catch (FileNotFoundException e) {
                System.out.println("An error occurred.");
```

```
            e.printStackTrace();
        }
        break;



case "17": // Retrieve
    sc.nextLine();
    System.out.println("Please enter the output filename ");
    final String outputFile = sc.nextLine();
    System.out.println("Connecting to the database...");
    // Get the database connection, execute procedure, as no user input need be collected
    try (final Connection connection = DriverManager.getConnection(URL)) {
        System.out.println("Dispatching the query...");
        try (
            final Statement statement = connection.createStatement();
            final ResultSet resultSet = statement.executeQuery(QUERY_TEMPLATE_17)) {

            try {
                FileWriter myWriter = new FileWriter(outputFile+".txt");

              myWriter.write("Contents of the QUERY: \n");
              myWriter.write("cust_name\n");
            //myWriter.write(resultSet.getString(1));
            while (resultSet.next()) {


              myWriter.write(resultSet.getString(1)+"\n");
           }

                myWriter.close();
                System.out.println("Successfully wrote to the file.");
              } catch (IOException e) {
                System.out.println("An error occurred.");
                e.printStackTrace();
              }
        }
    }

    break;

case "18": // Do nothing, the while loop will terminate upon the next iteration
    System.out.println("Exiting! Good-Bye!");
    break;
default: // Unrecognized option, re-prompt the user for the correct one
    System.out.println(String.format(
```

```
                "Unrecognized option: %s\n" +
                "Please try again!",
                option));
            break;
        }
    }

    sc.close(); // Close the scanner before exiting the application
}
}
```

Task 6

```
Please select one of the options below:
1) Insert new customer;
2) Insert new department;
3) Insert new process;
4) Insert new assembly;
5) Insert new account;
6) Insert new job;
7) Insert new job_end;
8) Insert update acct;
9) retrieve total labor time;
10)retreive total_labor_time;
11)retreive assm_id;
12)retrieve job;
13)retrieve category range;
14)delete job;
15)update paint;
16)import;
17)export;
18) Exit!
```

6.1. Screenshots showing testing of Query 1

```
1
Please enter Customer name:
Oyindamola
Please enter address:
2900 Beaumont
Please enter category
2
```

Customer

Results Messages

	cust_name	addr	category
1	Karen	Campus Lodge	9
2	Layefa	Lafayette	8
3	Obisesan	1800 Beaumont	8
4	Ochie	Oluyole Ibadan	10
5	Oyindamola	2900 Beaumont	2
6	Patience	Lekki, Lagos	3
7	Yetunde	Port Harcourt	9

6.2. Screenshots showing testing of Query 1

```
2
Please enter integer  department:
3
Please enter department details:
DSA
```

Departments

Results Messages

	dept_num	dep_data
1	1	ECO
2	3	DSA
3	4	PE
4	5	MATH
5	6	CHEM

6.3. Screenshots showing testing of Query 3

```
3
Please enter process id:
56
Please enter process data:
process data 1
Please enter department num:
4
Please enter type_process:
fit
Please enter cut type:

Please enter machine type:

Please enter fit type:
straight
Please enter paint type:

Please enter paint method:

Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Processes

Results Messages

	pid	process_data	dept_num
1	15	process data 1	4
2	56	process data 1	4
3	p111	process10	3
4	p123	process3	3
5	p339	process5	4
6	p345	Process1	5
7	p367	process2	6
8	p489	process4	4
9	p777	process7	1
10	p888	process8	6
11	p899	process6	1
12	p999	process9	5

Process_fit

Results Messages

	pid	fit_type	dept_num
1	15	straight	4
2	56	straight	4
3	p339	square	3
4	p367	sqaure	5
5	p777	circle	5
6	p899	fitted	4

Processes_paint

Results Messages

	pid	paint_type	paint_method	dept_num
1	p111	smooth	brush	4
2	p888	gloss	roller	6
3	p999	rough	roller	5

Processes_cut

straight

Results Messages

	pid	cut_type	machine_type	dept_num
1	p123	crook	saw	4
2	p345	straight	saw	3
3	p489	curve	jigjag	3

