UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Database Management Systems, A.Y. 2019/2020
Master Degree in Computer Engineering
Master Degree in ICT for Internet and Multimedia

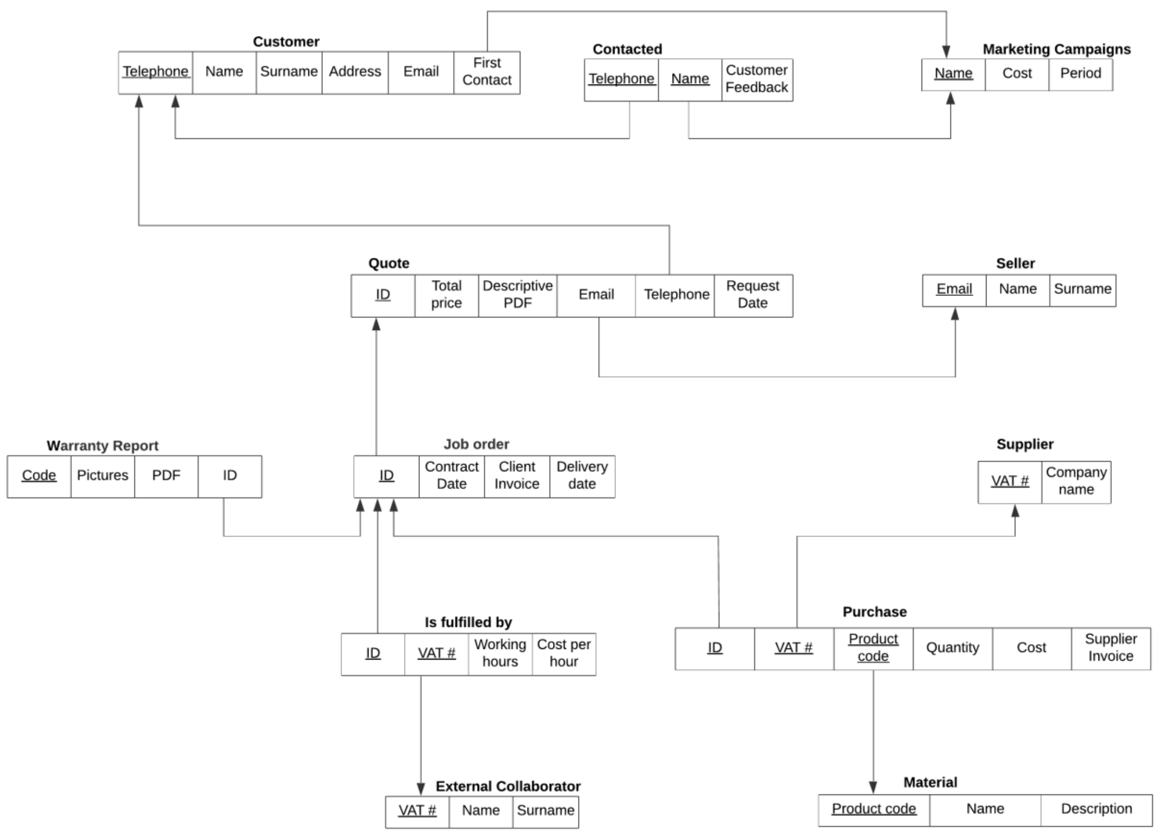
Homework 3 – Physical Design
Deadline: April 30, 2020

Group	Project	
BCCLMT	Vicentina Serramenti DBMS	
Last Name	First Name	Student Number
Bacchin	Alberto	1238166
Carta	Davide	1210702
Colotti	Giovanni	1237334
Lorenzi	Daniele	1237599
Michelon	Enrico	1210760
Trovò	Giacomo	1233438

Variations to the Relational Schema

We fixed the cardinality between Purchased and Material to [0,N], to admit insertion of materials without purchasing, as suggest from you at the end of homework 2. To cope with the external constraint on “First Contact” we thought that the simplest way is to put a foreign key to Marketing Campaign and eventually use a dummy instance called “other” that collects all the feedback from friends, word of mouth, etc..

