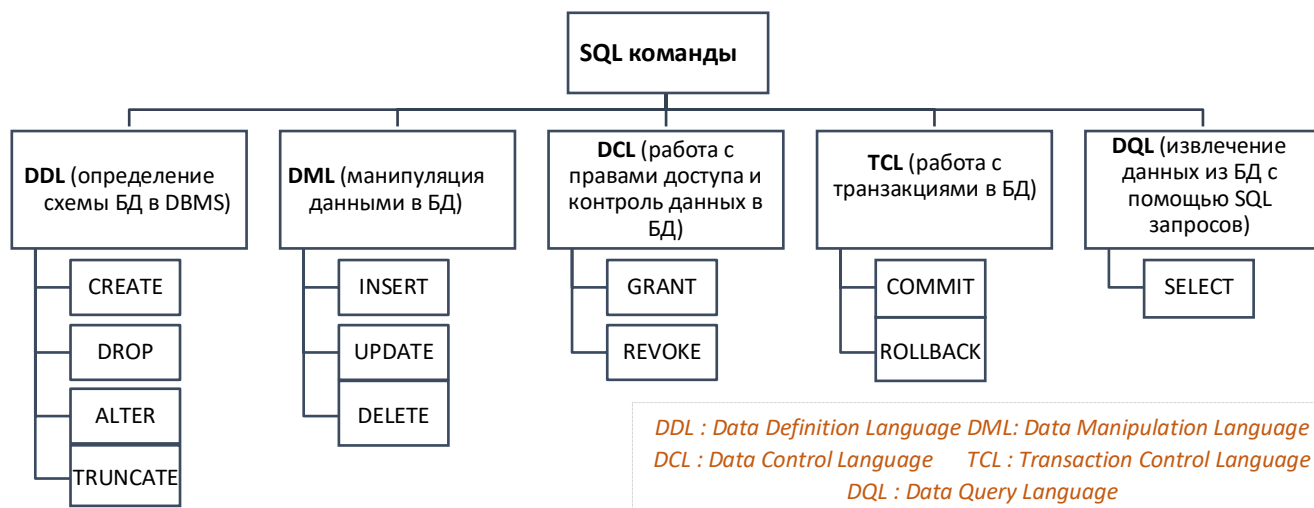


Structured Query language (SQL)



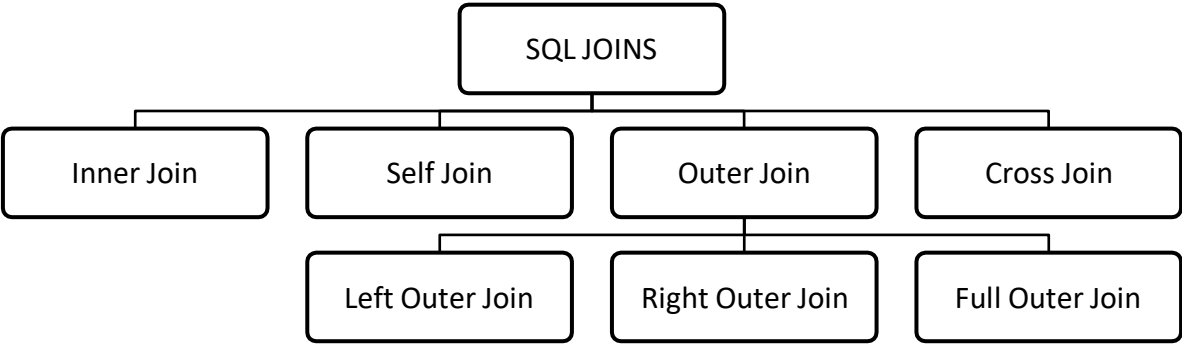
1. Создание БД (Create database)	<code>create database shop</code>
2. Использование БД (Use the database)	<code>use shop</code>
3. Создание таблицы (Create table)	<code>create table goods(item_id int identity(1,1) primary key, item_number int not null unique check (item_number>0), name varchar(30) not null, shop_id int default 71000, address varchar(50), country varchar(50) default 'Montenegro')</code>
4. Добавить данные в таблицу (Insert values into table)	<code>insert into goods values (101, 'T-Short Red', '418999', 'Budva 256', default), (102, 'T-Short Blue', default, 'Antalya', 'Turkey'), (103, 'Socks', default, 'Budva 256', default)</code>
5. Показать записи из таблицы (Display record from table)	<code>-- показать все записи select * from goods -- показать определенные столбцы select item_id, item_number, name from goods</code>
6. Добавить новый столбец в таблицу (Add new column to table)	<code>alter table goods add item_code varchar(20)</code>
7. Добавить значения в новый столбец / Обновление таблицы (Add values to newly added column/ Update table)	<code>update goods set item_code='1234545346' where item_id=1 update goods set item_code='45554654' where item_id=2</code>
8. Удаление столбца (Delete a column)	<code>alter table goods drop column item_code</code>
9. Удаление записи из таблицы (Delete record from table) --Если не указать оператор 'where' будут удалены все	<code>delete from goods where country='Montenegro'</code>

