Database (Fundmentals)

Course Content

- **Lec 1**
    1. DB Life Cycle
    2. DB Design
    3. ERD
    4. File Based System
    5. DB Systems

- **Lec 2**
    1. DB Mapping
    2. DB Scheme
    3. SQL
    4. Create DB In MSSQL

- **Lec 3**
    1. Joins
    2. Normalization

- **Lec 4**
    1. Aggregate Functions
    2. Grouping
    3. Union – Subqueries
    4. EERD

- **Lec 5**
    1. DB Engine
    2. Service
    3. Ranking Function
    4. Transact-SQL

Lec 1

1. **Define Data – Information – Database – Database System - DBMS ?**

   Data: refers to a collection of **facts** and statistics that are gathered for reference or analysis.

   Information: is a **collection of data** that has been processed and organized in a way that makes it useful and meaningful.

   Database: A collection of related data that is organized into tables.

   Database System: a system that contains databases. (Software + Database)

   DBMS: Database Management System is a software package that facilitate the creation of database Ex.(SQL Server)

2. **Explain Database Life Cycle?**
   1- **Analysis** produces (Requirment doc) | done by "System Analyst"
   2- **DB Design** produces (ERD) | done by "Data base designer"
   3- **DB Mapping** produces (Database Scheme)
   4- **DB Implementation using (RDBMS) SQL Server** produces (the Physical DB)
   5- **Application (GUI Interface)** uses the database to create App
   6- **Client (End-user)**

3. **Explain the file system and its types:**

   File Based System: a method of storing and organizing data using files.

File Based System Types:

1- **Delimited File:** each field in a record is separated by a delimiter character like ' , ' comma.
   - Example: .csv

2- **Fixed Width File:** each field in a record has a fixed width.
   - Example: .txt

4. **Explain Database Evolution?**
   1- **File Based System**
   2- **DBMS** ex. IBM System/38
   3- **RDBMS** ex. MSSQL Server, MySQL, Oracle, POSTGRESQL
   4- **Multidimensional DB** ex. IBM CONGOS, ORACLE OLAP
   5- **No SQL DB** ex. Redis, MongoDB
   6- **GRAPH DB**
   7- **NEWSQL DB**

5. **Explain The Difference between File base system and the Database System?**

Feature	File System	Database System
Data Structure	Unstructured	Structured
Data Access	Direct	Indirect
Data Integrity	Low	High
Data Security	Low	High
Scalability	Limited	Scalable
Data Relationships	Limited	Complex
Data Manipulation	Simple	Complex
Querying	Inefficient	Efficient
Data Redundancy	High	Low
Data Consistency	Low	High
Transaction Support	No	Yes
User Access Control	Limited	Comprehensive
Data Backup and Recovery	Manual	Automated

6. **What is SQL?**

   SQL stands for Structured Query Language, which is a Declarative Programming language for storing, manipulating, and retrieving data stored in a relational database.

7. **What are SQL Commands Categories?**

   1- DDL **"Data Definition Language"**
   - CREATE – ALTER – DROP – RENAME – TRUNCATE - COMMENT

   2- DML **"Data Manipulation Language"**
   - SELECT – INSERT – UPDATE - DELETE – MERGE – CALL

   3- TCL **"Transaction Query Language"**
   - COMMIT - ROLLBACK – SAVEPOINT – SET TRANSACTION

   4- DCL **"Data Control Language"**
   - GRANT – REVOKE

8. **Explain Database files in the system?**

   1- .mdf File: stands for "Master Database File" that stores the actual database

   2- .log File: stands for "Log Database File" that contain the logs for the database

9. **Explain Data Models?**

   1- Logical Model (Meta Data): provide **concepts** ex. ERD "Entity Relationship Diagram"

   2- Physical Model (Actual Stored Data): describes how data is **stored** and **accessed**.

10. Explain Three Level/ Scheme Architecture?

1- **External Level:** represents the **user's view** of the database. It defines how users interact with the database and how they **access** and manipulate data.

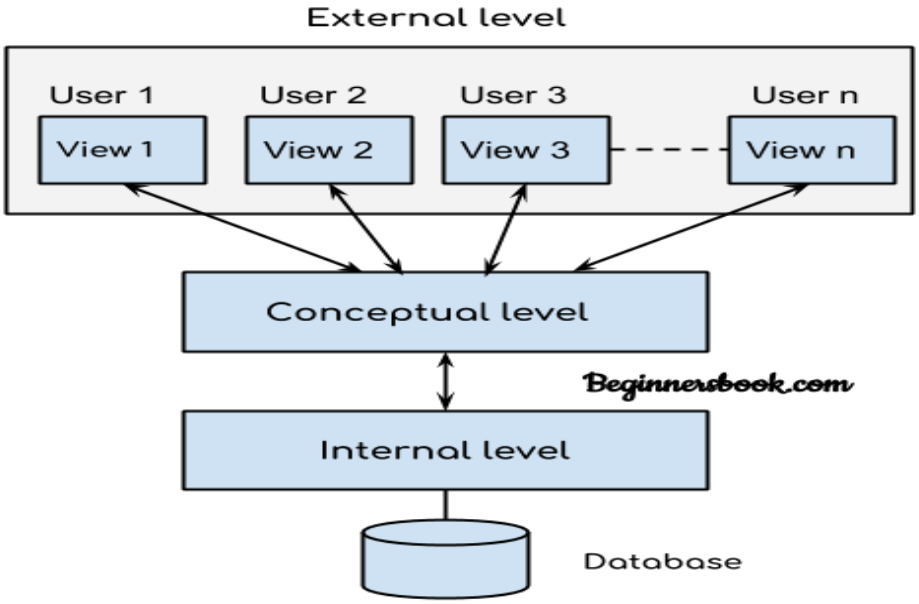
2- **Conceptual Level:** represents the **logical** structure of the database. It defines how data is **organized** and how different data elements relate to each other .

3- **Internal/ Physical Level:** represents the **physical** storage of data in the database. It is responsible for managing how data is **stored, accessed, and retrieved** from the database.

4- **Database (Disk / Storage)**

⇨ **In a 2-tier architecture** is a client-server architecture.

⇨ **In a 3-tier architecture** is a type of web-based application.



11. What is ERD?

ERD stands for "Entity Relationship Diagram."

1. **Entities:** person, place, object, event, concept
   a. **Strong Entity:** An Entity that has a primary key.
   b. **Weak Entity:** An Entity that depends on a strong entity for its existence.
      , Has a (Partial key instead of primary key)

Note: No Entity could exist without Attributes

2. **Attribute:** property or characteristic of an entity
   a. **Simple Attribute:** an attribute that can't be divided.
   b. **Composite Attribute:** An attribute that can be split into components.
   c. **Derived Attribute:** An attribute that exists only in run time. (dashed ellipse)
   d. **Multi-valued Attribute:** contain a list of values (double ellipse)
   e. **Complex Attribute:** Composite Attribute + Multi-valued Attribute

3. **Relationships:** link between entities

Note:

- A relationship may also have attributes. If this attribute is common between these entities.
- Two Entities could have more than one Relationship, but it must express different meaning.

Relation has three Properties:

a. **Degree of Relationships:** number of *Entity_types* in a relationship
   i. **Unary/ Recursive / Self**: between two *instances* of the same entity type
   ii. **Binary**: between the *instances* of two entity types
   iii. **Ternary**: among the *instances* of three or more entity types

b. **Cardinality:** How many *instances* of one entity will or must be connected to a single *instance* from the other entities.
   i. One-One Relationship
   ii. One-Many Relationship
   iii. Many- Many Relationship

c. **Participation:**
   i. **Total Participation: Ex.)** A Car **must** be assigned to particular employee.
   ii. **Partial Participation:** Ex.) An Employee **may** have a car meaning zero or more.

Note:

- any weak entity is a partial participation.
- a partial participation doesn't always mean a weak Entity.

