**Εικόνα που περιέχει κείμενο, γραμματοσειρά, λογότυπο, γραφικά

Περιγραφή που δημιουργήθηκε αυτόματα**

**Department:** Department of Management Science and Technology

**M.Sc.:** Business Analytics (Full-Time)

**Course:** Data Management and Business Intelligence

**Professor:** Damianos Chatziantoniou

**Deliverable:** Assignment #1 – SQL and Relational Databases

**Student Name:** Evangelos Lakkas-Pyknis

**Registration Number**: f2822306

**E-mail:** eva.lakkaspyknis@aueb.gr

**Task 1**

Use the Entity-Relationship Diagram (ERD) to model entities, relationships, attributes, cardinalities, and all necessary constraints. Use any tool you like to draw the ERD.

By using the Microsoft Visio tool I designed the following Entity-Relationship Diagram

Εικόνα που περιέχει στιγμιότυπο οθόνης, γραφικά, κύκλος, γραφιστική

Περιγραφή που δημιουργήθηκε αυτόματα

In this ERD we have 5 entities(Customers, Cities, Contracts, Plans and Calls) which are depicted with rectangles, the entities are further described by their attributes which are encircled in ellipses and each entity has a specific attribute which takes unique values (primary key) and identifies each observation for this entity, primary keys are the underlined attributes. These entities are related with each other as described in the description of the case through the diamond shapes. The plurality of each relation is represented by the type of line joining the 2 rectangles (entities) and the diamond (relationship). In this diagram there are only one-to-many/many-to-one type relationships; these relationships are represented by the simple line indicating the "many" and the arrow showing to "one" (i.e. one customer can live in only one city but one city can host many customers, a contract can be assigned to one plan but one plan can be assigned to many contracts etc).

**Task 2**

Create the relational schema in MySQL/SQLServer and insert a few records into the tables to test your queries below. You will have to hand in the CREATE TABLE statements.

I used MySQL to create the relational schema of telcox database, according to the ERD model that I designed previously I created one table for each entity as showed below, under each CREATE TABLE statement I also show the final form that these tables took after the insertion of the data, to insert values I used “INSERT INTO” statement to do batch insertions as shown below for the “cities”:

create database telcox;

use telcox;

CREATE TABLE telcox.cities (

CityID int not null auto\_increment primary key,

Name varchar(60) not null,

Population int not null,

Mean\_Income decimal(6,1) not null

);

INSERT INTO telcox.cities (Name, Population, Mean\_Income)

VALUES

('Athens', 664046, 18900),

('Thessaloniki', 325182, 17200),

('Patras', 167446, 15600),

('Heraklion', 173993, 16800),

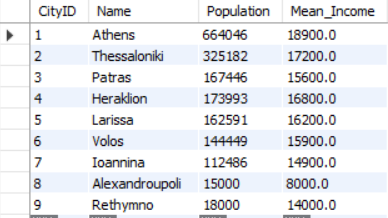
('Larissa', 162591, 16200),

('Volos', 144449, 15900),

('Ioannina', 112486, 14900),

('Alexandroupoli', 15000, 8000),

('Rethymno', 18000, 14000);



CREATE TABLE telcox.customers (

CustomerID int not null auto\_increment primary key,

First\_Name varchar(60) not null,

Last\_Name varchar(60) not null,

Date\_of\_birth DATE not null,

Gender varchar(8) not null,

CITY\_ID int,

foreign key (CITY\_ID)

references telcox.cities(CityID),

check(Gender in('Male', 'Female'))

);

INSERT INTO telcox.customers (First\_Name, Last\_Name, Date\_of\_birth, Gender, CITY\_ID)

VALUES

('Nikos', 'Papadopoulos', '1988-07-20', 'Male', 2),

('Maria', 'Koutroumpis', '1995-03-12', 'Female', 2),

('Giorgos', 'Papadakis', '1990-11-05', 'Male', 3),

('Eleni', 'Katsarou', '1983-09-15', 'Female', 4),

('Dimitris', 'Tassis', '1998-04-30', 'Male', 5),

('Vasiliki', 'Karagianni', '1992-01-18', 'Female', 6),

('Panagiotis', 'Pappas', '1980-12-08', 'Male', 3),

('Giannis', 'Dimitriadis', '1968-02-21','Male', 8),

('Katerina', 'Kazaki', '1975-05-18', 'Female',9);

