Project Report

COMP 353

Main Project

Group:

wmc353_4

Members:

Asher Klein 27262442

Sébastien Séguier 27010699

Database Design:

One part of the database focuses on the staff of the company.

The Employee table stores information on all employees. Employees can be full time or part time and the tables FullTime and PartTime store the required extra information on these employees. A table Dependants stores information on dependants of employees currently working at the company.

The Department table stores information on all departments. The tables WorksFor is a relationship between employees and departments recording the start and dates employees worked in a specific department. Another relationship ManagerOf records the the dates a specific employee managed a department.

The other part of the database focuses on the production and sale of items.

Information on Items is listed in the Item table. Each item can come in a variety of colors (all colors the company uses are listed in the Colors table) with all available colors of each item listed in the InColor table. A specific item in a specific color is referred to as a SKU. The Inventory table records the manufacture information of each SKU produced by the company as well as the manufacture date, quantity produced, and retail price. A specific manufacture run of a SKU in inventory is referred by it's lot number denoted LotNb.

The Customer table lists the data of all customers of the company. Information on orders of Customers is stored in the Orders table. Orders can be further broken down into OrderDetails recording which item in inventory was part of the order, referenced by its LotNb. Shipment information on an order is also broken down into OrderDetail and recorded in the Shipment table.

For each order the accounting department creates an invoice stored in the Account table. If the order was made with a payment plan (installment payments) details on the payment plan are recorded in the ReceivablesInstallments table.

Relations:

Account(InvoiceNb, OID, TotalAmount, Paid)

ReceivablesInstallments(InvoiceNb, PaymentDue, Installments)

Colors(Name)

Customer(Name, Address, Phone)

Department(ID, Name, Room, Phone1, Phone2, Fax)

Dependant(SIN, EID, Name, DateOfBirth)

Employee(ID, Name, SIN, Position, Phone, Address, DateOfBirth)

FullTime(ID, Salary)

PartTime(ID, HourlyRate, HoursWorked)

InColor(SKU, IID, CName)

Inventory(LotNb, SKU, DateOfManufacture, NbItems, UnitPrice)

Item(ID, Name)

ManagerOf(EID, StartDate, EndDate, DID)

WorksFor(EID, StartDate, EndDate, DID)

OrderDetail(DetailNb, OID, LotNb, Quantity, Price)

Orders(ID, CustName, OrderDate)

Shipment(OID, DateShipped)

Code:

Table definitions:

```
CREATE TABLE Employee (
	ID INT AUTO_INCREMENT PRIMARY KEY,
	Name VARCHAR(50),
	SIN VARCHAR(50),
	Position VARCHAR(50),
	Phone VARCHAR(50),
	Address VARCHAR(50),
	DateOfBirth DATE
);

CREATE TABLE FullTime (
	ID INT,
	Salary FLOAT,
	PRIMARY KEY (ID),
	FOREIGN KEY (ID) REFERENCES Employee(ID)
);

CREATE TABLE PartTime (
	ID INT,
	HourlyRate FLOAT,
	HoursWorked INT,
	PRIMARY KEY (ID),
	FOREIGN KEY (ID) REFERENCES Employee(ID)
);

CREATE TABLE Dependant (
	EID INT,
	Name VARCHAR(20),
	SIN VARCHAR(50),
	DateOfBirth DATE,
	PRIMARY KEY (SIN),
	FOREIGN KEY (EID) REFERENCES Employee(ID)
);


CREATE TABLE Department (
	ID INT AUTO_INCREMENT PRIMARY KEY,
	Name VARCHAR(20),
	Room VARCHAR(10),
	Phone1 VARCHAR(50),
	Phone2 VARCHAR(50),
        Fax VARCHAR(50)
);

CREATE TABLE ManagerOf (
        EID INT,
        DID INT,
        StartDate DATE,
        EndDate DATE,
        PRIMARY KEY (EID,StartDate),
        FOREIGN KEY (EID) REFERENCES Employee(ID)
);

CREATE TABLE WorksFor (
        EID INT,
        StartDate DATE,
        DID INT,
        EndDate DATE,
        PRIMARY KEY (EID,StartDate),
        FOREIGN KEY (EID) REFERENCES Employee(ID)
);

CREATE TABLE Customer (
        Name VARCHAR(20) PRIMARY KEY,
        Address VARCHAR(50),
        Phone VARCHAR(15)
);

CREATE TABLE Account(
        InvoiceNb INT AUTO_INCREMENT PRIMARY KEY,
        OID INT FOREIGN KEY REFERENCES Orders(ID),
        TotalAmount FLOAT,
        Paid BOOLEAN
);

CREATE TABLE ReceivablesInstallment (
        InvoiceNb INT PRIMARY KEY,
        PaymentDue BOOLEAN,
        Installments FLOAT,
        FOREIGN KEY (InvoiceNb) REFERENCES Account(InvoiceNb)
);

CREATE TABLE Item (
        ID INT AUTO_INCREMENT PRIMARY KEY,
        Name VARCHAR(20)
```

```
	);

CREATE TABLE Colors(
	Name VARCHAR(20) PRIMARY KEY
	);

CREATE TABLE InColor(
	SKU INT
	IID INT FOREIGN KEY REFERENCES Item(ID),
	CName FOREIGN KEY REFERENCES Colors(Name),
	PRIMARY KEY (SKU)
	);

CREATE TABLE Inventory(
	LotNb INT AUTO_INCREMENT PRIMARY KEY,
	SKU INT FOREIGN KEY REFERENCES InColor(SKU),
	DateOfManufacture DATE,
	NbItems INT,
	UnitPrice FLOAT
	);

CREATE TABLE OrderDetail(
	DetailNb INT AUTO_INCREMENT PRIMARY KEY,
	OID INT FOREIGN KEY REFERENCES Orders(ID),
	LotNb INT FOREIGN KEY REFERENCES Inventory(LotNb),
	Quantity INT,
	Price FLOAT,
	);

CREATE TABLE Orders (
	ID INT PRIMARY KEY AUTO_INCREMENT,
	CustName VARCHAR(20) FOREIGN KEY REFERENCES Customer(Name),
	OrderDate DATE
	);

CREATE TABLE Shipment(
	OID INT PRIMARY KEY,
	DateShipped DATE,
	FOREIGN KEY(OID) REFERENCES OrderDetail(DetailNb)
	);
```