6.4. Screenshots showing testing of Query 4

```
4
Please enter assembly ID:
45
Please enter process id:
34
Please enter customer name:
Gloria
Please enter assembly date:
2020-01-03
Please enter assembly detail:
an assmebly
Please enter integer  department:
4
Please enter JOB ID:
234
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Assemblies

Results Messages

	assm_id	pid	cust_name	dat_assm	detail
1	45	34	Gloria	2020-01-03	an assmebly
2	a111	p111	Obisesan	2019-01-11	assm for obi
3	a123	p123	Oyindamola	2020-03-01	assm for Oyin
4	a339	p339	Patience	2019-03-01	assm for pat
5	a345	p345	Oyindamola	2021-01-01	assm for Oyin
6	a367	p367	Karen	2020-01-01	assm for karen
7	a489	p489	Karen	2019-01-01	assm for karen
8	a777	p777	Patience	2013-01-01	assm for pat
9	a888	p888	Ochie	2020-10-10	assm for ochie
10	a899	p899	Ochie	2020-01-01	assm for ochie
11	a999	p999	Obisesan	2020-10-10	assm for obi

Orders

Results Messages

	assm_id	cust_name
1	45	Gloria
2	a367	Karen
3	a489	Karen
4	a111	Obisesan
5	a999	Obisesan
6	a888	Ochie
7	a899	Ochie
8	a123	Oyindamola
9	a345	Oyindamola
1…	a339	Patience
1…	a777	Patience

Has

Results Messages

	assm_id	pid
1	34	234
2	45	34
3	a111	p111
4	a123	p123
5	a339	p339
6	a345	p345
7	a367	p367
8	a489	p489
9	a777	p777
10	a888	p888
11	a899	p899
12	a999	p999

6.5. Screenshots showing testing of Query 5

```
5
Please enter account number:
3456
Please enter process id:
456
Please enter assembly id:
432
Please enter date est:
2020-02-10
Please enter department number:
45
Please enter type:
department
Please enter cost of assem:


0
Please enter cost of dept:
300
Please enter cost of process:
0
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Acct_assm

Results Messages

	acct_num	assm_id	date_est	cost_assm
1	1000	a123	2020-03-02	60
2	2000	a345	2019-01-01	95
3	3000	a367	2021-01-01	111
4	4000	a489	2012-01-01	123

Acct_dep

	acct_num	dept_num	date_est	cost_dept
1	3456	45	2020-02-10	300
2	5000	5	2018-01-01	134
3	9000	4	2020-03-02	432
4	9900	3	2020-01-01	53

Acct_pro

Results Messages

	pid	acct_num	date_est	cost_pro
1	p123	6000	2020-03-02	755
2	p999	7000	2020-01-01	49
3	p999	8000	2020-03-01	155

6.6. Screenshots showing testing of Query 6

```
6
Please enter job id:
1234
Please enter date start:
2020-01-01
Please enter integer  department:
4
Please enter process id:
567
Please enter assembly ID:
a123
Please enter job_type:
fit

Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

done

	dep_num	pid	job_id
1	1	p123	45
2	1	p339	67
3	1	234	345
4	3	p777	32
5	3	p888	34
6	3	4432	2390
7	4	p345	56
8	4	p899	99
9	4	567	1234
10	5	p489	90
11	5	p999	100
12	6	p111	87
13	6	p367	102

Job_cut

Results Messages

	job_id	type_of_mac	amt_time	mat_used	labor_time_cut	date_start	date_end
1	34	*NULL*	*NULL*	*NULL*	*NULL*	2020-02-03	*NULL*
2	56	*NULL*	*NULL*	*NULL*	*NULL*	2020-03-03	*NULL*
3	87	*NULL*	*NULL*	*NULL*	*NULL*	2020-10-10	*NULL*
4	90	*NULL*	*NULL*	*NULL*	*NULL*	2020-01-01	*NULL*

Job_fit

Results grid

	job_id	labor_time_fit	date_start	date_end
1	32	*NULL*	2020-03-03	*NULL*
2	45	*NULL*	2020-02-03	*NULL*
3	99	*NULL*	2020-01-01	*NULL*
4	100	*NULL*	2020-11-11	*NULL*
5	2390	*NULL*	2020-01-01	*NULL*

Job_paint

Results Messages

	job_id	color	volume	labor_time_paint	date_start	date_end
1	67	*NULL*	*NULL*	*NULL*	2021-03-03	*NULL*
2	102	*NULL*	*NULL*	*NULL*	2020-02-03	*NULL*

6.7. Screenshots showing testing of Query 7

Job_cut

Results Messages

	job_id	type_of_mac	amt_time	mat_used	labor_time_cut	date_start	date_end
1	34	saw	45	dust	34	2020-02-03	2020-12-12
2	56	saw	45	dust	34	2020-03-03	2020-12-12
3	87	saw	40	dust	30	2020-10-10	2020-12-12
4	90	saw	5	dust	4	2020-01-01	2020-12-12

Job_fit

	job_id	labor_time_fit	date_start	date_end
1	32	67	2020-03-03	2021-12-12
2	45	70	2020-02-03	2021-11-12
3	99	67	2020-01-01	2021-12-12
4	100	67	2020-11-11	2021-12-12
5	2390	6	2020-01-01	2021-12-12

Job_paint

Results Messages

	job_id	color	volume	labor_time_paint	date_start	date_end
1	67	red	340	89	2021-03-03	2021-11-12
2	102	yellow	300	80	2020-02-03	2021-12-12

6.8. Screenshots showing testing of Query 8

