

EZMedicalGroup.com

Phurba Sherpa

NYC College of Technology

Hospital Management System Database Design & Implementation Version 2.0

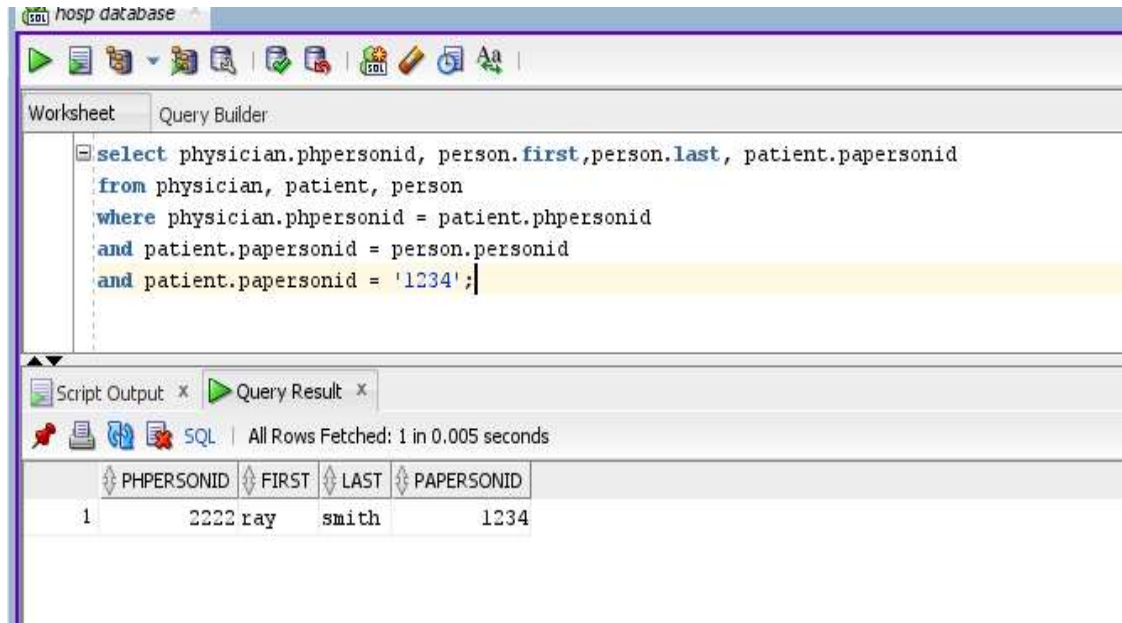


Executive Summary

This is a new upgrade on EZ Medical Group. In this Document there are business reports which helps the hospital system find different important data and information about the hospital. We will also create stored procedure which is needed to make business decision and business reports by executing the business reports that are created.

Business report 1

The patient was asking for the name of the doctor that he had check up with to the Front desk receptionist. The patient id was 1234. For this report, table physician, patient, person was used to look up for physician name and id number.



The screenshot shows a SQL query in a 'Query Builder' window. The query is:

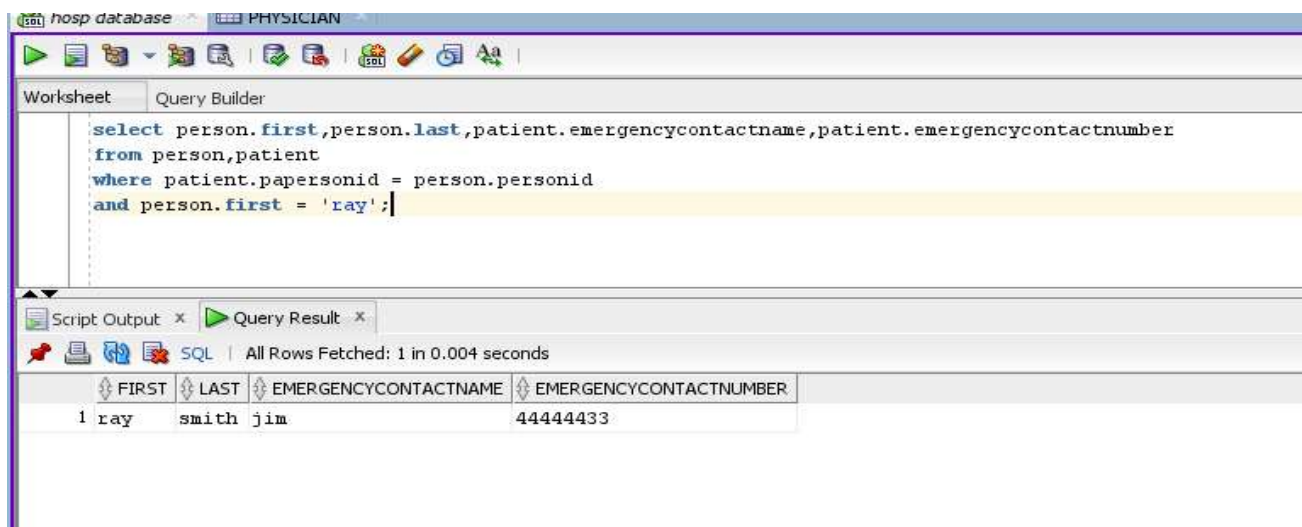
```
select physician.phpersonid, person.first, person.last, patient.papersonid
from physician, patient, person
where physician.phpersonid = patient.phpersonid
and patient.papersonid = person.personid
and patient.papersonid = '1234';
```

The query results are displayed in a table with the following columns: PHPERSONID, FIRST, LAST, and PAPERSONID. The results show one row with the following values: 1, 2222 ray, smith, 1234.

