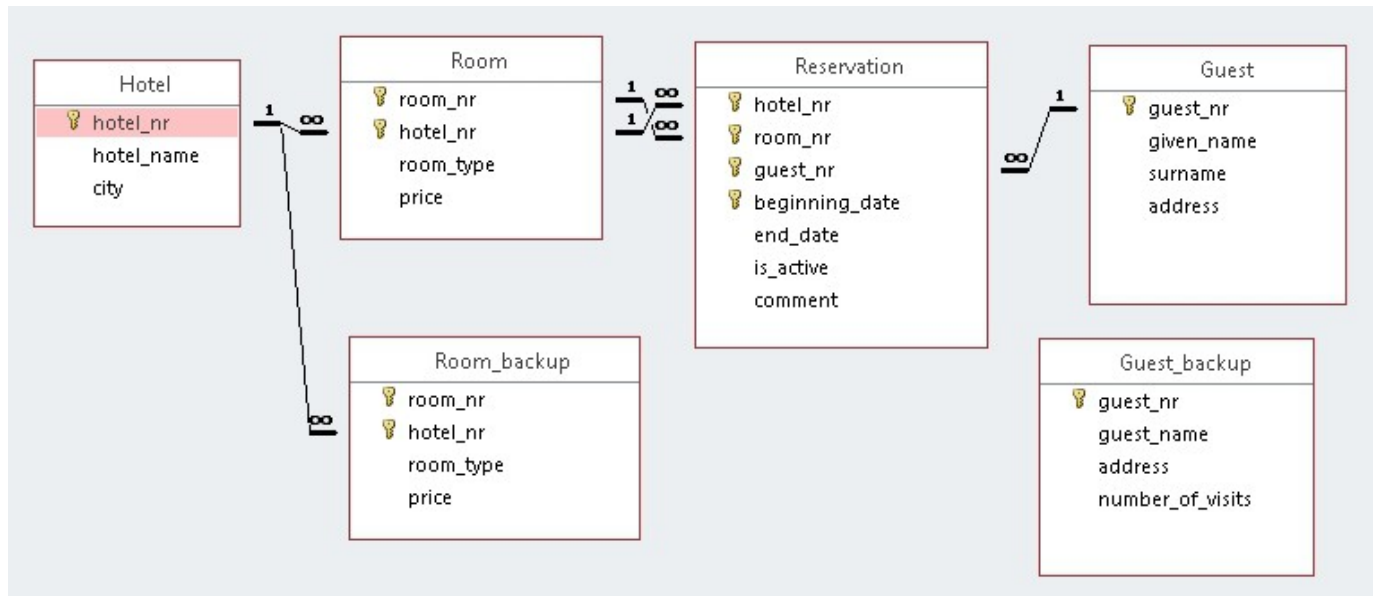


SQL Exercise 4

Database that is created by using MS Access: *Hotel* (**Hotel.mdb**). The structure of the base tables (tables in short) is the following.



All the presented tasks must be solved by using SQL statements. One can use multiple statements for solving a task if not stated otherwise. Save the statements in the MS Access database – for each statement you have to create a separate query object. Use the naming like *Task00*, *Task01*, *Task02*, etc.. If the solution involves more than one SQL statement, then use the names like *Task01_1*, *Task01_2*, etc.

30. Find hotels where the average reservation length is bigger than the average length of reservations over all the hotels. Consider only hotels that have at least one reservation. For each such hotel, present hotel number, hotel name, and the number of rooms in the hotel (in the column with the name *number_of_rooms*). Sort the result based on the number of rooms in the descending order.
31. Find all the hotels in Tallinn, where guests with the given name "Teet" and surname "Tee" have *never made any* reservations. For each such hotel, present hotel number and hotel name. Sort the result based on the hotel names in the alphabetical order. Please note that if there are hotels with no reservations at all, then these must also belong to the result.
32. Find the pairs of guests who have made reservations in the same hotel. For each such pair, present for both guests their guest number and surname. Do not present in the result the pairs of guests with themselves.
33. Find data about all the guests, whose number of associated reservations is bigger than the average number of associated reservations over all the guests. Consider only guests who have at least one associated reservation. For each such guest,

present guest number, surname, and the number of associated reservations with him/her (in the column *number_of_reservations*).

34. Find for each guest the percentage of hotels (from all the registered hotels) where he/she has made at least one reservation. If the guest has not made any reservations, then the percentage must be 0. For each guest, present guest number, surname, and the percentage (in the column *percentage*) that has been rounded to two decimal places. Avoid division by zero.
35. Create a copy of the table *Reservation* by using a `SELECT ... INTO` statement. The name of the copy table must be *Reservation_backup*. Delete from the table *Reservation_backup* the reservations of all the guests who have no reservations that have started during the last 2000 days.
36. In the table *Reservation_backup* change inactive the reservations of all such guests, whose earliest and latest reservations started more than 1000 days apart. Find information about such guests based on the table *Reservation*.
37. Insert into the table *Reservation_backup* data about all the reservations that are recorded in the table *Reservation* and are not recorded in the table *Reservation_backup*. In the inserted rows add the word "New" at the end of the comment.
38. Find the most popular given names of guests and find all the guests who have these given names. Consider only rows where the given name is registered. Present data from all the columns of the table *Guest*. Find in case of hotels with the biggest number of reservations their rooms that do not have any reservations at all. Present data from all the columns of the table *Room*. Sort the result based on the hotel number in the ascending order.
39. Find the number of rooms that have price more than 100 EUR. Find the number of luxurious suites. Do it with one statement.
40. Find data about all the rooms. Present data from all the columns. In addition, the query should have an additional column (with the name *has_reservations*) with the Boolean type. Value TRUE in the column shows that the room has at least one associated reservation and FALSE shows that it does not have any associated reservations.
41. Create a crosstable query that reports how many reservations have begun at different years during different months. The query result must contain one row for each year when at least one reservation has started (in the column *y*). The query result must contain a column for each of the twelve months of a year. The names of the columns must be *January, February, ..., December*. The value in a field of this columns shows how many reservations have started at the corresponding year and month. If no reservations have started at that time, then the value must be 0. Order the rows based on the year number. The solution is explained here: <https://modern-sql.com/use-case/pivot> Hint: In MS Access you cannot use *Extract* function and CASE statement. However, you can use the functions:
 1. *Year*: <https://www.techonthenet.com/access/functions/date/year.php> (extracts year identifier from a date).

2. *Month*: <https://www.techonthenet.com/access/functions/date/month.php> (extracts month identifier from a date).
3. *Iif*: <https://www.techonthenet.com/access/functions/advanced/iif.php> (can be used for the same purpose as the standard SQL's CASE statement).
42. If there is at least one reservation registered (in any hotel), then return all the data from the table *Room*, otherwise do not return any data from the table *Room*.
43. Find numbers and names of all the hotels. There is an additional requirement that if the hotel is in Tallinn, then find it only if there is at least one room in it.