SQL

Eng. Amira Fouda

01

Introduction

What is Database?

What is Database?

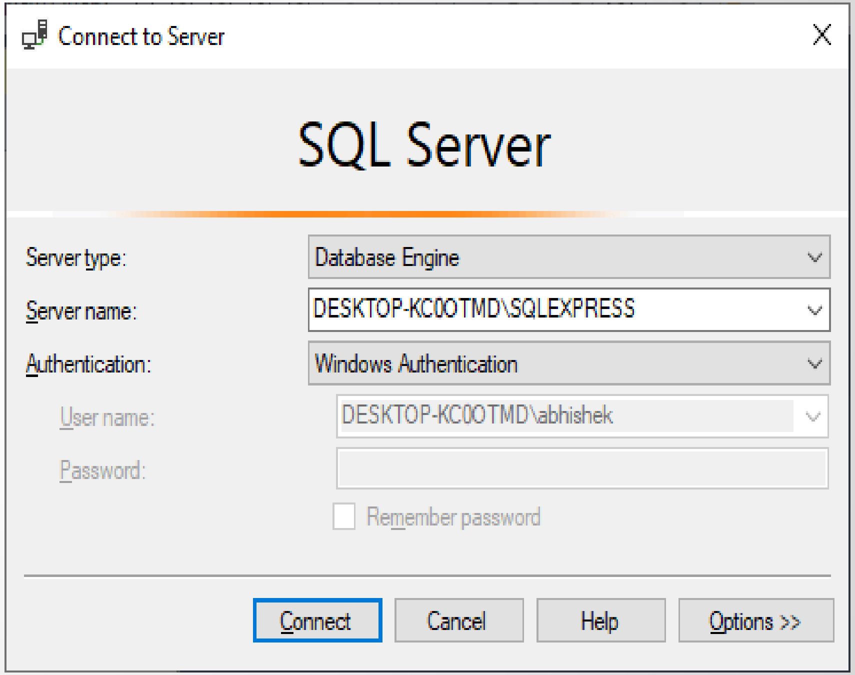
A collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views, and reports, etc.

System Database

System database	Description
master Database	Records all the system-level information for an instance of SQL Server.
msdb Database	Is used by SQL Server Agent for scheduling alerts and jobs.
model Database	Is used as the template for all databases created on the instance of SQL Server. Modifications made to the model database. Such as database size, collection, recovery model, and other database options, are applied to any databases created afterward.
tempdb Database	Is a workspace for holding temporary objects or intermediate result sets.

Create First Database

Step 1: Open the SSMS in administrator mode to avoid any permission issue. We will see the below screen where we will first connect with the server. Here, we must fill in the server name, server type, authentication information and then click on **Connect** button to continue.

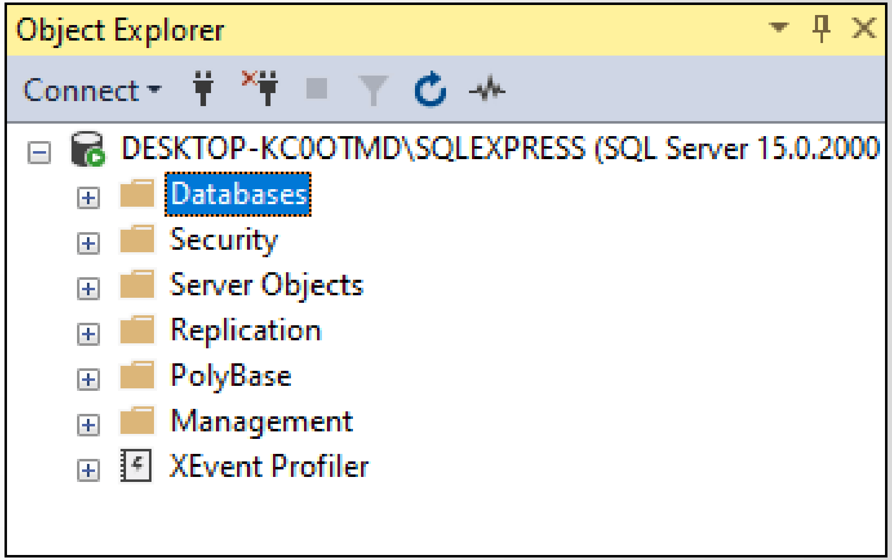

Connect to Server

SQL Server

Server type: Database Engine
Server name: DESKTOP-KC0OTMD\SQLEXPRESS
Authentication: Windows Authentication
User name: DESKTOP-KC0OTMD\abhishek
Password:
☐ Remember password

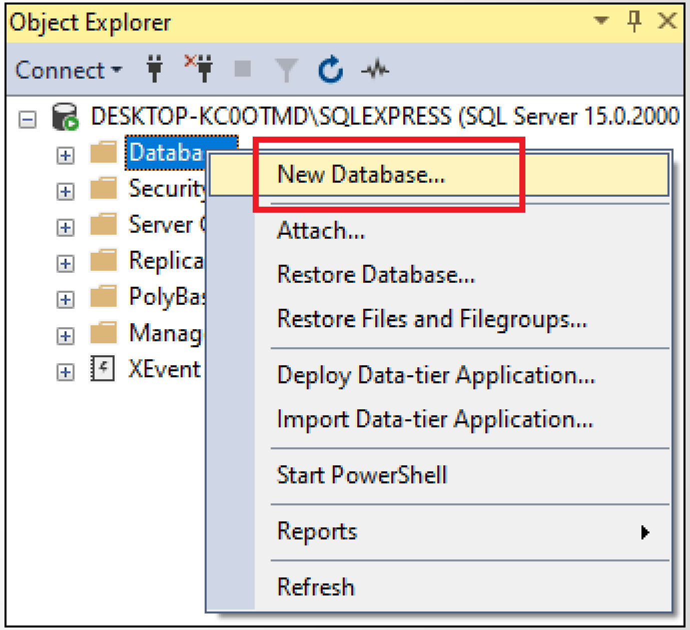
Create First Database

Step 2: Once the connection becomes successful, the **Object Explorer** window will appear on the left-hand side of the screen. The server we are connected to is shown at the top of the Object Explorer. To see the Database folder, click the "+" button if it isn't extended.



Create First Database

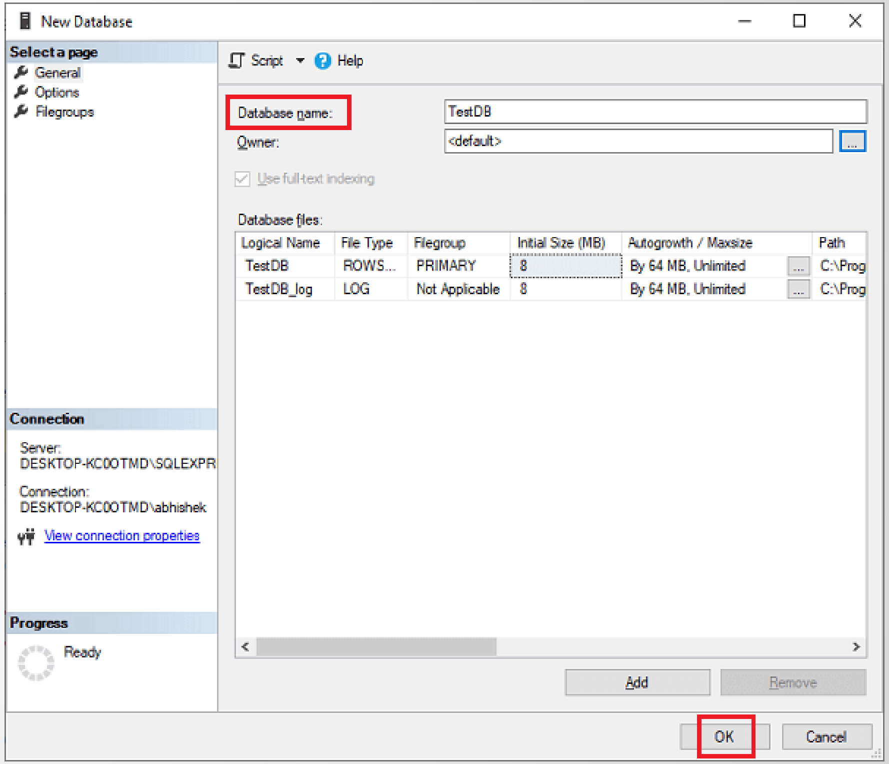
Step 3: The next step is to right-click on the **Databases** folder and choose a **New Database** from the dropdown list to create a database.



Object Explorer
Connect
DESKTOP-KC0OTMD\SQLEXPRESS (SQL Server 15.0.2000
Databa
Security
Server
Replica
PolyBas
Manag
XEvent
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

Create First Database

Step 4: The next step will open the New Database dialog box. Here we can **configure** the database before creating it. Now, type the database name, change the setting if required, and then click the Ok button. In most cases, the DBA leaves the settings at their default.



New Database

Select a page
- General
- Options
- Filegroups

Database name: TestDB

Owner: <default>

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize		Path
TestDB	ROWS...	PRIMARY	8	By 64 MB, Unlimited	...	C:\Prog
TestDB_log	LOG	Not Applicable	8	By 64 MB, Unlimited	...	C:\Prog

Connection

Server:
DESKTOP-KC0OTMD\SQLEXPR

Connection:
DESKTOP-KC0OTMD\abhishek

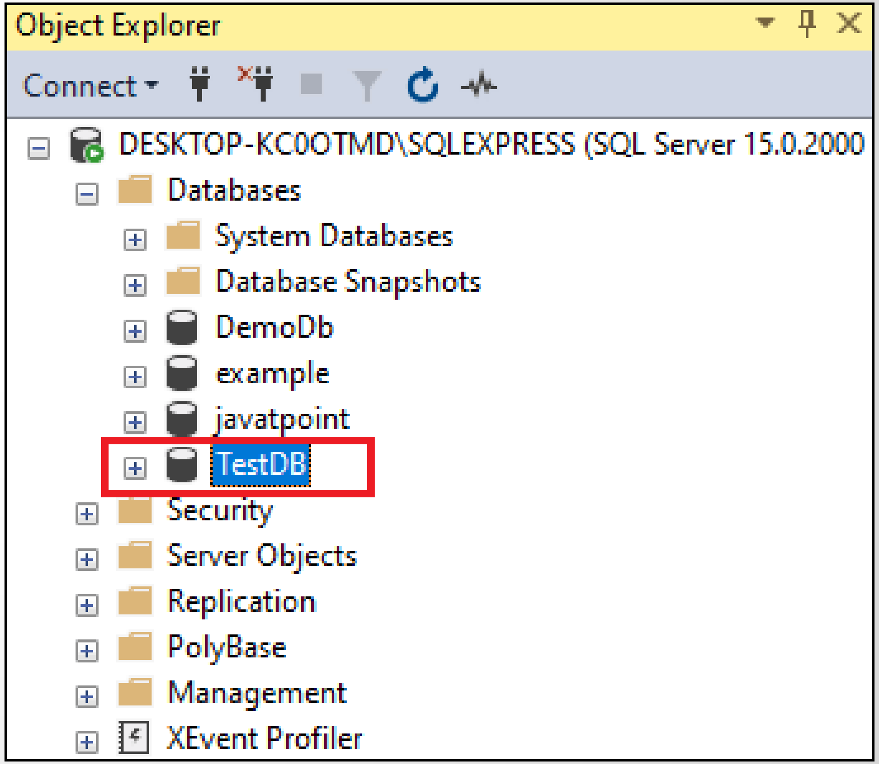
View connection properties

Progress

Ready

Create First Database

Step 5: Once the database creation is successful, we can see them by expanding the Databases folder under the Object Explorer. The database icon has a **cylinder icon**.

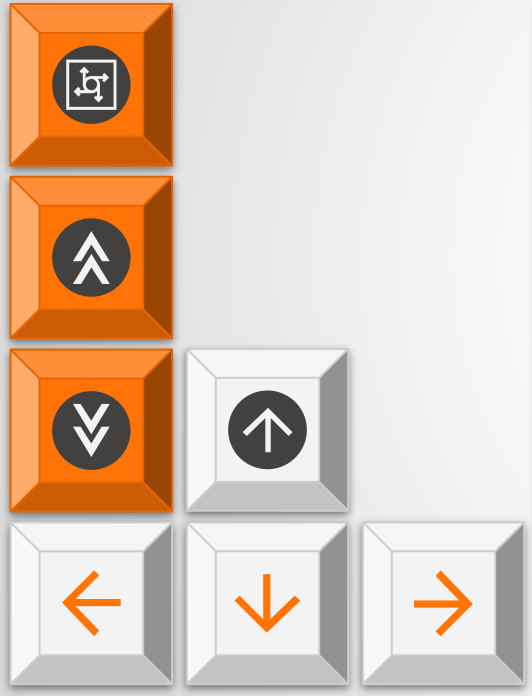


What is SQL?

SQL (Structured Query Language)
Is used to perform operations on the records stored in the database, such as updating records, inserting records, deleting records, creating and modifying database tables, views, etc.

SQL is not a database system, but it is a query language.

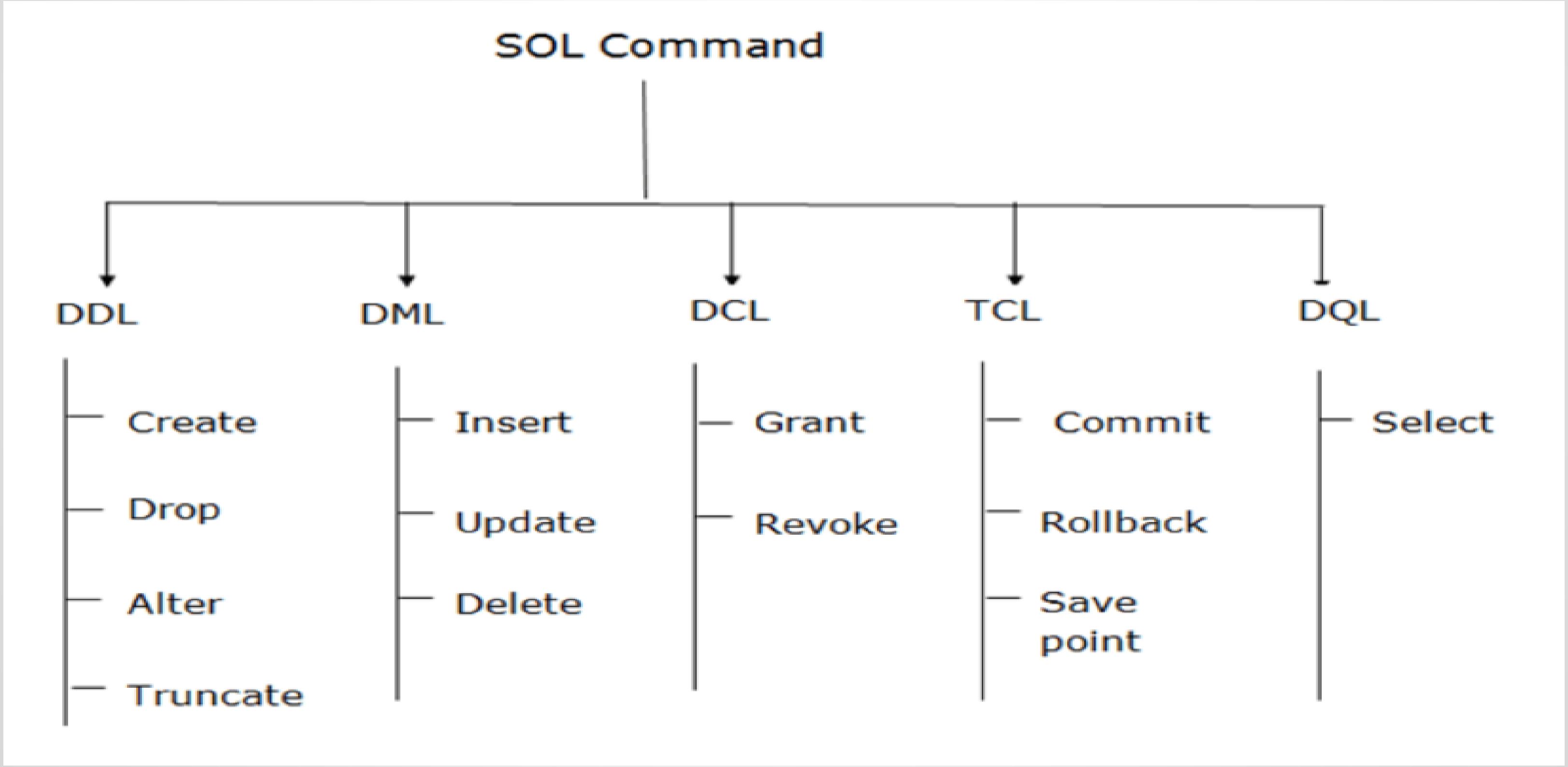
SQL Commands



- SQL commands are instructions. It is used to communicate with the database. It is also used to perform specific tasks, functions, and queries of data.

- SQL can perform various tasks like create a table, add data to tables, drop the table, modify the table, set permission for users.

Types of SQL Commands



Data Definition Language (DDL)

- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.

- All the command of DDL are auto-committed that means it permanently save all the changes in the database.

Data Definition Language (DDL)

Here are some commands that come under DDL:

- CREATE
- ALTER
- DROP
- TRUNCATE

Data Definition Language (DDL)

a. **CREATE** It is used to create a new table in the database.

```
CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES[,....]);
```

Example:

```
CREATE TABLE EMPLOYEE(Name VARCHAR2(20), Email VARCHAR2(100), DOB DATE);
```

Data Definition Language (DDL)

b. DROP: It is used to delete both the structure and record stored in the table.

```
DROP TABLE table_name;
```

Example:

```
DROP TABLE EMPLOYEE;
```

Data Definition Language (DDL)

c. ALTER: It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

```
ALTER TABLE table_name ADD column_name COLUMN-definition;
```

Example:

```
ALTER TABLE STU_DETAILS ADD(ADDRESS VARCHAR2(20));
ALTER TABLE STU_DETAILS MODIFY (NAME VARCHAR2(20));
```

Data Definition Language (DDL)

d. TRUNCATE: It is used to delete all the rows from the table and free the space containing the table.

```
TRUNCATE TABLE table_name;
```

Example:

```
TRUNCATE TABLE EMPLOYEE;
```

Data Manipulation Language (DML)

- DML commands are used to modify the database. It is responsible for all form of changes in the database.

- The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

Data Manipulation Language (DML)

Here are some commands that come under DML:

- INSERT
- UPDATE
- DELETE

Data Manipulation Language (DML)

a. INSERT: The INSERT statement is a SQL query. It is used to insert data into the row of a table.

```
INSERT INTO TABLE_NAME
VALUES (value1, value2, value3, .... valueN);
```

Example:

```
INSERT INTO javatpoint (Author, Subject) VALUES ("Sonoo", "DBMS");
```

Data Manipulation Language (DML)

b. UPDATE: This command is used to update or modify the value of a column in the table.

```
UPDATE table_name SET [column_name1= value1,...column_nameN = valueN] [WHERE CONDITION]
```

Example:

```
UPDATE students
SET User_Name = 'Sonoo'
WHERE Student_Id = '3'
```

Data Manipulation Language (DML)

c. **DELETE:** It is used to remove one or more row from a table.

```
DELETE FROM table_name [WHERE condition];
```

Example:

```
DELETE FROM javatpoint
WHERE Author="Sonoo";
```

Data Control Language (DCL)

- DCL commands are used to grant and take back authority from any database user.

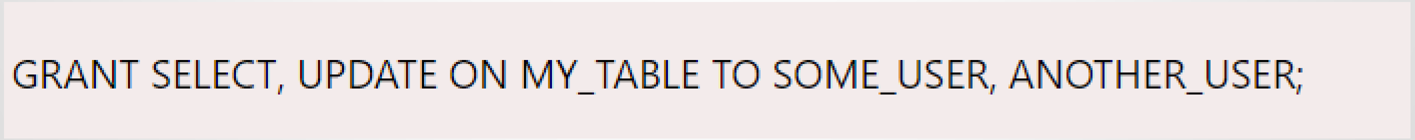
Here are some commands that come under DCL:

- Grant
- Revoke

Data Control Language (DCL)

a. Grant: It is used to give user access privileges to a database.

Example:

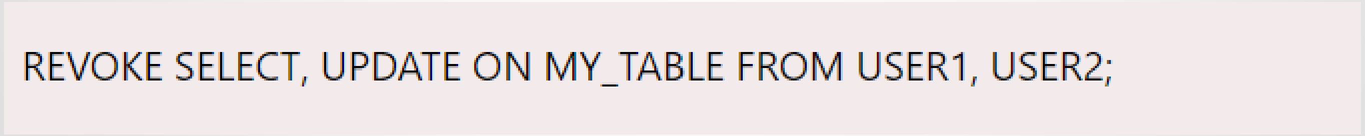


```
GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;
```

Data Control Language (DCL)

b. Revoke: It is used to take back permissions from the user.

Example:



```
REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;
```

Transaction Control Language(TCL)

- TCL commands can only use with DML commands like INSERT, DELETE and UPDATE only.
- These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

Transaction Control Language(TCL)

Here are some commands that come under TCL:

- COMMIT
- ROLLBACK
- SAVEPOINT

Transaction Control Language (TCL)

a. Commit: Commit command is used to save all the transactions to the database.

```
COMMIT;
```

Example:

```
DELETE FROM CUSTOMERS
WHERE AGE = 25;
COMMIT;
```

Transaction Control Language (TCL)

b. Rollback: Rollback command is used to undo transactions that have not already been saved to the database.

```
ROLLBACK;
```

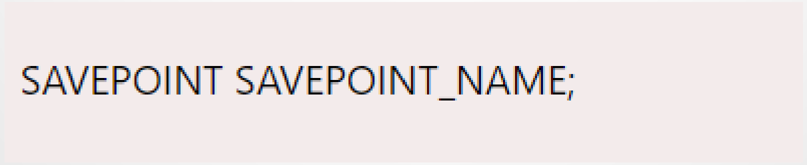
Example:

```
DELETE FROM CUSTOMERS
WHERE AGE = 25;
ROLLBACK;
```

Transaction Control Language (TCL)

c.SAVEPOINT: It is used to roll the transaction back to a certain point without rolling back the entire transaction.

Example:



```
SAVEPOINT SAVEPOINT_NAME;
```

Data Query Language (DQL)

- DQL is used to fetch the data from the database.

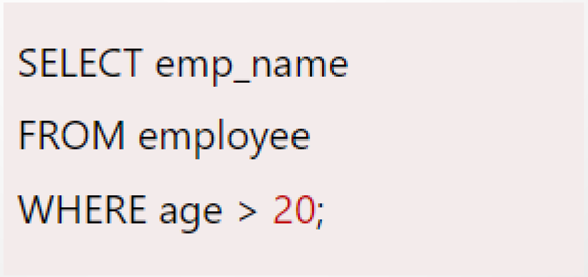
It uses only one command:

- SELECT

Data Query Language (DQL)

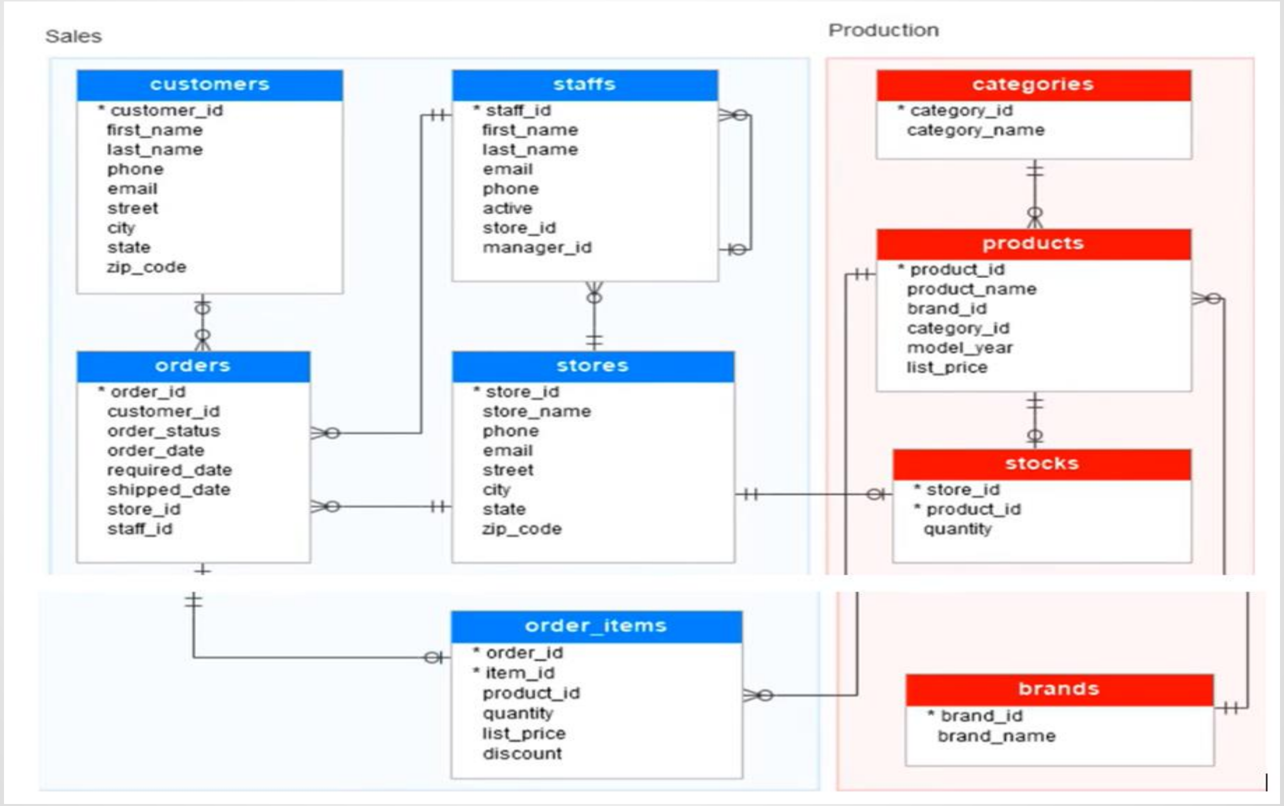
a. SELECT: This is the same as the projection operation of relational algebra. It is used to select the attribute based on the condition described by WHERE clause.

Example:

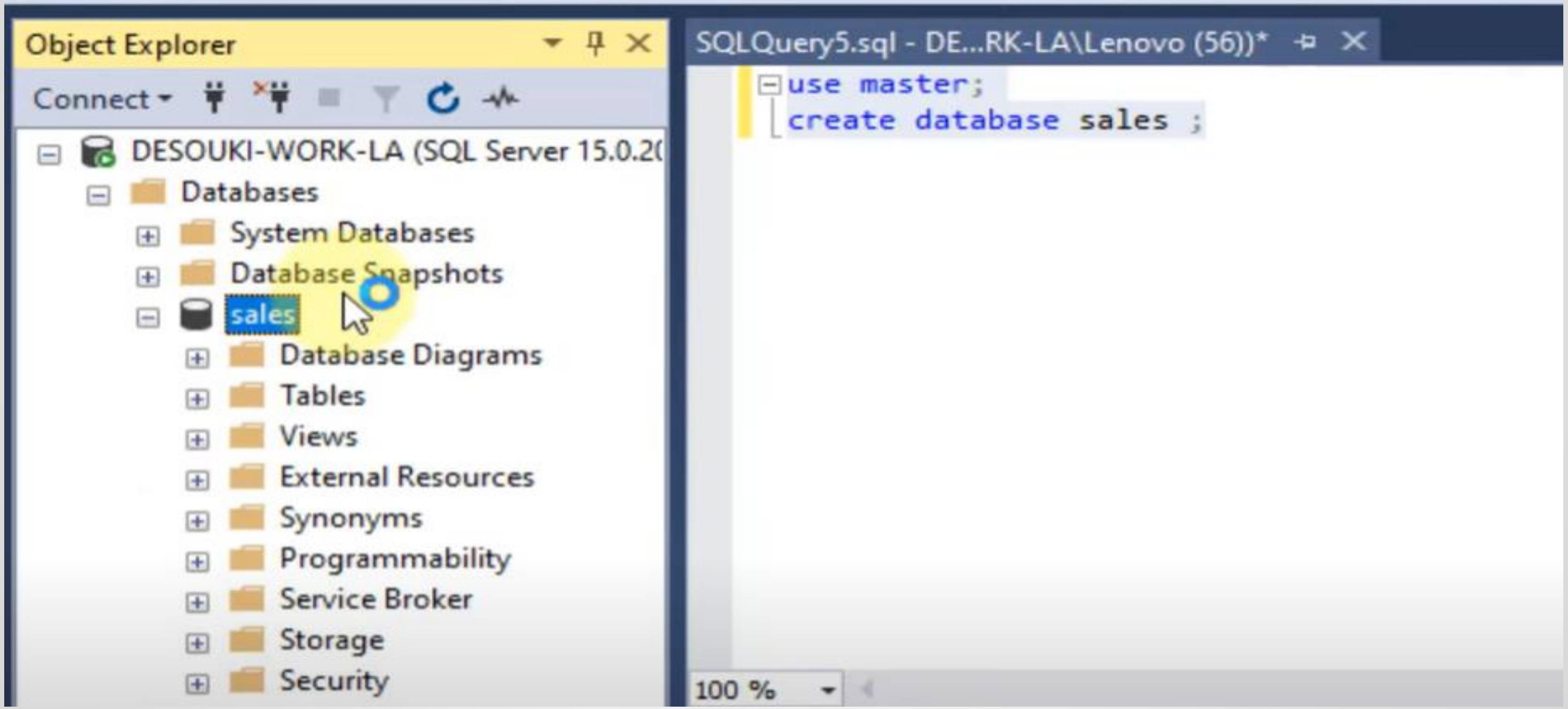


```
SELECT emp_name
FROM employee
WHERE age > 20;
```

Sample Database

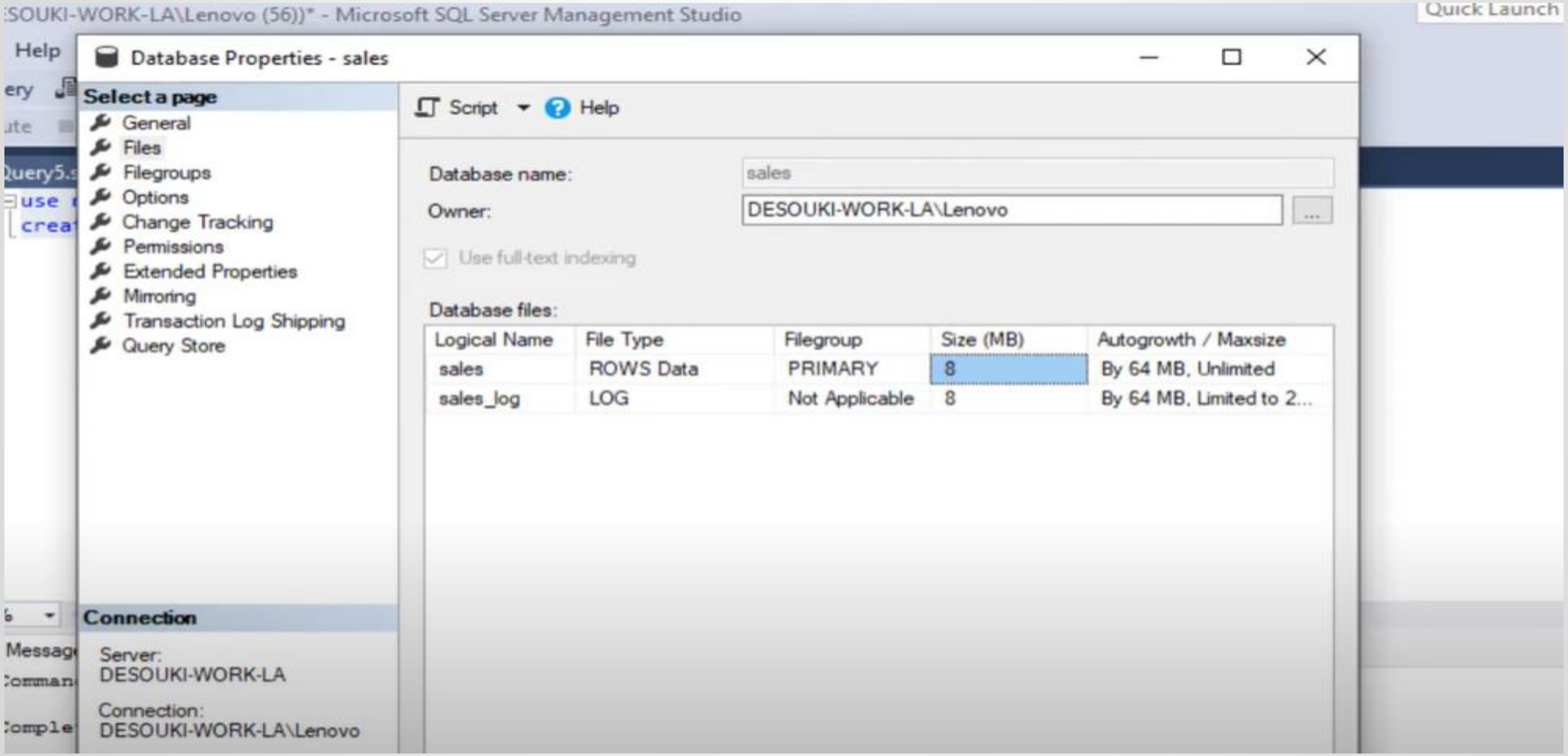


Create Database



```
use master;
create database sales ;
```

Create Database



:SOUKI-WORK-LA\Lenovo (56))* - Microsoft SQL Server Management Studio

Quick Launch

Help

ery

ute

Query5.s

use

crea

Messag

Comman

Comple

Database Properties - sales

Select a page

- General
- Files
- Filegroups
- Options
- Change Tracking
- Permissions
- Extended Properties
- Mirroring
- Transaction Log Shipping
- Query Store

Connection

Server:
DESOUKI-WORK-LA

Connection:
DESOUKI-WORK-LA\Lenovo

Script ▾ Help

Database name: sales

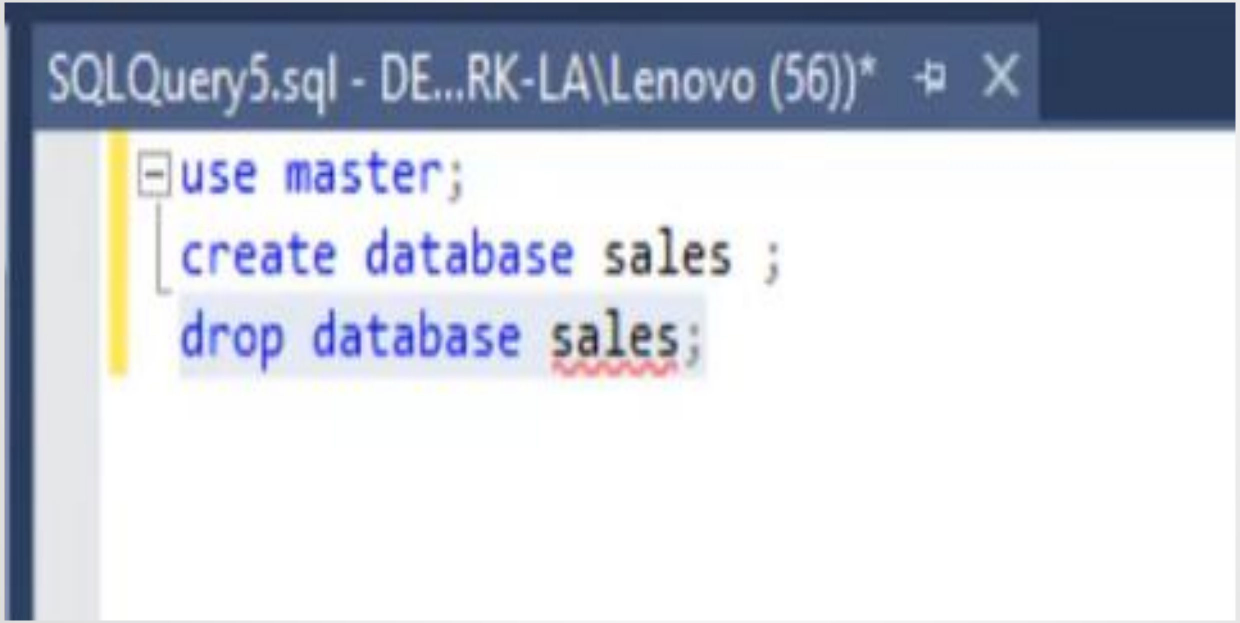
Owner: DESOUKI-WORK-LA\Lenovo

Use full-text indexing

Database files:

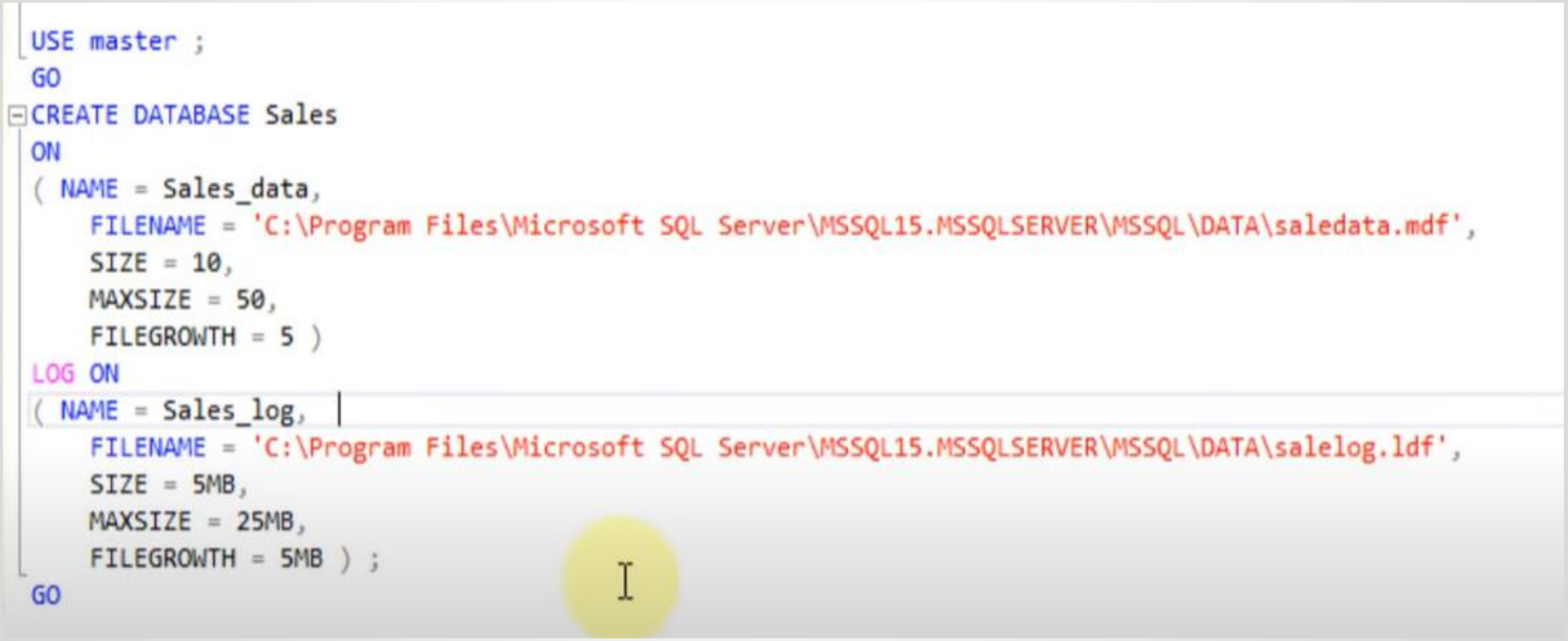
Logical Name	File Type	Filegroup	Size (MB)	Autogrowth / Maxsize
sales	ROWS Data	PRIMARY	8	By 64 MB, Unlimited
sales_log	LOG	Not Applicable	8	By 64 MB, Limited to 2...

