1. **Write a SQL query to find the names and salaries of the employees that take the minimal salary in the company. Use a nested SELECT statement.**

Select e.FirstName, e.LastName, e.Salary from Employees e

where e.Salary = (Select min(Salary) from Employees)

1. **Write a SQL query to find the names and salaries of the employees that have a salary that is up to 10% higher than the minimal salary for the company.**

Select e.FirstName, e.LastName, e.Salary from Employees e

where e.Salary > (Select min(Salary)\*1.1 from Employees)

1. **Write a SQL query to find the full name, salary and department of the employees that take the minimal salary in their department. Use a nested SELECT statement.**

Select e.FirstName + ' ' + e.LastName as [FullName], e.Salary, d.Name

from Employees e inner join Departments d on e.DepartmentID = d.DepartmentID

where e.Salary = (Select min(Salary) from Employees)

1. **Write a SQL query to find the average salary in the department #1.**

Select AVG(Salary) as [Avarage Salary] from Employees e inner join Departments d on e.DepartmentID = d.DepartmentID

where d.DepartmentID = 1

1. **Write a SQL query to find the average salary in the "Sales" department.**

Select AVG(Salary) as [Avarage Salary] from Employees e inner join Departments d on e.DepartmentID = d.DepartmentID

where d.Name = 'Sales'

1. **Write a SQL query to find the number of employees in the "Sales" department.**

Select Count(\*) as [Count] from Employees e inner join Departments d on e.DepartmentID = d.DepartmentID

where d.Name = 'Sales'

1. **Write a SQL query to find the number of all employees that have manager.**

Select Count(\*) from Employees where ManagerID is not null

1. **Write a SQL query to find the number of all employees that have no manager.**

Select Count(\*) from Employees where ManagerID is null

1. **Write a SQL query to find all departments and the average salary for each of them.**

Select d.Name, AVG(e.Salary) as [Avarage Salary]

from Departments d inner join Employees e on d.DepartmentID=e.DepartmentID

group by d.Name

1. **Write a SQL query to find the count of all employees in each department and for each town.**

Select d.Name, count(\*) as [Count]

from (Employees e inner join Departments d on e.DepartmentID = d.DepartmentID)

group by d.Name

UNION

Select t.Name, count(\*) as [Count]

from (Employees e inner join Addresses a on e.AddressID = a.AddressID)

inner join Towns t on t.TownID = a.TownID

group by t.Name

1. **Write a SQL query to find all managers that have exactly 5 employees. Display their first name and last name.**

Select m.FirstName, m.LastName, count(\*) as [Number of Employees] from Employees e inner join Employees m on m.EmployeeID = e.ManagerID

group by m.firstName, m.LastName

having count(\*) = 5

1. **Write a SQL query to find all employees along with their managers. For employees that do not have manager display the value "(no manager)".**

Select e.FirstName + ' ' + e.LastName as [Employee], ISNULL(m.FirstName + ' ' + m.LastName, 'No manager') as [Manager]

from Employees e left join Employees m on m.EmployeeID = e.ManagerID

1. **Write a SQL query to find the names of all employees whose last name is exactly 5 characters long. Use the built-in LEN(str) function.**

Select e.FirstName + ' ' + e.LastName from Employees e

where LEN(e.LastName) = 5

1. **Write a SQL query to display the current date and time in the following format "day.month.year hour:minutes:seconds:milliseconds". Search in Google to find how to format dates in SQL Server.**

Select GetDate()

1. **Write a SQL statement to create a table Users. Users should have username, password, full name and last login time. Choose appropriate data types for the table fields. Define a primary key column with a primary key constraint. Define the primary key column as identity to facilitate inserting records. Define unique constraint to avoid repeating usernames. Define a check constraint to ensure the password is at least 5 characters long.**

CREATE TABLE Users

(

UserId int IDENTITY,

Username nvarchar(50),

Pass nvarchar(20),

Fullname nvarchar(100),

LastLoginTime datetime,

CONSTRAINT PK\_Users PRIMARY KEY(UserId),

CONSTRAINT AK\_Username UNIQUE(Username),

CONSTRAINT CK\_Pass CHECK (len(Pass)>=5)

);

1. **Write a SQL statement to create a view that displays the users from the Users table that have been in the system today. Test if the view works correctly.**

CREATE VIEW [Today Login] AS

SELECT \* FROM Users where LastLoginTime = GETDATE()

