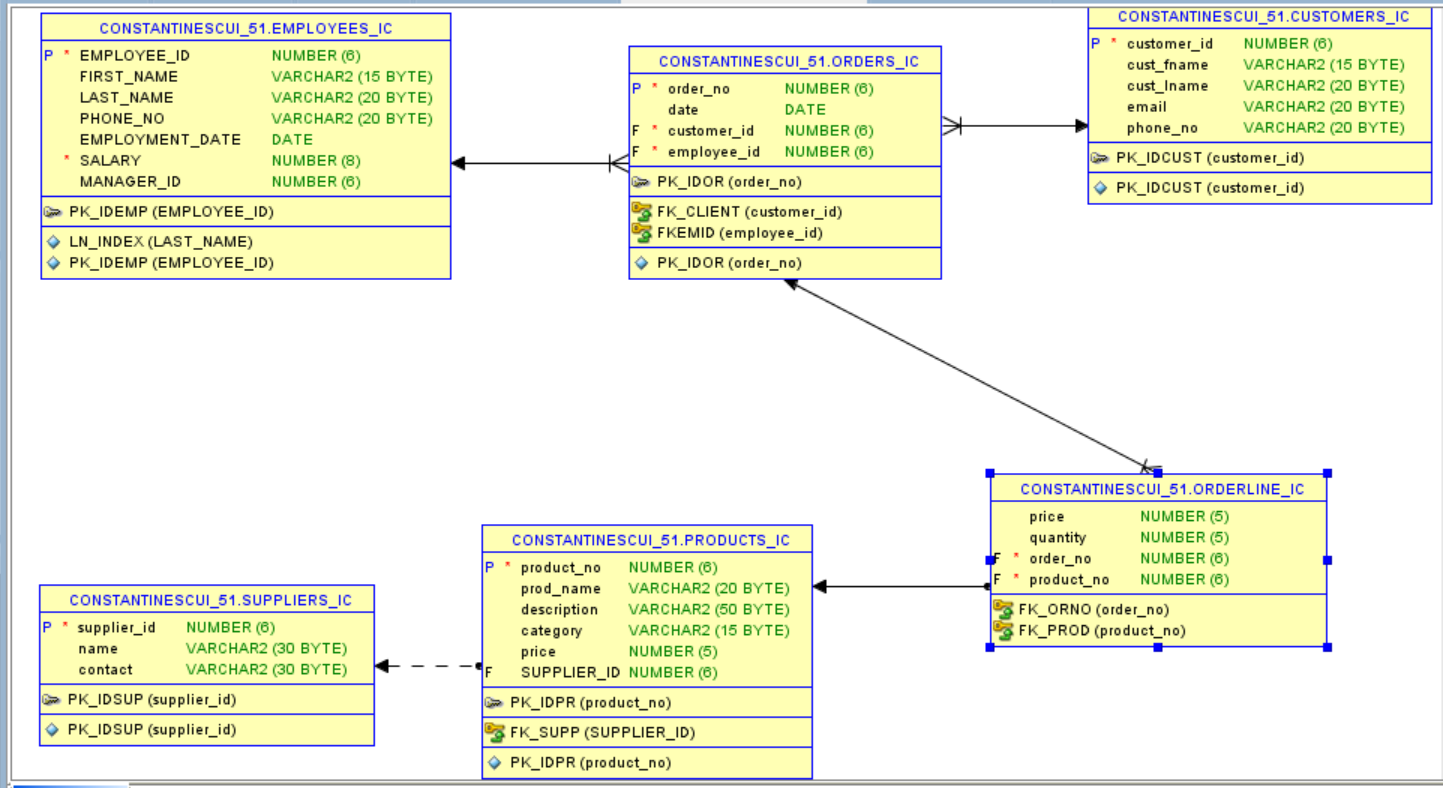
|  |  |
| --- | --- |
| DBMS PROJECT  MANAGEMENT OF A BAKERY | Ioana Constantinescu, Group 1051  Faculty of Cybernetics, Statistics and Informatics Economics  Professor Coordinator: Vlad Diaconiţă |

Description of the project

The objective of the project is to keep an organized evidence of a bakery store, allowing the user to quickly visualize key elements related to organizational and economic activity of the enterprise. The information in the database are fictional.

