School: Computer Science
Institution: University of Windsor
Term: Winter 2021
Course: Comp-3150 (03-60-315-1) : Database Management Systems
Instructor: Dr. C. I. Ezeife
Assignment #3 Solution: Total: 50 marks
Handed Out:Thurs. Feb. 25, 2021; Due Thurs Mar. 18, 2021

Objective of Assignment: To test on knowledge and use of relational database query languages SQL and relational algebra for implementing relational databases.
Scope: Assignment covers materials from Chapters 6, 7 and 8 of book discussed in class.
Electronic Assignment Submission: Done through http://blackboard.uwindsor.ca
Marking Sheme : The mark for each of the questions is indicated beside each question.
Academic Integrity Statement: Remember to submit only work that is yours and include the following confidentiality agreement and statement at the beginning of your assignment.

CONFIDENTIALITY AGREEMENT & STATEMENT OF HONESTY
I confirm that I will keep the content of this assignment/examination confidential.
I confirm that I have not received any unauthorized assistance in preparing for or doing this assignment/examination. I confirm knowing that a mark of 0 may be assigned for copied work.

_____ _____
Student Signature Student Name (please print)

_____ _____
Student I.D. Number Date

Marking Scheme : The mark for each question and sub question is shown with the question below. Place your solutions in tables where possible.

For office Use only

Question	Mark
1	/15
2	/10
3	/5
4	/5
5	/5
6	/10
Total	/50

CHAPTER 6 and 7: Basic SQL with all DDL an some DML

1. Given a database state of the 'Sales rep sells products in sales areas' database shown in Figure 1.1, with schema shown in Figure 1.2.

(Total for que 1 is 15 marks)

Fig 1.1: An Example Database State of Sales rep sells products in sales areas Database

Rep

RID	RNAME	RAGE	SALARY
11111	Mary Peters	42	150000
22222	Steve Markel	25	221000
33333	Melanie Good	61	188000
44444	John Doe	36	85000

Area

ARID	ANAME	CITY	BUDGET
WDT	Down Town	Windsor	500000.99
SW	South Windsor	Windsor	950000.00
TN	Tecumseh North	Tecumseh	420000.50

Product

PRID	PNAME	PDESC	PRICE
1	pots	cookware	101.00
2	towels	bath supply	45.50
3	apple basket	fruit	30.30
4	potato bag	food	5.50

RepSellsArea

Rid	Arid	hours	revenue
11111	TN	12	20000.00
11111	SW	10	1500.00
33333	SW	33	40000.00
44444	WDT	28	37200.00

RepSellsProduct

Rid	Prid	hours	revenue
11111	1	6	2000.00
11111	2	10	4500.00
33333	2	20	9000.00
44444	4	12	3300.00

AreaSellsProduct

ARid	Prid	hours	revenue
WDT	1	18	2000.00
WDT	2	12	6600.00
SW	2	20	9000.00
TN	4	12	3300.00

Fig 1.2: Schema of the 'Sales rep sells products in sales areas' database of Figure 1.1

Rep (Rid, Rname, Rage, salary)
Area (Arid, Aname, city, budget)
Product (Prid, Pname, Pdesc, price)
RepSellsArea (Rid, Arid, hours, revenue)
RepSellsProduct (Rid, Prid, hours, revenue)
AreaSellsProduct (Arid, Prid, hours, revenue)

i. List all the referential integrity constraints that should hold on the database schema? (2.5 marks)

ii. Write appropriate SQL DDL statements to define the database with the integrity constraints and store in a text file called userid_salesrep_schema.sql. Attach this file or also show it in your script file of (v) using the unix command more file.sql before or after running sqlplus. Do the same for the files in (iii) and (iv). (2.5 marks)

iii. To insert the data in the database tables, also write appropriate SQL DML instructions in a text file called userid_salesrep_data.sql. (2.5 marks)

iv. To remove any inserted data and destroy all created tables in the salesrep database, write appropriate SQL DML and DDL statements in a text file called userid_salesrep_droptable.sql to first delete all data in the tables and then drop the tables. (2.5 marks)

v. Using Oracle Sqlplus, implement this database design by creating all the tables with the integrity constraints using the SQL DDL you defined in (ii) above. You can create all these SQL DDL for creating the 6 tables by running your .sql file at the SQL prompt with the command:

@userid_salesrep_schema.sql. After creating your tables successfully, you load your data with the .sql file you created in (iii) above by running @userid_salesrep_data.sql. If there are errors and you need to correct them, you might want to delete the tuples and drop the tables first using the .sql file

you created in (iv) above as with @userid_salesrep_droptable.sql before re-creating the schema and re-loading the data.Then, using a script file, show the contents of all 6 tables in the database by doing select * from each of the tables and saving on script file called username_assn3que1.txt. You can do this using the following sequence of Unix/Linux commands after you have created the database and inserted data. (5 marks)

(Note: remember to create the entity tables with primary keys before the relationship tables that reference them through foreign key attributes. When inserting data, do the same. If you need to delete the data and tables at any time, go in the reverse order (that is, delete the tuples that reference a primary key attribute tuple in another table, before deleting the parent primary keyed tuple))

```
>script   username_assn3que1.txt
>SQL  <username>
>password
SOL>  select *  from Rep;   //repeat this instruction for each table
SQL> exit                        //to exit sqlplus
exit                               // to exit and create script file
```

**Now attach the saved log of your Unix session that is in username_assn3que1.txt which includes all these six .sql script files in questions (ii), (iii) an (iv) (that you already executed in Sqlplus) in your Unix script file. From your Unix script file, we are able to see your interaction with the Sqlplus and SQL that showed how you created and queried your database. Note that attachment of those .sql files without running the in Sqlplus does not constitute a solution as you have not built your database.

Solution 1 (i) (mark: 2.5)

We will write a referential integrity constraint as R.A --> S (or R.(X) --> T) whenever attribute A (or the set of attributes X) of relation R form a foreign key that references the primary key of relation S (or T).

REPSELLSAREA.RID --> REP
REPSELLSAREA.ARID --> AREA
REPSELLSPROD.RID --> REP
REPSELLSPROD.PRID --> PRODUCT
AREASELLSPROD.ARID --> AREA
AREASELLSPROD.PRID --> PRODUCT

Solution 1 (ii): (mark: 2.5)

One possible set of CREATE TABLE statements to define the database is given below given in the file userid_salesrep_schema.sql is:

```
CREATE TABLE REP
(RID  VARCHAR2(5) NOT NULL,
 Rname VARCHAR2(20),
 RAGE NUMBER(3),
Salary NUMBER(10, 2),
 PRIMARY KEY(RID));

CREATE TABLE AREA
```

```
(ARID VARCHAR2(3) NOT NULL,
 Aname VARCHAR2(20),
 City VARCHAR2(15),
 Budget  NUMBER(10, 2),
 PRIMARY KEY(ARID));

CREATE TABLE PRODUCT
(PRID VARCHAR2(3) NOT NULL,
 Pname VARCHAR2(20),
 PDESC VARCHAR2(15),
 Price NUMBER(10, 2),
 PRIMARY KEY(PRID));

CREATE TABLE REPSELLSAREA
(RID VARCHAR2(5) NOT NULL,
ARID VARCHAR2(3) NOT NULL,
Hours NUMBER(3) ,
Revenue  NUMBER(10, 2),
PRIMARY KEY(RID, ARID),
FOREIGN KEY (RID) REFERENCES REP(RID),
FOREIGN KEY (ARID) REFERENCES AREA(ARID));

CREATE TABLE REPSELLSPRODUCT
(RID VARCHAR2(5) NOT NULL,
PRID VARCHAR2(3) NOT NULL,
Hours NUMBER(3) ,
Revenue  NUMBER(10, 2),
PRIMARY KEY(RID, PRID),
FOREIGN KEY (RID) REFERENCES REP(RID),
FOREIGN KEY (PRID) REFERENCES PRODUCT(PRID));


CREATE TABLE AREASELLSPRODUCT
(ARID VARCHAR2(3) NOT NULL,
PRID VARCHAR2(3) NOT NULL,
Hours NUMBER(3) ,
Revenue  NUMBER(10, 2),
PRIMARY KEY(ARID, PRID),
FOREIGN KEY (ARID) REFERENCES AREA(ARID),
FOREIGN KEY (PRID) REFERENCES PRODUCT(PRID));

COMMIT;
```

