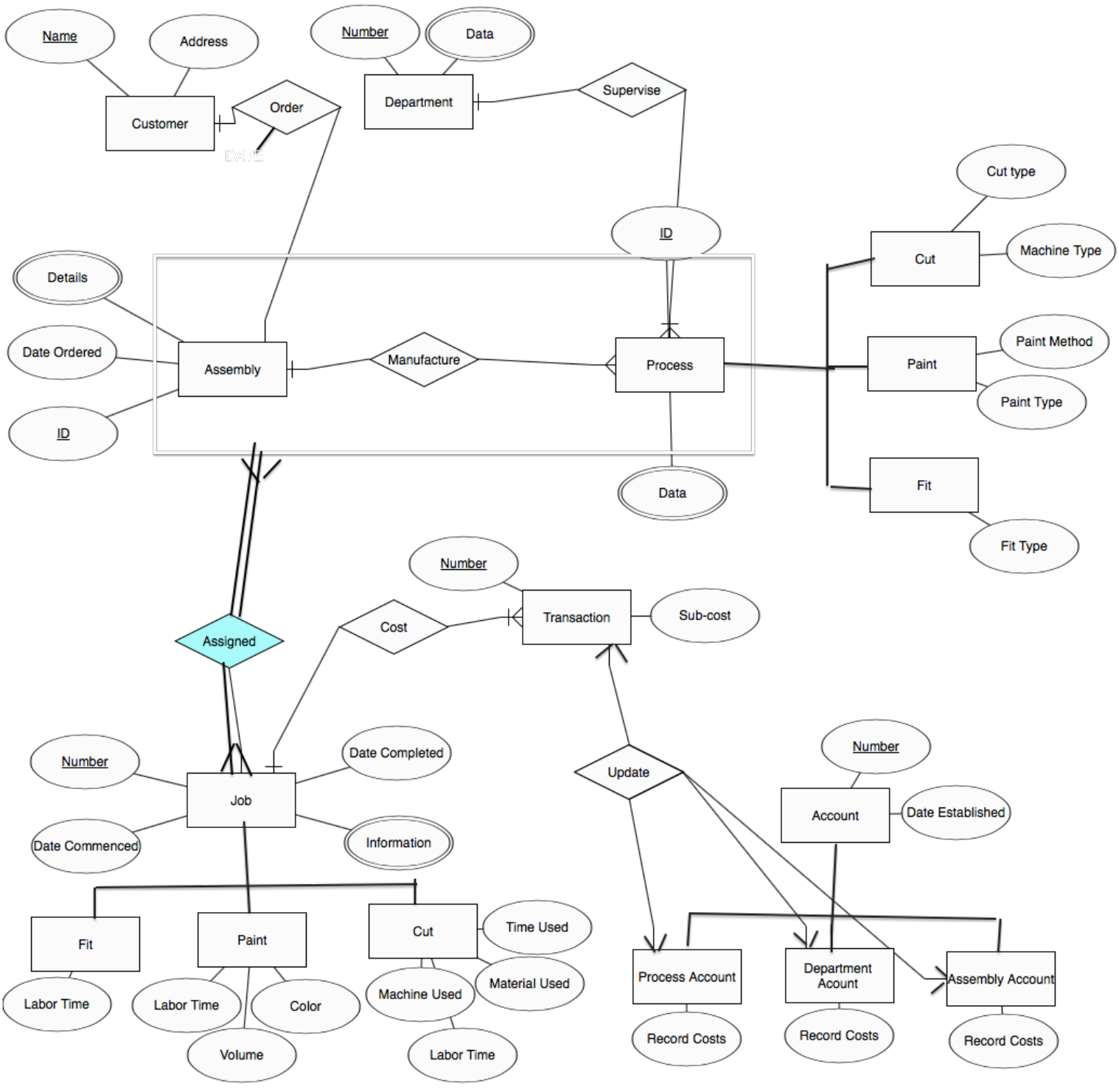
Individual Project

Database Management Systems
CS 4513 Section - 001
Dr. Gruenwald Fall 2015
By Koby Pascual
112766415
koby.pascual@gmail.com

Tasks Performed **Page Number**


Name
Address
Customer
Order
DATE
Number
Data
Department
Supervise
ID
Details
Date Ordered
ID
Assembly
Manufacture
Process
Data
Cut type
Cut
Machine Type
Paint
Paint Method
Paint Type
Fit
Fit Type
Assigned
Number
Cost
Transaction
Sub-cost
Number
Date Completed
Job
Date Commenced
Information
Update
Number
Account
Date Established
Fit
Labor Time
Paint
Labor Time
Color
Volume
Cut
Time Used
Machine Used
Material Used
Labor Time
Process Account
Record Costs
Department Acount
Record Costs
Assembly Account
Record Costs

TASK 2:

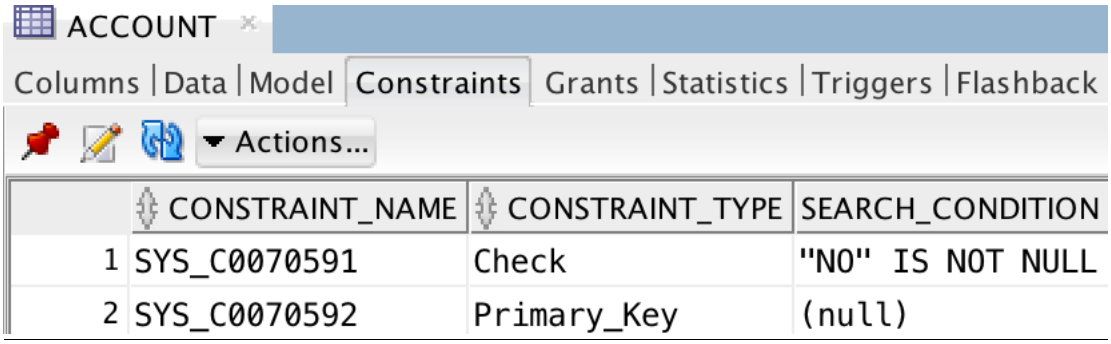
ACCOUNT:

ACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partiti

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	DATE_ESTABLISHED	DATE	Yes	(null)	2	(null)



ASSEMBLY:

ASSEMBLY

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID	NUMBER(5,0)	No	(null)	1	(null)
2	DATE_ORDERED	DATE	Yes	(null)	2	(null)
3	DETAILS	VARCHAR2(100 BYTE)	Yes	(null)	3	(null)
4	CUST	VARCHAR2(30 BYTE)	No	(null)	4	(null)

ASSEMBLY

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | De

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070555	Check	"ID" IS NOT NULL
2	SYS_C0070556	Check	"CUST" IS NOT NULL
3	SYS_C0070557	Primary_Key	(null)
4	SYS_C0070558	Foreign_Key	(null)

ASSEMBLY ACCOUNT:

ASSEMBLYACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partiti

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	RECORD_COSTS	NUMBER(8,2)	Yes	(null)	2	(null)
3	ASS_ID	NUMBER(5,0)	No	(null)	3	(null)

ASSEMBLYACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depe

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070593	Check	"NO" IS NOT NULL
2	SYS_C0070594	Check	"ASS_ID" IS NOT NULL
3	SYS_C0070595	Primary_Key	(null)
4	SYS_C0070596	Foreign_Key	(null)
5	SYS_C0070597	Foreign_Key	(null)

CUSTOMER:

CUSTOMER

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | D

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070553	Check	"NAME" IS NOT NULL
2	SYS_C0070554	Primary_Key	(null)

CUSTOMER

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NAME	VARCHAR2(30 BYTE)	No	(null)	1	(null)
2	ADDRESS	VARCHAR2(30 BYTE)	Yes	(null)	2	(null)

CUTJOB:

CUTJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070582	Check	"NO" IS NOT NULL
2	SYS_C0070583	Primary_Key	(null)
3	SYS_C0070584	Foreign_Key	(null)

CUTJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	MACHINE_USED	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)
3	TIME_USED	NUMBER(6,2)	Yes	(null)	3	(null)
4	MATERIAL_USED	VARCHAR2(20 BYTE)	Yes	(null)	4	(null)
5	LABOR_TIME	NUMBER(6,2)	Yes	(null)	5	(null)

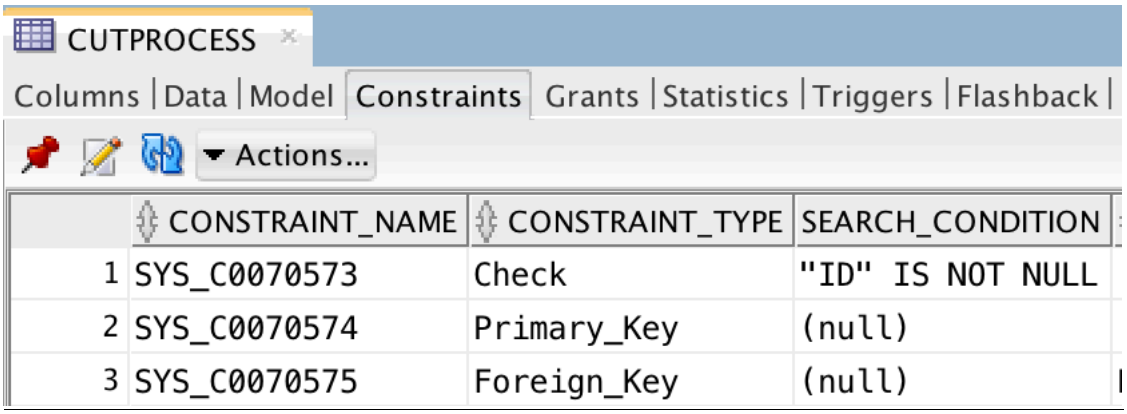
CUTPROCESS:

CUTPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID	NUMBER(5,0)	No	(null)	1	(null)
2	CUT_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)
3	MACHINE_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)