We use the telephone number as primary key in the Customer relation since the stakeholder said the customers may not have emails due to a generally older age and phone is often used as the main contact mean. We also fixed some typos on data dictionary.



Physical Schema

In the following the SQL instructions to build the database in Figure 1 are reported.

```
CREATE DATABASE serramenti;
-- domain that perform a simple check on syntax of emails
CREATE DOMAIN EMAIL AS VARCHAR(50) CHECK(
(VALUE)::text~*'^[A-Za-z0-9._%-]+@[A-Za-z0-9.-]+[.][A-Za-z]+$'::text);

-- Table creation
CREATE TABLE Seller(
  email EMAIL,
  name VARCHAR(50) NOT NULL,
  surname VARCHAR(50) NOT NULL,
  PRIMARY KEY(email)
);

CREATE TABLE ExternalCollaborator(
  vat VARCHAR(15),
  name VARCHAR(50) NOT NULL,
  surname VARCHAR(50) NOT NULL,
  PRIMARY KEY(vat)
);
```

```
CREATE TABLE MarketingCampaign(
  name VARCHAR(50),
  cost NUMERIC(10,2) CHECK(cost>=0 OR NULL),
  period DATERANGE,
  PRIMARY KEY(name)
);

CREATE TABLE Customer(
  telephone VARCHAR(20),
  name VARCHAR(50) NOT NULL,
  surname VARCHAR(50) NOT NULL,
  address VARCHAR(150) NOT NULL,
  email EMAIL,
  firstContact VARCHAR(50),
  PRIMARY KEY(telephone),
  FOREIGN KEY (firstContact) REFERENCES MarketingCampaign(name)
);

CREATE TABLE Quote(
  id SERIAL,
  totalPrice NUMERIC(10,2) NOT NULL CHECK(totalPrice>=0),
  descriptivePDF TEXT NOT NULL,
  email EMAIL NOT NULL,
  telephone VARCHAR(20) NOT NULL,
  requestDate DATE NOT NULL,
  PRIMARY KEY(id),
  FOREIGN KEY (telephone) REFERENCES Customer(telephone),
  FOREIGN KEY (email) REFERENCES Seller(email)
);

CREATE TABLE JobOrder(
  id INTEGER,
  contractDate DATE NOT NULL,
  clientInvoice TEXT,
  deliveryDate DATE,
  PRIMARY KEY(id),
  FOREIGN KEY (id) REFERENCES Quote(id),
  CHECK (deliveryDate>contractDate)
);

CREATE TABLE WarrantyReport(
  code SERIAL,
  picture TEXT NOT NULL,
  pdf TEXT NOT NULL,
  jobOrderId INTEGER NOT NULL,
  PRIMARY KEY(code),
  FOREIGN KEY (jobOrderId) REFERENCES JobOrder(id)
);

CREATE TABLE Supplier(
  vat VARCHAR(15),
  companyName VARCHAR(50) NOT NULL,
  PRIMARY KEY(vat)
);
```

```sql
CREATE TABLE Material(
  productCode VARCHAR(50),
  name VARCHAR(50) NOT NULL,
  description TEXT,
  PRIMARY KEY (productCode)
);

CREATE TABLE Contacted(
  telephone VARCHAR(20),
  name VARCHAR(50),
  customerFeedback BOOLEAN DEFAULT FALSE NOT NULL,
  PRIMARY KEY (telephone, name),
  FOREIGN KEY (telephone) REFERENCES Customer(telephone),
  FOREIGN KEY (name) REFERENCES MarketingCampaign(name)
);

CREATE TABLE IsFulfilledBy(
  vat VARCHAR(15),
  id INTEGER,
  workingHours FLOAT,
  costPerHours FLOAT NOT NULL CHECK(costPerHours>=0),
  PRIMARY KEY (vat, id),
  FOREIGN KEY (vat) REFERENCES ExternalCollaborator(vat),
  FOREIGN KEY (id) REFERENCES JobOrder(id)
);

CREATE TABLE Purchased(
  id INTEGER,
  vat VARCHAR(15),
  productCode VARCHAR(50),
  quantity INTEGER NOT NULL,
  cost NUMERIC(10,2) NOT NULL CHECK(cost>=0),
  supplierInvoice TEXT,
  PRIMARY KEY(id, vat, productCode),
  FOREIGN KEY (id) REFERENCES JobOrder(id),
  FOREIGN KEY (vat) REFERENCES Supplier(vat),
  FOREIGN KEY (productCode) REFERENCES Material(productCode)
);
```

