الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونيبرسيتي إسلام انتارابغسا مليسيا
(Company No. 101067-P)
Garden of Knowledge and Virtue
PREMIER INTERNATIONAL ISLAMIC RESEARCH UNIVERSITY

SEMESTER 2, 2018/2019

SECTION 1

COURSE: INFO 2103 Database Programming

GROUP PROJECT REPORT

Title : "Warranty System"

LECTURER

Dr. ZAINATUL SHIMA ABDULLAH

PREPARED BY:

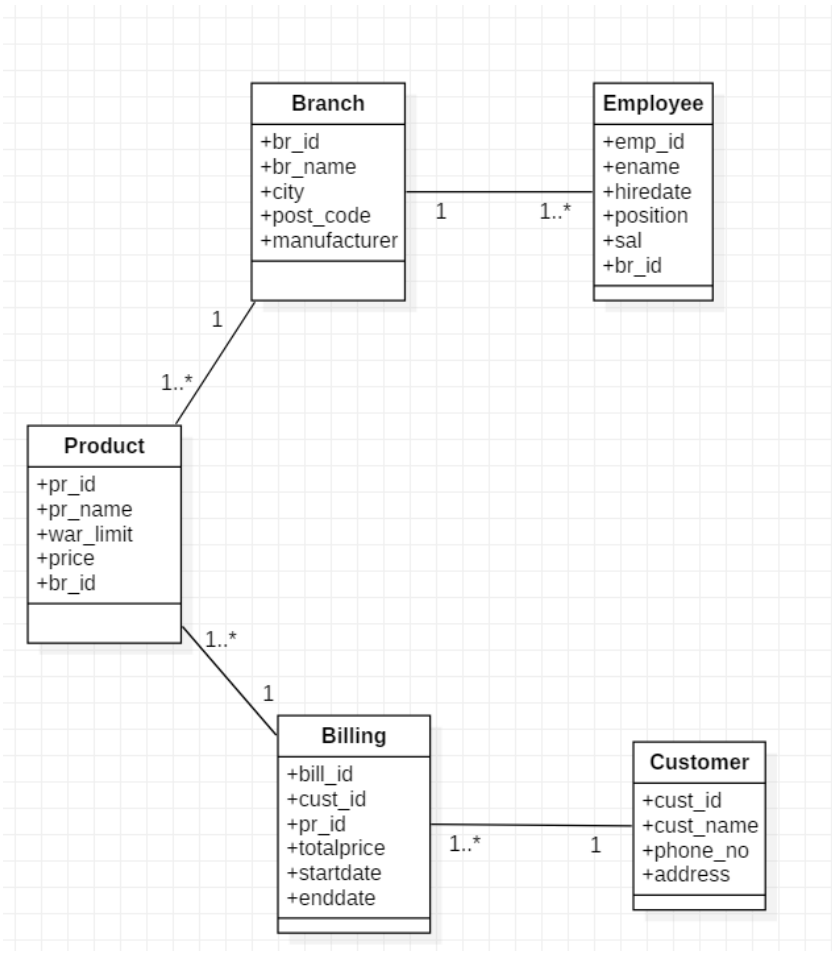
NAME	MATRIC NO.
Raisa Zesira	1813024
Sanjir Inam Salsabil	1431937
Hannah Binti Huda	1814022
Tengku Ebar Syarif Hamzah	1628959
Muhammad Ersyad Ghifari	1623143

1.0 GROUP PROJECT SCENARIO

Warranty is one privilege that one product can offer to their customers to show that the product can be repair/exchange for further usage incase the user uncoincidentally break a product. Nowadays, there are a lot of products that offer a various type of warranty based on time bought. These warranty datas need to be monitored by the system for management purposes.

The purpose of our project is to create a database system which can improve the efficiency of a warranty service so that the employees can manage and keep track of the current customer warranty request without missing out any important customer information also for the customer to have view on their warranty issues. Currently our group project cater warranty for 2 (two) devices only, which is laptop and gadgets.

2.0 ENTITY RELATIONSHIP DIAGRAM (ER-DIAGRAM)



3.0 DATA DICTIONARY

CUSTOMER

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
CUST_ID	VARCHAR2	4	NO (PK)
CUST_NAME	VARCHAR2	30	NO
PHONE_NO	NUMBER	15	NO
ADDRESS	VARCHAR2	30	YES

EMPLOYEE

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
EMP_ID	VARCHAR2	4	NO (PK)
ENAME	VARCHAR2	30	NO
hiredate	DATE		NO
POSITION	VARCHAR2	15	NO
SAL	NUMBER	10,2	NO
BR_ID	NUMBER	4	NO (FK)

PRODUCT

ATTRIBUTE NAME	DATA TYPE	SIZE	NULL
PR_ID	VARCHAR2	4	NO (PK)
PR_NAME	VARCHAR2	30	YES
WAR_LIMIT	DATE	5	NO
BR_ID	VARCHAR2	4	NO (FK)
PRICE	NUMBER	10	NO

BRANCH

ATTRIBUTE NAME	**DATA TYPE**	**SIZE**	**NULL**
BR_ID	VARCHAR2	4	NO (PK)
BR_NAME	VARCHAR2	30	YES
CITY	VARCHAR2	30	YES
POST_CODE	NUMBER	5	NO
MANUFACTURER	VARCHAR2	30	NO

BILLING

ATTRIBUTE NAME	**DATA TYPE**	**SIZE**	**NULL**
BILL_ID	VARCHAR2	5	NO (PK)
CUST_ID	VARCHAR2	4	NO (FK)
PR_ID	VARCHAR2	4	NO (FK)
TOTALPRICE	NUMBER	10,2	NO
STARTDATE	DATE		NO
ENDDATE	DATE		YES

4.0 SQL SYNTAX (DDL & DML)

Logical Database Design

Branch (Branch ID , Branch Name, City, Postcode , Manufacturer)
Primary Key - Branch ID

Employee (Employee ID , Employee Name , Date of Hire, Job Position, Salary, Branch ID)
Primary Key - Employee ID
Foreign Key - Branch ID References Branch (Branch ID)

Product (Product ID, Product Name , Warranty Limit , Branch ID , Price)
Primary Key - Product ID
Foreign Key - Branch ID References Branch (Branch ID)

Customer (Customer ID , Customer Name , Phone Number , Address)
Primary Key - Customer ID

Billing (Billing ID , Customer ID , Product ID , Total Price , Date of Start , Date of End)
Primary Key - Billing ID
Foreign Key - Customer ID References Customer (Customer ID)
Foreign Key - Product ID References Customer (Customer ID)

Physical Database Design

```
Create Table branch
(br_id varchar(4),
br_name varchar2(15),
city varchar2(30),
post_code number(5),
manufacturer   varchar2(30),
constraint branch_br_id_pk primary key (br_id));

create table employee
(emp_id varchar(4),
ename varchar2(30) not null,
hiredate date not null,
position varchar2(15) not null,
sal number(10,2) not null,
br_id varchar(4),
constraint EMP_emp_id_PK primary key (emp_id),
constraint EMP_br_id_FK foreign key (br_id) references branch(br_id));

create table product
(pr_id       varchar2(4),
pr_name      varchar2(30),
war_limit    date not null,
br_id           varchar2(4),
price           number (10) not null,
constraint pk_pr_id primary key (pr_id),
constraint fk_pr_br_id foreign key (br_id) references BRANCH (br_id));

create table customer
(cust_id  varchar(4),
cust_name  varchar2(30),
phone_no  number(15),
address  varchar2(30),
constraint customer_cust_id_pk primary key (cust_id));

CREATE TABLE billing
(bill_id varchar(5),
cust_id varchar(4),
pr_id  varchar(4),
totalprice  number(10,2),
startdate date,
enddate date,
constraint billing_bill_id primary key (bill_id),
constraint billing_cust_id_FK foreign key (cust_id) references customer(cust_id),
constraint billing_pr_id_FK foreign key (pr_id) references product(pr_id));
```

5.0 SQL SYNTAX (INSERT QUERY)

1. Branch table

```
insert into Branch values ('B123', 'KL_branch', 'Kuala Lumpur', 1423  , ‘ Apple’);
insert into Branch values ('B124', 'SN_branch', 'Selangor', 5317 , ‘acer’ );
insert into Branch values ('B125', 'PG_branch', 'Penang', 1357 , ‘Xiaomi’);
insert into Branch values ('B126', 'SW_branch', 'Serawak', 6879 , ‘oppo’);
insert into Branch values ('B127', 'SB_branch', 'Sabah', 4567 , ‘hp’);
insert into Branch values ('B128', 'JK_branch', 'Jakarta', 6575 , ‘OnePlus’);
insert into Branch values ('B129', 'JG_branch', 'Jogjakarta', 9827 , ‘asus’);
insert into Branch values ('B130', 'DH_branch', 'Dhaka', 5578 , ‘Samsung’);
insert into Branch values ('B131', 'CT_branch', 'Chittagong', 8966  , ‘lenovo’);
insert into Branch values ('B132', 'SG_branch', 'Singapore', 2767 , ‘Motorola’);
```

2. Employee table

```
insert into employee values ('E900', 'Ahmad Zaidi', to_date('1-NOV-2015'), 'Salesperson',
2000, 'B123');
insert into employee values ('E901', 'Aiman Hakim', to_date('1-MAY-2015' , 'DD-MM-YYYY'),
'Manager', 3000, 'B124');
insert into employee values ('E902', 'Linda Mraz', to_date('9-JUN-2012'), 'Salesperson',
2450, 'B125');
insert into employee values ('E903', 'Nur Amirah', to_date('2-APR-2012'), 'Salesperson',
2975, 'B126');
insert into employee values ('E904', 'Jason Lee', to_date('13-JUN-2016'), 'Salesperson',
3000, 'B127');
insert into employee values ('E905', 'Jackson Wang', to_date('3-DEC-2017'), 'Manager',
5000, 'B128');
insert into employee values ('E906', 'Hanbin Lee', to_date('17-DEC-2017'), 'Salesperson',
3000, 'B129');
insert into employee values ('E907', 'Mark Tuan', to_date('20-JAN-2018'), 'Manager', 5000,
'B130');
insert into employee values ('E908', 'Ramasamy', to_date('22-JAN-2018'), 'Salesperson',
1250, 'B131');
insert into employee values ('E909', 'Muhammad Ali', to_date('28-SEP-2018'), 'Salesperson',
1400, 'B132' );
insert into employee values ('E910', 'Adam Alias',to_date('28-FEB-2018'), 'Salesperson' ,
2000 , 'B123');
insert into employee values ('E911', 'Liza Johnson', to_date('22-MAR-2015'), 'Salesperson' ,
1500 , 'B124');
insert into employee values ('E912', 'Fuad Fauzi',to_date('11-JUL-2016'), 'Manager' , 6000 ,
'B125');
insert into employee values ('E913', 'Julie Lee',to_date('28-MAY-2015'), 'Salesperson' ,
2100 , 'B126');
insert into employee values ('E914', 'David Ford', to_date('14-FEB-2014'), 'Salesperson' ,
1750, 'B127');
```