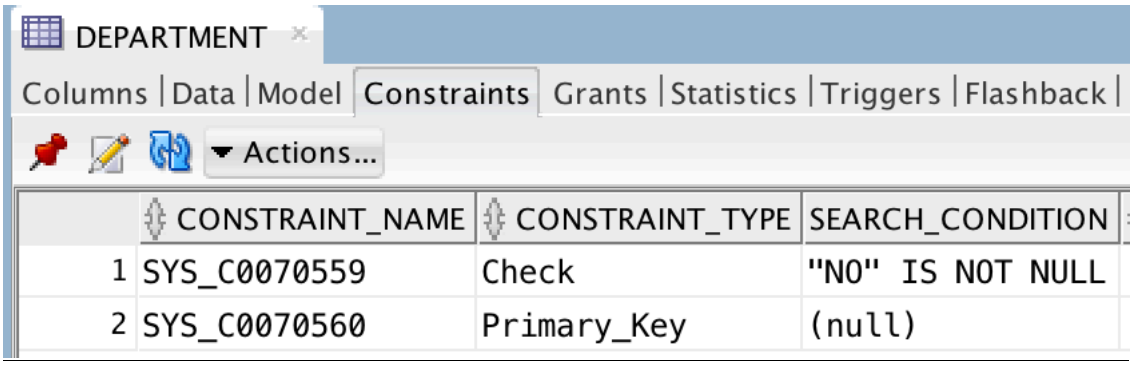
CUTPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070573	Check	"ID" IS NOT NULL
2	SYS_C0070574	Primary_Key	(null)
3	SYS_C0070575	Foreign_Key	(null)

DEPARTMENT:

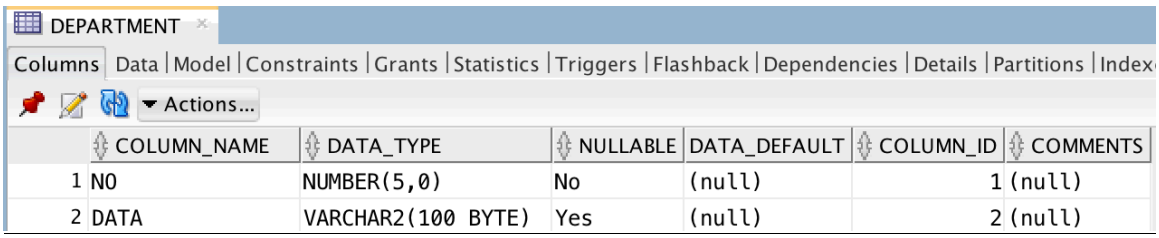


DEPARTMENT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070559	Check	"NO" IS NOT NULL
2	SYS_C0070560	Primary_Key	(null)



DEPARTMENT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	DATA	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)

DEPARTMENT ACCOUNT:

DEPARTMENTACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	RECORD_COSTS	NUMBER(8,2)	Yes	(null)	2	(null)
3	DEPT_ID	NUMBER(5,0)	No	(null)	3	(null)

DEPARTMENTACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depend

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070598	Check	"NO" IS NOT NULL
2	SYS_C0070599	Check	"DEPT_ID" IS NOT NULL
3	SYS_C0070600	Primary_Key	(null)
4	SYS_C0070601	Foreign_Key	(null)
5	SYS_C0070602	Foreign_Key	(null)

FITJOB:

FITJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070588	Check	"NO" IS NOT NULL
2	SYS_C0070589	Primary_Key	(null)
3	SYS_C0070590	Foreign_Key	(null)

FITJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	LABOR_TIME	NUMBER(6,2)	Yes	(null)	2	(null)

FITPROCESS:

FITPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070567	Check	"ID" IS NOT NULL
2	SYS_C0070568	Primary_Key	(null)
3	SYS_C0070569	Foreign_Key	(null)

FITPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexe

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID	NUMBER(5,0)	No	(null)	1	(null)
2	FIT_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)

JOB:

JOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depe

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070576	Check	"NO" IS NOT NULL
2	SYS_C0070577	Check	"PRO_NO" IS NOT NULL
3	SYS_C0070578	Check	"ASS_ID" IS NOT NULL
4	SYS_C0070579	Primary_Key	(null)
5	SYS_C0070580	Foreign_Key	(null)
6	SYS_C0070581	Foreign_Key	(null)

JOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	DATE_COMMENCED	DATE	Yes	(null)	2	(null)
3	DATE_COMPLETED	DATE	Yes	(null)	3	(null)
4	INFORMATION	VARCHAR2(100 BYTE)	Yes	(null)	4	(null)
5	PRO_NO	NUMBER(5,0)	No	(null)	5	(null)
6	ASS_ID	NUMBER(5,0)	No	(null)	6	(null)

PAINTJOB:

PAINTJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070585	Check	"NO" IS NOT NULL
2	SYS_C0070586	Primary_Key	(null)
3	SYS_C0070587	Foreign_Key	(null)

PAINTJOB

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	COLOR	VARCHAR2(10 BYTE)	Yes	(null)	2	(null)
3	VOLUME	VARCHAR2(10 BYTE)	Yes	(null)	3	(null)
4	LABOR_TIME	NUMBER(6,2)	Yes	(null)	4	(null)

PAINT PROCESS:

PAINTPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID	NUMBER(5,0)	No	(null)	1	(null)
2	PAINT_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)
3	PAINT_METHOD	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)

PAINTPROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback |

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070570	Check	"ID" IS NOT NULL
2	SYS_C0070571	Primary_Key	(null)
3	SYS_C0070572	Foreign_Key	(null)

PROCESS:

PROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depen

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	
1	SYS_C0070561	Check	"ID" IS NOT NULL	(
2	SYS_C0070562	Check	"ASS_ID" IS NOT NULL	(
3	SYS_C0070563	Check	"DEP_NO" IS NOT NULL	(
4	SYS_C0070564	Primary_Key	(null)	(
5	SYS_C0070565	Foreign_Key	(null)	P
6	SYS_C0070566	Foreign_Key	(null)	P

PROCESS

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID	NUMBER(5,0)	No	(null)	1	(null)
2	DATA	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)
3	ASS_ID	NUMBER(5,0)	No	(null)	3	(null)
4	DEP_NO	NUMBER(5,0)	No	(null)	4	(null)
5	TYPE	VARCHAR2(10 BYTE)	Yes	(null)	5	(null)

PROCESS ACCOUNT:

PROCESSACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	RECORD_COSTS	NUMBER(8,2)	Yes	(null)	2	(null)
3	PRO_ID	NUMBER(5,0)	No	(null)	3	(null)

PROCESSACCOUNT

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depe

Actions...

	CONSTRAINT_N...	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070603	Check	"NO" IS NOT NULL
2	SYS_C0070604	Check	"PRO_ID" IS NOT NULL
3	SYS_C0070605	Primary_Key	(null)
4	SYS_C0070606	Foreign_Key	(null)
5	SYS_C0070607	Foreign_Key	(null)

TRANSACTION:

TRANSACTION

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Depe

Actions...

	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION
1	SYS_C0070608	Check	"NO" IS NOT NULL
2	SYS_C0070609	Check	"JOB_NO" IS NOT NULL
3	SYS_C0070610	Primary_Key	(null)
4	SYS_C0070611	Foreign_Key	(null)

TRANSACTION

Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Index

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	NO	NUMBER(5,0)	No	(null)	1	(null)
2	SUP_COST	NUMBER(8,2)	Yes	(null)	2	(null)
3	JOB_NO	NUMBER(5,0)	No	(null)	3	(null)

TASK 3:

3.1

a) Table Customer:
    a. Since this is a fairly simple table, I would use an indexed table on the primary key name. However, since this is already happening by default, we would leave it. We would index-sequential. This is a problem if it gets large.
b) Table Assembly:
    a. I would probably use a B+ tree here. It is similar to Customer however, so indexed sequential could be used. It will grow rapidly, so b+ tree could be useful in the long run.
c) Table Fit Process, Print Process, Cut Process
    a. Indexed Sequential. Though these are often called on a daily basis, the amount of processes stays relatively constant over time. Thus, growing in the DB isn't really too much of a concern.
d) Table Department
    a. Just like part (c) departments are very rarely added. Thus, extra cost for fancy structures is not necessary. Department is also not called very often.
e) Table Account, Department Account, Assembly Account, Process Account
    a. Unlike some of the previous tables, a b+tree would be a much better structure here even though it is much more complicated for insertion. This is a prime example of a table needing search to be efficient. These tables are referenced many times per day, thus speedy searching is important. However, since not many accounts are added, maybe a hashing function would be better?
f) Table Job, Fit Job, Cut Job, Paint Job
    a. This would also be another case where a sophisticated structure would be preferred. Not only are hundreds of queries per day accessing these tables, but there are also large amounts of jobs being added daily. Speed is important here.
g) Table Transaction
    a. Jobs are often referenced, which means costs is also, so it is likely a great idea to have a B+ tree here as well. However, another type of structure could make an argument as well. This is because, even though a lot of transactions are happening, they aren't often being referenced in other queries. This is because the accounts hold their own record of the cost.

3.2

Since this was a smaller project (in terms of loading data) no further implementation of the data structures was needed when creating the tables. Indexing could have been easily accomplished, however.