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

CREATE TABLE telcox.plan (

PlanID INT not null primary key,

Plan\_Name VARCHAR(30) not null,

FreeSMS INT not null check (FreeSMS > 0),

FreeMinutes INT not null check (FreeMinutes > 0),

FreeMB INT not null check(FreeMB > 0)

);

INSERT INTO telcox.plan (PlanID, Plan\_Name, FreeSMS, FreeMinutes, FreeMB)

VALUES

(1, 'Basic Plan', 100, 500, 600),

(2, 'Standard Plan', 200, 1000, 1200),

(3, 'Premium Plan', 300, 1500, 2600),

(4, 'FreedomONE Plan', 300, 500, 1500);

UPDATE plan

SET FreeMinutes=15

WHERE PlanID=1;

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

CREATE TABLE telcox.contract(

ContractID int not null auto\_increment primary key,

PhoneNumber int not null unique,

Description varchar(60) not null,

Start\_Date date not null,

End\_Date date not null,

Customer\_ID int not null,

foreign key(Customer\_ID)

references telcox.customers(CustomerID),

Plan\_ID int not null,

foreign key(Plan\_ID)

references telcox.plan(PlanID)

);

INSERT INTO telcox.contract (PhoneNumber, Description, Start\_Date, End\_Date, Customer\_ID, Plan\_ID)

VALUES

(693123456, 'Contract 1', '2022-01-01', '2023-12-31', 1, 1),

(694987654, 'Contract 2', '2022-02-15', '2024-02-14', 2, 2),

(695555555, 'Contract 3', '2022-03-10', '2023-09-10', 3, 3),

(697111122, 'Contract 4', '2022-04-20', '2024-04-19', 4, 1),

(694333344, 'Contract 5', '2022-06-05', '2023-12-05', 5, 2),

(698555566, 'Contract 6', '2022-07-15', '2024-07-14', 6, 3),

(699777788, 'Contract 7', '2022-09-01', '2024-08-31', 7, 1),

(697053748, 'Freedom Contract','2021-07-25', '2023-12-31', 3, 4),

(697053769, 'Freedom Contract','2021-03-15', '2023-12-31', 6, 4),

(696666666, 'Contract 10', '2023-09-20', '2023-11-19', 7, 1),

(695555587, 'Contract 11', '2023-08-05', '2023-09-04', 8, 2);

**Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα**

CREATE TABLE telcox.calls(

CallID INT not null primary key auto\_increment,

CalledNumber INT not null,

Duration\_In\_Seconds INT not null,

Date\_of\_Call DATETIME not null,

ContractID INT not null,

foreign key(ContractID)

references telcox.contract(ContractID)

);

INSERT INTO telcox.calls (CalledNumber, Duration\_In\_Seconds, Date\_of\_Call, ContractID)