Populate the Database: Example

In the following, there are some examples of SQL instructions to insert a complete JobOrder.

```sql
-- The process of inserting a complete job order within the db is as
follows:
-- 1: The customer comes in the shop and its data are stored
-- 2: After a discussion, a quote with preliminary information is
produced
-- 3: The job order is inserted in the database after the customer signs
the contract with the company.
-- 4: Then all information about external collaborators that will
performs the job, together with the information about all the material
purchased, are inserted into the database.
-- 5: Finally, when work is completed, a warranty report is uploaded.
```

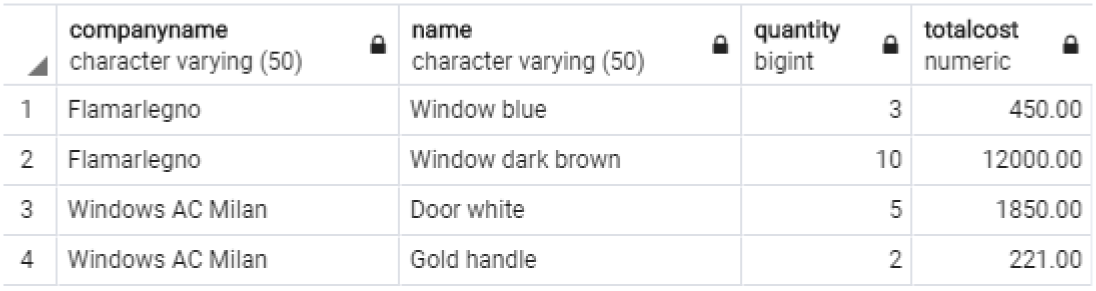
```
--note: PDF and images are stored as link to the company server

-- Insertion of a customer
INSERT INTO Customer VALUES ('+391112200987','Ernesto','Luigini','Via Vai
8','ernestluis@test.com','The bests');

-- Insertion of the related quote
INSERT INTO Quote VALUES (default,1200.00,
'https://www.vicentinaserramenti.it/quotes/123.pdf', 'carote@example.it',
'+391112200987', '20-01-2018');

-- Insertion of the job order

INSERT INTO JobOrder VALUES (1,'20-02-2018',
'https://www.vicentinaserramenti.it/invoices/123.pdf', '20-04-2018');

-- Insertion of information about external collaborators and materials
purchased

INSERT INTO IsFulfilledBy VALUES ('13243546576', 1, 40, 7);
INSERT INTO IsFulfilledBy VALUES ('10101215873', 1, 24, 7);
INSERT INTO Purchased VALUES (1, '98075643209', '1234569870', 3, 150.00,
'https://www.vicentinaserramenti.it/documents/invoice.pdf');
INSERT INTO Purchased VALUES (1, '12309856743', '9087060590', 5, 370.00,
'https://www.vicentinaserramenti.it/docs/invoice23.pdf');

-- Insertion of the warranty report

INSERT INTO WarrantyReport VALUES
(default,'/www.vicentinaserramenti.it/img/33.jpg',
'/www.vicentinaserramenti.it/reports/45.pdf', 1);
```

Principal Queries

In this section, we report four queries to navigate the database:

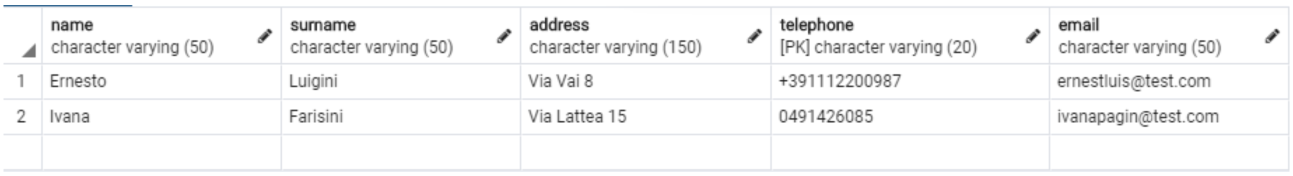
1. Retrieve the quantities of materials purchased from a supplier and the expenditure incurred during a certain period (e.g. 2017-2019).
2. Retrieve contact information to carry out corporate marketing campaigns aimed at a group of customers.
3. Get a statistical report of each marketing campaign or how many customers have been touched by each campaign and have requested a quote.
4. Allows to retrieve data useful to draw up the periodic balance sheets of the company, i.e. expenditure on materials, labour (expenditure) and the price of the estimate paid by the customer (income).

```
-- Recover the quantities of materials purchased from a supplier and the
expenditure incurred during a certain period (e.g. 2017-2019)

SELECT S.companyName AS companyName, M.name AS name, quantity, totalCost
  FROM
      (SELECT P.vat, P.productCode, SUM(P.quantity) AS
      quantity,  SUM(P.quantity*P.cost) AS totalCost
      FROM JobOrder J INNER JOIN Purchased P ON  P.id = J.id
      WHERE  '[2017-01-01,2019-12-31]'::daterange @> J.contractDate
      GROUP BY P.vat, P.productCode) AS G
   INNER JOIN Supplier S ON S.vat=G.vat
   INNER JOIN  Material M ON M.productCode = G.productCode
  ORDER BY S.companyName;
```



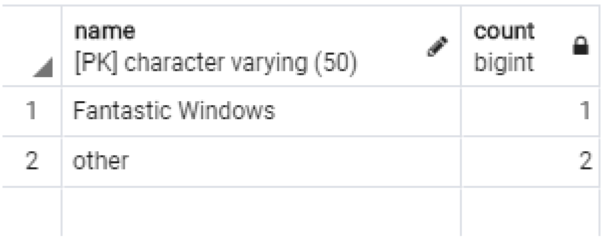
	companyname character varying (50)	name character varying (50)	quantity bigint	totalcost numeric
1	Flamarlegno	Window blue	3	450.00
2	Flamarlegno	Window dark brown	10	12000.00
3	Windows AC Milan	Door white	5	1850.00
4	Windows AC Milan	Gold handle	2	221.00

```
--Retrieve contact information to carry out corporate marketing campaigns
aimed at a specific group of customers (in this case who have bought
doors or windows)

SELECT DISTINCT C.Name, C.Surname, C.Address, C.telephone, C.email
     FROM Customer C INNER JOIN Quote Q ON C.telephone = Q.telephone
          INNER JOIN  JobOrder J ON Q.id = J.id
          INNER JOIN Purchased P ON P.id = J.id
          INNER JOIN Material M ON M.productCode = P.productCode
     WHERE lower(M.Name) LIKE '%window%' OR lower(M.Name) LIKE '%door%';
```