TASK 4:

```
drop table transaction;
drop table assemblyaccount;
drop table fitprocess;
drop table paintprocess;
drop table cutprocess;
drop table processaccount;
drop table cutjob;
drop table paintjob;
drop table fitjob;
drop table job;
drop table process;
drop table departmentaccount;
drop table department;
drop table account;
drop table assembly;
drop table customer;


create table customer
(
 name varchar(30) not null,
 address varchar(30),
 primary key(name)
);

create table assembly
(
 ID numeric(5,0) not null,
 date_ordered date,
 details varchar(100),
 cust varchar(30) not null,
 primary key(ID),
 foreign key(cust) references customer(name)
);

create table department
(
 no numeric(5,0) not null,
 data varchar(100),
 primary key(no)
);

create table process
(
```

```
  ID numeric(5,0) not null,
  data varchar(100),
  ass_id numeric(5,0) not null,
  dep_no numeric(5,0) not null,
  type varchar(10),
  primary key(ID),
  foreign key(ass_id) references assembly(ID),
  foreign key(dep_no) references department(no)
);

create table fitprocess
(
  ID numeric(5,0) not null references process(ID),
  fit_type varchar(50),
  primary key(ID)
);

create table paintprocess
(
  ID numeric(5,0) not null references process(ID),
  paint_type varchar(50),
  paint_method varchar(50),
  primary key(ID)
);

create table cutprocess
(
  ID numeric(5,0) not null references process(ID),
  cut_type varchar(50),
  machine_type varchar(50),
  primary key(ID)
);

create table job
(
  no numeric(5,0) not null,
  date_commenced date,
  date_completed date,
  information varchar(100),
  pro_no numeric(5,0) not null,
  ass_id numeric(5,0) not null,
  primary key(no),
  foreign key(pro_no) references process(ID),
  foreign key(ass_id) references assembly(ID)
);
```

```
create table cutjob
(
 no numeric(5,0) not null references job(no),
 machine_used varchar(50),
 time_used numeric(6,2),
 material_used varchar(20),
 labor_time numeric(6,2),
 primary key(no)
);

create table paintjob
(
 no numeric(5,0) not null references job(no),
 color varchar(10),
 volume varchar(10),
 labor_time numeric(6,2),
 primary key(no)
);

create table fitjob
(
 no numeric(5,0) not null references job(no),
 labor_time numeric(6,2),
 primary key(no)
);

create table account
(
 no numeric(5,0) not null,
 date_established date,
 primary key(no)
);

create table assemblyaccount
(
 no numeric(5,0) not null references account(no),
 record_costs numeric(8,2),
 ass_id numeric(5,0) not null,
 primary key(no),
 foreign key(ass_id) references assembly(ID)
);

create table departmentaccount
(
 no numeric(5,0) not null references account(no),
 record_costs numeric(8,2),
```

```
  dept_id numeric(5,0) not null,
  primary key(no),
  foreign key(dept_id) references department(no)
);

create table processaccount
(
  no numeric(5,0) not null references account(no),
  record_costs numeric(8,2),
  pro_id numeric(5,0) not null,
  primary key(no),
  foreign key(pro_id) references process(ID)
);

create table transaction
(
  no numeric(5,0) not null,
  sup_cost numeric(8,2),
  job_no numeric(5,0) not null,
  primary key(no),
  foreign key(job_no) references job(no)
);
```

TASK 5:

PROJECT CLASS:

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.io.UnsupportedEncodingException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Scanner;


public class Project
{
	// Project Class variables

	// The connection to the dbms
	private static Connection dbms;
	// The statement to be used to execute sql updates
	private Statement stmt;
	// Scanner to be used by the system
	Scanner scan = new Scanner(System.in);

	// Constructor for the project class - Entire program runs
through here
	public Project()
	{
		// Load the information to the sql db and start the
connection
		loadJdbc();
		// Create the DBMS by creating the tables to be used
	//	createDBMS();

		// Begin a while loop to prompt the user which query
they would like to perform. Execute query and don't quit until
user submits 20
		int choice = 0;
		while(choice != 20)
```

```
            {
                  // Find query wanted by user
                  choice = callQueries();
            }

            try
            {
                  // Close the connection
                  dbms.close();
            } catch (SQLException e)
            {
                  e.printStackTrace();
            }
      }

      // Load JDBC - Opens the connection to the DB
      private void loadJdbc()
      {
            // Load JDBC Driver
            try
            {
                  Class.forName("oracle.jdbc.OracleDriver");
            }
            catch(Exception e)
            {
                  System.out.println("Unable to load the driver
class!");
            }

            try
            {
                  // Create an open connection to the oracle db
                  dbms =
DriverManager.getConnection("jdbc:oracle:thin:@//oracle.cs.ou.edu
:1521/pdborcl.cs.ou.edu", "pasc6415", "WHji6Vv8");
                  stmt = dbms.createStatement();
            }
            catch(SQLException e)
            {
                  System.err.println("Couldn't get connection!");
            }
      }
      // Create DBMS method - Not Used
      private void createDBMS()
```

```
	{
		// Drop the tables
		dropTables();
		// Create the tables
		createTables();
	}
	// Drop Tables Method - Not Used
	private void dropTables()
	{
		// Create a string for each table to be dropped
		String customer, assembly, process, fitprocess,
paintprocess, cutprocess, department, job, cutjob, paintjob,
fitjob, account, assemblyaccount, departmentaccount,
processaccount, transaction;

		customer = "drop table customer";
		assembly = "drop table assembly";
		fitprocess = "drop table paintprocess";
		paintprocess = "drop table paintprocess";
		cutprocess = "drop table cutprocess";
		process = "drop table process";
		department = "drop table department";
		cutjob = "drop table cutjob";
		paintjob = "drop table paintjob";
		fitjob = "drop table fitjob";
		job = "drop table job";
		assemblyaccount = "drop table assemblyaccount";
		departmentaccount = "drop table departmentaccount";
		processaccount = "drop table processaccount";
		account = "drop table account";
		transaction = "drop table transaction";

		try
		{
			// Execute dropping each table
			stmt.executeUpdate(customer);
			stmt.executeUpdate(assembly);
			stmt.executeUpdate(fitprocess);
			stmt.executeUpdate(paintprocess);
			stmt.executeUpdate(cutprocess);
			stmt.executeUpdate(process);
			stmt.executeUpdate(department);
			stmt.executeUpdate(cutjob);
			stmt.executeUpdate(paintjob);
```

```
                stmt.executeUpdate(fitjob);
                stmt.executeUpdate(job);
                stmt.executeUpdate(assemblyaccount);
                stmt.executeUpdate(departmentaccount);
                stmt.executeUpdate(processaccount);
                stmt.executeUpdate(account);
                stmt.executeUpdate(transaction);
            }
            catch(SQLException e)
            {
                System.err.println("SQLException:" +
e.getMessage());
            }
        }
        // Create Tables Method - Not Used
        private void createTables()
        {
            // Create all of the tables that need to be created
            String customer, assembly, process, fitprocess,
paintprocess, cutprocess, department, job, cutjob, paintjob,
fitjob, account, assemblyaccount, departmentaccount,
processaccount, transaction;

            transaction = "create table transaction" +
                                "("+
                                "no numeric(5,0),"+
                                "sup_cost numeric(8,2),"+
                                "primary key(no)"+
                                ")";

            customer = "create table customer" +
                            "(" +
                            "name varchar(30),"+
                            "address varchar(30),"+
                            "primary key(name)"+
                            ")";

            assembly = "create table assembly"+
                            "("+
                            "ID numeric(5,0),"+
                            "date_ordered varchar(10),"+
                            "details varchar(100),"+
                            "primary key(ID)"+
                            ")";
```

```
            process = "create table process"+
                                "("+
                                "ID numeric(5,0),"+
                                "data varchar(100),"+
                                "primary key(ID)"+
                                ")";


            fitprocess = "create table fitprocess"+
                                        "("+
                                        "ID numeric(5,0) references
process(ID),"+
                                        "fit_type varchar(50),"+
                                        "primary key(ID)"+
                                        ")";


            paintprocess = "create table paintprocess"+
                        "("+
                        "ID numeric(5,0) references process(ID),"+
                        "paint_type varchar(50),"+
                        "paint_method varchar(50),"+
                        "primary key(ID)"+
                        ")";


            cutprocess = "create table cutprocess"+
                        "("+
                        "ID numeric(5,0) references process(ID),"+
                        "cut_type varchar(50),"+
                        "machine_type varchar(50),"+
                        "primary key(ID)"+
                        ")";


            department = "create table department"+
                                        "("+
                                        "no numeric(5,0),"+
                                        "data varchar(100),"+
                                        "primary key(no)"+
                                        ")";
```

```
job = "create table job"+
            "("+
            "no numeric(5,0),"+
            "date_commenced varchar(10),"+
            "date_completed varchar(10),"+
            "information varchar(100),"+
            "primary key(no)"+
            ")";

cutjob = "create table cutjob"+
                  "("+
                  "no numeric(5,0) references job(no),"+
                  "machine_used varchar(50),"+
                  "time_used varchar(20),"+
                  "material_used varchar(20),"+
                  "labor_time varchar(20),"+
                  "primary key(no)"+
                  ")";

paintjob = "create table paintjob"+
                  "("+
                  "no numeric(5,0) references job(no),"+
                  "color varchar(10),"+
                  "volume varchar(10),"+
                  "labor_time varchar(20),"+
                  "primary key(no)"+
                  ")";

fitjob = "create table fitjob"+
                  "("+
                  "no numeric(5,0) references job(no),"+
                  "labor_time varchar(20),"+
                  "primary key(no)"+
                  ")";

account = "create table account"+
                  "("+
                  "no numeric(5,0),"+
                  "date_established varchar(10),"+
                  "primary key(no)"+
                  ")";

assemblyaccount = "create table assemblyaccount" +
                              "("+
```

```
                                        "no numeric(5,0) references
account(no),"+
                                        "record_costs numeric(8,
2),"+
                                        "primary key(no)"+
                                        ")";

            departmentaccount = "create table departmentaccount"+
                                        "("+
                                        "no numeric(5,0) references
account(no),"+
                                        "record_costs
numeric(8,2),"+
                                        "primary key(no)"+
                                        ")";

            processaccount = "create table processaccount"+
                                        "("+
                                        "number numeric(5,0)
references account(no),"+
                                        "record_costs
numeric(8,2),"+
                                        "primary key(no)"+
                                        ")";
            try
            {
                  // Execute the creation of all the tables
                  int x = 0; System.out.println(x); x++;
                  stmt.executeUpdate(transaction);
System.out.println(x); x++;
                  stmt.executeUpdate(customer);
System.out.println(x); x++;
                  stmt.executeUpdate(assembly);
System.out.println(x); x++;
                  stmt.executeUpdate(process);
System.out.println(x); x++;
                  stmt.executeUpdate(fitprocess);
System.out.println(x); x++;
                  stmt.executeUpdate(paintprocess);
System.out.println(x); x++;
                  stmt.executeUpdate(cutprocess);
System.out.println(x); x++;
                  stmt.executeUpdate(department);
System.out.println(x); x++;
```

```
                stmt.executeUpdate(job); System.out.println(x);
x++;
                stmt.executeUpdate(cutjob);
System.out.println(x); x++;
                stmt.executeUpdate(paintjob);
System.out.println(x); x++;
                stmt.executeUpdate(fitjob);
System.out.println(x); x++;
                stmt.executeUpdate(account);
System.out.println(x); x++;
                stmt.executeUpdate(assemblyaccount);
System.out.println(x); x++;
                stmt.executeUpdate(departmentaccount);
System.out.println(x); x++;
                stmt.executeUpdate(processaccount);
System.out.println(x); x++;
                System.out.println("all good");
           }
           catch(SQLException e)
           {
                System.err.println("SQLException:" +
e.getMessage());
           }
     }
     // Method that will prompt the user and call the proper
queries
     private int  callQueries()
     {
           // Prompt the user which query they want to use
           System.out.println("");
           System.out.println("WELCOME TO THE JOB-SHOP ACCOUNTING
DATABASE SYSTEM");
           System.out.println("");
           System.out.println("Input 1: to Enter a New
Customer.");
           System.out.println("Input 2: to Enter a New
Department.");
           System.out.println("Input 3: to Enter a New
Assembly.");
           System.out.println("Input 4: to Enter a New Process ID
and its Department.");
           System.out.println("Input 5: to Create a New
Account.");
           System.out.println("Input 6: to Enter a New Job.");
```