Solution 1 (iii): (mark: 2.5)

One possible set of INSERT INTO TABLE statements to define the database is given below given in the file userid_salesrep_data.sql is:

```
-- Now inserting data into the table REP
INSERT INTO REP
VALUES ('11111', 'Mary Peters', 42, 150000.00);

COMMIT;

INSERT INTO REP
```

```
VALUES ('22222', 'Steve Markel', 25, 221000.00);

COMMIT;

INSERT INTO REP
VALUES ('33333', 'Melanie Good', 61, 188000.00);

COMMIT;

INSERT INTO REP
VALUES ('44444', 'John Doe', 36, 85000.00);

COMMIT;

-- Now inserting data into the table Area

INSERT INTO AREA
VALUES ('WDT', 'Down Town', 'Windsor' , 500000.99);

COMMIT;

INSERT INTO AREA
VALUES ('SW', 'South West', 'Windsor' , 950000.00);

COMMIT;

INSERT INTO AREA
VALUES ('TN', 'Tecumseh North', 'Tecumseh' , 420000.50);

COMMIT;

-- Now inserting data into the table Product

INSERT INTO PRODUCT
VALUES ('1', 'pots', 'cookware' , 101.00);

COMMIT;

INSERT INTO PRODUCT
VALUES ('2', 'towels', 'bath supply' , 45.50);

COMMIT;

INSERT INTO PRODUCT
VALUES ('3', 'apple basket', 'fruit' , 30.30);

COMMIT;

INSERT INTO PRODUCT
VALUES ('4', 'potato bag', 'food' , 5.50);

COMMIT;
```

```
-- Data for REPSELLSAREA starts next

INSERT INTO REPSELLSAREA
VALUES ('11111', 'TN', 12 , 20000.00);

COMMIT;

INSERT INTO REPSELLSAREA
VALUES ('11111', 'SW', 10 , 1500.00);

COMMIT;

INSERT INTO REPSELLSAREA
VALUES ('33333', 'SW', 33, 40000.00);

COMMIT;

INSERT INTO REPSELLSAREA
VALUES ('44444', 'WDT', 28, 37200.00);

COMMIT;


-- Data for RepSellsProd starts next

INSERT INTO REPSELLSPRODUCT
VALUES ('11111', '1', 6, 2000.00);

COMMIT;

INSERT INTO REPSELLSPRODUCT
VALUES ('11111', '2', 10, 4500.00);

COMMIT;

INSERT INTO REPSELLSPRODUCT
VALUES ('33333', '2', 20, 9000.00);

COMMIT;

INSERT INTO REPSELLSPRODUCT
VALUES ('44444', '4', 12, 3300.00);

COMMIT;

-- Data for AreaSellsProd starts next

INSERT INTO AREASELLSPRODUCT
VALUES ('WDT', '1', 18, 2000.00);

COMMIT;

INSERT INTO AREASELLSPRODUCT
```

```
VALUES ('WDT', '2', 12, 6600.00);

COMMIT;

INSERT INTO AREASELLSPRODUCT
VALUES ('SW', '2', 20, 9900.00);

COMMIT;

INSERT INTO AREASELLSPRODUCT
VALUES ('TN', '4', 12, 3300.00);

COMMIT;


COMMIT;
```

Solution 1 (iv): (mark: 2.5)

One possible set of DELETE FROM TABLE statements and DROP TABLE statements to delete data from the database and drop the tables is given below given in the file userid_salesrep_droptable.sql is:

```
delete from AreaSellsProduct;

commit;

delete from  RepSellsProduct;

commit;

delete from RepSellsArea;

commit;
delete from Product;
commit;

delete from Area;
commit;

delete from Rep;
commit;

drop table AreaSellsProduct cascade constraints;
drop table  RepSellsProduct cascade constraints;
drop table RepSellsArea cascade constraints;
drop table  Product cascade constraints;
drop table Area cascade  constraints;
drop table Rep cascade  constraints;

commit;
```

1 (v). (5 marks) for the script file showing correct interaction with Oracle Sqlplus creating and loading data in these 6 tables.

