

The University of Texas at Dallas
Database Design – Prof. Nurcan, Yuruk
Assignment 02

Part1 - 1. Creating Library Database:**a. creating the tables:**

```
--DROP TABLE if exists BOOKS;
```

```
CREATE TABLE BOOKS (  
    book_id          varchar(10),  
    title            varchar(20),  
    publ_name        varchar(20),  
    primary key(book_id));
```

```
CREATE TABLE BOOK_AUTHORS (  
    book_id          varchar(10),  
    author_name       varchar(20),  
    primary key(book_id, author_name));
```

```
CREATE TABLE PUBLISHER (  
    pname            varchar(20),  
    address           varchar(50),  
    phone             varchar(20),  
    primary key(pname));
```

```
CREATE TABLE BOOK_COPIES (  
    book_id          varchar(10),  
    branch_id        varchar(10),  
    no_of_copies      integer,  
    primary key(book_id, branch_id));
```

```
CREATE TABLE BOOK_LOANS (  
    book_id          varchar(10),  
    branch_id        varchar(10),  
    card_no           varchar(10),  
    date_out          date,  
    due_date          date,  
    return_date       date default null,  
    primary key(book_id, branch_id, card_no));
```

```
CREATE TABLE LIBRARY_BRANCH (  
    branch_id         varchar(10),  
    branch_name        varchar(20),  
    address            varchar(50),  
    primary key(branch_id));
```

```
CREATE TABLE BORROWER (  
    card_no           varchar(10),  
    cname             varchar(20),  
    address            varchar(50),  
    phone             varchar(20),  
    primary key(card_no));
```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Connections' pane displays a tree view of the database schema. The 'csorade' connection is selected, and the 'Tables (Filtered)' folder is expanded, showing a list of tables including BOOK, BOOK_AUTHORS, BOOK_COPIES, BOOK_LOANS, BORROWER, DEPARTMENT, DEPENDENT, DEPT_LOCATIONS, EMPLOYEE, LIBRARY_BRANCH, PROJECT, PUBLISHER, and WORKS_ON. The 'BOOK_LOANS' table is highlighted. On the right, the 'Columns' tab for the 'BOOK_LOANS' table is active, displaying a table with the following data:

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BOOK_ID	VARCHAR2(10 BYTE)	No	(null)	1	(null)
2	BRANCH_ID	VARCHAR2(10 BYTE)	No	(null)	2	(null)
3	CARD_NO	VARCHAR2(10 BYTE)	No	(null)	3	(null)
4	DATE_OUT	DATE	Yes	(null)	4	(null)
5	DUE_DATE	DATE	Yes	(null)	5	(null)
6	RETURN_DATE	DATE	Yes	null	6	(null)

**** The default value for RETURN_DATE is set to be NULL.**

b. triggered actions that will be attached to each foreign key constraint:

```
ALTER TABLE BOOK ADD CONSTRAINT fk1 FOREIGN KEY(publisher_name) REFERENCES publisher(pname)
ON DELETE CASCADE;
```

```
ALTER TABLE BOOK_AUTHORS ADD CONSTRAINT fk2 FOREIGN KEY(book_id) REFERENCES
BOOK(book_id) ON DELETE CASCADE;
```

```
ALTER TABLE BOOK_COPIES ADD CONSTRAINT fk3 FOREIGN KEY(book_id) REFERENCES BOOK(book_id)
ON DELETE CASCADE;
```