PHPERSONID	FIRST	LAST	PAPERSONID
1	2222 ray	smith	1234

Business report 2

The hospital had an emergency situation where they had to call for patient's relative for an emergency situation. The front desk emergency staff wants to know the number, emergency contact name and number. The patient first name is Ray. To check number of Emergency contact name and emergency contact number, I had to use table person and patient.



The screenshot shows a SQL query in a 'Query Builder' window. The query is:

```
select person.first, person.last, patient.emergencycontactname, patient.emergencycontactnumber
from person, patient
where patient.papersonid = person.personid
and person.first = 'ray';
```

The query results are displayed in a table with the following columns: FIRST, LAST, EMERGENCYCONTACTNAME, and EMERGENCYCONTACTNUMBER. The results show one row with the following values: 1 ray, smith jim, 44444433.

FIRST	LAST	EMERGENCYCONTACTNAME	EMERGENCYCONTACTNUMBER
1 ray	smith jim		44444433

Business report 3

The Front desk receptionist wants to know the payment information about patient ID number 1235. TO check the credit card information, Table patient, credit_card,patient, credit_card is used.

The screenshot shows a database query builder interface with a toolbar at the top and a 'hosp database' dropdown. The 'Query Builder' tab is active, displaying a SQL query in a text area. Below the query, the 'Query Result' section shows the execution status and a table of results.

```
select credit_card.ownername,patient.papersonid,credit_card.merchantname,credit_card.expdate,credit_card.creditcardnumber
from patient,credit_card,patient_credit_card
where credit_card.creditcardnumber = patient_credit_card.creditcardnumber
and patient_credit_card.papersonid = patient.papersonid
and patient.papersonid = '1235';
```

Query Result: All Rows Fetched: 1 in 0.005 seconds

	OWNERNAME	PAPERSONID	MERCHANTNAME	EXPDATE	CREDITCARDNUMBER
1	cane	1235	chase	11/25	1234567893

Business report 4

The Front desk receptionist wants to look up date admitted and date discharged for the resident patient. To view the date admitted and date discharged, table person, resident_patient, person was used.

The screenshot shows a database query builder interface with a toolbar at the top and a 'hosp database' dropdown. The 'Query Builder' tab is active, displaying a SQL query in a text area. Below the query, the 'Query Result' section shows the execution status and a table of results.

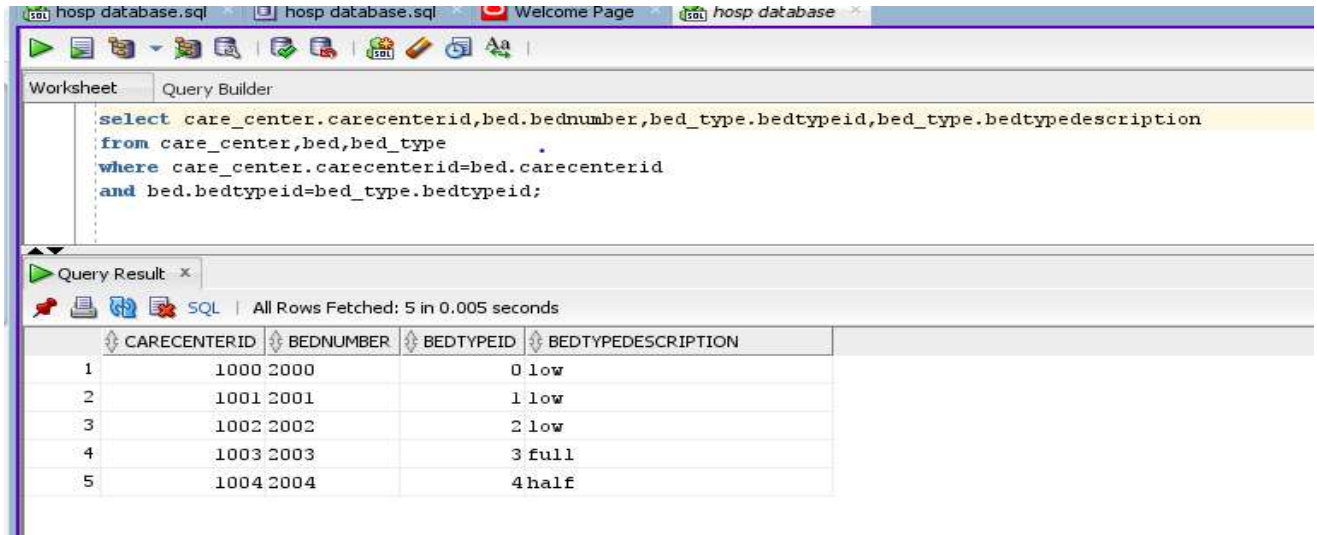
```
select patient.papersonid,resident_patient.dateadmitted,person.first,resident_patient.datedischarged
from patient,resident_patient,person
where patient.papersonid = resident_patient.papersonid
and patient.papersonid=person.personid;
```

Query Result: All Rows Fetched: 5 in 0.009 seconds

	PAPERSONID	DATEADMITTED	FIRST	DATEDISCHARGED
1	1234	10-DEC-04	ray	09-NOV-05
2	1235	10-DEC-05	allen	09-NOV-06
3	1236	10-DEC-05	james	09-NOV-06
4	1237	10-DEC-05	corey	09-NOV-06
5	1238	10-DEC-05	nancy	09-NOV-06

Business report 5

The hospital management staffs want to check which bed are used the most so they could notify the hospital executive if any type of bed is needed. TO check the bed type description, table care_center,bed and bed_type was used.