VALUES

(694987654, 300, '2022-01-05 10:30:00', 1),

(695555555, 180, '2022-02-20 15:45:00', 2),

(693123457, 420, '2022-04-12 08:00:00', 3),

(697111122, 240, '2022-06-18 12:30:00', 4),

(694333344, 360, '2022-08-02 20:15:00', 5),

(698555566, 480, '2022-09-20 09:45:00', 6),

(699777788, 600, '2022-11-10 14:00:00', 7),

(699777788, 25, '2022-06-20 08:30:00', 7),

(698555570, 68, '2022-06-20 09:30:00', 5),

(698555570, 20, '2022-06-25 09:30:00',5),

(693123465, 15, '2022-06-23 09:00:00',3),

(698555570, 500, '2022-09-23 09:00:00', 5),

(695478968, 740, '2022-07-15 06:00:00', 4),

(698555570, 620, '2022-10-15 06:00:00',4),

(694568913, 850, '2022-07-15 06:00:00',5),

(698555570, 1200, '2022-07-12 00:00:00',1),

(694568913, 1500, '2022-07-12 18:00:00',1),

(697852345, 678, '2022-05-23 00:00:00', 3),

(694987654, 300, '2021-01-15 09:45:00', 1),

(695555555, 180, '2021-02-25 14:20:00', 2),

(693123456, 420, '2021-03-10 08:30:00', 3),

(697111122, 240, '2021-04-18 11:15:00', 4),

(694333344, 360, '2021-05-05 18:45:00', 5),

(698555566, 480, '2021-06-15 12:00:00', 6),

(699777888, 600, '2021-07-22 16:30:00', 7),

(694986543, 300, '2021-08-08 09:00:00', 1),

(695555555, 180, '2021-09-20 14:45:00', 2),

(693234567, 420, '2021-10-12 08:15:00', 3),

(697111222, 240, '2021-11-18 11:30:00', 4),

(694333444, 360, '2021-12-05 18:00:00', 5),

(694976543, 300, '2021-04-05 10:30:00', 1),

(695555555, 180, '2021-04-15 14:20:00', 2),

(691234567, 420, '2021-04-25 08:30:00', 3),

(697111222, 240, '2021-06-10 11:15:00', 4),

(694333444, 360, '2021-06-20 18:45:00', 5),

(698555566, 480, '2021-06-30 12:00:00', 6),

(699777788, 600, '2021-09-08 16:30:00', 7),

(694987543, 300, '2021-09-15 09:00:00', 1),

(695555555, 180, '2021-09-25 14:45:00', 2),

(693134567, 420, '2021-10-05 08:15:00', 3),

(691211222, 240, '2021-10-18 11:30:00', 4),

(694333444, 360, '2021-10-28 18:00:00', 5),

(694333344, 360, '2022-03-05 18:45:00', 5),

(698555566, 480, '2022-12-15 12:00:00', 6);

**Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, κατάλογος, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, κατάλογος, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα**

The database telcox can also be visualized with the following relational model:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3**

***Task 3a:*** Show the call id of all calls that were made between 8am and 10am on June 2022 having duration < 30

By writing the following code I extracted the table showed below with 3 Call IDs

SELECT CallID

FROM calls

WHERE Date\_of\_Call BETWEEN '2022-06-01 08:00:00' AND '2022-06-30 10:00:00'

AND Duration\_In\_Seconds<30;

The table:

Εικόνα που περιέχει στιγμιότυπο οθόνης, κείμενο, λογισμικό

Περιγραφή που δημιουργήθηκε αυτόματα

|  |
| --- |
|  |

**Task 3b:** Show the first and last name of customers that live in a city with population greater than 20000

SELECT First\_Name, Last\_Name

FROM customers

WHERE CITY\_ID IN(

SELECT CityID

FROM cities

WHERE Population>20000);

The returned table that was generated:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3c:**  Show the customer id that have a contract in the plan with name LIKE ‘Freedom’ (use nested queries).

By using two nested queries I got the IDs of the two customers with the requested plan name

SELECT CustomerID

FROM customers

WHERE CustomerID IN(

SELECT Customer\_ID

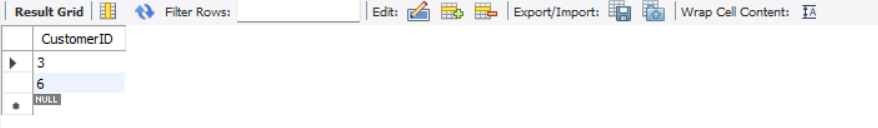
FROM contract

WHERE Plan\_ID IN(

SELECT PlanID

FROM plan

WHERE Plan\_Name LIKE '%Freedom%') );