2. Specify the following 5 queries in SQL on the Salesrep database schema of Figure 1.1.
(Total for que 2 is 10 marks)

i. List all your 5 queries in the table below first in SQL. (5 marks)
ii. Implement the answering of your 5 queries in 2(i) using Sqlplus and the same database you created in question 1, providing your execution and answers to these questions in a script file called username_assn3que2.txt. (5 marks)

(a) Retrieve the names, hours and revenue of products sold by sales rep Melanie Good.
(b) Retrieve the city and budget of all areas that have more than one product sold in it.
(c) For each sales rep, retrieve their name, age, salary, and number of areas they sold in.
(d) Retrieve the product name, sales rep's name, and sales area name of each sold product.
(e) Retrieve the product name, product description and the sales rep's name of all products that are not sold by more than one sales rep. (may use nested query here)

Solution 2 (i): Queries(5 marks) and 2(ii) Results (5 marks)

a) Retrieve the names, hours and revenue of products sold by sales rep Melanie Good.

```
SELECT P.PNAME, RP.HOURS, RP.REVENUE
FROM REPSELLSPRODUCT RP, REP R, PRODUCT P
WHERE  RP.RID = R.RID AND RP.PRID = P.PRID AND R.RNAME = 'Melanie Good';
```

```
PNAME                     HOURS    REVENUE
------------------- ---------- ----------
towels                       20       9000
```

(b) Retrieve the city and budget of all areas that have more than one product sold in it.

```
SELECT A.CITY, A.BUDGET
FROM AREA A,  AREASELLSPRODUCT AP, PRODUCT P
WHERE A.ARID = AP.ARID AND P.PRID = AP.PRID
GROUP BY A.CITY, A.BUDGET
HAVING COUNT(*) > 1;
```

```
CITY                BUDGET
--------------- ----------
Windsor          500000.99
```

(c) For each sales rep, retrieve their name, age, salary, and number of areas they sold in.

```
SELECT R.RNAME, R.RAGE, R.SALARY,  COUNT(*)
FROM REP R, REPSELLSAREA RA
WHERE R.RID = RA.RID
GROUP BY R.RNAME, R.RAGE, R.SALARY;
```

```
RNAME                      RAGE     SALARY   COUNT(*)
-------------------- ---------- ---------- ----------
John Doe                     36      85000          1
Mary Peters                  42     150000          2
Melanie Good                 61     188000          1
```

(d) Retrieve the product name, sales rep's name, and sales area name of each sold product.

```
SELECT P.PNAME, R.RNAME, A.ANAME
FROM PRODUCT P, REP R, AREA A, RepSellsProduct RP, AreaSellsProduct
AP
WHERE P.PRID = AP.PRID AND P.PRID = RP.PRID
AND R.RID = RP.RID AND A.ARID=AP.ARID;
```

```
PNAME                RNAME                ANAME
-------------------- -------------------- --------------------
towels               Mary Peters          Down Town
towels               Mary Peters          South West
pots                 Mary Peters          Down Town
towels               Melanie Good         Down Town
towels               Melanie Good         South West
```

```
potato bag            John Doe              Tecumseh North
```

(e) Retrieve the product name, product description and the sales rep's name of all products that are not sold by more than one sales rep.

```
select P.pname, P.pDesc, R.Rname
from Rep R, Product p, RepSellsProduct RP
Where RP.prid = p.prid and RP.Rid = R.Rid
and p.prid IN ( SELECT prid
                FROM RepSellsProduct RP
                Group by prid
                Having count(*) <= 1);
```

```
PNAME                PDESC           RNAME
-------------------- --------------- --------------------
pots                 cookware        Mary Peters
potato bag           food            John Doe
```

2 (ii). (5 marks) distributed as: 2.5 marks for the script file showing correct interaction with Oracle Sqlplus posing these queries; and 2.5 marks for correctly posing the queries and retrieving correct results.