The screenshot shows a database query tool interface. The top bar has tabs for 'hosp database.sql', 'hosp database.sql', 'Welcome Page', and 'hosp database'. Below the tabs is a toolbar with various icons. The main area is divided into 'Worksheet' and 'Query Builder' tabs. The 'Query Builder' tab is active, showing a SQL query:

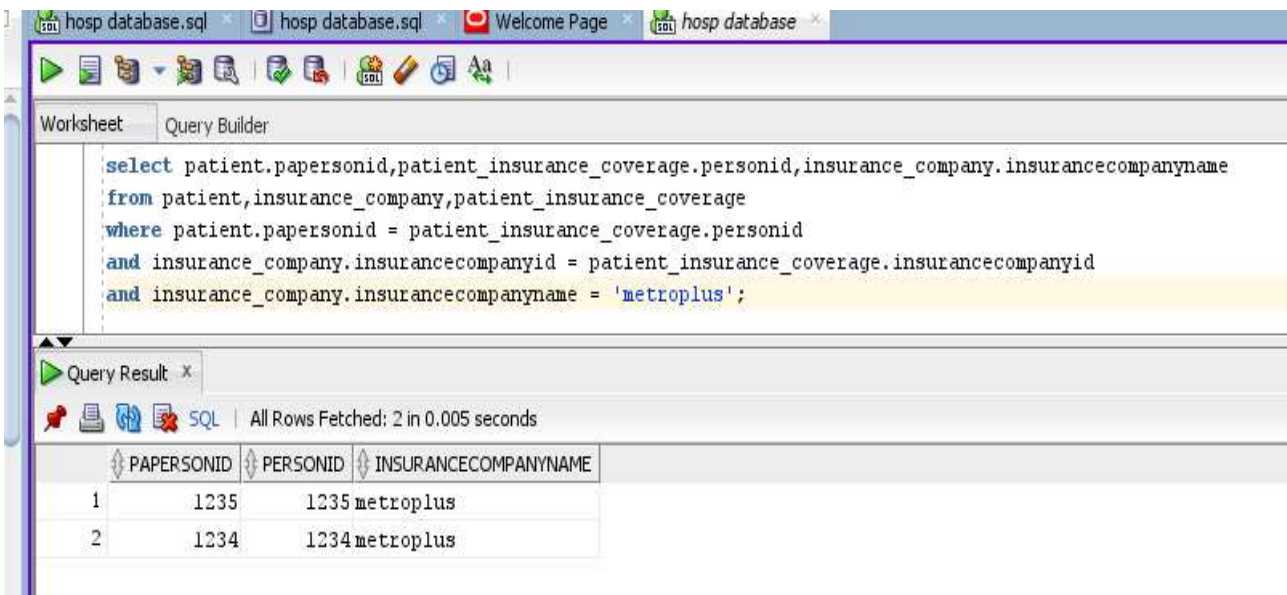
```
select care_center.carecenterid,bed.bednumber,bed_type.bedtypeid,bed_type.bedtypedescription
from care_center,bed,bed_type
where care_center.carecenterid=bed.carecenterid
and bed.bedtypeid=bed_type.bedtypeid;
```

Below the query is a 'Query Result' tab showing the results of the query. The results are displayed in a table with 4 columns: CARECENTERID, BEDNUMBER, BEDTYPEID, and BEDTYPEDESCRIPTION. The table contains 5 rows of data.

CARECENTERID	BEDNUMBER	BEDTYPEID	BEDTYPEDESCRIPTION
1	1000 2000	0	low
2	1001 2001	1	low
3	1002 2002	2	low
4	1003 2003	3	full
5	1004 2004	4	half

Business report 6

The hospital marketing staff wants to see how many Patient have metro-plus insurance, so they can notify them that the EZMedicalGroup will stop taking metro-plus insurance beginning 2022. To look up for insurance name, table patient id and Insurance company name was used.



The screenshot shows a database query tool interface. The top bar has tabs for 'hosp database.sql', 'hosp database.sql', 'Welcome Page', and 'hosp database'. Below the tabs is a toolbar with various icons. The main area is divided into 'Worksheet' and 'Query Builder' tabs. The 'Query Builder' tab is active, showing a SQL query:

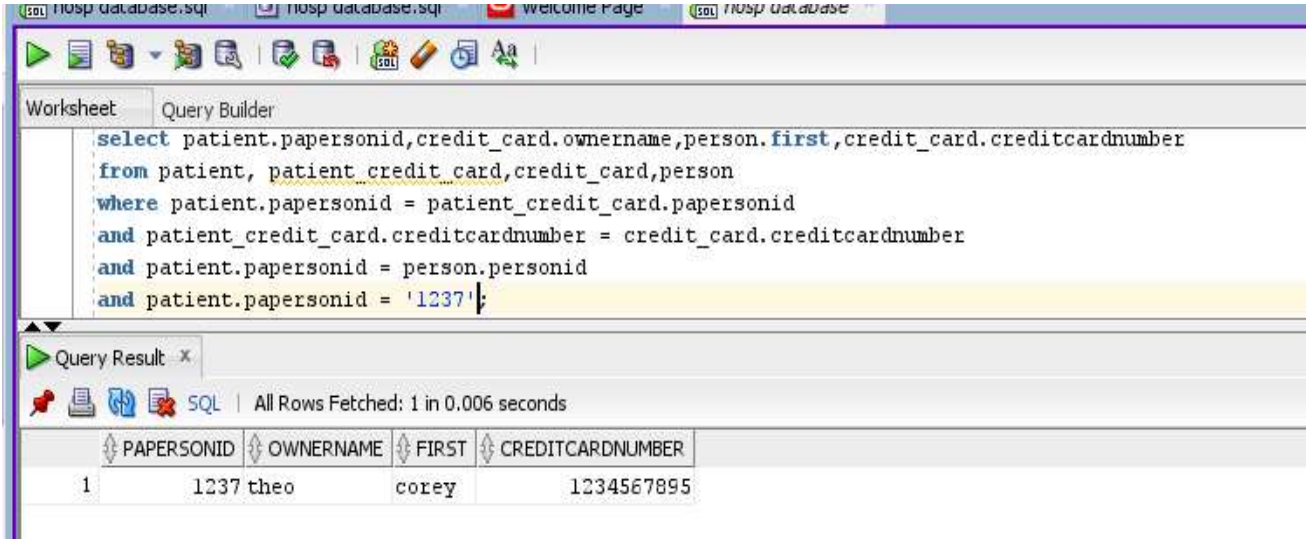
```
select patient.papersonid,patient_insurance_coverage.personid,insurance_company.insurancecompanyname
from patient,insurance_company,patient_insurance_coverage
where patient.papersonid = patient_insurance_coverage.personid
and insurance_company.insurancecompanyid = patient_insurance_coverage.insurancecompanyid
and insurance_company.insurancecompanyname = 'metroplus';
```

Below the query is a 'Query Result' tab showing the results of the query. The results are displayed in a table with 3 columns: PAPERSONID, PERSONID, and INSURANCECOMPANYNAME. The table contains 2 rows of data.

PAPERSONID	PERSONID	INSURANCECOMPANYNAME
1	1235	1235 metroplus
2	1234	1234 metroplus

Business report 7

The Front-desk receptionist wants to verify the payment information with patient number 1237, To view the payment information and the owner name of the card, table patient, patient_credit_card, credit_card, person is used.



The screenshot shows a SQL query in a 'Query Builder' window. The query selects patient information, credit card details, and the first name of the person associated with the card, filtered by patient number 1237. The results pane shows one row of data.

```
select patient.papersonid,credit_card.ownername,person.first,credit_card.creditcardnumber
from patient, patient_credit_card,credit_card,person
where patient.papersonid = patient_credit_card.papersonid
and patient_credit_card.creditcardnumber = credit_card.creditcardnumber
and patient.papersonid = person.personid
and patient.papersonid = '1237';
```

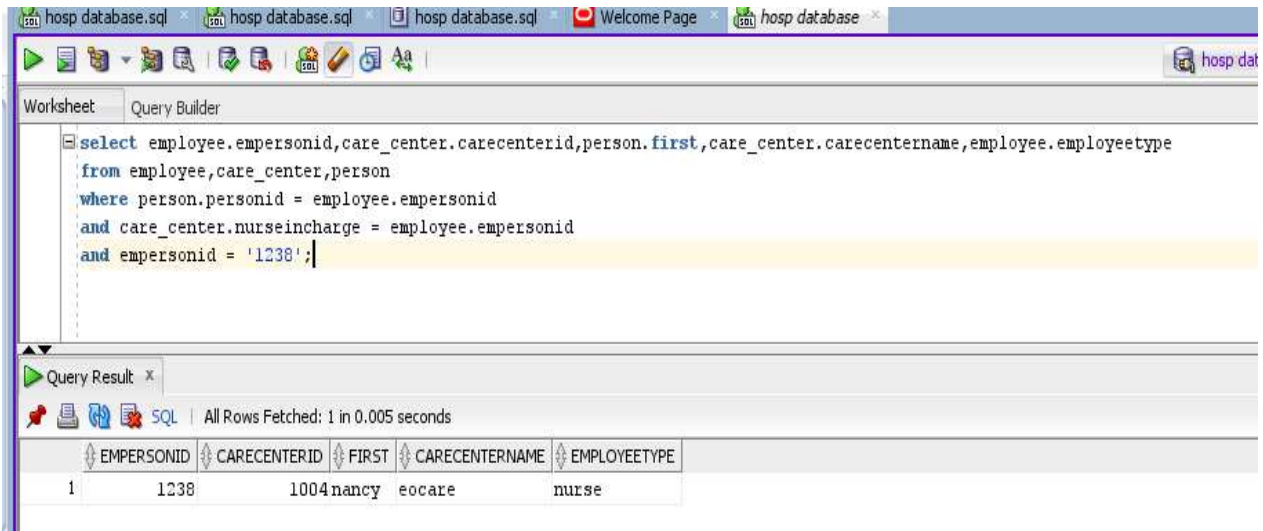
Query Result x

SQL | All Rows Fetched: 1 in 0.006 seconds

	PAPERSONID	OWNERNAME	FIRST	CREDITCARDNUMBER
1	1237	theo	corey	1234567895

Business report 8

The hospital executive wants to check the name, care-center they are assigned for the employee 1238 to make sure the employee is assigned to right care center. To view the care center name, table employee, care-center and person is used.



The screenshot shows a SQL query in a 'Query Builder' window. The query selects employee information, care center details, and the first name of the person, filtered by employee number 1238. The results pane shows one row of data.

```
select employee.empersonid,care_center.carecenterid,person.first,care_center.carecentername,employee.employeetype
from employee,care_center,person
where person.personid = employee.empersonid
and care_center.nurseincharge = employee.empersonid
and empersonid = '1238';
```

Query Result x

SQL | All Rows Fetched: 1 in 0.005 seconds

	EMPERSONID	CARECENTERID	FIRST	CARECENTERNAME	EMPLOYEE TYPE
1	1238	1004	nancy	eocare	nurse

Business report 9

The receptionist wants to check the date-admitted and date discharged for the patient id 1237. To view the date-admitted and date discharged, table person, patient, and resident_patient is used.

