

**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.