Triggers:

```
## Trigger replaces order quantity with max stock available
CREATE TRIGGER Out_Of_Stock
```

```
BEFORE INSERT ON OrderDetail
FOR EACH ROW
BEGIN
        IF (NEW.Quantity > (SELECT Remaining.Ct Ct
                        FROM (SELECT IQ.LotNb LotNb, (SumI - COALESCE(SumO, 0)) Ct
                                FROM(SELECT Inv.LotNb, SUM(Inv.NbItems) SumI
                                        FROM Inventory Inv
                                        GROUP BY Inv.LotNb) IQ LEFT OUTER JOIN
                                        (SELECT OD.LotNb, SUM(OD.Quantity) SumO
                                        FROM OrderDetail OD, Orders O
                                        WHERE O.ID = OD.OID
                                        GROUP BY OD.LotNb) OQ  ON IQ.LotNb = OQ.LotNb) Remaining
                        WHERE Remaining.LotNb = NEW.LotNb)) THEN

                SET NEW.Quantity = Remaining.Ct;
        END IF;
END;
```

Queries for Supported Operations:

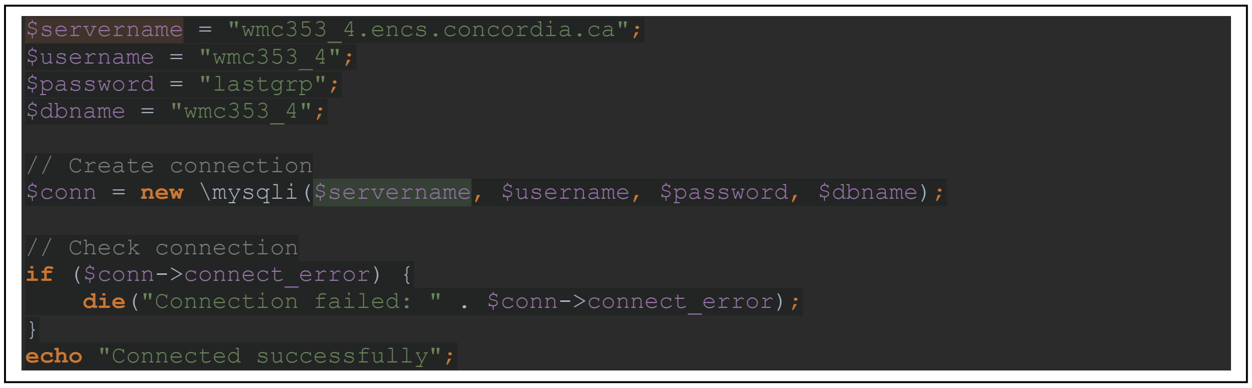
```
 #Queries

#1) need to insert the Dep ID or Name in the php parser,
# Params: department name or id

SELECT  E.ID, E.Name, E.SIN, E.Position, E.Address, E.Phone, E.DateOfBirth, Count(D.SIN)
NumberOfDependants,
        COALESCE((P.HourlyRate * P.HoursWorked * 4), (F.Salary / 12)) MonthlyWage, CASE WHEN
        P.HourlyRate IS NOT NULL THEN 'Part Time' ELSE 'Full Time' END AS Status
FROM ((WorksFor W, Department Dep, (Employee E LEFT OUTER JOIN Dependant D ON (E.ID = D.EID)))
        LEFT OUTER JOIN PartTime P ON E.ID = P.ID) LEFT OUTER JOIN FullTime F ON E.ID = F.ID
WHERE (Dep.ID = 'Sales' OR Dep.Name = 'Sales') AND # The requested Department
        Dep.ID = W.DID AND
        # There is a worksFor tuple between Emp and Dep
        W.EID = E.ID AND
        CURDATE() BETWEEN W.StartDate AND W.EndDate                         # Currently working
GROUP BY E.ID;

#2)
SELECT I.ID, I.Name, SUM(OD.Price) TotalSales, COUNT(OD.Price) NbOrders
FROM Item I, Orders O, OrderDetail OD, Inventory Inv, InColor SK
WHERE I.ID = SK.IID AND SK.SKU = Inv.SKU AND Inv.LotNb = OD.LotNb AND OD.OID = O.ID AND
        O.OrderDate <= CURDATE() AND O.Orderdate >= DATE_ADD(CURDATE(), INTERVAL - 12 MONTH)
GROUP BY I.ID
ORDER BY TotalSales DESC
LIMIT 3;

#3)
# Params: date needed 4 times!
```

```
SELECT I.ID, I.Name, TRUNCATE( SUM( CurCount.Ct * Inv.UnitPrice ) / SUM( CurCount.Ct) , 2 )
AvgPrice
FROM Item I, Inventory Inv, InColor SK, Orders O, OrderDetail OD,
        (SELECT IQ.LotNb LotNb, (SumI - COALESCE(SumO, 0)) Ct
        FROM(SELECT Inv.LotNb, SUM(Inv.NbItems) SumI
                FROM Inventory Inv
                WHERE MONTH("2016-02-19") <= MONTH(Inv.DateOfManufacture) AND YEAR("2016-02-19")
<= YEAR(Inv.DateOfManufacture)
                GROUP BY Inv.LotNb) IQ LEFT OUTER JOIN
                (SELECT OD.LotNb, SUM(OD.Quantity) SumO
                FROM OrderDetail OD, Orders O
                WHERE O.ID = OD.OID AND
                        MONTH("2016-02-19") < MONTH(O.OrderDate) AND YEAR("2016-02-19") <=
YEAR(O.OrderDate)
                GROUP BY OD.LotNb) OQ  ON IQ.LotNb = OQ.LotNb) CurCount
WHERE I.ID = SK.IID AND SK.SKU = Inv.SKU AND Inv.LotNb = OD.LotNb AND Inv.LotNb = CurCount.LotNb
AND CurCount.Ct > 0
GROUP BY I.ID
ORDER BY AvgPrice DESC;


#4) Every tuple will include Customer AND Order information, needs to be parsed for each customer

SELECT C.Name, C.Address, C.Phone, I.Name Item, OD.Price OrderPrice, OD.Quantity OrderQuantity
FROM Customer C, Item I, Orders O, OrderDetail OD, Inventory Inv, InColor SK
WHERE C.Name = O.CustName AND Inv.LotNb = OD.LotNb AND OD.OID = O.ID AND I.ID = SK.IID AND SK.SKU
= Inv.SKU AND
        C.Name IN (SELECT Name
                                FROM (SELECT C.Name Name, SUM(OD.Price) TotalSales
                                        FROM Customer C,Item I, Orders O, OrderDetail OD, Inventory
Inv, InColor SK
                                        WHERE I.ID = SK.IID AND SK.SKU = Inv.SKU AND Inv.LotNb =
OD.LotNb AND OD.OID = O.ID AND C.Name = O.CustName AND
                                                O.OrderDate <= CURDATE() AND O.Orderdate >=
DATE_ADD(CURDATE(), INTERVAL - 12 MONTH)
                                        GROUP BY Name
                                        ORDER BY TotalSales DESC
                                        LIMIT 3
                                        ) Best
                                )
ORDER BY C.Name DESC;

#5)

SELECT C.Name, C.Address, A.InvoiceNb, A.OID OrderID, OD.DetailNb, S.DateShipped,
COALESCE(R.PaymentDue, A.TotalAmount) AmountOwed
FROM (Customer C, Orders O, OrderDetail OD, Shipment S, Account A) LEFT OUTER JOIN
ReceivablesInstallment R ON R.InvoiceNb = A.InvoiceNb
WHERE C.Name = O.CustName AND O.ID = OD.OID AND OD.DetailNb = S.OID AND O.ID = A.OID AND A.Paid =
0
ORDER BY C.Name;

#6) Needs an invoice number as a parameter (last AND in where clause). Converts all nulls to
readable data
# Param: Invoice Nb

SELECT A.InvoiceNb, C.Name, C.Address, C.Phone, O.ID OrderID, O.OrderDate, I.Name Item, SK.CName
Color, Inv.UnitPrice, OD.Quantity, OD.Price ItemTotal, A.TotalAmount, COALESCE(CEIL(A.TotalAmount
/ R.Installments), "Due In Full") NbPayments, COALESCE(R.Installments, "N/A") PaymentAmt,
COALESCE(S.DateShipped, "Not Shipped") DateShipped
FROM (Customer C, Item I, Inventory Inv, Orders O, InColor SK),  Account A LEFT OUTER JOIN
ReceivablesInstallment R ON R.InvoiceNb = A.InvoiceNb, OrderDetail OD LEFT OUTER JOIN Shipment S
ON OD.DetailNb = S.OID
WHERE C.Name = O.CustName AND O.ID = A.OID AND O.ID = OD.OID AND OD.LotNb = Inv.LotNb AND Inv.SKU
= SK.SKU AND SK.IID = I.ID AND A.InvoiceNb = 1;
```