Drop Database



```
use master;
create database sales ;
drop database sales;
```

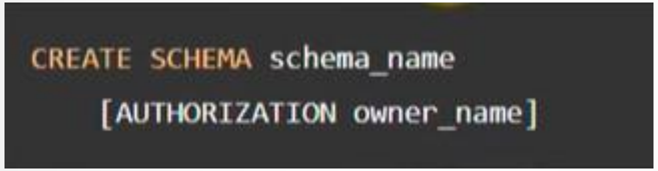
Create Database



```
USE master ;
GO
CREATE DATABASE Sales
ON
( NAME = Sales_data,
    FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\saledata.mdf',
    SIZE = 10,
    MAXSIZE = 50,
    FILEGROWTH = 5 )
LOG ON
( NAME = Sales_log,
    FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\salelog.ldf',
    SIZE = 5MB,
    MAXSIZE = 25MB,
    FILEGROWTH = 5MB ) ;
GO
```

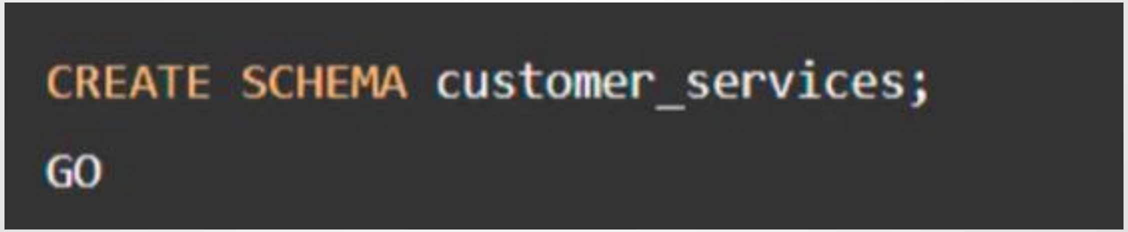
Create Schema

Create Schema Statement overview



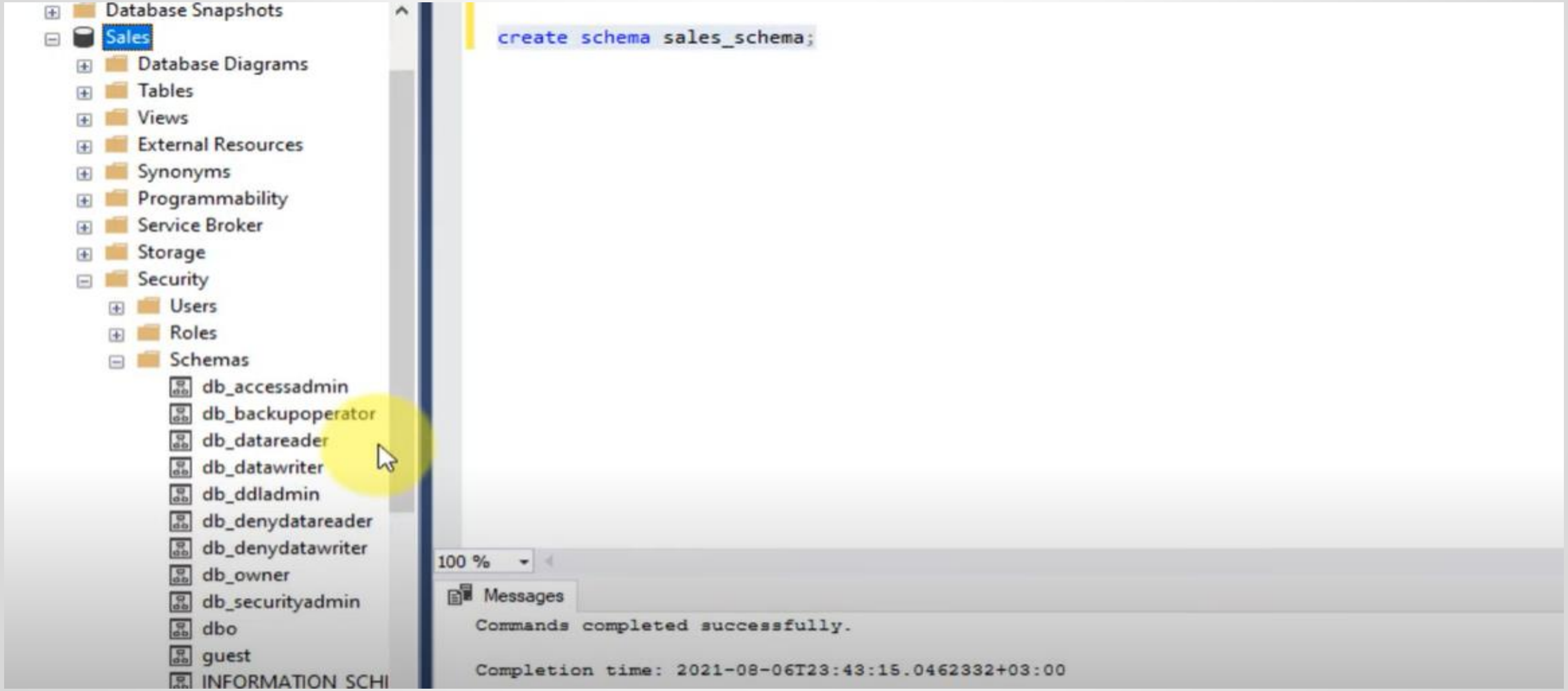
```
CREATE SCHEMA schema_name
    [AUTHORIZATION owner_name]
```

Example:



```
CREATE SCHEMA customer_services;
GO
```

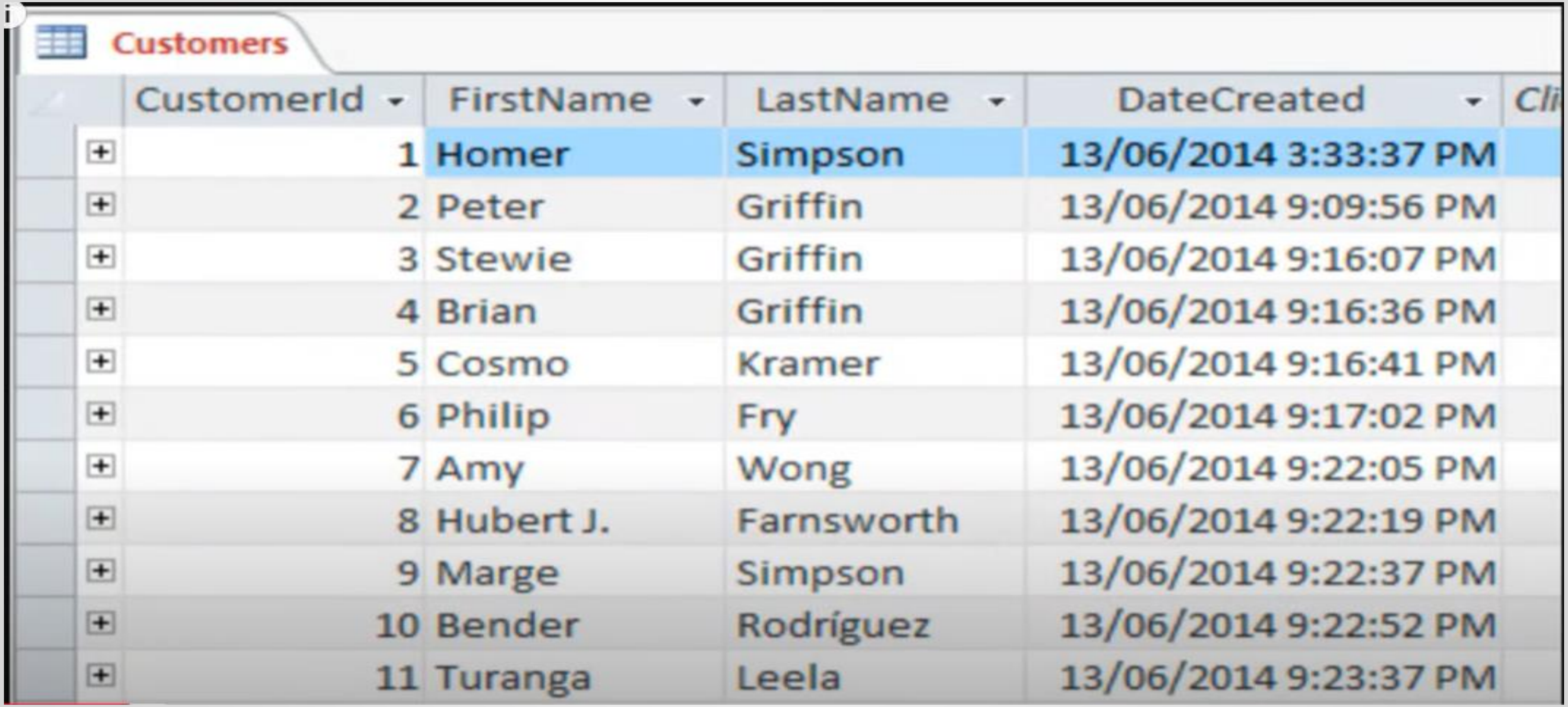
Create Schema



```sql
create schema sales_schema;
```

```
Commands completed successfully.

Completion time: 2021-08-06T23:43:15.0462332+03:00
```

Create Table

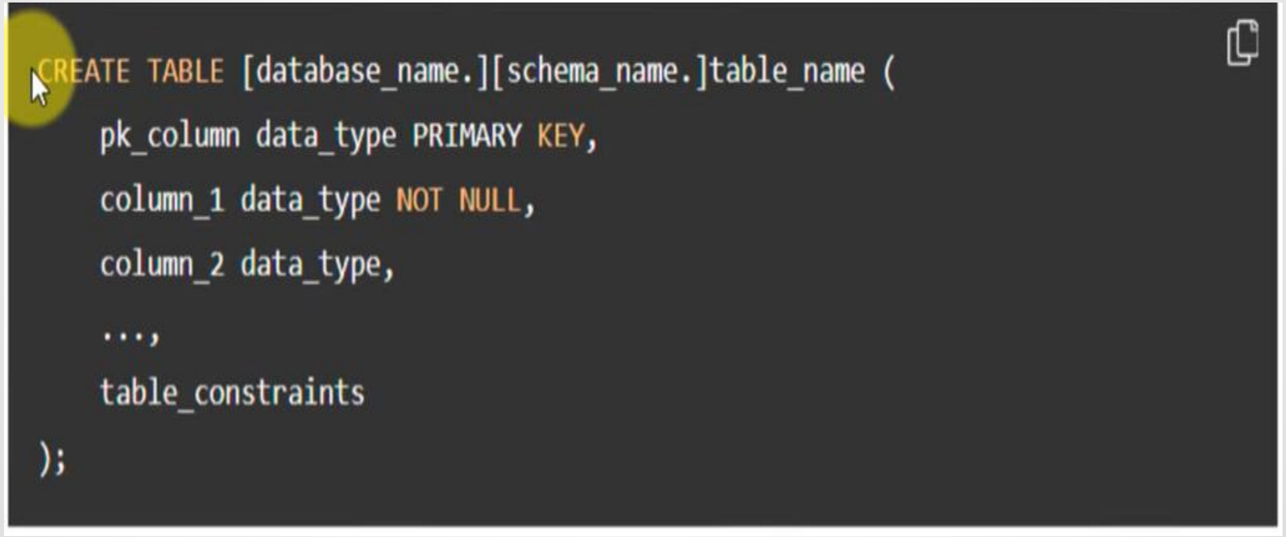


Customers

	CustomerId	FirstName	LastName	DateCreated	Cli
⊞	1	Homer	Simpson	13/06/2014 3:33:37 PM	
⊞	2	Peter	Griffin	13/06/2014 9:09:56 PM	
⊞	3	Stewie	Griffin	13/06/2014 9:16:07 PM	
⊞	4	Brian	Griffin	13/06/2014 9:16:36 PM	
⊞	5	Cosmo	Kramer	13/06/2014 9:16:41 PM	
⊞	6	Philip	Fry	13/06/2014 9:17:02 PM	
⊞	7	Amy	Wong	13/06/2014 9:22:05 PM	
⊞	8	Hubert J.	Farnsworth	13/06/2014 9:22:19 PM	
⊞	9	Marge	Simpson	13/06/2014 9:22:37 PM	
⊞	10	Bender	Rodríguez	13/06/2014 9:22:52 PM	
⊞	11	Turanga	Leela	13/06/2014 9:23:37 PM	

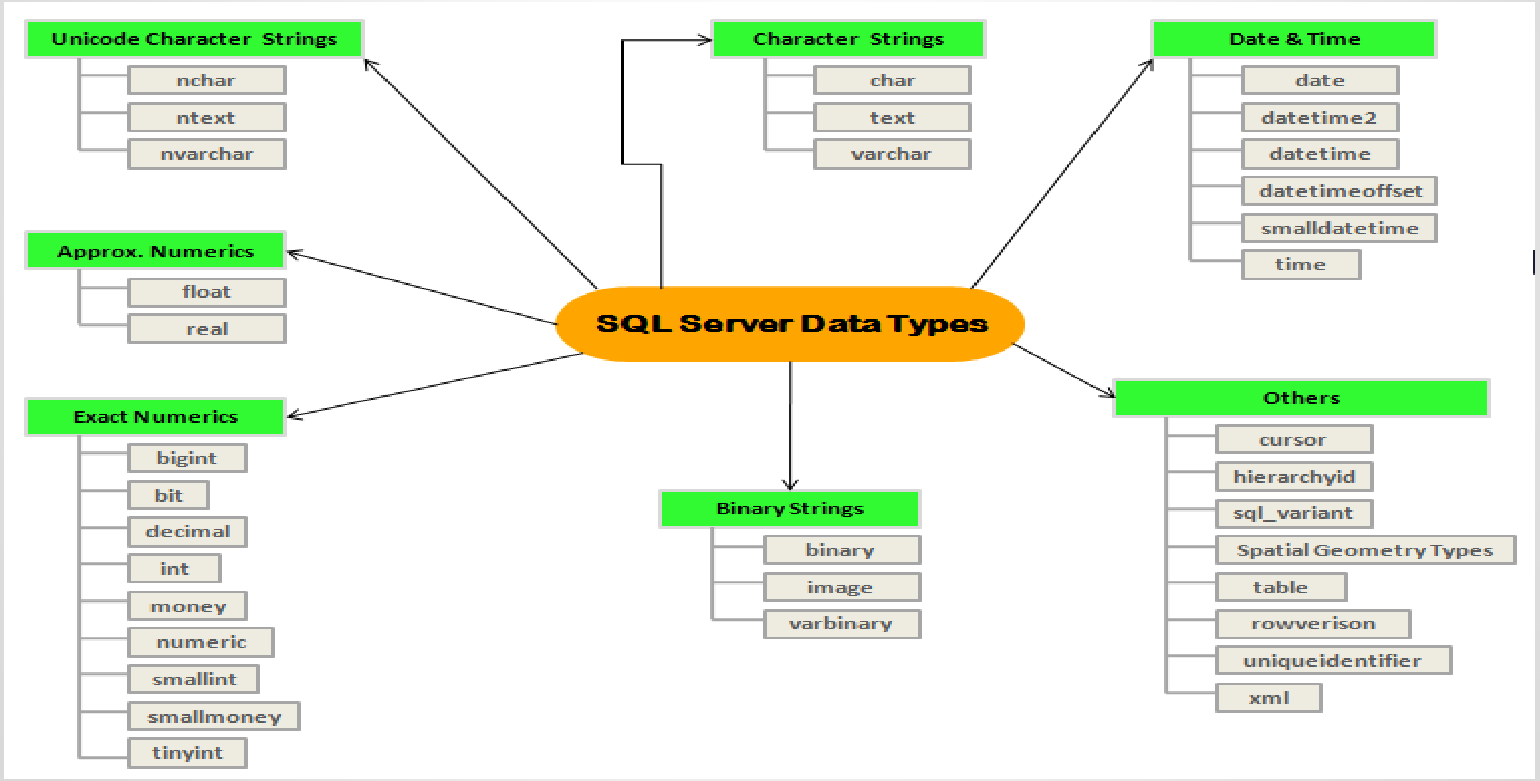
Create Table

Create Table Statement overview



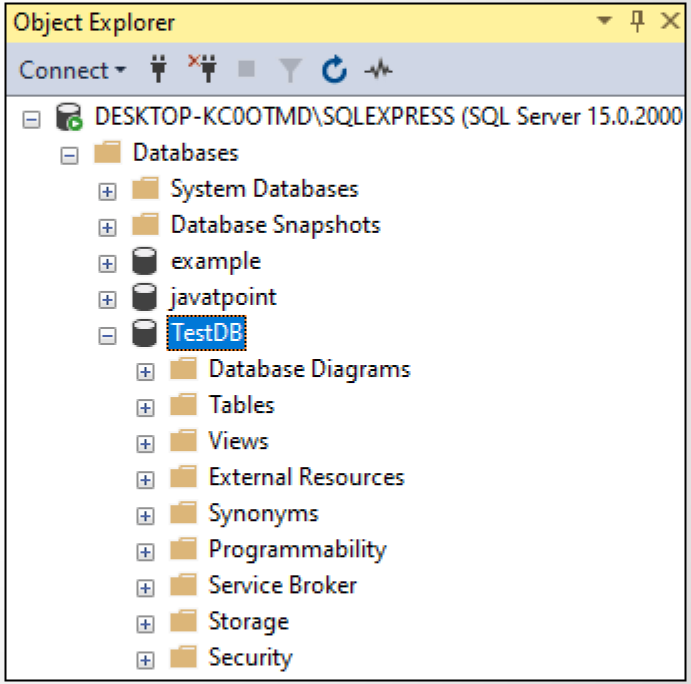
```
CREATE TABLE [database_name.][schema_name.]table_name (
    pk_column data_type PRIMARY KEY,
    column_1 data_type NOT NULL,
    column_2 data_type,
    ...,
    table_constraints
);
```

SQL Servr DataTypes



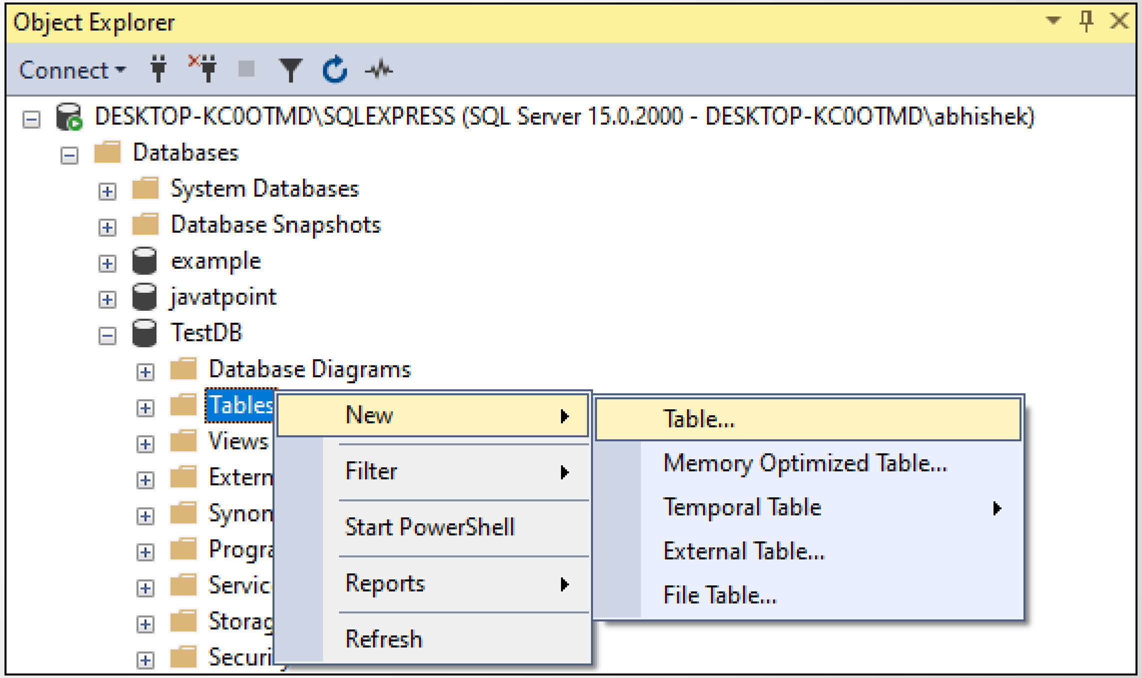
Create table

Step 1: select the desired database in which you want to create a table and expand it. It will display the sub-menu such as Database Diagrams, Tables, Views, and, as shown in the below screen.



Create table

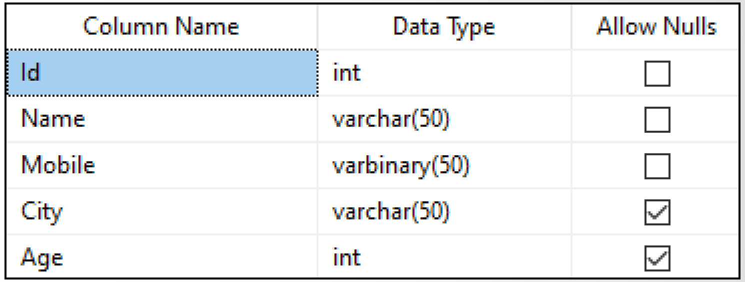
Step 2: The next step is to select the **Tables** folder, right-click on it, we will get the pop menu. Clicking on the **New** option will display a drop-down list where we will choose the **Table** option. See the below image:



Create table

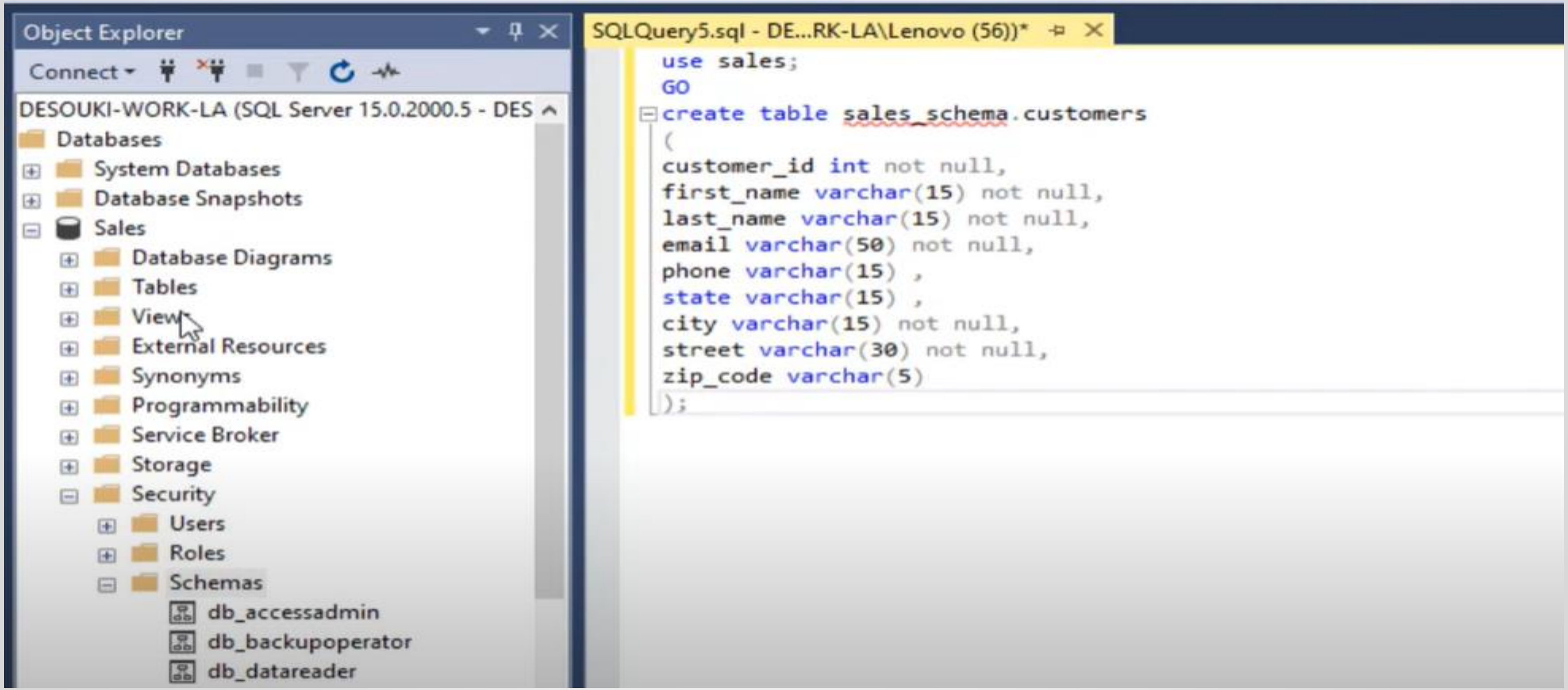
Step 3: Once we click the Table option, we will get the **Table Designer window**. This window will include the column name, data types, and Not Null constraint to select whether to allow nulls or not for each column. For example, we want to create a table named 'Person' that will store four columns:

- FirstName
- LastName
- Mobile



Column Name	Data Type	Allow Nulls
Id	int	☐
Name	varchar(50)	☐
Mobile	varbinary(50)	☐
City	varchar(50)	✓
Age	int	✓

Create table



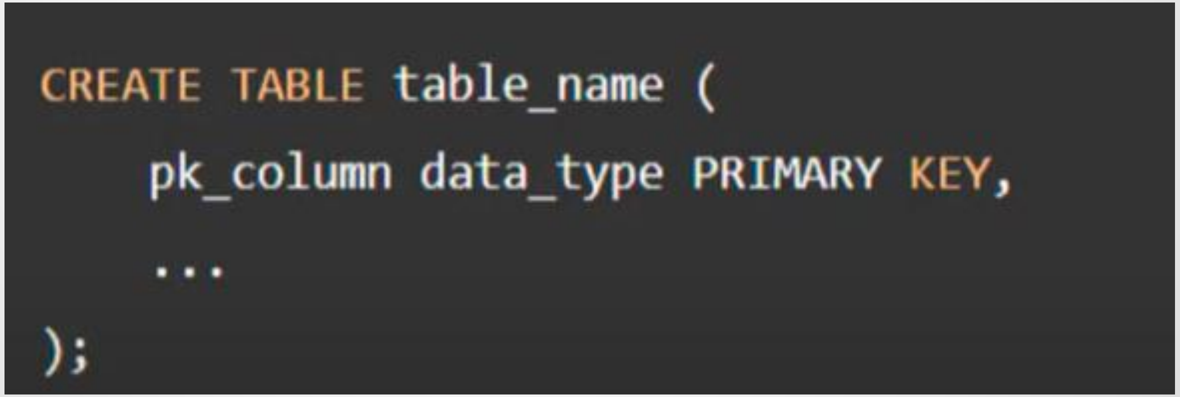
```sql
use sales;
GO
create table sales_schema.customers
(
customer_id int not null,
first_name varchar(15) not null,
last_name varchar(15) not null,
email varchar(50) not null,
phone varchar(15) ,
state varchar(15) ,
city varchar(15) not null,
street varchar(30) not null,
zip_code varchar(5)
);
```

Constraints

1. **Not Null**
2. **Primary key**
3. **Unique**
4. **Check**
5. **Foreign Key**

Constraints

Primary key

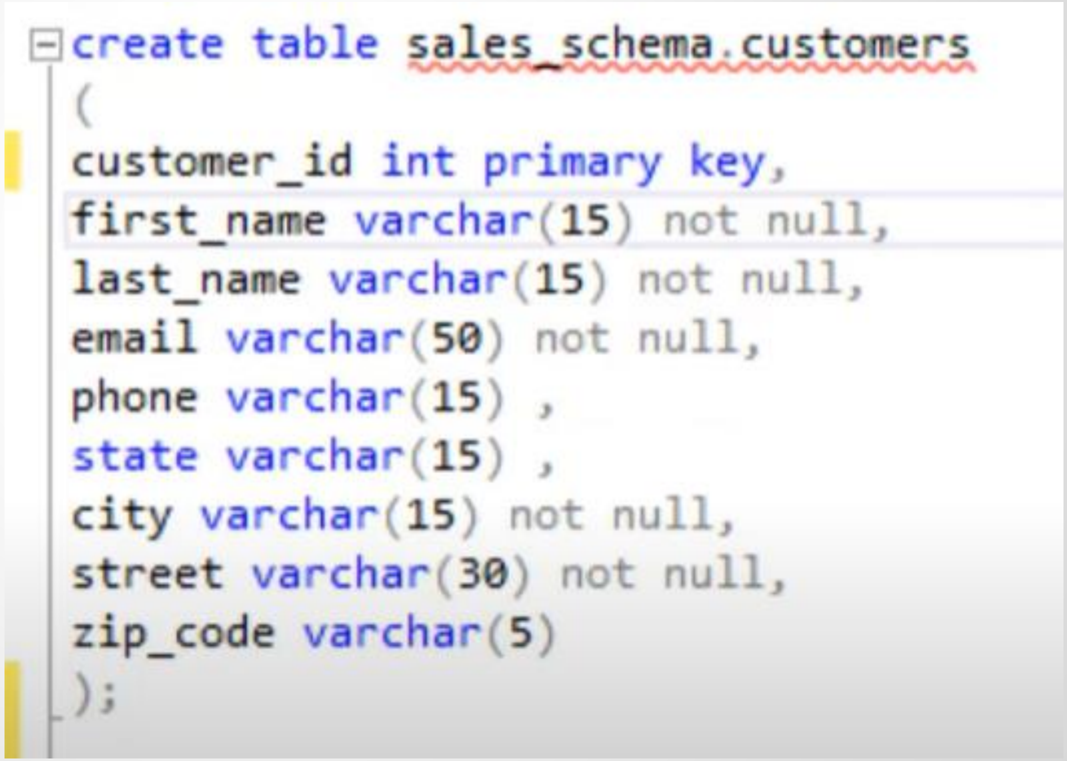


```
CREATE TABLE table_name (
    pk_column data_type PRIMARY KEY,
    ...
);
```

Constraints

Primary key

Column Level

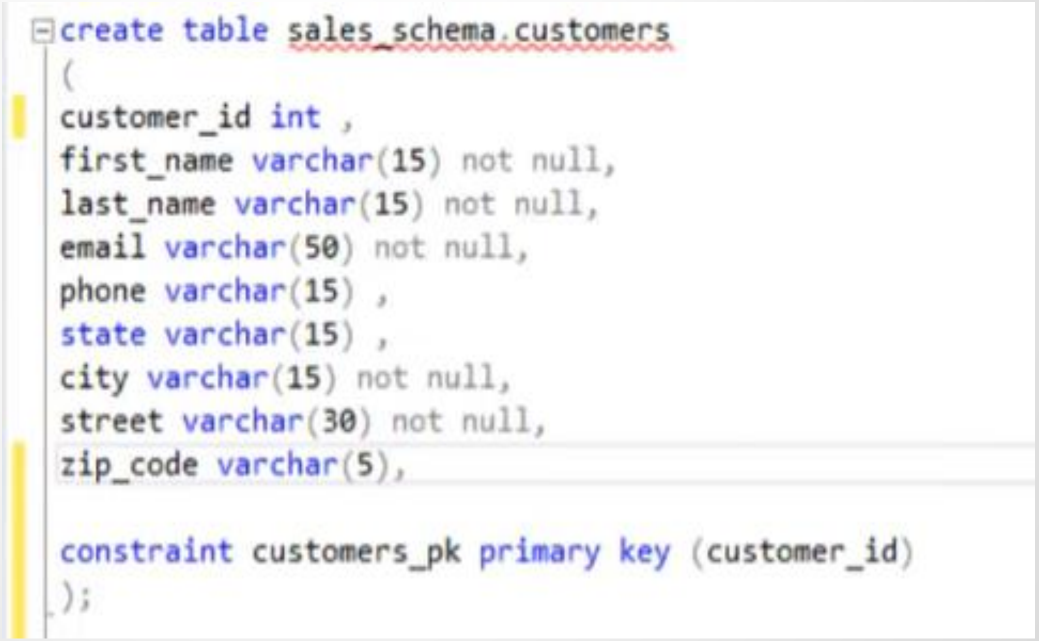


```
create table sales_schema.customers
(
customer_id int primary key,
first_name varchar(15) not null,
last_name varchar(15) not null,
email varchar(50) not null,
phone varchar(15) ,
state varchar(15) ,
city varchar(15) not null,
street varchar(30) not null,
zip_code varchar(5)
);
```

Constraints

Primary key

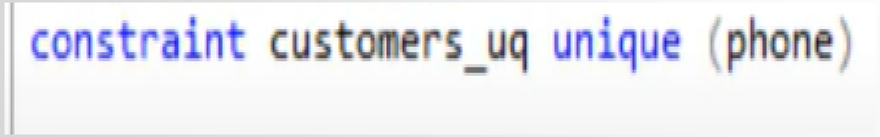
Table Level



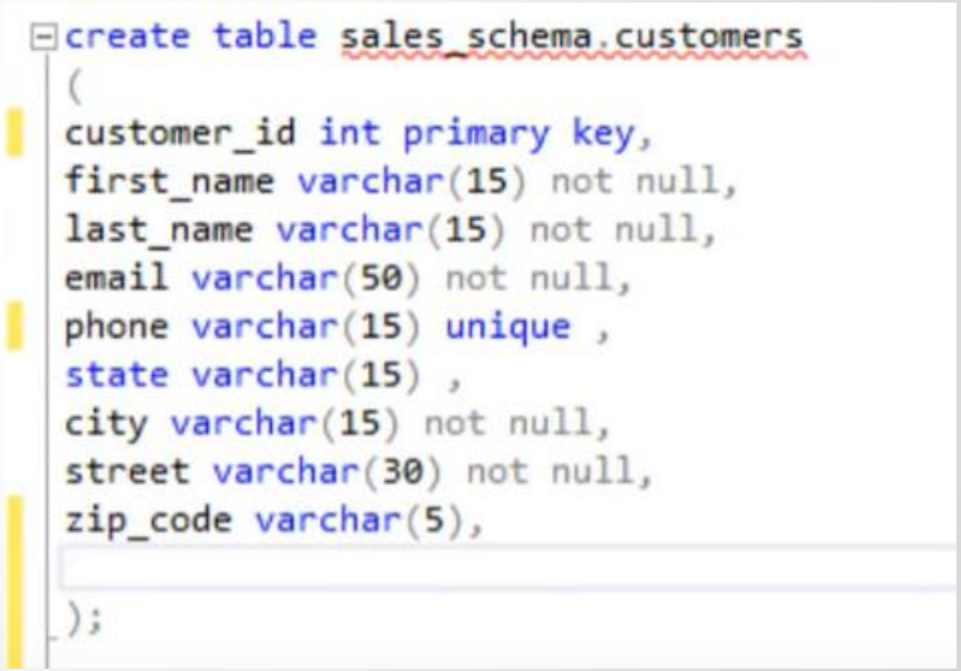
```
create table sales_schema.customers
(
customer_id int ,
first_name varchar(15) not null,
last_name varchar(15) not null,
email varchar(50) not null,
phone varchar(15) ,
state varchar(15) ,
city varchar(15) not null,
street varchar(30) not null,
zip_code varchar(5),

constraint customers_pk primary key (customer_id)
);
```

Constraints

Unique Constraint



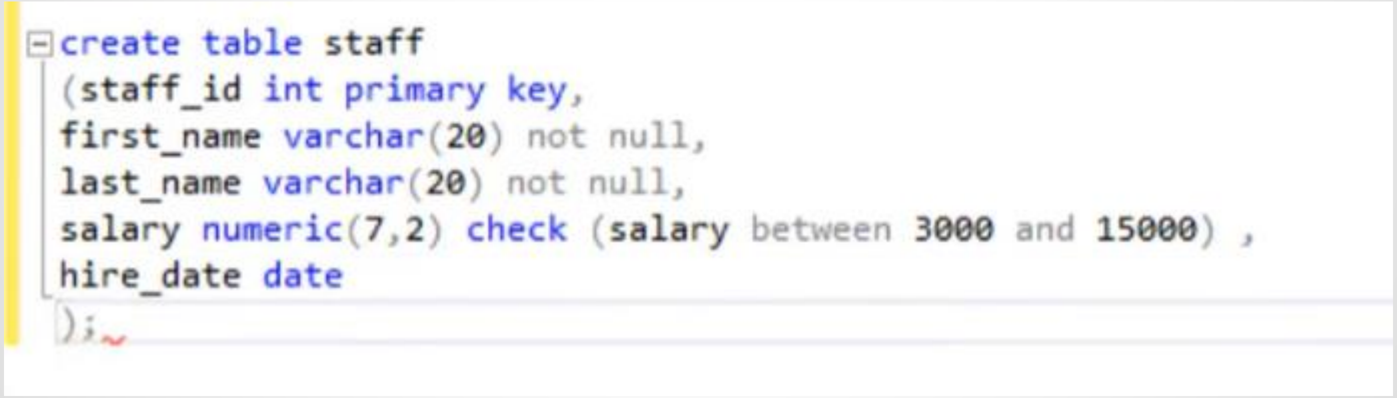
```
constraint customers_uq unique (phone)
```



```
create table sales_schema.customers
(
customer_id int primary key,
first_name varchar(15) not null,
last_name varchar(15) not null,
email varchar(50) not null,
phone varchar(15) unique ,
state varchar(15) ,
city varchar(15) not null,
street varchar(30) not null,
zip_code varchar(5),

);
```

Constraints

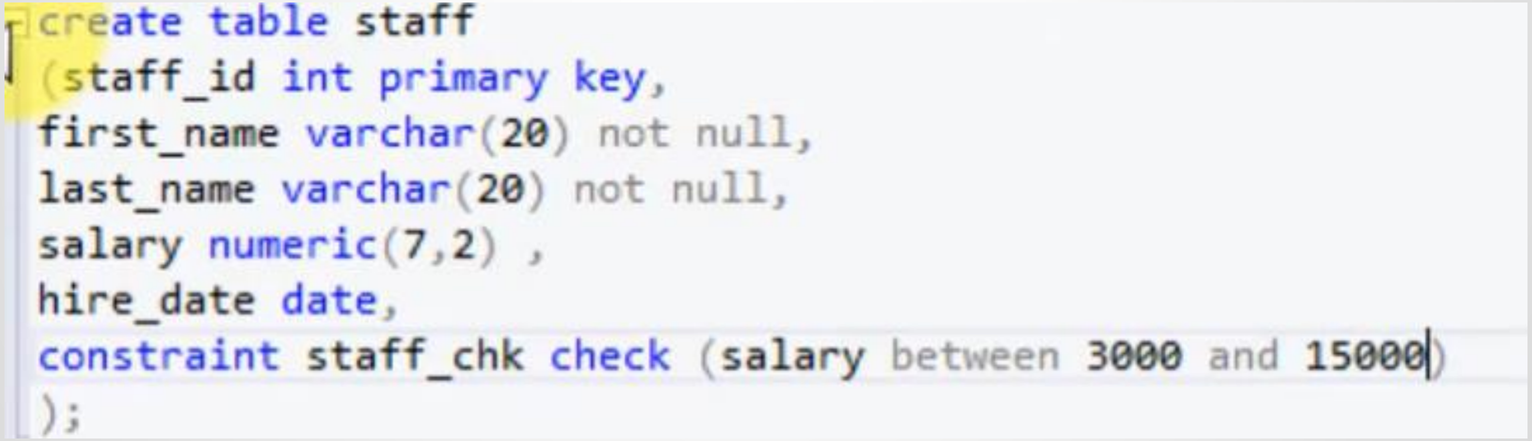
Check Constraint



```
create table staff
(staff_id int primary key,
first_name varchar(20) not null,
last_name varchar(20) not null,
salary numeric(7,2) check (salary between 3000 and 15000) ,
hire_date date
);
```

Constraints

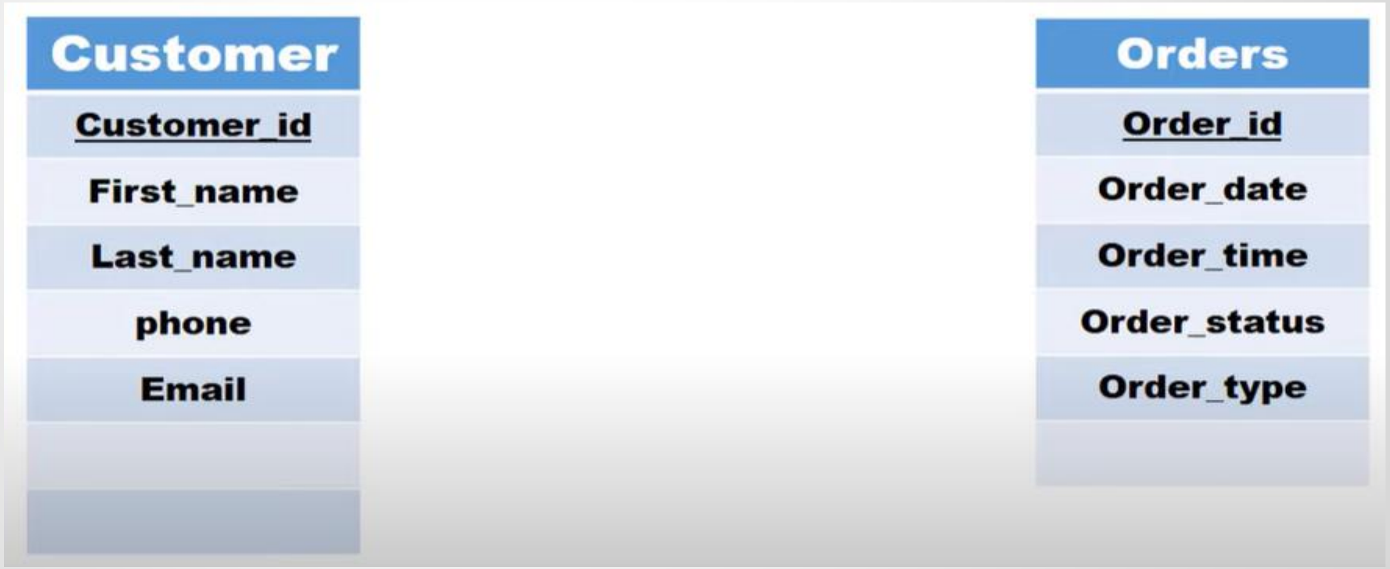
Check Constraint



```
create table staff
(staff_id int primary key,
first_name varchar(20) not null,
last_name varchar(20) not null,
salary numeric(7,2) ,
hire_date date,
constraint staff_chk check (salary between 3000 and 15000)
);
```

Constraints

Foreign Key

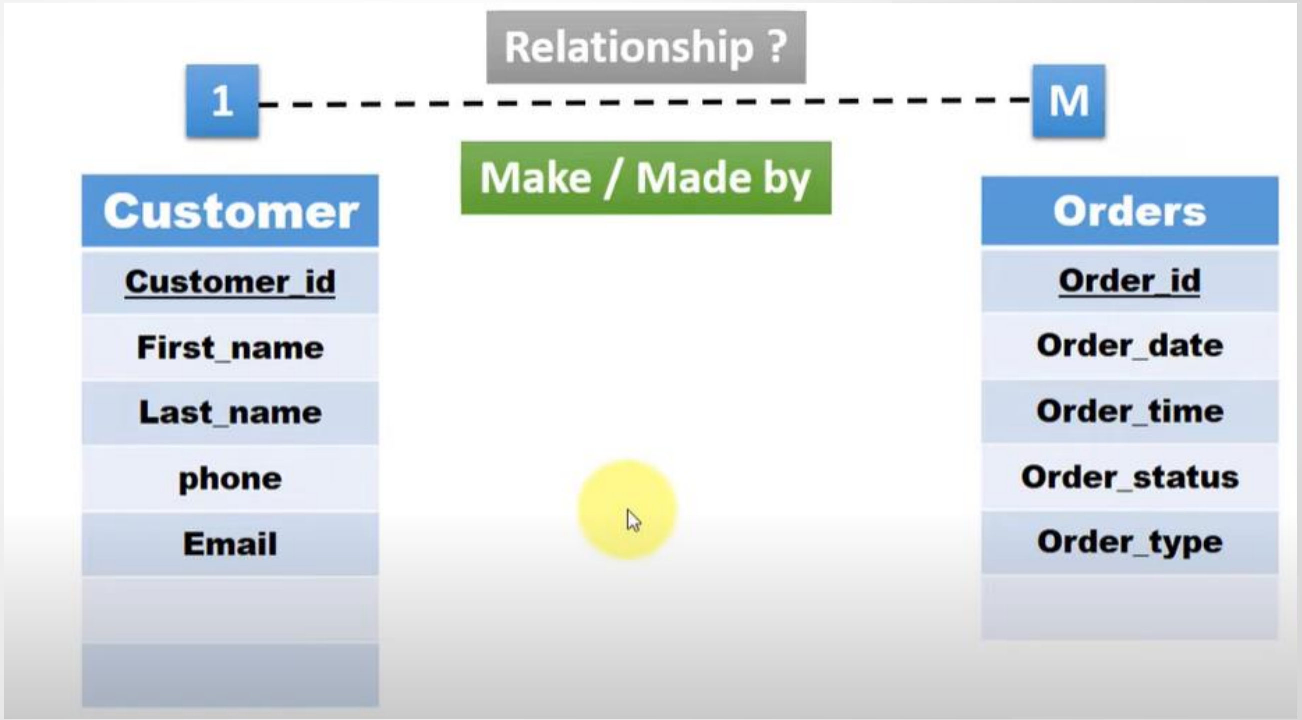


Customer
Customer_id
First_name
Last_name
phone
Email

Orders
Order_id
Order_date
Order_time
Order_status
Order_type

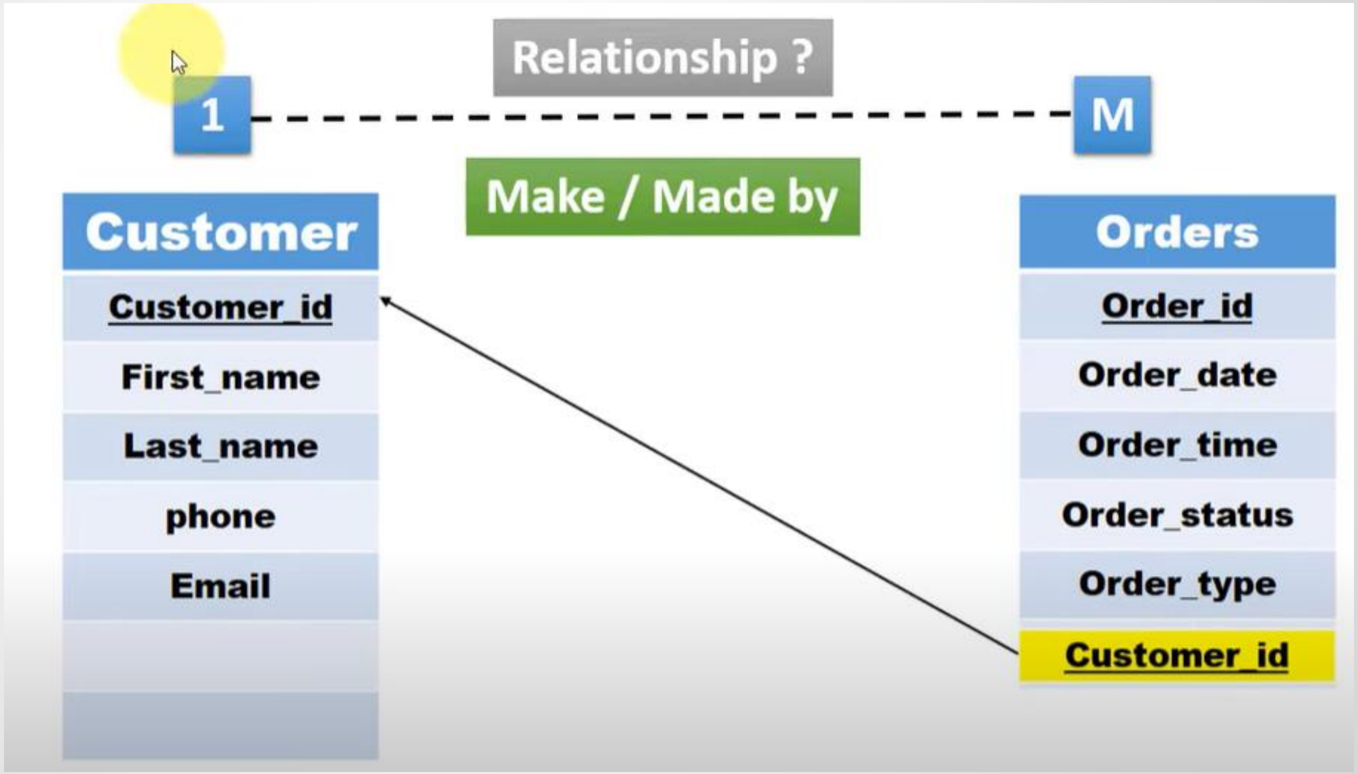
Constraints

Foreign Key



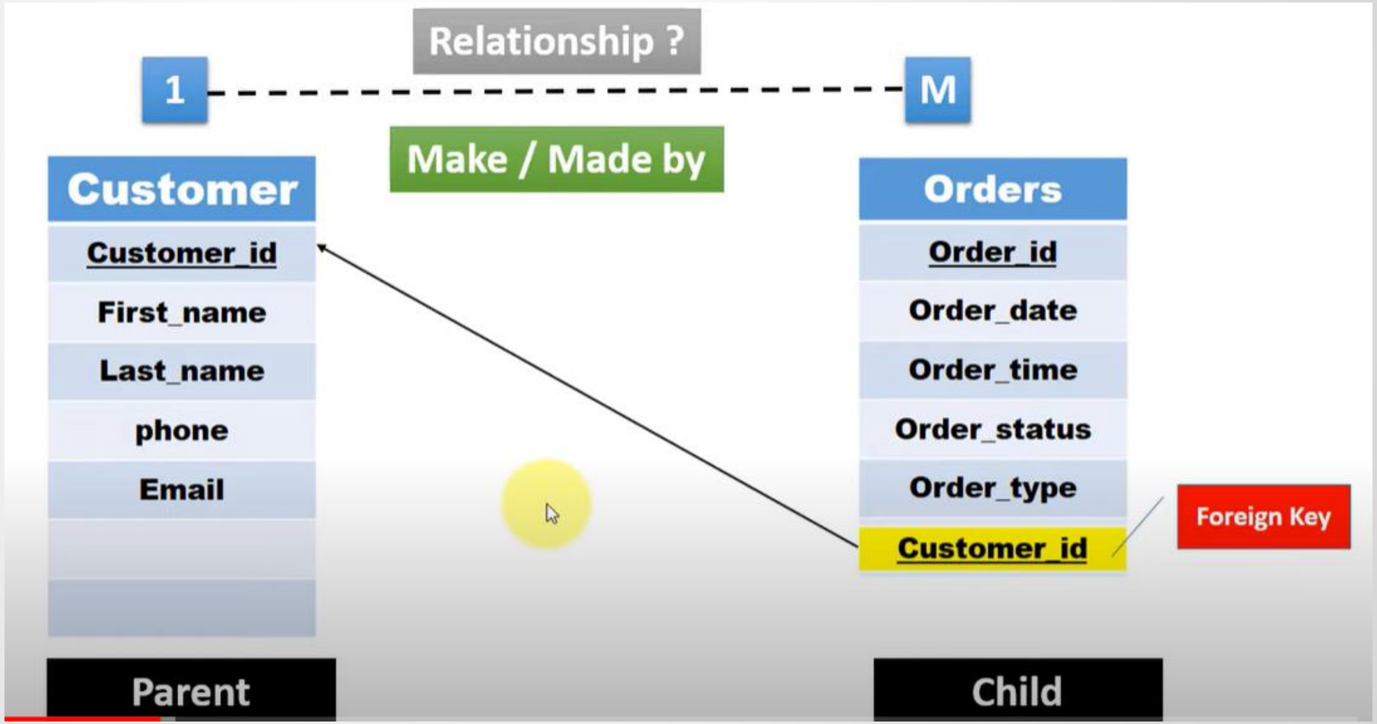
Constraints

Foreign Key



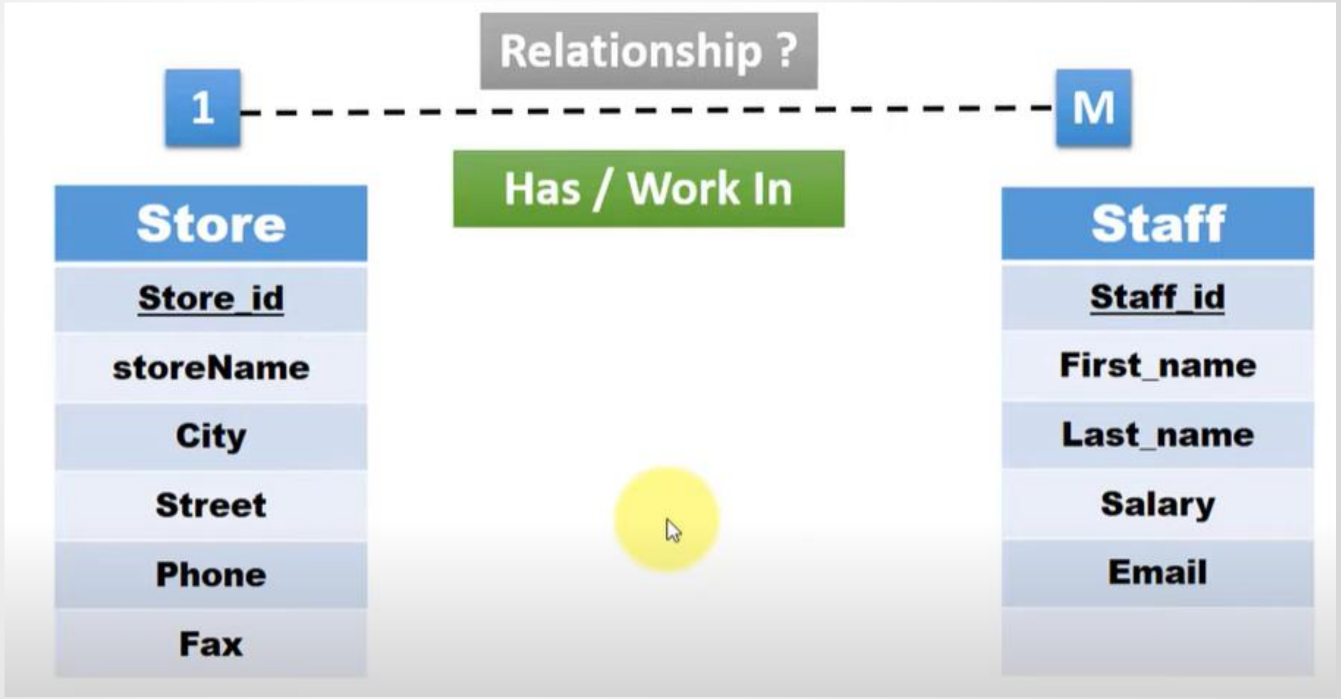
Constraints

Foreign Key



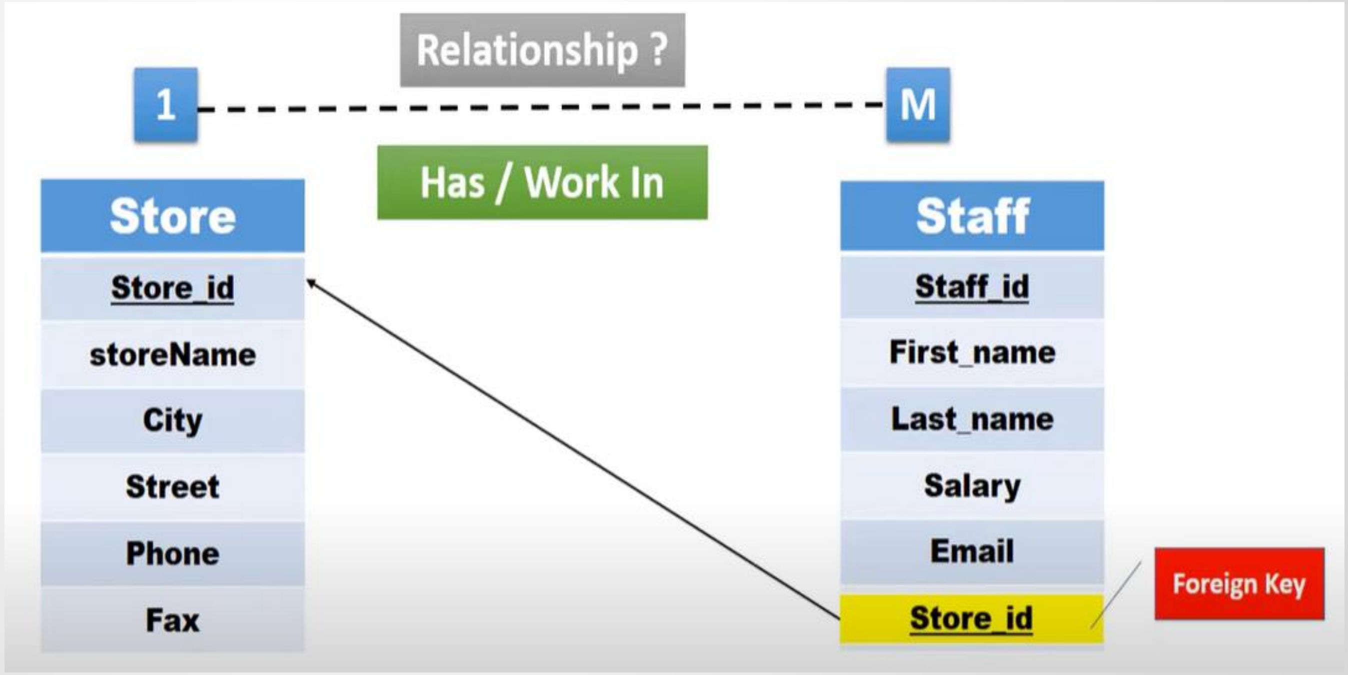
Constraints

Foreign Key



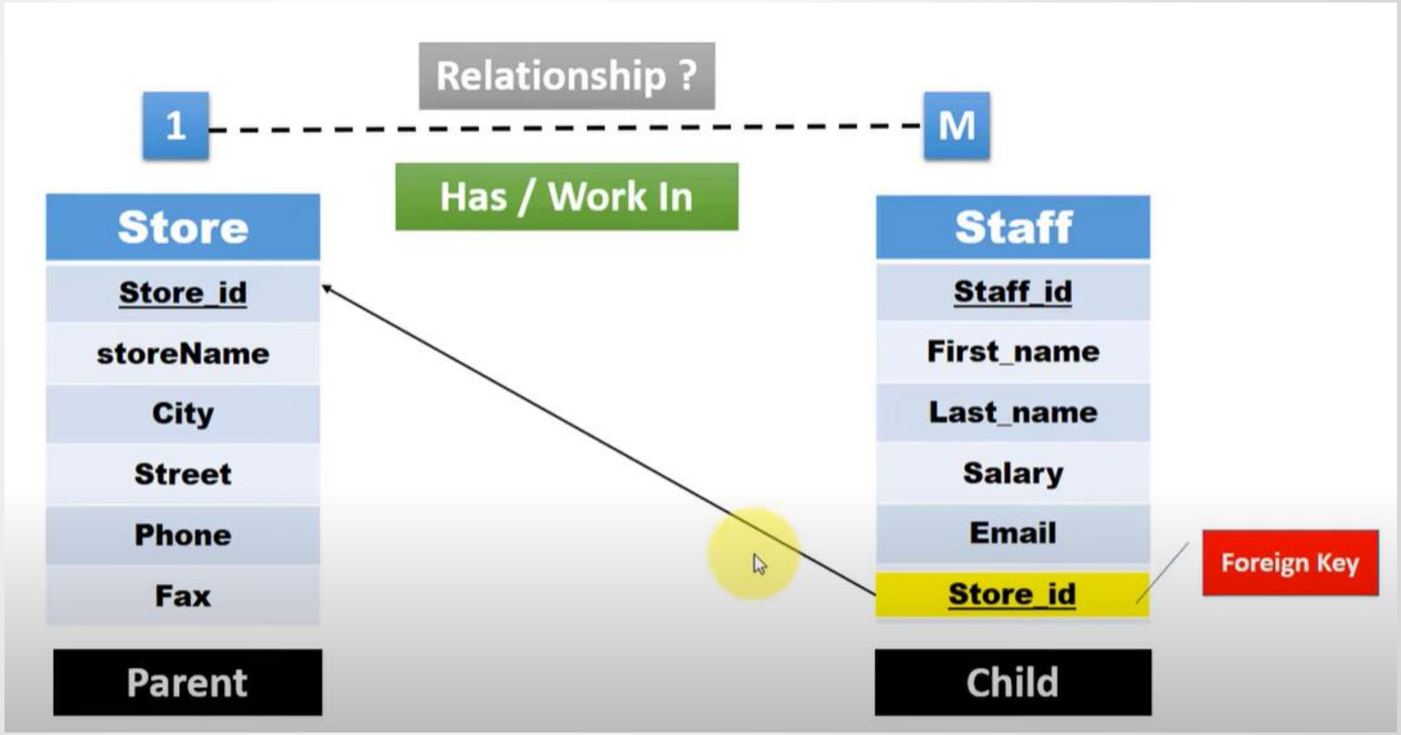
Constraints

Foreign Key



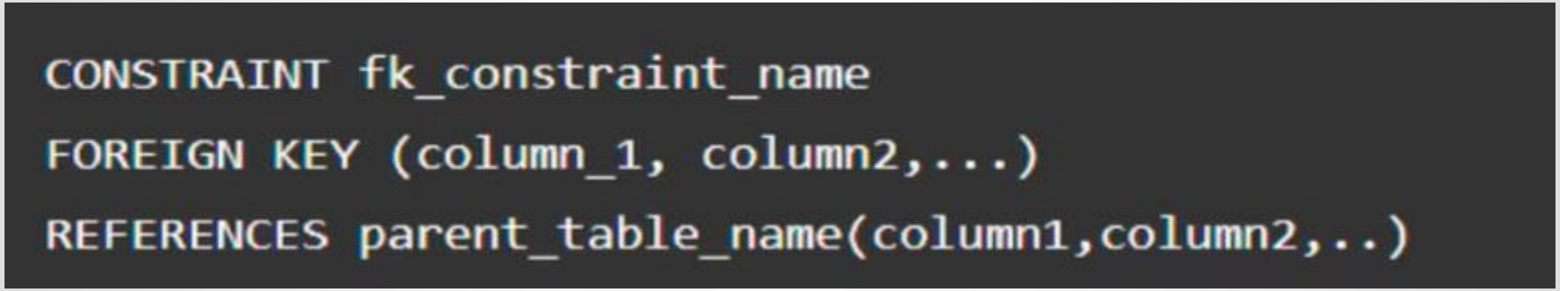
Constraints

Foreign Key



Constraints

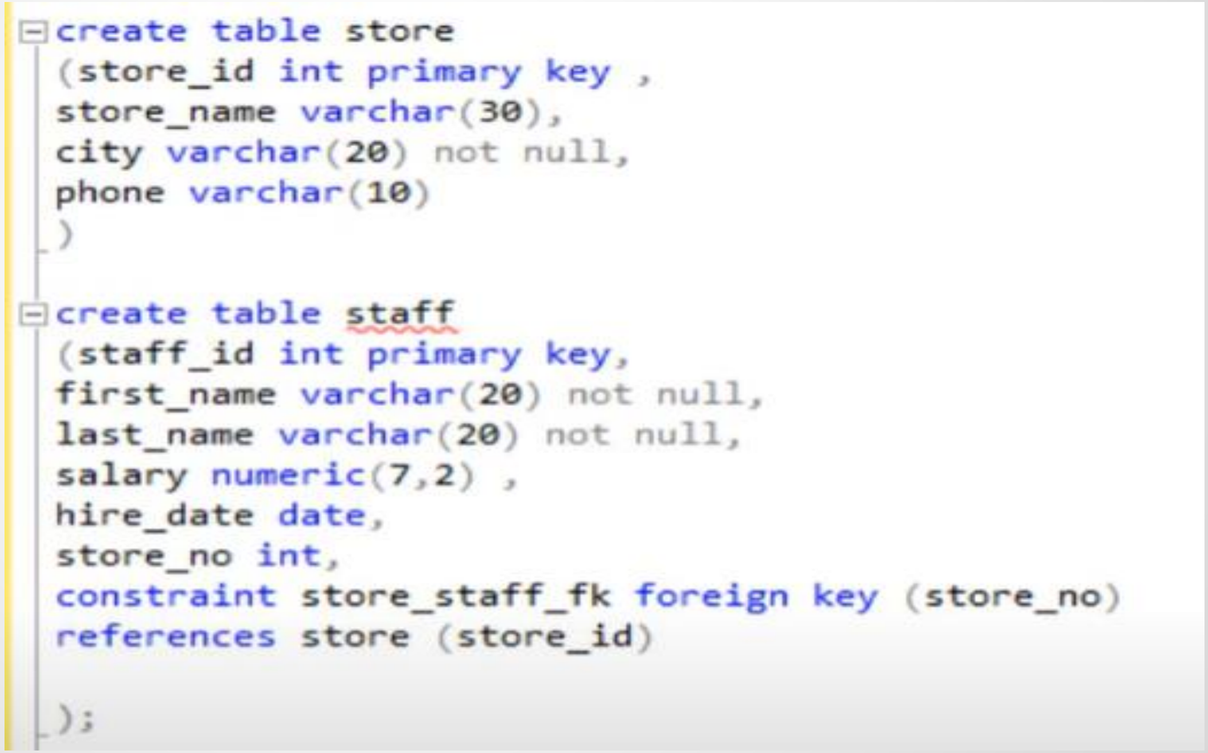
Foreign Key



```
CONSTRAINT fk_constraint_name
FOREIGN KEY (column_1, column2,...)
REFERENCES parent_table_name(column1,column2,..)
```

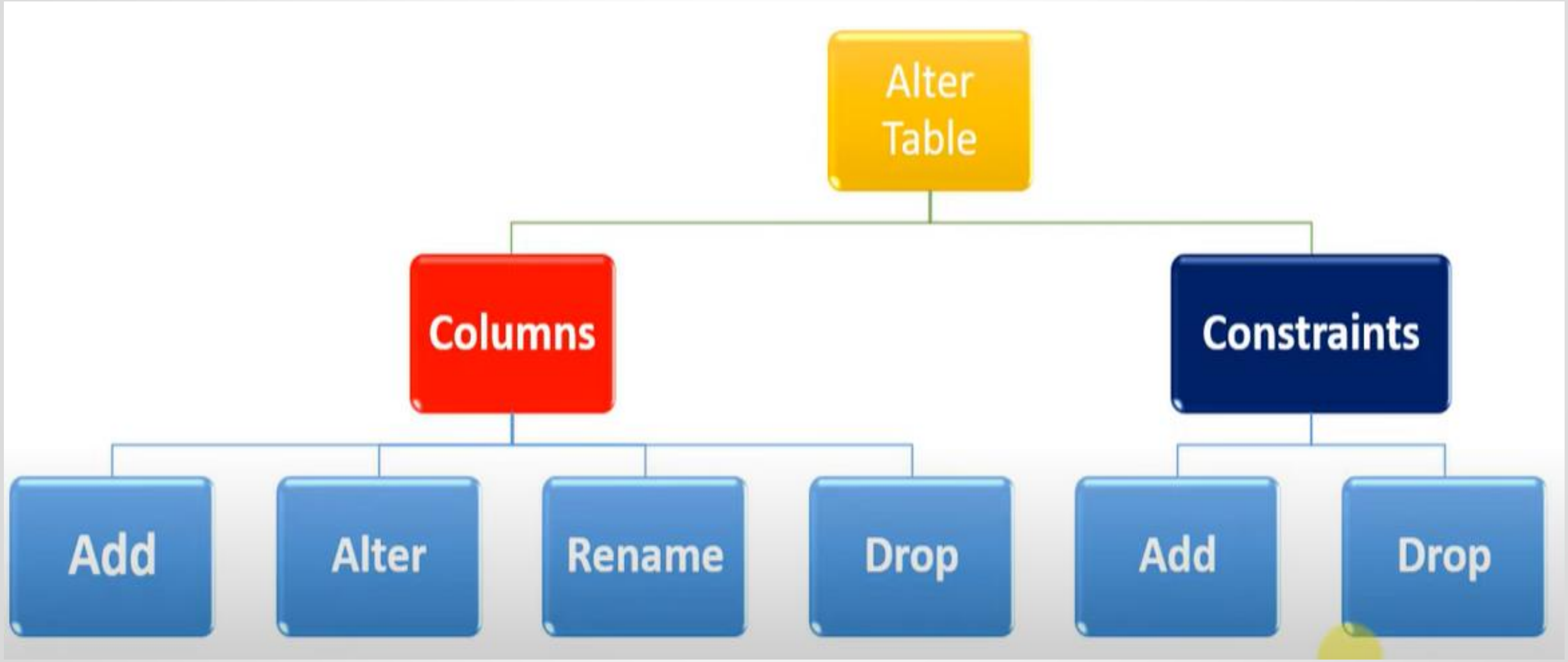
Constraints

Foreign Key



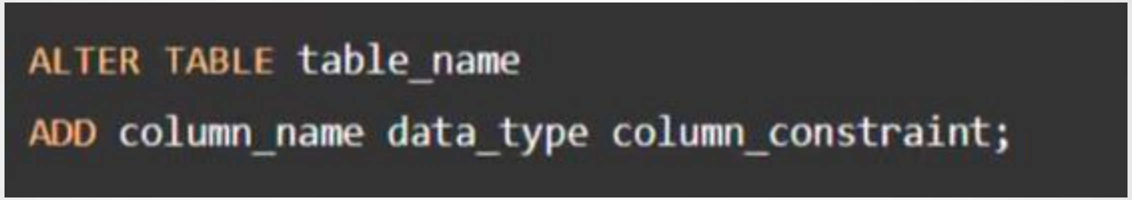
```
create table store
(store_id int primary key ,
store_name varchar(30),
city varchar(20) not null,
phone varchar(10)
)

create table staff
(staff_id int primary key,
first_name varchar(20) not null,
last_name varchar(20) not null,
salary numeric(7,2) ,
hire_date date,
store_no int,
constraint store_staff_fk foreign key (store_no)
references store (store_id)

);
```

Alter Table Statement

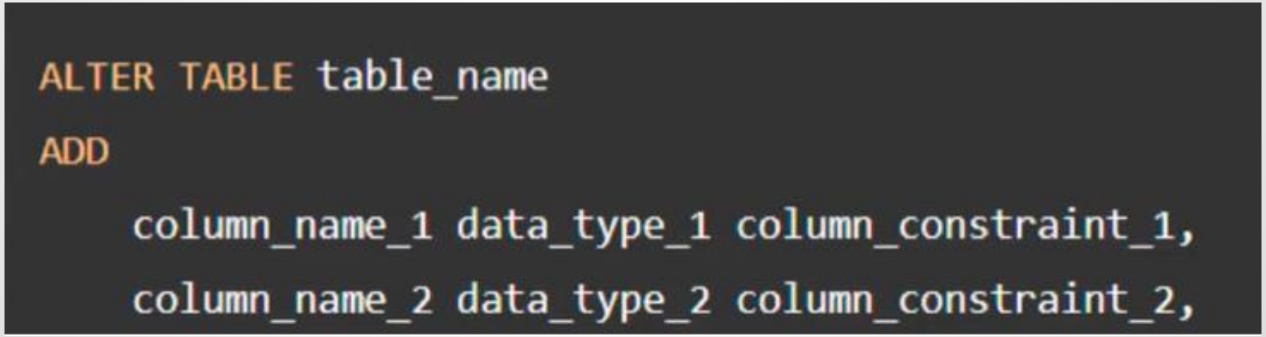


Alter Table Statement

Alter table ADD



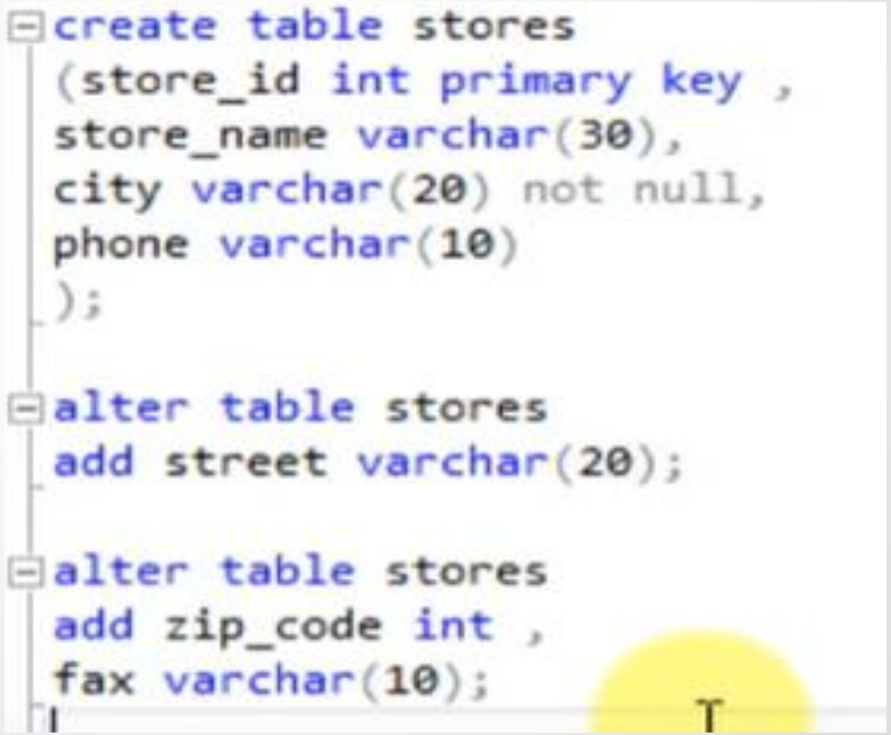
```
ALTER TABLE table_name
ADD column_name data_type column_constraint;
```



```
ALTER TABLE table_name
ADD
    column_name_1 data_type_1 column_constraint_1,
    column_name_2 data_type_2 column_constraint_2,
```

Alter Table Statement

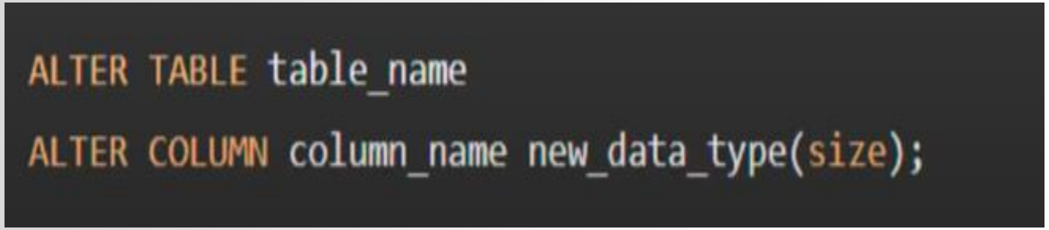
Alter table ADD



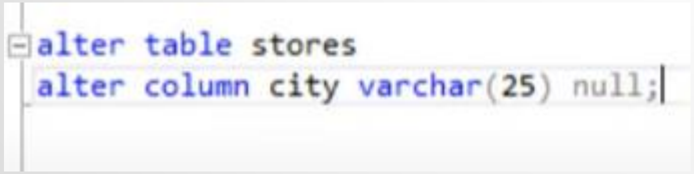
```
create table stores
(store_id int primary key ,
store_name varchar(30),
city varchar(20) not null,
phone varchar(10)
);

alter table stores
add street varchar(20);

alter table stores
add zip_code int ,
fax varchar(10);
```

Alter Table Statement

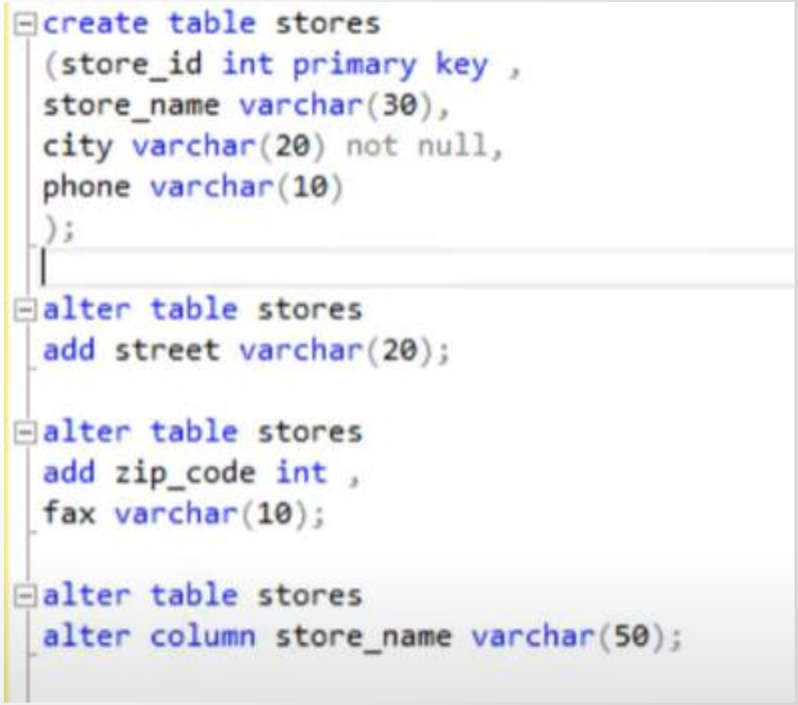
Alter table Modify



```
ALTER TABLE table_name
ALTER COLUMN column_name new_data_type(size);
```



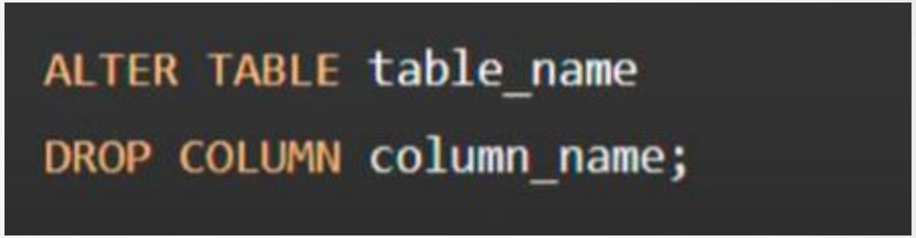
```
alter table stores
alter column city varchar(25) null;
```



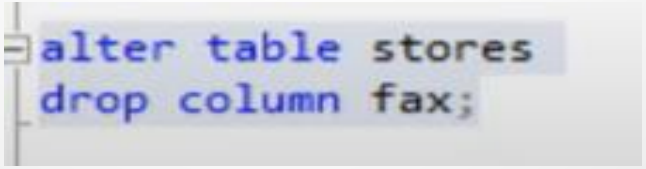
```
create table stores
(store_id int primary key ,
store_name varchar(30),
city varchar(20) not null,
phone varchar(10)
);

alter table stores
add street varchar(20);

alter table stores
add zip_code int ,
fax varchar(10);

alter table stores
alter column store_name varchar(50);
```

Alter Table Statement

Alter table Drop



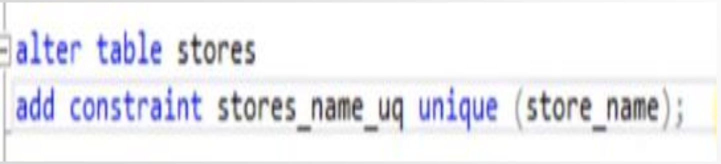
```
ALTER TABLE table_name
DROP COLUMN column_name;
```



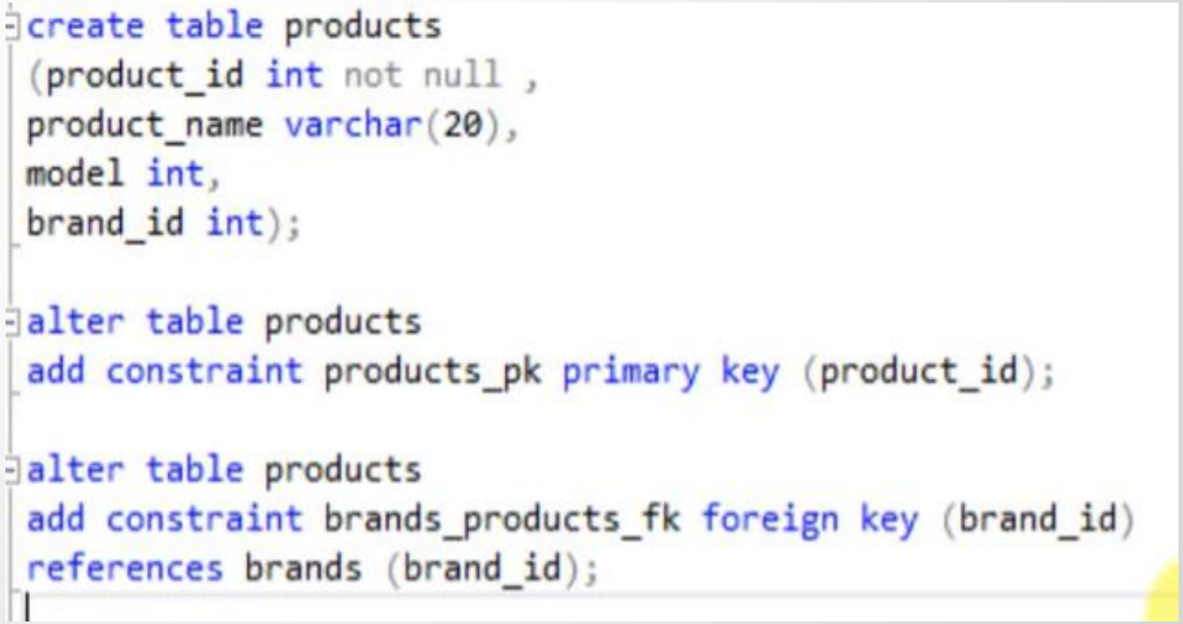
```
alter table stores
drop column fax;
```

Alter Table Statement

Alter table Constraint



```
alter table stores
add constraint stores_name_uq unique (store_name);
```

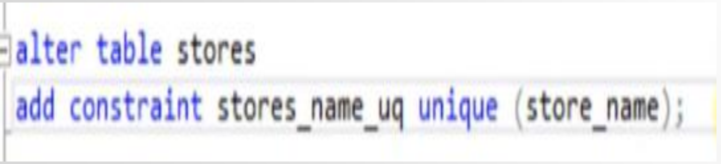


```
create table products
(product_id int not null ,
product_name varchar(20),
model int,
brand_id int);

alter table products
add constraint products_pk primary key (product_id);

alter table products
add constraint brands_products_fk foreign key (brand_id)
references brands (brand_id);
```

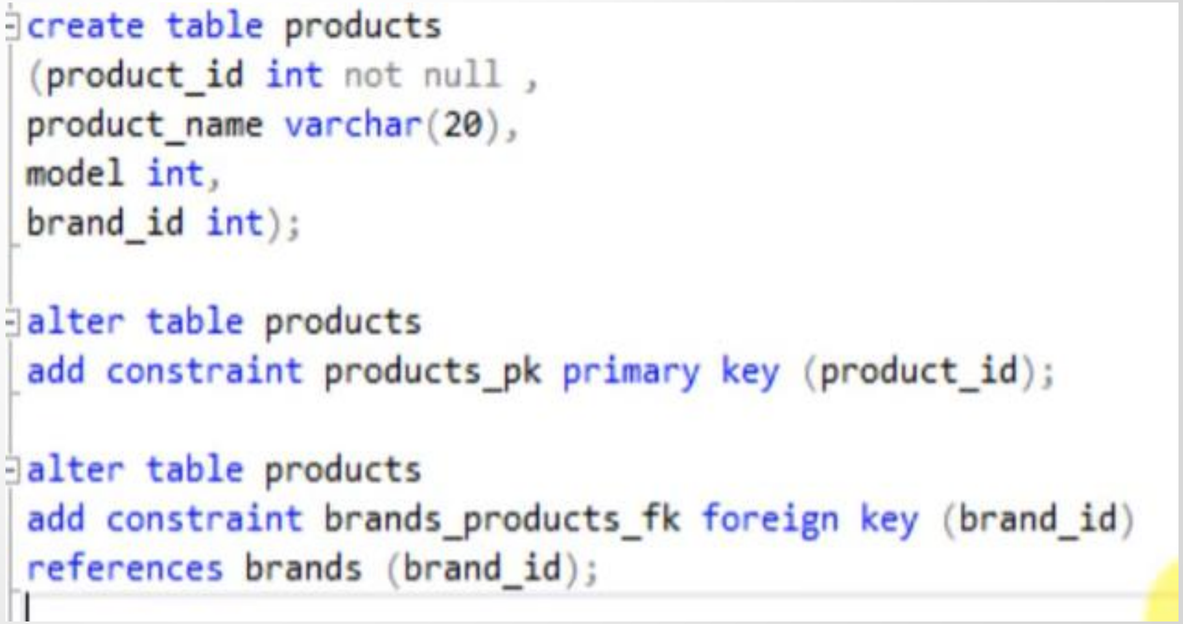
Alter Table Statement

Alter table Constraint

ADD Constraint



```
alter table stores
add constraint stores_name_uq unique (store_name);
```

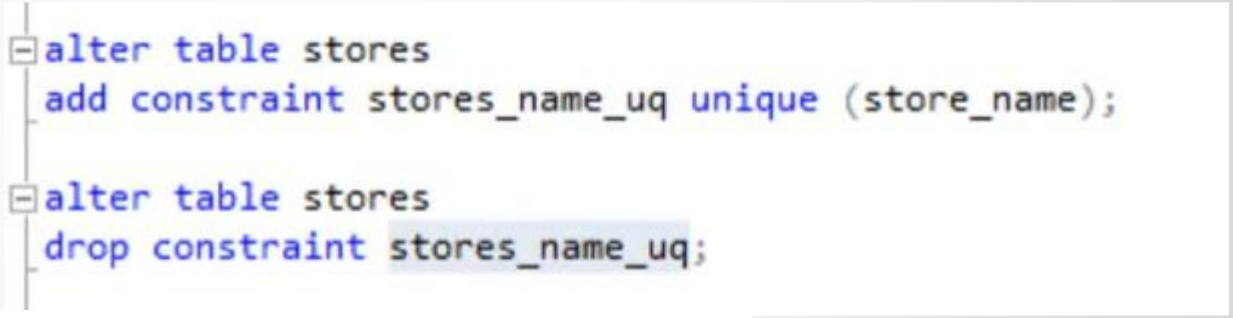


```
create table products
(product_id int not null ,
product_name varchar(20),
model int,
brand_id int);

alter table products
add constraint products_pk primary key (product_id);

alter table products
add constraint brands_products_fk foreign key (brand_id)
references brands (brand_id);
```

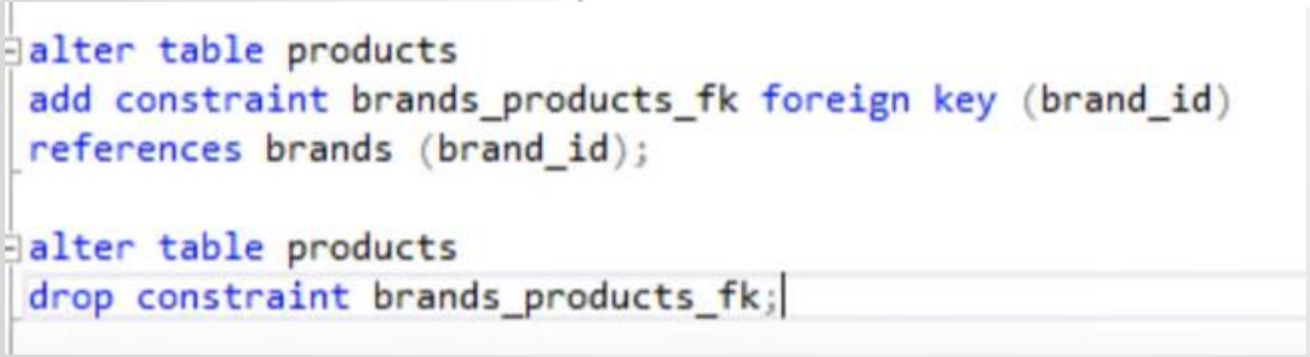
Alter Table Statement

Alter table Constraint

Drop Constraint



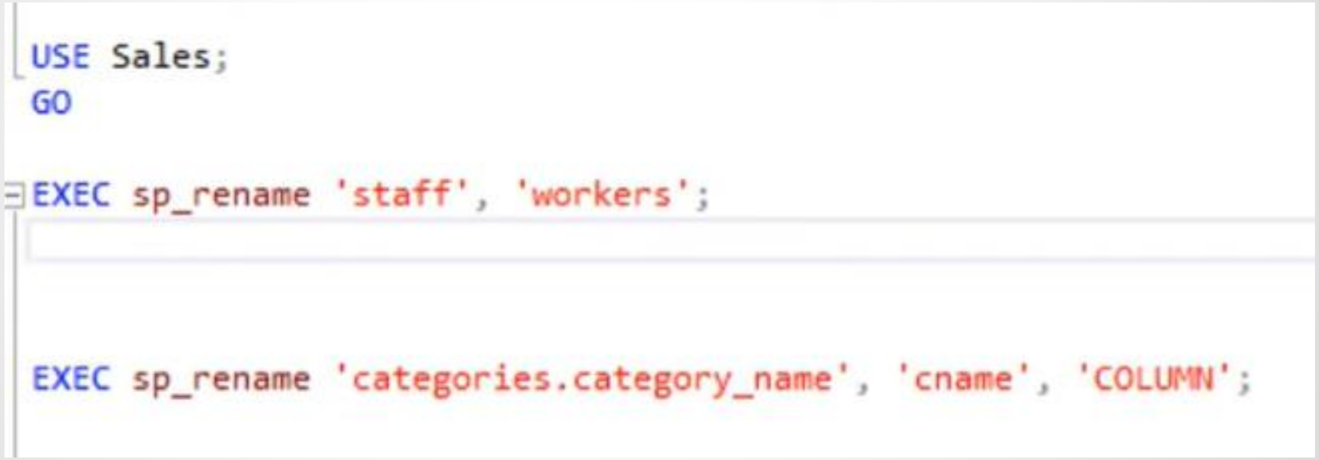
```
alter table stores
add constraint stores_name_uq unique (store_name);

alter table stores
drop constraint stores_name_uq;
```



```
alter table products
add constraint brands_products_fk foreign key (brand_id)
references brands (brand_id);

alter table products
drop constraint brands_products_fk;
```

Alter Table Statement

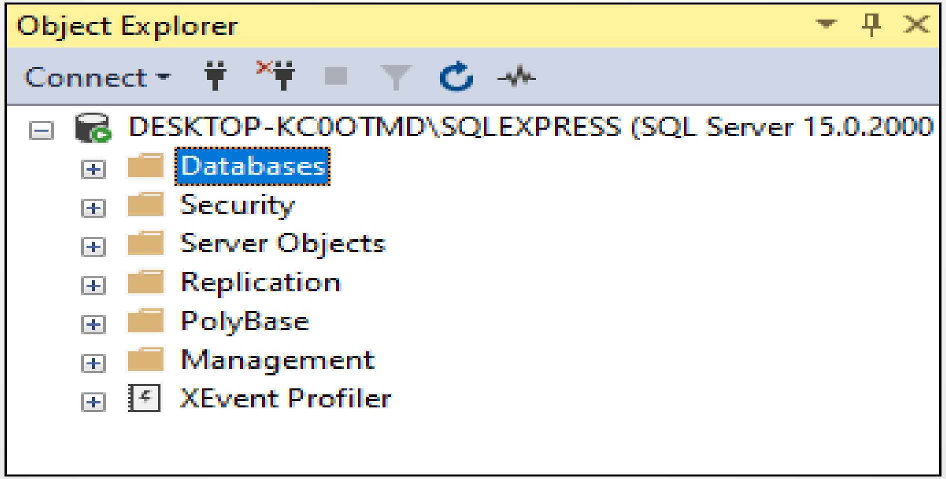
Alter table Rename Table Or Column



```
USE Sales;
GO

EXEC sp_rename 'staff', 'workers';



EXEC sp_rename 'categories.category_name', 'cname', 'COLUMN';
```

Create Database (Object Explorer)

Step1:

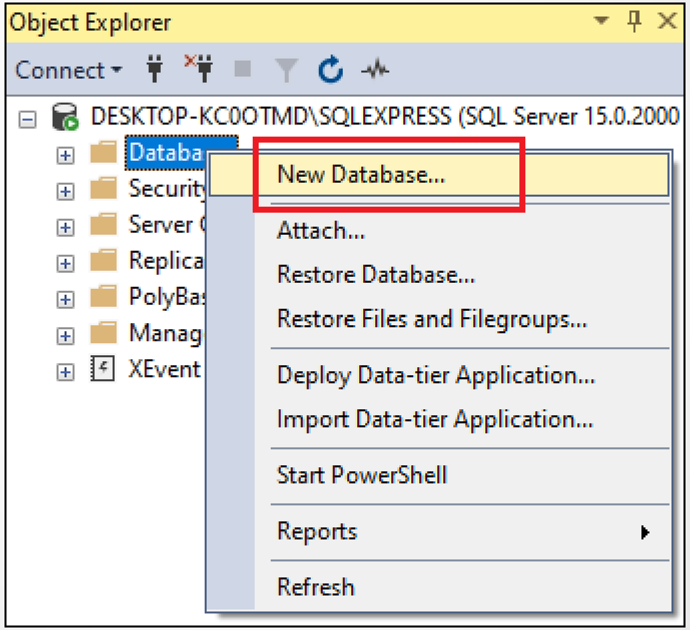

Object Explorer

Connect

DESKTOP-KC0OTMD\SQLEXPRESS (SQL Server 15.0.2000
- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Management
- XEvent Profiler

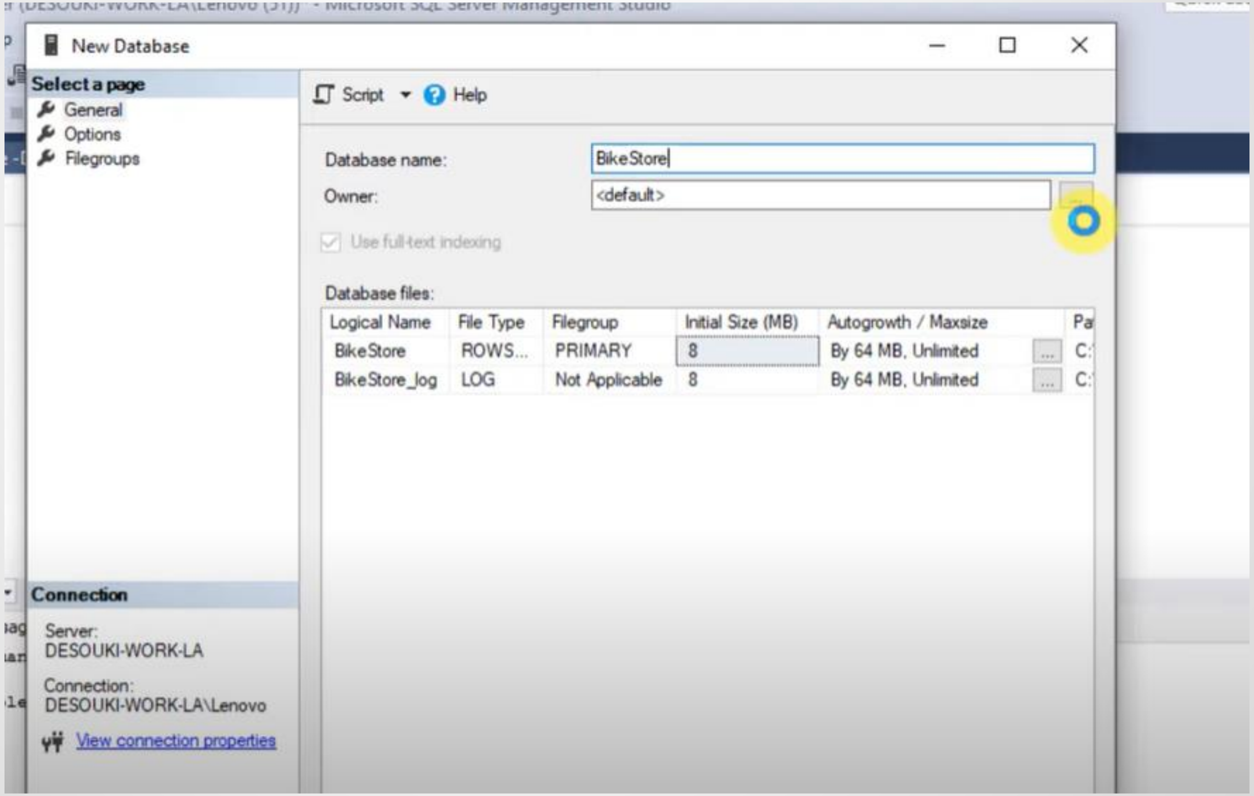
Create Database

Step2:


Object Explorer
Connect
DESKTOP-KC0OTMD\SQLEXPRESS (SQL Server 15.0.2000
Databa
Security
Server
Replica
PolyBas
Manag
XEvent
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

Create Database

Step3:



New Database

Select a page
- General
- Options
- Filegroups

Script ▾ Help

Database name: BikeStore

Owner: <default>

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth / Maxsize	Pa
BikeStore	ROWS...	PRIMARY	8	By 64 MB, Unlimited ...	C:
BikeStore_log	LOG	Not Applicable	8	By 64 MB, Unlimited ...	C:

Connection

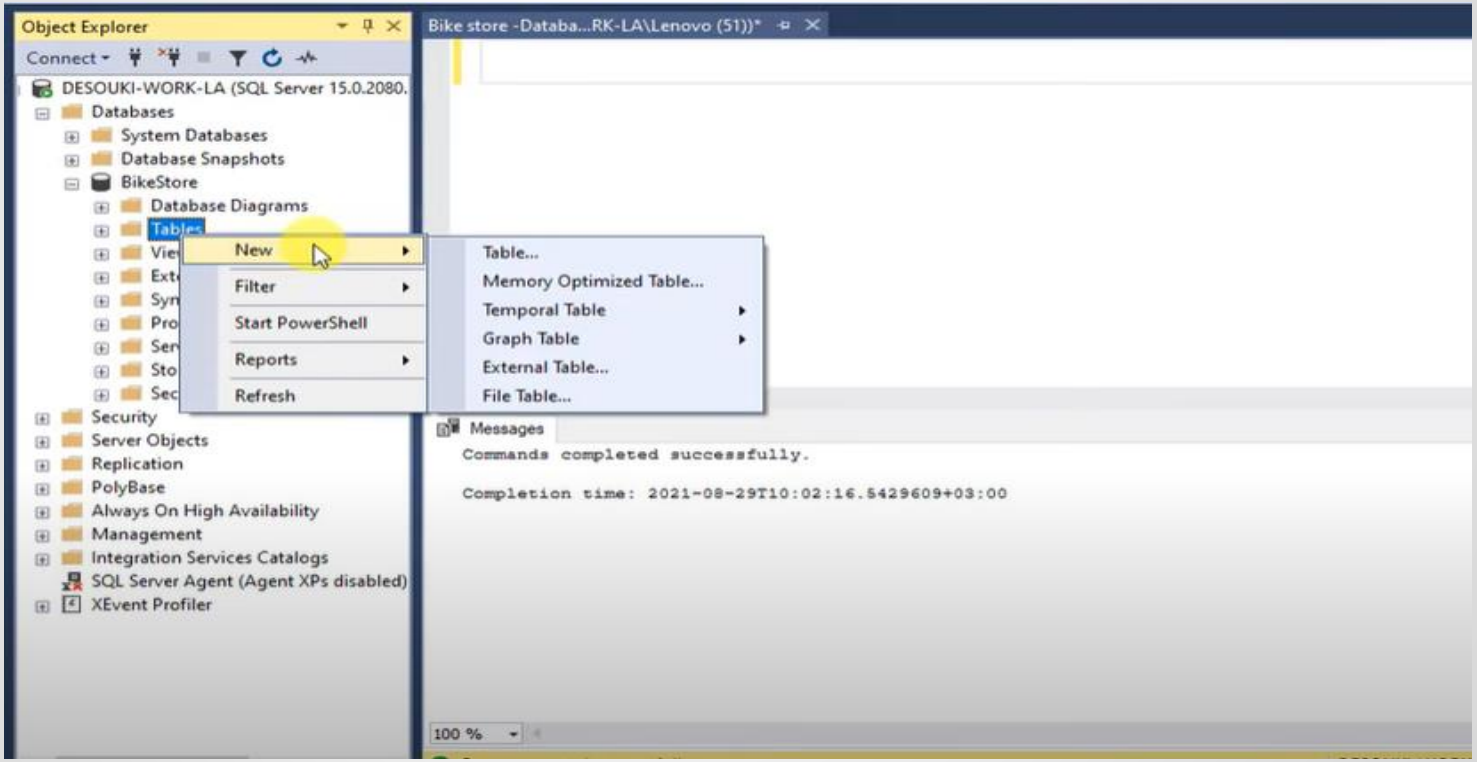
Server:
DESOUKI-WORK-LA

Connection:
DESOUKI-WORK-LA\Lenovo

View connection properties

Create Database

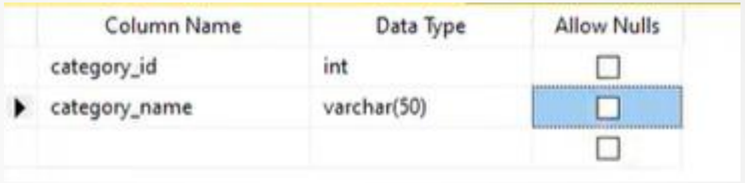
Step4:



```
Commands completed successfully.

Completion time: 2021-08-29T10:02:16.5429609+03:00
```

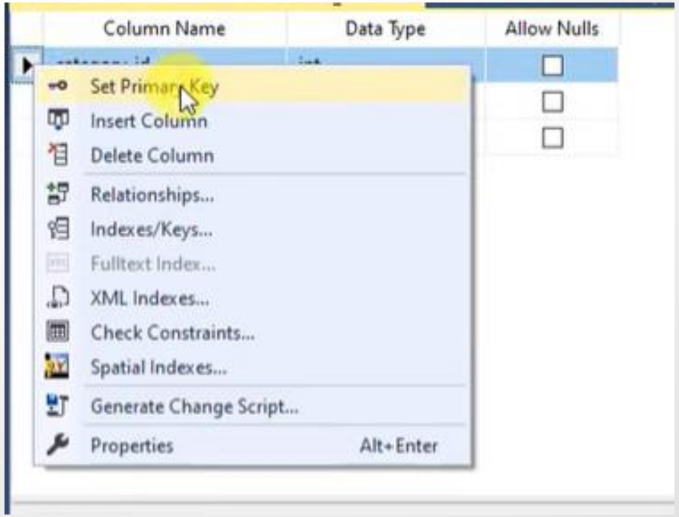
Create Database

Step4:



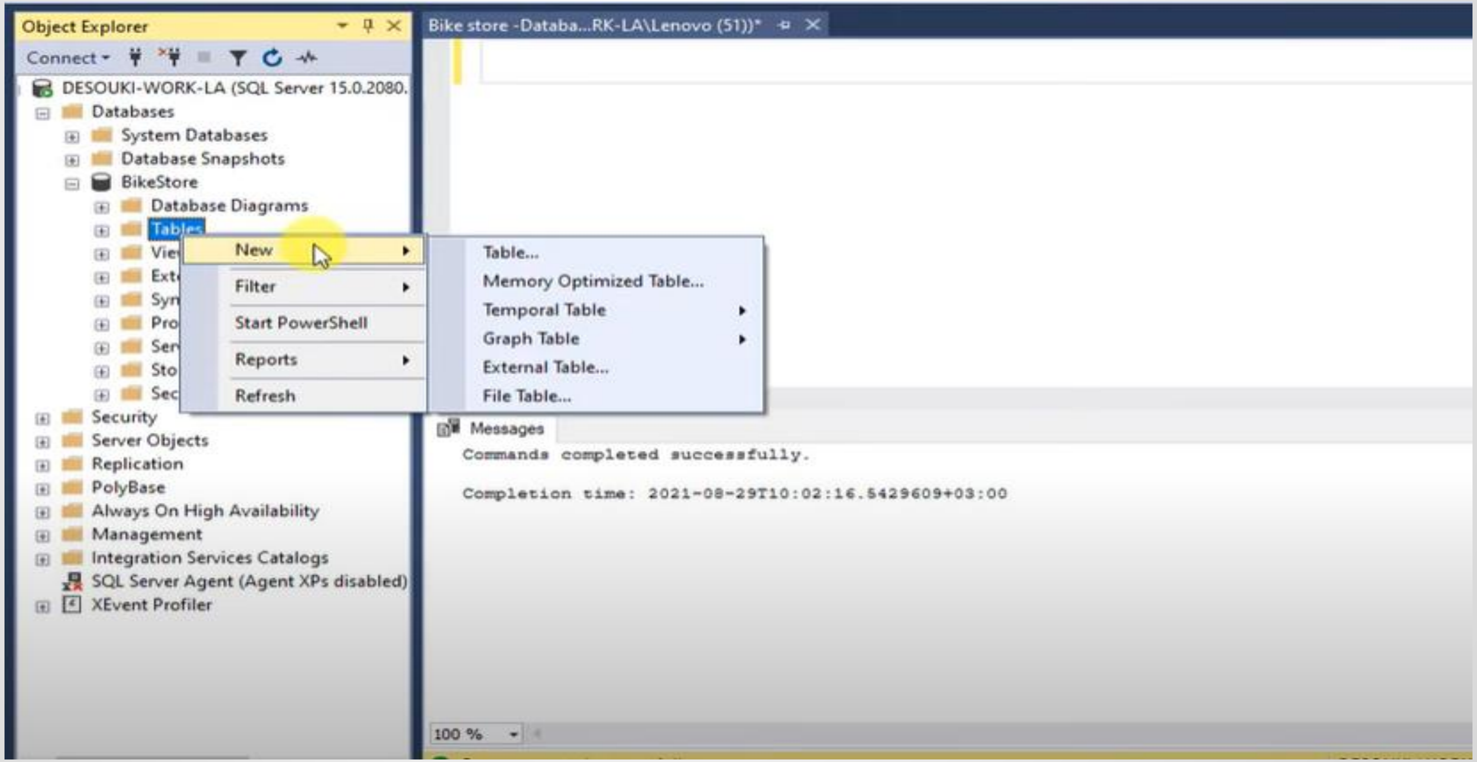
	Column Name	Data Type	Allow Nulls
	category_id	int	☐
▶	category_name	varchar(50)	☐
			☐

Step5:



Create Database

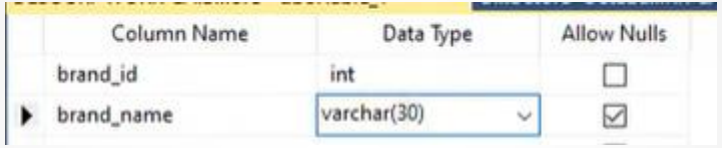
Step6:



```
Commands completed successfully.

Completion time: 2021-08-29T10:02:16.5429609+03:00
```

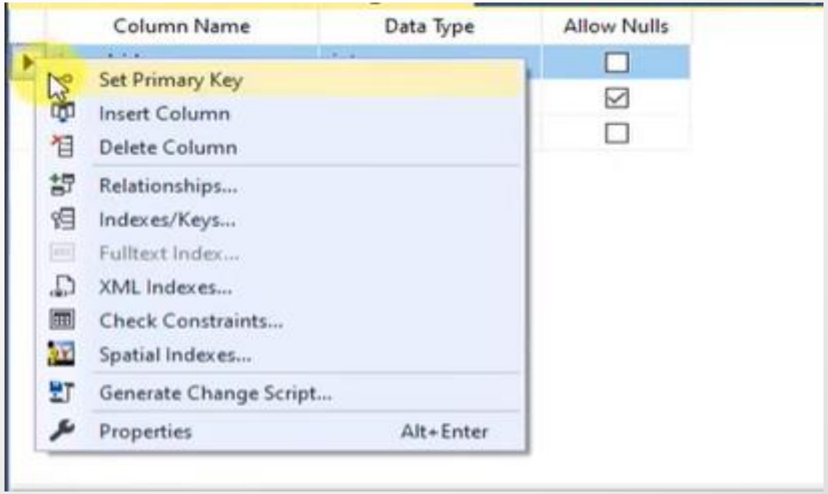
Create Database

Step7:



Column Name	Data Type	Allow Nulls
brand_id	int	☐
brand_name	varchar(30)	☑

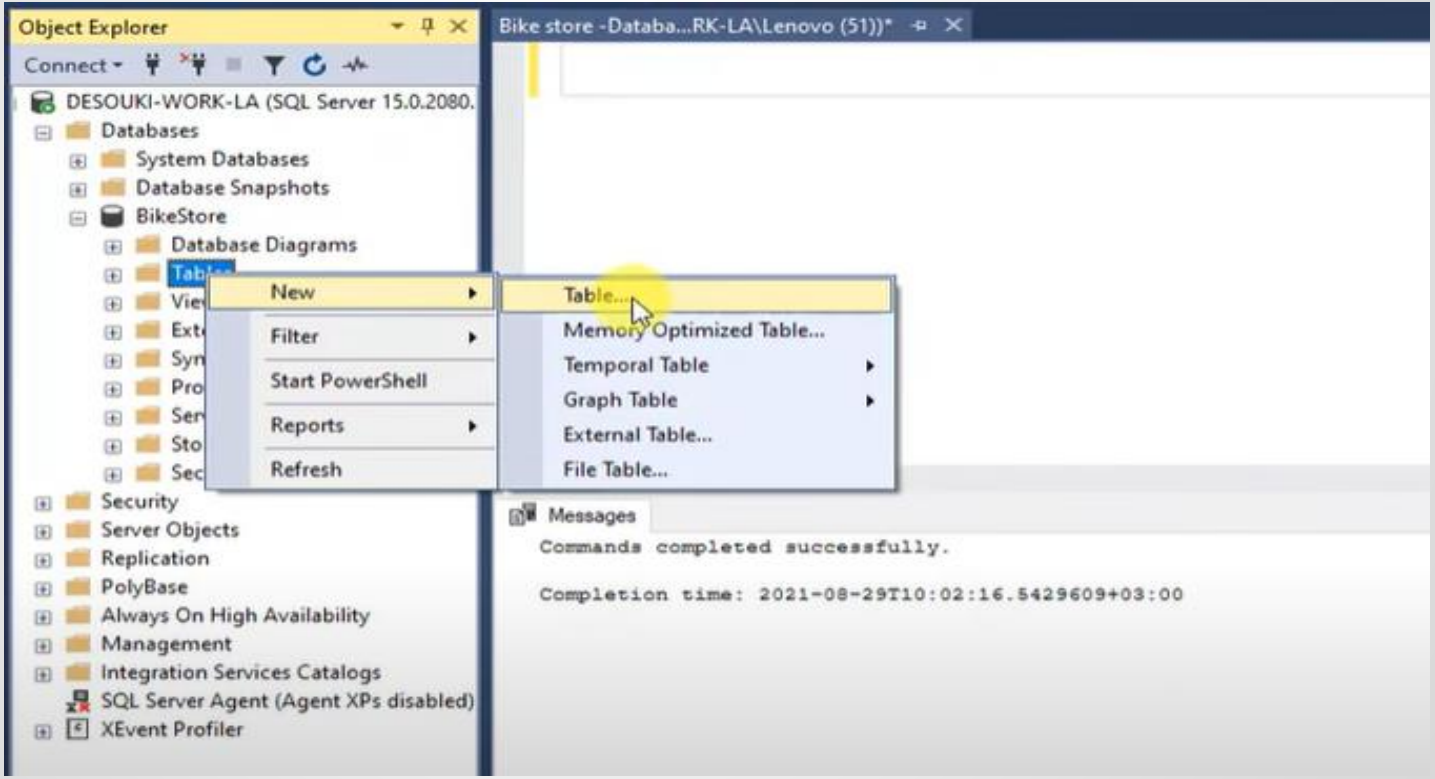
Step8:



Column Name	Data Type	Allow Nulls

Create Database

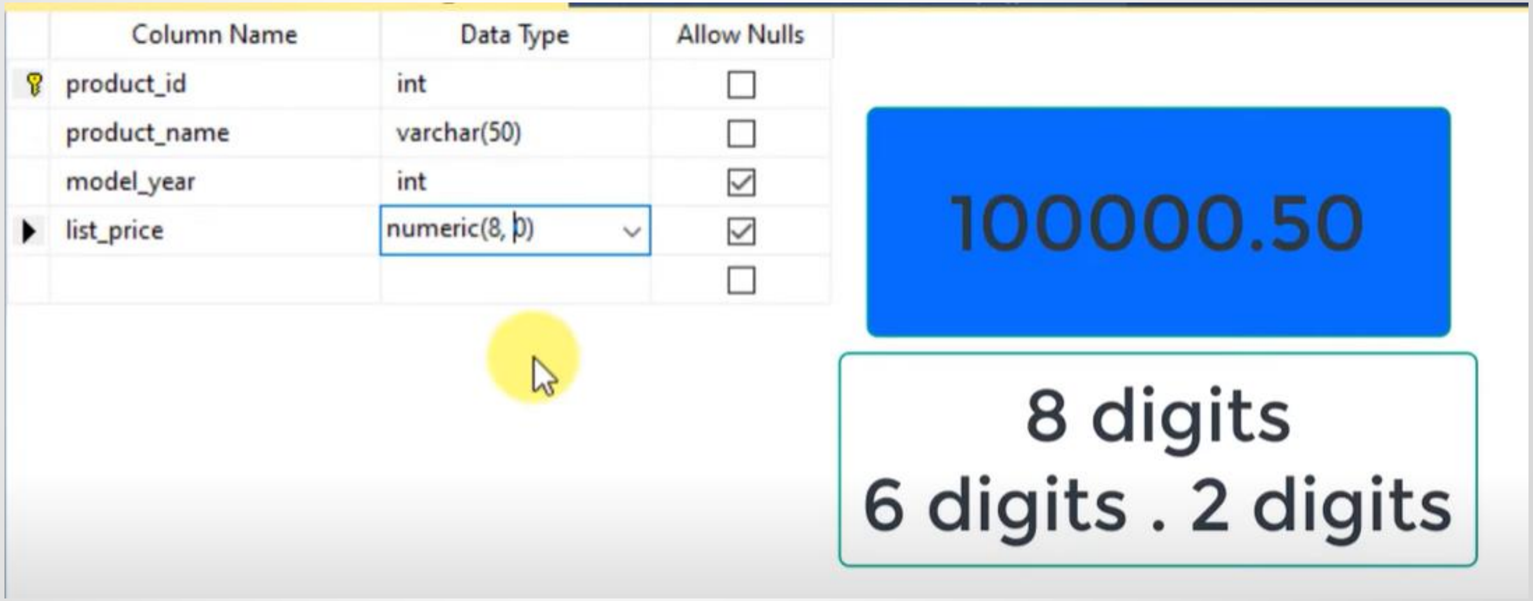
Step9:



```
Commands completed successfully.

Completion time: 2021-08-29T10:02:16.5429609+03:00
```

Create Database

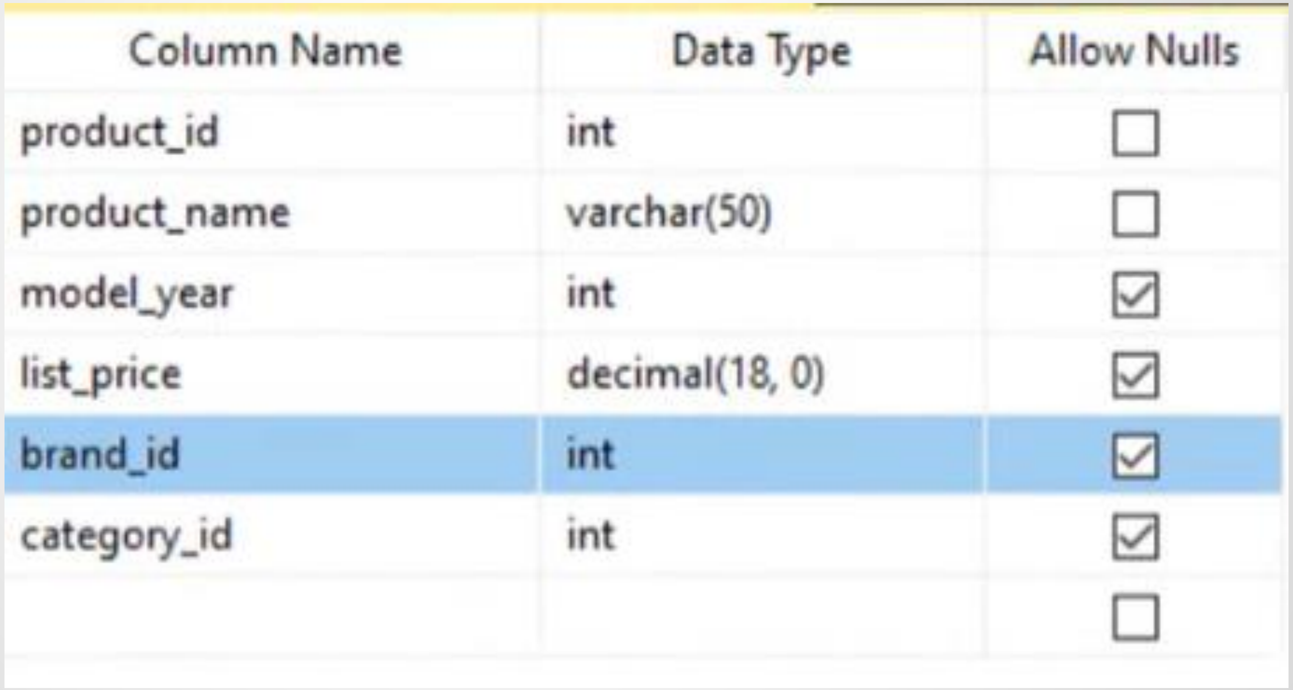
Step10:



	Column Name	Data Type	Allow Nulls
🔑	product_id	int	☐
	product_name	varchar(50)	☐
	model_year	int	✓
▶	list_price	numeric(8, 0)	✓
			☐

Create Database

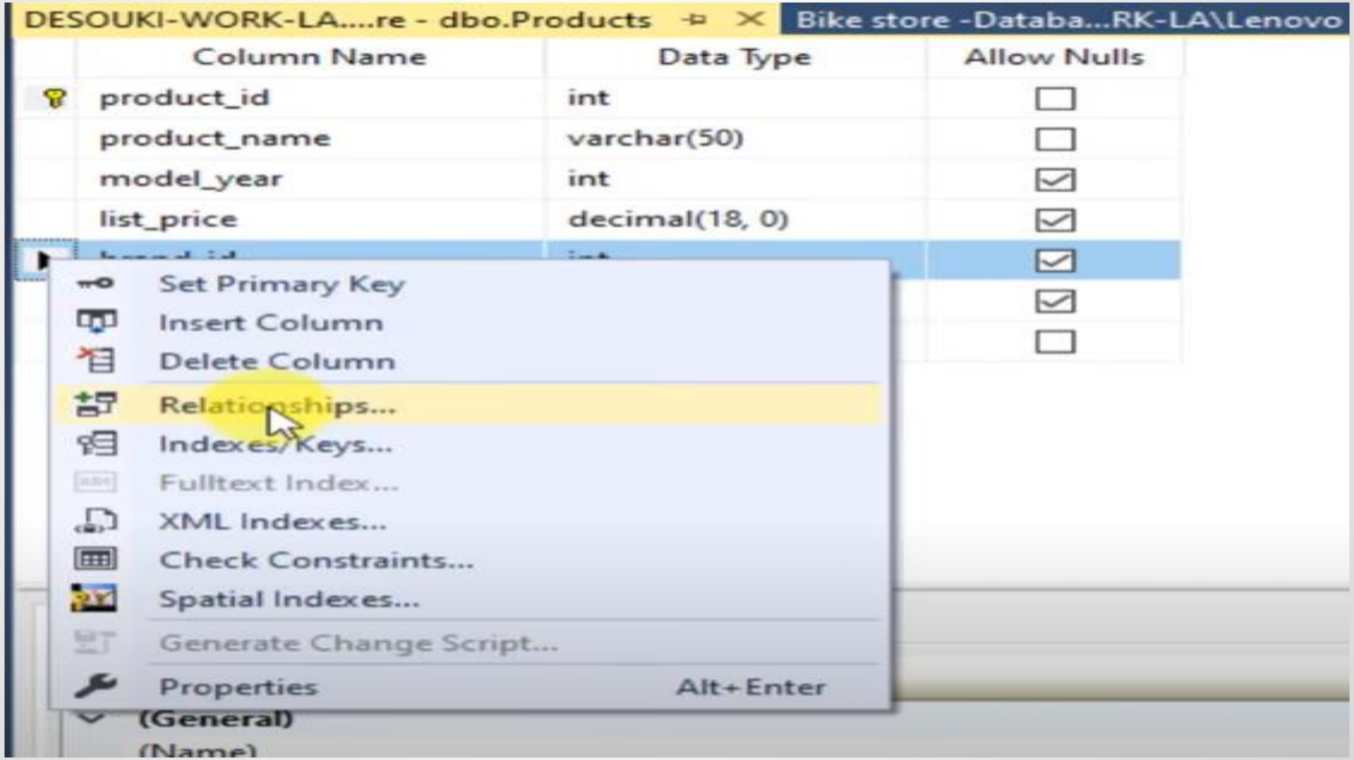
Step11:



Column Name	Data Type	Allow Nulls
product_id	int	☐
product_name	varchar(50)	☐
model_year	int	☑
list_price	decimal(18, 0)	☑
brand_id	int	☑
category_id	int	☑
		☐

Create Database

Step12:



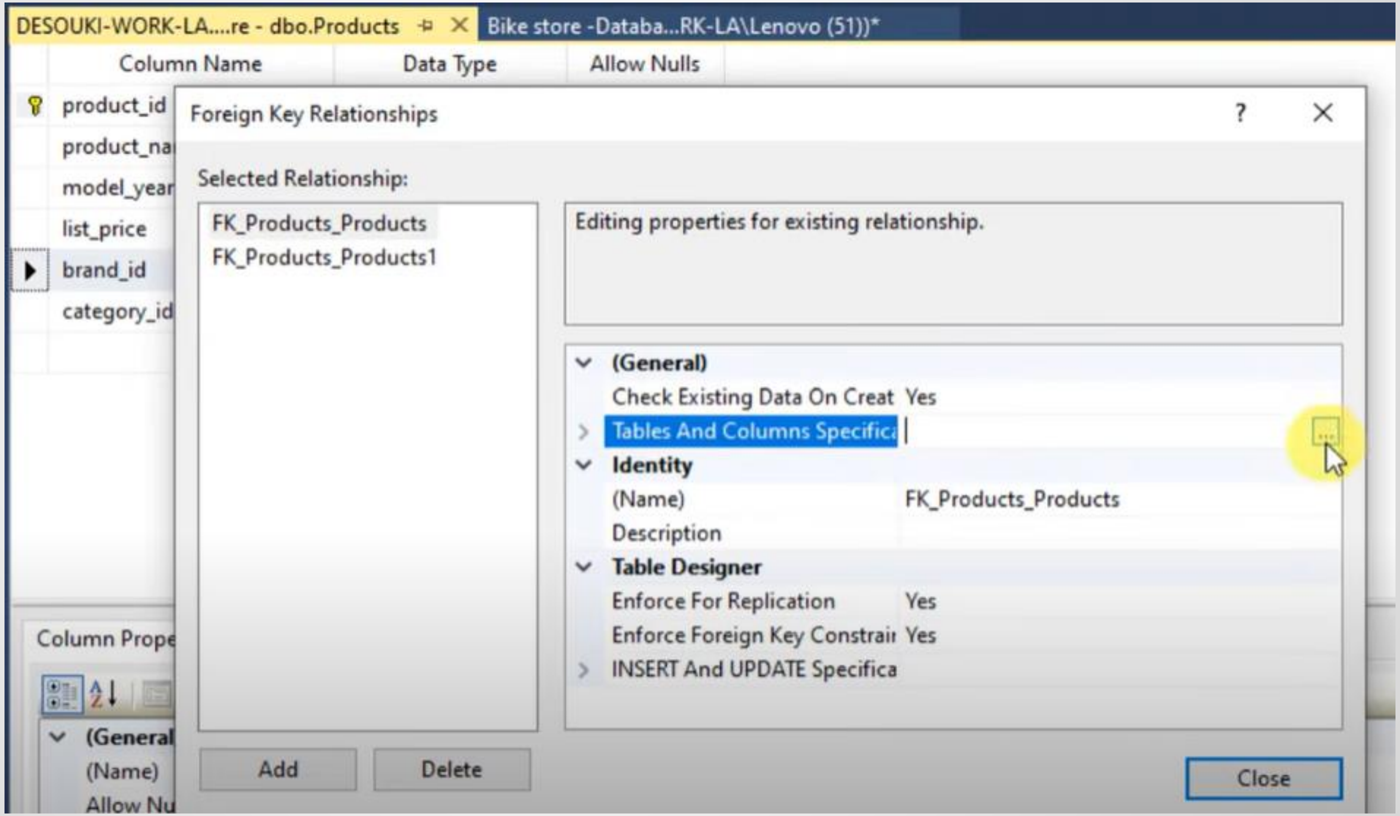
DESOUKI-WORK-LA....re - dbo.Products

Bike store -Databa...RK-LA\Lenovo

Column Name	Data Type	Allow Nulls
product_id	int	☐
product_name	varchar(50)	☐
model_year	int	☑
list_price	decimal(18, 0)	☑
brand_id	int	☑
		☑
		☐

Create Database

Step13:



DESOUKI-WORK-LA....re - dbo.Products | Bike store -Databa...RK-LA\Lenovo (51))*

Column Name	Data Type	Allow Nulls
product_id		
product_na		
model_year		
list_price		
brand_id		
category_id		

Foreign Key Relationships

Selected Relationship:

FK_Products_Products
FK_Products_Products1

Editing properties for existing relationship.

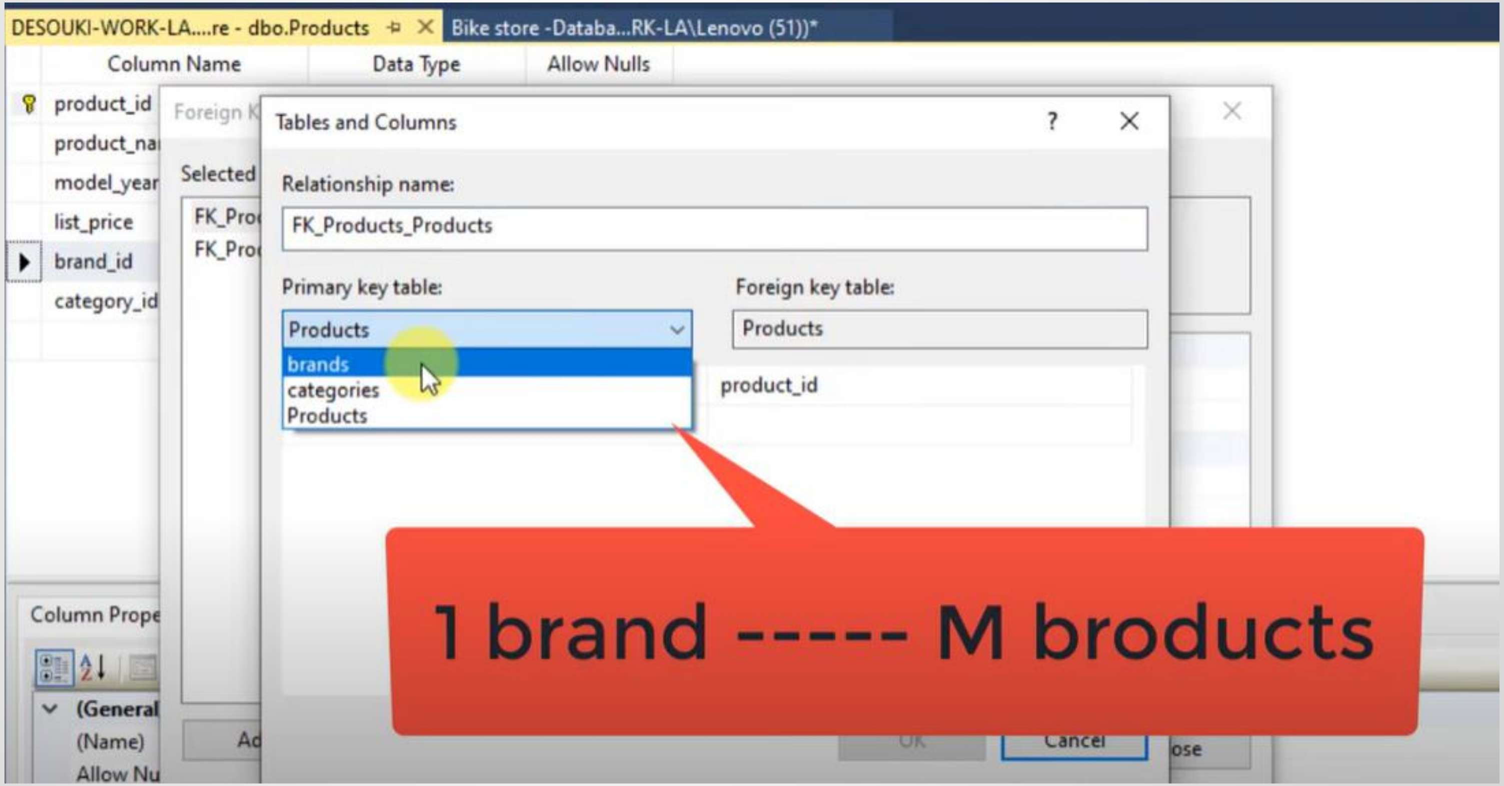
(General)
Check Existing Data On Creat Yes
Tables And Columns Specifica
Identity
(Name) FK_Products_Products
Description
Table Designer
Enforce For Replication Yes
Enforce Foreign Key Constrai Yes
INSERT And UPDATE Specifica

Add | Delete | Close

Column Prope
(General
(Name)
Allow Nu

Create Database

Step14:



DESOUKI-WORK-LA....re - dbo.Products

Bike store -Databa...RK-LA\Lenovo (51))*

Column Name	Data Type	Allow Nulls
product_id		
product_na		
model_year		
list_price		
brand_id		
category_id		

Tables and Columns

Relationship name: FK_Products_Products

Primary key table: Products

Foreign key table: Products

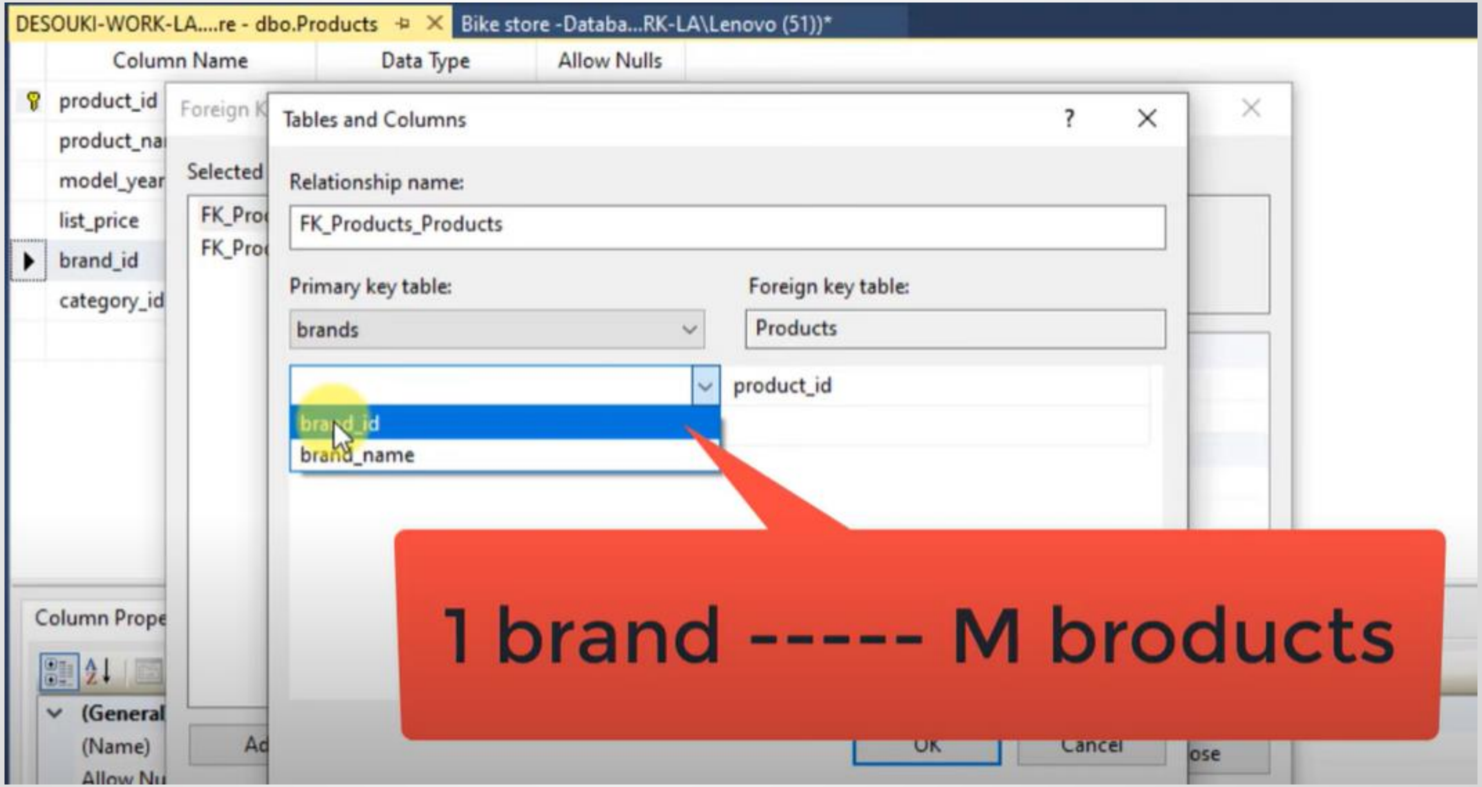
brands
categories
Products

product_id

1 brand ----- M broducts

Create Database

Step15:



DESOUKI-WORK-LA....re - dbo.Products

Bike store -Databa...RK-LA\Lenovo (51))*

Column Name	Data Type	Allow Nulls
product_id		
product_na		
model_year		
list_price		
brand_id		
category_id		

Tables and Columns

Relationship name: FK_Products_Products

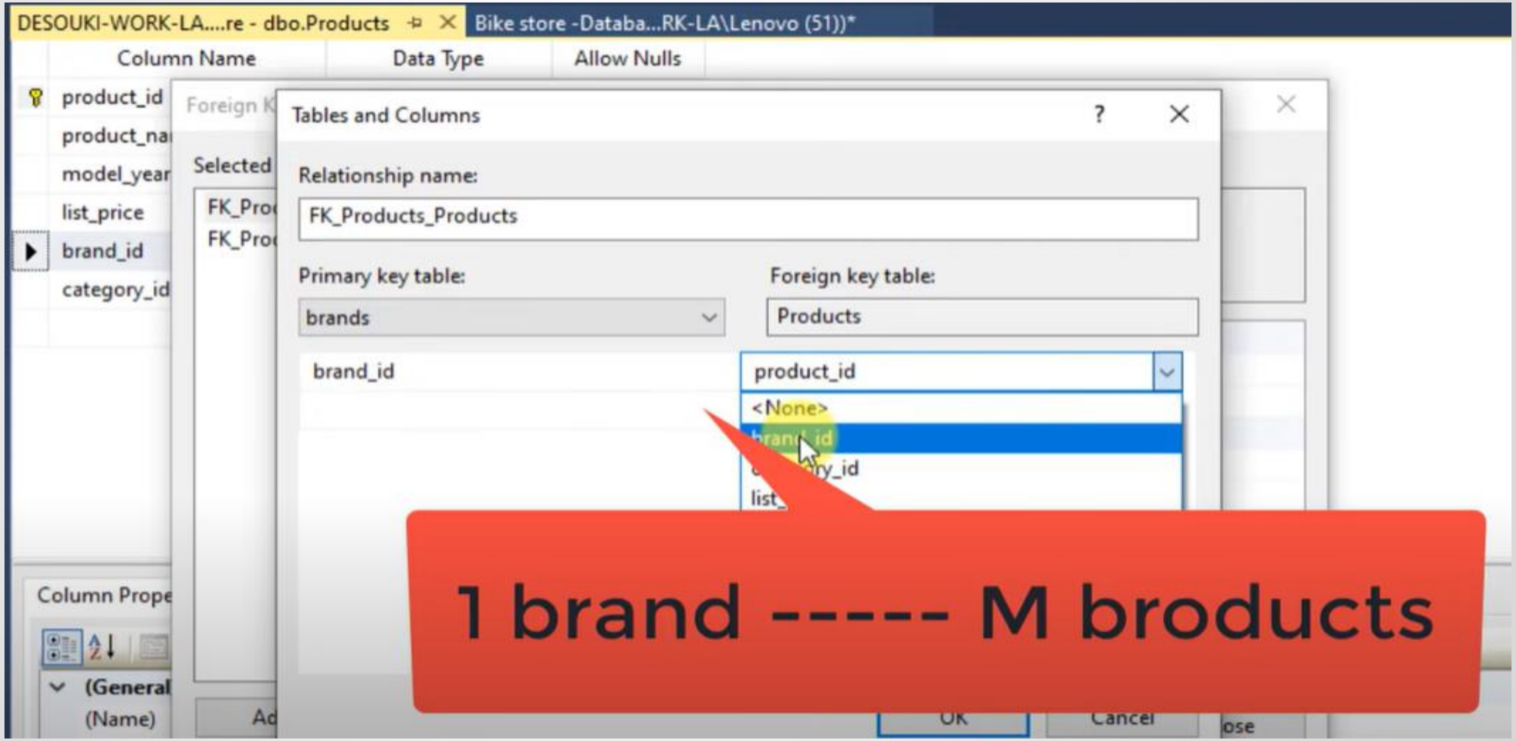
Primary key table: brands

Foreign key table: Products

brand_id → product_id

Create Database

Step16:



DESOUKI-WORK-LA....re - dbo.Products | Bike store -Databa...RK-LA\Lenovo (51))*

Column Name	Data Type	Allow Nulls
product_id		
product_na		
model_year		
list_price		
brand_id		
category_id		

Tables and Columns

Relationship name: FK_Products_Products

Primary key table: brands

Foreign key table: Products

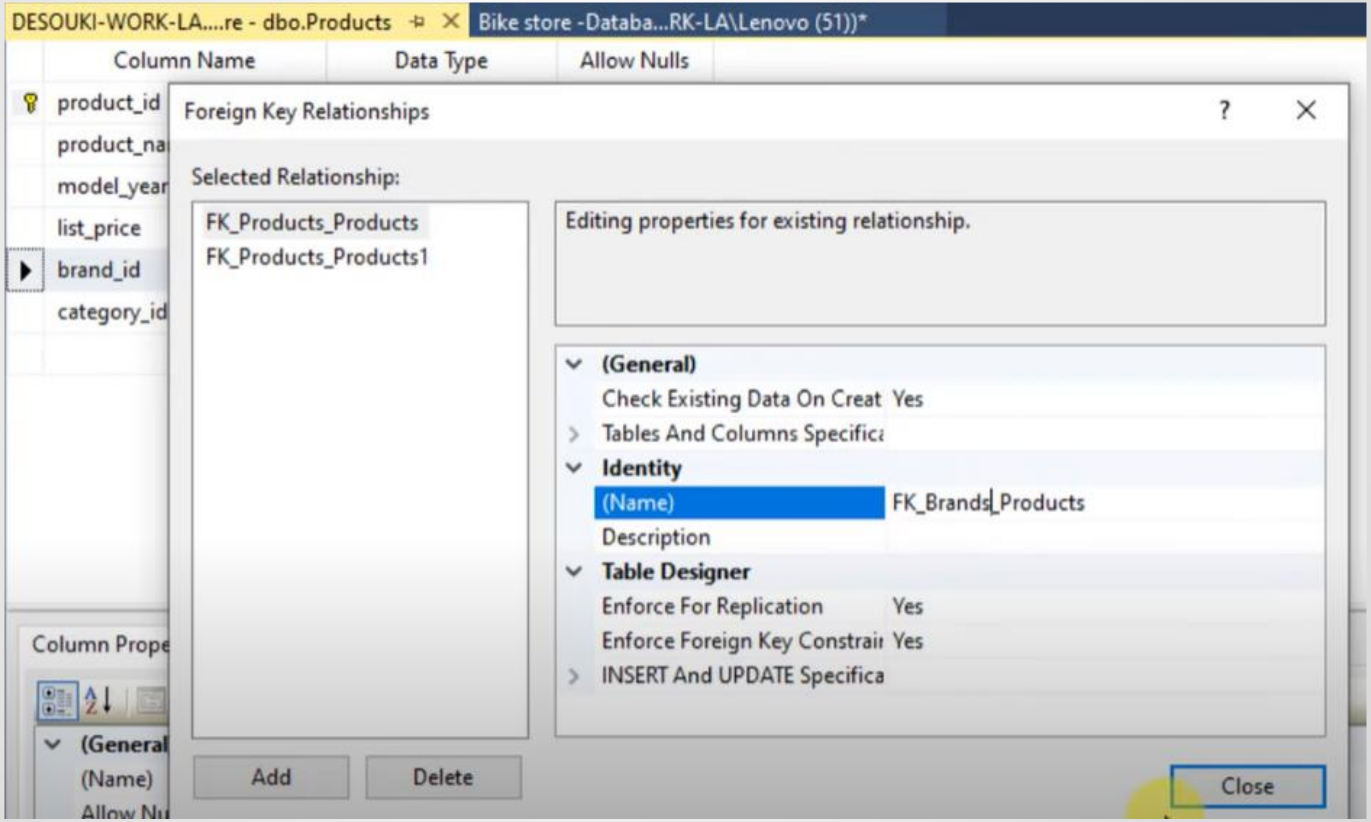
brand_id — product_id

<None>
brand_id
category_id
list_

1 brand ----- M broducts

Create Database

Step17:



DESOUKI-WORK-LA....re - dbo.Products | Bike store -Databa...RK-LA\Lenovo (51))*

Column Name	Data Type	Allow Nulls
product_id		
product_na		
model_year		
list_price		
brand_id		
category_id		

Foreign Key Relationships

Selected Relationship:

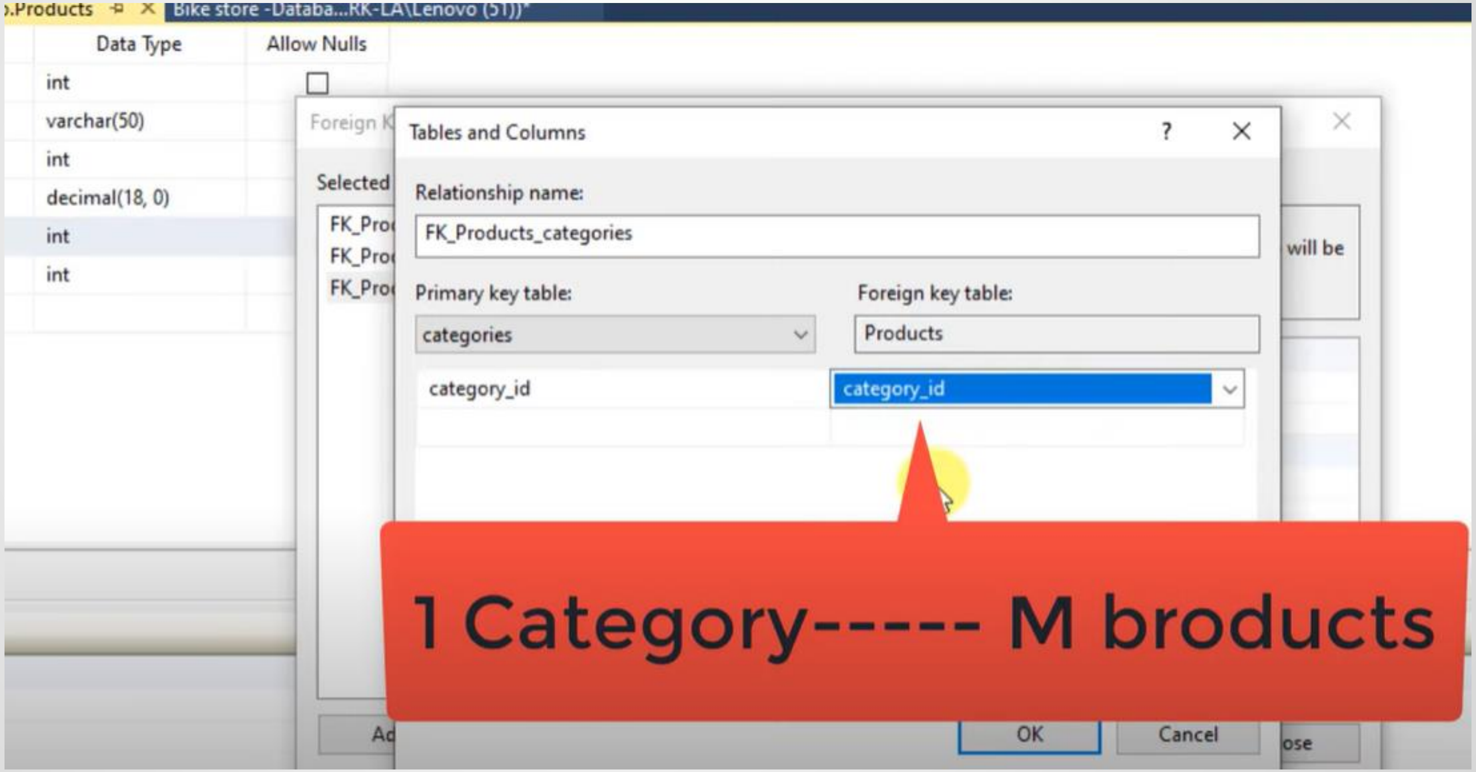
- FK_Products_Products
- FK_Products_Products1

Editing properties for existing relationship.

(General)	
Check Existing Data On Creat	Yes
Tables And Columns Specifica	
Identity	
(Name)	FK_Brands_Products
Description	
Table Designer	
Enforce For Replication	Yes
Enforce Foreign Key Constrai	Yes
INSERT And UPDATE Specifica	

Create Database

Step18:



.Products

Bike store -Databa...RK-LA\Lenovo (51))*

Data Type	Allow Nulls
int	☐
varchar(50)	
int	
decimal(18, 0)	
int	
int	

Foreign K

Selected

FK_Proc

FK_Proc

FK_Proc

will be

Tables and Columns

Relationship name:

FK_Products_categories

Primary key table: categories

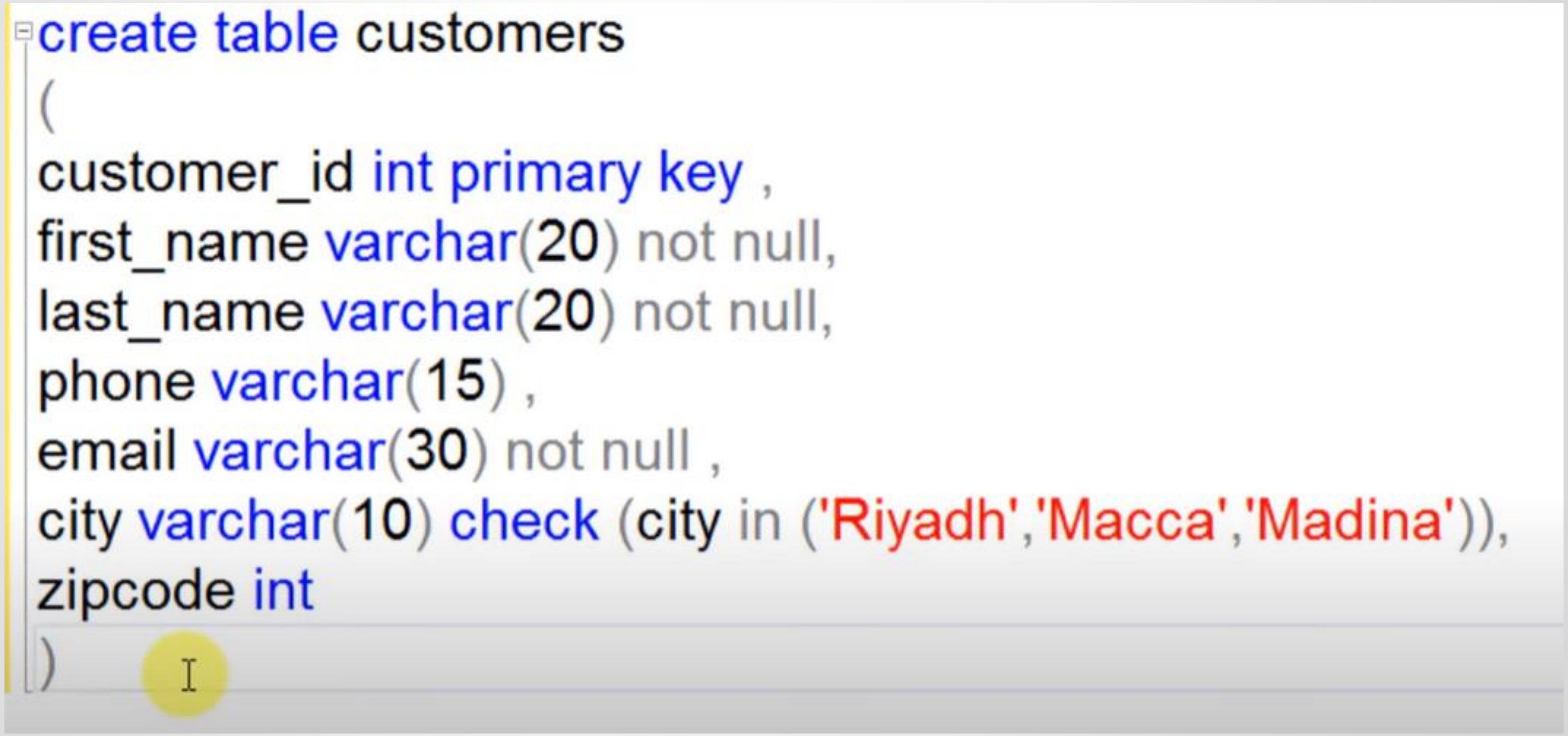
Foreign key table: Products

Primary key table	Foreign key table
category_id	category_id

1 Category----- M broducts

Create Database(Command SQL)

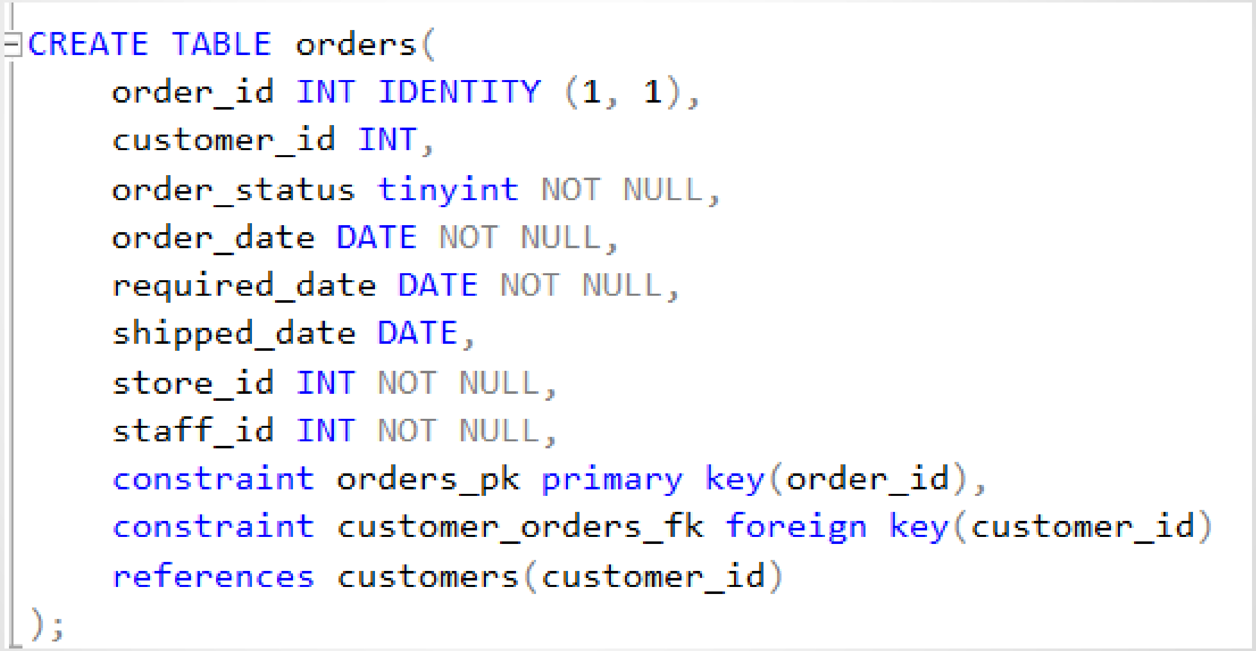
Step1:



```sql
create table customers
(
customer_id int primary key ,
first_name varchar(20) not null,
last_name varchar(20) not null,
phone varchar(15) ,
email varchar(30) not null ,
city varchar(10) check (city in ('Riyadh','Macca','Madina')),
zipcode int
)
```

Create Database(Command SQL)

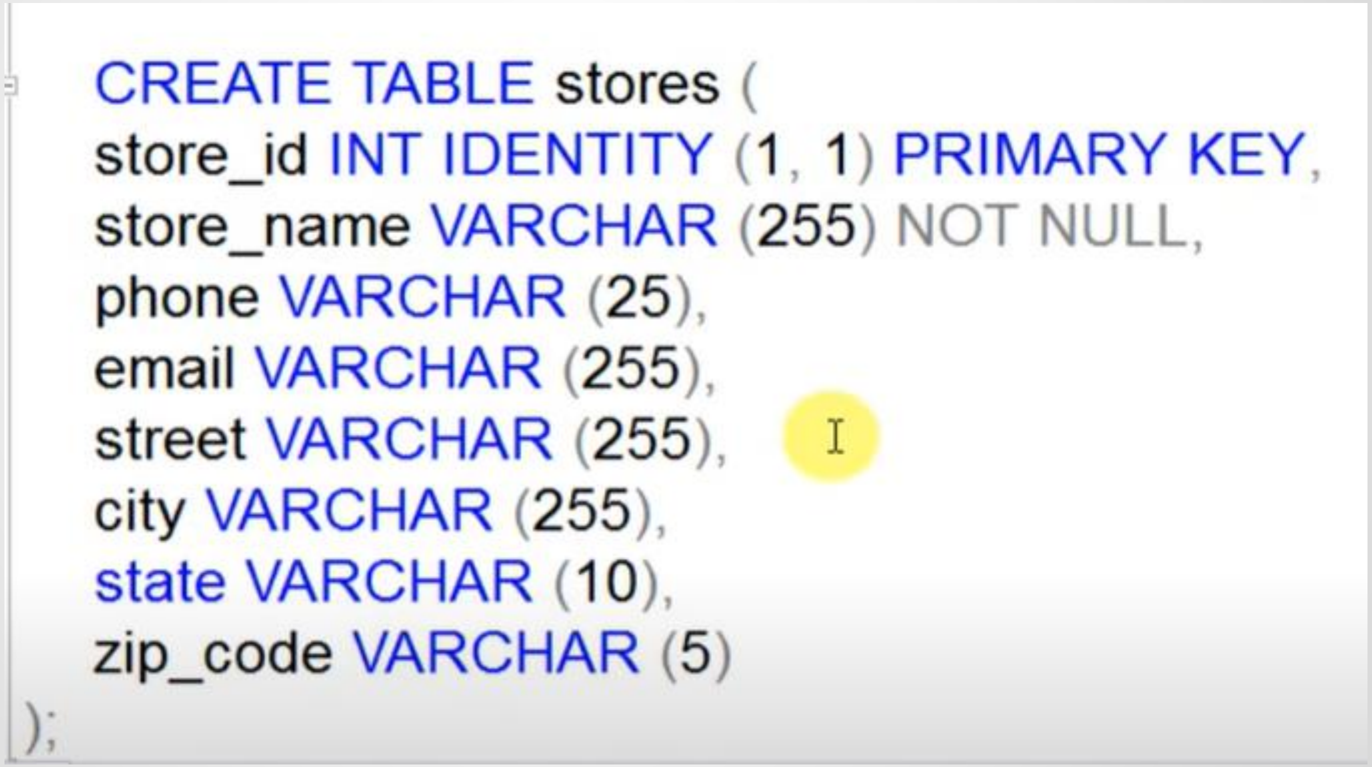
Step2:



```
CREATE TABLE orders(
    order_id INT IDENTITY (1, 1),
    customer_id INT,
    order_status tinyint NOT NULL,
    order_date DATE NOT NULL,
    required_date DATE NOT NULL,
    shipped_date DATE,
    store_id INT NOT NULL,
    staff_id INT NOT NULL,
    constraint orders_pk primary key(order_id),
    constraint customer_orders_fk foreign key(customer_id)
    references customers(customer_id)
);
```

Create Database(Command SQL)

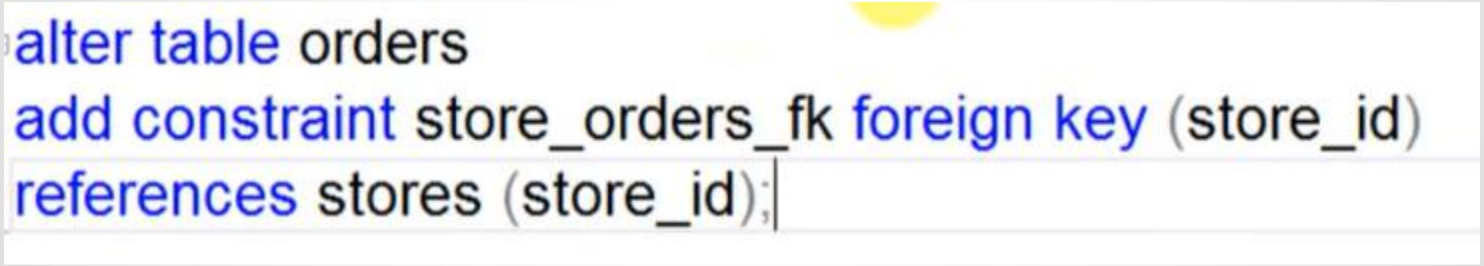
Step3:



```
CREATE TABLE stores (
store_id INT IDENTITY (1, 1) PRIMARY KEY,
store_name VARCHAR (255) NOT NULL,
phone VARCHAR (25),
email VARCHAR (255),
street VARCHAR (255),
city VARCHAR (255),
state VARCHAR (10),
zip_code VARCHAR (5)
);
```

Create Database(Command SQL)

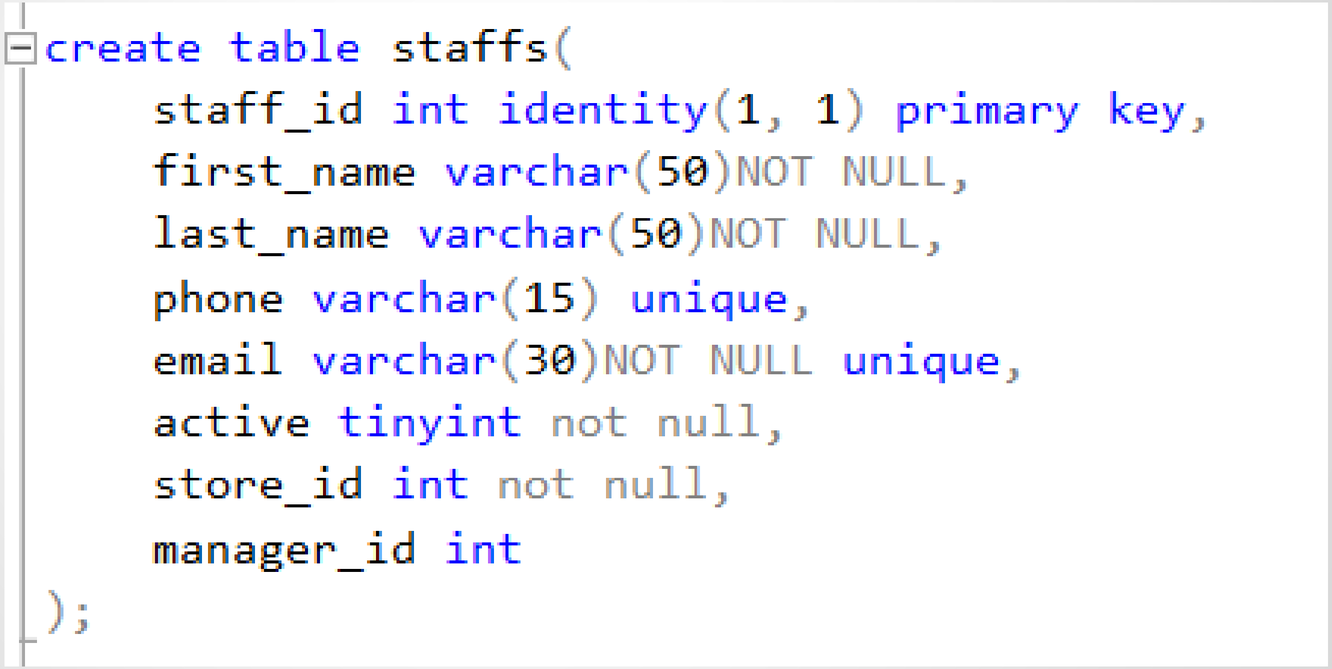
Step4:
Connect order with stores



```
alter table orders
add constraint store_orders_fk foreign key (store_id)
references stores (store_id);
```

Create Database(Command SQL)

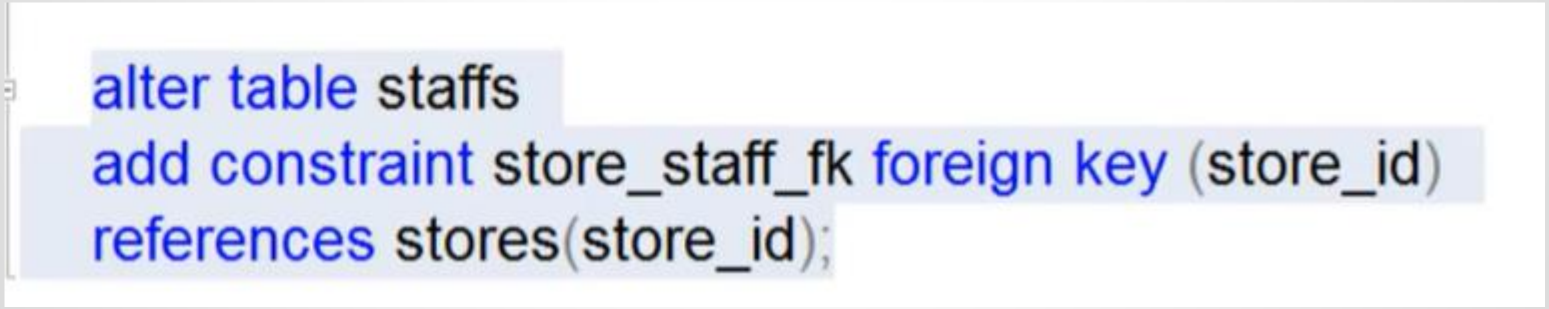
Step5:



```
create table staffs(
    staff_id int identity(1, 1) primary key,
    first_name varchar(50)NOT NULL,
    last_name varchar(50)NOT NULL,
    phone varchar(15) unique,
    email varchar(30)NOT NULL unique,
    active tinyint not null,
    store_id int not null,
    manager_id int
);
```

Create Database(Command SQL)

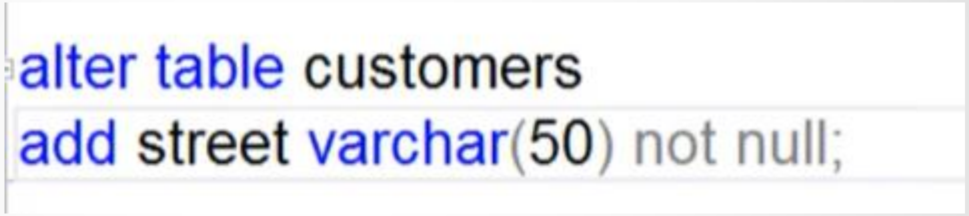
Step6:
Connect staff with store



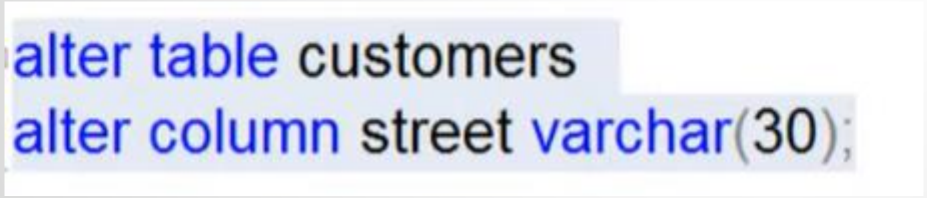
```
alter table staffs
add constraint store_staff_fk foreign key (store_id)
references stores(store_id);
```

Create Database(Command SQL)

Step7:



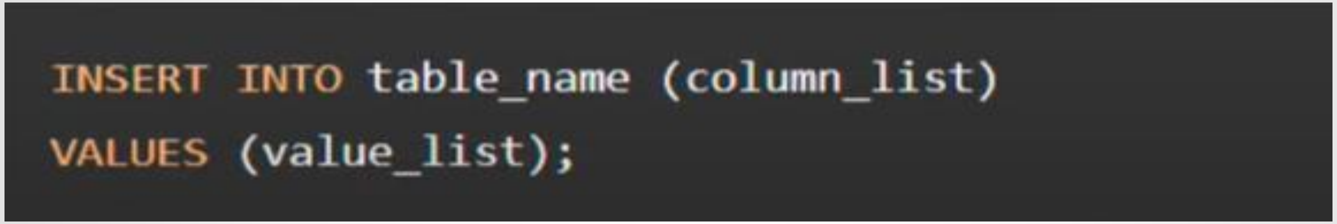
```
alter table customers
add street varchar(50) not null;
```



```
alter table customers
alter column street varchar(30);
```

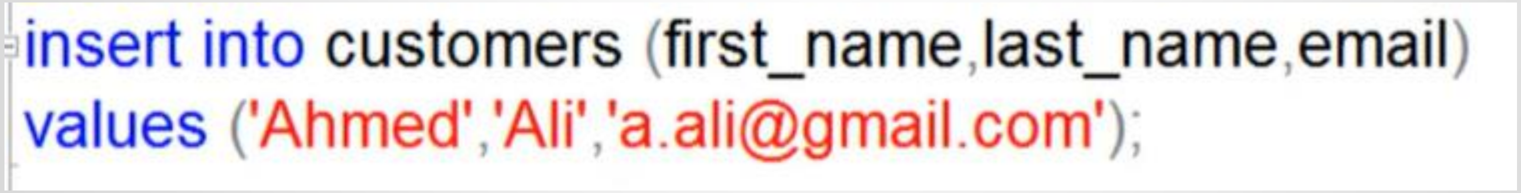
INSERT INTO STATEMENT

- To add one or more rows into a table



```
INSERT INTO table_name (column_list)
VALUES (value_list);
```

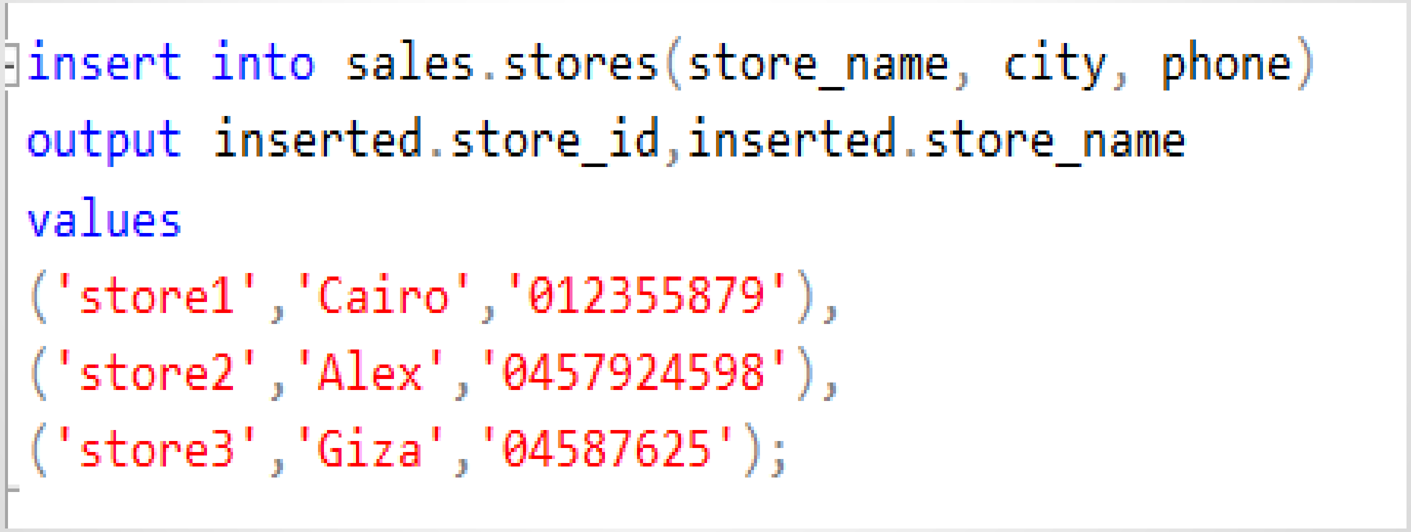
Example : Add one row



```
insert into customers (first_name,last_name,email)
values ('Ahmed','Ali','a.ali@gmail.com');
```

INSERT INTO STATEMENT

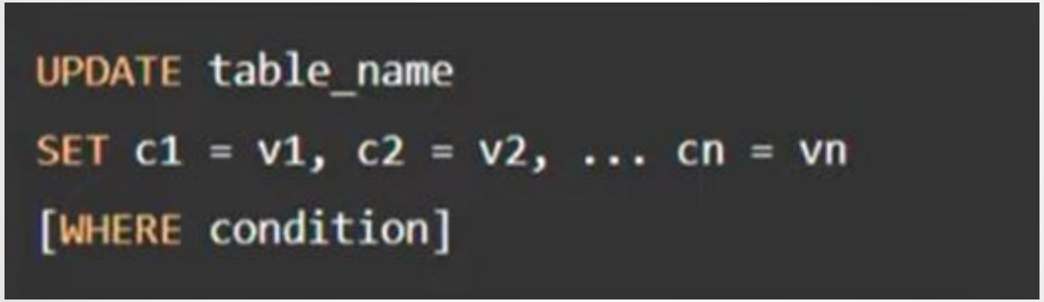
Example : Add multiple rows



```
insert into sales.stores(store_name, city, phone)
output inserted.store_id,inserted.store_name
values
('store1','Cairo','012355879'),
('store2','Alex','0457924598'),
('store3','Giza','04587625');
```

UPDATE STATEMENT

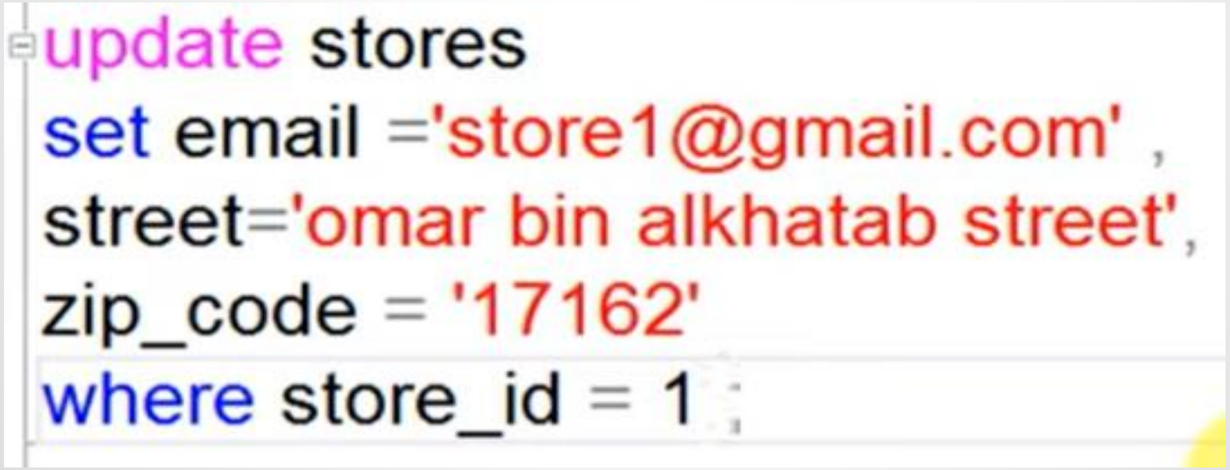
- To modify existing data in a table



```
UPDATE table_name
SET c1 = v1, c2 = v2, ... cn = vn
[WHERE condition]
```

UPDATE STATEMENT

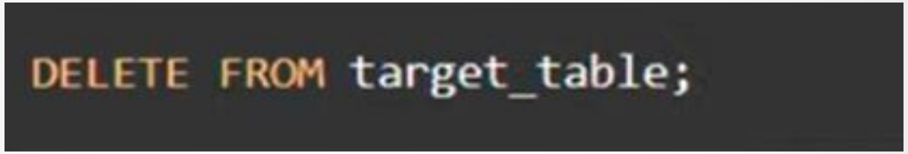
- **To modify multiple data in a table**



```
update stores
set email ='store1@gmail.com' ,
street='omar bin alkhatab street',
zip_code = '17162'
where store_id = 1 ;
```

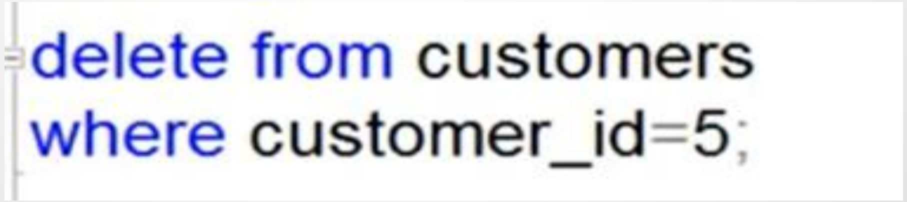
DELETE STATEMENT

- To delete all rows in a table



```
DELETE FROM target_table;
```

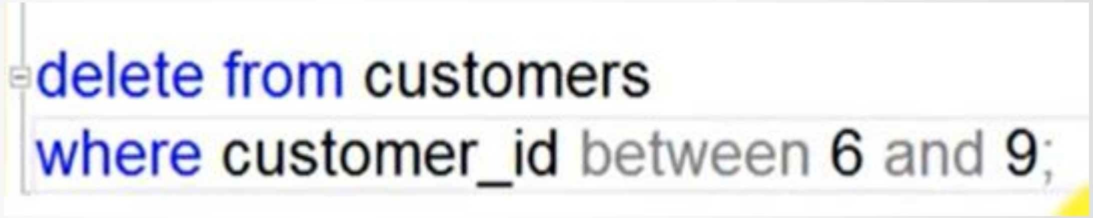
- To delete one row in a table



```
delete from customers
where customer_id=5;
```

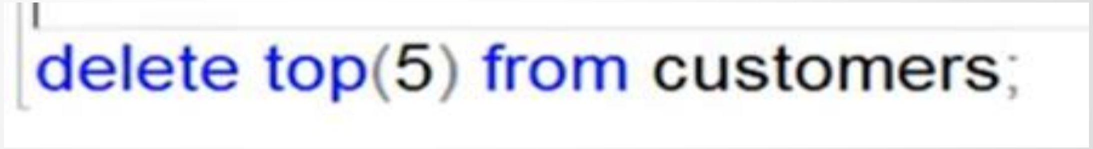
DELETE STATEMENT

- To delete multiple row in a table



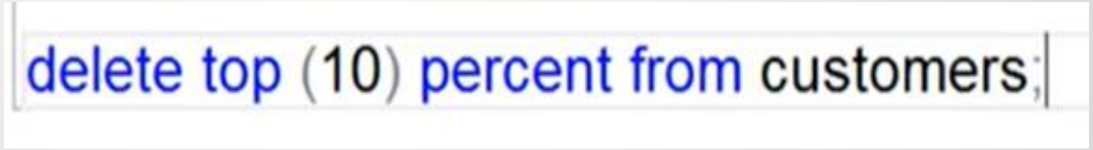
```
delete from customers
where customer_id between 6 and 9;
```

- To delete top rows in a table



```
delete top(5) from customers;
```

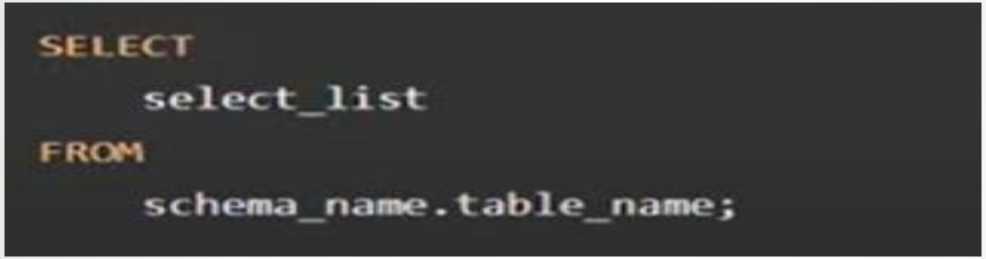
- To delete top percent rows in a table



```
delete top (10) percent from customers;
```

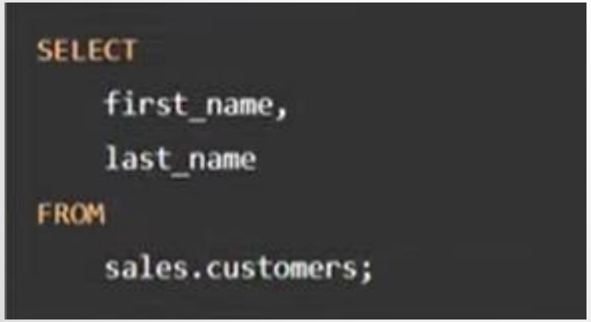
SELECT STATEMENT

- To query data from a table

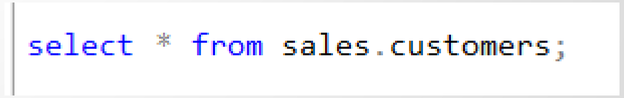


```
SELECT
    select_list
FROM
    schema_name.table_name;
```

- Example:



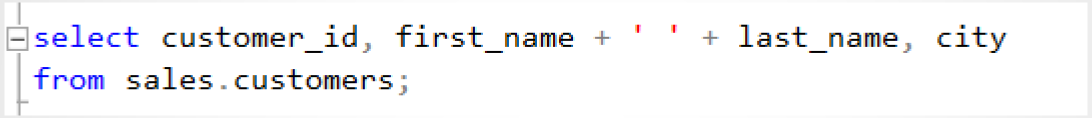
```
SELECT
    first_name,
    last_name
FROM
    sales.customers;
```



```
select * from sales.customers;
```

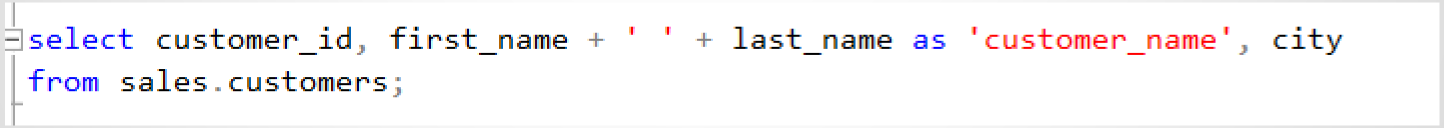
SELECT STATEMENT

- To concatenate two columns in a table



```sql
select customer_id, first_name + ' ' + last_name, city
from sales.customers;
```

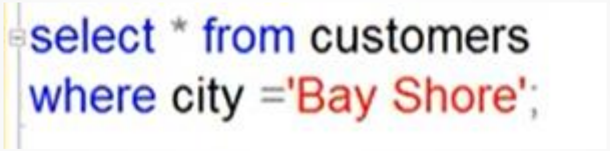
- To give it Elise name



```sql
select customer_id, first_name + ' ' + last_name as 'customer_name', city
from sales.customers;
```

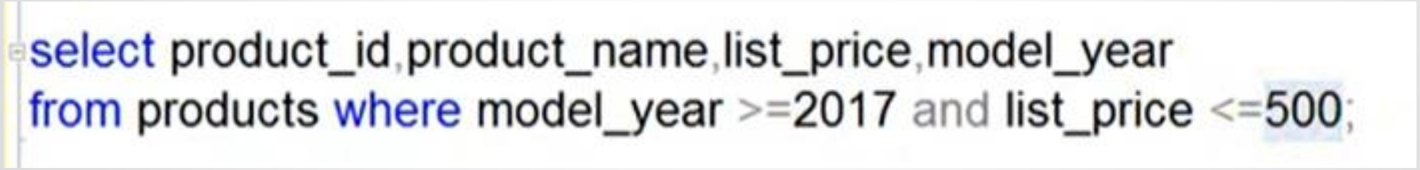
SELECT STATEMENT

- With condition



```
select * from customers
where city ='Bay Shore';
```

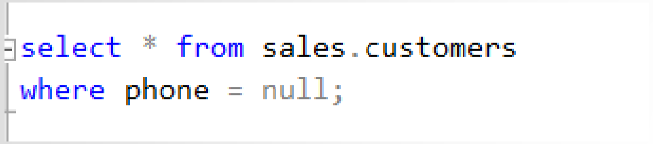
- With 2 conditions



```
select product_id,product_name,list_price,model_year
from products where model_year >=2017 and list_price <=500;
```

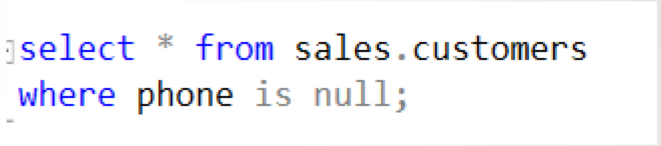
SELECT STATEMENT

- Select NULL values



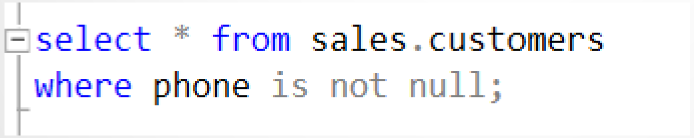
```sql
select * from sales.customers
where phone = null;
```

Wrong statement



```sql
select * from sales.customers
where phone is null;
```

Right statement

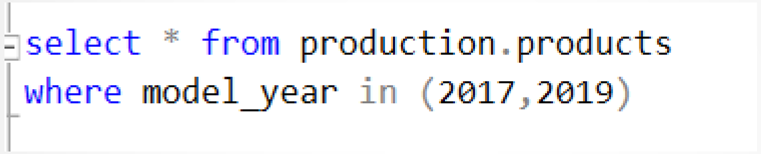


```sql
select * from sales.customers
where phone is not null;
```

Negative statement

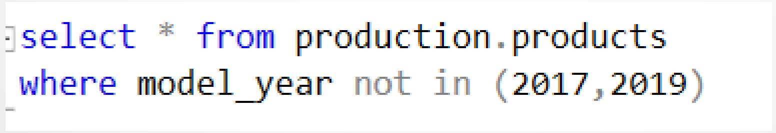
SELECT STATEMENT

- **Using IN Condition**



```
select * from production.products
where model_year in (2017,2019)
```

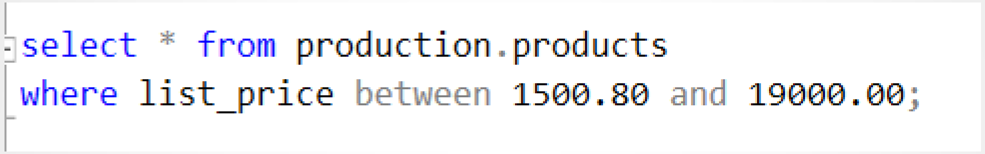
Negative statement



```
select * from production.products
where model_year not in (2017,2019)
```

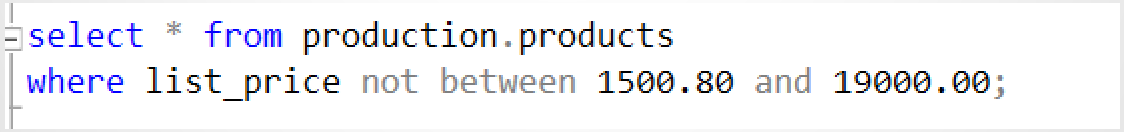
SELECT STATEMENT

- **Using Between Condition**



```
select * from production.products
where list_price between 1500.80 and 19000.00;
```

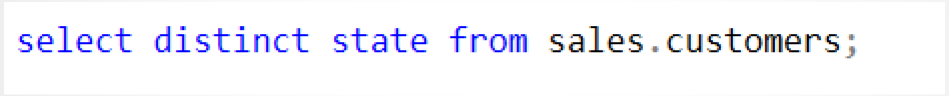
Negative statement



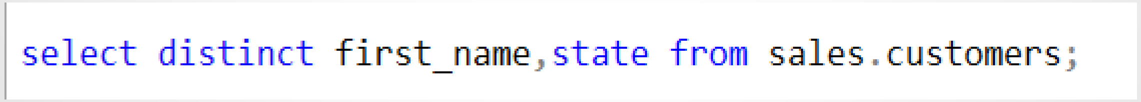
```
select * from production.products
where list_price not between 1500.80 and 19000.00;
```

SELECT STATEMENT

- **Using Distinct Condition**



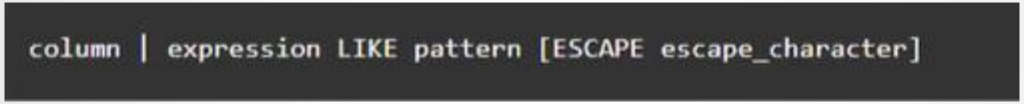
```
select distinct state from sales.customers;
```



```
select distinct first_name,state from sales.customers;
```

SELECT STATEMENT

- Using Like operator



```
column | expression LIKE pattern [ESCAPE escape_character]
```

Pattern

- Is a sequence of characters to search for in the column or expression.

SELECT STATEMENT

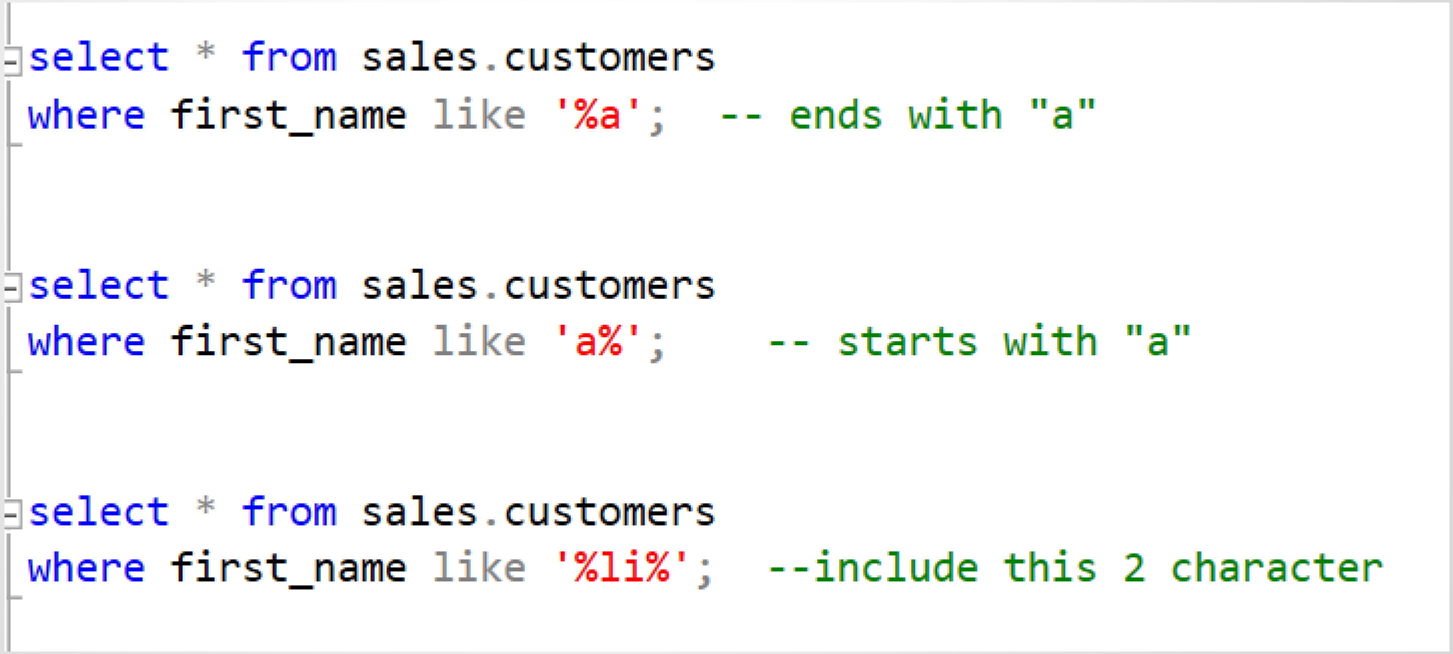
- **Using Like operator**

It can include the following valid wildcard characters:

- The percent wildcard (%): any string of zero or more characters.
- The underscore (_) wildcard: any single character.
- The [list of characters] wildcard: any single character within the specified set.
- The [character-character]: any single character within the specified range.
- The [^]: any single character not within a list or a range.

SELECT STATEMENT

- **Using Like operator (%)**
- **Example:**

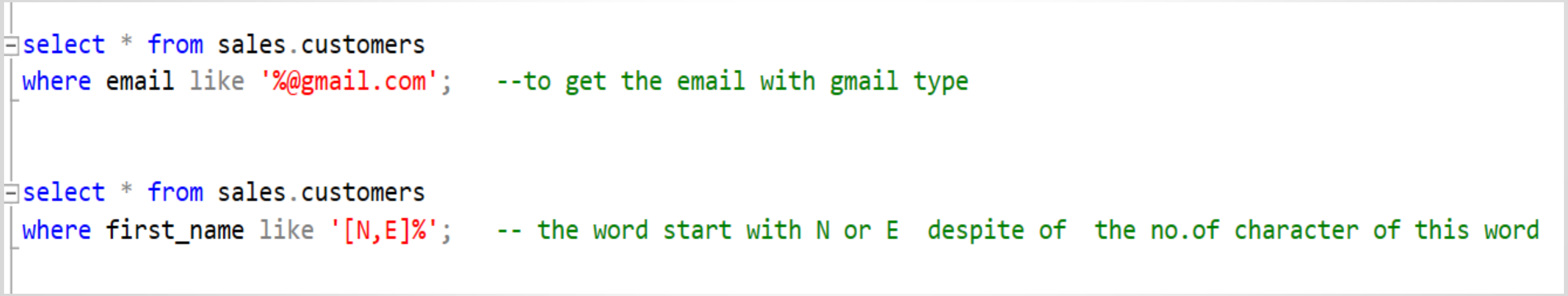


```sql
select * from sales.customers
where first_name like '%a';  -- ends with "a"


select * from sales.customers
where first_name like 'a%';    -- starts with "a"


select * from sales.customers
where first_name like '%li%';  --include this 2 character
```

SELECT STATEMENT

- **Using Like operator (%)**

- **Example:**

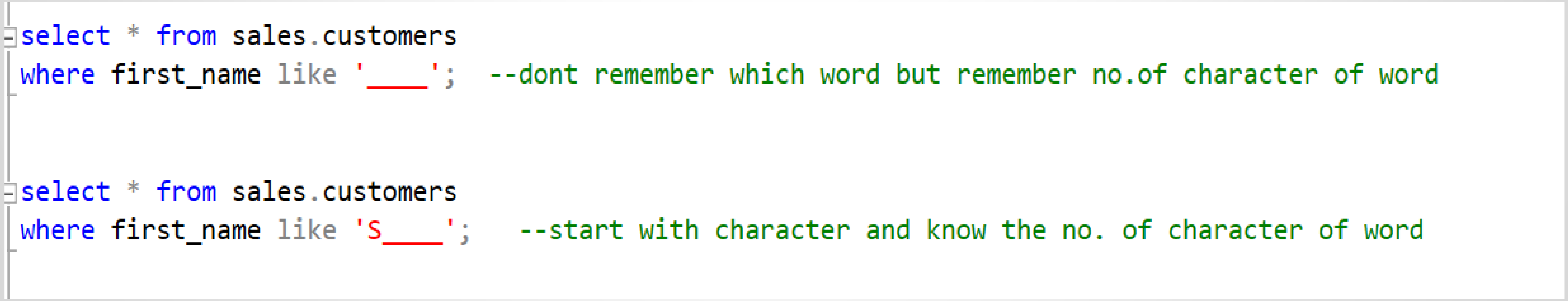


```
select * from sales.customers
where email like '%@gmail.com';   --to get the email with gmail type


select * from sales.customers
where first_name like '[N,E]%';   -- the word start with N or E  despite of  the no.of character of this word
```

SELECT STATEMENT

- **Using Like operator (_)**

- **Example:**

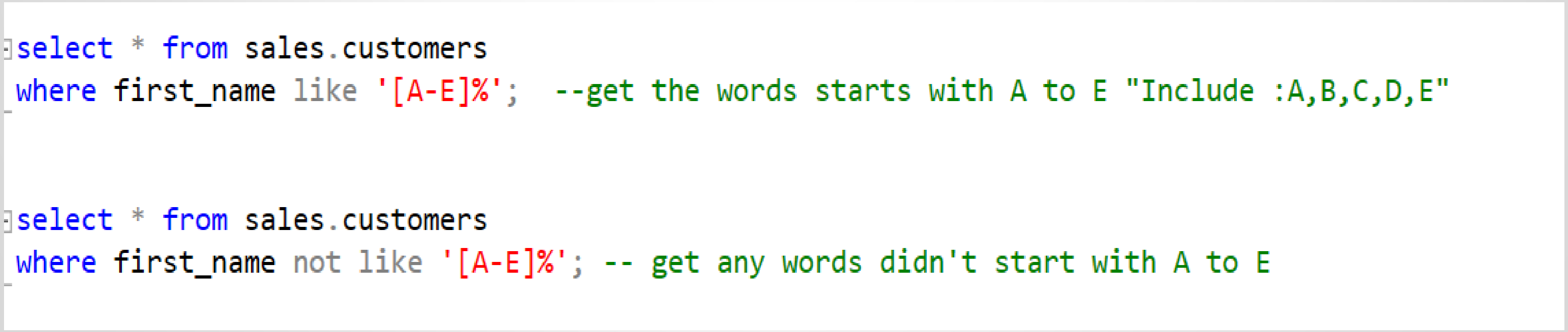


```
select * from sales.customers
where first_name like '____';  --dont remember which word but remember no.of character of word


select * from sales.customers
where first_name like 'S____';   --start with character and know the no. of character of word
```

SELECT STATEMENT

- Using Like operator (-)

- Example:

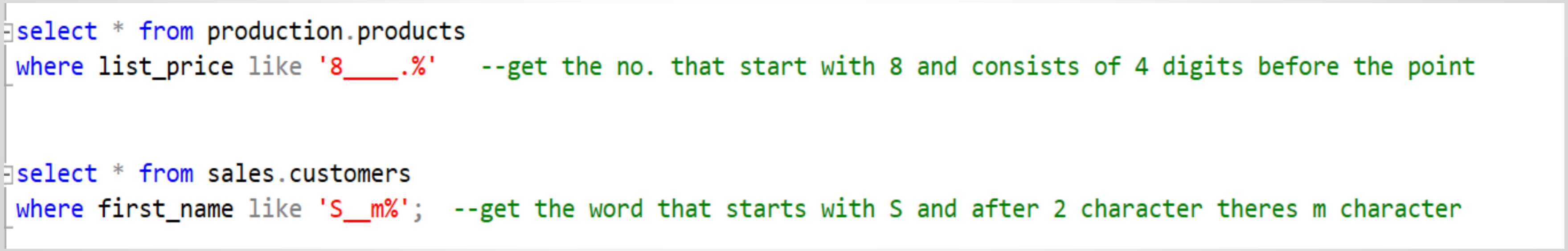


```
select * from sales.customers
where first_name like '[A-E]%';  --get the words starts with A to E "Include :A,B,C,D,E"


select * from sales.customers
where first_name not like '[A-E]%'; -- get any words didn't start with A to E
```

SELECT STATEMENT

- **Using Like operator (_%)**

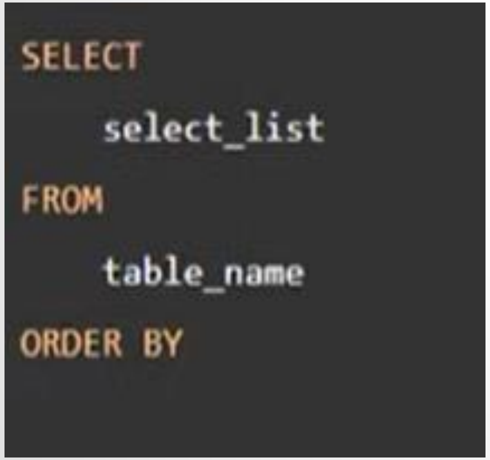
- **Example:**



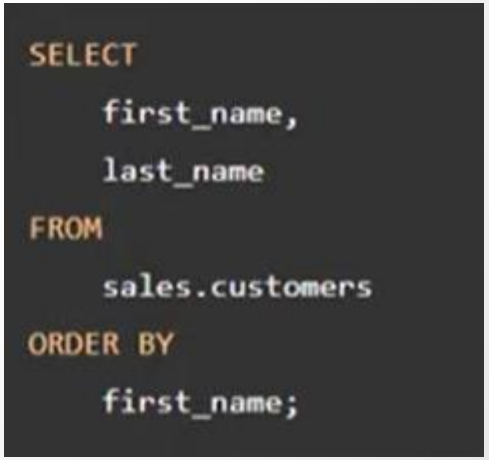
```
select * from production.products
where list_price like '8____.%'   --get the no. that start with 8 and consists of 4 digits before the point


select * from sales.customers
where first_name like 'S__m%';  --get the word that starts with S and after 2 character theres m character
```

ORDER BY

- Using to sort the results [ASC | DESC]



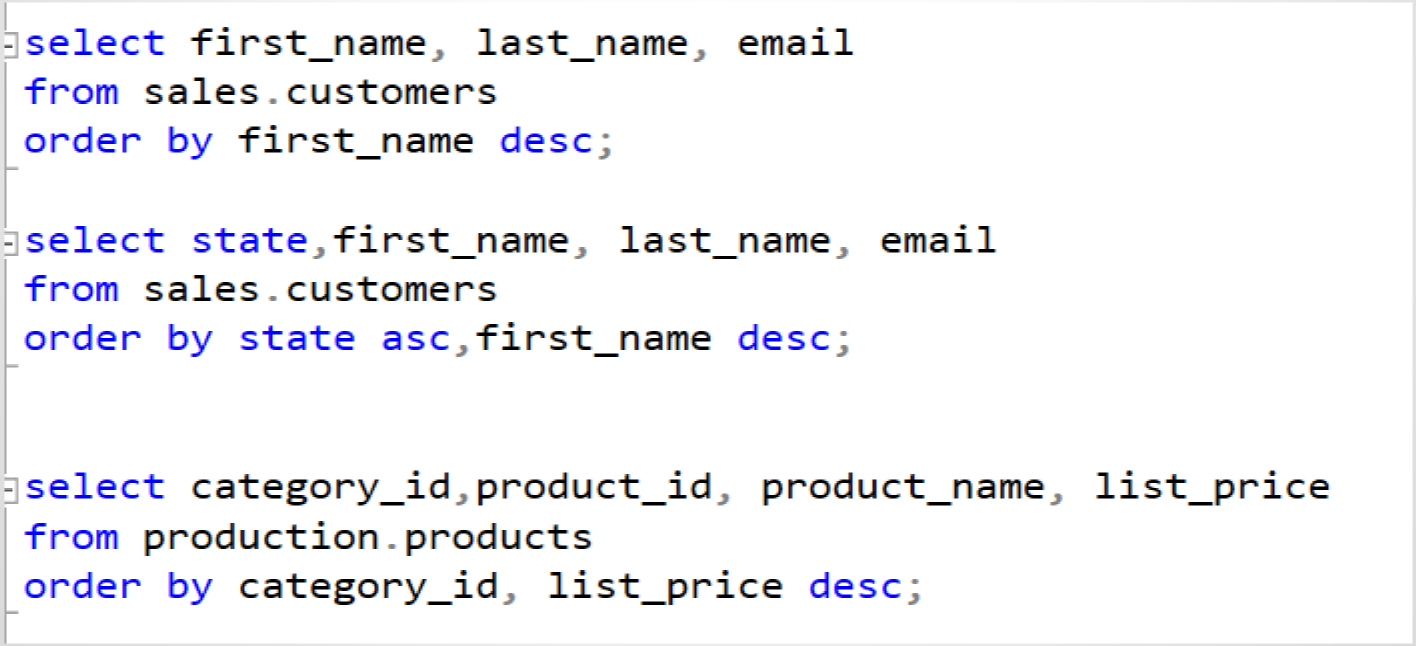
```
SELECT
    select_list
FROM
    table_name
ORDER BY
```



```
SELECT
    first_name,
    last_name
FROM
    sales.customers
ORDER BY
    first_name;
```

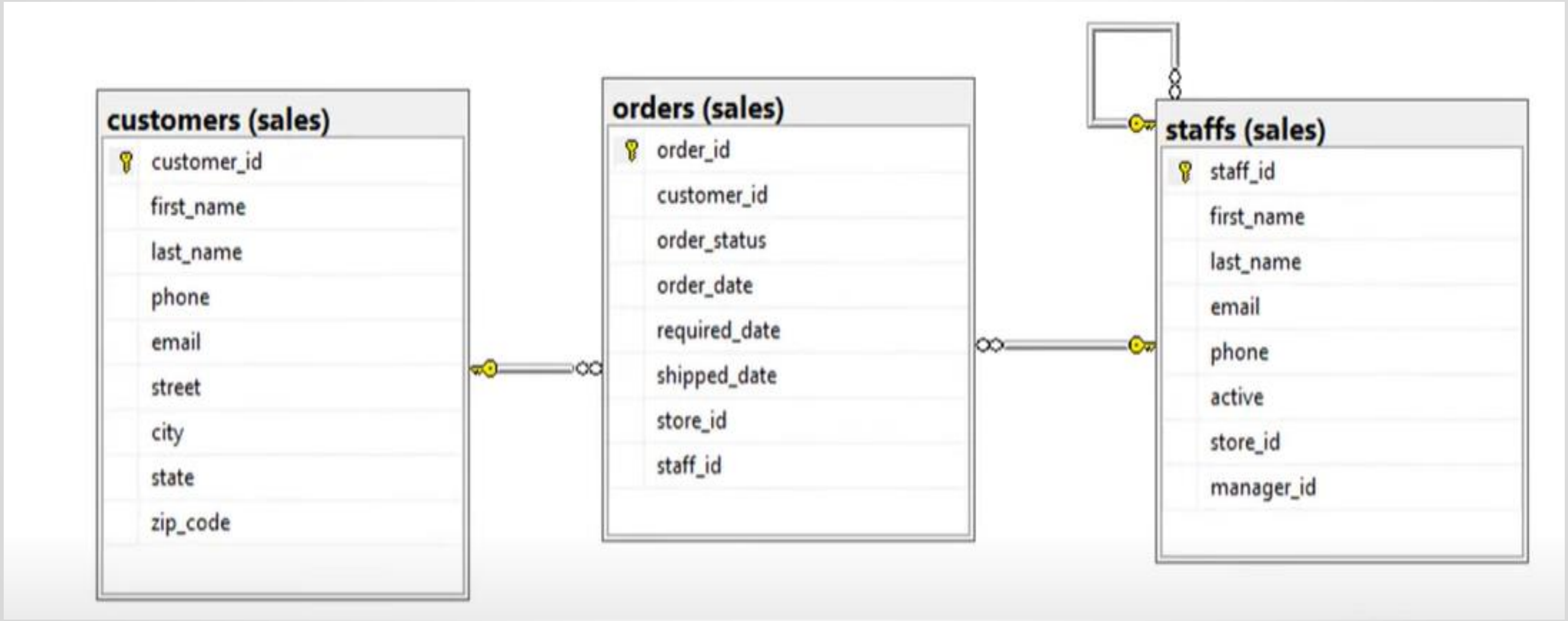
ORDER BY

- Using to sort the results [ASC | DESC]



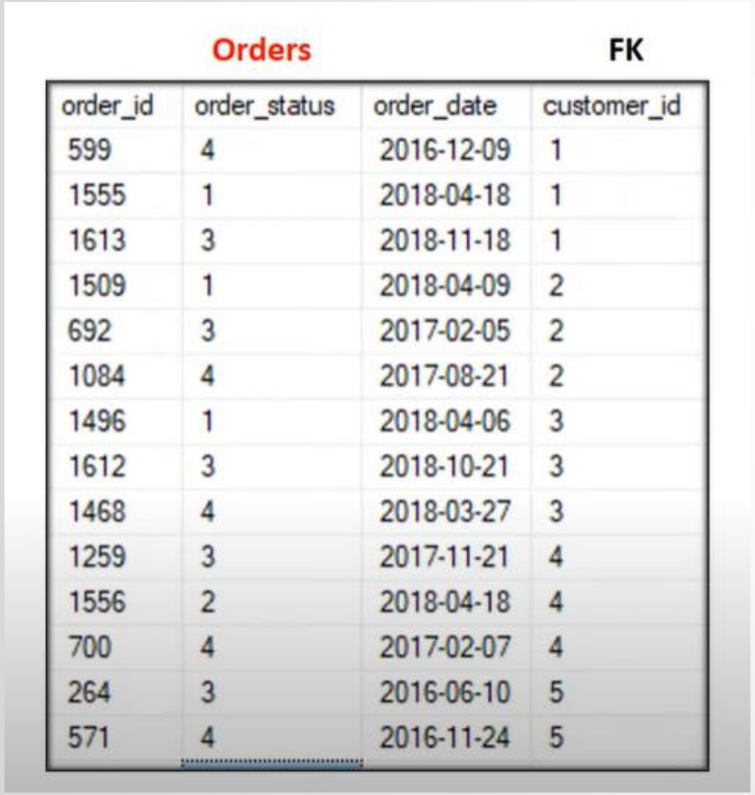
```sql
select first_name, last_name, email
from sales.customers
order by first_name desc;

select state,first_name, last_name, email
from sales.customers
order by state asc,first_name desc;


select category_id,product_id, product_name, list_price
from production.products
order by category_id, list_price desc;
```

Diagram



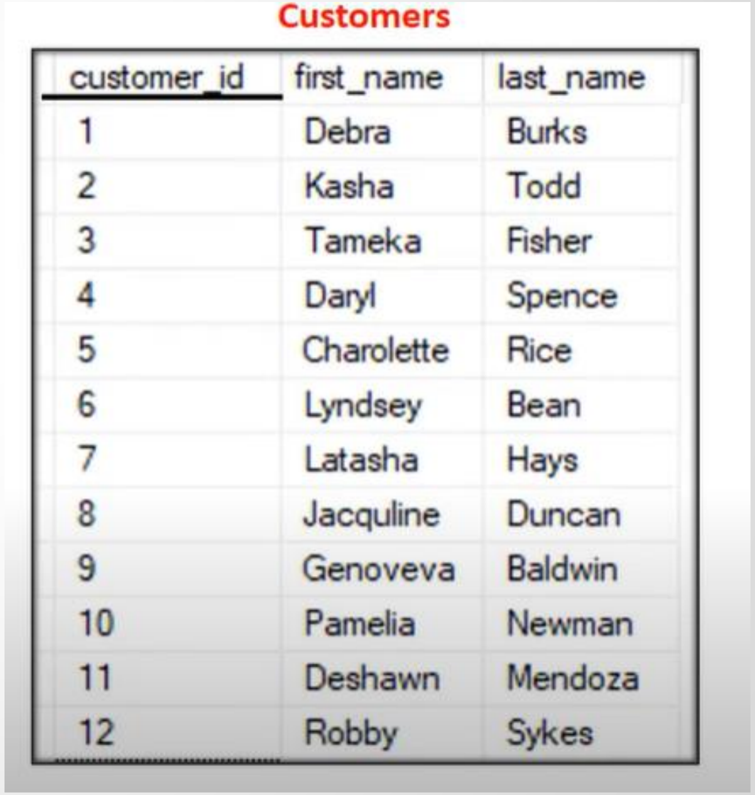
customers (sales)
customer_id
first_name
last_name
phone
email
street
city
state
zip_code
orders (sales)
order_id
customer_id
order_status
order_date
required_date
shipped_date
store_id
staff_id
staffs (sales)
staff_id
first_name
last_name
email
phone
active
store_id
manager_id

How to Join 2 tables



Orders FK

order_id	order_status	order_date	customer_id
599	4	2016-12-09	1
1555	1	2018-04-18	1
1613	3	2018-11-18	1
1509	1	2018-04-09	2
692	3	2017-02-05	2
1084	4	2017-08-21	2
1496	1	2018-04-06	3
1612	3	2018-10-21	3
1468	4	2018-03-27	3
1259	3	2017-11-21	4
1556	2	2018-04-18	4
700	4	2017-02-07	4
264	3	2016-06-10	5
571	4	2016-11-24	5

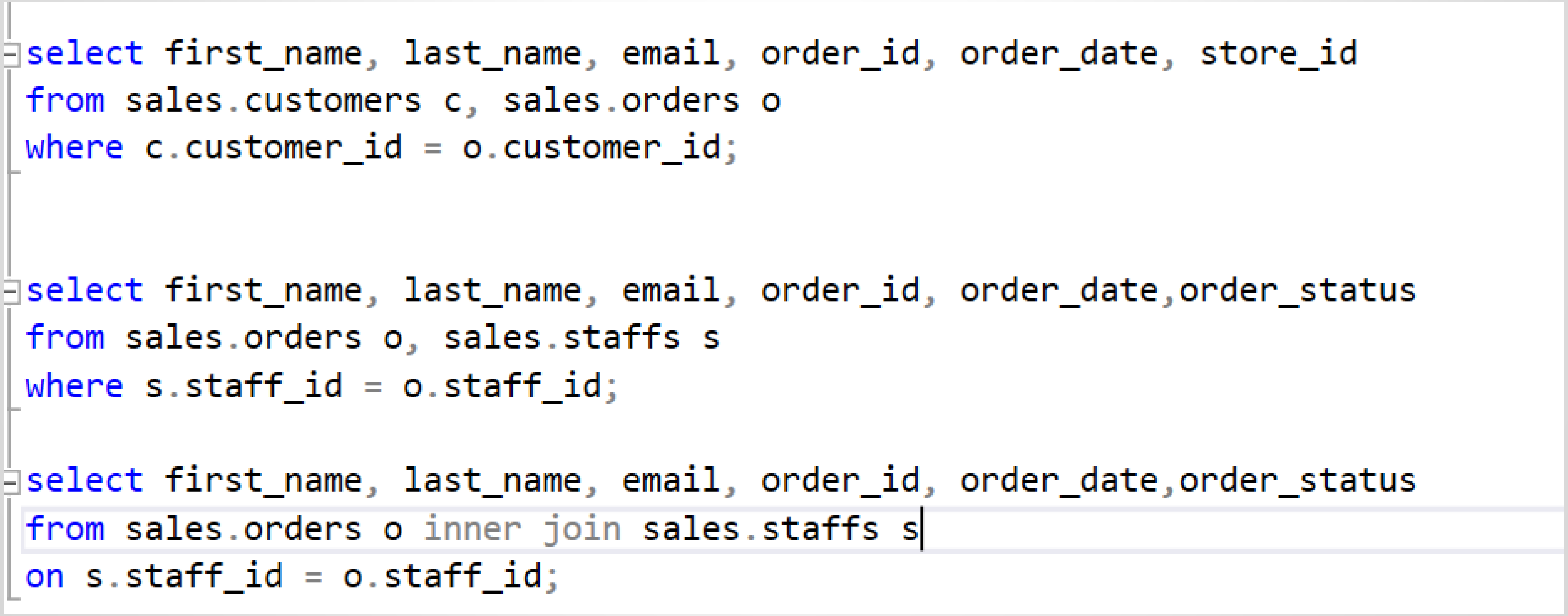


Customers

customer_id	first_name	last_name
1	Debra	Burks
2	Kasha	Todd
3	Tameka	Fisher
4	Daryl	Spence
5	Charolette	Rice
6	Lyndsey	Bean
7	Latasha	Hays
8	Jacquline	Duncan
9	Genoveva	Baldwin
10	Pamelia	Newman
11	Deshawn	Mendoza
12	Robby	Sykes

How to Join 2 tables

Inner Join



```
select first_name, last_name, email, order_id, order_date, store_id
from sales.customers c, sales.orders o
where c.customer_id = o.customer_id;



select first_name, last_name, email, order_id, order_date,order_status
from sales.orders o, sales.staffs s
where s.staff_id = o.staff_id;

select first_name, last_name, email, order_id, order_date,order_status
from sales.orders o inner join sales.staffs s
on s.staff_id = o.staff_id;
```

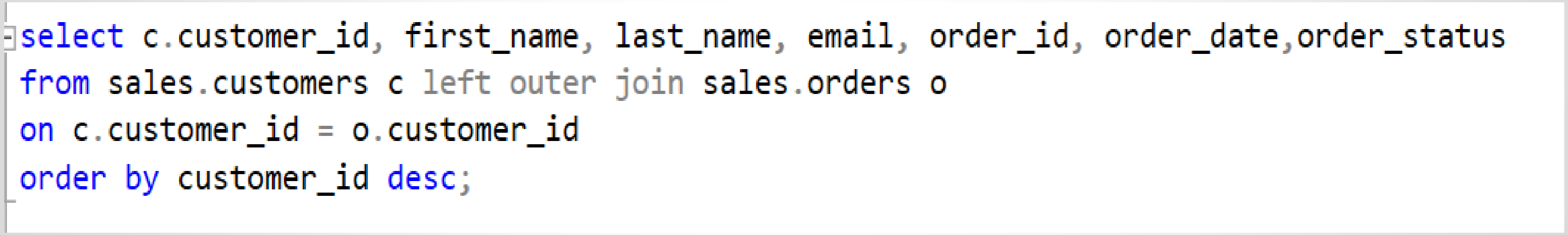
How to Join 2 tables

Natural Join:

If you have column with specific name in the table and you have the same name of column in other table and you want to join the two tables together then in this case we use Natural Join.

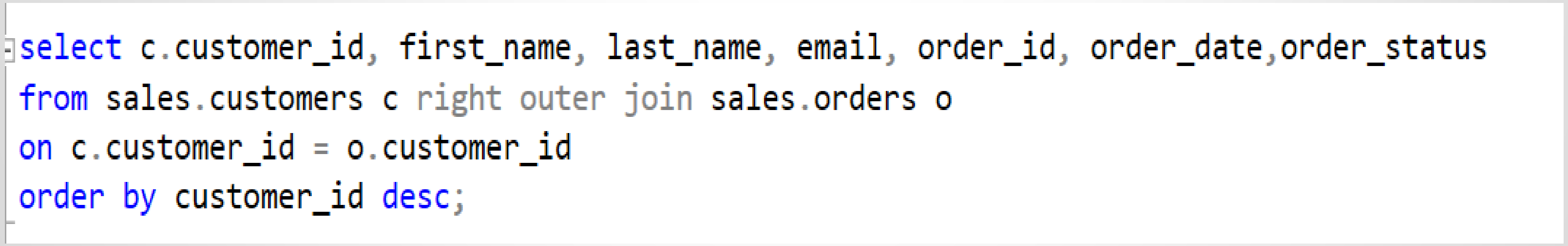
How to Join 2 tables

Left Outer Join


```
select c.customer_id, first_name, last_name, email, order_id, order_date,order_status
from sales.customers c left outer join sales.orders o
on c.customer_id = o.customer_id
order by customer_id desc;
```

How to Join 2 tables

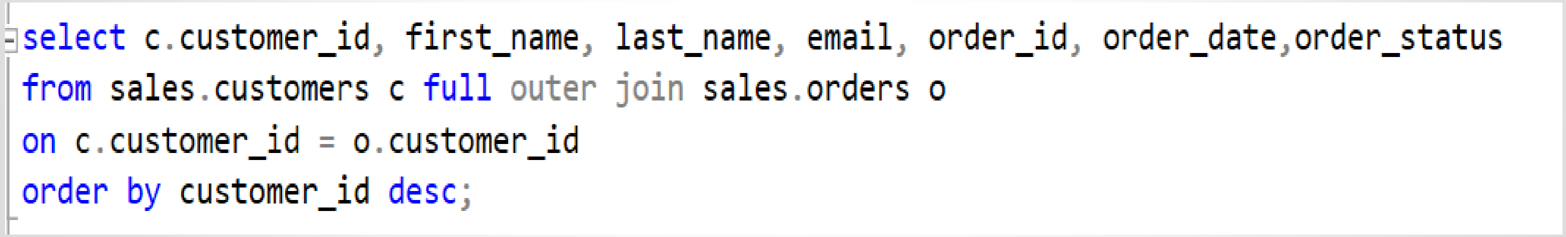
Right Outer Join



```
select c.customer_id, first_name, last_name, email, order_id, order_date,order_status
from sales.customers c right outer join sales.orders o
on c.customer_id = o.customer_id
order by customer_id desc;
```

How to Join 2 tables

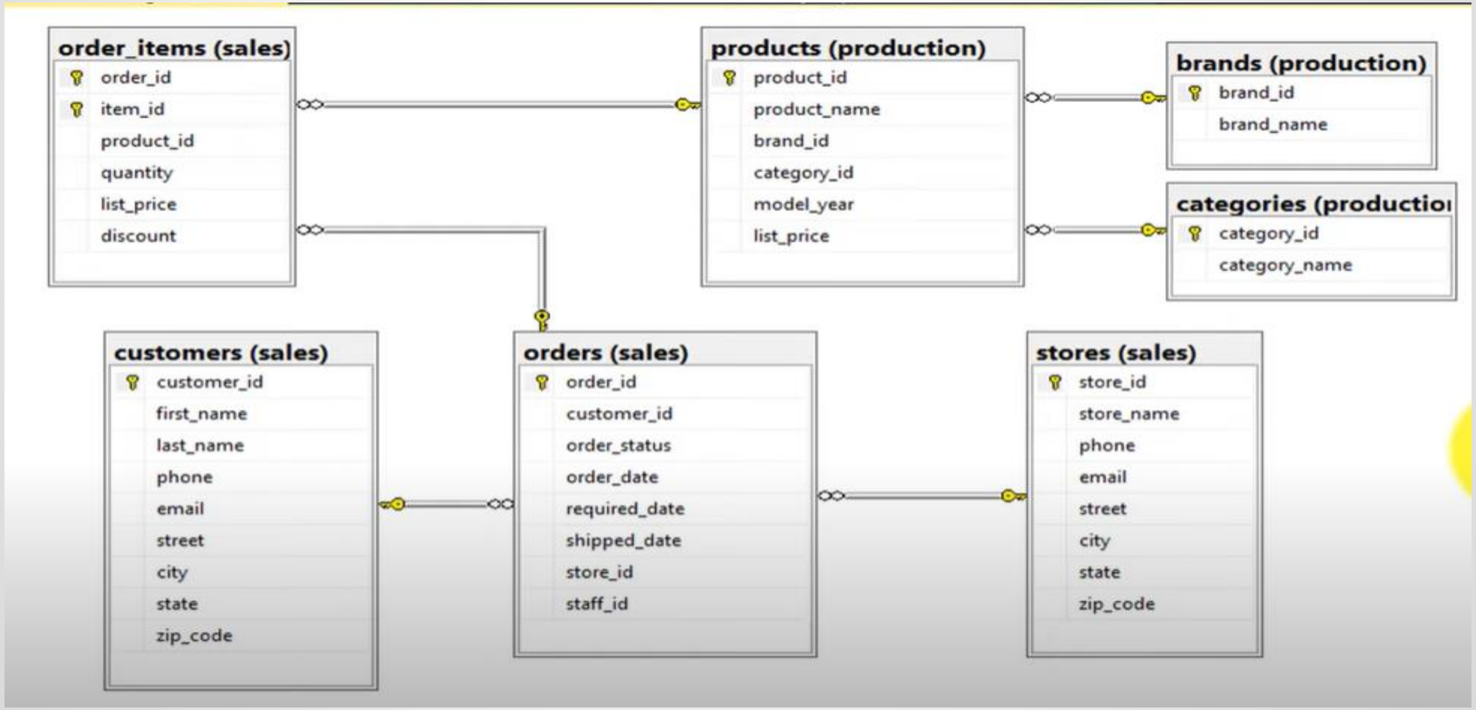
Full Outer Join



```sql
select c.customer_id, first_name, last_name, email, order_id, order_date,order_status
from sales.customers c full outer join sales.orders o
on c.customer_id = o.customer_id
order by customer_id desc;
```

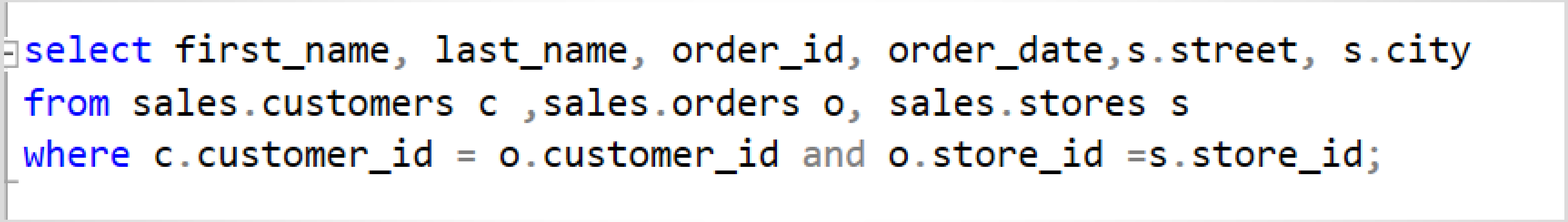
How to Join more than 2 tables

Diagram

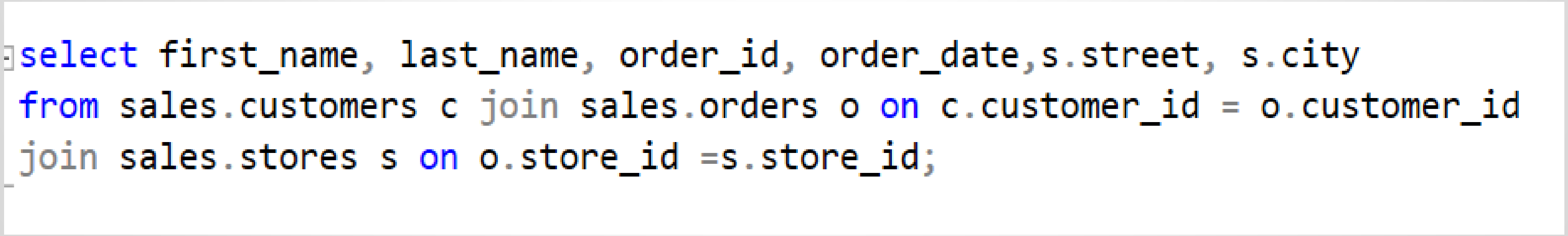


How to Join more than 2 tables

Join Customer table with Store table



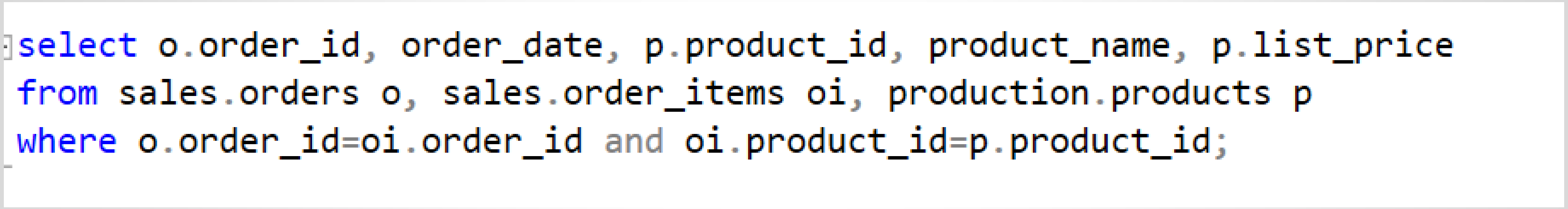
```sql
select first_name, last_name, order_id, order_date,s.street, s.city
from sales.customers c ,sales.orders o, sales.stores s
where c.customer_id = o.customer_id and o.store_id =s.store_id;
```



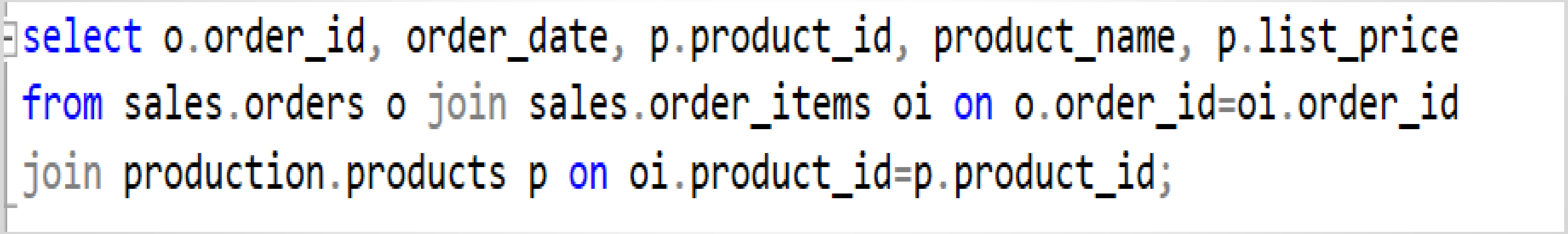
```sql
select first_name, last_name, order_id, order_date,s.street, s.city
from sales.customers c join sales.orders o on c.customer_id = o.customer_id
join sales.stores s on o.store_id =s.store_id;
```

How to Join more than 2 tables

Join Order table with Product table



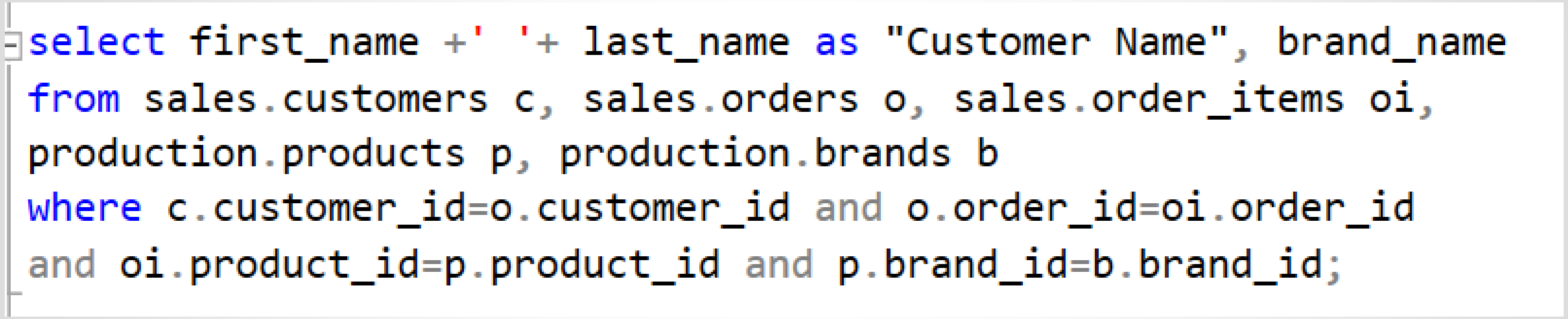
```
select o.order_id, order_date, p.product_id, product_name, p.list_price
from sales.orders o, sales.order_items oi, production.products p
where o.order_id=oi.order_id and oi.product_id=p.product_id;
```



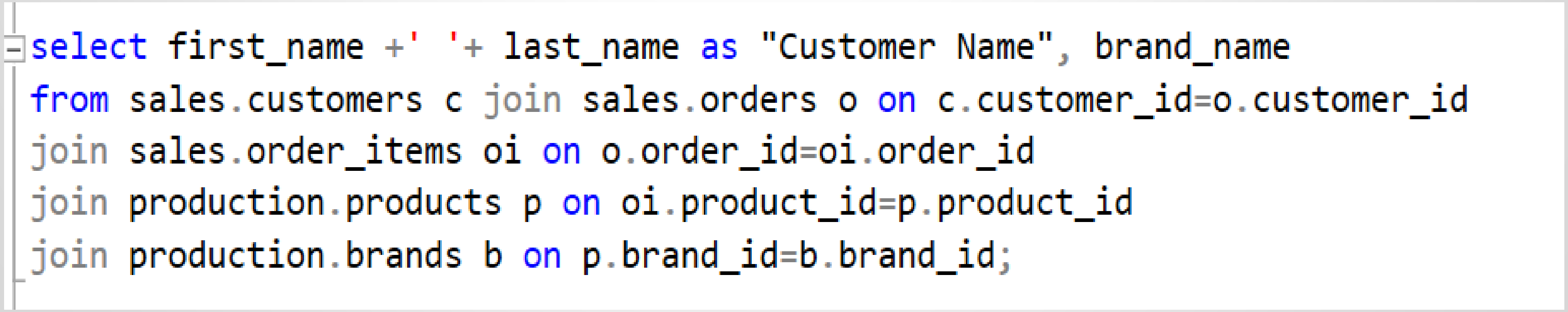
```
select o.order_id, order_date, p.product_id, product_name, p.list_price
from sales.orders o join sales.order_items oi on o.order_id=oi.order_id
join production.products p on oi.product_id=p.product_id;
```

How to Join more than 2 tables

Join Customer table with Brand table

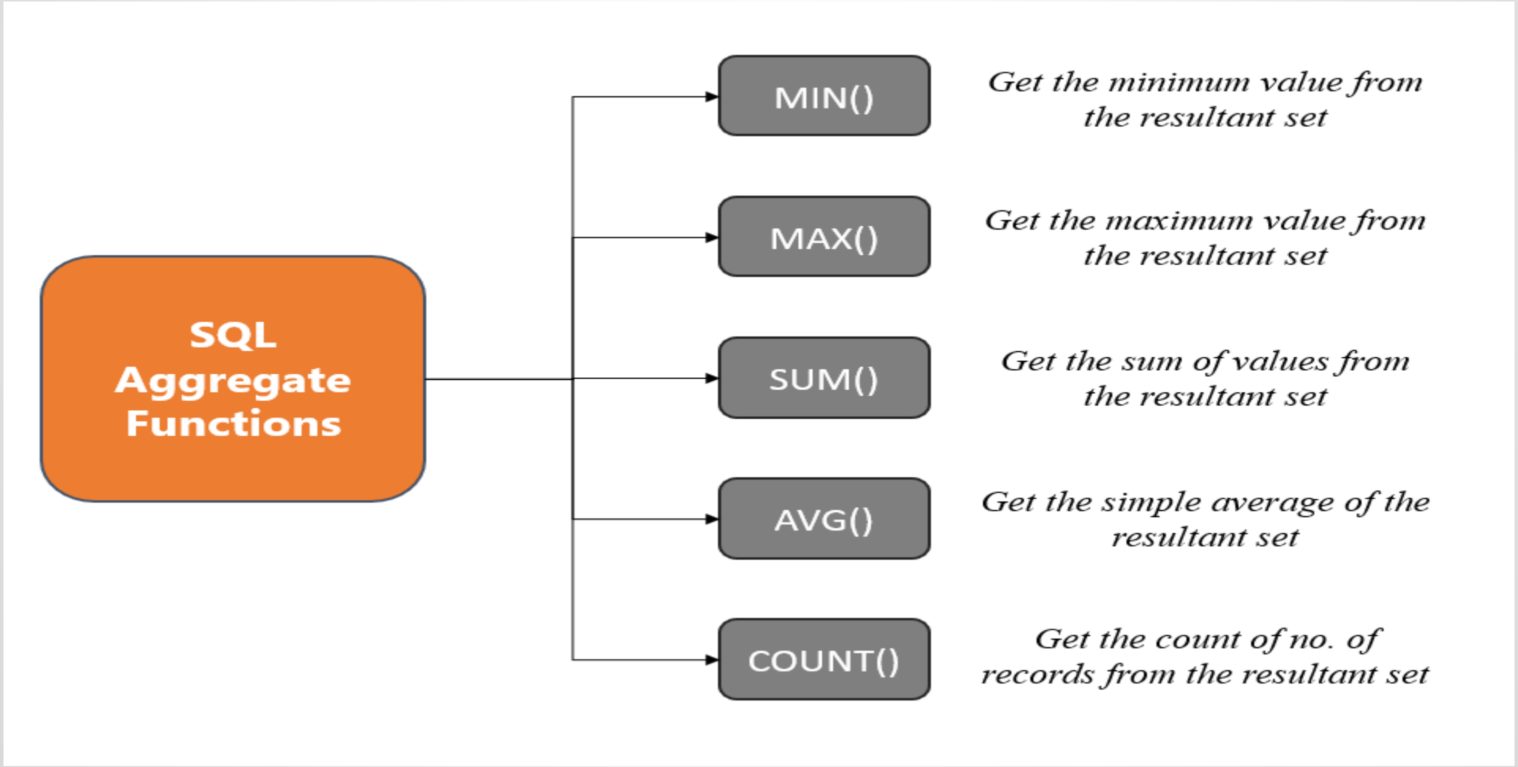


```sql
select first_name +' '+ last_name as "Customer Name", brand_name
from sales.customers c, sales.orders o, sales.order_items oi,
production.products p, production.brands b
where c.customer_id=o.customer_id and o.order_id=oi.order_id
and oi.product_id=p.product_id and p.brand_id=b.brand_id;
```



```sql
select first_name +' '+ last_name as "Customer Name", brand_name
from sales.customers c join sales.orders o on c.customer_id=o.customer_id
join sales.order_items oi on o.order_id=oi.order_id
join production.products p on oi.product_id=p.product_id
join production.brands b on p.brand_id=b.brand_id;
```

AGGREGATION FUNCTION


SQL Aggregate Functions
MIN()
Get the minimum value from the resultant set
MAX()
Get the maximum value from the resultant set
SUM()
Get the sum of values from the resultant set
AVG()
Get the simple average of the resultant set
COUNT()
Get the count of no. of records from the resultant set

AGGREGATION FUNCTION

Performs a calculation one or more values and returns a single value.

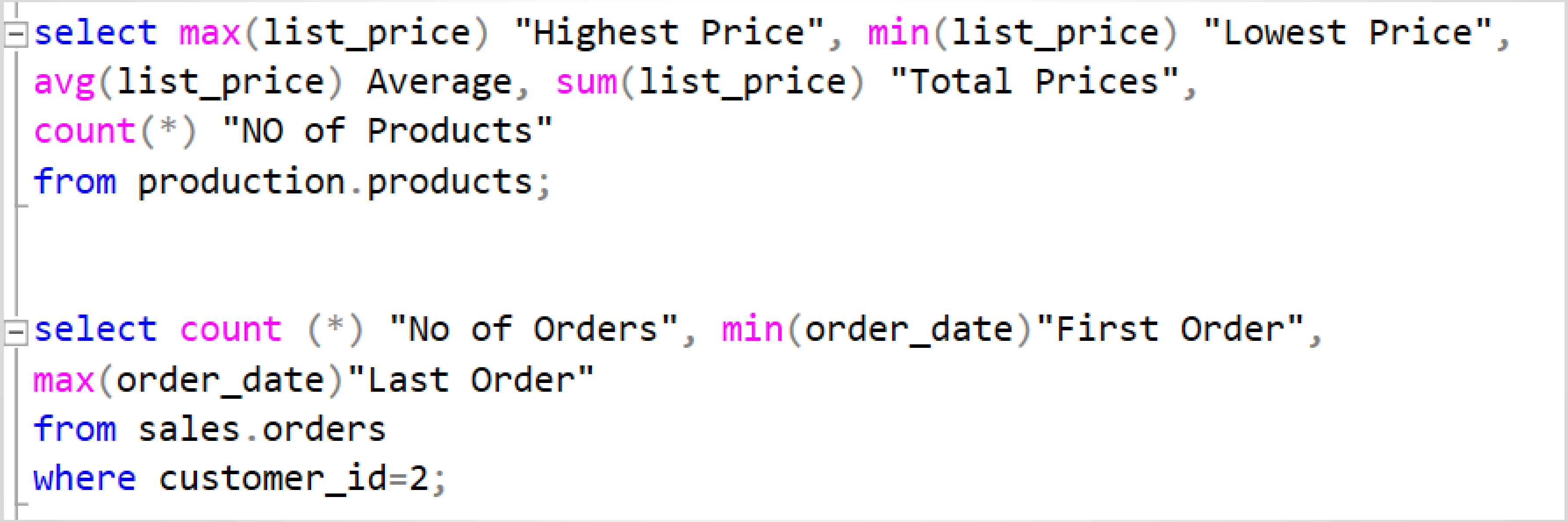
```
aggregate_function_name(DISTINCT | ALL expression)
```

Example:

```
SELECT
    AVG(list_price) avg_product_price
FROM
    production.products;
```

AGGREGATION FUNCTION

Example:

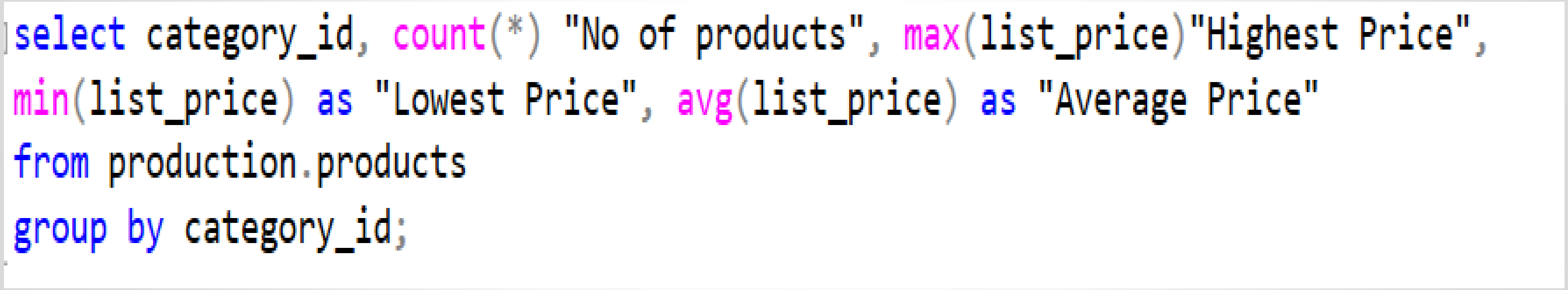


```sql
select max(list_price) "Highest Price", min(list_price) "Lowest Price",
avg(list_price) Average, sum(list_price) "Total Prices",
count(*) "NO of Products"
from production.products;


select count (*) "No of Orders", min(order_date)"First Order",
max(order_date)"Last Order"
from sales.orders
where customer_id=2;
```

AGGREGATION FUNCTION

Group By:

For each category, list category_id, max price, lowest price, average price;

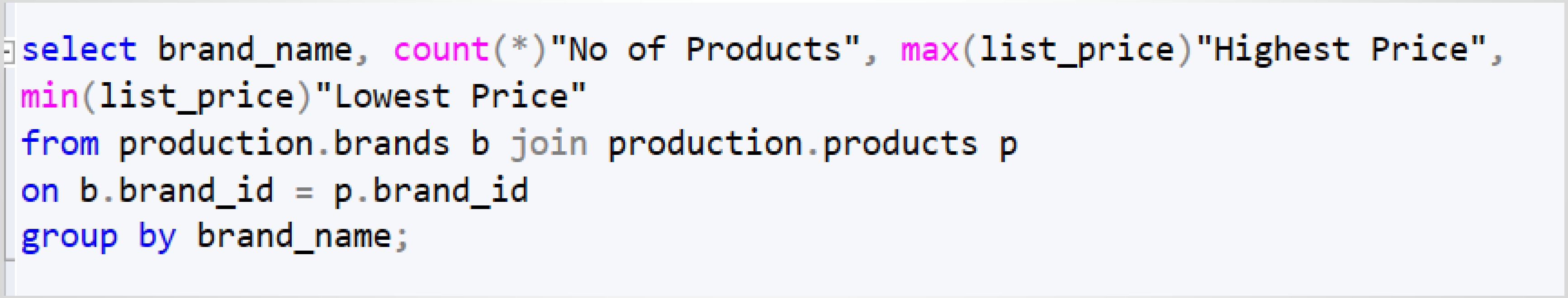


```
select category_id, count(*) "No of products", max(list_price)"Highest Price",
min(list_price) as "Lowest Price", avg(list_price) as "Average Price"
from production.products
group by category_id;
```

AGGREGATION FUNCTION

Group By:

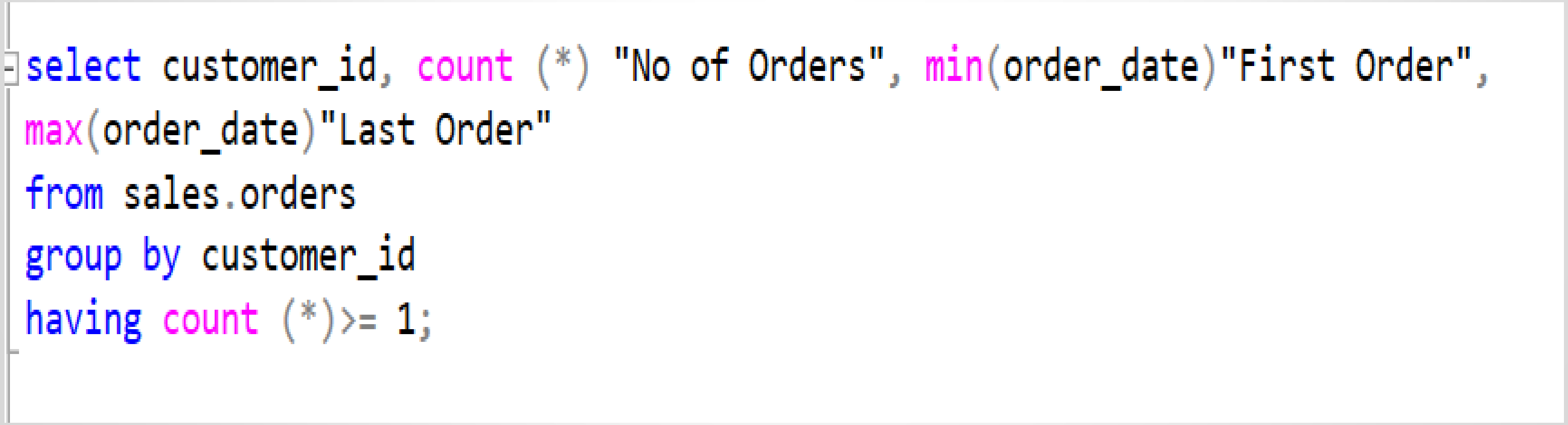
For each brand, display a list of brand name, no of products for that the highest and lowest price in the brand.



```
select brand_name, count(*)"No of Products", max(list_price)"Highest Price",
min(list_price)"Lowest Price"
from production.brands b join production.products p
on b.brand_id = p.brand_id
group by brand_name;
```

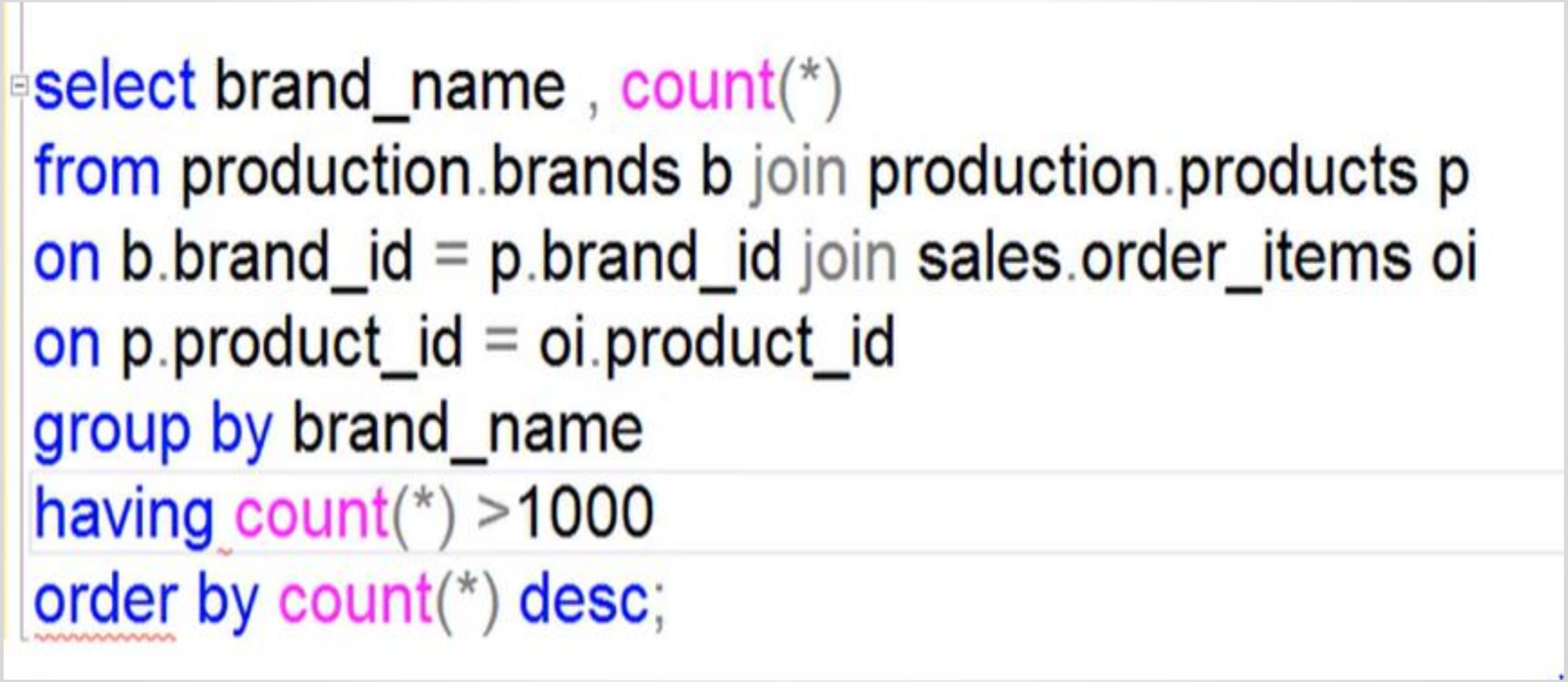
AGGREGATION FUNCTION

Having :



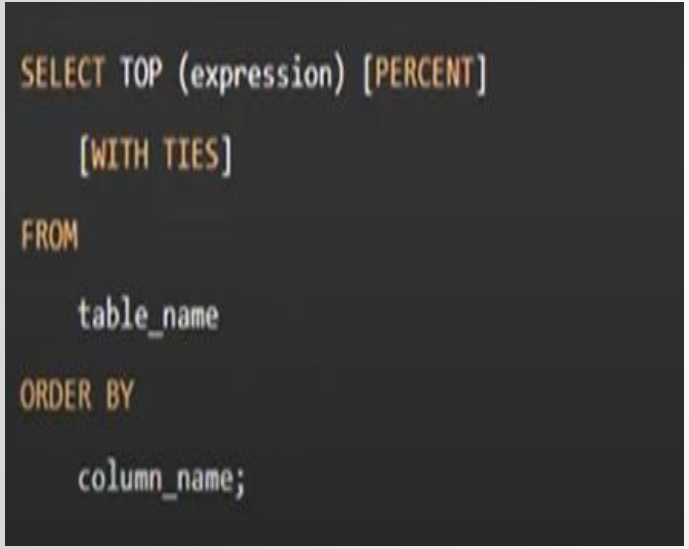
```
select customer_id, count (*) "No of Orders", min(order_date)"First Order",
max(order_date)"Last Order"
from sales.orders
group by customer_id
having count (*)>= 1;
```

Join with Group by and Order by

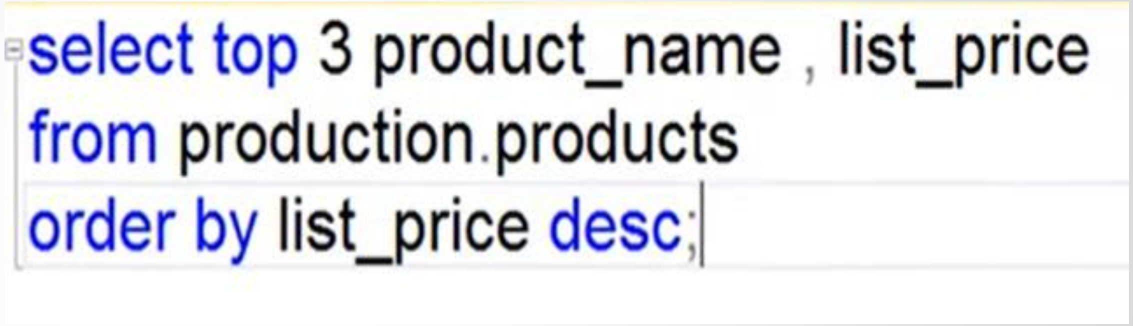


```
select brand_name , count(*)
from production.brands b join production.products p
on b.brand_id = p.brand_id join sales.order_items oi
on p.product_id = oi.product_id
group by brand_name
having count(*) >1000
order by count(*) desc;
```

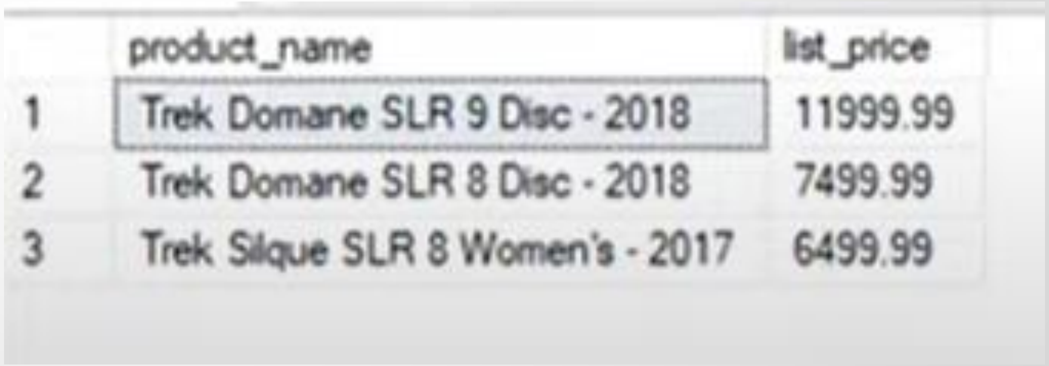
Select Top Records



```
SELECT TOP (expression) [PERCENT]
    [WITH TIES]
FROM
    table_name
ORDER BY
    column_name;
```

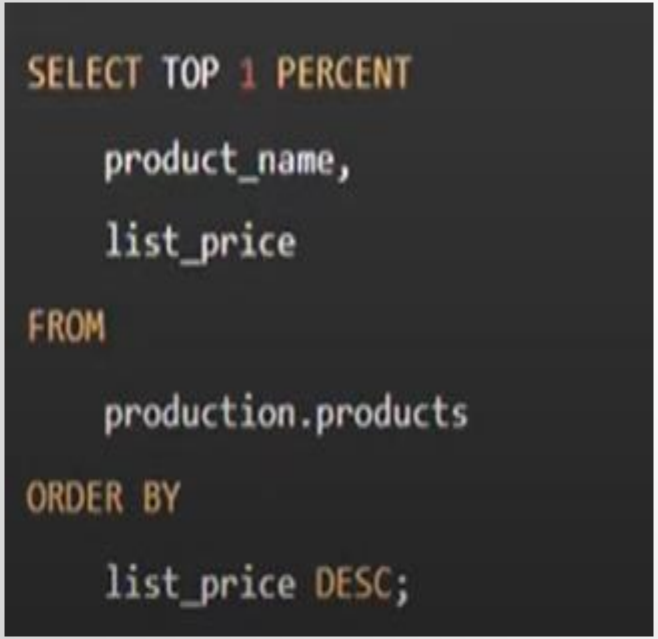


```
select top 3 product_name , list_price
from production.products
order by list_price desc;
```

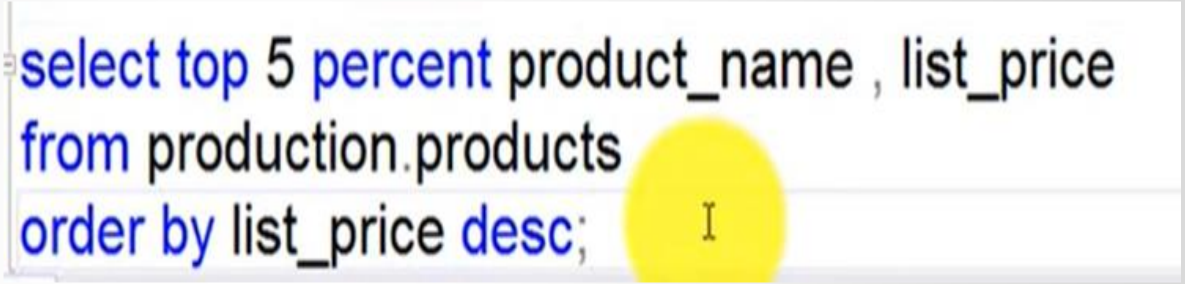


	product_name	list_price
1	Trek Domane SLR 9 Disc - 2018	11999.99
2	Trek Domane SLR 8 Disc - 2018	7499.99
3	Trek Silque SLR 8 Women's - 2017	6499.99

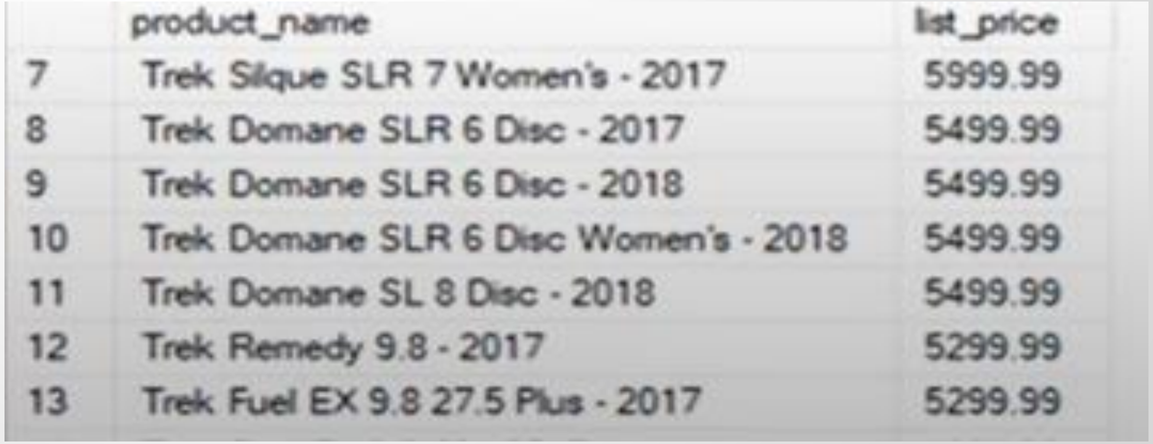
Select Top Records



```
SELECT TOP 1 PERCENT
    product_name,
    list_price
FROM
    production.products
ORDER BY
    list_price DESC;
```

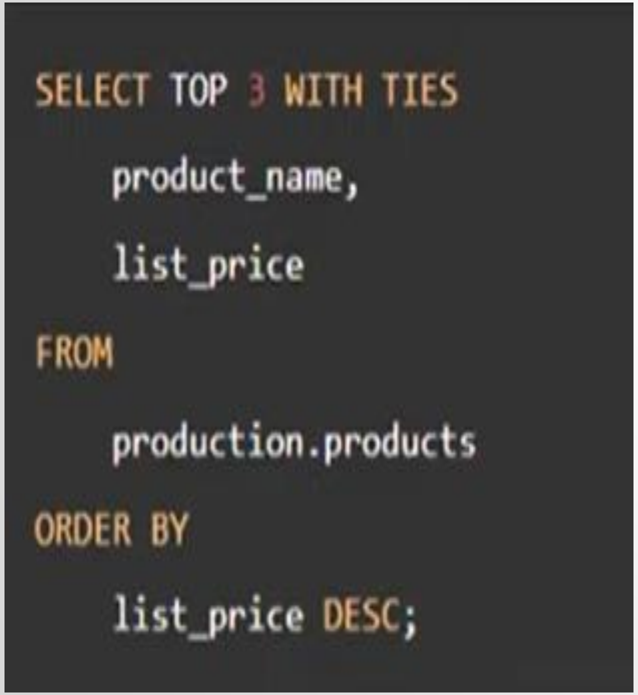


```
select top 5 percent product_name , list_price
from production.products
order by list_price desc;
```

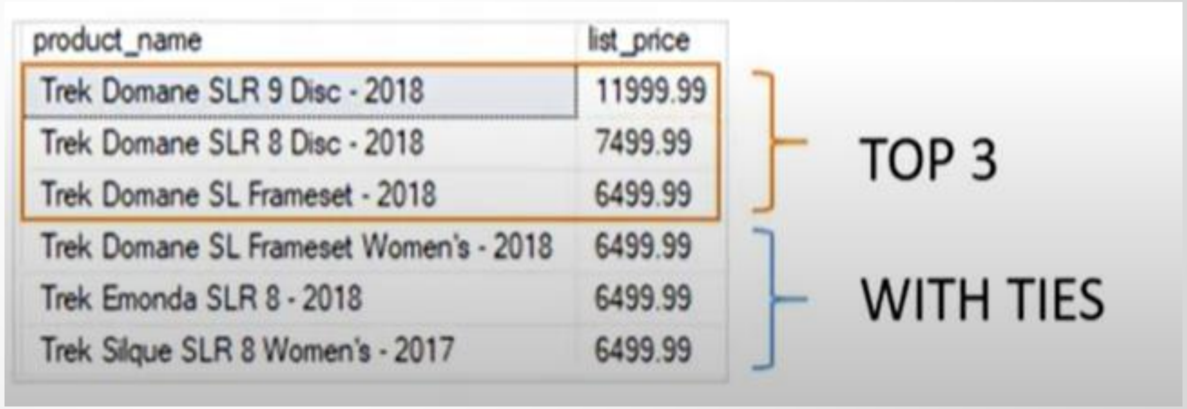


	product_name	list_price
7	Trek Silque SLR 7 Women's - 2017	5999.99
8	Trek Domane SLR 6 Disc - 2017	5499.99
9	Trek Domane SLR 6 Disc - 2018	5499.99
10	Trek Domane SLR 6 Disc Women's - 2018	5499.99
11	Trek Domane SL 8 Disc - 2018	5499.99
12	Trek Remedy 9.8 - 2017	5299.99
13	Trek Fuel EX 9.8 27.5 Plus - 2017	5299.99

Select Top Records

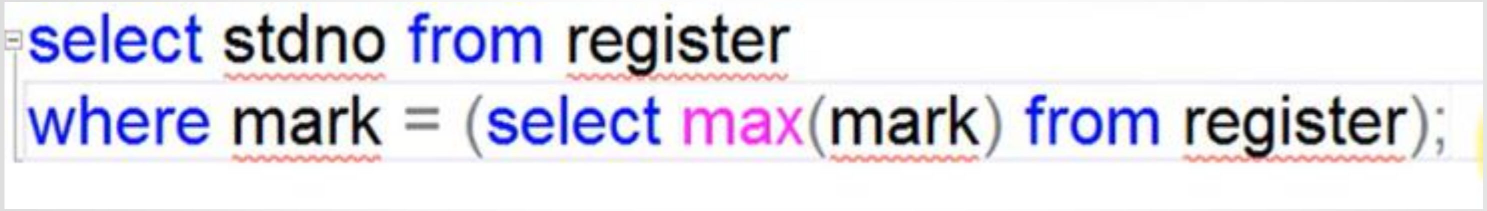


```
SELECT TOP 3 WITH TIES
    product_name,
    list_price
FROM
    production.products
ORDER BY
    list_price DESC;
```

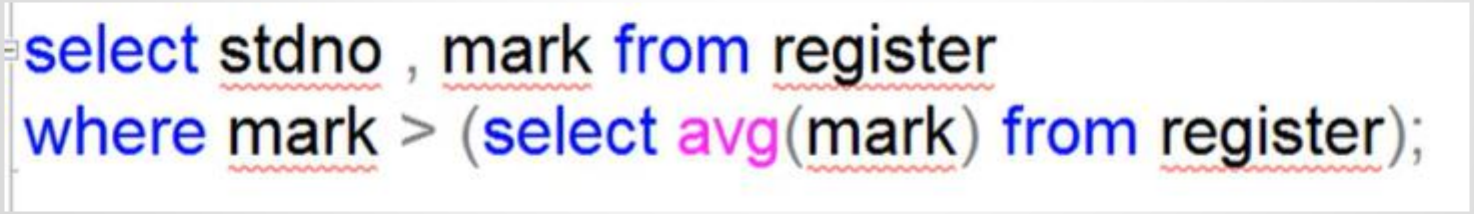


product_name	list_price
Trek Domane SLR 9 Disc - 2018	11999.99
Trek Domane SLR 8 Disc - 2018	7499.99
Trek Domane SL Frameset - 2018	6499.99
Trek Domane SL Frameset Women's - 2018	6499.99
Trek Emonda SLR 8 - 2018	6499.99
Trek Silque SLR 8 Women's - 2017	6499.99

Nested queries - Sub queries

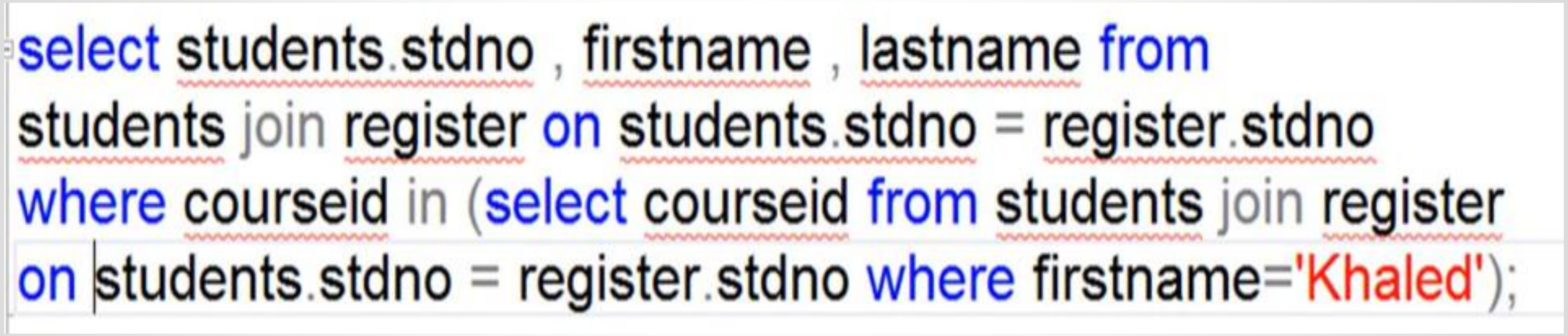


```
select stdno from register
where mark = (select max(mark) from register);
```



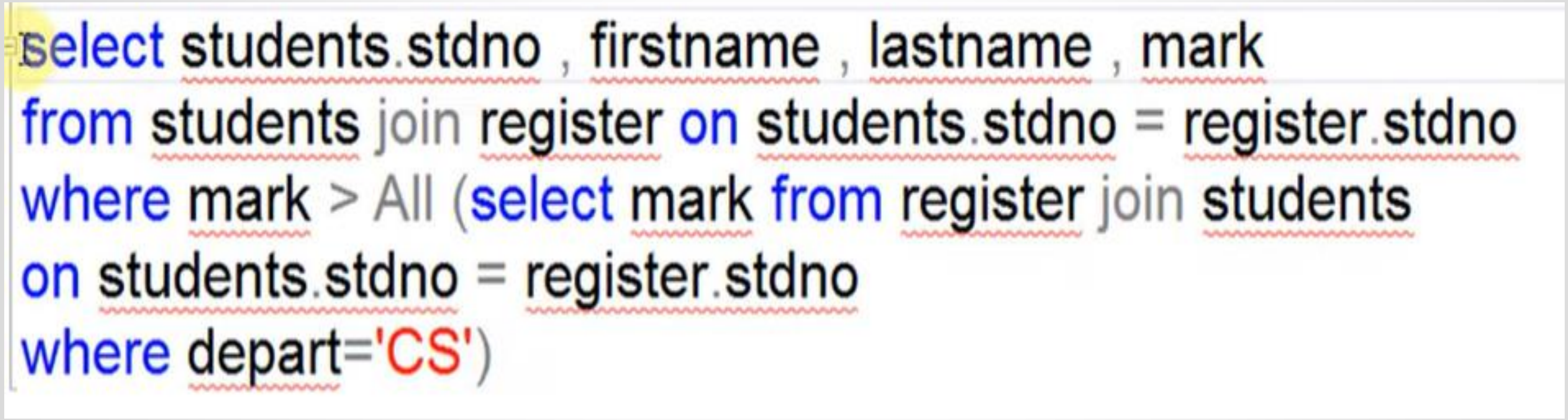
```
select stdno , mark from register
where mark > (select avg(mark) from register);
```

Nested queries - Sub queries



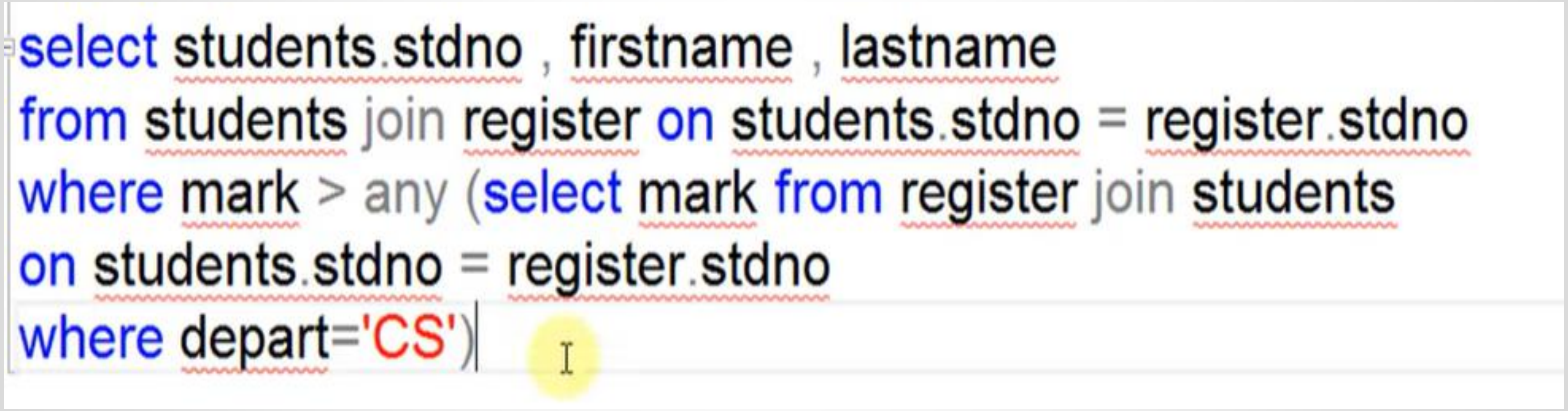
```
select students.stdno , firstname , lastname from
students join register on students.stdno = register.stdno
where courseid in (select courseid from students join register
on students.stdno = register.stdno where firstname='Khaled');
```

Nested queries - Sub queries



```
select students.stdno , firstname , lastname , mark
from students join register on students.stdno = register.stdno
where mark > All (select mark from register join students
on students.stdno = register.stdno
where depart='CS')
```

Nested queries - Sub queries

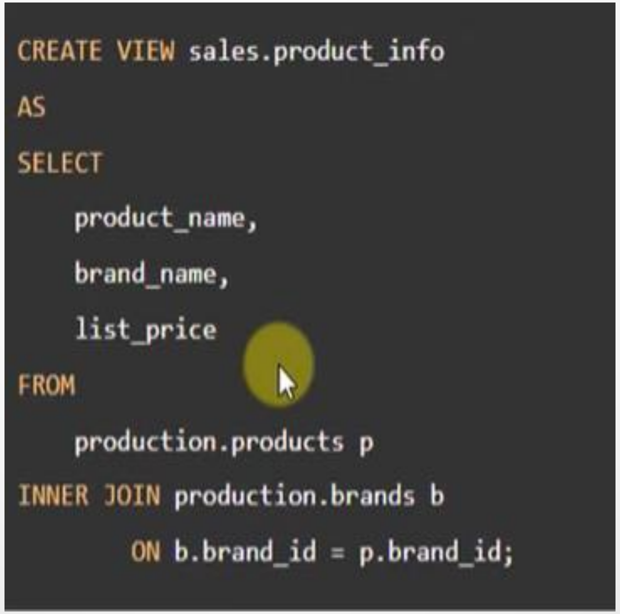


```
select students.stdno , firstname , lastname
from students join register on students.stdno = register.stdno
where mark > any (select mark from register join students
on students.stdno = register.stdno
where depart='CS')
```

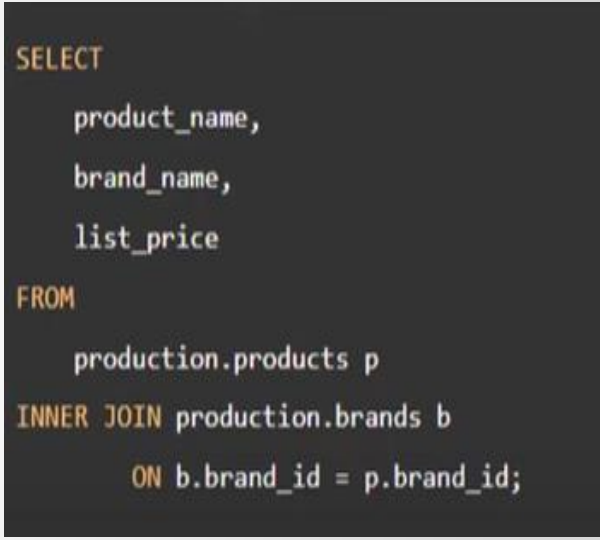
Create a View

Create View:

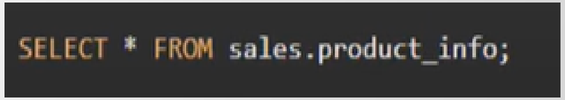
- Better way to save this query in the database catalog .
- Is named query stored in the database catalog that always you refer to it later



```
CREATE VIEW sales.product_info
AS
SELECT
    product_name,
    brand_name,
    list_price
FROM
    production.products p
INNER JOIN production.brands b
        ON b.brand_id = p.brand_id;
```



```
SELECT
    product_name,
    brand_name,
    list_price
FROM
    production.products p
INNER JOIN production.brands b
        ON b.brand_id = p.brand_id;
```



```
SELECT * FROM sales.product_info;
```

Thank you!

Do you have any questions?
amiraaaadel6669@gmail.com
+201021400185
Amira_Fouda_Linked_In