Lec 2

12. What is Database Mapping?

Database Mapping is a set of roles for converting conceptual design "ERD" to logical design "DB Scheme".

1. **Entity *will be* → Table | "Student"**
2. **Attribute *will be* → column | "Name"**
   - Row == record
   - Column crossed with Row is a cell ex. "Mohamed"
3. **Relationship *will be*: foreign key constraints**

Notes:

- foreign key exists on the **M** relation.
- Tables that have the foreign key called **child tables.**
- Tables that have the primary key are called **parent tables.**
- You cannot delete the primary key of parent that exists on a child table.
- Arrow of relationship points to the primary key in (ERD).
- Drag the primary key to the foreign key in (MSSQL Physical ERD).

- **ERD To DB Scheme Mapping (Best Practices)**
  - **Step 1: Mapping of Regular Entity Types** --> **regular table**
    i. Mapping simple attribute --> *regular column*
    ii. Mapping composite attribute --> *regular column**s***
    iii. Mapping multivalued attribute --> (New table + Composite PK of [parent + column])
    iv. Mapping complex attribute --> (New table + Composite PK of [parent + columns])
    v. ***No*** *Mapping for derived attribute*

  - **Step 2: Mapping of Weak Entity Types** --> **regular table** that has (Composite PK [weak entity partial key + regular entity pk])

  - **Step 3: Mapping of Binary 1:1 Relation Types** --> fk in any one of them
  - **Step 4: Mapping of Binary 1: N Relationship Types.** --> fk in the many
  - **Step 5: Mapping of Binary M: N Relationship Types** --> (New table + Composite PK [pk1+pk2+ relation attr])

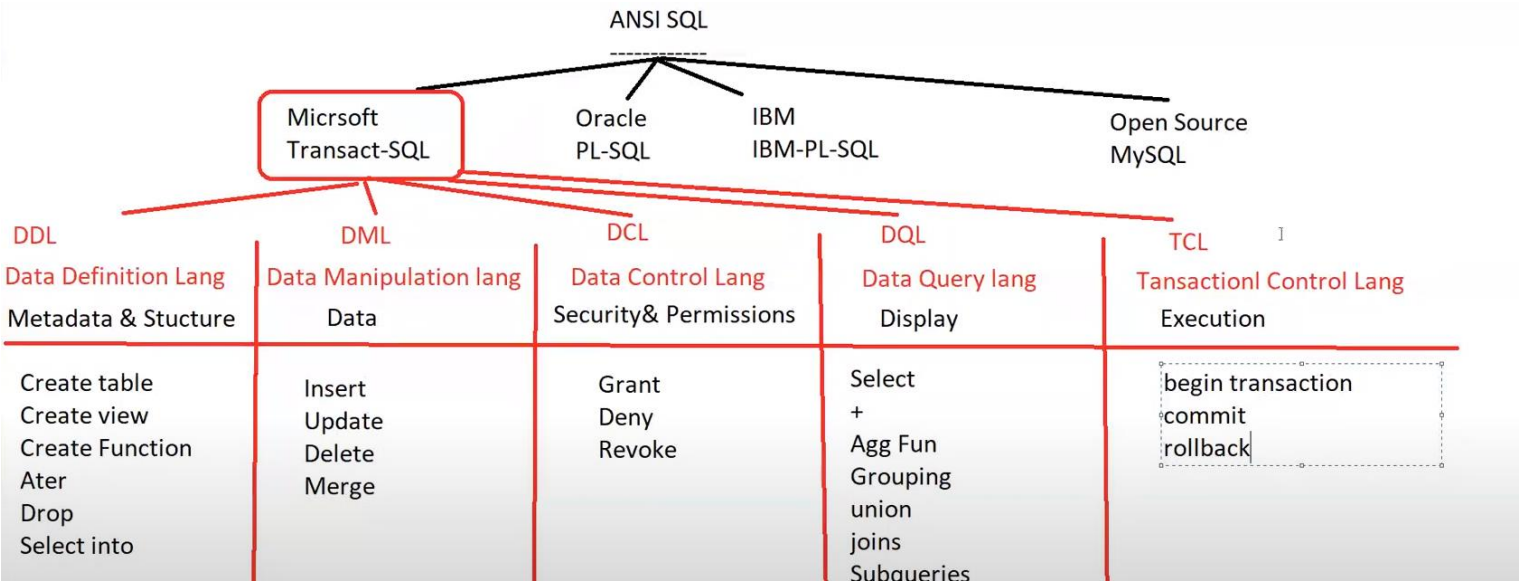
  - **Step 6: Mapping of N-ary Relationship Types.** --> (New table + Composite PK [find your unique combination])

  - **Step 7: Mapping of Unary/Self Relationship.** --> create new fk to the pk of the same table

13. What are the different types of keys in an attribute?

1- **Candidate Key:** One or more attributes that can be elected as a primary key
2- **Primary Key:** [Unique & Not Null] attribute
3- **Unique Key:** different key that isn't replicated in the table.
4- **Foreign Key:** a set of attributes in a table that refers to the primary key of another table
5- **Composite Primary Key:** Combining two or more attributes to form a primary key
6- **Partial Key:** a primary key for a weak entity
7- **Super Key**
8- **Sub Key**

14. Explain Different Types of ANSI SQL "American National Standards Institute Structured Query Language"?



15. Explain Database Physical Design?

Physical Design is converting logical design "DB Scheme" into the actual database using SQL.

1- DDL **"Data Definition Language" -->** Meta Data & Structure
   - CREATE – ALTER – DROP – RENAME – TRUNCATE - COMMENT
2- DML **"Data Manipulation Language"**
   - SELECT – INSERT – UPDATE - DELETE – MERGE – CALL
3- DCL **"Data Control Language"**
   - GRANT – REVOKE - DENY
4- DQL **"Data Query Language"**
   - Select + join – grouping – subqueries – Union – Agg Func
5- TCL **"Transaction Query Language"**
   - BEGIN TRANSACTION - COMMIT - ROLLBACK – SAVEPOINT

16. Define Database Schema?

A **database schema** is a **blueprint** that describes how data is organized within a relational database.
It defines the **structure** of the database, including the tables, fields, data types, and relationships.

17. Define Database Constraints?

- **Database constraints** are **rules** or **restrictions** that are applied to the data in a database to ensure **data integrity**, **consistency, and adherence** to business rules
- **Constraints** can be **column leve**l or **table level**.
- You can't delete a parent record that has a child record.

Example:
NOT NULL: Ensures that a column cannot have a NULL value.
UNIQUE: Ensures that all values in a column are different. "Cannot repeat"
PRIMARY KEY: A combination of NOT NULL and UNIQUE. Uniquely identifies each row in a table.
FOREIGN KEY: Prevents actions that would destroy links between tables.
CHECK: Ensures that the values in a column satisfy a specific condition.
DEFAULT: Sets a default value for a column if no value is specified.

18. Some Basic SQL queries?

```
-------------   DDL   -create  - drop  -alter ------------------------------------------

create database MyfirstDB    -- to create new database

drop database MyfirstDB      -- to delete an entire table  data + meta_data

use MyfirstDB

create table employee   -- create table in the MyFirstDB database
(
 eid int Primary key,    -- create table's column/ attribute   pk
 ename varchar(50) not null,
 eadd varchar(20) default 'cairo',
 hiredate date default getdate(), --builtin function   current system date
 salary int default 5000,      -- by default the value will be 5000
 age int,
 Dnum int
)

alter table employee add overtime int           -- edit table attribute by adding
alter table employee alter column overtime bigint   -- edit table attribute by replacing
alter table employee drop column overtime         -- edit table attribute by removing
```

```
---------------------  DML  -insert  -update  -delete-----------------------

insert into employee    -- insert record / row
values(1,'ahmed','alex','1/1/2000',8000,21,NULL)

update employee       -- update the eadd attribute in all rows
       set eadd='mansoura'

update employee        -- update the eadd attribute only where eid=1
       set eadd='aswan'
where eid=1

update employee       -- update the salary attribute in all rows
       set salary+=100

delete from employee    -- delete the entire record/ row  where  eid=1
where eid=1
```

```
----------------------DQL  -select --------------

use ITI

select * from Student   -- select all data in 'Student' table

select * from Student    -- select all data in 'Student' table where st_address='alex'
where st_address='alex'

select st_id,st_fname      -- select only st_id,st_fname attribute's records in 'Student' table
from Student

select st_id,st_fname
from Student
where st_age>25

select *
from Student
where st_fname is NULL



select distinct st_fname  --unique+order
from Student

select st_fname+' '+st_lname as fullname    -- alias name
from Student


select st_fname+' '+st_lname as [full name]  -- alias name   using square brackets to allow spaces in the
names
from Student

select * from Student           -- ordered  default ascending
order by st_age

select * from Student    -- ordered   descending
order by st_age desc

select *
from Student
where st_address='alex' or st_address='mansoura'    -- OR

select *
from Student
where st_address in('alex','cairo','mansoura')     -- Multiple OR

select *
from Student
where st_address='alex' and st_age>22    -- AND


select *
from Student
where dept_id not in(10,30)

select *
from Student
where st_age>20 and st_age <=27

select *
from Student
where st_age between 20 and 27  -- range


select st_fname+' '+convert(varchar(10),st_age)   -- convert int to varchar
from Student
```

19. What are built-in Functions in SQL?

- **getdate()** --> get the current system date
- **convert(*datatype*, *col*)** --> convert the attribute value int another datatype
- **year(*date*)** --> get the year format of a date
- **isnull(*col, replacement_data*)** --> if the attibute is null then it replace it with *replacement_data*
- **coalesce(*data, replacement_data1, replacement_data2, replacement_data3*)**
- **concat(*col*,' ',*col*)** --> concat two attibutes
- **select db_name()** --> returns the current database name
- **select suser_name()** --> returns the user name of the server
- **select lower(st_fname)** --> covert the string to lowercase
- **select upper(st_fname)** --> covert the string to uppercase
- **select substring(st_fname,1,3)** --> get the sub string from first char to third char
- **select len(st_fname)** --> get the length of a string

Lec 3

20. When do we use JOIN Statement?

The **JOIN** statement is used to **combine rows** from two or more tables based on a related column between them .

Student

Sid	Sname	did
1	Ahmed	10
2	Khalid	10
3	Eman	20
4	Omar	NULL

Department

Did	Dname	cdate
10	SD	11/10/2023
20	HR	11/10/2023
30	IS	11/10/2023
40	Admin	11/10/2023

21. What are types of JOINS?

1. **Cross Join / Cartesian product**
   - combines every row from one table with every row from another table, regardless of whether there is a match between the rows.

2. **Inner join / Equi Join:**
   - combines rows from two tables based on a matching value between the two tables

3. **Outer join**
   - I. **Left outer join.**
     - includes all rows from the left table, regardless of whether there is a matching row in the right table.
   - II. **Right outer join**
     - includes all rows from the right table, regardless of whether there is a matching row in the left table.
   - III. **Full outer join**
     - includes all rows from both tables, regardless of whether there is a matching row in the other table.
4. **Self-join**
   - is a join that joins a table to itself.

```sql
-- Cartsian product
select St_Fname,Dept_Name
from Student, Department           --or --> from Student cross join Department

-- equi join
select St_Fname,Dept_Name
from Student s, Department d
where d.Dept_Id = s.Dept_Id
--          PK  =  FK
--          parent  =  child

--inner join     ==> the same as equi join
select St_Fname,Dept_Name
from Student s inner join Department d
on d.Dept_Id = s.Dept_Id

-- outer join
-- left outer join
select St_Fname,Dept_Name
from Student s left outer join Department d
on d.Dept_Id = s.Dept_Id

-- Full outer join = left outer join + right outer join
select St_Fname,Dept_Name
from Student s full outer join Department d
on d.Dept_Id = s.Dept_Id

-- self join
select X.Ename as empname , Y.Ename as supername
from Employee X, Employee Y
where Y.id = X.superid

-- join multiple tables

select st_fname, crs_name, grade
from Student S, Stud_Course , Course C
where S.Std_ID= SC.St_ID and C.Crs_Id = SC.Crs_id

-- another way to join multiple tables

select st_fname,grade,crs_name
from Student s inner join Stud_Course sc
     on s.St_Id=sc.st_id
       inner join
       course c
       on c.crs_id=sc.crs_id

-- join with DML

update Stud_Course
set grade += 10
from Stud_Course C, Student S
where S.St_Id=C.St_Id and S.St_Address='cairo'
--   PK OF ID  =  FK OF ID
```

22. What are *'Like'* Statement Patterns in SQL?

```
select *
from Student
where st_fname like 'a%'  --> start with a

PATERNS
'a%h'       --> start with 'a' ends with 'h'
'%h_'       --> end with any char before end 'h'
'ahm%'      --> start with  ahm
'[ahm]%'    --> start with 'a' or 'h' or 'm'
'[^ahm]%'    --> string that doesn't start with 'a' or 'h' or 'm'
'[a-h]%'    --> start with range from 'a' to 'h'
'[^a-h]%'    --> string that doesn't start with range from 'a' to 'h'
'[(ah)(mo)]%'    --> start with 'ah' or 'mo'
'%[%]'           --> end with '%' character
```

23. What are Data base Design Techniques?

1- ERD

2- Normalization

24. What is Normalization?

- Is a database design technique to **refactor** the database design by removing **redundant data** and **data inconsistency**.
- This technique is usually done by breaking down a single table into two or more tables and defining relationships between those tables.

25. What is Denormalization?

Is a database design technique to improve search performance.

26. Functional Dependency

- is a constraint between two sets of attributes (Columns) in a relation
- A →B "existing of B depending on a value of A"
- Example: SSN -> ENAME EmployeeName depend on SSN
- if all columns depend on the primary key --> this is Good Design

27. What are Functional Dependency types:

1- Full Functional Dependency

- Attribute is fully Functional Dependency on a PK **if its value is determined by the whole PK.**

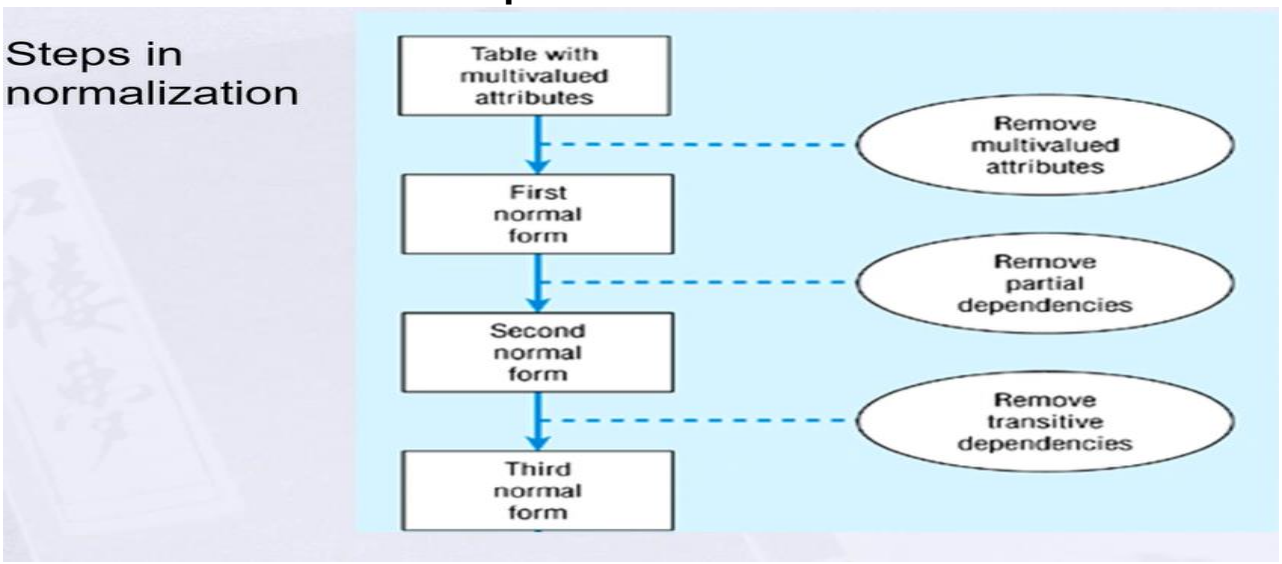
2- Partial Functional Dependency

- Attribute has a Partially Functional Dependency on a PK if **its value is determined by part of the PK(Composite Key)**

3- Transitive Functional Dependency

- Attribute is Transitively Functional Dependency on a table **if its value is determined by another non-key attribute which itself determined by PK.**

28. What are Normalization Steps?



- **First normal form (1NF):** if it contains no multivalued or composite attributes
  - ✓ **It's Done by:** removing repeating groups + multi-valued to a new table "Composite PK"

- **Second normal form (2NF):** if it is in first normal form AND every non-key attribute is fully functionally dependant on the primary key "composite"
  - ✓ **It's Done by:** removing partial dependancies to a new table

- **Third normal form (3NF):** if it is in second normal form AND **no transitive dependencies** (one attribute functionally determines a second, which functionally determines a third)
  - ✓ **It's Done by:** removing transitve dependancies to a new table

29.Normalization Steps Example?

- ✓ **Zero Normal Form:**

Student

SID	SName	Birthdate	City	Zip Code	Subject	Grade	Teacher
1	Ahmed	1/1/1980	Cairo	1010	DB	A	Hany
1	Ahmed	1/1/1980	Cairo	1010	Math	B	Eman
1	Ahmed	1/1/1980	Cairo	1010	WinXP	A	khalid
2	Ali	1/1/1983	Alex	1111	DB	B	Hany
2	Ali	1/1/1983	Alex	1111	SWE	B	Heba
3	Mohamed	1/1/1990	Cairo	1010	NC	C	Mona

- ✓ **1st Normal Form:**

Student

SID	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Cairo	1010

Student_Subject

SID	Subject	Grade	Teacher
1	DB	A	Hany
1	Math	B	Eman
1	WinXP	A	khalid
2	DB	B	Hany
2	SWE	B	Heba
3	NC	C	Mona

✓ **2nd Normal Form:**

Student

SID	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Mansoura	1210

Student_Subject

SID	Subject	Grade
1	DB	A
1	Math	B
1	WinXP	A
2	DB	B
2	SWE	B
3	NC	C

Subject

Subject	Teacher
DB	Hany
Math	Eman
WinXP	khalid
SWE	Heba
NC	Mona

✓ **3rd Normal Form:**

Student

SID	SName	Birthdate	City	Zip Code
1	Ahmed	1/1/1980	Cairo	1010
2	Ali	1/1/1983	Alex	1111
3	Mohamed	1/1/1990	Mansoura	1210

Student_Subject

SID	Subject	Grade
1	DB	A
1	Math	B
1	WinXP	A
2	DB	B
2	SWE	B
3	NC	C

Subject

Subject	Teacher
DB	Hany
Math	Eman
WinXP	khalid
SWE	Heba
NC	Mona

Student_City

City	Zip Code
Cairo	1010
Alex	1111

Lec 4

30. What is Aggregate Function?

In SQL, an **aggregate function** is a function that operates on a set of values and returns **a single value.**

- **AVG()**: Returns the average of a set.
- **COUNT()**: Returns the number of items in a set.
- **MAX()**: Returns the maximum value in a set.
- **MIN()**: Returns the minimum value in a set.
- **SUM()**: Returns the sum of all or distinct values in a set.

Note :

- When you have **aggregate function** and a column you must use group by with that column
- **Aggregate functions** are often used with the **GROUP BY** clause of the SELECT statement **GROUP BY** clause divides the result set into **groups of values**
- the **aggregate function** returns a **single value** for each group.

31. What is difference between having and where?

- **Where:** filter the records from a table that is based on a specified condition
- **Having:** filter the record from the **groups** based on the specified condition. Used with **Aggregate functions.**

32. What is the difference between GROUPBY and HAVING?

- The **GROUP BY** clause is used to **group** rows that have the same values in one or more columns. It is often used with aggregate functions such as SUM, COUNT, AVG, MIN, MAX, etc., to compute summary statistics for each group.

- The **HAVING** clause is used to **filter** groups based on a condition that involves an aggregate function. It is used in conjunction with the **GROUP BY** clause to filter the groups based on the results of the aggregate functions [1].

33. What are Sub-queries?

Subquery is a query that is nested inside another query. It also can be slow.

34. Union Family (union all – union – intersect - except)?

- **Union all:** unifying the result of two ore more queries in one sheet.
- **Union :** unifying the result of two or more queries in one sheet. But without repeating values + ORDERED

- **Intersect:** returns the matching values in the two sheets. Also, without repeating values + ORDERED.
- **Except:** returns the rows in first result set that **are not** in the other result set.

- **Union Rules:**
  - **1-** The number of columns should be the same.
  - **2-** Data types should be the same.

35. Some SQL About (Aggregate functions + Subquery+ Union):

```
use ITI

--------------------------------- AGGREGATE FUNCTION --------------------------

select Sum(salary) -- aggregate function returns one value
from Instructor

Select Max(salary) as Max_val,Min(salary) as Min_Val
from Instructor

select count(st_id),count(*),Count(st_age)
from Student

select avg(st_age)  -- the same as sum(st_age) / count(st_age)
from Student


select sum(st_age)/Count(*)   -- the same as sum(st_age) / count(*)    --> different value
from Student

select avg(isnull(st_age,0))  -- the same as sum(st_age) / count(*)
from Student

--------------------------------- GROUPING ----------------------------

select sum(salary),dept_id   -- error without group by
from Instructor               -- sum of salaries for each dept_id
group by dept_id

select sum(salary),d.dept_id,dept_name    -- ** General Rule --> if there are columns that exist in the
select besides the aggregate functions then you have to group by thos columns
from Instructor i inner join Department d
        on d.Dept_Id=i.Dept_Id
group by d.dept_id,dept_name


select sum(salary),dept_id
from Instructor
where salary>1000
group by dept_id

--------------------------------- GROUPING USING HAVING ----------------------------

select sum(salary),dept_id
from Instructor
group by dept_id
having sum(salary)>30000     -- using having  to  Filter the resulted Group
```

```sql
select sum(salary),dept_id
from Instructor
where salary>1000            -- 'where' can't be associated with aggregate function
group by dept_id
having sum(salary)>50000

select sum(salary)   -- Using Having  without group by the query assumes that the result is one group
from Instructor
having sum(salary)>50000
---------------------------------- SUBQUERY ---------------------------

select *
from Student
where st_age<(select avg(St_Age) from student)   -- where st_age < (23)

select dept_name
from Department
where dept_id in (select distinct dept_id       -- range
                            from Student
                                where dept_id is not null)

select distinct dept_name
from Student S inner join Department d    -- the same as the previous but without subquery
       on d.Dept_Id=s.Dept_Id

select *
from   (select  st_fname+' '+st_lname as fullname
              from Student) as newtable
where fullname='ahmed ali'

--Subqueries +DML


delete from Stud_Course
where st_id in (select st_id from Student
                          where st_address='cairo')

------------------------ Union family ----------------------------------
--union all    union      except   intersect

--Batch
select st_fname as names
from Student
union all          -- combine the two results with each other
select ins_name
from Instructor

select st_fname
from Student
union                -- combine the two results with each other (no repetion)
select ins_name
from Instructor

select st_fname
from Student
intersect          -- result the matching values of the  two results
select ins_name
from Instructor

select st_fname
from Student
except            -- returns the rows in first result set that are not in the other result set.
select ins_name
from Instructor
```

36. Define Datatype in SQL?

1- Numeric Datatype

- bit 1bit 0 or 1
- tinyint 1Byte
- smallint 2Byte
- int 4Byte
- bigint 8Byte

2- Decimal Datatype

- smallmoney 4Byte
- money 8Byte
- real
- float
- dec dec(5,2) ##### . ##

3- Char , string

- char(10) --> fixed length
- varchar(10) --> variable length
- nchar(10) --> fixed length + unicode 'multiple languages'
- nvarchar(10) --> variable length + unicode
- nvarchar(MAX)--> Up to 2GB

4- Data , time

- date MM/DD/YYYY
- time hh:mm:ss
- smalldatetime MM/DD/YYYY hh:mm:00 --> smaller range
- datatime MM/DD/YYYY hh:mm:ss.#### --> bigger range
- datetimeoffset MM/DD/YYYY hh:mm +2:00 --> time zone

5- Binary

- Binary(5) 010010

Note: Image in DB stored as (image path, bit stream)

6- Other Datatype

- Unique identifier
- XML
- Sql_varient

37. What is the difference between Drop - Truncate - Delete ?

1. **DROP**:
   - delete an **entire table** (data + meta data). it cannot be undone/ rolled back.
   - (DDL)

2. **DELETE**:
   - delete **individual rows** from a table (only data)
   - accepts **WHERE** statement
   - can be **rolled back** because of the **log** file
   - can be done on (child & parent) tables
   - doesn't reset the identity
   - (DML)

3. **TRUNCATE**:
   - delete **all of the rows** from a table (only data)
   - similar to the **DELETE** statement, but it is **faster** because it does not check for **foreign key constraints**.
   - Doesn't accept **WHERE** statement
   - Can't be **rolled back** because of the it doesn't have **log** file
   - can be done only on (child) tables
   - resets the identity
   - (DDL)

38. SQL Execution ORDER?

1- from
2- join
3- on
4- where
5- group by
6- having
7- select
8- order by
9- top

39. What is EERD "Enhanced Entity Relationship diagram"?

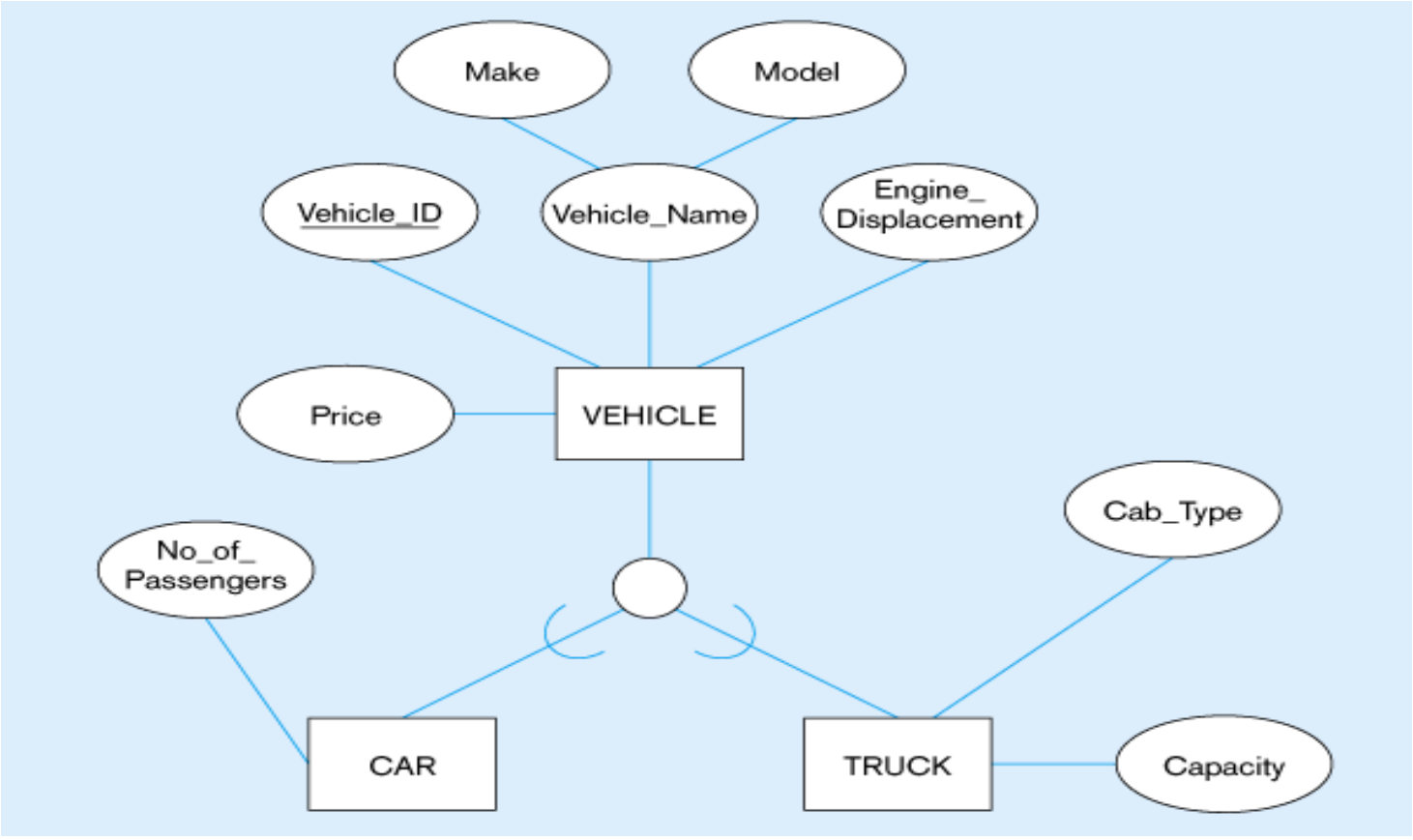
Is an ERD with Inheritance

40. Define subtype, super type entities?

- **Sub type entity:** subgrouping of the entities in an entity type which has attributes that are distinct from those in other subgroupings.
- **Super type entity:** A generic entity type that has a relationship with one or more subtypes.
- **Note: Inheritance must have** (is a) relationship.

41. Define Generalization, Specialization?

- ***Generalization*:** The process of defining a more general entity type from a set of more specialized entity types. **BOTTOM-UP**
- ***Specialization*:** The process of defining one or more subtypes of the supertype and forming supertype/subtype relationships. **TOP-DOWN**



42. What are the Constraints on Supertype:

- ***Completeness Constraints*:** specify whether or not all instances of a supertype entity **must** also be instances of one or more of its subtypes**.**
- **There are two types of completeness constraints:**
  - **Total Specialization** Rule: Yes (double line) --> This constraint specifies that all instances of a supertype entity **must** also be instances of one of its subtypes.
  - **Partial Specialization** Rule: No (single line) --> This constraint specifies that some instances of a supertype entity **may not** be instances of any of its subtypes.

- ***Disjointness Constraints*:** specify whether or not an instance of a supertype entity can be an instance of more than one of its subtypes**.**
- **There are also two types of disjointness constraints:**
  - **Total Disjoint (d):** This constraint specifies that an instance of a supertype entity can only be an instance of one of its subtypes.
  - **Overlap Rule (o):** This constraint specifies that an instance of a supertype entity can be an instance of more than one of its subtypes.

43. Mapping Super type / sub type entities:

1- Each super-type entity and sub-type entity will become a table

2- The primary key of each sub type table will be the same as their super-type table & will be their foreign key at the same time.

3- Check for the Disjointness Constraints *type* Discriminator:

- **if** it's Disjoint (d) **then** Use **Simple** attribute (Subtype discriminator)
- **if** it's Overlap Rule (o) **then** Use **Composite** attribute (Subtype discriminator)

Lec 5

44. What Database versions , Editions?

- **Versions: 1995 – 2000 – 2008 - 2014 - 2022**
- **Editions**
  - **Enterprise:** for large-scale business applications
  - **Standard – Developer:** for small/medium-scale business applications
  - **Bi Edition:** for Bi services
  - **Express:** for learning edition
  - **Azure:** for cloud

Note: MSSQL Server is a fully-RDBMS

45. What Are MSSQL [Services and Application]?

- **Service** (DB engine SQLServer (MSSQLserver) -SSIS - SSRS - SSAS - Data quality service)
- **Applications:**(SSMS - SQLServer Data tools - Data quality client – SQL Server Profiler – SQL Server Tuning Advisor)

46. What are DB Instances [default + named]?

1- **Default Instance:**
- is the instance that is installed and configured automatically when you install SQL Server.
- The default instance name is your pc name.

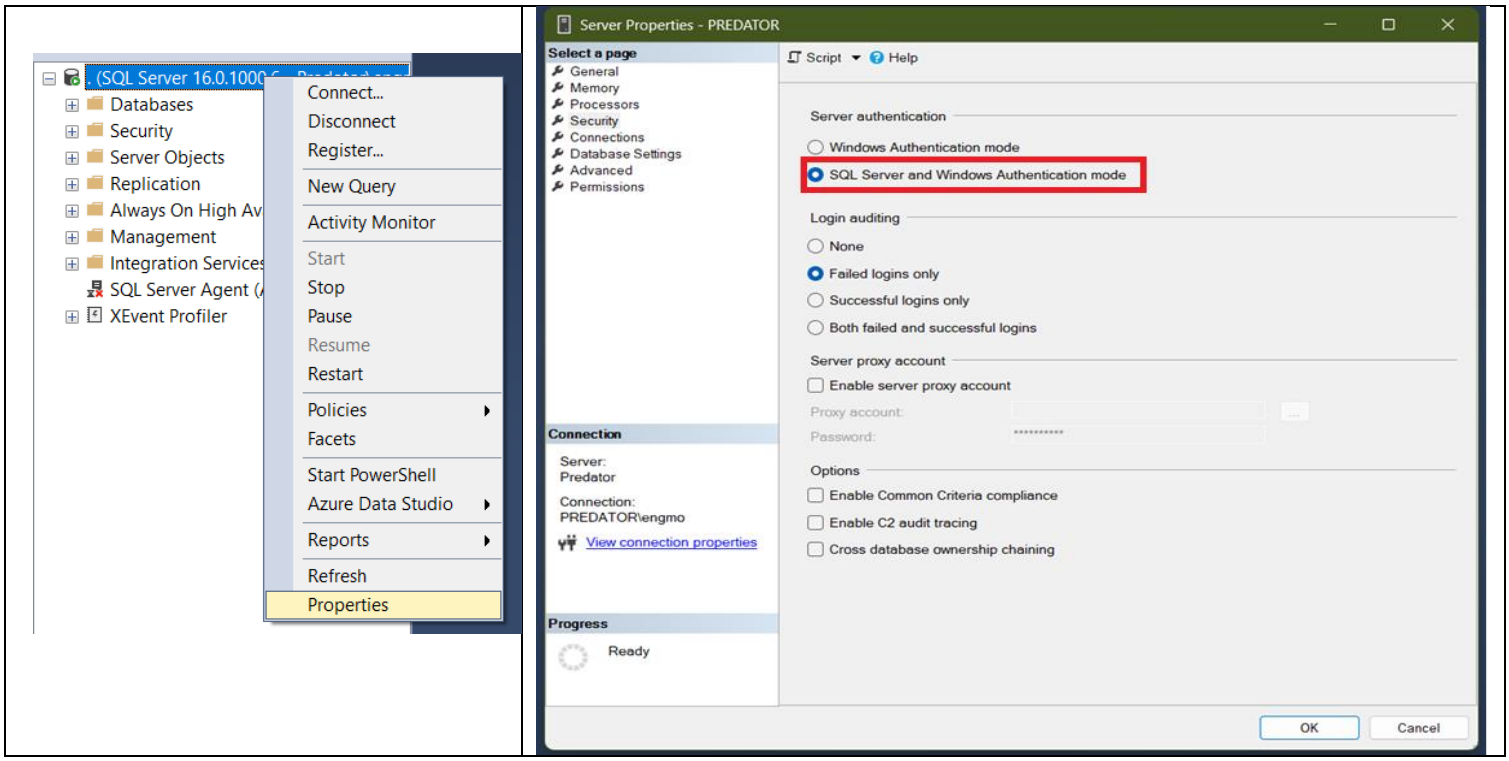
2- **Named Instance**
- is a user-defined instance of SQL Server during the installation process.
- You can install multiple named instances on a single computer.

47. DB Security?

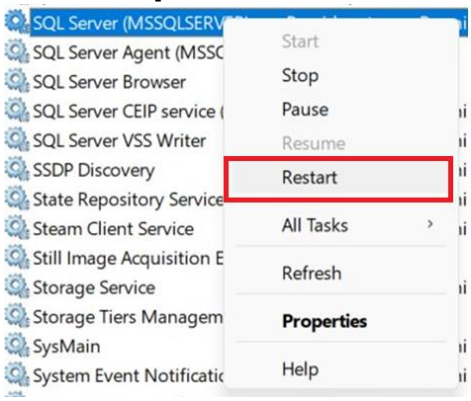
1. **Authentication** is the 'username + password' you enter to access the database.
   - **Windows Authentication:**
     - You don't have to enter username and password.
     - Used for connecting to **local** database.
   - **SQL Server Authentication:**
     - You must enter a username and password.
     - Used for connecting to **remote** database.
2. **Authorization** is the 'permissions' that is given to you by the admin like 'read – edit - delete.'

48. DB Steps to Create SQL Server auth User ?

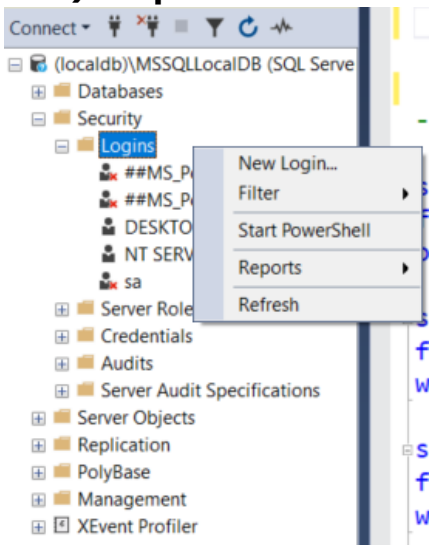
1) **Step 1:** allow windows authentication and SQL server Authentication.

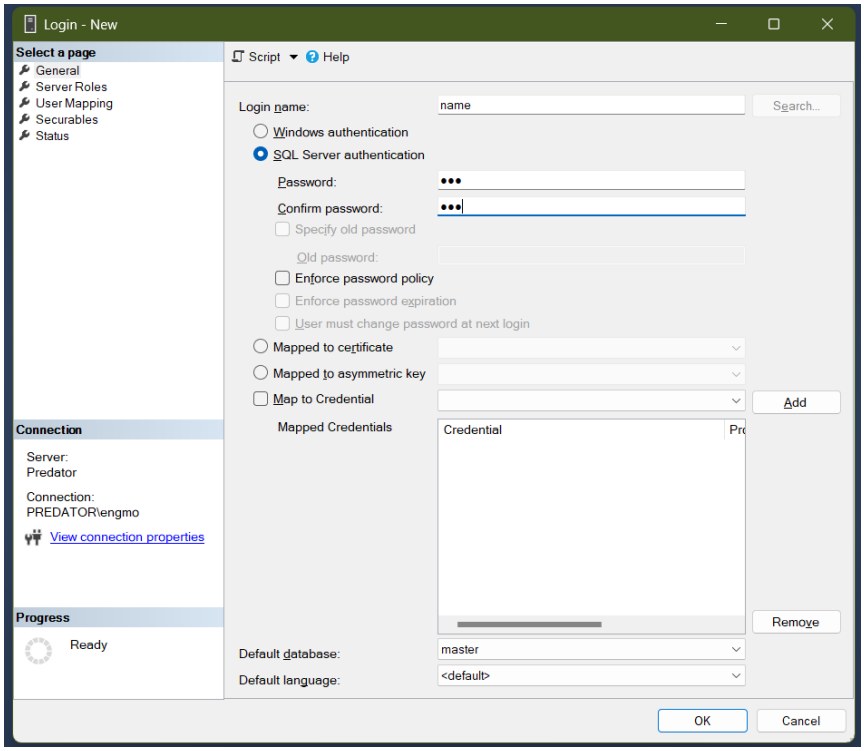


2) **Step 2:** restart the SQLServer Service in (task manager / services)

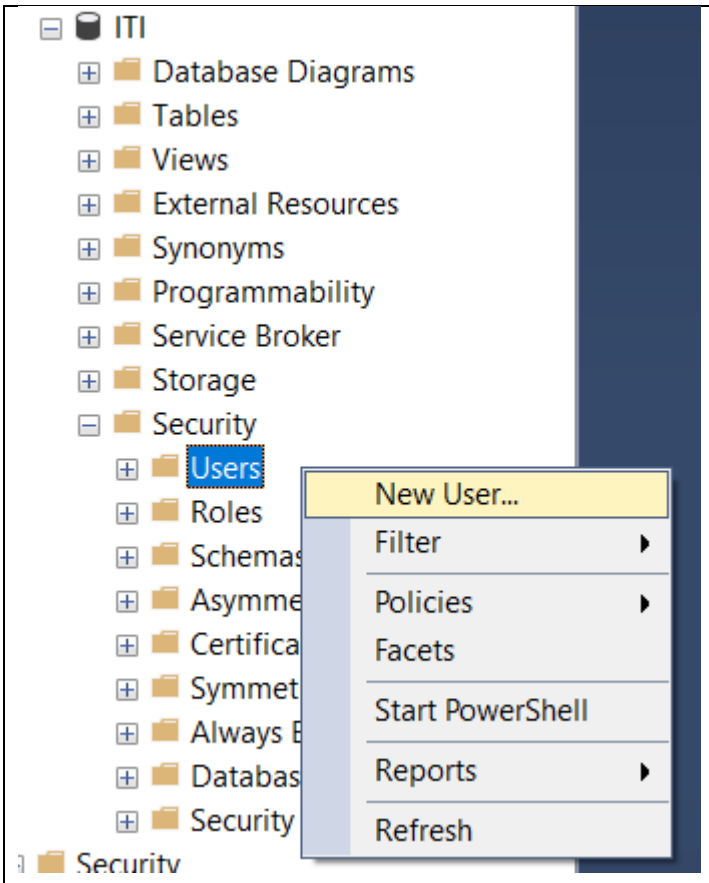


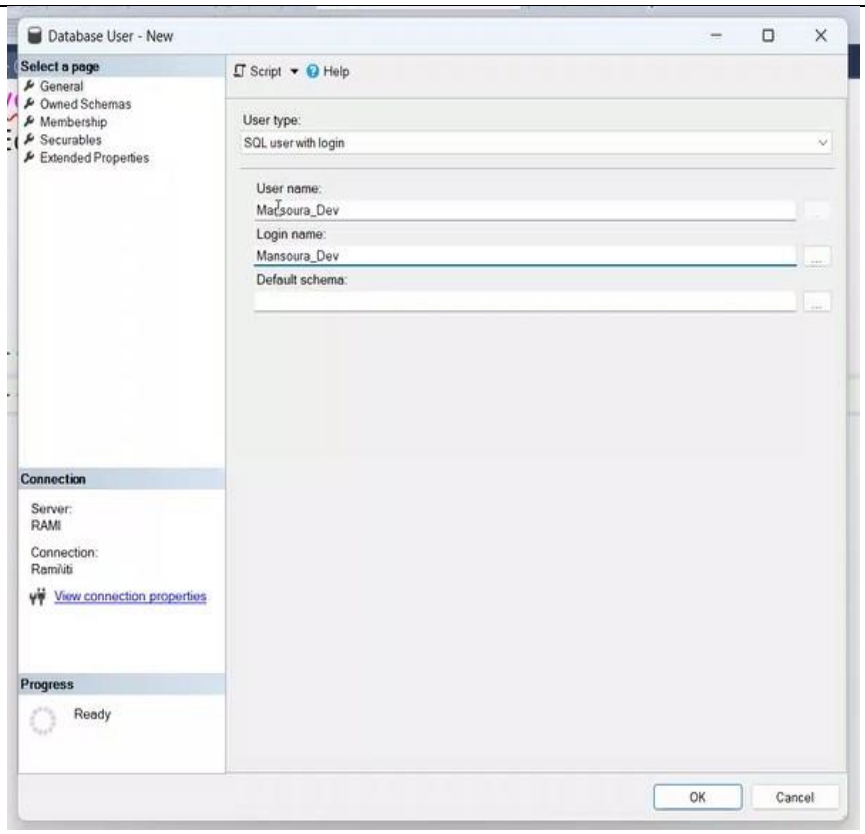
3) **Step 3:** create new login



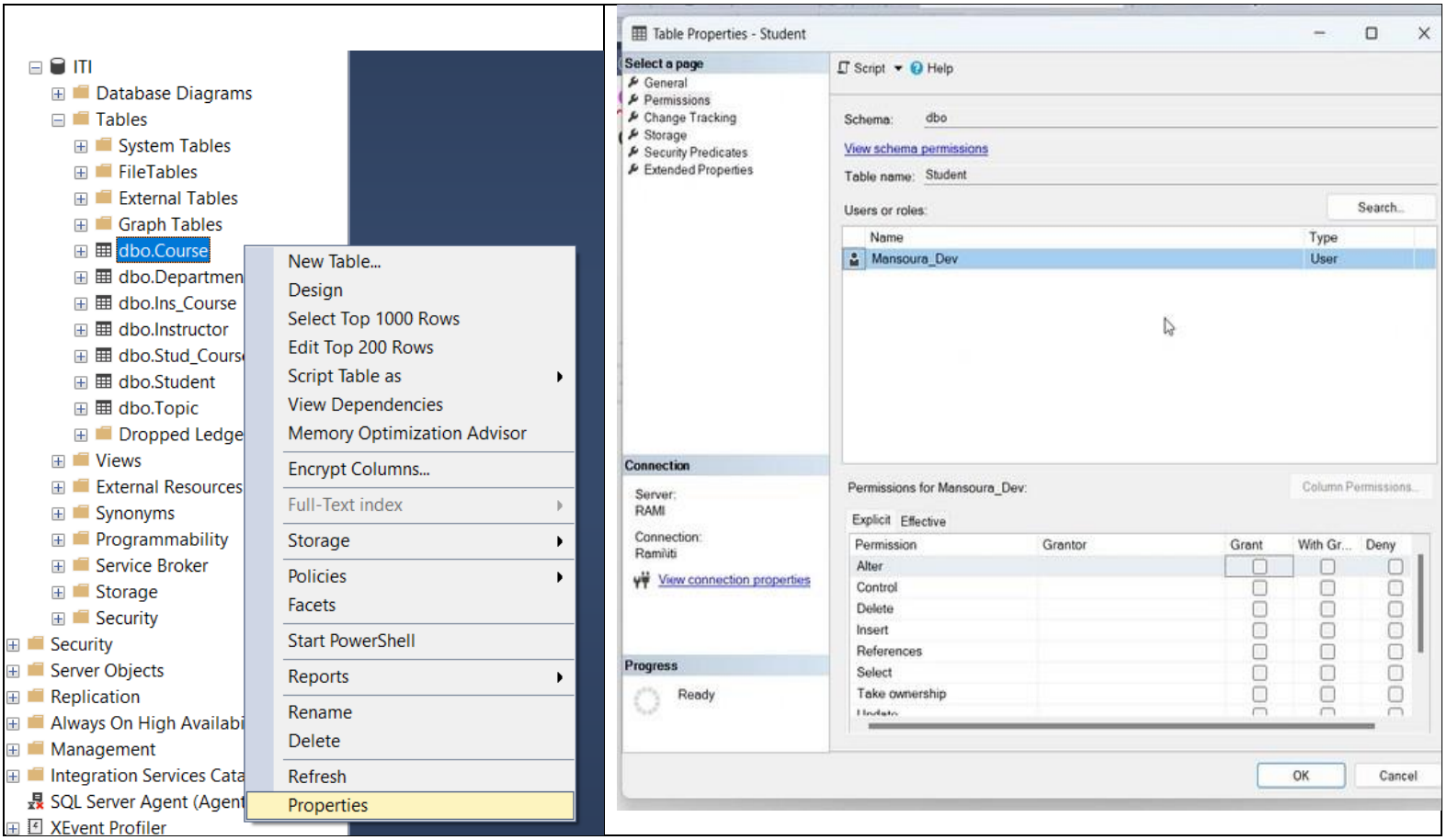


4) **Step 4:** the new user can login to the server but can't access DB I need to make him a user so he can edit DB.

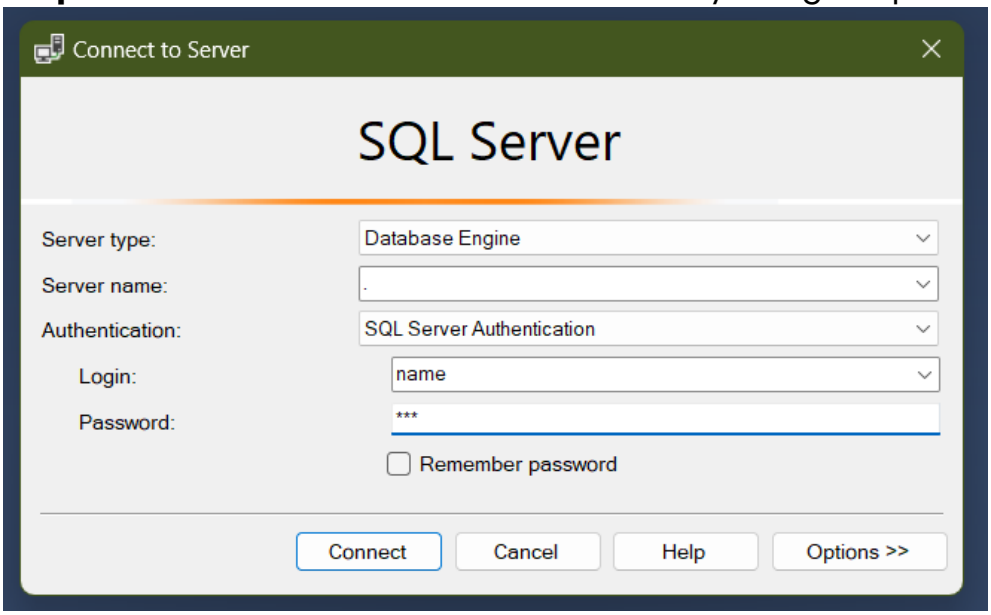




5) **Step 5:** add permissions to the new user



6) **Step 6:** now users can use the data base only using the permission you allow to them



49. Explain the difference between GUID vs normal ID vs unique identifier?

Characteristic	GUID	Normal ID	Unique identifier
Uniqueness	Extremely unique	Not as unique	Guaranteed to be unique
Size	128 bits	Typically, 32 bits	Varies
Performance	Slower	Faster	Varies
Usage	Unique identifiers for database records, files, and other objects	Primary keys and other unique identifiers in databases	General term for any identifier that is guaranteed to be unique

50. What is Object Full path?

Servername.Dbname.schemaName.Objectname
Example: [Rami].ITI.dbo.course
Note: default scheme is dbo

51. Explain Some of Transact SQL queries Types?

1- **Top():** It displays a specific number of rows from top.
2- **With ties**: It looks at last row in the top statement if it found a tie (repeat) for it then it will display all ties (repeats) for it.
3- **Ranking Functions:**
   - **Row_number():**will number the rows starting from 1 and increment by 1 for each row **it doesn't deal with ties results.**
   - **Dense_rank():** will number the rows starting from 1 and increment by 1 for each row **but it gives the tied result the same number.**
   - **Ntile(n):** it divides tables into equal groups if number of groups is not possible to divide equally then it gets groups as **close to equal as possible.**
   - **Rank() ** self-study**

4- **Select into (DDL):** It is a DDL query. It copies the select result and creates a new table and past the result in that new table.

5- **Insert into (DML):**
   - **Simple**
   - **Constructor**
   - **insert based on select.**
   - **Bulk insert** (delimited files only)

52. Some SQL queries about (top, withties, ranking functions, select into, insert into):

```
 ----------- Top ------------
select top(3)*
from student

---------- Top with ties ------------
select top(5) with ties *    -- selects the oldest 5 students with repetion
from student
order by st_age desc
-----------------GUID ---------------
select newid()

select top(3)*   --- using GUID to select random 3 students
from student
order by newid()

--------------RANKING FUNCTIONS-------------

------ Rownumber

Select *
From (select *,Row_number() over(order by st_age desc) as RN
        from student) as Newtable
where RN=1


------ DenseRank
Select *
From (select *,Dense_Rank() over(order by st_age desc) as DR
        from student) as NewTable
where DR=1


------ Rownumber + partition by
      ---  partition by  will divide the tables into groups by their department ids
         ---  Row_number()  will rank each of those groups by thier ages
         ---  where RN =1   will select the first rank for each group
Select *
From (select *,Row_number() over(partition by dept_id order by st_age desc) as RN
        from student) as Newtable
where RN=1

------ DenseRank + partition by
---  partition by  will divide the tables into groups by their department ids
         ---  Dense_Rank()  will rank each of those groups by thier ages + those who have the same age will
be given the same rank
         ---  where DR =1   will select the rank whose value is 1 for each group
Select *
From (select *,Dense_Rank() over(partition by dept_id order by st_age desc) as DR
        from student) as NewTable
where DR=1

------ Ntile()

select * ,Ntile(3) over(order by st_age desc) as G from student

-----------------Select Into--------

Select * into new_table   -- copies data+ metadata
from course

select st_id,st_fname into t3
from hr.student
where st_address='alex'
```

```sql
Select * into new_table   -- copies only metadata
from course
where 1 = 2   -- false condition

------------------Insert into ------

-- simple
insert into t3
values(7,'ali')

--Insert constructor
insert into t4
values (7700,'taher'), (43,'eman'), (66,'nada')

--based on select
insert into t5
select st_id,st_fname from hr.student
where st_address='mansoura'

-- Bulk insert   --> insert data from a file to a table
BULK INSERT tab5
FROM 'e:\Mydata.txt'

WITH(fieldterminator=',')
```

53. Using MSSQL Server Wizard to import / Export Database tables?

- **You can export your database to several types of files** (Oracle - Excel - Access - etc...)
- **You can import several types of files** (Oracle - Excel - Access - etc...) **into your database.**