3. Write four SQL update statements to do the following updates on the database schema shown in Figure 1.2. Show the affected tables after update through script file in sqlplus and in a script file created as before and named username_assn3que3.txt. (5 marks)

(Total for que 3 is 5 marks)

(a) Insert a new product <'5', 'banana bunch', 'fruit', 2.30> in the database.

(b) Change the age of rep ''John Doe' to 50.

(c) Insert a new area < 'ES', ' Essex South', 'Essex', 120000.00>.

(d) Delete all sales made in the area of 'TN'.

Solution 3 (i): (5 marks)

(a) Insert a new product <'5', 'banana bunch', 'fruit', 2.30> in the database.

```
INSERT INTO PRODUCT VALUES('5', 'banana bunch, 'fruit', 2.30);
```

(b) Change the age of rep ''John Doe' to 50.

```
UPDATE REP
SET RAGE = 50
WHERE RName='John Doe';
```

(c) Insert a new area < 'ES', ' Essex South', 'Essex', 120000.00>.

```
INSERT INTO AREA VALUES ('ES', 'Essex South', 'Essex', 120000.00);
```

(d) Delete all sales made in the area of 'SW'.

```
DELETE FROM AreaSellsProduct AP
WHERE ARID='TN' ;
```

CHAPTER 7: Some More SQL: Complex Queries, Triggers, Views, and Schema Modification

4. (i) Write the following 2 queries in SQL on the database schema of Figure 1.2 using EXISTS or NOT EXISTS as appropriate. (2.5 marks)
(ii) Implement the answering of your 2 queries in 4(i) using Sqlplus and the same database you created in question 1 and modified in earlier question with updates, deletes and inserts, providing your execution and answers to this question in a script file called username_assn3que4.
(2.5 marks)
(Total for que 4 is 5 marks)

(a) Retrieve the product name and price of all products with 2 sales from a rep.
(b) Retrieve the product name and price of all products that do not have 2 sales from a rep.

Solution 4 (i): (2.5 marks)

(a)) Retrieve the product name and price of all products with 2 sales from a rep.

```
select P.Pname, p.price
from Product P
where exists(
      select RP.Prid, RP.Rid from RepSellsProduct RP
      where RP.Prid=P.Prid group by Prid, RP.Rid
      having count(*)=2);
```

(b) Retrieve the product name and price of all products that do not have 2 sales from a rep.

```
select P.Pname, p.price
from Product P
where NOT exists(
      select RP.Prid, RP.Rid from RepSellsProduct RP
      where RP.Prid=P.Prid group by Prid, RP.Rid
      having count(*)=2);
```

4 (ii). (2.5 marks) distributed as: 0.5 marks for the script file showing correct interaction with Oracle Sqlplus posing these queries; and 2 marks for the correctly posing the queries and retrieving correct results.

5. In SQL, specify the following 3 queries on the COMPANY database of Figures 5.5 and 5.6 using the concept of nested queries and the concepts described in chapter 7. (Total for que 5 is 5 marks)

a. Retrieve the names of all projects that have the minimum number of hours worked on them per week (for example, the minimum hours worked per week is 5 and any project with this number should be retrieved).
b. Retrieve the hours of all projects whose number of hours worked on per week are less than the average number of hours worked on all projects. Also, print the average hours with each hours.
c. Retrieve the names of projects that are worked on at least 15 hours more than the project with the least number of hours worked on it per week.

