

## Introduction

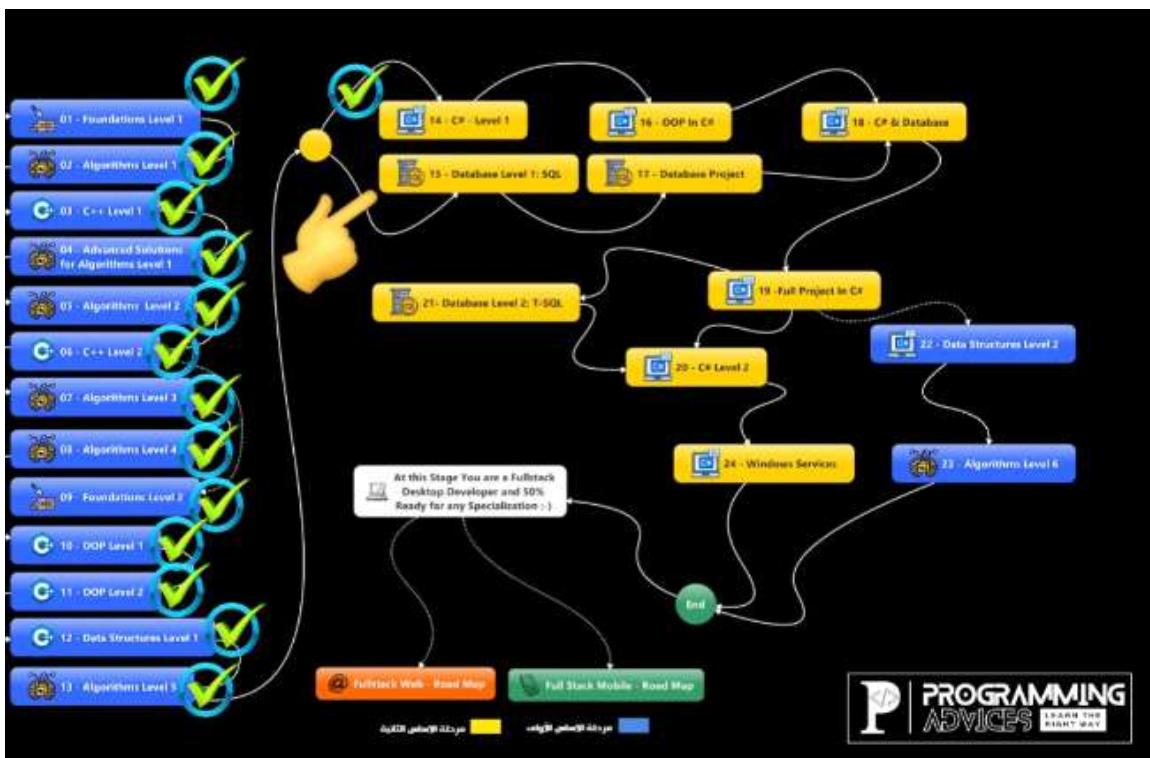
## About This Course

من الحاجات اللي بتؤثر في سرعة البرنامج هو الكود اللي بتعمله في الداتا بيز



**مميزات الكورس**

قواعد البيانات تعتبر من اهم المواضيع التي يجب على كل مبرمج اتقانها بشكل قوي جدا واعطاوها اهمية كبيرة ، لانها تؤثر بشكل كبير على اداء البرنامج وايضا على الوقت الذي يحتاجه المبرمج في اتمام البرنامج، اتقانها يوفر وقت كبير جدا في البرمجة، في هذا الكورس سنتعرف على قواعد البيانات وطرق التعامل معها وسنتعلم لغة SQL بكل تفاصيلها من واقع عملى وليس فقط نظري سيختصر عليكم خبرات سنين ، وسندرسها بشكل تدريجي مع التطبيق لترسيخ المعلومات واكتساب الخبرة.



## Telegram Group

<https://t.me/+qXlqgnw944VkNThk>

## How to install SQL Server 2022

هنزل SQL full featured و هيا SQL server developer edition من ال server وبتمكنك انك تعمل اللي انت عايشه على الداتا بيز

هندخل عالرابط ده

<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

## Try SQL Server on-premises or in the cloud



### SQL Server on Azure

on Azure SQL with built-in security and manageability.

Get started



### SQL Server at the edge

Extend SQL to IoT devices for real-time analysis with Azure SQL Edge.

Get started



### SQL Server on-premises

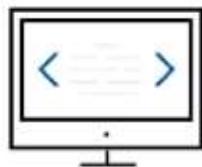
Get the performance and security of SQL Server 2022, a scalable, hybrid data platform, now Azure-enabled.

Download now

Or, download a free specialized edition



لازم تمشي معاه خطوة خطوه وماتعملش حاجه من نفسك عشان لو عملت حاجه  
غلط هتحصل مشاكل وهتضطر تعينه من الأول



## Developer

SQL Server 2022 Developer is a full-featured free edition, licensed for use as a development and test database in a non-production environment.

[Download now](#)

erver 2022



-

X

# Developer Edition

an installation type:

1

asic installation type to  
e SQL Server Database  
eature with default  
ration.

## Custom

Select Custom installation type to step through the SQL Server installation wizard and choose what you want to install. This installation type is detailed and takes longer than running the Basic install.

## Download Media

Download SQL Server setup files now and install them later on a machine of your choice.

Transmits information about your installation experience, as well as other usage and performance data, to Microsoft to help improve our products. To learn more about data processing and privacy controls, and to turn off the collection of this information after installation, see the [Privacy Statement](#).

16.2211.5693.3

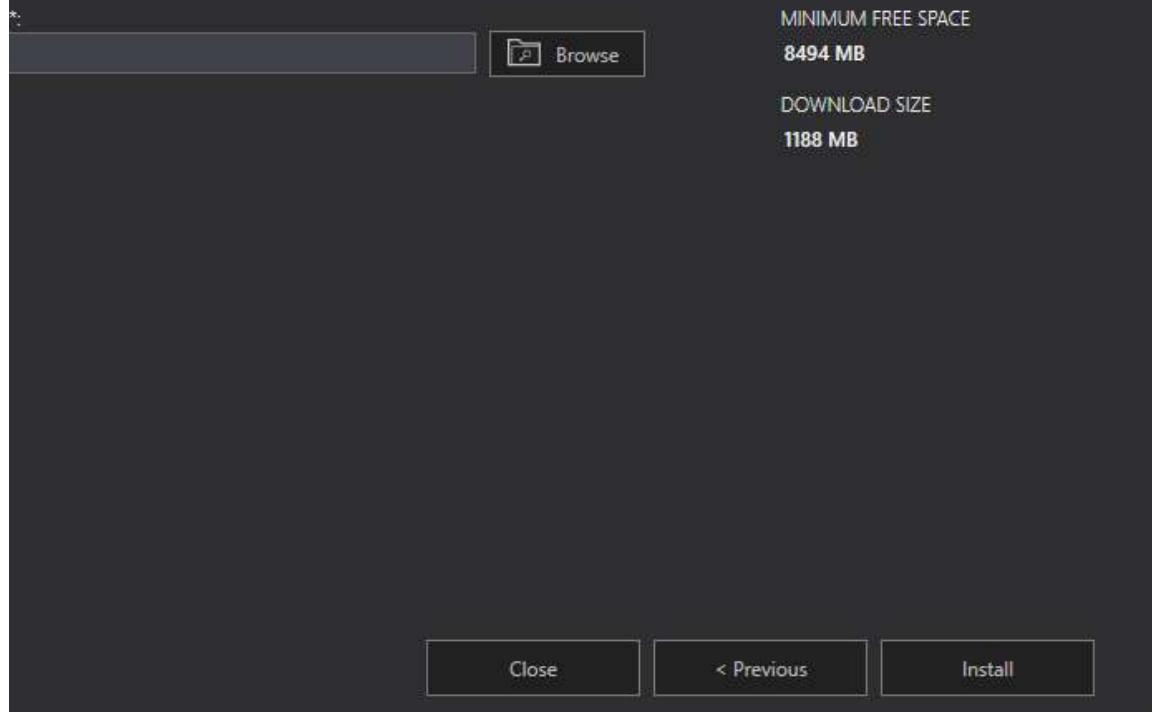
شوف انت عايز تنزله فين ونزله

ver 2022

😊 – ✕

# Developer Edition

SQL Server media download target location



rver 2022



-

X

# Developer Edition

Reading install package...

Uploading files... 4.353 MB / 1,176.702 MB 22.395 Mbps

is also available for Linux

For more information about Microsoft Server 2022 Linux images, including Containers, please see here (<https://go.microsoft.com/fwlink/?linkid=2197262>).

Pause

Cancel

## Installation Center

Server 2022

-  [Hardware and Software Requirements](#)  
View the hardware and software requirements.
-  [Security Documentation](#)  
View the security documentation.
-  [Online Release Notes](#)  
View the latest information about the release.
-  [Azure extension for SQL Server \(New\)](#)  
Azure extension for SQL Server enables Microsoft Defender for Cloud, Purview, Azure Active Directory and other Azure services.
-  [System Configuration Checker](#)  
Launch a tool to check for conditions that prevent a successful SQL Server installation.
-  [Download Data Migration Assistant \(DMA\)](#)  
Data Migration Assistant (DMA) analyzes SQL Server components that are installed and identifies issues to fix either before or after you upgrade to SQL Server 2022.
-  [Online Installation Help](#)  
Launch the online installation documentation.
-  [How to Get Started with SQL Server 2022 Failover Clustering](#)  
Read instructions on how to get started with SQL Server 2022 failover clustering.
-  [Upgrade Documentation](#)  
View the document about how to upgrade to SQL Server 2022 from a previous version of SQL Server.
-  [Download SQL Server Migration Assistant \(SSMA\)](#)

[New SQL Server standalone installation or add features to an existing installation](#)

Launch a wizard to install SQL Server 2022 in a non-clustered environment or to add features to an existing SQL Server 2022 instance.

[Install SQL Server Reporting Services](#)

Launch a download page that provides a link to install SQL Server Reporting Services. An internet connection is required to install SSRS.

[Install SQL Server Management Tools](#)

Launch a download page that provides a link to install SQL Server Management Studio, SQL Server command-line utilities (SQLCMD and BCP), SQL Server PowerShell provider, SQL Server Profiler and Database Tuning Advisor. An internet connection is required to install these tools.

[Install SQL Server Data Tools](#)

Launch a download page that provides a link to install SQL Server Data Tools (SSDT). SSDT provides Visual Studio integration including project system support for Microsoft Azure SQL Database, the SQL Server Database Engine, Reporting Services, Analysis Services and Integration Services. An internet connection is required to install SSDT.

[New SQL Server failover cluster installation](#)

Launch a wizard to install a single-node SQL Server 2022 failover cluster. This action is only available in the clustered environment.

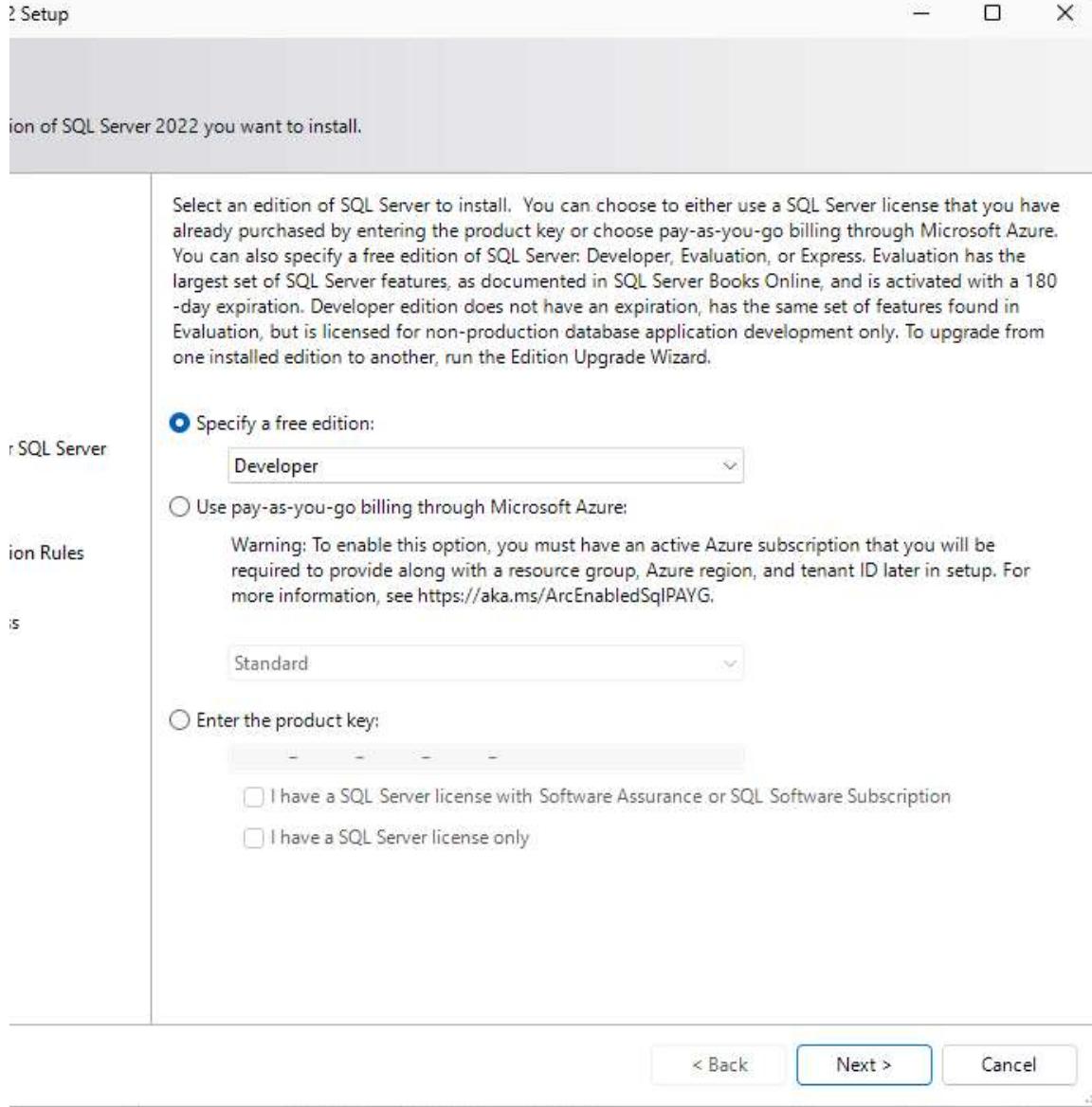
[Add node to a SQL Server failover cluster](#)

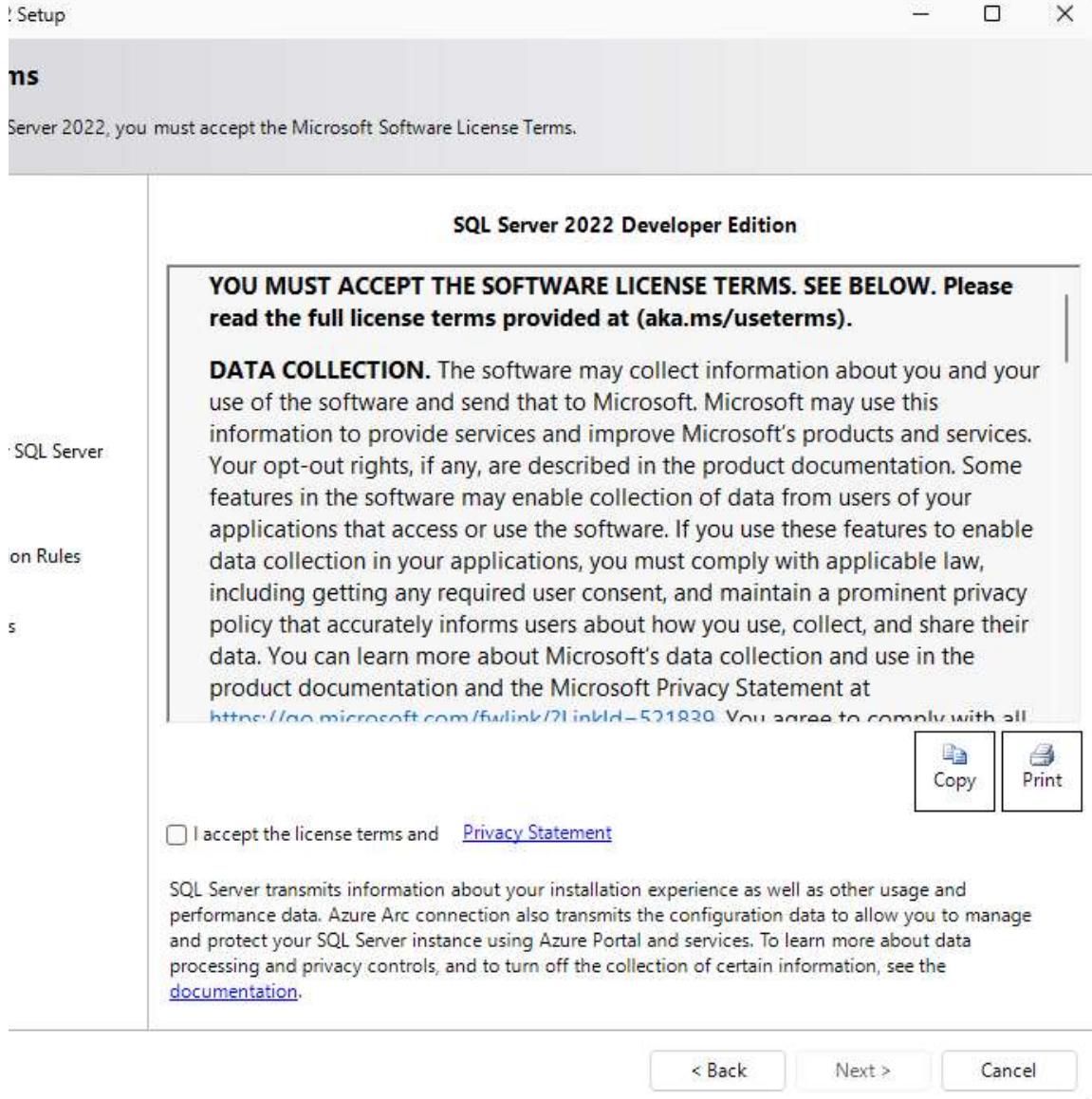
Launch a wizard to add a node to an existing SQL Server 2022 failover cluster. This action is only available in the clustered environment.

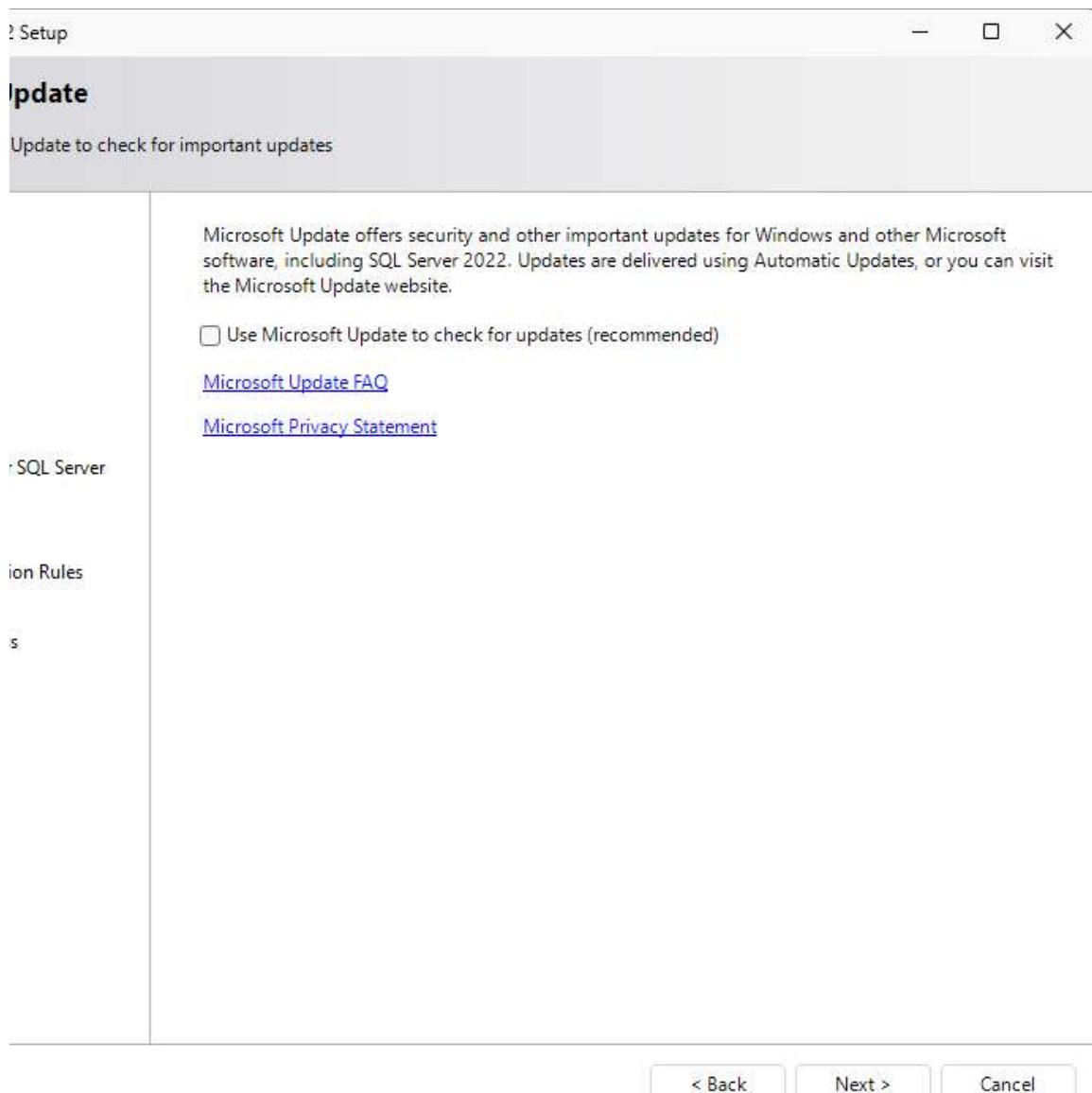
[Upgrade from a previous version of SQL Server](#)

Launch a wizard to upgrade a previous version of SQL Server to SQL Server 2022.

[Click here to first view Upgrade Documentation](#)







شيلها مش عايزيتها

? Setup

## vision for SQL Server

on for SQL Server is required to enable Microsoft Defender for Cloud, Purview, and Azure Active Directory.

or SQL Serv...

ion Rules

s

□ Azure Extension for SQL Server To install Azure extension for SQL Server, provide your Azure account or a service principal to authenticate the SQL Server instance to Azure. You also need to provide the Subscription ID, Resource Group, Region, and Tenant ID where this instance will be registered. For more information for each parameter, visit <https://aka.ms/arc-sql-server>.

Use Azure Login

Use Service Principal

Azure Service Principal ID\*

Azure Service Principal Secret\*

Azure Subscription ID\*

Azure Resource Group\*

Azure Region\*

Azure Tenant ID\*

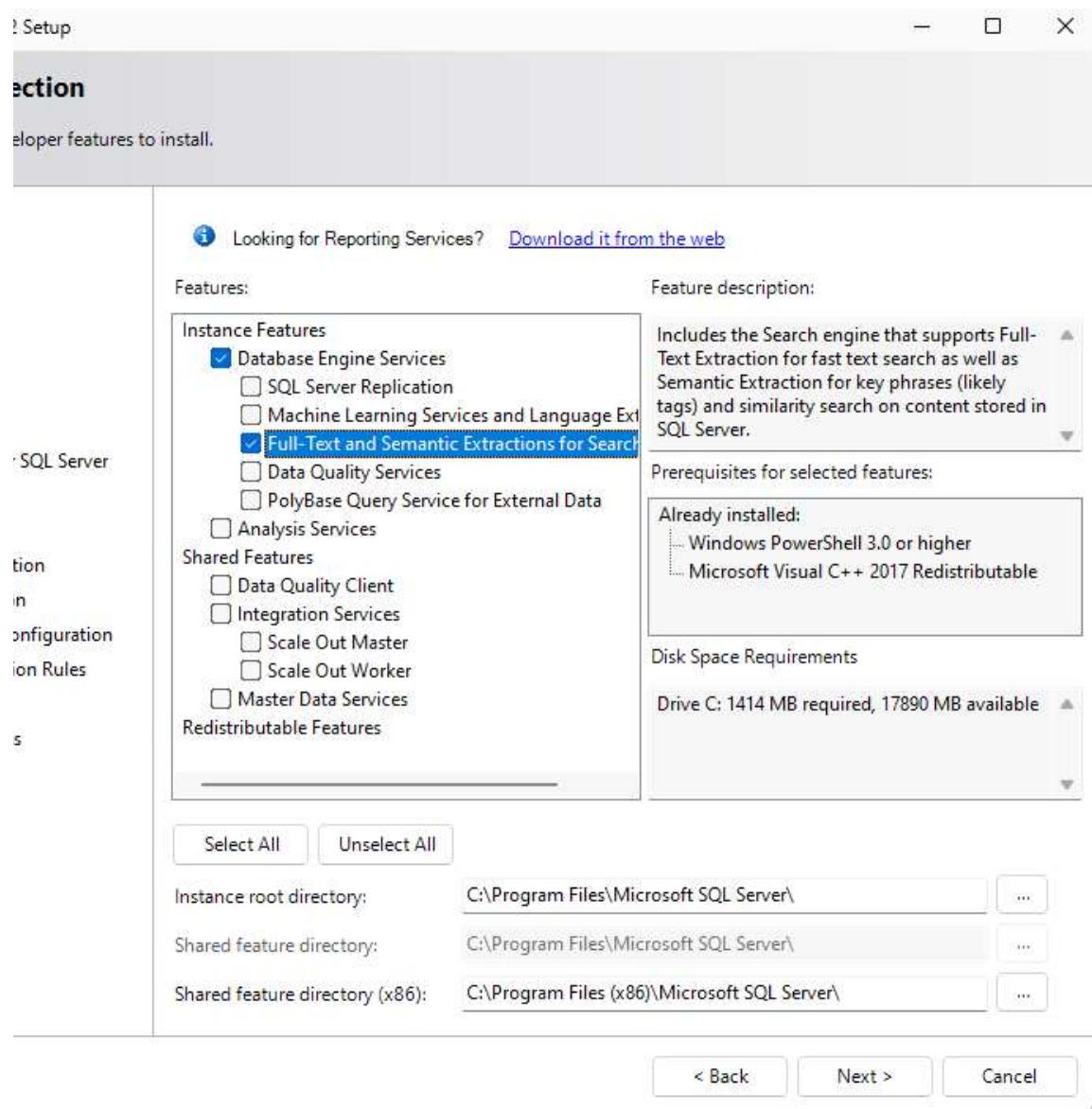
Proxy Server URL (optional)

< Back

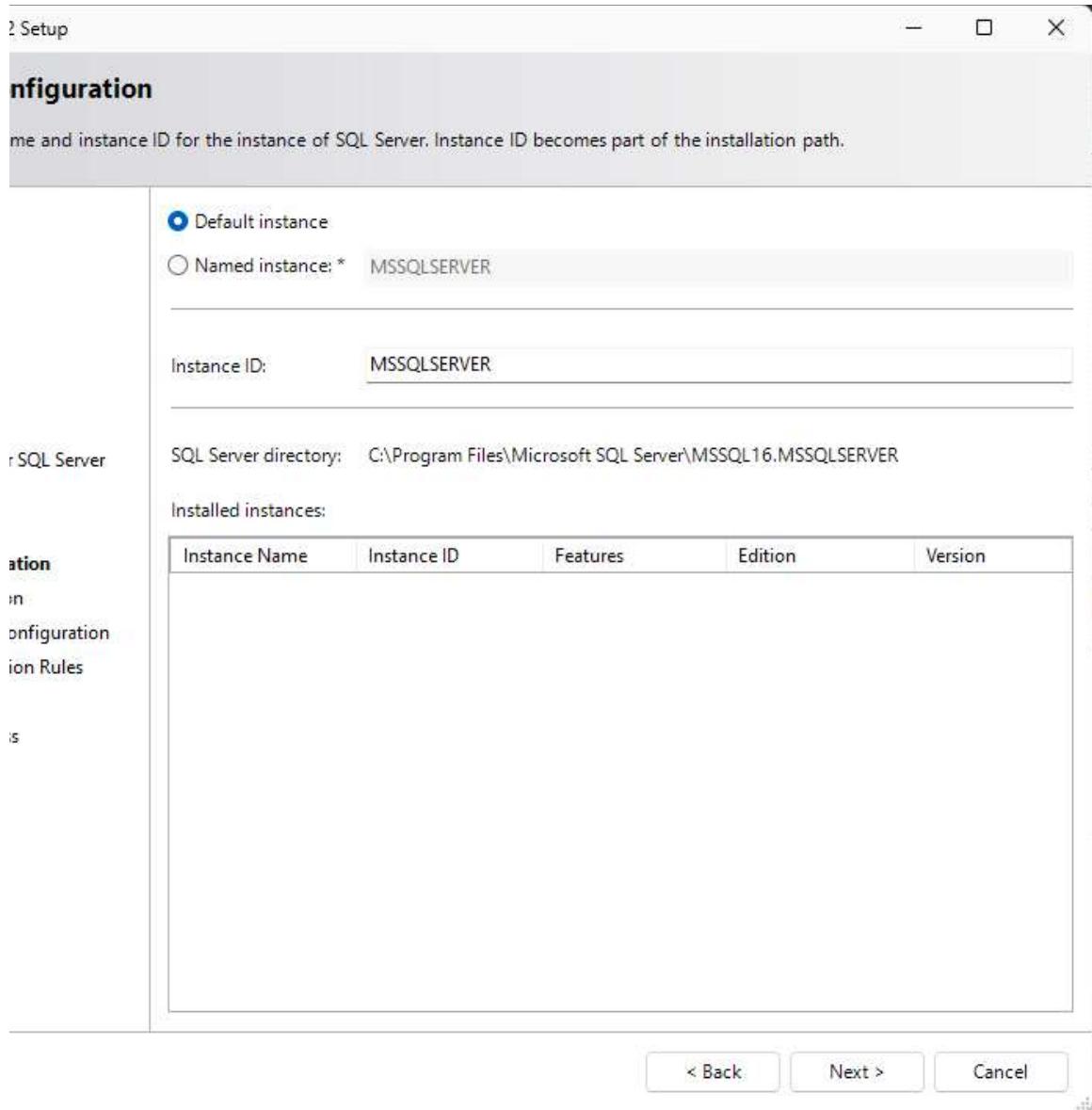
Next >

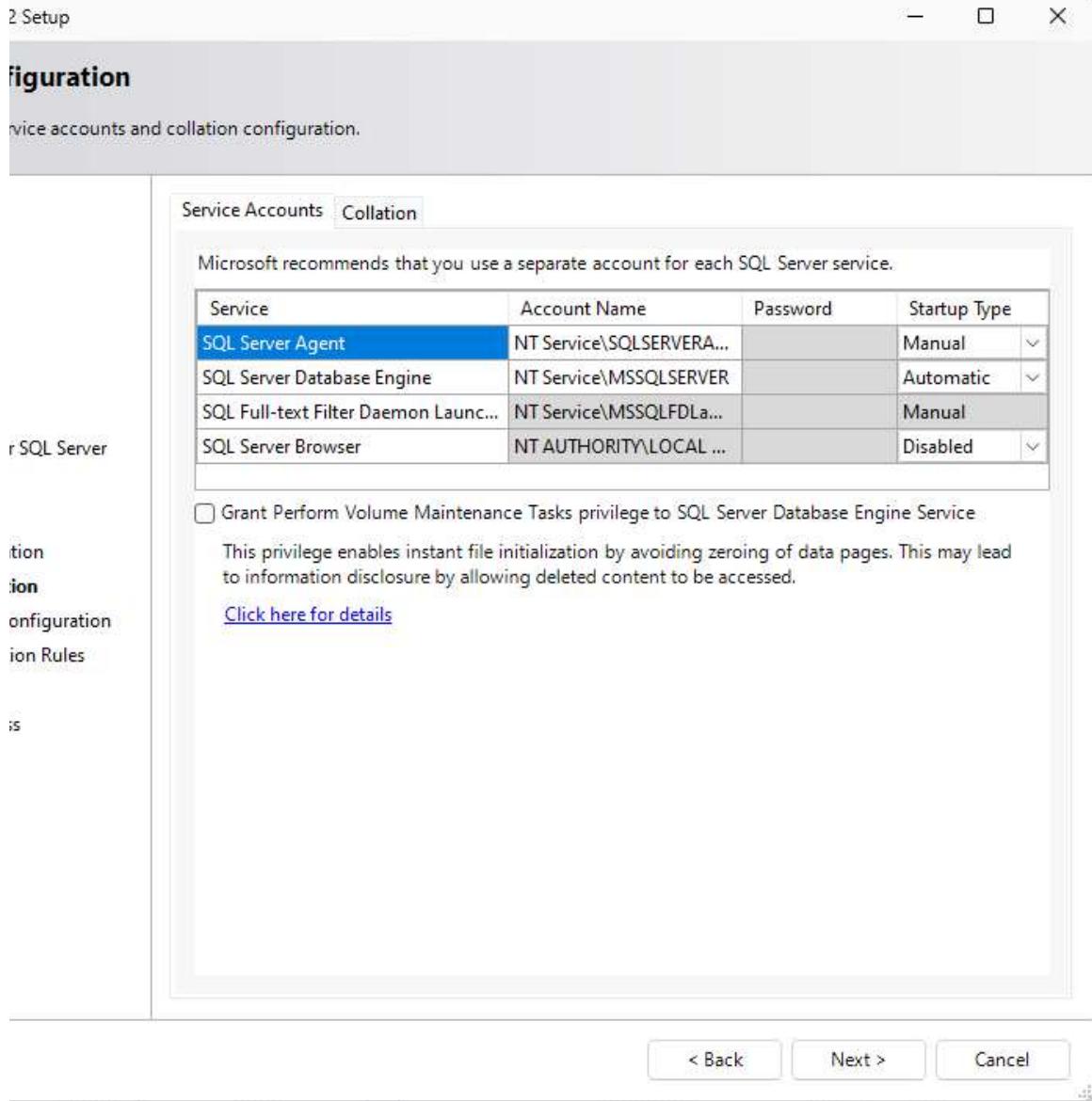
Cancel

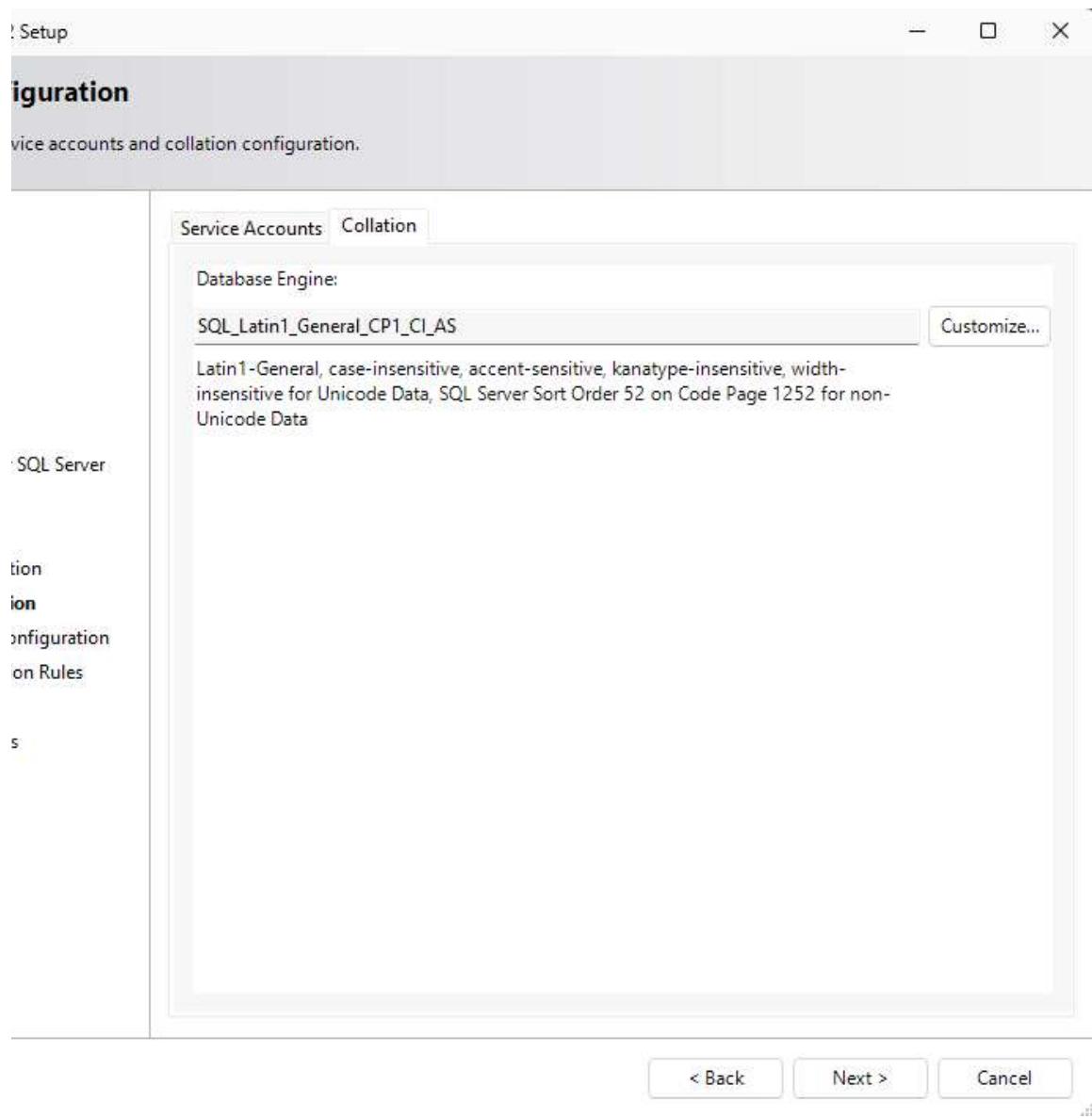
عایزین دی بس



ماتلعيش فيه







## Server 2022 Database Engine Collation

I you would like to use:

on designator and sort order

inator:

Albanian

- Binary-code point
- Kana-sensitive
- Width-sensitive
- Variation selector-sensitive

### Char/Varchar Storage Options

Windows Code Page (1250)

UTF-8

used for backwards compatibility

f\_CP1\_CI\_AS

al\_CI\_CI\_AI

al\_CI\_CI\_AS

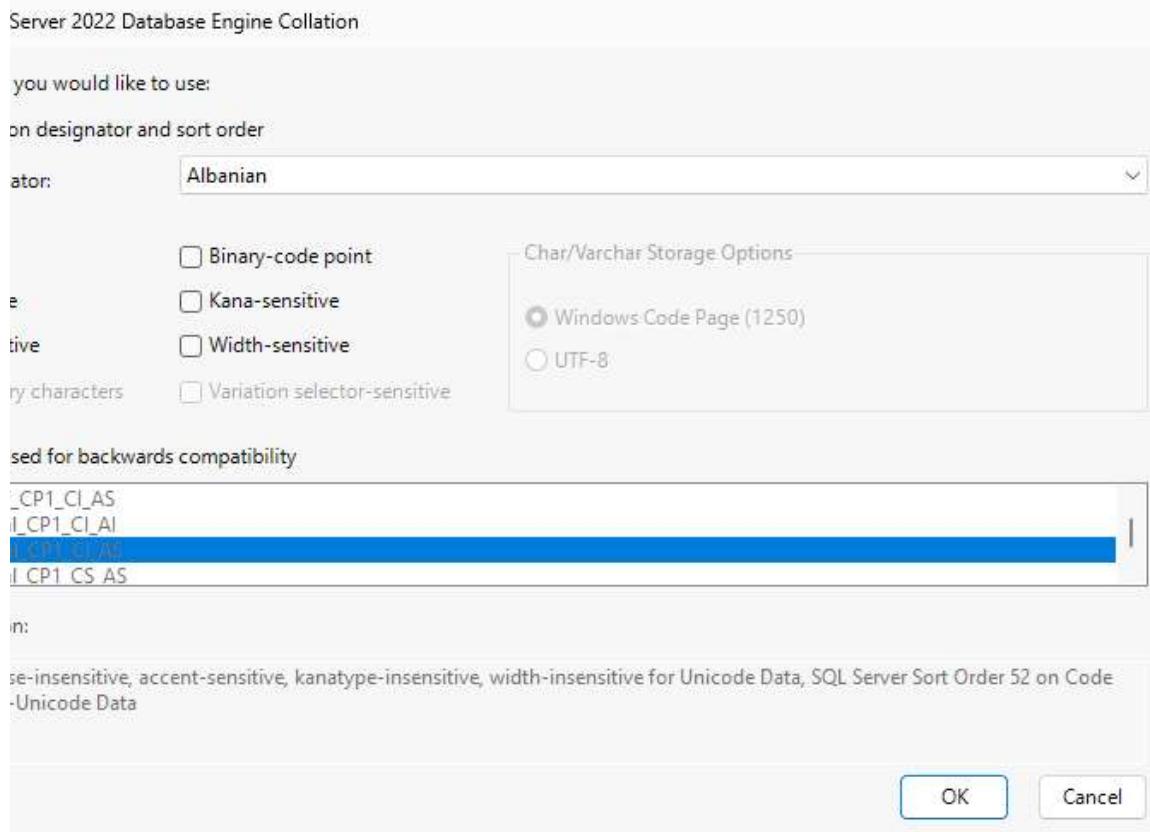
al\_CI\_CS\_AS

on:

case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive for Unicode Data, SQL Server Sort Order 52 on Code-Page-Unicode Data

OK

Cancel



## Server 2022 Database Engine Collation

you would like to use:

on desinator and sort order

ator: Arabic

- Binary-code point
- Kana-sensitive
- Width-sensitive
- Variation selector-sensitive

### Char/Varchar Storage Options

- Windows Code Page (1256)
- UTF-8

sed for backwards compatibility

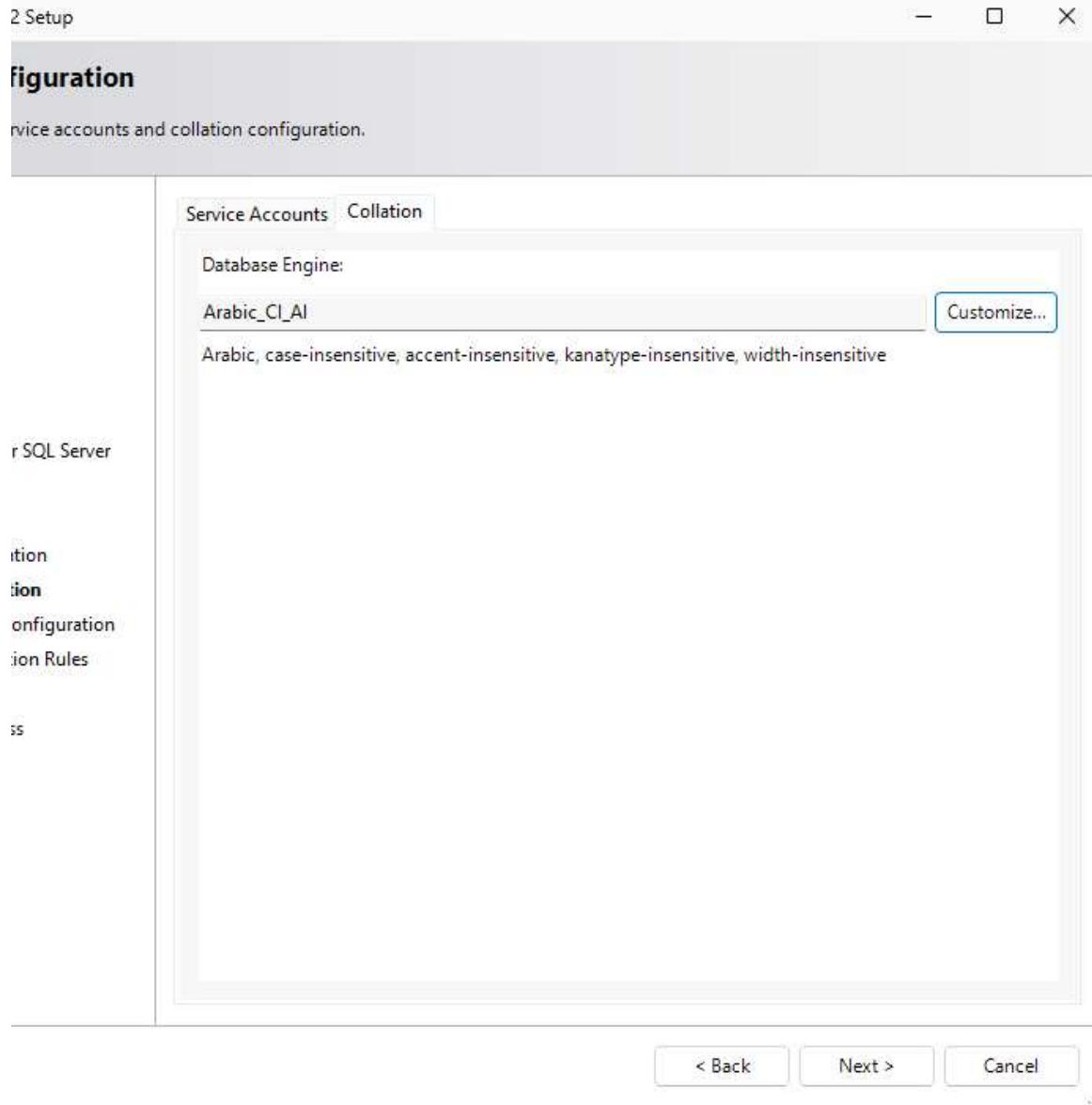
\_CP1\_CI\_AS  
\_CP1\_CI\_AI  
\_CP1\_CS\_AS  
| CP1\_CS\_AS

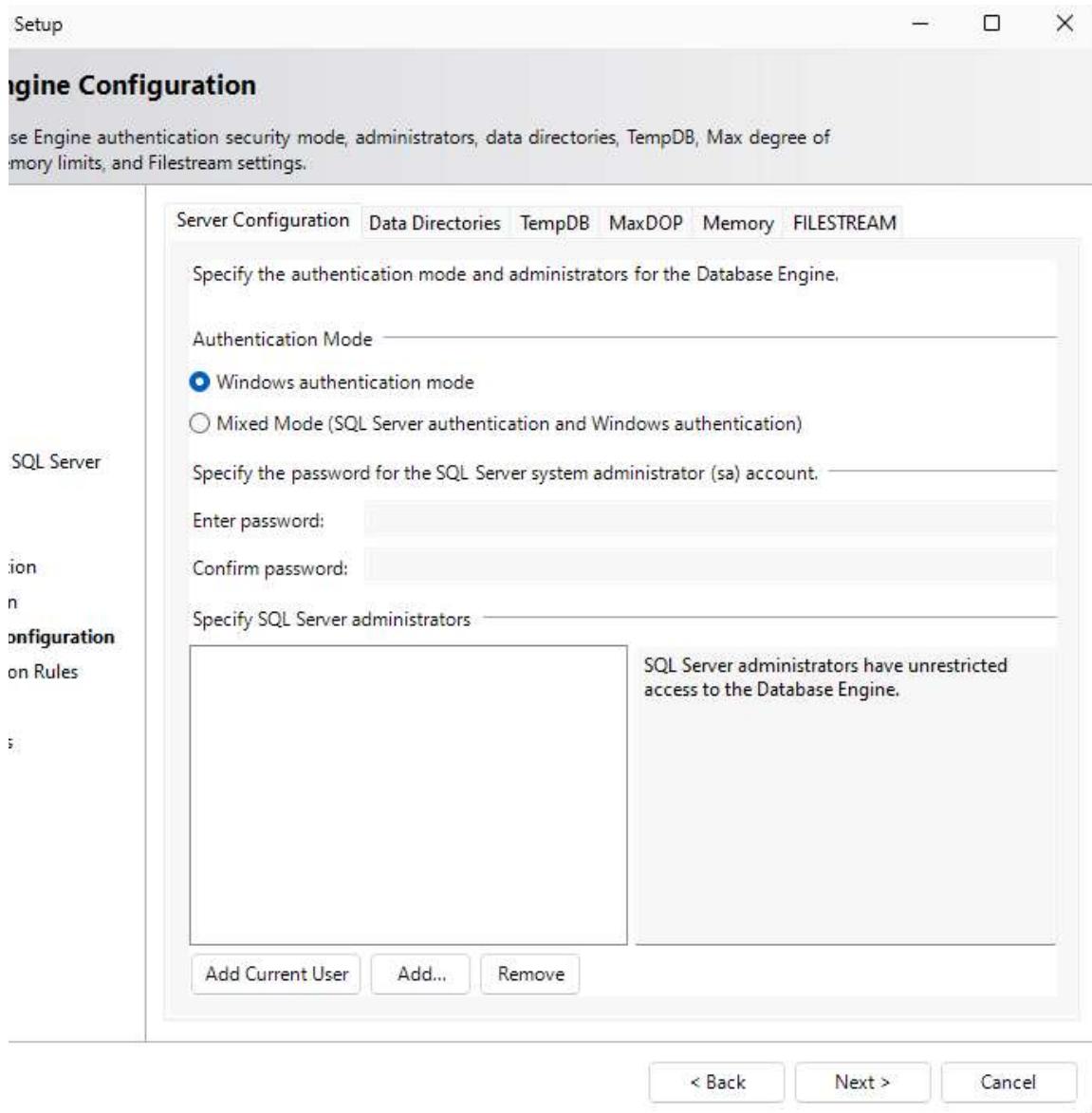
n:

case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive for Unicode Data, SQL Server Sort Order 52 on Code Page 1256 for Latin1\_General\_CI\_AS.

OK

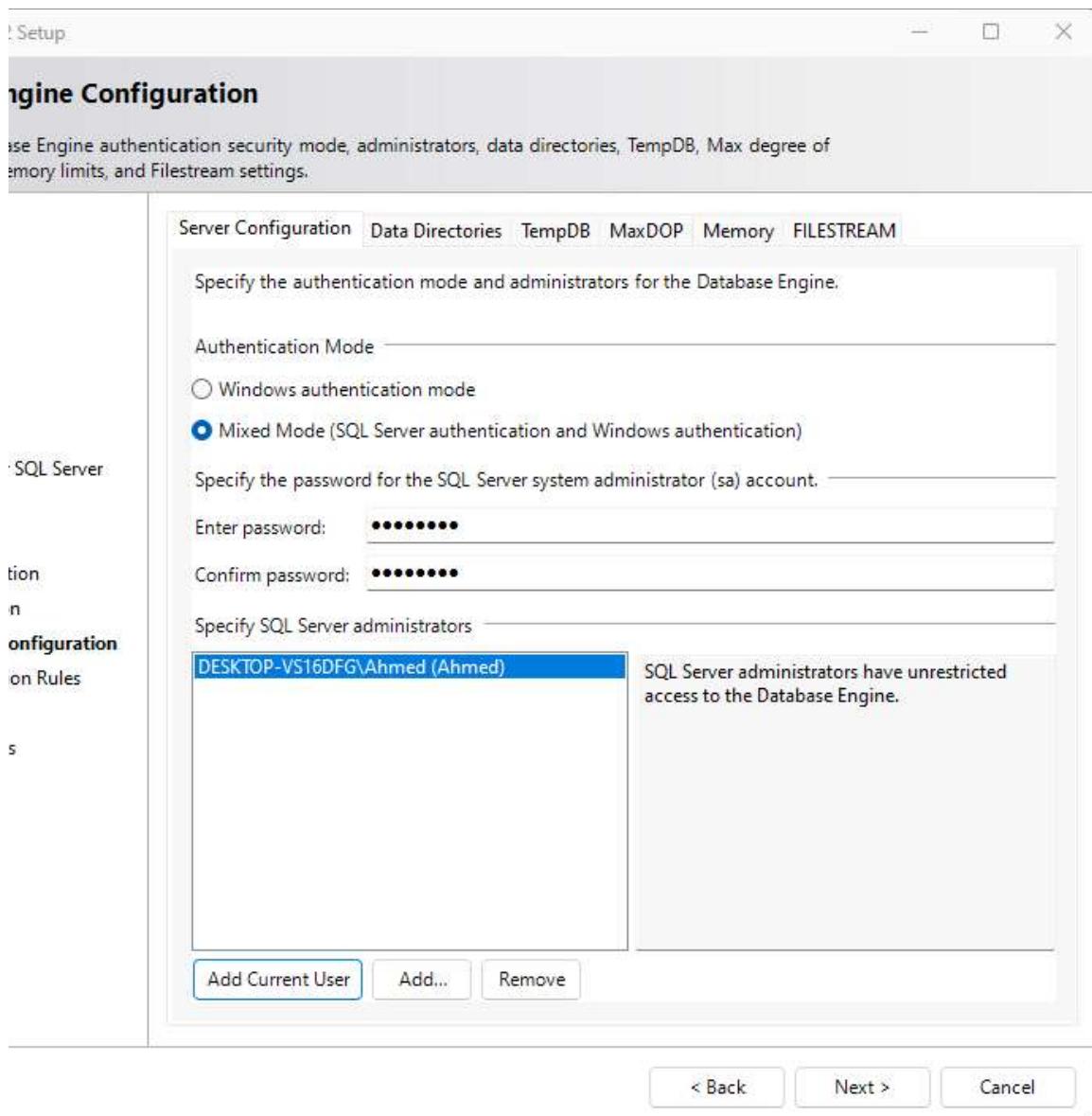
Cancel

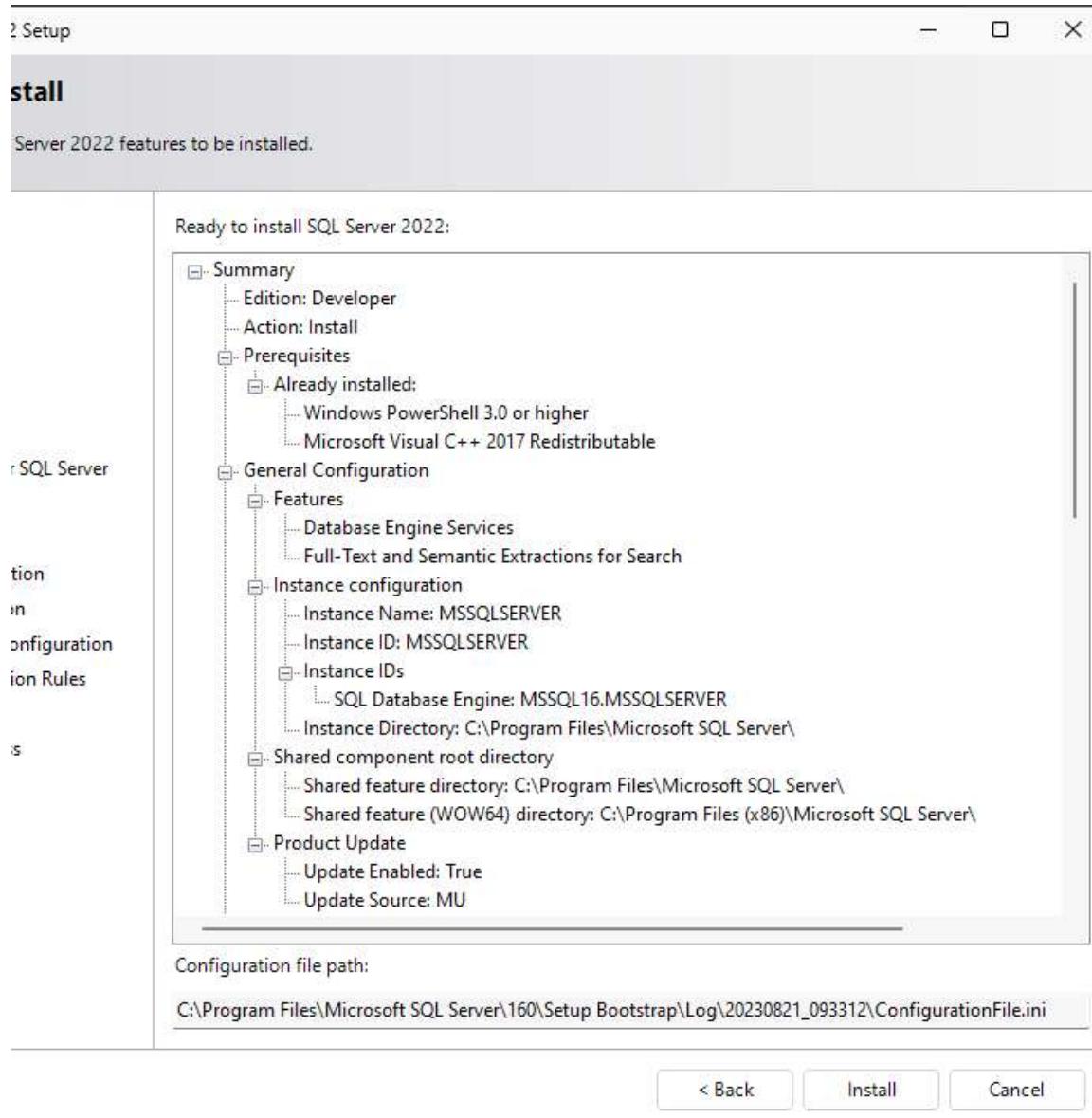




هوا دلوقتي هينزلك ال sql server من البرنامج بتاعك ففيه طريقتين عشان يشتغل  
اول واحد عن طريق اليوزر والباسورد بتوع الويندوز ودي اسمها windows authentication

والطريقه الثانيه عن طريق يوزر وباسورد خاصين بيک  
فالحنا هنختار mixed mode عشان يدعم الاتنين  
بعدين في الباسورد خليه sa123456  
وبعدين بتدوس علي add current user واصبر عليه هيأخذ وقت





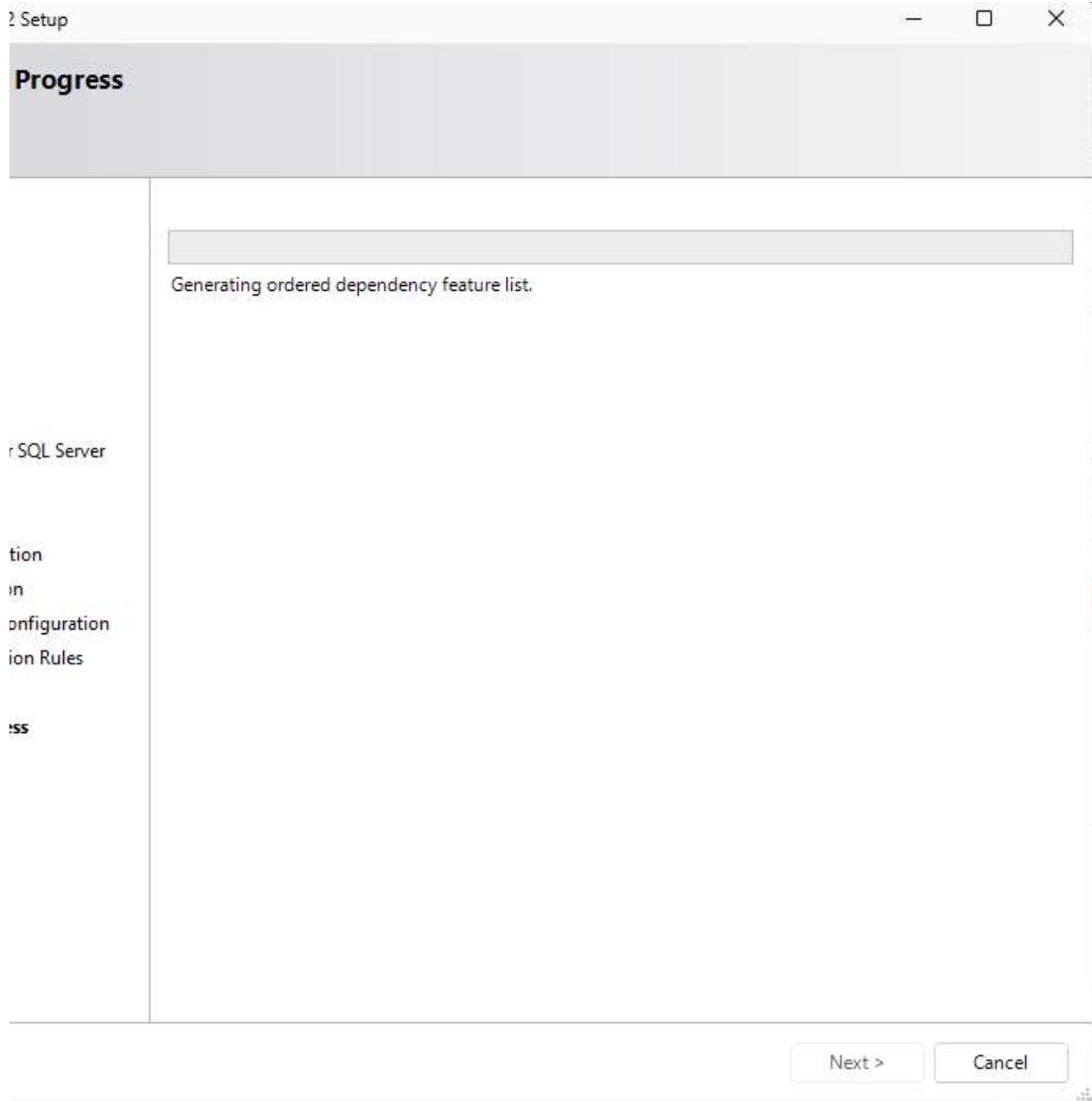
Configuration file path:

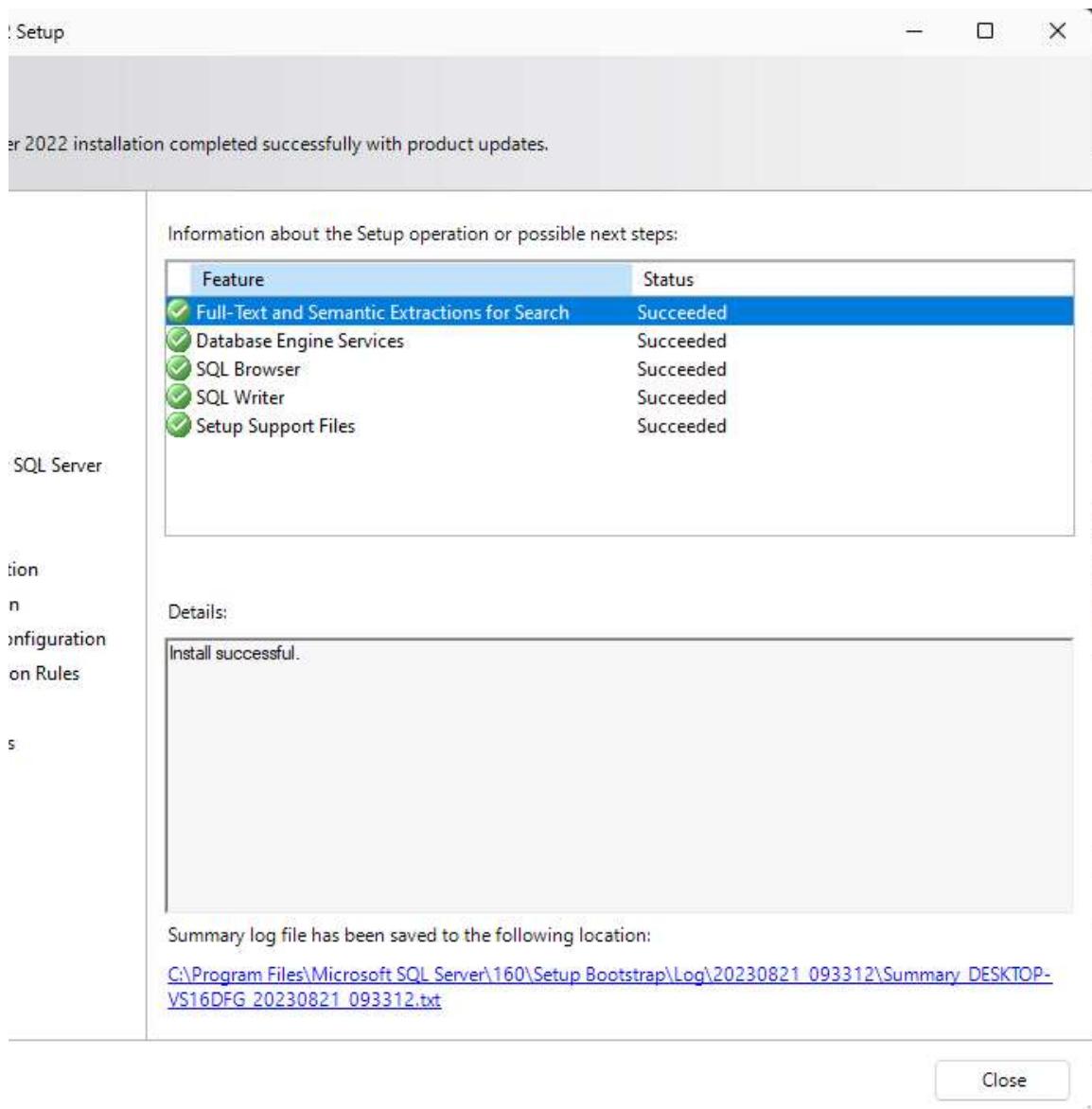
C:\Program Files\Microsoft SQL Server\160\Setup Bootstrap\Log\20230821\_093312\ConfigurationFile.ini

< Back

Install

Cancel





كده نزلنا ال engine هنزل ال server management studio واللي من خلاله هتقدر تعمل الداتا بيز بتاعتك

[New SQL Server standalone installation or add features to an existing installation](#)

Launch a wizard to install SQL Server 2022 in a non-clustered environment or to add features to an existing SQL Server 2022 instance.

[Install SQL Server Reporting Services](#)

Launch a download page that provides a link to install SQL Server Reporting Services. An internet connection is required to install SSRS.

[Install SQL Server Management Tools](#)

Launch a download page that provides a link to install SQL Server Management Studio, SQL Server command-line utilities (SQLCMD and BCP), SQL Server PowerShell provider, SQL Server Profiler and Database Tuning Advisor. An internet connection is required to install these tools.

[Install SQL Server Data Tools](#)

Launch a download page that provides a link to install SQL Server Data Tools (SSDT). SSDT provides Visual Studio integration including project system support for Microsoft Azure SQL Database, the SQL Server Database Engine, Reporting Services, Analysis Services and Integration Services. An internet connection is required to install SSDT.

[New SQL Server failover cluster installation](#)

Launch a wizard to install a single-node SQL Server 2022 failover cluster. This action is only available in the clustered environment.

[Add node to a SQL Server failover cluster](#)

Launch a wizard to add a node to an existing SQL Server 2022 failover cluster. This action is only available in the clustered environment.

[Upgrade from a previous version of SQL Server](#)

Launch a wizard to upgrade a previous version of SQL Server to SQL Server 2022.

[Click here to first view Upgrade Documentation](#)

The screenshot shows the Microsoft Learn website for SQL Server. The top navigation bar includes links for Documentation, Training, Certifications, Q&A, Code Samples, Assessments, Shows, Events, Search, and Sign in. Below the main content area, there are dropdown menus for Secure, Develop, Administer, Analyze, Reference, and Resources. A button for Azure Portal and another for Download SQL Server are also present. The main content area displays the title "Download SQL Server Management Studio (SSMS)" and an article summary. The article was published on 08/10/2023 by 49 contributors. It includes sections for "In this article" (with links to Download SSMS, Available languages, What's new, Previous versions, and Show 8 more), "Applies to" (listing SQL Server, Azure SQL Database, Azure SQL Managed Instance, Azure Synapse Analytics, SQL Endpoint in Microsoft Fabric, and Warehouse in Microsoft Fabric), and descriptions of SSMS's capabilities and cross-platform support. A large "Download SSMS" button is prominently displayed at the bottom.

The screenshot shows the Microsoft Download for SQL Server Management Studio (SSMS) 19.1 page. The title is "Download for SQL Server Management Studio (SSMS) 19.1". The page states that it is the latest general availability (GA) version. It advises users to uninstall any previous version before installing SSMS 19.1. The download link is provided for the 64-bit version. The page also lists the download number (19.1), the file number (19.1.56.0), and the release date (May 24, 2023). A note at the bottom indicates that by downloading, users agree to the license terms and privacy statement, and provides contact information for the SSMS team.



RELEASE 19.1



# Microsoft SQL Server Management Studio with Azure Data Studio

ing packages. Please wait...

Cancel

RELEASE 19.1

# Microsoft SQL Server Management Studio with Azure Data Studio

Setup Completed

The specified components have been installed successfully.

[Close](#)

Installation Center

— □ ×

New SQL Server standalone installation or add features to an existing installation  
Launch a wizard to install SQL Server 2022 in a non-clustered environment or to add features to an existing SQL Server 2022 instance.

Install SQL Server Reporting Services  
Launch a download page that provides a link to install SQL Server Reporting Services. An internet connection is required to install SSRS.

Install SQL Server Management Tools  
Launch a download page that provides a link to install SQL Server Management Studio, SQL Server command-line utilities (SQLCMD and BCP), SQL Server PowerShell provider, SQL Server Profiler and Database Tuning Advisor. An internet connection is required to install these tools.

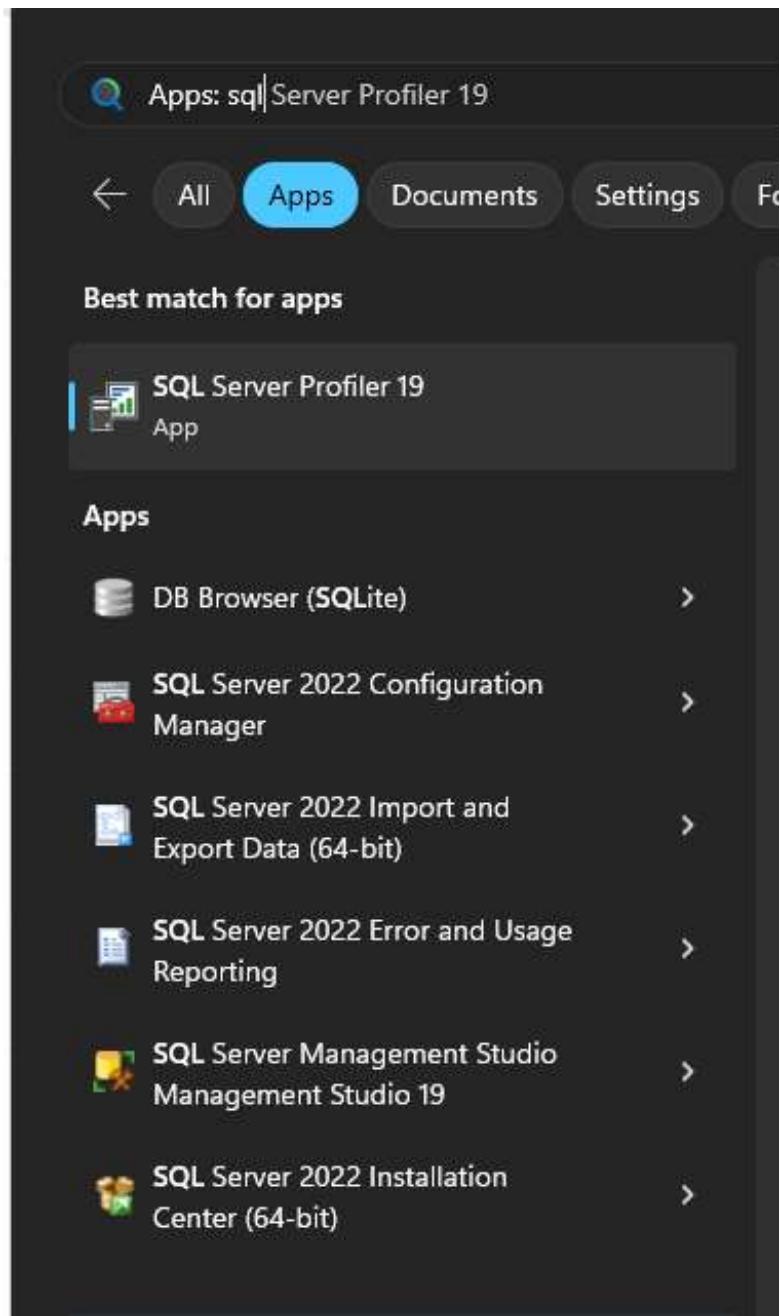
Install SQL Server Data Tools  
Launch a download page that provides a link to install SQL Server Data Tools (SSDT). SSDT provides Visual Studio integration including project system support for Microsoft Azure SQL Database, the SQL Server Database Engine, Reporting Services, Analysis Services and Integration Services. An internet connection is required to install SSDT.

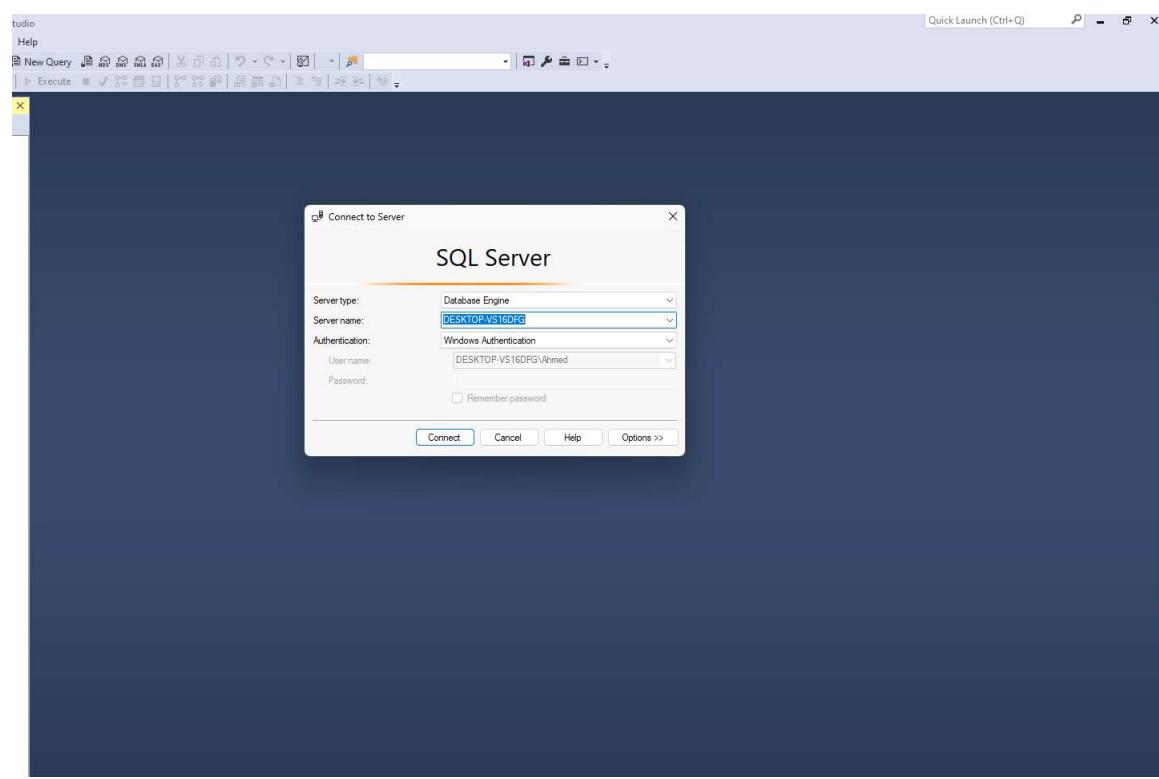
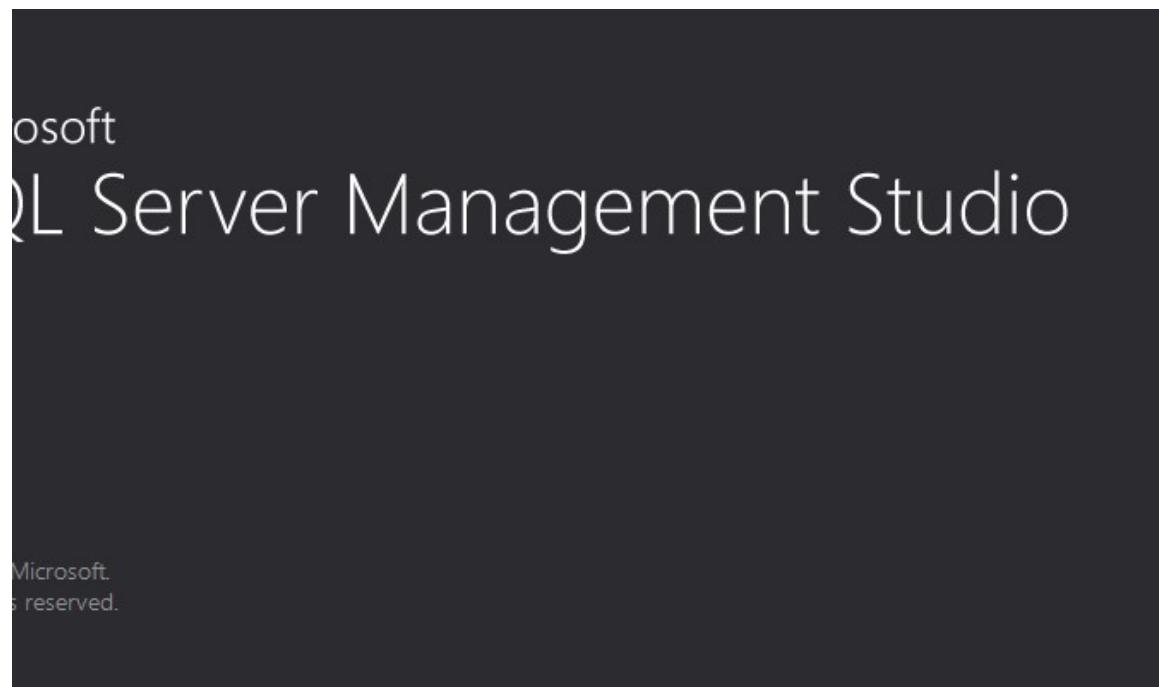
New SQL Server failover cluster installation  
Launch a wizard to install a single-node SQL Server 2022 failover cluster. This action is only available in the clustered environment.

Add node to a SQL Server failover cluster  
Launch a wizard to add a node to an existing SQL Server 2022 failover cluster. This action is only available in the clustered environment.

Upgrade from a previous version of SQL Server  
Launch a wizard to upgrade a previous version of SQL Server to SQL Server 2022.  
[Click here to first view Upgrade Documentation](#)

Server 2022





دلوقي احنا عاوزين نعمل connection عال sql server اللي موجود عندنا  
وتقدر تدخل بالليوزر بتاع الوييندرز او بالليوزر والباسورد اللي حطيتهم

ct to Server



# SQL Server

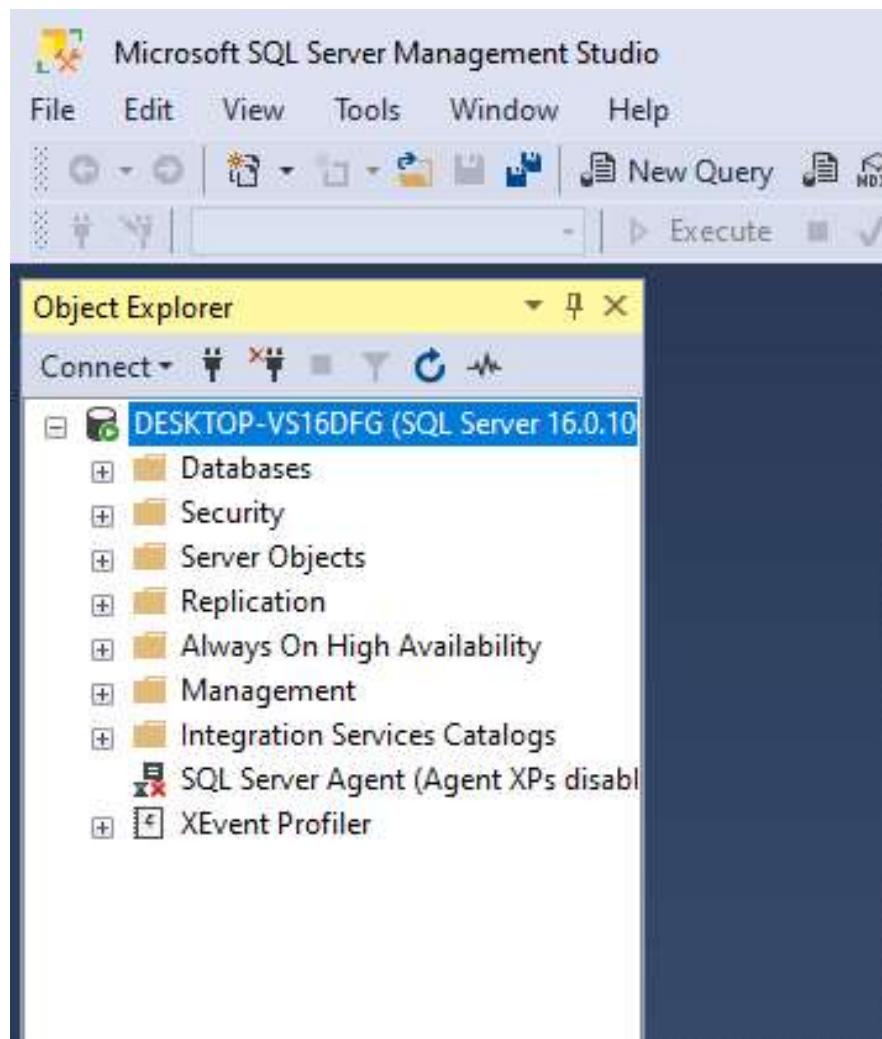
Type:	Database Engine
Server:	DESKTOP-VS16DFG
Authentication:	Windows Authentication
Username:	DESKTOP-VS16DFG\Ahmed
	Remember password <input type="checkbox"/>

Connect

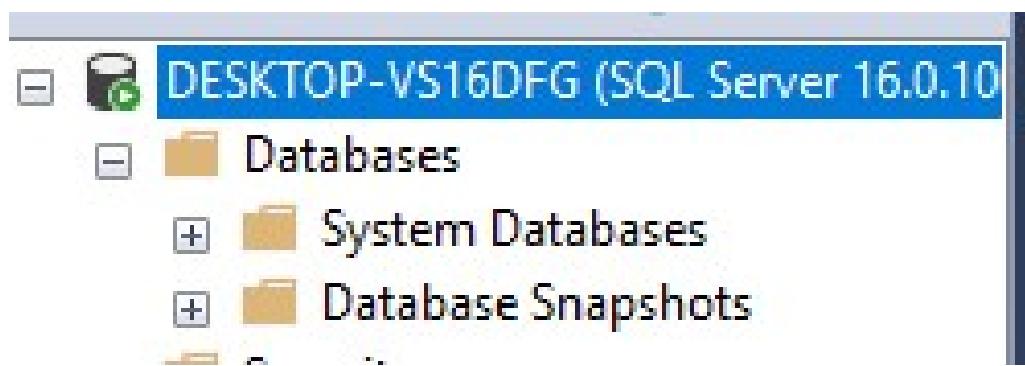
Cancel

Help

Options >>



اوی تلعب في دي



## Differences between Data, Information , Knowledge and

## Wisdom

يجب مراجعة هذا الدرس الذي اخذناه في سلسلة اساسيات مهمة لكل مبرمج



<https://youtu.be/qTVRs1wTwho>

## Differences between Data Structure and Database

يجب مراجعة هذا الدرس الذي اخذناه في كورس

Data Structure

<https://programmingadvices.com/courses/database-level-1-sql-concepts-and-practice/lectures/46066083>

## What is Database?

نتعرف على كل الأساسيات المشتركة بين كل ال databases والفرق بين كل داتا بيز والثانويه بتكون بسيطه

ماتفوتش ولا درس وافهم كل حاجه كويش

الداتا بيز هيا مجموعه من البيانات او الحقائق المنظمه عشان تقدر توصلها بسهوله  
وعشان تدير الداتا بيز لازم تستخدم DBMS وهيا اختصار ل  
- management system

- DBMS Non relational :- البيانات موجوده و بتخزن بس مفيش علاقه  
بينها زي ماكنا بنعمل ملف TEXT وبرضه زي ال XML والملفات

- RDBMS Relational :- زي ال ORACLE وال SQL SERVER وهيا عباره  
عن داتا فيه علاقات بينها وبين بعضها

## What is Database?

A database is an organized collection of data so that it can be easily managed. To manage these databases, Database Management Systems (DBMS) are used.

DBMS:

In general, there are two common types of databases:

Non-relational (DBMS): File System, XML..etc.

Relational (RDBMS): enhanced version of DBMS but with relations, examples include MySQL, Oracle, PostgreSQL, Microsoft SQL Server, MySQL ..etc.

وده مثال عال FILE SYSTEM زي مشروع البنك كده

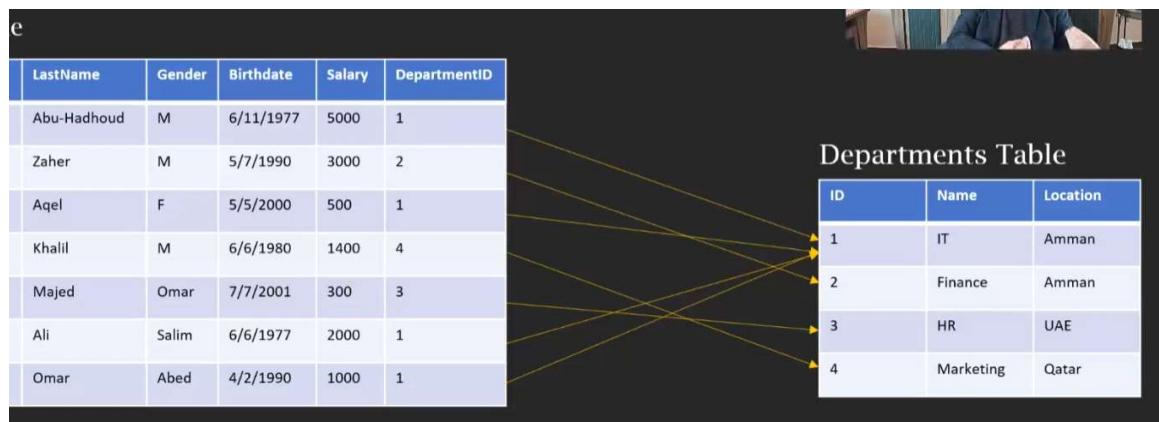
Name	Last Name	Gender	Birthdate	Salary	DeptID	DeptName	DeptLocation
Immed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
	Zaher	M	5/7/1990	3000	2	Finance	Amman
	Aqel	F	5/5/2000	500	1	IT	Amman
	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
	Majed	M	7/7/2001	300	3	HR	UAE
	Ali	M	6/6/1977	2000	1	IT	Amman
	Omar	F	4/2/1990	1000	1	IT	Amman

التعامل مع ال files صعب وفيه مشاكل زي ما شوفنا قبل كده ولما تيجي تدخل في الكورس هتعرف قيمة الداتا بيز

والداتا بيز بيتحط الداتا في جداول ويتعمل علاقات بين الجداول دي زي مثلا عندك جدول للموظفين ومن ضمن البيانات بتاعت الموظفين هيا البيانات بتاعت الأقسام اللي هما فيها ف ليه تحط بيانات الأقسام معاهم وانت ممكن تحطهم في جدول منفصل وترتبط بين الجدولين برقم بيمثل القسم

## DBMS .

record in an organized fashion in tables containing rows and columns along with relations between

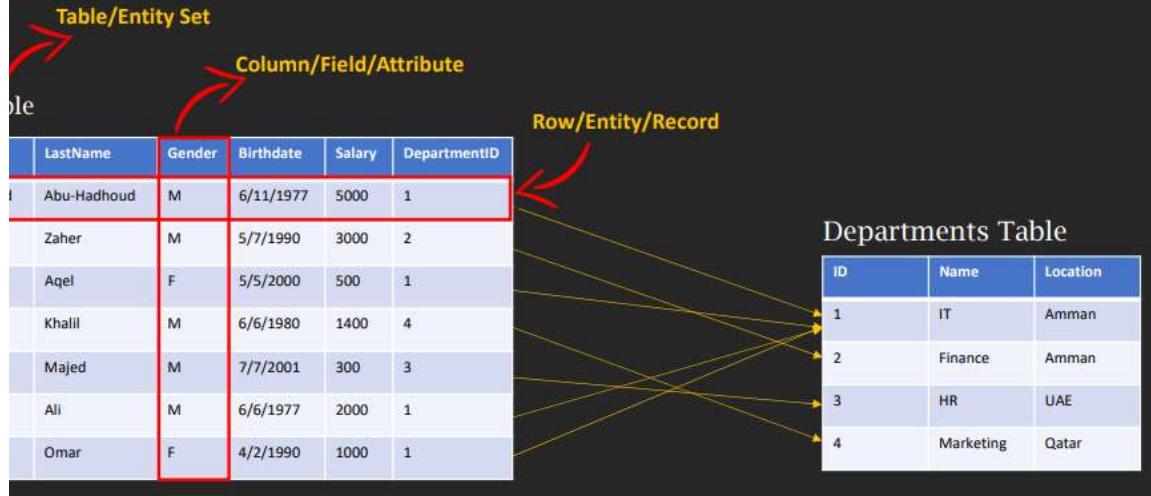


الصف في الجدول بيسمه row او record او entity

العمود في الجدول اسمه column او field او attribute  
 الجدول كله علي بعضه اسمه entity set او table

## BMS .

stored in an organized fashion in tables containing rows and columns along with relations between



**الواجب**

DBMS and RDBMS are the same.

True

False

DBMS stands for Database Management Systems

True

False

DBMS has relations between Data

True

False

RDBMS has relations between data.

True

False

RDBMS stands for Relational Database Management Systems .

True

False

DBMS has two types: None Relational Database and Relational Database.

True

False

File System and XML are samples of RDBMS.

True

False

File System, XML are samples of DBMS.

True

False

SQL Server , Oracle , MySQL are examples of RDBMS.

True

False

Dealing with RDBMS is much easier than dealing with DBMS.

True

False

In RDBMS: Data that is stored in an organized fashion in tables containing rows and columns along with relations between these tables.

True

False

Column/Field/Attribute are the same.

True

False

Row/Entity/Record are the same

True

False

Table/Entity Set are the same.

True

False

Relationships are represented using references to data from other tables.



### What is NULL?

ساعات بيكون فيه معلومات مش متوفره ومش عايز اعملها initial value عشان امشي الدنيا فبروح للعمود او ال field وبقوله allow null يعني اسمح بانه يكون في العمود ده قيم null وهيا معناها nothing يعني مفيش داتا يعني ماتجييش انه null معناها صفر او مسافة

وكمان بيعرفوا انها distinct value يعني قيمه مميزة ومهما استخدامها بحد رلانه بتتأثر على ال queries والعمليات الحسابيه في الداتا بيز



## is Null

database, "NULL" is a special marker used to indicate that a data does not exist in the database. It represents a missing or unknown a table column.

ield or column in a table has a NULL value, it means that the field value at all, and it's different from having an empty or zero value. not the same as a blank space or a zero, and it's a distinct value atabase.

be used in several ways, such as indicating missing data, ting optional fields, or as a placeholder for values that are not n. However, it's essential to use NULL values carefully because affect the results of queries and calculations in unexpected ways.

## ary

ry, NULL is a special value in a database that represents the absence of a value in a table column.

l to indicate missing or unknown data and should be used carefully to avoid unexpected results in queries and calculations.

**الواجب**

NULL is a special value in a database that represents the absence of a value in a table column.

True

False

NULL is used to indicate missing or unknown data and should be used carefully to avoid unexpected results in queries and calculations.

True

False

When a salary field in a table has a NULL value it means its value is zero

True

False

When a salary field in a table has a NULL value it means its value is missing or unknown.

True

False

When a field in a table has a NULL value it means its optional.

True

False

NULL is not the same as a blank space or a zero.

True

False

When a NumberOfChildren field in Employees table has a NULL value  
this means that employee has zero children.

True

False

When a NumberOfChildren field in Employees table has a NULL value  
this means that we don't know yet that value, it's missing or unknown to  
us.

True

False

### **Primary Key vs Foreign Key / Referential Integrity**

ال primary key وال foreign key بيتم استخدامهم في الربط بين الجداول في  
البيانات.

عشان يكون عندي relations بين الجداول لازم اقدر اربطهم مع بعض وعشان اقدر  
اربطهم مع بعض لازم يكون عندي field من ضمن ال fields بتاعت الجدول تكون  
القيم اللي فيه مش متكرره

فلو بصيت لجدول الموظفين تحت هتلaci انه ال id عباره عن primary key وزي  
ماانت شايف هوا عباره عن ارقام والأرقام دي بتمثل مرجع لل record اللي هيا فيه  
فماينفعش ابدا ان الرقم ده يتكرر باي شكل زي رقم البطاقه او رقم موبايلك كده  
ماينفعش يتكرر

لو بصيت في جدول ال department برضه هتلaci عمود ال id بيمثل primary key  
للجدول بحيث انك لما تيجي لجدول الموظفين وتقول انه محمد أبو هدهود  
بيشتغل في ال department رقم واحد لما تروح لجدول ال department وتدور  
عالرقم ده تبع اني قسم مايجيبلکش قسمين لا هوا قسم واحد

يعني من شروط ال primary key انه مايكونش متكرر ومايكونش ب null  
ومايتغيرش عشان هيكون مربوط علي اكتر من حاجه فلو غيرته يبقى انت كده بوظت  
الداتا بيز

بيقول انه ال primary key ممكن يكون اتنين مش واحد بس لا يفضل استخدامه  
فيه ناس بتيجي تصمم الداتا بيز وتعمل 4 اعمده مثلا ويحطوهم كلهم  
ممكن تعمله بس مش بيكون عملي وهيكون ابطأ

طيب بص في جدول ال department هل ينفع اخلي عمود ال name يشتغل  
primary key ؟

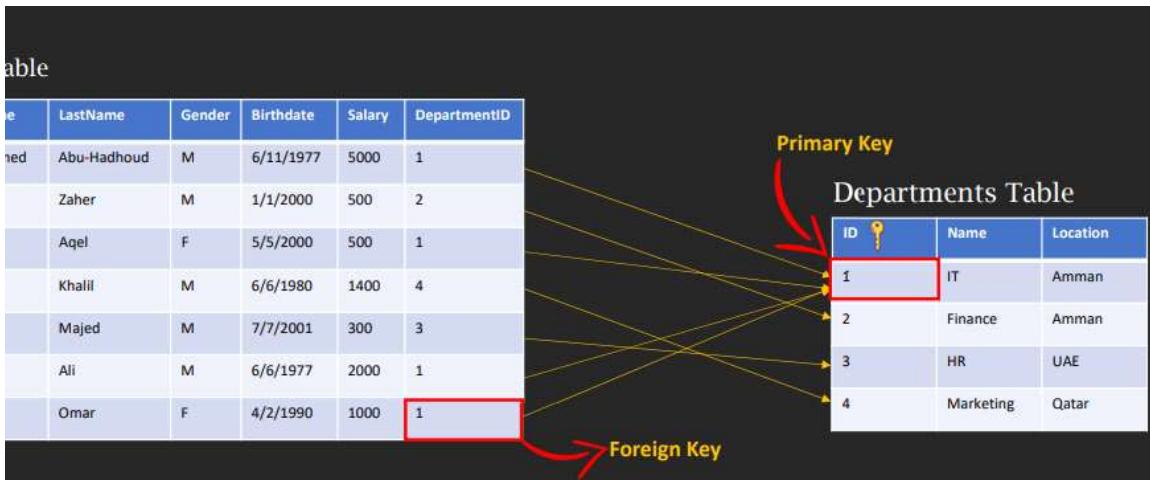
قالك اه ينفع لانه مش بيتكسر ومش بيساوي null لكن لايفضل لانه من النوع string  
لانه هيكون ابطأ في البحث وهياخد مساحات اكبر

ال primary key لما بيتنقل لجدول تاني عشان يدل عليه بيكون اسمه foreign key  
زي عمود ال department id في جدول ال employees كده

ال foreign key هو مؤشر علي primary key في جدول تاني وماينفعش تحط فيه  
رقم من دماغك عشان مايزعلکش لانه من نفسه بيروح يشوف الرقم ده موجود في  
الجدول الثاني ولا لا ودي اسمها referential integrity

ولو جيت تحذف record مربوط بجداول تانيه مش هيقبل منك ده  
وفيه اوبشن بيمثل مصيبه واسميه cascade delete وده لو جيت تحذف record  
من جدول معين بيحذفه وبيحذف أي record تاني في أي جدول تاني مرتبط بال

اللي انت عاوز تحذفه record



## Primary Key

A primary key is a column or set of columns in a relational database table that uniquely identifies each row or record in the table.

A unique identifier for each record and serves as a reference point for tables that have a relationship with the table in question.

Every table in a relational database must have a primary key, and it should be a non-null value that is unique and stable over time.

Primary Key should not be changed.

## Foreign Key

A foreign key, on the other hand, is a column or set of columns in a table that refers to the primary key of another table.

Establishes a relationship between two tables, allowing data to be linked between them.

A foreign key ensures that referential integrity is maintained by ensuring that any value in the foreign key column must exist in the primary key column of the related table.

## Primary Key

A primary key uniquely identifies a record in a table, while a foreign key establishes a relationship between two tables by referencing the primary key of another table.

### الواجب

A primary key is a column or set of columns in a relational database table that uniquely identifies each row or record in the table.

True

False

Primary Key is a unique identifier for each record and serves as a reference point for other tables that have a relationship with the table in question.

True

False

Each table in a relational database must have a primary key, and it should be a non-null value that is unique and stable over time.

True

False

Primary Key should not be changed.

True

False

A foreign key, on the other hand, is a column or set of columns in a table that refers to the primary key of another table.

True

False

Foreign Key establishes a relationship between two tables, allowing data to be shared and linked between them.

True

False

The foreign key ensures that referential integrity is maintained by ensuring that any value in the foreign key column must exist in the primary key column of the related table.

True

False

In summary, a primary key uniquely identifies a record in a table, while a foreign key establishes a relationship between two tables by referencing the primary key of another table.

True

False

Primary Key can be more than one filed.

True

False

It's better to use numbers as primary keys because they are fast in search.

True

False

Primary Key can be NULL.

True

False

Primary Key can be repeated.

True

False

Primary Key should be Unique, Not NULL , and should not be changed.

True
False

Value in the foreign key column must exist in the primary key column of the related table.

True
False

### **What is Redundancy? and why it's a problem?**

ال data redundancy معناها انه عندك تكرار في الداتا الديزائين السيئ للداتا بيز  
بيؤدي لحدوث تكرارات كتير في الداتا بيز وده ليه مميزاته وليه عيوبه  
بص للجدول ده اللي معمول في text file فيه تكرارات

## Employees File: Redundancy Problem

Name	Last Name	Gender	Birth Date	Salary	Dept ID	Dept Name	Dept Location
Immed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
Zaher		M	5/7/1990	3000	2	Finance	Amman
Aqel		F	5/5/2000	500	1	IT	Amman
Khalil		M	6/6/1980	1400	4	Marketing	Qatar
Majed		M	7/7/2001	300	3	HR	UAE
Ali		M	6/6/1977	2000	1	IT	Amman
Omar		F	4/2/1990	1000	1	IT	Amman

الكلام ده بيأخذ مساحه في الجهاز عالفاضي وده بيكون بدايه لمشاكل تانيه زي :-

- المساحه المهدره
- : inconsistency Data •
- : Corruption Data •
- Missing /incomplete data •
- Data integrity •
- Hard to maintain •

مثلا في الجدول اللي فوق انت حبيت تغير اسم ال information انت كده هتضطر انك تغيره في كل ال records ويه واحد ممكن تنسى تعديل فيه

فلما جت الداتا بيز فصلت بين الداتا فقالك خلي بيانات الموظفين لوحده وبيانات الأقسام لوحدها وبالتالي لو عايزة تعديل علي بيانات قسم معين هتعدله مره واحده وخلصنا وده شكل من اشكال ال normalization وهو حل لمشكلة ال redundancy

ال redundancy مش دايما تكون ضاره لأنها ساعات تكون مفيدة واسع في اني مثلا عاوز ازود عمود في جدول الأقسام يقولي عدد الموظفين في القسم ده دي هتكون redundancy ملهمه مشتقه من جدول الموظفين فالافضل هنا انك تستخدم ال redundancy

### Example: Redundancy Problem

stName	Gender	Birthdate	Salary	DeptID	DeptName	DeptLocation
Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
Zaher	M	5/7/1990	3000	2	Finance	Amman
Aqel	F	5/5/2000	500	1	IT	Amman
Khalil	M	6/6/1980	1400	4	Marketing	Qatar
Mahed	M	7/7/2001	300	3	HR	UAE
Omar	M	6/6/1977	2000	1	IT	Amman
Huda	F	4/2/1990	1000	1	IT	Amman

Duplicates  
Redundancy

- More Wasted Space
- Data Inconsistency
- Data Corruption
- Missing/Incomplete Data
- Data Integrity Problems
- Hard to maintain

### Normalization

#### Employees Table

ID	FirstName	LastName	Gender	Birthdate	Salary	DepartmentID
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1
2	Ali	Zaher	M	5/7/1990	3000	2
3	Lubna	Aqel	F	5/5/2000	500	1
4	Fadi	Khalil	M	6/6/1980	1400	4
5	Maha	Majed	M	7/7/2001	300	3
6	Omar	Ali	M	6/6/1977	2000	1
7	Huda	Omar	F	4/2/1990	1000	1

Primary Key

ID	Name	Location
1	IT	Amman
2	Finance	Amman
3	HR	UAE
4	Marketing	Qatar

This is call Normalization

Foreign Key

## Redundancy

Redundancy refers to the presence of duplicated data within the database. Redundancy can occur in different ways, such as storing the same information multiple times, or storing information that can be derived from other data in the database.

Redundancy can sometimes be useful, it can also cause problems. For example, redundant data takes up additional storage space and can make it difficult to maintain consistency within the database. If one copy of the data is updated, the other copies may become outdated, leading to inconsistencies and errors.

To handle redundancy in databases, normalization techniques can be used to organize the data in a way that minimizes duplication.

# alization

tion is the process of organizing data in a database in a way that redundancy and improves data integrity. There are several levels of alization, also known as normal forms, each with its own set of rules.

redundancy in databases, normalization techniques can be used to the data in a way that minimizes duplication.

lves breaking down the data into smaller, more atomic pieces and hem together through relationships, which can reduce the amount of data and make it easier to manage and update the database.

lly, enforcing data constraints, such as unique keys and foreign key hips, can help prevent redundant data from being inserted into the

alk more about normalization in the future not now.

## الواجب

Redundancy refers to the presence of duplicated data within the database.

True

False

Redundancy can occur in different ways, such as storing the same information multiple times, or storing information that can be derived from other data in the database.

True

False

Redundancy is always bad.

True

False

Redundancy can sometimes be useful.

True

False

Redundant data takes up additional storage space.

True

False

Redundancy can make it more difficult to maintain consistency within the database. If one copy of the data is updated, the other copies may become outdated, leading to inconsistencies and errors.

True

False

To avoid redundancy in databases, normalization techniques can be used to organize the data in a way that minimizes duplication.

True

False

Normalization is the process of organizing data in a database in a way that reduces redundancy.

True

False

Normalization improves data integrity.

True

False

Normalization involves breaking down the data into smaller, more atomic pieces and linking them together through relationships, which can reduce the amount of redundant data and make it easier to manage and update the database.

True

False

Enforcing data constraints, such as unique keys and foreign key relationships, can help prevent redundant data from being inserted into the database.

True

False

Constraints are Restrictions/Rules on Data.

True

False

### What is Data Integrity? and Why it's Important and Critical?

ال data integrity معناها تكامل البيانات وهذا بنقصد بيه انك لازم تتأكد ان كل البيانات اللي بتدخل الجدول عندك تكون مطبوعته وصحيحة لانه ساعات بيحصل أخطاء او مشاكل في الداتا بيز

بص كده الجدول ده

Key	FirstName	LastName	Gender	Birthdate	Salary	DepartmentID
	U@s#Z 9E	Abu-Hadhoud	M	6/11/1977	-400	1
	Ali	Zaher	M	5/7/1990	3000	2
	Lubna	1/1/2000	F	Hamdi	500	1
	Fadi	Khalil	M	6/6/1980	1400	4
	Maha	Majed	M	7/7/2001	300	3
	Omar	Ali	M	6/6/1977	2000	1
	Huda	Omar	F	4/2/1990	1000	5

**Primary Key**

Departments Table

ID	Name	Location
1	IT	Amman
2	Finance	Amman
3	HR	UAE
4	Marketing	Qatar

اول مشكلة عندك انه ال record الأول first name ودي احد مشاكل ال

migration وهيا ال data corruption وده بيحصل لما بتعمل لdata integrity للبيانات من سистем قديم لسيتم جديد يعني اثناء نقل البيانات من سистем للتاني بيحصل خطأ معين هو اللي بيحافظ على البيانات دي

تاني مشكله هيا ال salary في اول record جايبلوك قيمته بالسالب ممكن تكون عامل data validation في الكود بتاعك بس بيقولك ده مش كفايه وانك لازم تعمله تاني على مستوى البيانات ييز بانك تعمل constraint يعني قيد عال field ده انه مايقبلش غير ارقام موجبه فقط

بعض في ال last name في اول record رقم 3 هتلافقه كاتب تاريخ بدل مايكتب اسم الشخص دي برضه ممكن تكون ناتجه عن ال data migration او يكون حد مش واحد باله وكتب التاريخ بدل الاسم

بعض برضه عال field بتاريخ في نفس ال record حاطط الاسم مكان التاريخ ده برضه من مشاكل ال data integrity ان انا مش عامل ال data type صحيحة عامله date بدل مايكون string

بعض عال record الأخير هتلافقه كاتب في ال department id الرقم 5 وده مش

موجود في الجدول بتاع الأقسام وده بيكون بسبب ان انا مش عامل referential integrity بين الجدولين يعني مش معرف العلاقة بين الجدولين

عشان كده بيقولك انك تعمل داتا بيز صح يعتمد عليها ده اهم بكثير من انك تكتب query وده مش حته ادخال البيانات بس لا ده كمان بيشمل الحذف والتعديل والنقل واي حد بيتعامل مع الداتا لازم يحط الأنواع دي من ال data integrity اعتباره :-

• لازم تتأكد انه كل record ليه id من خلاله تقدر تستدعي أي شيء في ال record ده

• يعني الجداول مش مرتبطة ببعضها زي ماحصل وضفت الرقم 5 في عمود ال department id ومفيش في جدول ال department الرقم ده

• يعني ماتجيش تخلی ال salary بالسابق او التاريخ مثلًا مايكونش من ضمن فتره معينة

• انه الداتا تكون متوافقه مع شروط وقواعد الشركه زي مثلًا انه الحسابات البنكيه ماينفعش تقل او تزيد عن رقم معين او انه في مستشفى والمستشفى مانعه انه بيانات المرضى فيه يتم مشاركتها مع حد

## Integrity

grity refers to the accuracy, consistency, and reliability of data entire life cycle, from creation to deletion. In other words, it the assurance that data is complete, accurate, and trustworthy.

several factors that can impact data integrity, including hardware or software failure, security breaches, and data transfer

in data integrity, it is important to establish appropriate and procedures, and to implement appropriate technologies, such as backups, and access controls.

# Integrity

different types of data integrity that organizations need to consider:

**Entity integrity:** This ensures that each row or record in a table is unique and uniquely identified. This is typically achieved through the use of primary keys.

**Referential integrity:** This ensures that relationships between tables are maintained and that there are no orphaned records. This is typically achieved through the use of foreign keys.

**Business rules integrity:** This ensures that data is within acceptable ranges or values. For example, a date field should only contain valid dates, and a numeric field should only contain valid numbers.

# Integrity

**Business rules integrity:** This ensures that data meets business rules and requirements. For example, a bank might have rules around minimum and maximum account balances, or a hospital might have rules around patient data confidentiality.

# Integrity

Maintaining data integrity is critical for organizations that rely on accurate and trustworthy data to make informed decisions. Without data integrity, organizations risk making decisions based on incomplete, inaccurate, or unreliable data, which can lead to poor outcomes, financial loss, and damage to reputation.

To maintain data integrity we use **Constraints**, we will explain them in the next lesson ☺.

الواجب

Data integrity refers to the accuracy, consistency, and reliability of data over its entire life cycle, from creation to deletion.

True

False

Data Integrity refers to the assurance that data is complete, accurate, and trustworthy.

True

False

There are several factors that can impact data integrity, including human error, hardware or software failure, security breaches, and data transfer errors.

True

False

To maintain data integrity, it is important to establish appropriate policies and procedures, and to implement appropriate technologies, such as encryption, backups, and access controls.

True

False

There are different types of data integrity that organizations need to consider: Entity integrity, Referential integrity, Domain integrity, and Business integrity.

True

False

Referential integrity: This ensures that each row or record in a table is unique and can be uniquely identified. This is typically achieved through the use of primary keys.

True

False

Entity integrity: This ensures that each row or record in a table is unique and can be uniquely identified. This is typically achieved through the use of primary keys.

True

False

Referential integrity: This ensures that relationships between tables are maintained and that there are no orphaned records. This is typically achieved through the use of foreign keys.

True

False

Domain integrity is the same as Business Integrity.

True

False

Domain integrity: This ensures that data is within acceptable ranges or values. For example, a date field should only contain valid dates, and a numeric field should only contain valid numbers.



Business integrity: This ensures that data meets business rules and requirements. For example, a bank might have rules around minimum and maximum account balances, or a hospital might have rules around patient data confidentiality.



### What is Constraint? and Why it's Important?

ال constraints هيا شروط وقواعد بتطبّقها على الداتا عشان تضمن او تحقق ال data integrity

ودي تقدر تعمل على مستوى الاعمده او الجدول كله ودي بعض انواع ال :- constraints

• key Primary : - وده بيضمّنك انه كله صف في الجدول ليه رقم خاص بي  
و ماينفعش يتغيّر زي ماتكلّمنا قبل كده

• key Foreign : - وده بيضمّنك انه كل صف في الجدول بتاعك ليه علاقه  
بصف تاني في جدول تاني وان ما فيه حد بيحط رقم مش موجود في الجدول  
التاني

- constraints Unique :- وده انك بتعرف عمود معين انه الداتا اللي فيه لا تقبل التكرار زي انك مثلا بتدخل بيانات الشيك رقم الشيك ماينفعش يتغير او بيانات عملاء ليهم رقم قومي ماينفعش يتكرر العمود ده لا هوا primary key ولا حتى foreign key بس هوا بنفسه الداتا اللي فيه ماينفعش تتكرر زي ال serial number بتاع الموبايل
- null Not :- انك بتيجي تعرف عمود معين انه لازم يكون فيه داتا ماينفعش يكون ب null
- constraint Check :- انك تحط مثلا انه ماينفعش ال age يكون اقل من 18 سنة

وفيه معلومه هنا بيقولها لك انه الفرق بين ال primary key وا unique constraint انه ال primary key بقبل ال null unique constraint

## Constraints

In the context of databases, constraints are rules or conditions that are applied to the data to ensure its integrity and consistency. Constraints can be applied to individual columns or to entire tables, and they are used to enforce various rules and restrictions on the data.

By using constraints, you can help ensure that your data is accurate, consistent, and easy to manage.

# raints

ome common types of constraints used in databases:

**Key Constraint:** This constraint ensures that a column or a set of columns uniquely identifies each row in a table. This constraint helps to maintain data integrity and ensure that there are no duplicate rows in the table.

**Key Constraint:** This constraint establishes a relationship between two tables based on a key field. The foreign key constraint ensures that data in one table matches data in another table, and it helps to maintain referential integrity in the database.

**constraint:** This constraint ensures that the data in a column or set of columns is unique across all rows in the table. This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table.

# raints

ome common types of constraints used in databases:

**l Constraint:** This constraint ensures that a column or set of columns cannot contain null (empty) values. This constraint helps to ensure that data is complete and accurate, and it can help prevent errors in calculations.

**onstraint:** This constraint ensures that the data in a column or set of columns meets a specified condition. This constraint helps to enforce data integrity and prevent invalid data from being inserted into the table.

# view Question?

the difference between Primary Key Constraint and Unique Constraint?

Primary Key is Unique but it does not allow NULL while Unique allows NULL.

الواجب

In the context of databases, constraints are rules or conditions that are applied to the data to ensure its integrity and consistency. Constraints can be applied to individual columns or to entire tables, and they are used to enforce various rules and restrictions on the data.

True
False

By using constraints, you can help ensure that your data is accurate, consistent, and easy to manage.

True
False

The constraint that ensures that a column or a set of columns uniquely identifies each row in a table. This constraint helps to enforce data integrity and ensure that there are no duplicate rows in the table is?

Foreign Key Constraint

Unique Constraint

Primary Key Constraint

Check Constraint

Not Null Constraint

Which constraint ensures that the data in a column or set of columns meets a specified condition. This constraint helps to enforce data integrity and prevent invalid data from being inserted into the table?

Not Null Constraint

Primary Key Constraint

Check Constraint

Foreign Key Constraint

Unique Constraint

Which constraint establishes a relationship between two tables based on a key field. The foreign key constraint ensures that data in one table matches data in another table, and it helps to maintain referential integrity in the database?

Unique Constraint

Foreign Key Constraint

Primary Key Constraint

Not Null Constraint

Check Constraint

Which constraint ensures that a column or set of columns cannot contain null (empty) values. This constraint helps to ensure that the data is complete and accurate, and it can help prevent errors in queries and calculations?

Foreign Key Constraint

Primary Key Constraint

Unique Constraint

Not Null Constraint

Check Constraint

Which constraint ensures that the data in a column or set of columns is unique across all rows in the table. and allows Null as well, This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table?

Check Constraint

Unique Constraint

Primary Key Constraint

Foreign Key Constraint

Not Null Constraint

Which constraint(s) ensures that the data in a column or set of columns is unique across all rows in the table. This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table.

Primary Key Constraint

Foreign Key Constraint

Not Null Constraint

Unique Constraint

Check Constraint

Which of the following is TRUE?

Primary Key is Unique and allows NULL while Unique Does not allow NULL.

Primary Key is Unique Constraint but does not allow NULL while Unique Constraint allows NULL.

Both Primary Key Constraint and Unique Constraint prevent duplicates.

To achieve Entity integrity we use:

Foreign Key Constraints

Unique Constraints

Primary Key Constraints

Check Constraints

Not Null Constraints

To achieve Referential integrity we use:

---

Foreign Key Constraints

---

Unique Constraints

---

Primary Key Constraints

---

Check Constraints

---

Not Null Constraints

---

To achieve Domain integrity we use:

---

Foreign Key Constraint

---

Primary Key Constraint

---

Unique Constraint

---

Not Null Constraints

---

Check Constraints

---

## What is SQL?

بعض كده عالجدول ده وتخيل اننا كنا بنحفظ الداتا دي في ملف text

Name	LastName	Gender	Birthdate	Salary	DeptID	DeptName	DeptLocation
Jammed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
	Zaher	M	5/7/1990	3000	2	Finance	Amman
ia	Aqel	F	5/5/2000	500	1	IT	Amman
	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
a	Majed	M	7/7/2001	300	3	HR	UAE
ir	Ali	M	6/6/1977	2000	1	IT	Amman
a	Omar	F	4/2/1990	1000	1	IT	Amman

لو جيت قولتلك عايزين بيانات الموظفين الاناث هتعمل ايه ؟

كنا هنحمل الداتا اللي في الملف ده كلها وبعدين نلف عليهم كهم عشان نعملهم separation لانه كل record هو عباره عن string وبعدين نخزنهم في شكل معين او كلاس وبعدين نلف عليهم كلام ونشوف لو ال gender كان female بناخدها نعرضها

فعشان بس تعمل فلتر للداتا اخذت وقت وقت كبير واستهلكت موارد كتير

طيب لو عايز اعمل نفس الحوار ده في ال sql كل اللي هعمله اني استخدم حاجه اسمها query sql ودي بتكون بسيطه وسريعه

فكل اللي هنعمله انا نكتب select statement زي كده

```
Select * from Employees
Where Gender = 'F'
```

اللغه اللي كتبنا بيها هيا عباره عن query language وهيأ لغه قريبه من لغة الانسان واللغه دي هيأ جزء من حاجه اسمها sql وده اختصار ل structured query language وياما بننطقها q | s see qual

ال sql بنسخدمها عشان نتواصل مع الداتا بيز عشان نوصل للداتا او نعمل عليها

## عمليات معينة

وهنا بيقولك انه ال sql هيا standard يعني اللي هتتعلمها هنا هيمنفع تطبقه علي كل أنواع الداتا بيز اللي موجوده سواء sql server او oracle او حتى Microsoft access

هيا لغه مبنيه على أوامر وتقدر بيها تعمل ايه حاجه عالداتا بيز من اضافه وحذف وتعديل

وبعد كده بيديك امثله للاوامر دي  
وبعدين بيقولك ان ال sql جواها 5 لغات :-

- (DDL)language Data definition : - ودي من خلالها بتتم عمليات خاصه بالداتا بيز او الجداول نفسها التعريف بالجداول الجديدة والحذف والاستبدال والاوامر اللي فيها هيا (create-Drop – Alter - truncate)

- (DML)language Data manipulation : - ودي من خلالها بتتم عمليات التعامل مع الداتا او البيانات نفسها او ال زي (update – delete – insert)

- (DCL)Language DATA Control : - زي حوار الصلاحيات بتاعت المستخدمين والاوامر اللي فيها ( grant – revoke )

- (TCL Transaction Control)language : - ودي هنجيلها بعدين والاوامر اللي فيها (commit – rollback – save point)

- (DQL)Data Query language : - ودي اللي هيا الامر select ومن خلاله

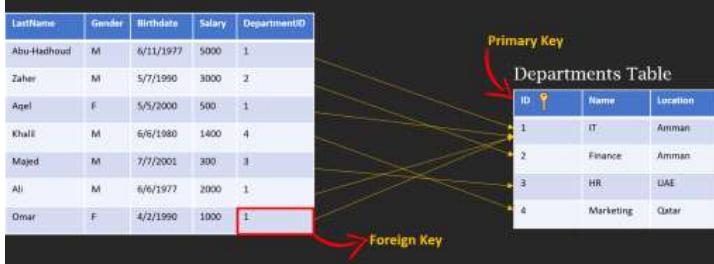
## بتقدير تستخرج تقارير

ile:

LastName	Gender	Birthdate	Salary	DeptID	DeptName	DeptLocation
Abu-Hadoud	M	6/11/1977	5000	1	IT	Amman
Zaher	M	5/7/1990	3000	2	Finance	Amman
Aqel	F	5/5/2000	500	1	IT	Amman
Khalil	M	6/6/1980	1400	4	Marketing	Qatar
Majed	M	7/7/2001	300	3	HR	UAE
Ali	M	6/6/1977	2000	1	IT	Amman
Omar	F	4/2/1990	1000	1	IT	Amman

To Get  
All Female  
Employees

- You have to write code and loops.
- Slow process
- Slow performance



- You Use SQL Query 😊
- Simple and fast.

Select \* from Employees  
Where Gender = 'F'

## is SQL

ds for Structured Query Language  
ed as “S-Q-L” or sometimes as “See-Quel”.  
sed to communicate with a database.  
you access and manipulate databases  
management systems that use SQL are: Oracle, Sybase, Microsoft SQL  
Access, Ingres, etc.

# What Can You Do With SQL?

- can execute queries against a database
- can retrieve data from a database
- can insert records in a database
- can update records in a database
- can delete records from a database
- can create new databases
- can create new tables in a database
- can create stored procedures in a database
- can create views in a database
- can set permissions on tables, procedures, and views

## Types Of SQL Statements:

```
FROM Employees WHERE Salary < 1000;

FirstName , LastName FROM Employees WHERE Salary < 1000 and Gender='M';

FROM Employees WHERE Salary between 500 and 1000;

SELECT(*) from Employees;

SUM(Salary) from Employees;

AVG(Salary) from Employees;

DELETE Employees where ID=10;

UPDATE Employees set FirstName = 'Amjad' where ID = 10;
```

# Examples:

- CREATE DATABASE MyDatabase;
- DROP DATABASE MyDatabase;
- CREATE TABLE Employees (  
    ID int,  
    FirstName varchar(255),  
    LastName varchar(255),  
    Address varchar(255),  
    City varchar(255)  
);

## Types of SQL Statements:

Definition Language (DDL)  
Manipulation Language (DML)  
Control Language (DCL)  
Transaction Control Language (TCL)  
Query Language (DQL)



الواجب

SQL stands for Structured Query Language

True

False

SQL is Pronounced as “S-Q-L” or sometimes as “See-Quel”.

True

False

SQL is used to communicate with a database.

True

False

SQL lets you access and manipulate databases.

True

False

Database management systems that use SQL are: Oracle, Sybase,  
Microsoft SQL Server, Access, Ingres, etc.

True

False

SQL can execute queries against a database.

True

False

SQL can retrieve data from a database.

True

False

SQL can insert and update records in a database.

True

False

**SQL can delete records from a database.**

True

False

**SQL can create new databases, Tables, Views ..etc in the database**

True

False

**SQL can set permissions on tables, procedures, and views.**

True

False

## What are 5 Types of SQL Statements?

Data Definition Language (DDL)

Data Manipulation Language (DML)

Data Control Language (DCL)

Transaction Control Language (TCL)

Data Query Language (DQL)

Data Handling Language (DHL)

### DBMs vs RDBMS Summary

## is Database?

is an organized collection of data so that it can be easily managed. To manage these databases, Database Management Systems (DBMS) are used.

BMS:

, there are two common types of databases:

ventional (DBMS): File System, XML..etc.

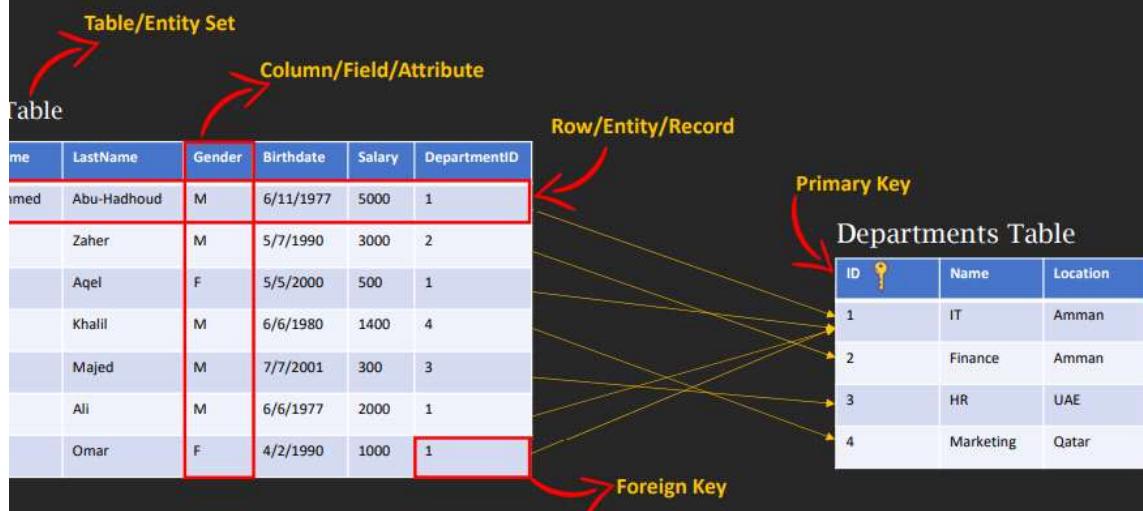
al (RDBMS): enhanced version of DBMS but with relations, examples Oracle, MySQL ..etc.

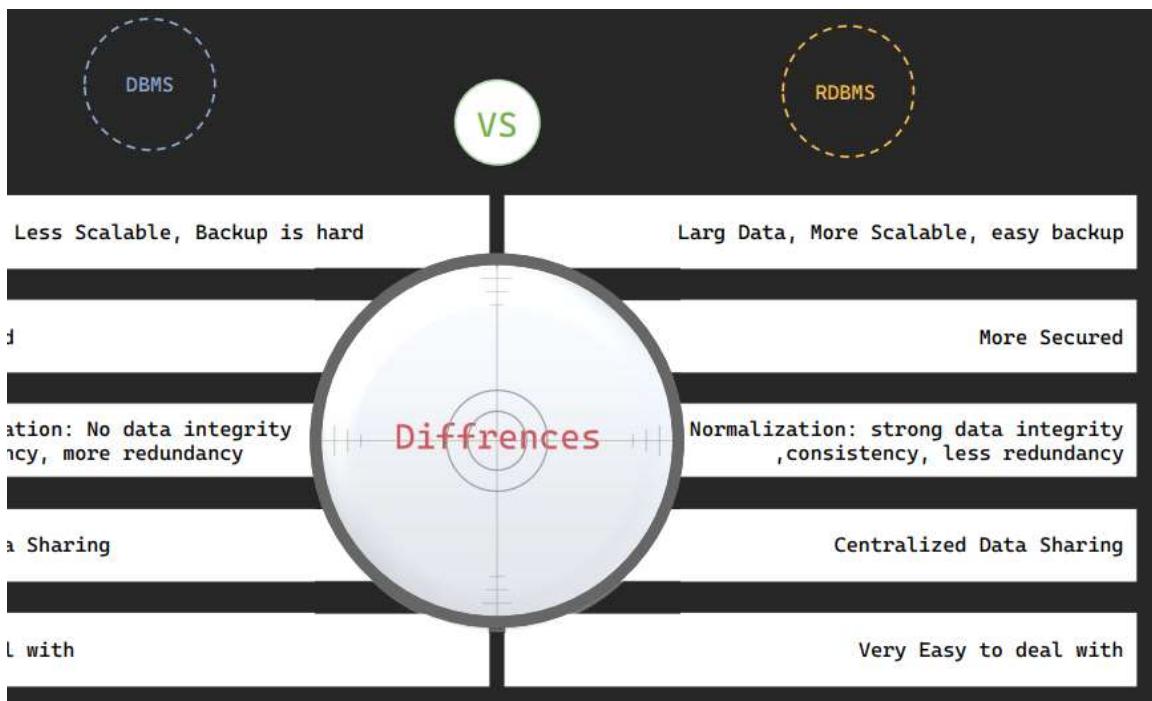
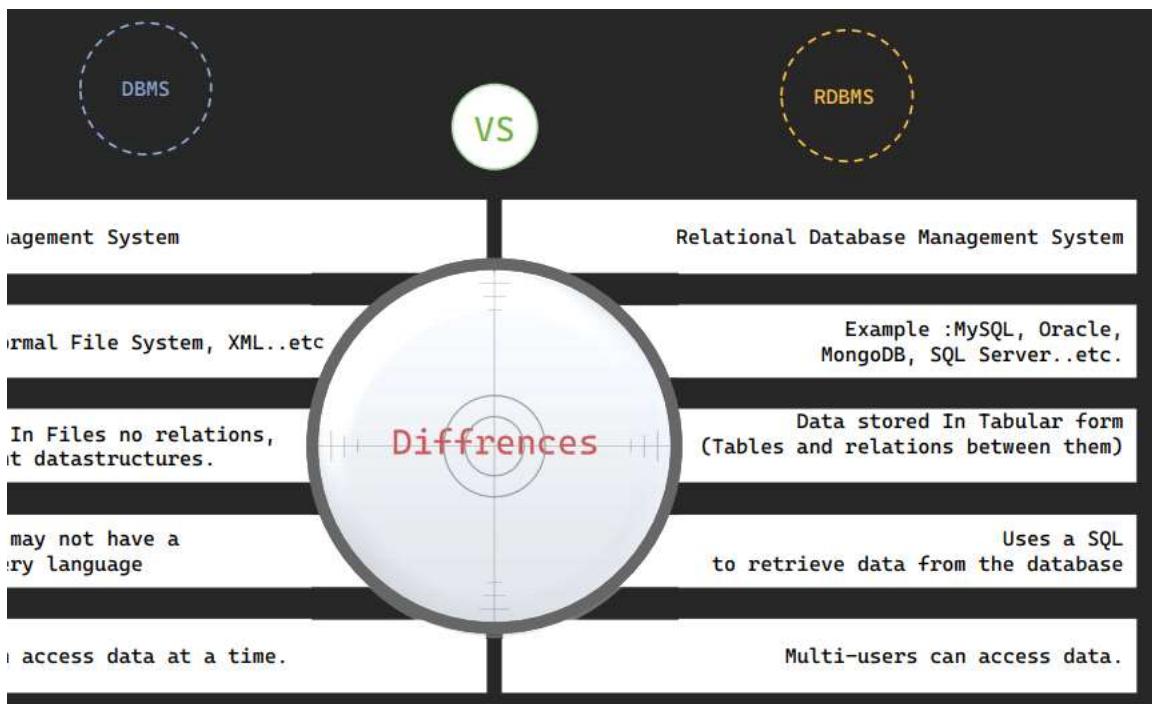
ile

FirstName	LastName	Gender	Birthdate	Salary	DeptID	DeptName	DeptLocation
Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
Ali	Zaher	M	5/7/1990	3000	2	Finance	Amman
Lubna	Aqel	F	5/5/2000	500	1	IT	Amman
Fadi	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
Maha	Majed	M	7/7/2001	300	3	HR	UAE
Omar	Ali	M	6/6/1977	2000	1	IT	Amman
Huda	Omar	F	4/2/1990	1000	1	IT	Amman

## DBMS .

stored in an organized fashion in tables containing rows and columns along with relations between





## Database Design: Conceptual Design

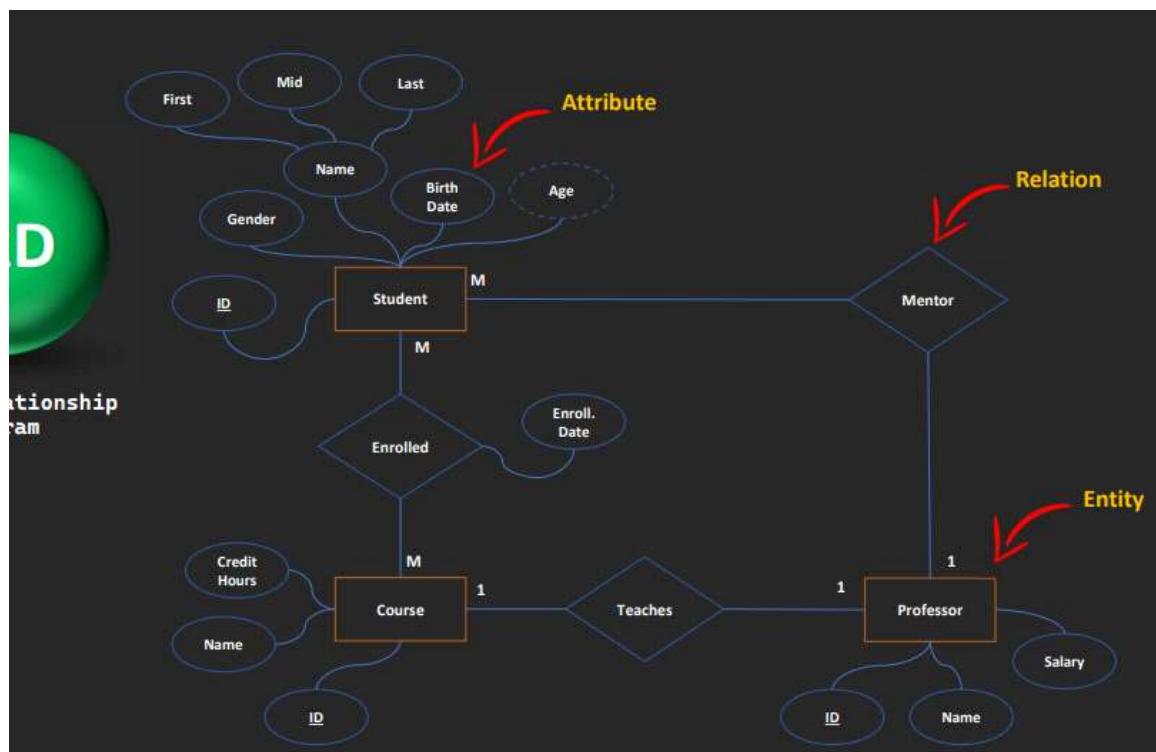
### What is ERD? and Why?

هنا بيقولك انه الديزاین بتاع الداتا بيز اهم حاجة لانك لو عامله غلط او مش زي

مالعميل عايزه وبدأت تبني عليه البرنامج بتاعك هتكون اهدرت وقت وتكليف وجهد  
عالفاضي

فعشان تعمل ديزاين مبدائي او conceptual design بتسخدم حاجه اسمها  
ER Model وهم اختصار entity relationship diagram وبيكون زي مخططات المبني

وده شكله



من خلال النظر لـ diagram ده بتعرف انه هيكون عندك 3 جداول في الداتا بيز هما professor و course و student وال العلاقات بين الجداول وكل جدول فيه انهي داتا

فانت لازم تحلل متطلبات النظام كلها قبل ما تبدأ في البرنامج بتاعك وبتقعد تعدل عليه لحد ما توصل للديزاين الصحيح

## **is ERD?**

Entity Relationship Diagram (ER Diagram) pictorially explains relationship between entities to be stored in a database.

mentally, the ER Diagram is a structural design of the database.

as a framework created with specialized symbols for the purpose of defining the relationship between the database entities.

Diagram is created based on three principal components: entities, attributes, and relationships.

## **is an ER Model?**

Entity-Relationship Model represents the structure of the database with the help of a diagram.

Designing is a systematic process to design a database as it requires you to analyze all data requirements before creating your database.

## Use ER Diagrams in DBMS?

ram helps you conceptualize the database and lets you know fields need to be embedded for a particular entity.

ram gives a better understanding of the information to be in a database.

ces complexity and allows database designers to build es quickly.

s to describe elements using Entity-Relationship models.

ws users to get a preview of the logical structure of the e.

## Conclusion

ram in RDBMS is widely used to describe the conceptual of databases.

es both users and database developers to preview the re of the database before implementing the database.

### الواجب

An Entity Relationship Diagram (ER Diagram) pictorially explains the relationship between entities to be stored in a database.

True

False

ER Diagram is a physical design of the database.

True

False

ER Diagram is a structural design of the database.

True

False

ER Diagram is a conceptual design of the database.

True

False

ER diagram is created based on three principal components: entities, attributes, and relationships.

True

False

An Entity-Relationship Model represents the structure of the database with the help of a diagram.

True

False

ER Modelling is a systematic process to design a database as it would require you to analyze all data requirements before implementing your database.

True

False

The cost of updating ERD is much cheaper than the cost of updating Database after you build it.

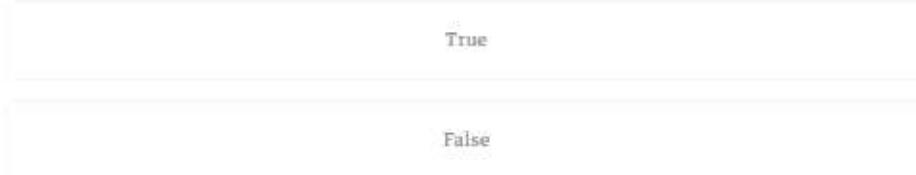
True

False

ER Diagram helps you conceptualize the database and lets you know which fields need to be embedded for a particular entity.



ER Diagram gives a better understanding of the information to be stored in a database.



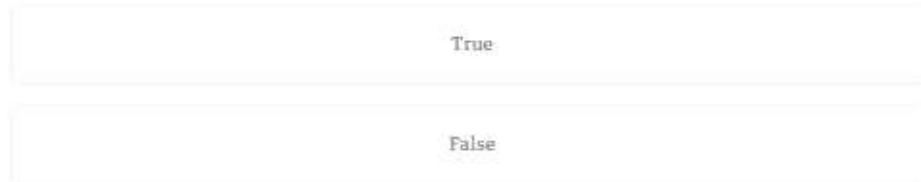
ERD reduces complexity and allows database designers to build databases quickly.



ER Diagram in RDBMS is widely used to describe the conceptual design of databases.

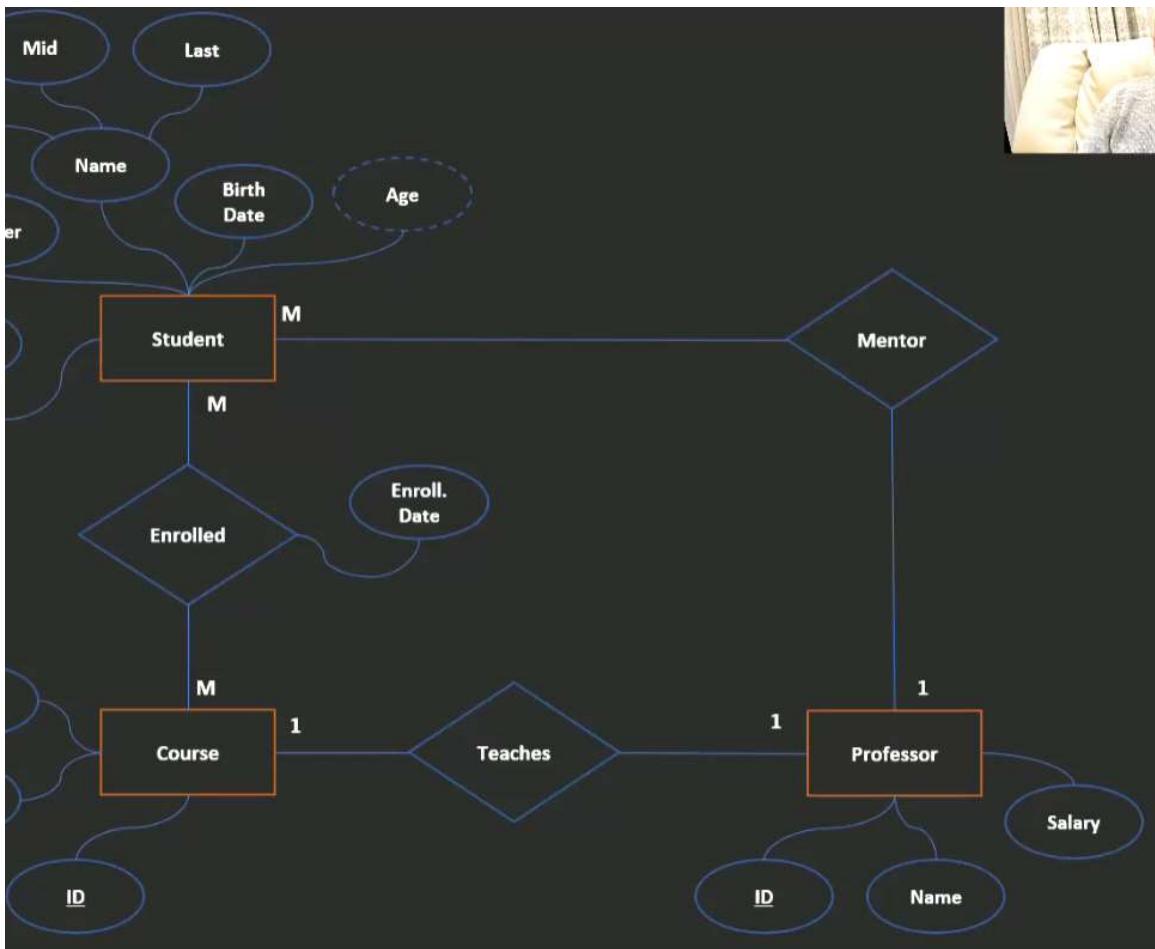


ERD helps both users and database developers to preview the structure of the database before implementing the database.

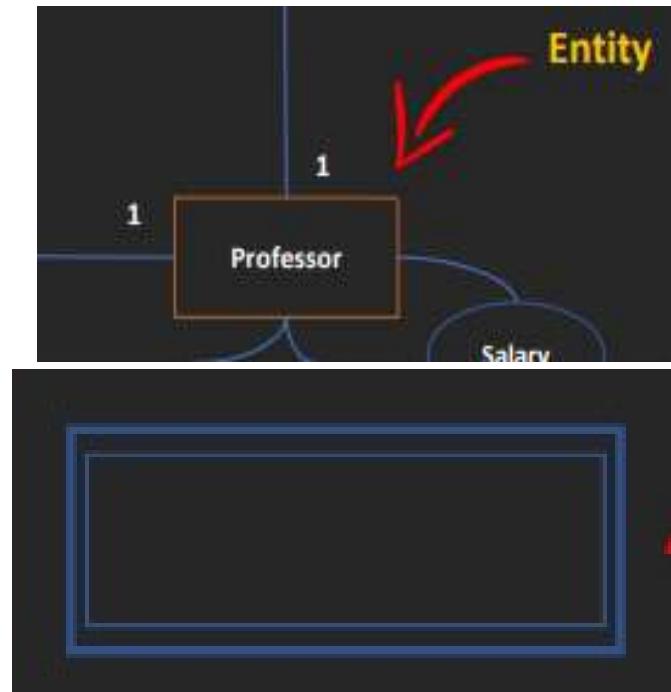


## ERD Symbols

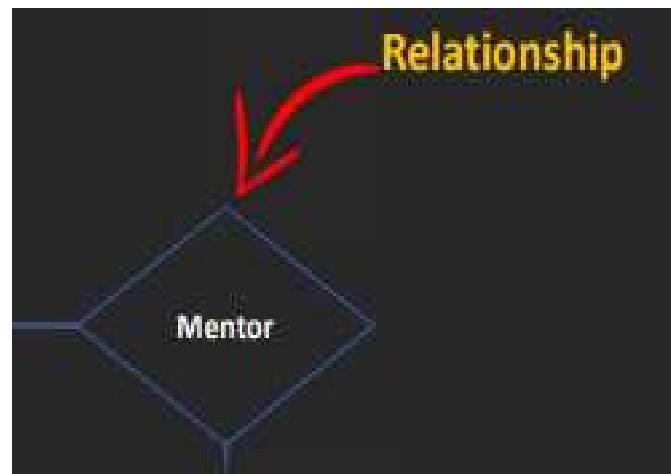
زي ال flow chart كان بيكون من رموز كانت ليها معاني بتساعدك في حل ال erd هنا ال ليها برضه رموز او symbols عشان تعرف تبنيها  
بص كده عال diagram ده



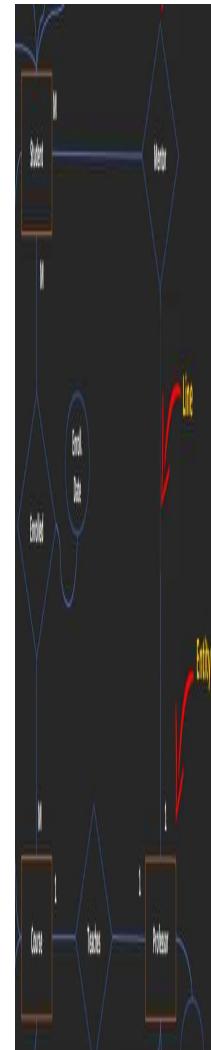
هتلaci أي entity بتمثela في مستطيل وبتعرف ان كنت محتاج تخزن الداتا دي على الجهاز لو محتاج يبقى الداتا entity وبيكون اسمها strong entity لانه ليه primary key ولو ملوش primary key وبيكون اسمها weak entity مستطيلين جوا بعض وده لايصح باستخدامه



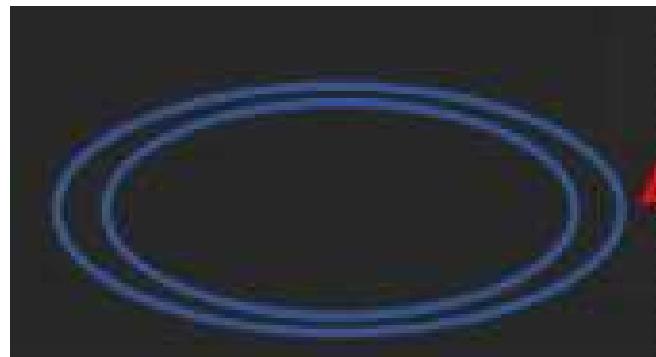
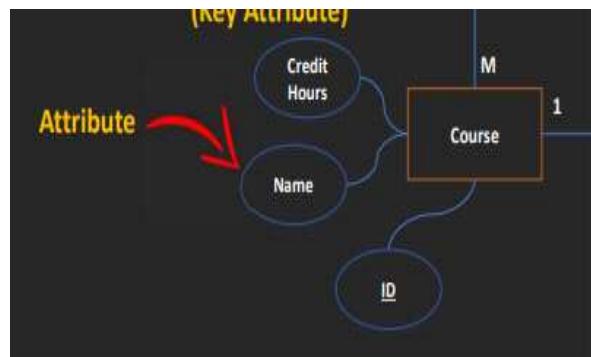
وبعدين بتسأل هل فيه علاقه بين entity و entity تانيه لو كانت الاجابه اه يبقى بتعمل relationship بينهم ودي بتديها الشكل ده اللي بيمثل العلاقة نفسها



وعشن تربط entity بال الثانيه بتوصليهم بخط



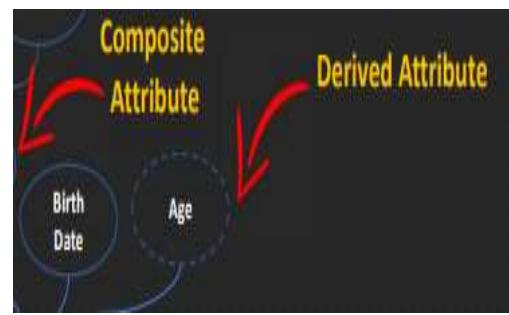
والعلاقات دي فيه منها أنواع هننشرحها بعدين  
 قالك بعد كده انه كل entity او صف بيكون فيه عدة اعمده او attributes ودي  
 بتمثلها في شكل بيضاوي  
 فيه حاجه اسمه bad practice وده multi valued attribute وهو عباره عن  
 انك تستخدم field عشان يمثل اكتر من قيمه وبيكون شكلين بيضاوين جوه بعض



ولو فيه attribute بينقسم لعدة انواع بيسمى composite attribute زي كده

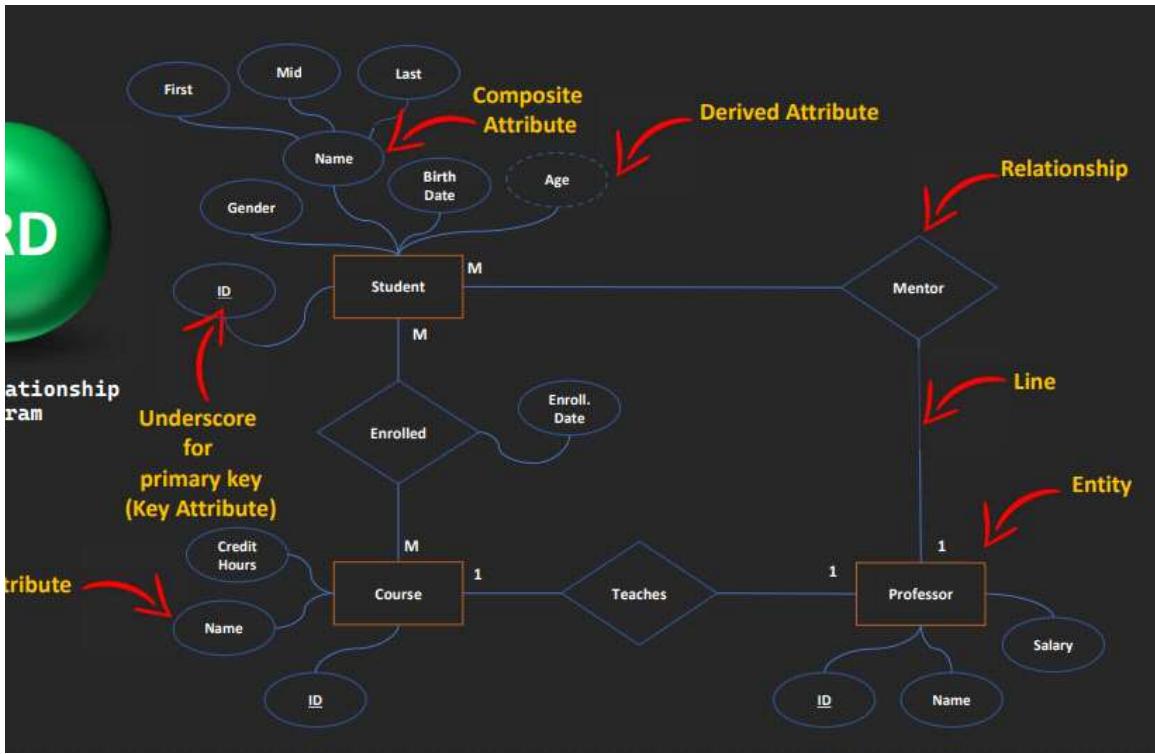


وفيه حاجه اسمها derived attribute او composite attribute وده عباره عن  
attribute عادي بس مش بتخزن فيه داتا بس بيتم اشتقاقه من اعمده تاني زي  
مايكون عندك تاريخ الميلاد وعايز تعمل عمود للعمر انت هنا مش محتاج تخزن  
العمر لانك مجرد ماتطرح تاريخ اليوم من تاريخ الميلاد هيكون عندك العمر وده  
بيتمثل في شكل بيضاوي بس بخط متقطع

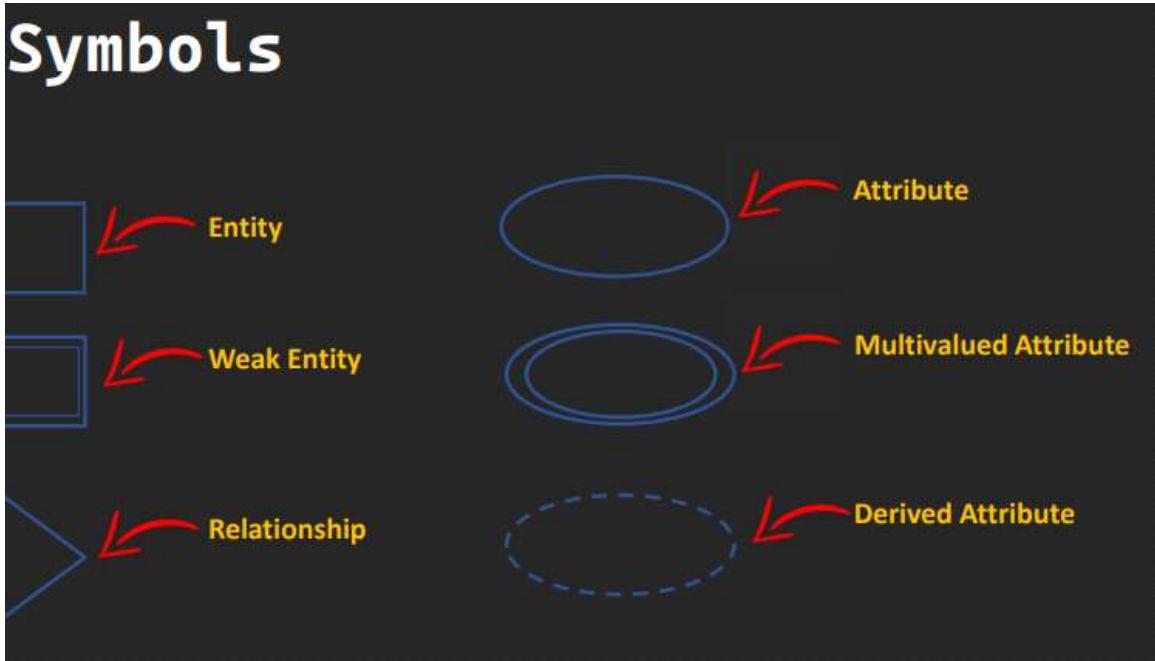


ای primary key موجود تحتیه خط ده بیکون





## Symbols



# Symbols Used in ER Diagrams:

Rectangles: This Entity Relationship Diagram symbol represents entity types

Diamonds: This symbol represents attributes

Ellipses: This symbol represents relationship types

Underline: It links attributes to entity types and entity types with other relationship types

Key: Here, it underlines the attributes

Double Ellipses: Represents multi-valued attributes



Which ERD symbol represents entity types?

---

Ellipses

---

Rectangles

---

Diamonds

---

Double Ellipses

---

Which ERD symbol represents attributes?

Rectangles

Diamonds

Ellipses

Lines

Which ERD symbol represents relationship types?

Diamonds

Ellipses

Rectangles

Double Ellipses

Which ERD symbol links attributes to entity types and entity types with other relationship types?

Rectangles

Ellipses

Diamonds

Lines

How to represent primary key in ERD?

Underline the attributes name

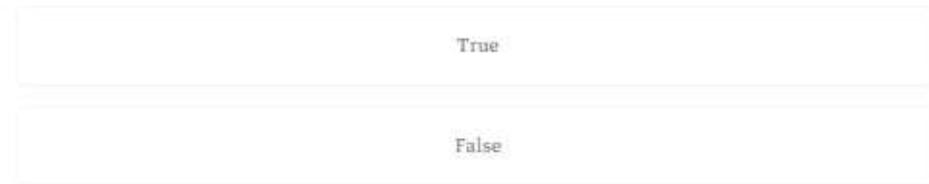
Double Ellipses

Strong Entity is the entity that has primary key(s).

True

False

Weak Entity is represented by double rectangles because it has no primary key for it and it depends on other entities to be identified.



### **Components of ERD**

هنا بيلملك الموضوع عشان يكون منظم اكتر وبيقولك انه بينقسم ل 3 أجزاء رئيسية  
ودي التقسيمه بتاعتهم

## **Components of ER Diagram:**

an ER Diagram on three basic concepts:

**Entities**  
Strong Entity  
Weak Entity

**Attributes**  
Simple Attribute  
Composite Attribute  
Multivalued Attribute  
Derived Attribute

**Relationships**  
One-to-One Relationships  
One-to-Many Relationships  
Many-to-One Relationships  
Many-to-Many Relationships

### **Entity (Strong) and Weak Entity**

قولنا قبل كده انه ال entity strong يااما weak نوعين يااما والفرق بينهم بيكون

انه ال primary key فيها strong entity انم ال weak entity مافيها primary key  
وقولنا انه ال entity بيمثل صف في الجدول بيعبّر عن بيانات عن أي حاجه  
ويتم تمثيله في شكل مستطيل

ال weak entity بيتتمتعريفها او استدعاءها عن طريق ال foreign key وانه يكون  
مربوط ب primary key ليها تانيه

## Entities

- Entity (Strong Entity).
- Weak Entity.

### ties(Strong Entities)

ty can be either a living or non-living component.

cases an entity as a rectangle in an ER diagram.

tity has a primary key

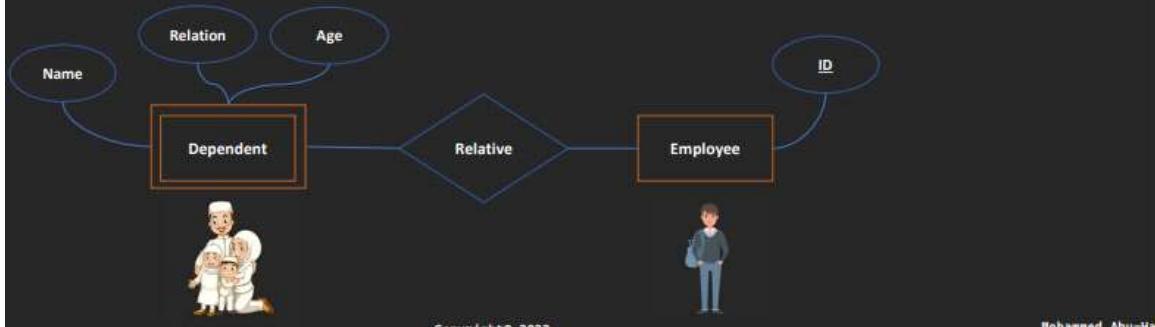
ample, in a student study course, both the student and the  
are entities.



# Entities

A entity that makes reliance over another entity is called a weak entity. We can represent the weak entity as a double rectangle in ER Diagram. A weak entity has no primary key.

In the example below, Employee is a strong entity because it has a primary attribute - ID. Unlike Dependent, the dependent is a weak entity because it does not have any primary key and the name here acts only as a descriptor.



الواجب

There are two types of entities, strong entities and weak entities

True

False

Strong Entity does not have primary key.

True

False

Strong Entity has to have primary key.

True

False

Weak Entity has no primary key(s)

True

False

An entity can be either a living or non-living component.

True

False

Which of the following assures the Entity Integrity.

Strong Entities

Weak Entities

Strong Entity is represented by double rectangles in ERD.

True

False

Strong Entity is represented by single rectangle in ERD.

True

False

Weak Entity is represented by double rectangles in ERD.

True

False

## Attributes

زي ما احنا عارفين انه كل جدول مكون من صفات واعمده والصفات بنسماها entities وبتعبر عن object او شيء معين

الاعمده بقى بيكون اسمها attributes او columns او fields وكل عمود بيمثل خاصية او صفة للشيء الممثل في شكل صف او entity وال attributes بتمثل في شكل بيضاوي في ال erd

ال primary key هو عمود بيكون key attribute ويتم تمثيله في ال erd في شكل بيضاوي وبتحط خط تحت اسمه

ال composite attribute هو عباره عن attribute عادي بس مركب بيكون من

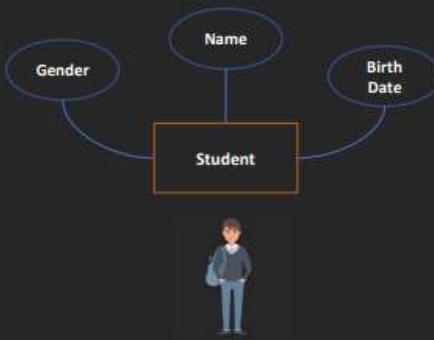
عدة attributes تانية وبيتم تمثيله بشكل بيضاوي عادي وطالع منه  
تانية

ال multivalued attribute بيمثل بشكلين بيضاوين جوا بعض وده بيعبّر عن عدة  
قيم في وقت واحد زي مثلا انه يتم تخزين عدة ارقام تيليفونات لنفس الشخص في  
عمود واحد وده شيء مش كويـس

ال derived attribute وده بيمثل بشكل بيضاوي متقطع وده مش داتا بتتخزن لا  
ده بيعتمد على عمود تاني وبيجيبلك منه معلومات تانية

## tribute

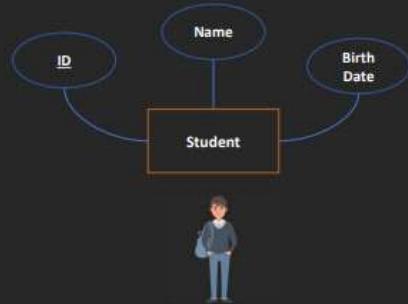
attribute exhibits the properties of an entity.  
You can illustrate an attribute with an oval shape in an ER diagram.



## Key Attribute

attribute uniquely identifies an entity from an entity set.  
underlines the text of a key attribute.

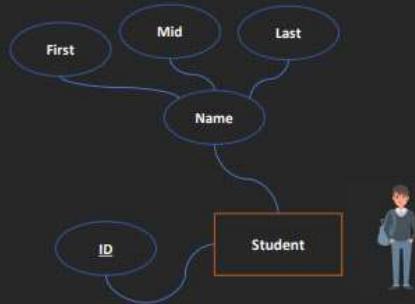
example: For a student entity, the ID can uniquely identify a student from a set of students.



## Composite Attribute

attribute that is composed of several other attributes is known as a composite attribute.

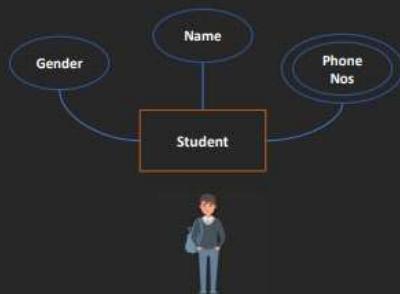
oval showcases the composite attribute, and the composite attribute oval is further connected with other ovals.



## Multivalued Attribute

If attributes can possess over one value, those attributes are called multivalued attributes.

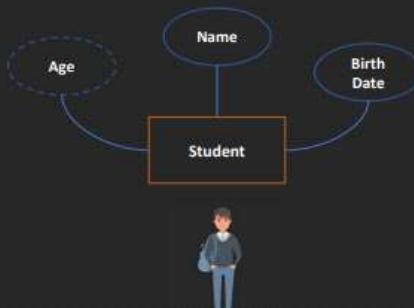
A double oval shape is used to represent a multivalued attribute.



## Derived Attribute

An attribute that can be derived from other attributes of the entity is known as a derived attribute.

In the ER diagram, the dashed oval represents the derived attribute.



الواجب

An attribute exhibits the properties of an entity.

True

False

You can illustrate an attribute with an oval shape in an ER diagram.

True

False

Key attribute uniquely identifies an entity from an entity set.

True

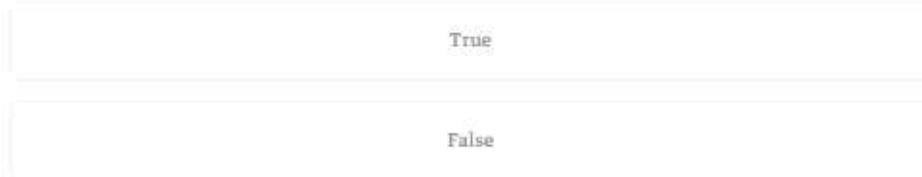
False

Key Attribute is represented by underlining the text of a key attribute.

True

False

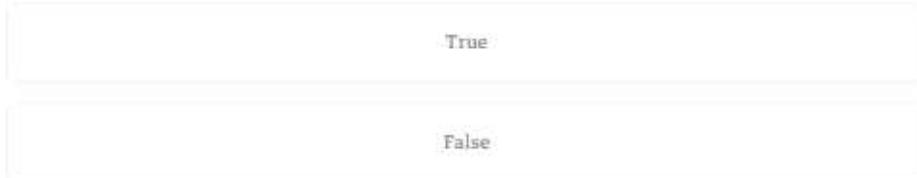
An attribute that is composed of several other attributes is known as a composite attribute.



An oval showcases the composite attribute, and the composite attribute oval is further connected with other ovals.



Some attributes can possess over one value, those attributes are called multivalued attributes.



The double oval shape is used to represent a multivalued attribute.



It's not recommended to have multivalued attributes.

True

False

An attribute that can be derived from other attributes of the entity is known as a derived attribute.

True

False

In the ER diagram, the dashed oval represents the derived attribute.

True

False

At the end the attribute is a field or column in the database.

True

False

## Relationships

ال relationships او العلاقات من اهم الحاجات اللي لازم تحددها صح في الداتا بيز بعد ما بتحدد ال entities بتيجي خطوة تحديد ال relationships

باختصار انت بتسال هل فيه علاقه بين الجدولين ولا لا لو فيه بترسم معين وبحط فيه ايه هيا العلاقة وبعد كده بترسم خط بيمشي من اول جدول بيروح للعلاقه ويبينتهي عند الجدول الثاني

العلاقات فيه منها انوع ومنها العلاقة بين الشئ ونفسه او علاقتين بين شئ وشئ اخر

## Relationship

Diamond shape showcases a relationship in the ER diagram.

Represents the relationship between two entities.

In the example below, both the student and the course are entities, and "Enrolled" is the relationship between them.



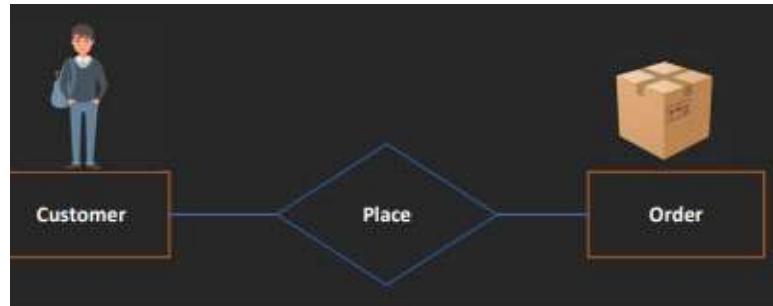
## Relationship

In the example below, both the student and the Identification-Card are entities, and "has" is the relationship between them.



# Relationship

example below, both the customer and the order are entities, “Place” is the relationship between them.



# Relationship

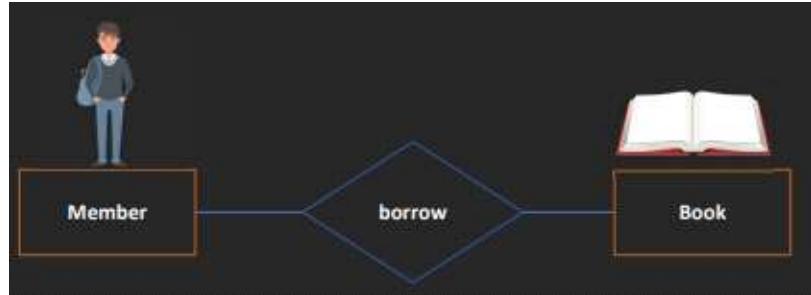
example below, both the customer and the order are entities, “Place” is the relationship between them.

A customer can have more than one product.



# Relationship

example below, both the Member and the Book are entities, “Arrow” is the relationship between them.



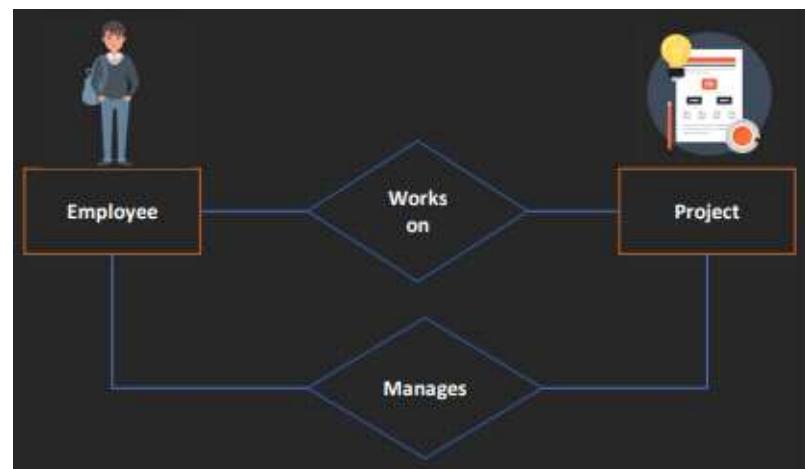
## Relationship

In the example below, both the Customer and the Car are entities, and “Rent” is the relationship between them.



## Relationship

In the example below, both the Employee and the Project are entities, and “Works on” is the relationship between them. An Employee manages projects.

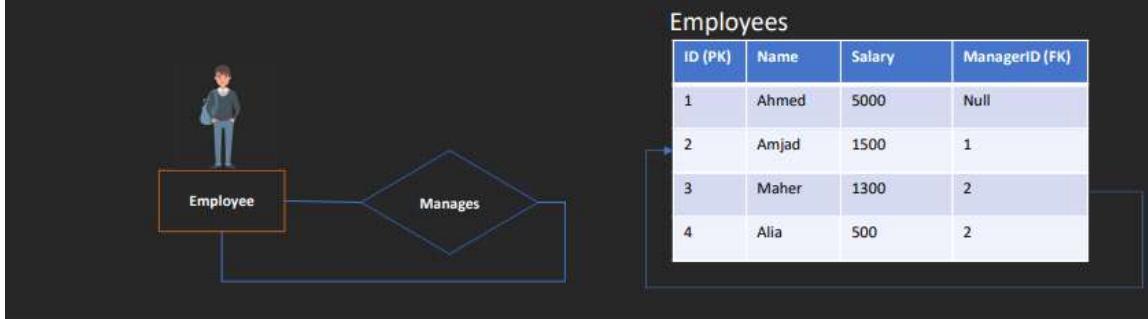


# Referencing Relationship

Example below, an employee has only one manager and manager has many employees.

This is the relationship between one entity.

If an element of an entity is associated with another element of the same entity, it is called self relationship.



وهنا بيقولك انه العلاقات أنواع ودي انواعها

## Relationship Types

- One-to-One Relationship.
- One-to-Many Relationship.
- Many-to-One Relationship.
- Many-to-Many Relationship.

الواجب

Relationship is always between two entities.

True

False

Relationship can be on one entity and we call it Self Reference Relationship.

True

False

We can have only one relationship between two entities.

True

False

We can have more than one relationship between two entities.

True

False

### **One-to-One Relationship**

هنتكلم عن علاقة ال one to one وهيا انه كل entity بيملك او بيكون ليه علاقة واحد بس من ال entity الثانية ماينفعش يكون ليه علاقة مع اكتر من record

رزي الانسان ماينفعش يكون ليه اكتر من رقم قومي  
والعلاقه دي بتمثل بانك تكتب رقم واحد جنب طرف العلاقة بص في الأمثلة الجايه

## One-to-One Relationship

Example below, both the Student and the Identification-Card entities, and "has" is the relationship between them.

For example, a student has only one identification card and an identification card is given to one person.



## One-to-One Relationship

example below, both the Student and the Person are entities, and "is a" is the relationship between them.

example below, both the Employee and the Person are entities, and "is a" is the relationship between them.



## to-One Relationship

example below, both the Member and the Book are entities, “borrow” is the relationship between them.

can borrow only one book, and book can be borrowed by one member.



## to-One Relationship

example below, both the Traveler and the Seat are entities, “Set” is the relationship between them.

can set on only one seat, and seat can be assigned to one traveler.



## One-to-One Relationship

In the example below, both the Employee and the Desk are entities, and “Set” is the relationship between them.

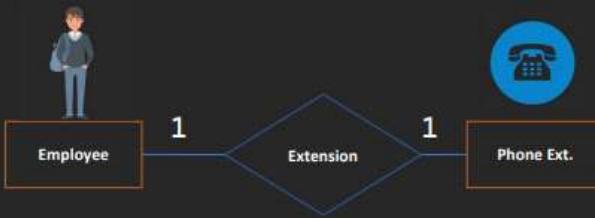
An Employee can set on only one Desk, and Desk can be assigned to one Employee.



## One-to-One Relationship

In the example below, both the Employee and the phone extension are entities, and “Extension” is the relationship between them.

An Employee has a specific phone extension, which can only reach that Employee.



## One-to-One Relationship

In the example below, both the Employee and the Task are entities, "Assigned" is the relationship between them.

For example, one employee can work on a single Task, and a Task can be assigned to one employee.



## One-to-One Relationship

In the example below, both the citizen and the Car are entities, "Own" is the relationship between them.

A citizen can own only one car, a car can be owned by one citizen.



**الواجب**

When a single element of an entity is associated with a single element of another entity, it is called a one-to-one relationship

True

False

### **One-to-Many/Many-to-One Relationship**

علاقة ال one to many وال many to one تقدر entity معينه تقدر ترتبط باكتر من entity تانيه زي مدرس واحد بيدرس لعدة طلاب ودي بتمثل بان ال الأول اللي هوا المدرس بيتحط جنبها الرقم 1 وال الثانيه اللي هيا الطلاب بيتحط جنبها حرف ال m و بتقرا من الشمال لليمين

يعني لو كان المدرس عالشمال فبتقول one to many ولو كان المدرس عاليمين بتقول many to one او علي حسب انت بدأت منين

## One-to-Many/Many-to-One Relationship

If a single element of an entity is associated with more than one element of another entity, it is called a one-to-many relationship. For example, a customer can place many orders, but an order cannot be placed by many customers.



## Many-to-Many/Many-to-One Relationship

If more than one element of an entity is related to a single element of another entity, then it is called a many-to-one relationship.

For example, an employee can work on a single project, but a project can have many employees.



## One-to-Many/Many-to-One Relationship

example below, both the Member and the Book are entities, “borrow” is the relationship between them.

Member can borrow Many books, and book can be borrowed by one



## One-to-Many/Many-to-One Relationship

example below, both the Customer and the Mobile-Line are entities, and “Subscribe” is the relationship between them.

Customer can subscribe to many lines, but each line can only be subscribed by one.



## One-Many/Many-to-One Relationship

example below, both the citizen and the Car are entities, "Own" is the relationship between them.

A citizen can own many cars, a car can be owned by one citizen.



## One-Many/Many-to-One Relationship

If one element of an entity is associated with one element of another entity, it is called self relationship.

For example, an employee has only one manager and manager can manage many employees.



الواجب

When a single element of an entity is associated with more than one element of another entity, it is called a one-to-many relationship.

True

False

When more than one element of an entity is associated with a single element of another entity, it is called a many-to-one relationship.

True

False

### Many-to-Many Relationship

علاقة ال many to many وهي تكون لما تقدر تربط اكتر من entity باكتر من entity في جدول ثاني مثلا اكتر من طالب يقدروا يشتراكوا باكتر من كورس والكورس الواحد يقدر يشترك فيه اكتر من طالب وتمثل العلاقة بكتابه ال m بجانب طرف العلاقة

حالة انك تحدد ايه اكتر عدد ممكن يدخل في العلاقة ده اسمه cardinality  
شوف الامثله

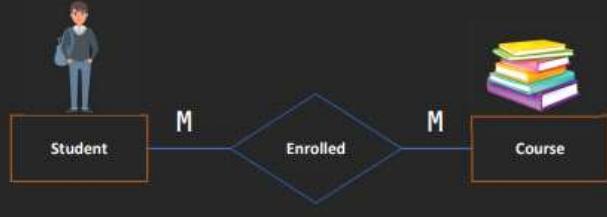
# Many-to-Many Relationship

If more than one element of an entity is associated with more than one element of another entity, it is called a Many-to-Many relationship.

In the example below, both the student and the course are entities, and "Enrolled" is the relationship between them.

A student can be enrolled in many courses.

A course can be studied by Many students.



# One-to-Many Relationship

In the example below, both the customer and the order are entities, and "Place" is the relationship between them.

A customer can have more than one product.

An order can be placed in many orders.



الواجب

When more than one element of an entity is associated with more than one element of another entity, it is called a Many-to-Many Relationship.

True
False

### **Cardinality vs Ordinality**

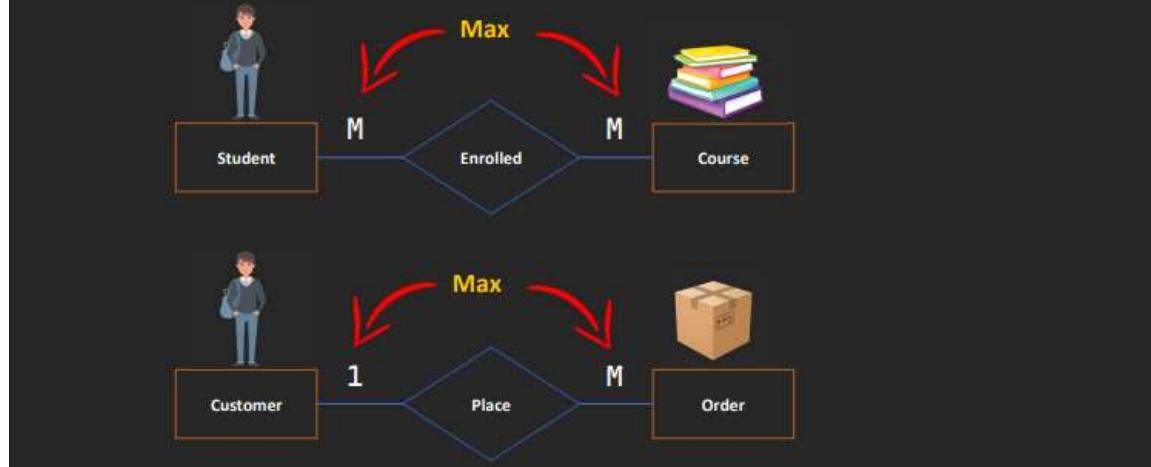
لما كنا بنحدد نوع العلاقة قبل كده كنا بنسأل نفسنا ايه اكبر عدد من ال entities في الجدول الثاني يقدروا يرتبطوا ب entity واحد في الجدول الأول والعكس زي ايه عدد الكورسات اللي الطالب يقدر يشتراك فيها وايه اكبر عدد من الطلاب يقدروا يشتراكوا في الكورس الواحد ده كده اسمه cardinality انك تسال ايه اكبر عدد مطلوب لتحقيق العلاقة بين جدولين

ال ordinality هو العكس انك تسال عن اقل عدد من ال entities مطلوب لتحديد العلاقة

زي مثلا انه العميل اقل عدد من ال orders اللي المفروض يعملاها هو صفر لكن اقل عدد من العملاء اللي لازم يطلبوا ال order ده هو واحد لانه الاوردر هو وطالما فيه اوردر اتعمل يبقى لازم يكون فيه عميل هو اللي طلبه ال cardinality وال ordinality ممكن يتكتبا مع بعض زي كده (min,max) علي جانب طرفي العلاقة

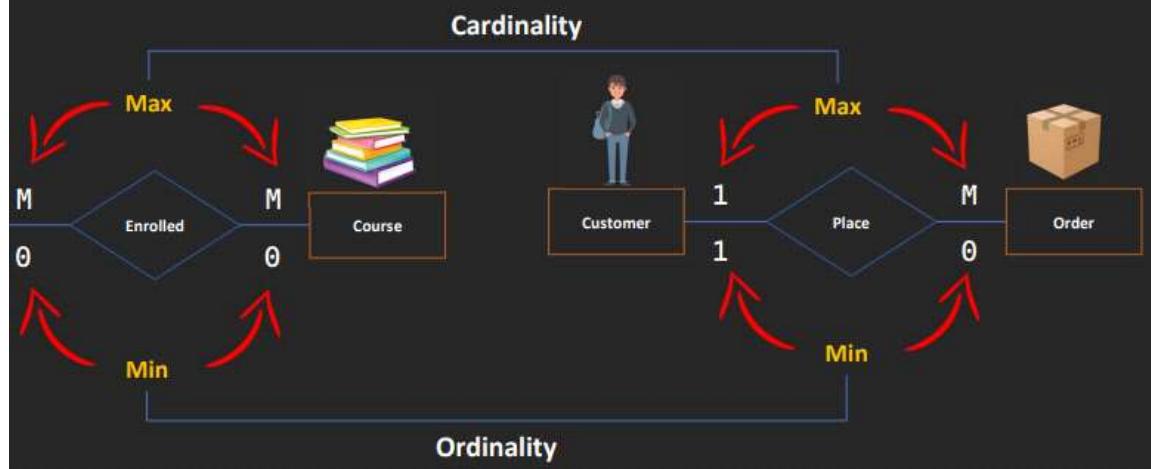
# Cardinality

Cardinality refers to the maximum number of times an instance in one entity can relate to instances of another entity



# Ordinality

Ordinality, on the other hand, is the minimum number of times an instance in one entity can be associated with an instance in the second entity. (in other words It specify if it's optional/mandatory/required or not).



# Cardinality and Ordinality

represent it like this:



**الواجب**

Cardinality refers to the maximum number of times an instance in one entity can relate to instances of another entity.



Ordinality, on the other hand, is the minimum number of times an instance in one entity can be associated with an instance in the related entity. (in other words It specifies if it's optional/mandatory/required or not).



(0,M) : the first number represents the cardinality and the second one represents the ordinality.

True

False

(0,M) : the first number represents the ordinality and the second one represents the cardinality.

True

False

When you ask "What is the Max?" you are identifying the cardinality.

True

False

When you ask "What is the Min?" you are identifying the ordinality.

True

False

### **Cardinality Symbols and Practices**

هوا بيقولك انه لما حد يقولك cardinality هوا بيقصد الاتنين الـ

## ordinality

وان ال ordinality موجوده عشان تحدد انه ال لازم يكون الطالب مشترك في كورس ولا لا يعني لازم تحدد entity من الجدول الثاني ولو لازمكم عدد ال entities دي

وكنا بنرمز لل cardinality بالطريقه دي  لكن بيقولك انه فيه symbols او رموز تانية لتوضيح العلاقة زي دي مثلا

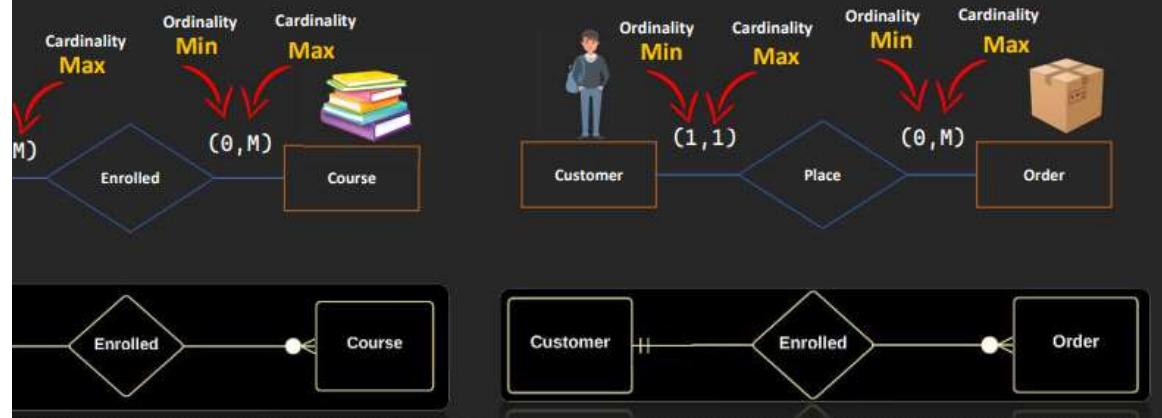


ال symbole دي اسمها crow-s-foot-notation يعني رجل الغراب احنا بنسميها رجل الفرخه

 والدایرة بترمز للصفر  لأنما الجزء ده بيرمز للجزء ده m وده بيرمز للرقم واحد  الجزء القريب من المستطيل او ال entity يرمز لل cardinality او ال max انما  الجزء بعيد بيرمز لل ordinality او ال min بص كده للمثال ده

# Cardinality and Ordinality

represent it like this:



وedo 3 انواع من الرموز

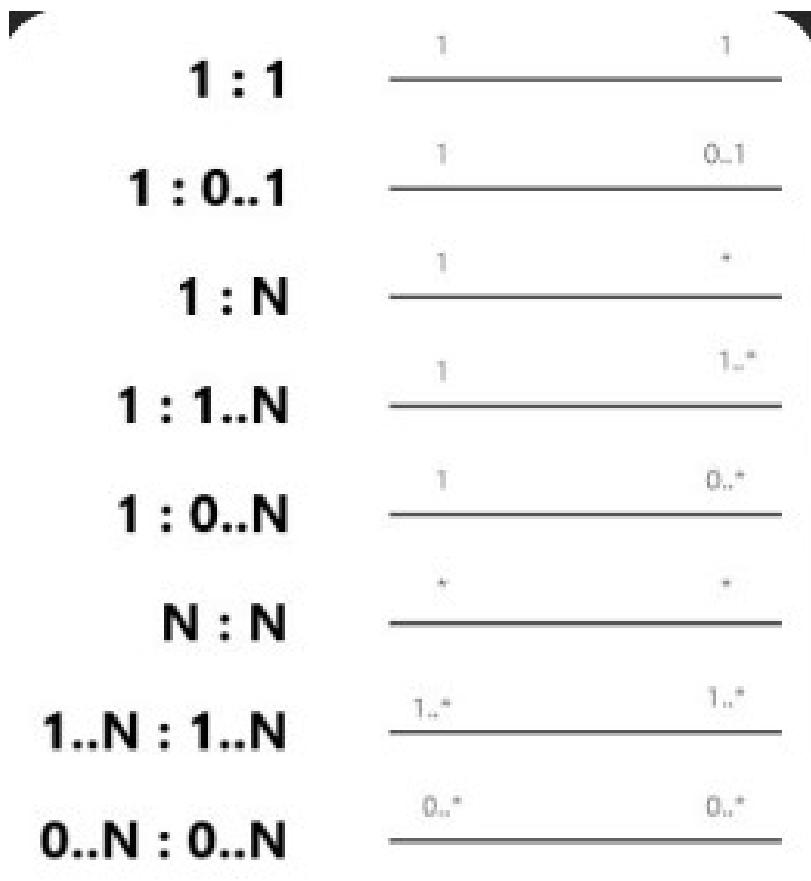
## Cardinality Notation Symbols:

### crow-s-foot-notation

<b>1 : 1</b>		One to One يرتبط بواحد بس من الجدول الثاني
<b>1 : 0..1</b>		One or One to zero أول يرتبط مع واحد بس او صفر من الجدول الثاني الثاني ال cardinality وال ordinality واحد
<b>1 : N</b>		One to Many يمكن ربطه مع اكتر من عنصر بالجدول الثاني
<b>1 : 1..N</b>		many or One to one جدول الأول يرتبط مع عنصر او اكتر من الجدول الثاني

<b>1 : 0..N</b>		many or One to zero جدول الأول مايرتبطش مع حد خالص او يرتبط مع أي عدد من العناصر من الجدول الثاني
<b>N : N</b>		Many to many يربط مع أي عدد من الجدول الثاني
<b>1..N : 1..N</b>		One or many to one or many احد من الجدول الأول يرتبط مع واحد او اكتر من جدول الثاني
<b>0..N : 0..N</b>		Zero or many to zero or many جيبيه ربنا كويس او اللي ييجي منه احسن منه

## min-max-notation

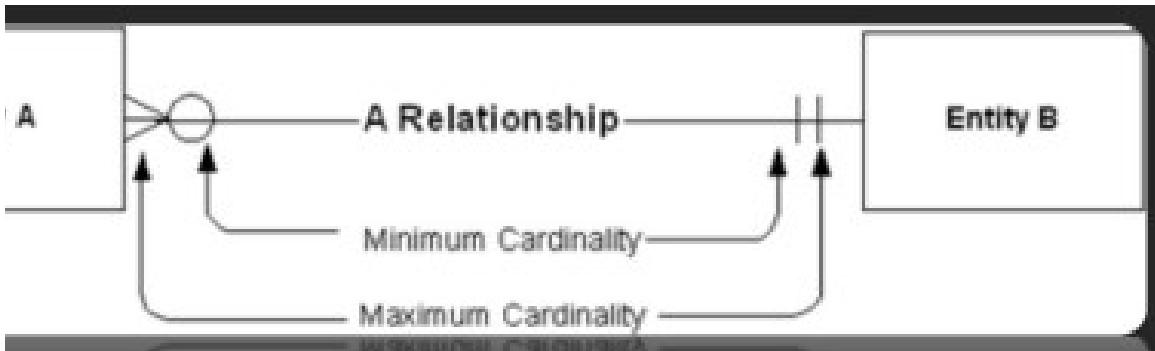


## bachman-notation

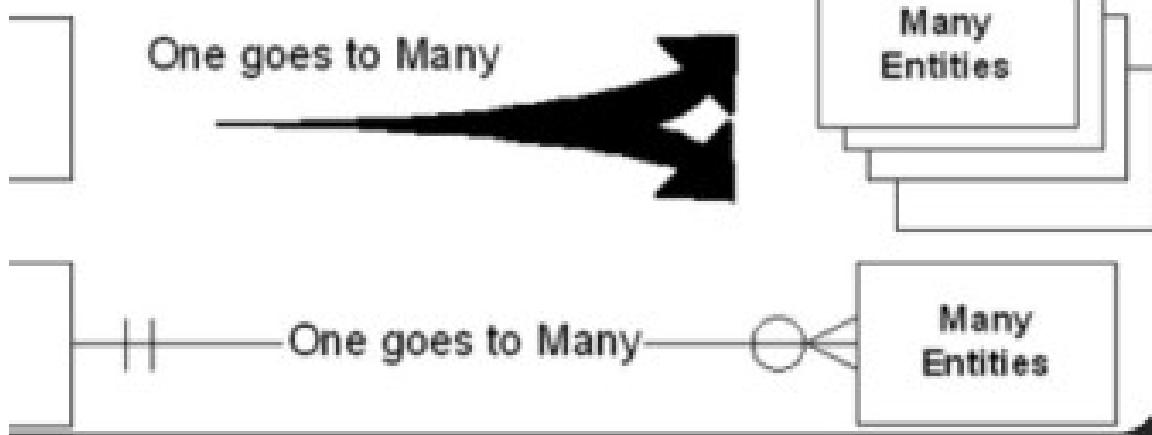
1 : 1	
1 : 0..1	
1 : N	
1 : 1..N	
1 : 0..N	
N : N	
1..N : 1..N	
0..N : 0..N	

وهي أمثلة

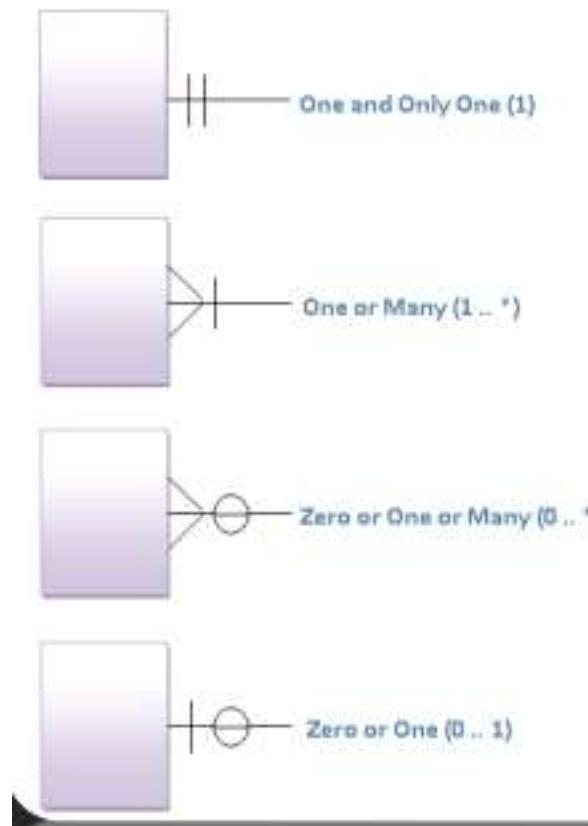
## 's foot notation Symbols



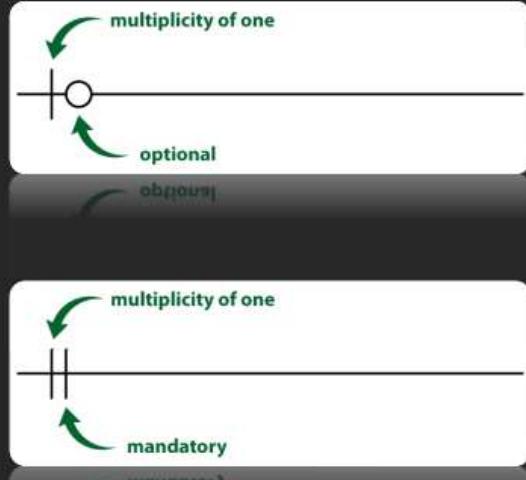
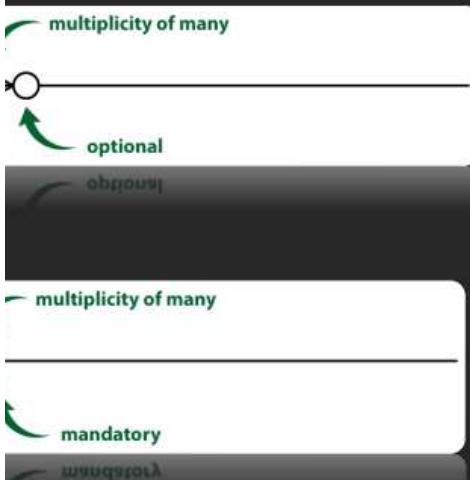
## A Crow's Foot Looks Like "Many"



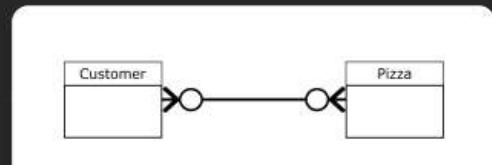
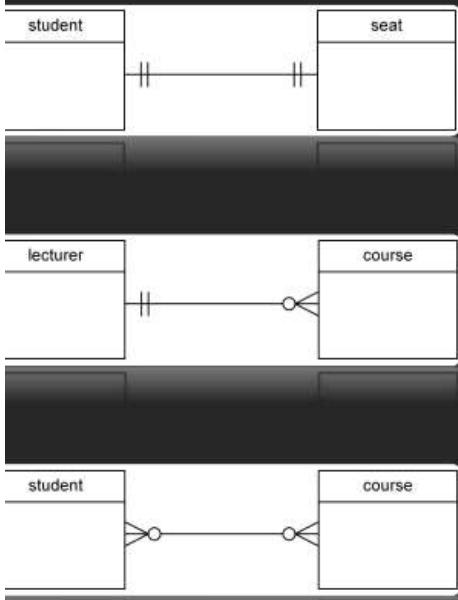
Crow Foot Notation Symbols



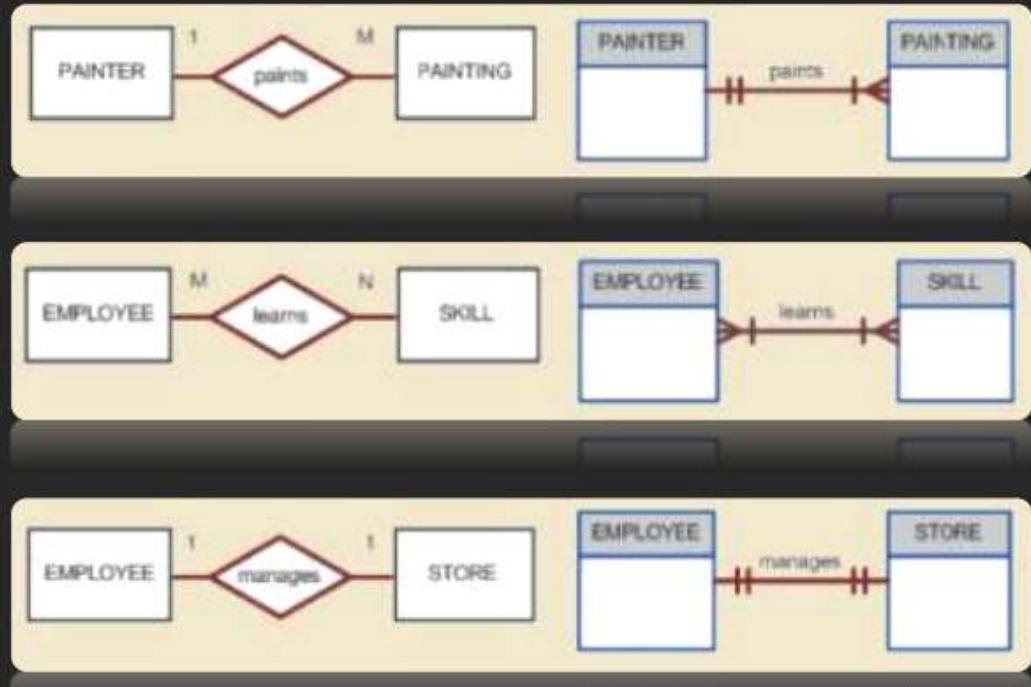
## s foot notation Symbols



## s foot notation Symbols



# 's foot notation Symbols



في اخر مثال هوا جايبلك مثال من النت وبيورييلك انه مجرد مثال اكاديمي لكن في ارض الواقع الكلام ده مش صح لانه لو كل موظف بيدير فرع طيب والعمال والسكرتاريه والمحاسبين وباقى الموظفين وديتهم فين يااما انت بقى شركتك عباره عن محلات وجایب في كل محل مرموطون بيعملوك كل حاجه وده شيء خيالي او انك فاتح ورشه وشغال فيها لوحد وجایبلي سيسystem وبرضه ه تكون العلاقة غلط تتحط كده وانه الصح انه تكون العلاقة one or one to zero or one

## Total Vs Partial Participation

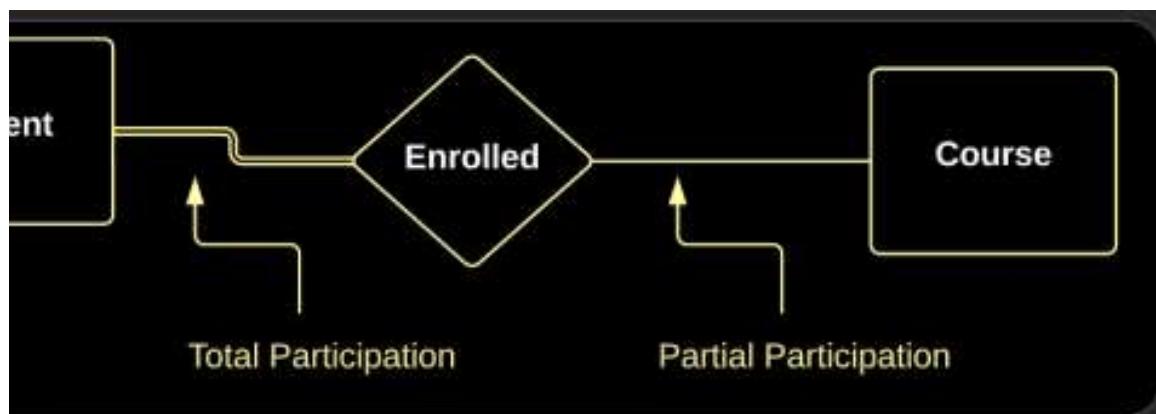
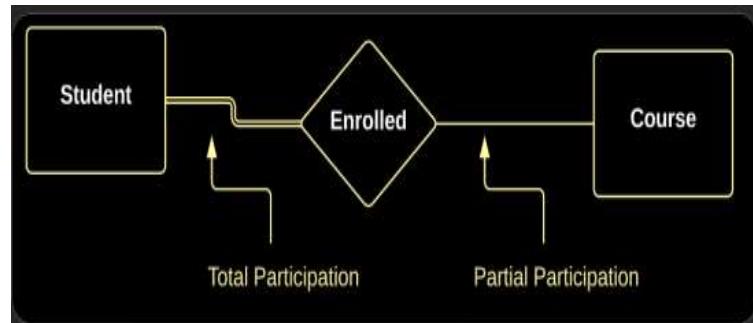
بيقولك انه زمان كانوا بيستخدموا ال total participation وال partial participation

ال total participation معناها انه كل عنصر من الجدول لازم يكون مرتبط بعنصر او اكتر من الجدول الثاني بغض النظر عن نوع العلاقة نفسها يعني كل طالب لازم غصب عنه ياخذ كورس وما عندناش طلاب في الشارع

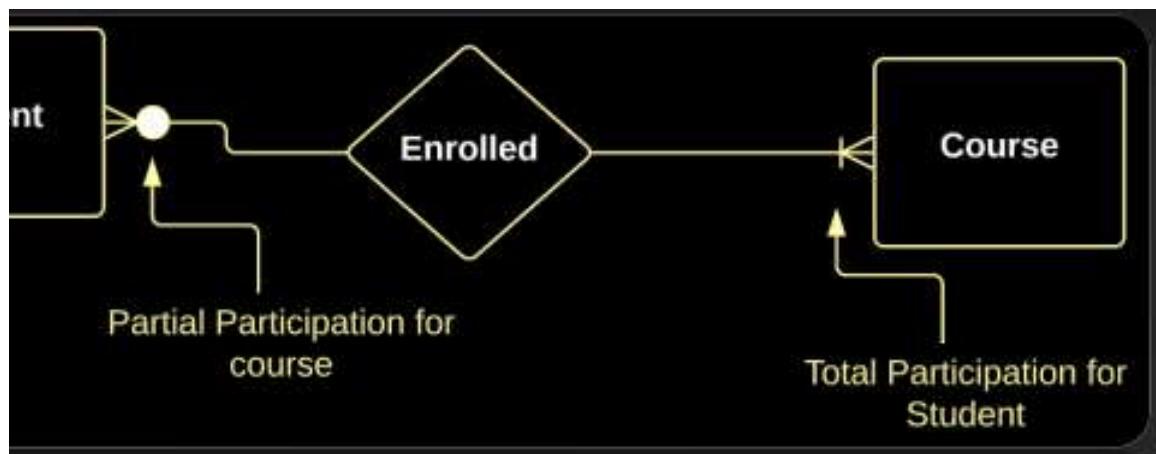
ال partial participation انه مش لازم كل العناصر من الجدول الأول تكون مرتبطه بعناصر من الجدول الثاني زي انه الكورسات مش لازم يكون فيه طلبه

مسجلين فيها

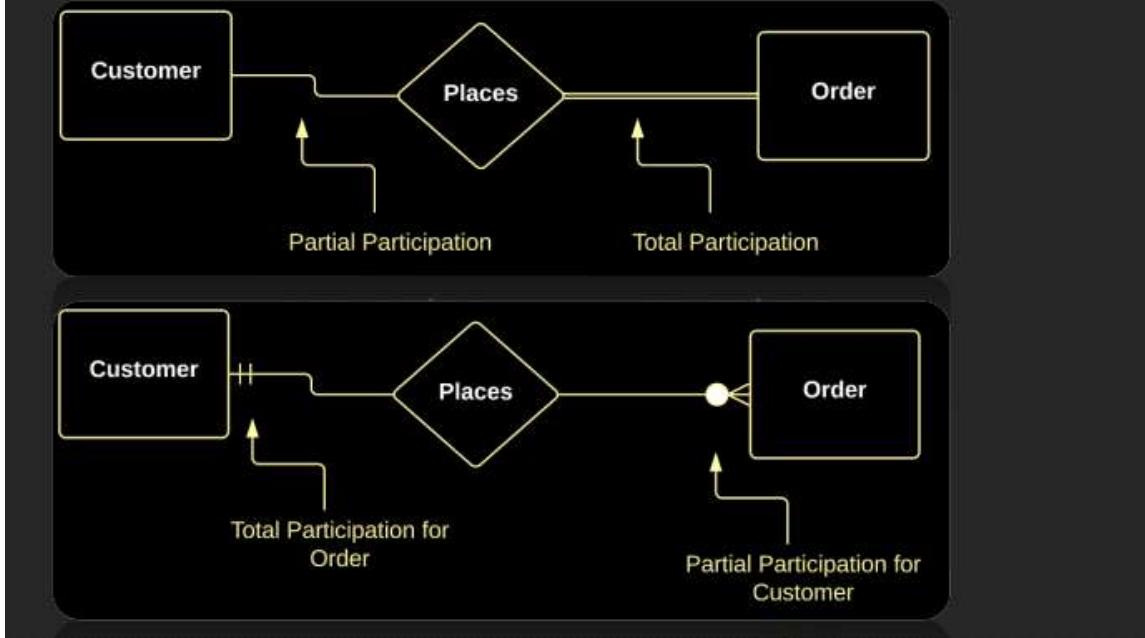
وهي تتحقق عبارة عن انه يجيلك في ناحيه من العاشه ويرسم خطين بدل خط واحد زي  
كده



ال الي فوق ده احنا بنمته كده



# Full participation vs Partial participation

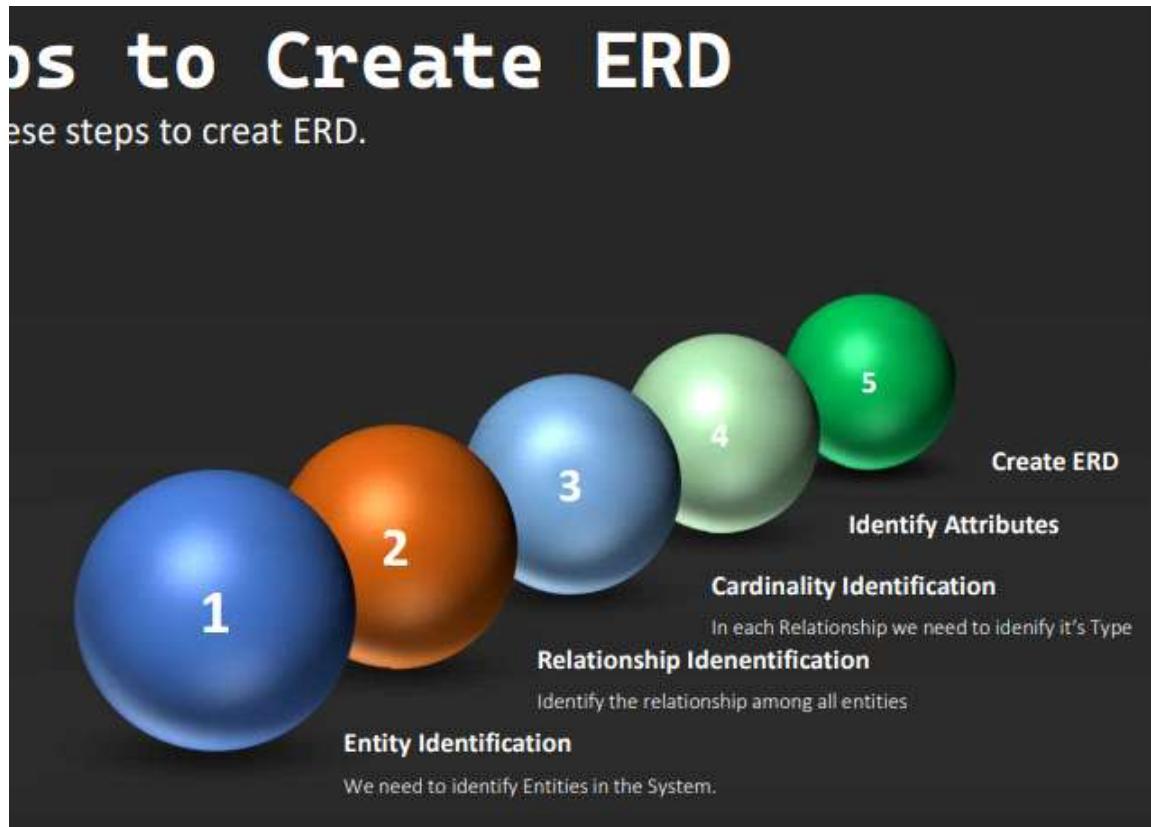


## Process of Creating ERD Step by Step - Small Project

هنا بيقولك انك عشان تطلع erd عندك خطوات بتتمشى عليها واحده واحده هوب  
تلاقي نفسك عامل ال :-

- وهيا انك تحدد ايه الجداول اللي تحتاج تعمليها  
بناءا علي المتطلبات بتاعت المشروع
- هل يوجد علاقه بين الجداول ببعضها ولا  
لا ؟
- لو وجدت علاقات ايه هوا نوع العلاقات  
دي
- انك تحدد الاعمده بتاعت كل جدول  
attributes Identify

• مبروك عليك ال diagram



تعالي نعمل ال diagram للمشروع ده :-

## Create ERD for University

nts:

In University, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.

A student has only one mentor(prof), Professor can mentor more than one student.

A professor can teach only one course.

A course can have only one professor.

A professor can have un-assigned professor.

A professor can teach no courses.

اول خطوه انك تحدد ال entities وهيا أي حاجه محتاج تخزن عنها معلومات  
فاحنا دوقتي محتاجين entities لـ student وال courses وال professor

## 1:

### Requirements:

- In a university, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.
- Each student has only one mentor(prof), Professor can mentor more than one student.
- Each Professor can teach only one course.
- Each Course can have only one professor.
- Course can have un-assigned professor.
- Professor can teach no courses.

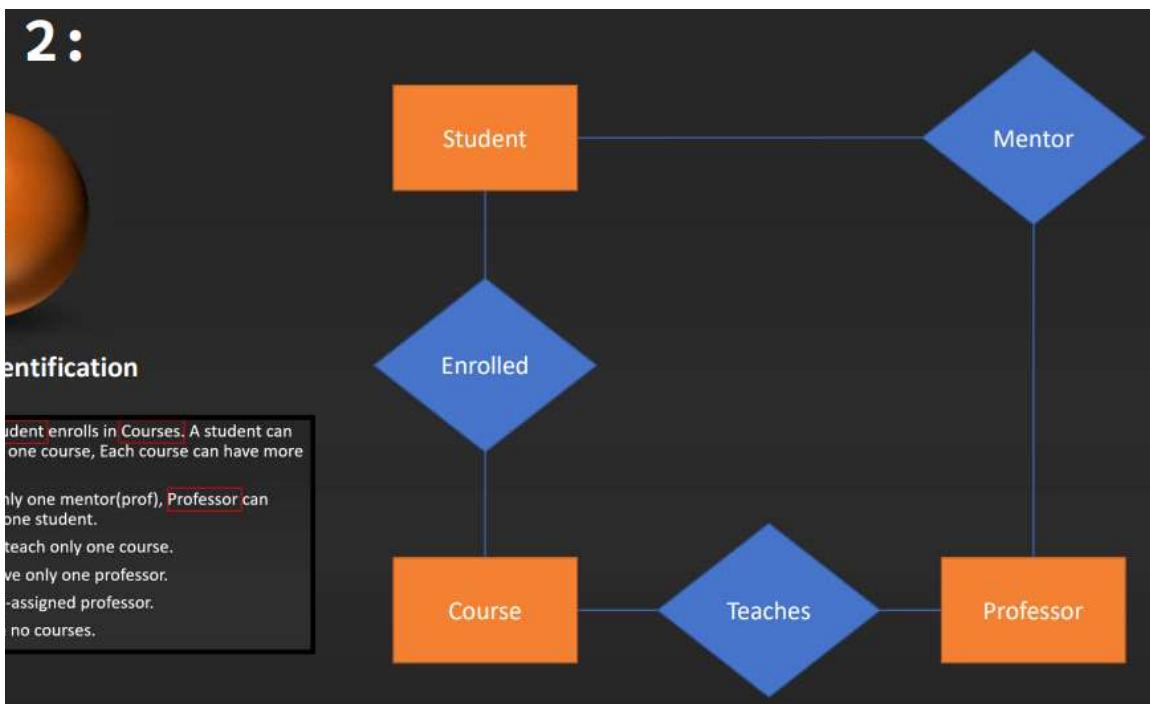
Student

Course

Professor

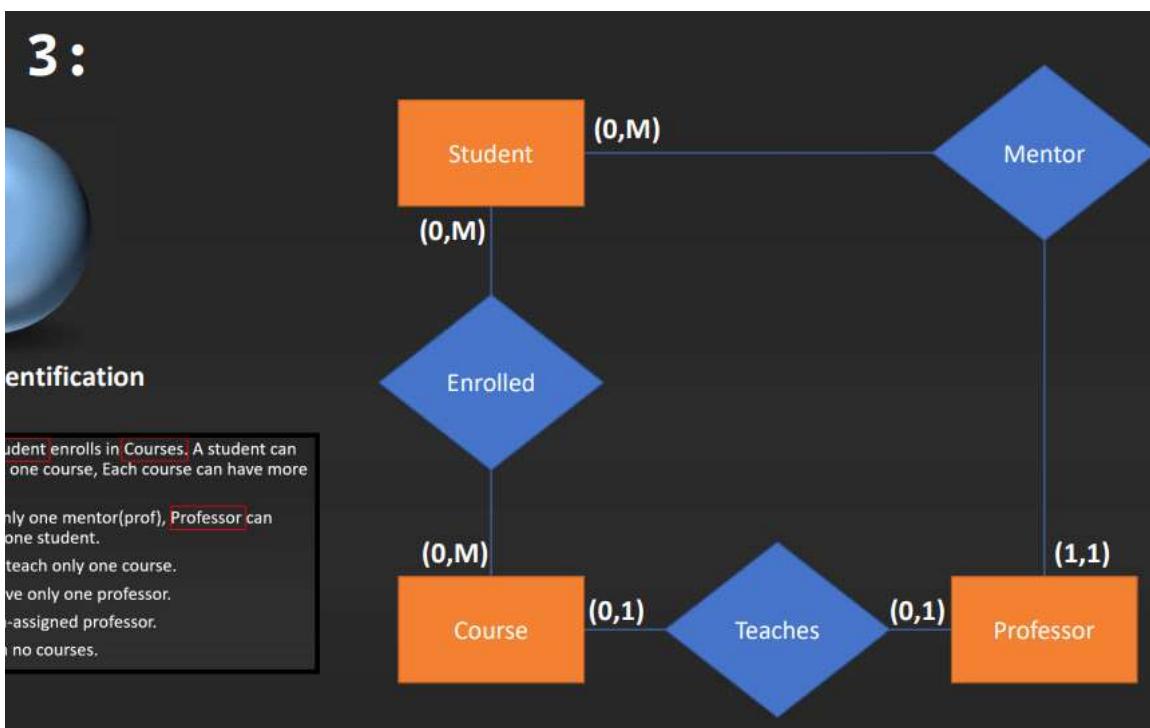
بعدين تعالى نشوف هل فيه علاقات بين ال entities وبعضها ولا لا

2:



بعد كده بنحدد نوع كل علاقه واي شيء لم يذكر فهو optional

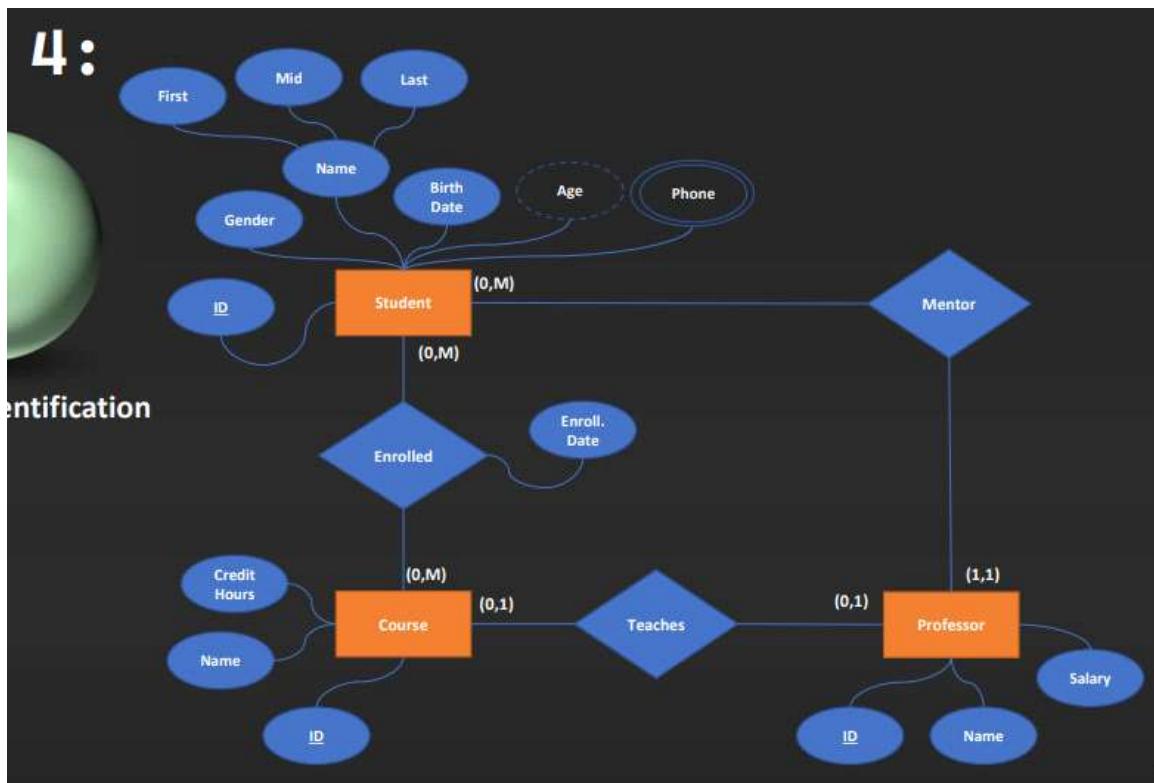
3:

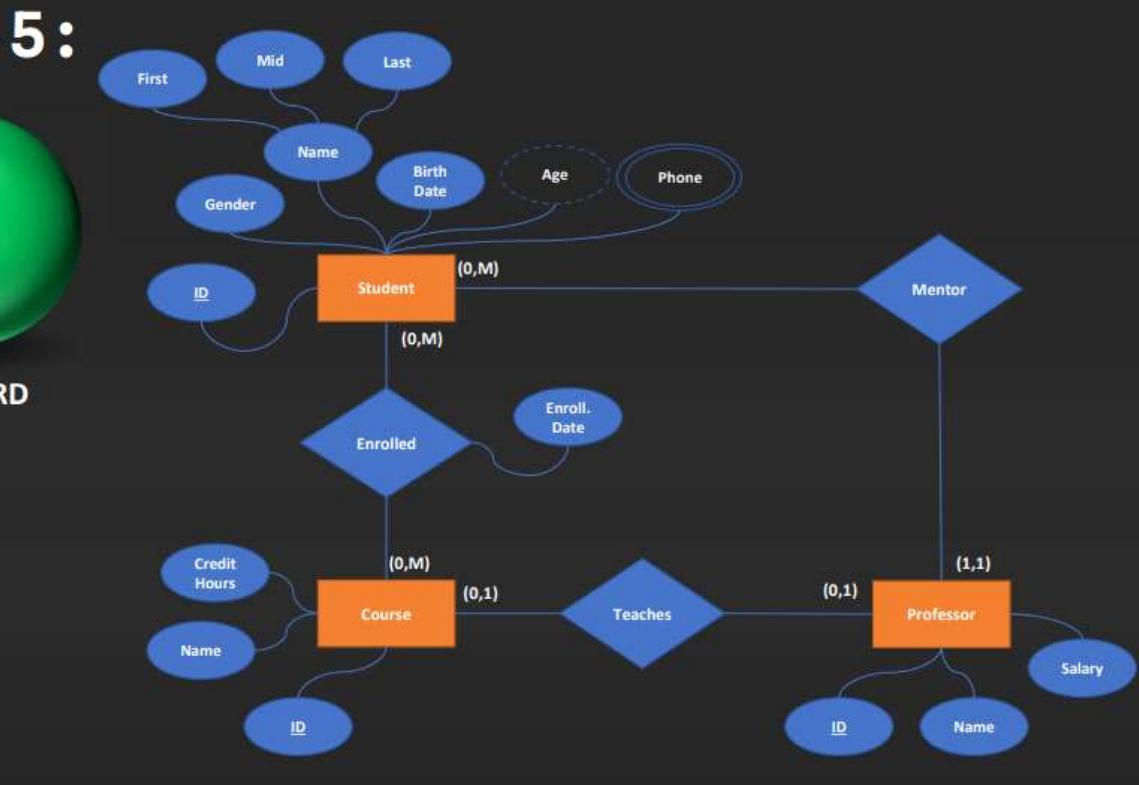


بعد كده هنحدد الاعمده بتاعت كل جدول

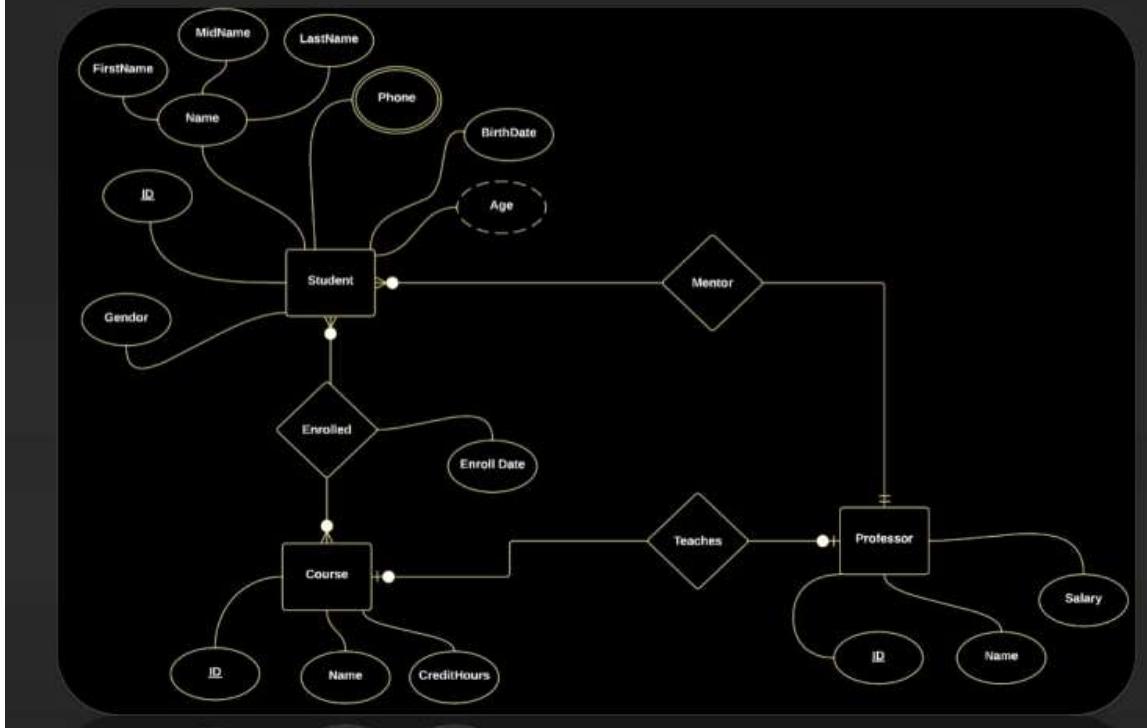
وبتسال بعدين هل لما بيحصل حدث معين زي انه الطالب يسجل في كورس هل فيه معلومات زياده تحتاج تخزنها ؟

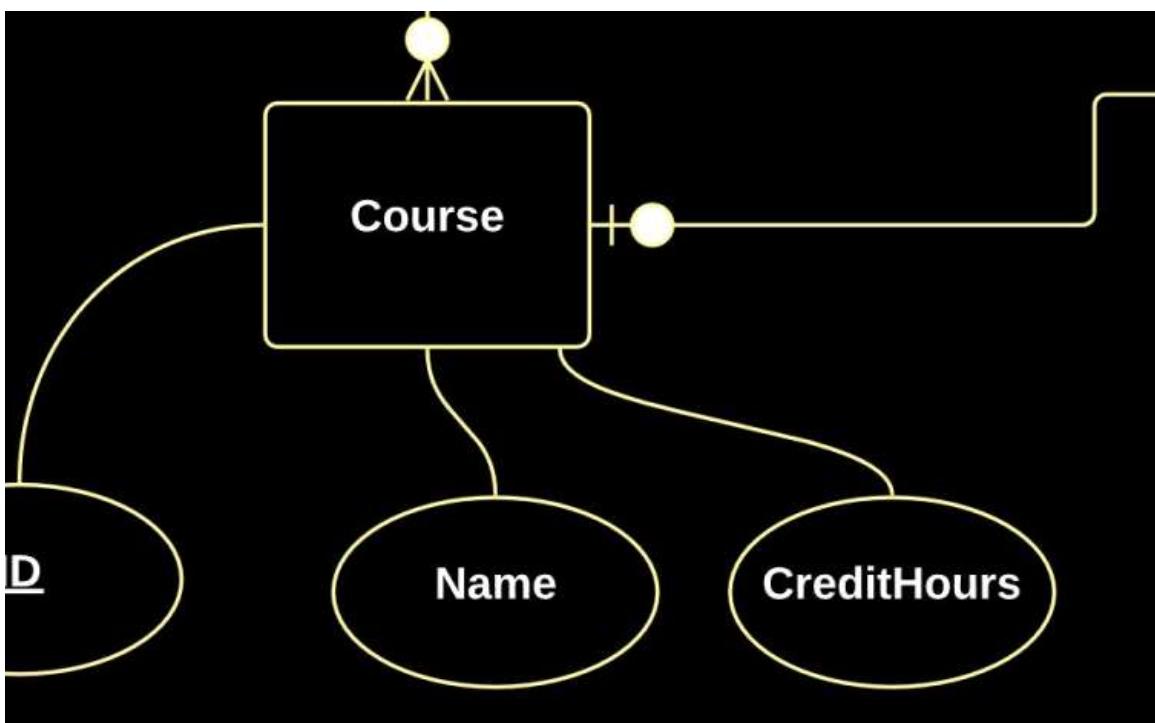
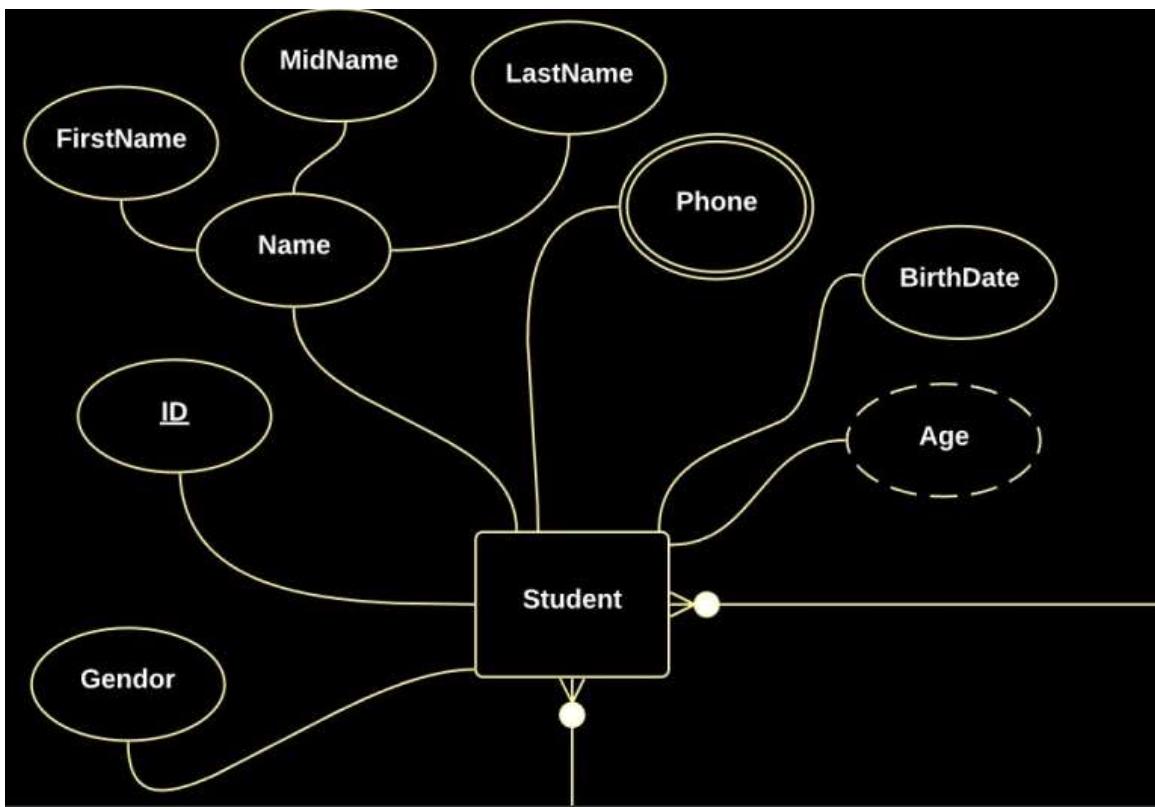
هقولك اه التاريخ بتاع الاشتراك ده مهم بعمله attribute بس بحطه عال نفسها

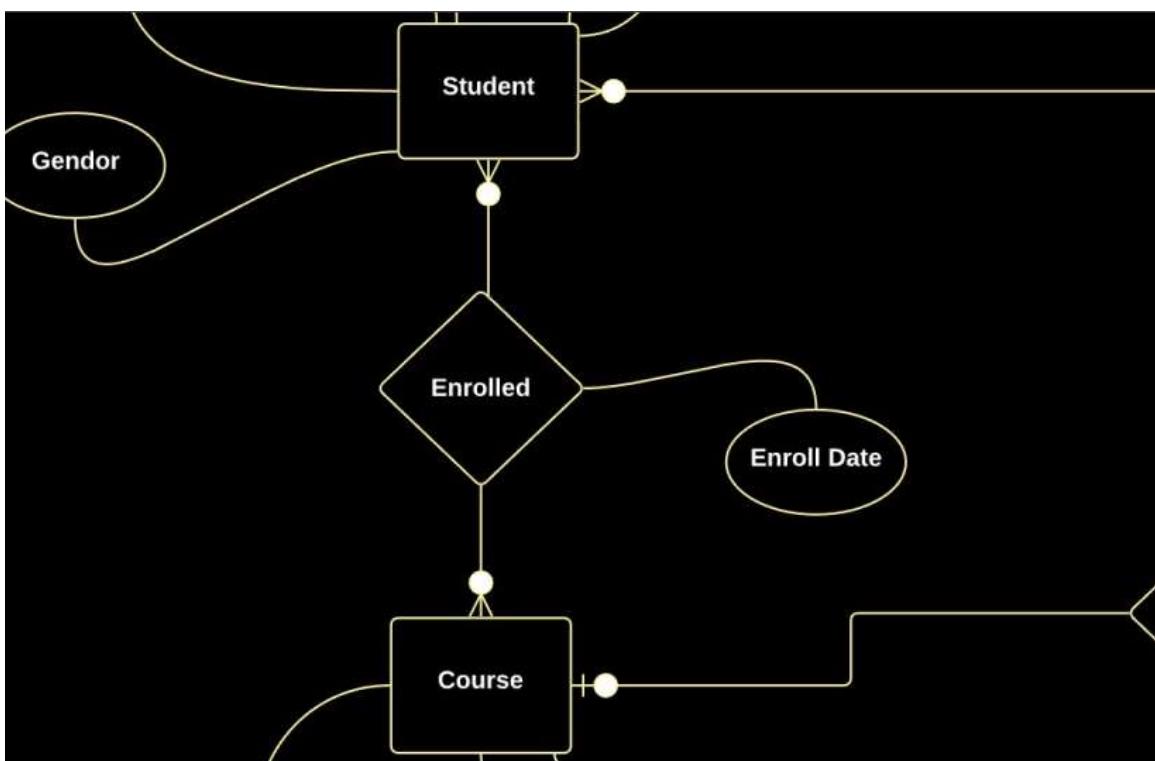
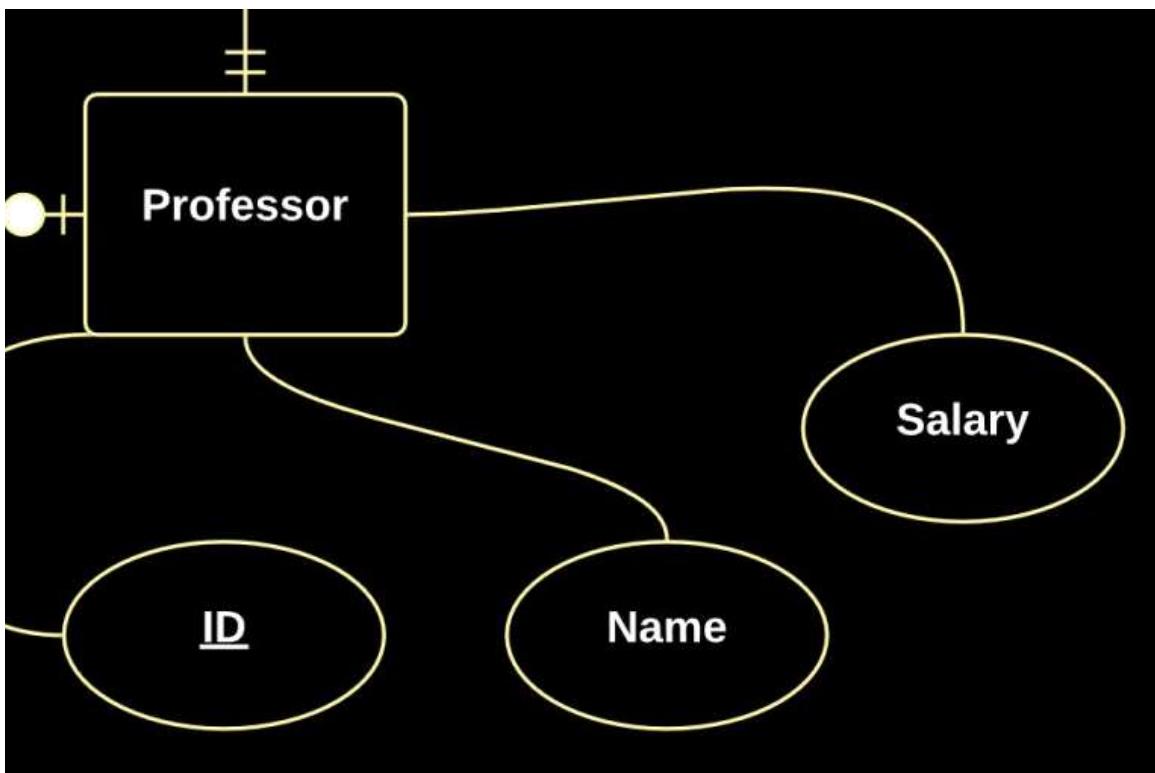


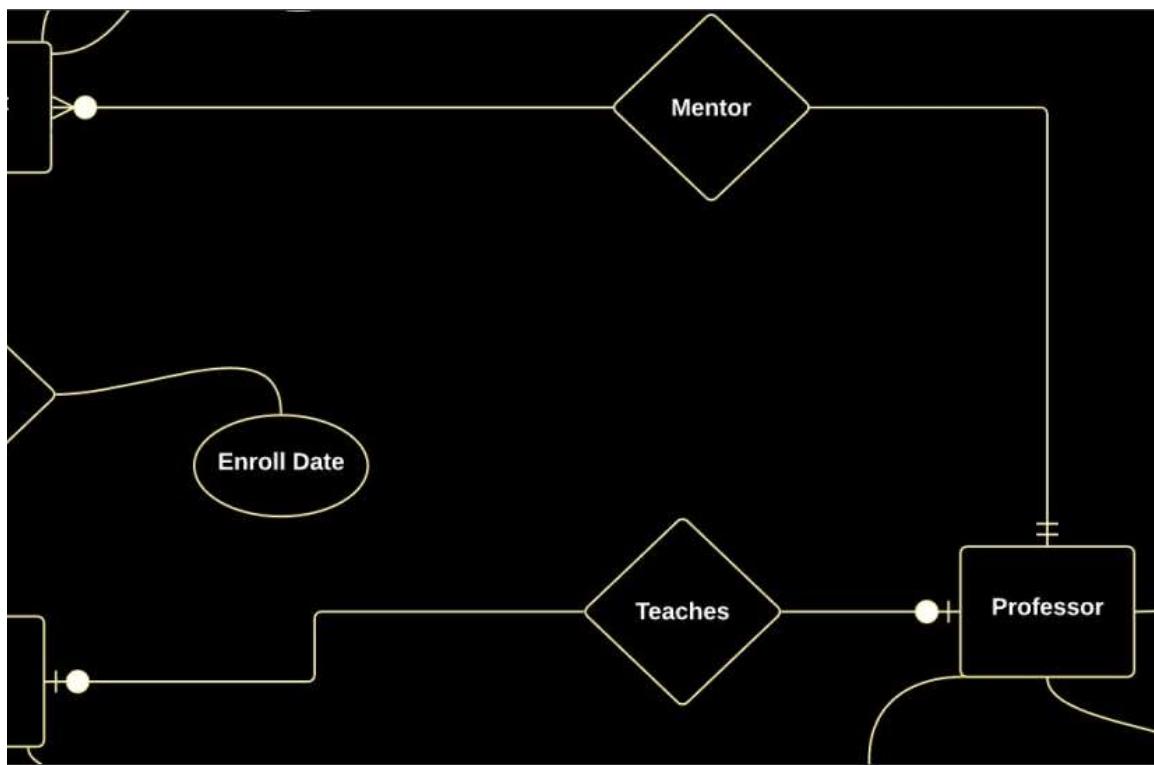


#### Using Crow's foot Notation:









### Recommended ERD Software to Use

يمكنكم استخدام هذا البرنامج سيسهل عليكم رسم ال

### **ERD**

البرنامج موجود اونلاين على الرابط التالي

<http://erdplus.com>

[/http://erdplus.com](http://erdplus.com)

### Aggregation / Associative Entities

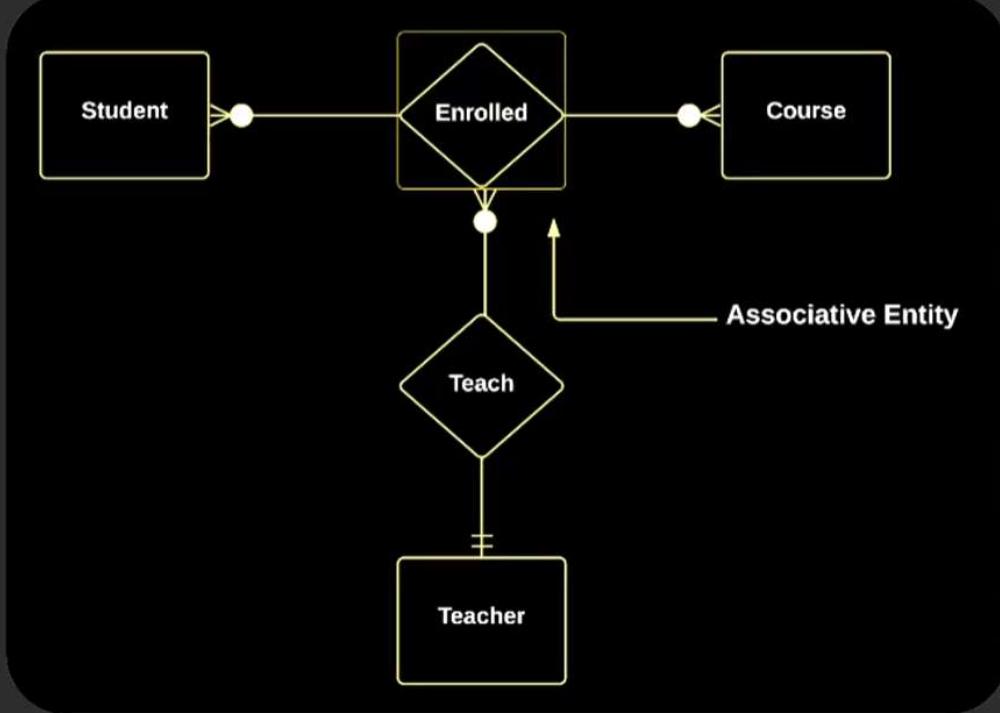
ال associative entities هي علاقه بتتبني على علاقه زي ان المدرس بيدرس الماده للطالب هنا فيه علاقه غير مباشره بين المدرس والطالب والקורס لانه التدريس مش هيحصل الا لما يتواجد علاقه بين الطالب والקורס وهيا انه الطالب

يشترك في الكورس

في الحاله دي انت بتحول العلاقه ل entity و بترمز ليها بشكل معين داخل مستطيل

ملحوظه :- ال many to many دائما بتكون ناتجه عن علاقه associative entity

## Associative Entities



الواجب

An associative entity is a type of entity in a database that is used to model a many-to-many relationship between two other entities. It is also sometimes called a junction table, a linking table, or a cross-reference table.

True
False

By using an associative entity, we can represent complex relationships between entities in a structured and efficient way, without having to duplicate data or create confusing relationships between tables. It allows us to model many-to-many relationships and avoid data redundancy, making it a useful concept in database design.

True
False

Representing relationship between an entity and a relationship which may be required in some scenarios. In those cases, a relationship with its corresponding entities is aggregated into a higher level entity. Aggregation is an abstraction through which we can represent relationships as higher level entity sets.

True

False

Aggregation is a specialized form of association between two or more Entities in which each Entity has its own Existence.

True

False

In aggregation, the relation between two entities is treated as a single entity. In aggregation, relationship with its corresponding entities is aggregated into a higher level entity.

True

False

## **Generalization**

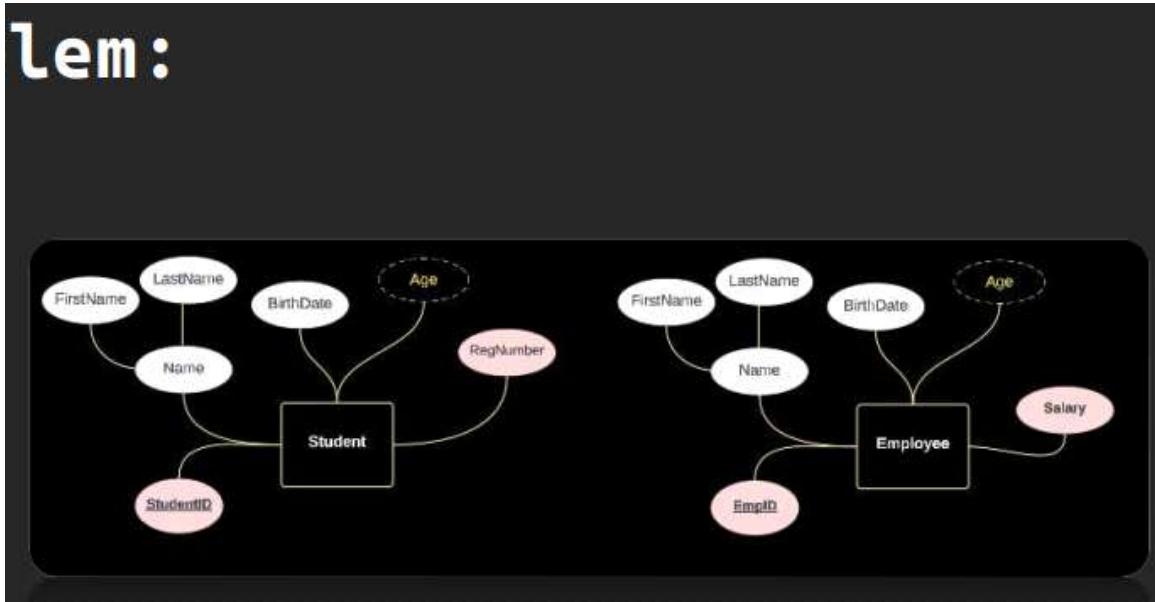
الفكرة فيه زي ما كنت بتعمل كلاس ال person وتعمل كلاس بال client واليورز

بيورثوا منه

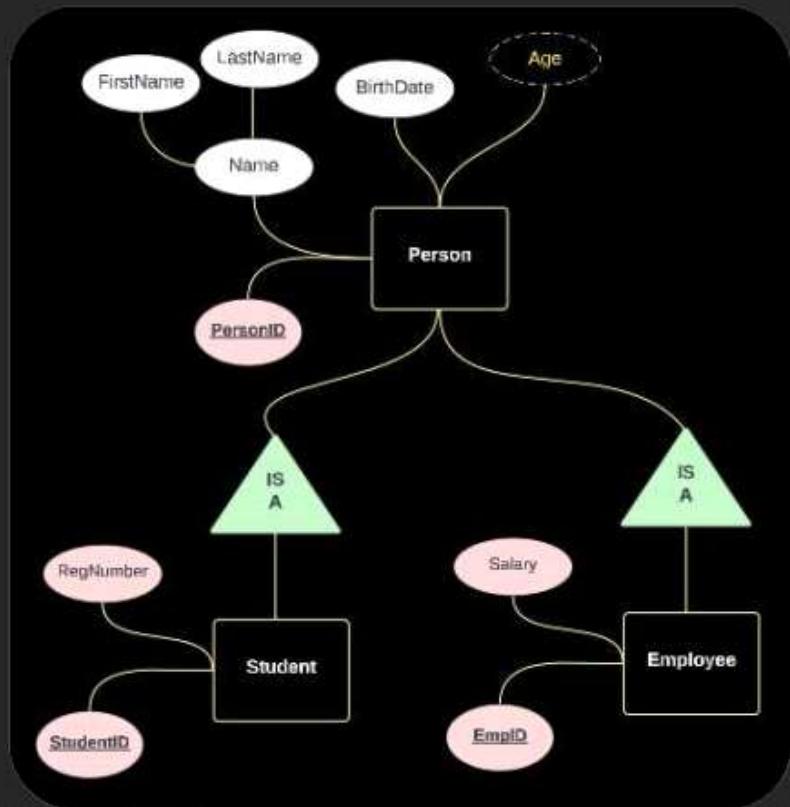
انك لماكون عندك اكتر من جدول بيتقاسموا attributes واحده فبدل ما تكتب نفس الاعمده في الجداول كلها لانت لم الحاجات المشتركه دي وحطها في جدول واحد عام وخلي باقي الجداول تاخذ منه

ماتفكرةش ازاي هنعمل ده في الداتا بيز فكر دلوقتي في ال erd وبس ومالكش دعوه بالداتا بيز

ال generalization بيمثل بمثلث



# Generalization:



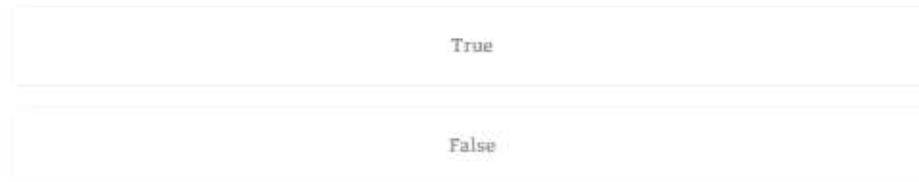
# Generalization

Generalization is the process of extracting common properties from multiple entities and create a generalized entity from it.

bottom-up approach in which two or more entities can be generalized to a higher level entity if they have some attributes in common.

الواجب

Generalization is the process of extracting common properties from a set of entities and create a generalized entity from it.



Generalization is Top-Down approach.



Generalization is Bottom-Up approach.

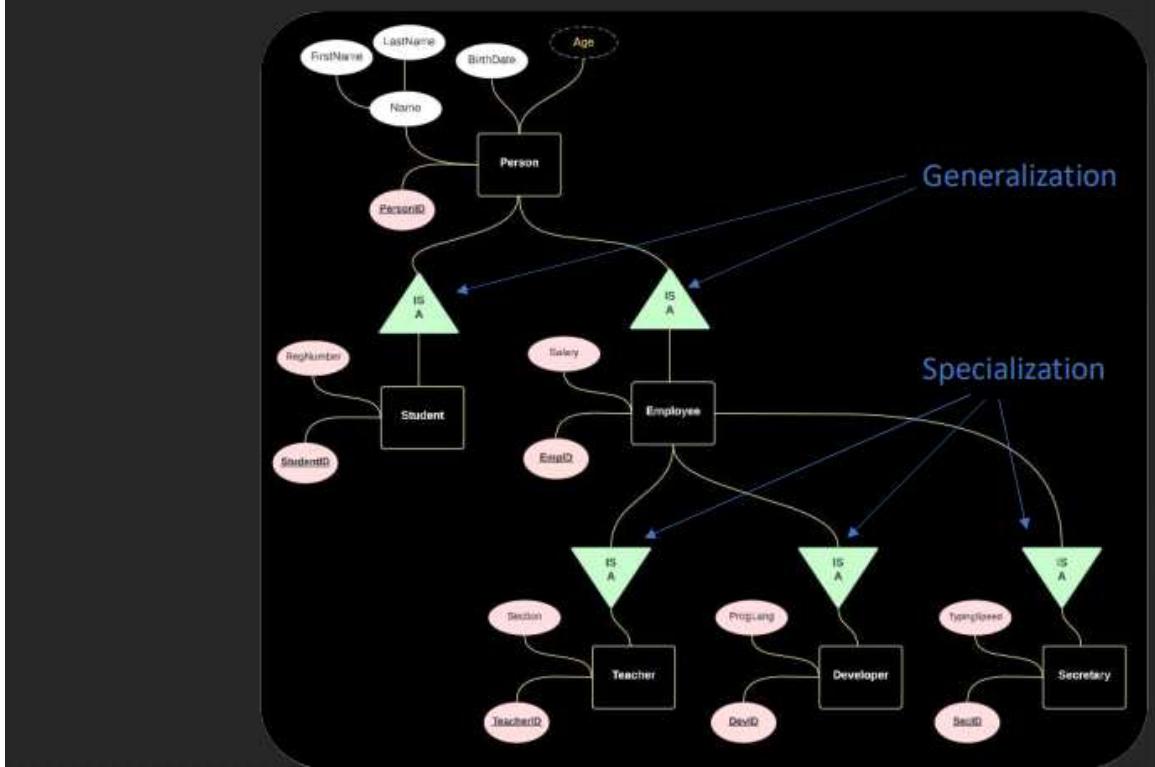


## Specialization

ال generalization هو عكس ال specialization وهو انك بتحدد او بتعرف اكتر او بتقسم ال entities ل entity تانية

الفرق بينهم انه ال generalization انت ممكن تحط فيه بيانات حد من الشارع انما ال specialization لا انت الناس موجوده عندك وبتقسمهم

# Generalization:



# Generalization

In generalization, an entity is divided into sub-entities based on characteristics.

Generalization is a top-down approach in which a higher-level entity is divided into multiple specialized lower-level entities.

الواجب

In specialization, an entity is divided into sub-entities based on their characteristics.

True

False

Specialization is a bottom-up approach

True

False

Specialization is a top-down approach.

True

False

Specialization is a top-down approach in which a higher-level entity is divided into multiple specialized lower-level entities.

True

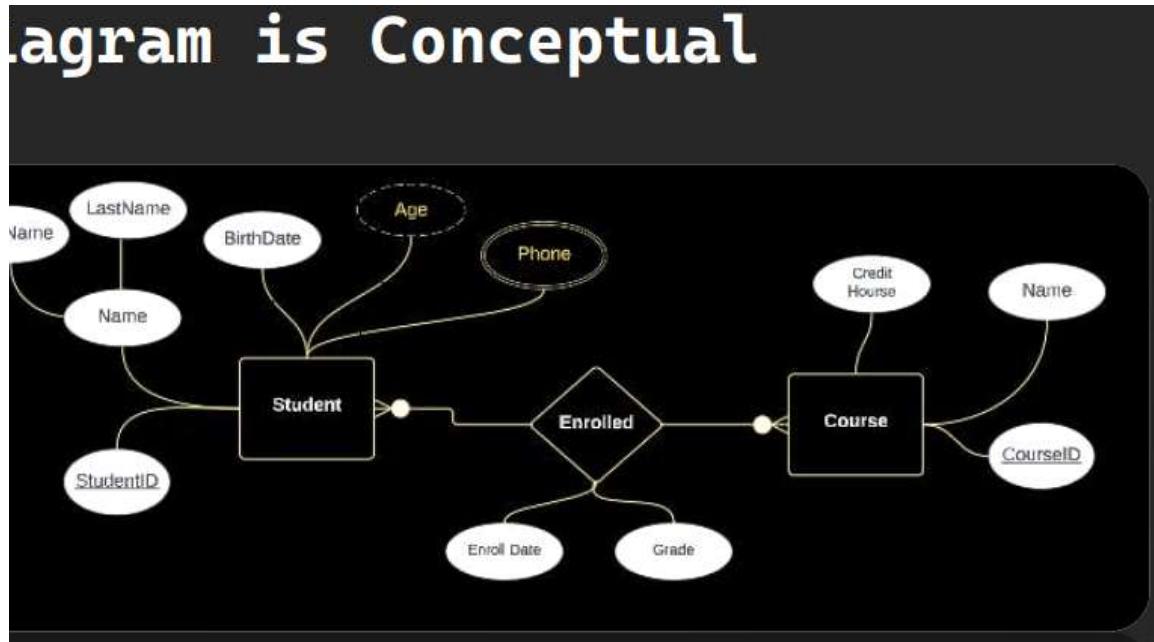
False

### What is Relational Schema?

هذا بيقولك انه ال erd مهم في انه يديك تصور عام لشكل الداتا بيز وبعد ما نرسمه

بنحوله لحاجه اسمها relational schema وهيا جزء من ال logical diagram وبيخلي ال erd اقرب للواقعيه وسهل في تحويله لدادا بيزي

د ه erd

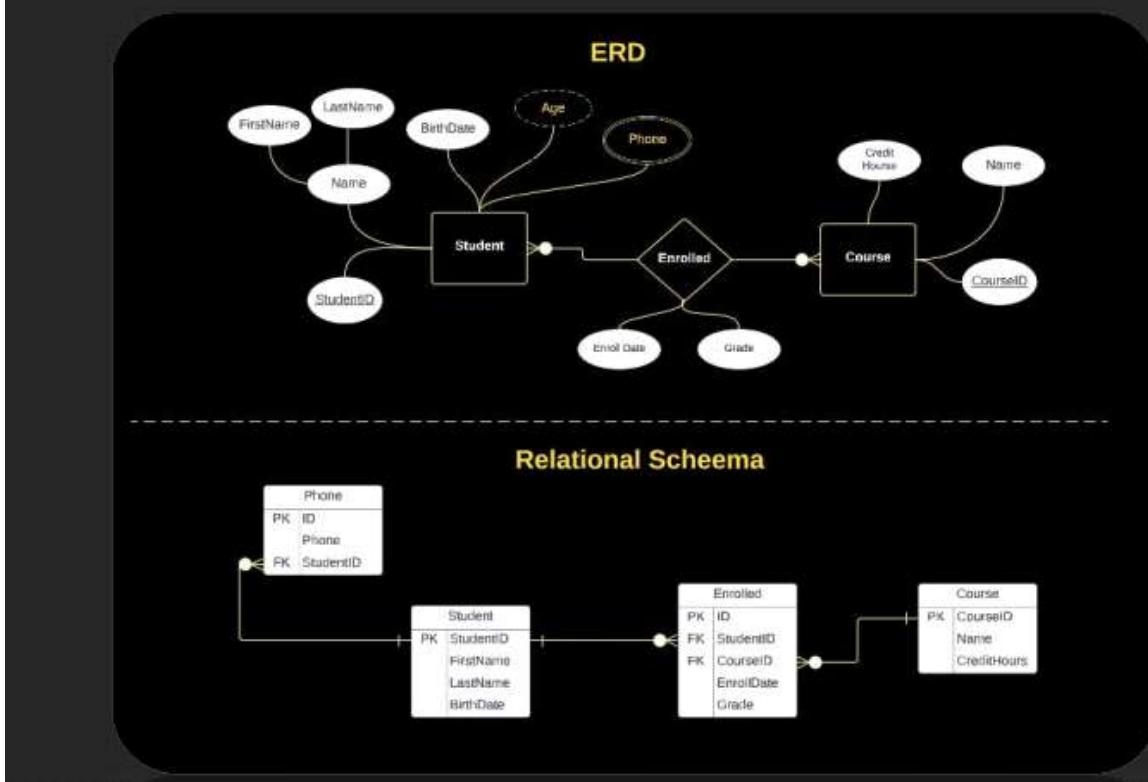


هل بالضرورة انی راسم جدولین یبقي انی هعمل داتا بیز فیها جدولین ؟

قالک لا لانه فيه حاجات تانيه هتحتاج انی اعملها جداول ازای وامتی وفین ده هییجی  
بعدین المطلوب منک انک تعرف انک بعد ماتعمل ال erd بتعمل relational schema

فلما احول ال erd اللي فوق ل relational schema هيطلع كده

# Diagram to Relational Schema



## Relational Schema

A relational schema is a set of relational tables and associated items that are related to one another.

A relational schema defines the design and structure of the relation like it consists of the relation name, set of attributes/field names/column names. Each attribute would have an associated domain.

الواجب

A relational schema is a set of relational tables and associated items that are related to one another.

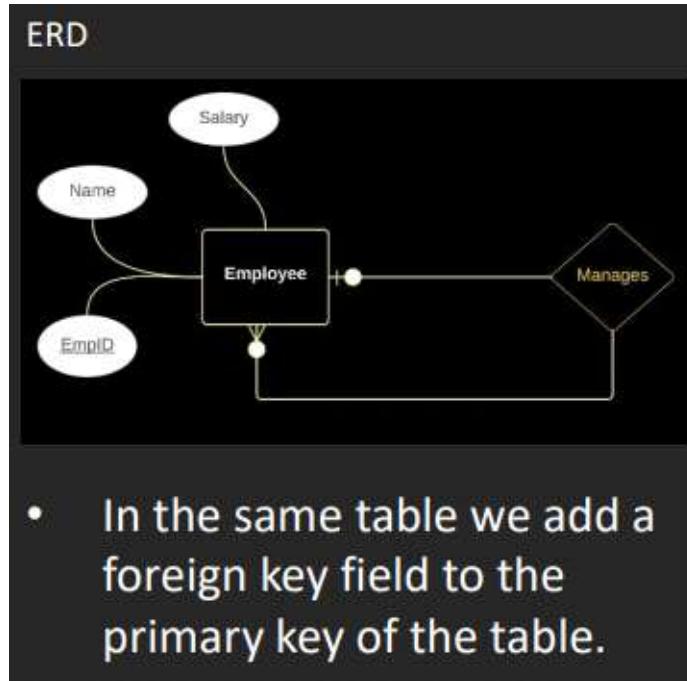
True
False

Relation schema defines the design and structure of the relation like it consists of the relation name, set of attributes/field names/column names. every attribute would have an associated domain.

True
False

### **Convert Self Referential <= Relational Schema**

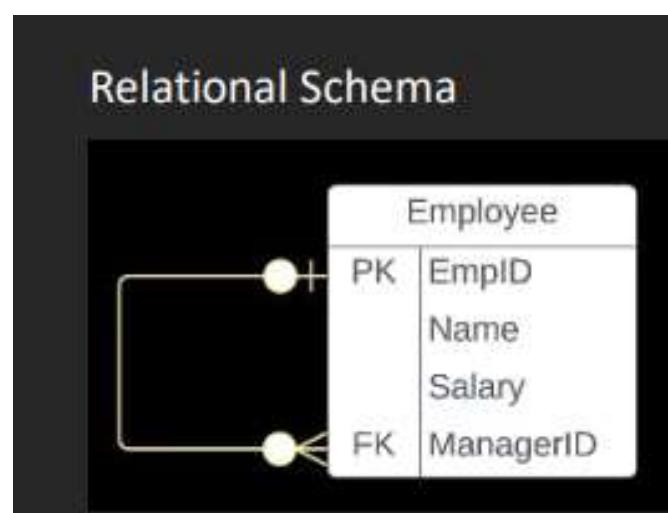
هنبدأ في تحويل ال erd ل relational schema  
وأول حاجة هنبدأ بيها هي ال entity self و هي ال entity اللي لها علاقة مع نفسها

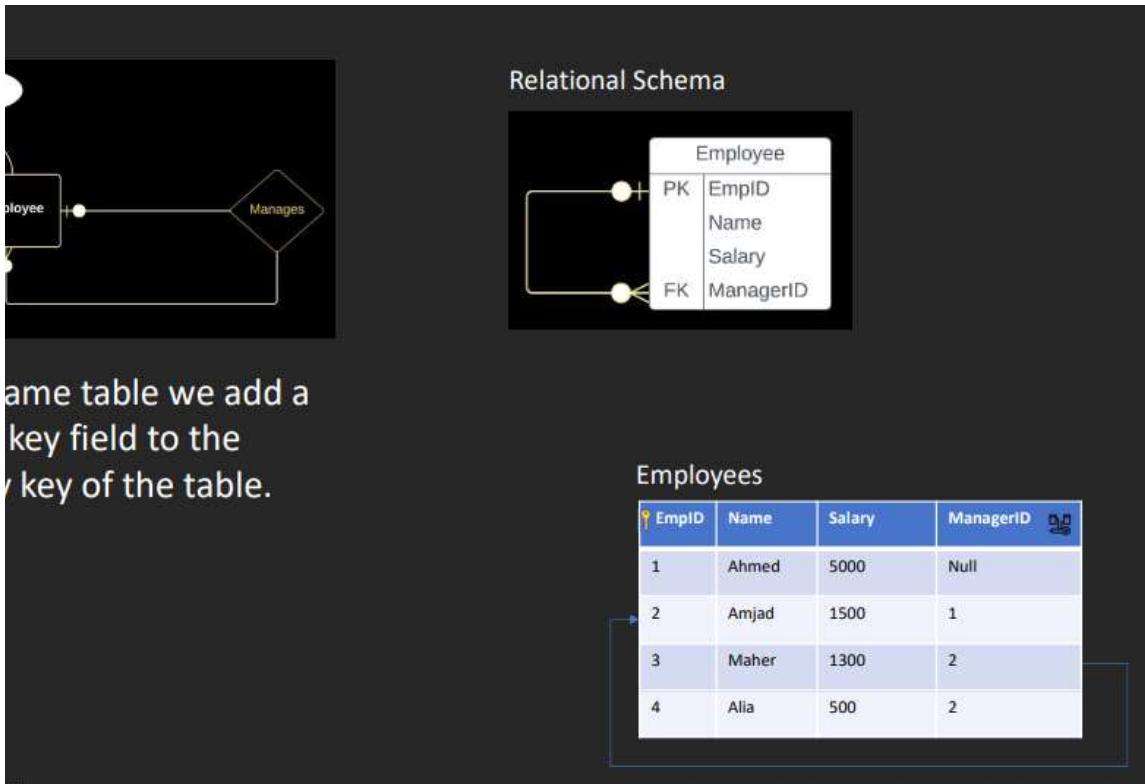


كل اللي بتعمله انك بتعمل مربع بكتب فيه اسم الجدول او ال entity وتشد سطر وبعدين تكتب ال attributes وجنب العمود اللي عايز تعمله primary key بتكتب pk

وبعدين بتزود عمود بيمثل ال foreign key وتكتب جنبه fk والعلقه نفسها بتمثلها باذلك بتوصل ال foreign key بال primary key وتحط نوع العلاقة

زي كده





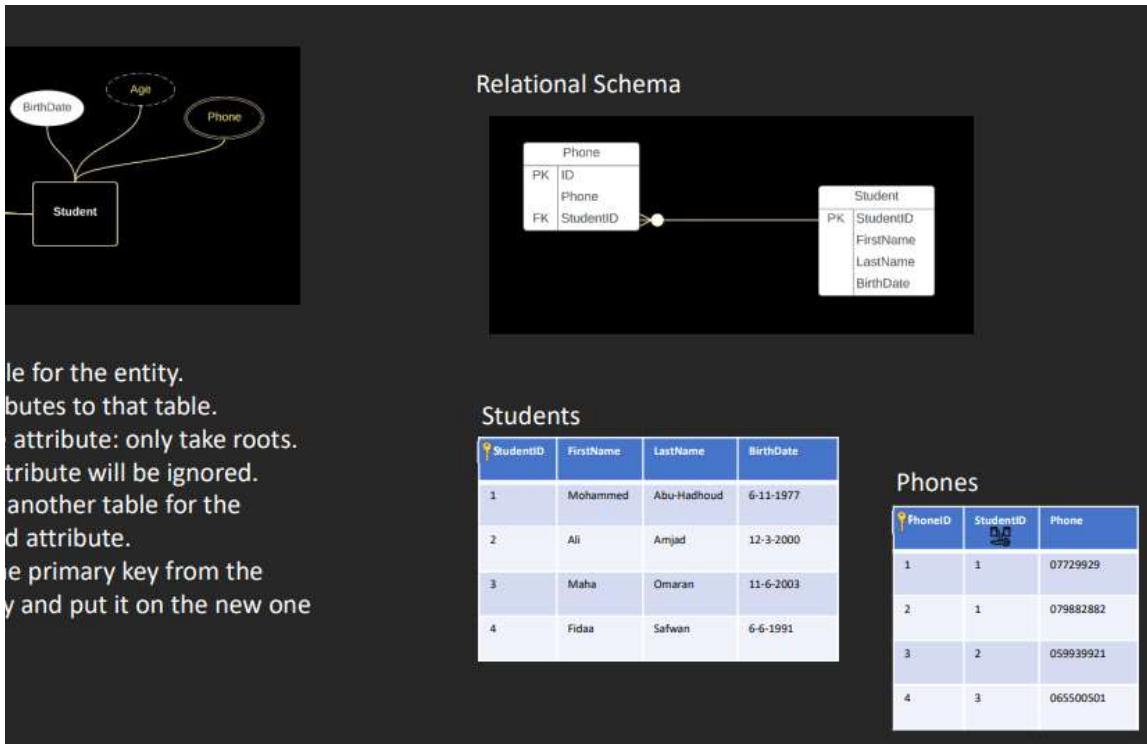
### Convert Composite-Multivalued-Derived Attributes <== Relational Schema

هنا بيقولك وانت بتعمل ال schema ولقيت في وشك composite attribute زي name ماتتع Geschäfts لأن كل اللي هتعمله هتاخذ ال attributes اللي هو واحد منها قيمة زي ال first name وال last name وتحطthem في نفس الجدول ولا كانك شايف ال name ده ولا يشغلk في حاجه

لو لقيت في وشك derived attribute ولا كانك شايفه

لو لقيت multi valued attribute هتاخذ تدليه كف تعمل منه entity صغيره يعني هتاخذه تعمله جدول لو وحده زي ال phone مثلاً لو الشخص ليه اكتر من رقم هتعمل جدول فيه id والتيليفون وال student id اللي هيكون foreign key يشاور على الطالب صاحب التيليفون ده

زي كده

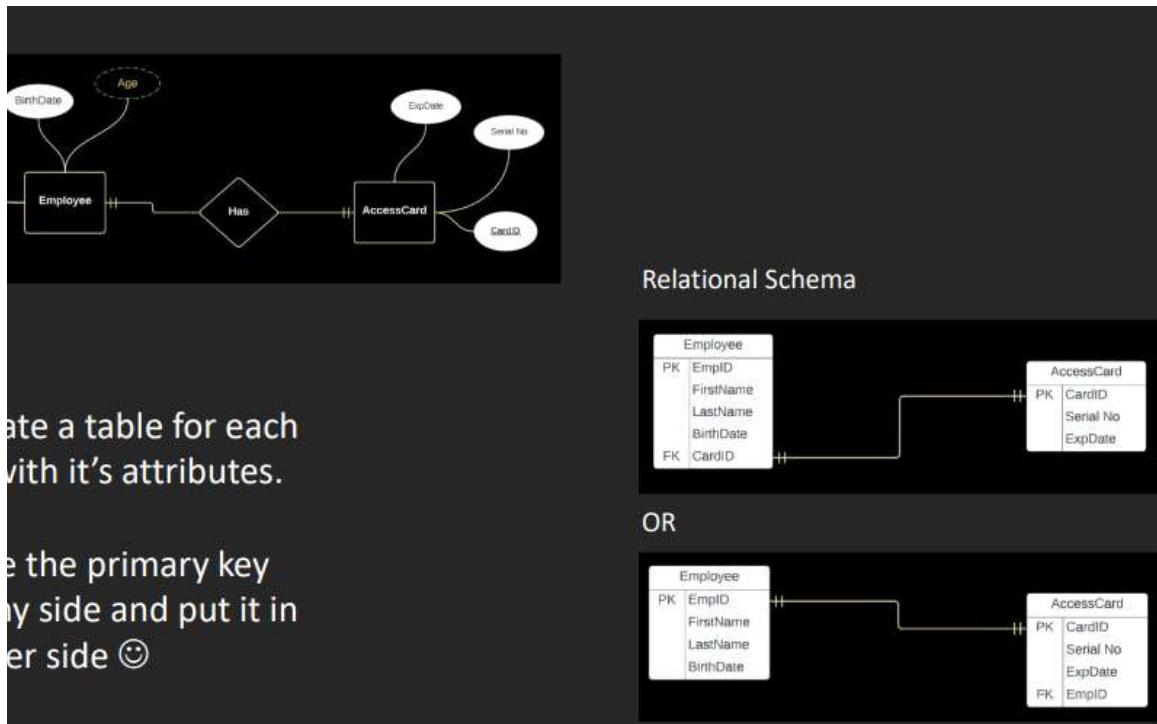
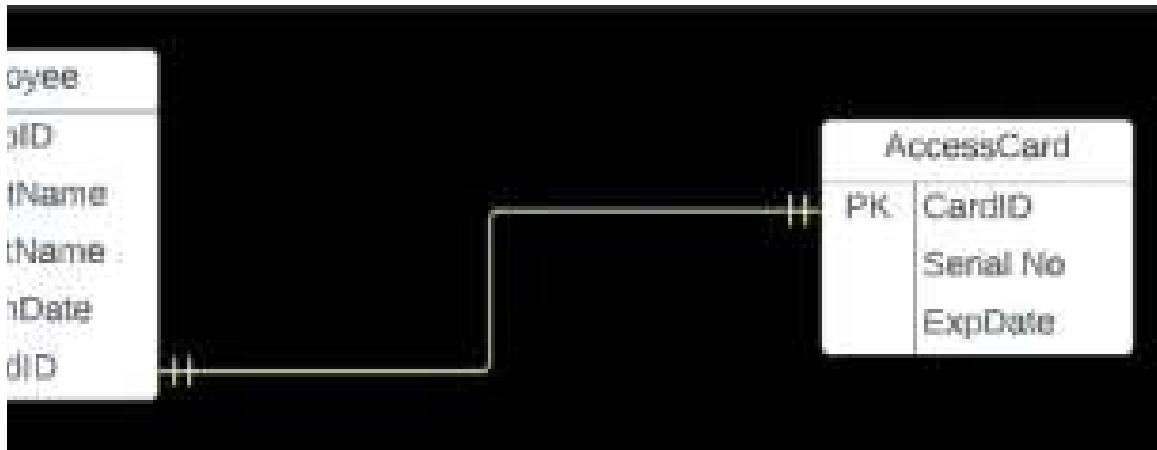


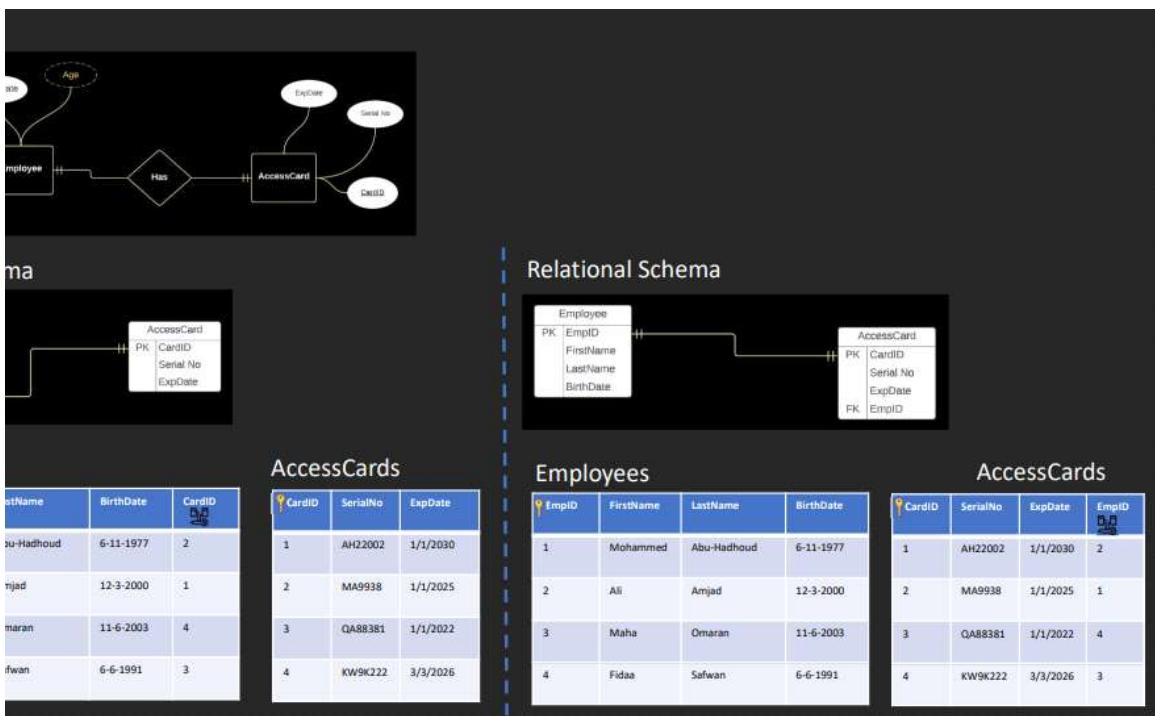
فيه ملحوظه هنا انك لما تتعامل مع ال erd او ال schema انت بتتعامل مع واحده لكن لما تتعامل مع جدول اصبح set of entities يعني ماتجيش تقولي ده جدول الطالب هقولك انت عامل جدول بحاله عشان طالب واحد؟

### Convert One-to-One <== Relational Schema

عشان تعبر عن ال one to one بتعمل الجدولين بال schema عادي بس بتيجي تزود عمود foreign key ومش هتفرق معاك تعمله في انهي جدول من الاثنين بس يفضل انك تعمل ال foreign key في الجدول اللي استخدامه اكتر في الداتا بيزعشان يكون فيه المعلومات كامله

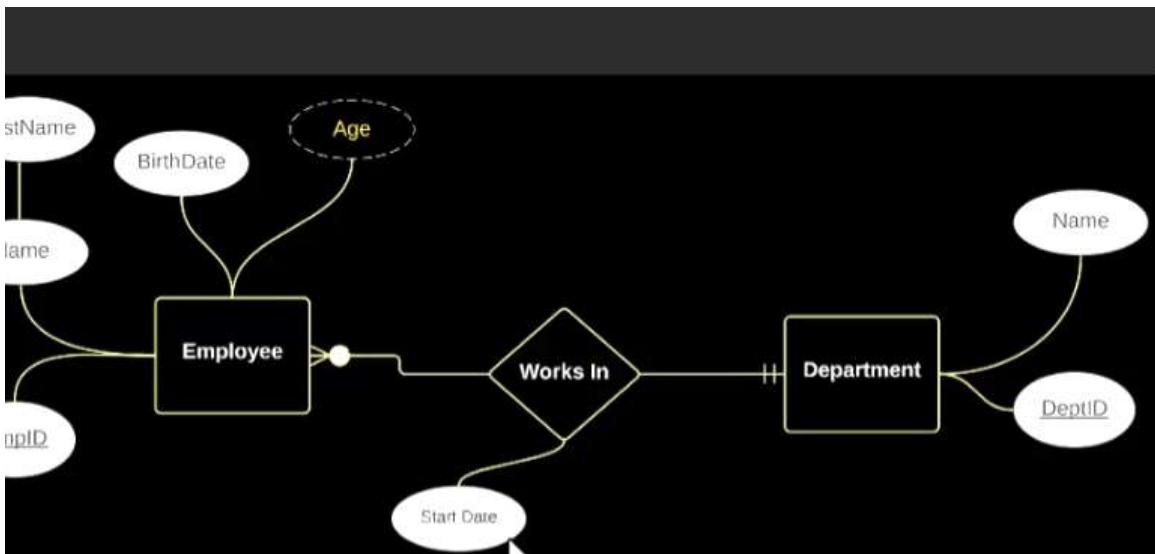
زي كده





## Convert One-to-Many/Many-to-One <= Relational Schema

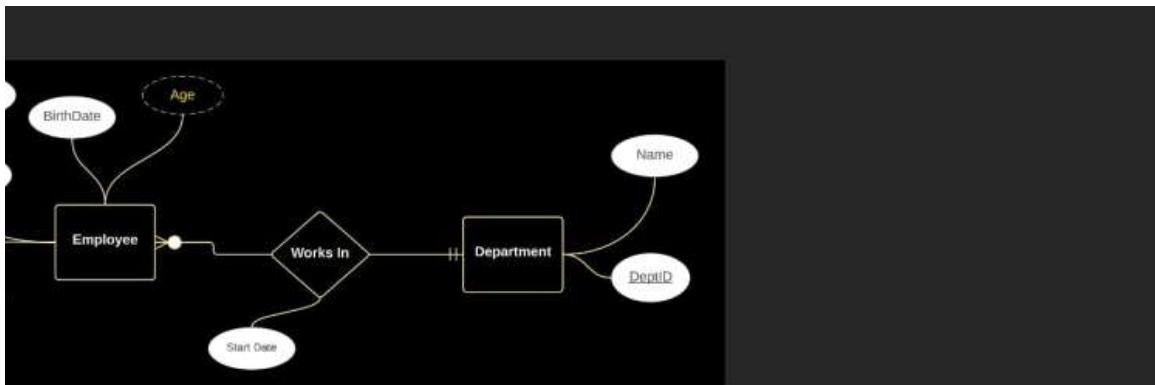
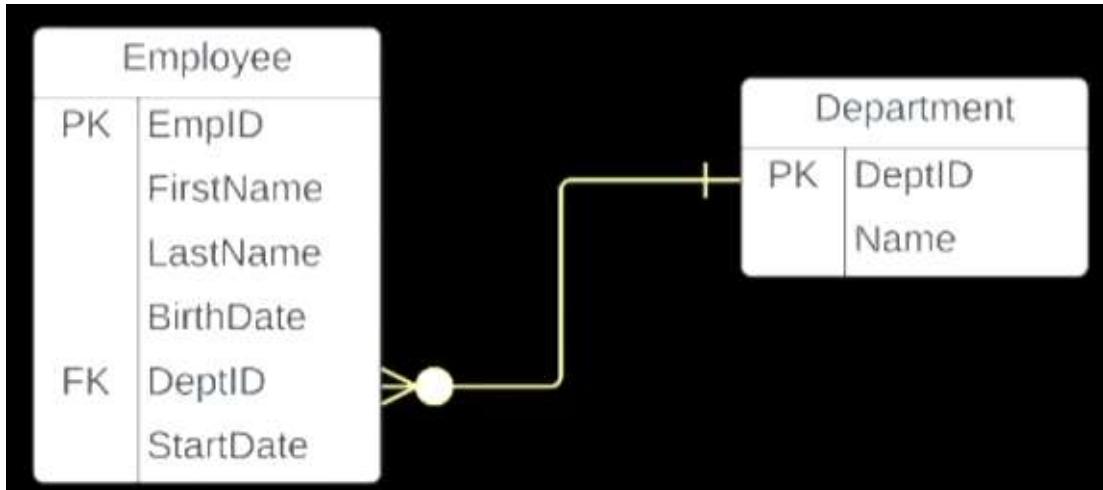
عاوزين نحول دي



لاحظ ان العلاقة هي many to one وانه لما تحصل العلاقة بين الجدولين عاوز اخزن ال start date

start date

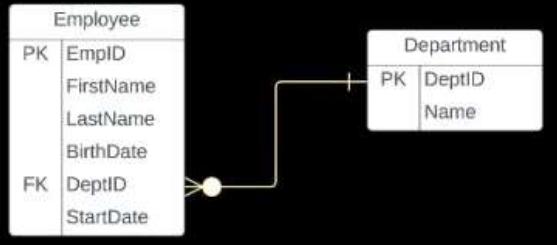
قالك عشان تعمل ال schema بتروح للجهه اللي ال one ناحيتها بتاخذ ال primary key بتعالها وترجع الناحيه الثانيه بتعمله foreign key وبتحط معاه ال المرتبط بالعلاقه اللي هوا في حالتنا ال start date attribute زي كده



Create a table for each  
with its attributes.

Take the primary key  
the “one” side and put  
foreign key in the  
“many” side

Relational Schema





### Convert Many-to-Many <= Relational Schema

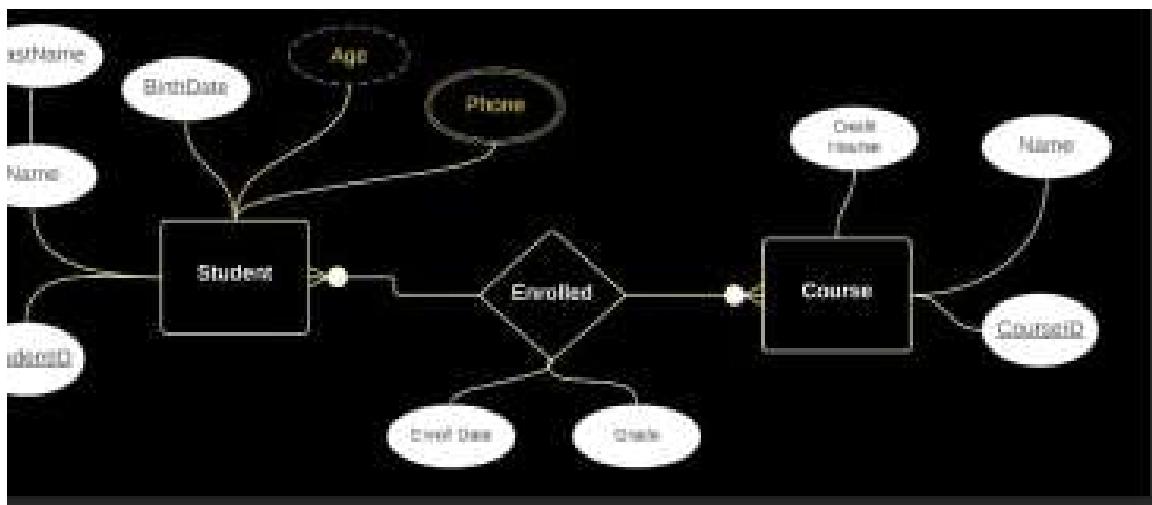
هنا بيقولك انك اول ماتلاقى علاقه many to many لازم تعمل جدول وسيط بينهم الجدول ده بتعمله id خاص بيها وبتعمل اتنين foreign keys كل واحد فيهم بيشاور على ال

بتعال كل جدول من الجدولين الثانيين وفي الجدول الوسيط ده بتحط فيه أي معلومات ناتجه عن العلاقة دي

العلاقه بين الجدولين والجدول الوسيط هيا علاقه one to many او many to one two many حسب الاتجاه بس ال many بنكون ناحية الجدول الوسيط

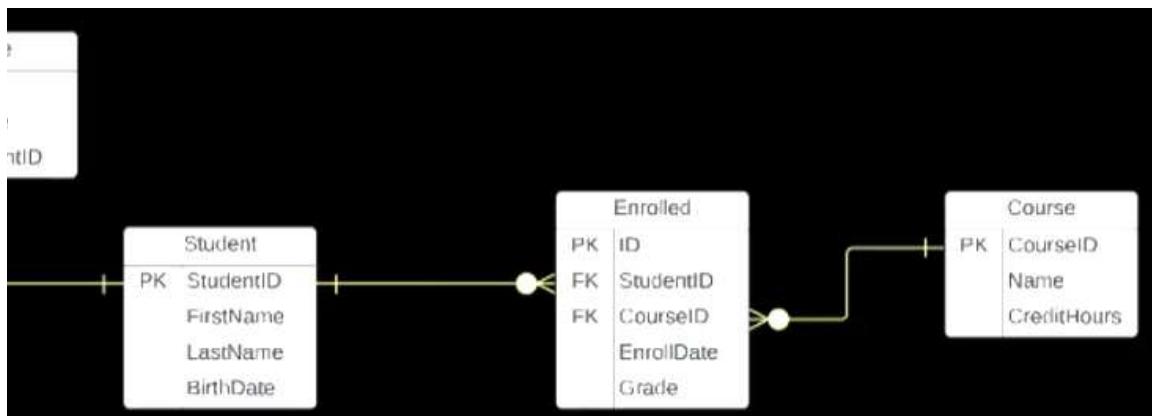
فيه ناس بتتشيل ال primary key وبيستعملوا الاتنين foreign keys مع بعض على انهم primary key هنا بيقولك ده هيختلي السيسitem ابطأ وهيقيده بعدين لانه مثلا لو فيه طالب عايزيعيد نفس الكورس مش هيقدر او العميل مش هيقدر يطلب نفس الاوردر تاني لانك في الحاله دي انت هتكرر عمود ال primary key اللي في الجدول الوسيط فعشان تقوله اعملي record تاني ال primary key record ممكن من نفس الكود ونفس الطالب هيرفض

عاوزين حول ال erd ده



العلاقه فيه فيه many to many enroll date و grade وفيه عادي اخزن many to many لوحدهم

هنا بيقولك خزن ال enroll date وال grade مع الجدول الوسيط



جدول ال phone ناتج عن ال multi valued attribute مالهوش علاقه بموضوعنا



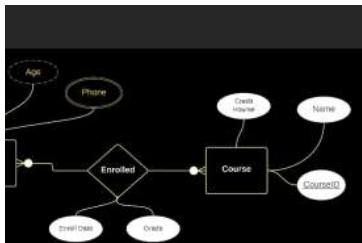
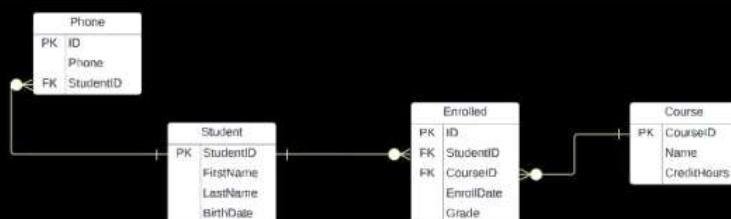
Create a table for each entity with primary key and attributes.

Create a bridge table for the many to many relationship.

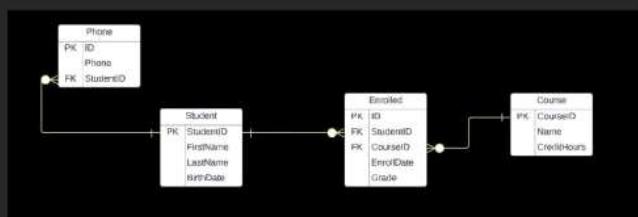
Get primary key from each tables and add them to the table and put them in the table.

Add primary key to the bridge table.

Relational Schema



Relational Schema



Students			
	StudentID	FirstName	LastName
29929			
882882			
339921	1	Mohammed	Abu-Hadoud
500501	2	Ali	Amjad
	3	Maha	Omaran
	4	Fidaa	Safwan
			6-11-1977
			12-3-2000
			11-6-2003
			6-6-1991

Enrollments

EnrollmentID	StudentID	CourseID	EnrollDate	Grade
1	1	2	1/1/2000	95
2	1	3	20/1/2005	80
3	2	2	5/5/2022	50
4	3	4	15/1/2021	45
5	3	4	25/6/2022	Null

Courses

CourseID	Name	CreditHours
1	OOP	3
2	Database	3
3	C#	3
4	Marketing	1

## Generalization and Specialization to Relational Schema

هنا بيقولك عشان تستخدم ال generalization انت بتربط بين الجدول الاب والجدول الابن بعلاقة واحد لواحد وال foreign key بتحطه في الجدول الابن لأنك

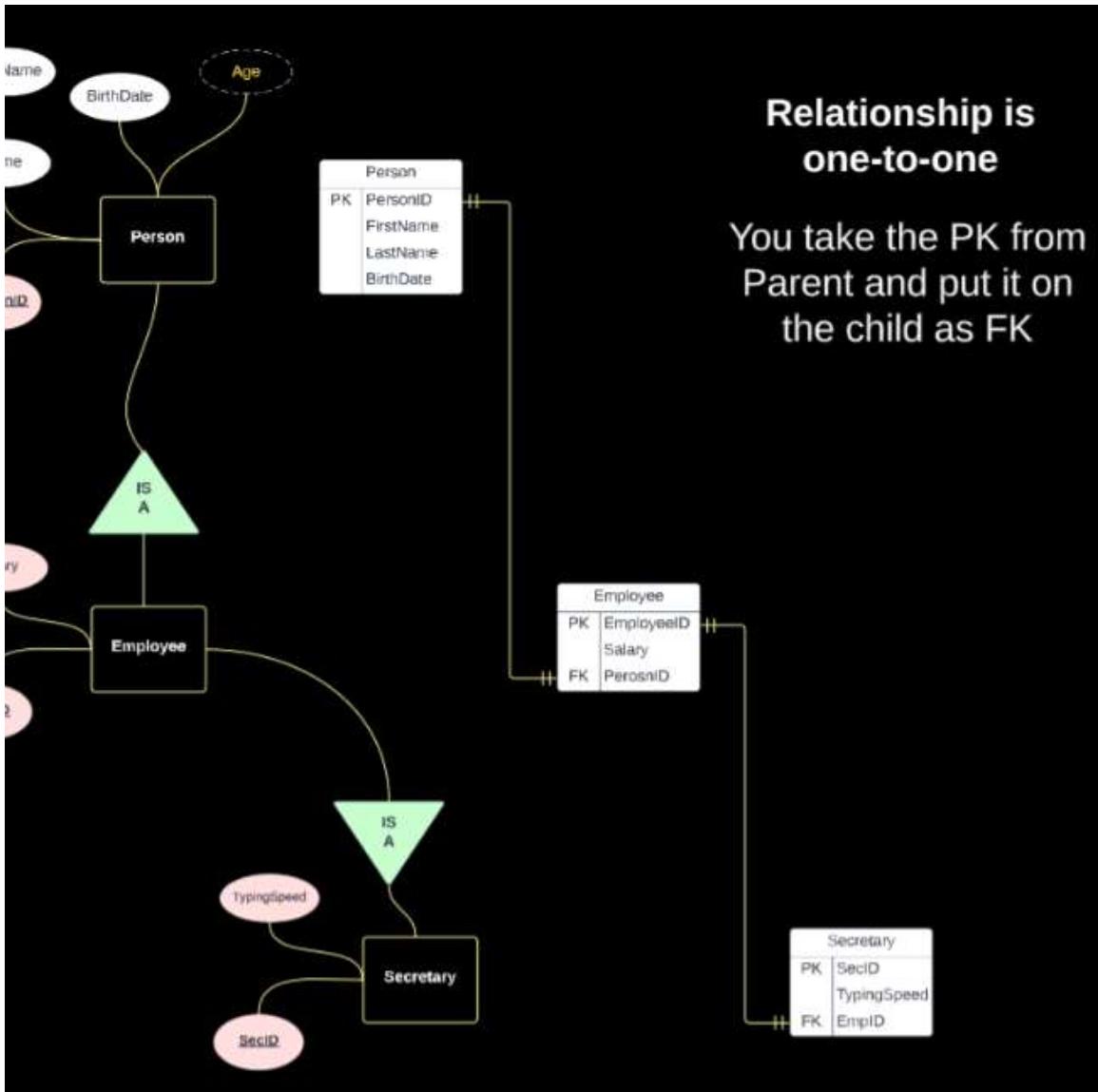
لو حطيته في جدول الاب هتعمل بيه ايه؟

## **Generalization and Specialization to Relational Schema**

### **Generalization of Specialization into Relational Schema:**

Remember that the relationship is always one-to-One.

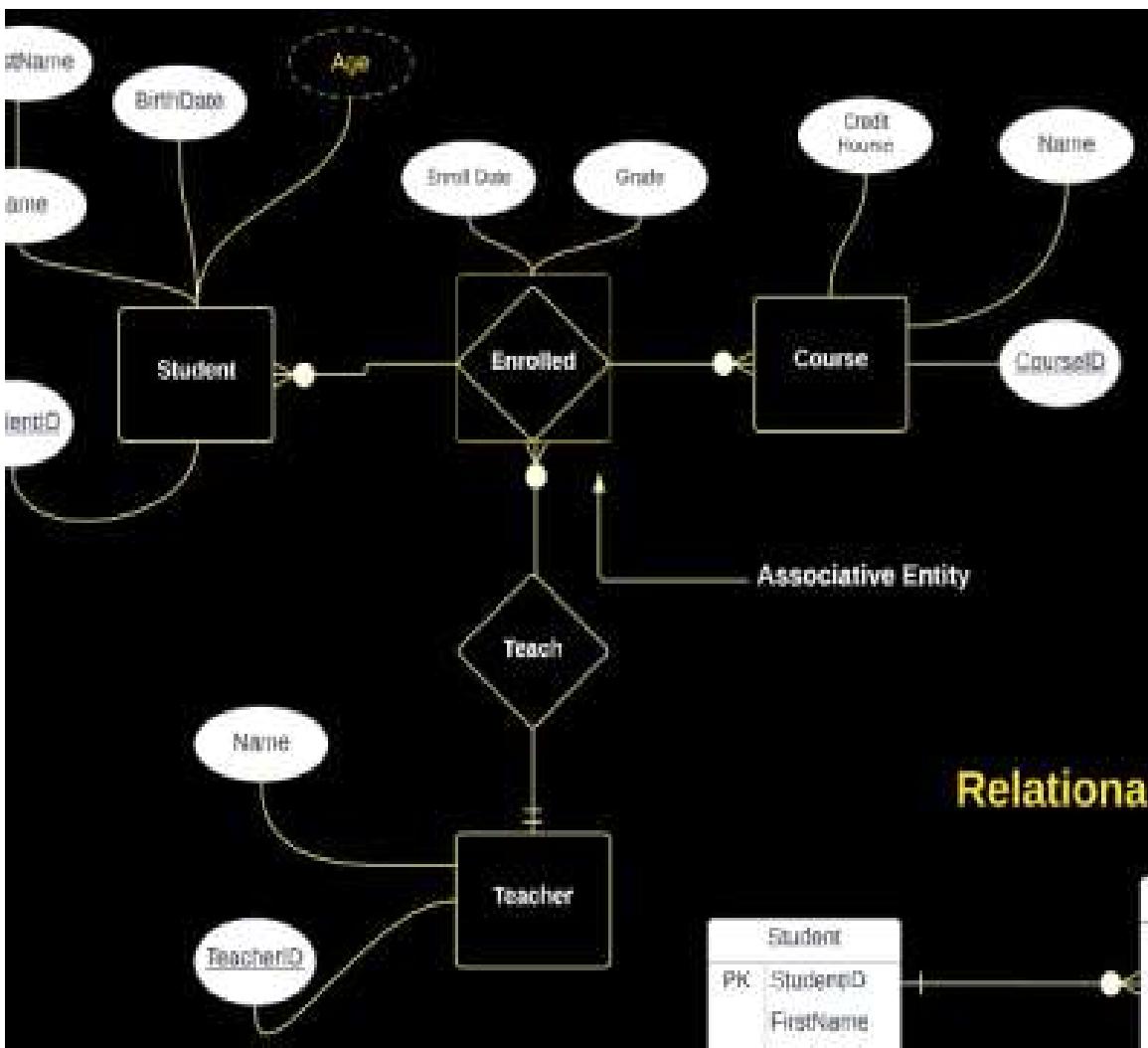
take the primary key from the parent entity and put it as foreign key in the child entity.



### Convert Associative Entity to Relational Schema

ال entity associative هي ال entity اللي مبنيه على علاقه تانيه والعلاقه التانيه دي بتكون many to many

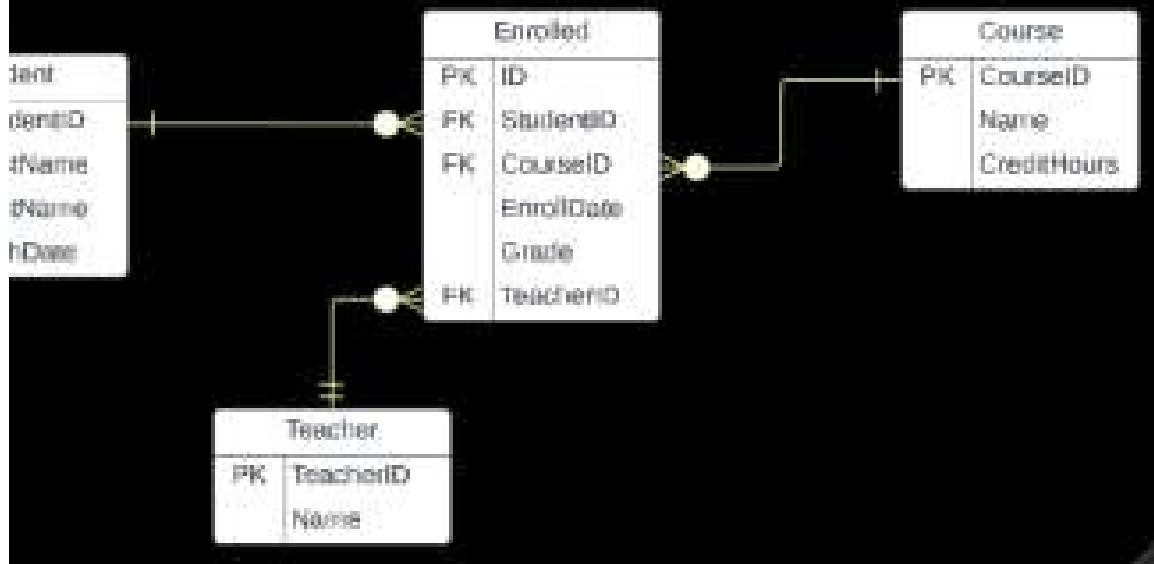
فعشان امثلها كل اللي بعمله اني بعمل جدول وبربطه بعلاقه one to many بالجدول الوسيط



Relational

Student	
PK	StudentID
	FirstName

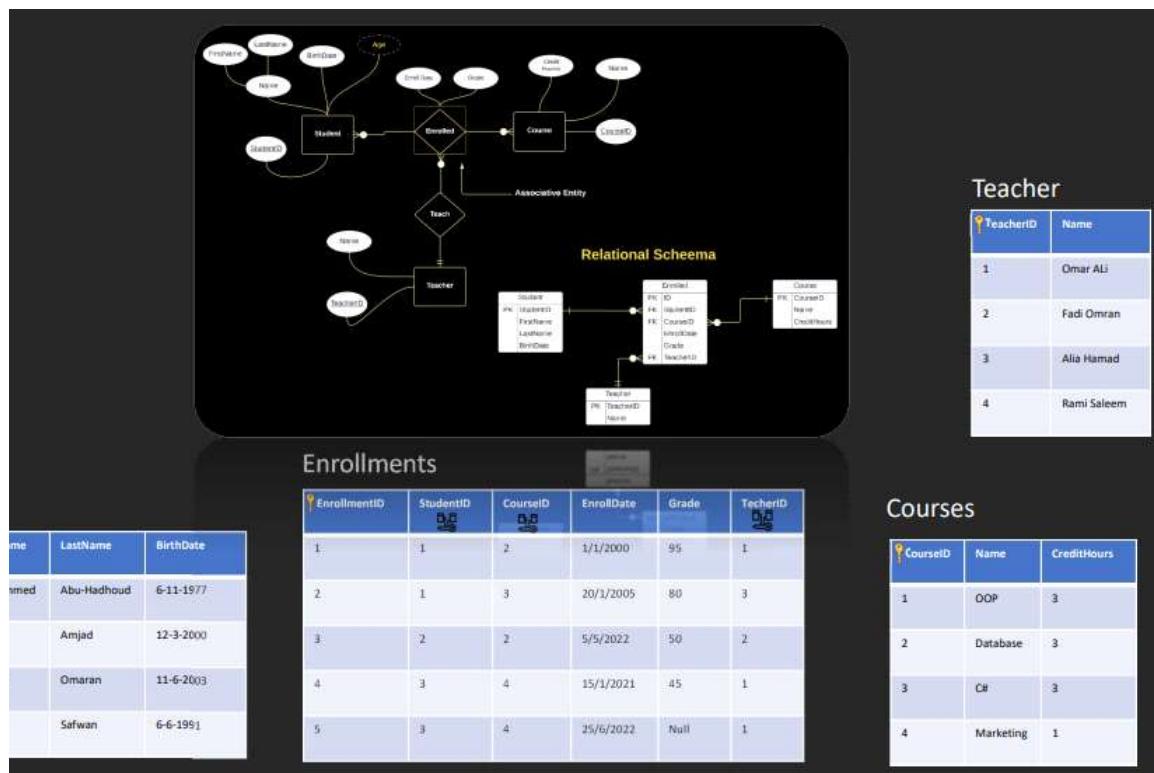
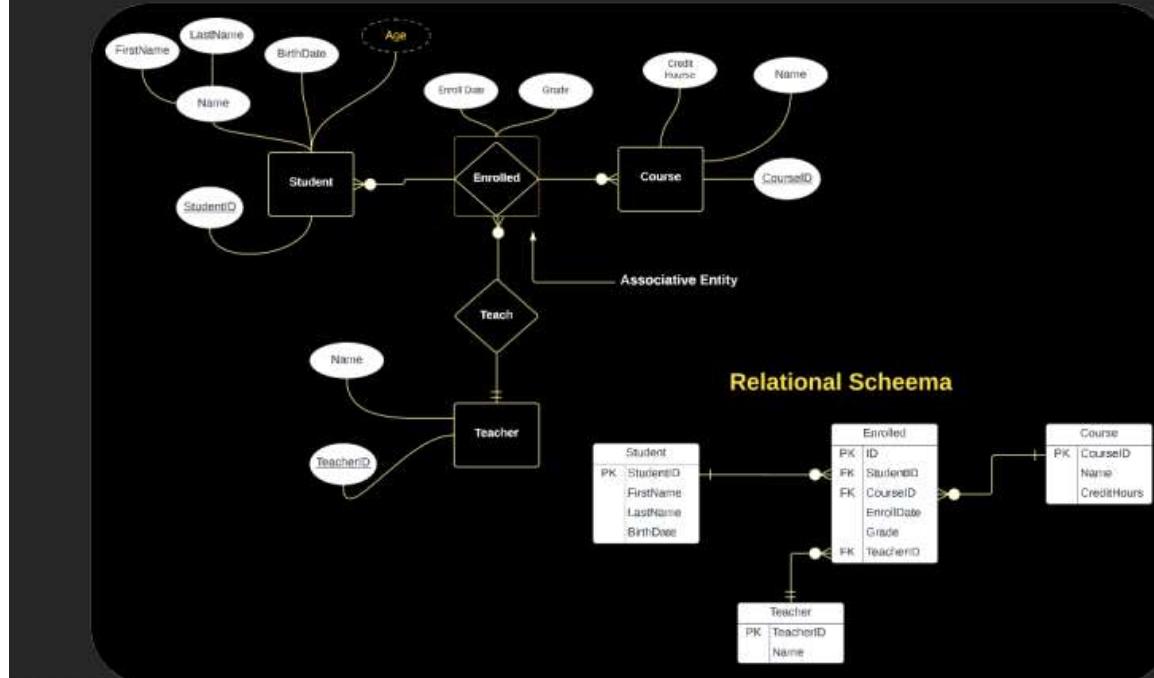
## Relational Schema



a table for each entity with its attributes.

bridge table for the many to many relationship.

primary key from each tables in the relation and put them in the bridge table. Also Take primary key from the associated relation and also put it in the bridge table .



## **How to create Relational Schema on ERDPlus.com?**

هنا عشان تعمل ال relational shema في الموقع بتختارها من هنا



### **Create New Diagram**

Create a new diagram.

Name

Type

ER Diagram

Relational Schema

Star Schema

[CANCEL \(ESC\)](#) [CREATE](#)

والباقي سهل

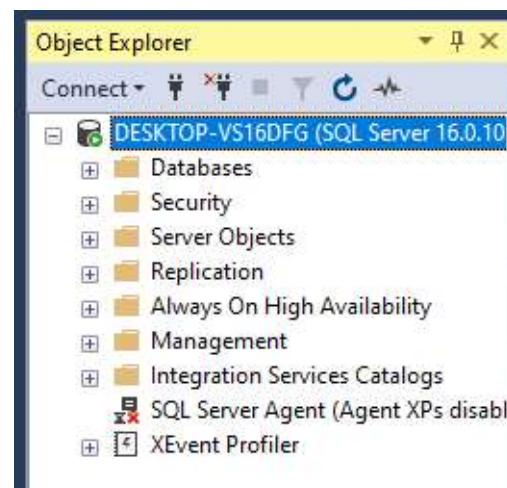
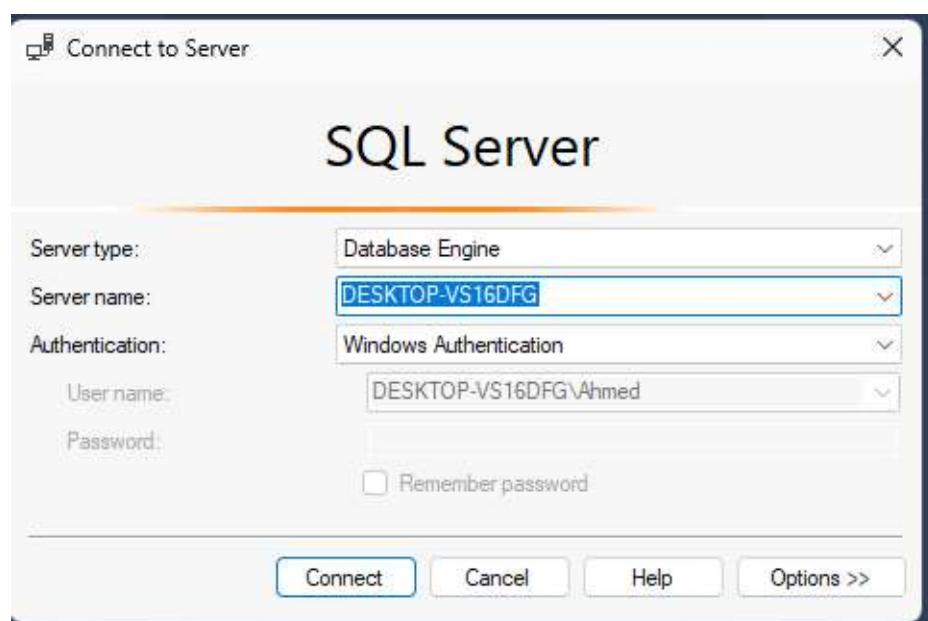
## **(revision) What is SQL?**

<https://programmingadvices.com/courses/database-level-1-sql-concepts-and-practice/lectures/46511156>

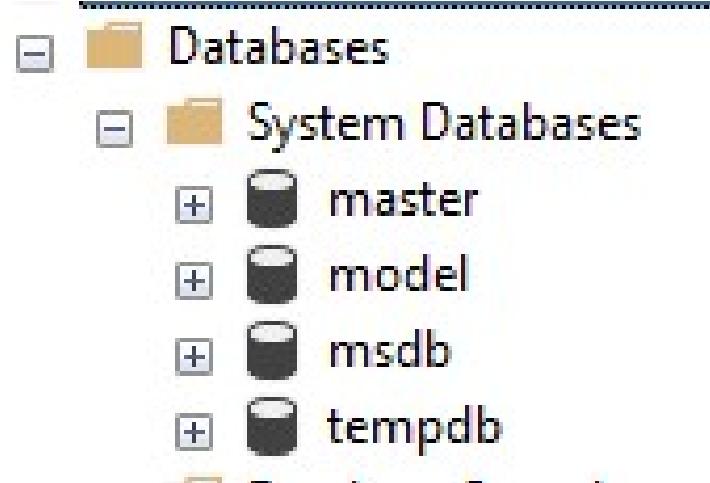
## **SQL - Data Definition Language - DDL**

### **Create Database**

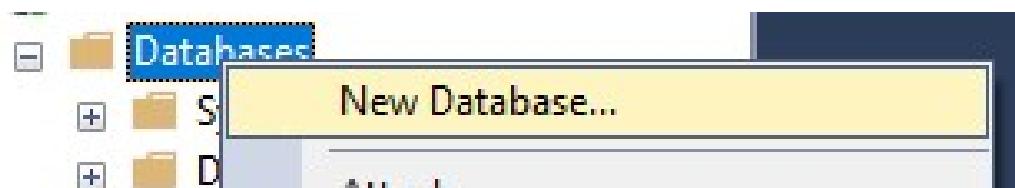
عشان نتواصل مع الداتا بيز لازم الاتصال يتم عن طريق  
sql management studio

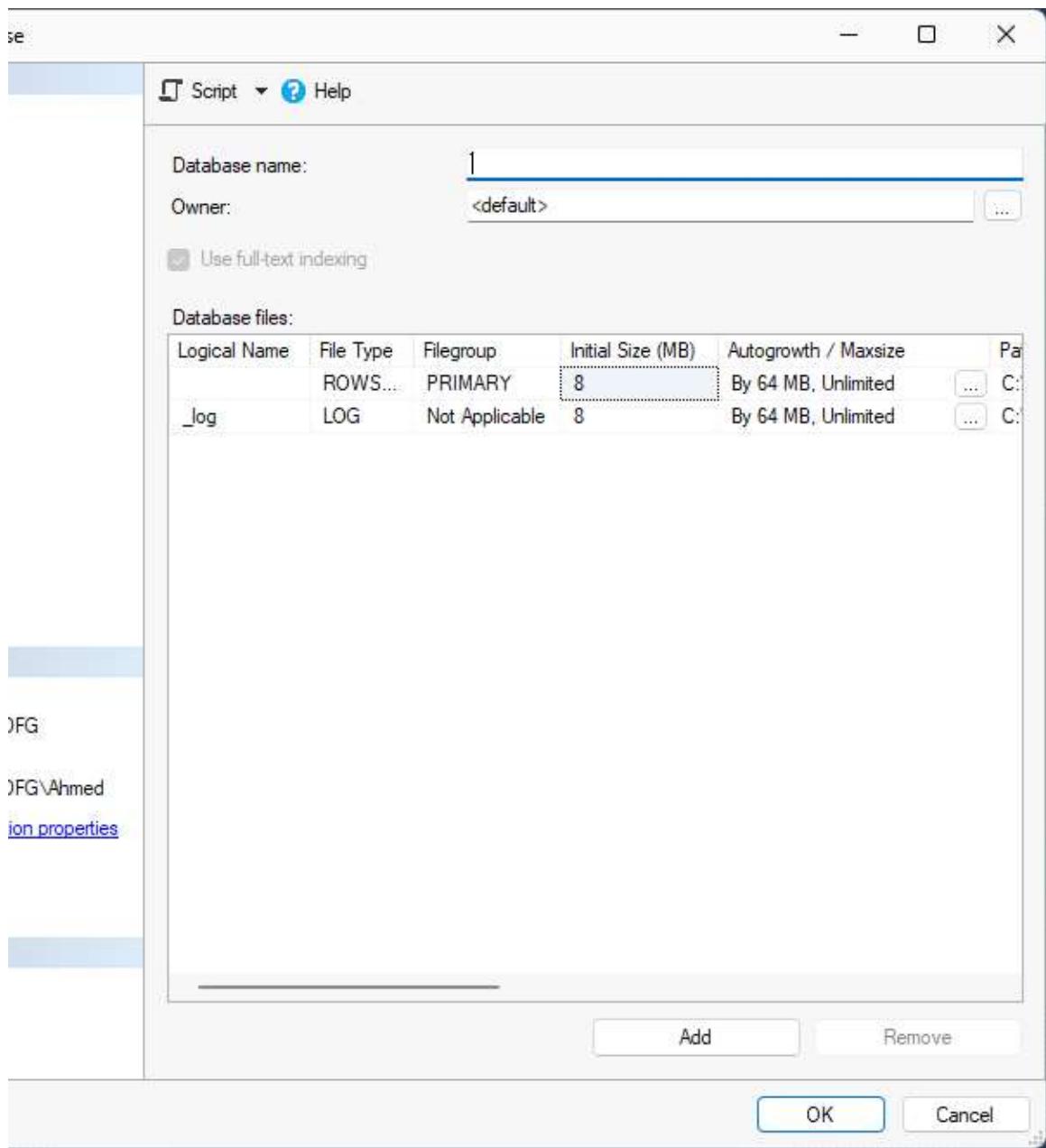


وزي ماقولنا اوعي تلعب في دول



وهنا بيقولك انه فيه طریقتین عشان تعمل داتابیز طریقه بالماوس و طریقة بالکود





Database name:

DB1

وبعد دوس OK هتظهر لك في الجانب



طيب دي الطريقة الاولى

الطريقة الثانية من خلال الكود وهنا بيقولك انه أي حاجه في ال SQL SERVER بتقدر تعملها بالماوس وبالكود عادي

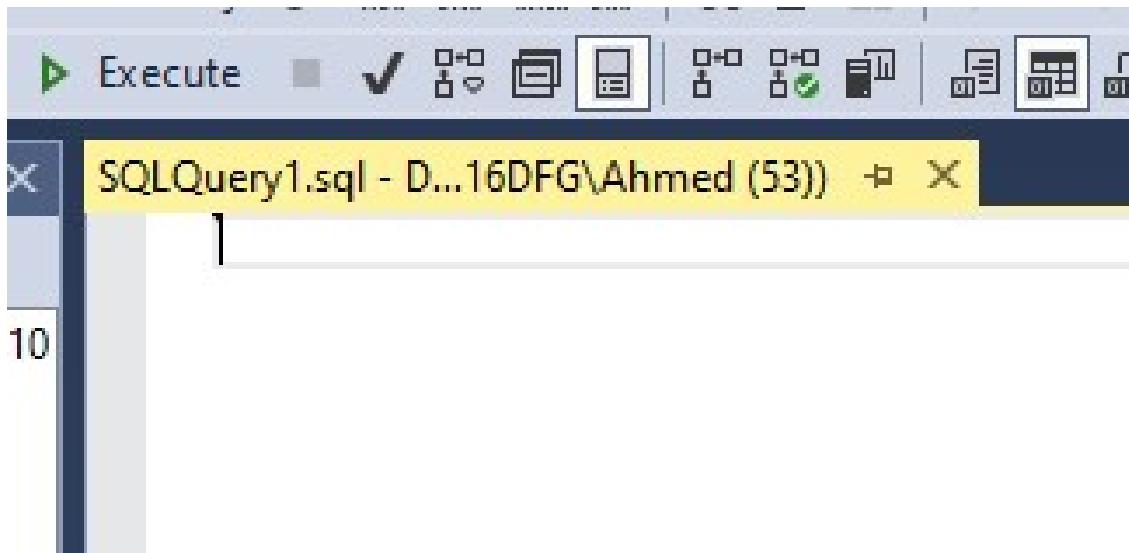
عشان تكتب أي كود بتدوس على NEW query من القايمه اللي فوق



هنا بتنفذ الكود بتاعك

هنا بتكتب الكود او ال query

هنا بتختبر الكود بتاعك لو فيه خطأ في ال syntax



عاوزين بقى نعمل داتا بيز جديده من خلال الكود

قالك كل اللي بتعمله انك تكتب create وبعدين الحاجه اللي عايز تعملها في حالتنا هنا هنكتب database وبعدين بتعين اسم للداتا بي

```
create database DB2
```

وبعدين لو عايز تعمل check عالكود بتدوس هنا



والنتيجه بتظهر تحت

```
100 % < 
Results
Commands completed successfully.

Completion time: 2023-08-24T09:34:52.6590989+03:00
```

لو عايز تنفذ الكود من هنا

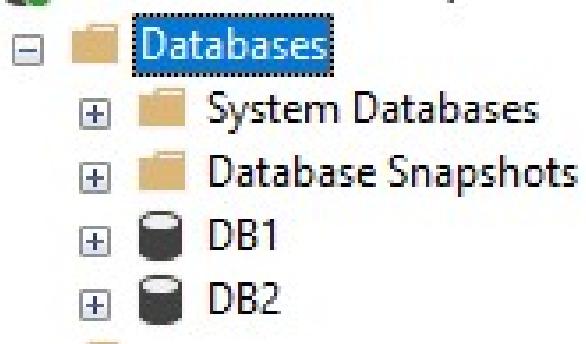
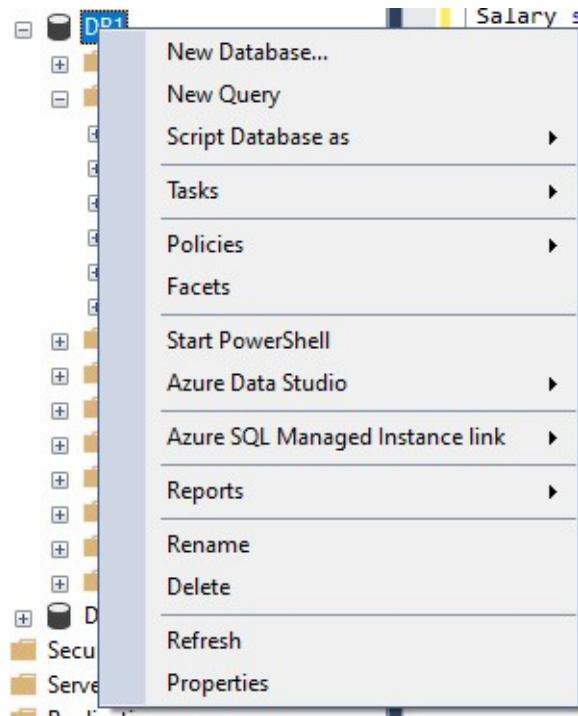


والنتيجه برضه بتظهر تحت

```
100 % < 
Messages
Commands completed successfully.

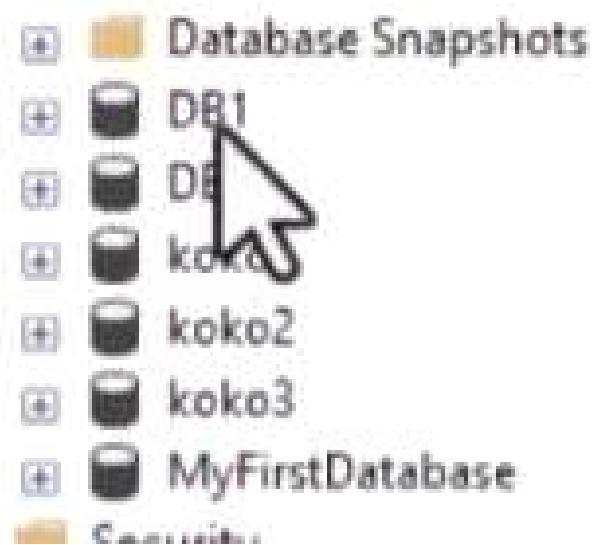
Completion time: 2023-08-24T09:36:34.2208082+03:00
```

وبعدين بتيجي تعمل refresh للقاييمه عشان تظهرلك الداتا بي



وهنا هوا بيعمل اكتر من واحده في نفس الوقت

```
create database koko;  
create database koko2;  
create database koko3;
```



# SQL CREATE DATABASE Statement

Before we can work with database tables, we must create a database first.

The `CREATE DATABASE` statement is used to create database tables. For

```
CREATE DATABASE Koko;
```

Here, the SQL command creates a database named Koko.

## Create Database IF NOT EXISTS

هنا بيقولك لو حاولت تعمل داتا بيز جديد بنفس الاسم بتاع داتا بيز موجوده عندك  
قبل كده هيضرب منك

```
CREATE DATABASE DB2
Msg 1801, Level 16, State 3, Line 1
Database 'DB2' already exists. Choose a different database name.
Completion time: 2023-08-24T09:43:48.9370007+03:00
```

عشان تعمل comment هنا بتكتب - علامتين طرح

```
--create database DB2
```

طيب احنا دلوقتي عاوزين نتجنب الخطأ ده ونقوله لو مالقيتش داتا بيز بالاسم ده  
اعمل واحده جديده

طيب جملة ال if statement في ال sql server بتكون من ايه؟  
قالك بتكتب If والشرط وبعدها بتكتب begin وبعدين الامر اللي عاوز تنفذه لو  
الشرط اتحقق ولما تخلص تكتب end  
دي كده اول حاجة

تاني حاجة وهيا NOT ودي زيها زى علامه ال ! في ال C++ او ال C# اللي بتغير ال  
شرط بتقلبه false والعكس true

ثالث حاجة وهيا ال function اللي اسمها exists ودي بتاخذ جمله او  
وترجع Boolean بتقولك ال query موجوده ولا لا او تم تنفيذ الكود ده ولا لا

تعالي بقى نركب البازل  
اول حاجة هحط ال if statement

```
IF  
BEGIN  
  
END;
```

بعدين انا عايز أقوله لو مش موجوده

```
IF NOT EXISTS()  
BEGIN  
  
END;
```

عشان أقوله يختار الداتا بيز اللي اسمها DB1 مثلا بكتب الجمله دي واللي معناها اختار كل حاجه من ال DATABASES اللي موجوده في السيستم بشرط انه الداتا بيز يكون اسمها كذا

```
IF NOT EXISTS)SELECT * FROM sys.databases where name='DB1'
BEGIN
END;
```

كده ناقص اكتب الكود اللي عايزة انفذه

```
IF NOT EXISTS)SELECT * FROM sys.databases where name='DB1'
BEGIN
create database DB2;
END;
```

فيه حاجه هنا عاوز يقولهالك وهيا انك لو حددت كود معين وجيته تشغل الأوامر اللي هيتنفذ هو الكود المظلل بس

```
IF NOT EXISTS(SELECT * FROM sys.databases where name='DB1')
BEGIN
create database DB2;
END;
```

messages						
base_id	source_database_id	owner_sid	create_date	compatibility_level	col	Ar
NULL	0x0105000000000005150000009D80740CDC06877702C6E3...		2023-08-24 09:22:51.143	160		

## TABASE IF NOT EXISTS

a database with the same name, SQL will throw an error while creating a database.

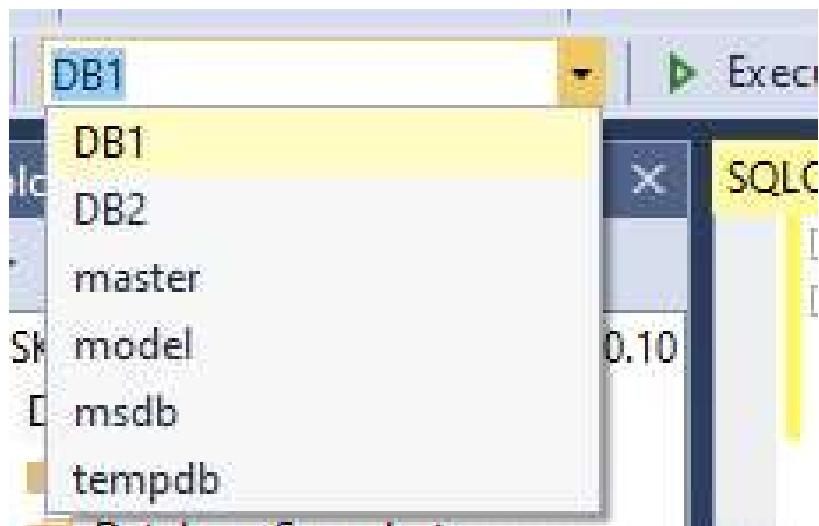
, we can use the [the following code](#) to create a database only if there is no existing database with the same name. For

```
IF NOT EXISTS (SELECT * FROM sys.databases WHERE name = 'koko')  
CREATE DATABASE koko;
```

ates a database named koko only if there is no existing database with the same name.

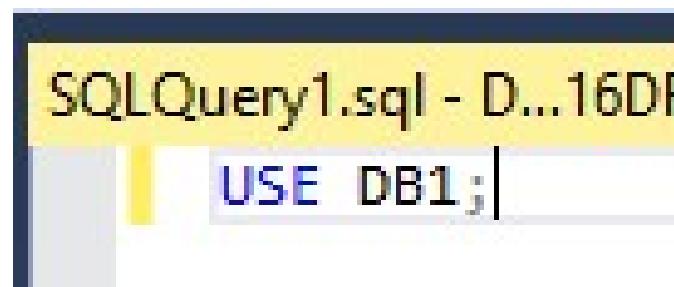
## Switch Database

هنا بيقولك انك ممكن تكون بتكتب query على داتا بيز غير اللي انت عايزة ودي  
مسيبه فلازم تتأكد من الداتا بيز اللي انت عايزة تنفذ عليها الأوامر بتاعتاك  
عشان نعرف انهي داتا بيز اللي بيتنفذ عليها الأوامر بنلاقي اسمها مكتوب هنا وبنقدر  
نغيره لو عايزين



طيب ده ساعات وبيحصل اننا ساعات بيكون علي الله حكايتنا وبننسى نبص عليها او  
نغيرها

فممك نحدد الداتا بيز عن طريق الامر `use` وبعدة نكتب اسم الداتا بيز



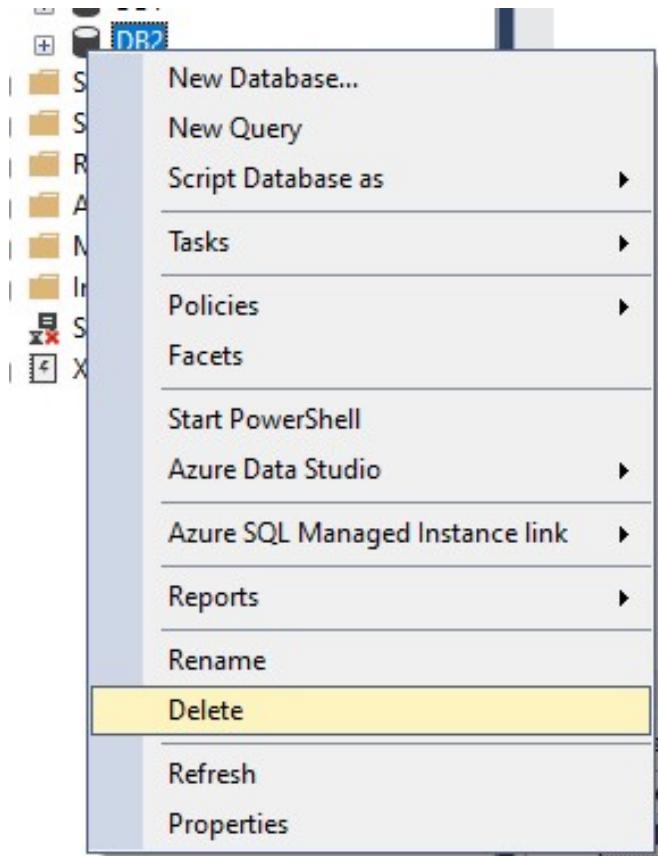
## bases

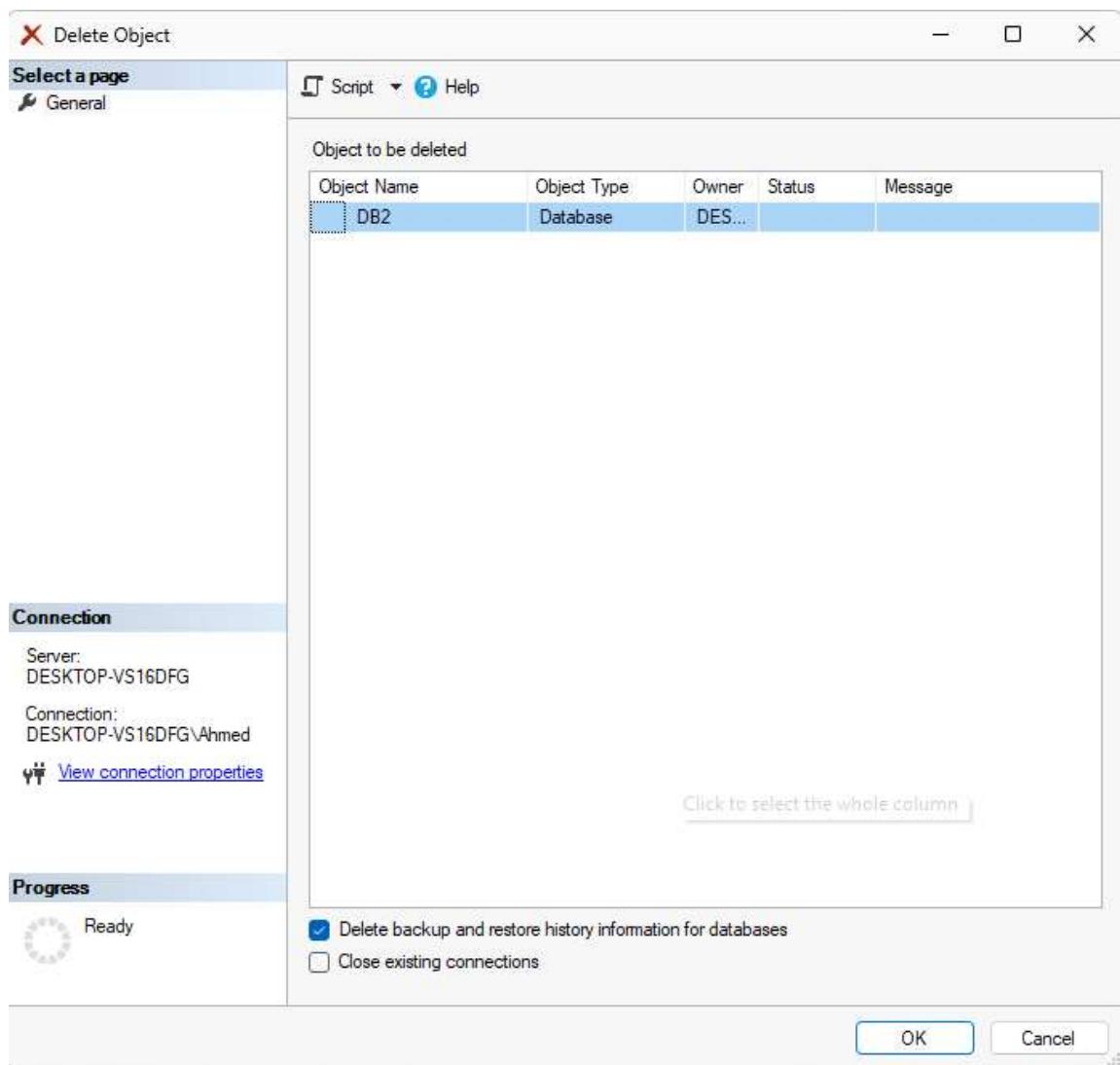
across multiple databases from time to time. To switch between available databases, we can run the following

the koko database and all SQL operations will be performed inside this database.

## Drop Database

عاوزين نلغي او نحذف داتابيز معينه لو بالماوس فالطريقه اهي





لو عايزين نعملها بالكود بنكتب drop وبعدها ايه نوع الحاجه اللي عاوزين نحذفها  
وفي حالتنا ه تكون database وبعددين اسم الداتا بيز

**drop database DB1;**

## OP DATABASE Statement

**DATABASE** is used to delete the database in our Database Management System. For example,

```
USE koko;
```

Command will delete a database named koko.

You have **admin** or **DROP** permission to run this command.

Delete a database, all tables and records within a database are also deleted.

### **Drop Database IF EXISTS**

مش محتاجه شرح لانه نفس اللي عملناه قبل كده عاوزين نقوله لو موجوده احذفها  
نفس اللي عملناه بس بدل CREATE هنكتب DROP وهنشيل كلمة NOT

```
IF EXISTS)SELECT * FROM sys.databases WHERE NAME='DB1'
BEGIN
DROP database DB1;
END ;
```

### **BASE IF EXISTS**

base with the same name, SQL will throw an error while dropping a database.

, we can use the **the following code** to drop a database only if there is existing database with the same name. For

```
LECT * FROM sys.databases WHERE name = 'koko')

BASE koko;
```

ps a database named koko only if there is an existing database with the same name.

### **Create Table**

قبل ما ندخل نعمل جدول جديد فيه حاجه لازم نعرفها و هناخد نبذه عنها دلوقتي  
ال int وده عادي مش هنختلف عليه وعارفينه

ال char() وده بيعبّر عن string وبياخد رقم الرقم ده هو عباره عن عدد الاحرف اللي  
عاوز اليوزر ميزودش عنها لو قولته 3 احرف هيدخل لحد 3 بس ولو عملتها 50  
حرف واليوزر دخل حرف واحد بس هيبدل ال 49 حرف بمسافات فهيكون في  
مساحه مهدره

المشكله الثانيه اللي فيه انه مش بيحفظ غير حروف انجلزيه بس ماتقدرش تكتب  
جواه باي لغه تانية

ال nchar() وده زي اللي فات بس بيقبل أي لغه تانية عادي  
ال varchar() وده زي ال char() بس لو جيت قولته انه عدد الحروف المسموح  
بيها 50 واليوزر دخل حرف واحد بس بيخزن حرف واحد بس من غير مسافات زياده  
وده احسن

ال nvarchar() وده زيه زي ال varchar() بس بيقبل باي لغه غير الإنجليزي وهو  
المفضل في الاستخدام

ال smallmoney هنتكلم عليها بعدين

طيب تعالى بقى نعمل جدول جديد بالماوس  
هنفتح الداتا بيز اللي عايزين نحط فيها الجدول



وبعدين كليك يمين علي الملف اللي اسمه tables

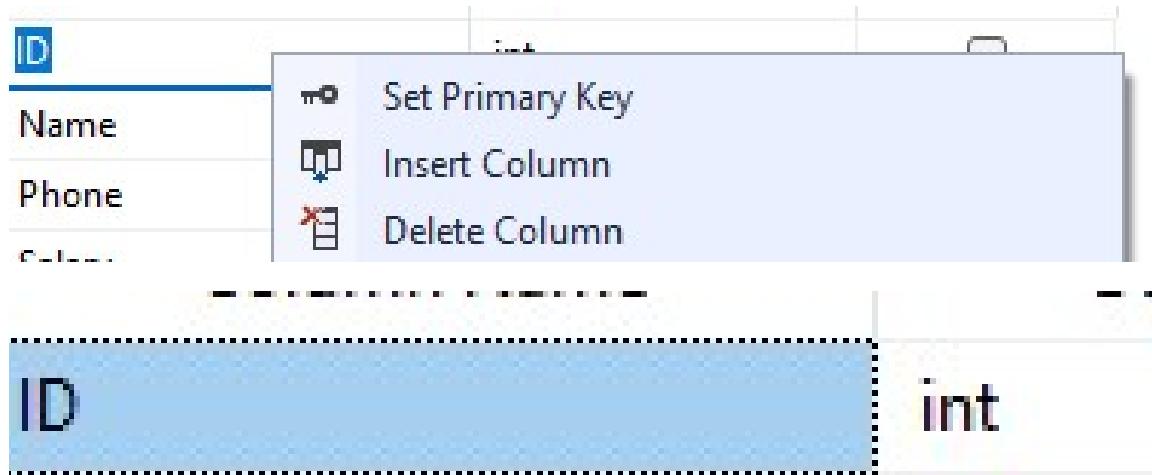
The screenshot shows the SQL Server Management Studio interface. At the top, there's a tree view of the database structure under 'DB1': Database Diagrams, Tables, Views, and External Resources. Below this, a context menu for 'Tables' is open, with 'New' selected and 'Table...' highlighted. A new table window titled '-VS16DFG.DB1 - dbo.Table\_1' is open, showing a single column definition:

Column Name	Data Type	Allow Nulls
		<input type="checkbox"/>

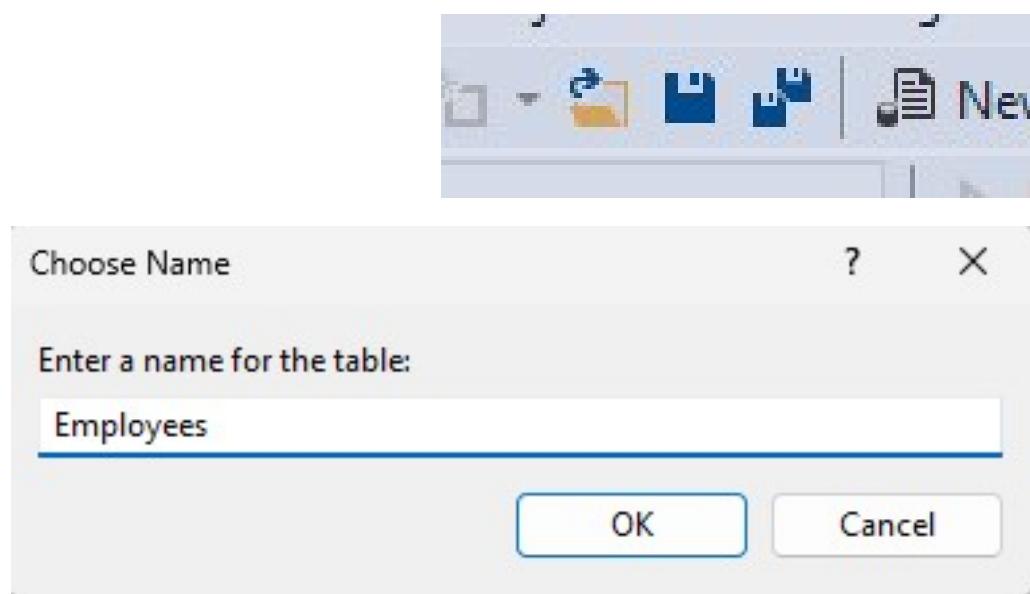
هذا بتكتب اسم العمود | هنا ال data type | عايز العمود يقبل ال  
null ولا يعني عايز  
يووزر يدخل في الداتا  
جاري ولا لا

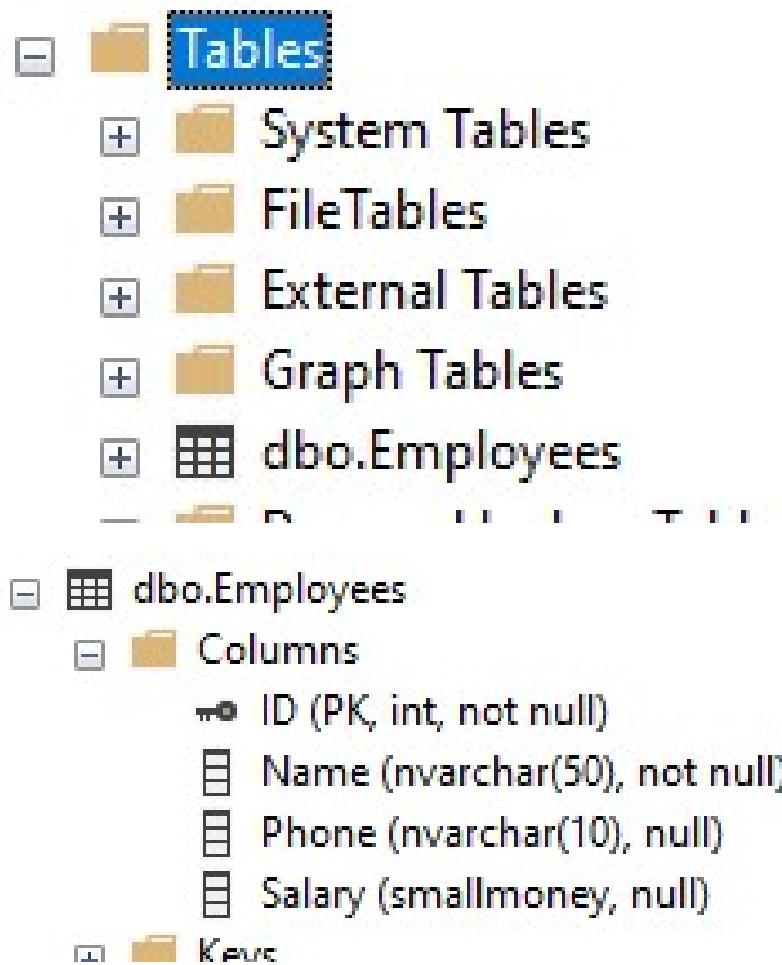
Column Name	Data Type	Allow Nulls
	int	<input type="checkbox"/>
name	nvarchar(50)	<input type="checkbox"/>
lname	nvarchar(10)	<input checked="" type="checkbox"/>
money	smallmoney	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

طيب دلوقتي عايزين نخلي ال id يكون primary key

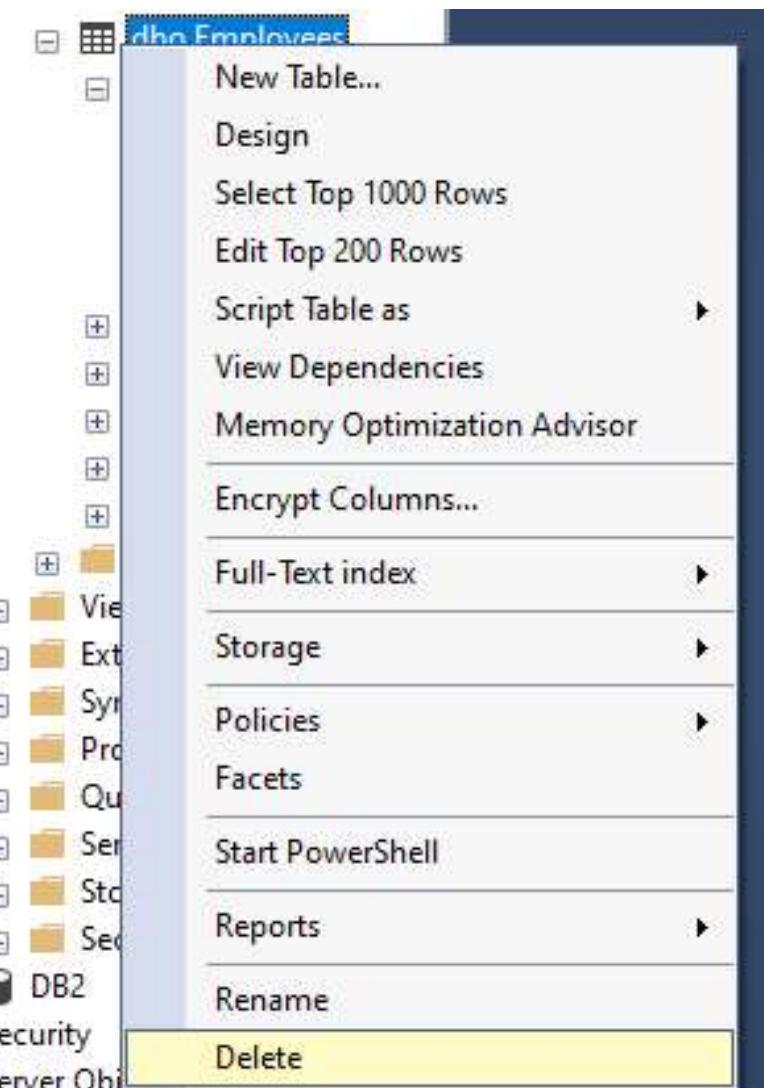


عاوزين نحفظ الجدول





دلوقي هنحذف الجدول ده عشان نروح نعمله بالكود



تعالي بقى نعمله بالكود

خلينا فاكرين انه اولمر ال DDL خاصه بالعمليات علي الداتابيز نفسها  
واحنا لحد دلوقتي عرفنا امرين واحد اسمه create وواحد اسمه delete

طيب عشان نعمل جدول جديد هنسخدم الامر create وبعدين نسأل احنا عاوزين  
نعمل ايه ؟

عاوزين نعمل جدول نقوم نكتب table وبعديه نحط اسم للجدول ونفتح قوسين

```
use DB1;
```

```
Create Table Employees();
```

طيب عاوزين نعمل الاعمدہ :-

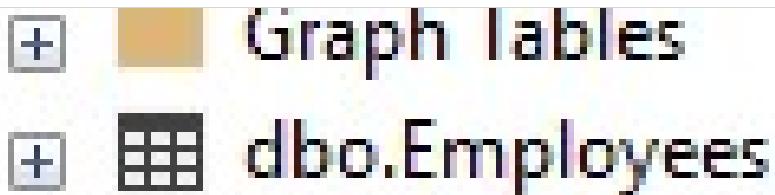
الاعمدہ هتكون جوه القوسین وبين كل عمود والثاني فاصله

هنعرف العمود ازاي قالك هتكتب اسم العمود وبعديه ال data type وبعدين null او not null

ولو فيه عمود عاوز تعرفه علي انه primary key هتكتب في الآخر كلمه key وبين قوسين اسم العمود

```
use DB1;
```

```
Create Table Employees(
ID int Not Null,
Name nvarchar(50)Not Null,
Phone nvarchar(10)NULL,
Salary smallmoney Null,
PRIMARY KEY(ID)
);
```



## **CREATE TABLE** Statement

is used to store records (data). To create a database table, we use the SQL **CREATE TABLE** statement. Syntax:

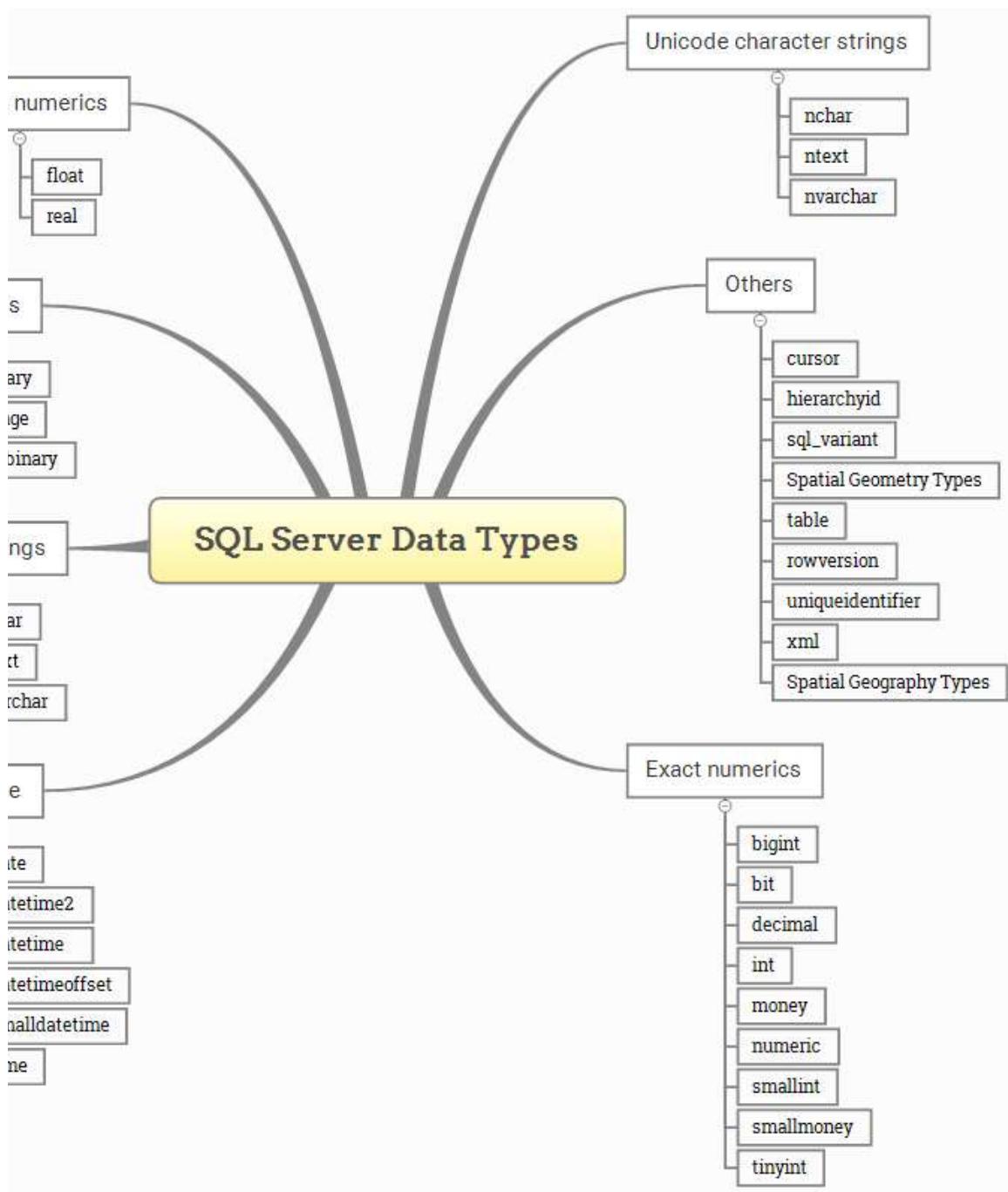
```
: table_name (  
datatype,  
datatype,  
datatype,
```

parameters specify the names of the columns of the table.

parameter specifies the type of data the column can hold (e.g. varchar, integer, date, etc.).

Provide data types for each column while creating a table. Learn more about [SQL Data Types](#) in the next lesson.

**SQL DataTypes**



بيقولك ماتستخدمش الانواع دي عشان هتلتغي قريب ntext وال text وال image  
واستخدم بدالهم ال varchar و nvarchar وال varbinary  
ودي الانواع و بتتشيل لحد قد ايه

## Exact data types

Exact data types store exact numbers such as integer, decimal, or monetary amount.

Boolean data type stores one of three values 0, 1, and NULL

tinyint, smallint, and tinyint data types store integer data.

Smallint and numeric data types store numbers that have fixed precision and scale. Note that decimal and numeric are similar data types.

Money and smallmoney data type store currency values.

The following table illustrates the characteristics of the exact numeric data types:

Lower limit	Upper limit	Memory
$-2^{63}$ (-9,223,372,036,854,775,808)	$2^{63}-1$ (-9,223,372,036,854,775,807)	8 bytes
$-2^{31}$ (-2,147,483,648)	$2^{31}-1$ (-2,147,483,647)	4 bytes
$-2^{15}$ (-32,767)	$2^{15}$ (-32,768)	2 bytes
0	255	1 byte
0	1	1 byte/8bit column
$-10^{38+1}$	$10^{38}-1$	5 to 17 bytes
$-10^{38+1}$	$10^{38}-1$	5 to 17 bytes
-922,337, 203, 685,477.5808	+922,337, 203, 685,477.5807	8 bytes
-214,478.3648	+214,478.3647	4 bytes

## te numeric data types

numeric data type stores floating point numeric data. They are often used in scientific calculations.

	<b>Lower limit</b>	<b>Upper limit</b>	<b>Memory</b>	<b>Precision</b>
	-1.79E+308	1.79E+308	Depends on the value of n	7 Digit
	-3.40E+38	3.40E+38	4 bytes	15 Digit

## trings data types

data types allow you to store either fixed-length (char) or variable-length data (varchar). The text data type can store up to 2^31 characters. The ntext data type can store up to 2,147,483,647 characters. The code page of the server.

	<b>Lower limit</b>	<b>Upper limit</b>	<b>Memory</b>
	0 chars	8000 chars	n bytes
	0 chars	8000 chars	n bytes + 2 bytes
ix)	0 chars	2^31 chars	n bytes + 2 bytes
	0 chars	2,147,483,647 chars	n bytes + 4 bytes

## Character string data types

Character string data types store either fixed-length (nchar) or variable-length (nvarchar) Unicode character data.

<b>Lower limit</b>	<b>Upper limit</b>	<b>Memory</b>
0 chars	4000 chars	2 times n bytes
0 chars	4000 chars	2 times n bytes + 2 bytes
0 chars	1,073,741,823 char	2 times the string length

---

## • data types

• data types store data and time data, and the date time offset.

	<b>Storage size</b>	<b>Accuracy</b>	<b>Lower Range</b>	<b>Upper Range</b>
time	8 bytes	Rounded to increments of .000, .003, .007	1753-01-01	9999-12-31
date	4 bytes, fixed	1 minute	1900-01-01	2079-06-06
datetime	3 bytes, fixed	1 day	0001-01-01	9999-12-31
datetime2	5 bytes	100 nanoseconds	00:00:00.0000000	23:59:59.9999999
datetimeoffset	10 bytes	100 nanoseconds	0001-01-01	9999-12-31
timeoffset	6 bytes	100 nanoseconds	0001-01-01	9999-12-31

In new application, you should use the **time**, **date**, **datetime2** and **datetimeoffset** data types. Because these types align standard and more portable. In addition, the **time**, **datetime2** and **datetimeoffset** have more seconds precision than **datetime**. **datetimeoffset** supports time zone.

## **String data types**

Types stores fixed and variable length binary data.

	<b>Lower limit</b>	<b>Upper limit</b>	<b>Memory</b>
	0 bytes	8000 bytes	n bytes
	0 bytes	8000 bytes	The actual length of data entered + 2 bytes
	0 bytes	2,147,483,647 bytes	

## types

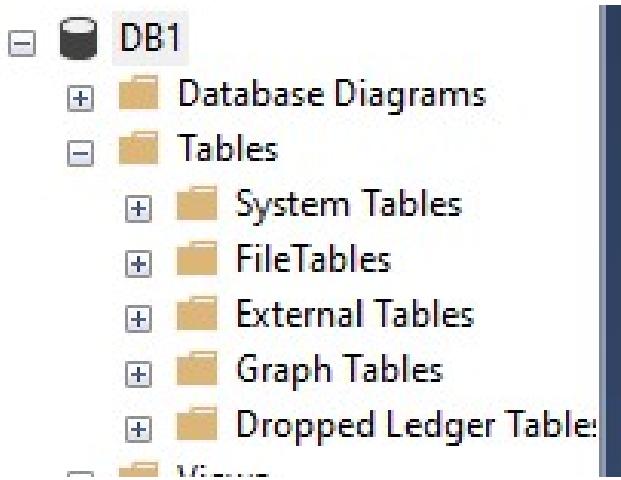
	<b>Description</b>
	for variables or stored procedure OUTPUT parameter that contains a reference to a cursor
	expose automatically generated, unique binary numbers within a database.
	represent a tree position in a tree hierarchy
tifier	16-byte GUID
	store values of other data types
	store XML data in a column, or a variable of XML type
metry	represent data in a flat coordinate system.
raphy	store ellipsoidal (round-earth) data, such as GPS latitude and longitude coordinates.
	store a result set temporarily for processing at a later time

### Drop Table

نحذف الجدول بالماوس او بالامر drop table وبعدين اسمه الجدول

```
use DB1;
```

```
drop table Employees;
```



## DROP TABLE Statement

**TABLE** is used to delete the tables in our database. For example,

```
Employees;
```

This command will delete a table named Employees.

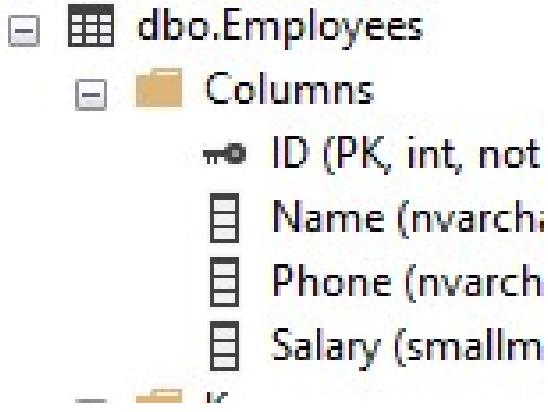
You have **admin** or **DROP** permission to run this command.

If you delete a database table, all records within a table are also deleted.

## DDL - Alter Table Statement

### Add Column

لحد دلوقتي احنا عملنا جدول وداتا بيز وحذفناهم عاوزين بقى نعدل عالجدول  
وأول تعديل هوا اننا نضيف عمود عالجدول  
دلوقتي احنا عندنا الجدول ده



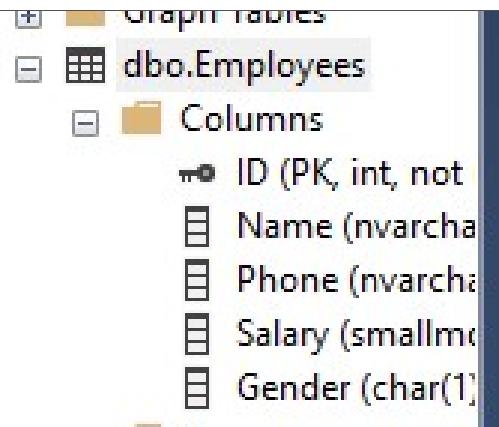
عاوزين نضيف عمود جديد اسمه gender يا male يا female ونوعه char

عشان اعملها بالكود بقوله alter table يعني عد عالجدول وبتكتب بعدها اسم الجدول

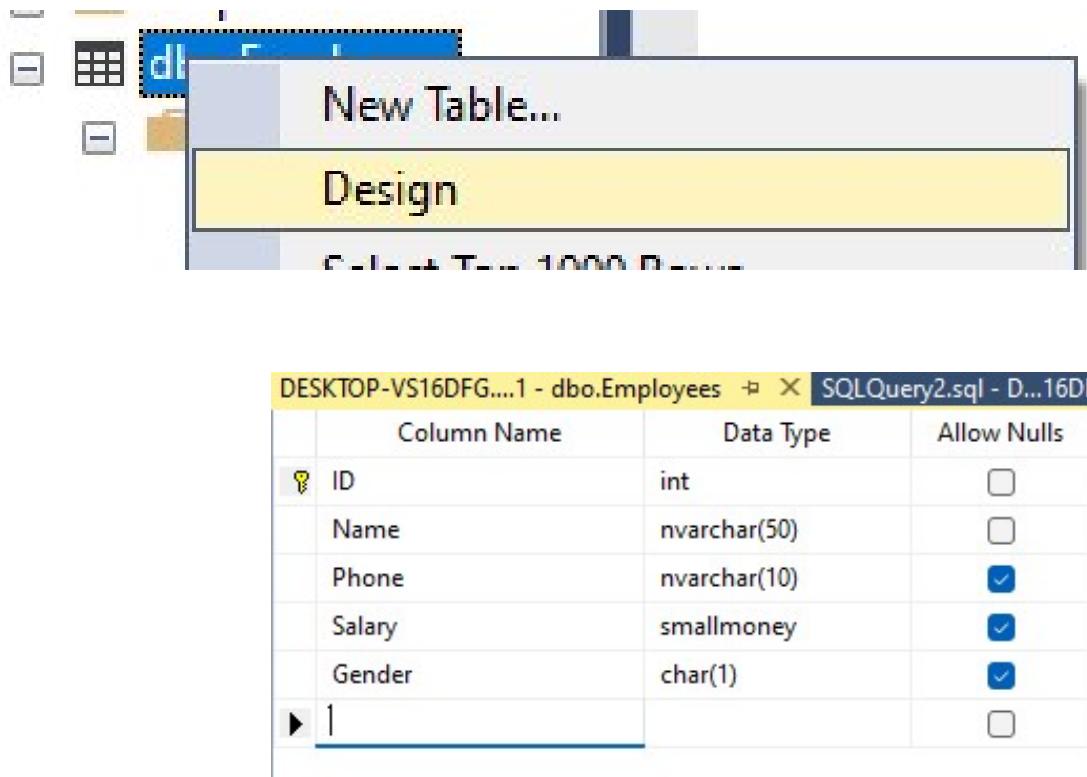
وبعدين بتكتب add وبعده اسم العمود وال data type  
لو ماحددتش null ولا not null هوا من نفسه بيعمله null يعني بيقبل ال null

```
use DB1;
```

```
alter table Employess
add Gender char(1);
```



لو عايز تضيف عمود بالماوس



## Add a Column in a Table

Adding a column in a table using the `ALTER TABLE` command with the `ADD` clause. For example,

```
Employees
    add Gendor char(1);
```

This command adds a column named `Gendor` in the `Employees` table.

## Rename Column

لـ عـاـيـز تـعـمـل `rename` لـعـمـود مـعـين بـالـمـاـوس بـتـرـوـح كـلـيـك يـمـين عـلـي الجـدـول وـتـخـتـار او كـلـيـك يـمـين عـالـعـمـود وـتـخـتـار `RENAME` وـتـعـدـلـه

انـما بـالـكـوـد هـتـكـتـب نـفـس السـطـر الـأـوـل الـي هـوـا

```
ALTER TABLE Employees
```

وـتـحـتـيه بـتـكـتـب `rename column` وـبـعـدـها الـاسـم القـدـيم وـبـعـدـين `to` وـبـعـدـين الـاسـم

الجديد

زي كده

```
ALTER TABLE Employees  
    RENAME COLUMN Gendor TO Gender;
```

بس هنا هيجبلك ال ERROR ده

Messages

```
Msg 102, Level 15, State 1, Line 4  
Incorrect syntax near 'RENAME'.
```

هنا بيقولك انه بيبقى فيه فروقات بسيطه بين كل داتايز والثانويه وهنا مش هيقبل ال SYNTAX اللي كتبناه

طب ايه البديل؟

قالك انه فيه function علي مستوى الداتا بيز ودي بتكون stored procedures عاديه

ال procedure دي اسمها sp\_rename بياخد اسم العمود والاسم الجديد وكلمة column لو في حالتنا هنغير اسم عمود

عشان نستدعي ال procedure ونشغلها بنكتب قبلها كلمه exec

زي كده

```
use DB1;
```

```
--ALTER TABLE Employees  
--RENAME COLUMN Gendor TO Gender;
```

```
exec sp_rename 'Employees.Gendor', 'Gender', 'COLUMN';
```

## Column in a Table (Most Databases)

Change columns in a table using the `ALTER TABLE` command with the `RENAME COLUMN` clause. For example,

```
USE Employees;
ALTER TABLE Employees
    RENAME COLUMN Gendor TO Gender;
```

This command changes the column name of `Gendor` to `Gender` in the `Employees` table.

## Column in table (in SQL Server)

You can use the `ALTER TABLE` statement in SQL Server to rename a column in a table. However, you can use `sp_rename`, though it's recommended that you drop and recreate the table so that scripts and stored procedures are not broken.

The syntax for renaming a column in an existing table in SQL Server (Transact-SQL) is:

```
EXEC sp_rename 'table_name.old_column_name', 'new_column_name', 'COLUMN';
```

### Rename a table

عاوزين نغير اسم الجدول مش العمود

في كل لغات ال sql بتسخدم الكود ده :-

بتكتب `alter table` وبعدها اسم الجدول وبعدين `to` وبعدين الاسم الجديد

زي كده

```
ALTER TABLE Emp
    RENAME TO Employees;
```

هنا لا بتسخدم ال procedure اللي اسمها `sp_rename`

بس المرادي بتبديها اسم الجدول القيم وبعددين الاسم الجديد يااما غيره بالماوس  
واخلص

زي كده

```
use DB1;

--ALTER TABLE Emp
-- RENAME TO Employees;

exec sp_rename 'Emp', 'Employees';
```

### able (Most Databases)

e name of a table using the `ALTER TABLE` command with the `RENAME` clause. For example,

```
>OldTableName
/TableName;
```

### le (In SQL Server)

re `ALTER TABLE` statement in SQL Server to rename a table. However, you can use `sp_rename`, though Microsoft  
you drop and recreate the table so that scripts and stored procedures are not broken.

ame a table in SQL Server (Transact-SQL) is:

```
re 'old_table_name', 'new_table_name';
```

### Modify a column

عاوزين نعدل على ال `data type` بتاعت العمود

هنسخدم برضه ال `alter table`

وبعدها هنقوله `alter column` وبعدها اسم العمود بتكتب التعديلات اللي

عاوز تعملها

زي كده

```
use DB1;
```

```
ALTER TABLE Employees  
ALTER COLUMN Name nvarchar(100) NOT NULL;
```

لو ماكتبتش NOT NULL هيعملها NULL تلقائي

في بعض أنواع الداتا بيز الثانيه بدل كلمه ALTER COLUMN بيكتب MODIFY COLUMN

### Column in a Table

غير the column's data type using the `ALTER TABLE` command with `MODIFY` or `ALTER COLUMN` clause. For

```
Employees  
Name VARCHAR(100);
```

```
Employees  
| Name VARCHAR(100);
```

```
Employees  
CHAR(100);
```

```
Employees  
Name VARCHAR(100);
```

mand changes the data type of the Name column to VARCHAR in the Employees table.

### **Delete a column**

عشان تحذف عمود بتكتب DROP وبعدها اسم العمود  
وطبعا ماتنساش السطر بتاع ALTER TABLE

```
ALTER TABLE Employees  
DROP COLUMN Gender;
```

### **Column in a Table**

remove) columns in a table using the ALTER TABLE command with the DROP clause. For example,

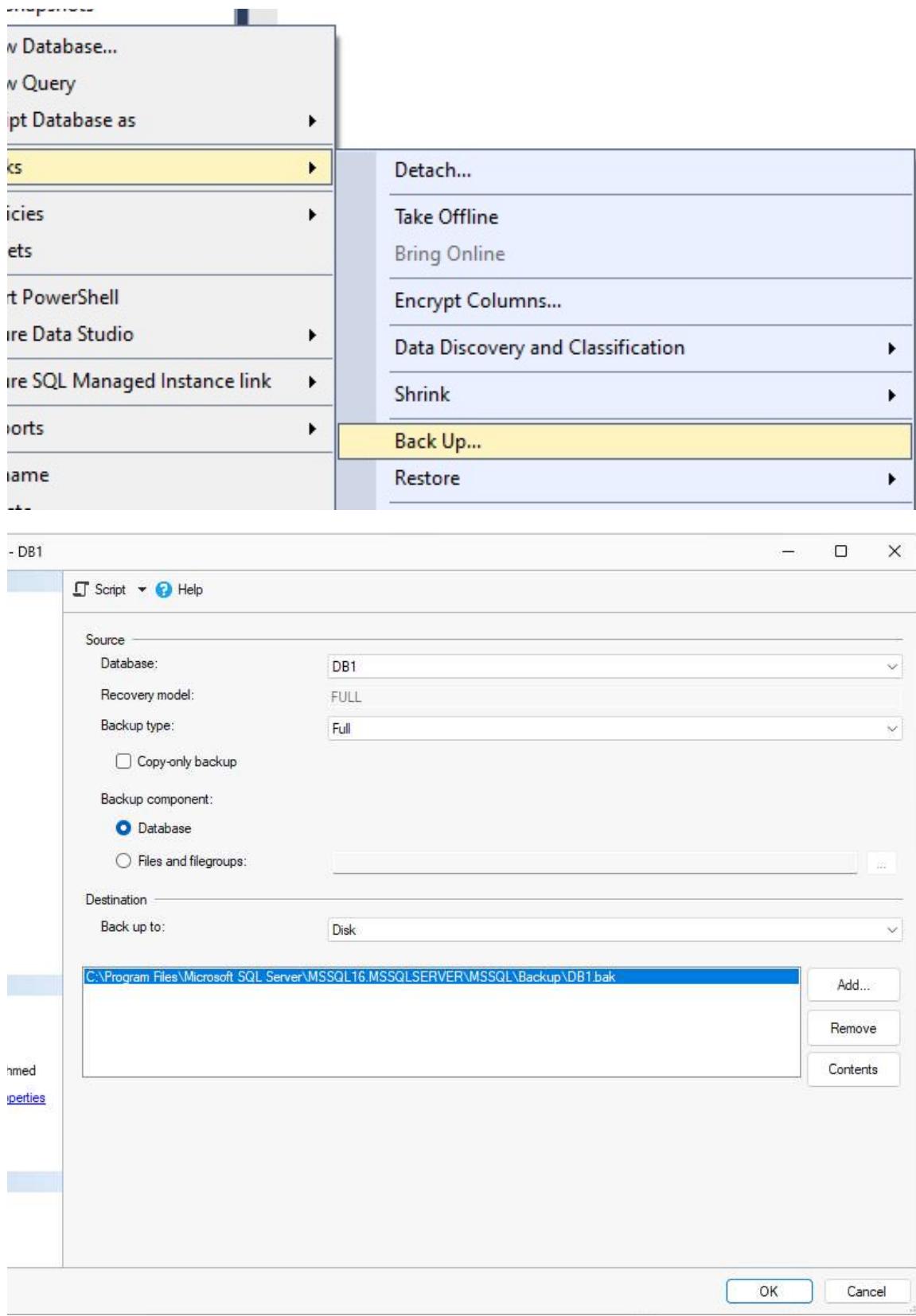
```
Employees  
Gender;
```

mand removes the Gender column from the Employees table.

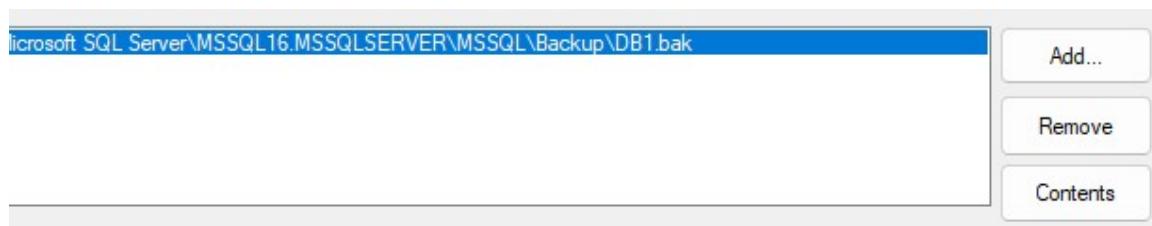
### **Backup & Restore Database**

#### **Backup Database**

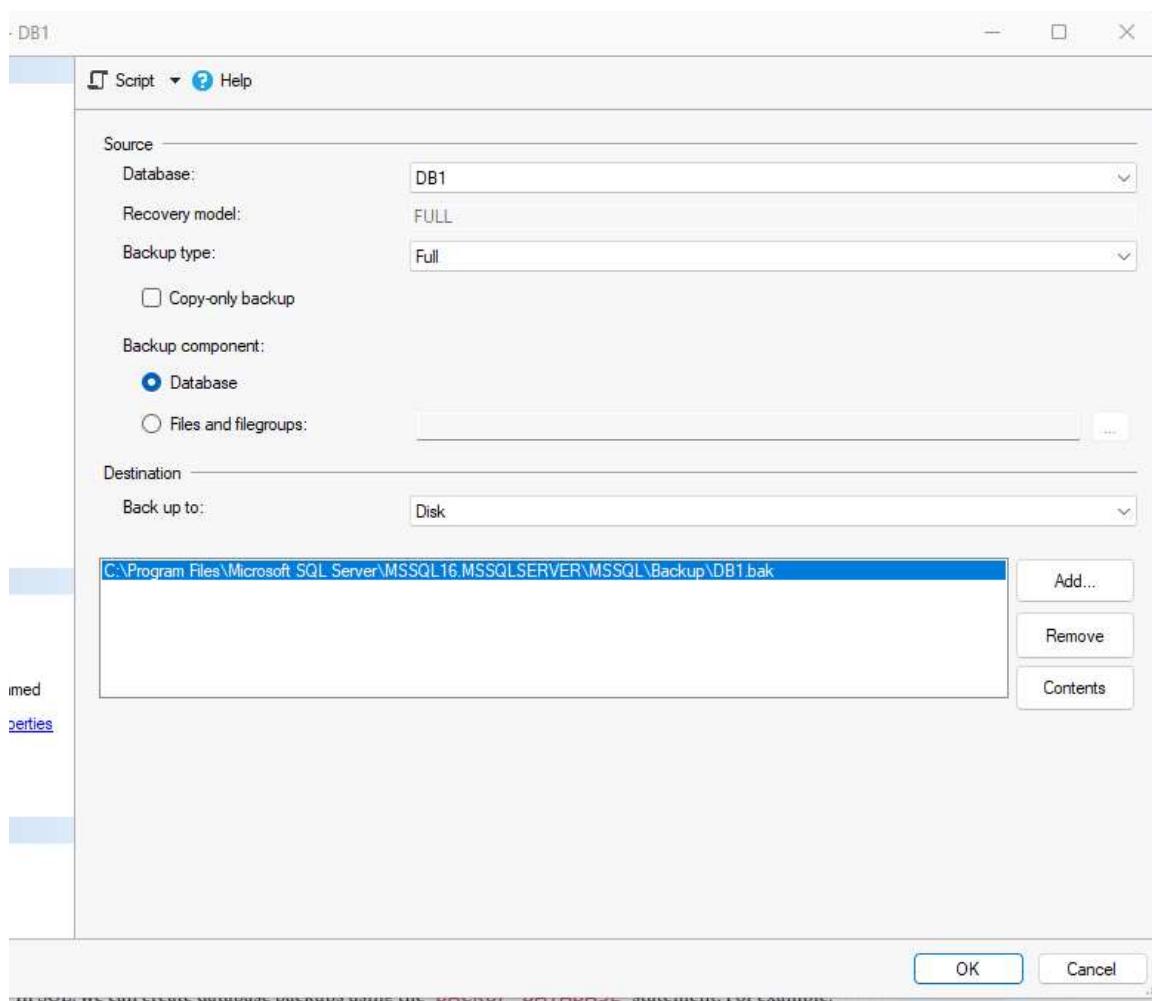
عاوزين ناخد نسخه احتياطيه من الداتابيز  
لو عاوزين نعملها بالماوس  
كليك يمين عالدادابيز اللي عاوزين نعملها BACKUP

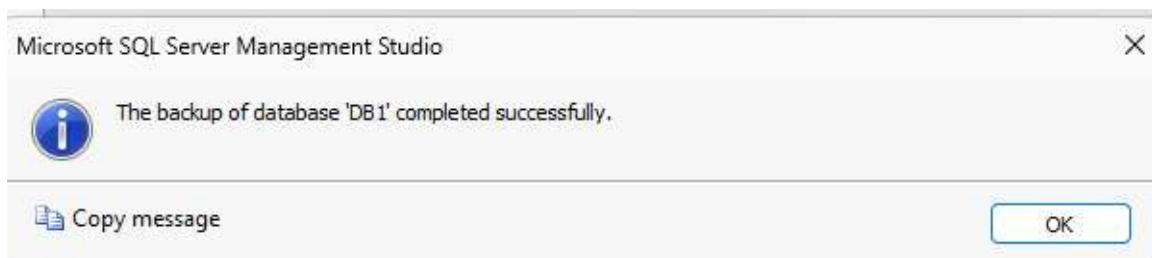


وبعدين من هنا بتشوف عاوز تحط ال BACKUP فين تقدر تشيل ال PATH ده  
وتعمل ADD وتحدد المكان اللي انت عاوزه

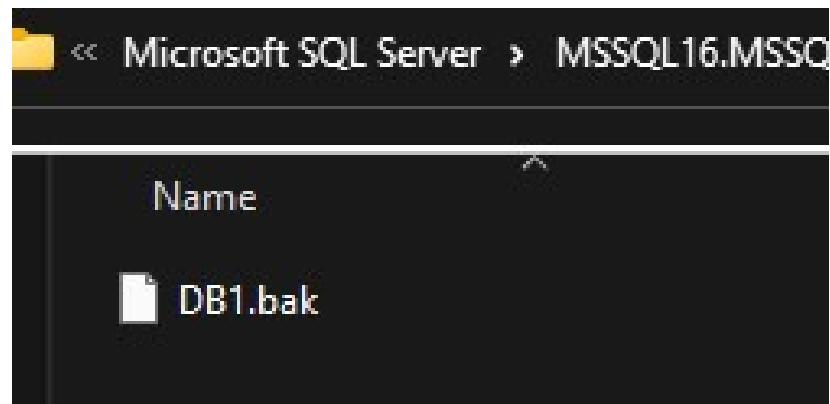


عادة بيحطوا bak. بعد اسم الملف بس لو ماجطيتهوش مش هيأثر  
وبعدين دوس ok





ده الملف اهـو



وهنا بيقولك لازم backup الـ يتعمل في حته غير اللي محظوظ فيها الويندوز عشان لو الويندوز او السيسـتم وقع انت هتقع معاـه حتى ماتحطهاش على نفس الجهاز

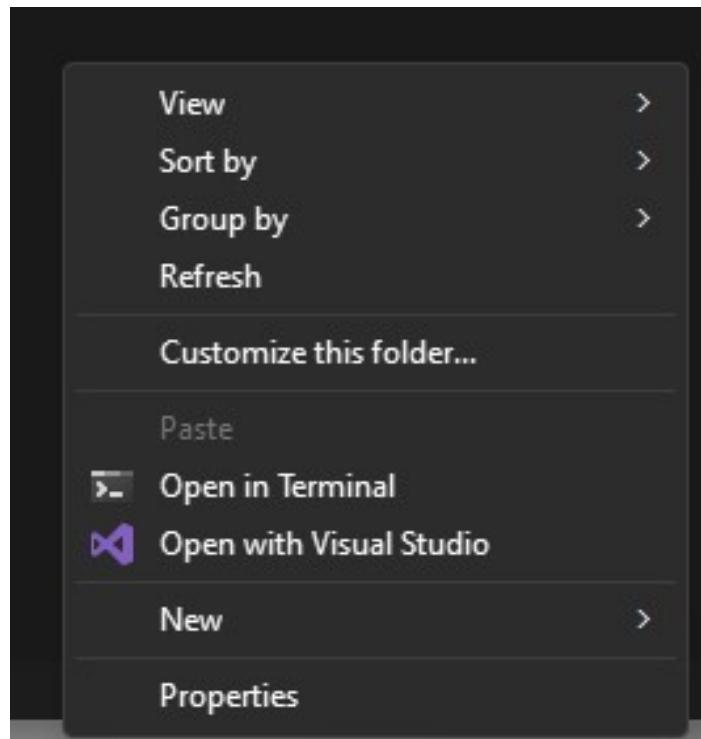
طيب عشان نعملها بالكود بنكتب `backup database` وبعدـها اسم الداتـا بـيز وبعدـين نـكتب `to disk` ونـكتب المسـار

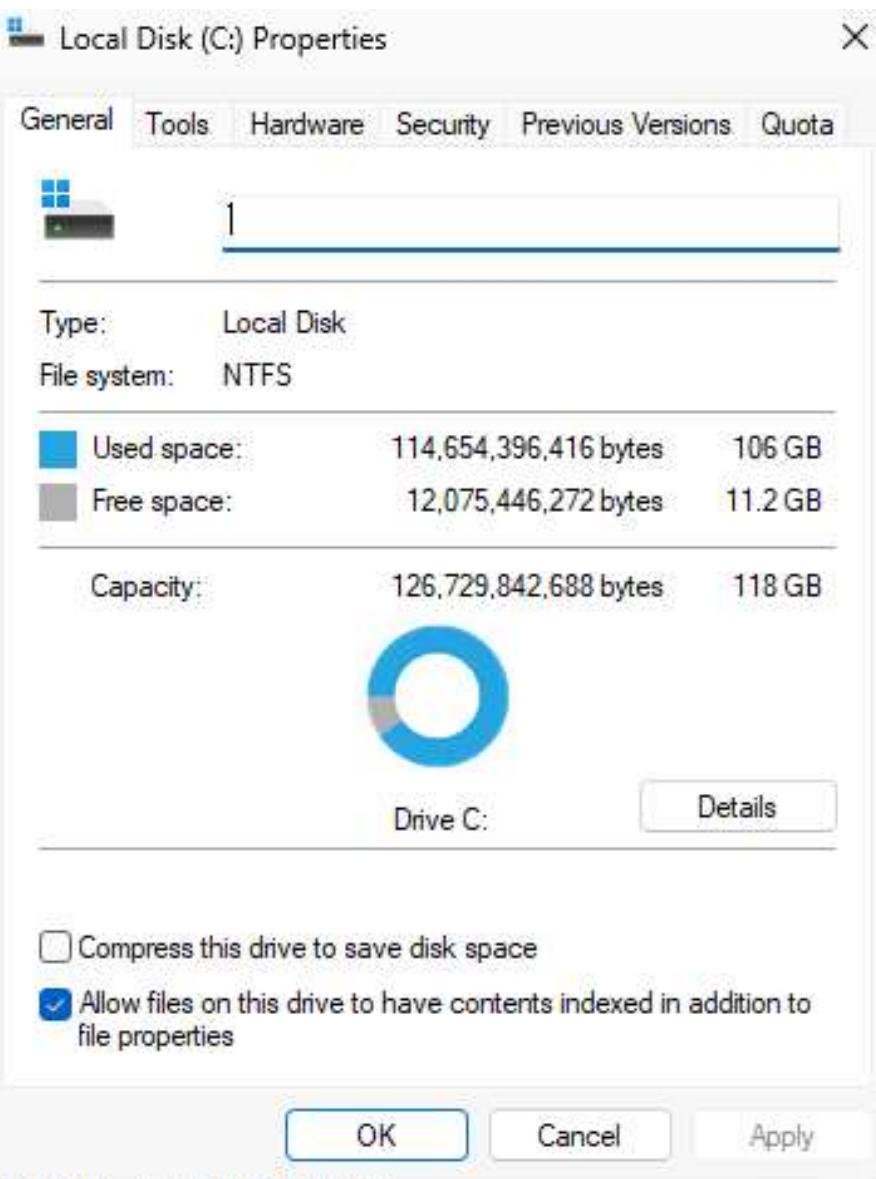
```
use DB1;
BACKUP database DB1
TO DISK ='C:\DB1.bak';
```

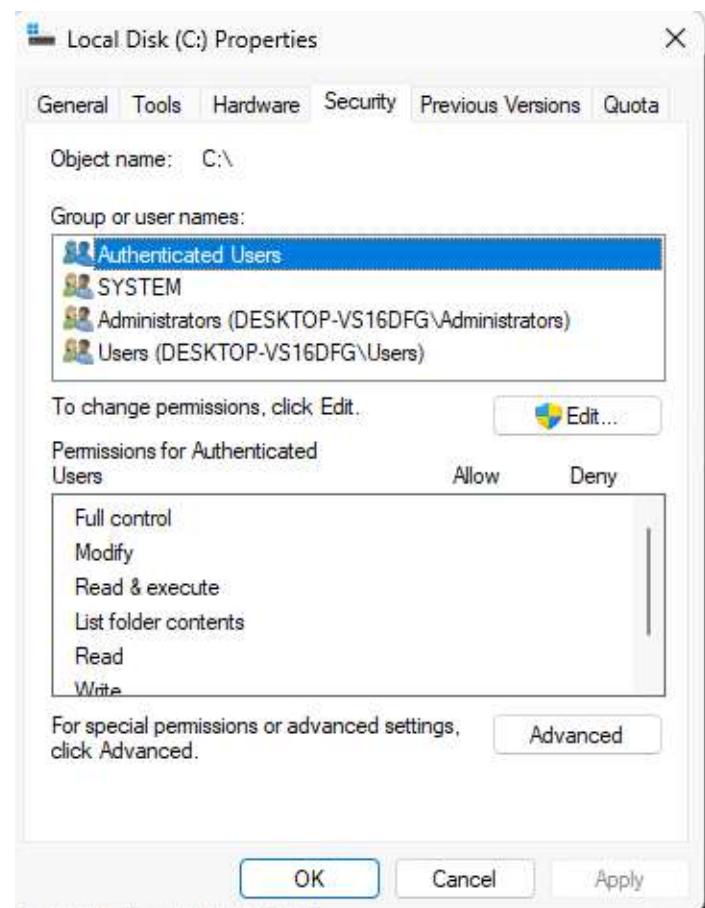
لو ادـاك `ERROR` من النوع Zi كـده

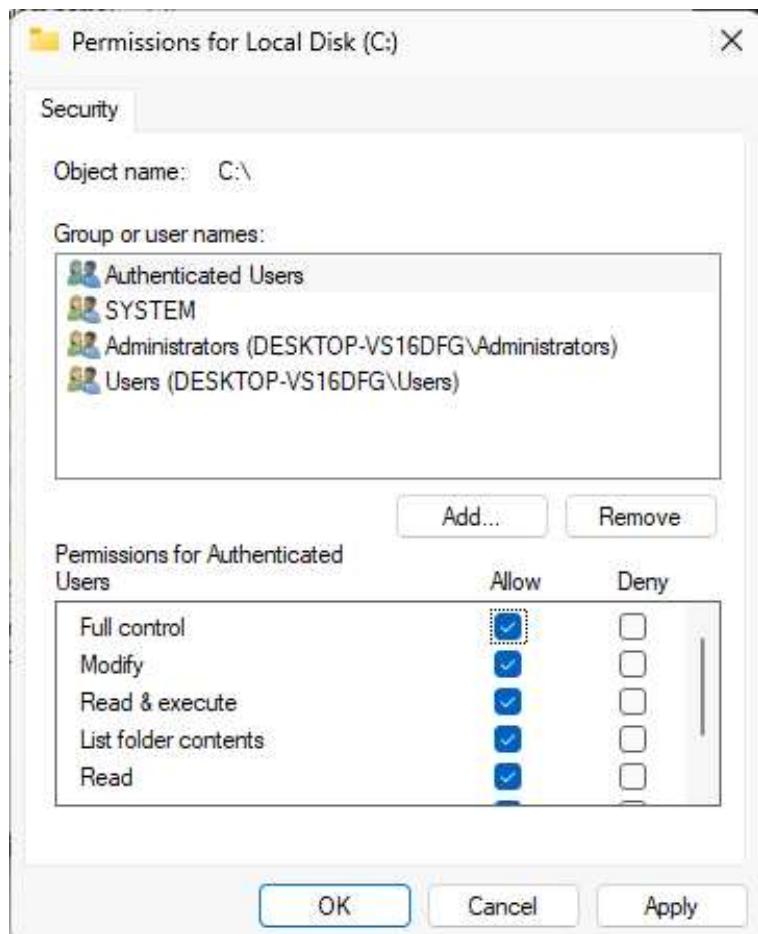
```
, Level 16, State 1, Line 3
open backup device 'C:\DB1.bak'. Operating system error 5(Access is denied.).
, Level 16, State 1, Line 3
!TABASE is terminating abnormally.
```

ده معناه انه اليوزر بتاعك بتاع الويندوز ماعندوش صلاحيات يدخل عال C او المكان  
اللي انت عاوز تحط فيه ال BACKUP  
فبتروح لل C DRIVE وكليك يمين









واقعد وافق على كل حاجه  
احنا كده بنأخذ FULL BACKUP لأننا بنأخذ الداتا بيز كلها هيله بيله ونسخها

## BACKUP DATABASE Statement

Create database backups regularly so that our data won't get lost if the database gets corrupted.

Create database backups using the `BACKUP DATABASE` statement. For example,

```
SE MyDatabase1  
\MyDatabase1_backup.bak';
```

This command creates a backup file of the MyDatabase1 database inside C drive, named MyDatabase1\_backup.bak.

It's a common convention to use the `.bak` file extension for database backup files, however, it's not mandatory.

If you backup the database to a different drive than the actual database. Then, if you get a disk crash, you will not lose your data, just the database.

### Differential Backup

ال Differential Backup مكلف من ناحية المساحة ومن ناحية الوقت عشان كده فيه حاجه تانيه اسمها differential backup ودي فكرتها انك اول مره بتاخذ

full backup عادي لكن بعد كده بتشوف ايه اللي اتحط زيادها وتعديل في البيانات وبيعدلها

طريقتها هيا نفس الطريقه اللي فاتت بس بزود كلمتين WITH DIFFERENTIAL

```
use DB1;  
  
BACKUP database DB1  
TO DISK = 'C:\DB1.bak'  
WITH DIFFERENTIAL;
```

## DIFFERENTIAL BACKUP DATABASE Statement

A differential backup backs up only the parts of the database that have changed since the last full database backup.

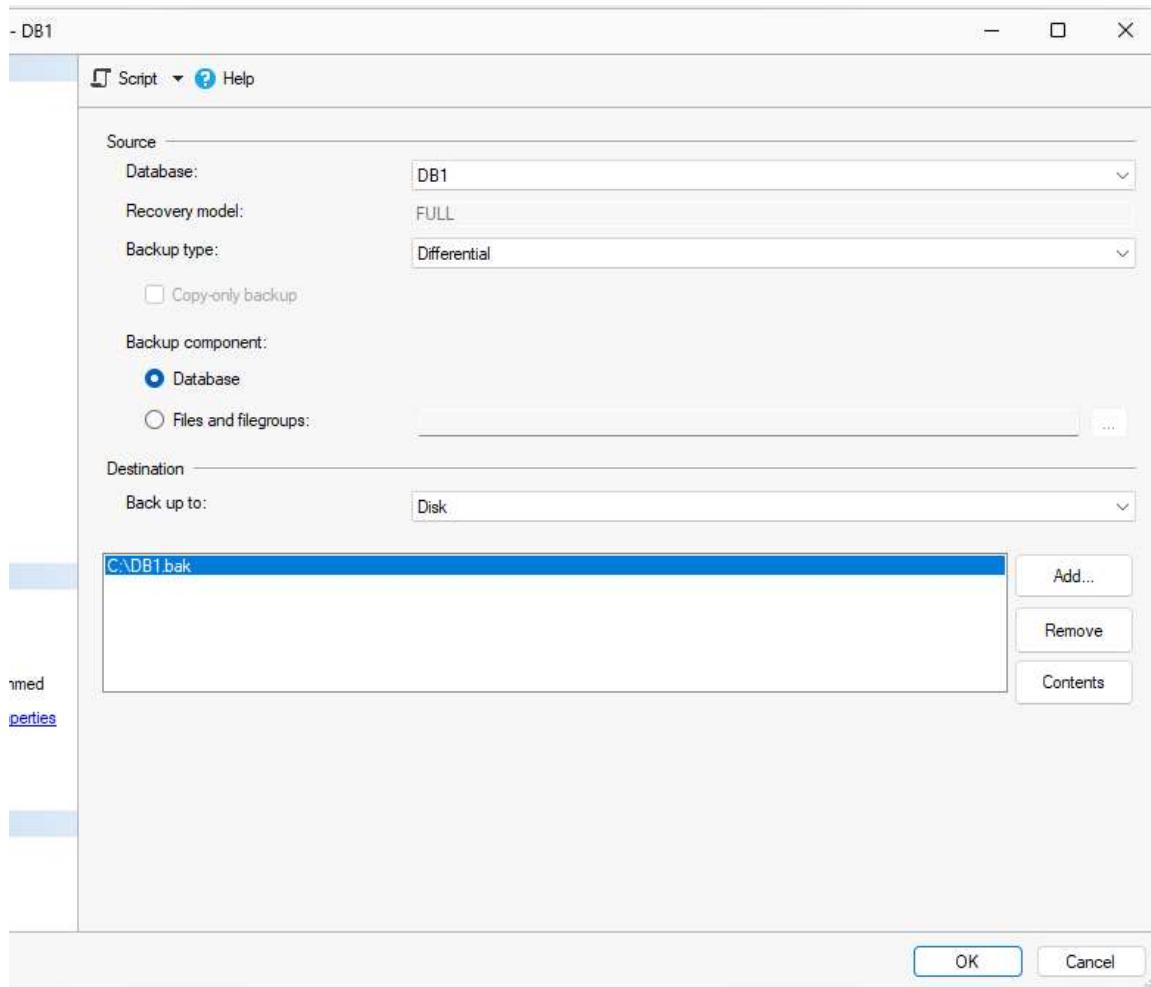
Create differential database backups using the `BACKUP DATABASE With Differential` statement. For example:

```
SE MyDatabase1
\MyDatabase1_backup.bak'
TIAL;
```

The command appends only new changes to the previous backup file. Hence, this command may work faster.

This command reduces the backup time (since only the changes are backed up).

وبالماوس نفس الطريقة بس بغير دى

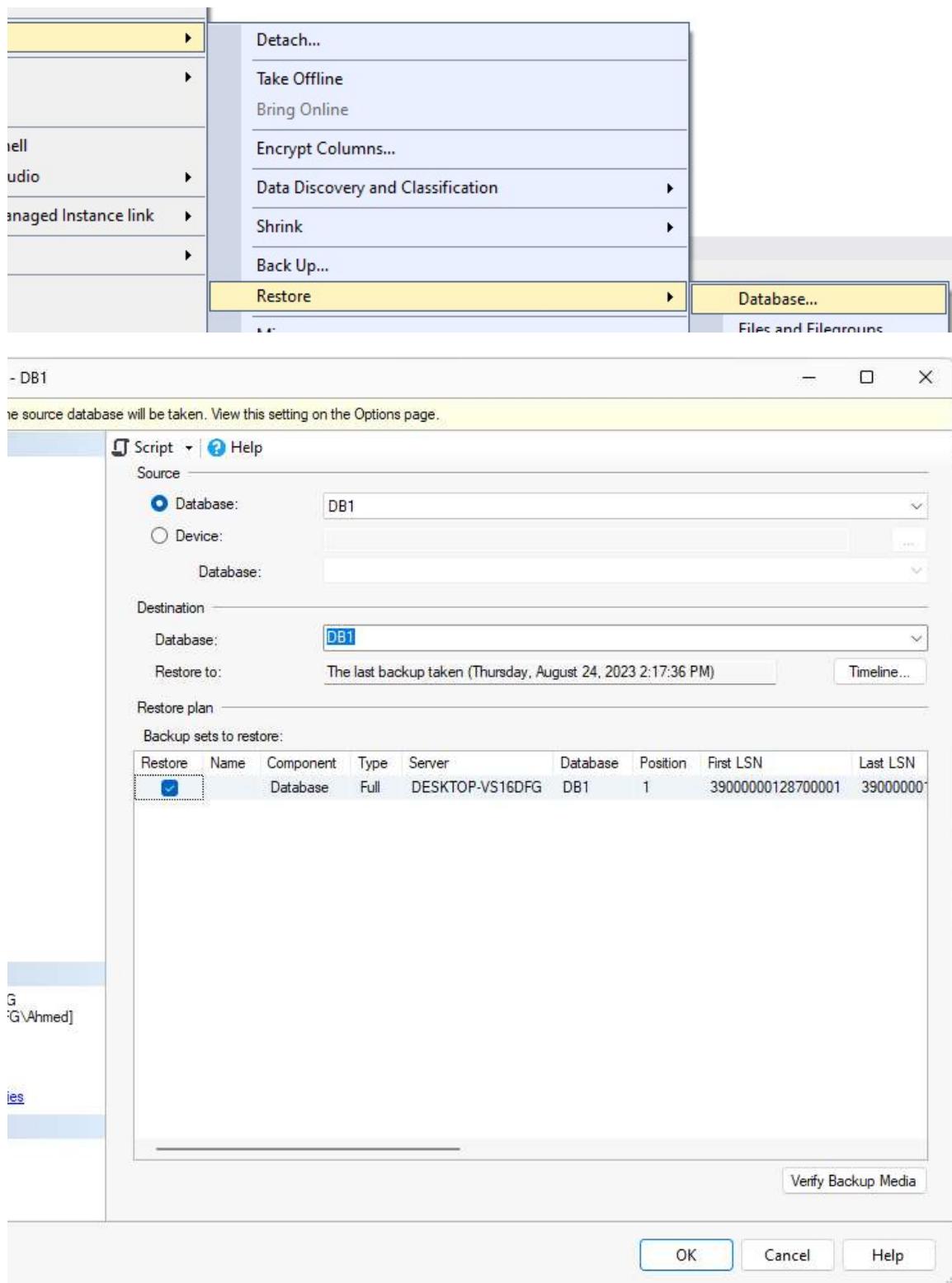


### Restore Database

عشان اعمل RESTORE لـ full backup بكتب نفس الكود بتاع ال backup بس  
بغير كلامه بكلمة restore بدل to بكتب restore backup

```
use master;
RESTORE database DB1
FROM DISK = 'C:\DB1.bak';
```

وبالماوس من هنا



## abase From Backup

If we want to restore a database from a backup file to the database management system, we can use the `RESTORE DATABASE` statement. For example,

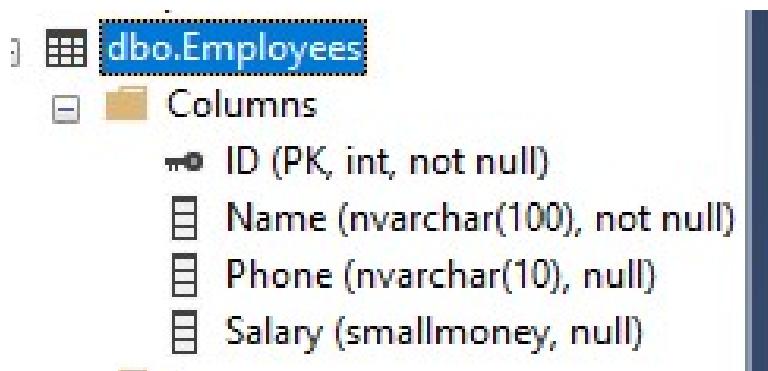
```
RESTORE DATABASE MyDatabase1  
FROM DISK = 'C:\MyDatabase1.bak';
```

This command restores the `MyDatabase1.bak` file in the database named `MyDatabase1`.

## Data Manipulation Language - DML

### Insert Into Statement

ال DML ودي اوامرها خاصه بالبيانات نفسها او ال records  
ال insert statement هيا بتخليك تدخل داتا جديدة او records عالجدول  
ده الجدول بتاعنا



اول حاجة عايز اعملها اني ابص عالجدول اشوف موجود فيه ايه وده مالوش علاقه  
بالكود اللي عايز اكتبه

عشان اعمل ده هقوله السطر ده واللي معناه اعرضلي كل حاجة من الجدول اللي  
اسمه employees

```
USE DB1;  
SELECT * FROM Employees;
```

وده الجدول اهو فاضي

طيب عاوزين بقى ندخل داتا عالجدول

قالك هتكتب `insert into` وبعدها اسم الجدول اللي هتدخل فيه البيانات وبعدين تكتب `values` وتفتح قوسين تحط جواهم القيم اللي عاوز تحطها بنفس ترتيب العمده اللي في الجدول

زي كده

```
INSERT INTO Employees
VALUES(1, 'Emp1', '979790', 10000);
```

وخليلك فاكير انك ممكن تحدد جزئية معينه بالماوس وينفذهالك هيا بس

لو جيت تدخل بيانات جديده وجيت تحط نفس ال id هيجيبلك خطأ انه ال id ده موجود قبل كده

تقدر تدخل اكتر من record بجمله `insert` واحده عن طريق تكرار الاقواس زي كده

```
INSERT INTO Employees
values
(2, 'Emp2', '1234', 700),
(3, 'Emp3', '5464', 400),
(4, 'Emp4', '7897', 900);
```

هنا لو عايز ادخل عمودين بس مش عايز ادخل الباقي  
 اول حاجه لازم الاعمدة اللي مش عايز تدخلها دي تكون بتقبل ال null  
 طيب عشان ادخل العمودين هعمل ايه ؟

قالك هتبيجي جنب ال insert into وتكلب اسم الجدول وتفتح قوسين تكتب فيهم  
 اسامي الاعمده اللي عاوز تحط فيهم الداتا وبعدين تشتعل عادي بس بالترتيب اللي  
 انت حاطه يعني لو كاتب انك عايز تدخل في ال name وبعدين ال id يبقى تكتب  
 الداتا بتاعت ال name الأول  
 زي كده

```
INSERT INTO Employees)ID,Name(
VALUES(5, 'Emp5');
```

	ID	Name	Phone	Salary
1	1	Emp1	979790	10000.00
2	2	Emp2	1234	700.00
3	3	Emp3	5464	400.00
4	4	Emp4	7897	900.00
5	5	Emp5	NULL	NULL

اقدر اعمل كده كمان

```
INSERT INTO Employees
VALUES(6, 'Emp6', null,null);
```

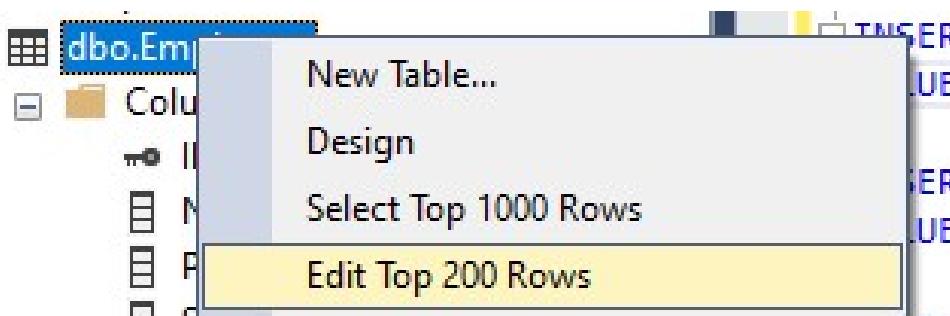
	ID	Name	Phone	Salary
1	1	Emp1	979790	10000.00
2	2	Emp2	1234	700.00
3	3	Emp3	5464	400.00
4	4	Emp4	7897	900.00
5	5	Emp5	NULL	NULL
6	6	Emp6	NULL	NULL

لو عايز احذف الداتا كلها بكتب delete from وبعدها اسم الجدول

```
DELETE FROM Employees;
```

ID	Name	Phone	Salary

لو حددت امر معين حتى لو كان معموله COMMENT تقدر تعمله execute عادي  
لو عايز تدخل بيانات بالماوس كليك يمين على الجدول وبتختار edit



The screenshot shows the context menu for the 'Employees' table in SQL Server Management Studio. The 'Edit Top 200 Rows' option is highlighted in yellow.

ID	Name	Phone	Salary
NULL	NULL	NULL	NULL

وبعدين اكتب اللي انت عاوزه

ممكن تعمل ده لو عاوز تجرب انما اعتقد في العادي انت هتحط جمله ال `insert` نفسها في البرنامج بتاعك

## O Statement

`ITO` statement is used to insert new records in a table.

### ) Syntax

rite the `INSERT INTO` statement in two ways:

ء column names and the values to be inserted:

```
e_name(column1, column2, column3, ...)  
value2, value3, ...);
```

g values for all the columns of the table, you do not need to specify the column names in the SQL query. However, either of the values is in the same order as the columns in the table. Here, the `INSERT INTO` syntax would be as

```
e_name  
value2, value3, ...);
```

وده الكود كله

```

--this will show all data in the table
select * from Employees;

--Insert one record at a time
Insert Into Employees
values
(10, 'Emp10', '079939', 1000);

--Insert one record at a time with some null values
Insert Into Employees
values
(11, 'Emp11', null, null);

--insert multiple records at a time.
Insert Into Employees
values
(2, 'Emp2', '552221', 700),
(3, 'Emp3', '55554', 300),
(4, 'Emp4', '322344', 400);

--insert only selected fields
Insert Into Employees (ID, Name)
values
(5, 'Emp5');

--if you forget to insert not null filed an error will occure.
Insert Into Employees (ID)
values (5);

select * from Employees;

--this will delete all records in table.
--delete from Employees;

```

### Update Statement

هنا عاوز اقولك انه التعديل عالبيانات اللي في الداتا بيز ما فيهاش undo يعني الداتا اللي هتعدلها مش هتعرف ترجعها تاني الا لو كنت عامل backup ليه بقولك كده ؟

بقولك كده عشان اوامر الحذف والتعديل لو ماحططيتش فيها جملة شرطيه بتحذف كل الداتا اللي في الجدول

او بتعدل علي كل الداتا في العمود اللي في الجدول

وقبل ماتعمل جمله حذف او تعديل لازم بتتأكد من الكود بتاعك قبل ما تشغله

طيب عشان نغير قيمة خليه او قيمه معينه في record معين بنكتب كلمة update وبعدها اسم الجدول اللي عايزين نعدل عليه وبعدين بنكتب كلمة set واسم العمود = القيمه الجديده ولو فيه حاجه تانيه في نفس ال record عايزين نعدلها بنعمل فاصله ونكتب اسم العمود = القيمه الجديده برضه

لحد هنا لو جينا شغلنا الامر ده هيغير كل القيم اللي في العمود بنفس القيمه اللي كتبناها

وعشان ده مايحصلش بنحط شرط وده بيتم بكلمة where

بص المثال ده

طيب دلوقتي احنا عندنا الجدول ده

ID	Name	Phone	Salary
1	2	Emp2	552221
2	3	Emp3	55554
3	4	Emp4	322344
4	5	Emp5	NULL
5	10	Emp10	079939
6	11	Emp11	NULL

عاوزين نغير اسم الموظف اللي ال id بتاعه = 2 نخليه abu Mohamed hadhoud

```
UPDATE Employees  
SET Name='Mohamed Abu-Hadhoud'  
where ID=2;
```

وده لو عاوزين نغير الاسم والمرتب

```

UPDATE Employees
SET Name='Mohamed Abu-Hadhoud', Salary=5000
where ID=2;

```

	ID	Name	Phone	Salary
1	2	Mohamed Abu-Hadhoud	552221	5000.00
2	3	Emp3	55554	300.00
3	4	Emp4	322344	400.00
4	5	Emp5	NULL	NULL
5	10	Emp10	079939	1000.00
6	11	Emp11	NULL	NULL

هنا ممكن نستخدم الشرط في صالحنا زي انتا مثلاً نعدل على كل الرواتب اللي اقل من 500 نزودها ب 200

```

UPDATE Employees
SET Salary=Salary+200
WHERE Salary<500;

```

	ID	Name	Phone	Salary
1	2	Mohamed Abu-Hadhoud	552221	5000.00
2	3	Emp3	55554	500.00
3	4	Emp4	322344	600.00
4	5	Emp5	NULL	NULL
5	10	Emp10	079939	1000.00
6	11	Emp11	NULL	NULL

طيب عاوزين المرتبات اللي اقل من 1000 نزودها ب 10%

```

UPDATE Employees
SET Salary=Salary*1.1
WHERE Salary<=1000;

```

		Results		Messages	
	ID	Name	Phone	Salary	
1	2	Mohamed Abu-Hadhoud	552221	5000.00	
2	3	Emp3	55554	550.00	
3	4	Emp4	322344	660.00	
4	5	Emp5	NULL	NULL	
5	10	Emp10	079939	1100.00	
6	11	Emp11	NULL	NULL	

## Statement

Statement is used to modify the existing records in a table.

### Syntax

ame

alue1, column2 = value2, ...

rn;

when updating records in a table! Notice the **WHERE** clause in the **UPDATE** statement. The **WHERE** clause specifies the record(s) that should be updated. If you omit the **WHERE** clause, all records in the table will be updated!

```

--this will show all data in the table
select * from Employees;

-- this will update one field at a time
Update Employees
set Name ='Mohammed Abu-Hadhoud'
where ID=2;

-- this will update multiple fields at a time.
Update Employees
set Name ='Mohammed Abu-Hadhoud' , Salary=5000
where ID=2;

-- this will increase the salary by 200 for all employees that their salaries are less than 500
update Employees
set Salary = Salary+ 200
where Salary < 500 ;

-- this will increase the salary by 10% for all employees that their salaries are less than or equal
1000
update Employees
set Salary = Salary *1.1
where Salary <= 1000;

```

### Delete Statement

لو عايز احذف الداتا كلها في الجدول كتب delete from و بعدها اسم الجدول

ولو عايز احذف record معين بحط شرط باستخدام ال where

**DELETE FROM Employees;**

The screenshot shows the SQL Server Management Studio interface with the 'Results' tab selected. A table named 'Employees' is displayed with columns: ID, Name, Phone, and Salary. The table is currently empty.

ID	Name	Phone	Salary

عيينا الجدول تاني

```

INSERT INTO Employees
VALUES
(1, 'Emp1', '079939', 1000),
(2, 'Emp2', '552221', 700),
(3, 'Emp3', '55554', 300),
(4, 'Emp4', '322344', 400),
(5, 'Emp5', null, null),
(11, 'Emp11', null, null);

```

	ID	Name	Phone	Salary
1	1	Emp1	079939	1000.00
2	2	Emp2	552221	700.00
3	3	Emp3	55554	300.00
4	4	Emp4	322344	400.00
5	5	Emp5	NULL	NULL
6	11	Emp11	NULL	NULL

دلوتي عاوزين نحذف أي record ال salary بتابعه ب null  
 كل اللي هنعمله هوا اننا هنزيد شرط علي امر الحذف  
 هنا مفيش حاجه اسمه null بقول بدلها = null

```

DELETE FROM Employees
WHERE Salary is null;

```

	ID	Name	Phone	Salary
1	1	Emp1	079939	1000.00
2	2	Emp2	552221	700.00
3	3	Emp3	55554	300.00
4	4	Emp4	322344	400.00

عاوزين نحذف اخر واحد اللي ال id بتاعه بـ 4

```
DELETE FROM Employees  
WHERE ID=4;
```

Results				
	ID	Name	Phone	Salary
1	1	Emp1	079939	1000.00
2	2	Emp2	552221	700.00
3	3	Emp3	55554	300.00

لو كتبت شرط مش متحقق مش هيعمل حاجه

## Statement

Statement is used to delete existing records in a table.

### NTAX

```
table_name WHERE condition;
```

when deleting records in a table! Notice the **WHERE** clause in the **DELETE** statement. The **WHERE** clause specifies which records should be deleted. If you omit the **WHERE** clause, all records in the table will be deleted!

```
--this will show all data in the table  
select * from Employees;  
  
-- this will delete all employees which their salary is null  
delete from Employees  
where salary is null;  
  
-- this will delete all employees that have their id=4 , which is one record in our case  
delete from Employees  
where ID=4;
```

## Select Into Statement

رجعنا الداتا اللي حذفناها

```

INSERT INTO Employees
VALUES
(1, 'Emp1', '079939', 1000),
(2, 'Emp2', '552221', 700),
(3, 'Emp3', '55554', 300),
(4, 'Emp4', '322344', 400),
(5, 'Emp5', null, null),
(11, 'Emp11', null, null);

```

	ID	Name	Phone	Salary
1	1	Emp1	079939	1000.00
2	2	Emp2	552221	700.00
3	3	Emp3	55554	300.00
4	4	Emp4	322344	400.00
5	5	Emp5	NULL	NULL
6	11	Emp11	NULL	NULL

دلوقي انا عايز انسخ الداتا اللي في الجدول ده لجدول تاني  
فيه اكتر من طريقة عشان تعمل ده  
بس فيه طريقة سهلة عشان تعمل كده

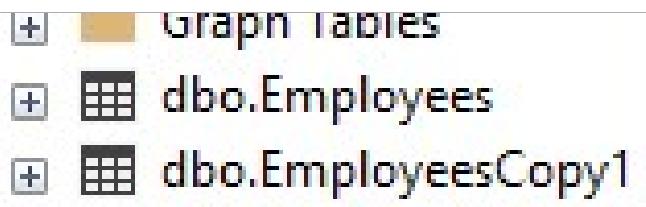
بنكتب SELECT \* INTO وكمان هيننسخ كل حاجة وبعدين بنعین اسم للجدول  
الجديد اللي هننسخ فيه الداتا والامر ده هو اللي هينشئ الجدول وبعدين تكتب  
وبعدها اسم الجدول اللي هتنسخ منه الداتا

لو عايز تنسخ داتا معينة بتزود الشرط ولو عايز تنسخ اعمدة محددة بتكتب اسم  
الاعمدة بدل ال \*

```

SELECT * INTO EmployeesCopy1
FROM Employees;

```



	ID	Name	Phone	Salary
1	1	Emp1	079939	1000.00
2	2	Emp2	552221	700.00
3	3	Emp3	55554	300.00
4	4	Emp4	322344	400.00
5	5	Emp5	NULL	NULL
6	11	Emp11	NULL	NULL

لو جيت انت عملت الجدول وكتبت نفس الامر هيطلعلك ERROR  
وهنا هننسخ عمودين بس

```
SELECT ID,Name INTO EmployeesCopy2
From Employees;
```

```
SELECT * FROM EmployeesCopy2;
```

	ID	Name
1	1	Emp1
2	2	Emp2
3	3	Emp3
4	4	Emp4
5	5	Emp5
6	11	Emp11

طيب لو عايز انسخ اسامي الاعمد بس  
حط شرط النتيجه بتاعته ب false

```
SELECT * INTO EmployeesCopy3  
FROM Employees  
WHERE 5=6;
```

```
SELECT * FROM EmployeesCopy3;
```

The screenshot shows a database query results window. At the top, there are two tabs: 'Results' (selected) and 'Messages'. Below the tabs is a table with four columns: 'ID', 'Name', 'Phone', and 'Salary'. There are no rows of data in the table.

ID	Name	Phone	Salary

## INTO Statement

Copy data from one database table to a new table using the **SELECT INTO** command. For example,

```
:sCopy  
:es;
```

Command copies all data from the Employees table to the new EmployeesCopy table.

**SELECT INTO** statement creates a new table. If the database already has a table with the same name, **SELECT** error.

If we want to copy data to an existing table (rather than creating a new table), we should use the **INSERT INTO SELECT** statement.

```
--this will show all data in the table
select * from Employees;

-- this will create a new table named EmployeesCopy1 based on the selected columns then it will copy
the data from Employees table based on the condition provided
SELECT *
INTO EmployeesCopy1
FROM Employees;

select * from EmployeesCopy1;

-- this will create a new table named EmployeesCopy1 based on the selected columns then it will copy
the data from Employees table based on the condition provided
SELECT ID, Name
INTO EmployeesCopy2
FROM Employees;

select * from EmployeesCopy2;

-- this will create a new table named EmployeesCopy1 based on the selected columns then it will
-- copy the data from Employees table based on the condition provided which is false means no data
will be copied
SELECT *
INTO EmployeesCopy3
FROM Employees
where 5=6;

select * from EmployeesCopy3;
```

### Insert Into ..Select From Statement

حذفنا الجداول القديمة

```
use DB1;
DROP TABLE Employees;
DROP TABLE EmployeesCopy1;
DROP TABLE EmployeesCopy2;
DROP TABLE EmployeesCopy3;
```

هنجعل جدولين تانيين

```

use DB1;

CREATE TABLE Persons(
ID int not null,
Name nvarchar(50)not null,
Age tinyint not null,
PRIMARY KEY(id)
);

SELECT * INTO OldPersons
FROM Persons;

ALTER TABLE OldPersons
ADD PRIMARY KEY(ID);

Insert INTO Persons
VALUES
(1, 'Mohamed', 45),
(2, 'Ali', 30),
(3, 'Amjad', 25),
(4, 'Maha', 20),
(5, 'Shadi', 22);

SELECT*FROM Persons;
SELECT*FROM OldPersons;

```

	ID	Name	Age
	ID	Name	Age
1	1	Mohamed	45
2	2	Ali	30
3	3	Amjad	25
4	4	Maha	20
5	5	Shadi	22

دلوقي انت لما تيجي تدخل داتا جديده عالجدول بتعمل ايه ؟

بكتب امر insert into

طيب للو عايز تختار كل الداتا من الجدول بتعمل ايه ؟

بكتب امر select from

طيب احنا دلوقي عاوزين ننسخ الداتا اللي موجوده في ال persons ونحطها في ال old persons

لو ماكانش جدول ال old persons موجود كنا استخدمنا ال select into عادي  
لكن الجدول موجود عندنا هنعمل ايه ؟

هستخدم جمله ال insert مع جملة ال select يعني هعمل select للداتا الموجوده في الجدول بتاع ال persons وھعمل بيها insert في الجدول بتاع ال old persons طيب ازاي ؟

بكل بساطه تكتبهم تحت بعض

```
INSERT INTO OldPersons  
SELECT * FROM Persons;
```

	ID	Name	Age
1	1	Mohamed	45
2	2	Ali	30
3	3	Amjad	25
4	4	Maha	20
5	5	Shadi	22

هوا كده هيأخذ الداتا اللي ناتجه عن ال SELECT STATEMENT وينسخها في الجدول

وبرضه تقدر تستخدم الشرط

طيب هنحذف الداتا اللي موجوده في ال old persons  
ونرجع ننسخ الأشخاص اللي سنهem اكتر من 30

```
DELETE FROM OldPersons;  
INSERT INTO OldPersons  
SELECT * FROM Persons  
WHERE AGE>=30;
```

Results		Messages	
	ID	Name	Age
1	1	Mohamed	45
2	2	Ali	30

## INTO SELECT Statement

copy records from one table to another with the help of examples.

**INTO SELECT** statement is used to copy records from one table to another existing table. For example,

```
OldPersons
;
```

This command copies all records from the Persons table to the OldPersons table.

This command,

must already have a table named OldPersons

The names of the OldPersons table and the Persons table must match

If we want to insert data to a new table (rather than copying in an existing table), we should use the **SELECT INTO** statement.

**from one table to another table:**

```
|e2
```

```
table1
```

```
;
```

olumns from one table into another table:

```
|e2(column1, column2, column3, ...)
```

```
column2, column3, ...
```

```
;
```

```
OldPersons  
on Persons  
30;
```

## Misc 1

### Identity Field (Auto Increment)

دلوقي احنا عاوزين عمود ال ID يدخل الرقم لوحده من غير ماكتبه بابدي الحركة  
دي اسمها

auto increment او auto numbering

الفكرة منها انه هيرقم ال id بدلالك احسن ماتتلخبط وانت بت رقم

تعالي نعمل جدول جديد بالماوس ونعمل ال auto increment فيه



	Column Name	Data Type	Allow Nulls
	ID	int	<input type="checkbox"/>
►	Department	nvarchar(200)	<input type="checkbox"/>
			<input type="checkbox"/>

	Column Name	Data Type	Allow Nulls
►	ID	Set Primary Key	
	Department	int	<input type="checkbox"/>

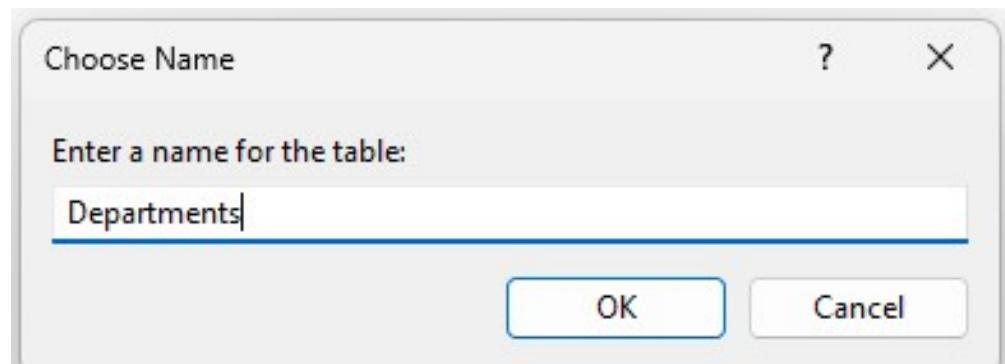
الـ column auto increment بـنلاقيها في الجزء اللي تحت اللي اسمه properties

Binding	ID No int

بتبعدين بتuros 2 كليك بالماوس علي identity هتقلب ب yes وبعدين هيظهر تحتها

و دي بتقولك عاوز لما تحط record جديد ال id يزيد بكم  
ال identity seed يتقولك عاوز العد يبدأ من كام

ification	Yes
y)	Yes
increment	1
eed	1
	✓



تعالي بقى نجرب ندخل أي حاجه بالماوس في الجدول لو جيت تكتب ال id بايديك  
مش هيقبل لانه اصبح بيتعدل من نفسه

	ID	Department
	1	Marketing
	2	Finance
	3	Computer
	4	Sales
**	NULL	NULL

حتي لما تيجي تعمل أوامر ال insert وجيتن تحط قيمة لل id هيطلع لك error

عشان كده طالما عملت عمود من الاعمده auto increment ماتحاوش تدخل قيم  
ليه

```
use DB1;

SELECT * FROM Departments;

INSERT INTO Departments
values( 'HR' );
```

ID	Department
1	Marketing
2	Finance
3	Computer
4	Sales
5	HR
6	HR
7	HR
8	HR
9	HR
10	HR
11	HR

طيب لو عايز اعرف اخر id موجود في الجدول بكتب السطر ده هنيجي لشرحه بعدين

```
print @@identity
```

Messages

11

Completion time: 2023-08-26T12:52:25.6505757+03:00

هوا هنا بيخزن ال id في متغير وبيبدأ يزود عليه

طيب لو انا جيت وحذفت الداتا اللي في الجدول كلها وبعدها ضيفت records  
جديده هل هيبدأ يعد من 1 تاني ؟

قالك لا بيكمel من العدد من حيث انتهي

```
use DB1;
SELECT * FROM Departments;
INSERT INTO Departments
values('HR');
print @@identity
delete from Departments;
```

ID	Department
1	HR

عشان اعمل نفس الجدول بالكود بعمله عادي جدا بس باجي بعد ال DATA TYPE  
وبكتب identity وبحط الرقم اللي هيزيد بيها والرقم اللي هيبدأ بيها العدد

```
create table Departments)
ID int identity(1,1)not null,
Department nvarchar(200) not null,
Primary key(ID)
);
```

---

## IDENTITY Field

lows a unique number to be generated automatically when a new record is inserted into a table.

primary key field that we would like to be created automatically every time a new record is inserted.

### SQL Server

The following SQL statement defines the "Personid" column to be an auto-increment primary key field in the "Persons" table:

```
CREATE TABLE Persons (
    Personid INT IDENTITY(1,1) PRIMARY KEY,
    FirstName NVARCHAR(255) NOT NULL,
    LastName NVARCHAR(255),
```

This query uses the `IDENTITY` keyword to perform an auto-increment feature.

In this example, the starting value for `IDENTITY` is 1, and it will increment by 1 for each new record.

If you want to change the starting value for the "Personid" column to 10 and increment by 5, change it to `IDENTITY(10,5)`.

When inserting a new record into the "Persons" table, we will NOT have to specify a value for the "Personid" column (a unique value will be assigned automatically by the database engine):

```
INSERT INTO Persons (FirstName,LastName)
VALUES ('Mohammed','Abu-Hadhoud');
```

This query would insert a new record into the "Persons" table. The "Personid" column would be assigned a unique value.

The "FirstName" column would be set to "Mohammed" and the "LastName" column would be set to "Abu-Hadhoud".

```
CREATE TABLE Departments (
    ID int identity(1,1) NOT NULL,
    Name nvarchar(50) NOT NULL,
    PRIMARY KEY (ID)
);
```

```
-----  
insert into Departments  
values ('HR');  
  
print @@identity;
```

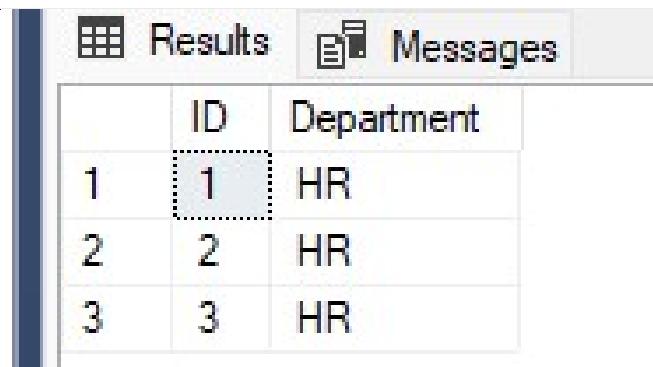
### Delete vs Truncate statement.

ال هيا ال truncate بس بتفرق عنها في حاجنیم اول حاجه انك ماينفعش تستخدم فيها ال where يعني ماينفعش تضييف شروط لأنها بتحذف كل ال records وتنبي حاجه انها بتتصفر ال identity يعني بعد ما تحذف وتضييف جديد هيبدا بعد معاك من البدايه record

ال delete بيسحهم واحد واحد انما ال truncate بيعمل للجدول

```
SELECT * FROM Departments;
```

```
insert into Departments  
values ('HR');
```



	ID	Department
1	1	HR
2	2	HR
3	3	HR

```
truncate table Departments;
```

The screenshot shows a SQL query results grid. The top bar has a zoom level of 100%. Below it are tabs for 'Results' and 'Messages'. The results grid has two columns: 'ID' and 'Department'. A single row is displayed with values 1 and HR respectively. The 'ID' column is highlighted with a dashed border.

	ID	Department
1	1	HR

## Truncate

The main difference between both statements is that `DELETE FROM` statement supports `WHERE` clause while `TRUNCATE` does not.

`DELETE FROM` statement does not reset the auto number (identity field) while the `TRUNCATE` does reset the identity.

You can delete single or multiple rows using the `DELETE FROM` statement while the `TRUNCATE` statement deletes all the rows in the table at once.

For example, if you want to use `TRUNCATE` statement with `DELETE FROM` statement by omitting the `WHERE` clause. For example,

```
TRUNCATE TABLE Customers;
```

```
DELETE Customers;
```

```

select * from Departments;

--this will delete all rows but will not reset the identity counter.
delete from Departments;

--this will delete all rows and reset the identity counter.
truncate table departments;

insert into Departments
values ('HR');

print @@identity;

```

## Foreign Key Constraint

ال primary key هو constraint not null وال constraint برضه primary key وكمان ال foreign key هو عباره عن constraint واتكلمنا عن ال foreign key قبل كده  
تعالي نعمل الجدولين دول جدول customers وجدول orders هنحط foreign key لجدول ال orders عشان يقولنا مين اللي اكل العجبنه (مين اللي طلب الاوردر) فبيقولك الجدول اللي من غير foreign key خليه هوا الأول في الكود

عشان اعمل foreign key بعمل عمود عادي وبعد ال data type بكتب references وبعدين اسم الجدول اللي عايز اربط بيها وافتح قوسين اكتب فيهم اسم العمود اللي بيمثل ال primary key في الجدول الاولاني

```

use DB1;

-- This table doesn't have a foreign key
CREATE TABLE Customers (
    id INT ,
    first_name VARCHAR(40),
    last_name VARCHAR(40),
    age INT,
    country VARCHAR(10),
    PRIMARY KEY (id)
);

-- Adding foreign key to the customer_id field
-- The foreign key references to the id field of the Customers table
CREATE TABLE Orders (
    order_id INT,
    item VARCHAR(40),
    amount INT,
    customer_id INT REFERENCES Customers(id),
    PRIMARY KEY (order_id)
);

```

### Columns

- order\_id (PK, int, not null)
- item (varchar(40), null)
- amount (int, null)
- customer\_id (FK, int, null)

عشنان تشووف العلاقة كليك يمين على شاشة ال query

SQLQuery1.sql - D...16DFG\Ahmed (58)\* X

```
use DB1;

-- This table doesn't have a foreign key
CREATE TABLE Customers (
    id INT,
    first_name VARCHAR(40),
    last_name VARCHAR(40),
    age INT,
    country VARCHAR(10),
    PRIMARY KEY (id)
);

-- Adding foreign key to the customer_id field
-- The foreign key references to the id field of the Customers table
CREATE TABLE Orders (
    order_id INT,
    item VARCHAR(40),
    amount INT,
    customer_id INT REFERENCES Customers(id),
    PRIMARY KEY (order_id)
);
```

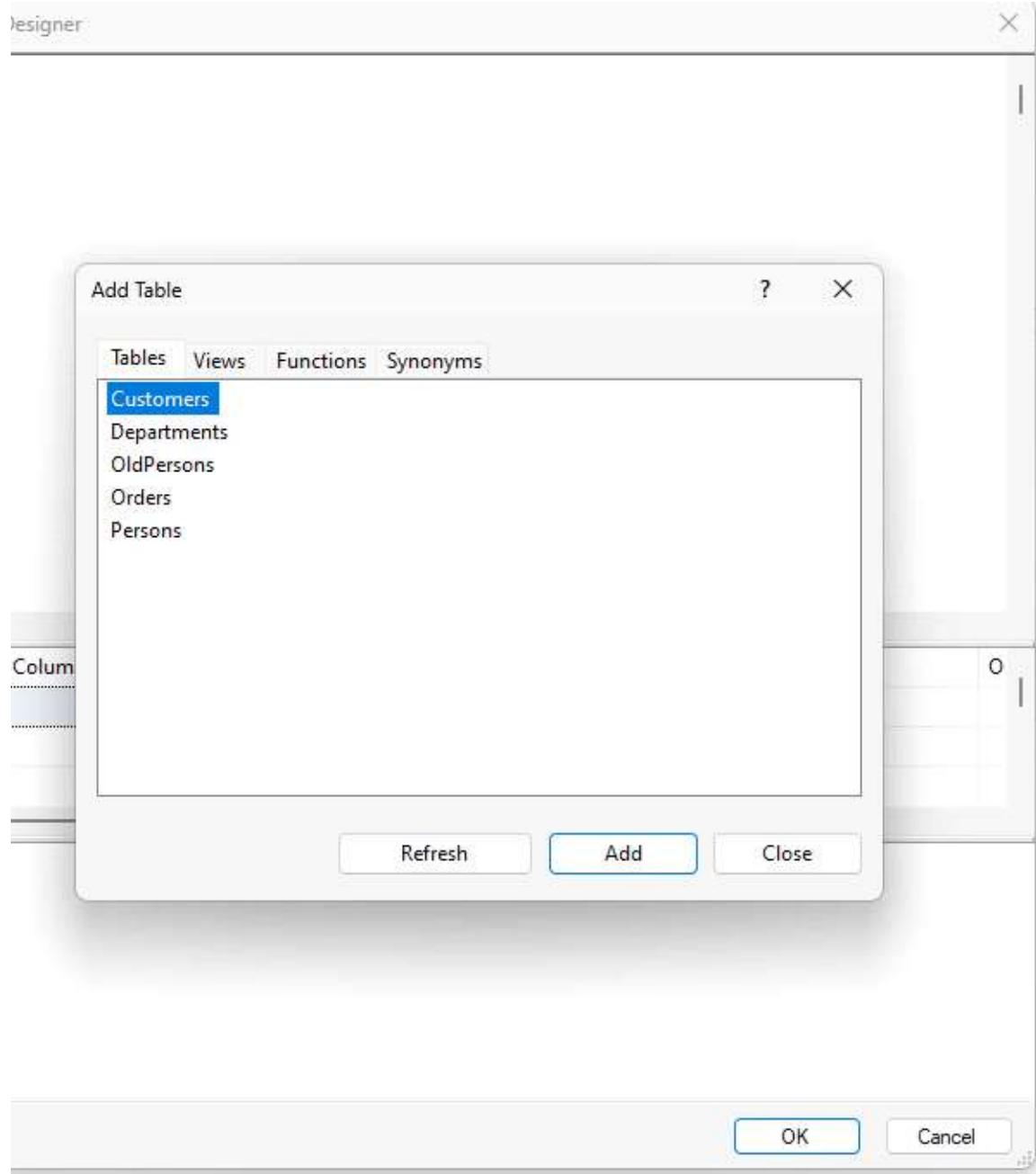
100 % ←

Messages

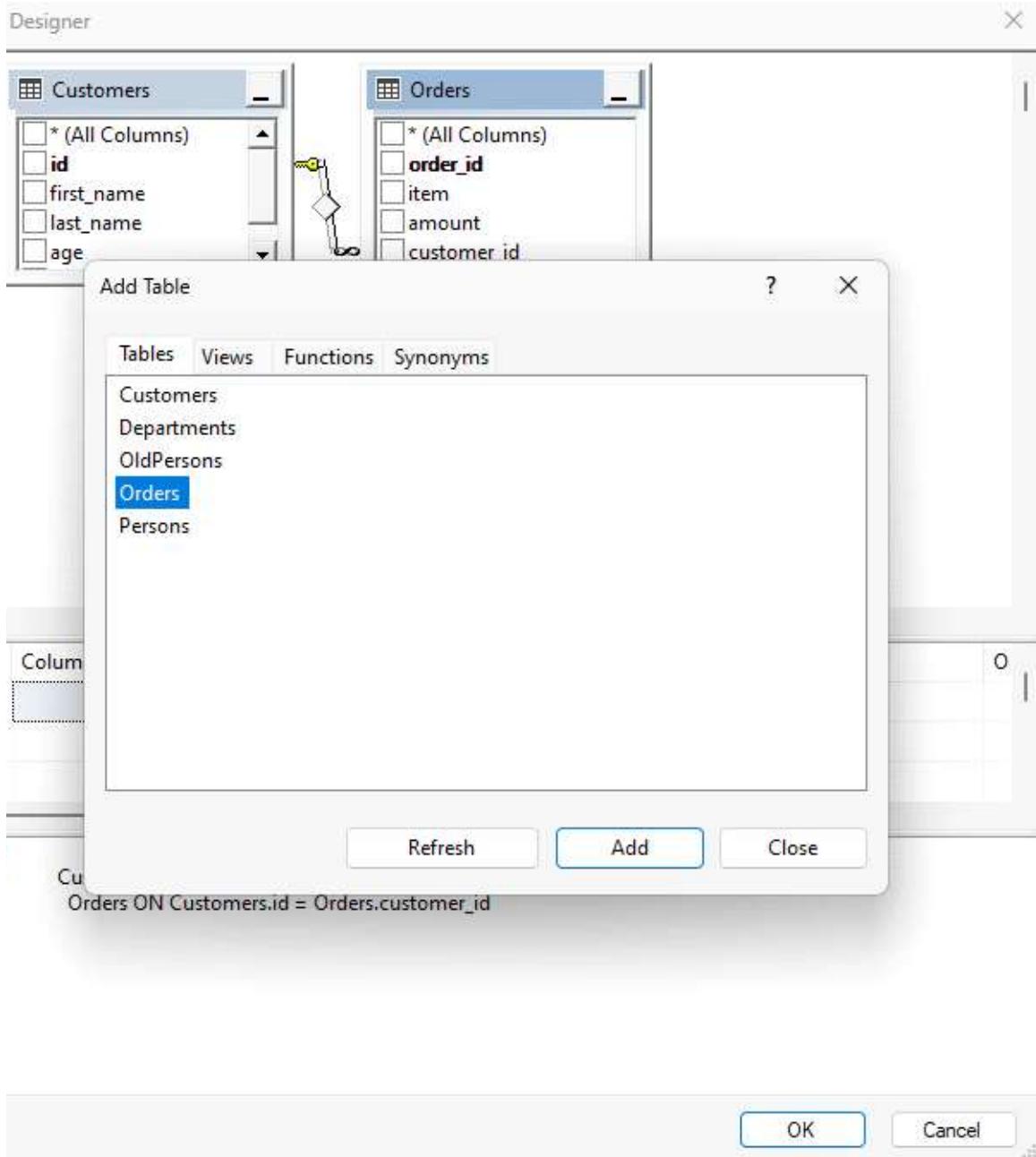
Commands completed successfully.

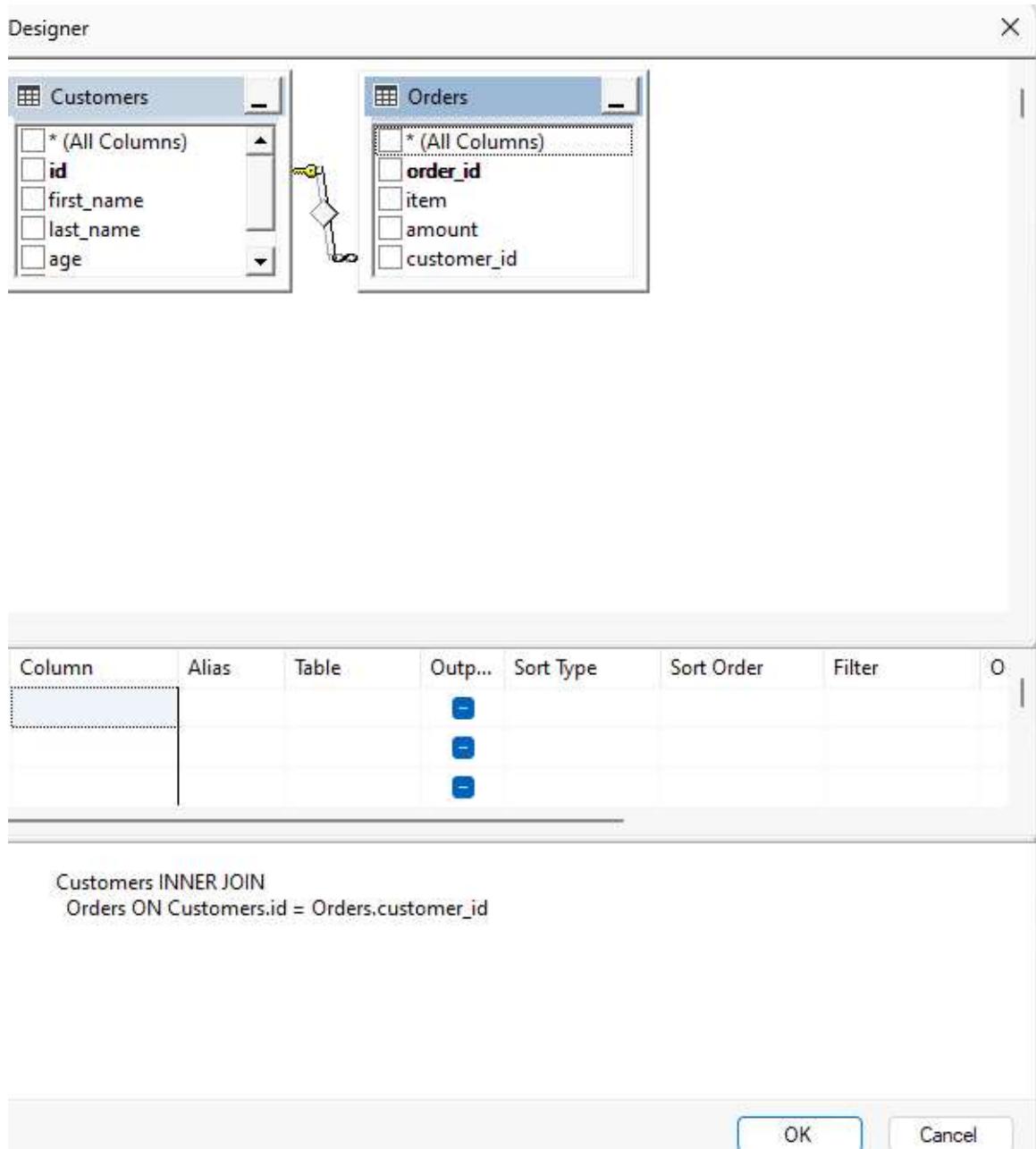
Completion time: 2023-08-26T13:25:44.7124514+03:00

Cut Ctrl+X  
Copy Ctrl+C  
Paste Ctrl+V  
Insert Snippet... Ctrl+K, Ctrl+X  
Surround With... Ctrl+K, Ctrl+S  
Connection  
Open Server in Object Explorer Alt+F8  
Execute F5  
Display Estimated Execution Plan Ctrl+L  
IntelliSense Enabled Ctrl+B, Ctrl+I  
Trace Query in SQL Server Profiler Ctrl+Alt+P  
Analyze Query in Database Engine Tuning Advisor  
Design Query in Editor... Ctrl+Shift+Q  
Include Actual Execution Plan Ctrl+M  
Include Live Query Statistics  
Include Client Statistics Shift+Alt+S  
Results To  
Properties Window F4  
Query Options...



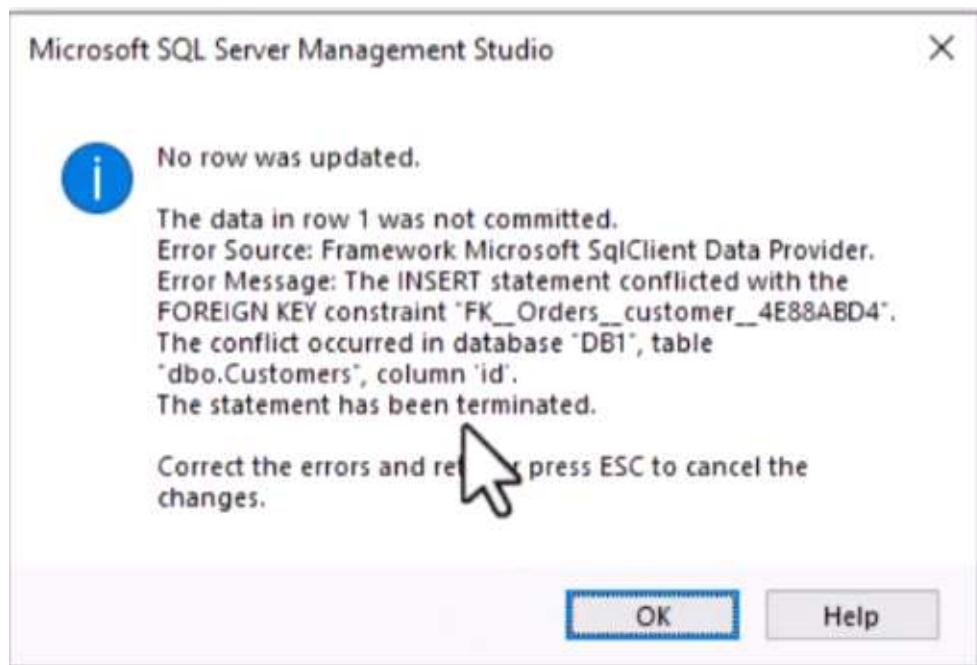
نقرتين بالماوس على اسم الجدول هيظهر لك ورا



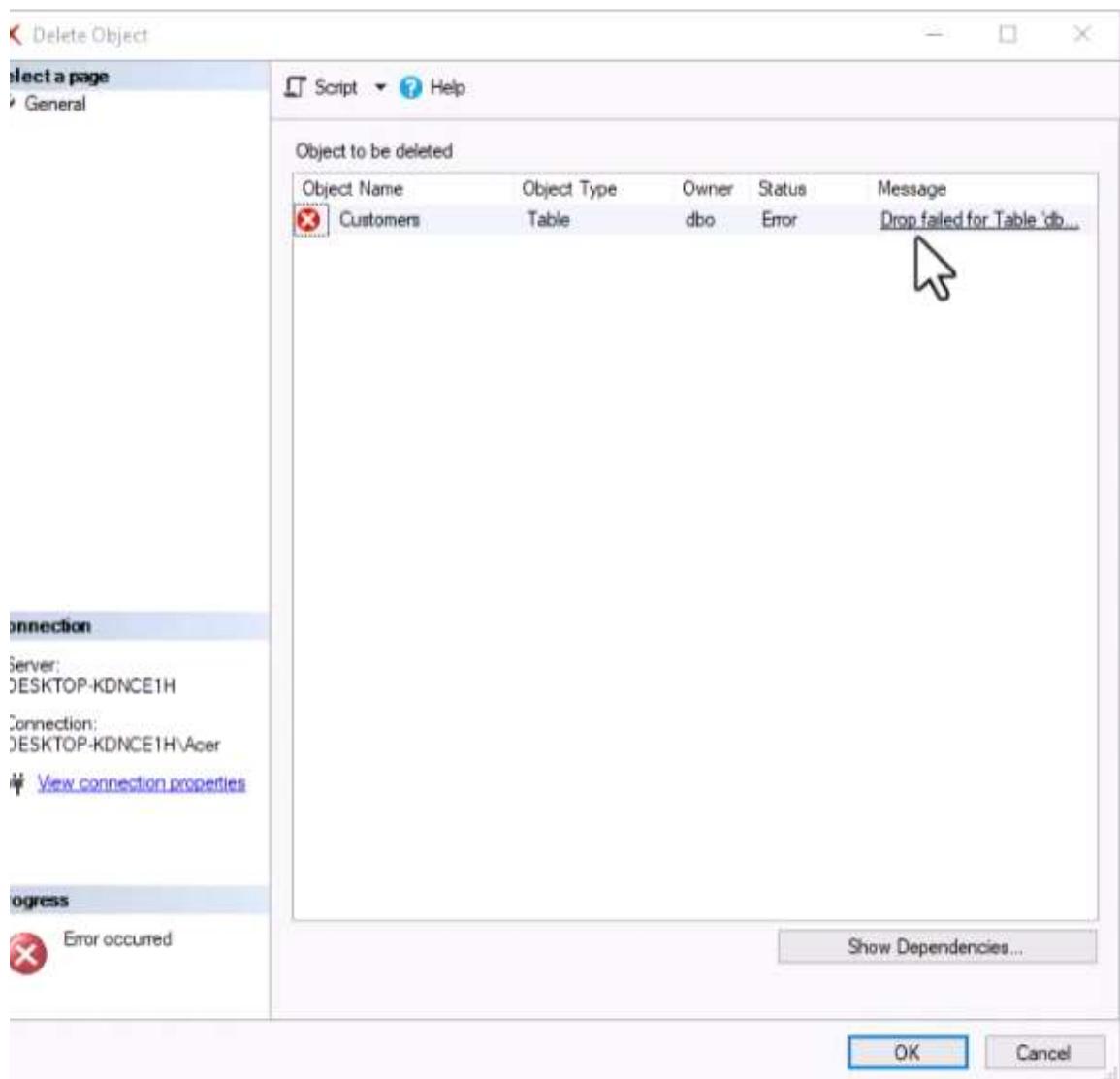


لوجينا في جدول ال orders ونزوود اوردر وجينا نحط في ال customer id رقم مش موجود في جدول ال customers هيططلعلي خطأ

P-KDNCE1H.DB1 - dbo.Orders		SQLQuery5.sql - D...KDNCE1H\Acer (6)	
order_id	item	amount	customer_id
1	! koko	! 10	! 10 !
NULL	NULL	NULL	NULL



لو جيت تحدف جدول ال customers هيطلعلك error لانه مرتبط بجدول تاني  
 لازم احذف جدول ال orders الأول عشان اقدر احذف جدول ال customers



لو جيت تعمل الجدولين من غير ماتضيف السطر ده هيتعملوا مش هيقولك لا بس كل جدول مش هيكون ليه علاقه بالجدول الثاني هيكون كل واحد لوحده

,(id)Customers REFERENCES INT

طيب عشان اضيف ال foreign reference هعمل alter table واقوله يضيف key واكتب اسم العمود اللي عايزة اعمله ال fk وبعدها بكتب السطر بتاع ال references عادي

```
alter table Orders  
add FOREIGN KEY(order_id) references Customers(ID)
```

طيب هنحذف الجدولين ونعملهم تاني من غير مانربطهم

```

DROP TABLE Orders;
DROP TABLE Customers;

-- This table doesn't have a foreign key
CREATE TABLE Customers (
    id INT ,
    first_name VARCHAR(40),
    last_name VARCHAR(40),
    age INT,
    country VARCHAR(10),
    PRIMARY KEY (id)
);

-- Adding foreign key to the customer_id field
-- The foreign key references to the id field of the Customers table
CREATE TABLE Orders (
    order_id INT,
    item VARCHAR(40),
    amount INT,
    customer_id INT ,
    PRIMARY KEY (order_id)
);

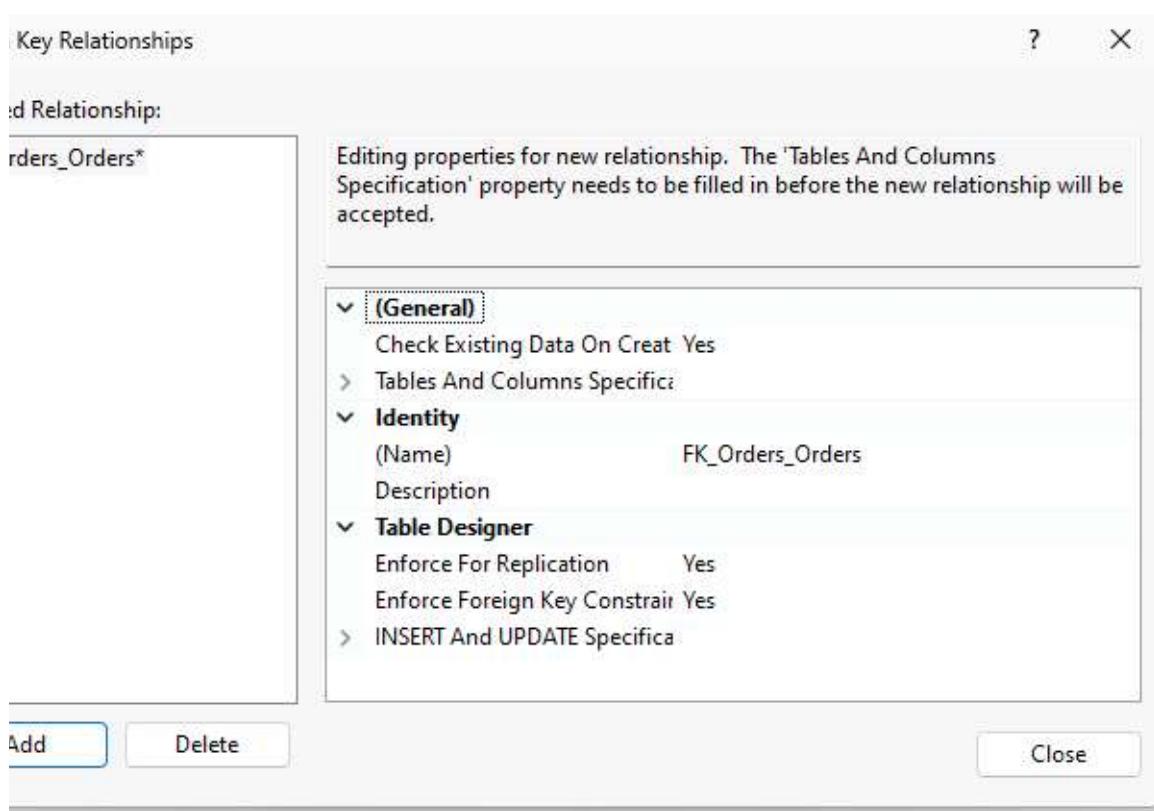
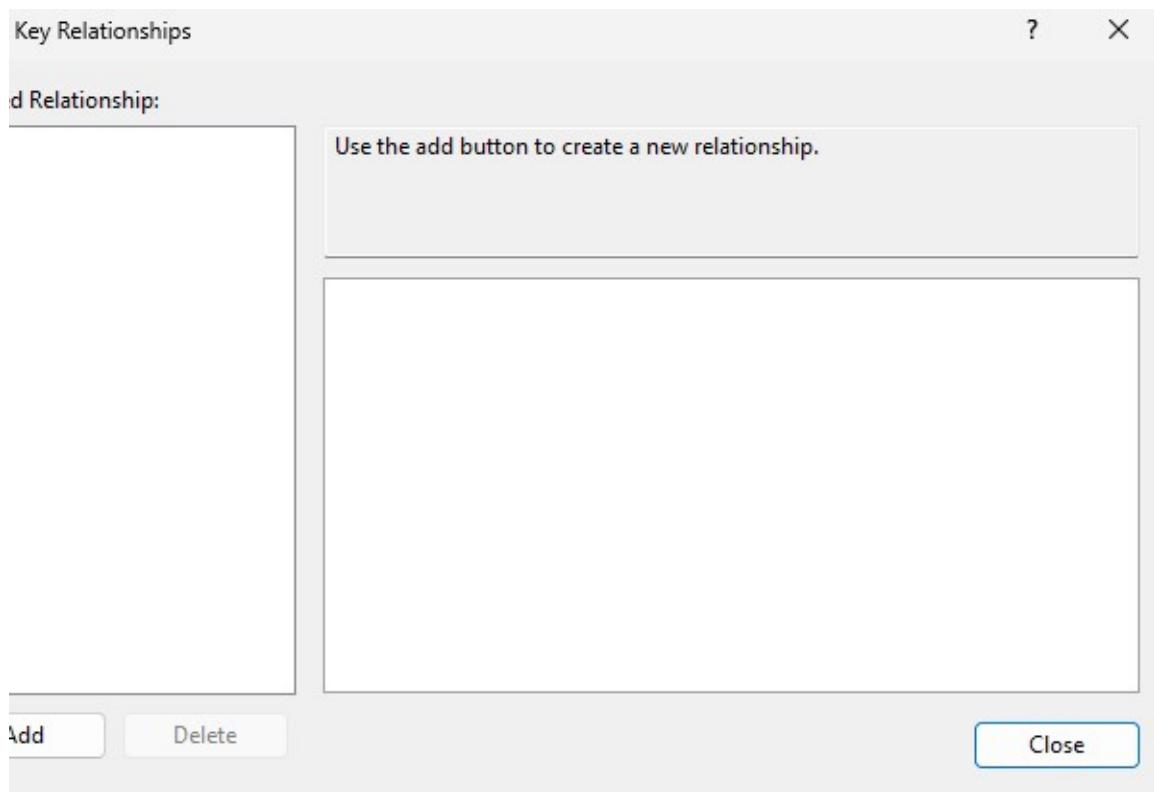
```

عاوزين نضيف ال FOREIGN KEY عن طريق ال DESIGN

Column Name	Data Type	Allow Nulls
order_id	int	<input type="checkbox"/>
item	varchar(40)	<input checked="" type="checkbox"/>
amount	int	<input checked="" type="checkbox"/>
customer_id	int	<input checked="" type="checkbox"/>

The 'customer\_id' column has a context menu open with the following options:

- Set Primary Key
- Insert Column
- Delete Column
- Relationships...**

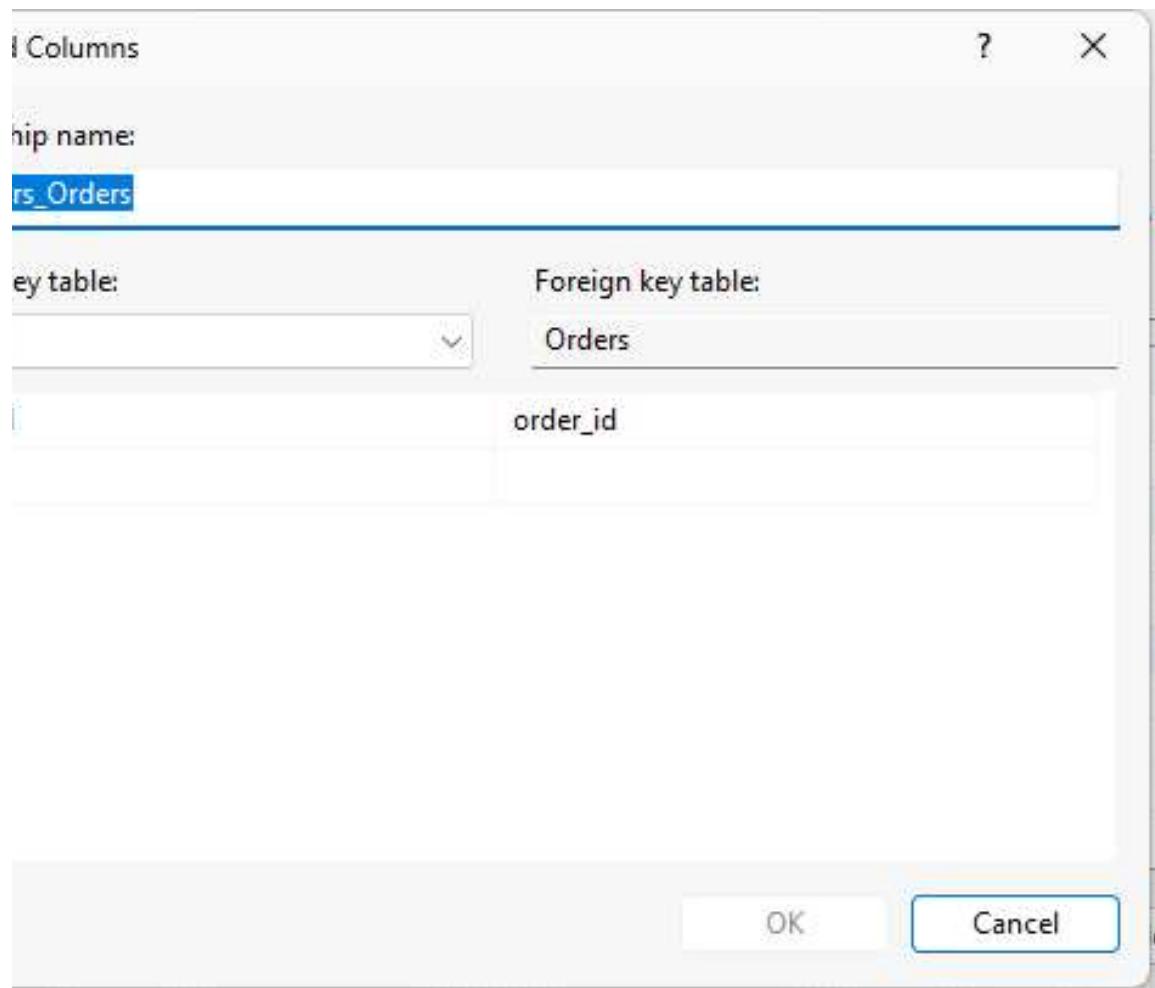


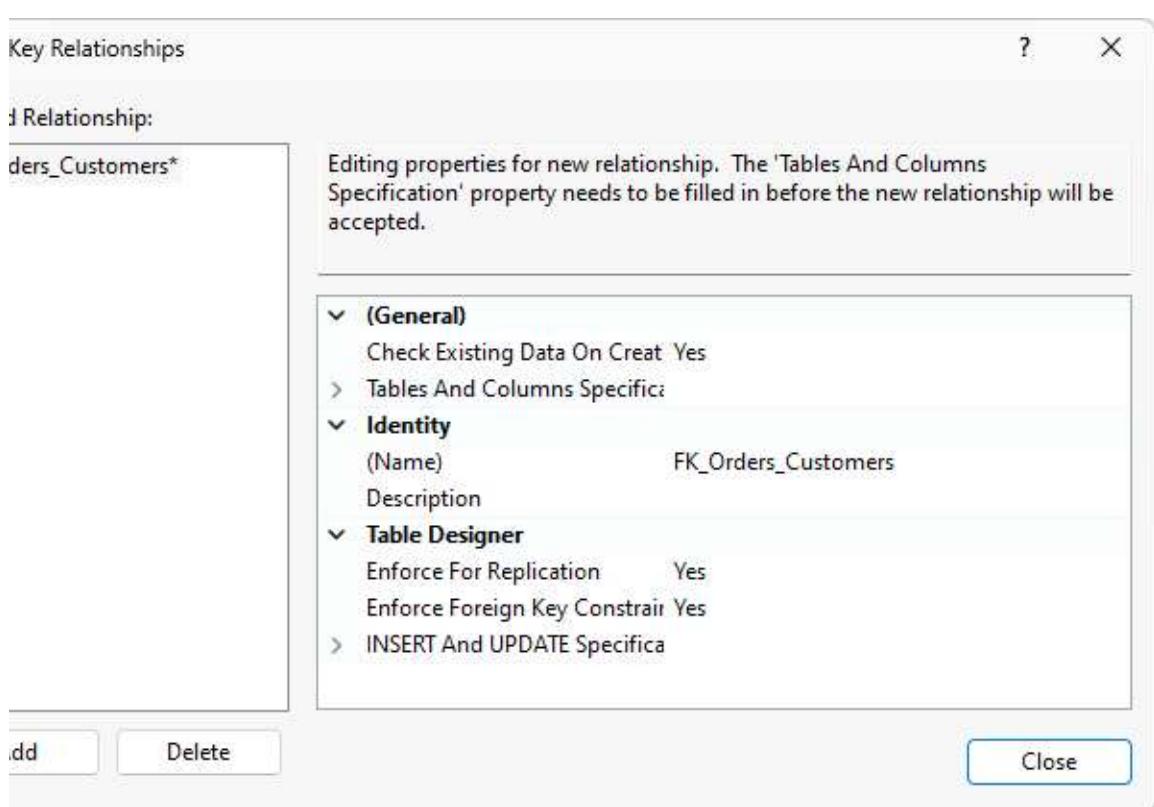
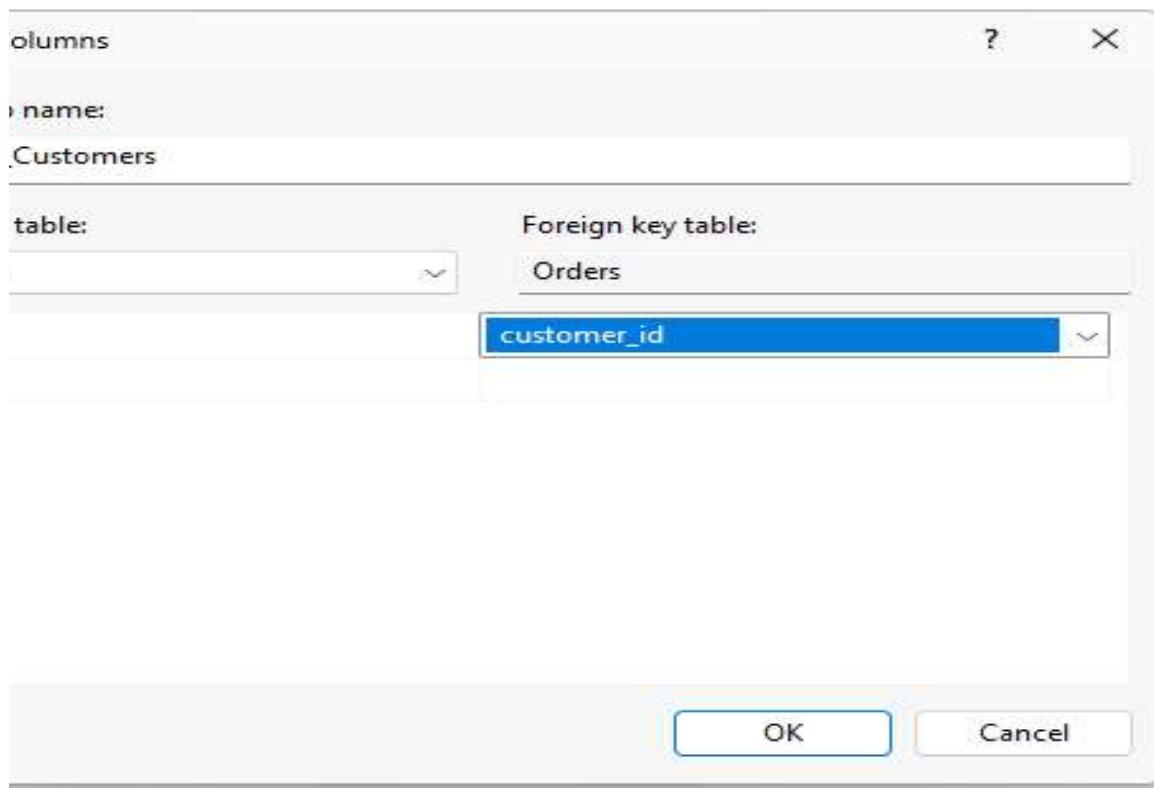
بحدد الجدول بتاع ال KEY PRIMARY

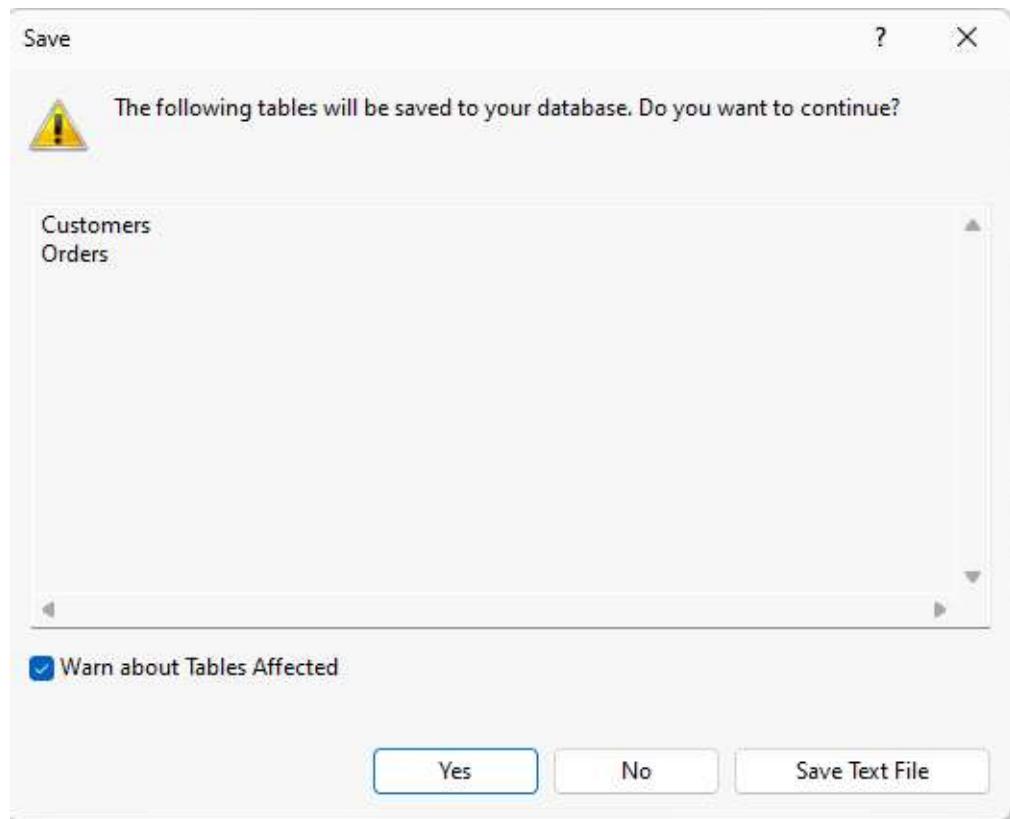
بحدد العمود بتاع ال PRIMARY KEY

بحدد العمود بتاع ال KEY FOREIGN

بحدد الجدول بتاع ال KEY FOREIGN







## SQL FOREIGN KEY

In this tutorial, we'll learn about the FOREIGN KEY in SQL and how to use them with the help of examples.

In SQL, we can create a relationship between two tables using the **FOREIGN KEY** constraint.

Table: Orders

Foreign Key

order_id	product	total	customer_id
1	Paper	500	5
2	Pen	10	2
3	Marker	120	3
4	Books	1000	1
5	Erasers	20	4

Table: Customers

id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

order\_id field in the Orders table is **FOREIGN KEY** which references the id field in the Customers table.

The value of the customer\_id (of the Orders table) must be a value from the id column (of the Customers table).

A key can be referenced to any column in the parent table. However, it is general practice to reference the foreign key column of the parent table.

## FOREIGN Key

Now we can create foreign key constraints in a database.

```
Table doesn't have a foreign key
CREATE TABLE Customers (
    name VARCHAR(40),
    address VARCHAR(40),
    phone VARCHAR(10),
    PRIMARY KEY (id)

Create a foreign key to the customer_id field
which key references to the id field of the Customers table
CREATE TABLE Orders (
    order_id INT,
    customer_id VARCHAR(40),
    quantity INT,
    PRIMARY KEY (order_id),
    FOREIGN KEY (customer_id)
        REFERENCES Customers(id),
```

the customer\_id column in the Orders table references the row in another table named Customers with

code works in all major database systems. However, there may be the alternate syntax to create foreign keys database. Refer to their respective database documentation for more information.

## with Alter Table

Add the FOREIGN KEY constraint to an existing table using the ALTER TABLE command. For example,

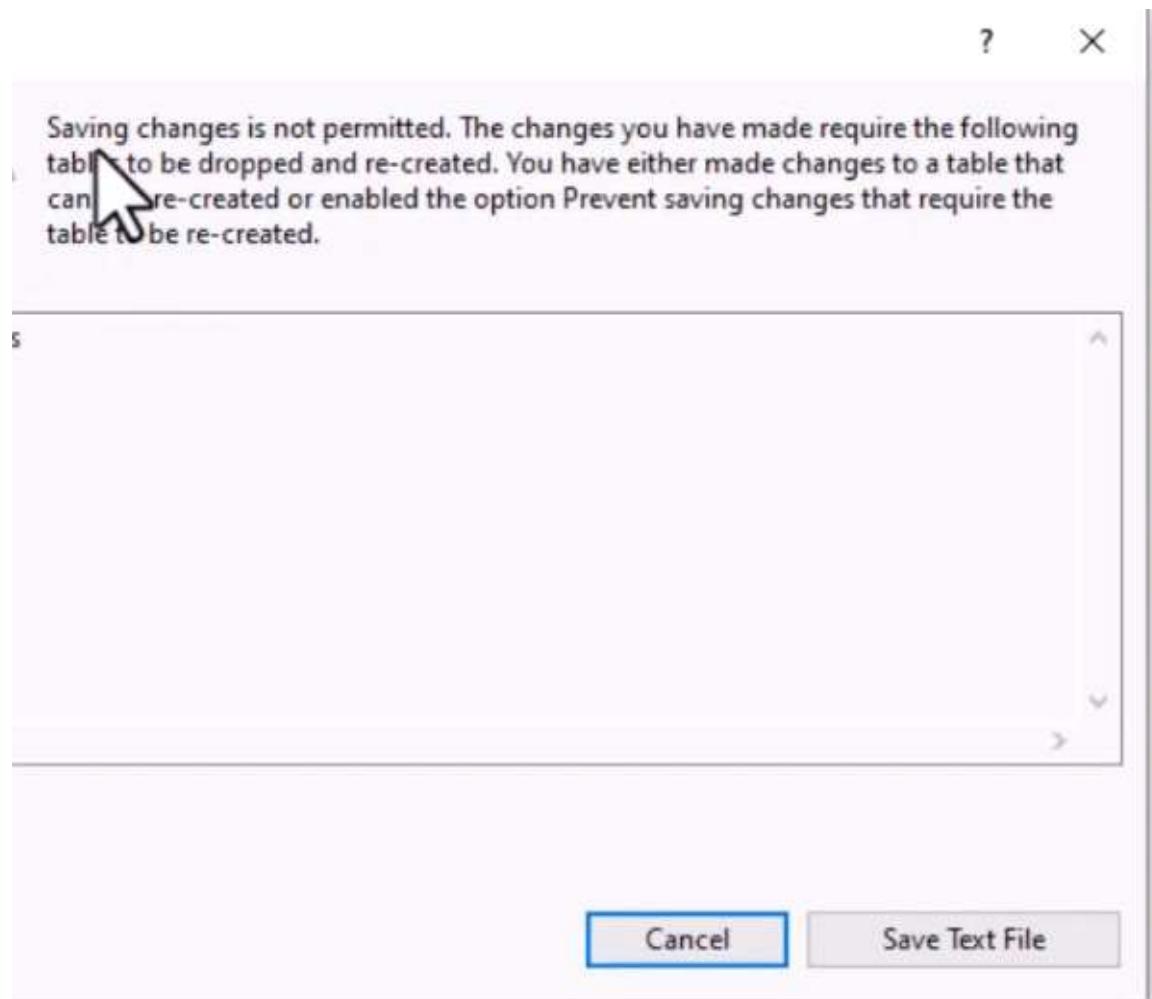
```
| doesn't have a foreign key
Customers (
| VARCHAR(40),
| VARCHAR(40),
| CHAR(10),
| (id)

Orders (
| INT,
| R(40),
| INT ,
| (order_id)

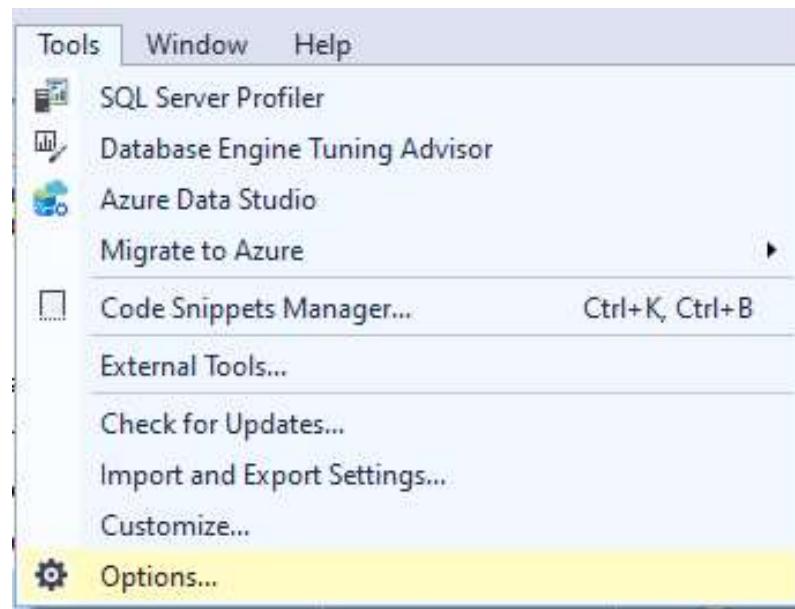
| eign key to the customer_id field using alter
| orders
| KEY (customer_id) REFERENCES Customers(id);
```

## Solution To: "Saving changes is not permitted" error

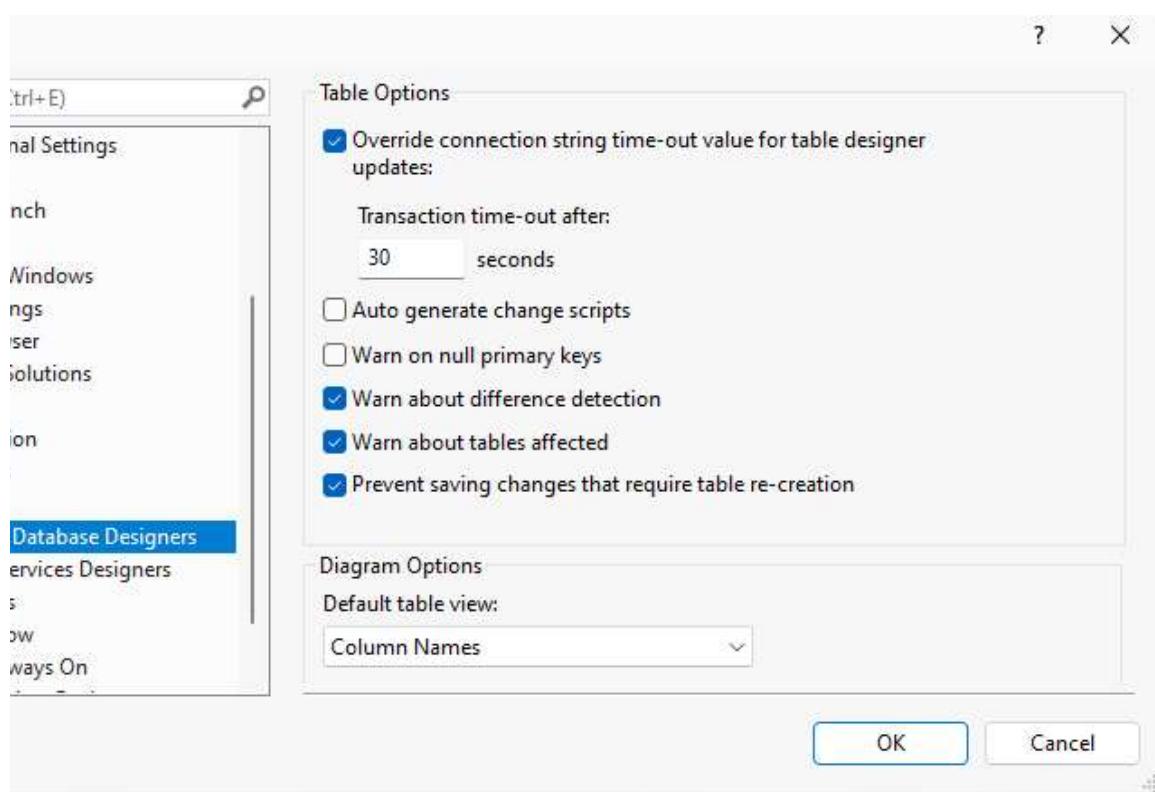
هنا بيقولك انه فيه ناس لما تيجي تعديل على جدول وتحفظه تطلعلك الرساله دي



بيقولك انه ده بيحصل لانه فيه اوبشن بيخليلك لما تيجي تعدل علي عمود او تضيف عمود جديد بيحذف الجدول ويعمل جدول جديد مكانه بنفس المواصفات عشان تصاحها بتعمل كده



وبعدين بتشيل ال check من هنا



## : "Saving changes is not permitted" error:

the "Saving changes is not permitted" error in SQL Server when attempting to modify a table that requires re-change a setting in SQL Server Management Studio (SSMS) to allow saving changes that require table re-creation. an do it:

. Server Management Studio.

"Tools" menu and select "Options."

tions window, navigate to "Designers" > "Table and Database Designers."

the option "Prevent saving changes that require table re-creation."

" to save the changes.

s configuration change, you should be able to modify and save changes to your tables without encountering the is not permitted" error. However, keep in mind that making significant changes to a table's structure can have existing data and may require careful consideration and backup procedures to avoid data loss or inconsistencies. when making structural changes to production databases.

## SQL - Queries

### Restore Sample HR Database and Get Ready .

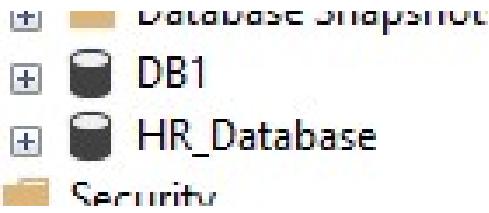
هتنزل الداتا بيز دي

<https://cdn.fs.teachablecdn.com/vB7M2lSPQ7WwpJrk8Ggi>

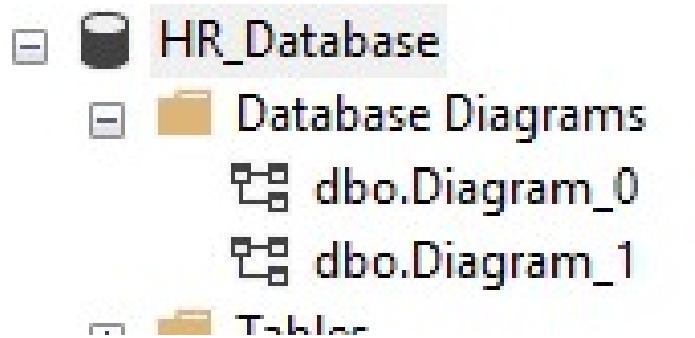
وتحطها عندك عال server sql

```
RESTORE DATABASE HR_Database  
FROM DISK='D:\HR_Database.bak';
```

وبقولك لو حصل خطأ حط السطر ده



وبعدين هنروح نبص عال DIAGRAM



لو مالقيتش الملفين دول وظهرلك رسالة خطأ

هتكتب السطرين دول في ال QUERY وتشغلهم

```
use HR_Database;
EXEC sp_changedbowner 'sa';
```

### Select Statement

هنبدأ في ال DQL اللي هيا عباره عن ال select statement معناها استعلام Query

زي ما عرفنا قبل كده انه الجمله دي بترجعلي كل حاجه من الجدول

```
USE HR_Database;
SELECT * FROM Countries;
SELECT * FROM Departments;
SELECT * FROM Employees;
```

فيه طريقة تانية اكتب فيها ال QUERY دي وهي اني اذكر اسم الجدول قبل علامة النجمة

```

SELECT Countries.* FROM Countries;
SELECT Departments.* FROM Departments;
SELECT Employees.* FROM Employees;

```

The screenshot shows three tables in Microsoft Access:

- Departments** table (left):
 

	ID	Name
1	1	Engineering
2	2	Accounting
3	3	Marketing
4	4	IT
5	5	HR
6	6	Finance
7	7	Sales
- Countries** table (middle):
 

	ID	Name
1	1	USA
2	2	UK
3	3	China
- Employees** table (right):
 

Last Name	Gender	Date of Birth	Country ID	Department ID	Hire Date	Exit Date	Monthly Salary	Bonus Percent
Le	M	1972-09-20 00:00:00	1	1	2022-02-05 00:00:00	NULL	709.00	0
Patel	M	1981-06-10 00:00:00	1	7	2013-10-23 00:00:00	NULL	2708.00	0
Lo	M	1994-05-26 00:00:00	2	4	2019-03-24 00:00:00	NULL	372.00	0
Castillo	F	1996-01-21 00:00:00	1	4	2018-04-07 00:00:00	NULL	922.00	0
Dominguez	F	1964-12-19 00:00:00	1	1	2005-06-18 00:00:00	NULL	535.00	0.24
Vu	M	1994-02-12 00:00:00	1	4	2004-04-22 00:00:00	2014-02-14 00:00:00	2701.00	0
Hu	F	1997-04-17 00:00:00	2	2	2009-06-27 00:00:00	NULL	395.00	0
Chang	M	1987-06-14 00:00:00	2	6	1999-02-19 00:00:00	NULL	1415.00	0
Holmes	F	1994-09-10 00:00:00	1	4	2011-09-09 00:00:00	NULL	694.00	0
Thomas	M	1962-12-23 00:00:00	1	6	2015-02-05 00:00:00	NULL	2377.00	0.1
Pena	M	1979-01-07 00:00:00	1	4	2003-10-12 00:00:00	NULL	1258.00	0
Wu	F	1995-02-09 00:00:00	1	6	2014-08-03 00:00:00	NULL	794.00	0
Wong	M	1990-07-03 00:00:00	2	4	2017-11-15 00:00:00	NULL	2374.00	0.23
Richardson	M	1983-10-04 00:00:00	1	3	2018-07-22 00:00:00	NULL	1751.00	0.08
Moore	M	1977-12-29 00:00:00	1	3	2021-03-24 00:00:00	NULL	659.00	0.15
Lu	F	1998-01-15 00:00:00	1	4	1997-07-26 00:00:00	NULL	1929.00	0
Tran	F	1967-06-01 00:00:00	2	1	2010-08-05 00:00:00	NULL	2555.00	0.33
Chau	F	1988-01-04 00:00:00	2	7	2019-03-03 00:00:00	NULL	465.00	0

The screenshot shows the **Employees** table in Microsoft Access:

Last Name	Gender	Date of Birth	Country ID	Department ID	Hire Date	Exit Date	Monthly Salary	Bonus Percent
Le	M	1972-09-20 00:00:00	1	1	2022-02-05 00:00:00	NULL	709.00	0
Patel	M	1981-06-10 00:00:00	1	7	2013-10-23 00:00:00	NULL	2708.00	0
Lo	M	1994-05-26 00:00:00	2	4	2019-03-24 00:00:00	NULL	372.00	0
Castillo	F	1996-01-21 00:00:00	1	4	2018-04-07 00:00:00	NULL	922.00	0
Dominguez	F	1964-12-19 00:00:00	1	1	2005-06-18 00:00:00	NULL	535.00	0.24
Vu	M	1994-02-12 00:00:00	1	4	2004-04-22 00:00:00	2014-02-14 00:00:00	2701.00	0
Hu	F	1997-04-17 00:00:00	2	2	2009-06-27 00:00:00	NULL	395.00	0
Chang	M	1987-06-14 00:00:00	2	6	1999-02-19 00:00:00	NULL	1415.00	0
Holmes	F	1994-09-10 00:00:00	1	4	2011-09-09 00:00:00	NULL	694.00	0
Thomas	M	1962-12-23 00:00:00	1	6	2015-02-05 00:00:00	NULL	2377.00	0.1
Pena	M	1979-01-07 00:00:00	1	4	2003-10-12 00:00:00	NULL	1258.00	0
Wu	F	1995-02-09 00:00:00	1	6	2014-08-03 00:00:00	NULL	794.00	0
Wong	M	1990-07-03 00:00:00	2	4	2017-11-15 00:00:00	NULL	2374.00	0.23
Richardson	M	1983-10-04 00:00:00	1	3	2018-07-22 00:00:00	NULL	1751.00	0.08
Moore	M	1977-12-29 00:00:00	1	3	2021-03-24 00:00:00	NULL	659.00	0.15
Lu	F	1998-01-15 00:00:00	1	4	1997-07-26 00:00:00	NULL	1929.00	0
Tran	F	1967-06-01 00:00:00	2	1	2010-08-05 00:00:00	NULL	2555.00	0.33
Chau	F	1988-01-04 00:00:00	2	7	2019-03-03 00:00:00	NULL	465.00	0

طيب انا جدول الموظفين مشش عايز منه كل الداتا دي عايز اعمده معينه زي ال ID والاسم والمرتب

قالك كل اللي هتعمله انك هتكتب اسم العمود او الاعمدة اللي عايزها بدل علامة النجمة

يعني علامة النجمة وظيفتها انها تختار كل الاعمده والشروط اللي كنا بنعملها بال  
RECORDS دي بتتحكم في عدد ال WHERE

```
SELECT ID,FirstName,LastName,MonthlySalary FROM Employees;
```

طيب لو عايزين نرجع عمود تاريخ الميلاد والاسم الأول

```
SELECT ID,FirstName,DateOfBirth FROM Employees;
```

## LECT Statement

Statement is used to select data from a database.

is stored in a result table, called the result-set.

tax

*column2, ...*

es:

column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in following syntax:

*table\_name;*

```
n Employees;
ees.* from Employees;
irstName, LastName,MonthlySalary From Employees;
irstName, DateOfBirth From Employees;
n Departments;
n Countries;
```

## Select Distinct Statement

كلمة DISTINCT بترجم الداتا بدون تكرار

لو عايز مثلا اني اعرف ايه الأقسام اللي موجود فيها موظفين عندي وجيit اعملها بامر  
هترجع مكرره انما لو زودت كلمة distinct departmentid هترجع من  
غير تكرار

```
SELECT DepartmentID FROM Employees;
```

```
SELECT DISTINCT DepartmentID FROM Employees;
```

	DepartmentID	Results	Messages
1	3	1	1
2	6	2	7
3	7	3	4
4	1	4	4
5	4	5	1
6	5	6	4
7	2	7	2
		8	6
		9	4
		10	6
		11	4
		12	6
		13	4
		14	3
		15	3
		16	4
		17	1
		18	7

اقدر اعمل ده علي أي عمود تاني زي ال FIRST NAME مثل

```
SELECT FirstName FROM Employees;
```

```
SELECT DISTINCT FirstName FROM Employees;
```

	First Name
195	Skylar
196	Sofia
197	Sophia
198	Sophie
199	Stella
200	Theodore
201	Thomas
202	Valentina
203	Victoria
204	Violet
205	Vivian
206	Wesley
207	William
208	Willow
209	Wyatt
210	Xavier
211	Zoe
212	Zoey

	First Name
983	Alexander
984	Logan
985	Henry
986	Delilah
987	Caroline
988	Jack
989	Luna
990	John
991	Charlotte
992	Miles
993	Violet
994	Isaac
995	Ian
996	Melody
997	Eliza
998	Layla
999	Thomas
1000	Willow

لو جيت اعملها علي عمودين هي عمل ايه ؟

اعتبر ان النتيجه اللي طالعه من العمودين هما record واحد علي بعضه عادي بغض النظر عن كام عمود في الجدول اللي انا جايب منه الداتا وهيشوف هل ال record ده كله علي بعضه متكرر في الجدول اللي طلع ولا لا لو متكرر هيشيل التكرار ولو مش متكرر هيسبيه

طيب لو فيه قيم في العمود الأول او العمود الثاني متكرره هيشيلهم ؟

ما خلاص بقى احنا قولنا انه هيتعامل مع القيم اللي في الجدولين كانهم record واحد علي بعضه وبيشوف هل ال record ده نفسه متكرر ولا لا

عاوز اشوف الأسماء اللي في كل قسم مثلاً من بسمح بالتكرار ومره لا هلاقي فيه قسم فيه واحد اسمه احمد مثلاً هل فيه حد تاني في نفس القسم اسمه احمد؟ لو فيه هيشيله

```
SELECT FirstName,DepartmentID FROM Employees;
```

```
SELECT DISTINCT FirstName,DepartmentID FROM Employees;
```

	FirstName	DepartmentID		FirstName	DepartmentID	
696	Wesley	7		983	Alexander	7
697	William	4		984	Logan	5
698	William	5		985	Henry	7
699	Willow	1		986	Delilah	4
700	Willow	2		987	Caroline	6
701	Willow	3		988	Jack	2
702	Willow	4		989	Luna	4
703	Willow	5		990	John	1
704	Willow	7		991	Charlotte	7
705	Wyatt	1		992	Miles	7
706	Wyatt	4		993	Violet	3
707	Wyatt	6		994	Isaac	6
708	Wyatt	7		995	Ian	1
709	Xavier	3		996	Melody	4
710	Zoe	2		997	Eliza	6
711	Zoey	1		998	Layla	7
712	Zoey	3		999	Thomas	1
713	Zoey	4		1000	Willow	7

```
USE HR_Database;
```

```
SELECT DepartmentID FROM Employees;
```

```
SELECT DISTINCT DepartmentID FROM Employees;
```

```
SELECT FirstName FROM Employees;
```

```
SELECT DISTINCT FirstName FROM Employees;
```

```
SELECT FirstName,DepartmentID FROM Employees;
```

```
SELECT DISTINCT FirstName,DepartmentID FROM Employees;
```

## **SELECT DISTINCT Statement**

**DISTINCT** statement is used to return only distinct (different) values.

Column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

### **DISTINCT Syntax**

`SELECT column1, column2, ...`

`ie;`

```
SELECT DepartmentID FROM Employees;  
SELECT DepartmentID FROM Employees;  
  
Name FROM Employees;  
SELECT FirstName FROM Employees;  
  
Name, DepartmentID FROM Employees;  
SELECT FirstName, DepartmentID FROM Employees;
```

## **Where Statement + AND , OR, NOT**

هنا بيتكلم عن اضافه شروط لأمر ال select

مثلا عايز اجيب كل الموظفين الاناث

```
USE HR_Database;
```

```
SELECT * FROM Employees  
WHERE Gender='F';
```

	ID	FirstName	LastName	Gendor	DateOfBirth	CountryID	DepartmentID	HireDate	ExitDate	MonthlySalary	BonusPerc
1	288	Harper	Castillo	F	1996-01-21 00:00:00	1	4	2018-04-07 00:00:00	NULL	922.00	0
2	289	Harper	Dominguez	F	1964-12-19 00:00:00	1	1	2005-06-18 00:00:00	NULL	535.00	0.24
3	291	Jade	Hu	F	1997-04-17 00:00:00	2	2	2009-06-27 00:00:00	NULL	395.00	0
4	293	Gianna	Holmes	F	1994-09-10 00:00:00	1	4	2011-09-09 00:00:00	NULL	694.00	0
5	296	Bella	Wu	F	1995-02-09 00:00:00	1	6	2014-08-03 00:00:00	NULL	794.00	0
6	300	Luna	Lu	F	1998-01-15 00:00:00	1	4	1997-07-26 00:00:00	NULL	1929.00	0
7	301	Bella	Tran	F	1967-06-01 00:00:00	2	1	2010-08-05 00:00:00	NULL	2555.00	0.33
8	302	Ivy	Chau	F	1988-01-04 00:00:00	2	7	2019-03-03 00:00:00	NULL	465.00	0
9	304	Sophia	Gutierrez	F	1971-09-28 00:00:00	1	2	2009-02-08 00:00:00	NULL	2081.00	0.06
10	306	Lillian	Lewis	F	1989-01-05 00:00:00	1	4	2013-08-14 00:00:00	2019-03-31 00:00:00	2790.00	0
11	307	Serenity	Cao	F	1992-07-08 00:00:00	2	7	2018-10-21 00:00:00	NULL	2684.00	0
12	312	Ivy	Thompson	F	1964-04-15 00:00:00	1	3	2004-08-11 00:00:00	NULL	2811.00	0.05
13	313	Peyton	Wright	F	1960-02-09 00:00:00	1	3	2017-05-13 00:00:00	NULL	2066.00	0.1
14	315	Ruby	Alexander	F	1994-03-07 00:00:00	1	6	2001-08-13 00:00:00	NULL	2982.00	0.36
15	318	Liliana	Chang	F	1978-08-19 00:00:00	2	4	2018-01-11 00:00:00	NULL	2043.00	0
16	321	Amelia	Dominguez	F	1973-08-17 00:00:00	3	2	2015-09-23 00:00:00	2018-07-27 00:00:00	1229.00	0.15
17	324	Chloe	Chin	F	1983-01-07 00:00:00	1	6	2021-11-13 00:00:00	NULL	2872.00	0.1
18	325	Ella	Martinez	F	1988-06-19 00:00:00	1	6	2012-04-06 00:00:00	NULL	1465.00	0

عايزين الموظفين اللي مرتباتهم اقل من 500

```
SELECT * FROM Employees
WHERE MonthlySalary<500;
```

LastName	Gendor	DateOfBirth	CountryID	DepartmentID	HireDate	ExitDate	MonthlySalary	BonusPerc
Lo	M	1994-05-26 00:00:00	2	4	2019-03-24 00:00:00	NULL	372.00	0
Hu	F	1997-04-17 00:00:00	2	2	2009-06-27 00:00:00	NULL	395.00	0
Chau	F	1988-01-04 00:00:00	2	7	2019-03-03 00:00:00	NULL	465.00	0
Jones	F	1965-12-26 00:00:00	1	2	1998-09-26 00:00:00	2016-11-02 00:00:00	362.00	0.09
Cheng	M	1994-07-09 00:00:00	1	7	1999-11-26 00:00:00	2003-11-27 00:00:00	330.00	0.17
Patel	M	1966-07-20 00:00:00	1	7	2022-09-13 00:00:00	NULL	213.00	0
Grant	F	1982-02-10 00:00:00	1	4	2007-02-16 00:00:00	NULL	453.00	0
Lai	M	1986-03-13 00:00:00	1	1	2019-09-22 00:00:00	NULL	336.00	0.12
Singh	F	1960-08-22 00:00:00	1	1	2022-03-16 00:00:00	NULL	481.00	0.26
Young	M	1961-07-27 00:00:00	1	4	2010-08-16 00:00:00	NULL	459.00	0.33
Hong	F	1981-09-19 00:00:00	2	4	2021-01-09 00:00:00	NULL	383.00	0
Ramos	M	1997-01-22 00:00:00	3	7	2018-07-28 00:00:00	NULL	235.00	0
Cheng	F	1971-01-27 00:00:00	2	6	1996-08-20 00:00:00	NULL	395.00	0
Espinosa	F	1980-06-22 00:00:00	3	2	2002-09-12 00:00:00	NULL	396.00	0
King	F	1962-12-31 00:00:00	1	3	2013-08-10 00:00:00	NULL	404.00	0.15
Jones	M	1980-10-14 00:00:00	1	3	1996-07-10 00:00:00	NULL	264.00	0.08
Lo	F	1973-06-18 00:00:00	1	4	2020-08-28 00:00:00	NULL	367.00	0
Jiang	F	1972-10-26 00:00:00	1	2	2015-06-19 00:00:00	NULL	389.00	0.38

لو عايز اجيب الموظفين اللي مرتباتهم اكبر من 500

هنا بيقولك ممكن تستخدم NOT وهيا بتنفي الشرط

```
SELECT * FROM Employees
WHERE MonthlySalary>500;
SELECT * FROM Employees
WHERE NOT MonthlySalary<=500;
```

LastName	Gendor	DateOfBirth	CountryID	DepartmentID	HireDate	ExitDate	MonthlySalary	BonusPerc
Le	M	1972-09-20 00:00:00	1	1	2022-02-05 00:00:00	NULL	709.00	0
Patel	M	1981-06-10 00:00:00	1	7	2013-10-23 00:00:00	NULL	2708.00	0
Castillo	F	1996-01-21 00:00:00	1	4	2018-04-07 00:00:00	NULL	922.00	0
Dominguez	F	1964-12-19 00:00:00	1	1	2005-06-18 00:00:00	NULL	535.00	0.24
Vu	M	1994-02-12 00:00:00	1	4	2004-04-22 00:00:00	2014-02-14 00:00:00	2701.00	0
Chang	M	1987-06-14 00:00:00	2	6	1999-02-19 00:00:00	NULL	1415.00	0
Holmes	F	1994-09-10 00:00:00	1	4	2011-09-09 00:00:00	NULL	694.00	0
Thomas	M	1962-12-23 00:00:00	1	6	2015-02-05 00:00:00	NULL	2377.00	0.1
Pena	M	1979-01-07 00:00:00	1	4	2003-10-12 00:00:00	NULL	1258.00	0
Wu	F	1995-02-09 00:00:00	1	6	2014-08-03 00:00:00	NULL	794.00	0
Wong	M	1990-07-03 00:00:00	2	4	2017-11-15 00:00:00	NULL	2374.00	0.23
Richardson	M	1983-10-04 00:00:00	1	3	2018-07-22 00:00:00	NULL	1751.00	0.08
Moore	M	1977-12-29 00:00:00	1	3	2021-03-24 00:00:00	NULL	659.00	0.15
Lu	F	1998-01-15 00:00:00	1	4	1997-07-26 00:00:00	NULL	1929.00	0
Tran	F	1967-06-01 00:00:00	2	1	2010-08-05 00:00:00	NULL	2555.00	0.33
Kumar	M	1992-10-31 00:00:00	1	4	2017-11-11 00:00:00	NULL	1085.00	0
Gutierrez	F	1971-09-28 00:00:00	1	2	2009-02-08 00:00:00	NULL	2081.00	0.06
Dang	M	1992-11-08 00:00:00	1	2	2015-11-16 00:00:00	NULL	654.00	0.12

طيب عاوزين كل الموظفين الاناث اللي مرتباتهم اقل من 500 هنا ممكن استخدم  
AND

```
SELECT * FROM Employees
WHERE MonthlySalary<500 AND Gendor='F';
```

LastName	Gendor	DateOfBirth	CountryID	DepartmentID	HireDate	ExitDate	MonthlySalary	BonusPerc
Hu	F	1997-04-17 00:00:00	2	2	2009-06-27 00:00:00	NULL	395.00	0
Chau	F	1988-01-04 00:00:00	2	7	2019-03-03 00:00:00	NULL	465.00	0
Jones	F	1965-12-26 00:00:00	1	2	1998-09-26 00:00:00	2016-11-02 00:00:00	362.00	0.09
Grant	F	1982-02-10 00:00:00	1	4	2007-02-16 00:00:00	NULL	453.00	0
Singh	F	1960-08-22 00:00:00	1	1	2022-03-16 00:00:00	NULL	481.00	0.26
Hong	F	1981-09-19 00:00:00	2	4	2021-01-09 00:00:00	NULL	383.00	0
Cheng	F	1971-01-27 00:00:00	2	6	1996-08-20 00:00:00	NULL	395.00	0
Espinosa	F	1980-06-22 00:00:00	3	2	2002-09-12 00:00:00	NULL	396.00	0
King	F	1962-12-31 00:00:00	1	3	2013-08-10 00:00:00	NULL	404.00	0.15
Lo	F	1973-06-18 00:00:00	1	4	2020-08-28 00:00:00	NULL	367.00	0
Jiang	F	1972-10-26 00:00:00	1	2	2015-06-19 00:00:00	NULL	389.00	0.38
Parker	F	1987-01-21 00:00:00	1	4	2018-08-05 00:00:00	NULL	330.00	0
Nunez	F	1993-09-07 00:00:00	1	6	2017-09-10 00:00:00	NULL	220.00	0.05
Herrera	F	1987-05-30 00:00:00	3	4	2015-10-03 00:00:00	NULL	331.00	0.1
Mendoza	F	1974-02-05 00:00:00	1	7	2017-05-07 00:00:00	NULL	473.00	0
Xu	F	1986-05-13 00:00:00	2	1	1997-10-16 00:00:00	NULL	263.00	0
Williams	F	1971-09-21 00:00:00	1	7	2019-02-12 00:00:00	NULL	316.00	0.36
Lu	F	1965-02-02 00:00:00	2	3	2007-03-06 00:00:00	NULL	201.00	0

عاوز كل الناس اللي من البلد اللي رقمها 1

```
SELECT * FROM Employees
WHERE CountryID =1;
```

عاوز كل الناس اللي مش من البلد اللي رقمها 1

```
SELECT * FROM Employees  
WHERE NOT CountryID =1;  
SELECT * FROM Employees  
WHERE CountryID <>1;
```

عاوز كل الناس اللي من البلد اللي رقمها ب 1 والنوع ذكر

```
SELECT * FROM Employees  
WHERE CountryID =1 AND Gendor='M';
```

عاوز كل الموظفين اللي في القسم رقم 1 ورقم 2 هنا نستخدم OR

```
SELECT * FROM Employees  
WHERE CountryID =1 OR CountryID=2;
```

لو استخدمت AND بدل OR مش هيرجعلك حاجه لانه علاقه الموظفين بالقسم هيا  
علاقه

MANY TO ONE يعني عمرك ماهتلقي موظف شغال في قسمين مختلفين لكن  
العكس ممكن انه القسم فيه اكتر من موظف

عاوز كل الناس اللي لسه شغالين معانا هنسخدم IS NULL

```
SELECT * FROM Employees  
WHERE ExitDate IS NULL;
```

عاوز كل الناس اللي مشيت من الشركه

```
SELECT * FROM Employees  
WHERE NOT ExitDate IS NULL;  
SELECT * FROM Employees  
WHERE ExitDate IS NOT NULL;
```

## SQL WHERE Clause

HERE clause is used to filter records.

ed to extract only those records that fulfill a specified condition.

### RE Syntax

T *column1, column2, ...*

table\_name

E *condition;*

The WHERE clause is not only used in SELECT statements, it is also used in UPDATE , DELETE , etc!

## SQL AND, OR and NOT Operators

HERE clause can be combined with AND , OR , and NOT operators.

ND and OR operators are used to filter records based on more than one condition:

The AND operator displays a record if all the conditions separated by AND are TRUE.

The OR operator displays a record if any of the conditions separated by OR is TRUE.

OT operator displays a record if the condition(s) is NOT TRUE.

### Syntax

T *column1, column2, ...*

table\_name

E *condition1 AND condition2 AND condition3 ...;*

## OR Syntax

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition1 OR condition2 OR condition3 ...;
```

## NOT Syntax

```
SELECT column1, column2, ...  
FROM table_name  
WHERE NOT condition;
```

```

Select * from Employees
where Gendor='F';

Select * from Employees
where MonthlySalary<=500;

Select * from Employees
where MonthlySalary>500;

Select * from Employees
where Not MonthlySalary<=500;

Select * from Employees
where MonthlySalary<500 and Gendor='F';

select * from Employees
where CountryID=1;

select * from Employees
where Not CountryID=1;

select * from Employees
where CountryID <> 1;

select * from Employees
where DepartmentID=1;

select * from Employees
where DepartmentID=1 and Gendor='M';

select * from Employees
where DepartmentID=1 Or DepartmentID=2;

select * from Employees
where DepartmentID=1 AND DepartmentID=2;

Select * from Employees
where ExitDate is Null;

Select * from Employees
where ExitDate is Not Null;

```

### **"In" Operator**

هنا بيقولك بل ماتفضل كل شوية تكتب OR لما تكون عاوز مثلاً تجيب الموظفين اللي في اكتر من قسم بتكتب IN وتفتح قوسين وتكتب كل القيم اللي انت عاوزها

```

select * from Employees
where DepartmentID=1 Or DepartmentID=2 or DepartmentID=5 or
DepartmentID=7;
select * from Employees
where DepartmentID IN(1,2,5,7);

```

لو عايز اجيب كل الموظفين اللي اساميهم يعقوب وبروك وهاربر

```

SELECT * FROM Employees
WHERE FirstName IN('Jacob', 'Brooks', 'Harper');

```

عايز اجيب ارقام الأقسام اللي فيها موظفين بيقبضوا اقل من او يساوي 210

```

SELECT DepartmentID FROM Employees
WHERE MonthlySalary<=210;

```

عايز اجيب اسامي الأقسام اللي فيها موظفين بيقبضوا اقل من او يساوي 210

هنا بيقولك انك ممكن تحط جملة SQL في ال IN

يعني هجيب كل اسامي الأقسام واحدش شرط الشرط ده هيكون IN وهيكون جوا ال IN  
دي QUERY تانيه تجييلي ال ID بتاع الأقسام اللي فيها موظفين بيقبضوا اقل من  
210

```

SELECT Name FROM Departments
WHERE ID IN(SELECT DepartmentID FROM Employees WHERE MonthlySalary<=
210);

```

عايز اسامي الأقسام اللي فيها موظفين بيقبضوا اعلي من 210

```

SELECT Name FROM Departments
WHERE ID NOT IN(SELECT DepartmentID FROM Employees WHERE
MonthlySalary<=210);

```

## The SQL IN Operator

The `IN` operator allows you to specify multiple values in a `WHERE` clause.

The `IN` operator is a shorthand for multiple `OR` conditions.

### IN Syntax

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

or:

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

```

select * from Employees
where DepartmentID=1 Or DepartmentID=2;

select * from Employees
where DepartmentID=1 Or DepartmentID=2 or DepartmentID=7;

select * from Employees
where DepartmentID=1 Or DepartmentID=2 or DepartmentID=5 or DepartmentID=7;

select * from Employees
where DepartmentID in (1,2,5,7);

select * from Employees
where FirstName in ('Jacob', 'Brooks', 'Harper');

select Departments.Name from Departments
where
ID in ( select DepartmentID from Employees where MonthlySalary <=210 );

select Departments.Name from Departments
where
ID not in ( select DepartmentID from Employees where MonthlySalary <=210 );

```

### Sorting : Order By

لما بترجع داتا من جدول بترجعلك حسب ما هي متخزنها ولو عايز ترتيبها بتيجي في اخر  
جملة ال SQL وبتكتب ORDER BY وبعدها اسم العمود وبعدها ASC لو تصاعدي  
او DESC لو تنازلي

ال DEFAULT بتاعه انه يكون تصاعدي يعني مش لازم تكتب ASC  
عاوز ارجع ال ID والاسم الأول والمرتب من القسم رقم 1 وارتبهم من حيث المرتب  
الأقل للعلى

```

SELECT ID,FirstName,MonthlySalary FROM Employees
Where DepartmentID=1
ORDER BY MonthlySalary;
SELECT ID,FirstName,MonthlySalary FROM Employees
Where DepartmentID=1
ORDER BY MonthlySalary ASC;

```

عاوزه تنازليا

```
SELECT ID,FirstName,MonthlySalary FROM Employees  
Where DepartmentID=1  
ORDER BY MonthlySalary DESC;
```

عاوز ارتب تصاعديا حسب الاسم الأول والمرتب هنا هوا ييرتب الاسامي حسب الحروف الأول وبيشوف لو فيه اسامي متكرره بيحط الاسم اللي مرتبه اقل الاول

```
SELECT ID,FirstName,MonthlySalary FROM Employees  
Where DepartmentID=1  
ORDER BY FirstName, MonthlySalary ;
```

طيب لو عايز ارتبعهم حسب الاسم تصاعديا وحسب المرتب تنازليا هنا هيرتب الاسامي تصاعديا عادي بس لو لقي اسم مكرر هيحط اللي مرتبه اعلي الأول

```
SELECT ID,FirstName,MonthlySalary FROM Employees  
Where DepartmentID=1  
ORDER BY FirstName ASC, MonthlySalary DESC;  
ORDER BY Keyword
```

**BY** keyword is used to sort the result-set in ascending or descending order.

**BY** keyword sorts the records in ascending order by default. To sort the records in descending order, use **DESC** keyword.

## Y Syntax

```
nn1, column2, ...  
name  
lumn1, column2, ... ASC|DESC;
```

```

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By FirstName ;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By FirstName ASC;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By FirstName desc;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By MonthlySalary ;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By MonthlySalary Asc;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By MonthlySalary Desc;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By FirstName , MonthlySalary ;

select ID, FirstName,MonthlySalary from Employees
where DepartmentID=1
Order By FirstName ASC, MonthlySalary Desc;

```

### Select Top Statement

كلمة TOP لو كتبتها بعد كلمة SELECT وكتبت بعدها رقم ول يكن 10 مثلا هيجيبلوك اول 10 RECORDS من الجدول

لو كتبت بعدها كلمة PERCENT هيجيبللي اول 10% من ال RECORDS يعني لو عندي 100 سجل هيجيبللي اول 10 منهم

عاوز اول خمس موظفين

```
SELECT TOP 5 * FROM Employees;
```

عاوز اول 5% من الموظفين

```
SELECT TOP 5 PERCENT * FROM Employees;
```

عاوز أسماء الموظفين اللي واخدin اعلي 3 مرتبات

امشي من الاخر للأول عشان توصل

عاوزين الأول نجيب المرتبات غير مكرره ومرتبه تنازليا

```
SELECT DISTINCT MonthlySalary FROM Employees  
ORDER BY MonthlySalary DESC
```

هاتلي اول 3 منهم

```
SELECT DISTINCT TOP 3 MonthlySalary FROM Employees  
ORDER BY MonthlySalary DESC
```

هاتلي اسامي الموظفين بقى

```
SELECT ID,FirstName FROM Employees  
WHERE MonthlySalary IN(  
SELECT DISTINCT TOP 3 MonthlySalary FROM Employees  
ORDER BY MonthlySalary DESC);
```

عاوز اللي واخدin اقل 3 مرتبات

```
SELECT ID,FirstName FROM Employees  
WHERE MonthlySalary IN(  
SELECT DISTINCT TOP 3 MonthlySalary FROM Employees  
ORDER BY MonthlySalary ASC);
```

## SELECT TOP Clause

The `TOP` clause is used to specify the number of records to return.

The `TOP` clause is useful on large tables with thousands of records. Returning a large number of records can impact

base systems support the `SELECT TOP` clause. MySQL supports the `LIMIT` clause to select a limited number of rows. Oracle uses `FETCH FIRST n ROWS ONLY` and `ROWNUM`.

### Access Syntax:

`[number|percent] column_name(s)`

`;`

`;`

```

Select * from Employees;

-- This will show top 5 employees.
Select top 5 * from Employees;

-- This will show top 10% of the data.
select top 10 percent * from Employees;

-- this will show the all salaries ordered from the heighest to lowest.
select MonthlySalary from employees
order by MonthlySalary Desc;

-- this will show the all salaries ordered from the heighest to lowest without redundancy.
select distinct MonthlySalary from employees
order by MonthlySalary Desc;

-- this will show the heighest 3 salaries.
select distinct top 3 MonthlySalary from employees
order by MonthlySalary Desc;

--This will show all employees who takes one of the heighest 3 salaries.

select ID , FirstName, MonthlySalary from Employees where MonthlySalary In
(
    select distinct top 3 MonthlySalary from employees
    order by MonthlySalary Desc
)
Order By MonthlySalary Desc

--This will show all employees who takes one of the Lowest 3 salaries.
select ID , FirstName, MonthlySalary from Employees where MonthlySalary In
(
    select distinct top 3 MonthlySalary from employees
    order by MonthlySalary ASC
)
Order By MonthlySalary ASC

```

### Select As

هنا بيقولك انك ممكن بد ماتكتب SELECT تعمل عمليات رياضية

`SELECT A=5+3, B=98/5;`

طلعك عمودين بالاسامي اللي انا حطيتها وحط تحت كل واحد الناتج بتاعه

	A	B
1	8	19

ولو كتبت هيكير النتيجه بعدد ال records اللي في جدول الموظفين بدون سبب

```
SELECT A=5+3,B=98/5 FROM Employees;
```

عاوز جدول باسم الموظفين ومرتباتهم مقسمه على 2

```
SELECT ID,FirstName,A=MonthlySalary/2 FROM Employees;
```

عاوز اجيب ال id وال full name بتوع كل موظف

```
SELECT ID,FullName=FirstName+' '+LastName FROM Employees;
```

بيقولك انه ممكن تحط ال EXPRESSION او المعادله الأول وبعدين تكتب AS وبعدها تسمى العمود

```
SELECT ID,FirstName+' '+LastName AS FullName FROM Employees;
```

عاوز ال id والاسم الأول والمرتب الشهري والسنووي

```
SELECT ID,FirstName,MonthlySalary,YearlySalary=MonthlySalary*12 FROM Employees;
```

عاوز ازود البونص كقيمه

```
SELECT ID,FirstName,MonthlySalary,
YearlySalary=MonthlySalary*12,
BonusAmount=MonthlySalary*BonusPerc
FROM Employees;
```

فيه في الـ server sql اسمها function DATEDIFF بتأخذ 3 متغيرات عاوز النتيجة تكون ازاي والتاريخ الأول والتاريخ الثاني وبتطرح التاريخين وبتطلعهولك بالـ FORMAT اللي انت حددته لو كتبت الـ FORMAT خليتها YEAR هتطلعهولك بالسنين عاوزين نجيب الـ ID والـ FULL NAME والـ AGE

```
SELECT ID,  
       FullName=FirstName+ ' ' +LastName,  
       Age=DATEDIFF(YEAR,DateOfBirth,GETDATE())  
  FROM Employees;
```

عاوزين تاريخ النهارده

```
SELECT NOW=GETDATE();
```

## SQL Aliases

SQL aliases are used to give a table, or a column in a table, a temporary name.

Aliases are often used to make column names more readable.

An alias only exists for the duration of that query.

An alias is created with the AS keyword.

### Alias Column Syntax

```
SELECT column_name AS alias_name  
      FROM table_name;
```

```
Select A= 5 * 4 , B= 6/2
```

```
Select A= 5 * 4 , B= 6/2  
from employees
```

```
Select ID, FirstName, A = MonthlySalary/2  
from employees
```

```
Select ID, FirstName + ' ' + LastName as FullName From Employees;
```

```
Select ID, FullName = FirstName + ' ' + LastName From Employees;
```

```
select ID, FirstName , MonthlySalary , YealySalary = MonthlySalary * 12 from employees;
```

```
select ID, FirstName , MonthlySalary , YealySalary = MonthlySalary* 12 , BonusAmount= MonthlySalary *  
BonusPerc from employees;
```

```
select Today = getDate()
```

```
select ID, FullName= FirstName + ' ' + LastName, Age = DATEDIFF(Year , DateOfBirth ,getDate()) from  
Employees;
```

## Between Operator

كلمة BETWEEN بتحط بعدها قيمة صغرى وبعدين AND وبعدين قيمة كبيرى  
وي يجعلك كل ال RECORDS اللي بين القيمتين دول تقدر تستخدمها على الأرقام  
والتواريخ وال STRING

عاوز اجيب الموظفين اللي مرتباتهم بين ال 500 وال 1000

```
SELECT * FROM Employees  
WHERE MonthlySalary >=500 AND MonthlySalary<=1000;  
SELECT * FROM Employees  
WHERE MonthlySalary BETWEEN 500 AND 1000;
```

## BETWEEN Operator

operator selects values within a given range. The values can be numbers, text, or dates.

operator is inclusive: begin and end values are included.

### Syntax

```
name(s)  
e  
name BETWEEN value1 AND value2;
```

```
SELECT * FROM Employees WHERE MonthlySalary >=500 AND MonthlySalary <=1000;  
  
SELECT * FROM Employees WHERE MonthlySalary Between 500 and 1000;
```

## Count, Sum, Avg, Min, Max Functions

ما FUNCTIONS بتديهم عمود معين ويرجعولك القيمه  
تعالي نجريهم على عمود المرتبات

```
SELECT  
TotalCount=Count(MonthlySalary),  
TotalSum=Sum(MonthlySalary),  
Average=AVG(MonthlySalary),  
MinimumSalary=MIN(MonthlySalary),  
MaximumSalary=MAX(MonthlySalary)  
FROM Employees
```

احسبهم على القسم رقم 1

```
SELECT
TotalCount=Count(MonthlySalary),
TotalSum=Sum(MonthlySalary),
Average=AVG(MonthlySalary),
MinimumSalary=MIN(MonthlySalary),
MaximumSalary=MAX(MonthlySalary)
FROM Employees

WHERE DepartmentID=1;
```

ال COUNT ببعد ال RECORDS اللي العمود اللي انت مختاره فيه قيم يعني مش  
ببعد ال NULL

هاتلي اجمالي عدد الموظفين.

```
SELECT TotalEmployees=Count(ID) FROM Employees;
```

هاتلي عدد الموظفين اللي مشيوا

```
SELECT ResignedEmployees=Count(ExitDate) FROM Employees;
```

ال average برضه مابتاخدش القيم اللي ب null

## The SQL COUNT(), AVG() and SUM() Functions

The **COUNT()** function returns the number of rows that matches a specified criterion.

### COUNT() Syntax

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

The **AVG()** function returns the average value of a numeric column.

### AVG() Syntax

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

The **SUM()** function returns the total sum of a numeric column.

### SUM() Syntax

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

## The SQL MIN() and MAX() Functions

The `MIN()` function returns the smallest value of the selected column.

The `MAX()` function returns the largest value of the selected column.

### MIN() Syntax

```
SELECT MIN(column_name)
      FROM table_name
 WHERE condition;
```

### MAX() Syntax

```
SELECT MAX(column_name)
      FROM table_name
 WHERE condition;
```

```
select TotalCount=Count(MonthlySalary),
       TotalSum=Sum(MonthlySalary),
       Average=Avg(MonthlySalary),
       MinSalary=Min(MonthlySalary),
       MaxSalary=Max(MonthlySalary)

     from Employees;

select  TotalCount=Count(MonthlySalary),
        TotalSum=Sum(MonthlySalary),
        Average=Avg(MonthlySalary),
        MinSalary=Min(MonthlySalary),
        MaxSalary=Max(MonthlySalary)

     from Employees where DepartmentID=1

select * from employees;

select TotalEmployees = count (ID) from Employees;

--count function only counts the not null values.
select ResignedEmployees= count(ExitDate)  from employees;
```

## **Group By**

بنستخدمها في اخراج تقرير بالاجماليات والعمود اللي بتاخده بيتم التجميع علي أساسه

عاوزين الدوال بتاعت الدرس اللي فات نعملها علي مستوى الأقسام

```
SELECT DepartmentID,  
TotalCount=Count(MonthlySalary),  
TotalSum=Sum(MonthlySalary),  
Average=AVG(MonthlySalary),  
MinimumSalary=MIN(MonthlySalary),  
MaximumSalary=MAX(MonthlySalary)  
FROM Employees  
  
GROUP BY DepartmentID  
ORDER BY DepartmentID;
```

### **GROUP BY Statement**

statement groups rows that have the same values into summary rows, like "find the number of customers in each

statement is often used with aggregate functions ( COUNT() , MAX() , MIN() , SUM() , AVG() ) to group the or more columns.

### **Syntax**

```
name(s)  
e  
i  
n_name(s)  
n_name(s);
```

```

select TotalCount=Count(MonthlySalary),
       TotalSum=Sum(MonthlySalary),
       Average=Avg(MonthlySalary),
       MinSalary=Min(MonthlySalary),
       MaxSalary=Max(MonthlySalary)

     from Employees;

select  TotalCount=Count(MonthlySalary),
        TotalSum=Sum(MonthlySalary),
        Average=Avg(MonthlySalary),
        MinSalary=Min(MonthlySalary),
        MaxSalary=Max(MonthlySalary)

      from Employees where DepartmentID=3

select DepartmentID, TotalCount=Count(MonthlySalary),
       TotalSum=Sum(MonthlySalary),
       Average=Avg(MonthlySalary),
       MinSalary=Min(MonthlySalary),
       MaxSalary=Max(MonthlySalary)

     from Employees
Group By DepartmentID
order by DepartmentID

```

## **Having**

ال HAVING بتعمل فلتر للنتائج اللي طالعه من ال GROUP BY وبتكتب بعد ال GROUP BY

عندنا ال QUERY دي

```

SELECT DepartmentID,
TotalCount=Count(MonthlySalary),
TotalSum=Sum(MonthlySalary),
Average=AVG(MonthlySalary),
MinimumSalary=MIN(MonthlySalary),
MaximumSalary=MAX(MonthlySalary)
FROM Employees

GROUP BY DepartmentID
ORDER BY DepartmentID;

```

عاوزين نفلتر عالاقسام اللي فيها اكابر من 100 موظف

```
SELECT DepartmentID,
TotalCount=Count(MonthlySalary),
TotalSum=Sum(MonthlySalary),
Average=AVG(MonthlySalary),
MinimumSalary=MIN(MonthlySalary),
MaximumSalary=MAX(MonthlySalary)
FROM Employees

GROUP BY DepartmentID
HAVING Count(MonthlySalary)>100
ORDER BY DepartmentID;
```

لو عايز اعمل نفس الشغل من غير having بحط الكود في sub query وبدىها اسم  
الاسم ده هيكون اسم الجدول اللي هيطلع وبعدين استخدم where عادي

```
select * from(
SELECT DepartmentID,
TotalCount=Count(MonthlySalary),
TotalSum=Sum(MonthlySalary),
Average=AVG(MonthlySalary),
MinimumSalary=MIN(MonthlySalary),
MaximumSalary=MAX(MonthlySalary)
FROM Employees

GROUP BY DepartmentID)R2

where R2.TotalCount>100;
```

## AVING Clause

lause was added to SQL because the **WHERE** keyword cannot be used with aggregate functions in a direct way.

### Syntax

*name(s)*

*re*

*in*

*in\_name(s)*

*in*

*in\_name(s);*

```

select DepartmentID, TotalCount=Count(MonthlySalary),
       TotalSum=Sum(MonthlySalary),
       Average=Avg(MonthlySalary),
       MinSalary=Min(MonthlySalary),
       MaxSalary=Max(MonthlySalary)

     from Employees

   Group By DepartmentID

   order by DepartmentID

--Having is the where statement for group by
select DepartmentID, TotalCount=Count(MonthlySalary),
       TotalSum=Sum(MonthlySalary),
       Average=Avg(MonthlySalary),
       MinSalary=Min(MonthlySalary),
       MaxSalary=Max(MonthlySalary)

     from Employees
Group By DepartmentID
having Count(MonthlySalary) > 100

-- Same solution without having :-
select * from
(
  select DepartmentID, TotalCount=Count(MonthlySalary),
         TotalSum=Sum(MonthlySalary),
         Average=Avg(MonthlySalary),
         MinSalary=Min(MonthlySalary),
         MaxSalary=Max(MonthlySalary)

       from Employees

     Group By DepartmentID
)
R1
where R1.TotalCount> 100;

```

Like

ال like بتخليك تعمل بحث علي ال records اللي فيها قيمه شبه القيمة اللي انت دخلتها كانك تبحث عن اسم شخص ومش عارف مكتوب ازاي مثلا فبتكتب حرف او جزء من الكلمة وهو بيجيلك كل الاسامي اللي بتحتوي على المقطع ده طريقتها انك مكان الجزء اللي مش عارفه بتكتب %

عاوز كل الموظفين اللي بيبدأ اسميهم بحرف a

```
SELECT *FROM Employees  
where FirstName like 'a%';
```

عاوز اللي بينتهي بحرف ال a

```
SELECT *FROM Employees  
where FirstName like '%a';
```

عاوز كل الموظفين اللي اسميهم فيه المقطع tell

```
SELECT *FROM Employees  
where FirstName like '%tell%';
```

عاوز اللي بيبدأ وينتهي بحرف a

```
SELECT *FROM Employees  
where FirstName like 'a%a';
```

عاوز اللي تاني حرف في اسمه بيكون a

```
SELECT *FROM Employees  
where FirstName like '_a%';
```

عاوز الحرف الثالث يكون a

```
SELECT *FROM Employees  
where FirstName like '___a%';
```

عاوز اول حرف يكون a وعدد الحروف عالاقل يكونوا 3 حروف (ال a من ضمنهم)

```
SELECT *FROM Employees  
where FirstName like 'a__%' ;
```

عالقل يكونوا 4 حروف

```
SELECT *FROM Employees  
where FirstName like 'a__%' ;
```

عاوزين اللي اساميهم بتبدأ بحرف ال a او ال b

```
SELECT *FROM Employees  
where FirstName like 'a%' or FirstName like 'b%' ;
```

## IKE Operator

The **LIKE** operator is used in a **WHERE** clause to search for a specified pattern in a column.

Wildcards often used in conjunction with the **LIKE** operator:

Percent sign (%) represents zero, one, or multiple characters

Underscore sign (\_) represents one, single character

Note: MySQL uses an asterisk (\*) instead of the percent sign (%), and a question mark (?) instead of the underscore (\_).

Multiple wildcards and the underscore can also be used in combinations!

X

column1, column2, ...

me

IN **LIKE** pattern;

To combine any number of conditions using **AND** or **OR** operators.

Example showing different **LIKE** operators with '%' and '\_' wildcards:

Description	
E 'a%	Finds any values that start with "a"
E '%a'	Finds any values that end with "a"
E '%or%	Finds any values that have "or" in any position
E '_%'	Finds any values that have "r" in the second position
E 'a__%	Finds any values that start with "a" and are at least 2 characters in length
E 'a___%	Finds any values that start with "a" and are at least 3 characters in length
E 'a%o'	Finds any values that start with "a" and ends with "o"

```

USE HR_Database;

select * from Employees;

--Finds any values that start with "a"
select ID, FirstName from Employees
where FirstName like 'a%';

--Finds any values that end with "a"
select ID, FirstName from Employees
where FirstName like '%a';

--Finds any values that have "tell" in any position
select ID, FirstName from Employees
where FirstName like '%tell%';

-- Finds any values that start with "a" and ends with "a"
select ID, FirstName from Employees
where FirstName like 'a%a';

--Finds any values that have "a" in the second position
select ID, FirstName from Employees
where FirstName like '_a%';

--Finds any values that have "a" in the third position
select ID, FirstName from Employees
where FirstName like '__a%';

--Finds any values that start with "a" and are at least 3 characters in length
select ID, FirstName from Employees
where FirstName like 'a__%';

--Finds any values that start with "a" and are at least 4 characters in length
select ID, FirstName from Employees
where FirstName like 'a___%';

--Finds any values that start with "a"
select ID, FirstName from Employees
where FirstName like 'a%' or FirstName like 'b%' ;

```

## WildCards

هنعمل الأول update للجدول (لاحظ انه كاتب محمد مره بال a ومره بال e

```
Update Employees  
set FirstName = 'Mohammed' , LastName='Abu-Hadhoud'  
where ID= 285;
```

```
Update Employees  
set FirstName = 'Mohammad' , LastName='Maher'  
where ID= 286;
```

ال野 cards يتم استخدامهم مع ال like  
لوانا عايز اجيب اللي اسمائهم محمد وانا عارف انه فيه ناس بتكتبه ad وفيه ناس  
بتكتبه ed

```
select ID, FirstName, LastName from Employees  
Where firstName = 'Mohammed' or FirstName = 'Mohammad';
```

طالما انا عارف انه الاختلاف ممكن يكون في مكان واحد في الكلمة ممكن اكتب []  
واحط بينهم الحرفين اللي شاكك فيهم

```
select ID, FirstName, LastName from Employees  
Where firstName like 'Mohamm[ae]d';
```

عاوز كل الناس اللي اسمائهم مش محمد

```
select ID, FirstName, LastName from Employees  
Where firstName not like 'Mohamm[ae]d';
```

عاوز كل الناس اللي اسمائهم بتبدأ بال a او b او c

```
select ID, FirstName, LastName from Employees  
Where firstName like 'a%' or firstName like 'b%' or firstName like 'c%';
```

```
select ID, FirstName, LastName from Employees  
Where firstName like '[abc]%' ;
```

علامة - الطرح بتعبر عن المدي او ال range

عاوز كل اللي بيبدأ من حرف a او b او c لحد حرف ال ا

```
select ID, FirstName, LastName from Employees  
Where firstName like '[a-zA-Z%];
```

## Wild Characters

Character is used to substitute one or more characters in a string.

Caracter are used with the `LIKE` operator. The `LIKE` operator is used in a `WHERE` clause to search for a specified pattern.

### Special Characters in SQL Server

Description	Example
Presents zero or more characters	bl% finds bl, black, blue, and blob
Presents a single character	h_t finds hot, hat, and hit
Presents any single character within the brackets	h[o-e]t finds hot and hat, but not hit
Presents any character not in the brackets	h[^o-e]t finds hit, but not hot and hat
Presents any single character within the specified range	c[a-b]t finds cat and cbt

```
--Execute these statements to update data
Update Employees
set FirstName = 'Mohammed' , LastName='Abu-Hadhoud'
where ID= 285;

Update Employees
set FirstName = 'Mohammad' , LastName='Maher'
where ID= 286;

-----
select ID, FirstName, LastName from Employees
Where firstName = 'Mohammed' or FirstName ='Mohammad';

-- will search form Mohammed or Mohammad
select ID, FirstName, LastName from Employees
Where firstName like 'Mohamm[ae]d';

-----
--You can use Not
select ID, FirstName, LastName from Employees
Where firstName Not like 'Mohamm[ae]d';

-----
select ID, FirstName, LastName from Employees
Where firstName like 'a%' or firstName like 'b%' or firstName like 'c%';

-- search for all employees that their first name start with a or b or c
select ID, FirstName, LastName from Employees
Where firstName like '[abc]%';

-----
-- search for all employees that their first name start with any letter from a to l
select ID, FirstName, LastName from Employees
Where firstName like '[a-l]%';
```

### Restore Shop Database

<https://cdn.fs.teachablecdn.com/7SRuonRdq1Cu2nea0SAS>

```
RESTORE DATABASE Shop_Database FROM DISK' =D:\Shop_Database.bak';
USE Shop_Database;
EXEC sp_changedbowner 'sa';
```

### (Inner) Join

قبل كده كنا بنجيب الداتا من جدول واحد فلو عايزين نجيب داتا من اكتر من جدول

بنستخدم ال JOIN STATEMENT ودي عباره عن جملة بترتبط اكتر من جدول بعضهم

طيب هنا بيقولك انه فيه اربع انوع من ال JOINS :-

• JOIN INNER : - وده معظم شغلك هيكون عليه وده بيطلع الداتا المشتركة بين الجدولين او ال RECORDS اللي حصل بينهم ربط زي انه يرجعلك العملاء اللي عملوا اوردرات بس

• JOIN LEFT : - وده بي يجعلك كل الداتا بتاعت الجدول اللي عالشمال بما فيهم الداتا المشتركة من الجدول اللي عاليمين

• JOIN RIGHT : - بيجعلك كل الداتا بتاعت الجدول اللي عاليمين ومعاه الداتا المرتبطه بيها من الجدول اللي عالشمال

• JOIN FULL OUTER : - وده بيجيبلوك الداتا بتاعت الجدولين كلهم هنبدأ في ال inner join وساعات بيقولوا عليها join بص كده عالجدائل دي

## SQL INNER JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
3	David	500
5	Betty	800

الجدول بتاعت ال orders بتلاقي فيه عملاء مش موجودين في جدول ال customers وده مش موجود في الحياه العمليه لانه عادة بتلاقي الجدولين مربوطين بعض انما هنا الجدولين مالهموش علاقه بعض

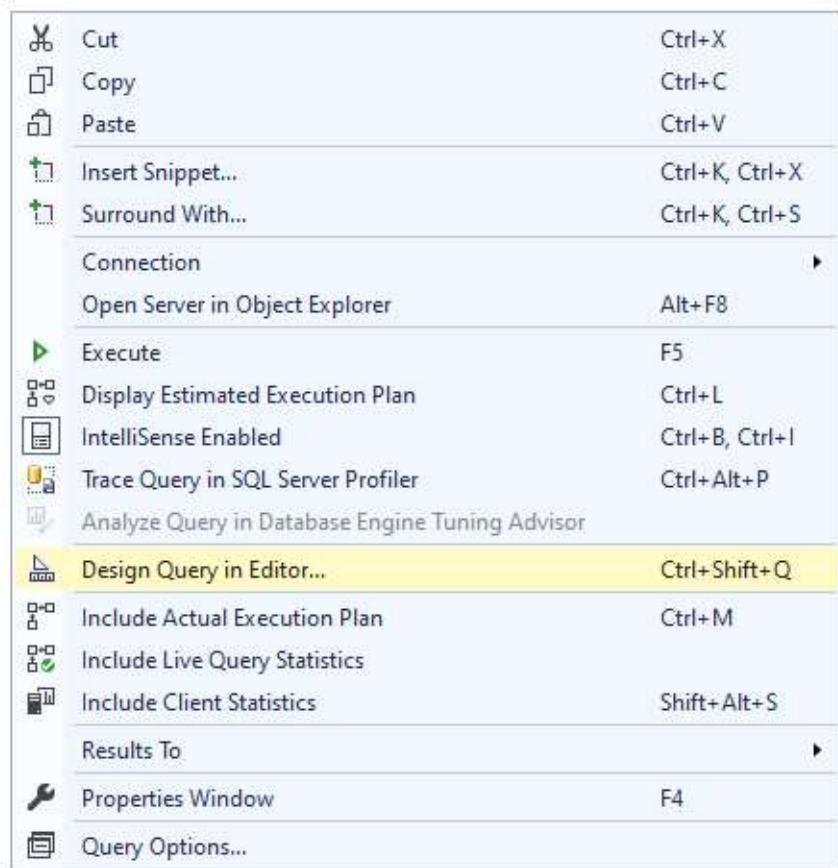
هنا لما جينا نعمل inner join جابلي الداتا المشتركه بس بين الجدولين او الاوردرات اللي ليها عملاء

طريقة كتابة ال select statement هيا انك تكتب اسامي الاعمده اللي انت عاوزها بس تكتب اسم الجدول قبليه عشان تعرف انت هتجيب العمود منين وبعددين on وبعدها اسم الجدول الأول وبعددين inner join باسم الجدول الثاني وبعددين on

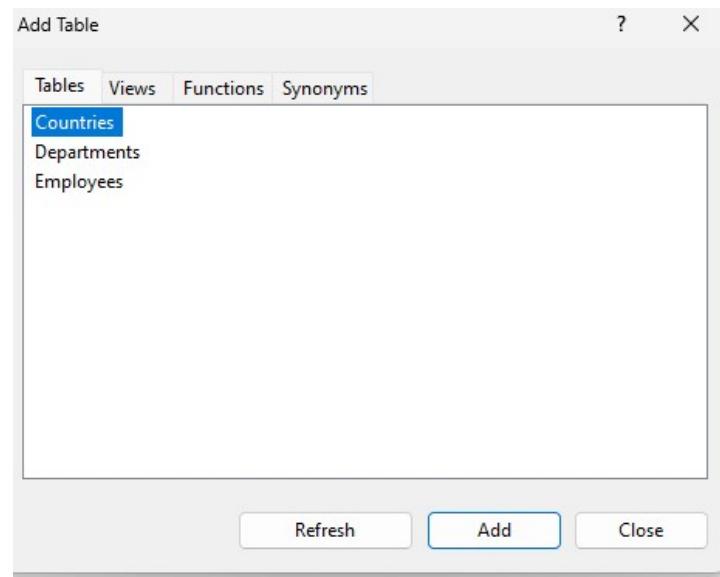
وبعدين الشرط او ال keys اللي بترتبط الجدولين بعض  
ودي ال select statement بتاعتها

```
SELECT Customers.CustomerID,Customers.Name,Orders.Amount
FROM Customers INNER JOIN Orders
ON Customers.CustomerID=Orders.CustomerID;
```

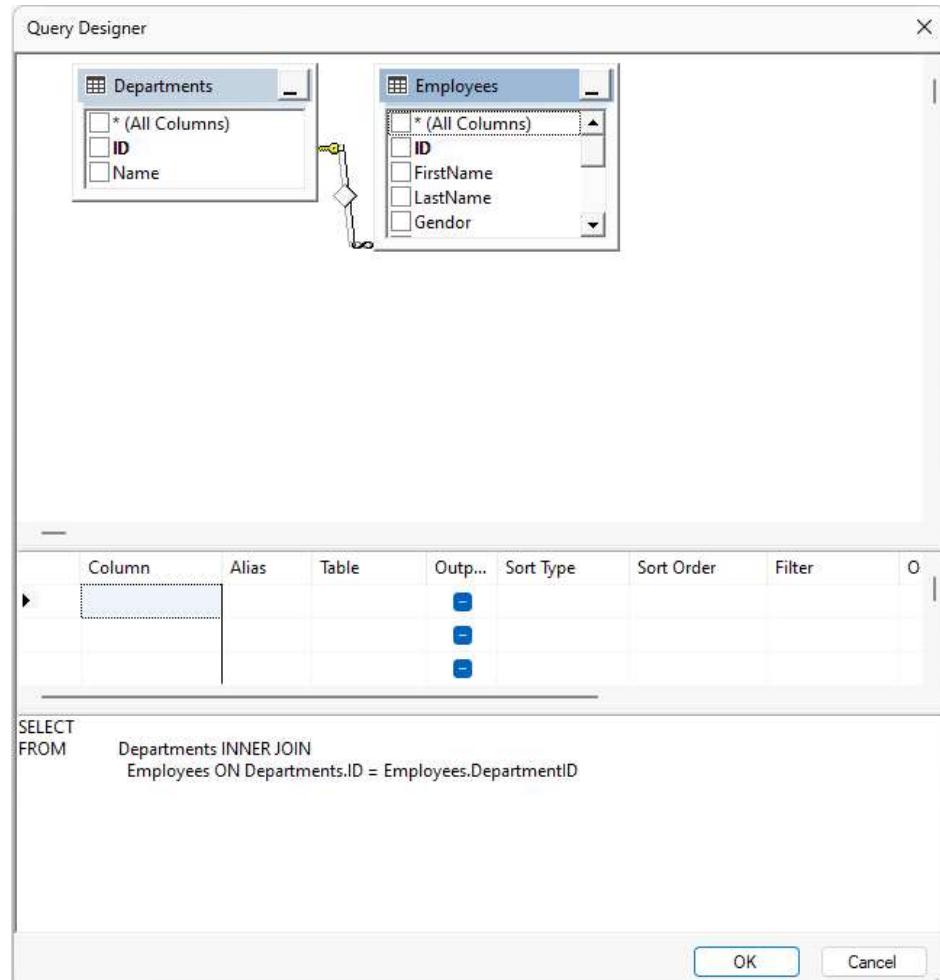
نقدر نعمل ده بالديزاین زي كده هنعملها عالداتا بيذ بتاعت ال HR



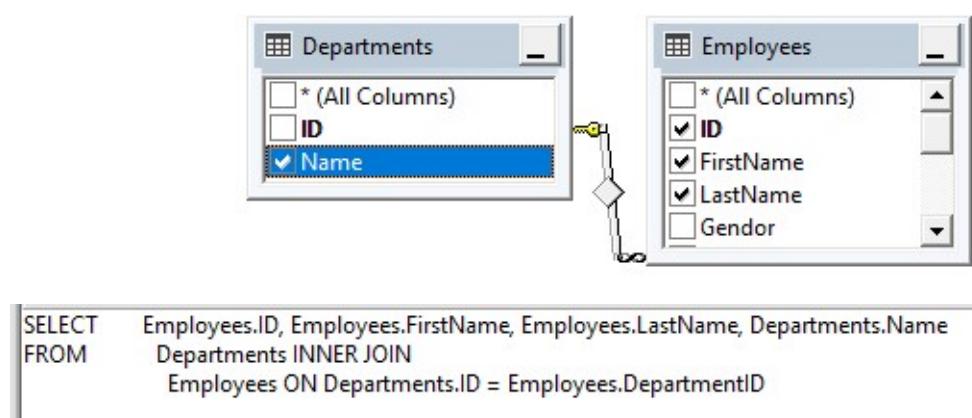
وبعدين هنختار جدول ال EMPLOYEES وجدول ال DEPARTMENT



هلاقیه کتبی ال SELECT STATEMENT لوحه في الشاشه تحت



هناختار الاعمدہ اللي عاوزین نعرضها



```
SELECT Employees.ID, Employees.FirstName, Employees.LastName, Departments.Name  
FROM Departments INNER JOIN  
                 Employees ON Departments.ID = Employees.DepartmentID
```

	ID	FirstName	LastName	Name
1	285	Mohammed	Abu-Hadhoud	Engineering
2	286	Mohammad	Maher	Sales
3	287	Cameron	Lo	IT
4	288	Harper	Castillo	IT
5	289	Harper	Dominguez	Engineering
6	290	Ezra	Vu	IT
7	291	Jade	Hu	Accounting

عاوز الموظفين اللي شغالين في ال it بس

```
SELECT Employees.ID, Employees.FirstName, Employees.LastName,
Departments.Name AS DeptName
FROM Departments INNER JOIN
Employees ON Departments.ID = Employees.DepartmentID
WHERE Departments.Name='IT';
```

لو ماشتغلتش روح اكتب أسماء الأقسام تاني في جدول ال DEPARTMENT

عاوزين نجيب الاسم والقسم والبلد

عشان تربط جدولين مع جدول بتكتب اول علاقه وبعدها تاني علاقه من غير ماتكرر  
اسم الجدول اللي مرتبط بيهم

```
SELECT Employees.ID,Employees.FirstName,Employees.LastName,Departments.
Name AS Department,Countries.Name AS Country

FROM Employees INNER JOIN Departments ON
Employees.DepartmentID=Departments.ID
INNER JOIN Countries ON Employees.CountryID=Countries.ID;
```

فلتر الداتا علي اللي بلدتهم أمريكا

```
SELECT Employees.ID,Employees.FirstName,Employees.LastName,Departments.  
Name AS Department,Countries.Name AS Country  
  
FROM Employees INNER JOIN Departments ON  
Employees.DepartmentID=Departments.ID  
INNER JOIN Countries ON Employees.CountryID=Countries.ID  
  
WHERE Countries.Name='USA';
```

; used to combine rows from two or more tables, based on a related column between them.

## Types of SQL JOINS

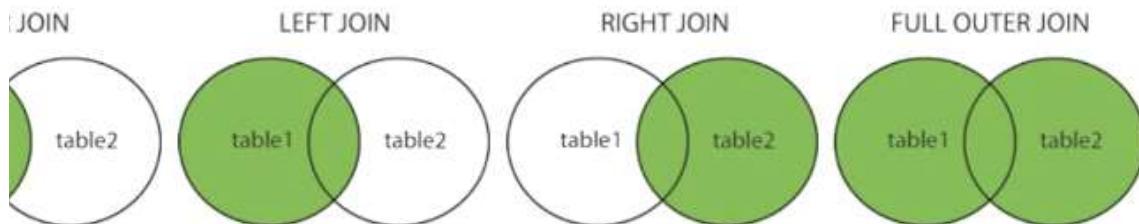
Four types of the JOINS in SQL:

**JOIN** : Returns records that have matching values in both tables

**LEFT JOIN** : Returns all records from the left table, and the matched records from the right table

**RIGHT JOIN** : Returns all records from the right table, and the matched records from the left table

**FULL OUTER JOIN** : Returns all records when there is a match in either left or right table



## INNER JOIN

**JOIN** joins two tables based on a common column, and selects records that have matching values in these

```
SELECT *  
FROM facebook  
JOIN linkedin  
ON facebook.name = linkedin.name
```

facebook

Name	# of Friends
Matt	300
Lisa	500
Jeff	600
Sarah	400

linkedin

Name	# of connections
Matt	500
Lisa	200
Sarah	100
Louis	300

facebook and linkedin JOINed

facebook.Name	facebook.# of Friends	linkedin.Name	linkedin.# of connections
Matt	300	Matt	500

```
Customers.customer_id, Customers.first_name, Orders.amount  
's  
Orders  
.customer_id = Orders.customer;
```

## SQL INNER JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

↓

customer_id	first_name	amount
3	David	500
5	Betty	800

Command selects **customer\_id** and **first\_name** columns (from the **Customers** table) and the **amount** column (from

will contain those rows where there is a match between **customer\_id** (of the **Customers** table) and **customer** (of

## INNER JOIN

**INNER JOIN** is:

```
s  
ble2  
umn_name = table2.column_name;
```

## With WHERE Clause

of the **INNER JOIN** with the **WHERE** clause:

```
lers.customer_id, Customers.first_name, Orders.amount  
s  
ders  
customer_id = Orders.customer  
amount >= 500;
```

This command joins two tables and selects rows where the amount is greater than or equal to 500.

```
-- Join and Inner Join are the same  
  
select * from Customers;  
  
select * from Orders;  
  
SELECT Customers.CustomerID, Customers.Name, Orders.Amount  
FROM Customers  
JOIN Orders  
ON Customers.CustomerID = Orders.CustomerID;  
  
SELECT Customers.CustomerID, Customers.Name, Orders.Amount  
FROM Customers  
Inner JOIN Orders  
ON Customers.CustomerID = Orders.CustomerID;
```

```
--This code for HR_Database

--Inner Join two Tables
SELECT      Employees.ID, Employees.FirstName, Employees.LastName, Departments.Name as DeptName
FROM        Employees INNER JOIN
                      Departments ON Employees.DepartmentID = Departments.ID

--Inner joind with where
SELECT      Employees.ID, Employees.FirstName, Employees.LastName, Departments.Name as DeptName
FROM        Employees INNER JOIN
                      Departments ON Employees.DepartmentID = Departments.ID
where Departments.Name = 'IT';

--Inner Join Three Tables
SELECT      Employees.ID, Employees.FirstName, Employees.LastName, Departments.Name as DeptName,
Countries.Name AS CountryName
FROM        Employees INNER JOIN
                      Departments ON Employees.DepartmentID = Departments.ID INNER JOIN
                      Countries ON Employees.CountryID = Countries.ID

--Inner Join Three Tables with where
SELECT      Employees.ID, Employees.FirstName, Employees.LastName, Departments.Name as DeptName,
Countries.Name AS CountryName
FROM        Employees INNER JOIN
                      Departments ON Employees.DepartmentID = Departments.ID INNER JOIN
                      Countries ON Employees.CountryID = Countries.ID
where Countries.Name = 'USA';
```

### Left (Outer) Join

ال LEFT JOIN بيأخذ كل الداتا من الجدول اللي عالشمال والداتا المشتركه بس من اللي عاليمين وساعات بيقولوا عليه LEFT OUTER JOIN  
اني اجيبي كل العملاء والاوردرات اللي عملوها في الجدول ده

## SQL LEFT JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

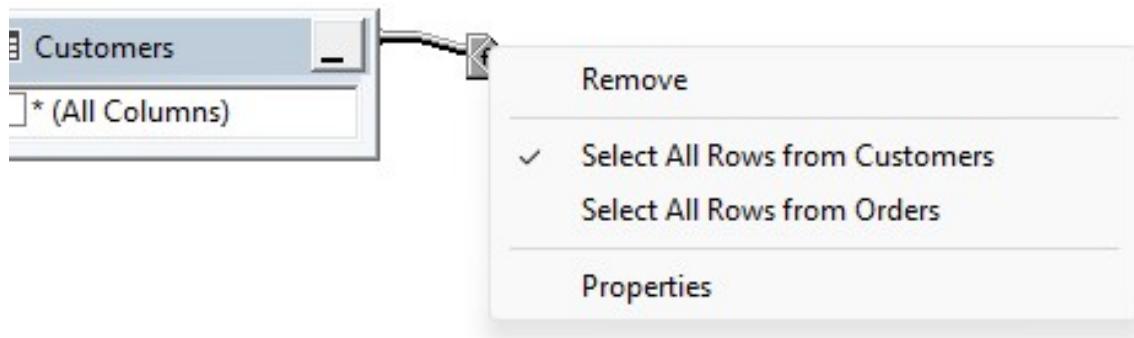
order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
1	John	
2	Robert	
3	David	500
4	John	
5	Betty	800

طريقتها اني بدل ماكنت بكتب INNER لا بكتب LEFT

```
USE Shop_Database;
SELECT Customers.CustomerID,Customers.Name,Orders.Amount
FROM Customers LEFT JOIN Orders
ON Customers.CustomerID=Orders.CustomerID;
```

عشان اعملها في ال DESIGNER بعمل كليك يمين عالسهم بين الجدولين



## DIN Keyword

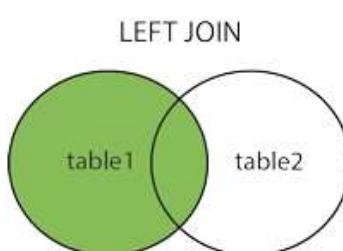
**I** keyword returns all records from the left table (table1), and the matching records from the right table (table2). The from the right side, if there is no match.

### Syntax

*name(s)*

*name = table2.column\_name;*

atabases LEFT JOIN is called LEFT OUTER JOIN.



**JOIN** joins two tables based on a common column, and selects records that have matching values in these columns ws from the left table.

```

SELECT *
FROM facebook
LEFT JOIN linkedin
ON facebook.name = linkedin.name

```

Name	# of Friends
Matt	300
Lisa	500
Jeff	600
Sarah	400

Name	# of connections
Matt	500
Lisa	200
Sarah	100
Louis	300

facebook and linkedin JOINed

facebook.Name	facebook.# of Friends	linkedin.Name	linkedin.# of connections
Matt	300	Matt	500
Lisa	500	Lisa	200

```

    iers.customer_id, Customers.first_name, Orders.amount
's
lers
customer_id = Orders.customer;

```

## SQL LEFT JOIN

Table: Customers

customer_id	first_name
1	John
2	Robert
<u>3</u>	David
4	John
<u>5</u>	Betty

Table: Orders

order_id	amount	customer
1	200	10
2	500	<u>3</u>
3	300	6
4	800	<u>5</u>
5	150	8

customer_id	first_name	amount
1	John	
2	Robert	
3	David	500
4	John	
5	Betty	800

The command selects **customer\_id** and **first\_name** columns (from the **Customers** table) and the **amount** column (from the **Orders** table).

It will contain those rows where there is a match between **customer\_id** (of the **Customers** table) and **customer** (of the **Orders** table) along with all the remaining rows from the **Customers** table.

--Left Join and Left Outer Join are the same.

--Left Join: gets all data from table customers and only matched data from table orders

```
SELECT Customers.CustomerID, Customers.Name, Orders.Amount
```

```
FROM Customers
```

```
Left JOIN Orders
```

```
ON Customers.CustomerID = Orders.CustomerID;
```

```
SELECT Customers.CustomerID, Customers.Name, Orders.Amount
```

```
FROM Customers
```

```
Left Outer JOIN Orders
```

```
ON Customers.CustomerID = Orders.CustomerID;
```

## Right (Outer) Join + Full (Outer) Join

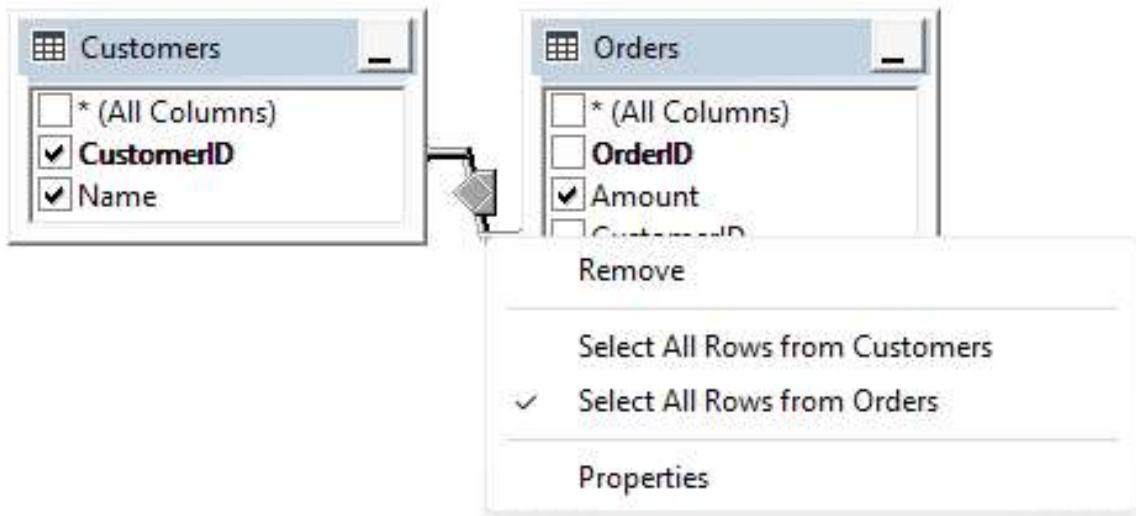
ال RIGHT JOIN او ال RIGHT OUTER JOIN بيجيب كل الداتا من اللي عاليمين  
والمشترك اللي في الشمال

وال FULL JOIN او ال FULL OUTER JOIN بيجيب كل الداتا من الجدولين

طريقتهم انك بس بتكتب اسمائهم

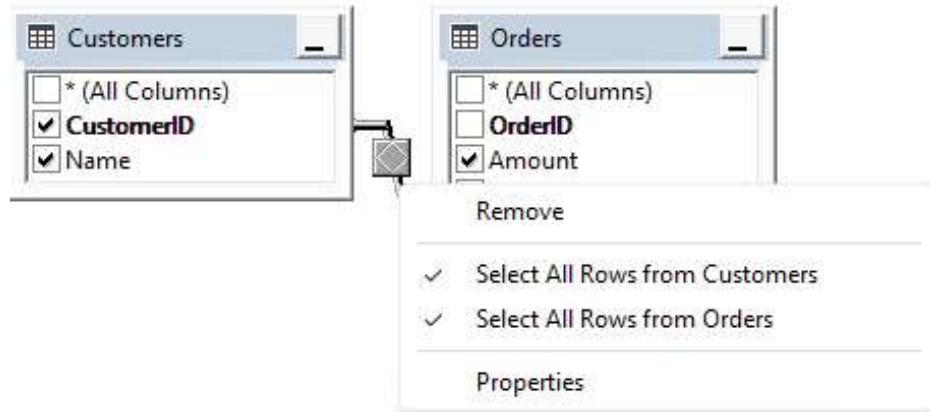
دي نفس ال QUERY بس بال

```
SELECT Customers.CustomerID,Customers.Name,Orders.Amount  
FROM Customers RIGHT JOIN Orders  
ON Customers.CustomerID=Orders.CustomerID;
```



دي نفس ال QUERY بس بال

```
SELECT Customers.CustomerID,Customers.Name,Orders.Amount  
FROM Customers FULL JOIN Orders  
ON Customers.CustomerID=Orders.CustomerID;
```



## JOIN Keyword

**N** keyword returns all records from the right table (table2), and the matching records from the left table (table1).  
rds from the left side, if there is no match.

SELECT \*  
FROM facebook  
JOIN linkedin  
ON facebook.name = linkedin.name

facebook	linkedin
Name	# of Friends
Matt	300
Lisa	500
Jeff	600
Sarah	400

3

facebook	linkedin
Name	# of connections
Matt	500
Lisa	200
Sarah	100
Louis	300

facebook and linkedin JOINed

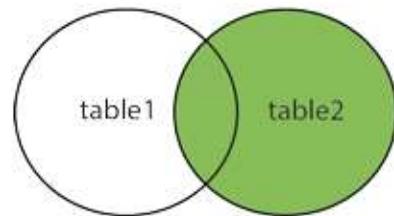
facebook.Name	facebook.# of Friends	linkedin.Name	linkedin.# of connections
Matt	300	Matt	500
Lisa	500	Lisa	200

## RIGHT JOIN Syntax

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```

**Note:** In some databases **RIGHT JOIN** is called **RIGHT OUTER JOIN**.

RIGHT JOIN



## OUTER JOIN Keyword

**TER JOIN** keyword returns all records when there is a match in left (table1) or right (table2) table records.

**TER JOIN** and **FULL JOIN** are the same.

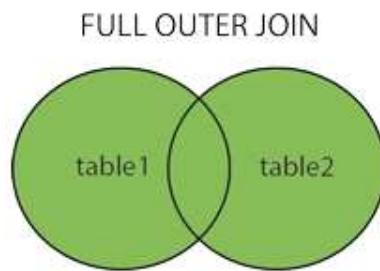
### IR JOIN Syntax

```
l_name(s)
```

```
IN table2
```

```
m_name = table2.column_name
```

```
on;
```



**UTER JOIN** can potentially return very large result-sets!

facebook and linkedin JOINed

**facebook**

Name	# of Friends
Matt	300
Lisa	500
Jeff	600
Sarah	400

**linkedin**

Name	# of connections
Matt	500
Lisa	200
Sarah	100
Louis	300

facebook.Name	facebook.# of Friends	linkedin.Name	linkedin.# of connections
Matt	300	Matt	500
Lisa	500	Lisa	200
Jeff	600	Null	Null
Sarah	400	Sarah	100

```
--Inner Join
SELECT      Customers.CustomerID, Customers.Name, Orders.Amount
FROM        Customers INNER JOIN
                  Orders ON Customers.CustomerID = Orders.CustomerID
--Left Join
SELECT      Customers.CustomerID, Customers.Name, Orders.Amount
FROM        Customers LEFT OUTER JOIN
                  Orders ON Customers.CustomerID = Orders.CustomerID
--Right Join
SELECT      Customers.CustomerID, Customers.Name, Orders.Amount
FROM        Customers RIGHT OUTER JOIN
                  Orders ON Customers.CustomerID = Orders.CustomerID
--Full Join
SELECT      Customers.CustomerID, Customers.Name, Orders.Amount
FROM        Customers FULL OUTER JOIN
                  Orders ON Customers.CustomerID = Orders.CustomerID
```

## Views

لو فيه عندي QUERY بستخدمها كتير اقدر احطها جوه VIEW وهو زيه زي ال  
FUNCTION

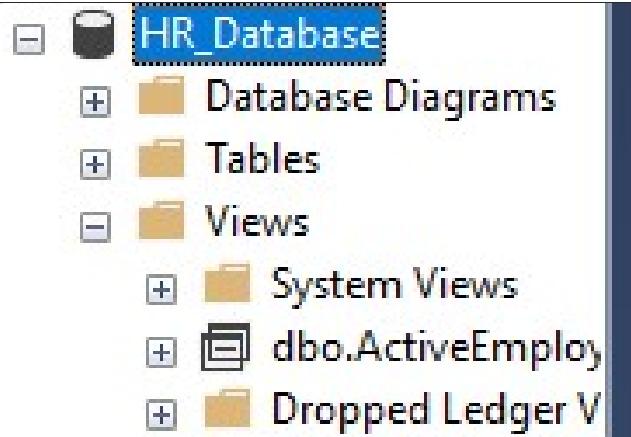
عشان اعمل VIEW بكتب CREATE VIEW وبعدين بكتب اسم ليه وبعدين AS  
وبعدين بحط ال QUERY جواه

عاوز الموظفين اللي لسه موجودين في الشركه

```
SELECT * FROM Employees  
WHERE ExitDate IS NULL;
```

عاوز احط ال VIEW اللي عملتها في

```
CREATE VIEW ActiveEmployees AS  
SELECT * FROM Employees  
WHERE ExitDate IS NULL;
```



كده اقدر اتعامل مع ال VIEW دي كانها جدول بس هيا مش جدول هيا اختصار  
للكود

```
SELECT* FROM ActiveEmployees;
```

عاوز اعمل view للناس اللي مشيت

```
CREATE VIEW ResignedEmployees AS  
SELECT * FROM Employees  
WHERE ExitDate IS NOT NULL
```

```
SELECT * FROM ResignedEmployees
```

عاوز اعمل VIEW للاسم والجنس بس

```
CREATE VIEW ShortDetailedEmployees AS  
SELECT ID, FirstName, LastName, Gender FROM Employees  
SELECT* FROM ShortDetailedEmployees
```

اقدر اعمل صلاحيات علي اليوزر انه مايشوفش غير ال VIEW ده

## THE VIEW Statement

a virtual table based on the result-set of an SQL statement.

rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

statements and functions to a view and present the data as if the data were coming from one single table.

with the `CREATE VIEW` statement.

## VIEW Syntax

```
view_name AS  
(, column2, ...  
ie  
n;
```

ays shows up-to-date data! The database engine recreates the view, every time a user queries it.

## SQL - More Queries

### Exists

ال EXIST زي ماقولنا قبل كده بتكتب جواها QUERY ولو في RECORD واحد او اكتر طبع بترجعلك TRUE

اعملني عمود اسمه X وحط فيه الكلمة YES لو عندك عملاء ال ID بتعهم ب 3 والطلبات بتاعتهم عامله اقل من 600

```

SELECT X='YES'
WHERE EXISTS(
SELECT * FROM Orders
WHERE Orders.CustomerID=3 AND Orders.Amount<600)

```

	X
1	YES

رجعلي جدول ال customers كله وسميه t1 لو فيه عميل من جدول ال order id بتابعه موجود في ال t1 وال فلوس اقل من 600

```

SELECT *from Customers T1
WHERE EXISTS(
SELECT * FROM Orders
WHERE Orders.CustomerID=T1.CustomerID AND Orders.Amount<600)

```

ال اللي عملناها دي بطئيه لأن ال exists مهمها كانت الداتا اللي هترجع  
false او true

عشان كده بيقولك بدل مايرجع الداتا كلها لا خليه يرجع اول record بس لأنه ده اقل  
شيء مطلوب عشان يرجع true

```

SELECT *from Customers T1
WHERE EXISTS(
SELECT top 1 * FROM Orders
WHERE Orders.CustomerID=T1.CustomerID AND Orders.Amount<600)

```

ممكن أخليها اسرع واسرع وهيا بدل مايجي بي كل الأعمده لا هخليه يرجع أي حاجة  
من عندي

```

SELECT *from Customers T1
WHERE EXISTS(
SELECT top 1 x='y' FROM Orders
WHERE Orders.CustomerID=T1.CustomerID AND Orders.Amount<600)

```

## The SQL EXISTS Operator

The **EXISTS** operator is used to test for the existence of any record in a subquery.

The **EXISTS** operator returns TRUE if the subquery returns one or more records.

### SQL EXISTS Syntax

**SELECT** *column\_name(s)*

**FROM** *table\_name*

**WHERE EXISTS**

**SELECT** *column\_name* **FROM** *table\_name* **WHERE** *condition*;

The SQL EXISTS operator executes the outer SQL query if the subquery is not NULL (empty result-set).

```

select X='yes'
where exists
(
    select * from Orders
    where customerID= 3 and Amount < 600
)

select * from Customers T1
where
exists
(
    select * from Orders
    where customerID= T1.CustomerID and Amount < 600
)

--More optimized and faster
select * from Customers T1
where
exists
(
    select top 1 * from Orders
    where customerID= T1.CustomerID and Amount < 600
)

--More optimized and faster
select * from Customers T1
where
exists
(
    select top 1 R='Y'  from Orders
    where customerID= T1.CustomerID and Amount < 600
)

```

## Union

لو عندي جدولين او اتنين result sets ليهم نفس الاعمده بنفس المسميات اقدر احطهم كلام في query واحده عن طريق انك تكتب union بينهم

```

select * from ActiveEmployees
union
select * from ResignedEmployees

```

لو فيه records متكرره بتشيلهم

```
select * from Departments  
union  
select * from Departments
```

لو كتبت كلمة ALL مش هييشيل المتكرر

```
select * from Departments  
union ALL  
select * from Departments
```

## SQL UNION Operator

**UNION** operator is used to combine the result-set of two or more **SELECT** statements.

every **SELECT** statement within **UNION** must have the same number of columns

ie columns must also have similar data types

ie columns in every **SELECT** statement must also be in the same order

### I Syntax

```
:column_name(s) FROM table1
```

```
:column_name(s) FROM table2;
```

### I ALL Syntax

**UNION** operator selects only distinct values by default. To allow duplicate values, use **UNION ALL** :

```
:column_name(s) FROM table1
```

```
ALL
```

```
:column_name(s) FROM table2;
```

the column names in the result-set are usually equal to the column names in the first **SELECT** statement.

وتقدير تضييف عمود يميز الداتا اللي راجعه من كل جدول عن طريق اضافه عمود جديد زي كده

```
select *,X=1 from Departments
union ALL
select *,X=2 from Departments
```

لو عندك جدول فيه فواتير مشتريات وجدول تاني فيه فواتير مبيعات وعايز تجيبيهم كلهم في جدول واحد مثلاً وتضييف عمود عمود يبينلك نوع الفاتوره ومن خالله تقدر تطلع تقرير بارصدة الأصناف

### Case

ال CASE هنا شبه مفهوم ال SWITCH في البرمجه  
طريقتها اني بكتب case وتحتها الكلمة end وفي النص هحط الشروط كانها if else statement

بكتب when وبعدها الشرط وبعدين then وبعدين اللي عايز اعمله  
ولو فيه حالات تانية بكتبها بنفس الطريقة  
وفي الآخر بكتب else وبحط الحاجه ال default  
لو عندي ال query دي

```
SELECT ID,FirstName,LastName from Employees;
```

وعايز استبدل الحرف m بكلمة male والحرف f بكلمة female وحطهم في عمود جديد

```
SELECT ID,FirstName,LastName ,GenderTitle=
CASE
WHEN Gendor='M' THEN 'Male'
WHEN Gendor='F' THEN 'Female'
ELSE 'Unknown'
END
from Employees;
```

عاوز ازود عمود يقولي لو الموظف شغال لا مشي

```
SELECT ID, FirstName, LastName ,GenderTitle=CASE  
WHEN Gendor='M' THEN 'Male'  
WHEN Gendor='F' THEN 'Female'  
ELSE 'Unknown'  
END  
,  
Status=CASE  
WHEN ExitDate IS NULL THEN 'NOT ACTIVE'  
WHEN ExitDate IS NOT NULL THEN 'ACTIVE'  
ELSE 'Unknown'  
END  
from Employees;
```

هنا لو النوع ذكر بيزيوده 10% من المرتب ولو انثى بتزيد 15%

```
select ID, FirstName, LastName, MonthlySalary,  
NewSalaryToBe =  
CASE  
WHEN Gendor='M' THEN MonthlySalary * 1.1  
WHEN Gendor='F' THEN MonthlySalary * 1.15  
END  
from Employees
```

## **ASE Expression**

ession goes through conditions and returns a value when the first condition is met (like an if-then-else statement).  
ion is true, it will stop reading and return the result. If no conditions are true, it returns the value in  
e.

**SE** part and no conditions are true, it returns NULL.

## **IX**

*on1 THEN result1*

*on2 THEN result2*

*onN THEN resultN*

```

select ID, FirstName, LastName, GendorTitle =
CASE
    WHEN Gendor='M' THEN 'Male'
    WHEN Gendor='F' THEN 'Female'
    ELSE 'Unknown'
END

from Employees
-----


select ID, FirstName, LastName, GendorTitle =
CASE
    WHEN Gendor='M' THEN 'Male'
    WHEN Gendor='F' THEN 'Female'
    ELSE 'Unknown'
END,
Status =
CASE
    WHEN ExitDate is null THEN 'Active'
    WHEN Gendor is Not null THEN 'Resigned'
END
from Employees
-----


select ID, FirstName, LastName, MonthlySalary,
NewSalaryToBe =
CASE
    WHEN Gendor='M' THEN MonthlySalary * 1.1
    WHEN Gendor='F' THEN MonthlySalary * 1.15
END
from Employees

```

## More about Constraints

### Constraints

## Constraints

be specified when the table is created with the `CREATE TABLE` statement, or after the table is created with `ALTER` statement.

```
table_name(  
    pe constraint,  
    pe constraint,  
    pe constraint,
```

aints

re used to specify rules for the data in a table.

sed to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table.

lation between the constraint and the data action, the action is aborted.

e column level or table level. Column level constraints apply to a column, and table level constraints apply to the

straints are commonly used in SQL:

L - Ensures that a column cannot have a NULL value

- Ensures that all values in a column are different

**KEY** - A combination of a **NOT NULL** and **UNIQUE**. Uniquely identifies each row in a table.

| KEY - Prevents actions that would destroy links between tables

Ensures that the values in a column satisfies a specific condition

`[ ] - Sets a default value for a column if no value is specified`

**TINDEX** - Used to create and retrieve data from the database very quickly.

## More about Constraints

## What is Constraint? and Why it's Important?

<https://programmingadvices.com/courses/database-level-1-sql-concepts-and-practice/lectures/46898430>

## Primary Key Constraint

هنا فيه معلومه لو عايز تستخدم اكتر من عمود ك PRIMARY KEY

بتكتب CONSTRAINT وبعدها بتكتب اسم عمود جديد وبعدين PRIMARY KEY وفتتح قوسين وتحط فيهم اسامي الاعمده اللي عاوزها تبقى PRIMARY KEY مع بعض

```

CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
);

```

	Column Name	Data Type	Allow Nulls
PK	ID	int	<input type="checkbox"/>
PK	LastName	varchar(255)	<input type="checkbox"/>
	FirstName	varchar(255)	<input checked="" type="checkbox"/>
	Age	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

عشان تحذف عمود ال primary key اللي عملته من عمودين بتكتب قبل اسمه constraint

```

ALTER TABLE Persons
DROP CONSTRAINT PK_Person;

```

## PRIMARY KEY Constraint

KEY constraint uniquely identifies each record in a table.

ast contain UNIQUE values, and cannot contain NULL values.

: only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

## PRIMARY KEY on CREATE TABLE

llowing SQL creates a **PRIMARY KEY** on the "ID" column when the "Persons" table is created:

```
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY KEY,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
```

ow naming of a **PRIMARY KEY** constraint, and for defining a **PRIMARY KEY** constraint on multiple columns, use the  
ing SQL syntax:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255),
    CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
```

In the example above there is only ONE **PRIMARY KEY** (PK\_Person). However, the VALUE of the primary key is made up of  
COLUMNS (ID + LastName).

## SQL PRIMARY KEY on ALTER TABLE

To create a **PRIMARY KEY** constraint on the "ID" column when the table is already created, use the following SQL:

```
ALTER TABLE Persons
```

```
ADD PRIMARY KEY (ID);
```

To allow naming of a **PRIMARY KEY** constraint, and for defining a **PRIMARY KEY** constraint on multiple columns, use the following SQL syntax:

```
ALTER TABLE Persons
```

```
ADD CONSTRAINT PK_Person PRIMARY KEY (ID,LastName);
```

**Note:** If you use **ALTER TABLE** to add a primary key, the primary key column(s) must have been declared to not contain NULL values (when the table was first created).

---

## DROP a PRIMARY KEY Constraint

To drop a **PRIMARY KEY** constraint, use the following SQL:

```
ALTER TABLE Persons
```

```
DROP CONSTRAINT PK_Person;
```

## Foreign Key Constraint

## SQL FOREIGN KEY Constraint

The **FOREIGN KEY** constraint is used to prevent actions that would destroy links between tables.

A **FOREIGN KEY** is a field (or collection of fields) in one table, that refers to the **PRIMARY KEY** in another table.

## SQL FOREIGN KEY on CREATE TABLE

The following SQL creates a **FOREIGN KEY** on the "PersonID" column when the "Orders" table is created:

```
CREATE TABLE Orders (
    OrderID int NOT NULL PRIMARY KEY,
    OrderNumber int NOT NULL,
    PersonID int FOREIGN KEY REFERENCES Persons(PersonID)
);
```

To allow naming of a **FOREIGN KEY** constraint, and for defining a **FOREIGN KEY** constraint on multiple columns, use the following SQL syntax:

```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
        REFERENCES Persons(PersonID)
);
```

## SQL FOREIGN KEY on ALTER TABLE

To create a **FOREIGN KEY** constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL:

```
ALTER TABLE Orders
```

```
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

To allow naming of a **FOREIGN KEY** constraint, and for defining a **FOREIGN KEY** constraint on multiple columns, use the following SQL syntax:

```
ALTER TABLE Orders  
ADD CONSTRAINT FK_PersonOrder  
FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

## DROP a FOREIGN KEY Constraint

To drop a **FOREIGN KEY** constraint, use the following SQL:

```
ALTER TABLE Orders  
DROP CONSTRAINT FK_PersonOrder;
```

## **Not Null Constraint**

## JLL on CREATE TABLE

, ensures that the "ID", "LastName", and "FirstName" columns will NOT accept NULL values when the "Persons" table

```
Persons (
    ,
    FirstName nvarchar(255) NOT NULL,
    LastName nvarchar(255) NOT NULL,
```

---

## JLL on ALTER TABLE

NULL constraint on the "Age" column when the "Persons" table is already created, use the following SQL:

```
sons
age int NOT NULL;
```

### DEFAULT Constraint

دھ بیبی default value لو الیوزر مادخلش داتا بحط قيمه افتراضيه ليها وطريقتها  
انی بكتب کلمة default بعد تعريف العمود في الجدول القيمه الافتراضيه تقدر  
تحطها بايدك او تستخدمن function يجيها لك

هنعمل داتا بيز جديده

```
Create database DB3;
```

هنعمل جدول اسمه persons وبنخلي ال city بتاعها amman default

```

use DB3;

CREATE TABLE Persons(
    ID INT NOT NULL ,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255) NOT NULL,
    Age int,
    City varchar(255) DEFAULT 'Amman'
);

```

لو مدخلتش قيمة للمدينه وجيit قفلت الجدول وفتحته تاني هتلaciها اتحولت بقت  
عمان

	LastName	FirstName	Age	City
	Abu-Hadhud	Mohamed	45	NULL
.	NULL	NULL	NULL	NULL

	LastName	FirstName	Age	City
	Abu-Hadhud	Mohamed	45	Amman
L	NULL	NULL	NULL	NULL

وطبعا تقدر تعديلها  
عاوزين نعمل جدول لل orders ونخلي التاريخ بتاعه هو تاريخ السيستم

```

CREATE TABLE Orders)
ID INT NOT NULL ,
OrderNumber int not null,
OrderDate date default GETDATE());

```

	ID	OrderNumber	OrderDate
1	1	1223123	NULL

ID	OrderNumber	OrderDate
1	1223123	2023-08-29

طيب هنحذف الجدولين ونعمل جدول ال PERSONS من غير default

```
drop table Persons;
drop table Orders;

CREATE TABLE Persons(
ID INT NOT NULL ,
LastName varchar(255) NOT NULL,
FirstName varchar(255) NOT NULL,
Age int,
City varchar(255)
);
```

عاوزين نرجع ال DEFAULT تاني

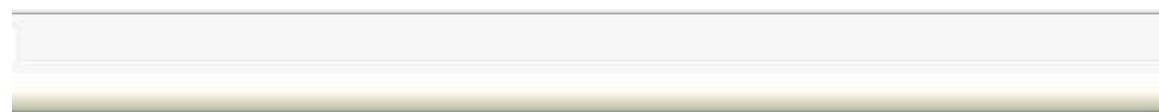
بنكتب ADD CONSTRAINT ونحط اسم لـ constraint عشان تقدر تحذفها  
بعدين لو حبيت دي وبعدين نكتب default ونحط القيمه وبعدين for ونكتب اسم العمود

```
ALTER TABLE Persons
ADD CONSTRAINT df_City
Default 'Amman' FOR City;
```

احذف ال constraint اللي عملتها

```
ALTER TABLE Persons
DROP CONSTRAINT df_City;
```

تقدر تعدها من هنا



or Binding

City
Yes
varchar
'Amman'
---

## SQL DEFAULT Constraint

The **DEFAULT** constraint is used to set a default value for a column.

The default value will be added to all new records, if no other value is specified.

## SQL DEFAULT on CREATE TABLE

The following SQL sets a **DEFAULT** value for the "City" column when the "Persons" table is created:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255) DEFAULT 'Amman'
```

The **DEFAULT** constraint can also be used to insert system values, by using functions like `GETDATE()`:

```
CREATE TABLE Orders (
    ID int NOT NULL,
    OrderNumber int NOT NULL,
    OrderDate date DEFAULT GETDATE()
```

---

## DEFAULT on ALTER TABLE

To add a **DEFAULT** constraint on the "City" column when the table is already created, use the following SQL:

```
TABLE Persons  
CONSTRAINT df_City  
    DEFAULT 'Amman' FOR City;
```

## Add a DEFAULT Constraint

To add a **DEFAULT** constraint, use the following SQL:

```
TABLE Persons  
CONSTRAINT df_City;
```

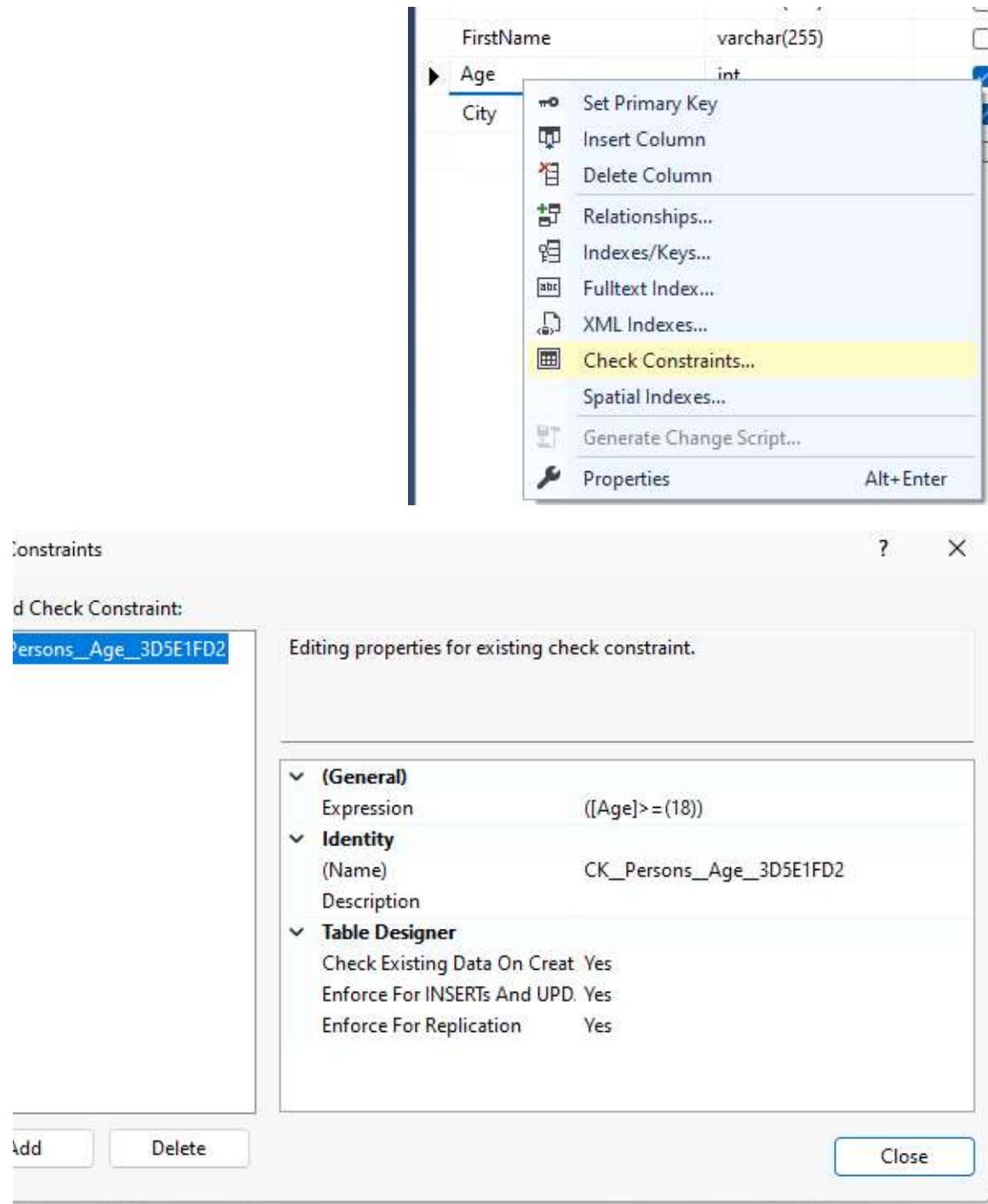
### Check Constraint

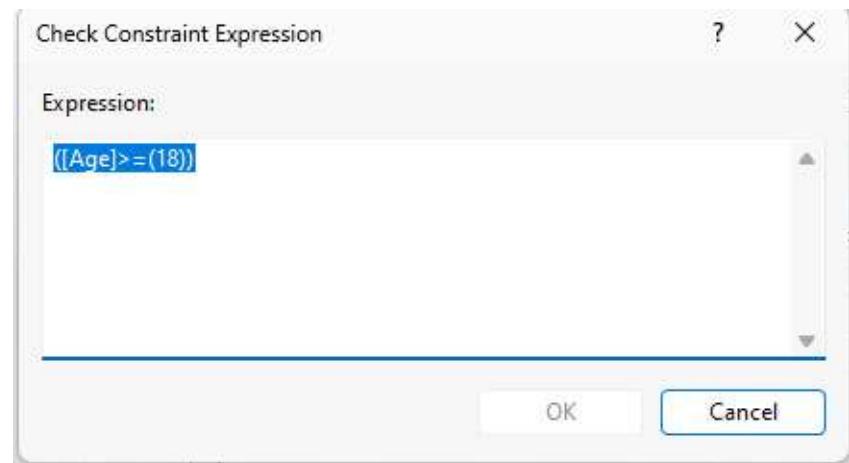
لـ عـاـيـزـ اـحـطـ قـيـوـدـ عـالـدـاـتـاـ الـيـ هـتـدـخـلـ اـنـهـ تـكـوـنـ مـثـلاـ اـكـبـرـ اوـ اـصـغـرـ مـنـ قـيـمـةـ مـعـيـنـهـ

طـرـيـقـتـهـ بـكـتـبـ **check** وـبـعـدـيـنـ الشـرـطـ  
عاـوـزـ اـحـطـ شـرـطـ لـلـسـنـ يـكـوـنـ 18ـ اوـ اـكـبـرـ

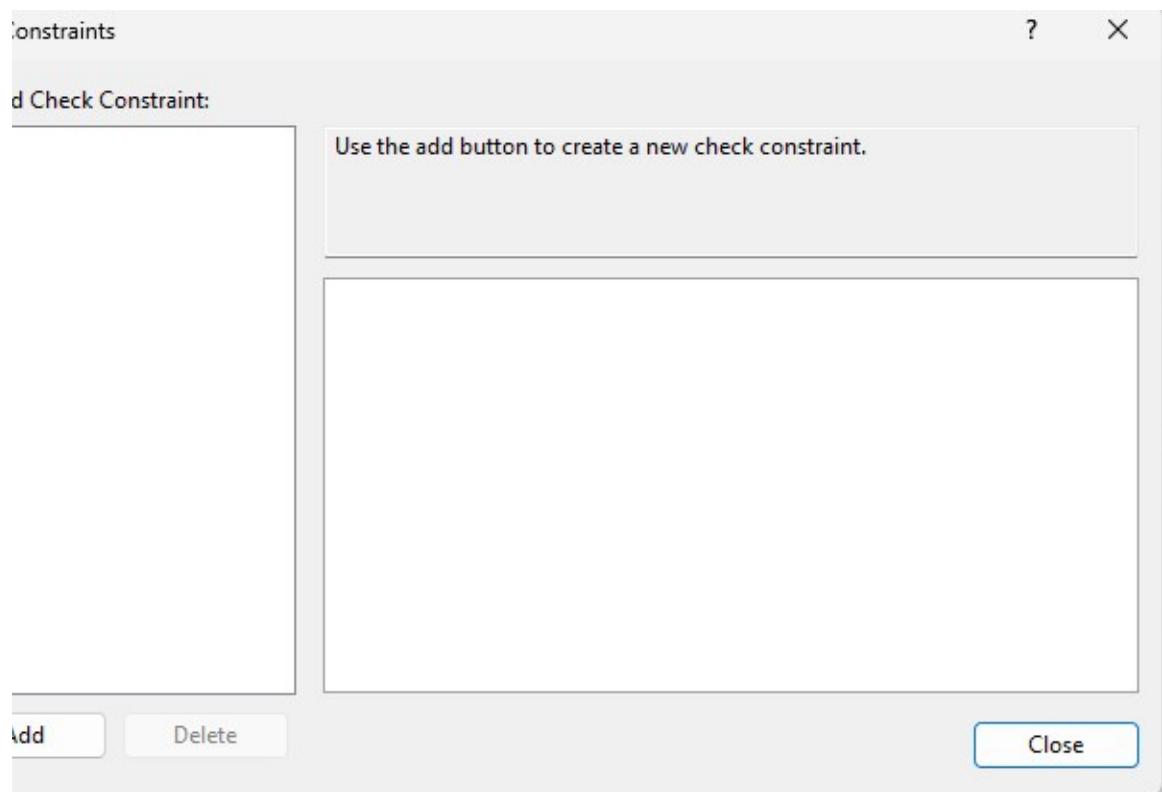
```
CREATE TABLE Persons)  
ID INT NOT NULL ,  
LastName varchar(255) NOT NULL,  
FirstName varchar(255) NOT NULL,  
Age int check(Age>=18),  
City varchar(255)  
);
```

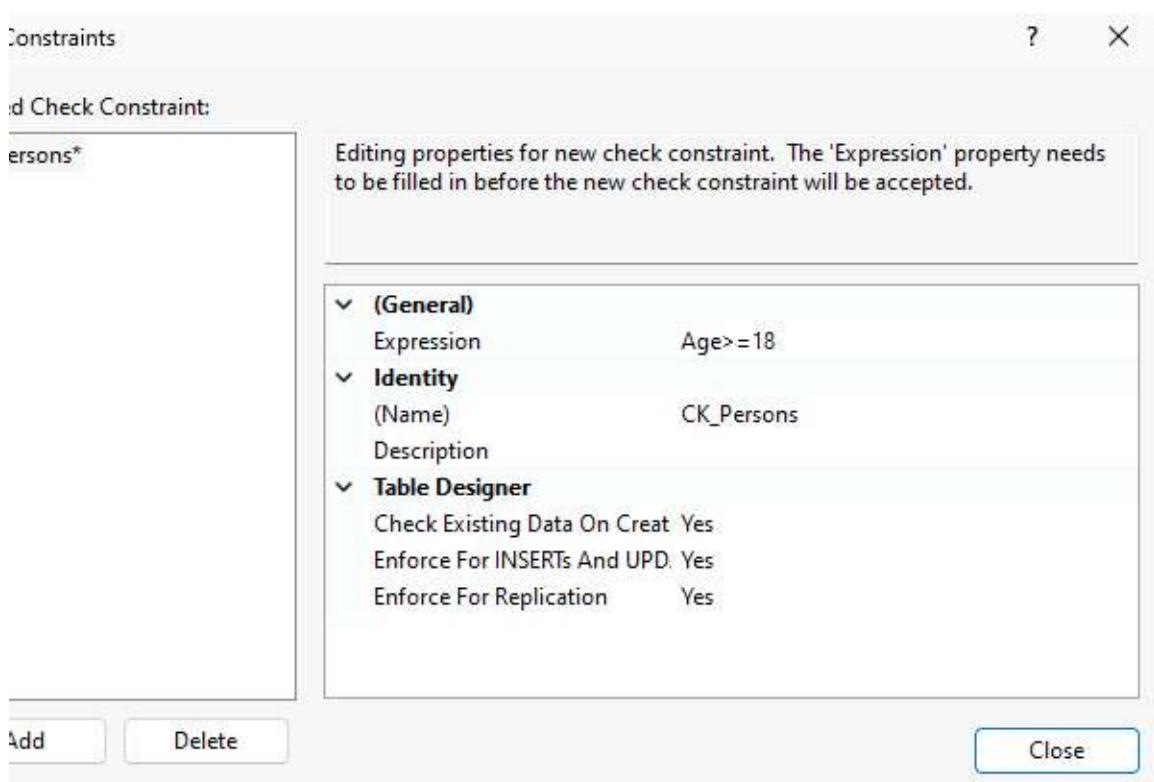
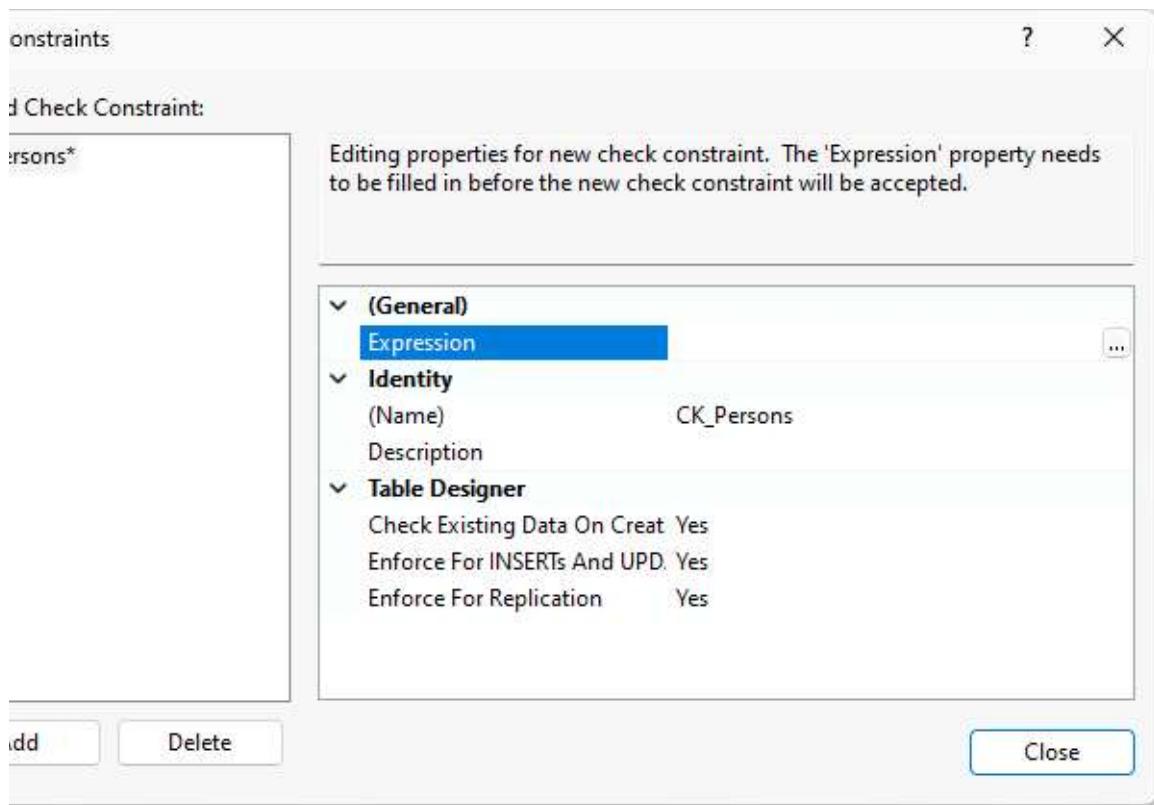
لـ عـاـيـزـ اـعـمـلـهـ فـيـ الـدـيـزاـيـنـ





حذفتها وعملتها تاني





احذف الجدول تاني

```
drop table Persons
```

عاوزين نعمل الجدول تاني بس هنضيف constraint نقول فيها ان ال city بتاعتتها تكون عمان اجباري وال عمر اكبر من او يساوي 18

في نفس ال constraint

```
CREATE TABLE Persons)
ID INT NOT NULL ,
LastName varchar(255) NOT NULL,
FirstName varchar(255) NOT NULL,
Age int,
City varchar(255),
constraint chk_Persons check(Age>=18 AND City='Amman')
);
```

عايز احذف ال constraint

```
ALTER TABLE Persons
drop constraint chk_Persons
```

## SQL CHECK Constraint

The **CHECK** constraint is used to limit the value range that can be placed in a column.

If you define a **CHECK** constraint on a column it will allow only certain values for this column.

If you define a **CHECK** constraint on a table it can limit the values in certain columns based on values in other columns in the row.

## SQL CHECK on CREATE TABLE

The following SQL creates a **CHECK** constraint on the "Age" column when the "Persons" table is created. The **CHECK** constraint ensures that the age of a person must be 18, or older:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int CHECK (Age>=18)
);
```

To allow naming of a **CHECK** constraint, and for defining a **CHECK** constraint on multiple columns, use the following SQL syntax:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255),
    CONSTRAINT CHK_Person CHECK (Age>=18 AND City='Amman')
);
```

## DROP a CHECK Constraint

To drop a **CHECK** constraint, use the following SQL:

```
ALTER TABLE Persons  
DROP CONSTRAINT CHK_Person;
```

### **Unique Constraint**

ال unique constraint مش بتقبل تكرار بس بيسمح لك تدخل null مره واحده  
بس في العمود وطريقتها انك تكتب كلمة unique جنب العمود

```
CREATE TABLE Persons)  
ID INT unique,  
LastName varchar(255) NOT NULL,  
FirstName varchar(255) ,  
Age int,  
City varchar(255),  
);
```

عايز احط ال unique علي عمودين

```
CREATE TABLE Persons)  
ID INT not null,  
LastName vanchar(255) NOT NULL,  
FirstName varchar(255) ,  
Age int,  
City varchar(255),  
  
constraint UC_UNIQUE UNIQUE(ID,LastName)  
);
```

لو عايز اعمله في ال alter table

```
alter table Persons  
add unique(ID);
```

## SQL UNIQUE Constraint

The **UNIQUE** constraint ensures that all values in a column are different.

Both the **UNIQUE** and **PRIMARY KEY** constraints provide a guarantee for uniqueness for a column or set of columns.

A **PRIMARY KEY** constraint automatically has a **UNIQUE** constraint.

However, you can have many **UNIQUE** constraints per table, but only one **PRIMARY KEY** constraint per table.

## SQL UNIQUE Constraint on CREATE TABLE

The following SQL creates a **UNIQUE** constraint on the "ID" column when the "Persons" table is created:

### SQL Server:

```
CREATE TABLE Persons (  
    ID int NOT NULL UNIQUE,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int  
);
```

To name a **UNIQUE** constraint, and to define a **UNIQUE** constraint on multiple columns, use the following SQL syntax:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CONSTRAINT UC_Person UNIQUE (ID,LastName)
);
```

## SQL UNIQUE Constraint on ALTER TABLE

To create a **UNIQUE** constraint on the "ID" column when the table is already created, use the following SQL:

```
ALTER TABLE Persons
ADD UNIQUE (ID);
```

To name a **UNIQUE** constraint, and to define a **UNIQUE** constraint on multiple columns, use the following SQL syntax:

```
ALTER TABLE Persons
ADD CONSTRAINT UC_Person UNIQUE (ID,LastName);
```

## DROP a UNIQUE Constraint

To drop a **UNIQUE** constraint, use the following SQL:

**SQL Server :**

**ALTER TABLE** Persons

**DROP CONSTRAINT** UC\_Person;

### SQL Index

ال indexes هيا طريقة لترتيب الداتا في الجدول بطريقه معينه

لما اليوزر بيدخل داتا في الجدول وانا مثلا عامل ال id ك auto increment هلاقيه  
بيرقم ال records بالترتيب اللي اليوزر دخلها بيه وال index هنا اسمه cluster  
يعني مجمد وبيكون اولويته لل primary key وب يكون اسرع في الاسترجاع لأن  
الدادا مرتبه بحسب ماليوزر دخلها

طيب لو انا عايزة ارتبعها حسب ال last name مثلا

هنا بتكتب جملة sql لما بتنفذها بيروح يعمالك جدول ماتقدر تشويفه ولا توصله  
وبتحيط الداتا فيه مرتبه حسب ال last name

فلما تروح تعمل query تور علي اسم محمد مثلا بيروح عالجدول الجديد اللي هو  
عمله اللي الداتا في مرتبه ابجديا حسب ال last name وبيدور علي محمد  
ويرجعهولك وهنا السرعه بتكون اضعاف السرعه لو كنت استعملت الجدول الأصلي

هنا انا بشوف اكتر اعمده بعمل عليها بحث وبروح احط عليها index  
ماينفعش تستحللي الموضوع وتعملها علي كل الاعمدہ لانه ال index بيبطأ عملية ال  
insert وال update لانه بيروح يعمالها مرتين مره في الجدول الأصلي ومره في الجدول  
المرب حسب ال index

طريقته انك تكتب create index وتحت اسماً ليه براحتك وبعد حين تكتب on وبعدها اسم الجدول وتفتح قوسين تحط فيهم اسم العمود او الاعمد

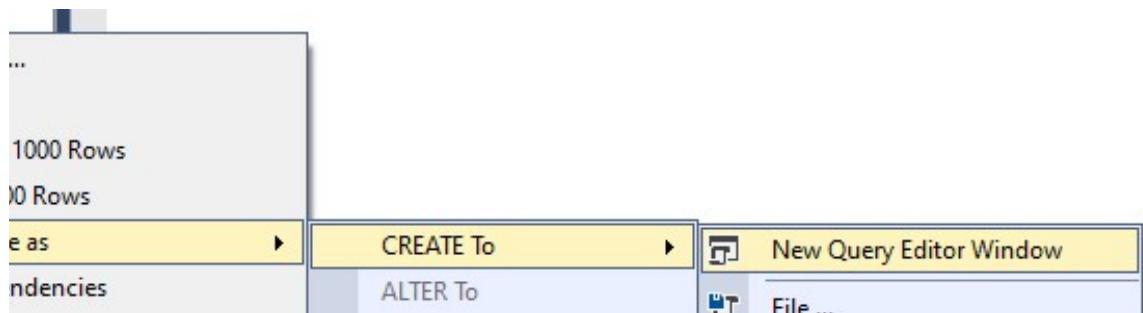
تعالي نعمل جدول

```
drop table Persons

CREATE TABLE Persons(
ID INT not null primary key,
LastName varchar(255) NOT NULL,
FirstName varchar(255) ,
Age int,
City varchar(255),
);
```

بمجرد اني حددت ال primary key وشغلت الكود هوا راح عمل ال clustered index

ولو عايز تعملها كليك يمين على الجدول



```

D..16DFG\Ahmed (58) => X [ SQLQuery1.sql - D..16DFG\Ahmed (74)* ]
3]

Object: Table [dbo].[Persons]      Script Date: 8/29/2023 2:58:29 PM ***
[NULLS ON

IDENTIFIER ON

TABLE [dbo].[Persons](
    id [int] NOT NULL,
    fName [varchar](255) NOT NULL,
    lName [varchar](255) NULL,
    age [int] NULL,
    city [varchar](255) NULL,
    KEY CLUSTERED
        id ASC
    PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON
    PRIMARY
)
```

هنا هو رتب ال id تصاعديا تقدر تخلية تنازليا وعمله  
 cluster index last name index لـ

```

create index idx_lastname
on Persons(LastName);
```

عاوزين نعمل واحد تاني لـ last name fisrt وال

```

CREATE INDEX idx_pname
ON Persons (LastName, FirstName);
```

عايز احذف ال index الاولاني

```
drop index Persons.idx_lastname
```

عايز احذف الثاني

```
drop index Persons.idx_pname
```

لو عايز اعمل INDEX عن طريق الماوس

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer, a database named 'temp' is selected. A table named 'Persons' is open in the 'Table Designer'. The 'LastName' column is currently selected, indicated by a blue border. A context menu is open over this column, listing several options: 'Set Primary Key', 'Insert Column', 'Delete Column', 'Relationships...', 'Indexes/Keys...', and 'Fulltext Index...'. The 'Indexes/Keys...' option is highlighted with a yellow background. Below this, a 'Keys' dialog box is displayed. The title bar of the dialog box says 'Editing properties for existing primary/unique key or index.' On the left side of the dialog, there is a tree view with nodes like '(General)', 'Identity', and 'Table Designer'. Under '(General)', the 'Columns' setting is 'ID (ASC)'. Under 'Identity', the '(Name)' setting is 'PK\_Persons\_3214EC272899B6F8'. Under 'Table Designer', the 'Create As Clustered' setting is 'Yes'. At the bottom of the dialog, there are buttons for 'dd', 'Delete', and 'Close'.

## SQL CREATE INDEX Statement

The **CREATE INDEX** statement is used to create indexes in tables.

Indexes are used to retrieve data from the database more quickly than otherwise.

The users cannot see the indexes, they are just used to speed up searches/queries.

**Note:** Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

### CREATE INDEX Syntax

Creates an index on a table. Duplicate values are allowed:

```
CREATE INDEX index_name
ON table_name (column1, column2, ...);
```

### CREATE UNIQUE INDEX Syntax

Creates a unique index on a table. Duplicate values are not allowed:

```
CREATE UNIQUE INDEX index_name
ON table_name (column1, column2, ...);
```

## CREATE INDEX Example

The SQL statement below creates an index named "idx\_lastname" on the "LastName" column in the "Persons" table:

```
CREATE INDEX idx_lastname  
ON Persons (LastName);
```

If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

```
CREATE INDEX idx_pname  
ON Persons (LastName, FirstName);
```

## DROP INDEX Statement

The **DROP INDEX** statement is used to delete an index in a table.

```
DROP INDEX table_name.index_name;
```

## SQL Server Clustered Index and Primary key constraint

When you create a table with a [primary key](#), SQL Server automatically creates a corresponding clustered index that includes primary key columns.

Clustered Index is much faster than normal Index.

## **Normalization**

### **What is Normalization?**

ال normalization هيا عباره عن تقسيم الداتا لاجزاء اصغر عشان تقدر تتحكم فيها (فرق تسد) عشان مايكونش فيه تكرار او تعقيد في الداتا

ال normalization بتحقق عن طريق ال normal forms وده عباره عن مجموعه شروط لو حققتها تكون حققت ال normalization وده بيكون من :-

-: form First normal •

-: form Second normal •

-: form Third normal •

third normal form :- وده يعتبر تطوير لـ form normal codd Boyce •

## **is Normalization?**

Normalization is the process of organizing data in a database to reduce redundancy and improve data consistency.

In other words, it is the process of breaking down a larger database into smaller, more manageable pieces, while ensuring that the data is properly organized and free from redundant information.

Normalization is achieved by applying a set of rules, called Normal forms, to the database tables. The higher the normal form, the more rules are applied, and the more normalized the database becomes.

The goal of normalization is to eliminate data anomalies, such as update or inconsistent data, which can lead to errors and inconsistencies in the database.

A normalized database is easier to maintain, update and modify, and can be queried more efficiently.

## Forms



### **First Normal Form 1NF**

هنا بيقولك مافيش أي normal form يقدر يتحقق من غير ما تحقق الشروط بتاعت  
ال first normal form  
الشروط هيا :-

- لازم يكون عندك primary key لكل record
- انه العمود الواحد بيمثل قيمة واحدة ملينفعش مثل values Atomic
- تحط رقمين هاتف في نفس المكان
- كل عمود يكون ليه اسم مایتكررش

## 1st Normal Form (1NF)

1. A primary key: A unique identifier for each record in the table.
2. **Atomic** values: Each column should contain only a single value, and each value should be indivisible.
  - Note: Here, **atomicity** states that a single cell cannot hold multiple values. It must hold only a single-valued attribute.
  - The First normal form disallows the multi-valued attribute, composite attribute, and their combinations.
3. No repeating groups: Each column should have a distinct name, and there should be no repeating groups of columns.

Customer Table:

Customer ID	Name	Address	Order ID
1	John	123 Main Street	1, 2, 3
2	Mary	456 Elm Street	4, 5, 6

Order Table:

Order ID	Customer ID	Product	Quantity
1	1	Item 1	3
2	1	Item 2	1
3	1	Item 3	2
4	2	Item 4	4
5	2	Item 5	1
6	2	Item 2	2

In the above example, the customer table was not in 1NF because the order ID column contained multiple values for each record. By creating a separate table for orders and linking it to the customer table with a foreign key, we have normalized the database to 1NF.

ing the First Normal Form, you achieve atomicity, and also every column has unique values.

## Second Normal Form 2NF

ال second normal form يعتمد على ال first normal form

الشروط :-

- تكون محقق ال first normal form

• انه كل عمود في الجدول تقدر ترجعه عن طريق key primary خاص بيه وكامل يعني لو فيه dependencies primary key معتمد من عمودين او عمود واحد او اكتر لازم باقي الاعمدہ في الجدول تكون معتمدہ عالمفتاح کله يعني ماينفعش تيجي تقول انه العمود الفلاني اقدر اجي به من العمود رقم 1 بينما عندك ال primary key معمول من العمودين 1 و 2 اللي هوا شيء غير منطقي

## 2<sup>nd</sup> Normal Form (2NF)

Second Normal Form (2NF) is a further level of database normalization that builds on the First Normal Form (1NF) rules.

It requires that each non-key column in a table be functionally dependent on the entire primary key, not just a part of it.

To satisfy the requirements of 2NF:

1. A table must first be in 1NF, and then have:
2. No partial dependencies: Each non-key column in the table must be fully dependent on the entire primary key.



**2 NF**  
Second  
Normal  
Form

Example  
Tables

For example, consider a table that contains information about courses and the students who have taken them:

Course Code	Course Name	Student ID	Student Name	Grade
101	Biology	001	John	A
101	Biology	002	Mary	B
102	Physics	001	John	C
102	Physics	003	Tom	A

In this table, the primary key is the combination of Course Code and Student ID. However, the Course Name column depends only on the Course Code, and not on the Student ID, which violates the rules of 2NF. To bring this table to 2NF, we would separate the Course Name column into a separate table:

Course Table:

Course Code	Course Name
101	Biology
102	Physics

Enrollment Table:

Course Code	Student ID	Grade
101	001	A
101	002	B
102	001	C
102	003	A

In this example, we have split the original table into two tables, each with its own primary key. The Course Table now contains only information about courses, while the Enrollment Table contains information about the students enrolled in each course. This satisfies the requirements of 2NF.

### Third Normal Form 3NF

ال second normal form بيعتمد اعتماد كلي على ال third normal form  
وبالتالي على ال first normal form

وهو level اعلى من الشروط

الشروط :-

• تكون محقق ال second normal form وال first normal form

• انه كل عمود يكون معتمد فقط على ال dependency No transitive  
ولا يعتمد على عمود اخر primary key

يعني باختصار لو لقيت انه فيه داتا عندك هتكرر حطها في جدول تاني وكل عمود

بيمثل قيمه واحده بس وده اللي كلنا بنعمله



## 3<sup>rd</sup> Normal Form (3NF)

Third Normal Form (3NF) is a higher level of database normalization that builds on the rules of First Normal Form (1NF) and Second Normal Form (2NF). It requires that each non-key column in a table be dependent only on the primary key, and not on any other non-key columns.

To satisfy the requirements of 3NF:

1. Table must first be in 1NF and 2NF, and then have:
2. No transitive dependencies: Each non-key column in the table must be dependent only on the primary key, and not on any other non-key columns.



**Books**

Book ID	Book Title	Author Name	Author Email
1	Pride and Prejudice	Jane Austen	jausten@example.com
2	1984	George Orwell	gorwell@example.com
3	Emma	Jane Austen	jausten@example.com
4	Animal Farm	George Orwell	gorwell@example.com
5	Sense and Sensibility	Jane Austen	jausten@example.com

**Authors Table:**

Author ID	Author Name	Author Email
1	Jane Austen	jausten@example.com
2	George Orwell	gorwell@example.com

**Books Table:**

Book ID	Book Title	Author ID
1	Pride and Prejudice	1
2	1984	2
3	Emma	1
4	Animal Farm	2
5	Sense and Sensibility	1

In this example, we have split the original table into two tables, each with its own primary key. The Authors Table now contains only information about authors, while the Books Table contains information about the books and the authors who wrote them. This satisfies the requirements of 3NF.

## Boyce Codd normal form (BCNF, ( 4NF and 5NF

BCNF , 4NF, and 5 NF will be discussed in Database Level 2.

بالنسبة للNormalization انتمن لن تحتاجوا اكثرا من 3NF باقي الانواع هي وضعت لمعالجة الاخطاء في الديزائن  وهذا الاخطاء  
لطالما استخدمنتم سياسة سياسة فرق تسد

**Course Completed**  
**Message**

تم بحمد الله الانتهاء من هذا الكورس

لا تنسى مواصلة التدريب بشكل دائم

سيكون هناك كورس تطبيقي كامل على جميع ما تم اخذه في هذا الكورس

كل التوفيق للجميع