The database is made of six tables and has the following conceptual schema:The six entities are: EMPLOYEES, ORDERS, ORDER LINE, CUSTOMERS, PRODUCTS, and SUPPLIERS.

* **EMPLOYEES\_IC** table contains the following fields: **employee\_id:** each employee has an unique number as a primary key; **first\_name**: the first name of the employee; **last\_name**: the last name of the employee; **address**: the address of the employee; **email**: the email of the employee; **phone\_no**: the phone number of the employee; **birth\_date**: the birth date of the employee; **employment\_date**: date (day, month, year) on which the person was employed in the enterprise; **salary**: the actual salary of the employee.
* **ORDERS\_IC** table contains the following fields: **orders\_no**: each purchase (comprising one or more products) has a unique identification number that is a primary key; **date**: the date the acquisition will be needed to be done; **client\_id:** foreign key (EMPLOYEES).
* **ORDER\_LINE\_IC** table contains the following fields: price: the total price a customer has to pay; quantity: the total quantity of the order; order\_no: foreign key (ORDERS), product\_id: foreign key (PRODUCTS).
* **CUSTOMERS\_IC** table contains the following fields: **customer\_id**: an unique number is the primary key; **cust\_fname**: customer surname; **cust\_lname**: last name of the customer; **phone**: the customer’s telephone number; **email**: e-mail of the customer.
* **PRODUCTS\_IC** table contains the following fields: **product\_id**: an unique number is the primary key; **name**: the name of the product; **description**: a short description for each product (ex: ingredients); **category**: the category of each product (ex: cupcakes, cakes, etc.); **price**: price of the product expressed in EURO; **suppliers\_id**: foreign key (SUPPLIERS).
* **SUPPLIERS\_IC** table contains the following fields: **supplier\_id**: an unique number is the primary key; **name**: the name of the supplier; **contact**: details of the supplier’s contact.

The tables have the following relations:

An **EMPLOYEE** can take one or many **ORDERS**, an **ORDER** can be taken by only one **EMPLOYEE**.

An **ORDER** can be made by only one **CUSTOMER**, a **CUSTOMER** can make one or many **ORDERS**.

An **ORDER** can have one or many **PRODUCTS**, a **PRODUCT** can be on one or many **ORDERS**. As this is a many-to-many relation, the **ORDER LINE** was created. Now the relations looks like this: an**ORDER LINE** can contain only one **ORDER**, an **ORDER** can be on one or many**ORDER LINES**. An **ORDER LINE** can contain only one **PRODUCT** at a time, a **PRODUCT** can be on one or many **ORDER LINES**.

A **PRODUCT** can have only one **SUPPLIER**, a **SUPPLIER** can make one or many **PRODUCTS**.

Interacting with Oracle server using SQL statements (DDL and DML)

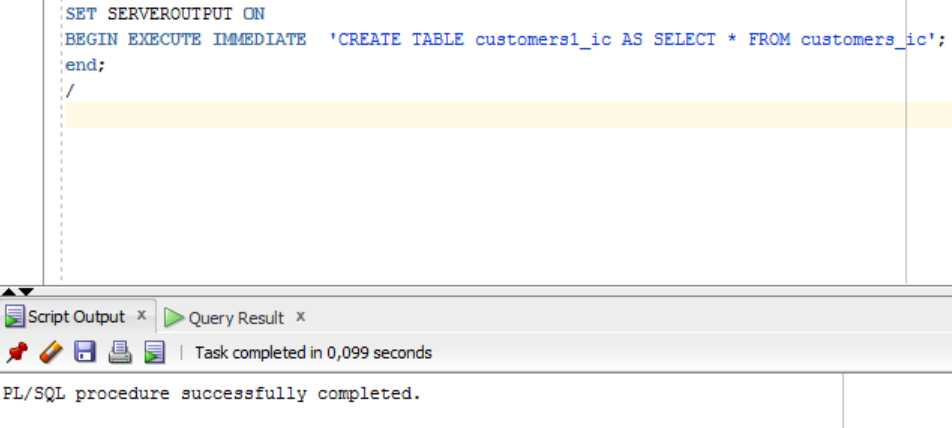
1. Create an identic table with the table customers\_ic.