3. Product table

```
insert into product  values( 'P134' , 'Apple X',    to_date('1-1-2020','dd-mm-yyyy')  , 'B123' ,
2000 );
insert into product   values(  'P135', 'Predator',   to_date('2-2-2020','dd-mm-yyyy'),
'B124',1500);
insert into product   values(   'P136', 'Redmi 5A',   to_date('3-3-2017','dd-mm-yyyy'), 'B125',
300);
insert into product   values(   'P137', 'OPPO F5',  to_date('4-4-2016','dd-mm-yyyy'), 'B126',
100);
insert into product   values(   'P138', 'Pavilion 15',   to_date('5-5-2021','dd-mm-rr'), 'B127',
1000);
insert into product values(  'P139', 'OnePlus 6T',      to_date('6-6-2021','dd-mm-yyyy'), 'B128',
900);
insert into product values(   'P140', 'Zenfone Max Pro',   to_date('7-7-2018','dd-mm-yyyy'),
'B129', 550);
insert into product values(  'P141', 'Samsung S10',   to_date('8-8-2022','dd-mm-yyyy'), 'B130',
800);
insert into product  values(   'P142', 'Z6 Pro',    to_date('9-9-2015','dd-mm-yyyy'), 'B131',
700);
insert into product  values(  'P143', 'G7 Play',    to_date('10-10-2022','dd-mm-yyyy'), 'B132',
600);
insert into product values(  'P144', 'Samsung A50',   to_date('10-6-2021','dd-mm-yyyy'),
'B130', 900);
insert into product values(   'P145', 'Zenfone 4',   to_date('7-3-2021','dd-mm-yyyy'), 'B129',
500);
insert into product values(  'P146', 'Samsung Fold',  to_date('1-8-2022','dd-mm-yyyy'), 'B130',
1000);
insert into product  values(   'P147', 'Xiaomi Pocophone',    to_date('9-9-2022','dd-mm-yyyy'),
'B125', 500);
insert into product  values(  'P148', 'OnePlus 5T',    to_date('10-1-2019','dd-mm-yyyy'),
'B128', 800);
```

4. Customer table

```
insert into customer values ('C111', 'Nur Nabilah', 0178270305, 'No 25, Jalan Tasek 9');
insert into customer values ('C112', 'Nabihah Kassim', 0123456789, 'No 9, Jalan
Rambutan');
insert into customer values ('C222', 'Mohamed Fawzy', 0145672804, '22 Deer Rd');
insert into customer values ('C234', 'Mohd Syahmi', 0111118239, 'Lot 1228 Jalan Nam
Heng');
insert into customer values ('C356', 'Lee Jing Young', 0182736228, '99 Main St');
insert into customer values (‘C377’,’Ada Wong’, 0135579924, ‘11 Sesame St’);
Insert into customer values (‘C555’ , ‘Claire Reinhardt’ , 0152293456 , ‘12 Sesame St’);
Insert into customer values (‘C666’ , ‘Bridgette White’ , 0124456332 , ‘30 Main St’);
insert into customer values (‘C442’ , ‘Pharah Tamriel’ , 0145562785 , ‘122 Moon St’);
insert into customer values (‘C001’ , ‘Tom Hawks’ , 0154492341 , ‘61 Sun St’);
Insert into customer values (‘C022’ , ‘Leon Kennedy’ , 0132212334 , ‘15 Sun St’);
```

```
insert into customer values (‘C812’ , ‘Joseph Oda’ , 0154432122 , ‘40 Sesame St’);
insert into customer values (‘C400’ , ‘Nicolla Bellic’ , 0154487001 , ‘70 Sesame St’);
insert into customer values (‘C096’ , ‘Hamidi Hamzah’ , 0132267901 , ‘01 Matahari St’);
insert into customer values (‘C123’ , ‘Anthony Addams’ , 0157751342 , ‘80 Matahari St’);
insert into customer values ('C467', 'Adam Alias', 0137486078, 'No 56, Jalan Suria');
insert into customer values ('C888', 'Liza Johnson', 0197465738, 'No 1990 Jalan Permata');
insert into customer values ('C902', 'Fuad Fauzi', 0122373988, '163 Main Rd');
insert into customer values ('C129', 'Julie Lee', 0129945679, '56 Cover Road Jr');
insert into customer values ('C976', 'David Ford', 0145678292, '16 Arryll Street');
```

5. Billing table

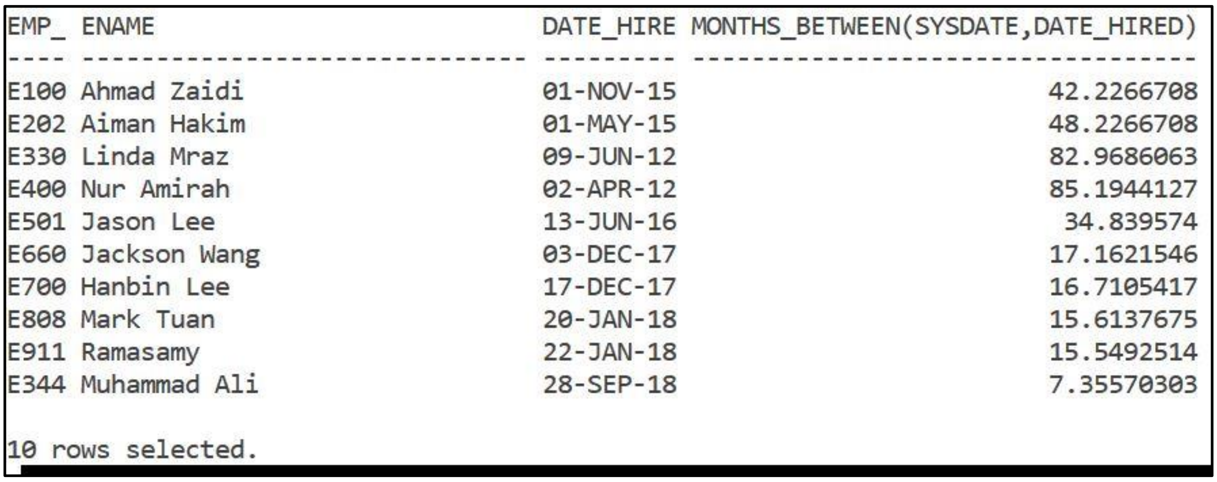
```
Insert into billing values ('BL100' , 'C100' , 'P134' , 0 ,  to_date('1-1-2018','dd-mm-yyyy'),
to_date('2-2-2018','dd-mm-yyyy'));
Insert into billing values ('BL101' , 'C101' , 'P135' , 0 ,  to_date('5-3-2018','dd-mm-yyyy'),
to_date('7-4-2018','dd-mm-yyyy'));
Insert into billing values ('BL102' , 'C102' , 'P136' , 300 ,  to_date('5-7-2018','dd-mm-yyyy'),
to_date('17-8-2018','dd-mm-yyyy'));
Insert into billing values ('BL103' , 'C103' , 'P137' , 100 ,  to_date('5-3-2018','dd-mm-yyyy'),
to_date('10-4-2018','dd-mm-yyyy'));
Insert into billing values ('BL104' , 'C104' , 'P138' , 0 ,  to_date('15-1-2019','dd-mm-yyyy'),
to_date('19-2-2019','dd-mm-yyyy'));
Insert into billing values ('BL105' , 'C105' , 'P139' , 0 ,  to_date('3-2-2019','dd-mm-yyyy'),
to_date('18-4-2019','dd-mm-yyyy'));
Insert into billing values ('BL106' , 'C106' , 'P140' , 550 ,  to_date('1-8-2018','dd-mm-yyyy'),
to_date('20-8-2018','dd-mm-yyyy'));
Insert into billing values ('BL107' , 'C107' , 'P141' , 0 ,  to_date('9-9-2018','dd-mm-yyyy'),
to_date('30-9-2018','dd-mm-yyyy'));
Insert into billing values ('BL108' , 'C108' , 'P142' , 700 ,  to_date('11-2-2018','dd-mm-yyyy'),
to_date('17-4-2018','dd-mm-yyyy'));
Insert into billing values ('BL109' , 'C109' , 'P143' , 0 ,  to_date('11-2-2018','dd-mm-yyyy'),
to_date('15-4-2018','dd-mm-yyyy'));
Insert into billing values ('BL110' , 'C110' , 'P144' , 0 ,  to_date('15-3-2019','dd-mm-yyyy'),
to_date('17-4-2019','dd-mm-yyyy'));
Insert into billing values ('BL111' , 'C111' , 'P145' , 0 ,  to_date('7-1-2019','dd-mm-yyyy'),
to_date('27-2-2018','dd-mm-yyyy'));
Insert into billing values ('BL112' , 'C112' , 'P146' , 0 ,  to_date('15-2-2019','dd-mm-yyyy'),
to_date('19-3-2019','dd-mm-yyyy'));
Insert into billing values ('BL113' , 'C113' , 'P147' , 0 ,  to_date('5-3-2019','dd-mm-yyyy'),
to_date('29-4-2019','dd-mm-yyyy'));
Insert into billing values ('BL114' , 'C114' , 'P148' , 800 ,  to_date('27-2-2019','dd-mm-yyyy'),
to_date('22-3-2019','dd-mm-yyyy'));
```

6.0 SQL SYNTAX

1. Listing out employee data

```
desc employee;
Select emp_id, ename, hiredate, months_between(sysdate, hiredate)
from employee;
```