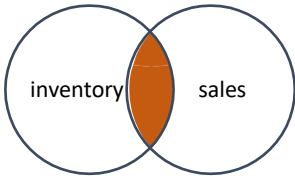
записи	
10. Удаление таблицы (Delete table)	<code>drop table goods</code>
11. Изменения типа данных (Change data type)	<code>alter table goods</code> <code>alter column item_code varchar(10)</code>

1. Создание БД (Create database)	<code>create database SaleOrder</code>
2. Использование БД (Use the database)	<code>use SaleOrder</code>
3. Создание нескольких таблиц (Create tables)	<pre> create table customer (CustomerID int NOT null primary key, CustomerFirstName varchar(50) NOT null, CustomerLastName varchar(50) NOT null, CustomerAddress varchar(50) NOT null, CustomerSuburb varchar(50) null, CustomerCity varchar(50) NOT null, CustomerPostCode char(4) null, CustomerPhoneNumber char(12) null,); create table inventory (InventoryID tinyint NOT null primary key, InventoryName varchar(50) NOT null, InventoryDescription varchar(255) null,); create table employee (EmployeeID tinyint NOT null primary key, EmployeeFirstName varchar(50) NOT null, EmployeeLastName varchar(50) NOT null, EmployeeExtension char(4) null,); create table sale (SaleID tinyint not null primary key, CustomerID int not null references customer(CustomerID), InventoryID tinyint not null references Inventory(InventoryID), EmployeeID tinyint not null references Employee(EmployeeID), SaleDate date not null, SaleQuantity int not null, SaleUnitPrice smallmoney not null); </pre>
4. Проверить, что внутри таблицы (Check what table inside)	<code>select * from information_schema.Название таблицы</code>
5. Посмотреть определенные записи таблицы (View specific row)	<code>--сверху: показать только первые две</code> <code>select top 2 * from goods</code>

6. Посмотреть определенные столбцы (View specific column)	--сортировать результаты (по умолчанию по возрастанию) select customerfirstname, customerlastname from customer order by customerlastname desc --показать только уникальные значения select distinct customerlastname from customer order by customerlastname
7. Сохранить таблицу в другую таблицу (Save table to another table)	select distinct customerlastname into temp from customer order by customerlastname select * from temp --посмотреть данные временной таблицы (типы данных останутся как были)
8. Оператор Like (выборка данных)	-- (нижнее подчеркивание) _ выборка любого одного символа -- (процент) % представляет ноль, один или несколько символов select * from customer where customerlastname like ' _r%'
9. Оператор In (выборка данных)	-- поиск соответствия по нескольким значениям одновременно select * from customer where customerlastname in ('Alex', 'Ivan', 'Jon')
10. Операторы =, >, <, >=, <= (выборка данных)	select * from customer where customerlastname = 'Alex' or customerlastname='Meshkov'
11. Оператор <> (неравенство)	select * from customer where customerlastname <> 'Ivan'
12. IS NULL	-- находит незаполненные значения select * from customer where customerlastname IS NULL
13. IS NOT NULL	select * from customer where customerlastname IS NOT NULL
14. Оператор Between	select * from sale where saleunitprice between 5 and 10 --не включает в поиск значения 5 и 10
15. Оператор Count AS	-- возвращает количество строк в таблице -- AS означает временное присвоение названия столбцу в результатах выборки select count(*) as [Number of Records] from customer where customerfirstname like 'A%'
16. Оператор Sum	select sale.employeeid ,EmployeeFirstName, EmployeeLastName , count(*) as[Number of order] , sum(salequantity) as [Total Quantity]from sale,employee where sale.employeeid = employee.employeeid group by sale.employeeid ,EmployeeFirstName, EmployeeLastName
17. Оператор Count month	select month(saledate) as [Month], count (*) as [Number of sale],sum(salequantity*saleunitprice) as [Total Amount] from sale group by month(saledate)
18. Оператор Max	SELECT MAX(Salary) FROM EmployeeSalary
19. Оператор Min	SELECT MIN(Salary) FROM EmployeeSalary