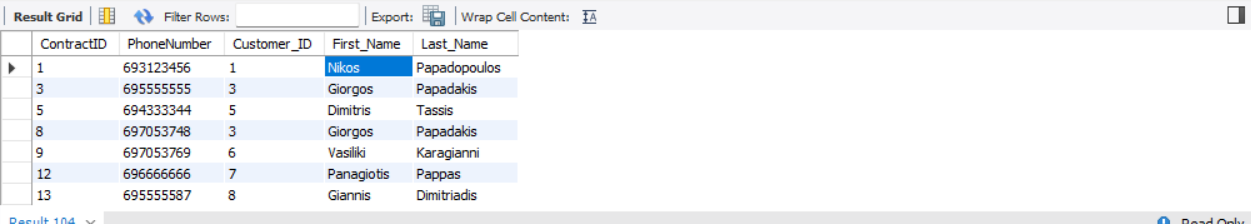
**Task 3d**:For each contract that ends in less than sixty days from today, show the contract id, the phone number, the customer’s id, his/her first name and his/her last name.

SELECT c.ContractID, c.PhoneNumber, c.Customer\_ID, cust.First\_Name, cust.Last\_Name

FROM contract c

JOIN customers cust ON c.Customer\_ID = cust.CustomerID

WHERE datediff(End\_Date, curdate())<60;



**Task 3e:** For each contract id and each month of 2022, show the average duration of calls

SELECT con.ContractID, month(cl.Date\_of\_call) as 'Month of Call', round(avg(cl.Duration\_In\_Seconds),1) as 'Average Duration(in seconds)'

FROM contract con

JOIN calls cl ON con.ContractID = cl.ContractID

WHERE year(Date\_of\_call) = 2022

GROUP BY con.ContractID, month(cl.Date\_of\_call)

ORDER BY con.ContractID asc, month(cl.Date\_of\_call) asc;

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3f:**  Show the total duration of calls in 2022 per plan id

SELECT p.PlanID, sum(cl.Duration\_In\_Seconds) as 'Total Duration in Seconds'

FROM calls cl

JOIN contract con ON cl.ContractID = con.ContractID

JOIN plan p ON p.PlanID = con.Plan\_ID

WHERE year(cl.Date\_of\_call) = 2022

GROUP BY p.PlanID;

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, γραμμή

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3g:** Show the top called number among TP’s customers in 2022

SELECT CalledNumber,

count(\*) as call\_counter

FROM calls

WHERE year(Date\_of\_call) = 2022

GROUP BY CalledNumber

HAVING count(\*) = ( SELECT max(count)

FROM (

SELECT count(\*) AS count

FROM calls

WHERE year(Date\_of\_call) = 2022

GROUP BY CalledNumber

) AS finding\_max

);

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, γραμμή

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3h:** Show the contract ids and the months where the total duration of the calls was greater than the free minutes offered by the plan of the contract

SELECT con.ContractID, month(cl.Date\_of\_Call) as 'Month', sum(cl.Duration\_In\_Seconds) as 'Total Duration(seconds)', p.FreeMinutes

FROM calls cl

JOIN contract con ON con.ContractID = cl.ContractID

JOIN plan p ON con.Plan\_ID = p.PlanID

GROUP BY con.ContractID, month(cl.Date\_of\_call)

HAVING sum(cl.Duration\_In\_Seconds) > p.FreeMinutes\*60

ORDER BY con.ContractID asc, month(cl.Date\_of\_Call) asc;

Εικόνα που περιέχει κείμενο, γραμματοσειρά, γραμμή, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 3i :** For each month of 2022, show the percentage change of the total duration of calls compared to the same month of 2021

SELECT year(Date\_of\_Call) as 'Year', month(Date\_of\_Call) as 'Month', round((sum(Duration\_In\_Seconds)-

LAG(sum(Duration\_In\_Seconds), 12) OVER(ORDER BY year(Date\_of\_Call), month(Date\_of\_Call)))/

LAG(sum(Duration\_In\_Seconds), 12) OVER(ORDER BY year(Date\_of\_Call), month(Date\_of\_Call))\*100,2) as 'Change of the total call Duration (as percentage)'

FROM calls

GROUP BY year(Date\_of\_Call), month(Date\_of\_Call)

ORDER BY year(Date\_of\_Call) desc, month(Date\_of\_Call) asc;

**Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα**

Note: Here I used the lag() function to perform operations for each month of 2022 with the corresponding month of 2021. Also the values of the “Change of the total call Duration” are null for all the months of 2021 (as expected) because 2021 is our base year.