```
            System.out.println("Input 7: to Enter Completion Date
and Info of a Job.");
            System.out.println("Input 8: to Enter a Transaction
number and it's Sup-cost.");
            System.out.println("Input 9: to Retrieve Cost Incurred
on Assembly-ID.");
            System.out.println("Input 10: to Retrieve Labor Time
Recorded on Assembly-ID.");
            System.out.println("Input 11: to Retrieve Labor Time
within a Department for Jobs Completed in the Department given a
Date.");
            System.out.println("Input 12: to Retrieve the
Processes through which a given Assembly-ID has passed so far and
Department Responsible for each Process.");
            System.out.println("Input 13: to Retrieve Jobs
Completed given a Date in a given Department.");
            System.out.println("Input 14: to Retreive the
Customers whose Assemblies are Painted RED using a given Painting
Method.");
            System.out.println("Input 15: to Delete all Cut Jobs
whose Job Number lies in a Range.");
            System.out.println("Input 16: to Change the Color of a
given Paint Job.");
            System.out.println("Input 17: to Retrieve the Average
Cost of all Accounts.");
            System.out.println("Input 18: Import");
            System.out.println("Input 19: Export");
            System.out.println("Input 20: Quit");

            // Take the users choice
            int choice = scan.nextInt();
            scan.nextLine();

            // If choice is less than 1 or greater than 20 it is
incorrect. Ask them again
            if(!((choice >= 1) && (choice <=20)))
            {
                  // Tell user that it was incorrect
                  System.out.println("Invalid input. Please try
again.");
                  choice = 0;
            }

            else
```

```
            {
                  // Take a valid choice and execute the query
                  if(choice == 1)
                        insertNewCustomer();  // insert a new
customer
                  else if(choice == 2)
                        insertNewDepartment();     // insert a new
department
                  else if(choice == 3)
                        insertNewAssembly();  // insert a new
assembly
                  else if(choice == 4)
                        insertNewProcess();        // insert a new
process
                  else if(choice == 5)
                        insertNewAccount();        // insert new
account
                  else if(choice == 6)
                        insertNewJob();            // insert new
job
                  else if(choice == 7)
                        completeJob();             // complete job
                  else if(choice == 8)
                        insertNewTransaction();    // insert new
transaction
                  else if(choice == 9)
                        retrieveCostOfAssembly(); // retrieve the
cost of an assembly
                  else if(choice == 10)
                        retrieveLaborTimeOfAssembly();        //
retrieve labor time of an assembly
                  else if(choice == 11)
                        retrieveTotalLaborTimeOfDepartment(); //
retrieve total labor time of a department
                  else if(choice == 12)
                        retrieveProcessAssemblyPassed(); //
retrieve processes an assembly has passed
                  else if(choice == 13)
                        retrieveJobs();            // retrieve jobs
                  else if(choice == 14)
                        retrieveRedAssemblies(); // retrieve red
assemblies
                  else if(choice == 15)
                        deleteCutJobs();     // delete cut jobs
```

```
                else if(choice == 16)
                        changePaintJobColor();      // change paint
job color
                else if(choice == 17)
                        retrieveAverageCostOfAccounts(); //
retrieve average cost of accounts
                else if(choice == 18)
                        importCustomers();          // import
customers
                else if(choice == 19)
                        exportCustomers();          // export
customers
                else if(choice == 20)
                        System.out.println("See you next time!");
            }
            return choice;
      }
      // Query 1 - Insert New Customer - Completed
      private void insertNewCustomer()
      {
            // Get customer info from user
            System.out.println("Enter the new Customer's unique
name");
            String name = scan.nextLine();
            System.out.println("Enter the new Customer's
address");
            String address = scan.nextLine();

            String insert = "insert into customer
values('"+name+"','"+address+"')";

            // Add customer to db
            try
            {
                  stmt.executeUpdate(insert);
            } catch(SQLException e)
            {
                  System.err.println("SQLException:" +
e.getMessage());
            }
      }
      // Query 2 - Insert New Department - Completed
      private void insertNewDepartment()
      {
```

```
            // Ask user for department information
            System.out.println("Enter the Department ID");
            int id = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter any Data associated with
this Department.");
            String data = scan.nextLine();

            String insert = "insert into department
values("+id+",'"+data+"')";

            // Insert department info into db
            try
            {
                  stmt.executeUpdate(insert);
            } catch(SQLException e)
            {
                  System.err.println("SQLException:" +
e.getMessage());
            }
      }
      // Query 3 - Insert New Assembly - Completed
      private void insertNewAssembly()
      {
            System.out.println("Enter the Assembly ID");
            int id = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the date ordered.
mmddyyyy");
            String date = scan.nextLine();
            System.out.println("Enter additional Assembly
details");
            String details = scan.nextLine();
            System.out.println("Enter the customers ID that
ordered the assembly");
            String customer = scan.nextLine();

            String insert = "insert into assembly values(" + id +
", to_date('" + date + "', 'mmddyyyy') ,'" + details + "','" +
customer + "')";

            try
            {
                  stmt.executeUpdate(insert);
```

```
            } catch(SQLException e)
            {
                  System.err.println("SQLException:" +
e.getMessage());
            }
      }
      // Query 4 - Insert New Process - Completed
      private void insertNewProcess()
      {
            System.out.println("Enter the type of process. 1 for
Fit, 2 for Paint, 3 for Cut");
            int type = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter new Process ID");
            int id = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter Process data");
            String data = scan.nextLine();
            System.out.println("Enter the Assembly ID");
            int assid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the Processes Department
Number");
            int depid = scan.nextInt();
            scan.nextLine();

            String t;
            String entertype;
            String insertType;
            while(true)
            {
                  if(type == 1)
                  {
                        t = "fitprocess";
                        entertype = "fit";
                        System.out.println("Enter the type of Fit
Process");
                        String fittype = scan.nextLine();

                        insertType = "insert into " + t + "
values(" + id + ",'" + fittype + "')";
                        break;
                  }
                  else if(type == 2)
```

```
                {
                        t = "paintprocess";
                        entertype = "paint";
                        System.out.println("Enter the paint type");
                        String painttype = scan.nextLine();
                        System.out.println("Enter the type of paint
method");
                        String paintmethod = scan.nextLine();

                        insertType = "insert into " + t + "
values(" + id + ",'" + painttype + "','" + paintmethod + "')";
                        break;
                }
                else if(type == 3)
                {
                        t = "cutprocess";
                        entertype = "cut";
                        System.out.println("Enter the cut type");
                        String cuttype = scan.nextLine();
                        System.out.println("Enter the machine
type");
                        String machinetype = scan.nextLine();

                        insertType = "insert into " + t + "
values(" + id + ",'" + cuttype + "','" + machinetype + "')";
                        break;
                }

                else
                {
                        System.out.println("Enter the type of
process. 1 for Fit, 2 for Paint, 3 for Cut");
                        type = scan.nextInt();
                        scan.nextLine();
                }
        }

        String insert = "insert into process values(" + id +
",'" + data + "'," + assid + "," + depid + ",'" + entertype +
"')";

        try
        {
                stmt.executeUpdate(insert);
```

```
                stmt.executeUpdate(insertType);
            } catch(SQLException e)
            {
                System.err.println("SQLException: " +
e.getMessage());
            }
      }
      // Query 5 - Insert New Account - Completed
      private void insertNewAccount()
      {
            System.out.println("Enter the type of Account. 1 for
Assembly, 2 for Department, 3 for Process");
            int type = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the account number");
            int number = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the date established.
mmddyyyy");
            String date = scan.nextLine();

            String t;
            String insertAccount;
            while(true)
            {
                  if(type == 1)
                  {
                        t = "assemblyaccount";
                        System.out.println("Enter the Assembly ID
that this account is for.");
                        int assid = scan.nextInt();
                        scan.nextLine();

                        insertAccount = "insert into " + t + "
values(" + number + "," + 0 + "," + assid + ")";
                        break;
                  }
                  else if(type == 2)
                  {
                        t = "departmentaccount";
                        System.out.println("Enter this Department
number that this account is for.");
                        int depid = scan.nextInt();
                        scan.nextLine();
```

```
                        insertAccount = "insert into " + t + "
values(" + number + "," + 0 + "," + depid + ")";
                        break;
                  }
                  else if(type == 3)
                  {
                        t = "processaccount";
                        System.out.println("Enter this Process that
this account is for.");
                        int proid = scan.nextInt();
                        scan.nextLine();

                        insertAccount = "insert into " + t + "
values(" + number + "," + 0 + "," + proid + ")";
                        break;
                  }

                  else
                  {
                        System.out.println("Enter the type of
Account. 1 for Assembly, 2 for Department, 3 for Process");
                        type = scan.nextInt();
                        scan.nextLine();
                  }
            }

            String insert = "insert into account values(" + number
+ ", to_date('"+date+"','mmddyyyy'))";

            try
            {
                  stmt.executeUpdate(insert);
                  stmt.executeUpdate(insertAccount);
            } catch(SQLException e)
            {
                  System.out.println("SQLException: " +
e.getMessage());
            }
      }
      // Query 6 - Insert New Job - Completed
      private void insertNewJob()
      {
            System.out.println("Enter the type of Job. 1 for Fit
```

```
Job, 2 for Paint Job, 3 for Cut Job.");
            int jobtype = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the job ID");
            int jobid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the assembly ID");
            int assid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the process ID");
            int proid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the Date Commenced.
mmddyyyy");
            String date = scan.nextLine();
            String info = "No info until completion";

            String t;
            String insertjob;
            while(true)
            {
                  if(jobtype == 1)
                  {
                        t = "fitjob";
                        System.out.println("Enter labor time.");
                        float labortime = scan.nextFloat();
                        scan.nextLine();