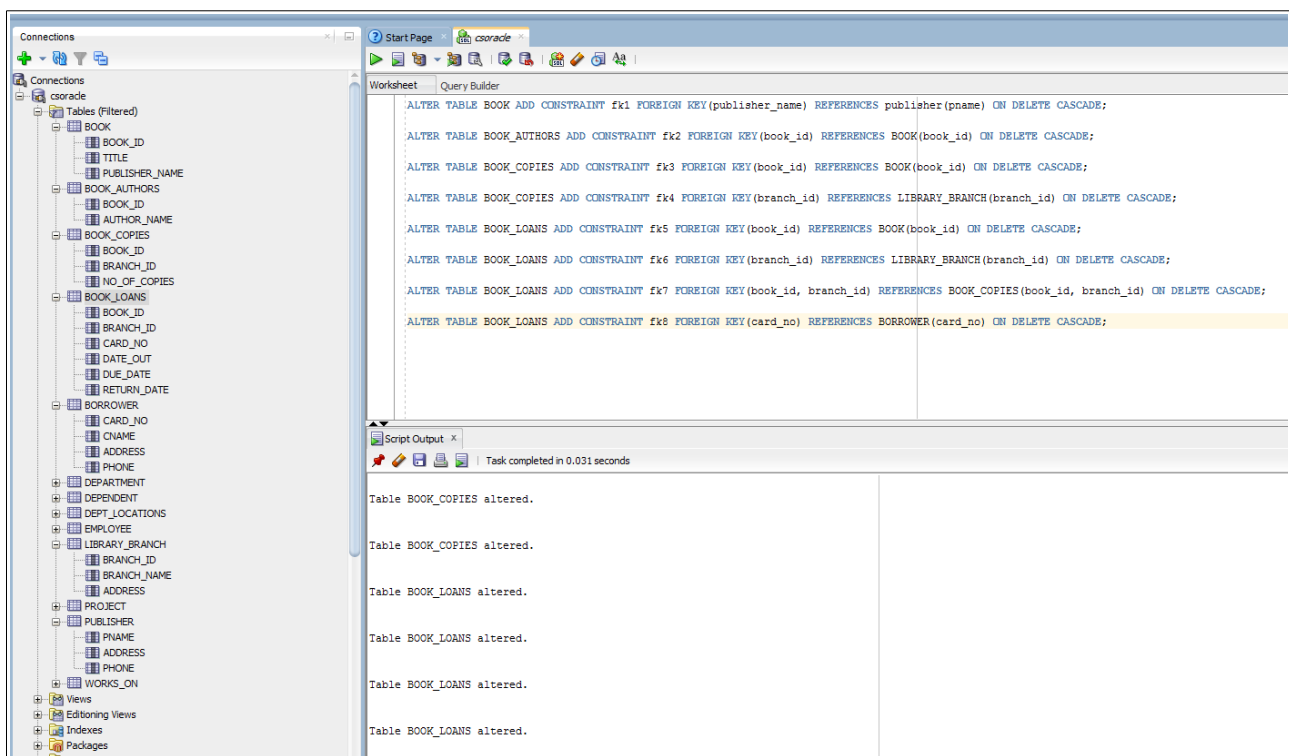
```
ALTER TABLE BOOK_COPIES ADD CONSTRAINT fk4 FOREIGN KEY(branch_id) REFERENCES
LIBRARY_BRANCH(branch_id) ON DELETE CASCADE;
```

```
ALTER TABLE BOOK_LOANS ADD CONSTRAINT fk5 FOREIGN KEY(book_id) REFERENCES BOOK(book_id)
ON DELETE CASCADE;
```

```
ALTER TABLE BOOK_LOANS ADD CONSTRAINT fk6 FOREIGN KEY(branch_id) REFERENCES
LIBRARY_BRANCH(branch_id) ON DELETE CASCADE;
```

```
ALTER TABLE BOOK_LOANS ADD CONSTRAINT fk7 FOREIGN KEY(book_id, branch_id) REFERENCES
BOOK_COPIES(book_id, branch_id) ON DELETE CASCADE;
```

***We dont need to refer BOOK_LOANS(book_id, branch_id) from BOOK_COPIES(book_id, branch_id)**



Part 2**1. Queries on COMPANY database.**

a. for each dept with AVG_EMP_SALARY > 30000, deptName and number of employees in it.

```
SELECT D.DName, COUNT(*) Num_Emp
FROM EMPLOYEE E1, DEPARTMENT D
WHERE 30000 < (
    SELECT AVG(salary)
    FROM EMPLOYEE E2
    WHERE E1.DNo = E2.DNo) AND
E1.DNo = D.DNo
GROUP BY E1.DNo, D.Dname;
```