**Task 3j:** For each city id and calls made in 2022, show the average call duration by females and the average call duration by males (i.e. three columns)

SELECT c.CityID,

COALESCE(round(avg(CASE WHEN cust.Gender = 'Female' THEN cl.Duration\_In\_Seconds END), 2),0) AS 'Avg Call Duration of Women',

COALESCE(round(avg(CASE WHEN cust.Gender = 'Male' THEN cl.Duration\_In\_Seconds END), 2),0) AS 'Avg Call Duration of Men'

FROM calls cl

JOIN contract con ON cl.ContractID = con.ContractID

JOIN customers cust ON con.Customer\_ID = cust.CustomerID

JOIN cities c ON cust.CITY\_ID = c.CityID

WHERE YEAR(cl.Date\_of\_Call) = 2022

GROUP BY c.CityID;

**Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα**

Note: Here I used coalesce() function to transform the null values that existed in cities 3,4,5 and 6 for men or women to zero

**Task 3k:** For each city id, show the city id, the ratio of the total duration of the calls made from customers staying in that city in 2022 over the total duration of all calls made in 2022, and the ratio of the city’s population over the total population of all cities (i.e three columns)

SELECT

c2.CityID,

COALESCE((SELECT SUM(cl.Duration\_In\_Seconds)

FROM calls cl

JOIN contract con ON cl.ContractID = con.ContractID

JOIN customers cust ON con.Customer\_ID = cust.CustomerID

JOIN cities c ON cust.CITY\_ID = c.CityID

WHERE YEAR(cl.Date\_of\_Call) = 2022 AND c2.CityID = c.CityID), 0) /

(SELECT SUM(cl2.Duration\_In\_Seconds) FROM calls cl2 WHERE YEAR(cl2.Date\_of\_Call) = 2022) AS 'Call Duration Ratio',

c2.Population / (SELECT SUM(Population) FROM cities) AS 'Population Ratio'

FROM cities AS c2;

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, αριθμός, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

**Task 4:** Using the programming language of your choice, connect to the database and implement query (k) above – ***without using GROUP BY SQL statements***,

I wrote the following code in Python to establish a connection with telcox database (using mysql-connector-python package) and then I created a cursor in order to write and execute the requested SQL query.

import mysql.connector  
  
db\_connection = mysql.connector.connect(  
 host='localhost',  
 user='root',  
 password='password',  
 port='3306',  
 database='telcox',  
 auth\_plugin='mysql\_native\_password'  
)  
  
cursor = db\_connection.cursor()  
query = """  
 SELECT   
 c2.CityID,  
 COALESCE((SELECT SUM(cl.Duration\_In\_Seconds)   
 FROM calls cl  
 JOIN telcox.contract con ON cl.ContractID = con.ContractID  
 JOIN telcox.customers cust ON con.Customer\_ID = cust.CustomerID  
 JOIN telcox.cities c ON cust.CITY\_ID = c.CityID  
 WHERE YEAR(cl.Date\_of\_Call) = 2022 AND c2.CityID = c.CityID), 0) /  
 (SELECT SUM(cl2.Duration\_In\_Seconds) FROM calls cl2 WHERE YEAR(cl2.Date\_of\_Call) = 2022) AS Call\_Duration\_Ratio,  
 c2.Population / (SELECT SUM(Population) FROM cities) AS Population\_Ratio  
 FROM telcox.cities AS c2;   
"""  
  
cursor.execute(query)  
results = cursor.fetchall()  
  
for row in results:  
 CityID, Call\_Duration\_Ratio, Population\_Ratio = row  
 print(f"City ID: {CityID}, Ratio Call Duration: {Call\_Duration\_Ratio}, Ratio City Population: {Population\_Ratio}")  
  
cursor.close()  
db\_connection.close()

The results that I got are the following and they are identical with the result table from MySQL

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα