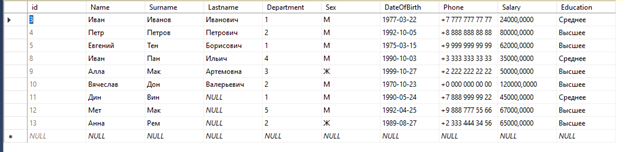
**Лабораторная работа 1. Расширенные возможности SELECT**

**Яновский Евгений Б1-ИФСТ21**

1. **Продемонстрируйте работу простого запроса на группировку с отбором записей в исходном наборе до группировки и отбором получившихся групп в результирующем наборе.**

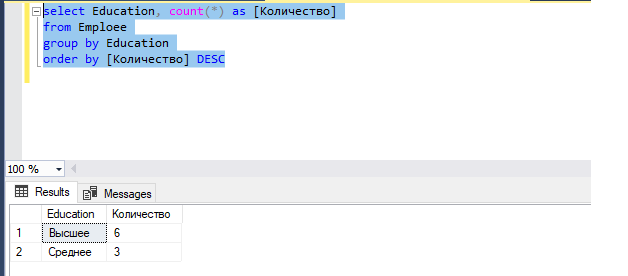


select Education, count(\*) as [Количество]

from Emploee

group by Education

order by [Количество] DESC



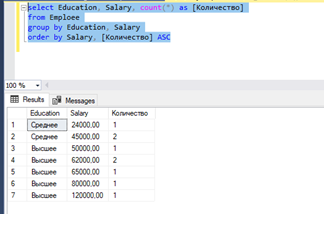
1. **Повторите п.1, используя при этом группировку по двум-трём столбцам**.

select Education, Salary, count(\*) as [Количество]

from Emploee

group by Education, Salary

order by Salary, [Количество] ASC



1. **Повторите п.1, используя группировку по вычисляемому выражению.**

select

substring (Lastname, 1, 1) as [А-Я],

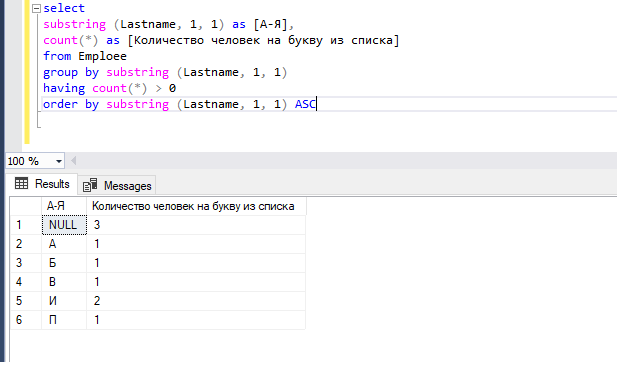
count(\*) as [Количество человек на букву из списка]

from Emploee

group by substring (Lastname, 1, 1)

having count(\*) > 0

order by substring (Lastname, 1, 1) ASC



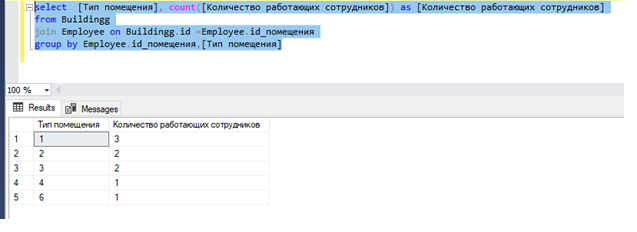
1. **Продемонстрируйте работу простого запроса с использованием группировки по результату соединения (join) имеющихся таблиц. Покажите проблему группировки кортежей подчинённой таблицы по неуникальному полю одной из связанных таблиц. Продемонстрируйте более правильный вариант группировки.**

select [Тип помещения], count([Количество работающих сотрудников]) as [Количество работающих сотрудников]

from Buildingg

join Employee on Buildingg.id =Employee.id\_помещения

group by Employee.id\_помещения,[Тип помещения]