```
8
Please enter trans id:
234
Please enter subcost:
30
Please enter acct type:
assembly
Please enter assembly ID:
12
Please enter acct num:
2390
Please enter date est:
2021-01-01
Connecting to the database...
Dispatching the query...
Done. 1 rows inserted.
```

Transactions

	trans_id	sub_cost
1	234	30
2	t1111	78
3	t1234	45
4	t1278	30
5	t1451	34
6	t6543	89
7	t6616	90
8	t6666	8
9	t6789	21
1…	t7777	10
1…	t7887	9

Updates

	acct_num	trans_id
1	1000	t1278
2	2000	t1234
3	2390	234
4	3000	t1451
5	4000	t6789
6	5000	t1111
7	6000	t7887
8	7000	t6666
9	8000	t7777
1…	9000	t6543
1…	9900	t6616

Acct_assem

Before:

	acct_num	assm_id	date_est	cost_assm
1	1000	a123	2020-03-02	180
2	2000	a345	2019-01-01	295
3	2390	5000	2020-01-01	53
4	3000	a367	2021-01-01	511
5	4000	a489	2012-01-01	215

After

	acct_num	assm_id	date_est	cost_assm
1	1000	a123	2020-03-02	210
2	2000	a345	2019-01-01	345
3	2390	5000	2020-01-01	63
4	3000	a367	2021-01-01	611
5	4000	a489	2012-01-01	238

Assem_pro

Before

	acct_num	dept_num	date_est	cost_dept
1	3456	45	2020-02-10	300
2	5000	5	2018-01-01	134
3	9000	4	2020-03-02	432
4	9900	3	2020-01-01	53

After

	acct_num	dept_num	date_est	cost_dept
1	3456	45	2020-02-10	331
2	5000	5	2018-01-01	227
3	9000	4	2020-03-02	531
4	9900	3	2020-01-01	89

Assm_pro

Before

Results Messages

	pid	acct_num	date_est	cost_pro
1	p123	6000	2020-03-02	1665
2	p999	7000	2020-01-01	139
3	p999	8000	2020-03-01	265

After

	pid	acct_num	date_est	cost_pro
1	p123	6000	2020-03-02	2120
2	p999	7000	2020-01-01	184
3	p999	8000	2020-03-01	320

6.9. Screenshots showing testing of Query 9

```
9
Please enter assembly ID:
a123
Connecting to the database...
        total_cost: 210
```

6.10. Screenshots showing testing of Query 10

```
10
Please department num:
3
Please date completed:
2021-12-12
Connecting to the database...
        total_labor_time: null
```

6.11. Screenshots showing testing of Query 11

```
11
Please enter assembly ID:
a999
Connecting to the database...
100 | 5
```

6.12. Screenshots showing testing of Query 12

```
12
Please enter job type:
fit
Please enter date ended:
2021-12-12
Please enter department number:
4
Connecting to the database...
a899 | p899 | Ochie | 2020-01-01 | assm for ochie | 4 | p899
```

6.13. Screenshots showing testing of Query 13

```
13
Please enter category 1:
3
Please enter category 2:
5
Connecting to the database...
        cust_name: Patience
```

6.14. Screenshots showing testing of Query 14

```
14
Please enter job_id 1:
10
Please enter job_id 2:
50
Connecting to the database...
Dispatching the query...
Done. 1 rows updated.
```

6.15. Screenshots showing testing of Query 15

```
15
Please enter job_id:
67
Please enter color:
purple
Connecting to the database...
Dispatching the query...
Done. 1 rows updated.
```

6.16. Screenshots showing testing of Query 16

Import

```
16
Please enter the input filename:
/Users/oyindamolaobisesan/Desktop/Project 1/Azure SQL Project 1/src/input
Joke
Dispatching the query...
Done. 1 rows inserted.