1. **Write a SQL statement to create a table Groups. Groups should have unique name (use unique constraint). Define primary key and identity column.**

CREATE TABLE Groups

(

GroupId int IDENTITY,

Name nvarchar(50),

CONSTRAINT PK\_Groups PRIMARY KEY(GroupId),

CONSTRAINT AK\_Name UNIQUE(Name)

);

1. **Write a SQL statement to add a column GroupID to the table Users. Fill some data in this new column and as well in the Groups table. Write a SQL statement to add a foreign key constraint between tables Users and Groups tables.**

ALTER TABLE Users

Add GroupId int

ALTER TABLE Users

ADD CONSTRAINT FK\_Users\_Groups

FOREIGN KEY (GroupID)

REFERENCES Groups(GroupID)

1. **Write SQL statements to insert several records in the Users and Groups tables.**

Insert into Groups values ('First group')

Insert into Groups values ('Second group')

Insert into Groups values ('Third group')

Insert into Users values ('Me', '12345', 'Me Me', 2014-08-21, 1)

Insert into Users values ('You', '12345', 'You You', 2014-08-21, 1)

1. **Write SQL statements to update some of the records in the Users and Groups tables.**

Update Users set Fullname = 'Real me' where Username = 'Me'

1. **Write SQL statements to delete some of the records from the Users and Groups tables.**

Delete from Groups where GroupId = 2

1. **Write SQL statements to insert in the Users table the names of all employees from the Employees table. Combine the first and last names as a full name. For username use the first letter of the first name + the last name (in lowercase). Use the same for the password, and NULL for last login time.**

Insert into Users(Username, Pass, Fullname, LastLoginTime) Select LEFT(e.FirstName, 1)+e.LastName, LEFT(e.FirstName, 1)+e.LastName, e.FirstName + ' ' + e.LastName, null from Employees e

Must be that, but not working as it should be, because there is repetition in first letter from first name and last name – AHill, and we have CONSTRAINT that username must be unique…

1. **Write a SQL statement that changes the password to NULL for all users that have not been in the system since 10.03.2010.**

Update Users

Set pass = Null where LastLoginTime < 2010-10-03

1. **Write a SQL statement that deletes all users without passwords (NULL password).**

Delete from Users

where pass is null

1. **Write a SQL query to display the average employee salary by department and job title.**

Select avg(Salary) as [Avarage Salary], e.JobTitle, d.Name

from Employees e inner join Departments d on e.DepartmentId = d.DepartmentId

group by e.JobTitle, d.Name

1. **Write a SQL query to display the minimal employee salary by department and job title along with the name of some of the employees that take it.**

Select min(Salary) as [Avarage Salary], e.JobTitle, d.Name, e.FirstName, e.LastName

from Employees e inner join Departments d on e.DepartmentId = d.DepartmentId

group by e.JobTitle, d.Name, e.FirstName, e.LastName

1. **Write a SQL query to display the town where maximal number of employees work.**

Select top 1 t.Name as [Town], count(\*) as [Number of Employees]from Employees e

inner join Addresses a on e.AddressId = a.AddressId

inner join Towns t on a.TownId = t.TownId

group by t.Name

order by count(\*) desc

1. **Write a SQL query to display the number of managers from each town.**

Select t.Name as [Town], count(\*) as [Number of Managers] from Employees e

inner join Addresses a on e.AddressId = a.AddressId

inner join Towns t on a.TownId = t.TownId

where e.ManagerId is null

group by t.Name

1. **Start a database transaction, delete all employees from the 'Sales' department along with all dependent records from the other tables. At the end rollback the transaction.**

BEGIN TRAN

ALTER TABLE Departments DROP CONSTRAINT FK\_Departments\_Employees

DELETE FROM Employees where DepartmentId = (Select DepartmentId from Departments where Name = 'Sales')

ROLLBACK TRAN

1. **Start a database transaction and drop the table EmployeesProjects. Now how you could restore back the lost table data?**

BEGIN TRAN

Drop table EmployeesProjects

ROLLBACK TRAN

If you like really to drop it use **COMMIT ☺**

1. **Find how to use temporary tables in SQL Server. Using temporary tables backup all records from EmployeesProjects and restore them back after dropping and re-creating the table.**

That’s how: <https://www.simple-talk.com/sql/t-sql-programming/temporary-tables-in-sql-server/>