                        insertjob = "insert into " + t + " values("
+ jobid + "," + labortime + ")";
                        break;
                  }
                  else if(jobtype == 2)
                  {
                        t = "paintjob";
                        System.out.println("Enter the color");
                        String color = scan.nextLine();
                        System.out.println("Enter the volume");
                        float volume = scan.nextFloat();
                        scan.nextLine();
                        System.out.println("Enter the labor time");
                        float labortime = scan.nextFloat();
                        scan.nextLine();
```

```
                    insertjob = "insert into " + t + " values("
+ jobid + ",'" + color + "'," + volume + "," + labortime + ")";
                    break;
               }
               else if(jobtype == 3)
               {
                    t = "cutjob";
                    System.out.println("Enter the machine
used");
                    String machineused = scan.nextLine();
                    System.out.println("Enter the time used");
                    float timeused = scan.nextFloat();
                    scan.nextLine();
                    System.out.println("Enter the material
used");
                    String materialused = scan.nextLine();
                    System.out.println("Enter the labor time");
                    float labortime = scan.nextFloat();
                    scan.nextLine();


                    insertjob = "insert into " + t + " values("
+ jobid + ",'" + machineused + "'," + timeused + ",'" +
materialused + "'," + labortime + ")";
                    break;
               }

               else
               {
                    System.out.println("Enter the type of Job.
1 for Fit Job, 2 for Paint Job, 3 for Cut Job.");
                    jobtype = scan.nextInt();
               }
          }

          String insert = "insert into job values(" + jobid + ",
to_date('"+date+"','mmddyyyy'),"+null+",'" + info + "'," + proid
+ "," + assid + ")";

          try
          {
               stmt.executeUpdate(insert);
               stmt.executeUpdate(insertjob);
          } catch(SQLException e)
```

```
            {
                  System.err.println("SQLException: " +
e.getMessage());
            }
      }
      // Query 7 - Complete a Job - Completed
      private void completeJob()
      {
            System.out.println("Enter the ID of the job that has
been completed");
            int jobid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the date this job completed.
mmddyyyy");
            String date = scan.nextLine();
            System.out.println("Enter information relevant to this
job");
            String info = scan.nextLine();

            String update = "update job set date_completed =
to_date('"+date+"','mmddyyyy') where job.no = " + jobid;
            String updateinfo = "update job set information = '" +
info + "' where job.no = " + jobid;
            try
            {
                  stmt.executeUpdate(update);
                  stmt.executeUpdate(updateinfo);
            } catch(SQLException e)
            {
                  System.err.println("SQLException: " +
e.getMessage());
            }
      }
      // Query 8 - Insert a Transaction - Completed
      private void insertNewTransaction()
      {
            System.out.println("Enter the transaction number");
            int number = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the job number needed for
this transaction.");
            int jobid = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the sup-cost for this
```

```
transaction");
            float cost = scan.nextFloat();
            scan.nextLine();

            try
            {
                  // Find the process linked to this job
                  String proID = "select pro_no from job where
job.no = "+jobid;
                  ResultSet r = stmt.executeQuery(proID);
                  r.next();
                  int pro = r.getInt("pro_no");
                  // Find the record_costs for the process account
and update it
                  String updatePro = "update processaccount set
record_costs = record_costs+ "+cost+" where processaccount.no =
"+pro;
                  stmt.executeUpdate(updatePro);
                  // Find the assembly linked to this job
                  String assID = "select ass_id from job where
job.no = "+jobid;
                  ResultSet rs = stmt.executeQuery(assID);
                  rs.next();
                  int ass = rs.getInt("ass_id");
                  // Find the record_costs for the assembly account
and update it
                  String updateAss = "update assemblyaccount set
record_costs = record_costs+" +cost+" where assemblyaccount.no =
"+ass;
                  stmt.executeUpdate(updateAss);

                  // Find the department linked to this job
                  String depID = "select no from department where
department.no in (select dep_no from process where process.id =
'"+pro+"')";
                  ResultSet rsss = stmt.executeQuery(depID);
                  rsss.next();
                  int dep = rsss.getInt("no");
                  // Find the record_costs for the department
account and update it
                  String updateDep = "update departmentaccount set
record_costs = record_costs+"+cost+" where departmentaccount.no =
"+dep;
                  stmt.executeUpdate(updateDep);
```

```
                // Finally, Insert the new transaction
                String insert = "insert into transaction
values("+number+","+cost+","+jobid+")";
                stmt.executeUpdate(insert);

          } catch(SQLException e)
          {

     System.err.println("SQLException:"+e.getMessage());
          }

     }
     // Query 9 - Retrieve Cost of Assembly - Completed
     private void retrieveCostOfAssembly()
     {
          System.out.println("Enter the Assembly ID you want to
retrieve the cost of.");
          int id = scan.nextInt();
          scan.nextLine();

          String retrieve = "select record_costs from
assemblyaccount where assemblyaccount.no = "+id;

          try
          {
                ResultSet rs = stmt.executeQuery(retrieve);
                rs.next();
                String cost = rs.getString("record_costs");
                System.out.println("Cost for Assembly ID: "+id+"
is: "+cost);
          } catch(SQLException e)
          {
                System.err.println("SQLException: " +
e.getMessage());
          }
     }
     // Query 10 - Retrieve Labor Time of Assembly - Completed
     private void retrieveLaborTimeOfAssembly()
     {
```

```
            System.out.println("Enter the Assembly ID");
            int assid = scan.nextInt();
            scan.nextLine();

            try
            {
                  // Retrieve the job numbers that are associated
with this assembly
                  String assJobs = "select no from job where
job.pro_no in (select id from process where process.ass_id in "+
assid+")";

                  // All the jobs
                  ResultSet r = stmt.executeQuery(assJobs);
                  ArrayList<Integer> allJobs = new
ArrayList<Integer>();
                  while(r.next())
                  {
                        allJobs.add(r.getInt("no"));
                  }

                  // All of the cut jobs
                  String cut = "select no from cutjob";
                  ResultSet s = stmt.executeQuery(cut);
                  ArrayList<Integer> cutJobs = new
ArrayList<Integer>();
                  while(s.next())
                  {
                        cutJobs.add(s.getInt("no"));
                  }

                  // Loop to gather labor_cost of the cut jobs
                  float cutTotal = 0;
                  for(int i = 0; i < allJobs.size(); i++)
                  {
                        for(int j = 0; j < cutJobs.size(); j++)
                        {
                              if(allJobs.get(i) == cutJobs.get(j))
                              {
                                    String cutLabor = "select
labor_time from cutjob where cutjob.no = "+cutJobs.get(j);
                                    ResultSet rs =
stmt.executeQuery(cutLabor); rs.next();
                                    cutTotal +=
```

```
rs.getFloat("labor_time");
                              }
                        }
                  }

                  // All of the cut jobs
                  String paint = "select no from paintjob";
                  ResultSet f = stmt.executeQuery(paint);
                  ArrayList<Integer> paintJobs = new
ArrayList<Integer>();
                  while(f.next())
                  {
                        paintJobs.add(f.getInt("no"));
                  }

                  // Loop to gather labor_cost of the paint jobs
                  float paintTotal = 0;
                  for(int i = 0; i < allJobs.size(); i++)
                  {
                        for(int j = 0; j < paintJobs.size(); j++)
                        {
                              if(allJobs.get(i) ==
paintJobs.get(j))
                              {
                                    String paintLabor = "select
labor_time from paintjob where paintjob.no = "+paintJobs.get(j);
                                    ResultSet rs =
stmt.executeQuery(paintLabor); rs.next();
                                    paintTotal +=
rs.getFloat("labor_time");
                              }
                        }
                  }

                  // All of the fit jobs
                  String fit = "select no from fitjob";
                  ResultSet z = stmt.executeQuery(fit);
                  ArrayList<Integer> fitJobs = new
ArrayList<Integer>();
                  while(z.next())
                  {
                        fitJobs.add(z.getInt("no"));
                  }
```

```
                // Loop to gather labor_cost of the fit jobs
                float fitTotal = 0;
                for(int i = 0; i < allJobs.size(); i++)
                {
                     for(int j = 0; j < fitJobs.size(); j++)
                     {
                          if(allJobs.get(i) == fitJobs.get(j))
                          {
                               String fitLabor = "select
labor_time from fitjob where fitjob.no = "+fitJobs.get(j);
                               ResultSet rs =
stmt.executeQuery(fitLabor); rs.next();
                               fitTotal +=
rs.getFloat("labor_time");
                          }
                     }
                }

                float total = fitTotal + paintTotal + cutTotal;
                System.out.println("Cost is: " + total);
           } catch(SQLException e)
           {
                System.err.println("SQLException: "+
e.getMessage());
           }

     }
     // Query 11 - Retrieve Total Labor Time within a Department
- Completed
     private void retrieveTotalLaborTimeOfDepartment()
     {
          System.out.println("Enter a department");
          String dep = scan.nextLine();
          System.out.println("Enter the date.");
          String date = scan.nextLine();

          try
          {
                // Retrieve job numbers associated with this
department
                String depJobs = "select no from job where
job.pro_no in (select id from process where process.dep_no in
"+dep+") and job.date_completed =
to_date('"+date+"','mmddyyyy')";
```

```
                ResultSet r = stmt.executeQuery(depJobs);
                ArrayList<Integer> allJobs = new
ArrayList<Integer>();
                while(r.next())
                {
                      allJobs.add(r.getInt("no"));
                }

                // All of the cut jobs
                String cut = "select no from cutjob";
                ResultSet s = stmt.executeQuery(cut);
                ArrayList<Integer> cutJobs = new
ArrayList<Integer>();
                while(s.next())
                {
                      cutJobs.add(s.getInt("no"));
                }

                // Loop to gather labor_cost of the cut jobs
                float cutTotal = 0;
                for(int i = 0; i < allJobs.size(); i++)
                {
                      for(int j = 0; j < cutJobs.size(); j++)
                      {
                            if(allJobs.get(i) == cutJobs.get(j))
                            {
                                  String cutLabor = "select
labor_time from cutjob where cutjob.no = "+cutJobs.get(j);
                                  ResultSet rs =
stmt.executeQuery(cutLabor); rs.next();
                                  cutTotal +=
rs.getFloat("labor_time");
                            }
                      }
                }

                // All of the cut jobs
                String paint = "select no from paintjob";
                ResultSet f = stmt.executeQuery(paint);
                ArrayList<Integer> paintJobs = new
ArrayList<Integer>();
                while(f.next())
                {
                      paintJobs.add(f.getInt("no"));
```