The screenshot shows a SQL Query Builder interface. The query is as follows:

```
select patient.papersonid, person.first, resident_patient.dateadmitted, resident_patient.datedischarged
from person, patient, resident_patient
where person.personid= patient.papersonid
and patient.papersonid = resident_patient.papersonid
and patient.papersonid = '1237';
```

The Query Result pane shows the following data:

	PAPERSONID	FIRST	DATEADMITTED	DATEDISCHARGED
1	1237	corey	10-DEC-05	09-NOV-06

Business Report 10

The hospital executive wants to check the employee name, ID who works in ercare center. To view employee who works in ercare, Table person, Employee and care_center is used.

The screenshot shows a SQL Query Builder interface. The query is as follows:

```
select employee.empersonid, care_center.carecenterid, person.first, employee.employeetype, care_center.carece
from employee, care_center, person
where person.personid = employee.empersonid
and care_center.nurseincharge = employee.empersonid
and care_center.carecentername = 'ercare';
```

The Query Result pane shows the following data:

	EMPERSIONID	CARECENTERID	FIRST	EMPLOYEE TYPE	CARECENTERNAME
1	1235	1001	allen	doctor	ercare

Store Procedure

Store Procedure # 1

```
create or replace PROCEDURE usp_patienttableinfo
IS
v_phpersonid NUMBER(4);
v_first      VARCHAR2(25);
v_last       VARCHAR2(25);
v_papersonid NUMBER(4);

CURSOR cur_patienttable is

select physician.phpersonid, person.first, person.last, patient.papersonid
from physician, patient, person
where physician.phpersonid = patient.phpersonid
and patient.papersonid = person.personid
and patient.papersonid = '1234';

BEGIN
OPEN cur_patienttable;
LOOP
FETCH cur_patienttable into v_phpersonid, v_first, v_last, v_papersonid;
EXIT WHEN cur_patienttable%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('physician info: ' || v_phpersonid || ' ' || v_first || ' ' || v_last || ' ' || v_papersonid);
END LOOP;
CLOSE cur_patienttable;
END usp_patienttableinfo;
```

The screenshot shows a database IDE interface. At the top, there is a toolbar with various icons. Below it, a tab labeled 'Worksheet' is active, displaying the following SQL code:

```
.SET SERVEROUT ON;
EXECUTE usp_personinfo;
```

Below the worksheet, there is a 'Script Output' window. It shows the results of the procedure execution:

```
person first name is : ray
person last name is : smith
person emergency contact name : jim
person emergency contact : 44444433

PL/SQL procedure successfully completed.
```

The output window also indicates that the task was completed in 0.035 seconds.

Store Procedure # 2

```
create or replace PROCEDURE usp_personinfo
IS
v_first varchar2(25);
v_last  varchar2(25);
v_emergencycontactname VARCHAR2(50);
v_emergencycontactnumber VARCHAR2(20);

CURSOR cur_person is

select person.first,person.last,patient.emergencycontactname, patient.emergencycontactnumber
from person,patient
where patient.papersonid = person.personid
and
person.first = 'ray';
BEGIN
OPEN cur_person;
LOOP
FETCH cur_person into v_first, v_last, v_emergencycontactname, v_emergencycontactnumber;
EXIT WHEN cur_person%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('person frist name is : ' ||v_first);
DBMS_OUTPUT.PUT_LINE('person last name is : ' ||v_last);
DBMS_OUTPUT.PUT_LINE('person emergency contact name : ' ||v_emergencycontactname);
DBMS_OUTPUT.PUT_LINE('person emergency contact : ' ||v_emergencycontactnumber);

END LOOP;
CLOSE cur_person;
END usp_personinfo;
```

Worksheet Query Builder

SET SERVEROUT on;
EXECUTE usp_personinfo();

Script Output x

Task completed in 0.045 seconds

person frist name is : ray
person last name is : smith
person emergency contact name : jim
person emergency contact : 44444433

PL/SQL procedure successfully completed.

Store Procedure # 3

```
create or replace PROCEDURE usp_paymentinfo
IS
v_ownername  VARCHAR2(50);
v_papersonid NUMBER(4);
v_expdate    char(5);
v_creditcardnumber NUMBER(16);

CURSOR cur_payment is
select credit_card.ownername,patient.papersonid,credit_card.expdate,credit_card.creditcardnumber
from patient, credit_card, patient_credit_card
where credit_card.creditcardnumber = patient_credit_card.creditcardnumber
and patient_credit_card.papersonid = patient.papersonid
and patient.papersonid = '1235';

BEGIN
OPEN cur_payment;
LOOP
FETCH cur_payment INTO v_ownername,v_papersonid,v_expdate,v_creditcardnumber;
EXIT when cur_payment%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('owner name is : ' ||v_ownername);
DBMS_OUTPUT.PUT_LINE('person ID is : ' ||v_papersonid);
DBMS_OUTPUT.PUT_LINE('Expiration date is : ' ||v_expdate);
DBMS_OUTPUT.PUT_LINE('credit card number is : ' ||v_creditcardnumber);
END LOOP;
CLOSE cur_payment;
END usp_paymentinfo;
```

The screenshot shows a database query tool interface. At the top, there are two tabs: "Worksheet" and "Query Builder". The "Query Builder" tab is active, displaying the following SQL code:

```
SET SERVEROUT on;
EXECUTE usp_paymentinfo();
```

Below the query editor, there is a "Script Output" window. It shows the results of the procedure execution:

```
owner name is : cane
person ID is : 1235
Expiration date is : 11/25
credit card number is : 1234567893
```

At the bottom of the "Script Output" window, it states: "PL/SQL procedure successfully completed." Above the output, there is a status bar that says "Task completed in 0.036 seconds".