Output Sample

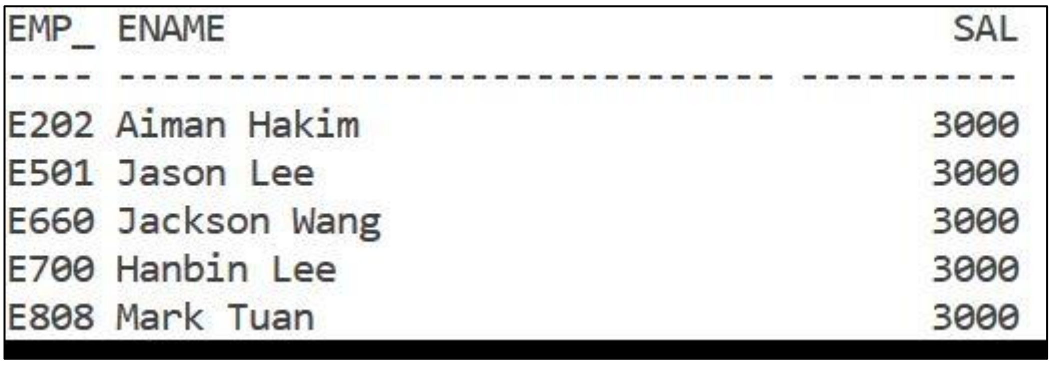


```
EMP_ ENAME                          DATE_HIRE MONTHS_BETWEEN(SYSDATE,DATE_HIRED)
---- ------------------------------ --------- ----------------------------------
E100 Ahmad Zaidi                    01-NOV-15                         42.2266708
E202 Aiman Hakim                    01-MAY-15                         48.2266708
E330 Linda Mraz                     09-JUN-12                         82.9686063
E400 Nur Amirah                     02-APR-12                         85.1944127
E501 Jason Lee                      13-JUN-16                          34.839574
E660 Jackson Wang                   03-DEC-17                         17.1621546
E700 Hanbin Lee                     17-DEC-17                         16.7105417
E808 Mark Tuan                      20-JAN-18                         15.6137675
E911 Ramasamy                       22-JAN-18                         15.5492514
E344 Muhammad Ali                   28-SEP-18                         7.35570303

10 rows selected.
```

2. List out employee that has salary of 3000 without repetition data.

```
select DISTINCT emp_id, ename, sal
from employee
where (sal = 3000);
```

Output Sample



```
EMP_ ENAME                                 SAL
---- ------------------------------ ----------
E202 Aiman Hakim                          3000
E501 Jason Lee                            3000
E660 Jackson Wang                         3000
E700 Hanbin Lee                           3000
E808 Mark Tuan                            3000
```

3. Using Subqueries for Employee Table

```
select emp_id, ename, hiredate

from employee
where hiredate > (select hiredate
                  from employee
                  where ename = 'Jason Lee');
```

4**. Selecting details of employee with related product details that is in KL_Branch**

```
Select employee.emp_id , ename , pr_name , manufacturer , br_name
From branch join product
On branch.br_id = product.br_id
Join employee
On employee.br_id = branch.br_id
Where br_name = 'KL_branch';
```

5. Selecting Details of employee that taking care of products which has standardized price over than 1000

```
Select employee.emp_id , ename , pr_name , price
From employee join product
On employee.br_id  = product.br_id
Where price > 1000;
```

6. Selecting details of employee with their branch then add 15% of salary based on their city and hiredate

```
Select emp_id , ename, br_name , hiredate , (sal + (sal * 0.15) ) AS salary_bonus
From employee join branch
On employee.br_id = branch.br_id
Where hiredate < to_date('2019' , 'yyyy')  And city = 'Selangor';
```

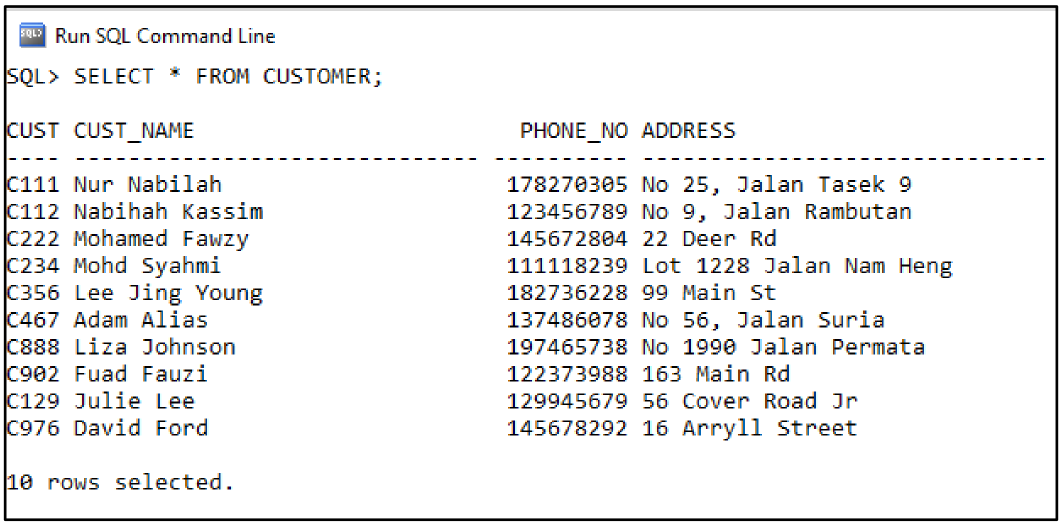
7.0 PROCEDURES

1. **A new procedure to add new customer**

```
CREATE OR REPLACE PROCEDURE addnewcust
(custID customer.cust_id%TYPE,
name customer.cust_name%TYPE,
phoneno customer.phone_no%TYPE)
AS
BEGIN
INSERT INTO customer (cust_id, cust_name, phone_no)
VALUES (custID, name, phoneno);
END;
/
```

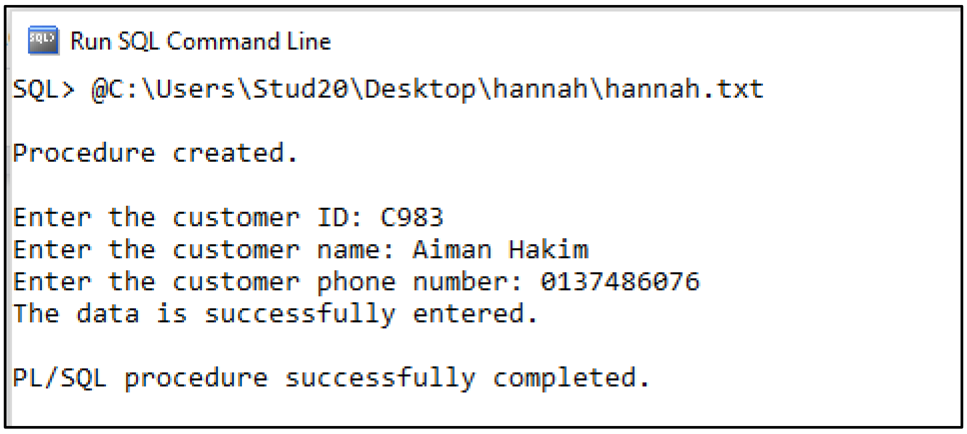
An anonymous block to call procedure *addnewcust*

```
ACCEPT custID PROMPT 'Enter the customer ID: '
ACCEPT name PROMPT 'Enter the customer name: '
ACCEPT phoneno PROMPT 'Enter the customer phone number: '
BEGIN
addnewcust ('&custID', '&name', '&phoneno');
DBMS_OUTPUT.put_line('The data is successfully entered.');
END;
/
```



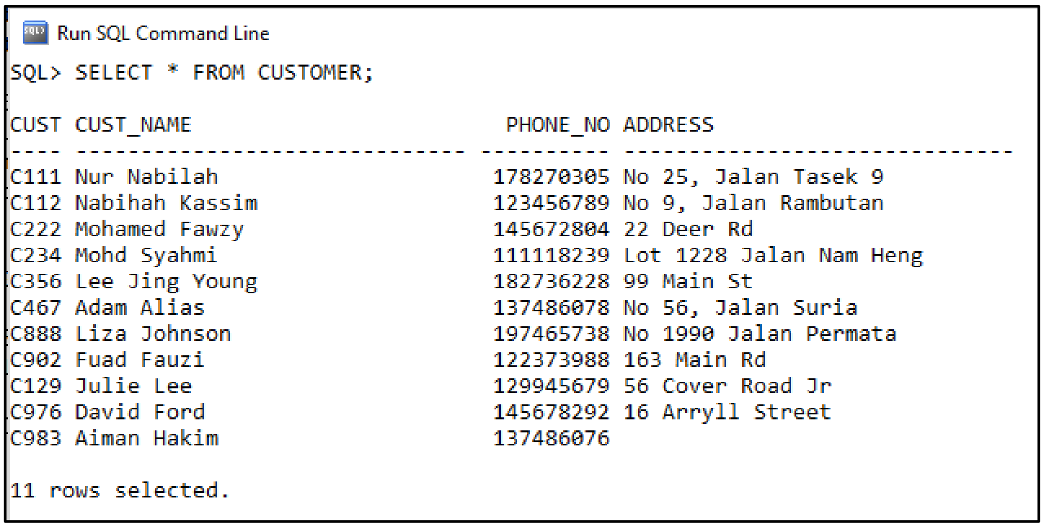
```
SQL> SELECT * FROM CUSTOMER;

CUST CUST_NAME                        PHONE_NO ADDRESS
---- ------------------------------ ---------- ------------------------------
C111 Nur Nabilah                     178270305 No 25, Jalan Tasek 9
C112 Nabihah Kassim                  123456789 No 9, Jalan Rambutan
C222 Mohamed Fawzy                   145672804 22 Deer Rd
C234 Mohd Syahmi                     111118239 Lot 1228 Jalan Nam Heng
C356 Lee Jing Young                  182736228 99 Main St
C467 Adam Alias                      137486078 No 56, Jalan Suria
C888 Liza Johnson                    197465738 No 1990 Jalan Permata
C902 Fuad Fauzi                      122373988 163 Main Rd
C129 Julie Lee                       129945679 56 Cover Road Jr
C976 David Ford                      145678292 16 Arryll Street

10 rows selected.
```

Data before run procedure



```
SQL> @C:\Users\Stud20\Desktop\hannah\hannah.txt

Procedure created.

Enter the customer ID: C983
Enter the customer name: Aiman Hakim
Enter the customer phone number: 0137486076
The data is successfully entered.

PL/SQL procedure successfully completed.
```

Executing the procedure



```
SQL> SELECT * FROM CUSTOMER;

CUST CUST_NAME                        PHONE_NO ADDRESS
---- ------------------------------ ---------- ------------------------------
C111 Nur Nabilah                     178270305 No 25, Jalan Tasek 9
C112 Nabihah Kassim                  123456789 No 9, Jalan Rambutan
C222 Mohamed Fawzy                   145672804 22 Deer Rd
C234 Mohd Syahmi                     111118239 Lot 1228 Jalan Nam Heng
C356 Lee Jing Young                  182736228 99 Main St
C467 Adam Alias                      137486078 No 56, Jalan Suria
C888 Liza Johnson                    197465738 No 1990 Jalan Permata
C902 Fuad Fauzi                      122373988 163 Main Rd
C129 Julie Lee                       129945679 56 Cover Road Jr
C976 David Ford                      145678292 16 Arryll Street
C983 Aiman Hakim                     137486076

11 rows selected.
```

