PROJECT REPORT

Course code: CSE 312

Course title: Database Management System Lab

Project title: Super shop management system.

Submitted to

Md. Zabirul Islam

Lecturer, Dept. of CSE

Daffodil International University

Submitted by

Abdur Rahman Apu

ID: 191-15-13025

Sec- O

Department of CSE,

Daffodil International University

Date of submission: 14th April, 2021

Project title: Super shop management system

Introduction:

This project is about super shop management system. Super shop is a very large shop where we can find various products in one shop. It is also known as a grocery store and it is a self-service store offering a wide variety of food and household merchandise and organized into departments. So, it is necessary to track products which is available or not in their shop. By using database management system, it is very easy to find products.

Oracle database is used for this project. Database provides security and easy to store and access information. The main reason to design this project to store –

- customer information,
- purchase information,
- shop's employee's information,
- product's information,
- stock information,
- supplier's and his/her company's information.

This project has 7 tables. They are –

- Customer table,
- Employee table,
- Bill_payment table,
- Product table,
- Stock table,
- Supplier table,
- Company table.

Here supplier's and company's information's will be stored. Because If a customer complains about any product, then shop owner can contact with them by getting their information which is stored in the database. Stock table can be used for tracking product which is available or not in their shop.

SQL command:

```
--RENAME COLUMN NAME OF EMPLOYEE
alter table Employee rename column employee_id to emp_id;
desc Employee;
alter table Employee rename column employee_name to emp_name;
desc Employee;

--MODIFY COLUMN NAME
alter table Employee modify salary number(15);
describe Employee;



--ADD AND DELETE COLUMN

alter table Supplier add salary number(10);
desc Supplier;
alter table Supplier drop column salary;
desc Supplier;



/*  ------------------EMPLOYEE  TABLE----------------------
   (1)SELECT,WHERE STATEMENT
```

```
    (2)WORK ON A COLUMN
    (3) ARITHMETIC OPERATOR (>,<)
    (4) LOGICAL OPERATOR (AND,OR,NOT),
    (5) between, not between,
    (6)in, not in
    ----------------------------------------------------------
*/
select emp_name,salary from Employee;
select emp_name, (salary/2) as update_salary from Employee;
select emp_name,salary from Employee where job_title='Manager';
select emp_name,salary from Employee where job_title='Security Guard';


select emp_name,salary from Employee where salary>8000 or salary<15000;
select emp_name,salary from Employee where salary between 8000 and 15000;
select emp_name,salary from Employee where salary not between 8000 and
15000;

select emp_name,salary from Employee where salary in (8000,40000);

select emp_name,salary from Employee where salary not in (8000,40000);

--AGGREGIATE FUNCTIONS
select count(emp_id) from Employee;
select max(salary) from Employee;
select min(salary) from Employee;
```

```
select sum(salary) from Employee;
select avg(salary) from Employee;

select emp_name,salary from Employee;

--UPDATE STATEMENT
update Employee set salary=15000 where job_title='Worker';

select emp_name,salary from Employee;


--ORDER BY ASCENDING AND DESCENDING
select emp_name,salary from Employee order by salary;
select emp_name,salary from Employee;
select emp_name,salary from Employee order by salary desc;
select emp_name,salary from Employee;
select emp_name,salary from Employee order by emp_name desc,salary asc;
select emp_name,salary from Employee order by salary asc,emp_name desc;


--LIKE STATEMENT
desc Customer;
select * from Customer where customer_id='C-121';
select customer_name,address from Customer where gender='Male';
select customer_name,address from Customer where gender='Male' and address
like '%Jatrabari%';
```

```
select customer_name,address from Customer where gender='Male' and address like '%Dhaka';

select customer_name,address from Customer where gender='Male' and address like '698/1%';


--NOT OPERATOR

desc Product;

select product_name,price from Product where not price>60;


--COUNT(*) AND COUNT(DISTINCT COLUMN NAME)

desc Bill_payment;

select * from Bill_payment where payment_method='Cash';

select count(*) from Bill_payment;

select count(distinct customer_id) from Bill_payment;



--GROUP BY AND HAVING

select customer_id,sum(amount) from Bill_payment group by customer_id;

select customer_id,sum(amount) from Bill_payment group by customer_id having sum(amount)>1000;

select customer_id,sum(amount) from Bill_payment group by customer_id having sum(amount)<1000;


--NESTED QUERY

select customer_name from Customer where customer_id in (select customer_id from Bill_payment);
```

```
select product_name,price from Product where not product_id in (select
product_id from Stock);

select supplier_name from Supplier where supplier_id in (select supplier_id from
Product);

select company_name from Company where company_id in (select company_id
from Supplier where supplier_id in (select supplier_id from Product));

select company_name from Company where company_id in (select company_id
from Supplier where supplier_id in (select supplier_id from Product where
price>100));


--SET OPERATIONS

select customer_id from Customer union select customer_id from Bill_payment;

select customer_id from Customer intersect select customer_id from Bill_payment;

select customer_id from Customer minus select customer_id from Bill_payment;



/*  .....JOIN.....*/



--CONDITION LIKE THETHA JOIN

select c.customer_name,c.gender,b.buy_date,b.amount from Customer
c,Bill_payment b where c.customer_id=b.customer_id;


--THETA JOIN

select c.customer_name,d.payment_method from Customer c join Bill_payment d
on c.customer_id=d.customer_id;
```

```
select c.customer_name,d.payment_method from Customer c join Bill_payment d
using(customer_id);


--NATURAL JOIN

select c.customer_name,d.payment_method from Customer c natural join
Bill_payment d;



--CROSS JOIN

select * from Stock cross join Product;


select * from Stock s cross join Product p where s.product_id=p.product_id;


--FULL OUTER JOIN

select p.product_name,s.quantity from Stock s full outer join product p on
s.product_id=p.product_id;


--LEFT OUTER JOIN

select  p.product_name,s.quantity from  product p left outer join Stock s on
s.product_id=p.product_id;


--RIGHT OUTER JOIN

select s.quantity,p.product_name from Stock s right outer join product p on
s.product_id=p.product_id;
```