Web User Interface And Database Implementation:

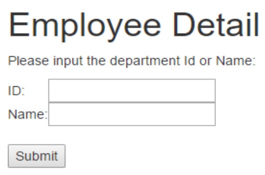
The User interface uses the following languages: HTML, CSS, JavaScript, jQuery, PHP and MySQL.

To link the user interface to the Database mysqli Is used, the connection is established the following way at the beginning of the file:



```
$servername = "wmc353_4.encs.concordia.ca";
$username = "wmc353_4";
$password = "lastgrp";
$dbname = "wmc353_4";

// Create connection
$conn = new \mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
```

We will demonstrate how data retrieval works by illustrating the first requirement: "Given the code or name of a department, print out details about each employee or employees of a department."



Employee Detail

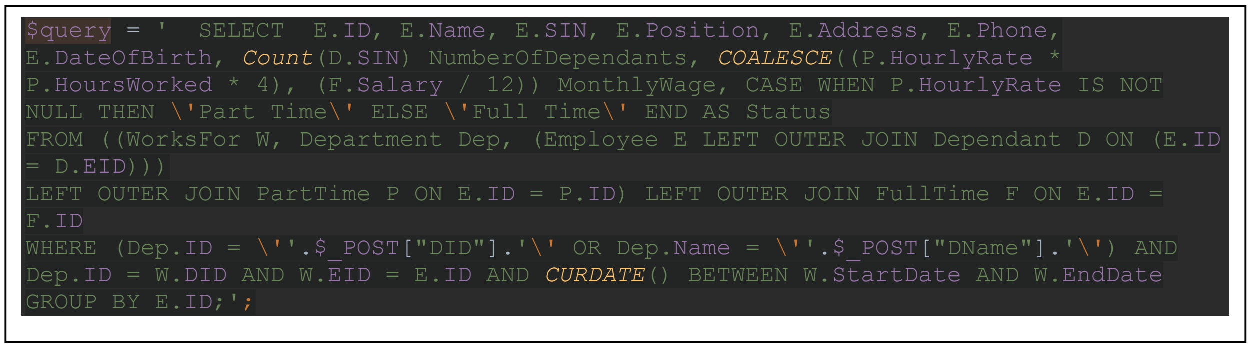
Please input the department Id or Name:

ID:

Name:

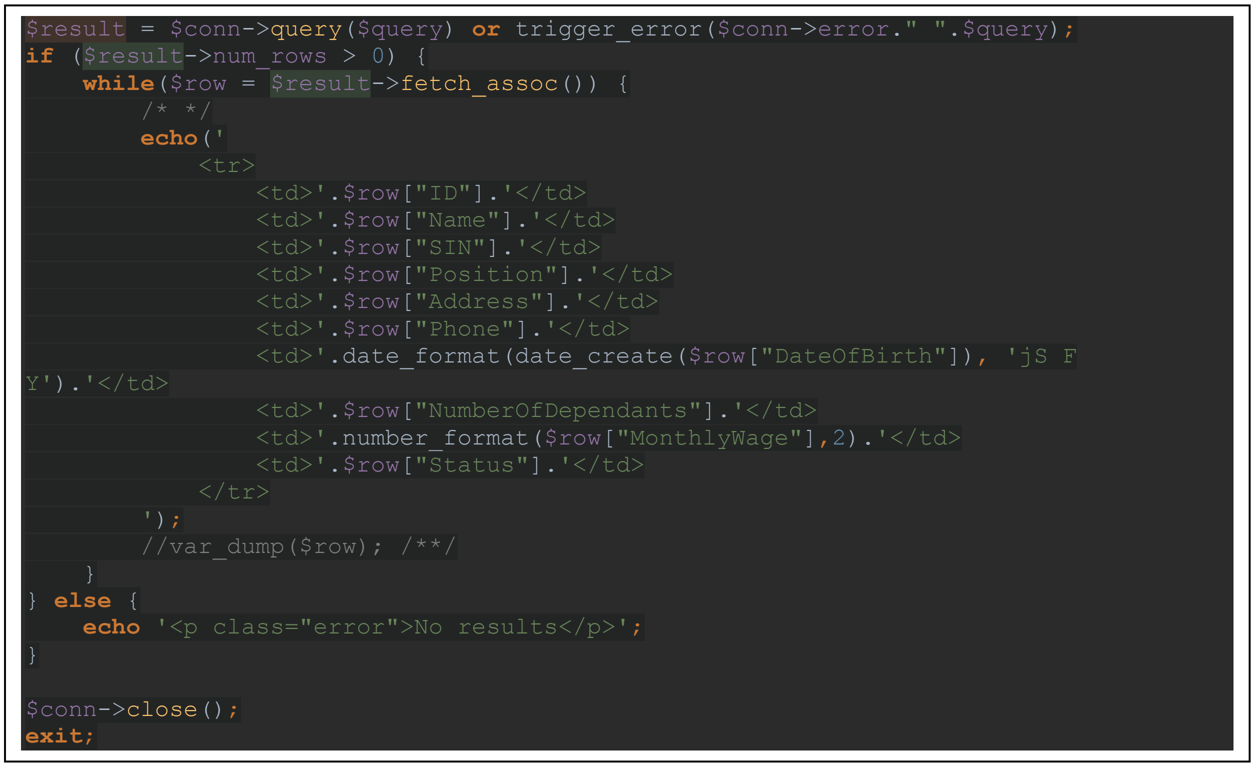
First the user is asked to give a department ID or name.

When he fills the field of his choice and submits an asynchronous call is made to the backend which starts by completing a predefined query with the user's input

