DATABASE ‘MEETINGS’

CodeFirstGirls Data Science and SQL course project

By Eleonora Agnelli

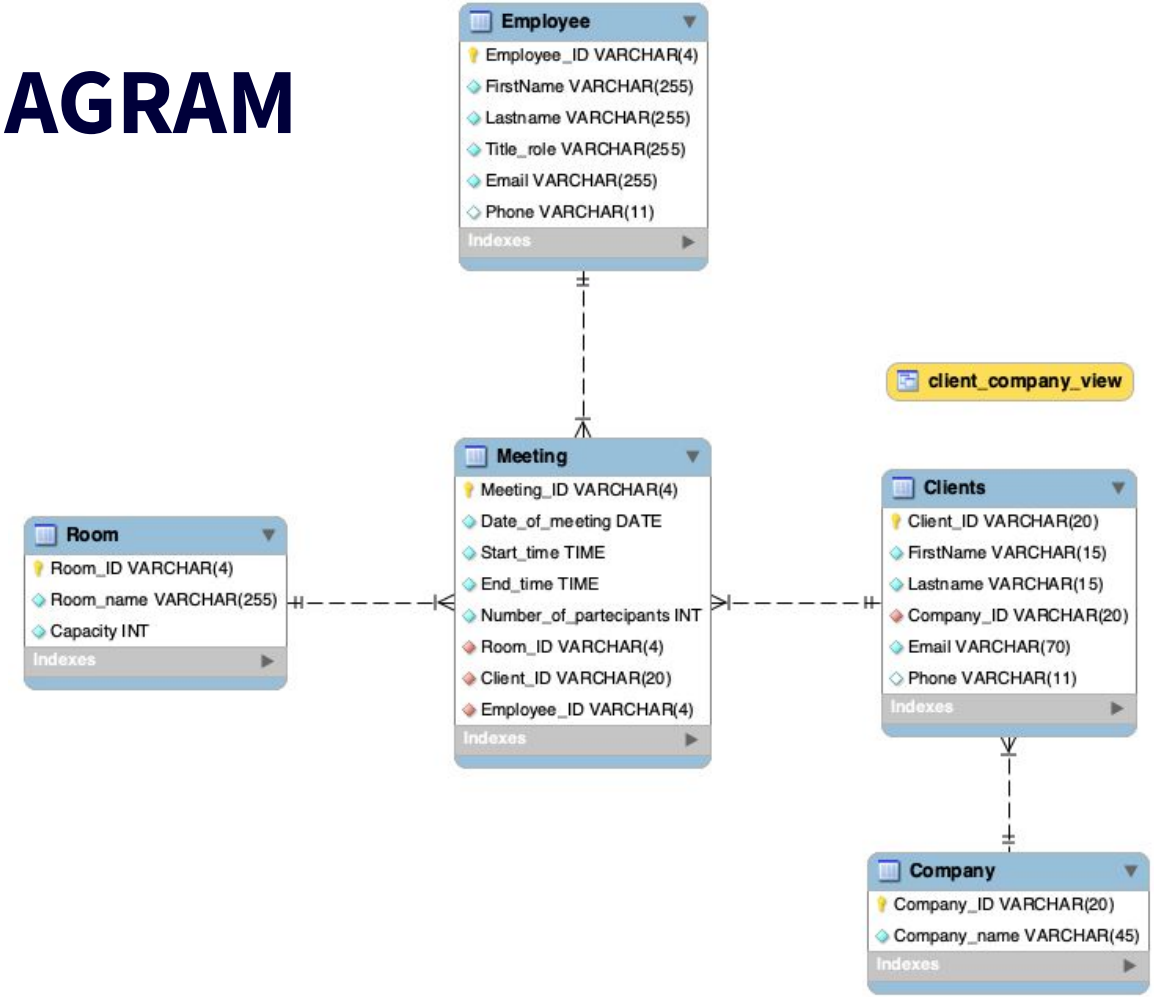
DATABASE OVERVIEW

A database designed specifically for the front-of-house desk to manage initial interactions with clients.

They require to know:

- **General information inherent to the meeting itself** (date, time, number of participants, client invited, employee hosting the meeting, and in which room it's held in).
- **Information of the participants:**
  - Employee details (name, role, work email, phone number)
  - Client details (name, work email, phone number) and **company** they work for.
- **And the details of the room:**
  - name of the room, and its capacity.

ER DIAGRAM


Employee
Employee_ID VARCHAR(4)
FirstName VARCHAR(255)
Lastname VARCHAR(255)
Title_role VARCHAR(255)
Email VARCHAR(255)
Phone VARCHAR(11)
Indexes
client_company_view
Meeting
Meeting_ID VARCHAR(4)
Date_of_meeting DATE
Start_time TIME
End_time TIME
Number_of_partecipants INT
Room_ID VARCHAR(4)
Client_ID VARCHAR(20)
Employee_ID VARCHAR(4)
Indexes
Room
Room_ID VARCHAR(4)
Room_name VARCHAR(255)
Capacity INT
Indexes
Clients
Client_ID VARCHAR(20)
FirstName VARCHAR(15)
Lastname VARCHAR(15)
Company_ID VARCHAR(20)
Email VARCHAR(70)
Phone VARCHAR(11)
Indexes
Company
Company_ID VARCHAR(20)
Company_name VARCHAR(45)
Indexes

CREATE TABLES

```sql
CREATE TABLE Company(
    Company_ID    VARCHAR(20) NOT NULL PRIMARY KEY,
    Company_name   VARCHAR(45) NOT NULL
);

CREATE TABLE Room(
    Room_ID VARCHAR(4) NOT NULL  PRIMARY KEY,
    Room_name VARCHAR(255) NOT NULL,
    Capacity INT NOT NULL
);

CREATE TABLE Employee(
    Employee_ID VARCHAR(4) NOT NULL  PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    Lastname VARCHAR(255) NOT NULL,
    Title_role VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL,
    Phone VARCHAR(11) NULL
);
```

```sql
CREATE TABLE Clients(
    Client_ID   VARCHAR(20) NOT NULL PRIMARY KEY,
    FirstName   VARCHAR(15) NOT NULL,
    Lastname VARCHAR(15) NOT NULL,
    Company_ID   VARCHAR(20) NOT NULL,
    Email VARCHAR(70) NOT NULL,
    Phone VARCHAR(11) NULL,
    FOREIGN KEY (Company_ID) REFERENCES Company (Company_ID)
);

CREATE TABLE Meeting(
    Meeting_ID   VARCHAR(4) NOT NULL PRIMARY KEY,
    Date_of_meeting DATE NOT NULL,
    Start_time TIME NOT NULL,
    End_time   TIME NOT NULL,
    Number_of_partecipants INT NOT NULL,
    Room_ID VARCHAR(4) NOT NULL,
    Client_ID VARCHAR(20) NOT NULL,
    Employee_ID VARCHAR(4) NOT NULL,
    FOREIGN KEY (Room_ID) REFERENCES Room (Room_ID),
    FOREIGN KEY (Client_ID) REFERENCES Clients (Client_ID),
    FOREIGN KEY (Employee_ID) REFERENCES Employee (Employee_ID)
);
```

CREATE A VIEW WITH JOIN

```sql
/* ------------------------------------------------------------------
CREATE A VIEW THAT COMBINES MULTIPLE TABLES USING ANY TYPE OF JOINS
------------------------------------------------------------------
creates a view showing solely the clients' names and the name of the company they work for */
CREATE VIEW client_company_view AS
SELECT T1.FirstName, T1.LastName, T2.Company_name FROM Clients T1
INNER JOIN Company T2 ON T1.Company_ID = T2.Company_ID;

SELECT * FROM client_company_view;
```

	FirstName	LastName	Company_name
▶	Souris	Ratatuille	Milk & Co
	Giorno	Giova	PurpleStar
	Toro	Kujo	PurpleStar
	Daphne	Viola	Flower Arranged
	Pam	Rose	Dunder Muffin
	Jim	Funny	Dunder Muffin
	Otto	Pongo	Brook 9-9
	Tommy	Nook	Nook's
	Timmy	Nook	Nook's
	Tyler	Hyde	Hades Undercrib
	Zagareus	Hades	Hades Undercrib
	Adam	Ren	Life Star
	Kylo	Driver	Life Star
	Darty	Skyguy	Life Star
	Mando	Fett	The Mandorin
	Venti	Bardo	Devil's Share
	Diluc	Saft	Devil's Share

Creates a view called '**client_company_view**'
It takes as parameters the **CLIENTS** and **COMPANY tables**, using the **COMPANY_ID** as a link to find a connection between them.

It then showcase the clients first name, last name(*from CLIENT table*), and the name of the company they work for(*from COMPANY table*).

CREATE A STORED FUNCTION

```
/* ------------------------------------------------------------------------
CREATE A STORED FUNCTION
------------------------------------------------------------------------
Each day the Reception desk needs to know all the meetings happening on the day,
or they might need to check past/future meetings on a specific day */
DELIMITER //
CREATE FUNCTION meetings_happening(Date_of_meeting DATE)
RETURNS VARCHAR(300)
DETERMINISTIC
BEGIN
  DECLARE output VARCHAR(300);
  SELECT CONCAT('TIME: ', T1.Start_time, ' - ', T1.End_time, ' CLIENT: ',
                T2.FirstName, ' ', T2.Lastname, ' FROM COMPANY: ', T3.Company_name,
                ' IN ROOM: ', T4.Room_name)
    INTO output
    FROM Meeting T1
    INNER JOIN Clients T2 ON T1.Client_ID = T2.Client_ID
    INNER JOIN Company T3 ON T3.Company_ID = T2.Company_ID
    INNER JOIN Room T4 ON T1.Room_ID = T4.Room_ID
    WHERE T1.Date_of_meeting = Date_of_meeting;
  RETURN output;
END//
DELIMITER ;

SELECT meetings_happening('2022-04-22');
```

Creates a stored function called ‘**meetings_happening’** which consider a specific date we give it to.

It then stores in ‘output’: the timings of the meeting(*from MEETING table*), the client name(*from CLIENT table*), the name of the company(*from COMPANY table*), and the room name(*from ROOM table*).

Using **JOINS** to link all the information.

```
meetings_happening('2022-04-22')
▶ TIME: 16:00:00 - 17:00:00 CLIENT: Pam Rose FROM COMPANY: Dunder Muffin IN ROOM: King's Cross
```

MULTIPLE TABLE STATEMENTS (subqueries)

```sql
/* ---------------------------------------------------------------------
MULTIPLE TABLE STATEMENTS (subqueries)
--------------------------------------------------------------------- */
Show which meetings took place in London Bridge from the most recent to the oldest. */
SELECT Meeting_ID, Date_of_meeting FROM Meeting WHERE Room_ID IN (SELECT Room_ID FROM Room WHERE Room_name = "London Bridge")
ORDER BY Date_of_meeting DESC;


/* Show which meeting was held with the company "Brook 9-9".  */
SELECT * FROM Meeting
WHERE Client_ID IN (SELECT Client_ID FROM Clients
                    WHERE Company_ID IN (SELECT Company_ID FROM Company WHERE Company_name = "Brook 9-9"));


/* Count how many clients the company is in contact with from the company which ID is "CP06". */
SELECT COUNT(Client_ID) AS N_of_Clients_working_for_Nooks FROM Clients
WHERE Company_ID IN (SELECT Company_ID FROM Company WHERE Company_ID = "CP06");


/* Show which clients (their IDs and names) had a meeting in November. */
SELECT Client_ID, FirstName, Lastname
FROM Clients
WHERE Client_ID IN (SELECT Client_ID FROM Meeting WHERE
                    MONTH(Date_of_meeting) = 11);
```

EXTRA: GROUP BY - HAVING

This statement joins together the MEETING, CLIENTS and COMPANY tables.

Groups the meetings by company, and counts the number of meetings they have been part of.

It then considers only the companies that have participated in more than 1 meeting and displays their name and frequency of visits.

```sql
/* ---------------------------------------------------------------------
EXTRA: Group by - Having
---------------------------------------------------------------------
Finds what companies have had more than meeting and show frequency of meetings they had. */
SELECT Company.Company_name, COUNT(*) AS number_of_meetings
FROM Meeting
JOIN Clients ON Meeting.Client_ID = Clients.Client_ID
JOIN Company ON Clients.Company_ID = Company.Company_ID
GROUP BY Company.Company_name
HAVING COUNT(Meeting.Meeting_ID) > 1;
```

Company_name	number_of_meetings
PurpleStar	2
Dunder Muffin	2
Nook's	2

Thank you

Eleonora Agnelli