Data after run procedure

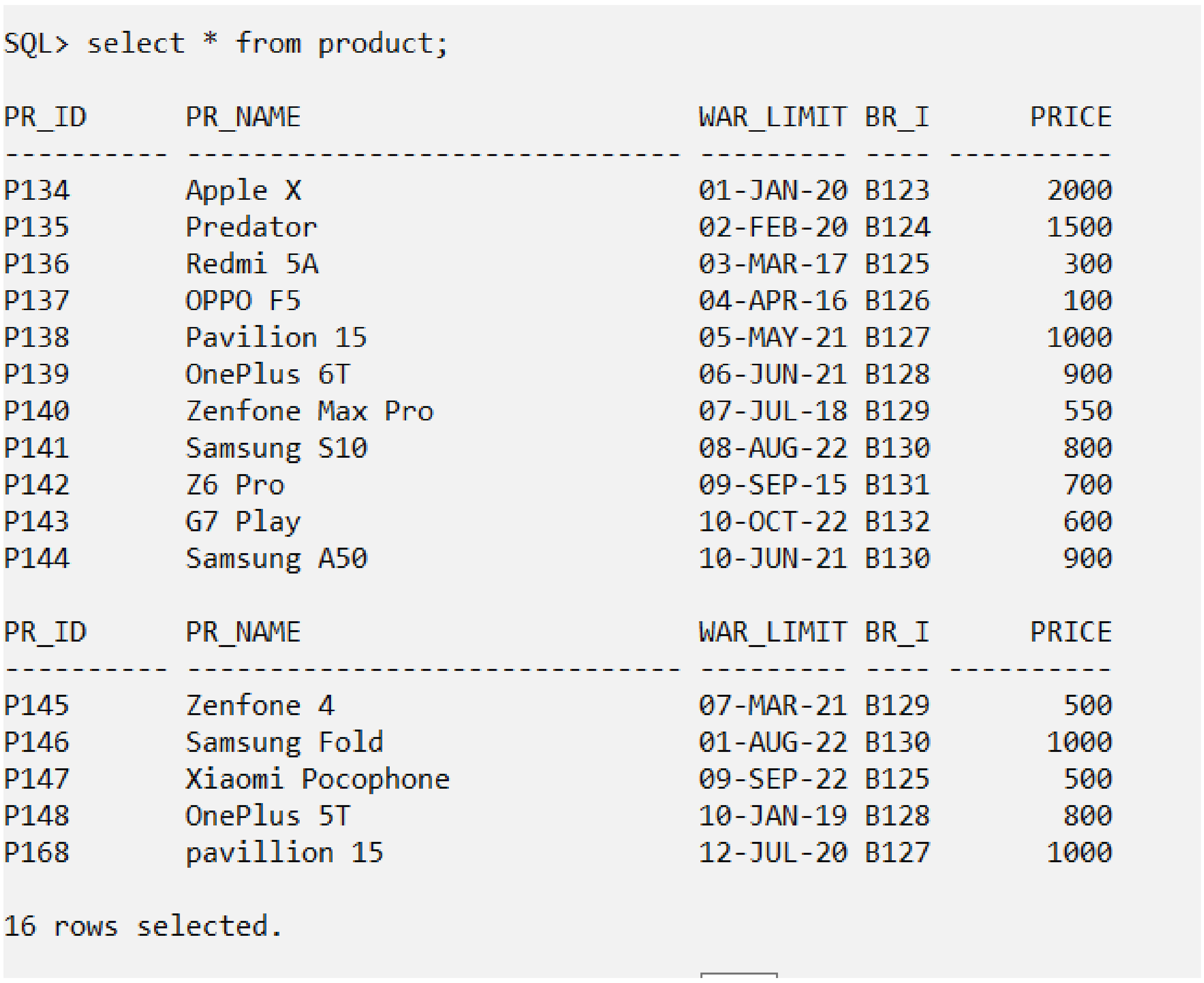
2. Add new product procedure creation

```
CREATE OR REPLACE PROCEDURE addnewprod
        (proID product.pr_id%TYPE,
        prodN product.pr_name%TYPE,
        wLIM date,
        brand branch.manufacturer%TYPE)
        IS
        v_price number(6,2);
        v_brid varchar2(6);
BEGIN


IF brand = 'Apple' THEN
        v_brid := 'B123';
        v_price := 2000;
ELSIF brand = 'acer' THEN
        v_brid := 'B124';
        v_price := 1500;
ELSIF brand = 'Xiaomi' THEN
        v_brid := 'B125';
        v_price := 300;
ELSIF brand = 'oppo' THEN
        v_brid := 'B126';
        v_price := 100;
ELSIF brand = 'hp' THEN
        v_brid := 'B127';
        v_price := 1000;
ELSIF brand = 'OnePlus' THEN
        v_brid := 'B128';
        v_price := 900;
ELSIF brand = 'asus' THEN
        v_brid := 'B129';
```

```
            v_price := 550;
ELSIF brand = 'Samsung' THEN
            v_brid := 'B130';
            v_price := 800;
ELSIF brand = 'lenovo' THEN
            v_brid := 'B131';
            v_price := 1000;
ELSE
            v_brid := 'B132';
            v_price := 700;
END IF;

INSERT INTO product (pr_id, pr_name, war_limit, br_id, price)
VALUES (proID, prodN, wLIM, v_brid, v_price);

DBMS_OUTPUT.put_line('The product data is successfully entered.');
END;
/
```

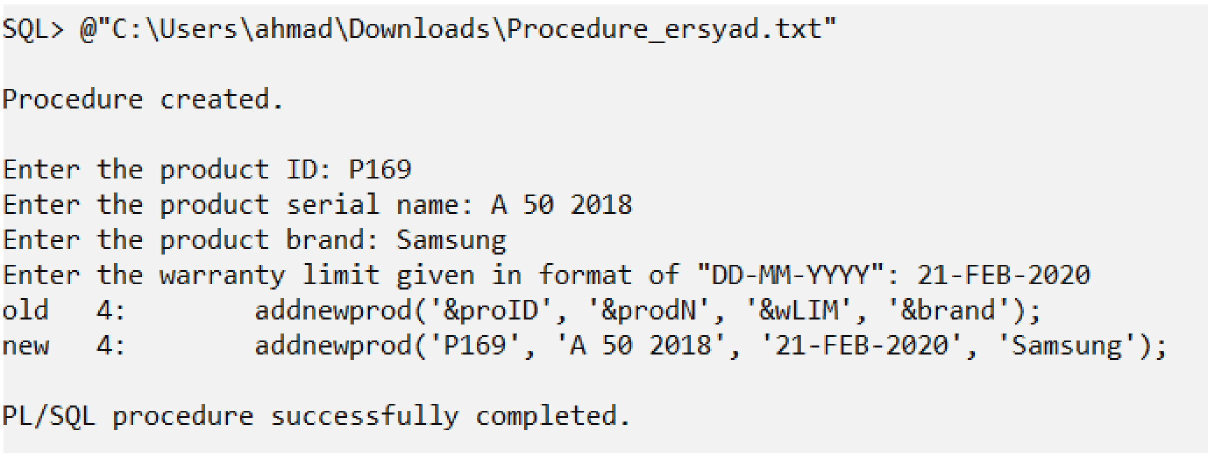
An anonymous block to call procedure *addnewprod*

```
ACCEPT proID PROMPT 'Enter the product ID: '
ACCEPT prodN PROMPT 'Enter the product serial name: '
ACCEPT brand PROMPT 'Enter the product brand: '
ACCEPT wLIM PROMPT 'Enter the warranty limit given in format of "DD-MM-
YYYY": '

DECLARE

BEGIN
            addnewprod('&proID', '&prodN', '&wLIM', '&brand');
END;
/
```