Store Procedure # 4

```
create or replace PROCEDURE usp_dateinfo
IS
  v_papersonid      number(4);
  v_dateadmitted    date;
  v_first           varchar2(25);
  v_datedischarged  date;

  CURSOR cur_dates is

  select patient.papersonid, resident_patient.dateadmitted, person.first, resident_patient.datedischarged
  from patient, resident_patient, person
  where patient.papersonid = resident_patient.papersonid
  and patient.papersonid = person.personid;
BEGIN
  OPEN cur_dates;
  LOOP
    FETCH cur_dates into v_papersonid, v_dateadmitted, v_first, v_datedischarged;
    EXIT WHEN cur_dates%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('person frist name is : ' ||v_papersonid);
    DBMS_OUTPUT.PUT_LINE('date admitted : ' ||v_dateadmitted);
    DBMS_OUTPUT.PUT_LINE('person frist name : ' ||v_first);
    DBMS_OUTPUT.PUT_LINE('person datedischarged : ' ||v_datedischarged);

  END LOOP;
  CLOSE cur_dates;
END usp_dateinfo;
```

```
SET SERVEROUT on;
EXECUTE usp_dateinfo();
```

Script Output x

Task completed in 0.031 seconds

```
person frist name is : 1234
date admitted : 10-DEC-04
person frist name : ray
person datedischarged : 09-NOV-05
person frist name is : 1235
date admitted : 10-DEC-05
person frist name : allen
person datedischarged : 09-NOV-06
person frist name is : 1236
date admitted : 10-DEC-05
person frist name : james
person datedischarged : 09-NOV-06
person frist name is : 1237
date admitted : 10-DEC-05
person frist name : corey
person datedischarged : 09-NOV-06
person frist name is : 1238
date admitted : 10-DEC-05
person frist name : nancy
person datedischarged : 09-NOV-06
```

PL/SQL procedure successfully completed.

Store Procedure # 5

```
create or replace PROCEDURE usp_bedinfo
IS
v_care_centerid number(4);
v_bednumber      char(5);
v_bedtypeid      number(2);
v_bedtypedescription char(25);

CURSOR cur_bed is

select care_center.carecenterid, bed.bednumber, bed_type.bedtypeid,bed_type.bedtypedescription
from care_center,bed,bed_type
where care_center.carecenterid = bed.carecenterid
and bed.bedtypeid = bed_type.bedtypeid;

BEGIN
OPEN cur_bed;
LOOP
FETCH cur_bed into v_care_centerid, v_bednumber, v_bedtypeid, v_bedtypedescription;
EXIT WHEN cur_bed%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('care center is : ' ||v_care_centerid);
DBMS_OUTPUT.PUT_LINE('bed number : ' ||v_bednumber);
DBMS_OUTPUT.PUT_LINE('bed type id : ' ||v_bedtypeid);
DBMS_OUTPUT.PUT_LINE('bed description : ' ||v_bedtypedescription);

END LOOP;
CLOSE cur_bed;
END usp_bedinfo;
```

Worksheet	Query Builder
<pre>SET SERVEROUT on; EXECUTE usp_bedinfo();</pre>	

Script Output x
<p>Task completed in 0.064 seconds</p> <pre>care center is : 1000 bed number : 2000 bed type id : 0 bed description : low care center is : 1001 bed number : 2001 bed type id : 1 bed description : low care center is : 1002 bed number : 2002 bed type id : 2 bed description : low care center is : 1003 bed number : 2003 bed type id : 3 bed description : full care center is : 1004 bed number : 2004 bed type id : 4 bed description : half PL/SQL procedure successfully completed.</pre>

Store Procedure # 6

```
create or replace PROCEDURE usp_insuranceinfo
IS
v_papersonid NUMBER(4);
v_personid NUMBER(4);
v_insurancecompanyname VARCHAR2(50);

CURSOR cur_insurance is

select patient.papersonid,patient_insurance_coverage.personid, insurance_company.insurancecompanyname
from patient, insurance_company, patient_insurance_coverage
where patient.papersonid = patient_insurance_coverage.personid
and insurance_company.insurancecompanyid = patient_insurance_coverage.insurancecompanyid
and insurance_company.insurancecompanyname = 'metroplus';

BEGIN
OPEN cur_insurance;
LOOP
FETCH cur_insurance into v_papersonid, v_personid, v_insurancecompanyname;
EXIT WHEN cur_insurance%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('patientid is : ' ||v_papersonid);
DBMS_OUTPUT.PUT_LINE('person id is : ' ||v_personid);
DBMS_OUTPUT.PUT_LINE('insurance company name is : ' ||v_insurancecompanyname);

END LOOP;
CLOSE cur_insurance;
END usp_insuranceinfo;
```

The screenshot displays a SQL development tool interface. At the top, there is a toolbar with various icons. Below it, a 'Worksheet' tab is active, showing a query editor with the following SQL code:

```
SET SERVEROUT on;
EXECUTE usp_insuranceinfo();
```

Below the query editor, a 'Script Output' window is open, showing the results of the execution. It indicates that the task was completed in 0.034 seconds. The output consists of six lines of text, grouped into three pairs, each representing a row from the cursor:

```
patientid is : 1235
person id is : 1235
insurance company name is : metroplus
patientid is : 1234
person id is : 1234
insurance company name is : metroplus
```

At the bottom of the output window, a message states: 'PL/SQL procedure successfully completed.'