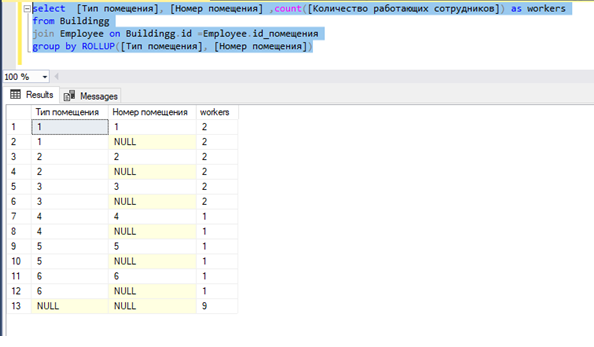
1. **Модифицируйте запрос из п.4 так, чтобы в нём появились подитоги по иерархии значений в столбцах группировки. Используйте для этого оператор ROLLUP. При этом покажите использование различного количества столбцов в операторе ROLLUP.**

select [Тип помещения], [Номер помещения] ,count([Количество работающих сотрудников]) as workers

from Buildingg

join Employee on Buildingg.id =Employee.id\_помещения

group by ROLLUP([Тип помещения], [Номер помещения])

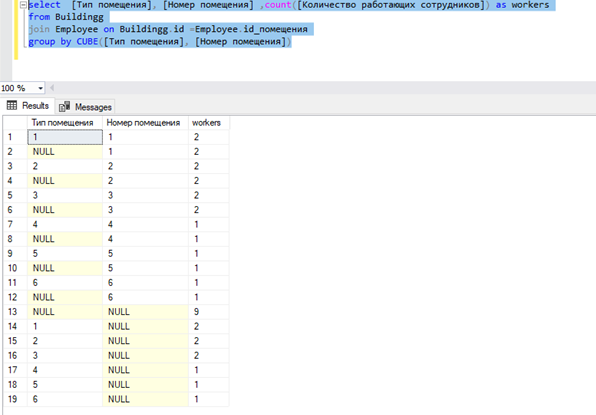


1. **Модифицируйте запрос из п.4 так, чтобы в нём появились подитоги по комбинациям значений в столбцах группировки. Используйте для этого оператор CUBE. При этом покажите использование различного количества столбцов в операторе CUBE.**

select [Тип помещения], [Номер помещения] ,count([Количество работающих сотрудников]) as workers

from Buildingg

join Employee on Buildingg.id =Employee.id\_помещения

group by CUBE([Тип помещения], [Номер помещения]) 

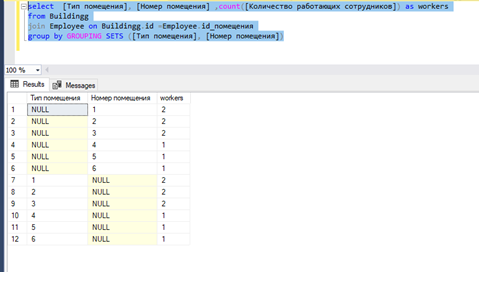
1. **Модифицируйте запрос из п.4. установив при помощи оператора GROUPING SETS произвольный набор конфигураций уровней блокирования.**

select [Тип помещения], [Номер помещения] ,count([Количество работающих сотрудников]) as workers

from Buildingg

join Employee on Buildingg.id =Employee.id\_помещения

group by GROUPING SETS ([Тип помещения], [Номер помещения])



1. **Продемонстрируйте примеры эквивалентных соотношений между операторами ROLLUP, CUBE и оператора GROUPING SETS.**

select D.id ,

count(S.[Количество работающих сотрудников]) AS [Количество работающих сотрудников],

CASE

D.Фамилия

WHEN D.Фамилия THEN D.Фамилия

ELSE 'ФИО'

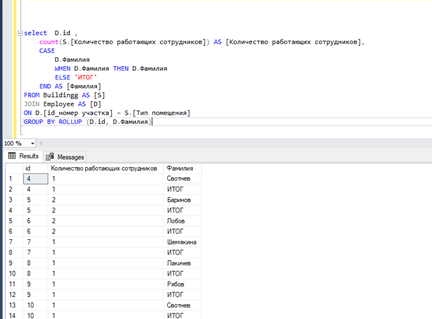
END AS [Фамилия]

FROM Buildingg AS [S]

JOIN Employee AS [D]

ON D.[id\_номер участка] = S.[Тип помещения]

GROUP BY ROLLUP (D.id, D.Фамилия)



1. **При помощи оператора GROUPING отделите в итоговом наборе запроса из п.4 значения NULL, показывающие исключение соответствующего атрибута из группирования от значений NULL, показывающие отсутствующие значения. Для этого замените первые - на строку «ВСЕ», а вторые – на строку «НЕИЗВЕСТНО».**

SELECT

ISNULL (Position, case when grouping(Position) = 1

then 'FINAL'

else 'UNKOWN'

end) Position,

ISNULL (CAST([id\_номер участка] as varchar(20)), case when grouping([id\_номер участка]) = 1

then 'FINAL'

else 'UNKOWN'

end) [id\_номер участка],

ISNULL (CAST([Номер помещения] as varchar(20)), case when grouping([Номер помещения]) = 1

then 'FINAL'

else 'UNKOWN'

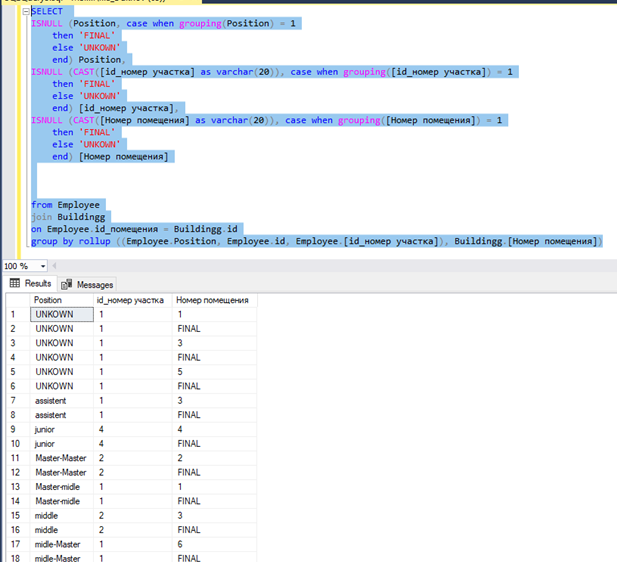
end) [Номер помещения]

from Employee

join Buildingg

on Employee.id\_помещения = Buildingg.id

group by rollup ((Employee.Position, Employee.id, Employee.[id\_номер участка]), Buildingg.[Номер помещения])



1. **Продемонстрируйте работу ранжирующих функций RANK, DENSE\_RANK, ROW\_NUMBER и NTILE. Наглядно покажите разницу между ними.**

Функция row\_number нумерует строки, возвращаемые запросом.

SELECT Employee.id, Employee.Фамилия, Employee.[Дата принятия на работу],

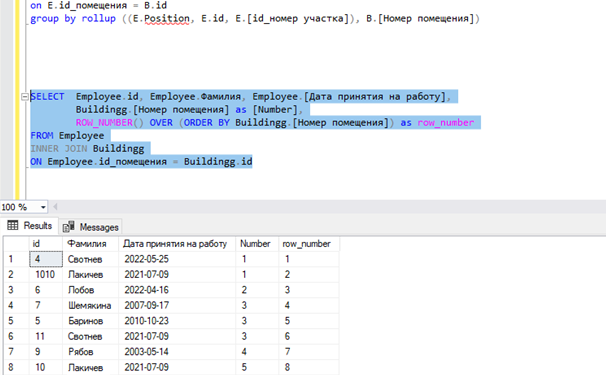
Buildingg.[Номер помещения] as [Number],

ROW\_NUMBER() OVER (ORDER BY Buildingg.[Номер помещения]) as row\_number

FROM Employee

INNER JOIN Buildingg

ON Employee.id\_помещения = Buildingg.id



SELECT Employee.Фамилия, Employee.Телефон,

Employee.[Дата принятия на работу],

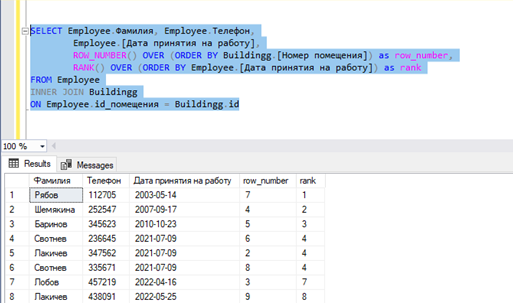
ROW\_NUMBER() OVER (ORDER BY Buildingg.[Номер помещения]) as row\_number,

RANK() OVER (ORDER BY Employee.[Дата принятия на работу]) as rank

FROM Employee

INNER JOIN Buildingg

ON Employee.id\_помещения = Buildingg.id



SELECT Employee.Фамилия, Employee.Телефон,

Employee.[Дата принятия на работу],

ROW\_NUMBER() OVER (ORDER BY Buildingg.[Номер помещения]) as row\_number,

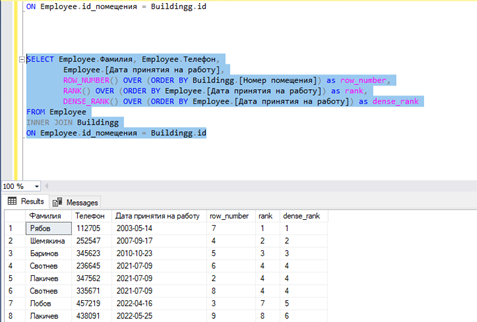
RANK() OVER (ORDER BY Employee.[Дата принятия на работу]) as rank,

DENSE\_RANK() OVER (ORDER BY Employee.[Дата принятия на работу]) as dense\_rank

FROM Employee

INNER JOIN Buildingg

ON Employee.id\_помещения = Buildingg.id



SELECT Employee.Фамилия, Employee.Телефон,

Employee.[Дата принятия на работу],

NTILE(4) OVER (ORDER BY Employee.[Дата принятия на работу])as Ntile,

ROW\_NUMBER() OVER (ORDER BY Buildingg.[Номер помещения]) as row\_number,

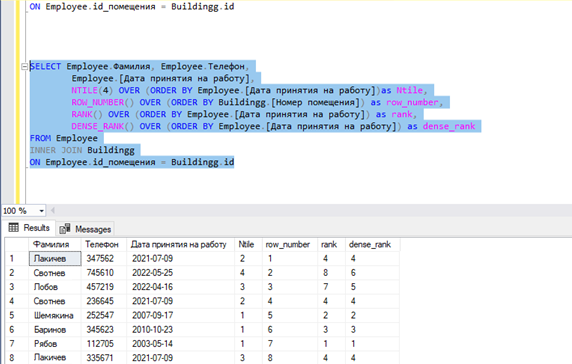
RANK() OVER (ORDER BY Employee.[Дата принятия на работу]) as rank,

DENSE\_RANK() OVER (ORDER BY Employee.[Дата принятия на работу]) as dense\_rank

FROM Employee

INNER JOIN Buildingg

ON Employee.id\_помещения = Buildingg.id



1. **Повторите пункт 10, но с применением оконных функций. В качестве критерия выделения окон можно выбрать отдел или должность для таблицы «Сотрудники» или категорию для таблицы «Товары».**

SELECT Фамилия, Position, Salary,

Employee.[Дата принятия на работу],

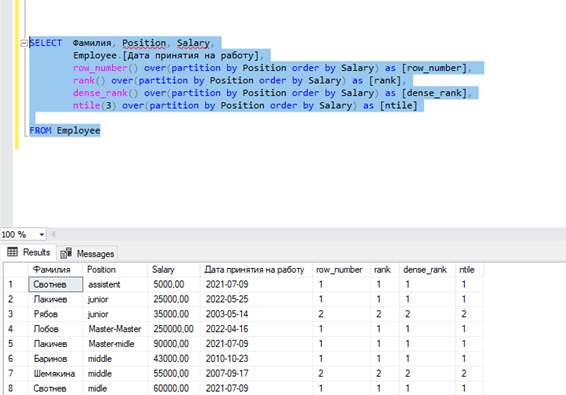
row\_number() over(partition by Position order by Salary) as [row\_number],

rank() over(partition by Position order by Salary) as [rank],

dense\_rank() over(partition by Position order by Salary) as [dense\_rank],

ntile(3) over(partition by Position order by Salary) as [ntile]

FROM Employee



1. **Добавьте в одну из таблиц Вашей схемы атрибут, который будет являться внешним ключом, указывающим на записи этой же таблицы. Например, в таблицу «Сотрудники» добавьте информацию о руководителе для каждого сотрудника или для таблицы «Товары» - информацию о сопутствующем товаре, который прилагается к данному товару в подарок по рекламной акции. Составьте рекурсивное табличное выражение, в котором наглядно выведите записи вашей таблицы в порядке их иерархии. Предусмотрите визуальное отображение иерархии, например, при помощи отступов различной величины.**

with

tmp ([level], [ID Сотрудника], [ФИО Сотрудника], Категория, Должность, [ID Начальника]) as

(select 0 as [level], s1.[ID Сотрудника], s1.[ФИО Сотрудника], s1.Категория, s1.Должность, s1.[ID Начальника]

from Рабочие as [s1]

where [ID Начальника] is null

union all

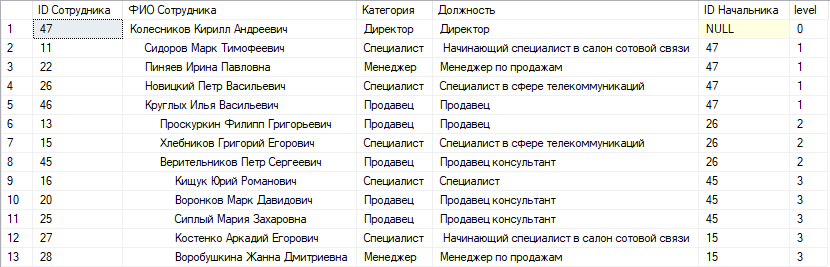
select [level] + 1, s2.[ID Сотрудника], s2.[ФИО Сотрудника], s2.Категория, s2.Должность, s2.[ID Начальника]

from Рабочие as [s2]

INNER JOIN tmp on s2.[ID Начальника] = tmp.[ID Сотрудника])

select [ID Сотрудника], replicate(' ', [level] \* 5) + [ФИО Сотрудника] as [ФИО Сотрудника], Категория, Должность, [ID Начальника], [level]

from tmp



1. **Продемонстрируете владение оператором слияния наборов MERGE. Для этого организуйте слияние с существующей в Вашей схеме таблицы набора данных, полученного при помощи запроса к любой другой таблице. Например, в качестве источника строк можно использовать базу данных Adventure Work, входящую в комплект поставки MS SQL Server. В качестве критерия слияния выберите соответствие значений в столбцах, которые могут служить естественным идентифицирующим признаком для сущностей (например, серия и номер паспорта для сотрудника или артикул для товара).**

MERGE HB AS [HB]

USING Employee AS [E]

ON ([HB].id = [E] .id)

WHEN MATCHED THEN

UPDATE SET

[HB].Имя = [E].Имя, [HB].Фамилия = [E].Фамилия, [HB].Отчество= [E].Отчество,

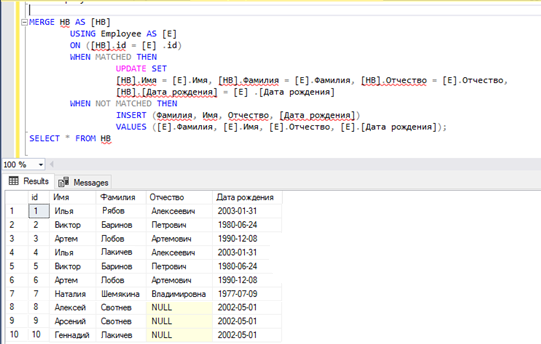
[HB].[Дата рождения] = [E] .[Дата рождения]

WHEN NOT MATCHED THEN

INSERT (Фамилия, Имя, Отчество, [Дата рождения])

VALUES ([E].Фамилия, [E].Имя, [E].Отчество, [E].[Дата рождения]);

SELECT \* FROM HB



Все записи были добавлены в новую таблицу.

**Вопросы к отчёту:**

1. Простая группировка данных. Операторы group by, having.
2. Ограничения оператора group by.
3. Группировка по вычисляемым выражениям.
4. Составная группировка по двум и более столбцам. В чём её смысл?
5. Промежуточные подитоги. ROLLUP, CUBE и GROUPING SETS. Принципы работы, отличия, взаимоотношения.
6. Ранжирующие функции. Их возможности и сферы применения. Указание критериев ранжирования.
7. Оконные функции. Применение совместно с агрегатными или ранжирующими функциями.
8. Обобщённые табличные выражения. Сферы применения обобщённых табличных выражений.
9. Организация рекурсивных запросов при помощи обобщённых табличных выражений.
10. Оператор слияния наборов MERGE. Возможности и сферы его применения.