



## Microsoft SQL Server

# Database – Core Concepts

- An organized collection of data stored and accessed electronically
- Small databases are stored on a file system, while large databases are hosted on a cluster or on the cloud
- How data is organized, depends on the **data model** being used
- Types of data models:
  - Flat
  - Hierarchical
  - Network
  - Relational
  - Object-Relational
  - etc., etc.

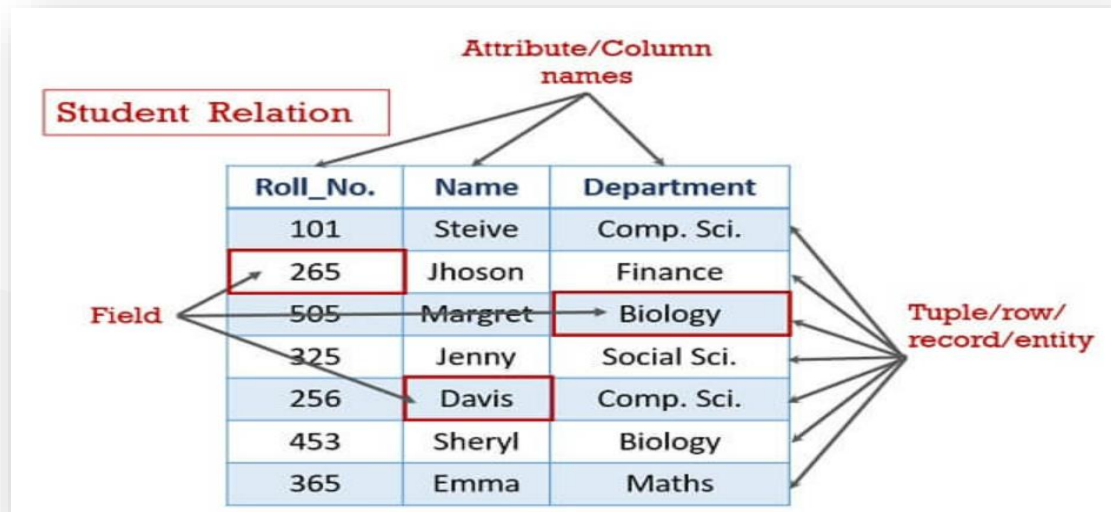
# DBMS and RDBMS

- A DBMS/RDBMS is a software which allows accessing and managing data stored in a non-relational and relational form respectively
- Differences:

DBMS	RDBMS
Data is stored as a file	Data is stored in the form of tables
Supports only a single user	Supports multiple users
Does not support client-server model	Supports client-server model
Lots of data redundancy	No data redundancy
No data integrity	Data Integrity is high
No normalization	Supports normalization
Does not support distributed DBs	Supports distributed DBs
Relationships not supported	Supports relationships
Lack of data security	Data security can be implemented at a very granular level
Individual access to data elements	Multiple data elements can be accessed easily using SQL