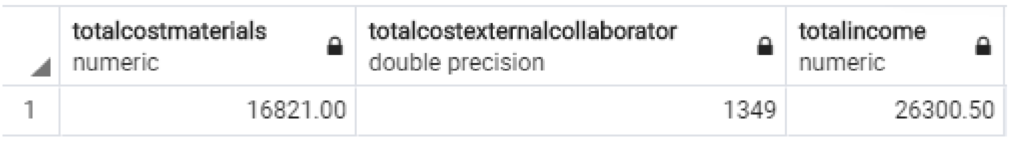
	name character varying (50)	surname character varying (50)	address character varying (150)	telephone [PK] character varying (20)	email character varying (50)
1	Ernesto	Luigini	Via Vai 8	+391112200987	ernestluis@test.com
2	Ivana	Farisini	Via Lattea 15	0491426085	ivanapagin@test.com

```
--Get a statistical report of each marketing campaign or how many
customers have been touched by each campaign and have requested a quote

SELECT M.name, COUNT(*)
FROM MarketingCampaign M INNER JOIN Contacted Co ON M.name = Co.name
     INNER JOIN Customer Cu ON Cu.telephone = Co.telephone
     INNER JOIN Quote Q ON Q.telephone = Cu.telephone
WHERE Co.customerFeedback AND (M.period @> Q.requestDate)
GROUP BY M.name;
```



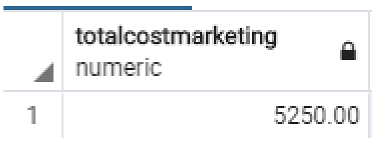
	name [PK] character varying (50)	count bigint
1	Fantastic Windows	1
2	other	2

```
--Retrieve data useful to draw up the periodic balance sheets of the
company, i.e. expenditure on materials, labour (expenditure) and the
price of the estimate paid by the customer (income).
--For efficiency and simplicity reasons we have separated the research of
totalCostMarkting, assuming that all information will be gathered at
application level

SELECT SUM(P.quantity*P.cost) AS totalCostMaterials,
  SUM(F.workingHours*F.costPerHours) AS totalCostExternalCollaborator,
  SUM(Q.totalPrice) AS totalIncome
  FROM Purchased P INNER JOIN JobOrder J ON P.id=J.id
    INNER JOIN isFulFilledBy F ON F.id = J.id
    INNER JOIN ExternalCollaborator E ON E.vat = F.vat
    INNER JOIN Quote Q ON Q.id = J.id
  WHERE '[2018-01-01,2019-12-31]'::daterange @> J.deliveryDate;
```



	totalcostmaterials numeric	totalcostexternalcollaborator double precision	totalincome numeric
1	16821.00	1349	26300.50

```
SELECT SUM(cost) AS totalCostMarketing
      FROM MarketingCampaign
      WHERE '[2018-01-01,2019-12-31]'::daterange @> period;
```



	totalcostmarketing numeric
1	5250.00

JDBC Implementations of the Principal Queries and Visualization

Hereafter, we report a java class which read the data from the database (with reference to the first query) and print the results on screen.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

/**
```

```
 * Allows to recover the quantities of materials purchased from a
supplier and the expenditure incurred
 *
 */
public class PurchasedMaterial {
      /**
       * The JDBC driver to be used
       */
      private static final String DRIVER = "org.postgresql.Driver";
      /**
       * The URL of the database to be accessed
       */
      private static final String DATABASE =
"jdbc:postgresql://localhost/serramenti";
      /**
       * The username for accessing the database
       */
      private static final String USER = "postgres";
      /**
       * The password for accessing the database
       */
      private static final String PASSWORD = "pwd";
      /**
       * The SQL statement to be executed
       */
      private static final String SQL = "SELECT S.companyName AS
companyName, M.name AS name, quantity, totalCost FROM (SELECT P.vat,
P.productCode, SUM(P.quantity) AS quantity,  SUM(P.quantity*P.cost) AS
totalCost FROM JobOrder J INNER JOIN Purchased P ON  P.id = J.id WHERE
'[2017-01-01,2019-12-31]'::daterange @> J.contractDate GROUP BY P.vat,
P.productCode) AS G INNER JOIN Supplier S ON S.vat=G.vat INNER JOIN
Material M ON M.productCode = G.productCode ORDER BY S.companyName;";