Dispatching the query...
Done. 1 rows inserted.
```

```
16
Please enter the input filename:
/Users/oyindamolaobisesan/Desktop/Project 1/Azure SQL Project 1/src/input
Deola
Dispatching the query...
Done. 1 rows inserted.
Judah
Dispatching the query...
Done. 1 rows inserted.
Jode
Dispatching the query...
Done. 1 rows inserted.
```

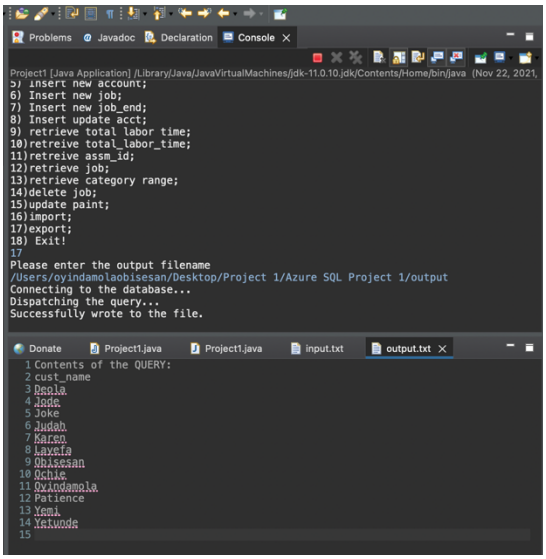
input.txt

```
Deola
Oyo
6
Judah
Campus Lodge
3
Jode
Nigeria
4
```

6.17. Screenshots showing testing of Query 17

Export

```
17
Please enter the output filename
/Users/oyindamolaobisesan/Desktop/Project 1/Azure SQL Project 1/output
Connecting to the database...
Dispatching the query...
Successfully wrote to the file.
```



6.17. Screenshots showing testing of Query 17

Quit

```
18
Exiting! Good-Bye!
```

Errors in Azure database

1. Category is an integer, and a string was supplied so it returned an error.

```
1
Please enter Customer name:
Akin
Please enter address:
Oluyole
Please enter category
here
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at Project1.main(Project1.java:94)
```

2. Department is an integer, and a float was supplied so it returned an error.

```
2
Please enter integer  department:
4.5
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at Project1.main(Project1.java:120)
```

3. account is an integer, and a string was supplied so it returned an error.

```
5
Please enter account number:
567y
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at Project1.main(Project1.java:278)
```

Task 7 – Web Database and its execution

7.1. Web database application source code

Datahandler

```
package project_azure;

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
     final static String host = "obis000-sql-
server.database.windows.net";
    final static String database = "cs-dsa-4513-sql-db";
    final static String username = "obis0000";
    final static String password = "Anec3030";

    // Resulting connection string
    final private String url =

String.format("jdbc:sqlserver://%s:1433;database=%s;user=%s;pass
word=%s;encrypt=true;trustServerCertificate=false;hostNameInCert
ificate=*.database.windows.net;loginTimeout=30;",
                    host, 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 getAllcust() throws SQLException {
        getDBConnection();

        final String sqlQuery = "SELECT * FROM customer;";
```

```
        final PreparedStatement stmt =
conn.prepareStatement(sqlQuery);
        return stmt.executeQuery();
    }

    // Inserts a record into the movie_night table with the
given attribute values
    public boolean addcustomer(
            String cust_name, String addr, int category) throws
SQLException {

        getDBConnection(); // Prepare the database connection

        // Prepare the SQL statement
        final String sqlQuery =
                "INSERT INTO customer " +
                    "(cust_name, addr, category) " +
                "VALUES " +
                "(?, ?, ?)";
        final PreparedStatement stmt =
conn.prepareStatement(sqlQuery);

        // Replace the '?' in the above statement with the given
attribute values
        stmt.setString(1, cust_name);
        stmt.setString(2, addr);
        stmt.setInt(3, category);


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

    //public ResultSet somecustomer() throws SQLException {
    public ResultSet somecustomer(int category_1, int
category_2) throws SQLException {
        getDBConnection(); // Prepare the database connection

        // Prepare the SQL statement
        final String sqlQuery =

              "select cust_name " +
             "from customer " +
             "where category between ? and ? " +
             "order by cust_name asc ";
```

```
        System.out.println(category_1);
        System.out.println(category_2);

        final PreparedStatement stmt =
conn.prepareStatement(sqlQuery);

        // Replace the '?' in the above statement with the given
attribute values
       //stmt.setInt(1, category_1);
       //stmt.setInt(2, category_2);
        //stmt.setInt(3, category);


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


}
```

Query 1

Get_all customers

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