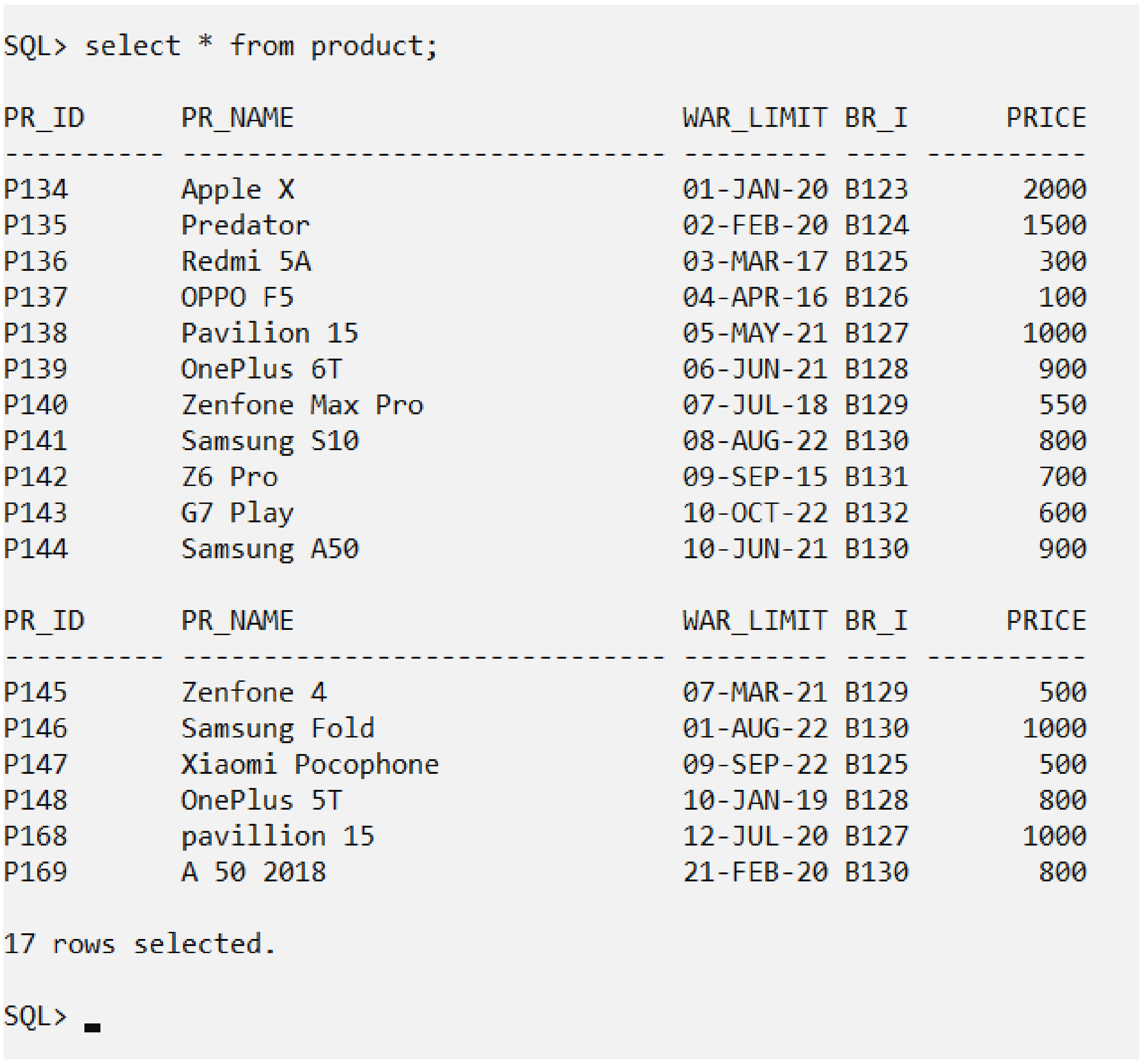
```
SQL> select * from product;

PR_ID      PR_NAME                        WAR_LIMIT BR_I      PRICE
---------- ------------------------------ --------- ---- ----------
P134       Apple X                        01-JAN-20 B123       2000
P135       Predator                       02-FEB-20 B124       1500
P136       Redmi 5A                       03-MAR-17 B125        300
P137       OPPO F5                        04-APR-16 B126        100
P138       Pavilion 15                    05-MAY-21 B127       1000
P139       OnePlus 6T                     06-JUN-21 B128        900
P140       Zenfone Max Pro                07-JUL-18 B129        550
P141       Samsung S10                    08-AUG-22 B130        800
P142       Z6 Pro                         09-SEP-15 B131        700
P143       G7 Play                        10-OCT-22 B132        600
P144       Samsung A50                    10-JUN-21 B130        900

PR_ID      PR_NAME                        WAR_LIMIT BR_I      PRICE
---------- ------------------------------ --------- ---- ----------
P145       Zenfone 4                      07-MAR-21 B129        500
P146       Samsung Fold                   01-AUG-22 B130       1000
P147       Xiaomi Pocophone               09-SEP-22 B125        500
P148       OnePlus 5T                     10-JAN-19 B128        800
P168       pavillion 15                   12-JUL-20 B127       1000

16 rows selected.
```

Data before run procedure



```
SQL> @"C:\Users\ahmad\Downloads\Procedure_ersyad.txt"

Procedure created.

Enter the product ID: P169
Enter the product serial name: A 50 2018
Enter the product brand: Samsung
Enter the warranty limit given in format of "DD-MM-YYYY": 21-FEB-2020
old   4:        addnewprod('&proID', '&prodN', '&wLIM', '&brand');
new   4:        addnewprod('P169', 'A 50 2018', '21-FEB-2020', 'Samsung');

PL/SQL procedure successfully completed.
```

Running the procedure and anonymous block



```
SQL> select * from product;

PR_ID      PR_NAME                        WAR_LIMIT BR_I      PRICE
---------- ------------------------------ --------- ---- ----------
P134       Apple X                        01-JAN-20 B123       2000
P135       Predator                       02-FEB-20 B124       1500
P136       Redmi 5A                       03-MAR-17 B125        300
P137       OPPO F5                        04-APR-16 B126        100
P138       Pavilion 15                    05-MAY-21 B127       1000
P139       OnePlus 6T                     06-JUN-21 B128        900
P140       Zenfone Max Pro                07-JUL-18 B129        550
P141       Samsung S10                    08-AUG-22 B130        800
P142       Z6 Pro                         09-SEP-15 B131        700
P143       G7 Play                        10-OCT-22 B132        600
P144       Samsung A50                    10-JUN-21 B130        900

PR_ID      PR_NAME                        WAR_LIMIT BR_I      PRICE
---------- ------------------------------ --------- ---- ----------
P145       Zenfone 4                      07-MAR-21 B129        500
P146       Samsung Fold                   01-AUG-22 B130       1000
P147       Xiaomi Pocophone               09-SEP-22 B125        500
P148       OnePlus 5T                     10-JAN-19 B128        800
P168       pavillion 15                   12-JUL-20 B127       1000
P169       A 50 2018                      21-FEB-20 B130        800

17 rows selected.

SQL>
```

After procedure run

3. Creating Procedure for adding new employee and generate in Branch Table

```
create or replace procedure Addnewemp
(e_emp_id employee.emp_id%TYPE,
e_ename employee.ename%TYPE,
e_hiredate employee.hiredate%TYPE,
e_position employee.position%TYPE,
e_sal employee.sal%TYPE,
e_br_id employee.br_id%TYPE )
AS

BEGIN
INSERT INTO employee
(emp_id, ename,hiredate, position, sal, br_id)
VALUES ( e_emp_id, e_ename, e_hiredate, e_position, e_sal, e_br_id);
END Addnewemp;
/

ACCEPT e_emp_id PROMPT 'Enter Employee ID: ' ;
ACCEPT e_ename PROMPT 'Enter Employee Name: ' ;
ACCEPT e_hiredate PROMPT 'Enter the date of hired: ' ;
ACCEPT e_position PROMPT 'Enter Position: ' ;
ACCEPT e_sal PROMPT 'Enter Employee salary: ' ;
ACCEPT e_br_id PROMPT 'Enter Employee Branch ID: ' ;

BEGIN

Addnewemp ('&e_emp_id', '&e_ename', '&e_hiredate', '&e_position' ,'&e_sal',
'&e_br_id');

DBMS_OUTPUT.put_line ('Data is successfully entered. Thanks for your time');

END;
/
```

8.0 FUNCTION

1. Checking whether a customer’s product is eligible for using the warranty to cover the payment by checking the warranty limits to date of enquiry. If it exceeds the warranty limit then customer would pay the standardized price of a warranty.

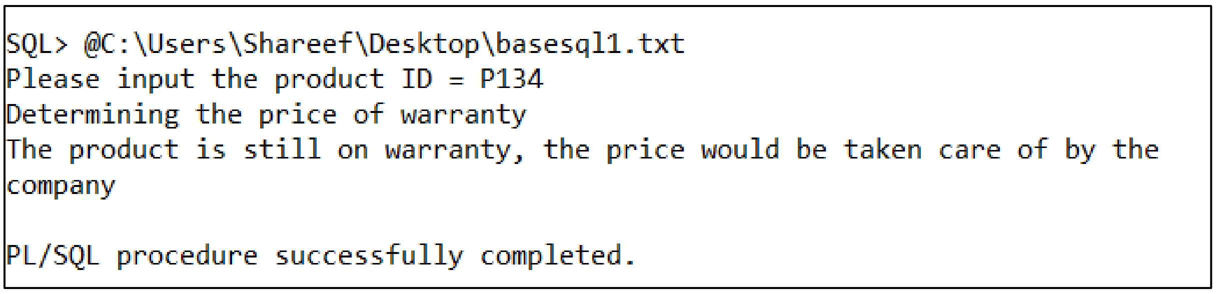
Function to check the warranty

```
CREATE OR REPLACE FUNCTION warranty_check
        (v_warranty  IN date,
        v_price     IN number ,
        v_startdate IN date)
        RETURN number
        IS
        f_price number(10,2);
BEGIN
        IF v_warranty > v_startdate THEN f_price := 0;
                DBMS_OUTPUT.put_line('The product is still on warranty, the price would be
                taken care of by the company');
        ELSE   f_price := v_price;
        DBMS_OUTPUT.put_line('The price of warranty would be ' || f_price);
        END IF;
        return(f_price);
END warranty_check;
/
```

Anonymous block

```
ACCEPT productID PROMPT 'Please input the product ID: '
DECLARE
        v_pr_id         product.pr_id%type := '&productID';
        v_warranty      product.war_limit%type;
        v_price         product.price%type;
        v_totalprice    billing.totalprice%type;
        v_startdate     billing.startdate%type;
BEGIN
        DBMS_OUTPUT.put_line('Determining the price of warranty');
        SELECT war_limit , price
        INTO v_warranty , v_price
        FROM product
        WHERE pr_id = v_pr_id;

        SELECT startdate
        INTO v_startdate
        FROM billing
        where pr_id = v_pr_id;

        v_totalprice := warranty_check( v_warranty , v_price , v_startdate);

        UPDATE billing
        set totalprice = v_totalprice
        where pr_id = v_pr_id;
end;
/
```

Executing the Function



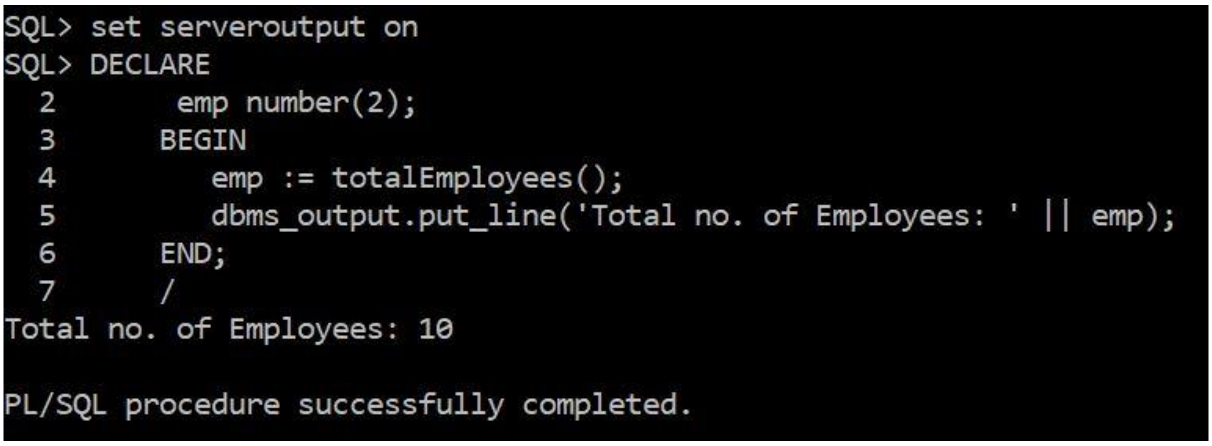
```
SQL> @C:\Users\Shareef\Desktop\basesql1.txt
Please input the product ID = P134
Determining the price of warranty
The product is still on warranty, the price would be taken care of by the
company

PL/SQL procedure successfully completed.
```

