DMBI (2021) Assignment #1:

Taklakoglou Argyrios

Panagiotis Lolos

Entity-Relationship Model (ER)

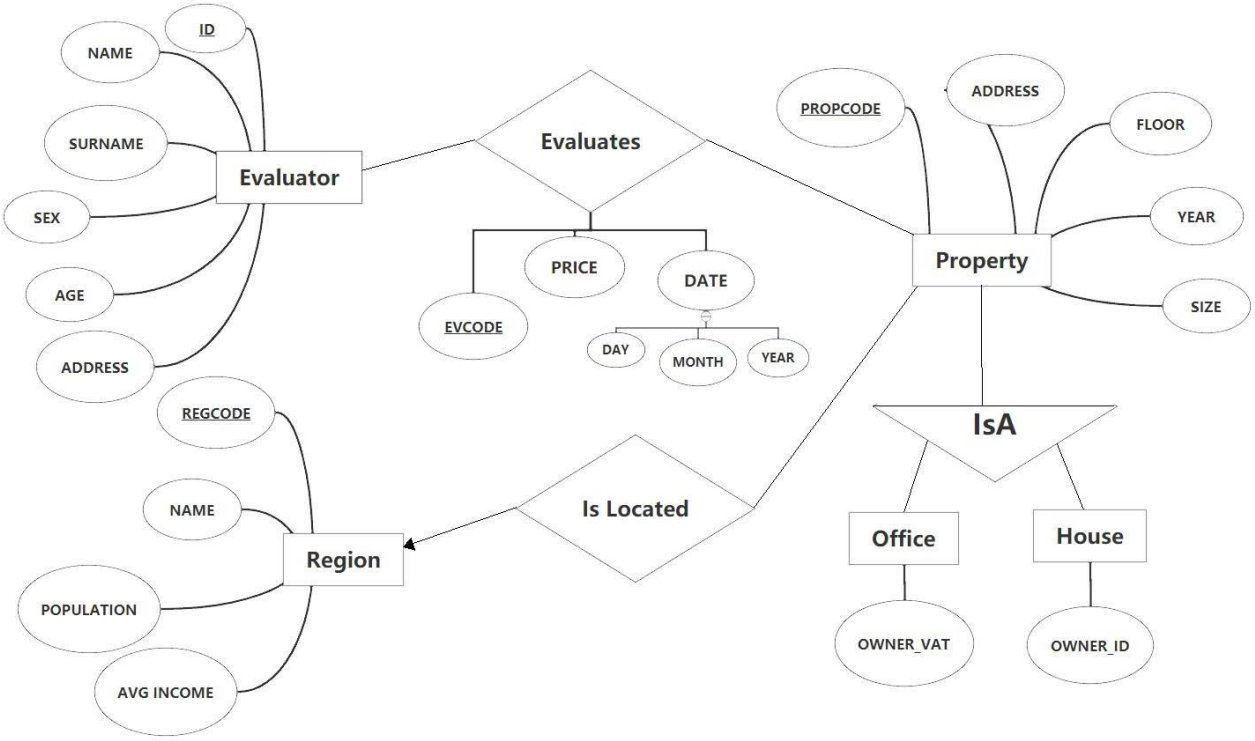

Figure 1: The above model conceptually illustrates the requested database. There are the entities Evaluator (with its primary key its ID), Region (with its primary key REGCODE) and Property (with its primary key PROPCODE), which is inherited by two additional entities: Office and House. There are also two correlations, Evaluates (with EVCODE primary key) linking Evaluator and Property in a many-to-many relationship and Is Located linking Property to a Region in a many-to-one relationship. The diagram was designed with the online designer gitmind.