PLSQL command:

```
--FIND MAXIMUM AND MINIMUM CUSTOMER ID
```

```
set serveroutput on
declare
    max_customer_id Customer.customer_id%type;
    min_customer_id Customer.customer_id%type;
begin
    select max(customer_id) into max_customer_id from Customer;
    select min(customer_id) into min_customer_id from Customer;

    --PRINT OUTPUT
    dbms_output.put_line('The maximum customer id is:'|| max_customer_id);
    dbms_output.put_line('The minimum customer id is:'|| min_customer_id);
end;
/
```

--FIND THE NAME OF THE EMPLOYEE WHO GET MAXIMUM SALARY.

```
set serveroutput on
declare
    s Employee.salary%type;
    nam Employee.emp_name%type;
begin
    select max(salary) into s from Employee;
    select emp_name into nam from Employee where salary=s;
```

```
    dbms_output.put_line('Employee name '||nam||' who get maximum salary= '||s);
end;
/



/*
DISPLAY THE COMPANY NAME WHERE A SUPPLIER WORK.

*/
set serveroutput on
declare
    comp_name Company.company_name%type;
    supp_name Supplier.supplier_name%type :='Md. Mujib';
begin
    select company_name into comp_name from Supplier,Company where
Supplier.company_id=Company.company_id and
supp_name=Supplier.supplier_name;
    dbms_output.put_line(supp_name || ' Works in '||comp_name ||' Company');
end;
/


/* IF ELSIF ELSE

NOW, LETS CHECK YOUR SEARCH PRODUCTS HAS ANY DISCOUNT OR
NOT.
```

CONDITIONS:

1. PRICE >= 2000 THEN 20% OFFER
2. PRICE >=1000 BUT LESS THEN FIRST CONDITION ,THEN 15% OFFER
3. PRICE >=500 BUT LESS THEN SECOND CONDITION ,THEN 10% OFFER
4. PRICE >=100 BUT LESS THEN THIRD CONDITION ,THEN 5% OFFER
5. PRICE <100 BUT LESS THEN FOURTH CONDITION ,THEN NO OFFER