2. **Creating Function for Employee Table to count total numbers of employees**

```
SET SERVEROUTPUT ON
SET VERIFY OFF

SQL> CREATE OR REPLACE FUNCTION totalEmployees
RETURN number IS
    total number(2) := 0;
    BEGIN
    SELECT count(*) INTO total
    from employee;
    RETURN total;
    END;
```

Anonymous block

```
SET SERVEROUTPUT ON
SET VERIFY OFF

SQL> DECLARE

   emp number(2);
   BEGIN
   emp := totalEmployees();
   dbms_output.put_line('Total no. of Employees: ' || emp);
   END;
```



```
SQL> set serveroutput on
SQL> DECLARE
  2       emp number(2);
  3      BEGIN
  4         emp := totalEmployees();
  5         dbms_output.put_line('Total no. of Employees: ' || emp);
  6      END;
  7      /
Total no. of Employees: 10

PL/SQL procedure successfully completed.
```

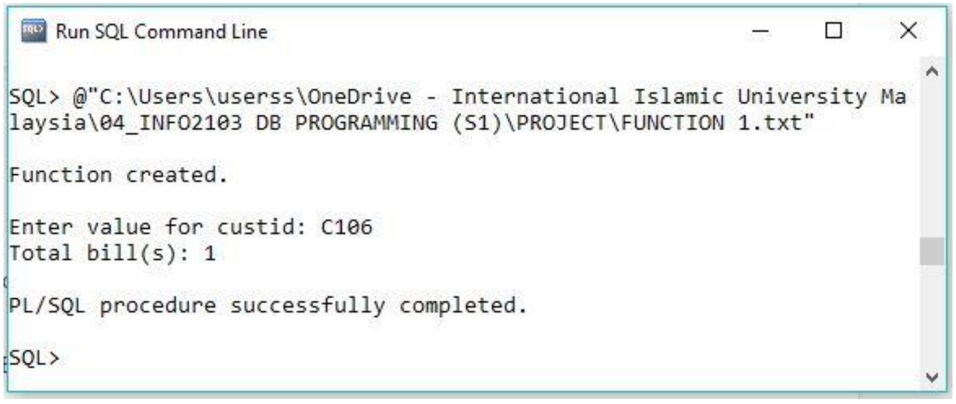
3. Calculate the total number of bills paid by a customer

```
CREATE OR REPLACE FUNCTION total_num_bills (custid IN varchar2)
        RETURN number
        IS
        totalnum number;
BEGIN
        SELECT COUNT(cust_id)
        INTO totalnum
        FROM billing
        WHERE cust_id = custid;
        RETURN (totalnum);
END;
/
```

Anonymous block to call function *total_num_bills*

```
DECLARE
        custid customer.cust_id%TYPE;
        finalamt NUMBER(5);
BEGIN
        finalamt := total_num_bills('&custid');
        dbms_output.put_line('Total bill(s): '||finalamt);
END;
/
```

Output



Run SQL Command Line

```
SQL> @"C:\Users\userss\OneDrive - International Islamic University Ma
laysia\04_INFO2103 DB PROGRAMMING (S1)\PROJECT\FUNCTION 1.txt"

Function created.

Enter value for custid: C106
Total bill(s): 1

PL/SQL procedure successfully completed.

SQL>
```

4. Function that calculate the total payment for the customer to pay depends on the branch. A discount of 5% will be given for customer that purchase any items at B128, 10% discount in branch B129 and 50% discount will be given to customer of branch B132.

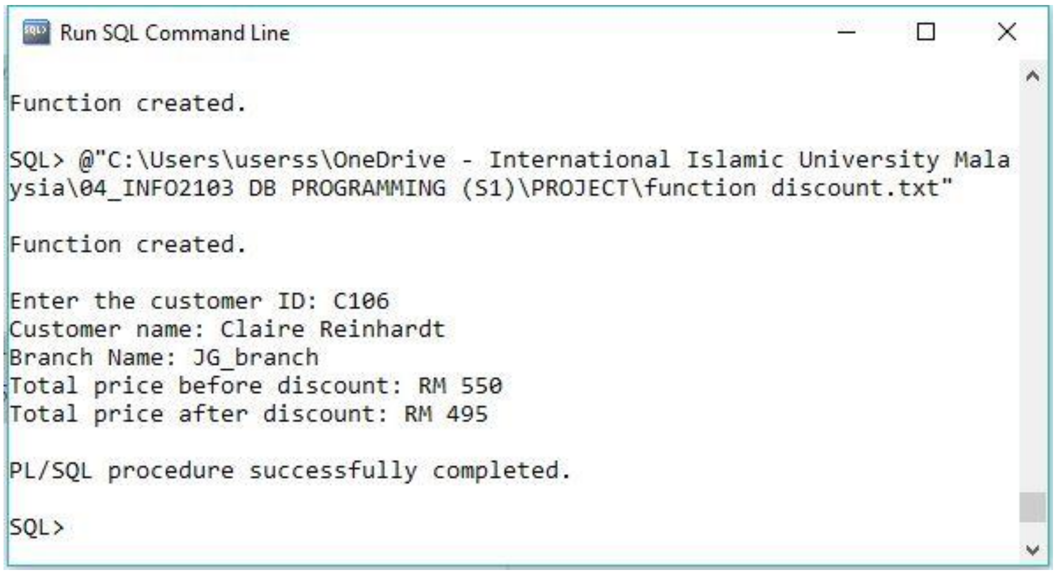
Function *discount*

```
CREATE OR REPLACE FUNCTION discount(v_brid IN varchar2)
        RETURN number
        IS
        discount number;
BEGIN
        IF v_brid = 'B128' THEN discount:= 0.05;
        ELSIF v_brid = 'B129' THEN discount:= 0.1;
        ELSIF v_brid = 'B132' THEN discount:= 0.5;
        ELSE discount:= 0.0;
        END IF;
        RETURN(discount);
END;
/
```

Anonymous block call function *discount*

```
ACCEPT custid PROMPT 'Enter the customer ID: '
DECLARE
        custid customer.cust_id%TYPE;
        custname customer.cust_name%TYPE;
        bname branch.br_name%TYPE;
        prid billing.pr_id%TYPE;
        brid product.br_id%TYPE;
        price billing.totalprice%TYPE;
        total NUMBER(6,2);
BEGIN
        SELECT cust_name
        INTO custname
        FROM customer
        WHERE cust_id = '&custid';

        SELECT cust_id, br_name, product.br_id, billing.pr_id, totalprice
        INTO custid, bname, brid, prid, price
        FROM billing, product, branch
        WHERE branch.br_id = product.br_id AND
            billing.pr_id = product.pr_id AND
            cust_id = '&custid';
        total := price - (price * discount(brid));
        dbms_output.put_line('Customer name: '||custname);
        dbms_output.put_line('Branch Name: '||bname);
        dbms_output.put_line('Total price before discount: RM '||price);
        dbms_output.put_line('Total price after discount: RM '||total);
END;
/
```

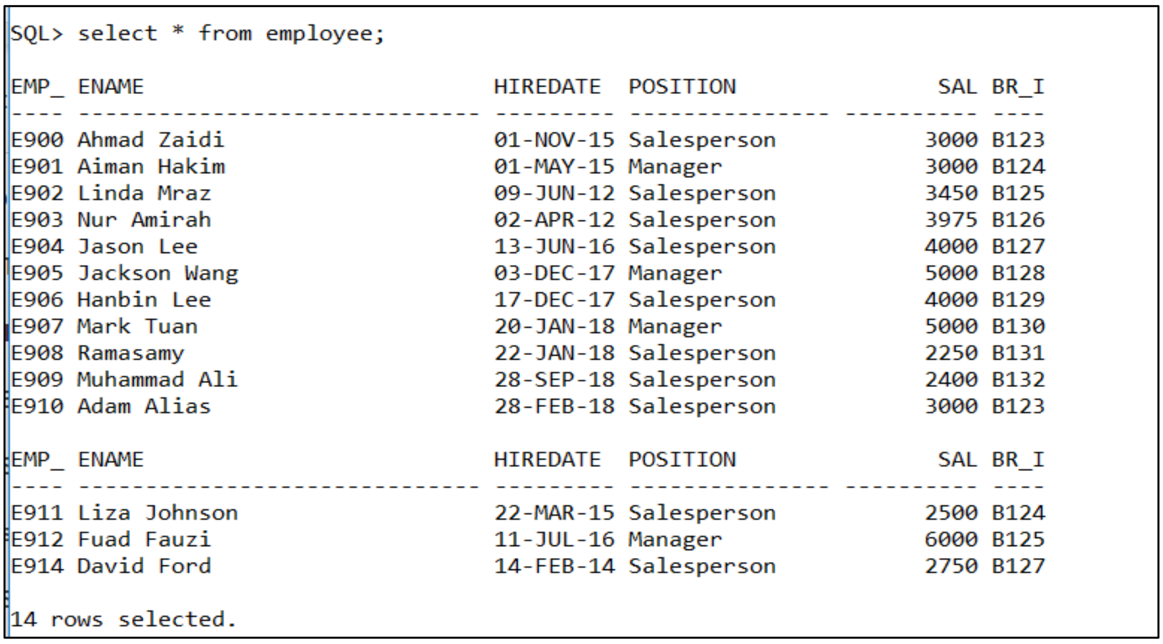
Output



```
Function created.

SQL> @"C:\Users\userss\OneDrive - International Islamic University Mala
ysia\04_INFO2103 DB PROGRAMMING (S1)\PROJECT\function discount.txt"

Function created.

Enter the customer ID: C106
Customer name: Claire Reinhardt
Branch Name: JG_branch
Total price before discount: RM 550
Total price after discount: RM 495

PL/SQL procedure successfully completed.

SQL>
```