SET SERVEROUTPUT ON

BEGIN EXECUTE IMMEDIATE 'CREATE TABLE customers1\_ic AS SELECT \* FROM customers\_ic';

end;

/



1. Print the email and the phone number from the table previously created for the consumer with the id 2.

SET SERVEROUTPUT ON

declare

c\_idc CUSTOMERS1\_IC.CUSTOMER\_ID%TYPE;

c\_email CUSTOMERS1\_IC.EMAIL%type;

c\_pn CUSTOMERS1\_IC.PHONE\_NO%type;

begin

select customer\_id, email , phone\_no into c\_idc, c\_email , c\_pn from CUSTOMERS1\_IC where customer\_id=2;

DBMS\_OUTPUT.PUT\_LINE ('Email:'||c\_email||' Phone Number:'||c\_pn);

end;

/



1. Insert in the table customers1\_ic a new column called age.

SET SERVEROUTPUT ON

declare

AGE\_IC NUMBER(8,2);

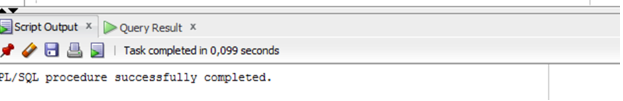
begin

AGE\_IC:='ALTER TABLE customers1\_ic ADD(AGE NUMBER(8,2))';

EXECUTE IMMEDIATE AGE\_IC ;

end;

/



1. Insert into the customers1\_ic table, on the age column where customers\_id=2.

set SERVEROUTPUT ON

accept key\_age prompt 'Insert the age:';

declare

age\_ic number(10):=&key\_age;

begin

update customers1\_ic

set AGE=age\_ic

where customer\_id=7;

end;

/

select \* from CUSTOMERS1\_IC;

1. Drop the table previously created.

SET SERVEROUTPUT ON

begin

execute immediate 'DROP table customers1\_ic';

end;

/

Decision and loop control structures (if, case, loop, for)

1. Display the first and last name of the customer whose salary is 1100 and check if they have equal lengths. If they do, please display the name, if not, to display the first name.

set serveroutput on

declare

name\_c EMPLOYEES\_IC.FIRST\_NAME%type;

surname\_c EMPLOYEES\_IC.LAST\_NAME%type;

begin

select first\_name, last\_name into name\_c, surname\_c from employees\_ic where salary= 1100;

if length(name\_c)! = length(surname\_c) then

dbms\_output.put\_line ('Surname:'|| surname\_c);

else

dbms\_output.put\_line ('Name:'|| name\_c);

end if;

end;

/

1. Blablabla
2. Blablabla

Managing exceptions

1. Create a table called VIP\_IC for the people that have a fidelity card at the bakery.

set serveroutput on

declare

table\_created EXCEPTION;

pragma EXCEPTION\_INIT(table\_created,-955); --non predefined exception

begin

execute immediate 'CREATE TABLE VIP\_IC (id\_p number(5) primary key, first\_name varchar2(20), last\_name varchar2(20))';

EXCEPTION

when table\_created then

dbms\_output.put\_line('The table already exists!');

execute immediate 'DROP TABLE PERSONS';

end;

/

1. Print the name of the product, the description and the price for the product id that is about to be read from the keyboard. If the product does not exist, treat the exception.

set serveroutput on

accept nr prompt 'Insert the product id: ';

declare

c\_nr number(4):=&nr;

c\_prod\_name products\_ic.prod\_name%type;

c\_description PRODUCTS\_IC.DESCRIPTION%type;

c\_price PRODUCTS\_IC.PRICE%type;

begin

select prod\_name, description, price into c\_prod\_name, c\_description, c\_price

from products\_ic where product\_no=c\_nr;

dbms\_output.put\_line ('Name Product:'||c\_prod\_name||' Description:'||c\_description||' Price:'||c\_price);

exception

when no\_data\_found then

dbms\_output.put\_line ('The product does not exist!');

end;

/

Insert product id: 1.

Name Product: Raspberry Donuts Description: Red Poison Price: 5