```
                }

                // Loop to gather labor_cost of the paint jobs
                float paintTotal = 0;
                for(int i = 0; i < allJobs.size(); i++)
                {
                      for(int j = 0; j < paintJobs.size(); j++)
                      {
                            if(allJobs.get(i) ==
paintJobs.get(j))
                            {
                                  String paintLabor = "select
labor_time from paintjob where paintjob.no = "+paintJobs.get(j);
                                  ResultSet rs =
stmt.executeQuery(paintLabor); rs.next();
                                  paintTotal +=
rs.getFloat("labor_time");
                            }
                      }
                }

                // All of the fit jobs
                String fit = "select no from fitjob";
                ResultSet z = stmt.executeQuery(fit);
                ArrayList<Integer> fitJobs = new
ArrayList<Integer>();
                while(z.next())
                {
                      fitJobs.add(z.getInt("no"));
                }

                // Loop to gather labor_cost of the fit jobs
                float fitTotal = 0;
                for(int i = 0; i < allJobs.size(); i++)
                {
                      for(int j = 0; j < fitJobs.size(); j++)
                      {
                            if(allJobs.get(i) == fitJobs.get(j))
                            {
                                  String fitLabor = "select
labor_time from fitjob where fitjob.no = "+fitJobs.get(j);
                                  ResultSet rs =
stmt.executeQuery(fitLabor); rs.next();
                                  fitTotal +=
```

```
rs.getFloat("labor_time");
                              }
                        }
                  }

                  float total = fitTotal + paintTotal + cutTotal;
                  System.out.println("Total labor time: "+ total);


            } catch(SQLException e)
            {
                  System.err.println("SQLException: "+
e.getMessage());
            }
      }
      // Query 12 - Retrieve Process Assembly Passed - Completed
      private void retrieveProcessAssemblyPassed()
      {
            System.out.println("Enter the assembly ID");
            int id = scan.nextInt();

            String retrieve = "select * from process where
process.ass_id ="+id;

            try
            {
                  ResultSet r = stmt.executeQuery(retrieve);
                  while(r.next())
                  {
                        System.out.println("Process: "+
r.getInt("ID") + " Department: "+r.getInt("dep_no"));
                  }
            } catch(SQLException e)
            {
                  System.err.println("SQLException: "+
e.getMessage());
            }
      }
      // Query 13 - Retrieve Jobs - Completed
      private void retrieveJobs()
      {
            System.out.println("Enter the date. mmddyyyy");
            String date = scan.nextLine();
            System.out.println("Enter the department.");
```

```
            String dept = scan.nextLine();

            try
            {
                  String retrieve = "select * from job where
date_completed = to_date('"+date+"','mmddyyyy')";
                  ResultSet r = stmt.executeQuery(retrieve);
                  while(r.next())
                  {
                        System.out.println("Job: " + r.getInt("no")
+ " completed on: " +date+" in Department: "+dept);
                        System.out.println("Additonal information
about this job: ");
                        System.out.println("Type Info: "+
r.getString("information") +" Assembly: "+r.getInt("ass_id"));
                  }

            } catch(SQLException e)
            {
                  System.err.println("SQLException: "+
e.getMessage());
            }
      }
      // Query 14 - Retrieve Customers Red Assemblies  -
INCOMPLETE
      private void retrieveRedAssemblies()
      {
            // Get the painting method from the user to prompt a
query
            System.out.println("Enter the painting method");
            String method = scan.nextLine();

            // Two queries. First to get customer names where red
is the paint color
            // The second to get customer names where the paint
method is the input
            String red = "select name from customer where
customer.name in (select cust from assembly where assembly.id in
(select ass_id from job where job.no in (select no from paintjob
where paintjob.color = 'red')))";
            String cmethod = "select name from customer where
customer.name in (select cust from assembly where assembly.id in
(select ass_id from job where job.pro_no in (select id from
process where process.id in (select id from paintprocess where
```

```
paint_method = '"+method+"'))))";

            try
            {
                  // Grab the names of the red customers
                  ResultSet r = stmt.executeQuery(red);
                  ArrayList<String> custRed = new
ArrayList<String>();

                  while(r.next())
                  {
                        custRed.add(r.getString("name"));
                  }

                  // Grab the names of the 'method' customers
                  ResultSet s = stmt.executeQuery(cmethod);
                  ArrayList<String> custMethod = new
ArrayList<String>();
                  while(s.next())
                  {
                        custMethod.add(s.getString("name"));
                  }

                  // Now create a union on custRed and custMethod
and output
                  for(int i = 0; i < custRed.size();i++)
                  {
                        for(int j = 0; j < custMethod.size();j++)
                        {

      if(custRed.get(i).equals(custMethod.get(j)))
                              {
                                    System.out.println("List of
Customers ordered red paint with method: " + method);
                                    System.out.println("Customer: "
+ custMethod.get(j));
                              }
                        }
                  }

            } catch(SQLException e)
            {
                  System.err.println("SQLException: " +
e.getMessage());
```

```
            }
      }
      // Query 15 - Delete Cut Jobs - Completed
      private void deleteCutJobs()
      {
            System.out.println("Enter the first number in the
range. (exclusive)");
            int first = scan.nextInt();
            scan.nextLine();
            System.out.println("Enter the second number in the
range. (exclusive)");
            int second = scan.nextInt();
            scan.nextLine();

            String find = "select no from cutjob where cutjob.no
between "+ first +" and "+ second;

            try
            {
                  ResultSet r = stmt.executeQuery(find);
                  ArrayList<Integer> job = new
ArrayList<Integer>();
                  while(r.next())
                  {
                        job.add(r.getInt("no"));
                  }

                  for(int i = 0; i < job.size(); i++)
                  {
                        String deletecut = "delete from cutjob
where cutjob.no = "+ job.get(i);
                        String delete = "delete from job where
job.no = " + job.get(i);
                        stmt.executeUpdate(deletecut);
                        stmt.executeUpdate(delete);
                  }

            } catch(SQLException e)
            {
                  System.err.println("SQLException:" +
e.getMessage());
            }
      }
      // Query 16 - Change Paint Job Color - Completed
```

```
	private void changePaintJobColor()
	{
		System.out.println("Enter the Paint Job Number to be
modified.");
		int number = scan.nextInt();
		scan.nextLine();
		System.out.println("Enter the color you would like to
change this job to.");
		String color = scan.nextLine();

		String change = "update paintjob set color =
'"+color+"' where paintjob.no = "+number;

		try
		{
			stmt.executeUpdate(change);
		} catch(SQLException e)
		{
			System.err.println("SQLException:" +
e.getMessage());
		}
	}
	// Query 17 - Retrieve Average Cost of Accounts - Completed
	private void retrieveAverageCostOfAccounts()
	{
		String countAssembly = "select count(record_costs)
from assemblyaccount";
		String countDepartment = "select count(record_costs)
from departmentaccount";
		String countProcess = "select count(record_costs) from
processaccount";

		String sumAssembly = "select sum(record_costs) from
assemblyaccount";
		String sumDepartment = "select sum(record_costs) from
departmentaccount";
		String sumProcess = "select sum(record_costs) from
processaccount";

		try
		{
			// Grab the total number of rows for the
denominator
			ResultSet r = stmt.executeQuery(countAssembly);
```

```
                r.next();
                int blockOne = r.getInt("count(record_costs)");
                r = stmt.executeQuery(countDepartment);
                r.next();
                int blockTwo = r.getInt("count(record_costs)");
                r = stmt.executeQuery(countProcess);
                r.next();
                int blockThree =
r.getInt("count(record_costs)");

                // total count
                int count = blockOne + blockTwo + blockThree;

                // Grab the sum from each column set for the
numerator
                r = stmt.executeQuery(sumAssembly);
                r.next();
                int blockSix = r.getInt("sum(record_costs)");
                r = stmt.executeQuery(sumDepartment);
                r.next();
                int blockFour = r.getInt("sum(record_costs)");
                r = stmt.executeQuery(sumProcess);
                r.next();
                int blockFive = r.getInt("sum(record_costs)");

                // total sum
                int sum = blockFour + blockFive + blockSix;

                // print the average
                System.out.println("Average of all accounts is:
" + sum/count);

            } catch(SQLException e)
            {
                System.err.println("SQLException:" +
e.getMessage());
            }
      }
      // Query 18 - Import Customers - Completed
      private void importCustomers()
      {
            System.out.println("Enter the file name.");
            String f = scan.nextLine();
            String line = null;
```

```
            try
            {
                  FileReader fileReader = new FileReader(f);
                  BufferedReader bufferedReader = new
BufferedReader(fileReader);

                  while((line = bufferedReader.readLine()) !=
null)
                  {
                        try
                        {
                              stmt.executeUpdate(line);
                              System.out.println("Customer added");
                        } catch(SQLException e)
                        {
                              System.err.println("SQLException:" +
e.getMessage());
                        }
                  }

                  bufferedReader.close();
            } catch(FileNotFoundException e)
            {
                  e.printStackTrace();
            } catch(IOException e)
            {
                  e.printStackTrace();
            }
      }
      // Query 19 - Export Customers - Completed
      private void exportCustomers()
      {
            System.out.println("Enter the file name you would like
to export to.");
            String f = scan.nextLine();

            try
            {
                  PrintWriter writer = new PrintWriter(f, "UTF-
8");
                  ArrayList<String> customers = new
ArrayList<String>();
                  ResultSet r = stmt.executeQuery("select * from
```

```
customer");
                writer.write("Name"+ ", " + "Address" + "\n");
                while(r.next())
                {
                      writer.write(r.getString("Name") + ", " +
r.getString("Address") + "\n");
                }

                writer.close();
           } catch (FileNotFoundException e)
           {
                e.printStackTrace();
           } catch (UnsupportedEncodingException e)
           {
                e.printStackTrace();
           } catch (SQLException e)
           {
                System.err.println("SQLException: " +
e.getMessage());
           }
     }
}
```

DRIVER CLASS:

```
public class Driver
{
     Project project = new Project();

     public static void main(String[] args)
     {
          // All work done in the Project class
          Driver n = new Driver();
     }
}
```

TASK 6:

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

Input 1: to Enter a New Customer.
Input 2: to Enter a New Department.
Input 3: to Enter a New Assembly.
Input 4: to Enter a New Process ID and its Department.
Input 5: to Create a New Account.
Input 6: to Enter a New Job.
Input 7: to Enter Completion Date and Info of a Job.
Input 8: to Enter a Transaction number and it's Sup-cost.
Input 9: to Retrieve Cost Incurred on Assembly-ID.
Input 10: to Retrieve Labor Time Recorded on Assembly-ID.
Input 11: to Retrieve Labor Time within a Department for Jobs Completed in the Department given a Date.
Input 12: to Retrieve the Processes through which a given Assembly-ID has passed so far and Department Responsible for each Process.
Input 13: to Retrieve Jobs Completed given a Date in a given Department.
Input 14: to Retreive the Customers whose Assemblies are Painted RED using a given Painting Method.
Input 15: to Delete all Cut Jobs whose Job Number lies in a Range.
Input 16: to Change the Color of a given Paint Job.
Input 17: to Retrieve the Average Cost of all Accounts.
Input 18: Import
Input 19: Export
Input 20: Quit
```

```
1
Enter the new Customer's unique name
Alissa
Enter the new Customer's address
Troy


1
Enter the new Customer's unique name
Lori
Enter the new Customer's address
Mustang
|
 1
 Enter the new Customer's unique name
 Juan
 Enter the new Customer's address
 Lima
 |
 1
 Enter the new Customer's unique name
 Michelle
 Enter the new Customer's address
 Choctaw
 |
```

	NAME	ADDRESS
1	Koby	Mustang
2	Lori	Mustang
3	Juan	Lima
4	Michelle	Choctaw
5	Alissa	Troy

```
1
Enter the new Customer's unique name
Koby
Enter the new Customer's address
Mustang
```

```
2
Enter the Department ID
1
Enter any Data associated with this Department.
OU

2
Enter the Department ID
2
Enter any Data associated with this Department.
OSU
2
Enter the Department ID
3
Enter any Data associated with this Department.
Minn
2
Enter the Department ID
4
Enter any Data associated with this Department.
KSU
2
Enter the Department ID
5
Enter any Data associated with this Department.
Cal
```

	NO	DATA
1	1	OU
2	2	OSU
3	3	Minn
4	4	KSU
5	5	Cal

```
3
Enter the Assembly ID
1
Enter the date ordered. mmddyyyy
01011990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Koby
3
Enter the Assembly ID
3
Enter the date ordered. mmddyyyy
03031990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Lori
3
Enter the Assembly ID
4
Enter the date ordered. mmddyyyy
04041990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Lori
```

```
3
Enter the Assembly ID
5
Enter the date ordered. mmddyyyy
05051990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Michelle
3
Enter the Assembly ID
6
Enter the date ordered. mmddyyyy
06061990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Michelle
3
Enter the Assembly ID
7
Enter the date ordered. mmddyyyy
09092015
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Juan
3
Enter the Assembly ID
8
Enter the date ordered. mmddyyyy
10102015
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Juan
```

```
3
Enter the Assembly ID
9
Enter the date ordered. mmddyyyy
12122014
Enter additional Assembly details
Busy
Enter the customers ID that ordered the assembly
Alissa

3
Enter the Assembly ID
10
Enter the date ordered. mmddyyyy
11112014
Enter additional Assembly details
overnight
Enter the customers ID that ordered the assembly
Alissa

3
Enter the Assembly ID
2
Enter the date ordered. mmddyyyy
02021990
Enter additional Assembly details
none
Enter the customers ID that ordered the assembly
Koby
```

	ID	DATE_ORDERED	DETAILS	CUST
1	1	01-JAN-90	none	Koby
2	2	02-FEB-90	none	Koby
3	3	03-MAR-90	none	Lori
4	4	04-APR-90	none	Lori
5	5	05-MAY-90	none	Michelle
6	6	06-JUN-90	none	Michelle
7	7	09-SEP-15	none	Juan
8	8	10-OCT-15	none	Juan
9	9	12-DEC-14	Busy	Alissa
10	10	11-NOV-14	overnight	Alissa

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
1
Enter new Process ID
1
Enter Process data
fun
Enter the Assembly ID
1
Enter the Processes Department Number
1
Enter the type of Fit Process
fast


4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
1
Enter new Process ID
2
Enter Process data
none
Enter the Assembly ID
2
Enter the Processes Department Number
2
Enter the type of Fit Process
fast
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
1
Enter new Process ID
3
Enter Process data
none
Enter the Assembly ID
3
Enter the Processes Department Number
1
Enter the type of Fit Process
slow
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
2
Enter new Process ID
4
Enter Process data
interesting
Enter the Assembly ID
4
Enter the Processes Department Number
2
Enter the paint type
acrylic
Enter the type of paint method
brush
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
2
Enter new Process ID
5
Enter Process data
overnight
Enter the Assembly ID
5
Enter the Processes Department Number
2
Enter the paint type
lead
Enter the type of paint method
roller
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
2
Enter new Process ID
6
Enter Process data
none
Enter the Assembly ID
4
Enter the Processes Department Number
2
Enter the paint type
acrylic
Enter the type of paint method
roller
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
3
Enter new Process ID
7
Enter Process data
none
Enter the Assembly ID
5
Enter the Processes Department Number
1
Enter the cut type
fine
Enter the machine type
lathe


4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
3
Enter new Process ID
8
Enter Process data
careful
Enter the Assembly ID
3
Enter the Processes Department Number
2
Enter the cut type
straight
Enter the machine type
saw
```

```
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
3
Enter new Process ID
9
Enter Process data
1
Enter the Assembly ID
5
Enter the Processes Department Number
2
Enter the cut type
short
Enter the machine type
pencil
4
Enter the type of process. 1 for Fit, 2 for Paint, 3 for Cut
3
Enter new Process ID
10
Enter Process data
slow
Enter the Assembly ID
5
Enter the Processes Department Number
2
Enter the cut type
swift
Enter the machine type
guillotine
```

	ID	DATA	ASS_ID	DEP_NO	TYPE
1	1	fun	1	1	fit
2	2	none	2	2	fit
3	3	none	3	1	fit
4	4	interesting	4	2	paint
5	5	overnight	5	2	paint
6	6	none	4	2	paint
7	7	none	5	1	cut
8	8	careful	3	2	cut
9	9	1	5	2	cut
10	10	slow	5	2	cut

```
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
1
Enter the account number
1
Enter the date established. mmddyyyy
01011990
Enter the Assembly ID that this account is for.
8
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
1
Enter the account number
2
Enter the date established. mmddyyyy
12122015
Enter the Assembly ID that this account is for.
10

5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
1
Enter the account number
3
Enter the date established. mmddyyyy
06061970
Enter the Assembly ID that this account is for.
5
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
2
Enter the account number
4
Enter the date established. mmddyyyy
08081970
Enter this Department number that this account is for.
4
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
2
Enter the account number
5
Enter the date established. mmddyyyy
08081960
Enter this Department number that this account is for.
2
```

```
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
2
Enter the account number
6
Enter the date established. mmddyyyy
10101999
Enter this Department number that this account is for.
4

5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
2
Enter the account number
7
Enter the date established. mmddyyyy
10102000
Enter this Department number that this account is for.
2

5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
3
Enter the account number
8
Enter the date established. mmddyyyy
11112000
Enter this Process that this account is for.
3

5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
3
Enter the account number
9
Enter the date established. mmddyyyy
10102001
Enter this Process that this account is for.
5
5
Enter the type of Account. 1 for Assembly, 2 for Department, 3 for Process
3
Enter the account number
10
Enter the date established. mmddyyyy
12122005
Enter this Process that this account is for.
4
```

	NO	DATE_ESTABLISHED
1	1	01-JAN-90
2	2	12-DEC-15
3	3	06-JUN-70
4	4	08-AUG-70
5	5	08-AUG-60
6	6	10-OCT-99
7	7	10-OCT-00
8	8	11-NOV-00
9	9	10-OCT-01
10	10	12-DEC-05

	NO	RECORD_COSTS	ASS_ID
1	1	0	8
2	2	0	10
3	3	0	5

	NO	RECORD_COSTS	DEPT_ID
1	4	0	4
2	5	0	2
3	6	0	4
4	7	0	2

	NO	RECORD_COSTS	PRO_ID
1	8	0	3
2	9	0	5
3	10	0	4

```
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
1
Enter the assembly ID
5
Enter the process ID
4
Enter the Date Commenced. mmddyyyy
01011990
Enter labor time.
5.5
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
2
Enter the assembly ID
5
Enter the process ID
4
Enter the Date Commenced. mmddyyyy
07071999
Enter labor time.
7.2
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
3
Enter the assembly ID
5
Enter the process ID
5
Enter the Date Commenced. mmddyyyy
09011990
Enter labor time.
17.6
```

```
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
4
Enter the assembly ID
2
Enter the process ID
3
Enter the Date Commenced. mmddyyyy
06061997
Enter labor time.
9
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
5
Enter the assembly ID
6
Enter the process ID
2
Enter the Date Commenced. mmddyyyy
01011989
Enter labor time.
4
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
2
Enter the job ID
6
Enter the assembly ID
8
Enter the process ID
2
Enter the Date Commenced. mmddyyyy
01012001
Enter the color
red
Enter the volume
5
Enter the labor time
7
```

```
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
3
Enter the job ID
7
Enter the assembly ID
6
Enter the process ID
6
Enter the Date Commenced. mmddyyyy
02021976
Enter the machine used
lathe
Enter the time used
5
Enter the material used
wood
Enter the labor time
2
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
2
Enter the job ID
8
Enter the assembly ID
5
Enter the process ID
2
Enter the Date Commenced. mmddyyyy
050516
Enter the color
red
Enter the volume
9
Enter the labor time
10
```