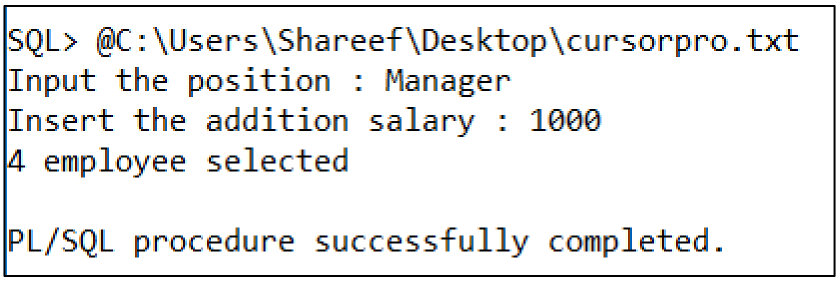
8.0 CURSOR

```
ACCEPT pos PROMPT 'Input the position : ';
ACCEPT add PROMPT 'Insert the addition salary : ';
DECLARE
    total_emp number(2);
    v_sal     employee.sal%TYPE;
BEGIN
    UPDATE employee
    SET sal = sal + 500
    WHERE position = '&pos';

    IF sql%notfound THEN
            dbms_output.put_line('no employee selected');
    ELSIF sql%found THEN
            total_emp := sql%rowcount;
            dbms_output.put_line( total_emp || ' employee selected ');
    END IF;
END;
/
```



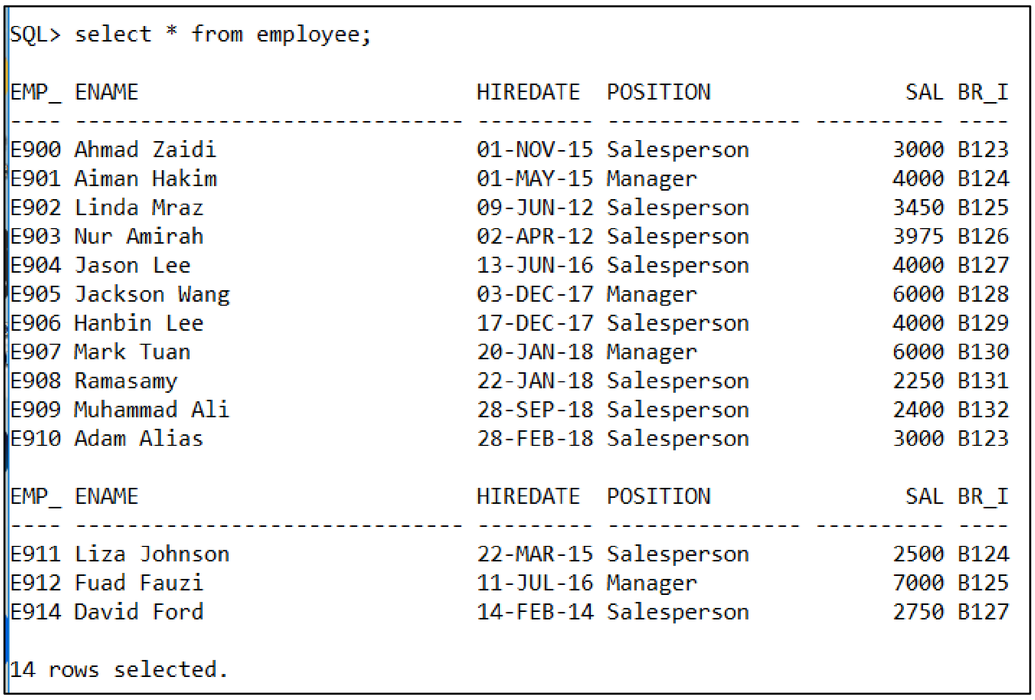
```
SQL> select * from employee;

EMP_ ENAME                          HIREDATE  POSITION               SAL BR_I
---- ------------------------------ --------- --------------- ---------- ----
E900 Ahmad Zaidi                    01-NOV-15 Salesperson           3000 B123
E901 Aiman Hakim                    01-MAY-15 Manager               3000 B124
E902 Linda Mraz                     09-JUN-12 Salesperson           3450 B125
E903 Nur Amirah                     02-APR-12 Salesperson           3975 B126
E904 Jason Lee                      13-JUN-16 Salesperson           4000 B127
E905 Jackson Wang                   03-DEC-17 Manager               5000 B128
E906 Hanbin Lee                     17-DEC-17 Salesperson           4000 B129
E907 Mark Tuan                      20-JAN-18 Manager               5000 B130
E908 Ramasamy                       22-JAN-18 Salesperson           2250 B131
E909 Muhammad Ali                   28-SEP-18 Salesperson           2400 B132
E910 Adam Alias                     28-FEB-18 Salesperson           3000 B123

EMP_ ENAME                          HIREDATE  POSITION               SAL BR_I
---- ------------------------------ --------- --------------- ---------- ----
E911 Liza Johnson                   22-MAR-15 Salesperson           2500 B124
E912 Fuad Fauzi                     11-JUL-16 Manager               6000 B125
E914 David Ford                     14-FEB-14 Salesperson           2750 B127

14 rows selected.
```

Data before the anonymous procedure



```
SQL> @C:\Users\Shareef\Desktop\cursorpro.txt
Input the position : Manager
Insert the addition salary : 1000
4 employee selected

PL/SQL procedure successfully completed.
```

Executing the cursor



```
SQL> select * from employee;

EMP_ ENAME                          HIREDATE  POSITION               SAL BR_I
---- ------------------------------ --------- --------------- ---------- ----
E900 Ahmad Zaidi                    01-NOV-15 Salesperson           3000 B123
E901 Aiman Hakim                    01-MAY-15 Manager               4000 B124
E902 Linda Mraz                     09-JUN-12 Salesperson           3450 B125
E903 Nur Amirah                     02-APR-12 Salesperson           3975 B126
E904 Jason Lee                      13-JUN-16 Salesperson           4000 B127
E905 Jackson Wang                   03-DEC-17 Manager               6000 B128
E906 Hanbin Lee                     17-DEC-17 Salesperson           4000 B129
E907 Mark Tuan                      20-JAN-18 Manager               6000 B130
E908 Ramasamy                       22-JAN-18 Salesperson           2250 B131
E909 Muhammad Ali                   28-SEP-18 Salesperson           2400 B132
E910 Adam Alias                     28-FEB-18 Salesperson           3000 B123

EMP_ ENAME                          HIREDATE  POSITION               SAL BR_I
---- ------------------------------ --------- --------------- ---------- ----
E911 Liza Johnson                   22-MAR-15 Salesperson           2500 B124
E912 Fuad Fauzi                     11-JUL-16 Manager               7000 B125
E914 David Ford                     14-FEB-14 Salesperson           2750 B127

14 rows selected.
```

Data after the anonymous cursor

9.0 Record Listing

EMP_ID	ENAME	HIREDATE	POSITION	SAL	BR_ID
E900	Ahmad Zaidi	01-NOV-15	Salesperson	2000	B123
E901	Aiman Hakim	01-MAY-15	Manager	3000	B124
E902	Linda Mraz	09-JUN-12	Salesperson	2450	B125
E903	Nur Amirah	02-APR-12	Salesperson	2975	B126
E904	Jason Lee	13-JUN-16	Salesperson	3000	B127
E905	Jackson Wang	03-DEC-17	Manager	5000	B128
E906	Hanbin Lee	17-DEC-17	Salesperson	3000	B129
E907	Mark Tuan	20-JAN-18	Manager	5000	B130
E908	Ramasamy	22-JAN-18	Salesperson	1250	B131
E909	Muhammad Ali	28-SEP-18	Salesperson	1400	B132
E910	Adam Alias	28-FEB-18	Salesperson	2000	B123
E911	Liza Johnson	22-MAR-15	Salesperson	1500	B124
E912	Fuad Fauzi	11-JUL-16	Manager	6000	B125
E914	David Ford	14-FEB-14	Salesperson	1750	B127
E913	Julie Lee	28-MAY-15	Salesperson	2100	B126

CUST_ID	CUST_NAME	PHONE_NO	ADDRESS
C100	Nur Nabilah	178270305	No 25, Jalan Tasek 9
C101	Nabihah Kassim	123456789	No 9, Jalan Rambutan
C102	Mohamed Fawzy	145672804	22 Deer Rd
C103	Mohd Syahmi	111118239	Lot 1228 Jalan Nam Heng
C104	Lee Jing Young	182736228	99 Main St
C105	Ada Wong	135579924	11 Sesame St
C106	Claire Reinhardt	152293456	12 Sesame St
C107	Bridgette White	124456332	30 Main St
C108	Pharah Tamriel	145562785	122 Moon St
C109	Tom Hawks	154492341	61 Sun St
C110	Leon Kennedy	132212334	15 Sun St
C111	Joseph Oda	154432122	40 Sesame St
C112	Nicolla Bellic	154487001	70 Sesame St
C113	Hamidi Hamzah	132267901	01 Matahari St
C114	Anthony Addams	157751342	80 Matahari St

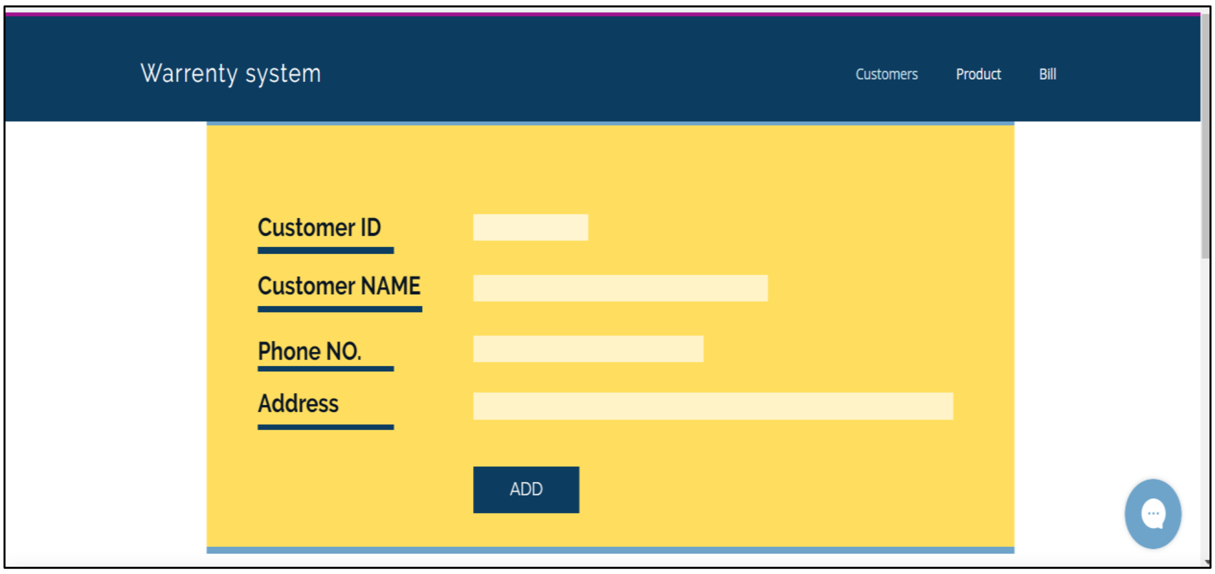
BR_ID	BR_NAME	CITY	POST_CODE	MANUFACTURER
B123	KL_branch	Kuala Lumpur	1423	Apple
B124	SN_branch	Selangor	5317	acer
B125	PG_branch	Penang	1357	Xiaomi
B126	SW_branch	Serawak	6879	oppo
B127	SB_branch	Sabah	4567	hp
B128	JK_branch	Jakarta	6575	OnePlus
B129	JG_branch	Jogjakarta	9827	asus