Relational Model

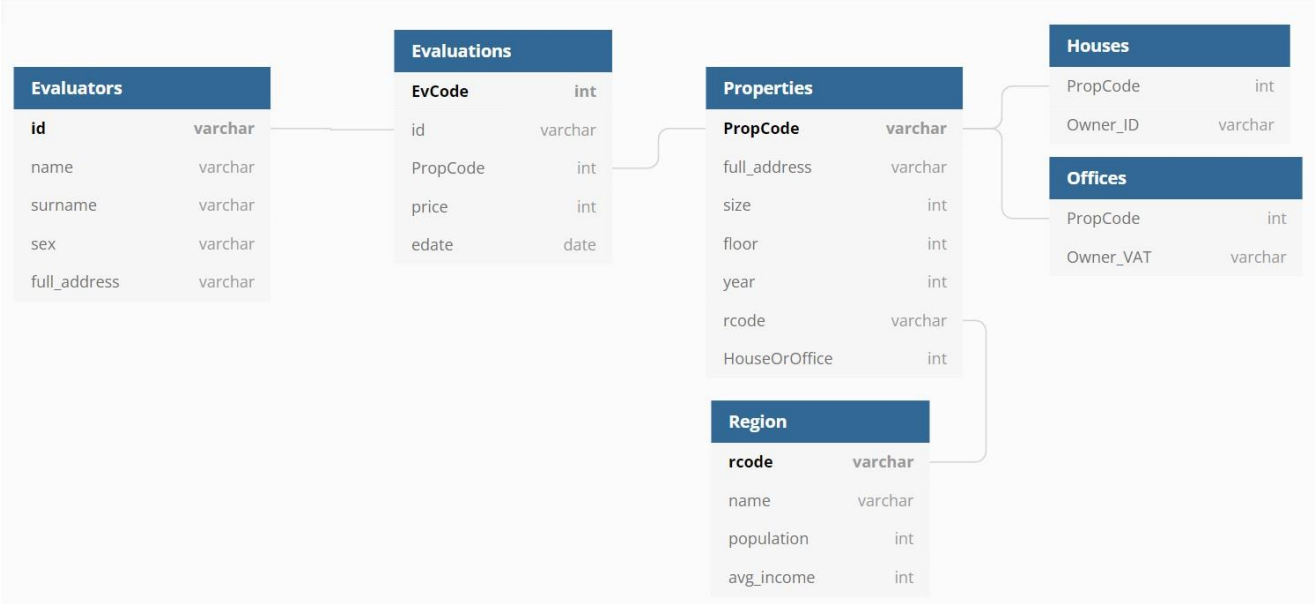

Figure 2: The relational model with tables represents the aforementioned ER. There are six in total, representing entities and correlations. Specifically, the Evaluators entity (with primary key ID) is linked to the Properties entity (with PropCode primary key) via the Evaluations correlation table (with EvCode Primary key). At the same time the Properties table is extracted from the Region table (with rcode primary key) the foreign key rcode and in this way the representation of the previous IsLocated relation of the ER is avoided, which was judged that no additional table is needed. Similarly, the representation of IsA inheritance is avoided by placing an additional foreign PropCode key in the Houses and Offices boards that refer to the properties property.

Assumptions: In addition to avoiding IsLocated and IsA tables the primary keys id (Evaluators), PropCode (Properties) and rcode (Region) were judged to be varchar, as they are usually represented by strings rather than numbers, and will not be included in mathematical operations. . Accordingly, the primary EvCode (Evaluations) key became integer as it is useful to represent and generate as a serial number. Additionally, an additional HouseOrOffice value was created in the Properties panel that takes integer values of 0 or 1 (House or Office respectively) for faster checks on some queries. It could well receive binary prices, but it was judged that in the future there may be other types of real estate (eg empty site, public space, recreational site, etc.) that receive their own prices (2,3,4…).

Database Creation - Importing Values:

The MySQL Workbench environment was used to create the database, where the e-properties1.sql file was created. The file contains the CREATE TABLE commands of the tables described in the relational model, accompanied by the corresponding INSERT INTO for sample values to be used in the queries below. (code ends on page 9)

```
CREATE DATABASE  IF NOT EXISTS `e-propertiesdb`
/*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */
/*!80016 DEFAULT ENCRYPTION='N' */;

USE `e-propertiesdb`;

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
--
-- Table structure for table `evaluator`
--
DROP TABLE IF EXISTS `evaluator`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `evaluator` (
  `evid` varchar(10) NOT NULL,
  `e_name` varchar(45) NOT NULL,
  `e_surname` varchar(45) NOT NULL,
  `age` int(10) DEFAULT NULL,
  `sex` varchar(15) DEFAULT NULL,
```

```
  `address` varchar(45) NOT NULL,
  PRIMARY KEY (`evid`),
  UNIQUE KEY `evid_UNIQUE` (`evid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`evaluator` (`evid`, `e_name`, `e_surname`, `age`, `sex`, `address`) VALUES
('A001', 'GEORGE', 'PANOPOULOS', '37', 'MALE', 'SEIZANI 14'),
('A002', 'JOHN', 'PAPAGEORGIOU', '42', 'MALE', 'PATISION 132'),
('A003', 'THODORIS', 'KARAKOSTAS', '52', 'MALE', 'AXARNON 10'),
('A004', 'PARIS', 'OIKONOMOU', '25', 'MALE', 'LEOF. DEKELIAS 68'),
('A005', 'MARIA', 'NIKOLAOU', '30', 'FEMALE', 'KAPODISTRIOU 4'),
('A006', 'CHRISTINA', 'ANASTASAKOU', '44', 'FEMALE', 'PELOPONISOU 72'),
('A007', 'IOANNA', 'KIOUTZIDOU', '29', 'FEMALE', 'AKADIMIAS 40'),
('A008', 'LINA', 'IOANNOU', '38', 'FEMALE', 'STADIOU 37'),
('A009', 'KOSTANTINA', 'PALAIOLOGOU', '51', 'FEMALE', 'VIZANTIOU 28'),
('A010', 'VASILIOS', 'PAPADIMITRIOU', '39', 'MALE', 'PANEPISTIMIOU 80'),
('A011', 'SPYROS', 'PETRAKIS', '33', 'MALE', 'POSIDONOS 87'),
('A012', 'ANNA', 'KARAKOULAKI', '30', 'FEMALE', 'MARKOU MPOTSARI 31'),
('A013', 'MARIA', 'KOTSIRA', '27', 'FEMALE', 'TRALAION 12'),
('A014', 'JOHN', 'KARAMANOS', '46', 'MALE', 'ANDIANOU 55'),
('A015', 'DIMITRIS', 'PITAS', '24', 'MALE', 'LEOF KIFISIAS 204');
--
-- Table structure for table `property`
--
DROP TABLE IF EXISTS `property`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `property` (
  `propid` varchar(10) NOT NULL,
  `address` varchar(45) NOT NULL,
  `floor` int(10) NOT NULL,
```

```
  `year` int NOT NULL,
  `size` int(10) NOT NULL,
  `location` varchar(10) DEFAULT NULL,
  `houseORoffice` int(4) NOT NULL,  /* indicator for property type. 0  = 'house', 1 = 'office' */
  PRIMARY KEY (`propid`),
  KEY `location_idx` (`location`),
  CONSTRAINT `location` FOREIGN KEY (`location`) REFERENCES `region` (`regid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`property` (`propid`, `address`, `floor`, `year`, `size`, `location`, `houseORoffice`)
VALUES
('HER001', 'LEONIDOU 15', '2', '1972', '80', 'HER', '1'),
('CHA001', 'KISAMOU 42', '1', '1981', '65', 'CHA', '1'),
('ATH001', 'KOLOKOTRONI 63', '4', '2005', '34', 'ATH', '0'),
('THES001', 'MITROPOLAIOS 25', '0', '1985', '97', 'THES', '1'),
('ATH002', 'OLYMPOU 7', '3', '2012', '78', 'ATH', '1'),
('THES002', 'TSIMISKI 71', '3', '2008', '42', 'THES', '0'),
('ATH003', 'EFKALIPTON 16', '4', '2015', '110', 'ATH', '1'),
('THES003', 'AIRATOUS 82', '1', '1990', '25', 'THES', '1'),
('ATH004', 'PARNITHOS 58', '0', '2019', '103', 'ATH', '1'),
('THES004', 'SAADI LEVI 90', '1', '2003', '81', 'THES', '1'),
('ATH005', 'MITROPOLEOS 89', '2', '1998', '39', 'ATH', '0'),
('HER002', 'MINOOS 10', '3', '1975', '36', 'HER', '0'),
('RET001', 'STAMATHIOUDAKI 72', '0', '2001', '40', 'RET', '1'),
('HER003', 'PAPANDRAIOU 50', '1', '1996', '92', 'HER', '1'),