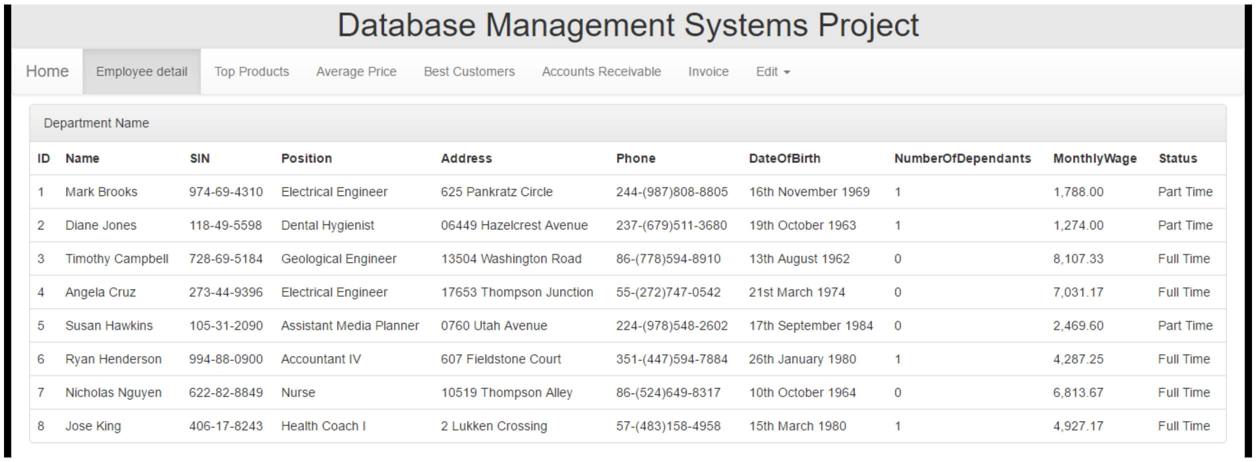


```
$query = '  SELECT  E.ID, E.Name, E.SIN, E.Position, E.Address, E.Phone,
E.DateOfBirth, Count(D.SIN) NumberOfDependants, COALESCE((P.HourlyRate *
P.HoursWorked * 4), (F.Salary / 12)) MonthlyWage, CASE WHEN P.HourlyRate IS NOT
NULL THEN \'Part Time\' ELSE \'Full Time\' END AS Status
FROM ((WorksFor W, Department Dep, (Employee E LEFT OUTER JOIN Dependant D ON (E.ID
= D.EID)))
LEFT OUTER JOIN PartTime P ON E.ID = P.ID) LEFT OUTER JOIN FullTime F ON E.ID =
F.ID
WHERE (Dep.ID = \''.$_POST["DID"].'\' OR Dep.Name = \''.$_POST["DName"].'\') AND
Dep.ID = W.DID AND W.EID = E.ID AND CURDATE() BETWEEN W.StartDate AND W.EndDate
GROUP BY E.ID;';
```

The query is then sent to the database and the data returned by the database is formatted and presented to the user. Once the data was retrieved the connection with the database is closed.



```
$result = $conn->query($query) or trigger_error($conn->error." ".$query);
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        /* */
        echo('
            <tr>
                <td>'.$row["ID"].'</td>
                <td>'.$row["Name"].'</td>
                <td>'.$row["SIN"].'</td>
                <td>'.$row["Position"].'</td>
                <td>'.$row["Address"].'</td>
                <td>'.$row["Phone"].'</td>
                <td>'.date_format(date_create($row["DateOfBirth"]), 'jS F
Y').'</td>
                <td>'.$row["NumberOfDependants"].'</td>
                <td>'.number_format($row["MonthlyWage"],2).'</td>
                <td>'.$row["Status"].'</td>
            </tr>
        ');
        //var_dump($row); /**/
    }
} else {
    echo '<p class="error">No results</p>';
}

$conn->close();
exit;
```



Database Management Systems Project

Department Name

ID	Name	SIN	Position	Address	Phone	DateOfBirth	NumberOfDependants	MonthlyWage	Status
1	Mark Brooks	974-69-4310	Electrical Engineer	625 Pankratz Circle	244-(987)808-8805	16th November 1969	1	1,788.00	Part Time
2	Diane Jones	118-49-5598	Dental Hygienist	06449 Hazelcrest Avenue	237-(679)511-3680	19th October 1963	1	1,274.00	Part Time
3	Timothy Campbell	728-69-5184	Geological Engineer	13504 Washington Road	86-(778)594-8910	13th August 1962	0	8,107.33	Full Time
4	Angela Cruz	273-44-9396	Electrical Engineer	17653 Thompson Junction	55-(272)747-0542	21st March 1974	0	7,031.17	Full Time
5	Susan Hawkins	105-31-2090	Assistant Media Planner	0760 Utah Avenue	224-(978)548-2602	17th September 1984	0	2,469.60	Part Time
6	Ryan Henderson	994-88-0900	Accountant IV	607 Fieldstone Court	351-(447)594-7884	26th January 1980	1	4,287.25	Full Time
7	Nicholas Nguyen	622-82-8849	Nurse	10519 Thompson Alley	86-(524)649-8317	10th October 1964	0	6,813.67	Full Time
8	Jose King	406-17-8243	Health Coach I	2 Lukken Crossing	57-(483)158-4958	15th March 1980	1	4,927.17	Full Time

All the communications with the database are done in a similar fashion.

Database design and queries done by Sébastien Séguier and Asher Klein.

PHP parser done by Sébastien Séguier.

Project report by Asher Klein.