Stored Procedure # 7

```
create or replace PROCEDURE usp_patientpayment
IS
v_papersonid number(4);
v_owername varchar2(50);
v_first varchar2(50);
v_creditcardnumber number(16);

CURSOR cur_ppayment is

select patient.papersonid, credit_card.owername, person.first, credit_card.creditcardnumber
from patient, patient_credit_card, credit_card, person
where patient.papersonid = patient_credit_card.papersonid
and patient_credit_card.creditcardnumber = credit_card.creditcardnumber
and patient.papersonid = person.personid
and patient.papersonid = '1237';

BEGIN
OPEN cur_ppayment;
LOOP
FETCH cur_ppayment into v_papersonid, v_owername, v_first, v_creditcardnumber;

EXIT WHEN cur_ppayment%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('person id is : ' || v_papersonid);
DBMS_OUTPUT.PUT_LINE('owner name is : ' || v_owername);
DBMS_OUTPUT.PUT_LINE('person first name is : ' || v_first);
DBMS_OUTPUT.PUT_LINE('credit card number : ' || v_creditcardnumber);

END LOOP;
CLOSE cur_ppayment;
END usp_patientpayment;
```

The screenshot displays a database query tool interface. At the top, a toolbar contains various icons for file operations, execution, and formatting, with a timer showing 0.048 seconds. Below the toolbar, the 'Query Builder' tab is active, showing a SQL script with two lines: 'SET SERVEROUT on;' and 'EXECUTE usp_patientpayment();'. The 'Script Output' window at the bottom shows the results of the procedure execution, including the person's ID, owner name, first name, and credit card number, followed by a confirmation message that the PL/SQL procedure was successfully completed.

Worksheet Query Builder

```
SET SERVEROUT on;
EXECUTE usp_patientpayment();
```

Script Output x

Task completed in 0.048 seconds

```
person id is : 1237
owner name is : theo
person first name is : corey
credit card number : 1234567895

PL/SQL procedure successfully completed.
```


Stored Procedure # 8

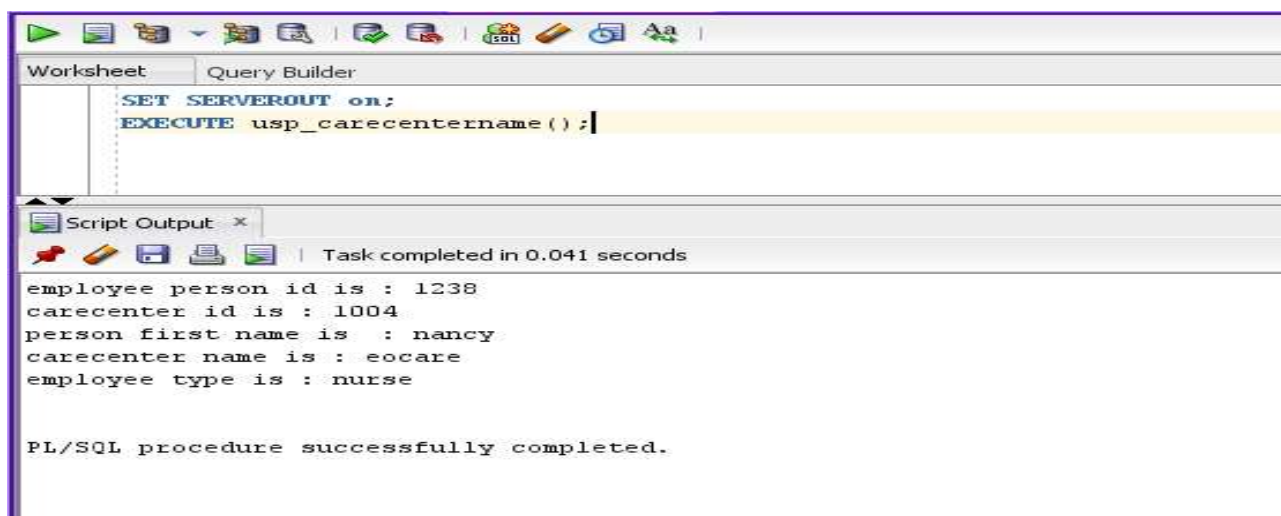
```
create or replace PROCEDURE usp_carecentername
IS
v_empersonid number(4);
v_carecenterid number(4);
v_first varchar2(25);
v_carecentername varchar2(25);
v_employeetype varchar(30);

CURSOR cur_carecenter is
select employee.empersonid, care_center.carecenterid, person.first, care_center.carecentername, employee.employeetype
from employee, care_center, person
where person.personid = employee.empersonid
and care_center.nurseincharge = employee.empersonid
and empersonid = '1238';

BEGIN
OPEN cur_carecenter;
LOOP
FETCH cur_carecenter into v_empersonid, v_carecenterid, v_first, v_carecentername,v_employeetype;

EXIT WHEN cur_carecenter%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('employee person id is : ' ||v_empersonid);
DBMS_OUTPUT.PUT_LINE('carecenter id is : ' ||v_carecenterid);
DBMS_OUTPUT.PUT_LINE('person first name is : ' ||v_first);
DBMS_OUTPUT.PUT_LINE('carecenter name is : ' ||v_carecentername);
DBMS_OUTPUT.PUT_LINE('employee type is : ' ||v_employeetype);
END LOOP;
CLOSE cur_carecenter;
END usp_carecentername;
```