```
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
2
Enter the job ID
9
Enter the assembly ID
5
Enter the process ID
1
Enter the Date Commenced. mmddyyyy
06061977
Enter the color
black
Enter the volume
2
Enter the labor time
10
 6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
3
Enter the job ID
10
Enter the assembly ID
8
Enter the process ID
9
Enter the Date Commenced. mmddyyyy
01011966
Enter the machine used
saw
Enter the time used
5
Enter the material used
steel
Enter the labor time
6
```

	NO	DATE_COMMENCED	DATE_COMPLETED	INFORMATION	PRO_NO	ASS_ID
1	1	01-JAN-90	(null)	No info until completion	4	5
2	2	07-JUL-99	(null)	No info until completion	4	5
3	3	01-SEP-90	(null)	No info until completion	5	5
4	4	06-JUN-97	(null)	No info until completion	3	2
5	5	01-JAN-89	(null)	No info until completion	2	6
6	6	01-JAN-01	(null)	No info until completion	2	8
7	7	02-FEB-76	(null)	No info until completion	6	6
8	8	05-MAY-16	(null)	No info until completion	2	5
9	9	06-JUN-77	(null)	No info until completion	1	5
10	10	01-JAN-66	(null)	No info until completion	9	8

	NO	MACHINE_USED	TIME_USED	MATERIAL_USED	LABOR_TIME
1	7	lathe	5	wood	2
2	10	saw	5	steel	6

	NO	LABOR_TIME
1	1	5.5
2	2	7.2
3	3	17.6
4	4	9
5	5	4

	NO	COLOR	VOLUME	LABOR_TIME
1	6	red	5	7
2	8	red	9	10
3	9	black	2	10

```
7
Enter the ID of the job that has been completed
1
Enter the date this job completed. mmddyyyy
12122010
Enter information relevant to this job
success
7
Enter the ID of the job that has been completed
2
Enter the date this job completed. mmddyyyy
08082014
Enter information relevant to this job
none
7
Enter the ID of the job that has been completed
3
Enter the date this job completed. mmddyyyy
12252014
Enter information relevant to this job
xmas
7
Enter the ID of the job that has been completed
4
Enter the date this job completed. mmddyyyy
01012015
Enter information relevant to this job
late
7
Enter the ID of the job that has been completed
02022015
Enter the date this job completed. mmddyyyy
02022015
Enter information relevant to this job
old
```

```
7
Enter the ID of the job that has been completed
6
Enter the date this job completed. mmddyyyy
02021999
Enter information relevant to this job
old
7
Enter the ID of the job that has been completed
7
Enter the date this job completed. mmddyyyy
07072009
Enter information relevant to this job
success
7
Enter the ID of the job that has been completed
8
Enter the date this job completed. mmddyyyy
08082010
Enter information relevant to this job
late
7
Enter the ID of the job that has been completed
9
Enter the date this job completed. mmddyyyy
10102011
Enter information relevant to this job
late
7
Enter the ID of the job that has been completed
10
Enter the date this job completed. mmddyyyy
11232015
Enter information relevant to this job
ontime
```

```
7
Enter the ID of the job that has been completed
5
Enter the date this job completed. mmddyyyy
02022014
Enter information relevant to this job
very late
```

	NO	DATE_COMMENCED	DATE_COMPLETED	INFORMATION	PRO_NO	ASS_ID
1	1	01-JAN-90	12-DEC-10	success	4	5
2	2	07-JUL-99	08-AUG-14	none	4	5
3	3	01-SEP-90	25-DEC-14	xmas	5	5
4	4	06-JUN-97	01-JAN-15	late	3	2
5	5	01-JAN-89	02-FEB-14	very late	2	6
6	6	01-JAN-01	02-FEB-99	old	2	8
7	7	02-FEB-76	07-JUL-09	success	6	6
8	8	05-MAY-16	08-AUG-10	late	2	5
9	9	06-JUN-77	10-OCT-11	late	1	5
10	10	01-JAN-66	23-NOV-15	ontime	9	8

```
8
Enter the transaction number
1
Enter the job number needed for this transaction.
1
Enter the sup-cost for this transaction
10
 8
 Enter the transaction number
 5
 Enter the job number needed for this transaction.
 9
 Enter the sup-cost for this transaction
 11
 8
Enter the transaction number
4
Enter the job number needed for this transaction.
8
Enter the sup-cost for this transaction
5.5
 8
 Enter the transaction number
 3
 Enter the job number needed for this transaction.
 1
 Enter the sup-cost for this transaction
 19
8
Enter the transaction number
2
Enter the job number needed for this transaction.
2
Enter the sup-cost for this transaction
12.2
```

```
8
Enter the transaction number
6
Enter the job number needed for this transaction.
9
Enter the sup-cost for this transaction
1
8
Enter the transaction number
7
Enter the job number needed for this transaction.
6
Enter the sup-cost for this transaction
1
8
Enter the transaction number
8
Enter the job number needed for this transaction.
8
Enter the sup-cost for this transaction
0
8
Enter the transaction number
9
Enter the job number needed for this transaction.
5
Enter the sup-cost for this transaction
0
8
Enter the transaction number
10
Enter the job number needed for this transaction.
1
Enter the sup-cost for this transaction
24
```

	NO	SUP_COST	JOB_NO
1	4	5.5	8
2	5	11	9
3	6	1	9
4	1	10	1
5	2	12.2	2
6	3	19	1
7	7	1	6
8	8	0	8
9	9	0	5
10	10	24	1

		RECORD_COSTS	PRO_ID
1	Filter...	2	3
2	9	0	5
3	10	0	4

	NO	RECORD_COSTS	DEPT_ID
1	4	74	4
2	5	0	2
3	6	0	4
4	7	0	2

	NO	RECORD_COSTS	ASS_ID
1	1	79.15	8
2	2	77	10
3	3	0	5

```
9
Enter the Assembly ID you want to retrieve the cost of.
1
Cost for Assembly ID: 1 is: 79.15


9
Enter the Assembly ID you want to retrieve the cost of.
3
Cost for Assembly ID: 3 is: 0


9
Enter the Assembly ID you want to retrieve the cost of.
2
Cost for Assembly ID: 2 is: 77
```

```
10
Enter the Assembly ID
3
Cost is: 9.0
10
Enter the Assembly ID
2
Cost is: 21.0
10
Enter the Assembly ID
1
Cost is: 10.0
```

```
11
Enter a department
9
Enter the date.
06072009
Total labor time: 0.0
```

```
11
Enter a department
1
Enter the date.
01012015
Total labor time: 9.0
```

```
11
Enter a department
2
Enter the date.
02021999
Total labor time: 7.0
```

```
12
Enter the assembly ID
1
Process: 1 Department: 1
```

```
12
Enter the assembly ID
2
Process: 2 Department: 2
```

```
12
Enter the assembly ID
5
Process: 5 Department: 2
Process: 7 Department: 1
Process: 9 Department: 2
Process: 10 Department: 2
```

```
13
Enter the date. mmddyyyy
01012015
Enter the department.
2
Job: 4 completed on: 01012015 in Department: 2
Additonal information about this job:
Type Info: late Assembly: 2


13
Enter the date. mmddyyyy
08082014
Enter the department.
1
Job: 2 completed on: 08082014 in Department: 1
Additonal information about this job:
Type Info: none Assembly: 5


13
Enter the date. mmddyyyy
12122010
Enter the department.
1
Job: 1 completed on: 12122010 in Department: 1
Additonal information about this job:
Type Info: success Assembly: 5
```

```
14
Enter the painting method
finger
List of Customers ordered red paint with method: finger
Customer: Miguel
```

```
14
Enter the painting method
brush
List of Customers ordered red paint with method: brush
Customer: Michelle
```

```
14
Enter the painting method
roller
List of Customers ordered red paint with method: roller
Customer: Michelle
```

	NO	MACHINE_USED	TIME_USED	MATERIAL_...	LABOR_TIME
1	7	lathe	5	wood	2
2	10	saw	5	steel	6

```
15
Enter the first number in the range. (exclusive)
0
Enter the second number in the range. (exclusive)
6
15
Enter the first number in the range. (exclusive)
5
Enter the second number in the range. (exclusive)
8
```

```
15
Enter the first number in the range. (exclusive)
0
Enter the second number in the range. (exclusive)
100
```

NO	MACHINE...	TIME_USED	MATERIA...	LABOR_TI...

```
16
Enter the Paint Job Number to be modified.
6
Enter the color you would like to change this job to.
blue

16
Enter the Paint Job Number to be modified.
9
Enter the color you would like to change this job to.
red

16
Enter the Paint Job Number to be modified.
8
Enter the color you would like to change this job to.
purple
```

	NO	COLOR	VOLUME	LABOR_TIME
1	6	blue	5	7
2	8	purple	9	10
3	9	red	2	10

```
17
Average of all accounts is: 23
```

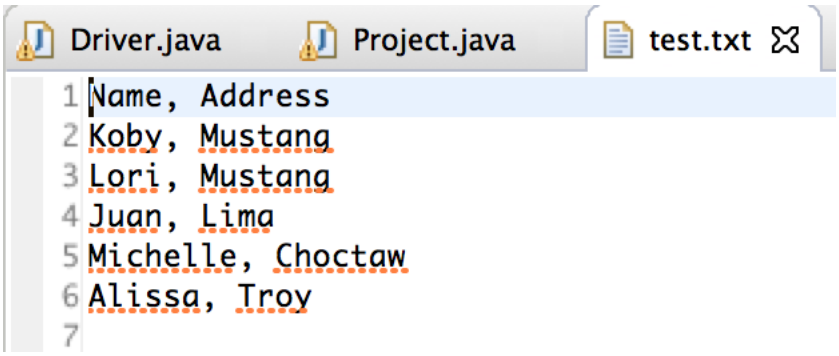
Driver.java Project.java test.txt

```
1 insert into customer values('ben', 'minneapolis')
```

```
18
Enter the file name.
test.txt
```

	NAME	ADDRESS
1	Koby	Mustang
2	Lori	Mustang
3	Juan	Lima
4	Michelle	Choctaw
5	Alissa	Troy
6	ben	minneapolis

```
19
Enter the file name you would like to export to.
text.txt
```




```
20
See you next time!
```

```
6
Enter the type of Job. 1 for Fit Job, 2 for Paint Job, 3 for Cut Job.
1
Enter the job ID
1
Enter the assembly ID
1
Enter the process ID
1
Enter the Date Commenced. mmddyyyy
1
Enter labor time.
1
SQLException: ORA-01840: input value not long enough for date format
```

```
1
Enter the new Customer's unique name
Koby
Enter the new Customer's address
Mustang
SQLException:ORA-00001: unique constraint (PASC6415.SYS_C0069205) violated
```

```
3
Enter the Assembly ID
100
Enter the date ordered. mmddyyyy
01012015
Enter additional Assembly details
null
Enter the customers ID that ordered the assembly
-1
SQLException:ORA-02291: integrity constraint (PASC6415.SYS_C0069207) violated - parent key not found
```