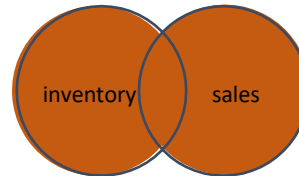
20. Оператор Average	<pre>SELECT AVG(Salary) FROM EmployeeSalary</pre>
21. Оператор having -- HAVING аналогичен оператору WHERE за тем исключением, что применяется не для всего набора столбцов таблицы, а для набора созданного оператором GROUP BY и применяется всегда строго после него.	<pre>SELECT JobTitle, COUNT(JobTitle)FROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID GROUP BY JobTitle HAVING COUNT(JobTitle) > 1</pre>
22. Изменить временно тип данных для использования (Change data type temporary for use)	<pre>-- CAST(expression AS datatype(length)) SELECT CAST('2017-08-25 00:00:00.000' AS date) -- CONVERT(data_type(length), expression, style) SELECT CONVERT(date,'2017-08-25 00:00:00.000')</pre>
23. Вложенные запросы (Subquery)	<pre>-- Вложенный запрос в Select SELECT EmployeeID, Salary, (SELECT AVG(Salary) FROM EmployeeSalary) AS AllAvgSalary FROM EmployeeSalary -- Вложенный запрос в From SELECT a.EmployeeID, AllAvgSalary FROM (SELECT EmployeeID, Salary, AVG(Salary) OVER () ASAllAvgSalary FROM EmployeeSalary) a ORDER BY a.EmployeeID -- Вложенный запрос в Where SELECT EmployeeID, JobTitle, SalaryFROM EmployeeSalary WHERE EmployeeID in (SELECT EmployeeID FROM EmployeeDemographics WHERE Age > 30) SELECT EmployeeID, JobTitle, SalaryFROM EmployeeSalary WHERE Salary in (SELECT Max(Salary) FROM EmployeeSalary)</pre>



1. Получение данных из нескольких таблиц (без использования JOIN)	<code>select * from inventory,sale where sale.inventoryid=inventory.inventoryid</code>
	<code>select inventoryname,saledate,saleunitprice,salequantity,salequantity*saleunitprice as [Total amount] from sale,inventory where sale.inventoryid=inventory.inventoryid group by sale.inventoryid,inventoryname,saledate,salequantity,saleunitprice order by inventoryname</code>
2. Получение данных из нескольких таблиц (с использованием Inner, Outer JOIN)	<div><code>--inner join select * from inventory inner join sale on sale.inventoryid=inventory.inventoryid</code> <code>select inventoryname,saledate,saleunitprice,salequantity,saleunitprice*salequantity as [Total Amount] from inventory inner join sale on sale.inventoryid=inventory.inventoryid order by inventoryname</code></div> <div></div>

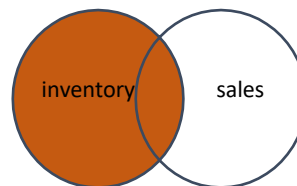
--full outer join (показывает все)

```
select sale.inventoryid,inventoryname  
from inventory  
full outer join sale on  
sale.inventoryid=inventory.inventoryid  
where sale.inventoryid is NULL
```



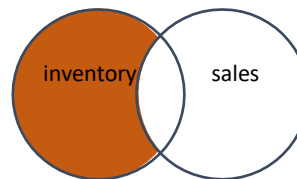
--left join (может показывать пустые значения таблицы так как не все записи таблицы inventory могут иметь данные в таблице sales)

```
select inventory.inventoryid,inventoryname  
from inventory left join sale on  
sale.inventoryid=inventory.inventoryid
```



--left join

```
select inventory.inventoryid,inventoryname  
from inventory left join sale on  
sale.inventoryid=inventory.inventoryid  
where sale.inventoryid is NULL
```

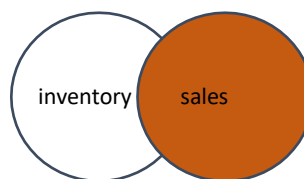


-- без использования join: вложенный запрос

```
select inventoryid,inventoryname from inventory where inventoryid not in (select  
inventoryid from sale)
```

--right join

```
select sale.inventoryid,inventoryname from inventory right join sale on
sale.inventoryid=inventory.inventoryid
```

**3. Self Join**

--используется для объединения таблицы с ней самой таким образом, будто это две разные таблицы, временно переименовывая одну из них.