('CHA002', 'SMYRNIS 28', '2', '1988', '59', 'CHA', '1');
--
-- Table structure for table `house`
--
DROP TABLE IF EXISTS `house`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
```

```
CREATE TABLE `house` (
  `house_propid` varchar(10) NOT NULL,
  `owner_id` varchar(45) NOT NULL,
  PRIMARY KEY (`owner_id`),
  UNIQUE KEY `owner_id_UNIQUE` (`owner_id`),
  KEY `propid_idx` (`house_propid`),
  CONSTRAINT `house_propid` FOREIGN KEY (`house_propid`) REFERENCES `property` (`propid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`house` (`house_propid`,`owner_id`) VALUE
('ATH001','AE281566'),
('THES002','AB100782'),
('ATH005','AA780004'),
('HER002','AB100783');
--
-- Table structure for table `office`
--
DROP TABLE IF EXISTS `office`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `office` (
  `office_propid` varchar(10) NOT NULL,
  `owner_vat` varchar(45) NOT NULL,
  PRIMARY KEY (`owner_vat`),
  UNIQUE KEY `owner_vat_UNIQUE` (`owner_vat`),
  KEY `office_propid_idx` (`office_propid`),
  CONSTRAINT `office_propid` FOREIGN KEY (`office_propid`) REFERENCES `property` (`propid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`office` (`office_propid`,`owner_vat`) VALUES
```

```
('HER001', '163556447'),
('CHA001', '172777897'),
('THES001', '454647821'),
('ATH002', '155145195'),
('ATH003', '156222384'),
('THES003', '142988756'),
('ATH004', '178178178'),
('THES004', '252525369'),
('RET001', '146232365'),
('HER003', '137585964'),
('CHA002', '155847999');
--
-- Table structure for table `region`
--
DROP TABLE IF EXISTS `region`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `region` (
  `regid` varchar(15) NOT NULL,
  `name` varchar(45) NOT NULL,
  `population` int(10) NOT NULL,
  `avg_inc` int(10) NOT NULL,
  PRIMARY KEY (`regid`),
  UNIQUE KEY `regid_UNIQUE` (`regid`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`region` (`regid`, `name`, `population`, `avg_inc`) VALUES
 ('ATH', 'ATHENS', '3218218', '25000'),
 ('CHA', 'CHANIA', '108642', '50000'),
 ('HER', 'HERAKLION', '140730', '55000'),
 ('RET', 'RETHYMNO', '45525', '42000'),
```

```
 ('THES', 'THESSALONIKI', '1110912', '19000');

--

-- Table structure for table `evaluation`

--

DROP TABLE IF EXISTS `evaluation`;

/*!40101 SET @saved_cs_client     = @@character_set_client */;

/*!50503 SET character_set_client = utf8mb4 */;

CREATE TABLE `evaluation` (

  `evcode` int NOT NULL AUTO_INCREMENT,

  `price` int(10) NOT NULL,

  `est_date` datetime NOT NULL,

  `evaluator_id` varchar(10) NOT NULL,

  `prop_id` varchar(10) NOT NULL,

  PRIMARY KEY (`evcode`),

  UNIQUE KEY `evcode_UNIQUE` (`evcode`),

  KEY `evaluator_id_idx` (`evaluator_id`),

  KEY `prop_id_idx` (`prop_id`),

  CONSTRAINT `evaluator_id` FOREIGN KEY (`evaluator_id`) REFERENCES `evaluator` (`evid`),

  CONSTRAINT `prop_id` FOREIGN KEY (`prop_id`) REFERENCES `property` (`propid`)

) ENGINE=MyISAM DEFAULT CHARSET=latin1;

/*!40101 SET character_set_client = @saved_cs_client */;

INSERT INTO `e-propertiesdb`.`evaluation` (`evcode`, `price`, `est_date`, `evaluator_id`, `prop_id`) VALUES

('100', '62000', '2020/12/24', 'A005', 'CHA001'),

('101', '67000', '2020/12/31', 'A015', 'CHA001'),

('102', '30000', '2020/12/29', 'A002', 'ATH005'),

('103', '100000', '2020/12/27', 'A001', 'ATH002'),

('104', '110000', '2020/10/15', 'A003', 'ATH002'),

('105', '50000', '2020/08/05', 'A004', 'ATH001'),

('106', '200000', '2019/12/20', 'A005', 'ATH003'),

('107', '220000', '2020/01/15', 'A006', 'ATH003'),

('108', '250000', '2020/01/18', 'A001', 'ATH004'),

('109', '245000', '2019/09/19', 'A002', 'ATH004'),
```