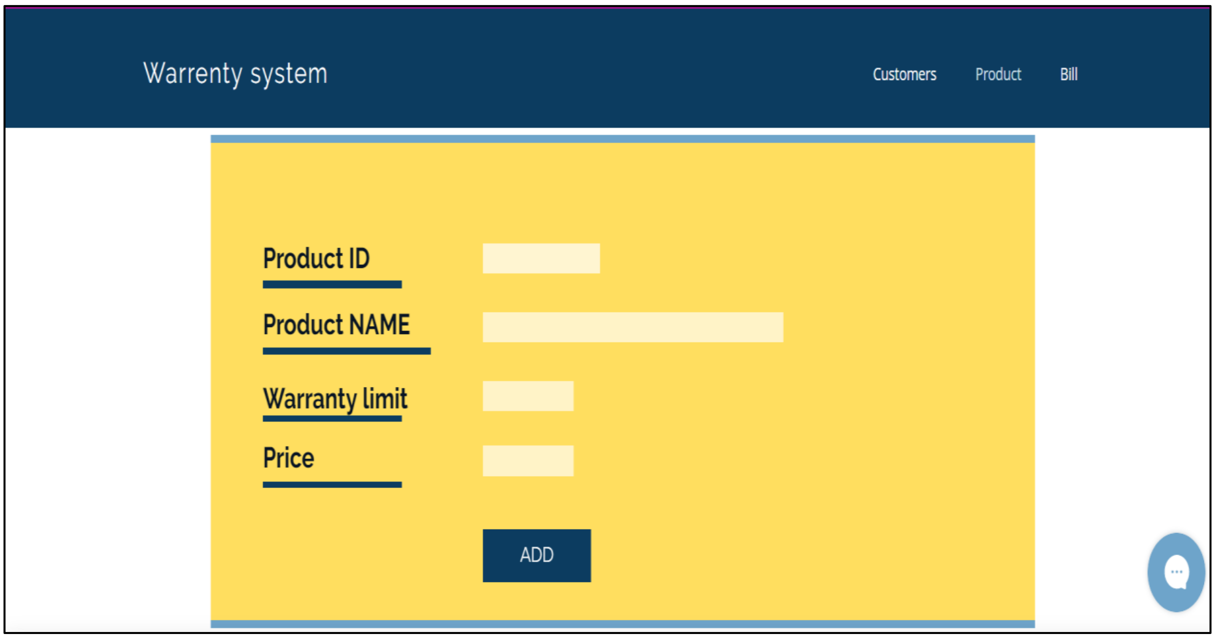
B130	DH_branch	Dhaka	5578	Samsung
B131	CT_branch	Chittagong	8966	lenovo
B132	SG_branch	Singapore	2767	Motorola

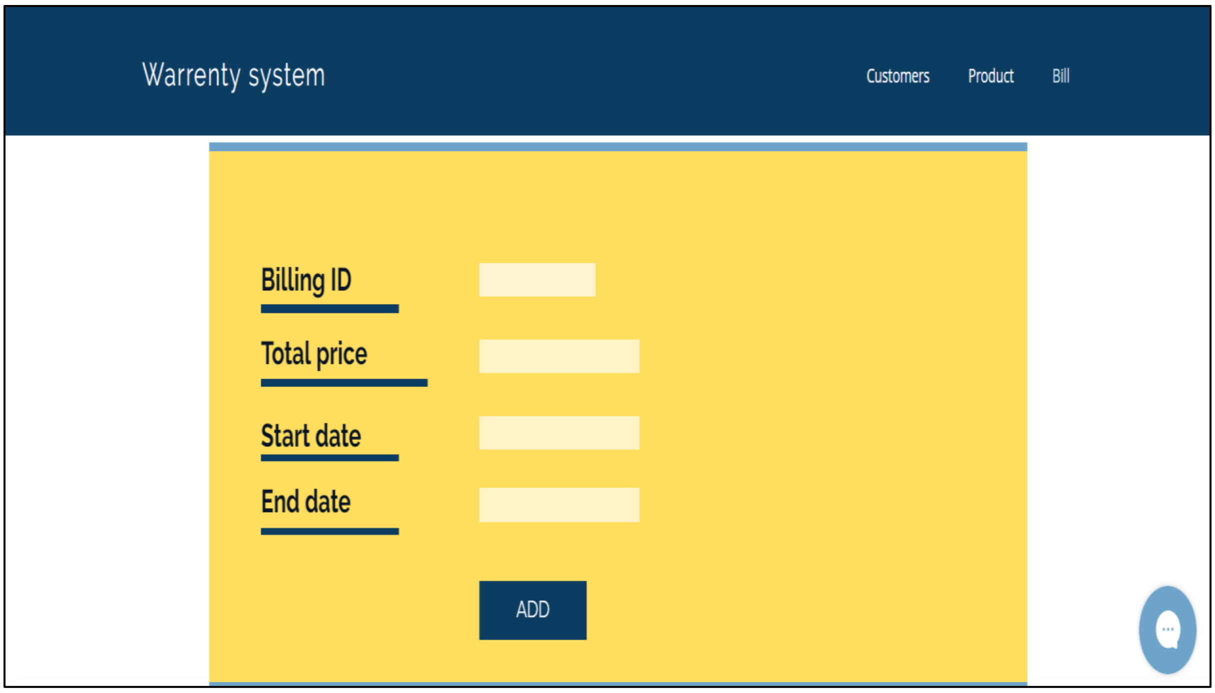
PR_ID	PR_NAME	WAR_LIMIT	BR_ID	PRICE
P134	Apple X	01-JAN-20	B123	2000
P135	Predator	02-FEB-20	B124	1500
P136	Redmi 5A	03-MAR-17	B125	300
P137	OPPO F5	04-APR-16	B126	100
P138	Pavilion 15	05-MAY-21	B127	1000
P139	OnePlus 6T	06-JUN-21	B128	900
P140	Zenfone Max Pro	07-JUL-18	B129	550
P141	Samsung S10	08-AUG-22	B130	800
P142	Z6 Pro	09-SEP-15	B131	700
P143	G7 Play	10-OCT-22	B132	600
P144	Samsung A50	10-JUN-21	B130	900
P145	Zenfone 4	07-MAR-21	B129	500
P146	Samsung Fold	01-AUG-22	B130	1000
P147	Xiaomi Pocophone	09-SEP-22	B125	500
P148	OnePlus 5T	10-JAN-19	B128	800

BILL_ID	CUST_ID	PR_ID	TOTALPRICE	STARTDATE	ENDDATE
BL100	C100	P134	0	01-JAN-18	02-FEB-18
BL101	C101	P135	0	05-MAR-18	07-APR-18
BL102	C102	P136	300	05-JUL-18	17-AUG-18
BL103	C103	P137	100	05-MAR-18	10-APR-18
BL104	C104	P138	0	15-JAN-19	19-FEB-19
BL105	C105	P139	0	03-FEB-19	18-APR-19
BL106	C106	P140	550	01-AUG-18	20-AUG-18
BL107	C107	P141	0	09-SEP-18	30-SEP-18
BL108	C108	P142	700	11-FEB-18	17-APR-18
BL109	C109	P143	0	11-FEB-18	15-APR-18
BL110	C110	P144	0	15-MAR-19	17-APR-19
BL111	C111	P145	0	07-JAN-19	27-FEB-18
BL112	C112	P146	0	15-FEB-19	19-MAR-19
BL113	C113	P147	0	05-MAR-19	29-APR-19
BL114	C114	P148	800	27-FEB-19	22-MAR-19

9.0 INTERFACES





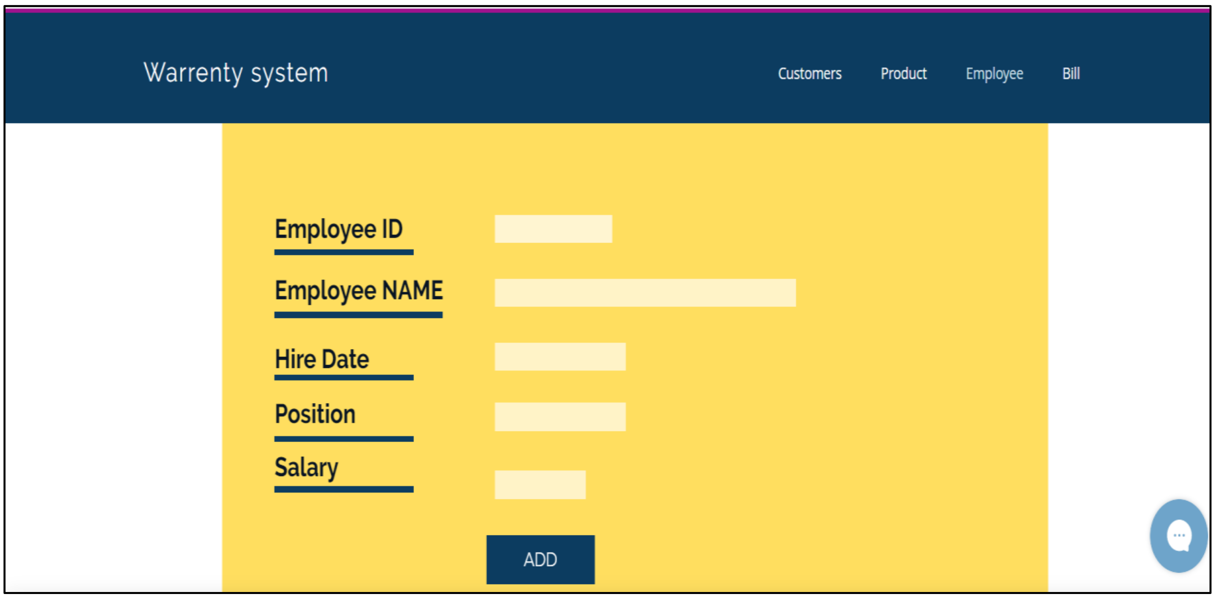


Billing ID

Total price

Start date

End date



Employee ID

Employee NAME

Hire Date

Position

Salary