--inner join

Staff Table

employeeID	employeefirstname	employeeelastname	managerID
1001	Alex	Meshkov	NULL
1002	Fedor	Ivanov	1001
1003	Igor	Petrov	1002

```
Select E.employeeID, E.employeefirstname+' '+E.employeeelastname as [Full
Name], E.managerID, , M.employeefirstname+' '+M.employeeelastname as
[Manager Name]
from staff E
inner join staff
M
on E.managerID = M.employeeID
```

Output:

employeeID	Full Name	managerID	managerName
1002	Fedor Ivanov	1001	Alex Meshkov
1003	Igor Petrov	1002	Fedor Ivanov

--left outer join (список всех сотрудников)

```
select E.employeeID, E.employeefirstname+' '+E.employeeelastname as [F
Name], E.managerID, , M.employeefirstname+' '+M.employeeelastname as
[Manager Name]
from staff E
left outer join staff M
on E.managerID = M.employeeID
```

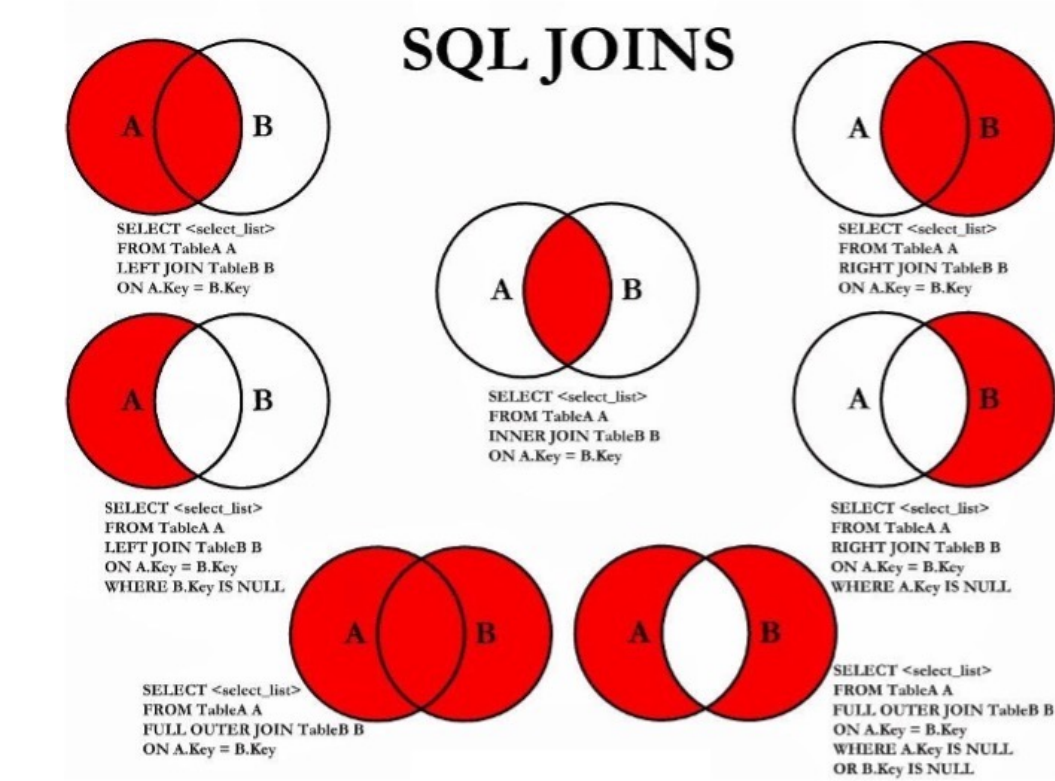
Output:

employeeID	Full Name	managerID	managerName
1001	Alex Meshkov		
1002	Fedor Ivanov	1001	Alex Meshkov
1003	Igor Petrov	1002	Fedor Ivanov

4. Cross Join

--создает все комбинации записей (все варианты)

```
select * from inventory1 cross join inventory2
```



SQL UNIONS

<div>1. Union</div> <div>--позволяет объединить две таблицы вместе (но количество столбцов и типы данных каждого столбца для 2 таблиц должны совпадать)</div> <div>--не требуется общий ключ, нужны только общие атрибуты</div> <div>--при объединении не показывается дубликаты</div>	<div><pre>select cust_lastname,cust_firstname from customer union select cust_lastname,cust_firstname from customer_2</pre></div>
<div>2. Union all</div> <div>--при объединении показывает все записи, в том числе дубликаты</div>	<div><pre>select cust_lastname,cust_firstname from customer union all select cust_lastname,cust_firstname from customer_2</pre></div>