Figure 5.5 Schema diagram for the COMPANY relational database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Solution 5: (5 marks)

a) SELECT PNAME FROM PROJECT WHERE PNUMBER IN
(SELECT PNO FROM WORKS_ON WHERE HOURS IN
(SELECT MIN(HOURS) FROM WORKS_ON));

b) SELECT HOURS FROM WORKS_ON WHERE HOURS <=ANY
(SELECT AVG(HOURS) FROM WORKS_ON);

c) SELECT PNO, HOURS FROM WORKS_ON WHERE HOURS >= 15 +
(SELECT MIN(HOURS) FROM WORKS_ON);

CHAPTER 8: THE RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

6. Specify the following 5 queries on the COMPANY relational database schema shown in Figure 5.5, using the relational operators discussed in chapter 8. Also show the result of each query as it would apply to the database state of Figure 5.6. (Total for que 6 is 10 marks)

Some symbols for solving queries you may copy and reuse are: π, σ , $\Im$, ρ, $\bowtie$

Figure 5.6 One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

(i) List the names of employees who have a dependent with the same first name as themselves.

(ii) Find the names of employees that are directly supervised by 'James Borg'.

(iii) For each project, list the project name and the total hours per week (by all employees) spent on that project.

(iv) Retrieve the names of employees who work on every project.

(v) Retrieve the maximum salary of all female employees.

Solution 6: (10 marks)

In the relational algebra, as in other languages, it is possible to specify the same query in multiple ways. We give one possible solution for each query. We use the symbol S for SELECT, P for PROJECT, J for EQUIJOIN, * for NATURAL JOIN, and f for FUNCTION.

(i)

$E \leftarrow (\ EMPLOYEE \bowtie_{Ssn = Essn\ and\ Fname = Dependent\_name}\ (DEPENDENT))$

$R \leftarrow \pi_{Lname,Fname}\ (E)$

Result (empty):

LNAME FNAME

(ii))$Borg\_ssn \leftarrow \pi_{Ssn}\ (\sigma_{Fname= \text{'James'}\ and\ Lname = \text{'Borg'}}\ (EMPLOYEE))$

$Borg\_emps \leftarrow (\ EMPLOYEE \bowtie_{Super\_ssn = Ssn}\ (Borg\_ssn))$

$Result \leftarrow \pi_{Lname,Fname}\ (Borg\_emps)$

Result:

```
Select e.fname, e.lname
From employee e, employee m
Where e.super_ssn = m.ssn and m.fname = 'James' and m.lname='Borg';
```

```
FNAME           LNAME
--------------- ---------------
Franklin        Wong
Jennifer        Wallace
```

(iii)

$PROJ\_HOURS\ (Pno, Tot\_Hrs) \leftarrow Pno\ \Im_{Sum\ Hours}\ (WORKS\_ON)$

$RESULT \leftarrow \pi_{Pname,Tot\_hrs}\ (PROJ\_HOURS \bowtie_{Pno = Pnumber}\ (PROJECT))$

Result:

PNAME TOT_HRS
ProductX 52.5
ProductY 37.5
ProductZ 50.0
Computerization 55.0
Reorganization 25.0
Newbenefits 55.0

(iv)

$PROJ\_EMPS(PNO,SSN) \leftarrow \pi_{pno,\ Essn}\ (WORKS\_ON)$

$ALL\_PROJS(PNO) \leftarrow \pi_{PNUMBER}\ (PROJECT)$

EMPS_ALL_PROJS <-- PROJ_EMPS -:- ALLPROJS /* DIVISION operation */

RESULT <-- $\pi_{LNAME,FNAME}$ (EMPLOYEE $\bowtie$ EMP_ALL_PROJS) /*natural join on ssn*/

Result (empty):
LNAME FNAME

(v) RESULT(MAX_F_SAL) ← $\Im_{\text{maximum salary}}$ (($\sigma_{SEX = 'F'}$ EMPLOYEE))
Or
(v) $\rho_{RESULT(MAX\text{-}F\text{-}SAL)}$ ($\Im_{\text{maximum salary}}$ ($\sigma_{SEX = 'F'}$ EMPLOYEE))

Result:
max_F_SAL
43000