('110', '80000', '2019/10/08', 'A003', 'CHA002'),

('111', '95000', '2019/11/14', 'A007', 'CHA002'),

('112', '78000', '2020/12/28', 'A008', 'HER001'),

('113', '38000', '2020/06/01', 'A009', 'HER002'),

('114', '113000', '2020/12/26', 'A010', 'HER003'),

('115', '60000', '2019/06/02', 'A008', 'RET001'),

('116', '75000', '2020/03/17', 'A011', 'THES001'),

('117', '60000', '2020/12/30', 'A012', 'THES002'),

('118', '54000', '2020/11/20', 'A010', 'THES002'),

('119', '32000', '2020/12/17', 'A007', 'THES003'),

('120', '30000', '2020/12/26', 'A006', 'THES003'),

('121', '28000', '2020/11/08', 'A013', 'THES003'),

('122', '113000', '2019/11/11', 'A014', 'THES004'),

('123', '118000', '2019/12/25', 'A015', 'THES004'),

('124', '111000', '2020/02/13', 'A002', 'THES004'),

('125', '35000', '2020/10/10', 'A015', 'ATH005'),

('126', '47000', '2020/07/26', 'A008', 'ATH001'),

('127', '105000', '2020/08/12', 'A012', 'ATH002'),

('128', '210000', '2020/02/04', 'A001', 'ATH003'),

('129', '225000', '2019/11/29', 'A011', 'ATH003'),

('130', '72000', '2020/03/09', 'A003', 'THES001');

/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;

/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;

/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;

/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

The produced tables:

Evaluator Table:

evid	e_name	e_surname	age	sex	address
A001	GEORGE	PANOPOULOS	37	MALE	SEIZANI 14
A002	JOHN	PAPAGEORGIOU	42	MALE	PATISION 132
A003	THODORIS	KARAKOSTAS	52	MALE	AXARNON 10
A004	PARIS	OIKONOMOU	25	MALE	LEOF. DEKELIAS 68
A005	MARIA	NIKOLAOU	30	FEMALE	KAPODISTRIOU 4
A006	CHRISTINA	ANASTASAKOU	44	FEMALE	PELOPONISOU 72
A007	IOANNA	KIOUTZIDOU	29	FEMALE	AKADIMIAS 40
A008	LINA	IOANNOU	38	FEMALE	STADIOU 37
A009	KOSTANTINA	PALAIOLOGOU	51	FEMALE	VIZANTIOU 28
A010	VASILIOS	PAPADIMITRIOU	39	MALE	PANEPISTIMIOU 80
A011	SPYROS	PETRAKIS	33	MALE	POSIDONOS 87
A012	ANNA	KARAKOULAKI	30	FEMALE	MARKOU MPOTSA...
A013	MARIA	KOTSIRA	27	FEMALE	TRALAION 12
A014	JOHN	KARAMANOS	46	MALE	ANDIANOU 55
A015	DIMITRIS	PITAS	24	MALE	LEOF KIFISIAS 204

Property Table:

propid	address	floor	year	size	location	houseORoffice
HER001	LEONIDOU 15	2	1972	80	HER	1
CHA001	KISAMOU 42	1	1981	65	CHA	1
ATH001	KOLOKOTRONI 63	4	2005	34	ATH	0
THES001	MITROPOLAIOS 25	0	1985	97	THES	1
ATH002	OLYMPOU 7	3	2012	78	ATH	1
THES002	TSIMISKI 71	3	2008	42	THES	0
ATH003	EFKALIPTON 16	4	2015	110	ATH	1
THES003	AIRATOUS 82	1	1990	25	THES	1
ATH004	PARNITHOS 58	0	2019	103	ATH	1
THES004	SAADI LEVI 90	1	2003	81	THES	1
ATH005	MITROPOLEOS 89	2	1998	39	ATH	0
HER002	MINOOS 10	3	1975	36	HER	0
RET001	STAMATHIOUDAK...	0	2001	40	RET	1
HER003	PAPANDRAIOU 50	1	1996	92	HER	1
CHA002	SMYRNIS 28	2	1988	59	CHA	1

House Table:

house_propid	owner_id
ATH001	AE281566
THES002	AB100782
ATH005	AA780004
HER002	AB100783

Office Table:

office_propid	owner_vat
HER001	163556447
CHA001	172777897
THES001	454647821
ATH002	155145195
ATH003	156222384
THES003	142988756
ATH004	178178178
THES004	252525369
RET001	146232365
HER003	137585964
CHA002	155847999

Evaluations Table:

evcode	price	est_date	evaluator_id	prop_id
100	62000	2020-12-24 00:00:00	A005	CHA001
101	67000	2020-12-31 00:00:00	A015	CHA001
102	30000	2020-12-29 00:00:00	A002	ATH005
103	100000	2020-12-27 00:00:00	A001	ATH002
104	110000	2020-10-15 00:00:00	A003	ATH002
105	50000	2020-08-05 00:00:00	A004	ATH001
106	200000	2019-12-20 00:00:00	A005	ATH003
107	220000	2020-01-15 00:00:00	A006	ATH003
108	250000	2020-01-18 00:00:00	A001	ATH004
109	245000	2019-09-19 00:00:00	A002	ATH004
110	80000	2019-10-08 00:00:00	A003	CHA002
111	95000	2019-11-14 00:00:00	A007	CHA002
112	78000	2020-12-28 00:00:00	A008	HER001
113	38000	2020-06-01 00:00:00	A009	HER002
114	113000	2020-12-26 00:00:00	A010	HER003
115	60000	2019-06-02 00:00:00	A008	RET001
116	75000	2020-03-17 00:00:00	A011	THES001
117	60000	2020-12-30 00:00:00	A012	THES002
118	54000	2020-11-20 00:00:00	A010	THES002
119	32000	2020-12-17 00:00:00	A007	THES003
120	30000	2020-12-26 00:00:00	A006	THES003
121	28000	2020-11-08 00:00:00	A013	THES003
122	113000	2019-11-11 00:00:00	A014	THES004
123	118000	2019-12-25 00:00:00	A015	THES004
124	111000	2020-02-13 00:00:00	A002	THES004
125	35000	2020-10-10 00:00:00	A015	ATH005
126	47000	2020-07-26 00:00:00	A008	ATH001
127	105000	2020-08-12 00:00:00	A012	ATH002
128	210000	2020-02-04 00:00:00	A001	ATH003
129	225000	2019-11-29 00:00:00	A011	ATH003
130	72000	2020-03-09 00:00:00	A003	THES001

SQL Questions (queries):

Queries are listed in images along with their corresponding results when executed at the data base.

Task_a

```
#Task_a
SELECT DISTINCT p.propid AS "Property" , p.address AS "Address"
FROM property AS p, region AS r, evaluation AS e
WHERE p.location = r.regid
    AND r.avg_inc> 40000
    AND p.propid = e.prop_id
    AND e.est_date BETWEEN '2020-12-24' AND '2020-12-31';
```

Property	Address
CHA001	KISAMOU 42
HER001	LEONIDOU 15
HER003	PAPANDRAIOU 50

Task_b

```
#Task_b
CREATE VIEW myV (e, con) AS
SELECT e.evid, count(*)
FROM evaluator AS e, evaluation as ev
WHERE  e.evid = ev.evaluator_id
    AND YEAR(est_date) = 2020
GROUP BY e.evid;

SELECT evid AS "Evaluator", IFNULL(con,0) AS "Total Evaluations for 2020"
FROM evaluator
LEFT JOIN myV ON evid = e
ORDER BY evid;
```

Evaluator	Total Evaluations for 2020
A001	3
A002	2
A003	2
A004	1
A005	1
A006	2
A007	1
A008	2
A009	1
A010	2
A011	1
A012	2
A013	1
A014	0
A015	2

Task_c

```
#Task_c
SELECT propid AS "Property", count(*) AS "Evaluations"
FROM property AS p, evaluation as e
WHERE p.propid = e.prop_id
    AND YEAR(est_date) = 2020
GROUP BY propid
HAVING count(*) > 2;
```

Property	Evaluations
ATH002	3
THES003	3

Task_d

```
#Task_d
SELECT evcode AS "Evaluation"
FROM evaluation
WHERE prop_id IN (SELECT propid
            FROM property
            WHERE location IN (SELECT regid
                        FROM region
                        WHERE avg_inc> 25000));
```

	Evaluation
▶	100
	101
	110
	111
	112
	113
	114
	115

Task_e

```
#Task_e
SELECT  count(*) AS "#of Evaluations"
FROM evaluation AS e, property AS p, region AS r
WHERE e.prop_id = p.propid AND
    p.location = r.regid AND
    r.population> 50000 AND
    YEAR(e.est_date) = 2020;
```

	#of Evaluations
▶	23

Task_f

```
#Task_f
SELECT r.regid AS 'Region', (AVG(e.price))/p.size AS 'Price/square meter'
FROM region AS r, evaluation AS e, property AS p
WHERE e.prop_id = p.propid AND p.location = r.regid
GROUP BY r.regid
ORDER BY (AVG(e.price))/p.size ASC;
```

	Region	Price/square meter
▶	THES	714.43298969
	HER	954.16666625
	CHA	1169.23076923
	RET	1500.00000000
	ATH	4133.48416176

Task_g

```
#Task_g
CREATE VIEW V1(Evaluators, count_Houses) as
SELECT tor.evid AS Evaluators, count(*) AS count_Houses
FROM evaluator AS tor, evaluation AS ion, property AS p, house AS h
WHERE ion.evaluator_id = tor.evid AND ion.prop_id = p.propid
 AND h.house_propid = p.propid
 AND YEAR(ion.est_date) = 2020 AND p.houseORoffice = 0
GROUP BY tor.evid;

CREATE VIEW V2(Evaluators, count_Offices) as
SELECT tor.evid AS Evaluators, count(*) AS count_Offices
FROM evaluator AS tor, evaluation AS ion, property AS p, office AS o
WHERE ion.evaluator_id = tor.evid AND ion.prop_id = p.propid
 AND o.office_propid = p.propid
 AND YEAR(ion.est_date) = 2020 AND p.houseORoffice = 1
GROUP BY tor.evid;

SELECT evid AS "Evaluator ID",
 IFNULL(count_Houses, 0) AS "Houses Evaluated",
 IFNULL(count_Offices, 0) AS "Offices Evaluated"
FROM evaluator
LEFT JOIN V1 ON evid = V1.Evaluators
LEFT JOIN V2 ON evid = V2.Evaluators;
```

Evaluator ID	Houses Evaluated	Offices Evaluated
A001	0	3
A002	1	1
A003	0	2
A004	1	0
A005	0	1
A006	0	2
A007	0	1
A008	1	1
A009	1	0
A010	1	1
A011	0	1
A012	1	1
A013	0	1
A014	0	0
A015	1	1

Task_h

```
#Task_h
CREATE VIEW reg2019 (REG,PSM)
AS
SELECT r.regid AS 'REG', (AVG(e.price))/p.size AS 'PSM'
FROM region AS r, evaluation AS e, property AS p
WHERE e.prop_id = p.propid AND p.location = r.regid AND YEAR(e.est_date) = 2019
GROUP BY r.regid
ORDER BY (AVG(e.price))/p.size ASC;

CREATE VIEW reg2020 (REG,PSM)
AS
SELECT r.regid AS 'REG', (AVG(e.price))/p.size AS 'PSM'
FROM region AS r, evaluation AS e, property AS p
WHERE e.prop_id = p.propid AND p.location = r.regid AND YEAR(e.est_date) = 2020
GROUP BY r.regid
ORDER BY (AVG(e.price))/p.size ASC;

SELECT r.regid as "Region",
 IFNULL(reg2020.PSM-reg2019.PSM,0) as "Diff in 2019-2020"
FROM region AS r
LEFT JOIN reg2019 ON r.regid = reg2019.REG
LEFT JOIN reg2020 ON r.regid = reg2020.REG
GROUP BY r.regid;
```

Region	Diff in 2019-2020
ATH	1372.63814647
CHA	-490.74315515
HER	0.00000000
RET	0.00000000
THES	-830.56510119

Task_i

```
#task i
CREATE VIEW TEV (REG, Teval) AS
SELECT r.regid AS 'Region', concat(round(( count(*)/
(SELECT count(*) AS 'Total Evaluations'
FROM evaluation as e
WHERE YEAR(e.est_date) = 2020) *100),2),'%') AS "RegEvals"
FROM region as r, evaluation as ev, property as p
WHERE r.regid = p.location AND p.propid = ev.prop_id AND YEAR(ev.est_date) = 2020
GROUP BY r.regid;

CREATE VIEW TPOP (REG,Tpopul) AS
SELECT r.regid AS 'Region', concat(round((r.population/
(SELECT sum(r.population) AS "Total Population"
FROM region as r) *100),2),'%') AS "% of Total Population"
FROM region as r
GROUP BY r.regid;

SELECT regid AS 'Region' , IFNULL(TEV.Teval,concat(0.0,'%')) AS "% of Total Evaluations",
 IFNULL(TPOP.Tpopul,concat(0.0,'%')) AS "% of Total Population"
FROM region
LEFT JOIN TEV ON TEV.REG = regid
LEFT JOIN TPOP ON TPOP.REG = regid;
```