*/

```
set serveroutput on
declare
   actual_price Product.price%type;
   discount_price Product.price%type;
   product_search varchar2(50);

begin
    product_search := '&product_search';
   select price into actual_price from Product where product_name like
product_search;
   if actual_price >= 2000 THEN
       discount_price := actual_price- (actual_price*0.20);
       dbms_output.put_line('Your Search product is ='|| product_search);
       dbms_output.put_line('ITs actual price is = '|| actual_price || ' Your discount
price= '|| discount_price);
```

```
    elsif actual_price >= 1000 and actual_price < 2000 THEN
        discount_price := actual_price- (actual_price*0.15);
        dbms_output.put_line('Your Search product is ='|| product_search);
        dbms_output.put_line('ITs actual price is = '|| actual_price || ' Your discount price= '|| discount_price);

    elsif actual_price >= 500 and actual_price < 1000 THEN
        discount_price := actual_price- (actual_price*0.10);
        dbms_output.put_line('Your Search product is ='|| product_search);
        dbms_output.put_line('ITs actual price is = '|| actual_price || ' Your discount price= '|| discount_price);

    elsif actual_price >= 100 and actual_price < 500 THEN
        discount_price := actual_price- (actual_price*0.05);
        dbms_output.put_line('Your Search product is ='|| product_search);
        dbms_output.put_line('ITs actual price is = '|| actual_price || ' Your discount price= '|| discount_price);

    else
        dbms_output.put_line('SORRY,NO OFFERS AVAILABLE RIGHT NOW FOR YOUR PRODUCT.');
    end if;
    EXCEPTION
        WHEN others THEN
            DBMS_OUTPUT.PUT_LINE ('THIS PRODUCT NOT FOUND');
end;
/
```

```
/* LOOP WITH CURSOR
*****************************************
SHOW SUPPLIER NAME WITH HIS COMPANY NAME
*****************************************

*/
set serveroutput on
declare
   CURSOR name_cur is
   select c.company_name,s.supplier_name from Company c,Supplier s where c.company_id=s.company_id;

   show name_cur%rowtype;
begin
   open name_cur;
   dbms_output.put_line('**************   *************');
   dbms_output.put_line('SUPPLIER NAME   COMPANY NAME');
   dbms_output.put_line('**************   *************');
   LOOP
      fetch name_cur into show;
      exit when name_cur%rowcount>6;
      dbms_output.put_line(show.supplier_name||'      '||show.company_name);
   end loop;
   close name_cur;
```

```
end;
/


--WHILE LOOP


--FIND PRODUCT NAME WITH ITS QUANTITY WHICH IS AVAILABLE IN
STOCK.


SET SERVEROUTPUT ON
DECLARE
   CURSOR hold is
   select p.product_name,s.quantity from Product p join Stock s on
p.product_id=s.product_id;
   show hold%rowtype;
BEGIN
   OPEN hold;


   dbms_output.put_line('**************        ************');
   dbms_output.put_line('PRODUCT NAME        QUANTITY');
   dbms_output.put_line('**************        ************');
   while hold%rowcount<10
   LOOP
      FETCH hold INTO show;
      DBMS_OUTPUT.PUT_LINE(show.product_name||'          '||show.quantity);
   END LOOP;
   close hold;
```

```
END;
/

-- FOR LOOP FOR PRINTING EMPLOYEE NAME AND SALARY.



set serveroutput on
declare
   counter    NUMBER(2);
   cursor hold is
   select * from Employee;
   data hold%rowtype;
begin
   open hold;
   dbms_output.put_line('**************         *************         **********
***********');
   dbms_output.put_line('EMPLOYEE NAME         EMPLOYEE ID
SALARY              JOBTITLE');
   dbms_output.put_line('**************         *************         **********
***********');
   for counter in 1..6
   loop
      fetch hold into data;
      dbms_output.put_line(data.emp_name||'              '||data.emp_id||'
'||data.salary||'              '||data.job_title);
```

```
    end loop;
    close hold;


end;
/
```

Conclusion:

Database is an important part for any company or organization. Company needs to store their information into a safe place. And, database provide security, easy to store and access information capability. Without database it is quite difficult to store data into a safe place and manage data is also difficult. Super shop is a most important part in our daily life. People are more easily shopping through them. By using this super shop management system, any super shop can store customer, employee, supplier, company, product’s information and can manage their data very easily and can save their time and money also.