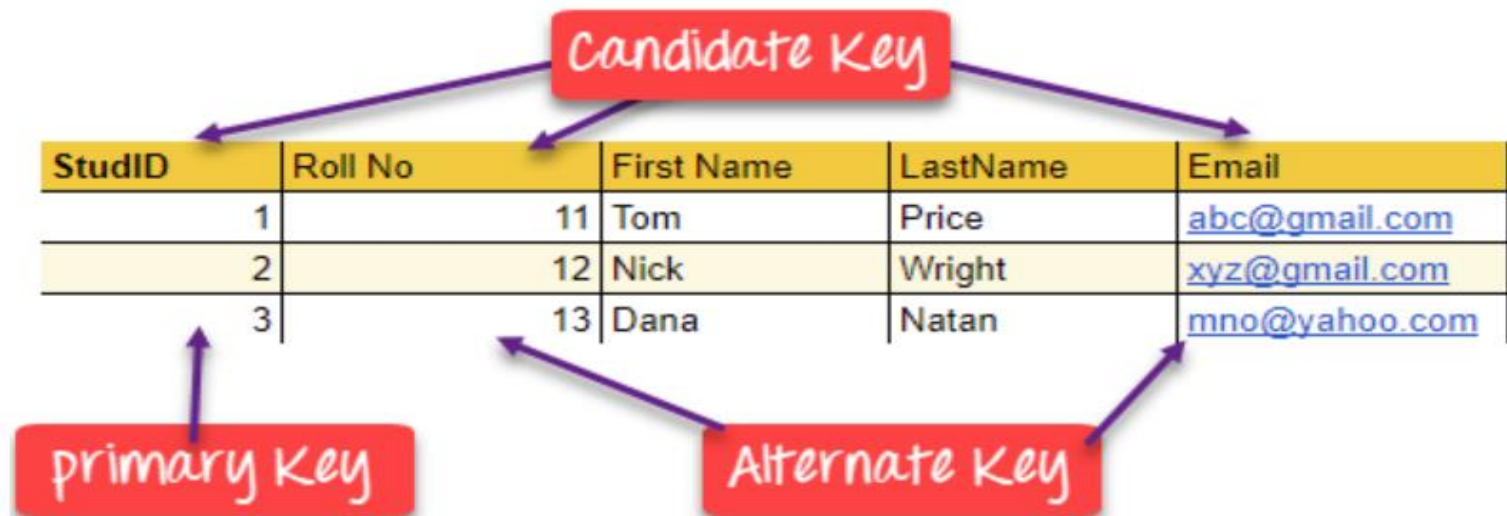
# Relational Model – Core Concepts

- Proposed by Dr. E.F. Codd at IBM in 1970
  - Codd's **12 rules** - designed to define what is required from a DBMS in order for it to be considered an RDBMS
    - [https://en.wikipedia.org/wiki/Codd%27s\\_12\\_rules](https://en.wikipedia.org/wiki/Codd%27s_12_rules)
- Organizes data into one or more tables with each table consisting of a set of rows and columns. Each row is identified uniquely using a **key**
  - Row → also known as a **tuple** or a **record**
  - Column → also known as an **attribute** or a **field**




# Relational Model – Keys - PK, AK, CK

- An attribute or set of attributes which helps to identify a row in a table
- Why keys?
  - Uniqueness of a row (*identity constraint*)
  - Establish relationship between tables (*integrity constraint*)



# Relational Model – Composite & Unique Key(s)

Composite Key



A diagram with a triangle pointing to the first two columns of the table below, indicating they form a composite key.

Order_ID	Product_ID	P_Name	Quantity
1153	8	abc	2
1155	9	xyz	6
1153	10	klm	1

Candidate\_Detail



Two arrows point from the text 'Unique Key' to the 'Aadhar\_no' and 'Other\_Id' columns, which are circled in red in the table below.

Candidate_no	Name	Aadhar_no	Other_Id
2018101	X	1432113211621	4361
2018211	Y	Null	8121
2018121	Z	842191428136	Null

# What is SQL?

- Acronym for **Structured Query Language**
- Developed at IBM & was initially known as **SEQUEL** (*Structured English Query Language*)
- A **declarative** language with procedural elements designed for managing data in an RDBMS
- Each RDBMS has it's own flavors/dialects of SQL
  - SQL Server – T-SQL
  - Oracle – PL/SQL
  - etc., etc.
- SQL statements/commands can be categorized as:
  - DQL, DDL, DCL, DML, TCL

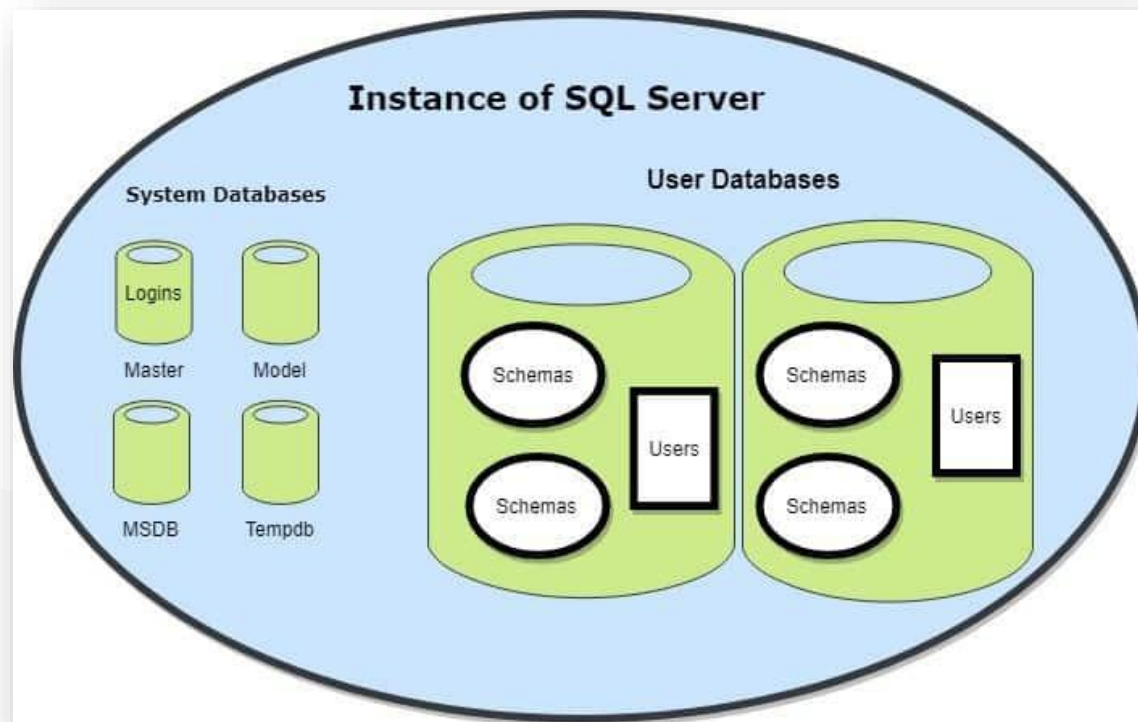
# Introducing MS SQL Server

- An RDBMS based on a **client-server** architecture, developed by Microsoft
- Available in the following editions:
  - Enterprise
    - High performance and speed
    - Suitable for large scale organizations
  - Standard
    - Basic data management features
    - Suitable for small organizations
  - Web
    - Low total-cost-of-ownership option for Web hosters
  - Developer
    - Includes all the functionality of Enterprise edition, but is licensed for use as a development and test system
    - Ideal choice for people who build and test applications
  - Express
    - Entry-level, free database
    - Ideal for learning and building desktop and small server data-driven applications



# SQL Server – Server

- An instance of the Database Engine is a copy of the **sqlservr.exe** executable that runs as an operating system service
- Each computer can run multiple instances of the Database Engine
- Applications connect to the instance in order to perform work in a database managed by the instance



# SQL Server – Client (SSMS)

- SQL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure
- Combines a broad group of graphical tools with many rich script editors to provide access to SQL Server for developers and database administrators

# SQL Server authentication

- Windows authentication
  - Default mode
  - Also known as **Integrated Security**
  - Specific Windows user and group accounts are trusted to log in to SQL Server
  - Windows users who have already been authenticated do not have to present additional credentials
  - When to use?
    - Application and DB are on the same computer
    - When using SQL Express
- Mixed Mode authentication
  - Supports authentication both by Windows and by SQL Server
  - User name and password pairs are maintained within SQL Server
  - When to use?
    - Users connect from different, non-trusted domains
    - Internet apps like ASP.NET, etc.

# SQL Server database

- Each instance of SQL Server can contain one or many databases
- Within a database, there are one or many object ownership groups called **schemas**
  - A schema is a **logical collection** of database objects
- Each database is a collection of objects like tables, views, stored procedures, functions, triggers, etc.
- Can be **user-defined** or **system**

# SQL Server system database(s)

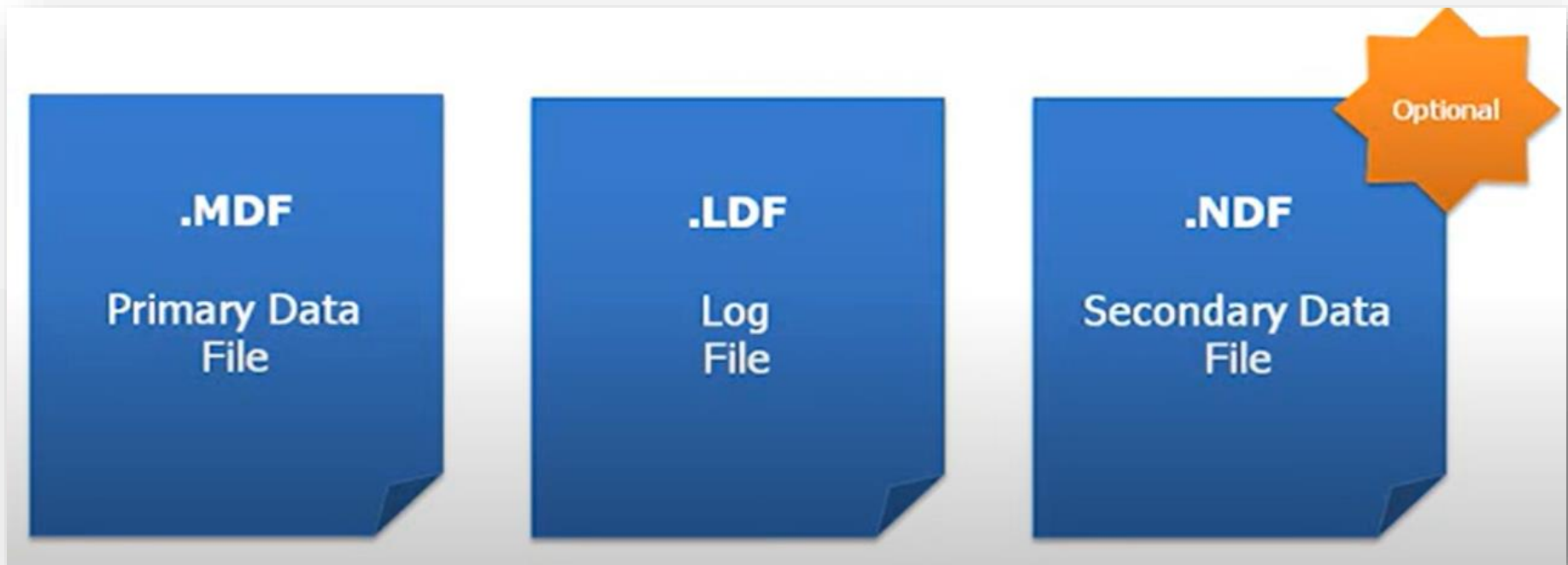
- Defined by Microsoft and are needed for SQL Server to operate
- Cannot be directly modified by users, instead SQL Server provides set of administrative tools for management
- **master** database
  - Includes instance-wide metadata such as logon accounts, endpoints, linked servers, and system configuration settings
  - Records the existence of all other databases and the location of those database files
  - SQL Server cannot start if the **master** database is unavailable
- **msdb** database
  - Used by SQL Server Agent for scheduling alerts and jobs
  - Contains tables which store details of backup and restore history

# SQL Server system database(s)

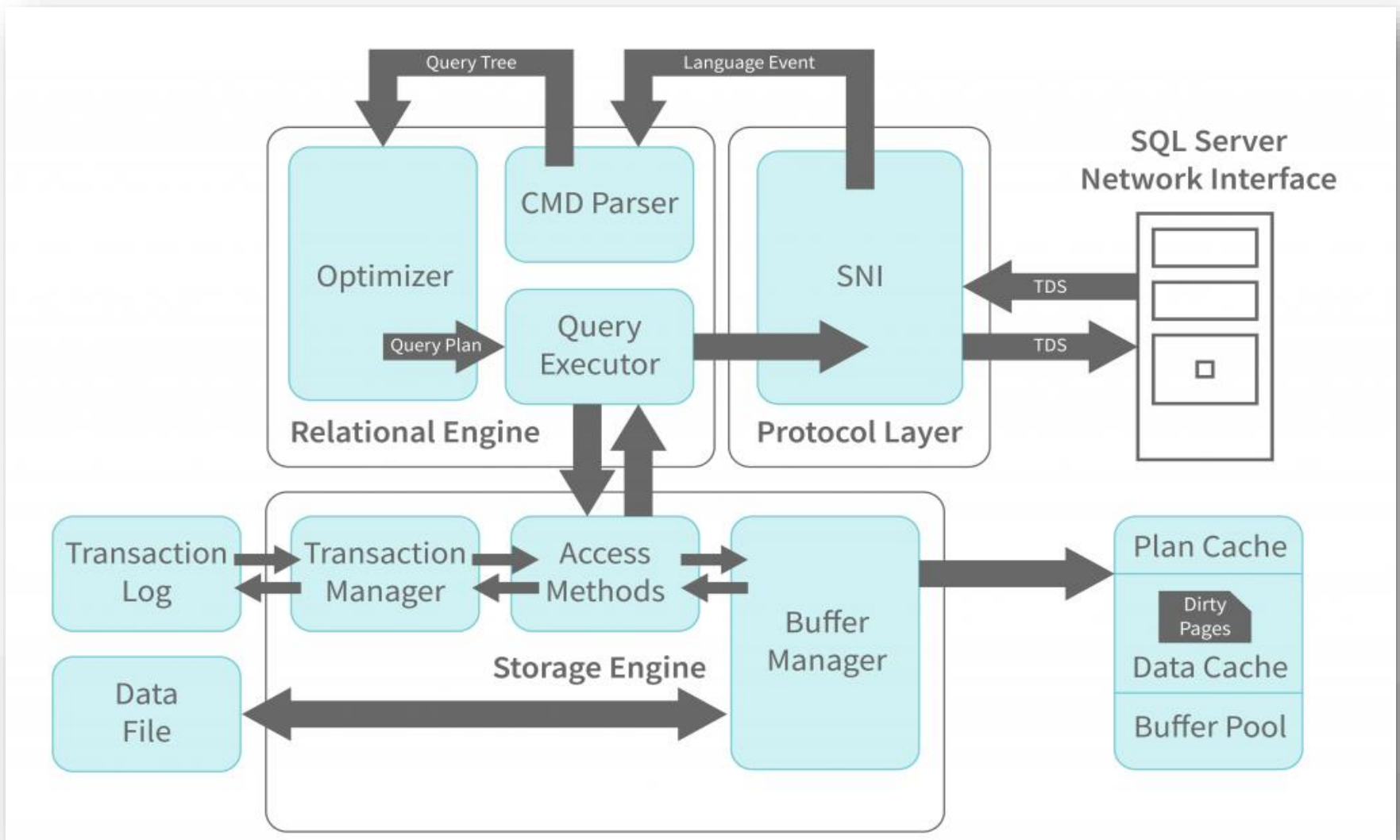
- **model** database
  - Used as the template for all databases created on an instance of SQL Server
  - Modifications made to this database, such as database size, recovery model, and other database options, are applied to any databases created afterward
- **resource** database
  - A **read-only** database that contains all the system objects that are included with SQL Server
  - Does not contain user data
  - Makes upgrading to a new version of SQL Server an easier and faster procedure
    - Because this database file contains all system objects, an upgrade is now accomplished simply by copying the single Resource database file to the local server
- **tempdb** database
  - Holds temporary **user objects** that are explicitly created and also **internal objects** that the database engine creates
  - Re-created every time SQL Server is started so that the system always starts with a clean copy of the database
  - ***No backup and restore operations are allowed***

# SQL Server Physical Storage

- At a minimum, every SQL Server database has two operating system files: **a data file and a log file**
- Data files contain data related to tables, indexes, stored procedures, and views, etc.
- Log files contain the information that is required to recover all transactions in the database



# SQL Server Architecture – The BIG PICTURE



<https://www.interviewbit.com/blog/sql-server-architecture/>



# SQL Server Architecture – The 3 Components

- **The Protocol Layer (SNI)**

- In charge of client and server communication
- This can be using
  - **Shared Memory protocol** → client & server on the same machine
  - **TCP/IP** → client & server are on remote machines
  - **Named Pipes** → client & server are on LAN
  - **TDS** → Used by all 3 protocols and is in charge of transferring of data packets

- **The Relational Engine**

- Also called the **Query Processor**
- In charge of determining how best a query can be executed
- Consists of
  - **CMD Parser** → checks the correctness of a query syntax
  - **Optimizer** → uses built-in algorithms to determine the best possible way to execute the query & creates an execution plan
  - **Query Executor** → executes the query based on the execution plan by fetching data from the storage engine

# SQL Server Architecture – The 3 Components

- **The Storage Engine**

- In charge of storing the data in a storage system and retrieving it when needed
- Consists of
  - Access method → determines whether a query comprises a SELECT or a non-SELECT statement & based on it passes it on to Buffer Manager or Transaction Manager
  - Buffer Manager → Interface to the Plan Cache and Data Cache
  - Transaction Manager → Manages transactions
  - Plan Cache → caches query execution plans
  - Data Cache → caches data

# Creating and dropping SQL Server Database

```
CREATE DATABASE freshers2022db
```

```
DROP DATABASE freshers2022db
```

# Creating and dropping SQL Server schema

```
USE freshers2022
```

```
CREATE SCHEMA training
```

```
DROP SCHEMA training
```

# SQL Server data types

- Each column, local variable, expression, and parameter has a related **data type**
  - A data type is an attribute that specifies the type of data that the object can hold
- Each column in a table must have a data type
- Broad Categories
  - NUMERICS (integers & floats)
    - tinyint
    - smallint
    - int
    - bigint
    - bit
    - smallmoney
    - money
    - decimal(p, s)
    - float(n)/real

# SQL Server data types

- Broad Categories
  - DATE & TIME
    - date
    - time
    - smalldatetime
    - datetime
    - datetime2
    - Datetimeoffset
  - CHARACTER STRINGS (fixed & variable length – non unicode)
    - char(n) → fixed length
    - varchar(n | max) → variable length
    - text → variable length
  - CHARACTER STRINGS (fixed & variable length – unicode)
    - nchar(n)
    - nvarchar(n)
    - ntext

# SQL Server data types

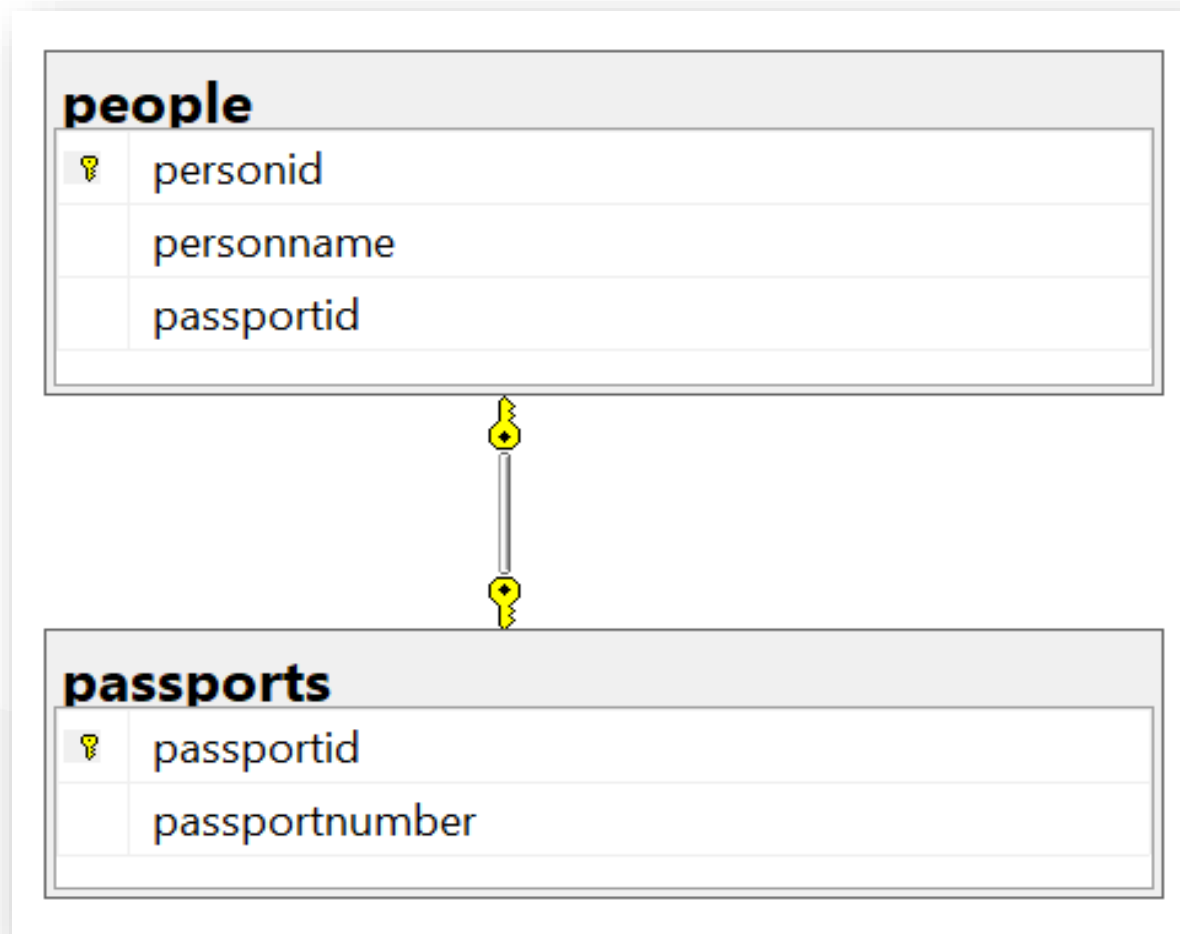
- Broad Categories
  - BINARY
    - binary(n)
    - varbinary(n | max)

# SQL Server table relationships

- A **relationship** determines how tables in a database are **related** to each other
  - It also presupposes that one of them has a **foreign key** that **references the primary key** of another table
- The relationship between tables is the **cardinality** which specifies how many entities in one table relate to how many entities in another table
- Types of relationships/cardinalities
  - One to One (1 : 1)
  - One to Many (1: N)
  - Many to One (N : 1)
  - Many to Many (M : N)

# SQL Server table relationships – 1:1

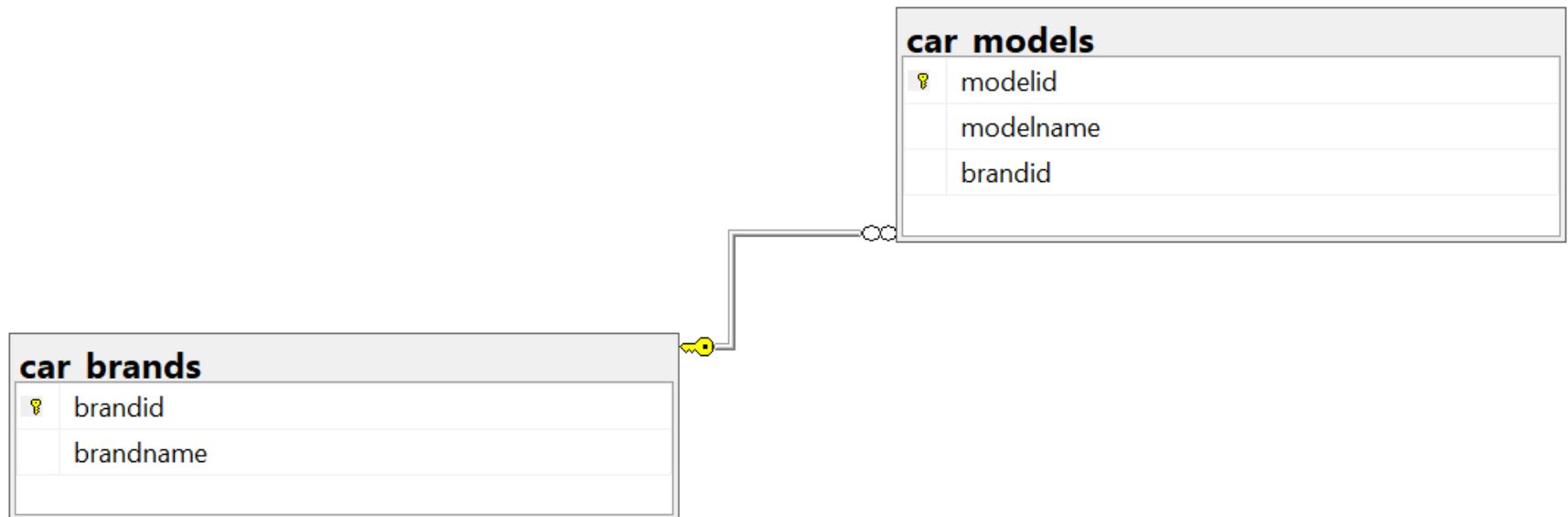
- Exists when **each record of one table** is related to **only one record of the other table**





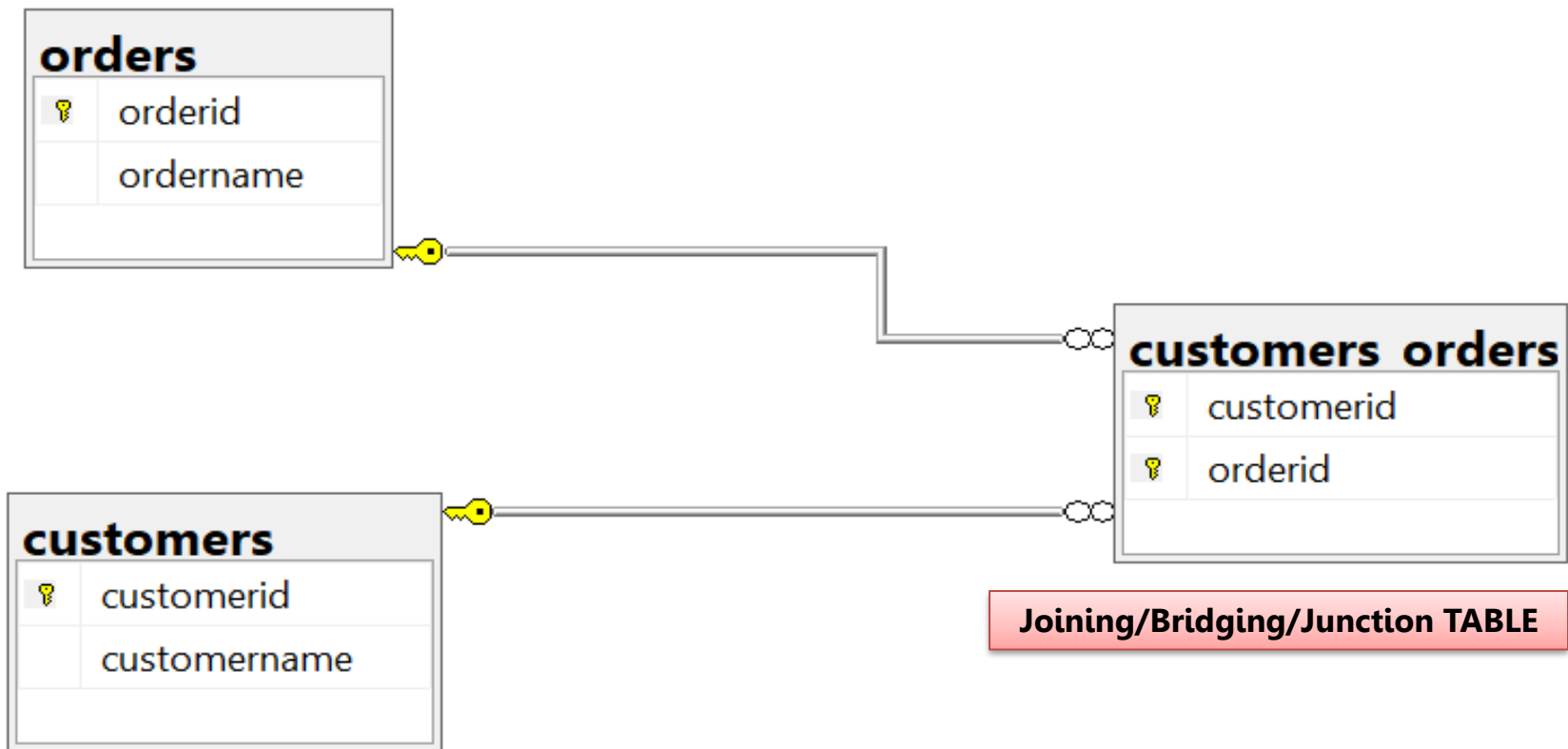
# SQL Server table relationships – 1:N

- Exists when **each record of one table** can be related to **one or more records of the other table**
  - Can **also be said as a many-to-one relationship** depending on the way it is viewed



# SQL Server table relationships – M:N

- Exists when **each record of the first table** can be related to **one or more records of the second table** and a **single record of the second table** can be related to **one or more records of the first table**



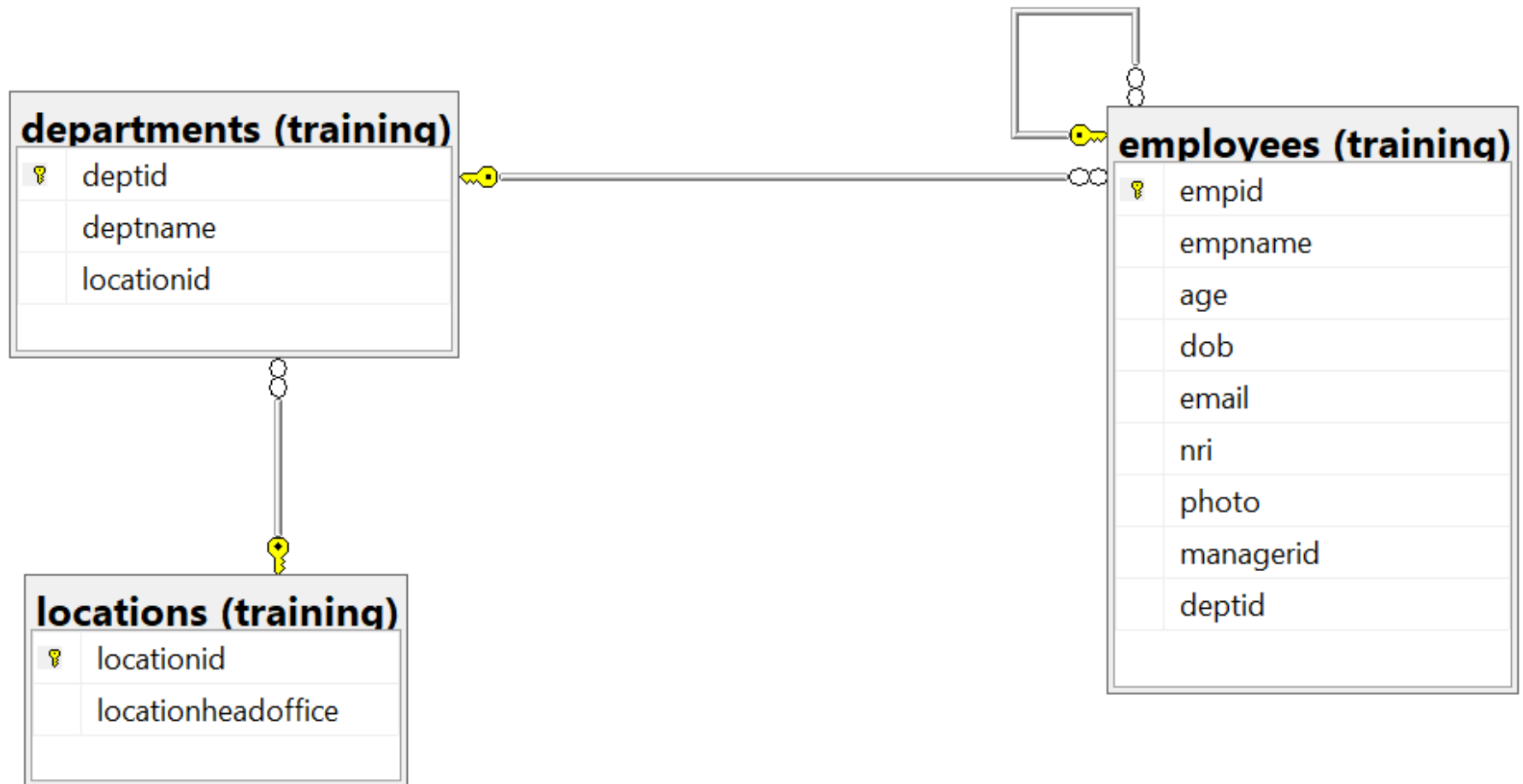
# SQL Server – DDL Statements

- Defines data structures
- Used to create, alter, or drop data structures in a database
- These statements include:
  - CREATE
  - ALTER
  - DROP
  - RENAME
  - TRUNCATE TABLE

# SQL Server – DDL Statements

- Defines data structures
- Used to create, alter, or drop data structures in a database
- These statements include:
  - CREATE
  - ALTER
  - DROP
  - RENAME
  - TRUNCATE TABLE

# Tables to be used



# Backup and Restore in SQL Server

- What is a backup?
  - A copy of data that can be used to restore and recover the data after a failure
- What is a restore?
  - A process that copies all the data from a specified SQL Server backup to a specified database, and then rolls forward all the transactions that are logged in the backup by applying changes to bring the data forward in time
- Why backup?
  - Recover data in case of...
    - Hardware failures (*damaged disk drive, server loss, etc.*)
    - Dropping a table by mistake
  - Routine administrative purposes, such as copying a database from one server

<https://docs.microsoft.com/en-us/sql/relational-databases/backup-restore/quickstart-backup-restore-database?view=sql-server-ver16>