      public static void main(String[] args) {
            // the connection to the DBMS
            Connection con = null;
            // the statement to be executed
            Statement stmt = null;
            // the results of the statement execution
            ResultSet rs = null;
            // start time of a statement
            long start;
            // end time of a statement
            long end;
            // "data structures" for the data to be read from the database
            // the data recover
            String companyName = null;
            String name = null;
            Integer quantity = null;
            Float totalCost = null;
            try {
                  // register the JDBC driver
                  Class.forName(DRIVER);
                  System.out.printf("Driver %s successfully
registered.%n", DRIVER);
            } catch (ClassNotFoundException e) {
                  System.out.printf("Driver %s not found: %s.%n", DRIVER,
e.getMessage());
                  // terminate with a generic error code
                  System.exit(-1);
            }
            try {
                  // connect to the database
                  start = System.currentTimeMillis();
                  con = DriverManager.getConnection(DATABASE, USER,
PASSWORD);
                  end = System.currentTimeMillis();
                  System.out.printf("Connection to database %s
successfully established in %,d milliseconds.%n",DATABASE, end-start);
                  // create the statement to execute the query
                  start = System.currentTimeMillis();
                  stmt = con.createStatement();
                  end = System.currentTimeMillis();
                  System.out.printf("Statement successfully created in %,d
milliseconds.%n",end-start);
                  // execute the query
                  start = System.currentTimeMillis();
                  rs = stmt.executeQuery(SQL);
                  end = System.currentTimeMillis();
                  System.out.printf("Query %s successfully executed %,d
milliseconds.%n",SQL, end - start);
                  System.out.printf("Query results:%n");
                  // cycle on the query results and print them
                  while (rs.next()) {
                      companyName = rs.getString("companyName");
                      name = rs.getString("name");
                      quantity = rs.getInt("quantity");
                      totalCost = rs.getFloat("totalCost");
                      System.out.printf("- %s, %s, %d, %f%n",
companyName, name, quantity, totalCost);
                  }
            } catch (SQLException e) {
                  System.out.printf("Database access error:%n");
                  // cycle in the exception chain
                  while (e != null) {
                      System.out.printf("- Message: %s%n",
e.getMessage());
                      System.out.printf("- SQL status code: %s%n",
e.getSQLState());
                      System.out.printf("- SQL error code: %s%n",
e.getErrorCode());
                      System.out.printf("%n");
                      e = e.getNextException();
                  }
            } finally {
                  try {
                      // close the used resources
                      if (rs != null) {
                          start = System.currentTimeMillis();
                          rs.close();
                          end = System.currentTimeMillis();
```

```
                              System.out.printf("Result set successfully
closed in %,d milliseconds.%n",end-start);
                        }
                        if (stmt != null) {
                              start = System.currentTimeMillis();
                              stmt.close();
                              end = System.currentTimeMillis();
                              System.out.printf("Statement successfully
closed in %,d milliseconds.%n",end-start);
                        }
                        if (con != null) {
                              start = System.currentTimeMillis();
                              con.close();
                              end = System.currentTimeMillis();
                              System.out.printf("Connection successfully
closed in %,d milliseconds.%n",end-start);
                        }
                        System.out.printf("Resources successfully
released.%n");

                  } catch (SQLException e) {
                        System.out.printf("Error while releasing
resources:%n");
                        // cycle in the exception chain
                        while (e != null) {
                              System.out.printf("- Message: %s%n",
e.getMessage());
                              System.out.printf("- SQL status code: %s%n",
e.getSQLState());
                              System.out.printf("- SQL error code: %s%n",
e.getErrorCode());
                              System.out.printf("%n");
                              e = e.getNextException();
                        }
                  } finally {
                        // release resources to the garbage collector
                        rs = null;
                        stmt = null;
                        con = null;
                        System.out.printf("Resources released to the
garbage collector.%n");
                  }
            }
            System.out.printf("Program end.%n");
      }
}
```