```
                <td align="center">
                  <h4>addr</h4>
                </td>
                <td align="center">
                  <h4>category</h4>
                </td>
              </tr>
              <%
                 while(customer.next()) { // For each customer
record returned...
                    // Extract the attribute values for every row
returned
                    final String cust_name =
customer.getString("cust_name");
                    final String addr =
customer.getString("addr");
                    final String category =
customer.getString("category");


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

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

    // Get the attribute values passed from the input form.
    String cust_name = request.getParameter("cust_name");
    String address = request.getParameter("addr");
    String categoryString = request.getParameter("category");


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

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

            <ul>
                <li>cust_name: <%=cust_name%></li>
                <li>address: <%=address%></li>
                <li>category: <%=categoryString%></li>

            </ul>

            <h2>Was successfully inserted.</h2>

            <a href="get_all_cust.jsp">See all customers.</a>
```

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

Add_cust_form

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

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

For getting customer within a range – query 13

Get_some_cust (form)

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Get Customer</title>
    </head>
    <body>
        <h2>Get  Customer</h2>
        <!--
            Form for collecting user input for the new customer
record.
            Upon form submission, cust_some_.jsp file will be
invoked.
        -->
        <form action="cust_some.jsp">
            <!-- The form organized in an HTML table for better
clarity. -->
            <table border=1>
                <tr>
                    <th colspan="2">Enter the customer
Data:</th>
                </tr>
                <tr>
                    <td>Category_1:</td>
                    <td><div style="text-align: center;">
```

```
                    <input type=text name=category_1>
                    </div></td>
                </tr>
                <tr>
                    <td>Category_2:</td>
                    <td><div style="text-align: center;">
                    <input type=text name=category_2>
                    </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>

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

    // Get the attribute values passed from the input form.
    //String cust_name = request.getParameter("cust_name");
    //String address = request.getParameter("addr");
```

```
    String categoryString_1 =
request.getParameter("category_1");
    String categoryString_2 =
request.getParameter("category_2");



    /*
     * If the user hasn't filled out all the time, movie name
and duration. This is very simple checking.
     */
    if (categoryString_1.equals("") ||
categoryString_2.equals("")) {
        response.sendRedirect("get_some_cust.jsp");
    }
    else {
        int category_1 = Integer.parseInt(categoryString_1);
        int category_2 = Integer.parseInt(categoryString_2);

        // Now perform the query with the data from the form.
        //boolean success = handler.somecustomer(category_1,
category_2);
        //if (!success) { // Something went wrong
       // } else { // Confirm success to the user
           %>
            <h2>The Customers:</h2>

            <ul>
                <li>category: <%=categoryString_1%></li>
                <li>category: <%=categoryString_2%></li>

            </ul>

            <h2>customers were retrieved.</h2>

            <a href="get_all_cust.jsp">See all customers.</a>
        <%
            //final DataHandler handler = new DataHandler();
            final ResultSet customer =
handler.somecustomer(category_1, category_2);
           // final ResultSet customer =
handler.somecustomer(category_1, category_2);
        %>
        <!-- The table for displaying all the movie records -->
        <table cellspacing="2" cellpadding="2" border="1">
            <tr> <!-- The table headers row -->
```

```
                <td align="center">
                  <h4>cust_name</h4>
                </td>
              </tr>
              <%
                 while(customer.next()) { // For each movie_night
record returned...
                    // Extract the attribute values for every row
returned
                    final String cust_name =
customer.getString("cust_name");


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

7.2. screenshots showing the testing of the web database application

Query 13;

Get Customer

Enter the customer Data:	
Category_1:	2
Category_2:	10
Clear	Insert

The Customers:

- category: 2
- category: 10

customers were retrieved.

See all customers.

cust_name
Deola
Jode
Joke
Judah
Karen
Layefa
Obisesan
Ochie
Oyindamola
Patience
Yemi
Yetunde

Query 1;

Add Customer

Enter the customer Data:	
Customer Name:	Boluwatife
Address:	Bowling Green
Category:	3
Clear	Insert

The Customer:

- cust_name: Boluwatife
- address: Bowling Green
- category: 3

Was successfully inserted.

See all customers.

Query 13 – re run

Get Customer

Enter the customer Data:	
Category_1:	2
Category_2:	10
Clear	Insert

The Customers:

- category: 2
- category: 10

customers were retrieved.

See all customers.

cust_name
Boluwatife
Deola
Jode
Joke
Judah
Karen
Layefa
Obisesan
Ochie
Oyindamola
Patience
Yemi
Yetunde