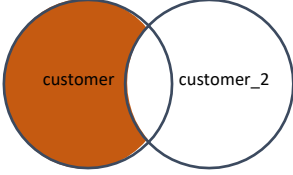
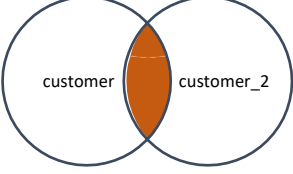
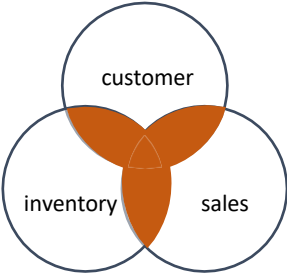
<p>3. Intersect</p> <p>--показывает строки общие для обоих запросов</p> <p>--не показывает дублирующие записи</p>	<pre>select cust_lastname,cust_firstname from customerintersect select cust_lastname,cust_firstname from customer_2</pre>  <pre>select c.cust_lastname,c.cust_firstname from customer c,customer_2 c2 where c.cust_lastname=c2.cust_lastname and c.cust_firstname=c2.cust_firstname</pre>
<p>4. Except</p> <p>--показывает только записи, которые уникальны для таблицы CUSTOMER</p>	<pre>select cust_lastname,cust_firstname from customer except select cust_lastname,cust_firstname from customer_2</pre>  <pre>--использование вложенного запроса select cust_lastname,cust_firstname from customer where(cust_lastname) not in (select cust_lastname from customer_2) and (cust_firstname) not in (select cust_firstname from customer_2)</pre>

Table & View

<p>1. Создание представлений (Create view) (Представление будет обнуляться при обновлении БД)</p> <p>--представление – это виртуальная таблица, содержимое которой определяется запросом.</p>	<pre>create view CustomerView as select customerfirstname+' '+customerlastname as [Customer Name] , customerphonenumber, inventoryname,saledate,salequantity,saleunitprice,salequantity*saleunitprice as [Total Amount] from customer inner join sale on customer.customerid=sale.customerid inner join inventory on sale.inventoryid=inventory.inventoryid</pre> 
<p>2. Временная таблица (временная таблица не обновляется при обновлении)</p> <p>-- перед именем таблицы при создании необходимо добавить хэштег (#) -- используется для временного хранения данных, физически созданных в БД Tempdb -- может выполнять CRUD, объединение и некоторые другие операции, аналогично обычным таблицам БД</p>	<pre>DROP TABLE IF EXISTS #temp_Employee Create table #temp_Employee (JobTitle varchar(100), EmployeesPerJob int, AvgAge int, AvgSalary int) Insert INTO #temp_Employee SELECT JobTitle, Count(JobTitle), Avg(Age), AVG(salary)FROM EmployeeDemographics emp JOIN EmployeeSalary sal ON emp.EmployeeID = sal.EmployeeID group by JobTitle SELECT * FROM #temp_Employee</pre>
<p>3. Обобщенные табличные выражения CTE (Common Table Expression)</p> <p>-- создает временный набор результатов, который используется для написания сложных запросов -- результаты табличных выражений создаются в памяти, а не в БД Tempdb, но к ним можно обращаться повторно.</p>	<pre>WITH CTE_Employee AS(SELECT FirstName, LastName, Gender, Salary, COUNT(Gender) OVER (PARTITION BY Gender) AS TotalGenderFROM EmployeeDemographics ED JOIN EmployeeSalary ES ON ED.EmployeeID = ES.EmployeeID WHERE Salary > '45000') SELECT FirstName, LastName, Gender, TotalGenderFROM CTE_Employee WHERE TotalGender = (SELECT MIN(TotalGender) FROM CTE_Employee)</pre>

Author - Alex Meshkov

Telegram - <https://t.me/QAtestgrow>

Youtube channel -

<https://www.youtube.com/@AlexMeshkovQA>

4. Дубликат таблицы(Duplicate Table)	<pre>select customerfirstname+' '+customerlastname as [Customer Name] , customerphonenumner, inventoryname,saledate,salequantity,saleunitprice,salequantity*saleunitprice as [Total Amount] into customerRec from customer inner join sale on customer.customerid=sale.customerid inner join inventory on sale.inventoryid=inventory.inventoryid order by customerfirstname +' '+ customerlastname,inventoryname</pre>