**** There can be differences in query outputs, because of different databases data values. So, I have outputs based on the employee database that I have.**

The screenshot shows the SQL Developer interface with the query from the previous block entered in the Query Builder. The Query Result pane shows the following data:

DNAME	NUM_EMP
1 Research	4
2 Sales	14
3 Headquarters	1
4 Software	8
5 Administration	3
6 Hardware	10

b. for each dept with AVG_EMP_SALARY > 30000, deptName and number of MALE employees in it.

```
SELECT D.DName,
COUNT(*) Num_Male_Emp
FROM EMPLOYEE E1, DEPARTMENT D
WHERE 30000 < (
    SELECT AVG(salary)
    FROM EMPLOYEE E2
    WHERE E1.DNo = E2.DNo) AND
E1.DNo = D.DNo AND
E1.GENDER = 'M'
GROUP BY E1.DNo, D.DName;
```

The screenshot shows the SQL Developer interface with the query from the previous block entered in the Query Builder. The Query Result pane shows the following data:

DNAME	NUM_MALE_EMP
1 Headquarters	1
2 Software	7
3 Administration	1
4 Research	3
5 Sales	10
6 Hardware	7

c. for the department with highest emp salary, giving names of employees in it.

```
SELECT E1.FName || ' ' || E1.LName Emp_Name
FROM EMPLOYEE E1
WHERE E1.DNo = (
    SELECT E.DNo
    FROM EMPLOYEE E
    WHERE salary = ( SELECT MAX(salary)
    FROM EMPLOYEE));
```

**** There can be different total no of tuples from other's databases.**

For e.g., I have 14 employees in Dno=8, others may have 15, or so.

Kindly take a note of this.

The screenshot shows the Oracle SQL Developer interface. The Query Builder window contains the following SQL query:

```
SELECT E1.FName || ' ' || E1.LName Emp_Name
FROM EMPLOYEE E1
WHERE E1.DNo = (
  SELECT E.DNo
  FROM EMPLOYEE E
  WHERE salary = (
    SELECT MAX(salary)
    FROM EMPLOYEE));
```

The Query Result window shows the results of the query, displaying 14 rows of employee names:

EMP_NAME
1 Bob Bender
2 Jill Jarvis
3 Kate King
4 Lyle Leslie
5 Billie King
6 Jon Kramer
7 Ray King
8 Gerald Small
9 Arnold Head
10 Helga Pataki
11 Naveen Drew
12 Carl Reedy
13 Sammy Hall
14 Red Bacher

d. employees making atleast 10000 more than the least salaried employee in the company.

```
SELECT E1.FName || ' ' || E1.LName
Emp_Name
FROM EMPLOYEE E1
WHERE E1.salary >= (
  SELECT MIN(salary)+10000
  FROM EMPLOYEE);
```

The screenshot shows the Oracle SQL Developer interface. The Query Builder window contains the following SQL query:

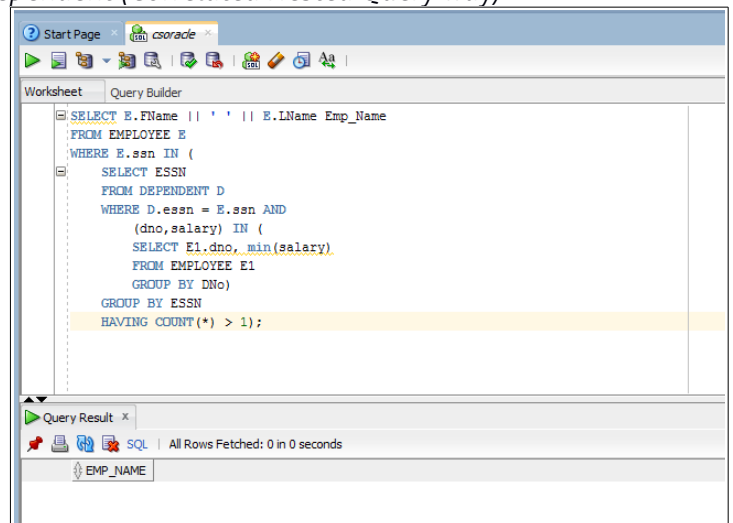
```
SELECT E1.FName || ' ' || E1.LName Emp_Name
FROM EMPLOYEE E1
WHERE E1.salary >= (SELECT MIN(salary)+10000 FROM EMPLOYEE);
```

The Query Result window shows the results of the query, displaying 30 rows of employee names:

EMP_NAME
1 Andy Vile
2 Brad Knight
3 Evan Wallis
4 Josh Zell
5 Jared James
6 Justin Mark
7 Jon Jones
8 John James
9 Alex Freed
10 Ramesh Narayan
11 Jennifer Wallace
12 Franklin Wong
13 James Borg
14 Tom Brand
15 Jenny Vos
16 Chris Carter
17 Kim Grace
18 Jeff Chase
19 Bonnie Bays
20 Alec Best
21 Sam Snedden
22 Nandita Ball
23 Bob Bender
24 Jill Jarvis
25 Kate King
26 Lyle Leslie
27 Billie King
28 Jon Kramer
29 Ray King
30 Sammy Hall

e. employees making least in their dept, with >1 dependent (Correlated Nested Query way)

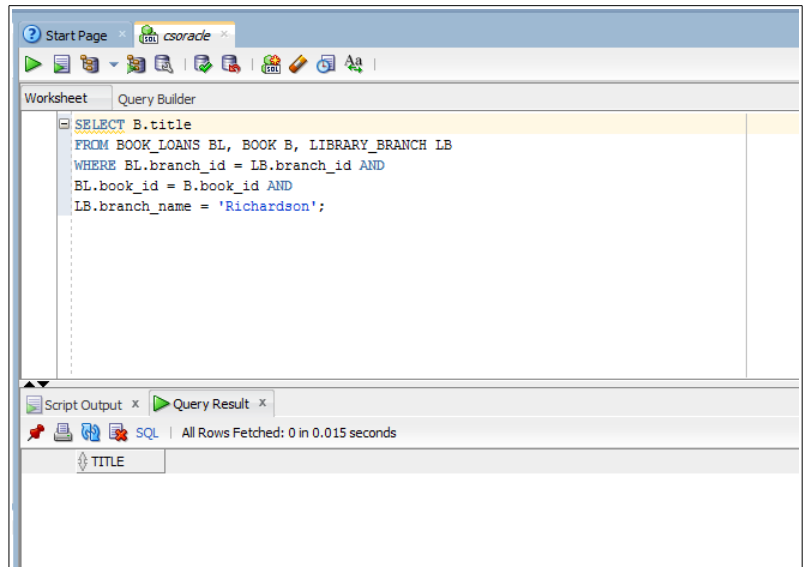
```
SELECT E.FName || ' ' || E.LName Emp_Name
FROM EMPLOYEE E
WHERE E.ssn IN (
    SELECT D.essn
    FROM DEPENDENT D
    WHERE D.essn = E.ssn AND
        (dno,salary) IN (
            SELECT E1.dno, min(salary)
            FROM EMPLOYEE E1
            GROUP BY DNo)
    GROUP BY essn
    HAVING COUNT(*) > 1);
```



2. Queries on LIBRARY database.

a. books borrowed form 'Richardson' library

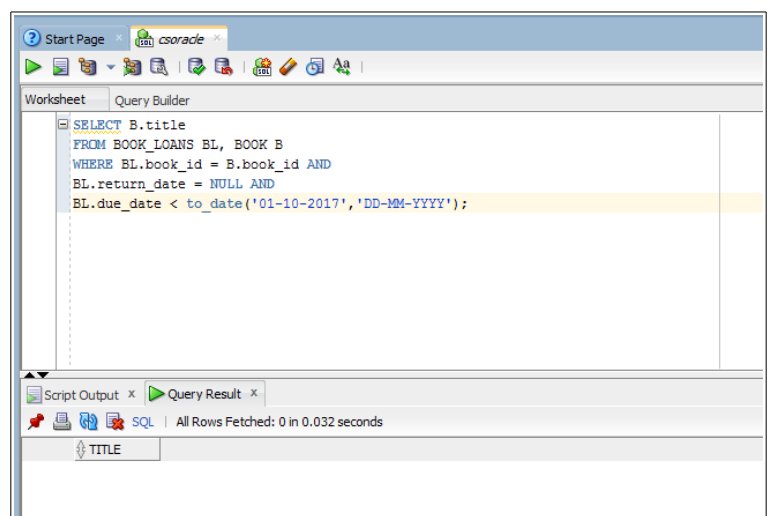
```
SELECT B.title
FROM
BOOK_LOANS BL, BOOK B,
LIBRARY_BRANCH LB
WHERE
BL.branch_id = LB.branch_id AND
BL.book_id = B.book_id AND
LB.branch_name = 'Richardson';
```



b. books that are overdue

```
SELECT B.title
FROM BOOK_LOANS BL, BOOK B
WHERE BL.book_id = B.book_id AND
    BL.return_date = NULL AND
    BL.due_date < to_date(
    '01-10-2017','DD-MM-YYYY');
```

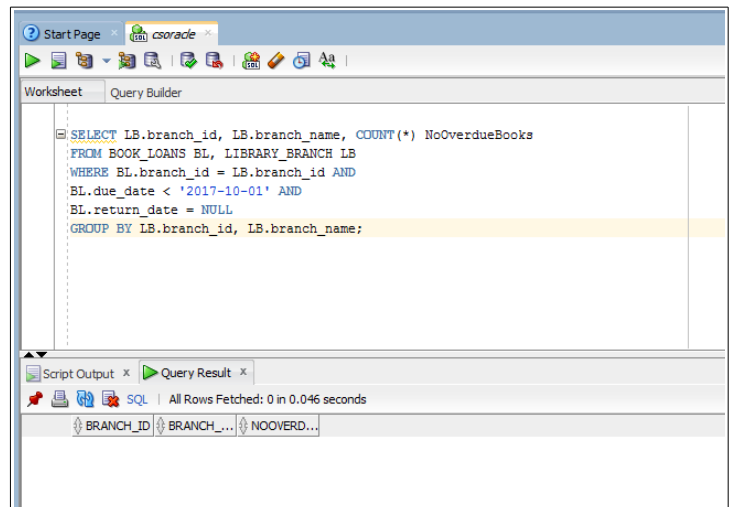
****OR BL.due_date < current_date;**



c. for each library branch, no of overdue books

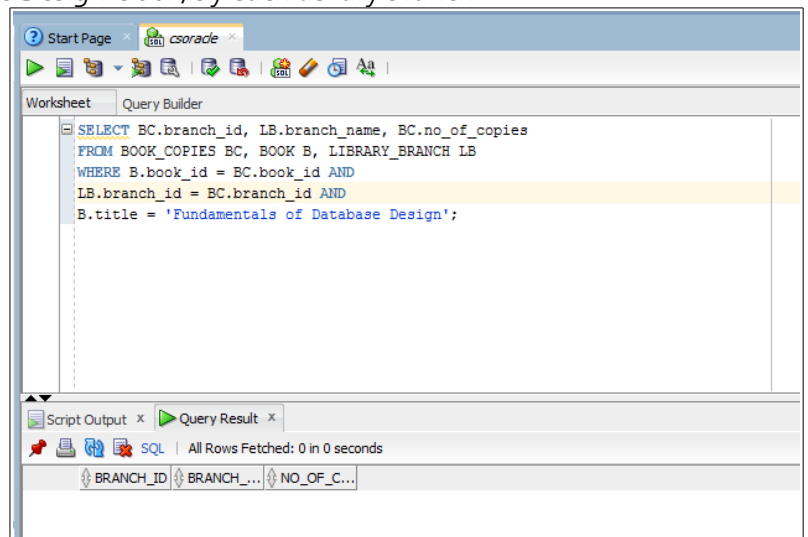
```
SELECT LB.branch_id, LB.branch_name,
COUNT(*) NoOverdueBooks
FROM
BOOK_LOANS BL, LIBRARY_BRANCH LB
WHERE
BL.branch_id = LB.branch_id AND
BL.due_date < '2017-10-01' AND
BL.return_date = NULL
GROUP BY LB.branch_id, LB.branch_name;
```

****OR BL.due_date < current_date;**



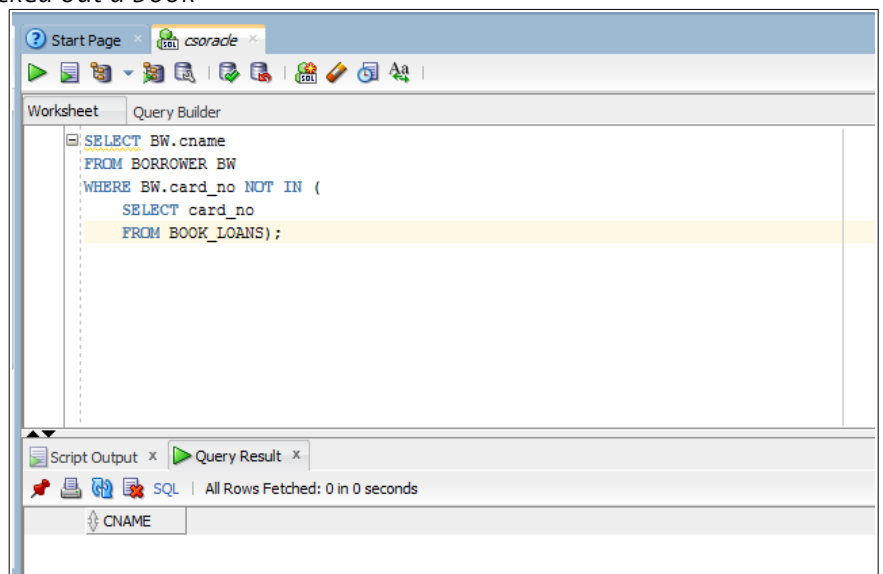
d. no of copies of 'Fundamentals of Database Design' book, by each library branch

```
SELECT BC.branch_id,
LB.branch_name, BC.no_of_copies
FROM BOOK_COPIES BC, BOOK B,
LIBRARY_BRANCH LB
WHERE B.book_id = BC.book_id AND
LB.branch_id = BC.branch_id AND
B.title = 'Fundamentals of
Database Design';
```



e. names of borrowers who hasn't checked out a book

```
SELECT BW.cname
FROM BORROWER BW
WHERE BW.card_no NOT IN (
SELECT card_no
FROM BOOK_LOANS);
```



3. Specifying VIEWS on COMPANY database using Correlated Nested Queries (except a.)

a. VIEW for each department, it's name, manager name and manager salary

```
CREATE VIEW DEPT_MANAGER AS (
    SELECT D.DName,
           E.Fname|| ' '|| E.LName MgrName,
           E.salary MgrSalary
    FROM
        EMPLOYEE E, DEPARTMENT D
    WHERE D.MgrSSN = E.ssn
);
```

```
SELECT * FROM DEPT_MANAGER;
```

DNAME	MGRNAME	SALARY
1 Software	Jared James	85000
2 Research	Franklin Wong	40000
3 Hardware	Alex Freed	89000
4 Sales	Nandita Ball	62000
5 Headquarters	James Borg	55000
6 Administration	Jennifer Wallace	43000

b. VIEW for each dept, it's name, mngr name, no of empl, no of projects

```
CREATE VIEW DEPT_DETAILS AS (
    SELECT D.DName, E.Fname|| ' '|| E.LName MgrName,
           (SELECT COUNT(*) FROM EMPLOYEE E1 WHERE D.DNo = E1.DNo) Num_Emp,
           (SELECT COUNT(*) FROM PROJECT P1 WHERE D.DNo = P1.DNo) Num_Proj
    FROM
        EMPLOYEE E, DEPARTMENT D
    WHERE D.MgrSSN = E.ssn);
```

```
FROM
EMPLOYEE E, DEPARTMENT D
WHERE D.MgrSSN = E.ssn);
```

```
SELECT * FROM DEPT_DETAILS;
```

DNAME	MGRNAME	NUM_EMP	NUM_PROJ
1 Software	Jared James	8	3
2 Research	Franklin Wong	4	3
3 Hardware	Alex Freed	10	2
4 Sales	Nandita Ball	14	0
5 Headquarters	James Borg	1	1
6 Administration	Jennifer Wallace	3	2

c. VIEW for each project, its name, deptname, no of empl, total hrs per week

```
CREATE VIEW PROJECT_DETAILS AS (
    SELECT P.PName, D.DNo, D.Dname,
    (SELECT COUNT(W1.SSN) FROM WORKS_ON W1 WHERE W1.PNo = P.PNo) No_Empl,
    (SELECT SUM(W2.HOURS) FROM WORKS_ON W2 WHERE W2.PNo = P.PNo) Tot_Hrs

    FROM PROJECT P, DEPARTMENT D
    WHERE P.DNo = D.DNo);
```

```
SELECT * FROM PROJ_DETAILS;
```

****Instead for DeptName, I've here Dno.**
:) Rest of the query is working fine, I guess.

The screenshot shows the Oracle SQL Developer interface. The top pane displays the SQL script for creating the view `PROJECT_DETAILS` and querying it. The bottom pane shows the query results in a table format.

PNAME	DNO	NUM_EMP	TOT_HOURS
1 ProductX	5	2	52.5
2 ProductY	5	3	37.5
3 ProductZ	5	2	50
4 Computerization	4	3	55
5 Reorganization	1	3	25
6 Newbenefits	4	3	55
7 OperatingSystems	6	9	350
8 DatabaseSystems	6	8	298
9 Middleware	6	4	136
10 InkjetPrinters	7	8	320
11 LaserPrinters	7	3	124

d. VIEW for each project, its name, deptname, no of empl, total hrs per week, with > 1 empl

```
CREATE VIEW PROJ_DETAILS AS (
    SELECT P.PName, D.DNo, D.Dname,
    (SELECT COUNT(W1.SSN) FROM WORKS_ON W1 WHERE W1.PNo = P.PNo) Num_Emp,
    (SELECT SUM(W2.HOURS) FROM WORKS_ON W2 WHERE W2.PNo = P.PNo) Tot_Hours

    FROM PROJECT P, DEPARTMENT D
    WHERE P.DNo = D.Dno AND
    (SELECT COUNT(W1.SSN) FROM WORKS_ON W1 WHERE W1.PNo = P.PNo) > 1);
```

```
SELECT * FROM PROJ_DETAILS;
```

****Instead for DeptName, I've here Dno.**
:) Rest of the query is working fine, I guess.

```

CREATE VIEW PROJ_DETAILS AS (
  SELECT P.PName, D.DNo,
    (SELECT COUNT(W1.SSN) FROM WORKS_ON W1 WHERE W1.PNo = P.PNo) Num_Emp,
    (SELECT SUM(W2.HOURS) FROM WORKS_ON W2 WHERE W2.PNo = P.PNo) Tot_Hours
  FROM PROJECT P, DEPARTMENT D
  WHERE P.DNo = D.Dno AND
    (SELECT COUNT(W1.SSN) FROM WORKS_ON W1 WHERE W1.PNo = P.PNo) > 1);

SELECT * FROM PROJ_DETAILS;