	Region	% of Total Evaluations	% of Total Population
▶	ATH	43.48%	69.60%
	CHA	8.70%	2.35%
	HER	13.04%	3.04%
	RET	0.0%	0.98%
	THES	34.78%	24.02%

Question I using a combination of SQL and another programming language (without using GROUP BY)

For this query, Script python was created using only the mysql.connector library. The script connects to the database (assuming the database already runs on SQL Server), and saves the Regions, Properties, and Evaluations tables with three consecutive Fetches and a unique condition (where needed) for the Properties to participate in Evaluations made in 2020. Then, the rest of the filtering and the calculation of the percentage values (%) are done exclusively using python. The end result is a list of lists with columns "RegionID", "% of Total Population", "% of Total Evaluations" containing exactly the same results as the equivalent SQL query of Task_i.

Note: we avoided using pandas dataframe as we wanted to use as simple tools as possible in the implementation.

The full code (with annotation) is below:

```
import mysql.connector

#establish connection via socket to server, assign a database
mydb = mysql.connector.connect(
    host = "localhost",
    user= "root",
    password = "Yourpassword",  #password must be changed according to whatever is set in the root server
    database = "e-propertiesdb"
)

mycursor = mydb.cursor()    #create parsing cursor
```

```
#Alternative SQL queries. To create a multiple line string use """\ at start and """ at the end
myquery = """\
SELECT *
FROM region;"""
#myvals = ()
mycursor.execute(myquery)

#Fetch the table returned by executing myquery in the MySQL server. Table in this case contains all regions
regions = mycursor.fetchall()

#Print to check correctly fetched data
print("Region IDs, Regions, Population, AVG Inc")
for x in regions:
    print(x)
```

```
myquery = """\
SELECT *
FROM evaluation
WHERE YEAR(evaluation.est_date) = %s;
"""
myvals = ("2020",) #separate tuple for explicit values to avoid SQL injection
mycursor.execute(myquery,myvals)

#Fetch the table returned by executing myquery in the MySQL server. Table in this case contains all evaluations in the year 2020
evaluations = mycursor.fetchall()

print("Evaluations:")
for x in evaluations:
    print(x)
```

```
myquery = """\
SELECT DISTINCT p.*
FROM property as p, evaluation as e
WHERE p.propid = e.prop_id AND YEAR(e.est_date) = %s;
"""
myvals = ("2020",) #separate tuple for explicit values to avoid SQL injection
mycursor.execute(myquery,myvals)

#Fetch the table returned by executing myquery in the MySQL server. Table in this case contains all properties evaluated in the year 2020
properties = mycursor.fetchall()

print("Properties:")
for x in properties:
    print(x)
```

```
#Database socket is shut down to minimize connection time
mycursor.close()

#From now on, only python and list processing is used to create the end result:
result = list()
totalpop = 0
for x in regions:           #store all region IDs into newly created list of lists
    result.append([x[0]])
    totalpop += x[2]        #calculate total population of all regions

for x in range(0,len(regions)): #for each region, calculate its' population as a % of the total population
    result[x].append(regions[x][2]/totalpop*100)
    result[x].append(0)     #initialize a 0 value to be used as a counter of total evaluations

totalevals = len(evaluations) #total evaluations is the length of the list evaluations

for x in evaluations:       #iterate through evaluations
    id = x[4]
    for y in properties:
        if y[0] == id:      #find the property evaluated, will use the region number to access result list
            for z in result:
                if y[5] == z[0]:    #access result list
                    z[2] += 1       #increment counter

for x in result:
    x[2] = x[2]/totalevals*100 #replace all counters with their representations as a % of total evaluations

result.insert(0,["RegionID","% of Total Population","% of Total Evaluations"]) #add column names as requested
```