Stored Procedure # 9

```
create or replace PROCEDURE usp_dateadmittedinfo
IS
v_papersonid number(4);
v_first varchar2(25);
v_dateadmitted date;
v_datedischarged date;

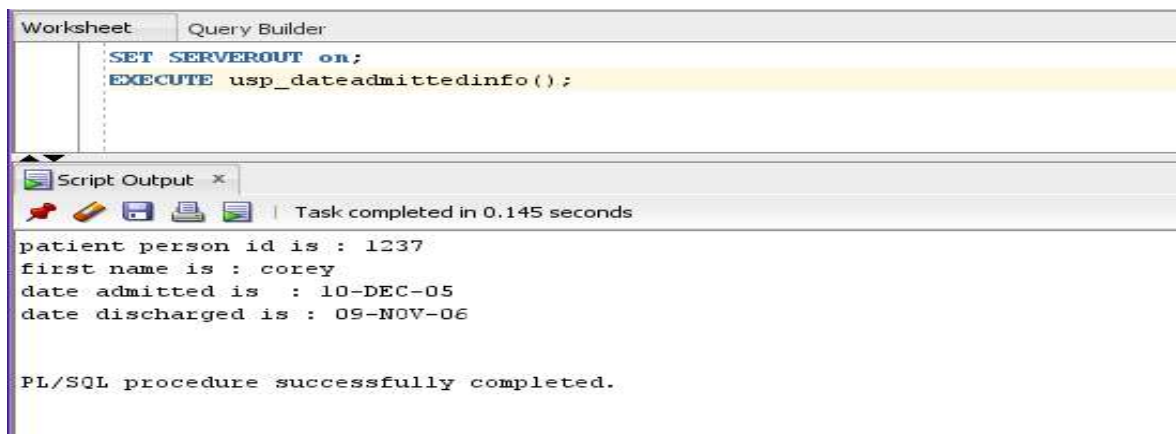
CURSOR cur_date is

select patient.papersonid, person.first, resident_patient.dateadmitted, resident_patient.datedischarged
from person, patient, resident_patient
where person.personid = patient.papersonid
and patient.papersonid = resident_patient.papersonid
and patient.papersonid = '1237';
|
BEGIN
OPEN cur_date;
LOOP
FETCH cur_date into v_papersonid, v_first, v_dateadmitted, v_datedischarged;

EXIT WHEN cur_date%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('patient person id is : ' ||v_papersonid);
DBMS_OUTPUT.PUT_LINE('first name is : ' ||v_first);
DBMS_OUTPUT.PUT_LINE('date admitted is : ' ||v_dateadmitted);
DBMS_OUTPUT.PUT_LINE('date discharged is : ' ||v_datedischarged);

END LOOP;
CLOSE cur_date;
END usp_dateadmittedinfo;
```



The screenshot shows a database query tool interface. At the top, there are tabs for 'Worksheet' and 'Query Builder'. Below the tabs, the SQL code is displayed in a text area. The code consists of two lines: 'SET SERVEROUT on;' and 'EXECUTE usp_dateadmittedinfo();'. The second line is highlighted in yellow. Below the code area, there is a 'Script Output' window. It shows the results of the procedure execution: 'patient person id is : 1237', 'first name is : corey', 'date admitted is : 10-DEC-05', and 'date discharged is : 09-NOV-06'. At the bottom of the output window, it states 'PL/SQL procedure successfully completed.'.

```
Worksheet Query Builder
SET SERVEROUT on;
EXECUTE usp_dateadmittedinfo();

Script Output x
Task completed in 0.145 seconds

patient person id is : 1237
first name is : corey
date admitted is : 10-DEC-05
date discharged is : 09-NOV-06

PL/SQL procedure successfully completed.
```

Stored Procedure # 10

```
create or replace PROCEDURE usp_employeeename
IS
v_empersonid number(4);
v_carecenterid number(4);
v_first varchar2(25);
v_employeetype varchar(30);
v_carecentername varchar2(25);

CURSOR cur_employee is
select employee.empersonid, care_center.carecenterid, person.first, employee.employeetype, care_center.carecentername
from employee, care_center, person
where person.personid = employee.empersonid
and care_center.nurseincharge = employee.empersonid
and care_center.carecentername = 'ercare';

BEGIN
OPEN cur_employee ;
LOOP
FETCH cur_employee into v_empersonid, v_carecenterid, v_first,v_employeetype, v_carecentername;

EXIT WHEN cur_employee%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('employee id is : ' ||v_empersonid);
DBMS_OUTPUT.PUT_LINE('care center id is : ' ||v_carecenterid);
DBMS_OUTPUT.PUT_LINE('first name is : ' ||v_first);
DBMS_OUTPUT.PUT_LINE('employee type is : ' ||v_employeetype);
DBMS_OUTPUT.PUT_LINE(' care center name is ' ||v_carecentername);
END LOOP;
CLOSE cur_employee;
END usp_employeeename;
```

```
set SERVEROUT on;
EXECUTE usp_employeenamename();
```

Script Output x

Task completed in 0.069 seconds

PL/SQL procedure successfully completed.

employee id is : 1235
care center id is : 1001
first name is : allen
employee type is : doctor
care center name is ercare

PL/SQL procedure successfully completed.

Conclusion

With the help of business reports and store procedure, it helps to access different important data's in an organization. Thought planning, analyzing, designing, development & operation this project was completed. There were business reports that had to be created In-order to find the information and data about the EZMedicalGroup. And with the help of Store Procedure, we used the 10 business reports and executed them in a proper format.