```

	PNAME	DNO	NUM_EMP	TOT_HOURS
1	ProductX	5	2	52.5
2	ProductY	5	3	37.5
3	ProductZ	5	2	50
4	Computerization	4	3	55
5	Reorganization	1	3	25
6	Newbenefits	4	3	55
7	OperatingSystems	6	9	350
8	DatabaseSystems	6	8	298
9	Middleware	6	4	136
10	InkjetPrinters	7	8	320
11	LaserPrinters	7	3	124

e. VIEW for each employee, it's name, salary, dept, dept mngr, mngr salary and avg dept salary

```

CREATE VIEW EMP_DETAILS AS (
  SELECT E.Fname||' '||E.Lname EmpName, E.salary, D.Dname,
    (SELECT E2.Fname||' '||E2.Lname FROM EMPLOYEE E2 WHERE D.MgrSSN = E2.SSN) MgrName,
    (SELECT E2.salary FROM EMPLOYEE E2 WHERE E2.SSN = D.MgrSSN) MgrSalary,
    ROUND((SELECT AVG(salary) FROM EMPLOYEE E2 WHERE E2.Dno = E.Dno),2) AvgDeptSalary

```

```

FROM
  EMPLOYEE E,
  DEPARTMENT D
  WHERE E.DNo = D.DNo);

```

```

SELECT * FROM EMP_DETAILS;

```

```

CREATE VIEW EMP_DETAILS AS (
  SELECT E.Fname||' '||E.Lname EmpName, E.salary, D.Dname,
    (SELECT E2.Fname||' '||E2.Lname FROM EMPLOYEE E2 WHERE D.MgrSSN = E2.SSN) MgrName,
    (SELECT E2.salary FROM EMPLOYEE E2 WHERE E2.SSN = D.MgrSSN) MgrSalary,
    ROUND((SELECT AVG(salary) FROM EMPLOYEE E2 WHERE E2.Dno = E.Dno), 2) AvgDeptSalary
  FROM
    EMPLOYEE E, DEPARTMENT D
  WHERE E.DNo = D.DNo);

SELECT * FROM EMP_DETAILS;

```

	EMPNAME	SALARY	DNAME	MGRNAME	MGRSALARY	AVGDEPTSALARY
1	James Borg	55000	Headquarters	James Borg	55000	55000
2	John James	81000	Software	Jared James	85000	60000
3	Brad Knight	44000	Software	Jared James	85000	60000
4	Jon Jones	45000	Software	Jared James	85000	60000
5	Jeff Chase	44000	Software	Jared James	85000	60000
6	Kim Grace	79000	Software	Jared James	85000	60000
7	Nandita Ball	62000	Software	Jared James	85000	60000
8	Justin Mark	40000	Software	Jared James	85000	60000
9	Jared James	85000	Software	Jared James	85000	60000
10	Ahmad Jabbar	25000	Administration	Jennifer Wallace	43000	31000
11	Alicia Zelaya	25000	Administration	Jennifer Wallace	43000	31000
12	Jennifer Wallace	43000	Administration	Jennifer Wallace	43000	31000
13	Franklin Wong	40000	Research	Franklin Wong	40000	33250
14	Joyce English	25000	Research	Franklin Wong	40000	33250
15	Ramesh Narayan	38000	Research	Franklin Wong	40000	33250
16	John Smith	30000	Research	Franklin Wong	40000	33250
17	Ray King	44500	Sales	Nandita Ball	62000	40821.43
18	Sammy Hall	37000	Sales	Nandita Ball	62000	40821.43